ZETLab

# ZETLab studio.

# Руководство разработчика

# ООО «Электронные технологии и метрологические системы»

© ООО "ЭТМС" 1992-2024. Все права защищены

ЭТМС.05000-01 34 РО

# Содержание

## HELP ZETLab studio. Руководство разработчика.

**ADC информация и данные в adcinfo.h** ... **16**

**Подключение ZET017 U4/U8/T8 по Ethernet** ... **22**

**1.Введение** ... **23**

1.1.Режимы работы устройства ... 23

1.2.Порты TCP/IPv4 ... 24

**2.Формат пакетов** ... **24**

2.1.Команды ... 24

2.1.1. Команда 0x0000 GetInfo ... 25

2.1.2. Команда 0x0001 ReadDigitalPort ... 26

2.1.3. Команда 0x0002 WriteDigitalPort ... 27

2.1.4. Команда 0x0003 EnableDigitalPort ... 27

2.1.5. Команда 0x0012 PutInfo ... 28

2.1.6. Команда 0x0207 Restart ... 28

**3.Подключение к устройству** ... **28**

**4.Получение данных АЦП** ... **29**

**5.Работа генератора** ... **31**

**6.Пример программы** ... **33**

**О ZETLab** ... **39**

**1.Введение в ZETLab studio** ... **40**

1.1.Разработка законченной системы ... 41

1.2.Построение собственного виртуального прибора ... 42

1.3.Структура ZETLab studio ... 43

1.3.1. Структура задач измерений и испытаний ... 44

1.3.2. Накопление данных ... 46

1.3.3. Обработка и анализ данных ... 49

1.3.4. Представление результатов ... 51

**2.Введение в ПО ZETLab** ... **55**

2.1.О виртуальных приборах ... 55

**3.Установка и обновление ПО и драйверов** ... **58**

3.1.Требования к ПК ... 58

3.2.Проблемы с драйверами ... 59

3.2.1. в ОС Windows 7 ... 59

3.2.2. в ОС Windows 10 ... 60

3.3.Устранение возможных неполадок ПО ... 61

3.4.Установка ПО ZETLAB ... 66

3.5.Удаление ПО ZETLAB ... 76

3.6.Обновление ПО ZETLAB ... 79

3.7.Обновление драйверов ... 80

3.8.Проверка целостности установочного файла дистрибутивов ZETLAB ... 83

**4.Подключение приборов** ... **84**

4.1.Подключение по USB ... 84

4.2.Подключение по Ethernet ... 85

4.3.Подключение по Wi-Fi ... 91

*© ООО "ЭТМС" 1992-2024. Все права защищены*

**5.О работе с ПО ZETLAB ... 92**
5.1.Приступая к работе ... 93
5.2.Время усреднения данных и частота дискретизации сигналов ... 94
5.3.Элементы управления ... 99
5.3.1. Управление курсором и масштабирование графиков ... 99
5.3.2. Интегральный уровень сигнала ... 102
5.3.3. Перенос графической и численной информации в текстовые редакторы ... 102
5.3.4. Настройка внешнего вида графиков ... 104
**6.Панель управления ZETLAB ... 111**
6.1.Главное меню панели ZETLAB ... 111
6.2.Меню запуска программ ... 116
6.3.Многоэкранный интерфейс ... 118
**Доступ к драйверам ZET через библиотеку Zadc.dll ... 121**
**1.Полное описание функций Zadc.dll ... 121**
1.1.Работа с устройствами фирмы ZET ... 121
1.2.Подключение к драйверу и отключение ... 123
1.2.1. Подключение к драйверу ... 124
1.2.2. Отключение от драйвера ... 125
1.3.Сброс и инициализация ... 126
1.3.1. Сброс и останов сигнального процессора ... 126
1.3.2. Инициализация сигнального процессора ... 127
1.3.3. Получение серийного номера DSP ... 128
1.3.4. Получение номера ревизии микропрограммы DSP ... 129
1.3.5. Получение имени устройства ... 129
1.4.Сервисные функции ... 130
1.4.1. Опрос версии программ и драйвера ... 130
1.4.2. Получение типа интерфейса ... 132
1.4.3. Опрос аппаратных возможностей устройства ... 132
1.4.4. Прочитать всю информацию о модуле АЦП, ЦАП из драйвера ... 133
1.4.5. Прочитать всю информацию о модуле АЦП, ЦАП из DSP ... 133
1.4.6. Передать всю информацию в DSP ... 134
1.4.7. Запустить тест шлейфа для плат PnP ... 134
1.4.8. Чтение кода ошибки ... 135
1.4.9. Опрос изменения режима работы сигнального процессора ... 135
1.5.Установка режима работы сигнального процессора ... 136
1.5.1. Установка работы с модулем АЦП ... 136
1.5.2. Установка работы с модулем ЦАП ... 136
1.6.Опрос основных характеристик модулей АЦП и ЦАП ... 137
1.6.1. Опрос возможности работы сигнального процессора с модулем АЦП ... 137
1.6.2. Опрос возможности работы сигнального процессора с модулем ЦАП ... 138
1.6.3. Опрос максимального количества каналов модуля АЦП/ЦАП ... 138
1.6.4. Опрос веса младшего разряда АЦП/ЦАП ... 139
1.6.5. Опрос количества двоичных разрядов АЦП/ЦАП ... 140
1.6.6. Опрос размера каждого отсчета АЦП/ЦАП в 16-разрядных словах ... 140
1.7.Установка частоты дискретизации и режима синхронизации АЦП/ЦАП ... 141
1.7.1. Получение списка возможных частот дискретизации АЦП/ЦАП ... 141
1.7.2. Установка большей или меньшей частоты дискретизации АЦП/ЦАП ... 143

*© ООО "ЭТМС" 1992-2024. Все права защищены*

1.7.3. Опрос текущей частоты дискретизации АЦП/ЦАП ... 144
1.7.4. Установка частоты дискретизации АЦП/ЦАП ... 144
1.7.5. Опрос текущей опорной частоты АЦП/ЦАП ... 145
1.7.6. Установка значения внешней опорной частоты АЦП/ЦАП ... 145
1.7.7. Опрос режима работы от внешней опорной частоты ... 146
1.7.8. Установка режима работы от внешней опорной частоты АЦП ... 146
1.7.9. Опрос разрешения внешнего запуска накопления данных ... 147
1.7.10. Установка внешнего запуска накопления данных ... 148
1.7.11. Определить установленный коэффициент деления АЦП/ЦАП ... 148
1.7.12. Установить частоту преобразования АЦП/ЦАП ... 149
1.7.13. Прочитать статус синхронизации по внешней частоте АЦП/ЦАП ... 150
1.7.14. Включение/Отключение синхронизации по внешней частоте АЦП/ЦАП ... 150
1.7.15. Прочитать статус внешнего запуска АЦП/ЦАП ... 151
1.7.16. Включение/Отключение внешнего запуска АЦП/ЦАП ... 152
1.8.Управление каналами ввода (вывода) АЦП/ЦАП ... 152
1.8.1. Опрос количества включенных каналов АЦП/ЦАП ... 152
1.8.2. Опрос разрешения канала для ввода (вывода) АЦП/ЦАП ... 153
1.8.3. Включение канала для ввода (вывода) АЦП/ЦАП ... 154
1.8.4. Включение/выключение каналов с помощью битовой маски АЦП/ЦАП ... 154
1.8.5. Опрос типа канала ввода АЦП ... 155
1.8.6. Установка типа канала ввода АЦП ... 156
1.8.7. Чтение режима работы АЦП ... 156
1.8.8. Установить режим работы АЦП ... 157
1.9.Управление коэффициентами усиления АЦП ... 157
1.9.1. Получение списка возможных коэффициентов усиления АЦП ... 158
1.9.2. Установка большего или меньшего коэффициента усиления выбранного канала АЦП ... 158
1.9.3. Опрос коэффициента усиления выбранного канала АЦП ... 159
1.9.4. Установка коэффициента усиления выбранного канала АЦП ... 160
1.10.Управление коэффициентами усиления ПУ 8/10 ... 161
1.10.1. Получение списка возможных коэффициентов усиления предварительного усилителя ... 162
1.10.2. Установка большего или меньшего коэффициента усиления предварительного усилителя ... 162
1.10.3. Опрос коэффициента усиления предварительного усилителя выбранного канала ... 163
1.10.4. Установка коэффициента усиления предварительного усилителя выбранного канала ... 164
1.11.Управление коэффициентами ослабления аттенюатора ЦАП ... 165
1.11.1. Опрос поддерживается ли программный аттенюатор ... 165
1.11.2. Опрос коэффициента ослабления аттенюатора выбранного канала ... 165
1.11.3. Установка коэффициента ослабления аттенюатора выбранного канала ... 166
1.12.Управление процессом перекачки данных ... 167
1.12.1. Установка режима внешней подкачки данных или внутренней генерации сигналов ЦАП ... 168
1.12.2. Опрос максимального размера буфера в ЦАП в DSP ... 169
1.12.3. Опрос размера буфера в ЦАП в DSP ... 169
1.12.4. Установить размер буфера в ЦАП в DSP ... 170
1.12.5. Опрос размера буфера для перекачки данных АЦП/ЦАП ... 170

© ООО "ЭТМС" 1992-2024. Все права защищены

1.12.6. Опрос максимального размера буфера для перекачки данных АЦП/ЦАП ... 171
1.12.7. Установка размера буфера для перекачки данных АЦП/ЦАП ... 171
1.12.8. Установка размера буфера памяти накопления АЦП/ЦАП ... 173
1.12.9. Отображение буфера памяти накопления АЦП/ЦАП ... 174
1.12.10. Освобождение буфера памяти накопления АЦП/ЦАП ... 175
1.12.11. Установка циклического или одноразового накопления АЦП/ЦАП ... 176
1.12.12. Пересылка последних накопленных данных АЦП ... 176
1.12.13. Опрос указателя накопления в буфер данных ... 177
1.12.14. Опрос флага прерываний ... 178
1.12.15. Опрос состояния накопления данных АЦП/ЦАП ... 178
1.12.16. Старт накопления данных АЦП/ЦАП ... 179
1.12.17. Останов накопления данных АЦП/ЦАП ... 179
1.12.18. Определить установленный размер пакета данных DSP АЦП/ЦАП ... 180
1.12.19. Определить максимально возможный размер пакета данных DSP АЦП/ЦАП ... 181
1.12.20. Установить размер пакета данных DSP АЦП/ЦАП ... 181
1.12.21. Определить установленное количество пакетов за одно прерывание АЦП/ЦАП ... 182
1.12.22. Определить максимальное возможное количество пакетов за одно прерывание АЦП/ЦАП ... 182
1.12.23. Установить количество пакетов за одно прерывание АЦП/ЦАП ... 183
1.13.Управление модулем HCP ... 184
1.13.1. Опрос поддержки и подключения модуля HCP ... 184
1.13.2. Опрос режима работы заданного канала модуля HCP ... 184
1.13.3. Установка режима работы заданного канала модуля HCP ... 185
1.14.Управление цифровым портом (вход-выход) ... 186
1.14.1. Опрос маски выходов цифрового порта ... 186
1.14.2. Опрос количества линий цифрового порта ... 186
1.14.3. Установка маски выходов цифрового порта ... 187
1.14.4. Чтение данных с входов цифрового порта ... 187
1.14.5. Чтение данных, выдаваемых на выходы цифрового порта ... 188
1.14.6. Запись данных в цифровой порт ... 188
1.14.7. Установить вывод цифрового порта в «1», без изменения состояния других выводов ... 189
1.14.8. Установить маску выводов цифрового порта в «1», без изменения состояния других выводов ... 190
1.14.9. Установить вывод цифрового порта в «0», без изменения состояния других выводов ... 190
1.14.10. Установить маску выводов цифрового порта в «0», без изменения состояния других выводов ... 191
1.14.11. Опрос режим потокового ввода-вывода ... 191
1.14.12. Переключить режим потокового ввода-вывода ... 192
1.14.13. Установить линию цифрового порта на выход ... 193
1.14.14. Установить маску линий цифрового порта на выход ... 193
1.14.15. Установить линию цифрового порта на вход ... 194
1.14.16. Установить маску линий цифрового порта на вход ... 194
1.15.Управление устройством через Ethernet ... 195
1.15.1. Опрос IP-адреса по Ethernet ... 195
1.15.2. Подключиться с устройством по Ethernet ... 195
1.15.3. Отключиться от устройства по Ethernet ... 196
1.15.4. Опрос на установление соединения с устройством через Ethernet ... 197

*© ООО "ЭТМС" 1992-2024. Все права защищены*

1.15.5. Опрос на установление соединения с устройством через Ethernet с опросом IP-адреса ... 197
1.16.Управление настройками TCP/IP ... 198
1.16.1. Прочитать настройки Ethernet устройства для TCP/IP из ПЗУ ... 198
1.16.2. Сохранить настройки Ethernet устройства для TCP/IP в ПЗУ ... 199
1.17.Управление настройками PTP ... 200
1.17.1. Прочитать настройки PTP из ПЗУ ... 200
1.17.2. Сохранить настройки PTP в ПЗУ ... 201
**2.Краткое описание zadc.dll ... 202**
**Unit-2. Предназначен для связи с программами ... 221**
**1.Установка компонента ... 222**
**2.Работа с программами ... 222**
2.1.Результаты ... 223
2.2.Параметры ... 224
2.3.Проверки ... 225
2.4.Другие функции ... 226
2.5.Номера данных многоканальных программ ... 226
2.6.Дополнительные разделы ресурсов програмы-сервера ... 226
2.7.Замечания ... 227
2.7.1. Таблица 1. Функции UnitCln ... 227
2.7.2. Таблица 2. Функции UnitSrv ... 228
2.7.3. Таблица 3. Функции структур типа Unit2DataStruct ... 229
**3.Описание свойств, методов и событий Unit-2.1 в idl ... 230**
3.1.Функции структуры Unit2DataStruct ... 230
3.2.Серверная часть ... 231
3.3.Клиентская часть ... 232
**4.Переход с UNIT на UNIT-2 ... 233**
4.1.Серверная часть ... 233
4.2.Клиентская часть ... 234
**Unit.ocx Предназначен для связи с программами ZETLab ... 235**
**1.Установка компонента ... 235**
**2.Изменения в zetlab studio на 30.06.2015 ... 237**
**3.Описание методов и событий в idl ... 239**
3.1.Серверная часть ... 239
3.1.1. Методы ... 240
3.1.2. События ... 242
3.2.Клиентская часть ... 243
3.2.1. Методы ... 243
3.2.2. События ... 247
**4.Перечень общих параметров многоканальных/многоконтейнерных программ ... 248**
**5.Установка параметров программ ZETLab ... 249**
5.1.меню Анализ сигналов ... 249
5.1.1. Узкополосный спектр (spectr.exe) ... 249
5.1.2. Долеоктавный спектр (dspectr.exe) ... 255
5.1.3. Взаимный узкополосный спектр (vspectr.exe) ... 259
5.1.4. Взаимный долеоктавный спектр (dvspectr.exe) ... 315
5.1.5. Взаимный корреляционный анализ (corr.exe) ... 316
5.1.6. Анализ нелинейных искажений (harmdist.exe) ... 320
5.1.7. Синхронное накопление (PrdkAnaliz.exe) ... 321
5.1.8. Модальный анализ (PrqsAnaliz.exe) ... 323
5.1.9. Гистограмма (ZETHistograph.exe) ... 325
5.1.10. Детектор STA\LTA (STA_LTA.exe) ... 326
5.1.11. Вейвлет (FDWT.exe) ... 327

*© ООО "ЭТМС" 1992-2024. Все права защищены*

5.2.меню Измерение ... 328
5.2.1. Вольтметр переменного тока (VoltMeter.exe) ... 328
5.2.2. Вольтметр постоянного тока (VoltMeterDC.exe) ... 329
5.2.3. Селективный вольтметр (VoltMeterSel.exe) ... 331
5.2.4. Частотомер (FreqMeter.exe) ... 331
5.2.5. Фазометр (PhaseMeter.exe) ... 331
5.2.6. Тахометр (TahoMeter.exe) ... 332
5.2.7. Торсиограф (Torsiograph.exe) ... 333
5.2.8. Энкодер (Encoder.exe) ... 334
5.2.9. Термометр ТС (ThermoMeter.exe) ... 335
5.2.10. Термометр ТП (ThermoPara.exe) ... 335
5.2.11. Тензометр (TenzoMeter.exe) ... 336
5.2.12. Виброметр (VibroMeter.exe) ... 337
5.2.13. Мультиметр Agilent_HP34401A (Agilent_HP34401A.exe) ... 340
5.2.14. Блок питания LPS305 (LPS305.exe) ... 340
5.2.15. Блок питания PPE3323 (PPE3323.exe) ... 341
5.2.16. Блок питания PSM2010 (PSM2010.exe) ... 342
5.3.меню Отображение ... 343
5.3.1. Многоканальный осциллограф (OscGraph.exe) ... 344
5.3.2. XYZ-осциллограф (XYOscGraph.exe) ... 346
5.3.3. XY-плоттер (XYPlotter.exe) ... 346
5.4.меню Генераторы ... 348
5.4.1. Генератор сигналов (DAC_OCX.exe) ... 348
5.4.2. Синхронный генератор (SynchroChanDac.exe) ... 356
5.5.меню Регистрация ... 359
5.5.1. Запись сигналов (SignalWriter.exe) ... 359
5.5.2. Воспроизведение сигналов (reader.exe) ... 360
5.5.3. Многоканальный самописец (multiSWvm.exe) ... 360
5.6.меню Автоматизация ... 362
5.6.1. Регулятор (Regulator.exe) ... 362
5.6.2. Арифмометр (ArithmoMeter.exe) ... 364
5.6.3. Адаптивный фильтр 50 Гц (filtr50.exe) ... 364
5.6.4. Фильтрация сигналов (filtrdiff.exe) ... 365
5.6.5. Формула (ZETFormula.exe) ... 367
5.6.6. Синхронизация по GPS (Synchronization.exe) ... 367
5.6.7. Управление реле (RelayCommutation.exe) ... 368
5.6.8. Управление релейными ключами (PortCommutation.exe) ... 369
5.7.меню Сетевые программы ... 369
5.7.1. Подключение устройств по Ethernet (NetWizard.exe) ... 369
5.7.2. Подключение устройств по BlueTooth (BthWizard.exe) ... 370
**6.Установка параметров программ ZETScope ... 370**
**7.Формат посылаемых данных ZETLab ... 374**
7.1.меню Анализ сигналов ... 375
7.1.1. Узкополосный спектр (spectr.exe) ... 375
7.1.2. Долеоктавный спектр (dspectr.exe) ... 376
7.1.3. Взаимный узкополосный спектр (vspectr.exe) ... 377
7.1.4. Взаимный долеоктавный спектр (dvspectr.exe) ... 378
7.1.5. Взаимный корреляционный анализ (corr.exe) ... 379
7.1.6. Анализ нелинейных искажений (harmdist.exe) ... 379
7.1.7. Синхронное накопление (PrdkAnaliz.exe) ... 379
7.1.8. Модальный анализ (PrqsAnaliz.exe) ... 380
7.1.9. Гистограмма (ZETHistograph.exe) ... 381
7.1.10. Детектор STA\LTA (STA_LTA.exe) ... 381
7.2.меню Измерение ... 381

*© ООО "ЭТМС" 1992-2024. Все права защищены*

7.2.1. Вольтметр переменного тока (VoltMeter.exe) ... 381
7.2.2. Вольтметр постоянного тока (VoltMeterDC.exe) ... 382
7.2.3. Селективный вольтметр (VoltMeterSel.exe) ... 382
7.2.4. Частотомер (FreqMeter.exe) ... 382
7.2.5. Фазометр (PhaseMeter.exe) ... 382
7.2.6. Тахометр (TahoMeter.exe) ... 383
7.2.7. Торсиограф (Torsiograph.exe) ... 383
7.2.8. Энкодер (Encoder.exe) ... 383
7.2.9. Термометр ТС (ThermoMeter.exe) ... 383
7.2.10. Термометр ТП (ThermoPara.exe) ... 383
7.2.11. Тензометр (TenzoMeter.exe) ... 383
7.2.12. Виброметр (VibroMeter.exe) ... 384
7.2.13. Мультиметр Agilent_HP34401A (Agilent_HP34401A.exe) ... 384
7.2.14. Блок питания LPS305 (LPS305.exe) ... 384
7.2.15. Блок питания PPE3323 (PPE3323.exe) ... 385
7.2.16. Блок питания PSM2010 (PSM2010.exe) ... 386
7.3.меню Отображение ... 387
7.3.1. Многоканальный осциллограф (OscGraph.exe) ... 387
7.3.2. XYZ-осциллограф (XYOscGraph.exe) ... 387
7.4.меню Генераторы ... 388
7.4.1. Генератор сигналов (DAC_OCX.exe) ... 388
7.4.2. Синхронный генератор (SynchroChanDac.exe) ... 388
7.5.меню Регистрация ... 388
7.5.1. Запись сигналов (SignalWriter.exe) ... 388
7.5.2. Воспроизведение сигналов (reader.exe) ... 389
7.5.3. Многоканальный самописец (multiSWvm.exe) ... 389
7.6.меню Автоматизация ... 389
7.6.1. Регулятор (Regulator.exe) ... 389
7.6.2. Арифмометр (ArithmoMeter.exe) ... 389
7.6.3. Адаптивный фильтр 50 Гц (filtr50.exe) ... 390
7.6.4. Фильтрация сигналов (filtrdiff.exe) ... 390
7.6.5. Формула (ZETFormula.exe) ... 390
7.6.6. Синхронизация по GPS (Synchronization.exe) ... 390
7.6.7. Управление реле (RelayCommutation.exe) ... 390
7.6.8. Управление релейными ключами (PortCommutation.exe) ... 391
7.7.меню Сетевые программы ... 391
7.7.1. Подключение устройств по Ethernet (NetWizard.exe) ... 391
7.7.2. Подключение устройств по BlueTooth (BthWizard.exe) ... 391
**8.Формат посылаемых данных ZETScope ... 391**
**9.Перечень функций, используемые Unit.ocx в базовом классе ... 392**
9.1.Серверная часть ... 392
9.1.1. Методы ... 393
9.1.2. События ... 395
9.2.Клиентская часть ... 396
9.2.1. Методы ... 396
9.2.2. События ... 401
**ZETPath.ocx Позволяет работать со стандартными каталогами ZETLab ... 402**
**1.Установка компонента ... 402**
**2.Описание свойств, методов и событий в idl ... 404**
**3.Перечень функций, используемые ZETPath.ocx в базовом классе ... 405**
**SRV.ocx для работы с ZETLab ... 408**
**1.Установка компонента SRV.ocx ... 408**

© ООО "ЭТМС" 1992-2024. Все права защищены

**2.Описание методов и событий в idl ... 411**
2.1.Методы в SRV.ocx ... 411
2.1.1. Подключение и отключение ... 411
2.1.2. Опрос параметров каналов системы ... 415
2.1.3. Опрос физических параметров каналов ... 422
2.1.4. Управление каналами АЦП ... 432
2.1.5. Управление каналами ЦАП ... 437
2.1.6. Управление виртуальными каналами ... 440
2.1.7. Работа с цифровым портом ... 449
2.1.8. Работа с ШИМ ... 456
2.1.9. Работа с ICP ... 461
2.1.10. Работа с GPS ... 463
2.1.11. Прием и передача данных ... 465
2.1.12. Вспомогательные информационные функции ... 473
2.2.События в SRV.ocx ... 491
**3.Перечень функций, используемые SRV.ocx в базовом классе ... 492**
**4.Сообщения программы из Журнала ошибок и ZETServer ... 505**
**ZetWindowsLog.ocx Позволяет заполнять стандартный журнал ошибок ... 507**
**1.Установка компонента ... 507**
**2.Описание свойств ... 509**
**ListMultiChannels.ocx предназначен для списка с использованием группового имени и фильтра каналов ... 511**
**1.Описание свойств ... 512**
**ZETAnpRecorder.ocx позволяет сохранять результаты в бинарном виде ... 515**
**1.Установка компонента ... 515**
**2.Описание свойств ... 517**
**ZETChanSelector.ocx предназначен для выбора каналов ... 521**
**1.Установка компонента ZETChanSelector.ocx ... 521**
**2.Интерфейс компонента ZETChanSelector.ocx ... 522**
**3.Полное описание свойств, методов и событий ZETChanSelector.ocx ... 523**
3.1.Свойства ZETChanSelector.ocx ... 523
3.1.1. Цвета ... 524
3.1.2. Отображение границ ... 524
3.1.3. Параметры ... 524
3.2.Методы ZETChanSelector.ocx ... 525
3.3.События от компоненты ZETChanSelector.ocx ... 525
**4.Краткий перечень функций, используемые ZETChanSelector.ocx ... 525**
**GdiPlusComboBox.ocx предназначен для комбинированного списка ... 530**
**1.Установка компонента GdiPlusComboBox.ocx ... 530**
**2.Интерфейс компонента GdiPlusComboBox.ocx ... 531**
**3.Полное описание свойств, методов и событий GdiPlusComboBox.ocx ... 532**
3.1.Свойства GdiPlusComboBox.ocx ... 532
3.1.1. Цвета ... 533
3.1.2. Отображение рамки ... 533
3.1.3. Параметры ... 533
3.2.Методы GdiPlusComboBox.ocx ... 534
3.3.События от компоненты GdiPlusComboBox.ocx ... 534
**4.Краткий перечень функций, используемые GdiPlusComboBox.ocx ... 534**

*© ООО "ЭТМС" 1992-2024. Все права защищены*

**ZetMultiLevelIndicator.ocx используется для многоканального индикатора уровня ... 538**
**1.Полное описание свойств, методов и событий ZetMultiLevelIndicator.ocx ... 538**
1.1.Методы ZetMultiLevelIndicator.ocx ... 538
**2.Краткий перечень функций, используемые ZetMultiLevelIndicator.ocx ... 539**
**Описание программных модулей индикаторов ... 541**
**1.zet_boolean.ocx представляет собой лампочку-индикатор ... 541**
**2.zet_horizontal_slider.ocx, zet_vertical_slider.ocx линейные аналоговые регуляторы шкалы ... 542**
**3.zet_horizontal_level, zet_vertical_level линейные аналоговые индикаторы уровня ... 544**
**4.ZETTaho.ocx предназначен для измерения частоты вращения ... 547**
4.1.Установка компонента ZETTaho.ocx ... 547
4.2.Интерфейс компонента ZETTaho.ocx ... 548
4.3.Полное описание свойств, методов и событий ZETTaho.ocx ... 549
4.3.1. Свойства ZETTaho.ocx ... 549
4.3.2. Методы ZETTaho.ocx ... 549
4.4.Краткий перечень функций, используемые ZETTaho.ocx ... 550
**5.GdiPlusHorScale.ocx предназначен для отображения интегральных уровней сигналов ... 551**
5.1.Установка компонента GdiPlusHorScale.ocx ... 551
5.2.Интерфейс компонента GdiPlusHorScale.ocx ... 552
5.3.Полное описание свойств, методов и событий GdiPlusHorScale.ocx ... 553
5.3.1. Свойства GdiPlusHorScale ... 553
5.3.1.1. Цвета ... 553
5.3.1.2. Отображение рамки ... 553
5.3.1.3. Ориентация компонента ... 554
5.3.1.4. Параметры ... 554
5.3.2. Методы GdiPlusHorScale.ocx ... 554
5.4.Краткий перечень функций, используемые GdiPlusHorScale.ocx ... 555
**6.GdiPlusTextDisp.ocx предназначен для отображения текстово-численной информации ... 556**
6.1.Установка компонента GdiPlusTextDisp.ocx ... 556
6.2.Интерфейс компонента GdiPlusTextDisp.ocx ... 557
6.3.Полное описание свойств, методов и событий GdiPlusTextDisp.ocx ... 558
6.3.1. Свойства GdiPlusTextDisp.ocx ... 558
6.3.1.1. Цвета ... 558
6.3.1.2. Отображение рамки ... 559
6.3.1.3. Параметры ... 559
6.3.2. Методы GdiPlusTextDisp.ocx ... 560
6.3.3. События от компоненты GdiPlusTextDisp.ocx ... 560
6.4.Краткий перечень функций, используемые GdiPlusTextDisp.ocx ... 560
**7.ZIndicator.ocx предназначен для вывода целых и дробных десятичных числовых величин ... 562**
7.1.Установка компонента ZIndicator.ocx ... 562
7.2.Интерфейс компонента ZIndicator.ocx ... 563
7.3.Полное описание свойств, методов и событий ZIndicator.ocx ... 564
7.3.1. Свойства ZIndicator.ocx ... 565
7.3.1.1. Цвета ... 565
7.3.1.2. Отображение рамки ... 565
7.3.1.3. Параметры шрифта ... 565
7.3.1.4. Параметры управления значением индикатора ... 566
7.3.2. Методы ZIndicator.ocx ... 566
7.3.3. События от компоненты ZIndicator.ocx ... 566
7.4.Краткий перечень функций, используемые ZIndicator.ocx ... 567
**8.TextDisp.ocx предназначен для отображения текстово-численной информации (уст.) ... 570**
8.1.Установка компонента TextDisp.ocx ... 570

*© ООО "ЭТМС" 1992-2024. Все права защищены*

8.2.Интерфейс компонента TextDisp.ocx ... 571
8.3.Полное описание свойств, методов и событий TextDisp.ocx ... 572
8.3.1. Свойства GdiPlusTextDisp.ocx ... 572
8.3.1.1. Цвета ... 573
8.3.1.2. Отображение рамки ... 573
8.3.1.3. Параметры ... 573
8.3.2. Методы TextDisp.ocx ... 573
8.4.Краткий перечень функций, используемые TextDisp.ocx ... 574
**9.ColScale.ocx задает соответствие цвета графика уровню сигнала при отображении спектрограмм**
9.1.Установка компонента ColScale.ocx ... 576
9.2.Интерфейс компонента ColScale.ocx ... 577
9.3.Полное описание свойств, методов и событий ColScale.ocx ... 578
9.3.1. Свойства ColScale.ocx ... 578
9.3.1.1. Цвета ... 578
9.3.1.2. Параметры шкалы ... 579
9.3.1.3. Параметры отображения ... 579
9.3.2. Методы ColScale.ocx ... 579
9.3.3. События от компоненты ColScale.ocx ... 580
9.4.Краткий перечень функций, используемые ColScale.ocx ... 580
**10.Horizontalslider.ocx для выбора значения посредством перемещения указателя на шкале ... 582**
10.1.Установка компонента Horizontalslider.ocx ... 582
10.2.Интерфейс компонента Horizontalslider.ocx ... 583
10.3.Полное описание свойств, методов и событий Horizontalslider.ocx ... 585
10.3.1. Свойства Horizontalslider.ocx ... 585
10.3.1.1. Цвета ... 585
10.3.1.2. Параметры шкалы ... 586
10.3.1.3. Параметры отображения ... 587
10.3.2. Методы Horizontalslider.ocx ... 587
10.3.3. События от компоненты Horizontalslider.ocx ... 587
10.4.Краткий перечень функций, используемые Horizontalslider.ocx ... 587
**Описание программных модулей графиков ... 592**
**1.Grid.ocx предназначена для графического отображения данных ... 592**
1.1.Установка компонента Grid.ocx ... 592
1.2.Назначение Grid.ocx ... 593
1.3.Полное описание свойств, методов и событий Grid.ocx ... 594
1.3.1. Свойства Grid.ocx ... 594
1.3.1.1. Цвета ... 594
1.3.1.2. Параметры графиков ... 595
1.3.1.3. Представление графиков ... 596
1.3.1.4. Параметры шкалы и подписи осей ... 598
1.3.1.5. Параметры отображения ... 599
1.3.2. Методы Grid.ocx ... 600
1.3.3. События от компоненты Grid.ocx ... 603
1.4.Краткий перечень функций, используемые Grid.ocx ... 605
**2.GridGL.ocx предназначена для графического отображения данных (улучшенная) ... 610**
2.1.Установка компонента GidGL.ocx ... 610
2.2.Назначение GridGL.ocx ... 611
2.3.Полное описание свойств, методов и событий GridGL.ocx ... 612
2.3.1. Свойства GridGL.ocx ... 612
2.3.1.1. Цвета ... 613
2.3.1.2. Параметры графиков ... 613
2.3.1.3. Параметры шкалы и дополнительного окна ... 614
2.3.1.4. Параметры надписи ... 617

*© ООО "ЭТМС" 1992-2024. Все права защищены*

2.3.1.5. Параметры отображения ... 617
2.3.1.6. Представление графиков ... 617
2.3.1.7. Область отображения графиков ... 620
2.3.2. Методы GridGL.ocx ... 621
2.3.2.1. Настройка GridGL.ocx ... 621
2.3.2.2. Отрисовка данных GridGL.ocx ... 622
2.3.2.3. Передача данных в GridGL.ocx ... 623
2.3.2.4. Сохранение данных в файл или в Clipboard ... 627
2.3.2.5. Параметры меток в GridGL.ocx ... 629
2.3.2.6. Параметры полос в GridGL.ocx ... 630
2.3.2.7. Индикационная рамка в GridGL.ocx ... 631
2.3.3. События от компоненты GridGL.ocx ... 631
2.4.Краткий перечень функций, используемые GridGL.ocx ... 633
**3.PloterXY.ocx графический компонент отображения зависимостей X(t)-Y(t)-Z(t) в 2- и 3-мерном виде**
3.1.Установка компонента PloterXY.ocx ... 649
3.2.Назначение PloterXY.ocx ... 650
3.3.Полное описание свойств, методов и событий PloterXY.ocx ... 651
3.3.1. Свойства PloterXY.ocx ... 651
3.3.1.1. Цвета ... 651
3.3.1.2. Параметры графиков ... 652
3.3.1.3. Представление графиков ... 653
3.3.1.4. Типы координат, линий сетки и подписи для осей ... 655
3.3.2. Методы PloterXY.ocx ... 655
3.3.2.1. Настройки PloterXY.ocx ... 656
3.3.2.2. Передача данных PloterXY.ocx ... 656
3.3.2.3. Границы видимости PlotterXY.ocx ... 657
3.3.2.4. Настройки курсора PloterXY.ocx ... 658
3.4.События от компоненты PloterXY.ocx ... 658
3.5.Краткий перечень функций, используемые PloterXY.ocx ... 660
**4.Polar.ocx компонент отображения графиков в полярных координатах ... 667**
4.1.Установка компонента Polar.ocx ... 667
4.2.Назначение Polar.ocx ... 668
4.3.Полное описание свойств, методов и событий Polar.ocx ... 669
4.3.1. Свойства Polar.ocx ... 669
4.3.1.1. Цвета ... 669
4.3.1.2. Параметры графиков ... 670
4.3.1.3. Представление графика ... 670
4.3.2. Методы Polar.ocx ... 671
4.3.2.1. Параметры графика ... 671
4.3.2.2. Передача данных ... 672
4.4.Краткий перечень функций, используемые Polar.ocx ... 672
**5.Gramma.ocx предназначена для графического отображения массивов данных ... 675**
5.1.Установка компонента Gramma.ocx ... 676
5.2.Назначение Gramma.ocx ... 677
5.3.Полное описание свойств, методов и событий Gramma.ocx ... 678
5.3.1. Свойства Gramma.ocx ... 678
5.3.1.1. Цвета ... 678
5.3.1.2. Параметры графиков ... 679
5.3.1.3. Представление графика ... 680
5.3.2. Методы Gramma.ocx ... 682
5.3.2.1. Параметры графика ... 682
5.3.2.2. Передача данных ... 682
5.3.3. События от компоненты Gramma.ocx ... 684
5.4.Краткий перечень функций, используемые Gramma.ocx ... 685

*© ООО "ЭТМС" 1992-2024. Все права защищены*

**6.GrammaGL.ocx предназначена для графического отображения массивов данных (улучшенная)**
6.1.Установка компонента GrammaGL.ocx ... 689
6.2.Назначение GrammaGL.ocx ... 690
6.3.Полное описание свойств, методов и событий GrammaGL.ocx ... 691
6.3.1. Свойства GrammaGL.ocx ... 692
6.3.1.1. Цвета ... 692
6.3.1.2. Параметры графиков ... 692
6.3.1.3. Параметры шкалы ... 693
6.3.1.4. Представление графика ... 695
6.3.2. Методы GrammaGL.ocx ... 696
6.3.2.1. Параметры графика ... 696
6.3.2.2. Передача данных ... 698
6.3.3. События от компоненты GrammaGL.ocx ... 700
6.4.Краткий перечень функций, используемые GrammaGL.ocx ... 701
**Описание программных модулей кнопок ... 709**
**1.ZETButtonOCX.ocx предназначена для кнопок ZETLab в меню ... 709**
1.1.Установка компонента ZETButtonOCX ... 709
1.2.Назначение ZETButtonOCX ... 710
1.3.Полное описание свойств, методов и событий ZETButtonOCX ... 711
1.3.1. Свойства ZETButtonOCX ... 711
1.3.1.1. Цвета ... 711
1.3.1.2. Параметры кнопки ... 712
1.3.1.3. Параметры текста ... 712
1.3.1.4. Параметры рисунка ... 712
1.3.2. Методы ZETButtonOCX ... 713
1.3.3. События от компоненты ZETButtonOCX ... 713
1.4.Краткий перечень функций, используемые ZETButtonOCX ... 714
**2.ZRegulator.ocx кнопка для выбора элемента из списка ... 718**
2.1.Установка компонента ZRegulator.ocx ... 718
2.2.Интерфейс компонента ZRegulator.ocx ... 719
2.3.Полное описание свойств, методов и событий ZRegulator.ocx ... 720
2.3.1. Свойства ZRegulator.ocx ... 720
2.3.1.1. Цвета ... 721
2.3.1.2. Отображение рамки ... 721
2.3.1.3. Параметры шрифта ... 721
2.3.1.4. Параметры управления списком кнопки ... 722
2.3.2. Методы ZRegulator.ocx ... 722
2.3.3. События от компоненты ZRegulator.ocx ... 722
2.4.Краткий перечень функций, используемые ZRegulator.ocx ... 723
**3.Zbutton.ocx кнопка для выбора элемента из списка ... 726**
3.1.Установка компонента Zbutton.ocx ... 726
3.2.Назначение и использование Zbutton.ocx ... 727
3.3.Полное описание свойств, методов и событий Zbutton.ocx ... 728
3.3.1. Свойства Zbutton.ocx ... 728
3.3.1.1. Цвета ... 728
3.3.1.2. Параметры шрифта ... 729
3.3.1.3. Параметры управления списком кнопки ... 729
3.3.2. Методы Zbutton.ocx ... 729
3.3.3. События от компоненты Zbutton.ocx ... 730
3.4.Краткий перечень функций, используемые Zbutton.ocx ... 730
**4.ZETButtonEx.ocx кнопка для свертывания и развертывания дополнительных окон ... 733**
4.1.Установка компонента ZETButtonEx.ocx ... 733
4.2.Назначение и использование ZETButtonEx.ocx ... 734
4.3.Полное описание свойств, методов и событий ZETButtonEx.ocx ... 735

*© ООО "ЭТМС" 1992-2024. Все права защищены*

4.3.1. Свойства ZETButtonEx.ocx ... 736
4.3.2. Методы Zbutton.ocx ... 736
**Примеры программирования ... 737**
**1.Примеры работы с CBuilder ... 740**
**2.Примеры работы с драйверами ... 740**
**3.Примеры работы с VisualBasic6 ... 740**
3.1.Пример SAMPLE_I ... 743
3.2.Пример SAMPLE_2 ... 749
3.3.Пример SAMPLE_3 ... 751
3.4.Пример SAMPLE_4 ... 752
**4.Установка компонента Grid.ocx ... 753**
**5.Примеры работы с VisualStudio2010 ... 755**
**6.Примеры программирования на .Net платформе ... 756**
6.1.Общие сведения о подключении компонентов к C# ... 756
6.2.Подключение компонента ActiveX к проекту C# ... 757
6.3.Использование компонентов ActiveX Zet в тексте программы C# ... 763
6.4.Особенности работы с указателями на массив на .Net платформе ... 764
6.5.Пример программы опроса компонента ZETServer на C# ... 765
**7.Пример управления генератором Dac ... 767**
**8.Примеры работы с драйверами ZET ... 768**
**9.Примеры использования Unit.ocx (на MS Visual Basic 6.0) ... 772**
9.1.Программа "Узкополосный спектр" ... 773
9.2.Программа "Три вольтметра - в одном" ... 774
**10.Примеры подключения ZETSENSOR к сторонним системам ... 775**
10.1.Взаимодействие с ZETSENSOR посредством пр-мы Simply Modbus ... 775
10.2.Взаимодействие с ZETSENSOR посредством программы на C ... 776
10.3.Взаимодействие с ZETSENSOR посредством компонента SCADA ZETVIEW "OPC Сервер" ... 777
**Ошибки, возникающие при работе с элементами *.OCX ... 778**
**Modbaszetlab для связи с измерительными программами ZET 7000 ZETSensor ... 780**
**Справка и техническая поддержка ... 782**

*© ООО "ЭТМС" 1992-2024. Все права защищены*

© ООО "ЭТМС" 1992-2024. Все права защищены

# ADC информация и данные в adcinfo.h

Аппаратная конфигурация DSP

1. ADC_INFO состоит из:

   - *short* ***numberDSP*** - порядковый номер DSP на плате.
   - *short* ***typeADC*** - режим работы DSP: 0 - ADC, 1 - DAC.
   - *short* ***EnableADC*** - наличие АЦП.
   - *short* ***EnableDAC -*** наличие ЦАП.
   - *short* ***EnableExtDevice -*** наличие аппаратной поддержки: 0 - ничего нет, 0x01 - PRUS, 0x02 - HCP (ICP), 0x04 - Atten, 0x08 - SD-card, 0x10 - Ethernet, 0x20 - Bluetooth, 0x40 - PWM (ШИМ), 0x80 - DigAnaliz (Логический анализатор), 0x100 - 1-Wire (TEDs), 0x200 - Подстройка по GPS, 0x400 - Синхронизация по PTP.

2. Запуск DSP:
   - *short* ***Start*** - состояние ввода данных 1- ввод.

3. Конфигурация каналов:
   - *long* ***Channel*** - битовая комбинация вводимых каналов.
   - *long* ***HCPChannel*** - битовая комбинация включенных каналов HCP.
   - *long* ***OverloadChannel*** - битовая комбинация перегруженных каналов.
   - *short* ***WorkChADC*** - количество работающих каналов АЦП.
   - *short* ***WorkChDAC*** - количество работающих каналов ЦАП.

4. Конфигурация коэффициентов усиления:
   - *short* ***KodAmplify[32]*** - коды усиления на АЦП.
   - *short* ***TypeConnection*** - тип интерфейса (1 - PCI, 2 - USB, 3 - Ethernet, 4 - Bluetouch).
   - *double* ***Amplitude[32]*** - коэффициент усиления усилителя АЦП.
   - *short* ***PRUS[32]*** - коды усиления усилителя заряда / вкл. (выкл.) HCP.
   - *double* ***PreAmplitude[32]*** - коэффициент усиления предусилителя.
   - *short* ***Atten[4]*** - коды аттенюатора ЦАП.
   - *double* ***Reduction[4]*** - коэффициент передачи аттенюатора ЦАП.

5. Конфигурация частот:

© ООО "ЭТМС" 1992-2024. Все права защищены

- *short* ***EnaExtStart*** - разрешение внешнего старта АЦП 0 - внутренний, 1 - внешний.
- *short* ***EnaExtFreq*** - разрешение внешней частоты ЦАП 0 - внутренний, 1 - внешний.
- *short* ***EnaCycle*** - разрешение циклического накопления 0-однократно, 1-циклически.
- *short* ***ExtCycle*** - MasterSynchr: 1 - Задатчик синхронизации, 0 - ведомое устройство в синхронизации.
- *short* ***ModaADC*** - мода работы АЦП 0-1, 1-2, 2-4, 3-5, 4-8, 5-10 прореживание.
- *short* ***EnaExtStartDAC*** - разрешение внешнего старта ЦАП 0 - внутренний, 1- внешний.
- *short* ***EnaExtFreqDAC*** - разрешение внешней частоты ЦАП 0 - внутренний, 1- внешний.
- *short* ***Reserv1*** - резерв 1 используется в программе CableTest.exe
- *short* ***Reserv2*** - резерв 1 используется в программе CableTest.exe
- *double* ***Freq*** - текущая частота преобразования (Freq = OporFreq/Rate).
- *double* ***OporFreq*** - внешняя опорная частота.
- *long* ***Rate*** - коэффициент деления опорной частоты.

6. Конфигурация буферов:
   - *long* ***sizeInterrupt*** - размер для перекачки данных при каждом прерывании АЦП (sizeInterrupt = QuantityPacketsADC * SizePacketADC).
   - *short* ****AddressBuffer*** - адрес буфера в системной области.
   - *short* ****AddressBufferDAC*** - адрес буфера в системной области ЦАП.
   - *short* ****AddressUserBuffer*** - адреса буфера в пользовательской области.
   - *short* ****AddressUserBufferDAC*** - адреса буфера в пользовательской области ЦАП.
   - *long* ***SizeBuffer*** - размер буфера ввода не более 1 000 000 отсчетов.
   - *long* ***SizeBufferDAC*** - размер буфера генератора не более 1 000 000 отсчетов.
   - *long* ***Pointer*** - индекс текущего накопления в буфере АЦП.
   - *long* ***PointerDAC*** - индекс текущего накопления в буфере ЦАП.
   - *long* ***PointerDSP*** - индекс прошедшего накопления в DSP.

7. Аппаратные параметры АЦП (ЦАП):
   - *short* ***Bits*** - количество бит в АЦП преобразователе.
   - *short* ***BitsDAC*** - количество бит в ЦАП.
   - *short* ***Words*** - количество 16-разрядных слов в одном отсчете.
   - *short* ***WordsDAC*** - количество 16-разрядных слов в одном отсчете ЦАП.
   - *short* ***QuantityChannelDigPort*** - количество линий цифрового порта.
   - *double* ***DigitalResolution*** - вес младшего разряда АЦП.
   - *double* ***DigitalResolutionDAC*** - вес младшего разряда ЦАП.

© ООО "ЭТМС" 1992-2024. Все права защищены

- *double* ***MaxFreq*** - максимальная частота преобразования.
- *short* ***QuantityChannelADC*** - максимальное количество каналов в модуле АЦП.
- *short* ***QuantityChannelDAC*** - максимальное количество каналов в модуле ЦАП.
- *short* ***Reserv3*** - не используется в программах из setup.

8. Переменные отслеживающие временные параметры:
   - *ulonglong* ***Flag*** - количество прерываний.
   - *ulonglong* ***CurrentCounter*** - значение счетчика в момент опроса.
   - *ulonglong* ***IrqCounter*** - значение счетчика в текущее прерывание.
   - *ulonglong* ***IrqPeriod*** - текущий период прерываний.
   - *ulonglong* ***FreqCounter*** - частота счетчика.

9. Прочие переменные:
   - *long* ***MyError*** - ошибка ввода или задания параметров .
   - *short* ***modify*** - номер изменения, каждое существенное изменение параметров устройства, делает инкремент этого параметра.
   - *long* ***AdrDSP*** - адрес из DSP.
   - *void* ****AdrMem*** - адрес памяти для DSP.
   - *long* ***count*** - количество слов для передачи.
   - *long* ***numberCHNL*** - номер канала (служебная переменная.
   - *short* ***TestCode[6]*** - тестовые проверки. Используется для информации об включении кнопки генератора на передней панели анализатора для анализаторов последних версий \ \\192.168.0.7\Common\Разработка\ZET017U8\Прошивки DSP\ZET017U8_DSP_R6 v6.7 с индикацией состояния красной кнопки от 2013.09.26\

   Код используется для проверки в программах СУВ:

```
Пример:ADC_INFO info;
        if (!ZGetInfo(tadc, ndsp, &info))
                {
         if (info.TestCode[2] & 0x8000)
                {
          // "красная" кнопка нажата (горит светодиод на кнопке) - ЦАП работает

          // старые анализаторы без поддержки "красной" кнопки всегда будут выдавать данный флаг
                }
         else
                {
         // "красная" кнопка отпущена (светодиод погашен) - ЦАП остановлен
```

© ООО "ЭТМС" 1992-2024. Все права защищены

}
}

- *char* ***verDSP[32]*** - версия DSP.
- *char* ***DeviceName[16]*** - имя прибора.
- *char* ***Reserv4[16]*** - зарезервировано.
- *char* ***verDriver[64]*** - версия драйвера.

10. Прочие переменные ЦАП:
    - *long* ***sizeInterruptDAC*** **-** размер для перекачки данных при каждом прерывании ЦАП.(sizeInterruptDAC = QuantityPacketsDAC * SizePacketDAC).
    - *long* ***StartDAC*** - состояние ввода данных ЦАП 1- ввод
    - *long* ***MaxSizeInterrupt*** - максимально возможный размер для перекачки данных при каждом прерывании.
    - *long* ***MaxSizeInterruptDAC*** - максимально возможный размер для перекачки данных при каждом прерывании ЦАП.
    - *long* ***ChannelDAC*** - битовая комбинация выводимых каналов ЦАП.
    - *long* ***EnaCycleDAC*** - разрешение циклического накопления ЦАП: 0-однократно, 1-циклически.
    - *long* ***ExtCycleDAC*** - 0 - генерация сигнала из внутренней памяти DSP, без подкачки из внешней памяти, 1 - генерация сигнала из внешней памяти с подкачкой по прерываниям
    - *double* ***FreqDAC*** - текущая частота преобразования ЦАП (FreqDAC = OporFreqDAC/RateDAC).
    - *double* ***OporFreqDAC*** - опорная частота ЦАП
    - *double* ***MaxFreqDAC*** **-** максимальная частота преобразования ЦАП.
    - *long* ***RateDAC*** - коэффициент деления опорной частоты ЦАП.

11. Прочие переменные цифрового порта:
    - *long* ***DigitalInput*** - входные данные цифрового порта.
    - *long* ***DigitalOutput*** - выходные данные цифрового порта.
    - *long* ***DigitalOutEnable*** **-** битовая комбинация выходов цифрового порта.
    - *float* ***DigitalResolChanADC[32]*** - откалиброванный вес младшего разряда АЦП.
    - *float* ***DigitalResolChanDAC[4]*** - откалиброванный вес младшего разряда ЦАП.

12. Управление устройством:
    - *long* ***StopDevice*** **-** останов устройства.
    - *long* ***NumVersionDriver*** - бинарная версия драйвера.
    - *long* ***ChannelDiff*** - битовая комбинация дифференциальных каналов.
    - *short* ***SizePacketADC*** **-** количество слов данных в пакете АЦП.
    - *short* ***SizePacketDAC*** - количество слов данных в пакете ЦАП.

© ООО "ЭТМС" 1992-2024. Все права защищены

- *short* ***MaxSizePacketADC*** - максимальное количество слов данных в пакете АЦП.
- *short* ***MaxSizePacketDAC*** - максимальное количество слов данных в пакете ЦАП.
- *short* ***QuantityPacketsADC*** - количество пакетов передаваемых за одно прерывание АЦП.
- *short* ***QuantityPacketsDAC*** - количество пакетов передаваемых за одно прерывание ЦАП.
- *short* ***MaxQuantityPacketsADC*** - максимальное количество пакетов передаваемых за одно прерывание АЦП.
- *short* ***MaxQuantityPacketsDAC*** - максимальное количество пакетов передаваемых за одно прерывание ЦАП.
- *short* ***Revision*** - ревизия устройства.
- *short* **Reserved[9]** - зарезервировано ADC_INFO, *PADC_INFO;

13. Управление устройством с помощью ШИМ:
    - *short* ***StartPWM*** - запуск ШИМ (0x01 - старт 1 канала, 0x02 - старт 2 канала, 0x04 - старт 3 канала, 0x100 - стоп 1 канала, 0x200 - стоп 2 канала, 0x400 - стоп 3 канала)
    - *short* ***RatePWM*** - коэффициент. деления опорной частоты для ШИМ.
    - *short* ***PeriodPWM*** - период ШИМ.
    - *short* ***OnDutyPWM0*** - скважность ШИМ (канал 0).
    - *short* ***OnDutyPWM1*** - скважность ШИМ (канал 1).
    - *short* ***OnDutyPWM2*** - скважность ШИМ (канал 2).
    - *short* ***ShiftPWM1*** - сдвиг ШИМ (канал 0).
    - *short* ***ShiftPWM2*** - сдвиг ШИМ (канал 1).

14. Прочие:
    - *long* ***SizeBufferDSP_DAC*** - размер буфера SDRAM ЦАП в DSP.
    - *long* ***MaxSizeBufferDSP_DAC*** - максимальный размер буфера SDRAM ЦАП в DSP.
    - *double* ***ExtOporFreqDAC*** - внешняя опорная частота ЦАП.
    - *long* ***OffsetChanADC[32]*** - откалиброванное смещение АЦП.
    - *long* ***OffsetChanDAC[4]*** - откалиброванное смещение ЦАП.
    - *ulonglong* ***StartTime*** - дата и время синхронного запуска.
    - *ulonglong* ***GPSTime*** - дата и время приемника GPS.
    - *longlong* ***Drift*** - текущее смещение отн-но ведущего устр-ва (младшие 4 байта).
    - *longlong* ***SyncDrift*** - текущее смещение в момент последней синхронизации.
    - *char* ***SyncCount*** - счетчик синхронизаций.
    - *char* ***SyncFlag*** - флаг синхронизации.

© ООО "ЭТМС" 1992-2024. Все права защищены

- *char* ***SatsInUse*** - количество навигационных спутников, доступных для использования.
- *char* ***StatePTP*** - состояние порта PTP: 0 - PTP недоступен, 3 - Disabled, 6 - Master, 9 - Slave.
- *unsigned short* ***ReservedExt[264]*** - зарезервировано.
- *ulonglong* ***FreqCounterPC*** - используется внутри драйвера.
- *ulonglong* ***StartCounterPC*** - используется внутри драйвера.
- *ulonglong* ***EndCounterPC*** - используется внутри драйвера.
- *ulonglong* ***DurationSynhrPC*** -используется внутри драйвера.
- *unsigned long* ***TimerDSP*** - используется внутри драйвера.
- *unsigned long* ***NumberSecondDSP*** - используется внутри драйвера.
- *unsigned long* ***FreqQuartzDSP*** - используется внутри драйвера.
- *unsigned long* ***Reserved1*** - зарезервировано.
- *unsigned long* ***Reserved2*** - зарезервировано.
- *unsigned long* ***Reserved3*** - зарезервировано.
- *unsigned long* ***Reserved4*** - зарезервировано.
- *unsigned long* ***Reserved5*** - зарезервировано.

15.Команды DSP:

- *unsigned short* ***Command*** - команда.
- *unsigned short* ***Error*** -дополнительный код или ошибка.
- *unsigned long* ***Size*** - размер в байтах.
- *unsigned short* ***Data[252]*** - данные.

© ООО "ЭТМС" 1992-2024. Все права защищены

# Подключение ZET017 U4/U8/T8 по Ethernet

Описание протокола

Введение
- Режимы работы устройства
- Порты TCP/IPv4

Формат пакетов
- Команды
  - Команда 0x0000 GetInfo
  - Команда 0x0001 ReadDigitalPort
  - Команда 0x0002 WriteDigitalPort
  - Команда 0x0003 EnableDigitalPort
  - Команда 0x0012 PutInfo
  - Команда 0x0207 Restart

Подключение к устройству
- Установка соединения TCP/IPv4
- Согласование
- Инициализация
- Управление устройством
- Разрыв соединения

Получение данных АЦП
- Настройка каналов АЦП
- Запуск АЦП
- Получение данных
- Остановка АЦП

Работа генератора
- Настройка генератора
- Запуск генератора
- Отправка данных генератора
- Остановка генератора
- Синхронная работа АЦП и ЦАП

Пример программы

© ООО "ЭТМС" 1992-2024. Все права защищены

# Глава 1.Введение

В документе описывается сетевой протокол, используемый для подключения к анализаторам и тензостанциям ZET017 U4/U8/T8 производства ООО «ЭТМС».

Далее в документе под *устройством* будет иметься в виду анализатор или тензостанция ZET017 U4/U8/T8, а под *клиентом* — ПК либо другое техническое средство, которое подключается к устройству для управления им и обмена данными АЦП/ЦАП.

## 1.1.Режимы работы устройства

Устройство может работать в одном из трех режимов:

1. Подключение по USB.
2. Подключение по Ethernet.
3. Автономный регистратор.

Режим USB доступен всегда, а режимы Ethernet и автономного регистратора доступны в виде опций.

Для работы устройства в режиме Ethernet используется протокол, описанный в этом документе.

Протокол работает поверх транспортного протокола TCP/IPv4 в режиме сервера (то есть устройство не подключается к кому-нибудь, а ожидает подключения от клиента).

Перед использованием устройства в режиме Ethernet, в нем должна быть предварительно произведена настройка параметров сетевого интерфейса. Настройка сетевого интерфейса выполняется в режиме USB с помощью ПО ZETLAB.

Параметры сетевого интерфейса:

адрес IPv4;
маска подсети;
адрес сетевого шлюза;
номер порта TCP/IPv4;
режим дуплекс/полудуплекс (обычно дуплекс);
скорость 100 Мбит/с или 10 Мбит/с (обычно 100 МБит/с).

Адрес IPv4 настраивается статически, так как устройство не поддерживает протоколы DHCP/BOOTP для автоматического определения адреса. По умолчанию в устройстве настроены адрес 192.168.0.100, маска подсети 255.255.255.0 и шлюз 192.168.0.1.

© ООО "ЭТМС" 1992-2024. Все права защищены

Номер порта TCP/IPv4 доступен в настройках, однако фактически принимается только значение 1808.

## 1.2.Порты TCP/IPv4

Устройство открывает три порта TCP/IPv4:

| Номер порта | Название порта | Назначение |
|---|---|---|
| 1808 | Командный порт | Для управления устройством |
| 2320 (1808 + 512) | Порт выдачи АЦП | Для получения данных по каналам АЦП |
| 3344 (1808 + 3*512) | Порт приема ЦАП | Для отправки данных для ЦАП |

Командный порт двусторонний: клиент отправляет по нему команды, а устройство — ответы на команды.

Порты АЦП и ЦАП односторонние и используются для передачи данных АЦП от устройства клиенту и для передачи данных ЦАП от клиенту устройства, соответственно.

# Глава 2.Формат пакетов

В документе описывается сетевой протокол, используемый для подключения к анализаторам и тензостанциям ZET017 U4/U8/T8 производства ООО «ЭТМС».

Далее в документе под *устройством* будет иметься в виду анализатор или тензостанция ZET017 U4/U8/T8, а под *клиентом* — ПК либо другое техническое средство, которое подключается к устройству для управления им и обмена данными АЦП/ЦАП.

## 2.1.Команды

Первые два байта командного пакета содержат код команды. Содержимое остальных 1022 байтов зависит от значения кода команды.

| Код команды | Команда | Описание |
|---|---|---|
| 0x0000 | GetInfo | Получение текущей информации об устройстве |
| 0x0001 | ReadDigitalPort | Чтение цифрового порта |
| 0x0002 | WriteDigitalPort | Запись в цифровой порт |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | | |
|---|---|---|
| 0x0003 | EnableDigitalPort | Управление цифровым портом |
| 0x0012 | PutInfo | Изменение текущих настроек устройства |
| 0x0207 | Restart | Программный перезапуск устройства |

В таблицах ниже типы полей указаны через <stdint.h> для точного указания разрядности.
Столбец **Упр.** (управление) определяет, является ли поле настраиваемым (R/W — read/write) или нет (R/O — read only).

## Команда 0x0000 GetInfo

### Команда 0x0000 GetInfo

Формат запроса:

| Смещение | Поле (тип и название) | Упр. | Описание |
|---|---|---|---|
| 0x0000 | uint16_t Command | R/W | Код команды (0x0000 — GetInfo) |

Остальные поля не имеют значения и не проверяются устройством.

Формат ответа:

| Смещение | Поле (тип и название) | Упр. | Описание |
|---|---|---|---|
| 0x0000 | uint16_t Command | R/W | Код команды (0x0000 — GetInfo) |
| 0x0004 | int16_t StartADC | R/W | Управление АЦП (-1, 0, 1) |
| 0x0006 | int16_t StartDAC | R/W | Управление ЦАП (-1, 0, 1) |
| 0x000e | uint32_t QuantityChannelADC | R/O | Общее количество каналов АЦП (4, 8) |
| 0x0010 | uint32_t QuantityChannelDAC | R/O | Общее количество каналов ЦАП (1) |
| 0x0012 | uint8_t TypeDataADC | R/O | Тип данных АЦП (0 — int16_t, 1 — int32_t) |
| 0x0013 | uint8_t TypeDataDAC | R/O | Тип данных ЦАП (0 — int16_t) |
| 0x0014 | uint32_t ChannelADC | R/W | Маска активных каналов АЦП (0x00..0xFF) |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | | | |
|---|---|---|---|
| 0x0018 | uint32_t ChannelDAC | R/W | Маска активных каналов ЦАП (0x00..0x01) |
| 0x001C | uint32_t ICPChannel | R/W | Маска ICP в каналах АЦП (0x00..0xFF) |
| 0x0024 | uint16_t WorkChADC | R/W | Количество активных каналов АЦП (1..8) |
| 0x0026 | uint16_t WorkChDAC | R/W | Количество активных каналов ЦАП (0..1) |
| 0x0028 | uint16_t CodAmplify[8] | R/W | Коэффициенты усиления (0, 1, 2) |
| 0x00BA | uint16_t ModaADC | R/W | Режим работы АЦП (0..4) |
| 0x00BE | uint16_t RateDAC | R/W | Режим работы ЦАП (400, 800, 1600, 3200) |
| 0x00D8 | uint32_t DigitalInput | R/O | Состояние входов цифрового порта |
| 0x00DC | uint32_t DigitalOutput | R/W | Состояние выходов цифрового порта |
| 0x00EC | char VersionDSP[32] | R/O | Строка с версией устройства |
| 0x010C | char DeviceName[16] | R/O | Строка с названием устройства |
| 0x012C | uint32_t SerialNumber | R/O | Серийный номер устройства |
| 0x013C | uint32_t DigitalOutEnable | R/W | Разрешения на запись цифрового порта |
| 0x014C | float DigitalResolutionADC[16] | R/O | Вес младшего разряда АЦП |

## Команда 0x0001 ReadDigitalPort

Формат запроса:

| Смещение | Поле (тип и название) | Упр. | Описание |
|---|---|---|---|
| 0x0000 | uint16_t Command | R/W | Код команды (0x0001 — ReadDigitalPort) |

Остальные поля не имеют значения и не проверяются устройством.

Формат ответа:

| Смещение | Поле (тип и название) | Упр. | Описание |
|---|---|---|---|

© ООО "ЭТМС" 1992-2024. Все права защищены

| | | | |
|---|---|---|---|
| 0x0000 | uint16_t Command | R/W | Код команды (0x0001 — ReadDigitalPort) |
| 0x00D8 | uint32_t DigitalInput | R/O | Состояние входов цифрового порта |

Остальные поля должны быть проигнорированы.

### Команда 0x0002 WriteDigitalPort

Формат запроса:

| **Смещение** | **Поле (тип и название)** | **Упр.** | **Описание** |
|---|---|---|---|
| 0x0000 | uint16_t Command | R/W | Код команды (0x0002 — WriteDigitalPort) |
| 0x00DC | uint32_t DigitalOutput | R/W | Состояние выходов цифрового порта |

Остальные поля не имеют значения и не проверяются устройством.

Формат ответа:

| **Смещение** | **Поле (тип и название)** | **Упр.** | **Описание** |
|---|---|---|---|
| 0x0000 | uint16_t Command | R/W | Код команды (0x0002 — WriteDigitalPort) |

Остальные поля должны быть проигнорированы.

### Команда 0x0003 EnableDigitalPort

Формат запроса:

| **Смещение** | **Поле (тип и название)** | **Упр.** | **Описание** |
|---|---|---|---|
| 0x0000 | uint16_t Command | R/W | Код команды (0x0003 — EnableDigitalPort) |
| 0x013C | uint32_t DigitalOutEnable | R/W | Разрешения на запись цифрового порта |

Остальные поля не имеют значения и не проверяются устройством.

Формат ответа:

| **Смещение** | **Поле (тип и название)** | **Упр.** | **Описание** |
|---|---|---|---|
| 0x0000 | uint16_t Command | R/W | Код команды (0x0003 — EnableDigitalPort) |

Остальные поля должны быть проигнорированы.

© ООО "ЭТМС" 1992-2024. Все права защищены

### Команда 0x0012 PutInfo

Формат запроса:

| Смещение | Поле (тип и название) | Упр. | Описание |
|---|---|---|---|
| 0x0000 | uint16_t Command | R/W | Код команды (0x0012 — PutInfo) |

Остальные данные пакета соответствуют формату ответа на команду 0x0000 GetInfo.

Перед тем, как отправить пакет с запросом, необходимо сначала считать текущие настройки с помощью команды 0x0000 GetInfo и затем изменить в полученном пакете значения требуемых полей. Поля R/O изменять нельзя.

Формат ответа соответствует формату ответа на команду 0x0000 GetInfo, включая значения кода команды (Command == 0x0000).

### Команда 0x0207 Restart

Формат запроса:

| Смещение | Поле (тип и название) | Упр. | Описание |
|---|---|---|---|
| 0x0000 | uint16_t Command | R/W | Код команды (0x0207 — Restart) |
| 0x0008 | uint16_t Restart | W | Перезапуск (0x0001) |

Остальные данные пакета должны быть обнулены.

Устройство будет перезапущено сразу же, без отправки ответа.

# Глава 3.Подключение к устройству

## Подключение к устройству

Подключение к устройству производится в несколько этапов.

### Установка соединения TCP/IPv4

Клиент должен установить соединение по протоколу TCP/IPv4 ко всем трем портам устройства. Порядок подключений не важен.

Адрес IPv4 должен быть известен клиенту заранее, так как в устройстве нет средств для его обнаружения в локальной сети с помощью широковещательных или групповых запросов (таких как, например, SNMP).

### Согласование

После установки соединения по любому порту, устройство отправляет клиенту пакет для согласования.

© ООО "ЭТМС" 1992-2024. Все права защищены

| Смещение | Поле (тип и название) | Описание |
|---|---|---|
| 0x0000 | uint32_t size | Размер последующих данных |
| 0x0004 | uint8_t handshake[size] | Данные |

Размер данных обычно находится в диапазоне от 0 до 2048, включительно.

Сами данные не несут никакой полезной информации — их просто требуется вычитать по порту для согласования с устройством.

### Инициализация

После согласования требуется инициализация устройства. Для этого требуется отправить по командному порту две команды:

1. 0x0000 GetInfo — для получения текущей информации об устройстве.
2. 0x0012 PutInfo — для инициализации устройства.

### Управление устройством

После успешной инициализации устройство становится готовым для работы.

По командному порту можно отправлять команды для изменения настроек, управления цифровым портом устройства, включения и отключения АЦП или ЦАП.

### Разрыв соединения

При разрыве соединения по любому из портов (по инициативе клиента или из-за потери связи) устройство отключает все остальные порты, останавливает работу АЦП и ЦАП, после чего переходит в режим ожидания нового соединения.

Факт потери связи определяется по факту отсоединения патч-корда (link down) или через TCP Keep-Alive.

# Глава 4.Получение данных АЦП

### Настройка каналов АЦП

Перед тем, как получать данные с АЦП, требуется настройка каналов АЦП.

Настройка производится командой PutInfo.

Всего в устройстве доступно Info.QuantityChannelADC каналов (4 или 8).

Любой канал может быть активен или неактивен (при условии, что активным остается как минимум один канал), а также он может быть включен в режиме ICP или без него.

© ООО "ЭТМС" 1992-2024. Все права защищены

Активация каналов управляется полем Info.ChannelADC, в котором биты от нулевого по седьмой отвечают за активность соответствующего канала. Например, для активации каналов 1, 2 и 4 поле должно принимать значение Info.ChannelADC == 0x000B (установлены 0-й, 1-й и 3-й биты).

Общее количество активных каналов указывается в поле Info.WorkChADC. В случае активации каналов 1, 2 и 4 поле принимает значение 3.

Режим ICP управляется полем Info.ICPChannel, которое также содержит битовую маску для каждого канала.

Все каналы оцифровываются данные синхронно, частота дискретизации у них общая. Частота управляется полем Info.ModaADC:

- 50 кГц (Info.ModaADC == 1);
- 25 кГц (Info.ModaADC == 2);
- 5 кГц (Info.ModaADC == 3);
- 2 500 Гц (Info.ModaADC == 4);
- 25 кГц (по умолчанию для всех других значений Info.ModaADC).

Для каждого канала можно задать коэффициент усиления с помощью массива Info.CodAmpllify:

- КУ 1 (Info.CodAmplify[i] == 0, по умолчанию);
- КУ 10 (Info.CodAmplify[i] == 1);
- КУ 100 (Info.CodAmplify[i] == 2);

## Запуск АЦП

Для запуска АЦП после настройки каналов требуется отправить команду PutInfo с полем Info.StartADC, равным 1.

## Получение данных

Данные передаются по порту АЦП по мере их готовности и только при запущенном АЦП (Info.StartADC == 1).

Данные передаются пакетами по 1024 байта. Первые 1008 байтов содержат очередную порцию отчетов. Отчеты передаются кадрами — группами по одному отчету с каждого активного канала.

Например, если в устройстве активированы 1-й, 2-й и 4-й каналы, то данные будут передаваться в следующей последовательности:

- [отчет №1 канала 1], [отчет №1 канала 2], [отчет №1 канала 4];
- [отчет №2 канала 1], [отчет №2 канала 2], [отчет №2 канала 4];
- [отчет №3 канала 1], [отчет №3 канала 2], [отчет №3 канала 4];
- и так далее.

© ООО "ЭТМС" 1992-2024. Все права защищены

Каждый отчет передается в «сыром» виде в формате int16_t (2 байта) или long (4 байта), в зависимости от разрядности АЦП в устройстве (Info.TypeDataADC).

Байты с 1008-го по 1015-й, включительно, не задействованы — их следует игнорировать.

Последние 8 байтов (с 1016-го по 1023-й, включительно) содержат счетчик пакетов, который инкрементируется на 1 в каждом следующем отправленном пакете. Счетчик пакетов служит для проверки их последовательности.

### Остановка АЦП

Для остановки АЦП требуется отправить команду с полем Info.StartADC == -1 (запрос на остановку АЦП).

После отправки команды следует продолжать чтение пакетов по порту АЦП до тех пор, пока устройство не отправит нулевой пакет (состоящие из 1024 нулевых байтов).

После этого требуется отправить команду с полем Info.StartADC == 0 (окончание остановки АЦП).

# Глава 5.Работа генератора

### Настройка генератора

В устройстве есть только один канал генератора.

Настройка производится с помощью команды PutInfo.

Частота дискретизации ЦАП настраивается полем Info.RateDAC:

- 200 кГц (Info.RateDAC == 400);
- 100 кГц (Info.RateDAC == 800);
- 50 кГц (Info.RateDAC == 1600);
- 25 кГц (Info.RateDAC == 3200).

Значения Info.RateDAC, отличные от приведенных в списке, возможны, но могут привести к некорректной работе генератора.

Итоговая частота дискретизации вычисляется по следующей формуле:

FreqDAC = RefFreqDAC / RateDAC, где RefFreqDAC == 80 МГц.

© ООО "ЭТМС" 1992-2024. Все права защищены

### Запуск генератора

Запуск ЦАП синхронизирован с запуском АЦП, поэтому для запуска ЦАП требуется отправить команду PutInfo с полем Info.StartDAC == 1, а затем запустить АЦП.

### Отправка данных генератора

Устройство принимает пакеты ЦАП только при включенном генераторе (Info.StartDAC == 1).

Пакет ЦАП состоит из 1024 байтов, содержащих очередные 512 отчетов ЦАП в «сыром» виде в формате int16_t (2 байта).

Следует учитывать, что данные ЦАП поступают на аппаратном выходе генератора с задержкой относительно их отправки, поэтому данные требуется отправлять с небольшим опережением.

В устройстве имеется буфер для накопления данных ЦАП. Устройство принимает новые данные по порту ЦАП только когда в буфере есть свободное место. При заполнении буфера прием данных приостанавливается до тех пор, пока буфер не будет освобожден по мере фактической выдачи данных на выходе генератора.

Устройство может отправлять данные по порту ЦАП. Их следует вычитывать во избежание переполнения буфера отправки в устройстве. Сами данные можно игнорировать.

### Остановка генератора

Для остановки ЦАП следует отправить команду PutInfo с полем Info.StartDAC == -1, а затем команду PutInfo с полем Info.StartDAC == 0.

## Синхронная работа АЦП и ЦАП

Для синхронной работы АЦП и ЦАП необходимо, чтобы данные для ЦАП уже находились во внутреннем буфере устройства к моменту их фактического преобразования на аналоговом выходе генератора.

Во время запуска ЦАП буфер уже должен быть предварительно заполнен. Этого можно добиться следующим образом:

1. Остановить АЦП и ЦАП (Info.StartADC == 0, Info.StartDAC == 0).
2. Подать команду на запуск ЦАП (Info.StartDAC == 1). По этой команде устройство не производит фактический запуск ЦАП, а только начинает прием данных во внутренний буфер.

© ООО "ЭТМС" 1992-2024. Все права защищены

3. Отправить первоначальные данные для ЦАП. Данные необходимо отправлять либо до тех пор, пока устройство не приостановит прием (это означает, что внутренний буфер полностью заполнен), либо просто в течение некоторого времени (от 100 мс до нескольких секунд).
4. Подать команду на запуск АЦП (Info.StartADC == 1), при этом оставив ЦАП в запущенном состоянии (Info.StartDAC == 1). По этой команде производится синхронный запуск АЦП и ЦАП. Данные для ЦАП берутся из внутреннего буфера.
5. Отправлять очередные данные для ЦАП по мере того, как их будет запрашивать устройство. Внутренний буфер будет освобождаться со скоростью, соответствующей частоте дискретизации ЦАП.

# Глава 6.Пример программы

Ниже приведен пример небольшой программы. Программа подключается к устройству с IP-адресом 192.168.0.100, запускает АЦП (8 каналов, 25 кГц) и ЦАП (200 кГц) и выдает сообщение раз в секунду.

```
#include <string.h>
#include <stdlib.h>
#include <stdint.h>
#include <unistd.h>
#include <fcntl.h>

#include <sys/socket.h>
#include <sys/select.h>
#include <sys/ioctl.h>
#include <arpa/inet.h>
#include <netinet/in.h>

#include <stdio.h>

#define ZET_CMD_GET_INFO 0x0000
#define ZET_CMD_PUT_INFO 0x0012

#define ZET_NET_PACKET_SIZE 1024
#define ZET_NET_MAX_FLUSH_SIZE 2048

struct zet_info
{
    uint16_t Command;               //0x000 Код команды (0x0000 – GetInfo)
    uint8_t reserv1[2];

    int16_t StartADC;               //0x004 Управление АЦП (1, 0, 1)
    int16_t StartDAC;               //0x006 Управление ЦАП (1, 0, 1)
    uint8_t reserv2[6];
    uint16_t QuantityChannelADC;    //0x00e Общее количество каналов АЦП (4, 8)
    uint16_t QuantityChannelDAC;    //0x010 Общее количество каналов ЦАП (1)
    uint8_t TypeDataADC;            //0x012 Тип данных АЦП (0 – int16_t, 1 – long)
```

© ООО "ЭТМС" 1992-2024. Все права защищены

```
    uint8_t TypeDataDAC;            //0x013 Тип данных ЦАП (0 — int16_t)
    uint32_t ChannelADC;            //0x014 Маска активных каналов АЦП (0x00..0xFF)
    uint32_t ChannelDAC;            //0x018 Маска активных каналов ЦАП (0x00..0x01)
    uint32_t ICPChannel;            //0x01c Маска ICP в каналах АЦП (0x00..0xFF)
    uint8_t reserv3[4];
    uint16_t WorkChADC;             //0x024 Количество активных каналов АЦП (1..8)
    uint16_t WorkChDAC;             //0x026 Количество активных каналов ЦАП (0..1)
    uint16_t CodAmplify[8];         //0x028 Коэффициенты усиления каналов АЦП (0, 1,
2)
    uint8_t reserv4[130];
    uint16_t ModaADC;               //0x0ba Режим работы АЦП (0..4)
    uint8_t reserv5[2];
    uint16_t RateDAC;
    uint8_t reserv6[24];
    uint32_t DigitalInput;          //0x0d8 Состояние входов цифрового порта
    uint32_t DigitalOutput;         //0x0dc Состояние выходов цифрового порта
    uint8_t reserv7[12];
    char VersionDSP[32];            //0x0ec Строка с версией устройства
    char DeviceName[16];            //0x10c Строка с названием устройства
    uint8_t reserv8[16];
    uint32_t SerialNumber;          //0x12c Серийный номер устройства
    uint8_t reserv9[12];
    uint32_t DigitalOutEnable;      //0x13c Разрешения на запись цифрового порта
    uint8_t reserv10[16];
    float DigitalResolutionADC[16]; //0x14c Вес младщего разряда каналов АЦП
};

struct zet_adc16
{
    int16_t samples[(ZET_NET_PACKET_SIZE - sizeof(uint64_t)) / sizeof(int16_t)];
    uint64_t flag;
};

struct zet_adc32
{
    int16_t samples[(ZET_NET_PACKET_SIZE - sizeof(uint64_t)) / sizeof(int16_t)];
    uint64_t flag;
};

struct zet_dac16
{
    int16_t samples[ZET_NET_PACKET_SIZE / sizeof(int16_t)];
};

union zet_net_packet
{
    uint8_t raw[ZET_NET_PACKET_SIZE];
    struct zet_info info;
    struct zet_adc16 adc16;
    struct zet_adc32 adc32;
    struct zet_dac16 dac16;
};

#define ZET017_CMD_PORT 1808
#define ZET017_ADC_PORT 2320
#define ZET017_DAC_PORT 3344

struct zet017_device
{
    union zet_net_packet cmd;
```

© ООО "ЭТМС" 1992-2024. Все права защищены

```
    int cmd_sock;
    int adc_sock;
    int dac_sock;
};

// Создает сокет, подключается, производит согласование и делает сокет неблокирующим
static int connect_sock(const char *ip_str, unsigned short port, int
make_nonblocking)
{
    char flush_data[ZET_NET_MAX_FLUSH_SIZE];
    uint32_t flush_size;

    struct sockaddr_in addr;
    int s;
    int rc;

    fprintf(stdout, "Connecting to <%s:%d>\n", ip_str, port);

    s = socket(AF_INET, SOCK_STREAM, 0);
    if (s < 0) {
        fprintf(stderr, "socket create error\n");
        return -1;
    }

    memset(&addr, 0, sizeof(addr));
    addr.sin_family = AF_INET;
    inet_aton(ip_str, &addr.sin_addr);
    addr.sin_port = htons(port);

    rc = connect(s, (struct sockaddr *) &addr, sizeof(addr));
    if (rc < 0) {
        close(s);
        return -1;
    }

    rc = recv(s, &flush_size, sizeof(flush_size), 0);
    if (rc < 0) {
        fprintf(stderr, "handshake error %d\n", rc);
        close(s);
        return -1;
    }

    rc = recv(s, flush_data, flush_size, 0);
    if (rc < 0) {
        fprintf(stderr, "handshake error %d\n", rc);
        close(s);
        return -1;
    }

    if (make_nonblocking) {
        int on = 1;
        rc = ioctl(s, FIONBIO, &on);
        if (rc < 0) {
            fprintf(stderr, "socket ioctl error %d\n", rc);
            close(s);
            return -1;
        }
    }
```

© ООО "ЭТМС" 1992-2024. Все права защищены

```
    return s;
}

int zet017_process_command(struct zet017_device *d, uint16_t command)
{
    int rc;

    d->cmd.info.Command = command;
    rc = send(d->cmd_sock, &d->cmd, sizeof(d->cmd), 0);
    if (rc != sizeof(d->cmd)) {
        fprintf(stderr, "send command error %d\n", rc);
        return rc;
    }
    rc = recv(d->cmd_sock, &d->cmd, sizeof(d->cmd), 0);
    if (rc != sizeof(d->cmd)) {
        fprintf(stderr, "recv command error %d\n", rc);
        return rc;
    }
    return 0;
}

int zet017_connect(struct zet017_device *d, const char *device_ip_str)
{
    memset(d, 0, sizeof(struct zet017_device));

    d->cmd_sock = connect_sock(device_ip_str, ZET017_CMD_PORT, 0);
    d->adc_sock = connect_sock(device_ip_str, ZET017_ADC_PORT, 1);
    d->dac_sock = connect_sock(device_ip_str, ZET017_DAC_PORT, 1);

    // Иницализация настройками по умолчанию
    if (zet017_process_command(d, ZET_CMD_GET_INFO) < 0)
        return -1;
    d->cmd.info.ChannelADC = 0xff;
    d->cmd.info.ChannelDAC = 0x01;
    d->cmd.info.ModaADC = 2;
    d->cmd.info.RateDAC = 800;
    d->cmd.info.StartADC = 0;
    d->cmd.info.StartDAC = 0;
    if (zet017_process_command(d, ZET_CMD_PUT_INFO) < 0)
        return -1;

    fprintf(stdout, "Device initialized\n");
    return 0;
}

int zet017_disconnect(struct zet017_device *d)
{
    close(d->cmd_sock);
    close(d->adc_sock);
    close(d->dac_sock);
    d->cmd_sock = -1;
    return 0;
}

int main( int argc, char *argv[] )
{
    struct zet017_device zet017;
```

© ООО "ЭТМС" 1992-2024. Все права защищены

```
union zet_net_packet adc_recv;
union zet_net_packet dac_send;
union zet_net_packet dac_recv;

const uint32_t adc_rate = 25000;
uint32_t adc_count;
uint32_t adc_time;

const uint32_t dac_rate = 200000;
uint32_t dac_count;
uint32_t dac_time;

int width;
fd_set rfd, wfd;
struct timeval tv;
int rc;
int i;

//Подключиться к устройству
if (zet017_connect(&zet017, "192.168.0.100") < 0)
    return -1;

//Сформировать данные ЦАП
for (i = 0; i < 512; i++)
{
    dac_send.dac16.samples[i] = (i & 1) ? 0x3333 : 0x1111;
}

//Запустить АЦП и ЦАП
zet017.cmd.info.ChannelADC = 0xff;
zet017.cmd.info.ChannelDAC = 0x01;
zet017.cmd.info.ModaADC = 2;
zet017.cmd.info.RateDAC = 400;
zet017.cmd.info.StartADC = 1;
zet017.cmd.info.StartDAC = 1;
zet017_process_command(&zet017, ZET_CMD_PUT_INFO);

printf("ADC/DAC started\n");
width = zet017.adc_sock > zet017.dac_sock ? zet017.adc_sock : zet017.dac_sock;
width++;

adc_count = 0;
adc_time = 0;

dac_count = 0;
dac_time = 0;

while (1) {
    // Ожидаем сокеты АЦП или ЦАП
    FD_ZERO(&rfd);
    FD_SET(zet017.adc_sock, &rfd);
    FD_SET(zet017.dac_sock, &rfd);
    FD_ZERO(&wfd);
    FD_SET(zet017.dac_sock, &wfd);
    tv.tv_sec = 20;
    tv.tv_usec = 0;
    rc = select(width, &rfd, &wfd, 0, &tv);
    switch (rc) {
    case -1:
```

© ООО "ЭТМС" 1992-2024. Все права защищены

```
            fprintf(stderr, "Device socket error %d\n", rc);
            return -1;
        case 0:
            fprintf(stderr, "Device timed out\n");
            return -1;
        }

        if (FD_ISSET(zet017.dac_sock, &wfd)) {
            // Отправляем очередной пакет ЦАП
            rc = send(zet017.dac_sock, &dac_send, sizeof(dac_send), 0);
            if (rc != sizeof(dac_send)) {
                fprintf(stderr, "Send DAC error %d\n", rc);
                return -1;
            }
            dac_count += 512;
            if (dac_count >= dac_rate) {
                dac_count -= dac_rate;
                dac_time++;
                fprintf(stdout, "DAC time %u\n", dac_time);
            }
        }
        if (FD_ISSET(zet017.adc_sock, &rfd)) {
            // Принимаем очередной пакет ЦАП
            rc = recv(zet017.adc_sock, &adc_recv, sizeof(adc_recv), 0);
            if (rc != sizeof(adc_recv)) {
                fprintf(stderr, "Recv ADC error %d\n", rc);
                return -1;
            }
            if (zet017.cmd.info.TypeDataADC == 0)
                adc_count += 504 / 8; // 8 каналов, 16 битов
            else
                adc_count += 504 / 8 / 2; // 8 каналов, 32 бита

            if (adc_count >= adc_rate) {
                adc_count -= adc_rate;
                adc_time++;
                fprintf(stdout, "ADC time %u\n", adc_time);
            }
        }
        if (FD_ISSET(zet017.dac_sock, &rfd)) {
            // Вычитываем и игнорируем данные с ЦАП
            rc = recv(zet017.dac_sock, &dac_recv, sizeof(dac_recv), 0);
        }
    }

    zet017_disconnect(&zet017);
    return 0;
}
```

© ООО "ЭТМС" 1992-2024. Все права защищены

# О ZETLab

Настоящий документ является руководством оператора программного обеспечения **ZETLAB**, разработанного в ООО "Электронные технологии и метрологические системы" (ООО "ЭТМС"), г. Зеленоград.
В данном документе изложены требования к компьютеру для установки и работы с ПО **ZETLAB**, порядок установки, удаления и переустановки ПО **ZETLAB**, порядок настройки измерительных каналов, описания программ (интерфейс, порядок работы, настройка), а также схемы подключения датчиков к измерительным устройствам и примеры работы.

Программное обеспечение **ZETLAB** – это виртуальная лаборатория, предоставляющая пользователю мощные средства для визуализации, спектрального анализа, измерения электрических параметров, генерации, записи и воспроизведения сигналов.
Виртуальные приборы **ZETLAB** предназначены для решения задач измерения и управления в области сейсмики, вибрации, термометрии, тензометрии и т.д. Программы из состава **ZETLAB** обрабатывают сигналы, поступающие на входные каналы анализаторов спектра, сейсмостанций, тензостанций, плат АЦП/ЦАП, интеллектуальных датчиков и т.д.

Программы АЦП (раздел "Измерение") и программы ЦАП (раздел "Генерация сигналов") образуют базу виртуальной лаборатории **ZETLAB**, на которой строятся более сложные приборы (разделы "Метрология", "Автоматизация").

Набор поставляемых программ **ZETLAB** зависит от типа используемого прибора:

- **ZETLAB BASE** - программное обеспечение, поставляемое с платами АЦП/ЦАП,
- **ZETLAB ANALIZ** - программное обеспечение, поставляемое с анализаторами спектра,
- **ZETLAB VIBRO** - программное обеспечение, поставляемое с системой управления вибростендами,
- **ZETLAB TENZO** - программное обеспечение, поставляемое с тензостанциями,
- **ZETLAB SEISMO** - программное обеспечение, поставляемое с сейсмостанциями,
- **ZETLAB NOISE** - программное обеспечение, поставляемое с виброметром-шумомером,
- **ZETLAB SENSOR** - программное обеспечение, поставляемое с интеллектуальными датчиками ZETSENSOR (опционно).

В описании каждой программы в разделе "Поддерживаемое оборудование и входные данные" приведен перечень приборов, с которыми программа поставляется в базовой конфигурации или опционно

© ООО "ЭТМС" 1992-2024. Все права защищены

# Глава 1.Введение в ZETLab studio

**ZETLab Studio - идеальное средство для построения пользовательских систем измерений, автоматизации и управления!**

Создание собственных приборов? Теперь это стало еще доступнее, быстрее и проще!

Вы когда-нибудь задумывались над тем, сколько времени, сил и средств необходимо для создания нового измерительного прибора, например, вольтметра? Группе разработчиков надо провести за работой не один день для воплощения идеи в готовое изделие.

А как быть, если Вам необходимо к разработанному прибору добавить еще какую-нибудь полезную функцию? Опять "ваять" что-то новое, высиживая долгие дни над созданием чего-то неповторимого? Здесь мы с уверенностью можем сказать: "Нет!" Со средствами разработки ZETLab-Studio разработка, создание и модернизация измерительных приборов теперь превращается в простой и неутомительный процесс! Как "раз-два-три":

превращается в простой и неутомительный процесс! Как "раз-два-три":

1. **Формулируем задание**
2. **Используем средства разработки ZETLab Studio**
3. **Обрабатываем результаты**

Компоненты ***ZETLab Studio*** призваны максимально упростить процесс создания пользовательских приборов и приложений для измерения и обработки сигналов. Здесь есть все, что необходимо для построения мощных измерительно-вычислительных комплексов. Прилагая минимум усилий, средств и времени, Вы можете сконструировать поистине универсальный прибор, необходимый для решения именно Вашей задачи. Как быть с модернизацией? Все очень просто: добавляйте компоненты ZETLab-Studio в свои приложения - и получайте результаты обработки сигналов!

Программные и аппаратные средства ZETLab компьютерной автоматизации измерений, управления и моделирования находят большое применение в различных областях промышленности, научных исследованиях, а также в образовании. В составе аппаратных средств присутствуют практически все компоненты современных измерительно-управляющих комплексов: универсальные платы сбора и вывода аналоговых и цифровых сигналов, мультиметры, генераторы, распределенные измерительно-управляющие контроллеры, согласующие устройства на шинах PCI, USB и Ethernet и т.д. Концепция виртуальных приборов позволяет значительно расширить функциональность создаваемых испытательных и измерительных систем при одновременном сокращении трудозатрат на их разработку. ZETLab-Studio представляет собой набор встраиваемых компонент для быстрой и эффективной разработки

© ООО "ЭТМС" 1992-2024. Все права защищены

измерительных, контрольных и управляющих программ. Наш 20-летний опыт позволил создать удобный инструмент для создания высокопроизводительных систем обработки сигналов в реальном масштабе времени.

Представьте себе инструмент, прибор или систему, которые в точности соответствуют требованиям вашей задачи; инструмент, который собирает, анализирует, представляет данные и осуществляет управление именно необходимым Вам способом. С помощью ZETLab таким инструментом может стать обычный компьютер, стоящий у вас в лаборатории или на производстве, либо небольшая портативная машина типа Notebook, оснащенные дополнительными устройствами ввода информации. ZETLab Studio, подобно Labview - интегрированная среда разработчика для создания программ сбора, обработки данных и управления периферийными устройствами. Программирование осуществляется на любом объектно-ориентированном языке программирования MS Visual Basic, MS Visual C++, Borland Delphi, Borland C++ Builder* с использованием библиотечных элементов и готовых программ ZETLab. Сочетание широкоиспользуемого языка программирования и большого количества разнообразных компонент позволяет значительно сократить время разработки сложных систем при сохранении высокой скорости выполнения программ. Библиотеки современных алгоритмов обработки и анализа данных превращают ZETLab в универсальный инструмент создания интегрированных систем на базе персональных компьютеров.

* При программировании на MicroSoft C#, Borland Delphi и Borland C++ Builder обеспечивается ограниченная функциональность по причине особенностей данных языков программирования.

В комплект ZETLab входит более 100 различных готовых программ, компонент и библиотек, которые вы можете интегрировать в свои приложения. В основу пакета программ-приборов ZETLab заложен принцип одновременной работы многих программ. При использовании других пакетов, которые монопольно владеют ресурсами устройств ввода/вывода вам необходимо в одной программе осуществлять установку параметров ввода сигналов, вводить сигнал, обрабатывать его, создавать сигналы и отображать результаты. В пакете ZETLab Studio вам необходимо всего лишь подобрать набор необходимых инструментов и связать их в один проект. Таким образом, ZETLab Studio дает возможность избежать сложностей обычного "текстового" и "графического" программирования. Если вы ищете лучший способ программирования своих измерительных и управляющих систем без потери производительности, то ZETLab Studio - именно то, что вам нужно.

## 1.1.Разработка законченной системы

Как правило, любой программный пакет покрывает только один аспект поставленной задачи, но не решает все проблемы: сбор данных, их анализ, представление и управление. ZETLab предоставляет Вам все необходимые средства, объединенные единой методологией, поэтому вам вряд ли понадобится покидать среду ZETLab. В Вашем распоряжении имеется свыше 50 различных готовых программ - виртуальных приборов общего назначения: осциллографы, самописцы, вольтметры, частотомеры, узкополосные и долеоктавные анализаторы, коррелляторы, регистраторы, генераторы различных сигналов, фильтры верхних и нижних частот, устройство

© ООО "ЭТМС" 1992-2024. Все права защищены

цифрового ввода-вывода и специализированных приборов: измерители нелинейных искажений, измерители амплитудно-фазовых-частотных характеристик, генераторы с обратной связью, программы для модального и порядкового анализа.

На основе готовых приборов Вы собираете свой испытательный или измерительный стенд или систему управления производственным циклом или систему мониторинга. Нажатием на одну кнопку Вы сохраняете свой проект и можете теперь запускать его по мере необходимости. Все виртуальные приборы-программы работают как в реальном времени, так и в режиме обработки оцифрованных сигналов в виде файлов. Средства регистрации и воспроизведения сигналов позволяет записывать сигнал и обрабатывать его с применением различных алгоритмов. Это существенно минимизирует время разработки и отладки законченной системы.

Масштабируемость пакета ZETLab позволяет использовать одновременно в одном персональном компьютере несколько различных устройств ввода/вывода. Так для медленноменяющихся сигналов можете использовать многоканальные устройства АЦП, для быстроменяющихся - высокопроизводительные АЦП.

Связав в локальную сеть несколько компьютеров у вас есть возможность работать с одним измерительным трактом на нескольких компьютерах в реальном масштабе времени. Это особенно полезно при проведении учебного процесса. Также это широко используется в системах непрерывного контроля и мониторинга, когда один компьютер используется для непрерывной записи сигналов и выдачи предупреждающих сигналов, и другой компьютер используется для проведения диагностики контролируемых узлов.

Существенным достоинством пакета ZETLab является то, что многие виртуальные приборы в комплекте с устройствами ввода/вывода сертифицированы как средства измерения (СИ) и внесены в реестр СИ Российской Федерации.

Вы можете также написать собственные приложения, управляющие виртуальными приборами и собирающими от них результаты. В этом случае существенно упрощается метрологическая аттестация собранной таким образом системы. Для управления существующими программами используется компонент Unit. Описание компонента Unit и примеры его использования приводятся ниже. Все виртуальные приборы имеют возможность записать результаты в файл. В пакете ZETLab предусмотрено все для создания отчетов в Microsoft Excel и Word с минимальными затратами сил. Кроме того, вы имеете широкие возможности по манипулированию данными - запись/чтение с диска, передача по сети и печать на принтере или плоттере.

## 1.2.Построение собственного виртуального прибора

В ZETLab вы можете написать собственную программу виртуального прибора. Поскольку программное обеспечение ZETLab позволяет запускать и выполнять множество программ, то вам необходимо разделить свою задачу на несколько независимых программ. Программа виртуального прибора может быть написана на любом объектно-ориентированном языке программирования. В программу устанавливаются различные программные компоненты, отвечающие за ввод-вывод аналоговых и цифровых данных, графическое отображение двухмерных и трехмерных

© ООО "ЭТМС" 1992-2024. Все права защищены

графиков, X-Y графиков, графиков в полярных координатах, интегральных уровней, цифровых индикаторов. В программу также можно ставить стандартные компоненты объектно-ориентированного языка: кнопки, текстовые блоки, диалоги открытия файлов и многие другие. Большое количество учебников и примеров по существующим языкам программирования позволяет изучать их до любой степени детализации. Все компоненты самодокументированны, что позволяет достаточно быстро освоить необходимые команды. В результате компиляции вы получаете исполняемый код программы, что позволяет полностью использовать вычислительные возможности компьютера и позволяет распространять исполняемый рабочий файл программы без исходного текста программы. Полученную программу вы можете оформить в своем индивидуальном дизайне и использовать наравне с программами ZETLab.

## 1.3.Структура ZETLab studio

***ZETLab Studio*** - это интегрированный набор инструментов и библиотек классов на подобии Labview для Visual Studio.NET, Visual Studio 6.0, Borland Delphi, которые используются при решении задач измерений и автоматизации. ***ZETLab Studio*** существенно ускоряет процесс разработки приложений благодаря поддержке ActiveX- и .NET-объектов, объектно-ориентированных аппаратных измерительных интерфейсов, а также наличию дополнительных библиотек анализа данных, элементов управления, средств передачи данных по сети, мощных графических библиотек для представления данных.

Какие бы средства вы ни использовали для сбора данных - PCI, USB, Ethernet модули от 24 разрядов до 10 МГц - ***ZETLab Studio*** предоставляет Вам все средства разработки высокоуровневого интерфейса программирования приложений (API) в удобной вам среде разработки.

***ZETLab Studio*** предоставляет полный набор функций анализа и обработки данных измерений. С помощью ZETLab-Studio вы сможете воспользоваться широким набором таких средств анализа и обработки данных, как спектральный анализ, статистическая и цифровая обработка сигналов, фильтрация сигналов и быстрое преобразование Фурье. В силу того, что анализ выполняется вашим приложением сбора данных, вы получаете возможность сохранения в файл уже обработанных результатов измерений.

Теперь Вам не нужно тратить месяцы на создание профессиональных графических пользовательских интерфейсов для программ измерения и автоматизации. Для каждого типа измерений ***ZETLab Studio***, как и Labview предоставляет пользовательские элементы интерфейса, которые можно при необходимости размещать и совмещать произвольным образом для решения каждой конкретной задачи. Среди доступных элементов управления имеются различные кнопки, ручки, ползунки, светодиоды и измерительные приборы. Для представления результатов анализа имеются программы для представления данных в графическом виде, X-Y-представлении, двух и трехмерной графике, в полярных координатах, с аналоговым эффектом послесвечения электронно-лучевой трубки. Удобная система масштабирования графиков, плавное перемещение курсора, сохранение графических данных для отчетов в редакторах Microsoft Excel и Word позволяют быстро получать

© ООО "ЭТМС" 1992-2024. Все права защищены

необходимые результаты для последующей печати. Широкий набор элементов, имеющихся в ***ZETLab Studio***, позволяют вам осуществлять более информативное представление данных, по сравнению с традиционными приборами.

Вне зависимости от задачи, скорость выполнения программы является важнейшим фактором анализа данных. Библиотеки анализа используют максимум вычислительных возможностей Вашего компьютера. Виртуальные приборы оптимизированы для использования математического сопроцессора, MMX, SSE1, SSE2 и технологии HyperThreading. Кроме того, существуют специализированные библиотеки, использующие вычислительные возможности цифровых DSP-процессоров, установленных на платах АЦП ЦАП нашего предприятия.

Вы можете потратить часы, для того чтобы продумать, как ввести данные в Вашу программу. Еще больше времени уйдет на графику реального времени без мерцания и перерисовки. Мы предлагаем Вам передовую технологию программирования, которая позволяет существенно экономить время на программирование приложений обработки и отображения.

## Структура задач измерений и испытаний

Большинство задач испытаний, измерений или исследований можно представить в виде последовательности логических действий – накопления – обработки – представления результатов (рисунок 1.1.). В ***ZETLab Studio*** предусмотрены отдельные компоненты для каждой операции. Вы можете компоновать их в своей программе для создания своих приложений как в конструкторе Lego. Все эти кубики оптимизированы по быстродействию и надежности. Для любой задачи могут быть подобраны оптимальные аппаратные и программные средства, для того чтобы эффективно решить задачу:

- накопление данных;
- обработка и анализ данных;
- представление результатов;
- примеры программирования.

© ООО "ЭТМС" 1992-2024. Все права защищены




Рисунок 1.1

В отличие от многих других аналогичных систем, таких как Labview в ZETLab Studio не предполагается отдельного языка программирования и/или интерпретатора. Пользовательские программы создаются на языках программирования Visual Basic, Visual C, Delphi. Для поддержки пользовательских программ написаны следующие подпрограммы:

- **ZETServer** - сервер данных;
- **Grid** - графический компонент отображения зависимостей Y(x);
- **GridGL** - графический компонент отображения зависимостей Y(x) усовершенствованный;

© ООО "ЭТМС" 1992-2024. Все права защищены

- **Gramma** - графический компонент отображения двухмерной и трехмерной графики;
- **ZADC** - библиотека работы с модулями АЦП и ЦАП;
- **PlotterXY** - графический компонент отображения зависимостей X(t)-Y(t) в двухмерном и трехмерном виде;
- **Polar** - графический компонент отображения графиков в полярных координатах;
- **Unit** - компонент управления приборами (модуль управления и автоматизации);
- **DSP** - библиотека программ обработки сигналов с использованием возможности процессоров Pentium IV (MMX, SSE, SSE2).

Существует несколько уровней доступа к оцифрованным данным и получаемым результатам.

Пользовательские программы могут работать непосредственно с драйверами модулей АЦП ЦАП, при этом с одним драйвером могут работать несколько программ и в том числе программа сервера данных ZETServer. В этом случае программы получают данные от АЦП в целочисленном формате без всяких преобразований.

Пользовательские программы могут работать с сервером данных. Сервер поддерживает одновременную работу с несколькими драйверами различного типа и частоты опроса АЦП ЦАП. Сервер создает виртуальный канал ЦАП, как входной и поддерживает программы виртуальных каналов. При работе с программой "Регистратор" в режиме чтения оцифрованных данных из файлов, сервер создает каналы по количеству существующих файлов. Т.е. программа пользователя, написанная для работы с сервером, может работать без всякой адаптации с различными модулями АЦП ЦАП, с реальными данными, поступающими от модулей АЦП в реальном масштабе времени, с оцифрованными данными, записанными в файлы данных, с данными, получаемыми в результате моделирования. При работе с сервером программы получают данные в формате числа с плавающей запятой с учетом всех коэффициентов усиления, поправочных коэффициентов в заданных единицах измерения – мВ, м/с2, Па, мА и пр.

Пользовательские программы могут создавать виртуальные каналы. Самым простым примером программы, работающей с виртуальными каналами, виртуальных каналов является программа "Фильтрация сигналов". Эта программа в реальном времени производит фильтрацию сигнала, и результирующий сигнал записывает в виртуальный канал. Виртуальный канал создается на уровне сервера и поддерживается им же. Все программы-приборы могут обрабатывать дополнительный виртуальный канал.

Пользовательские программы при помощи программного модуля управления и автоматизации Unit могут запускать программы-приборы, устанавливать в этих программах различные параметры обработки и считывать текущие показания у программ-приборов.

## Накопление данных

Ввод-вывод аналоговых и цифровых сигналов производится через сервер данных SRV. Сервер спроектирован в соответствии с требованиями

© ООО "ЭТМС" 1992-2024. Все права защищены

общепромышленного стандарта для **SCADA** систем – OPC. Сервер осуществляет подключение к драйверам устройств, синхронизацию потоков данных от различных устройств ввода-вывода. Сервер данных обеспечивает одновременное подключение нескольких различных типов устройств:

- 8-канальный модуль АЦП 16/100, 3-канальный модуль ЦАП 16/200, контроллер KADSP/PCI, усилитель заряда для акселерометров ПУ 8/10. Частота дискретизации АЦП 100 кГц, ЦАП 2000 кГц, количество двоичных разрядов АЦП – 16, ЦАП – 16. интерфейс – шина PCI.

- 8-канальный модуль АЦП APC 216, 3-канальный модуль ЦАП CAP 316, контроллер KADSP/PCI, усилитель для акселерометров с интерфейсом ICP. Частота дискретизации АЦП 200 кГц, ЦАП 2000 кГц, количество двоичных разрядов АЦП – 16, ЦАП – 16, интерфейс – шина PCI, механизм автоопределения и диагностики соединений.

- 2-канальный модуль АЦП 16/1000, 3-канальный модуль ЦАП 16/2000, контроллер KADSP/PCI, усилитель заряда для акселерометров ПУ 8/10. Частота дискретизации АЦП 1 МГц. ЦАП 2000 кГц, количество двоичных разрядов АЦП – 16, ЦАП – 16, интерфейс – шина PCI.

- 2-канальный модуль АЦП 16/1000, 3-канальный модуль ЦАП 16/2000, контроллер KADSP/PDP, усилитель заряда для акселерометров ПУ 8/10. Частота дискретизации АЦП 1 МГц. ЦАП 2000 кГц, количество двоичных разрядов АЦП – 16, ЦАП – 16, интерфейс – шина PCI.

- 1-канальный модуль АЦП 14/2. 1 канал АЦП, 1 канал ЦАП. Частота дискретизации АЦП 10 МГц, ЦАП 10 МГц, количество двоичных разрядов АЦП – 14, ЦАП – 16. Интерфейс – шина PCI.

- 2-канальный модуль АЦП 24/4. 2 канала АЦП, 2 канала ЦАП. Частота дискретизации АЦП 1 кГц, ЦАП 8 кГц, количество двоичных разрядов АЦП – 24, ЦАП – 18. Интерфейс – шина PCI.

- 32-канальный модуль АЦП 14/32. 32 канала АЦП, 1 канал ЦАП. Частота дискретизации АЦП 400 кГц, ЦАП 400 кГц, количество двоичных разрядов АЦП – 14, ЦАП – 12. Интерфейс – шина PCI.

- 8-канальный модуль АЦП 16/8. 8 каналов АЦП, 1 канал ЦАП. Частота дискретизации АЦП 48 кГц по каждому каналу, ЦАП 48 кГц, количество двоичных разрядов АЦП – 16, ЦАП – 16. Интерфейс – шина PCI.

- 8-канальный модуль АЦП APC 216, 3-канальный модуль ЦАП CAP 316, контроллер KADSP/USB, усилитель для акселерометров с интерфейсом ICP. Частота дискретизации АЦП 200 кГц, ЦАП 2000 кГц, количество двоичных

© ООО "ЭТМС" 1992-2024. Все права защищены

разрядов АЦП – 16, ЦАП – 16, интерфейс – шина USB 2.0, механизм автоопределения и диагностики соединений.

- 2-канальный модуль АЦП АСРВ. 2 канала АЦП, 1 канал ЦАП. Частота дискретизации АЦП 500 кГц, ЦАП 500 кГц, количество двоичных разрядов АЦП – 16, ЦАП – 16, интерфейс – шина USB 2.0.

- 16-канальный модуль АЦП 16/16 – SIGMA/USB. 16 канала АЦП, 2 канала ЦАП. Частота дискретизации АЦП 500 кГц, ЦАП 500 кГц, количество двоичных разрядов АЦП – 16, ЦАП – 16. Цифровой ввод-вывод – 14 разрядов Интерфейс – шина USB 2.0 high-speed.

- 8-канальный модуль АЦП – анализатор спектра А-17 (ADC8500).

- 1-канальный шумомер-виброметр Delta.

- тип платы Осциллографа 20 МГц

- тип модуля анализатора ZET017U8, A17U8 (Аналоговый вход: 8, Суммарная частота преобразования 400 кГц, Количество разрядов АЦП - 16.)

- тип модуля анализатора ZET017U4, A17U4 (Аналоговый вход: 4, Суммарная частота преобразования 200 кГц, Количество разрядов АЦП - 16.)

- тип модуля анализатора ZET017T8, A17T8 (Аналоговый вход: 8, Суммарная частота преобразования 400 кГц, Количество разрядов АЦП - 16.)

- тип модуля анализатора ZET017U2, A17U2 (Аналоговый вход: 2, Суммарная частота преобразования 100 кГц, Количество разрядов АЦП - 20.)

- тип модуля анализатора A19U2 (Аналоговый вход: 2, Суммарная частота преобразования 500 кГц, Количество разрядов АЦП - 20.)

- тип платы ZET210 (Аналоговый вход: 16 синфазных/8 дифференциальных, Суммарная частота преобразования 20 кГц, АЦП - 16 бит. ЦАП 16 бит, 2 канала)

- тип платы ZET220 (Аналоговый вход: 16 синфазных / 8 дифференциальных, Суммарная частота преобразования 8 кГц, АЦП - 24 бита. ЦАП - 16 бит, 2 канала)

- тип платы ZET230 (Аналоговый вход: 4 синфазных / 4 дифференциальных, Суммарная частота преобразования 100 кГц, АЦП - 24 бита. ЦАП - 24 бита, 4 канала.)

- тип платы ZET240, ZET048 (тип входных каналов: дифференциальное, Суммарная частота преобразования 80 кГц, АЦП - 24 бита, 4-32 канала.)

© ООО "ЭТМС" 1992-2024. Все права защищены

- тип платы USB Осциллограф ZET302 (Диапазон до 20 МГц, Эффективная частота до 500 Мвыб/с. 2 канала).

Пользовательская программа подключается к серверу данных при помощи одной команды. Одновременно к серверу может подключаться несколько пользовательских программ. Данные от аналогоцифровых преобразователей поступают в программу пользователя в плавающей запятой в заданных единицах измерения: в Вольтах, Паскалях, м/с^2. Единицы измерения задаются в программе «Редактирование файлов параметров». Программа пользователя может создавать виртуальные каналы. Виртуальные каналы идут наравне с физическими каналами и могут обрабатываться другими программами, также как и физические каналы. Программа пользователя также может создавать данные для цифро-аналогового преобразователя и передавать их через сервер. Сервер может работать в режиме реального времени и в режиме чтения оцифрованных данных. Причем пользовательская программа будет с одинаковым успехом работать и в реальном режиме и в режиме чтения данных из файла. При объединении нескольких компьютеров в одну локальную сеть можно объединить и потоки данных от серверов данных и таким образом реализовать распределенную систему обработки сигналов.

Для передачи результатов измерений и управляющих команд из программ виртуальных приборов в пользовательскую служит сервер команд UNIT. Использование сервера UNIT подразумевает связь между программами типа ведущий – ведомый.

## Обработка и анализ данных

Любая программа, связанная с измерениями, автоматизацией и управления должна обрабатывать оцифрованные аналоговые данные и цифровые данные. Для упрощения работы с такими данными используется библиотека обработки сигналов в виде DLL.

Библиотека обработки сигналов включает в себя программы работы с массивами данных и оптимизирована для процессоров Intel IV с системой команд MMX и SSE:

1. выделение и освобождение памяти для массивов данных. Данные могут быть представлены в виде:
    - действительных массивов целых знаковых 16-разрядных чисел,
    - действительных массивов плавающей запятой одинарной точности 32 разряда,
    - действительных массивов плавающей запятой двойной точности 64 разряда,
    - комплексных массивов целых знаковых 16-разрядных чисел,
    - комплексных массивов плавающей запятой одинарной точности 32 разряда,
    - комплексных массивов плавающей запятой двойной точности 64 разряда.
2. операции с массивами всех типов:
    - сложение массивов и сложение с константой,
    - вычитание массивов и вычитание константы,
    - умножение массивов и умножение на константу,

© ООО "ЭТМС" 1992-2024. Все права защищены

- деление массивов и деление на константу,
- заполнение массива константой,
- копирование массивов,
- нормализация массивов,
- ограничение значений массивов,
- модуль (абсолютное значение) массивов,
- квадратный корень массивов,
- возведение в квадрат массивов,
- логарифмирование значений массива,
- расчет экспоненты массива,
- арктангенс значений массива,
- логические операции с массивами – и – или – инверсия – отрицание - сдвиг,

3. расчет параметров массивов данных всех типов:
    - расчет максимального значения по массиву,
    - расчет минимального значения по массиву,
    - расчет среднего значения по массиву,
    - расчет стандартной девиации значений массива,
    - расчет интеграла по массиву.
4. изменение частоты дискретизации массива всех типов:
    - уменьшение частоты – прореживание данных,
    - увеличение частоты – интерполяция данных,
    - изменение частоты с фильтрацией.
5. расчет корреляции:
    - автокорреляция массива данных,
    - взаимная корреляция массивов данных.
6. преобразование типов данных:
    - преобразование действительные в комплексные,
    - выделение действительной и мнимой частей из комплексного массива,
    - выделение фазы и амплитуды из комплексного массива,
    - преобразование массива целых чисел в плавающую запятую,
    - преобразование массива плавающей запятой в целые числа,
    - преобразование из декартовых координат в полярные координаты и обратно.
7. генерация сигналов:
    - создание массива синусоидального, треугольного сигналов,
    - создание равномерного шума и шума с распределением Гаусса.
8. функции генерации оконных функций:
    - Хана, Ханнинга, Блэкмана, треугольного взвешивающих массивов.
9. быстрое и дискретное преобразование Фурье:

© ООО "ЭТМС" 1992-2024. Все права защищены

- прямое, обратное действительных и комплексных массивов,
- дискретное преобразование Фурье для заданной частоты,
- БПФ двух действительных сигналов.

10. фильтрация сигналов:
    - фильтрация с конечно-импульсной характеристикой,
    - проектирование КИХ фильтров ФНЧ, ФВЧ, полосовых, режекторных, с различными оконными функциями,
    - фильтрация с бесконечно-импульсной характеристикой,
    - медианный фильтр,
    - адаптивный фильтр наименьших квадратов.
11. свертка сигналов:
    - одномерная и двумерная свертка сигналов.
12. функции вейвлетного преобразования:
    1. разложение сигналов,
    2. восстановление сигналов.

## Представление результатов

Результаты обработки могут быть представлены в графическом виде. Все что вам надо сделать, это поместить на свою форму графический ActiveX элемент в нужном месте, придать ему необходимые свойства – цвета сетки, надписей, графиков, типы линий, количество отображаемых графиков, количество точек графика. Свойства можно устанавливать на этапе проектирования программы и на этапе выполнения программы. Затем в процессе работы программы, полученные результаты в виде массива необходимо передавать в графический элемент. Это делается одной командой. Графический элемент сам прорисовывает все графики без мерцания и “снега” на экране. В графических элементах реализовано масштабирование графиков по всем осям, передвижение курсора и отображение положения курсора. Для этого вам не надо писать ни единой строчки кода. Программа пользователя может считывать положение и выполнять какие-либо действия связанные с вашими требованиями.

Для отображения графиков в виде зависимости y=y(x), используйте компоненты *Grid* и *GridGL* (рисунки 1.2 и 1.3 соответственно).

© ООО "ЭТМС" 1992-2024. Все права защищены

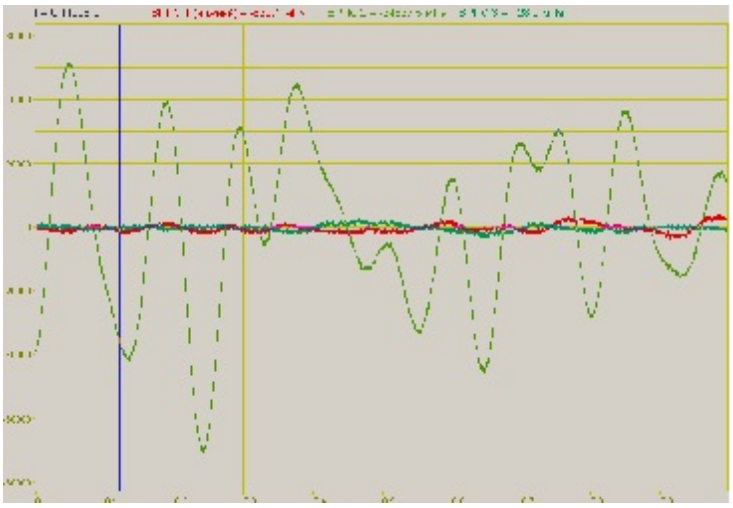


Рисунок 1.2

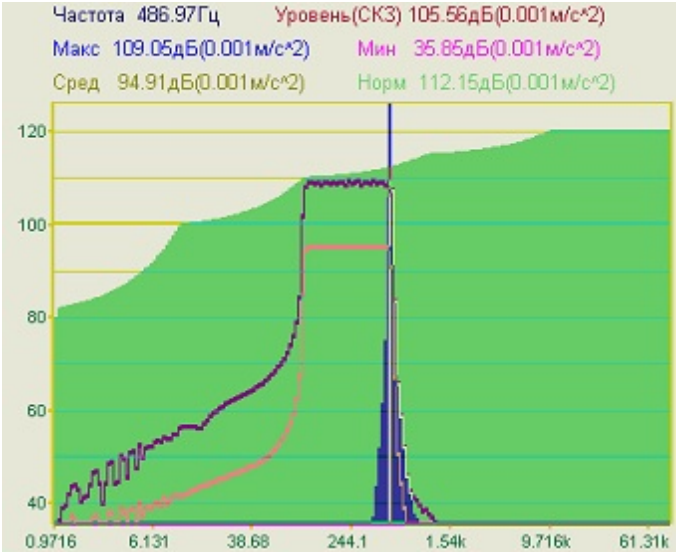



Рисунок 1.3

Для отображения графиков в виде параметрических зависимостей y=y(t), x=x(t) в двухмерном и трехмерном виде используйте компонент *PlotterXY* (рисунки 1.4 и 1.5 соответственно).

© ООО "ЭТМС" 1992-2024. Все права защищены

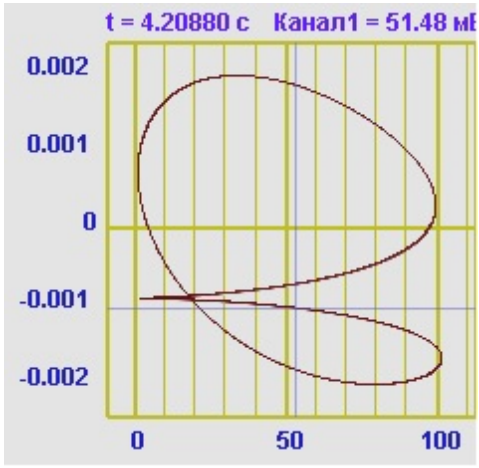



Рисунок 1.4

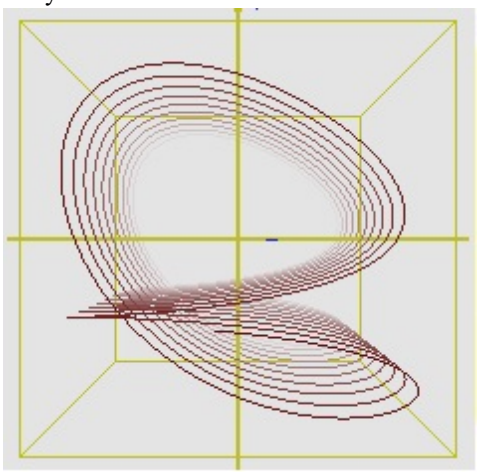
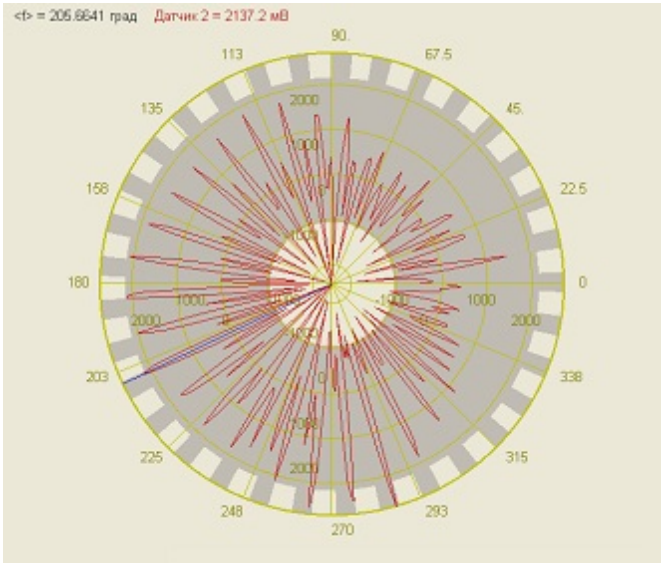


Рисунок 1.5

Для отображения графиков в полярных координатах используйте компонент *Polar* (рисунок 1.6).




© ООО "ЭТМС" 1992-2024. Все права защищены

Рисунок 1.6

Для отображения графиков в виде зависимости z=z(x,y) используйте компоненты *Gramma* и *GrammaGL* (рисунки 1.7 и 1.8).

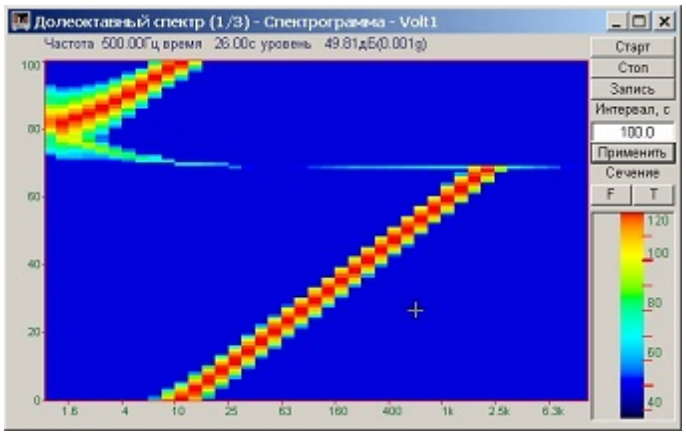



Рисунок 1.7

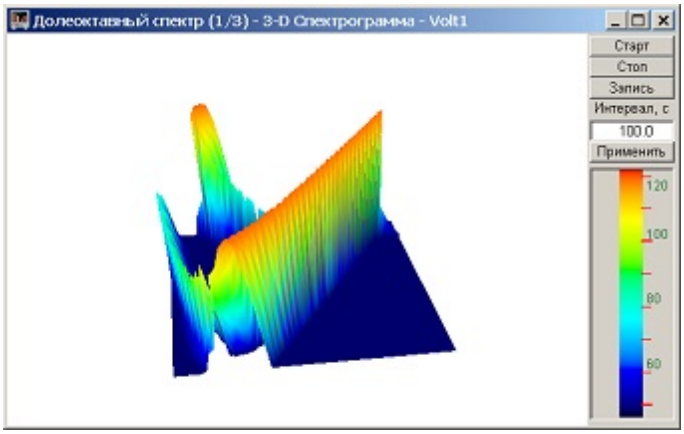



Рисунок 1.8

Для отображения интегрального уровня сигналов и перегрузки используйте компонент *Scale* (рисунок 1.9).

Для отображения текстовых сообщений и цифровых значений используйте компонент *TextDisp* (рисунок 1.10).



Рисунок 1.9




Рисунок 1.10

Пользовательские программы могут иметь множество параметров для настройки режимов работы, отображения, номера каналов, цвета графиков. Для сохранения

© ООО "ЭТМС" 1992-2024. Все права защищены

параметров программы предусмотрены библиотеки записи и чтения файлов параметров и сохранения параметров в проектах ZETLab.

# Глава 2.Введение в ПО ZETLab

Настоящий документ является руководством оператора программного обеспечения ZETLAB, разработанного в ООО "Электронные технологии и метрологические системы" (ООО "ЭТМС"), г. Зеленоград.
В данном документе изложены требования к компьютеру для установки и работы с ПО ZETLAB, порядок установки, удаления и переустановки ПО ZETLAB, порядок настройки измерительных каналов, описания программ (интерфейс, порядок работы, настройка), а также схемы подключения датчиков к измерительным устройствам и примеры работы.

## 2.1.О виртуальных приборах

**Концепция виртуальных приборов**

Времена, когда измерительные системы состояли из множества приборов и занимали целые лаборатории, уходят в прошлое. Мощность и доступность современных компьютеров позволяют использовать их для реализации алгоритмов, заложенных в традиционных приборах. Таким образом, роль измерительного устройства сводится к оцифровке сигналов, а их обработка и вывод результатов на экран осуществляется программными средствами:




*Структурная схема автономного измерительного устройства*



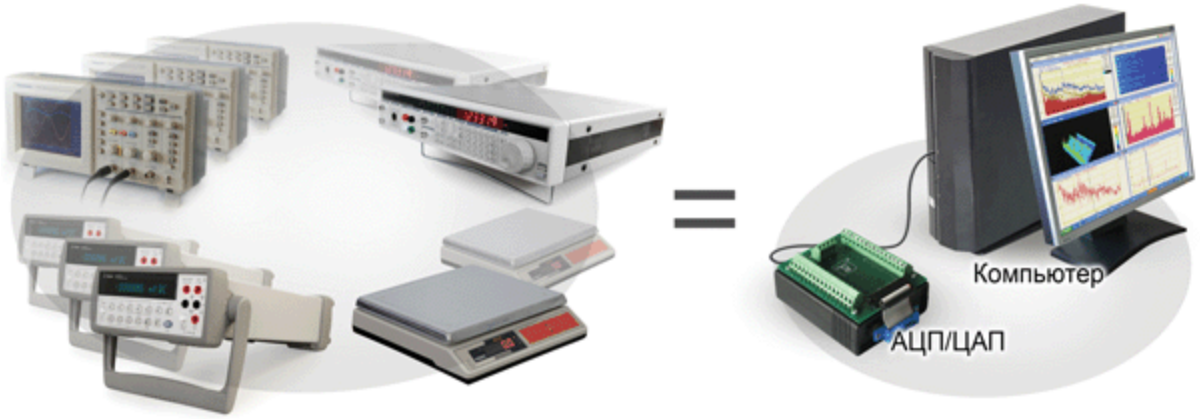

Структурная схема виртуального прибора

**Преимущества виртуальных приборов**

Поскольку функциональные характеристики системы, построенной на базе виртуальных приборов, определяются программным обеспечением, простая плата АЦП/ЦАП может быть одновременно и вольтметром, и осциллографом, и генератором,

© ООО "ЭТМС" 1992-2024. Все права защищены

и тензометром и каким угодно другим прибором, экономя рабочее пространство и средства пользователя.




*Замена реальных приборов виртуальными*

Применение модулей АЦП/ЦАП и ноутбуков с программным обеспечением, эквивалентными целой измерительной лаборатории, значительно расширило возможности измерений в полевых условиях.

Использование технологии беспроводной связи между оцифровщиком данных и компьютером позволяет проводить измерения на подвижных элементах конструкции. Например, комплект плата АЦП/ЦАП + модуль Wi-Fi или Bluetooth устанавливается на подвижной части (на крутящемся валу медленно вращающейся турбины, на автомобиле), а ПК устанавливается на расстояние до 500 м (в зависимости от поставляемой антенны) на неподвижный участок.

Оцифровка сигналов может проводиться в автономном режиме с записью на флэш-накопитель, а обработка – после перенесения записанных сигналов на ПК. Помимо собственно значений сигнала могут записываться, например, координаты расположения устройства. При использовании программ картографии можно воссоздать траекторию движения объекта в процессе эксперимента, наблюдая одновременно положение объекта и его параметры в каждый отсчет времени.

Разделение аппаратных и программных ресурсов позволило строить распределенные измерительные системы. В узлах сбора данных располагаются платы АЦП/ЦАП, которые подключаются к компьютеру по линиям Ethernet. С синхронизацией устройств по GPS и/или ГЛОНАСС, сигналы от всех плат поступают в компьютер единым потоком и обрабатываются одновременно.

Развитие концепции виртуальных приборов открыло новый этап в создании автоматизированных систем. Использование объектно-ориентированных прикладных программных средств позволяет создавать уникальные приложения с системой анализа и сценариями работы не только программистами, а, например, технологами.

Таким образом, можно выделить следующие преимущества виртуальных приборов:

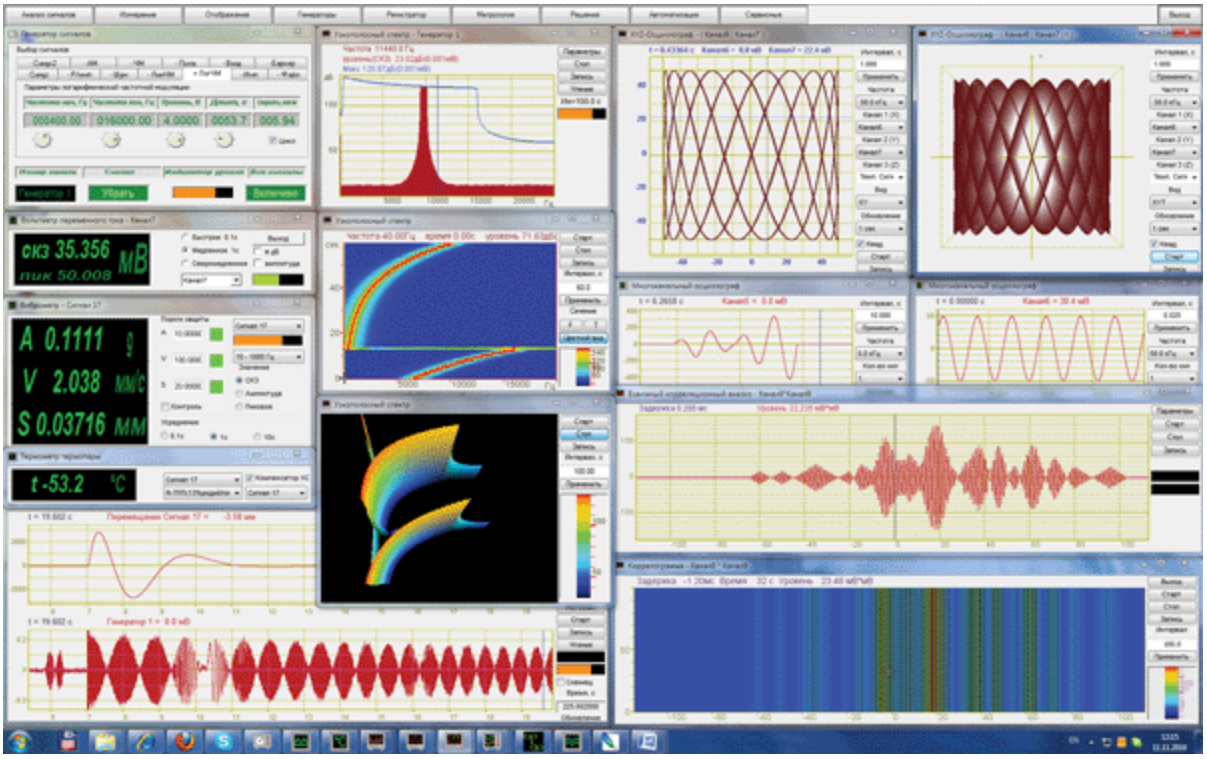
- снижение затрат экономия места в лаборатории
- параллельный анализ множества параметров
- расширение областей применения: в полевых условиях, измерения на подвижных элементах конструкций, в автономном режиме

© ООО "ЭТМС" 1992-2024. Все права защищены

- построение многоканальных распределенных систем
- упрощение создания автоматизированных систем.

**Виртуальные приборы ZETLAB**

Программное обеспечение ZETLAB представляет собой виртуальную лабораторию, состоящую более чем из 50 приборов. Все приборы разделены на группы по назначению: анализ, измерение, отображение, генераторы и т.д. К различным устройствам АЦП/ЦАП поставляются различные пакеты программы, соответствующих их назначению и характеристикам.




*Виртуальная лаборатория ZETLAB*

Также программы ZETLAB различаются по уровню сложности. Например, вольтметр имеет лишь три опции: переключатель времени усреднения, переход на представление результатов в дБ, переключатель измерения СКЗ или амплитудного значения. В то время как в генераторе реализовано 12 различных типов сигналов, начиная от синусоидальных и импульсных, и заканчивая кодами Баркера, каждый из которых имеет свои параметры.

Помимо базовых устройств, таких, как вольтметр, генератор и т.п., ZETLAB предоставляет мощные средства для обработки сигналов. Одна только программа Формула реализует десятки математических и измерительных функций, содержит более 20 видов фильтров и позволяет работать как с исходными, так и с обработанными сигналами.

В ZETLAB данные могут анализироваться параллельно несколькими программами, кроме того, программы могут использовать результат измерений друг

© ООО "ЭТМС" 1992-2024. Все права защищены

друга. Например, оцифрованный сигнал может быть сначала подвергнут фильтрации, после чего произведены вычисления, а конечный результат отображен на графике в двух- или трехмерном виде.

В программное обеспечение ZETLAB также входят решения, такие как измерение АЧХ с обратной связью, регулятор, обнаружитель событий, которые используют в своей работе результаты измерений одних программ (например, вольтметр, термометр), управляя при этом подключенными к ПК устройствами с помощью других программ (например, генератор, коммутационный блок).

# Глава 3.Установка и обновление ПО и драйверов

В данном разделе предоставлена информация о последних возможностях программных продуктов ZETLAB, известных проблемах и их возможных устранений, а также рекомендации и помощь по их использованию программных продуктов ZETLAB.

## 3.1.Требования к ПК

Программное обеспечение **ZETLAB** предназначено для использования на персональных компьютерах типа IBM PC Intel® Pentium®/Celeron®/ или совместимые с ними, работающих под управлением русскоязычной (локализованной) либо корректно русифицированной версии операционных систем:

1. Microsoft® Windows® XP с пакетом обновления не ниже SP3 (не поддерживается с 11.07.2014)
2. Microsoft® Windows® Vista с пакетом обновления SP1 (не поддерживается с 11.07.2014)
3. Microsoft® Windows® 7 32 разрядная с пакетом обновления SP1.
4. Microsoft® Windows® 7 64 разрядная с пакетом обновления SP1.
5. Microsoft® Windows® 8 32 разрядная.
6. Microsoft® Windows® 8 64 разрядная.
7. Microsoft® Windows® 8.1 32 разрядная.
8. Microsoft® Windows® 8.1 64 разрядная
9. Microsoft® Windows® 10 32 разрядная.
10. Microsoft® Windows® 10 64 разрядная.
11. Microsoft® Windows® Server 2003.
12. Microsoft® Windows® Server 2008 32 разрядная
13. Microsoft® Windows® Server 2008 64 разрядная с пакетом обновления SP2.
14. Microsoft® Windows® Server 2008 R2 с пакетом обновления SP1.
15. Microsoft® Windows® Server 2012 64 разрядная
16. Microsoft® Windows® Server 2012 R2 64 разр
17. Microsoft® Windows® Starter (без ограничения на количество запущенных программ).

Конфигурация компьютера для установки и запуска программного обеспечения **ZETLAB** и драйверов устройств:

- двухядерный процессор или более;
- тактовая частота процессора – не менее 1,6 ГГц;
- наличие интерфейса HighSpeed USB 2.0*;

© ООО "ЭТМС" 1992-2024. Все права защищены

- оперативная память – не менее 2 Гб;
- свободное место на жестком диске – не менее 20 Гб;
- видеокарта с 3D-графическим ускорителем, поддержкой OpenGL, DirectX, не менее 128 Мб памяти;
- разрешение экрана не менее 1280×1024;
- наличие манипулятора «мышь» или иного указательного устройства (сенсорный экран, трекбол (track ball), тачпад (TouchPad), графический планшет);
- наличие стандартной клавиатуры или иного устройства ввода (сенсорный экран, графический планшет);
- привод CD-ROM для установки программ.

*Приборы ZET поддерживают только интерфейс HighSpeed USB 2.0. Но приборы ZET можно подключать к ПК по шине USB 3.0, если контроллер данной шины обратно совместим с интерфейсом USB 2.0, как, например, контроллеры NEC.
Примечание: на данный момент при работе с контроллерами USB 3.0 производства Asmedia могут возникать проблемы (при установке драйвера возникает ошибка с кодом "10"). В таком случае рекомендуем использовать для подключения к ПК шину USB 2.0.

*Приборы ZET поддерживают только интерфейс HighSpeed USB 2.0. Но приборы ZET можно подключать к ПК по шине USB 3.0, если контроллер данной шины обратно совместим с интерфейсом USB 2.0, как, например, контроллеры NEC.
Примечание: на данный момент при работе с контроллерами USB 3.0 производства Asmedia могут возникать проблемы (при установке драйвера возникает ошибка с кодом "10"). В таком случае рекомендуем использовать для подключения к ПК шину USB 2.0.

**При использовании промышленных компьютеров для работы на них в ПО ZETLAB и ZETVIEW, мы рекомендуем использовать 64-разрядную версию ОС Windows**

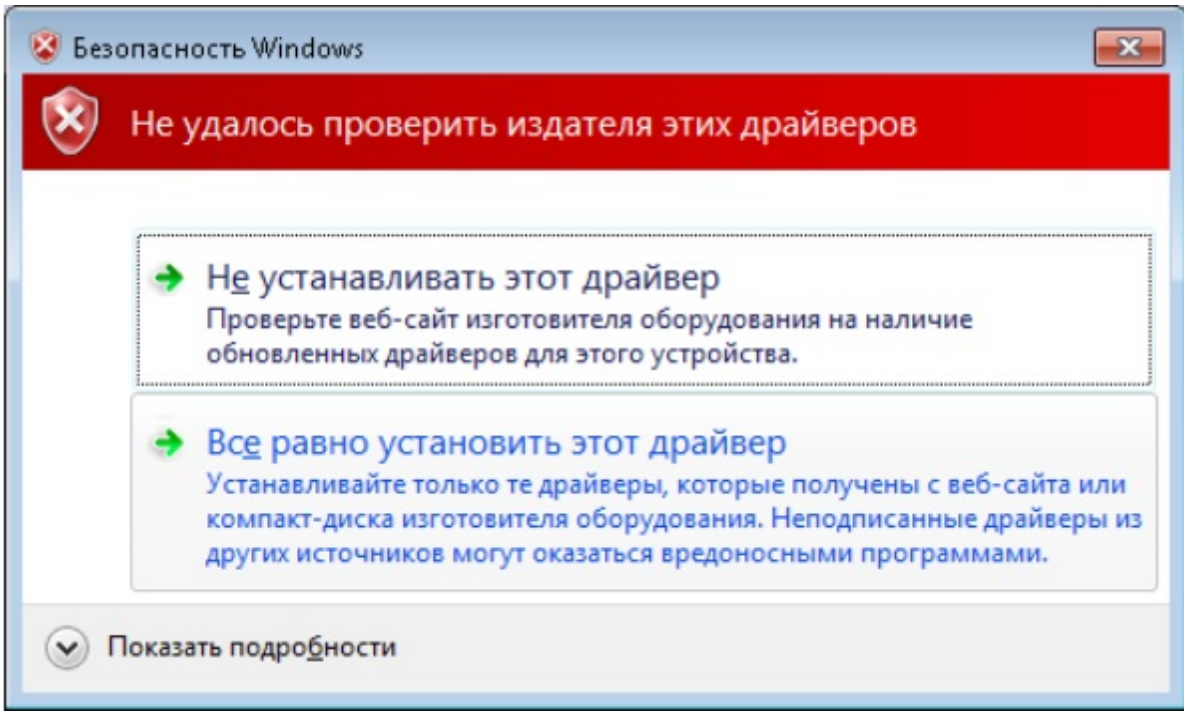
## 3.2.Проблемы с драйверами

**При использовании промышленных компьютеров для работы на них в ПО ZETLAB и ZETVIEW, мы рекомендуем использовать 64-разрядную версию ОС Windows**

### в ОС Windows 7

При установке ПО ZETLAB в Windows 7
В Windows 7 при установке драйверов может появляться предупреждение красного цвета представленное на рисунке ниже.

© ООО "ЭТМС" 1992-2024. Все права защищены




Это связано с тем, что драйверы подписываются по алгоритму SHA256, который не полностью поддерживается в Windows 7, если в ней давно не устанавливались обновления ОС. Сами драйверы исправны и подписаны правильно.
Обновление для системы безопасности Windows 7 для систем на базе 64-разрядных (x64) процессоров (KB3033929) можно скачать:

- на сайте **Microsoft;**
- с нашего файлового ресурса **https://file.zetlab.com/Utilities/Microsoft/Windows6.1-KB3033929-x64.msu**.

Можно также скачать и установить **бета версию ПО ZETLAB**. В этой версии необходимое обновление уже включено в состав дистрибутива.

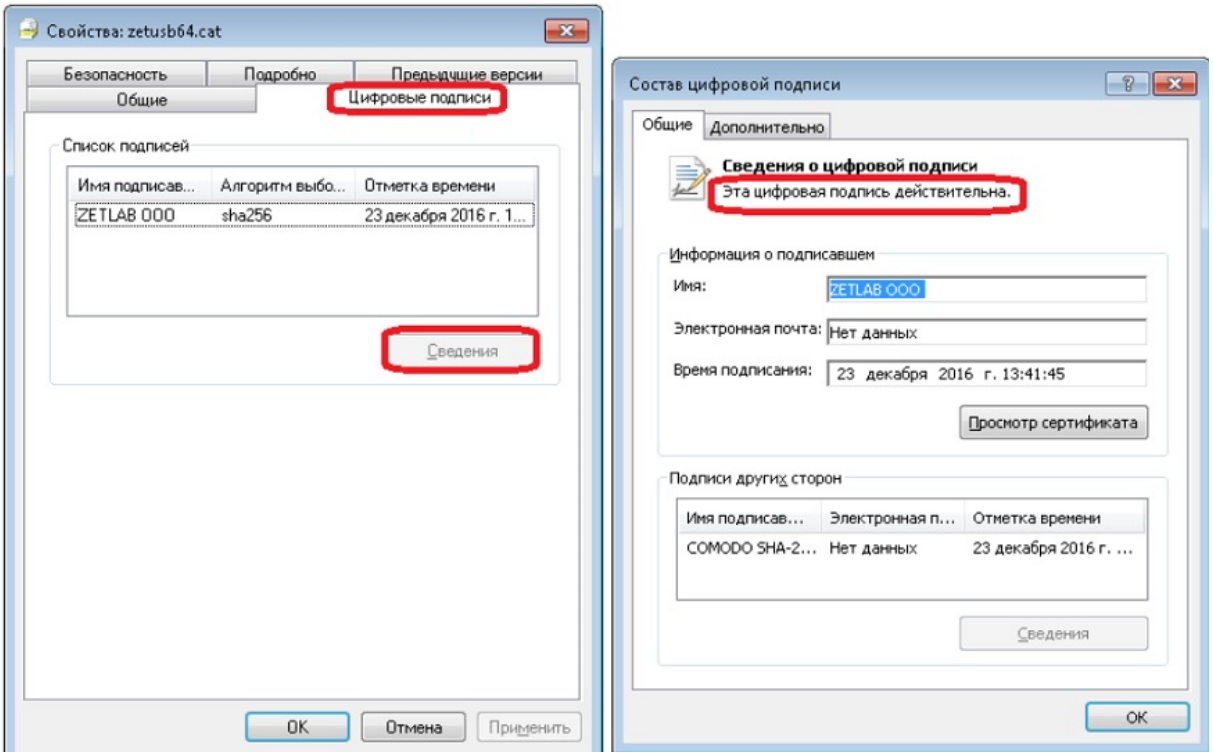
### в ОС Windows 10

При установке ПО ZETLAB в Windows 10

В новых версиях Windows 10 были ужесточены требования к цифровой подписи драйверов режима ядра. Если раньше достаточно было самостоятельно подписать их доверенным сертификатом, то теперь необходимо, чтобы подпись была проставлена корпорацией Microsoft.
Мы уже начали процесс сертификации драйверов в Microsoft, но это может занять некоторое время.
Для временного решения можно скачать и установить **предыдущую версию драйверов**, так как к сертификатам, выданным до 29 июля 2015 года, такие требования не предъявляются.

© ООО "ЭТМС" 1992-2024. Все права защищены

Перед установкой рекомендуется закрыть все программы ПО ZETLAB и отключить от ПК устройства ZETLAB.
Если у Вас возникли сомнения, то проверить цифровую подпись можно с помощью проводника, выбрав один из файлов (например, «C:ZETLabdriverszetusb64.cat» или «C:ZETLabdriversamd64Kdu8500.sys») и открыв его Свойства. Если продолжить установку драйверов (пункт «Все равно установить этот драйвер»), то они будут работать правильно, предупреждения возникать не будут. Но рекомендуется все же установить обновления для ОС Windows 7.




## 3.3.Устранение возможных неполадок ПО

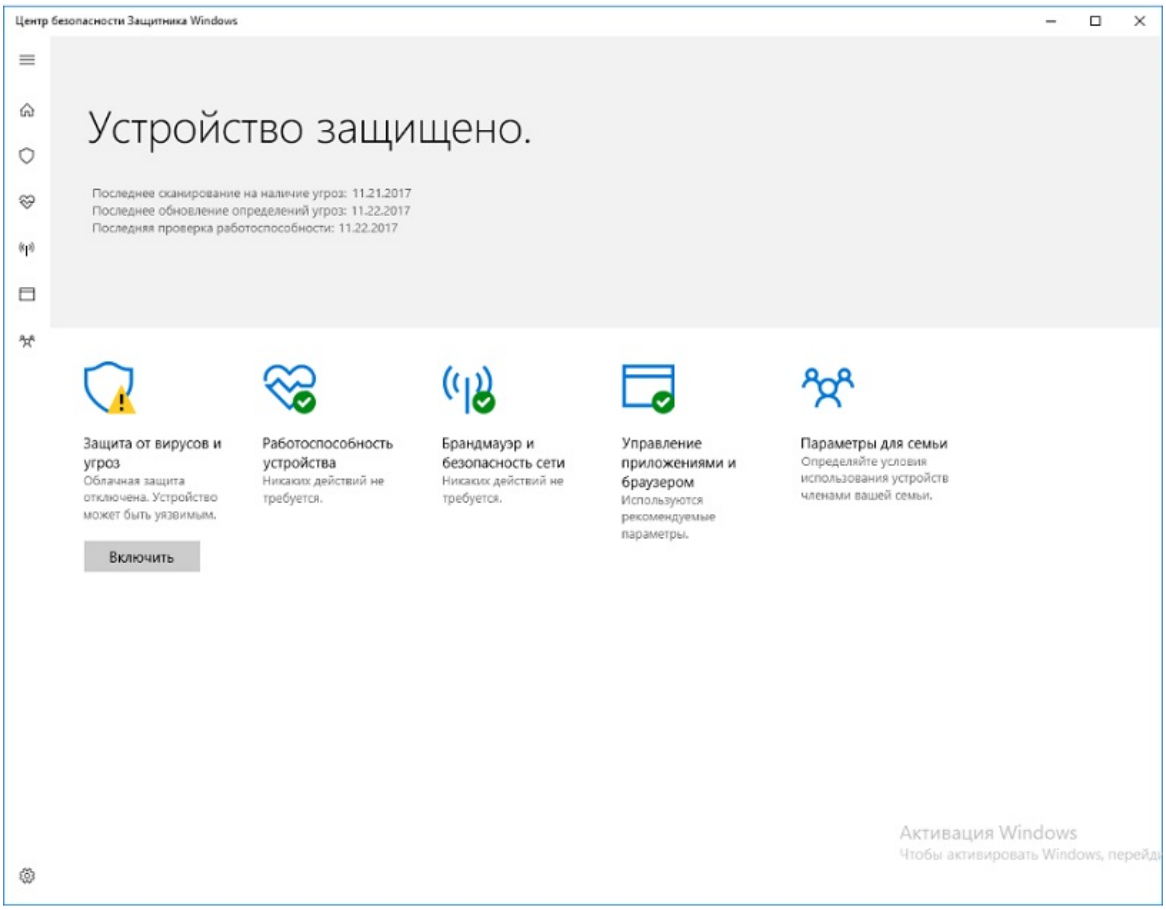
### Устранение возможных неполадок

при работе с программным обеспечением ZETLAB

Во время работы некоторых программ из состава ПО ZETLAB создаётся большое количество файлов, которые могут вызвать подозрения у антивируса. Он будет их проверять, что скажется на процессе создания файлов и работе программы. Для увеличения скорости сохранения графиков и работы программы, добавьте папку, куда производится сохранение, в исключение антивируса. Решение проблемы показано на примере «Windows Defender».

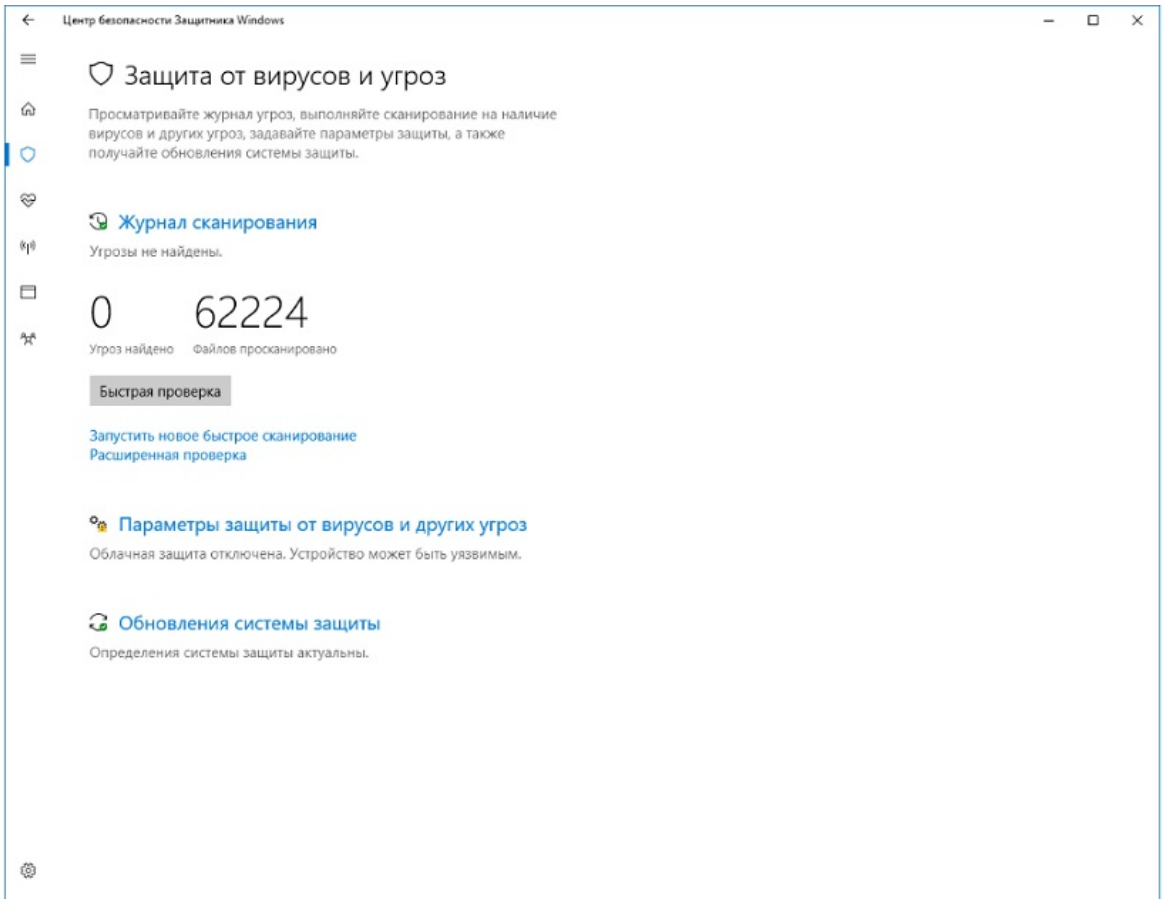
- Откройте программу «Windows Defender», и перейдите раздел «Защита от вирусов и угроз».

© ООО "ЭТМС" 1992-2024. Все права защищены



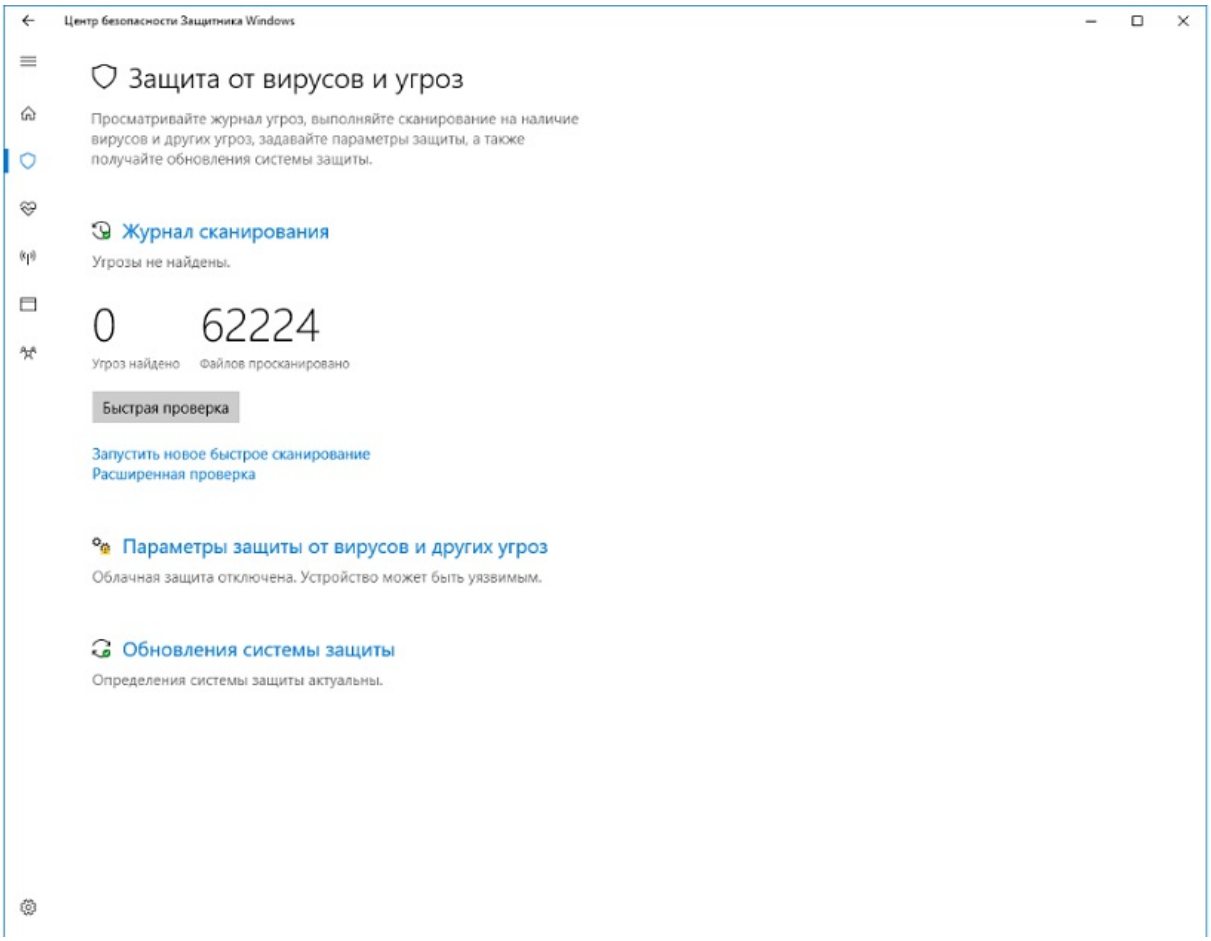

- Перейти в раздел «Параметры защиты от вирусов и других угроз».

© ООО "ЭТМС" 1992-2024. Все права защищены



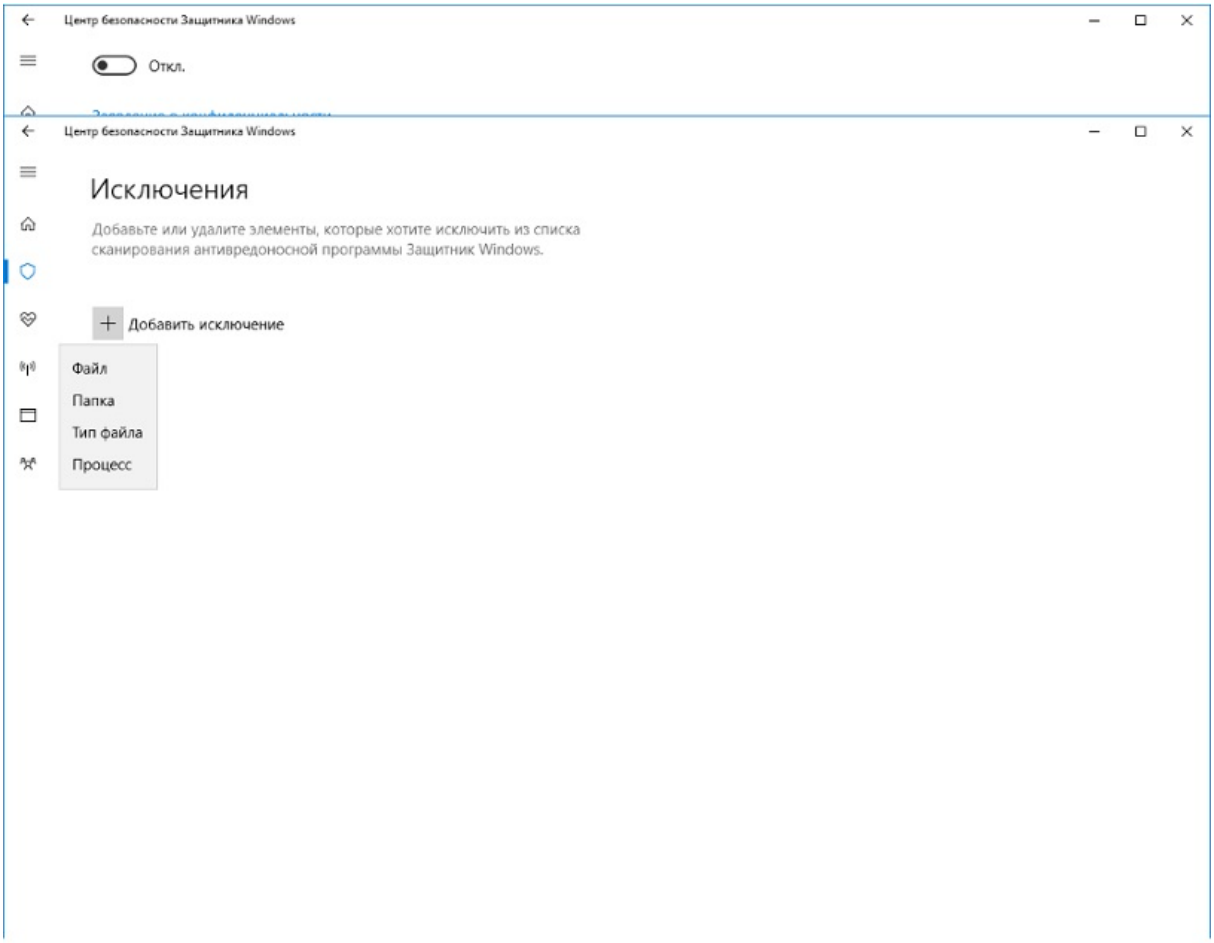

- Находим пункт «Исключения» и нажимаем кнопку «Добавление или удаление исключений».

© ООО "ЭТМС" 1992-2024. Все права защищены

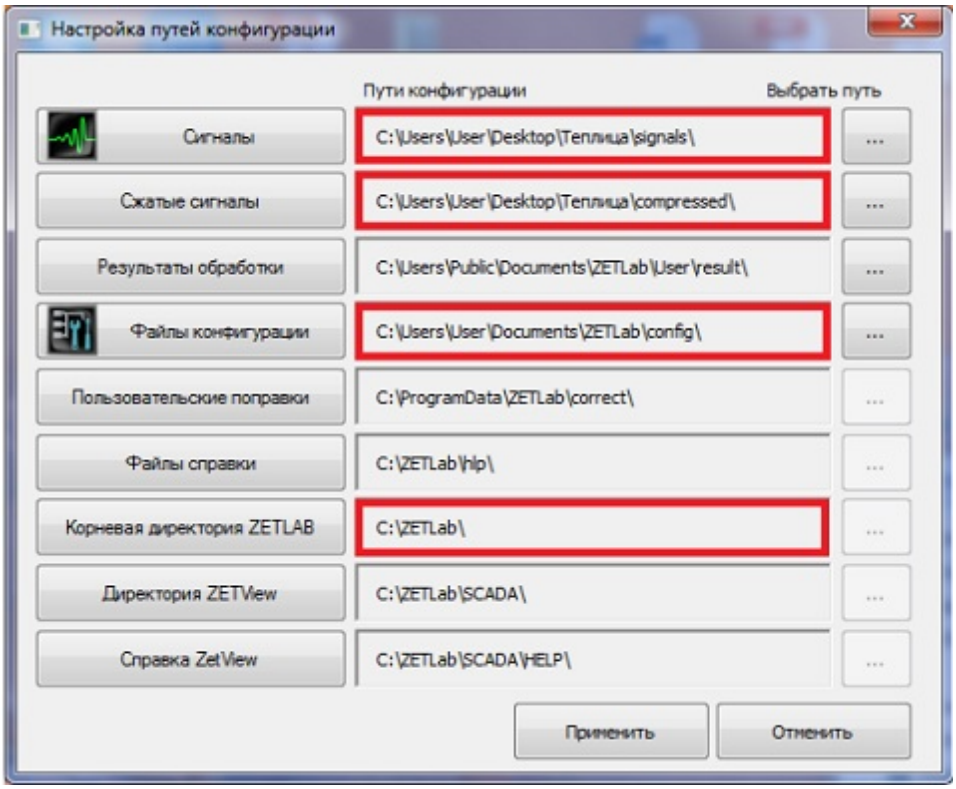



Нажимаем «Добавить в исключение» и в выпадающем списке выбираем «Папка» и добавляем папку куда сохраняются наши файлы. В исключение рекомендуется добавлять следующие папки: ZETLab, config, compressed, signals.

© ООО "ЭТМС" 1992-2024. Все права защищены




© ООО "ЭТМС" 1992-2024. Все права защищены




## 3.4.Установка ПО ZETLAB

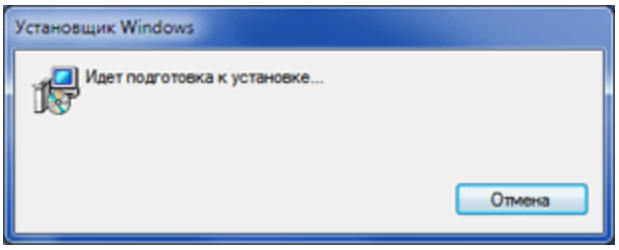
Установка программного обеспечения **ZETLAB** может производиться как при подключенном, так и при отключенном оборудовании ZET. В последнем случает после установки ПО и подключении оборудования будет произведено обновление драйверов к устройству.
Установка **ZETLAB** производится с установочного диска, поставляемого с приобретаемым ZET-устройством или, при обновлении **ZETLAB**, с помощью установочного файла (имеет расширение «.msi»).
При установке **ZETLAB** с установочного диска необходимо вставить диск в дисковод компьютера.

Дистрибутивы расположены на диске :\ZETLab setup\ZETLab.msi и установить ZETLab.msi.

При установке ZETLab с помощью файла «ZETLab.msi» необходимо запустить файл двойным кликом «мыши», при этом запустится программа установки ZETLab (рисунок 1.3.1).

В главном окне программы установки программного обеспечения ZETLab (рисунок 1.3.1) необходимо выбрать пункт «Далее»,

© ООО "ЭТМС" 1992-2024. Все права защищены




Подготовка к установке

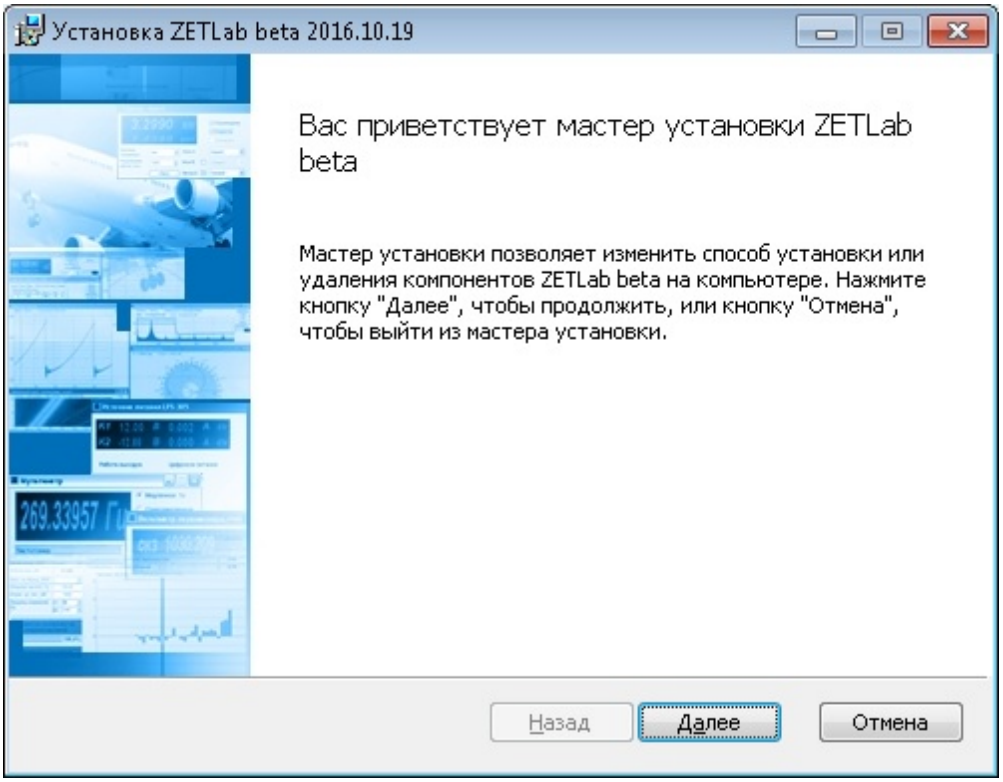
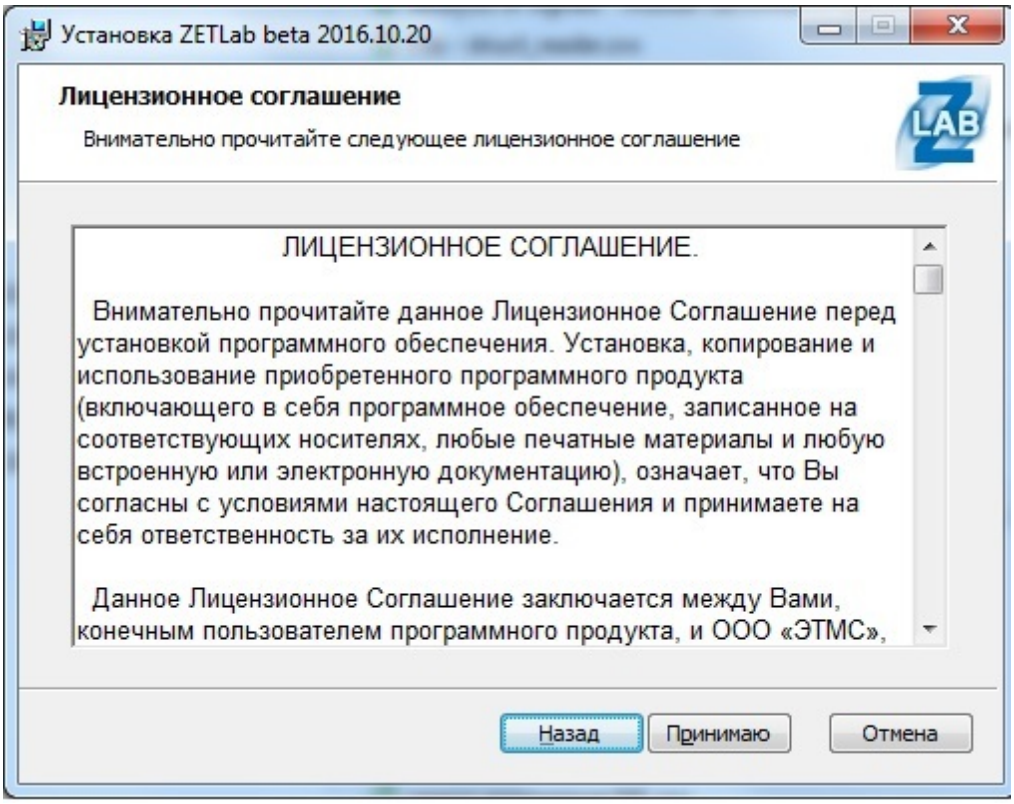



Рисунок 1.3.1.Установка ZETLab

Далее программа установки ZETLab предложит ознакомиться с лицензионным соглашением (рисунок 1.3.2). Лицензионное соглашение можно прочитать в окне программы установки. Без принятия условий лицензионного соглашения установка ZETLab будет прервана. При согласии с условиями соглашения необходимо выбрать кнопку «Принимаю» для продолжения установки ZETLab.

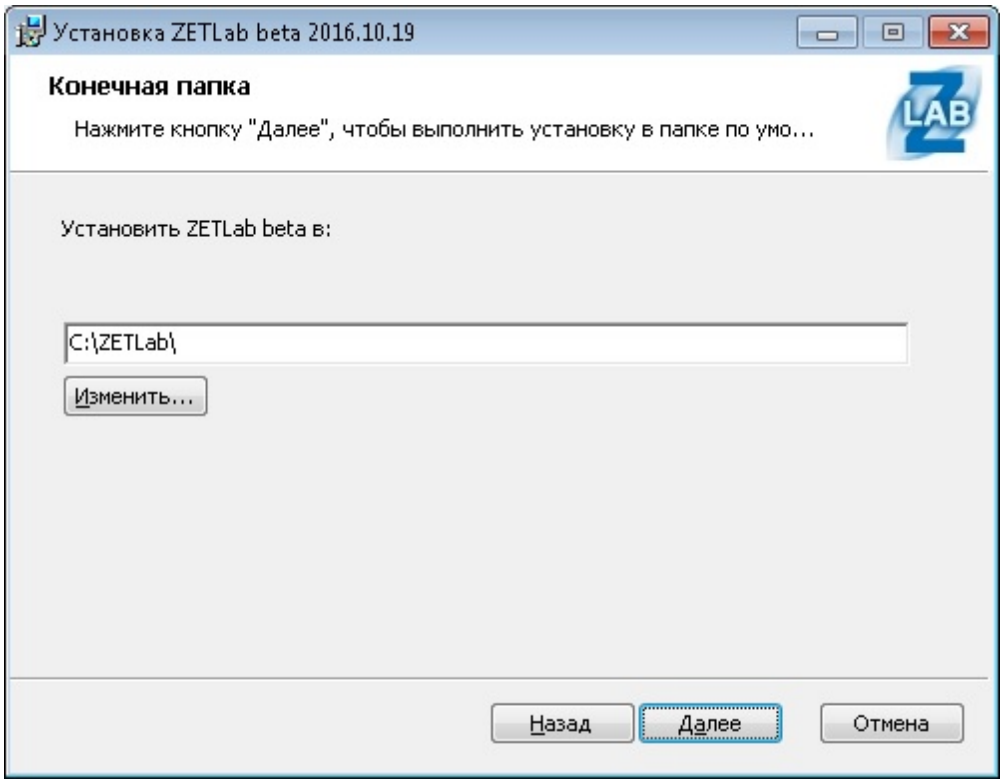
© ООО "ЭТМС" 1992-2024. Все права защищены




Главное окно установки программного обеспечения ZETLAB (рисунок 1.3.2)

Далее будет предложено выбрать директорию установки **ZETLAB**. По умолчанию программы **ZETLAB** устанавливаются в директорию **ZETLAB** (рисунок 1.3.2).

© ООО "ЭТМС" 1992-2024. Все права защищены

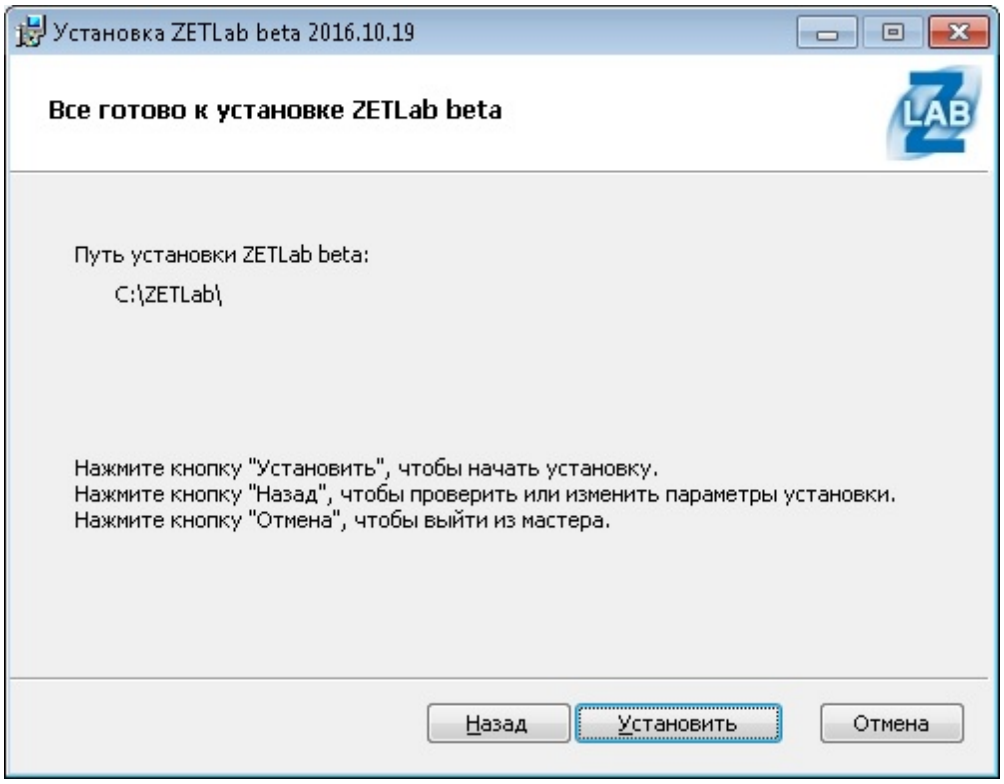



Выбор директории установки (рисунок 1.3.3)

Далее мастер установки выведет сообщение о готовности к установке. Для установки необходимо нажать кнопку "Установить".

Далее программа установки предложит установить ZETLab (рисунок 1.3.4). Дождитесь окончания установки.

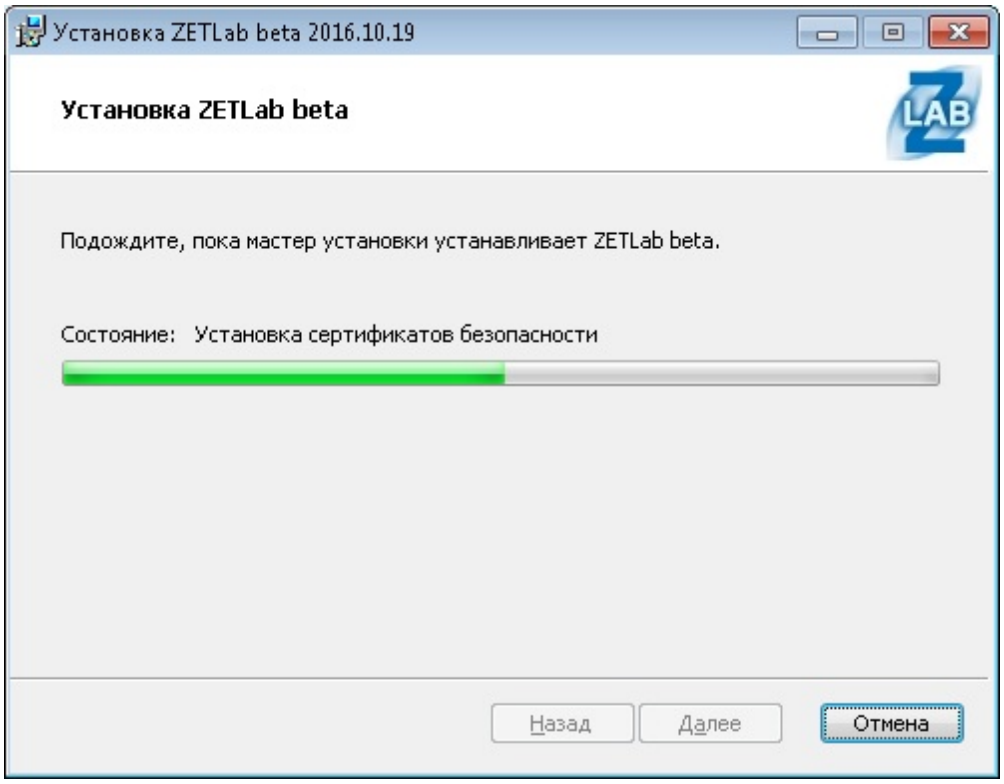
© ООО "ЭТМС" 1992-2024. Все права защищены




Запуск установки ZETLAB (рисунок 1.3.4)

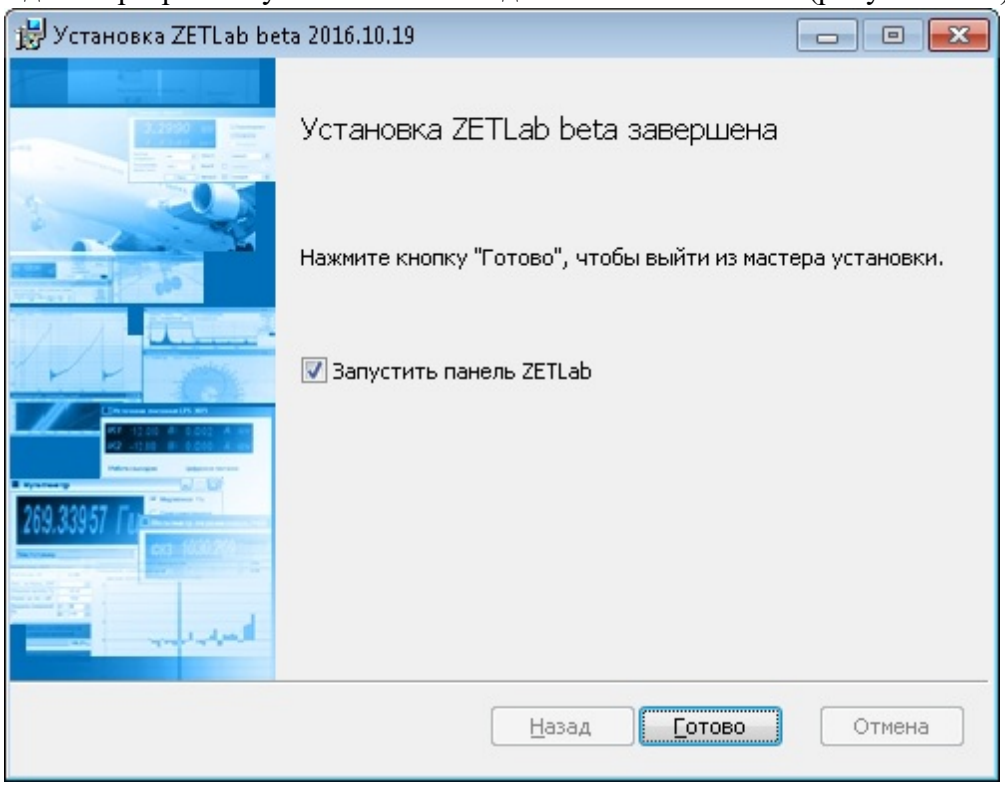
Установка **ZETLAB** займет несколько минут. По окончании установки станет активной кнопка "Далее", которую необходимо нажать для завершения установки (рисунок 1.3.5)

© ООО "ЭТМС" 1992-2024. Все права защищены




Установка ZETLAB ( (рисунок 1.3.5))

Для выхода из программы установки необходимо нажать "Готово" (рисунок 1.3.6).



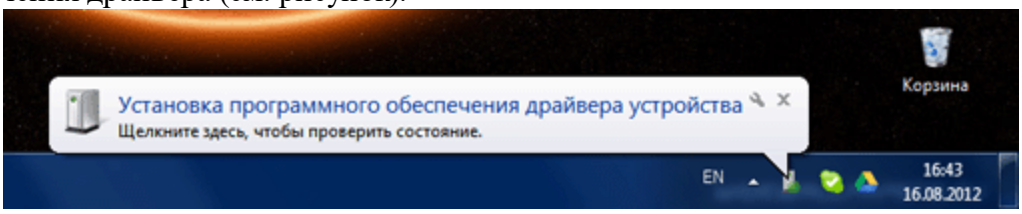

Завершение установки (рисунок 1.3.6)

© ООО "ЭТМС" 1992-2024. Все права защищены

Взведенный Флаг Запустить панель **ZETLab** позволяет сразу после установки Setup запустить ZETlab без запуска с ярлычка программы.

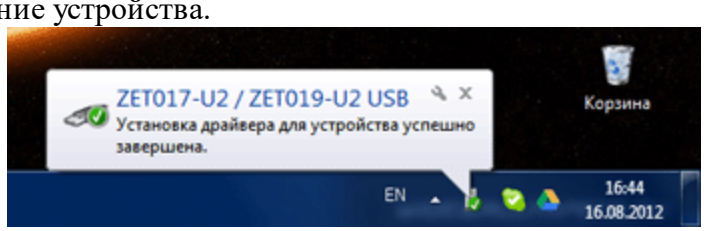
## Установка драйверов

В процессе установки ПО **ZETLAB** также были скопированы на компьютер драйвера на оборудование ZET (анализаторы спектра, платы АЦП/ЦАП, тензостанции, сейсмостанции, интеллектуальные датчики и др.). Если при установки ПО был подключен какой-либо ZET прибор, драйвера на него были автоматически установлены и прибор готов к работе по завершению процесса установки.
Если прибор во время установки ПО не был подключен, то при первом подключении прибора к ПК в системном трее появится сообщение об установке программного обеспечения драйвера (см. рисунок).




Установка драйвера при подключении прибора к ПК

Установка драйвера может занять несколько минут. По окончании установки драйвера появится сообщение об успешном завершении, при этом в заголовке сообщения отобразится название устройства.



