ZETLab

ZETLab Studio

Руководство разработчика

ЗТМС.00068-01 34

ЗАО «Электронные технологии и метрологические системы»

СОДЕРЖАНИЕ

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

1 Введение

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. ***ZetLabStudio*** – интегрированная среда разработчика для создания программ сбора, обработки данных и управления периферийными устройствами. Программирование осуществляется на любом объектно-ориентированном языке программирования Visual Basic, C++, Delphi с использованием библиотечных элементов и готовых программ ***ZETLab***. Сочетание широко используемого 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.

В комплект ZETLab входит более 100 различных готовых программ, компонент и библиотек, которые вы можете интегрировать в свои приложения. В основу пакета программ 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 вам необходимо всего лишь подобрать набор необходимых инструментов и связать их в один проект. Таким образом, ZETLab 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 – 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, корреляторы, регистраторы, генераторы различных сигналов, фильтры верхних и нижних частот, устройство 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 компь-

ютере несколько различных устройств ввода-вывода. Так для медленноменяющихся сигналов, Вы 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 России. 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 компонента 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 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 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 для Visual Studio.NET и Visual Studio 6.0, 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, элемен-

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

тов 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*** 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 Excel и Word позволяют быстро получать 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 и ЦАП фирмы Zet.

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.

1.3.1 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.

Рисунок 1.1

1.3.2 Накопление данных

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

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

- 8-канальный модуль АЦП 16/100, 3-канальный модуль ЦАП 16/2000, 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 АРС 216, 3-канальный модуль ЦАП САР 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 АРС 216, 3-канальный модуль ЦАП САР 316, контроллер KADSP/USB, усилитель для акселерометров с интерфейсом ICP. Частота дискретизации АЦП 200 кГц, ЦАП 2000 кГц, количество двоичных 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 (Аналоговый вход: 4, Частота преобразования по каждому каналу до 500 кГц, Количество разрядов АЦП - 16.)

- тип модуля анализатора ZET017U2 (Аналоговый вход: 2, Частотный диапазон: 0... 10 000 Гц, Частота преобразования по каждому каналу до 50 кГц, Количество разрядов АЦП - 20.)

- тип платы Zet-210 (Аналоговый вход: 16 синфазных/8 дифференциальных, Суммарная частота преобразования 500 кГц, АЦП - 16 бит. ЦАП 16 бит, 2 канала)

- тип платы Zet-220 (Аналоговый вход: 16 синфазных / 8 дифференциальных, Суммарная частота преобразования до 8 кГц, АЦП - 24 бита. ЦАП - 16 бит, 2 канала)

- тип платы Zet-230 (Аналоговый вход: 4 синфазных / 4 дифференциальных, Частота преобразования до 100 кГц по каждому каналу, АЦП - 24 бита. ЦАП - 24 бита, 4 канала.)

- тип платы Zet-048 (тип входных каналов: дифференциальное, Частота преобразования до 1000 Гц по каждому каналу, АЦП - 24 бита, 4-32 канала.)

- тип платы USB Осциллограф ZET 302 (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.

1.3.3 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:

- 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,

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

- ✓ 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.

➢ операции с массивами всех типов:

- ✓ сложение массивов и сложение с константой,
- ✓ вычитание массивов и вычитание константы,
- ✓ умножение массивов и умножение на константу,
- ✓ 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,

➢ 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.

➢ изменение частоты дискретизации массива всех типов:

- ✓ уменьшение частоты – прореживание данных,
- ✓ увеличение частоты – интерполяция данных,
- ✓ изменение частоты с фильтрацией.

➢ расчет корреляции:

- ✓ автокорреляция массива данных,
- ✓ взаимная корреляция массивов данных.

➢ 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.

➢ генерация сигналов:

- ✓ создание массива синусоидального, треугольного сигналов,
- ✓ создание равномерного шума и шума с распределением Гаусса.

➢ функции генерации оконных функций:

- ✓ Хана, Ханнинга, Блэкмана, треугольного взвешивающих массивов.

➢ быстрое и дискретное преобразование Фурье:

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

➢ 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.

➢ свертка сигналов:

- ✓ одномерная и двумерная свертка сигналов.

➢ функции вейвлетного преобразования:

- ✓ разложение сигналов,
- ✓ восстановление сигналов.

1.3.4 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 курсора на графике 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).

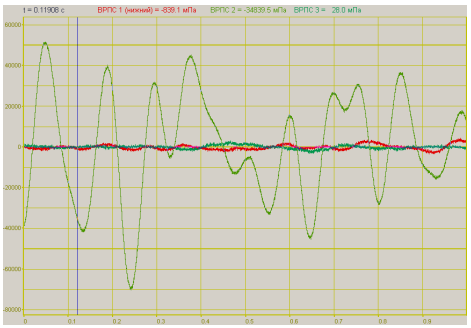


Рисунок 1.2

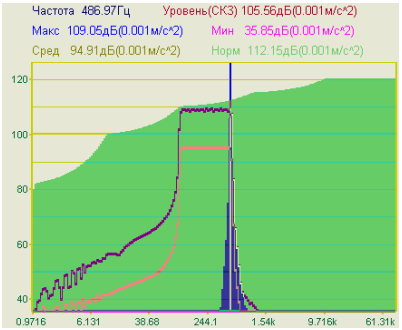


Рисунок 1.3

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).

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

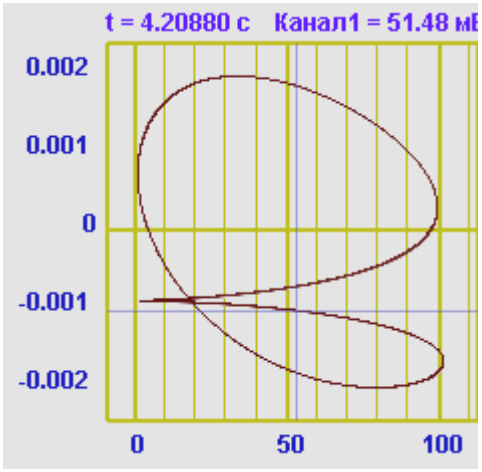


Рисунок 1.4

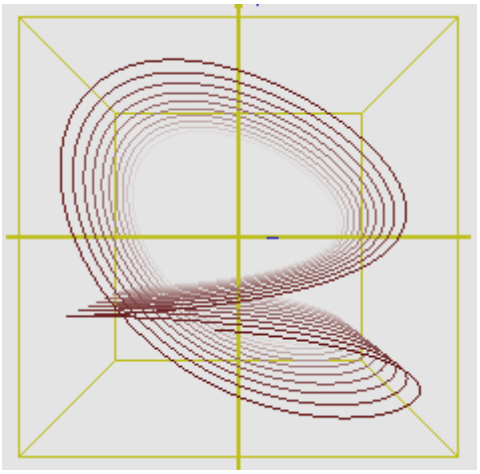


Рисунок 1.5

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

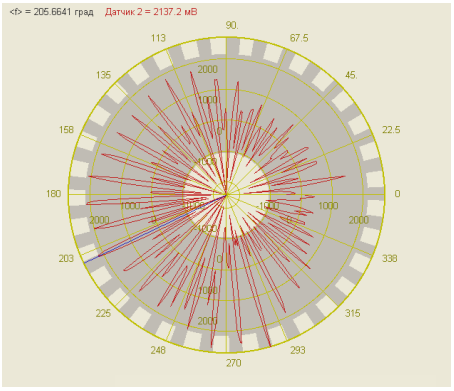


Рисунок 1.6

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

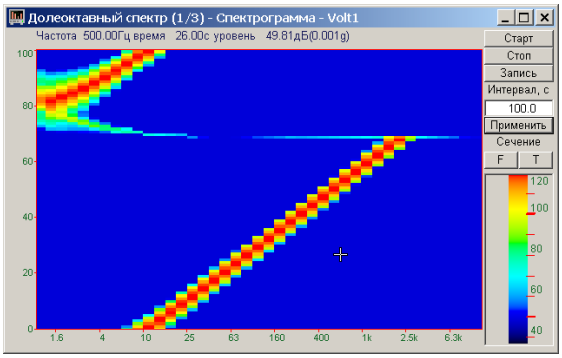


Рисунок 1.7

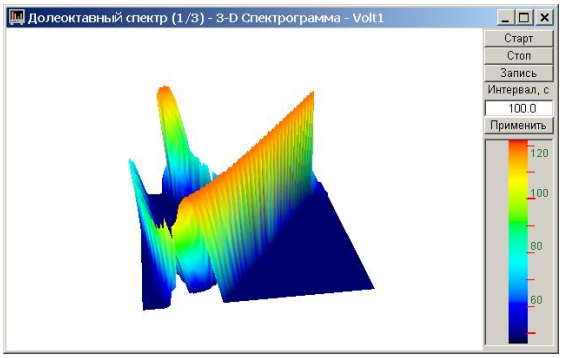


Рисунок 1.8

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

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

2 Программный модуль SRV.ocx для работы с ZETLab

2.1 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».

2.2 Установка компонента

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 60). Основным понятием при обработке сигналов в реальном времени является текущее время. Текущее время сервера показывает, сколько секунд прошло со 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 2.1)

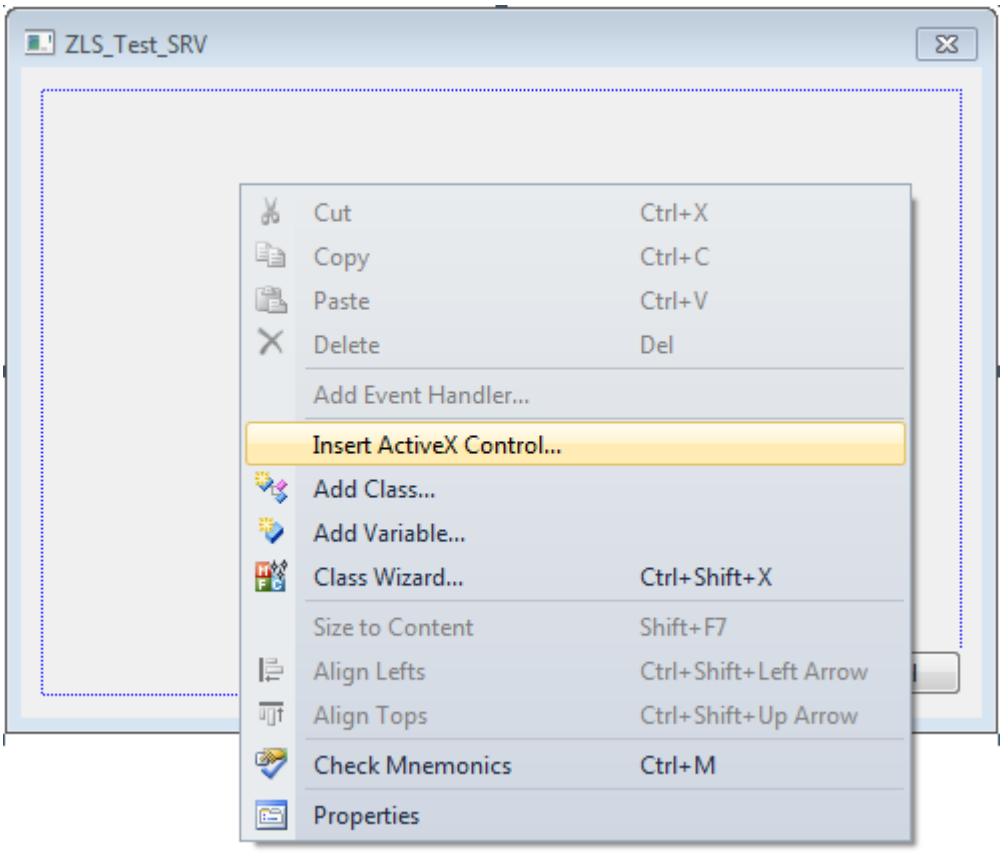


Рисунок 2.1

Из появившегося диалогового окна следует выбрать компонент *SRV Control* и нажать *OK* (рисунок 2.2).

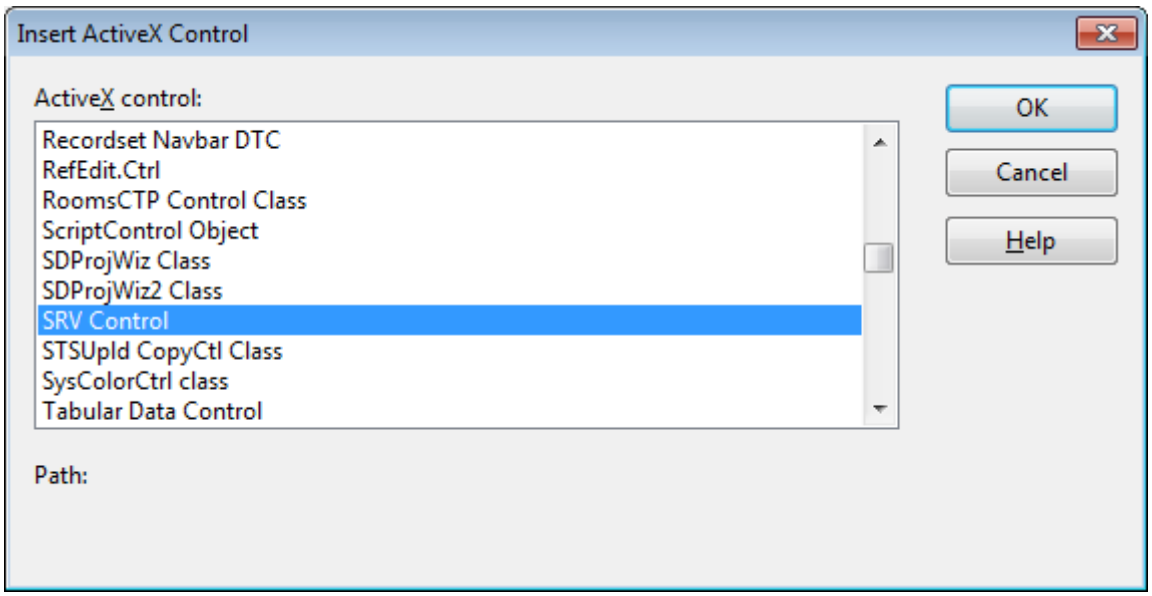


Рисунок 11.2

После этого компонент *SRV.ocx* появится на форме диалога (рисунок 2.3).

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

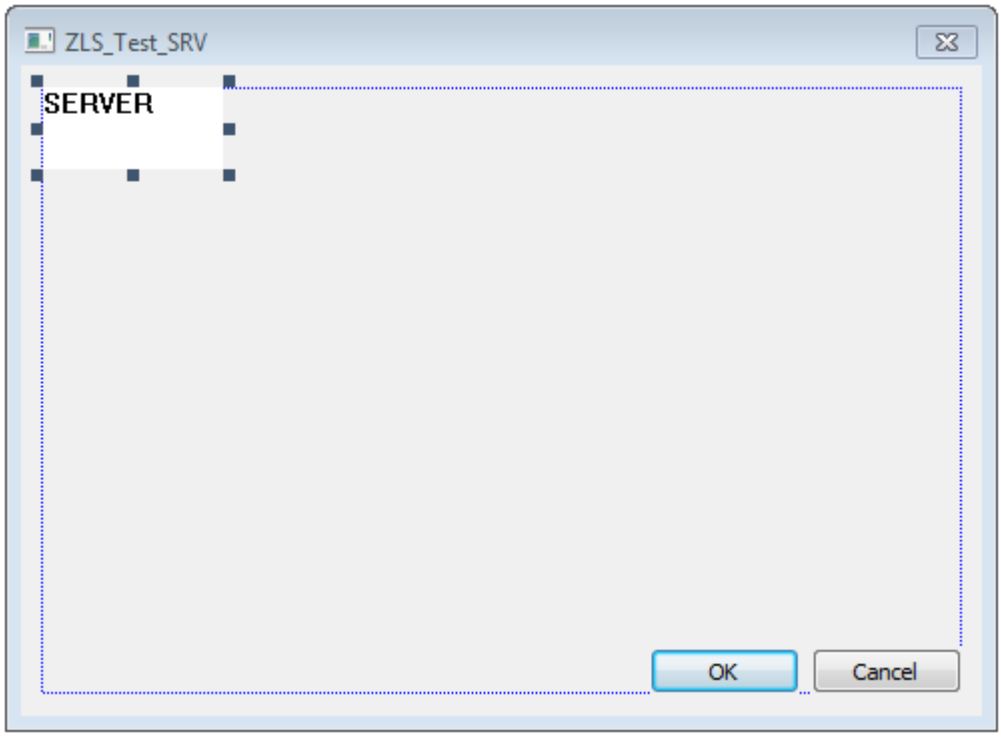


Рисунок 2.3

Для того чтобы компонента не была видна на форме во время выполнения программы, необходимо установить свойство *Visible* равным *False*.

2.3 Описание методов и событий

2.3.1 Методы

2.3.1.1 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

0 – нормальное выполнение функции.

-1 – нет свободного места для подключения (слишком много программ подключено к севреру данных).

-2 – невозможно запустить сервер данных (программа *ZETServer.exe*).

-3 – программа *ZETServer.exe* есть, но она не проходит инициализацию (мало оперативной памяти или дискового пространства).

-7 – серверу данных не хватает памяти для организации буферов данных.

-8 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

Disconnect – отключение от сервера данных. Метод используется при выходе из программы, работающей с данными АЦП.

long ***Disconnect*** *(void)*

Возвращаемое значение

0 – нормальное выполнение функции.

-1 – не было подключения к серверу данных.

-2 – сбой в программе *ZETServer.exe*.

-3 – программа *ZETServer.exe* выгружена из памяти процессора.

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 lNumberDac)*

Параметры

lNumberDac – 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

0 – нормальное выполнение функции.

-1 – нет свободного места для подключения (слишком много программ подключено к севреру данных).

-2 – невозможно запустить сервер данных (программа *ZETServer.exe*).

-3 – программа *ZETServer.exe* есть, но она не проходит инициализацию (мало оперативной памяти или дискового пространства).

-7 – серверу данных не хватает памяти для организации буферов данных.

-8 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

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 lNumberDac)*

Параметры

lNumberDac – 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

0 – нормальное выполнение функции.

-1 – не было подключения к серверу данных.

-2 – сбой в программе *ZETServer.exe*.

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

-3 – программа *ZETServer.exe* выгружена из памяти процессора.

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 lNumChan)*

Параметры

lNumChan – 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

0 – нормальное выполнение функции.

-1 – нет свободного места для подключения (слишком много программ подключено к севреру данных).

-2 – невозможно запустить сервер данных (программа *ZETServer.exe*).

-3 – программа *ZETServer.exe* есть, но она не проходит инициализацию (мало оперативной памяти или дискового пространства).

-5 – нет свободного места для подключения виртуальных каналов (подключено слишком много виртуальных каналов).

-6 – невозможно подключить виртуальные каналы.

-7 – серверу данных не хватает памяти для организации буферов данных.

-8 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

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.

2.3.1.2 Опрос параметров каналов системы

QuanChan *– опрос количества работающих каналов на сервере данных*

long ***QuanChan*** *(void)*

Возвращаемое значение

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа *ZETServer.ocx* выгружена из памяти процессора.

≥ 0 – суммарное количество работающих каналов на сервере (каналы АЦП, виртуальные каналы, виртуальные каналы ЦАП)

WorkChanADC *– опрос количества работающих каналов на модуле АЦП-ЦАП, которому принадлежит заданный канал.*

long ***WorkChanADC*** *(long lChannel)*

Параметры

lChannel – номер канала, принадлежащий тому модулю АЦП-ЦАП, количество работающих каналов которого требуется узнать. Параметр *lChannel* 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 *lChannel < 0.*

-4 – параметр *lChannel* > (*QuanChan() – 1).*

-5 – заданный канал не является каналом модуля АЦП-ЦАП.

MaxQuanChanDac – опрос суммарного количества каналов ЦАП, которые можно задействовать в системе.

long ***MaxQuanChanDac*** *(void)*

Возвращаемое значение

-1 – на момент вызова метода не было подключения к серверу данных.

-2 – программа *ZETServer.ocx* выгружена из памяти процессора.

≥ 0 – суммарное количество каналов ЦАП, которые можно задействовать в системе

Frequency – опрос частоты дискретизации заданного канала.

float ***Frequency*** *(long lChannel)*

Параметры

lChannel – номер канала, 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 *lChannel* может принимать значение от *0* до (*QuanChan() – 1).*

Возвращаемое значение

> 0 – частота дискретизации заданного канала.

Commentary – опрос имени (названия) заданного канала.

CString ***Commentary*** *(long lChannel)*

Параметры

lChannel – номер канала, имя (название) которого требуется узнать. Имя (название) канала 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

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 *lChannel* > (*QuanChan() – 1).*

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

непустая строка – имя заданного канала.

Conversion – опрос единицы измерения по заданному каналу

CString ***Conversion*** *(long lChannel)*

Параметры

lChannel – 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 *lChannel* 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 *lChannel* > (*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 lChannel)*

Параметры

lChannel – номер канала, максимально допустимый уровень которого требуется узнать. Параметр *lChannel* может принимать значение от *0* до (*QuanChan() – 1)*.

Возвращаемое значение

-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)*.
≥ 0 – максимально допустимый уровень по заданному каналу в единицах измерения

MinLevel – опрос минимально различимого уровня по заданному каналу в единицах измерения (вес младшего разряда по каналу)

float ***MinLevel*** *(long lChannel)*

Параметры

lChannel – номер канала, минимально различимый уровень которого требуется узнать. Параметр *lChannel* может принимать значение от *0* до (*QuanChan() – 1)*.

Возвращаемое значение

-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)*.
≥ 0 – минимально допустимый уровень по заданному каналу в единицах измерения

Reference – опрос значения опоры для расчета уровней в децибелах по заданному каналу.

float ***Reference*** *(long lChannel)*

Параметры

lChannel – 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

-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).*
≥ 0 – значение опоры для расчета уровней в децибелах по заданному каналу.

ShifLevel – опрос смещения постоянной составляющей по заданному каналу в единицах измерения.

*float **ShifLevel** (long lChannel)*

Параметры

lChannel – 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 *lChannel* может принимать значение от *0* до (*QuanChan() – 1).*

Возвращаемое значение

Величина *channel* может принимать значение от *0* до *QuanChan()-1*.

Возвращаемое значение

-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).*
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 lChannel)*

Параметры

lChannel – номер канала, чувствительность которого требуется узнать. Чувствительность 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

любое значение – чувствительность (коэффициент преобразования) заданного канала в вольтах на единицу измерения.

AFCH – опрос названия файла, в котором может храниться пользовательская информация о канале, например, частотно-зависимые поправки АЧХ тракта.

*CString **AFCH** (long lChannel)*

Параметры

lChannel – номер канала, название вспомогательного файла которого требуется узнать. Название вспомогательного файла задается в таблице конфигурации *Devices.cfg* и

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

программой «Диспетчер устройств» из группы «Сервисные». Параметр *lChannel* 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 *lChannel* > (*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 lChannel)*

Параметры

lChannel – номер канала, нормированный текущий уровень которого требуется узнать. Сервер данных нормирует получаемый текущий уровень по максимальному уровню по заданному каналу. Параметр *lChannel* 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 *lChannel < 0*.

-4 – параметр *lChannel* > (*QuanChan() – 1).*

GetStatus – опрос типа заданного канала

long ***GetStatus*** *(long lChannel)*

Параметры

lChannel – номер канала, тип которого требуется узнать. Параметр *lChannel* может принимать значение от *0* до (*QuanChan() – 1).*

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 интеллектуальнлого датчика.

8 – заданный канал является каналом скоростного устройства, которое выдаёт данные порциями по таймеру или по событию.

2.3.1.3 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 lNumberDSP)*

Параметры

lNumberDSP – номер устройства (сигнального процессора), количество физических каналов которого требуется узнать. Параметр *lNumberDSP* может принимать значение от *0* до (*QuanDSP() – 1).*

Возвращаемое значение

≥ 0 – количество физических каналов устройства.
-1 – параметр *lNumberDSP* < *0*.
-2 – параметр *lNumberDSP* > (максимальное количество устройств в системе – *1).*
-3 – параметр *lNumberDSP* > (*QuanDSP() – 1).*

TypeAdc – опрос типа устройства (платы) АЦП-ЦАП, физическим каналом которого является заданный канал.

long ***TypeAdc****(long lChannel)*

Параметры

lChannel – номер канала, тип устройства которого требуется узнать. Параметр *lChannel* может принимать значение от *0* до (*QuanChan() – 1).*

Возвращаемое значение

0 – АЦП устройства ADC 16/200 (драйвер Kd1610.sys).
1 – АЦП устройства АРС 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 (драйвер Kdu1616.sys).
11 – АЦП устройства PD14 USB (драйвер KduPD14.sys)
12 – АЦП устройства ZET110 (драйвер KduVN.sys).
13 – АЦП устройства ZET302 (драйвер KduOsc.sys).
14 – АЦП устройства ZET017 (драйвер Kdu8500.sys).
15 – АЦП устройства ZET017-U2, ZET019-U2 (драйвер Kdu2500.sys).
16 – АЦП устройства ZET220 (драйвер Kdu1624.sys).
17 – АЦП устройства ZET230 (драйвер Kdu0424.sys).
18 – АЦП устройства ZET240 (драйвер Kdu0414.sys).
19 – АЦП устройства ZET240, ZET048 (драйвер Kdu0824.sys).
100 – ЦАП устройства ADC 16/200 (драйвер Kd1610.sys).
101 – ЦАП устройства АРС 216 (драйвер Kd216.sys).
102 – ЦАП устройства ADC 16/500 (драйвер Kd500.sys).
103 – ЦАП устройства ADC 16/500P (драйвер Kd500p.sys).
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).

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

108 – ЦАП устройства ADC 1432 (драйвер Kd1432.sys).
109 – ЦАП устройства ACPB USB (драйвер KduACPB.sys).
110 – ЦАП устройства ZET210 (драйвер Kdu1616.sys).
111 – ЦАП устройства PD14 USB (драйвер KduPD14.sys)
112 – ЦАП устройства ZET110 (драйвер KduVN.sys).
113 – ЦАП устройства ZET302 (драйвер KduOsc.sys).
114 – м устройства ZET017 (драйвер Kdu8500.sys).
115 – ЦАП устройства ZET017-U2, ZET019-U2 (драйвер Kdu2500.sys).
116 – ЦАП устройства ZET220 (драйвер Kdu1624.sys).
117 – ЦАП устройства ZET230 (драйвер Kdu0424.sys).
118 – ЦАП устройства ZET240 (драйвер Kdu0414.sys).
119 – ЦАП устройства ZET240, ZET048 (драйвер Kdu0824.sys).
-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 *lChannel* не является ни каналом АЦП, ни каналом ЦАП.

NumDSP – опрос номера сигнального процессора, физическим каналом которого является заданный канал.

*long **NumDSP** (long lChannel)*

Параметры
lChannel – номер канала, номер сигнального процессора которого требуется узнать. Параметр *lChannel* 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 *lChannel < 0*.
-4 – параметр *lChannel* > (*QuanChan() – 1)*.
-5 – заданный канал *lChannel* не является ни каналом АЦП, ни каналом ЦАП.