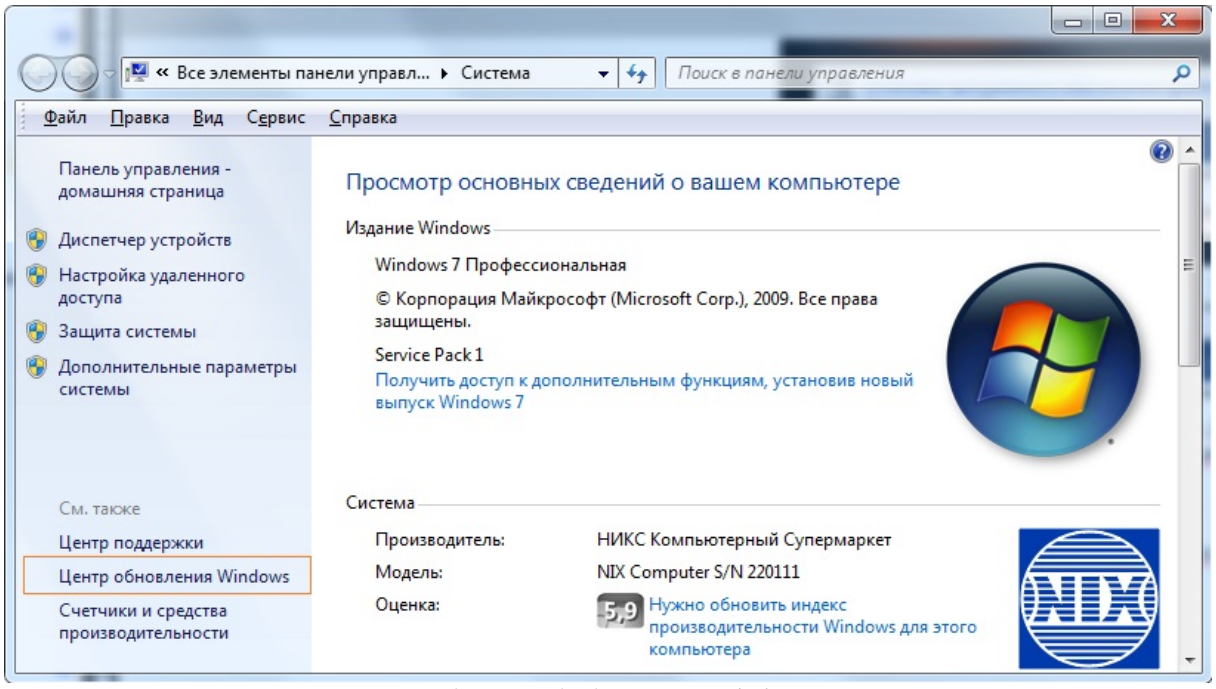

Сообщении об успешной установке драйвера

Если установка драйверов не была произведена автоматически (такое могло произойти, например, при обновлении **ZETLAB**, если в процессе удаления работали какие-либо из программ), то обновить драйвера можно самостоятельно. Указания по обновлению драйверов приведены в подразделе Обновление драйверов 80 настоящего документа.

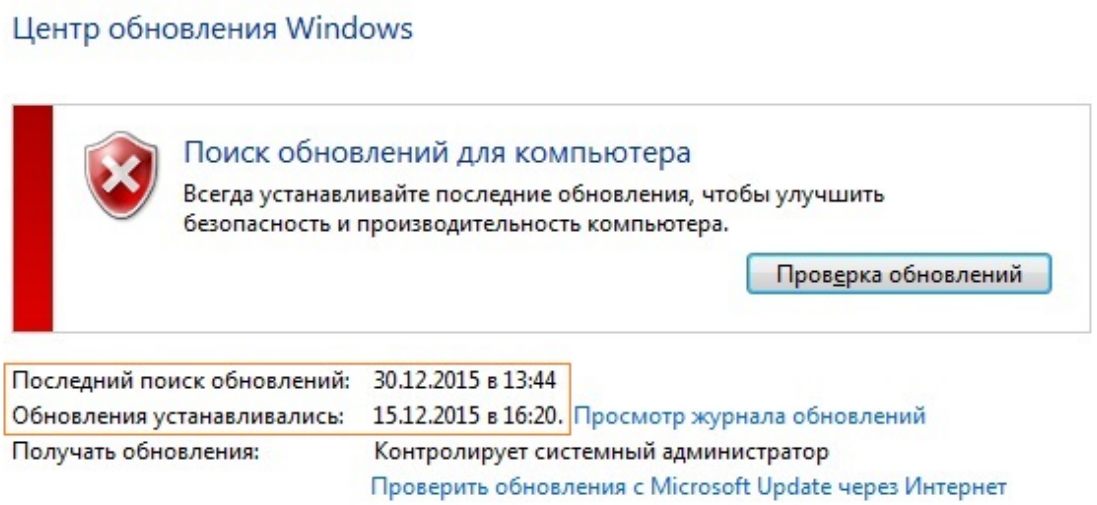
## Примечание:

Отличительные особенности установки драйверов на компьютеры с Windows 7, на которых не идет поиск и не устанавливаются обновления Windows.
Данную информацию можно получить Пуск=>Компьютер=>Свойства

© ООО "ЭТМС" 1992-2024. Все права защищены



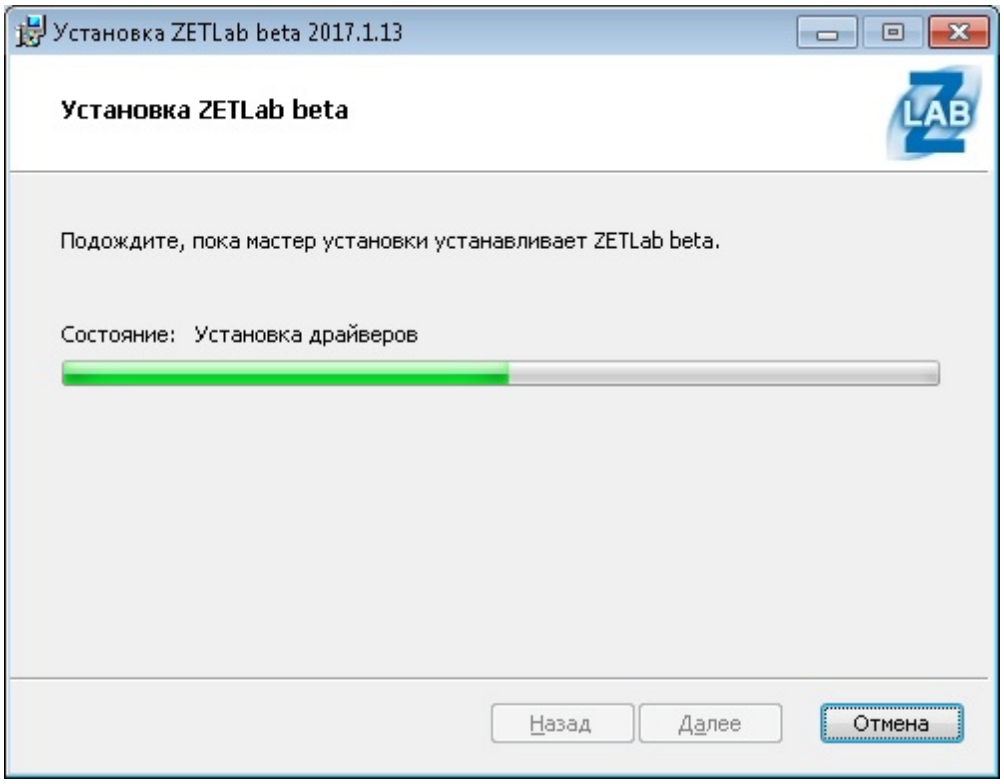

Сообщение об обновлении Windows



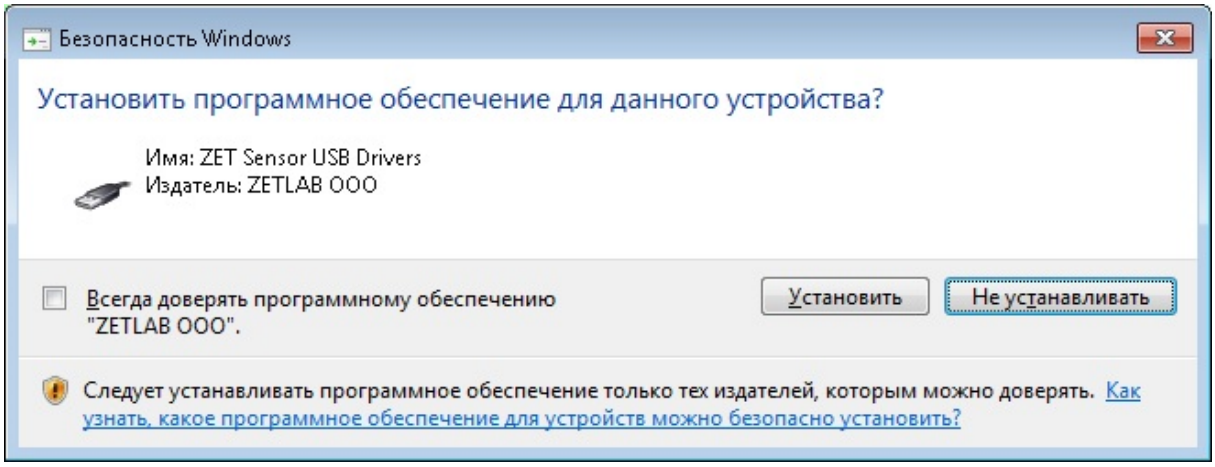

Сообщение об устаревших обновлениях Windows

При установке ZETLab_beta.msi или ZETLab.msi от 01.1.2017 в Windows 7 может появиться сообщение.

© ООО "ЭТМС" 1992-2024. Все права защищены




Сообщение об установки setup в данном месте прерывается и выводится следующее сообщение




Сообщение об установке драйверов на компьютер

Установка флага "Всегда доверять программному обеспечению ZETLAB ООО" никакого влияния на установку драйверов не оказывает, сообщение появляется снова при установке другого драйвера.

Необходимо нажать кнопку "Установить", и установка драйверов для приборов продолжится.

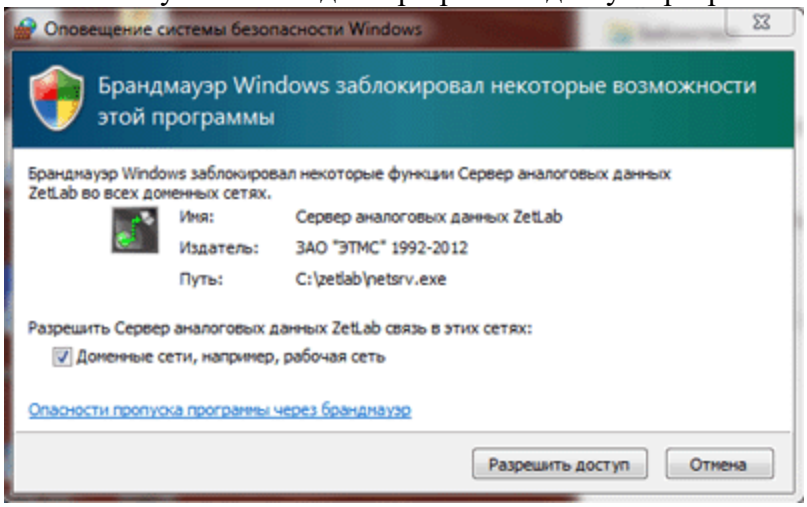
© ООО "ЭТМС" 1992-2024. Все права защищены

Для радикального избавления от данных сообщений при установке драйверов, необходимо поставить обновления Windows на компьютере или скачать и установить хотфикс kb2921916:

**https://support.microsoft.com/en-us/kb/2921916**

## Доступ программ ZETLAB к сети

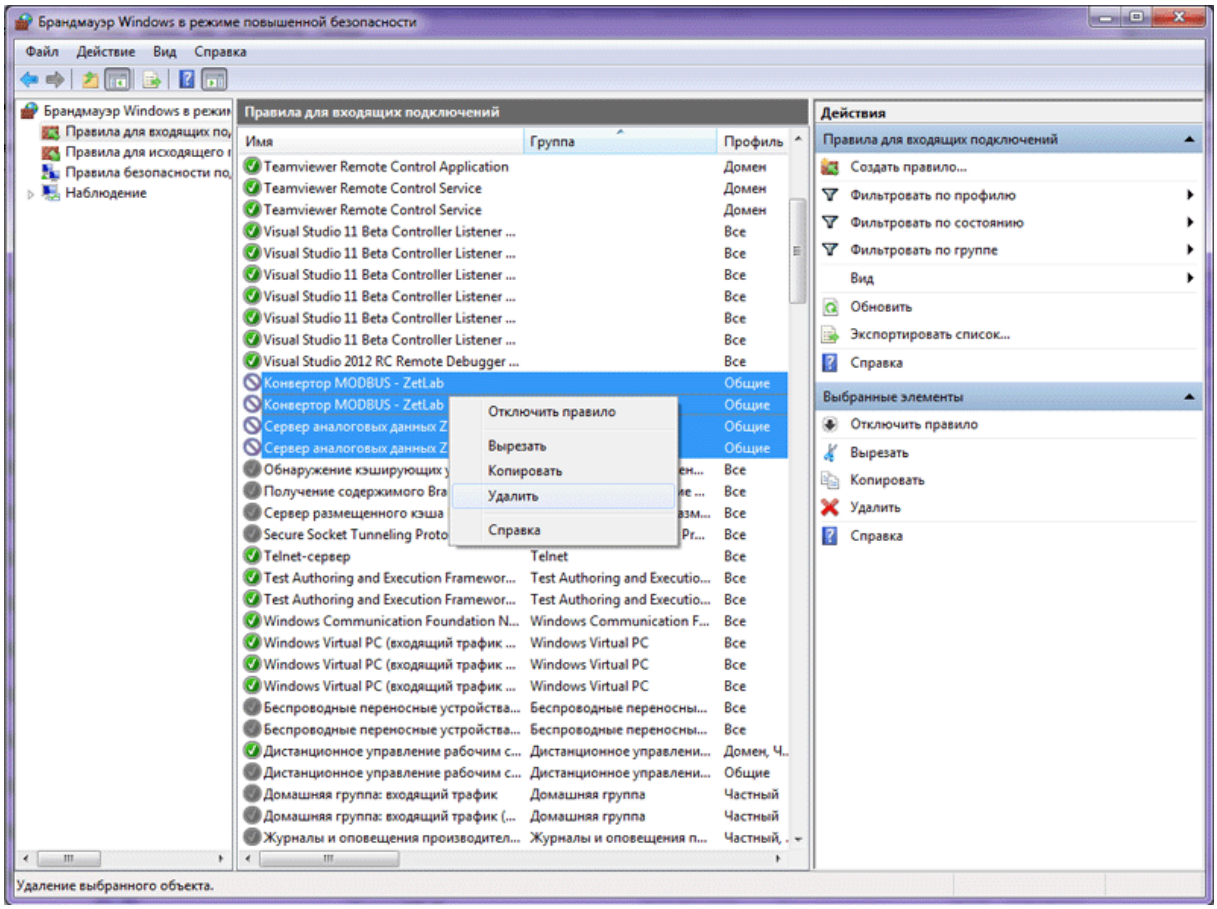
При установке может открыться предупреждение системы безопасности (пример - рисунок ниже). В таком случае необходимо разрешить доступ программы к сети.




Предупреждение системы безопасности

Если при установке **ZETLAB** на запрос о предоставлении сети был выбран пункт "Отмена", необходимо проделать следующее: "Панель управления - Брандмауэр - Дополнительные параметры" и удалить записи как в примере на рисунке ниже.

© ООО "ЭТМС" 1992-2024. Все права защищены



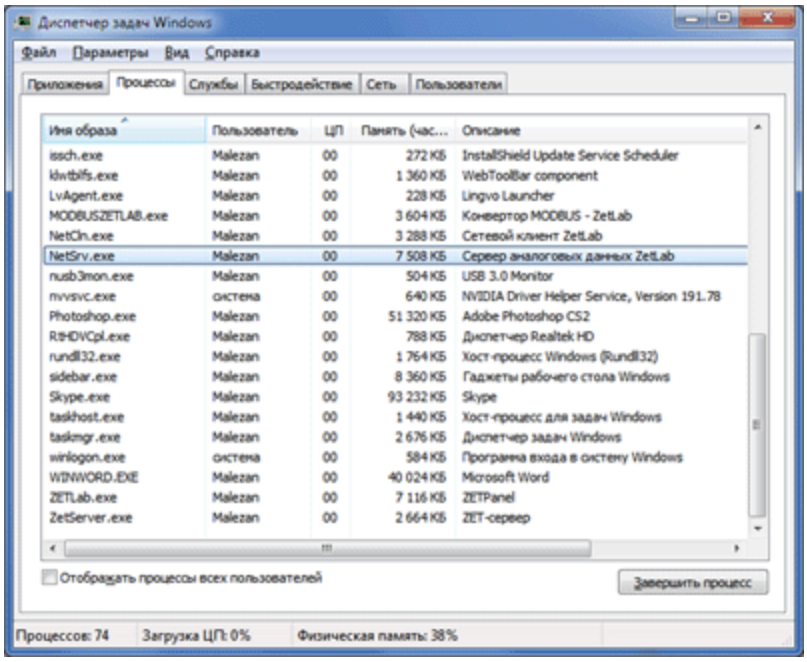

Разрешение доступа к сети

## 3.5.Удаление ПО ZETLAB

Последовательность действий при удалении ПО:

1. Закрыть все используемые программы ZETVIEW, ZETSCOPE, ZETCABLETEST, ZETLAB.
2. Выключить приборы ZET (анализаторы спектра, платы АЦП/ЦАП, осциллографы и др.) и отключить их от ПК.
3. Проверить в диспетчере задач (см рисунок ниже), что нет программ ZETLAB, работающих в скрытом режиме:
   - Modbus.exe
   - NetServer.exe
   - NetSrv.exe
   - ZetServer.exe
4. Если какая-либо из перечисленных программ запущена, то завершить ее выполнение.
5. Удалить ПО через панель управления ОС.

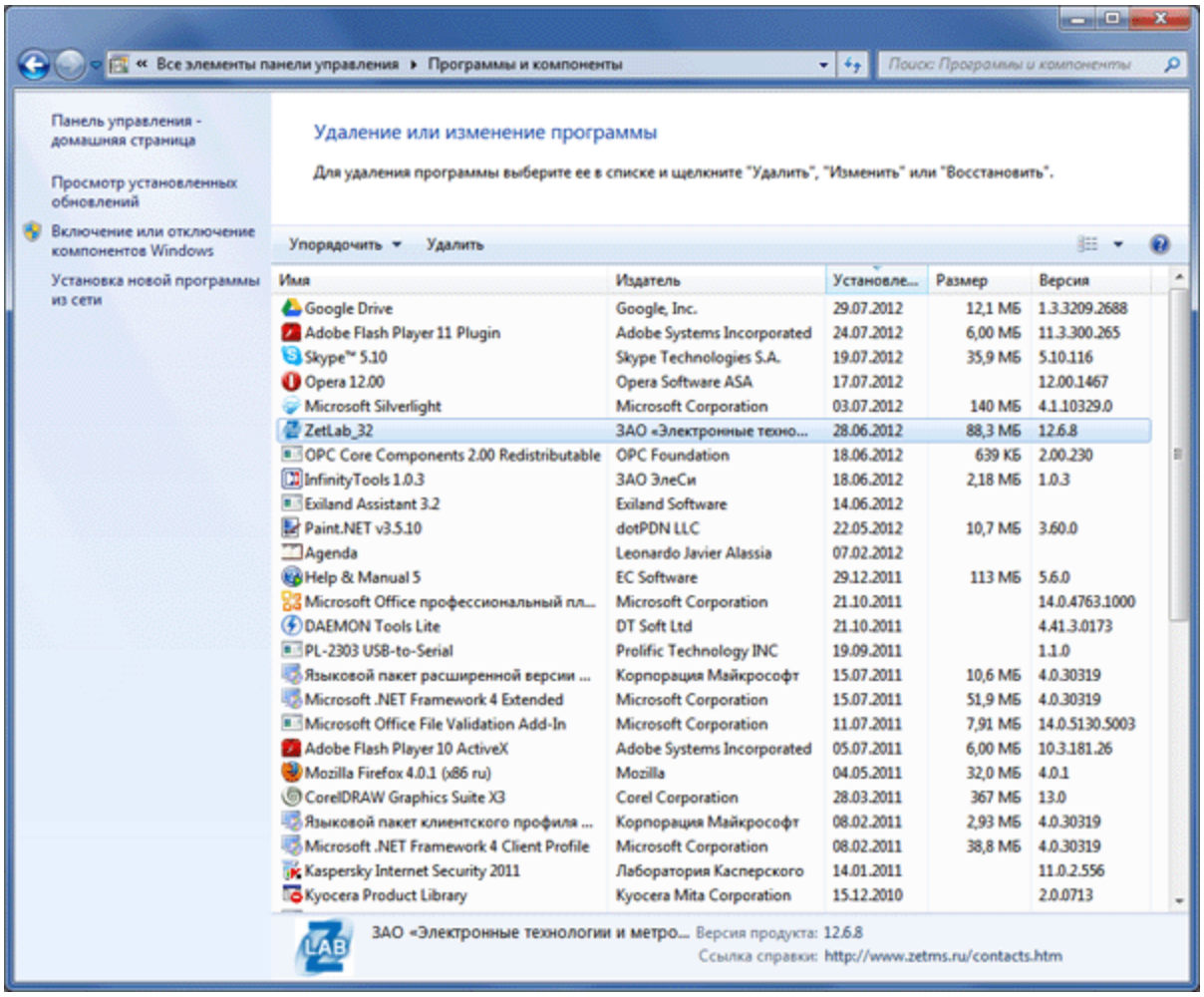
© ООО "ЭТМС" 1992-2024. Все права защищены




Диспетчер задач Windows

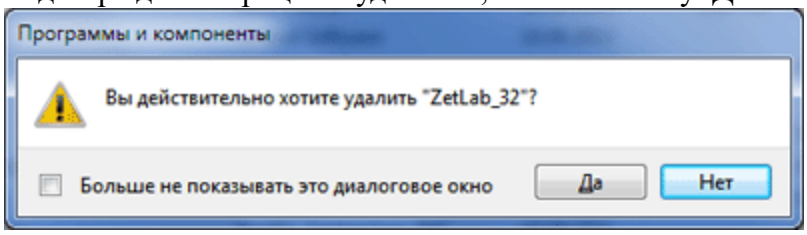
Для удаления программы необходимо из меню Пуск панели задач Windows выбрать команду Панель управления → Программы и компоненты, после чего запустится окно "Удаление или изменение программы" (см. рисунок ниже).

© ООО "ЭТМС" 1992-2024. Все права защищены




Панель управления, окно удаления программ

В открывшемся окне "Удаление или изменение программы" из списка установленных программ необходимо выбрать пакет программного обеспечения ZETLAB, нажав на него левой клавишей «мыши». Выбранное программное обеспечение выделится синим цветом. После нажатия кнопки Удалить появится информационное окно (см. рисунок ниже), запрашивающее подтверждение удаления программного обеспечение ZETLAB. В этом окне, для подтверждения процесса удаления, нажать кнопку "Да".




Подтверждение удаления ZETLAB

*Примечание: отключение ZET-устройства и закрытие всех программ ZETLAB требуется для полного удаления ZETLAB и драйверов. Только в этом случае гарантируется корректная установка новой версии программного обеспечения и драйверов во всех операционных системах при последующих установках ZETLAB.*

© ООО "ЭТМС" 1992-2024. Все права защищены

## 3.6.Обновление ПО ZETLAB

Установка обновлений ZETLAB (а также ZETCABLETEST, ZETVIEW и ZETSCOPE) производится только после удаления установленного ранее программного обеспечения.
*Примечание 1: для работы программного обеспечение ZETVIEW требуется наличие установленного программного обеспечения ZETLAB.*
*Примечание 2: Программное обеспечение ZETSCOPE, поставляемое с осциллографами ZET 302, имеет функции записи результатов измерений. Для просмотра записанных файлов используется программа "Просмотр и обработка результатов" из состава ПО ZETLAB. Таким образом, для работы с ПО ZETSCOPE не требуется наличие ПО ZETLAB, но оно может являться дополнением к ПО ZETSCOPE.*
Порядок действий
Порядок действий при обновлении ПО:

| Набор ПО | ZETLAB | ZETLAB+ZETSCOPE | ZETLAB+ZETVIEW | ZETLAB+ZETCOPE+ZETVIEW |
|---|---|---|---|---|
| 1) Удалить текущие версии в последовательности: | ZETLAB | 1) ZETSCOPE<br>2) ZETLAB | 1) ZETVIEW<br>2) ZETLAB | 1) ZETVIEW<br>2) ZETSCOPE<br>3) ZETLAB |
| 2) Установить новые версии в последовательности: | ZETLAB | 1) ZETLAB<br>2) ZETSCOPE | 1) ZETLAB<br>2) ZETVIEW | 1) ZETLAB<br>2) ZETSCOPE<br>3) ZETVIEW |

Удаление ZETLAB производить в соответствии с подразделом Удаление ПО ZETLAB 76 настоящего документа, удаление ZETVIEW и ZETSCOPE - в соответствии с руководствами оператора на них.
Установку ZETLAB производить в соответствии с подразделом Установка ПО ZETLAB 58 настоящего документа, установку ZETVIEW и ZETSCOPE - в соответствии с руководствами оператора на них.
Следование всем рекомендациям по удалению и установке ПО гарантирует корректное обновление ZETLAB и драйверов. В подразделе Обновление драйверов 80 приведены указания по обновлению драйверов в случае, если оно не было произведено автоматически.
**Чистка реестра**
Если при соблюдении всех приведенных рекомендаций после обновления ZETLAB возникают проблемы, возможно, требуется чистка реестра Windows. Последовательность действий следующая:

1. Удалить все продукты ZETLab через панель управления, начиная с ZETView и заканчивая ZETLab_32 (или ZETLab_64).
2. Удалить вручную папку, куда был установлен ZETLab (например, C:\ZETLab).
3. Запустить редактор реестра: Win+R, regedit, Enter

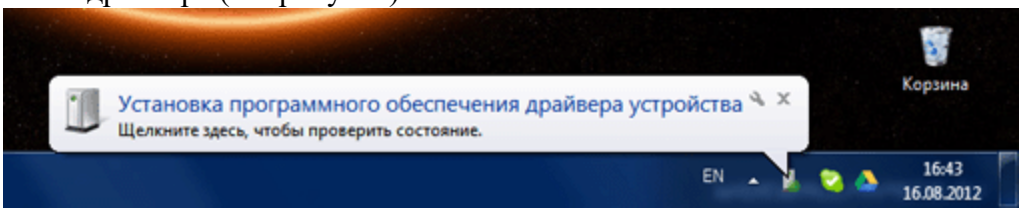
© ООО "ЭТМС" 1992-2024. Все права защищены

4. В дереве реестра зайти на ветку: - для 32-битной системы: HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\SharedDLLs - для 64-битной системы: HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Microsoft\Windows\CurrentVersion\SharedDLLs
5. Найти и удалить в данной ветке все строки, которые начинаются с пути установки (например, с "C:\ZETLab").
6. Установить ZETLab_32 (или ZETLab_64), затем остальные программные продукты.

*Примечание: работать с реестром Windows следует с осторожностью. Перед удалением строк желательно сохранить данную ветку, чтобы в случае удаления не тех строк была возможность восстановить изменения.*

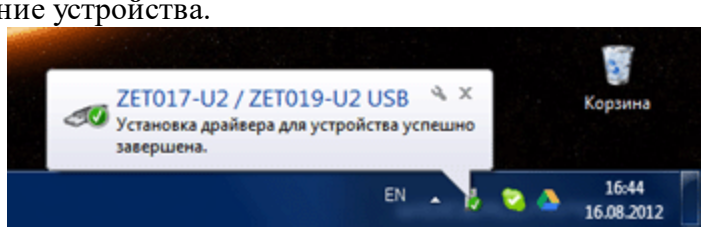
## 3.7.Обновление драйверов

В процессе установки ПО ZETLAB на компьютер копируются драйвера на оборудование ZET (анализаторы спектра, платы АЦП/ЦАП, тензостанции, сейсмостанции, интеллектуальные датчики и др.). Если при установки ПО был подключен какой-либо ZET прибор, драйвера на него были автоматически установлены и прибор готов к работе по завершению процесса установки.
Если прибор во время установки ПО не был подключен, то при первом подключении прибора к ПК в системном трее появится сообщение об установке программного обеспечения драйвера (см. рисунок).




Установка драйвера при подключении прибора к ПК

Установка драйвера может занять несколько минут. По окончании установки драйвера появится сообщение об успешном завершении, при этом в заголовке сообщения отобразится название устройства.



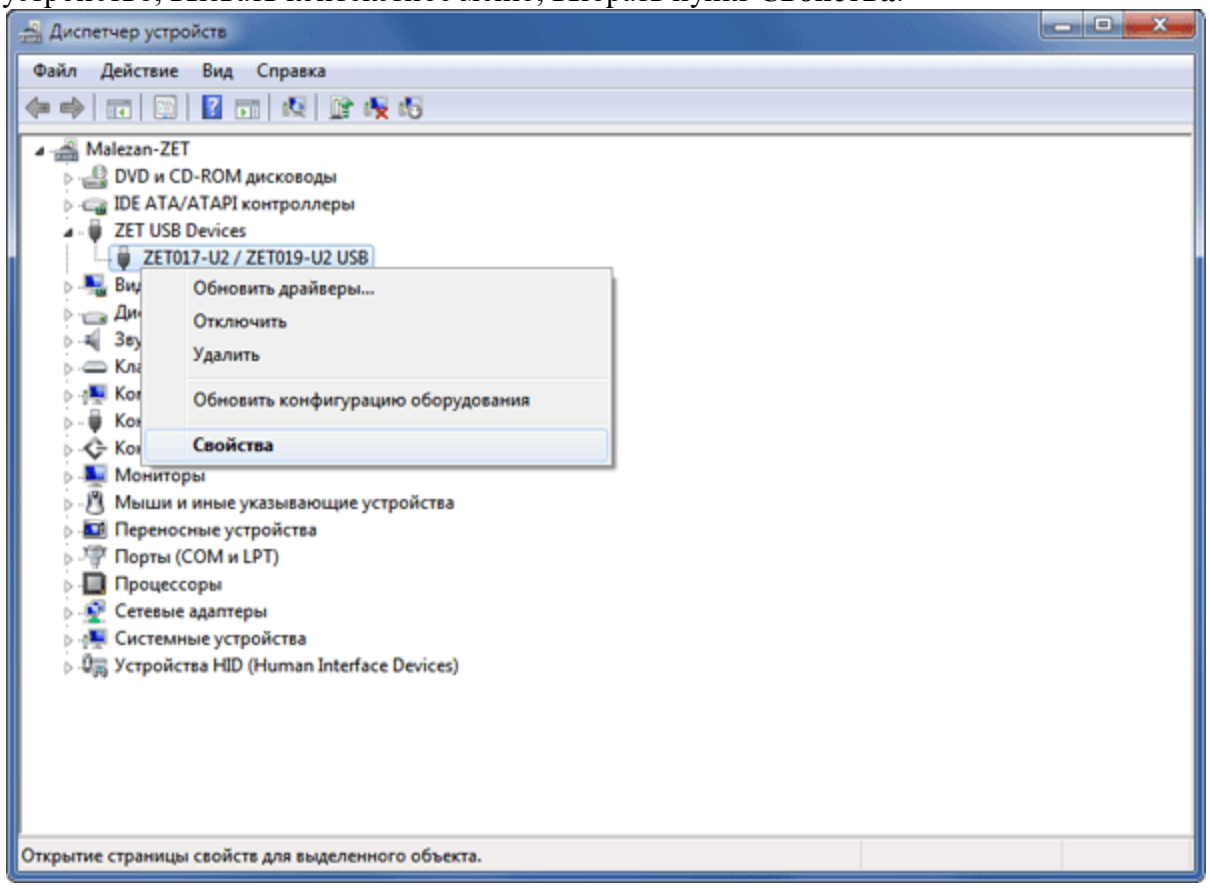

Сообщении об успешной установке драйвера

Если установка драйверов не была произведена автоматически (такое могло произойти, например, при обновлении ZETLAB, если в процессе удаления работали какие-либо из программ), то обновить драйвера можно самостоятельно. Для этого необходимо зайти в "Мой компьютер", вызвать контекстное меню (щелчком правой кнопки "мыши" по свободному полю окна "Мой компьютер"), в контекстном меню выбрать "Свойства", в открывшемся окне выбрать "Диспетчер устройств" (в Windows 7 ссылка "Диспетчер

© ООО "ЭТМС" 1992-2024. Все права защищены

устройств" находится слева, в Windows XP в окне свойств компьютера необходимо перейти на вкладку "Оборудование").

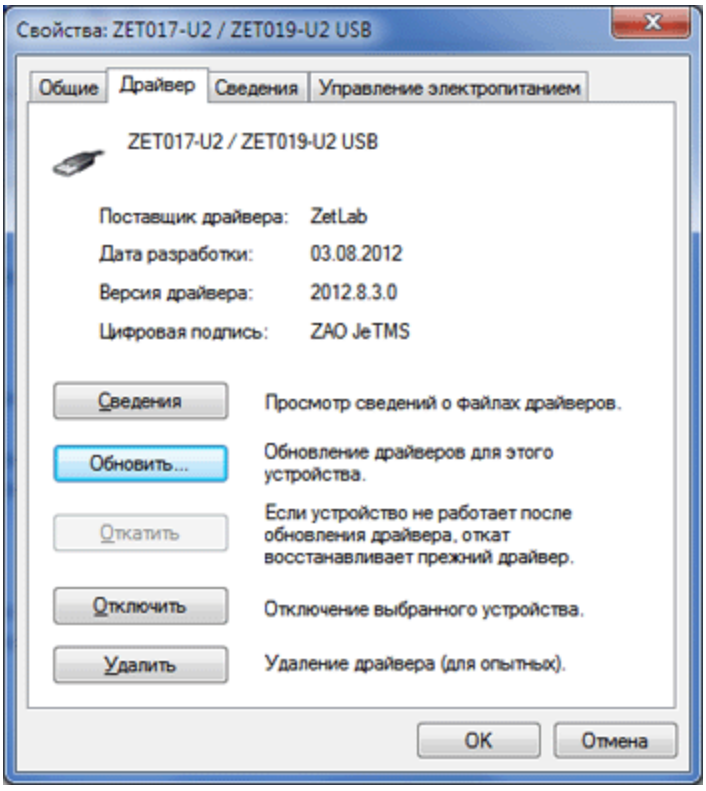
В диспетчере устройств раскрыть список "ZET USB Devices" выбрать подключенное устройство, вызвать контекстное меню, выбрать пункт **Свойства**.




Диспетчер устройств

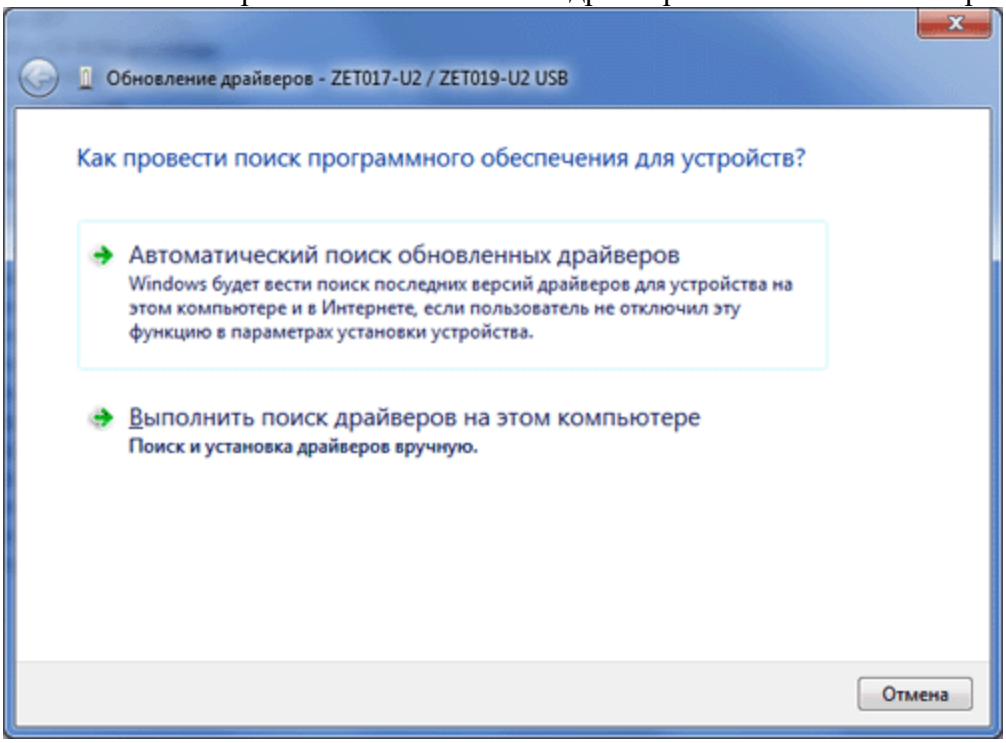
В окне свойств устройства перейти на вкладку "Драйвер" и нажать кнопку "Обновить".

© ООО "ЭТМС" 1992-2024. Все права защищены




Обновление драйвера

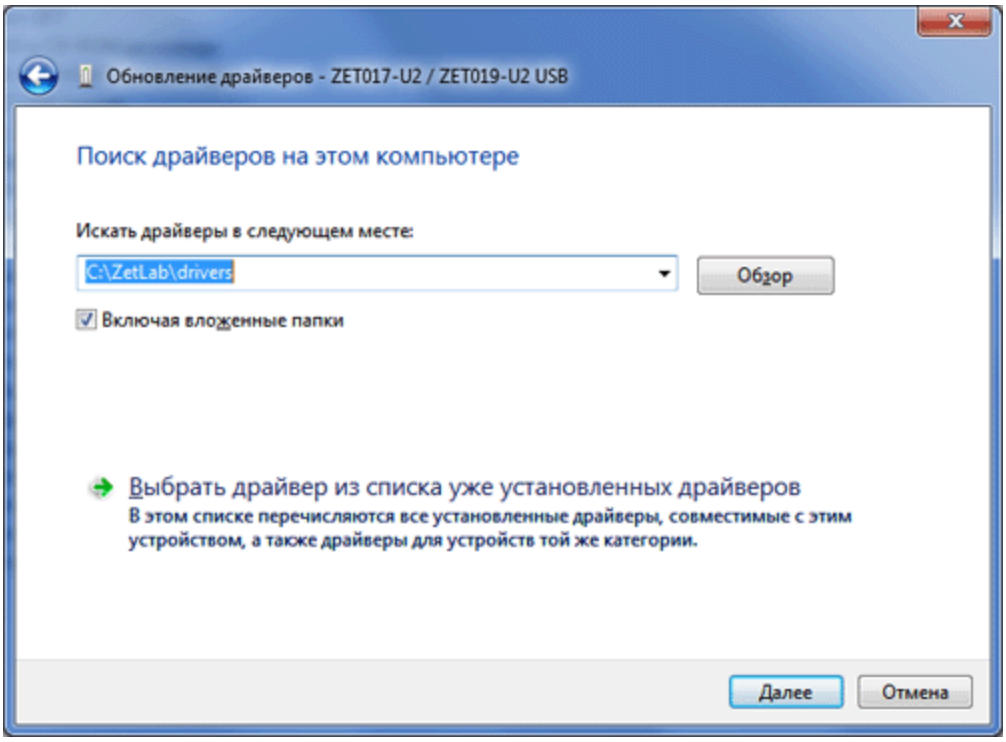
В открывшемся окне выбрать "Выполнить поиск драйверов на этом компьютере"




Выбор места обновлений драйвера

Указать директорию драйверов к ZET-устройствам: C:\ZETLab\drivers и нажать кнопку "Далее".

© ООО "ЭТМС" 1992-2024. Все права защищены




Директория драйверов к ZET-устройствам

Далее будут установлены драйвера на подключенное оборудование и появится сообщение об успешном завершении установки.

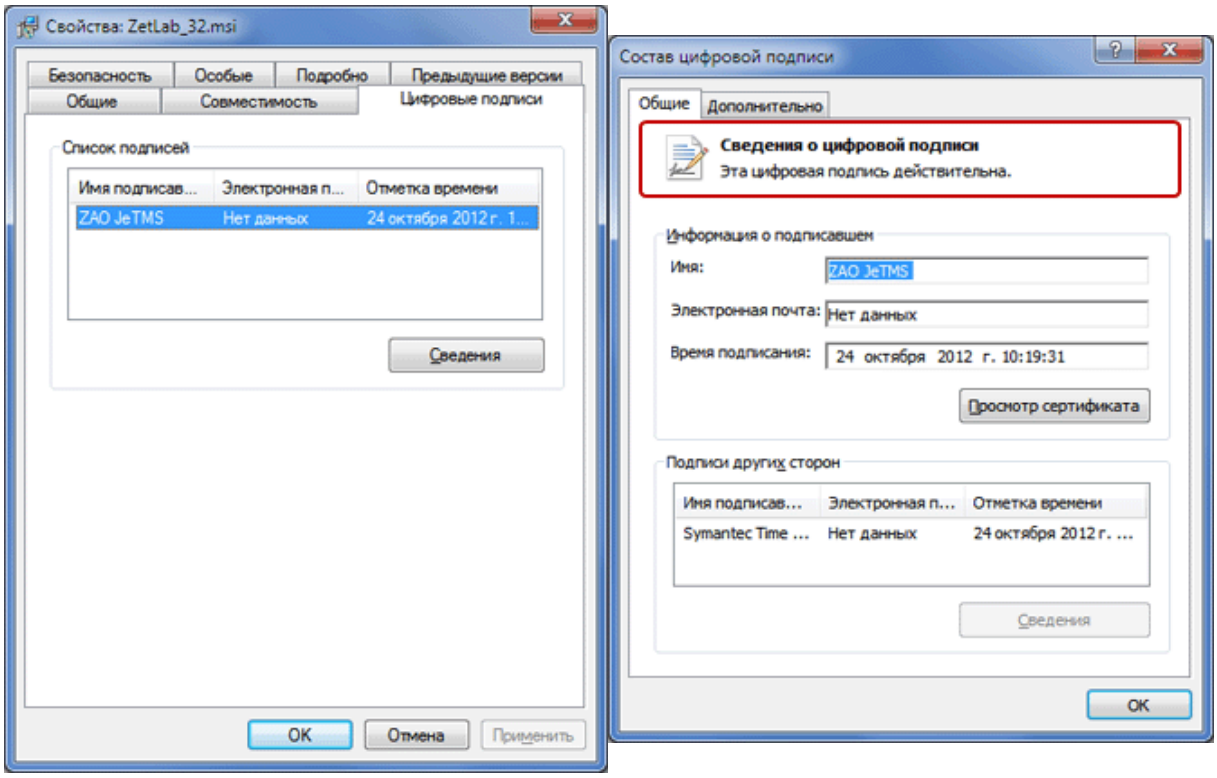
## 3.8.Проверка целостности установочного файла дистрибутивов ZETLAB

Обновления программного обеспечения, производимого ООО "ЭТМС", доступны в сети (их можно скачать в фтп сервера). Для проверки целостности загруженных из сети файлов можно воспользоваться стандартными средствами Windows по проверке цифровой подписи или программами по проверке контрольной суммы md5.

Контрольная сумма md5 файла дистрибутива указывается в одноименном текстовом файле. Проверяется бесплатными программами загруженными из интернета.

Для проверки цифровой подписи необходимо открыть свойства установочного файла (щелчок правой кнопкой мыши по названию файла и выбор пункта "Свойства"), в окне свойств перейти на вкладку "Цифровые подписи", выбрать ZETLAB ООО и нажать кнопку "Сведения". В окне "Состав цифровой подписи" должно быть указано, что цифровая подпись действительна.

© ООО "ЭТМС" 1992-2024. Все права защищены




# Глава 4.Подключение приборов

В данном разделе приведено описание настроек программ ZETLAB при подключении приборов ZET по различным интерфейсам.

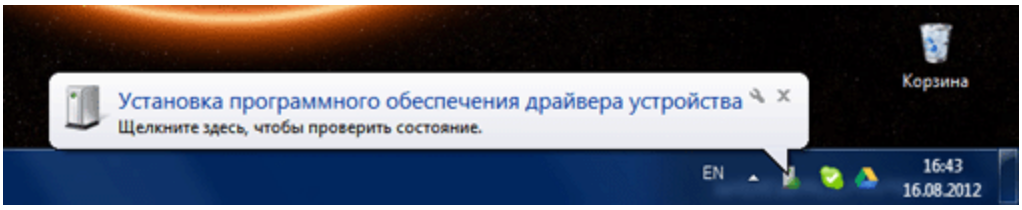
## 4.1.Подключение по USB

Подключение устройств ZET к компьютеру осуществляется входящим в комплект кабелем HighSpeed USB 2.0 к порту HighSpeed USB 2.0 ПЭВМ, при выключенном или включенном питании компьютера.

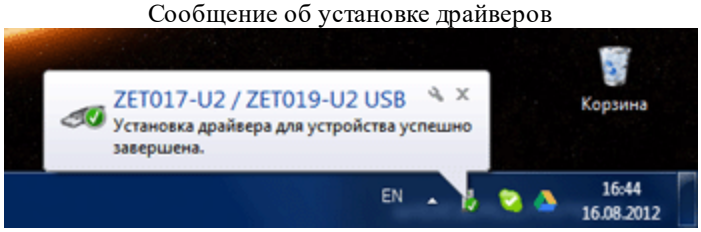
*Примечание: подключение интеллектуальных датчиков ZETSENSOR к компьютеру производится с использованием преобразователей интерфейсов ZET 7070 или ZET 7174.*

Если программное обеспечение ZETLAB уже установлено на компьютере, то при первом подключении устройства ZET драйвера на него установятся автоматически: при подключении устройства появится сообщение о том, что производится установка ПО драйвера устройства, затем сообщение об успешной установке драйверов.

© ООО "ЭТМС" 1992-2024. Все права защищены




Сообщение об установке драйверов




Сообщении об успешной установке драйвера

Если программное обеспечение ZETLAB не было установлено, его необходимо установить. При установке ПО ZETLAB с подключенным ZET устройством драйвера на устройство будут установлены в процессе установки ZETLAB.
Таким образом, драйвера на устройство ZET будут установлены автоматически:

- в процессе установки ZETLAB, если устройство было подключено во время установки ZETLAB,
- или при первом подключении устройства к компьютеру, на котором установлено ZETLAB.

Если драйвера на устройство не были установлены автоматически, их можно установить с помощью диспетчера устройств Windows - см раздел Обновление драйверов 80.

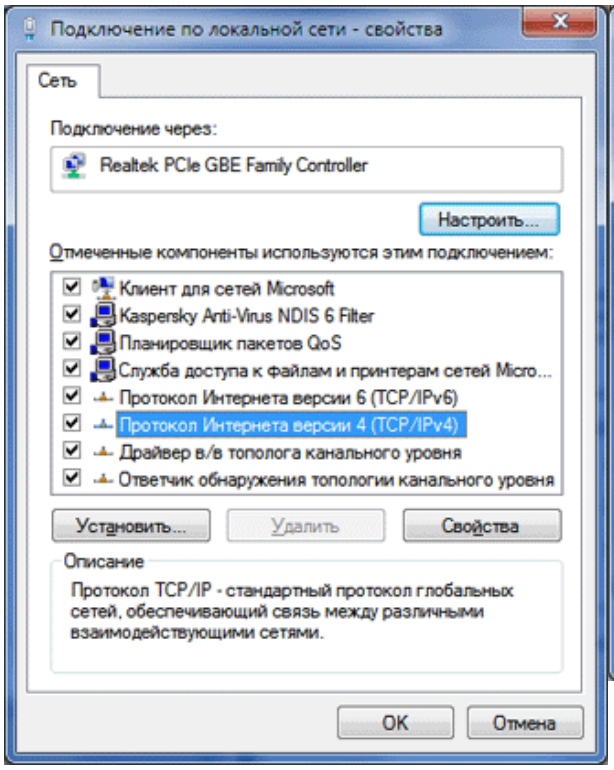
## 4.2.Подключение по Ethernet

Устройства ZET с опцией "Интерфейс Ethernet" можно подключать к ПК по локальной сети.
Для работы устройств ZET по локальной сети необходимо выполнить следующие действия:

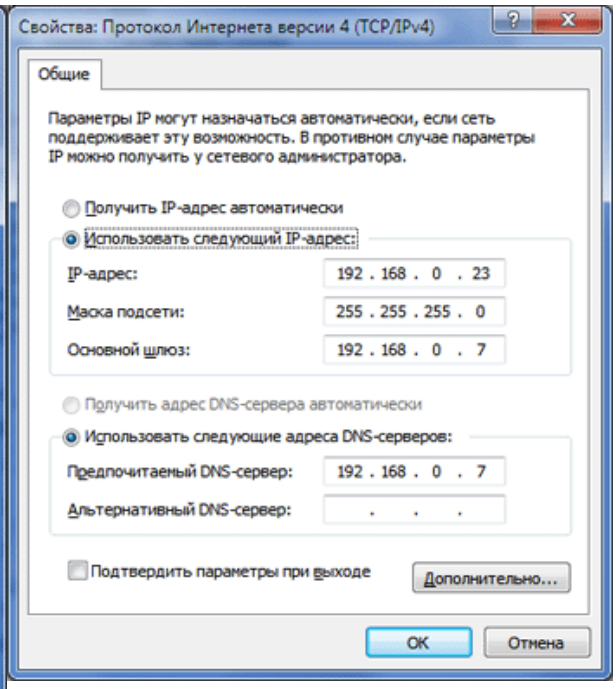
1. Подключить устройство ZET к ПК по USB и задать IP-адрес и маску подсети в программе "Диспетчер устройств" меню "Сервисные".
2. Подключить устройство ZET к локальной сети и настроить подключение в программе "Подключение устройств по Ethernet"

Настройка устройств ZET для работы по локальной сети
Для связи устройства ZET с компьютером по локальной сети необходимо, чтобы они находились в одной сети. Узнать IP-адрес компьютера можно через панель управления Windows — пункт "Центр управления сетями и общим доступом", далее — "Подключение по локальной сети". В открывшемся окне "Подключение по локальной сети - свойства" выбрать пункт "Протокол Интернета 4 (TCP/IP)" и нажать кнопку "Свойства", в открывшемся окне списать IP-адрес и маску подсети.

© ООО "ЭТМС" 1992-2024. Все права защищены

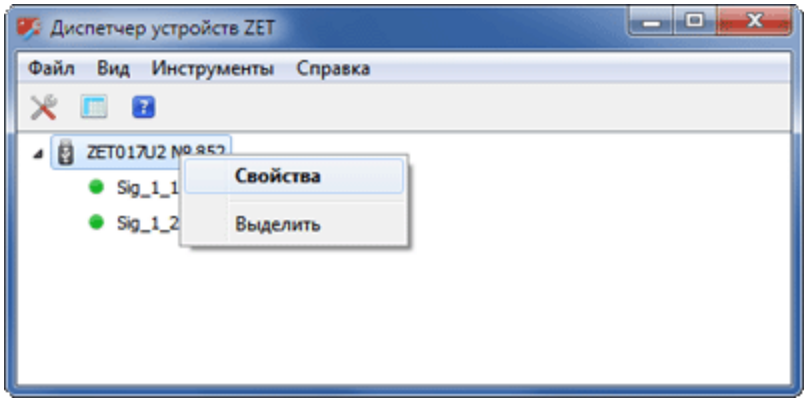



Вызов свойств подключения компьютера к сети




Окно свойств подключения компьютера к сети

Далее необходимо настроить сетевые свойства ZET-устройства. Для этого запустить программу "Диспетчер устройств" из меню "Сервисные" панели ZETLAB и, щелкнув правой кнопкой "мыши" по названию устройства, выбрать в появившемся контекстном меню пункт "Свойства".




Вызов окна свойств ZET-устройства

Окно свойств устройства открывается на вкладке "Общие". Для задания сетевых параметров необходимо перейти на вкладку "Ethernet". Установить IP-адрес и маску подсети так, чтобы компьютер и настраиваемое устройство были в одной группе.

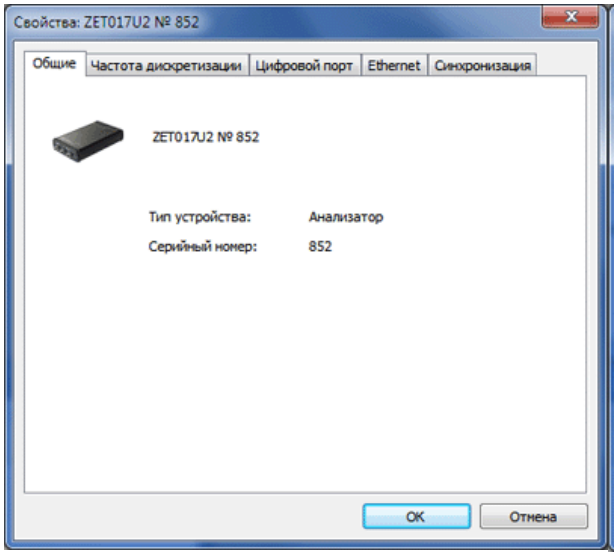
Допустим, мы имеем следующие параметры сети:

- IP-адрес используемого в работе компьютера: 192.168.0.23;
- Маска подсети: 255.255.255.0;
- Основной шлюз: 192.168.0.1;

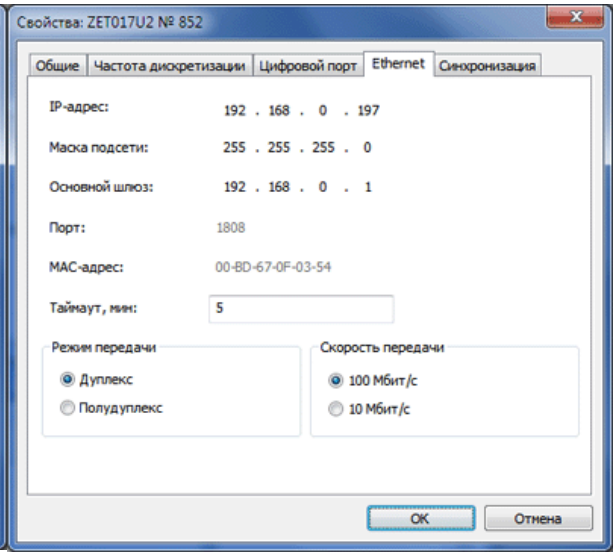
© ООО "ЭТМС" 1992-2024. Все права защищены

В таком случае для настраиваемого устройства необходимо указать следующие значения:

- IP-адрес: 192.168.0.197 (последнее число адреса может быть и другим, но с начала надо убедиться, что данный IP-адрес не занят другими компьютерами либо иными устройствами);
- Маска подсети 255.255.255.0;
- Основной шлюз 192.168.0.1.




Общие свойства ZET-устройства




Сетевые настройки ZET-устройства

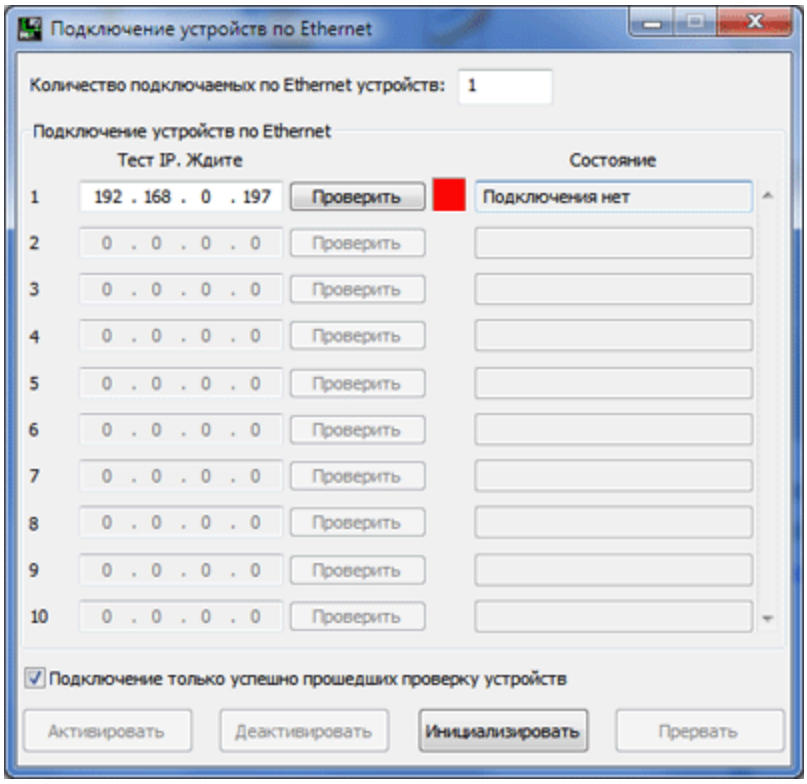
После сохранения сетевых настроек ZET-устройства его можно отключать от шины USB компьютера и подключить к сети на месте эксплуатации с помощью кабеля Ethernet. Далее необходимо настроить подключение по Ethernet на компьютере, находящимся в той же сети.

Настройка подключения устройств ZET по локальной сети

Для настройки подключения устройства ZET к компьютеру по локальной сети необходимо запустить программу "Подключение по Ethernet" из меню "Сетевые" панели ZETLAB.

Далее необходимо указать количество подключаемых по Ethernet устройств и указать их IP-адреса.

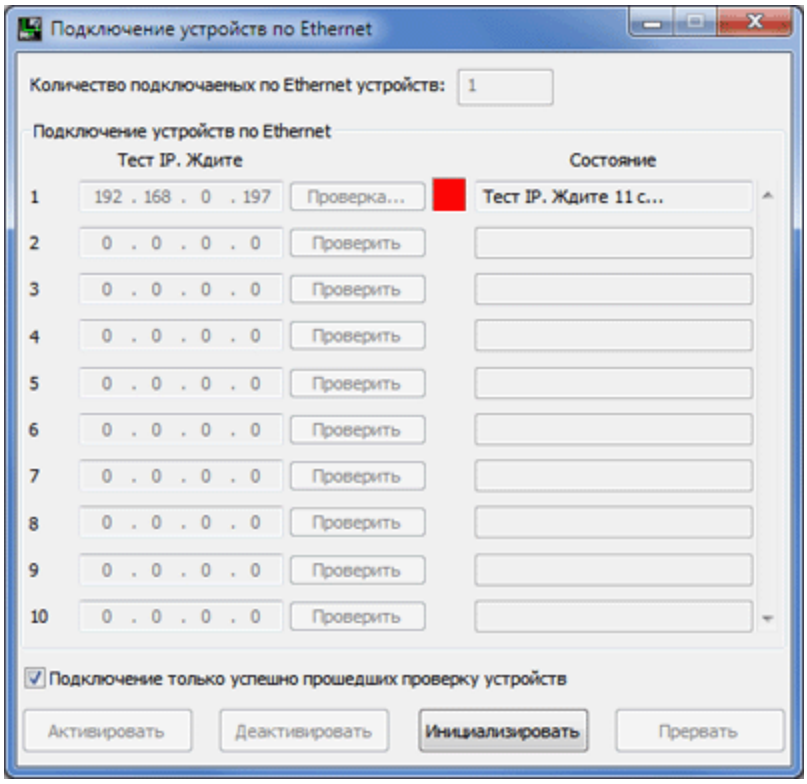
© ООО "ЭТМС" 1992-2024. Все права защищены




Шаг 1: указать IP-адрес ZET-устройства

Далее необходимо нажать кнопку "Проверить" справа от IP-адреса устройства, с которым осуществляется соединение.

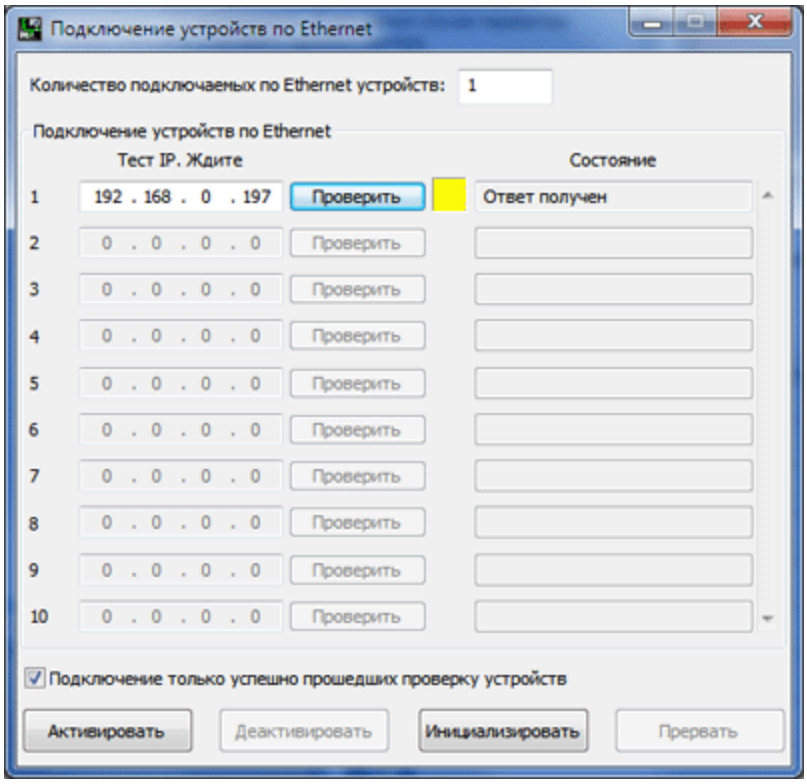
© ООО "ЭТМС" 1992-2024. Все права защищены




Шаг 2: Нажать кнопку "Проверить" и дождаться окончания проверки

Когда получены ответы от всех устройств, с которыми планируется вести работу по сети, можно инициализировать связь с ними (кнопка "Инициализировать").

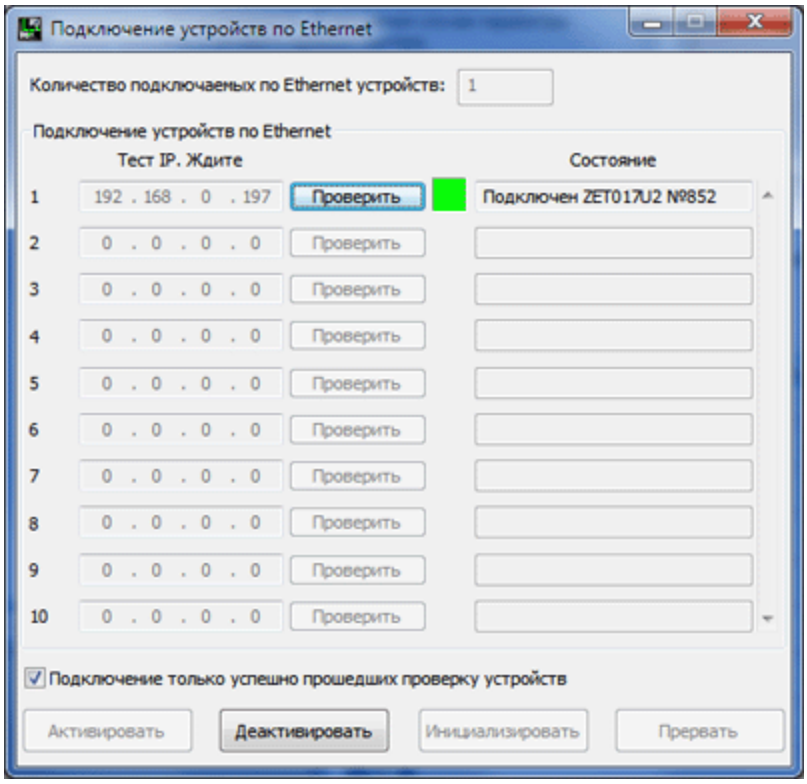
© ООО "ЭТМС" 1992-2024. Все права защищены




Шаг 3: Инициализировать подключение при получении ответа

При успешной инициализации устройств права от IP-адреса отобразится его обозначение и номер и измерительные каналы появятся в списке каналов ZET-сервера.

© ООО "ЭТМС" 1992-2024. Все права защищены




Устройство готово к работе по локальной сети

## 4.3.Подключение по Wi-Fi

Беспроводной интерфейс Bluetooth реализован на модуле АЦП ZET 210. Такое подключение обеспечивает полную гальваническую развязку измерительных цепей и цифровых цепей компьютера. Подключение по Bluetooth позволяет размещать измерительную часть на подвижных элементах конструкции, например на валах, тележках и пр., а регистрирующую на неподвижной части.

Для запуска программы необходимо в панели управления выбрать группу "Сетевые программы" и нажать кнопку "**Подключение устройств по Bluetooth**". На экране монитора отображается рабочее окно программы (см. рисунок 2).

Примечание: программу можно запустить непосредственно из рабочей директории ZETLab (по умолчанию: c:\ZETLab\). Имя запускаемого файла: BthWizard.exe.

© ООО "ЭТМС" 1992-2024. Все права защищены

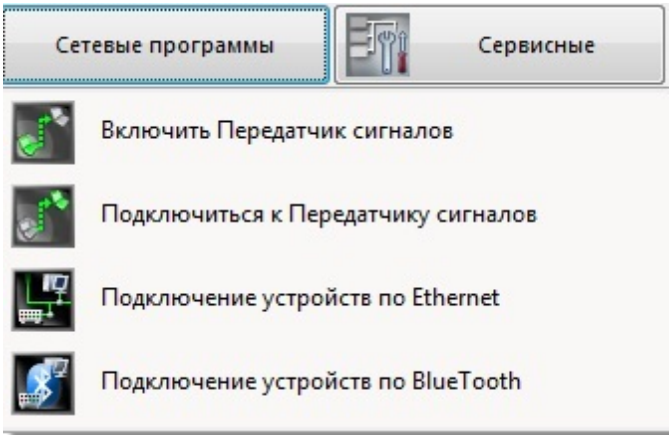



Рисунок 1

При подключении нескольких устройств по Bluetooth используется утилита, которая позволяет проверить наличие Bluetooth устройств и обеспечивает соединение типа точка-точка одновременно со всеми выбранными устройствами.

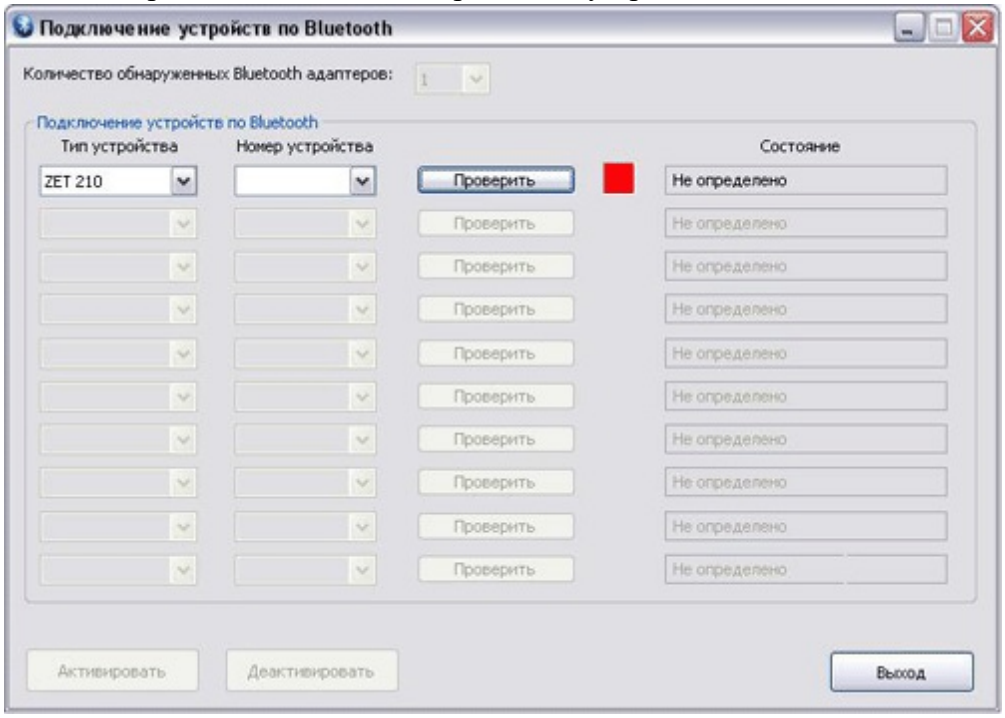



Рисунок 2

# Глава 5.О работе с ПО ZETLAB

В данном разделе будут рассмотрены общие вопросы работы с программным обеспечением ZETLAB: настройки измерительных каналов, настройки программ и т.п.
Многие ZET-приборы являются универсальными устройствами и при решении конкретной задачи возникает вопрос правильной настройки измерительных каналов. Кроме того, получаемые результаты напрямую зависят от настроек программ. Описания

© ООО "ЭТМС" 1992-2024. Все права защищены

параметров программ приводятся в разделах, посвященных каждой конкретной программе, а в данной главе рассматриваются общие принципы, такие как время усреднения данных, частота дискретизации сигнала и т.п.

## 5.1.Приступая к работе

Программное обеспечение ZETLAB предназначено для обработки сигналов, регистрируемых измерительным оборудованием ZET с различных первичных преобразователей (датчиков).

Работа с программным обеспечением ZETLAB начинается с:
- изучения эксплуатационной документации на ПО, измерительные приборы и датчики,
- установки ПО,
- подключения датчиков к измерительному прибору и измерительного прибора к ПК,
- настройки измерительных каналов,
- настройки программы.

Описание установки ПО ZETLAB и требования к ПК приведены в разделе 2 настоящего руководства "Установка ПО ZETLAB 58".
Подключение датчиков к измерительному прибору рассматривается в описаниях программ измерения, таких как "Термометр термопары", "Термометр термосопротивления", "Тензометр", а также на нашем сайте http://zetlab.ru и в руководствах по эксплуатации на измерительные приборы и датчики.
Настройка ПО при подключении измерительного прибора к ПК рассматривается в разделе "Подключение приборов 84" настоящего руководства. Порядок подключения измерительного прибора к ПК рассматривается в руководстве по эксплуатации на прибор.
Итак, документация изучена, датчики установлены в местах измерений, подключены к приборам, приборы подключены к компьютеру. С чего начать измерения?

***Для получения результатов необходимо:***
1. Настроить измерительный прибор, как минимум требуется подобрать подходящую частоту дискретизации сигнала - см. подраздел "Время усреднения данных и частота дискретизации сигналов 94" настоящего документа. Частота дискретизации задается в программе "Диспетчер устройств" меню "Сервисные".
2. Дальше требуется настроить измерительные каналы. Настройки измерительных каналов зависят от типа подключенного датчика и типа программы, которая будет использоваться для измерений. Например, для вибродатчиков указывается коэффициент преобразования и единицы измерений - измерения программой "Виброметр" осуществляются с учетом этих данных. То же самое касается датчиков оборотов, датчиков перемещений - в настройках измерительных каналов указываются чувствительность (коэффициент преобразования) и единицы измерений согласно паспорту. А при подключении датчиков силы, термометров и т.п. все

© ООО "ЭТМС" 1992-2024. Все права защищены

настройки задаются в программах измерения, а параметры измерительных каналов должны оставаться заданными по умолчанию. Настройки измерительных каналов задаются в программе "Диспетчер устройств" меню "Сервисные".

3. Выбрать программу для анализа или измерения. При необходимости провести предварительную обработку сигналов, например, фильтрацию. Для записи результатов может использоваться самописец.

***О питании датчиков***

Питание некоторых датчиков обеспечивается внешним источником, например, при подключении датчиков к платам АЦП/ЦАП через усилитель ZET 410/412 используются встроенные источники тока и источники напряжения усилителя.
Датчики стандарта ICP подключаются к анализатору спектра и в настройках измерительных каналов устанавливается флаг ICP.
При запитывании датчиков также может использоваться генератор измерительного прибора (выход ЦАП). Управление этим генератором осуществляется с помощью программы "Генератор сигналов различной формы".
*Примечание: не все ZET приборы имеют ЦАП.*

***Рекомендуем:***

Программы ZETLAB могут запускаться непосредственно из директории их установки (по умолчанию "C:\ZETLab") или из панели управления ZETLAB.
Панель управления ZETLAB позволяет не только быстро запустить нужный виртуальный прибор, но и имеет множество полезных функций, таких как сохранение проектов (особенно актуально, когда работа ведется с большим количеством программ). Поскольку панель ZETLAB не является измерительным прибором, пользователи обходят своим вниманием ее описание, и в результате упускают из вида некоторые удобные функции.

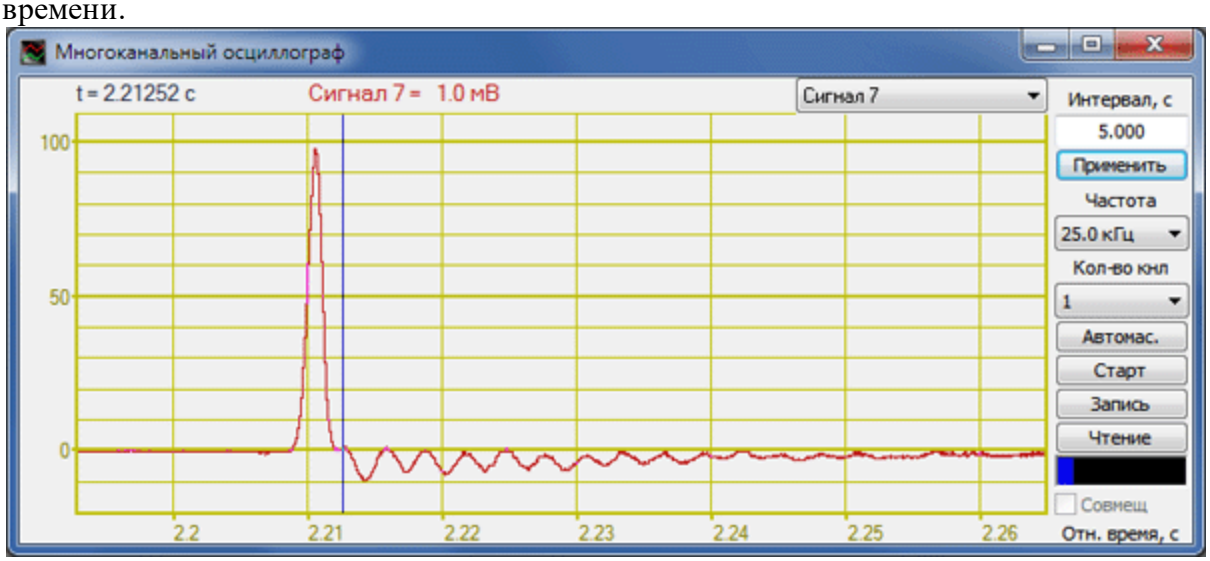
## 5.2.Время усреднения данных и частота дискретизации сигналов

В большинстве программ ZETLAB встречается такой параметр, как время усреднения данных. На что он влияет и с чем связан? Установленное время усреднения напрямую влияет на получаемый результат измерений и неразрывно связан с частотой дискретизации сигнала.
Параметры измерений напрямую связаны с изменением обрабатываемого сигнала во времени. В целом различают:

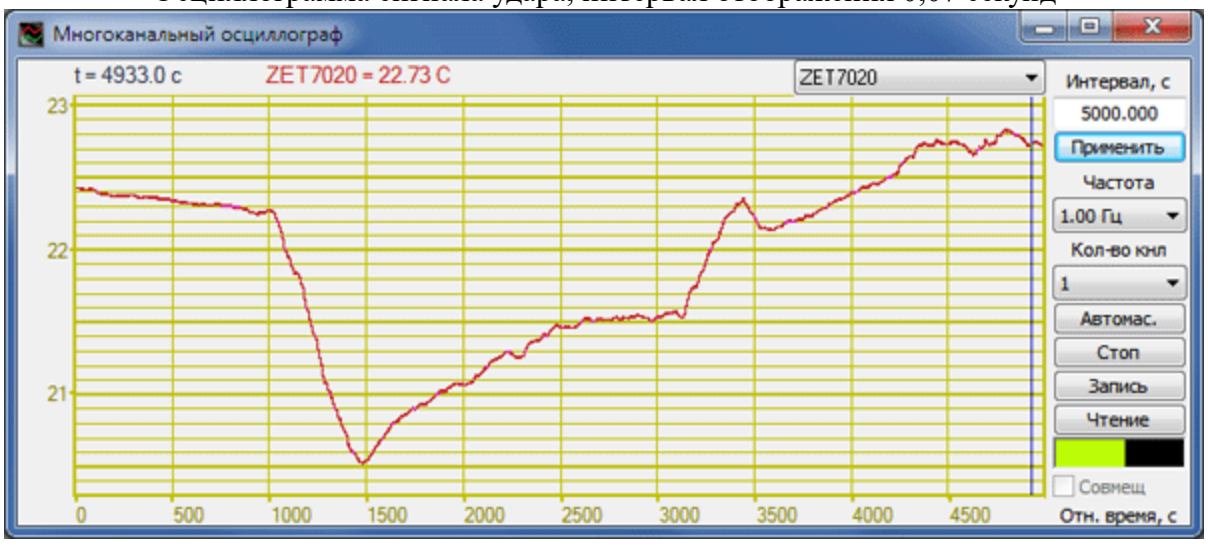
- быстро протекающие процессы (взрывы, ударные воздействия) или высокочастотные
- медленно протекающие процессы (изменение температуры на улице) или низкочастотные
- сверхмедленно протекающие процессы (сейсмические воздействия).

© ООО "ЭТМС" 1992-2024. Все права защищены

Сравним 2 сигнала: один получен при испытаниях изделий на ударной установке, другой при мониторинге температуры в помещении. Для анализа сигнала удара развертка по оси времени составляет 0,07 секунд. Сигнал температуры в офисном помещении за такое время едва ли заметно изменится. Из приведенных примеров очевидно, что анализ разных сигналов необходимо проводить в разные интервалы времени.




Осциллограмма сигнала удара, интервал отображения 0,07 секунд

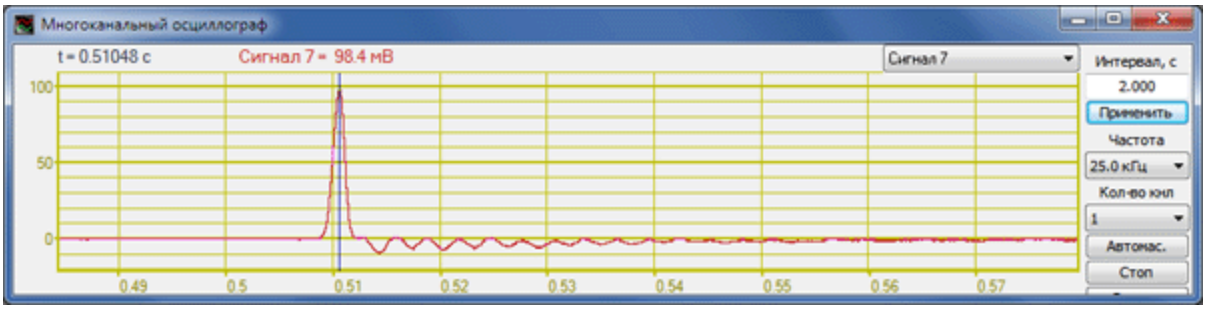



Осциллограмма сигнала температуры, интервал отображения 5000 с

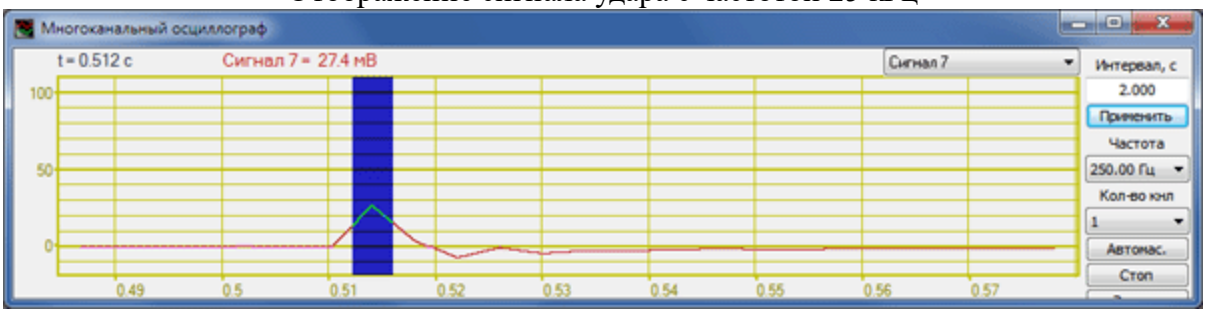
Но ни одна программа не даст достоверных результатов, если исходный сигнал имеет недостаточную или, наоборот, избыточную степень детализации.

На рисунках ниже приведен пример отображения одного и того же сигнала удара с разной частотой: 25 кГц и 250 Гц. Невооруженным глазом видно, что второй вариант представления сигнала для анализа не пригоден.

© ООО "ЭТМС" 1992-2024. Все права защищены

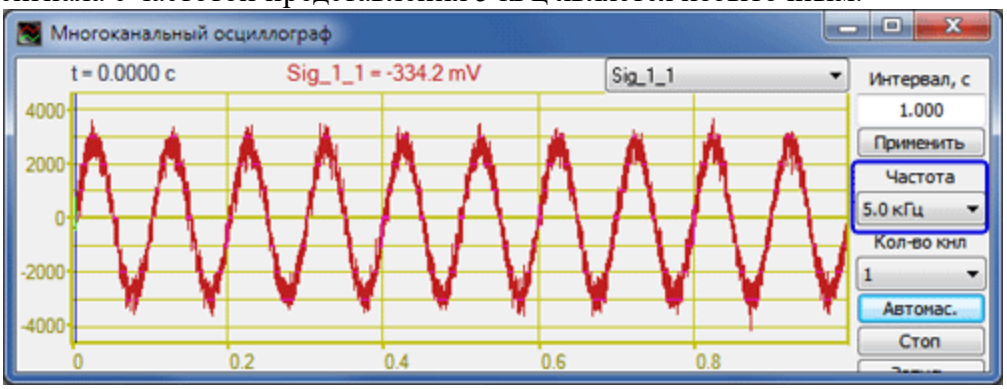



Отображение сигнала удара с частотой 25 кГц

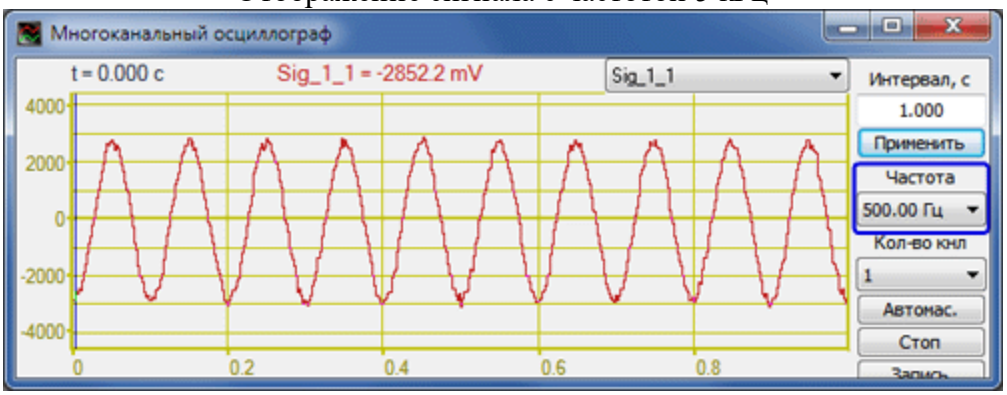



Отображение сигнала удара с частотой 250 Гц

На рисунках ниже представлен пример отображения одного и того же сигнала (синусоидальный сигнал с большим уровнем шума) с разной частотой: 5 кГц и 500 Гц. Поскольку частота самого сигнала составляет 10 Гц, частоты представления 500 Гц вполне достаточно: на один период сигнала приходится 50 точек (500/10=50). А вот анализ сигнала с частотой представления 5 кГц является избыточным.




Отображение сигнала с частотой 5 кГц




Отображение сигнала с частотой 500 Гц

© ООО "ЭТМС" 1992-2024. Все права защищены

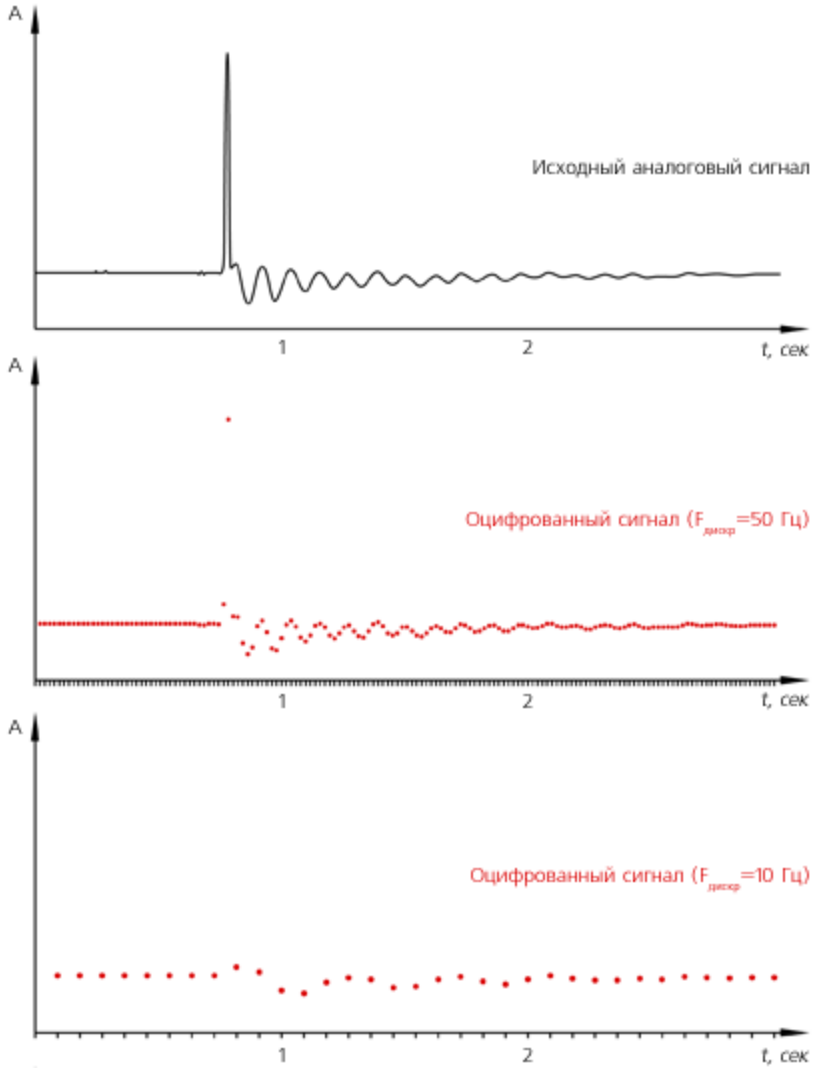
Приведенные выше примеры получены с помощью программы "Многоканальный осциллограф" и являются лишь наглядной демонстрацией влияния временных и частотных параметров на результат. Изменяемая в программе "Многоканальный осциллограф" частота представления сигнала влияет лишь на отображаемые данные, и зависит от частоты дискретизации сигнала. Не все программы ZETLAB имеют параметр *Частота представления*, но результат работы любой программы зависит от частоты дискретизации сигнала.

Частота дискретизации сигнала

Начнем с того, что программы ZETLAB предназначены для обработки оцифрованных сигналов. Аналоговый сигнал с датчика оцифровывается измерительным прибором и полученный цифровой сигнал формируется в виде канала сервера данных ZETSERVER. Канал данных представляет собой мгновенные отсчёты сигнала - определенное количество в секунду. Количество отсчетов в секунду называется частотой дискретизации $F_{ADC}$ и задается для измерительного прибора в программе "Диспетчер устройств" меню "Сервисные".

На рисунках ниже приводится пример оцифровки одного и того же сигнала с разными частотами дискретизации. Пример наглядно демонстрирует, что частота дискретизации при оцифровке сигнала должна быть достаточной для анализа исследуемого процесса.

© ООО "ЭТМС" 1992-2024. Все права защищены




Оцифровка аналогового сигнала с разными частотами дискретизации

Общее правило при выборе частоты дискретизации: чем быстрее протекает процесс, чем выше должна быть частота дискретизации.

Частотный диапазон

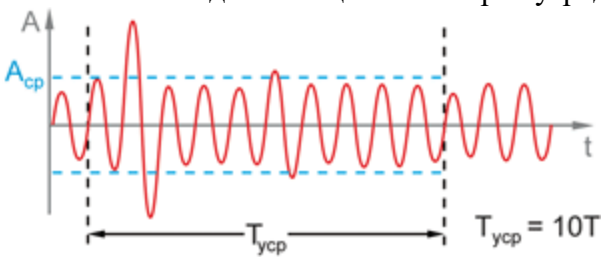
Поскольку сигналы оцифровываются с определенной частотой дискретизации, их анализ может проводиться в ограниченном диапазоне частот. Значения параметров сигнала могут быть вычислены программами, если на период сигнала приходится не менее 2-3-х точек. Например, при частоте дискретизации сигнала 25 кГц возможен анализ в диапазоне до 10 кГц.

Время усреднения данных

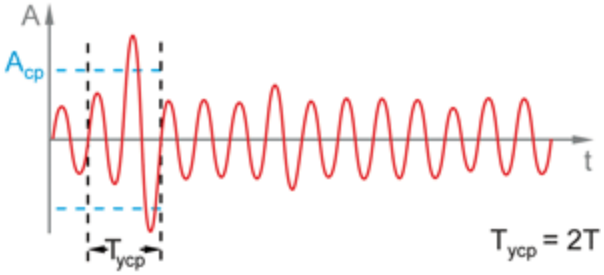
Оцифрованный сигнал представляет собой последовательность мгновенных значений сигнала. И на данном этапе из всех временных параметров измерений имеется только один - частота дискретизации, т.е. количество точек в секунду. При вычислении какого-либо параметра сигнала появляется другой параметр - время усреднения данных. На рисунках ниже приведен пример расчета СКЗ (среднеквадратичное значение сигнала) с

© ООО "ЭТМС" 1992-2024. Все права защищены

усреднением, равным двум периодам сигнала T, и с усреднением, равным десяти периодам сигнала T. Необходимо подбирать время усреднения в соответствии с решаемой задачей. Если требуется сглаживание помех, то рекомендуется увеличить время усреднения данных при измерениях. При исследованиях переходных процессов, наоборот, требуется высокая степень детализации и быстрое усреднение.




Вычисление СКЗ за время, равное 10 периодам сигнала




Вычисление СКЗ за время, равное 2 периодам сигнала

## 5.3.Элементы управления

Программы ZETLAB имеют интуитивно понятный интерфейс, удобную систему управления. При разработке программ ZETLAB учитываются общепринятые стандарты и обозначения, что делает работы с ними простой и удобной

В данном разделе будет описана работа с элементами управления программ ZETLAB.

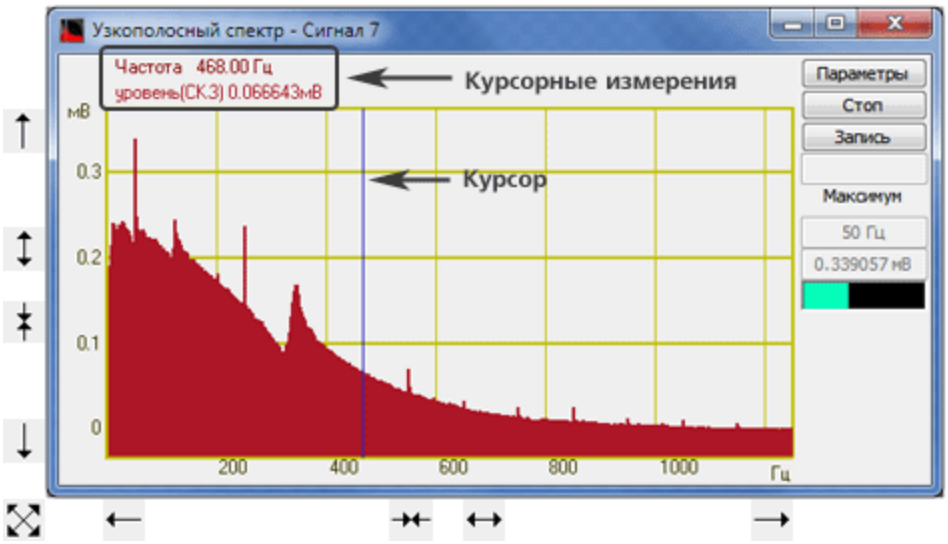
### Управление курсором и масштабирование графиков

Программы ZETLAB, имеющие поле для отображения результатов в графическом виде, также имеют удобную систему масштабирования графиков и управления курсором.

**Курсорные измерения**

В легенде каждого графика отображается значение графика в положении курсора

© ООО "ЭТМС" 1992-2024. Все права защищены




Узкополосный спектр, управление курсором и масштабирование графиков

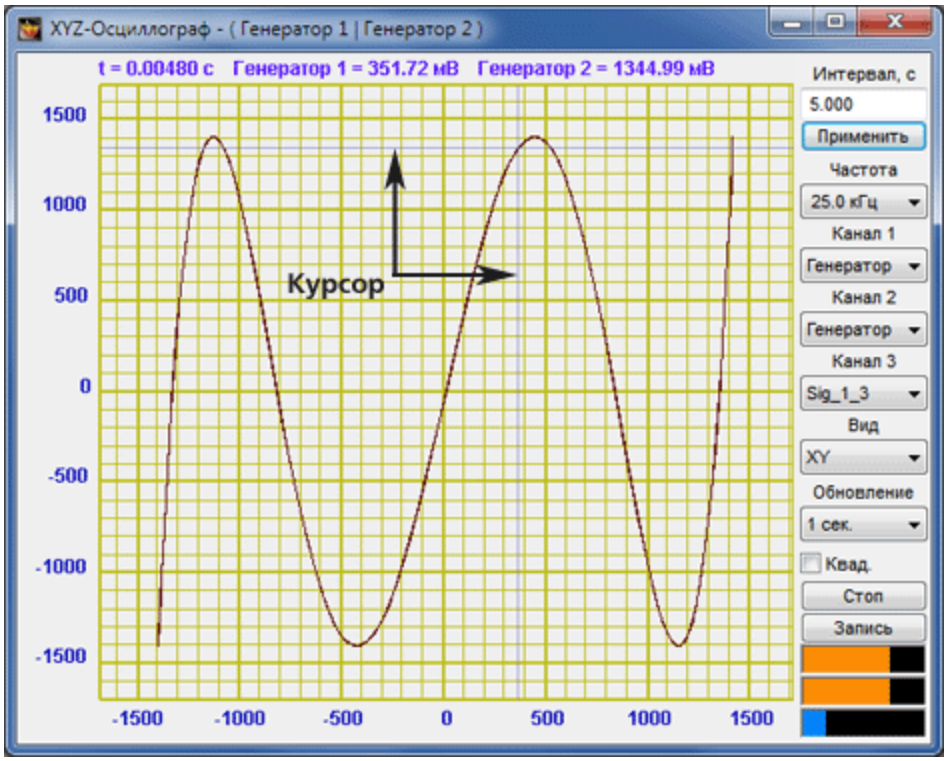
**Перемещение курсора**

Перемещение курсора графика на нужное время или нужную частоту осуществляется несколькими способами:

- подвести указатель «мыши» на нужное значение, нажать, и удерживая нажатой левую кнопку «мыши», дождаться пока курсор графика (вертикальная линия) не сравняется с установленным указателем «мыши». При нажатой и удерживаемой левой клавише «мыши» курсор графика будет следовать за перемещением указателя «мыши» по графику;
- при активном окне программы нажать левой кнопкой «мыши» на поле графика, и, при помощи ролика «мыши», перемещать курсор графика;
- при активном окне программы перемещение курсора влево осуществляется нажатием и удерживанием кнопки клавиатуры <A> (в латинской раскладке), вправо – <D>;
- при активном окне программы перемещение курсора вверх осуществляется нажатием и удерживанием кнопки клавиатуры <W> (в латинской раскладке), вниз – <S>.

В программах **XYZ-Осциллограф** и **XYZ-Плоттер** перемещение курсоров для плоскостей XT, YT и ZT осуществляется как описано выше. Перемещение курсора графика на нужное значение для плоскостей XY, YZ, XZ осуществляется следующим способом подвести указатель «мыши» на нужное значение, нажать, и удерживая нажатой левую кнопку «мыши», дождаться пока курсор графика (перекрещенные горизонтальная и вертикальная линии) не сравняется с установленным указателем «мыши». При нажатой и удерживаемой левой клавише «мыши» курсор графика будет следовать за перемещением указателя «мыши» по графику.

© ООО "ЭТМС" 1992-2024. Все права защищены



XYZ-Осциллограф

**Масштабирование графика**

Масштабирование числовых осей осуществляется при помощи манипулятора «мышь». При перемещении указателя «мыши» вдоль числовых значений осей указатель «мыши» будет принимать внешний вид в соответствии с предполагаемым действием масштабирования графика. После установки указателя нажать левой кнопкой «мыши», либо прокрутить ролик:

- растяжение или сжатие графика происходит при помощи указателей принявших вид: ↔, ⇹ – для горизонтальной оси и ↕, ⇳ – для вертикальной оси;
- сдвинуть график вправо/влево или вверх/вниз можно при помощи указателей принявших вид: ←, → – для горизонтальной оси и ↑, ↓ – для вертикальной оси;
- если поместить указатель «мыши» на пересечение числовых осей, то он примет вид ⤢. При нажатии левой клавиши «мыши» при указателе такого вида происходит автомасштабирование по оси уровня сигнала.

В программах **XYZ-Осциллограф** и **XYZ-Плоттер в** трехмерном виде (XYT, XZT, YZT и XYZ), а также в **3D-спектрограммах** сигнал (отношение сигналов) можно визуально рассматривать с любой стороны, вращая его вокруг трех взаимоперпендикулярных осей. Вращение вокруг трех взаимоперпендикулярных осей

© ООО "ЭТМС" 1992-2024. Все права защищены

осуществляется следующим образом – нажимая и удерживая левую кнопку «мыши», перемещая ее по полю графика вращать график вокруг любой из осей.

Увеличение или уменьшение вида XYT, XZT, YZT и XYZ осуществляется вращением ролика «мыши».

Двойное нажатие левой кнопки «мыши» по графическому полю вида XYT, XZT, YZT и XYZ возвращает график в исходное положение по отношению к осям и масштабу.

## Интегральный уровень сигнала

Индикатор  показывает интегральный уровень сигнала и, при превышении максимально допустимого уровня, перегрузку соответствующего канала. Две третьих части поля индикатора отведены для уровня сигнала, не превышающего максимально допустимый уровень. Чем выше уровень, тем больше заполняется индикатор. При превышении максимально допустимого уровня, индикатор заполняется полностью красным цветом. Правый край индикатора останется красным до тех пор, пока перегрузка по каналу не будет снята и пользователь не нажмет на индикатор левой кнопкой «мыши».

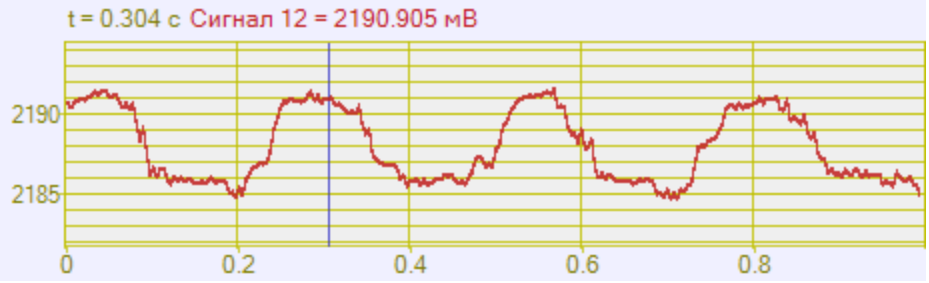
## Перенос графической и численной информации в текстовые редакторы

Результаты измерений программ ZETLAB, отображаемые в графическом виде, могут быть скопированы в текстовые редакторы.

### Копирование графика

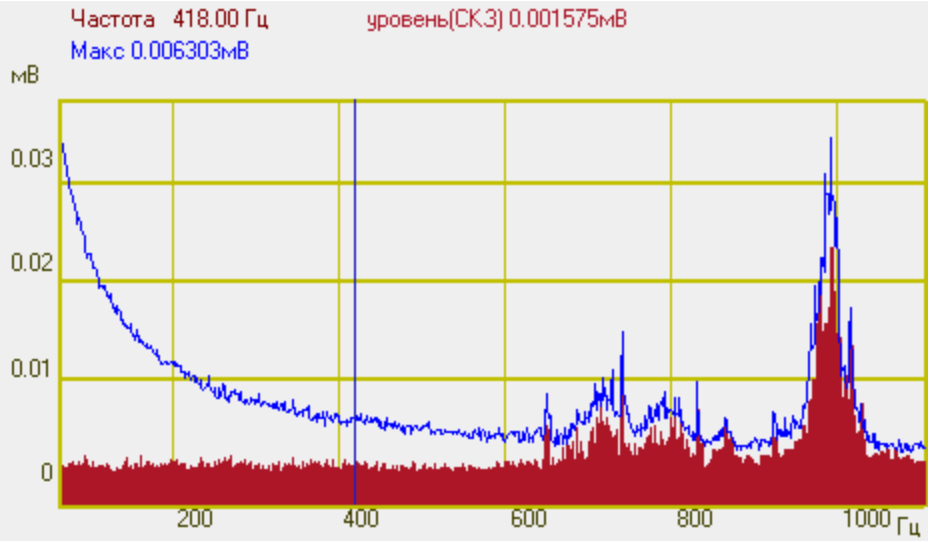
Для копирования графика нажать левой кнопкой «мыши» на поле графика и нажать комбинацию «горячих клавиш» клавиатуры <Ctrl> + <C>. График запишется в буфер обмена (*Clipboard*). Вставить график в любой открытый документ Microsoft Word или Excel можно нажатием «горячих клавиш» клавиатуры <Ctrl> + <V>, или нажатием на правую кнопку «мыши» и выбором в появившемся меню команды *Вставить*.

На рисунках ниже представлены примеры копирования графиков узкополосного спектра, 3D-спектрограммы и формы сигналов.

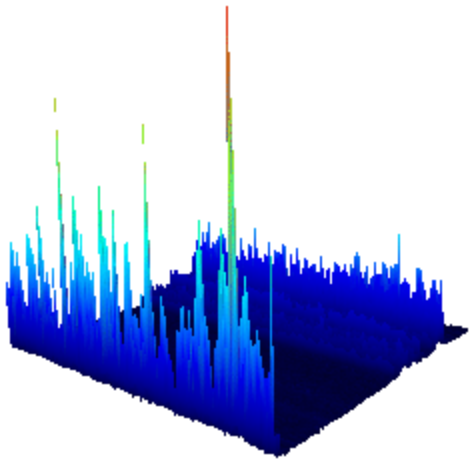



Копирование графика осциллографа

© ООО "ЭТМС" 1992-2024. Все права защищены




Копирование графика спектра



Копирование графика спектрограммы

**Копирование сопроводительной информации**

Для копирования сопроводительной информации нажать левой кнопкой «мыши» на поле графика программы и нажать на кнопку клавиатуры <T> (в латинской раскладке клавиатуры). Вставить эту информацию в любой открытый текстовый документ можно нажатием «горячих клавиш» клавиатуры <Ctrl> + <V>, или нажатием на правую кнопку «мыши» и выбором в появившемся меню команды *Вставить*.

Сопроводительная информация имеет следующую структуру: в первой строке пишется заголовок окна; во второй и последующих строках – измеряемые величины в положении курсора, т.е. легенда графика.

Пример 1: Узкополосный спектр с дополнительным графиком "Максимальный"

| Узкополосный спектр - Сигнал 9 | | | |
|---|---|---|---|
| Частота 449.64 Гц | | | |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | | | |
|---|---|---|---|
| уровень(СКЗ) 0.001131мВ | | | |
| Макс 0.004890мВ | | | |

Пример 2: Многоканальный осциллограф

| | | | |
|---|---|---|---|
| Многоканальный осциллограф | | | |
| t = 0.436 с | | | |
| Сигнал 9 = -8.030 мВ | | | |

**Копирование численных значений**

Для копирования численных значений видимой части графика нажать левой кнопкой «мыши» на поле графика и нажать на кнопку клавиатуры <N> (в латинской раскладке клавиатуры). Вставить эту информацию в любой открытый текстовый документ можно нажатием «горячих клавиш» клавиатуры <Ctrl> + <V>, или нажатием на правую кнопку «мыши» и выбором в появившемся меню команды *Вставить*.

Вставленная информация в текстовый документ будет иметь следующую структуру: сначала идет сопроводительная информация, в следующих строках будут располагаться значения графиков (в первом столбце - значения по оси абсцисс, в последующих - соответствующие значения графиков).

При копировании и вставки численных значений графиков в документы Excel, возможна обработка этой информации и построение графиков.

Пример 1: Узкополосный спектр с дополнительным графиком "Максимальный"

| | | | |
|---|---|---|---|
| Узкополосный спектр - Сигнал 1 | | | |
| Частота 950.0 Гц | | | |
| уровень(СКЗ) 0.010361мВ | | | |
| Макс -1000.000000мВ | | | |
| | | | |
| 10 | 0,109256 | -1000 | |
| 20 | 0,093899 | -1000 | |

Пример 2: Многоканальный осциллограф, режим совмещенного отображения 3-х сигналов

| | | | | |
|---|---|---|---|---|
| Многоканальный осциллограф | | | | |
| t = 0.4260 с | | | | |
| Сигнал 1 = 4.8 мВ | | | | |
| Сигнал 2 = -37.7 мВ | | | | |
| Сигнал 3 = -11.9 мВ | | | | |
| | | | | |
| 0,000000 | 4,535515 | 47,064354 | 56,736637 | |
| 0,000400 | 4,497148 | 41,935211 | 52,541870 | |
| 0,000800 | 4,495026 | 38,754341 | 50,536747 | |
| ... | | | | |

## Настройка внешнего вида графиков

При нажатии на правую кнопку «мыши» на поле графика в программах;

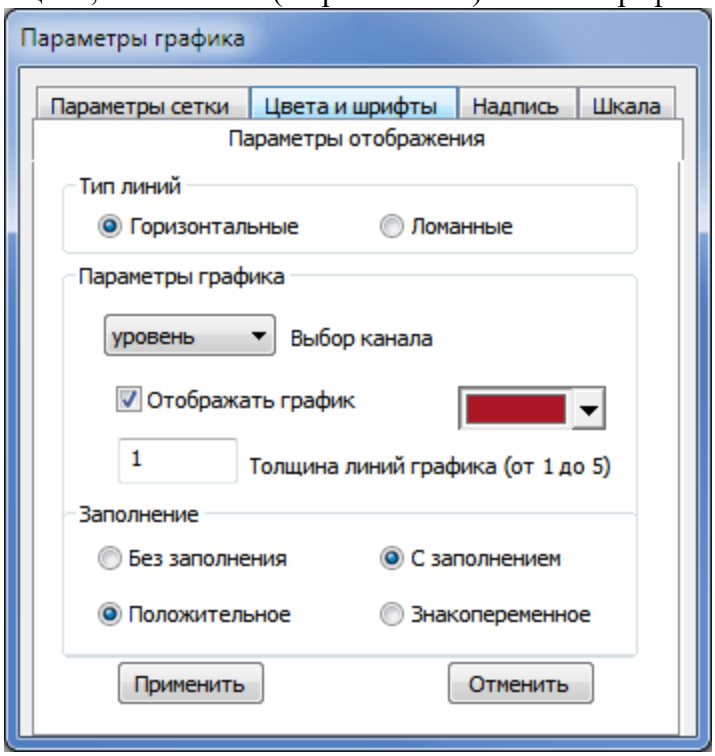
- Узкополосный спектр,
- Долеоктавный спектр,
- Взаимный узкополосный спектр,

© ООО "ЭТМС" 1992-2024. Все права защищены

- Взаимный долеоктавный спектр,
- Взаимный корреляционный анализ,
- Гистограмма

появляется дополнительное окно **Параметры графика**.

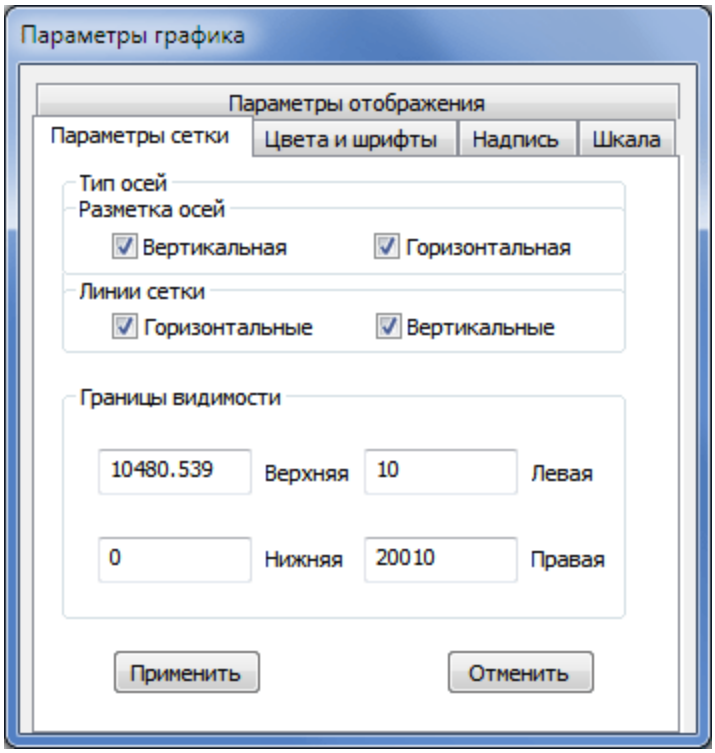
На вкладке *Параметры отображения* настраиваются тип линий и параметры графика. Типы линий графиков могут быть в виде горизонтальных (ступенек) или ломаных линий. В этой вкладке также устанавливаются параметры отображения каждого из графиков, цвет, толщина, заполнение (закрашивание) области графика.




Параметры отображения

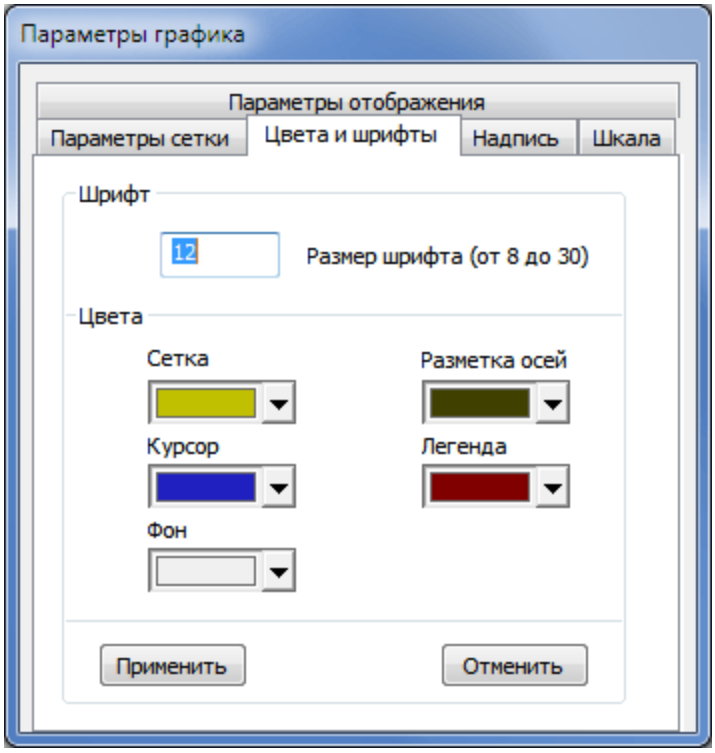
На вкладке *Параметры сетки* можно включать или отключать отображение горизонтальной и вертикальной разметки осей и линий сетки. В этой вкладке также задается область видимости (область отображения) графиков: верхняя, нижняя, правая и левая границы графиков.

© ООО "ЭТМС" 1992-2024. Все права защищены




Параметры сетки

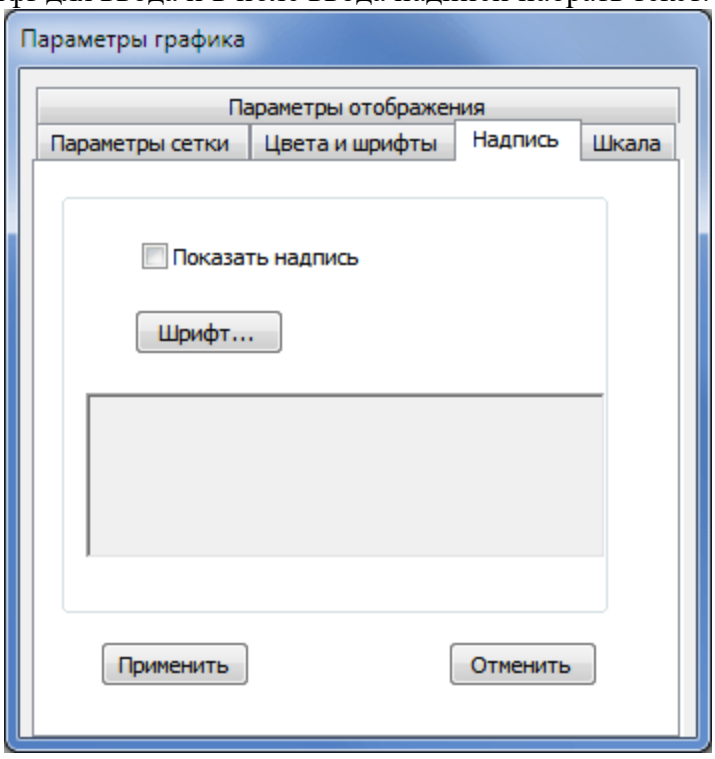
На вкладке *Цвета и шрифты* можно изменять размер шрифта числовых значений осей и измеряемых величин. В этой вкладке также задается цвет сетки, курсора, фона, разметки осей, легенды.




Цвета и шрифты

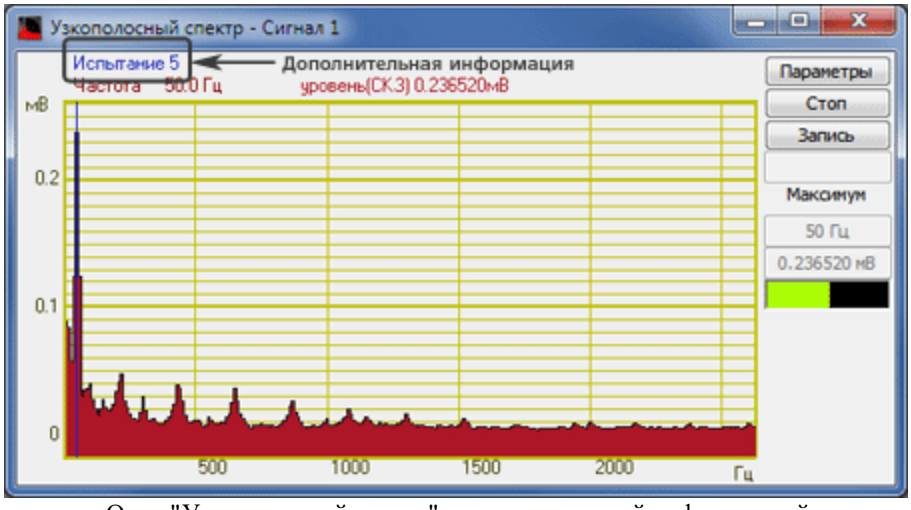
© ООО "ЭТМС" 1992-2024. Все права защищены

На вкладке *Надпись* можно записать дополнительную текстовую информацию, которая будет отображаться при копировании и вставки графика спектра в текстовый документ. Для записи этой информации необходимо поставить флажок Показать надпись, выбрать необходимый шрифт для ввода и в поле ввода надписи набрать текст.




Надпись

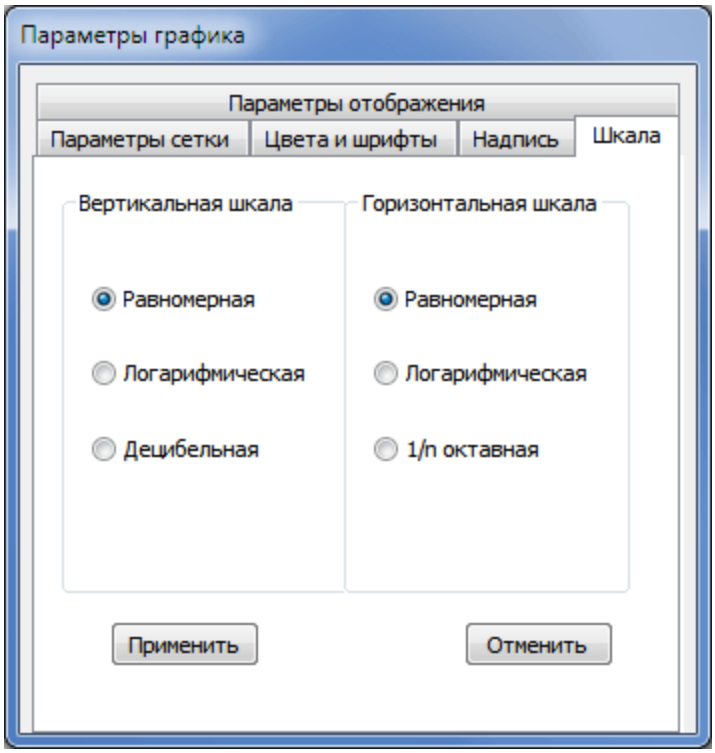
На рисунке ниже показано окна программы **Узкополосный спектр** с дополнительной информацией.




Окно "Узкополосный спектр" с дополнительной информацией

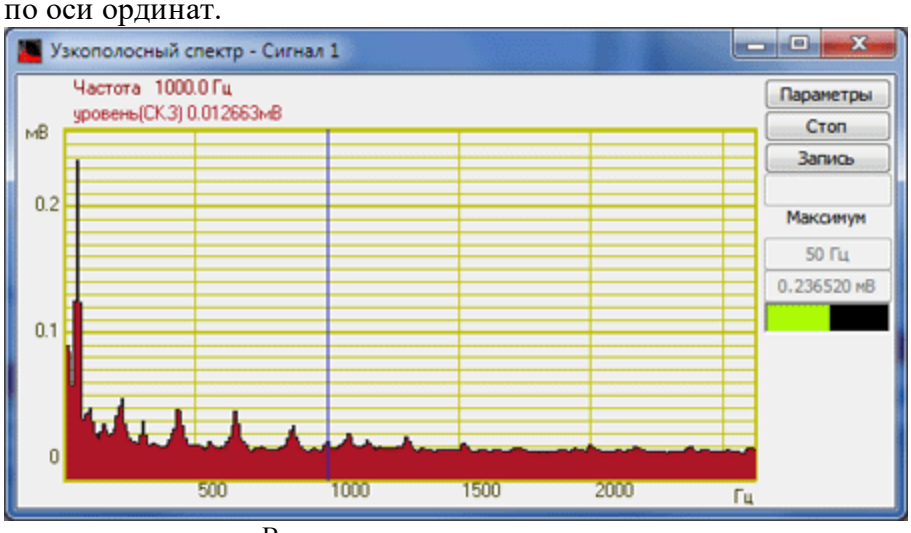
На вкладке *Шкала* можно выбрать тип представления горизонтальной и вертикальной шкал.

© ООО "ЭТМС" 1992-2024. Все права защищены



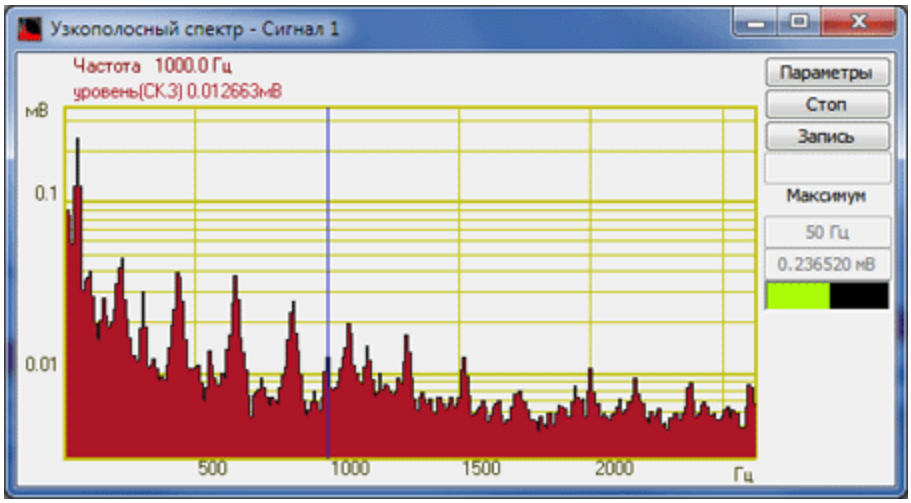

Шкала

Вертикальная шкала может быть представлена в равномерном, логарифмическом или децибельном виде. При отображении шкалы в децибельном виде используется опорное значение для вычисления уровня сигнала в дБ, указанное в параметрах измерительного канала. При этом значения графика не изменяются - изменяются только подписи по оси ординат. На рисунках ниже представлен спектр одного и того же сигнала в разных развертках по оси ординат.

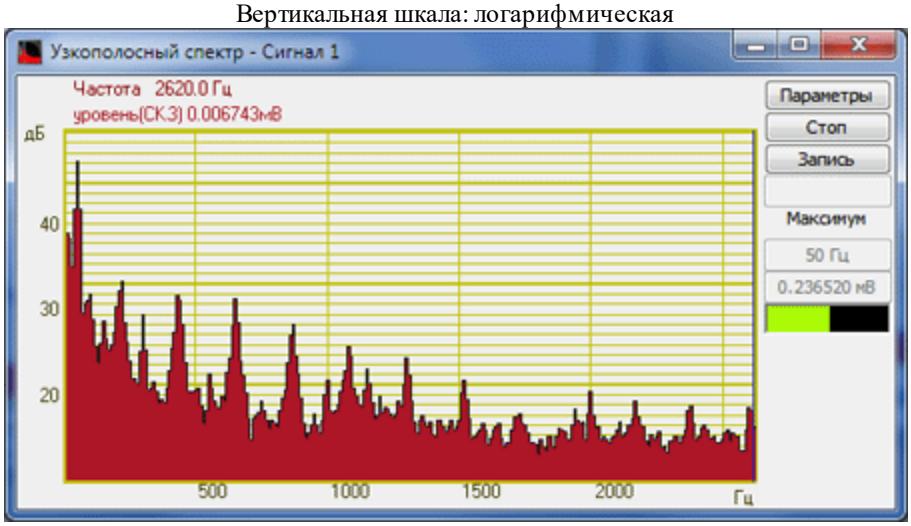



Вертикальная шкала: равномерная

© ООО "ЭТМС" 1992-2024. Все права защищены



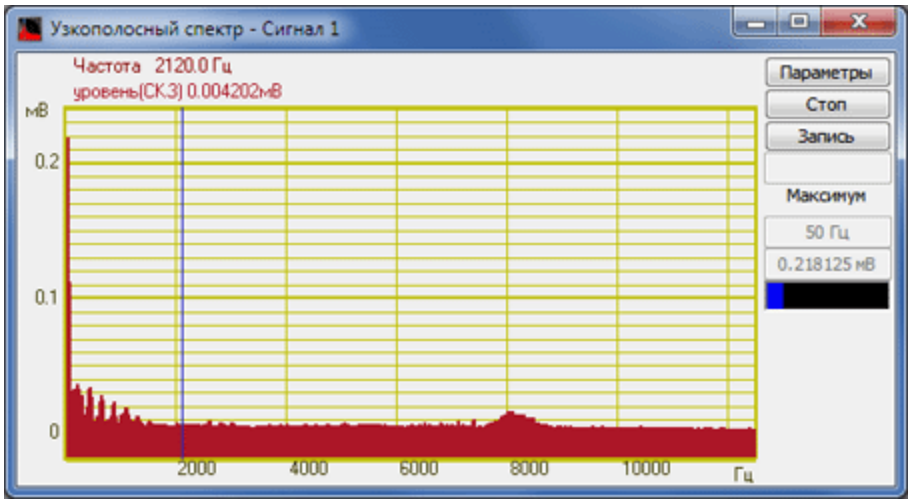

Вертикальная шкала: логарифмическая



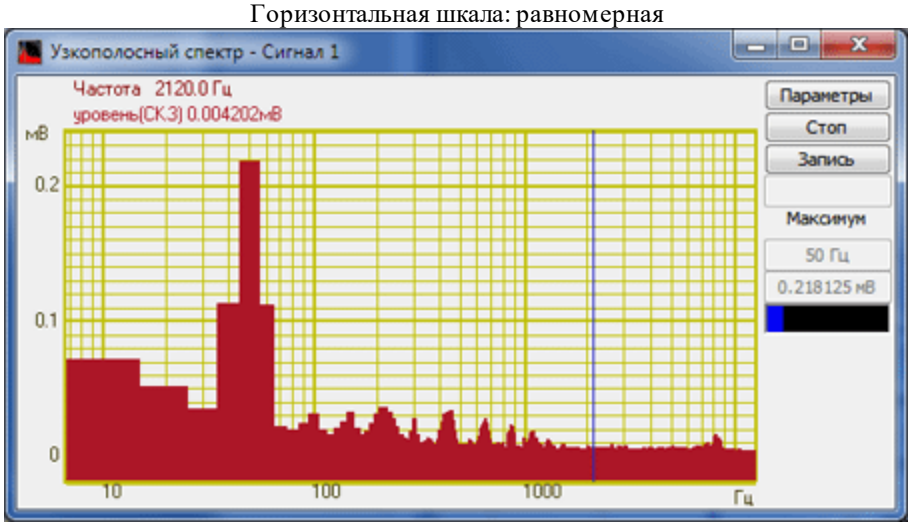

Вертикальная шкала: децибельная

Горизонтальная шкала может быть представлена в равномерном, логарифмическом или 1/n-октавном (долеоктавном) виде. При выборе неравномерной шкалы абсцисс значения не изменяются- изменяется только масштаб представления и/или подписи по оси. На рисунках ниже представлен спектр одного и того же сигнала в разных развертках по оси абсцисс.

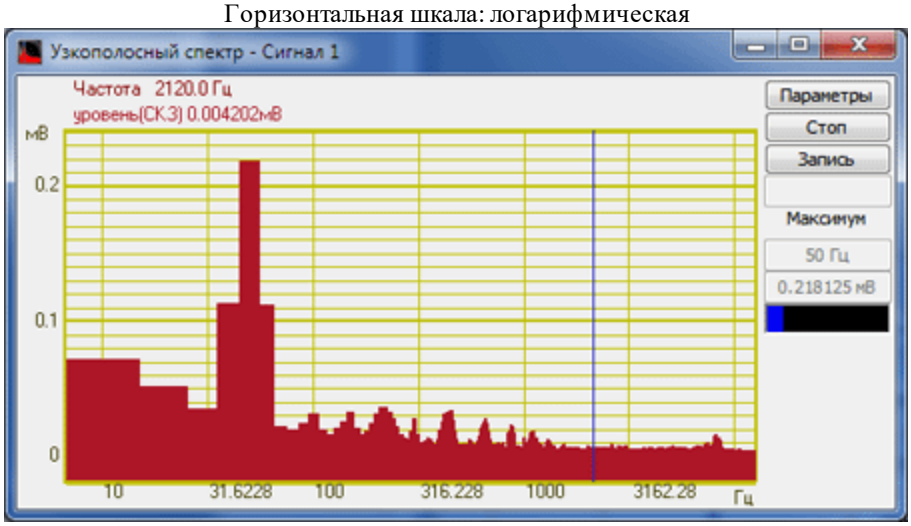
© ООО "ЭТМС" 1992-2024. Все права защищены




Горизонтальная шкала: равномерная




Горизонтальная шкала: логарифмическая




Горизонтальная шкала: 1/n октавная

Сохранение измененных настроек осуществляется нажатием на кнопку *Применить*, при этом окно **Параметры** закроется, а выбранные настройки вступят в силу.

© ООО "ЭТМС" 1992-2024. Все права защищены

Выход из окна **Параметры** без сохранения настроек осуществляется нажатием на кнопку *Отменить*, либо на кнопку "x", расположенную в правом верхнем углу окна, либо нажатием любой кнопкой «мыши» на любое место экрана, не занимаемое окном **Параметры**.

# Глава 6.Панель управления ZETLAB

Панель управления ZETLAB - это программа, позволяющая быстро найти нужный инструмент в виртуальной лаборатории ZETLAB. Панель управления ZETLAB также предоставляет пользователю средства для удобной работы: прямой доступ к записанным файлам и настройкам, функции авторазмещения окон программ и сохранения рабочих проектов, возможность работать в многоэкранном интерфейсе.
В процессе установки программного обеспечения ZETLAB на рабочем столе и в меню Пуск ОС Windows создается ярлык для запуска панели ZETLAB:




При запуске панели управления ZETLAB она располагается в верхней части рабочего стола ОС Windows, а ее значок - в системном трее.



Панель управления ZETLAB




Значок панели ZETLAB в системном трее

Панель управления ZETLAB представляет собой горизонтальную панель с главным меню, меню запуска различных программ и меню работы в многоэкранном интерфейсе:




Фрагмент панели ZETLAB

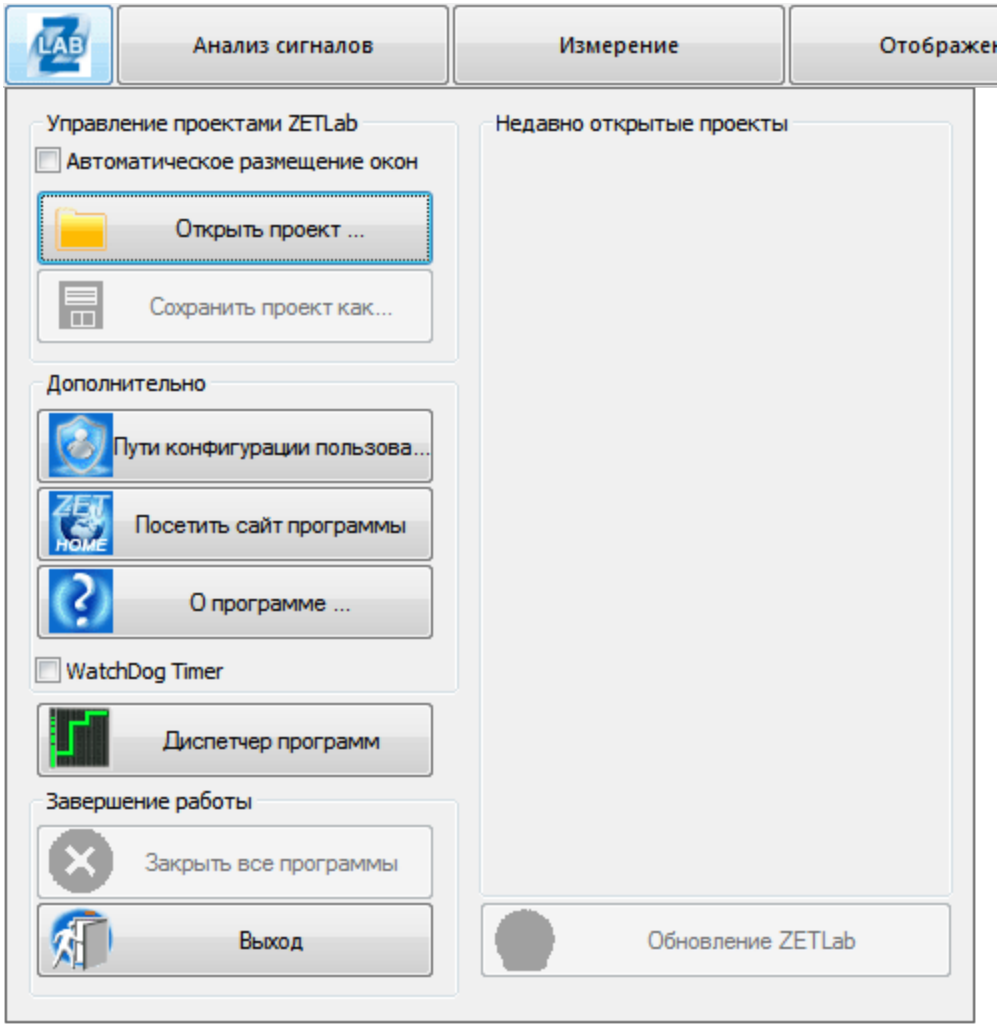
Количество меню запуска программ ZETLAB зависит от типа используемого ZET-прибора.

## 6.1.Главное меню панели ZETLAB

Главное меню панели управления ZETLAB позволяет:

- включать/отключать функцию автоматического размещения окон программ ZETLAB,
- сохранять и запускать сохраненные проекты,
- открыть директорию сохранения файлов конфигурации,
- посетить сайт http://zetlab.ru
- открыть информацию о программе,
- запустить WatchDog Timer,
- запустить диспетчер программ ZETLAB,
- закрыть все запущенные программы ZETLAB и панель управления ZETLAB.

© ООО "ЭТМС" 1992-2024. Все права защищены




Панель управления ZETLAB: главное меню

Автоматическое размещение окон

Функция автоматического размещения окон предназначена для автоматического размещения окон запускаемых пользователем программ ZETLAB на экране монитора, что позволяет экономить время при запуске большого числа программ ZETLAB и максимально эффективно использовать все рабочее пространство. В зависимости от количества и типа запускаемых программ их окна размещаются по всей полезной площади экрана монитора.

Существуют 2 типа окон программ:

- автоматически масштабируемые по всей полезной площади (например, программы типа **Многоканальный осциллограф**, **Узкополосный спектральный анализ**);
- имеющие привязку к левому верхнему краю экрана монитора (например, программы типа **Вольтметр переменного тока**, **Генератор сигналов**).

При выключенной функции *Автоматическое размещение окон* окна запускаемых программ располагаются в центре экрана. Пользователь по своему усмотрению может задавать размеры и положение окон программ на рабочем столе ОС Windows.

© ООО "ЭТМС" 1992-2024. Все права защищены

Функция *Автоматическое размещение окон* включается/отключается в главном меню панели ZETLAB.

**Проекты** ZETLAB

Проект ZETLAB - это набор программ ZETLAB. при сохранении проектов ZETLAB сохраняется не только перечень программ, но также их настройки и размещение.

Сохранение проектов ZETLAB и последующую их загрузку удобно использовать при большом количестве запущенных программ и настройке этих программ, а также при каждодневных однотипных измерениях. Один раз, запустив все необходимые программы и настроив их должным образом, сохраняется проект ZETLAB. В дальнейшем просто достаточно загрузить сохраненный ранее проект и все программы, которые были запущены и настроены перед сохранением проекта ZETLAB, будут запущенны с такими же настройками и расположением на экране (экранах), как и в момент сохранения этого проекта.

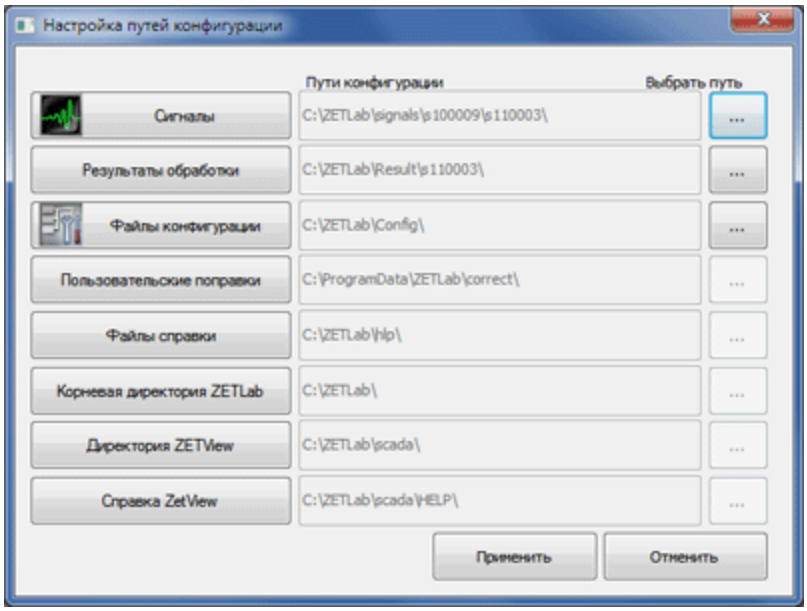
При сохранении проекта все запущенные программы, их настройки и параметры сохраняются в файл с расширением *.zpx. Можно записать несколько различных проектов ZETLAB в различные файлы. Проекты ZETLAB сохраняются как в одноэкранном, так и в многоэкранном режимах.

Для сохранения проекта необходимо в контекстном меню выбрать команда *Сохранить проект*, после чего откроется окно **Сохранить проект…**. В окне **Сохранить проект…** необходимо ввести имя проекта (имя файла) и нажать кнопку *Сохранить*, после чего проект будет сохранен в указанную директорию. По умолчанию имя файла проекта Project_01.zpr. Директория по умолчанию – C:\ZETLab\config\. Пользователь может сам назначать директорию для сохранения проектов, но при каждом новом сохранении проекта ZETLAB будет предложена для сохранения директория по умолчанию.

Файлы конфигурации пользователей

Файлы конфигурации пользователей отображаются в окне **Настройка путей конфигурации**, которая запускается командой *Пути конфигурации* из главного меню панели ZETLAB. Путь к файлам отображается в поле *Пути конфигурации*, слева от пути располагается кнопка, позволяющая открыть соответствующую директорию, справа - кнопка, позволяющая ее изменить (выбрать другую папку).

© ООО "ЭТМС" 1992-2024. Все права защищены




Настройка путей конфигурации пользователей

Директория *Сигналы* предназначена для сохранения файлов сигналов, записываемых программой **Запись сигналов**. Сигналы сохраняются в двоичном виде (файл в формате "ana") с паре файлом-описателем (в формате "anp").

Директория *Результаты обработки* предназначена для сохранения файлов результатов работы программ ZETLAB. Файлы сохраняются в формате "dtu" и могут открываться программами **Галерея сигналов** и **Просмотр результатов**, а также любым текстовым редактором.

В директории *Файлы конфигурации* хранятся файлы с настройками программ ZETLAB. Это значительно упрощает работу с ZETLAB, поскольку не требует от пользователя ежедневной настройки программ - даже при использовании нескольких экземпляров программ, при их последующем запуске каждая программа "помнит" свои настройки.

В директории *Пользовательские поправки* хранятся файлы калибровки измерительных каналов и каналов генераторов используемого ZET-прибора.

Директория *Файлы справки* содержит файл-справку (данное руководство оператора в формате "chm") по программному обеспечение ZETLAB и примеры, используемые в данном руководстве оператора.

В *корневой директории ZETLAB* хранятся файлы всех программ ZETLAB и используемым ими компонентам.

В *директории ZETVIEW* хранятся файлы, используемые SCADA системой ZETVIEW, а также примеры программ, реализованные в ZETVIEW (запускаются из меню "Специальные программы" панели управления ZETLAB).

В директории *Справка ZETVIEW* содержится файл-справка по ZETVIEW (руководство пользователя) с используемыми примерами.

*Примечание: SCADA система ZETVIEW поставляется с ZET-прибора опционно.*

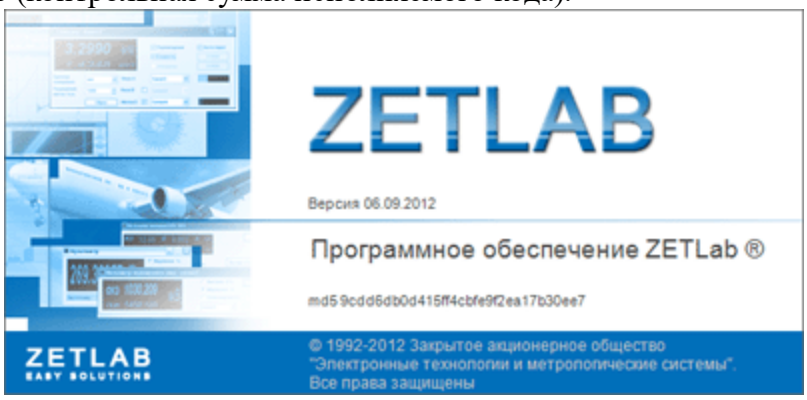
Сайт программы

Команда *Посетить сайт программы* запускает браузер, используемый по умолчанию, и переходит на сайт производителя продукции ZETLAB: http://zetlab.ru.

О программе

© ООО "ЭТМС" 1992-2024. Все права защищены

Функция *О программе* главного меню панели ZETLAB вызывает окно с информацией о программном обеспечении ZETLAB: обозначение, номер версии, цифровой идентификатор (контрольная сумма исполняемого кода).



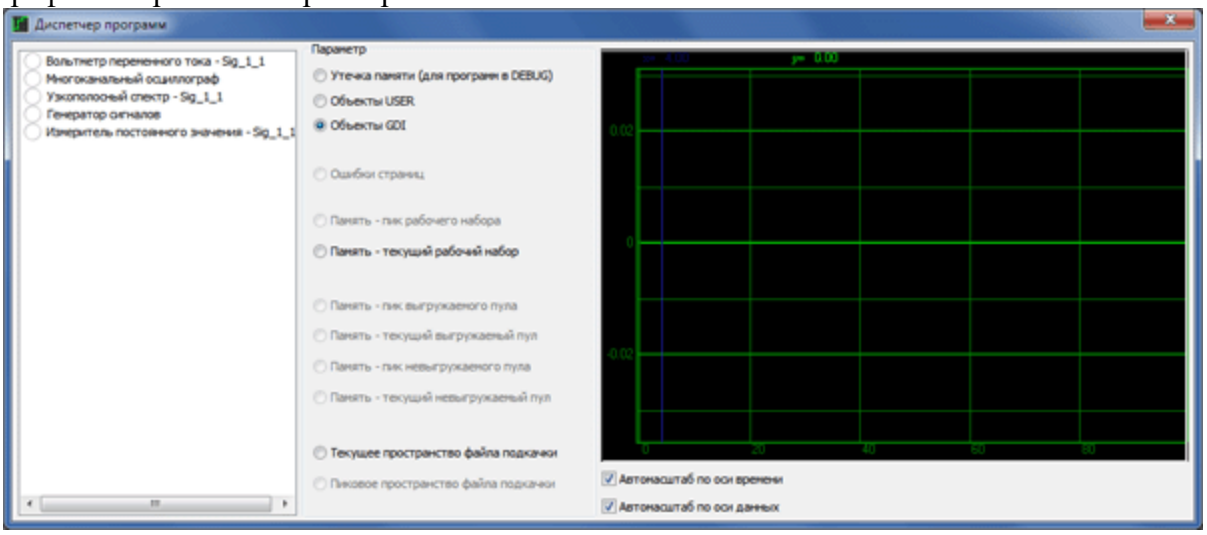

Окно "О программе"

WatchDog Timer

Программный сторожевой таймер (WatchDog Timer) используется в автоматизированных системах мониторинга (непрерывного контроля) параметров сигналов. В случае зависания какой-либо программы ZETLAB, сторожевой таймер перезагружает компьютер и вновь запускает все необходимые программы.

Диспетчер программ

Диспетчер программ ZETLAB предназначен для мониторинга состояния памяти и других параметров системы, связанных с работой программ ZETLAB. В левой части окна программы отображается список запущенных программ ZETLAB, в правой - график выбранного параметра.




Диспетчер программ ZETLAB

Программа находится в стадии разработки.

Завершение работы

Главное меню панели ZETLAB также содержит команды *Закрыть все программы* и *Выход*. Функция *Закрыть все программы* закрывает все запущенные программы ZETLAB, кроме панели ZETLAB. Команда *Выход* закрывает все программы ZETLAB, включая панель ZETLAB.

© ООО "ЭТМС" 1992-2024. Все права защищены

## 6.2.Меню запуска программ

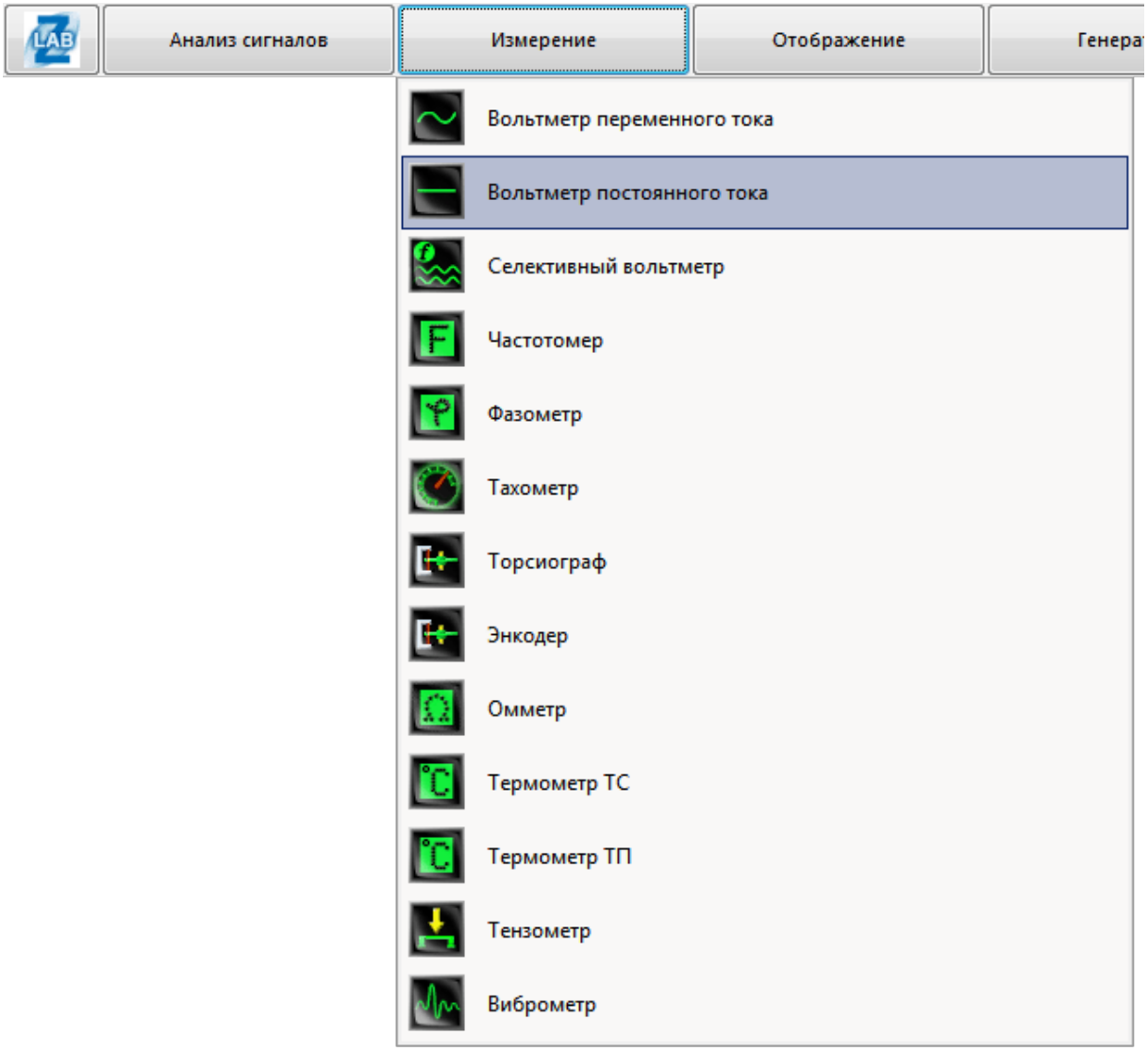
Основную часть панели ZETLAB составляют меню запуска программ ZETLAB: виртуальных приборов, сервисных и сетевых программ, программ, реализованных в SCADA системе ZETVIEW.

Запуск программ

Для запуска какой-либо программы нужно нажать левой кнопкой «мыши» на название соответствующего меню (группы программ), из развернувшегося списка программ этого раздела выбрать нужную и нажать на нее левой кнопкой «мыши».

Каждое меню группы программ содержит список программ, отображаемый при нажатии левой клавишей «мыши» на кнопку вызова меню группы программ на панели ZETLAB. В раскрывающемся меню группы программ рядом с названиями программ находятся пиктограммы, позволяющие быстро связать команду запуска программы с ее графическим изображением. При наведении курсора на название программы появляется подсказка с кратким описанием программы.

© ООО "ЭТМС" 1992-2024. Все права защищены




Панель управления ZETLAB: меню "Измерение"

Перечень меню, и перечень программ в каждом меню зависит от типа подключаемого ZET-прибора и определяется списком поставляемых с ним программ (базовых и опционных). Панель ZETLAB может иметь следующие меню:

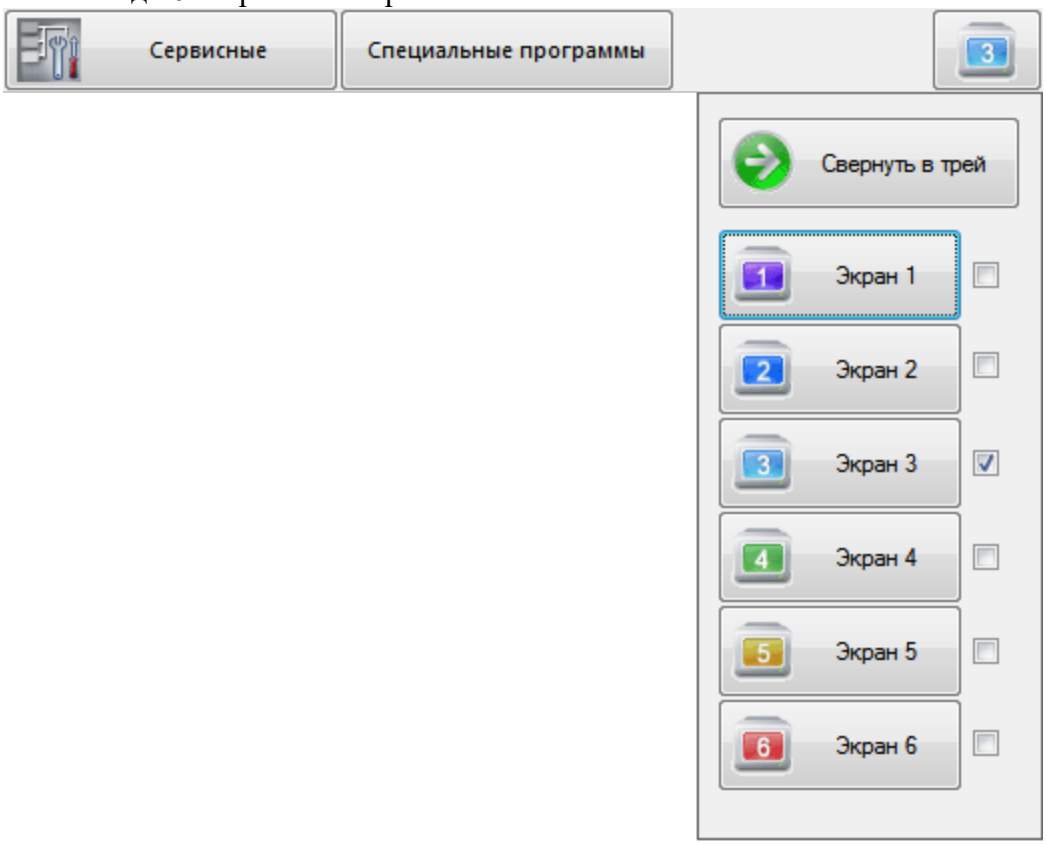
- Главное - предоставляет быстрый доступ к файлам конфигурации и записанным сигналам, позволяет сохранять/загружать проекты, содержит функции автоматического размещения окон и сторожевого таймера.
- Анализ сигналов - программы анализа сигналов с применением различных алгоритмов: преобразование Фурье, Вейвлет-анализ, преобразование STA/LTA, статический анализ, корреляционный анализ и др.
- Измерение - программы измерения параметров сигналов.
- Отображение - программы визуализации сигналов, результатов измерений и работы с записанными данными.
- Генераторы - программы генерации сигналов.
- Регистрация - программы записи и воспроизведения.

© ООО "ЭТМС" 1992-2024. Все права защищены

- Метрология - программы снятия АЧХ и ФЧХ (в версия ZETLAB начиная с 06/11 не поддерживаются, реализованы аналогичные проекты в ZETVIEW)
- Автоматизация - программы фильтрации, регулирования, управления.
- Сетевые программы - программы приема/передачи данных по сети.
- Сервисные - настроечные средства.
- Специальные программы - проекты, реализованные в SCADA ZETVIEW
- Работа с экранами - меню управления функцией Многоэкранный интерфейс 118.

## 6.3.Многоэкранный интерфейс

Функция *Многоэкранный интерфейс* предназначена для создания эффекта использования до 6-ти рабочих экранов.




Многоэкранный интерфейс

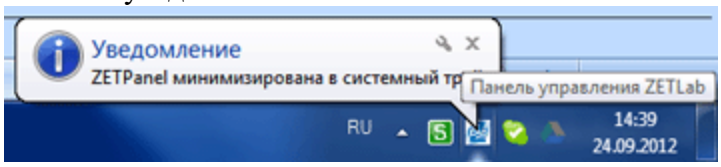
Меню переключения экранов располагается в правой части панели ZETLAB. Задействованный экран отмечен флагом, все запускаемые программы отображаются в нем. При переходе на другой экран, окон программ, запущенных на других экранах, не видно, в том числе не отображаются значки программ в панели задач ОС Windows. В этом и заключается удобство работы в многоэкранном интерфейсе - программы, предназначенные для предварительной обработки сигналов (например, **Фильтрация сигналов**) или измерения параметров сигналов (например, **Виброметр**), можно запустить на одном экране, а **Многоканальный осциллограф**, в котором отображаются результаты измерений - в другом. Таким образом, программы,

© ООО "ЭТМС" 1992-2024. Все права защищены

необходимые для обработки сигналов, но не несущие пользователю полезной информации, могут работать в "скрытом" режиме и сгруппированы по экранам в зависимости от своего назначения и функций, выполняемых в конкретном случае. А программы, с которыми оператор ведет непосредственную работу, выведены на отдельный экран, в котором используется максимум полезной площади рабочего стола ОС Windows.

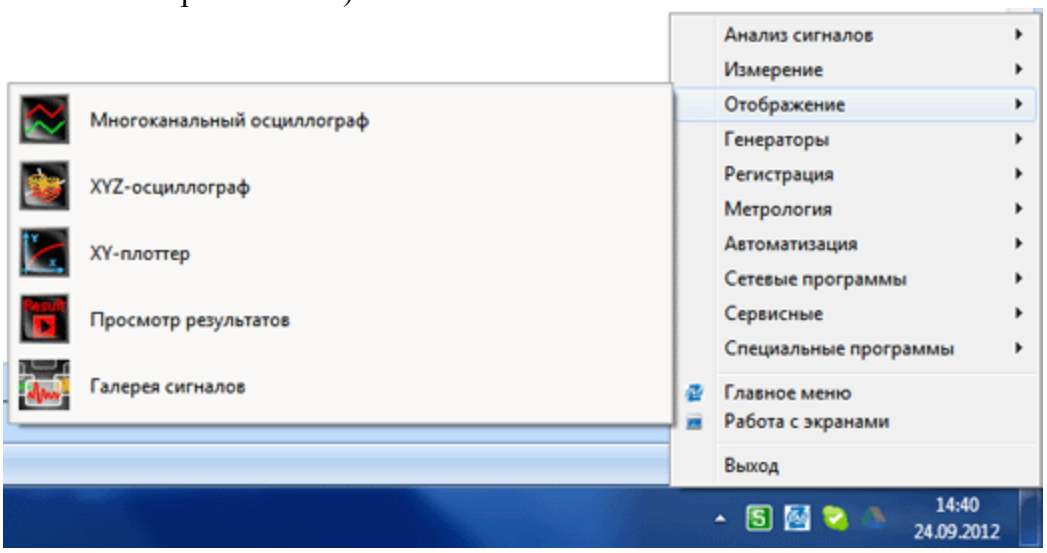
Свернуть в трей

При выборе команды *Свернуть в трей* панель ZETLAB сворачивается в системный трей, при этом появляется уведомление:




Уведомление о минимизации панели ZETLAB в системный трей

При нажатии на значок панели ZETLAB в системном трее открывается меню панели ZETLAB в компактном виде (меню вызывается вне зависимости от того, свернута панель ZETLAB в трей или нет):




Меню панели ZETLAB, открываемое из системного трея

Для восстановления положения панели ZETLAB в верхней части рабочего стола ОС Windows, необходимо в меню панели ZETLAB, вызываемом из системного трея, выбрать пункт *Работа с экранами*, и в открывшемся окне команду *Развернуть программу*.

© ООО "ЭТМС" 1992-2024. Все права защищены




© ООО "ЭТМС" 1992-2024. Все права защищены

# Доступ к драйверам ZET через библиотеку Zadc.dll

Пользователь может работать с устройствами фирмы "ЭТМС" с помощью вызовов функций библиотеки ZADC.dll.

## Глава 1.Полное описание функций Zadc.dll

Пользователь может работать с устройствами фирмы "ЗЭТ" с помощью вызовов функций библиотеки ZADC.dll.

### 1.1.Работа с устройствами фирмы ZET

### Работа с устройствами фирмы ZET

Пользователь может работать с устройствами фирмы ZET с помощью вызовов функций библиотеки Zadc.dll. Данное руководство приведено для библиотеки Zadc.dll с версией 5.0 и старше.

Библиотека написана на C++ и имеет интерфейс WINAPI (Microsoft® Windows® application programming interface (API)). Все объявления в настоящем руководстве приведены на языке C/C++. Все функции библиотеки Zadc.dll возвращают код ошибки. Значение 0 – говорит о выполнении функции без ошибок.

Функции Zadc.dll имеют вид ZXXX(…). Если функция влияет на работу только АЦП, то она заканчивается на ADC, если функция влияет на работу только ЦАП – она заканчивается на DAC. Общие функции не имеют специфичных обозначений.

Функции делятся на две категории: информационные и управляющие, информационные функции, возвращают в программу пользователя различные параметры и не меняют режим работы драйвера, сигнального процессора и модулей АЦП и ЦАП. Управляющие программы меняют режим работы драйвера и устройств, подключенных к драйверу. При этом в драйвере устанавливается признак изменения режима работы (*modify*). Если несколько программ одновременно работают с одним драйвером, то каждая программа должна при каждом обращении к драйверу считывать этот признак и должна менять свой режим работы: обработки сигналов, отображения и пр.

Структурная схема управления устройством на примере KADSP/PCI представлена на рисунке 1.1

© ООО "ЭТМС" 1992-2024. Все права защищены

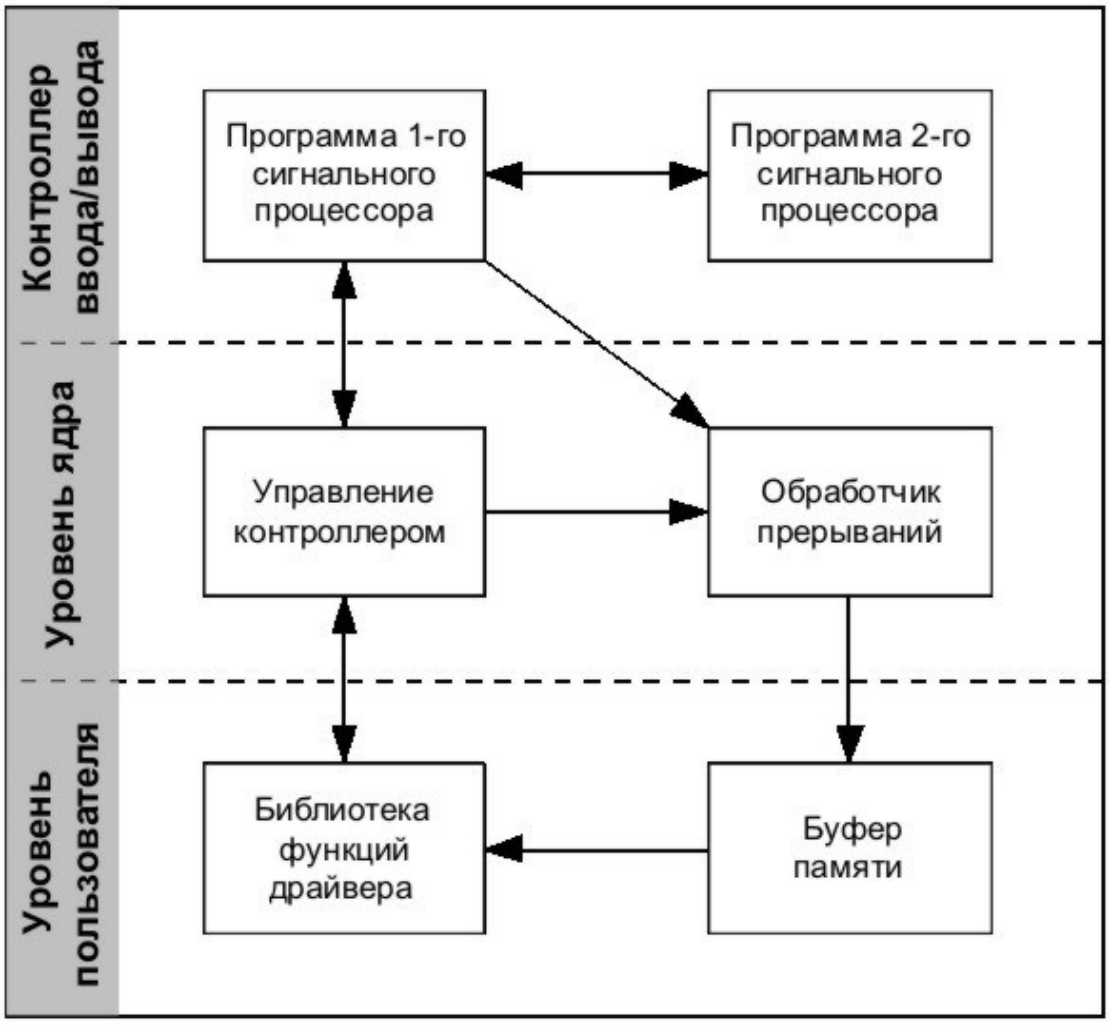



Рисунок 1.1

Сигнальный процессор выполняет следующие функции:

- устанавливает коэффициенты усиления на модуле АЦП и коэффициент затухания на модуле ЦАП;
- устанавливает частоту дискретизации на модулях АЦП и ЦАП;
- обеспечивает синхронизацию нескольких сигнальных процессоров;
- по завершению преобразования производит опрос выбранных каналов АЦП и формирует массив данных во внутренней памяти сигнального процессора;
- проводит первичную цифровую фильтрацию и прореживание оцифрованных входных сигналов;
- проводит каскадную декадную фильтрацию и прореживание
- устанавливает и снимает сигнал прерывания для центрального процессора при заполнении внутреннего буфера памяти;
- обеспечивает перекачку данных из внутренней памяти в память центрального процессора порции данных, определяемой драйвером как размер буфера для перекачки.

© ООО "ЭТМС" 1992-2024. Все права защищены

Драйвер на уровне ядра обеспечивает обмен управляющими данными с сигнальным процессором. Программа обработчика прерываний от сигнального процессора обеспечивает прием оцифрованных сигналов и размещение их в буфере памяти. Для каждого сигнального процессора создается свой буфер данных.

На уровне пользователя функционирует библиотека подпрограмм для связи пользовательских программ с частью драйвера, расположенного на уровне ядра. Драйвер поддерживается операционными системами Windows 2000 и Windows XP, Windows 2003, Windows Vista, Windows 7, Server 2003, Server 2008, Starter.

Индексы контроллеров PCI, как правило, распределяются последовательно в направлении от разъема AGP к краю системной платы. Индексы контроллеров USB зависят от порядка подключения и от номера порта USB.

В системе может быть установлено несколько контроллеров KADSP/PCI или KADSP/PDP, к каждому сигнальному процессору модуля может быть подключен только один модуль АЦП или ЦАП. Дополнительно, к любому из сигнальных процессоров, управляющих модулем АЦП, можно подключить усилитель заряда ПУ 8/10 и управлять его программируемым коэффициентом усиления. Два сигнальных процессора на плате KADSP/PCI соединены по схеме ведущий/ведомый. Оба процессора работают на одной тактовой частоте от одного тактового генератора. Ведущим процессором на плате является процессор, расположенный справа и ближе к краю платы (см. рисунок в РЭ). Соответственно справа расположен и 50-контактный разъем для подключения модулей АЦП и ЦАП. Ведомым процессором на плате является процессор, расположенный слева и ближе к разъемам «лемо». Соответственно слева расположен и 50-контактный разъем для подключения модулей АЦП и ЦАП.

Для удобства программирования реализована сквозная адресация сигнальных процессоров и подключенных к ним модулей. Ключевым параметром во всех функциях является номер сигнального процессора (*numberDSP*).

## 1.2.Подключение к драйверу и отключение

Все программы, которые используют функции Zadc.dll, должны начинаться и заканчиваться процедурами из этого раздела. При этом никаких действий связанных с модулями АЦП, ЦАП и сигнальными процессорами не происходит. Это позволяет подключаться одновременно нескольким программам к одним модулям и сигнальным процессорам. Все текущие настройки сигнального процессора и модулей АЦП и ЦАП сохраняются во внутренних структурах драйвера. Для того чтобы определить, сколько установлено сигнальных процессоров в системе, необходимо выполнить 137 попыток подключения к драйверу с номерами сигнальных процессоров от 0 до 136 для соответствующего типа устройства. Количество удачных попыток подключения будет информировать о количестве установленных сигнальных процессоров в системе.

© ООО "ЭТМС" 1992-2024. Все права защищены

## Подключение к драйверу

***ZOpen -*** подключение к драйверу. При работе с платами АЦП и ЦАП – это первая функция, к которой необходимо произвести обращение.

*long* ***ZOpen*** *(long typeDevice, long numberDSP)*

Параметры:

*typeDevice* – тип устройства. Может принимать значения:
0 – тип устройства ADC 16/200, название драйвера Kd1610.sys;
1 – тип устройства APC 216, название драйвера Kd216.sys;
2 – тип устройства ADC 16/500, название драйвера Kd500.sys;
3 – тип устройства ADC 16/500P, название драйвера Kd500p.sys;
4 – тип устройства ADC 816, название драйвера Kd816.sys;
5 – тип устройства ADC 1002, название драйвера Kd1002.sys;
6 – тип устройства ADC 216 USB, название драйвера Kdu216.sys;
7 – тип устройства ADC 24, название драйвера Kd24.sys;
8 – тип устройства ADC 1432, название драйвера Kd1432.sys;
9 – тип устройства ACPB USB, название драйвера KduACPB.sys;
10 – тип устройства ZET210 USB, название драйвера Kdu1616.sys;
11 – тип устройства PD14 USB, название драйвера KduPD14.sys;
12 – тип устройства ZET110 VN USB, название драйвера KduVN.sys;
13 – тип устройства ZET302 Osc USB, название драйвера KduOsc.sys;
14 – тип устройства ZET017 U8 USB, название драйвера Kdu8500.sys;
15 – тип устройства ZET017-U2 (A19-U2 USB), название драйвера Kdu2500.sys;
16 – тип устройства ZET220 USB (ZET017 USB Seismic), название драйвера Kdu1624.sys (24 – разряда, 16 – каналов);
17 – тип устройства ZET230 USB (ZET017 USB Audio), название драйвера Kdu0424.sys (24 – разряда, 4 – канала);
18 – тип устройства ZET048 USB, название драйвера Kdu0414.sys;
19 – тип устройства ZET240 USB (ZET048 USB Seismic), название драйвера Kdu0824.sys (14 – разрядов, 4 канала).

*numberDSP* – номер сигнального процессора (значение от 0 до 136).

Возвращаемое значение:
0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800, 0x0900, 0x0A00, 0x0B00 – ошибка открытия драйвера,
0x0C00 – неподдерживаемая версия драйвера.

Функция не меняет признак *modify* в драйвере.

© ООО "ЭТМС" 1992-2024. Все права защищены

## Отключение от драйвера

***ZClose -*** отключение от драйвера (сигнального процессора). Эта функция – последнее обращение к драйверу перед выходом из программы.

*long* ***ZClose*** *(long typeDevice, long numberDSP)*

Параметры:

*typeDevice* – тип устройства. Может принимать значения:

0 – тип устройства ADC 16/200, название драйвера Kd1610.sys;
1 – тип устройства APC 216, название драйвера Kd216.sys;
2 – тип устройства ADC 16/500, название драйвера Kd500.sys;
3 – тип устройства ADC 16/500P, название драйвера Kd500p.sys;
4 – тип устройства ADC 816, название драйвера Kd816.sys;
5 – тип устройства ADC 1002, название драйвера Kd1002.sys;
6 – тип устройства ADC 216 USB, название драйвера Kdu216.sys;
7 – тип устройства ADC 24, название драйвера Kd24.sys;
8 – тип устройства ADC 1432, название драйвера Kd1432.sys;
9 – тип устройства ACPB USB, название драйвера KduACPB.sys;
10 – тип устройства ZET210 USB, название драйвера Kdu1616.sys;
11 – тип устройства PD14 USB, название драйвера KduPD14.sys;
12 – тип устройства ZET110 VN USB, название драйвера KduVN.sys;
13 – тип устройства ZET302 Osc USB, название драйвера KduOsc.sys;
14 – тип устройства ZET017 USB, название драйвера Kdu8500.sys;
15 – тип устройства ZET017-U2 (A19-U2 USB), название драйвера Kdu2500.sys;
16 – тип устройства ZET220 USB (ZET017 USB Seismic), название драйвера Kdu1624.sys (24 – разряда, 16 – каналов);
17 – тип устройства ZET230 USB (ZET017 USB Audio), название драйвера Kdu0424.sys (24 – разряда, 4 – канала);
18 – тип устройства ZET240 USB, название драйвера Kdu0414.sys;
19 – тип устройства ZET240 USB (ZET048 USB Seismic), название драйвера Kdu0824.sys (14 – разрядов, 4 канала).

*numberDSP* – номер сигнального процессора (значение от 0 до 136).

Возвращаемое значение:
0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка закрытия драйвера,

Функция не меняет признак *modify* в драйвере.

© ООО "ЭТМС" 1992-2024. Все права защищены

## 1.3.Сброс и инициализация

### Сброс и останов сигнального процессора

***ZResetDSP -*** сбрасывает все сигнальные процессоры одного устройства и останавливает. После вызова данной функции для нормальной работы с сигнальными процессорами нужно их проинициализировать с помощью ZInitDSP(). Например, плата KADSP/PCI содержит два процессора, поэтому для ее инициализации нужно вызвать один раз ZResetDSP() (для любого из двух процессоров), а затем вызвать два раза ZInitDSP() для каждого процессора.

*long **ZResetDSP** (long typeDevice, long numberDSP)*

Параметры:

*typeDevice* – тип устройства.

0 – тип устройства ADC 16/200, название драйвера Kd1610.sys;
1 – тип устройства APC 216, название драйвера Kd216.sys;
2 – тип устройства ADC 16/500, название драйвера Kd500.sys;
3 – тип устройства ADC 16/500P, название драйвера Kd500p.sys;
4 – тип устройства ADC 816, название драйвера Kd816.sys;
5 – тип устройства ADC 1002, название драйвера Kd1002.sys;
6 – тип устройства ADC 216 USB, название драйвера Kdu216.sys;
7 – тип устройства ADC 24, название драйвера Kd24.sys;
8 – тип устройства ADC 1432, название драйвера Kd1432.sys;
9 – тип устройства ACPB USB, название драйвера KduACPB.sys;
10 – тип устройства ZET210 USB, название драйвера Kdu1616.sys;
11 – тип устройства PD14 USB, название драйвера KduPD14.sys;
12 – тип устройства ZET110 VN USB, название драйвера KduVN.sys;
13 – тип устройства ZET302 Osc USB, название драйвера KduOsc.sys;
14 – тип устройства ZET017 USB, название драйвера Kdu8500.sys;
15 – тип устройства ZET017-U2 (A19-U2 USB), название драйвера Kdu2500.sys;
16 – тип устройства ZET220 USB (ZET017 USB Seismic), название драйвера Kdu1624.sys (24 – разряда, 16 – каналов);
17 – тип устройства ZET230 USB (ZET017 USB Audio), название драйвера Kdu0424.sys (24 – разряда, 4 – канала);
18 – тип устройства ZET240 USB, название драйвера Kdu0414.sys;
19 – тип устройства ZET240 USB(ZET048 USB Seismic), название драйвера Kdu0824.sys (14 – разрядов, 4 канала).

*numberDSP* – номер сигнального процессора (значение от 0 до 136).

Возвращаемое значение:

© ООО "ЭТМС" 1992-2024. Все права защищены

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,

Функция меняет признак *modify* в драйвере.

## Инициализация сигнального процессора

***ZInitDSP -*** инициализирует и загружает в сигнальный процессор исполняемую программу (если устройство не содержит прошитую программу). При выключении питания программа в памяти сигнального процессора стирается. Для функционирования программ достаточно один раз загрузить программу после включения питания.

*long* ***ZInitDSP*** *(long typeDevice, long numberDSP, char *fileName)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*fileName* – строка с именем файла для загрузки. Если строка пустая, то загружается исполняемая программа по умолчанию (файл *.*ios*, поставляемый с драйвером). Если строка не пустая, то загружается файл, указанный в строке *fileName* из системной папки Windows, где находятся драйвера.

Возвращаемое значение:
0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0200 – *filename* задан неправильно,
0x0800 – ошибка открытия драйвера,
0x0900…0x09FF – ошибка чтения записи контроллера.
Функция меняет признак *modify* в драйвере.

**Пример.**

Пример программы инициализации сигнального процессора.

```
#include <windows.h>
#include <stdio.h>
#include <conio.h>
#include "zadc_int.h"          // интерфейс библиотеки Zadc.dll

int main(void)
{
        long err;
```

© ООО "ЭТМС" 1992-2024. Все права защищены

```
        long typeDevice = KDU1616;        // здесь используется ZET210 USB
        long numberDSP = 0;               // первое найденное устройство

        err = ZOpen(typeDevice, numberDSP);
        if (err == 0)
                printf("Devices found, handle open.\n\r");
        else
        {
                printf("ERROR opening device: (%0x) \n\r", err);
                return 0;
        }
        err = ZResetDSP(typeDevice, numberDSP);
        if (err == 0)
                printf("DSP was reset\n\r");
        else
        {
                printf("ERROR device: (%0x) returned from ZResetDSP\n\r", err);
                err = ZClose(typeDevice, numberDSP);
                return 0;
        }
        err = ZInitDSP(typeDevice, numberDSP , "");
        if (err == 0)
                printf("DSP programs was initialized\n");
        else
                printf("ERROR device: (%0x) returned from ZInitDSP\n", err);

        err = ZClose(typeDevice, numberDSP);
        printf("Press any key for exit...\n");
        getch();

        return 0;
}
```

## Получение серийного номера DSP

***ZGetSerialNumberDSP -*** получает серийный номер DSP.

*long* ***ZGetSerialNumberDSP*** *(long typeDevice, long numberDSP, long *serialNumber)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*serialNumber* – серийный номер устройства.

Возвращаемое значение:

© ООО "ЭТМС" 1992-2024. Все права защищены

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0200 – *filename* задан неправильно,
0x0800 – ошибка открытия драйвера,
0x0900…0x09FF – ошибка чтения записи контроллера.

## Получение номера ревизии микропрограммы DSP

***ZGetRevisionDSP -*** получает номера ревизии микропрограммы DSP.

*long* ***ZGetRevisionDSP*** *(long typeDevice, long numberDSP, long *revisionNumber)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*revisionNumber* – номер ревизии.

Возвращаемое значение:
0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0200 – *filename* задан неправильно,
0x0800 – ошибка открытия драйвера,
0x0900…0x09FF – ошибка чтения записи контроллера.

## Получение имени устройства

***ZGetNameDevice -*** получает имя устройства

*long* ***ZGetNameDevice*** *(long typeDevice, long numberDSP, char *strName, long maxSizeStr)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*strName* – имя устройства, получаемая строка имени должна быть не более 16 символов вместе с завершающим 0.
*maxSizeStr* - размер буфера strName[]

Возвращаемое значение:
0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,

© ООО "ЭТМС" 1992-2024. Все права защищены

0x0200 – *filename* задан неправильно,
0x0800 – ошибка открытия драйвера,
0x0900…0x09FF – ошибка чтения записи контроллера.

## 1.4.Сервисные функции

### Опрос версии программ и драйвера

***ZGetVersion -*** опрос версии прошивки сигнального процессора, драйвера и библиотеки доступа к драйверу.

*long* ***ZGetVersion*** *(long typeDevice, long numberDSP, char* verDSP, char* verDRV, char* verLIB)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*verDSP* – строковый массив, резервируемый в памяти пользовательской программы длиной не менее 100 байт. После обращения к подпрограмме возвращает строку с указанием версии прошивки сигнального процессора. Если прошивка не загружена в сигнальный процессор, то возвращается строка нулевой длины.
*verDRV*– строковый массив, резервируемый в памяти пользовательской программы длиной не менее 100 байт. После обращения к подпрограмме возвращает строку с указанием версии драйвера.
*verLIB*– строковый массив, резервируемый в памяти пользовательской программы длиной не менее 100 байт. После обращения к подпрограмме возвращает строку с указанием версии динамической библиотеки.

Возвращаемое значение:
0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,

Функция не меняет признак *modify* в драйвере.

**Пример 1.**

В данном примере считывается версия прошивки DSP, драйвера и библиотеки.

```
long PrintVersion(void)
{
	char dsp[100],drv[100],lib[100];
	long type = KD1610;		// здесь тип устройства ADC 16/200
```

© ООО "ЭТМС" 1992-2024. Все права защищены

```
        long number = 0;                        // первый DSP

        err = ZOpen(type, numberDSP);
        if (err == 0)
                printf("Devices found, handle open.\n\r");
        else
        {
                printf("ERROR opening device: (%0x) \n\r", err);
                return err;
        }

        err = ZGetVersion(type, number, dsp, drv, lib);
        printf("%s\n", dsp);
        printf("%s\n", drv);
        printf("%s\n", lib);

        err = ZClose(type, number);
        printf("Press any key for exit...\n");
        getch();
        return 0;
}
```

**Пример 2.**

В данном примере на Visual Basic считывается версия прошивки DSP.

```
Public Sub GetVerDSP()
   Dim typeDevice As Long              ' код ошибки
   Dim numberDSP As Long               ' длина строки
   Dim Err As Long                     ' код ошибки
   Dim Length As Long          ' длина строки
   Dim strVer As String
   Dim strVerDrv As String, strVerDSP As String, strVerLib As String

   typeDevice = KDU1616                ' здесь выбрано устройство ZET210 USB
   numberDSP = 0                       ' первое устройство

   Err = ZOpen(typeDevice, numberDSP)
   If Err <> 0 Then MsgBox ("Device not find, Error = " + CStr(Err))

   strVerDrv = Space$(100)
   strVerDSP = Space$(100)
   strVerLib = Space$(100)
   Err = ZGetVersion(typeDevice, numberDSP, strVerDrv, strVerDSP, strVerLib)
   strVer = Trim(strVerDSP)            ' отрезаем пробелы
   Length = Len(strVer)
   If strLength > 0 Then strVer = Left(strVer, strLength - 1)     ' отрезаем нулевой символ
```

© ООО "ЭТМС" 1992-2024. Все права защищены

```
    MsgBox (strVer)
    Err = ZClose(typeDevice, numberDSP)
End Sub
```

## Получение типа интерфейса

***ZGetTypeConnection -*** получает тип интерфейса.

*long* ***ZGetTypeConnection*** *(long typeDevice, long numberDSP, long *type)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*type* – определяется тип интерфейса:
1 - PCI,
2 - USB,
3 - Ethernet.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,

## Опрос аппаратных возможностей устройства

***ZGetDeviceCapabilities*** - опрашивает аппаратные возможности устройства

*long* ***ZGetDeviceCapabilities*** *(long typeDevice, long numberDSP, long *capabilities)*

Параметры:.

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*capabilities* – битовая маска аппаратных возможностей устройства.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,

© ООО "ЭТМС" 1992-2024. Все права защищены

## Прочитать всю информацию о модуле АЦП, ЦАП из драйвера

***ZGetData -*** получает всю информацию об АЦП из драйвера.
***ZGetDataExt -*** получает всю информацию об ЦАП из драйвера.

*long* ***ZGetData*** *(long typeDevice, long numberDSP, ADC_INFO *Data*)
*long* ***ZGetDataExt*** *(long typeDevice, long numberDSP, ADC_INFO_EXT *Data)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*Data* – структура с информацией об устройстве, хранящаяся в драйвере

Возвращаемое значение:
0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,

## Прочитать всю информацию о модуле АЦП, ЦАП из DSP

***ZGetInfo -*** получает всю информацию об АЦП, ЦАП из DSP.

*long* ***ZGetInfo*** *(long typeDevice, long numberDSP, ADC_INFO *Data)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*Data* – структура с информацией об устройстве, полученной напрямую от устройства

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,

© ООО "ЭТМС" 1992-2024. Все права защищены

## Передать всю информацию в DSP

***ZPutInfo*** - передает всю информацию в DSP (кроме размера прерывания и старта)

*long* ***ZPutInfo*** *(long typeDevice, long numberDSP, ADC_INFO *Data)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*Data* – структура с информацией об устройстве, передаваемая напрямую устройству
Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,

## Запустить тест шлейфа для плат PnP

***ZTestCode -*** передает всю информацию в DSP (кроме размера прерывания и старта)

*long* ***ZTestCode*** *(long typeDevice, long numberDSP, long *code0, long *code1, long *code2)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*code0, code1, code2* – тестовые коды,

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,

© ООО "ЭТМС" 1992-2024. Все права защищены

## Чтение кода ошибки

***ZGetError -*** возвращает код последней внутренней ошибки драйвера (обработчика прерываний)

*long* ***ZGetError*** *(long typeDevice, long numberDSP, long *error)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*error* – код ошибки.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,

Функция не меняет признак *modify* в драйвере.

## Опрос изменения режима работы сигнального процессора

***ZGetModify -*** возвращает параметр изменения режима работы сигнального процессора. Драйвер позволяет одновременно работать нескольким программам. В этом случае каждая программа должна следить за параметром *modify* и в случае его изменения опрашивать необходимые параметры и соответственно менять свои параметры работы.

*long* ***ZGetModify*** *(long typeDevice, long numberDSP, long *modify)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*modify* – признак изменения работы драйвера. При каждом существенном изменении параметров работы драйвера происходит инкремент признака *modify*.

Программа пользователя должна считывать признак и запоминать текущее значение признака и при изменении этого значения производить необходимые действия.

Возвращаемое значение:
0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,

© ООО "ЭТМС" 1992-2024. Все права защищены

0x0800 – ошибка открытия драйвера,

Функция не меняет признак *modify* в драйвере.

## 1.5.Установка режима работы сигнального процессора

В устройствах ADC 16/200, ADC 16/500, ADC 16/500P (конфигурация KADSP/PCI - АЦП16/200 - ЦАП16/2000; KADSP/PCI - АЦП16/1000 - ЦАП16/2000; KADSP/PDP - АЦП16/1000 - ЦАП16/2000) отсутствует режим автоопределения подключенных устройств. Т.е изначально в системе неизвестно к какому сигнальному процессору подключен модуль АЦП, а к какому модуль ЦАП. Необходимо, в соответствии со схемой подключения устройств, программно установить режимы работы сигнальных процессоров. По умолчанию, после загрузки программы в сигнальный процессор, установлен режим работы с модулем АЦП.

### Установка работы с модулем АЦП

***ZSetTypeADC -*** устанавливает сигнальный процессор в режим работы с модулем АЦП.

*long* ***ZSetTypeADC*** *(long typeDevice, long numberDSP)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136).

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция меняет признак *modify* в драйвере.

### Установка работы с модулем ЦАП

***ZSetTypeDAC*** - устанавливает сигнальный процессор в режим работы с модулем ЦАП.

© ООО "ЭТМС" 1992-2024. Все права защищены

*long* ***ZSetTypeDAC*** *(long typeDevice, long numberDSP)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136).

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция меняет признак *modify* в драйвере.

## 1.6.Опрос основных характеристик модулей АЦП и ЦАП

Все драйверы для модулей АЦП и ЦАП построены по единому принципу. Программа пользователя не должна знать параметры модулей и их количество. Все параметры должны определяться по функциям опроса характеристик.

### Опрос возможности работы сигнального процессора с модулем АЦП

***ZGetEnableADC*** - опрос возможности работы сигнального процессора с модулем АЦП. В одних устройствах сигнальный процессор может поддерживать режим работы только с АЦП или только с ЦАП, в других - может работать одновременно с АЦП и ЦАП.

*long* ***ZGetEnableADC*** *(long typeDevice, long numberDSP, long *enable)*

*Параметры:*

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*enable* – возможность работы с модулем АЦП. 0 – не поддерживается , 1 – поддерживается.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,

© ООО "ЭТМС" 1992-2024. Все права защищены

0x0800 – ошибка открытия драйвера,

Функция не меняет признак *modify* в драйвере.

### Опрос возможности работы сигнального процессора с модулем ЦАП

***ZGetEnableDAC -*** опрос возможности работы сигнального процессора с модулем ЦАП.

*long* ***ZGetEnableDAC*** *(long typeDevice, long numberDSP, long *enable)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*enable* – возможность работы с модулем ЦАП. 0 – не поддерживается, 1 – поддерживается.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,

Функция не меняет признак *modify* в драйвере.

### Опрос максимального количества каналов модуля АЦП/ЦАП

***ZGetQuantityChannelADC -*** опрос максимального количества каналов модуля АЦП

***ZGetQuantityChannelDAC -*** опрос максимального количества каналов модуля ЦАП

*long* ***ZGetQuantityChannelADC*** *(long typeDevice, long numberDSP, long *quantityChannel)*
*long* ***ZGetQuantityChannelDAC*** *(long typeDevice, long numberDSP, long *quantityChannel)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*quantityChannel* - максимальное количество каналов модуля АЦП/ЦАП

© ООО "ЭТМС" 1992-2024. Все права защищены

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция не меняет признак *modify* в драйвере.

## Опрос веса младшего разряда АЦП/ЦАП

***ZGetDigitalResolutionADC -*** опрос веса младшего разряда АЦП.
***ZGetDigitalResolutionDAC -*** опрос веса младшего разряда ЦАП.

*long* ***ZGetDigitalResolutionADC*** *(long typeDevice, long numberDSP, double *digitalResolution)*
*long* ***ZGetDigitalResolutionDAC*** *(long typeDevice, long numberDSP, double *digitalResolution)*

Вес определяется как напряжение, измеренное в Вольтах для одного младшего разряда АЦП/ЦАП. Эта функция необходима, для преобразования двоично-дополнительного кода поступающего с АЦП (в ЦАП) в напряжение в Вольтах установленное на входе АЦП (на выходе ЦАП). Оцифрованные двоичные данные хранятся в буфере памяти (рисунок 1). Для того чтобы преобразовать двоичные данные в напряжения на входе модуля АЦП, необходимо умножить двоичное число на вес младшего разряда АЦП и разделить на коэффициент усиления выбранного канала. Для того чтобы преобразовать двоичные данные в напряжения на выходе модуля ЦАП, необходимо умножить двоичное число на вес младшего разряда ЦАП и умножить на коэффициент затухания (аттенюатора).

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*digitalResolution* – вес младшего разряда в Вольтах.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция не меняет признак *modify* в драйвере.

© ООО "ЭТМС" 1992-2024. Все права защищены

## Опрос количества двоичных разрядов АЦП/ЦАП

***ZGetBitsADC*** - опрос разрядности АЦП.
***ZGetBitsDAC***- опрос разрядности ЦАП.

*long **ZGetBitsADC** (long typeDevice, long numberDSP, long *numberBits)*
*long **ZGetBitsDAC** (long typeDevice, long numberDSP, long *numberBits)*

Эта информация полезна для оценки максимального входного напряжения. Чтобы получить максимальный размах напряжения, необходимо 2 возвести в степень количества двоичных разрядов АЦП/ЦАП и умножить на вес младшего разряда АЦП/ЦАП. Максимальная амплитуда напряжение определяется как половина размаха. Например, для модуля АЦП 16/200 максимальная амплитуда входного сигнала равна 10.14 Вольт при единичном коэффициенте усиления.

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*numberBits* – количество двоичных разрядов АЦП/ЦАП.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция не меняет признак *modify* в драйвере.

## Опрос размера каждого отсчета АЦП/ЦАП в 16-разрядных словах

***ZGetWordsADC*** **-** опрос размера каждого отсчета АЦП в 16-разрядных словах.

***ZGetWordsDAC*** **-** опрос размера каждого отсчета ЦАП в 16-разрядных словах.

*long **ZGetWordsADC** (long typeDevice, long numberDSP, long *numberWords)*
*long **ZGetWordsDAC** (long typeDevice, long numberDSP, long *numberWords)*

Этот размер важен для того, чтобы задать тип данных при работе с оцифрованными значениями. Если количество 16-разрядных слов равно 1, то тип данных должен быть:

short в C++,

© ООО "ЭТМС" 1992-2024. Все права защищены

integer в Visual Basic.
smallint в Delphi
Если количество 16-разрядных слов равно 2, то тип данных должен быть:
long в C++,
long в Visual Basic.
integer в Delphi

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*numberWords* – количество 16-разрядных слов, занимаемое каждым отсчетом.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция не меняет признак *modify* в драйвере.

## 1.7.Установка частоты дискретизации и режима синхронизации АЦП/ЦАП

Частота дискретизации сигнала $F_d$ зависит от частоты опорного генератора $F_o$ и коэффициента деления частоты опорного генератора $K_d$. Коэффициент - целое положительное число.

$$F_d = \frac{F_o}{K_d}$$

Некоторые типы устройств позволяют подключение внешнего опорного генератора. Для удобства программирования существует два метода установки частоты дискретизации.
В драйвере существует список возможных частот дискретизации. При помощи функции ***ZGetListFreqADC()*** (***ZGetListFreqDAC()***) можно получить список возможных частот дискретизации. А при помощи функции ***ZSetNextFreqADC()*** (***ZSetNextFreqDAC()***) можно поменять текущее значение частоты дискретизации в большую или меньшую сторону на один шаг.
Задается требуемая частота дискретизации, драйвер через функцию ***ZSetFreqADC()*** (***ZSetFreqDAC()***) устанавливает ближайшую возможную частоту дискретизации и возвращает точное значение установленной частоты дискретизации.

### Получение списка возможных частот дискретизации АЦП/ЦАП

***ZGetListFreqADC*** - получение списка возможных частот дискретизации АЦП. При многократном обращении к этой функции, она возвращает возможные частоты дискретизации.

***ZGetListFreqDAC*** - получение списка возможных частот дискретизации ЦАП. При многократном обращении к этой функции, она возвращает возможные частоты дискретизации.

*long* ***ZGetListFreqADC*** *(long typeDevice, long numberDSP, long next, double *freq)*
*long* ***ZGetListFreqDAC*** *(long typeDevice, long numberDSP, long next, double *freq)*

© ООО "ЭТМС" 1992-2024. Все права защищены

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*next* – параметр для получения списка. При первом обращении этот параметр должен быть равен 0, при последующих обращениях этот параметр не должен быть равен 0. После чтения последнего элемента функция возвращает код ошибки 0x0200 и значение частоты равное 0.
*freq* – следующее возвращаемое значение частоты дискретизации в Гц.

Возвращаемое значение (код ошибки):

0 – нормальное выполнение функции,
0x0002 – функция и режим работы DSP несовместимы (ADC – DAC),
0x0100 – *numberDSP* задан неправильно,
0x0200 – next задан неправильно (конец списка)
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция не меняет признак *modify* в драйвере.

**Пример.**

В данном примере подпрограмма выдает на экран список возможных частот

```
#include <windows.h>
#include <stdio.h>
#include <conio.h>
#include "zadc_int.h"          // интерфейс библиотеки Zadc.dll

int main(void)
{
        long err;
        long typeDevice;
        long numberDSP;
        long next;
        double freq;

        typeDevice = KD1610;        // тип устройства - ADC 16/200
        numberDSP = 0;              // первый сигнальный процессор

        err = ZOpen(typeDevice, numberDSP);    //открывается доступ к драйверу
        if(err != 0) return err;          //возврат из программы в случае неудачи
        next = 0;
        while(1)                                //бесконечный цикл
         {
          err=ZGetListFreqADC(typeDevice, numberDSP, next, &freq);          //получить
```

© ООО "ЭТМС" 1992-2024. Все права защищены

```
                                                                    //значение частоты
        if(err != 0) break;                    //если ошибка, то выход из цикла
        printf("%g Hz\n",freq);                //распечатать значение частоты
        next=1;                                //при следующем обращении, будет
                                               //считываться следующее значение частоты
    }
    err=ZClose(typeDevice, numberDSP);     //закрыть доступ к драйверу
    printf("Press any key for exit...\n");
    getch();
    return 0;
}
```

## Установка большей или меньшей частоты дискретизации АЦП/ЦАП

***ZSetNextFreqADC*** **-** устанавливает большую или меньшую частоту дискретизации по сравнению с текущей частотой дискретизации АЦП.

***ZSetNextFreqDAC*** **-** устанавливает большую или меньшую частоту дискретизации по сравнению с текущей частотой дискретизации ЦАП.

*long* ***ZSetNextFreqADC****(long typeDevice, long numberDSP, long next, double *freq)*
*long* ***ZSetNextFreqDAC****(long typeDevice, long numberDSP, long next, double *freq)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*next* – параметр для изменения частоты дискретизации.

- -1 уменьшает частоту дискретизации
- +1 увеличивает частоту дискретизации
- 0 не меняет частоту дискретизации

*freq* – возвращаемое значение установленной частоты дискретизации в Гц.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0002 – функция и режим работы DSP несовместимы (ADC – DAC),
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция меняет признак *modify* в драйвере.

© ООО "ЭТМС" 1992-2024. Все права защищены

## Опрос текущей частоты дискретизации АЦП/ЦАП

***ZGetFreqADC -*** опрос текущего значения частоты дискретизации АЦП.
***ZGetFreqDAC -*** опрос текущего значения частоты дискретизации ЦАП.

*long* ***ZGetFreqADC****(long typeDevice, long numberDSP, double *freq)*
*long* ***ZGetFreqDAC****(long typeDevice, long numberDSP, double *freq)*

Параметры:

*typeDevice* – тип устройства.

*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*freq* – значение установленной частоты дискретизации в Гц,

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0002 – функция и режим работы DSP несовместимы (ADC – DAC),
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция не меняет признак *modify* в драйвере.

## Установка частоты дискретизации АЦП/ЦАП

***ZSetFreqADC*** **-** устанавливает частоту дискретизации, наиболее близкую к требуемой, и возвращает значение установленной частоты дискретизации АЦП. Эти частоты могут различаться.

***ZSetFreqDAC*** **-** устанавливает частоту дискретизации, наиболее близкую к требуемой, и возвращает значение установленной частоты дискретизации ЦАП. Эти частоты могут различаться.

*long* ***ZSetFreqADC****(long typeDevice, long numberDSP, double freqIn, double *freqOut)*
*long* ***ZSetFreqDAC****(long typeDevice, long numberDSP, double freqIn, double *freqOut)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*freqIn* – требуемая частота дискретизации в Гц,
*freqOut* – установленная частота дискретизации в Гц,

© ООО "ЭТМС" 1992-2024. Все права защищены

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0002 – функция и режим работы DSP несовместимы (ADC – DAC),
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция меняет признак *modify* в драйвере.

### Опрос текущей опорной частоты АЦП/ЦАП

***ZGetExtFreqADC -*** опрос значения текущей опорной частоты АЦП.
***ZGetExtFreqDAC -*** опрос значения текущей опорной частоты ЦАП.

*long* ***ZGetExtFreqADC****(long typeDevice, long numberDSP, double *oporFreq)*
*long* ***ZGetExtFreqDAC****(long typeDevice, long numberDSP, double *oporFreq)*

Параметры:

*typeDevice* – тип устройства.

*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*oporFreq* – значение опорной частоты в Гц.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0002 – функция и режим работы DSP несовместимы (ADC – DAC),
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция не меняет признак *modify* в драйвере.

### Установка значения внешней опорной частоты АЦП/ЦАП

***ZSetExtFreqADC -*** установление значения внешней опорной частоты АЦП.

***ZSetExtFreqDAC-*** установление значения внешней опорной частоты ЦАП.

*long* ***ZSetExtFreqADC****(long typeDevice, long numberDSP, double extFreq)*
*long* ***ZSetExtFreqDAC****(long typeDevice, long numberDSP, double extFreq)*

© ООО "ЭТМС" 1992-2024. Все права защищены

Параметры:

*typeDevice* – тип устройства.

*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*extFreq* – значение опорной частоты в Гц.

Возвращаемое значение (код ошибки):

0 – нормальное выполнение функции,
0x0002 – функция и режим работы DSP несовместимы (ADC – DAC),
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция меняет признак *modify* в драйвере.

## Опрос режима работы от внешней опорной частоты

***ZGetEnableExtFreq*** **-** опрос режима работы от внешней опорной частоты.

*long* ***ZGetEnableExtFreq****(long typeDevice, long numberDSP, long *enable)*

Параметры:

*typeDevice* – тип устройства.

*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*enable* – 0 – режим внутренней опорной частоты, 1 – режим внешней опорной частоты для формирования частоты дискретизации.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция не меняет признак *modify* в драйвере.

## Установка режима работы от внешней опорной частоты АЦП

***ZSetEnableExtFreq*** **-** установление или сбрасывание режима работы от внешней опорной частоты. Влияет и на АЦП и на ЦАП.

© ООО "ЭТМС" 1992-2024. Все права защищены

*long* ***ZSetEnableExtFreq****(long typeDevice, long numberDSP, long enable)*

Параметры:

*typeDevice* – тип устройства.

*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*enable* – 0 – режим внутренней опорной частоты, 1 – режим внешней опорной частоты для формирования частоты дискретизации.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция меняет признак *modify* в драйвере.

## Опрос разрешения внешнего запуска накопления данных

***ZGetEnableExtStart -*** опрос разрешение/запрещение начала накопления данных от внешнего сигнала запуска.

*long* ***ZGetEnableExtStart****(long typeDevice, long numberDSP, long *enable)*

Параметры:

*typeDevice* – тип устройства.

*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*enable* – параметр, разрешающий/запрещающий использование внешнего сигнала запуска.
0 – запрещение внешнего запуска. Накопление данных АЦП начинается по команде ***ZStartADC****()*.
1 – разрешение внешнего запуска. Накопление данных начинается после подачи команды ***ZStartADC****()* и последующем появлении внешнего сигнала “запуск”.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция не меняет признак *modify* в драйвере.

© ООО "ЭТМС" 1992-2024. Все права защищены

## Установка внешнего запуска накопления данных

***ZSetEnableExtStart -*** разрешает/запрещает начало накопления данных от внешнего сигнала запуска. Влияет и на АЦП и на ЦАП. Например, для устройств ADC 16/200, APC 216 на кронштейне контроллера ввода/вывода расположены четыре разъема Lemo. Один из этих разъемов является входом сигнала внешнего запуска, другой разъем является выходом сигнала “запуск” (этот сигнал переходит в другое состояние в начале процесса накопления). Эти сигналы позволяют подключать контроллеры последовательно друг с другом для реализации синхронной схемы накопления данных. Более подробно разъемы описаны в руководстве по эксплуатации контроллера KADSP/PCI. Функция не начинает процесс накопления данных – она устанавливает режим запуска накопления данных

*long* ***ZSetEnableExtStart****(long typeDevice, long numberDSP, long enable)*

Параметры:

*typeDevice* – тип устройства.

*numberDSP* – номер сигнального процессора (значение от 0 до 136),

*enable* – параметр, разрешающий/запрещающий использование внешнего сигнала запуска.

0 – разрешение программного запуска. Накопление данных АЦП начинается по команде ***ZStartADC()***

1 – разрешение внешнего сигнала запуска. Накопление данных начинается после подачи команды ***ZStartADC()*** и последующем появлении внешнего сигнала “запуск”.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция не меняет признак *modify* в драйвере.

## Определить установленный коэффициент деления АЦП/ЦАП

***ZGetRateADC -*** определяет установленный коэффициент деления АЦП (служебн., пользоваться не рекомендуется).

***ZGetRateDAC -*** определяет установленный коэффициент деления АЦП (служебн., пользоваться не рекомендуется).

*long* ***ZGetRateADC*** *(long typeDevice, long numberDSP, long *rate)*

© ООО "ЭТМС" 1992-2024. Все права защищены

*long* ***ZGetRateDAC*** *(long typeDevice, long numberDSP, long *rate)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*rate* – коэффициент деления опорной частоты 96 МГц. Внутри модуля установлен кварцевый генератор с тактовой частотой 96 МГц. Параметр *rate* может принимать значение от 1 до 96000000.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция не меняет признак *modify* в драйвере.

## Установить частоту преобразования АЦП/ЦАП

***ZSetRateADC -*** устанавливает частоту преобразования АЦП (служебн., пользоваться не рекомендуется).
***ZSetRateDAC -*** устанавливает частоту преобразования ЦАП (служебн., пользоваться не рекомендуется).

*long* ***ZSetRateADC*** *(long typeDevice, long numberDSP, long rate)*
*long* ***ZSetRateDAC*** *(long typeDevice, long numberDSP, long rate)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*rate* – коэффициент деления опорной частоты 96 МГц. Внутри модуля установлен кварцевый генератор с тактовой частотой 96 МГц. Параметр *rate* может принимать значение от 1 до 96000000.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

© ООО "ЭТМС" 1992-2024. Все права защищены

## Прочитать статус синхронизации по внешней частоте АЦП/ЦАП

***ZGetEnableExtFreqADC*** **-** прочитывает статус синхронизации по внешней частоте АЦП.

***ZGetEnableExtFreqDAC*** **-** прочитывает статус синхронизации по внешней частоте ЦАП.

*long* ***ZGetEnableExtFreqADC*** *(long typeDevice, long numberDSP, long *enable)*
*long* ***ZGetEnableExtFreqDAC*** *(long typeDevice, long numberDSP, long *enable)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*enable* – параметр, разрешающий/запрещающий использование внешнего сигнала запуска.
*0* – запрещение внешнего запуска. Накопление данных АЦП начинается по команде ***ZStartADC()***.
*1* – разрешение внешнего запуска. Накопление данных начинается после подачи команды **ZStartADC**() и последующем появлении внешнего сигнала “запуск”.

Возвращаемое значение (код ошибки):

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

## Включение/Отключение синхронизации по внешней частоте АЦП/ЦАП

***ZSetEnableExtFreqADC*** **-** включает или отключает синхронизации по внешней частоте АЦП.

***ZSetEnableExtFreqDAC*** **-** включает или отключает синхронизации по внешней частоте ЦАП.

*long* ***ZSetEnableExtFreqADC*** *(long typeDevice, long numberDSP, long enable)*
*long* ***ZSetEnableExtFreqDAC*** *(long typeDevice, long numberDSP, long enable)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),

© ООО "ЭТМС" 1992-2024. Все права защищены

*enable* – параметр, разрешающий/запрещающий использование внешнего сигнала запуска.

*0* – запрещение внешнего запуска. Накопление данных АЦП начинается по команде **ZStartADC**().

*1* – разрешение внешнего запуска. Накопление данных начинается после подачи команды **ZStartADC**() и последующем появлении внешнего сигнала “запуск”.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

## Прочитать статус внешнего запуска АЦП/ЦАП

***ZGetEnableExtStartADC -*** читает статус внешнего запуска АЦП.
***ZGetEnableExtStartADC-*** читает статус внешнего запуска ЦАП.

*long **ZGetEnableExtStartADC** (long typeDevice, long numberDSP, long *enable)*
*long **ZGetEnableExtStartDAC** (long typeDevice, long numberDSP, long *enable)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*enable* – параметр, разрешающий/запрещающий использование внешнего сигнала запуска.

*0* – запрещение внешнего запуска. Накопление данных АЦП начинается по команде **ZStartADC**().

*1* – разрешение внешнего запуска. Накопление данных начинается после подачи команды **ZStartADC**() и последующем появлении внешнего сигнала “запуск”.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

© ООО "ЭТМС" 1992-2024. Все права защищены

### Включение/Отключение внешнего запуска АЦП/ЦАП

***ZSetEnableExtStartADC*** **-** включает или отключает внешний запуск АЦП.

***ZSetEnableExtStartDAC*** **-** включает или отключает внешний запуск ЦАП.

*long* ***ZSetEnableExtStartADC*** *(long typeDevice, long numberDSP, long enable)*
*long* ***ZSetEnableExtStartDAC*** *(long typeDevice, long numberDSP, long enable)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*enable* – параметр, разрешающий/запрещающий использование внешнего сигнала запуска.
*0* – запрещение внешнего запуска. Накопление данных АЦП начинается по команде **ZStartADC**().
*1* – разрешение внешнего запуска. Накопление данных начинается после подачи команды **ZStartADC**() и последующем появлении внешнего сигнала “запуск”.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

## 1.8.Управление каналами ввода (вывода) АЦП/ЦАП

### Опрос количества включенных каналов АЦП/ЦАП

***ZGetNumberInputADC*** **-** опрос количества включенных каналов для ввода данных АЦП.

***ZGetNumberOutputDAC*** **-** опрос количества включенных каналов для ввода данных ЦАП.

*long* ***ZGetNumberInputADC*** *(long typeDevice, long numberDSP, long *workChannel)*
*long* ***ZGetNumberOutputDAC*** *(long typeDevice, long numberDSP, long *workChannel)*

© ООО "ЭТМС" 1992-2024. Все права защищены

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*workChannel* – количество включенных каналов.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0002 – функция и режим работы DSP несовместимы (ADC – DAC),
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция не меняет признак *modify* в драйвере.

## Опрос разрешения канала для ввода (вывода) АЦП/ЦАП

***ZGetInputADC -*** опрос разрешения канала для ввода АЦП.
***ZGetInputDAC -*** опрос разрешения канала для ввода ЦАП.

*long **ZGetInputADC** (long typeDevice, long numberDSP, long numberChannel, long *enable)*
*long **ZGetInputDAC** (long typeDevice, long numberDSP, long numberChannel, long *enable)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*numberChannel* – номер канала, нумерация начинается с 0 (например, 0,1,2,3 и т.д.)
*enable* – параметр разрешения канала для ввода (вывода)
0 – заданный канал не выбран,
1 – заданный канал выбран,

Возвращаемое значение (код ошибки):

0 – нормальное выполнение функции,
0x0002 – функция и режим работы DSP несовместимы (ADC – DAC),
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция не меняет признак *modify* в драйвере.

© ООО "ЭТМС" 1992-2024. Все права защищены

## Включение канала для ввода (вывода) АЦП/ЦАП

***ZSetInputADC -*** выбор или отмена канала для ввода АЦП.
***ZSetOutputDAC -*** выбор или отмена канала для ввода ЦАП.

*long **ZSetInputADC** (long typeDevice, long numberDSP, long numberChannel, long enable)*
*long **ZSetOutputDAC** (long typeDevice, long numberDSP, long numberChannel, long enable)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*numberChannel* – номер канала, нумерация начинается с 0 (например, 0,1,2,3 и т.д.)
*enable* – параметр для включения-выключения канала для ввода (вывода)
0 – выключение заданного канала,
1 – включение заданного канала,

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0002 – функция и режим работы DSP несовместимы (ADC – DAC),
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция меняет признак *modify* в драйвере.

## Включение/выключение каналов с помощью битовой маски АЦП/ЦАП

***ZSetChannelADC*** **-** включения/выключения каналов с помощью битовой маски АЦП, (служебн., пользоваться не рекомендуется).
***ZSetChannelDAC*** **-** включения/выключения каналов с помощью битовой маски ЦАП, (служебн., пользоваться не рекомендуется).

*long **ZSetChannelADC** (long typeDevice, long numberDSP, unsigned long channelMask)*
*long **ZSetChannelDAC** (long typeDevice, long numberDSP, unsigned long channelMask)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),

© ООО "ЭТМС" 1992-2024. Все права защищены

*channelMask* – битовая маска каналов АЦП или ЦАП (бит 1 означает, что соответствующий канал будет включен),

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0002 – функция и режим работы DSP несовместимы (ADC – DAC),
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция меняет признак *modify* в драйвере.

## Опрос типа канала ввода АЦП

***ZGetInputDiffADC*** **-** опрос типа канала для ввода (синфазный или дифференциальный).

*long* ***ZGetInputDiffADC*** *(long typeDevice, long numberDSP, long numberChannel, long *enable)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*numberChannel* – номер канала, нумерация начинается с 0 (например, 0,1,2,3 и т.д.)
*enable* – параметр разрешения дифференциального режима канала для ввода
0 – синфазный тип заданного канала,
1 – дифференциальный тип заданного канала.

Возвращаемое значение (код ошибки):

0 – нормальное выполнение функции,
0x0002 – функция и режим работы DSP несовместимы (ADC – DAC),
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция не меняет признак *modify* в драйвере.

© ООО "ЭТМС" 1992-2024. Все права защищены

## Установка типа канала ввода АЦП

***ZSetInputDiffADC -*** установление типа канала для ввода (синфазный или дифференциальный).

*long* ***ZSetInputDiffADC*** *(long typeDevice, long numberDSP, long numberChannel, long enable)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*numberChannel* – номер канала, нумерация начинается с 0 (например, 0,1,2,3 и т.д.)
*enable* – параметр для установки дифференциального режима канала для ввода
0 – установка синфазного типа заданного канала,
1 – установка дифференциального типа заданного канала (объединяются два входа и используются как один дифференциальный),

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0002 – функция и режим работы DSP несовместимы (ADC – DAC),
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция меняет признак *modify* в драйвере.

## Чтение режима работы АЦП

***ZGetModaADC -*** чтение режима работы АЦП.

*long* ***ZGetModaADC*** *(long typeDevice, long numberDSP, long *moda)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*moda* – режим работы устройства (значения специфичны для разных типов устройства),

Возвращаемое значение:
0 – нормальное выполнение функции,

© ООО "ЭТМС" 1992-2024. Все права защищены

0x0002 – функция и режим работы DSP несовместимы (ADC – DAC),
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция не меняет признак *modify* в драйвере.

### Установить режим работы АЦП

***ZSetModaADC*** **-** установление режима работы АЦП, (служебн., пользоваться не рекомендуется)

*long* ***ZSetModaADC*** *(long typeDevice, long numberDSP, long moda)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*moda* – режим работы устройства (значения специфичны для разных типов устройства).

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0002 – функция и режим работы DSP несовместимы (ADC – DAC),
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция меняет признак *modify* в драйвере.

## 1.9.Управление коэффициентами усиления АЦП

На некоторых типах устройств на каждый из каналов аналогового ввода установлен усилитель с программируемым коэффициентом усиления. Каждый канал выбирается независимо от других каналов. Коэффициент усиления задается по каждому каналу индивидуально.

Существует два метода установки коэффициента усиления усилителя:

В драйвере существует список возможных коэффициентов усиления. При помощи функции ***ZGetListAmplifyADC()*** можно получить список возможных коэффициентов усиления. А при помощи функции ***ZSetNextAmplifyADC()*** можно поменять текущее значение коэффициента усиления в большую или меньшую сторону на один шаг.

Задается требуемый коэффициента усиления, драйвер через функцию ***ZSetAmplifyADC()*** устанавливает ближайший возможный коэффициент усиления и возвращает точное значение установленного коэффициента усиления.

© ООО "ЭТМС" 1992-2024. Все права защищены

## Получение списка возможных коэффициентов усиления АЦП

***ZGetListAmplifyADC*** **-** получение списка возможных коэффициентов усиления. При многократном обращении к этой функции, она возвращает возможные коэффициенты усиления.

*long* ***ZGetListAmplifyADC*** *(long typeDevice, long numberDSP, long next, double *amplify)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*next* – параметр для получения списка, при первом обращении этот параметр должен быть равен 0, при последующих обращениях этот параметр не должен быть равным 0. После чтения последнего элемента функция возвращает код ошибки 0x200 и значение коэффициента усиления равно 1.
*amplify* – следующее возвращаемое значение коэффициента усиления.

Возвращаемое значение (код ошибки):

0 – нормальное выполнение функции,
0x0002 – функция и режим работы DSP несовместимы (ADC – DAC),
0x0100 – *numberDSP* задан неправильно,
0x0200 – next задан неправильно (конец списка),
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция не меняет признак *modify* в драйвере.

## Установка большего или меньшего коэффициента усиления выбранного канала АЦП

***ZSetNextAmplifyADC*** **-** установление большего или меньшего коэффициента усиления выбранного канала.

*long* ***ZSetNextAmplifyADC*** *(long typeDevice, long numberDSP, long numberChannel, long next, double *amplify)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),

© ООО "ЭТМС" 1992-2024. Все права защищены

*numberChannel* – номер канала, нумерация начинается с 0 (например, 0,1,2,3 и т.д.)

*next* – параметр для изменения коэффициента усиления,

- -1 уменьшает коэффициент усиления,
- +1 увеличивает коэффициент усиления,
- 0 не меняет коэффициент усиления.

*amplify* – возвращаемое значение установленного коэффициента усиления.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0002 – функция и режим работы DSP несовместимы (ADC – DAC),
0x0011 – таймаут установки коэф. усиления,
0x0100 – *numberDSP* задан неправильно,
0x0200 – *numberChannel* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция меняет признак *modify* в драйвере.

## Опрос коэффициента усиления выбранного канала АЦП

***ZGetAmplifyADC*** - опрос коэффициента усиления выбранного канала.

*long **ZGetAmplifyADC** (long typeDevice, long numberDSP, long numberChannel, double *amplify)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*numberChannel* – номер канала, нумерация начинается с 0 (например, 0,1,2,3 и т.д.).
*amplify* – возвращаемое значение установленного коэффициента усиления.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0002 – функция и режим работы DSP несовместимы (ADC – DAC),
0x0100 – *numberDSP* задан неправильно,
0x0200 – *numberChannel* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

© ООО "ЭТМС" 1992-2024. Все права защищены

Функция не меняет признак *modify* в драйвере.

## Установка коэффициента усиления выбранного канала АЦП

***ZSetAmplifyADC*** **-** установление коэффициента усиления выбранного канала.

*long* ***ZSetAmplifyADC*** *(long typeDevice, long numberDSP, long numberChannel, double amplifyIn, double *amplifyOut)*
Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*numberChannel* – номер канала, нумерация начинается с 0 (например, 0,1,2,3 и т.д.).
*amplifyIn* – задаваемое значение коэффициента усиления,
*amplifyOut* – возвращаемое значение установленного коэффициента усиления.

Возвращаемое значение (код ошибки):

0 – нормальное выполнение функции,
0x0002 – функция и режим работы DSP несовместимы (ADC – DAC),
0x0011 – таймаут установки коэффициента усиления,
0x0100 – *numberDSP* задан неправильно,
0x0200 – *numberChannel* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция меняет признак *modify* в драйвере.

**Пример.**

В данном примере выбирается один канал и устанавливается коэффициент усиления равный четырем.

```
#include <windows.h>
#include <stdio.h>
#include <conio.h>
#include "zadc_int.h"          // интерфейс библиотеки Zadc.dll

int main(void)
{
        long typeDevice;
        long numberDSP;
        long err, numberChannel;
        double amplify;
```

© ООО "ЭТМС" 1992-2024. Все права защищены

```
        typeDevice = KD1610;        // тип устройства - ADC 16/200
        numberDSP = 0;              // первый сигнальный процессор

        err = ZOpen(typeDevice, numberDSP);
        if (err == 0)               printf("Devices found, handle open.\n");
        else
        {
        printf("ERROR opening device: (%0x) returned from CreateFile\n",
        GetLastError());
        return 0;
        }
        err = ZSetInputADC(typeDevice, numberDSP,0,1);      // выбирается 1-ый канал
        err = ZSetInputADC(typeDevice, numberDSP,1,0);      // остальные каналы
выкл.
        err = ZSetInputADC(typeDevice, numberDSP,2,0);
        err = ZSetInputADC(typeDevice, numberDSP,3,0);
        err = ZSetInputADC(typeDevice, numberDSP,4,0);
        err = ZSetInputADC(typeDevice, numberDSP,5,0);
        err = ZSetInputADC(typeDevice, numberDSP,6,0);
        err = ZSetInputADC(typeDevice, numberDSP,7,0);
                                                    // опрос количества включенных
                                                    каналов
        err = ZGetNumberInputADC(typeDevice, numberDSP,&numberChannel);
                                                // установка коэффициента усиления равным 4
        err = ZSetAmplifyADC(typeDevice, numberDSP, 0, 4., &amplify);
        err = ZClose(typeDevice, numberDSP);
        printf("Press any key for exit...\n");
        getch();
        return 0;
}
```

## 1.10.Управление коэффициентами усиления ПУ 8/10

.Если подключен модуль усилителя заряда ПУ 8/10, то можно управлять коэффициентом усиления его каналов. Коэффициент усиления задается по каждому каналу индивидуально.

Существует два метода установки коэффициента усиления предварительного усилителя:

В драйвере существует список возможных коэффициентов усиления. При помощи функции ***ZGetListPreAmplifyADC()*** можно получить список возможных коэффициентов усиления. А при помощи функции ***ZSetNextPreAmplifyADC()*** можно поменять текущее значение коэффициента усиления в большую или меньшую сторону на один шаг.

© ООО "ЭТМС" 1992-2024. Все права защищены

Задается требуемый коэффициента усиления, драйвер через функцию ***ZSetPreAmplifyADC()*** устанавливает ближайший возможный коэффициент усиления и возвращает точное значение установленного коэффициента усиления.

## Получение списка возможных коэффициентов усиления предварительного усилителя

***ZGetListPreAmplifyADC -*** получение списка возможных коэффициентов усиления предварительного усилителя. При многократном обращении к этой функции, она возвращает возможные коэффициенты усиления предварительного усилителя.

*long* ***ZGetListPreAmplifyADC*** *(long typeDevice, long numberDSP, long next, double *amplify)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*next* – параметр для получения списка. При первом обращении этот параметр должен быть равен 0, при последующих обращениях этот параметр не должен быть равен 0. После чтения последнего элемента функция возвращает код ошибки 0x200 и значение коэффициента усиления равно 1.
*amplify* – следующее возвращаемое значение коэффициента усиления предварительного усилителя.

Возвращаемое значение (код ошибки):

0 – нормальное выполнение функции,
0x0002 – функция и режим работы DSP несовместимы (ADC – DAC),
0x0100 – *numberDSP* задан неправильно,
0x0200 – next задан неправильно (конец списка),
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция не меняет признак *modify* в драйвере.

## Установка большего или меньшего коэффициента усиления предварительного усилителя

***ZSetNextPreAmplifyADC -*** установление большего или меньшего коэффициента усиления предварительного усилителя выбранного канала.

© ООО "ЭТМС" 1992-2024. Все права защищены

*long* ***ZSetNextPreAmplifyADC*** *(long typeDevice, long numberDSP, long numberChannel, long next, double *amplify)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*numberChannel* – выбор номера канала, в котором необходимо произвести изменение коэффициента усиления предварительного усилителя и должен принимать значения от 0 – для первого канала до 7 для восьмого канала,
*next* – параметр для изменения коэффициента усиления предварительного усилителя,

- -1 уменьшает коэффициент усиления,
- +1 увеличивает коэффициент усиления,
- 0 не меняет коэффициент усиления.

*amplify* – возвращаемое значение установленного коэффициента усиления предварительного усилителя.

Возвращаемое значение (код ошибки):

0 – нормальное выполнение функции,
0x0002 – функция и режим работы DSP несовместимы (ADC – DAC),
0x0100 – *numberDSP* задан неправильно,
0x0200 – *numberChannel* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция меняет признак *modify* в драйвере.

## Опрос коэффициента усиления предварительного усилителя выбранного канала

***ZGetPreAmplifyADC -*** опрос коэффициента усиления предварительного усилителя выбранного канала.

*long* ***ZGetPreAmplifyADC*** *(long typeDevice, long numberDSP, long numberChannel, double *amplify)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),

© ООО "ЭТМС" 1992-2024. Все права защищены

*numberChannel* – выбор номера канала, в котором необходимо произвести опрос коэффициента усиления предварительного усилителя и должен принимать значения от 0 – для первого канала до 7 для восьмого канала,

*amplify* – возвращаемое значение установленного коэффициента усиления предварительного усилителя.

Возвращаемое значение (код ошибки):

0 – нормальное выполнение функции,
0x0002 – функция и режим работы DSP несовместимы (ADC – DAC),
0x0100 – *numberDSP* задан неправильно,
0x0200 – *numberChannel* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция не меняет признак *modify* в драйвере.

## Установка коэффициента усиления предварительного усилителя выбранного канала

***ZSetPreAmplifyADC*** **-** установление коэффициента усиления предварительного усилителя выбранного канала.

*long* ***ZSetPreAmplifyADC*** *(long typeDevice, long numberDSP, long numberChannel, double amplifyIn, double *amplifyOut)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*numberChannel* – выбор номера канала, в котором необходимо произвести изменение коэффициента усиления предварительного усилителя и должен принимать значения от 0 – для первого канала до 7 для восьмого канала,
*amplifyIn* – задаваемое значение коэффициента усиления предварительного усилителя,
*amplifyOut* – возвращаемое значение установленного коэффициента усиления предварительного усилителя.

Возвращаемое значение (код ошибки):

0 – нормальное выполнение функции,
0x0002 – функция и режим работы DSP несовместимы (ADC – DAC),
0x0100 – *numberDSP* задан неправильно,
0x0200 – *numberChannel* задан неправильно,

© ООО "ЭТМС" 1992-2024. Все права защищены

0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция меняет признак *modify* в драйвере.

## 1.11.Управление коэффициентами ослабления аттенюатора ЦАП

### Опрос поддерживается ли программный аттенюатор

***ZFindSoftAtten*** **-** опрос коэффициента ослабления аттенюатора выбранного канала

*long* ***ZFindSoftAtten*** *(long typeDevice, long numberDSP, long *present)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*present* – подключен ли модуль НСР. Значение
1 – модуль НСР подключен и им можно управлять,
0 – модуль НСР не найден.

Возвращаемое значение (код ошибки):

0 – нормальное выполнение функции,
0x0002 – функция и режим работы DSP несовместимы (ADC – DAC),
0x0100 – *numberDSP* задан неправильно,
0x0200 – *numberChannel* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция не меняет признак *modify* в драйвере.

### Опрос коэффициента ослабления аттенюатора выбранного канала

***ZGetAttenDAC*** **-** опрос коэффициента ослабления аттенюатора выбранного канала

© ООО "ЭТМС" 1992-2024. Все права защищены

*long* ***ZGetAttenDAC*** *(long typeDevice, long numberDSP, long numberChannel, double *reduction)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*numberChannel* – номер канала, нумерация начинается с нуля.
*reduction* – точное значение установленного коэфф. ослабления (значение от 0.001 до 1).

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0002 – функция и режим работы DSP несовместимы (ADC – DAC),
0x0100 – *numberDSP* задан неправильно,
0x0200 – *numberChannel* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция не меняет признак *modify* в драйвере.

## Установка коэффициента ослабления аттенюатора выбранного канала

***ZSetAttenDAC -*** установление коэффициента ослабления аттенюатора заданного канала аналогового вывода.

*long* ***ZSetAttenDAC*** *(long typeDevice, long numberDSP, long numberChannel, double reductionIn, double *reductionOut)*
Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*numberChannel* – номер канала, нумерация начинается с нуля.
*reductionIn* – задаваемый коэффициента ослабления (значение от 0.001 до 1)
*reductionOut* – точное значение установленного коэффициента ослабления.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0002 – функция и режим работы DSP несовместимы (ADC – DAC),
0x0100 – *numberDSP* задан неправильно,
0x0200 – *numberChannel* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

© ООО "ЭТМС" 1992-2024. Все права защищены

Функция меняет признак *modify* в драйвере.

## 1.12.Управление процессом перекачки данных

Оцифрованные данные от аналого-цифрового преобразователя, вернее, от его контроллера поступают в память центрального процессора порциями с размером, равным размеру буфера перекачки. Эти порции данных перекачиваются в процедуре обработки прерываний. Буфер для перекачки, как правило, намного меньше буфера памяти центрального процессора для хранения данных. Такая структура построения обеспечивает непрерывный (без пропусков) ввод оцифрованных данных в буфер памяти центрального процессора и последующую обработку данных программой пользователя. Признаком накопления данных может служить счетчик прерываний, который можно получить при помощи функции ***ZGetFlag()***, или указатель процесса накопления в буфере, который можно получить при помощи функции ***ZGetPointerADC() (ZGetPointerDAC())***. Одномоментно оцифрованные данные, поступающие от различных каналов аналого-цифрового преобразователя, находятся в одном буфере накопления последовательно друг за другом по мере возрастания номера канала. Например, выбраны каналы для ввода 1, 3, 5, 6, 7. Тогда данные АЦП будут расположены в памяти в следующем порядке (см. табл. 1)

Таблица 1.

| индекс буфера накопления | номер канала | номер кадра |
|---|---|---|
| 0 | 1 | 0 |
| 1 | 3 | 0 |
| 2 | 5 | 0 |
| 3 | 6 | 0 |
| 4 | 7 | 0 |
| 5 | 1 | 1 |
| 6 | 3 | 1 |
| 7 | 5 | 1 |
| 8 | 6 | 1 |
| 9 | 7 | 1 |
| 10 | 1 | 2 |
| 11 | 3 | 2 |
| 12 | 5 | 2 |
| 13 | 6 | 2 |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | | |
|---|---|---|
| 14 | 7 | 2 |

## Установка режима внешней подкачки данных или внутренней генерации сигналов ЦАП

***ZSetExtCycleDAC -*** установление режима работы ЦАП.

*long* ***ZSetExtCycleDAC*** *(long typeDevice, long numberDSP, long enable)*

Цифроаналоговый преобразователь может функционировать в двух режимах:

Цифроаналоговый преобразователь может функционировать в двух режимах:

- в режиме генерации сигналов из внутренней памяти сигнального процессора. В этом случае основным достоинством является то, что не требуется каждый раз пересылать данные в сигнальный процессор и не используются ресурсы центрального процессора. Ограничением является размер буфера для циклической или однократной генерации сигнала. Этот буфер ограничен памятью сигнального процессора и его размер можно узнать с помощью функции ZGetMaxInterruptDAC() (Размер буфера равен удвоенному размеру порции данных за прерывание).

в режиме генерации сигналов из памяти центрального процессора. В этом режиме используются ресурсы центрального компьютера.

1-ый случай: используется модуль KADSP/PCI и к нему подключены АЦП и ЦАП, при этом модуль АЦП подключен к ведущему сигнальному процессору (с четным номером) и модуль ЦАП – к ведомому процессору (с нечетным номером). В этом случае для корректной работы модуля ЦАП размеры буферов прерываний и частоты дискретизации модулей АЦП и ЦАП должны быть связаны следующим соотношением:

$$\frac{\frac{Size\_in\_ADC}{Number\_of\_channel\_ADC}}{\frac{Size\_in\_DAC}{Number\_of\_channel\_DAC}} = \frac{Freq\_sample\_ADC}{Freq\_sample\_DAC}$$

где *Size_in_ADC* - размер буфера прерывания для АЦП,
*Size_in_DAC* - размер буфера прерывания для ЦАП,
*Number_of_channel_ADC* - количество включенных каналов АЦП,
*Number_of_channel_DAC* - количество включенных каналов ЦАП,
*Freq_sample_ADC* - частота дискретизации модуля АЦП,
*Freq_sample_DAC* - частота дискретизации модуля АЦП.

2-ой случай: используется модуль KADSP/PCI и к нему подключены два АЦП. В этом случае для корректной работы модулей АЦП размеры буферов прерываний и частоты дискретизации модулей должны быть связаны аналогичным соотношением.

3-ий случай: используются контроллеры USB. В этом случае размер буфера прерывания либо жестко задан, либо кратен объему полезных данных в одном пакете USB. Потоки данных АЦП и ЦАП являются независимыми и обрабатываются параллельно. Поэтому следить за соотношением кол-ва включенных каналов частот дискретизации и размеров пакетов прерываний для этих типов устройств не нужно.

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*enable* – параметр для выбора режима работы ЦАП
0 – режим генерации сигнала из внутренней памяти сигнального процессора,
не 0 – режим генерации сигнала из памяти центрального процессора.

Возвращаемое значение (код ошибки):

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

© ООО "ЭТМС" 1992-2024. Все права защищены

Функция меняет признак *modify* в драйвере.

### Опрос максимального размера буфера в ЦАП в DSP

***ZGetMaxSizeBufferDSPDAC -*** опрашивает максимальный размер буфера в ЦАП в DSP

*long* ***ZGetMaxSizeBufferDSPDAC*** *(long typeDevice, long numberDSP, long *size)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*size* – размер буфера ЦАП в DSP.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция не меняет признак *modify* в драйвере.

### Опрос размера буфера в ЦАП в DSP

***ZGetSizeBufferDSPDAC -*** опрашивает размер буфера в ЦАП в DSP

*long* ***ZGetSizeBufferDSPDAC*** *(long typeDevice, long numberDSP, long *size)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*size* – размер буфера ЦАП в DSP,

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

© ООО "ЭТМС" 1992-2024. Все права защищены

Функция не меняет признак *modify* в драйвере.

### Установить размер буфера в ЦАП в DSP

***ZSetSizeBufferDSPDAC -*** устанавливает размер буфера в ЦАП в DSP

*long* ***ZSetSizeBufferDSPDAC*** *(long typeDevice, long numberDSP, long size)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*size* – размер буфера ЦАП в DSP.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция меняет признак *modify* в драйвере.

### Опрос размера буфера для перекачки данных АЦП/ЦАП

***ZGetInterruptADC -*** опрос размера буфера для перекачки данных за одно прерывание АЦП.
***ZGetInterruptDAC -*** опрос размера буфера для перекачки данных за одно прерывание ЦАП.

*long* ***ZGetInterruptADC*** *(long typeDevice, long numberDSP, long * size)*
*long* ***ZGetInterruptDAC*** *(long typeDevice, long numberDSP, long * size)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*size* – размер буфера для перекачки данных за одно прерывание.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0002 – функция и режим работы DSP несовместимы (ADC – DAC),

© ООО "ЭТМС" 1992-2024. Все права защищены

0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция не меняет признак *modify* в драйвере.

### Опрос максимального размера буфера для перекачки данных АЦП/ЦАП

***ZGetMaxInterruptADC*** **-** опрос размера буфера для перекачки данных за одно прерывание АЦП.
***ZGetMaxInterruptDAC*** **-** опрос размера буфера для перекачки данных за одно прерывание ЦАП.

*long* ***ZGetMaxInterruptADC*** *(long typeDevice, long numberDSP, long * size)*
*long* ***ZGetMaxInterruptDAC*** *(long typeDevice, long numberDSP, long * size)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*size* – размер буфера для перекачки данных за одно прерывание.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция не меняет признак *modify* в драйвере.

### Установка размера буфера для перекачки данных АЦП/ЦАП

***ZSetInterruptADC*** **-** установление размера буфера памяти сигнального процессора для перекачки данных из памяти сигнального процессора в память компьютера во время каждого прерывания АЦП.
***ZSetInterruptDAC*** **-** установление размера буфера памяти сигнального процессора для перекачки данных из памяти сигнального процессора в память компьютера во время каждого прерывания ЦАП.

*long* ***ZSetInterruptADC*** *(long typeDevice, long numberDSP, long size)*

© ООО "ЭТМС" 1992-2024. Все права защищены

*long* ***ZSetInterruptDAC*** *(long typeDevice, long numberDSP, long size)*

Размер *size* не может превышать значения максимального размера. Чем меньше размер буфера, тем меньше время запаздывания от оцифровки данных до обработки данных центральным процессором и меньше время реакции в системах реального времени с обратной связью, но при этом возрастают требования к ресурсам компьютера для обработки прерываний.

Для удобства обработки оцифрованных сигналов размер буфера рекомендуется делать кратным количеству включенных каналов, например 256*numberChannel.

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*size* – размер буфера для перекачки данных, значение задается в словах (отсчетах АЦП/ЦАП).

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0002 – функция и режим работы DSP несовместимы (ADC – DAC),
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция меняет признак *modify* в драйвере.

***Примечание по поводу функции ZSetInterruptADC, то есть по размеру прерывания:***

Под прерыванием понимается обновление указателя PointerADC (например, время в программе ZetServer обновляется при изменении этого указателя). Размер прерывания SizeInterrupt определяет, через сколько полученных 16-битных слов (отсчетов) оно возникает.Размер прерывания определяется автоматически по формуле:

SizeInterruptADC = SizePacketADC * QuantityPacketsADC, где SizePacketADC — это количество слов, передаваемых в одном пакете USB, а QuantityPacketsADC — это количество пакетов USB, передаваемых за одно прерывание.

Чтобы изменить размер прерывания, нужно менять именно эти два параметра.Функция ZSetInterruptADC ничего не изменяет и считается устаревшей.Размер пакета меняется функцией ZSetSizePacketADC (от 1 до ZGetMaxSizePacketADC, включительно, по умолчанию 252).Количество пакетов меняется функцией ZSetQuantityPacketsADC (от 1 до ZGetMaxQuantityPacketsADC, включительно, по умолчанию 16).Таким образом, размер прерывания по умолчанию:

SizeIterrupt = 252 * 16 = 4032.

© ООО "ЭТМС" 1992-2024. Все права защищены

Теперь по поводу выбора оптимального размера. Здесь нас интересует результирующая частота прерываний. Частота возникновения прерываний зависит от размера прерывания и от суммарной частоты дискретизации по включенным каналам. Так, например, при размере прерывания по умолчанию и при включении двух каналов с суммарной частотой 10 кГц, прерывание будет возникать с частотой:

FreqInterrupt = FreqSum / SizeInterrupt = 10000 / 4032 ~ 2.5 раза в сек.

Если сделать размер прерывания очень маленьким, то прерывание будет возникать слишком часто, что отрицательно скажется на производительности операционной системы. Если же сделать его очень большими, то прерывание, наоборот, будет возникать слишком редко (до одного раза в несколько секунд), что сильно снизит время реакции.

Следует также учитывать, функции ZSetSizePacketADC и ZSetQuantityPacketsADC предоставляют низкий уровень управления устройством. Программа ZetServer (с которой работают почти все программы из состава ПО ZETLAB) может попытаться выставить эти параметры автоматически,что может привести к конфликтам в настройках при совместной работе со сторонней программой.

Таким образом, рекомендуем не менять эти параметры, оставив значения по умолчанию. Если же частота прерываний покажется слишком низкой, то можно попробовать снизить значения параметров, но с учетом возможных конфликтов.

В выражении 256*numberChannel, приведенном в руководстве, множитель 256 указан просто в качестве примера. Что касается кратности количеству включенных каналов, то это замечание относится к параметру SizePacketADC, чтобы в одном пакете USB всегда содержалось одинаковое количество отсчетов по всем каналам. Это требование не является обязательным, но поможет избежать рассихнронизации данных в случае возникновения непредвиденных ошибок, например, при случайной потере USB пакетов.

## Установка размера буфера памяти накопления АЦП/ЦАП

***ZSetBufferSizeADC -*** устанавливает размер буфера накопления в памяти процессора АЦП.

***ZSetBufferSizeDAC-*** устанавливает размер буфера накопления в памяти процессора ЦАП.

*long **ZSetBufferSizeADC** (long typeDevice, long numberDSP, long size)*
*long **ZSetBufferSizeDAC** (long typeDevice, long numberDSP, long size)*

© ООО "ЭТМС" 1992-2024. Все права защищены

Размер не может превышать размер буфера драйвера, запрашиваемый при загрузке операционной системы. Накопление в память центрального процессора может быть циклическим или однократным. Метод накопления задается функцией ***ZSetCycleSampleADC() (ZSetCycleSampleDAC())***. При циклическом накоплении, программа пользователя может отслеживать текущий указатель накопления буфера по функции ***ZGetPointerADC() (ZGetPointerDAC())***, и определять, сколько новых данных записалось в память после последнего обращения. Для удобства работы размер буфера желательно делать кратным количеству включенных каналов.

Размер буфера в циклическом режиме накопления желательно устанавливать большим, чтобы не возникало эффекта уничтожения предыдущих оцифрованных данных.

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*size* – размер буфера в памяти центрального процессора, размер задается в 16-разрядных словах.

Возвращаемое значение (код ошибки):

0 – нормальное выполнение функции,
0x0002 – функция и режим работы DSP несовместимы (ADC – DAC),
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция меняет признак *modify* в драйвере.

***Примечание по поводу функции ZSetCycleSampleADC:***

Функция однократного накопления данных ZSetCycleSampleADC не поддерживается устройством ZET210.

## Отображение буфера памяти накопления АЦП/ЦАП

***ZGetBufferADC -*** отображение буфера данных, опрос адреса и размера буфера данных АЦП.

***ZGetBufferDAC -*** отображение буфера данных, опрос адреса и размера буфера данных ЦАП.

*long **ZGetBufferADC** (long typeDevice, long numberDSP, void **buffer, long *size)*
*long **ZGetBufferDAC** (long typeDevice, long numberDSP, void **buffer, long *size)*

© ООО "ЭТМС" 1992-2024. Все права защищены

Буфер расположен в системной области памяти и резервируется драйвером. Это позволяет использовать данные из буфера одновременно нескольким программам. В конце программы необходимо дать команду освобождения буфера *ZRemBufferADC() (ZRemBufferDAC()).*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*buffer* – адрес буфера в памяти процессора.
*size* – размер буфера в памяти центрального процессора в 16-разрядных словах.

Возвращаемое значение (код ошибки):

0 – нормальное выполнение функции,
0x0002 – функция и режим работы DSP несовместимы (ADC – DAC),
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция не меняет признак *modify* в драйвере.

## Освобождение буфера памяти накопления АЦП/ЦАП

***ZRemBufferADC*** - освобождает дескрипторы отображения памяти для доступа к буферу данных драйвера АЦП.

***ZRemBufferDAC***- освобождает дескрипторы отображения памяти для доступа к буферу данных драйвера ЦАП.

*long **ZRemBufferADC** (long typeDevice, long numberDSP, void **buffer)*
*long **ZRemBufferDAC** (long typeDevice, long numberDSP, void **buffer)*

Эта команда подается в конце программы для того, чтобы освободить ресурсы операционной системы. Операционная система может поддерживать открытый доступ для ~ 2000 буферов.

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*buffer* – адрес буфера в памяти процессора.

© ООО "ЭТМС" 1992-2024. Все права защищены

Возвращаемое значение (код ошибки):

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция не меняет признак *modify* в драйвере.

## Установка циклического или одноразового накопления АЦП/ЦАП

***ZSetCycleSampleADC*** **-** установление циклического или одноразового накопления в буфер данных центрального процессора АЦП.

***ZSetCycleSampleDAC*** **-** установление циклического или одноразового накопления в буфер данных центрального процессора ЦАП.

*long* ***ZSetCycleSampleADC*** *(long typeDevice, long numberDSP, long enable)*
*long* ***ZSetCycleSampleDAC*** *(long typeDevice, long numberDSP, long enable)*

Параметры:.

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*enable* – тип накопления, 0 – однократное накопление в буфер данных, 1 – циклическое накопление в буфер данных.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0002 – функция и режим работы DSP несовместимы (ADC – DAC),
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция меняет признак *modify* в драйвере.

## Пересылка последних накопленных данных АЦП

***ZGetLastDataADC*** **-** пересылает последние полученные отсчеты по заданному каналу в буфер данных пользователя АЦП.

*long* ***ZGetLastDataADC*** *(long typeDevice, long numberDSP, long numberChannel, void *buffer, long size)*

© ООО "ЭТМС" 1992-2024. Все права защищены

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*numberChannel* – номер канала. Нумерация начинается с нуля.
*buffer* – адрес зарезервированного буфера в программе пользователя,
*size* – размер буфера в памяти центрального процессора (в 16-разрядных словах).

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0002 – функция и режим работы DSP несовместимы (ADC – DAC),
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция не меняет признак *modify* в драйвере.

## Опрос указателя накопления в буфер данных

***ZGetPointerADC*** **-** опрос указателя накопления в буфер данных центрального процессора АЦП.
***ZGetPointerDAC*** **-** опрос указателя накопления в буфер данных центрального процессора ЦАП.

*long* ***ZGetPointerADC*** *(long typeDevice, long numberDSP, long *pointer)*
*long* ***ZGetPointerDAC*** *(long typeDevice, long numberDSP, long *pointer)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*pointer* – адрес указателя следуещего отсчета АЦП/ЦАП (индекс целочисленного массива 16-разрядных слов), может принимать значения от 0 до *size*-1 (размер буфера в памяти центрального процессора минус единица).

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0002 – функция и режим работы DSP несовместимы (ADC – DAC),
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

© ООО "ЭТМС" 1992-2024. Все права защищены

Функция не меняет признак *modify* в драйвере.

### Опрос флага прерываний

***ZGetFlag -*** опрашивает флаг прерываний.

*long* ***ZGetFlag*** *(long typeDevice, long numberDSP, unsigned long *flag)*

Флаг это количество прерываний произошедших после команды ***ZStartADC()*** Функция необходима для проверки накопления данных. В программе можно организовать периодический опрос флага и при изменении значения флага обрабатывать очередную порцию данных.

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*flag* – количество прерываний с начала накопления данных.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция не меняет признак *modify* в драйвере.

### Опрос состояния накопления данных АЦП/ЦАП

***ZGetStartADC -*** проверяет состояние сигнального процессора АЦП.
***ZGetStartDAC-*** проверяет состояние сигнального процессора ЦАП.

*long* ***ZGetStartADC*** *(long typeDevice, long numberDSP, long *status)*
*long* ***ZGetStartDAC*** *(long typeDevice, long numberDSP, long *status)*

Эта функция полезна при однократном накоплении данных.

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),

© ООО "ЭТМС" 1992-2024. Все права защищены

*status* – состояние сигнального процессора. Значение
1 – процесс ввода продолжается,
0 – накопление закончено или процесс ввода не начат.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0002 – функция и режим работы DSP несовместимы (ADC – DAC),
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция не меняет признак *modify* в драйвере.

### Старт накопления данных АЦП/ЦАП

***ZStartADC -*** запускает процесс накопления данных в память буфера накопления данных АЦП.
***ZStartDAC-*** запускает процесс накопления данных в память буфера накопления данных ЦАП.

*long* ***ZStartADC*** *(long typeDevice, long numberDSP)*
*long* ***ZStartDAC*** *(long typeDevice, long numberDSP)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136).

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0002 – функция и режим работы DSP несовместимы (ADC – DAC),
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция меняет признак *modify* в драйвере.

### Останов накопления данных АЦП/ЦАП

***ZStopADC -*** останавливает процесс накопления данных АЦП.
***ZStopDAC -*** останавливает процесс накопления данных ЦАП.

© ООО "ЭТМС" 1992-2024. Все права защищены

*long* ***ZStopADC*** *(long typeDevice, long numberDSP)*
*long* ***ZStopDAC*** *(long typeDevice, long numberDSP)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136).

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция меняет признак *modify* в драйвере.

## Определить установленный размер пакета данных DSP АЦП/ЦАП

***ZGetSizePacketADC -*** определяет установленный размер пакета данных DSP АЦП.

***ZGetSizePacketDAC -*** определяет установленный размер пакета данных DSP ЦАП.

*long* ***ZGetSizePacketADC*** *(long typeDevice, long numberDSP, long *size)*
*long* ***ZGetSizePacketDAC*** *(long typeDevice, long numberDSP, long *size)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136).
size – размер пакета данных.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция не меняет признак *modify* в драйвере.

© ООО "ЭТМС" 1992-2024. Все права защищены

## Определить максимально возможный размер пакета данных DSP АЦП/ЦАП

***ZGetMaxSizePacketADC*** **-** определяет максимально возможный размер пакета данных DSP АЦП.
***ZGetMaxSizePacketDAC*** **-** определяет максимально возможный размер пакета данных DSP ЦАП.

*long* ***ZGetMaxSizePacketADC****(long typeDevice, long numberDSP, long *size)*
*long* ***ZGetMaxSizePacketDAC****(long typeDevice, long numberDSP, long *size)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136).
size – максимально возможный размер пакета данных.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция не меняет признак *modify* в драйвере.

## Установить размер пакета данных DSP АЦП/ЦАП

***ZSetSizePacketADC*** **-** устанавливает размер пакета данных DSP АЦП.
***ZSetSizePacketDAC-*** устанавливает размер пакета данных DSP АЦП.

*long* ***ZSetSizePacketADC*** *(long typeDevice, long numberDSP, long size)*
*long* ***ZSetSizePacketDAC*** *(long typeDevice, long numberDSP, long size)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136).
size – размер пакета данных.

Возвращаемое значение:

© ООО "ЭТМС" 1992-2024. Все права защищены

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция меняет признак *modify* в драйвере.

## Определить установленное количество пакетов за одно прерывание АЦП/ЦАП

***ZGetQuantityPacketsADC*** **-** определяет установленное количество пакетов за одно прерывание АЦП.
***ZGetQuantityPacketsDAC*** **-** определяет установленное количество пакетов за одно прерывание ЦАП.

*long* ***ZGetQuantityPacketsADC*** *(long typeDevice, long numberDSP, long *size)*
*long* ***ZGetQuantityPacketsDAC*** *(long typeDevice, long numberDSP, long *size)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136).
size – количество пакетов данных в одном прерывании.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция не меняет признак *modify* в драйвере.

## Определить максимальное возможное количество пакетов за одно прерывание АЦП/ЦАП

***ZGetMaxQuantityPacketsADC*** **-** определяет максимальное возможное количество пакетов за одно прерывание АЦП.
***ZGetMaxQuantityPacketsDAC*** **-** определяет максимальное возможное количество пакетов за одно прерывание ЦАП.

© ООО "ЭТМС" 1992-2024. Все права защищены

*long* ***ZGetMaxQuantityPacketsADC*** *(long typeDevice, long numberDSP, long *size)*
*long* ***ZGetMaxQuantityPacketsDAC*** *(long typeDevice, long numberDSP, long *size)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136).
size – максимально возможное количество пакетов данных в одном прерывании.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция не меняет признак *modify* в драйвере.

## Установить количество пакетов за одно прерывание АЦП/ЦАП

***ZSetQuantityPacketsADC*** **-** устанавливает количество пакетов за одно прерывание АЦП.

***ZSetQuantityPacketsDAC*** **-** устанавливает количество пакетов за одно прерывание ЦАП.

*long* ***ZSetQuantityPacketsADC*** *(long typeDevice, long numberDSP, long size)*
*long* ***ZSetQuantityPacketsDAC*** *(long typeDevice, long numberDSP, long size)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136).
size – количество пакетов данных в одном прерывании.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция меняет признак *modify* в драйвере.

© ООО "ЭТМС" 1992-2024. Все права защищены

# 1.13.Управление модулем HCP

## Опрос поддержки и подключения модуля HCP

***ZFindHCPADC -*** опрашивает поддерживается ли модуль HCP для питания датчиков стандарта ICP.

*long* ***ZFindHCPADC*** *(long typeDevice, long numberDSP, long *present)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*present* – подключен ли модуль HCP. Значение
1 – модуль HCP подключен и им можно управлять,
0 – модуль HCP не найден.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0002 – функция и режим работы DSP несовместимы (ADC – DAC),
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается (не поддерживается HCP).

Функция не меняет признак *modify* в драйвере.

## Опрос режима работы заданного канала модуля HCP

***ZGetHCPADC -*** опрашивает, включено ли питание постоянным током по заданному каналу модуля HCP.

*long* ***ZGetHCPADC*** *(long typeDevice, long numberDSP, long numberChannel long *enable)*

Параметры:

*typeDevice* – тип устройства.

© ООО "ЭТМС" 1992-2024. Все права защищены

*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*numberChannel* – номер канала, нумерация начинается с 0 (например, 0,1,2,3 и т.д.).
*enable* – режим работы модуля НСР для заданного канала. Значение
1 – питание по заданному каналу модуля НСР включено,
0 – питание по заданному каналу модуля НСР выключено.

Возвращаемое значение (код ошибки):

0 – нормальное выполнение функции,
0x0002 – функция и режим работы DSP несовместимы (ADC – DAC),
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается (не поддерживается НСР).

Функция не меняет признак *modify* в драйвере.

## Установка режима работы заданного канала модуля НСР

***ZSetHCPADC*** **-** включает/выключает питание постоянным током по заданному каналу модуля НСР.

*long* ***ZSetHCPADC*** *(long typeDevice, long numberDSP, long numberChannel, long enable)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*numberChannel* – номер канала, нумерация начинается с 0 (например, 0,1,2,3 и т.д.).
*enable* – установка режима работы модуля НСР для заданного канала. Значение
1 – включить питание по заданному каналу модуля НСР,
0 – выключить питание по заданному каналу модуля НСР.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0002 – функция и режим работы DSP несовместимы (ADC – DAC),
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,

© ООО "ЭТМС" 1992-2024. Все права защищены

0xFF00 – для данного типа устройства функция не поддерживается (не поддерживается НСР).

Функция меняет признак *modify* в драйвере.

## 1.14.Управление цифровым портом (вход-выход)

### Опрос маски выходов цифрового порта

***ZGetDigOutEnable -*** опрашивает битовую маску разрешения работы на выход входов-выходов цифрового порта.

*long* ***ZGetDigOutEnable*** *(long typeDevice, long numberDSP, unsigned long *digitalOutEnableMask)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*digitalOutEnableMask* – маска выходов цифрового порта. 1 – разрешение работы в качестве выхода, 0 – работа в качестве входа.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция не меняет признак *modify* в драйвере.

### Опрос количества линий цифрового порта

***ZGetQuantityChannelDigPort*** **-** опрашивает количество линий цифрового порта.

*long* ***ZGetQuantityChannelDigPort*** *(long typeDevice, long numberDSP, long *quantityChannel)*

Параметры:

© ООО "ЭТМС" 1992-2024. Все права защищены

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
quantityChannel - максимальное количество каналов модуля АЦП/ЦАП.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция не меняет признак *modify* в драйвере.

### Установка маски выходов цифрового порта

***ZSetDigOutEnable -*** устанавливает входы-выходы цифрового порта в режим входа согласно битовой маске.

*long* ***ZSetDigOutEnable*** *(long typeDevice, long numberDSP, unsigned long digitalOutEnableMask)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*digitalOutEnableMask* – маска выходов цифрового порта. 1 – разрешение работы в качестве выхода, 0 – работа в качестве входа.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция меняет признак *modify* в драйвере.

### Чтение данных с входов цифрового порта

***ZGetDigInput -*** производит чтение цифрового порта.

© ООО "ЭТМС" 1992-2024. Все права защищены

*long* ***ZGetDigInput*** *(long typeDevice, long numberDSP, unsigned long *digitalInput)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*digitalInput* – данные цифрового порта.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция не меняет признак *modify* в драйвере.

### Чтение данных, выдаваемых на выходы цифрового порта

***ZGetDigOutput*** **-** считывает данные, которые выдаются на выходы цифрового порта.

*long* ***ZGetDigOutput*** *(long typeDevice, long numberDSP, unsigned long *digitalOutput)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*digitalOutput* – данные, выдаваемые в цифровой порт.

Возвращаемое значение (код ошибки):

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция не меняет признак *modify* в драйвере.

### Запись данных в цифровой порт

***ZSetDigOutput*** **-** записывает данные по выходам цифрового порта.

© ООО "ЭТМС" 1992-2024. Все права защищены

*long* ***ZSetDigOutput*** *(long typeDevice, long numberDSP, unsigned long digitalOutput)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*digitalOutput* – данные, выдаваемые в цифровой порт.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция меняет признак *modify* в драйвере.

## Установить вывод цифрового порта в «1», без изменения состояния других выводов

***ZSetBitDigOutput*** -устанавливает вывод цифрового порта в «1», без изменения состояния других выводов.

*long* ***ZSetBitDigOutput*** *(long typeDevice, long numberDSP, long numberOfBit*)

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
numberOfBit – устанвливается от 0 до максимальное количество каналов модуля АЦП/ЦАП-1.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция меняет признак *modify* в драйвере.

© ООО "ЭТМС" 1992-2024. Все права защищены

## Установить маску выводов цифрового порта в «1», без изменения состояния других выводов

***ZSetBitMaskDigOutput -***устанавливает маску выводов цифрового порта в «1», без изменения состояния других выводов.

*long* ***ZSetBitMaskDigOutput*** *(long typeDevice, long numberDSP, unsigned long maskOfBits)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*maskOfBits* – битовая маска, указывающая выводы, которые будут установлены в «1»

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция меняет признак *modify* в драйвере.

## Установить вывод цифрового порта в «0», без изменения состояния других выводов

***ZClrBitDigOutput -*** устанавливает вывод цифрового порта в «0», без изменения состояния других выводов.

*long* ***ZClrBitDigOutput*** *(long typeDevice, long numberDSP, long numberOfBit)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
numberOfBit – номер вывода, который будет сброшен в «0».

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,

© ООО "ЭТМС" 1992-2024. Все права защищены

0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция меняет признак *modify* в драйвере.

### Установить маску выводов цифрового порта в «0», без изменения состояния других выводов

***ZClrBitMaskDigOutput*** **-** устанавливает маску выводов цифрового порта в «0», без изменения состояния других выводов.

*long* ***ZClrBitMaskDigOutput*** *(long typeDevice, long numberDSP, unsigned long maskOfBits)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*maskOfBits* – битовая маска, указывающая выводы, которые будут установлены в «0»

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция меняет признак *modify* в драйвере.

### Опрос режим потокового ввода-вывода

***ZGetDigitalMode*** **-** опрашивает режим потокового ввода-вывода.

*long* ***ZGetDigitalMode*** *(long typeDevice, long numberDSP, long *mode)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*mode* – получает;
mode=0 - аналоговый порт,

© ООО "ЭТМС" 1992-2024. Все права защищены

mode=1 - цифрового порт,
mode=2 – UART, где // +0x04 - Инвертированный UART
// +0x08 - 9 бит UART
// +0x10 - 2 стоп-бита UART (не исп.)
// +0x20 - контроль четности UART (не исп.)
// +0x40 - IrDA UART (не исп.)
Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция не меняет признак *modify* в драйвере.

## Переключить режим потокового ввода-вывода

***ZSetDigitalMode -*** переключает режим потокового ввода.

*long* ***ZSetDigitalMode*** *(long typeDevice, long numberDSP, long mode)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
mode – получает;
mode=0 - аналоговый порт,
mode=1 - цифрового порт,
mode=2 – UART, где // +0x04 - Инвертированный UART
// +0x08 - 9 бит UART
// +0x10 - 2 стоп-бита UART (не исп.)
// +0x20 - контроль четности UART (не исп.)
// +0x40 - IrDA UART (не исп.)
Возвращаемое значение:

0 – нормальное выполнение функции,

© ООО "ЭТМС" 1992-2024. Все права защищены

0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция меняет признак *modify* в драйвере.

### Установить линию цифрового порта на выход

***ZSetBitDigOutEnable*** **-** устанавливает линию цифрового порта на выход (без изменения состояния других линий, numberOfBit = 0..quantityChannel-1)

*long* ***ZSetBitDigOutEnable*** *(long typeDevice, long numberDSP, long numberOfBit)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*numberOfBit* - устанвливается от 0 до максимальное количество каналов модуля АЦП/ЦАП-1.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция меняет признак *modify* в драйвере.

### Установить маску линий цифрового порта на выход

***ZSetBitMaskDigOutEnable*** **-** устанавливает маску линий цифрового порта на выход (без изменения состояния других линий).

*long* ***ZSetBitMaskDigOutEnable*** *(long typeDevice, long numberDSP, unsigned long maskOfBits)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*maskOfBits* – битовая маска выводов, которые будут направлены на выход,

© ООО "ЭТМС" 1992-2024. Все права защищены

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция меняет признак *modify* в драйвере.

### Установить линию цифрового порта на вход

***ZClrBitDigOutEnable -*** устанавливает линию цифрового порта на вход (без изменения состояния других линий, numberOfBit = 0..quantityChannel-1)

*long* ***ZClrBitDigOutEnable*** *(long typeDevice, long numberDSP, long numberOfBit)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*numberOfBit* - устанвливается от 0 до максимальное количество каналов модуля АЦП/ЦАП-1.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция меняет признак *modify* в драйвере.

### Установить маску линий цифрового порта на вход

***ZClrBitMaskDigOutEnable -*** устанавливает маску линий цифрового порта на выход (без изменения состояния других линий).

*long* ***ZClrBitMaskDigOutEnable*** *(long typeDevice, long numberDSP, unsigned long maskOfBits)*

© ООО "ЭТМС" 1992-2024. Все права защищены

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*maskOfBits* – битовая маска выводов, которые будут направлены на вход,

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция меняет признак *modify* в драйвере.

## 1.15.Управление устройством через Ethernet

### Опрос IP-адреса по Ethernet

***ZnetPing -*** опрашивает IP-адрес подключения по Ethernet.

*long* ***ZnetPing*** *(char *strIP, long *Timeout)*

Параметры:

*strIP* – строка, содержащая IP-адрес устройства (например, «192.168.0.100»).
*Timeout* – время ожидания ответа, мс.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0002 – функция и режим работы DSP несовместимы (ADC – DAC),
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция не меняет признак *modify* в драйвере.

### Подключиться с устройством по Ethernet

© ООО "ЭТМС" 1992-2024. Все права защищены

***ZnetOpen*** - подключает устройство по Ethernet.*typeDevice* – тип устройства.

*long* ***ZnetOpen*** *(long typeDevice, long numberDSP, long OpenTimeout, long CommandTimeout, char *AddressIP, long PortIP)*

Параметры:

*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*OpenTimeout* – время ожидания подключения, мс (обычно от 10000 до 60000).
*CommandTimeout* – время ожидания выполнения команды, мс (обычно 10000).
*AddressIP* – строка, содержащая IP-адрес устройства (например, «192.168.0.100»).
*PortIP* – номер порта (например, 1808).

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0002 – функция и режим работы DSP несовместимы (ADC – DAC),
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция не меняет признак *modify* в драйвере.

## Отключиться от устройства по Ethernet

***ZnetClose -*** отключает устройство от Ethernet.

*long* ***ZnetClose*** *(long typeDevice, long numberDSP, long Timeout)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*Timeout* – время ожидания ответа, мс.

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0002 – функция и режим работы DSP несовместимы (ADC – DAC),
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

© ООО "ЭТМС" 1992-2024. Все права защищены

Функция не меняет признак *modify* в драйвере.

### Опрос на установление соединения с устройством через Ethernet

***ZnetGetReady -*** осуществляет опрос установления соединения с устройством через Ethernet.

*long* ***ZnetGetReady*** *(long typeDevice, long numberDSP)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136).

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0002 – функция и режим работы DSP несовместимы (ADC – DAC),
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция не меняет признак *modify* в драйвере.

### Опрос на установление соединения с устройством через Ethernet с опросом IP-адреса

***ZnetGetReadyExt -*** опрашивает установление соединения с устройством через Ethernet с опросом IP-адреса.

*long* ***ZnetGetReadyExt*** *(long typeDevice, long numberDSP, char *AddressIP, long *PortIP)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*AddressIP* – строка, содержащая IP-адрес устройства (например, «192.168.0.100»).
*PortIP* – номер порта (например, 1808).

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0002 – функция и режим работы DSP несовместимы (ADC – DAC),
0x0100 – *numberDSP* задан неправильно,

© ООО "ЭТМС" 1992-2024. Все права защищены

0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция не меняет признак *modify* в драйвере.

## 1.16.Управление настройками TCP/IP

### Прочитать настройки Ethernet устройства для TCP/IP из ПЗУ

***ZReadSettingsEthernet*** **-** читает настройки Ethernet устройства для TCP/IP из ПЗУ.

long ***ZReadSettingsEthernet****(long typeDevice, long numberDSP, unsigned char *sourceIPAdr, unsigned short *sourcePort0, unsigned char *subnetMask, unsigned char *gatewayIPAdr, unsigned char *sourceMACAdr, unsigned char *duplex, unsigned char *speed100Mb, unsigned short *TimeoutWDOG, unsigned short *Reserved)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*sourceIPAdr* – IP-адрес устройства, четыре байта (например, для адреса 192.168.0.100 байты будут иметь значения 100, 0, 168, 192).
*sourcePort0* – номер порта (например, 1808).
*subnetMask* – маска подсети, четыре байта (например, 0, 255, 255, 255).
*gatewayIPAdr* – IP-адрес сетевого шлюза, четыре байта (например, 1, 0, 168, 192).
*sourceMACAdr* – MAC-адрес устройства, шесть байтов.
*duplex* – полный дуплекс (1) или полудуплекс (0) (обычно 1).
*speed100Mb* – скорость: 100 Мбит/сек (1) или 10 Мбит/сек (0) (обычно 1).
*TimeoutWDOG* – время ожидания команд от пользователя, сек (устройство будет автоматически перезагружено, если в течение указанного времени от пользователя или компьютера не придет ни одна команда).
*Reserved* – зарезервировано (значение 0).

Возвращаемое значение:

© ООО "ЭТМС" 1992-2024. Все права защищены

0 – нормальное выполнение функции,
0x0002 – функция и режим работы DSP несовместимы (ADC – DAC),
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция не меняет признак *modify* в драйвере.

## Сохранить настройки Ethernet устройства для TCP/IP в ПЗУ

***ZSaveSettingsEthernet*** **-** сохраняет настройки Ethernet устройства для TCP/IP из ПЗУ.

*long* ***ZSaveSettingsEthernet****(long typeDevice, long numberDSP, unsigned char *sourceIPAdr, unsigned short *sourcePort0, unsigned char *subnetMask, unsigned char *gatewayIPAdr, unsigned char *sourceMACAdr, unsigned char *duplex, unsigned char *speed100Mb, unsigned short TimeoutWDOG, unsigned short Reserved)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*sourceIPAdr* – IP-адрес устройства, четыре байта (например, для адреса 192.168.0.100 байты будут иметь значения 100, 0, 168, 192).
*sourcePort0* – номер порта (например, 1808).
*subnetMask* – маска подсети, четыре байта (например, 0, 255, 255, 255).
*gatewayIPAdr* – IP-адрес сетевого шлюза, четыре байта (например, 1, 0, 168, 192).
*sourceMACAdr* – MAC-адрес устройства, шесть байтов.
*duplex* – полный дуплекс (1) или полудуплекс (0) (обычно 1).
*speed100Mb* – скорость: 100 Мбит/сек (1) или 10 Мбит/сек (0) (обычно 1).
*TimeoutWDOG* – время ожидания команд от пользователя, сек (устройство будет автоматически перезагружено, если в течение указанного времени от пользователя или компьютера не придет ни одна команда).
*Reserved* – зарезервировано (значение 0).

Возвращаемое значение:

© ООО "ЭТМС" 1992-2024. Все права защищены

0 – нормальное выполнение функции,
0x0002 – функция и режим работы DSP несовместимы (ADC – DAC),
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция меняет признак *modify* в драйвере.

## 1.17.Управление настройками PTP

### Прочитать настройки PTP из ПЗУ

***ZReadSettingsPTP -*** читает настройки PTP из ПЗУ.

*long* ***ZReadSettingsPTP****(long typeDevice, long numberDSP, unsigned char *enableMaster, unsigned char *enableSlave, unsigned char *domain, unsigned char *absolutePriority, unsigned char *relativePriority)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*enableMaster* – разрешать устройству выполнять роль мастера синхронизации (1) или не разрешать (0). Выбор мастера синхронизации осуществляется путем сравнения их характеристик и расставленных приоритетов (см. *absolutePriority* и *relativePriority*).
*enableSlave* – разрешать устройству выполнять роль ведомого (1) или не разрешать (0). Если оба параметра *enableMaster* и *enableSlave* будет иметь значение 0, то модуль синхронизации по PTP будет отключен.
*domain* – номер логической подсети (от 0 до 127). В одной сети Ethernet может быть одновременно задействовано несколько логических подсетей, не связанных друг с другом, в каждой будет свой мастер синхронизации.
*absolutePriority* – абсолютный приоритет устройства при выборе мастера синхронизации (от 0 до 255). Мастером будет выбрано устройство с наименьшим значением.
*relativePriority* – относительный приоритет при выборе мастера синхронизации (от 0 до 255). Если у двух устройств, претендующих на роль мастера синхронизации, окажутся одинаковыми значения абсолютного приоритета и характеристики (качество синхронизации, источник времени и т.д.), то будет выбрано устройство, у которого значение относительного приоритет меньше. Если же одинаковыми будут также и относительные приоритеты, то выбор будет сделан на основе лексикографического сравнения их MAC-адресов.

© ООО "ЭТМС" 1992-2024. Все права защищены

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0002 – функция и режим работы DSP несовместимы (ADC – DAC),
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция не меняет признак *modify* в драйвере.

## Сохранить настройки PTP в ПЗУ

***ZSaveSettingsPTP -*** сохраняет настройки PTP в ПЗУ

*long* ***ZSaveSettingsPTP****(long typeDevice, long numberDSP, unsigned char enableMaster, unsigned char enableSlave, unsigned char domain, unsigned char absolutePriority, unsigned char relativePriority)*

Параметры:

*typeDevice* – тип устройства.
*numberDSP* – номер сигнального процессора (значение от 0 до 136),
*enableMaster* – разрешать устройству выполнять роль мастера синхронизации (1) или не разрешать (0). Выбор мастера синхронизации осуществляется путем сравнения их характеристик и расставленных приоритетов (см. *absolutePriority* и *relativePriority*).
*enableSlave* – разрешать устройству выполнять роль ведомого (1) или не разрешать (0). Если оба параметра *enableMaster* и *enableSlave* будет иметь значение 0, то модуль синхронизации по PTP будет отключен.
*domain* – номер логической подсети (от 0 до 127). В одной сети Ethernet может быть одновременно задействовано несколько логических подсетей, не связанных друг с другом, в каждой будет свой мастер синхронизации.
*absolutePriority* – абсолютный приоритет устройства при выборе мастера синхронизации (от 0 до 255). Мастером будет выбрано устройство с наименьшим значением.
*relativePriority* – относительный приоритет при выборе мастера синхронизации (от 0 до 255). Если у двух устройств, претендующих на роль мастера синхронизации, окажутся одинаковыми значения абсолютного приоритета и характеристики (качество синхронизации, источник времени и т.д.), то будет выбрано устройство, у которого значение относительного приоритет меньше. Если же одинаковыми будут также и относительные приоритеты, то выбор будет сделан на основе лексикографического сравнения их MAC-адресов.

© ООО "ЭТМС" 1992-2024. Все права защищены

Возвращаемое значение:

0 – нормальное выполнение функции,
0x0002 – функция и режим работы DSP несовместимы (ADC – DAC),
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.

Функция меняет признак *modify* в драйвере.

# Глава 2.Краткое описание zadc.dll

| | |
|---|---|
| **Подключение к драйверу и отключение** | |
| *long **ZOpen** (long typeDevice, long numberDSP)* | подключение к драйверу. |
| *long **ZClose** (long typeDevice, long numberDSP)* | отключение от драйвера (сигнального процессора) |
| **Сброс и инициализация** | |
| *long **ZResetDSP** (long typeDevice, long numberDSP)* | сбрасывает все сигнальные процессоры одного устройства и останавливает. |
| *long **ZInitDSP** (long typeDevice, long numberDSP, char *fileName)* | инициализирует и загружает в сигнальный процессор исполняемую программу (если устройство не содержит прошитую программу) |
| *long **ZGetSerialNumberDSP** (long typeDevice, long numberDSP, long *serialNumber)* | получает серийный номер DSP |
| *long **ZGetRevisionDSP** (long typeDevice, long numberDSP, long *revisionNumber)* | получает номера ревизии микропрограммы DSP. |
| *long **ZGetNameDevice** (long typeDevice, long numberDSP, char *strName, long maxSizeStr)* | получает имя устройства |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Сервисные функции | |
|---|---|
| *long **ZGetVersion** (long typeDevice, long numberDSP, char* verDSP, char* verDRV, char* verLIB)* | опрос версии прошивки сигнального процессора, драйвера и библиотеки доступа к драйверу |
| *long **ZGetTypeConnection** (long typeDevice, long numberDSP, long *type)* | получает тип интерфейса |
| *long **ZGetDeviceCapabilities** (long typeDevice, long numberDSP, long *capabilities)* | опрашивает аппаратные возможности устройства |
| *long **ZGetData** (long typeDevice, long numberDSP, ADC_INFO *Data*) | получает всю информацию об АЦП из драйвера. |
| *long **ZGetDataExt** (long typeDevice, long numberDSP, ADC_INFO_EXT *Data)* | получает всю информацию об ЦАП из драйвера. |
| *long **ZGetInfo** (long typeDevice, long numberDSP, ADC_INFO *Data)* | получает всю информацию об АЦП, ЦАП из DSP |
| *long **ZPutInfo** (long typeDevice, long numberDSP, ADC_INFO *Data)* | передает всю информацию в DSP (кроме размера прерывания и старта) |
| *long **ZTestCode** (long typeDevice, long numberDSP, long *code0, long *code1, long *code2)* | передает всю информацию в DSP (кроме размера прерывания и старта) |
| *long **ZGetError** (long typeDevice, long numberDSP, long *error)* | возвращает код последней внутренней ошибки драйвера (обработчика прерываний) |
| *long **ZGetModify** (long typeDevice, long numberDSP, long *modify)* | возвращает параметр изменения режима работы сигнального процессора |
| **Установка режима работы сигнального процессора** | |
| *long **ZSetTypeADC** (long typeDevice, long numberDSP)* | устанавливает сигнальный процессор в режим работы с модулем АЦП. |
| *long **ZSetTypeDAC** (long typeDevice, long numberDSP)* | устанавливает сигнальный процессор в режим работы с модулем ЦАП. |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Опрос основных характеристик модулей АЦП и ЦАП | |
|---|---|
| *long **ZGetEnableADC** (long typeDevice, long numberDSP, long *enable)* | опрос возможности работы сигнального процессора с модулем АЦП |
| *long **ZGetEnableDAC** (long typeDevice, long numberDSP, long *enable)* | опрос возможности работы сигнального процессора с модулем ЦАП |
| *long **ZGetQuantityChannelADC** (long typeDevice, long numberDSP, long *quantityChannel)* | опрос максимального количества каналов модуля АЦП |
| *long **ZGetQuantityChannelDAC** (long typeDevice, long numberDSP, long *quantityChannel)* | опрос максимального количества каналов модуля ЦАП |
| *long **ZGetDigitalResolutionADC** (long typeDevice, long numberDSP, double *digitalResolution)* | опрос веса младшего разряда АЦП. |
| *long **ZGetDigitalResolutionDAC** (long typeDevice, long numberDSP, double *digitalResolution)* | опрос веса младшего разряда ЦАП. |
| *long **ZGetBitsADC** (long typeDevice, long numberDSP, long *numberBits)* | опрос разрядности АЦП. |
| *long **ZGetBitsDAC** (long typeDevice, long numberDSP, long *numberBits)* | опрос разрядности ЦАП. |
| *long **ZGetWordsADC** (long typeDevice, long numberDSP, long *numberWords)* | опрос размера каждого отсчета АЦП в 16-разрядных словах. |
| *long **ZGetWordsDAC** (long typeDevice, long numberDSP, long *numberWords)* | опрос размера каждого отсчета ЦАП в 16-разрядных словах. |
| **Установка частоты дискретизации и режима синхронизации АЦП/ЦАП** | |
| **Опрос и установление частоты дискретизации** | |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *long* ***ZGetListFreqADC*** *(long typeDevice, long numberDSP, long next, double *freq)* | получение списка возможных частот дискретизации АЦП |
| *long* ***ZGetListFreqDAC*** *(long typeDevice, long numberDSP, long next, double *freq)* | получение списка возможных частот дискретизации ЦАП |
| *long* ***ZSetNextFreqADC****(long typeDevice, long numberDSP, long next, double *freq)* | устанавливает большую или меньшую частоту дискретизации по сравнению с текущей частотой дискретизации АЦП |
| *long* ***ZSetNextFreqDAC****(long typeDevice, long numberDSP, long next, double *freq)* | устанавливает большую или меньшую частоту дискретизации по сравнению с текущей частотой дискретизации ЦАП |
| *long* ***ZGetFreqADC****(long typeDevice, long numberDSP, double *freq)* | опрос текущего значения частоты дискретизации АЦП |
| *long* ***ZGetFreqDAC****(long typeDevice, long numberDSP, double *freq)* | опрос текущего значения частоты дискретизации ЦАП |
| *long* ***ZSetFreqADC****(long typeDevice, long numberDSP, double freqIn, double *freqOut)* | устанавливает частоту дискретизации, наиболее близкую к требуемой, и возвращает значение установленной частоты дискретизации АЦП |
| *long* ***ZSetFreqDAC****(long typeDevice, long numberDSP, double freqIn, double *freqOut)* | устанавливает частоту дискретизации, наиболее близкую к требуемой, и возвращает значение установленной частоты дискретизации ЦАП |
| **Опрос и установление внешней опорной частоты** | |
| *long* ***ZGetExtFreqADC****(long typeDevice, long numberDSP, double *oporFreq)* | опрос значения текущей опорной частоты АЦП |
| *long* ***ZGetExtFreqDAC****(long typeDevice, long numberDSP, double *oporFreq)* | опрос значения текущей опорной частоты ЦАП |
| *long* ***ZSetExtFreqADC****(long typeDevice, long numberDSP, double extFreq)* | установление значения внешней опорной частоты АЦП |
| *long* ***ZSetExtFreqDAC****(long typeDevice, long numberDSP, double extFreq)* | установление значения внешней опорной частоты ЦАП |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *long **ZGetEnableExtFreq**(long typeDevice, long numberDSP, long *enable)* | опрос режима работы от внешней опорной частоты. |
| *long **ZSetEnableExtFreq**(long typeDevice, long numberDSP, long enable)* | установление или сбрасывание режима работы от внешней опорной частоты. |
| **Опрос и установление начала накопления данных от внешнего сигнала** | |
| *long **ZGetEnableExtStart**(long typeDevice, long numberDSP, long *enable)* | опрос разрешение/запрещение начала накопления данных от внешнего сигнала |
| *long **ZSetEnableExtStart**(long typeDevice, long numberDSP, long enable)* | разрешает/запрещает начало накопления данных от внешнего сигнала запуска. |
| **Определение коэффициента деления и устанавления частоты преобразования** | |
| *long **ZGetRateADC** (long typeDevice, long numberDSP, long *rate)* | определяет установленный коэффициент деления АЦП (служебн., пользоваться не рекомендуется) |
| *long **ZGetRateDAC** (long typeDevice, long numberDSP, long *rate)* | определяет установленный коэффициент деления АЦП (служебн., пользоваться не рекомендуется). |
| *long **ZSetRateADC** (long typeDevice, long numberDSP, long rate)* | устанавливает частоту преобразования АЦП (служебн., пользоваться не рекомендуется). |
| *long **ZSetRateDAC** (long typeDevice, long numberDSP, long rate)* | устанавливает частоту преобразования ЦАП (служебн., пользоваться не рекомендуется). |
| **Опрос и установление синхронизации по внешней частоте** | |
| *long **ZGetEnableExtFreqADC** (long typeDevice, long numberDSP, long *enable)* | прочитывает статус синхронизации по внешней частоте АЦП. |
| *long **ZGetEnableExtFreqDAC** (long typeDevice, long numberDSP, long *enable)* | прочитывает статус синхронизации по внешней частоте ЦАП. |
| *long **ZSetEnableExtFreqADC** (long typeDevice, long numberDSP, long* | включает или отключает синхронизации по внешней частоте АЦП. |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *enable)* | |
| *long **ZSetEnableExtFreqDAC** (long typeDevice, long numberDSP, long enable)* | включает или отключает синхронизации по внешней частоте ЦАП. |
| *long **ZGetEnableExtStartADC** (long typeDevice, long numberDSP, long *enable)* | читает статус внешнего запуска АЦП. |
| *long **ZGetEnableExtStartDAC** (long typeDevice, long numberDSP, long *enable)* | читает статус внешнего запуска ЦАП. |
| *long **ZSetEnableExtStartADC** (long typeDevice, long numberDSP, long enable)* | включает или отключает внешний запуск АЦП. |
| *long **ZSetEnableExtStartDAC** (long typeDevice, long numberDSP, long enable)* | включает или отключает внешний запуск ЦАП. |
| **Управление каналами ввода (вывода) АЦП/ЦАП** | |
| *long **ZGetNumberInputADC** (long typeDevice, long numberDSP, long *workChannel)* | опрос количества включенных каналов для ввода данных АЦП. |
| *Long **ZGetNumberOutputDAC** (long typeDevice, long numberDSP, long *workChannel)* | опрос количества включенных каналов для ввода данных ЦАП |
| *long **ZGetInputADC** (long typeDevice, long numberDSP, long numberChannel, long *enable)* | опрос разрешения канала для ввода АЦП |
| *long **ZGetInputDAC** (long typeDevice, long numberDSP, long numberChannel, long *enable)* | опрос разрешения канала для ввода ЦАП |
| *Long **ZSetInputADC** (long typeDevice, long numberDSP, long numberChannel, long enable)* | выбор или отмена канала для ввода АЦП |
| *long **ZSetOutputDAC** (long typeDevice, long numberDSP, long numberChannel, long enable)* | выбор или отмена канала для ввода ЦАП |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *long **ZSetChannelADC** (long typeDevice, long numberDSP, unsigned long channelMask)* | включения/выключения каналов с помощью битовой маски АЦП, (служебн., пользоваться не рекомендуется). |
| *long **ZSetChannelDAC** (long typeDevice, long numberDSP, unsigned long channelMask)* | включения/выключения каналов с помощью битовой маски ЦАП, (служебн., пользоваться не рекомендуется). |
| *long **ZGetInputDiffADC** (long typeDevice, long numberDSP, long numberChannel, long *enable)* | опрос типа канала для ввода (синфазный или дифференциальный). |
| *long **ZSetInputDiffADC** (long typeDevice, long numberDSP, long numberChannel, long enable)* | установление типа канала для ввода (синфазный или дифференциальный). |
| *long **ZGetModaADC** (long typeDevice, long numberDSP, long *moda)* | чтение режима работы АЦП |
| *long **ZSetModaADC** (long typeDevice, long numberDSP, long moda)* | установление режима работы АЦП, (служебн., пользоваться не рекомендуется) |
| **Управление коэффициентами усиления АЦП** | |
| *long **ZGetListAmplifyADC** (long typeDevice, long numberDSP, long next, double *amplify)* | получение списка возможных коэффициентов усиления |
| *long **ZSetNextAmplifyADC** (long typeDevice, long numberDSP, long numberChannel, long next, double *amplify)* | установление большего или меньшего коэффициента усиления выбранного канала. |
| *long **ZGetAmplifyADC** (long typeDevice, long numberDSP, long numberChannel, double *amplify)* | опрос коэффициента усиления выбранного канала. |
| *long **ZSetAmplifyADC** (long typeDevice, long numberDSP, long numberChannel, double amplifyIn, double *amplifyOut)* | установление коэффициента усиления выбранного канала. |
| **Управление коэффициентами усиления ПУ 8/10** | |
| *long **ZGetListPreAmplifyADC** (long typeDevice, long numberDSP, long* | получение списка возможных коэффициентов усиления предварительного усилителя |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *next, double *amplify)* | |
| *long **ZSetNextPreAmplifyADC** (long typeDevice, long numberDSP, long numberChannel, long next, double *amplify)* | установление большего или меньшего коэффициента усиления предварительного усилителя выбранного канала |
| *long **ZGetPreAmplifyADC** (long typeDevice, long numberDSP, long numberChannel, double *amplify)* | опрос коэффициента усиления предварительного усилителя выбранного канала |
| *long **ZSetPreAmplifyADC** (long typeDevice, long numberDSP, long numberChannel, double amplifyIn, double *amplifyOut)* | установление коэффициента усиления предварительного усилителя выбранного канала |
| **Управление коэффициентами ослабления аттенюатора ЦАП** | |
| *long **ZFindSoftAtten** (long typeDevice, long numberDSP, long *present)* | опрос коэффициента ослабления аттенюатора выбранного канала |
| *long **ZGetAttenDAC** (long typeDevice, long numberDSP, long numberChannel, double *reduction)* | опрос коэффициента ослабления аттенюатора выбранного канала |
| *long **ZSetAttenDAC** (long typeDevice, long numberDSP, long numberChannel, double reductionIn, double *reductionOut)* | установление коэффициента ослабления аттенюатора заданного канала аналогового вывода |
| **Управление процессом перекачки данных** | |
| *long **ZSetExtCycleDAC** (long typeDevice, long numberDSP, long enable)* | установление режима работы ЦАП |
| *long **ZGetMaxSizeBufferDSPDAC** (long typeDevice, long numberDSP, long *size)* | опрашивает максимальный размер буфера в ЦАП в DSP |
| *long **ZGetSizeBufferDSPDAC** (long typeDevice, long numberDSP, long *size)* | опрашивает размер буфера в ЦАП в DSP |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *long* ***ZSetSizeBufferDSPDAC*** *(long typeDevice, long numberDSP, long size)* | устанавливает размер буфера в ЦАП в DSP |
| *long* ***ZGetInterruptADC*** *(long typeDevice, long numberDSP, long * size)* | опрос размера буфера для перекачки данных за одно прерывание АЦП |
| *long* ***ZGetInterruptDAC*** *(long typeDevice, long numberDSP, long * size)* | опрос размера буфера для перекачки данных за одно прерывание ЦАП |
| *long* ***ZGetMaxInterruptADC*** *(long typeDevice, long numberDSP, long * size)* | опрос размера буфера для перекачки данных за одно прерывание АЦП |
| *long* ***ZGetMaxInterruptDAC*** *(long typeDevice, long numberDSP, long * size)* | опрос размера буфера для перекачки данных за одно прерывание ЦАП |
| *long* ***ZSetInterruptADC*** *(long typeDevice, long numberDSP, long size)* | установление размера буфера памяти сигнального процессора для перекачки данных из памяти сигнального процессора в память компьютера во время каждого прерывания АЦП |
| *long* ***ZSetInterruptDAC*** *(long typeDevice, long numberDSP, long size)* | установление размера буфера памяти сигнального процессора для перекачки данных из памяти сигнального процессора в память компьютера во время каждого прерывания ЦАП |
| *long* ***ZSetBufferSizeADC*** *(long typeDevice, long numberDSP, long size)* | устанавливает размер буфера накопления в памяти процессора АЦП |
| *long* ***ZSetBufferSizeDAC*** *(long typeDevice, long numberDSP, long size)* | устанавливает размер буфера накопления в памяти процессора ЦАП |
| *long* ***ZGetBufferADC*** *(long typeDevice, long numberDSP, void **buffer, long *size)* | отображение буфера данных, опрос адреса и размера буфера данных АЦП. |
| *long* ***ZGetBufferDAC*** *(long typeDevice, long numberDSP, void **buffer, long *size)* | отображение буфера данных, опрос адреса и размера буфера данных ЦАП |
| *long* ***ZRemBufferADC*** *(long typeDevice, long numberDSP, void **buffer)* | освобождает дескрипторы отображения памяти для доступа к буферу данных драйвера АЦП |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *long **ZRemBufferDAC** (long typeDevice, long numberDSP, void **buffer)* | освобождает дескрипторы отображения памяти для доступа к буферу данных драйвера ЦАП |
| *long **ZSetCycleSampleADC** (long typeDevice, long numberDSP, long enable)* | установление циклического или одноразового накопления в буфер данных центрального процессора АЦП |
| *long **ZSetCycleSampleDAC** (long typeDevice, long numberDSP, long enable)* | установление циклического или одноразового накопления в буфер данных центрального процессора ЦАП |
| *long **ZGetLastDataADC** (long typeDevice, long numberDSP, long numberChannel, void *buffer, long size)* | пересылает последние полученные отсчеты по заданному каналу в буфер данных пользователя АЦП |
| *long **ZGetPointerADC** (long typeDevice, long numberDSP, long *pointer)* | опрос указателя накопления в буфер данных центрального процессора АЦП. |
| *long **ZGetPointerDAC** (long typeDevice, long numberDSP, long *pointer)* | опрос указателя накопления в буфер данных центрального процессора ЦАП |
| *long **ZGetFlag** (long typeDevice, long numberDSP, unsigned long *flag)* | опрашивает флаг прерываний |
| *long **ZGetStartADC** (long typeDevice, long numberDSP, long *status)* | проверяет состояние сигнального процессора АЦП |
| *long **ZGetStartDAC** (long typeDevice, long numberDSP, long *status)* | проверяет состояние сигнального процессора ЦАП |
| *long **ZStartADC** (long typeDevice, long numberDSP)* | запускает процесс накопления данных в память буфера накопления данных АЦП. |
| *long **ZStartDAC** (long typeDevice, long numberDSP)* | запускает процесс накопления данных в память буфера накопления данных ЦАП |
| *long **ZStopADC** (long typeDevice, long numberDSP)* | останавливает процесс накопления данных АЦП |
| *long **ZStopDAC** (long typeDevice, long numberDSP)* | останавливает процесс накопления данных ЦАП |
| *long **ZGetSizePacketADC** (long typeDevice, long numberDSP, long *size)* | определяет установленный размер пакета данных DSP АЦП |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *long* ***ZGetSizePacketDAC*** *(long typeDevice, long numberDSP, long *size)* | определяет установленный размер пакета данных DSP ЦАП |
| *long* ***ZGetMaxSizePacketADC****(long typeDevice, long numberDSP, long *size)* | определяет максимально возможный размер пакета данных DSP АЦП |
| *long* ***ZGetMaxSizePacketDAC****(long typeDevice, long numberDSP, long *size)* | определяет максимально возможный размер пакета данных DSP ЦАП |
| *long* ***ZSetSizePacketADC*** *(long typeDevice, long numberDSP, long size)* | устанавливает размер пакета данных DSP АЦП |
| *long* ***ZSetSizePacketDAC*** *(long typeDevice, long numberDSP, long size)* | устанавливает размер пакета данных DSP АЦП |
| *long* ***ZGetQuantityPacketsADC*** *(long typeDevice, long numberDSP, long *size)* | определяет установленное количество пакетов за одно прерывание АЦП |
| *long* ***ZGetQuantityPacketsDAC*** *(long typeDevice, long numberDSP, long *size)* | определяет установленное количество пакетов за одно прерывание ЦАП |
| *long* ***ZGetMaxQuantityPacketsADC*** *(long typeDevice, long numberDSP, long *size)* | определяет максимальное возможное количество пакетов за одно прерывание АЦП |
| *long* ***ZGetMaxQuantityPacketsDAC*** *(long typeDevice, long numberDSP, long *size)* | определяет максимальное возможное количество пакетов за одно прерывание ЦАП |
| *long* ***ZSetQuantityPacketsADC*** *(long typeDevice, long numberDSP, long size)* | устанавливает количество пакетов за одно прерывание АЦП |
| *long* ***ZSetQuantityPacketsDAC*** *(long typeDevice, long numberDSP, long size)* | устанавливает количество пакетов за одно прерывание ЦАП |
| **Управление модулем HCP** | |
| *long* ***ZFindHCPADC*** *(long typeDevice, long numberDSP, long* | опрашивает поддерживается ли модуль HCP для питания датчиков стандарта ICP |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| **present)* | |
| *long* ***ZGetHCPADC*** *(long typeDevice, long numberDSP, long numberChannel long *enable)* | опрашивает, включено ли питание постоянным током по заданному каналу модуля НСР. |
| *long* ***ZSetHCPADC*** *(long typeDevice, long numberDSP, long numberChannel, long enable)* | включает/выключает питание постоянным током по заданному каналу модуля НСР |
| **Управление цифровым портом (вход-выход)** | |
| *long* ***ZGetDigOutEnable*** *(long typeDevice, long numberDSP, unsigned long *digitalOutEnableMask)* | опрашивает битовую маску разрешения работы на выход входов-выходов цифрового порта |
| *long* ***ZGetQuantityChannelDigPort*** *(long typeDevice, long numberDSP, long *quantityChannel)* | опрашивает количество линий цифрового порта |
| *long* ***ZSetDigOutEnable*** *(long typeDevice, long numberDSP, unsigned long digitalOutEnableMask)* | устанавливает входы-выходы цифрового порта в режим входа согласно битовой маске |
| *long* ***ZGetDigInput*** *(long typeDevice, long numberDSP, unsigned long *digitalInput)* | производит чтение цифрового порта |
| *long* ***ZGetDigOutput*** *(long typeDevice, long numberDSP, unsigned long *digitalOutput)* | считывает данные, которые выдаются на выходы цифрового порта |
| *long* ***ZSetDigOutput*** *(long typeDevice, long numberDSP, unsigned long digitalOutput)* | записывает данные по выходам цифрового порта |
| *long* ***ZSetBitDigOutput*** *(long typeDevice, long numberDSP, long numberOfBit*) | устанавливает вывод цифрового порта в «1», без изменения состояния других выводов |
| *long* ***ZSetBitMaskDigOutput*** *(long typeDevice, long numberDSP, unsigned long maskOfBits)* | устанавливает маску выводов цифрового порта в «1», без изменения состояния других выводов |
| *long* ***ZClrBitDigOutput*** *(long typeDevice, long numberDSP, long numberOfBit)* | устанавливает вывод цифрового порта в «0», без изменения состояния других выводов. |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *long* ***ZClrBitMaskDigOutput*** *(long typeDevice, long numberDSP, unsigned long maskOfBits)* | устанавливает маску выводов цифрового порта в «0», без изменения состояния других выводов |
| *long* ***ZGetDigitalMode*** *(long typeDevice, long numberDSP, long *mode)* | опрашивает режим потокового ввода-вывода |
| *long* ***ZSetDigitalMode*** *(long typeDevice, long numberDSP, long mode)* | переключает режим потокового ввода |
| *long* ***ZSetBitDigOutEnable*** *(long typeDevice, long numberDSP, long numberOfBit)* | устанавливает линию цифрового порта на выход (без изменения состояния других линий, numberOfBit = 0..quantityChannel-1) |
| *long* ***ZSetBitMaskDigOutEnable*** *(long typeDevice, long numberDSP, unsigned long maskOfBits)* | устанавливает маску линий цифрового порта на выход (без изменения состояния других линий). |
| *long* ***ZClrBitDigOutEnable*** *(long typeDevice, long numberDSP, long numberOfBit)* | устанавливает линию цифрового порта на вход (без изменения состояния других линий, numberOfBit = 0..quantityChannel-1) |
| *long* ***ZClrBitMaskDigOutEnable*** *(long typeDevice, long numberDSP, unsigned long maskOfBits)* | устанавливает маску линий цифрового порта на выход (без изменения состояния других линий). |
| **Управление цифровым портом (синхронизация)** | |
| **Работа с GPS** | |
| *long* ***ZGetMasterSynchr*** *(long typeDevice, long numberDSP, long *enable)* | опрашивает разрешения генерации сигналов синхронизации на цифровой порт |
| *long* ***ZSetMasterSynchr*** *(long typeDevice, long numberDSP, long enable)* | разрешает генерацию сигналов синхронизации на цифровой порт |
| *long* ***ZStartSynhrQuartz*** *(long typeDevice, long numberDSP)* | запускает синхронизацию опорного кварцевого генератора |
| *long* ***ZStopSynhrQuartz*** *(long typeDevice, long numberDSP)* | запускает синхронизацию опорного кварцевого генератора |
| *long* ***ZGetSynhrTimeQuartz*** *(long typeDevice, long numberDSP,* | опрашивает временные параметры синхронизации |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *SYNHR_QUARTZ_INFO *synhrElement)* | |
| *long **ZTrimmingQuartz** (long typeDevice, long numberDSP, long size)* | подстраивает частоту опорного кварцевого генератора |
| *long **ZFindUART** (long typeDevice, long numberDSP, long *present)* | проверяет поддерживается ли модуль GPS |
| *long **ZGetGPSTime** (long typeDevice, long numberDSP, ULONGLONG *time)* | опрашивает дату и время GPS |
| *long **ZGetStartTime** (long typeDevice, long numberDSP, ULONGLONG *time)* | опрашивает дату и время старта |
| *long **ZSetStartTime** (long typeDevice, long numberDSP, ULONGLONG time)* | устанавливает дату и время старта, time=0 - отключить запуск по времени |
| **Работа с UART** | |
| *long **ZSetConfigUART** (long typeDevice, long numberDSP, long baudRate, short stopBits, short ConfigParity)* | сконфигурирует UART |
| *long **ZReadUART** (long typeDevice, long numberDSP, void *buffer, long *size)* | принимает данные напрямую с UART |
| *long **ZWriteUART** (long typeDevice, long numberDSP, void *buffer, long *size)* | посылает данные напрямую в UART |
| *long **ZStartBufReadingUART** (long typeDevice, long numberDSP)* | запускает поток для буферизированного чтения UART |
| *long **ZStartBufWritingUART** (long typeDevice, long numberDSP)* | запускает поток для буферизированной записи UART |
| *long **ZStopBufReadingUART** (long typeDevice, long numberDSP)* | останавливает поток для буферизированного чтения UART |
| *long **ZStopBufWritingUART** (long typeDevice, long numberDSP)* | останавливает поток для буферизированной записи UART |
| *long **ZReadBufUART** (long typeDevice, long numberDSP, void* | читает данные UART через буфер |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| **buffer, long *size)* | |
| *long **ZWriteBufUART** (long typeDevice, long numberDSP, void *buffer, long *size)* | записать данные UART через буфер |
| *long **ZSwitchUART**(long typeDevice, long numberDSP, long switchRead, long switchWrite)* | переключает порта UART между GPS и акустическим модемом |
| *long **ZReadFromUART** (long typeDevice, long numberDSP, void *buffer, long *size)* | читает некоторое количество байт побайтно непосредственно с UART |
| *long **ZSetRegistratorMode** (long typeDevice, long numberDSP)* | переходит в режим регистрации |
| *long **ZResetAM** (long typeDevice, long numberDSP)* | перезагружает акустический модем |
| *long **ZReadUT** (long typeDevice, long numberDSP, float *U1, float *U2, float *U3, float *U4, float *T)* | считывает уровень напряжения на входах дополнительного АЦП и температуры |
| *long **ZAsynchrGate** (long typeDevice, long numberDSP, long gate)* | управляет асинхронным импульсом |
| **Работа с ШИМ** | |
| *long **ZFindPWM** (long typeDevice, long numberDSP, long* present)* | опрос поддерживается ли модуль ШИМ |
| *long **ZStartPWM** (long typeDevice, long numberDSP,long Start0, long Start1, long Start2)* | запускает каналы ШИМ |
| *long **ZStopPWM** (long typeDevice, long numberDSP, long Stop0, long Stop1, long Stop2)* | останавливает каналы ШИМ |
| *long **ZSetFreqPWM** (long typeDevice, long numberDSP, long Rate, long Period)* | задает частоту ШИМ через коэф. деления опорной частоты и период |
| *long **ZSetOnDutyPWM** (long typeDevice, long numberDSP, long OnDutyPWM0, long OnDutyPWM1, long OnDutyPWM2, long ShiftPWM1, long ShiftPWM2)* | задает скважность и сдвиг каналов ШИМ |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *long **ZRegulatorPWM** (long typeDevice, long numberDSP, void *data, long *size)* | осуществляет обмен массивом коэффициентов управления ШИМ с DSP |
| **Работа с программами** | |
| *long **ZExclusive1Open** (long typeDevice, long numberDSP)* | выполняет переход в эксклюзивный режим №1 (заблокирован только доступ к устройству, открытие и выполнение функций драйвера - разрешено) |
| *long **ZExclusive1Close** (long typeDevice, long numberDSP)* | выполняется выход из эксклюзивного режима №1 (разблокировать только доступ к устройству, открытие и выполнение функций драйвера - разрешено) |
| *long **ZExclusive2Open** (long typeDevice, long numberDSP)* | выполняет переход в эксклюзивный режим №2 (заблокирован и к устройству и к драйверу, т.е. устройство становится скрытым) |
| *long **ZExclusive2Close** (long typeDevice, long numberDSP)* | выполняется выход из эксклюзивного режима №2 (разблокировать устройство и драйвер, т.е. устройство становится видимым) |
| *long **ZSendCommand** (long typeDevice, long numberDSP, void *buffer, long *size)* | послать команду *DSP* |
| *long **ZReceiveCommand** (long typeDevice, long numberDSP, void *buffer, long *size)* | принять команду *DSP* |
| *long **ZSendDataDAC** (long typeDevice, long numberDSP, void *buffer, long *size)* | послать данные ЦАП |
| *long **ZReceiveDataADC** (long typeDevice, long numberDSP, void *buffer, long *size)* | принять данные АЦП |
| **Управление устройством через Ethernet** | |
| *long **ZnetPing** (char *strIP, long *Timeout)* | опрашивает IP-адрес подключения по Ethernet |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *long* ***ZnetOpen*** *(long typeDevice, long numberDSP, long OpenTimeout, long CommandTimeout, char *AddressIP, long PortIP)* | подключает устройство по Ethernet.*typeDevice* – тип устройства |
| *long* ***ZnetClose*** *(long typeDevice, long numberDSP, long Timeout)* | отключает устройство от Ethernet |
| *long* ***ZnetGetReady*** *(long typeDevice, long numberDSP)* | осуществляет опрос установления соединения с устройством через Ethernet |
| *long* ***ZnetGetReadyExt*** *(long typeDevice, long numberDSP, char *AddressIP, long *PortIP)* | опрашивает установление соединения с устройством через Ethernet с опросом IP-адреса |
| **Управление настройками TCP/IP** | |
| long ***ZReadSettingsEthernet****(long typeDevice, long numberDSP, unsigned char *sourceIPAdr, unsigned short *sourcePort0, unsigned char *subnetMask, unsigned char *gatewayIPAdr, unsigned char *sourceMACAdr, unsigned char *duplex, unsigned char *speed100Mb, unsigned short *TimeoutWDOG, unsigned short *Reserved)* | читает настройки Ethernet устройства для TCP/IP из ПЗУ |
| *long* ***ZSaveSettingsEthernet****(long typeDevice, long numberDSP, unsigned char *sourceIPAdr, unsigned short *sourcePort0, unsigned char *subnetMask, unsigned char *gatewayIPAdr, unsigned char *sourceMACAdr, unsigned char *duplex, unsigned char *speed100Mb, unsigned short TimeoutWDOG, unsigned short Reserved)* | сохраняет настройки Ethernet устройства для TCP/IP из ПЗУ |
| **Управление настройками PTP** | |
| *long* ***ZReadSettingsPTP****(long typeDevice, long numberDSP, unsigned char *enableMaster,* | читает настройки PTP из ПЗУ |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *unsigned char *enableSlave, unsigned char domain, unsigned char *absolutePriority, unsigned char *relativePriority)* | |
| *long* ***ZSaveSettingsPTP****(long typeDevice, long numberDSP, unsigned char enableMaster, unsigned char enableSlave, unsigned char domain, unsigned char absolutePriority, unsigned char relativePriority)* | сохраняет настройки PTP в ПЗУ |
| **Функции калибровки АЦП/ЦАП** | |
| *long* ***ZGetCalibrDataADC*** *(long typeDevice, long numberDSP, long numberChannel, double *digitalResolChan, double *offsetChan)* | читает калибровочные константы устройства АЦП из ПЗУ |
| *long* ***ZGetCalibrDataDAC****(long typeDevice, long numberDSP, long numberChannel, double *digitalResolChan, double *offsetChan)* | читает калибровочные константы устройства ЦАП из ПЗУ |
| *long* ***ZSetCalibrDataADC*** *(long typeDevice, long numberDSP, long numberChannel, double digitalResolChan, double offsetChan)* | записывает калибровочные константы устройства АЦП в ПЗУ |
| *long* ***ZSetCalibrDataDAC****(long typeDevice, long numberDSP, long numberChannel, double digitalResolChan, double offsetChan)* | записывает калибровочные константы устройства ЦАП в ПЗУ |
| *long* ***ZReadCalibrData*** *(long typeDevice, long numberDSP)* | читает калибровочные константы из ПЗУ в драйвер |
| *long* ***ZSetDefaultCalibrData*** *(long typeDevice, long numberDSP)* | устанавливает калибровочные константы по-умолчанию из драйвера в ПЗУ |
| *long* ***ZSaveCalibrData*** *(long typeDevice, long numberDSP)* | записывает калибровочные константы из драйвера в ПЗУ |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *long* ***ZGetDigitalResolChanADC****(long typeDevice, long numberDSP, long numberChannel, double *digitalResolChan)* | чтения откалиброванных поканально весов младшего разряда АЦП (в вольтах). |
| *long* ***ZGetDigitalResolChanDAC****(long typeDevice, long numberDSP, long numberChannel, double *digitalResolChan)* | чтения откалиброванных поканально весов младшего разряда ЦАП (в вольтах). |

© ООО "ЭТМС" 1992-2024. Все права защищены

# Unit-2. Предназначен для связи с программами

## 1 Общие положения

1.1 UNIT – это механизм передачи данных между двумя приложениями, запущенными на одном компьютере. Он предназначен для передачи в одну строну результатов, в другую сторону – параметров. Данные передаются через уникальный FileMapping, создаваемый для каждой пары приложений.

1.2 UNIT-2 состоит из двух частей, двух ActiveX-компонента: серверной части *UnitSrv.ocx* и клиетской части *UnitCln.ocx*. Серверная часть – это та часть, экземпляр которой реализован в приложении, которое получает некоторый результат и передаёт его клиентской части. Клиентская часть – эта та часть, экземпляр которой реализован в приложении, которое запускает серверное приложение, настраивает его с помощью параметров и принимает результаты от серверного приложения для принятия соответствующий действий. Пример сервера – программа типа вольтметр. Пример клиента – SCADA-проект.

1.3 У сервера может быть только один клиент. Поэтому серверное приложение должно иметь один экземпляр *UnitSrv.ocx*. Однако клиент может получать результаты от разных серверов, поэтому для каждого сервера в клиентском приложении должно быть по экземпляру *UnitCln.ocx*.

1.4 Серверное приложение может в свою очередь получать результаты от третьего приложения, т.е. выступать в роли клиента по отношению к этому третьему приложению. В этом случае у серверного приложения должен быть экземпляр *UnitCln.ocx* для работы с этим третьим приложением.

1.5 Во время работы клиент и сервер должны периодически проверять друг друга на предмет существования и отсутствия зависаний.

1.6. При работе обе части не используют глобальные переменные. Клиентское приложение запускает серверное приложение с ключом "-u" в командной строке. Для каждой пары клиент/сервер создаётся своя общая область памяти, т.е. именованный FileMapping, имя которого образуется с помощью вновь созданного GUID. Это имя также передаётся серверному приложению с помощью командной строки. Объём общей области памяти – адаптивная величина, определяемая серверным приложением (для одного сервера результат – это число *float,* для другого – это массив *float*). Кроме этого, размер общей области памяти должен быть кратен размеру страницы памяти.

© ООО "ЭТМС" 1992-2024. Все права защищены

1.7 Функции обеих частей возвращают значение типа *long*. Если это значение отрицательное, то это код ошибки. Все функции имеют одинаковые коды ошибок.

1.8 В настоящее время базовый класс *CDialog_ZET* доработан для работы с UNIT-2. На эту версию UNIT переведены несколько программы, в том числе программы "Вольтметр переменного тока" и " Вольтметр постоянного тока".

1.9 Номера результатов и свойств программ для UNIT-2 повторяют соответствующие номера UNIT.

# Глава 1.Установка компонента

ActiveX компоненты UNIT-2 (*UnitSrv.ocx и UnitCln.ocx*), как и любой другой ActiveX-компонент можно установить несколькими способами, например:

- в редакторе ресурсов мышкой перетащить компонент из панели элементов на форму диалогового окна и с помощью мастера добавления создать в классе диалогового окна переменную, задав её имя;
- в h-файле диалогового окна объявив переменную типа указатель на класс CDUnitSrv или CDUnitCln, в дальненйшем воспользовавшись оператором new.

В обоих случаях необходимо создавать компонент с помощью функции Create, например так:

```
#define IDC_UNIT_Srv 1705
m_pUnitSrv = new CDUnitSrv();
m_pUnitSrv->Create(L"", WS_CHILD, CRect(0,0,0,0), this,
IDC_UNIT_Srv);
```

где создаётся экземпляр класса сервера UNIT-2 без названия, с заданием родителя и идентификатора.

При разработке программы-сервера UNIT-2 с помощью наследования класса диалогового окна от базового класса CDialog_ZET, то можно воспользоваться экземпляром класса серваера, объявленным в этом базовом классе:

```
CDUnitSrv *m_punitSrv;
```

который создаётся динамически при выполнении функции CDialog_ZET::OnInitDialog, и удаляется при выполнении функции CDialog_ZET::OnDestroy.

# Глава 2.Работа с программами

Описана в главах, описанных ниже.

© ООО "ЭТМС" 1992-2024. Все права защищены

## 2.1.Результаты

**2. Результаты работы UNIT-2**

Сервер UNIT-2 выполняет некоторые вычисления, результаты которых отправляет клиенту. Передаваемые результаты имеют номер и тип.

2.1 Номер результата – это целое число типа long. Результаты разных номеров могут иметь разные типы. Например: под номером 0 программа может передавать среднее амплитудное значение, под номером 1 – пиковое значение, под номером 2 – массив мгновенных амплитуд значений и т.д.

2.2 Серверыв UNIT-2 поддерживаются следующие типы результатов:

2.2.1. строка символов *wchar_t* с завершающим нулём;

2.2.2. число *long*, *float* или *double*;

2.2.3. массив чисел по п. 2.2.2;

2.2.4. экземпляр структуры широкий результат *WideResult* (результат некоторого типа, время типа *double* и качество типа *long*) с результатом типа число по п. 2.2.2;

2.2.5. экземпляр структуры *WideResult* с результатом типа массив чисел по п. 2.2.2;

2.2.6. произвольная структура, первым полем которой является размер этой структуры типа DWORD.

2.3 У сервера для каждого типа результата имеется своя функция записи с именем, которая начинается со слов *WriteResult*. Для отличия функций тип результатов задаётся в имени функций окончанием через символ нижнего подчёркивания (по аналогии с функциями библиотеки IPP 7.0), см. таблицу в конце текста. Второе окончание *arr* (сокращенно от английского *array*) означает, что результат – это массив.

2.4 Результаты могут посылаться серверным приложением пачками с произвольным количеством и типом результатов. Однако для правильной работе пары сервер-клиент лучше этого не делать. Переданные результаты до их чтения хранятся в *UnitCln*. Читать и обрабатывать результаты необходимо в том же порядке, в котором они были записаны.

2.5 При передаче результатов от серверного приложения в клиентское приложение серверная часть посылает сообщение *WM_UNIT2_APP* клиентской части, по получении которого генерируется событие клиентской части *ReadyResult*. Событие должно обрабатываться в классе главного окна клиентского приложения. Это сделано

© ООО "ЭТМС" 1992-2024. Все права защищены

для того, чтобы упростить работу клиента с несколькими серверами. Поэтому клиентская часть должна иметь свой *m_hWnd*, т.е. клиентское приложение должно разрабатываться в среде программирования MS Visual Studio. В сообщении *WM_UNIT2_APP* параметры *wParam* и *lParam* всегда раны 0. Поэтому событие *ReadyResult* не имеет параметров.

2.6 Чтение результатов выполняется клиентским приложением в два этапа. На первом этапе результат с помощью функции *ReadResult* вычитывается в экземпляр структуры *Unit2DataStruct*, имеющей поля для хранения результатов любых типов. Экземпляр этой структуры должен храниться в клиентском приложении. Данный тип структур имеет конструктор, деструктор, переопределённый оператор равенства ("="), а также для каждого типа результата шаблонные функции копирования результатов в структуру (имена функций начинаются на *Copy*) и шаблонные функции извлечения результатов из структуры (имена функций начинаются на *Extract*). На втором этапе по номеру результата определяется его тип, и далее из этой структуры с помощью функции *Extract*, соответствующей данному типу, вычитывается сам результат.

2.7 У клиента есть функция *GetAmountResult*, которая предназначена для получения количества не прочитанных результатов.

2.8 В будущем перечень типов результатов может пополняться. В этом случае в сервер добавляется новая функция *WriteResult* и для структуры *Unit2DataStruct* добавляются функции *Copy* и *Extract* с соответствующим окончанием.

## 2.2.Параметры

**3. Параметры**

Параметры сервера UNIT-2 предназначены для конфигурирования сервера. Параметры имеют номер и тип. Номер параметра – это целое число типа long.

3.1 Параметры разных номеров могут иметь разные типы. Как правило, номер 0 – это задание канала сервера данных, номер 1 – это код временного усреднения. Отрицательные номера параметров используются в многоканальных программах. Номер -1 (минус 1) – это команда на перевод программы в многоканальный режим и задание количества каналов. Номер -2 (минус 2) – это задание индекса текущего канала измерений. Номер -3 (минус 3) – это задание индекса канала, работа с которым останавливается. Номер -4 (минус 4) – это задание индекса канала, работа с которым возобновляется. Номер -5 (минус 5) – это номер, добавляемого канала.

3.2 UNIT-2 поддерживатn следующие типы параметров:

3.2.1. строка символов *wchar_t* с завершающим нулём;

© ООО "ЭТМС" 1992-2024. Все права защищены

3.2.2. число *long*, *float* или *double*.

3.3 Для передачи параметров клиент использует функции *SetParam* с окончаниями, аналогичными окончаниям функций записи результатов.

3.4 Параметры могут посылаться клиентским приложением пачками с произвольным количеством и типом параметров, а также без временной задержки между посылками отдельных параметров. Переданные параметры до их вычитывания хранятся в *UnitSrv*. Читать и обрабатывать параметры необходимо в том же порядке, в котором они были записаны.

3.5 При передаче параметров от клиентского приложения в серверное приложение клиентская часть посылает сообщение *WM_UNIT2_APP* непосредственно главному окну серверного приложения. Т.к. у сервера может быть только один клиент, то создавать событие *Ready* для серверной части нецелесообразно. Это позволяет использовать при разработке серверного приложения любую среду программирования, например Builder. В сообщении WM_UNIT2_APP параметры *wParam* и *lParam* всегда раны 0. Поэтому событие *ReadyParam* не имеет параметров.

3.6 Чтение параметров полностью аналогично чтению результатов. Для чтения параметров используются функции сервера *GetAmountParam* и *GetParam*.

3.7 В будущем перечень типов параметров может пополняться. В этом случае в клиентскую часть добавляется новая функция *SetParam* и для нового типа параметров в структуре *Unit2DataStruct* добавляются функции *Copy* и *Extract* с соответствующим окончанием.

## 2.3.Проверки

**4. Проверки**

4.1 При инициализации клиентской части проверяется загрузка серверного приложения. В случае успешной загрузки клиентская часть генерирует событие *ServerLoad*.

4.2 При записи результатов серверное приложение проверяет клиентское приложение. Если определяется, что клиентского приложения нет, то серверное приложение должно завершить свою работу.

4.3 Каждая из частей UNIT-2 по таймеру, с периодичностью 3 сек, проверяет своего антагониста на существование.

4.4 При определении, что серверного приложения нет, клиентская часть генерирует событие *ServerLost*. При определении зависания – событие *ServerNotActive*.

4.5 Обработка клиентской частью событий *ServerLoad*, *ServerLost*, и *ServerNotActive* не обязательна.

© ООО "ЭТМС" 1992-2024. Все права защищены

## 2.4.Другие функции

**5 Другие функции**

5.1 Клиентская часть имеет функцию *ShowUnit*, предназначенную для изменения режима отображения главного окна серверного приложения (отображать главное окно сервера или нет)

## 2.5.Номера данных многоканальных программ

**6 Номера данных (параметров / результатов) многоканальных программ**

6.1 Номера данных многоканальных программ предлагается строить следующим образом: number = indexChannel * 100 + numberData. Так, например, номер 21707 означает, что это данные № 7 канала с индексом № 217.

## 2.6.Дополнительные разделы ресурсов програмы-сервера

**7 Дополнительные разделы ресурсов програмы-сервера**

7.1. Программа-сервер должна содержать в своих ресурсах дополнительные подразделы в разделе RCDATA. Ниже представлен этот раздел для программы "Вольтметр постоянного тока".

```
//--------------------------------------------------------------------------
    // RCDATA
    //
    UNIT2_PARAM_SIZE_MAX RCDATA
    BEGIN
        4L
    END

    UNIT2_RESULT_SIZE_MAX RCDATA
    BEGIN
```

© ООО "ЭТМС" 1992-2024. Все права защищены

```
        16L
    END
    //---------------------------------------------------------------------------
```

7.2. Подраздел *UNIT2_PARAM_SIZE_MAX* содержит максимальный размер параметров в байтах. Подраздел UNIT2_RESULT_SIZE_MAX содержит максимальный размер результатов в байтах. Эти размеры используются для определения размера FileMapping.

7.3. Наличие описанныъ выше параметров ресурсов позволяет оптимизировать размер используемого FileMapping..

## 2.7.Замечания

**8 Замечания**

8.1 Т.к. технология COM "не знает" структур, реализующих широкие результаты, то самостоятельно создавать файлы UnitCln.h и UnitSrv.h не рекомендуется. Доработанные эти файлы выложены в папку "svn\ZETTools\Dialog_ZET".

8.2. В таблицах 1, 2 и 3 перечислены функции, реализованные в UNIT-2.

### Таблица 1. Функции UnitCln

| Функции инициализации |
|---|
| long Activate(LPCOLESTR name); |
| long DisActivate(); |
| Функции чтения результатов |
| long GetAmountResult(long * pAmount); |
| long ReadResult(long number,); |
| long ReadSizeCurResult(DWORD * pSize); |
| long ReadSizeMaxResult(DWORD * pSize); |
| Функции чтения и записи параметров |
| long GetParam(long number, long number, VARIANT_BOOL paramCln); |
| long ParamIsOk(long number, long *pOk); |
| long GetSizeMaxParam(DWORD *pSize); |
| |

© ООО "ЭТМС" 1992-2024. Все права защищены

| |
|---|
| long SetParam_txt(long number, LPCOLESTR text); |
| long SetParam_32s(long number, long val); |
| long SetParam_32f(long number, float val); |
| long SetParam_64f(long number, double val); |
| Другие функции |
| long ShowUnit(short view); |
| long SetServerSize(long left, long top, long width, long height); |
| long MoveServer(long left, long top); |
| long GetServerLeft(long * pLeft); |
| long GetServerTop(long * pTop); |
| long GetServerWidth(long * pWidth); |
| long GetServerHeight(long * pHeight); |

**Таблица 2. Функции UnitSrv**

| |
|---|
| Функции инициализации |
| long Activate(); |
| long DisActivate(); |
| Функции чтения и записи параметров |
| long GetAmountParam (long* pAmount); |
| long GetParam (Unit2DataStruct * pValue); |
| long GetSizeCurParam(DWORD *pSize); |
| long GetSizeMaxParam(DWORD *pSize); |
| |
| long SetParamBack_32s(long number, long value); |
| long SetParamBack_32f(long number, float value); |
| long SetParamBack_64f(long number, double value)' |
| long SetParamBack_txt(long number, LPCTSTR text); |
| Функции записи результатов |
| long WriteResult_txt(long number, LPCOLESTR text); |
| |
| long WriteResult_32s(long number, long value); |
| long WriteResult_32f(long number, float value); |
| long WriteResult_64f(long number, double value); |
| |
| long WriteResult_32s_arr(long number, std::vector<long> * pValue); |
| long WriteResult_32f_arr(long number, std::vector<float> * pValue); |
| long WriteResult_64f_arr(long number, std::vector<double> * pValue); |
| |
| long WriteResult_wrs_32s (long number, WideResult<long> * pValue); |

© ООО "ЭТМС" 1992-2024. Все права защищены

| |
|---|
| long WriteResult_wrs_32f (long number, WideResult<float> * pValue); |
| long WriteResult_wrs_64f (long number, WideResult<double> * pValue); |
| |
| long WriteResult_wrs_32s_arr (long number, WideResult<std::vector<long>> * pValue); |
| long WriteResult_wrs_32f_arr (long number, WideResult<std::vector<float>> * pValue); |
| long WriteResult_wrs_64f_arr (long number, WideResult<std::vector<double>> * pValue); |
| |
| long WriteResult_str(long number, DWORD size, PVOID pointer); |

**Таблица 3. Функции структур типа Unit2DataStruct**

| Функции копирования данных в структуру |
|---|
| long Copy_txt(const long num, LPCOLESTR text); |
| long Copy_str(const long num, DWORD size, PVOID pointer); |
| |
| long Copy_val(const long num, const T &Val); |
| long Copy_arr(const long num, const std::vector<T> * pVal); |
| long Copy_wrs(const long num, const WideResult<T> * pVal); |
| long Copy_wrs_arr(const long num, const WideResult<std::vector<T>> * pVal); |
| Функции извлечения данных из структуры |
| long Extract_txt(CString &text); |
| long Extract_unit1(double &val); |
| |
| long Extract_val(T &Val); |
| long Extract_arr(std::vector<T> * pVal); |
| long Extract_wrs(WideResult<T> * pVal); |
| long Extract_wrs_arr(WideResult<std::vector<T>> * pVal); |
| |
| long Extract_str(PVOID pointer); |
| Другие функции |
| long TestSize(const DWORD newSize); |
| long GetSizeForExtract_str(DWORD *pSize); |

© ООО "ЭТМС" 1992-2024. Все права защищены

# Глава 3.Описание свойств, методов и событий Unit-2.1 в idl

## 3.1.Функции структуры Unit2DataStruct

| Функции инициализации |
|---|
| Unit2DataStruct(); |
| ~Unit2DataStruct(); |
| long TestSize(const DWORD newSize); |
| long operator = (const Unit2DataStruct &uds); |
| Функции копирования данных в структуру |
| long Copy_txt(const long num, LPCTSTR text); |
| template <class T><br>long Copy_val(const long num, const T &val); |
| template <class T><br>long Copy_arr(const long num, const DWORD size_Array, const T *pArray); |
| template <class T><br>long Copy_wrs_val(const long num, const WideResult<T> *pVal); |
| template <class T><br>long Copy_wrs_arr(const long num, const DWORD size_Array, const WideResult<T*> *pVal); |
| Функции копирования данных из структуры |
| long Extract_txt(CString &str); |
| long Extract_unit1(double &val)); |
| template <class T><br>long Extract_val(T &val); |
| template <class T><br>long Extract_arr(T *pArray); |
| template <class T><br>long Extract_wrs_val(WideResult<T> *pVal); |
| template <class T><br>long Extract_wrs_arr(WideResult<T*> *pVal); |

© ООО "ЭТМС" 1992-2024. Все права защищены

| |
|---|
| long Extract_str(PVOID pointer); |
| long GetSizeForExtract_str(DWORD *pSize); |

## 3.2.Серверная часть

| Функции инициализации |
|---|
| long Activate(); |
| long DisActivate(); |
| Функции чтения параметров |
| long GetAmountParam (long* pAmount); |
| long GetParam (long*  pFullValue); |
| Функции записи результатов |
| long WriteResult_txt(long number, BSTR text); |
| |
| long WriteResult_32s(long number, long value); |
| long WriteResult_32f(long number, float value); |
| long WriteResult_64f(long number, double value); |
| |
| long WriteResult_32s_arr(LONG number, LONG* pValue); |
| long WriteResult_32f_arr(LONG number, LONG* pValue); |
| long WriteResult_64f_arr(LONG number, LONG* pValue); |
| |
| long WriteResult_wrs_32s(LONG number, LONG* pValue); |

© ООО "ЭТМС" 1992-2024. Все права защищены

| |
|---|
| long WriteResult_wrs_32f(LONG number, LONG* pValue);<br>long WriteResult_wrs_64f(LONG number, LONG* pValue); |
| |
| long WriteResult_wrs_32s_arr(LONG number, LONG* pValue |
| long WriteResult_wrs_32f_arr(LONG number, LONG* pValue); |
| long WriteResult_wrs_64f_arr(LONG number, LONG* pValue); |
| |
| long WriteResult_str(LONG number, ULONG size, LONG* pValue) |
| Функции размера данных для установки |
| long GetSizeCurParam(ULONG* pSize); |
| long GetSizeMaxParam(ULONG* pSize); |
| |
| События |
| 1 - получен очередной параметр |

## 3.3.Клиентская часть

| |
|---|
| Функции инициализации |
| long Activate(BSTR nameServer); |
| long DisActivate(void); |
| Функции отображения сервера |
| long ShowUnit(SHORT view); |
| Функции чтения результатов от сервера |
| long GetAmountResult(LONG* pAmount); |
| long ReadResult(LONG* pFullValue); |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Функции записи параметров в сервер |
|---|
| long SetParam_txt(LONG number, BSTR text); |
| long SetParam_32s(LONG number, LONG value); |
| long SetParam_32f(LONG number, FLOAT value); |
| long SetParam_64f(LONG number, DOUBLE value); |
| Функции получения размера результатов сервера |
| long GetSizeCurResult(ULONG* pSize); |
| long GetSizeMaxResult(ULONG* pSize); |
| События |
| 1 - получен очередной результат от сервера |
| 2 - не найден процесс сервера |
| 3 - процесс сервера не отвечает на запросы клиента |
| 4 - успешный запуск программы сервера |

# Глава 4.Переход с UNIT на UNIT-2

Автоматический переход с UNIN на UNIT-2 невозможен, т.к. в отличие от UNIN UNIT-2 состоит из двух частей: серверной и клиентской.

## 4.1.Серверная часть

Следует выполнить следующие действия.

1. Заменить экземпляр класса CDUnit на экземпляр класса CDUnitSrv.
2. В ресурсах сервера, в разделе RC_DATA объявить и соответствующим образом задать значения для UNIT2_PARAM_SIZE_MAX и UNIT2_RESULT_SIZE_MAX.
3. Объявить экземпляр структуры Unit2DataStruct для получения от клиента параметров (например с именем m_paramUnit2).
4. Функцию-обработчик события 1 (OnReady) UNIT заменить на функцию-обработчик сообщения WM_UNIT2_APP (получение параметра от клиента), например: с именем OnReadyUnit2Param.
5. В функции OnReadyUnit2Param:

- с помощью функции CDUnitSrv::GetParam в структуру m_paramUnit2 скопировать данные с полученным параметром;
- в соответствии с номером параметра проверить его тип;

© ООО "ЭТМС" 1992-2024. Все права защищены

- в случае правильного типа с помощью функции Unit2DataStruct::Extract, соответствующей данному типу параметра, вычитать из структуры m_paramUnit2 значение параметра;

- применить значение полученного параметра.

6. Для передачи клиенту результата вместо функции CDUnit::UnitWrute воспользоваться функцией CDUnitSrv::WriteResult с окончанием, соответствующим типу передаваемого результата.

## 4.2.Клиентская часть

Следует выполнить следующие действия.

1. Заменить экземпляр класса CDUnit на экземпляр класса CDUnitCln.
2. Объявить экземпляп структуры Unit2DataStruct для получения от сервера результатов (например с именем m_resultUnit2).
3. Функцию-обработчик события 1 (OnReady) UNIT заменить на функцию-обработчик события 1 UNIT-2 (получение результата от сервера), например: с именем OnReadyResult.
4. Задать функцию-обработчик события 2 UNIT-2 (не найден процесс сервера).
5. Задать функцию-обработчик события 3 UNIT-2 (процесс сервера не отвечает на запросы клиента).
6. Задать функцию-обработчик события 4 UNIT-2 (успешный запуск программы сервера).
7. В функции OnReadyResult:

   - с помощью функции CDUnitCln::GetSizeCurResult получить размер результата;

   - с помощью функции Unit2DataStruct::TestSize передать в структуру m_resultUnit2 полученный размер результата (данный шаг очень важен, т.к. по умолчанию экземпляры струткуры Unit2DataStruct создаётся с нулевым размером!!!);

   - с помощью функции CDUnitCln::ReadResult скопировать результат в структуру m_resultUnit2;

   - с помощью функции Unit2DataStruct::Extract, соответствующей типу полученного результата., скопировать из m_resultUnit2 данные этого результата;

   - использовать данные полученногно результата для их отображения и/или для выполнения необходимых расчётов.

8. Для передачи серверу параметра вместо функции CDUnit::SetParamTimeString воспользоваться функцией CDUnitSrv::SetParam с окончанием, соответствующим типу передаваемого параметра.

© ООО "ЭТМС" 1992-2024. Все права защищены

# Unit.ocx Предназначен для связи с программами ZETLab

**Назначение**

Модуль предназначен для работы с устройствами АЦП-ЦАП, для обработки сигналов в реальном времени и обработки оцифрованных сигналов, записанных в файлы. Модуль обеспечивает обмен данными и потоками данных с драйверами, и другими пользовательскими программами по COM-интерфейсу с использованием технологии клиент-сервер. Модуль поддерживает одновременную работу с несколькими платами АЦП-ЦАП различного или одинакового типа. Суммарное количество каналов не более 200. Суммарное количество работающих устройств в системе не более 50. Оцифрованные данные в программу пользователя передаются в формате плавающей запятой одинарной точности, в заданных единицах измерения. Различные модули АЦП преобразуют аналоговый сигнал в 12, 14, 16 или 24 разрядный код. Компонент *SRV* преобразует эти целочисленные значения в формат плавающей запятой с учетом коэффициентов усиления, чувствительности преобразователей, смещения постоянной составляющей, поправками по измерительным каналам. Эти параметры задаются в программе «Диспетчер устройств» из группы «Сервисные».

## Глава 1.Установка компонента

Для работы с модулем *Unit* его необходимо сначала установить в программу. При загрузке программы необходимо загрузить с помощью компонента необходимую программу, разместить окно программы на экране в удобном месте, установить необходимые параметры в программе и затем по событиям, происходящим от программы считывать данные из программы.

Для установки компонента *Unit* в проект *Microsoft Visual Studio 2010*, необходимо кликнуть правой кнопкой мыши на форме диалога и из появившегося меню выбрить пункт *Insert ActiveX Control*...(рисунок 1.1)

© ООО "ЭТМС" 1992-2024. Все права защищены

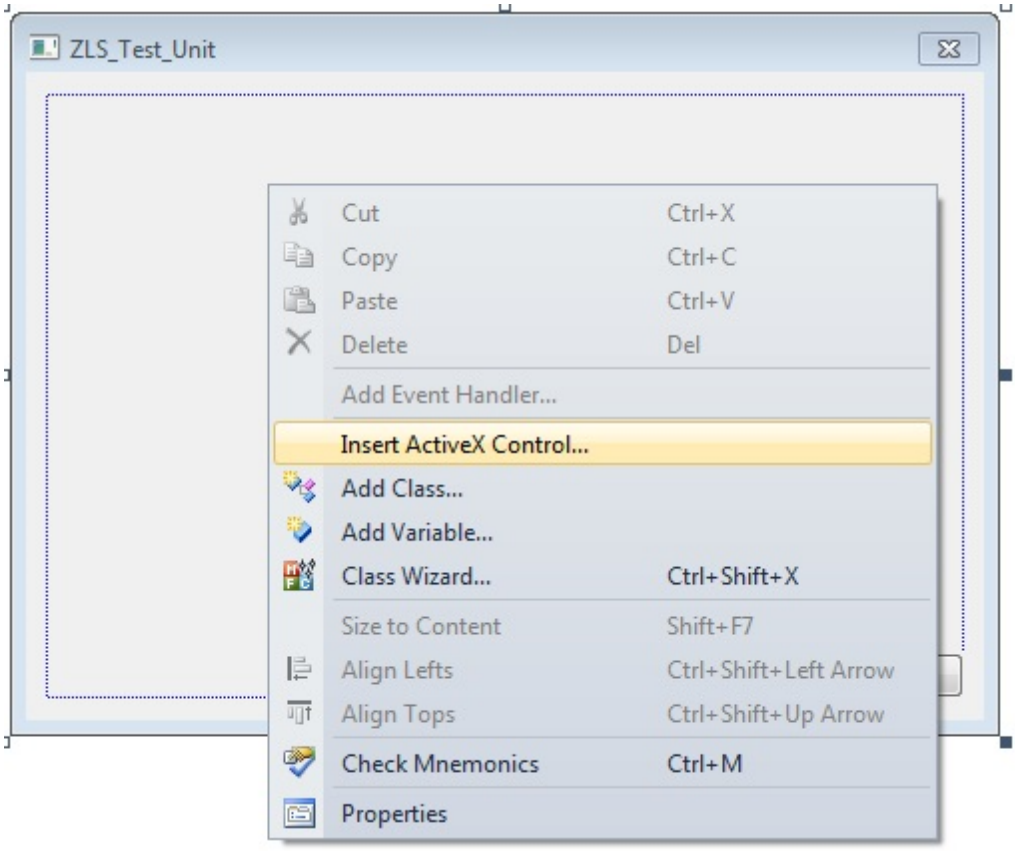



Рисунок 1.1

Из появившегося диалогового окна следует выбрать компонент *Unit Control* и нажать *OK* (рисунок 1.2).

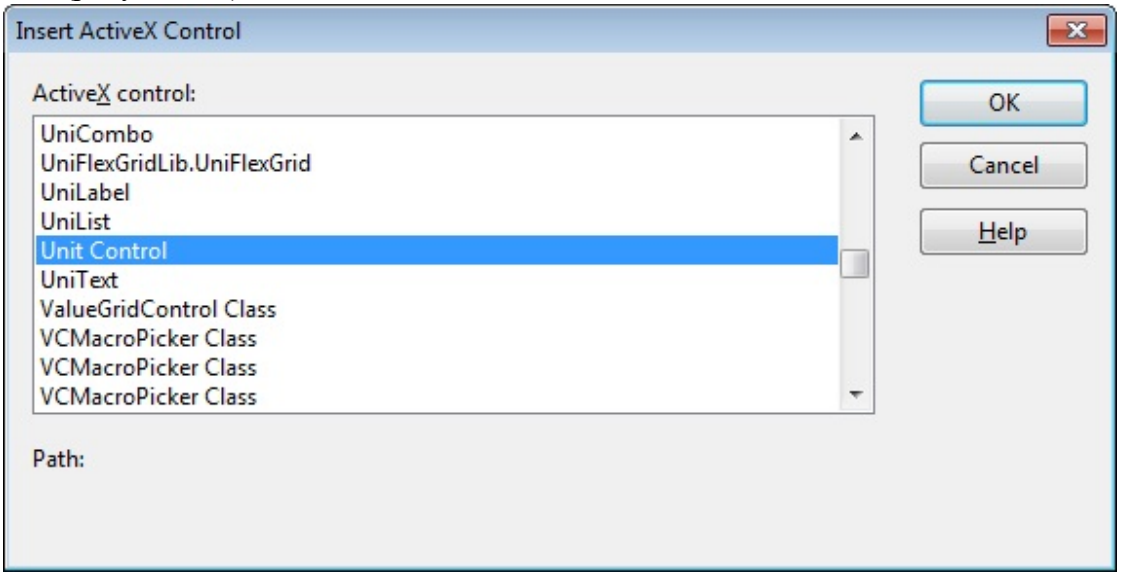



Рисунок 1.2

После этого компонент *Unit.ocx* появится на форме диалога (рисунок 1.3).

© ООО "ЭТМС" 1992-2024. Все права защищены

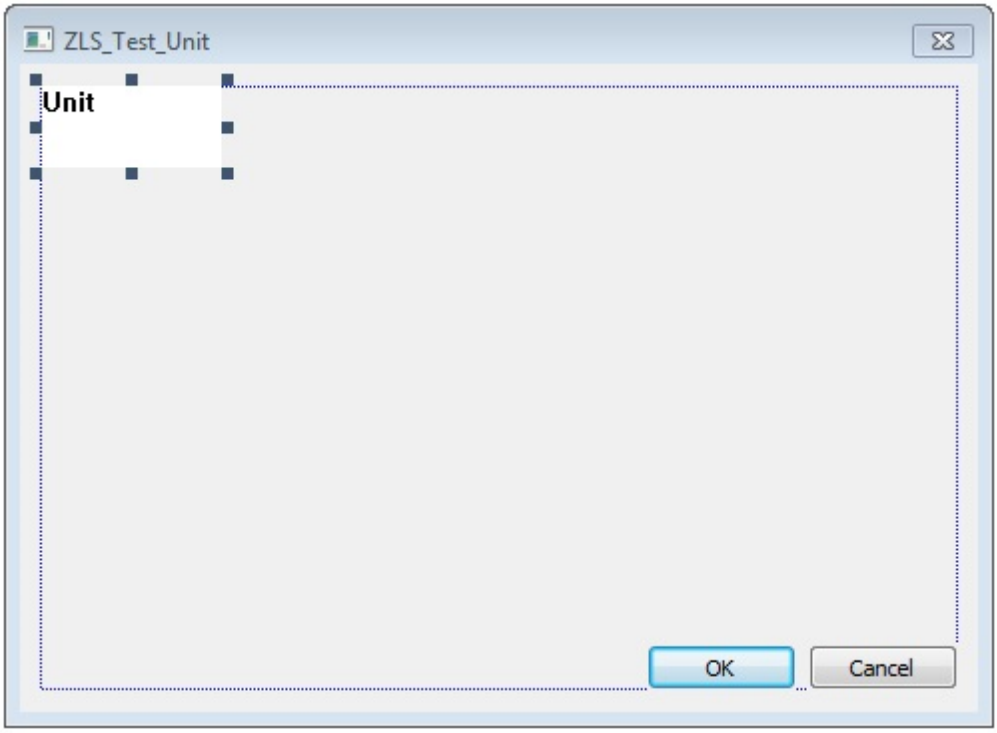



Рисунок 1.3

Одна установленная на форме компонента позволяет управлять одной программой. При необходимости управлять несколькими программами, необходимо установить на форму несколько компонент. Для того чтобы компонента не была видна на форме во время выполнения программы, необходимо установить свойство *Visible* равным *False*.

# Глава 2.Изменения в zetlab studio на 30.06.2015

Программное обеспечение ZetLab регулярно обновляется с целью улучшения его функциональности. При этом некоторые невостребованные возможности из него исключаются.

В последней версии Unit.ocx функция ReadNet не используется. Вместо неё предполагается использовать следующие функции:

*long Read(long *size, float* data, long* param);*
*long UnitRead(long *size, float* data, long param);*

Эти функции отличаются последним параметром. Функция *UnitRead* вычитывает массив чисел, у которого параметр равен заданному. Функция *Read* вычитывает не только массив чисел, но и его параметр. Если размер передаваемого массива не известен, то его можно узнать вызовом функции с *data = NULL*.

© ООО "ЭТМС" 1992-2024. Все права защищены

Пример использования функции *Read*.

```
//-----------------------------------------------------
long size,  param, ret;
float *data = NULL;
// чтение параметра и размера массива
ret = Read(&size, NULL, &param);
if (ret == 0)
{
        if (param == 4)
        {
                data = new float[size];
                ret = Read(&size, data, &param);
                if (ret == 0)
                {
                        // использование считанных данных
                }
                else
                {
                        // ошибка чтения
                }
        }
}
else
{
        // ошибка чтения
}
.....
if (data)
        delete [] data;
//-----------------------------------------------------
```

Пример использования функции *UnitRead*.

```
//-----------------------------------------------------
long size,  ret;
float *data = NULL;
// чтение размера массива
ret = UnitRead(&size, NULL, 4);
if (ret == 0)
{
        data = new float[size];
        ret = UnitRead(&size, data, 4);
        if (ret == 0)
        {
                // использование считанных данных
        }
```

© ООО "ЭТМС" 1992-2024. Все права защищены

```
        else
        {
                // ошибка чтения
        }
}
else if (ret == -65)
{
        // массив имеет параметр, отличный от 4
}
else
{
        // ошибка чтения
}
.....
if (data)
        delete [] data;
//-------------------------------------------------------
```

С уважением, коллектив ZetLab.

P.S. Компания ZetLab прекратила поддержку программ, написанных на Vbasic.

# Глава 3.Описание методов и событий в idl

Запуск программ при помощи интерфейсного программного модуля *Unit* представляет собой технологию клиент-сервер, где серверной частью является запускаемое и управляемое приложение, а клиентской – приложение, которое запускает заданную программу и управляет ей, получая от нее определенные данные. При этом методы и события компонента делятся на те, которые используются для настройки клиентской части, и те, которые используются для настройки серверной части.

## 3.1.Серверная часть

1. При инициализации клиентской части проверяется загрузка серверного приложения. В случае успешной загрузки клиентская часть генерирует событие *ServerLoad*.

2 При записи результатов серверное приложение проверяет клиентское приложение. Если определяется, что клиентского приложения нет, или оно зависло, то серверное приложение должно завершить свою работу.

3 Клиентское приложение по таймеру, с периодичностью 3 сек, проверяет серверное приложение.

© ООО "ЭТМС" 1992-2024. Все права защищены

4 При определении, что серверного приложения нет, клиентская часть генерирует событие *ServerLost*. При определении зависания – событие *ServerNotActive*.

5 Обработка клиентской частью событий *ServerLoad*, *ServerLost*, и *ServerNotActive* не обязательна.

## Методы

***UnitReg*** – установление режима работы программы с возможностью управления ею через программный модуль *Unit* (вызывается при старте программы).

*long* ***UnitReg****(long hw)*

Параметры

*hw* – хэндлер окна приложения.

Возвращаемое значение

*0* – нормальное выполнение функции.

*-1* – хэндлер окна приложения равен *NULL* или не было запуска программы через компонент *Unit*.

*-2* – не хватает хэндлеров памяти.

*-3* – не хватает хэндлеров памяти.

***UnitUnReg*** – прекращение режима работы программы с возможностью управления ею через программный модуль *Unit* (вызывается при завершении программы).

*long* ***UnitUnReg****(long hw)*

Параметры

*hw* – хэндлер окна приложения.

Возвращаемое значение

*0* – нормальное выполнение функции.

*-1* – не было запуска программы через компонент *Unit*.

*-2* – хэндлер окна приложения равен *NULL*.

***UnitParam*** – считывание параметра, присланного в программу через компонент Unit запускающим приложением.

*long* ***UnitParam*** *(long* param, double* value)*

Параметры

*param* – возвращает номер устанавливаемого параметра.

*value* – возвращает значение устанавливаемого параметра.

Возвращаемое значение

*0* – нормальное выполнение функции.

*-1* – не было подключения к программе через компонент *Unit*.

*-2* – хэндлер окна приложения равен *NULL*.

© ООО "ЭТМС" 1992-2024. Все права защищены

*-3* – не хватает хэндлеров памяти.
*-4* – не хватает хэндлеров памяти.
*-5* – место в памяти для возврата номера устанавливаемого параметра равно *NULL*.
*-6* – место в памяти для возврата значения устанавливаемого параметра равно *NULL*.

***UnitReadString*** – считывание строкового параметра, присланного в программу через компонент Unit запускающим приложением.

*long* ***UnitReadString*** *(long* param, char* StringVal)*

Параметры
*param* – возвращает номер устанавливаемого параметра.

*value* – возвращает значение устанавливаемого строкового параметра.

Возвращаемое значение
*0* – нормальное выполнение функции.
*-1* – не было подключения к программе через компонент *Unit*.
*-2* – хэндлер окна приложения равен *NULL*.
*-3* – не хватает хэндлеров памяти.
*-4* – не хватает хэндлеров памяти.
*-5* – место в памяти для возврата номера устанавливаемого параметра равно *NULL*.
*-6* – место в памяти для возврата значения устанавливаемого строкового параметра равно *NULL*.

***UnitReadTimeString*** – считывание строкового параметра, присланного в программу через компонент Unit запускающим приложением с меткой синхронизации по времени *ZETServer*.

*long* ***UnitReadTimeString*** *(long* param, BSTR* StringVal, DOUBLE* timer)*

Параметры
*param* – возвращает номер устанавливаемого параметра.

*value* – возвращает значение устанавливаемого строкового параметра.

*time* – возвращает значение времени ZETServer.

Возвращаемое значение
*0* – нормальное выполнение функции.
*-1* – не было подключения к программе через компонент *Unit*.
*-2* – хэндлер окна приложения равен *NULL*.
*-3* – не хватает хэндлеров памяти.
*-4* – не хватает хэндлеров памяти.
*-5* – место в памяти для возврата номера устанавливаемого параметра равно *NULL*.

© ООО "ЭТМС" 1992-2024. Все права защищены

*-6* – место в памяти для возврата значения устанавливаемого строкового параметра равно *NULL*.

*-7* – превышена максимальная допустимая длина считываемого строкового параметра.

***UnitWrite*** – передача данных в запущенную программу.

*long* ***UnitWrite****(long size, float* data, long param)*

Параметры

*size* – размер передаваемого массива данных.

*data* – указатель на передаваемый массив данных.

*param* – параметр.

Возвращаемое значение

*0* – нормальное выполнение функции.

*-1* – не было подключения к программе через компонент *Unit*.

*-2* – не было подключения к программе через компонент *Unit*.

*-3* – приложение, которое запустило данную программу через компонент *Unit*, было выгружено из памяти.

*-4* – не хватает хэндлеров памяти.

*-5* – не хватает хэндлеров памяти.

***IsReadyParam*** – запрос на наличие данных от клиента для запущенной через компонент *Unit* программы (сервера).. Функция запоминает количество сообщений, передаваемых от клиента к приложению и возвращает “1”, декрементирую счетчик сообщений до тех пор, пока он не обнулится.. .

*long* ***IsReadyParam*** *(void)*

Возвращаемое значение

*0* – нет доступных данных.

*1* – есть доступные данные.

## События

***Ready*** – возникает при установке параметров программы через функции *SetParam(...)*, *SetParamString(...)* и *SetParamTimeString(...)* через программный модуль *Unit*. Данное событие генерируется только тогда, когда есть идентификатор окна, т.е. при написании програм на тех языках программирования, где отсутствуют идентификаторы окна, событие генерироваться не будет.

*void* ***Ready*** *(long par)*

Параметры

par – параметр.

© ООО "ЭТМС" 1992-2024. Все права защищены

***Load*** – Данное событие сразу после завершения загрузки программs.

*void* ***Load*** *()*

***NotActive*** – Данное событие генерируется в случае, когда запущенная программа прекращает сообщать в UNIT о своей деятельности. См. метод ***IamActive*** клиентской части.

*void* ***NotActive*** *()*

***Lost*** – Данное событие генерируется в случае, когда запущенная программа завершает свою работу не по команде от UNIT.

*void* ***Lost*** *()*

## 3.2.Клиентская часть

1. При инициализации клиентской части проверяется загрузка серверного приложения. В случае успешной загрузки клиентская часть генерирует событие *ServerLoad*.

2 При записи результатов серверное приложение проверяет клиентское приложение. Если определяется, что клиентского приложения нет, или оно зависло, то серверное приложение должно завершить свою работу.

3 Клиентское приложение по таймеру, с периодичностью 3 сек, проверяет серверное приложение.

4 При определении, что серверного приложения нет, клиентская часть генерирует событие *ServerLost*. При определении зависания – событие *ServerNotActive*.

5 Обработка клиентской частью событий *ServerLoad*, *ServerLost*, и *ServerNotActive* не обязательна.

### Методы

***Activate*** – загрузка программы для управления и подключение к ней.

*long* ***Activate****(BSTR name)*

Параметры

*name* – имя программы без расширения.

Возвращаемое значение

*0* – нормальное выполнение функции.

*-1* – нет свободного места для подключения к компоненту *Unit* (слишком много программ подключено к компоненту *Unit*).

*-2* – не хватает хэндлеров памяти.

*-3* – не хватает хэндлеров памяти.

© ООО "ЭТМС" 1992-2024. Все права защищены

*-4* – не запускается программа *name* (нет на диске, не та версия и т.д.).

*-5* – программа *name* есть, но она не проходит инициализацию с компонентом *Unit* (мало оперативной памяти или дискового пространства).

*-6* – не хватает хэндлеров памяти.

*-7* – клиент не обнаружил окно запущенной программы-сервер.

***DisActivate*** – закрытие и выгрузка запущенной программы.

*long* ***DisActivate****(void)*

Возвращаемое значение

*0* – нормальное выполнение функции.

*-1* – не было подключения к программе через компонент *Unit*.

***Read*** – чтение данных из подключенной программы.

*long* ***Read****(long* size, float* data, long* param)*

Параметры

*size* – возвращает размер переданного массива данных.

*data* – адрес массива, куда будут записаны требуемые данные.

*param* – возвращает параметр. Для каждой программы существует свой набор параметров, который определяет возвращаемые данные. Значения параметров для каждой программы представлены в соответствующем разделе.

Возвращаемое значение

*0* – нормальное выполнение функции.

*-1* – не было подключения к программе через компонент *Unit*.

*-2* – хэндлера окна запускаемого приложения не существует.

*-3* – не хватает хэндлеров памяти.

*-4* – не хватает хэндлеров памяти.

***UnitRead*** – чтение данных из подключенной программы по заданному параметру.

*long* **Unit*****Read****(long* size, float* data, long param)*

Параметры

*size* – возвращает размер переданного массива данных.

*data* – адрес массива, куда будут записаны требуемые данные.

*param* – параметр для чтения данных.

Возвращаемое значение

*0* – нормальное выполнение функции.

*-1* – не было подключения к программе через компонент *Unit*.

*-2* – хэндлера окна запускаемого приложения не существует.

*-3* – не хватает хэндлеров памяти.

*-4* – не хватает хэндлеров памяти.

*-7* – по запрашиваемому параметру нет данных для чтения.

© ООО "ЭТМС" 1992-2024. Все права защищены

***SetParam*** – установка параметра запущенной программы. Для каждой программы существует свой набор параметров управления. Значения параметров для каждой программы представлены в соответствующем разделе.

*long* ***SetParam****(long param, double value)*

Параметры

*param* – номер устанавливаемого параметра.
*value* – значение устанавливаемого параметра.

Возвращаемое значение

*0* – нормальное выполнение функции.
*-1* – не было подключения к программе через компонент *Unit*.
*-2* – запущенная через компонент *Unit* программа выгружена из памяти.
*-3* – не хватает хэндлеров памяти.
*-4* – не хватает хэндлеров памяти.

***SetParamString*** – установка строкового параметра запущенной программы. Для каждой программы существует свой набор параметров управления. Значения параметров для каждой программы представлены в соответствующем разделе.

*long* ***SetParamString*** *(long param, char* StringVal)*

Параметры

*param* – номер устанавливаемого параметра.
*value* – значение устанавливаемого строкового параметра.

Возвращаемое значение

*0* – нормальное выполнение функции.
*-1* – не было подключения к программе через компонент *Unit*.
*-2* – запущенная через компонент *Unit* программа выгружена из памяти.
*-3* – не хватает хэндлеров памяти.
*-4* – не хватает хэндлеров памяти.
*-5* – превышена максимальная допустимая длина строкового параметра.

***SetParamTimeString*** – установка строкового параметра запущенной программы с синхронизацией по времени ZETServer. Для каждой программы существует свой набор параметров управления. Значения параметров для каждой программы представлены в соответствующем разделе.

*long* ***SetParamTimeString*** *(long param, BSTR StringVal, DOUBLE timer)*

Параметры

*param* – номер устанавливаемого параметра.
*value* – значение устанавливаемого строкового параметра.
*time* – значение времени ZETServer.

Возвращаемое значение

*0* – нормальное выполнение функции.

© ООО "ЭТМС" 1992-2024. Все права защищены

*-1* – не было подключения к программе через компонент *Unit*.
*-2* – запущенная через компонент *Unit* программа выгружена из памяти.
*-3* – не хватает хэндлеров памяти.
*-4* – не хватает хэндлеров памяти.
*-5* – превышена максимальная допустимая длина строкового параметра.

***ShowUnit*** – показать или спрятать запущенную через компонент *Unit* программу.

*long* ***ShowUnit****(short view)*

Параметры
*view* – параметр видимости программы. *0* – спрятать программу, *1* – показать программу.

Возвращаемое значение
*0* – нормальное выполнение функции.
*-1* – не было подключения к программе через компонент *Unit*.
*-2* – запущенная через компонент *Unit* программа выгружена из памяти.
*-3* – не хватает хэндлеров памяти.
*-4* – не хватает хэндлеров памяти.

***SetUSize*** – изменение размеров области видимости запущенной через компонент *Unit* программы.

*long* ***SetUSize****(double left, double top, double width, double height)*

Параметры
*left* – левая граница области видимости.
*top* – верхняя граница области видимости.
*width* – ширина области видимости.
*height* – высота области видимости.

Возвращаемое значение
*0* – нормальное выполнение функции.
*-1* – не было подключения к программе через компонент *Unit*.
*-2* – запущенная через компонент *Unit* программа выгружена из памяти.
*-3* – не хватает хэндлеров памяти.
*-4* – не хватает хэндлеров памяти.

***MoveUnit*** – перемещение окна запущенной через компонент *Unit* программы.

*long* ***MoveUnit*** *(double x, double y)*

Параметры
*x* – абсцисса точки перемещения.
y – ордината точки перемещения.

Возвращаемое значение
*0* – нормальное выполнение функции.

© ООО "ЭТМС" 1992-2024. Все права защищены

*-1* – не было подключения к программе через компонент *Unit*.
*-2* – запущенная через компонент *Unit* программа выгружена из памяти.
*-3* – не хватает хэндлеров памяти.
*-4* – не хватает хэндлеров памяти.

***GetUnitWidth***, ***GetUnitHeight***, ***GetUnitLeft***, ***GetUnitTop*** – узнать ширину, высоту, левую границу и верхнюю границу окна запущенной через компонент *Unit* программы.

*double* ***GetUnitWidth****(void)*
*double* ***GetUnitHeight*** *(void)*
*double* ***GetUnitLeft*** *(void)*
*double* ***GetUnitTop*** *(void)*

Возвращаемое значение
*≥ 0* – соответствующее значение параметров окна программы.
*-1* – не было подключения к программе через компонент *Unit*.
*-2* – запущенная через компонент *Unit* программа выгружена из памяти.
*-3* – не хватает хэндлеров памяти.
*-4* – не хватает хэндлеров памяти.

***IsReady*** – запрос на наличие данных от запущенной через компонент *Unit* программы.

*long* ***IsReady*** *(void)*

Возвращаемое значение
*0* – нет доступных данных.
*1* – есть доступные данные.

## События

***Ready*** – возникает при обновлении передаваемых данных из запущенной через компонент *Unit* программы через функции *Write(...)*, *WriteNet(...)* и *UnitWrite(...)* через программный модуль *Unit*. Например, если запустить через компонент *Unit* какую-либо программу, которая рассчитывает различные значения и передает их по интерфейсу *Unit*, то в обработчике данного события можно организовать чтение данных с помощью метода *Read(...)*

*void* ***Ready*** *(long par)*

Параметры
par – параметр, означающий готовность данных определенного типа. Для каждой программы существует свой набор параметров, который представлен в соответствующем разделе.

© ООО "ЭТМС" 1992-2024. Все права защищены

***Load*** – Данное событие сразу после завершения загрузки программs.

*void* ***Load*** *()*

***NotActive*** – Данное событие генерируется в случае, когда запущенная программа прекращает сообщать в UNIT о своей деятельности. См. метод ***IamActive*** клиентской части.

*void* ***NotActive*** *()*

***Lost*** – Данное событие генерируется в случае, когда запущенная программа завершает свою работу не по команде от UNIT.

*void* ***Lost*** *()*

# Глава 4.Перечень общих параметров многоканальных/многоконтейнерных программ

| Номер параметра | Параметр | Диапазон значений параметра | Примечание |
|---|---|---|---|
| -1 | Задание кол-ва каналов | От 1 до Nsrv* | |
| -2 | Задание индекса текущего канала | От 0 до (N-1)** | |
| -3 | Задание индекса останавливаемого канала | От 0 до (N-1)** | |
| -4 | Задание индекса запускаемого канала | От 0 до (N-1)** | |
| -5 | Добавление канала | От -1 до (Nsrv - 1) *** | |
| -6 | Начало передачи серии параметров текущего контейнера | Любой | |
| -7 | Конец передачи серии параметров текущего контейнера | Любой | |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit | Тип параметра - tdu_str | Только UNIT-2 |
| -9 | Запрос на передачу UNIT-клиенту структуры параметров ParamOneKit текущего контейнера | Любой | Только UNIT-2 |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | | |
|---|---|---|
| -10 | Остановка работы программы | Любой |
| -11 | Запуск работы программы | Любой |

* Nsrv - кол-во каналов ZetServer.
** N - кол-во каналов программы.
*** для контейнеров, содержащих один канал, иначе любой.

# Глава 5.Установка параметров программ ZETLab

Для установки параметров программ используются методы *SetParam(...)*, *SetParamString(...)* и *SetParamTimeString(...)*, где аргумент *param* отвечает за установку того или иного значения управляющего параметра. Ниже приведены значения аргумента *param* для настройки приложений при работе через компонент *Unit*.

## 5.1.меню Анализ сигналов

Содержит перечень программ из вкладки меню Анализ сигналов из ZETLab, которыми можно управлять через Unit или Unit-2.

### Узкополосный спектр (spectr.exe)

**Навигация:** HELP ZETLab studio. Руководство разработчика.249 > Unit.ocx Предназначен для связи с программами ZETLab249 > Установка параметров программ ZETLab249 > меню Анализ сигналов249 >

# Узкополосный спектр (spectr.exe)

# Unit-1 и Unit-2

Nsrv - количество каналов ZETServer.
Nctn - количество контейнеров программы.
Fadc – частота дискретизации канала в Гц.
d1 – дистанция датчика канала 1, м.
d2 – дистанция датчика канала 2, м.
dt – межканальная задержка, мс.

Контейнер программы «Узкополосный спектр» - это пара каналов ZETServer. Если хотя-бы один канал из этой пары не задан (номер канала равен -1 или нулевой GUID), то контейнер считается не заданным, и работа с ним не выполняется.

© ООО "ЭТМС" 1992-2024. Все права защищены

Программа "Узкополосный спектр" предназначена для расчёта и отображения результатов спектральной обработки сигналов. Для расчётов программа использует алгоритм быстрого Фурье преобразования (БПФ) или дискретного преобразования Фурье (ДПФ). Для выбора типа преобразования в окне параметров программы во втором столбце элементов слева есть выпадающий список "Тип обработки".

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| | |
| 0 | Установка порядкового номера канала (от 0 до (количество каналов - 1)) . Задание номера текущего канала (-1 допустим). |
| 0 | Используется только для Unit-2. Установка текущего канала по его GUID. |
| 1 | Установка декады (частотного диапазона анализа) Гц:<br>0 – от 0 до (частота дискретизации / 2)<br>1 – от 0 до (частота дискретизации / 20)<br>2 – от 0 до (частота дискретизации / 200)<br>3 – от 0 до (частота дискретизации / 2000)<br>4 – от 0 до (частота дискретизации / 20000)<br>По умолчанию при запуске под UNIT программа будет использовать 10000.частотный диапазон. |
| 3 | Установка времени усреднения (от 0.01 секунды до 1000 секунд)<br>Данный параметр предназначен для задания времени усреднения в сек. При иных значениях аргумента параметр игнорируется.<br><br>По умолчанию при запуске под UNIT программа будет использовать 1.время усреднения. |
| 4 | Установка типа представления расчета спектра:<br>0 – спектральная плотность<br>1 – спектральная мощность<br>2 – среднее квадратичное значение<br>3 – амплитудное (пиковое) значение<br>По умолчанию при запуске под UNIT программа будет использовать среднее квадратичное значение. |
| 8 | Установка кода задания частотного разрешения при использовании БПФ:<br>df = freqADC / 2^(7 + code),<br>где:<br>df - растотное разрешение в Гц;<br>code - код от 0 до 12.<br>По умолчанию используется значение code = 3.<br>Так при freqADC = 25000 Гц значение code = 0 соответствует df = 195,3 Гц; code = 3 соответствует df = 24,4 Гц; code = 12 соответствует df = 0,048 Гц. |
| 9 | Установка очистки спектра медианным фильтром:<br>0 – выключение медианного фильтра<br>1 – включение медианного фильтра<br>По умолчанию при запуске под UNIT программа будет без медианного фильтра.. |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| 0 | Установка порядкового номера канала (от 0 до (количество каналов - 1)) . Задание номера текущего канала (-1 допустим). |
| 10 | Установка типа анализа:<br>0 – быстрое преобразование Фурье (БПФ)<br>1 – дискретное преобразование Фурье (ДПФ)<br>По умолчанию при запуске под UNIT программа будет использовать ДПФ. |
| 11 | Установка типа весовой функции:<br>0 – прямоугольная<br>1 – весовая функция Ханна<br>2 – весовая функция Хэннинга<br>3 - весовая функция Хэмминга<br>4 – весовая функция Блэкмана<br>5 – весовая функция Барлета<br>6 – весовая функция Кайзера<br>7 - весовая функция Рифа-Винсента (4)<br>8 - весовая функция Блэкмана-Харриса (3)<br>9 - весовая функция Блэкмана-Харриса (4)<br>10 -весовая функция Наталла<br>11 -весовая функция Блэкмана-Наталла<br>12 -весовая функция с плоской вершиной<br>По умолчанию при запуске под UNIT программа будет использовать Ханна (Хэмминга). |
| 13 | Установка типа обработки сигнала:<br>0 – дифференцирование второго порядка<br>1 – дифференцирование первого порядка<br>2 – без обработки интегрирования и дифференцирования<br>3 – интегрирование первого порядка<br>4 – интегрирование второго порядка<br>По умолчанию при запуске под UNIT программа будет использовать без обработки . |
| 14 | Установка типа представления уровня спектральных компонент (расчёт значений по Y):<br>0 – линейный масштаб (в единицах измерения)<br>1 – логарифмический масштаб (в децибелах)<br>По умолчанию при запуске под UNIT программа будет использовать линейное. |
| 19 | Установка передачи 4 массивов -> частоты, спектры Cur, Max, Aver) |
| 20 | Установка на расчет максимального спектра:<br>0 – запрет расчета максимального спектра<br>1 – разрешение расчета максимального спектра |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| 0 | Установка порядкового номера канала (от 0 до (количество каналов - 1)) . Задание номера текущего канала (-1 допустим). |
| | По умолчанию при запуске под UNIT программа без расчета максимального спектра |
| 21 | Установка на расчет среднего спектра:<br>0 – запрет расчета среднего спектра<br>1 – разрешение расчета среднего спектра<br>По умолчанию при запуске под UNIT программа без расчета среднего спектра |
| 22 | Установка интервала расчета дополнительных спектров (от 6 до 600)<br>По умолчанию при запуске под UNIT программа будет использовать интервал расчета 10 |
| 23 | Установка на расчет минимального спектра:<br>0 – запрет расчета минимального спектра<br>1 – разрешение расчета минимального спектра<br>По умолчанию при запуске под UNIT программа без расчета минимального спектра |
| 24 | Установка нижней границы отображения графиков по Y |
| 25 | Установка верхней границы отображения графиков по Y |
| 26 | Посылает результаты вычисления спектра в вида одного массива собранного из четырёх:<br>1. массив частот;<br>2. массив значений мгновенного спектра;<br>3. массив значений максимального спектра;<br>4. массив значений усреднённого спектра. |
| 27 | Установка частотного разрешения при ДПФ в Гц, значение которого должно входить в интервал от freqADC/100000 до freqADC/100, где freqADC - частота дискретизации в Гц. По умолчанию задаётся значение 1 Гц. |
| 28 | Установка типа усреднения (0 - линейное, 1 - экспоненциальное).<br>По умолчанию при запуске под UNIT программа будет использовать линейное. |
| 29 | Установка количества полос при FFT<br>По умолчанию при запуске под UNIT программа будет использовать частотное разрешение 24.41 |
| 30 | Установка количества полос при DFT<br>По умолчанию при запуске под UNIT программа будет использовать частотное разрешение 1 |
| 31 | Установка типа оси X на графике (0 - линейное, 1 - логарифм.) |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| 0 | Установка порядкового номера канала (от 0 до (количество каналов - 1)) . Задание номера текущего канала (-1 допустим). |
| 32 | Установка типа оси Y на графике (0 - линейное, 1 - логарифм.) |
| 33 | Установка имени файла нормы. Имя файла устанавливается предварительно функцией SetParamString(long param, signed char *value). Для правильного отображения файла необходимо устанавливать расширение *.nrm. Unit-параметр 33 (задание имени файла нормы) - теперь и запускает отображение графика нормы. |
| 34 | Команда графику выполнить автомасштабирование |
| 35 | Установка длины медианного фильтра |
| 36 | Установка фильтрации пост. сост. (да / нет)<br>По умолчанию при запуске под UNIT программа постоянной. составляющей не используется |
| 37 | Установка режима синхронизации (да / нет)<br>По умолчанию при запуске под UNIT программа постоянной. составляющей не используется |
| 38 | Установка режима синхронизации по датчику оборотов (да / нет)<br>По умолчанию при запуске под UNIT программа постоянной. составляющей не используется |
| 39 | Установка номера канала режима синхронизации |
| 39 | Используется только для Unit-2. Задание GUID канала режима синхронизации |
| 40 | Установка типа режима синхронизации (не СДО) |
| 41 | Установка порогового значения режима синхронизации, ед. канала synchro) |
| 42 | Установка временной задержки режима синхронизации по датчику оборотов, мсек |
| 43 | Установка на задание флага "Расчёт резонансов"<br>По умолчанию при запуске под UNIT программа постоянной. составляющей не используется |
| 44 | Установка на задание флага "Режим слежения"<br>По умолчанию при запуске под UNIT программа постоянной. составляющей не используется |
| 45 | Установка на задание частоты 1 при расчёте резонансов |
| 46 | Установка на задание частоты 2 при расчёте резонансов |
| 1000 | Подача команды на сохранение текущего графика в dtx-файл. Параметр типа tdu_txt, в котором записано имя файла для графика. |

© ООО "ЭТМС" 1992-2024. Все права защищены

## Долеоктавный спектр (dspectr.exe)

**Навигация:** HELP ZETLab studio. Руководство разработчика.255 > Unit.ocx Предназначен для связи с программами ZETLab255 > Установка параметров программ ZETLab255 > меню Анализ сигналов255 >

# Долеоктавный спектр (dspectr.exe)

# Unit-1 и Unit-2

Nsrv - количество каналов ZETServer.

Nctn - количество контейнеров программы.

Fadc – частота дискретизации канала в Гц.

d1 – дистанция датчика канала 1, м.

d2 – дистанция датчика канала 2, м.

dt – межканальная задержка, мс.

Контейнер программы «Долеоктавный спектр» - это пара каналов ZETServer. Если хотя-бы один канал из этой пары не задан (номер канала равен -1 или нулевой GUID), то контейнер считается не заданным, и работа с ним не выполняется.

| **Параметр (*param*)** | **Аргумент** |
|---|---|
| **Параметры многоконтейнерного режима** | |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| 0 | Установка порядкового номера канала (от 0 до (количество каналов - 1)) . Задание номера текущего канала (-1 допустим). |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| 0 | Используется только для Unit-2. Установка текущего канала по его GUID. |
| 1 | Установка времени усреднения, сек |
| 2 | Задание интервала расчёта доп. графиков (от 10 секунд до 10000 секунд) |
| 3 | Запрос на выдачу размера, текущего спектра и частот |
| 4 | Запрос на выдачу размера, всех спектров и частот |
| 5 | Задание запрета на выдачу данных в UNIT |
| 6 | Запрос на выдачу частот |
| 7 | Задание типа отображаемых значений:<br>0 – в единицах измерения<br>1 – логарифмический масштаб (в децибелах) |
| 8 | Установка типа октавного анализа:<br>0 – октавный<br>1 – 1/3 октавный<br>2 – 1/12 октавный<br>3 – 1/24 октавный |
| 9 | Задание типа рассчитываемых значений:<br>0 - СКЗ;<br>1 - амплитуда |
| 10 | Задание расчета максимального доп. графика:<br>0 - Да;<br>1 - Нет |
| 11 | Задание расчета минимального доп. графика:<br>0 - Да;<br>1 - Нет |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| 12 | Задание расчета среднего доп. графика:<br>0 - Да;<br>1 - Нет |
| 14 | Установка типа обработки сигнала:<br>0 – дифференцирование второго порядка<br>1 – дифференцирование первого порядка<br>2 – без обработки интегрирования и дифференцирования<br>3 – интегрирование первого порядка<br>4 – интегрирование второго порядка |
| 15 | Задание типа частотной коррекции для акустики:<br>0 - линейная<br>1 - коррекция A<br>2 - коррекция B<br>3 - коррекция C<br>4 - коррекция D |
| 16 | Задание передачи всех данных одним массивом |
| 17 | Задание типа усреднения:<br>0 - линейное<br>1 - экспоненциальное |
| 18 | Задание типа частотной коррекции для сейсмики |
| 19 | Задание типа частотной коррекции для виброускорения |
| 20 | Задание флага частотной коррекции:<br>0 - Да;<br>1 - Нет |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| 21 | Задание отображения графиков нормы (0 - выключить, 1 - включить график нормы). Этот пункт выполняется только после выполнения параметра 22, так как он устанавливает флаг отображения выбранной нормы. |
| 22 | Задание имени файла нормы. Пример: D:\ZETLab\config\norma.nrm данные вносятся в поле Передача из файла, нажимаем Передать String. |
| 23 | Задание типа частотной коррекции для виброскорости |
| 1000 | Задание имени файла dtx для сохранения графиков |

### Взаимный узкополосный спектр (vspectr.exe)

**Навигация:** HELP ZETLab studio. Руководство разработчика. 259 > Unit.ocx Предназначен для связи с программами ZETLab 259 > Установка параметров программ ZETLab 259 > меню Анализ сигналов 259 >

# Взаимный узкополосный спектр (spectr.exe)

# Unit-1 и Unit-2

Nsrv - количество каналов ZETServer.
Nctn - количество контейнеров программы.
Fadc – частота дискретизации канала в Гц.
d1 – дистанция датчика канала 1, м.
d2 – дистанция датчика канала 2, м.
dt – межканальная задержка, мс.