NumModule – опрос номера модуля устройства (сигнального процессора) в многомодульной системе.

*long **NumModule** (long lChannel)*

Параметры
lChannel – номер канала, номер модуля сигнального процессора которого требуется узнать. Параметр *lChannel* может принимать значение от *0* до (*QuanChan() – 1)*.

Возвращаемое значение
≥ 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 *lChannel < 0*.
-4 – параметр *lChannel* > (*QuanChan() – 1)*.

-5 – заданный канал *lChannel* не является ни каналом АЦП, ни каналом ЦАП.

NumPhChan – опрос физического номера канала на модуле для заданного канала.

*long **NumPhChan** (long lChannel)*

Параметры

lChannel – номер канала, физический номер которого требуется узнать. Параметр *lChannel* может принимать значение от *0* до (*QuanChan() – 1).*

Возвращаемое значение

≥ 0 и < 100 – физический номер заданного канала на модуле АЦП.
100 – заданный канал является каналом ЦАП.
-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 *lChannel* не является ни каналом АЦП, ни каналом ЦАП.

GetFrequency – опрос частоты дискретизации заданного устройства.

*float **GetFrequency** (long lModule)*

Параметры

lModule – номер устройства (сигнального процессора) в системе, частоту дискретизации которого требуется узнать. Параметр *lModule* может принимать значение от *0* до (*QuanDSP() – 1),* и тогда возвращаемым значением будет частота дискретизации модуля АЦП устройства. Параметр *lModule* может принимать значение от *-1* до (- *QuanDSP()),* и тогда возвращаемым значением будет частота дискретизации модуля ЦАП устройства.

Возвращаемое значение

≥ 0 – частота дискретизации заданного устройства.

Пример 1. Для установления частоты дискретизации на приборе АЦП используют функцию SetFrequency (0,50000), а для ЦАП SetFrequency (-1,200000). Сделано на примере одного прибора Анализатора ZET017U8.

Пример 2. Программа находит максимальную частоту дискретизации прибора, а затем устанавлиает ее на АЦП и ЦАП.

```
max_ADC = GetNextFreq(0, 0);
SetFrequency(0, max_ADC);

max_DAC = GetNextFreq(-1, 0);
SetFrequency(-1, max_DAC);
```

GetNextFreq – опрос списка возможных частот дискретизации заданного устройства. При многократном обращении к этой функции, она возвращает возможные частоты дискретизации.

*float **GetNextFreq** (long lModule, long lParametr)*

Параметры

lModule – номер устройства (сигнального процессора) в системе, список частот дискретизации которого требуется узнать. Параметр *lModule* может принимать значение от

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

0 до (*QuanDSP() – 1),* и тогда возвращаемым значением будет частота дискретизации модуля АЦП устройства. Параметр *lModule* может принимать значение от *-1* до (- *QuanDSP()),* и тогда возвращаемым значением будет частота дискретизации модуля ЦАП устройства.

lParametr – 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 lChannel)*

Параметры

lChannel – номер канала, коэффициент усиления по которому требуется узнать. Параметр *lChannel* может принимать значение от *0* до (*QuanChan() – 1).*

Возвращаемое значение

≥ 0 – коэффициент усиления по заданному каналу.

long ***IdChan*** *(long lChannel, long* lId)*

Параметры

lChannel – номер канала, идентификатор которого требуется узнать. Параметр *lChannel* может принимать значение от *0* до (*QuanChan() – 1).*

lId – возвращает 32-разрядный идентификатор. Для физического канала АЦП и ЦАП *lId* 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 *lId* 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

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).*

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 lChannelDac, long* lId)*

Параметры

lChannel – номер канала, идентификатор которого требуется узнать. Параметр *lChannel* может принимать значение от *0* до (*MaxQuanChanDac() – 1).*

lId – 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.
-1 – на момент вызова метода не было подключения к серверу данных.
-2 – программа *ZETServer.ocx* выгружена из памяти процессора.
-3 – параметр *lChannel < 0.*
-5 – параметр *lChannel* > (*MaxQuanChanDac () – 1).*
-6 – канал ЦАП с заданным номером не подключен.

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 lChannel, long* lID)*

Параметры

lChannel – номер виртуального канала, идентификатор которого требуется узнать. Параметр *lChannel* может принимать значение значение от *0* до (количество виртуальных каналов, установленных функцией *ConnectVrtCh()* минус 1).

lId – возвращает 32-разрядный идентификатор. Для виртуального канала *lId* 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

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 – нет организованного программой виртуального канала с номером *lChannel*
-6 – нет организованного программой виртуального канала с номером *lChannel*
-7 – нет организованного программой виртуального канала с номером *lChannel*

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.

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

2.3.1.4 Управление каналами АЦП

SetFrequency – установка частоты дискретизации заданного устройства.

*float **SetFrequency** (long lModule, float fFrequency)*

Параметры

lModule – номер устройства (сигнального процессора) в системе, частоту дискретизации которого требуется установить. Параметр *lModule* может принимать значение от *0* до (*QuanDSP() – 1),* и тогда будет установлена частота дискретизации модуля АЦП устройства. Параметр *lModule* может принимать значение от *-1* до (*- QuanDSP()),* и тогда будет установлена частота дискретизации модуля ЦАП устройства.

fFrequency – частота дискретизации, которую требуется установить.

Возвращаемое значение

частота дискретизации устройства, которая установилась.

OnOffChannel – установка состояния канала АЦП заданного устройства.

*long **OnOffChannel** (long lModule,long lChannel, long lState)*

Параметры

lModule – номер устройства (сигнального процессора) в системе, состояние канала которого требуется установить. Параметр *lModule* может принимать значение от *0* до (*QuanDSP() – 1).*

lChannel – номер канала устройства, состояние которого требуется установить. Параметр *lChannel* может принимать значение от *0* до (*QuanPhChan () – 1).*

lState – состояние канала, которое требуется установить. Если параметр *lState* 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 lModule,long lChannel, long lState)*

Параметры

lModule – номер устройства (сигнального процессора) в системе, режим работы канала которого требуется установить. Параметр *lModule* может принимать значение от *0* до (*QuanDSP() – 1).*

lChannel – номер канала устройства, режим работы которого требуется установить. Параметр *lChannel* может принимать значение от *0* до (*QuanPhChan () – 1).*

lState – режим работы канала, который требуется установить. Если параметр *lState* 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.

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_DAC = GetNextFreq(-1, 0);
SetFrequency(-1, max_DAC);
```

2.3.1.5 Управление каналами ЦАП

IsFreeChanDac – определение свободного (незанятого другой программой) канала ЦАП.

long ***IsFreeChanDac*** *(long lChannelDac)*

Параметры

lChannelDac – номер канала ЦАП, состояние которого требуется узнать. Параметр *lChannelDac* может принимать значение от *0* до (*MaxQuanChanDac () – 1).*

Возвращаемое значение

-2 – программа *ZETServer.ocx* выгружена из памяти процессора.
-3 – параметр *lChannelDac < 0.*
-5 – параметр *lChannelDac* > (*MaxQuanChanDac () – 1).*
0 – заданный канал ЦАП свободен (не занят другой программой)
1 – заданный канал ЦАП занят.

SetChanDac – подключение заданного канала ЦАП.

long ***SetChanDac*** *(long lChannelDac)*

Параметры

lChannelDac – номер канала ЦАП, который требуется подключить. Параметр *lChannelDac* может принимать значение от *0* до (*MaxQuanChanDac () – 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 *lChannelDac < 0.*
-5 – параметр *lChannelDac* > (*MaxQuanChanDac () – 1).*
-20 – невозможно подключить заданный канал ЦАП.

RemChanDac – отключение заданного канала ЦАП.

long ***RemChanDac*** *(long lChannelDac)*

Параметры

lChannelDac – номер канала ЦАП, который требуется отключить. Параметр *lChannelDac* может принимать значение от *0* до (*MaxQuanChanDac () – 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 *lChannelDac < 0.*
-5 – параметр *lChannelDac* > (*MaxQuanChanDac() – 1).*
-20 – невозможно отключить заданный канал ЦАП.

GetAttenuation – определение коэффициента затухания заданного канала ЦАП.

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

float ***GetAttenuation*** *(long lChannelDac)*

Параметры

lChannelDac – номер канала ЦАП, коэффициент затухания которого требуется узнать. Параметр *lChannelDac* 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 *lChannelDac < 0.*

-5 – параметр *lChannelDac* > (*MaxQuanChanDac () – 1).*

-6 – заданный канал ЦАП программой не задействован.

-12 – подключение к серверу было произведено методами *Connect()* или *ConnectVrtCh()* , которые не организуют каналов ЦАП.

SetAttenuation – установка коэффициента затухания заданного канала ЦАП.

long ***SetAttenuation*** *(long lChannelDac, float fAttenuator)*

Параметры

lChannelDac – номер канала ЦАП, коэффициент затухания которого следует установить. Параметр *lChannelDac* может принимать значение от *0* до (*MaxQuanChanDac () – 1).*

fAttenuator – коэффициент затухания, который следует установить. Параметр *fAttenuator* может принимать значения от 0 до 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 *lChannelDac < 0.*

-5 – параметр *lChannelDac* > (*MaxQuanChanDac () – 1).*

-6 – заданный канал ЦАП программой не задействован.

-12 – подключение к серверу было произведено методами *Connect()* или *ConnectVrtCh()* , которые не организуют каналов ЦАП.

FrequencyDac – опрос частоты дискретизации заданного канала ЦАП.

float ***Frequency*** *(long lChannelDac)*

Параметры

lChannelDac – 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 *lChannelDac* может принимать значение от *0* до (*MaxQuanChanDac () – 1).*

Возвращаемое значение

> 0 – частота дискретизации заданного канала ЦАП.

-3 – параметр *lChannelDac < 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 *lChannelDac* > (*MaxQuanChanDac () – 1).*

Пример 1. Для установления частоты дискретизации на приборе для 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);
```

2.3.1.6 Управление виртуальными каналами

VrtChnInfo – задание информации для виртуального канала.

long ***VrtChnInfo*** *(long lChannelVirt, BSTR* sCommentary, BSTR* sConversion, float fFrequency, float fMaxLevel, float fMinLevel, float fReferense)*

Параметры

lChannelVirt – номер виртуального канала, информацию по которому требуется задать. Параметр *lChannelVirt* может принимать значение от *0* до (количество виртуальных каналов, установленных функцией *ConnectVrtCh (...)* минус 1).

sCommentary – строка с названием виртуального канала.

sConversion – строка с названием единицей измерения по виртуальному каналу.

fFrequency – частота дискретизации по виртуальному каналу.

fMaxLevel – максимально допустимый уровень по виртуальному каналу.

fMinLevel – минимально различимый уровень по виртуальному каналу.

fReferense – значение опорного уровня для расчета уровней сигнала в децибелах.

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 *lChannelVirt < 0.*

-4 – параметр *lChannelVirt* > (*QuanChan() – 1).*

-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 – нет организованного программой виртуального канала с номером *lChannelVirt*

2.3.1.7 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.

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

SetDigOutEnable – установка маски вывода на выбранному устройстве.

long ***SetDigOutEnable****(long lModule, long lOutMask)*

Параметры

lModule – номер устройства (сигнального процессора) в системе, маску вывода цифрового порта которого следует установить. Параметр *lModule* может принимать значение от *0* до (*QuanDSP() – 1).*

lOutMask – 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

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 *lModule < 0.*
-4 – параметр *lModule* > (количество устройств в системе – *1).*
-5 – невозможно подключиться к устройству.
-6 – невозможно установить маску вывода для цифрового порта устройства.

GetDigOutEnable – опрос маски вывода на выбранном устройстве.

long ***GetDigOutEnable****(long lModule)*

Параметры

lModule – номер устройства (сигнального процессора) в системе, маску вывода цифрового порта которого требуется узнать. Параметр *lModule* 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 *lModule < 0.*
-4 – параметр *lModule* > (количество устройств в системе – *1).*
-5 – невозможно подключиться к устройству.
-6 – невозможно опросить маску вывода для цифрового порта устройства.

GetDigInput – опрос входных данных цифрового порта на выбранном устройстве.

long ***GetDigInput****(long lModule)*

Параметры

lModule – номер устройства (сигнального процессора) в системе, входные данные цифрового порта которого требуется узнать. Параметр *lModule* 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 *lModule < 0.*
-4 – параметр *lModule* > (количество устройств в системе – *1).*
-5 – невозможно подключиться к устройству.
-6 – невозможно опросить входные данные цифрового порта устройства.

GetDigOutput – опрос выходных данных цифрового порта на выбранном устройстве.

long ***GetDigOutput****(long lModule)*

Параметры

lModule – номер устройства (сигнального процессора) в системе, выходные данные цифрового порта которого требуется узнать. Параметр *lModule* 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 *lModule < 0.*
-4 – параметр *lModule* > (количество устройств в системе – *1).*
-5 – невозможно подключиться к устройству.
-6 – невозможно опросить выходные данные цифрового порта устройства.

SetDigOutput – установка выходных данных цифрового порта на выбранном устройстве.

long ***SetDigOutput****(long lModule, long lOutData)*

Параметры

lModule – номер устройства (сигнального процессора) в системе, выходные данные цифрового порта которого следует установить. Параметр *lModule* может принимать значение от *0* до (*QuanDSP() – 1).*

lOutData – 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.

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 *lModule < 0.*
-4 – параметр *lModule* > (количество устройств в системе – *1).*
-5 – невозможно подключиться к устройству.
-6 – невозможно установить выходные данные для цифрового порта устройства.

GetDigBits – опрос количества бит цифрового порта на выбранном устройстве.

long ***GetDigBits*** *(long lModule)*

Параметры

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

lModule – номер устройства (сигнального процессора) в системе, количество бит цифрового порта которого требуется узнать. Параметр *lModule* может принимать значение от *0* до (*QuanDSP() – 1)*.

Возвращаемое значение

≥ 0 – количество бит цифрового порта устройства.

SetBitDigOutput – установка состояния выходного бита на заданном устройстве в логическую единицу.

long ***SetBitDigOutput****(long lModule, long lBit)*

Параметры

lModule – номер устройства (сигнального процессора) в системе, состояние выходного бита которого следует установить в логическую единицу. Параметр *lModule* может принимать значение от *0* до (*QuanDSP() – 1)*.

lBit – номер бита, состояние которого следует установить в логическую единицу. Параметр *lModule* 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 *lModule < 0*.

-4 – параметр *lModule* > (количество устройств в системе – *1)*.

-5 – невозможно подключиться к устройству.

-6 – невозможно установить состояние выходного бита в логическую единицу для цифрового порта заданного устройства.

ClrBitDigOutput – установка состояния выходного бита на заданном устройстве в логический ноль.

long ***ClrBitDigOutput*** *(long lModule, long lBit)*

Параметры

lModule – номер устройства (сигнального процессора) в системе, состояние выходного бита которого следует установить в логический ноль. Параметр *lModule* может принимать значение от *0* до (*QuanDSP() – 1)*.

lBit – номер бита, состояние которого следует установить в логический ноль. Параметр *lModule* 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 *lModule < 0*.

-4 – параметр *lModule* > (количество устройств в системе – *1)*.

-5 – невозможно подключиться к устройству.

-6 – невозможно установить состояние выходного бита в логический ноль для цифрового порта заданного устройства.

2.3.1.8 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 lModule)*

Параметры

lModule – номер устройства (сигнального процессора) в системе, наличие ШИМ у которого требуется узнать. Параметр *lModule* 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 *lModule < 0.*
-4 – параметр *lModule* > (количество устройств в системе – *1).*
-5 – невозможно подключиться к устройству.
-6 – невозможно опросить наличие ШИМ устройства.

SetFreqPWM – установка частоты ШИМ.

long ***SetFreqPWM*** *(long lModule, long lRate, long lPeriod)*

Параметры

lModule – номер устройства (сигнального процессора) в системе, частоту ШИМ которого следует установить. Параметр *lModule* может принимать значение от *0* до (*QuanDSP() – 1).*

lRate – коэффициент деления опорной частоты 96 МГц. Внутри модуля установлен кварцевый генератор с тактовой частотой 96 МГц. Параметр *lRate* может принимать значение от 1 до 96000000.

lPeriod – период ШИМ. Параметр *lPeriod* может принимать значение от 2 до 1024. Количество тактов модуляции задает степень скважности ШИМ. Выходная частота ШИМ равна (96000 / *lRate* / *lPeriod*) кГц

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 *lModule < 0.*
-4 – параметр *lModule* > (количество устройств в системе – *1).*
-5 – невозможно подключиться к устройству.
-6 – невозможно установить частоту ШИМ устройства.
-7 – выбранное устройство не поддерживает ШИМ.

SetOnDutyPWM – установка скважности и фазы ШИМ по заданному каналу выбранного устройства.

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

long ***SetOnDutyPWM*** *(long lModule, long lChannel, long lOnDutyPWM, long lShiftPWM)*

Параметры

lModule – номер устройства (сигнального процессора) в системе, скважность и фазу ШИМ которого следует установить. Параметр *lModule* может принимать значение от *0* до (*QuanDSP() – 1).*

lChannel – номер канала, скважность и фазу ШИМ которого следует установить. Параметр *lChannel* может принимать значение от 0 до 2.

lOnDutyPWM – скважность заданного канала ШИМ.

lShiftPWM – 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 *lModule < 0.*

-4 – параметр *lModule* > (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.

RegulatorPWM – установка произвольных данных для ШИМ.

long ***RegulatorPWM*** *(long lModule, float* pData, long lSize)*

Параметры

lModule – номер устройства (сигнального процессора) в системе, данные ШИМ которого следует установить. Параметр *lModule* может принимать значение от *0* до (*QuanDSP() – 1).*

pData – данные для установки в ШИМ.

lSize – 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 *lModule < 0.*

-4 – параметр *lModule* > (количество устройств в системе – *1).*

-5 – невозможно подключиться к устройству.

-6 – невозможно установить произвольные данные для ШИМ.

-7 – выбранное устройство не поддерживает ШИМ.

StartPWM – запуск ШИМ по заданному каналу выбранного устройства.

long ***StartPWM*** *(long lModule, long lChannel)*

Параметры

lModule – номер устройства (сигнального процессора) в системе, запуск ШИМ которого следует осуществить. Параметр *lModule* может принимать значение от *0* до (*QuanDSP() – 1).*

lChannel – номер канала, запуск ШИМ которого следует осуществить. Параметр *lChannel* 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 *lModule < 0.*
-4 – параметр *lModule* > (количество устройств в системе – *1).*
-5 – невозможно подключиться к устройству.
-6 – невозможно запустить ШИМ по заданному каналу устройства.
-7 – выбранное устройство не поддерживает ШИМ.

StopPWM – останов ШИМ по заданному каналу выбранного устройства.

*long **StopPWM** (long lModule, long lChannel)*

lModule – номер устройства (сигнального процессора) в системе, останов ШИМ которого следует осуществить. Параметр *lModule* может принимать значение от *0* до (*QuanDSP() – 1).*

lChannel – номер канала, останов ШИМ которого следует осуществить. Параметр *lChannel* 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 *lModule < 0.*
-4 – параметр *lModule* > (количество устройств в системе – *1).*
-5 – невозможно подключиться к устройству.
-6 – невозможно остановить ШИМ по заданному каналу устройства.
-7 – выбранное устройство не поддерживает ШИМ.

2.3.1.9 Работа с ICP

PrusEna – опрос наличия усилителя заряда (ICP) для подключения вибродатчиков и микрофонов к заданному каналу.

*long **PrusEna** (long lChannel)*

Параметры

lChannel – номер канала, наличие усилителя заряда у которого требуется узнать. Параметр *lChannel* 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 *lChannel < 0.*
-4 – параметр *lChannel* > (*QuanChan() – 1).*

GetICP – опрос состояния усилителя заряда (ICP) для подключения вибродатчиков и микрофонов к заданному каналу.

*long **GetICP** (long lModule, long lChannel)*

Параметры

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

lModule – номер устройства (сигнального процессора) в системе, состояние усилителя заряда канала которого требуется узнать. Параметр *lModule* может принимать значение от *0* до (*QuanDSP() – 1).*

lChannel – номер канала, состояние усилителя заряда у которого требуется узнать. Параметр *lChannel* 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 lModule, long lChannel, long lType)*

Параметры

lModule – номер устройства (сигнального процессора) в системе, состояние усилителя заряда канала которого следует установить. Параметр *lModule* может принимать значение от *0* до (*QuanDSP() – 1).*

lChannel – номер канала, состояние усилителя заряда которого следует установить. Параметр *lChannel* может принимать значение от *0* до (*QuanChan() – 1).*

lType – состояние усилителя заряда, которое требутся установить.

Возвращаемое значение

0 – нормальное выполнение функции

-2 – усилитель заряда у каналов заданного устройства отсутствует.

2.3.1.10 Работа с GPS

PutDataGPS – запись данных NMEA-потока в сервер данных.

*long **PutDataGPS** (LPCTSTR sNMEA, long lSize)*

Параметры

sNMEA – строка с данными NMEA-потока для записи в сервер данных.

lSize – 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().*

*CString **GetDataGPS** (long lMaxSize, long* lRealSize)*

Параметры

lMaxSize – максимальный размер запрашиваемых данных.

lRealSize – 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 fShift)*

Параметры

fShift – 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*** *()*

Возвращаемое значение

значение смещения времени компьютера относительно времени по GPS.

enaGPSync – установка возможности синхронизации устройств по GPS.

long ***enaGPSync*** *(long lEnable)*

Параметры

lEnable – флаг синхронизации по GPS (0 – отключение синхронизации, ≥ 0 – включение синхронизации устройств по GPS)

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 lTime)*

Параметры

lTime – 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 lSatellites)*

Параметры

lSatellites – количество спутников GPS, наблюдаемое на данный момент.

Возвращаемое значение

0 – нормальное выполнение функции.

IsFileGPS – опрос источника данных NMEA-потока.

long ***IsFileGPS*** *()*

Возвращаемое значение

0 – источник данных – устройство, принимающее информацию.

1 – источник данных – файл, записанный ранее при помощи программы «Запись сигналов».

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

SetStartUTCTime – установка времени, в которое должна произойти синхронизация по GPS.

long ***SetStartUTCTime*** *(unsigned __int64 uiTime)*

Параметры

uiTime – 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.

2.3.1.11 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 lChannel)*

Параметры

lChannel – номер канала, текущее время которого требуется узнать. Параметр *lChannel* 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 *lChannel < 0.*

-4 – параметр *lChannel* > (*QuanChan() – 1).*

-5 – значение частоты дискретизации по заданному каналу меньше нуля.

GetData – запрос данных по заданному каналу в буфер данных пользователя.

long ***GetData*** *(long lChannel, long lDecade, double dTime, long lSize, float* pData)*

Параметры

lChannel – номер канала, данные по которому требуется запросить. Параметр *lChannel* может принимать значение от *0* до (*QuanChan() – 1).*

lDecade – номер декады от 0 до 4.

dTime – момент времени, в который необходимо взять данные для обработки.

lSize – количество отсчетов данных для передачи. Параметр *lSize* 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

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).*

-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 *lSize ≤ 0* или *lSize ≥ DecadeBufferSize().*

-9 – номер декады *lDecade < 0.*

-10 – номер декады *lDecade > 4.*

-11 – момент времени *dTime* больше текущего времени для данного канала.

-12 – момент времени *dTime < 0.*

-13 – указатель на массив данных *pData = NULL.*

-14 - момент времени *dTime* меньше нижней границы кольцевого буфера накопленных данных для данного канала.

Сервер производит цифровую фильтрацию всех входных потоков данных со всех каналов АЦП. Например, оцифровка сигналов производится на частоте 25 кГц. Поток 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. *dTime* – 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(lChannel, 1, 10.5, 1250, pData);

передает в массив пользователя *pData* 1250 отсчетов потока выбранного канала с частотой оцифровки 2500 Гц, что составляет 0,5 с. В массиве *pData* 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 lChannel, long lDecade, double dTime, long lSize, VARIANT* pData)*

GetDataNet – запрос данных по заданному каналу в буфер данных пользователя в .net языках.

*long **GetDataNet** (long lChannel, long lDecade, double dTime, long lSize, long pData)*

Параметры

lChannel – номер канала, данные по которому требуется запросить. Параметр *lChannel* может принимать значение от *0* до (*QuanChan() – 1).*

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

lDecade – номер декады от 0 до 4.

dTime – момент времени, в который необходимо взять данные для обработки.

lSize – количество отсчетов данных для передачи. Параметр *lSize* 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

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).*

-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 *lSize ≤ 0* или *lSize ≥ DecadeBufferSize().*

-9 – номер декады *lDecade < 0.*

-10 – номер декады *lDecade > 4.*

-11 – момент времени *dTime* больше текущего времени для данного канала.

-12 – момент времени *dTime < 0.*

PutData – передача данных из буфера пользователя в буфер виртуального канала или буфер ЦАП.

long ***PutData*** *(long lChannel, double dTime, long lSize, float* pData)*

Параметры

lChannel – номер канала. Если функция используется для передачи данных в буфер виртуального канала, то параметр *lChannel* 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 *lChannel* может принимать значение от *0* до (*MaxQuanChanDac () – 1).*

dTime – момент времени, в который необходимо передать данные в буфер виртуального канала или буфер ЦАП.

lSize – количество отсчетов данных для передачи. Параметр *lSize* 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 *lChannel < 0*.

-5 – параметр *lChannel* > (*MaxQuanChanDac () – 1).*

-6 – передача данных в неподключенный канал ЦАП.

-7 – серверу данных не хватает памяти для организации буферов данных.

-8 – не хватает хэндлеров памяти в системе. Для устранения этой ошибки необходимо перезагрузить компьютер.

-10 – размер передаваемых данных *lSize ≤ 0*

-11 – момент времени *dTime < 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 *dTime* отстает от внутреннего времени, которое можно определить с помощью метода *CurrentTime().*

-105 – нет организованного программой виртуального канала с номером *lChannel.*

-205 – параметр *lChannel* > (*QuanChan() – 1).*

-106 – значение частоты дискретизации по заданному каналу меньше нуля.

-109 – размер передаваемых данных в виртуальный канал *lSize* ≥ *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 *dTime* 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(lChannel, 0, 10.5, 12500, pData);

передает в массив пользователя *pData* 12500 отсчетов потока выбранного канала с частотой оцифровки 25000 Гц, что составляет 0,5 с. В массиве *pData* 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(lChannel2, 0, 10.5, 12500, pData2);

Теперь можно просуммировать данные

```
for(int i = 0; i < 12500; ++i )
        pData3[i] = pData[i] + pData2[i];
```

и передать данные в сервер данных в виртуальный канал

myerr = PutData(lChannelVirt, 11.0, 12500, pData3);

Метод передал из массива *pData3*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 *dTime* 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.

long ***PutDataNet*** *(long lChannel, double dTime, long lSize, long pData)*

Параметры

lChannel – номер канала. Если функция используется для передачи данных в буфер виртуального канала, то параметр *lChannel* 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 *lChannel* может принимать значение от *0* до (*MaxQuanChanDac () – 1).*

dTime – момент времени, в который необходимо передать данные в буфер виртуального канала или буфер ЦАП.

lSize – количество отсчетов данных для передачи. Параметр *lSize* может принимать значение от 1 до *BufferSize(). BufferSize()* – метод для определения размера.

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

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 *lChannel < 0*.

-5 – параметр *lChannel* > (*MaxQuanChanDac () – 1)* для канала ЦАП или нет организованного программой виртуального канала с номером *lChannel* или *lChannel* > (*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 *lSize ≤ 0*

-11 – момент времени *dTime < 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 *dTime* отстает от внутреннего времени, которое можно определить с помощью метода *CurrentTime()*.

DecadeBufferSize – опрос размера буфера сервера данных по декаде.

long ***DecadeBufferSize*** *(long lDecade)*

Параметры

lDecade – 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 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 – номер декады *lDecade < 0*.

-4 – номер декады *lDecade > 4*.

Буфер данных также может переполняться. При этом чтобы избежать ошибок, нужно отлавливать эти моменты. Например, существует два варианта настройки системы:

Количество каналов q = 8	Количество каналов q = 4
Частота дискретизации f = 50000	Частота дискретизации f = 50000
Размер буфера size = 625000	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*** *()*

Возвращаемое значение

≥ 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 lChannel)*

Параметры

lChannel – номер канала, команду на прочтение очередной порции данных которому требуется послать. Параметр *lChannel* может принимать значение от *0* до (*QuanChan() – 1).*

Возвращаемое значение

0 – нормальное выполнение функции.

-1 – программа *ZETServer.ocx* выгружена из памяти процессора.

-2 – на момент вызова метода не было подключения к серверу данных.

-3 – программа *ZETServer.ocx* выгружена из памяти процессора.

-4 –данные находятся в процессе чтения из файла.

-5 – параметр *lChannel < 0.*

-6 – параметр *lChannel* > (*QuanChan() – 1).*

2.3.1.12 Вспомогательные информационные функции

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

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

≥ 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. 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 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.

CString ***GroupName*** *(long lChannel)*

Параметры

lChannel – номер канала, групповое имя которого требуется узнать. Параметр *lChannel* может принимать значение от *0* до (*QuanChan() – 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.

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.

CString ***DacGroupName*** *(long lChannelDac)*

Параметры

lChannelDac – номер канала ЦАП, групповое имя которого требуется узнать. Параметр *lChannelDac* может принимать значение от *0* до (*MaxQuanChanDac () – 1).*

Возвращаемое значение

непустая строка – групповое имя канала.

2.3.2 События

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 lParam)*

Параметры

lParam – параметр события. Если *lParam* = 0, то произошло подключение каналов АЦП, виртуальных каналов. Если *lParam* = 1, то произошло подключение каналов ЦАП. Если *lParam* = 2, то произошло изменение информации по виртуальному каналу.

FileMode – событие, которое происходит при смене режима работы сервера данных.

*void **FileMode** (long lParam)*

Параметры

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.

2.4 Примеры программирования на VBASIC

2.4.1 Пример SAMPLE_I

Запускаем Visual Basic 6.0 и создаем новый проект. На панели *ToolBox* нажатием правой кнопкой мыши и во всплывающем меню (рисунок 2.1) выбираем **Components…** (панель *ToolBox* можно вызвать из главного меню командой **View→ToolBox**). Другой способ открыть окно **Components** – выбрать меню **Projects → Components** (рисунок 2.2)

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

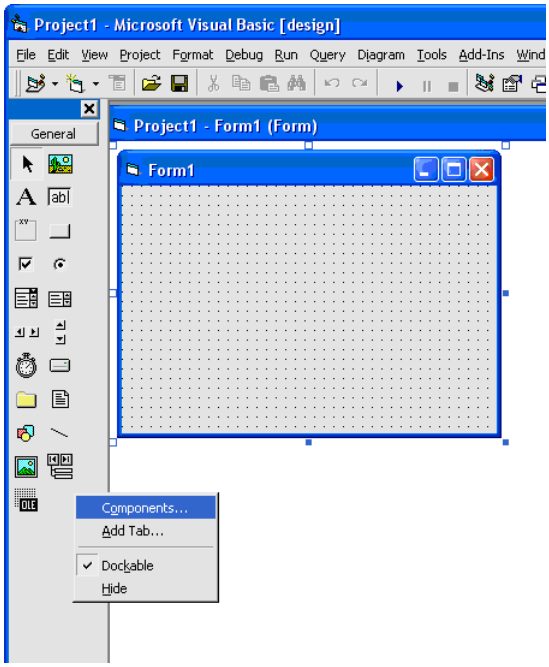


Рисунок 2.1

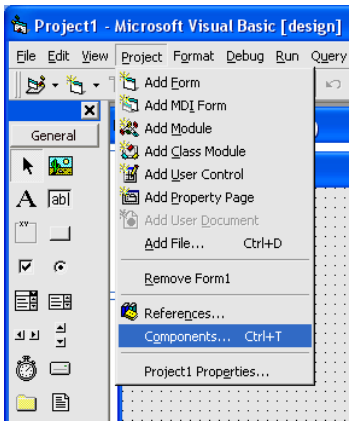


Рисунок 2.2

В окне **Components** (рисунок 2.3) 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 2.4).

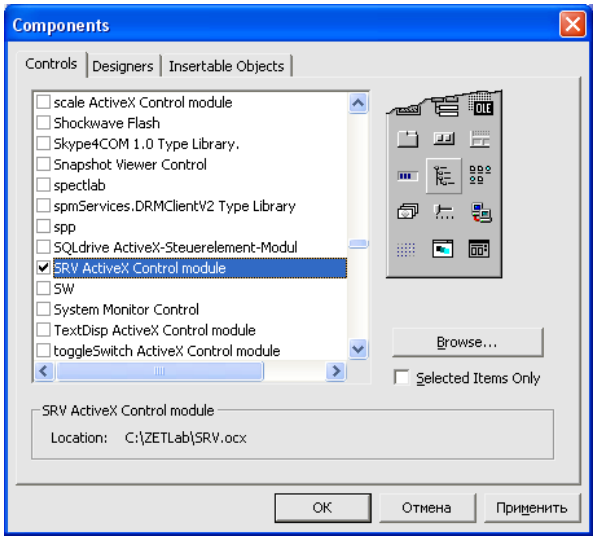


Рисунок 2.3

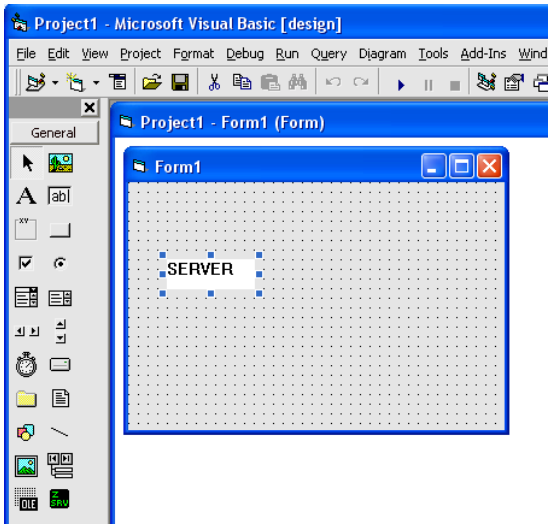
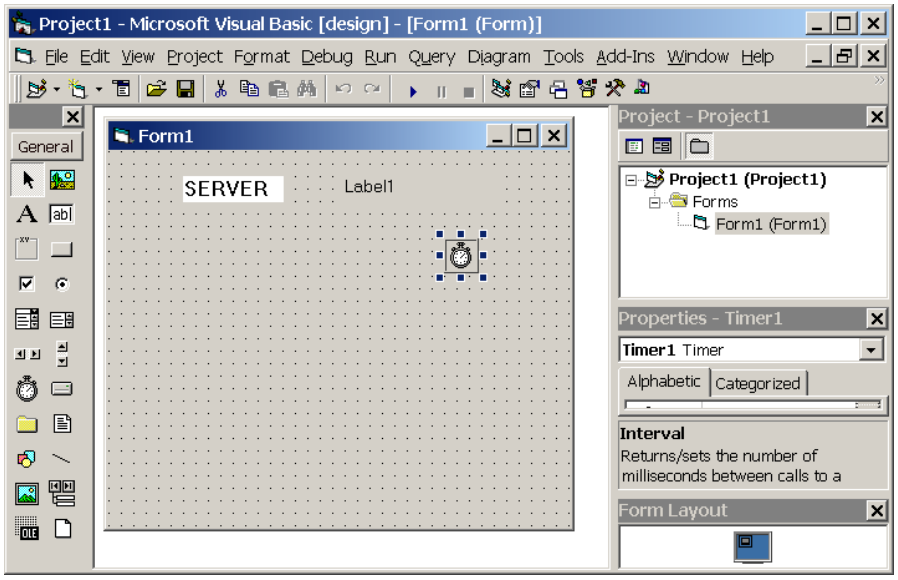


Рисунок 2.4

Устанавливаем аналогично Label и Timer

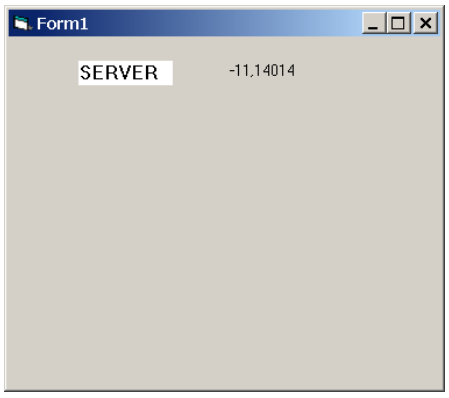


Дважды нажимаем на форму
и добавляем в код программы

```
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
   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.

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

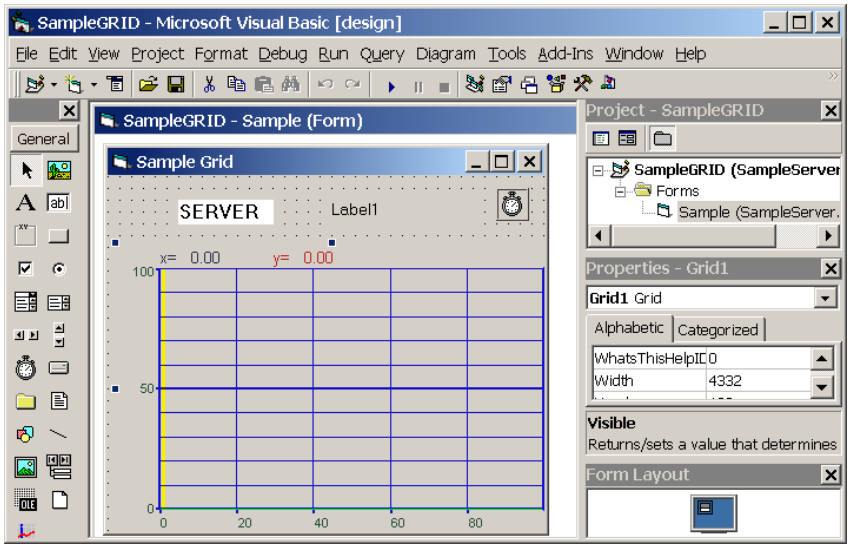


Для того чтобы во время выполнения программы не было видно надписи компонента «SERVER» необходимо свойство компонента SRV.Visible сделать False.

2.4.2 Пример SAMPLE_II

Подключение графики

Для того чтобы отобразить данные в виде графика в режиме осциллографа, необходимо добавить компонент Grid. Тогда внешний вид формы будет иметь вид:

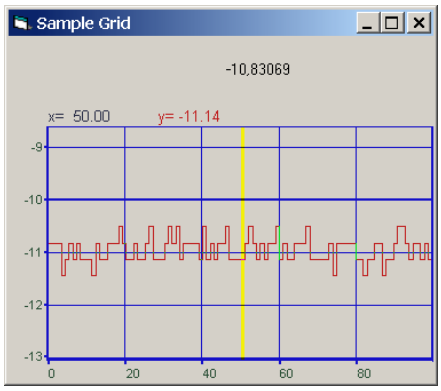


Необходимо добавить в процедуру обработки функцию построения графика.

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:



2.4.3 Пример SAMPLE_III

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()

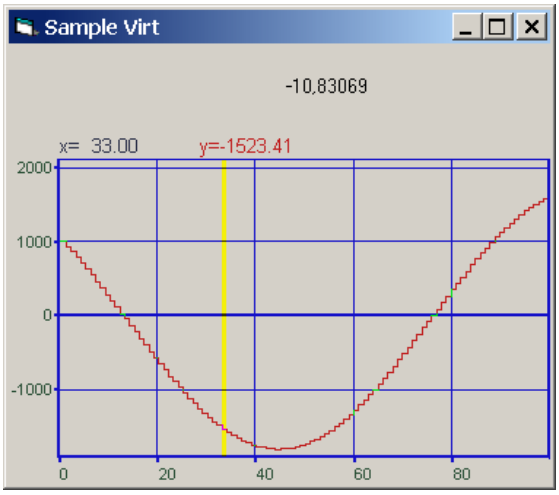
Исправлена процедура обработки сигналов. Добавлены массивы для обработки. Размер выбирается из частоты дискретизации АЦП и заданного интервала обработки.

```
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
      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
```


ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

```
   End If
End If
End Sub
```

Внешний вид программы



Для просмотра и анализа полученного сигнала можно использовать программы ZETLab осциллографы, анализаторы, вольтметры. Созданный виртуальный канал будет доступен всем программам.

2.4.4 Пример SAMPLE_IV

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.

```
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
```

2.5 Примеры программирования на .Net платформе

В данной статье на примере компонента ZETServer Control (SRV.ocx) проведем интеграцию ActiveX компонента Zet, с Вашей программой.

2.5.1 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’.

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

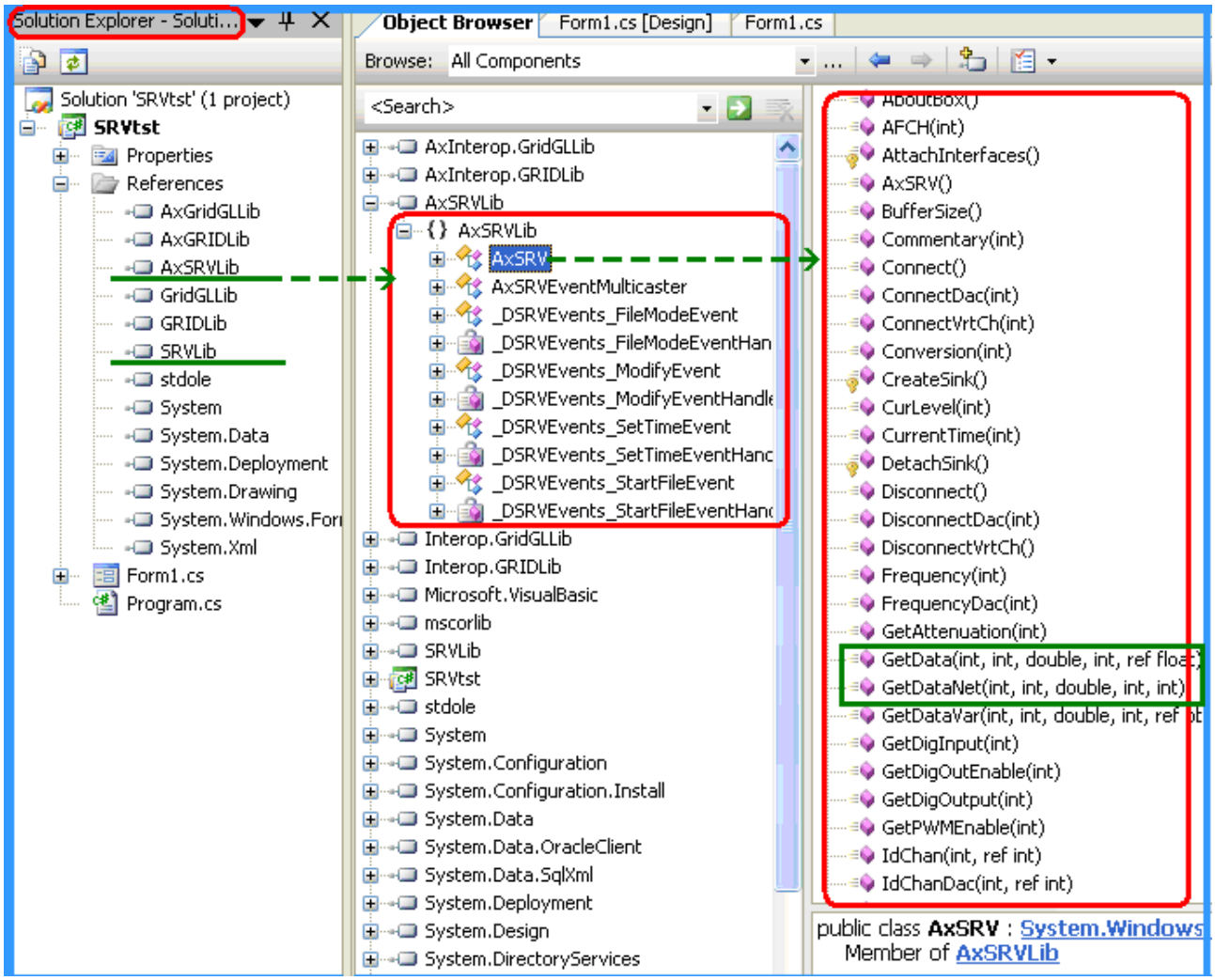


Рисунок 1'. Представление осх файлов в виде dll и их содержимое.

2.5.2 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 .ocx 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 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 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. AxSrvLib и SrvLib.
8. О дальнейших действиях с подключенным компонентом, а так же пример использования будут приведены в следующих главах.

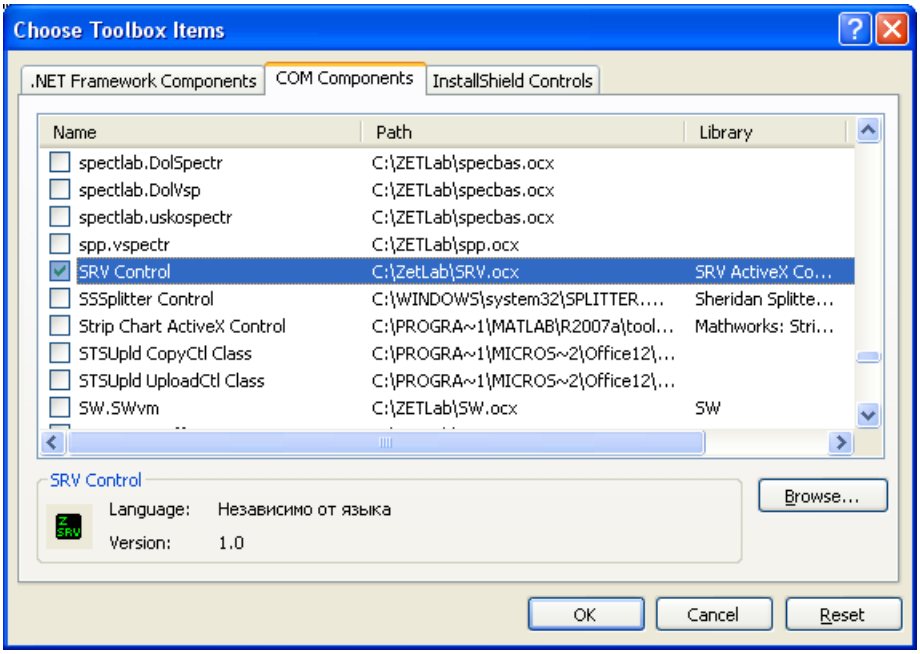


Рисунок 2'. Подключение компонента.

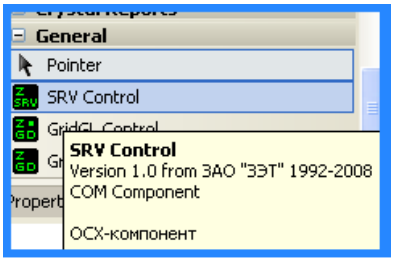


Рисунок 3'. Внешний вид ToolBox с подключенным компонентом ZETServer.

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

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

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 4'.

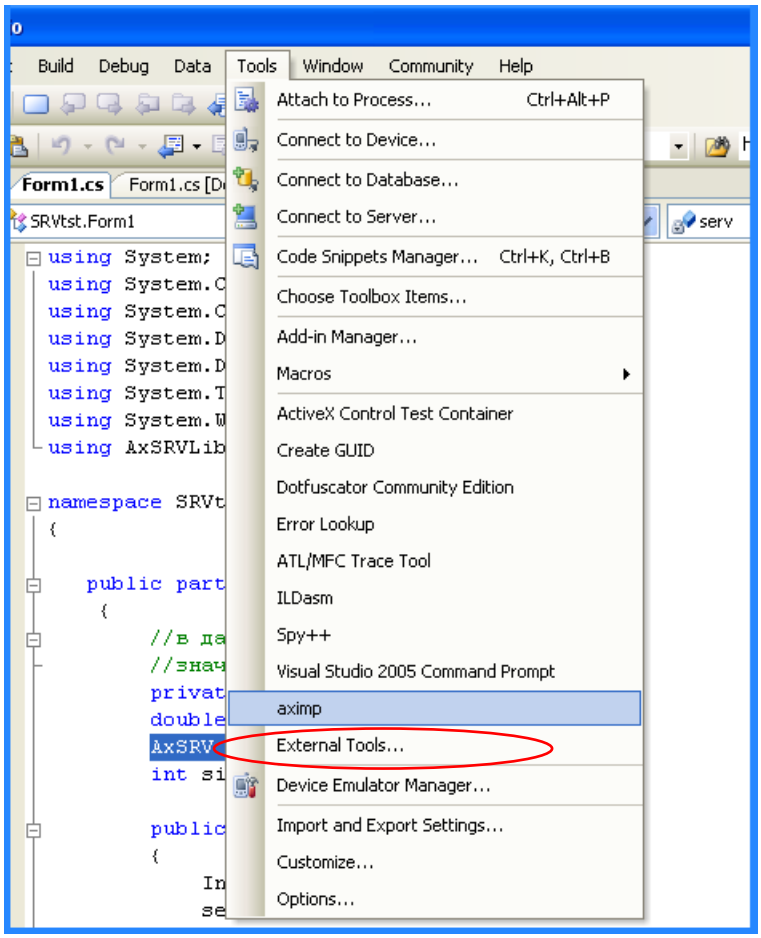


Рисунок 4'. Внешний вид меню Tools после подключения утилиты aximp.

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 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.

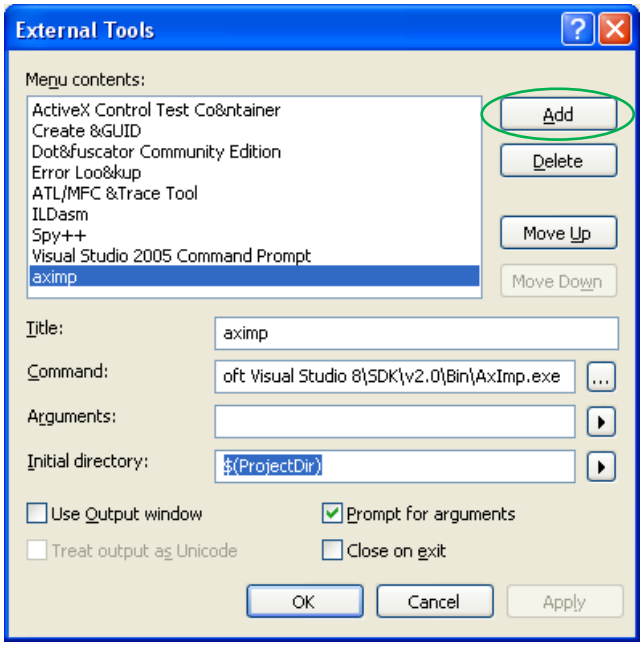


Рисунок 5'. Добавление Утилиты aximp.exe

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 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 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.

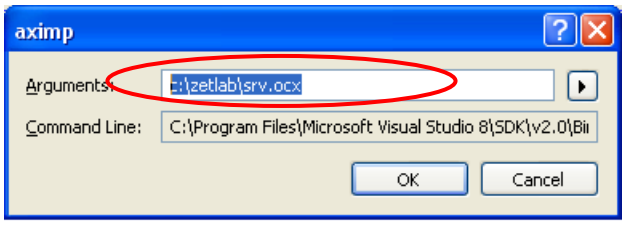


Рисунок 6. Создание обертки ActiveX Wrapper.

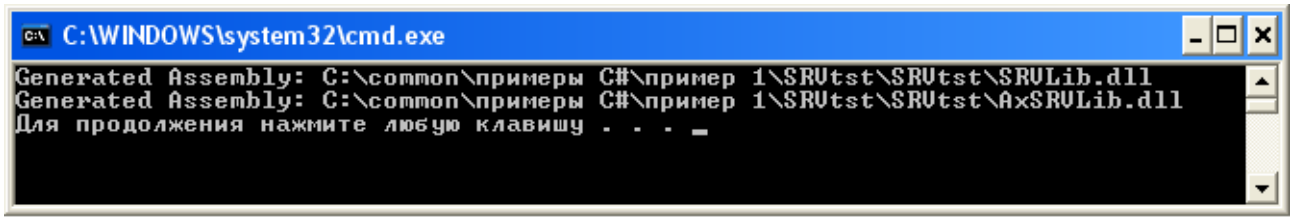


Рисунок 7. Работа утилиты Aximp.

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

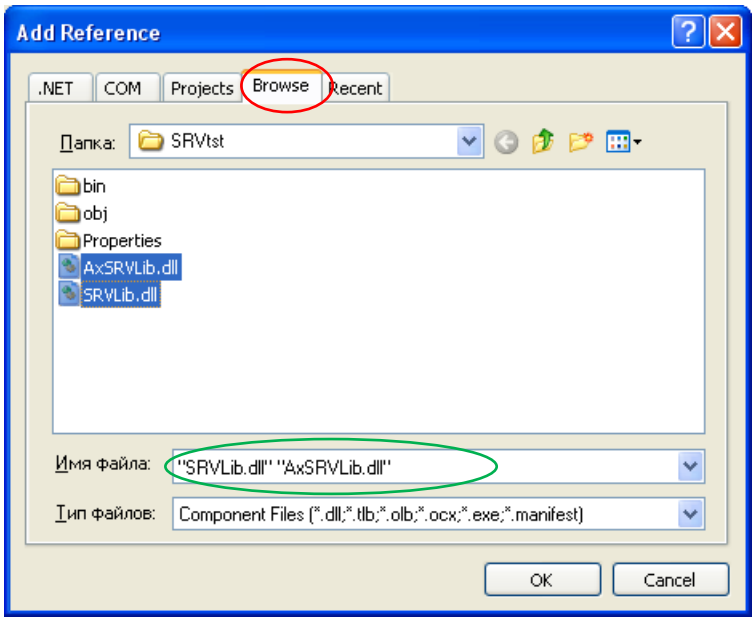


Рисунок 8. Добавление созданных файлов к проекту.

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 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.

2.5.3 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
                  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.

2.5.4 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
```


ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

```
            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);
                    axGrid1.PaintNet((int)p);
                }
            }
        }
```

Листинг 3. Применение *.*Net функций и указателей на массив в коде unsafe.