Контейнер программы «Взаимный узкополосный спектр» - это пара каналов ZETServer. Если хотя-бы один канал из этой пары не задан (номер канала равен -1 или нулевой GUID), то контейнер считается не заданным, и работа с ним не выполняется.

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| 0 | Установка порядкового номера первого канала (от 0 до (количество каналов - 1)) |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| 1 | Установка порядкового номера второго канала (от 0 до (количество каналов - 1)) |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| 2 | Установка декады (частотного диапазона анализа), Гц:<br>0 – от 0 до (частота дискретизации / 2)<br>1 – от 0 до (частота дискретизации / 20) |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| | 2 – от 0 до (частота дискретизации / 200)<br>3 – от 0 до (частота дискретизации / 2000)<br>4 – от 0 до (частота дискретизации / 20000) |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| 3 | Установка времени усреднения (от 0.1 секунды до 1000 секунд) |
| 4 | Установка типа анализа: |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| | 0 – быстрое преобразование Фурье (БПФ)<br>1 – дискретное преобразование Фурье (ДПФ) |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| 5 | Установка типа весовой функции:<br>0 – прямоугольная<br>1 – весовая функция Ханна (Хэннинга) |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| | 2 - весовая функция Хэмминга<br>3 – весовая функция Блэкмана<br>4 – весовая функция Барлетта |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| | 5 – весовая функция Кайзера<br>6 - весовая функция Рифа-Винсента (4)<br>7 - весовая функция Блэкмана-Харриса (3) |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| | 8 - весовая функция Блэкмана-Харриса (4)<br>9 - весовая функция Наталла<br>10 - весовая функция Блэкмана-Наталла |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| | 11 - весовая функция Окна с плоской вершиной<br>12 - весовая функция Синус-окно |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| 6 | Установка типа обработки сигнала:<br>0 – дифференцирование второго порядка<br>1 – дифференцирование первого порядка |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| | 2 – без обработки интегрирования и дифференцирования<br>3 – интегрирование первого порядка<br>4 – интегрирование второго порядка |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| 7 | Установка типа усреднения:<br>0 – линейное<br>1 – экспоненциальное |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| 8 | Установка порядка для FFT |
| 9 | Установка количества полос для DFT |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| 10 | Установка включения медианного фильтра (Фильтрация дискрет) (аргумент может принимать любое значение) |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| 11 | Установка длины медианного фильтра (Длина дискрет) |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| 12 | Установка включения полосового фильтра (аргумент может принимать любое значение) |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| 13 | Установка частоты ФВЧ полосового фильтра:<br>0 – запрет установки частоты ФВЧ полосового фильтра;<br>1 – разрешение установки частоты ФВЧ полосового фильтра. |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| 14 | Установка частоты ФНЧ полосового фильтра:<br>0 – запрет установки частоты ФНЧ полосового фильтра;<br>1 – разрешение установки частоты ФНЧ полосового фильтра. |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| 15 | Установка включения устранения отражений:<br>0 – запрет расчета мнимой части спектра<br>1 – разрешение расчета мнимой части спектра |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| 16 | Установка уровня подавления антирезонансов (от 30 до 80 дБ) |
| 17 | Установка включения фильтра удаления постоянной составляющей: |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| | 0 – запрет фильтра удаления постоянной составляющей;<br>1 – разрешение фильтра удаления постоянной составляющей. |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| 18 | Установка типа обработки сигнала для первого канала:<br>0 – дифференцирование второго порядка<br>1 – дифференцирование первого порядка |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| | 2 – без обработки интегрирования и дифференцирования<br>3 – интегрирование первого порядка<br>4 – интегрирование второго порядка |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| 19 | Установка типа обработки сигнала для второго канала:<br>0 – дифференцирование второго порядка<br>1 – дифференцирование первого порядка |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| | 2 – без обработки интегрирования и дифференцирования<br>3 – интегрирование первого порядка<br>4 – интегрирование второго порядка |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| 50 | Подача команды на сохранение текущего графика в файл. Имя файла устанавливается предварительно функцией SetParamString (long param, |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| | signed char *value). Для правильного отображения файла необходимо устанавливать расширение *.dtx (передаётся имя файла) |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| 255 | Установка на включение взаимного узкополосного спектрального анализа (аргумент может принимать любое значение) |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| 256 | Установка на включение взаимного узкополосного спектрального анализа (асинхронное, без выравнивания времени) (аргумент может принимать любое значение) |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| 127 | Установка на выключение взаимного узкополосного спектрального анализа (аргумент может принимать любое значение) |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| 500 | Установка флага бесконечности фазы (непрерывно) (аргумент может принимать любое значение) |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| 520 | Установка вида импульсной характеристики (обратная/прямая) |
| 521 | Установка учета фазы при расчете импульсной характеристики |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| 522 | Установка типа импульсной характеристики (0 - Несимметричная, 1 - Симметричная, 2 -Огибающая) |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| 523 | Установка источника расчета импульсной характеристики |
| 524 | Установка флага автоопределения длины импульсной характеристики |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| 525 | Установка начала импульсной характеристики |
| 527 | Установка разрешения сглаживания импульсной характеристики |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| 1000 | Установка получения модуля взаимного спектра |
| 1000 | Установка получения модуля взаимного спектра |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| 1001 | Установка получения фазы |
| 1002 | Установка получения коэффициента когерентности |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| 1003 | Установка получения передаточной характеристики H1 |
| 1004 | Установка получения передаточной характеристики H2 |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| 1005 | Установка получения передаточной характеристики Hv |
| 1006 | Установка получения передаточной характеристики Himp |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| 1007 | Установка получения прямой импульсной характеристики |
| 1008 | Установка получения собственных шумов по каналам |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| 1009 | Установка получения квадрата автоспектра первого канала |
| 1010 | Установка получения квадрата автоспектра второго канала |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| 1012 | Установка получения комплексного взаимного спектра |
| 1013 | Установка получения получения передаточной характеристики Hc |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| 1014 | Установка получения фазы Hc |
| 1015 | Установка получения фазы по Imp |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| 1016 | Установка получения добротности по фазе |
| 1017 | Установка получения добротности по фазе Hc |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| 1018 | Установка получения добротности по передаточной характеристике |
| 1019 | Установка получения добротности по передаточной характеристике Hc |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| 1020 | Установка получения фазовой задержки |
| 1021 | Установка получения групповой задержки по фазовой характеристике |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| 1022 | Установка получения групповой задержки по амплитудной характеристике |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| 1023 | Установка получения фазовой задержки Hc |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| 1024 | Установка получения групповой задержки по фазовой характеристике Hc |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| 1025 | Установка получения групповой задержки по амплитудной характеристике Hc |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| 1026 | Установка получения добротности по компл. передаточной характеристике |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| 1027 | Установка получения добротности по компл. передаточной характеристике Hc |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |
| **Установка параметров текущего контейнера** | |
| Параметр (*param*) | Аргумент (*value*) |
| 1028 | Установка получения результата расчета импульсной характеристки |

© ООО "ЭТМС" 1992-2024. Все права защищены

## Взаимный долеоктавный спектр (dvspectr.exe)

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 0 | Установка порядкового номера первого канала (от 0 до (количество каналов - 1)) |
| 1 | Установка порядкового номера второго канала (от 0 до (количество каналов - 1)) |
| 2 | Установка времени усреднения (от 0.1 секунды до 100 секунд) |
| 4 | Установка на получение размера массива спектра по операции *Read(...)* (аргумент может принимать любое значение). После задания этого параметра необходимо дождаться события готовности *Ready(...)* и затем прочитать размер массива (частот и спектра) операцией *Read(...)* |
| 5 | Установка на получение массива значений частотного ряда, текущего спектра, действительной и мнимой частей спектра, фазы и коэффициента когерентности по операции *Read(...)* (аргумент может принимать любое значение). После задания этого параметра необходимо дождаться события готовности *Ready(...)* и затем прочитать массив значений частотного ряда, текущего спектра, действительной и мнимой частей спектра, фазы и коэффициента когерентности операцией *Read(...)* |
| 6 | Установка на получение массива значений частотного ряда по операции *Read(...)* (аргумент может принимать любое значение). После задания этого параметра необходимо дождаться события готовности *Ready(...)* и затем прочитать массив значений частотного ряда операцией *Read(...)* |
| 7 | Установка на получение массива значений текущего спектра по операции *Read(...)* (аргумент может принимать любое значение). После задания этого параметра необходимо дождаться события готовности *Ready(...)* и затем прочитать массив значений текущего спектра операцией *Read(...)* |
| 8 | Установка на получение массива значений действительной части спектра по операции *Read(...)* (аргумент может принимать любое значение). После задания этого параметра необходимо дождаться события готовности *Ready(...)* и затем прочитать массив значений действительной части спектра операцией *Read(...)* |
| 9 | Установка на получение массива значений мнимой части спектра по операции *Read(...)* (аргумент может принимать любое значение). После задания этого параметра необходимо дождаться события готовности |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| | *Ready(...)* и затем прочитать массив значений мнимой части спектра операцией *Read(...)* |
| 10 | Установка запрета на получение данных (аргумент может принимать любое значение) |
| 11 | Установка типа анализа:<br>0 – 1/1 октавный<br>1 – 1/3 октавный<br>2 – 1/12 октавный<br>3 – 1/24 октавный |
| 12 | Установка типа представления уровня спектральных компонент:<br>0 – линейный масштаб (в единицах измерения)<br>1 – логарифмический масштаб (в децибелах) |
| 13 | Установка на расчет действительной части:<br>0 – запрет расчета действительной части спектра<br>1 – разрешение расчета действительной части спектра |
| 14 | Установка на расчет мнимой части:<br>0 – запрет расчета мнимой части спектра<br>1 – разрешение расчета мнимой части спектра |
| 15 | Установка на расчет фазы:<br>0 – запрет расчета фазы<br>1 – разрешение расчета фазы |
| 16 | Установка на расчет коэффициента когерентности:<br>0 – запрет расчета коэффициента когерентности<br>1 – разрешение расчета коэффициента когерентности |
| 17 | Подача команды на сохранение текущего графика в файл. Имя файла устанавливается предварительно функцией SetParamString(long param, signed char *value). Для правильного отображения файла необходимо устанавливать расширение *.dtu |
| 255 | Установка на включение взаимного долеоктавного спектрального анализа (аргумент может принимать любое значение) |
| 127 | Установка на выключение взаимного долеоктавного спектрального анализа (аргумент может принимать любое значение) |

## Взаимный корреляционный анализ (corr.exe)

Nsrv - количество каналов ZETServer.

© ООО "ЭТМС" 1992-2024. Все права защищены

Nctn - количество контейнеров программы.
Fadc – частота дискретизации канала в Гц.
d1 – дистанция датчика канала 1, м.
d2 – дистанция датчика канала 2, м.
dt – межканальная задержка, мс.

Контейнер программы «Взаимный корреляционный анализ» - это пара каналов ZETServer. Если хотя-бы один канал из этой пары не задан (номер канала равен -1 или нулевой GUID), то контейнер считается не заданным, и работа с ним не выполняется.

| Параметр (*param*) | Аргумент |
|---|---|
| **Параметры многоконтейнерного режима** | |
| -1 | Задание количества контейнеров. Параметр типа long, допустимое значение от 1 до кол-ва каналов сервера (Nsrv). |
| -2 | Задание индекса текущего контейнера. Параметр типа long, допустимое значение от 0 до кол-ва контейнеров программы (Nctn) без единицы (Nctn - 1). |
| -3 | Задание индекса останавливаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -4 | Задание индекса запускаемого контейнера. Параметр типа long, допустимое значение от 0 до Nctn - 1. |
| -5 | Добавление контейнера с не заданными каналами. Параметр типа long, значение — любое. |
| -6 | Начало передачи серии параметров текущего контейнера (остановка передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -7 | Конец передачи серии параметров текущего контейнера (возобновление передачи результатов контейнера UNIT-клиенту). Параметр любого типа, значение — любое. |
| -8 | Передача текущему контейнеру структуры параметров ParamOneKit. Тип параметра – tdu_str. |
| -9 | Запрос на передачу Unit-клиенту структуры параметров ParamOneKit текущего контейнера. Параметр любого типа, значение — любое. |
| -10 | Остановка работы программы. Параметр любого типа, значение — любое. |
| -11 | Запуск работы программы. Параметр любого типа, значение — любое. |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| **Установка параметров текущего контейнера** | |
| 0 | Установка первого канала контейнера по его порядковому номеру. Параметр типа long, допустимое значение от -1 до Nsrv – 1. |
| 0 | Установка первого канала контейнера по его GUID. Параметр типа tdu_guid. |
| 1 | Установка второго канала контейнера по его порядковому номеру. Параметр типа long, допустимое значение от -1 до Nsrv – 1. |
| 1 | Установка второго канала контейнера по его GUID. Параметр типа tdu_guid. |
| 2 | Установка времени усреднения в сек. Параметр типа float, допустимое значение от 0.1 до 86400 (сутки в сек). |
| 3 | Установка кода (k) размера корреляционной функции. Параметр типа long, допустимое значение от 0 до 11.<br><br>Диапазон рассчитываемых задержек в мс от -T до +T, где:<br>T = 2^(k+6) / Fadc * 1000.<br><br>Диапазон рассчитываемых дистанций в м от x1 до x2, где:<br>x1 = (d2 + d1) / 2 - 2^(k+6) / Fadc * \|d2 – d1\| / dt * 1000<br>x2 = (d2 + d1) / 2 + 2^(k+6) / Fadc * \|d2 – d1\| / dt * 1000 |
| 4 | Установка полосового фильтра. Параметр типа long, допустимые значения:<br>0 – выключение фильтра;<br>1 – включение фильтра. |
| 5 | Установка частоты среза фильтра высоких частот. Параметр типа float, допустимое значение от 0 Гц до частоты среза фильтра низких частот. |
| 6 | Установка частоты среза фильтра низких частот. Параметр типа float, допустимое значение от частоты среза фильтра высоких частот до Fadc/2. |
| 7 | Установка фильтрации дискрет. Параметр типа long, допустимые значения:<br>0 – выключение фильтра;<br>1 – включение фильтра. |
| 8 | Установка инверсии сигнала. Параметр типа long, допустимые значения:<br>0 – выключение инверсии;<br>1 – включение инверсии. |
| 9 | Установка режима работы с постоянными составляющими сигналов. Параметр типа long, допустимые значения:<br>0 — удаление; |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| | 1- удаление с помощью специализированного ФВЧ;<br>2 — без изменения. |
| 10 | Включение/отключение текущего контейнера. Параметр типа long, допустимые значения:<br>0 – выключение;<br>1 – включение. |
| 11 | Не используется |
| 12 | Установка декады. Параметр типа long, допустимые значения 0 или 2. |
| 13 | Задание дистанции канала 1 в метрах. Параметр типа float. |
| 14 | Задание дистанции канала 2 в метрах. Параметр типа float. |
| 15 | Установка расчета дистанций. Параметр типа long, допустимые значения:<br>0 – расчет временных задержек;<br>1 – расчет дистанций. |
| 16 | Установка кода рассчитываемой функции. Параметр типа long, допустимые значения:<br>0 - корр. функция;<br>1 - коэфф. корреляции;<br>2 - функция правдоподобия;<br>3 - разностная функция с задаваемым положением максимума корр. функции;<br>4 - разностная функция с автоматическим определением положения максимума корр. функции;<br>5 - подавление помех канала 2;<br>6 - подавление помех. |
| 17 | Задание межканальной задержки между каналами в режиме расчёта дистанций, мсек. Параметр типа float. Значение должно быть строго больше 0. |
| 18 | Задание величины положения максимума корр. Функции (используется только при расчёте разностной функция с задаваемым положением максимума корр. Функции). Параметр типа float. Значение в мсек или метрах. |
| 19 | Задание типа усреднения. Параметр типа long, допустимые значения:<br>0 – линейное;<br>1 – экспоненциальное. |
| 20 | Задание кода уменьшение размера. Параметр типа long, допустимые значения: |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент |
|---|---|
| | 1 - отображается весь интервал рассчитываемых значений;<br>2 - отображается половина интервала рассчитываемых значений;<br>3 - отображается четверть интервала рассчитываемых значений;<br>4 - отображается восьмая часть интервала рассчитываемых значений; |
| 21 | Задание левой видимой части графика в текущих единицах оси X (мсек или м). Параметр типа double. |
| 22 | Задание правой видимой части графика в текущих единицах оси X (мсек или м). Параметр типа double. |
| 23 | Задание установки типа графика коррелограмм. Параметр типа long, допустимые значения 0 или 2;<br>0 - результат;<br>1 - огибающая результата;<br>2 - частота несущей; |
| 24 | Задание установки, при необходимости, корректирующего окна:<br>0 – не устанавливать;<br>1 – устанавливать. |
| 1000 | Подача команды на сохранение текущего графика в dtu-файл. Параметр типа tdu_txt, в котором записано имя файла для графика. |

## Анализ нелинейных искажений (harmdist.exe)

Используется Unit2.

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 0 | Установка порядкового номера канала (от 0 до (количество каналов - 1)) |
| 1 | Установка типа представления расчета коэффициента нелинейных искажений:<br>0 – линейный масштаб (в процентах)<br>1 – логарифмический масштаб (в децибелах) |
| 2 | Установка декады (частотного диапазона анализа) Гц:<br>0 – от 0 до (частота дискретизации / 2)<br>1 – от 0 до (частота дискретизации / 20) |
| 3 | Установка времени усреднения (от 0.1 секунды до 10 секунд) |
| 4 | Установка автомасштабирования: |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| | 0 – установлено автомасштабирование<br>1 – отключено автомасштабирование |
| 127 | Установка на выключение анализа нелинейных искажений (аргумент может принимать любое значение) |
| 255 | Установка на включение анализа нелинейных искажений (аргумент может принимать любое значение) |
| 1000 | Подача команды на сохранение текущего графика в файл. Имя файла устанавливается предварительно функцией SetParamString (long param, signed char *value). Для правильного отображения файла необходимо устанавливать расширение *.dtx (передаётся имя файла) |

## Синхронное накопление (PrdkAnaliz.exe)

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 0 | Установка порядкового номера измерительного канала (от 0 до (количество каналов - 1)) |
| 1 | Установка порядкового номера опорного канала (от 0 до (количество каналов -1)) |
| 2 | Установка типа фронта запуска:<br>0 – ниспадающий фронт<br>1 – восходящий фронт |
| 3 | Установка типа обработки сигнала:<br>0 – дифференцирование второго порядка<br>1 – дифференцирование первого порядка<br>2 – без обработки интегрирования и дифференцирования<br>3 – интегрирование первого порядка<br>4 – интегрирование второго порядка |
| 4 | Установка типа представления уровня спектральных компонент гармоник, отображение по Y:<br>0 – линейный масштаб (в единицах измерения)<br>1 – логарифмический масштаб (в децибелах) |
| 5 | Установка времени усреднения (от 0.1 секунды до 10 секунд) |
| 6 | Установка на включение спектра гармоник:<br>0 – выключение спектра гармоник<br>1 – включение спектра гармоник |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 7 | Установка на включение спектрограммы гармоник:<br>0 – выключение спектрограммы гармоник<br>1 – включение спектрограммы гармоник |
| 8 | Установка на включение трехмерной спектрограммы гармоник:<br>0 – выключение трехмерной спектрограммы гармоник<br>1 – включение трехмерной спектрограммы гармоник |
| 9 | Установка количества гармоник, отображаемых в спектре гармоник (от 1 до 255) |
| 10 | Установка на включение отображения в полярных координатах:<br>0 – выключение отображения в полярных координатах<br>1 – включение отображения в полярных координатах |
| 11 | Установка на отображение шестеренки на графике в полярных координатах:<br>0 – не отображать шестеренку на графике в полярных координатах<br>1 – отображать шестеренку на графике в полярных координатах |
| 12 | Установка количества зубьев в шестеренке на графике в полярных координатах (от 1 до 99) |
| 13 | Установка декады (частотного диапазона анализа):<br>0 – от 0 до (частота дискретизации / 1)<br>1 – от 0 до (частота дискретизации / 10)<br>2 – от 0 до (частота дискретизации / 100)<br>3 – от 0 до (частота дискретизации / 1000)<br>4 – от 0 до (частота дискретизации / 10000) |
| 14 | Установка вида установки передаточного числа:<br>0 – в виде рациональной дроби (через параметры 16-21)<br>1 – в виде числа с плавающей запятой (через параметр 15) |
| 15 | Установка передаточного числа в плавающей запятой (от 0.001 до 1000) |
| 16 | Установка первого числителя кинематического параметра (от 1 до 99) |
| 17 | Установка второго числителя кинематического параметра (от 1 до 99) |
| 18 | Установка третьего числителя кинематического параметра (от 1 до 99) |
| 19 | Установка первого знаменателя кинематического параметра (от 1 до 99) |
| 20 | Установка второго знаменателя кинематического параметра (от 1 до 99) |
| 21 | Установка третьего знаменателя кинематического параметра (от 1 до 99) |
| 22 | Установка режима передачи данных:<br>0 – передача формы сигнала |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| | 1 – передача спектра гармоник<br>После задания этого параметра необходимо дождаться события готовности *Ready(...)* и затем прочитать массив передаваемых данных операцией *Read(...)* |
| 127 | Установка на выключение синхронного накопления (аргумент может принимать любое значение). При повторной подаче любого аргумента происходит включение синхронного накопления. |
| 255 | Установка на включение синхронного накопления (аргумент может принимать любое значение) |

## Модальный анализ (PrqsAnaliz.exe)

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 0 | Установка декады опорного канала (частотного диапазона анализа) Гц:<br>0 – от 0 до (частота дискретизации / 1)<br>1 – от 0 до (частота дискретизации / 10)<br>2 – от 0 до (частота дискретизации / 100)<br>3 – от 0 до (частота дискретизации / 1000)<br>4 – от 0 до (частота дискретизации / 10000) |
| 1 | Установка декады измерительного канала (частотного диапазона анализа) Гц:<br>0 – от 0 до (частота дискретизации / 1)<br>1 – от 0 до (частота дискретизации / 10)<br>2 – от 0 до (частота дискретизации / 100)<br>3 – от 0 до (частота дискретизации / 1000)<br>4 – от 0 до (частота дискретизации / 10000) |
| 2 | Установка частотного диапазона, Гц |
| 3 | Установка интервала расчета (в секундах) |
| 4 | Установка инверсии опорного канала:<br>0 – не инвертировать опорный канал<br>1 – инвертировать опорный канал |
| 5 | Установка инверсии измерительного канала:<br>0 – не инвертировать измерительный канал<br>1 – инвертировать измерительный канал |
| 6 | Установка нормировки измерительного канала: |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| | 0 – не нормировать измерительный канал<br>1 – нормировать измерительный канал |
| 7 | Установка типа порога по СКЗ шумов:<br>0 – адаптивный (СКЗ * К)<br>1 – абсолютный |
| 8 | Установка множителя СКЗ адаптивного порога |
| 9 | Установка абсолютного уровня СКЗ опорного канала, ед. измерений |
| 10 | Установка абсолютного уровня СКЗ измерительного канала, ед. измерений |
| 11 | Установка типа фронта запуска:<br>0 – любой фронт<br>1 – только положительный фронт |
| 12 | Установка интервала расчета добротностей (от 0.1% до 100%) |
| 13 | Установка автозапуска интервала анализа:<br>0 – запрет автозапуска анализа<br>1 – разрешение автозапуска анализа |
| 14 | Установка интервала времени, через которое происходит автозапуск (от 0.1 секунды до 1000 секунд) |
| 16 | Установка на включение фильтрация дискрет (аргумент может принимать любое значение) |
| 17 | Установка на изменение длины дискрет (от 0 до 3) |
| 18 | Установка на включение на включение полосового фильтра (аргумент может принимать любое значение) |
| 19 | Установка на включение на включение полосового фильтра и изменение ФВЧ (от 0 до 5000), не больше суммы ФВЧ и ФНЧ. |
| 20 | Установка на включение на включение полосового фильтра и изменение ФНЧ (от 0 до 5000), не больше суммы ФВЧ и ФНЧ. |
| 21 | Установка на включение на изменение порядка фильтра ФВЧ (от 0 до 12) |
| 22 | Установка на включение на изменение порядка фильтра ФНЧ (от 0 до 12) |
| 127 | Установка на выключение модального анализа (аргумент может принимать любое значение) |
| 255 | Установка на включение модального анализа (аргумент может принимать любое значение) |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 1000 | Подача команды на сохранение текущего графика в файл. Имя файла устанавливается предварительно функцией SetParamString(long param, signed char *value). Для правильного отображения файла необходимо устанавливать расширение *.dtu |

## Гистограмма (ZETHistograph.exe)

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 0 | Установка типа расчета значений:<br>0 – по напряжению<br>1 – по разрядам |
| 1 | Установка ширины столбца гистограммы, ед. изм.:<br>– при установке типа расчета значений по напряжению задается от веса младшего разряда АЦП до максимально допустимого входного уровня, деленного на 16<br>– при установке типа расчета значений по разрядам задается от 1 до (разрядность АЦП – 5) |
| 2 | Установка порядкового номера канала (от 0 до (количество каналов - 1)) |
| 3 | Установка типа представления расчета:<br>0 – гистограмма<br>1 – плотность вероятности<br>2 – вероятность |
| 4 | Установка времени накопления данных (от времени усреднения данных до 500 секунд) |
| 5 | Установка времени усреднения:<br>0 - 0.1 секунды<br>1 - 1 секунда<br>2 - 10 секунд |
| 6 | Установка на расчет нормального распределения (Гаусса)<br>0 – запрет расчета нормального распределения<br>1 – разрешение расчета нормального распределения |
| 7 | Установка на расчет гармонического распределения<br>0 – запрет расчета гармонического распределения<br>1 – разрешение расчета гармонического распределения |
| 8 | Установка на расчет распределения Хи-квадрат |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| | 0 – запрет расчета распределения Хи-квадрат<br>1 – разрешение расчета распределения Хи-квадрат |
| 9 | Установка количества степеней свободы распределения Хи-квадрат (от 1 до 15) |
| 10 | Подача команды на сохранение текущего графика в файл. Имя файла устанавливается предварительно функцией SetParamString(long param, signed char *value). Для правильного отображения файла необходимо устанавливать расширение *.dtu |
| 127 | Установка на выключение гистограммы (аргумент может принимать любое значение) |
| 255 | Установка на включение гистограммы (аргумент может принимать любое значение) |

### Детектор STA\LTA (STA_LTA.exe)

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 0 | Установка размерности исходного сигнала:<br>1 – скалярный;<br>3 – векторный 3D. |
| 1 | зарезервировано |
| 2 | Установка порядкового номера канала X (от 0 до (количество каналов - 1)) |
| 3 | Установка порядкового номера канала Y (от 0 до (количество каналов - 1)) |
| 4 | Установка порядкового номера канала Z (от 0 до (количество каналов - 1)), при установке скалярной размерности исходного сигнала в качестве канала выставляется именно компонента Z. |
| 5 | зарезервировано |
| 6 | Установка нижней частоты среза полосового фильтра (от 0 до верхней частоты среза полосового фильтра), Гц |
| 7 | Установка верхней частоты среза полосового фильтра (от нижней частоты среза полосового фильтра до (частота дискретизации / 2)), Гц |
| 8 | Установка длительности короткого окна детектора STA\LTA (длительность STA) (от 0 до (длительность LTA / 10)), с |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 9 | Установка длительности длинного окна детектора STA\LTA (длительность LTA) (от (длительность STA * 10) до бесконечности), с |
| 10 | Установка порога детектирования (от 3 до 60) |
| 11 | Установка на создание виртуального канала STA_LTA:<br>0 – запрет создания виртуального канала STA_LTA;<br>1 – разрешение создания виртуального канала STA_LTA. |
| 12 | Установка на запись сигналов при обнаружении события в файл результатов *.dtu:<br>0 – запрет записи сигналов;<br>1 – разрешение записи сигналов. |
| 13 | Установка на отображение информации в окне программы:<br>0 – запрет отображения информации в окне программы;<br>1 – разрешение отображения информации в окне программы. |
| 14 | Установка записи сообщений в log-файл:<br>0 – запрет записи;<br>1 – разрешение записи. |
| 15 | Установка режима мониторинга:<br>0 – запрет режима мониторинга;<br>1 – разрешение режима мониторинга. |
| 16 | Установка коэффициента сжатия в режиме мониторинга:<br>0 – запрет коэффициента сжатия в режиме мониторинга;<br>1 – разрешение коэффициента сжатия в режиме мониторинга. |
| 17 | Установка на применение устанавливаемых настроек программы (аргумент может принимать любое значение). После задания этого параметра произойдет применение настроек, выставленных другими параметрами. |

### Вейвлет (FDWT.exe)

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 0 | Установка порядкового номера канала (от 0 до (количество каналов - 1)) |
| 1 | Тип материнского вейвлета: 0 Haar, 1 db1, 2 db2, 3 db3, 4 db4, 5 db5, 6 db6, 7 db7, 8 db8, 9 db9, 10 db10, 11 sym1, 12 sym2, 13 sym3, 14 sym4, 15 sym5, 16 sym6, 17 sym7, 18 coif 1, 19 coif 2, 20 coif 3, 21 coif 4, 22 coif 5 |
| 2 | Уровень разложения при выполнении вейвлет-преобразования |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 3 | Временной интервал длительности отображаемого сигнала |
| 4 | Величина задания цветовой шкалы отображения значений рассчитанных коэффициентов (от 1 до 13) |

## 5.2.меню Измерение

Содержит перечень программ из вкладки меню Измерение из ZETLab, которыми можно управлять через Unit или Unit-2.

### Вольтметр переменного тока (VoltMeter.exe)

Данная программа имеет два режима работы: одноканальный и многоканальный. Многоканальный режим возможен только при запуске программы через UNIT.

Программа всегда запускается в одноканальном режиме. Перевод программы в многоканальный режим осуществляется передачей требуемого количества каналов вольтметра с параметром "-1". Далее следует задать параметры (номер канала сервера и время усреднения) каналам вольтметра. Параметры каналам вольтметра задаются с помощью текущего канала вольтметра, например, как в приведённом ниже примере, где 5-ому (счёт с 0) каналу вольтметра задаётся 7-ой (счёт с 0) канал сервера и время усреднения 1 сек:

1. посылка величины "5" с параметром "-2" (5-й канал вольтметра назначается текущим);
2. посылка величины "7" с параметром "0" (текущему каналу вольтметра задаётся 7-ой канала сервера);
3. посылка величины "1" с параметром "1" (текущему каналу вольтметра задаётся временя усреднения 1 сек).

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 0 | Установка порядкового номера канала (от 0 до (количество каналов - 1))/<br>В многоканальном режиме работы установка номера канала сервера для текущего номера канала вольтметра |
| 1 | Установка времени усреднения:<br>0 – быстро (0.1 секунды)<br>1 – медленно (1 секунда)<br>2 – очень медленно (10 секунд)<br>В многоканальном режиме работы установка времени усреднения для текущего номера канала вольтметра |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 2 | Установка типа значения, передаваемого в UNIT:<br>0 – среднеквадратичное значение<br>1 – пиковое значение<br>2 – амплитудное значение<br>В многоканальном режиме работы установка типа передаваемого значения для текущего канала |
| 3 | Установка типа представления расчета:<br>0 – линейный масштаб (в единицах измерения)<br>1 – логарифмический масштаб (в децибелах)<br>В многоканальном режиме работы установка типа для текущего канала |
| -1 | Перевод программы в многоканальный режим работы (без возможности вернуться в одноканальный режим работы). Установка количества каналов вольтметра. Нулевой канал вольтметра становится текущим. Максимальное кол-во каналов вольтметра - 128. |
| -2 | Многоканальный режим работы. Установка текущего номера канал вольтметра для задания его параметров (номер канала сервера и время усреднения) |
| -3 | Многоканальный режим работы. Номер канала вольтметра, работа с данными (считывание и обработка) которого приостанавливается. При этом сам канал вольтметра не уничтожается |
| -4 | Многоканальный режим работы. Номер канала вольтметра, работа с данными (считывание и обработка) которого возобновляется после приостановки |
| -5 | Многоканальный режим работы. Добавление одного нового канала вольтметра (если не превышено максимальное число каналов). Значение *value* - это номер канала сервера, который будет задан новому каналу вольтметра, который автоматически становится текущим |

## Вольтметр постоянного тока (VoltMeterDC.exe)

Данная программа имеет два режима работы: одноканальный и многоканальный. Многоканальный режим возможен только при запуске программы через UNIT.

Программа всегда запускается в одноканальном режиме. Перевод программы в многоканальный режим осуществляется передачей требуемого количества каналов вольтметра с параметром "-1". Далее следует задать параметры (номер канала сервера и время усреднения) каналам вольтметра. Параметры каналам вольтметра задаются с помощью текущего канала вольтметра, например, как в приведённом ниже примере, где

© ООО "ЭТМС" 1992-2024. Все права защищены

5-ому (счёт с 0) каналу вольтметра задаётся 7-ой (счёт с 0) канал сервера и время усреднения 1 сек:

1. посылка величины "5" с параметром "-2" (5-й канал вольтметра назначается текущим);
2. посылка величины "7" с параметром "0" (текущему каналу вольтметра задаётся 7-ой канала сервера);
3. посылка величины "1" с параметром "1" (текущему каналу вольтметра задаётся временя усреднения 1 сек).

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 0 | Установка номера канала сервера (от 0 до (количество каналов - 1)).<br>В многоканальном режиме работы установка номера канала сервера для текущего номера канала вольтметра |
| 1 | Установка времени усреднения:<br>0 – быстро (0.1 секунды);<br>1 – медленно (1 секунда);<br>2 – очень медленно (10 секунд).<br>При посылки аргумента менее 0, считается, что *value* = 0. При посылки аргумента более 2, считается, что *value* = 2.<br>В многоканальном режиме работы установка времени усреднения для текущего номера канала вольтметра |
| -1 | Перевод программы в многоканальный режим работы (без возможности вернуться в одноканальный режим работы). Установка количества каналов вольтметра. Нулевой канал вольтметра становится текущим. Максимальное кол-во каналов вольтметра - 128. |
| -2 | Многоканальный режим работы. Установка текущего номера канал вольтметра для задания его параметров (номер канала сервера и время усреднения) |
| -3 | Многоканальный режим работы. Номер канала вольтметра, работа с данными (считывание и обработка) которого приостанавливается. При этом сам канал вольтметра не уничтожается |
| -4 | Многоканальный режим работы. Номер канала вольтметра, работа с данными (считывание и обработка) которого возобнавляется после приостановки |
| -5 | Многоканальный режим работы. Добавление одного нового канала вольтметра (если не превышено максимальное число каналов). Значение *value* - это номер канала сервера, который будет задан новому каналу вольтметра, который автоматически становится текущим |

© ООО "ЭТМС" 1992-2024. Все права защищены

### Селективный вольтметр (VoltMeterSel.exe)

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 0 | Установка порядкового номера канала (от 0 до (количество каналов- 1)) |
| 1 | Установка времени усреднения:<br>0 – быстро (0.1 секунды)<br>1 – медленно (1 секунда)<br>2 – очень медленно (10 секунд) |
| 2 | Установка варианта выбора частоты и полосы:<br>0 – ручной режим<br>1 – автоматический режим |
| 3 | Установка центральной частоты селективного фильтра, Гц |
| 4 | Установка полосы селективного фильтра, Гц |

### Частотомер (FreqMeter.exe)

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 0 | Установка порядкового номера канала (от 0 до (количество каналов - 1)) |
| 1 | Установка времени усреднения:<br>0 – быстро (0.1 секунды)<br>1 – медленно (1 секунда)<br>2 – очень медленно (10 секунд) |
| 2 | Сброс накопленных данных частотомера:<br>1 - сброс данных. |

### Фазометр (PhaseMeter.exe)

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 0 | Установка порядкового номера первого канала (от 0 до (количество каналов - 1)) |
| 1 | Установка порядкового номера второго канала (от 0 до (количество каналов - 1)) |
| 2 | Установка времени усреднения: |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| | 0 – быстро (0.1 секунды)<br>1 – медленно (1 секунда) |
| 3 | Установка варианта расчета фазы:<br>0 – в градусах<br>1 – в радианах |

### Тахометр (TahoMeter.exe)

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 0 | Установка порядкового номера канала (от 0 до (количество каналов - 1)) |
| 2 | Установка на сброс счетчика оборотов (аргумент может принимать любое значение) |
| 3 | Установка первого числителя кинематического параметра (от 1 до 199) |
| 4 | Установка второго числителя кинематического параметра (от 1 до 199) |
| 5 | Установка третьего числителя кинематического параметра (от 1 до 199) |
| 6 | Установка первого знаменателя кинематического параметра (от 1 до 199) |
| 7 | Установка второго знаменателя кинематического параметра (от 1 до 199) |
| 8 | Установка третьего знаменателя кинематического параметра (от 1 до 199) |
| 9 | Установка варианта выбора порога:<br>0 – ручной режим<br>1 – автоматический режим |
| 10 | Установка верхнего порога, ед.изм. |
| 11 | Установка нижнего порога, ед. изм. |
| 12 | Установка множителя (1, 10, 100, 1000, 10000, 100000) |
| 13 | Установка размерности частоты вращения (0 - об/мин, 1 - об/сек) |
| 14 | Создание виртуального канала кол-ва оборотов (1 - создавать, 0 - нет) |

© ООО "ЭТМС" 1992-2024. Все права защищены

## Торсиограф (Torsiograph.exe)

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 0 | Установка порядкового номера канала фазы А (от 0 до (количество каналов - 1)) |
| 1 | Установка порядкового номера канала фазы Б (от 0 до (количество каналов - 1)) |
| 2 | Установка порядкового номера канала реперной метки 0 (от 0 до (количество каналов - 1)) |
| 3 | Установка работы канала фазы Б:<br>0 – выключение канала фазы Б<br>1 – включение канала фазы Б |
| 4 | Установка работы канала реперной метки 0:<br>0 – выключение канала реперной метки 0<br>1 – включение канала реперной метки 0 |
| 5 | Установка единица измерения:<br>0 – миллиметр<br>1 – сантиметр<br>2 – метр<br>3 – градус<br>4 – оборот |
| 6 | Установка разрешающей способности (количества меток на единицу измерения) |
| 7 | Установка на сброс счетчика оборотов (аргумент может принимать любое значение) |
| 8 | Установка на создание виртуального канала перемещения:<br>0 – запрет создания виртуального канала перемещения<br>1 – разрешение создания виртуального канала перемещения |
| 9 | Установка на создание виртуального канала скорости:<br>0 – запрет создания виртуального канала скорости<br>1 – разрешение создания виртуального канала скорости |
| 10 | Установка на инвертирование направления перемещения (работает при установке на включение канала фазы Б параметром 3):<br>0 – не инвертировать направление перемещения<br>1 – инверсия направления перемещения |
| 11 | Установка варианта выбора порога:<br>0 – ручной режим |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| | 1 – автоматический режим |
| 12 | Установка верхнего порога, ед. измер. канала |
| 13 | Установка нижнего порога, ед. измер. канала |

### Энкодер (Encoder.exe)

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 0 | Установка порядкового номера канала фазы А (от 0 до (количество каналов - 1)) |
| 1 | Установка порядкового номера канала фазы Б (от 0 до (количество каналов - 1)) |
| 2 | Установка порядкового номера канала реперной метки 0 (от 0 до (количество каналов - 1)) |
| 3 | Установка работы канала фазы Б:<br>0 – выключение канала фазы Б<br>1 – включение канала фазы Б |
| 4 | Установка работы канала реперной метки 0:<br>0 – выключение канала реперной метки 0<br>1 – включение канала реперной метки 0 |
| 5 | Установка единица измерения:<br>0 – миллиметр<br>1 – сантиметр<br>2 – метр<br>3 – градус<br>4 – оборот |
| 6 | Установка разрешающей способности (количества меток на единицу измерения) |
| 7 | Установка на сброс счетчика оборотов (аргумент может принимать любое значение) |
| 8 | Установка на создание виртуального канала перемещения:<br>0 – запрет создания виртуального канала перемещения<br>1 – разрешение создания виртуального канала перемещения |
| 9 | Установка на создание виртуального канала скорости:<br>0 – запрет создания виртуального канала скорости |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| | 1 – разрешение создания виртуального канала скорости |
| 10 | Установка на инвертирование направления перемещения (работает при установке на включение канала фазы Б параметром 3):<br>0 – не инвертировать направление перемещения<br>1 – инверсия направления перемещения |
| 11 | Установка варианта выбора порога:<br>0 – ручной режим<br>1 – автоматический режим |
| 12 | Установка верхнего порога, ед. измер. канала |
| 13 | Установка нижнего порога, ед. измер. канала |

### Термометр ТС (ThermoMeter.exe)

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 0 | Установка порядкового номера измерительного канала (от 0 до (количество каналов - 1)) |
| 1 | Установка порядкового номера опроного канала (от 0 до (количество каналов - 1)) |
| 2 | Установка типа термопреобразователя:<br>0 – Pt W = 1.391 (0 – платина = 1.391)<br>1 – Pt W = 1.385 (1 – платина = 1.385)<br>2 – Cu W = 1.428 (2 – медь = 1.428)<br>3 – Cu W = 1.426 (3 – медь = 1.426)<br>4 – Nc W = 1.617 (4 – никель = 1.617) |
| 3 | Установка на включение поправки поправки в градусах (от минус 100 до 100):<br>0 – выключение поправки в градусах;<br>1 – включение поправки в градусах |

### Термометр ТП (ThermoPara.exe)

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 0 | Установка порядкового номера измерительного канала (от 0 до (количество каналов - 1)) |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 1 | Установка порядкового номера канала компенсатора холодного спая (от 0 до (количество каналов - 1)) |
| 2 | Установка типа термопары:<br>0 – R (ТПП13 – 13% родий/платина)<br>1 – S (ТПП10 – 10% родий/платина)<br>2 – B (ТПР – платина/родий)<br>3 – J (ТЖК – железо/константан)<br>4 – T (ТМКн – медь/константан)<br>5 – E (ТХКн – хромель/константан)<br>6 – K (ТХА – хромель/алюмель)<br>7 – N (ТНН – нихросил/нисил)<br>8 – A (ТВР – вольфрам/рений)<br>9 – L (ТХК – хромель/копель) |
| 3 | Установка на включение поправки по температуре холодного спая:<br>0 – выключение поправки по температуре холодного спая<br>1 – включение поправки по температуре холодного спая |

### Тензометр (TenzoMeter.exe)

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 0 | Установка порядкового номера измерительного канала (от 0 до (количество каналов - 1)) |
| 1 | Установка порядкового номера опорного канала (от 0 до (количество каналов -1)) |
| 2 | Установка на использование файла калибровки:<br>0 – запрет использования файла калибровки<br>1 – разрешение использования файла калибровки |
| 3 | Установка режима измерений:<br>0 –тензорезистор<br>1 – тензодатчик |
| 4 | Установка чувствительности тензодатчика, мВ/В |
| 5 | Установка предела измерений по тензодатчику |
| 6 | Установка имени файла калибровки тензорезистора |
| 7 | Установка единицы измерений |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 8 | Установка типа измерений:<br>0 – абсолютные измерения<br>1 – относительные измерения |
| 9 | Установка типа питания датчика:<br>0 – постоянный ток<br>1 – переменный ток |
| 10 | Установка на инвертирование измеренного значения нагрузки:<br>0 – не инвертировать измеренное значение нагрузки<br>1 – инверсия измеренного значения нагрузки |
| 11 | Установка значения поправки величины нагрузки, которое получено программой при последнем сбросе. |
| 12 | Установка числа, в которое сбрасывается текущий показатель при сбросе через параметр 14 |
| 13 | Установка длины фильтра сглаживания, мс |
| 14 | Установка на сброс текущего значения (аргумент может принимать любое значение) |
| 15 | Установка на использование канала термокомпенсации |
| 16 | Установка порядкового номера канала термокомпенсации |
| 17 | Установка коэффициента усиления по измерительному каналу |
| 18 | Установка значения коэффициента |
| 19 | Установка разрешения выдачи виртуальногго канала под UNIT-1 (команда однократная) |

## Виброметр (VibroMeter.exe)

Данная программа имеет два режима работы: одноканальный и многоканальный. Многоканальный режим возможен только при запуске программы через UNIT.

Программа всегда запускается в одноканальном режиме. Перевод программы в многоканальный режим осуществляется передачей требуемого количества каналов виброметра с параметром "-1". Далее следует задать параметры (номер канала сервера, код времи усреднения, тип полосового фильтра и тип рассчитываемого значения) каналам вольтметра. Параметры каналам виброметра задаются с помощью текущего канала виброметра, например, как в приведённом ниже примере, где 5-ому (счёт с 0) каналу вольтметра задаётся 7-ой (счёт с 0) канал сервера и время усреднения 1 сек:

1. посылка величины "5" с параметром "-2" (5-й канал виброметра назначается текущим);

© ООО "ЭТМС" 1992-2024. Все права защищены

2. посылка величины "7" с параметром "0" (текущему каналу виброметра задаётся 7-ой канала сервера);
3. посылка величины "1" с параметром "1" (текущему каналу виброметра задаётся временя усреднения 1 сек);
4. посылка величины "1" с параметром "2" (текущему каналу виброметра задаётся расчет СКЗ);
5. посылка величины "2" с параметром "7" (текущему каналу виброметра задаётся полосовой фильтр типа 2 - от 3 Гц до 10000 Гц).

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 0 | Установка порядкового номера канала (от 0 до (количество каналов - 1))<br>В многоканальном режиме установка номера канала сервера для текущего номера канала виброметра |
| 1 | Установка времени усреднения:<br>0 – быстро (0.1 секунды)<br>1 – медленно (1 секунда)<br>2 – очень медленно (10 секунд)<br>В многоканальном режиме установка времени усреднения для текущего номера канала виброметра |
| 2 | Установка типа рассчитываемого значения:<br>0 – пиковое значение<br>1 – среднеквадратичное значение<br>2 – амплитудное значение<br>В многоканальном режиме установка типа для текущего номера канала виброметра |
| 3 | Установка порога защиты по ускорению<br>В многоканальном режиме не поддерживается |
| 4 | Установка порога защиты по скорости<br>В многоканальном режиме не поддерживается |
| 5 | Установка порога защиты по перемещению<br>В многоканальном режиме не поддерживается |
| 6 | Установка на включение контроля по превышению порога:<br>0 – выключение контроля по превышению порога<br>1 – включение контроля по превышению порога<br>В многоканальном режиме не поддерживается |
| 7 | Установка типа полосового фильтра (для частоты дискретизации используемого АЦП 25 кГц):<br>0 – 1 Гц - 200 Гц<br>1 – 10 Гц - 1000 Гц |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| | 2 – 3 Гц - 10000 Гц<br>3 – 1 Гц - 10 Гц<br>В многоканальном режиме установка типа для текущего номера канала вольтметра |
| 8 | Установка использования выхода сухого контакта:<br>0 – не использовать выход сухого контакта<br>1 – использования выхода сухого контакта<br>В многоканальном режиме не поддерживается |
| 9 | Установка длительности удержания события на сухом контакте (от 1 секунды до 100 секунд)<br>В многоканальном режиме не поддерживается |
| 10 | Установка выдачи данных по сухому контакту в виртуальный канал:<br>0 – не выдавать данные по сухому контакту в виртуальный канал<br>1 – выдача данных по сухому контакту в виртуальный канал<br>В многоканальном режиме не поддерживается |
| 11 | Установка выдачи данных по сухому контакту через цифровой выход:<br>0 – не выдавать данные по сухому контакту через цифровой выход<br>1 – выдача данных по сухому контакту через цифровой выход<br>В многоканальном режиме не поддерживается |
| 12 | Установка номера бита цифрового порта дл я выдачи данных по сухому контакту (от 0 до 7)<br>В многоканальном режиме не поддерживается |
| 13 | Установка флага необходимости расчёта данных по скорости |
| 14 | Установка флага необходимости расчёта данных по перемещению |
| -1 | Перевод программы в многоканальный режим работы (без возможности вернуться в одноканальный режим работы). Установка количества каналов виброметра. Нулевой канал вольтметра становится текущим. Максимальное кол-во каналов вольтметра - 128. |
| -2 | Многоканальный режим работы. Установка текущего номера канал виброметра для задания его параметров (номер канала сервера, код времи усреднения, тип полосового фильтра и тип рассчитываемого значения) |
| -3 | Многоканальный режим работы. Номер канала виброметра, работа с данными (считывание и обработка) которого приостанавливается. При этом сам канал виброметра не уничтожается |
| -4 | Многоканальный режим работы. Номер канала виброметра, работа с данными (считывание и обработка) которого возобнавляется после |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| | приостановки |
| -5 | Многоканальный режим работы. Добавление одного нового канала вольтметра (если не превышено максимальное число каналов). Значение *value* - это номер канала сервера, который будет задан новому каналу виброметра, который автоматически становится текущим |

### Мультиметр Agilent_HP34401A (Agilent_HP34401A.exe)

***Использование примера требует подключенного ZET-устройства, содержащего ЦАП, мультиметра Agilent 34401a, и опции ПО ZETLab Внешние приборы***

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 0 | Установка режима измерений:<br>0 – вольтметр постоянного тока<br>1 – вольтметр переменного тока<br>2 – амперметр постоянного тока<br>3 – амперметр переменного тока<br>4 – частотомер<br>5 – омметр |
| 1 | Установка времени усреднения:<br>0 – медленно (1 секунда)<br>1 – очень медленно (10 секунд) |
| 2 | Установка типа представления расчета:<br>0 – линейный масштаб (в единицах измерения)<br>1 – логарифмический масштаб (в децибелах) |

### Блок питания LPS305 (LPS305.exe)

***Использование примера требует подключенного ZET-устройства, содержащего ЦАП и блока питания LPS305, и опции ПО ZETLab Внешние приборы***

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 0 | Установка ограничения по напряжению на первом канале |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 1 | Установка ограничения по напряжению на втором канале |
| 2 | Установка ограничения по току на первом канале |
| 3 | Установка ограничения по току на втором канале |
| 4 | Установка режима привязки:<br>0 – независимая работа каналов<br>1 – слежение за первым каналом |
| 5 | Установка включения звукового сигнала:<br>0 – выключение звукового сигнала<br>1 – включение звукового сигнала |
| 6 | Установка состояния питания цифрового выхода:<br>0 – цифровой выход отключен<br>1 – питание 3.3 В<br>2 – питание 5 В |
| 7 | Установка компенсация выхода |
| 8 | Установка состояния выходов:<br>0 – выключение выходов<br>1 – включение выходов |

## Блок питания PPE3323 (PPE3323.exe)

***Использование примера требует подключенного ZET-устройства, содержащего ЦАП и блока питания PPE3323, и опции ПО ZETLab Внешние приборы***

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 0 | Установка ограничения по напряжению на первом канале |
| 1 | Установка ограничения по напряжению на втором канале |
| 2 | Установка ограничения по току на первом канале |
| 3 | Установка ограничения по току на втором канале |
| 4 | Установка режима привязки:<br>0 – независимая работа каналов<br>1 – слежение за первым каналом |
| 6 | Установка состояния питания цифрового выхода: |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| | 1 – питание 3.3 В<br>2 – питание 5 В |
| 7 | Установка защиты выхода от перегрузки (OCP):<br>0 – защита выхода от перегрузки выключена<br>1 – защита выхода от перегрузки включена |
| 8 | Установка состояния выходов:<br>0 – выключение выходов<br>1 – включение выходов |
| 9 | Установка последовательного соединения выходов:<br>0 – последовательное соединение выходов выключено<br>1 – последовательное соединение выходов включено |
| 10 | Установка разблокировки выхода после срабатывания схемы защиты от перенапряжения (аргумент может принимать любые значения) |
| 11 | Установка уровня защиты от перенапряжения на первом канале |
| 12 | Установка уровня защиты от перенапряжения на втором канале |
| 13 | Установка уровня защиты от перенапряжения при последовательном соединении выходов |
| 14 | Установка ограничения по напряжению при последовательном соединении выходов |
| 15 | Установка ограничения по току при последовательном соединении выходов |

## Блок питания PSM2010 (PSM2010.exe)

***Использование примера требует подключенного ZET-устройства, содержащего ЦАП и блока питания PSM2010, и опции ПО ZETLab Внешние приборы***

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 0 | Установка ограничения по напряжению |
| 1 | Установка ограничения по току |
| 2 | Установка состояния выходов:<br>0 – выключение выходов<br>1 – включение выходов |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 3 | Установка защиты выхода от перегрузки (OCP):<br>0 – защита выхода от перегрузки выключена<br>1 – защита выхода от перегрузки включена |
| 4 | Установка защиты выхода от перенапряжения (OVP):<br>0 – защита выхода от перенапряжения выключена<br>1 – защита выхода от перенапряжения включена |
| 5 | Установка рабочего диапазона:<br>0 – 8V, 20A<br>1 – 20V, 10A |
| 6 | Установка уровня защиты по напряжению |
| 7 | Установка уровня защиты от перегрузки по току |
| 8 | Установка воспроизведения профиля:<br>0 – выключение воспроизведения профиля<br>1 – включение воспроизведения профиля |
| 9 | Установка количества циклов воспроизведения профиля (от 0 до 99999, 0 – бесконечный цикл) |
| 10 | Установка задержки воспроизведения профиля (от 1 до 35999, одна единица равна 100 мс) |
| 11 | Установка номера ячейки памяти, с которой начинается воспроизведение профиля (от 0 до 99) |
| 12 | Установка номера ячейки памяти, на которой заканчивается воспроизведение профиля (от 0 до 99) |
| 13 | Установка периодического опроса статуса прибора:<br>0 – выключение периодического опроса статуса прибора<br>1 – включение периодического опроса статуса прибора |

## 5.3.меню Отображение

Содержит перечень программ из вкладки меню Отображение из ZETLab, которыми можно управлять через Unit или Unit-2.

© ООО "ЭТМС" 1992-2024. Все права защищены

## Многоканальный осциллограф (OscGraph.exe)

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 0 | Установка индекса текущего отображаемого окна (от 0 до 7, либо от 0 до (количество каналов - 1), если количество каналов меньше 8) |
| 1 | Установка порядкового номера канала по заданному индексу (от 0 до (количество каналов - 1)) |
| 2 | Установка частоты обновления данных:<br>0 – быстро (0.1 секунды)<br>1 – медленно (1 секунда) |
| 3 | Установка количества отображаемых осциллограм (от 1 до 8, либо от 1 до количества каналов, если количество каналов меньше 8):<br>0 – 1 канал<br>1 – 2 канала<br>2 – 3 канала<br>3 – 4 канала<br>4 – 5 каналов<br>5 – 6 каналов<br>6 – 7 каналов<br>7 – 8 каналов |
| 4 | Установка декады (частотного диапазона):<br>0 – от 0 до (частота дискретизации / 1)<br>1 – от 0 до (частота дискретизации / 10)<br>2 – от 0 до (частота дискретизации / 100)<br>3 – от 0 до (частота дискретизации / 1000)<br>4 – от 0 до (частота дискретизации / 10000) |
| 5 | Установка интервала отображения (0,001 - …) сек. |
| 6 | Установка на запись данных в файл (аргумент может принимать любое значение) |
| 7 | Установка на получение размера массива данных по операции *Read(...)* (аргумент может принимать любое значение). После задания этого параметра необходимо дождаться события готовности *Ready(...)* и затем прочитать размер массива данных операцией *Read(...)* |
| 8 | Установка на получение массива данных по операции *Read(...)* (аргумент принимает значение индекса требуемого массива данных, т.е. от 0 до 7, либо от 0 до (количество каналов - 1), если количество каналов меньше 8). После задания этого параметра необходимо дождаться события готовности *Ready(...)* и затем прочитать массив данных операцией *Read(...)* |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 9 | Установка режима отображения данных:<br>0 – каждый график в своем окне<br>1 – совмещенный режим отображения |
| 10 | Нижняя граница отображения данных по оси Y в единицах измерения |
| 11 | Верхняя граница отображения данных по оси Y в единицах измерения |
| 12 | Установка режима маркеры неинтерполированных точек:<br>0 – скрыть<br>1 – показать |
| 13 | Установка свечение:<br>0 - отключить,<br>1 - включить |
| 14 | Установка автомасштаба (аргумент может принимать любое значение) |
| 15 | Установка постоянного автомасштаба:<br>0 - отключить,<br>1 - включить.<br>Real-time автомасштаб в многоканальном осциллографе, чтобы сигнал, отображаемый в текущем окне осциллографа всегда занимал всё поле. Это особенно актуально при просмотре оконных данных. |
| 16 | Установка скрытых настроек:<br>0 - отключить,<br>1 - включить.<br>При уменьшении размера Многоканального осциллографа. Появилась возможность отключить часть ненужной информации в данный момент. В основном нужен интервал и выбор частоты, остальная информация нужна по необходимости. Тогда можно было бы отсмотреть больше полезной информации с формой сигнала. |
| 17 | Установка автоматического отображения новых каналов.<br>0 - отключить,<br>1 - включить. |
| 255 | Установка на включение многоканального осциллографа (аргумент может принимать любое значение) |
| 127 | Установка на выключение многоканального осциллографа (аргумент может принимать любое значение) |

© ООО "ЭТМС" 1992-2024. Все права защищены

## XYZ-осциллограф (XYOscGraph.exe)

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 0 | Установка порядкового номера канала компоненты X (от 0 до (количество каналов - 1)) |
| 1 | Установка порядкового номера канала компоненты Y (от 0 до (количество каналов - 1)) |
| 2 | Установка порядкового номера канала компоненты Z (от 0 до (количество каналов - 1)) |
| 4 | Установка декады (частотного диапазона):<br>0 – от 0 до (частота дискретизации / 1)<br>1 – от 0 до (частота дискретизации / 10)<br>2 – от 0 до (частота дискретизации / 100)<br>3 – от 0 до (частота дискретизации / 1000)<br>4 – от 0 до (частота дискретизации / 10000) |
| 5 | Установка интервала отображения, с |
| 6 | Установка на запись данных в файл (аргумент может принимать любое значение) |
| 7 | Установка на получение размера массива данных по операции *Read(...)* (аргумент может принимать любое значение). После задания этого параметра необходимо дождаться события готовности *Ready(...)* и затем прочитать размер массива данных операцией *Read(...)* |
| 8 | Установка на получение массива данных по операции *Read(...)* (аргумент принимает значение индекса требуемого массива данных, т.е. от 0 до 2). После задания этого параметра необходимо дождаться события готовности *Ready(...)* и затем прочитать массив данных операцией *Read(...)* |
| 255 | Установка на включение XYZ-осциллографа (аргумент может принимать любое значение) |
| 127 | Установка на выключение XYZ-осциллографа (аргумент может принимать любое значение) |

## XY-плоттер (XYPlotter.exe)

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 0 | Установка режима отображения: |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| | 0 - XT<br>1 - YT<br>2 - XY<br>3 - XYT |
| 1 | Установка времени измерения (10, 30, 60, 100, 300, 600 или 1000 сек) |
| 2 | Установка на "Неограниченного времени" (1 Вкл \| 0 Выкл) |
| 3 | Установка Автомасштаба (1 Вкл \| 0 Выкл) |
| 4 | Установка Скорости обновления данных:<br>0 - Быстро 0.1 с<br>1 - Медленно 1 с |
| 5 | Выбор программы для оси X:<br>0 - Вольтметр переменного тока<br>1 - Вольтметр постоянного тока<br>2 - Вольтметр селективный<br>3 - Частотомер<br>4 - Фазометр |
| 6 | Выбор программы для оси Y:<br>0 - Вольтметр переменного тока<br>1 - Вольтметр постоянного тока<br>2 - Вольтметр селективный<br>3 - Частотомер<br>4 - Фазометр |
| 7 | Выбор канала по X (аргумент принимает номер канала начиная с 0) |
| 8 | Выбор канала по Y (аргумент принимает номер канала начиная с 0) |
| 9 | Выбор второго канала по X. Только для Фазометра (аргумент принимает номер канала начиная с 0) |
| 10 | Выбор второго канала по Y. Только для Фазометра (аргумент принимает номер канала начиная с 0) |
| 11 | Нажатие кнопки Старт/Стоп |
| 12 | Нажатие кнопки Сброс |
| 13 | Нажатие кнопки Сброс в ноль/Отключение сброса в ноль |

© ООО "ЭТМС" 1992-2024. Все права защищены

## 5.4.меню Генераторы

Содержит перечень программ из вкладки меню Генераторы из ZETLab, которыми можно управлять через Unit или Unit-2.

### Генератор сигналов (DAC_OCX.exe)

| **Параметр (*param*)** | **Аргумент (*value*)** |
|---|---|
| 0 | Установка типа генерируемого сигнала:<br>1 – синусоидальный сигнал<br>2 – радиоимпульсный сигнал<br>3 – шум<br>4 – линейная частотная модуляция<br>5 – логарифмическая частотная модуляция<br>6 – импульсный сигнал<br>7 – сигнал из файла<br>8 – второй синусоидальный сигнал<br>9 – амплитудная модуляция<br>10 – частотная модуляция<br>11 – пилообразный сигнал<br>12 – сигнал с входного или виртуального канала<br>13 – сигнал кодов Баркера<br>14 – линейная амплитудная модуляция<br>15 – логарифмическая амплитудная модуляция<br>16 – КИХ-фильтр |
| 1 | **//Параметры синусоидального сигнала (3 параметра)**<br>Установка частоты синусоидального сигнала (от 0.01 Гц до (частота дискретизации ЦАП / 2)) и начальной частоты линейной частотной модуляции (от 0.01 Гц до конечной частоты линейной частотной модуляции) |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| | и логарифмической частотной модуляции (от 0.01 Гц до конечной частоты логарифмической частотной модуляции) |
| 2 | Установка уровня синусоидального сигнала, линейной частотной модуляции и логарифмической частотной модуляции (от 0 В до максимально допустимого уровня сигнала ЦАП) |
| 5 | Установка смещения постоянной составляющей синусоидального сигнала (от минус максимально допустимого уровня сигнала ЦАП до максимально допустимого уровня сигнала ЦАП), В5 |
| 3 | **//Параметры линейной частотной модуляции (6 параметра)**<br>Установка цикличности линейной частотной модуляции:<br>0 – однократное воспроизведение сигнала<br>1 – многократное воспроизведение сигнала |
| 4 | Скорость развертки линейной частотной модуляции не может быть менее 1 Гц/с, в соответствии с этим, скорость развертки должна быть такой, что б удовлетворить этому неравенству.<br>- формула расчёта, следующая интервал = (частота конечная - частота начальная) / длительность, с |
| 6 | Установка конечной частоты линейной частотной модуляции (от начальной частоты линейной частотной модуляции до (частота дискретизации ЦАП / 2)) |
| 1 | Установка начальной частоты линейной частотной модуляции (от 0.01 Гц до конечной линейной частотной модуляции) |
| 2 | Установка уровня частоты линейной частотной модуляции (от 0 В до максимально допустимого уровня сигнала ЦАП) |
| 1 | **//Параметры логарифмической частотной модуляции (6 параметра)**<br>Установка начальной частоты логарифмической частотной модуляции (от 0.01 Гц до конечной линейной частотной модуляции) |
| 6 | Установка конечной частоты логарифмической частотной модуляции (от начальной частоты линейной частотной модуляции до (частота дискретизации ЦАП / 2)) |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| 2 | Установка уровня частоты логарифмической частотной модуляции (от 0 В до максимально допустимого уровня сигнала ЦАП) |
| 4 | Скорость развертки логарифмической частотной модуляции не может быть менее 1 окт/мин, в соответствии с этим, скорость развертки должна быть такой, что б удовлетворить этому неравенству.<br>- формула расчёта следующая интервал = (частота конечная - частота начальная) / длительность, с для логарифмической интервал = (частота конечная / частота начальная) / log(2.0) / длительность, с * 60.0 |
| 3 | Установка цикличности логарифмической частотной модуляции:<br>0 – однократное воспроизведение сигнала<br>1 – многократное воспроизведение сигнала |
| 8 | **//Параметры радиоимпульсного сигнала (6 параметра)**<br>Установка частоты, несущей радиоимпульсного сигнала (от частоты следования радиоимпульсного сигнала до (частота дискретизации ЦАП / 2)), Гц |
| 9 | Установка амплитуды радиоимпульсного сигнала (от 0 В до максимально допустимого уровня сигнала ЦАП) |
| 10 | Установка частоты следования радиоимпульсного сигнала (от 0.01 Гц до частоты заполнения радиоимпульсного сигнала) |
| 11 | Установка длительности радиоимпульсного сигнала в периодах частоты заполнения (от 0 до (частота заполнения радиоимпульсного сигнала / частота следования радиоимпульсного сигнала)) |
| 45 | Установка цикличности радиоимпульсного сигнала<br>0 – однократное воспроизведение сигнала<br>1 – многократное воспроизведение сигнала |
| 46 | Установка генерации окна Тьюки |
| 47 | Установка задержки сигнала |
| 12 | **//Параметры уровня шума (5 параметров)**<br>Установка уровня шума (от 0 В до (максимально допустимый уровень сигнала ЦАП / 5)) |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| 13 | Установка начальной частоты шума (от 0.01 Гц до конечной частоты шума) |
| 14 | Установка конечной частоты шума (от начальной частоты шума до (частота дискретизации ЦАП / 2)), Гц |
| 15 | Установка типа шума:<br>0 – белый шум;<br>1 – полосовой шум;<br>2 – розовый шум;<br>3 – красный шум;<br>3 – детерминированный шум;<br>4 – предбеленный. |
| 55 | Установка кутозиса (от 2 до 20) |
| 16 | **//Параметры частоты импульсного сигнала (4 параметра)**<br>Установка частоты импульсного сигнала (от 0.01 Гц до (частота дискретизации ЦАП / 2)) |
| 17 | Установка амплитуды импульсного сигнала (от 0 В до максимально допустимого уровня сигнала ЦАП) |
| 18 | Установка смещения постоянной составляющей импульсного сигнала (от минус максимально допустимого уровня сигнала ЦАП до максимально допустимого уровня сигнала ЦАП), В |
| 19 | Установка скважности импульсного сигнала (от 0 до 1) |
| 20 | **//Параметры синусоидального сигнала (3 параметра)**<br>Установка частоты второго синусоидального сигнала (от 0.01 Гц до (частота дискретизации ЦАП / 2)) |
| 21 | Установка уровня второго синусоидального сигнала (от 0 В до максимально допустимого уровня сигнала ЦАП) |
| 22 | Установка смещения постоянной составляющей второго синусоидального сигнала (от минус максимально допустимого уровня сигнала ЦАП до |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| | максимально допустимого уровня сигнала ЦАП), В |
| 23 | **//Параметры амплитудной модуляции (4 параметра)**<br>Установка несущей частоты сигнала амплитудной модуляции, Гц |
| 24 | Установка амплитуды сигнала амплитудной модуляции (от 0 В до максимально допустимого уровня сигнала ЦАП) |
| 25 | Установка частоты модуляции сигнала амплитудной модуляции, Гц |
| 26 | Установка глубины модуляции сигнала амплитудной модуляции (от 0 до 1) |
| 27 | **//Параметры частотной модуляции (4 параметра)**<br>Установка несущей частоты сигнала частотной модуляции, Гц |
| 28 | Установка амплитуды сигнала частотной модуляции (от 0 В до максимально допустимого уровня сигнала ЦАП) |
| 29 | Установка частоты модуляции сигнала частотной модуляции, Гц |
| 30 | Установка глубины модуляции сигнала частотной модуляции (от 0 до 1) |
| 31 | **//Параметры пилообразного сигнала (5 параметра)**<br>Установка частоты пилообразного сигнала (от 0.01 Гц до (частота дискретизации ЦАП / 2)) |
| 32 | Установка амплитуды пилообразного сигнала (от 0 В до максимально допустимого уровня сигнала ЦАП) |
| 33 | Установка смещения постоянной составляющей пилообразного сигнала (от минус максимально допустимого уровня сигнала ЦАП до максимально допустимого уровня сигнала ЦАП), В |
| 34 | Установка типа пилы:<br>0 – нарастающий<br>1 – ниспадающий<br>2 – треугольный |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| 35 | Установка цикличности пилообразного сигнала<br>0 – однократное воспроизведение сигнала<br>1 – многократное воспроизведение сигнала |
| 37 | **//Параметры пилообразного сигнала (6 параметров)**<br>Установка частоты сигнала кодов Баркера (от 0.01 Гц до (частота дискретизации ЦАП / 2)) |
| 38 | Установка количества периодов в дискрете сигнала кодов Баркера (от 0 до 5000) |
| 39 | Установка амплитуды сигнала кодов Баркера (от 0 В до максимально допустимого уровня сигнала ЦАП) |
| 40 | Установка периода повторения посылок сигнала кодов Баркера (от 0.1 Гц до 5000 Гц) |
| 41 | Установка кода Баркера:<br>0 – 2<br>1 – 3<br>2 – 4<br>3 – 5<br>4 – 7<br>5 – 11<br>6 – 13 |
| 42 | Установка цикличности сигнала кодов Баркера<br>0 – однократное воспроизведение сигнала<br>1 – многократное воспроизведение сигнала |
| 43 | **//Параметры входного канала (2 параметров)**<br>Установка порядкового номера канала для генерации сигнала с канала (от 0 до (количество каналов - 1)) |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| 44 | Установка коэффициента усиления (ослабления) канала файл для генерации сигнала с канала (от 0.001 до 99.9) |
| 48 | **//Параметры линейной амплитудной модуляции (4 параметра)**<br>ЛинАМ частота несущего сигнала (0.01 - частота дискретизации ЦАП / 2 Гц) |
| 49 | ЛинАМ конечная амплитуда (0.0001 - максимально допустимого уровня сигнала ЦАП, В) |
| 3 | Установка цикличности сигнала линейной амплитудной модуляции:<br>0 – однократное воспроизведение сигнала<br>1 – многократное воспроизведение сигнала |
| 50 | ЛинАМ скорость развертки (0.0001 - 9.999 В/с) |
| 51 | **//Параметры логарифмической амплитудной модуляции (5 параметров)**<br>ЛогАМ частота несущего сигнала (0.01 - частота дискретизации ЦАП / 2 Гц) |
| 52 | ЛогАМ начальная амплитуда (0.0001 - максимально допустимого уровня сигнала ЦАП, В) |
| 53 | ЛогАМ конечная амплитуда (0.0001 - максимально допустимого уровня сигнала ЦАП, В) |
| 3 | Установка цикличности сигнала логарифмической амплитудной модуляции:<br>0 – однократное воспроизведение сигнала<br>1 – многократное воспроизведение сигнала |
| 54 | ЛогАМ скорость развертки (0.10 - .69 дБ/с) |
| 55 | Куртозис шума используется для Белого, Предбеленного и полосовых шумов, который изменяется от 2 до 20. |
| 56 | Установление флага генерации синуса с линейной частотной модуляцией в обе стороны изменения частоты: |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| | 0 – отключить генерацию синуса с линейной частотной модуляцией в обе стороны изменения частоты<br>1 – включить генерацию синуса с линейной частотной модуляцией в обе стороны изменения частоты |
| 57 | Плавное нарастание и плавное затухание сигнала с линейной частотной модуляцией:<br>0 – отключить плавное затухание сигнала с линейной частотной модуляцией<br>1 – включить плавное затухание сигнала с линейной частотной модуляцией |
| | |
| 70 | **//Параметры сигнала из файла (3 параметра)**<br>Установка коэффициента усиления (ослабления) сигнала из файла (от 0.001 до 999.9) |
| 71 | Установка цикличности сигнала из файла<br>0 – однократное воспроизведение сигнала<br>1 – многократное воспроизведение сигнала |
| 72 | Установка имени файла сигнала из файла в wchar, но в TestUnit параметры 72 и 73 в wchar. |
| 73 | Установка имени файла сигнала из файла в char, но в TestUnit параметры 72 и 73 в wchar. |
| | |
| 74 | **//Параметры коррекции (1 параметр)**<br>Установка имени файла импульсной характеристики для КИХ-фильтра в wchar, но в TestUnit параметры 74 и 75 в wchar. |
| 75 | Установка имени файла импульсной характеристики для КИХ-фильтра в char, но в TestUnit параметры 74 и 75 в wchar. |
| | |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| 125 | Установка идентификатора канала генератора. После этого через юнит будет возвращён номер канала генератора из списка доступных генераторов сервера, который был установлен. Если канала генератора с указанным идентификатором нет на сервере, то будет установлен первый доступный канал генератора. |
| 126 | Установка порядкового номера генератора (от 0 до (количество генераторов - 1)). После этого через юнит будет возвращён номер канала генератора из списка доступных генераторов сервера, который был установлен. |
| 128 | Установка на выключение текущего вида сигнала, установленного параметром 0 (аргумент может принимать любое значение) |
| 127 | Установка на выключение генератора (аргумент может принимать любое значение) |
| 256 | Установка на включение текущего вида сигнала, установленного параметром 0 (аргумент может принимать любой значение) |
| 255 | Установка на включение генератора (аргумент может принимать любое значение) |

## Синхронный генератор (SynchroChanDac.exe)

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 0 | Установка количества каналов генераторов (от 1 до 10, либо от 1 до (количество генераторов - 1)) |
| 1 | Установка частоты сигнала (от 0 до (частота дискретизации ЦАП / 2)), Гц |
| 2 | Установка индекса текущего генератора (от 0 до (количество каналов генераторов - 1)) |
| 3 | Установка номера текущего канала генератора (от 0 до (количество генераторов - 1)) |
| 4 | Установка типа сигнала текущего канала генератора:<br>0 – синусоидальный сигнал |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| | 1 – импульсный сигнал<br>2 – сигнал из файла<br>3 – логарифмическая частотная модуляция<br>4 – радиоимпульсный сигнал |
| 10 | Установка уровня синусоидального сигнала текущего канала генератора (от 0 В до максимально допустимого уровня сигнала ЦАП) |
| 11 | Установка смещения постоянной составляющей синусоидального сигнала текущего канала генератора (от минус максимально допустимого уровня сигнала ЦАП до максимально допустимого уровня сигнала ЦАП), В |
| 12 | Установка смещения фазы синусоидального сигнала текущего канал генератора (от 0° до 360°) |
| 20 | Установка амплитуды импульсного сигнала текущего канала генератора (от 0 В до максимально допустимого уровня сигнала ЦАП) |
| 21 | Установка смещения постоянной составляющей импульсного сигнала текущего канала генератора (от минус максимально допустимого уровня сигнала ЦАП до максимально допустимого уровня сигнала ЦАП), В |
| 22 | Установка скважности импульсного сигнала текущего канала генератора (от 0 до 1) |
| 23 | Установка смещения фазы импульсного сигнала текущего канала генератора (от 0° до 360°) |
| 30 | Установка имени файла сигнала из файла текущего канала генератора |
| 31 | Установка коэффициента усиления (ослабления) сигнала из файла текущего канала генератора (от 0.001 до 999.9) |
| 32 | Установка времени задержки начала воспроизведения сигнала из файла текущего канала генератора (от 0 с до 60 с) |
| 33 | Установка сжатия (компрессии) сигнала из файла текущего канала генератора (от -1000 % до 90 %) |
| 34 | Установка цикличности сигнала из файла текущего канала генератора:<br>0 – однократное воспроизведение сигнала<br>1 – многократное воспроизведение сигнала |
| 35 | Установка места начала воспроизведения сигнала из файла текущего канала генератора (от 0 с до (длительность файла * сжатие + время задержки) с) |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 40 | Установка уровня сигнала логарифмической частотной модуляции текущего канала генератора (от 0 В до максимально допустимого уровня сигнала ЦАП) |
| 41 | Установка начальной частоты сигнала логарифмической частотной модуляции текущего канала генератора (от 0.01 Гц до (частота дискретизации ЦАП / 2) Гц) |
| 42 | Установка конечной частоты сигнала логарифмической частотной модуляции текущего канала генератора (от 0.01 Гц до (частота дискретизации ЦАП / 2) Гц) |
| 43 | Установка длительности сигнала логарифмической частотной модуляции текущего канал генератора, с |
| 44 | Установка скорости развертки сигнала логарифмической частотной модуляции текущего канала генератора, окт/мин |
| 45 | Установка начальной фазы, град. |
| 46 | Установка времени выдержки на начальной частоте, с |
| 50 | Установка уровня радиоимпульсного сигнала текущего канала генератора (от 0 В до максимально допустимого уровня сигнала ЦАП) |
| 51 | Установка частоты заполнения радиоимпульсного сигнала текущего канала генератора (от частоты следования радиоимпульсного сигнала до (частота дискретизации ЦАП / 2)), Гц |
| 52 | Установка частоты следования радиоимпульсного сигнала текущего канала генератора (от 0.01 Гц до частоты заполнения радиоимпульсного сигнала) |
| 53 | Установка смещения нуля радиоимпульсного сигнала текущего канала генератора (от 0 В до максимально допустимого уровня сигнала ЦАП) |
| 54 | Установка количества периодов радиоимпульсного сигнала текущего канала (от 0 до 10) |
| 55 | Установка дополнительного уровня радиоимпульсного сигнала текущего канала генератора (от 0 В до максимально допустимого уровня сигнала ЦАП) |
| 56 | Установка дополнительной фазы, град. |
| 127 | Установка на выключение синхронного генератора (аргумент может принимать любое значение) |
| 255 | Установка на включение синхронного генератора (аргумент может принимать любое значение) |

© ООО "ЭТМС" 1992-2024. Все права защищены

## 5.5.меню Регистрация

Содержит перечень программ из вкладки меню Регистрация из ZETLab, которыми можно управлять через Unit или Unit-2.

### Запись сигналов (SignalWriter.exe)

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 0 | Установка следующей директории для записи сигнала (аргумент может принимать любое значение) |
| 1 | Установка преамбулы записи (от 0 с до 600 с) |
| 2 | Установка длительности записи (от 10 с до 84600 с) |
| 3 | Установка каналов для записи:<br>номер > 0 – включить канал для записи<br>номер < 0 – выключить канал для записи |
| 4 | Установка текущей директории для записи |
| 5 | Установка типа записи:<br>0 – детерминированная по времени запись<br>1 – непрерывная запись |
| 6 | Установка на обновление списка каналов и выбор для записи каналов АЦП (аргумент может принимать любое значение) |
| 7 | Добавить комментарии к записи сигналов |
| 8 | Установка измененной директории, подаем строку куда хотим писать прерывную запись, к поданной строке допишется \sXXXX_XXXX \(год, месяц, день и т.д.) |
| 9 | Сохраняет файл с комментарием при включенной записи |
| 127 | Установка на останов записи (аргумент может принимать любое значение) |
| 255 | Установка на начало записи (аргумент может принимать любое значение) |

***Примечание:*** Для задания новой директории программе SignalWriter.exe, необходимо сначала создать директорию, а затем вызвать функцию SetParamString.
Пример:
char dirname[255] = "C:\ZETLAB\newdir\";
long reserror = m_unit.SetParamString(4, &dirname[0]); Воспроизведение сигналов (reader)

© ООО "ЭТМС" 1992-2024. Все права защищены

## Воспроизведение сигналов (reader.exe)

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 0 | Установка директории, в которой находятся сигналы для воспроизведения |
| 1 | Установка скорости воспроизведения сигналов:<br>0 – быстрая;<br>1 – нормальная. |
| 2 | Установка типа воспроизведения:<br>0 – воспроизведение записи из текущей директории;<br>1 – непрерывное воспроизведение. |
| 3 | Установка типа циклического воспроизведения:<br>0 – флаг снимается с циклического воспроизведения записи;<br>1 – флаг устанавливается на циклическое воспроизведение записи. |
| 127 | Установка на останов воспроизведения сигналов (аргумент может принимать любое значение) |
| 255 | Установка на начало воспроизведения сигналов (аргумент может принимать любое значение) |
| 256 | Установка на приостановку воспроизведения сигналов (аргумент может принимать любое значение) |

## Многоканальный самописец (multiSWvm.exe)

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 0 | Задание номера текущего канала (-1 допустим). Параметр типа long. |
| 1 | Задание интервала отображения, сек. Параметр типа double. От 10 до 86400 (сутки). |
| 2 | Задание времени усреднения, сек. Параметр типа double. От 0,1 до 3600 (час). |
| 3 | Задание кода широкого усреднения. Параметр типа long:<br>0 – усреднение x5;<br>1 – усреднение x10;<br>2 – усреднение x20. |
| 4 | Задание флага мониторинга объекта. Параметр типа long. 0 или 1. |
| 5 | Задание флага записи трендов сигналов. Параметр типа long. 0 или 1. |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| 6 | Задание имени объекта мониторинга объекта. Строка символов типа WCHAR с завершающим нулём. |
| 7 | Задание кода типа обработки текущего канала. Параметр типа long:<br>0 – двойное дифференцирование исходного сигнала;<br>1 – дифференцирование исходного сигнала;<br>2 – без обработки;<br>3 – интегрирование исходного сигнала;<br>4 – двойное интегрирование исходного сигнала. |
| 8 | Задание кода типа рассчитываемого параметра по данным текущего канала. Параметр типа long:<br>0 – среднеквадратичное значение (в единицах измерения);<br>1 – амплитудное значение (в единицах измерения);<br>2 – пиковое значение (в единицах измерения);<br>3 – размах (в единицах измерения);<br>4 – отклонение от 0 (в единицах измерения);<br>5 – среднее значение сигнала (в единицах измерения);<br>6 – среднее значение сигнала со сбросом 0(в единицах измерения);<br>7 – частота изменения сигнала, Гц;<br>8 – частота изменения сигнала, обороты в мин;<br>9 – куртозис;<br>10 – формула куртозис по скз;<br>11 – формула ско по скз;<br>12 – формула sta_lta(AC) по амплитуде сигнала;<br>13 – формула sta_lta(DC) по постоянной составляющей сигнала;<br>14 – инклинометр, градусы (если размерность сигнала после обработки является ускорением);<br>15 – баллы по шкале MSK-64 (если размерность сигнала после обработки является ускорением). |
| 9 | Задание частоты среза ФВЧ текущего канала, Гц. |
| 10 | Задание частоты среза ФНЧ текущего канала, Гц. |
| 11 | Задание порогового значения 1 текущего канала. Параметр типа double (в единицах измерения). |
| 12 | Задание порогового значения 2 текущего канала. Параметр типа double (в единицах измерения). |
| 13 | Задание порогового значения 3 текущего канала. Параметр типа double (в единицах измерения). |
| 14 | Задание порогового значения 4 текущего канала. Параметр типа double (в единицах измерения). |
| 15 | Задание типа событий текущего канала при использовании мониторинга. Параметр типа long. От 0 до 5 включительно. |
| 16 | Задание цвета графика текущего канала. Параметр типа COLORREF. |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| 17 | Эмуляция клика по кнопке «Сброс 0». |
| 18 | Задание флага "Мониторинг потери канала". Параметр типа long. 0 или 1. |
| 19 | Задание флага "OPC UA". Параметр типа long. 0 или 1. |
| 20 | Задание флага "Визуализация 3D". Параметр типа long. 0 или 1. |
| -1 | Задание количества рассчитываемых параметров. Параметр типа long. От 1 до 400 включительно. С параметрами новым каналам, заданным значениями по умолчанию. |
| -2 | Задание индекса текущего параметра. Параметр типа long. От 0 до кол-ва параметров -1. |
| -5 | Добавление нового рассчитываемого параметра по данным канала с номером value. Параметр типа long. От 1 до кол-ва каналов ZetServer без 1. |

## 5.6.меню Автоматизация

Содержит перечень программ из вкладки меню Автоматизация из ZETLab, которыми можно управлять через Unit или Unit-2.

### Регулятор (Regulator.exe)

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 0 | Установка порядкового номера входного канала (от 0 до (количество каналов - 1)) |
| 1 | Установка порядкового номера выходного канала (от 0 до (количество генераторов - 1), если выбран аналоговый выход, либо от 0 до (количество цифровых линий - 1), если выбран цифровой выход) |
| 2 | Установка типа выходного сигнала:<br>0 – аналоговый выход<br>1 – цифровой выход |
| 3 | Установка инверсии выхода:<br>0 – выход без инверсии<br>1 – инверсный выход |
| 4 | Установка типа регулятора:<br>0 – релейный регулятор |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| | 1 – ПИД-регулятор |
| 5 | Установка использования пропорционального коэффициента ПИД-регулятора:<br>0 – запрет использования пропорционального коэффициента<br>1 – разрешение использования пропорционального коэффициента |
| 6 | Установка использования интегрального коэффициента ПИД-регулятора:<br>0 – запрет использования интегрального коэффициента<br>1 – разрешение использования интегрального коэффициента |
| 7 | Установка использования дифференциального коэффициента ПИД-регулятора:<br>0 – запрет использования дифференциального коэффициента<br>1 – разрешение использования дифференциального коэффициента |
| 8 | Установка значения пропорционального коэффициента ПИД-регулятора |
| 9 | Установка значения интегрального коэффициента ПИД-регулятора |
| 10 | Установка значения дифференциального коэффициента ПИД-регулятора |
| 11 | Установка периода квантования |
| 12 | Установка разрешения амплитудной модуляции:<br>0 – запрет амплитудной модуляции<br>1 – разрешение амплитудной модуляции |
| 13 | Установка разрешения широтно-импульсной модуляции:<br>0 – запрет широтно-импульсной модуляции<br>1 – разрешение широтно-импульсной модуляции |
| 15 | Установка длительности истории, с |
| 16 | Установка режима работы регулятора:<br>0 – работа по профилю<br>1 – удержание уровня |
| 17 | Установка значения уровня для удержания, ед. измерения |
| 18 | Установка частоты заполнения широтно-импульсной модуляции, Гц |
| 19 | Установка верхнего уровня выходного сигнала, мВ |
| 20 | Установка нижнего уровня выходного сигнала, мВ |
| 21 | Установка коридора допуска по входному каналу, ед. измерения |
| 22 | Установка максимальной скважности широтно-импульсной модуляции (от минимальной скважности широтно-импульсной модуляции до 100%) |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 23 | Установка минимальной скважности широтно-импульсной модуляции (от 0% до максимальной скважности широтно-импульсной модуляции) |
| 24 | Установка имени файла исполняемого профиля |
| 255 | Установка на включение регулятора (аргумент может принимать любое значение) |
| 127 | Установка на выключение регулятора (аргумент может принимать любое значение) |

### Арифмометр (ArithmoMeter.exe)

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 0 | Установка порядкового номера первого канала (от 0 до (количество каналов - 1)) |
| 1 | Установка порядкового номера второго канала (от 0 до (количество каналов - 1)) |
| 2 | Установка операции над каналами:<br>0 – сложение каналов<br>1 – вычитание каналов<br>2 – перемножение каналов<br>3 – деление каналов<br>4 – расчет максимального значения из двух каналов<br>5 – расчет минимального значения из двух каналов<br>6 – расчет среднего арифметического значения из двух каналов<br>7 – расчет модуля сигналов<br>8 – расчет среднего геометрического значения из двух каналов |
| 3 | Установка константы для умножения |
| 4 | Установка константы для сложения |

### Адаптивный фильтр 50 Гц (filtr50.exe)

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 0 | Установка порядкового номера канала (от 0 до (количество каналов - 1)) |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 1 | Установка времени оценки частоты сигнала:<br>0 – 10 с<br>1 – 20 с<br>2 – 50 с<br>3 – 100 с |
| 2 | Установка времени оценки амплитуды сигнала:<br>0 – 0.1 с<br>1 – 0.2 с<br>2 – 0.5 с<br>3 – 1 с<br>4 – 2 с<br>5 – 5 с<br>6 – 10 с |

## Фильтрация сигналов (filtrdiff.exe)

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 0 | Установка количества фильтров (от 1 до 100) |
| 10, 20, … 1000 | Установка порядкового номера первого … сотого фильтруемого исходного канала (от 0 до (количество каналов - 1)) |
| 11, 21, … 1001 | Установка типа первого … сотого фильтра:<br>0 – линейный<br>1 – дифференцирующий 1-го порядка<br>2 – дифференцирующий 2-го порядка<br>3 – интегрирующий 1-го порядка<br>4 – интегрирующий 2-го порядка |
| 12, 22, … 1002 | Установка разрешения работы фильтра верхних частот первого … сотого фильтра:<br>0 – выключение фильтра верхних частот<br>1 – включение фильтра верхних частот |
| 13, 23, … 1003 | Установка частоты среза фильтра верхних частот первого … сотого фильтра (от (частота дискретизации / 10000) до (частота дискретизации / 2)) Гц |
| 14, 24, … 1004 | Установка разрешения работы фильтра нижних частот первого … сотого фильтра: |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| | 0 – выключение фильтра нижних частот<br>1 – включение фильтра нижних частот |
| 15, 25, … 1005 | Установка частоты среза фильтра нижних частот первого … сотого фильтра (от (частота дискретизации / 10000) до (частота дискретизации / 2)) Гц |
| 16, 26, … 1006 | Установка разрешения работы детектора огибающей первого … сотого фильтра:<br>0 – выключение детектора огибающей<br>1 – включение детектора огибающей |
| 17, 27, … 1007 | Установка времени интегрирования детектора огибающей первого … сотого фильтра (от (100 / частота дискретизации) мс до (25000 / частота дискретизации) мс) |
| 18, 28, … 1008 | Установка на получение идентификатора виртуального канала первого … сотого фильтра по операции *Read(...)* (аргумент может принимать любое значение). После задания этого параметра необходимо дождаться события готовности *Ready(...)* и затем прочитать идентификатор виртуального канала первого … сотого фильтра операцией *Read(...)* |
| 19, 29, … 1009 | Установка имени первого … сотого фильтра |

Добавлены дополнительные параметры управления фильтром.

| | |
|---|---|
| 1011, 1021, ... , 2001 | Установка частотной коррекции.<br>Номер соответствует порядку в комбобоксе программы |
| 1012, 1022, ... , 2002 | Установка разрешения работы фильтра действительного резонатора первого … сотого фильтра<br>0 – выключение фильтра действительного резонатора<br>1 – включение фильтра действительного резонатора |
| 1013, 1023, ... , 2003 | Установка резонансной частоты фильтра действительного резонатора первого … сотого фильтра |
| 1015, 1025, ... , 2005 | Установка декремента затухания фильтра действительного резонатора первого … сотого фильтра |

© ООО "ЭТМС" 1992-2024. Все права защищены

## Формула (ZETFormula.exe)

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 0 | Установка количества каналов формулы (от 1 до 100) |
| 10, 20, … 1000 | Установка названия первого … сотого канала формулы |
| 11, 21, … 1001 | Установка единица измерения первого … сотого канала формулы. |
| 12, 22, … 1002 | Установка максимального уровня первого … сотого канала формулы в единицах измерения |
| 13, 23, … 1003 | Установка опорного значения для расчета децибел первого … сотого канала формулы |
| 14, 24, … 1004 | Установка выражения формулы первого … сотого канала формулы. |
| 15, 25, … 1005 | Установка минимального уровня первого … сотого канала формулы в единицах измерения |
| 16, 26, … 1006 | Установка частоты представления (дискретизации) первого … сотого канала формулы (от 1 до 4000000 Гц) |

## Синхронизация по GPS (Synchronization.exe)

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 0 | Установка номера последовательного входного порта или модуля АЦП для обмена с GPS-приемником (от 0 до (количество COM-портов + количество устройств, поддерживающих GPS - 1)) |
| 1 | Установка скорости приема данных NMEA потока:<br>0 – 57600 бит/с<br>1 – 38400 бит/с<br>2 – 19200 бит/с<br>3 – 9600 бит/с<br>4 – 4800 бит/с |
| 2 | Установка номера последовательного порта для вывода данных GPS (от 0 до (количество COM-портов - 1)) |

© ООО "ЭТМС" 1992-2024. Все права защищены

## Управление реле (RelayCommutation.exe)

<table>
<tr><th>Параметр (param)</th><th>Аргумент (value)</th></tr>
<tr><td>0</td><td>Установка порядкового номера модуля АЦП для управления платой реле через цифровой порт (от 0 до (количество устройств - 1))</td></tr>
<tr><td>1</td><td>Установка на переключение всех реле в положение A (аргумент может принимать любое значение)</td></tr>
<tr><td>2</td><td>Установка на переключение всех реле в положение B (аргумент может принимать любое значение)</td></tr>
<tr><td>10, 11, … 25</td><td>Установка на переключение одного реле в положение B из положения A (аргумент может принимать любое значение)
<table>
<tr><td>10</td><td>// 1 реле</td></tr>
<tr><td>11</td><td>// 2 реле</td></tr>
<tr><td>12</td><td>// 3 реле</td></tr>
<tr><td>13</td><td>// 4 реле</td></tr>
<tr><td>14</td><td>// 5 реле</td></tr>
<tr><td>15</td><td>// 6 реле</td></tr>
<tr><td>16</td><td>// 7 реле</td></tr>
<tr><td>17</td><td>// 8 реле</td></tr>
<tr><td>18</td><td>// 9 реле</td></tr>
<tr><td>19</td><td>// 10 реле</td></tr>
<tr><td>20</td><td>// 11 реле</td></tr>
<tr><td>21</td><td>// 12 реле</td></tr>
<tr><td>22</td><td>// 13 реле</td></tr>
<tr><td>23</td><td>// 14 реле</td></tr>
<tr><td>24</td><td>// 15 реле</td></tr>
<tr><td>25</td><td>// 16 реле</td></tr>
</table></td></tr>
<tr><td>30, 31, … 45</td><td>Установка на переключение одного реле в положение A из положения B (аргумент может принимать любое значение)
<table>
<tr><td>30</td><td>// 1 реле</td></tr>
<tr><td>31</td><td>// 2 реле</td></tr>
<tr><td>32</td><td>// 3 реле</td></tr>
<tr><td>33</td><td>// 4 реле</td></tr>
<tr><td>34</td><td>// 5 реле</td></tr>
<tr><td>35</td><td>// 6 реле</td></tr>
<tr><td>36</td><td>// 7 реле</td></tr>
<tr><td>37</td><td>// 8 реле</td></tr>
<tr><td>38</td><td>// 9 реле</td></tr>
<tr><td>39</td><td>// 10 реле</td></tr>
<tr><td>40</td><td>// 11 реле</td></tr>
</table></td></tr>
</table>

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| | 41 // 12 реле<br>42 // 13 реле<br>43 // 14 реле<br>44 // 15 реле<br>45 // 16 реле |

### Управление релейными ключами (PortCommutation.exe)

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 0,1…19 | Установка на переключение одного релейного ключа TX в положение противоположное нынешнему положению (аргумент может принимать любое значение). |
| 20,21…39 | Установка на переключение одного релейного ключа RX в положение противоположное нынешнему состоянию (аргумент может принимать любое значение). |
| 40 | Установка на переключение всех реле TX в положение «разомкнуто» (аргумент может принимать любое значение). |
| 41 | Установка на переключение всех реле TX в положение «замкнуто» (аргумент может принимать любое значение». |
| 42 | Установка на переключение всех реле RX в положение «разомкнуто» (аргумент может принимать любое значение». |
| 43 | Установка на переключение всех реле RX в положение «замкнуто» (аргумент может принимать любое значение). |

## 5.7.меню Сетевые программы

Содержит перечень программ из вкладки меню Сетевые программы из ZETLab, которыми можно управлять через Unit или Unit-2.

### Подключение устройств по Ethernet (NetWizard.exe)

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 0 | Установка количества подключаемых устройств (от 0 до 1024) |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 1 | Установка разрешения подключения только успешно прошедших проверку устройств:<br>0 – подключать все устройства<br>1 – подключать только успешно прошедшие проверку устройства |
| 127 | Установка на деактивацию подключения устройств по Ethernet |
| 255 | Установка на активацию подключения устройств по Ethernet |
| 1000, 1001, … , 2024 | Установка IP-адреса первого … тысяча двадцать четвертого устройства, подключаемого по Ethernet (IP-адрес устройства задается в виде строки вида xxx.xxx.xxx.xxx) |

### Подключение устройств по BlueTooth (BthWizard.exe)

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 0 | Установка типа подключаемого модуля АЦП:<br>10 – ZET 210<br>13 – ZET 302 |
| 1 | Установка на проверку возможности подключения к модулям АЦП по BlueTooth (аргумент может принимать любое значение) |
| 2 | Установка порядкового номера модуля АЦП для подключения по BlueTooth (от 0 до (количество устройств - 1)) |
| 3 | Установка на активацию соединения (аргумент может принимать любое значение) |
| 4 | Установка на деактивацию соединения (аргумент может принимать любое значение) |

# Глава 6.Установка параметров программ ZETScope

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 0 | Установка канала, настройки которого будут устанавливаться:<br>1 – первый канал<br>2 – второй канал |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 1 | Установка видимости выбранного канала на экране:<br>0 – канал виден<br>1 – канал не виден |
| 2 | Установка вертикальной развертки по выбранному каналу в вольтах/клетку экрана (0.002, 0.005, 0.01, 0.02, 0.05, 0.1, 0.2, 0.5, 1, 2, 5) |
| 3 | Установка типа входа выбранного канала:<br>0 – по постоянному току (DC)<br>1 – по переменному току (AC) |
| 4 | Установка коэффициента ослабления пробника выбранного канала:<br>0 – 1x<br>1 – 10x |
| 5 | Установка инверсии сигнала выбранного канала:<br>0 – сигнал не инвертируется<br>1 – сигнал инвертируется |
| 6 | Установка вертикального положения маркера выбранного канала на экране в клетках относительно центра экрана (от минус 5 до 5) |
| 10 | Установка видимости математического канала на экране:<br>0 – канал виден<br>1 – канал не виден |
| 11 | Установка вертикального положения маркера математического канала на экране в клетках относительно центра экрана (от минус 5 до 5) |
| 12 | Установка математической операции математического канала:<br>0 – первый канал + второй канал<br>1 – первый канал - второй канал<br>2 – второй канал - первый канал |
| 20 | Установка задержки синхронизации на экране по горизонтали в клетках относительно центра экрана (от минус 5 до 5) |
| 21 | Установка временной развертки в секундах/клетку экрана (0.00000001, 0.000000025, 0.00000005, 0.0000001, 0.00000025, 0.0000005, 0.000001, 0.0000025, 0.000005, 0.00001, 0.000025, 0.00005, 0.0001, 0.00025, 0.0005, 0.001, 0.0025, 0.005, 0.01, 0.025, 0.05, 0.1, 0.25, 0.5, 1, 2.5, 5, 10, 25, 50) |
| 22 | Установка уровня запуска синхронизации на экране по горизонтали в клетках относительно центра экрана (от минус 5 до 5) |
| 23 | Установка канала синхронизации:<br>0 – первый канал<br>1 – второй канал |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| 24 | Установка фронта синхронизации:<br>0 – восходящий<br>1 – нисходящий |
| 25 | Установка режима синхронизации:<br>0 – автоматический<br>1 – обычный |
| 26 | Установка интерполяции сигнала:<br>0 – без интерполяции<br>1 – линейная интерполяция<br>2 – интерполяция sin(x)/x |
| 27 | Установка послесвечения:<br>0 – послесвечение отключено<br>1 – послесвечение 1 секунда<br>2 – послесвечение 2 секунды<br>3 – послесвечение 5 секунд<br>4 – бесконечное послесвечение<br>5 – цифровое послесвечение |
| 28 | Установка режима сбора данных:<br>0 – усреднение<br>1 – пиковый<br>2 – выборка |
| 29 | Установка количества усреднений (в режиме сбора данных «усреднение»):<br>0 – 4<br>1 – 8<br>2 – 16<br>3 – 32<br>4 – 64 |
| 30 | Установка видимости панели управления:<br>0 – панель управления видна<br>1 – панель управления не видна |
| 41 | Установка номера поля измерений, настройки которого будут устанавливаться (от 0 до 4) |
| 42 | Установка канала для выбранного поля:<br>0 – первый канал<br>1 – второй канал |
| 43 | Установка вида измерения для выбранного поля:<br>– при обычных измерениях: |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| | 0 – частота<br>1 – период<br>2 – среднее<br>3 – размах<br>4 – СКЗ<br>5 – минимум<br>6 – максимум<br>7 – нарастание<br>8 – спад<br>9 – положительный импульс<br>10 – отрицательный импульс<br>11 – положительный выброс<br>12 – отрицательный выброс<br>13 – высокое<br>14 – низкое<br>15 – нет измерения<br>– при измерениях с вертикальными курсорами:<br>0 – разность в вольтах<br>1 – разность в секундах<br>2 – курсор 1 в секундах<br>3 – курсор 1 в вольтах<br>4 – курсор 2 в секундах<br>5 – курсор 2 в вольтах<br>6 – частота<br>7 – нет измерения<br>– при измерениях с горизонтальными курсорами:<br>0 – разность в вольтах<br>1 – курсор 1 в вольтах<br>2 – курсор 2 в вольтах<br>3 – нет измерения |
| 44 | Установка типа измерений:<br>1 – обычные измерения<br>2 – измерения с использованием вертикальных курсоров<br>3 – измерения с использованием горизонтальных курсоров |
| 45 | Установка сохранения изменений в окне измерений (при открытом окне измерений):<br>0 – отменить изменения и закрыть окно<br>1 – сохранить изменения и закрыть окно |
| 50 | Установка работы окна спектра:<br>0 – закрытие окна спектра |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Аргумент (*value*) |
|---|---|
| | 1 – открытие окна спектра |
| 51 | Установка видимости графика спектра выбранного канала:<br>0 – спектр не виден<br>1 – спектр виден |
| 52 | Установка типа весового окна при расчете спектра:<br>0 – прямоугольное<br>1 – Хана<br>2 – Хэмминга<br>3 – Блэкмана<br>4 – Барлета<br>5 – Блэкмана стд. |
| 255 | Установка ведения сбора данных:<br>0 – сбор данных не ведется<br>1 – сбор данных ведется |

# Глава 7.Формат посылаемых данных ZETLab

Для пересылки данных в программах используются методы *Write(...)*, *WriteNet(...)* и *UnitWrite(...)* на стороне сервера и методы *Read(...)*, *ReadNet(...)* и *UnitRead(...)* на стороне клиента, где аргумент *param* отвечает за посылку определенных данных. Ниже приведены значения аргумента *param* для пересылки данных при работе через компонент *Unit*.

В настоящее время обмен данными между процессами можно осуществлять через механизм Unit, реализованный в нашем ПО. Для этого вам необходимо изучить документ "ZETLAB STUDIO".

Суть работы механизма Unit заключается в обмене данными через общую область памяти. В вашей программе вы создаёте экземпляр Active-X элемента UnitCln.ocx и через него запускаете программу "Анализ нелинейных искажений". Через механизм Unit вы можете изменять настройки программы и получать результаты.

**Пример кода для извлечения принимаемых данных через Unit**

```
#include <CDUnitCln.h>
CDUnitCln unit;
unit.Create(L"", WS_CHILD, CRect(0, 0, 0, 0), this, UNIT_ID);

BEGIN_EVENTSINK_MAP(CZetHarmDistCtrl, CZetBasicOle)
	ON_EVENT(CZetHarmDistCtrl, UNIT_ID, 1, ReadyToGetData, VTS_NONE)
```

© ООО "ЭТМС" 1992-2024. Все права защищены

```
END_EVENTSINK_MAP()

struct Unit2DataStruct
{// Структура, содержащая свойства данных (параметров и результатов) Unit-2
	Unit2DataType type;		// тип данных
	long number;			// номер (код) данных
	BOOL dynamicArea;		// динамическая область памяти (или статическая)
	DWORD sizeArray;		// размер массива (если данные - массив)
	DWORD sizeDataInByte;	// размер данных (которые сейчас в структуре) в
байтах
	DWORD sizeAreaInByte;	// размер области памяти в байтах
	BYTE *pData;			// указатель на область памяти для данных
}

void CZetHarmDistCtrl::ReadyToGetData()
{
	Unit2DataStruct data;
	DWORD size_car;
	if ((unit.GetSizeCurResult(&size_car) == NOERROR) &&
		(data.TestSize(size_car) == NOERROR) &&
		(unit.ReadResult(&data) == NOERROR))
	{
		if (data.type == tdu_32f_arr)
		{
			std::vector<float> fdata(data.sizeArray);
			if (data.Extract_arr(fdata.data()) == NOERROR && data.sizeArray >
8)
			{
				if (data.number == 2)
				{
					//	Place your code here
				}
			}
		}
	}
}
```

## 7.1.меню Анализ сигналов

Содержит перечень программ из вкладки меню Анализ сигналов из ZETLab, которыми можно управлять через Unit или Unit-2.

### Узкополосный спектр (spectr.exe)

***Узкополосный спектральный анализ (spectr)***

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Данные (*data*) |
|---|---|
| 0 | Передача массива значений частотного ряда (если была установка на последовательное получение данных параметром 19) |
| 1 | Передача массива значений текущего спектра (если была установка на последовательное получение данных параметром 19) |
| 2 | Передача массива значений максимального спектра (если была установка на последовательное получение данных параметром 19 и разрешение расчета максимального спектра параметром 20) |
| 3 | Передача массива значений среднего спектра (если была установка на последовательное получение данных параметром 19 и разрешение расчета среднего спектра параметром 21) |
| 5 | Передача размера массива спектра (если была установка на получение размера массива спектра параметром 5) |
| 6 | Передача массива значений частотного ряда (если была установка на получение значений частотного ряда параметром 6) |
| 7 | Передача массива значений текущего спектра (если была установка на получение значений текущего спектра параметром 7) |

### Долеоктавный спектр (dspectr.exe)

| Параметр (*param*) | Данные (*data*) |
|---|---|
| 1 | Передача размера массива спектра (если была установка на получение размера массива спектра параметром 3) |
| 2 | Передача массива, состоящего из:<br>– значений текущего спектра (если была установка на получение данных параметром 4)<br>– значений частотного ряда (если была установка на получение данных параметром 4)<br>– значений максимального спектра (если была установка на получение данных параметром 4 и разрешение расчета максимального спектра параметром 10)<br>– значений минимального спектра (если была установка на получение данных параметром 4 и разрешение расчета минимального спектра параметром 11)<br>– значений среднего спектра (если была установка на получение данных параметром 4 и разрешение расчета среднего спектра параметром 12) |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Данные (*data*) |
|---|---|
| 3 | Передача массива значений частотного ряда (если была установка на получение значений частотного ряда параметром 6) |

## Взаимный узкополосный спектр (vspectr.exe)

| Параметр (*param*) | Данные (*data*) |
|---|---|
| 0 | Передача:<br>– размера массива спектра (если была установка на получение размера массива спектра параметром 9)<br>– массива значений частотного ряда (если была установка на получение значений частотного ряда параметром 10 или была установка на последовательное получение данных параметром 12)<br>– массива значений текущего спектра (если была установка на получение значений текущего спектра параметром 11) |
| 1 | Передача массива значений текущего спектра (если была установка на последовательное получение данных параметром 12) |
| 2 | Передача массива значений действительной части спектра (если была установка на последовательное получение данных параметром 12 и разрешение расчета действительной части спектра параметром 14) |
| 3 | Передача массива значений мнимой части спектра (если была установка на последовательное получение данных параметром 12 и разрешение расчета мнимой части спектра параметром 15) |
| 4 | Передача массива значений фазы (если была установка на последовательное получение данных параметром 12 и разрешение расчета фазы параметром 16) |
| 5 | Передача массива значений коэффициента когерентности (если была установка на последовательное получение данных параметром 12 и разрешение расчета коэффициента когерентности параметром 17) |
| 6 | Передача массива с действительной частью переходной характеристики |
| 7 | Передача массива с мнимой частью переходной характеристики |
| 8 | Передача спектра первого канала |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Данные (*data*) |
|---|---|
| 9 | Передача спектра второго канала |

## Взаимный долеоктавный спектр (dvspectr.exe)

| Параметр (*param*) | Данные (*data*) |
|---|---|
| 0 | Передача:<br>– размера массива спектра (если была установка на получение размера массива спектра параметром 4)<br>– массива, состоящего из:<br>1) значений частотного ряда (если была установка на получение данных параметром 5)<br>2) значений текущего спектра (если была установка на получение данных параметром 5)<br>3) значений действительной части спектра (если была установка на получение данных параметром 5 и разрешение расчета действительной части спектра параметром 13)<br>4) значений мнимой части спектра (если была установка на получение данных параметром 5 и разрешение расчета мнимой части спектра параметром 14)<br>5) значений фазы (если была установка на получение данных параметром 5 и разрешение расчета фазы параметром 15)<br>6) значений коэффициента когерентности (если была установка на получение данных параметром 5 и разрешение расчета коэффициента когерентности параметром 16)<br>– массива значений частотного ряда (если была установка на получение данных параметром 6)<br>– массива значений текущего спектра (если была установка на получение данных параметром 7)<br>– массива значений действительной части спектра (если была установка на получение данных параметром 8 и разрешение расчета действительной части спектра параметром 13)<br>– массива значений мнимой части спектра (если была установка на получение данных параметром 9 и разрешение расчета мнимой части спектра параметром 14) |

© ООО "ЭТМС" 1992-2024. Все права защищены

### Взаимный корреляционный анализ (corr.exe)

| Параметр (*param*) | Данные (*data*) |
|---|---|
| 7 | Передача размера массива корреляционной функции (если была установка на получение размера массива корреляционной функции параметром 7) |
| 8 | Передача массива значений временного ряда (если была установка на получение значений временного ряда параметром 8) |
| 9 | Передача массива значений корреляционной функции (если была установка на получение значений корреляционной функции параметром 9) |
| 10 | Передача массива значений огибающей корреляционной функции (если была установка на разрешение расчета и получение значений огибающей корреляционной функции параметром 10) |

### Анализ нелинейных искажений (harmdist.exe)

| Параметр (*param*) | Данные (*data*) |
|---|---|
| 2 | 0 - Основная частота<br>1 - Коэффициент гармонических искажений (THD)<br>2 - Коэффициент нелинейных искажений с шумом (THD+N)<br>3 - Динамический диапазон свободный от паразитных составляющих (SFDR)<br>4 - Отношение сигнал к шуму (SNR)<br>5 - Отношение сигнал к шуму с искажениями (SINAD)<br>6 - Эффективное количество бит (ENOB)<br>7 - Полный динамический диапазон (FS)<br>8 - Эффективное количество бит в полном динамическом диапазоне (ENOBFS) |

### Синхронное накопление (PrdkAnaliz.exe)

| Параметр (*param*) | Данные (*data*) |
|---|---|
| 0 | Передача массива значений формы сигнала (если была установка на получение значений формы сигнала параметром 22 путем установки его в 0) |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Данные (*data*) |
|---|---|
| 1 | Передача массива значений спектра гармоник (если была установка на получение значений спектра гармоник параметром 22 путем установки его в 1) |
| 2 | Передача значения скорости |

### Модальный анализ (PrqsAnaliz.exe)

| Параметр (*param*) | Данные (*data*) |
|---|---|
| 0 | Передача массива, состоящего из 22 элементов:<br>0 – амплитуда удара (оп. канал)<br>1 – амплитуда удара (изм. канал)<br>2 – ширина импульса по 50-процентному уровню (оп. канал)<br>3 – ширина импульса по 50-процентному уровню (изм. канал)<br>4 – ширина импульса по 10-процентному уровню (оп. канал)<br>5 – ширина импульса по 10-процентному уровню (изм. канал)<br>6 – СКЗ шума до импульса (оп. канал)<br>7 – СКЗ шума до импульса (изм. канал)<br>8 – порог срабатывания на импульс (оп. канал)<br>9 – порог срабатывания на импульс (изм. канал)<br>10 – интеграл первого порядка от импульса (оп. канал)<br>11 – интеграл первого порядка от импульса (изм. канал)<br>12 – интеграл второго порядка от импульса (оп. канал)<br>13 – интеграл второго порядка от импульса (изм. канал)<br>14 – интеграл произведения силы на перемещение (оп. канал)<br>15 – интеграл произведения силы на перемещение (изм. канал)<br>16 – время прохождения импульса (оп. канал)<br>17 – время прохождения импульса (изм. канал)<br>18 – соотношение сигнал-шум (оп. канал)<br>19 – соотношение сигнал-шум (изм. канал)<br>20 – отношение амплитуд импульсов на измерительном и опорном каналах<br>21 – разность времени следования импульсов на измерительном и опорном каналах |
| 1 | Передача массива значений формы сигнала опорного канала |
| 2 | Передача массива значений формы сигнала измерительного канала |
| 3 | Передача массива значений временной развертки сигналов опорного и измерительного каналов |

© ООО "ЭТМС" 1992-2024. Все права защищены

### Гистограмма (ZETHistograph.exe)

| Параметр (*param*) | Данные (*data*) |
|---|---|
| 0 | Передача массива значений разбиения оси абсцисс |
| 1 | Передача массива значений расчета гистограммы |
| 2 | Передача массива значений нормального распределения (если была установка на разрешение расчета значений нормального распределения параметром 6) |
| 3 | Передача массива значений гармонического распределения (если была установка на разрешение расчета значений гармонического распределения параметром 7) |
| 4 | Передача массива значений распределения Хи-квадрат (если была установка на разрешение расчета значений распределения Хи-квадрат параметром 8) |

### Детектор STA\LTA (STA_LTA.exe)

| Параметр (*param*) | Данные (*data*) |
|---|---|
| 0 | Передача значения, информирующего о событии:<br>0 – событие закончилось;<br>1 – событие началось |
| 1 | Передача значения идентификатора виртуального канала STA_LTA (если была установка на разрешение передачи данных в виртуальный канал STA_LTA) |

## 7.2.меню Измерение

Содержит перечень программ из вкладки меню Измерение из ZETLab, которыми можно управлять через Unit или Unit-2.

### Вольтметр переменного тока (VoltMeter.exe)

| Параметр (*param*) | Данные (*data*) |
|---|---|
| 0 | Одноканальный режим работы - передача измеренного значения переменной составляющей сигнала |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Данные (*data*) |
|---|---|
| N | Многоканальный режим работы - передача измеренного значения переменной составляющей сигнала N-ого канала вольтметра (N от 0 до кол-ва каналов вольтметра без единицы) |

### Вольтметр постоянного тока (VoltMeterDC.exe)

| Параметр (*param*) | Данные (*data*) |
|---|---|
| 0 | Одноканальный режим работы - передача измеренного значения постоянной составляющей сигнала |
| N | Многоканальный режим работы - передача измеренного значения постоянной составляющей сигнала N-ого канала вольтметра (N от 0 до кол-ва каналов вольтметра без единицы) |

### Селективный вольтметр (VoltMeterSel.exe)

| Параметр (*param*) | Данные (*data*) |
|---|---|
| 0 | Передача измеренного значения переменной составляющей сигнала |

### Частотомер (FreqMeter.exe)

| Параметр (*param*) | Данные (*data*) |
|---|---|
| 0 | Передача измеренного значения частоты сигнала |

### Фазометр (PhaseMeter.exe)

| Параметр (*param*) | Данные (*data*) |
|---|---|
| 0 | Передача измеренного значения разности фаз сигналов |

© ООО "ЭТМС" 1992-2024. Все права защищены

### Тахометр (TahoMeter.exe)

| Параметр (*param*) | Данные (*data*) |
|---|---|
| 0 | Передача измеренного значения частоты вращения |
| 1 | Передача измеренного значения общего количества оборотов с момента последнего сброса счетчика оборотов |

### Торсиограф (Torsiograph.exe)

| Параметр (*param*) | Данные (*data*) |
|---|---|
| 0 | Передача измеренного значения перемещения |

### Энкодер (Encoder.exe)

| Параметр (*param*) | Данные (*data*) |
|---|---|
| 0 | Передача измеренного значения перемещения |

### Термометр ТС (ThermoMeter.exe)

| Параметр (*param*) | Данные (*data*) |
|---|---|
| 0 | Передача измеренного значения температуры |

### Термометр ТП (ThermoPara.exe)

| Параметр (*param*) | Данные (*data*) |
|---|---|
| 0 | Передача измеренного значения температуры |

### Тензометр (TenzoMeter.exe)

| Параметр (*param*) | Данные (*data*) |
|---|---|
| 0 | Передача измеренного значения нагрузки |

© ООО "ЭТМС" 1992-2024. Все права защищены

### Виброметр (VibroMeter.exe)

| Параметр (*param*) | Данные (*data*) |
|---|---|
| 0 | Передача массива, состоящего из 3 элементов:<br>0 – значение ускорения<br>1 – значение скорости<br>2 – значение перемещения |
| N | Многоканальный режим работы - передача измеренного значения сигнала N-ого канала виброметра (N от 0 до кол-ва каналов виброметра без единицы) |

### Мультиметр Agilent_HP34401A (Agilent_HP34401A.exe)

| Параметр (*param*) | Данные (*data*) |
|---|---|
| 0 | Передача измеренного значения величины |

### Блок питания LPS305 (LPS305.exe)

| Параметр (*param*) | Данные (*data*) |
|---|---|
| 0 | Передача массива, состоящего из 17 элементов:<br>0 – текущее напряжение по первому каналу<br>1 – текущее напряжение по второму каналу<br>2 – текущий ток по первому каналу<br>3 – текущий ток по второму каналу<br>4 – режим ограничения по первому каналу:<br>0 – ограничение по напряжению (CV)<br>1 – ограничение по току (CC)<br>5 – режим ограничения по первому каналу:<br>0 – ограничение по напряжению (CV)<br>1 – ограничение по току (CC)<br>6 – режим слежения:<br>0 – независимая работа каналов<br>1 – слежение за первым каналом<br>7 – состояние питания цифрового выхода:<br>0 – цифровой выход отключен<br>1 – питание 3.3 В<br>2 – питание 5 В<br>8 – состояние выходов: |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Данные (*data*) |
|---|---|
| | 0 – выходы выключены<br>1 – выходы включены<br>9 – состояние перегрузки цифрового выхода:<br>0 – цифровой выход не перегружен<br>1 – цифровой выход перегружен<br>10 – состояние кулера:<br>0 – кулер выключен<br>1 – кулер включен<br>11 – состояние звукового сигнала:<br>0 – звуковой сигнал выключен<br>1 – звуковой сигнал включен<br>12 – состояние работы компенсации выхода по току<br>0 – компенсация выхода по току выключена<br>1 – компенсация выхода по току включена<br>13 – заданное ограничение по напряжению на первом канале<br>14 – заданное ограничение по напряжению на втором канале<br>15 – заданное ограничение по току на первом канале<br>16 – заданное ограничение по току на втором канале |

## Блок питания PPE3323 (PPE3323.exe)

| Параметр (*param*) | Данные (*data*) |
|---|---|
| 0 | Передача:<br>– значения ошибки при работе с прибором:<br>256 – устройство отсутствует<br>– массива, состоящего из 22 элементов:<br>0 – текущее напряжение по первому каналу<br>1 – текущее напряжение по второму каналу<br>2 – текущий ток по первому каналу<br>3 – текущий ток по второму каналу<br>4 – режим стабилизации по напряжению<br>0 – режим стабилизации по напряжению выключен<br>1 – режим стабилизации по напряжению включен<br>5 – режим стабилизации по току<br>0 – режим стабилизации по току выключен<br>1 – режим стабилизации по току включен<br>6 – режим слежения:<br>0 – независимая работа каналов<br>1 – слежение за первым каналом |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Данные (*data*) |
|---|---|
| | 7 – состояние питания цифрового выхода:<br>1 – питание 3.3 В<br>2 – питание 5 В<br>8 – состояние выходов:<br>0 – выходы выключены<br>1 – выходы включены<br>9 – последовательное соединение выходов:<br>0 – выбран первый канал<br>1 – выбран второй канал<br>2 – последовательное соединение выходов<br>10 – 0 (не используется)<br>11 – 0 (не используется)<br>12 – состояние защиты от перегрузки<br>0 – защита от перегрузки выключена<br>1 – защита от перегрузки включена<br>13 – заданное ограничение по напряжению на первом канале<br>14 – заданное ограничение по напряжению на втором канале<br>15 – заданное ограничение по току на первом канале<br>16 – заданное ограничение по току на втором канале<br>17 – заданный уровень защиты по напряжению на первом канале<br>18 – заданный уровень защиты по напряжению на втором канале<br>19 – заданное ограничение по напряжению при последовательном соединении выходов<br>20 – заданное ограничение по току при последовательном соединении выходов<br>21 – заданный уровень защиты по напряжению при последовательном соединении выходов |

## Блок питания PSM2010 (PSM2010.exe)

| Параметр (*param*) | Данные (*data*) |
|---|---|
| 0 | Передача массива, состоящего из 13 элементов:<br>0 – текущее напряжение<br>1 – текущее напряжение<br>2 – состояние выходов:<br>0 – выходы выключены<br>1 – выходы включены<br>3 – состояние защиты от перегрузки (OCP)<br>0 – защита от перегрузки выключена |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Данные (*data*) |
|---|---|
| | 1 – защита от перегрузки включена<br>4 – состояние защиты от перенапряжения (OVP)<br>0 – защита от перенапряжения выключена<br>1 – защита от перенапряжения включена<br>5 – текущий рабочий диапазон:<br>0 – 8V, 20A<br>1 – 20V, 10A<br>6 – заданное ограничение по напряжению<br>7 – заданное ограничение по току<br>8 – заданный уровень защиты по напряжению<br>9 – заданный уровень защиты от перегрузки<br>10 – индикация события перенапряжения<br>0 – перенапряжение отсутствует<br>1 – существует перенапряжение<br>11 – индикация события перегрузки<br>0 – перегрузка отсутствует<br>1 – существует перегрузка<br>12 – код ошибки |

## 7.3.меню Отображение

Содержит перечень программ из вкладки меню Отображение из ZETLab, которыми можно управлять через Unit или Unit-2.

### Многоканальный осциллограф (OscGraph.exe)

| Параметр (*param*) | Данные (*data*) |
|---|---|
| 0 | Передача:<br>– размера массива данных (если была установка на получение размера массива данных параметром 7)<br>– массива данных (если была установка на получение массива данных параметром 8) |

### XYZ-осциллограф (XYOscGraph.exe)

| Параметр (*param*) | Данные (*data*) |
|---|---|
| 0 | Передача: |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Данные (*data*) |
|---|---|
| | – размера массива данных (если была установка на получение размера массива данных параметром 7)<br>– массива данных (если была установка на получение массива данных параметром 8) |

## 7.4.меню Генераторы

Содержит перечень программ из вкладки меню Генераторы из ZETLab, которыми можно управлять через Unit или Unit-2.

### Генератор сигналов (DAC_OCX.exe)

| Параметр (*param*) | Данные (*data*) |
|---|---|
| 0 | Передача значения текущей частоты сигнала линейной или логарифмической частотной модуляции |

### Синхронный генератор (SynchroChanDac.exe)

Программа не передает в вызывающую программу никаких данных.

## 7.5.меню Регистрация

Содержит перечень программ из вкладки меню Регистрация из ZETLab, которыми можно управлять через Unit или Unit-2.

### Запись сигналов (SignalWriter.exe)

| Параметр (*param*) | Данные (*data*) |
|---|---|
| 1 | Передача значения, равного 0 – означает, что произошла остановка записи. |
| 2 | Передача значения, равного 0 – означает, что произошли изменения в ZETServer (произошло подключение, отключение устройств, подключение, отключение каналов и т.д.) |

© ООО "ЭТМС" 1992-2024. Все права защищены

### Воспроизведение сигналов (reader.exe)

| Параметр (*param*) | Данные (*data*) |
|---|---|
| 0 | Передача значения, равного 0 – означает, что произошла остановка воспроизведения, так как воспроизведение дошло до конца файлов. |
| 1 | Передача значения, равного 0 – означает, что произошла приостановка воспроизведения сигналов |

### Многоканальный самописец (multiSWvm.exe)

| Параметр (*param*) | Данные (*data*) |
|---|---|
| 0 | Передача:<br>– массива текущих значений по всем измеряемым параметрам (если была установка на получение массива текущих значений параметром 16)<br>– размера массива данных (если была установка на получение размера массива данных параметром 16) |

## 7.6.меню Автоматизация

Содержит перечень программ из вкладки меню Автоматизация из ZETLab, которыми можно управлять через Unit или Unit-2.

### Регулятор (Regulator.exe)

| Параметр (*param*) | Данные (*data*) |
|---|---|
| 0 | Передача измеренного значения контролируемого сигнала |

### Арифмометр (ArithmoMeter.exe)

| Параметр (*param*) | Данные (*data*) |
|---|---|
| 0 | Передача измеренного значения результирующего сигнала |

© ООО "ЭТМС" 1992-2024. Все права защищены

### Адаптивный фильтр 50 Гц (filtr50.exe)

| Параметр (*param*) | Данные (*data*) |
|---|---|
| 0 | Передача массива, состоящего из 2 элементов:<br>0 – значение частоты сигнала<br>1 – значение амплитуды сигнала |

### Фильтрация сигналов (filtrdiff.exe)

| Параметр (*param*) | Данные (*data*) |
|---|---|
| В качестве параметра передается идентификатор виртуального канала первого … сотого канала фильтра, заданного параметром 18 … 1008 | 0 |

### Формула (ZETFormula.exe)

Программа не передает в вызывающую программу никаких данных.

### Синхронизация по GPS (Synchronization.exe)

| Параметр (*param*) | Данные (*data*) |
|---|---|
| 0 | Передача массива, состоящего из 11 элементов:<br>0 – год<br>1 – месяц<br>2 – день<br>3 – час<br>4 – минута<br>5 – секунда<br>6 – широта<br>7 – долгота<br>8 – скорость<br>9 – количество спутников<br>10 – сдвиг по времени |

### Управление реле (RelayCommutation.exe)

Программа не передает в вызывающую программу никаких данных.

© ООО "ЭТМС" 1992-2024. Все права защищены

### Управление релейными ключами (PortCommutation.exe)

Программа не передает в вызывающую программу никаких данных.

## 7.7.меню Сетевые программы

Содержит перечень программ из вкладки меню Сетевые программы из ZETLab, которыми можно управлять через Unit или Unit-2.

### Подключение устройств по Ethernet (NetWizard.exe)

Программа не передает в вызывающую программу никаких данных.

### Подключение устройств по BlueTooth (BthWizard.exe)

| Параметр (*param*) | Данные (*data*) |
|---|---|
| 0 | Передача массива, состоящего из 10 элементов, равные серийным номерам найденных модулей АЦП. |
| 256 | Передача значения, равного 0 – означает, что нет BlueTooth адаптеров |
| 257 | Передача значения, отражающего результат выполнения команды активации соединения:<br>0 – активация прошла успешно<br>1 – не удалось активировать модуль<br>2 – активация недоступна |
| 258 | Передача значения, равного 0 – означает, что при поиске доступных для подключения модулей АЦП не было найдено ни одного устройства |
| 259 | Передача значения, отражающего результат выполнения команды деактивации соединения:<br>0 – деактивация прошла успешно<br>1 – не удалось деактивировать модуль<br>2 – деактивация недоступна |

# Глава 8.Формат посылаемых данных ZETScope

| Параметр (*param*) | Данные (*data*) |
|---|---|
| 0 | Передача массива, состоящего из:<br>– значений по первому каналу осциллографа |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Параметр (*param*) | Данные (*data*) |
|---|---|
| | – значений по второму каналу осциллографа |
| 1 | Передача размера массива значений по первому и второму каналу осциллографа |
| 2 | Передача значения кода ошибки:<br>1 – невозможно подключиться к прибору<br>2 – прибор не отвечает на обмен данными<br>3 – прибор не запускается<br>4 – не найдено ни одного устройства |

# Глава 9.Перечень функций, используемые Unit.ocx в базовом классе

Запуск программ при помощи интерфейсного программного модуля *Unit* представляет собой технологию клиент-сервер, где серверной частью является запускаемое и управляемое приложение, а клиентской – приложение, которое запускает заданную программу и управляет ей, получая от нее определенные данные. При этом методы и события компонента делятся на те, которые используются для настройки клиентской части, и те, которые используются для настройки серверной части.

## 9.1.Серверная часть

1. При инициализации клиентской части проверяется загрузка серверного приложения. В случае успешной загрузки клиентская часть генерирует событие *ServerLoad*.

2 При записи результатов серверное приложение проверяет клиентское приложение. Если определяется, что клиентского приложения нет, или оно зависло, то серверное приложение должно завершить свою работу.

3 Клиентское приложение по таймеру, с периодичностью 3 сек, проверяет серверное приложение.

4 При определении, что серверного приложения нет, клиентская часть генерирует событие *ServerLost*. При определении зависания – событие *ServerNotActive*.

5 Обработка клиентской частью событий *ServerLoad*, *ServerLost*, и *ServerNotActive* не обязательна.

© ООО "ЭТМС" 1992-2024. Все права защищены

## Методы

***UnitReg*** – установление режима работы программы с возможностью управления ею через программный модуль *Unit* (вызывается при старте программы).

*long* ***UnitReg****(long hw)*

Параметры

*hw* – хэндлер окна приложения.

Возвращаемое значение

*0* – нормальное выполнение функции.

*-1* – хэндлер окна приложения равен *NULL* или не было запуска программы через компонент *Unit*.

*-2* – не хватает хэндлеров памяти.

*-3* – не хватает хэндлеров памяти.

***UnitUnReg*** – прекращение режима работы программы с возможностью управления ею через программный модуль *Unit* (вызывается при завершении программы).

*long* ***UnitUnReg****(long* `UHw`*)*

Параметры

`UHw` – хэндлер окна приложения.

Возвращаемое значение

*0* – нормальное выполнение функции.

*-1* – не было запуска программы через компонент *Unit*.

*-2* – хэндлер окна приложения равен *NULL*.

***UnitParam*** – считывание параметра, присланного в программу через компонент Unit запускающим приложением.

*long* ***UnitParam*** *(long* param, double* value)*

Параметры

*param* – возвращает номер устанавливаемого параметра.

*value* – возвращает значение устанавливаемого параметра.

Возвращаемое значение

*0* – нормальное выполнение функции.

*-1* – не было подключения к программе через компонент *Unit*.

*-2* – хэндлер окна приложения равен *NULL*.

*-3* – не хватает хэндлеров памяти.

*-4* – не хватает хэндлеров памяти.

*-5* – место в памяти для возврата номера устанавливаемого параметра равно *NULL*.

© ООО "ЭТМС" 1992-2024. Все права защищены

*-6* – место в памяти для возврата значения устанавливаемого параметра равно *NULL*.

***UnitReadString*** – считывание строкового параметра, присланного в программу через компонент Unit запускающим приложением.\

*long* ***UnitReadString*** *(long* param, char* StringVal)*

Параметры

*param* – возвращает номер устанавливаемого параметра.

*StringVal* – возвращает значение устанавливаемого строкового параметра.

Возвращаемое значение

*0* – нормальное выполнение функции.

*-1* – не было подключения к программе через компонент *Unit*.

*-2* – хэндлер окна приложения равен *NULL*.

*-3* – не хватает хэндлеров памяти.

*-4* – не хватает хэндлеров памяти.

*-5* – место в памяти для возврата номера устанавливаемого параметра равно *NULL*.

*-6* – место в памяти для возврата значения устанавливаемого строкового параметра равно *NULL*.

***UnitReadTimeString*** – считывание строкового параметра, присланного в программу через компонент Unit запускающим приложением с меткой синхронизации по времени *ZETServer*.

*long* ***UnitReadTimeString*** *(long* param, BSTR* StringVal, DOUBLE* timer)*

Параметры

*param* – возвращает номер устанавливаемого параметра.

*StringVal* – возвращает значение устанавливаемого строкового параметра.

*timer* – возвращает значение времени ZETServer.

Возвращаемое значение

*0* – нормальное выполнение функции.

*-1* – не было подключения к программе через компонент *Unit*.

*-2* – хэндлер окна приложения равен *NULL*.

*-3* – не хватает хэндлеров памяти.

*-4* – не хватает хэндлеров памяти.

*-5* – место в памяти для возврата номера устанавливаемого параметра равно *NULL*.

*-6* – место в памяти для возврата значения устанавливаемого строкового параметра равно *NULL*.

© ООО "ЭТМС" 1992-2024. Все права защищены

*-7* – превышена максимальная допустимая длина считываемого строкового параметра.

***UnitWrite*** – передача данных в запущенную программу.

*long* ***UnitWrite****(long size, float* data, long param)*

Параметры

*size* – размер передаваемого массива данных.

*data* – указатель на передаваемый массив данных.

*param* – параметр.

Возвращаемое значение

*0* – нормальное выполнение функции.

*-1* – не было подключения к программе через компонент *Unit*.

*-2* – не было подключения к программе через компонент *Unit*.

*-3* – приложение, которое запустило данную программу через компонент *Unit*, было выгружено из памяти.

*-4* – не хватает хэндлеров памяти.

*-5* – не хватает хэндлеров памяти.

***IsReadyParam*** – запрос на наличие данных от клиента для запущенной через компонент *Unit* программы (сервера).. Функция запоминает количество сообщений, передаваемых от клиента к приложению и возвращает “1”, декрементирую счетчик сообщений до тех пор, пока он не обнулится.. .

*long* ***IsReadyParam*** *(void)*

Возвращаемое значение

*0* – нет доступных данных.

*1* – есть доступные данные.

## События

***Ready*** – возникает при установке параметров программы через функции *SetParam(...)*, *SetParamString(...)* и *SetParamTimeString(...)* через программный модуль *Unit*. Данное событие генерируется только тогда, когда есть идентификатор окна, т.е. при написании програм на тех языках программирования, где отсутствуют идентификаторы окна, событие генерироваться не будет.

*void* ***Ready*** *(long par)*

Параметры

par – параметр.

***Load*** – Данное событие сразу после завершения загрузки программs.

© ООО "ЭТМС" 1992-2024. Все права защищены

*void* ***Load*** *()*

***NotActive*** – Данное событие генерируется в случае, когда запущенная программа прекращает сообщать в UNIT о своей деятельности. См. метод ***IamActive*** клиентской части.

*void* ***NotActive*** *()*

***Lost*** – Данное событие генерируется в случае, когда запущенная программа завершает свою работу не по команде от UNIT.

*void* ***Lost*** *()*

## 9.2.Клиентская часть

1. При инициализации клиентской части проверяется загрузка серверного приложения. В случае успешной загрузки клиентская часть генерирует событие *ServerLoad*.

2 При записи результатов серверное приложение проверяет клиентское приложение. Если определяется, что клиентского приложения нет, или оно зависло, то серверное приложение должно завершить свою работу.

3 Клиентское приложение по таймеру, с периодичностью 3 сек, проверяет серверное приложение.

4 При определении, что серверного приложения нет, клиентская часть генерирует событие *ServerLost*. При определении зависания – событие *ServerNotActive*.

5 Обработка клиентской частью событий *ServerLoad*, *ServerLost*, и *ServerNotActive* не обязательна.

### Методы

***Activate*** – загрузка программы для управления и подключение к ней.

*long* ***Activate****(LPCTSTR name)*

Параметры

*name* – имя программы без расширения.

Возвращаемое значение

*0* – нормальное выполнение функции.

*-1* – нет свободного места для подключения к компоненту *Unit* (слишком много программ подключено к компоненту *Unit*).

*-2* – не хватает хэндлеров памяти.

*-3* – не хватает хэндлеров памяти.

*-4* – не запускается программа *name* (нет на диске, не та версия и т.д.).

© ООО "ЭТМС" 1992-2024. Все права защищены

*-5* – программа *name* есть, но она не проходит инициализацию с компонентом *Unit* (мало оперативной памяти или дискового пространства).

*-6* – не хватает хэндлеров памяти.

***DisActivate*** – закрытие и выгрузка запущенной программы.

*long* ***DisActivate****()*

Возвращаемое значение

*0* – нормальное выполнение функции.

*-1* – не было подключения к программе через компонент *Unit*.

***Read*** – чтение данных из подключенной программы.

*long* ***Read****(long* size, float* data, long* param)*

Параметры

*size* – возвращает размер переданного массива данных.

*data* – адрес массива, куда будут записаны требуемые данные.

*param* – возвращает параметр. Для каждой программы существует свой набор параметров, который определяет возвращаемые данные. Значения параметров для каждой программы представлены в соответствующем разделе.

Возвращаемое значение

*0* – нормальное выполнение функции.

*-1* – не было подключения к программе через компонент *Unit*.

*-2* – хэндлера окна запускаемого приложения не существует.

*-3* – не хватает хэндлеров памяти.

*-4* – не хватает хэндлеров памяти.

***UnitRead*** – чтение данных из подключенной программы по заданному параметру.

*long* **Unit*****Read*****(long* size, float* data, long param)**

Параметры

*size* – возвращает размер переданного массива данных.

*data* – адрес массива, куда будут записаны требуемые данные.

*param* – параметр для чтения данных.

Возвращаемое значение

*0* – нормальное выполнение функции.

*-1* – не было подключения к программе через компонент *Unit*.

*-2* – хэндлера окна запускаемого приложения не существует.

*-3* – не хватает хэндлеров памяти.

*-4* – не хватает хэндлеров памяти.

*-7* – по запрашиваемому параметру нет данных для чтения.

© ООО "ЭТМС" 1992-2024. Все права защищены

***SetParam*** – установка параметра запущенной программы. Для каждой программы существует свой набор параметров управления. Значения параметров для каждой программы представлены в соответствующем разделе.

*long* ***SetParam****(long param, double value)*

Параметры

*param* – номер устанавливаемого параметра.
*value* – значение устанавливаемого параметра.

Возвращаемое значение

*0* – нормальное выполнение функции.
*-1* – не было подключения к программе через компонент *Unit*.
*-2* – запущенная через компонент *Unit* программа выгружена из памяти.
*-3* – не хватает хэндлеров памяти.
*-4* – не хватает хэндлеров памяти.

***SetParamString*** – установка строкового параметра запущенной программы. Для каждой программы существует свой набор параметров управления. Значения параметров для каждой программы представлены в соответствующем разделе.

*long* ***SetParamString*** *(long param, signed char * StringVal)*

Параметры

*param* – номер устанавливаемого параметра.
*StringVal* – значение устанавливаемого строкового параметра.

Возвращаемое значение

*0* – нормальное выполнение функции.
*-1* – не было подключения к программе через компонент *Unit*.
*-2* – запущенная через компонент *Unit* программа выгружена из памяти.
*-3* – не хватает хэндлеров памяти.
*-4* – не хватает хэндлеров памяти.
*-5* – превышена максимальная допустимая длина строкового параметра.

***SetParamTimeString*** – установка строкового параметра запущенной программы с синхронизацией по времени ZETServer. Для каждой программы существует свой набор параметров управления. Значения параметров для каждой программы представлены в соответствующем разделе.

*long* ***SetParamTimeString*** *(long param, LPCTSTR StringVal, DOUBLE timer)*

Параметры

*param* – номер устанавливаемого параметра.
*StringVal* – значение устанавливаемого строкового параметра.
*timer* – значение времени ZETServer.

Возвращаемое значение

© ООО "ЭТМС" 1992-2024. Все права защищены

*0* – нормальное выполнение функции.
*-1* – не было подключения к программе через компонент *Unit*.
*-2* – запущенная через компонент *Unit* программа выгружена из памяти.
*-3* – не хватает хэндлеров памяти.
*-4* – не хватает хэндлеров памяти.
*-5* – превышена максимальная допустимая длина строкового параметра.

***ShowUnit*** – показать или спрятать запущенную через компонент *Unit* программу.

*long* ***ShowUnit****(short* `View`*)*

Параметры
`View` – параметр видимости программы. *0* – спрятать программу, *1* – показать программу.
Возвращаемое значение
*0* – нормальное выполнение функции.
*-1* – не было подключения к программе через компонент *Unit*.
*-2* – запущенная через компонент *Unit* программа выгружена из памяти.
*-3* – не хватает хэндлеров памяти.
*-4* – не хватает хэндлеров памяти.

***SetUSize*** – изменение размеров области видимости запущенной через компонент *Unit* программы.

*long* ***SetUSize****(double left, double top, double width, double height)*

Параметры
*left* – левая граница области видимости.
*top* – верхняя граница области видимости.
*width* – ширина области видимости.
*height* – высота области видимости.

Возвращаемое значение
*0* – нормальное выполнение функции.
*-1* – не было подключения к программе через компонент *Unit*.
*-2* – запущенная через компонент *Unit* программа выгружена из памяти.
*-3* – не хватает хэндлеров памяти.
*-4* – не хватает хэндлеров памяти.

***MoveUnit*** – перемещение окна запущенной через компонент *Unit* программы.

*long* ***MoveUnit*** *(double x, double y)*

© ООО "ЭТМС" 1992-2024. Все права защищены

Параметры

*x* – абсцисса точки перемещения.

y – ордината точки перемещения.

Возвращаемое значение

*0* – нормальное выполнение функции.

*-1* – не было подключения к программе через компонент *Unit*.

*-2* – запущенная через компонент *Unit* программа выгружена из памяти.

*-3* – не хватает хэндлеров памяти.

*-4* – не хватает хэндлеров памяти.

***GetUnitWidth***, ***GetUnitHeight***, ***GetUnitLeft***, ***GetUnitTop*** – узнать ширину, высоту, левую границу и верхнюю границу окна запущенной через компонент *Unit* программы.

*double* ***GetUnitWidth****(void)*

*double* ***GetUnitHeight*** *(void)*

*double* ***GetUnitLeft*** *(void)*

*double* ***GetUnitTop*** *(void)*

Возвращаемое значение

*≥ 0* – соответствующее значение параметров окна программы.

*-1* – не было подключения к программе через компонент *Unit*.

*-2* – запущенная через компонент *Unit* программа выгружена из памяти.

*-3* – не хватает хэндлеров памяти.

*-4* – не хватает хэндлеров памяти.

***IsReady*** – запрос на наличие данных от запущенной через компонент *Unit* программы.

*long* ***IsReady*** *()*

Возвращаемое значение

*0* – нет доступных данных.

*1* – есть доступные данные.

*long* ***WriteUnitData****(LPCTSTR FileName)* – позволяет записать массив данных в файле. Файл записывается в текстовом виде со всеми комментариями. Это позволяет использовать данные из этого файла для других программ обработки. *FileName* – строка полного пути и имени файла (в настоящее время не используется)

© ООО "ЭТМС" 1992-2024. Все права защищены

*long* ***UnitRecFile****(signed char * FileName)*- позволяет принимать массив данных из файла. *FileName* – строка полного пути и имени файла. Файл читается в текстовом виде со всеми комментариями (в настоящее время не используется)

## События

***Ready*** – возникает при обновлении передаваемых данных из запущенной через компонент *Unit* программы через функции: *UnitWrite(...)* через программный модуль *Unit*. Например, если запустить через компонент *Unit* какую-либо программу, которая рассчитывает различные значения и передает их по интерфейсу *Unit*, то в обработчике данного события можно организовать чтение данных с помощью метода *Read(...)*

*void* ***Ready*** *(long par)*

Параметры

par – параметр, означающий готовность данных определенного типа. Для каждой программы существует свой набор параметров, который представлен в соответствующем разделе.

***Load*** – Данное событие сразу после завершения загрузки программs.

*void* ***Load*** *()*

***NotActive*** – Данное событие генерируется в случае, когда запущенная программа прекращает сообщать в UNIT о своей деятельности.

*void* ***NotActive*** *()*

***Lost*** – Данное событие генерируется в случае, когда запущенная программа завершает свою работу не по команде от UNIT.

*void* ***Lost*** *()*

© ООО "ЭТМС" 1992-2024. Все права защищены

# ZETPath.ocx Позволяет работать со стандартными каталогами ZETLab

**Назначение**

Позволяет работать со стандартными каталогами ZETLab, такими как каталог установки, а также каталоги с файлами записанных сигналов, результатами обработки, файлами конфигурации, файлами справки и т.д.

Данный компонент нужен для удобства использования стандартных каталогов ZETLab и содержащихся в них файлов.

Используется во всех программах ZETLab.

Пути хранятся в реестре Windows, при этом используется две ветки: глобальная (HKLM) и пользовательская (HKCU).

Пути в глобальное ветке назначаются при установке ZETLab и не могут быть изменены.

Пути в пользовательской ветке могут быть переопределены программами для текущего пользователя.

Некоторые пути не хранятся в прямом виде, а генерируются автоматически из других путей.

## Глава 1.Установка компонента

Для работы с модулем *ZETPath* его необходимо сначала установить в программу. При загрузке программы необходимо загрузить с помощью компонента необходимую программу, разместить окно программы на экране в удобном месте, установить необходимые параметры в программе и затем по событиям, происходящим от программы считывать данные из программы.

Для установки компонента *ZETPath* в проект *Microsoft Visual Studio 2010*, необходимо кликнуть правой кнопкой мыши на форме диалога и из появившегося меню выбрить пункт *Insert ActiveX Control*...(рисунок 1.1)

© ООО "ЭТМС" 1992-2024. Все права защищены

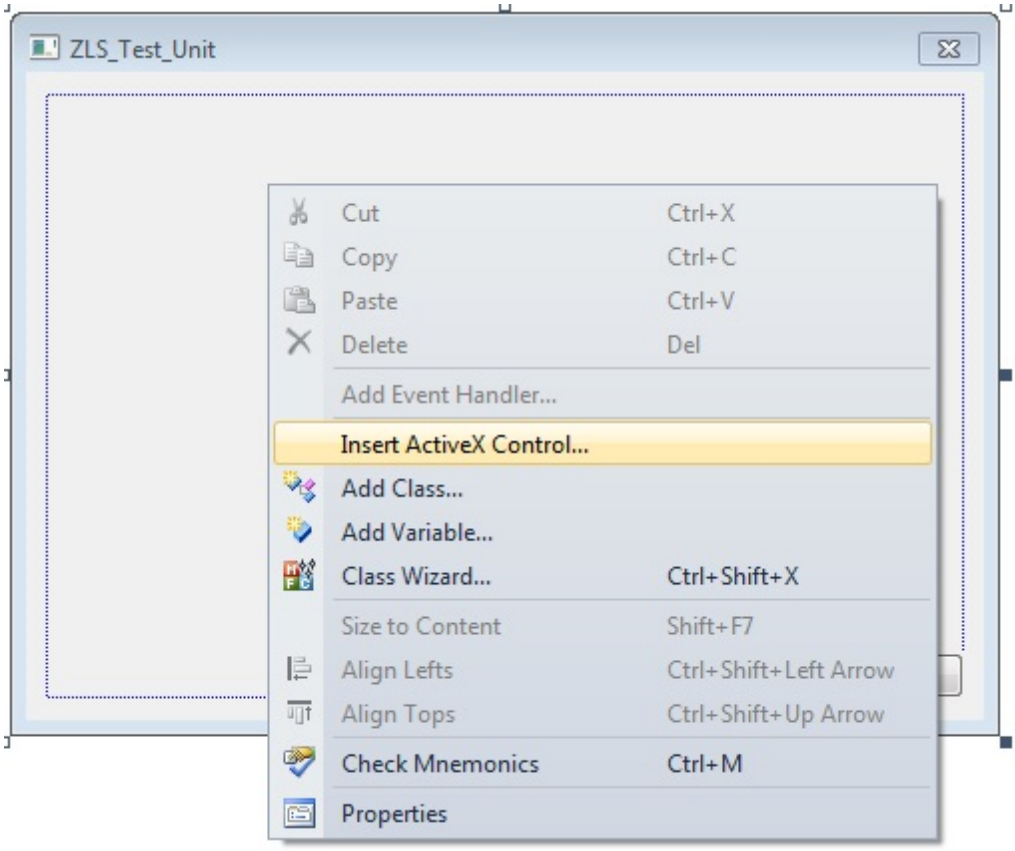



Рисунок 1.1

Из появившегося диалогового окна следует выбрать компонент *ZETPath Control* и нажать *OK* (рисунок 1.2).

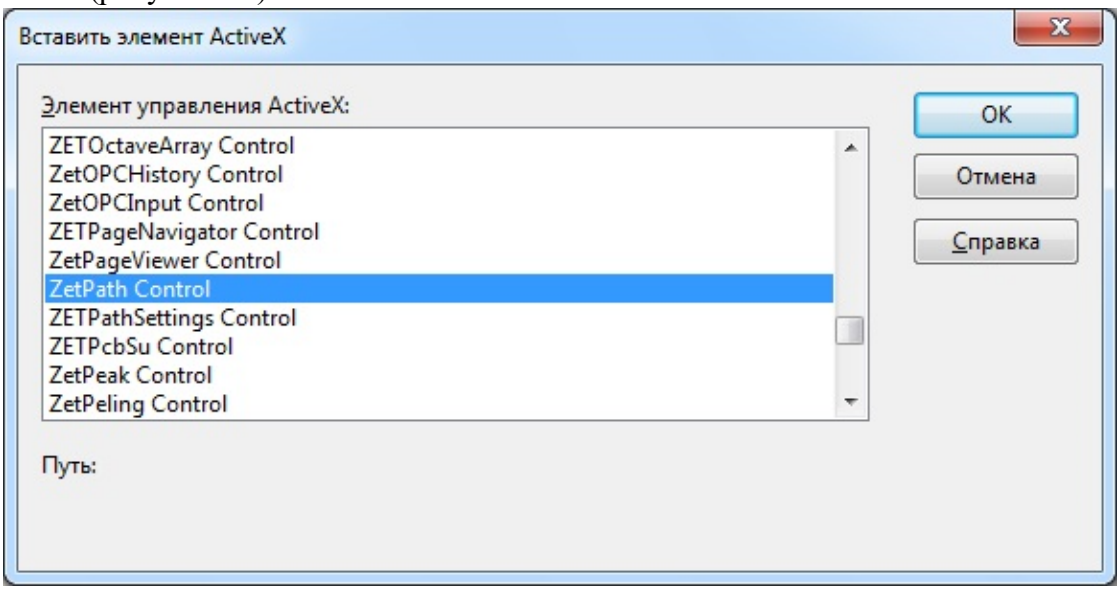



Рисунок 1.2

После этого компонент *ZETPath.ocx* появится на форме диалога (рисунок 1.3).

© ООО "ЭТМС" 1992-2024. Все права защищены

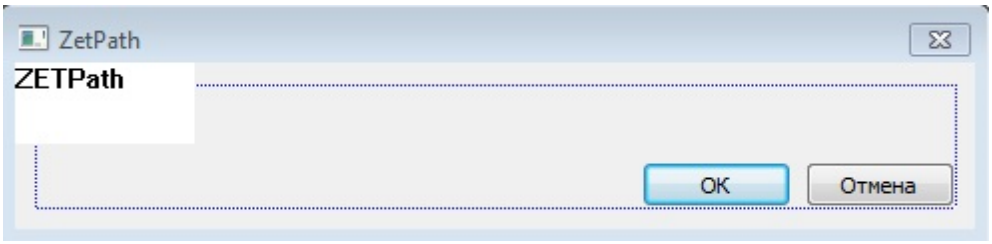



Рисунок 1.3

Одна установленная на форме компонента позволяет управлять одной программой. При необходимости управлять несколькими программами, необходимо установить на форму несколько компонент. Для того чтобы компонента не была видна на форме во время выполнения программы, необходимо установить свойство *Visible* равным *False*.

# Глава 2.Описание свойств, методов и событий в idl

## Свойства

*BSTR ZetPath( BSTR* param );*
*BSTR ZetPathW( BSTR* param );*
Запрос пути к одному из стандартных каталогов ZETLab.
Возвращается абсолютный путь к указанному каталогу.

param — тип каталога в виде строки (регистры символов игнорируются):
"InstallLocation" — каталог установки ZETLab (HKLM);
"DirConfig" — каталог с файлами конфигурации (HKCU);
"DirSignal" — каталог с файлами записанных сигналов (HKCU);
"DirResult" — каталог с результатами обработки (HKCU);
"DirCorrect" — каталог с файлами программных калибровок (HKLM);
"DirHelp" — каталог с файлами справки (автоматически из "InstallLocation");
"DirScada" — каталог установки SCADA ZETView (HKLM);
"DirScadaHelp" — каталог с фалами справки SCADA ZETView (автоматически из "DirScada");
"DirWatchDogTimer" — каталог с файлами таймеров (HKLM);
любая другая непустая строка — пользовательский каталог (HKCU, при наличии соответствующей записи в реестре).

*LONG ZetLetPath( BSTR* param, BSTR* path );*
*LONG ZetLetPathW( BSTR* param, BSTR* path );*
*LONG ZetSetPath( BSTR* param, BSTR* path );*
Изменение пути к одному из каталогов ZETLab и сохранение пути в пользовательской ветке реестра Windows.

© ООО "ЭТМС" 1992-2024. Все права защищены

Возвращается 0 в случае успеха.

param — тип каталога (только HKCU, см. описание функции ZetPath).

path — новый абсолютный путь к указанному каталогу.

*BSTR GetNextSignals();*

Создание очередного подкаталога в каталоге сигналов для записи очередного файла сигналов.

Новый подкаталог имеет имя в формате "sYYMMDD_HHMMSS" с использованием текущего времени.

Возвращается абсолютный путь к созданному подкаталогу.

*BSTR GetNextSignalsByDate( LONGLONG date );*

Создание очередного подкаталога в каталоге сигналов для записи очередного файла сигналов.

Новый подкаталог имеет имя в формате "sYYMMDD_HHMMSS" с использованием указанного времени.

Возвращается абсолютный путь к созданному подкаталогу.

data — дата и время, используемые для формирования имени подкаталога, в формате time64_t.

*BSTR GetNextFileInDir( BSTR* dir, BSTR* file, BSTR* ext );*

Получение очередного пронумерованного имени файла.

Сначала производится поиск максимального номера по всем файлам "dir\file%d.ext".

Затем возвращается новое имя файла с номером, большим максимального на 1.

dir — абсолютный путь к каталогу, в котором производится поиск.

file — начальная часть имени файла (за исключением номера).

ext — расширение файла.

# Глава 3.Перечень функций, используемые ZETPath.ocx в базовом классе

## Свойства

© ООО "ЭТМС" 1992-2024. Все права защищены

***CString*** *ZetPath( BSTR* param )*
***CString*** *ZetPathW( BSTR* param )*

Запрос пути к одному из стандартных каталогов ZETLab.
Возвращается абсолютный путь к указанному каталогу.

param — тип каталога в виде строки (регистры символов игнорируются):
"InstallLocation" — каталог установки ZETLab (HKLM);
"DirConfig" — каталог с файлами конфигурации (HKCU);
"DirSignal" — каталог с файлами записанных сигналов (HKCU);
"DirResult" — каталог с результатами обработки (HKCU);
"DirCorrect" — каталог с файлами программных калибровок (HKLM);
"DirHelp" — каталог с файлами справки (автоматически из "InstallLocation");
"DirScada" — каталог установки SCADA ZETView (HKLM);
"DirScadaHelp" — каталог с фалами справки SCADA ZETView (автоматически из "DirScada");
"DirWatchDogTimer" — каталог с файлами таймеров (HKLM);
любая другая непустая строка — пользовательский каталог (HKCU, при наличии соответствующей записи в реестре).

***long*** *ZetLetPath( BSTR* param, BSTR* path )*
***long*** *ZetLetPathW( BSTR* param, BSTR* path )*
***long*** *ZetSetPath( BSTR* param, BSTR* path )*

Изменение пути к одному из каталогов ZETLab и сохранение пути в пользовательской ветке реестра Windows.
Возвращается 0 в случае успеха.

param — тип каталога (только HKCU, см. описание функции ZetPath).

path — новый абсолютный путь к указанному каталогу.

***CString*** *GetNextSignals()*

Создание очередного подкаталога в каталоге сигналов для записи очередного файла сигналов.
Новый подкаталог имеет имя в формате "sYYMMDD_HHMMSS" с использованием текущего времени.
Возвращается абсолютный путь к созданному подкаталогу.

***CString*** *GetNextSignalsByDate( LONGLONG date )*

© ООО "ЭТМС" 1992-2024. Все права защищены

Создание очередного подкаталога в каталоге сигналов для записи очередного файла сигналов.

Новый подкаталог имеет имя в формате "sYYMMDD_HHMMSS" с использованием указанного времени.

Возвращается абсолютный путь к созданному подкаталогу.

data — дата и время, используемые для формирования имени подкаталога, в формате time64_t.

***CString*** *GetNextFileInDir( BSTR* dir, BSTR* file, BSTR* ext )*

Получение очередного пронумерованного имени файла.

Сначала производится поиск максимального номера по всем файлам "dir\file%d.ext".

Затем возвращается новое имя файла с номером, большим максимального на 1.

dir — абсолютный путь к каталогу, в котором производится поиск.

file — начальная часть имени файла (за исключением номера).

ext — расширение файла.

© ООО "ЭТМС" 1992-2024. Все права защищены

# SRV.ocx для работы с ZETLab

**Назначение**

- Модуль предназначен для создания программных комплексов, позволяющих:
- запускать программы из набора программ ZETLab, поддерживающих интерфейс компонента Unit;
- устанавливать параметры в программах обработки сигналов;
- получать результаты обработки от программ;
- скрывать или открывать внешний вид программ.
- Модуль предназначен для работы с устройствами АЦП-ЦАП, для обработки сигналов в реальном времени и обработки оцифрованных сигналов, записанных в файлы. Модуль обеспечивает обмен данными и потоками данных с драйверами, и другими пользовательскими программами по COM-интерфейсу с использованием технологии клиент-сервер. Модуль поддерживает одновременную работу с несколькими платами АЦП-ЦАП различного или одинакового типа. Суммарное количество каналов не более 200. Суммарное количество работающих устройств в системе не более 50. Оцифрованные данные в программу пользователя передаются в формате плавающей запятой одинарной точности, в заданных единицах измерения. Различные модули АЦП преобразуют аналоговый сигнал в 12, 14, 16 или 24 разрядный код. Компонент SRV преобразует эти целочисленные значения в формат плавающей запятой с учетом коэффициентов усиления, чувствительности преобразователей, смещения постоянной составляющей, поправками по измерительным каналам. Эти параметры задаются в программе «Диспетчер устройств» из группы «Сервисные».

## Глава 1.Установка компонента SRV.ocx

Для работы с модулем *SRV* его необходимо сначала установить в программу, а далее использовать его свойства, методы и события. При запуске программы пользователя, в самом начале программы необходимо подключиться к модулям АЦП-ЦАП. В зависимости от типа программы используются различные схемы подключения. При выходе из программы необходимо отключиться от модулей АЦП-ЦАП. Для каждой схемы подключения существует своя схема отключения.

При работе с компонентом *SRV* можно опросить и установить параметры устройств АЦП-ЦАП. Компонент *SRV* обеспечивает одновременную работу нескольких программ (до 200). Основным понятием при обработке сигналов в реальном времени является текущее время. Текущее время сервера показывает, сколько секунд прошло со

© ООО "ЭТМС" 1992-2024. Все права защищены

времени запуска АЦП и ЦАП. Пользовательская программа должна постоянно следить за текущим временем сервера при помощи соответствующего метода и при увеличении текущего времени на определенный интервал запрашивать у сервера данные. Интервал в программе может быть произвольным. Текущее время для ввода данных АЦП показывает, что в буфере есть данные для обработки до этого момента времени. Текущее время для вывода данных ЦАП показывает, что данные из буфера до этого момента времени переданы в тракт ЦАП. Пользователь должен загружать данные в буфер ЦАП с опережением, например, на 1 секунду.

Для установки компонента *SRV* в проект *Microsoft Visual Studio 2010*, необходимо кликнуть правой кнопкой мыши на форме диалога и из появившегося меню выбрить пункт *Insert ActiveX Control*...(рисунок 1.1)

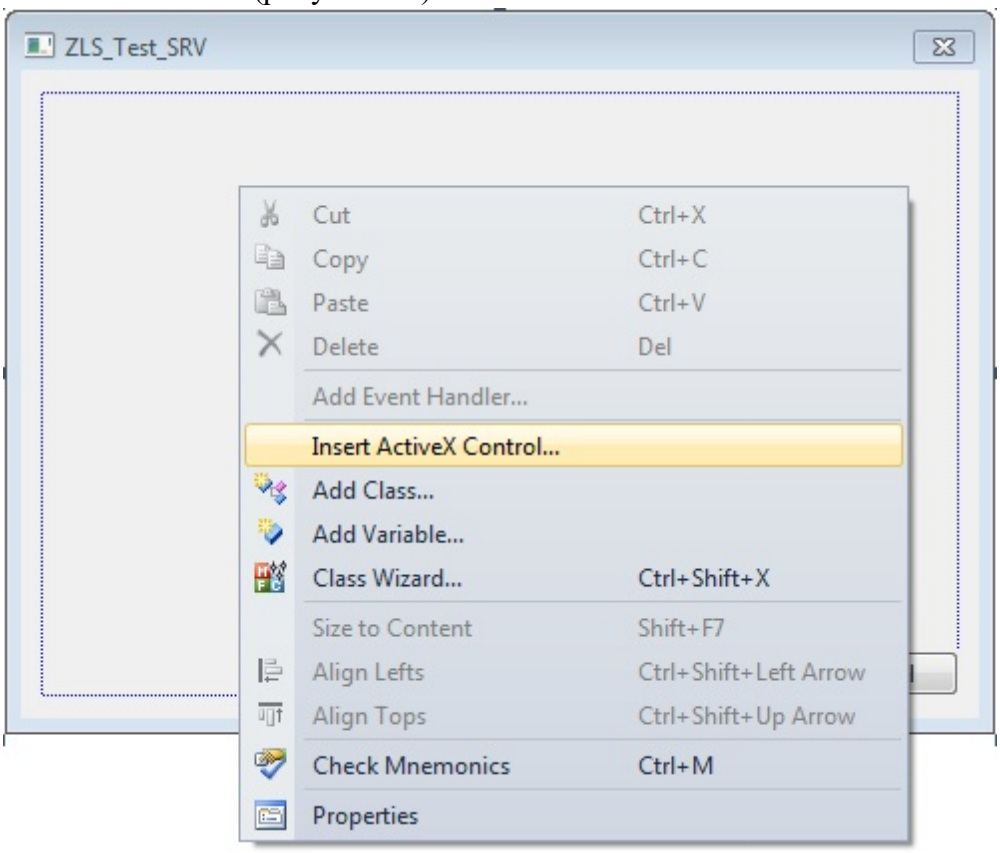



Рисунок 1.1

Из появившегося диалогового окна следует выбрать компонент *SRV Control* и нажать *OK* (рисунок 1.2).

© ООО "ЭТМС" 1992-2024. Все права защищены

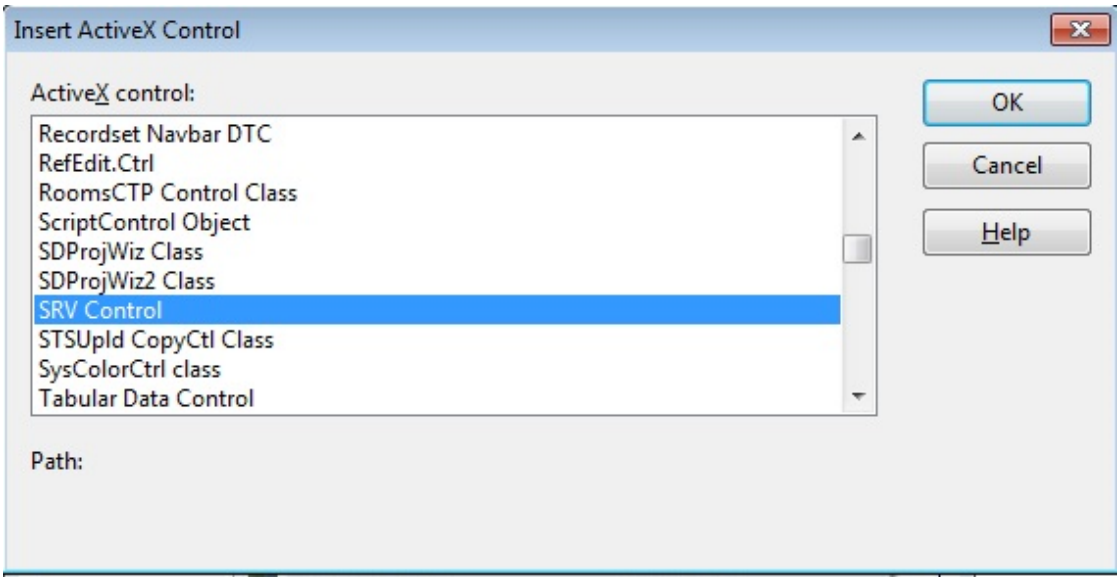



Рисунок 1.2

После этого компонент Srv.*ocx* появится на форме диалога (рисунок 1.3).

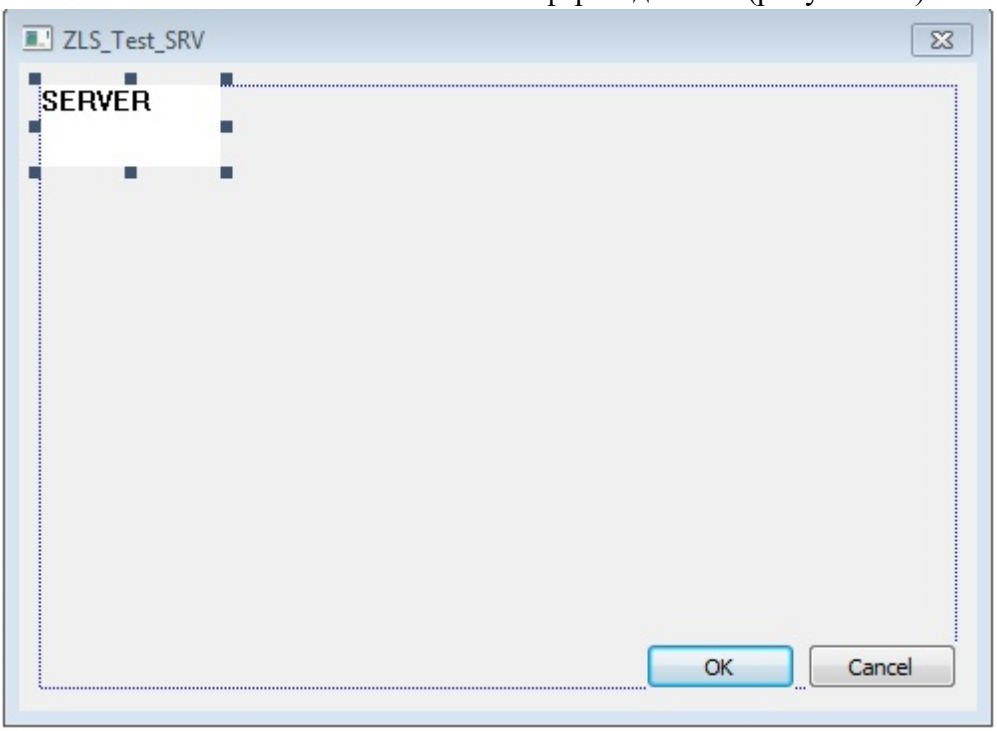



Рисунок 1.3

Одна установленная на форме компонента позволяет управлять одной программой. При необходимости управлять несколькими программами, необходимо установить на форму несколько компонент. Для того чтобы компонента не была видна на форме во время выполнения программы, необходимо установить свойство *Visible* равным *False*.

© ООО "ЭТМС" 1992-2024. Все права защищены

# Глава 2.Описание методов и событий в idl

## 2.1.Методы в SRV.ocx

### Подключение и отключение

***Connect*** – подключение к серверу данных для программ обработки данных от АЦП или файлов, при этом запускаются АЦП, если они не запущены. Метод используется один раз при инициализации программы. Пользовательской программе доступны все измерительные каналы, виртуальные каналы, созданные в других программах, виртуальные каналы от ЦАП, если они созданы в других программах.

*long* ***Connect*** *(void)*

Возвращаемое значение

1 - данные содержат не менее одного значения NaN

0 – нормальное выполнение функции.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа ZETServer.ocx выгружена из памяти процессора.

-3 – параметр channel < 0.

-4 – параметр channel > (QuanChan() – 1).

-5 – заданный канал не является каналом модуля АЦП-ЦАП.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

***Disconnect*** – отключение от сервера данных. Метод используется при выходе из программы, работающей с данными АЦП.

*long* ***Disconnect*** *(void)*

Возвращаемое значение

1 - данные содержат не менее одного значения NaN

© ООО "ЭТМС" 1992-2024. Все права защищены

0 – нормальное выполнение функции.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа ZETServer.ocx выгружена из памяти процессора.

-3 – параметр channel < 0.

-4 – параметр channel > (QuanChan() – 1).

-5 – заданный канал не является каналом модуля АЦП-ЦАП.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

***ConnectDac*** – подключение к серверу данных для программ генерации сигналов для ЦАП и работы с данными от АЦП, при этом запускаются ЦАП, если они не запущены. Метод используется один раз при инициализации программы. Пользовательской программе доступны все измерительные каналы, виртуальные каналы, созданные в других программах, виртуальные каналы от ЦАП, если они созданы в других программах.

*long* ***ConnectDac*** *(long NumberDac)*

Параметры

*NumberDac* – номер канала ЦАП. Формирование данных для ЦАП производится через сервер данных и включается *lNumberDac* канал ЦАП. Для подключения дополнительных каналов ЦАП необходимо пользоваться функциями *SetChanDac(...)* – подключение канала ЦАП, *RemChanDac(...)* – освобождение канала ЦАП. Для определения, что канал ЦАП не подключен к другой программе, существует функция *IsFreeChanDac(...)*. Каждый канал имеет свой идентификатор. Методы *IdChan(...)* и *IdChanDac(...)* возвращают идентификатор канала. Метод *GetCurrentTime(...)* возвращает текущее время в буфере уже переданных данных в ЦАП. Пользователь должен записывать данные в буфер ЦАП с опережением, например, на 1 секунду.

Возвращаемое значение

1 - данные содержат не менее одного значения NaN

0 – нормальное выполнение функции.

© ООО "ЭТМС" 1992-2024. Все права защищены

-1 – на момент вызова метода не было подключения к серверу данных.
-2 – программа ZETServer.ocx выгружена из памяти процессора.
-3 – параметр channel < 0.
-4 – параметр channel > (QuanChan() – 1).
-5 – заданный канал не является каналом модуля АЦП-ЦАП.
-6 – серверу данных не хватает памяти для организации буферов данных.
-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.
-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().
-9 – номер декады decade < 0.
-10 – номер декады decade > 4.
-11 – момент времени time больше текущего времени для данного канала.
-12 – момент времени time < 0.
-13 – указатель на массив данных data = NULL.
-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".
-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().
-16 - пустой указатель на буфер.
-20 - превышено время обработки функции.

***DisconnectDac*** – отключение от сервера данных в режиме работы с ЦАП. Метод используется при выходе из программы, работающей с данными от АЦП и формирующей сигналы для ЦАП.

*long* ***DisconnectDac*** *(long NumberDac)*

Параметры

*NumberDac* – номер канала ЦАП, который освобождается программой. В программе пользователя при работе с каналами ЦАП, необходимо подключаться и отключаться от выбранных каналов ЦАП командами *SetChanDac(...)* и *RemChanDac(...)*. Если не происходит отключения от канала ЦАП, то этот канал остается включенным и по этому каналу воспроизводится нулевой уровень.

Возвращаемое значение

1 - данные содержат не менее одного значения NaN
0 – нормальное выполнение функции.
-1 – на момент вызова метода не было подключения к серверу данных.
-2 – программа ZETServer.ocx выгружена из памяти процессора.
-3 – параметр channel < 0.
-4 – параметр channel > (QuanChan() – 1).
-5 – заданный канал не является каналом модуля АЦП-ЦАП.
-6 – серверу данных не хватает памяти для организации буферов данных.
-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.
-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

© ООО "ЭТМС" 1992-2024. Все права защищены

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

***ConnectVrtCh*** – подключение к серверу для программ, создающих дополнительные виртуальные каналы и работающих с данными от АЦП. Метод используется один раз при инициализации программы. Пользовательской программе доступны все измерительные каналы, виртуальные каналы, созданные в других программах, виртуальные каналы от ЦАП, если они созданы в других программах.

*long* ***ConnectVrtCh*** *(long NumChan)*

Параметры

*NumChan* – количество дополнительных виртуальных каналов, организуемые программой. Пользовательская программа должна дать информацию о каждом виртуальном канале методом *VrtChnInfo(...)*, а затем периодически записывать в буфер данных виртуального канала данные методом *PutData(...).* Эти виртуальные каналы становятся доступными для всех других программ, как обычные каналы. Если в любой пользовательской программе сделать запрос методом *CurrentTime(...)* по данному виртуальному каналу, то это время будет соответствовать времени указанному в *PutData(...)*. Если в виртуальный канал не записывать данные, то и время по этому каналу не будет идти.

Возвращаемое значение

1 - данные содержат не менее одного значения NaN

0 – нормальное выполнение функции.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа ZETServer.ocx выгружена из памяти процессора.

-3 – параметр channel < 0.

-4 – параметр channel > (QuanChan() – 1).

-5 – заданный канал не является каналом модуля АЦП-ЦАП.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

© ООО "ЭТМС" 1992-2024. Все права защищены

-12 – момент времени time < 0.
-13 – указатель на массив данных data = NULL.
-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".
-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().
-16 - пустой указатель на буфер.
-20 - превышено время обработки функции.

***DisconnectVrtCh*** – отключение от сервера данных в режиме работы с дополнительными виртуальными каналами. Метод используется при выходе из программы, создающей виртуальные каналы.

*long* ***DisconnectVrtCh*** *(void)*

Возвращаемое значение
0 – нормальное выполнение функции.

***Start*** – запустить АЦП и ЦАП.

*long* ***Start*** *(void)*

***Stop*** – остановить АЦП и ЦАП.

*long* ***Stop*** *(void)*

## Опрос параметров каналов системы

***QuanChan*** – опрос количества работающих каналов на сервере данных

*long* ***QuanChan*** *(void)*

Возвращаемое значение
≥ 0 – суммарное количество работающих каналов на сервере (каналы АЦП, виртуальные каналы, виртуальные каналы ЦАП)
-1 – на момент вызова метода не было подключения к серверу данных.
-2 – программа ZETServer.ocx выгружена из памяти процессора.
-3 – параметр channel < 0.
-4 – параметр channel > (QuanChan() – 1).
-5 – заданный канал не является каналом модуля АЦП-ЦАП.
-6 – серверу данных не хватает памяти для организации буферов данных.
-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.
-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().
-9 – номер декады decade < 0.
-10 – номер декады decade > 4.
-11 – момент времени time больше текущего времени для данного канала.

© ООО "ЭТМС" 1992-2024. Все права защищены

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

***WorkChanADC*** – опрос количества работающих каналов на модуле АЦП-ЦАП, которому принадлежит заданный канал.

*long* ***WorkChanADC*** *(long channel)*

Параметры

*channel* – номер канала, принадлежащий тому модулю АЦП-ЦАП, количество работающих каналов которого требуется узнать. Параметр *channel* может принимать значение от *0* до (*QuanChan() – 1)*.

Возвращаемое значение

≥ 0 – суммарное количество работающих каналов на модуле АЦП-ЦАП, которому принадлежит заданный канал.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа ZETServer.ocx выгружена из памяти процессора.

-3 – параметр channel < 0.

-4 – параметр channel > (QuanChan() – 1).

-5 – заданный канал не является каналом модуля АЦП-ЦАП.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

***MaxQuanChanDac*** – опрос суммарного количества каналов ЦАП, которые можно задействовать в системе.

© ООО "ЭТМС" 1992-2024. Все права защищены

*long **MaxQuanChanDac** (void)*

Возвращаемое значение

≥ 0 – суммарное количество каналов ЦАП, которые можно задействовать в системе

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа ZETServer.ocx выгружена из памяти процессора.

-3 – параметр channel < 0.

-4 – параметр channel > (QuanChan() – 1).

-5 – заданный канал не является каналом модуля АЦП-ЦАП.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

***Commentary*** – опрос имени (названия) заданного канала.

*BSTR **Commentary** (long channel)*

Параметры

*channel* – номер канала, имя (название) которого требуется узнать. Имя (название) канала задается в таблице конфигурации *Devices.cfg* и программой «Диспетчер устройств» из группы «Сервисные». Параметр *channel* может принимать значение от *0* до (*QuanChan() – 1)*.

Возвращаемое значение

пустая строка – на момент вызова метода не было подключения к серверу данных, или программа *ZETServer.ocx* выгружена из памяти процессора, или параметр *channel < 0*, или параметр *channel* > (*QuanChan() – 1)*.

непустая строка – имя заданного канала.

***Conversion*** – опрос единица измерения по заданному каналу

*BSTR **Conversion** (long channel)*

Параметры

© ООО "ЭТМС" 1992-2024. Все права защищены

*channel* – номер канала, единицу измерения по которому требуется узнать. Единица измерения по каналу задается в таблице конфигурации *Devices.cfg* и программой «Диспетчер устройств» из группы «Сервисные». Параметр *channel* может принимать значение от *0* до (*QuanChan() – 1)*.

Возвращаемое значение

пустая строка – на момент вызова метода не было подключения к серверу данных, или программа *ZETServer.ocx* выгружена из памяти процессора, или параметр *lChannel < 0*, или параметр *channel* > (*QuanChan() – 1)*.

непустая строка – единица измерения по заданному каналу.

***MaxLevel*** – опрос максимально допустимого уровня по заданному каналу в единицах измерения (максимальный входной диапазон).

*float* ***MaxLevel*** *(long channel)*

Параметры

*channel* – номер канала, максимально допустимый уровень которого требуется узнать. Параметр *channel* может принимать значение от *0* до (*QuanChan() – 1)*.

Возвращаемое значение

≥ 0 – максимально допустимый уровень по заданному каналу в единицах измерения

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа ZETServer.ocx выгружена из памяти процессора.

-3 – параметр channel < 0.

-4 – параметр channel > (QuanChan() – 1).

-5 – заданный канал не является каналом модуля АЦП-ЦАП.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

***MinLevel*** – опрос минимально различимого уровня по заданному каналу в единицах измерения (вес младшего разряда по каналу)

© ООО "ЭТМС" 1992-2024. Все права защищены

*float* ***MinLevel*** *(long channel)*

Параметры

*lChannel* – номер канала, минимально различимый уровень которого требуется узнать. Параметр *channel* может принимать значение от *0* до (*QuanChan() – 1)*.

Возвращаемое значение

≥ 0 – минимально допустимый уровень по заданному каналу в единицах измерения

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа ZETServer.ocx выгружена из памяти процессора.

-3 – параметр channel < 0.

-4 – параметр channel > (QuanChan() – 1).

-5 – заданный канал не является каналом модуля АЦП-ЦАП.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

***Reference*** – опрос значения опоры для расчета уровней в децибелах по заданному каналу.

*float* ***Reference*** *(long channel)*

Параметры

*channel* – номер канала, значение опоры по которому требуется узнать. Значение опоры по каналу задается в таблице конфигурации *Devices.cfg* и программой «Диспетчер устройств» из группы «Сервисные». Параметр *lChannel* может принимать значение от *0* до (*QuanChan() – 1)*.

Возвращаемое значение

≥ 0 – значение опоры для расчета уровней в децибелах по заданному каналу.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа ZETServer.ocx выгружена из памяти процессора.

-3 – параметр channel < 0.

-4 – параметр channel > (QuanChan() – 1).

-5 – значение частоты дискретизации по заданному каналу меньше нуля.

-6 – серверу данных не хватает памяти для организации буферов данных.

© ООО "ЭТМС" 1992-2024. Все права защищены

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

***ShifLevel*** – опрос смещения постоянной составляющей по заданному каналу в единицах измерения.

*float* ***ShifLevel*** *(long channel)*

Параметры

*channel* – номер канала, значение опоры по которому требуется узнать. Смещение постоянной составляющей по каналу задается в таблице конфигурации *Devices.cfg* и программой «Диспетчер устройств» из группы «Сервисные». Параметр *channel* может принимать значение от *0* до (*QuanChan() – 1)*.

Возвращаемое значение

любое значение – смещение постоянной составляющей по заданному каналу в единицах измерения.

***Sense*** – опрос чувствительности (коэффициента преобразования) заданного канала в вольтах на единицу измерения.

*float* ***Sense*** *(long channel)*

Параметры

*channel* – номер канала, чувствительность которого требуется узнать. Чувствительность задается в таблице конфигурации *Devices.cfg* и программой «Диспетчер устройств» из группы «Сервисные». Параметр *channel* может принимать значение от *0* до (*QuanChan() – 1)*.

Возвращаемое значение

любое значение – чувствительность (коэффициент преобразования) заданного канала в вольтах на единицу измерения.

© ООО "ЭТМС" 1992-2024. Все права защищены

***AFCH*** – опрос названия файла, в котором может храниться пользовательская информация о канале, например, частотно-зависимые поправки АЧХ тракта.

*BSTR* ***AFCH*** *(long channel)*

Параметры

*channel* – номер канала, название вспомогательного файла которого требуется узнать. Название вспомогательного файла задается в таблице конфигурации *Devices.cfg* и программой «Диспетчер устройств» из группы «Сервисные». Параметр *channel* может принимать значение от *0* до (*QuanChan() – 1)*.