2.5.5 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;

     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
                 MyTime = MyTime + 0.12;
```


ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

```
                    prgTimelbl.Text = MyTime.ToString();
                    //выводим на экран значение одного отсчета
                    dataVal.Text = p[0].ToString();

                    //вывод полученных значений на график
                    axGridGL1.PaintNet((int)p);
                    axGrid1.PaintNet((int)p);

                }

            }

        }

    }
}
```

Листинг 4. Пример программы на C#

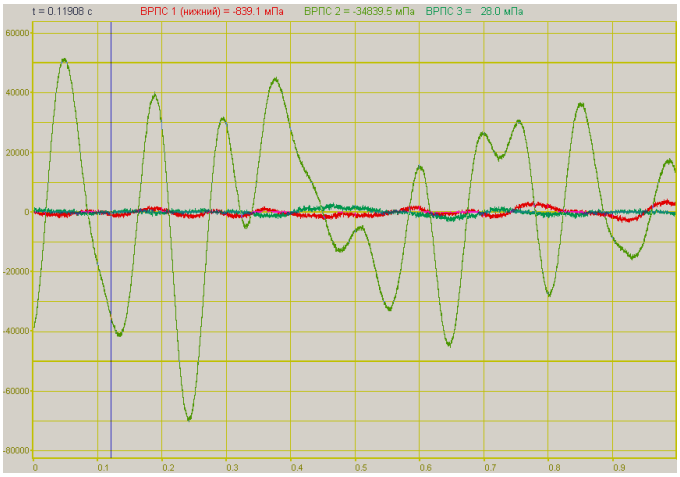
3 Описание GRID.OCX

3.1 Назначение GRID.OCX

Компонента **Grid** 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.

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.



3.1.1 Управление курсором и масштабирование графиков

Перемещение курсора на нужную абсциссу осуществляется несколькими способами:

- поставить курсор «мыши» на нужную координату и нажать на левую кнопку «мыши»;
- при активном окне программы при помощи ролика «мыши»;
- при активном окне программы при помощи стрелок, расположенных на цифровом поле клавиатуры при включенном индикаторе ***Num Lock***;
- при активном окне программы при помощи кнопок клавиатуры: **«A»** - влево; **«W»** - вверх; **«D»** - вправо; **«S»** - вниз.

Масштабирование числовой оси происходит при помощи манипулятора «мышь». Перемещая указатель «мыши» вдоль осей, указатель, в зависимости от своего местонахождения, меняет внешний вид. Надо дождаться, когда указатель «мыши» примет нужный внешний вид и, либо щелкнуть левой кнопкой «мыши», либо прокрутить «ролик». Растяжение или сжатие графиков происходит при помощи указателя вида: ↔ , ⇹ – для горизонтальной оси и ↕ , ⇳ – для вертикальной оси.

Сдвинуть графики вправо-влево или вверх вниз можно при помощи указателя ←, → – для горизонтальной оси и ↑ , ↓ – для вертикальной оси. Если поставить «мышь» в начало координат, то указатель примет вид ⤢. При нажатии на указатель такого вида выполняется команда «автомасштабирование» по оси Y (автомасштабирование происходит по уровню сигнала).

Для копирования графика спектра при активном окне программы нажмите комбинацию кнопок клавиатуры **Ctrl + C**. График запишется в буфер Clipboard в формате *.bmp. Вставить график в любой текстовый документ можно одновременным нажатием на кнопки клавиатуры **Ctrl + V** или нажатием на правую кнопку мыши и выбором в появившемся меню команды **[Вставить]**.

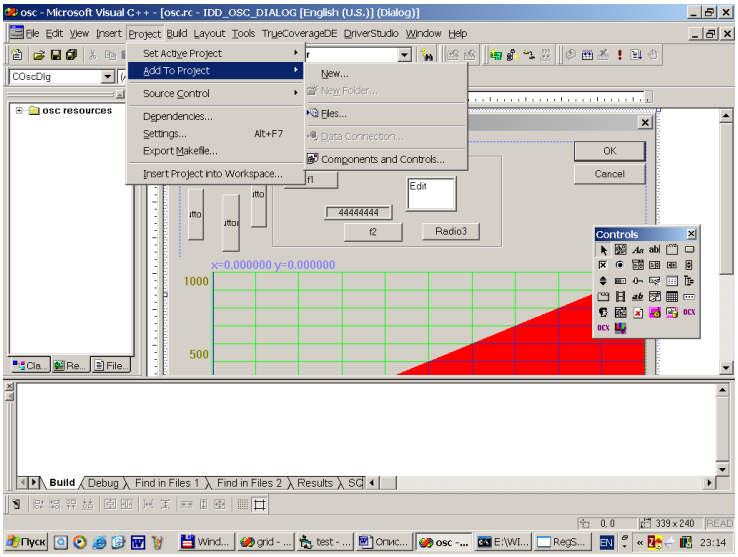
Для копирования ***сопроводительной информации*** в буфер Clipboard в формате редактора текста Word при активном окне программы нажмите на кнопку клавиатуры **T** (латинская) или **Е** (русская). Вставить эту информацию можно в любой документ Word можно одновременным нажатием на клавиши **Ctrl + V** или нажатием на правую кнопку мыши и выбором из появившегося меню команды **Вставить.**

Сопроводительная информация имеет следующую структуру: в первой строке пишется заголовок окна, в данном случае название спектра и название отображаемого канала. Во второй и третьей строках – измеряемые величины, а именно значение частоты и значение уровня, которые показывает курсор. Если включены дополнительные графики, то их значения пишутся в следующих строках.

Для копирования ***всей цифровой информации*** видимой части графика в буфер Clipboard в формате Excel при активном окне программы нажмите кнопку клавиатуры **N** (латинская) или **Т** (русская). Вставить эту информацию можно в любой документ Excel одновременным нажатием на клавиши клавиатуры **Ctrl + V** или нажатием на правую кнопку мыши и выбором из появившегося меню команды **Вставить**. Мы получим следующую информацию: сначала идет ***сопроводительная информация*** (см. абзац выше), далее идут данные в формате Excel.

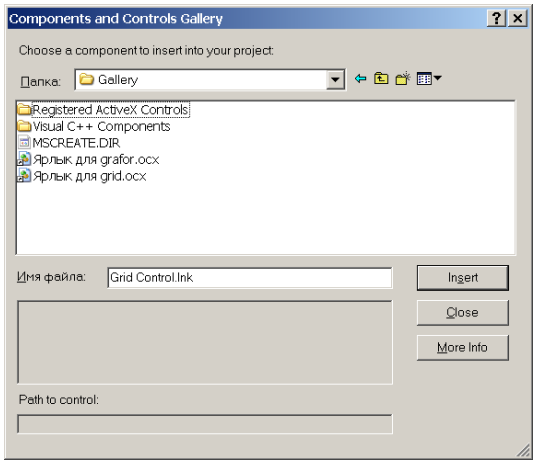
3.2 Установка компонента в программу пользователя

Для работы с компонентом необходимо установить его в проект:
В проекте VISUAL C++ v6.0:



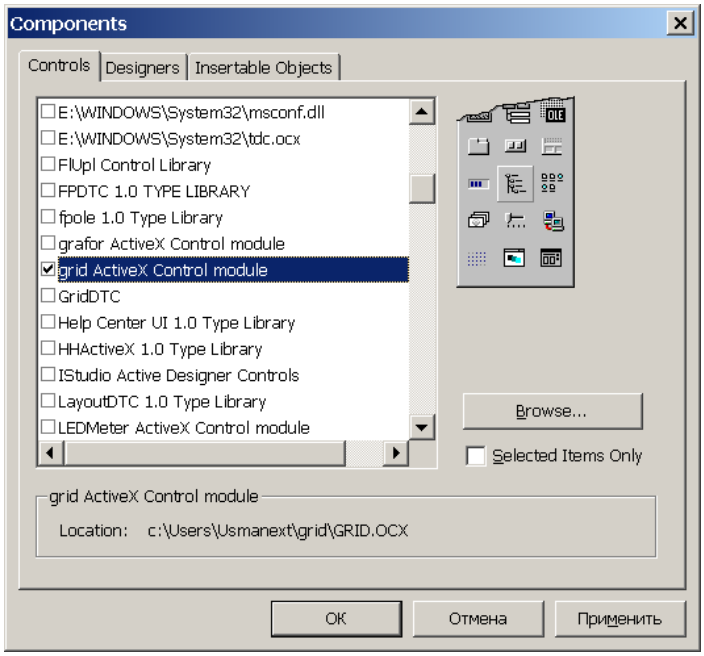
ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

После выбора в меню
Project
Add To Project ->
Components and Controls
Появится окно:



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



Необходимо выбрать необходимый компонент и установить напротив него знак **V**.
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.

3.3 Описание свойств, методов и событий

3.3.1 Свойства

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.

3.3.1.1 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.

ClrCrs – цвет курсора.

ClrDig – цвет цифр на сетке.

ClrFon – цвет фона.

ClrGrd – цвет сетки.

ClrGrf – цвет графика. Для каждого графика цвет задается отдельно. Графики выбираются при помощи свойства *Ind*.

ClrLeg – цвет легенды.

3.3.1.2 Параметры графиков

Number – количество отображаемых графиков. Количество одновременно отображаемых графиков в одних осях не может превышать 20.

Ind – индекс (номер) графика. Индекс может принимать значения от 0 до Number-1.

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

Size – количество отсчетов в массиве данных для отображения графика. Параметр задается для всех графиков одновременно. Размер не может быть меньше или равен 0. Размер не может быть больше 500000.

NumVisiblePoints – количество видимых точек. Отображаются только указанное количество точек. Параметр задается для всех графиков одновременно.

Log – логарифмическая шкала по Y. Параметр может принимать значения 0 или 1.

0 – линейный масштаб.

1 – логарифмический масштаб.

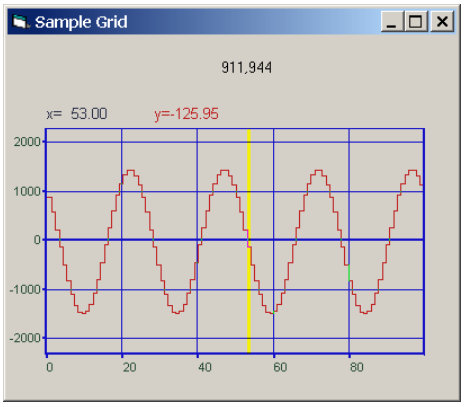
ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

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.

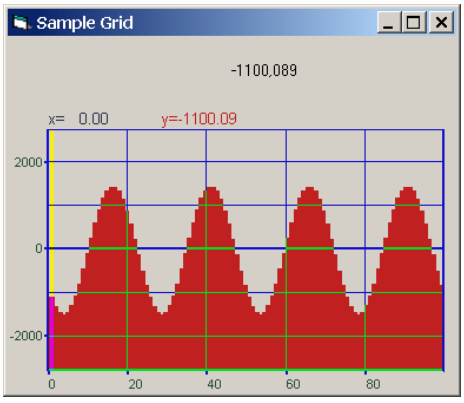
3.3.1.3 Представление графиков

TypeLine – тип линий графика. Для каждого графика параметр задается отдельно. Графики выбираются при помощи свойства *Ind*.

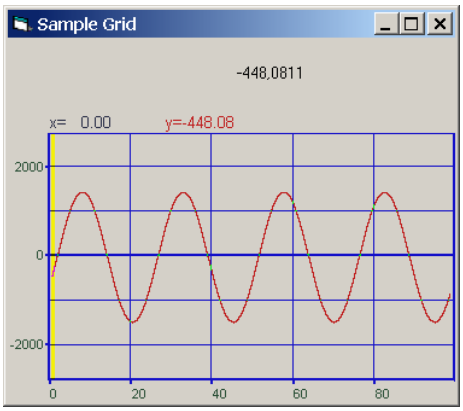
0 – горизонтальными линиями. График рисуется в виде ступенек от точки к точке.



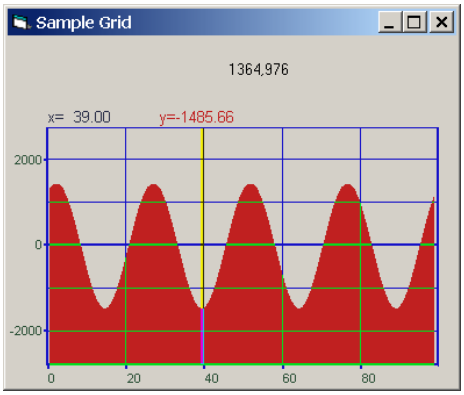
1 – сплошной горизонтальными линиями. График рисуется с заполнением в пространстве под графиком в виде столбиков.



2 – линиями. График рисуется в виде ломаной линии от точки к точке



3 – линиями с заполнением под графиком



4 – интерполяция сплайном.

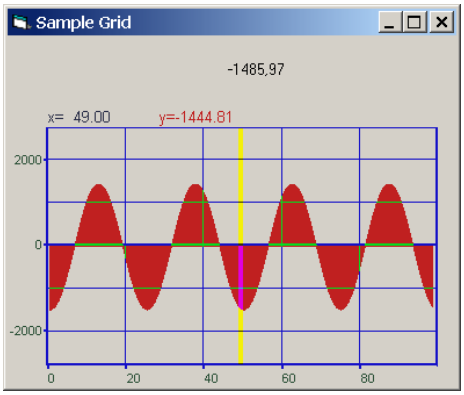
5 – интерполяция по теореме Котельникова-Шеннона.

TypeGrf – тип отображаемого графика. Для каждого графика параметр задается отдельно. Графики выбираются при помощи свойства *Ind*. Параметр может принимать значения 0 или 1.

0 – положительный

1 – знакопеременный.

Этот параметр влияет на изображение графика, когда график рисуется с заполнением, для *TypeLine*=1 или 3.



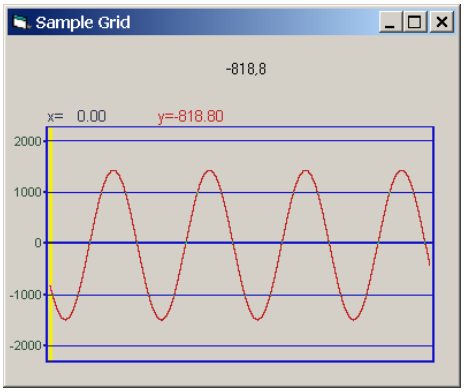
Этот рисунок сделан при *TypeGrf* равном 1. Все верхние рисунки сделаны при *TypeGrf* равном 0.

TypeAbs – тип сетки.

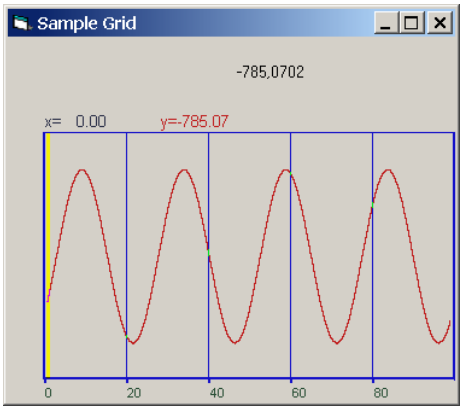
0 – рисуется сетка и по X и по Y равномерная шкала по X

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

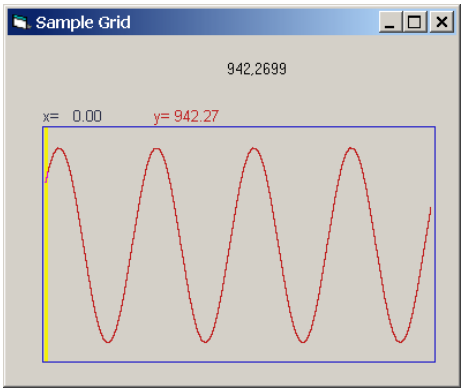
1 – не рисуются вертикальные линии и вертикальные разметки графика.



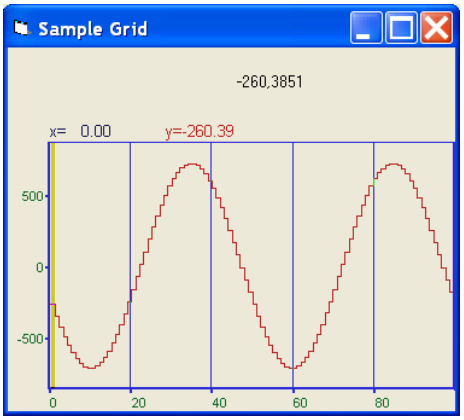
4 – не рисуются горизонтальные линии и горизонтальные разметки.



Комбинация параметров, например, 1 и 4, т.е. параметр равен 5, не рисует сетку.

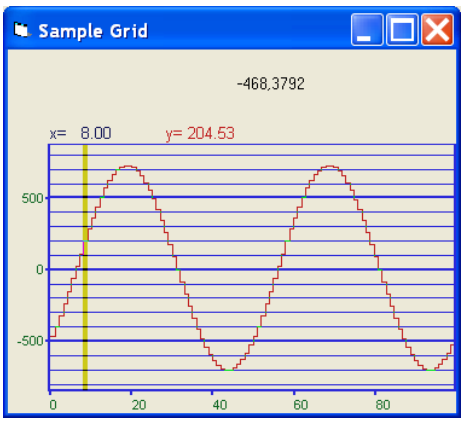


8 – рисуются надписи разметки сетки при отсутствии самой горизонтальной сетки.



TypeAbs=12

16 – рисуются надписи разметки сетки при отсутствии вертикальной сетки.



TypeAbs=17

2 – произвольная сетка по *X*. При этом разметка по оси *X* загружается методом *AltC(float* buffer).*

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.

UporNis – минимальная величина по разрешению по координате *Y*, которая может отобразиться на графике. Это свойство работает при включенных ограничениях по оси *Y*.

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

Xfirst – начальное значение по координате *X*.

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.

MathLX – область отображения данных по оси *X*. Левая граница отображения данных.

MathDX– область отображения данных по оси *X*. Длина отображения данных.

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

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.

MathLY – область отображения данных по оси *Y*. Нижняя граница отображения данных.

MathDY – область отображения данных по оси *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* .

Luminescence – эффект, подобный послесвечению ЭЛТ осциллографа. Полупрозрачность предыдущего кадра. Значения от 0 до 255.

0 – непрозрачное,

255 – отсутствует.

markPlainData – при большом увеличении показывать маркеры неинтреполированных данных.

0 – не показывать,

1 – показывать.

3.3.2 Методы

Методы можно использовать только во время выполнения программы.

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:

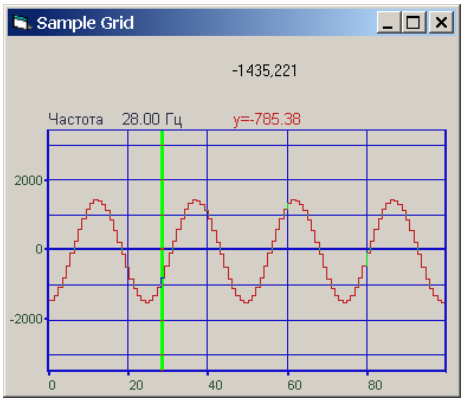
float Buffer* – адрес массива данных в программе пользователя. В компоненту копируется количество данных определенное свойством *Size*.

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(LPCTSTR fmt) – метод передает в компонент строку для форматного вывода информации по координате *X* в легенде графика. Формат вывода соответствует требованиям форматного вывода языка С.

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

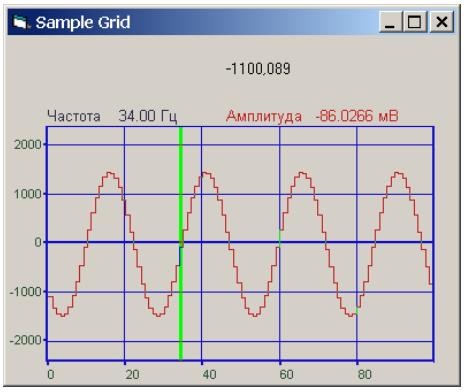
%e – общий формат представления числа в плавающей запятой в виде числа с экспонентой (*1000 = 1e3, 0.001 = 1e-3*)

%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(LPCTSTR fmt) – метод передает в компонент строку для форматного вывода информации по координате *Y* в легенде графика. Формат вывода соответствует требованиям форматного вывода языка С. 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.

void Formatxy(LPCTSTR fmt) – метод передает в компонент строку для форматного вывода информации в Clipboard. Формат вывода соответствует требованиям форматного вывода языка С. Это может быть произвольная пользовательская информация о данном графике.

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

*void AltC(float *buffer)* – 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
……

short Display(void) – метод перерисовывает весь элемент – фон, сетку, надписи и все графики.

void autoScaleY(void) – однократно применяет автомасштаб по оси Y.

void PushToClipBoard(void) – метод позволяет сохранить изображение графика в Clipboard в графическом (bmp) формате.

void savePng(LPCTSTR FileName) – сохраняет изображение графика в PNG-файл, параметр – имя файла.

void SetmarkPlainData(int value) – отобразить или скрыть маркеры данных в виде кружочков в обольшом масштабе.

3.3.3 События от компоненты

DblClick(void) – событие возникает при двойном нажатии на кнопку манипулятора «мышь», при нахождении мыши над графиком.

KeyDown(SHORT KeyCode, SHORT Shift)* – событие возникает при нажатии на клавишу клавиатуры, когда фокус находится на компоненте.

KeyCode – код клавиши

Shift – код нажатия служебных клавиш <*Ctrl*>,<*Shift*>,<*Alt*>.

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.

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)

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.

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)

MouseWheel(void) – событие возникает при вращении ролика манипулятора типа «мышь».

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

4 Описание GRIDGL.OCX

4.1 Назначение

Компонент **GridGL.ocx** является расширенной версией компонента **Grid.ocx**. Компонент **GridGL.ocx** поддерживает многие свойства, методы и события компоненты **Grid** и имеет дополнительные свойства.

AltCoords - Равномерные и неравномерные координаты 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

$$x_i = cord[i]$$

$$y_i = buffer[i]$$

$$i = 0...Size - 1$$

FontName – название шрифта для всех надписей

FontSize – размер шрифта

LineWidth – толщина линии графика. Для каждого графика толщина линии задается индивидуально. Номер графика выбирается свойством Ind.

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.

Reference – значение для расчета уровней 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.

ShowScalePropPage -

Osc – 1 - Вид осциллографа нельзя включить логарифмический масштаб по X, 0 – нормальный вид.

Spectr – 1 – вид спектра, можно включить логарифмический масштаб по У, но не отображаются отрицательные координаты по осям, 0 – нормальный вид.

TextXAxis -

TextYAxis -

TypeFill -

TypeAbs -

TypeGrf -

TypeLine -

TypeXAxis 0-равномерно
1-логарифмическое
2-долеоктавное

TypeYAxis 0-равномерное
1-логарифмическое
2-в дБ

TypeYData 0 - данные поступают в дБ;
1 - данные поступают в единицах измерения;

getStartPoint – возвращает индекс первой видимой точки;

getEndPoint – возвращает индекс первой видимой точки.

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

5 Описание PLOTTER.OCX

AutoMX (bool) – автоматическое масштабирование (bool)

CommentIND (long) – порядковый номер комментария

CommentSTR (string) – срока комментария

Цвета:

ClrCur (long) – цвет курсора

ClrFon (long) – цвет фона графика

ClrGrd (long) – цвет сетки

ClrGrfBeg (long) – начальный цвет графика

ClrGrfEnd (long) – конечный цвет графика

ClrLeg (long) – цвет «шапки» графика

ClrNum (long) – цвет цифр по осям

DeltaT (single) – временной интервал (скорость) изменения графика

FixedTime (bool) – фиксированное (true) или неограниченное (false) время отображения графика

NumPoints (long) – количество точек в графике

TypeCoord – тип координат:
1 – 2-мерный (XT)
2 – 2-мерный (YT)
4 – 2-мерный (XY)
8 – 3-мерный (XYT)

TypeLine (long) – тип линий сетки

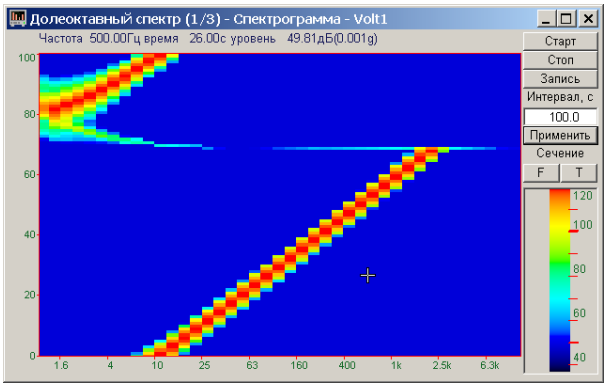
6 Описание GRAMMA.OCX

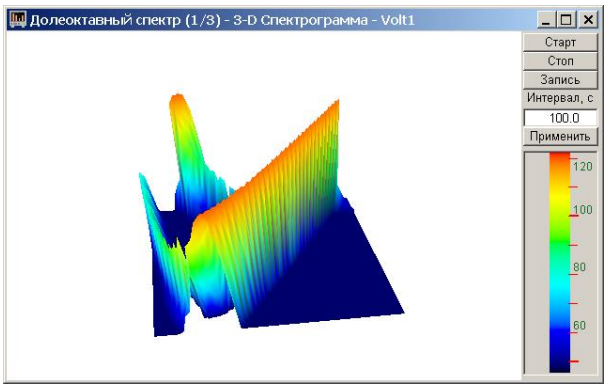
6.1 Назначение

Компонента **Gramma** 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.

Компонента 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 Clipboard в графическом виде для создания отчетов в редакторах Word и Excel.





6.1.1 Управление курсором и масштабирование графиков

Перемещение курсора на нужную координату осуществляется несколькими способами:

- поставить курсор «мыши» на нужную координату и нажать на левую кнопку «мыши»;
- при активном окне программы при помощи ролика «мыши»;

- при активном окне программы при помощи стрелок, расположенных на цифровом поле клавиатуры при включенном индикаторе ***Num Lock***;
- при активном окне программы при помощи кнопок клавиатуры: **«A»** - влево; **«W»** - вверх; **«D»** - вправо; **«S»** - вниз.

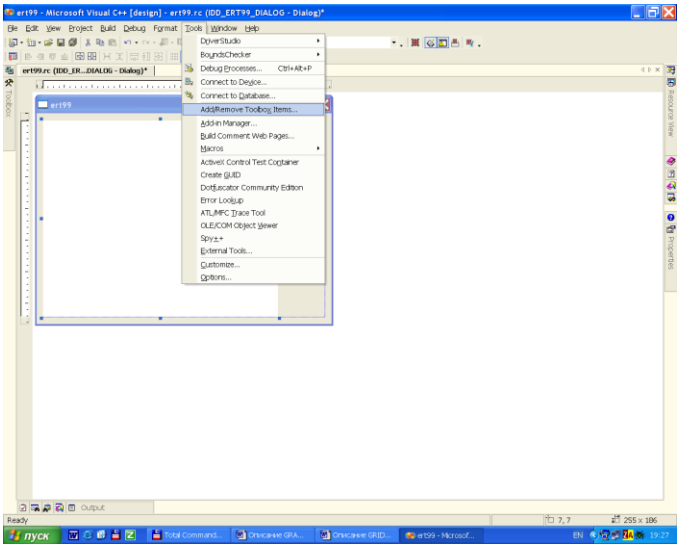
Масштабирование числовой оси происходит при помощи манипулятора «мышь». Перемещая указатель «мыши» вдоль осей, указатель, в зависимости от своего местонахождения, меняет внешний вид. Надо дождаться, когда указатель «мыши» примет нужный внешний вид и, либо щелкнуть левой кнопкой «мыши», либо прокрутить «ролик». Растяжение или сжатие графиков происходит при помощи указателя вида: ↔ , ⇹ – для горизонтальной оси и ↕ , ⇳ – для вертикальной оси.

Сдвинуть графики вправо-влево или вверх вниз можно при помощи указателя ← , → – для горизонтальной оси и ↑ , ↓ – для вертикальной оси. Если поставить «мышь» в начало координат, то указатель примет вид ⤢ . При нажатии на указатель такого вида выполняется команда «автомасштабирование» по оси *Z* (автомасштабирование происходит по уровню сигнала).

Для копирования графика спектра при активном окне программы нажмите комбинацию кнопок клавиатуры **Ctrl + C**. График запишется в буфер Clipboard в формате *.bmp. Вставить график в любой текстовый документ можно одновременным нажатием на кнопки клавиатуры **Ctrl + V** или нажатием на правую кнопку мыши и выбором в появившемся меню команды **[Вставить]**.

6.2 Установка компонента в программу пользователя

Для работы с компонентом необходимо установить его в проект:
В проекте VISUAL C++ v7.0:



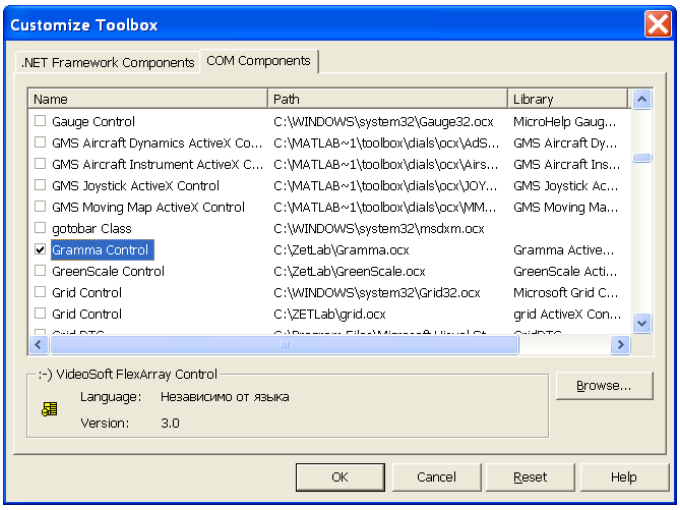
После выбора в меню

Tools->

Add/Remove Toolbox Items ->

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

Появится окно:



В этом окне на закладке <COM Components> необходимо найти компонент Gramma Control и выбрать его. После этого компонент появится в окне ToolBox при проектировании диалоговых окон.

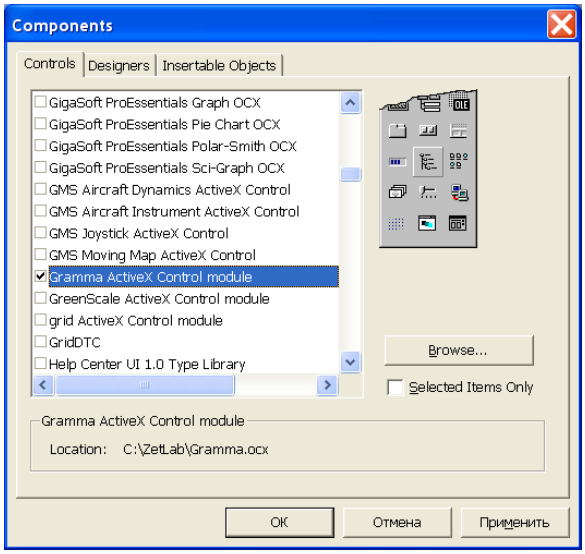
В проекте VISUAL BASIC

После выбора в меню

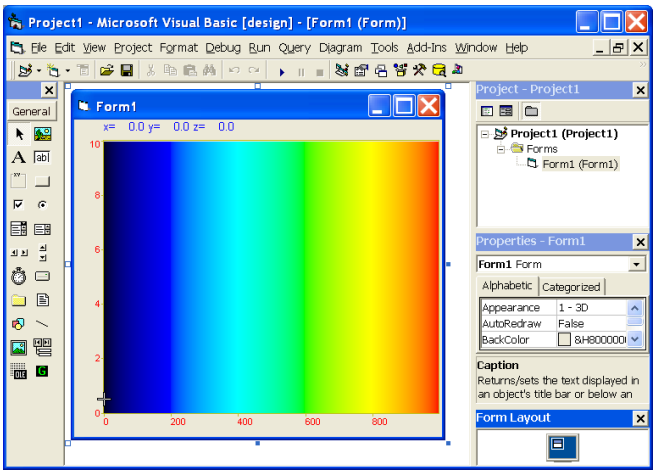
Project->

Components…

появится окно



Необходимо выбрать необходимый компонент и установить напротив него знак **v.**
После этого ActiveX-элемент появится на поле компонентов.



6.3 *Описание свойств, методов и событий*

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.

6.3.1 *Свойства*

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.

6.3.1.1 *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.

ClrDig – цвет цифр разметки.

ClrFon – цвет фона.

ClrGrd – цвет разметки и окантовки графика.

ClrLeg – цвет легенды.

Black – цветное или черно-белое изображение.
0 – цветное изображение.
1 – черно-белое изображение.

6.3.1.2 *Параметры графиков*

Size – количество отсчетов в строке для отображения графика. Размер не может быть меньше или равен 0. Размер не может быть больше 50000.

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

Number – количество отображаемых строк. Количество строк не может превышать 200.

Ind – индекс (номер) строки. Индекс может принимать значения от 0 до Number-1.

DeltaT – интервал между строками в единицах измерения по оси Y. Например, при отображении сонограммы отображаются последовательность спектров получаемых каждые 0.1 с и тогда *DeltaT* = 0.1.

Log – логарифмическая шкала по Y. Параметр может принимать значения 0 или 1.
0 – линейный масштаб.
1 – логарифмический масштаб.

TypeAbs – тип графика.
– плоский график с курсором с курсором.
– трехмерный график, типа водопад без курсора.

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.

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.

6.3.1.3 Представление графиков

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.

UporNis – минимальная величина по разрешению по координате *Z*, которая может отобразиться на графике. Это свойство работает при включенных ограничениях по оси *Z*.

UporVerh – максимальная величина по координате *Z*, которая может отображаться на графике. Это свойство работает при включенных ограничениях по оси *Z*.

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

Xfirst – начальное значение по координате *X*.

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.

MathLX – область отображения данных по оси *X*. Левая граница отображения данных.

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.

6.3.2 Методы

long Paint(float Buffer)* – метод, позволяет передать данные для отображения графика и перерисовать график. 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 Formatxy(LPCTSTR 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 курсора на графике и уровень. Формат вывода соответствует требованиям форматного вывода языка С.

Например, обращение вида:

FormatX(“Частота %7.2f Гц Время %7.2f с Уровень %7.3f В”)

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.

short Display(void) – метод перерисовывает весь элемент - фон, сетку, надписи и все графики.

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

void PushToClipBoard(void) – метод, позволяет сохранить изображение графика в Clipboard в графическом (bmp) формате.

long SaveGLDataToFile(LPCTSTR FileName) – метод, позволяет сохранить массив данных в компоненте в файл. 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(LPCTSTR 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.

CommentInd – индекс строки комментария для записи в файл.

CommentSTR – строка комментария для записи в файл.

float GetPointAt(long X, long Y) – метод позволяет прочитать значение данных по координатам *X* и *Y*.

6.3.3 События от компоненты

KeyDown(SHORT KeyCode, SHORT Shift)* – событие возникает при нажатии на клавишу клавиатуры, когда фокус находится на компоненте.
KeyCode – код клавиши
Shift – код нажатия служебных клавиш <*Ctrl*>, <*Shift*>, <*Alt*>.

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.
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)

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.
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)

MouseWheel(void) – событие возникает при вращении ролика манипулятора типа «мышь».

7 Описание ZetWindowsLog.ocx