Возвращаемое значение

пустая строка – на момент вызова метода не было подключения к серверу данных, или программа *ZETServer.ocx* выгружена из памяти процессора, или параметр *channel < 0*, или параметр *channel >* (*QuanChan() – 1)*.

непустая строка – название файла, в котором может храниться пользовательская информация о заданном канале.

***CurLevel*** – опрос нормированного текущего значения по заданному каналу (для контроля перегрузки).

*float* ***CurLevel*** *(long channel)*

Параметры

*channel* – номер канала, нормированный текущий уровень которого требуется узнать. Сервер данных нормирует получаемый текущий уровень по максимальному уровню по заданному каналу. Параметр *channel* может принимать значение от *0* до (*QuanChan() – 1)*.

Возвращаемое значение

≥ 0 и ≤ 1 – нормированный текущий уровень по заданному каналу.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа ZETServer.ocx выгружена из памяти процессора.

-3 – параметр channel < 0.

-4 – параметр channel > (QuanChan() – 1).

-5 – заданный канал не является каналом модуля АЦП-ЦАП.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

© ООО "ЭТМС" 1992-2024. Все права защищены

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

***GetStatus*** – опрос типа заданного канала

*long* ***GetStatus*** *(long channel)*

Параметры

*channel* – номер канала, тип которого требуется узнать. Параметр *channel* может принимать значение от *0* до (*QuanChan() – 1)*.

Возвращаемое значение:

-4: channel >= QuanChan.

-3: channel < 0.

-2: не запущен ZetServer.exe.

-1: не было Connect.

0: заданный канал является каналом АЦП.

1: заданный канал является каналом ЦАП.

2: заданный канал является виртуальным каналом.

3: заданный канал является цифровым каналом.

4: заданный канал является отключенным каналом устройства.

5: заданный канал является каналом отключенного устройства.

6: заданный канал является интеллектуальным датчиком.

7: заданный канал является каналом отключенного интеллектуального датчика.

8: заданный канал является каналом скоростного устройства, которое выдаёт данные порциями по таймеру или по событию.

9: неактивный виртуальный канал.

10: спрятанный канал АЦП.

11: спрятанный канал ЦАП.

## Опрос физических параметров каналов

***QuanDSP*** **-** опрос количества устройств АЦП-ЦАП (сигнальных процессоров) в системе.

*long* ***QuanDSP*** *(void)*

Возвращаемое значение

≥ 0 – количество устройств АЦП-ЦАП (сигнальных процессоров) в системе.

***QuanPhChan*** – опрос количества физических каналов устройства (сколько может быть включено каналов, а не сколько каналов фактически включено).

*long* ***QuanPhChan*** *(long numDSP)*

Параметры

© ООО "ЭТМС" 1992-2024. Все права защищены

*numDSP* – номер устройства (сигнального процессора), количество физических каналов которого требуется узнать. Параметр *numDSP* может принимать значение от *0* до (*QuanDSP() – 1)*.

Возвращаемое значение

≥ 0 – количество физических каналов устройства.

-1 – параметр *numDSP* < *0*.

-2 – параметр *numDSP* > (максимальное количество устройств в системе – *1)*.

-3 – параметр *numDSP* > (*QuanDSP() – 1)*.

***TypeAdc*** – опрос типа устройства (платы) АЦП-ЦАП, физическим каналом которого является заданный канал.

*long* ***TypeAdc*** *(long channel)*

Параметры

*channel* – номер канала, тип устройства которого требуется узнать. Параметр *channel* может принимать значение от *0* до (*QuanChan() – 1)*.

Возвращаемое значение

0 – АЦП устройства ADC 16/200 (драйвер Kd1610.sys).

1 – АЦП устройства APC 216 (драйвер Kd216.sys).

2 – АЦП устройства ADC 16/500 (драйвер Kd500.sys).

3 – АЦП устройства ADC 16/500P (драйвер Kd500p.sys).

4 – АЦП устройства ADC 816 (драйвер Kd816.sys).

5 – АЦП устройства ADC 1002 (драйвер Kd1002.sys).

6 – АЦП устройства ADC 216 USB (драйвер Kdu216.sys).

7 – АЦП устройства ADC 24 (драйвер Kd24.sys).

8 – АЦП устройства ADC 1432 (драйвер Kd1432.sys).

9 – АЦП устройства ACPB USB (драйвер KduACPB.sys).

10 – АЦП устройства ZET210 USB (драйвер Kdu1616.sys).

11 – АЦП устройства PD14 USB (драйвер KduPD14.sys)

12 – АЦП устройства ZET110 VN USB, (драйвера KduVN.sys);

13 – АЦП устройства ZET302 Osc (драйвер KduOsc.sys).

14 – АЦП устройства ZET017 U8 USB, A17U8 USB (драйвер Kdu8500.sys).

15 – АЦП устройства ZET017-U2, A17U2 (A19-U2 USB) (драйвер Kdu2500.sys).

16 – АЦП устройства ZET220 USB (ZET017 USB Seismic) (драйвер Kdu1624.sys) (24 – разряда, 16 – каналов).

17 – АЦП устройства ZET230 USB (ZET017 USB Audio) (драйвер Kdu0424.sys) (24 – разряда, 4 – канала).

18 – АЦП устройства ZET048 USB (драйвер Kdu0414.sys).

19 – АЦП устройства ZET240 USB(ZET048 USB Seismic) (драйвер Kdu0824.sys) (14 – разрядов, 4 канала).

100 – ЦАП устройства ADC 16/200 (драйвер Kd1610.sys).

101 – ЦАП устройства APC 216 (драйвер Kd216.sys).

102 – ЦАП устройства ADC 16/500 (драйвер Kd500.sys).

103 – ЦАП устройства ADC 16/500P (драйвер Kd500p.sys).

© ООО "ЭТМС" 1992-2024. Все права защищены

104 – ЦАП устройства ADC 816 (драйвер Kd816.sys).
105 – ЦАП устройства ADC 1002 (драйвер Kd1002.sys).
106 – ЦАП устройства ADC 216 USB (драйвер Kdu216.sys).
107 – ЦАП устройства ADC 24 (драйвер Kd24.sys).
108 – ЦАП устройства ADC 1432 (драйвер Kd1432.sys).
109 – ЦАП устройства ACPB USB (драйвер KduACPB.sys).
110 – ЦАП устройства ZET210 USB (драйвер Kdu1616.sys).
111 – ЦАП устройства PD14 USB (драйвер KduPD14.sys)
112 – ЦАП устройства ZET110 VN USB (драйвер KduVN.sys).
113 – ЦАП устройства ZET302 Osc (драйвер KduOsc.sys).
114 – ЦАП устройства ZET017 U8 USB, A17U8 USB (драйвер Kdu8500.sys).
115 – ЦАП устройства ZET017-U2, A17U2 (A19-U2 USB) (драйвер Kdu2500.sys).
116 – ЦАП устройства ZET220 USB (ZET017 USB Seismic) (драйвер Kdu1624.sys).
117 – ЦАП устройства ZET230 USB (ZET017 USB Audio) (драйвер Kdu0424.sys).
118 – ЦАП устройства ZET048 USB (драйвер Kdu0414.sys).
119 – ЦАП устройства ZET240 USB(ZET048 USB Seismic) (драйвер Kdu0824.sys).

Возвращаемое значение

≥ 0 – текущее время по заданному каналу в секундах с момента запуска сервера данных.
-1 – на момент вызова метода не было подключения к серверу данных.
-2 – программа ZETServer.ocx выгружена из памяти процессора.
-3 – параметр channel < 0.
-4 – параметр channel > (QuanChan() – 1).
-5 – заданный канал не является каналом модуля АЦП-ЦАП.
-6 – серверу данных не хватает памяти для организации буферов данных.
-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.
-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().
-9 – номер декады decade < 0.
-10 – номер декады decade > 4.
-11 – момент времени time больше текущего времени для данного канала.
-12 – момент времени time < 0.
-13 – указатель на массив данных data = NULL.
-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".
-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().
-16 - пустой указатель на буфер.
-20 - превышено время обработки функции.

© ООО "ЭТМС" 1992-2024. Все права защищены

***NumDSP*** – опрос номера сигнального процессора, физическим каналом которого является заданный канал.

*long **NumDSP** (long channel)*

Параметры

*channel* – номер канала, номер сигнального процессора которого требуется узнать. Параметр *channel* может принимать значение от *0* до (*QuanChan() – 1)*.

Возвращаемое значение

≥ 0 – номер сигнального процессора, физическим каналом которого является заданный канал. Если к компьютеру подключено несколько разнотипных модулей АЦП-ЦАП, то возвращаемое занчение принимает значение от 0 до (количество устройств одного типа – 1).

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа ZETServer.ocx выгружена из памяти процессора.

-3 – параметр channel < 0.

-4 – параметр channel > (QuanChan() – 1).

-5 – заданный канал не является каналом модуля АЦП-ЦАП.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

***NumModule*** – опрос номера модуля устройства (сигнального процессора) в многомодульной системе.

*long **NumModule** (long channel)*

Параметры

*channel* – номер канала, номер модуля сигнального процессора которого требуется узнать. Параметр *channel* может принимать значение от *0* до (*QuanChan() – 1)*.

Возвращаемое значение

© ООО "ЭТМС" 1992-2024. Все права защищены

≥ 0 – номер модуля сигнального процессора в многомодульной системе. Если к компьютеру подключено несколько разнотипных модулей АЦП-ЦАП, то возвращаемой значение принимает значение от 0 до (количество всех устройств – 1).

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа ZETServer.ocx выгружена из памяти процессора.

-3 – параметр channel < 0.

-4 – параметр channel > (QuanChan() – 1).

-5 – заданный канал не является каналом модуля АЦП-ЦАП.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

***NumPhChan*** – опрос физического номера канала на модуле для заданного канала.

*long* ***NumPhChan*** *(long channel)*

Параметры

*channel* – номер канала, физический номер которого требуется узнать. Параметр *channel* может принимать значение от *0* до (*QuanChan() – 1)*.

Возвращаемое значение

≥ 0 и < 100 – физический номер заданного канала на модуле АЦП.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа ZETServer.ocx выгружена из памяти процессора.

-3 – параметр channel < 0.

-4 – параметр channel > (QuanChan() – 1).

-5 – заданный канал не является каналом модуля АЦП-ЦАП.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

© ООО "ЭТМС" 1992-2024. Все права защищены

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

***GetAttenuation*** – опрос коэффициента затухания заданного канала ЦАП.

*float* ***GetAttenuation*** *(long channel)*

Параметры

*channel* – номер канала ЦАП, коэффициент затухания которого требуется узнать. Параметр *channel* может принимать значение от *0* до (*MaxQuanChanDac () – 1)*.

Возвращаемое значение

≥ 0 и ≤ 1 – коэффициент затухания заданного канала ЦАП.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа ZETServer.ocx выгружена из памяти процессора.

-3 – параметр channel < 0.

-4 – параметр channel > (QuanChan() – 1).

-5 – заданный канал не является каналом модуля АЦП-ЦАП.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

***Frequency*** – опрос частоты дискретизации АЦП по выбранному каналу.

*float* ***Frequency*** *(long Channel)*

Параметры

© ООО "ЭТМС" 1992-2024. Все права защищены

*Channel* – номер устройства (сигнального процессора) в системе, частоту дискретизации которого требуется узнать. Параметр *Channel* может принимать значение от *0* до (*QuanDSP() – 1),* и тогда возвращаемым значением будет частота дискретизации модуля АЦП устройства. Параметр *Channel* может принимать значение от *-1* до (*-QuanDSP()),* и тогда возвращаемым значением будет частота дискретизации модуля ЦАП устройства.

Возвращаемое значение

≥ 0 – частота дискретизации заданного устройства.

***FrequencyDac*** – опрос частоты дискретизации ЦАП по выбранному каналу.

*float* ***Frequency*** *(long channelDac)*

Параметры

*channelDac* – номер канала ЦАП, частоту дискретизации которого требуется узнать. При подключении нескольких разных модулей АЦП-ЦАП, частота дискретизации на каждом модуле может быть произвольной и не кратной друг другу. Параметр *channelDac* может принимать значение от *0* до (*MaxQuanChanDac () – 1)*.

Возвращаемое значение

> 0 – частота дискретизации заданного канала ЦАП.

-3 – параметр *channelDac < 0*.

20100 – на момент вызова метода не было подключения к серверу данных.

20200 – программа *ZETServer.ocx* выгружена из памяти процессора.

20300 – подключение к серверу было произведено методами *Connect()* или *ConnectVrtCh()* , которые не организуют каналов ЦАП.

20400 – серверу данных не хватает памяти для организации буферов данных.

20500 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

20600 – заданный канал ЦАП программой не задействован.

20700 – параметр *channelDac* > (*MaxQuanChanDac () – 1)*.

***GetFrequency*** – опрос частоты дискретизации заданного устройства.

*float* ***GetFrequency*** *(long Modul)*

Параметры

*Modul* – номер устройства (сигнального процессора) в системе, частоту дискретизации которого требуется узнать. Параметр *Modul* может принимать значение от *0* до (*QuanDSP() – 1),* и тогда возвращаемым значением будет частота дискретизации модуля АЦП устройства. Параметр *Modul* может принимать значение от *-1* до (*-QuanDSP()),* и тогда возвращаемым значением будет частота дискретизации модуля ЦАП устройства.

Возвращаемое значение

≥ 0 – частота дискретизации заданного устройства.

© ООО "ЭТМС" 1992-2024. Все права защищены

***GetNextFreq*** – опрос списка возможных частот дискретизации заданного устройства. При многократном обращении к этой функции, она возвращает возможные частоты дискретизации.

*float* ***GetNextFreq*** *(long Modul, long Par)*

Параметры

*Modul* – номер устройства (сигнального процессора) в системе, список частот дискретизации которого требуется узнать. Параметр *Modul* может принимать значение от *0* до (*QuanDSP() – 1),* и тогда возвращаемым значением будет частота дискретизации модуля АЦП устройства. Параметр *Modul* может принимать значение от *-1* до (*-QuanDSP()),* и тогда возвращаемым значением будет частота дискретизации модуля ЦАП устройства.

*Par* – параметр для получения списка частот дискретизации. При первом обращении этот параметр должен быть равен 0, при последующих обращениях этот параметр не должен быть равен 0.

Возвращаемое значение

0 – список возможных частот дискретизации устройства прочитан полностью.

≥ 0 – возможная частота дискретизации заданного устройства.

***GetAmplify*** – опрос коэффициента усиления по заданному каналу.

*float* ***GetAmplify*** *(long channel)*

Параметры

*channel* – номер канала, коэффициент усиления по которому требуется узнать. Параметр *channel* может принимать значение от *0* до (*QuanChan() – 1)*.

Возвращаемое значение

≥ 0 – коэффициент усиления по заданному каналу.

***IdChan -*** опрос идентификатора заданного канала АЦП. Этот метод полезно использовать в программах, в которых необходимо производить привязку канала сервера с физическим каналом АЦП.

*long* ***IdChan*** *(long channel, long* Id)*

Параметры

*channel* – номер канала, идентификатор которого требуется узнать. Параметр *lChannel* может принимать значение от *0* до (*QuanChan() – 1)*.

*Id* – возвращает 32-разрядный идентификатор. Для физического канала АЦП и ЦАП *Id* имеет следующую структуру: младший байт (0-7 бит) – номер физического канала на модуле АЦП, следующий байт (8-15 бит) – тип платы АЦП, старшее 16-разрядное слово (16-31 бит) – заводской номер платы АЦП для плат АЦП на шине USB

© ООО "ЭТМС" 1992-2024. Все права защищены

и LAN или порядковый номер контроллера расположенного на шине PCI. Для виртуального канала *Id* имеет следующую структуру: младший байт (0-7 бит) – номер виртуального канала в программе, следующий байт (8-15 бит) – порядковый номер запущенной программы. Например, если запущено две одинаковые программы, то у первой программы номер 1, у второй программы номер 2, старшее 16-разрядное слово (16-31 бит) – идентификатор программы. Каждая программа имеет свой идентификатор.

Возвращаемое значение

1 - данные содержат не менее одного значения NaN

0 – нормальное выполнение функции.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа *ZETServer.ocx* выгружена из памяти процессора.

-3 – параметр *channel < 0*.

-4 – параметр *channel* > (*QuanChan() – 1)*.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных *Size ≤ 0* или *size ≥ DecadeBufferSize().*

-9 – номер декады *decade < 0.*

-10 – номер декады *decade > 4.*

-11 – момент времени *time* больше текущего времени для данного канала.

-12 – момент времени *time < 0.*

-13 – указатель на массив данных *data = NULL.*

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

***IdChanDac*** – опрос идентификатора заданного канала ЦАП. Этот метод полезно использовать в программах, в которых необходимо производить привязку канала сервера с физическим каналом ЦАП.

*long **IdChanDac** (long channelDac, long* Id)*

Параметры

*channelDac* – номер канала, идентификатор которого требуется узнать. Параметр *channelDac* может принимать значение от *0* до (*MaxQuanChanDac() – 1)*.

*Id* – возвращает 32-разрядный идентификатор. Для физического канала ЦАП имеет следующую структуру: младший байт (0-7 бит) – номер физического канала на модуле ЦАП, следующий байт (8-15 бит) – тип платы ЦАП, старшее 16-разрядное слово (16-31 бит) – заводской номер платы ЦАП для плат ЦАП на шине USB и LAN или порядковый номер контроллера расположенного на шине PCI.

Возвращаемое значение

≥ 0 – номер канала ЦАП среди всех каналов сервера данных.

© ООО "ЭТМС" 1992-2024. Все права защищены

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа *ZETServer.ocx* выгружена из памяти процессора.

-3 – параметр *channel < 0*.

-4 – параметр *channel* > (*QuanChan() – 1)*.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных *Size ≤ 0* или *size ≥ DecadeBufferSize().*

-9 – номер декады *decade < 0.*

-10 – номер декады *decade > 4.*

-11 – момент времени *time* больше текущего времени для данного канала.

-12 – момент времени *time < 0*.

-13 – указатель на массив данных *data = NULL.*

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

В программе, которая работает с каналом ЦАП, происходит соединение с сервером по команде *ConnectDac()* . Для генерации сигнала по каналу ЦАП программа использует команду *PutData()*. Этот сигнал отображается в виде виртуального канала. Сигнал по этому каналу можно посмотреть по команде *GetData()*. Идентификатор канала генерации определяется командой *IdChanDac()* и он должен совпадать с одним из идентификаторов каналов для чтения (перебор делается командой *IdChan()*). Тот канал, который совпадает и есть виртуальный канал генератора.

***IdChanVirt*** – опрос идентификатора заданного виртуального канала. Этот метод полезно использовать в программах, в которых необходимо производить привязку канала сервера с виртуальным каналом.

*long* ***IdChanVirt*** *(long channel, long* ID)*

Параметры

*channel* – номер виртуального канала, идентификатор которого требуется узнать. Параметр *channel* может принимать значение от *0* до (количество виртуальных каналов, установленных функцией *ConnectVrtCh()* минус 1).

*Id* – возвращает 32-разрядный идентификатор. Для виртуального канала *Id* имеет следующую структуру: младший байт (0-7 бит) – номер виртуального канала в программе, следующий байт (8-15 бит) – порядковый номер запущенной программы. Например, если запущено две одинаковые программы, то у первой программы номер 1, у второй программы номер 2, старшее 16-разрядное слово (16-31 бит) – идентификатор программы. Каждая программа имеет свой идентификатор.

Возвращаемое значение

© ООО "ЭТМС" 1992-2024. Все права защищены

0 – нормальное выполнение функции.
-1 – на момент вызова метода не было подключения к серверу данных.
-2 – программа *ZETServer.ocx* выгружена из памяти процессора.
-3 – параметр *channel* < *0*.
-4 – параметр *channel* > (*ConnectVrtCh() – 1)*.
-5 – нет организованного программой виртуального канала с номером *channel*
-6 – нет организованного программой виртуального канала с номером *channel*
-7 – нет организованного программой виртуального канала с номером *channel*

В программе, которая создает виртуальный канал, происходит соединение с сервером по команде *ConnectVrt()*. Для генерации сигнала по виртуальному каналу программа использует команду *PutData()*. Этот сигнал отображается в виде виртуального канала. Сигнал по этому каналу можно посмотреть по команде *GetData()*. Идентификатор канала генерации определяется командой *IdChanVirt()* и он должен совпадать с одним из идентификаторов каналов для чтения (перебор делается командой *IdChan(...)*). Тот канал, который совпадает и есть виртуальный канал генератора.

## Управление каналами АЦП

***SetFrequency*** – установка частоты дискретизации заданного устройства АЦП.

*float* ***SetFrequency*** *(long Modul, float Freq)*

Параметры

*Modul* – номер устройства (сигнального процессора) в системе, частоту дискретизации которого требуется установить. Параметр *Modul* может принимать значение от *0* до (*QuanDSP() – 1),* и тогда будет установлена частота дискретизации модуля АЦП устройства. Параметр *Modul* может принимать значение от *-1* до (*-QuanDSP()),* и тогда будет установлена частота дискретизации модуля АЦП устройства.

*Freq* – частота дискретизации, которую требуется установить.

Возвращаемое значение

частота дискретизации устройства, которая установилась.

**Пример 1.** Для установления частоты дискретизации на приборе АЦП SetFrequency (0,50000) Сделано на примере одного прибора Анализатора ZET017U8.
**Пример 2.** Программа находит максимальную частоту дискретизации АЦП прибора, а затем устанавлиает ее.

```
max_ADC = GetNextFreq(0, 0);
SetFrequency(0, max_ADC);
```

© ООО "ЭТМС" 1992-2024. Все права защищены

***OnOffChannel*** – установка состояния канала АЦП заданного устройства.

*long* ***OnOffChannel*** *(long Modul, long Channel, long Par)*

Параметры

*Modul* – номер устройства (сигнального процессора) в системе, состояние канала которого требуется установить. Параметр *Modul* может принимать значение от *0* до (*QuanDSP() – 1).*

*Channel* – номер канала устройства, состояние которого требуется установить. Параметр Channel – номер канала устройства, состояние которого требуется установить. Параметр Channel может принимать значение от 0 до (QuanPhChan () – 1). Важно! Выключать все каналы нельзя. Хотя бы один канал должен всегда оставаться включённым.

*Par* – состояние канала, которое требуется установить. Если параметр *Par* равен 0, то канал выключается, а если 1 – то включается.

Возвращаемое значение

0 – канал АЦП отключился.

1 – канал АЦП включился.

***SetSinDiffChannel*** – установка режима работы канала АЦП заданного устройства.

*long* ***SetSinDiffChannel*** *(long Modul, long Channel, long Par)*

Параметры

*Modul* – номер устройства (сигнального процессора) в системе, режим работы канала которого требуется установить. Параметр *Modul* может принимать значение от *0* до (*QuanDSP() – 1).*

*Channel* – номер канала устройства, режим работы которого требуется установить. Параметр *Channel* может принимать значение от *0* до (*QuanPhChan () – 1).*

*Par* – режим работы канала, который требуется установить. Если параметр *Par* равен 0, то канал переключается в синфазный режим работы, а если 1 – то в дифференциальный режим работы.

Возвращаемое значение

0 – канал АЦП переключился в синфазный режим работы

1 – канал АЦП переключился в дифференциальный режим работы.

***IdChanGUID*** - опрос расширенного идентификатора заданного канала АЦП. Этот метод полезно использовать в программах, в которых необходимо производить привязку канала сервера с физическим каналом АЦП.

*long* ***IdChanGUID*** *(long channel_, long* id_)*

Параметры

© ООО "ЭТМС" 1992-2024. Все права защищены

*virtChannel_* – номер канала, идентификатор которого требуется узнать. Параметр *virtChannel_* может принимать значение от *0* до (*QuanChan() – 1)*.

*long* id_* – возвращает 32-разрядный идентификатор. Для физического канала АЦП и ЦАП *Id* имеет следующую структуру: младший байт (0-7 бит) – номер физического канала на модуле АЦП, следующий байт (8-15 бит) – тип платы АЦП, старшее 16-разрядное слово (16-31 бит) – заводской номер платы АЦП для плат АЦП на шине USB и LAN или порядковый номер контроллера расположенного на шине PCI. Для виртуального канала *Id* имеет следующую структуру: младший байт (0-7 бит) – номер виртуального канала в программе, следующий байт (8-15 бит) – порядковый номер запущенной программы. Например, если запущено две одинаковые программы, то у первой программы номер 1, у второй программы номер 2, старшее 16-разрядное слово (16-31 бит) – идентификатор программы. Каждая программа имеет свой идентификатор.

Возвращаемое значение

1 - данные содержат не менее одного значения NaN

0 – нормальное выполнение функции.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа *ZETServer.ocx* выгружена из памяти процессора.

-3 – параметр *channel < 0*.

-4 – параметр *channel* > (*QuanChan() – 1)*.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных *Size ≤ 0* или *size ≥ DecadeBufferSize().*

-9 – номер декады *decade < 0.*

-10 – номер декады *decade > 4.*

-11 – момент времени *time* больше текущего времени для данного канала.

-12 – момент времени *time < 0*.

-13 – указатель на массив данных *data = NULL.*

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

***GetTimeFromStartAdc****(void)* - опрос времени с момента старта АЦП

*double* ***GetTimeFromStartAdc****(void)*

***Comment*** - получение комментария по каналу АЦП.

BSTR ***Comment****(long channel)*

© ООО "ЭТМС" 1992-2024. Все права защищены

Параметры

*channel* – номер канала, принадлежащий тому модулю АЦП-ЦАП, количество работающих каналов которого требуется узнать. Параметр *channel* может принимать значение от *0* до (*QuanChan() – 1)*.

Возвращаемое значение

≥ 0 – суммарное количество работающих каналов на модуле АЦП-ЦАП, которому принадлежит заданный канал.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа ZETServer.ocx выгружена из памяти процессора.

-3 – параметр channel < 0.

-4 – параметр channel > (QuanChan() – 1).

-5 – заданный канал не является каналом модуля АЦП-ЦАП.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

***Coordinate*** - *получение значения координат.*

*long* ***Coordinate****(long channel, double* X, double* Y, double* Z, long* P)*

Параметры

*channel* – номер канала, принадлежащий тому модулю АЦП-ЦАП, количество работающих каналов которого требуется узнать. Параметр *channel* может принимать значение от *0* до (*QuanChan() – 1)*.

Возвращаемое значение

≥ 0 – суммарное количество работающих каналов на модуле АЦП-ЦАП, которому принадлежит заданный канал.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа ZETServer.ocx выгружена из памяти процессора.

-3 – параметр channel < 0.

-4 – параметр channel > (QuanChan() – 1).

© ООО "ЭТМС" 1992-2024. Все права защищены

-5 – заданный канал не является каналом модуля АЦП-ЦАП.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

***SetCoordinate*** - *установка значения координат.*

*long* ***SetCoordinate****(long VrtChannel, double X, double Y, double Z, long P)*

Параметры

*VrtChannel* – номер канала, принадлежащий тому модулю АЦП-ЦАП, количество работающих каналов которого требуется узнать. Параметр *channel* может принимать значение от *0* до (*QuanChan() – 1)*.

Возвращаемое значение

≥ 0 – суммарное количество работающих каналов на модуле АЦП-ЦАП, которому принадлежит заданный канал.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа ZETServer.ocx выгружена из памяти процессора.

-3 – параметр channel < 0.

-4 – параметр channel > (QuanChan() – 1).

-5 – заданный канал не является каналом модуля АЦП-ЦАП.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

© ООО "ЭТМС" 1992-2024. Все права защищены

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

## Управление каналами ЦАП

***SetFrequency*** – установка частоты дискретизации заданного устройства АЦП и ЦАП.

*float* ***SetFrequency*** *(long Modul, float Freq)*

Параметры

*Modul* – номер устройства (сигнального процессора) в системе, частоту дискретизации которого требуется установить. Параметр *Modul* может принимать значение от *0* до (*QuanDSP() – 1),* и тогда будет установлена частота дискретизации модуля ЦАП устройства. Параметр *Modul* может принимать значение от *-1* до (*-QuanDSP()),* и тогда будет установлена частота дискретизации модуля ЦАП устройства.

*Freq* – частота дискретизации, которую требуется установить.

Возвращаемое значение

частота дискретизации устройства, которая установилась.

**Пример 1.** Для установления частоты дискретизации на приборе ЦАП SetFrequency (-1,200000). Сделано на примере одного прибора Анализатора ZET017U8.

**Пример 2.** Программа находит максимальную частоту дискретизации ЦАП прибора, а затем устанавлиает ее.

```
max_DAC = GetNextFreq(-1, 0);
SetFrequency(-1, max_DAC);
```

***IsFreeChanDac*** – определение свободного (незанятого другой программой) канала ЦАП.

*long* ***IsFreeChanDac*** *(long Channel)*

Параметры

*Channel* – номер канала ЦАП, состояние которого требуется узнать. Параметр *Channel* может принимать значение от *0* до (*MaxQuanChanDac () – 1)*.

Возвращаемое значение

0 – заданный канал ЦАП свободен (не занят другой программой)

1 – заданный канал ЦАП занят.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа ZETServer.ocx выгружена из памяти процессора.

-3 – параметр channel < 0.

-4 – параметр channel > (QuanChan() – 1).

© ООО "ЭТМС" 1992-2024. Все права защищены

-5 – заданный канал не является каналом модуля АЦП-ЦАП.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

***SetChanDac*** – подключение заданного канала ЦАП.

*long* ***SetChanDac*** *(long Channel)*

Параметры

*Channel* – номер канала ЦАП, который требуется подключить. Параметр *Channel* может принимать значение от *0* до (*MaxQuanChanDac () – 1)*.

Возвращаемое значение

-20 – невозможно подключить заданный канал ЦАП.

1 - данные содержат не менее одного значения NaN

0 – нормальное выполнение функции.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа ZETServer.ocx выгружена из памяти процессора.

-3 – параметр channel < 0.

-4 – параметр channel > (QuanChan() – 1).

-5 – заданный канал не является каналом модуля АЦП-ЦАП.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

© ООО "ЭТМС" 1992-2024. Все права защищены

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

***RemChanDac*** – отключение заданного канала ЦАП.

*long* ***RemChanDac*** *(long Channel)*

Параметры

*Channel* – номер канала ЦАП, который требуется отключить. Параметр *Channel* может принимать значение от *0* до (*MaxQuanChanDac () – 1)*.

Возвращаемое значение

-20 – невозможно подключить заданный канал ЦАП.

1 - данные содержат не менее одного значения NaN

0 – нормальное выполнение функции.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа ZETServer.ocx выгружена из памяти процессора.

-3 – параметр channel < 0.

-4 – параметр channel > (QuanChan() – 1).

-5 – заданный канал не является каналом модуля АЦП-ЦАП.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

***SetAttenuation*** – установка коэффициента затухания заданного канала ЦАП.

*long* ***SetAttenuation*** *(long channel, float attenuator)*

Параметры

*channel* – номер канала ЦАП, коэффициент затухания которого следует установить. Параметр *channel* может принимать значение от *0* до (*MaxQuanChanDac () – 1)*.

© ООО "ЭТМС" 1992-2024. Все права защищены

*attenuator* – коэффициент затухания, который следует установить. Параметр *attenuator* может принимать значения от 0 до 1.

Возвращаемое значение

1 - данные содержат не менее одного значения NaN

0 – нормальное выполнение функции.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа ZETServer.ocx выгружена из памяти процессора.

-3 – параметр channel < 0.

-4 – параметр channel > (QuanChan() – 1).

-5 – заданный канал не является каналом модуля АЦП-ЦАП.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

## Управление виртуальными каналами

***VrtChnInfo*** – задание информации для виртуального канала.

*long **VrtChnInfo** (long virtchannel, BSTR* comment, BSTR* conversion, float freq, float MaxLevel, float MinLevel, float refer)*

Параметры

*virtchannel* – номер виртуального канала, информацию по которому требуется задать. Параметр *virtchannel* может принимать значение от *0* до (количество виртуальных каналов, установленных функцией *ConnectVrtCh (...)* минус 1).

*comment* – строка с названием виртуального канала.

*conversion* – строка с названием единицей измерения по виртуальному каналу.

*freq* – частота дискретизации по виртуальному каналу.

*MaxLevel* – максимально допустимый уровень по виртуальному каналу.

*MinLevel* – минимально различимый уровень по виртуальному каналу.

*refer* – значение опорного уровня для расчета уровней сигнала в децибелах.

Возвращаемое значение

© ООО "ЭТМС" 1992-2024. Все права защищены

1 - данные содержат не менее одного значения NaN

0 – нормальное выполнение функции.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа ZETServer.ocx выгружена из памяти процессора.

-3 – параметр channel < 0.

-4 – параметр channel > (QuanChan() – 1).

-5 – заданный канал не является каналом модуля АЦП-ЦАП.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – нет организованного программой виртуального канала с номером *virtchannel.*

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

***SetGroupName*** – устанавливает определение группового имени виртуального канала

*long* ***SetGroupName*** *(LONG VirtChannel, BSTR GrName)*

Параметры

*virtchannel* – номер виртуального канала, информацию по которому требуется задать. Параметр *virtchannel* может принимать значение от *0* до (количество виртуальных каналов, установленных функцией *ConnectVrtCh (...)* минус 1).

*BSTR GrName* - название группы

Возвращаемое значение

непустая строка – групповое имя канала.

пустая строка – на момент вызова метода не было подключения к серверу данных, программа *ZETServer.ocx* выгружена из памяти процессора.

***SetStatus*** – опрос статуса виртуального канала

*long* ***SetStatus*** *(LONG lVirtChannel, LONG lStatus)*

*lVirtChannel* – номер виртуального канала, информацию по которому требуется задать. Параметр *lVirtChannel* может принимать значение от *0* до (количество виртуальных каналов, установленных функцией *ConnectVrtCh (...)* минус 1).

© ООО "ЭТМС" 1992-2024. Все права защищены

*lStatus* - определяет существование виртуального канала.

Возвращаемое значение

1 - данные содержат не менее одного значения NaN

0 – нормальное выполнение функции.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа ZETServer.ocx выгружена из памяти процессора.

-3 – параметр channel < 0.

-4 – параметр channel > (QuanChan() – 1).

-5 – заданный канал не является каналом модуля АЦП-ЦАП.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – нет организованного программой виртуального канала с номером *virtchannel.*

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

***SetID*** - установка идентификатора виртуального канала

*long* ***SetID*** *(LONG lVirtChannel, LONG lID)*

Параметры

*lVirtChannel* – номер виртуального канала, информацию по которому требуется задать. Параметр *lVirtChannel* может принимать значение от *0* до (количество виртуальных каналов, установленных функцией *ConnectVrtCh (...)* минус 1).

*LONG lID* - идентификатор виртуального канала

Возвращаемое значение

1 - данные содержат не менее одного значения NaN

0 – нормальное выполнение функции.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа ZETServer.ocx выгружена из памяти процессора.

-3 – параметр channel < 0.

-4 – параметр channel > (QuanChan() – 1).

-5 – заданный канал не является каналом модуля АЦП-ЦАП.

-6 – серверу данных не хватает памяти для организации буферов данных.

© ООО "ЭТМС" 1992-2024. Все права защищены

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

***SetID*** (в настоящее время не реализована) - установка расширенного идентификатора виртуального канала.

*long **SetID** (LONG virtChannel_, CString id_)*

Параметры

*virtChannel_* – номер виртуального канала, информацию по которому требуется задать. Параметр *virtChannel_* может принимать значение от *0* до (количество виртуальных каналов, установленных функцией *ConnectVrtCh (...)* минус 1).

*CString id_* - идентификатор виртуального канала

Возвращаемое значение

1 - данные содержат не менее одного значения NaN

0 – нормальное выполнение функции.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа ZETServer.ocx выгружена из памяти процессора.

-3 – параметр channel < 0.

-4 – параметр channel > (QuanChan() – 1).

-5 – заданный канал не является каналом модуля АЦП-ЦАП.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

© ООО "ЭТМС" 1992-2024. Все права защищены

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

## Работа с памятью внутри сервера данных. Сделано так, чтобы каналы сохраняли за собой место в памяти при различных изменениях в сервере, используется в AddVirtualChannel, DeleteVirtualChannel

***AddVirtualChannel*** - добавление виртуальных каналов

*long* ***AddVirtualChannel*** *(LONG lIndex, LONG lQuantity)*

Параметры

*LONG lIndex* - индекс указателя виртуального канала.

*LONG lQuantity* - количество виртуальных каналов.

***DeleteVirtualChannel*** - удаление виртуальных каналов

*long* ***DeleteVirtualChannel*** *(LONG lIndex, LONG lQuantity)*

Параметры

*LONG lIndex* - индекс указателя виртуального канала.

*LONG lQuantity* - количество виртуальных каналов.

***IdChanVirt*** (в настоящее время не реализована) – опрос расширенного идентификатора заданного виртуального канала. Этот метод полезно использовать в программах, в которых необходимо производить привязку канала сервера с виртуальным каналом.

*long* ***IdChanVirt*** *(long virtChannel_, GUID* id_)*

Параметры

*virtChannel_* – номер виртуального канала, идентификатор которого требуется узнать. Параметр *virtChannel_* может принимать значение от *0* до (количество виртуальных каналов, установленных функцией *ConnectVrtCh()* минус 1).

*GUID* id_* – возвращает 32-разрядный идентификатор. Для виртуального канала *GUID* id_* имеет следующую структуру: младший байт (0-7 бит) – номер виртуального канала в программе, следующий байт (8-15 бит) – порядковый номер запущенной программы. Например, если запущено две одинаковые программы, то у первой программы номер 1, у второй программы номер 2, старшее 16-разрядное слово (16-31 бит) – идентификатор программы. Каждая программа имеет свой идентификатор.

© ООО "ЭТМС" 1992-2024. Все права защищены

Возвращаемое значение

0 – нормальное выполнение функции.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа *ZETServer.ocx* выгружена из памяти процессора.

-3 – параметр *virtChannel_ < 0*.

-4 – параметр *virtChannel_* > (*QuanChan() – 1)*.

-5 – нет организованного программой виртуального канала с номером *virtChannel_*

-6 – нет организованного программой виртуального канала с номером *virtChannel_*

-7 – нет организованного программой виртуального канала с номером *virtChannel_*

В программе, которая создает виртуальный канал, происходит соединение с сервером по команде *ConnectVrt()*. Для генерации сигнала по виртуальному каналу программа использует команду *PutData()*. Этот сигнал отображается в виде виртуального канала. Сигнал по этому каналу можно посмотреть по команде *GetData()*. Идентификатор канала генерации определяется командой *IdChanVirt()* и он должен совпадать с одним из идентификаторов каналов для чтения (перебор делается командой *IdChan(...)*). Тот канал, который совпадает и есть виртуальный канал генератора.

## Добавление трех функций для работы с выборками сигналов *PutPulse, GetInfLast, GetInfIndex*

***PutPulse*** - создание виртуального канала.

*long* ***PutPulse*** *(long channel, long index, double * time, long * size, float * data)*

*channel* – номер виртуального канала, информацию по которому требуется задать. Параметр *channel* может принимать значение от *0* до (количество виртуальных каналов, установленных функцией *ConnectVrtCh (...)* минус 1).

Возвращаемое значение

непустая строка – групповое имя канала.

пустая строка – на момент вызова метода не было подключения к серверу данных, программа *ZETServer.ocx* выгружена из памяти процессора.

*time* – момент времени, в который необходимо взять данные для обработки.

*size* – количество отсчетов данных для передачи. Параметр *size* может принимать значение от 1 до *DecadeBufferSize()*. *DecadeBufferSize()* – метод для определения размера буфера по декаде;

*data* – указатель на массив данных пользователя, массив данных в плавающей запятой одинарной точности.

Возвращаемое значение

1 - данные содержат не менее одного значения NaN

0 – нормальное выполнение функции.

© ООО "ЭТМС" 1992-2024. Все права защищены

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа ZETServer.ocx выгружена из памяти процессора.

-3 – параметр channel < 0.

-4 – параметр channel > (QuanChan() – 1).

-5 – заданный канал не является каналом модуля АЦП-ЦАП.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

***GetPulse*** - опрос виртуальных каналов.

*long* ***GetPulse*** *(long channel, long index, double * time, long * size, float * data)*

*channel* – номер виртуального канала, информацию по которому требуется задать. Параметр *channel* может принимать значение от *0* до (количество виртуальных каналов, установленных функцией *ConnectVrtCh (...)* минус 1).

Возвращаемое значение

непустая строка – групповое имя канала.

пустая строка – на момент вызова метода не было подключения к серверу данных, программа *ZETServer.ocx* выгружена из памяти процессора.

*time* – момент времени, в который необходимо взять данные для обработки.

*size* – количество отсчетов данных для передачи. Параметр *size* может принимать значение от 1 до *DecadeBufferSize()*. *DecadeBufferSize()* – метод для определения размера буфера по декаде;

*data* – указатель на массив данных пользователя, массив данных в плавающей запятой одинарной точности.

Возвращаемое значение

1 - данные содержат не менее одного значения NaN

0 – нормальное выполнение функции.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа ZETServer.ocx выгружена из памяти процессора.

-3 – параметр channel < 0.

-4 – параметр channel > (QuanChan() – 1).

© ООО "ЭТМС" 1992-2024. Все права защищены

-5 – заданный канал не является каналом модуля АЦП-ЦАП.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

***GetInfLast*** - опрос последнего индекса виртуального канала.

*long* ***GetInfLast*** *(long channel, double * time, long * size, long * index)*

Параметры

*channel* – номер канала, данные по которому требуется запросить. Параметр *channel* может принимать значение от *0* до (количество виртуальных каналов, установленных функцией *ConnectVrtCh (...)* минус 1).

*time* – момент времени, в который необходимо взять данные для обработки.

*size* – количество отсчетов данных для передачи. Параметр *size* может принимать значение от 1 до *BufferSize()*. *BufferSize()* – метод для определения размера.

*index* - индекс указателя канала.

Возвращаемое значение

1 - данные содержат не менее одного значения NaN

0 – нормальное выполнение функции.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа ZETServer.ocx выгружена из памяти процессора.

-3 – параметр channel < 0.

-4 – параметр channel > (QuanChan() – 1).

-5 – заданный канал не является каналом модуля АЦП-ЦАП.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

© ООО "ЭТМС" 1992-2024. Все права защищены

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

***GetInfIndex*** - опрос индекса виртуального канала.

*long* ***GetInfIndex*** *(long channel, double * time, long * size, long index)*

Параметры

*channel* – номер канала, данные по которому требуется запросить. Параметр *channel* может принимать значение от *0* до (количество виртуальных каналов, установленных функцией *ConnectVrtCh (...)* минус 1).

*time* – момент времени, в который необходимо взять данные для обработки.

*size* – количество отсчетов данных для передачи. Параметр *size* может принимать значение от 1 до *BufferSize()*. *BufferSize()* – метод для определения размера.

*index* - индекс указателя канала.

Возвращаемое значение

1 - данные содержат не менее одного значения NaN

0 – нормальное выполнение функции.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа ZETServer.ocx выгружена из памяти процессора.

-3 – параметр channel < 0.

-4 – параметр channel > (QuanChan() – 1).

-5 – заданный канал не является каналом модуля АЦП-ЦАП.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

© ООО "ЭТМС" 1992-2024. Все права защищены

***QuanVirtChan*** - количество виртуальных каналов на модуле.

*long* ***QuanVirtChan****(LONG numDSP)*

Параметры

*numDSP* – номер устройства (сигнального процессора), количество физических каналов которого требуется узнать. Параметр *numDSP* может принимать значение от *0* до (*QuanDSP() – 1)*.

Возвращаемое значение

≥ 0 – количество физических каналов устройства.

-1 – параметр *numDSP* < *0*.

-2 – параметр *numDSP* > (максимальное количество устройств в системе – *1)*.

-3 – параметр *numDSP* > (*QuanDSP() – 1)*.

## Работа с цифровым портом

Для обмена данными с цифровыми портом ввода-вывода, установленного на платах АЦП-ЦАП, необходимо выбирать номер модуля. Если в системе установлена только одна плата, то номер должен быть равен 0. Если в системе установлены несколько плат, то номера должны меняться от 0 до (количество установленных плат – 1). Порядок следования модулей в системе можно определить по порядку следования устройств в программе «Диспетчер устройств». Для инициализации цифрового вывода необходимо установить битовую маску. Если бит равен 0, то этот бит становится запрещенным для вывода данных. Если бит установлен в 1, то можно выводить данные по этому биту. Все цифровые биты доступны по чтению. Инициализировать цифровой ввод достаточно один раз при запуске программы.

***SetDigOutEnable*** – установка маски вывода на выбранному устройстве.

*long* ***SetDigOutEnable****(long Zmodule, long OutMask)*

Параметры

*Zmodule* – номер устройства (сигнального процессора) в системе, маску вывода цифрового порта которого следует установить. Параметр *Zmodule* может принимать значение от *0* до (*QuanDSP() – 1)*.

*OutMask* – бинарная комбинация (маска) выходных битов цифрового порта устройства, которую следует установить. Например, если маска равна 3 (3h, 11b), то первый и второй биты цифрового порта будут работать в режиме выхода, а остальные в режиме входа.

Возвращаемое значение

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа *ZETServer.ocx* выгружена из памяти процессора.

© ООО "ЭТМС" 1992-2024. Все права защищены

-3 – параметр *Zmodule* < *0*.

-4 – параметр *Zmodule* > (количество устройств в системе – *1)*.

-5 – невозможно подключиться к устройству.

-6 – невозможно установить маску вывода для цифрового порта устройства.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

***GetDigOutEnable*** – опрос маски вывода на выбранном устройстве.

*long* ***GetDigOutEnable****(long Zmodule)*

Параметры

*Zmodule* – номер устройства (сигнального процессора) в системе, маску вывода цифрового порта которого требуется узнать. Параметр *Zmodule* может принимать значение от *0* до (*QuanDSP() – 1)*.

Возвращаемое значение

≥ 0 – бинарная комбинация (маска) выходных битов цифрового порта устройства. Например, если маска равна 3 (3h, 11b), то первый и второй биты цифрового порта работают в режиме выхода, а остальные в режиме входа.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа *ZETServer.ocx* выгружена из памяти процессора.

-3 – параметр *Zmodule* < *0*.

-4 – параметр *Zmodule* > (количество устройств в системе – *1)*.

-5 – невозможно подключиться к устройству.

-6 – невозможно установить маску вывода для цифрового порта устройства.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

© ООО "ЭТМС" 1992-2024. Все права защищены

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

***GetDigInput*** – опрос входных данных цифрового порта на выбранном устройстве.

*long* ***GetDigInput****(long Zmodule)*

Параметры

*Zmodule* – номер устройства (сигнального процессора) в системе, входные данные цифрового порта которого требуется узнать. Параметр *Zmodule* может принимать значение от *0* до (*QuanDSP() – 1)*.

Возвращаемое значение

≥ 0 – бинарная комбинация (маска) входных данных цифрового порта устройства. Например, если входные данные равны 3 (3h, 11b), то на первом и втором битах цифрового порта на входе логическая единица, а на остальных логический ноль.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа *ZETServer.ocx* выгружена из памяти процессора.

-3 – параметр *Zmodule* < *0*.

-4 – параметр *Zmodule* > (количество устройств в системе – *1)*.

-5 – невозможно подключиться к устройству.

-6 – невозможно установить маску вывода для цифрового порта устройства.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

***GetDigOutput*** – опрос выходных данных цифрового порта на выбранном устройстве.

© ООО "ЭТМС" 1992-2024. Все права защищены

*long* ***GetDigOutput****(long Zmodule)*

Параметры

*Zmodule* – номер устройства (сигнального процессора) в системе, выходные данные цифрового порта которого требуется узнать. Параметр *Zmodule* может принимать значение от *0* до (*QuanDSP() – 1)*.

Возвращаемое значение

≥ 0 – бинарная комбинация (маска) выходных данных цифрового порта устройства. Например, если выходные данные равны 3 (3h, 11b), то на первом и втором битах цифрового порта на выходе логическая единица, а на остальных логический ноль.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа *ZETServer.ocx* выгружена из памяти процессора.

-3 – параметр *Zmodule* < *0*.

-4 – параметр *Zmodule* > (количество устройств в системе – *1)*.

-5 – невозможно подключиться к устройству.

-6 – невозможно установить маску вывода для цифрового порта устройства.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

***SetDigOutput*** – установка выходных данных цифрового порта на выбранном устройстве.

*long* ***SetDigOutput****(long Zmodule, long DigOut)*

Параметры

*Zmodule* – номер устройства (сигнального процессора) в системе, выходные данные цифрового порта которого следует установить. Параметр *lModule* может принимать значение от *0* до (*QuanDSP() – 1)*.

*DigOut* – бинарная комбинация (маска) выходных данных цифрового порта устройства, которую следует установить. Например, если выходные данные равны 3 (3h, 11b), то у первого и второго бита цифрового порта на выходе будет логическая единица, а у остальных логический ноль.

© ООО "ЭТМС" 1992-2024. Все права защищены

Возвращаемое значение

≥ 0 – бинарная комбинация (маска) выходных данных цифрового порта устройства. Например, если выходные данные равны 3 (3h, 11b), то на первом и втором битах цифрового порта на выходе логическая единица, а на остальных логический ноль.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа *ZETServer.ocx* выгружена из памяти процессора.

-3 – параметр *Zmodule < 0*.

-4 – параметр *Zmodule* > (количество устройств в системе – *1)*.

-5 – невозможно подключиться к устройству.

-6 – невозможно установить маску вывода для цифрового порта устройства.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

***GetDigBits*** – опрос количества бит цифрового порта на выбранном устройстве.

*long* ***GetDigBits*** *(long Modul)*

Параметры

*Modul* – номер устройства (сигнального процессора) в системе, количество бит цифрового порта которого требуется узнать. Параметр *Modul* может принимать значение от *0* до (*QuanDSP() – 1)*.

Возвращаемое значение

≥ 0 – количество бит цифрового порта устройства.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа *ZETServer.ocx* выгружена из памяти процессора.

-3 – параметр *Zmodule < 0*.

-4 – параметр *Zmodule* > (количество устройств в системе – *1)*.

-5 – невозможно подключиться к устройству.

-6 – невозможно установить маску вывода для цифрового порта устройства.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

© ООО "ЭТМС" 1992-2024. Все права защищены

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

***SetBitDigOutput*** – установка состояния выходного бита на заданном устройстве в логическую единицу.

*long* ***SetBitDigOutput****(long Zmodule, long bit)*

Параметры

*Zmodule* – номер устройства (сигнального процессора) в системе, состояние выходного бита которого следует установить в логическую единицу. Параметр *Zmodule* может принимать значение от *0* до (*QuanDSP() – 1)*.

*bit* – номер бита, состояние которого следует установить в логическую единицу. Параметр *Zmodule* может принимать значение от *0* до (*GetDigBits() – 1)*.

Возвращаемое значение

0 – нормальное выполнение функции.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа ZETServer.ocx выгружена из памяти процессора.

-3 – параметр *Zmodule* < *0*.

-4 – параметр *Zmodule* > (количество устройств в системе – *1)*.

-5 – невозможно подключиться к устройству.

-6 – невозможно установить состояние выходного бита в логическую единицу для цифрового порта заданного устройства.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

© ООО "ЭТМС" 1992-2024. Все права защищены

***ClrBitDigOutput*** – установка состояния выходного бита на заданном устройстве в логический ноль.

*long* ***ClrBitDigOutput*** *(long Zmodule, long bit)*

Параметры

*Zmodule* – номер устройства (сигнального процессора) в системе, состояние выходного бита которого следует установить в логический ноль. Параметр *Zmodule* может принимать значение от *0* до (*QuanDSP() – 1)*.

*bit* – номер бита, состояние которого следует установить в логический ноль. Параметр *Zmodule* может принимать значение от *0* до (*GetDigBits() – 1)*.

Возвращаемое значение

0 – нормальное выполнение функции.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа ZETServer.ocx выгружена из памяти процессора.

-3 – параметр *Zmodule* < *0*.

-4 – параметр *Zmodule* > (количество устройств в системе – *1)*.

-5 – невозможно подключиться к устройству.

-6 – невозможно установить состояние выходного бита в логическую единицу для цифрового порта заданного устройства.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

***Triger*** - настраивает синхронизацию по цифровому порту

*long* ***Triger*** *(LONG Modul, LONG Parametr, LONG* ValInt, FLOAT* ValFloat)-*

Параметры

*Modul* – номер устройства (сигнального процессора) в системе, список частот дискретизации которого требуется узнать. Параметр *Modul* может принимать значение от *0* до (*QuanDSP() – 1),* и тогда возвращаемым значением будет частота дискретизации

© ООО "ЭТМС" 1992-2024. Все права защищены

модуля АЦП устройства. Параметр *Modul* может принимать значение от *-1* до (-*QuanDSP()),* и тогда возвращаемым значением будет частота дискретизации модуля ЦАП устройства.

*Parametr* – параметр для получения списка частот дискретизации. При первом обращении этот параметр должен быть равен 0, при последующих обращениях этот параметр не должен быть равен 0.

Возвращаемое значение

0 – список возможных частот дискретизации устройства прочитан полностью.

≥ 0 – возможная частота дискретизации заданного устройства.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа ZETServer.ocx выгружена из памяти процессора.

-3 – параметр *Zmodule < 0*.

-4 – параметр *Zmodule* > (количество устройств в системе – *1)*.

-5 – невозможно подключиться к устройству.

-6 – невозможно установить состояние выходного бита в логическую единицу для цифрового порта заданного устройства.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

## Работа с ШИМ

На модулях ZET 2XX АЦП-ЦАП 16/16 может быть установлен широтно-импульсный модулятор (ШИМ). Трех канальный модулятор может применяться для управления приводами, твердотельными реле в цепях управления и регулирования.

***GetPWMEnable*** – опрос наличия ШИМ на цифровых выходах выбранного устройства.

*long* ***GetPWMEnable****(long Zmodule)*

Параметры

© ООО "ЭТМС" 1992-2024. Все права защищены

*Zmodule* – номер устройства (сигнального процессора) в системе, наличие ШИМ у которого требуется узнать. Параметр *Zmodule* может принимать значение от *0* до (*QuanDSP() – 1)*.

Возвращаемое значение

0 – выбранное устройство не поддеживает ШИМ.

1 – выбранной устройство поддерживает ШИМ.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа ZETServer.ocx выгружена из памяти процессора.

-3 – параметр Zmodule < 0.

-4 – параметр Zmodule > (количество устройств в системе – 1).

-5 – невозможно подключиться к устройству.

-6 – невозможно опросить наличие ШИМ устройства.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер

-20 - превышено время обработки функции.

***SetFreqPWM*** – установка частоты ШИМ.

*long* ***SetFreqPWM*** *(long Zmodule, LONG Rate, LONG Period)*

Параметры

*Zmodule* – номер устройства (сигнального процессора) в системе, частоту ШИМ которого следует установить. Параметр *Zmodule* может принимать значение от *0* до (*QuanDSP() – 1)*.

*Rate* – коэффициент деления опорной частоты 96 МГц. Внутри модуля установлен кварцевый генератор с тактовой частотой 96 МГц. Параметр *Rate* может принимать значение от 1 до 96000000.

*Period* – период ШИМ. Параметр *Period* может принимать значение от 2 до 1024. Количество тактов модуляции задает степень скважности ШИМ. Выходная частота ШИМ равна (96000 / *Rate* / *Period*) кГц

Возвращаемое значение

0 – нормальное выполнение функции.

-1 – на момент вызова метода не было подключения к серверу данных.

© ООО "ЭТМС" 1992-2024. Все права защищены

-2 – программа *ZETServer.ocx* выгружена из памяти процессора.
-3 – параметр *Zmodule* < *0*.
-4 – параметр *Zmodule* > (количество устройств в системе – *1)*.
-5 – невозможно подключиться к устройству.
-6 – невозможно установить частоту ШИМ устройства.
-7 – выбранное устройство не поддерживает ШИМ.
-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().
-9 – номер декады decade < 0.
-10 – номер декады decade > 4.
-11 – момент времени time больше текущего времени для данного канала.
-12 – момент времени time < 0.
-13 – указатель на массив данных data = NULL.
-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".
-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().
-16 - пустой указатель на буфер
-20 - превышено время обработки функции.

***SetOnDutyPWM*** – установка скважности и фазы ШИМ по заданному каналу выбранного устройства.

*long* ***SetOnDutyPWM*** *(long Zmodule, LONG channel, LONG OnDutyPWM, LONG ShiftPWM)*

Параметры

*Zmodule* – номер устройства (сигнального процессора) в системе, скважность и фазу ШИМ которого следует установить. Параметр *Zmodule* может принимать значение от *0* до (*QuanDSP() – 1)*.

*channel* – номер канала, скважность и фазу ШИМ которого следует установить. Параметр *channel* может принимать значение от 0 до 2.

*OnDutyPWM* – скважность заданного канала ШИМ.

*ShiftPWM* – фаза заданного канала ШИМ относительно нулевого канала.

Возвращаемое значение

0 – нормальное выполнение функции.
-1 – на момент вызова метода не было подключения к серверу данных.
-2 – программа *ZETServer.ocx* выгружена из памяти процессора.
-3 – параметр *Zmodule* < *0*.
-4 – параметр *Zmodule* > (количество устройств в системе – *1)*.
-5 – невозможно подключиться к устройству.
-6 – невозможно установить скважность и фазу ШИМ по заданному каналу устройства.
-7 – выбранное устройство не поддерживает ШИМ.
-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().
-9 – номер декады decade < 0.

© ООО "ЭТМС" 1992-2024. Все права защищены

-10 – номер декады decade > 4.
-11 – момент времени time больше текущего времени для данного канала.
-12 – момент времени time < 0.
-13 – указатель на массив данных data = NULL.
-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".
-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().
-16 - пустой указатель на буфер
-20 - превышено время обработки функции.

***RegulatorPWM*** – установка произвольных данных для ШИМ.

*long* ***RegulatorPWM*** *(long Zmodule, FLOAT* data, LONG size)*

Параметры
*Zmodule* – номер устройства (сигнального процессора) в системе, данные ШИМ которого следует установить. Параметр *lModule* может принимать значение от *0* до (*QuanDSP() – 1)*.
*data* – данные для установки в ШИМ.
*size* – размер данных для установки в ШИМ.

Возвращаемое значение
0 – нормальное выполнение функции.
-1 – на момент вызова метода не было подключения к серверу данных.
-2 – программа *ZETServer.ocx* выгружена из памяти процессора.
-3 – параметр *Zmodule* < *0*.
-4 – параметр *Zmodule* > (количество устройств в системе – *1)*.
-5 – невозможно подключиться к устройству.
-6 – невозможно установить произвольные данные для ШИМ.
-7 – выбранное устройство не поддерживает ШИМ.
-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().
-9 – номер декады decade < 0.
-10 – номер декады decade > 4.
-11 – момент времени time больше текущего времени для данного канала.
-12 – момент времени time < 0.
-13 – указатель на массив данных data = NULL.
-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".
-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().
-16 - пустой указатель на буфер
-20 - превышено время обработки функции.

© ООО "ЭТМС" 1992-2024. Все права защищены

***StartPWM*** – запуск ШИМ по заданному каналу выбранного устройства.

*long* ***StartPWM*** *(long Zmodule, long Channel)*

Параметры

*Zmodule* – номер устройства (сигнального процессора) в системе, запуск ШИМ которого следует осуществить. Параметр *lModule* может принимать значение от *0* до (*QuanDSP() – 1)*.

*Channel* – номер канала, запуск ШИМ которого следует осуществить. Параметр *Channel* может принимать значение от 0 до 2.

Возвращаемое значение

0 – нормальное выполнение функции.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа *ZETServer.ocx* выгружена из памяти процессора.

-3 – параметр *Zmodule* < *0*.

-4 – параметр *Zmodule* > (количество устройств в системе – *1)*.

-5 – невозможно подключиться к устройству.

-6 – невозможно запустить ШИМ по заданному каналу устройства.

-7 – выбранное устройство не поддерживает ШИМ.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер

-20 - превышено время обработки функции.

***StopPWM*** – останов ШИМ по заданному каналу выбранного устройства.

*long* ***StopPWM*** *(long Zmodule, long lChannel)*

*Zmodule* – номер устройства (сигнального процессора) в системе, останов ШИМ которого следует осуществить. Параметр *lModule* может принимать значение от *0* до (*QuanDSP() – 1)*.

*Channel* – номер канала, останов ШИМ которого следует осуществить. Параметр *Channel* может принимать значение от 0 до 2.

Возвращаемое значение

0 – нормальное выполнение функции.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа *ZETServer.ocx* выгружена из памяти процессора.

-3 – параметр *Zmodule* < *0*.

© ООО "ЭТМС" 1992-2024. Все права защищены

-4 – параметр *Zmodule* > (количество устройств в системе – *1)*.
-5 – невозможно подключиться к устройству.
-6 – невозможно остановить ШИМ по заданному каналу устройства.
-7 – выбранное устройство не поддерживает ШИМ.
-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().
-9 – номер декады decade < 0.
-10 – номер декады decade > 4.
-11 – момент времени time больше текущего времени для данного канала.
-12 – момент времени time < 0.
-13 – указатель на массив данных data = NULL.
-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".
-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().
-16 - пустой указатель на буфер
-20 - превышено время обработки функции.

## Работа с ICP

***PrusEna*** – опрос наличия усилителя заряда (ICP) для подключения вибродатчиков и микрофонов к заданному каналу.

*long* ***PrusEna*** *(long channel)*

Параметры
*channel* – номер канала, наличие усилителя заряда у которого требуется узнать. Параметр *channel* может принимать значение от *0* до (*QuanChan() – 1)*.

Возвращаемое значение
0 – усилитель заряда отсутствует.
1 – наличие усилителя заряда ПУ 8/10.
2 – наличие усилителя заряда ICP.
-1 – на момент вызова метода не было подключения к серверу данных.
-2 – программа ZETServer.ocx выгружена из памяти процессора.
-3 – параметр channel < 0.
-4 – параметр channel > (QuanChan() – 1).
-5 – заданный канал не является каналом модуля АЦП-ЦАП.
-6 – серверу данных не хватает памяти для организации буферов данных.
-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.
-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().
-9 – номер декады decade < 0.
-10 – номер декады decade > 4.
-11 – момент времени time больше текущего времени для данного канала.
-12 – момент времени time < 0.
-13 – указатель на массив данных data = NULL.

© ООО "ЭТМС" 1992-2024. Все права защищены

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

***GetICP*** – опрос состояния усилителя заряда (ICP) для подключения вибродатчиков и микрофонов к заданному каналу.

*long* ***GetICP*** *(long Mod, LONG Chan)*

Параметры

*Mod* – номер устройства (сигнального процессора) в системе, состояние усилителя заряда канала которого требуется узнать. Параметр *lModule* может принимать значение от *0* до (*QuanDSP() – 1)*.

*Chan* – номер канала, состояние усилителя заряда у которого требуется узнать. Параметр *Chan* может принимать значение от *0* до (*QuanChan() – 1)*.

Возвращаемое значение

0 – усилитель заряда ICP выключен.

1 – усилитель заряда ICP подключен.

-2 – усилитель заряда у каналов заданного устройства отсутствует.

***SetICP*** – установка состояния усилителя заряда (ICP) для подключения вибродатчиков и микрофонов к заданному каналу.

*long* ***SetICP*** *(long Mod, LONG Chan, LONG type)*

Параметры

*Mod* – номер устройства (сигнального процессора) в системе, состояние усилителя заряда канала которого следует установить. Параметр *Mod* может принимать значение от *0* до (*QuanDSP() – 1)*.

*Chan* – номер канала, состояние усилителя заряда которого следует установить. Параметр *Chan* может принимать значение от *0* до (*QuanChan() – 1)*.

*type* – состояние усилителя заряда, которое требутся установить.

0 – усилитель заряда ICP выключен.

1 – усилитель заряда ICP подключен.

Возвращаемое значение

0 – нормальное выполнение функции

-2 – усилитель заряда у каналов заданного устройства отсутствует.

© ООО "ЭТМС" 1992-2024. Все права защищены

## Работа с GPS

***PutDataGPS*** – запись данных NMEA-потока в сервер данных.

*long* ***PutDataGPS*** *(LPCTSTR nmea, LONG size)*

Параметры

*nmea* – строка с данными NMEA-потока для записи в сервер данных.
*size* – размер данных NMEA-потока для записи в сервер данных.

Возвращаемое значение

0 – нормальное выполнение функции

***GetDataGPS*** – запрос данных NMEA-потока, которые записываются в сервер данных методом *PutDataGPS()*.

*BSTR* ***GetDataGPS*** *(long maxsize, LONG* realsize)*

Параметры

*maxsize* – максимальный размер запрашиваемых данных.
*realsize* – возвращает истинный размер прочитанных из сервера данных.

Возвращаемое значение

строка – данные NMEA-потока.

***SetShiftGPS*** – установка смещения времени компьютера относительно времени по GPS.

*long* ***SetShiftGPS*** *(float seconds)*

Параметры

*seconds* – значение смещения времени компьютера относительно времени по GPS.

Возвращаемое значение

0 – нормальное выполнение функции

***GetShiftGPS*** – запрос смещения времени компьютера относительно времени по GPS.

*float* ***GetShiftGPS*** *(void)*

Возвращаемое значение

значение смещения времени компьютера относительно времени по GPS.

***enaGPSync*** – установка возможности синхронизации устройств по GPS.

*long* ***enaGPSync*** *(long enable)*

Параметры

*enable* – флаг синхронизации по GPS (0 – отключение синхронизации, ≥ 0 – включение синхронизации устройств по GPS)

© ООО "ЭТМС" 1992-2024. Все права защищены

Возвращаемое значение

0 – синхронизация по GPS отключена.

≥ 0 – синхронизация по GPS включена.

***advanceGPS*** – установка времени опережения на синхронизацию устройств по GPS (время через которое должна произойти синхронизация).

*long* ***advanceGPS*** *(long seconds)*

Параметры

*seconds* – время, через которое должна произойти синхронизация устройств по GPS (в секундах)

Возвращаемое значение

≥ 0 – время, через которое должна произойти синхронизация устройств по GPS (в секундах).

***satGPS*** – установка количества спутников GPS, наблюдаемых на данный момент.

*long* ***satGPS*** *(long satinfo)*

Параметры

*long satinfo* – количество спутников GPS, наблюдаемое на данный момент.

Возвращаемое значение

0 – нормальное выполнение функции.

***IsFileGPS*** – опрос источника данных NMEA-потока.

*long* ***IsFileGPS*** *(void)*

Возвращаемое значение

0 – источник данных – устройство, принимающее информацию.

1 – источник данных – файл, записанный ранее при помощи программы «Запись сигналов».

***SetStartUTCTime*** – установка времени, в которое должна произойти синхронизация по GPS.

*long* ***SetStartUTCTime*** *(ULONGLONG UtcTime)*

Параметры

*UtcTime* – время, в которое должна произойти синхронизация по GPS. Время представляется как количество секунд с момента времени 00:00:00 01.01.1970.

Возвращаемое значение

0 – нормальное выполнение функции.

***timeGPS***– показвает время в которое произошла синхронизация.

*long* ***timeGPS*** *(ULONGLONG timeUTC)*

Параметры

© ООО "ЭТМС" 1992-2024. Все права защищены

*UtcTime* – время, в которое должна произойти синхронизация по GPS. Время представляется как количество секунд с момента времени 00:00:00 01.01.1970.

Возвращаемое значение

0 – нормальное выполнение функции.

## Прием и передача данных

***CurrentTime*** – определение текущего времени по заданному каналу. Сервер данных ведет учет текущего времени каждого канала. В общем случае значения текущего времени для различных каналов могут быть разными. Например, при организации виртуального канала, как фильтра физического канала, виртуальный канал будет отставать от физического канала. Для совместной обработки каналов необходимо дождаться, чтобы по всем каналам было произведено необходимое накопление. Время указывается от начала запуска АЦП при работе в режиме реального времени, или в режиме чтения данных из файлов данных – время от начала файла. При включении программ генераторов и виртуальных каналов, при изменении параметров АЦП (количества каналов, частоты дискретизации и т.д.) сервер перезапускает модули АЦП и время сбрасывается в 0. При этих изменениях сервер посылает события во все программы, присоединенные к серверу.

*double* ***CurrentTime*** *(long channel)*

Параметры

*channel* – номер канала, текущее время которого требуется узнать. Параметр *channel* может принимать значение от *0* до (*QuanChan() – 1)*.

Возвращаемое значение

≥ 0 – текущее время по заданному каналу в секундах с момента запуска сервера данных.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа ZETServer.ocx выгружена из памяти процессора.

-3 – параметр channel < 0.

-4 – параметр channel > (QuanChan() – 1).

-5 – значение частоты дискретизации по заданному каналу меньше нуля.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

© ООО "ЭТМС" 1992-2024. Все права защищены

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

***GetData*** – запрос данных по заданному каналу в буфер данных пользователя.

*long* ***GetData*** *(long LONG channel, LONG decade, DOUBLE time, LONG size, FLOAT* data)*

Параметры

*channel* – номер канала, данные по которому требуется запросить. Параметр *channel* может принимать значение от *0* до (*QuanChan() – 1)*.

*decade* – номер декады от 0 до 4.

*time* – момент времени, в который необходимо взять данные для обработки.

*size* – количество отсчетов данных для передачи. Параметр *lSize* может принимать значение от 1 до *DecadeBufferSize()*. *DecadeBufferSize()* – метод для определения размера буфера по декаде;

*data* – указатель на массив данных пользователя, массив данных в плавающей запятой одинарной точности.

Возвращаемое значение:

1 - данные содержат не менее одного значения NaN

0 – нормальное выполнение функции.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа *ZETServer.ocx* выгружена из памяти процессора.

-3 – параметр *channel < 0*.

-4 – параметр *channel* > (*QuanChan() – 1)*.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных *Size ≤ 0* или *size ≥ DecadeBufferSize().*

-9 – номер декады *decade < 0.*

-10 – номер декады *decade > 4.*

-11 – момент времени *time* больше текущего времени для данного канала.

-12 – момент времени *time < 0*.

-13 – указатель на массив данных *data = NULL.*

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

Сервер производит цифровую фильтрацию всех входных потоков данных со всех каналов АЦП. Например, оцифровка сигналов производится на частоте 25 кГц. Поток

© ООО "ЭТМС" 1992-2024. Все права защищены

на этой частоте соответствует декаде с номером 0. Сервер производит цифровую фильтрацию сигналов и прореживание, и после этого получаются потоки с частотами 2500 Гц, 250 Гц, 25 Гц, 2,5 Гц. Потоку с частотой оцифровки в 10 раз меньше частоты дискретизации, т.е. с частотой 2500 Гц соответствует декада номер 1. Потоку с частотой оцифровки в 100 раз меньше частоты дискретизации, т.е. с частотой 250 Гц соответствует декада номер 2 и так далее до декады номер 4. Цифровая фильтрация необходима при работе с медленно меняющимися сигналами. При этом можно устанавливать на модуле максимально возможную частоту дискретизации и работать с теми потоками, которые необходимы. В результате цифровой фильтрации подавляются все помехи вне полосы пропускания декады (*f / 2.5*) и повышается точность представления данных в данном потоке. *time* – это момент времени, в который необходимо взять данные для обработки. Например, текущий момент времени сервера по заданному каналу (метод *CurrentTime()*) равен 10,5 секундам. Это значит, что есть в наличии данные по выбранному каналу до момента времени 10.5. Если частота дискретизации равна 25 кГц, то метод

*myerr = GetData(channel, 1, 10.5, 1250, data);*

передает в массив пользователя *data* 1250 отсчетов потока выбранного канала с частотой оцифровки 2500 Гц, что составляет 0,5 с. В массиве *data* находятся данные с 10 секунды по 10,5 секунду.

***GetDataVar*** – запрос данных по заданному каналу в буфер данных пользователя. (зарезервирована под будущее использование).

*long* ***GetDataVar*** *(long channel, LONG decade, DOUBLE time, LONG size, VARIANT* datvar)*

***GetDataNet*** – запрос данных по заданному каналу в буфер данных пользователя в .net языках.

*long* ***GetDataNet*** *(long channel, LONG decade, DOUBLE time, LONG size, LONG pData)*

Параметры

*channel* – номер канала, данные по которому требуется запросить. Параметр *channel* может принимать значение от *0* до (*QuanChan() – 1)*.

*decade* – номер декады от 0 до 4.

*time* – момент времени, в который необходимо взять данные для обработки.

*size* – количество отсчетов данных для передачи. Параметр *size* может принимать значение от 1 до *DecadeBufferSize()*. *DecadeBufferSize()* – метод для определения размера буфера по декаде;

*pData* – указатель на массив данных пользователя, массив данных в плавающей запятой одинарной точности.

Возвращаемое значение

1 - данные содержат не менее одного значения NaN

0 – нормальное выполнение функции.

-1 – на момент вызова метода не было подключения к серверу данных.

© ООО "ЭТМС" 1992-2024. Все права защищены

-2 – программа *ZETServer.ocx* выгружена из памяти процессора.

-3 – параметр *channel < 0*.

-4 – параметр *channel* > (*QuanChan() – 1)*.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных *Size ≤ 0* или *size ≥ DecadeBufferSize().*

-9 – номер декады *decade < 0.*

-10 – номер декады *decade > 4.*

-11 – момент времени *time* больше текущего времени для данного канала.

-12 – момент времени *time < 0*.

-13 – указатель на массив данных *data = NULL.*

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

***PutData –*** передача данных из буфера пользователя в буфер виртуального канала или буфер ЦАП.

*long* ***PutData*** *(long channel, DOUBLE time, LONG size, FLOAT* data)*

Параметры

*channel* – номер канала. Если функция используется для передачи данных в буфер виртуального канала, то параметр *channel* может принимать значение от *0* до (количество виртуальных каналов, установленных функцией *ConnectVrtCh (...)* минус 1). Если функция используется для передачи данных в буфер ЦАП, то параметр *channel* может принимать значение от *0* до (*MaxQuanChanDac () – 1)*.

*time* – момент времени, в который необходимо передать данные в буфер виртуального канала или буфер ЦАП.

*size* – количество отсчетов данных для передачи. Параметр *size* может принимать значение от 1 до *BufferSize()*. *BufferSize()* – метод для определения размера.

*data* – указатель на массив данных пользователя, массив данных в плавающей запятой одинарной точности.

Возвращаемое значение

0 – нормальное выполнение функции.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа *ZETServer.ocx* выгружена из памяти процессора.

-3 – параметр *channel < 0*.

-5 – параметр *channel* > (*MaxQuanChanDac () – 1)*.

-6 – передача данных в неподключенный канал ЦАП.

-7 – серверу данных не хватает памяти для организации буферов данных.

© ООО "ЭТМС" 1992-2024. Все права защищены

-8 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-10 – размер передаваемых данных *size ≤ 0*

-11 – момент времени *time < 0*.

-12 – подключение к серверу было произведено методом *Connect()*, который не организует в сервере виртуальных каналов. Метод *PutData()* работает только при подключении к серверу методами *ConnectDac(), ConnectVrtCh()*.

-15 – момент времени *time* отстает от внутреннего времени, которое можно определить с помощью метода *CurrentTime()*.

-105 – нет организованного программой виртуального канала с номером *channel.*

-205 – параметр *channel* > (*QuanChan() – 1)*.

-106 – значение частоты дискретизации по заданному каналу меньше нуля.

-109 – размер передаваемых данных в виртуальный канал *size ≥ BufferSize() / 2*.

Программа пользователя, организующая виртуальный канал должна периодически обращаться к методу *PutData()*. Текущее время по заданному каналу, т.е. время, определяемое методом *CurrentTime()*, будет задаваться параметром *time* метода *PutData()*.

Рассмотрим фрагмент программы суммирования двух сигналов. Например, текущий момент времени сервера по нулевому каналу (метод *CurrentTime()*) равен 10,5. Это значит, что есть данные по выбранному каналу до момента времени 10.5. Если частота дискретизации равна 25 кГц, то метод

```
myerr = GetData(channel, 0, 10.5, 12500, data);
```

передает в массив пользователя *data* 12500 отсчетов потока выбранного канала с частотой оцифровки 25000 Гц, что составляет 0,5 с. В массиве *data* находятся данные с 10 секунды по 10,5 секунду. Необходимо дождаться, чтобы по второму каналу текущее время было не меньше 10,5 с. И затем передать данные от этого канала в пользовательский буфер

```
myerr = GetData(channel2, 0, 10.5, 12500, data2);
```

Теперь можно просуммировать данные

```
for(int i = 0; i < 12500; ++i )
        data3[i] = data[i] + data2[i];
```

и передать данные в сервер данных в виртуальный канал

```
myerr = PutData(channelVirt, 11.0, 12500, data3);
```

Метод передал из массива *data3*12500 отсчетов во внутренний буфер сервера. Теперь во внутреннем буфере виртуального канала сервера находятся данные с 10,5 по 11,0 секунду. Необходимо, чтобы параметр *time* был больше текущего времени по каналу, иначе данные попадут в «прошлое» и потеряются.

***PutDataNet*** **–** передача данных из буфера пользователя в буфер виртуального канала или буфер ЦАП в .net языках.

© ООО "ЭТМС" 1992-2024. Все права защищены

*long* ***PutDataNet*** *(long channel, LONG decade, DOUBLE time, LONG size, LONG pData)*

Параметры

*channel* – номер канала. Если функция используется для передачи данных в буфер виртуального канала, то параметр *channel* может принимать значение от *0* до (количество виртуальных каналов, установленных функцией *ConnectVrtCh (...)* минус 1). Если функция используется для передачи данных в буфер ЦАП, то параметр *channel* может принимать значение от *0* до (*MaxQuanChanDac () – 1)*.

*time* – момент времени, в который необходимо передать данные в буфер виртуального канала или буфер ЦАП.

*size* – количество отсчетов данных для передачи. Параметр *size* может принимать значение от 1 до *BufferSize()*. *BufferSize()* – метод для определения размера.

*pData* – указатель на массив данных пользователя, массив данных в плавающей запятой одинарной точности.

Возвращаемое значение

0 – нормальное выполнение функции.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа *ZETServer.ocx* выгружена из памяти процессора.

-3 – параметр *channel* < *0*.

-5 – параметр *channel* > (*MaxQuanChanDac () – 1)* для канала ЦАП или нет организованного программой виртуального канала с номером *channel* или *channel* > (*QuanChan () – 1)* для виртуального канала.

-6 – передача данных в неподключенный канал ЦАП или значение частоты дискретизации по заданному виртуальному каналу меньше нуля.

-7 – серверу данных не хватает памяти для организации буферов данных.

-8 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-9 – размер передаваемых данных в виртуальный канал *lSize* ≥ *BufferSize() / 2*

-10 – размер передаваемых данных *size* ≤ *0*

-11 – момент времени *time* < *0*.

-12 – подключение к серверу было произведено методом *Connect()*, который не организует в сервере виртуальных каналов. Метод *PutData()* работает только при подключении к серверу методами *ConnectDac(), ConnectVrtCh()*.

-15 – момент времени *time* отстает от внутреннего времени, которое можно определить с помощью метода *CurrentTime()*.

***DecadeBufferSize*** – опрос размера буфера сервера данных по декаде.

*long* ***DecadeBufferSize*** *(long decade)*

Параметры

*decade* – номер декады от 0 до 4.

Возвращаемое значение

≥ 0 – размер буфера сервера данных в отсчетах для каждого канала по декаде. Под отсчетом в данном случае подразумевается полученное с заданной частотой

© ООО "ЭТМС" 1992-2024. Все права защищены

дискретизации одно значение, т.е. при частоте дискретизации 25 кГц за секунду можно получить 25000 отсчетов. При работе программы обработки сигналов возвращаемое значение – это максимальный размер данных, которые можно передать методом *GetData(...)*.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа *ZETServer.ocx* выгружена из памяти процессора.

-3 – параметр *channel < 0*.

-4 – параметр *channel* > (*QuanChan() – 1)*.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных *Size ≤ 0* или *size ≥ DecadeBufferSize().*

-9 – номер декады *decade < 0.*

-10 – номер декады *decade > 4.*

-11 – момент времени *time* больше текущего времени для данного канала.

-12 – момент времени *time < 0*.

-13 – указатель на массив данных *data = NULL.*

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

Буфер данных также может переполняться. При этом чтобы избежать ошибок, нужно отлавливать эти моменты. Например, существует два варианта настройки системы:

Количество каналов q = 8
Частота дискретизации f = 50000
Размер буфера size = 625000

Количество каналов q = 4
Частота дискретизации f = 50000
Размер буфера size = 1250000

Отсюда следует, что переполнение буфера наступит через:

1) *t* = (*size* = 625000) / *f* = 50000) = 12,5 секунд;
2) *t* = (*size* = 1250000) / *f* = 50000) = 25 секунд.

***BufferSize*** – опрос размера буфера сервера данных по нулевой декаде.

*long* ***BufferSize*** *(void)*

Возвращаемое значение

© ООО "ЭТМС" 1992-2024. Все права защищены

≥ 0 – размер буфера сервера данных в отсчетах для каждого канала по нулевой декаде. Под отсчетом в данном случае подразумевается полученное с заданной частотой дискретизации одно значение, т.е. при частоте дискретизации 25 кГц за секунду можно получить 25000 отсчетов. При работе программы обработки сигналов возвращаемое значение – это максимальный размер данных, которые можно передать методом *GetData(...)* по нулевой декаде.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа *ZETServer.ocx* выгружена из памяти процессора.

***NumFileUsed*** – команда для сервера данных прочитать очередную порцию данных из файла данных при работе в режиме чтения файлов. При работе программы в режиме чтения файлов необходимо отмечать номера каналов, по которым производится чтение данных. Для этого в процедуре обработки данных, после чтения данных функцией *GetData()*, необходимо обращаться к функции *NumFileUsed()*, чтобы сервер прочитал новую порцию данных из файла. При обработке одновременно нескольких каналов, необходимо выполнить несколько обращений *NumFileUsed()* по каждому каналу.

*long* ***NumFileUsed*** *(long NumFile)*

Параметры

*NumFile* – номер канала, команду на прочтение очередной порции данных которому требуется послать. Параметр *NumFile* может принимать значение от *0* до (*QuanChan() – 1)*.

Возвращаемое значение:

0 – нормальное выполнение функции.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа *ZETServer.ocx* выгружена из памяти процессора.

-3 – параметр *NumFile < 0*.

-4 – параметр *NumFile* > (*QuanChan() – 1)*.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных *Size ≤ 0* или *size ≥ DecadeBufferSize().*

-9 – номер декады *decade < 0.*

-10 – номер декады *decade > 4.*

-11 – момент времени *time* больше текущего времени для данного канала.

-12 – момент времени *time < 0*.

-13 – указатель на массив данных *data = NULL.*

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

© ООО "ЭТМС" 1992-2024. Все права защищены

## Вспомогательные информационные функции

***AboutBox*** – содержит информацию о программе

*void* ***AboutBox****()*

***OrderConnected*** – определение номера копии программы, загруженной в память. При запуске большого количества одной и той же программы, например, программы вольтметры переменного тока, необходимо чтобы каждая копия программы настраивалась по-разному. При помощи метода *OrderConnected()* программа определяет, какая ее копия загрузилась и считывает соответствующий файл конфигурации **.cfg*.

*long* ***OrderConnected*** *(void)*

Возвращаемое значение

≥ 0 – номер копии программы, загруженной в память.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа *ZETServer.ocx* выгружена из памяти процессора.

***GetStartTime*** – определение времени старта сервера данных (по времени компьютера).

*double* ***GetStartTime*** *(void)*

Возвращаемое значение

≥ 0 – время старта сервера данных, которое определяется, как количество секунд, прошедших с момента 00:00:00 01.01.1970.

***WorkingTime*** – определение времени работы сервера данных (по времени компьютера).

*double* ***WorkingTime*** *(void)*

Возвращаемое значение

≥ 0 – время работы сервера данных, которое определяется, как количество секунд, прошедших с момента старта сервера данных.

***GroupName*** – определение группового имени канала АЦП. Поскольку на сервере данных могут быть каналы от разных устройств, разных компьютеров, находящихся на большом расстоянии друг от друга, необходимо группировать каналы по некоторым признакам. Таким признаком выступает устройство или программа, которая порождает конкретный виртуальный канал. Это первый уровень вложенности. Если канал порождается путем передачи данных с других компьютеров, то его групповое имя дополняется именем компьютера, откуда он передается на данный компьютер.

© ООО "ЭТМС" 1992-2024. Все права защищены

*BSTR* ***GroupName*** *(long channel)*
Параметры

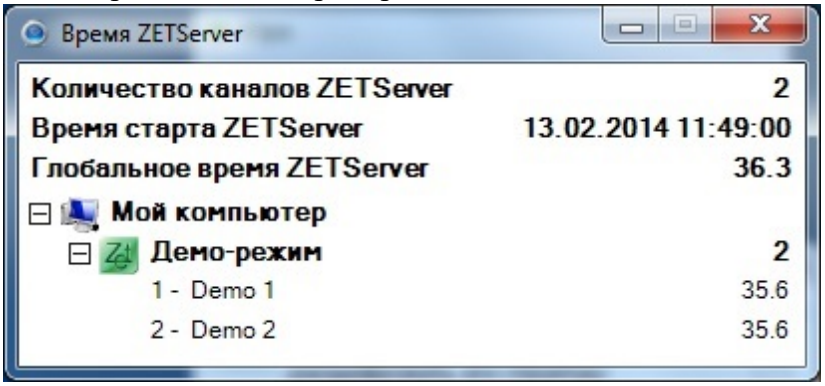
*channel* – определение группового имени канала - имя устройства или имя программы, которая порождает этот канал. Параметр *channel* может принимать значение от *0* до (*QuanChan () – 1)*.

Возвращаемое значение

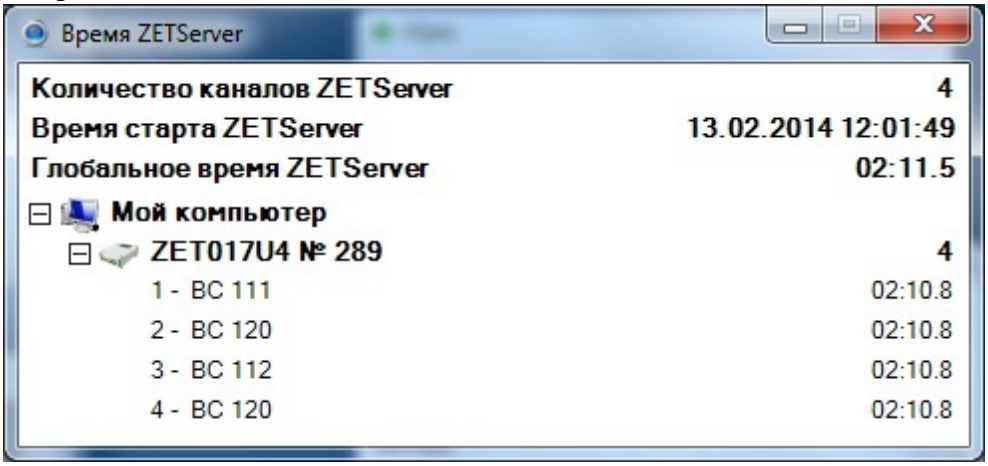
непустая строка – групповое имя канала.

Примеры, групповых имен и серийных номеров приборов АЦП, подключенных разными способами.
Создание и использование групповых имен и номеров можно увидеть через программу "**Время ZETServer**" меню "**Сервисные**":

- Демо-режим, когда прибор не подключен.

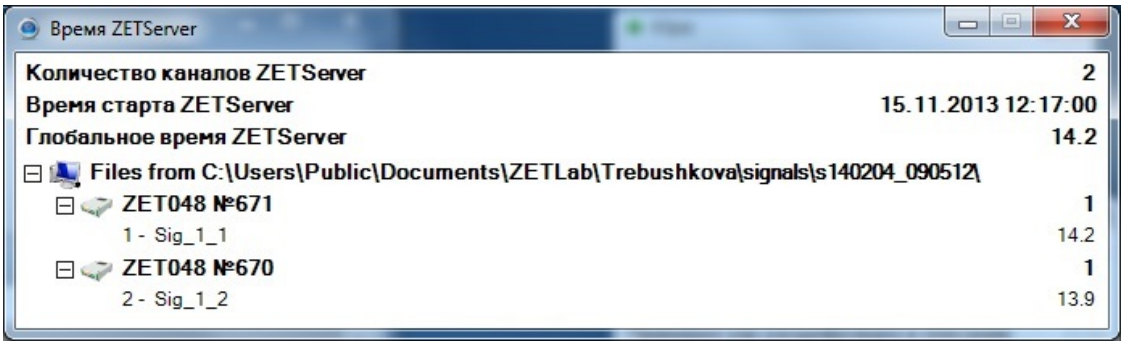



- Определяется имя и номер прибора, когда прибор подключен через USB или через Ethernet.

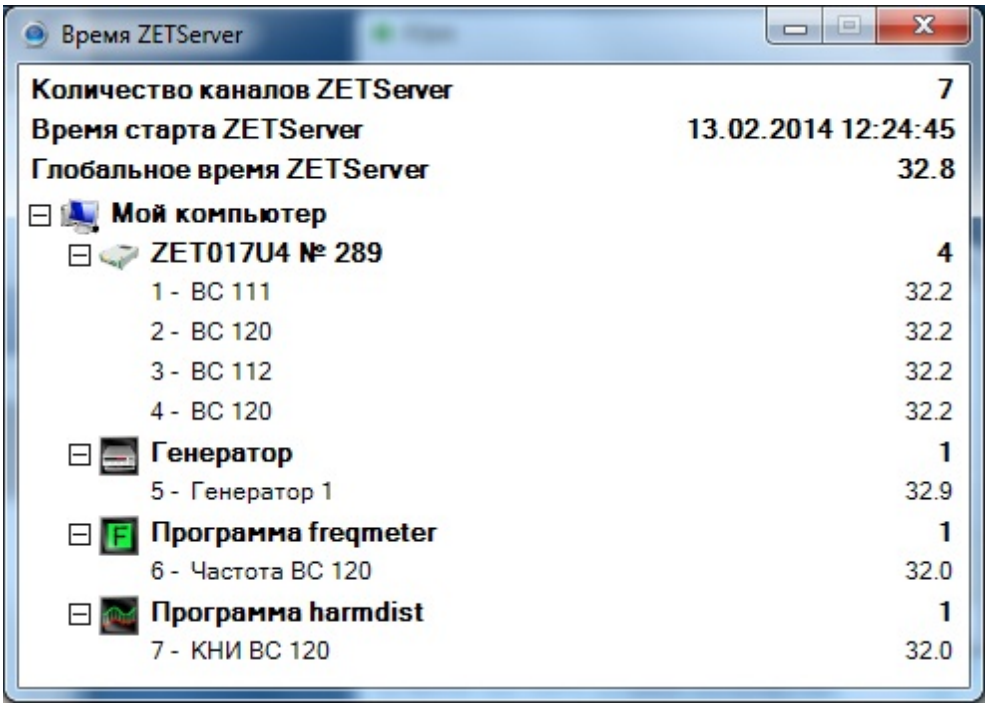



- Определяется имя и номер прибора, когда запущена программа Воспроизведение.

© ООО "ЭТМС" 1992-2024. Все права защищены

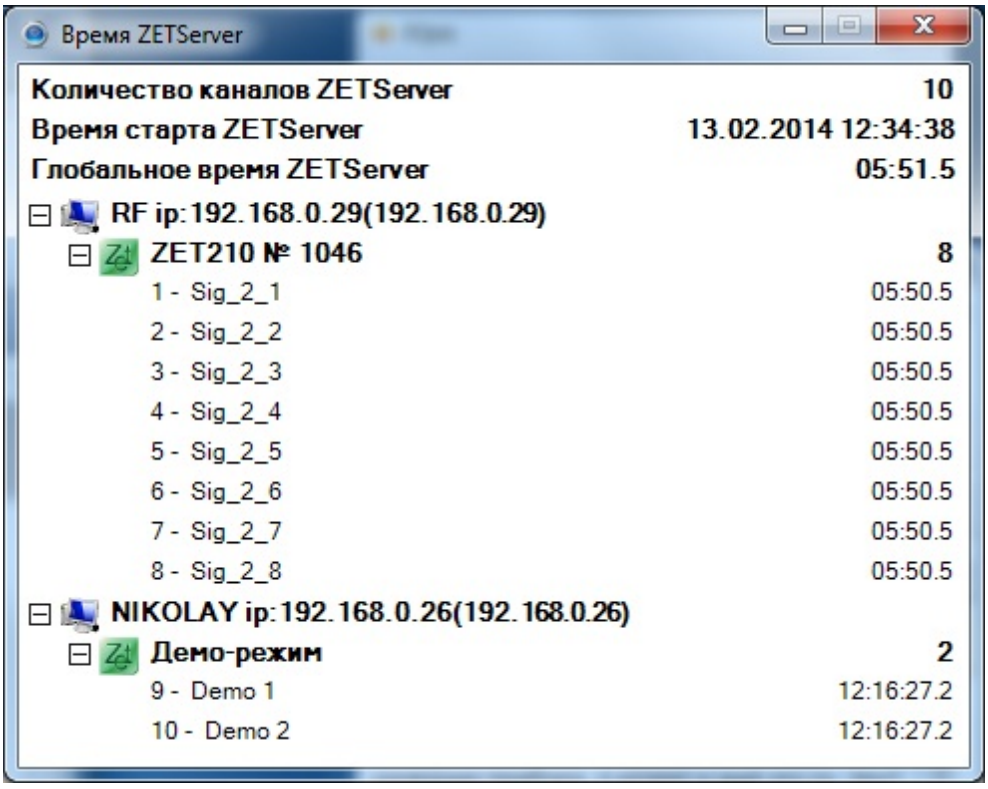



- Определяется имя и номер прибора, когда прибор подключен через USB или через Ethernet, а также включены программы, которые порождают виртуальные каналы.




- Определяется имя и номер прибора, если канал порождается путем передачи данных с других компьютеров, то его групповое имя дополняется именем компьютера, откуда он передается на данный компьютер.

© ООО "ЭТМС" 1992-2024. Все права защищены




***GroupName*** – определение группового имени канала АЦП. Поскольку на сервере данных могут быть каналы от разных устройств, разных компьютеров, находящихся на большом расстоянии друг от друга, необходимо группировать каналы по некоторым признакам. Таким признаком выступает устройство или программа, которая порождает конкретный виртуальный канал. Это первый уровень вложенности. Если канал порождается путем передачи данных с других компьютеров, то его групповое имя дополняется именем компьютера, откуда он передается на данный компьютер.

*BSTR* ***GroupNameGUID****(GUID* GUIDchannel)*
Параметры

*GUIDchannel* – определение группового имени канала - имя устройства или имя программы, которая порождает этот канал по Guid.

Возвращаемое значение

непустая строка – групповое имя канала.

Примеры, групповых имен и серийных номеров приборов АЦП, подключенных разными способами.
Создание и использование групповых имен и номеров можно увидеть через программу "**Время ZETServer**" меню "**Сервисные**":

© ООО "ЭТМС" 1992-2024. Все права защищены

***DacGroupName*** – определение группового имени канала ЦАП. Поскольку на сервере данных могут быть каналы ЦАП от разных устройств, разных компьютеров, находящихся на большом расстоянии друг от друга, необходимо группировать каналы ЦАП по некоторым признакам. Таким признаком выступает устройство, которая порождает конкретный канал ЦАП.

*BSTR* ***DacGroupName*** *(long dacchannel)*

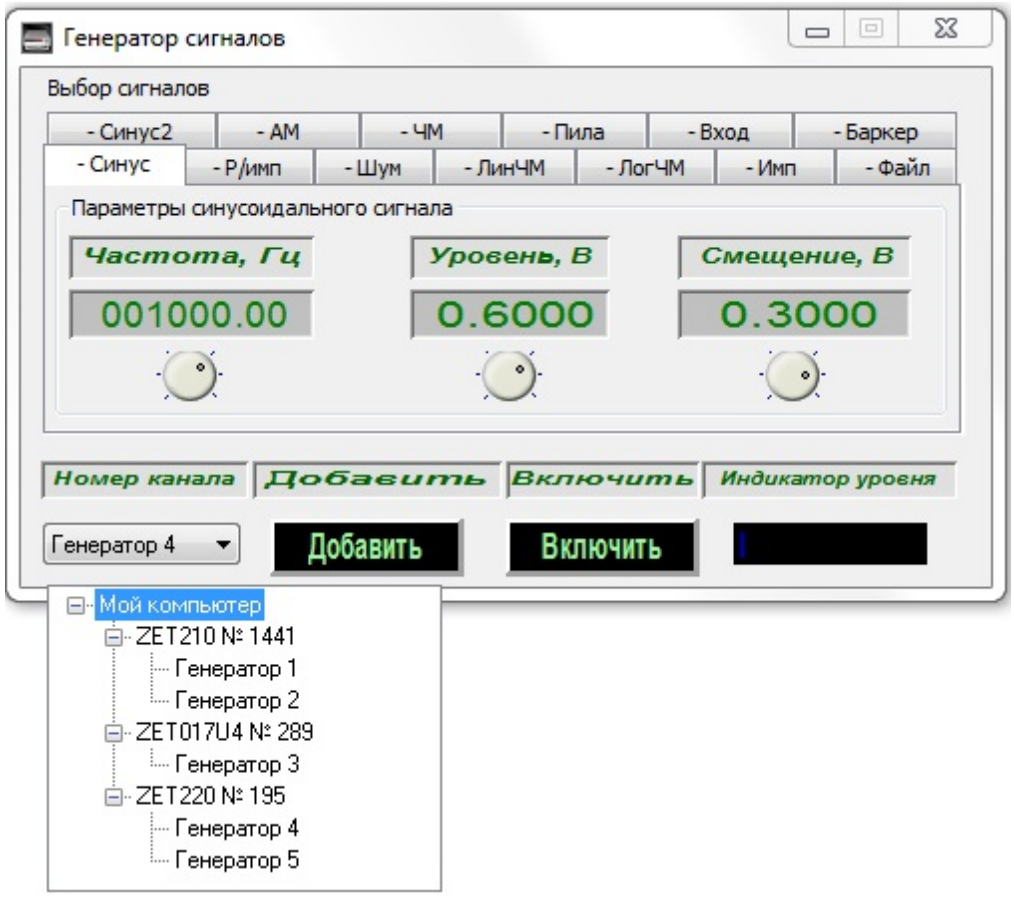
Параметры

*dacchannel* – номер канала ЦАП, групповое имя которого требуется узнать. Параметр *dacchannel* может принимать значение от *0* до (*MaxQuanChanDac () – 1)*.

Возвращаемое значение

непустая строка – групповое имя канала.

Пример, групповых имен и серийные номера прибора ЦАП, используется в программах "**Генераторе сигналов**" и "**Синхронном генераторе**".




***PointAllInfo*** – содержит информацию о программе.

*long* ***PointAllInfo*** *(IDispatch** point)*

Параметры

© ООО "ЭТМС" 1992-2024. Все права защищены

*IDispatch** point -*

***NullPointer –*** программа пользователя может отслеживать начальный указатель накопления буфера, и определять, сколько новых данных записалось в память после последнего обращения. Для удобства работы размер буфера желательно делать кратным количеству включенных каналов.

*long **NullPointer** (LONG channel, LONG decade)*

Параметры

*channel* – номер канала, имя (название) которого требуется узнать. Имя (название) канала задается в таблице конфигурации *Devices.cfg* и программой «Диспетчер устройств» из группы «Сервисные». Параметр *channel* может принимать значение от *0* до (*QuanChan() – 1)*.

*decade* – номер декады от 0 до 4.

Возвращаемое значение

0 – нормальное выполнение функции.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа *ZETServer.ocx* выгружена из памяти процессора.

-3 – параметр *channel* < *0*.

-4 – параметр *channel* > (*QuanChan() – 1)*.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных *size* ≤ *0* или *size* ≥ *DecadeBufferSize().*

-9 – номер декады *decade* < *0.*

-10 – номер декады *decade* > *4.*

-11 – момент времени *time* больше текущего времени для данного канала.

-12 – момент времени *time* < *0*.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

***Pointer -*** программа пользователя может отслеживать текущий указатель накопления буфера, и определять, сколько новых данных записалось в память после последнего обращения. Для удобства работы размер буфера желательно делать кратным количеству включенных каналов.

*long **Pointer** (LONG channel, LONG decade)*

Параметры

© ООО "ЭТМС" 1992-2024. Все права защищены

*channel* – номер канала, имя (название) которого требуется узнать. Имя (название) канала задается в таблице конфигурации *Devices.cfg* и программой «Диспетчер устройств» из группы «Сервисные». Параметр *channel* может принимать значение от *0* до (*QuanChan() – 1)*.

*decade* – номер декады от 0 до 4.

Возвращаемое значение

0 – нормальное выполнение функции.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа *ZETServer.ocx* выгружена из памяти процессора.

-3 – параметр *channel* < *0*.

-4 – параметр *channel* > (*QuanChan() – 1)*.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных *size* ≤ *0* или *size* ≥ *DecadeBufferSize().*

-9 – номер декады *decade* < *0.*

-10 – номер декады *decade* > *4.*

-11 – момент времени *time* больше текущего времени для данного канала.

-12 – момент времени *time* < *0*.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

***TalkAbout*** - программа пользователя, подключенная через компонент *Unit*

*long* ***TalkAbout*** *(LONG Param, LONG Value)*

Параметры

*param* – номер устанавливаемого параметра.

*value* – значение устанавливаемого числового параметра.

Возвращаемое значение

*0* – нормальное выполнение функции.

*-1* – не было подключения к программе через компонент *Unit*.

*-2* – запущенная через компонент *Unit* программа выгружена из памяти.

*-3* – не хватает хэндлеров памяти.

*-4* – не хватает хэндлеров памяти.

*-5* – превышена максимальная допустимая длина числового параметра.

***SetSrvWin*** - установление не известному каналу названия

© ООО "ЭТМС" 1992-2024. Все права защищены

*long* ***SetSrvWin*** *(IUnknown* Hwnd)*

Параметры

*Hwnd* - новое название канала вместо не известного

***SetChannelStatus*** – установление статуса канала.

*long* ***SetChannelStatus*** *(LONG channel, LONG status)*

Параметры

*channel* – номер канала, тип которого требуется узнать. Параметр *channel* может принимать значение от *0* до (*QuanChan() – 1)*.

*status - определяет статус канала.*

Возвращаемое значение

0 – заданный канал является каналом АЦП.
1 – заданный канал является каналом ЦАП.
2 – заданный канал является виртуальным каналом.
3 – заданный канал является цифровым каналом.
4 – заданный канал является отключенным каналом устройства.
5 – заданный канал является каналом отключенного устройства.
6 – заданный канал является интеллектуальным датчиком.
7 – заданный канал является каналом отключенного интеллектуального датчика.
8 – заданный канал является каналом скоростного устройства, которое выдаёт данные порциями по таймеру или по событию.

***GetPulse*** - опрос частоты опроса канала.

*long* ***GetPulse*** *(LONG channel, LONG index, DOUBLE* time, LONG* size, FLOAT* data)*

Параметры

*channel* – номер канала, тип которого требуется узнать. Параметр *channel* может принимать значение от *0* до (*QuanChan() – 1)*.

*index* - индекс указателя канала.

*time* – момент времени, в который необходимо взять данные для обработки.
*size* – количество отсчетов данных для передачи. Параметр *size* может принимать значение от 1 до *BufferSize()*. *BufferSize()* – метод для определения размера.
*pData* – указатель на массив данных пользователя, массив данных в плавающей запятой одинарной точности.

Возвращаемое значение

1 - данные содержат не менее одного значения NaN
0 – нормальное выполнение функции.
-1 – на момент вызова метода не было подключения к серверу данных.

© ООО "ЭТМС" 1992-2024. Все права защищены

-2 – программа ZETServer.ocx выгружена из памяти процессора.

-3 – параметр channel < 0.

-4 – параметр channel > (QuanChan() – 1).

-5 – заданный канал не является каналом модуля АЦП-ЦАП.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

***GetDatawFreq*** – опрос данных с другой частотой передискретизации

*long* ***GetDatawFreq*** *(long channel, float freq, double time, long size, float * data)*

Параметры

*channel* – номер канала, данные по которому требуется запросить. Параметр *channel* может принимать значение от *0* до (*QuanChan() – 1)*.

*freq* - опрос частоты дискретизации заданного канала

*time* – момент времени, в который необходимо взять данные для обработки.

*size* – количество отсчетов данных для передачи. Параметр *size* может принимать значение от 1 до *BufferSize()*. *BufferSize()* – метод для определения размера.

*data* – указатель на массив данных пользователя, массив данных в плавающей запятой одинарной точности.

Возвращаемое значение

1 - данные содержат не менее одного значения NaN

0 – нормальное выполнение функции.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа ZETServer.ocx выгружена из памяти процессора.

-3 – параметр channel < 0.

-4 – параметр channel > (QuanChan() – 1).

-5 – заданный канал не является каналом модуля АЦП-ЦАП.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

© ООО "ЭТМС" 1992-2024. Все права защищены

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

***GetWatermarkFlag*** - Запрос наличия watermark по каналу

*long* ***GetWatermarkFlag****(LONG lChannel, LONG* pFlag)*

Параметры

*lChannel* – номер канала, данные по которому требуется запросить. Параметр *lChannel* может принимать значение от *0* до (*QuanChan() – 1)*.

*pFlag* - количество watermark.

Возвращаемое значение

1 - данные содержат не менее одного значения NaN

0 – нормальное выполнение функции.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа ZETServer.ocx выгружена из памяти процессора.

-3 – параметр *lChannel* < 0.

-4 – параметр *lChannel* > (QuanChan() – 1).

-5 – заданный канал не является каналом модуля АЦП-ЦАП.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

© ООО "ЭТМС" 1992-2024. Все права защищены

***PutDACDataByPointer*** - добавление данных в канал генератора по указателю буфера ЦАП. PointerDAC - указывается без преобразований с разрядностью.

*long* ***PutDACDataByPointer****(long Channel, long PointerDAC, long size, float* data);*

Возвращаемое значение

1 - данные содержат не менее одного значения NaN

0 – нормальное выполнение функции.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа ZETServer.ocx выгружена из памяти процессора.

-3 – параметр *Channel* < 0.

-4 – параметр *Channel* > (QuanChan() – 1).

-5 – заданный канал не является каналом модуля АЦП-ЦАП.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

***GetChannelInfoExt*** - возвращает структуру каналов с информацией об устройстве ADC_INFO_EXT.

*long* ***GetChannelInfoExt****(long channel, BYTE* pADCInfoExt, ULONG size);*

Возвращаемое значение

1 - данные содержат не менее одного значения NaN

0 – нормальное выполнение функции.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа ZETServer.ocx выгружена из памяти процессора.

-3 – параметр channel < 0.

-4 – параметр channel > (QuanChan() – 1).

-5 – заданный канал не является каналом модуля АЦП-ЦАП.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

© ООО "ЭТМС" 1992-2024. Все права защищены

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

***GetModulelInfoExt*** - возвращает структуру модуля с информацией об устройстве ADC_INFO_EXT.

*long* ***GetModulelInfoExt****(LONG Modul, BYTE* pADCInfoExt, ULONG size)*

Возвращаемое значение

1 - данные содержат не менее одного значения NaN

0 – нормальное выполнение функции.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа ZETServer.ocx выгружена из памяти процессора.

-3 – параметр channel < 0.

-4 – параметр channel > (QuanChan() – 1).

-5 – заданный канал не является каналом модуля АЦП-ЦАП.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

© ООО "ЭТМС" 1992-2024. Все права защищены

***PutDataByte*** - передать данные в ЦАП в виде массива байт, Alignment - на сколько выравниваются данные

*long* ***PutDataByte****(long channel, double time, long size, BYTE* data, long Alignment);*

Возвращаемое значение

1 - данные содержат не менее одного значения NaN

0 – нормальное выполнение функции.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа ZETServer.ocx выгружена из памяти процессора.

-3 – параметр channel < 0.

-4 – параметр channel > (QuanChan() – 1).

-5 – заданный канал не является каналом модуля АЦП-ЦАП.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

-20 - превышено время обработки функции.

***EthIP*** - *получение значения IP-адреса.*

*long* ***EthIP****(long channel, BYTE* IP1, BYTE* IP2, BYTE* IP3, BYTE* IP4)*

Параметры

*channel* – номер канала, принадлежащий тому модулю АЦП-ЦАП, количество работающих каналов которого требуется узнать. Параметр *channel* может принимать значение от *0* до (*QuanChan() – 1)*.

Возвращаемое значение

≥ 0 – суммарное количество работающих каналов на модуле АЦП-ЦАП, которому принадлежит заданный канал.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа ZETServer.ocx выгружена из памяти процессора.

-3 – параметр channel < 0.

© ООО "ЭТМС" 1992-2024. Все права защищены

-4 – параметр channel > (QuanChan() – 1).

-5 – заданный канал не является каналом модуля АЦП-ЦАП.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

***EthIPGUID*** - *получение значения IP-адреса по Guid.*

*long* ***EthIPGUID(GUID* GUIDchannel, BYTE* IP1, BYTE* IP2, BYTE* IP3, BYTE* IP4);***

Параметры

***GUIDchannel*** – номер канала, принадлежащий тому модулю АЦП-ЦАП, количество работающих каналов которого требуется узнать. Параметр ***GUIDchannel*** может принимать значение от *0* до (*QuanChan() – 1)*.

Возвращаемое значение

≥ 0 – суммарное количество работающих каналов на модуле АЦП-ЦАП, которому принадлежит заданный канал.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа ZETServer.ocx выгружена из памяти процессора.

-3 – параметр ***GUIDchannel*** < 0.

-4 – параметр ***GUIDchannel*** > (QuanChan() – 1).

-5 – заданный канал не является каналом модуля АЦП-ЦАП.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

© ООО "ЭТМС" 1992-2024. Все права защищены

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

***GUIDchannel*** - *получение способа подключения. 0-локальный или по Ethernet, 1-удаленное через клиент данных*

*long* ***RemoteDev****(LONG channel);*

Возвращаемое значение

≥ 0 – суммарное количество работающих каналов на модуле АЦП-ЦАП, которому принадлежит заданный канал.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа ZETServer.ocx выгружена из памяти процессора.

-3 – параметр channel < 0.

-4 – параметр channel > (QuanChan() – 1).

-5 – заданный канал не является каналом модуля АЦП-ЦАП.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

***RemoteDevGUID*** - *проверить способа подключения Guid. 0-локальный или по Ethernet, 1-удаленное через клиент данных*

*long* ***RemoteDevGUID****(GUID* GUIDchannel)*

Возвращаемое значение

≥ 0 – суммарное количество работающих каналов на модуле АЦП-ЦАП, которому принадлежит заданный канал.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа ZETServer.ocx выгружена из памяти процессора.

© ООО "ЭТМС" 1992-2024. Все права защищены

-3 – параметр *GUIDchannel* < 0.

-4 – параметр *GUIDchannel* > (QuanChan() – 1).

-5 – заданный канал не является каналом модуля АЦП-ЦАП.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

***VrtChnSetEthIP*** - *Установить IP виртуального канала.*

*long* ***VrtChnSetEthIP(LONG virtchannel, BYTE IP1, BYTE IP2, BYTE IP3, BYTE IP4)***

Параметры

***virtchannel*** – номер канала, принадлежащий тому модулю АЦП-ЦАП, количество работающих каналов которого требуется узнать. Параметр ***virtchannel*** может принимать значение от *0* до (*QuanChan() – 1)*.

Возвращаемое значение

≥ 0 – суммарное количество работающих каналов на модуле АЦП-ЦАП, которому принадлежит заданный канал.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа ZETServer.ocx выгружена из памяти процессора.

-3 – параметр ***virtchannel*** < 0.

-4 – параметр ***virtchannel*** > (QuanChan() – 1).

-5 – заданный канал не является каналом модуля АЦП-ЦАП.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

© ООО "ЭТМС" 1992-2024. Все права защищены

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

***PutDataGUID*** - *Передать данные в ЦАП или виртуальный канал по GUID*

*long* ***PutDataGUID(GUID* _channel, double time, long size, const float* data);***

Параметры

***_channel*** – номер канала, принадлежащий тому модулю АЦП-ЦАП, Передать данные в ЦАП или виртуальный канал по GUID. Параметр ***_channel*** может принимать значение от *0* до (*QuanChan() – 1)*.

Возвращаемое значение

≥ 0 – суммарное количество работающих каналов на модуле АЦП-ЦАП, которому принадлежит заданный канал.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа ZETServer.ocx выгружена из памяти процессора.

-3 – параметр ***_channel*** < 0.

-4 – параметр ***_channel*** > (QuanChan() – 1).

-5 – заданный канал не является каналом модуля АЦП-ЦАП.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

***GetQualityGUID*** - *Качество данных по каналу по его по GUID*

© ООО "ЭТМС" 1992-2024. Все права защищены

*long* ***GetQualityGUID(GUID* GUIDchannel);***

Параметры

***GUIDchannel*** – номер канала, принадлежащий тому модулю АЦП-ЦАП, Позволяет передать качество данных по каналу по его по GUID. Параметр ***GUIDchannel*** может принимать значение от *0* до (*QuanChan() – 1)*.

Возвращаемое значение

≥ 0 – суммарное количество работающих каналов на модуле АЦП-ЦАП, которому принадлежит заданный канал.

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа ZETServer.ocx выгружена из памяти процессора.

-3 – параметр ***GUIDchannel*** < 0.

-4 – параметр ***GUIDchannel*** > (QuanChan() – 1).

-5 – заданный канал не является каналом модуля АЦП-ЦАП.

-6 – серверу данных не хватает памяти для организации буферов данных.

-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().

-9 – номер декады decade < 0.

-10 – номер декады decade > 4.

-11 – момент времени time больше текущего времени для данного канала.

-12 – момент времени time < 0.

-13 – указатель на массив данных data = NULL.

-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".

-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().

-16 - пустой указатель на буфер.

***FullName*** - Позволяет передать полное имя канала для добавления объектов

*BSTR* ***FullName****(LONG channel);*

Параметры

*channel* – номер канала, принадлежащий тому модулю АЦП-ЦАП, Позволяет передать качество данных по каналу по его GUID. Параметр *channel* может принимать значение от *0* до (*QuanChan() – 1)*.

Возвращаемое значение

≥ 0 – суммарное количество работающих каналов на модуле АЦП-ЦАП, которому принадлежит заданный канал.

© ООО "ЭТМС" 1992-2024. Все права защищены

-1 – на момент вызова метода не было подключения к серверу данных.
-2 – программа ZETServer.ocx выгружена из памяти процессора.
-3 – параметр *channel* < 0.
-4 – параметр *channel* > (QuanChan() – 1).
-5 – заданный канал не является каналом модуля АЦП-ЦАП.
-6 – серверу данных не хватает памяти для организации буферов данных.
-7 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.
-8 – размер запрашиваемых данных Size ≤ 0 или size ≥ DecadeBufferSize().
-9 – номер декады decade < 0.
-10 – номер декады decade > 4.
-11 – момент времени time больше текущего времени для данного канала.
-12 – момент времени time < 0.
-13 – указатель на массив данных data = NULL.
-14 - в случаях, когда у сервера нет данных, относящихся к запрошенному времени. И когда время из "прошлого", и когда оно из "будущего".
-15 – момент времени time отстает от внутреннего времени, которое можно определить с помощью метода CurrentTime().
-16 - пустой указатель на буфер.

## 2.2.События в SRV.ocx

В процессе работы сервер данных может менять режимы работы. При существенных изменениях сервер рассылает события (*Events*) ко всем подключенным к серверу программам. Программы должны в соответствии с параметрами событий, также учитывать изменения, если это необходимо.

***Modify*** – событие, которое происходит при включении и отключении каналов АЦП, ЦАП, виртуальных каналов, изменении коэффициентов усиления и т.д.

*void **Modify** (long GlobalPar)*

Параметры

*GlobalPar* – параметр события. Если *GlobalPar* = 0, то произошло подключение каналов АЦП, виртуальных каналов. Если *GlobalPar* = 1, то произошло подключение каналов ЦАП. Если *GlobalPar* = 2, то произошло изменение информации по виртуальному каналу.

***FileMode*** – событие, которое происходит при смене режима работы сервера данных.

*void **FileMode** (long lParam)*

Параметры

© ООО "ЭТМС" 1992-2024. Все права защищены

*lParam* – параметр события. При переходе в режим реального времени приходит событие с параметром 0. При переходе в режим чтения данных из файла – 1. В режим «чтения данных из файла» сервер переходит при запуске программы «Воспроизведение сигналов» из группы «Регистратор». При выходе из программы «Воспроизведение сигналов» сервер переходит в режим работы реального времени. Текущее время сервера по всем каналам сбрасывается в 0.

***StartFile*** – событие, которое происходит при работе сервера в режиме чтения данных из файла.

*void* ***StartFile*** *(long lParam)*

Параметры

*lParam* – параметр события. При нажатии на кнопку «Начало воспроизведения сигналов» в программе «Воспроизведение сигналов» из группы «Регистратор» создается событие с параметром 0.

***SetTime*** **–** событие, которое происходит при работе сервера в режиме чтения данных из файла.

*void* ***SetTime*** *(float fParam)*

Параметры

*fParam* – параметр события. Сервер передает события по мере чтения данных из файла и при позиционировании по файлу. Параметр *fParam* равен секундам от начала файла.

# Глава 3.Перечень функций, используемые SRV.ocx в базовом классе

| **Методы.ocx** | |
|---|---|
| **Подключение и отключение** | |
| *long* ***Connect****(void)* | подключение к серверу данных для программ обработки данных от АЦП или файлов, при этом запускаются АЦП, если они не запущены. |
| *long* ***Disconnect****(void)* | отключение от сервера данных. |
| *long* ***ConnectDac****(long NumberDac)* | подключение к серверу данных для программ генерации сигналов для ЦАП и работы с данными |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| | от АЦП, при этом запускаются ЦАП, если они не запущены |
| *long* ***DisconnectDac****(long NumberDac)* | отключение от сервера данных в режиме работы с ЦАП. |
| *long* ***ConnectVrtCh****(long NumChan)* | подключение к серверу для программ, создающих дополнительные виртуальные каналы и работающих с данными от АЦП. |
| *long* ***DisconnectVrtCh****(void)* | отключение от сервера данных. |
| *long* ***BufferSize****(void)* | определяется размер буфера данных сервера по нулевой декаде. |
| *long* ***DecadeBufferSize****(long decade)* | определяется размер буфера сервера данных по декаде. |
| *long* ***Start****(void)* | запустить АЦП и ЦАП. |
| *long* ***Stop****(void)* | остановить АЦП и ЦАП. |
| *long* ***GetMode****(void)* | определяется режим работы сервера |
| **Опрос параметров каналов системы** | |
| *float* ***Frequency****(long channel)* | определяется частота дискретизации заданного канала. |
| *long* ***QuanChan****(void)* | определяется количество работающих каналов на сервере данных |
| *long* ***WorkChanADC****(long channel)* | определяется количество работающих каналов на модуле АЦП-ЦАП, которому принадлежит заданный канал |
| *long* ***MaxQuanChanDac****(void)* | определяется суммарное количество каналов ЦАП, которые можно задействовать в системе. |
| *CString* ***Commentary****(long channel)* | определяется имя (название) заданного канала. |
| *CString* ***Conversion****(long channel)* | определяется единица измерения по заданному каналу |
| *float* ***MaxLevel****(long channel)* | определяется максимально допустимый уровень по заданному каналу в единицах измерения (максимальный входной диапазон). |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *float **MinLevel**(long channel)* | определяется минимально различимый уровень по заданному каналу в единицах измерения (вес младшего разряда по каналу) |
| *float **Reference**(long channel)* | определяется значения опоры для расчета уровней в децибелах по заданному каналу. |
| *float **ShifLevel**(long channel)* | определяется смещение постоянной составляющей по заданному каналу в единицах измерения. |
| *float **Sense**(long channel)* | определяется чувствительность (коэффициента преобразования) заданного канала в вольтах на единицу измерения. |
| *CString **AFCH**(long channel)* | определяется название файла, в котором может храниться пользовательская информация о канале, например, частотно-зависимые поправки АЧХ тракта. |
| *float **CurLevel**(long channel)* | определяется нормирование текущего значения по заданному каналу (СКЗ) от 0 до 1 (для контроля перегрузки). |
| *long **GetStatus**(long channel)* | определяется тип заданного канала |
| double ***CurrentTime***(*long* channel) | определяется текущее время работы сервера в секундах |
| *long **GetChannelPar**(long channel, BYTE* pChannelPar, ULONG size);* | определяются структуры канала |
| *long **GetChannelPar2**(long channel, BYTE* pChannelPar, ULONG size);* | определяются структуры канала |
| *long **GetQuality**(long channel);* | определяется качество данных по каналу |
| *long **IsFeedBackChan**(LONG channel);* | определяется является ли канал каналом с обратной связью: 1 - является, 0 - не является. |
| *USHORT **Hpf**(LONG channel);* | определяется состояние работы ФВЧ 0.1 Гц по каналу |
| **Опрос физических параметров каналов** | |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *long **QuanDSP**(void)* | опрос количества устройств АЦП-ЦАП (сигнальных процессоров) в системе. |
| *long **QuanPhChan**(long NumDSP)* | опрос количества физических каналов устройства (сколько может быть включено каналов, а не сколько каналов фактически включено). |
| *long **TypeAdc**(long channel)* | опрос типа устройства (платы) АЦП-ЦАП, физическим каналом которого является заданный канал. |
| *long **NumDSP**(long channel)* | опрос номера сигнального процессора, физическим каналом которого является заданный канал. |
| *long **NumModule**(long channel)* | опрос номера модуля устройства (сигнального процессора) в многомодульной системе. |
| *long **NumPhChan**(long channel)* | опрос физического номера канала на модуле для заданного канала. |
| *Float **GetAttenuation**(long channel)* | опрос коэффициента затухания заданного канала ЦАП. |
| *float **Frequency** (long Channel)* | опрос частоты дискретизации АЦП по выбранному каналу. |
| *float **FrequencyDac** (long channelDac)* | опрос частоты дискретизации ЦАП по выбранному каналу. |
| *float **GetFrequency**(long Modul)* | опрос частоты дискретизации заданного устройства. |
| *float **GetNextFreq**(long Modul, long Par)* | опрос списка возможных частот дискретизации заданного устройства. |
| *float **GetAmplify**(long channel)* | опрос коэффициента усиления по заданному каналу. |
| *long **IdChan**(long channel, long * Id)* | опрос идентификатора заданного канала АЦП. |
| *long **IdChanDac**(long channel, long * Id)* | опрос идентификатора заданного канала ЦАП. |
| *long **IdChanVirt**(long channel, long * Id)* | опрос идентификатора заданного виртуального канала. |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| **Управление каналами АЦП** | |
| *float* ***SetFrequency****(long Modul, float freq)* | установка частоты дискретизации заданного устройства. |
| *long* ***OnOffChannel****(long Modul, long channel, long Par)* | установка состояния канала АЦП заданного устройства. |
| *long* ***SetSinDiffChannel****(long Modul, long channel, long Par)* | установка режима работы канала АЦП заданного устройства. |
| *float* ***GetNextFreq****(long Modul, long Par);* | Получение следующей частоты по списку |
| *long* ***IdChan****( long Channel, long * Id);* | определение расширенного идентификатора заданного канала АЦП. |
| *long* ***TypeAdc****(long channel);* | установка типа АЦП заданного устройства. |
| *long* ***WorkChanADC****(long channel)* | установка текущего канала АЦП. |
| *long* ***IdChanGUID****(long channel_, long* id_);* | определение идентификатора канала |
| *DOUBLE* ***GetTimeFromStartAdc****(void)* | Время с момента старта АЦП |
| **Управление каналами ЦАП** | |
| *float* ***FrequencyDac****(long channelDac);* | установка частоты ЦАП по выбранному каналу |
| *float* ***SetFrequency****(long Modul, float freq)* | установка частоты дискретизации заданного устройства АЦП и ЦАП. |
| *long* ***IsFreeChanDac****(long channel)* | определение свободного (незанятого другой программой) канала ЦАП. |
| *long* ***SetChanDac****(long channel)* | подключение заданного канала ЦАП. |
| *long* ***RemChanDac****(long channel)* | отключение заданного канала ЦАП. |
| *long* ***MaxQuanChanDac****(void)* | определение количества каналов ЦАП |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *float* ***GetAttenuation****(long channel)* | опрос коэффициента затухания заданного канала ЦАП. |
| *long* ***SetAttenuation****(long channel, float attenuator)* | установка коэффициента затухания заданного канала ЦАП. |
| *long* ***IdChanDac****(LONG Channel, LONG* Id);* | определение идентификатора канала ЦАП |
| **Управление виртуальными каналами** | |
| *long* ***VrtChnInfo****(long virtchannel, BSTR * comment, BSTR * Conversion, float freq, float MaxLevel, float MinLevel, float refer)* | задание информации для виртуального канала. |
| *long* ***SetGroupName****(long virtchannel, LPCTSTR GrName)* | устанавливает определение группового имени виртуального канала |
| *long* ***SetStatus****(long lVirtChannel, long lStatus)* | опрос статуса виртуального канала |
| *long* ***SetID****(long lVirtChannel, long lID)* | установка идентификатора виртуального канала |
| *long* ***SetID****(long virtChannel_, GUID id_)* | (не реализована) - установка расширенного идентификатора виртуального канала. |
| *long* ***AddVirtualChannel****(long lIndex, long lQuantity)* | добавление виртуальных каналов |
| *long* ***DeleteVirtualChannel****(long lIndex, long lQuantity)* | удаление виртуальных каналов |
| *long* ***IdChanVirt****(long Channel, long* Id);* | опрос расширенного идентификатора заданного виртуального канала. |
| *long* ***PutPulse****(long lChannel, double dTime, long lSize, float* pData)* | создание виртуального канала. |
| *long* ***GetPulse****(long lChannel, long lIndex, double* pTime, long* lSize, float* pData)* | опрос виртуальных каналов. |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *long* ***GetInfLast****(long lChannel, double* pTime, long* pSize, long* pIndex)* | опрос последнего индекса виртуального канала. |
| *long* ***GetInfIndex****(long lChannel, double* pTime, long* pSize, long lIndex)* | опрос индекса виртуального канала. |
| *long* ***SetIDGUID****(long virtChannel_, BSTR id_);* | установка идентификатора виртуального канала |
| *long* ***IdChanVirtGUID****(long Channel, long* Id_);* | опрос идентификатора виртуального канала |
| **Работа с цифровым портом** | |
| *long* ***SetDigOutEnable****(long Zmodule, long OutMask)* | установка маски вывода на выбранному устройстве. |
| *long* ***GetDigOutEnable****(long Zmodule)* | опрос маски вывода на выбранном устройстве. |
| *long* ***GetDigInput****(long Zmodule)* | опрос входных данных цифрового порта на выбранном устройстве. |
| *long* ***GetDigOutput****(long Zmodule)* | опрос выходных данных цифрового порта на выбранном устройстве. |
| *long* ***SetDigOutput****(long Zmodule, long DigOut)* | установка выходных данных цифрового порта на выбранном устройстве. |
| *long* ***GetDigBits****(long Modul)* | опрос количества бит цифрового порта на выбранном устройстве. |
| *long* ***SetBitDigOutput****(long Zmodule, long bit)* | установка состояния выходного бита на заданном устройстве в логическую единицу. |
| *long* ***ClrBitDigOutput(****long Zmodule, long bit)* | установка состояния выходного бита на заданном устройстве в логический ноль. |
| *long* ***Triger****(long Modul, long Parametr, long * ValInt, float * ValFloat)* | настраивает синхронизацию по цифровому порту. 1 - вкл./выкл. внешнего запуска, 2 - вкл./выкл. синхронизации по внешней частоте, 3 - установить значение внешней опорной частоты, 4- Разрешить генерацию сигналов синхронизации |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| | на цифровой порт, -1 - прочитать статус внешнего запуска, -2 - прочитать статус синхронизации по внешней частоте, -3 - прочитать значение текущей опорной частоты, -4 - Опрос разрешения генерации сигналов синхронизации на цифровой порт |
| **Работа по цифровому каналу** | |
| *long **SetDigitalChannelEvent**(long * guid_, long seconds, long nanoseconds, long front);* | Генерирование события изменения текущего состояния по цифровому каналу |
| *long **SetStartChannelEvent**(long * guid_, long seconds, long nanoseconds);* | Генерирование события начала передачи данных по каналу |
| *long **SetEndlChannelEvent**(long * guid_, long seconds, long nanoseconds);* | Генерирование события конца передачи данных по каналу |
| *long **SetProcessChannelEvent**(long guid_, long seconds, long nanoseconds, long process);* | Генерирование события текущего состояния передачи данных по каналу |
| *long **SetCurLevel**(long channel, float level)* | Генерирование события текущего уровня передачи данных по каналу |
| **Управление внутренним генератором** | |
| *long **IsHaveSineBuiltinGen**(long Modul);* | Проверка есть ли у канала встроеный генератор: 1 - есть, 0 - нету |
| *long **GetSineBuiltinGenParam**(long Modul, long* Enabled, DOUBLE* Freq, DOUBLE* Level, DOUBLE* Offset);* | Получение параметров встроеного генератора синуса по каналу |
| *long **SetSineBuiltinGenParam**(long Modul, long Enabled, DOUBLE* | Задание параметров встроеного генератора синуса по каналу |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *Freq, DOUBLE Level, DOUBLE Offset);* | |
| *long* ***SetEnableBuiltinGenerator****(long Modul, long Enable);* | Включение/отключение внутреннего генератора |
| *long* ***GetEnableBuiltinGenerator****(long Modul, long* Enable);* | Получение статуса встроеного генератора. Enabled : 1 - включен, 0 - отключен |
| *long* ***Get1PPSBuiltinGen****(long Modul, long* Enabled);* | Получение статуса PPS встроеного генератора. Enabled : 1 - включен, 0 - отключен |
| *long* ***Set1PPSBuiltinGen****(long Modul, long Enabled);* | Установка статуса PPS встроеного генератора. Enabled : 1 - включен, 0 - отключен |
| **Работа с ШИМ** | |
| *long* ***GetPWMEnable****(long Zmodule)* | опрос наличия ШИМ на цифровых выходах выбранного устройства |
| *long* ***SetFreqPWM****(long Zmodule, long Rate, long Period)* | установка частоты ШИМ. |
| *long* ***SetOnDutyPWM****(long Zmodule, long channel, long OnDutyPWM, long ShiftPWM)* | установка скважности и фазы ШИМ по заданному каналу выбранного устройства. |
| *long* ***RegulatorPWM****(long Zmodule, float * data, long size)* | установка произвольных данных для ШИМ. |
| *long* ***StartPWM****(long Zmodule, long channel)* | запуск ШИМ по заданному каналу выбранного устройства. |
| *long* ***StopPWM****(long Zmodule, long channel)* | останов ШИМ по заданному каналу выбранного устройства. |
| **Работа с ICP** | |
| *long* ***PrusEna****(long channel)* | опрос наличия усилителя заряда (ICP) для подключения вибродатчиков и микрофонов к заданному каналу. |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *long **GetICP**(long lModule, long lChannel)* | опрос состояния усилителя заряда (ICP) для подключения вибродатчиков и микрофонов к заданному каналу. |
| *long **SetICP**(long lModule, long lChannel, long lType)* | установка состояния усилителя заряда (ICP) для подключения вибродатчиков и микрофонов к заданному каналу. |
| **Работа с GPS** | |
| *long **PutDataGPS**(LPCTSTR nmea, long size)* | запись данных NMEA-потока в сервер данных. |
| *CString **GetDataGPS**(long maxsize, long * realsize)* | запрос данных NMEA-потока, которые записываются в сервер данных методом PutDataGPS(). |
| *long **SetShiftGPS**(float seconds)* | установка смещения времени компьютера относительно времени по GPS. |
| *float **GetShiftGPS**(void)* | запрос смещения времени компьютера относительно времени по GPS. |
| *long **enaGPSync**(long enable)* | установка возможности синхронизации устройств по GPS. |
| *long **advanceGPS**(long seconds)* | установка времени опережения на синхронизацию устройств по GPS (время через которое должна произойти синхронизация). |
| *long **satGPS**(long satinfo)* | установка количества спутников GPS, наблюдаемых на данный момент. |
| *long **IsFileGPS**()* | опрос источника данных NMEA-потока. |
| *long **SetStartUTCTime**(unsigned __int64 UtcTime)* | установка времени, в которое должна произойти синхронизация по GPS. |
| *long **TimeGPS**(unsigned __int64 UtcTime)* | показывает время, в которое произошла синхронизация. |
| **Прием и передача данных** | |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *double* ***CurrentTime****(long channel)* | определение текущего времени по заданному каналу. Сервер данных ведет учет текущего времени каждого канала. |
| *long* ***GetData****(long channel, long decade, double time, long size, float * data)* | запрос данных по заданному каналу в буфер данных пользователя. |
| *long* ***GetDataVar****(long channel, long decade, double time, long size, VARIANT * datvar)* | запрос данных по заданному каналу в буфер данных пользователя. (зарезервирована под будущее использование). |
| *long* ***GetDataNet****(long channel, long decade, double time, long size, long pData)* | запрос данных по заданному каналу в буфер данных пользователя в .net языках. |
| *long* ***PutData****(long channel, double time, long size, float * data)* | передача данных из буфера пользователя в буфер виртуального канала или буфер ЦАП. |
| *long* ***PutDataNet****(long channel, double time, long size, long pData)* | передача данных из буфера пользователя в буфер виртуального канала или буфер ЦАП в .net языках. |
| *long* ***NumFileUsed****(long NumFile)* | команда для сервера данных прочитать очередную порцию данных из файла данных при работе в режиме чтения файлов. |
| **Вспомогательные информационные функции** | |
| *void* ***AboutBox****(void)* | содержит информацию о программе |
| *long* ***OrderConnected****(void)* | определение номера копии программы, загруженной в память. |
| *double* ***GetStartTime****(void)* | определение времени старта сервера данных (по времени компьютера). |
| *double* ***WorkingTime****(void)* | определение времени работы сервера данных (по времени компьютера). |
| *CString* ***GroupName****(long channel)* | определение группового имени канала АЦП. |
| *long* ***SetGroupName****(long VirtChannel, BSTR GrName);* | установление группового имени канала АЦП. |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *CString* ***DacGroupName****(long dacchannel)* | определение группового имени канала ЦАП. |
| *long* ***PointAllInfo****(LPDISPATCH * point)* | содержит информацию о программе. |
| *long* ***NullPointer****(long channel, long decade)* | программа пользователя может отслеживать начальный указатель накопления буфера, и определять, сколько новых данных записалось в память после последнего обращения. |
| *long* ***Pointer****(long channel, long decade)* | программа пользователя может отслеживать текущий указатель накопления буфера, и определять, сколько новых данных записалось в память после последнего обращения. |
| *long* ***TalkAbout****(long Param, long Value)* | программа пользователя, подключенная через компонент Unit |
| *long* ***SetSrvWin****(LPUNKNOWN Hwnd)* | установление не известному каналу названия |
| *long* ***SetChannelStatus****(long channel, long status)* | установление статуса канала. |
| *long* ***GetStatus****(long channel);* | опрос статуса канала: 0 - АЦП, 1 - ЦАП, 2 - виртуальный, 3 - цифровой, 4 - отключенный, 5 - канал отключенного устройства, 6 - MODBUS |
| *long* ***GetPulse*** *(long channel, long index, double* time, long* size, float* data)* | опрос частоты опроса канала. |
| *long* ***GetDatawFreq****(long lChannel, float fFrequency, double dTime, long lSize, float* pData)* | опрос данных с другой частотой передискретизации |
| *long* ***GetWatermarkFlag****(long lChannel, long* pFlag)* | добавление флага watermark |
| *long* ***PutDACDataByPointer****(long Channel, long PointerDAC, long size, float* data);* | добавление данных в канал генератора по указателю буфера ЦАП. PointerDAC - указывается без преобразований с разрядностью |
| *long* ***GetChannelInfoExt****(long channel, BYTE* pADCInfoExt,* | возвращает структуру с информацией об устройстве ADC_INFO_EXT |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *ULONG size);* | |
| *long* ***PutDataByte****(long channel, double time, long size, BYTE* data, long Alignment);* | передать данные в ЦАП в виде массива байт, Alignment - на сколько выравниваются данные |
| *long* ***EthIP****(long channel, BYTE* IP1, BYTE* IP2, BYTE* IP3, BYTE* IP4)* | получение значения IP-адреса. |
| *long* ***EthIPGUID****(GUID* GUIDchannel, BYTE* IP1, BYTE* IP2, BYTE* IP3, BYTE* IP4);* | получение значения IP-адреса по Guid |
| *long* ***RemoteDev****(LONG channel);* | получение способа подключения. 0-локальный или по Ethernet, 1-удаленное через клиент данных |
| *long* ***RemoteDevGUID****(GUID* GUIDchannel)* | проверить способа подключения Guid. 0-локальный или по Ethernet, 1-удаленное через клиент данных |
| *long* ***VrtChnSetEthIP****(LONG virtchannel, BYTE IP1, BYTE IP2, BYTE IP3, BYTE IP4)* | установить IP виртуального канала. |
| *long* ***PutDataGUID(GUID* _channel, double time, long size, const float* data);*** | Передать данные в ЦАП или виртуальный канал по GUID |
| *long* ***GetQualityGUID(GUID* GUIDchannel);*** | Позволяет передать качество данных по каналу по его GUID |
| *BSTR* ***FullName****(LONG channel);* | Позволяет передать полное имя канала для добавления объектов |
| **События в SRV.ocx** | |
| *void* ***Modify*** *(long GlobalPar)* | событие, которое происходит при включении и отключении каналов АЦП, ЦАП, виртуальных каналов, изменении коэффициентов усиления и т.д. |
| *void* ***FileMode*** *(long lParam)* | событие, которое происходит при смене режима работы сервера данных. |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *void* ***StartFile*** *(long lParam)* | событие, которое происходит при работе сервера в режиме чтения данных из файла. |
| *void* ***SetTime*** *(float fParam)* | событие, которое происходит при работе сервера в режиме чтения данных из файла. |

# Глава 4.Сообщения программы из Журнала ошибок и ZETServer

Программа имеет возможность работы без участия оператора, поэтому свои сообщения программа выдаёт не в виде диалоговых окон, а записывает их в системный журнал приложений, для просмотра которого можно использовать программу **Журнал ошибок ZETLAB** из вкладки **Сервисные** панели управления ZETLAB.
Формат сообщений, записываемых программами ПО ZETLAB в журнал, следующий:
"Имя программы №xx. Текст сообщения",
где xx - номер запущенной копии программы.
В системный журнал программа записывает не только сообщения об ошибках, но также и сообщения об изменениях своих параметрах. Записанные сообщения позволяют восстановить последовательность действий программы, что часто бывает полезным при анализе возникающих во время работы программы ошибок. В таблице ниже приведены сообщения программы.

| Текст сообщения | **Категория** |
|---|---|
| **Сообщения из Журнала ошибок ZETLAB** | |
| 101 Ошибка при подключении к серверу данных | ошибка |
| 102 Ошибка при чтении данных из реестра | ошибка |
| 103 Файл конфигурации в папке DirConfig недоступен | ошибка |
| 104 Файл справки отсутствует | ошибка |
| 105 Папка DirHelp недоступна | ошибка |
| 106 Папка DirSignal недоступна | ошибка |
| 107 Папка DirResult недоступна | ошибка |
| 108 Папка DirCorrect недоступна | ошибка |
| 109 Папка InstallLocation недоступна | ошибка |
| 110 Ошибка при создании экземпляра класса CAutoScaleXY | ошибка |
| 111 Ошибка при подключении к Unit | ошибка |
| 112 Программа запущена через Unit | ошибка |
| 113 Программа запущена из Z-панели | ошибка |
| 114 Программа начала работать | ошибка |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| 115 Частота дискретизации АЦП Гц | ошибка |
| 116 Код ошибки | ошибка |
| 117 Нет рабочих каналов сервера данных. Программа не будет загружена | ошибка |
| 118 Ошибка при вызове справки программы | ошибка |
| 119 Программа завершила свою работу | ошибка |
| 120 При чтении данных с канала произошла ошибка | ошибка |
| 121 При обработке данных произошла ошибка | ошибка |
| 122 Нет сервера данных | ошибка |
| 123 Программа будет закрыта | ошибка |
| 124 Ошибка при запуске программы обработки параметров | ошибка |
| **Сообщения из ZETServer** | |
| 1. Не было подключения к серверу | ошибка |
| 2. Сервер не загружен на компьютере | ошибка |
| 3. К серверу подключено слишком много программ | ошибка |
| 4. На компьютере не загружается сервер | ошибка |
| 5. Мало оперативной памяти или дискового пространства | ошибка |
| 6. Номер канала меньше нуля | ошибка |
| 7. Номер канала больше максимально возможного значения | ошибка |
| 8. Невозможно создать экземпляр компонента Server | ошибка |

При работе с запущенной панелью управления сообщения об ошибках программ ZETLAB дублируются временными всплывающими текстами в системном трее (область уведомлений - элемент панели инструментов рабочего стола или "панель задач" в Windows, используется для нужд постоянно используемых программ).
Получение сообщения "Сервер данных имеет слишком много каналов. Не достаточно памяти для работы программы в данном режиме. Программа будет закрыта" говорит о том, что в данный момент времени загружено слишком много программ, работающих с сервером данных ZETLAB, либо о том, что используемый компьютер имеет недостаточный объём ОЗУ. В первом случае следует закрыть неиспользуемые программы и перезапустить программу. Во втором случае необходимо либо заметь компьютер, либо в используемом увеличить объём ОЗУ.

© ООО "ЭТМС" 1992-2024. Все права защищены

# ZetWindowsLog.ocx Позволяет заполнять стандартный журнал ошибок

**Назначение**

Компонент **ZetWindowsLog.ocx** позволяет в процессе функционирования приложения, в котором он установлен, заполнять стандартный журнал ошибок Windows при возникновении нештатных ситуаций, что упрощает отладку программ и их дальнейшую поддержку.

## Глава 1.Установка компонента

Для работы с модулем ZetWindowsLog его необходимо сначала установить в программу. При загрузке программы необходимо загрузить с помощью компонента необходимую программу, разместить окно программы на экране в удобном месте, установить необходимые параметры в программе и затем по событиям, происходящим от программы считывать данные из программы.

Для установки компонента *ZetWindowsLog* в проект *Microsoft Visual Studio 2010*, необходимо кликнуть правой кнопкой мыши на форме диалога и из появившегося меню выбрить пункт *Insert ActiveX Control*...(рисунок 1.1)

© ООО "ЭТМС" 1992-2024. Все права защищены

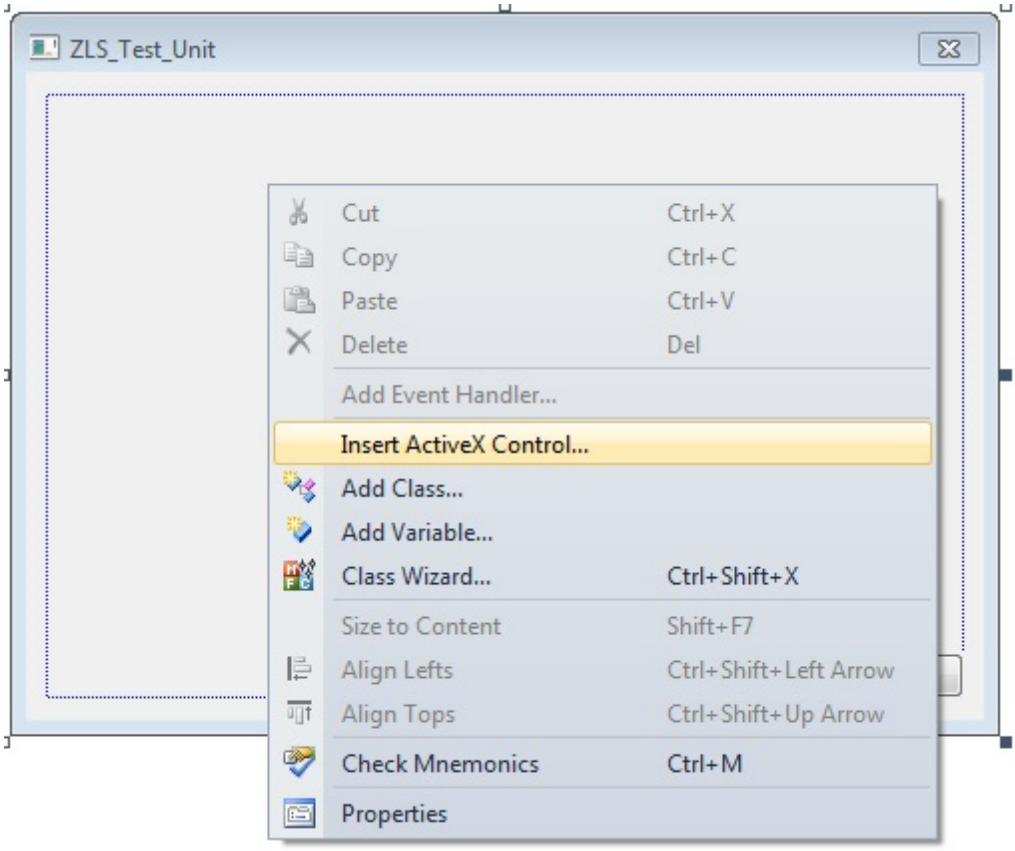



Рисунок 1.1

Из появившегося диалогового окна следует выбрать компонент *ZetWindowsLog Control* и нажать *OK* (рисунок 1.2).

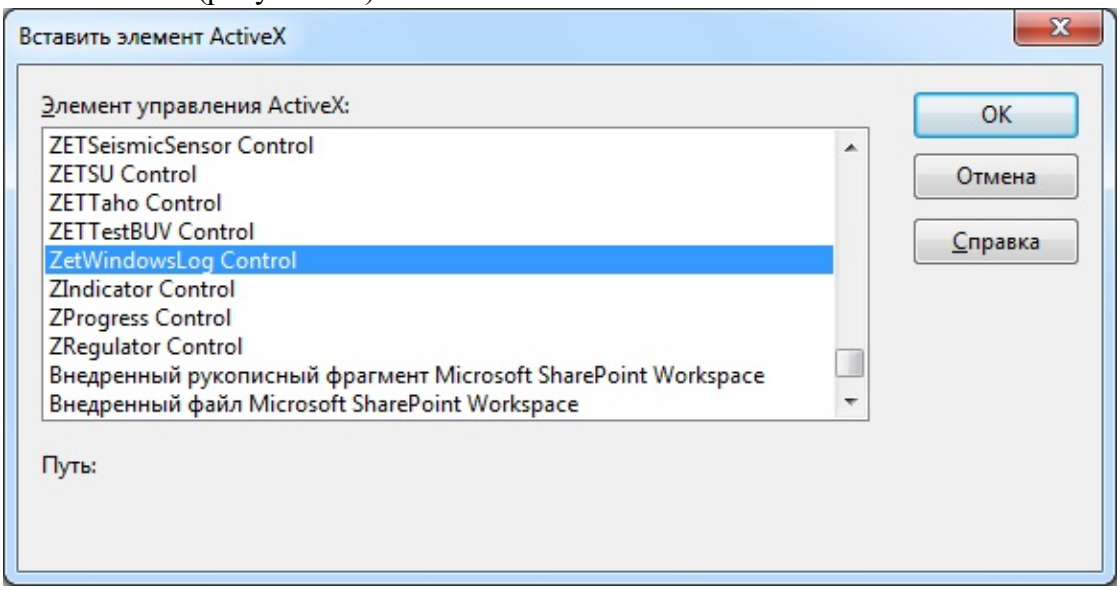



Рисунок 1.2

© ООО "ЭТМС" 1992-2024. Все права защищены

После этого компонент ZetWindowsLog*.ocx* появится на форме диалога (рисунок 1.3).

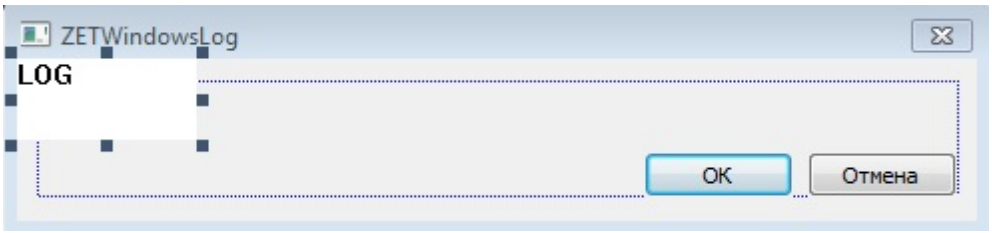



Рисунок 1.3

Одна установленная на форме компонента позволяет управлять одной программой. При необходимости управлять несколькими программами, необходимо установить на форму несколько компонент. Для того чтобы компонента не была видна на форме во время выполнения программы, необходимо установить свойство *Visible* равным *False*.

# Глава 2.Описание свойств

## Свойства

*void* ***AddMessage****(LONG eventType, LONG eventID, BSTR message)* - добавление сообщения в журнал событий.

Для посылки сообщений используется метод AddMessage(LONG eventType, LONG eventID, LPCTSTR message):

eventType - тип сообщения:

(0x00) EVENTLOG_SUCCESS - успешное выполнение операции

(0x01) EVENTLOG_ERROR_TYPE - сообщение об ошибке

(0x02) EVENTLOG_WARNING_TYPE - предупреждение

(0x04) EVENTLOG_INFORMATION_TYPE - информационное сообщение

(0x08) EVENTLOG_AUDIT_SUCCESS - успешная авторизация

(0x10) EVENTLOG_AUDIT_FAILURE - сбой авторизации

eventID - идентификатор сообщения:

(0x04) CONFIG_ERROR – ошибка конфигурации, неверно заданы параметры

(0x08) NOT_CONNECTED – модуль (может быть как программный, так и аппаратный), обеспечивающий наличие информации не подключен

(0x0C) DEVICE_FAILURE – испорчен модуль, обеспечивающий наличие информации

(0x10) SENSOR_FAILURE – испорчен источник информации

(0x14) LAST_KNOWN – связь с модулем, обеспечивающим наличие информации, потеряна

(0x18) COMM_FAILURE – не удалось связаться с модулем, обеспечивающим наличие информации

© ООО "ЭТМС" 1992-2024. Все права защищены

(0x1C) OUT_OF_SERVICE – не поддерживается (версией программы, версией ОС и т.д.)

(0x44) LAST_USABLE

(0x50) SENSOR_NOT_ACCURATE – значение искусственно подогнано

(0x54) EGU_ACCEEDED – значение вышло за пределы

(0x58) SUB_NORMAL

(0xD8) – хорошо

message - текст записи

*void* ***Read****(void)*- чтение информации из журнала событий.

*void* ***SetResultDir****(BSTR dirResult)* - установка пути для извлечения журнала событий.

*BSTR* ***LocalString****(LOCALSTRINGNUMBER number)* - запрос сообщения для базового класса.

© ООО "ЭТМС" 1992-2024. Все права защищены

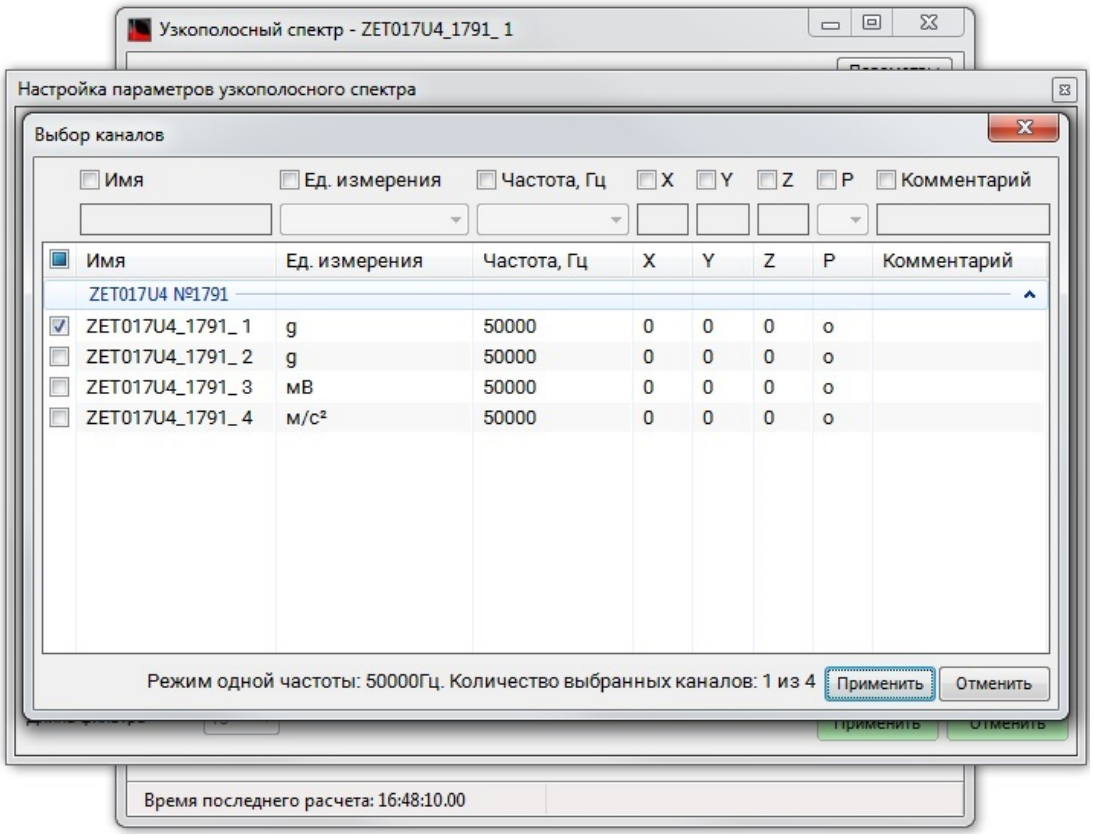
# ListMultiChannels.ocx предназначен для списка с использованием группового имени и фильтра каналов

**Назначение**

ListMultiChannels.ocx предназначен для списка с использованием группового имени и фильтра каналов

**Использование**

В Узкополосном спектре и в других программах.




© ООО "ЭТМС" 1992-2024. Все права защищены

# Глава 1.Описание свойств

## Свойства

LONG Mode- предназначен выбора режим задания каналов.

VARIANT_BOOL FilterByName_OnOff - предназначен для использования фильтра задания каналов по названию.

VARIANT_BOOL FilterByUnit_OnOff - предназначен для фильтра задания каналов по частоте дискретизаци

VARIANT_BOOL FilterByCoordinateX_OnOff - предназначен для фильтра задания каналов по координенте X.

VARIANT_BOOL FilterByCoordinateY_OnOff - предназначен для фильтра задания каналов по координенте Y.

VARIANT_BOOL FilterByCoordinateZ_OnOff - предназначен для фильтра задания каналов по координенте Z.

VARIANT_BOOL FilterByPosition_OnOff - предназначен для фильтра задания каналов по типу позиции.

VARIANT_BOOL FilterByComment_OnOff - предназначен для фильтра задания каналов по комментарию.

BSTR FilterByName_Value - предназначен для значение фильтра задания каналов по названию.

VARIANT_BOOL FilterByUnit_Type - предназначен для типа фильтра задания каналов по единицам измерения (по названию или по типу)").

BSTR FilterByUnit_ValueName - предназначен для названия единица измерения фильтра задания каналов по единицам измерения.

ULONGLONG FilterByUnit_ValueType - предназначен для типа единица измерения фильтра задания каналов по единицам измерения.

FLOAT FilterByFreqADC_Value - предназначен для ввода значения фильтра задания каналов по частоте дискретизации.

FLOAT FilterByCoordinateX_Value - предназначен для ввода значения фильтра задания каналов по координенте X.

© ООО "ЭТМС" 1992-2024. Все права защищены

FLOAT FilterByCoordinateY_Value - предназначен для ввода значения фильтра задания каналов по координенте Y.

FLOAT FilterByCoordinateZ_Value - предназначен для ввода значения фильтра задания каналов по координенте Z.

LONG FilterByPosition_Value - предназначен для ввода значения фильтра задания каналов по типу позиции.

BSTR FilterByComment_Value - предназначен для ввода значения фильтра задания каналов по комментарию.

VARIANT_BOOL ViewTableOcx_OnOff - предназначен для отображения таблицы компонента.

## Методы

long **SetChannels**(LONG size, LONG* data) - предназначен для первоначального задания каналов по индексам.

LONG **SetChannelAcceptableID**(LONG size, LONG* data) - предназначен для задания допустимых каналов по GUID (тип фильтра 1 только отображает, 0 скрывает).

LONG **SetChannelExcludeID**(LONG size, LONG* data) - предназначен для задания нежелательных каналов по GUID (тип фильтра 1 только отображает, 0 скрывает).

void **ClearFilter**() - предназначен для очистки фильтра.

void **ClearChannels**() - предназначен для очистки каналов.

long **GetConfig**(TCHAR* ptr) - предназначен для выдачи в программу данных с текущими параметрами модального окна.

long **SetConfig**(TCHAR* ptr) - предназначен для получения из программы данных с прежними параметрами модального окна.

void **SetSRV**(BYTE* pointer) - предназначен для задания внешнего CDSRV

void **UpDataChannels**()B - предназначен для выдачи составе каналов ZetServer произошли изменения.

void **SetTitle**(BSTR pText) - предназначен для задания заголовка модальному окну компонента.

ULONG **GetNumberChanSelect**() - предназначен для получения текущего кол-ва выбранных каналов.

© ООО "ЭТМС" 1992-2024. Все права защищены

void **StartChanSelection**() - предназначен для старта выбора каналов.

void **EndChanSelectionByNotApply**() - предназначен для завершения выбора каналов без применения выбранных каналов.

void **EnableNoChanSelection**() - предназначен для разрешения выбора пустого списка каналов.

© ООО "ЭТМС" 1992-2024. Все права защищены

# ZETAnpRecorder.ocx позволяет сохранять результаты в бинарном виде

**Назначение**

Модуль предназначен для использования в программах, которые сохраняют результаты в бинарном виде и позволяет:

- создавать заголовочные *.anp файлы;
- создавать бинарные файлы данных и добавлять в них значения.

## Глава 1.Установка компонента

Для работы с модулем ZETAnpRecorder его необходимо сначала установить в программу. При загрузке программы необходимо загрузить с помощью компонента необходимую программу, разместить окно программы на экране в удобном месте, установить необходимые параметры в программе и затем по событиям, происходящим от программы считывать данные из программы.

Для установки компонента ZETAnpRecorder.ocx в проект *Microsoft Visual Studio 2010*, необходимо кликнуть правой кнопкой мыши на форме диалога и из появившегося меню выбрить пункт *Insert ActiveX Control*...(рисунок 1.1)

© ООО "ЭТМС" 1992-2024. Все права защищены

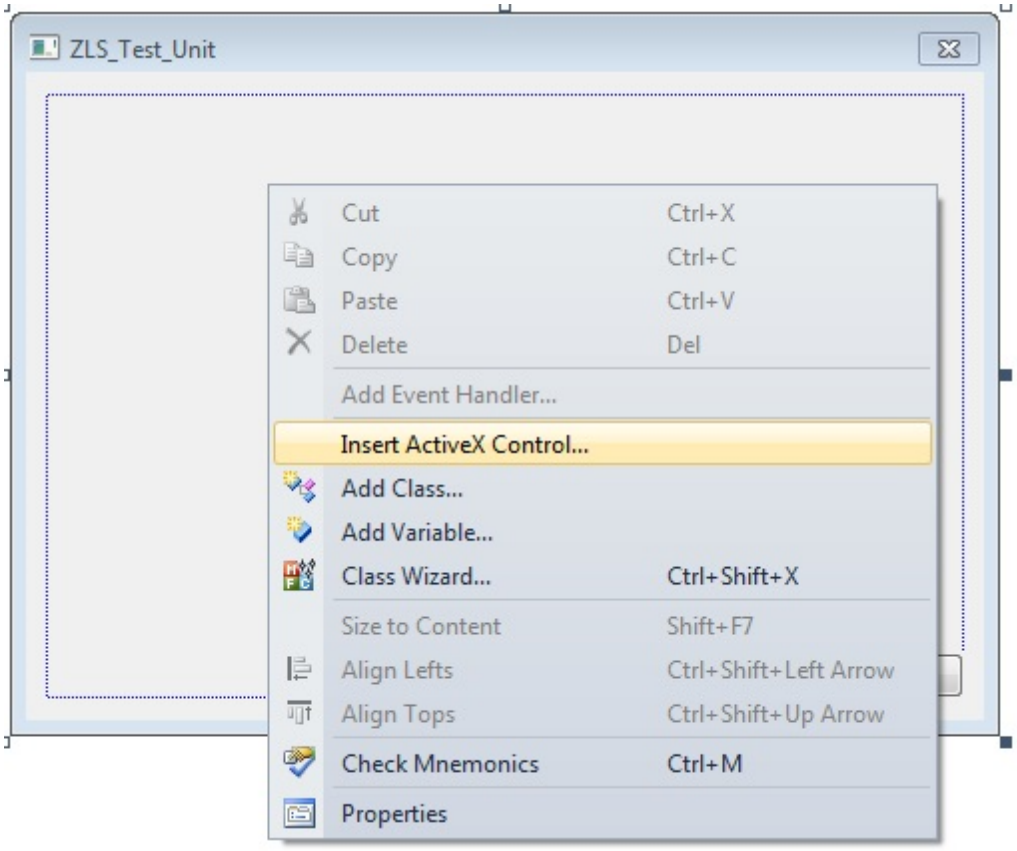



Рисунок 1.1

Из появившегося диалогового окна следует выбрать компонент *ZETAnpRecorder Control* и нажать *OK* (рисунок 1.2).

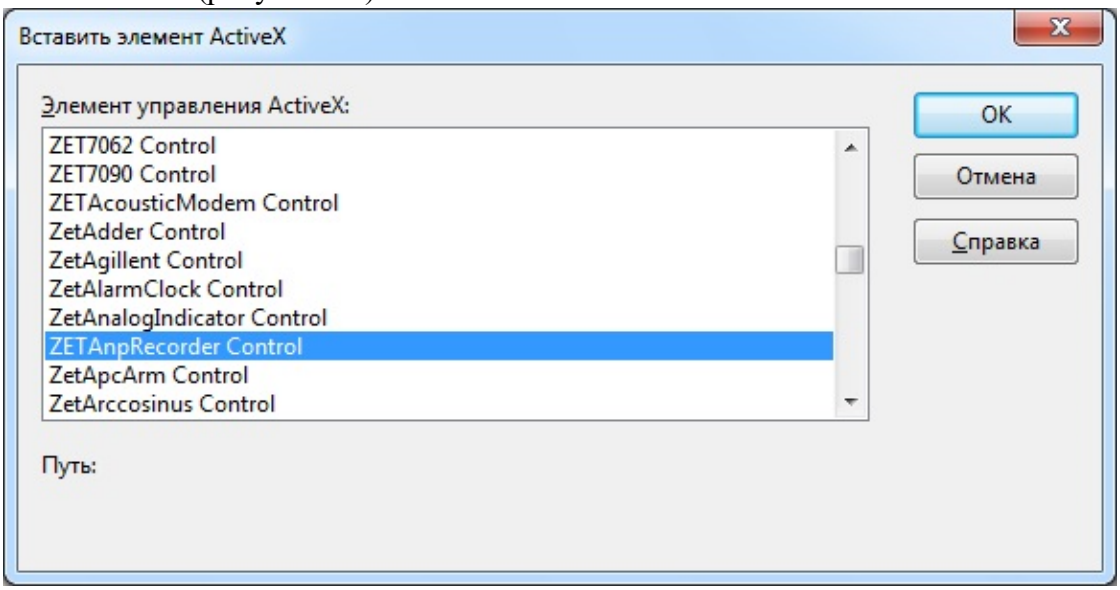



Рисунок 1.2

© ООО "ЭТМС" 1992-2024. Все права защищены

После этого компонент *ZETAnpRecorder.ocx* появится на форме диалога (рисунок 1.3).

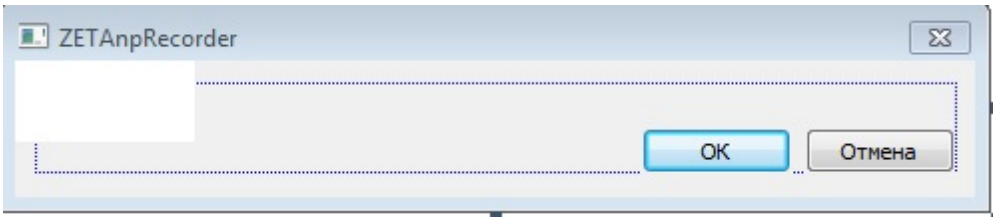



Рисунок 1.3

Одна установленная на форме компонента позволяет управлять одной программой. При необходимости управлять несколькими программами, необходимо установить на форму несколько компонент. Для того чтобы компонента не была видна на форме во время выполнения программы, необходимо установить свойство *Visible* равным *False*.

# Глава 2.Описание свойств

## Свойства

***GetFileName*** – запрос имени файла, с которым производится работа.

*BSTR* ***GetFileName*** *(void)*

Возвращаемое значение

строка – имя файла, с которым производится работа.

***SetFileName*** **–** установка имени файла для работы.

*void* ***SetFileName****(LPCTSTR newVal)*

Параметры

newVal – полный, абсолютный путь файла. Расширение *.ana / *.anp указывать не обязательно.

***RecAnpLpctstr*** **–** запись заголовочного *.anp файла с указанными значениями (формат *wchar*).

*bool* ***RecAnpLpctstr*** *(LPCTSTR COMMENT, LPCTSTR GAIN, LPCTSTR ABSVOLT, LPCTSTR FRQ, LPCTSTR TMI, LPCTSTR FRL, LPCTSTR FRH, LPCTSTR FORMAT, LPCTSTR START, LPCTSTR DATE, LPCTSTR CHANNEL, LPCTSTR TypeAdc, LPCTSTR NumberAdc, LPCTSTR MAXLEVEL, LPCTSTR SENSE, LPCTSTR CONVERT, LPCTSTR AMPL, LPCTSTR REFER, LPCTSTR AFCH, LPCTSTR DC)*

Параметры

*COMMENT* – название канала.
*GAIN* – коэффициент усиления по каналу.
*ABSVOLT* – мультипликатор канала.
*FRQ* – частота дискретизации канала.

© ООО "ЭТМС" 1992-2024. Все права защищены

*TMI* – не используется, зарезервирован под будущее использование.
*FRL* – нижняя частота полосового фильтра сигнала.
*FRH* – верхняя частота полосового фильтра сигнала.
*FORMAT* – формат представления данных.
*START* – время начала записи.
*DATE* – дата начала записи.
*CHANNEL* – номер канала в устройстве.
*TypeAdc* – тип устройства.
*NumberAdc* – номер сигнального процессора.
*Serial* – серийный номер устройства.
*MAXLEVEL* – максимальное значение сигнала.
*SENSE* – чувствительность по каналу.
*CONVERT* – единица измерения по каналу.
*AMPL* – коэффициент усиления внешнего усилителя.
*REFER* – опорное значение для вычисления децибел.
*AFCH* – файл поправки АЧХ.
*DC* – смещение постоянной составляющей.

Возвращаемое значение

true – удачное открытие файла.
false – неудачное открытие файла.

***RecAnpChar –*** запись заголовочного *.anp файла с указанными значениями (формат *char*).

*bool* ***RecAnpChar*** *(CHAR* COMMENT, CHAR* GAIN, CHAR* ABSVOLT, CHAR* FRQ, CHAR* TMI, CHAR* FRL, CHAR* FRH, CHAR* FORMAT, CHAR* START, CHAR* DATE, CHAR* CHANNEL, CHAR* TypeAdc, CHAR* NumberAdc, CHAR* MAXLEVEL, CHAR* SENSE, CHAR* CONVERT, CHAR* AMPL, CHAR* REFER, CHAR* AFCH, CHAR* DC)*

Параметры

*COMMENT* – название канала.
*GAIN* – коэффициент усиления по каналу.
*ABSVOLT* – мультипликатор канала.
*FRQ* – частота дискретизации канала.
*TMI* – не используется, зарезервирован под будущее использование.
*FRL* – нижняя частота полосового фильтра сигнала.
*FRH* – верхняя частота полосового фильтра сигнала.
*FORMAT* – формат представления данных.
*START* – время начала записи.
*DATE* – дата начала записи.
*CHANNEL* – номер канала в устройстве.
*TypeAdc* – тип устройства.
*NumberAdc* – номер сигнального процессора.
*Serial* – серийный номер устройства.
*MAXLEVEL* – максимальное значение сигнала.

© ООО "ЭТМС" 1992-2024. Все права защищены

*SENSE* – чувствительность по каналу.
*CONVERT* – единица измерения по каналу.
*AMPL* – коэффициент усиления внешнего усилителя.
*REFER* – опорное значение для вычисления децибел.
*AFCH* – файл поправки АЧХ.
*DC* – смещение постоянной составляющей.

Возвращаемое значение
true – удачное открытие файла.
false – неудачное открытие файла.

***PutValueToFile_short*** – запись в файл одного значения типа *short*. Если файл не существует, то создается новый файл, в противном случае добавляется значение в конец файла.

*bool* ***PutValueToFile_short****(SHORT shortVal)*

Параметры
*shortVal* – значение для записи в файл.

Возвращаемое значение
true – удачная запись требуемого количества байт.
false – неудачная запись требуемого количества байт.

***PutValueToFile_long*** – запись в файл одного значения типа *long*. Если файл не существует, то создается новый файл, в противном случае добавляется значение в конец файла.

*bool* ***PutValueToFile_long****(LONG longVal)*

Параметры
*longVal* – значение для записи в файл.

Возвращаемое значение
true – удачная запись требуемого количества байт.
false – неудачная запись требуемого количества байт.

***PutValueToFile_float*** – запись в файл одного значения типа *float*. Если файл не существует, то создается новый файл, в противном случае добавляется значение в конец файла.

*bool* ***PutValueToFile_float****(FLOAT floatVal)*

Параметры
*floatVal* – значение для записи в файл.

Возвращаемое значение
true – удачная запись требуемого количества байт.
false – неудачная запись требуемого количества байт.

© ООО "ЭТМС" 1992-2024. Все права защищены

***PutValueToFile_double*** – запись в файл одного значения типа *double*. Если файл не существует, то создается новый файл, в противном случае добавляется значение в конец файла.

*bool* ***PutValueToFile_double****(DOUBLE doubleVal)*

Параметры

*doubleVal* – значение для записи в файл.

Возвращаемое значение

true – удачная запись требуемого количества байт.
false – неудачная запись требуемого количества байт.

© ООО "ЭТМС" 1992-2024. Все права защищены

# ZETChanSelector.ocx предназначен для выбора каналов

Компонент **ZETChanSelector.ocx** используется для выбора каналов. Применяется практически во всех программах из состава ZETLab, например, в программах вольтметров, частотомера, фазометра и т.д. Компактные размеры и большая информативность данного компонента делают очень удобным его использование в таких программах как Многоканальный осциллограф или Самописец сигналов, где необходимо выбирать необходимые каналы, поступающих с модулей АЦП ЦАП или анализаторов спектра.

## Глава 1.Установка компонента ZETChanSelector.ocx

Для работы с модулем ZETChanSelector его необходимо сначала установить в программу. При загрузке программы необходимо загрузить с помощью компонента необходимую программу, разместить окно программы на экране в удобном месте, установить необходимые параметры в программе и затем по событиям, происходящим от программы считывать данные из программы.

Для установки компонента ZETChanSelector в проект *Microsoft Visual Studio 2010*, необходимо кликнуть правой кнопкой мыши на форме диалога и из появившегося меню выбрить пункт *Insert ActiveX Control*...(рисунок 1.1)

© ООО "ЭТМС" 1992-2024. Все права защищены

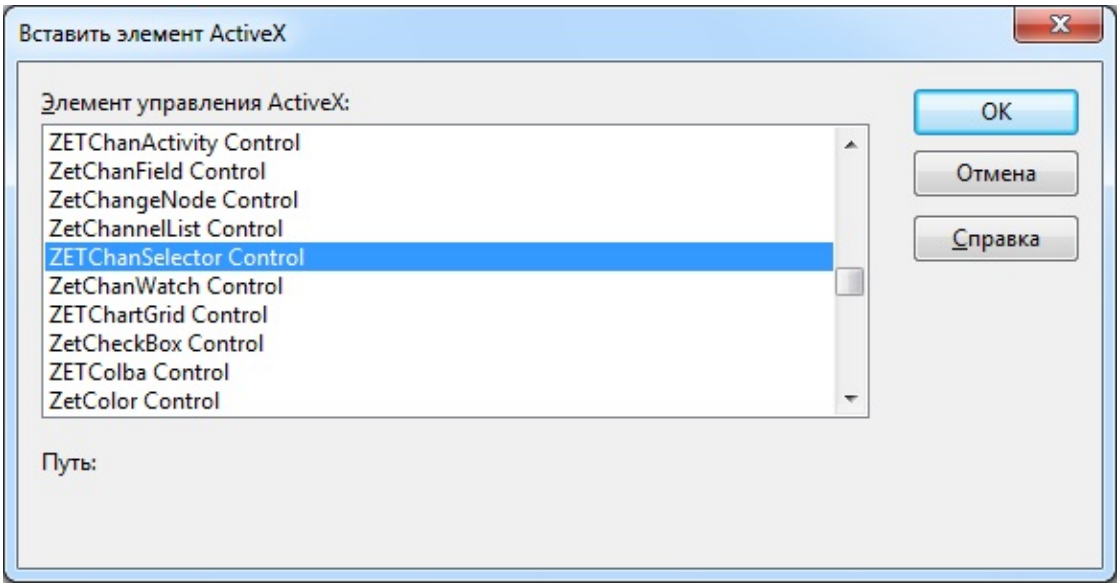



Рисунок 1.1

Из появившегося диалогового окна следует выбрать компонент *ZETChanSelector Control* и нажать *OK* (рисунок 1.2). После этого компонент ZETChanSelector.*ocx* появится на форме диалога (рисунок 1.2).

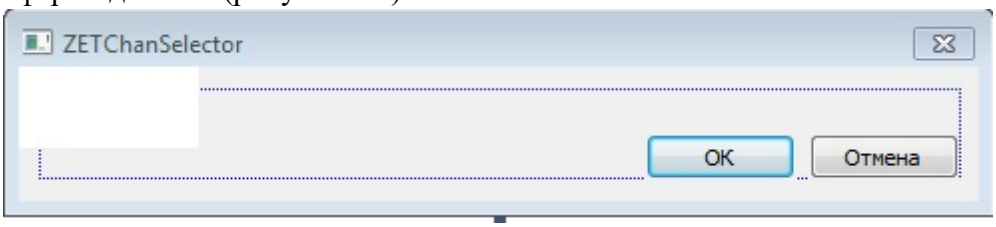
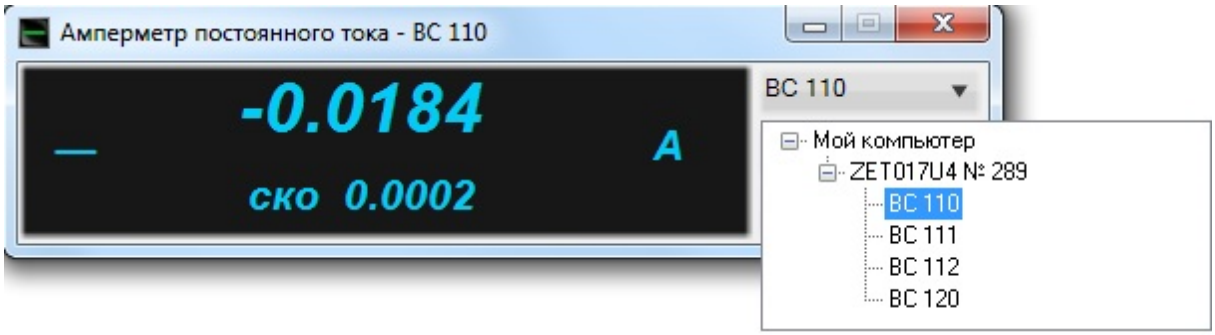



Рисунок 1.2

Одна установленная на форме компонента позволяет управлять одной программой. При необходимости управлять несколькими программами, необходимо установить на форму несколько компонент. Для того чтобы компонента не была видна на форме во время выполнения программы, необходимо установить свойство *Visible* равным *False*.

# Глава 2.Интерфейс компонента ZETChanSelector.ocx

© ООО "ЭТМС" 1992-2024. Все права защищены




При клике левой кнопкой "мыши" по верхнему элементу появляется выпадающий перечень доступных каналов сервера ZETLAB (ZETChanSelector.ocx), пример которого представлен на рисунке выше. Выбор канала осуществляется путём клика левой кнопки "мыши" на требуемом канале. После этого перечень каналов исчезает, в заголовке окна появляется название выбранного канала, правый индикатор отображает размерность нового канала, а также возможна замена заголовка программы в соответствии с таблицей, приведенной ниже.

Строка заголовка программы содержит сам заголовок и имя измеряемого канала. Заголовок изменяется в зависимости от размерности выбранного канала для изменений. В таблице ниже представлены все возможные варианты заголовков программы.

| Заголовок | Размерность канала |
|---|---|
| Вольтметр постоянного тока | В, мВ или мкВ |
| Амперметр постоянного тока | А, мА или мкА |
| Температура | град или С |
| Измеритель постоянного значения | остальные случаи |

# Глава 3.Полное описание свойств, методов и событий ZETChanSelector.ocx

Следует отметить, что компонент автомасштабируется при изменении его размеров, поэтому необходимо следить за тем, чтобы высота компонента в приложении была не менее 80 пикселей, иначе он может отображаться некорректно.

## 3.1.Свойства ZETChanSelector.ocx

Свойства можно задавать на этапе проектирования и во время выполнения программы. На этапе проектирования при выборе компонента и при нажатии клавиши <F4> появляется окно свойств. Эти свойства можно устанавливать на этапе проектирования программы.

© ООО "ЭТМС" 1992-2024. Все права защищены

## Цвета

Цвета задаются при помощи макроса *RGB(RED, GREEN, BLUE)*. Значения *RED*, *GREEN*, *BLUE* задают интенсивности цветов красного, зеленого и синего. Интенсивности изменяются в пределах от 0 до 255.

*OLE_COLOR* ***backColor*** – цвет фона

*OLE_COLOR* ***ColorText*** – цвет текста

Например, для изменении фона графика с желтого на зеленый, могут быть записаны числа:

```
OLE_COLOR a = 0xFF00;
OLE_COLOR b = 0xFFFF;
```

## Отображение границ

*VARIANT_BOOL* ***LeftBound*** – граница слева.

*VARIANT_BOOL* ***RightBound*** – граница справа.

*VARIANT_BOOL* ***TopBound*** – верхняя граница.

*VARIANT_BOOL* ***BottomBound*** – нижняя граница.

Изменение видимости отображения границ :
true - Видимость отображения границ.
false - Видимость отображения границ - выключена.

## Параметры

*long* ***Channel*** – порядковый номер канала.

*long* ***Type*** - тип списка.

*VARIANT_BOOL* ***Filter*** - установка фильтра по каналам (true(1) - включен, false(0) - выключен).

*BSTR* ***ConversionFilter*** – установка фильтра по единицам измерения

*BSTR* ***ToolTipText*** - текст подсказки

© ООО "ЭТМС" 1992-2024. Все права защищены

*VARIANT_BOOL* ***ZETLabStyle*** – установка стиля (true(1) - включен, false(0) - выключен).

*BSTR* ***TextString***- текстовая строка

*BSTR* ***Caption*** – название компонента

## 3.2.Методы ZETChanSelector.ocx

Методы можно использовать только во время выполнения программы.

*void* ***AboutBox****()* –содержит информацию о программе.

## 3.3.События от компоненты ZETChanSelector.ocx

*void* ***ModifyChannel****(LONG channelNum, LONG type)* - изменяет канал

# Глава 4.Краткий перечень функций, используемые ZETChanSelector.ocx

| **Свойства ZETChanSelector.ocx** | |
|---|---|
| **Цвета** | |
| *long* ***GetbackColor*** *()* | опрашивается цвет фона |
| *void* ***SetbackColor*** *(unsigned long propVal)* | устанавливается цвет фона |
| *long* ***GetColorText*** *()* | опрашивается цвет текста |
| *void* ***SetColorText*** *(unsigned long propVal)* | устанавливается цвет текста |
| **Отображение границ** | |
| *BOOL* ***GetLeftBound****()* | опрашивается видимость границы слева (true(1) - Видимость отображения границ, false(0) - Видимость отображения границ – выключена) |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *void* ***SetLeftBound****(BOOL propVal)* | устанавливается видимость границы слева (true(1) - Видимость отображения границ, false(0) - Видимость отображения границ – выключена) |
| *BOOL* ***GetRightBound*** *()* | опрашивается видимость границы справа (true(1) - Видимость отображения границ, false(0) - Видимость отображения границ – выключена) |
| *void* ***SetRightBound*** *(BOOL propVal)* | устанавливается видимость границы справа (true(1) - Видимость отображения границ, false(0) - Видимость отображения границ – выключена) |
| *BOOL* ***GetTopBound*** *()* | опрашивается видимость границы сверху (true(1) - Видимость отображения границ, false(0) - Видимость отображения границ – выключена) |
| *void* ***SetTopBound*** *(BOOL propVal)* | устанавливается видимость границы сверху (true(1) - Видимость отображения границ, false(0) - Видимость отображения границ – выключена) |
| *BOOL* ***GetBottomBound*** *()* | опрашивается видимость границы снизу (true(1) - Видимость отображения границ, false(0) - Видимость отображения границ – выключена) |
| *void* ***SetBottomBound*** *(BOOL propVal)* | устанавливается видимость границы снизу (true(1) - Видимость отображения границ, false(0) - Видимость отображения границ – выключена) |
| **Параметры** | |
| *long* ***GetChannel*** *()* | опрашивается порядковый номер канала |
| *void* ***SetChannel*** *(long propVal)* | устанавливается порядковый номер канала |
| *long* ***GetType*** *()* | опрашивается тип списка |
| *void* ***SetType*** *(long propVal)* | устанавливается тип списка |
| *BOOL* ***GetFilter****()* | опрашивается фильтр по каналам (true(1) - включен, false(0) - выключен) |
| *void* ***SetFilter****(BOOL propVal)* | устанавливается фильтр по каналам (true(1) - включен, false(0) - выключен) |
| *CString* ***GetConversionFilter****()* | опрашивается фильтр по единицам измерения |
| *void* ***SetConversionFilter(****CString propVal)* | устанавливается фильтр по единицам измерения |
| *CString* ***GetToolTipText****()* | опрашивается текст подсказки |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *void* ***SetToolTipText****(CString propVal)* | устанавливается текст подсказки |
| *BOOL* ***GetZETLabStyle****()* | опрашивается стиль (true(1) - включен, false(0) - выключен) |
| *void* ***SetZETLabStyle****(BOOL propVal)* | устанавливается стиль (true(1) - включен, false(0) - выключен) |
| *CString* ***GetTextString*** *()* | опрашивается текстовая строка |
| *void* ***SetTextString*** *(CString propVal)* | устанавливается текстовая строка |
| *CString* ***GetCaption*** *()* | опрашивается название компонента |
| *void* ***SetCaption*** *(CString propVal)* | устанавливается название компонента |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Методы ZETChanSelector.ocx | |
|---|---|
| *void* ***AboutBox****()* | содержит информацию о программе |

© ООО "ЭТМС" 1992-2024. Все права защищены

| **События от компоненты ZETChanSelector.ocx** | |
|---|---|
| *void* ***ModifyChannel****(LONG channelNum, LONG type)* | изменяет канал |

© ООО "ЭТМС" 1992-2024. Все права защищены

# GdiPlusComboBox.ocx предназначен для комбинированного списка

Компонент **GdiPlusComboBox.ocx** используется для комбинированного списка. Применяется в программах из состава ZETLab, например, в программах вольтметров, виброметра.