7.1 Назначение

Компонент ZetWindowsLog 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.

7.2 Описание свойств, методов и 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
(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

8 Программный модуль *ZETAnpRecorder.ocx* для записи результатов

8.1 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.

8.2 Установка компонента

Для работы с модулем *ZETAnpRecorder* его необходимо сначала установить в программу. При загрузке программы необходимо установить имя файла и затем записывать в него результаты данные.

Для установки компонента *ZETAnpRecorder* в проект *Microsoft Visual Studio 2010*, необходимо кликнуть правой кнопкой мыши на форме диалога и из появившегося меню выбрить пункт *Insert ActiveX Control…*(рисунок 11.1)

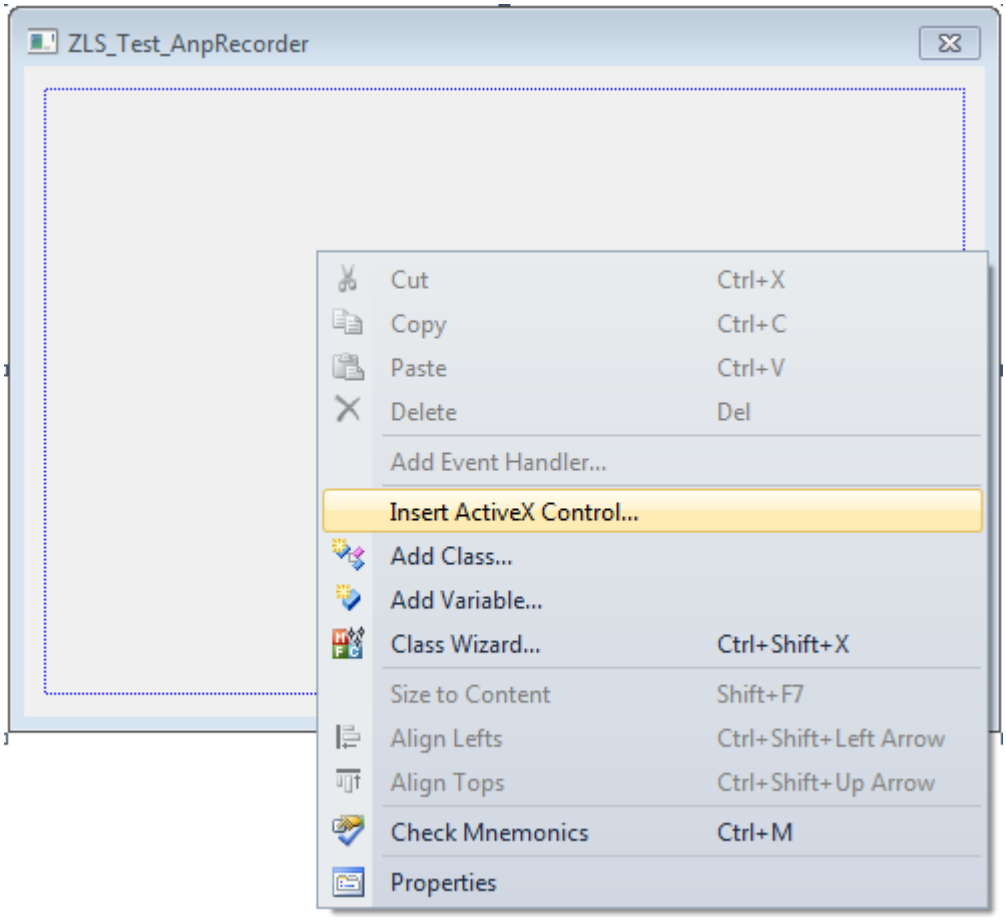


Рисунок 8.1

Из появившегося диалогового окна следует выбрать компонент *ZETAnpRecorder Control* и нажать *OK* (рисунок 8.2).

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

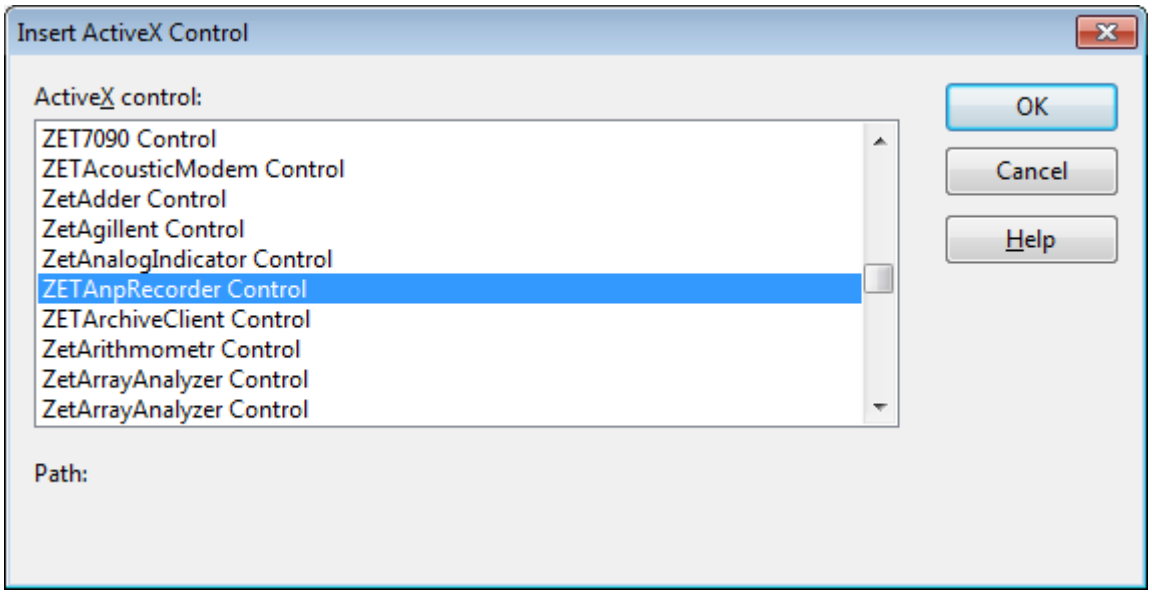


Рисунок 8.2

После этого компонент *ZETAnpRecorder.ocx* появится на форме диалога (рисунок 8.3).

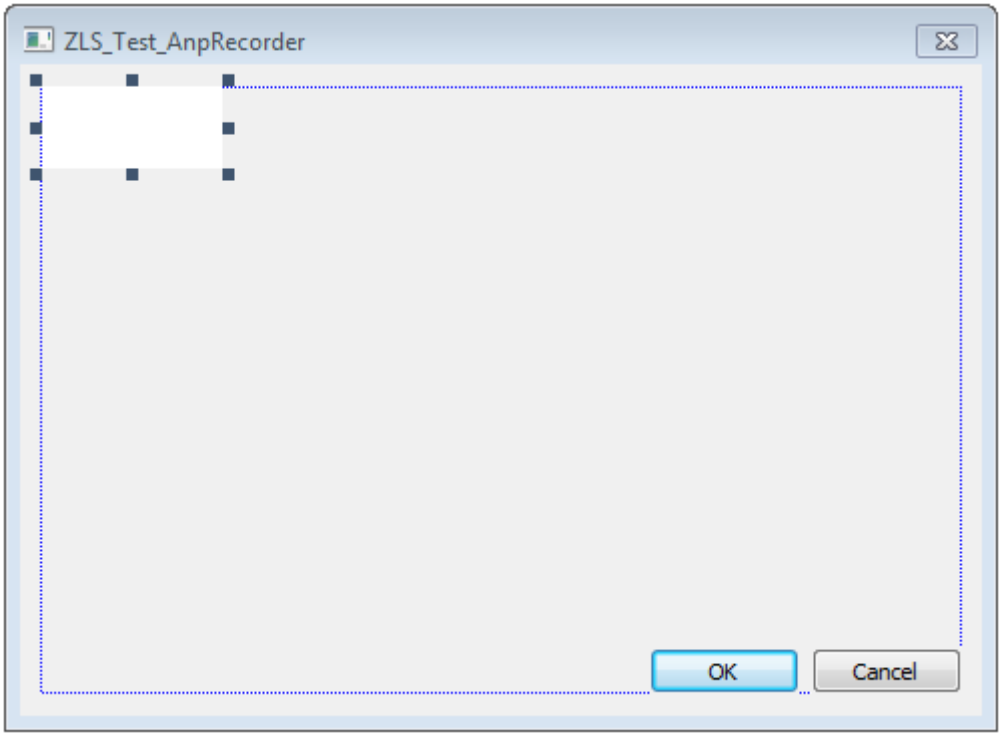


Рисунок 8.3

Для того чтобы компонента не была видна на форме во время выполнения программы, необходимо установить свойство *Visible* равным *False*.

8.3 Описание методов и событий

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.
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

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

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.
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.

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.

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

9 Описание программных модулей кнопок и индикаторов.

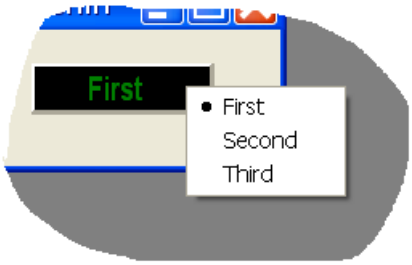
9.1 Zbutton

9.1.1 Назначение компонента

Элемент **Zbutton** 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:

9.1.2 Использование компонента.

При нажатии на правую кнопку мыши появляется список



9.1.3 Свойства, методы и события.

9.1.3.1 Свойства задания цветовой схемы.

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.

ClrText – цвет надписи на кнопке.

ClrBgr – цвет фона.

ClrPressed – цвет фона нажатой кнопки (если кнопка имеет фиксирующий стиль). Это свойство необходимо для создания эффекта подсветки кнопки.

9.1.3.2 Свойства для задания шрифта

FontFace – название шрифта, как строковая переменная,

Bold – включить/выключить утолщенный шрифт,

Italic – включить/выключить наклонный шрифт.

9.1.3.3 Свойства для задания поведения кнопки

PressStyle – свойство для стиля кнопки. 0 – без фиксации, 1 – с фиксацией.

Down – состояние нажатой (1) или отжатой (0) кнопки. Или команда нажать или отжать кнопку.

Caption – надпись на кнопке. Надпись на кнопке можно прочитать или задать.

9.1.3.4 Свойства и методы управления списком кнопки.

Reset(void) – очистить список,

*AddString(*LPCTSTR sValue*)* – добавить элемент *sValue* в список,

ItemIndex – индекс (порядковый номер) выбранного элемента из списка (нумерация идет с 0).

SetMinStringLength(SHORT Value) – установить минимальное количество символов в строке. Элементы списка, имеющие меньшую длину автоматически дополняются пробелами.

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.*

9.1.3.5 События от компонента.

Modify(BSTR sValue) – событие возникает при выборе элемента из списка.

sValue – строковая величина выбранного элемента.

Click(void) – событие возникает при нажатии на левую кнопку мыши над компонентом.

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.

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)

9.2 ZIndicator

9.2.1 Назначение компонента.

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.

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

Внешний вид компонента:

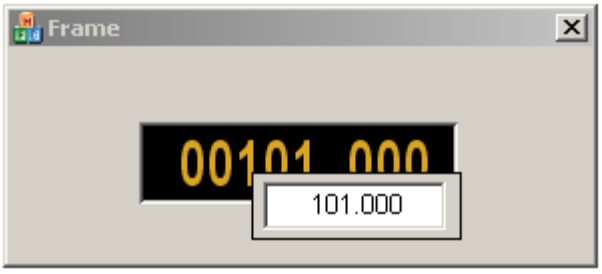
9.2.2 Использование компонента.

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.



9.2.3 Свойства, методы и события.

9.2.3.1 Свойства отображения рамки.

Используются для создания эффекта объемного элемента.

BevelUp – рисовать верхнюю границу рамки

BevelDown – рисовать нижнюю границу рамки

BevelLeft – рисовать левую границу рамки

BevelRight – рисовать правую границу рамки

9.2.3.2 Свойства задания цветовой схемы:

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.

ClrText – цвет текста

ClrBgr – цвет фона

9.2.3.3 Свойства шрифта:

FontFace – название шрифта, как строковая переменная,

Bold – включить/выключить утолщенный шрифт,

Italic – включить/выключить наклонный шрифт.

9.2.3.4 Свойства управления значением индикатора.

Max – максимальное устанавливаемое значение индикатора

Min – минимальное устанавливаемое значение индикатора

Value – текущее значение индикатора.

NumCnt – общее количество отображаемых десятичных знаков.

NumDivCnt – количество отображаемых десятичных знаков после запятой.

9.2.3.5 События.

Modify(FLOAT fValue) – возникает при изменении пользователем значения индикатора.

fValue – установленное значение.

9.3 ZRegulator

9.3.1 Назначение компонента.

Элемент представляет собой список
Общий вид компонента:

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

9.3.2 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.

9.3.3 Свойства, методы и события.

9.3.3.1 Свойства задания цветовой схемы.

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.

ClrText – цвет надписи на кнопке.

ClrBgr – цвет фона.

ClrPressed – цвет фона нажатой кнопки (если кнопка имеет фиксирующий стиль). Это свойство необходимо для создания эффекта подсветки кнопки.

9.3.3.2 Свойства для задания шрифта

FontFace – название шрифта, как строковая переменная,

Bold – включить/выключить утолщенный шрифт,

Italic – включить/выключить наклонный шрифт.

9.3.3.3 Свойства и методы управления списком кнопки.

Reset(void) – очистить список,

AddString(LPCTSTR sValue) – добавить элемент *sValue* в список,

ItemIndex – индекс (порядковый номер) выбранного элемента из списка (нумерация идет с 0).

SetMinStringLength(SHORT Value) – установить минимальное количество символов в строке. Элементы списка, имеющие меньшую длину автоматически дополняются пробелами.

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.*

9.3.3.4 События от компонента.

Modify(BSTR sValue) – событие возникает при выборе элемента из списка.
sValue – строковая величина выбранного элемента.

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

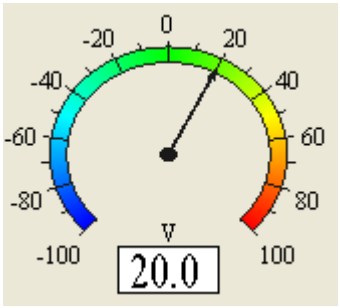
10 Описание компонентов отображения.

10.1 zet_circle_indicator, zet_arc_indicator

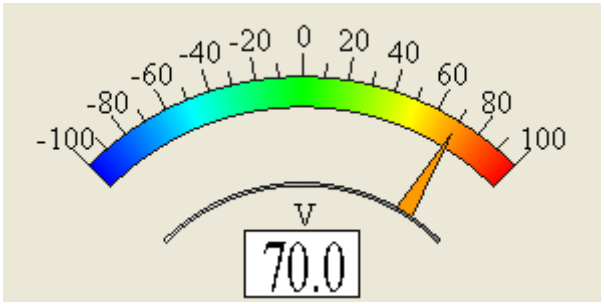
10.1.1 Назначение компонента.

Компоненты представляют собой стрелочные индикаторы и имеют несколько вариантов представления. В зависимости от назначения компонента, фон шкалы может иметь градиентную окраску, однотонную, или разбит на зоны. Значение, на которое указывает стрелка компонента, также выводится на табло. Для наибольшей информативности цвет табло меняется при критических значениях. Пользователь может самостоятельно подобрать удобную для него шкалу делений, а также шрифт и цвет фона.

Общий вид компонентов:



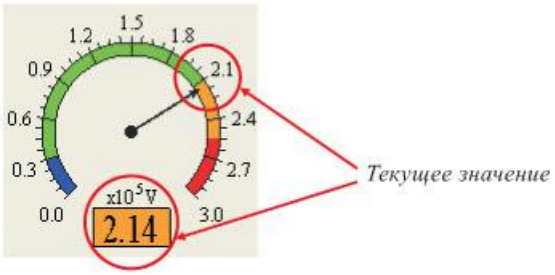
zet_circle_indicator



zet_arc_indicator

10.1.2 Использование компонента.

Когда компонент активен, стрелка индикатора указывает на текущее значение, которое также выводится на табло. Если шкала разбита на зоны, цвет табло становится оранжевым для значений из опасного диапазона и мигает красным при значениях из аварийной зоны. Цвет стрелки компонента **zet_arc_indicator** совпадает с цветом шкалы, соответствующим текущему значению.



zet_circle_indicator

10.1.3 Свойства, методы и события.

10.1.3.1 Режим работы.

Status – 0 - выкл. 1 - вкл.

10.1.3.2 Цифровая шкала.

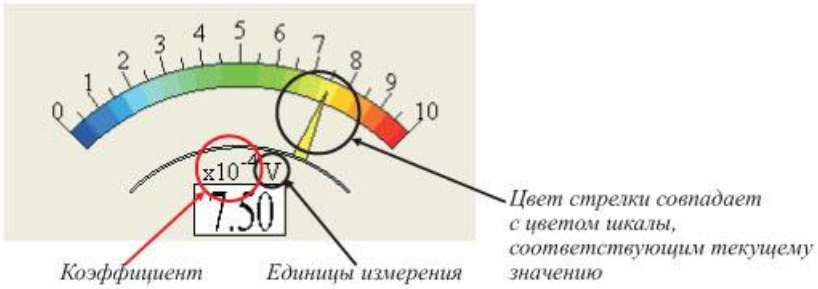
Все значения имею тип double. Точность шкалы – 2 знака после запятой. При значениях по модулю больше 1000 и меньше 0,1 вводится коэффициент, который отображается над табло.

Value – текущее значение индикатора.

MinValue – минимальное устанавливаемое значение индикатора.

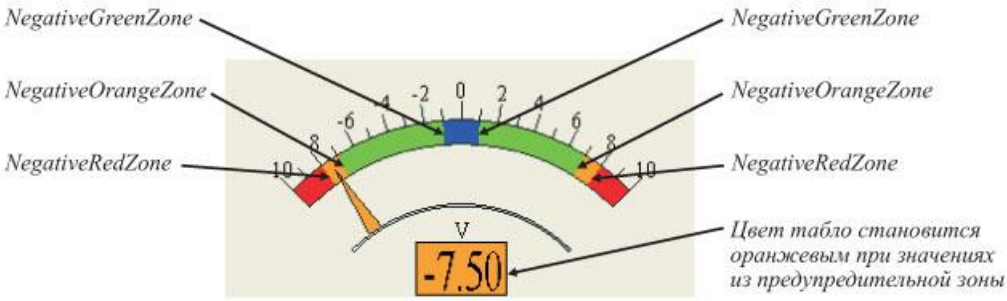
MaxValue – максимальное устанавливаемое значение индикатора.

Unit – единицы измерения.



zet_arc_indicator

Также при необходимости шкалу можно разбить на сектора. Предусмотрено всего 8 секторов:



zet_arc_indicator

2 нерабочие зоны:

По сути 1 зона, которая автоматически определяется началом правой рабочей зоны и началом левой рабочей зоны (или началом шкалы).

2 рабочие зоны:

GreenZone – начало правой зоны

NegativeGreenZone – начало левой зоны

2 предупредительные зоны (зоны опасности):

OrangeZone – начало правой зоны

NegativeOrangeZone – начало левой

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

2 аварийные зоны

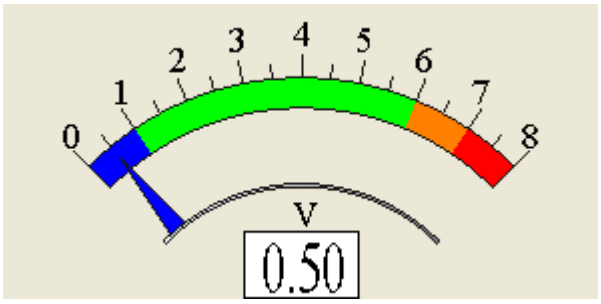
RedZone – начало правой зоны

NegativeRedZone – начало левой

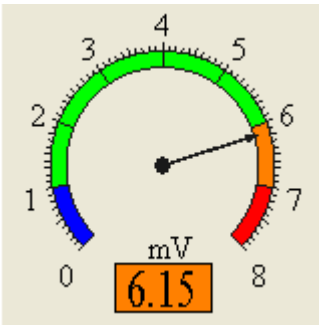
10.1.3.3 Цветовая шкала.

ColorScale – определяет раскраску шкалы: 0 – произвольная (плавный переход от синего к зеленому потом к красному), 1 – односторонняя шкала, 2 – двухсторонняя.

Односторонняя шкала:



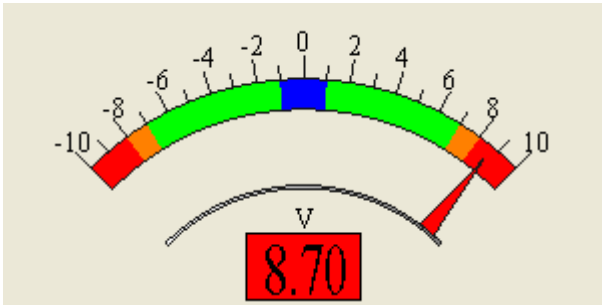
zet_arc_indicator



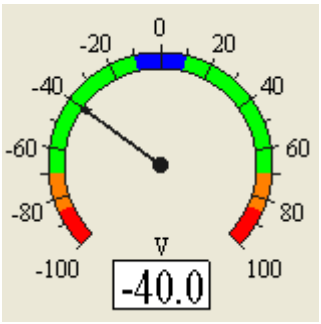
zet_circle_indicator

от левого края до *GreenZone* - синий цвет
от *GreenZone* до *OrangeZone* - зеленый цвет
от *OrangeZone* до *RedZone* - оранжевый цвет
от *RedZone* до правого края – красный.

Двухсторонняя шкала:



zet_arc_indicator



zet_circle_indicator

от левого края до *NegativeRedZone* - красный цвет
от *NegativeRedZone* до *NegativeOrangeZone* - оранжевый цвет
от *NegativeOrangeZone* до *NegativeGreenZone* - зеленый цвет
от *NegativeGreenZone* до центра - синий цвет
от ценра до *GreenZone* - синий цвет
от *GreenZone* до *OrangeZone* - зеленый цвет
от *OrangeZone* до *RedZone* - оранжевый цвет
от *RedZone* до правого края – красный

10.1.3.4 Шкала делений.

Mark – количество основных делений (от 2 до 20).

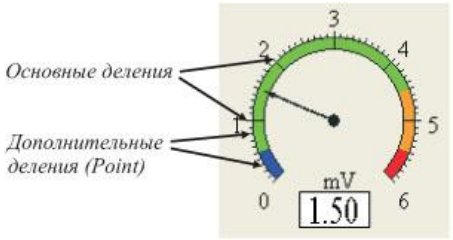
Line – количество промежуточных делений между основными (от 1 до 5).

Point – 1 – разделить каждое основное деление на 10 частей, 0 – без малых меток.

Цифры проставляются над основными делениями. У компонента **zet_arc2_indicator** цифры также можно проставить над дополнительными метками (*Numbers*=1)

zet_arc_indicator:
(произвольная шкала)

ColorScale=0;
Mark=5;
Line=2;
Point=0;
Numbers=1;



zet_circle_indicator:
(односторонняя шкала)

ColorScale=1;
Mark=7;
Line=1;
Point=1;

10.1.3.5 Цвет фона.

FonColor. 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.

10.1.3.6 Шрифт.

FontName – имя шрифта.

FontSize – размер шрифта.

Выбранным шрифтом отображаются значения над основными делениями цифровой шкалы, коэффициент и единицы измерения. Степень коэффициента выводится меньшим шрифтом. Дополнительные значения выводятся меньшим шрифтом и курсивом. Шрифт для значения на табло подбирается автоматически пропорционально размерам элемента.

10.1.3.7 Методы.

ShowPropSheet – метод выводит на экран страницу свойств, где пользователь может установить режим работы, вид шкалы, цвет фона, тип шрифта, максимальное и минимальное значения.

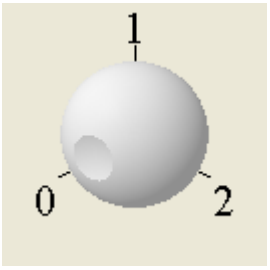
ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

10.2 zet_line_selector, zet_pit_selector

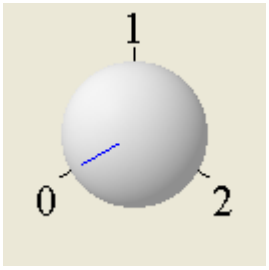
1.1.1 Назначение компонента.

Компоненты представляют собой аналоговые регуляторы и предназначены для изменения целых и дробных десятичных числовых величин. Необходимое значение устанавливается с помощью мыши. Компоненты различаются только внешним видом.

Общий вид компонентов:



zet_pit_selector



zet_line_selector

10.2.1 Использование компонента.

Когда компонент активен, необходимое значение можно установить, вращая колесико мыши. Другой способ - удерживая левую кнопку мыши нажатой, «ухватиться» за стрелочку индикатора и переместить ее до нужной отметки. Когда кнопка мыши будет отпущена, будет установлено выбранное значение. Когда установлен режим работы с фиксацией, стрелка индикатора перемещается только по основным значениям.

10.2.2 Свойства, методы и события.

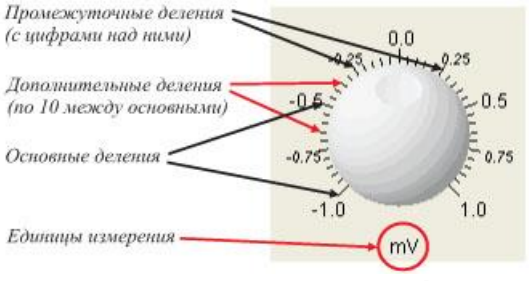
10.2.2.1 Режим работы.

Status – 0 - выкл, 1 - вкл.

Fix – 1 – с фиксацией, 0 – без фиксации.

10.2.2.2 Свойства шкалы.

В зависимости от назначения компонента, шкала может иметь различное количество делений или быть без цифр.



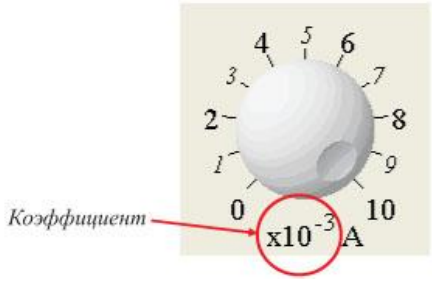
zet_pit_selector

Numbers – 0 – шкала без цифр, 1 – цифры только над основными делениями, 2 – цифры над основными и промежуточными делениями.

MinValue – минимальное значение шкалы.

MaxValue – максимальное значения шкалы.

Все значения имеет тип double. Максимальная точность – 0,01. При значениях *MinValue* и *MaxValue* больше ±1000 и меньше ±1 вводится коэффициент.



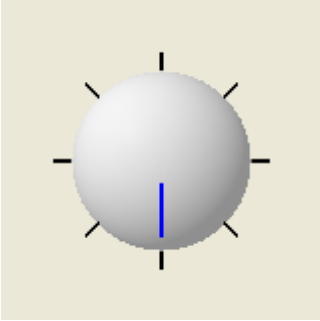
zet_pit_selector:
Numbers=2;
MinValue=0;
MaxValue=0,01;
Mark=6;
Line=1;
Point=0;

Mark – количество основный делений шкалы.

Line – количество промежуточных делений между основными (от 1 до 5)

Point – 1 – разделить каждый промежуток между основными делениями на 10 частей, 0 – без дополнительных делений.

Если выбрана шкала без цифр, компонент имеет 8 основных делений (*Mark*) по всей окружности, *Line* и *Point* равны 0.



zet_line_selector:
Numbers=0;
Mark=8;
Line=0;
Point=0.

zet_line_selector:
Numbers=1;
MinValue=0;
MaxValue=6;
Mark=7;
Line=3;
Point=0.

10.2.2.3 Свойства шрифта.

FontName – имя шрифта (по умолчанию «Times New Roman»).

FontSize – размер шрифта (высота)

Выбранным шрифтом отображаются значения над основными делениями цифровой шкалы, коэффициент и единицы измерения. Степень коэффициента выводится меньшим шрифтом. Дополнительные значения выводятся меньшим шрифтом и курсивом.

10.2.2.4 Цвет фона.

FonColor. 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.

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

10.2.2.5 Методы.

ShowPropSheet – метод выводит на экран страницу свойств, где пользователь может установить режим работы, вид шкалы, цвет фона, тип шрифта, максимальное и минимальное и текущее значения.

10.3 zet_knob

1.1.2 Назначение компонента.

Компонент представляет собой кнопку-переключатель.
Общий вид компонента:

выкл.

вкл.

10.3.1 Использование компонента.

Когда компонент активен, его состояние изменяется при нажатии левой кнопки мыши на «кнопку». При этом курсор меняет свой вид со «стрелки» на «руку».

10.3.2 Свойства, методы и события.

Status – состояние (вкл./выкл.)

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.

10.4 zet_boolean

1.1.3 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.

10.4.1 Свойства, методы и события.

Status – состояние (вкл./выкл.)

Color – цвет компонента.

FonColor – цвет фона.

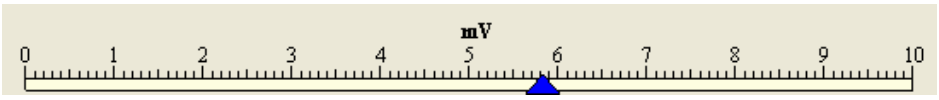
Цвет задается при помощи макроса *RGB(RED, GREEN, BLUE)*. Значения *RED, GREEN, BLUE* задают интенсивности цветов красного, зеленого и синего. Интенсивности изменяются в пределах от 0 до 255.

10.5 zet_horizontal_slider, zet_vertical_slider

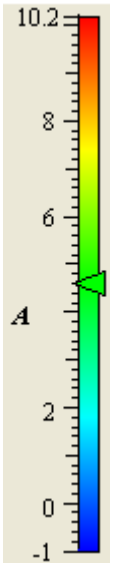
1.1.4 Назначение компонента.

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 единицы измерения задаются. Разметка шкалы выполняется автоматически. Цвета компонента могут быть заданы пользователем.

Общий вид компонентов:



zet_horizontal_slider



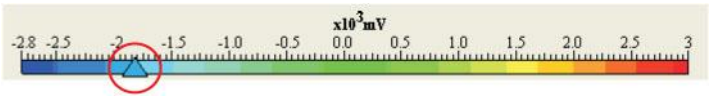
zet_vertical_slider

10.5.1 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.

При разноцветной окраске шкалы цвет стрелки компонентов совпадает с цветом шкалы, соответствующим текущему значению.

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru



zet_horizontal_slider

10.5.2 Свойства, методы и события.

10.5.2.1 Режим работы.

Status – 0 - выкл. (компонент пассивен), 1 - вкл. (компонент активен).

10.5.2.2 Цифровая шкала.

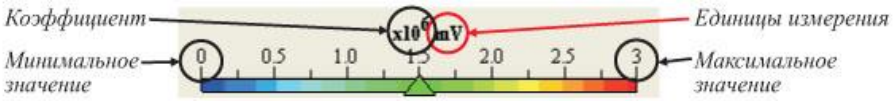
Все значения имею тип double. Точность шкалы – 2 знака после запятой. При значениях по модулю больше 1000 и меньше 0,1 вводится коэффициент, который отображается над табло.

Value – текущее значение индикатора.

MinValue – минимальное устанавливаемое значение индикатора.

MaxValue – максимальное устанавливаемое значение индикатора.

Unit – единицы измерения.



zet_horizontal_slider (ForeColor = 0)

10.5.2.3 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.

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.

10.5.2.4 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.

10.5.2.5 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.

10.5.2.6 Методы.

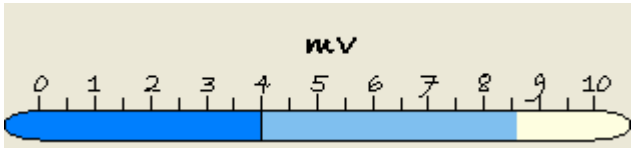
ShowPropSheet – метод выводит на экран страницу свойств, где пользователь может установить режим работы, вид шкалы, цвет фона, тип шрифта, максимальное и минимальное значения.

10.6 zet_horizontal_level, zet_vertical_level

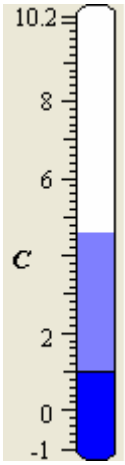
1.1.5 Назначение компонента.

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.

Общий вид компонентов:



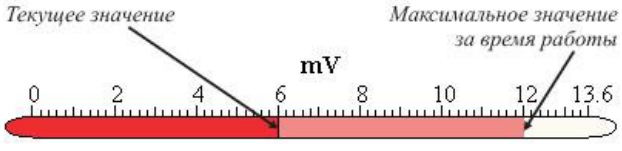
*zet_horizontal_level (*шрифт Bradley Hand*)*



zet_vertical_level

10.6.1 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

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

10.6.2 Свойства, методы и события.

10.6.2.1 Режим работы.

Status – 0 - выкл. (компонент пассивен), 1 - вкл. (компонент активен).

10.6.2.2 Цифровая шкала.

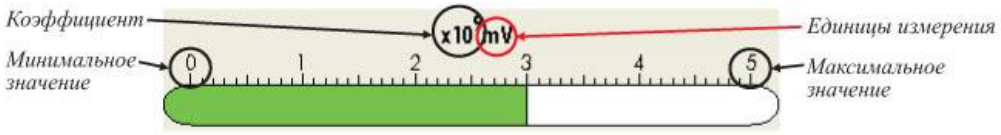
Все значения имею тип double. Точность шкалы – 2 знака после запятой. При значениях по модулю больше 1000 и меньше 0,1 вводится коэффициент, который отображается над табло.

Value – текущее значение индикатора.

MinValue – минимальное устанавливаемое значение индикатора.

MaxValue – максимальное устанавливаемое значение индикатора.

Unit – единицы измерения.



*zet_horizontal_level (*Track = 0*)*

10.6.2.3 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.

10.6.2.4 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.

10.6.2.5 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.

10.6.2.6 Методы.

ShowPropSheet – метод выводит на экран страницу свойств, где пользователь может установить режим работы, вид шкалы, цвет фона, тип шрифта, максимальное и минимальное значения.

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

11 Программный модуль Unit.ocx для связи с измерительными программами ZETLab.

11.1 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.

11.2 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 11.1)

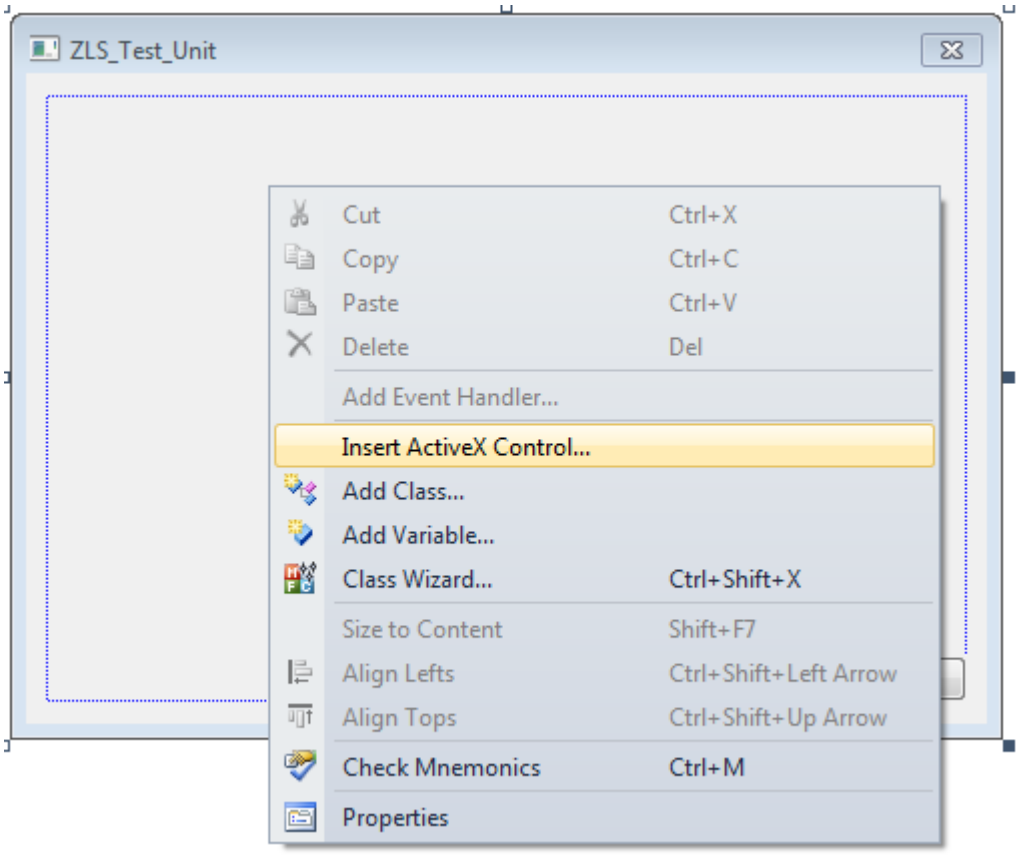


Рисунок 11.1

Из появившегося диалогового окна следует выбрать компонент *Unit Control* и нажать *OK* (рисунок 11.2).

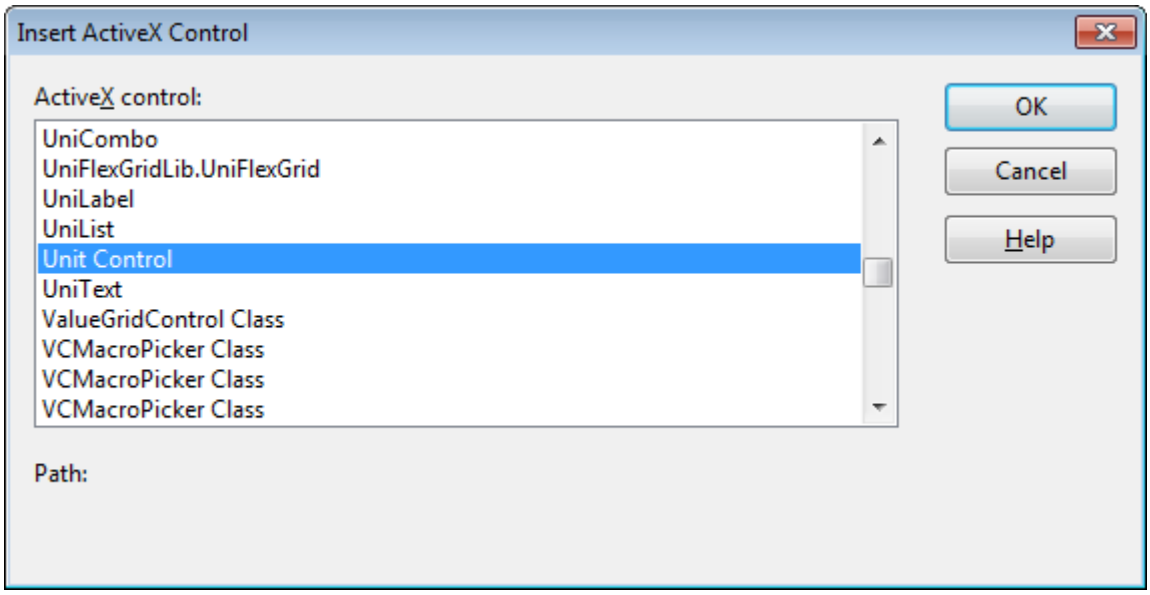


Рисунок 11.2

После этого компонент *Unit.ocx* появится на форме диалога (рисунок 11.3).

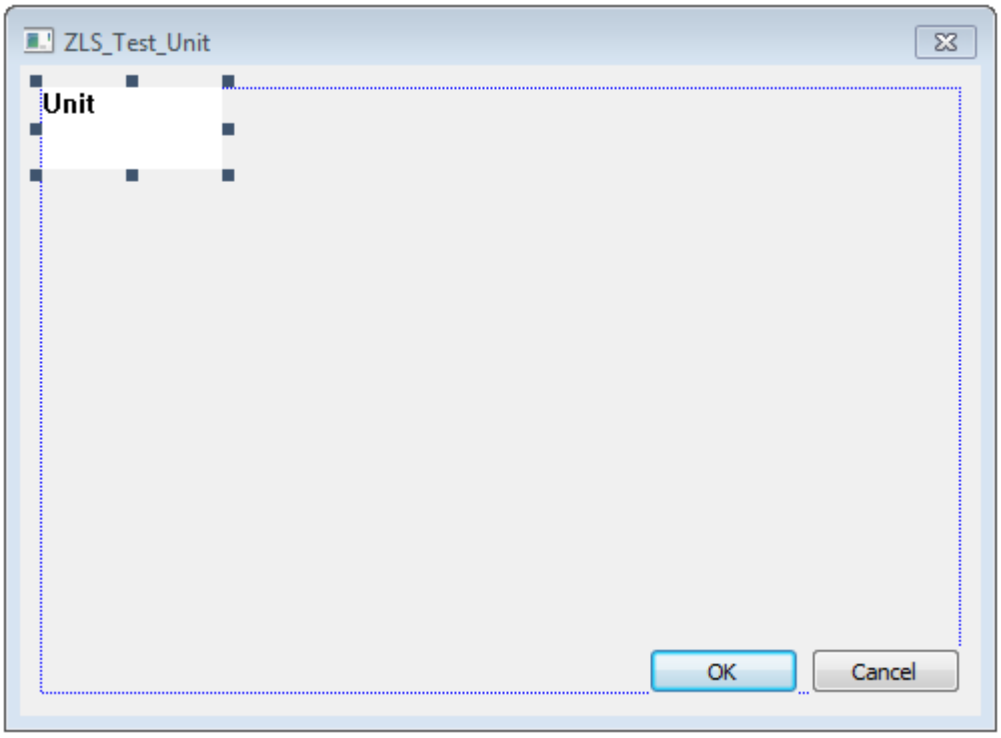


Рисунок 11.3

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*.

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

11.3 Описание методов и событий

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.

11.3.1 Серверная часть

11.3.1.1 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*.

-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, signed char* value)*

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* value, double* time)*

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*.
-6 – место в памяти для возврата значения устанавливаемого строкового параметра равно *NULL*.
-7 – превышена максимальная допустимая длина считываемого строкового параметра.

Write – передача данных в запущенную программу.

long ***Write****(long size, float* data, long param)*

Параметры
size – размер передаваемого массива данных.

data – указатель на передаваемый массив данных.

param – параметр.

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

Возвращаемое значение

0 – нормальное выполнение функции.

-1 – не было подключения к программе через компонент *Unit*.

-2 – приложение, которое запустило данную программу через компонент *Unit*, было выгружено из памяти.

-3 – не хватает хэндлеров памяти.

-4 – не хватает хэндлеров памяти.

-5 – размер передаваемого массива данных меньше нуля, либо равен нулю.

WriteNet – передача данных в запущенную программу.

long ***WriteNet****(long size, long pData, 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 – приложение, которое запустило данную программу через компонент *Unit*, было выгружено из памяти.

-3 – не хватает хэндлеров памяти.

-4 – не хватает хэндлеров памяти.

-5 – размер передаваемого массива данных меньше нуля, либо равен нулю.