Описание: элемент управления ComboBox очень похож на элемент управления DropDownList, однако в отличие от последнего, позволяет в качестве альтернативы выбору пункта из списка вводить текст в свободном формате. Пример:

```
Gui, Add, ComboBox, vColorChoice, Red|Green|Blue|Black|White
```

В дополнение к опциям, присущим элементу управления DropDownList, о которых рассказывалось выше, в опции ComboBox можно включить слово Limit, ограничивающее ввод данных шириной редактируемого поля ввода. Можно также задать слово Simple, в результате чего ComboBox будет вести себя как редактируемое поле ввода со списком (ListBox) внизу.

При использовании команды Gui Submit, выходной переменной, ассоциированной с элементом управления (если такая переменная существует), присваивается текст выбранного на тот момент пункта. Однако если элемент управления имеет свойство AltSubmit, то вместо текста выходной переменной будет присвоен номер позиции этого пункта (номер позиции первого пункта - 1, второго - 2, и т. д.). Если пункт не выбран, то и в том и в другом случае выходной переменной будет присвоено содержимое редактируемого поля ввода.
В опциях элемента управления можно поместить метку перехода, например, gMySubroutine. Это приведет к тому, что метка будет автоматически загружаться при выборе каждого нового пункта.

## Глава 1.Установка компонента GdiPlusComboBox.ocx

Для работы с модулем GdiPlusComboBox его необходимо сначала установить в программу. При загрузке программы необходимо загрузить с помощью компонента необходимую программу, разместить окно программы на экране в удобном месте,

© ООО "ЭТМС" 1992-2024. Все права защищены

установить необходимые параметры в программе и затем по событиям, происходящим от программы считывать данные из программы.

Для установки компонента GdiPlusComboBox в проект *Microsoft Visual Studio 2010*, необходимо кликнуть правой кнопкой мыши на форме диалога и из появившегося меню выбрить пункт *Insert ActiveX Control*...(рисунок 1.1)

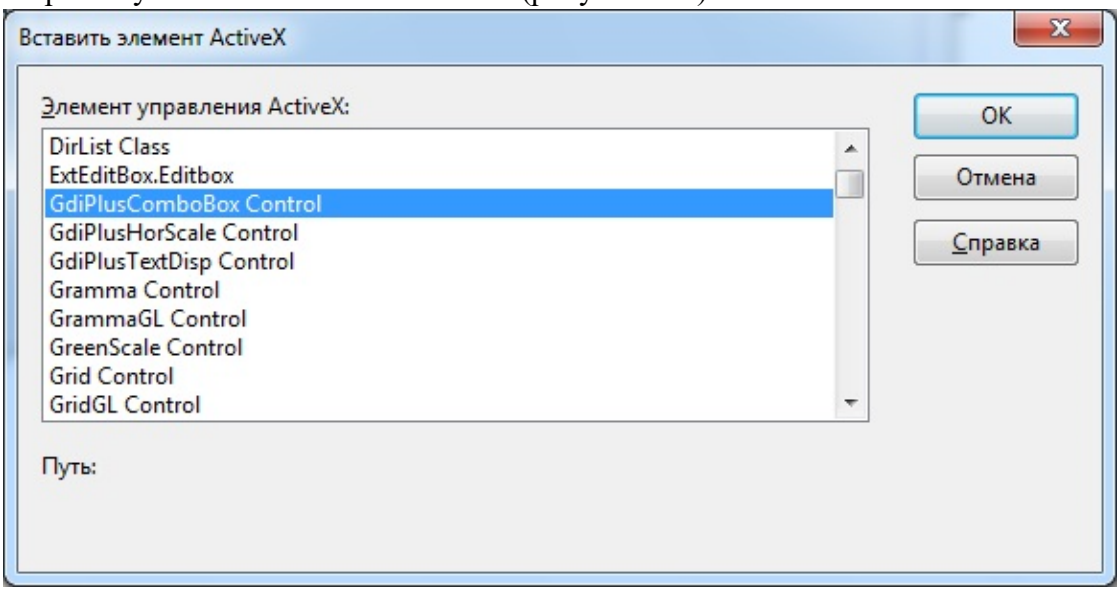



Рисунок 1.1

Из появившегося диалогового окна следует выбрать компонент *GdiPlusComboBox Control* и нажать *OK* (рисунок 1.2). После этого компонент *GdiPlusComboBox.ocx* появится на форме диалога (рисунок 1.2).

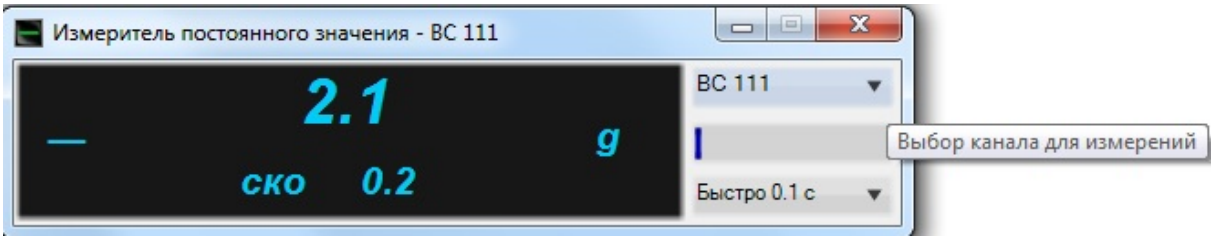



Рисунок 1.2

Одна установленная на форме компонента позволяет управлять одной программой. При необходимости управлять несколькими программами, необходимо установить на форму несколько компонент. Для того чтобы компонента не была видна на форме во время выполнения программы, необходимо установить свойство *Visible* равным *False*.

# Глава 2.Интерфейс компонента GdiPlusComboBox.ocx

© ООО "ЭТМС" 1992-2024. Все права защищены




При клике левой кнопкой "мыши" по верхнему элементу (GdiPlusComboBox.ocx) появляется выпадающий перечень доступных каналов сервера ZETLAB, пример которого представлен на рисунке выше. Выбор канала осуществляется путём клика левой кнопки "мыши" на требуемом канале. После этого перечень каналов исчезает, в заголовке окна появляется название выбранного канала, правый индикатор отображает размерность нового канала, а также возможна замена заголовка программы в соответствии с таблицей, приведенной ниже.

Строка заголовка программы содержит сам заголовок и имя измеряемого канала. Заголовок изменяется в зависимости от размерности выбранного канала для изменений. В таблице ниже представлены все возможные варианты заголовков программы.

| Заголовок | Размерность канала |
|---|---|
| Вольтметр постоянного тока | В, мВ или мкВ |
| Амперметр постоянного тока | А, мА или мкА |
| Температура | град или С |
| Измеритель постоянного значения | остальные случаи |

# Глава 3.Полное описание свойств, методов и событий GdiPlusComboBox.ocx

Следует отметить, что компонент автомасштабируется при изменении его размеров, поэтому необходимо следить за тем, чтобы высота компонента в приложении была не менее 80 пикселей, иначе он может отображаться некорректно.

## 3.1.Свойства GdiPlusComboBox.ocx

Свойства можно задавать на этапе проектирования и во время выполнения программы. На этапе проектирования при выборе компонента и при нажатии клавиши <F4> появляется окно свойств. Эти свойства можно устанавливать на этапе проектирования программы.

© ООО "ЭТМС" 1992-2024. Все права защищены

## Цвета

Цвета задаются при помощи макроса *RGB(RED, GREEN, BLUE)*. Значения *RED*, *GREEN*, *BLUE* задают интенсивности цветов красного, зеленого и синего. Интенсивности изменяются в пределах от 0 до 255.

*OLE_COLOR* ***backColor*** – цвет фона

*OLE_COLOR* ***ColorText*** – цвет текста

Например, для изменении фона графика с желтого на зеленый, могут быть записаны числа:

```
OLE_COLOR a = 0xFF00;
OLE_COLOR b = 0xFFFF;
```

## Отображение рамки

Используются для создания эффекта объемного элемента.

*VARIANT_BOOL* ***LeftBound*** – рисовать левую границу рамки.

*VARIANT_BOOL* ***RightBound*** – рисовать правую границу рамки.

*VARIANT_BOOL* ***TopBound*** – рисовать верхнюю границу рамки.

*VARIANT_BOOL* ***BottomBound*** – рисовать нижнюю границу рамки.

Изменение видимости отображения границ :
true - Видимость отображения границ.
false - Видимость отображения границ - выключена.

## Параметры

*long* ***TextSize*** – размер шрифта.

*long* ***TextFontType*** - тип шрифта.

*BSTR* ***TextString***- текстовая строка

*BSTR* ***Caption*** – название компонента

© ООО "ЭТМС" 1992-2024. Все права защищены

## 3.2.Методы GdiPlusComboBox.ocx

Методы можно использовать только во время выполнения программы.

*void* ***AboutBox****()* –содержит информацию о программе.

## 3.3.События от компоненты GdiPlusComboBox.ocx

*void* ***SelectedItem****(BSTR itemString, LONG index)* - изменяет индекс

# Глава 4.Краткий перечень функций, используемые GdiPlusComboBox.ocx

| **Свойства GdiPlusComboBox.ocx** | |
|---|---|
| **Цвета** | |
| *long* ***GetbackColor*** *()* | опрашивается цвет фона |
| *void* ***SetbackColor*** *(unsigned long propVal)* | устанавливается цвет фона |
| *long* ***GetColorText*** *()* | опрашивается цвет текста |
| *void* ***SetColorText*** *(unsigned long propVal)* | устанавливается цвет текста |
| **Отображение границ** | |
| *BOOL* ***GetLeftBound****()* | опрашивается рисовать левую границу рамки (true(1) - рисовать, false(0) - не рисовать) |
| *void* ***SetLeftBound****(BOOL propVal)* | устанавливается рисовать левую границу рамки (true(1) - рисовать, false(0) - не рисовать) |
| *BOOL* ***GetRightBound*** *()* | опрашивается рисовать правую границу рамки (true(1) - рисовать, false(0) - не рисовать) |
| *void* ***SetRightBound*** *(BOOL propVal)* | устанавливается рисовать правую границу рамки (true(1) - рисовать, false(0) - не рисовать) |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *BOOL* ***GetTopBound*** *()* | опрашивается рисовать верхнюю границу рамки (true(1) - рисовать, false(0) - не рисовать) |
| *void* ***SetTopBound*** *(BOOL propVal)* | устанавливается рисовать верхнюю границу рамки (true(1) - рисовать, false(0) - не рисовать) |
| *BOOL* ***GetBottomBound*** *()* | опрашивается рисовать нижнюю границу рамки (true(1) - рисовать, false(0) - не рисовать) |
| *void S****etBottomBound*** *(BOOL propVal)* | устанавливается рисовать нижнюю границу рамки (true(1) - рисовать, false(0) - не рисовать) |
| **Параметры** | |
| *long* ***GetTextSize****()* | опрашивается размер шрифта |
| *void* ***SetTextSize****(long propVal)* | устанавливается размер шрифта |
| *long* ***GetTextFontType*** *()* | опрашивается тип шрифта |
| *void* ***SetTextFontType****(long propVal)* | устанавливается тип шрифта |
| *CString* ***GetTextString*** *()* | опрашивается текстовая строка |
| *void* ***SetTextString*** *(CString propVal)* | устанавливается текстовая строка |
| *CString* ***GetCaption*** *()* | опрашивается название компонента |
| *void* ***SetCaption*** *(CString propVal)* | устанавливается название компонента |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Методы GdiPlusComboBox.ocx | |
|---|---|
| *void* ***AboutBox****()* | содержит информацию о программе |

© ООО "ЭТМС" 1992-2024. Все права защищены

| **События от компоненты GdiPlusComboBox.ocx** | |
|---|---|
| *void **SelectedItem**(BSTR itemString, LONG index)* | изменяет индекс |

© ООО "ЭТМС" 1992-2024. Все права защищены

# ZetMultiLevelIndicator.ocx используется для многоканального индикатора уровня

Компонент **ZetMultiLevelIndicator.ocx** используется для многоканального индикатора уровня.

## Глава 1.Полное описание свойств, методов и событий ZetMultiLevelIndicator.ocx

Следует отметить, что компонент автомасштабируется при изменении его размеров, поэтому необходимо следить за тем, чтобы высота компонента в приложении была не менее 80 пикселей, иначе он может отображаться некорректно.

### 1.1.Методы ZetMultiLevelIndicator.ocx

Методы можно использовать только во время выполнения программы.

*void* ***SetMinIndicatorValue****(FLOAT newVal)* – Минимальное значение в индикаторе.

*void* ***SetMaxIndicatorValue****(FLOAT newVal)* – Максимальное значение в индикаторе.

*void* ***SetNumberOfMarks****(LONG newVal)* – Количество маркеров.

*void* ***SetMarkerValue****(LONG number, FLOAT value)* – Задать значение определённого маркера.

© ООО "ЭТМС" 1992-2024. Все права защищены

*void* ***SetMarkerColor****(LONG number, ULONG color)* – Задать цвет определённого маркера.

*void* ***SetMarkerValuesByArray(LONG quantity, FLOAT* pointer)*** – Задать значения всех маркеров с помощью массива.

*void* ***SetIndicatorOffsetFromTop****(LONG newVal)* – Смещение рамки индикатора от верхнего края окна контрола.

*void* ***SetIndicatorOffsetFromBottom****(LONG newVal)* – Смещение рамки индикатора от нижнего края окна контрола.

*void* ***SetCustomAxisParams****(LONG* pointer)* – Настроить параметры отображения индикатора с помощью структуры связанной с GridGL.

*void* ***SetDrawDigits****(BOOL newVal)* – Флаг отображения цифр.

*void* ***SetMarkerThickness****(LONG newVal)* – Толщина маркеров.

# Глава 2.Краткий перечень функций, используемые ZetMultiLevelIndicator.ocx

| **Методы ZetMultiLevelIndicator.ocx** | |
|---|---|
| *void* ***SetMinIndicatorValue****(FLOAT newVal)* – | Минимальное значение в индикаторе. |
| *void* ***SetMaxIndicatorValue****(FLOAT newVal)* | Максимальное значение в индикаторе. |
| *void* ***SetNumberOfMarks****(LONG newVal)* | Количество маркеров. |
| *void* ***SetMarkerValue****(LONG number, FLOAT value))* | Задать цвет определённого маркера. |
| *void* ***SetMarkerColor****(LONG number, ULONG color)* | Задать цвет определённого маркера. |
| *void* ***SetMarkerValuesByArray****(LONG quantity, FLOAT* pointer)* | Задать значения всех маркеров с помощью массива. |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *void* ***SetIndicatorOffsetFromTop****(LONG newVal)* | Смещение рамки индикатора от верхнего края окна контрола. |
| *void* ***SetIndicatorOffsetFromBottom****(LONG newVal)* | Смещение рамки индикатора от нижнего края окна контрола. |
| *void* ***SetCustomAxisParams****(LONG* pointer)* | Настроить параметры отображения индикатора с помощью структуры связанной с GridGL.) |
| *BOOL* ***GetTopBound*** *()* | Флаг отображения цифр. |
| *void* ***SetDrawDigits****(BOOL newVal)* | Толщина маркеров. |
| *void* ***SetMarkerThickness****(LONG newVal)* | устанавливается видимость границы снизу (true(1) - Видимость отображения границ, false(0) - Видимость отображения границ – выключена) |

© ООО "ЭТМС" 1992-2024. Все права защищены

# Описание программных модулей индикаторов

## Глава 1.zet_boolean.ocx представляет собой лампочку-индикатор

### *Назначение компонента*

Компонент представляет собой лампочку-индикатор, которая «горит» во включенном состоянии и «не горит» в выключенном.

**Внешний вид компонента:**

| Цвет компонента: *красный (255, 0, 0)* | | Цвет компонента: *зеленый (0, 255, 0)* | | Цвет компонента: *голубой (0, 255, 255)* | |
|---|---|---|---|---|---|
|  |  |  |  |  |  |
| выкл. | вкл. | выкл. | вкл. | выкл. | вкл. |

### *Свойства*

*Status* – состояние (вкл./выкл.)

*Color* – цвет компонента.

*FonColor* – цвет фона.

Цвет задается при помощи макроса *RGB(RED, GREEN, BLUE)*. Значения *RED*, *GREEN*, *BLUE* задают интенсивности цветов красного, зеленого и синего. Интенсивности изменяются в пределах от 0 до 255.

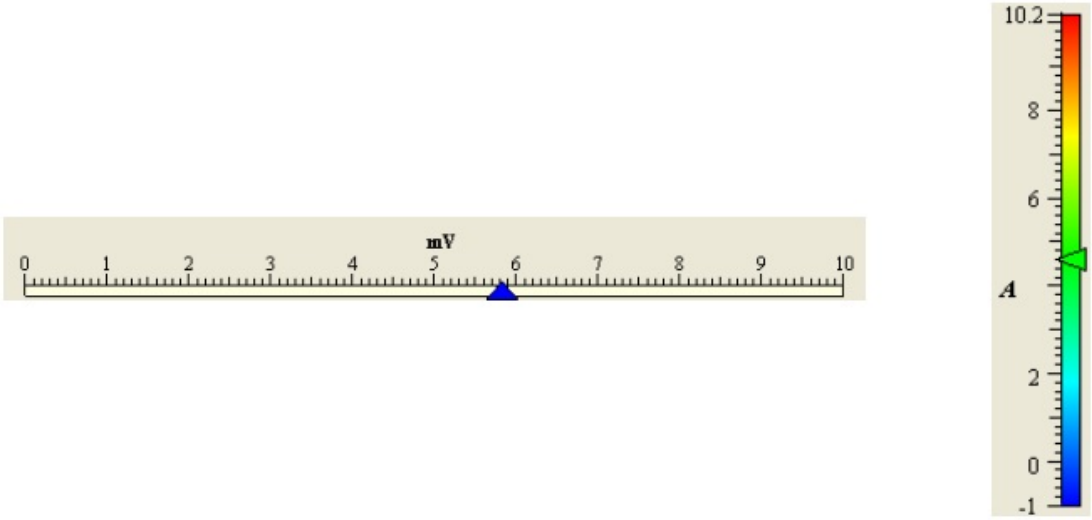
© ООО "ЭТМС" 1992-2024. Все права защищены

# Глава 2.zet_horizontal_slider.ocx, zet_vertical_slider.ocx линейные аналоговые регуляторы шкалы

## *Назначение компонента*

Компоненты представляют собой линейные аналоговые регуляторы и предназначены для изменения целых и дробных десятичных числовых величин. Необходимое значение устанавливается с помощью мыши. Пределы шкалы и единица измерения задаются. Разметка шкалы выполняется автоматически. Цвета компонента могут быть заданы пользователем.

**Общий вид компонентов:**

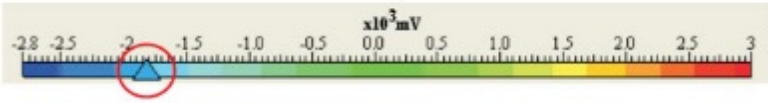



## *Использование компонента*

Когда компонент активен, необходимое значение устанавливается следующим образом: удерживая левую кнопку мыши нажатой, необходимо «ухватиться» за стрелочку индикатора и переместить ее до нужной отметки. Когда кнопка мыши будет отпущена, будет установлено выбранное значение. Так же стрелочка индикатора перемещается по основным и промежуточным значениям при вращении колесика мышки.

© ООО "ЭТМС" 1992-2024. Все права защищены

При разноцветной окраске шкалы цвет стрелки компонентов совпадает с цветом шкалы, соответствующим текущему значению.




zet_horizontal_slider

## *Свойства*

### *Режим работы*

*Status(Cmamyc)* – 0 - выкл. (компонент пассивен), 1 - вкл. (компонент активен).

### *Цифровая шкала*

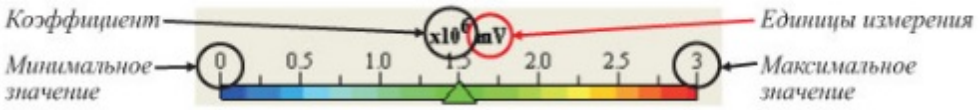
Все значения имею тип double. Точность шкалы – 2 знака после запятой. При значениях по модулю больше 1000 и меньше 0,1 вводится коэффициент, который отображается над табло.

*Value* – текущее значение индикатора.

*MinValue* – минимальное устанавливаемое значение индикатора.

*MaxValue* – максимальное устанавливаемое значение индикатора.

*Unit* – единица измерения.




*zet_horizontal_slider (ForeColor = 0)*

### *Цвета*

*ForeColor* – определяет раскраску шкалы: 0 – произвольная (плавный переход от синего к зеленому потом к красному) при этом цвет стрелки совпадает с цветом шкалы, соответствующим текущему значению, 1 – однотонная шкала, цвет шкалы и цвет стрелки задается пользователем.

*ScaleColor* – цвет шкалы при *ForeColor = 1*.

*NeedleColor* – цвет стрелки при *ForeColor = 1*.

*FonColor* – цвет фона.

© ООО "ЭТМС" 1992-2024. Все права защищены

Цвет задается при помощи макроса *RGB(RED, GREEN, BLUE)*. Значения *RED*, *GREEN*, *BLUE* задают интенсивности цветов красного, зеленого и синего. Интенсивности изменяются в пределах от 0 до 255.

### *Шкала делений*

Пользователем задается минимальное и максимальное значение шкалы делений, и разметка производится автоматически. Цифры проставляются над основными делениями, между которыми проставляются промежуточные метки. Повысить точность шкалы можно развиб каждый промежуток между основными значениями на 10 частей.

*Point* – 1 – разделить каждое основное деление на 10 частей, 0 – без малых меток.

### *Шрифт*

*FontName* – название шрифта.

*FontSize* – размер шрифта.

Выбранным шрифтом отображаются значения над основными делениями цифровой шкалы, коэффициент и единица измерения выводятся жирным шрифтом. Степень коэффициента выделяется курсивом.

## События от компонент

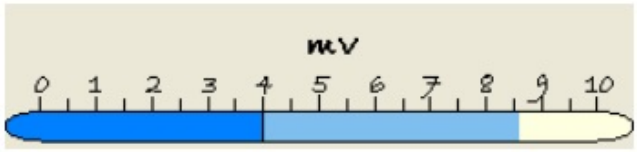
*ModifyValue* – Событие выводит страницу свойств, где пользователь может установить режим работы, вид шкалы, цвет фона, тип шрифта, максимальное и минимальное и текущее значения.

# Глава 3.zet_horizontal_level, zet_vertical_level линейные аналоговые индикаторы уровня

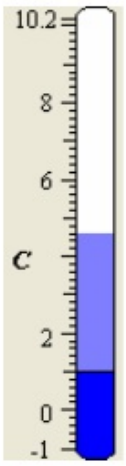
### *Назначение компонента*

Компоненты представляют собой линейные аналоговые индикаторы и предназначены для отображения целых и дробных десятичных числовых величин на цифровой шкале. Пределы шкалы и единицы измерения задаются. Разметка шкалы выполняется автоматически. Цвета компонента могут быть заданы пользователем.

© ООО "ЭТМС" 1992-2024. Все права защищены



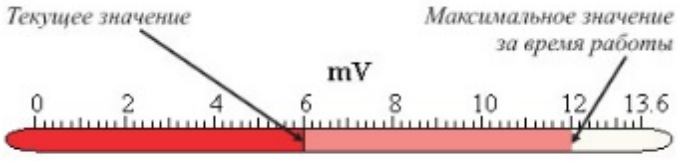

*zet_horizontal_level (*шрифт Bradley Hand*)*




*zet_vertical_level*

## *Использование компонента*

Когда компонент активен, уровень столбца находится на отметке текущего значения. Также у шкалы может быть «след» уровень которого указывает на максимальное значение, которое было достигнуто за время работы компонента.




*zet_horizontal_level*

## *Свойства*

### *Режим работы*

*Status* – 0 - выкл. (компонент пассивен), 1 - вкл. (компонент активен).

LeftRight - Расположение левостороннее - 0 или правостороннее - 1" используется только в zet_vertical_level.ocx.

### *Цифровая шкала*

Все значения имею тип double. Точность шкалы – 2 знака после запятой. При значениях по модулю больше 1000 и меньше 0,1 вводится коэффициент, который отображается над табло.

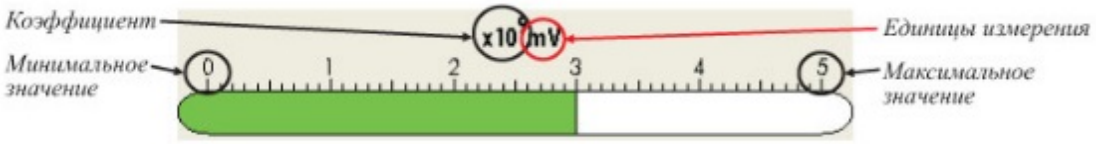
*Value* – текущее значение индикатора.

© ООО "ЭТМС" 1992-2024. Все права защищены

*MinValue* – минимальное устанавливаемое значение индикатора.

*MaxValue* – максимальное устанавливаемое значение индикатора.

*Unit* – единицы измерения.




*zet_horizontal_level* (Track = 0)

### *Цвета*

*ScaleColor* – цвет шкалы.

*ColumnColor* – цвет столбца

*FonColor* – цвет фона.

*Track* – 0 – не прорисовывать «след», 1 – прорисовывать цвет.

Цвет задается при помощи макроса *RGB(RED, GREEN, BLUE)*. Значения *RED*, *GREEN*, *BLUE* задают интенсивности цветов красного, зеленого и синего. Интенсивности изменяются в пределах от 0 до 255.

### *Шкала делений*

Пользователем задается минимальное и максимальное значение шкалы делений и разметка производится автоматически. Цифры проставляются над основными делениями, между которыми проставляются промежуточные метки. Повысить точность шкалы можно развиб каждый промежуток между основными значениями на 10 частей.

*Point* – 1 – разделить каждое основное деление на 10 частей, 0 – без малых меток.

### *Шрифт*

*FontName* – имя шрифта.

*FontSize* – размер шрифта.

Выбранным шрифтом отображаются значения над основными делениями цифровой шкалы, коэффициент и единицы измерения выводятся жирным шрифтом. Степень коэффициента выделяется курсивом.

© ООО "ЭТМС" 1992-2024. Все права защищены

# Глава 4.ZETTaho.ocx предназначен для измерения частоты вращения

Компонент **ZETTaho.ocx** для измерения скорости вращения роторов, валов, вращающихся механизмов и т.п.

## 4.1.Установка компонента ZETTaho.ocx

Для работы с модулем ZETTaho.ocx его необходимо сначала установить в программу. При загрузке программы необходимо загрузить с помощью компонента необходимую программу, разместить окно программы на экране в удобном месте, установить необходимые параметры в программе и затем по событиям, происходящим от программы считывать данные из программы.

Для установки компонента ZETTaho в проект *Microsoft Visual Studio 2010*, необходимо кликнуть правой кнопкой мыши на форме диалога и из появившегося меню выбрить пункт *Insert ActiveX Control*...(рисунок 1.1)

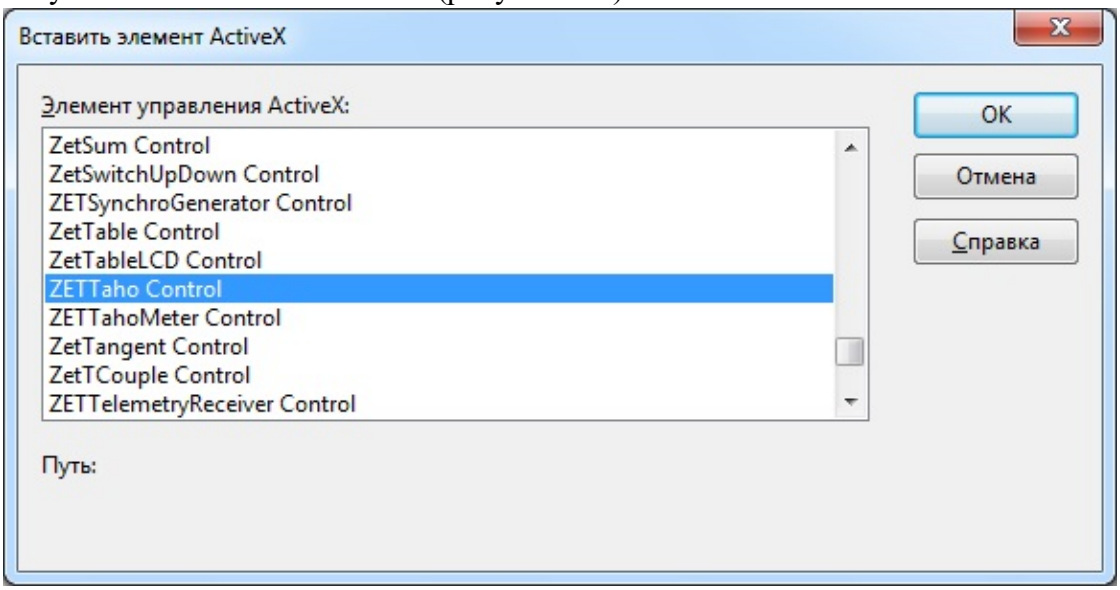



Рисунок 1.1

Из появившегося диалогового окна следует выбрать компонент *ZETTaho Control* и нажать *OK* (рисунок 1.2). После этого компонент *ZETTaho.ocx.ocx* появится на форме диалога (рисунок 1.2).

© ООО "ЭТМС" 1992-2024. Все права защищены

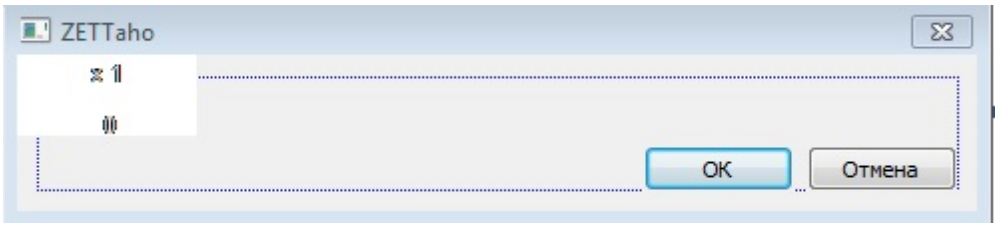
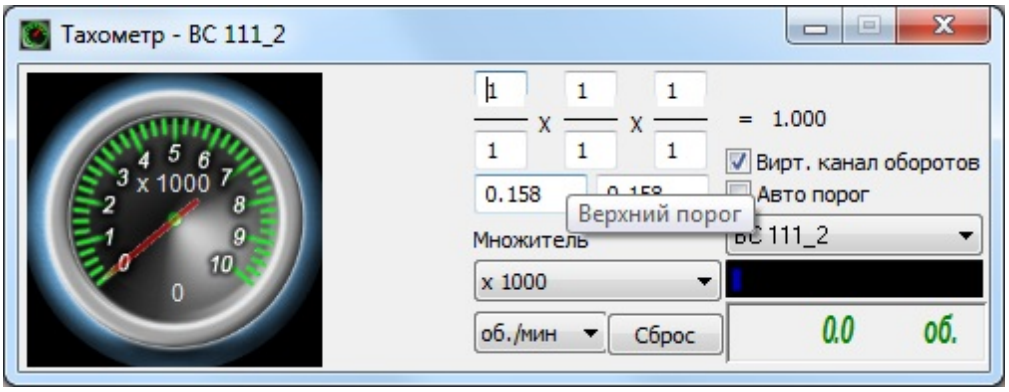



Рисунок 1.2

Одна установленная на форме компонента позволяет управлять одной программой. При необходимости управлять несколькими программами, необходимо установить на форму несколько компонент. Для того чтобы компонента не была видна на форме во время выполнения программы, необходимо установить свойство *Visible* равным *False*.

## 4.2.Интерфейс компонента ZETTaho.ocx




Компонент **ZETTaho.ocx** используется в программе Тахометр, которая предназначена для измерения частоты вращения валов машин и механизмов, а также для подсчета количества полных оборотов.

**Назначение:**

Для сложных узлов, например, коробки передач, устанавливается передаточное число.

Параметры сигнала тахометра являются ценными при диагностике, а также в практических исследованиях характеристик двигателей внутреннего сгорания (ДВС) и других роторных механизмов (турбины, компрессоры, насосы, вентиляторы и т.д.).

Для исследования, диагностики и балансировки различных вращающихся механизмов используется программа Синхронное накопление.

Для исследования и диагностики нестационарных процессов во время вращения используется программа Индикатор угловых перемещений.

При использовании программы Многоканальный самописец в режиме отображения частоты сигнала можно получить количество оборотов в минуту (частота

© ООО "ЭТМС" 1992-2024. Все права защищены

* 60 = об/мин). По полученным параметрам графика, определяются время разгона и время торможения ДВС. Также по этим параметрам можно определить демпфирование системы с целью дальнейшего анализа.

При работе механизма с постоянной частотой вращения по данному графику можно анализировать:

·поведение регулятора оборотов системы (восстановление заданных оборотов) при подключении или изменении нагрузки потребителя мощности;

·определение степени неравномерности вращения при постоянной нагрузке (особенно для ДВС).

При интегрировании значений мгновенной угловой скорости вращения можно получить значения крутильных колебаний вала в радианах или градусах.

Измерение разности мгновенных скоростей (в двух сечениях вала) позволит определять угол скручивания вала (постоянный или переменный) и вычислять такие параметры, как напряжение в вале и мощность механизма.

## 4.3.Полное описание свойств, методов и событий ZETTaho.ocx

Следует отметить, что компонент автомасштабируется при изменении его размеров, поэтому необходимо следить за тем, чтобы высота компонента в приложении была не менее 80 пикселей, иначе он может отображаться некорректно.

### Свойства ZETTaho.ocx

Свойства можно задавать на этапе проектирования и во время выполнения программы. На этапе проектирования при выборе компонента и при нажатии клавиши <F4> появляется окно свойств. Эти свойства можно устанавливать на этапе проектирования программы.

*FLOAT* ***CurrValue*** – текущее значение частоты вращения.

*long* ***multiplier*** – множитель данного компонента.

### Методы ZETTaho.ocx

Методы можно использовать только во время выполнения программы.

*void* ***AboutBox****()* – содержит информацию о программе.

*void* ***Format****(LONG real, LONG fractional)* – формат отображаемого числа частоты вращения на индикаторе

© ООО "ЭТМС" 1992-2024. Все права защищены

## 4.4.Краткий перечень функций, используемые ZETTaho.ocx

| Свойства ZETTaho.ocx | |
|---|---|
| **Параметры** | |
| *float* ***GetCurrValue****()* | опрашивается текущее значение частоты вращения |
| *void* ***SetCurrValue****(float propVal)* | устанавливается текущее значение частоты вращения |
| *long* ***Getmultiplier****()* | опрашивается множитель данного компонента |
| *void* ***Setmultiplier****(long propVal)* | устанавливается множитель данного компонента |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Методы ZETTaho.ocx | |
|---|---|
| *void* ***AboutBox****()* | содержит информацию о программе |
| *void* ***Format****(LONG real, LONG fractional)* | формат отображаемого числа частоты вращения на индикаторе |

# Глава 5.GdiPlusHorScale.ocx предназначен для отображения интегральных уровней сигналов

Компонент **GdiPlusHorScale.ocx** используется для отображения интегральных уровней сигналов. Применяется практически во всех программах из состава ZETLab, например, в программах вольтметров, частотомеров, фазометров и т.д. Компактные размеры и большая информативность данного компонента делают очень удобным его использование в таких программах как Многоканальный осциллограф или XYZ-Осциллограф, XY-плоттер, где необходимо наблюдать за уровнем сразу нескольких сигналов, поступающих с модулей АЦП и ЦАП или анализаторов спектра.

В зависимости от измеряемого уровня сигнала изменяется цвет и заполненность компонента от черного до красного. В компоненте существует опция запоминания состояния перегрузки - в этом случае компонент заполняется полностью и в правой части загорается и остается поле красного цвета. При нажатии на это поле состояние перегрузки сбрасывается..

## 5.1.Установка компонента GdiPlusHorScale.ocx

Для работы с модулем GdiPlusHorScale его необходимо сначала установить в программу. При загрузке программы необходимо загрузить с помощью компонента необходимую программу, разместить окно программы на экране в удобном месте, установить необходимые параметры в программе и затем по событиям, происходящим от программы считывать данные из программы.

Для установки компонента GdiPlusHorScale в проект *Microsoft Visual Studio 2010*, необходимо кликнуть правой кнопкой мыши на форме диалога и из появившегося меню выбрить пункт *Insert ActiveX Control*...(рисунок 1.1)

© ООО "ЭТМС" 1992-2024. Все права защищены

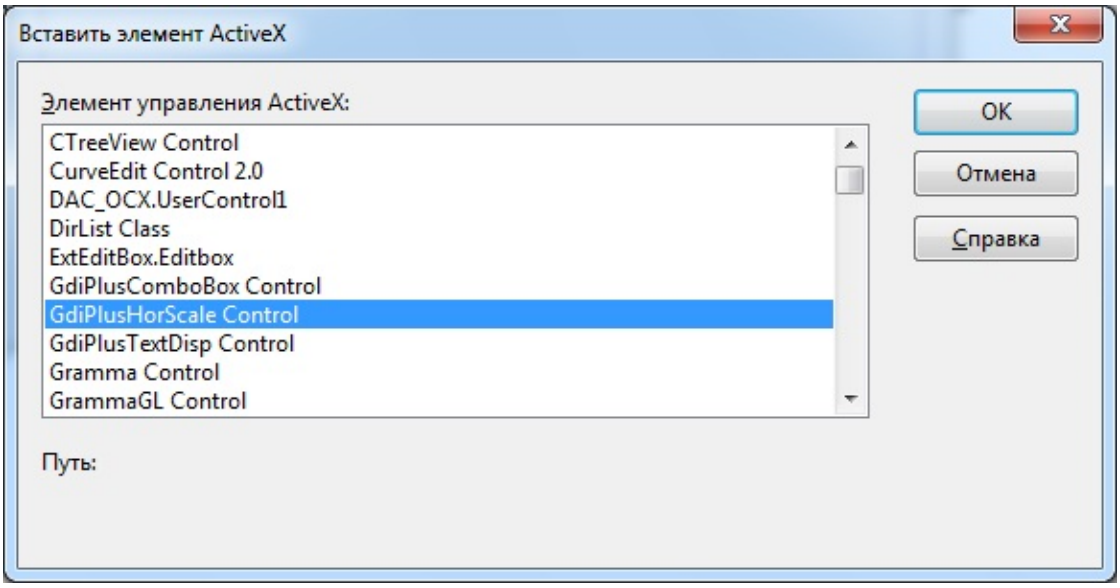



Рисунок 1.1

Из появившегося диалогового окна следует выбрать компонент *GdiPlusHorScale Control* и нажать *OK* (рисунок 1.2). После этого компонент *GdiPlusHorScale.ocx* появится на форме диалога (рисунок 1.2).

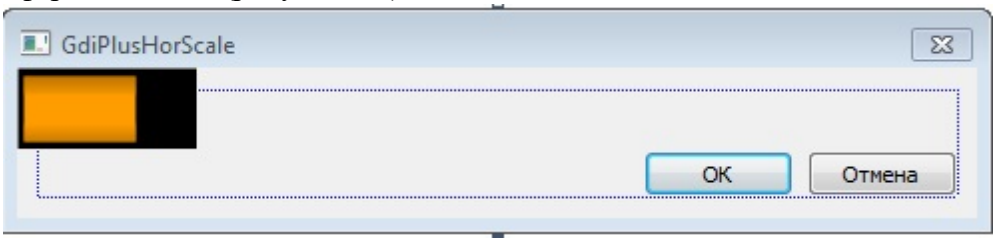



Рисунок 1.2

Одна установленная на форме компонента позволяет управлять одной программой. При необходимости управлять несколькими программами, необходимо установить на форму несколько компонент. Для того чтобы компонента не была видна на форме во время выполнения программы, необходимо установить свойство *Visible* равным *False*.

## 5.2.Интерфейс компонента GdiPlusHorScale.ocx




© ООО "ЭТМС" 1992-2024. Все права защищены

Второй элемент показывает интегральный уровень сигнала. Отображение осуществляется с помощью изменения цвета индикатора и ширины цветного поля индикатора, который может занимать до двух третей всего поля. При изменении уровня канала от минимального до максимального допустимого уровня цвет меняется от синего до красного. Чем выше уровень, тем больше заполняется индикатор. При превышении максимально допустимого уровня индикатор заполняется полностью красным цветом. Правый край индикатора останется красным до тех пор, пока перегрузка по каналу не будет снята и пользователь не кликнет по индикатору левой кнопкой "мыши".

## 5.3.Полное описание свойств, методов и событий GdiPlusHorScale.ocx

Следует отметить, что компонент автомасштабируется при изменении его размеров, поэтому необходимо следить за тем, чтобы высота компонента в приложении была не менее 80 пикселей, иначе он может отображаться некорректно.

### Свойства GdiPlusHorScale

Свойства можно задавать на этапе проектирования и во время выполнения программы. На этапе проектирования при выборе компонента и при нажатии клавиши <F4> появляется окно свойств. Эти свойства можно устанавливать на этапе проектирования программы.

Цвета задаются при помощи макроса *RGB(RED, GREEN, BLUE)*. Значения *RED*, *GREEN*, *BLUE* задают интенсивности цветов красного, зеленого и синего. Интенсивности изменяются в пределах от 0 до 255.

*OLE_COLOR* ***backColor*** – цвет фона

Например, для изменении фона графика с желтого на зеленый, могут быть записаны числа:

```
OLE_COLOR a = 0xFF00;
OLE_COLOR b = 0xFFFF;
```

Используются для создания эффекта объемного элемента.

© ООО "ЭТМС" 1992-2024. Все права защищены

*VARIANT_BOOL* ***LeftBound*** – рисовать левую границу рамки.

*VARIANT_BOOL* ***RightBound*** – рисовать правую границу рамки.

*VARIANT_BOOL* ***TopBound*** – рисовать верхнюю границу рамки.

*VARIANT_BOOL* ***BottomBound*** – рисовать нижнюю границу рамки.

Изменение видимости отображения границ :
true - Видимость отображения границ.
false - Видимость отображения границ - выключена.

Используются для ориентации компонента в пространстве.

*VARIANT_BOOL* ***Horisontal*** – ориентация компонента.

Изменение видимости отображения границ :
true - Ориентация компонента горизонтальня;
false - Ориентация компонента вертикальная.

*double* ***Amplitude*** – амплитуда индикатора уровня.

*long* ***Log*** – линейная или логарифмическая шкала по Y. Параметр может принимать значения 0 или 1.

0 – линейный масштаб.

1 – логарифмический масштаб.

*double* ***Memory*** – размер выделяемой памяти

## Методы GdiPlusHorScale.ocx

Методы можно использовать только во время выполнения программы.

*void* ***AboutBox()*** –содержит информацию о программе.

© ООО "ЭТМС" 1992-2024. Все права защищены

## 5.4.Краткий перечень функций, используемые GdiPlusHorScale.ocx

| Свойства GdiPlusHorScale.ocx | |
|---|---|
| **Цвета** | |
| *long* ***GetbackColor*** *()* | опрашивается цвет фона |
| *void* ***SetbackColor*** *(*unsigned *long propVal)* | устанавливается цвет фона |
| **Отображение рамки** | |
| *BOOL* ***GetLeftBound****()* | опрашивается рисовать левую границу рамки (true(1) - рисовать, false(0) - не рисовать) |
| *void* ***SetLeftBound****(BOOL propVal)* | устанавливается рисовать левую границу рамки (true(1) - рисовать, false(0) - не рисовать) |
| *BOOL* ***GetRightBound*** *()* | опрашивается рисовать правую границу рамки (true(1) - рисовать, false(0) - не рисовать) |
| *void* ***SetRightBound*** *(BOOL propVal)* | устанавливается рисовать правую границу рамки (true(1) - рисовать, false(0) - не рисовать) |
| *BOOL* ***GetTopBound*** *()* | опрашивается рисовать верхнюю границу рамки (true(1) - рисовать, false(0) - не рисовать) |
| *void* ***SetTopBound*** *(BOOL propVal)* | устанавливается рисовать верхнюю границу рамки (true(1) - рисовать, false(0) - не рисовать) |
| *BOOL* ***GetBottomBound*** *()* | опрашивается рисовать нижнюю границу рамки (true(1) - рисовать, false(0) - не рисовать) |
| *void* ***SetBottomBound*** *(BOOL propVal)* | устанавливается рисовать нижнюю границу рамки (true(1) - рисовать, false(0) - не рисовать) |
| **Ориентация компонента** | |
| *BOOL* ***GetHorisontal****()* | опрашивается ориентация компонента (true(1) - горизонтальный, false(0) - вертикальный) |
| *void* ***GetHorisontal****(BOOL propVal)* | устанавливается ориентация компонента (true(1) - горизонтальный, false(0) - вертикальный) |
| **Параметры** | |
| *double* ***GetAmplitude****()* | опрашивается амплитуда индикатора уровня |
| *void* ***SetAmplitude****(double propVal)* | устанавливается амплитуда индикатора уровня |
| *long* ***GetLinLog****()* | опрашивается линейная или логарифмическая шкала по Y. Параметр может принимать значения 0 или 1 (0 – линейный |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| | масштаб, 1 – логарифмический масштаб) |
| *void* ***SetLinLog****(long propVal)* | устанавливается линейная или логарифмическая шкала по Y. Параметр может принимать значения 0 или 1 (0 – линейный масштаб, 1 – логарифмический масштаб) |
| *double* ***GetMemory****()* | опрашивается размер выделяемой памяти |
| *void* ***SetMemory****(double propVal)* | устанавливается размер выделяемой памяти |
| **Методы GdiPlusHorScale.ocx** | |
| *void* ***AboutBox****()* | содержит информацию о программе |

# Глава 6.GdiPlusTextDisp.ocx предназначен для отображения текстово-численной информации

Компонент GdiPlusTextDisp.ocx используется для отображения текстово-численной информации. Применяется в программах измерения параметров сигналов, поступающих на входные каналы анализаторов спектра и плат АЦП и ЦАП (например, в программах вольтметров переменного и постоянного тока, виброметр).

## 6.1.Установка компонента GdiPlusTextDisp.ocx

Для работы с модулем GdiPlusTextDisp его необходимо сначала установить в программу. При загрузке программы необходимо загрузить с помощью компонента необходимую программу, разместить окно программы на экране в удобном месте, установить необходимые параметры в программе и затем по событиям, происходящим от программы считывать данные из программы.

Для установки компонента GdiPlusTextDisp в проект *Microsoft Visual Studio 2010*, необходимо кликнуть правой кнопкой мыши на форме диалога и из появившегося меню выбрить пункт *Insert ActiveX Control…*(рисунок 1.1)

© ООО "ЭТМС" 1992-2024. Все права защищены

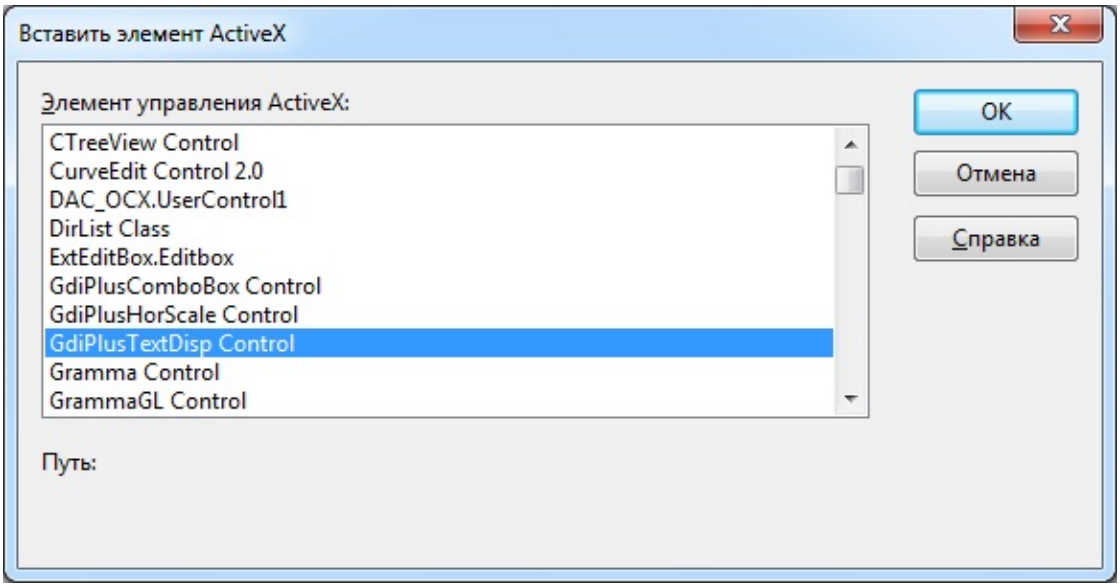



Рисунок 1.1

Из появившегося диалогового окна следует выбрать компонент *GdiPlusTextDisp Control* и нажать *OK* (рисунок 1.2). После этого компонент *GdiPlusTextDisp.ocx* появится на форме диалога (рисунок 1.2).

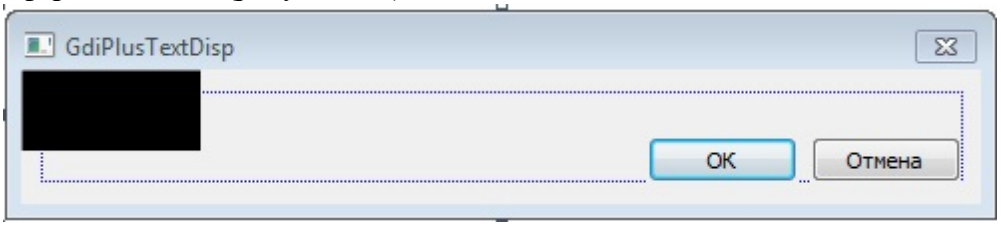



Рисунок 1.2

Одна установленная на форме компонента позволяет управлять одной программой. При необходимости управлять несколькими программами, необходимо установить на форму несколько компонент. Для того чтобы компонента не была видна на форме во время выполнения программы, необходимо установить свойство *Visible* равным *False*.

## 6.2.Интерфейс компонента GdiPlusTextDisp.ocx




© ООО "ЭТМС" 1992-2024. Все права защищены

Компонент позволяет:

- отображать численную и текстовую информацию, в т.ч. на русском языке;
- задавать такие параметры шрифта, как стиль и размер;
- устанавливать цвет фона и шрифта;
- выбирать для каких сторон компонента необходимо отображать рамку;

Свойство GdiPlusTextDisp позволяет комбинировать компоненты в общее информационное поле. На рисунке показан пример программы Вольтметр переменного тока, в которой используется 3 компонента GdiPlusTextDisp ("скз 0.000", "пик 0.000" и "мВ") для отображения уровня среднеквадратичного и пикового значений сигнала, а также единиц измерения

- устанавливать индикатор с жёстко заданными местами для цифр (1(true) - Да, 0(false) - нет) с количеством знаков до запятой и после запятой.
- выбирать название шрифта;

## 6.3.Полное описание свойств, методов и событий GdiPlusTextDisp.ocx

Следует отметить, что компонент автомасштабируется при изменении его размеров, поэтому необходимо следить за тем, чтобы высота компонента в приложении была не менее 80 пикселей, иначе он может отображаться некорректно.

### Свойства GdiPlusTextDisp.ocx

Свойства можно задавать на этапе проектирования и во время выполнения программы. На этапе проектирования при выборе компонента и при нажатии клавиши <F4> появляется окно свойств. Эти свойства можно устанавливать на этапе проектирования программы.

Цвета задаются при помощи макроса *RGB(RED, GREEN, BLUE)*. Значения *RED*, *GREEN*, *BLUE* задают интенсивности цветов красного, зеленого и синего. Интенсивности изменяются в пределах от 0 до 255.

*OLE_COLOR* ***backColor*** – цвет фона

*OLE_COLOR* ***ColorText*** – цвет текста

© ООО "ЭТМС" 1992-2024. Все права защищены

Например, для изменении фона графика с желтого на зеленый, могут быть записаны числа:

*OLE_COLOR a = 0xFF00;*
*OLE_COLOR b = 0xFFFF;*

Используются для создания эффекта объемного элемента.

*VARIANT_BOOL* ***LeftBound*** – рисовать левую границу рамки.

*VARIANT_BOOL* ***RightBound*** – рисовать правую границу рамки.

*VARIANT_BOOL* ***TopBound*** – верхнюю границу рамки.

*VARIANT_BOOL* ***BottomBound*** – рисовать нижнюю границу рамки.

Изменение видимости отображения границ :
true - Видимость отображения границ.
false - Видимость отображения границ - выключена.

*BSTR* ***TextString*** – текстовая строка

*long* ***TextSize*** – размер текста.

*long* ***TextFontType*** – стиль шрифта.

*long* ***AlignString*** – выровнять строку.

*VARIANT_BOOL* ***SimpleBtn*** – рисовать как кнопку (1(true) - Да, 0(false) - нет).

*VARIANT_BOOL* ***Positioning*** – индикатор с жёстко заданными местами для цифр (1(true) - Да, 0(false) - нет).

*LONG* ***PrefixNumber*** – количество знаков до запятой.

*LONG* ***PostfixNumber*** – количество знаков после запятой.

*BSTR* ***TextFontName*** – устанавливать название шрифта.

© ООО "ЭТМС" 1992-2024. Все права защищены

### Методы GdiPlusTextDisp.ocx

*void* ***AboutBox****()* –содержит информацию о программе.

### События от компоненты GdiPlusTextDisp.ocx

*void* ***LMouseUp****(void)*– событие возникает при отжатии кнопки манипулятора типа «мышь», при нахождении мыши над текстом.

## 6.4.Краткий перечень функций, используемые GdiPlusTextDisp.ocx

| **Свойства GdiPlusTexDisp.ocx** | |
|---|---|
| **Цвета** | |
| *long* ***GetbackColor*** *()* | опрашивается цвет фона |
| *void* ***GetbackColor*** *(unsigned long propVal)* | устанавливается цвет фона |
| *long* ***GetColorText*** *()* | опрашивается цвет текста |
| *void* ***SetColorText*** *(unsigned long propVal)* | устанавливается цвет текста |
| **Отображение границ** | |
| *BOOL* ***GetLeftBound****()* | опрашивается рисовать левую границу рамки (true(1) - рисовать, false(0) - не рисовать) |
| *void* ***SetLeftBound****(BOOL propVal)* | устанавливается рисовать левую границу рамки (true(1) - рисовать, false(0) - не рисовать) |
| *BOOL* ***GetRightBound*** *()* | опрашивается рисовать правую границу рамки (true(1) - рисовать, false(0) - не рисовать) |
| *void* ***SetRightBound*** *(BOOL propVal)* | устанавливается рисовать правую границу рамки (true(1) - рисовать, false(0) - не рисовать) |
| *BOOL* ***GetTopBound*** *()* | опрашивается рисовать верхнюю границу рамки (true(1) - рисовать, |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| | false(0) - не рисовать) |
| *void* ***SetTopBound*** *(BOOL propVal)* | устанавливается рисовать верхнюю границу рамки (true(1) - рисовать, false(0) - не рисовать) |
| *BOOL* ***GetBottomBound*** *()* | опрашивается рисовать нижнюю границу рамки (true(1) - рисовать, false(0) - не рисовать) |
| *void* ***SetBottomBound*** *(BOOL propVal)* | устанавливается рисовать нижнюю границу рамки (true(1) - рисовать, false(0) - не рисовать) |
| **Параметры** | |
| *CString* ***GetTextString*** *()* | опрашивается текстовая строка |
| *void* ***SetTextString*** *(CString propVal)* | устанавливается текстовая строка |
| *long* ***GetTextSize****()* | опрашивается размер шрифта |
| *void* ***SetTextSize****(long propVal)* | устанавливается размер шрифта |
| *long* ***GetTextFontType*** *()* | опрашивается тип шрифта |
| *void* ***SetTextFontType****(long propVal)* | устанавливается тип шрифта |
| *long* ***GetAlignString****()* | опрашивается выровнять строку. |
| *void* ***SetAlignString*** *(long propVal)* | устанавливается выровнять строку. |
| *BOOL* ***GetSimpleBtn****()* | опрашивается рисовать как кнопку (1(true) - Да, 0(false) - нет) |
| *void* ***SetSimpleBtn****(BOOL propVal)* | устанавливается рисовать как кнопку (1(true) - Да, 0(false) - нет) |
| *BOOL* ***GetPositioning****()* | опрашивается индикатор с жёстко заданными местами для цифр (1(true) - Да, 0(false) - нет). |
| *void* ***SetPositioning****(BOOL propVal)* | устанавливается индикатор с жёстко заданными местами для цифр (1(true) - Да, 0(false) - нет). |
| *long* ***GetPrefixNumber*** *()* | опрашивается количество знаков до запятой. |
| *void* ***SetPrefixNumber*** *(long propVal)* | устанавливается количество знаков до запятой. |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *long* ***GetPostfixNumber*** *()* | опрашивается количество знаков после запятой. |
| *void* ***SetPrePostfixNumber*** *(long propVal)* | устанавливается количество знаков после запятой. |
| *CString* ***GetTextFontName*** *()* | опрашивается название шрифта. |
| *void* ***SetTextString*** *(CString propVal)* | устанавливается название шрифта. |
| **Методы GdiPlusTexDisp.ocx** | |
| *void* ***AboutBox****()* | содержит информацию о программе |
| **События от компоненты GdiPlusTexDisp.ocx** | |
| *void* ***LMouseUp****(void)* | событие возникает при отжатии кнопки манипулятора типа «мышь», при нахождении мыши над текстом. |

# Глава 7.ZIndicator.ocx предназначен для вывода целых и дробных десятичных числовых величин

Компонент **ZIndicator.ocx** предназначен для вывода целых и дробных десятичных числовых величин и изменения их значений. Изменение значения может происходить как вводом нового значения целиком, так и изменением значения одного из разрядов. Применяется в программах измерения параметров сигналов, поступающих на входные каналы анализаторов спектра и плат АЦП и ЦАП (например, в программах источников питания и генератора сигналов).

## 7.1.Установка компонента ZIndicator.ocx

Для работы с модулем ZIndicator его необходимо сначала установить в программу. При загрузке программы необходимо загрузить с помощью компонента необходимую программу, разместить окно программы на экране в удобном месте, установить необходимые параметры в программе и затем по событиям, происходящим от программы считывать данные из программы.

Для установки компонента ZIndicator в проект *Microsoft Visual Studio 2010*, необходимо кликнуть правой кнопкой мыши на форме диалога и из появившегося меню выбрить пункт *Insert ActiveX Control*...(рисунок 1.1)

© ООО "ЭТМС" 1992-2024. Все права защищены

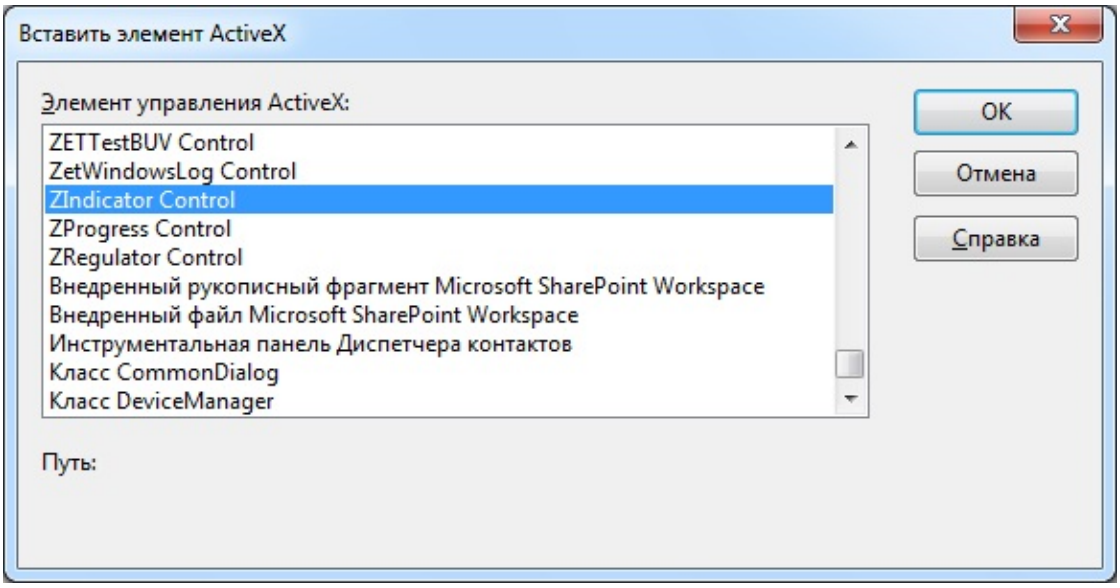



Рисунок 1.1

Из появившегося диалогового окна следует выбрать компонент *ZIndicator Control* и нажать *OK* (рисунок 1.2). После этого компонент *ZIndicator.ocx* появится на форме диалога (рисунок 1.2).

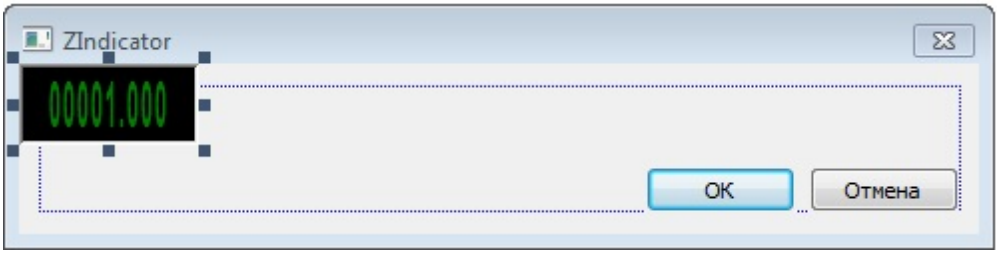



Рисунок 1.2

Одна установленная на форме компонента позволяет управлять одной программой. При необходимости управлять несколькими программами, необходимо установить на форму несколько компонент. Для того чтобы компонента не была видна на форме во время выполнения программы, необходимо установить свойство *Visible* равным *False*.

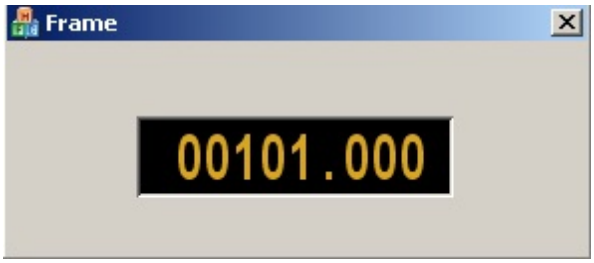
## 7.2.Интерфейс компонента ZIndicator.ocx

**Назначение**

Элемент предназначен для вывода целых и дробных десятичных числовых величин и изменения их значений. Изменение значения может происходить как вводом нового значения целиком, так и изменением значения одного из разрядов.

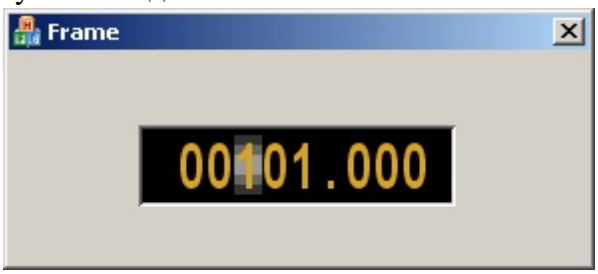
Внешний вид компонента:

© ООО "ЭТМС" 1992-2024. Все права защищены




**Использование компонента**

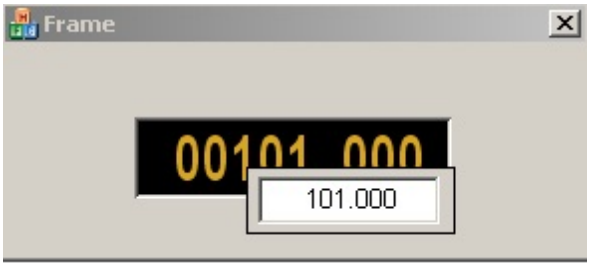
При нахождении курсора мыши над одной из цифр индикатора, данная цифра становится активной и визуально подсвечивается:




При прокрутке колеса мыши над компонентом вверх (вниз) значение индикатора увеличивается (уменьшается) на единицу текущего активного разряда.

Также можно редактировать значение текущего активного разряда вводом цифр с клавиатуры.

При двойном щелчке на компоненте открывается форма редактирования, позволяющая ввести значение индикатора с клавиатуры.




**ZIndicator.ocx** используется в таких программах, как "Генератор сигналов", "BP05MC", а также в источниках питания "LPS305.exe", "PPE3323.exe", "PSM2010.exe".

## 7.3.Полное описание свойств, методов и событий ZIndicator.ocx

Следует отметить, что компонент автомасштабируется при изменении его размеров, поэтому необходимо следить за тем, чтобы высота компонента в приложении была не менее 80 пикселей, иначе он может отображаться некорректно.

© ООО "ЭТМС" 1992-2024. Все права защищены

## Свойства ZIndicator.ocx

Свойства можно задавать на этапе проектирования и во время выполнения программы. На этапе проектирования при выборе компонента и при нажатии клавиши <F4> появляется окно свойств. Эти свойства можно устанавливать на этапе проектирования программы.

Цвета задаются при помощи макроса *RGB(RED, GREEN, BLUE)*. Значения *RED*, *GREEN*, *BLUE* задают интенсивности цветов красного, зеленого и синего. Интенсивности изменяются в пределах от 0 до 255.

*OLE_COLOR* ***ClrBgr*** – цвет фона

*OLE_COLOR* ***ClrText*** – цвет текста

Например, для изменении фона графика с желтого на зеленый, могут быть записаны числа:

```
OLE_COLOR a = 0xFF00;
OLE_COLOR b = 0xFFFF;
```

Используются для создания эффекта объемного элемента.

*VARIANT_BOOL* ***BevelLeft*** – рисовать левую границу рамки.

*VARIANT_BOOL* ***BevelRight*** – рисовать правую границу рамки.

*VARIANT_BOOL* ***BevelUp*** – рисовать верхнюю границу рамки.

*VARIANT_BOOL* ***BevelDown*** – рисовать нижнюю границу рамки.

Изменение видимости отображения границ :
true - Видимость отображения границ.
false - Видимость отображения границ - выключена.

*VARIANT_BOOL* ***Bold*** – включить/выключить утолщенный шрифт,

© ООО "ЭТМС" 1992-2024. Все права защищены

*VARIANT_BOOL* ***Italic*** – включить/выключить наклонный шрифт.

*BSTR* ***FontFace*** – название шрифта, как строковая переменная.

*FLOAT* ***Max*** – максимальное устанавливаемое значение индикатора.

FLOAT ***Min*** – минимальное устанавливаемое значение индикатора.

FLOAT ***Value*** – текущее значение индикатора.

SHORT ***NumCnt*** – общее количество отображаемых десятичных знаков.

*SHORT* ***NumDivCnt*** – количество отображаемых десятичных знаков после запятой.

## Методы ZIndicator.ocx

Методы можно использовать только во время выполнения программы.

*void* ***AboutBox****()* –содержит информацию о программе.

*void* ***SetIndValue****(FLOAT value)* - значение выводимое на индикатор.

*long* ***EnaWindow****(LONG enable)* - определяется включенность, а если выключен, то окрашивается другим цветом.

*long* ***StepUpDown****(FLOAT step)* - возвращает шаг изменения индикатора.

*FLOAT* ***GetStepUpDown****(void)* - вычитывает шаг изменения индикатора

## События от компоненты ZIndicator.ocx

*void* ***Modify****(FLOAT fValue)* – событие возникает при изменении пользователем значения индикатора.

Параметры:
fValue – установленное значение.

© ООО "ЭТМС" 1992-2024. Все права защищены

## 7.4.Краткий перечень функций, используемые ZIndicator.ocx

| Свойства ZIndicator.ocx | |
|---|---|
| **Цвета** | |
| *long* ***GetClrBgr*** *()* | опрашивается цвет фона |
| *void* ***GetClrBgr*** *(unsigned long propVal)* | устанавливается цвет фона |
| *long* ***GetClrText*** *()* | опрашивается цвет текста |
| *void* ***SetClrText*** *(unsigned long propVal)* | устанавливается цвет текста |
| **Отображение рамки** | |
| *BOOL* ***GetLeftBound****()* | опрашивается рисовать левую границу рамки (true(1) - рисовать, false(0) - не рисовать) |
| *void* ***SetLeftBound****(BOOL propVal)* | устанавливается рисовать левую границу рамки (true(1) - рисовать, false(0) - не рисовать) |
| *BOOL* ***GetRightBound*** *()* | опрашивается рисовать правую границу рамки (true(1) - рисовать, false(0) - не рисовать) |
| *void* ***SetRightBound*** *(BOOL propVal)* | устанавливается рисовать правую границу рамки (true(1) - рисовать, false(0) - не рисовать) |
| *BOOL* ***GetTopBound*** *()* | опрашивается рисовать верхнюю границу рамки (true(1) - рисовать, false(0) - не рисовать) |
| *void* ***SetTopBound*** *(BOOL propVal)* | устанавливается рисовать верхнюю границу рамки (true(1) - рисовать, false(0) - не рисовать) |
| *BOOL* ***GetBottomBound*** *()* | опрашивается рисовать нижнюю границу рамки (true(1) - рисовать, false(0) - не рисовать) |
| *void* ***SetBottomBound*** *(BOOL propVal)* | устанавливается рисовать нижнюю границу рамки (true(1) - рисовать, false(0) - не рисовать) |
| **Параметры шрифта** | |
| *BOOL* ***GetBold****()* | опрашивается включить/выключить утолщенный шрифт (1(true) - Да, 0(false) - нет) |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *void* ***SetBold****(BOOL propVal)* | устанавливается включить/выключить утолщенный шрифт (1(true) - Да, 0(false) - нет) |
| *BOOL* ***GetItalic****()* | опрашивается включить/выключить наклонный шрифт (1(true) - Да, 0(false) - нет) |
| *void* ***SetItalic****(BOOL propVal)* | устанавливается включить/выключить наклонный шрифт (1(true) - Да, 0(false) - нет) |
| *CString* ***GetFontFace****()* | опрашивается название шрифта, как строковая переменная |
| *void* ***SetFontFace****(CString propVal)* | устанавливается название шрифта, как строковая переменная |
| **Параметры управления значением индикатора** | |
| *float* ***GetMax****()* | опрашивается максимальное значение индикатора |
| *void* ***SetMax****(float propVal)* | устанавливается максимальное значение индикатора |
| *float* ***GetMin****()* | опрашивается минимальное значение индикатора |
| *void* ***SetMin****(float propVal)* | устанавливается минимальное значение индикатора |
| *float* ***GetValue*** *()* | опрашивается текущее значение индикатора |
| *void* ***SetValue****(float propVal)* | устанавливается текущее значение индикатора |
| *short* ***GetNumCnt****()* | опрашивается общее количество отображаемых десятичных знаков |
| *void* ***SetNumCnt****(short propVal)* | устанавливается общее количество отображаемых десятичных знаков |
| *short* ***GetNumDivCnt****()* | опрашивается количество отображаемых десятичных знаков после запятой |
| *void* ***SetNumDivCnt****(short propVal)* | устанавливается количество отображаемых десятичных знаков после запятой |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Методы ZIndicator.ocx | |
|---|---|
| *void **AboutBox**()* | содержит информацию о программе |
| *void **SetIndValue**(FLOAT value)* | значение выводимое на индикатор |
| *long **EnaWindow**(LONG enable)* | определяется включенность, а если выключен, то окрашивается другим цветом |
| *long **StepUpDown**(FLOAT step)* | возвращает шаг изменения индикатора |
| *FLOAT **GetStepUpDown**(void)* | вычитывает шаг изменения индикатора |

© ООО "ЭТМС" 1992-2024. Все права защищены

| События от компоненты ZIndicator.ocx | |
|---|---|
| *void* ***Modify****(FLOAT fValue)* | событие возникает при при изменении пользователем значения индикатора. |

# Глава 8.TextDisp.ocx предназначен для отображения текстово-численной информации (уст.)

Компонент **TextDisp.ocx** используется для отображения текстово-численной информации (устаревший). Применяется в программах измерения параметров сигналов, поступающих на входные каналы анализаторов спектра и плат АЦП и ЦАП (например, в программах частотомера, селективного вольтметра, фазометра, энкодера, тахометра, тензометра, термометра термопары, термометра термосопротивления).

## 8.1.Установка компонента TextDisp.ocx

Для работы с модулем TextDisp его необходимо сначала установить в программу. При загрузке программы необходимо загрузить с помощью компонента необходимую программу, разместить окно программы на экране в удобном месте, установить необходимые параметры в программе и затем по событиям, происходящим от программы считывать данные из программы.

Для установки компонента TextDisp в проект *Microsoft Visual Studio 2010*, необходимо кликнуть правой кнопкой мыши на форме диалога и из появившегося меню выбрить пункт *Insert ActiveX Control…*(рисунок 1.1)

© ООО "ЭТМС" 1992-2024. Все права защищены

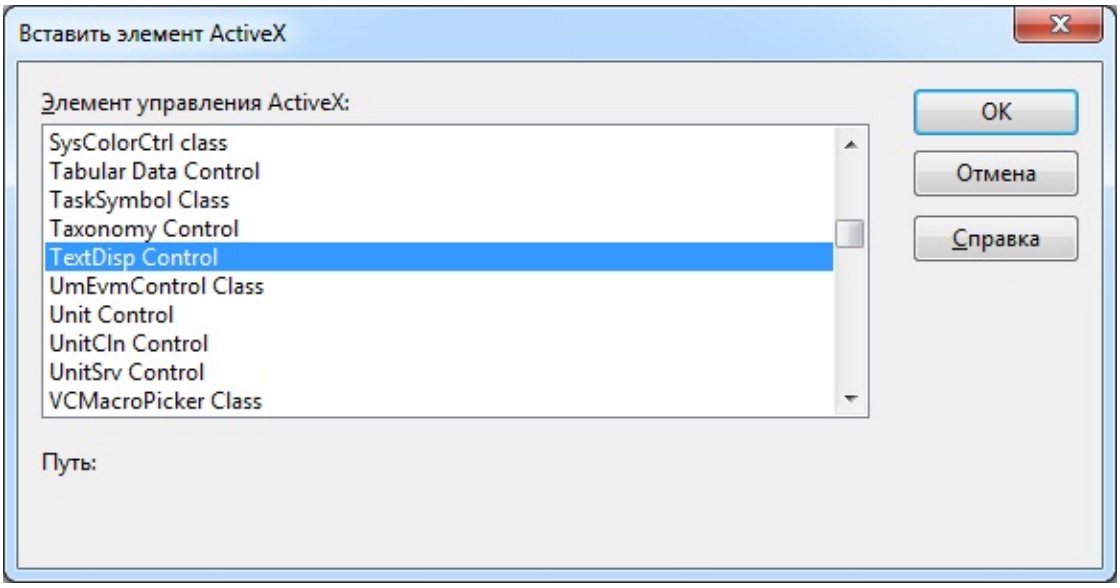



Рисунок 1.1

Из появившегося диалогового окна следует выбрать компонент *TextDisp Control* и нажать *OK* (рисунок 1.2). После этого компонент *TextDisp.ocx* появится на форме диалога (рисунок 1.2).

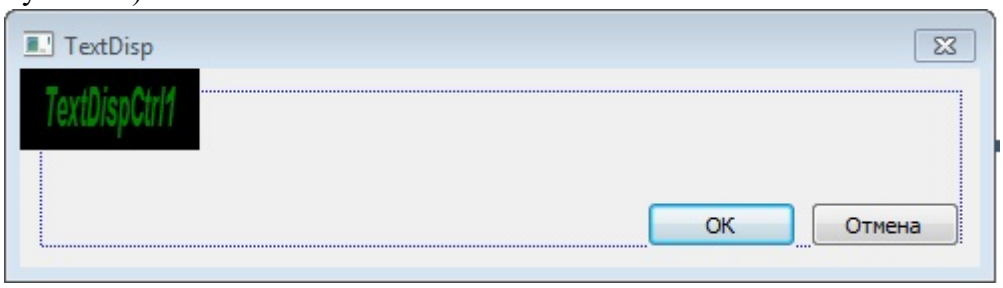
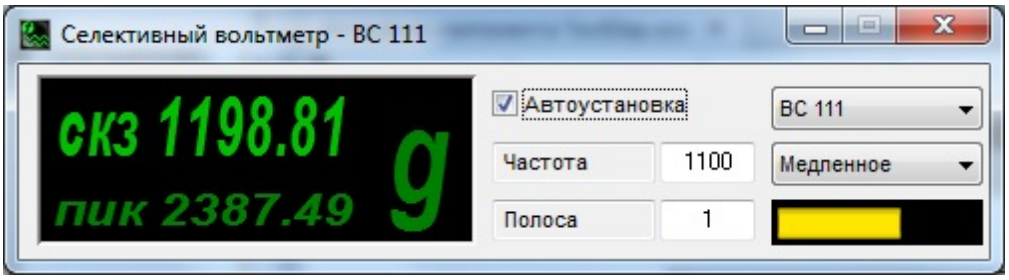



Рисунок 1.2

Одна установленная на форме компонента позволяет управлять одной программой. При необходимости управлять несколькими программами, необходимо установить на форму несколько компонент. Для того чтобы компонента не была видна на форме во время выполнения программы, необходимо установить свойство *Visible* равным *False*.

## 8.2.Интерфейс компонента TextDisp.ocx

© ООО "ЭТМС" 1992-2024. Все права защищены




Компонент позволяет:

- отображать численную и текстовую информацию, в т.ч. на русском языке;
- задавать такие параметры шрифта, как жирное и курсивное начертание;
- устанавливать цвет фона и шрифта;
- выбирать для каких сторон компонента необходимо отображать рамку;

последнее свойство TextDisp позволяет комбинировать компоненты в общее информационное поле. На рисунке показан пример программы Вольтметр переменного тока, в которой используется 3 компонента TextDisp ("скз 1198.81", "пик 2387.49" и "g") для отображения уровня среднеквадратичного и пикового значений сигнала, а также единиц измерения.

**TextDisp.ocx** применяется практически во всех программах измерения параметров сигналов, поступающих на входные каналы анализаторов спектра и плат АЦП и ЦАП (например, в программах селективного вольтметра, частотомера, фазометра, тензометра, торсиографа, энкодера, термометра ТС, термометра ТП и т.д.)

## 8.3.Полное описание свойств, методов и событий TextDisp.ocx

Следует отметить, что компонент автомасштабируется при изменении его размеров, поэтому необходимо следить за тем, чтобы высота компонента в приложении была не менее 80 пикселей, иначе он может отображаться некорректно.

### Свойства GdiPlusTextDisp.ocx

Свойства можно задавать на этапе проектирования и во время выполнения программы. На этапе проектирования при выборе компонента и при нажатии клавиши <F4> появляется окно свойств. Эти свойства можно устанавливать на этапе проектирования программы.

© ООО "ЭТМС" 1992-2024. Все права защищены

Цвета задаются при помощи макроса *RGB(RED, GREEN, BLUE)*. Значения *RED*, *GREEN*, *BLUE* задают интенсивности цветов красного, зеленого и синего. Интенсивности изменяются в пределах от 0 до 255.

*OLE_COLOR* ***ClrFon*** – цвет фона

*OLE_COLOR* ***ClrDig*** – цвет текста

Например, для изменении фона графика с желтого на зеленый, могут быть записаны числа:

```
OLE_COLOR a = 0xFF00;
OLE_COLOR b = 0xFFFF;
```

Используются для создания эффекта объемного элемента.

*long* ***Rleft*** - рисовать левую границу рамки.

*long* ***Rright -*** рисовать правую границу рамки.

*long* ***Rup -*** рисовать верхнюю границу рамки.

*long* ***Rdown -*** рисовать нижнюю границу рамки.

*long* ***GetItallic -*** стиль шрифта: курсив

*long* ***GetBold -*** стиль шрифта: жирный

*BSTR* ***GetCaption -*** название компонента

*BSTR* ***BstrText -*** строка в формате unicode.

## Методы TextDisp.ocx

Методы можно использовать только во время выполнения программы.

*void* ***AboutBox()*** –содержит информацию о программе.

*long* ***Text****(BSTR* stroka)* – содержит текстовую строку.

© ООО "ЭТМС" 1992-2024. Все права защищены

*BYTE* ***SetText****(BSTR TexT)* – содержит текстовую строку в Unicod.

*void* ***SetMinTextLength****(SHORT Value)* –устанавливает минимальную длину текста.

*SHORT* ***GetMinTextLength****(void)* – получить минимальную длину текста.

## 8.4.Краткий перечень функций, используемые TextDisp.ocx

| **Свойства TexDisp.ocx** | |
|---|---|
| **Цвета** | |
| *long* ***GetClrFon*** *()* | опрашивается цвет фона |
| *void* ***SetClrFon*** *(unsigned long propVal)* | устанавливается цвет фона |
| *long* ***GetClrDig*** *()* | опрашивается цвет текста |
| *void* ***SetClrDig*** *(unsigned long propVal)* | устанавливается цвет текста |
| **Отображение границ** | |
| *long* ***GetRleft****()* | опрашивается рисовать левую границу рамки (true(1) - рисовать, false(0) - не рисовать) |
| *void* ***SetRleft*** *(BOOL propVal)* | устанавливается рисовать левую границу рамки (true(1) - рисовать, false(0) - не рисовать) |
| *long* ***GetRright*** *()* | опрашивается рисовать правую границу рамки (true(1) - рисовать, false(0) - не рисовать) |
| *void* ***SetRright*** *(BOOL propVal)* | устанавливается рисовать правую границу рамки (true(1) - рисовать, false(0) - не рисовать) |
| *long* ***GetRup*** *()* | опрашивается рисовать верхнюю границу рамки (true(1) - рисовать, false(0) - не рисовать) |
| *void* ***SetRup*** *(BOOL propVal)* | устанавливается рисовать верхнюю границу рамки (true(1) - рисовать, false(0) - не рисовать) |
| *long* ***GetRdown*** *()* | опрашивается рисовать нижнюю границу рамки (true(1) - рисовать, false(0) - не рисовать) |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *void* ***SetRdown*** *(BOOL propVal)* | устанавливается рисовать нижнюю границу рамки (true(1) - рисовать, false(0) - не рисовать) |
| **Параметры** | |
| *long* ***GetItallic****()* | опрашивается стиль шрифта: курсив |
| *void* ***SetItallic****(long propVal)* | устанавливается стиль шрифта: курсив |
| *long* ***GetBold*** *()* | опрашивается стиль шрифта: жирный |
| *void* ***SetBold*** *(long propVal)* | устанавливается стиль шрифта: жирный |
| *CString* ***GetCaption*** *()* | опрашивается название компонента |
| *void* ***SetCaption*** *(CString propVal)* | устанавливается название компонента |
| *CString* ***GetBstrText****()* | опрашивается строка в формате unicode. |
| *void* ***SetBstrText****(CString propVal)* | устанавливается строка в формате unicode. |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Методы TexDisp.ocx | |
|---|---|
| *void **AboutBox**()* | содержит информацию о программе |
| *long **Text**(BSTR* stroka)* | содержит текстовую строку. |
| *BYTE **SetText**(BSTR TexT)* | содержит текстовую строку в Unicod |
| *void **SetMinTextLength** (SHORT Value)* | устанавливает минимальную длину текста. |
| *SHORT **GetMinTextLength** (void)* | получить минимальную длину текста. |

# Глава 9.ColScale.ocx задает соответствие цвета графика уровню сигнала при отображении спектрограмм

Компонента **ColScale.ocx** задает соответствие цвета графика уровню сигнала при отображении спектрограмм и коррелограмм и используется в таких программах, как "Узкополосный спектр", "Долеоктавный спектр", "Взаимный узкополосный спектр", "Взаимный долеоктавный спектр", "Взаимный корреляционный анализ".

## 9.1.Установка компонента ColScale.ocx

Для работы с модулем ColScale его необходимо сначала установить в программу. При загрузке программы необходимо загрузить с помощью компонента необходимую программу, разместить окно программы на экране в удобном месте, установить необходимые параметры в программе и затем по событиям, происходящим от программы считывать данные из программы.

Для установки компонента ColScale в проект *Microsoft Visual Studio 2010*, необходимо кликнуть правой кнопкой мыши на форме диалога и из появившегося меню выбрить пункт *Insert ActiveX Control*...(рисунок 1.1)

© ООО "ЭТМС" 1992-2024. Все права защищены

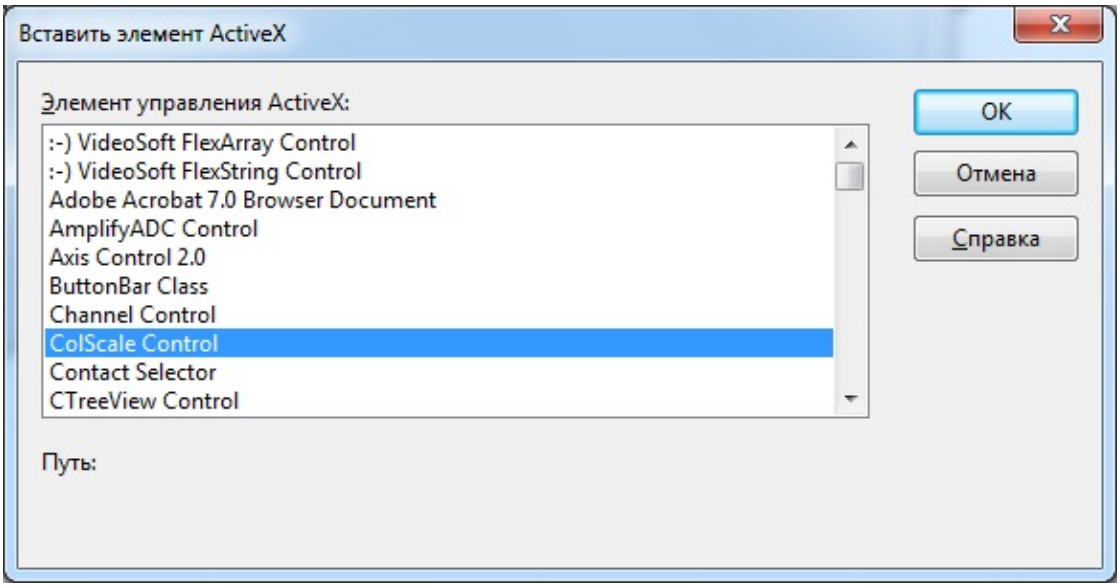



Рисунок 1.1

Из появившегося диалогового окна следует выбрать компонент *ColScale Control* и нажать *OK* (рисунок 1.2). После этого компонент *ColScale.ocx* появится на форме диалога (рисунок 1.2).

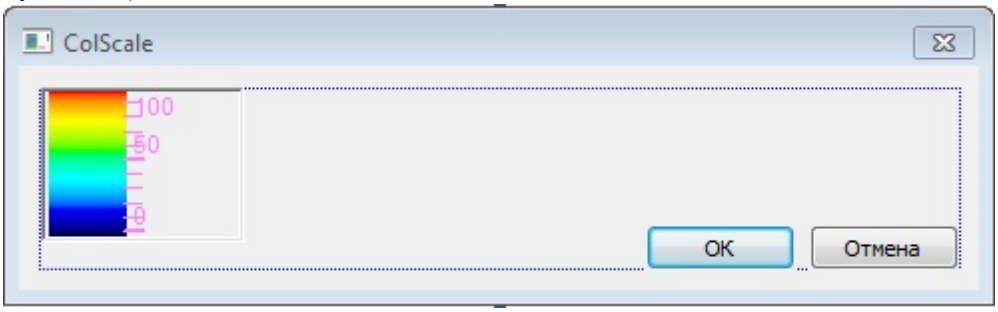
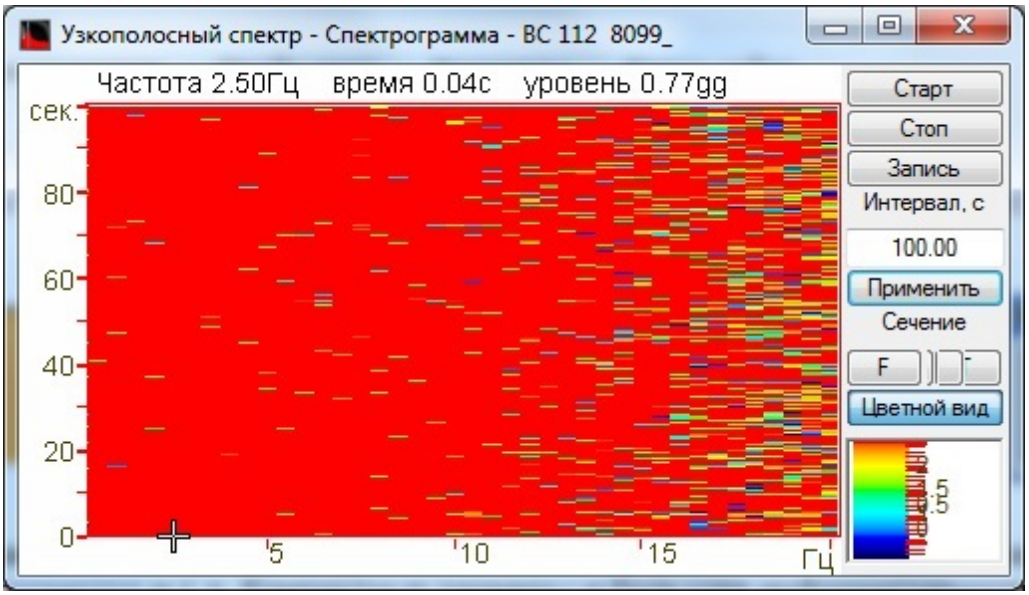



Рисунок 1.2

Одна установленная на форме компонента позволяет управлять одной программой. При необходимости управлять несколькими программами, необходимо установить на форму несколько компонент. Для того чтобы компонента не была видна на форме во время выполнения программы, необходимо установить свойство *Visible* равным *False*.

## 9.2.Интерфейс компонента ColScale.ocx

© ООО "ЭТМС" 1992-2024. Все права защищены




Компонент **ColScale** дополняет компонент Gramma в программах спектрального анализа сигналов.
**ColScale** задает соответствие цвета графика уровню сигнала при отображении спектрограмм и коррелограмм и используется в таких программах, как "Узкополосный спектр", "Долеоктавный спектр", "Взаимный узкополосный спектр", "Взаимный долеоктавный спектр", "Взаимный корреляционный анализ", "Вейвлет анализ", "Спектр со сверхразрешением"и "Просмотр результатов".

## 9.3.Полное описание свойств, методов и событий ColScale.ocx

Следует отметить, что компонент автомасштабируется при изменении его размеров, поэтому необходимо следить за тем, чтобы высота компонента в приложении была не менее 80 пикселей, иначе он может отображаться некорректно.

### Свойства ColScale.ocx

Свойства можно задавать на этапе проектирования и во время выполнения программы. На этапе проектирования при выборе компонента и при нажатии клавиши <F4> появляется окно свойств. Эти свойства можно устанавливать на этапе проектирования программы.

Цвета задаются при помощи макроса *RGB(RED, GREEN, BLUE)*. Значения *RED*, *GREEN*, *BLUE* задают интенсивности цветов красного, зеленого и синего. Интенсивности изменяются в пределах от 0 до 255.

© ООО "ЭТМС" 1992-2024. Все права защищены

*OLE_COLOR* ***ClrDig*** – цвет цифр на сетке.

*OLE_COLOR* ***ClrFon*** – цвет фона.

*OLE_COLOR* ***ClrGrd*** – цвет меток.

*OLE_COLOR* ***Black*** – цветное или черно-белое изображение.

0 – цветное изображение.
1 – черно-белое изображение.

Например, для изменении фона графика с желтого на зеленый, могут быть записаны числа:

*OLE_COLOR a = 0xFF00;*
*OLE_COLOR b = 0xFFFF;*

*long* ***AutoColor*** - автомасштабирование

*double* ***MathLY*** – текущее минимальное значение по Y. Нижняя граница отображения данных.

*double* ***MathDY*** – текущий диапазон отображения по Y. Высота отображения данных.

Растяжение или сжатие графиков происходит при помощи указателя вида:  ,  – для вертикальной оси. Сдвинуть графики вверх или вниз можно при помощи указателя  ,  – для вертикальной оси. При этом меняются параметры *MathLY, MathDY.* В пользовательской программе в режиме исполнения можно прочитать текущие параметры области отображения.

## Методы ColScale.ocx

Методы можно использовать только во время выполнения программы.

*long* ***Redraw****(void)* - метод перерисовывает элемент

© ООО "ЭТМС" 1992-2024. Все права защищены

### События от компоненты ColScale.ocx

*void* ***KeyDown*** *(SHORT* KeyCode, SHORT Shift)* – событие возникает при нажатии на клавишу клавиатуры, когда фокус находится на компоненте.

Параметры:
*KeyCode* – код клавиши
*Shift* – код нажатия служебных клавиш <*Ctrl*>,<*Shift*>,<*Alt*>.

*void* ***MouseDown*** *(short Button, short Shift, OLE_XPOS_PIXELS x, OLE_YPOS_PIXELS y)* – событие возникает при нажатии на кнопку манипулятора типа «мышь», при нахождении мыши над графиком.

Параметры:
*Button* – код кнопки мыши
*Shift* – код нажатия служебных клавиш <*Ctrl*>,<*Shift*>,<*Alt*>.
*x* – координата по оси *X* в пикселах (разрешении экрана)
*y* – координата по оси *Y* в пикселах (разрешении экрана)

*void* ***MouseUp*** *(short Button, short Shift, OLE_XPOS_PIXELS x, OLE_YPOS_PIXELS y)* – событие возникает при отжатии кнопки манипулятора типа «мышь», при нахождении мыши над графиком.

Параметры:
*Button* – код кнопки мыши
*Shift* – код нажатия служебных клавиш <*Ctrl*>,<*Shift*>,<*Alt*>.
*x* – координата по оси *X* в пикселях (разрешении экрана)
*y* – координата по оси *Y* в пикселях (разрешении экрана)

void ***MouseWheel*** *(void)* – событие возникает при вращении ролика манипулятора типа «мышь».

## 9.4.Краткий перечень функций, используемые ColScale.ocx

| **Свойства ColScale.ocx** | |
|---|---|
| **Цвета** | |
| *long* ***GetClrDig****()* | опрашивается цвет цифр на сетке |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *void* ***SetClrDig****(unsigned long propVal)* | устанавливается цвета цифр на сетке |
| *long* ***GetClrFon*** *()* | опрашивается цвет фона |
| *void* ***SetClrFon*** *(unsigned long propVal)* | устанавливается цвет фона |
| *long* ***GetClrGrd*** *()* | опрашивается цвет меток |
| *void* ***SetClrGrd*** *(unsigned long propVal)* | устанавливается цвет меток |
| *long* ***GetBack*** *()* | опрашивается цветное или черно-белое изображение (0 – цветное /1 – черно-белое) |
| *void* ***SetBack*** *(unsigned long propVal)* | устанавливается цветное или черно-белое изображение (0 – цветное /1 – черно-белое) |
| **Параметры шкалы** | |
| *long* ***GetAutoColor****()* | опрашивается команда автомасштабирование |
| *void* ***SetAutoColor****(long propVal)* | устанавливается команда автомасштабирование |
| **Область отображении шкалы** | |
| *double* ***GetMathly****()* | опрашивается текущее минимальное значение по Y. Нижняя граница отображения данных |
| *void* ***SetMathly****(double propVal)* | Устанавливается текущее минимальное значение по Y. Нижняя граница отображения данных |
| *double* ***GetMathdy****()* | опрашивается текущий диапазон отображения по Y. Высота отображения данных |
| *void* ***SetMathdy****(double propVal)* | устанавливается текущий диапазон отображения по Y. Высота отображения данных |
| **Методы ColScale.ocx** | |
| *long* ***Redraw****()* | метод перерисовки элемента |
| **События от компоненты ColScale.ocx** | |
| *void* ***KeyDown****(SHORT* KeyCode, SHORT Shift)* | событие возникает при нажатии на клавишу клавиатуры, когда фокус находится на компоненте |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *void* ***MouseDown****(short Button, short Shift, OLE_XPOS_PIXELS x, OLE_YPOS_PIXELS y)* | событие возникает при нажатии на кнопку манипулятора типа «мышь», при нахождении мыши над графиком |
| *void* ***MouseUp****(short Button, short Shift, OLE_XPOS_PIXELS x, OLE_YPOS_PIXELS y)* | событие возникает при отжатии кнопки манипулятора типа «мышь», при нахождении мыши над графиком |
| *void* ***MouseWheel****(void)* | событие возникает при вращении ролика манипулятора типа «мышь» |

# Глава 10.Horizontalslider.ocx для выбора значения посредством перемещения указателя на шкале

Компонент **Horizontalslider.ocx** предназначен для выбора значения посредством перемещения указателя на шкале и последующей передачи значения в канал.

Элемент состоит из двух частей – неподвижного основания и движущегося ползунка. Положение ползунка относительно основания не фиксировано, т.е. он может находиться в любой точке слайдера. Перемещать ползунок возможно двумя способами: кликая мышью по активным областям в начале и конце слайдера и непосредственно перемещением ползунка с зажатой левой клавишей мыши.

## 10.1.Установка компонента Horizontalslider.ocx

Для работы с модулем **Horizontalslider.ocx** его необходимо сначала установить в программу. При загрузке программы необходимо загрузить с помощью компонента необходимую программу, разместить окно программы на экране в удобном месте, установить необходимые параметры в программе и затем по событиям, происходящим от программы считывать данные из программы.

Для установки компонента Horizontalslider в проект *Microsoft Visual Studio 2010*, необходимо кликнуть правой кнопкой мыши на форме диалога и из появившегося меню выбрить пункт *Insert ActiveX Control*...(рисунок 1.1)

© ООО "ЭТМС" 1992-2024. Все права защищены

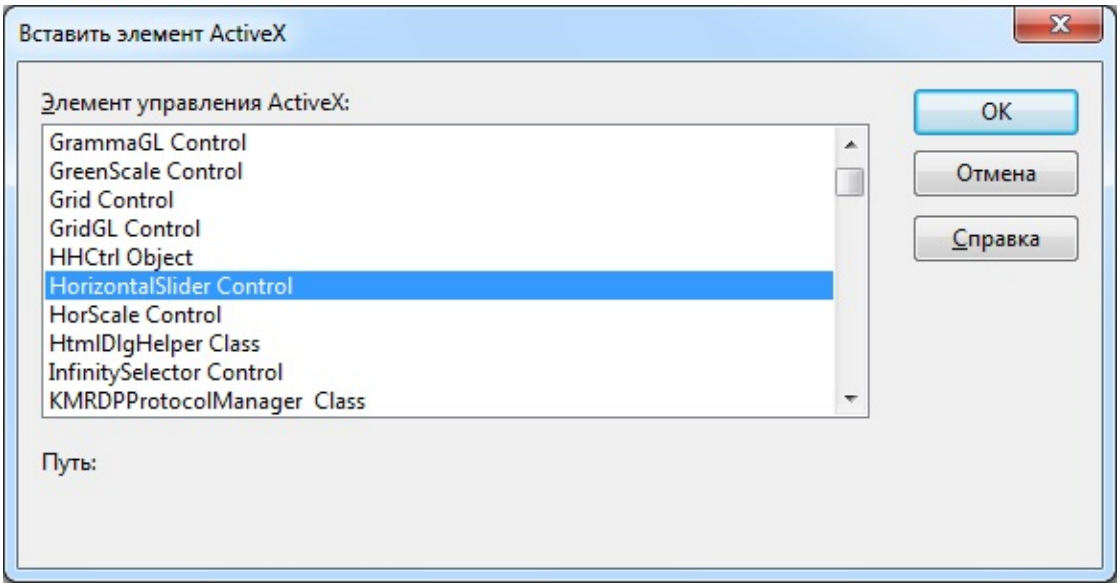



Рисунок 1.1

Из появившегося диалогового окна следует выбрать компонент *Horizontalslider Control* и нажать *OK* (рисунок 1.2). После этого компонент *Horizontalslider.ocx.ocx* появится на форме диалога (рисунок 1.2).

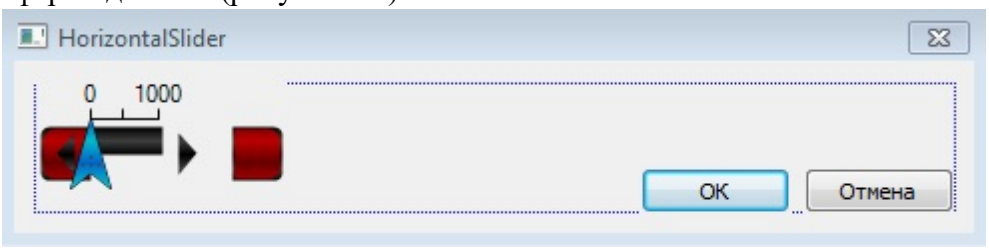
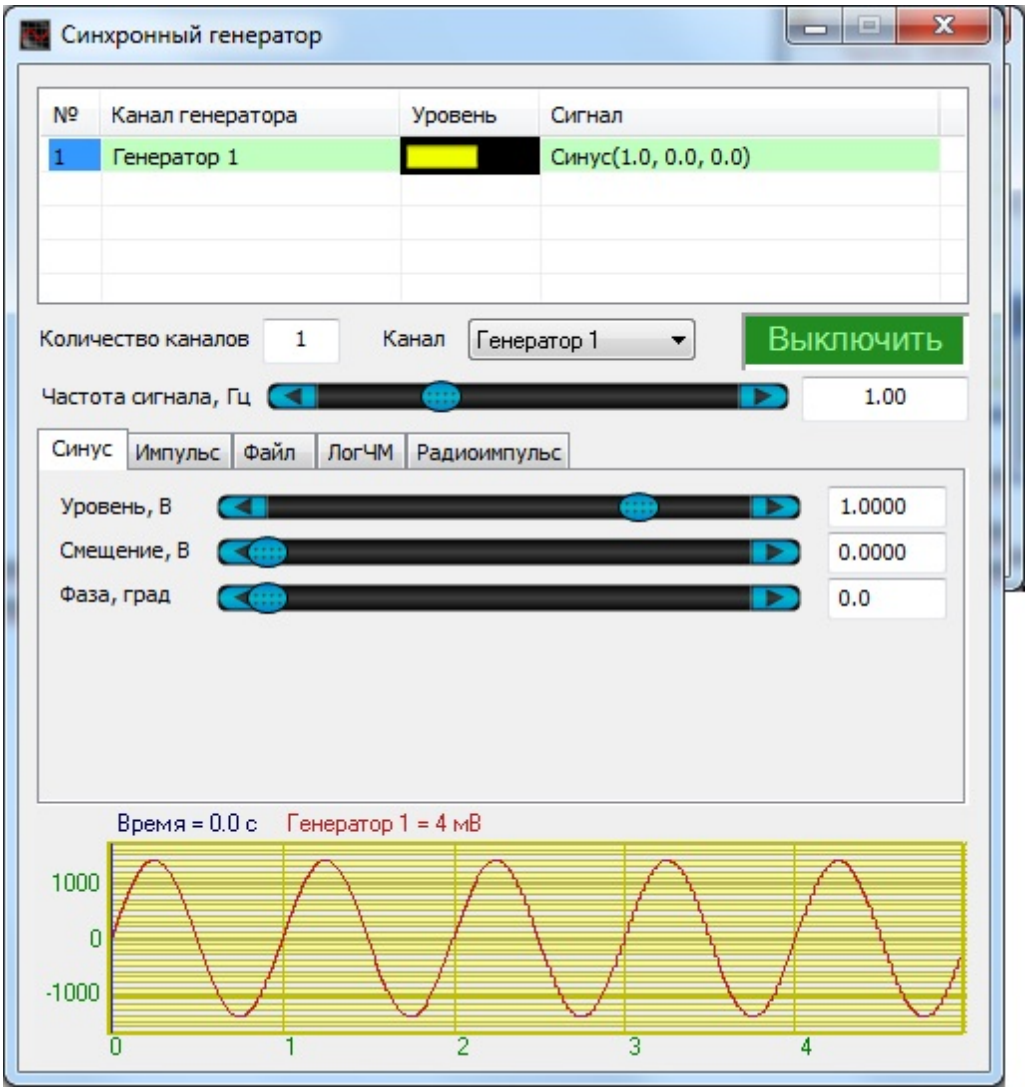



Рисунок 1.2

Одна установленная на форме компонента позволяет управлять одной программой. При необходимости управлять несколькими программами, необходимо установить на форму несколько компонент. Для того чтобы компонента не была видна на форме во время выполнения программы, необходимо установить свойство *Visible* равным *False*.

## 10.2.Интерфейс компонента Horizontalslider.ocx

© ООО "ЭТМС" 1992-2024. Все права защищены




Компонент **Horizontalslider.ocx** предназначен для выбора значения посредством перемещения указателя на шкале и последующей передачи значения в канал.

Элемент состоит из двух частей – неподвижного основания и движущегося ползунка. Положение ползунка относительно основания не фиксировано, т.е. он может находиться в любой точке слайдера. Перемещать ползунок возможно двумя способами: кликая мышью по активным областям в начале и конце слайдера и непосредственно перемещением ползунка с зажатой левой клавишей мыши. **Horizontalslider** используется в таких программах, как "Синхронный генератор", "Формула", "Фильтрация сигналов".

© ООО "ЭТМС" 1992-2024. Все права защищены

## 10.3.Полное описание свойств, методов и событий Horizontalslider.ocx

Следует отметить, что компонент автомасштабируется при изменении его размеров, поэтому необходимо следить за тем, чтобы высота компонента в приложении была не менее 80 пикселей, иначе он может отображаться некорректно.

### Свойства Horizontalslider.ocx

Свойства можно задавать на этапе проектирования и во время выполнения программы. На этапе проектирования при выборе компонента и при нажатии клавиши <F4> появляется окно свойств. Эти свойства можно устанавливать на этапе проектирования программы.

Цвета задаются при помощи макроса *RGB(RED, GREEN, BLUE)*. Значения *RED*, *GREEN*, *BLUE* задают интенсивности цветов красного, зеленого и синего. Интенсивности изменяются в пределах от 0 до 255.

*OLE_COLOR* ***TextColor*** – цвет текста.

*OLE_COLOR* ***BackgroundColor*** – цвет фона.

*OLE_COLOR* ***LineGradientSideColor*** – цвет края полосы прокрутки при заполнении ее градиентом.

*OLE_COLOR* ***LineGradientCenterColor*** – цвет центра полосы прокрутки при заполнении ее градиентом.

*OLE_COLOR* ***EndRectGradientSideColor*** – цвет края концов полосы прокрутки при заполнении их градиентом.

*OLE_COLOR* ***EndRectGradientCenterColor*** – цвет центра концов полосы прокрутки при заполнении их градиентом.

*OLE_COLOR* ***CursorGradientSideColor*** – цвет края курсора при заполнении его градиентом.

© ООО "ЭТМС" 1992-2024. Все права защищены

*OLE_COLOR* ***CursorGradientCenterColor*** – цвет центра курсора при заполнении его градиентом.

Например, для изменении фона графика с желтого на зеленый, могут быть записаны числа:

```
OLE_COLOR a = 0xFF00;
OLE_COLOR b = 0xFFFF;
```

*long* ***FontSize*** – размер шрифта

*BSTR* ***FontName*** – название шрифта

*long* ***TextStyle*** – стиль текста

*DOUBLE* ***StartValue*** – начальное значение шкалы

*DOUBLE* ***EndValue*** – конечное значение шкалы

*DOUBLE* ***CurrentValue*** – текущее значение

*long* ***EndRectWidth*** – ширина концов полосы прокрутки

*long* ***CursorWidth*** – ширина указателя

*long* ***CursorForm*** – форма указателя

*long* ***LinearPosition*** – положение разметки полосы прокрутки

*DOUBLE* ***CursorStep*** – шаг указателя

*VARIANT_BOOL* ***Enable*** – состояние включено/выключено (true(1) - включено, false(0) - выключено)

© ООО "ЭТМС" 1992-2024. Все права защищены

*long* ***Scale*** – масштаб  Параметр может принимать значения 0 или 1.

0 – линейный масштаб.

1 – логарифмический масштаб.

*VARIANT_BOOL* ***CurrentValueVisibility*** – отображение текущего значения

*VARIANT_BOOL* ***UnderlineVisibility*** – отображение подчеркивающей линии

*VARIANT_BOOL* ***LinearVisibility*** – отображение разметки полосы прокрутки

### Методы Horizontalslider.ocx

Методы можно использовать только во время выполнения программы.

*void* ***AboutBox****()* –содержит информацию о программе.

### События от компоненты Horizontalslider.ocx

*void* ***ModifyValue****(DOUBLE value)* – событие возникает при изменении значения.

## 10.4.Краткий перечень функций, используемые Horizontalslider.ocx

| **Свойства Horizontalslider.ocx** | |
|---|---|
| **Цвета** | |
| *long* ***GetTextColor****()* | опрашивается цвет текста |
| *void* ***SetTextColor****(unsigned long propVal)* | устанавливается цвет текста |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *long* ***GetBackgroundColor()*** | опрашивается цвет фона |
| *void* ***SetBackgroundColor*** *(unsigned long propVal)* | устанавливается цвет фона |
| *long* ***GetLineGradientSideColor*** *()* | опрашивается цвет края полосы прокрутки при заполнении ее градиентом |
| *void* ***SetLineGradientSideColor*** *(unsigned long propVal)* | устанавливается цвет края полосы прокрутки при заполнении ее градиентом |
| *long* ***GetLineGradientCenterColor()*** | цвет центра полосы прокрутки при заполнении ее градиентом |
| *void* ***SetLineGradientCenterColor****(unsigned long propVal)* | цвет центра полосы прокрутки при заполнении ее градиентом |
| *long* ***GetEndRectGradientSideColor()*** | опрашивается цвет края концов полосы прокрутки при заполнении их градиентом |
| *void* ***SetEndRectGradientSideColor*** *(unsigned long propVal)* | устанавливается цвет края концов полосы прокрутки при заполнении их градиентом |
| *long* ***GetEndRectGradientCenterColor()*** | опрашивается цвет центра концов полосы прокрутки при заполнении их градиентом |
| *void* ***SetEndRectGradientCenterColor*** *(unsigned long propVal)* | устанавливается цвет центра концов полосы прокрутки при заполнении их градиентом |
| *long* ***GetCursorGradientSideColor*** *()* | опрашивается цвет края курсора при заполнении его градиентом |
| *void* ***SetCursorGradientSideColor*** *(unsigned long propVal)* | устанавливается цвет края курсора при заполнении его градиентом |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *long* ***GetCursorGradientCenterColor()*** | опрашивается цвет центра курсора при заполнении его градиентом |
| *void* ***SetCursorGradientCenterColor*** *(unsigned long propVal)* | устанавливается цвет центра курсора при заполнении его градиентом |
| **Параметры шкалы** | |
| *double* ***GetStartValue()*** | опрашивается начальное значение шкалы |
| *void* ***SetStartValue****(double propVal)* | устанавливается начальное значение шкалы |
| *double* ***GetEndValue ()*** | опрашивается конечное значение шкалы |
| *void* ***SetEndValue*** *(double propVal)* | устанавливается команда конечное значение шкалы |
| *double* ***GetCurrentValue ()*** | опрашивается текущее значение |
| *void* ***SetCurrentValue****(double propVal)* | устанавливается текущее значение |
| *long* ***GetEndRectWidth()*** | опрашивается ширина концов полосы прокрутки |
| *void* ***SetEndRectWidth****(long propVal)* | устанавливается ширина концов полосы прокрутки |
| *long* ***GetCursorWidth()*** | опрашивается ширина указателя |
| *void* ***SetCursorWidth****(long propVal)* | устанавливается ширина указателя |
| *long* ***GetCURSORFORM()*** | опрашивается форма указателя |
| *void* ***SetCURSORFORM****(long propVal)* | устанавливается команда форма указателя |
| *long* ***GetLINEARPOSITION()*** | опрашивается положение разметки полосы прокрутки |
| *void* ***SetLINEARPOSITION****(long propVal)* | устанавливается команда положение разметки полосы прокрутки |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *double* ***GetCursorStep****()* | опрашивается шаг указателя |
| *void* ***SetCursorStep****(double propVal)* | устанавливается команда шаг указателя |
| *BOOL* ***GetEnable****()* | опрашивается состояние включено/выключено (true(1) - включено, false(0) - выключено) |
| *void* ***SetEnable****(BOOL propVal)* | устанавливается состояние включено/выключено (true(1) - включено, false(0) - выключено) |
| *long* ***GetScale****()* | опрашивается масштаб. (0 – линейный масштаб, 1 – логарифмический масштаб) |
| *void* ***SetScale****(long propVal)* | устанавливается команда опрашивается масштаб. (0 – линейный масштаб, 1 – логарифмический масштаб) |
| **Параметры отображении** | |
| *BOOL* ***GetCurrentValueVisibility****()* | опрашивается отображение текущего значения |
| *void* ***SetCurrentValueVisibility****(BOOL propVal)* | устанавливается отображение текущего значения |
| *BOOL* ***GetUnderlineVisibility*** *()* | опрашивается отображение подчеркивающей линии |
| *void* ***SetUnderlineVisibility****(BOOL propVal)* | устанавливается отображение подчеркивающей линии |
| *BOOL* ***GetLinearVisibility****()* | опрашивается отображение разметки полосы прокрутки |
| *void* ***SetLinearVisibility****(BOOL propVal)* | устанавливается отображение разметки полосы прокрутки |
| **События от компоненты Horizontalslider.ocx** | |
| *void* ***ModifyValue****(DOUBLE value)* | событие возникает при изменении значения |

© ООО "ЭТМС" 1992-2024. Все права защищены

© ООО "ЭТМС" 1992-2024. Все права защищены

# Описание программных модулей графиков

## Глава 1.Grid.ocx предназначена для графического отображения данных

Компонента **Grid.ocx** предназначена для графического отображения массивов данных в виде графиков зависимостей $y = y(x)$. Процедура рисования графиков написана таким образом, что отсутствует мерцание при перерисовке графиков

### 1.1.Установка компонента Grid.ocx

Для работы с модулем Grid его необходимо сначала установить в программу. При загрузке программы необходимо загрузить с помощью компонента необходимую программу, разместить окно программы на экране в удобном месте, установить необходимые параметры в программе и затем по событиям, происходящим от программы считывать данные из программы.

Для установки компонента Grid в проект *Microsoft Visual Studio 2010*, необходимо кликнуть правой кнопкой мыши на форме диалога и из появившегося меню выбрить пункт *Insert ActiveX Control*...(рисунок 1.1)

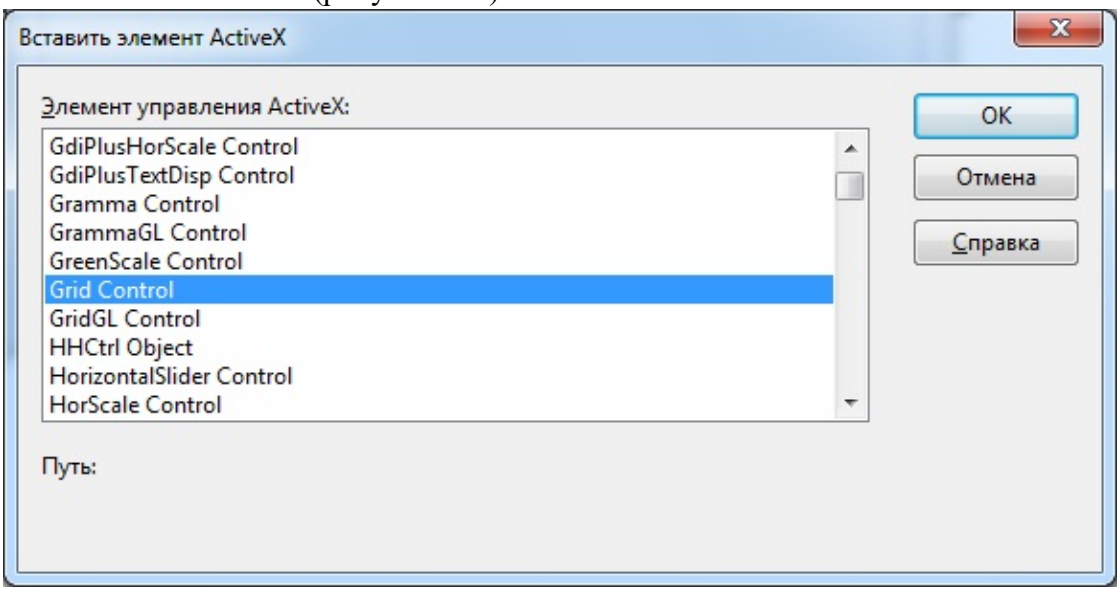



Рисунок 1.1

© ООО "ЭТМС" 1992-2024. Все права защищены

Из появившегося диалогового окна следует выбрать компонент *Grid Control* и нажать *OK* (рисунок 1.2). После этого компонент *Grid.ocx* появится на форме диалога (рисунок 1.2).

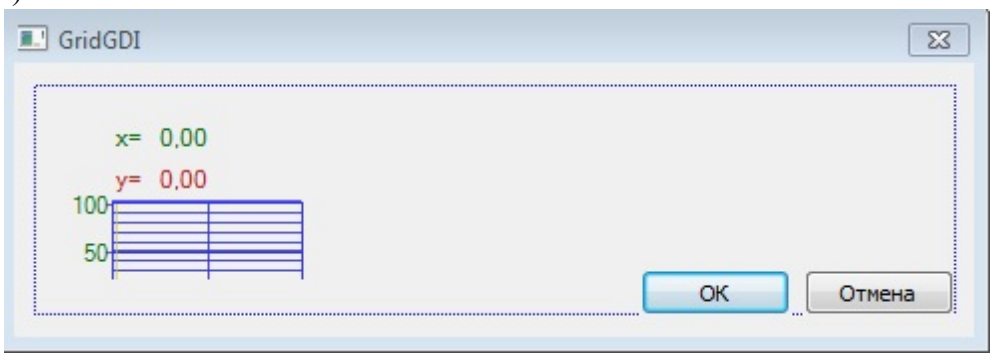



Рисунок 1.2

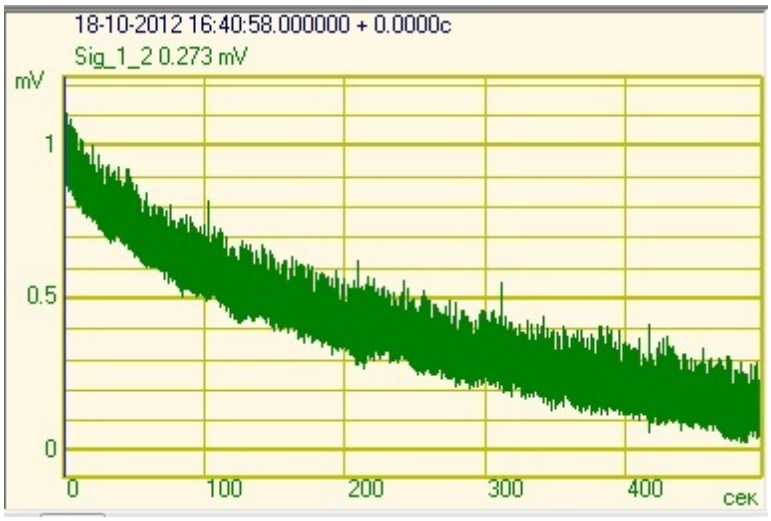
Одна установленная на форме компонента позволяет управлять одной программой. При необходимости управлять несколькими программами, необходимо установить на форму несколько компонент. Для того чтобы компонента не была видна на форме во время выполнения программы, необходимо установить свойство *Visible* равным *False*.

## 1.2.Назначение Grid.ocx

Компонента позволяет

- отображать одновременно несколько графиков в одних осях,
- масштабировать график по двум координатам,
- перемещать курсор при помощи кнопки мыши, ролика мыши и клавиатуры,
- показывать значения абсцисс и ординат на позиции курсора,
- отображать графики в виде сглаженных линий, ломаных линий, столбиков и т.д.,
- сохранять данные в Clipboard в графическом, текстовом и численном виде для создания отчетов в редакторах Word и Excel.

© ООО "ЭТМС" 1992-2024. Все права защищены




Компонент **Grid** используется в **осциллографе** и в программах, где необходимо представить результаты работы в графическом виде, будь то АЧХ и ФЧХ, спектры сигналов, огибающие и проходные, гистограммы, а также в программе **Просмотр и обработка результатов**.

## 1.3.Полное описание свойств, методов и событий Grid.ocx

Следует отметить, что компонент автомасштабируется при изменении его размеров, поэтому необходимо следить за тем, чтобы высота компонента в приложении была не менее 80 пикселей, иначе он может отображаться некорректно.

### Свойства Grid.ocx

Свойства можно задавать на этапе проектирования и во время выполнения программы. На этапе проектирования при выборе компонента и при нажатии клавиши <F4> появляется окно свойств. Эти свойства можно устанавливать на этапе проектирования программы.

Цвета задаются при помощи макроса *RGB(RED, GREEN, BLUE)*. Значения *RED*, *GREEN*, *BLUE* задают интенсивности цветов красного, зеленого и синего. Интенсивности изменяются в пределах от 0 до 255.

*OLE_COLOR* ***ClrCrs*** – цвет курсора.

*OLE_COLOR* ***ClrDig*** – цвет цифр на сетке.

*OLE_COLOR* ***ClrFon*** – цвет фона.

*OLE_COLOR* ***ClrGrd*** – цвет сетки.

© ООО "ЭТМС" 1992-2024. Все права защищены

*OLE_COLOR* ***ClrGrf*** – цвет графика. Для каждого графика цвет задается отдельно. Графики выбираются при помощи свойства Ind.

*OLE_COLOR* ***ClrLeg*** – цвет легенды

*OLE_COLOR* ***GetBackColor*** - устанавливается цвет подложки фона AciveX компонента (в настоящее время не используется)

Например, для изменении фона графика с желтого на зеленый, могут быть записаны числа:

*OLE_COLOR a = 0xFF00;*
*OLE_COLOR b = 0xFFFF;*

*long* ***Number*** – количество отображаемых графиков. Количество одновременно отображаемых графиков в одних осях не может превышать 20.

*long* ***Ind*** – индекс (номер) графика. Индекс может принимать значения от 0 до Number-1.

*long* ***Valid*** – Параметр для прорисовки графика. Для каждого графика параметр задается отдельно. Графики выбираются при помощи свойства *Ind*.

0 – не рисует выбранный график

1 – рисует выбранный график

*long* ***Size*** – количество точек по горизонтали. Параметр задается для всех графиков одновременно. Размер не может быть меньше или равен 0. Размер не может быть больше 500000.

*long* ***NumVisiblePoints*** – количество видимых точек при отображении графика. Может принимать значение от 1 до Size-1. По умолчанию равно -1. При значении -1 отображаются все данные. Задается для всех графиков одновременно.

*long* ***Px*** – позиция курсора по оси X. Этот параметр принимает значение от 0 до *Size*-1. Параметр можно читать и записывать. При перемещении курсора при помощи мыши или клавиатуры этот параметр отслеживает положение курсора.

*BYTE* ***luminescence*** – эффект, подобный послесвечению ЭЛТ осциллографа. Полупрозрачность предыдущего кадра. Значения от 0 до 255.

0 – непрозрачное,

255 – отсутствует.

*VARIANT_BOOL* ***markPlainData*** – при большом увеличении показывать маркеры не интерполированных данных.

0 – не показывать,

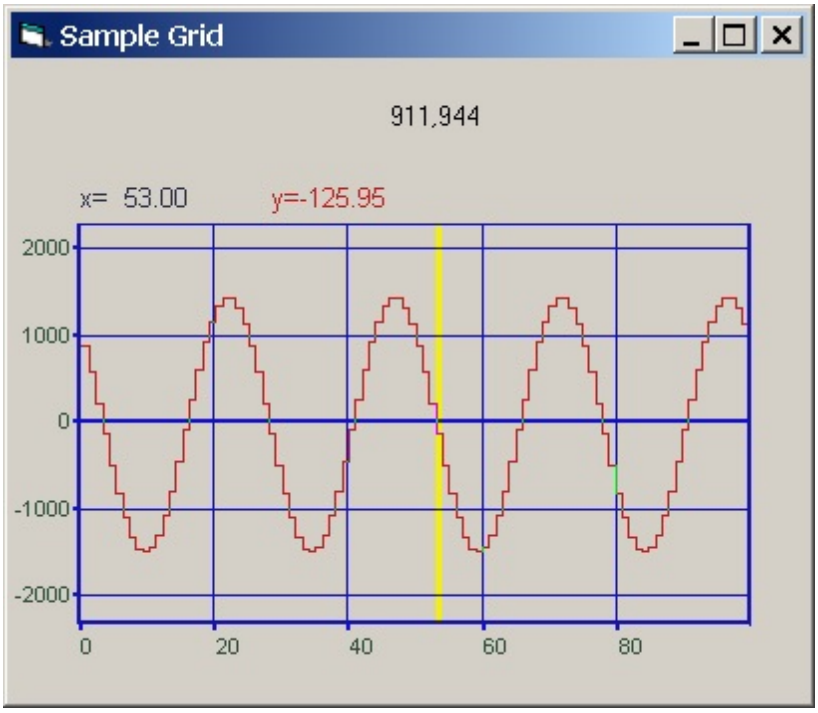
1 – показывать.

© ООО "ЭТМС" 1992-2024. Все права защищены

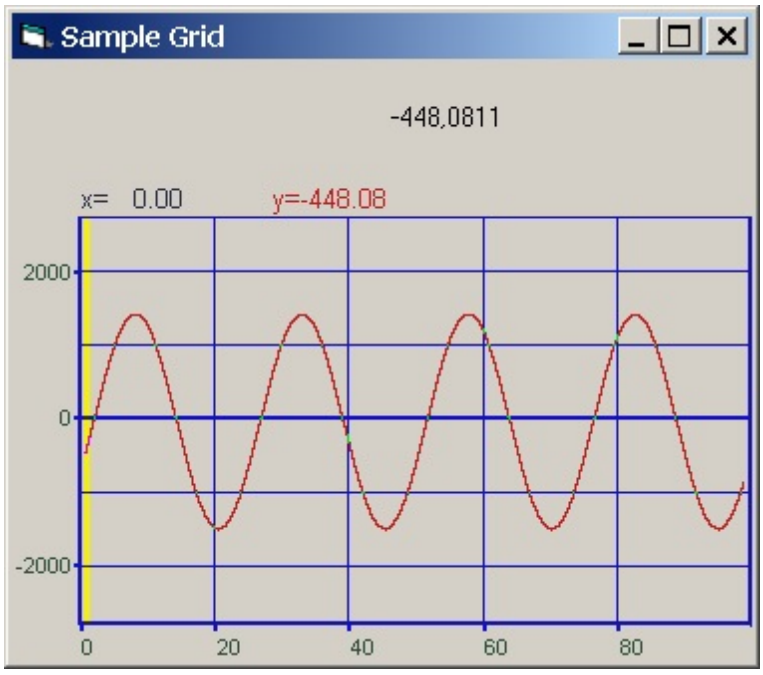
*long* ***TypeLine*** – тип линий графика. Для каждого графика параметр задается отдельно. Графики выбираются при помощи свойства *Ind*.

© ООО "ЭТМС" 1992-2024. Все права защищены

0 – горизонтальными линиями. График рисуется в виде ступенек от точки к точке.




1 – сплошными горизонтальными линиями. График рисуется в виде сглаженных линий.




2 – линиями. График рисуется в виде ломаной линии от точки к точке

© ООО "ЭТМС" 1992-2024. Все права защищены

3 – сплошными линиями. График рисуется в виде сглаженной линии от точки к точке

*long* ***TypeGrf*** – тип отображаемого графика. Для каждого графика параметр задается отдельно. Графики выбираются при помощи свойства *Ind*. Параметр может принимать значения:

0 – положительный

1 – знакопеременный.

*long* ***Log*** – логарифмическая шкала по Y. Параметр может принимать значения 0 или 1.

0 – линейный масштаб.

1 – логарифмический масштаб.

*double* ***UporDelta -*** минимальное разрешение по вертикали (при включенном ограничении).

*long* ***TypeAbs*** – тип осей координат.

0 – рисуется сетка и по X и по Y равномерная шкала по X

1 – логарифмическая сетка по X.

2 – не рисуются горизонтальные линии.

4 – рисуются вертикальные линии.

6 – не рисуются сетка.

*long* ***MakeUpor*** – включение ограничений по оси *Y*. При масштабировании графиков при помощи клавиатуры и мыши на стадии выполнения программы возможна ситуация, когда происходит переполнение данных. Для включения ограничений по масштабированию необходимо установить параметр *MakeUpor* в 1. Для снятия ограничений в 0.

*double* ***UporNis*** – минимальная величина по координате *Y*, которая может отобразиться на графике. Это свойство работает при включенных ограничениях по оси *Y*.

*double* ***UporVerh*** – максимальная величина по координате *Y*, которая может отображаться на графике. Это свойство работает при включенных ограничениях по оси *Y*.

© ООО "ЭТМС" 1992-2024. Все права защищены

**Пример**. Отображение временной реализации оцифрованного АЦП сигнала в виде осциллограммы. Входной диапазон работы АЦП от -10 В до + 10 В, вес младшего разряда 16-разрядного АЦП 0.0003 В. В этом случае можно задать следующие параметры.

MakeUpor = 1
*UporNis* = 0.0003
*UporVerh* = 10.0

*double* ***Xfirst*** – начальное значение по координате *X*.

*double* ***Xend*** – конечное значение по координате *X*.

Например, необходимо отобразить массив из 100 чисел в равномерной сетке по оси X. Числа представляют собой спектр мощности сигнала в диапазоне от 1000 Гц до 1100 Гц. Т.е. 0-ому элементу массива соответствует частота 1000 Гц, 99-му – частота 1100 Гц. В этом случае задаются следующие параметры:

*Xfirst* = 1000.
*Xend* = 1100.

BSTR TextXAxis - Подпись для оси X.

BSTR TextYAxis - Подпись для оси Y.

BSTR TextTAxis - Подпись для оси T.

BSTR TextZAxis - Подпись для оси Z.

*double* ***MathLX*** - текущее минимальное значение по X.. Левая граница отображения данных.

*double* ***MathDX***– текущий диапазон отображения по X. Длина отображения данных.

Например, необходимо отобразить часть данных из предыдущего примера в диапазоне от 1050 Гц до 1070 Гц. В этом случае задаются следующие параметры:

*MathLX* = 1050.
*MathDX* = 20.

При работе программы в режиме исполнения пользователь может менять область отображения графиков при помощи курсора мыши. Растяжение или сжатие графиков происходит при помощи указателя вида: ↔ , ⇹ – для горизонтальной оси. Сдвинуть графики вправо-влево можно при помощи указателя ← , → – для

© ООО "ЭТМС" 1992-2024. Все права защищены

горизонтальной оси. При этом меняются параметры *MathLX, MathDX.* В пользовательской программе в режиме исполнения можно прочитать текущие параметры области отображения.

*double* ***MathLY*** – текущее минимальное значение по Y. Нижняя граница отображения данных.

*double* ***MathDY*** – текущий диапазон отображения по Y. Высота отображения данных.

Растяжение или сжатие графиков происходит при помощи указателя вида:  ,  – для вертикальной оси. Сдвинуть графики вверх или вниз можно при помощи указателя  ,  – для вертикальной оси. При этом меняются параметры *MathLY, MathDY.* В пользовательской программе в режиме исполнения можно прочитать текущие параметры области отображения. При включенных ограничениях отображения графиков *MakeUpor* = 1 , параметр *MathDY* не может быть меньше *UporNis* , сумма параметров *MathLY* + *MathDY* не может быть больше *UporVerh*.

## Методы Grid.ocx

Методы можно использовать только во время выполнения программы.

*void* ***AboutBox()*** –содержит информацию о программе.

*void* ***autoScaleY*** *(void)* – однократно применяет автомасштаб по оси Y.

*long* ***Paint*** *(float* Buffer)* – метод, позволяет передать данные для отображения графика и перерисовать график. При построении нескольких графиков в одних осях необходимо при помощи свойства *Ind* выбирать, какой график требует перерисовки.

Параметры:

*float* Buffer* – адрес массива данных в программе пользователя. В компоненту копируется количество данных определенное свойством *Size*.

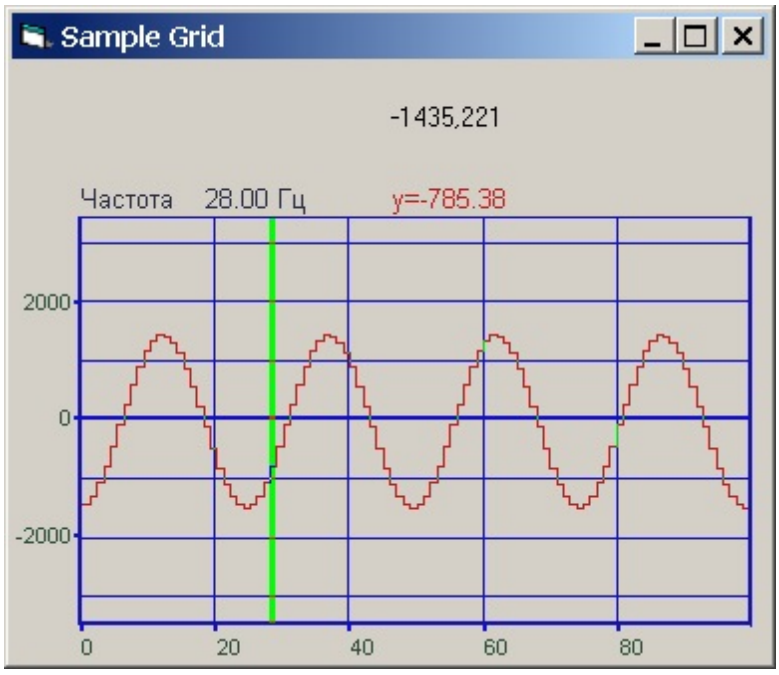
При создании программ обработки сигналов в реальном времени с многократной перерисовкой графиков, типа осциллографов или анализаторов спектра, необходимо один раз задать параметры компонента, и затем последовательно обращаться только к методу *Paint(...)* для создания эффекта движения.

*long* ***FormatX*** *(BSTR fmt)* – метод передает в компонент строку для форматного вывода информации по координате *X* в легенде графика. Форматный ввод-вывод позволяет передавать программе и выдавать из программы на внешнее устройство символьные, строковые и числовые значения, обеспечивая преобразование данных в процессе ввода-вывода.

Например, обращение вида:

*FormatX("Частота %7.2f Гц")*

© ООО "ЭТМС" 1992-2024. Все права защищены




позволяет отображать координаты по оси X в значениях по частоте.

Формат представления чисел.

*%f* – общий формат представления числа в плавающей запятой в виде числа с точкой

*%e* – общий формат представления числа в плавающей запятой в виде числа с экспонентой (*1000 = 1e3, 0.001 = 1e-3*)

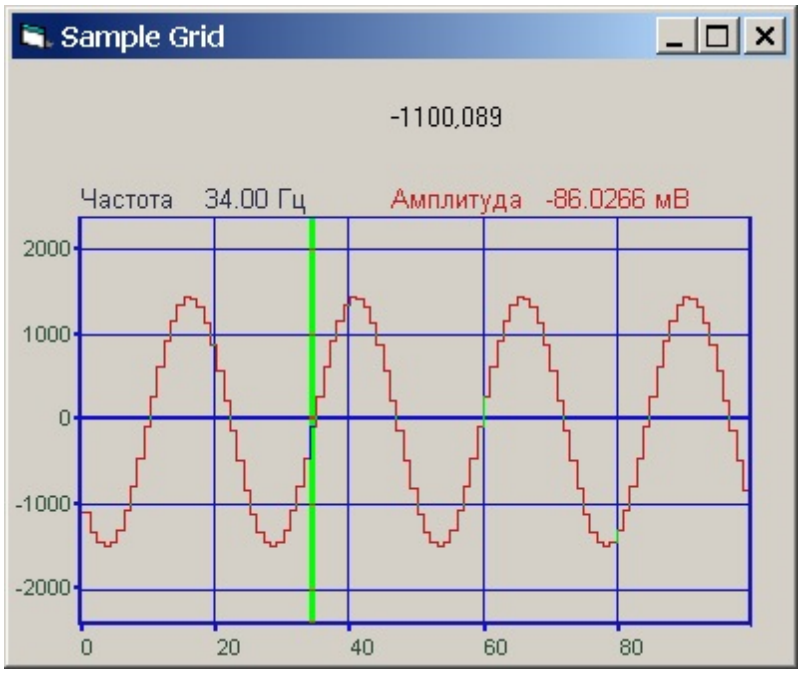
*%7.2f* – представление числа в фиксированном формате. В данном примере *7* – количество знакомест для представления числа, *2* – количество цифр после запятой в представлении числа.

*long* ***FormatY*** *(BSTR fmt)* – метод передает в компонент строку для форматного вывода информации по координате *Y* в легенде графика. Форматный ввод-вывод позволяет передавать программе и выдавать из программы на внешнее устройство символьные, строковые и числовые значения, обеспечивая преобразование данных в процессе ввода-вывода. Метод должен выполняться для каждого графика. Графики выбираются при помощи свойства *Ind*.

Например, обращение вида:

*FormatY ("Амплитуда %9.4f мВ")*

© ООО "ЭТМС" 1992-2024. Все права защищены




позволяет отображать координаты по оси *Y* в значениях по амплитуде.

*void* ***Formatxy*** *(BSTR fmt)* – метод содержит надпись в графике. Форматный ввод-вывод позволяет передавать программе и выдавать из программы на внешнее устройство символьные, строковые и числовые значения, обеспечивая преобразование данных в процессе ввода-вывода. Это может быть произвольная пользовательская информация о данном графике.

*void* ***AltC*** *(float *buffer)* – метод, загружает массив разметки данных по X. Этот метод работает при установленном свойстве типа сетки *TypeAbs* = 2. В пользовательском буфере *buffer* размера *Size* необходимо записать значения координаты *X*. Например, при логарифмической развертке по оси *X*, в буфере могут быть записаны числа:

*buffer[0] = 1.*
*buffer[1] = 1.25*
*buffer[2] = 1.6*
*buffer[3] = 2.*
*buffer[4] = 2.5*
……

*void* ***PushDataToClipBoard*** *(void)* - позволяет сохранить изображение графика в Clipboard в графическом (bmp) формате.

*void* ***savePng*** *(BSTR FileName)* – сохраняет изображение графика в PNG-файл, параметр – имя файла.

© ООО "ЭТМС" 1992-2024. Все права защищены

*short* ***Display****()* - перерисовывает графики

*long* ***PaintNet****(long pBuffer)* - метод, позволяет передать данные для отображения графика и перерисовать график. При построении нескольких графиков в одних осях необходимо при помощи свойства *Ind* выбирать, какой график требует перерисовки.

Параметры:

*float* Buffer* – адрес массива данных в программе пользователя. В компоненту копируется количество данных определенное свойством *Size*.

При создании программ обработки сигналов в реальном времени с многократной перерисовкой графиков, типа осциллографов или анализаторов спектра, необходимо один раз задать параметры компонента, и затем последовательно обращаться только к методу *Paint(...)* для создания эффекта движения.

*void* ***AltCNet****(long pBuffer)* - метод, загружает массив разметки данных по X. Этот метод работает при установленном свойстве типа сетки *TypeAbs* = 2. В пользовательском буфере *buffer* размера *Size* необходимо записать значения координаты *X*.

Например, при логарифмической развертке по оси *X*, в буфере могут быть записаны числа:

*buffer[0] = 1.*
*buffer[1] = 1.25*
*buffer[2] = 1.6*
*buffer[3] = 2.*
*buffer[4] = 2.5*
……

*void* ***autoScaleY****(void)* - позволяет однократно применить автомасштаб по Y.

*void* ***savePng****(BSTR FileName)* - позволяет Сохранить изображение в файле.

**Примечание:** Следует отметить, то при работе в .net языках функции, в которых в качестве параметров выступают указатели на массив, например GetData в .net языках работают некорректно, поэтому следует использовать обновленные функции, имеющие такое же название + приписка Net, например, GetDataNet.

## События от компоненты Grid.ocx

*void* ***DblClick*** *(void)* – событие возникает при двойном нажатии на кнопку манипулятора «мышь», при нахождении мыши над графиком.

© ООО "ЭТМС" 1992-2024. Все права защищены

*void* ***KeyDown*** *(SHORT* KeyCode, SHORT Shift)* – событие возникает при нажатии на клавишу клавиатуры, когда фокус находится на компоненте.

Параметры:
*KeyCode* – код клавиши
*Shift* – код нажатия служебных клавиш *<Ctrl>,<Shift>,<Alt>*.

<T> - *Копирование сопроводительной информации.* Для копирования сопроводительной информации нажать левой кнопкой «мыши» на поле графика программы и
нажать на кнопку клавиатуры <T>(в латинской раскладке клавиатуры). Сопроводительная информация скопируется в буфер обмена(Clipboard).
<C> - *Копирование графика.* Для копирования графика нажать левой кнопкой «мыши» на поле графика и нажать комбинацию «горячих клавиш» клавиатуры <Ctrl> +<C>.
График запишется в буфер обмена(Clipboard).
<N> *Копирование численных значений.* Для копирования численных значений нажать левой кнопкой «мыши» на поле графика программы и
нажать на кнопку клавиатуры <N>(в латинской раскладке клавиатуры). Численные значения скопируются в буфер обмена(Clipboard).
<A> *Управление движением курсора влево.* Нужно нажать левой кнопкой «мыши» на поле графика программы и нажать на кнопку клавиатуры <A>(в латинской раскладке клавиатуры)
или движением колесика мыши вниз.
<D> *Управление движением курсора вправо.* Нужно нажать левой кнопкой «мыши» на поле графика программы и нажать на кнопку клавиатуры <D>(в латинской раскладке клавиатуры).
или движением колесика мыши вверх.
<F> Сохранения гриды в виде изображения по нажатию на кнопку "F".

*void* ***MouseDown*** *(short Button, short Shift, OLE_XPOS_PIXELS x, OLE_YPOS_PIXELS y)* – событие возникает при нажатии на кнопку манипулятора типа «мышь», при нахождении мыши над графиком.

Параметры:
*Button* – код кнопки мыши
*Shift* – код нажатия служебных клавиш *<Ctrl>,<Shift>,<Alt>*.
*x* – координата по оси *X* в пикселах (разрешении экрана)
*y* – координата по оси *Y* в пикселах (разрешении экрана)

*void* ***MouseUp*** *(short Button, short Shift, OLE_XPOS_PIXELS x, OLE_YPOS_PIXELS y)* – событие возникает при отжатии кнопки манипулятора типа «мышь», при нахождении мыши над графиком.

Параметры:
*Button* – код кнопки мыши
*Shift* – код нажатия служебных клавиш *<Ctrl>,<Shift>,<Alt>*.
*x* – координата по оси *X* в пикселях (разрешении экрана)
*y* – координата по оси *Y* в пикселях (разрешении экрана)

v*oid* ***MouseWheel*** *(void)* – событие возникает при вращении ролика манипулятора типа «мышь».

© ООО "ЭТМС" 1992-2024. Все права защищены

## 1.4.Краткий перечень функций, используемые Grid.ocx

| Свойства Grid.ocx | |
|---|---|
| **Цвета** | |
| *long* ***GetClrCrs****()* | опрашивается цвет курсора |
| *void* ***SetClrCrs****(unsigned long propVal)* | устанавливается цвет курсора |
| *long* ***GetClrDig****()* | опрашивается цвет цифр на сетке |
| *void* ***SetClrDig****(unsigned long propVal)* | устанавливается цвета цифр на сетке |
| *long* ***GetClrFon*** *()* | опрашивается цвет фона |
| *void* ***SetClrFon*** *(unsigned long propVal)* | устанавливается цвет фона |
| *long* ***GetClrGrd*** *()* | опрашивается цвет сетки |
| *void* ***SetClrGrd*** *(unsigned long propVal)* | устанавливается цвет сетки |
| *long* ***GetClrGrf*** *()* | опрашивается цвет графика |
| *void* ***SetClrGrf*** *(unsigned long propVal)* | устанавливается цвет графика. Для каждого графика цвет задается отдельно. Графики выбираются при помощи свойства ***Ind*** |
| *long* ***GetClrLeg*** *()* | опрашивается цвет легенды графика |
| *void* ***SetClrLeg*** *(unsigned long propVal)* | устанавливается цвет легенды графика |
| *long* ***GetBackColor*** *()* | опрашивается цвет подложки фона AciveX компонента (в настоящее время не используется) |
| *void* ***SetBackColor*** *(unsigned long propVal)* | устанавливается цвет подложки фона AciveX компонента (в настоящее время не используется) |
| **Параметры графиков** | |
| *long* ***GetNumber****()* | опрашивается количество отображаемых графиков. Количество одновременно отображаемых графиков в одних осях не может превышать 20 |
| *void* ***SetNumber****(long propVal)* | устанавливается количество отображаемых графиков. Количество одновременно отображаемых графиков в одних осях не может превышать 20 |
| *long* ***GetInd****()* | опрашивается индекс (номер) графика. Индекс может принимать значения от 0 до Number-1 |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *void* ***SetInd*** *(long propVal)* | устанавливается индекс (номер) графика. Индекс может принимать значения от 0 до Number-1 |
| *long* ***GetValid****()* | опрашивается прорисовки графика (0 – не рисует выбранный график,1 – рисует выбранный график) |
| *void* ***SetValid****(long propVal)* | устанавливается прорисовки графика (0 – не рисует выбранный график,1 – рисует выбранный график) |
| *long* ***GetSize*** *()* | опрашивается количество точек по горизонтали. Размер не может быть больше 500000 |
| *void* ***SetSize*** *(long propVal)* | устанавливается количество точек по горизонтали. Размер не может быть больше 500000 |
| *long* ***GetNumVisiblePoints*** *()* | опрашивается количество видимых точек в графике |
| *void* ***SetNumVisiblePoints*** *(long propVal)* | устанавливается количество видимых точек в графике |
| *long* ***GetPx*** *()* | опрашивается позиция курсора по оси X |
| *void* ***SetPx*** *(long propVal)* | устанавливается позиция курсора по оси X |
| *char* ***Getluminescence****()* | опрашивается эффект, подобный послесвечению ЭЛТ осциллографа |
| *void* ***Setluminescence****(unsigned char propVal)* | устанавливается эффект, подобный послесвечению ЭЛТ осциллографа |
| *BOOL* ***GetmarkPlainData****()* | опрашивается показывать маркеры не интерполированных данных (0 – не показывать, 1 – показывать) при большом увеличении |
| *void* ***SetmarkPlainData****(BOOL propVal)* | Устанавливается показывать маркеры не интерполированных данных (0 – не показывать, 1 – показывать) при большом увеличении |
| **Представление графиков** | |
| *long* ***GetTypeLine****()* | опрашивается тип линий графика (0-горизонтальными линиями, 1 – сплошными горизонтальными линиями , 2 – ломаными линиями, 3 – сплошными ломанными) |
| *void* ***SetTypeLine****(long propVal)* | устанавливается тип линий графика(0-горизонтальными линиями, 1 – сплошными горизонтальными линиями , 2 – ломаными линиями, 3 – сплошными ломанными) |
| *long* ***GetTypeGrf****()* | опрашивается тип отображаемого графика ( 0 – положительный,1 – знакопеременный) |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *void* ***SetTypeGrf****(long propVal)* | устанавливается тип отображаемого графика ( 0 – положительный,1 – знакопеременный) |
| *long* ***GetMakeUpor****()* | опрашивается включение ограничений по оси Y. Для включения ограничений по масштабированию необходимо установить параметр MakeUpor в 1. Для снятия ограничений в 0 |
| *void* ***SetMakeUpor****(long propVal)* | устанавливается включение ограничений по оси Y. Для включения ограничений по масштабированию необходимо установить параметр MakeUpor в 1. Для снятия ограничений в 0 |
| **Область отображении шкалы** | |
| *long* ***Getlog*** *()* | опрашивается тип шкалы Y (0- линейная или 1 - логарифмическая) |
| *void* ***Getlog*** *(long propVal)* | устанавливается тип шкалы Y (0- линейная или 1 - логарифмическая) |
| *char* ***GetUporDelta*** *()* | опрашивается минимальное разрешение по вертикали (при включенном ограничении) |
| *void* ***SetUporDelta****(double propVal)* | устанавливается минимальное разрешение по вертикали (при включенном ограничении) |
| *long* ***GetTypeAbs****()* | опрашивается тип осей координат (0 – рисуется сетка и по X и по Y (равномерная шкала по X), 1 – логарифмическая сетка по X, 2 – не рисуются горизонтальные линии, 4 – рисуются вертикальные линии, 6 – не рисуется сетка) |
| *void* ***SetTypeAbs****(long propVal)* | устанавливается тип осей координат (0 – рисуется сетка и по X и по Y (равномерная шкала по X), 1 – логарифмическая сетка по X, 2 – не рисуются горизонтальные линии, 4 – рисуются вертикальные линии, 6 – не рисуется сетка) |
| *double* ***GetUporNis****()* | опрашивается минимальная величина по координате Y, которая может отобразиться на графике. Это свойство работает при включенных ограничениях по оси Y |
| *void* ***SetUporNis****(double propVal)* | устанавливается минимальная величина по координате Y, которая может отобразиться на графике. Это свойство работает при включенных ограничениях по оси Y |
| *double* ***GetUporVerh****()* | опрашивается максимальная величина по координате Y, которая может отобразиться на |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| | графике. Это свойство работает при включенных ограничениях по оси Y |
| *void* ***SetUporVerh****(double propVal)* | устанавливается максимальная величина по координате Y, которая может отобразиться на графике. Это свойство работает при включенных ограничениях по оси Y |
| *double* ***GetXfirst****()* | опрашивается начальное значение по координате *X* |
| *void* ***SetXfirst****(double propVal)* | устанавливается начальное значение по координате *X* |
| *double* ***GetXend****()* | опрашивается конечное значение по координате *X* |
| *void* ***SetXend****(double propVal)* | устанавливается конечное значение по координате *X* |
| **Область отображении графиков** | |
| *double* ***GetMathlx****()* | опрашивается текущее минимальное значение по X. Левая граница отображения данных |
| *void* ***SetMathlx****(double propVal)* | устанавливается текущее минимальное значение по X. Левая граница отображения данных |
| *double* ***GetMathdx****()* | опрашивается текущий диапазон отображения по X. Длина отображения данных |
| *void* ***SetMathdx****(double propVal)* | устанавливается текущий диапазон отображения по X. Длина отображения данных |
| *double* ***GetMathly****()* | опрашивается текущее минимальное значение по Y. Нижняя граница отображения данных |
| *void* ***SetMathly****(double propVal)* | Устанавливается текущее минимальное значение по Y. Нижняя граница отображения данных |
| *double* ***GetMathdy****()* | опрашивается текущий диапазон отображения по Y. Высота отображения данных |
| *void* ***SetMathdy****(double propVal)* | устанавливается текущий диапазон отображения по Y. Высота отображения данных |
| **Подписи для осей** | |
| CString TextXAxis() | опрашивается подпись для оси X. |
| void SetTextXAxis(CString propVal) | устанавливается подпись для оси X. |
| CString TextYAxis() | опрашивается подпись для оси Y. |
| void SetTextYAxis(CString propVal) | устанавливается подпись для оси Y. |
| CString TextTAxis() | опрашивается подпись для оси T. |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| void SetTextTAxis(CString propVal) | устанавливается подпись для оси T. |
| CString TextZAxis() | опрашивается подпись для оси Z. |
| void SetTextZAxis(CString propVal) | устанавливается подпись для оси Z. |
| **Методы Grid.ocx** | |
| *void **AboutBox**()* | содержит информацию о программе |
| *void **autoScaleY**()* | однократно применяет автомасштаб по оси Y |
| *long **Paint**(float * buffer)* | позволяет передать данные для отображения графика и перерисовать график |
| | |
| *long **FormatX**(LPCTSTR fmt)* | передает в компонент строку для форматного вывода информации по координате X в легенде графика |
| *long **FormatY**(LPCTSTR fmt)* | передает в компонент строку для форматного вывода информации по координате Y в легенде графика |
| *void **Formatxy**(LPCTSTR fmt)* | содержит надпись в графике |
| *void **AltC**(float * buffer)* | позволяет загружает массив разметки данных по X. Этот метод работает при установленном свойстве типа сетки TypeAbs = 2 |
| *void **PushToClipBoard**()* | позволяет сохранить изображение графика в Clipboard в графическом (bmp) формате |
| *void **savePng**(LPCTSTR FileName)* | сохраняет изображение графика в PNG-файл, параметр – имя файла |
| *short **Display**()* | перерисовывает графики |
| *long **PaintNet**(long pBuffer)* | позволяет передать данные для отображения графика и перерисовать график, используется при работе в .net языках |
| *void **AltCNet**(long pBuffer)* | позволяет задать значения по оси X. Этот метод работает при установленном свойстве типа сетки TypeAbs = 2, используется при работе в .net языках |
| *void **autoScaleY**(void)* | позволяет однократно применить автомасштаб по Y |
| *void **savePng**(BSTR FileName)* | позволяет Сохранить изображение в файле |
| **События от компоненты Grid.ocx** | |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *void* ***DblClick*** *()* | событие возникает при двойном нажатии на кнопку манипулятора «мышь», при нахождении мыши над графиком |
| *void* ***KeyDown****(SHORT* KeyCode, SHORT Shift)* | событие возникает при нажатии на клавишу клавиатуры, когда фокус находится на компоненте |
| *void* ***MouseDown****(short Button, short Shift, OLE_XPOS_PIXELS x, OLE_YPOS_PIXELS y)* | событие возникает при нажатии на кнопку манипулятора типа «мышь», при нахождении мыши над графиком |
| *void* ***MouseUp****(short Button, short Shift, OLE_XPOS_PIXELS x, OLE_YPOS_PIXELS y)* | событие возникает при отжатии кнопки манипулятора типа «мышь», при нахождении мыши над графиком |
| *void* ***MouseWheel****(void)* | событие возникает при вращении ролика манипулятора типа «мышь» |

# Глава 2.GridGL.ocx предназначена для графического отображения данных (улучшенная)

Компонент **GridGL.ocx** является расширенной версией компонента **Grid.ocx**. Компонент **GridGL.ocx** поддерживает многие свойства, методы и события компоненты **Grid** и имеет дополнительные свойства. Процедура рисования графиков написана таким образом, что отсутствует мерцание при перерисовке графиков.

## 2.1.Установка компонента GidGL.ocx

Для работы с модулем GridGL его необходимо сначала установить в программу. При загрузке программы необходимо загрузить с помощью компонента необходимую программу, разместить окно программы на экране в удобном месте, установить необходимые параметры в программе и затем по событиям, происходящим от программы считывать данные из программы.

Для установки компонента GridGL в проект *Microsoft Visual Studio 2010*, необходимо кликнуть правой кнопкой мыши на форме диалога и из появившегося меню выбрить пункт *Insert ActiveX Control*...(рисунок 1.1)

© ООО "ЭТМС" 1992-2024. Все права защищены

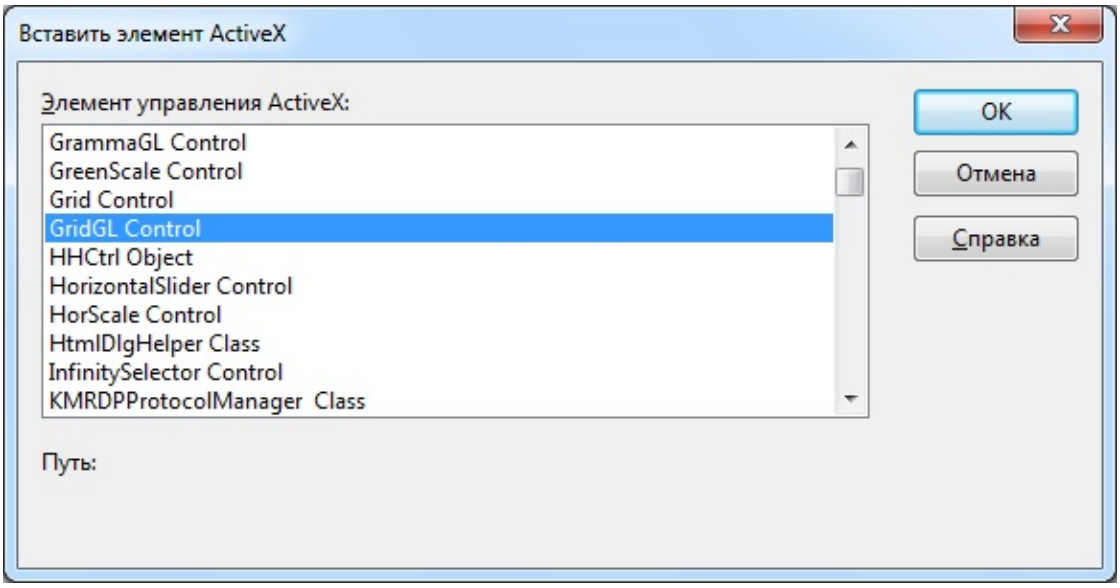



Рисунок 1.1

Из появившегося диалогового окна следует выбрать компонент *GridGL Control* и нажать *OK* (рисунок 1.2). После этого компонент GridGL.*ocx* появится на форме диалога (рисунок 1.2).

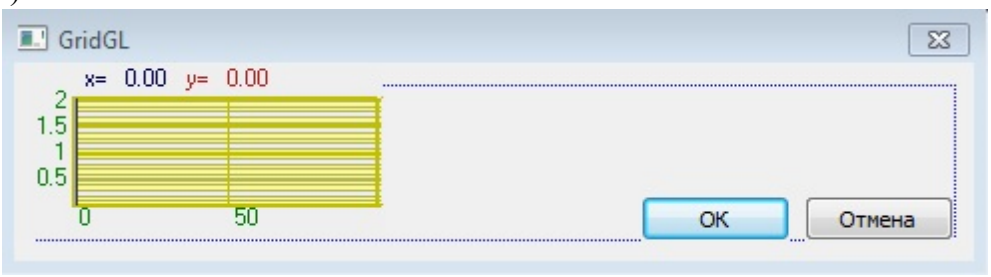



Рисунок 1.2

Одна установленная на форме компонента позволяет управлять одной программой. При необходимости управлять несколькими программами, необходимо установить на форму несколько компонент. Для того чтобы компонента не была видна на форме во время выполнения программы, необходимо установить свойство *Visible* равным *False*.

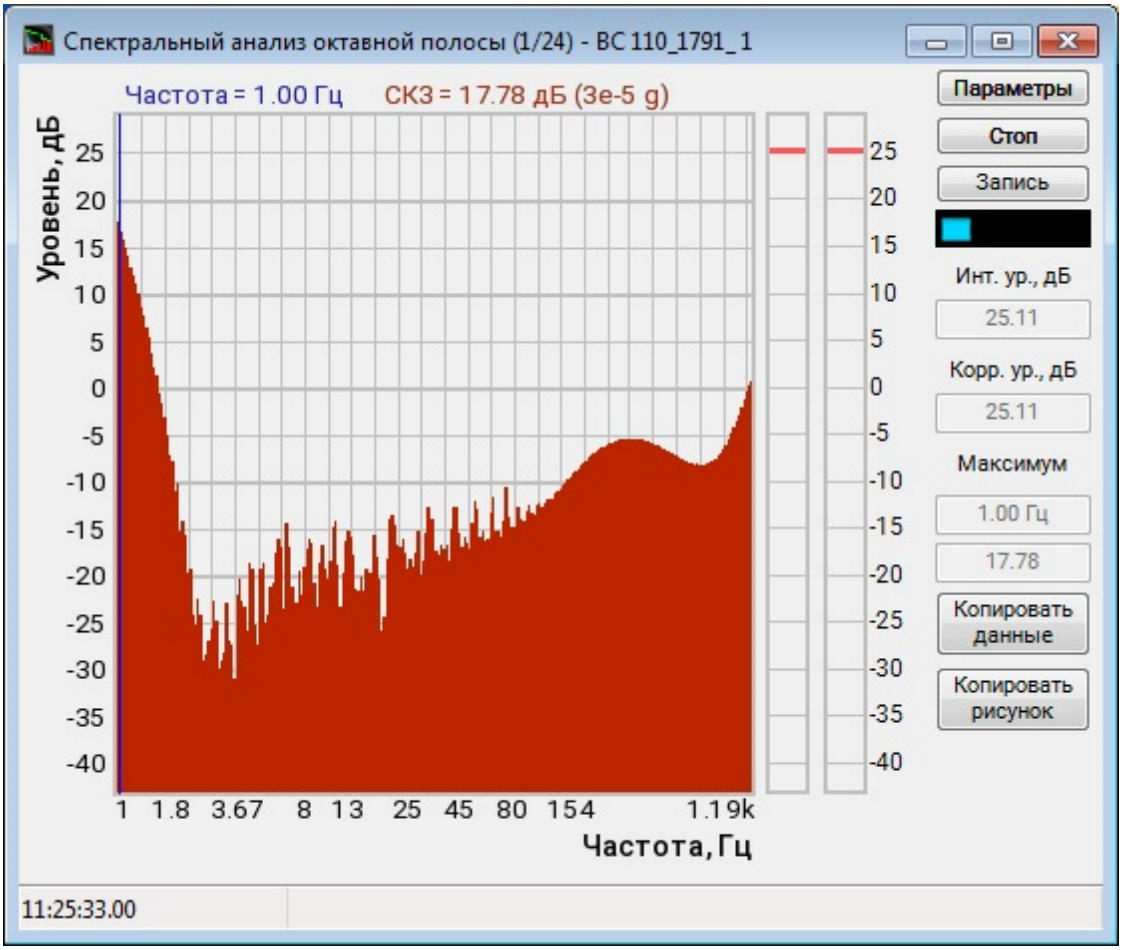
## 2.2.Назначение GridGL.ocx

Компонента позволяет:

- отображать одновременно несколько графиков в одних осях,
- масштабировать график по двум координатам,
- перемещать курсор при помощи кнопки мыши, ролика мыши и клавиатуры,

© ООО "ЭТМС" 1992-2024. Все права защищены

- показывать значения абсцисс и ординат на позиции курсора,
- отображать графики в виде сглаженных линий, ломаных линий, столбиков и т.д.,
- сохранять данные в Clipboard в графическом, текстовом и численном виде для создания отчетов в редакторах Word и Excel.




## 2.3.Полное описание свойств, методов и событий GridGL.ocx

### Свойства GridGL.ocx

Свойства можно задавать на этапе проектирования и во время выполнения программы. На этапе проектирования при выборе компонента и при нажатии клавиши <F4> появляется окно свойств. Эти свойства можно устанавливать на этапе проектирования программы.

© ООО "ЭТМС" 1992-2024. Все права защищены

Цвета задаются при помощи макроса *RGB(RED, GREEN, BLUE)*. Значения *RED*, *GREEN*, *BLUE* задают интенсивности цветов красного, зеленого и синего. Интенсивности изменяются в пределах от 0 до 255.

*OLE_COLOR* ***ClrCrs*** – цвет курсора.

*OLE_COLOR* ***ClrDig*** – цвет цифр на сетке.

*OLE_COLOR* ***ClrFon*** – цвет фона.

*OLE_COLOR* ***ClrGrd*** – цвет сетки.

*OLE_COLOR* ***ClrGrf*** – цвет графика. Для каждого графика цвет задается отдельно. Графики выбираются при помощи свойства *Ind*.

*OLE_COLOR* ***ClrLeg*** – цвет легенды.

Например, для изменении фона графика с желтого на зеленый, могут быть записаны числа:

*OLE_COLOR a = 0xFF00;*
*OLE_COLOR b = 0xFFFF;*

*long* ***Number*** – количество отображаемых графиков. Количество одновременно отображаемых графиков в одних осях не может превышать 60.

*long* ***Ind*** – индекс (номер) графика. Индекс может принимать значения от 0 до Number-1.

*long* ***Valid*** – Параметр для прорисовки графика. Для каждого графика параметр задается отдельно. Графики выбираются при помощи свойства *Ind*.

0 – не рисует выбранный график

1 – рисует выбранный график

*long* ***Size*** – количество точек по горизонтали. Параметр задается для всех графиков одновременно. Размер не может быть меньше или равен 0. Размер не может быть больше 500000.

*long* ***NumVisiblePoints*** – количество видимых точек при отображении графика. Может принимать значение от 1 до Size-1. По умолчанию равно -1. При значении -1 отображаются все данные. Задается для всех графиков одновременно.

*DOUBLE* ***Reference*** – Опорное значение для расчета уровня в дБ.
Уровень в дБ равен:

$$L_{dB} = 20\log\frac{U}{R}$$

R - значение Reference, U – значение сигнала в физических единицах измерения, например в Вольтах. $L_{dB}$ - уровень сигнала в дБ.

© ООО "ЭТМС" 1992-2024. Все права защищены

*long* ***TypeFill*** –тип заполнения графика.

0 – с заполнением

1 – без заполнения

*long* ***LineWidth*** - толщина линии графика.

*long* ***Px*** – позиция курсора по оси X. Этот параметр принимает значение от 0 до *Size*-1. Параметр можно читать и записывать. При перемещении курсора при помощи мыши или клавиатуры этот параметр отслеживает положение курсора.

*VARIANT_BOOL* ***HideZeros -*** Скрывать нулевые значения.

*void* ***AutoScaleX() -*** устанавливается однократно применить автомасштаб по X.

*long* ***LeftBorderGrid -*** устанавливается левая граница GridGL.

*long* ***ShowCursor -*** устанавливается отображение основного курсора.

*long* ***Log*** – логарифмическая шкала по Y. Параметр может принимать значения 0 или 1.

0 – линейный масштаб.

1 – логарифмический масштаб.

*double* ***UporDelta -*** минимальное разрешение по вертикали (при включенном ограничении).

*long* ***TypeXAxis*** –тип шкалы по оси X.

0 - линейная

1- логарифмическое

2 - долеоктавная

*long* ***TypeYAxis*** – тип шкалы по оси Y.

0 - линейная

1- логарифмическое

*long* ***ShowScalePropPage*** – показывать страницу свойств шкалы.

*VARIANT_BOOL* ***IsShowPropSheet*** – состояние выводить/не выводить на экран страницу свойств, где пользователь может установить режим работы, вид шкалы, цвет фона, тип шрифта, максимальное и минимальное значения.

© ООО "ЭТМС" 1992-2024. Все права защищены

*long* ***TypeYData*** - тип данных по Y:

0- линейная;
1- логарифмическая;

*BSTR* ***TextXAxis*** - единица измерения по оси X.

*BSTR* ***TextYAxis*** - единица измерения по оси Y.

*long* ***AltCoords*** – данные по равномерным и неравномерным координатам по X.

0 – равномерные.
Данные для отображения по координатам X и Y представляются в виде:
*X={Xfirst…Xend};*
*Y={buffer[0]…buffer[Size-1]};*
Массив данных buffer для отображения по Y передается методом Paint(float* buffer).
При отображении координаты по оси считаются равномерными

$$x_i = (Xend - Xfirst) / Size \times i$$
$$y_i = buffer[i]$$
$$i = 0...Size - 1$$

1 – произвольные координаты, координаты X задаются в методе AltC(float* cord) в массиве cord. Шкала становится 1/n октавной при любом TypeXAxis.
Данные для отображения по координатам X и Y представляются в виде:
*X={cord[0]…cord[Size-1]};*
*Y={buffer[0]…buffer[Size-1]};*
Массив данных buffer для отображения по Y передается методом Paint(float* buffer).
При отображении координаты по оси X считаются произвольными

$$x_i = cord[i]$$
$$y_i = buffer[i]$$
$$i = 0...Size - 1$$

Для каждого графика толщина линии задается индивидуально. Номер графика выбирается свойством Ind.

VARIANT_BOOL ***IsShowPropSheet*** - Показывать или не показывать страницу настроек

VARIANT_BOOL ***PowerDecibell*** **-** При отображении децибел использовать формулу мощности.

© ООО "ЭТМС" 1992-2024. Все права защищены

*long* ***SetAxisSetting -*** Установка параметров для шкалы, \Index - индекс настраиваемой оси (от 0 до 4) привязку осей к индексу можно посмотреть в GridGL::AxisSide, \
UseAxis - 1 использовать шкалу / 0 не использовать,\
AxisText - Текст на шкале, AxisColor - цвет шкалы и надписи, \
CoeffMultiplicative и CoeffAdditive - коэффициенты на которые отличаются значения шкалы

*long* ***SetAdditionalWindowSettings -*** Установка параметров для дополнительного окна, \
Index - индекс настраиваемого окна (от 0 до 99), \
UseWindow - 1 использовать окно / 0 не использовать, \
GraphNumber - номер графика, к которому привязано окно, \
IndexPoint - индекс точки на графике к которой привязано окно, \
TextColor и BackgroundColor - цвета использующиеся при отрисовке окна, \
WindowText - текст в окне

*void* ***DrawRiftY****(DOUBLE riftMathStartY, DOUBLE riftMathStopY, LONG riftMinWidthPy)* - Установка параметров для нарисовать просвет.

*LONG* ***ShowLegend -*** Установка флага отображения легенды и курсора: 1 - отображать | 0 - не отображать

*VARIANT_BOOL* ***EnableYDecibelScale -*** Установка доступности выбора отображения по оси Y в дБ.

*VARIANT_BOOL* ***EnableXOctaveScale -*** Установка доступности отображения октавной шкалы.

*LONG* ***GraphAxis -*** Установка шкалы к которой привязан график | 0 - главная шкала слева / 1 - дополнительная шкала справа.

*LONG* ***DrawHeader -*** Установка рисовать или нет шапку.

LONG ***LegendSize -*** Установка задания размера легенды.

void ***SetEnableYLogScale*** **-** Установка запрета выбора лог шкалы по Y.

© ООО "ЭТМС" 1992-2024. Все права защищены

void ***GetDrawPointsInfoPointer*** **-** Получить координаты точек для рисования сетки

*BSTR* ***FontName*** – название шрифта для надписей.

*long* ***FontSize*** – размер шрифта надписей.

*long* ***CommentInd*** **-** порядковый номер комментария

*BSTR* ***CommentSTR*** –строка комментария.

*long* ***Spectr*** – признак отображения спектра.

Spectr – 1 – вид спектра, можно включить логарифмический масштаб по У, но не отображаются отрицательные координаты по осям, 0 – нормальный вид.

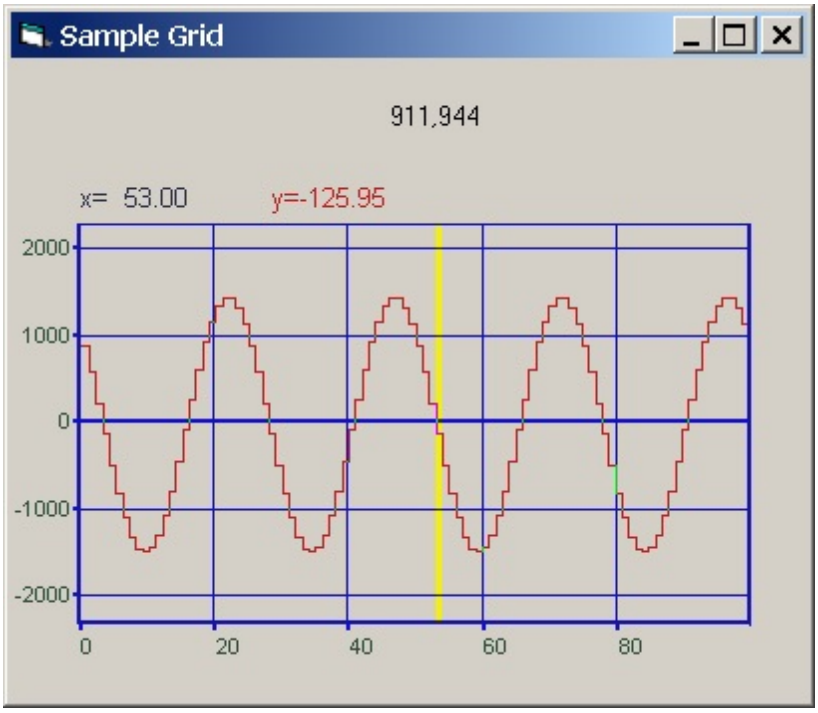
*long* ***Osc*** – признак отображения сигнала, как в осциллографе.

Osc – 1 - Вид осциллографа нельзя включить логарифмический масштаб по X, 0 – нормальный вид.

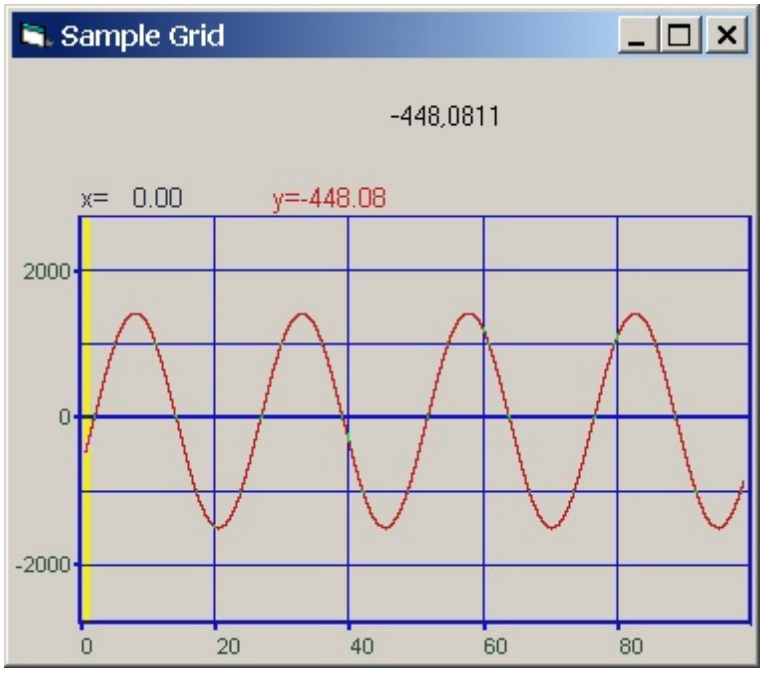
*long* ***TypeLine*** – тип линий графика. Для каждого графика параметр задается отдельно. Графики выбираются при помощи свойства *Ind*.

© ООО "ЭТМС" 1992-2024. Все права защищены

0 – горизонтальными линиями. График рисуется в виде ступенек от точки к точке.




1 – линиями. График рисуется в виде ломаной линии от точки к точке




© ООО "ЭТМС" 1992-2024. Все права защищены

*long* ***TypeGrf*** – тип отображаемого графика. Для каждого графика параметр задается отдельно. Графики выбираются при помощи свойства *Ind*. Параметр может принимать значения:

0 – положительный

1 – знакопеременный.

2 - от предыдущего графика

*long* ***TypeAbs*** – тип осей координат.

0 – рисуется сетка и по X и по Y равномерная шкала по X

1 – логарифмическая сетка по X.

2 – не рисуются горизонтальные линии.

4 – рисуются вертикальные линии.

6 – не рисуются сетка.

*long* ***MakeUpor*** – включение ограничений по оси *Y*. При масштабировании графиков при помощи клавиатуры и мыши на стадии выполнения программы возможна ситуация, когда происходит переполнение данных. Для включения ограничений по масштабированию необходимо установить параметр *MakeUpor* в 1. Для снятия ограничений в 0.

*double* ***UporNis*** – минимальная величина по координате *Y*, которая может отобразиться на графике. Это свойство работает при включенных ограничениях по оси *Y*.

*double* ***UporVerh*** – максимальная величина по координате *Y*, которая может отображаться на графике. Это свойство работает при включенных ограничениях по оси *Y*.

long **PointerToFloatData()** – Получение указателя на отображаемые данные текущего графика, которые хранятся в виде Float.

long **PointerToXCoords()** – Получение указателя на массив оси X. Используется для альтернативных координат X.

**Пример**. Отображение временной реализации оцифрованного АЦП сигнала в виде осциллограммы. Входной диапазон работы АЦП от -10 В до + 10 В, вес младшего разряда 16-разрядного АЦП 0.0003 В. В этом случае можно задать следующие параметры.

MakeUpor = 1

© ООО "ЭТМС" 1992-2024. Все права защищены

*UporNis* = 0.0003
*UporVerh* = 10.0

*double* ***Xfirst*** – начальное значение по координате *X*.

*double* ***Xend*** – конечное значение по координате *X*.

Например, необходимо отобразить массив из 100 чисел в равномерной сетке по оси X. Числа представляют собой спектр мощности сигнала в диапазоне от 1000 Гц до 1100 Гц. Т.е. 0-ому элементу массива соответствует частота 1000 Гц, 99-му – частота 1100 Гц. В этом случае задаются следующие параметры:
*Xfirst* = 1000.
*Xend* = 1100.

long **GraphType - т**ип графика: 0 - основной;1 - предупредительный; 2 - информационный.

*double* ***MathLX*** – текущее минимальное значение по X. Левая граница отображения данных

*double* ***MathDX***– текущий диапазон отображения по X. Длина отображения данных.

Например, необходимо отобразить часть данных из предыдущего примера в диапазоне от 1050 Гц до 1070 Гц. В этом случае задаются следующие параметры:
*MathLX* = 1050.
*MathDX* = 20.
При работе программы в режиме исполнения пользователь может менять область отображения графиков при помощи курсора мыши. Растяжение или сжатие графиков происходит при помощи указателя вида: ↔ , ⇹ – для горизонтальной оси. Сдвинуть графики вправо-влево можно при помощи указателя ← , → – для горизонтальной оси. При этом меняются параметры *MathLX, MathDX.* В пользовательской программе в режиме исполнения можно прочитать текущие параметры области отображения.

*double* ***MathLY*** – текущее минимальное значение по *Y*. Нижняя граница отображения данных.

*double* ***MathDY*** – текущий диапазон отображения по *Y*. Высота отображения данных.

© ООО "ЭТМС" 1992-2024. Все права защищены

Растяжение или сжатие графиков происходит при помощи указателя вида:  ,  – для вертикальной оси. Сдвинуть графики вверх или вниз можно при помощи указателя  ,  – для вертикальной оси. При этом меняются параметры *MathLY, MathDY.* В пользовательской программе в режиме исполнения можно прочитать текущие параметры области отображения. При включенных ограничениях отображения графиков *MakeUpor* = 1 , параметр *MathDY* не может быть меньше *UporNis* , сумма параметров *MathLY* + *MathDY* не может быть больше *UporVerh*.

*void* ***DrawRiftY*** – нарисовать просвет.

*long* ***StaticWidthAxis*** – *устанавливается ширина оси статическая.*

*double* ***Mathrx*** – устанавливается текущее максимальное значение по X.

*BSTR* ***TextHeader*** – *устанавливается текст заголовка (надписи).*

*long* ***EnableBoundaryXvalues*** – *устанавливается возможность отображения граничных значений оси X.*

## Методы GridGL.ocx

Методы можно использовать только во время выполнения программы.

*void* ***AboutBox()*** –содержит информацию о программе.

*void* ***autoScaleY*** *(void)* – однократно применяет автомасштаб по оси Y.

*long* ***PutChnText*** *(BSTR text, LONG index)* - разметка графика.

*long* ***ShowPropSheet*** *(void)* - показывает страницу свойств.

*long* ***InWindPtr*** *(signed char * ptr)* – получает данные в самостоятельную структуру Struct_gridGL настройки.

© ООО "ЭТМС" 1992-2024. Все права защищены

*long* ***OutWindPtr*** *(signed char * ptr)* –выдает данные в самостоятельную структуру Struct_gridGL настройки.

void ***GetMathX_cursor***(*double* * pCursorX) – получает значение курсора (математическое).

*long* ***AItDateStartByFirstTime****(double firstTime, double intervalTimeInSec)* – инициализация массива времён, отображаемого по оси X, по первой точке.

*long* ***GetSelectedGraph()*** – получает индекс выделенного графика.

void ***SetMathX_cursor***(*double* * pCursorX) – устанавливает значение курсора (математическое).

*short* ***Display*** *(void)* – метод перерисовывает графики.

v*oid* ***DrawPicture*** *(unsigned long hdc, long x, long y, long width, long height)* – отрисовывает картинку графика на указанном HDC.

В Windows есть понятия дескрипторов (Handle) окна, дескрипторов контекста устройства и дескрипторов объекта, соответственно английские термины handle window (hWND), handle device context (hDC) и handle object (это общее название). Под hWND понимаются такие элементы как форма, кнопки, панели и т.д., с помощью этого дескриптора элементам передаются сообщения и данные, а вот с помощью hDC производятся графические операции с содержимым самого объекта.

Контекст устройства - структура, которая определяет набор графических объектов, их свойств и способов воздействия на объект. Графические объекты включают перо (pen) для линии, кисть (brush) для заполнения, битовая плоскость (bitmap) для копирования или прокручивания частей экрана, палитра (palette) для определения набора доступных цветов, регион (region) для усечения областей и других действий.

Дескриптор объекта - это вообще независимая штука, она может существовать отдельно от hWND и hDC (эти же работают в паре). К дескрипторам объекта (далее hOBJ) относятся перья системы рисования (Pen - цвет пера, толщина и тип линии), кисть, заполняющая область (Brush - цвет кисти, стиль и вид шаблона заполнения) и т.д.

*void* ***DrawRift*** *(Double riftMathStart, Double riftMathStop, long riftMinWidthPx)* – нарисовать просвет. Рисование выполняется по математическим отсчетам и еще добавляется параметр минимальной ширины.

© ООО "ЭТМС" 1992-2024. Все права защищены

*long* ***ClearGraphs*** (*long value*) – очистить графики на гриде. Перерисовывает каждый график (если графиков два [***SetNumber(2)***] - перерисует оба) массивом из 0, если *value* = 0, или массивом из NAN, если *value* любое число кроме 0.

*void* ***GetMathX_lr***(*Double** pLeftX, *Double** pRigthX) – получение границ отображения по оси X.

*void* ***SetMathX_lr***(*Double* leftX, *Double* rigthX) – задание границ отображения по оси X.

*long* ***ClearGraph*** (*long value*) – очистить данный график на гриде. Перерисовывает график массивом из 0, если *value* = 0, или массивом из NAN, если *value* любое число кроме 0.

*long* ***GetIndexX_lr(****long* pIndexLeft, long* pIndexRight****)*** – получение индексов крайних отображаемых точек.

*long* ***SetIndexX_lr(****long indexLeft, long indexRight****)*** – задание индексов крайних отображаемых точек.

*long* ***GetIndexX_lcr(****long* pIndexLeft, long* pIndexCusor, long* pIndexRight)* – получение индексов крайних отображаемых точек и курсора.

*long* ***SetIndexX_lcr(****long indexLeft, long indexCusor, long indexRight****)*** – задание индексов крайних отображаемых точек и курсора.

*long* ***Paint*** *(float* Buffer)* – метод, позволяет передать данные для отображения графика и перерисовать график. При построении нескольких графиков в одних осях необходимо при помощи свойства *Ind* выбирать, какой график требует перерисовки.

Параметры:

*float* Buffer* – адрес массива данных в программе пользователя. В компоненту копируется количество данных определенное свойством *Size*.

При создании программ обработки сигналов в реальном времени с многократной перерисовкой графиков, типа осциллографов или анализаторов спектра, необходимо один раз задать параметры компонента, и затем последовательно обращаться только к методу *Paint(...)* для создания эффекта движения.

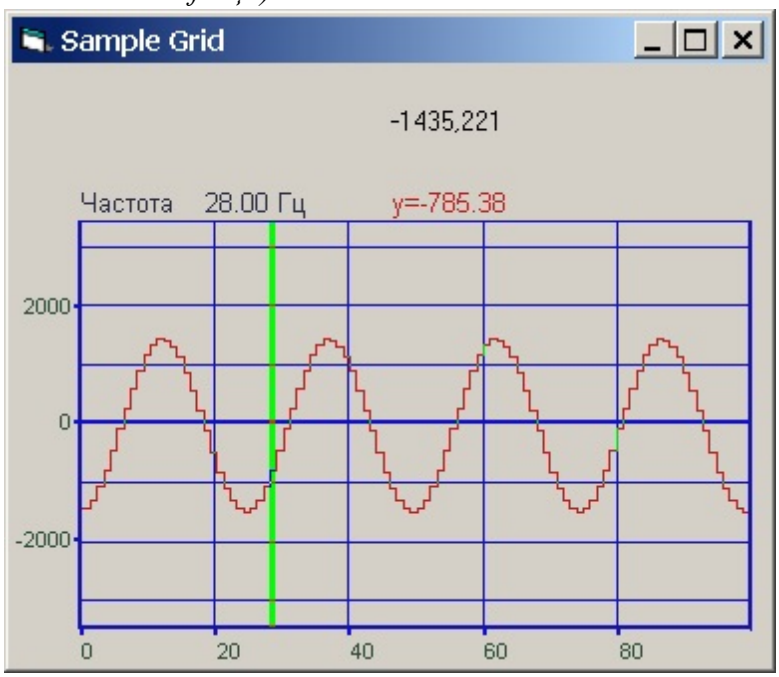
*long* ***FormatX*** *(BSTR fmt)* – метод передает в компонент строку для форматного вывода информации по координате *X* в легенде графика. Форматный ввод-вывод

© ООО "ЭТМС" 1992-2024. Все права защищены

позволяет передавать программе и выдавать из программы на внешнее устройство символьные, строковые и числовые значения, обеспечивая преобразование данных в процессе ввода-вывода.

Например, обращение вида:

*FormatX("Частота %7.2f Гц")*




позволяет отображать координаты по оси X в значениях по частоте.

Формат представления чисел.

*%f* – общий формат представления числа в плавающей запятой в виде числа с точкой

*%e* – общий формат представления числа в плавающей запятой в виде числа с экспонентой (*1000 = 1e3, 0.001 = 1e-3*)

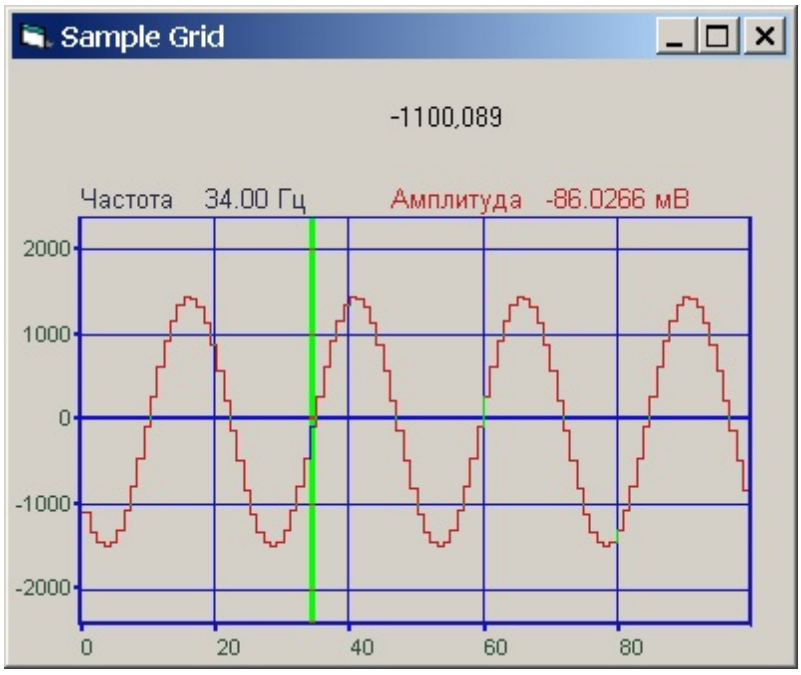
*%7.2f* – представление числа в фиксированном формате. В данном примере *7* – количество знакомест для представления числа, *2* – количество цифр после запятой в представлении числа.

*long* ***FormatY*** *(BSTR fmt)* – метод передает в компонент строку для форматного вывода информации по координате *Y* в легенде графика. Форматный ввод-вывод позволяет передавать программе и выдавать из программы на внешнее устройство символьные, строковые и числовые значения, обеспечивая преобразование данных в процессе ввода-вывода. Метод должен выполняться для каждого графика. Графики выбираются при помощи свойства *Ind*.

Например, обращение вида:

*FormatY ("Амплитуда %9.4f мВ")*

© ООО "ЭТМС" 1992-2024. Все права защищены




позволяет отображать координаты по оси *Y* в значениях по амплитуде.

*void* ***Formatxy*** *(BSTR fmt)* – метод содержит надпись в графике. Форматный ввод-вывод позволяет передавать программе и выдавать из программы на внешнее устройство символьные, строковые и числовые значения, обеспечивая преобразование данных в процессе ввода-вывода. Это может быть произвольная пользовательская информация о данном графике.

*void* ***AltC*** *(float *buffer)* – загружает массив разметки данных по X. Этот метод работает при установленном свойстве типа сетки *TypeAbs* = 2. В пользовательском буфере *buffer* размера *Size* необходимо записать значения координаты *X*. Например, при логарифмической развертке по оси *X*, в буфере могут быть записаны числа:

*buffer[0] = 1.*
*buffer[1] = 1.25*
*buffer[2] = 1.6*
*buffer[3] = 2.*
*buffer[4] = 2.5*
……

*void* ***AltDate*** *(DATE * buffer)* - позволяет загрузить массив дат для разметки данных по X

Параметры:

* *buffer* – адрес данных в программе пользователя. В компоненту копируется количество данных определенное свойством *Size*.

При создании программ обработки сигналов в реальном времени с многократной перерисовкой графиков, типа осциллографов или анализаторов спектра, необходимо

© ООО "ЭТМС" 1992-2024. Все права защищены

один раз задать параметры компонента, и затем последовательно обращаться только к методу *Paint(...)* для создания эффекта движения.

*long* ***PaintNet*** *(long pBuffer)* – метод, позволяет передать данные для отображения графика и перерисовать график. При построении нескольких графиков в одних осях необходимо при помощи свойства *Ind* выбирать, какой график требует перерисовки.

Параметры:

*long pBuffer* – адрес числового массива данных в программе пользователя. В компоненту копируется количество данных определенное свойством *Size*.

При создании программ обработки сигналов в реальном времени с многократной перерисовкой графиков, типа осциллографов или анализаторов спектра, необходимо один раз задать параметры компонента, и затем последовательно обращаться только к методу *Paint(...)* для создания эффекта движения.

*void* ***AltCNet*** *(long pBuffer)* – метод, позволяет задать значения по оси *X*. Этот метод работает при установленном свойстве типа сетки *TypeAbs* = 2. В пользовательском буфере *buffer* размера *Size* необходимо записать значения координаты *X*. Например, при логарифмической развертке по оси *X*, в буфере могут быть записаны числа:

*buffer[0] = 1.*
*buffer[1] = 1.25*
*buffer[2] = 1.6*
*buffer[3] = 2.*
*buffer[4] = 2.5*

……

*void* ***AltDateNet*** *(long pBuffer)* – позволяет загрузить массив дат для разметки данных по X

Параметры:

*long pBuffer* – адрес данных в программе пользователя. В компоненту копируется количество данных определенное свойством *Size*.

При создании программ обработки сигналов в реальном времени с многократной перерисовкой графиков, типа осциллографов или анализаторов спектра, необходимо один раз задать параметры компонента, и затем последовательно обращаться только к методу *Paint(...)* для создания эффекта движения.

**Примечание:** Следует отметить, то при работе в .net языках функции, в которых в качестве параметров выступают указатели на массив, например GetData в .net языках работают некорректно, поэтому следует использовать обновленные функции, имеющие такое же название + приписка Net, например, GetDataNet.

Методы, возвращающие индексы первого и последнего значений графика, используемых в логарифмических данных.

*long* ***getStartPoint*** *(void)* – возвращает индекс первой видимой точки.

© ООО "ЭТМС" 1992-2024. Все права защищены

*long* ***getEndPoint*** *(void)* – возвращает индекс последней видимой точки.

*DOUBLE* ***GetStep*** *(void)* – определяет число пикселей на отсчет.

*long* ***VisibleWidthPx*** *(void)* – получает ширину сетки в пикселях. Добавление метода получения ширины таблицы

*VARIANT_BOOL* ***IsShowPropSheet*** - скрывает нулевые значения на графике.

*void* ***PushDataToClipBoard*** *(void)* - позволяет сохранить изображение графика в Clipboard в графическом (bmp) формате.

*void* ***PushToClipBoard****(void)* - позволяет загрузить график в ClipBoard в графическом (bmp) формате.

*long* ***SaveGLDataToFile*** *(BSTR FileName)* – метод, позволяет сохранить массив данных в файле. Сохранить изображение можно с расширением *.dtu; *.dtx ; *.png; *.xls; *.csv" .

Файл записывается в текстовом виде со всеми комментариями. Это позволяет использовать данные из этого файла для других программ обработки. *FileName* – строка полного пути и имени файла.

В файл записываются разметка по оси X и значения всех графиков в текстовом виде. Это позволяет использовать данные из этого файла для других программ обработки. По-умолчанию файл открывается программой "Просмотр результатов". Это для *.dtu и *.dtx.

"*.xls". Файл записывается в формате EXCEL. В таблицу EXCEL записывается заголовок графика, разметка по оси X и данные графиков. Наличие установленных программы Microsoft Office Excel или подобных для сохранения файла не требуется.

*long* ***SaveGridToFile*** *(BSTR FileName)* - сохранить график в файл BMP. Это позволяет использовать данные из этого файла для других программ обработки. *FileName* – строка полного пути и имени файла.

long ***PaintDouble*** *(DOUBLE* buffer)* - загрузка выбранного графика (тип double).

long ***SaveGLDataToFileLimit*** *(BSTR FileName, INT startIndex, INT endIndex)* - позволяет сохранить изображение в текстовый файл с расширением *.dtx

long ***GetGridGLBitmapInfo***(LONG* pInfo) - позволяет получить данные о графике в виде структуры BITMAPINFO

© ООО "ЭТМС" 1992-2024. Все права защищены

long ***CopyGridGLBitmapData***(BYTE* pData, LONG size) - позволяет получить изображение графика в виде указателя на структуру BITMAP.

long ***SaveGridGLToMetaFile***(BSTR FileName) - позволяет сохранить сетку как метафайл с расширением WMF или EMF.

*FileName* – строка полного пути и имени файла.

BSTR ***GetFormatSaveGrid()*** - позволяет получить формат для сохранения графика.

long **SetCursorSetting**(LONG index, LONG shape, LONG viewLine, LONG visible, LONG indexGraph, ULONG color) - позволяет создавать вспомогательные курсоры:

index - индекс курсора \

shape - форма курсора \

viewLine - отображение линии до курсора \

visible - отображение всего курсора \

indexGraph - индекс графика, к которому относится курсор \

color - цвет курсора".

long ***SetPositionCursor***(LONG index, LONG positionX, DOUBLE positionY) - позволяет изменять положения вспомогательных курсоров:

index - индекс курсора \

positionX - позиция курсора по оси X (значение от 0 до размера, переданного в grid) \

positionY - позиция курсора по оси Y (не в пикселях)".

void ***ClearCollectionCursors()*** - позволяет скрыть все вспомогательные курсоры.

long ***Paint_s***(FLOAT * buffer, LONG size) - позволяет загрузить выбранный график с учетом размера (тип double).

long ***PaintDouble_s***(DOUBLE * buffer, LONG size) - позволяет загрузить выбранный график с учетом размера (тип float).

void ***AItCDouble***(DOUBLE* buffer) - позволяет загрузить массив разметки данных по X.

void ***AItCDouble***(DOUBLE* buffer) - позволяет загрузить массив разметки данных по X.

void ***UsingAddCursors(LONG flag)*** -позволяет использовать курсоры.

© ООО "ЭТМС" 1992-2024. Все права защищены

void ***GetMathX_Icr***(DOUBLE* pLeftX, DOUBLE* pCursorX, DOUBLE* pRigthX) - позволяет получить границы отображения по оси X и положения курсора:

pLeftX - значение левой точки границы по шкале X \

pCursorX - значение точки границы курсора по шкале X \

pRigthX - значение правой точки границы по шкале X.

long ***SetMathX_Icr***(DOUBLE leftX, DOUBLE cursorX, DOUBLE rigthX) - позволяет задать границ отображения по оси X и положения курсора

pLeftX - значение левой точки границы по шкале X \

pCursorX - значение точки границы курсора по шкале X \

pRigthX - значение правой точки границы по шкале X.

long ***LoadGLDataFromFile***( BSTR FileName )- позволяет загрузить структуры графика из текстового файла с расширением *.dtx.

*FileName* – строка полного пути и имени файла.

long ***AltDateStart***(DOUBLE lastTime, DOUBLE intervalTimeInSec) - позволяет инициализировать массив времён, отображаемого по оси X, по последней точке

long ***AltDateStart***() - позволяет сдвигать массивы времён, отображаемого по оси X, на одну точку.

long ***AltDateNewTimeInterval***(LONG sizeInterval)- позволяет сдвигать массива времён, отображаемого по оси X, на несколько точек.

long ***AltDateStartByFirstTime***(DOUBLE firstTime, DOUBLE intervalTimeInSec)- инициализация массива времён, отображаемого по оси X, по первой точке

Добавили возможность установки меток, привязанных к значению отсчетов. Пока данная функция используется только для вывода комментариев к папке в Галерее сигналов

*VARIANT_BOOL* ***AddToolTip*** *(BSTR text, LONG pos)* – состояние включенности/выключенности метки.

*VARIANT_BOOL* ***RemoveAllToolTips*** *(void)* – состояние изменения включенности/выключенности всех меток.

© ООО "ЭТМС" 1992-2024. Все права защищены

Добавили маркеры, а также функции для управления ими.

*long* ***AddMark****(DOUBLE x, BSTR text)* - добавить текстовую метку на график.

*long* ***EditMark****(LONG index, BSTR newText)* - редактировать текст метки на графике.

*long* ***GetMark****(LONG index, DOUBLE* x, BSTR* text)*- получить информацию о метке на графике.

*long* ***MoveMark****(LONG index, DOUBLE newX)* - передвинуть метку на новую позицию.

*long* ***RemoveMark****(LONG index)* - удалить метку

Добавили возможность установки выделения полос на графике.

*long* ***SetColorLineCount****(LONG Count)* – устанавливает количество полос с отображением событий.

*long* ***SetColorLineFont****(LONG Number, OLE_COLOR _Color)* – устанавливает общий фон для полосы с номером Number.

*long* ***SetColorLineName****(LONG Number, BSTR Name)* - устанавливает текст для полосы с номером Number.

*long* ***SetColorLineSpans****(LONG Number, LONG* IndexBeginArray, LONG* IndexEndArray, ULONG* ColorsArray, LONG ArraySize)*

- устанавливает временные интервалы и цвета для интервалов на которые будет разбита полоса: \
Number - номер полосы \
IndexBeginArray - массив начальных индексов с которых будет начинаться разбиение для каждого интервала \
IndexEndArray = массив конечных индексов до которых будет идти каждый интервал \
ColorsArray - массив цветов для каждого интервала, подается !!!OLE_COLOR *!!! для каждого интервала \
ArraySize общее количество интервалов

*long* ***SetColorLineSpansByTime****(LONG Number, DATE* DateBeginArray, DATE* DateEndArray, ULONG* ColorsArray, LONG ArraySize)*

- устанавливает временные интервалы и цвета для интервалов на которые будет разбита полоса: \

© ООО "ЭТМС" 1992-2024. Все права защищены

Number - номер полосы \
DateBeginArray - массив начальных дат с которых будет начинаться разбиение для каждого интервала \
DateEndArray = массив конечных дат до которых будет идти каждый интервал \
ColorsArray - массив цветов для каждого интервала, подается !!!OLE_COLOR *!!! для каждого интервала \
ArraySize общее количество интервалов

Добавили возможность установки индикационной рамки на графике.

void ***ShowBorderIndication***( LONG isShow ) – устанавливает отображение индикационной рамки.

void ***SetParametresBorderIndication***( OLE_COLOR color, LONG alpha, LONG width ) – устанавливает параметры индикационной рамки

color - цвет;
alpha - прозрачность;
width - ширина.

## События от компоненты GridGL.ocx

*void* ***DblClick*** *(void)* – событие возникает при двойном нажатии на кнопку манипулятора «мышь», при нахождении мыши над графиком.

*void* ***KeyDown*** *(SHORT* KeyCode, SHORT Shift)* – событие возникает при нажатии на клавишу клавиатуры, когда фокус находится на компоненте.

Параметры:
*KeyCode* – код клавиши
*Shift* – код нажатия служебных клавиш *<Ctrl>,<Shift>,<Alt>*.

**Примечание управляющим клавишам:**

<T> - *Копирование сопроводительной информации.* Для копирования сопроводительной информации нажать левой кнопкой «мыши» на поле графика программы и нажать на кнопку клавиатуры <T>(в латинской раскладке клавиатуры). Сопроводительная информация скопируется в буфер обмена(Clipboard).

<C> - *Копирование графика.* Для копирования графика нажать левой кнопкой «мыши» на поле графика и нажать комбинацию «горячих клавиш» клавиатуры <Ctrl> +<C> .График запишется в буфер обмена(Clipboard).

<N> *Копирование численных значений.* Для копирования численных значений нажать левой кнопкой «мыши» на поле графика программы и

© ООО "ЭТМС" 1992-2024. Все права защищены

нажать на кнопку клавиатуры <N>(в латинской раскладке клавиатуры). Численные значения скопируются в буфер обмена(Clipboard).

<A> *Управление движением курсора влево.* Нужно нажать левой кнопкой «мыши» на поле графика программы и нажать на кнопку клавиатуры <A>(в латинской раскладке клавиатуры)

или движением колесика мыши вниз.

<D> *Управление движением курсора вправо.* Нужно нажать левой кнопкой «мыши» на поле графика программы и нажать на кнопку клавиатуры <D>(в латинской раскладке клавиатуры).

или движением колесика мыши вверх.

*void* ***MouseDown*** *(short Button, short Shift, OLE_XPOS_PIXELS x, OLE_YPOS_PIXELS y)* – событие возникает при нажатии на кнопку манипулятора типа «мышь», при нахождении мыши над графиком.

Параметры:
*Button* – код кнопки мыши
*Shift* – код нажатия служебных клавиш *<Ctrl>,<Shift>,<Alt>*.
*x* – координата по оси *X* в пикселах (разрешении экрана)
*y* – координата по оси *Y* в пикселах (разрешении экрана)

*void* ***MouseUp*** *(short Button, short Shift, OLE_XPOS_PIXELS x, OLE_YPOS_PIXELS y)* – событие возникает при отжатии кнопки манипулятора типа «мышь», при нахождении мыши над графиком.

Параметры:
*Button* – код кнопки мыши
*Shift* – код нажатия служебных клавиш *<Ctrl>,<Shift>,<Alt>*.
*x* – координата по оси *X* в пикселях (разрешении экрана)
*y* – координата по оси *Y* в пикселях (разрешении экрана)

void ***MouseWheel*** *(void)* – событие возникает при вращении ролика манипулятора типа «мышь».

*void* ***Modify*** *(void)* - событие, которое происходит при включении и отключении каналов АЦП, ЦАП, виртуальных каналов, изменении коэффициентов усиления и т.д.

*void* ***ChangeSetting****(void)* – событие, которое возникает при изменение настроек *gridGL.*

void ***OnAutoScale()*** – событие, которое возникает при клике мышью по зоне AutoScale

void ***OnChangeRect***(LONG changeX, LONG changeY) - событие, которое возникает при изменении прямоугольника с сеткой

**Примечание:**

*Копирование графика*

© ООО "ЭТМС" 1992-2024. Все права защищены

Для копирования графика нажать левой кнопкой «мыши» на поле графика и нажать комбинацию «горячих клавиш» клавиатуры <Ctrl> + <C>. График запишется в буфер обмена (Clipboard). Вставить график в любой открытый документ Microsoft Word или Excel можно нажатием «горячих клавиш» клавиатуры <Ctrl > + <V>, или нажатием на правую кнопку «мыши» и выбором в появившемся меню команды Вставить.

*Копирование сопроводительной информации*

Для копирования сопроводительной информации нажать левой кнопкой «мыши» на поле графика программы и нажать на кнопку клавиатуры <T> (в латинской раскладке клавиатуры). Вставить эту информацию в любой открытый текстовый документ можно нажатием «горячих клавиш» клавиатуры <Ctrl > + <V>, или нажатием на правую кнопку «мыши» и выбором в появившемся меню команды Вставить.

*Копирование численных значений*

Для копирования численных значений видимой части графика нажать левой кнопкой «мыши» на поле графика и нажать на кнопку клавиатуры <N> (в латинской раскладке клавиатуры). Вставить эту информацию в любой открытый текстовый документ можно нажатием «горячих клавиш» клавиатуры <Ctrl> + <V>, или нажатием на правую кнопку «мыши» и выбором в появившемся меню команды Вставить.

*Движение курсора вправо*

Движение курсора вправо нажать левой кнопкой «мыши» на поле графика и нажать на кнопку клавиатуры <D> (в латинской раскладке клавиатуры).

*Движение курсора влево*

Движение курсора влево нажать левой кнопкой «мыши» на поле графика и нажать на кнопку клавиатуры <A> (в латинской раскладке клавиатуры).

*Сохранения гриды в виде изображения*

Сохранения гриды в виде изображения по нажатию на кнопку "F".

## 2.4.Краткий перечень функций, используемые GridGL.ocx

| **Свойства GridGL.ocx** |
|---|

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| **Цвета** | |
| *long* ***GetClrCrs****()* | опрашивается цвет курсора. |
| *void* ***SetClrCrs****(unsigned long propVal)* | устанавливается цвет курсора. |
| *long* ***GetClrDig****()* | опрашивается цвет цифр на сетке. |
| *void* ***SetClrDig****(unsigned long propVal)* | устанавливается цвета цифр на сетке. |
| *long* ***GetClrFon*** *()* | опрашивается цвет фона. |
| *void* ***SetClrFon*** *(unsigned long propVal)* | устанавливается цвет фона. |
| *long* ***GetClrGrd*** *()* | опрашивается цвет сетки. |
| *void* ***SetClrGrd*** *(unsigned long propVal)* | устанавливается цвет сетки. |
| *long* ***GetClrGrf*** *()* | опрашивается цвет графика. |
| *void* ***SetClrGrf*** *(unsigned long propVal)* | устанавливается цвет графика. Для каждого графика цвет задается отдельно. Графики выбираются при помощи свойства ***Ind***. |
| *long* ***GetClrLeg*** *()* | опрашивается цвет легенды графика. |
| *void* ***SetClrLeg*** *(unsigned long propVal)* | устанавливается цвет легенды графика. |
| **Параметры графиков** | |
| *long* ***GetNumber****()* | опрашивается количество отображаемых графиков. Количество одновременно отображаемых графиков в одних осях не может превышать 60. |
| *void* ***SetNumber****(long propVal)* | устанавливается количество отображаемых графиков. Количество одновременно отображаемых графиков в одних осях не может превышать 60. |
| *long* ***GetInd****()* | опрашивается индекс (номер) графика. Индекс может принимать значения от 0 до Number-1. |
| *void* ***SetInd*** *(long propVal)* | устанавливается индекс (номер) графика. Индекс может принимать значения от 0 до Number-1. |
| *long* ***GetValid****()* | опрашивается прорисовки графика (0 – не рисует выбранный график,1 – рисует выбранный график). |
| *void* ***SetValid****(long propVal)* | устанавливается прорисовки графика (0 – не рисует выбранный график,1 – рисует выбранный график). |
| *long* ***GetSize*** *()* | опрашивается количество точек по горизонтали. Размер не может быть больше 500000. |
| *void* ***SetSize*** *(long propVal)* | устанавливается количество точек по горизонтали. Размер не может быть больше 500000. |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *long* ***GetNumVisiblePoints*** *()* | опрашивается количество видимых точек в графике. |
| *void* ***SetNumVisiblePoints*** *(long propVal)* | устанавливается количество видимых точек в графике. |
| *long* ***GetIog*** *()* | опрашивается тип шкалы Y (0 - линейная или 1 - логарифмическая). |
| *void* ***SetIog*** *(long propVal)* | устанавливается тип шкалы Y (0- линейная или 1 - логарифмическая). |
| *double* ***GetReference****()* | опрашивается опорное значение для расчета уровня в дБ. |
| *void* ***SetReference****(double propVal)* | устанавливается опорное значение для расчета уровня в дБ. |
| *long* ***GetTypeFiII*** *()* | опрашивается тип заполнения графика (0 – с заполнением, 1 – без заполнения). |
| *void* ***SetTypeFiII*** *(long propVal)* | устанавливается тип заполнения графика (0 – с заполнением, 1 – без заполнения). |
| *long* ***GetLineWidth*** *()* | опрашивается толщина линии графика. |
| *void* ***SetLineWidth*** *(long propVal)* | устанавливается толщина линии графика. |
| *long* ***GetPx****()* | Опрашивается позиция курсора по оси X. Этот параметр принимает значение от 0 до Size-1. |
| *void* ***SetPx*** *(long propVal)* | устанавливается позиция курсора по оси X. Этот параметр принимает значение от 0 до Size-1. |
| *VARIANT_BOOL* ***HideZeros*** | Скрывать нулевые значения (0 – скрывать, 1 – без скрытия). |
| *long* ***LeftBorderGrid*** | устанавливается левая граница GridGL. |
| *long* ***ShowCursor*** | устанавливается отображение основного курсора. |

## Параметры шкалы и дополнительного окна

| | |
|---|---|
| *long* ***GetIog*** *()* | опрашивается тип шкалы Y (0 - линейная или 1 - логарифмическая). |
| *void* ***SetIog*** *(long propVal)* | устанавливается тип шкалы Y (0 - линейная или 1 - логарифмическая). |
| *char* ***GetUporDelta*** *()* | опрашивается минимальное разрешение по вертикали (при включенном ограничении). |
| *void* ***SetUporDelta****(double propVal)* | устанавливается минимальное разрешение по вертикали (при включенном ограничении). |
| *long* ***GetTypeXAxis****()* | опрашивается тип шкалы по оси X (0 – линейная, 1 – логарифмическое, 2 - долеоктавная). |
| *void* ***SetTypeXAxis****(long* | устанавливается тип шкалы по оси X (0 – линейная, 1 – логарифмическое, 2 - долеоктавная). |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *propVal)* | |
| *long* ***GetTypeYAxis****()* | опрашивается тип шкалы по оси Y (0 – линейная, 1 – логарифмическое,). |
| *void* ***SetTypeYAxis****(long propVal)* | устанавливается тип шкалы по оси Y(0 – линейная, 1 – логарифмическое). |
| *long* ***GetShowScalePropPage****()* | Опрашивается страница свойств шкалы. |
| *void* ***SetShowScalePropPage****(long propVal)* | устанавливается страница свойств шкалы. |
| *BOOL* ***GetIsShowPropSheet****()* | опрашивается состояние выводить/не выводить на экран страницу свойств, где пользователь может установить режим работы, вид шкалы, цвет фона, тип шрифта, максимальное и минимальное значения. |
| *void* ***SetIsShowPropSheet****(BOOL propVal)* | устанавливается состояние выводить/не выводить на экран страницу свойств, где пользователь может установить режим работы, вид шкалы, цвет фона, тип шрифта, максимальное и минимальное значения. |
| *long* ***GetTypeYData****()* | опрашивается тип данных по Y (0-линейная; 1-логарифмическая; 3- децибельная). |
| *void* ***SetTypeYData****(long propVal)* | устанавливается тип данных по Y (0-линейная; 1-логарифмическая; 3- децибельная). |
| *CString* ***GetTextXAxis****()* | опрашивается единица измерения по оси X. |
| *void* ***SetTextXAxis****(CString propVal)* | устанавливается единица измерения по оси X. |
| *CString* ***GetTextYAxis****()* | опрашивается единица измерения по оси Y. |
| *void* ***SetTextYAxis****(CString propVal)* | устанавливается единица измерения по оси Y. |
| *long* ***GetAltCoords****()* | опрашивается равномерные и неравномерные координаты по X\n0-равномерные\n1-координаты задаются в методе AltC(float* buffer). |
| *void* ***SetAltCoords****(long propVal)* | устанавливается равномерные и неравномерные координаты по X\n0-равномерные\n1-координаты задаются в методе AltC(float* buffer). |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *VARIANT_BOOL* ***IsShowPropSheet*** | Показывать или не показывать страницу настроек (0 - показывать; 1 - не показывать). |
| *VARIANT_BOOL* ***PowerDecibell*** | При отображении децибел использовать формулу мощности (0 - использовать; 1 - не использовать). |
| *long* ***SetAxisSetting****(LONG Index, LONG UseAxis, BSTR AxisText, OLE_COLOR AxisColor, FLOAT CoeffMultiplicative, FLOAT CoeffAdditive)* | Установка параметров для шкалы, \Index - индекс настраиваемой оси (от 0 до 4) привязку осей к индексу можно посмотреть в GridGL::AxisSide, \<br>UseAxis - 1 использовать шкалу / 0 не использовать,\<br><br>AxisText - Текст на шкале, AxisColor - цвет шкалы и надписи, \<br><br>CoeffMultiplicative и CoeffAdditive - коэффициенты на которые отличаются значения шкалы. |
| *long* ***SetAdditionalWindowSettings****(LONG Index, LONG UseWindow, LONG GraphNumber, LONG IndexPoint, OLE_COLOR TextColor, OLE_COLOR BackgroundColor, BSTR WindowText)* | Установка параметров для дополнительного окна, \<br>Index - индекс настраиваемого окна (от 0 до 99), \<br><br>UseWindow - 1 использовать окно / 0 не использовать, \<br><br>GraphNumber - номер графика, к которому привязано окно, \<br><br>IndexPoint - индекс точки на графике к которой привязано окно, \<br><br>TextColor и BackgroundColor - цвета использующиеся при отрисовке окна, \<br><br>WindowText - текст в окне. |
| *void* ***DrawRiftY****(DOUBLE riftMathStartY, DOUBLE riftMathStopY, LONG riftMinWidthPy);* | Установка параметров для нарисовать просвет. |
| *LONG* ***ShowLegend*** | Установка флага отображения легенды и курсора: 1 - отображать \| 0 - не отображать. |
| *VARIANT_BOOL* ***EnableYDecibelScale*** | Установка доступности выбора отображения по оси Y в дБ. |
| *VARIANT_BOOL* ***EnableXOctaveScale*** | Установка доступности отображения октавной шкалы. |
| *LONG* ***GraphAxis*** | Установка шкалы к которой привязан график \| 0 - главная шкала слева / 1 - дополнительная шкала справа. |
| *LONG* ***DrawHeader*** | Установка рисовать или нет шапку. |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| LONG ***LegendSize*** | Установка задания размера легенды. |
| void ***SetEnableYLogScale*** (VARIANT_BOOL flag); | Установка запрета выбора лог шкалы по Y. |
| void ***GetDrawPointsInfoPointer*** | Получить координаты точек для рисования сетки. |
| **Параметры надписей** | |
| *CString* ***GetFontName****()* | опрашивается название шрифта для надписей. |
| *void* ***SetFontName****(CString propVal)* | устанавливается название шрифта для надписей. |
| *long* ***GetFontSize****()* | опрашивается шрифт надписей. |
| *void* ***SetFontSize****(long propVal)* | устанавливается надписей. |
| *long* ***GetCommentInd****()* | опрашивается порядковый номер комментария. |
| *void* ***SetCommentInd****(long propVal)* | устанавливается порядковый номер комментария. |
| *CString* ***GetCommentSTR****()* | опрашивается строка комментария. |
| *void* ***SetCommentSTR****(CString propVal)* | устанавливается строка комментария. |
| **Параметры отображения** | |
| *long* ***GetSpectr****()* | опрашивается признак отображения спектра. (1 – вид спектра, можно включить логарифмический масштаб по У, но не отображаются отрицательные координаты по осям, 0 – нормальный вид). |
| *void* ***SetSpectr****(long propVal)* | устанавливается признак отображения спектра(1 – вид спектра, можно включить логарифмический масштаб по У, но не отображаются отрицательные координаты по осям, 0 – нормальный вид). |
| *long* ***GetOsc****()* | опрашивается признак отображения сигнала, как в осциллографе. (1 - Вид осциллографа нельзя включить логарифмический масштаб по X, 0 – нормальный вид). |
| *void* ***SetOsc****(long propVal)* | Устанавливается признак отображения сигнала, как в осциллографе(1 - Вид осциллографа нельзя включить логарифмический масштаб по X, 0 – нормальный вид). |

© ООО "ЭТМС" 1992-2024. Все права защищены

| **Представление графиков** | |
|---|---|
| *long* ***GetTypeLine****()* | опрашивается тип линий графика (0-горизонтальными линиями, 1 – ломаными линиями). |
| *void* ***SetTypeLine****(long propVal)* | устанавливается тип линий графика (0-горизонтальными линиями, 1 – ломаными линиями). |
| *long* ***GetTypeGrf****()* | опрашивается тип отображаемого графика ( 0 – положительный,1 – знакопеременный, 2 - от предыдущего графика). |
| *void* ***SetTypeGrf****(long propVal)* | устанавливается тип отображаемого графика ( 0 – положительный,1 – знакопеременный, 2 - от предыдущего графика). |
| *long* ***GetMakeUpor****()* | опрашивается включение ограничений по оси Y. Для включения ограничений по масштабированию необходимо установить параметр MakeUpor в 1. Для снятия ограничений в 0. |
| *void* ***SetMakeUpor****(long propVal)* | устанавливается включение ограничений по оси Y. Для включения ограничений по масштабированию необходимо установить параметр MakeUpor в 1. Для снятия ограничений в 0. |
| *long* ***GetTypeAbs****()* | опрашивается тип осей координат (0 – рисуется сетка и по X и по Y (равномерная шкала по X), 1 – логарифмическая сетка по X, 2 – не рисуются горизонтальные линии, 4 – рисуются вертикальные линии, 6 – не рисуется сетка). |
| *void* ***SetTypeAbs****(long propVal)* | устанавливается тип осей координат (0 – рисуется сетка и по X и по Y (равномерная шкала по X), 1 – логарифмическая сетка по X, 2 – не рисуются горизонтальные линии, 4 – рисуются вертикальные линии, 6 – не рисуется сетка). |
| *double* ***GetUporNis****()* | опрашивается минимальная величина по координате Y, которая может отобразиться на графике. Это свойство работает при включенных ограничениях по оси Y. |
| *void* ***SetUporNis****(double propVal)* | устанавливается минимальная величина по координате Y, которая может отобразиться на графике. Это свойство работает при включенных ограничениях по оси Y. |
| *double* ***GetUporVerh****()* | опрашивается максимальная величина по координате Y, которая может отобразиться на графике. Это свойство работает при включенных ограничениях по оси Y. |
| *void* ***SetUporVerh****(double propVal)* | устанавливается максимальная величина по координате Y, которая может отобразиться на графике. Это свойство работает при включенных ограничениях по оси Y. |
| *double* ***GetXfirst****()* | опрашивается начальное значение по координате *X*. |
| *void* ***SetXfirst****(double propVal)* | устанавливается начальное значение по координате *X*. |
| *double* ***GetXend****()* | опрашивается конечное значение по координате *X*. |
| *void* ***SetXend****(double propVal)* | устанавливается конечное значение по координате *X*. |
| *long* ***GetPointerToXCoords****()* | опрашивается получение указателя на массив оси X. Используется для альтернативных координат X. |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *void* ***SetPointerToXCoords*** *(long propVal)* | устанавливается получение указателя на массив оси X. Используется для альтернативных координат X. |
| *long* ***GetPointerToFloatData()*** | опрашивается получение указателя на отображаемые данные текущего графика, которые хранятся в виде float. |
| *void* ***SetPointerToFloatData****(long propVal)* | устанавливается получение указателя на отображаемые данные текущего графика, которые хранятся в виде float. |
| *void* ***AutoScaleX()*** | устанавливается однократно применить автомасштаб по X. |
| *long* ***GraphType*** | устанавливается тип графика: 0 - основной;1 - предупредительный; 2 - информационный. |
| **Область отображении графиков** | |
| *double* ***GetMathlx****()* | опрашивается текущее минимальное значение по X. Левая граница отображения данных. |
| *void* ***SetMathlx****(double propVal)* | устанавливается текущее минимальное значение по X. Левая граница отображения данных. |
| *double* ***GetMathdx****()* | опрашивается текущий диапазон отображения по X. Длина отображения данных. |
| *void* ***SetMathdx****(double propVal)* | устанавливается текущий диапазон отображения по X. Длина отображения данных. |
| *double* ***GetMathly****()* | опрашивается текущее минимальное значение по Y. Нижняя граница отображения данных. |
| *void* ***SetMathly****(double propVal)* | Устанавливается текущее минимальное значение по Y. Нижняя граница отображения данных. |
| *double* ***GetMathdy****()* | опрашивается текущий диапазон отображения по Y. Высота отображения данных. |
| *void* ***SetMathdy****(double propVal)* | устанавливается текущий диапазон отображения по Y. Высота отображения данных. |
| *void* ***DrawRiftY****(double riftMathStartY, double riftMathStopY, long riftMinWidthPy)* | нарисовать просвет. |
| *long* ***StaticWidthAxis*** | устанавливается ширина оси статическая. |
| *double Mathrx* | устанавливается текущее максимальное значение по X. |
| *BSTR* ***TextHeader*** | устанавливается текст заголовка (надписи). |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *long* ***EnableBoundaryXvalues*** | устанавливается возможность отображения граничных значений оси X. |

## Методы GridGL.ocx

### Настройка GridGL

| | |
|---|---|
| *void* ***AboutBox****()* | содержит информацию о программе. |
| *void* ***autoScaleY****()* | однократно применяет автомасштаб по оси Y. |
| *long* ***PutChnText****(BSTR text, long index)* | разметка графика. |
| *long* ***ShowPropSheet*** *(void)* | показывает страницу свойств. |
| *long* ***InWindPtr*** *(signed char * ptr)* | получает данные в самостоятельную структуру Struct_gridGL настройки. |
| *long* ***OutWindPtr*** *(signed char * ptr)* | выдает данные в самостоятельную структуру Struct_gridGL настройки. |
| *void* ***GetMathX_cursor****(double * pCursorX)* | получает значение курсора (математическое). |
| *long* ***AltDateStartByFirstTime****(double firstTime, double intervalTimeInSec)* | инициализация массива времён, отображаемого по оси X, по первой точке. |
| *long* ***GetSelectedGraph****()* | получает индекс выделенного графика. |
| *void* ***SetMathX_cursor*** (*double* pCursorX) | задание значения курсора (математическое). |

### Отрисовка данных GridGL/

| | |
|---|---|
| *short* ***Display*** *(void)* | метод перерисовывает графики. |
| *void* ***DrawPicture****(unsigned long hdc, long x, long y, long width, long height)* | отрисовывает картинку графика на указанном HDC. |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *Void* ***DrawRift****(double riftMathStart, double riftMathStop, LONG riftMinWidthPx)* | нарисовать просвет. Рисование выполняется по математическим отсчетам и еще добавляется параметр минимальной ширины. |
| *long* ***GetPointerToDoubleData****()* | Получение указателя на отображаемые данные текущего графика, которые хранятся в виде double. |
| long **ClearGraphs** (*long* value) | Очистка графики на гриде. Перерисовывает каждый график (если графиков два [***SetNumber(2)***] - перерисует оба) массивом из 0, если *value* = 0, или массивом из NAN, если *value* любое число кроме 0. |
| *long* **ClearGraph** (*long value*) | Очистить данный график на гриде. Перерисовывает график массивом из 0, если *value* = 0, или массивом из NAN, если *value* любое число кроме 0. |
| *void* ***GetMathX_lr***(*double** pLeftX, *double** pRigthX) | Получение границ отображения по оси X. |
| *long* ***SetMathX_lr***(*double* leftX, *double* rigthX) | Задание границ отображения по оси X. |
| *long* ***GetIndexX_lr****(long* pIndexLeft, long* pIndexRight****)*** | получение индексов крайних отображаемых точек. |
| *long* ***SetIndexX_lr****(long indexLeft, long indexRight****)*** | задание индексов крайних отображаемых точек. |
| *long* ***GetIndexX_lcr****(long* pIndexLeft, long* pIndexCusor, long* pIndexRight)* | получение индексов крайних отображаемых точек и курсора. |
| *long* ***SetIndexX_lcr****(long indexLeft, long indexCusor, long indexRight****)*** | задание индексов крайних отображаемых точек и курсора. |
| **Передача данных в GridGL** | |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *long* ***Paint****(float * buffer)* | позволяет передать данные для отображения графика и перерисовать график. |
| *long* ***FormatX****(LPCTSTR fmt)* | передает в компонент строку для форматного вывода информации по координате X в легенде графика. |
| *long* ***FormatY****(LPCTSTR fmt)* | передает в компонент строку для форматного вывода информации по координате Y в легенде графика. |
| *void* ***Formatxy****(LPCTSTR fmt)* | содержит надпись в графике. |
| *void* ***AltC****(float * buffer)* | позволяет загружает массив разметки данных по X. Этот метод работает при установленном свойстве типа сетки TypeAbs = 2. |
| *void* ***AltDate*** *(DATE * buffer)* | позволяет загрузить массив дат для разметки данных по X. |
| *long* ***PaintNet****(long pBuffer)* | позволяет передать данные для отображения графика и перерисовать график, используется при работе в .net языках. |
| *void* ***AltCNet****(long pBuffer)* | позволяет задать значения по оси X. Этот метод работает при установленном свойстве типа сетки TypeAbs = 2, используется при работе в .net языках. |
| *void* ***AltDateNet****(long pBuffer)* | позволяет загрузить массив дат для разметки данных по X, используется при работе в .net языках. |
| *long* ***getStartPoint*** *(void)* | возвращает индекс первой видимой точки. Методы, возвращающие индексы первого и последнего значений графика, используемых в логарифмических данных. |
| *long* ***getEndPoint*** *(void)* | возвращает индекс последней видимой точки. Методы, возвращающие индексы первого и последнего значений графика, используемых в логарифмических данных. |
| *DOUBLE* ***GetStep*** *(void)* | определяет число пикселей на отсчет. |
| *long* ***VisibleWidthPx*** *(void)* | получает ширину сетки в пикселях. Добавление метода получения ширины таблицы. |
| **Сохранение данных в файл или в Clipboard, загрузка выбранного графика, разметки, курсоры** | |
| *void* ***PushDataToClipBoard*** *(void)* | позволяет сохранить изображение графика в Clipboard в графическом (bmp) формате. |
| *void* ***PushToClipBoard****(void* | позволяет загрузить график в ClipBoard в графическом (bmp) формате. |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *)* | |
| *long* ***SaveGLDataToFile*** *(BSTR FileName)* | Сохранить изображение можно с расширением *.dtu; *.dtx ; *.png; *.xls; *.csv".<br>В файл записываются разметка по оси X и значения всех графиков в текстовом виде. Это позволяет использовать данные из этого файла для других программ обработки. По-умолчанию файл открывается программой "Просмотр результатов". Это используется для "*.Dtu" и "*.Dtx".<br><br>"*.xls". Файл записывается в формате EXCEL. В таблицу EXCEL записывается заголовок графика, разметка по оси X и данные графиков. Наличие установленных программы Microsoft Office Excel или подобных для сохранения файла не требуется. |
| *long* ***SaveGridToFile*** *(BSTR FileName)* | позволяет сохранить график в файл BMP. Это позволяет использовать данные из этого файла для других программ обработки. |
| *long* ***PaintDouble*** *(DOUBLE* buffer);* | позволяет загрузить выбранный график (тип double). |
| *long* ***SaveGLDataToFileLimit****(BSTR FileName, INT startIndex, INT endIndex)* | позволяет сохранить изображение в текстовый файл с расширением *.dtx. |
| *long* ***GetGridGLBitmapInfo*** *(LONG* pInfo)* | позволяет получить данные о графике в виде структуры BITMAPINFO. |
| long ***CopyGridGLBitmapData****(BYTE* pData, LONG size)* | позволяет получить изображение графика в виде указателя на структуру BITMAP. |
| long ***SaveGridGLToMetaFile****(BSTR FileName)* | позволяет сохранить сетку как метафайл с расширением WMF или EMF. |
| *BSTR* ***GetFormatSaveGrid()*** | позволяет получить формат для сохранения графика. |
| *long* ***SetCursorSetting****(LONG index, LONG shape, LONG viewLine, LONG visible, LONG indexGraph, ULONG color)* | позволяет создавать вспомогательные курсоры:<br>index - индекс курсора \<br>shape - форма курсора \<br>viewLine - отображение линии до курсора \<br>visible - отображение всего курсора \ |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| | indexGraph - индекс графика, к которому относится курсор \<br>color - цвет курсора". |
| *long* ***SetPositionCursor****(LONG index, LONG positionX, DOUBLE positionY)* | позволяет изменять положения вспомогательных курсоров:<br>index - индекс курсора \<br>positionX - позиция курсора по оси X (значение от 0 до размера, переданного в grid) \<br>positionY - позиция курсора по оси Y (не в пикселях)". |
| *void ClearCollectionCursors( )* | позволяет сделать невидимыми все курсоры. |
| *long Paint_s(FLOAT * buffer, LONG size)* | позволяет загрузить выбранный график с учетом размера (тип double). |
| *long PaintDouble_s(DOUBLE * buffer, LONG size)* | позволяет загрузить выбранный график с учетом размера (тип float). |
| *void* ***AItCDouble***(DOUBLE* buffer) | позволяет загрузить массив разметки данных по X. |
| void ***UsingAddCursors(LONG flag)*** | позволяет использовать курсоры. |
| void ***GetMathX_Icr***(DOUBLE* pLeftX, DOUBLE* pCursorX, DOUBLE* pRigthX) | позволяет получить границы отображения по оси X и положения курсора. |
| long ***SetMathX_Icr***(DOUBLE leftX, DOUBLE cursorX, DOUBLE rigthX) | позволяет задать границ отображения по оси X и положения курсора. |
| long ***LoadGLDataFromFile***( BSTR FileName ) | позволяет загрузить структуры графика из текстового файла с расширением *.dtx. |
| long ***AItDateStart***(DOUBLE lastTime, DOUBLE intervalTimeInSec) | позволяет инициализировать массив времён, отображаемого по оси X, по последней точке. |
| long ***AItDateStart****()* | позволяет сдвигать массивы времён, отображаемого по оси X, на одну точку. |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| long ***AItDateNewTimeInterval***( LONG sizeInterval) | позволяет сдвигать массива времён, отображаемого по оси X, на несколько точек. |
| long ***AItDateStartByFirstTime***( DOUBLE firstTime, DOUBLE intervalTimeInSec) | позволяет инициализировать массив времён, отображаемого по оси X, по первой точке. |
| **Параметры меток в GridGL** | |
| *VARIANT_BOOL* ***AddToolTip****(BSTR text, LONG pos)* | состояние включенности/выключенности метки. |
| *VARIANT_BOOL* ***RemoveAllToolTips****(void)* | состояние изменения включенности/выключенности всех меток. |
| *long* ***AddMark****(DOUBLE x, BSTR text)* | добавить текстовую метку на график. |
| *long* ***EditMark****(LONG index, BSTR newText)* | редактировать текст метки на графике. |
| *LONG* ***GetMark****(LONG index, DOUBLE* x, BSTR* text)* | получить информацию о метке на графике. |
| *long* ***MoveMark****(LONG index, DOUBLE newX)* | передвинуть метку на новую позицию. |
| *long* ***RemoveMark****(LONG index)* | удалить метку. |
| **Параметры полос в GridGL** | |
| *long* ***SetColorLineCount****(LONG Count)* | устанавливает количество полос с отображением событий. |
| *long* ***SetColorLineFont****(LONG Number, OLE_COLOR _Color)* | устанавливает общий фон для полосы с номером Number. |
| *long* ***SetColorLineName****(LONG Number, BSTR Name)* | устанавливает текст для полосы с номером Number. |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *long* ***SetColorLineSpans****(LONG Number, LONG* IndexBeginArray, LONG* IndexEndArray, ULONG* ColorsArray, LONG ArraySize);* | устанавливает временные интервалы и цвета для интервалов на которые будет разбита полоса: \<br><br>Number - номер полосы \<br><br>IndexBeginArray - массив начальных индексов с которых будет начинаться разбиение для каждого интервала \<br><br>IndexEndArray = массив конечных индексов до которых будет идти каждый интервал \<br><br>ColorsArray - массив цветов для каждого интервала, подается !!!OLE_COLOR *!!! для каждого интервала \<br><br>ArraySize общее количество интервалов. |
| *long* ***SetColorLineSpansByTime****(LONG Number, DATE* DateBeginArray, DATE* DateEndArray, ULONG* ColorsArray, LONG ArraySize)* | устанавливает временные интервалы и цвета для интервалов на которые будет разбита полоса: \<br><br>Number - номер полосы \<br><br>DateBeginArray - массив начальных дат с которых будет начинаться разбиение для каждого интервала \<br><br>DateEndArray = массив конечных дат до которых будет идти каждый интервал \<br><br>ColorsArray - массив цветов для каждого интервала, подается !!!OLE_COLOR *!!! для каждого интервала \<br><br>ArraySize общее количество интервалов. |
| **Индикационная рамка в GridGL.ocx** | |
| void ***ShowBorderIndication*** ( LONG isShow ) | устанавливает отображение индикационной рамки. |
| void ***SetParametresBorderIndication***( OLE_COLOR color, LONG alpha, LONG width ) | – устанавливает параметры индикационной рамки:<br>color - цвет;<br>alpha - прозрачность;<br>width - ширина. |
| **События от компоненты GridGl.ocx** | |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *void* ***DblClick*** *()* | событие возникает при двойном нажатии на кнопку манипулятора «мышь», при нахождении мыши над графиком. |
| *void* ***KeyDown****(SHORT* *KeyCode, SHORT* *Shift)* | событие возникает при нажатии на клавишу клавиатуры, когда фокус находится на компоненте. |
| *void* ***MouseDown****(short* *Button, short Shift,* *OLE_XPOS_PIXELS x,* *OLE_YPOS_PIXELS y)* | событие возникает при нажатии на кнопку манипулятора типа «мышь», при нахождении мыши над графиком. |
| *void* ***MouseUp****(short* *Button, short Shift,* *OLE_XPOS_PIXELS x,* *OLE_YPOS_PIXELS y)* | событие возникает при отжатии кнопки манипулятора типа «мышь», при нахождении мыши над графиком. |
| *void* ***MouseWheel****(void)* | событие возникает при вращении ролика манипулятора типа «мышь». |
| *void* ***Modify*** *(void)* | событие, которое происходит при включении и отключении каналов АЦП, ЦАП, виртуальных каналов, изменении коэффициентов усиления и т.д. |
| *void* ***ChangeSetting****(void)* | событие, изменение настроек gridGL. |
| void ***OnAutoScale****()* | событие, которое возникает при клике мышью по зоне AutoScale. |
| void ***OnChangeRect***(LONG changeX, LONG changeY); | событие, которое возникает при изменении прямоугольника с сеткой. |

# Глава 3.PloterXY.ocx графический компонент отображения зависимостей X(t)-Y(t)-Z(t) в 2- и 3-мерном виде

**PlotterXYZ.ocx** - графический компонент отображения зависимостей X(t)-Y(t)-Z(t) в 2- и 3-мерном виде. Компонент PlotterXYZ предназначен для визуализации массивов данных в двухмерном и трехмерном виде. Процедура рисования графиков написана таким образом, что отсутствует мерцание при перерисовке графиков

Имеются три последовательности, зависящие от времени: x=X(t), y=Y(t) и z=Z(t), T<t<T+dT. Для этих последовательностей компонент PlotterXYZ позволяет построить следующие графики:

- временную реализацию *(x,t)*, *(y,t)* или *(z,t)* на плоскостях XT, YT или ZT ;

© ООО "ЭТМС" 1992-2024. Все права защищены

- параметрическую кривую (x,y), (x,z) или (y,z) на плоскостях XY, XZ или YZ;
- параметрическую кривую (x,y,t), (x,z,t) или (y,z,t) в трехмерном виде в пространствах XYT, XZT или YZT;
- параметрическую кривую (x, y, z) в трехмерном виде в пространстве XYZ.

## 3.1.Установка компонента PloterXY.ocx

Для работы с модулем PloterXY его необходимо сначала установить в программу. При загрузке программы необходимо загрузить с помощью компонента необходимую программу, разместить окно программы на экране в удобном месте, установить необходимые параметры в программе и затем по событиям, происходящим от программы считывать данные из программы.

Для установки компонента PloterXY в проект *Microsoft Visual Studio 2010*, необходимо кликнуть правой кнопкой мыши на форме диалога и из появившегося меню выбрить пункт *Insert ActiveX Control*...(рисунок 1.1)

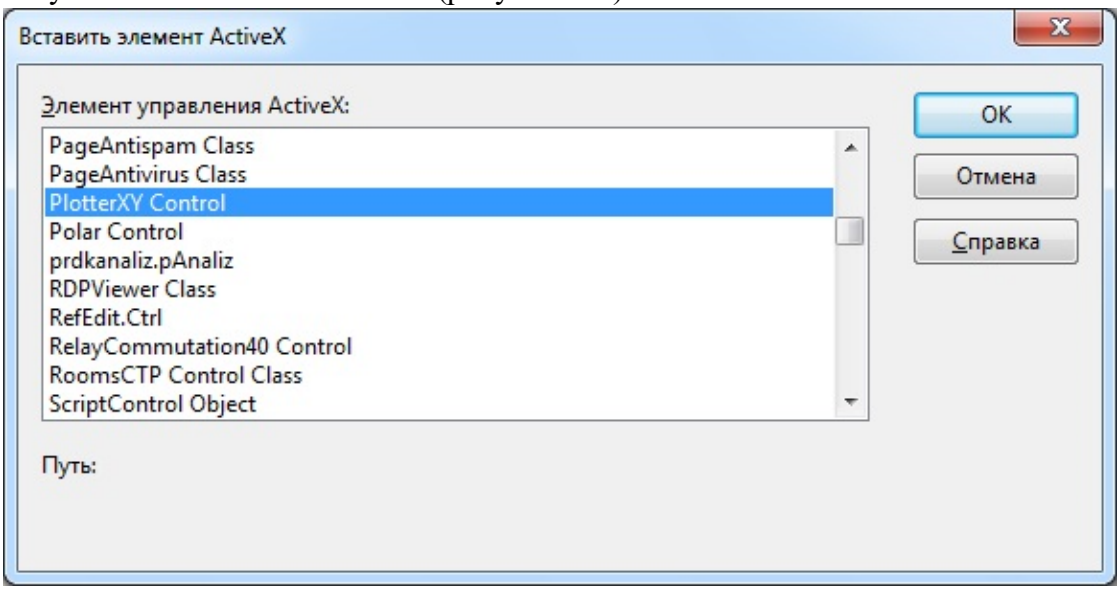



Рисунок 1.1

Из появившегося диалогового окна следует выбрать компонент *PloterXY Control* и нажать *OK* (рисунок 1.2). После этого компонент PloterXY.*ocx* появится на форме диалога (рисунок 1.2).

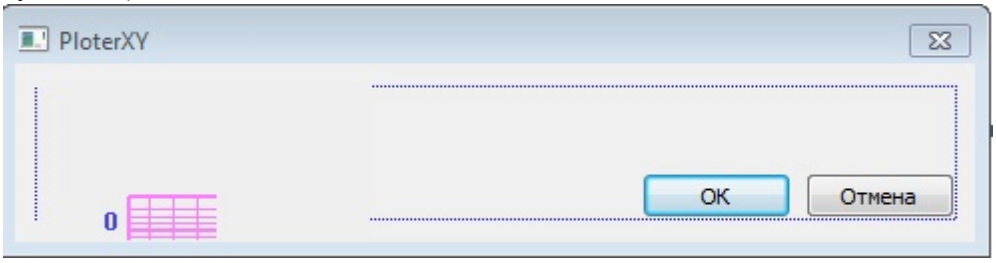



Рисунок 1.2

© ООО "ЭТМС" 1992-2024. Все права защищены

Одна установленная на форме компонента позволяет управлять одной программой. При необходимости управлять несколькими программами, необходимо установить на форму несколько компонент. Для того чтобы компонента не была видна на форме во время выполнения программы, необходимо установить свойство *Visible* равным *False*.

## 3.2.Назначение PloterXY.ocx

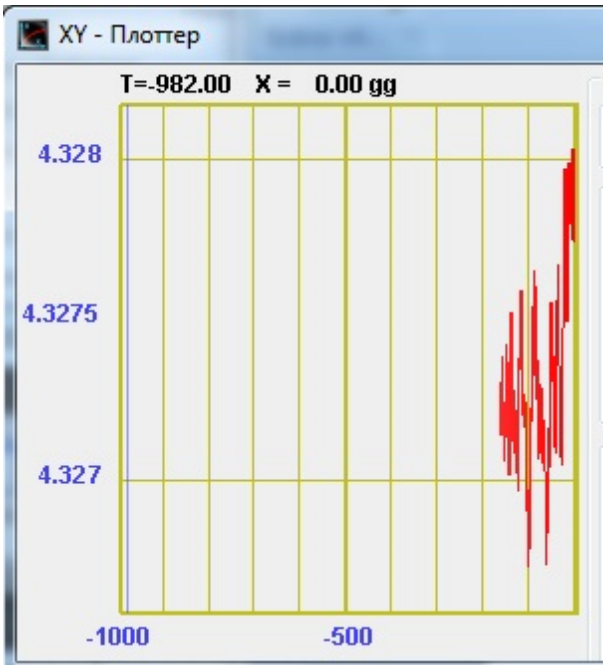
Компонент **PlotterXYZ** используется в виртуальных приборах на базе ПК, таких, как **Осциллограф**, **Плоттер**, в программах спектрального анализа сигнала для отображения диаграммы Найквиста.

Компонента позволяет

- масштабировать график по всем координатам;
- вращать график вокруг любой из осей в трехмерном виде;
- плавно менять цвет графика от начальной точки к конечной;
- перемещать курсор при помощи кнопки мыши, ролика мыши и клавиатуры;
- значения абсцисс и ординат на позиции курсора при отображении графиков в двухмерном виде;
- сохранять данные в буфере обмена (clipboard) в графическом, текстовом и численном виде для создания отчетов в редакторах Microsoft Word и Excel.

**PlotterXYZ** также является компонентом SCADA системы ZETView

© ООО "ЭТМС" 1992-2024. Все права защищены




## 3.3.Полное описание свойств, методов и событий PloterXY.ocx

Следует отметить, что компонент автомасштабируется при изменении его размеров, поэтому необходимо следить за тем, чтобы высота компонента в приложении была не менее 80 пикселей, иначе он может отображаться некорректно.

### Свойства PloterXY.ocx

Свойства можно задавать на этапе проектирования и во время выполнения программы. На этапе проектирования при выборе компонента и при нажатии клавиши <F4> появляется окно свойств. Эти свойства можно устанавливать на этапе проектирования программы.

Цвета задаются при помощи макроса *RGB(RED, GREEN, BLUE)*. Значения *RED*, *GREEN*, *BLUE* задают интенсивности цветов красного, зеленого и синего. Интенсивности изменяются в пределах от 0 до 255.

*OLE_COLOR* ***ClrCur*** – цвет курсора.

*OLE_COLOR* ***ClrFon*** – цвет фона.

*OLE_COLOR* ***ClrGrd*** – цвет сетки.

© ООО "ЭТМС" 1992-2024. Все права защищены

*OLE_COLOR* ***ClrGrfBeg*** – цвет начального графика. Для каждого графика цвет задается отдельно. Графики выбираются при помощи свойства *Ind*.

*OLE_COLOR* ***ClrGrfEnd*** – цвет последнего графика. Для каждого графика цвет задается отдельно. Графики выбираются при помощи свойства *Ind*.

*OLE_COLOR* ***ClrNum*** – цвет цифр по осям.

*OLE_COLOR* ***ClrLeg*** – цвет легенды

Например, для изменении фона графика с желтого на зеленый, могут быть записаны числа:

*OLE_COLOR a = 0xFF00;*
*OLE_COLOR b = 0xFFFF;*

*long* ***NumPoints*** *(long propVal)* – максимальное количество точек на графике, В случае непрерывного графика – неучитывается.

*BSTR* ***FontName*** - название шрифта, например, Arial.

*long* ***FontSize*** - размер шрифта (пикселей).

*long* ***SquareScale*** - автомасштаб по осям X и Y (одновременно).
Параметры.

0 – выкл.

1 – вкл.

*VARIANT_BOOL* ***AutoMX*** – автомасштаб по времени
Параметры.

0 – выкл.

1 – вкл.

*VARIANT_BOOL* ***LeftRight*** - направление "сползания" графика.

Параметры

0(false) – влево.

1(true) – вправо

*long* ***CurNPoints*** - количество точек графика

*long* ***CommentIND*** - порядковый номер комментария.

*BSTR* ***CommentSTR*** – строки комментария.

© ООО "ЭТМС" 1992-2024. Все права защищены

*FLOAT* MathLX – текущее минимальное значение по X. Левая граница отображения данных.

FLOAT MathDX – Текущий диапазон отображения по X. Длина отображения данных.

Например, необходимо отобразить часть данных из предыдущего примера в диапазоне от 1050 Гц до 1070 Гц. В этом случае задаются следующие параметры:

*MathLX* = 1050.

*MathDX* = 20.

При работе программы в режиме исполнения пользователь может менять область отображения графиков при помощи курсора мыши. Растяжение или сжатие графиков происходит при помощи указателя вида:  ,  – для горизонтальной оси. Сдвинуть графики вправо-влево можно при помощи указателя  ,  – для горизонтальной оси. При этом меняются параметры *MathLX, MathDX.* В пользовательской программе в режиме исполнения можно прочитать текущие параметры области отображения.

*FLOAT* ***MathLY*** – текущее минимальное значение по Y. Нижняя граница отображения данных.

*FLOAT* ***MathDY*** – текущий диапазон отображения по Y. Высота отображения данных.

Растяжение или сжатие графиков происходит при помощи указателя вида:  ,  – для вертикальной оси. Сдвинуть графики вверх или вниз можно при помощи указателя  ,  – для вертикальной оси. При этом меняются параметры *MathLY, MathDY.* В пользовательской программе в режиме исполнения можно прочитать текущие параметры области отображения.

*FLOAT* ***MathLZ*** – текущее минимальное значение по времени. Нижняя граница по времени.

*FLOAT* ***MathDZ*** – текущий размер по времени. Высота данных по времени.

Растяжение или сжатие графиков происходит при помощи указателя вида:  ,  – для вертикальной оси. Сдвинуть графики вверх или вниз можно при помощи

© ООО "ЭТМС" 1992-2024. Все права защищены

указателя ↑ , ↓ – для вертикальной оси. При этом меняются параметры *MathLZ, MathDZ.* В пользовательской программе в режиме исполнения можно прочитать текущие параметры области отображения. При включенных ограничениях отображения графиков *MakeUpor* = 1 , параметр *MathDY* не может быть меньше *UporNis ,* сумма параметров *MathLZ* + *MathDZ* не может быть больше *UporVerh*.

*FLOAT* ***UporXNis*** – минимальная величина по разрешению по координате *X*, которая может отобразиться на графике.

*FLOAT* ***UporXVerh*** – максимальная величина по координате X, которая может отображаться на графике.

*FLOAT* ***UporYNis*** – минимальная величина по координате *Y*, которая может отобразиться на графике.

*FLOAT* ***UporYVerh*** – максимальная величина по координате Y, которая может отображаться на графике.

*FLOAT* ***UporZNis*** – минимальная величина по времени, которая может отобразиться на графике. Это свойство работает при включенных ограничениях по времени.

*FLOAT* ***UporZVerh*** – максимальная величина по времени, которая может отображаться на графике.

*FLOAT* ***DeltaT*** – размер математического деления по времени.

*FLOAT* ***UporDelta*** - интервала между строками в единицах измерения.

*VARIANT_BOOL* ***FixedTime*** - признак ограничения времени.

Параметры:

0(false) - время неограниченно, количество точек и пределы по времени изменяются автоматически;

1(true) - время ограниченно, количество точек и пределы задаются вручную.

*VARIANT_BOOL* ***ShowFPS*** - показать отображение FPS.

Параметры.

0(false) – не показывать.

n1(true) – показывать.

*FLOAT* ***MathLZSignal*** - текущее минимальное значение сигнала по оси Z. Нижняя граница по оси Z .

© ООО "ЭТМС" 1992-2024. Все права защищены

*FLOAT* ***MathDZSignal*** – текущий размер сигнала по оси Z. Высота данных по оси Z

*FLOAT* ***UporZSignalNis*** – минимальная величина сигнала по оси Z, которая может отобразиться на графике. Нижняя граница по оси Z

*FLOAT* ***UporZSignalVerh*** – максимальная величина сигнала по оси Z, которая может отображаться на графике. Верхняя граница по оси Z.

*long* ***TypeCoord*** – тип координат сетки.

Параметры:
1 – плоскость XT
2 – плоскость YT
4 – плоскость XY
8 – 3-х мерный вид XYT
16 -3-х мерный вид XYZ

*long* ***TypeLine*** – тип линий сетки.

1 – рисует горизонтальные линии.
2 – не рисует линии разметки.
4 – рисует вертикальные линии.
5 – (1+4) - рисует всю сетку.

BSTR TextXAxis - Подпись для оси X.

BSTR TextYAxis - Подпись для оси Y.

BSTR TextTAxis - Подпись для оси T.

BSTR TextZAxis - Подпись для оси Z.

## Методы PloterXY.ocx

Методы можно использовать только во время выполнения программы.

© ООО "ЭТМС" 1992-2024. Все права защищены

*long* ***SaveDateToFile*** *(BSTR FileName)* - сохранение данных в текстовый файл с расширением *.dtu. В файл записываются накопленные данные за установленный интервал.

*long* ***SaveGridToFile****(BSTR FileName)* - сохранение данных в формате хранения растровых изображений с расширением *.BMP. В файл записываются накопленные данные за установленный интервал.

*long* ***LoadDataFromFile****(BSTR FileName, LONG pXArray, LONG pYArray, VARIANT_BOOL DrawData)* – чтение данных из файла с расширением *.dtu

*long* ***LoadDataFromFileNet****(BSTR FileName, LONG pXArray, LONG pYArray, VARIANT_BOOL DrawData)* – чтение данных из файла с расширением *.dtu , используется при работе в .net языках

*void* ***ChangeProperties*** *(void)* – настройка свойства отображения.

*long* ***LoadComments(****LPCTSTR FileName)* –чтение комментария из файла.

*void* ***PrintScreen****(void)* – позволяет сохранить ***PrintScreen.***

*void* ***PutDataToClipboard*** *(void)* - копировать данные графика в буфер обмена.

*long* ***AutoScale*** *(void)* – однократно применяет автомасштаб по оси Y.

*long* ***ResetPlotter*** *(void)* – обнуляет массивы с координатами.

*long* ***FormatX*** (*BSTR* format) – метод передает в компонент строку для форматного вывода информации по координате X в легенде графика. Форматный ввод-вывод позволяет передавать программе и выдавать из программы на внешнее устройство символьные, строковые и числовые значения, обеспечивая преобразование данных в процессе ввода-вывода.

Например, обращение вида:

*FormatX(“Частота %7.2f Гц”)*

позволяет отображать координаты по оси X в значениях по частоте.

Формат представления чисел.

*%f* – общий формат представления числа в плавающей запятой в виде числа с точкой

*%e* – общий формат представления числа в плавающей запятой в виде числа с экспонентой (*1000 = 1e3, 0.001 = 1e-3*)

© ООО "ЭТМС" 1992-2024. Все права защищены

*%7.2f* – представление числа в фиксированном формате. В данном примере *7* – количество знакомест для представления числа, *2* – количество цифр после запятой в представлении числа.

*long* ***FormatY*** *(BSTR format)* – метод передает в компонент строку для форматного вывода информации по координате *Y* в легенде графика. Форматный ввод-вывод позволяет передавать программе и выдавать из программы на внешнее устройство символьные, строковые и числовые значения, обеспечивая преобразование данных в процессе ввода-вывода. Метод должен выполняться для каждого графика. Графики выбираются при помощи свойства *Ind*.

Например, обращение вида:

*FormatY ("Амплитуда %9.4f мВ")*

позволяет отображать координаты по оси *Y* в значениях по амплитуде.

*long* ***FormatT*** *(BSTR format)* - позволяет отображать координаты по оси T с временным интервалом..

*long* ***FormatZ*** *(BSTR format)* метод передает в компонент строку для форматного вывода информации по координате *Z* в легенде графика. Форматный ввод-вывод позволяет передавать программе и выдавать из программы на внешнее устройство символьные, строковые и числовые значения, обеспечивая преобразование данных в процессе ввода-вывода. Метод должен выполняться для каждого графика. Графики выбираются при помощи свойства *Ind*.

Например, обращение вида:

*FormatZ ("Амплитуда %9.4f мВ")*

позволяет отображать координаты по оси *Z* в значениях по амплитуде.

Методы можно использовать только во время выполнения программы.

*long* ***PutNewPoint*** *(float x, float y)* – добавить новую точку в двумерном пространстве X и Y на графике.

*long P****utNewPointXYZ****(float x, float y, float z)* - добавить новую точку в трехмерном пространстве X, Y, Z на графике.

*long* ***PutXPoints*** *(float * Xpoints)* – записать массив координат по X, где размер массива должен быть не меньше NumPoints.

*long* ***PutYPoints*** *(float * Ypoints)* – записать массив координат по Y, где размер массива должен быть не меньше NumPoints.

*long* ***PutXYPoints****(float * Xpoints, float * Ypoints)* - записать массив координат по X и Y на графике, где размер массива должен быть не меньше NumPoints.

© ООО "ЭТМС" 1992-2024. Все права защищены

*long* ***PutZPoints****(float * Zpoints)* - записать массив координат по Z на графике, где размер массива должен быть не меньше NumPoints.

*long* ***PutXYZPoints****(float * Xpoints, float * Ypoints, float * Zpoints)* - записать массив координат по X, Y и Z на графике, где размер массива должен быть не меньше NumPoints.

*long* ***PutXPointsNet****(long pXpoints)* - записать массив координат по X на графике (для языков программирования .NET), где размер массива должен быть не меньше NumPoints.

*long* ***PutYPointsNet****(long pYpoints)* - записать массив координат по Y на графике (для языков программирования .NET), где размер массива должен быть не меньше NumPoints.

*long* ***PutZPointsNet****(long pZpoints)* - записать массив координат по Z на графике (для языков программирования .NET) , где размер массива должен быть не меньше NumPoints.

*long* ***PutXYPointsNet****(long pXpoints, long pYpoints)* - записать массив координат по X и Y на графике (для языков программирования .NET) , где размер массива должен быть не меньше NumPoints.

*long* ***PutXYZPointsNet****(long pXpoints, long pYpoints, long pZpoints)* - записать массив координат по X, Y и Z на графике (для языков программирования .NET) , где размер массива должен быть не меньше NumPoints.

*long* ***SetCursorPositionByIndex****(LONG index)* – Установить курсор в позицию точки с индексом Index

Если calcForT == true, позиция индекса в любых осях вычисляется от значения T

Возвращает 0 в случае ошибки, в остальных случаях 1

*long* ***GetIndexForCursorPosition*****()** – получить значение индекса для позиции курсора

Возвращает -1 в случае ошибки, в остальных случаях значение индекса

## 3.4.События от компоненты PloterXY.ocx

*void* ***KeyDown*** *(SHORT* KeyCode, SHORT Shift)* – событие возникает при нажатии на клавишу клавиатуры, когда фокус находится на компоненте.

Параметры:
*KeyCode* – код клавиши

© ООО "ЭТМС" 1992-2024. Все права защищены

*Shift* – код нажатия служебных клавиш <*Ctrl*>,<*Shift*>,<*Alt*>.

**Примечание управляющим клавишам:**

<C> - *Копирование графика.* Для копирования графика нажать левой кнопкой «мыши» на поле графика и нажать комбинацию «горячих клавиш» клавиатуры <Ctrl> +<C> .График запишется в буфер обмена(Clipboard).

<N> *Копирование численных значений.* Для копирования численных значений нажать левой кнопкой «мыши» на поле графика программы и нажать на кнопку клавиатуры <N>(в латинской раскладке клавиатуры). Численные значения скопируются в буфер обмена(Clipboard) (устаревшее).

*void* ***MouseDown*** *(short Button, short Shift, OLE_XPOS_PIXELS x, OLE_YPOS_PIXELS y)* – событие возникает при нажатии на кнопку манипулятора типа «мышь», при нахождении мыши над графиком.

Параметры:
*Button* – код кнопки мыши
*Shift* – код нажатия служебных клавиш <*Ctrl*>,<*Shift*>,<*Alt*>.
*x* – координата по оси *X* в пикселах (разрешении экрана)
*y* – координата по оси *Y* в пикселах (разрешении экрана)

*void* ***MouseUp*** *(short Button, short Shift, OLE_XPOS_PIXELS x, OLE_YPOS_PIXELS y)* – событие возникает при отжатии кнопки манипулятора типа «мышь», при нахождении мыши над графиком.

Параметры:
*Button* – код кнопки мыши
*Shift* – код нажатия служебных клавиш <*Ctrl*>,<*Shift*>,<*Alt*>.
*x* – координата по оси *X* в пикселях (разрешении экрана)
*y* – координата по оси *Y* в пикселях (разрешении экрана)

v*oid* ***MouseWheel*** *(void)* – событие возникает при вращении ролика манипулятора типа «мышь».

*void* ***Modify*** *(void)* - событие, которое происходит при включении и отключении каналов АЦП, ЦАП, виртуальных каналов, изменении коэффициентов усиления и т.д.

**Примечание:**

*Копирование графика*

Для копирования графика нажать левой кнопкой «мыши» на поле графика и нажать комбинацию «горячих клавиш» клавиатуры <Ctrl> + <C>. График запишется в буфер обмена (Clipboard). Вставить график в любой открытый документ Microsoft Word или Excel можно нажатием «горячих клавиш» клавиатуры <Ctrl > + <V>, или нажатием на правую кнопку «мыши» и выбором в появившемся меню команды Вставить.

© ООО "ЭТМС" 1992-2024. Все права защищены

*Копирование численных значений*

Для копирования численных значений видимой части графика нажать левой кнопкой «мыши» на поле графика и нажать на кнопку клавиатуры <N> (в латинской раскладке клавиатуры). Вставить эту информацию в любой открытый текстовый документ можно нажатием «горячих клавиш» клавиатуры <Ctrl> + <V>, или нажатием на правую кнопку «мыши» и выбором в появившемся меню команды Вставить.

## 3.5.Краткий перечень функций, используемые PloterXY.ocx

| **Свойства PlotterXY.ocx** | |
|---|---|
| **Цвета** | |
| *long* ***GetClrCur****()* | опрашивается цвет курсора |
| *void* ***SetClrCur*** *(unsigned long propVal)* | устанавливается цвет курсора |
| *long* ***GetClrFon*** *()* | опрашивается цвет фона |
| *void* ***SetClrFon*** *(unsigned long propVal)* | устанавливается цвет фона |
| *long* ***GetClrGrfBeg*** *()* | опрашивается цвет начала графика |
| *void* ***SetClrGrfBeg*** *(unsigned long propVal)* | устанавливается цвет начала графика |
| *long* ***GetClrGrfEnd****()* | опрашивается цвет конца графика |
| *void* ***SetClrGrfEnd*** *(unsigned long propVal)* | устанавливается цвет конца графика |
| *long* ***GetClrGrd*** *()* | опрашивается цвет сетки |
| *void* ***SetClrGrd*** *(unsigned long propVal)* | устанавливается цвет сетки |
| *long* ***GetClrLeg*** *()* | опрашивается цвет легенды графика |
| *void* ***SetClrLeg*** *(unsigned long propVal)* | устанавливается цвет легенды графика |
| *long* ***GetClrNum*** *()* | опрашивается цвет цифр по осям |
| *void* ***SetClrNum*** *(unsigned long* | устанавливается цвет цифр по осям |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *propVal)* | |
| **Параметры графиков** | |
| *long* ***GetNumber****()* | опрашивается количество отображаемых графиков. |
| *void* ***SetNumber****(long propVal)* | устанавливается количество отображаемых графиков. |
| *CString* ***GetFontName****()* | опрашивается название шрифта, например, Arial |
| *void* ***SetFontName****(CString propVal)* | устанавливается название шрифта, например, Arial |
| *long* ***GetSize*** *()* | опрашивается размер шрифта (пикселей) |
| *void* ***SetSize*** *(long propVal)* | устанавливается размер шрифта (пикселей) |
| *long* ***GetSquareScale*** *()* | опрашивается автомасштаб по осям X и Y (одновременно , где 0 – выкл.,1 – вкл.) |
| *void* ***SetSquareScale*** *(long propVal)* | устанавливается автомасштаб по осям X и Y (одновременно , где 0 – выкл.,1 – вкл.) |
| *BOOL* ***GetAutoMX****()* | опрашивается автомасштаб по времени (0 – выкл.,1 – вкл.) |
| *void* ***SetAutoMX****(BOOL propVal)* | устанавливается автомасштаб по времени (0 – выкл.,1 – вкл.) |
| *BOOL* ***GetLeftRight*** *()* | опрашивается направление "сползания" графика (0(false) – влево , 1(true) – вправо) |
| *void* ***SetLeftRight*** *(BOOL propVal)* | Устанавливается направление "сползания" графика (0(false) – влево , 1(true) – вправо) |
| *long* ***GetCurNPoints*** *()* | опрашивается количество точек графика |
| *void* ***SetCurNPoints*** *(long propVal)* | устанавливается количество точек графика |
| *long* ***GetCommentIND*** *()* | опрашивается порядковый номер комментария |
| *void* ***SetCommentIND*** *(long propVal)* | устанавливается порядковый номер комментария |
| *CString* ***GetCommentSTR****()* | опрашивается строки комментария |
| *void* ***SetCommentSTR****(CString propVal)* | устанавливается строки комментария |
| **Представление графиков** | |
| *double* ***GetMathlx****()* | опрашивается минимальное значение по X. Левая граница отображения данных |
| *void* ***SetMathlx****(double propVal)* | устанавливается минимальное значение по X. Левая граница отображения данных |
| *double* ***GetMathdx****()* | опрашивается диапазон отображения по X. Длина отображения данных |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *void* ***SetMathdx****(double propVal)* | устанавливается диапазон отображения по X. Длина отображения данных |
| *double* ***GetMathly****()* | опрашивается минимальное значение по Y. Нижняя граница отображения данных |
| *void* ***SetMathly****(double propVal)* | Устанавливается минимальное значение по Y. Нижняя граница отображения данных |
| *double* ***GetMathdy****()* | опрашивается диапазон отображения по Y. Высота отображения данных |
| *void* ***SetMathdy****(double propVal)* | устанавливается диапазон отображения по Y. Высота отображения данных |
| *double* ***GetMathlz****()* | опрашивается минимальное значение по времени. Нижняя граница по времени. |
| *void* ***SetMathlz****(double propVal)* | устанавливается минимальное значение по времени. Нижняя граница по времени. |
| *double* ***GetMathdz****()* | опрашивается размер по времени. Высота данных по времени |
| *void* ***SetMathdz****(double propVal)* | устанавливается размер по времени. Высота данных по времени |
| *float* ***GetUporXNis*** *()* | опрашивается минимальная величина по разрешению по координате *X*, которая может отобразиться на графике |
| *void* ***SetUporXNis****(float propVal)* | устанавливается минимальная величина по разрешению по координате *X*, которая может отобразиться на графике |
| *float* ***GetUporXVerh*** *()* | опрашивается максимальная величина по координате X, которая может отображаться на графике |
| *void* ***SetUporXVerh*** *(float propVal)* | устанавливается максимальная величина по координате X, которая может отображаться на графике |
| *float* ***GetUporYNis*** *()* | опрашивается минимальная величина по координате *Y*, которая может отобразиться на графике |
| *void* ***SetUporYNis*** *(float propVal)* | Устанавливается минимальная величина по координате *Y*, которая может отобразиться на графике |
| *float* ***GetUporYVerh*** *()* | опрашивается максимальная величина по координате Y, которая может отображаться на графике |
| *void* ***SetUporYVerh*** *(float propVal)* | устанавливается максимальная величина по координате Y, которая может отображаться на графике |
| *float* ***GetUporZNis*** *()* | опрашивается минимальная величина по времени, которая может отобразиться на графике.<br>Это свойство работает при включенных ограничениях по времени |
| *void* ***SetUporZNis*** *(float propVal)* | устанавливается минимальная величина по времени, которая может отобразиться на графике.<br>Это свойство работает при включенных ограничениях по времени. |
| *float* ***GetUporZVerh*** *()* | опрашивается максимальная величина по времени, которая может отображаться на графике |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *void **SetUporZVerh** (float propVal)* | устанавливается максимальная величина по времени, которая может отображаться на графике |
| *float **GetDeltaT** ()* | опрашивается размер математического деления по времени |
| *void **SetDeltaT** (float propVal)* | устанавливается размер математического деления по времени |
| *float **GetUporDelta** ()* | опрашивается интервал между строками в единицах измерения |
| *void **SetUporDelta** (float propVal)* | устанавливается интервал между строками в единицах измерения |
| *BOOL **GetFixedTime**()* | опрашивается признак ограничения времени (0(false) - время неограниченно, кол-во точек и пределы по времени изменяются автоматически;1(true) - время ограничено, кол-во точек и пределы задаются вручную) |
| *void **SetFixedTime**(BOOL propVal)* | устанавливается признак ограничения времени (0(false) - время неограниченно, кол-во точек и пределы по времени изменяются автоматически;1(true) - время ограничено, кол-во точек и пределы задаются вручную) |
| *BOOL **GetShowFPS** ()* | опрашивается показать отображение FPS (0(false) – не показывать. 1(true) – показывать) |
| *void **SetShowFPS** (BOOL propVal)* | устанавливается показать отображение FPS (0(false) – не показывать. 1(true) – показывать) |
| *float **GetMathLZSignal** ()* | опрашивается минимальное значение сигнала по оси Z. Нижняя граница по оси Z |
| *void **SetMathLZSignal** (float propVal)* | устанавливается минимальное значение сигнала по оси Z. Нижняя граница по оси Z |
| *float **GetMathDZSignal** ()* | опрашивается размер сигнала по оси Z. Высота данных по оси Z |
| *void **SetMathDZSignal** (float propVal)* | устанавливается размер сигнала по оси Z. Высота данных по оси Z |
| *float **GetUporZSignalNi s**()* | опрашивается минимальная величина сигнала по оси Z, которая может отобразиться на графике. Нижняя граница по оси Z |
| *void **SetUporZSignalNi s**(float propVal)* | устанавливается минимальная величина сигнала по оси Z, которая может отобразиться на графике. Нижняя граница по оси Z |
| *float **GetUporZSignalVerh** ()* | опрашивается максимальная величина сигнала по оси Z, которая может отображаться на графике. Верхняя граница по оси Z |
| *void **SetUporZSignalVerh** (float propVal)* | устанавливается максимальная величина сигнала по оси Z, которая может отображаться на графике. Верхняя граница по оси Z |
| **Типы координат, линий сетки и подписи для осей** | |
| *long **GetTypeCoord**()* | опрашивается тип координат сетки (1 - Плоскость XT, 2 - Плоскость YT, 4 - Плоскость XY, - 3-х мерный вид XYT, 3-х мерный вид XYZ) |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *void **SetTypeCoord**(long propVal)* | устанавливается тип координат сетки (1 - Плоскость XT, 2 - Плоскость YT, 4 - Плоскость XY, - 3-х мерный вид XYT, 3-х мерный вид XYZ) |
| *long **GetTypeLine**()* | опрашивается тип линий сетки графика (1-горизонтальными линиями, 2 – не рисует линии разметки, 4 – рисует вертикальные линии, 5– (1+4) - рисует всю сетку) |
| *void **SetTypeLine**(long propVal)* | устанавливается тип линий сетки графика (1-горизонтальными линиями, 2 – не рисует линии разметки, 4 – рисует вертикальные линии, 5– (1+4) - рисует всю сетку) |
| *CString **TextXAxis**()* | опрашивается подпись для оси X. |
| void ***SetTextXAxis***(CString propVal) | устанавливается подпись для оси X. |
| *CString **TextYAxis**()* | опрашивается подпись для оси Y. |
| void ***SetTextYAxis***(CString propVal) | устанавливается подпись для оси Y. |
| *CString **TextTAxis**()* | опрашивается подпись для оси T. |
| void ***SetTextTAxis***(CString propVal) | устанавливается подпись для оси T. |
| *CString **TextZAxis**()* | опрашивается подпись для оси Z. |
| void ***SetTextZAxis***(CString propVal) | устанавливается подпись для оси Z. |
| **Методы PlotterXY.ocx** | |
| **Настройки PlotterXY.ocx** | |
| *long **SaveDateToFile** (BSTR FileName)* | сохранение данных в текстовый файл с расширением *.dtu. В файл записываются накопленные данные за установленный интервал |
| *long **SaveGridToFile**(BSTR FileName)* | сохранение данных в формате хранения растровых изображений с расширением *.BMP. В файл записываются накопленные данные за установленный интервал. |
| *long **LoadDataFromFile**(BSTR FileName, LONG pXArray, LONG pYArray, VARIANT_BOOL DrawData)* | чтение данных из файла с расширением *.dtu |
| *long **LoadDataFromFileNet** (BSTR FileName, LONG* | чтение данных из файла с расширением *.dtu (для языков программирования .NET) |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *pXArray, LONG pYArray, VARIANT_BOOL DrawData)* | |
| *void **ChangeProperties** (void)* | настройка свойства отображения |
| *long **LoadComments(**LPCTSTR FileName)* | чтение комментария из файла |
| *void **PrintScreen**(void)* | позволяет сохранить ***PrintScreen*** |
| *void **PutDataToClipboard** (void)* | копировать данные графика в буфер обмена |
| *long **AutoScale** (void)* | однократно применяет автомасштаб по оси Y |
| *long **ResetPlotter** (void)* | обнуляет массивы с координатами |
| **Передача данных PlotterXY.ocx** | |
| *long **FormatX**(LPCTSTR format)* | передает строку формата по координате X в легенде графика |
| *long **FormatY**(LPCTSTR format)* | передает строку формата по координате Y в легенде графика |
| *long **FormatZ**(LPCTSTR format)* | передает строку формата по координате Z в легенде графика |
| *long **FormatT**(LPCTSTR format)* | передает строку формата по времени в легенде графика |
| **Границы видимости PlotterXY.ocx** | |
| *long **PutNewPoint** (float x, float y)* | добавить новую координату в двумерном пространстве X и Y на графике |
| *long P**utNewPointXYZ**(float x, float y, float z)* | добавить новую точку в трехмерном пространстве X, Y, Z на графике |
| *long **PutXPoints** (float * Xpoints)* | записать массив координат по X, где размер массива должен быть не меньше NumPoints |
| *long **PutYPoints** (float * Ypoints)* | записать массив координат по Y, где размер массива должен быть не меньше NumPoints |
| *long **PutXYPoints**(float * Xpoints, float * Ypoints)* | записать массив координат по X и Y на графике, где размер массива должен быть не меньше NumPoints |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *long* ***PutZPoints****(float * Zpoints)* | записать массив координат по Z на графике, где размер массива должен быть не меньше NumPoints |
| *long* ***PutXYZPoints****(float * Xpoints, float * Ypoints, float * Zpoints)* | записать массив координат по X, Y и Z на графике, где размер массива должен быть не меньше NumPoints |
| *long* ***PutXPointsNet****(long pXpoints)* | записать массив координат по X на графике (для языков программирования .NET), где размер массива должен быть не меньше NumPoints |
| *long* ***PutYPointsNet****(long pYpoints)* | записать массив координат по Y на графике (для языков программирования .NET), где размер массива должен быть не меньше NumPoints |
| *long* ***PutZPointsNet****(long pZpoints)* | записать массив координат по Z на графике (для языков программирования .NET) , где размер массива должен быть не меньше NumPoints |
| *long* ***PutXYPointsNet****(long pXpoints, long pYpoints)* | записать массив координат по X и Y на графике (для языков программирования .NET), где размер массива должен быть не меньше NumPoints. |
| *long* ***PutXYZPointsNet****(long pXpoints, long pYpoints, long pZpoints)* | записать массив координат по X, Y и Z на графике (для языков программирования .NET) , где размер массива должен быть не меньше NumPoints |
| **Настройки курсора PlotterXY.ocx** | |
| *long* ***SetCursorPositionByIndex****(LONG index)* | устанавливает курсор в позицию точки с индексом Index\nВозвращает 0 в случае ошибки, в остальных случаях 1 |
| *long* ***GetIndexForCursorPosition****()* | получить значение индекса для позиции курсора\nВозвращает -1 в случае ошибки, в остальных случаях значение индекса |
| **События от компоненты PloterXY.ocx** | |
| *void* ***KeyDown****(SHORT* KeyCode, SHORT Shift)* | событие возникает при нажатии на клавишу клавиатуры, когда фокус находится на компоненте |
| *void* ***MouseDown****(short Button, short Shift, OLE_XPOS_PIXELS x, OLE_YPOS_PIXELS y)* | событие возникает при нажатии на кнопку манипулятора типа «мышь», при нахождении мыши над графиком |
| *void* ***MouseUp****(short Button, short Shift,* | событие возникает при отжатии кнопки манипулятора типа «мышь», при нахождении мыши над графиком |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *OLE_XPOS_PIXELS x, OLE_YPOS_PIXELS y)* | |
| *void* ***MouseWheel****(void)* | событие возникает при вращении ролика манипулятора типа «мышь» |
| *void* ***Modify*** *(void)* | событие, которое происходит при включении и отключении каналов АЦП, ЦАП, виртуальных каналов, изменении коэффициентов усиления и т.д. |

# Глава 4.Polar.ocx компонент отображения графиков в полярных координатах

**Polar.ocx** - графический компонент отображения графиков в полярных координатах

временную реализацию (x,t), (y,t) или (z,t) на плоскостях XT, YT или ZT ;

Графический компонент Polar предназначен для отображения графической информации в полярных координатах.

## 4.1.Установка компонента Polar.ocx

Для работы с модулем Polar его необходимо сначала установить в программу. При загрузке программы необходимо загрузить с помощью компонента необходимую программу, разместить окно программы на экране в удобном месте, установить необходимые параметры в программе и затем по событиям, происходящим от программы считывать данные из программы.

Для установки компонента Polar в проект *Microsoft Visual Studio 2010*, необходимо кликнуть правой кнопкой мыши на форме диалога и из появившегося меню выбрить пункт *Insert ActiveX Control*...(рисунок 1.1)

© ООО "ЭТМС" 1992-2024. Все права защищены

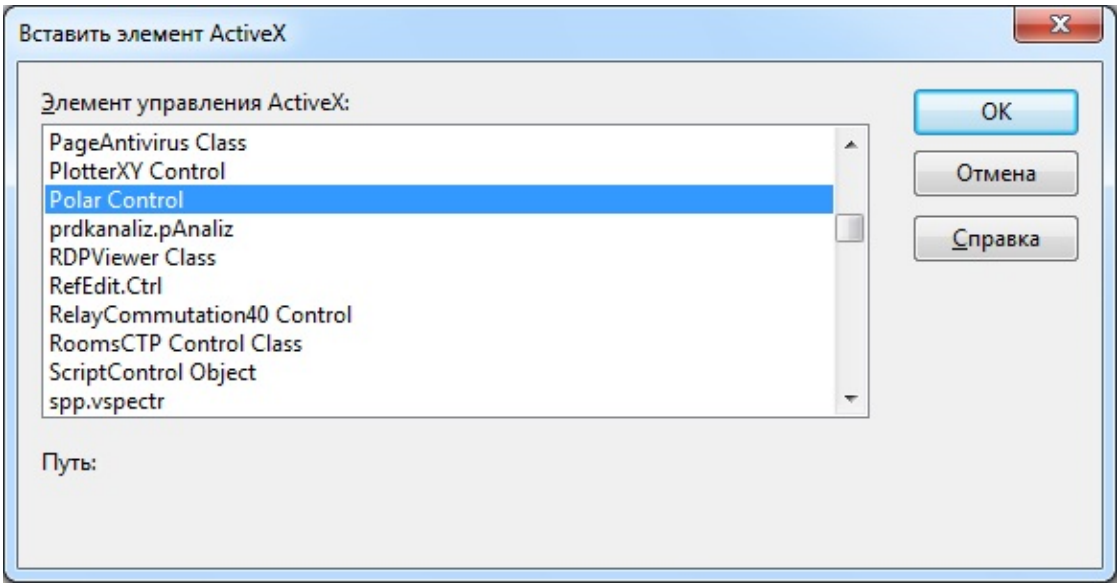



*Рисунок 1.1*

Из появившегося диалогового окна следует выбрать компонент *Polar Control* и нажать *OK* (рисунок 1.2). После этого компонент *Polar.ocx* появится на форме диалога (рисунок 1.2).

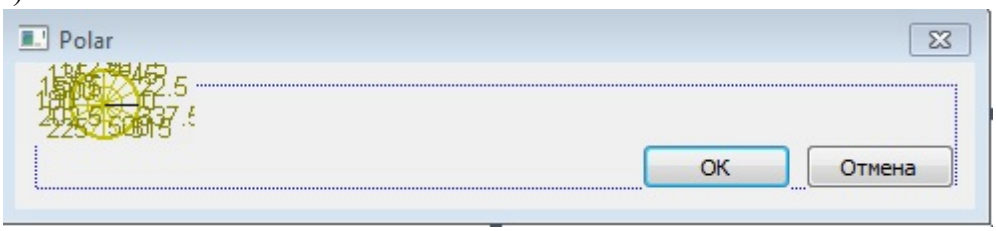



Рисунок 1.2

Одна установленная на форме компонента позволяет управлять одной программой. При необходимости управлять несколькими программами, необходимо установить на форму несколько компонент. Для того чтобы компонента не была видна на форме во время выполнения программы, необходимо установить свойство *Visible* равным *False*.

## 4.2.Назначение Polar.ocx

Компонент позволяет:

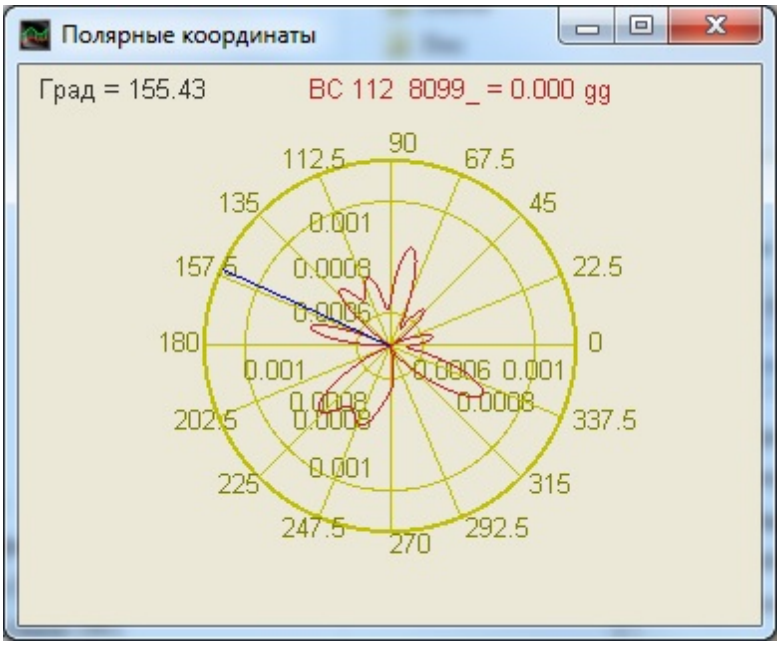
- задавать начальный, конечный угол и направление отображения;
- масштабировать график по уровню;
- перемещать курсор при помощи кнопки мыши, ролика мыши и клавиатуры;
- показывать значения угла и уровня на позиции курсора;

© ООО "ЭТМС" 1992-2024. Все права защищены

- задавать цвет графика, осей, цифр, фона, курсора;

сохранять данные в буфере обмена (clipboard) в графическом, текстовом и численном виде для создания отчетов в редакторах Microsoft Word и Excel.

Применяется, например, в программе "**Анализ нелинейных искажений**", "**Синхронный анализ**". На рисунках представлены графики виброускорения и виброперемещения с наложением на зубчатую передачу, условно изображенную серым цветом в программе "**Анализ нелинейных искажений**".




## 4.3.Полное описание свойств, методов и событий Polar.ocx

Следует отметить, что компонент автомасштабируется при изменении его размеров, поэтому необходимо следить за тем, чтобы высота компонента в приложении была не менее 80 пикселей, иначе он может отображаться некорректно.

### Свойства Polar.ocx

Свойства можно задавать на этапе проектирования и во время выполнения программы. На этапе проектирования при выборе компонента и при нажатии клавиши <F4> появляется окно свойств. Эти свойства можно устанавливать на этапе проектирования программы.

Цвета задаются при помощи макроса *RGB(RED, GREEN, BLUE)*. Значения *RED*, *GREEN*, *BLUE* задают интенсивности цветов красного, зеленого и синего. Интенсивности изменяются в пределах от 0 до 255.

© ООО "ЭТМС" 1992-2024. Все права защищены

*OLE_COLOR* ***ClrCur*** –цвет курсора.

*OLE_COLOR* ***ClrFon*** – цвет фона.

*OLE_COLOR* ***ClrOsi*** - цвет осей.

*OLE_COLOR* ***ClrGrf*** – цвет выбранного графика. Для каждого графика цвет задается отдельно. Графики выбираются при помощи свойства *Ind*.

*OLE_COLOR* ***ClrNum*** – цвет цифр по осям.

*OLE_COLOR* ***ClrGear*** - цвет шестеренки

Например, для изменении фона графика с желтого на зеленый, могут быть записаны числа:

```
OLE_COLOR a = 0xFF00;
OLE_COLOR b = 0xFFFF;
```

*FLOAT* ***startu*** - начальный угол.

*FLOAT* ***endu*** - конечный угол.

*LONG* ***mysize*** - количество точек в графике.

*LONG* ***Number*** - количество графиков.

*USHORT* ***LineWidth*** - толщина линии графика.

*LONG* ***ind*** - номер графика.

*FLOAT* ***Zero*** - поворот нуля.

*LONG* ***degrade*** - отображение в градусах или радианах:

Параметры:

0 - в радианах;
1 - в градусах.

© ООО "ЭТМС" 1992-2024. Все права защищены

*LONG* ***pochas*** - отображение по часовой или против часовой стрелки.

Параметры:

0 - по часовой;
1 - против часовой.

*LONG* ***pol*** - отображение полного круга или полукруга.

*Параметры:*

0 - полный круг;
1 - полукруг.

*FLOAT* ***MathLA*** - минимальное значение по амплитуде для отображения.

*FLOAT* ***MathDA*** - диапазон отображаемых амплитуд.

*FLOAT* ***UporANis*** - минимально допустимая амплитуда.

*FLOAT* ***UporAVerh*** - максимально допустимая амплитуда.

*FLOAT* ***UporDelta*** - минимальное разрешение по амплитуде.

При работе программы в режиме исполнения пользователь может менять область отображения графиков при помощи курсора мыши. Растяжение или сжатие графиков происходит при помощи указателя вида: ↔ , ⇹ – для горизонтальной оси.

Сдвинуть графики вправо-влево можно при помощи указателя ← , → – для горизонтальной оси. При этом меняются параметры *MathLA, MathDA.* В пользовательской программе в режиме исполнения можно прочитать текущие параметры области отображения.

## Методы Polar.ocx

Методы можно использовать только во время выполнения программы.

*void* ***AboutBox()*** –содержит информацию о программе.

© ООО "ЭТМС" 1992-2024. Все права защищены

*void* ***Reset*** *(void)* – обнуляет массивы с координатами.

*short* ***Display*** *(void)* – метод перерисовывает весь график.

*long* ***Mpaint*** *(FLOAT* bf)* – координата точки на графике.

*void* ***put_ChhTextPol****(BSTR ch_text,LONG num)* - задает текст и номер канала

*void* ***put_cht****(BSTR text, LONG num)* - задает текст и номер канала (похожа на предыдущую функцию)

*SHORT* ***DrawGear****(LONG num, LONG vkl)* – отрисовать шестеренку

*void* ***PutPictureToClipboard****(void)* - копировать график в буфер обмена

*void* ***PutDataToClipboard****(void)* - скопировать данные в буфер обмена

*long* ***FormatAmpl*** (*BSTR* fmt) – метод передает в компонент строку для форматного вывода информации по амплитуде в легенде графика. Форматный ввод-вывод позволяет передавать программе и выдавать из программы на внешнее устройство символьные, строковые и числовые значения, обеспечивая преобразование данных в процессе ввода-вывода. Метод должен выполняться для каждого графика. Графики выбираются при помощи свойства *Ind*.

*long* ***FormatUgol*** *(BSTR fmt)* – метод передает в компонент строку для форматного вывода информации по величине угла в легенде графика. Форматный ввод-вывод позволяет передавать программе и выдавать из программы на внешнее устройство символьные, строковые и числовые значения, обеспечивая преобразование данных в процессе ввода-вывода. Метод должен выполняться для каждого графика. Графики выбираются при помощи свойства *Ind*.

*long* ***FormatCap*** *(BSTR fmt)* - позволяет выводить информацию в заголовок графика

## 4.4.Краткий перечень функций, используемые Polar.ocx

© ООО "ЭТМС" 1992-2024. Все права защищены

| Свойства Polar.ocx | |
|---|---|
| **Цвета** | |
| *long **GetClrCur**()* | опрашивается цвет курсора |
| *void **SetClrCur** (unsigned long propVal)* | *устанавливается цвет курсора* |
| *long **GetClrFon** ()* | опрашивается цвет фона |
| *void **SetClrFon** (unsigned long propVal)* | устанавливается цвет фона |
| *long **GetClrOsc** ()* | опрашивается цвет осей |
| *void **SetClrOsc**(unsigned long propVal)* | устанавливается цвет осей |
| *long **GetClrGrf**()* | опрашивается цвет выбранного графика |
| *void **SetClrGrf**(unsigned long propVal)* | устанавливается цвет выбранного графика |
| *long **GetClrNum** ()* | опрашивается цвет цифр по осям |
| *void **SetClrNum** (unsigned long propVal)* | устанавливается цвет цифр по осям |
| *long **GetClrGear**()* | опрашивается цвет шестеренки |
| *void **SetClrGear**(unsigned long propVal)* | устанавливается цвет шестеренки |
| **Параметры графика** | |
| *float **Getstartu**()* | опрашивается начальный угол |
| *void **Setstartu**(float propVal)* | устанавливается начальный угол |
| *float **Getendu**()* | опрашивается конечный угол |
| *void **Setendu**(float propVal)* | устанавливается конечный угол |
| *long **Getmysize** ()* | опрашивается количество точек в графике |
| *void **Setmysize** (long propVal)* | устанавливается количество точек в графике |
| *long **GetNumber** ()* | опрашивается количество графиков |
| *void **SetNumber** (long propVal)* | устанавливается количество графиков |
| *short **GetLineWidth**()* | опрашивается толщина линии графика |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *void* ***SetLineWidth****(unsigned short propVal)* | устанавливается толщина линии графика |
| *long* ***Getind****()* | опрашивается номер графика |
| *void* ***Setind*** *(long propVal)* | устанавливается номер графика |
| *float* ***GetZero****()* | опрашивается поворот нуля |
| *void* ***SetZero****(float propVal)* | устанавливается поворот нуля |
| **Представление графика** | |
| *long* ***Getdegrad****()* | опрашивается отображение в градусах или радианах (0 - в радианах;<br>1 - в градусах) |
| *void* ***Setdegrad*** *(long propVal)* | устанавливается отображение в градусах или радианах (0 - в радианах;<br>1 - в градусах) |
| *long* ***Getpochas*** *()* | опрашивается отображение по часовой или против часовой стрелки (0 - по часовой; 1 - против часовой) |
| *void* ***Setpochas****(long propVal)* | устанавливается отображение по часовой или против часовой стрелки (0 - по часовой; 1 - против часовой) |
| *long* ***Getpol*** *()* | опрашивается отображение полного круга или полукруга (0 - полный круг; 1 – полукруг) |
| *void* ***Setpol*** *(long propVal)* | Устанавливается отображение полного круга или полукруга (0 - полный круг; 1 – полукруг) |
| *float* ***GetMathLA*** *()* | опрашивается минимальное значение по амплитуде для отображения |
| *void* ***SetMathLA*** *(float propVal)* | устанавливается минимальное значение по амплитуде для отображения |
| *double* ***GetMathDA*** *()* | опрашивается диапазон отображаемых амплитуд |
| *void* ***SetMathDA*** *(double propVal)* | устанавливается диапазон отображаемых амплитуд |
| *float* ***GetUporANis*** *()* | опрашивается минимально допустимая амплитуда |
| *void* ***SetUporANis*** *(float propVal)* | Устанавливается минимально допустимая амплитуда |
| *float* ***GetUporAVerh*** *()* | опрашивается максимально допустимая амплитуда |
| *void* ***SetUporAVerh*** *(float propVal)* | устанавливается максимально допустимая амплитуда |
| *float* ***GetUporDelta*** *()* | опрашивается минимальное разрешение по амплитуде |
| *void* ***SetUporDelta*** *(float propVal)* | устанавливается минимальное разрешение по амплитуде |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Методы Polar.ocx | |
|---|---|
| **Параметры графика** | |
| *void* ***AboutBox****()* | содержит информацию о программе |
| *void* ***Reset*** *(void)* | обнуляет массивы с координатами |
| *short* ***Display*** *(void)* | перерисовывает весь график |
| *long* ***Mpaint*** *(FLOAT* bf)* | координата точки на графике |
| *void* ***put_ChhTextPol****(BSTR ch_text,LONG num* | задает текст и номер канала |
| *void* ***put_cht****(BSTR text, LONG num)* | задает текст и номер канала (похожа на предыдущую функцию) |
| *short* ***DrawGear****(long num, long vkl)* | отрисовать шестеренку |
| *void* ***PutPictureToClipboard****(void)* | копировать график в буфер обмена |
| *void* ***PutDataToClipboard****(void)* | скопировать данные в буфер обмена |
| **Передача данных** | |
| *long* ***FormatAmpl*** (*BSTR* fmt) | передает в компонент строку для форматного вывода информации по амплитуде в легенде графика |
| *long* ***FormatUgol*** *(BSTR fmt)* | передает в компонент строку для форматного вывода информации по величине угла в легенде графика |
| *long* ***FormatCap*** *(BSTR fmt)* | выводить информацию в заголовок графика |

# Глава 5.Gramma.ocx предназначена для графического отображения массивов данных

Компонента **Gramma.ocx** предназначена для графического отображения массивов данных в виде графиков зависимостей $z = z(x, y)$ Процедура рисования

© ООО "ЭТМС" 1992-2024. Все права защищены

графиков написана таким образом, что отсутствует мерцание при перерисовке графиков.

## 5.1.Установка компонента Gramma.ocx

Для работы с модулем Gramma его необходимо сначала установить в программу. При загрузке программы необходимо загрузить с помощью компонента необходимую программу, разместить окно программы на экране в удобном месте, установить необходимые параметры в программе и затем по событиям, происходящим от программы считывать данные из программы.

Для установки компонента Gramma в проект *Microsoft Visual Studio 2010*, необходимо кликнуть правой кнопкой мыши на форме диалога и из появившегося меню выбрить пункт *Insert ActiveX Control*...(рисунок 1.1)

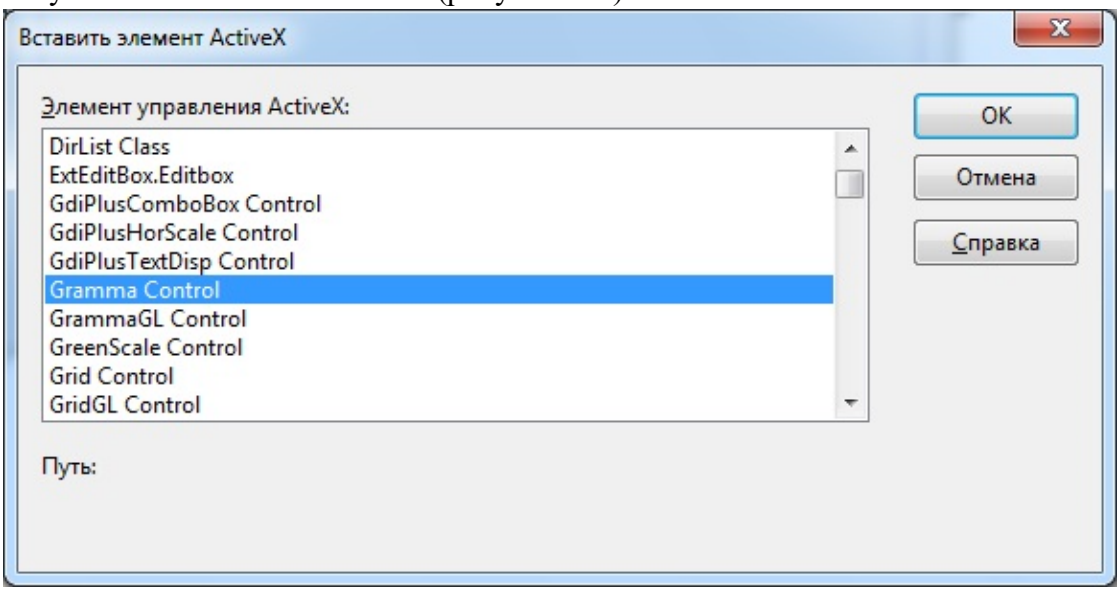



*Рисунок 1.1*

Из появившегося диалогового окна следует выбрать компонент Gramma *Control* и нажать *OK* (рисунок 1.2). После этого компонент *Gramma.ocx* появится на форме диалога (рисунок 1.2).

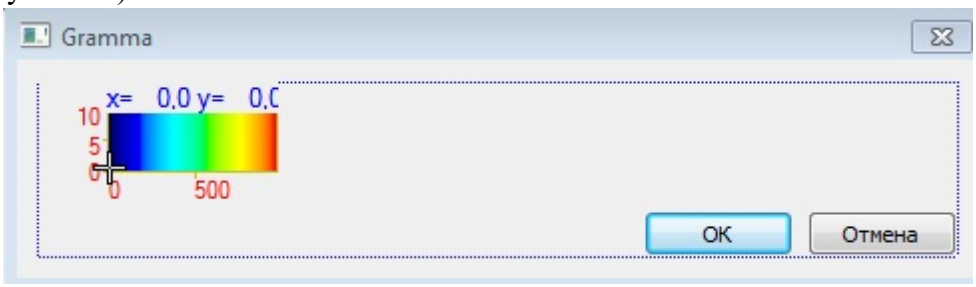



Рисунок 1.2

Одна установленная на форме компонента позволяет управлять одной программой. При необходимости управлять несколькими программами, необходимо

© ООО "ЭТМС" 1992-2024. Все права защищены

установить на форму несколько компонент. Для того чтобы компонента не была видна на форме во время выполнения программы, необходимо установить свойство *Visible* равным *False*.

## 5.2.Назначение Gramma.ocx

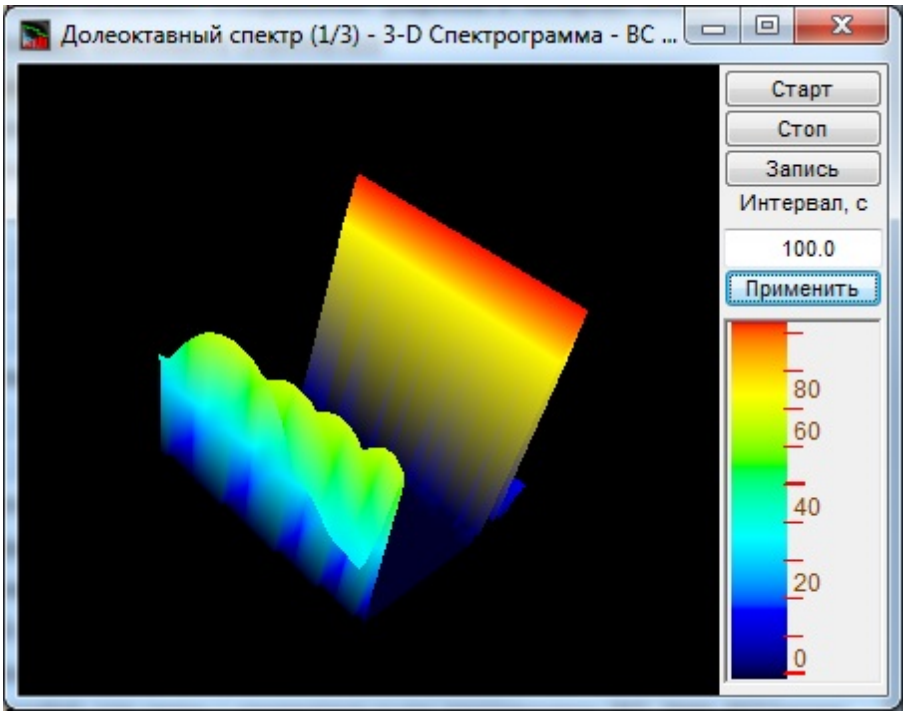
Компонент позволяет:

- масштабировать график;
- отображать данные в трехмерном и двумерном виде;
- отображать графики в цветном и черно-белом виде;
- перемещать курсор при помощи кнопки мыши, ролика мыши и клавиатуры;
- показывать значения абсцисс и ординат на позиции курсора;
- сохранять данные в буфере обмена (clipboard) в графическом, текстовом и численном виде для создания отчетов в редакторах Microsoft Word и Excel.

Применяется, например, в программах: "**Просмотр результатов**", "**Узкополосный спектр", "Взаимный узкополосный спектр",** "**Долеоктавный спектр**", "**Взаимный долеоктавный спектр", "Взаимный корреляционный анализ**", "**Взаимный долеоктавный спектр", "Синхронное накопление".**

На рисунках представлена 3D спектрограммы из программы "**Долеоктавный спектр**".

© ООО "ЭТМС" 1992-2024. Все права защищены




3D спектрограмма

## 5.3.Полное описание свойств, методов и событий Gramma.ocx

Следует отметить, что компонент автомасштабируется при изменении его размеров, поэтому необходимо следить за тем, чтобы высота компонента в приложении была не менее 80 пикселей, иначе он может отображаться некорректно.

### Свойства Gramma.ocx

Свойства можно задавать на этапе проектирования и во время выполнения программы. На этапе проектирования при выборе компонента и при нажатии клавиши <F4> появляется окно свойств. Эти свойства можно устанавливать на этапе проектирования программы.

График задается в виде двумерного массива. В компоненте задаются размеры по координате X и Y при помощи свойств Size и Number. Данные передаются в компонент массивом строки размером Size с помощью метода Paint(). Номер строки задается свойством Ind. После пересылки каждой строки прорисовывается график.

Цвета задаются при помощи макроса *RGB(RED, GREEN, BLUE)*. Значения *RED*, *GREEN*, *BLUE* задают интенсивности цветов красного, зеленого и синего. Интенсивности изменяются в пределах от 0 до 255.

© ООО "ЭТМС" 1992-2024. Все права защищены

*OLE_COLOR* ***ClrCrs*** – цвет курсора.

*OLE_COLOR* ***ClrDig*** – цвет цифр.

*OLE_COLOR* ***ClrFon*** – цвет фона.

*OLE_COLOR* ***ClrGrd*** – цвет сетки-меток.

*OLE_COLOR* ***ClrGrf*** – цвет выбранного графика. Для каждого графика цвет задается отдельно. Графики выбираются при помощи свойства *Ind*.

*OLE_COLOR* ***ClrLeg*** – цвет легенды.

*long* ***Black*** – цветное или черно-белое изображение.
0 – цветное изображение.
1 – черно-белое изображение.

Например, для изменении фона графика с желтого на зеленый, могут быть записаны числа:

*OLE_COLOR a = 0xFF00;*
*OLE_COLOR b = 0xFFFF;*

*long* ***Size*** – Количество точек по горизонтали. Размер не может быть меньше или равен 0. Размер не может быть больше 50000.

*long* ***Number*** – количество строк в изображении. Количество строк не может превышать 200.

*long* ***Ind*** – индекс (номер) строки. Индекс может принимать значения от 0 до Number-1.

*DOUBLE* ***DeltaT*** – *DOUBLE* ***DeltaT*** – шаг по времени по оси Y. Например, при отображении сонограммы отображаются последовательность спектров получаемых каждые 0.1 с и тогда *DeltaT* = 0.1.

*long* ***Log*** – логарифмическая шкала по Y. Параметр может принимать значения 0 или 1.

0 – линейный масштаб.
1 – логарифмический масштаб.

*long* ***TypeAbs*** – тип осей координат:

0 - рисуется сетка и по X и по Y (равномерная шкала по X),
1 - логарифмическая сетка по X,
2 - не рисуются горизонтальные линии,

© ООО "ЭТМС" 1992-2024. Все права защищены

4 - рисуются вертикальные линии,
6 - не рисуется сетка.

*long* ***Px*** – позиция курсора по оси X. Этот параметр принимает значение от 0 до *Size*-1. Параметр можно читать и записывать. При перемещении курсора при помощи мыши или клавиатуры этот параметр отслеживает положение курсора.

*long* ***Py*** – позиция курсора по оси Y. Этот параметр принимает значение от 0 до *Number*-1. Параметр можно читать и записывать. При перемещении курсора при помощи мыши или клавиатуры этот параметр отслеживает положение курсора.

*long* ***CommentInd*** – порядковый номер комментария для записи в файл.

*BSTR* ***CommentSTR*** – строка комментария для записи в файл.

*long* ***MaxBuffer*** – определяется максимальный размер буфера.

*long* ***MakeUpor*** – включение ограничений по оси *Z*. При масштабировании графиков при помощи клавиатуры и мыши на стадии выполнения программы возможна ситуация, когда происходит переполнение данных. Для включения ограничений по масштабированию необходимо установить параметр *MakeUpor* в 1. Для снятия ограничений в 0.

*DOUBLE* ***DeltaT*** - шаг по времени (по вертикали)

*DOUBLE* ***MathLX*** – область отображения данных по оси *X*. Левая граница отображения данных.

*DOUBLE* ***MathDX***– область отображения данных по оси *X*. Длина отображения данных.

Например, необходимо отобразить часть данных из предыдущего примера в диапазоне от 1050 *Гц* до 1070 *Гц*. В этом случае задаются следующие параметры:
*MathLX* = 1050.
*MathDX* = 20.
При работе программы в режиме исполнения пользователь может менять область отображения графиков при помощи курсора мыши. Растяжение или сжатие графиков происходит при помощи указателя вида: ↔ , ⇹ – для горизонтальной оси.

Сдвинуть графики вправо-влево можно при помощи указателя ← , → – для горизонтальной оси. При этом меняются параметры *MathLX, MathDX.* В

© ООО "ЭТМС" 1992-2024. Все права защищены

пользовательской программе в режиме исполнения можно прочитать текущие параметры области отображения.

*DOUBLE* ***MathLY*** – начало отображения данных по уровню. Левая граница отображения данных.

*DOUBLE* ***MathDY***– диапазон отображения по уровню. Длина отображения данных.

Растяжение или сжатие графиков происходит при помощи указателя вида:  ,  – для вертикальной оси. Сдвинуть графики вверх или вниз можно при помощи указателя  ,  – для вертикальной оси. При этом меняются параметры *MathLY, MathDY.* В пользовательской программе в режиме исполнения можно прочитать текущие параметры области отображения. При включенных ограничениях отображения графиков *MakeUpor* = 1 , параметр *MathDY* не может быть меньше *UporNis ,* сумма параметров *MathLY* + *MathDY* не может быть больше *UporVerh*.

*DOUBLE* ***Xfirst*** – начальное значение по координате *X*.

*DOUBLE* ***Xend*** – конечное значение по координате *X*.

Например, необходимо отобразить массив из 100 чисел в равномерной сетке по оси *X*. Числа представляют собой спектр мощности сигнала в диапазоне от 1000 Гц до 1100 Гц. Т.е. 0-ому элементу массива соответствует частота 1000 Гц, 99-му – частота 1100 Гц. В этом случае задаются следующие параметры:

Xfirst = 1000.

*Xend* = 1100.

*double* ***UporNis*** – минимальная величина допустимого диапазона отображения по Y (при включенном ограничении), которая может отобразиться на графике. Это свойство работает при включенных ограничениях по оси *Y*.

*double* ***UporVerh*** – максимальная величина по допустимого диапазона отображения по Y (при включенном ограничении), которая может отображаться на графике. Это свойство работает при включенных ограничениях по оси *Y*.

**Пример**. Отображение временной реализации оцифрованного АЦП сигнала в виде осциллограммы. Входной диапазон работы АЦП от -10 В до + 10 В, вес младшего разряда 16-разрядного АЦП 0.0003 В. В этом случае можно задать следующие параметры.

MakeUpor = 1

*UporNis* = 0.0003

*UporVerh* = 10.0

© ООО "ЭТМС" 1992-2024. Все права защищены

## Методы Gramma.ocx

Методы можно использовать только во время выполнения программы.

*void* ***AboutBox****()* –содержит информацию о программе.

*short* ***Display*** *(void )* – метод перерисовывает изображения

*void* ***Reset*** *(void)* – сброс к первоначальному состоянию.

*void* ***PushToClipBoard****(void)* – метод, позволяет сохранить изображение графика в Clipboard в графическом (bmp) формате.

*long* ***SaveGLDataToFile****(BSTR FileName)* – метод, позволяет сохранить массив данных в файле. Сохранение данных в текстовый файл с расширением *.dtu". Файл записывается в текстовом виде со всеми комментариями. Это позволяет использовать данные из этого файла для других программ обработки. *FileName* – строка полного пути и имени файла.

*long* ***GetGLDataFromFile****(BSTR FileName)* – метод позволяет взять данные из файла и отобразить в графики. Файл должен быть создан при помощи метода *SaveGLDataToFile(...).*

В файл создаваемый методом *SaveGLDataToFile(...)* можно записывать 8 строк дополнительной информации. Для записи дополнительной информации необходимо установить номер индекса комментария от 0 до 7 и записать строку комментария.

*float* ***GetPointAt****(long X, long Y)* – – метод позволяет получить позицию курсора.

*long* ***Paint*** *(float* Buffer)* – метод для передачи во внутренний буфер очередной строки изображения. При построении необходимо при помощи свойства *Ind* выбирать, номер строки для перерисовки.

***Параметры****:*

*float* Buffer* – адрес массива данных в программе пользователя. В компоненту копируется количество данных определенное свойством *Size*.

*long* ***PaintNet*** *(long pBuffer)* – метод для передачи во внутренний буфер очередной строки изображения. При построении необходимо при помощи свойства *Ind* выбирать, номер строки для перерисовки.

Параметры:

© ООО "ЭТМС" 1992-2024. Все права защищены

*long pBuffer* – адрес числового массива данных в программе пользователя. В компоненту копируется количество данных определенное свойством *Size*.

При создании программ обработки сигналов в реальном времени с многократной перерисовкой графиков, типа осциллографов или анализаторов спектра, необходимо один раз задать параметры компонента, и затем последовательно обращаться только к методу ***PaintNet****(...)* для создания эффекта движения.

*void* ***AltC*** *(float *buffer)* – метод, позволяет загрузить массив разметки данных по *X*. Этот метод работает при установленном свойстве типа сетки *TypeAbs* = 2. В пользовательском буфере *buffer* размера *Size* необходимо записать значения координаты *X*. Например, при логарифмической развертке по оси *X*, в буфере могут быть записаны числа:

*buffer[0] = 1.*
*buffer[1] = 1.25*
*buffer[2] = 1.6*
*buffer[3] = 2.*
*buffer[4] = 2.5*

*void* ***AltCNet*** *(long pBuffer)* – метод, позволяет загрузить массив разметки данных по *X*. Этот метод работает при установленном свойстве типа сетки *TypeAbs* = 2. В пользовательском буфере *buffer* размера *Size* необходимо записать значения координаты *X*. Например, при логарифмической развертке по оси *X*, в буфере могут быть записаны числа:

*buffer[0] = 1.*
*buffer[1] = 1.25*
*buffer[2] = 1.6*
*buffer[3] = 2.*
*buffer[4] = 2.5*

**Примечание:** Следует отметить, то при работе в .net языках функции, в которых в качестве параметров выступают указатели на массив, например GetData в .net языках работают некорректно, поэтому следует использовать обновленные функции, имеющие такое же название + приписка Net, например, GetDataNet.

*long* ***Formatxy*** *(BSTR fmt)* – метод передает в компонент строку для форматного вывода информации по координатам *X, Y* и *Z* в легенде графика. В соответствии с заданным форматом отображается положение и уровень Форматный ввод-вывод позволяет передавать программе и выдавать из программы на внешнее устройство символьные, строковые и числовые значения, обеспечивая преобразование данных в процессе ввода-вывода.

Например, обращение вида:

FormatX(“Частота %7.2fГц Время %7.2fс Уровень %7.3fВ”)

позволяет отображать координаты по оси X,Y и Z в значениях по частоте.

Формат представления чисел.

© ООО "ЭТМС" 1992-2024. Все права защищены

%f – общий формат представления числа в плавающей запятой в виде числа с точкой

%e – общий формат представления числа в плавающей запятой в виде числа с экспонентой (1000 = 1e3, 0.001 = 1e-3)

%7.2f – представление числа в фиксированном формате. В данном примере 7 – количество знакомест для представления числа, 2 – количество цифр после запятой в представлении числа.

long ***PaintDouble***(double* buffer) - метод, позволяет передать данные для отображения графика и перерисовать график.

Параметры:

double* buffer – адрес числового массива данных в программе пользователя. В компоненту копируется количество данных определенное свойством *Size*.

При создании программ обработки сигналов в реальном времени с многократной перерисовкой графиков, типа осциллографов или анализаторов спектра, необходимо один раз задать параметры компонента, и затем последовательно обращаться только к методу ***PaintDouble****(...)* для создания эффекта движения.

void ***AltCDouble***(double* buffer) - метод, позволяет загрузить массив разметки данных по *X*.

Этот метод работает при установленном свойстве типа сетки *TypeAbs* = 2. В пользовательском буфере *buffer* размера *Size* необходимо записать значения координаты *X*. Например, при логарифмической развертке по оси *X*, в буфере могут быть записаны числа:

*buffer[0] = 1.*
*buffer[1] = 1.25*
*buffer[2] = 1.6*
*buffer[3] = 2.*
*buffer[4] = 2.5*

## События от компоненты Gramma.ocx

*void* ***KeyDown*** *(SHORT* KeyCode, SHORT Shift)* – событие возникает при нажатии на клавишу клавиатуры, когда фокус находится на компоненте.

Параметры:

*KeyCode* – код клавиши

*Shift* – код нажатия служебных клавиш <*Ctrl*>,<*Shift*>,<*Alt*>.

*void* ***MouseDown*** *(short Button, short Shift, OLE_XPOS_PIXELS x, OLE_YPOS_PIXELS y)* – событие возникает при нажатии на кнопку манипулятора типа «мышь», при нахождении мыши над графиком.

**Примечание управляющим клавишам:**

© ООО "ЭТМС" 1992-2024. Все права защищены

<C> - Копирование графика. Для копирования графика нажать левой кнопкой «мыши» на поле графика и нажать комбинацию «горячих клавиш» клавиатуры <Ctrl> +<C>.

График запишется в буфер обмена(Clipboard).

<A> Управление движением курсора влево. Нужно нажать левой кнопкой «мыши» на поле графика программы и нажать на кнопку клавиатуры <A>(в латинской раскладке клавиатуры).

<D> Управление движением курсора вправо. Нужно нажать левой кнопкой «мыши» на поле графика программы и нажать на кнопку клавиатуры <D>(в латинской раскладке клавиатуры).

<W> Управление движением курсором вверх. Нужно нажать левой кнопкой «мыши» на поле графика программы и нажать на кнопку клавиатуры <W>(в латинской раскладке клавиатуры).

<S> Управление движением курсором вниз. Нужно нажать левой кнопкой «мыши» на поле графика программы и нажать на кнопку клавиатуры <S>(в латинской раскладке клавиатуры).

Параметры:
*Button* – код кнопки мыши
*Shift* – код нажатия служебных клавиш *<Ctrl>,<Shift>,<Alt>*.
*x* – координата по оси *X* в пикселах (разрешении экрана)
*y* – координата по оси *Y* в пикселах (разрешении экрана)

*void* ***MouseUp*** *(short Button, short Shift, OLE_XPOS_PIXELS x, OLE_YPOS_PIXELS y)* – событие возникает при отжатии кнопки манипулятора типа «мышь», при нахождении мыши над графиком.

Параметры:
*Button* – код кнопки мыши
*Shift* – код нажатия служебных клавиш *<Ctrl>,<Shift>,<Alt>*.
*x* – координата по оси *X* в пикселях (разрешении экрана)
*y* – координата по оси *Y* в пикселях (разрешении экрана)

v*oid* ***MouseWheel*** *(void)* – событие возникает при вращении ролика манипулятора типа «мышь».

## 5.4.Краткий перечень функций, используемые Gramma.ocx

| **Свойства Gramma.ocx** | |
|---|---|
| **Цвета** | |
| *long* ***GetClrCrs****()* | опрашивается цвет курсора |
| *void* ***SetClrCrs****(unsigned long propVal)* | устанавливается цвет курсора |
| *long* ***GetClrDig****()* | опрашивается цвет цифр |
| *void* ***SetClrDig****(unsigned long propVal)* | устанавливается цвета цифр |
| *long* ***GetClrFon*** *()* | опрашивается цвет фона |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *void **SetClrFon** (unsigned long propVal)* | устанавливается цвет фона |
| *long **GetClrGrd** ()* | опрашивается цвет сетки-меток |
| *void **SetClrGrd** (unsigned long propVal)* | устанавливается цвет сетки-меток |
| *long **GetClrGrf** ()* | опрашивается цвет графика |
| *void **SetClrGrf** (unsigned long propVal)* | устанавливается цвет графика. Для каждого графика цвет задается отдельно. Графики выбираются при помощи свойства ***Ind***. |
| *long **GetClrLeg** ()* | опрашивается цвет легенды графика |
| *void **SetClrLeg** (unsigned long propVal)* | устанавливается цвет легенды графика |
| *long **GetBack** ()* | опрашивается цветное или черно-белое изображение (0 – цветное /1 – черно-белое) |
| *void **SetBack** (unsigned long propVal)* | устанавливается цветное или черно-белое изображение (0 – цветное /1 – черно-белое) |
| **Параметры графиков** | |
| *long **GetSize** ()* | опрашивается количество точек на графике. Размер не может быть меньше или равен 0. Размер не может быть больше 50000 |
| *void **SetSize** (long propVal)* | устанавливается количество точек на графике. Размер не может быть меньше или равен 0. Размер не может быть больше 50000 |
| *long **GetNumber**()* | опрашивается количество отображаемых графиков. Количество одновременно отображаемых графиков в одних осях не может превышать 20 |
| *void **SetNumber**(long propVal)* | устанавливается количество отображаемых графиков. Количество одновременно отображаемых графиков в одних осях не может превышать 20 |
| *long **GetInd**()* | опрашивается индекс (номер) графика. Индекс может принимать значения от 0 до Number-1 |
| *void **SetInd** (long propVal)* | устанавливается индекс (номер) графика. Индекс может принимать значения от 0 до Number-1 |
| *long **Getlog** ()* | опрашивается тип шкалы Y (0 - линейная или 1 - логарифмическая) |
| *void **Setlog** (long propVal)* | устанавливается тип шкалы Y (0- линейная или 1 - логарифмическая) |
| *long **GetTypeAbs** ()* | опрашивается тип осей координат (0 - рисуется сетка и по X и по Y (равномерная шкала по X), 1 - логарифмическая сетка по X, 2 - не рисуются горизонтальные линии, 4 - рисуются вертикальные линии, 6 - не рисуется сетка) |
| *void **SetTypeAbs** (long propVal)* | устанавливается тип осей координат (0 - рисуется сетка и по X и по Y (равномерная шкала по X), 1 - логарифмическая сетка по X, 2 - не рисуются горизонтальные линии, 4 - рисуются вертикальные линии, 6 - не рисуется сетка) |
| *long **GetPx**()* | опрашивается позиция курсора по оси X. Этот параметр принимает значение от 0 до Size-1 |
| *void **SetPx** (long propVal)* | устанавливается позиция курсора по оси X. Этот параметр принимает значение от 0 до Size-1 |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *long* ***GetPy****()(* | опрашивается позиция курсора по оси Y. Этот параметр принимает значение от 0 до Size-1 |
| *long* ***SetPy****()(long propVal)* | Устанавливается позиция курсора по оси Y. Этот параметр принимает значение от 0 до Size-1 |
| *long* ***GetCommentInd ()*** | опрашивается порядковый номер комментария для записи в файл |
| *void* ***SetCommentInd*** *(long propVal)* | устанавливается порядковый номер комментария для записи в файл |
| *CString* ***GetCommentSTR****()* | Опрашивается строка комментария для записи в файл |
| *void* ***SetCommentSTR****(CString propVal)* | Устанавливается строка комментария для записи в файл |
| *long* ***GetMaxBuffer ()*** | опрашивается максимальный размер буфера |
| *void* ***SetMaxBuffer*** *(long propVal)* | устанавливается максимальный размер буфера |
| **Представление графиков** | |
| *long* ***GetMakeUpor****()* | опрашивается включение ограничений по оси Z. При масштабировании графиков при помощи клавиатуры и мыши на стадии выполнения программы возможна ситуация, когда происходит переполнение данных. Для включения ограничений по масштабированию необходимо установить параметр MakeUpor в 1. Для снятия ограничений в 0 |
| *void* ***SetMakeUpor****(long propVal)* | устанавливается включение ограничений по оси Z. При масштабировании графиков при помощи клавиатуры и мыши на стадии выполнения программы возможна ситуация, когда происходит переполнение данных. Для включения ограничений по масштабированию необходимо установить параметр MakeUpor в 1. Для снятия ограничений в 0 |
| *double* ***GetDeltaT*** *()* | опрашивается шаг по времени (по вертикали) |
| *void* ***DeltaT*** *(float propVal)* | устанавливается шаг по времени (по вертикали) |
| *double* ***GetMathlx****()* | опрашивается минимальное значение по X. Левая граница отображения данных |
| *void* ***SetMathlx****(double propVal)* | устанавливается минимальное значение по X. Левая граница отображения данных |
| *double* ***GetMathdx****()* | опрашивается диапазон отображения по X. Длина отображения данных |
| *void* ***SetMathdx****(double propVal)* | устанавливается диапазон отображения по X. Длина отображения данных |
| *double* ***GetMathly****()* | опрашивается минимальное значение по Y. Нижняя граница отображения данных |
| *void* ***SetMathly****(double propVal)* | Устанавливается минимальное значение по Y. Нижняя граница отображения данных |
| *double* ***GetMathdy****()* | опрашивается диапазон отображения по Y. Высота отображения данных |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *void* ***SetMathdy****(double propVal)* | устанавливается диапазон отображения по Y. Высота отображения данных |
| *double* ***GetXend*** *()* | опрашивается начальное значение по координате X |
| *void* ***SetXend*** *(double propVal)* | устанавливается начальное значение по координате X |
| *double* ***GetXfirst****()* | опрашивается конечное значение по координате X |
| *void* ***SetXfirst*** *(double propVal)* | устанавливается конечное значение по координате X |
| *double* ***GetUporNis*** *()* | опрашивается нижний допустимый диапазон отображения по Y (при включенном ограничении) |
| *void* ***SetUporNis*** *(float propVal)* | устанавливается допустимый диапазон отображения по Y (при включенном ограничении) |
| *double* ***GetUporVerh*** *()* | опрашивается верхний допустимый диапазон отображения по Y (при включенном ограничении) |
| *void* ***SetUporVerh*** *(float propVal)* | устанавливается верхний допустимый диапазон отображения по Y (при включенном ограничении) |
| **Методы Gramma.ocx** | |
| **Настройка Gramma.ocx** | |
| *void* ***AboutBox****()* | содержит информацию о программе |
| *short* ***Display*** *(void)* | перерисовывает изображения |
| *void* ***Reset*** *()* | сброс к первоначальному состоянию |
| *void* ***PushDataToClipBoard*** *(void)* | позволяет сохранить изображение графика в Clipboard в графическом (bmp) формате |
| *long* ***SaveGLDataToFile*** *(BSTR FileName)* | позволяет сохранить массив данных в файле. Сохранение данных в текстовый файл с расширением *.dtu". Файл записывается в текстовом виде со всеми комментариями. Это позволяет использовать данные из этого файла для других программ обработки |
| *long* ***GetGLDataToFile*** *(BSTR FileName)* | Взятие данных из файла и отображение в графике. Файл должен быть создан при помощи метода SaveGLDataToFile(…) |
| *float* ***GetPointAt****(LONG X, LONG Y)* | получить позицию курсора |
| **Передача данных в Gramma.ocx** | |
| *long* ***Paint****(float * buffer)* | передает во внутренний буфер очередной строки изображения |
| *long* ***PaintNet****(long pBuffer)* | Передает во внутренний буфер очередной строки изображения, используется при работе в .net языках |
| *void* ***AltC****(float * buffer)* | позволяет загружает массив разметки данных по X. Этот метод работает при установленном свойстве типа сетки TypeAbs = 2 |
| *void* ***AltCNet****(long pBuffer)* | загружает массив разметки данных по оси X. Этот метод работает при установленном свойстве типа сетки TypeAbs = 2, используется при работе в .net языках |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *void* ***Formatxy****(LPCTSTR fmt)* | содержит надпись в графике |
| *long* ***PaintDouble****(double* buffer)* | метод, позволяет передать данные для отображения графика и перерисовать график. |
| *void* ***AItCDouble****(double* buffer)* | метод, позволяет загрузить массив разметки данных по X. |
| **События от компоненты Gramma.ocx** | |
| *void* ***KeyDown****(SHORT* KeyCode, SHORT Shift)* | событие возникает при нажатии на клавишу клавиатуры, когда фокус находится на компоненте |
| *void* ***MouseDown****(short Button, short Shift, OLE_XPOS_PIXELS x, OLE_YPOS_PIXELS y)* | событие возникает при нажатии на кнопку манипулятора типа «мышь», при нахождении мыши над графиком |
| *void* ***MouseUp****(short Button, short Shift, OLE_XPOS_PIXELS x, OLE_YPOS_PIXELS y)* | событие возникает при отжатии кнопки манипулятора типа «мышь», при нахождении мыши над графиком |
| *void* ***MouseWheel****(void)* | событие возникает при вращении ролика манипулятора типа «мышь» |

# Глава 6.GrammaGL.ocx предназначена для графического отображения массивов данных (улучшенная)

.Компонента **GrammaGL.ocx** предназначена для графического отображения массивов данных в виде графиков зависимостей $z = z(x, y)$. Процедура рисования графиков написана таким образом, что отсутствует мерцание при перерисовке графиков.

## 6.1.Установка компонента GrammaGL.ocx

Для работы с модулем GrammaGL его необходимо сначала установить в программу. При загрузке программы необходимо загрузить с помощью компонента необходимую программу, разместить окно программы на экране в удобном месте, установить необходимые параметры в программе и затем по событиям, происходящим от программы считывать данные из программы.

Для установки компонента GrammaGL в проект *Microsoft Visual Studio 2010*, необходимо кликнуть правой кнопкой мыши на форме диалога и из появившегося меню выбрить пункт *Insert ActiveX Control*...(рисунок 1.1)

© ООО "ЭТМС" 1992-2024. Все права защищены

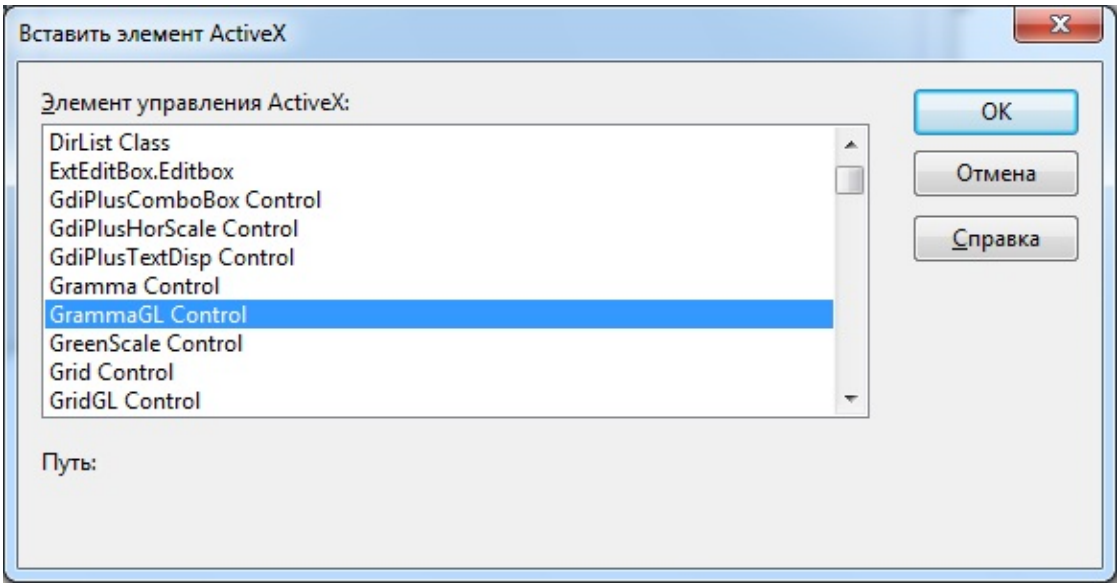



*Рисунок 1.1*

Из появившегося диалогового окна следует выбрать компонент GrammaGL *Control* и нажать *OK* (рисунок 1.2). После этого компонент *GrammaGL.ocx* появится на форме диалога (рисунок 1.2).

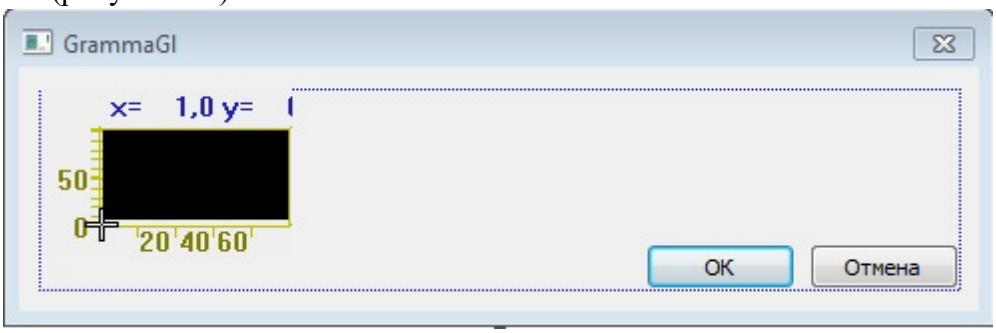



Рисунок 1.2

Одна установленная на форме компонента позволяет управлять одной программой. При необходимости управлять несколькими программами, необходимо установить на форму несколько компонент. Для того чтобы компонента не была видна на форме во время выполнения программы, необходимо установить свойство *Visible* равным *False*.

## 6.2.Назначение GrammaGL.ocx

Компонент позволяет:

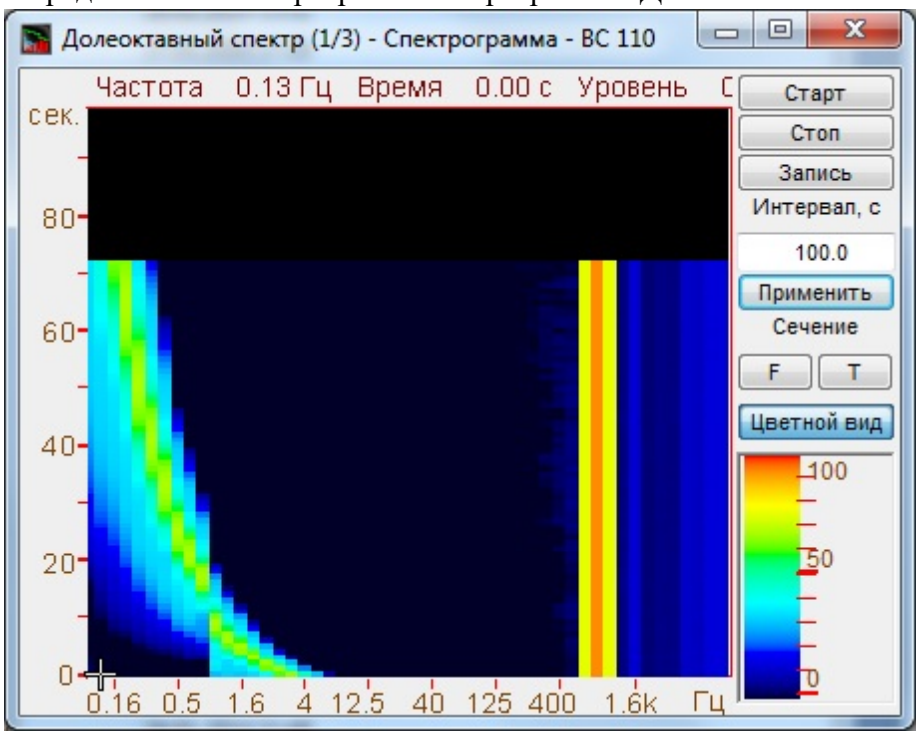
- масштабировать график;
- отображать данные в трехмерном и двумерном виде;

© ООО "ЭТМС" 1992-2024. Все права защищены

- отображать графики в цветном и черно-белом виде;
- перемещать курсор при помощи кнопки мыши, ролика мыши и клавиатуры;
- показывать значения абсцисс и ординат на позиции курсора;
- сохранять данные в буфере обмена (clipboard) в графическом, текстовом и численном виде для создания отчетов в редакторах Microsoft Word и Excel.

Применяется, например, в программах: "**Просмотр результатов**", "**Узкополосный спектр**", "**Взаимный узкополосный спектр**", "**Долеоктавный спектр**", "**Взаимный долеоктавный спектр**", "**Взаимный корреляционный анализ**", "**Взаимный долеоктавный спектр**", "**Спектр со сверхразрешением**", "**Вейвлет анализ**", "**Синхронное накопление**".

На рисунках представлена спектрограмма из программы "**Долеоктавный спектр**".




Спектрограмма

## 6.3.Полное описание свойств, методов и событий GrammaGL.ocx

Следует отметить, что компонент автомасштабируется при изменении его размеров, поэтому необходимо следить за тем, чтобы высота компонента в приложении была не менее 80 пикселей, иначе он может отображаться некорректно.

© ООО "ЭТМС" 1992-2024. Все права защищены

## Свойства GrammaGL.ocx

Свойства можно задавать на этапе проектирования и во время выполнения программы. На этапе проектирования при выборе компонента и при нажатии клавиши <F4> появляется окно свойств. Эти свойства можно устанавливать на этапе проектирования программы.

График задается в виде двумерного массива. В компоненте задаются размеры по координате X и Y при помощи свойств Size и Number. Данные передаются в компонент массивом строки размером Size с помощью метода Paint(). Номер строки задается свойством Ind. После пересылки каждой строки прорисовывается график.

Цвета задаются при помощи макроса *RGB(RED, GREEN, BLUE)*. Значения *RED*, *GREEN*, *BLUE* задают интенсивности цветов красного, зеленого и синего. Интенсивности изменяются в пределах от 0 до 255.

*OLE_COLOR* ***ClrCrs*** – цвет курсора.

*OLE_COLOR* ***ClrDig*** – цвет цифр.

*OLE_COLOR* ***ClrFon*** – цвет фона.

*OLE_COLOR* ***ClrGrd*** – цвет -сетки-меток.

*OLE_COLOR* ***ClrGrf*** – цвет выбранного графика. Для каждого графика цвет задается отдельно. Графики выбираются при помощи свойства *Ind*.

*OLE_COLOR* ***ClrLeg*** – цвет легенды.

*long* ***Black*** – цветное или черно-белое изображение.
0 – цветное изображение.
1 – черно-белое изображение.

Например, для изменении фона графика с желтого на зеленый, могут быть записаны числа:

```
OLE_COLOR a = 0xFF00;
OLE_COLOR b = 0xFFFF;
```

*long* ***Size*** – Количество точек по горизонтали. Размер не может быть меньше или равен 0. Размер не может быть больше 50000.

© ООО "ЭТМС" 1992-2024. Все права защищены

*long* ***Number*** – количество строк в изображении. Количество строк не может превышать 200.

*long* ***Ind*** – индекс (номер) строки. Индекс может принимать значения от 0 до Number-1.

*DOUBLE* ***DeltaT*** – *DOUBLE* ***DeltaT*** – шаг по времени по оси Y. Например, при отображении сонограммы отображаются последовательность спектров получаемых каждые 0.1 с и тогда *DeltaT* = 0.1.

*long* ***Log*** – логарифмическая шкала по Y. Параметр может принимать значения 0 или 1.

0 – линейный масштаб.
1 – логарифмический масштаб.

*long* ***Px*** – позиция курсора по оси X. Этот параметр принимает значение от 0 до *Size*-1. Параметр можно читать и записывать. При перемещении курсора при помощи мыши или клавиатуры этот параметр отслеживает положение курсора.

*long* ***Py*** – позиция курсора по оси Y. Этот параметр принимает значение от 0 до *Number*-1. Параметр можно читать и записывать. При перемещении курсора при помощи мыши или клавиатуры этот параметр отслеживает положение курсора.

*BSTR* ***FontName*** – название шрифта для надписей.

*long* ***FontSize*** – размер шрифта надписей.

*long* ***CommentInd*** – порядковый номер комментария для записи в файл.

*BSTR* ***CommentSTR*** – строка комментария для записи в файл.

*DOUBLE* ***Reference*** – Опорное значение для расчета уровня в дБ.
Уровень в дБ равен:

$$L_{dB} = 20\log\frac{U}{R}$$

R - значение Reference, U – значение сигнала в физических единицах измерения, например в Вольтах. $L_{dB}$ - уровень сигнала в дБ.

*FLOAT* ***ColorMin -*** минимальный уровень по цвету.

*FLOAT* ***ColorWidth***- диапазон уровня по цвету.

*LONG* ***bwithPicture*** – минимальный уровень по цвету (0-запретить, 1-разрешить)

*BSTR* ***sPictureName*** - загружается имя файла текстуры

*long* ***TypeXAxis*** - тип отображаемых данных по X (для последующих расширений):

© ООО "ЭТМС" 1992-2024. Все права защищены

100 - автомасштаб

*long* ***TypeYAxis*** – тип отображаемых данных по Y (для последующих расширений)

*long* ***TypeYData*** - тип отображаемых данных по Y (для последующих расширений).

*BSTR* ***TextXAxis*** - единица измерения по оси X.

*BSTR* ***TextYAxis*** - единица измерения по оси Y.

*long* ***AltCoords*** – данные по равномерным и неравномерным координатам по X.

0 – равномерные.
Данные для отображения по координатам X и Y представляются в виде:
*X={Xfirst…Xend};*
*Y={buffer[0]…buffer[Size-1]};*
Массив данных buffer для отображения по Y передается методом Paint(float* buffer).
При отображении координаты по оси считаются равномерными

$$x_i = (Xend - Xfirst) / Size \times i$$
$$y_i = buffer[i]$$
$$i = 0...Size - 1$$

1 – произвольные координаты, координаты X задаются в методе AltC(float* cord) в массиве cord. Шкала становится 1/n октавной при любом TypeXAxis.
Данные для отображения по координатам X и Y представляются в виде:
*X={cord[0]…cord[Size-1]};*
*Y={buffer[0]…buffer[Size-1]};*
Массив данных buffer для отображения по Y передается методом Paint(float* buffer).
При отображении координаты по оси X считаются произвольными

$$x_i = cord[i]$$
$$y_i = buffer[i]$$
$$i = 0...Size - 1$$

Для каждого графика толщина линии задается индивидуально. Номер графика выбирается свойством Ind.

*VARIANT_BOOL* ***OrdinateData*** – использовать по оси ординат дату и время.

© ООО "ЭТМС" 1992-2024. Все права защищены

*long* ***MakeUpor*** – включение ограничений по оси *Z*. При масштабировании графиков при помощи клавиатуры и мыши на стадии выполнения программы возможна ситуация, когда происходит переполнение данных. Для включения ограничений по масштабированию необходимо установить параметр *MakeUpor* в 1. Для снятия ограничений в 0.

*DOUBLE* ***DeltaT -*** шаг по времени (по вертикали)

*DOUBLE* ***MathLX*** – область отображения данных по оси *X*. Левая граница отображения данных.

*DOUBLE* ***MathDX***– область отображения данных по оси *X*. Длина отображения данных.

Например, необходимо отобразить часть данных из предыдущего примера в диапазоне от 1050 *Гц* до 1070 *Гц*. В этом случае задаются следующие параметры:

*MathLX* = 1050.

*MathDX* = 20.

При работе программы в режиме исполнения пользователь может менять область отображения графиков при помощи курсора мыши. Растяжение или сжатие графиков происходит при помощи указателя вида:  ,  – для горизонтальной оси. Сдвинуть графики вправо-влево можно при помощи указателя  ,  – для горизонтальной оси. При этом меняются параметры *MathLX, MathDX.* В пользовательской программе в режиме исполнения можно прочитать текущие параметры области отображения.

*DOUBLE* ***MathLY*** – начало отображения данных по уровню. Левая граница отображения данных.

*DOUBLE* ***MathDY***– диапазон отображения по уровню. Длина отображения данных.

Растяжение или сжатие графиков происходит при помощи указателя вида:  ,  – для вертикальной оси. Сдвинуть графики вверх или вниз можно при помощи указателя  ,  – для вертикальной оси. При этом меняются параметры *MathLY, MathDY.* В пользовательской программе в режиме исполнения можно прочитать текущие параметры области отображения. При включенных ограничениях отображения

© ООО "ЭТМС" 1992-2024. Все права защищены

графиков *MakeUpor* = 1 , параметр *MathDY* не может быть меньше *UporNis* , сумма параметров *MathLY* + *MathDY* не может быть больше *UporVerh*.

*DOUBLE* ***Xfirst*** – начальное значение по координате *X*.

*DOUBLE* ***Xend*** – конечное значение по координате *X*.

Например, необходимо отобразить массив из 100 чисел в равномерной сетке по оси *X*. Числа представляют собой спектр мощности сигнала в диапазоне от 1000 Гц до 1100 Гц. Т.е. 0-ому элементу массива соответствует частота 1000 Гц, 99-му – частота 1100 Гц. В этом случае задаются следующие параметры:

Xfirst = 1000.

*Xend* = 1100.

*double* ***UporNis*** – минимальная величина по координате *Y*, которая может отобразиться на графике. Это свойство работает при включенных ограничениях по оси *Y*.

*double* ***UporVerh*** – максимальная величина по координате *Y*, которая может отображаться на графике. Это свойство работает при включенных ограничениях по оси *Y*.

**Пример**. Отображение временной реализации оцифрованного АЦП сигнала в виде осциллограммы. Входной диапазон работы АЦП от -10 В до + 10 В, вес младшего разряда 16-разрядного АЦП 0.0003 В. В этом случае можно задать следующие параметры.

MakeUpor = 1

*UporNis* = 0.0003

*UporVerh* = 10.0

*DOUBLE* ***MathCursorX*** *– положение курсора по X*

## Методы GrammaGL.ocx

Методы можно использовать только во время выполнения программы.

*void* ***AboutBox()*** –содержит информацию о программе.

*float* ***GetPointAt****(long X, long Y)* – метод позволяет получить позицию курсора.

© ООО "ЭТМС" 1992-2024. Все права защищены

*void* ***Reset*** *(void)* – сброс к первоначальному состоянию.

*long* ***SaveGLDataToFile****(BSTR FileName)* – метод, позволяет сохранить массив данных в файле. Сохранение данных в текстовый файл с расширением *.dtu". Файл записывается в текстовом виде со всеми комментариями. Это позволяет использовать данные из этого файла для других программ обработки. *FileName* – строка полного пути и имени файла.

*long* ***GetGLDataFromFile****(BSTR FileName)* – метод позволяет взять данные из файла и отобразить в графики. Файл должен быть создан при помощи метода *SaveGLDataToFile(...).*

В файл создаваемый методом *SaveGLDataToFile(...)* можно записывать 8 строк дополнительной информации. Для записи дополнительной информации необходимо установить номер индекса комментария от 0 до 7 и записать строку комментария.

*short* ***Display*** *(void )* – метод перерисовывает изображения - фон, сетку, надписи и все графики.

*long* ***PaintNotDraw*** *(FLOAT* buffer)* – передает во внутренний буфер очередной строки изображения, но не перерисовывает.

*long* ***PaintAll*** *(FLOAT* buffer, LONG xsize, LONG ysize)* - отрисовываем картинку с заданным размером

*long* ***GetGrammaLine****(LONG line, FLOAT* buffer)* - получение буфера с заданной линией.

*long* ***GetMassivePointer****(LONG i, LONG j)* - получение указателя на строку в двумерном массиве.

*long* ***SetData****(DOUBLE data)* - установить текущую дату и время для очередной строки изображения

*long* ***GetPointerOfLineData****()* - получаем указатель на буфер времён в формате DATE.

*long* ***GetGrammaColumn****(LONG column, FLOAT* buffer)* - получаем буфер с заданным столбцом.

void ***DrawPicture***(*ULONG hdc, INT x, INT y, INT width, INT height*) - отрисовывает картинку графика на указанном HDC.

*long* ***PaintLine***(*LONG indexLine, FLOAT* buffer*) - прорисовка одной заданной строки на графике

© ООО "ЭТМС" 1992-2024. Все права защищены

*long* ***GetQuantityUsedLines()*** – передаёт количество заданных строк

*long* ***GetPointerLineMassive(****long iLine****)*** – *п*ередаёт указатель на данные заданной строки

*long* ***SetDataNext()*** – присваивание очередной строке следующего времени, альтернатива функции SetData

*long* ***PaintNaN()*** – задаёт очередную строку значениями NaN

*long* ***Paint*** *(float* Buffer)* – метод для передачи во внутренний буфер очередной строки изображения. При построении необходимо при помощи свойства *Ind* выбирать, номер строки для перерисовки.

***Параметры****:*

*float* Buffer* – адрес массива данных в программе пользователя. В компоненту копируется количество данных определенное свойством *Size*.

*long* ***PaintNet*** *(long pBuffer)* – метод для передачи во внутренний буфер очередной строки изображения. При построении необходимо при помощи свойства *Ind* выбирать, номер строки для перерисовки

.

Параметры:

*long pBuffer* – адрес числового массива данных в программе пользователя. В компоненту копируется количество данных определенное свойством *Size*.

При создании программ обработки сигналов в реальном времени с многократной перерисовкой графиков, типа осциллографов или анализаторов спектра, необходимо один раз задать параметры компонента, и затем последовательно обращаться только к методу *Paint(...)* для создания эффекта движения.

*void* ***AltC*** *(float *buffer)* – метод, позволяет загрузить массив разметки данных по *X*. Этот метод работает при установленном свойстве типа сетки *TypeAbs* = 2. В пользовательском буфере *buffer* размера *Size* необходимо записать значения координаты *X*. Например, при логарифмической развертке по оси *X*, в буфере могут быть записаны числа:

*buffer[0] = 1.*
*buffer[1] = 1.25*
*buffer[2] = 1.6*
*buffer[3] = 2.*
*buffer[4] = 2.5*

*void* ***AltCNet*** *(long pBuffer)* – метод, позволяет задать значения по оси *X*. Этот метод работает при установленном свойстве типа сетки *TypeAbs* = 2. В

© ООО "ЭТМС" 1992-2024. Все права защищены

пользовательском буфере *buffer* размера *Size* необходимо записать значения координаты *X*. Например, при логарифмической развертке по оси *X*, в буфере могут быть записаны числа:

*buffer[0] = 1.*
*buffer[1] = 1.25*
*buffer[2] = 1.6*
*buffer[3] = 2.*
*buffer[4] = 2.5*

*LONG GetMassivePointer(LONG i, LONG j)* - метод возвращающий указатель на строку двумерного массива.

**Примечание:** Следует отметить, то при работе в .net языках функции, в которых в качестве параметров выступают указатели на массив, например GetData в .net языках работают некорректно, поэтому следует использовать обновленные функции, имеющие такое же название + приписка Net, например, GetDataNet.

*void* ***Formatxy*** *(BSTR fmt)* – метод передает в компонент строку для форматного вывода информации по координатам *X, Y* и *Z* в легенде графика. В соответствии с заданным форматом отображается положение и уровень Форматный ввод-вывод позволяет передавать программе и выдавать из программы на внешнее устройство символьные, строковые и числовые значения, обеспечивая преобразование данных в процессе ввода-вывода.

Например, обращение вида:

FormatX(“Частота %7.2fГц Время %7.2fс Уровень %7.3fВ”)

позволяет отображать координаты по оси X,Y и Z в значениях по частоте.

Формат представления чисел.

%f – общий формат представления числа в плавающей запятой в виде числа с точкой

%e – общий формат представления числа в плавающей запятой в виде числа с экспонентой (1000 = 1e3, 0.001 = 1e-3)

%7.2f – представление числа в фиксированном формате. В данном примере 7 – количество знакомест для представления числа, 2 – количество цифр после запятой в представлении числа.

*long* ***GetFormatXY*** *(BSTR fmt)* – метод берет из компонента строку для форматного вывода информации по координатам *X, Y* и *Z* в легенде графика. В соответствии с заданным форматом отображается положение и уровень Форматный ввод-вывод позволяет передавать программе и выдавать из программы на внешнее устройство символьные, строковые и числовые значения, обеспечивая преобразование данных в процессе ввода-вывода.

Например, обращение вида:

FormatX(“Частота %7.2fГц Время %7.2fс Уровень %7.3fВ”)

позволяет отображать координаты по оси X,Y и Z в значениях по частоте.

© ООО "ЭТМС" 1992-2024. Все права защищены

Формат представления чисел.

%f – общий формат представления числа в плавающей запятой в виде числа с точкой

%e – общий формат представления числа в плавающей запятой в виде числа с экспонентой (1000 = 1e3, 0.001 = 1e-3)

%7.2f – представление числа в фиксированном формате. В данном примере 7 – количество знакомест для представления числа, 2 – количество цифр после запятой в представлении числа.

HRESULT ***GetSaveFormat***([out, retval] BSTR* pVal) - позволяет получить формат для сохранения графика.

*long* ***PaintDouble***(DOUBLE* buffer) - позволяет передавать во внутренний буфер очередной строки изображения.

void ***AltCDouble***(double* buffer) - метод, позволяет загрузить массив разметки данных по *X*.

Этот метод работает при установленном свойстве типа сетки *TypeAbs* = 2. В пользовательском буфере *buffer* размера *Size* необходимо записать значения координаты *X*. Например, при логарифмической развертке по оси *X*, в буфере могут быть записаны числа:

*buffer[0] = 1.*
*buffer[1] = 1.25*
*buffer[2] = 1.6*
*buffer[3] = 2.*
*buffer[4] = 2.5*

## События от компоненты GrammaGL.ocx

*void* ***KeyDown*** *(SHORT* KeyCode, SHORT Shift)* – событие возникает при нажатии на клавишу клавиатуры, когда фокус находится на компоненте.

Параметры:
*KeyCode* – код клавиши
*Shift* – код нажатия служебных клавиш *<Ctrl>,<Shift>,<Alt>*.

**Примечание управляющим клавишам:**

<C> - Копирование графика. Для копирования графика нажать левой кнопкой «мыши» на поле графика и нажать комбинацию «горячих клавиш» клавиатуры <Ctrl> +<C>.

График запишется в буфер обмена(Clipboard).

<A> Управление движением курсора влево. Нужно нажать левой кнопкой «мыши» на поле графика программы и нажать на кнопку клавиатуры <A>(в латинской раскладке клавиатуры).

<D> Управление движением курсора вправо. Нужно нажать левой кнопкой «мыши» на поле графика программы и нажать на кнопку клавиатуры <D>(в латинской раскладке клавиатуры).

<W> Управление движением курсором вверх. Нужно нажать левой кнопкой «мыши» на поле графика программы и нажать на кнопку клавиатуры <W>(в латинской раскладке клавиатуры).

© ООО "ЭТМС" 1992-2024. Все права защищены

<S> Управление движением курсором вниз. Нужно нажать левой кнопкой «мыши» на поле графика программы и нажать на кнопку клавиатуры <S>(в латинской раскладке клавиатуры).

*void* ***MouseDown*** *(short Button, short Shift, OLE_XPOS_PIXELS x, OLE_YPOS_PIXELS y)* – событие возникает при нажатии на кнопку манипулятора типа «мышь», при нахождении мыши над графиком.

Параметры:
*Button* – код кнопки мыши
*Shift* – код нажатия служебных клавиш *<Ctrl>,<Shift>,<Alt>*.
*x* – координата по оси *X* в пикселах (разрешении экрана)
*y* – координата по оси *Y* в пикселах (разрешении экрана)

*void* ***MouseUp*** *(short Button, short Shift, OLE_XPOS_PIXELS x, OLE_YPOS_PIXELS y)* – событие возникает при отжатии кнопки манипулятора типа «мышь», при нахождении мыши над графиком.

Параметры:
*Button* – код кнопки мыши
*Shift* – код нажатия служебных клавиш *<Ctrl>,<Shift>,<Alt>*.
*x* – координата по оси *X* в пикселях (разрешении экрана)
*y* – координата по оси *Y* в пикселях (разрешении экрана)

v*oid* ***MouseWheel*** *(void)* – событие возникает при вращении ролика манипулятора типа «мышь».

## 6.4.Краткий перечень функций, используемые GrammaGL.ocx

| **Свойства GrammaGL.ocx** | |
|---|---|
| **Цвета** | |
| *long* ***GetClrDig****()* | опрашивается цвет цифр |
| *void* ***SetClrDig****(unsigned long propVal)* | устанавливается цвета цифр |
| *long* ***GetClrFon*** *()* | опрашивается цвет фона |
| *void* ***SetClrFon*** *(unsigned long propVal)* | устанавливается цвет фона |
| *long* ***GetClrGrd*** *()* | опрашивается цвет сетки-меток |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *void* ***SetClrGrd*** *(unsigned long propVal)* | устанавливается цвет сетки-меток |
| *long* ***GetClrLeg*** *()* | опрашивается цвет легенды графика |
| *void* ***SetClrLeg*** *(unsigned long propVal)* | устанавливается цвет легенды графика |
| *long* ***GetBack*** *()* | опрашивается цветное или черно-белое изображение (0 – цветное /1 – черно-белое) |
| *void* ***SetBack*** *(unsigned long propVal)* | устанавливается цветное или черно-белое изображение (0 – цветное /1 – черно-белое) |
| **Параметры графиков** | |
| *long* ***GetSize*** *()* | опрашивается количество точек по горизонтали. Размер не может быть меньше или равен 0. Размер не может быть больше 50000 |
| *void* ***SetSize*** *(long propVal)* | устанавливается количество точек по горизонтали. Размер не может быть меньше или равен 0. Размер не может быть больше 50000 |
| *long* ***GetNumber****()* | опрашивается количество строк в изображении. Количество строк не может превышать 200 |
| *void* ***SetNumber****(long propVal)* | устанавливается количество строк в изображении. Количество строк не может превышать 200 |
| *double* ***GetDeltaT*** *()* | опрашивается шаг по времени (по вертикали) |
| *void* ***SetDeltaT*** *(float propVal)* | устанавливается шаг по времени (по вертикали) |
| *long* ***GetPx****()* | опрашивается позиция курсора по оси X. Этот параметр принимает значение от 0 до Size-1 |
| *void* ***SetPx*** *(long propVal)* | устанавливается позиция курсора по оси X. Этот параметр принимает значение от 0 до Size-1 |
| *long* ***GetPy****()(* | опрашивается позиция курсора по оси Y. Этот параметр принимает значение от 0 до Size-1 |
| *long* ***SetPy****()(long propVal)* | Устанавливается позиция курсора по оси Y. Этот параметр принимает значение от 0 до Size-1 |
| *long* ***GetFontName*** *()* | опрашивается название шрифта для надписей |
| *void* ***Set FontName*** *(long propVal)* | устанавливается название шрифта для надписей |
| *long* ***GetFontSize*** *()* | опрашивается размер шрифта надписей |
| *void* ***SetFontSize*** *(long propVal)* | устанавливается размер шрифта надписей |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *long* ***GetCommentInd ()*** | опрашивается порядковый номер комментария для записи в файл |
| *void* ***SetCommentInd*** *(long propVal)* | устанавливается порядковый номер комментария для записи в файл |
| *CString* ***GetCommentSTR****()* | опрашивается строка комментария для записи в файл |
| *void* ***SetCommentSTR****(CString propVal)* | устанавливается строка комментария для записи в файл |
| *double* ***GetReference*** *()* | опрашивается опорное значение уровня для расчета в дБ |
| *void* ***SetReference*** *(float propVal)* | устанавливается опорное значение уровня для расчета в дБ |
| *float* ***GetColorMin****()* | опрашивается минимальный уровень по цвету |
| *void* ***SetColorMin****(float propVal)* | устанавливается минимальный уровень по цвету |
| *float* ***GetColorWidth*** *()* | опрашивается диапазон уровня по цвету |
| *void* ***SetColorWidth*** *(float propVal)* | устанавливается диапазон уровня по цвету |
| *long* ***GetbwithPicture*** *()* | опрашивается разрешения или запрещения текстуры (0-запретить, 1-разрешить) |
| *void* ***SetbwithPicture*** *(long propVal)* | устанавливается минимальный уровень по цвету (0-запретить, 1-разрешить) |
| *CString* ***GetsPictureName****()* | опрашивается загружается имя файла текстуры |
| *void* ***SetsPictureName****(CString propVal)* | устанавливается загружается имя файла текстуры |
| **Параметры шкалы** | |
| *long* ***GetTypeAbs****()* | опрашивается тип отображаемых данных по X (для последующих расширений) |
| *void* ***SetTypeAbs****(long propVal)* | устанавливается тип отображаемых данных по X (для последующих расширений) |
| *long* ***GetTypeXAxis****()* | опрашивается тип отображаемых данных по X(для последующих расширений) (100-автомасштаб) |
| *void* ***SetTypeXAxis****(long propVal)* | устанавливается тип отображаемых данных по X(для последующих расширений) (100-автомасштаб) |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *long* ***GetTypeYAxis****()* | опрашивается тип отображаемых данных по Y (для последующих расширений) |
| *void* ***SetTypeYAxis****(long propVal)* | устанавливается тип отображаемых данных по Y (для последующих расширений) |
| *long* ***GetTypeYData****()* | опрашивается тип отображаемых данных по Y (для последующих расширений) |
| *void* ***SetTypeYData****(long propVal)* | устанавливается тип отображаемых данных по Y (для последующих расширений) |
| *CString* ***GetTextXAxis****()* | опрашивается единица измерения по оси X |
| *void* ***SetTextXAxis****(CString propVal)* | устанавливается единица измерения по оси X |
| *CString* ***GetTextYAxis****()* | опрашивается единица измерения по оси Y |
| *void* ***SetTextYAxis****(CString propVal)* | устанавливается единица измерения по оси Y |
| *long* ***GetAltCoords****()* | опрашивается равномерные и неравномерные координаты по X\n0-равномерные\n1-координаты задаются в методе AltC(float* buffer) |
| *void* ***SetAltCoords****(long propVal)* | устанавливается равномерные и неравномерные координаты по X\n0-равномерные\n1-координаты задаются в методе AltC(float* buffer) |
| *long* ***GetOrdinateData****()* | опрашивается использовать по оси ординат дату и время\n0-Да\n1-Нет. |
| *void* ***SetOrdinateData****(VARIANT_BOOL propVal)* | устанавливается использовать по оси ординат дату и время\n0-Да\n1-Нет. |
| **Представление графиков** | |
| *long* ***GetMakeUpor****()* | опрашивается включение ограничений по оси Z. При масштабировании графиков при помощи клавиатуры и мыши на стадии выполнения программы возможна ситуация, когда происходит переполнение данных. Для включения ограничений по масштабированию необходимо установить параметр MakeUpor в 1. Для снятия ограничений в 0 |
| *void* ***SetMakeUpor****(long propVal)* | устанавливается включение ограничений по оси Z. При масштабировании графиков при помощи клавиатуры и мыши на стадии выполнения программы возможна ситуация, когда происходит переполнение данных. Для включения ограничений |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| | по масштабированию необходимо установить параметр MakeUpor в 1. Для снятия ограничений в 0 |
| *double* ***GetMathlx****()* | опрашивается минимальное значение по X. Левая граница отображения данных |
| *void* ***SetMathlx****(double propVal)* | устанавливается минимальное значение по X. Левая граница отображения данных |
| *double* ***GetMathdx****()* | опрашивается диапазон отображения по X. Длина отображения данных |
| *void* ***SetMathdx****(double propVal)* | устанавливается диапазон отображения по X. Длина отображения данных |
| *double* ***GetMathly****()* | опрашивается минимальное значение по Y. Нижняя граница отображения данных |
| *void* ***SetMathly****(double propVal)* | Устанавливается минимальное значение по Y. Нижняя граница отображения данных |
| *double* ***GetMathdy****()* | опрашивается диапазон отображения по Y. Высота отображения данных |
| *void* ***SetMathdy****(double propVal)* | устанавливается диапазон отображения по Y. Высота отображения данных |
| *double* ***GetXend*** *()* | опрашивается конечное значение по координате X |
| *void* ***SetXend*** *(double propVal)* | устанавливается конечное значение по координате X |
| *double* ***GetXfirst****()* | опрашивается начальное значение по координате X |
| *void* ***SetXfirst*** *(double propVal)* | устанавливается начальное значение по координате X |
| *double* ***GetUporNis*** *()* | опрашивается нижний допустимый диапазон отображения по Y (при включенном ограничении) |
| *void* ***SetUporNis*** *(float propVal)* | устанавливается нижний допустимый диапазон отображения по Y (при включенном ограничении) |
| *double* ***GetUporVerh*** *()* | опрашивается верхний допустимый диапазон отображения по Y (при включенном ограничении) |
| *void* ***SetUporVerh*** *(float propVal)* | устанавливается верхний допустимый диапазон отображения по Y (при включенном ограничении) |
| *double* ***MathCursorX*** *()* | опрашивается положение курсора по X |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Методы GrammaGL.ocx | |
|---|---|
| **Настройка GrammaGL.ocx** | |
| *void **AboutBox**()* | содержит информацию о программе |
| *short **Display** (void)* | перерисовывает изображения |
| *void **Reset** (void)* | сброс к первоначальному состоянию |
| *long (BSTR FileName)* | позволяет сохранить массив данных в файле. Сохранение данных в текстовый файл с расширением *.dtu". Файл записывается в текстовом виде со всеми комментариями. Это позволяет использовать данные из этого файла для других программ обработки |
| *long **GetGLDataToFile** (BSTR FileName)* | Взятие данных из файла и отображение в графике. Файл должен быть создан при помощи метода SaveGLDataToFile(…) |
| *float **GetPointAt**(LONG X, LONG Y)* | получить позицию курсора |
| *long **PaintNotDraw** (FLOAT* buffer)* | передает во внутренний буфер очередной строки изображения, но не перерисовывает |
| *long **GetPaintAll** (FLOAT* buffer, LONG xsize, LONG ysize)* | отрисовываем картинку с заданным размером |
| *long **GetGrammaLine**(LONG line, FLOAT* buffer)* | Получение буфера с заданной линией |
| *long **GetMassivePointer**(LONG i, LONG j)* | Получение указателя на строку в двумерном массиве. |
| *long **GetMassivePointer**(LONG i, LONG j)* | Возвращающий указатель на строку двумерного массива |
| *long **SetData**(DOUBLE data);* | Установить текущую дату и время для очередной строки изображения |
| *long **GetPointerOfLineData**()* | Получаем указатель на буфер времён в формате DATE. |
| *long **GetGrammaColumn**(LONG column, FLOAT* buffer)* | Получаем буфер с заданным столбцом |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *void **DrawPicture**(ULONG hdc, INT x, INT y, INT width, INT height)* | Отрисовывает картинку графика на указанном HDC |
| *long **PaintLine***(LONG indexLine, FLOAT* buffer) | Прорисовка одной заданной строки на графике. |
| *long **GetQuantityUsedLines**()* | передаёт количество заданных строк |
| *long **GetPointerLineMassive**(long iLine)* | передаёт указатель на данные заданной строки |
| *long **SetDataNext()*** | присваивание очередной строке следующего времени, альтернатива функции SetData |
| *long **PaintNaN()*** | задаёт очередную строку значениями NaN |
| **Передача данных в GrammaGL.ocx** | |
| *long **Paint**(float * buffer)* | Передает во внутренний буфер очередной строки изображения |
| *long **PaintNet**(long pBuffer)* | Передает во внутренний буфер очередной строки изображения, используется при работе в .net языках |
| *void **AltC**(float * buffer)* | позволяет загружает массив разметки данных по X. Этот метод работает при установленном свойстве типа сетки TypeAbs = 2 |
| *void **AltCNet**(long pBuffer)* | загружает массив разметки данных по оси X. Этот метод работает при установленном свойстве типа сетки TypeAbs = 2, используется при работе в .net языках |
| *void **Formatxy**(LPCTSTR fmt)* | содержит строку для вывода данных по осям X, Y, Z |
| *void **GetFormatXY** (LPCTSTR fmt)* | взятие строки для вывода данных по осям X, Y, Z |
| *HRESULT **GetSaveFormat**([out, retval] BSTR* pVal)* | позволяет получить формат для сохранения графика. |
| *long **PaintDouble**(DOUBLE* buffer)* | позволяет передавать во внутренний буфер очередной строки изображения. |
| *void **AltCDouble**(double* buffer)* | метод, позволяет загрузить массив разметки данных по X. |

© ООО "ЭТМС" 1992-2024. Все права защищены

| **События от компоненты GrammaGL.ocx** | |
|---|---|
| *void **KeyDown**(SHORT* KeyCode, SHORT Shift)* | событие возникает при нажатии на клавишу клавиатуры, когда фокус находится на компоненте |
| *void **MouseDown**(short Button, short Shift, OLE_XPOS_PIXELS x, OLE_YPOS_PIXELS y)* | событие возникает при нажатии на кнопку манипулятора типа «мышь», при нахождении мыши над графиком |
| *void **MouseUp**(short Button, short Shift, OLE_XPOS_PIXELS x, OLE_YPOS_PIXELS y)* | событие возникает при отжатии кнопки манипулятора типа «мышь», при нахождении мыши над графиком |
| *void **MouseWheel**(void)* | событие возникает при вращении ролика манипулятора типа «мышь» |

© ООО "ЭТМС" 1992-2024. Все права защищены

# Описание программных модулей кнопок

## Глава 1.ZETButtonOCX.ocx предназначена для кнопок ZETLab в меню

Компонента **ZETButtonOCX.ocx** предназначена для кнопок в программах ZETLab, а также в меню.