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.

11.3.1.2 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.

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*** *()*

11.3.2 Клиентская часть

11.3.2.1 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.).

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

-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****(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.

ReadNet – чтение данных из подключенной программы (для языков программирования .NET).

long ***ReadNet****(long* size, long pData, long* param)*

Параметры

size – возвращает размер переданного массива данных.

pData – 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.

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* value)*

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, LPCTSTR value, double time)*

Параметры

param – номер устанавливаемого параметра.
value – значение устанавливаемого строкового параметра.
time – значение времени ZETServer.

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

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.

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.
-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.

IamActive– сообщение в UNIT о том, что программа активна и функционирует. Если UNIT в течение 5 сек не получит хотя-бы одного данного сообщения, то будет сгенерировано событие ***NotActive***. Если не планируется работа с этим событием, то вызывать данную функцию не обязательно.

void ***IamActive*** *(void)*

11.3.2.2 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.

11.3.3 Установка параметров программ

Для установки параметров программ используются методы *SetParam(…)*, *SetParamString(…)* и *SetParamTimeString(…)*, где аргумент *param* отвечает за установку того или иного

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

значения управляющего параметра. Ниже приведены значения аргумента *param* для настройки приложений при работе через компонент *Unit*.

11.3.3.1 Узкополосный спектральный анализ (spectr)

Параметр (*param*)	Аргумент (*value*)
0	Установка порядкового номера канала (от 0 до (количество каналов - 1))
1	Установка декады (частотного диапазона анализа) Гц: 0 – от 0 до (частота дискретизации / 2) 1 – от 0 до (частота дискретизации / 20) 2 – от 0 до (частота дискретизации / 200) 3 – от 0 до (частота дискретизации / 2000) 4 – от 0 до (частота дискретизации / 20000)
3	Установка времени усреднения (от 0.01 секунды до 100 секунд)
4	Установка типа представления расчета спектра: 0 – спектральная плотность 1 – спектральная плотность мощности 2 – среднеквадратичное значение 3 – амплитудное (пиковое) значение
5	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(...)*
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 частотного диапазона путем выставления требуемого количества полос, умноженного на 2 (от 100 до 250000)
9	Установка очистки спектра медианным фильтром: 0 – выключение медианного фильтра 1 – включение медианного фильтра
10	Установка типа анализа: 0 – быстрое преобразование Фурье (БПФ) 1 – дискретное преобразование Фурье (ДПФ)
11	Установка типа весовой функции: 0 – прямоугольная 1 – весовая функция Ханна 2 – весовая функция Хэмминга 3 – весовая функция Блэкмана 4 – весовая функция Барлета 5 – весовая функция Блэкмана (стд)
12	Установка запрета на получение данных (аргумент может принимать любое значение)
13	Установка типа обработки сигнала:

Параметр (*param*)	Аргумент (*value*)
	0 – дифференцирование второго порядка 1 – дифференцирование первого порядка 2 – без обработки интегрирования и дифференцирования 3 – интегрирование первого порядка 4 – интегрирование второго порядка
14	Установка типа представления уровня спектральных компонент: 0 – линейный масштаб (в единицах измерения) 1 – логарифмический масштаб (в децибелах)
15	Установка на включение, чтение и прорисовку нормы (аргумент может принимать любое значение)
16	Установка на отключение и прорисовку нормы (аргумент может принимать любое значение)
18	Установка положения курсора на графике (от 0 до (количество полос - 1))
19	Установка на последовательное получение данных по операции *Read(…)* (аргумент может принимать любое значение). После задания этого параметра необходимо дождаться события готовности *Ready(…)* и затем прочитать массив значений частотного ряда, значений текущего, максимального и среднего спектров операцией *Read(…)*.
20	Установка на расчет максимального спектра: 0 – запрет расчета максимального спектра 1 – разрешение расчета максимального спектра
21	Установка на расчет среднего спектра: 0 – запрет расчета среднего спектра 1 – разрешение расчета среднего спектра
22	Установка интервала расчета дополнительных спектров (от 10 секунд до 100000 секунд)
23	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
24	Нижняя граница отображения данных по оси Y в единицах измерения
25	Верхняя граница отображения данных по оси Y в единицах измерения
255	Установка на включение узкополосного спектрального анализа (аргумент может принимать любое значение)
127	Установка на выключение узкополосного спектрального анализа (аргумент может принимать любое значение)

11.3.3.2 Долеоктавный спектральный анализ (dspectr)

Параметр (*param*)	Аргумент (*value*)
0	Установка порядкового номера канала (от 0 до (количество каналов - 1))
1	Установка времени усреднения (от 0.1 секунды до 100 секунд)
2	Установка интервала расчета дополнительных спектров (от 10 секунд до 100000 секунд)
3	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 (ча-

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

Параметр (*param*)	Аргумент (*value*)
	стот и спектра) операцией Read(…)
4	Установка на получение массива значений частотного ряда, текущего, максимального, минимального и среднего спектров 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, максимального, минимального и среднего спектров операцией Read(…)
5	Установка запрета на получение данных (аргумент может принимать любое значение)
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 типа представления уровня спектральных компонент: 0 – линейный масштаб (в единицах измерения) 1 – логарифмический масштаб (в децибелах)
8	Установка типа анализа: 0 – октавный 1 – 1/3 октавный 2 – 1/12 октавный 3 – 1/24 октавный
9	Установка типа представления расчета спектра: 0 – среднеквадратичное значение 1 – пиковое значение
10	Установка на расчет максимального спектра: 0 – запрет расчета максимального спектра 1 – разрешение расчета максимального спектра
11	Установка на расчет минимального спектра: 0 – запрет расчета минимального спектра 1 – разрешение расчета минимального спектра
12	Установка на расчет среднего спектра: 0 – запрет расчета среднего спектра 1 – разрешение расчета среднего спектра
13	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
14	Установка типа обработки сигнала: 0 – дифференцирование второго порядка 1 – дифференцирование первого порядка 2 – без обработки интегрирования и дифференцирования 3 – интегрирование первого порядка 4 – интегрирование второго порядка
255	Установка на включение долеоктавного спектрального анализа (аргумент может принимать любое значение)
127	Установка на выключение долеоктавного спектрального анализа (аргумент может принимать любое значение)

11.3.3.3 Взаимный узкополосный спектральный анализ (vspectr)

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 декады (частотного диапазона анализа) Гц: 0 – от 0 до (частота дискретизации / 2) 1 – от 0 до (частота дискретизации / 20) 2 – от 0 до (частота дискретизации / 200) 3 – от 0 до (частота дискретизации / 2000) 4 – от 0 до (частота дискретизации / 20000)
3	Установка времени усреднения (от 0.1 секунды до 1000 секунд)
4	Установка типа анализа: 0 – быстрое преобразование Фурье (БПФ) 1 – дискретное преобразование Фурье (ДПФ)
5	Установка типа весовой функции: 0 – прямоугольная 1 – весовая функция Ханна 2 – весовая функция Хэмминга 3 – весовая функция Блэкмана 4 – весовая функция Барлета 5 – весовая функция Блэкмана (стд)
6	Установка типа обработки сигнала: 0 – дифференцирование второго порядка 1 – дифференцирование первого порядка 2 – без обработки интегрирования и дифференцирования 3 – интегрирование первого порядка 4 – интегрирование второго порядка
7	Установка типа усреднения: 0 – линейное 1 – экспоненциальное
8	Установка частотного диапазона путем выставления требуемого количества полос, умноженного на 2 (от 100 до 25000)
9	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(...)*
10	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(...)*
11	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(...)*
12	Установка на последовательное получение данных по операции *Read(...)* (аргумент может принимать любое значение). После задания этого параметра необходимо дождаться события готовности *Ready(...)* и затем прочитать массив значений частотного ряда, значений текущего спектра, значений действительной и мнимой части спектра, значений фазы, коэффициента когерентности и переход-

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

Параметр (*param*)	Аргумент (*value*)
	ной характеристики операцией *Read(...)*
13	Установка запрета на получение данных (аргумент может принимать любое значение)
14	Установка на расчет действительной части дополнительного окна: 0 – запрет расчета действительной части спектра 1 – разрешение расчета действительной части спектра
15	Установка на расчет мнимой части дополнительного окна: 0 – запрет расчета мнимой части спектра 1 – разрешение расчета мнимой части спектра
16	Установка на расчет фазы дополнительного окна: 0 – запрет расчета фазы 1 – разрешение расчета фазы
17	Установка на расчет коэффициента когерентности дополнительного окна: 0 – запрет расчета коэффициента когерентности 1 – разрешение расчета коэффициента когерентности
18	Установка на расчет переходной характеристики дополнительного окна: 0 – запрет расчета переходной характеристики 1 – разрешение расчета переходной характеристики
19	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 узкополосного спектрального анализа (аргумент может принимать любое значение)
127	Установка на выключение взаимного узкополосного спектрального анализа (аргумент может принимать любое значение)

11.3.3.4 Взаимный долеоктавный спектральный анализ (dvspectr)

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

Параметр (*param*)	Аргумент (*value*)
	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 *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: 0 – 1/1 октавный 1 – 1/3 октавный 2 – 1/12 октавный 3 – 1/24 октавный
12	Установка типа представления уровня спектральных компонент: 0 – линейный масштаб (в единицах измерения) 1 – логарифмический масштаб (в децибелах)
13	Установка на расчет действительной части: 0 – запрет расчета действительной части спектра 1 – разрешение расчета действительной части спектра
14	Установка на расчет мнимой части: 0 – запрет расчета мнимой части спектра 1 – разрешение расчета мнимой части спектра
15	Установка на расчет фазы: 0 – запрет расчета фазы 1 – разрешение расчета фазы
16	Установка на расчет коэффициента когерентности: 0 – запрет расчета коэффициента когерентности 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)

11.3.3.5 Взаимный корреляционный анализ (corr)

Параметр (*param*)	Аргумент (*value*)

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

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)
3	Установка декады (частотного диапазона анализа) Гц: 0 – от 0 до (частота дискретизации / 2) 1 – от 0 до (частота дискретизации / 20) 2 – от 0 до (частота дискретизации / 200) 3 – от 0 до (частота дискретизации / 2000) 4 – от 0 до (частота дискретизации / 20000)
4	Установка полосового фильтра: 0 – выключение полосового фильтра 1 – включение полосового фильтра
5	Установка частоты среза фильтра высоких частот (от 0 Гц до частоты среза фильтра низких частот)
6	Установка частоты среза фильтра низких частот (от частоты среза фильтра высоких частот до (частота дискретизации / 2)) Гц
7	Установка на получение размера массива корреляционной функции по операции *Read(...)* (аргумент может принимать любое значение). После задания этого параметра необходимо дождаться события готовности *Ready(...)* и затем прочитать размер массива (корреляционной функции и времени) операцией *Read(...)*
8	Установка на получение массива значений временного ряда по операции *Read(...)* (аргумент может принимать любое значение). После задания этого параметра необходимо дождаться события готовности *Ready(...)* и затем прочитать массив значений временного ряда операцией *Read(...)*
9	Установка на получение массива значений корреляционной функции по операции *Read(...)* (аргумент может принимать любое значение). После задания этого параметра необходимо дождаться события готовности *Ready(...)* и затем прочитать массив значений корреляционной функции операцией *Read(...)*
10	Установка на расчет огибающей корреляционной функции: 0 – запрет расчета огибающей корреляционной функции 1 – разрешение расчета огибающей корреляционной функции Установка на получение массива значений огибающей корреляционной функции по операции *Read(...)* (аргумент может принимать любое значение). После задания этого параметра необходимо дождаться события готовности *Ready(...)* и затем прочитать массив значений огибающей корреляционной функции операцией *Read(...)*
11	Установка размера корреляционной функции путем выставления требуемого количества значений функции, умноженного на 2 (может принимать значения, получаемые возведением 2 в степень от 8 до 19)
12	Установка на компенсацию основного корреляционного пика: 0 – запрет компенсации основного корреляционного пика 1 – разрешение компенсации основного корреляционно пика
13	Установка на фильтрацию дискретных помех (помех в виде тональных стационарных сигналов): 0 – запрет фильтрации дискретных помех 1 – разрешение фильтрации дискретных помех
14	Установка инверсии одного из сигналов: 0 – не инвертировать сигнал

Параметр (*param*)	Аргумент (*value*)
	1 – инвертировать один из сигналов
16	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 корреляционного анализа (аргумент может принимать любое значение)
127	Установка на выключение взаимного корреляционного анализа (аргумент может принимать любое значение)

11.3.3.6 Анализ нелинейных искажений (harmdist)

Параметр (*param*)	Аргумент (*value*)
0	Установка порядкового номера канала (от 0 до (количество каналов - 1))
1	Установка типа представления расчета коэффициента нелинейных искажений: 0 – линейный масштаб (в процентах) 1 – логарифмический масштаб (в децибелах)
2	Установка декады (частотного диапазона анализа) Гц: 0 – от 0 до (частота дискретизации / 2) 1 – от 0 до (частота дискретизации / 20) 2 – от 0 до (частота дискретизации / 200) 3 – от 0 до (частота дискретизации / 2000) 4 – от 0 до (частота дискретизации / 20000)
3	Установка времени усреднения (от 0.1 секунды до 10 секунд)
4	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 анализа нелинейных искажений (аргумент может принимать любое значение)
255	Установка на включение анализа нелинейных искажений (аргумент может принимать любое значение)

11.3.3.7 Синхронное накопление (PrdkAnaliz)

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: 0 – ниспадающий фронт 1 – восходящий фронт
3	Установка типа обработки сигнала: 0 – дифференцирование второго порядка 1 – дифференцирование первого порядка 2 – без обработки интегрирования и дифференцирования

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

Параметр (*param*)	Аргумент (*value*)
	3 – интегрирование первого порядка 4 – интегрирование второго порядка
4	Установка типа представления уровня спектральных компонент гармоник, отображение по Y: 0 – линейный масштаб (в единицах измерения) 1 – логарифмический масштаб (в децибелах)
5	Установка времени усреднения (от 0.1 секунды до 10 секунд)
6	Установка на включение спектра гармоник: 0 – выключение спектра гармоник 1 – включение спектра гармоник
7	Установка на включение спектрограммы гармоник: 0 – выключение спектрограммы гармоник 1 – включение спектрограммы гармоник
8	Установка на включение трехмерной спектрограммы гармоник: 0 – выключение трехмерной спектрограммы гармоник 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: 0 – выключение отображения в полярных координатах 1 – включение отображения в полярных координатах
11	Установка на отображение шестеренки на графике в полярных координатах: 0 – не отображать шестеренку на графике в полярных координатах 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): 0 – от 0 до (частота дискретизации / 1) 1 – от 0 до (частота дискретизации / 10) 2 – от 0 до (частота дискретизации / 100) 3 – от 0 до (частота дискретизации / 1000) 4 – от 0 до (частота дискретизации / 10000)
14	Установка вида установки передаточного числа: 0 – в виде рациональной дроби (через параметры 16-21) 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: 0 – передача формы сигнала 1 – передача спектра гармоник 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)

Параметр (*param*)	Аргумент (*value*)
255	Установка на включение синхронного накопления (аргумент может принимать любое значение)

11.3.3.8 Модальный анализ (PrqsAnaliz)

Параметр (*param*)	Аргумент (*value*)
0	Установка порядкового номера опорного канала (от 0 до (количество каналов -1))
1	Установка порядкового номера измерительного канала (от 0 до (количество каналов - 1))
2	Установка декады (частотного диапазона анализа) Гц: 0 – от 0 до (частота дискретизации / 1) 1 – от 0 до (частота дискретизации / 10) 2 – от 0 до (частота дискретизации / 100) 3 – от 0 до (частота дискретизации / 1000) 4 – от 0 до (частота дискретизации / 10000)
3	Установка интервала расчета (в секундах)
4	Установка инверсии опорного канала: 0 – не инвертировать опорный канал 1 – инвертировать опорный канал
5	Установка инверсии измерительного канала: 0 – не инвертировать измерительный канал 1 – инвертировать измерительный канал
6	Установка нормировки измерительного канала: 0 – не нормировать измерительный канал 1 – нормировать измерительный канал
7	Установка типа порога по СКЗ шумов: 0 – адаптивный (СКЗ * К) 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: 0 – любой фронт 1 – только положительный фронт
12	Установка интервала расчета добротностей (от 0.1% до 100%)
13	Установка автозапуска анализа: 0 – запрет автозапуска анализа 1 – разрешение автозапуска анализа
14	Установка интервала времени, через которое происходит автозапуск (от 0.1 секунды до 1000 секунд)
15	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 модального анализа (аргумент может принимать любое значение)
255	Установка на включение модального анализа (аргумент может принимать любое

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

Параметр (*param*)	Аргумент (*value*)
	значение)

11.3.3.9 Гистограмма (ZETHistograph)

Параметр (*param*)	Аргумент (*value*)
0	Установка типа расчета значений: 0 – по напряжению 1 – по разрядам
1	Установка ширины столбца гистограммы, ед. изм.: – при установке типа расчета значений по напряжению задается от веса младшего разряда АЦП до максимально допустимого входного уровня, деленного на 16 – 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: 0 – гистограмма 1 – плотность вероятности 2 – вероятность
4	Установка времени накопления данных (от времени усреднения данных до 500 секунд)
5	Установка времени усреднения: 0 - 0.1 секунды 1 - 1 секунда 2 - 10 секунд
6	Установка на расчет нормального распределения (Гаусса) 0 – запрет расчета нормального распределения 1 – разрешение расчета нормального распределения
7	Установка на расчет гармонического распределения 0 – запрет расчета гармонического распределения 1 – разрешение расчета гармонического распределения
8	Установка на расчет распределения Хи-квадрат 0 – запрет расчета распределения Хи-квадрат 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)

11.3.3.10 Детектор STA\LTA (STA_LTA)

Параметр (*param*)	Аргумент (*value*)
0	Установка размерности исходного сигнала: 1 – скалярный 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
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: 0 – запрет создания виртуального канала STA_LTA 1 – разрешение создания виртуального канала STA_LTA
12	Установка на запись сигналов при обнаружении события в файл результатов *.dtu: 0 – запрет записи сигналов 1 – разрешение записи сигналов
13	Установка на отображение информации в окне программы: 0 – запрет отображения информации в окне программы 1 – разрешение отображения информации в окне программы
14	Установка записи сообщений в log-файл: 0 – запрет записи 1 – разрешение записи