### 1.1.Установка компонента ZETButtonOCX

Для работы с модулем ZETButtonOCX его необходимо сначала установить в программу. При загрузке программы необходимо загрузить с помощью компонента необходимую программу, разместить окно программы на экране в удобном месте, установить необходимые параметры в программе и затем по событиям, происходящим от программы считывать данные из программы.

Для установки компонента ZETButtonOCX в проект *Microsoft Visual Studio 2010*, необходимо кликнуть правой кнопкой мыши на форме диалога и из появившегося меню выбрить пункт *Insert ActiveX Control*...(рисунок 1.1)

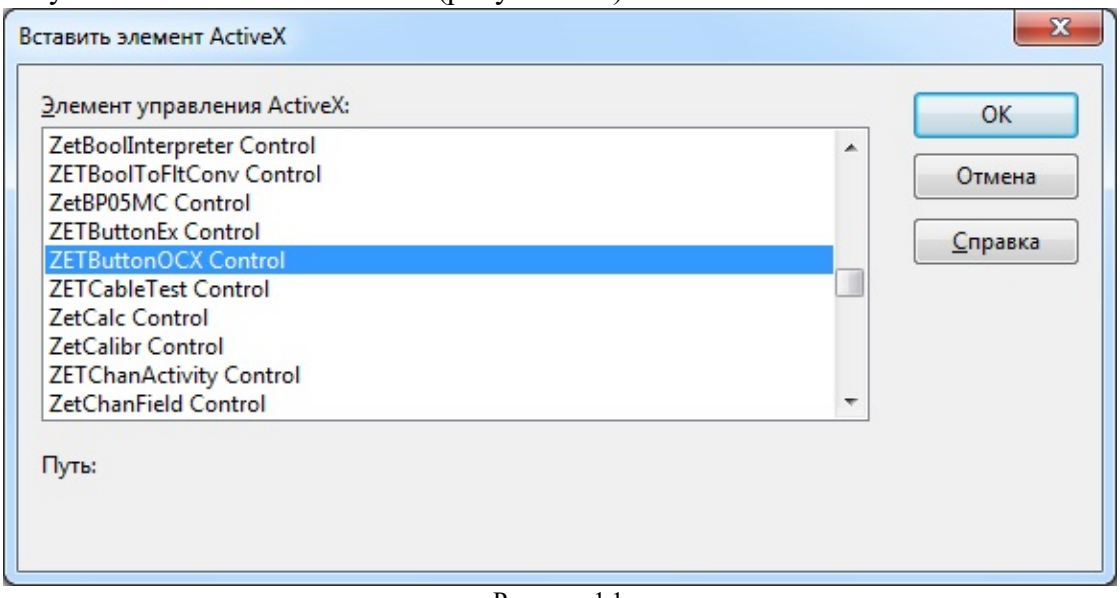



Рисунок 1.1

© ООО "ЭТМС" 1992-2024. Все права защищены

Из появившегося диалогового окна следует выбрать компонент *ZETButtonOCX Control* и нажать *OK* (рисунок 1.2). После этого компонент*ZETButtonOCX .ocx* появится на форме диалога (рисунок 1.2).

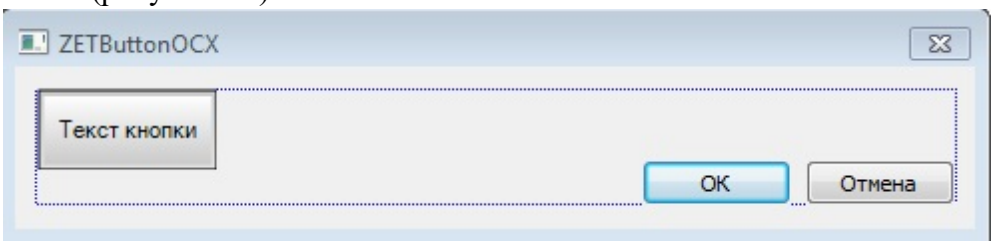
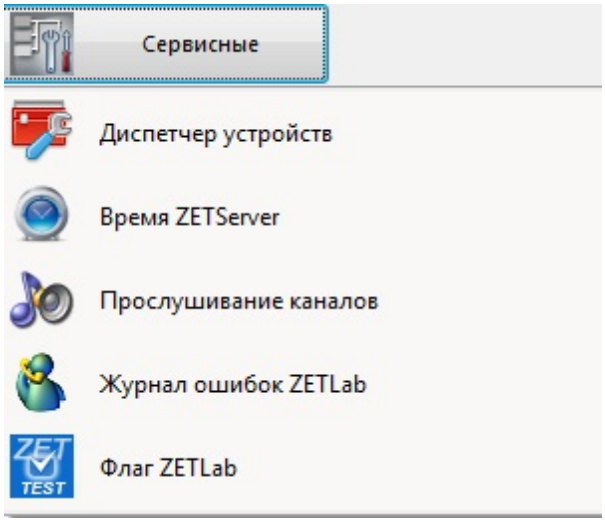



Рисунок 1.2

Одна установленная на форме компонента позволяет управлять одной программой. При необходимости управлять несколькими программами, необходимо установить на форму несколько компонент. Для того чтобы компонента не была видна на форме во время выполнения программы, необходимо установить свойство *Visible* равным *False*.

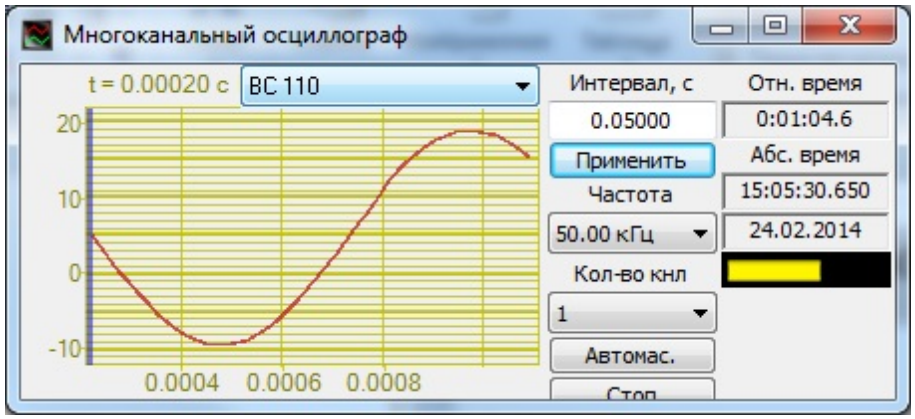
## 1.2.Назначение ZETButtonOCX




Кнопка Сервисные в меню ZETLab

Компонент **ZETButtonOCX** используется в **осциллографе** и в программах, где необходимо применить или отменить выбранные настройки.

© ООО "ЭТМС" 1992-2024. Все права защищены



Кнопка Применить в Осциллографе

## 1.3.Полное описание свойств, методов и событий ZETButtonOCX

Следует отметить, что компонент автомасштабируется при изменении его размеров, поэтому необходимо следить за тем, чтобы высота компонента в приложении была не менее 80 пикселей, иначе он может отображаться некорректно.

### Свойства ZETButtonOCX

Свойства можно задавать на этапе проектирования и во время выполнения программы. На этапе проектирования при выборе компонента и при нажатии клавиши <F4> появляется окно свойств. Эти свойства можно устанавливать на этапе проектирования программы.

Цвета задаются при помощи макроса *RGB(RED, GREEN, BLUE)*. Значения *RED*, *GREEN*, *BLUE* задают интенсивности цветов красного, зеленого и синего. Интенсивности изменяются в пределах от 0 до 255.

*OLE_COLOR* ***ZB_ColorMouseHover*** – цвет кнопки при движении над ней курсора мыши.

*OLE_COLOR* ***ZB_ColorPressedState*** – цвет кнопки в нажатом состоянии.

*OLE_COLOR* ***ZB_ColorUnPressedState*** – цвет кнопки в отжатом состоянии.

Например, для изменении фона графика с желтого на зеленый, могут быть записаны числа:

```
OLE_COLOR a = 0xFF00;
OLE_COLOR b = 0xFFFF;
```

© ООО "ЭТМС" 1992-2024. Все права защищены

*long* ***ZB_Visible*** – видимость кнопки (0 - Спрятана, 1 - Видима).

*long* ***ZB_Enabled*** - реагирует ли кнопка на действия пользователя (0 - Не реагирует, 1 - Реагирует)

*long* ***ZB_Style*** - стиль кнопки (0 - Стандартная, 1 - Плоская, 2 - Плоская типа кнопки ToolBarXP).

*long* ***ZB_State*** - состояние кнопки (0 - Отжата, 1 - Нажата). Текущее состояние компонента. Если параметр True, то при запуске проекта кнопка будет уже нажатой.

*long* ***ZB_PressStyle -*** тип нажатия кнопки (0 - Без фиксации, 1 - С фиксацией). Кнопка фиксирует свое состояние. Передает значение "1" при переходе в состояние "Включено", "0" - при переходе в состояние "Выключено".

*long* ***ZB_TextAlignment -*** выравнивание текста (0 - Влево, 1 - По центру, 2 - Вправо, 3 - Вверх, 4 - Вниз).

*BSTR* ***ZB_Text -*** текст кнопки.

*long* ***ZB_AutoWidth*** - автоподбор ширины кнопки по содержимому текста (0 - Нет, 1 - Да).

*COleFont* ***ZB_Font*** - шрифт текста.

*BSTR* ***ZB_ToolTipText*** - текст подсказки.

*long* ***ZB_TextWidth -*** ширина текста.

*IPictureDisp** ***ZB_PictureUnPressedState*** - картинка на кнопке в отжатом состоянии.

*long* **ZB_PicturePosition** - позиция картинки относительно текста (0 - Слева, 1 - По центру, 2 - Справа, 3 - Вверху, 4 - Внизу).

© ООО "ЭТМС" 1992-2024. Все права защищены

*long* ***ZB_AlignmentResponse*** **-** зависимость взаимного размещения текста и картинки (0 - Независимо, 1 - Зависимо).

*IPictureDisp** ***ZB_PictureMouseHover*** **-** картинка на кнопке при движении над ней курсора мыши.

*IPictureDisp** ***ZB_PicturePressedState*** - картинка на кнопке в нажатом состоянии.

*long* ***ZB_PictureXSize*** **-** размер рисунка по оси X.

*long* ***ZB_PictureYSize*** **-** размер рисунка по оси Y.

## Методы ZETButtonOCX

Методы можно использовать только во время выполнения программы.

*void* ***AboutBox****()* - содержит информацию о программе

## События от компоненты ZETButtonOCX

*void* ***Click****(void)* – событие возникает при одинарном нажатии на кнопку манипулятора «мышь», при нахождении мыши над кнопкой.

*void* ***DblClick*** *(void)* – событие возникает при двойном нажатии на кнопку манипулятора «мышь», при нахождении мыши над кнопкой.

*void* ***MouseDown*** *(short Button, short Shift, OLE_XPOS_PIXELS x, OLE_YPOS_PIXELS y)* – событие возникает при нажатии на кнопку манипулятора типа «мышь», при нахождении мыши над кнопкой.

Параметры:
*Button* – код кнопки мыши
*Shift* – код нажатия служебных клавиш <*Ctrl*>,<*Shift*>,<*Alt*>.
*x* – координата по оси *X* в пикселах (разрешении экрана)
*y* – координата по оси *Y* в пикселах (разрешении экрана)

*void* ***MouseUp*** *(short Button, short Shift, OLE_XPOS_PIXELS x, OLE_YPOS_PIXELS y)* – событие возникает при отжатии кнопки манипулятора типа «мышь», при нахождении мыши над кнопкой.

Параметры:
*Button* – код кнопки мыши

© ООО "ЭТМС" 1992-2024. Все права защищены

*Shift* – код нажатия служебных клавиш <*Ctrl*>,<*Shift*>,<*Alt*>.
*x* – координата по оси *X* в пикселях (разрешении экрана)
*y* – координата по оси *Y* в пикселях (разрешении экрана)

*void* ***MouseMove****(SHORT Button, SHORT Shift, OLE_XPOS_PIXELS x, OLE_YPOS_PIXELS y)* - событие возникает при движении на кнопке манипулятора «мышь»

Параметры:
*Button* – код кнопки мыши
*Shift* – код нажатия служебных клавиш <*Ctrl*>,<*Shift*>,<*Alt*>.
*x* – координата по оси *X* в пикселях (разрешении экрана)
*y* – координата по оси *Y* в пикселях (разрешении экрана)

*void* ***MouseLeave****(void)* - событие активируется единожды, когда указатель покидает область элемента.

## 1.4.Краткий перечень функций, используемые ZETButtonOCX

| **Свойства ZETButtonOCX.ocx** | |
|---|---|
| **Цвета кнопки** | |
| *long* ***GetZB_ColorMouseHover()*** | опрашивается цвет кнопки при движении над ней курсора мыши |
| *void* ***SetZB_ColorMouseHover*** *(unsigned long propVal)* | устанавливается цвет кнопки при движении над ней курсора мыши |
| *long* ***GetZB_ColorPressedState()*** | опрашивается цвет кнопки в нажатом состоянии |
| *void* ***SetZB_ColorPressedState*** *(unsigned long propVal)* | устанавливается цвет кнопки в нажатом состоянии |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *long* ***GetZB_ColorUnPressedState*** *()* | опрашивается цвет кнопки в отжатом состоянии |
| *void* ***SetZB_ColorUnPressedState*** *(unsigned long propVal)* | устанавливается цвет кнопки в отжатом состоянии |
| **Параметры кнопки** | |
| *long* ***GetZB_Visible*** *()* | опрашивается видимость кнопки (0 - Спрятана, 1 - Видима). |
| *void* ***SetZB_Visible****(long propVal)* | устанавливается видимость кнопки (0 - Спрятана, 1 - Видима). |
| *long* ***GetZB_Enabled*** *()* | опрашивается реагирует ли кнопка на действия пользователя (0 - Не реагирует, 1 - Реагирует) |
| *void* ***SetZB_Enabled*** *(long propVal)* | устанавливается реагирует ли кнопка на действия пользователя (0 - Не реагирует, 1 - Реагирует) |
| *long* ***GetZB_Style****()* | опрашивается стиль кнопки (0 - Стандартная, 1 - Плоская, 2 - Плоская типа кнопки ToolBarXP) |
| *void* ***SetZB_Style****(long propVal)* | устанавливается стиль кнопки (0 - Стандартная, 1 - Плоская, 2 - Плоская типа кнопки ToolBarXP) |
| *long* ***GetZB_State*** *()* | опрашивается состояние кнопки (0 - Отжата, 1 - Нажата). Текущее состояние компонента. Если параметр True, то при запуске проекта кнопка будет уже нажатой. |
| *void* ***SetZB_State*** *(long propVal)* | устанавливается состояние кнопки (0 - Отжата, 1 - Нажата). Текущее состояние компонента. Если параметр True, то при запуске проекта кнопка будет уже нажатой. |
| *long* ***GetZB_PressStyle****()* | опрашивается тип нажатия кнопки (0 - Без фиксации, 1 - С фиксацией).<br>Кнопка фиксирует свое состояние. Передает значение "1" при переходе в состояние "Включено", "0" - при переходе в состояние "Выключено". |
| *void* ***SetZB_PressStyle****(long propVal)* | устанавливается тип нажатия кнопки (0 - Без фиксации, 1 - С фиксацией).<br>Кнопка фиксирует свое состояние. Передает значение "1" при переходе в состояние "Включено", "0" - при переходе в состояние "Выключено". |
| **Параметры текста кнопки** | |
| *long* ***GetZB_TextAlignment*** *()* | опрашивается выравнивание текста (0 - Влево, 1 - По центру, 2 - Вправо, 3 - Вверх, 4 - Вниз). |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *void* ***SetZB_TextAlignment*** *(long propVal)* | устанавливается выравнивание текста (0 - Влево, 1 - По центру, 2 - Вправо, 3 - Вверх, 4 - Вниз). |
| *CString* ***GetZB_Text()*** | опрашивается текст кнопки |
| *void* ***SetZB_Text****(CString propVal)* | устанавливается текст кнопки |
| *long* ***ZB_AutoWidth()*** | опрашивается автоподбор ширины кнопки по содержимому текста (0 - Нет, 1 - Да) |
| *void* ***ZB_AutoWidth****(long propVal)* | устанавливается автоподбор ширины кнопки по содержимому текста (0 - Нет, 1 - Да) |
| *COleFont* ***GetZB_Font()*** | опрашивается шрифт текста |
| *void* ***SetZB_Font****(LPDISPATCH propVal)* | устанавливается шрифт текста |
| *CString* ***GetZB_ToolTipText*** *()* | опрашивается текст подсказки |
| *void* ***SetZB_ToolTipText****(CString propVal)* | устанавливается текст подсказки |
| *long* ***GetZB_TextWidth*** *()* | опрашивается ширина текста |
| *void* ***SetZB_TextWidth****(long propVal)* | устанавливается ширина текста |
| **Параметры рисунка** | |
| *CPicture* ***GetZB_PictureUnPressedState()*** | опрашивается картинка на кнопке в отжатом состоянии |
| *void* ***SetZB_PictureUnPressedState****(LPDISPATCH propVal)* | устанавливается картинка на кнопке в отжатом состоянии |
| *long* ***GetZB_PicturePosition*** *()* | опрашивается позиция картинки относительно текста (0 - Слева, 1 - По центру, 2 - Справа, 3 - Вверху, 4 - Внизу) |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *void* ***SetZB_PicturePosition*** *(long propVal)* | устанавливается позиция картинки относительно текста (0 - Слева, 1 - По центру, 2 - Справа, 3 - Вверху, 4 - Внизу). |
| *long* ***GetZB_AlignmentResponse*** *()* | опрашивается зависимость взаимного размещения текста и картинки (0 - Независимо, 1 - Зависимо). |
| *void* ***SetZB_AlignmentResponse*** *(long propVal)* | устанавливается зависимость взаимного размещения текста и картинки (0 - Независимо, 1 - Зависимо). |
| *CPicture* ***GetZB_PictureMouseHover()*** | опрашивается картинка на кнопке при движении над ней курсора мыши |
| *void* ***SetZB_PictureMouseHover*** *(LPDISPATCH propVal)* | устанавливается картинка на кнопке при движении над ней курсора мыши |
| *CPicture* ***GetZB_PicturePressedState*** *()* | опрашивается картинка на кнопке в нажатом состоянии |
| *void* ***SetZB_PicturePressedState*** *(LPDISPATCH propVal)* | устанавливается картинка на кнопке в нажатом состоянии |
| *long* ***GetZB_PictureXSize()*** | опрашивается размер рисунка по оси X |
| *void* ***SetZB_PictureXSize****(long propVal)* | устанавливается размер рисунка по оси X |
| *long* ***GetZB_PictureYSize()*** | опрашивается размер рисунка по оси Y |
| *void* ***SetZB_PictureYSize****(long propVal)* | устанавливается размер рисунка по оси Y |
| **Методы ZETButtonOCX.ocx** | |
| *void* ***AboutBox()*** | содержит информацию о программе |
| **События от компоненты ZETButtonOCX.ocx** | |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *void* ***Click*** *()* | событие возникает при одинарном нажатии на кнопку манипулятора «мышь», при нахождении мыши над кнопкой |
| *void* ***DblClick*** *()* | событие возникает при двойном нажатии на кнопку манипулятора «мышь», при нахождении мыши над кнопкой |
| *void* ***MouseDown****(short Button, short Shift, OLE_XPOS_PIXELS x, OLE_YPOS_PIXELS y)* | событие возникает при нажатии на кнопку манипулятора типа «мышь», при нахождении мыши над кнопкой |
| *void* ***MouseUp****(short Button, short Shift, OLE_XPOS_PIXELS x, OLE_YPOS_PIXELS y)* | событие возникает при отжатии кнопки манипулятора типа «мышь», при нахождении мыши над кнопкой |
| *void* ***MouseMove****(SHORT Button, SHORT Shift, OLE_XPOS_PIXELS x, OLE_YPOS_PIXELS y)* | событие возникает при движении на кнопке манипулятора «мышь» |
| *void* ***MouseLeave****(void)* | событие активируется единожды, когда указатель покидает область элемента |

# Глава 2.ZRegulator.ocx кнопка для выбора элемента из списка

Компонент **ZRegulator.ocx** предназначен для выбора элемента из списка, отображается на кнопке. Также выбор элемента списка можно осуществлять прокруткой колесика мыши при нахождении курсора над компонентом.

## 2.1.Установка компонента ZRegulator.ocx

Для работы с модулем ZRegulator его необходимо сначала установить в программу. При загрузке программы необходимо загрузить с помощью компонента необходимую программу, разместить окно программы на экране в удобном месте, установить необходимые параметры в программе и затем по событиям, происходящим от программы считывать данные из программы.

© ООО "ЭТМС" 1992-2024. Все права защищены

Для установки компонента ZRegulator в проект *Microsoft Visual Studio 2010*, необходимо кликнуть правой кнопкой мыши на форме диалога и из появившегося меню выбрить пункт *Insert ActiveX Control*...(рисунок 1.1)

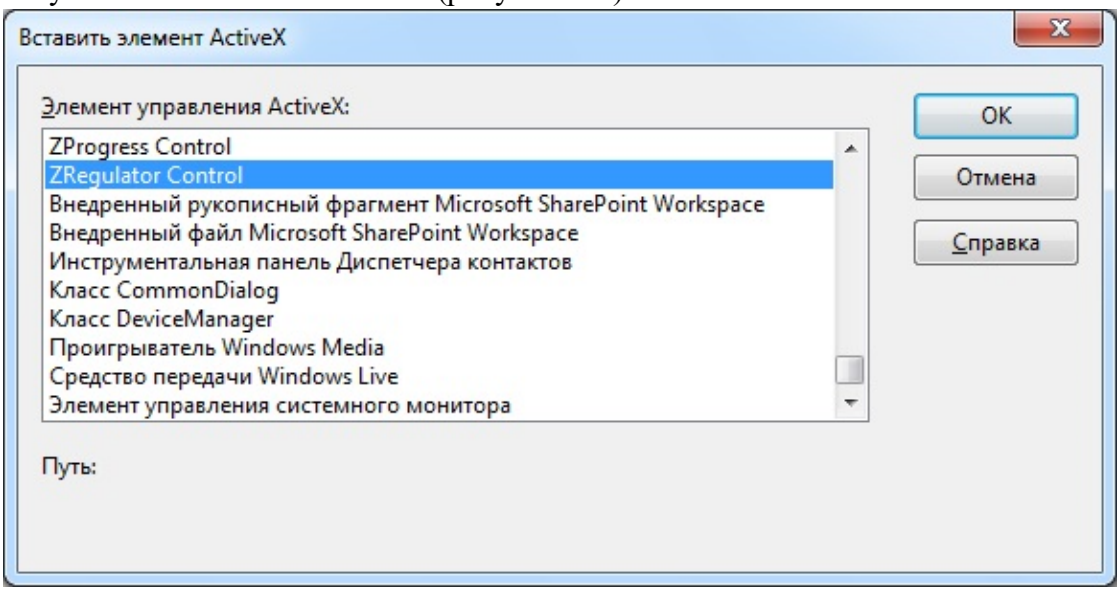



Рисунок 1.1

Из появившегося диалогового окна следует выбрать компонент *ZRegulator Control* и нажать *OK* (рисунок 1.2). После этого компонент *ZRegulator.ocx* появится на форме диалога (рисунок 1.2).

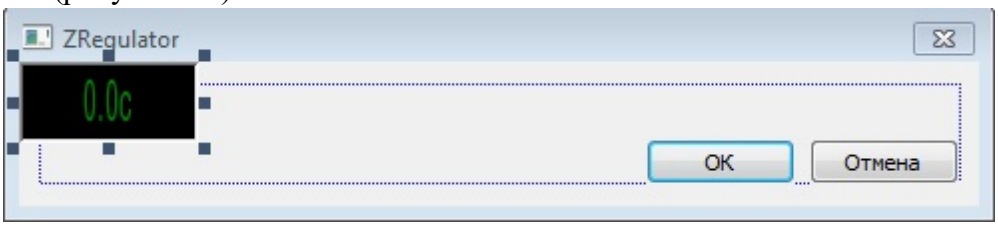



Рисунок 1.2

Одна установленная на форме компонента позволяет управлять одной программой. При необходимости управлять несколькими программами, необходимо установить на форму несколько компонент. Для того чтобы компонента не была видна на форме во время выполнения программы, необходимо установить свойство *Visible* равным *False*.

## 2.2.Интерфейс компонента ZRegulator.ocx

**Назначение**

Элемент представляет кнопку со списком

Внешний вид компонента:

© ООО "ЭТМС" 1992-2024. Все права защищены




**Использование компонента**

При нажатии на правую кнопку мыши появляется список. Выбранный элемент списка отображается на компоненте. Также выбор элемента списка можно осуществлять прокруткой колесика мыши при нахождении курсора над компонентом.




**ZRegulator** используется в таких программах, как "Генератор с ОС (Классический удар)", "Генератор с ОС (Виброудар)", "Генератор с ОС (Синусоидальная вибрация)", "Генератор с ОС (ШСВ) ".

## 2.3.Полное описание свойств, методов и событий ZRegulator.ocx

Следует отметить, что компонент автомасштабируется при изменении его размеров, поэтому необходимо следить за тем, чтобы высота компонента в приложении была не менее 80 пикселей, иначе он может отображаться некорректно.

### Свойства ZRegulator.ocx

Свойства можно задавать на этапе проектирования и во время выполнения программы. На этапе проектирования при выборе компонента и при нажатии клавиши <F4> появляется окно свойств. Эти свойства можно устанавливать на этапе проектирования программы.

© ООО "ЭТМС" 1992-2024. Все права защищены

Цвета задаются при помощи макроса *RGB(RED, GREEN, BLUE)*. Значения *RED*, *GREEN*, *BLUE* задают интенсивности цветов красного, зеленого и синего. Интенсивности изменяются в пределах от 0 до 255.

*OLE_COLOR* ***ClrBgr*** – цвет фона

*OLE_COLOR* ***ClrText*** – цвет текста

Например, для изменении фона графика с желтого на зеленый, могут быть записаны числа:

```
OLE_COLOR a = 0xFF00;
OLE_COLOR b = 0xFFFF;
```

Используются для создания эффекта объемного элемента.

*VARIANT_BOOL* ***BevelLeft*** – рисовать левую границу рамки.

*VARIANT_BOOL* ***BevelRight*** – рисовать правую границу рамки.

*VARIANT_BOOL* ***BevelUp*** – рисовать верхнюю границу рамки.

*VARIANT_BOOL* ***BevelDown*** – рисовать нижнюю границу рамки.

Изменение видимости отображения границ :
true - Видимость отображения границ.
false - Видимость отображения границ - выключена.

*VARIANT_BOOL* ***Bold*** – включить/выключить утолщенный шрифт,

*VARIANT_BOOL* ***Italic*** – включить/выключить наклонный шрифт.

*BSTR* ***FontFace*** – название шрифта, как строковая переменная.

© ООО "ЭТМС" 1992-2024. Все права защищены

*long* ***ItemIndex*** – индекс (порядковый номер) выбранного элемента из списка (нумерация идет с 0).

*BSTR* ***Value*** – количество символов. При Value<0 ограничения не накладываются.

Весь список отображается единым шрифтом, длина строки равна длине максимального элемента списка или принудительно установленной длине строки через метод *SetMinStringLength.*

*long* ***StringCount*** – число строк в списке значений (не меняется пользователем).

## Методы ZRegulator.ocx

Методы можно использовать только во время выполнения программы.

*void* ***AboutBox****()* –содержит информацию о программе.

*void* ***Reset****(void)* – очистить список.

*void* ***AddString****(BSTR sItem)* – добавить элемент *sItem* в список.

*long* ***EnaWindow****(LONG enable)* - определяется включенность, а если выключен, то окрашивается другим цветом.

*SHORT* ***GetMinStringLength****(void)* - опрос минимальной длины текста

*void* ***SetMinStringLength****(SHORT Value)* – установить минимальное количество символов в строке. Элементы списка, имеющие меньшую длину автоматически дополняются пробелами.

## События от компоненты ZRegulator.ocx

*void* ***Modify****(BSTR sValue)* – событие возникает при выборе элемента из списка.

Параметры:
*sValue* – строковая величина выбранного элемента.

© ООО "ЭТМС" 1992-2024. Все права защищены

## 2.4.Краткий перечень функций, используемые ZRegulator.ocx

| Свойства ZRegulator.ocx | |
|---|---|
| **Цвета** | |
| *long* ***GetClrBgr*** *()* | опрашивается цвет фона |
| *void* ***GetClrBgr*** *(unsigned long propVal)* | устанавливается цвет фона |
| *long* ***GetClrText*** *()* | опрашивается цвет текста |
| *void* ***SetClrText*** *(unsigned long propVal)* | устанавливается цвет текста |
| **Отображение рамки** | |
| *BOOL* ***GetLeftBound****()* | опрашивается рисовать левую границу рамки (true(1) - рисовать, false(0) - не рисовать) |
| *void* ***SetLeftBound****(BOOL propVal)* | устанавливается рисовать левую границу рамки (true(1) - рисовать, false(0) - не рисовать) |
| *BOOL* **GetRightBound** *()* | опрашивается рисовать правую границу рамки (true(1) - рисовать, false(0) - не рисовать) |
| *void* ***SetRightBound*** *(BOOL propVal)* | устанавливается рисовать правую границу рамки (true(1) - рисовать, false(0) - не рисовать) |
| *BOOL* ***GetTopBound*** *()* | опрашивается рисовать верхнюю границу рамки (true(1) - рисовать, false(0) - не рисовать) |
| *void* ***SetTopBound*** *(BOOL propVal)* | устанавливается рисовать верхнюю границу рамки (true(1) - рисовать, false(0) - не рисовать) |
| *BOOL* ***GetBottomBound*** *()* | опрашивается рисовать нижнюю границу рамки (true(1) - рисовать, false(0) - не рисовать) |
| *void* ***SetBottomBound*** *(BOOL propVal)* | устанавливается рисовать нижнюю границу рамки (true(1) - рисовать, false(0) - не рисовать) |
| **Параметры шрифта** | |
| *BOOL* ***GetBold****()* | опрашивается включить/выключить утолщенный шрифт (1(true) - Да, 0(false) - нет) |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *void* ***SetBold****(BOOL propVal)* | устанавливается включить/выключить утолщенный шрифт (1(true) - Да, 0(false) - нет) |
| *BOOL* ***GetItalic****()* | опрашивается включить/выключить наклонный шрифт (1(true) - Да, 0(false) - нет) |
| *void* ***SetItalic****(BOOL propVal)* | устанавливается включить/выключить наклонный шрифт (1(true) - Да, 0(false) - нет) |
| *CString* ***GetFontFace****()* | опрашивается название шрифта, как строковая переменная |
| *void* ***SetFontFace****(CString propVal)* | устанавливается название шрифта, как строковая переменная |
| **Параметры управления списком кнопки** | |
| *long* ***GetItemIndex****()* | опрашивается индекс (порядковый номер) выбранного элемента из списка (нумерация идет с 0). |
| *void* ***SetItemIndex****(long propVal)* | устанавливается индекс (порядковый номер) выбранного элемента из списка (нумерация идет с 0). |
| *CString* ***GetValue****()* | опрашивается количество символов. При Value<0 ограничения не накладываются |
| *void* ***SetValue****(CString propVal)* | устанавливается количество символов. При Value<0 ограничения не накладываются |
| *long* ***GetStringCount****()* | опрашивается число строк в списке значений (не меняется пользователем) |
| *void* ***SetStringCount****(long propVal)* | устанавливается число строк в списке значений (не меняется пользователем) |

© ООО "ЭТМС" 1992-2024. Все права защищены

| Методы ZRegulator.ocx | |
|---|---|
| *void **AboutBox**()* | содержит информацию о программе |
| *void **Reset**(void)* | очистить список |
| *void **AddString**(BSTR sItem)* | добавить элемент *sItem* в список |
| *long **EnaWindow**(LONG enable)* | определяется включенность, а если выключен, то окрашивается другим цветом |
| *SHORT **GetMinStringLength**(void)* | опрос минимальной длины текста |
| *void **SetMinStringLength**(SHORT Value)* | установить минимальное количество символов в строке. Элементы списка, имеющие меньшую длину автоматически дополняются пробелами. |

© ООО "ЭТМС" 1992-2024. Все права защищены

| События от компоненты ZRegulator.ocx | |
|---|---|
| *void* ***Modify****(BSTR sValue)* | событие возникает при выборе элемента из списка |

# Глава 3.Zbutton.ocx кнопка для выбора элемента из списка

Компонента Zbutton.ocx совмещает в себе свойства кнопки и свойства элемента выбора

## 3.1.Установка компонента Zbutton.ocx

Для работы с модулем Zbutton его необходимо сначала установить в программу. При загрузке программы необходимо загрузить с помощью компонента необходимую программу, разместить окно программы на экране в удобном месте, установить необходимые параметры в программе и затем по событиям, происходящим от программы считывать данные из программы.

Для установки компонента **Zbutton** в проект *Microsoft Visual Studio 2010*, необходимо кликнуть правой кнопкой мыши на форме диалога и из появившегося меню выбрить пункт *Insert ActiveX Control*...(рисунок 1.1)

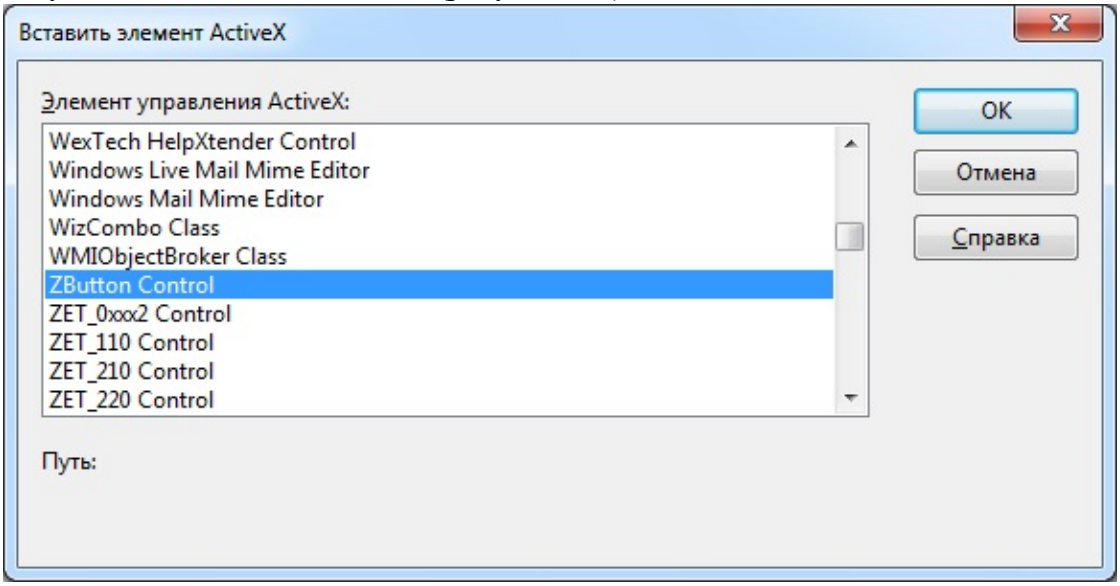



Рисунок 1.1

Из появившегося диалогового окна следует выбрать компонент *Zbutton Control* и нажать *OK* (рисунок 1.2). После этого компонент *Zbutton.ocx* появится на форме диалога (рисунок 1.2).

© ООО "ЭТМС" 1992-2024. Все права защищены

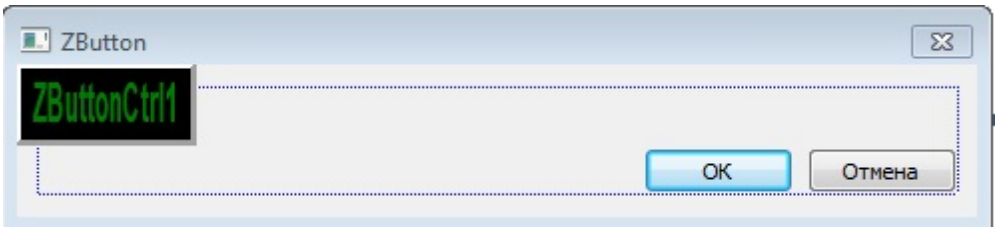



Рисунок 1.2

Одна установленная на форме компонента позволяет управлять одной программой. При необходимости управлять несколькими программами, необходимо установить на форму несколько компонент. Для того чтобы компонента не была видна на форме во время выполнения программы, необходимо установить свойство *Visible* равным *False*.

## 3.2.Назначение и использование Zbutton.ocx

**Назначение**

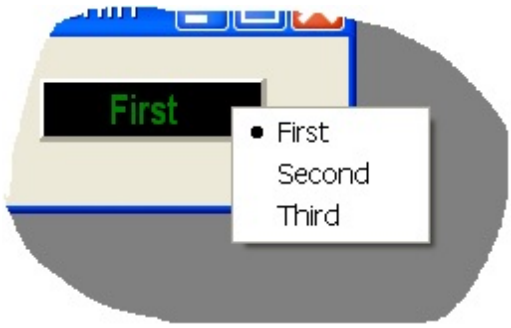
Элемент **Zbutton.ocx** совмещает в себе свойства кнопки и свойства элемента выбора. При нажатии на левую кнопку мыши возникает событие *Click*(). При нажатии на правую кнопку мыши возникает список. Из этого списка необходимо выбрать элемент. Этот элемент отображается на кнопке и возникает событие Modify(…). Надпись на кнопке масштабируется по размеру кнопки. Элемент имеет следующий вид:




**Использование компонента**

При нажатии на правую кнопку мыши появляется список

© ООО "ЭТМС" 1992-2024. Все права защищены




**Zbutton.ocx** используется в таких программах, как "Посылка телеметрической информации", в источнике питания "PSM2010", "Управление коммутационным блоком", "Формула", "Синхронный генератор", "Генератор сигналов".

## 3.3.Полное описание свойств, методов и событий Zbutton.ocx

Следует отметить, что компонент автомасштабируется при изменении его размеров, поэтому необходимо следить за тем, чтобы высота компонента в приложении была не менее 80 пикселей, иначе он может отображаться некорректно.

### Свойства Zbutton.ocx

Свойства можно задавать на этапе проектирования и во время выполнения программы. На этапе проектирования при выборе компонента и при нажатии клавиши <F4> появляется окно свойств. Эти свойства можно устанавливать на этапе проектирования программы.

*long* ***ItemIndex*** – индекс (порядковый номер) выбранного элемента из списка (нумерация идет с 0).

Цвета задаются при помощи макроса *RGB(RED, GREEN, BLUE)*. Значения *RED*, *GREEN*, *BLUE* задают интенсивности цветов красного, зеленого и синего. Интенсивности изменяются в пределах от 0 до 255.

*OLE_COLOR* ***ClrText*** – надписи на кнопке.

*OLE_COLOR* ***ClrBgr*** – цвет фона

© ООО "ЭТМС" 1992-2024. Все права защищены

*OLE_COLOR* ***ClrPressed*** – цвет фона нажатой кнопки (если кнопка имеет фиксирующий стиль). Это свойство необходимо для создания эффекта подсветки кнопки.

Например, для изменении фона графика с желтого на зеленый, могут быть записаны числа:

```
OLE_COLOR a = 0xFF00;
OLE_COLOR b = 0xFFFF;
```

*long* ***Bold*** – включить/выключить утолщенный шрифт,

*long* ***Italic*** – включить/выключить наклонный шрифт.

*BSTR* ***FontFace*** – название шрифта, как строковая переменная,

*long* ***PressStyle*** – свойство для стиля кнопки. 0 – без фиксации, 1 – с фиксацией.

*long* ***Down*** – состояние нажатой (1) или отжатой (0) кнопки. Или команда нажать или отжать кнопку.

*long* ***FlatButton*** – вид кнопки обычная (0) или плоская (1).

*BSTR* ***Caption*** – надпись на кнопке. Надпись на кнопке можно прочитать или задать.

## Методы Zbutton.ocx

Методы можно использовать только во время выполнения программы.

*void* ***AboutBox()*** – содержит информацию о программе.

© ООО "ЭТМС" 1992-2024. Все права защищены

*void* ***AddString****(BSTR sValue)* – добавить элемент *sValue* в меню.

*void* ***Reset****(void)* – очистить меню,

*SHORT* ***GetMinStringLength****(void)* – получить минимальное количество символов в строке.

*void* ***SetMinStringLength****(SHORT Value)* – установить минимальное количество символов в строке. Элементы списка, имеющие меньшую длину автоматически дополняются пробелами.

Параметры:

*Value* – количество символов. При Value<0 ограничения не накладываются.

Весь список отображается единым шрифтом, длина строки равна длине максимального элемента списка или принудительно установленной длине строки через метод *SetMinStringLength.*

### События от компоненты Zbutton.ocx

*void* ***Click*** *(void)* – событие возникает при одинарном нажатии на кнопку манипулятора «мышь», при нахождении мыши над индикатором.

*void* ***MouseUp*** *(short Button, short Shift, OLE_XPOS_PIXELS x, OLE_YPOS_PIXELS y)* – событие возникает при отжатии кнопки манипулятора типа «мышь», при нахождении мыши над индикатором.

Параметры:
*Button* – код кнопки мыши
*Shift* – код нажатия служебных клавиш <*Ctrl*>,<*Shift*>,<*Alt*>.
*x* – координата по оси *X* в пикселях (разрешении экрана)
*y* – координата по оси *Y* в пикселях (разрешении экрана)

*void* ***Modify****(BSTR sValue)* – событие возникает при выборе элемента из списка.

Параметры:
sValue – строковая величина выбранного элемента.

## 3.4.Краткий перечень функций, используемые Zbutton.ocx

© ООО "ЭТМС" 1992-2024. Все права защищены

| Свойства Zbutton.ocx | |
|---|---|
| **Цвета** | |
| *long* ***GetClrText****()* | опрашивается надписи на кнопке |
| *void* ***SetClrText****(unsigned long propVal)* | устанавливается надписи на кнопке |
| *long* ***GetClrBgr****()* | опрашивается цвет фона |
| *void* ***SetClrBgr****(unsigned long propVal)* | устанавливается цвета фона |
| *long* ***GetClrPressed****()* | опрашивается цвет фона |
| *void* ***Set ClrPressed*** *(unsigned long propVal)* | устанавливается цвет фона |
| **Параметры шрифта** | |
| *long* ***GetItallic****()* | опрашивается стиль шрифта: курсив |
| *void* ***SetItallic****(long propVal)* | устанавливается стиль шрифта: курсив |
| *long* ***GetBold****()* | опрашивается стиль шрифта: жирный |
| *void* ***SetBold****(long propVal)* | устанавливается стиль шрифта: жирный |
| *CString* ***GetFontFace****()* | опрашивается название шрифта, как строковая переменная |
| *void* ***SetFontFace****(CString propVal)* | устанавливается название шрифта, как строковая переменная |
| **Параметры управления списком кнопки** | |
| *long* ***GetPressStyle****()* | опрашивается стиль кнопки. 0 – без фиксации, 1 – с фиксацией |
| *void* ***SetPressStyle****(long propVal)* | опрашивается стиль кнопки. 0 – без фиксации, 1 – с фиксацией |

© ООО "ЭТМС" 1992-2024. Все права защищены

| | |
|---|---|
| *long* ***GetDown****()* | опрашивается состояние нажатой (1) или отжатой (0) кнопки. Или команда нажать или отжать кнопку. |
| *void* ***SetDown****(long propVal)* | устанавливается состояние нажатой (1) или отжатой (0) кнопки. Или команда нажать или отжать кнопку. |
| *long* ***GetFlatButton****()* | опрашивается вид кнопки обычная (0) или плоская (1) |
| *void* ***SetFlatButton****(long propVal)* | устанавливается вид кнопки обычная (0) или плоская (1) |
| *CString* ***GetCaption****()* | опрашивается надпись на кнопке. Надпись на кнопке можно прочитать или задать |
| *void* ***SetCaption****(CString propVal)* | устанавливается надпись на кнопке. Надпись на кнопке можно прочитать или задать |
| **Методы Zbutton.ocx** | |
| *void* ***AboutBox****()* | содержит информацию о программе |
| *void* ***AddString****(BSTR sValue)* | добавить элемент *sValue* в меню. |
| *void* ***Reset****(void)* | очистить меню |
| *SHORT* ***GetMinStringLength****(void)* | получить минимальное количество символов в строке. |
| *void* ***SetMinStringLength****(SHORT Value)* | установить минимальное количество символов в строке. Элементы списка, имеющие меньшую длину автоматически дополняются пробелами. |
| **События от компоненты Zbutton.ocx** | |
| *void* ***Click*** *(void)* | событие возникает при одинарном нажатии на кнопку манипулятора «мышь», при нахождении мыши над индикатором. |
| *void* ***MouseUp*** *(short Button, short Shift, OLE_XPOS_PIXELS x, OLE_YPOS_PIXELS y)* | событие возникает при отжатии кнопки манипулятора типа «мышь», при нахождении мыши над индикатором. |
| *void* ***Modify****(BSTR sValue)* | событие возникает при выборе элемента из списка. |

© ООО "ЭТМС" 1992-2024. Все права защищены

# Глава 4.ZETButtonEx.ocx кнопка для свертывания и развертывания дополнительных окон

Компонента Zbutton.ocx совмещает в себе свойства кнопки и свойства элемента выбора

## 4.1.Установка компонента ZETButtonEx.ocx

Для работы с модулем ZETButtonEx его необходимо сначала установить в программу. При загрузке программы необходимо загрузить с помощью компонента необходимую программу, разместить окно программы на экране в удобном месте, установить необходимые параметры в программе и затем по событиям, происходящим от программы считывать данные из программы.

Для установки компонента **ZETButtonEx** в проект *Microsoft Visual Studio 2010*, необходимо кликнуть правой кнопкой мыши на форме диалога и из появившегося меню выбрить пункт *Insert ActiveX Control*...(рисунок 1.1)

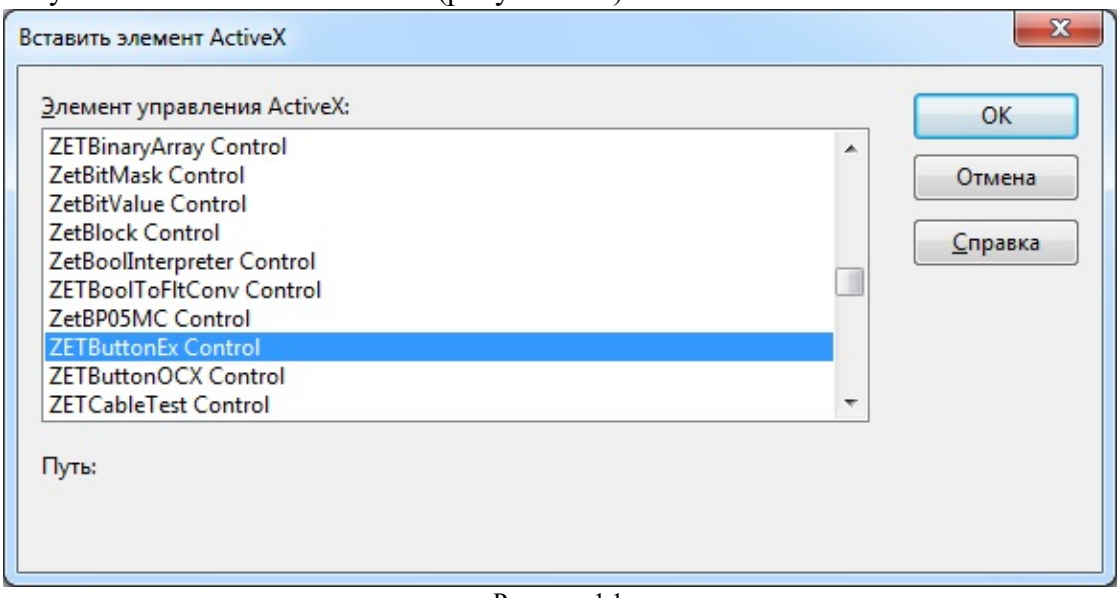



Рисунок 1.1

Из появившегося диалогового окна следует выбрать компонент *ZETButtonEx Control* и нажать *OK* (рисунок 1.2). После этого компонент *ZETButtonEx.ocx* появится на форме диалога (рисунок 1.2).

© ООО "ЭТМС" 1992-2024. Все права защищены

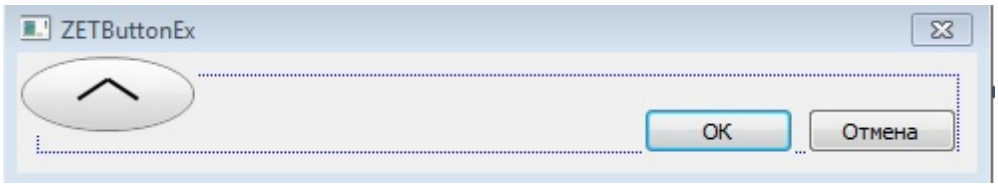
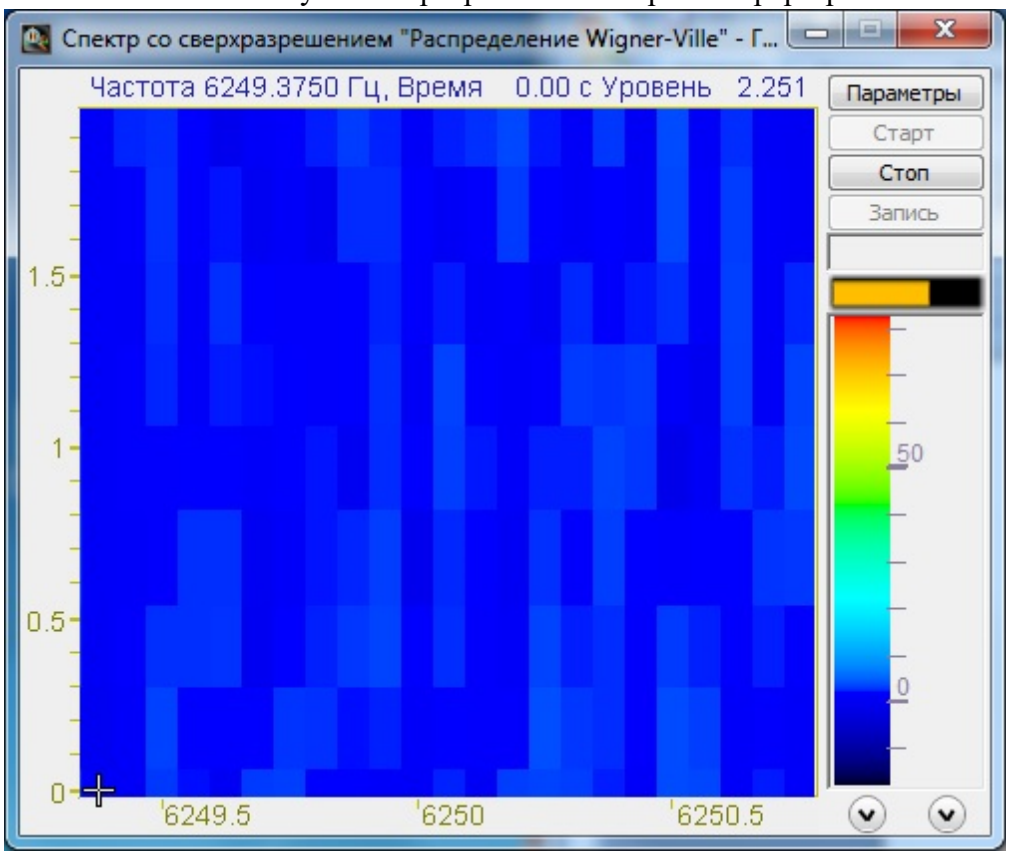



Рисунок 1.2

Одна установленная на форме компонента позволяет управлять одной программой. При необходимости управлять несколькими программами, необходимо установить на форму несколько компонент. Для того чтобы компонента не была видна на форме во время выполнения программы, необходимо установить свойство *Visible* равным *False*.

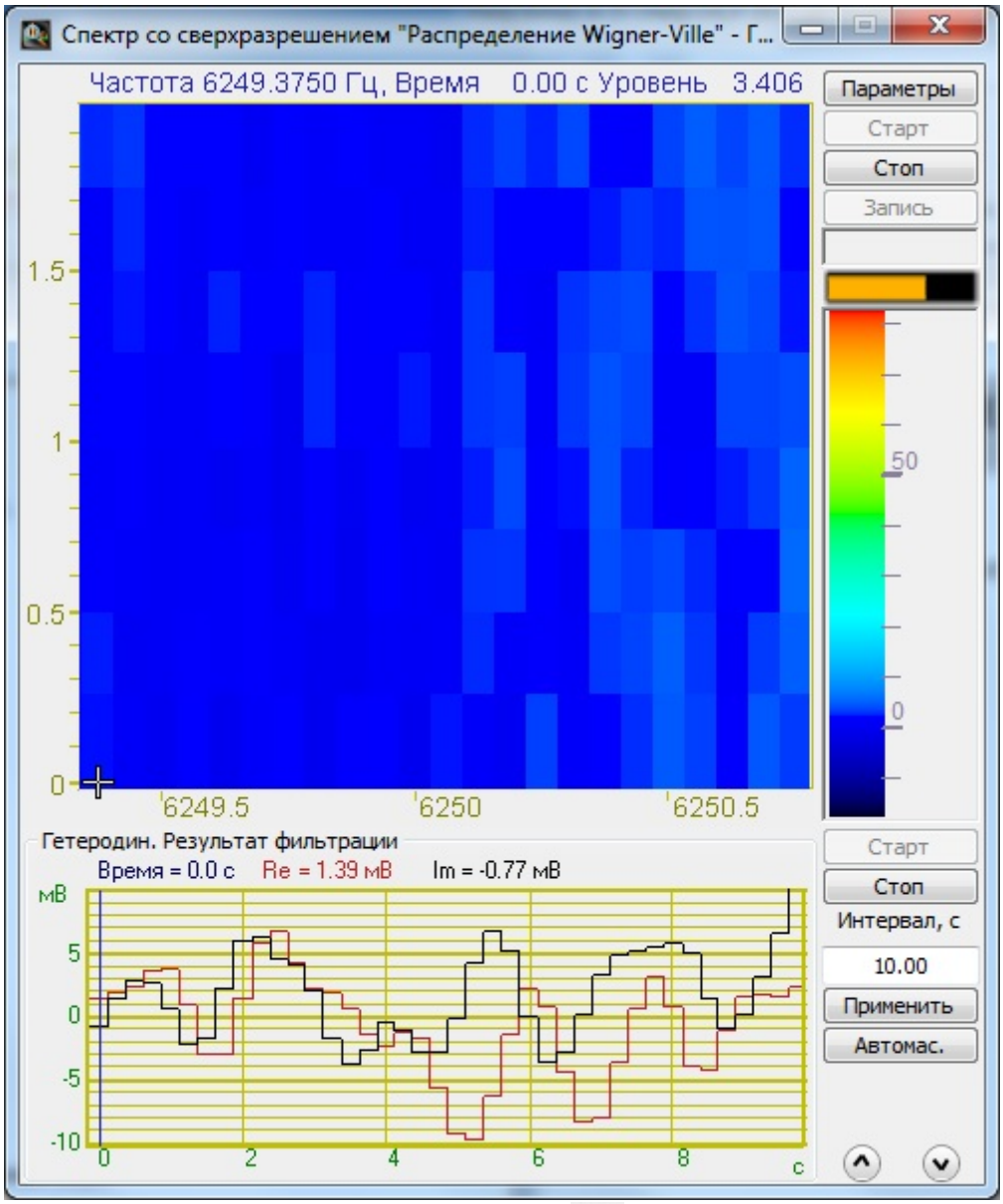
## 4.2.Назначение и использование ZETButtonEx.ocx

**ZETButtonEx.ocx** используется в программе "Спектр со сверхразрешением".




Запущенная программа

© ООО "ЭТМС" 1992-2024. Все права защищены




После нажатия на кнопку **ZETButtonEx.ocx** открывается дополнительное окно Результат фильтрации, после нажатия окно закрывается.

## 4.3.Полное описание свойств, методов и событий ZETButtonEx.ocx

Следует отметить, что компонент автомасштабируется при изменении его размеров, поэтому необходимо следить за тем, чтобы высота компонента в приложении была не менее 80 пикселей, иначе он может отображаться некорректно.

© ООО "ЭТМС" 1992-2024. Все права защищены

## Свойства ZETButtonEx.ocx

Свойства можно задавать на этапе проектирования и во время выполнения программы. На этапе проектирования при выборе компонента и при нажатии клавиши <F4> появляется окно свойств. Эти свойства можно устанавливать на этапе проектирования программы.

*DIRECTION* ***Direction*** - направление, в которое указывает стрелка.

*BSTR* ***ToolTipText*** - текст подсказки

## Методы Zbutton.ocx

Методы можно использовать только во время выполнения программы.

*void* ***AboutBox()*** – содержит информацию о программе.

© ООО "ЭТМС" 1992-2024. Все права защищены

# Примеры программирования

## Программный модуль SRV.ocx

Модуль предназначен для работы с устройствами АЦП-ЦАП, для обработки сигналов в реальном времени и обработки оцифрованных сигналов, записанных в файлы. Модуль обеспечивает обмен данными и потоками данных с драйверами, и другими пользовательскими программами по COM-интерфейсу с использованием технологии клиент-сервер. Модуль поддерживает одновременную работу с несколькими платами АЦП-ЦАП различного или одинакового типа. Суммарное количество каналов не более 200. Суммарное количество работающих устройств в системе не более 50. Оцифрованные данные в программу пользователя передаются в формате плавающей запятой одинарной точности, в заданных единицах измерения. Различные модули АЦП преобразуют аналоговый сигнал в 12, 14, 16 или 24 разрядный код. Компонент *SRV* преобразует эти целочисленные значения в формат плавающей запятой с учетом коэффициентов усиления, чувствительности преобразователей, смещения постоянной составляющей, поправками по измерительным каналам. Эти параметры задаются в программе «Диспетчер устройств» из группы «Сервисные».

## ZETServer - сервер данных

Сервер является основным средством взаимодействия между модулями АЦП ЦАП (или анализаторами спектра) и всеми приборами-программами виртуальной лаборатории ZETLab. Он предназначен для работы с модулями АЦП и ЦАП, для обработки сигналов в реальном времени и обработки оцифрованных сигналов записанных в файлы. Модуль обеспечивает обмен данными и потоками данных с драйверами, и другими пользовательскими программами по COM-интерфейсу с использованием технологии клиент-сервер. Модуль поддерживает одновременную работу с несколькими платами АЦП ЦАП различного или одинакового типа. Суммарное количество каналов не более 60. Оцифрованные данные в программу пользователя передаются в формате плавающей запятой одинарной точности, в заданных единицах измерения. Различные модули АЦП преобразуют аналоговый сигнал в 12, 14, 16 или 24 разрядный код. Сервер преобразует эти целочисленные значения в формат плавающей запятой с учетом коэффициентов усиления, чувствительности преобразователей, смещения постоянной составляющей, поправками по измерительным каналам. Компонент ZETServer обеспечивает одновременную работу с несколькими программами (до 60).

Пример работы с сервером: программа формирования на выходе генератора синусоидального сигнала заданной частоты и уровня СКЗ:

© ООО "ЭТМС" 1992-2024. Все права защищены

**Вопросы и ответы пользователю при работе с Zet 7176 ethernet канала**

**Вопрос 1:**

Subject: Вопрос по программному чтению Zet 7176 ethernet канала

Нам необходимо программно читать поступающие данные в устройство Zet 7176. Чтение будет осуществляться через библиотеку ZADC.dll и код C++ (если это возможно).

Изучив руководство программиста ZXXX, перечень функций ZADC.dll, вопрос, как осуществить это чтение, остается открытым.

Просьба показать пример подобного чтения, и\или показать возможные пути его создания. Мы обладаем лицензией Zetlab Studio.

**Ответ 1:**

Через Zadc.dll читать из ZET7176 невозможно. Необходимо работать через SRV.ocx

**Вопрос 2:**

А имеется ли какой-нибудь пример работы с датчиками через ethernet, SRV.ocx посредством языков программирования C++, C# или Delphi?

**Ответ 2:**

Примеры программ есть на диске с ПО и на ftp-сервере https://file.zetlab.com/Document/Samples/

**Вопрос 3:**

К сожалению, я не нашел там примеров по работе с датчиками через ethernet... Не подскажите, где его можно увидеть?

**Ответ 3:**

Если работать через SRV.ocx, то нет разницы как подключен датчик. Он может быть подключен через USB или через Ethernet. Взаимодействие с данными идет на уровне

канала. Соответственно можно взять в качестве примера общение с SRV.ocx

**Вопрос 4:** Мы пытаемся прочесть вихретоковый преобразователь через Zet 7140-S подключенный в Ethernet.Я нашел пример работы с SRV.ocx (пример был в папке

ZetlabStudio_GeneratorSinusSRV). Однако это пример для подключения к АЦП и работе с ЦАП. Очевидно, что для чтения нашего датчика нужно использовать другие функции.

Какие функции и как использовать? Просьба показать, или дать пример работы в подобном режиме.

**Ответ 4:**

В папке примеров для VisualBasic6 есть примеры работы с SRV.ocx для обработки данных (Sample_I - Sample_IV). Если у Вас нечем просмотреть эти

проекты, то можно просто открыть файлы с расширением *.frm любым текстовым редактором и увидеть код. Вкратце все сводится к следующему: слежению за временем по

© ООО "ЭТМС" 1992-2024. Все права защищены

каналу функцией CurrentTime и забору данных канала функцией GetData при достижении определенного времени. Следить за временем можно и по таймеру и в потоке. В

примерах данные забираются порциями либо по 0.1 секунде, либо по 1 секунде. Ну и еще необходимо не забыть про подключение к серверу при начале работы программы

функцией Connect и про отключение от сервера по окончанию работы программы функцией Disconnect.

**Вопрос 5:**

Подтверждаю, данный метод чтения сработал. Однако, возникла странная проблема. В примере ZetlabStudio_GeneratorSinusSRV невозможно открыть режим редактирования

основного окна программы, как на Visual studio 2010, так и на Visual studio 2015. При открытии выскакивает окно (см. скриншот), после нажатия "Да" закрывается как это окно,

так и вся Visual Studio. При создании подобного проекта заново, выявилось, что проблема находится при добавлении компонента ActiveX SRV Control. После его добавления

эта проблема возникает.Важное уточнение - и пример и созданный проект заново компилируется, и работает правильно, в том числе используя этот компонент SRV.ocx.

Просьба пояснить как устранить данную проблему.

**Ответ 5:**

Установите ранее приобретенный вами ключ с Zetlab Studio, проблема должна решиться.

**Вопрос 6:**

Появился новый вопрос. Программирование собственного чтения датчика на основе SRV.ocx прошло успешно. Однако, частота обновления данных у опрашиваемого канала

равна 0.5 секунд. Необходимо минимизировать эту частоту, чтобы уменьшить задержку поступления данных. Эта проблема с вихретоковым преобразователем Zet7140-S,

который входит в Zet7176 который переводит данные в канал Ethernet, поток этих данных идет по кабелю в компьютер.Стоит добавить важную информацию - в моей

программе (сделанной с использованием SRV.ocx) скорость обновления этого датчика такая же как в программе ZETLab (измерение вольтметр постоянного тока). Там он,

несмотря на установленную частоту обновления раз в 0.1 секунду, выдает обновление раз в 0.5 секунд для этого датчика. Так же стоит добавить, что в моей программе, как

© ООО "ЭТМС" 1992-2024. Все права защищены

и в программе ZETLab, данные, подаваемые напрямую с АЦП Zet220, приходят с задержкой 0.1 сек и менее при тех же настройках. Таймер опроса данных с канала стоит на

частоте 10 мс, установка нулевой или первой декады в функции ReadData не помогает. Как я понимаю проблема в том, что функция GetCurrentTime данного канала

поднимается на 0.5 раз в полсекунды. Скриншот свойств токовихревого преобразователя и Zet7176 прилагаю.

**Ответ 6:**

Данная ситуация вполне нормальна для тех настроек, что у Вас выставлены. Обновление данных 2 раза в секунду связано с буферизацией их в преобразователе ZET7176.

Преобразователь передает данные порциями и при выставленных настройках получается, что они приходят 2 раза в секунду. Для того, чтобы данные приходили чаще,

нужно увеличить частоту дискретизации измерительного датчика, например со 100 Гц до 500 Гц в настройках.

# Глава 1.Примеры работы с CBuilder

Примеры находятся на сервере по адресу: \\192.168.0.7\Setup ZET\Setup_for_diskWriting\Doc\Samples\CBuilder

или на поставляемым диском: Doc\Samples\CBuilder

# Глава 2.Примеры работы с драйверами

Примеры находятся на сервере по адресу: \\192.168.0.7\Setup ZET\Setup_for_diskWriting\Doc\Samples\drivers

или на поставляемым диском: Doc\Samples\drivers

# Глава 3.Примеры работы  с VisualBasic6

Примеры находятся на сервере по адресу: \\192.168.0.7\Setup ZET\Setup_for_diskWriting\Doc\Samples\VisualBasic6

или на поставляемым диском: Doc\Samples\VisualBasic6

© ООО "ЭТМС" 1992-2024. Все права защищены

В приложении на компакт-диске представлены разнообразные примеры программ работы с устройствами АЦП-ЦАП в реальном времени и по записанным файлам с исходными текстами и комментариями.

Sample_I
Sample_II
Sample_III
Sample_IV

В примерах приведено последовательное создание программы «Осциллограф». Программа подключается к серверу и рисует форму сигнала выбранного канала. Программа создана в среде VB6.0. Размер программы – 45 строк.

Sample_V

Пример программы, которая выдает на выходе ЦАП сигнал, усиленный от выбранного канала АЦП или виртуального канала. Программа создана в среде VB6.0. Размер программы – 44 строки.

Sample_VI

Пример программы генератора синусоидальных сигналов заданной частоты и амплитуды. Программа создана в среде VB6.0. Размер программы – 44 строки.

Sample_VII

Пример программы узкополосного фильтра. В программе задается центральная частота фильтра, и полоса фильтра, по этим параметрам рассчитывается квадратурный фильтр. Программа создает три виртуальных канала – выходы квадратурного фильтра действительного и мнимого частей и выход огибающей. Фильтр реализован в виде фильтра с конечно-импульсной характеристикой с весовым окном Хана. Программа создана в среде VB6.0. Размер программы – 100 строк.

Sample_VIII

Пример программы расчета активного сопротивления. Программа включает генератор и вольтметр переменного тока. Сопротивление ставится между выходом ЦАП и входом АЦП. По считываемым значениям вольтметра программа определяет падение напряжения и рассчитывает и отображает величину сопротивления. Программа сохраняет параметры в файле параметров и использует их при следующем запуске. Реализована процедура калибровки. Программа создана в среде VB6.0.

Sample_IX

Во время запуска данной программы параллельно в скрытом режиме запускаются еще 3 программы: вольтметр переменного тока, вольтметр постоянного тока и селективный вольтметр переменного тока. Пользователь выбирает канал модуля АЦП и в реальном времени в окне программы отображаются показания всех вольтметров по выбранному каналу. Программа создана в среде VB6.0.

Sample_X

© ООО "ЭТМС" 1992-2024. Все права защищены

Программа предназначена для работы с цифровым входом в ручном режиме. По таймеру программа 10 раз в секунду просматривает статус (разрешение ввода-вывода) цифрового порта и значение цифрового порта и отображает эту информацию в мнемоническом виде на экране. Программа создана в среде VB6.0.

Sample_XI

Программа – простой цифровой автомат. В зависимости от уровня входного сигнала включается и выключается заданный выходной бит по цифровому порту. В программе выбирается входной аналоговый канал, устанавливаются верхний и нижний предел срабатывания, выбирается выходной цифровой бит, по которому выполняется управление. Алгоритм работы автомата следующий: при превышении уровня сигнала верхнего уровня бит отключается, т.е. туда устанавливается 0, при снижении уровня сигнала нижнего уровня бит включается. Этот автомат может применяться в простых системах регулирования – термостатах, системах освещения, системах сигнализации. Программа создана в среде VB6.0.

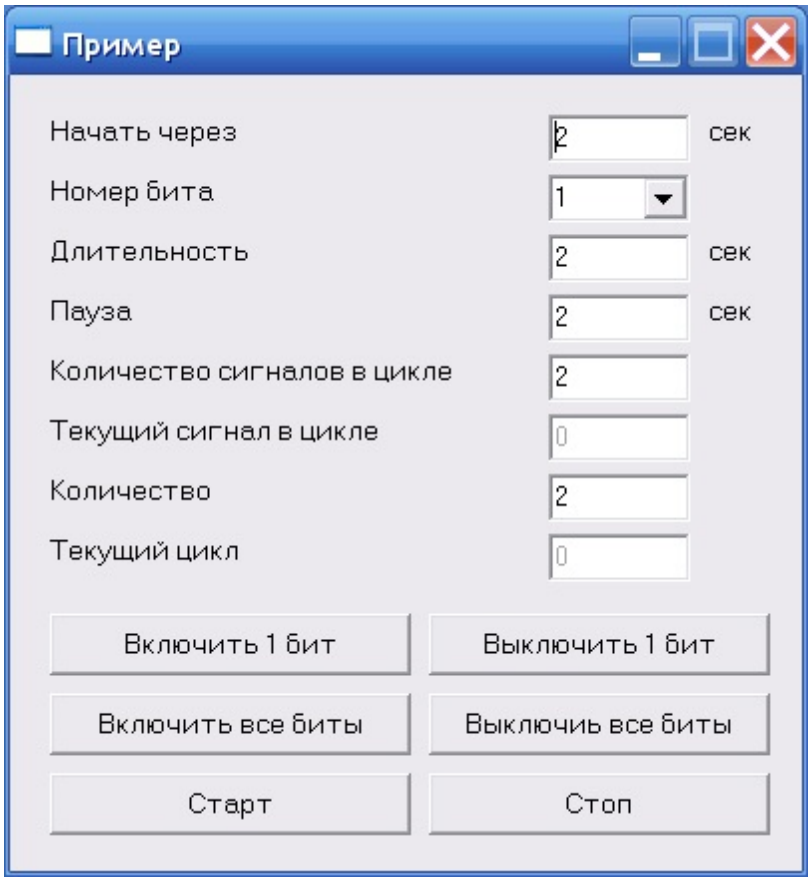
Sample_XII

Программа цифровой фильтрации с конечно-импульсной характеристикой. Программа создает виртуальный канал отфильтрованного сигнала. В программе задается входной канал для фильтрации, тип фильтра – фильтр низких частот, верхних частот, режекторный или полосовой фильтр, тип оконной функции, порядок фильтра – длина фильтра в отсчетах. В программе отображаются в графическом виде частотная и импульсная характеристики фильтра. Программа создана в среде VB6.0.

Sample_XIII

Программа работы с цифровым портом ввода-вывода. Программа создана в среде VB6.0. Программа выполняет функции таймера, счетчика и делителя частоты.

© ООО "ЭТМС" 1992-2024. Все права защищены




Sample_XIV

Программа суммирования сигналов. Программа создает виртуальный канал суммы или разности двух сигналов. Программа создана в среде VB6.0.

Sample_XV

Программа коррекции уровня сигнала. Эта программа предназначена для снятия показаний с датчиков с нелинейной характеристикой, например, тензометрических датчиков, датчиков давления, температуры.

## 3.1.Пример SAMPLE_I

Запускаем Visual Basic 6.0 и создаем новый проект. На панели *ToolBox* нажатием правой кнопкой мыши и во всплывающем меню (рисунок 1.1) выбираем **Components…** (панель *ToolBox* можно вызвать из главного меню командой **View→ToolBox**). Другой способ открыть окно **Components** – выбрать меню **Projects → Components** (рисунок 1.2)

© ООО "ЭТМС" 1992-2024. Все права защищены

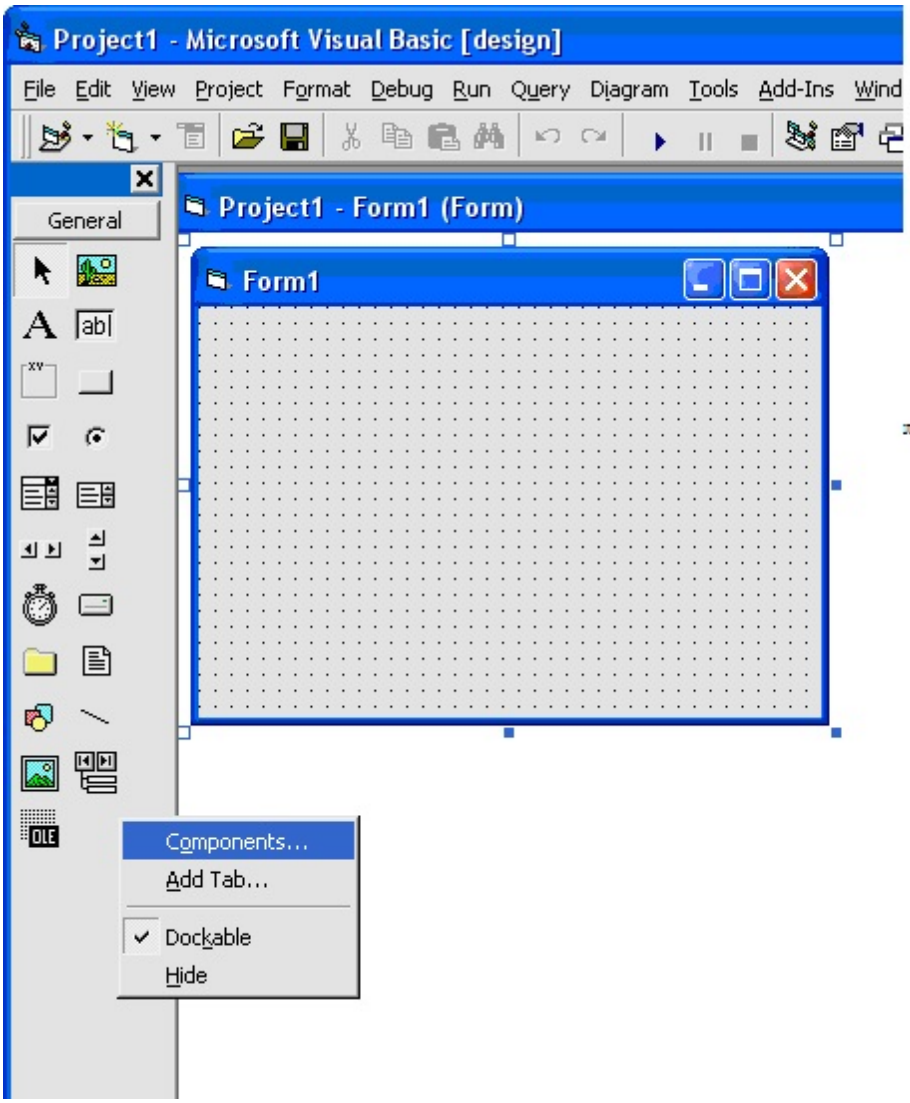



Рисунок 1.1

© ООО "ЭТМС" 1992-2024. Все права защищены

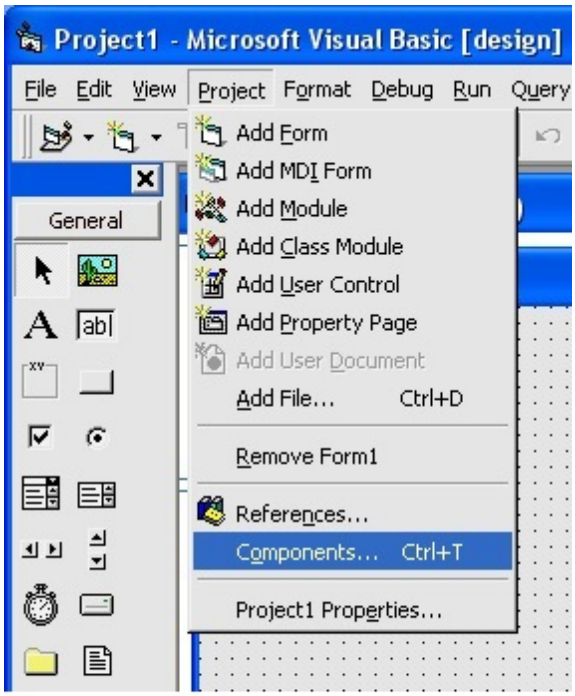



Рисунок 1.2

В окне **Components** (рисунок 1.3) в перечне компонент находим *SRV ActiveX Control Module* и ставим напротив него галочку. При этом на панели *ToolBox* появляется компонент ***SRV***. Нажимаем на этот компонент и устанавливаем его на форму (рисунок 1.4).

© ООО "ЭТМС" 1992-2024. Все права защищены

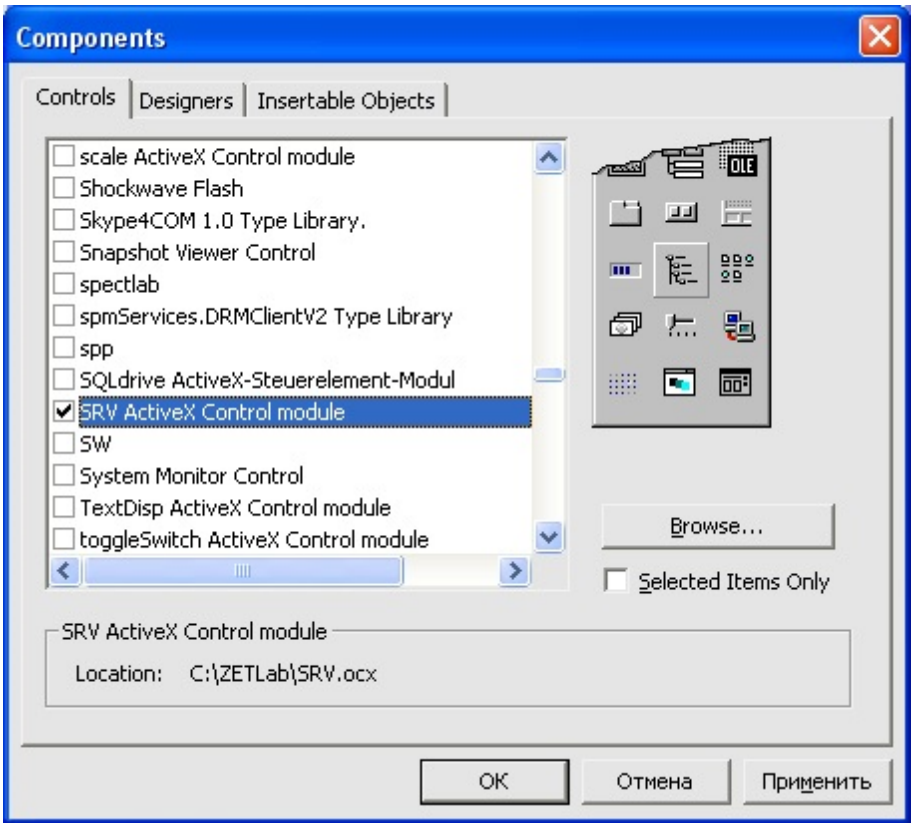



Рисунок 1.3

© ООО "ЭТМС" 1992-2024. Все права защищены

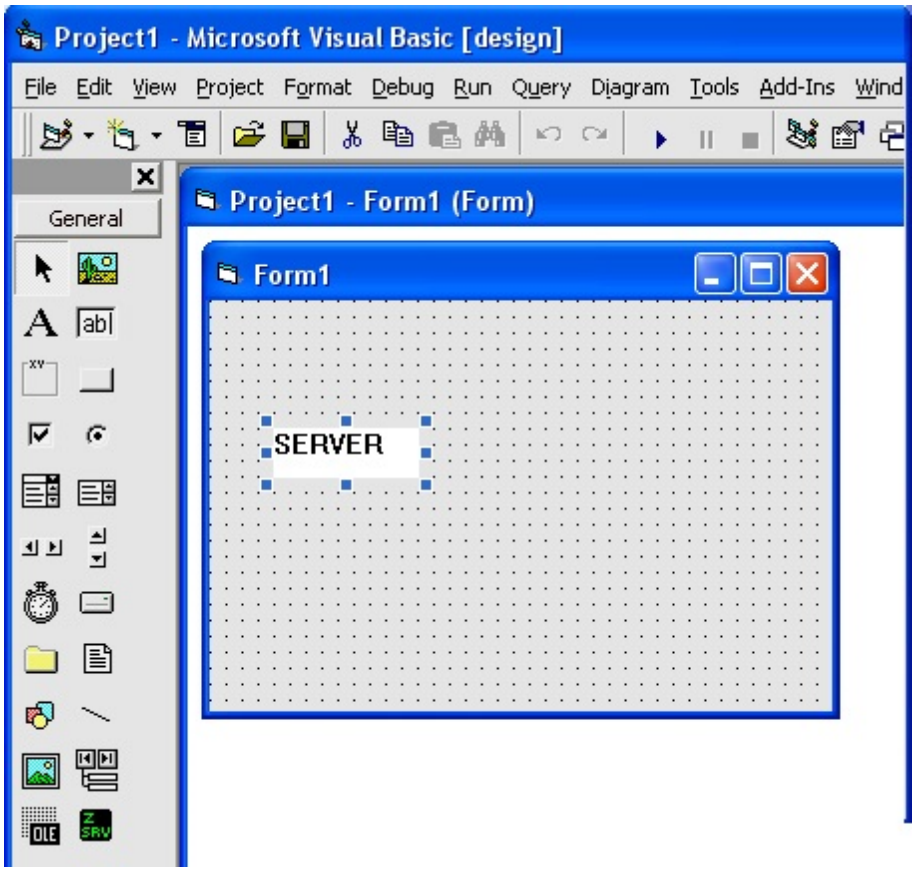



Рисунок 1.4

Устанавливаем аналогично Label и Timer (рисунок 1.5).

© ООО "ЭТМС" 1992-2024. Все права защищены

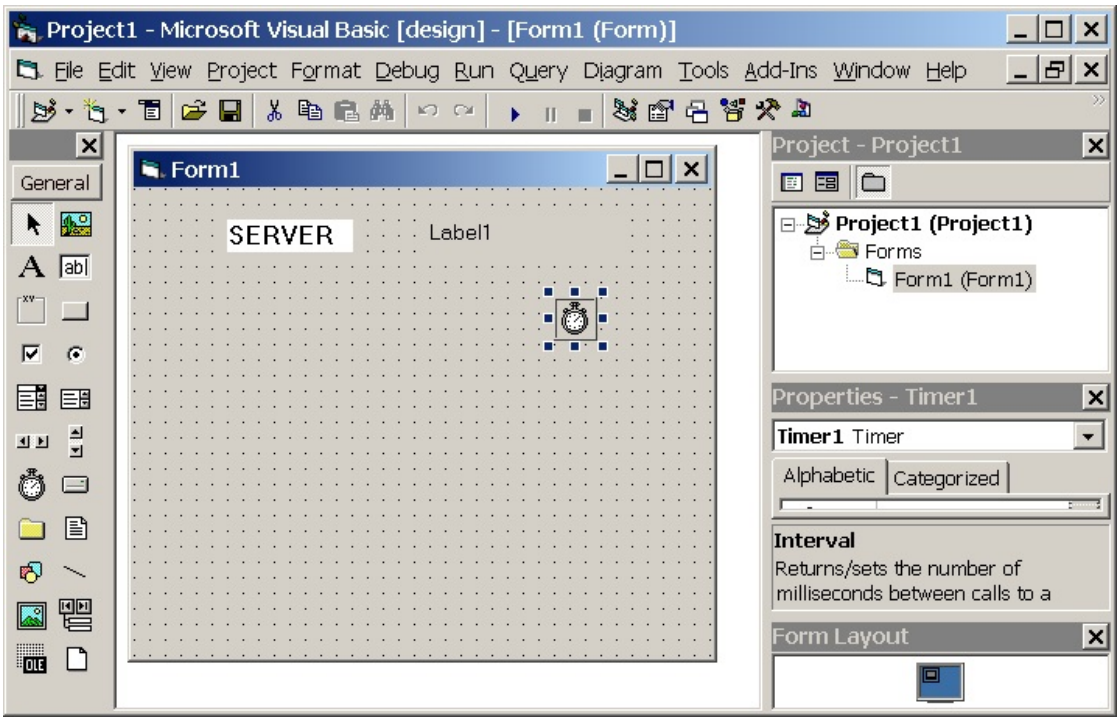



Рисунок 1.5

Дважды нажимаем на форму
и добавляем в код программы

```
Dim MyTime As Double                ' текущее время обработки
Dim MyError As Integer
-------------------------------------------------------------------------------------------
Private Sub Form_Load()             ' вход в программу
MyError = SRV1.Connect()            ' соединиться с сервером
MyTime = SRV1.CurrentTime (0)       ' начальное время
Timer1.Interval = 100           ' запуск таймера с интервалом 100 мс
End Sub
-------------------------------------------------------------------------------------------
Private Sub Form_Unload(Cancel As Integer)  ' выход из программы
MyError = SRV1.Disconnect() ' отсоединиться от сервера
End Sub
-------------------------------------------------------------------------------------------
Private Sub SRV1_Modify(ByVal GlobalPar As Long) ' событие
MyTime = 0                      ' перезапускается сервер и сбрасываем наше время в 0
End Sub
-------------------------------------------------------------------------------------------
Private Sub Timer1_Timer()          ' процедура обработки данных в таймере
Dim Channel As Integer
Dim massiv(100) As Single           ' пользовательский буфер данных
Channel = 0                         ' номер канала АЦП
If MyTime < SRV1.CurrentTime(Channel) Then              ' сравниваем текущее время
```

© ООО "ЭТМС" 1992-2024. Все права защищены

```
    MyErr = SRV1.GetData(Channel, 0, MyTime, 100, massiv(0))          ' берем данные
    MyTime = MyTime + 1                ' увеличиваем время на 1 секунду
    Label1.Caption = massiv(0)  ' выводим на экран значение одного отсчета
End If
End Sub
```

На экране должна появиться следующая картина (рисунок 1.6).

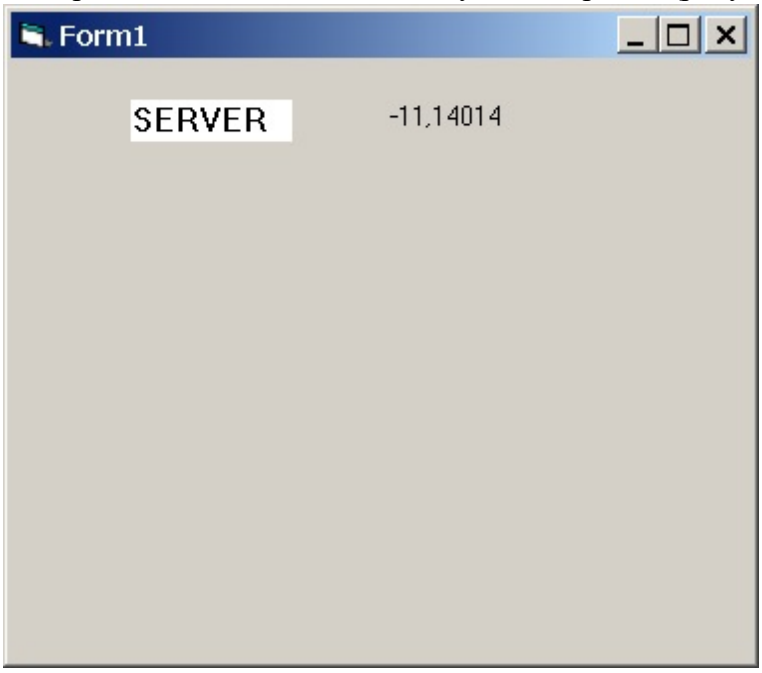



Рисунок 1.6

Для того чтобы во время выполнения программы не было видно надписи компонента «SERVER» необходимо свойство компонента SRV.Visible сделать False.

## 3.2.Пример SAMPLE_2

Подключение графики

Для того чтобы отобразить данные в виде графика в режиме осциллографа, необходимо добавить компонент Grid. Тогда внешний вид формы будет иметь вид (Рисунок 2.1):

© ООО "ЭТМС" 1992-2024. Все права защищены

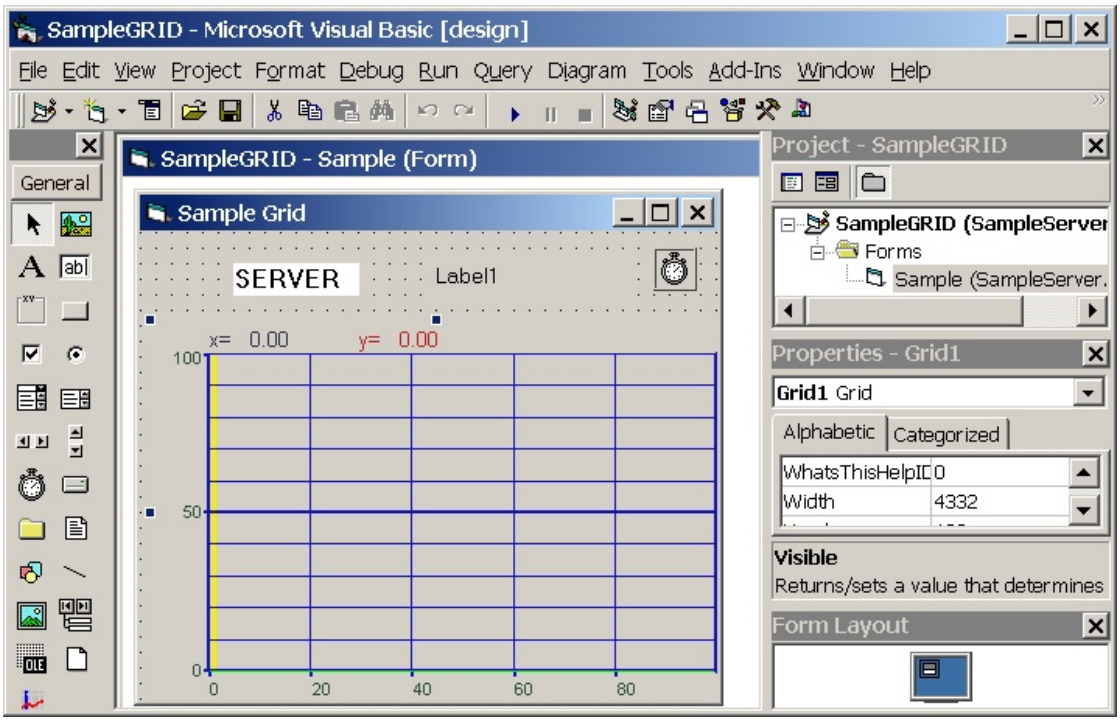



Рисунок 2.1

Необходимо добавить в процедуру обработки функцию построения графика. Новая процедура:

```
Private Sub Timer1_Timer()              ' процедура обработки данных в таймере
Dim Channel As Integer
Dim massiv(100) As Single               ' пользовательский буфер данных
Channel = 0                             ' номер канала АЦП
If MyTime < SRV1.CurrentTime(Channel) Then                  ' сравниваем текущее время
   MyErr = SRV1.GetData(Channel, 0, MyTime, 100, massiv(0))            ' берем данные
   MyTime = MyTime + 0.1                ' увеличиваем время на 0,1 секунду
   Label1.Caption = massiv(0)  ' выводим на экран значение одного отсчета
   Myerr = Grid1.Paint(massiv(0))       'отображение графика
End If
End Sub
```

В результате выполнения получаем следующую картину (Рисунок 2.2):

© ООО "ЭТМС" 1992-2024. Все права защищены

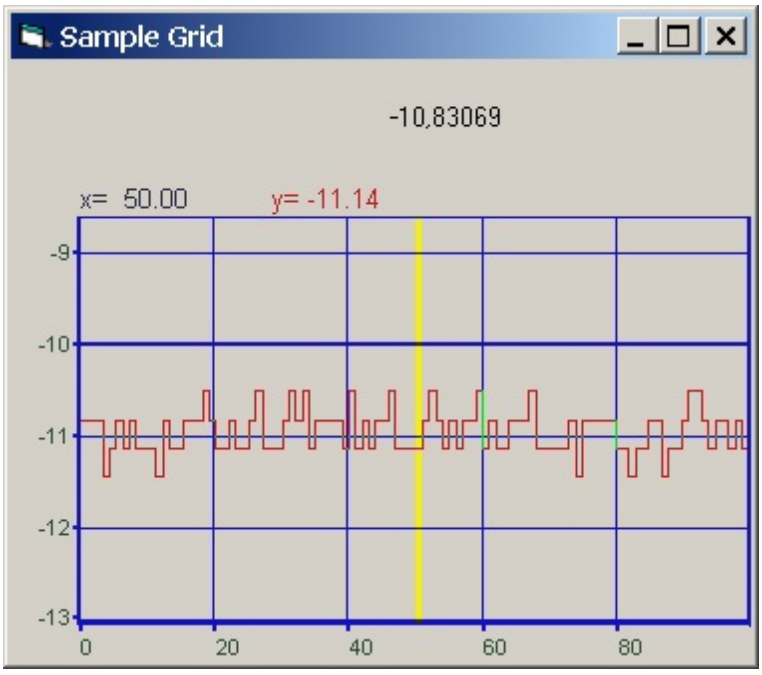


Рисунок 2.2

## 3.3.Пример SAMPLE_3

Создание виртуального канала
В этом примере создается виртуальный канал в виде суммы двух каналов.
В программе добавлена функция
MyError = SRV1.ConnectVrtCh(1) ' соединиться с сервером и создать 1
' виртуальный канал
вместо функции SRV1.Connect()

Исправлена процедура обработки сигналов. Добавлены массивы для обработки. Размер выбирается из частоты дискретизации АЦП и заданного интервала обработки.

```
Private Sub Timer1_Timer()
Dim Channel As Integer, channel2 As Integer
Dim massiv(20000) As Single
Dim massiv2(20000) As Single
Dim massiv3(20000) As Single
Dim interval As Single
Dim size As Integer
interval = 0.1                          ' интервал расчета
Channel = 0                             ' номер канала АЦП
channel2 = 1                            ' номер канала АЦП
size = interval * Freq                  ' порция данных накопленная за интервал времени
If MyTime < SRV1.CurrentTime(Channel) Then          ' сравниваем текущее время
   If MyTime < SRV1.CurrentTime(channel2) Then      ' сравниваем текущее время
      Myerr = SRV1.GetData(Channel, 0, MyTime, size, massiv(0))      ' берем данные по
                                                                     ' первому каналу
```

© ООО "ЭТМС" 1992-2024. Все права защищены

```
      Myerr = SRV1.GetData(channel2, 0, MyTime, size, massiv2(0))  ' берем данные
                                                                    'по второму каналу
      For i = 0 To size - 1
         massiv3(i) = massiv(i) + massiv2(i)                ' суммируем временные отсчеты
                                                            'двух каналов
      Next i
      Myerr = SRV1.PutData(0, MyTime, size, massiv3(0))     ' кладем данные
      MyTime = MyTime + interval                ' увеличиваем время на 0.1 секунду
      Label1.Caption = massiv(0)                ' выводим на экран значение одного отсчета
      Myerr = Grid1.Paint(massiv3(0))  ' выводим график сигнала
   End If
End If
End Sub
```

Внешний вид программы (Рисунок 3.1)

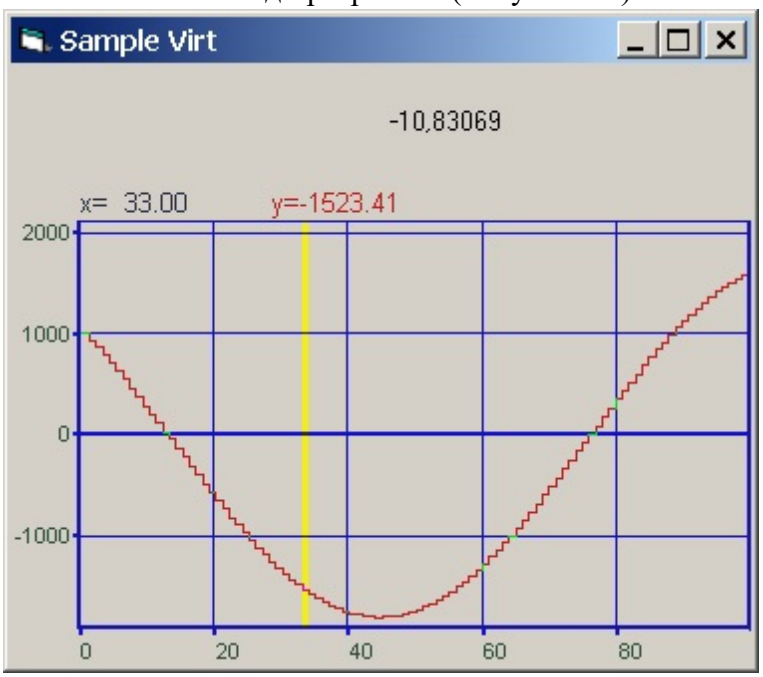



Рисунок 3,1

Для просмотра и анализа полученного сигнала можно использовать программы ZETLab осциллографы, анализаторы, вольтметры. Созданный виртуальный канал будет доступен всем программам.

## 3.4.Пример SAMPLE_4

Определение параметров канала

Для того чтобы в пользовательских программах можно было задавать различные измерительные каналы, необходимо добавить следующие компоненты и функции в программу. Сначала надо добавить Combo Box (стандартная) и Scale ActiveX (ZETLab) на форму. Затем добавить в процедуре загрузки формы Load() следующий текст.

© ООО "ЭТМС" 1992-2024. Все права защищены

```
Combo1.Clear            ' очистить комбо-бокс
For i = 0 To SRV1.QuanChan() - 1              ' количество каналов
   Combo1.AddItem (SRV1.Commentary(i))   ' добавить в комбо-бокс
Next i
Combo1.ListIndex = 0 ' индекс указателя канала
```

В обработке сигналов Timer() необходимо добавить

```
Channel = Combo1.ListIndex                    ' номер канала АЦП из списка
HorScale1.Amplitude = SRV1.CurLevel(Channel)       ' отображение общего уровня сигнала
```

# Глава 4.Установка компонента Grid.ocx

Для работы с компонентом необходимо установить его в проект:
В проекте VISUAL C++ v6.0:

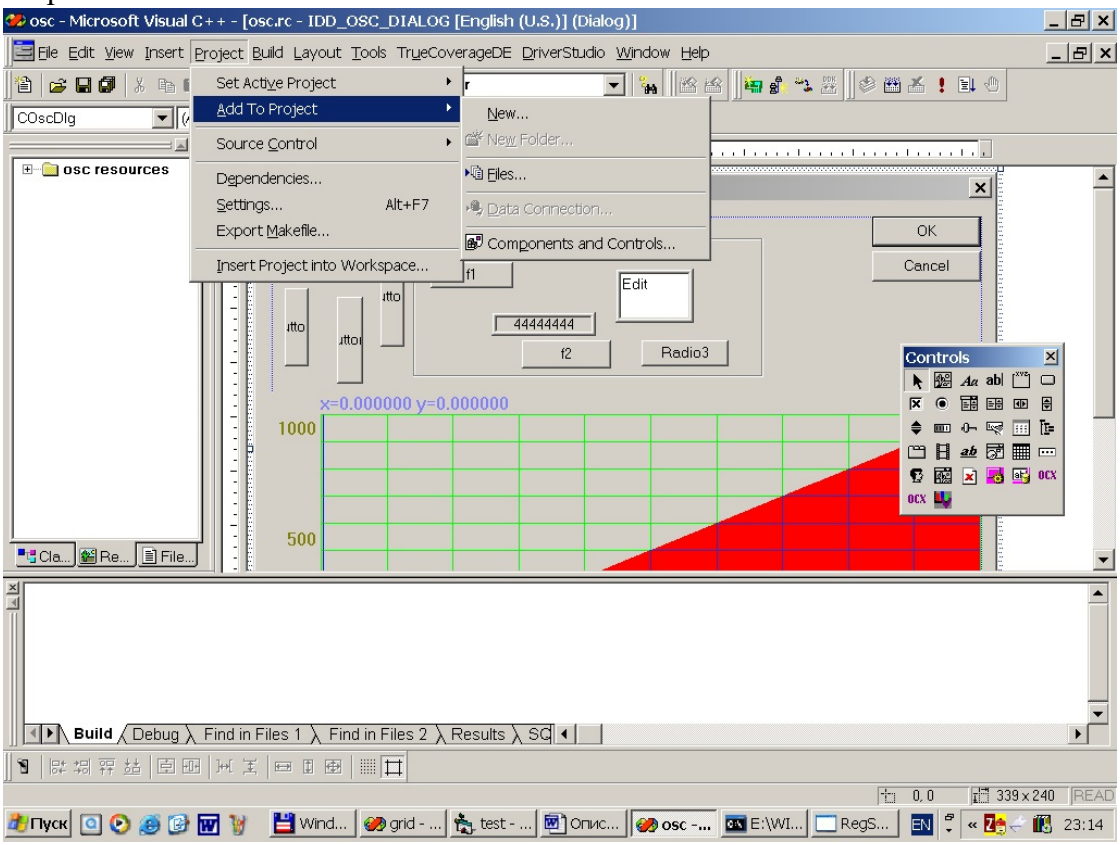
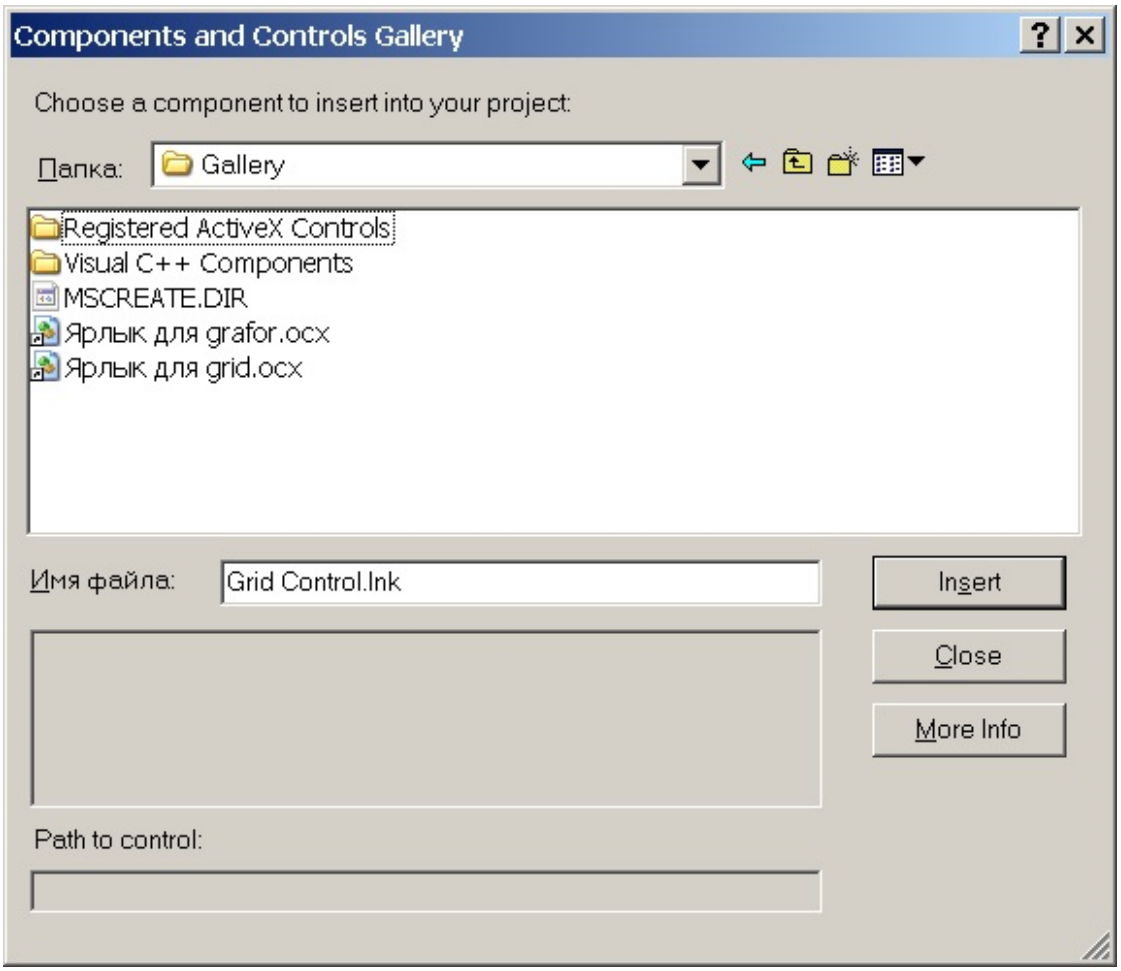



Рисунок 1.1

После выбора в меню
Project
Add To Project ->
Components and Controls
Появится окно:

© ООО "ЭТМС" 1992-2024. Все права защищены




В этом окне в папке зарегистрированных ActiveX Controls необходимо найти компонент GRID.OCX и выбрать его. После этого компонент появится в окне Controls при проектировании диалоговых окон.

В проекте VISUAL BASIC

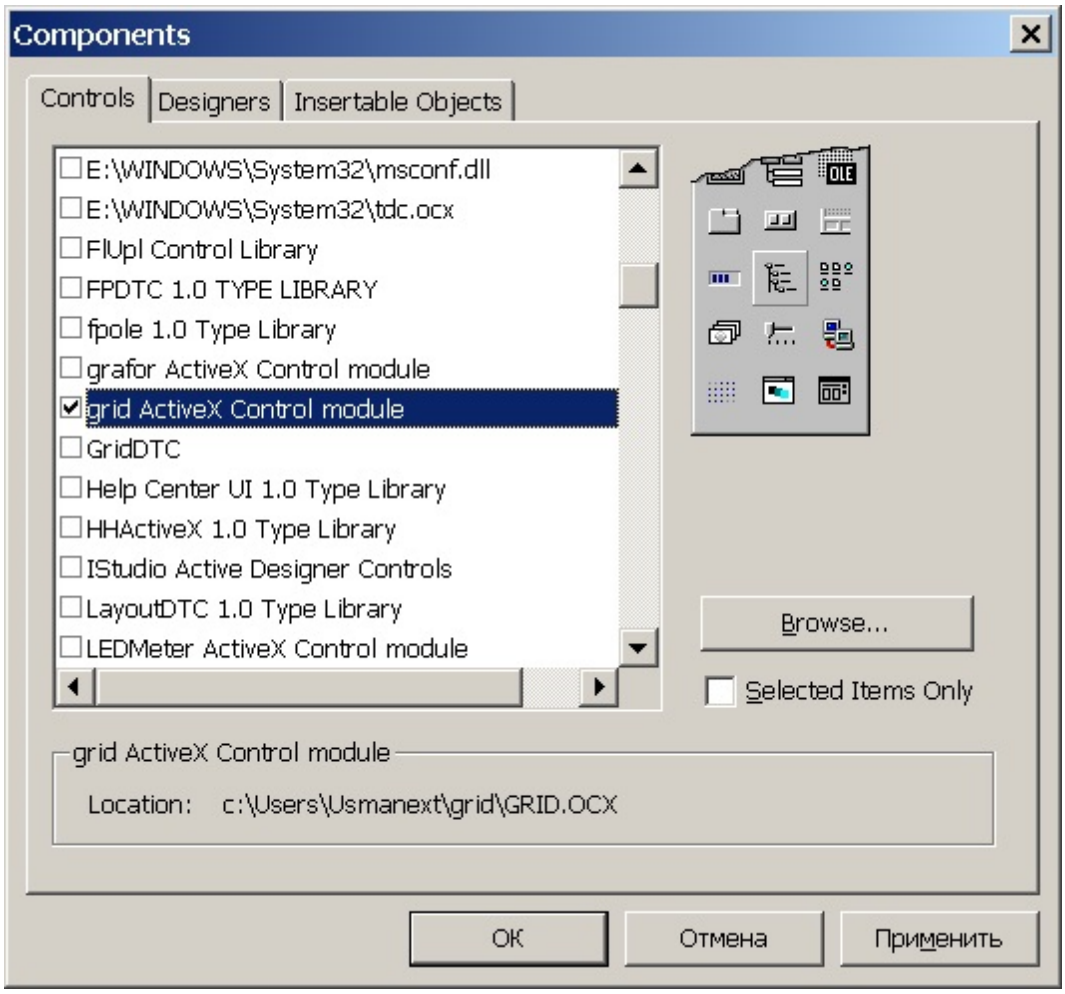
После выбора в меню

Project->

Components…

появится окно

© ООО "ЭТМС" 1992-2024. Все права защищены




Необходимо выбрать необходимый компонент и установить напротив него знак ✓.

После этого ActiveX-элемент появится на поле компонентов.

Следует отметить, что компонент автомасштабируется при изменении его размеров, поэтому необходимо следить за тем, чтобы высота компонента в приложении была не менее 80 пикселей, иначе он может отображаться некорректно.

# Глава 5.Примеры работы с VisualStudio2010

Примеры находятся на сервере по адресу: \\192.168.0.7\Setup ZET\Setup_for_diskWriting\Doc\Samples\VisualStudio2010

или на поставляемым диском: Doc\Samples\VisualStudio2010

© ООО "ЭТМС" 1992-2024. Все права защищены

# Глава 6.Примеры программирования на .Net платформе

В данной статье на примере компонента ZETServer Control (SRV.ocx) проведем интеграцию ActiveX компонента Zet, с Вашей программой.

## 6.1.Общие сведения о подключении компонентов к C#

Для использования ActiveX необходимо несколько условий:

1. Элемент OCX должен быть зарегистрирован.
   - Если вы использовали установочный диск ZETLab, то необходимость в регистрации компонента отпадает, поскольку все входящие в комплект компоненты регистрируются автоматически.
   - При использовании дополнительных компонентов, необходимо их зарегистрировать. Путем использования утилиты RegSvr32, входящей в стандартный комплект операционной системы Windows, либо с применением тестового контейнера ActiveX, непосредственно, из Visual Studio. (tools -> ActiveX Test Container -> File -> Register Controls… -> в появившемся диалоговом окне нажать кнопку “Register…” и выбрать необходимый для регистрации файл с расширением .ocx).
2. Подключение ActiveX компонента к проекту C#.

В Visual Studio C# возможны несколько способов подключения ActiveX компонента к уже существующему проекту.

Приведем их:

- С панели инструментов ToolBox.
- С помощью применения специальных .dll файлов – ActiveX Wrapper (Обертка) созданных на основе существующего .ocx компонента. Для создания обертки существует утилита aximp, входящая в состав SDK Visual Studio.

При использовании обоих подходов итог один — во вкладке *References* вашего проекта C# появляются два файла AxSrvLib.dll и SrvLib.dll.

Программист должен обратить особое внимание именно на первый файл — AxSrvLib.dll, так как в нем находятся рабочее пространство и классы ZETServer Control. Содержимое файла можно посмотреть с помощью Object Browser (В Visual Studio, в Solution Explorer -> References -> AxSrvLib -> AxSrv -> смотреть содержимое).

Именно AxSrv представляет класс внедряемого компонента. Смотри Рисунок 1.1.

© ООО "ЭТМС" 1992-2024. Все права защищены

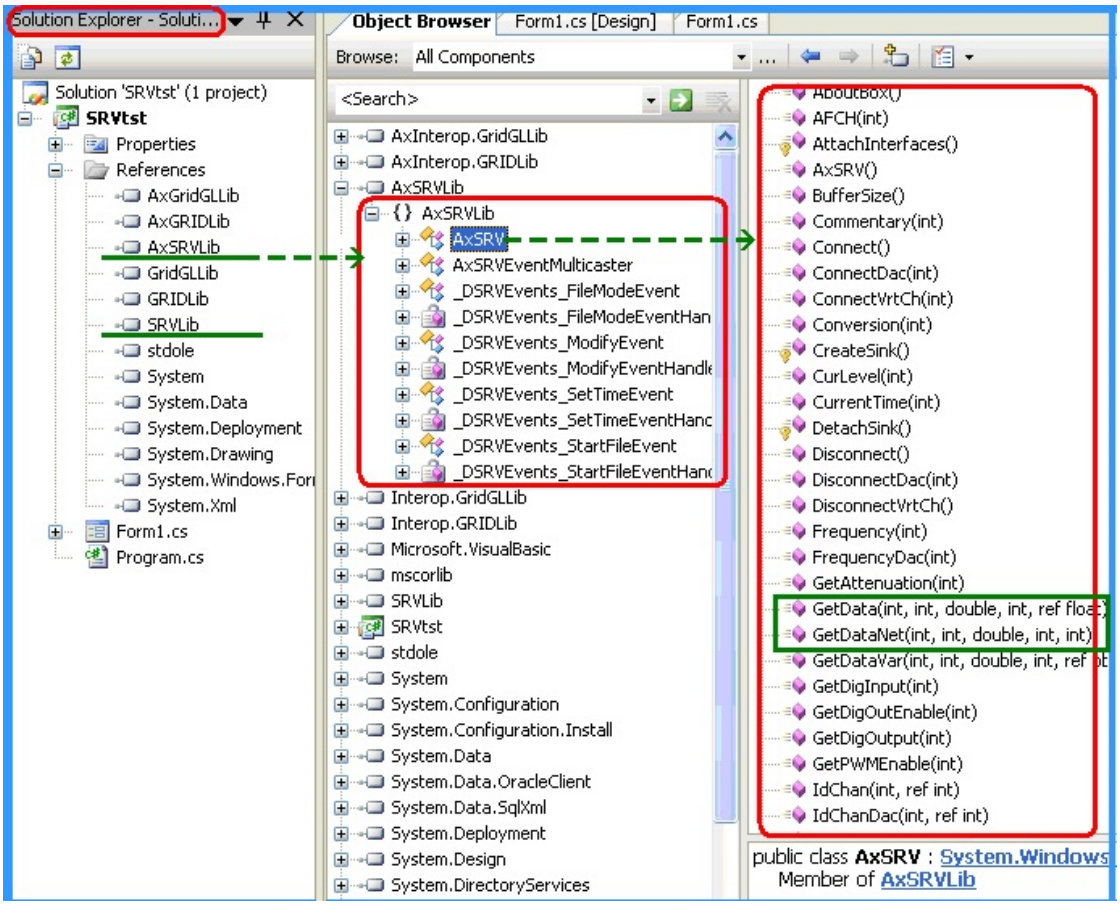



Рисунок 1.1. Представление осх файлов в виде dll и их содержимое

## 6.2.Подключение компонента ActiveX к проекту C#

Как уже говорилось, для применения компонента в Вашем проекте помимо регистрации необходимо его подключение к проекту.

Для пользователя наиболее простой способ – применение панели инструментов ToolBox.

Более сложный способ — создание вышеупомянутых файлов .dll вручную, с применением утилиты aximp, входящей в состав SDK (Source Development Kit). Утилита позволяет выбрать необходимый вам .осх файл и создать из него ActiveX Wrapper, “обертку”.

**Использование панели ToolBox**

Опишем основные шаги:

1. Открываем панель ToolBox. Если панели на дисплее не видно необходимо ее подключить. Для этого надо вызвать ее из меню View -> ToolBox, либо нажать комбинацию (Ctrl+Alt+X).
2. Щелкнуть правой кнопкой мыши на одной из вкладок со стандартными элементами управления VS, либо создать свою, новую вкладку — например “Zet Controls”, путем выбора из всплывающего меню кнопки Add

© ООО "ЭТМС" 1992-2024. Все права защищены

Tab. Путем выбора пункта меню Choose Item… вызвать диалоговое окно Choose ToolBox Items.

3. В данном диалоговом окне выбрать вторую вкладку – Com components. В ней хранятся ссылки на все зарегистрированные в системе ActiveX компоненты, в том числе и Zet компоненты. После выбора данной вкладки возможна задержка на некоторое время, связано это с тем, что VS опрашивает си систему в поисках зарегистрированных компонентов.
4. Выбрать из списка SRV Control. Необходимо отметить компонент галочкой и нажать кнопку OK. Рисунок 1.2.
5. После чего компонент появится в ToolBox. Рисунок 3'.
6. Для использования компонента на форме необходимо перетащить компонент с панели ToolBar на форму, при этом форма должна быть в режиме дизайна (View Design).
7. После Данных манипуляций в ваш проект будет подключен компонент, с именем класса AxSrv. Смотри рисунок 1.3. AxSrvLib и SrvLib.
8. О дальнейших действиях с подключенным компонентом, а так же пример использования будут приведены в следующих главах.

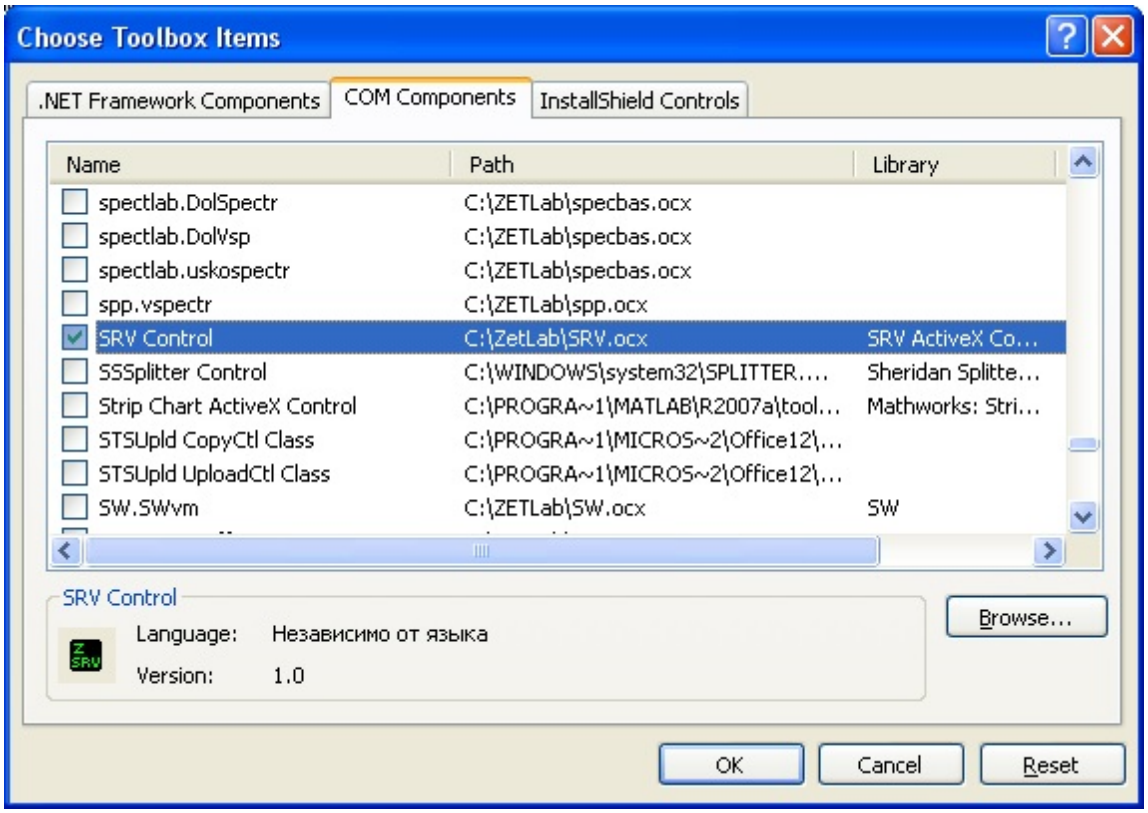



Рисунок 1.2. Подключение компонента

© ООО "ЭТМС" 1992-2024. Все права защищены

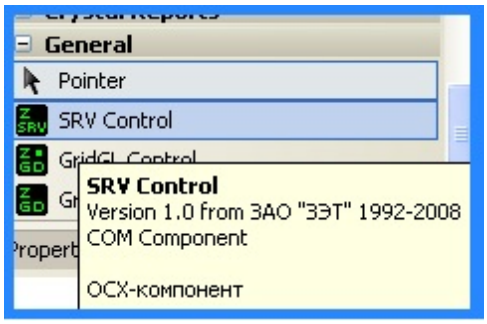



Рисунок 1.3. Внешний вид ToolBox с подключенным компонентом ZETServer.

**Создание ActiveX Wrapper вручную**

При создании ActiveX Wrapper вручную нужно провести более сложные манипуляции. Прежде чем приступить к процессу подключения настроим среду Visual Studio.

**Настройка среды Visual Studio.Net 2005. Подключение утилиты aximp**

Утилита aximp — специальная программа, которая умеет делать обертку ActiveX.

Обычно, по умолчанию утилита устанавливается по следующему пути —C: \Program Files\Microsoft Visual Studio 8\SDK\v2.0\Bin\AxImp.exe

Для создания обертки можно ее оттуда и вызвать, но для более легкого применения данной программы, возможно ее подключение к среде Visual Studio, в главном меню в Tools. Рисунок 1,4.

© ООО "ЭТМС" 1992-2024. Все права защищены

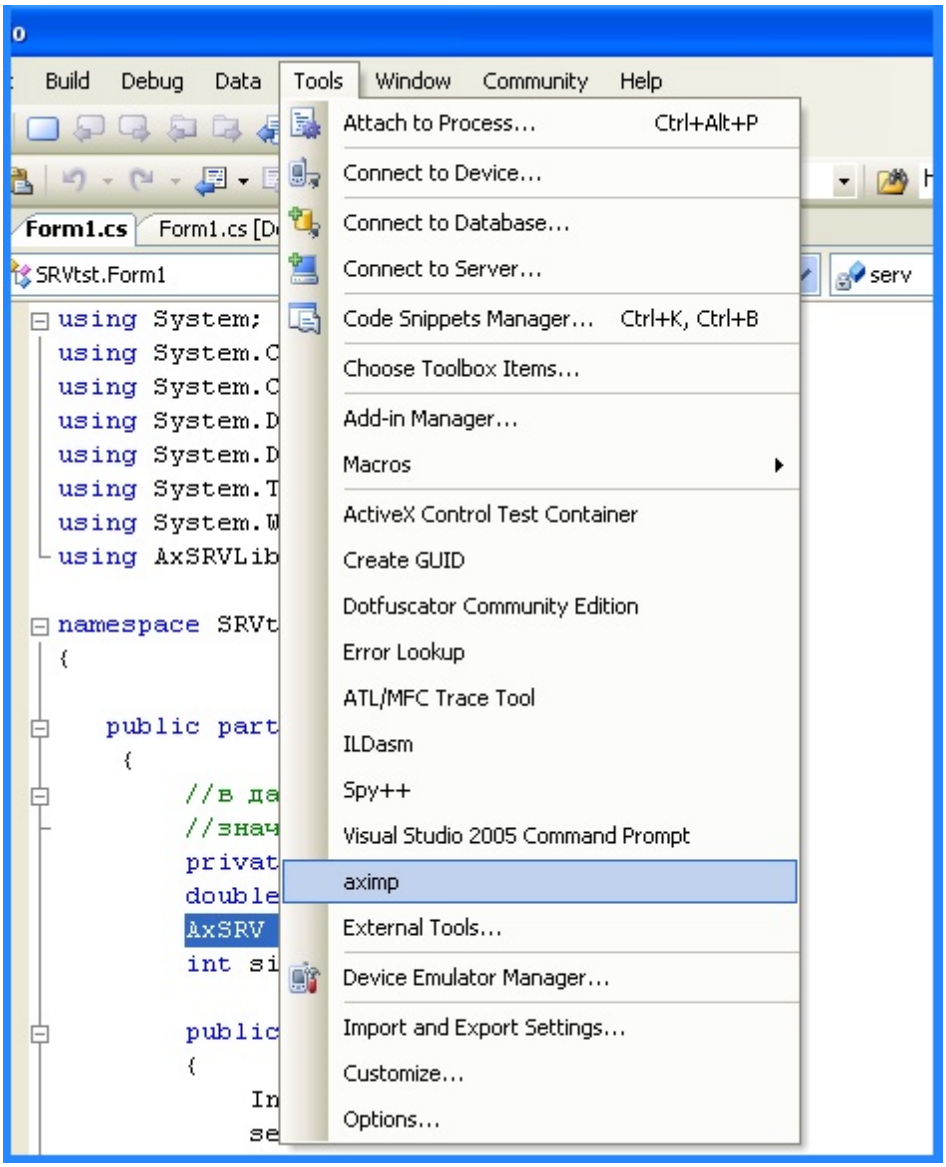



Рисунок 1.4. Внешний вид меню Tools после подключения утилиты aximp.

**Подключение Aximp:**

1. Tools->External Tools… -> Появится диалоговое окно в котором можно добавить новый элемент для меню Tools.
2. Добавить новый элемент кнопкой Add
3. Ввести следующие параметры, смотри рисунок 1.5:
   - Title: aximp
   - Command: C:\Program Files\Microsoft Visual Studio 8\SDK\v2.0\Bin\AxImp.exe (Путь для Aximp.exe по умолчанию).
   - Initial: $(ProjectDir) (Данный параметр не надо набивать с клавиатуры, он выбирается из ниспадающего списка справа от поля).
4. Теперь вы можете вызывать ее из прямо из меню Tools.

© ООО "ЭТМС" 1992-2024. Все права защищены

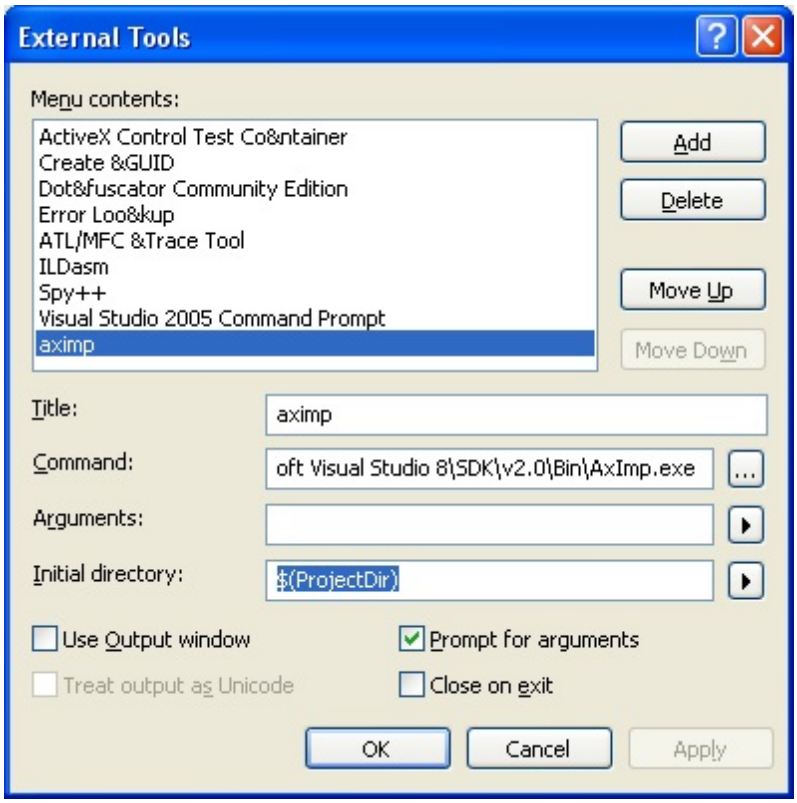



Рисунок 1.5. Добавление Утилиты aximp.exe

**Создание обертки и подключение компонента**

После настройки среды Visual Studio 2005 необходимо приступить, непосредственно, к созданию компонента.

Шаги:

1. Tools -> aximp -> появится диалоговое окно утилиты aximp. Рисунок 1.6.
2. В поле Arguments надо ввести путь к кайлу. В данном случае это путь к файлу srv.ocx. Нажать OK.
3. Появится консоль утилиты aximp, выполняющая создание обертки. Рисунок 1,7.
4. В итоге, в корневом каталоге Вашего проекта возникнут два файла dll, которые необходимо подключить к проекту. Рисунок 8.
5. Щелкнуть в Solution Explorer по вкладке References -> Add Reference… -> Выбрать Вкладку Browse -> Выделить оба файла -> нажать кнопку OK.
6. К проекту будут добавлены оба файла. Смотри рисунок 1. AxSrvLib.dll и SrvLib.dll.

© ООО "ЭТМС" 1992-2024. Все права защищены

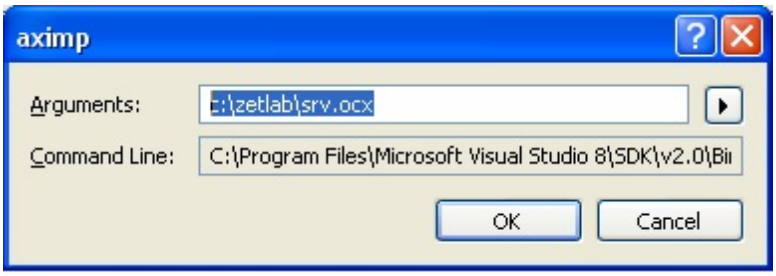



Рисунок 1.6. Создание обертки ActiveX Wrapper.

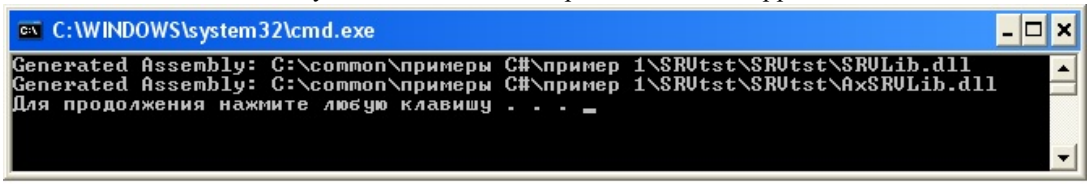



Рисунок 7. Работа утилиты Aximp.

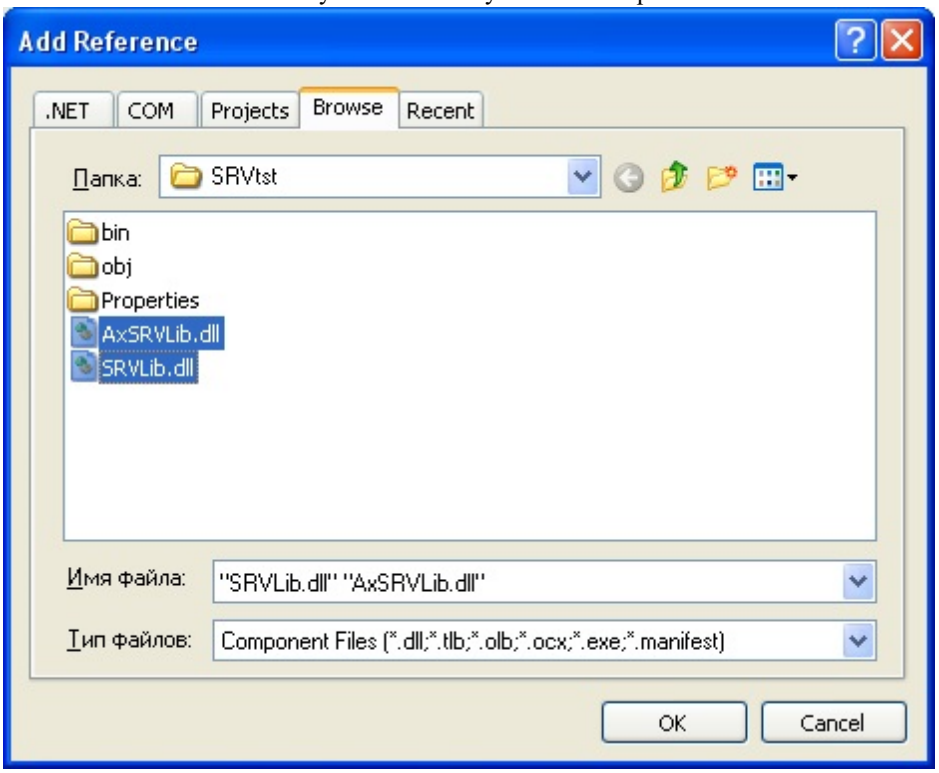



Рисунок 1.8. Добавление созданных файлов к проекту.

**Подключение компонентов ActiveX в C#. Резюме**

Как у первого, так и у второго способа есть свои преимущества и недостатки. Например, второй способ проигрывает первому в простоте использования. Но подключение компонента с помощью создания обертки компонента вручную более выгоден разработчику ActiveX компонентов, если, помимо, Наших компонентов вы захотите использовать создаваемые лично Вами.

Поскольку, в Visual Studio есть некоторые сложности при обновлении библиотеки типов, и в этом способе, после обновления, или, после перекомпиляции внедряемого ActiveX компонента, надо, просто, один раз воспользоваться уже добавленной в меню

© ООО "ЭТМС" 1992-2024. Все права защищены

Tools утилитой aximp. А поскольку файлы AxSrvLib и SrvLib уже были добавлены, еще раз добавлять их не надо, поскольку в проекте указывается лишь ссылка на существующие компоненты.

## 6.3.Использование компонентов ActiveX Zet в тексте программы C#

После добавления в проект всех необходимых компонентов вы имеете класс компонента. В данном случае, относительно сервера — это AxSrv. На рисунке 1 в центре сам класс, справа — функции, доступные при работе с компонентом.

При создании нового проекта C# Code Wizard создает следующий программный код:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;

namespace WindowsApplication1
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }
    }
}
```

**Листинг** 1. Текст программы, генерируемый с помощью Code Wizard.

Для использования компонента надо добавить к коду формы добавить пространство имен ActiveX компонента. Следует заметить, что при использовании ToolBox этого делать не надо, поскольку данная строчка записывается автоматически.

```
using AxSRVLib;
```

Для обращения к компоненту надо объявить его и создать объект. Назовем его SRV. Добавлять его необходимо следующим образом:

```
   public partial class Form1 : Form
      {
            …
            AxSRV serv;
      …
            public Form1()
            {
                  InitializeComponent();


                //при использовании ToolBox этого тоже деать не надо
```

© ООО "ЭТМС" 1992-2024. Все права защищены

```
            serv = new AxSRV();
            serv.Parent = this;
            Controls.Add(serv);
            //-------------------------------------------------

        }
    }
```

**Листинг** 2.

На этом можно приступить к полноценной работе с компонентом Zet Srv Control. Теперь можно пользоваться функциональностью компонента и всеми его возможностями, например, такими как получение сигналов с линий АЦП подключенного прибора, либо создание своих виртуальных каналов.

## 6.4.Особенности работы с указателями на массив на .Net платформе

Следует отметить, то при работе в .net языках функции, в которых в качестве параметров выступают указатели на массив, например GetData в .net языках работают некорректно, поэтому следует использовать обновленные функции, имеющие такое же название + приписка Net, например, GetDataNet.

При этом вызов подобных функций следует использовать в коде unsafe. В настройках параметров проекта необходимо поставить галочку Project->Properties->Build->allow unsafe code.

Чтоб в этом разобраться Вам поможет листинг 3. В этом примере по таймеру идет опрос сервера.

```
private void timer1_Tick(object sender, EventArgs e)
 {
     int Channel = 0; //номер канала АЦП
     //смотрим текущее значение времени сервера по выбранному каналу
     double ServerTime;
     ServerTime = serv.CurrentTime(Channel);

     //сравниваем текущее время
     if (MyTime < serv.CurrentTime(Channel))
     {
         unsafe
         {

             float* p = stackalloc float[size];

             serv.GetDataNet(Channel, 0, MyTime, size, (int)p);

             //увеличиваем счетчик времени
             MyTime = MyTime + 0.12;
             //выводим на экран значение одного отсчета
             dataVal.Text = p[0].ToString();

             //вывод полученных значений на график
             axGridGL1.PaintNet((int)p);
```

© ООО "ЭТМС" 1992-2024. Все права защищены

```
                    axGrid1.PaintNet((int)p);
                }
            }
        }
```

**Листинг** 3. Применение *.*Net функций и указателей на массив в коде unsafe.

## 6.5.Пример программы опроса компонента ZETServer на C#

Далее будет приведен пример программы с использованием трех компонентов ZETLab:

- SRV,
- Grid,
- GridGL.

Притом, компонент Server Control добавлен в проект ручным способом, с помощью утилиты aximp, и добавлением в References проекта. А компоненты Grid и GridGL добавлены из ToolBox.

В программе производится опрос нулевой линии подключенного к компьютеру прибора, в данном случае, какого именно, роли не играет, поскольку работа с любыми каналами любых приборов ZET унифицирована. После опроса АЦП происходит вывод информации на график.

Опрос АЦП происходит самым простым способом — по таймеру.

Проект приведенной программы на Visual Studio поставляется с ZETLab Studio и находится по следующему пути:

c:\zetlab\doc\zetlabstudio\Samples\Примеры C#\Пример 1

Листинг приведен ниже.

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using AxSRVLib;

namespace SRVtst
{

   public partial class Form1 : Form
    {
        //в данной переменной хранится текущее
        //значение времение по каналу в ZetSrv
        private double MyTime;
        double delta1;
        AxSRV serv;
        int size = 10000;
```

© ООО "ЭТМС" 1992-2024. Все права защищены

```
        public Form1()
        {
            InitializeComponent();
            serv = new AxSRV();

            serv.Parent = this;

            Controls.Add(serv);
        }

        private void button1_Click(object sender, EventArgs e)
        {
            //возвращаемая ошибка, при попытке подсоединиться к серверу
            long MyError;
            MyTime = 0;
            MyError = serv.Connect();
            MyTime = serv.CurrentTime(0);
            timer1.Interval = 100;
            timer1.Enabled = true;
        }
        private void button2_Click(object sender, EventArgs e)
        {
            timer1.Enabled = false;
            long MyError;
            MyError = serv.Disconnect();
        }
        private void Form1_FormClosing(object sender, FormClosingEventArgs
e)
        {
            timer1.Enabled = false;
            long MyError;
            MyError = serv.Disconnect();
        }
        private void timer1_Tick(object sender, EventArgs e)
        {
            //номер канала АЦП
            int Channel = 0;
            //выводим текущее значение времени сервера по выбранному каналу
            double ServerTime;
            ServerTime = serv.CurrentTime(Channel);
            srvTimelbl.Text = ServerTime.ToString();
            //сравниваем текущее время

            if (MyTime < serv.CurrentTime(Channel))
            {
                unsafe
                {

                    float* p = stackalloc float[size];

                    serv.GetDataNet(Channel, 0, MyTime, size, (int)p);

                    //увеличиваем счетчик времени
```

© ООО "ЭТМС" 1992-2024. Все права защищены

```
                    MyTime = MyTime + 0.12;
                    prgTimelbl.Text = MyTime.ToString();
                    //выводим на экран значение одного отсчета
                    dataVal.Text = p[0].ToString();

                    //вывод полученных значений на график
                    axGridGL1.PaintNet((int)p);
                    axGridl.PaintNet((int)p);

                }

            }

        }

    }
}
```

**Листинг** 4. Пример программы на C#

# Глава 7.Пример управления генератором Dac

**Программа формирования на выходе генератора синусоидального сигнала заданной частоты и уровня СКЗ.**

**На сайте** http://www.zetlab.ru/catalog/programs/zetlab_studio/zetserver.php?sphrase_id=15968

**на сереверe (для сотрудников)** \\192.168.0.7\Setup ZET\Setup_for_diskWriting\Doc\Samples\VisualStudio2010

**на поставляемым диске**: Doc\Samples\VisualStudio2010

---

## *Пример текста программы GeneratorSinus на Visiual C++ 10*

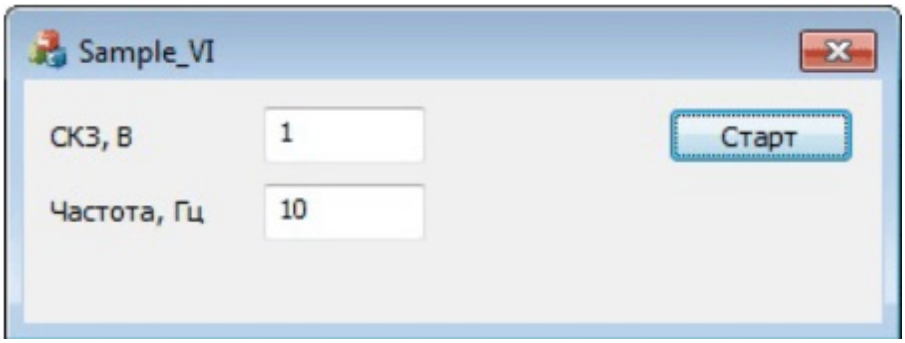



Рисунок 2.1

Данный пример непосредственно в ZETLab

© ООО "ЭТМС" 1992-2024. Все права защищены

# Глава 8.Примеры работы с драйверами ZET

Работа с модулями АЦП-ЦАП или анализаторами спектра возможна с помощью вызовов функций библиотеки Zadc.dll. При этом, пользователь свободен в выборе языка программирования. Примеры Test_DAC и Test_Zadc, написанные на Visual C++ 6, Visual Basic 6, Borland C++ Builder 6 и Delphi 2007 демонстрируют работу с библиотекой Zadc.dll

Программа *Test_Zadc* включает два канала **АЦП** и отображает мгновенное амплитудное значение по двум каналам 2 раза в секунду.
Программа *Test_DAC* включает два канала **ЦАП** и отображает мгновенное амплитудное значение по двум каналам 2 раза в секунду.

## Примеры иллюстрируют следующие возможности библиотеки

**для АЦП**

- подключение к драйверу;
- проверка поддержки АЦП;
- опрос количества каналов АЦП;
- включение/выключение каналов АЦП;
- опрос коэффициентов усиления по каналу АЦП;
- опрос веса младшего разряда АЦП по каналу;
- опрос количества включенных каналов АЦП;
- опрос количества слов в одном отсчете АЦП;
- запрос буфера АЦП;
- остановка АЦП;
- запуск АЦП;
- отключение от драйвера;
- освобождение буфера АЦП;
- запрос текущего значения указателя в буфере;

**для ЦАП**

- проверка поддержки ЦАП;
- включение/выключение каналов ЦАП;
- опрос количества включенных каналов ЦАП;
- опрос веса младшего разряда ЦАП;
- опрос количества слов в одном отсчете ЦАП;
- опрос частоты дискретизации ЦАП;
- опрос размера буфера прерывания ЦАП;
- установить/опросить коэффициент затухания аттенюатора по каналу ЦАП;
- запрос буфера ЦАП;
- остановка ЦАП;
- запуск ЦАП;
- освобождение буфера ЦАП.

**Примеры находятся по адресу:**

**на сайте** http://www.zetlab.ru/catalog/programs/zetlab_studio/test_drivers.php?sphrase_id=15965

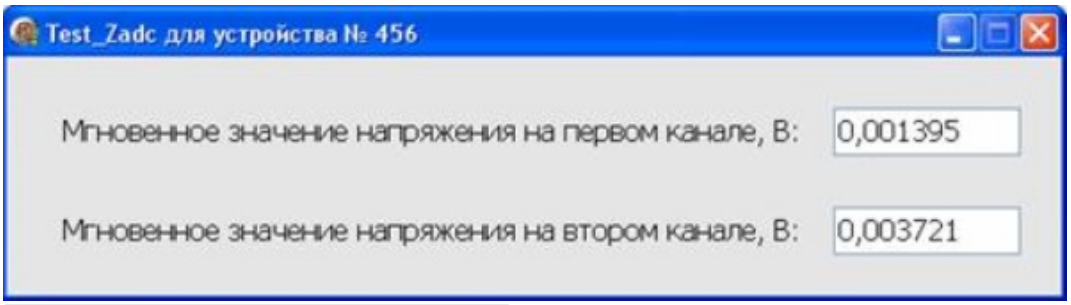
© ООО "ЭТМС" 1992-2024. Все права защищены

**на серевере (для сотрудников)** \\192.168.0.7\Setup ZET\Setup_for_diskWriting\Doc\Doc\Samples\drivers

**на поставляемым диске**: Doc\Samples\drivers

---

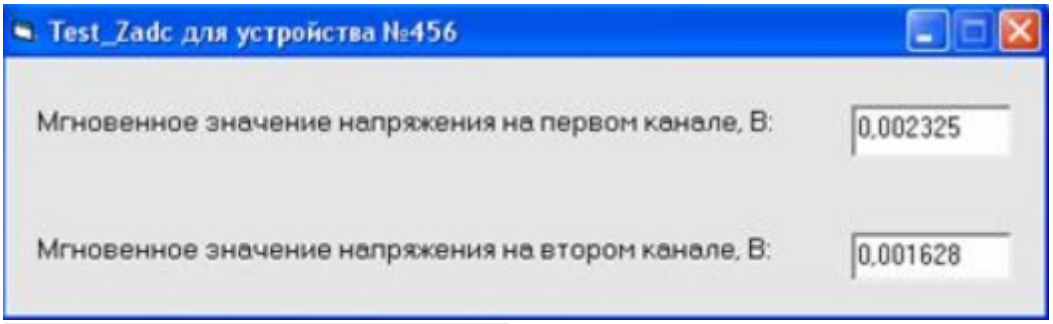
## *Пример текста программы Test_Zadc на Delphi 2007*




Данный пример непосредственно в ZETLab

---

## *Пример текста программы Test_Zadc на Visual Basic 6*

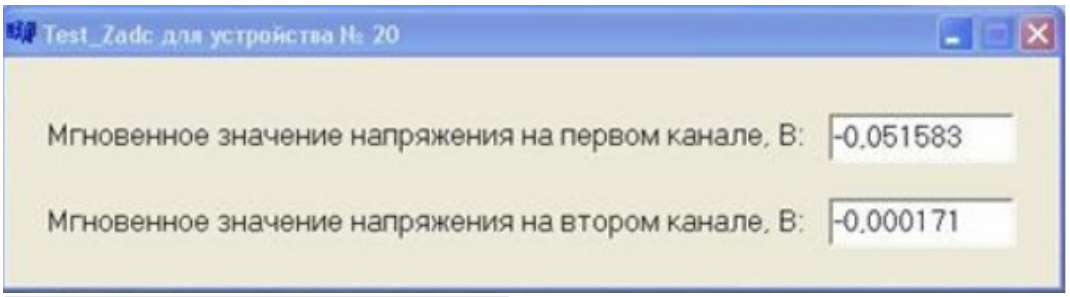



Данный пример непосредственно в ZETLab

---

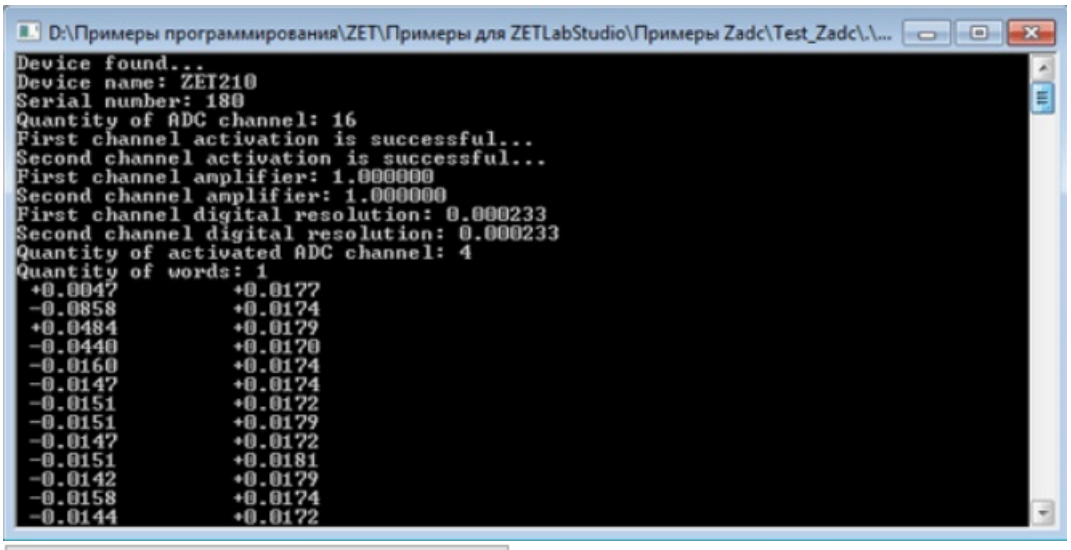
## *Пример текста программы Test_Zadc на Borland C++ Builder 6*

© ООО "ЭТМС" 1992-2024. Все права защищены



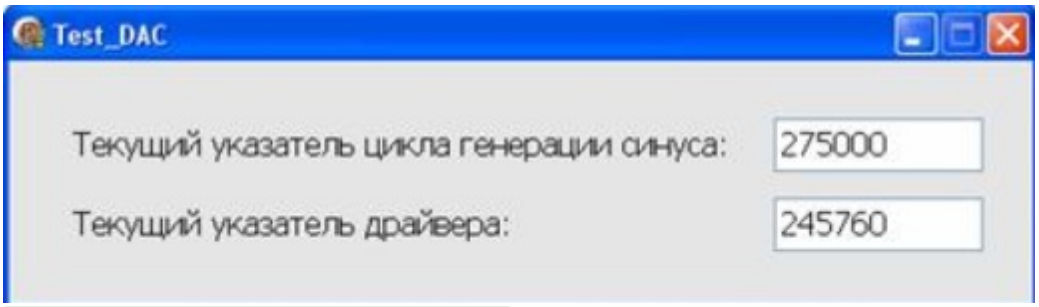

Данный пример непосредственно в ZETLab

## *Пример текста программы Test_Zadc на Visual C++ 10*




Данный пример непосредственно в ZETLab

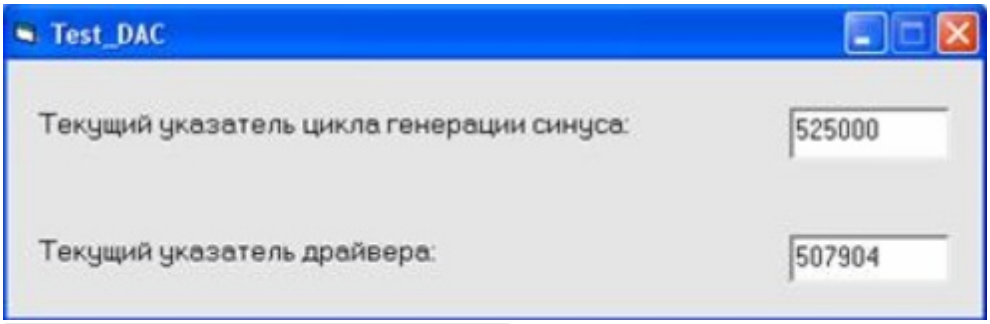
## *Пример текста программы Test_Dac на Delphi 2007*




Данный пример непосредственно в ZETLab

© ООО "ЭТМС" 1992-2024. Все права защищены

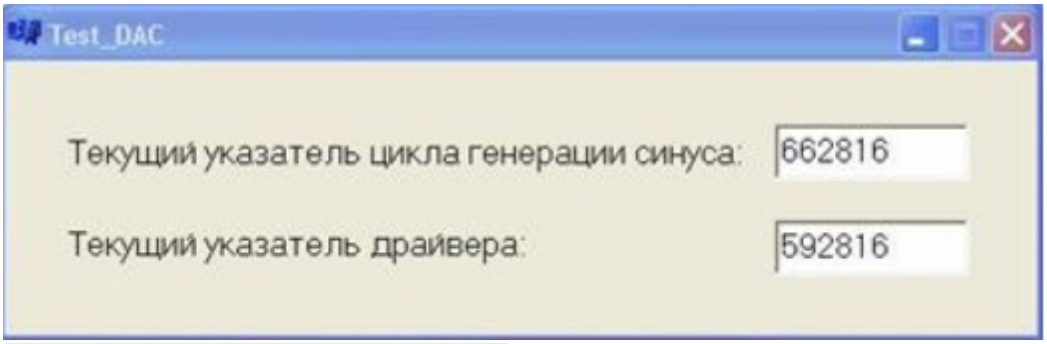
### *Пример текста программы Test_Dac на Visual Basic 6*




Данный пример непосредственно в ZETLab

---

### *Пример текста программы Test_Dac на Borland C++ Builder 6*

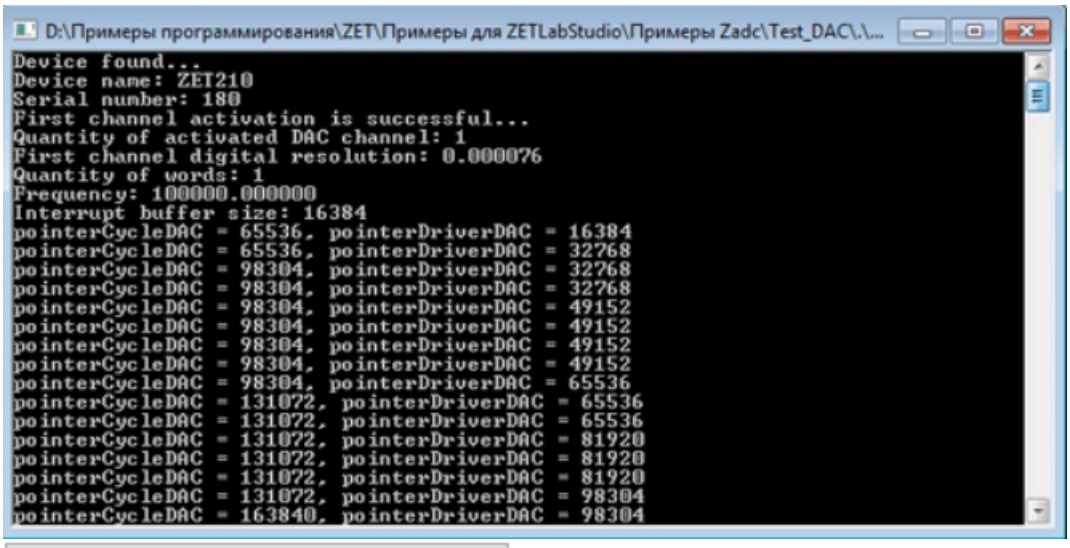



Данный пример непосредственно в ZETLab

---

### *Пример текста программы Test_Dac на Visual C++ 10*

© ООО "ЭТМС" 1992-2024. Все права защищены




Данный пример непосредственно в ZETLab

# Глава 9.Примеры использования Unit.ocx (на MS Visual Basic 6.0)

## Unit - модуль управления и автоматизации

Данный модуль предназначен для управления и автоматизации процесса измерений при построении различных технологических программно-аппаратных комплексов на базе модулей АЦП ЦАП и анализаторов спектра.

Пользователю предоставляется возможность реализации любого алгоритма работы программ средств измерений. На любом удобном объектно-ориентированном языке программирования (MS Visual Basic, MS Visual C++, Borland Delphi, Borland C++ Builder *) пользователь создает графическую оболочку, обеспечивающую требуемый интерфейс. Затем, используя модуль управления и автоматизации Unit активизируются необходимые виртуальные приборы, и посредством команд пользовательская программа получает полный доступ к операциям, функциям и данным используемого виртуального прибора.

* При программировании на Borland Delphi и Borland C++ Builder обеспечивается ограниченная функциональность по причине особенностей данных языков программирования.
Т.о., определив сценарий работы измерительной системы и сформировав его в виде команд, пользователь имеет возможность автоматизировать процесс получения и обработки входных/выходных данных с модулей АЦП ЦАП и анализаторов спектра.

© ООО "ЭТМС" 1992-2024. Все права защищены

Данный модуль обеспечивает гибкость при построении собственных измерительных систем и, в то же время, сохраняет метрологическую целостность комплекса.

## 9.1.Программа "Узкополосный спектр"

**Пример находится на сайте: http://www.zetlab.ru/catalog/programs/zetlab_studio/unit_and_spectr.php?sphrase_id=15964 (первый пример)**
**или http://www.zetlab.ru/catalog/programs/zetlab_studio/unit_and_spectr.php (второй пример)**

Программы SpectrTest, написанные на разных языках программирования, демонстрируют совместную работу нескольких ActiveX компонентов, входящих в состав средств разработчика ZETLab Studio.

Используемые компоненты:

- GridGL.ocx;
- Unit.ocx;
- программа “Узкополосный спектр”.

Примеры показывают, как обеспечить взаимодействие между компонентами на языках программирования C++, C# и Basic в среде разработки VS 2005.

В каждой программе реализован следующий алгоритм:

1. Unit запускает программу Spectr.exe.
2. Далее, порционно (по готовности Spectr.exe), Unit считывает массив, содержащий частотную сетку и спектр.
3. В зависимости от количества данных в массиве настраивается сетка GridGL.
4. С помощью GridGL отображается график частотного спектра.

Результат работы каждой программы будет одинаков, вне зависимости от выбранного языка программирования. На рисунке 1 справа представленно окно непосредственно программы SpectrTest, а слева - окно запущенной из нее программы Spectr.exe.

© ООО "ЭТМС" 1992-2024. Все права защищены

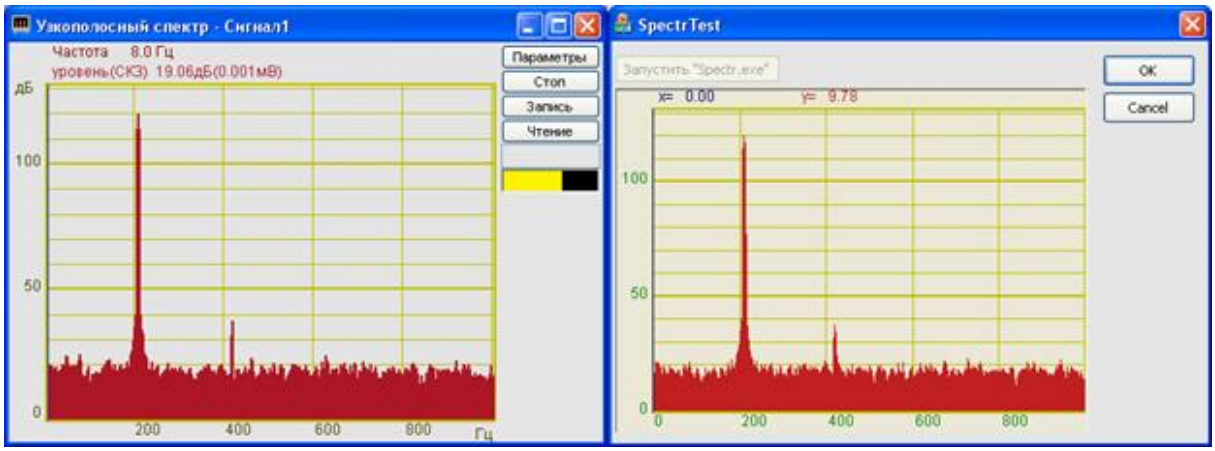



Рисунок 1

# Программа SpectrTest на Visual Studio C++

Ниже приведен фрагмент программы SpectrTest на VS C++

При создании программы на форму диалогового окна (рисунок 2) были добавлены компоненты GridGL.ocx для отображения спектра, Unit.ocx для обеспечения запуска и управления программой Spectr.exe и кнопка Button, для запуска пользователем программы Spectr.exe.

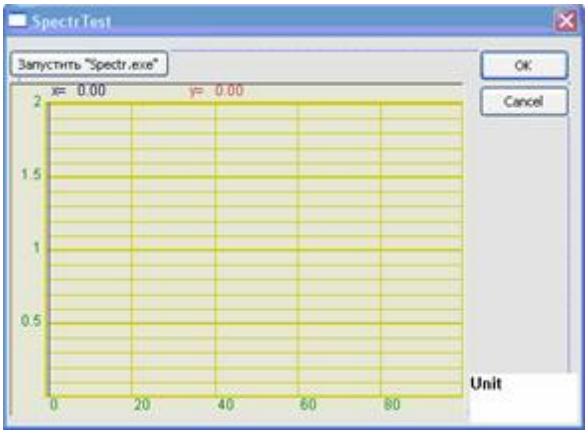


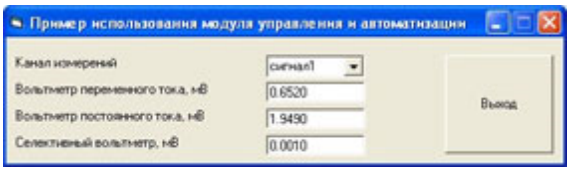

Рисунок 2

Данный пример непосредственно в ZETLab

## 9.2.Программа "Три вольтметра - в одном"

**Пример находится на сайте: http://www.zetlab.ru/catalog/programs/zetlab_studio/unit.php#2**

© ООО "ЭТМС" 1992-2024. Все права защищены




Во время запуска данной программы параллельно в скрытом режиме запускаются еще 3 программы: вольтметр постоянного тока, вольтметр переменного тока и селективный вольтметр переменного тока. Пользователь выбирает канал модуля АЦП или анализатора спектра и в реальном времени в окне программы отображаются показания всех вольтметров по выбранному каналу.

Последовательность действий при создании программы:

1. Создаем новый проект "Standart EXE";
2. Через меню "Project -> Components" добавляем в проект два компонента: "SRV ActiveX Control Module" и "Unit ActiveX Control Module";
3. Размещаем на форме один компонент "SRV ActiveX Control Module" и три компонента "Unit ActiveX Control Module";
4. Добавляем и размещаем на форме необходимые элементы (Label, TextBox, ComboBox, EditBox, Button) (см. рисунок);
5. Обрабатываем нажатие клавиши и сообщения от модуля управления:

Данный пример непосредственно в ZETLab

# Глава 10.Примеры подключения ZETSENSOR к сторонним системам

## 10.1.Взаимодействие с ZETSENSOR посредством пр-мы Simply Modbus

Программа Simply Modbus позволяет сформировать запрос нашему интеллектуальному датчику (рис. 1).
Посмотреть номер COM порта или настроить скорость передачи данных можно с помощью программы "Диспетчер устройств", находящуюся во вкладке «Сервисные» панели ZETPanel.
У всех датчиков **ZET 70XX** 0x14 - это адрес в памяти датчика, содержащий значения канала (возможно лишь считывание).

© ООО "ЭТМС" 1992-2024. Все права защищены

1. Способ передачи.
2. COM порт, к которому подключен интеллектуальный датчик ZETSENSOR.
3. Скорость передачи данных по протоколу Modbus.
4. Количество бит.
5. Количество бит, которыми обозначается конец пакета передачи данных.
6. Проверка на нечетность.
7. Полученный ответ.
8. Номер ноды у 70XX.
9. Адрес регистра, данные которого будут считываться.
10. Количество и тип данных, запрашиваемого регистра.
11. Стандартная функция чтения значения.
12. Лог программы.

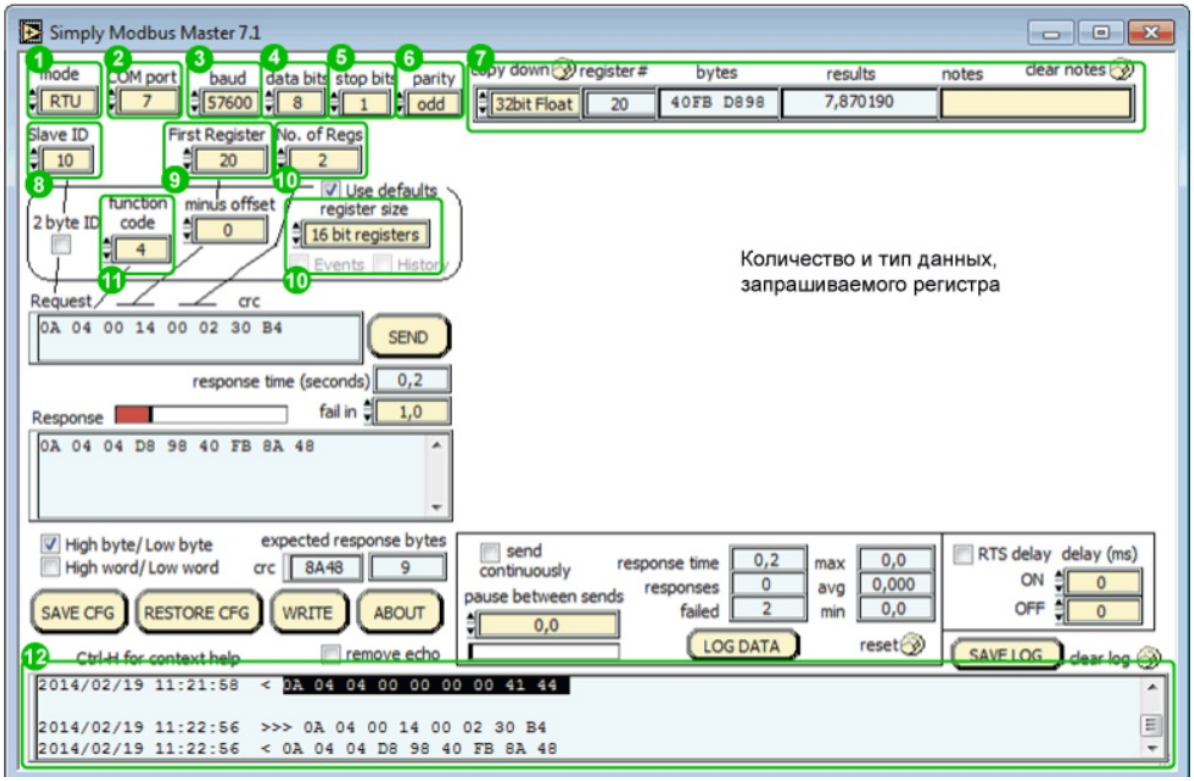



**Таблица адресов памяти**

**На сайте указана полная информация:http://www.zetlab.ru/catalog/smart_sensor/connecting-to-third-party-systems/third_party_systems_examples.php**

## 10.2.Взаимодействие с ZETSENSOR посредством программы на С

В примере показано взаимодействие с ZET 7010 посредством программы, работающей с устройством ICP CON I-7561 как с COM портом.
Пример находится в файле на сервере:\\192.168.0.7\Program release\ZetLab-Release-For-Testing-alpha\Тексты\7xxx_External.rar

© ООО "ЭТМС" 1992-2024. Все права защищены

# 10.3.Взаимодействие с ZETSENSOR посредством компонента SCADA ZETVIEW "OPC Сервер"

### Взаимодействие с ZETSENSOR посредством компонента SCADA ZETVIEW "OPC Сервер"

Сервер OPC является широко используемым в промышленной автоматизации. Он обеспечивает обмен данными между клиентской программой и физическими устройствами.

В программном обеспечении **ZETLAB** предусмотрено три типа программ, организующих взаимодействие по OPC:

1. Программа **«ModbusOPC»** поддерживает обмен данными по протоколу **Modbus**. При подключении такого устройства к компьютеру автоматически запускается программа **«ModbusOPC»** из состава **ZETLAB**, являющаяся полноценным OPC-сервером. Структурная схема, позволяющая понять принцип обмена данными представлена на рисунке 2.

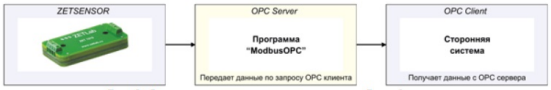



*Рис. 2. Структурная схема, описывающая обмен данными*

2. OPC-сервер **"ZET.OPC"** из состава **ZETLAB** предназначен для подключения аппаратуры к ПО сторонних производителей, если оно удовлетворяет стандарту OPC (рис.3).




*Рис. 3. Работа компонента «Обмен данными с OPC» и OPC-сервера ZETLAB*

3. SCADA-компонент **ZETVIEW «Клиент OPC»** - обмен данными по OPC. Упрощенный **SCADA**-проект, демонстрирующий возможности связи различных компонентов с OPC для предоставления результатов вычисления в другие системы.

Наш OPC-сервер позволяет, помимо обмена данными со SCADA, выполнять следующие полезные функции:

- создавать иерархическое представление имен тегов;
- наблюдать значения тегов.

Также в соответствии со стандартом, OPC-сервер во время инсталляции автоматически регистрируется в реестре Windows. Запуск сервера осуществляется так же, как и любой другой программы или автоматически из клиентской программы.

Клиентская программа и OPC-сервер могут быть установлены на одном и том же компьютере или на разных компьютерах сети Ethernet. Благодаря технологии DCOM, использующей удаленный вызов процедур (***RPC - Remote Procedure Call***) любой OPC-клиент с любого компьютера может обращаться к любому OPC-серверу. Например, SCADA на рис. 4 может обратиться за данными к модулю ввода-вывода по пути, указанному на рис. 4 штриховой линией. ПК и контроллеры в такой архитектуре могут работать с разными промышленными сетями.

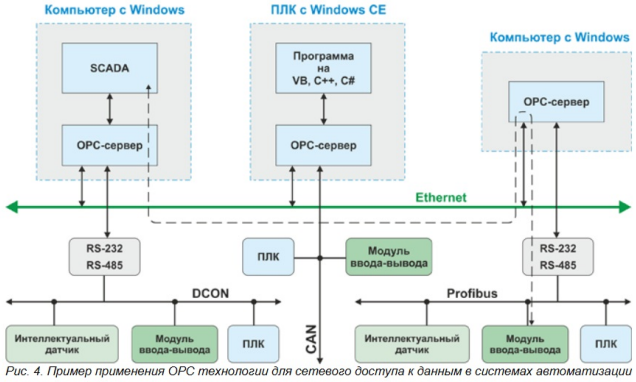


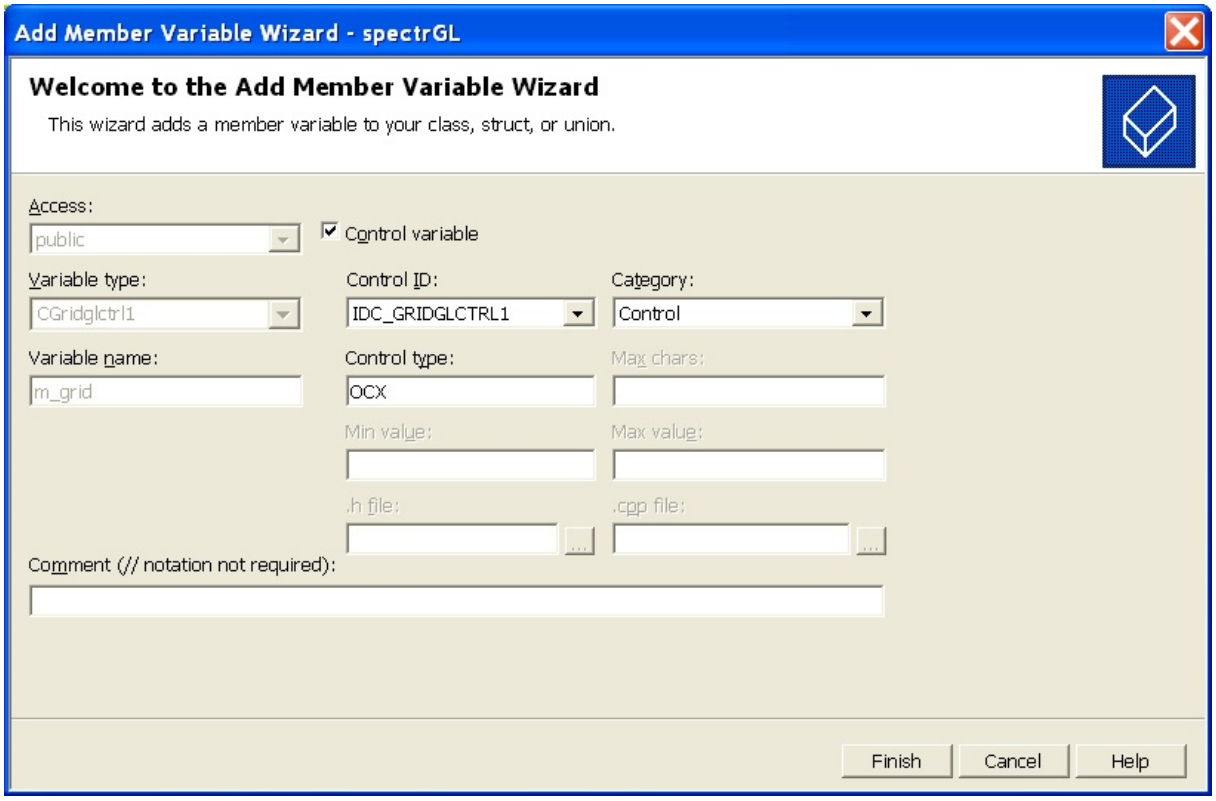

*Рис. 4. Пример применения OPC технологии для сетевого доступа к данным в системах автоматизации*

При использовании оборудования разных производителей на компьютере (контроллере) может быть установлено несколько OPC-серверов разных производителей, но не стоит забывать, что OPC-сервер монопольно занимает COM-порт ПК (поскольку непрерывно выполняет обновление данных), поэтому количество портов должно быть равно количеству OPC-серверов.

© ООО "ЭТМС" 1992-2024. Все права защищены

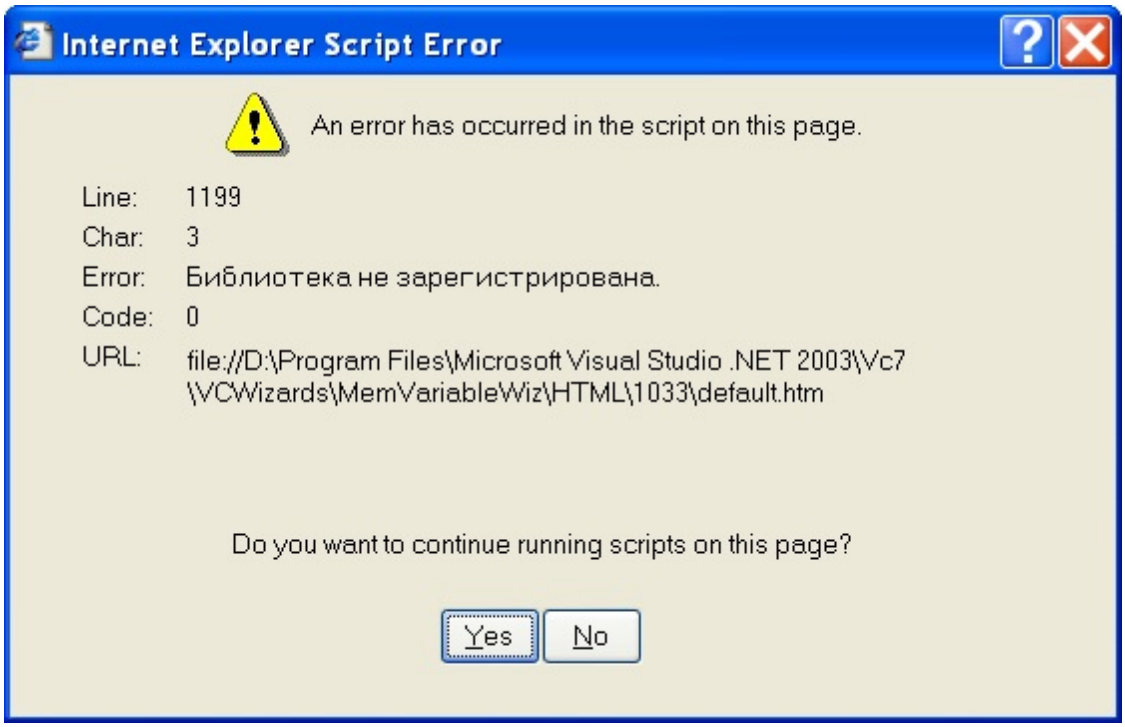
# Ошибки, возникающие при работе с элементами *.OCX

При установке компоненты на диалоговое окно в среде VC++ (2003) и назначении имени переменной этой компоненте командой Add Variable…




возникает ошибка и выдается сообщение:

© ООО "ЭТМС" 1992-2024. Все права защищены




Для того, чтобы эта ошибка не проявлялась, необходимо удалить из директории C:\ZETLab\ файлы с расширением *.OCA. Эти файлы образуются при установке компоненты на диалоговую форму в среде Visual Basic 6.

© ООО "ЭТМС" 1992-2024. Все права защищены

# Modbaszetlab для связи с измерительными программами ZET 7000 ZETSensor

Modbus — открытый коммуникационный протокол, основанный на архитектуре «клиент-сервер». Широко применяется в промышленности для организации связи между электронными устройствами. Может использоваться для передачи данных через последовательные линии связи RS-485, RS-422, RS-232, а также сети TCP/IP (Modbus TCP).

Документ, описывающий протокол называется "Общая структура команд протокола Modbus-Zetlab"

Shared memory создаёт modbuszetlab, и с её помощью считываются параметры датчика.

**struct _STRUCT7000EX()**

*long Virtchan - IN номер виртуального канала в программе MODBUSZETLAB, если -1, то зачитывается информация из линии FTDI и ноды node*
*long Stepcomm - IN команда запуска 1 - запустить процесс чтения информации по заданному виртуальному каналу, -1 запись измененной информации (команда 1 переходит в 2 - этап выполнения-чтение первой зоны памяти, 3- чтение)*
*long GUID - OUT GUID измерительного канала*
*long IsReady - OUT ответ о готовности программы 0- не готов, 1- готов*
*long Ftdindex - OUT номер линии к которой подключен модуль, IN - номер линии по которой надо получить информацию, если к ней не подключен модуль, тогда номер virtchan = -1*
*Long node - OUT номер ноды которая создает измерительный канал*
*Long size_dev_struct - OUT размер структуры структур данных*
*Long size_define - OUT размер структуры структур описателей*
*Char dev_struct [SIZE_OF_STRUCT + 1000] - OUT структуры данных*
*Char define_struct[SIZE_OF_STRUCT + 1000] - OUT структуры описателей*
*Long numftdindex - количество линий FTDI от 0 до NUMLINES*
*LINE_INFO linfo [NUMLINES] - данные о FTDI контроллерах*
*DeviceInfo devinfo[NUMCELLS] - данные об устройствах*
*double lastsend - время последнего запроса, если потерялся, то давай повторим*
*long adtable - адрес чтения из памяти устройства*

© ООО "ЭТМС" 1992-2024. Все права защищены

*long quantity – размер чтения в словах (в 2-ух байтах)*
*long command – номер команды*
*long numelem - количество элементов отображаемых на дереве, при обновлении занулять*
*long numerr - количество перезапусков после 10 надо останавливать*
*HTREEITEM pos - позиция в дереве, на котором стоит курсор*
*FAST_COMMAND fastCommands[NUMLINES][MAX_FAST_COMMAND_PER_LINE] -20 возможных быстрых команд по NUMLINES линиям*

© ООО "ЭТМС" 1992-2024. Все права защищены

# Справка и техническая поддержка



Контактные данные

ООО "Электронные технологии и метрологические системы"

Адрес предприятия ООО «ЭТМС»: 124460, г. Москва, г. Зеленоград, ул. Конструктора Лукина, д. 14, стр. 12, этаж 4, комната 423

Телефон: Тел./факс: +7 (495) 739-39-19 (многоканальный),

+7 (499) 116-70-69 (многоканальный)

Сайт в Internet: https://www.zetlab.com

E-mail:

ZETLAB@ZETLAB.COM - по вопросам подбора и приобретения оборудования

INFO@ZETLAB.COM - по вопросам технической поддержки и консультации

SNAB@ZETLAB.COM - по вопросам снабжения, логистики, сертификации и охране труда

REKLAMA@ZETLAB.COM - по вопросам публикаций и рекламным предложениям

**ЧАСЫ РАБОТЫ** пн–пт: $09^{00}$–$18^{00}$ (MSK time)

**GPS КООРДИНАТЫ** 56.008067, 37.153907

***Техподдержка***

При возникновении вопросов, касающихся выбора нашего оборудования, эксплуатации или обслуживания Вы может обращаться к нам по e-mail или задать вопрос на форуме на нашем сайте. На все Ваши вопросы ответят квалифицированные специалисты.

Для получения консультации у специалиста по вопросам функционирования нашей аппаратуры Вам необходимо подготовить перечень исходных данных. Исходных данных очень много и поэтому лучше их написать в письменном виде и передать по

© ООО "ЭТМС" 1992-2024. Все права защищены

электронной почте. Попытки объяснять ситуацию на пальцах по телефону как правило приводят к потере времени.

Нам необходимы следующие данные:

- Как Вас зовут и как с вами связаться;
- Название изделия и серийный номер;
- Какой компьютер (процессор, память, видео) и операционная система;
- Какой версией и конфигурацией ZETLAB пользуетесь. Дата последнего обновления;
- Какие программные настройки - частота дискретизации, количество каналов, коэффициенты усиления, синфазные, дифференциальные каналы;
- Схема внешнего подключения - текстовое описание, эскиз схемы, фотография подключенного устройства или карандашного эскиза;
- Номера контактов разъемов, длина связей, тип применяемого кабеля: экранированный, витая пара;
- Какие источники сигналов используются, или какие у них внутренние сопротивления;
- Оценка уровней сигналов, приложенных к контактам изделия, какой характер сигнала используется (укажите специфические параметры сигнала, если они известны, импульсный или синусоидальный, случайный или периодичный, ширина полосы частот);
- В каких условиях эксплуатируется изделие (лаборатория, производство);
- Описать, как выполнены цепи заземления компьютера, заземлены ли источники сигналов; если да, то каким образом;
- И, наконец, опишите наблюдаемые помехи, межканальное прохождение или другой негативный эффект, снабдив это описание хотя бы какими-то количественными характеристиками или оценками! Желательно прикладывать копии экрана.

Если Вы потрудитесь немного и предоставите эти полные исходные данные, это позволит специалисту в кратчайшие сроки дать Вам наиболее точный и правильный ответ, что, безусловно, в Ваших интересах!

© ООО "ЭТМС" 1992-2024. Все права защищены

Back Cover