15	зарезервировано
16	зарезервировано
17	Установка на применение устанавливаемых настроек программы (аргумент может принимать любое значение). После задания этого параметра произойдет применение настроек, выставленных другими параметрами.

11.3.3.11 Вольтметр переменного тока (VoltMeter)

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

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

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))/ В многоканальном режиме работы установка номера канала сервера для текущего номера канала вольтметра
1	Установка времени усреднения: 0 – быстро (0.1 секунды) 1 – медленно (1 секунда) 2 – очень медленно (10 секунд) В многоканальном режиме работы установка времени усреднения для текущего номера канала вольтметра
2	Установка типа значения, передаваемого в UNIT: 0 – среднеквадратичное значение 1 – пиковое значение 2 – амплитудное значение В многоканальном режиме работы установка типа передаваемого значения для текущего канала
3	Установка типа представления расчета: 0 – линейный масштаб (в единицах измерения) 1 – логарифмический масштаб (в децибелах) В многоканальном режиме работы установка типа тип 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 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

11.3.3.12 Вольтметр постоянного тока (VoltMeterDC)

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:

4. посылка величины "5" с параметром "-2" (5-й канал вольтметра назначается текущим);
5. посылка величины "7" с параметром "0" (текущему каналу вольтметра задаётся 7-ой канала сервера);
6. 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)). В многоканальном режиме работы установка номера канала сервера для текущего номера канала вольтметра
1	Установка времени усреднения: 0 – быстро (0.1 секунды); 1 – медленно (1 секунда); 2 – очень медленно (10 секунд). При посылки аргумента менее 0, считается, что *value* = 0. При посылки аргумента более 2, считается, что *value* = 2. 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

11.3.3.13 Селективный вольтметр (VoltMeterSel)

Параметр (*param*)	Аргумент (*value*)
0	Установка порядкового номера канала (от 0 до (количество каналов- 1))
1	Установка времени усреднения: 0 – быстро (0.1 секунды)

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

Параметр (*param*)	Аргумент (*value*)
	1 – медленно (1 секунда) 2 – очень медленно (10 секунд)
2	Установка варианта выбора частоты и полосы: 0 – ручной режим 1 – автоматический режим
3	Установка центральной частоты селективного фильтра, Гц
4	Установка полосы селективного фильтра, Гц

11.3.3.14 Частотомер (FreqMeter)

Параметр (*param*)	Аргумент (*value*)
0	Установка порядкового номера канала (от 0 до (количество каналов - 1))
1	Установка времени усреднения: 0 – быстро (0.1 секунды) 1 – медленно (1 секунда) 2 – очень медленно (10 секунд)

11.3.3.15 Фазометр (PhaseMeter)

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: 0 – быстро (0.1 секунды) 1 – медленно (1 секунда)
3	Установка варианта расчета фазы: 0 – в градусах 1 – в радианах

11.3.3.16 Тахометр (TahoMeter)

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: 0 – ручной режим 1 – автоматический режим

Параметр (*param*)	Аргумент (*value*)
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)

11.3.3.17 Торсиограф (Torsiograph)

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: 0 – выключение канала фазы Б 1 – включение канала фазы Б
4	Установка работы канала реперной метки 0: 0 – выключение канала реперной метки 0 1 – включение канала реперной метки 0
5	Установка единицы измерения: 0 – миллиметр 1 – сантиметр 2 – метр 3 – градус 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: 0 – запрет создания виртуального канала перемещения 1 – разрешение создания виртуального канала перемещения
9	Установка на создание виртуального канала скорости: 0 – запрет создания виртуального канала скорости 1 – разрешение создания виртуального канала скорости
10	Установка на инвертирование направления перемещения (работает при установке на включение канала фазы Б параметром 3): 0 – не инвертировать направление перемещения 1 – инверсия направления перемещения
11	Установка варианта выбора порога: 0 – ручной режим 1 – автоматический режим
12	Установка верхнего порога, ед. измер. канала
13	Установка нижнего порога, ед. измер. канала

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

11.3.3.18 Энкодер (Encoder)

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: 0 – выключение канала фазы Б 1 – включение канала фазы Б
4	Установка работы канала реперной метки 0: 0 – выключение канала реперной метки 0 1 – включение канала реперной метки 0
5	Установка единицы измерения: 0 – миллиметр 1 – сантиметр 2 – метр 3 – градус 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: 0 – запрет создания виртуального канала перемещения 1 – разрешение создания виртуального канала перемещения
9	Установка на создание виртуального канала скорости: 0 – запрет создания виртуального канала скорости 1 – разрешение создания виртуального канала скорости
10	Установка на инвертирование направления перемещения (работает при установке на включение канала фазы Б параметром 3): 0 – не инвертировать направление перемещения 1 – инверсия направления перемещения
11	Установка варианта выбора порога: 0 – ручной режим 1 – автоматический режим
12	Установка верхнего порога, ед. измер. канала
13	Установка нижнего порога, ед. измер. канала

11.3.3.19 Термометр ТС (ThermoMeter)

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 термометра сопротивления: 0 – платина 1.391 1 – платина 1.385 2 – медь 1.428

Параметр (*param*)	Аргумент (*value*)
	3 – медь 1.426 4 – никель 1.617
3	Установка поправки к показаниям в градусах (от минус 100 до 100)

11.3.3.20 Термометр ТП (ThermoPara)

Параметр (*param*)	Аргумент (*value*)
0	Установка порядкового номера измерительного канала (от 0 до (количество каналов - 1))
1	Установка порядкового номера канала компенсатора холодного спая (от 0 до (количество каналов - 1))
2	Установка типа термопары: 0 – R (ТПП13 – 13% родий/платина) 1 – S (ТПП10 – 10% родий/платина) 2 – B (ТПР – платина/родий) 3 – J (ТЖК – железо/константан) 4 – T (ТМКн – медь/константан) 5 – E (ТХКн – хромель/константан) 6 – K (ТХА – хромель/алюмель) 7 – N (ТНН – нихросил/нисил) 8 – A (ТВР – вольфрам/рений) 9 – L (ТХК – хромель/копель)
3	Установка на включение поправки по температуре холодного спая: 0 – выключение поправки по температуре холодного спая 1 – включение поправки по температуре холодного спая

11.3.3.21 Тензометр (TenzoMeter)

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: 0 – запрет использования файла калибровки 1 – разрешение использования файла калибровки
3	Установка режима измерений: 0 –тензорезистор 1 – тензодатчик
4	Установка чувствительности тензодатчика, мВ/В
5	Установка предела измерений по тензодатчику
6	Установка имени файла калибровки тензорезистора
7	Установка единицы измерений
8	Установка типа измерений: 0 – абсолютные измерения 1 – относительные измерения
9	Установка типа питания датчика:

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

Параметр (*param*)	Аргумент (*value*)
	0 – постоянный ток 1 – переменный ток
10	Установка на инвертирование измеренного значения нагрузки: 0 – не инвертировать измеренное значение нагрузки 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 сглаживания, мс
14	Установка на сброс текущего значения (аргумент может принимать любое значение)

11.3.3.22 Виброметр (VibroMeter)

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);
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)) В многоканальном режиме установка номера канала сервера для текущего номера канала виброметра
1	Установка времени усреднения: 0 – быстро (0.1 секунды) 1 – медленно (1 секунда) 2 – очень медленно (10 секунд) В многоканальном режиме установка времени усреднения для текущего номера канала виброметра
2	Установка типа рассчитываемого значения: 0 – пиковое значение

Параметр (*param*)	Аргумент (*value*)
	1 – среднеквадратичное значение 2 – амплитудное значение В многоканальном режиме установка типа для текущего номера канала виброметра
3	Установка порога защиты по ускорению В многоканальном режиме не поддерживается
4	Установка порога защиты по скорости В многоканальном режиме не поддерживается
5	Установка порога защиты по перемещению В многоканальном режиме не поддерживается
6	Установка на включение контроля по превышению порога: 0 – выключение контроля по превышению порога 1 – включение контроля по превышению порога В многоканальном режиме не поддерживается
7	Установка типа полосового фильтра (для частоты дискретизации используемого АЦП 25 кГц): 0 – 1 Гц - 200 Гц 1 – 10 Гц - 1000 Гц 2 – 3 Гц - 10000 Гц 3 – 1 Гц - 10 Гц В многоканальном режиме установка типа для текущего номера канала вольтметра
8	Установка использования выхода сухого контакта: 0 – не использовать выход сухого контакта 1 – использования выхода сухого контакта В многоканальном режиме не поддерживается
9	Установка длительности удержания события на сухом контакте (от 1 секунды до 100 секунд) В многоканальном режиме не поддерживается
10	Установка выдачи данных по сухому контакту в виртуальный канал: 0 – не выдавать данные по сухому контакту в виртуальный канал 1 – выдача данных по сухому контакту в виртуальный канал В многоканальном режиме не поддерживается
11	Установка выдачи данных по сухому контакту через цифровой выход: 0 – не выдавать данные по сухому контакту через цифровой выход 1 – выдача данных по сухому контакту через цифровой выход В многоканальном режиме не поддерживается
12	Установка номера бита цифрового порта дл я выдачи данных по сухому контакту (от 0 до 7) 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

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

Параметр (*param*)	Аргумент (*value*)
	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 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

11.3.3.23 Мультиметр Agilent_HP34401A (Agilent_HP34401A)

Параметр (*param*)	Аргумент (*value*)
0	Установка режима измерений: 0 – вольтметр постоянного тока 1 – вольтметр переменного тока 2 – амперметр постоянного тока 3 – амперметр переменного тока 4 – частотомер 5 – омметр
1	Установка времени усреднения: 0 – медленно (1 секунда) 1 – очень медленно (10 секунд)
2	Установка типа представления расчета: 0 – линейный масштаб (в единицах измерения) 1 – логарифмический масштаб (в децибелах)

11.3.3.24 Блок питания LPS305 (LPS305)

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: 0 – независимая работа каналов 1 – слежение за первым каналом
5	Установка включения звукового сигнала: 0 – выключение звукового сигнала 1 – включение звукового сигнала
6	Установка состояния питания цифрового выхода: 0 – цифровой выход отключен 1 – питание 3.3 В 2 – питание 5 В
8	Установка состояния выходов:

Параметр (*param*)	Аргумент (*value*)
	0 – выключение выходов 1 – включение выходов

11.3.3.25 Блок питания PPE3323 (PPE3323)

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: 0 – независимая работа каналов 1 – слежение за первым каналом
6	Установка состояния питания цифрового выхода: 1 – питание 3.3 В 2 – питание 5 В
7	Установка защиты выхода от перегрузки (OCP): 0 – защита выхода от перегрузки выключена 1 – защита выхода от перегрузки включена
8	Установка состояния выходов: 0 – выключение выходов 1 – включение выходов
9	Установка последовательного соединения выходов: 0 – последовательное соединение выходов выключено 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

11.3.3.26 Блок питания PSM2010 (PSM2010)

Параметр (*param*)	Аргумент (*value*)
0	Установка ограничения по напряжению
1	Установка ограничения по току
2	Установка состояния выходов: 0 – выключение выходов 1 – включение выходов
3	Установка защиты выхода от перегрузки (OCP): 0 – защита выхода от перегрузки выключена

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

Параметр (*param*)	Аргумент (*value*)
	1 – защита выхода от перегрузки включена
4	Установка защиты выхода от перенапряжения (OVP): 0 – защита выхода от перенапряжения выключена 1 – защита выхода от перенапряжения включена
5	Установка рабочего диапазона: 0 – 8V, 20A 1 – 20V, 10A
6	Установка уровня защиты по напряжению
7	Установка уровня защиты от перегрузки
8	Установка воспроизведения профиля: 0 – выключение воспроизведения профиля 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: 0 – выключение периодического опроса статуса прибора 1 – включение периодического опроса статуса прибора

11.3.3.27 Многоканальный осциллограф (OscGraph)

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: 0 – быстро (0.1 секунды) 1 – медленно (1 секунда)
3	Установка количества отображаемых осциллограм (от 1 до 8, либо от 1 до количества каналов, если количество каналов меньше 8): 0 – 1 канал 1 – 2 канала 2 – 3 канала 3 – 4 канала 4 – 5 каналов 5 – 6 каналов 6 – 7 каналов 7 – 8 каналов
4	Установка декады (частотного диапазона): 0 – от 0 до (частота дискретизации / 1) 1 – от 0 до (частота дискретизации / 10)

Параметр (*param*)	Аргумент (*value*)
	2 – от 0 до (частота дискретизации / 100) 3 – от 0 до (частота дискретизации / 1000) 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(...)*
9	Установка режима отображения данных: 0 – каждый график в своем окне 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	Маркеры неинтерполированных точек: 0 – скрыть 1 – показать
127	Установка на выключение многоканального осциллографа (аргумент может принимать любое значение)
255	Установка на включение многоканального осциллографа (аргумент может принимать любое значение)

11.3.3.28 XYZ-осциллограф (XYOscGraph)

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): 0 – от 0 до (частота дискретизации / 1) 1 – от 0 до (частота дискретизации / 10) 2 – от 0 до (частота дискретизации / 100) 3 – от 0 до (частота дискретизации / 1000) 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(...)* (аргу-

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

Параметр (*param*)	Аргумент (*value*)
	мент 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(...)*
127	Установка на выключение XYZ-осциллографа (аргумент может принимать любое значение)
255	Установка на включение XYZ-осциллографа (аргумент может принимать любое значение)

11.3.3.29 Генератор сигналов (DAC_OCX)

Параметр (*param*)	Аргумент (*value*)
0	Установка типа генерируемого сигнала: 1 – синусоидальный сигнал 2 – радиоимпульсный сигнал 3 – шум 4 – линейная частотная модуляция 5 – логарифмическая частотная модуляция 6 – импульсный сигнал 7 – сигнал из файла 8 – второй синусоидальный сигнал 9 – амплитудная модуляция 10 – частотная модуляция 11 – пилообразный сигнал 12 – сигнал с входного или виртуального канала 13 – сигнал кодов Баркера
1	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) 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)
3	Установка цикличности линейной частотной модуляции и логарифмической частотной модуляции: 0 – однократное воспроизведение сигнала 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. - формула расчета следующая интервал = (частота конечная - частота начальная) / длительность, с (для линейной модуляции) или для логарифмической интервал = (частота конечная / частота начальная) / log(2.0) / длительность, с * 60.0

Параметр (*param*)	Аргумент (*value*)
5	Установка смещения постоянной составляющей синусоидального сигнала (от минус максимально допустимого уровня сигнала ЦАП до максимально допустимого уровня сигнала ЦАП), В
6	Установка конечной частоты линейной частотной модуляции (от начальной частоты линейной частотной модуляции до (частота дискретизации ЦАП / 2)) и логарифмической частотной модуляции (от начальной частоты логарифмической частотной модуляции до (частота дискретизации ЦАП / 2)), Гц
7	Установка длительности развертки линейной частотной модуляции не может быть менее 1 с, в соответствии с этим, длительность развертки должна быть такой, что б удовлетворить этому неравенству. - формула расчета следующая интервал = (частота конечная - частота начальная) / скорость, Гц/с (для линейной модуляции) или для логарифмической интервал = (частота конечная / частота начальная) / log(2.0) / скорость, Гц/с * 60.0
8	Установка частоты заполнения 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))
12	Установка уровня шума (от 0 В до (максимально допустимый уровень сигнала ЦАП / 5))
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: 0 – белый шум 1 – полосовой шум 2 – розовый шум 3 – детерминированный шум
16	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	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 максимально допустимого уровня сигнала ЦАП), В

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

Параметр (*param*)	Аргумент (*value*)
23	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	Установка несущей частоты сигнал 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	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: 0 – нарастающий 1 – ниспадающий 2 – треугольный
35	Установка цикличности пилообразного сигнала 0 – однократное воспроизведение сигнала 1 – многократное воспроизведение сигнала
37	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: 0 – 2 1 – 3 2 – 4 3 – 5 4 – 7 5 – 11 6 – 13
42	Установка цикличности сигнала кодов Баркера 0 – однократное воспроизведение сигнала 1 – многократное воспроизведение сигнала
43	Установка порядкового номера канала для генерации сигнала с канала (от 0 до (количество каналов - 1))
44	Установка коэффициента усиления (ослабления) канала файл для генерации сигнала с канала (от 0.001 до 99.9)
70	Установка коэффициента усиления (ослабления) сигнала из файла (от 0.001 до 999.9)

Параметр (*param*)	Аргумент (*value*)
71	Установка цикличности сигнала из файла 0 – однократное воспроизведение сигнала 1 – многократное воспроизведение сигнала
72	Установка имени файла сигнала из файла
126	Установка порядкового номера генератора (от 0 до (количество генераторов - 1))
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)

11.3.3.30 Многоканальный генератор (ManyChanDac)

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
10, 20, … 100	Установка номера канала первого … десятого генератора (от 0 до (количество генераторов – 1))
11, 21, … 101	Установка типа сигнала первого … десятого генератора: 0 – синусоидальный сигнал 1 – импульсный сигнал
12, 22, … 102	Тип заполнения первого … десятого генератора: 0 – непрерывный сигнал 1 – 1/8 2 – 6/12
13, 23, … 103	Установка амплитуды первого … десятого генератора (от 0 В до максимально допустимого уровня сигнала ЦАП)
14, 24, … 104	Установка фазы первого … десятого генератора (от 0° до 360°)
15, 25, … 105	Установка смещения постоянной составляющей первого … десятого генератора (от минус максимально допустимого уровня сигнала ЦАП до максимально допустимого уровня сигнала ЦАП), В
16, 26, … 106	Установка второй амплитуды первого … десятого генератора (от 0 В до максимально допустимого уровня сигнала ЦАП)
17, 27, … 107	Установка второй фазы первого … десятого генератора (от 0° до 360°)
18, 28, … 108	Установка на получение идентификатора виртуального канала первого … десятого генератора по операции *Read(...)* (аргумент может принимать любое значение). После задания этого параметра необходимо дождаться события готовности *Ready(...)* и затем прочитать идентификатор виртуального канала первого … десятого генератора операцией *Read(...)*

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

11.3.3.31 Синхронный генератор (SynchroChanDac)

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: 0 – синусоидальный сигнал 1 – импульсный сигнал 2 – сигнал из файла 3 – логарифмическая частотная модуляция 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 канал генератора (от 0 до 1)
23	Установка смещения фазы импульсного сигнала текущего канал 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: 0 – однократное воспроизведение сигнала 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)
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 те-

Параметр (*param*)	Аргумент (*value*)
	кущего 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 разрешения инверсии сигнала логарифмической частотной модуляции текущего канала генератора: 0 – запрет инверсии 1 – разрешение инверсии
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 длительности радиоимпульсного сигнала текущего канала генератора в периодах частоты заполнения (от 0 до (частота заполнения радиоимпульсного сигнала / частота следования радиоимпульсного сигнала))
127	Установка на выключение синхронного генератора (аргумент может принимать любое значение)
255	Установка на включение синхронного генератора (аргумент может принимать любое значение)

11.3.3.32 Генератор с обратной связью (синус) (DAC_OS_sin)

Параметр (*param*)	Аргумент (*value*)
0	Установка порядкового номера канала обратной связи (от 0 до (количество каналов – 1))
1	Установка частоты сигнала (от 0.01 до (частота дискретизации ЦАП / 2)), Гц
2	Установка амплитуды сигнала (от 0 мВ до максимально допустимого уровня сигнала ЦАП)
3	Установка длительности выдержки, мин.
4	Установка максимально допустимого выходного уровня с генератора (от 0 В до максимально допустимого уровня сигнала ЦАП)
5	Установка типа задаваемого параметра: 0 – ускорение (g) 1 – ускорение (мм/с^2) 2 – скорость (мм/с) 3 – перемещение (мм)
255	Установка на включение генератора (аргумент может принимать любое значение)
127	Установка на выключение генератора (аргумент может принимать любое значение)

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

11.3.3.33 Запись сигналов (writer)

Параметр (*param*)	Аргумент (*value*)
0	Установка следующей директории для записи сигнала (аргумент может принимать любое значение)
1	Установка преамбулы записи (от 0 с до 5 с)
2	Установка длительности записи (от 10 с до 3600 с)
3	Установка каналов для записи: номер > 0 – включить канал для записи номер < 0 – выключить канал для записи
4	Установка текущей директории для записи
5	Установка типа записи: 0 – детерминированная по времени запись 1 – непрерывная запись
6	Установка на обновление списка каналов и выбор для записи каналов АЦП (аргумент может принимать любое значение)
127	Установка на останов записи (аргумент может принимать любое значение)
255	Установка на начало записи (аргумент может принимать любое значение)

11.3.3.34 Воспроизведение сигналов (reader)

Параметр (*param*)	Аргумент (*value*)
0	Установка директории, в которой находятся сигналы для воспроизведения
1	Установка скорости воспроизведения сигналов: 0 – быстрая 1 – нормальная
2	Установка типа воспроизведения: 0 – воспроизведение записи из текущей директории 1 – непрерывное воспроизведение
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)

11.3.3.35 Многоканальный самописец (multiSWvm)

Параметр (*param*)	Аргумент (*value*)
0	Установка количества измеряемых параметров (от 1 до 80)
1	Установка индекса текущего измеряемого параметра (от 0 до (количество измеряемых параметров - 1))
2	Установка метода обработки по текущему индексу измеряемого параметра: 0 – среднеквадратичное значение (в единицах измерения) 1 – амплитудное значение (в единицах измерения) 2 – пиковое значение (в единицах измерения) 3 – размах (в единицах измерения)

Параметр (*param*)	Аргумент (*value*)
	4 – постоянная составляющая сигнала (в единицах измерения) 5 – среднеквадратичное значение (в децибелах) 6 – амплитудное значение (в децибелах) 7 – пиковое значение (в децибелах) 8 – размах (в децибелах) 9 – постоянная составляющая сигнала (в децибелах) 8 – частота сигнала в герцах 9 – частота оборотов в минуту 10 – разность фаз между заданным каналом и опорной частотой, выраженная в градусах.
3	Установка порядкового номера канала по текущему индексу (от 0 до (количество каналов - 1))
4	Установка единицы измерения времени: 0 – секунда 1 – минута 2 – час
5	Установка усреднения в единицах измерения времени
6	Установка времени по горизонтальной оси: 0 – календарное время 1 – относительное время
7	Установка режима слежения за временем (при установке режима относительного времени параметром 6): 0 – режим смещения 1 – режим слежения
8	Установка смещения временной шкалы (от 0% до 100%)
9	Установка длительности отображаемой части в единицах измерения времени
10	Установка отображения таблицы: 0 – запрет отображения таблицы 1 – разрешение отображения таблицы
11	Установка варианта деления файлов при записи: 0 – без деления 1 – деление файла по суткам 2 – деление файла по неделям
12	Установка количества точек при записи в файл
13	Установка опорной частоты для расчета разности фаз
14	Установка состояния процесса записи непрерывного файла: 0 – выключение процесса записи непрерывного файла 1 – включение процесса записи непрерывного файла
15	Установка имени файла для непрерывной записи
16	Установка на получение данных по операции *Read(...)*: 0 – запрет передачи данных 1 – передача текущих значений 2 – передача размера массива данных После задания этого параметра необходимо дождаться события готовности *Ready(...)* и затем прочитать размер массива (частот и спектра) операцией *Read(...)*
255	Установка на включение многоканального самописца (аргумент может принимать любое значение)

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

Параметр (*param*)	Аргумент (*value*)
127	Установка на выключение многоканального самописца (аргумент может принимать любое значение)

11.3.3.36 Регулятор (Regulator)

Параметр (*param*)	Аргумент (*value*)
0	Установка порядкового номера 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: 0 – аналоговый выход 1 – цифровой выход
3	Установка инверсии выхода: 0 – выход без инверсии 1 – инверсный выход
4	Установка типа регулятора: 0 – релейный регулятор 1 – ПИД-регулятор
5	Установка использования пропорционального коэффициента ПИД-регулятора: 0 – запрет использования пропорционального коэффициента 1 – разрешение использования пропорционального коэффициента
6	Установка использования интегрального коэффициента ПИД-регулятора: 0 – запрет использования интегрального коэффициента 1 – разрешение использования интегрального коэффициента
7	Установка использования дифференциального коэффициента ПИД-регулятора: 0 – запрет использования дифференциального коэффициента 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: 0 – запрет амплитудной модуляции 1 – разрешение амплитудной модуляции
13	Установка разрешения широтно-импульсной модуляции: 0 – запрет широтно-импульсной модуляции 1 – разрешение широтно-импульсной модуляции
15	Установка длительности истории, с
16	Установка режима работы регулятора: 0 – работа по профилю 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

Параметр (*param*)	Аргумент (*value*)
22	Установка максимальной скважности широтно-импульсной модуляции (от минимальной скважности широтно-импульсной модуляции до 100%)
23	Установка минимальной скважности широтно-импульсной модуляции (от 0% до максимальной скважности широтно-импульсной модуляции)
24	Установка имени файла исполняемого профиля
127	Установка на выключение регулятора (аргумент может принимать любое значение)
255	Установка на включение регулятора (аргумент может принимать любое значение)

11.3.3.37 Арифмометр (ArithmoMeter)

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: 0 – сложение каналов 1 – вычитание каналов 2 – перемножение каналов 3 – деление каналов 4 – расчет максимального значения из двух каналов 5 – расчет минимального значения из двух каналов 6 – расчет среднего арифметического значения из двух каналов 7 – расчет модуля сигналов 8 – расчет среднего геометрического значения из двух каналов
3	Установка константы для умножения
4	Установка константы для сложения

11.3.3.38 Адаптивный фильтр 50 Гц (filtr50)

Параметр (*param*)	Аргумент (*value*)
0	Установка порядкового номера канала (от 0 до (количество каналов - 1))
1	Установка времени оценки частоты сигнала: 0 – 10 с 1 – 20 с 2 – 50 с 3 – 100 с
2	Установка времени оценки амплитуды сигнала: 0 – 0.1 с 1 – 0.2 с 2 – 0.5 с 3 – 1 с 4 – 2 с 5 – 5 с 6 – 10 с

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

11.3.3.39 Фильтрация сигналов (filtrdiff)

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: 0 – линейный 1 – дифференцирующий 1-го порядка 2 – дифференцирующий 2-го порядка 3 – интегрирующий 1-го порядка 4 – интегрирующий 2-го порядка
12, 22, … 1002	Установка разрешения работы фильтра верхних частот первого … сотого фильтра: 0 – выключение фильтра верхних частот 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: 0 – выключение фильтра нижних частот 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: 0 – выключение детектора огибающей 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
1010, 1020, … 2000	Установка разрешения работы фильтра частотной коррекции первого … сотого фильтра: 0 – выключение фильтра частотной коррекции 1 – включение фильтра частотной коррекции
1011, 1021, … 2001	Установка функции частотной коррекции: 0 – Wb 1 – Wc 2 – Wd 3 – We 4 – Wf 5 – Wh 6 – Wj 7 – Wk

Параметр (*param*)	Аргумент (*value*)
	8 – Wm 9 – A 10 – B 11 – C 12 – D

11.3.3.40 Формула (ZETFormula)

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 единицы 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)

11.3.3.41 Синхронизация по GPS (GPSSync)

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: 0 – 57600 бит/с 1 – 38400 бит/с 2 – 19200 бит/с 3 – 9600 бит/с 4 – 4800 бит/с
2	Установка номера последовательного порта для вывода данных GPS (от 0 до (количество COM-портов - 1))
3	Установка разрешения синхронизации: 0 – запрет синхронизации 1 – разрешение синхронизации
4	Установка времени опережения запуска (от 1 секунды до 30 секунд)

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

11.3.3.42 Управление реле (ReleComm)

Параметр (*param*)	Аргумент (*value*)
0	Установка порядкового номера модуля АЦП для управления платой реле через цифровой порт (от 0 до (количество устройств - 1))
1	Установка на переключение всех реле в положение A (аргумент может принимать любое значение)
2	Установка на переключение всех реле в положение B (аргумент может принимать любое значение)
10, 11, … 22	Установка первого … тринадцатого реле в положение B (аргумент может принимать любое значение)
30, 31, … 42	Установка первого … тринадцатого реле в положение A (аргумент может принимать любое значение)

11.3.3.43 Управление релейными ключами (ReleyComm)

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 значени».
42	Установка на переключение всех реле RX в положение «разомкнуто» (аргумент может принимать любое значение».
43	Установка на переключение всех реле RX в оложение «замкнуто» (аргумент может принимать любое значение).

11.3.3.44 Подключение устройств по Ethernet (NetWizard)

Параметр (*param*)	Аргумент (*value*)
0	Установка количества подключаемых устройств (от 0 до 1024)
1	Установка разрешения подключения только успешно прошедших проверку устройств: 0 – подключать все устройства 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)

11.3.3.45 Подключение устройств по BlueTooth (BthWizard)

Параметр (*param*)	Аргумент (*value*)
0	Установка типа подключаемого модуля АЦП: 10 – ZET 210 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)

11.3.3.46 Контроль конфигурации (ConfigControl)

Параметр (*param*)	Аргумент (*value*)
0	Установка на обновление текущей конфигурации (аргумент может принимать любое значение)
1	Установка на открытие сохраненного файла конфигурации (аргумент может принимать любое значение). После задания этого параметра необходимо дождаться события готовности *Ready(…)* и затем прочитать произошедшие изменения в конфигурации операцией *Read(…)*
2	Установка на сохранение текущей конфигурации в файл (аргумент может принимать любое значение)
3	Установка на откат к последней открытой конфигурации (аргумент может принимать любое значение)

11.3.3.47 Высокочастотный осциллограф (ZetScope)

Параметр (*param*)	Аргумент (*value*)
0	Установка канала, настройки которого будут устанавливаться: 1 – первый канал 2 – второй канал
1	Установка видимости выбранного канала на экране: 0 – канал виден 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: 0 – по постоянному току (DC) 1 – по переменному току (AC)
4	Установка коэффициента ослабления пробника выбранного канала: 0 – 1x 1 – 10x
5	Установка инверсии сигнала выбранного канала:

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

Параметр (*param*)	Аргумент (*value*)
	0 – сигнал не инвертируется 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: 0 – канал виден 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: 0 – первый канал + второй канал 1 – первый канал - второй канал 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: 0 – первый канал 1 – второй канал
24	Установка фронта синхронизации: 0 – восходящий 1 – нисходящий
25	Установка режима синхронизации: 0 – автоматический 1 – обычный
26	Установка интерполяции сигнала: 0 – без интерполяции 1 – линейная интерполяция 2 – интерполяция sin(x)/x
27	Установка послесвечения: 0 – послесвечение отключено 1 – послесвечение 1 секунда 2 – послесвечение 2 секунды 3 – послесвечение 5 секунд 4 – бесконечное послесвечение 5 – цифровое послесвечение
28	Установка режима сбора данных: 0 – усреднение 1 – пиковый 2 – выборка
29	Установка количества усреднений (в режиме сбора данных «усреднение»): 0 – 4 1 – 8

Параметр (*param*)	Аргумент (*value*)
	2 – 16 3 – 32 4 – 64
30	Установка видимости панели управления: 0 – панель управления видна 1 – панель управления не видна
41	Установка номера поля измерений, настройки которого будут устанавливаться (от 0 до 4)
42	Установка канала для выбранного поля: 0 – первый канал 1 – второй канал
43	Установка вида измерения для выбранного поля: – при обычных измерениях: 0 – частота 1 – период 2 – среднее 3 – размах 4 – СКЗ 5 – минимум 6 – максимум 7 – нарастание 8 – спад 9 – положительный импульс 10 – отрицательный импульс 11 – положительный выброс 12 – отрицательный выброс 13 – высокое 14 – низкое 15 – нет измерения – при измерениях с вертикальными курсорами: 0 – разность в вольтах 1 – разность в секундах 2 – курсор 1 в секундах 3 – курсор 1 в вольтах 4 – курсор 2 в секундах 5 – курсор 2 в вольтах 6 – частота 7 – нет измерения – при измерениях с горизонтальными курсорами: 0 – разность в вольтах 1 – курсор 1 в вольтах 2 – курсор 2 в вольтах 3 – нет измерения
44	Установка типа измерений: 1 – обычные измерения 2 – измерения с использованием вертикальных курсоров 3 – измерения с использованием горизонтальных курсоров
45	Установка сохранения изменений в окне измерений (при открытом окне измерений):

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

Параметр (*param*)	Аргумент (*value*)
	0 – отменить изменения и закрыть окно 1 – сохранить изменения и закрыть окно
50	Установка работы окна спектра: 0 – закрытие окна спектра 1 – открытие окна спектра
51	Установка видимости графика спектра выбранного канала: 0 – спектр не виден 1 – спектр виден
52	Установка типа весового окна при расчете спектра: 0 – прямоугольное 1 – Хана 2 – Хэмминга 3 – Блэкмана 4 – Барлета 5 – Блэкмана стд.
255	Установка ведения сбора данных: 0 – сбор данных не ведется 1 – сбор данных ведется

11.3.4 Формат посылаемых данных

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*.

11.3.4.1 Узкополосный спектральный анализ (spectr)

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)

11.3.4.2 Долеоктавный спектральный анализ (dspectr)

Параметр (*param*)	Данные (*data*)
1	Передача размера массива спектра (если была установка на получение размера массива спектра параметром 3)
2	Передача массива, состоящего из: – значений текущего спектра (если была установка на получение данных параметром 4) – значений частотного ряда (если была установка на получение данных параметром 4) – значений максимального спектра (если была установка на получение данных параметром 4 и разрешение расчета максимального спектра параметром 10) – значений минимального спектра (если была установка на получение данных параметром 4 и разрешение расчета минимального спектра параметром 11) – значений среднего спектра (если была установка на получение данных параметром 4 и разрешение расчета среднего спектра параметром 12)
3	Передача массива значений частотного ряда (если была установка на получение значений частотного ряда параметром 6)

11.3.4.3 Взаимный узкополосный спектральный анализ (vspectr)

Параметр (*param*)	Данные (*data*)
0	Передача: – размера массива спектра (если была установка на получение размера массива спектра параметром 9) – массива значений частотного ряда (если была установка на получение значений частотного ряда параметром 10 или была установка на последовательное получение данных параметром 12) – 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 значений переходной характеристики (если была установка на последовательное получение данных параметром 12 и разрешение расчета переходной характеристики параметром 18)

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

11.3.4.4 Взаимный долеоктавный спектральный анализ (dvspectr)

Параметр (*param*)	Данные (*data*)
0	Передача: – размера массива спектра (если была установка на получение размера массива спектра параметром 4) – массива, состоящего из: 1) значений частотного ряда (если была установка на получение данных параметром 5) 2) значений текущего спектра (если была установка на получение данных параметром 5) 3) значений действительной части спектра (если была установка на получение данных параметром 5 и разрешение расчета действительной части спектра параметром 13) 4) значений мнимой части спектра (если была установка на получение данных параметром 5 и разрешение расчета мнимой части спектра параметром 14) 5) значений фазы (если была установка на получение данных параметром 5 и разрешение расчета фазы параметром 15) 6) значений коэффициента когерентности (если была установка на получение данных параметром 5 и разрешение расчета коэффициента когерентности параметром 16) – массива значений частотного ряда (если была установка на получение данных параметром 6) – массива значений текущего спектра (если была установка на получение данных параметром 7) – массива значений действительной части спектра (если была установка на получение данных параметром 8 и разрешение расчета действительной части спектра параметром 13) – массива значений мнимой части спектра (если была установка на получение данных параметром 9 и разрешение расчета мнимой части спектра параметром 14)

11.3.4.5 Взаимный корреляционный анализ (corr)

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)

11.3.4.6 Анализ нелинейных искажений (harmdist)

Параметр (*param*)	Данные (*data*)
0	Передача массива, состоящего из 2 элементов: 0 – значение преобладающей гармоники 1 – коэффициент нелинейных искажений

11.3.4.7 Синхронное накопление (PrqsAnaliz)

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)
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

11.3.4.8 Модальный анализ (PrqsAnaliz)

Параметр (*param*)	Данные (*data*)
0	Передача массива, состоящего из 22 элементов: 0 – амплитуда удара (оп. канал) 1 – амплитуда удара (изм. канал) 2 – ширина импульса по 50-процентному уровню (оп. канал) 3 – ширина импульса по 50-процентному уровню (изм. канал) 4 – ширина импульса по 10-процентному уровню (оп. канал) 5 – ширина импульса по 10-процентному уровню (изм. канал) 6 – СКЗ шума до импульса (оп. канал) 7 – СКЗ шума до импульса (изм. канал) 8 – порог срабатывания на импульс (оп. канал) 9 – порог срабатывания на импульс (изм. канал) 10 – интеграл первого порядка от импульса (оп. канал) 11 – интеграл первого порядка от импульса (изм. канал) 12 – интеграл второго порядка от импульса (оп. канал) 13 – интеграл второго порядка от импульса (изм. канал) 14 – интеграл произведения силы на перемещение (оп. канал) 15 – интеграл произведения силы на перемещение (изм. канал) 16 – время прохождения импульса (оп. канал) 17 – время прохождения импульса (изм. канал) 18 – соотношение сигнал-шум (оп. канал) 19 – соотношение сигнал-шум (изм. канал) 20 – отношение амплитуд импульсов на измерительном и опорном каналах 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

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

11.3.4.9 Гистограмма (ZETHistograph)

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)

11.3.4.10 Детектор STA\LTA (STA_LTA)

Параметр (*param*)	Данные (*data*)
0	Передача значения, информирующего о событии: 0 – событие закончилось; 1 – событие началось
1	Передача значения идентификатора виртуального канала STA_LTA (если была установка на разрешение передачи данных в виртуальный канал STA_LTA)

11.3.4.11 Вольтметр переменного тока (VoltMeter)

Параметр (*param*)	Данные (*data*)
0	Передача измеренного значения переменной составляющей сигнала

11.3.4.12 Вольтметр постоянного тока (VoltMeterDC)

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)

11.3.4.13 Селективный вольтметр (VoltMeterSel)

Параметр (*param*)	Данные (*data*)
0	Передача измеренного значения переменной составляющей сигнала

11.3.4.14 Частотомер(FreqMeter)

Параметр (*param*)	Данные (*data*)

Параметр (*param*)	Данные (*data*)
0	Передача измеренного значения частоты сигнала

11.3.4.15 Фазометр(FasoMeter)

Параметр (*param*)	Данные (*data*)
0	Передача измеренного значения разности фаз сигналов

11.3.4.16 Тахометр (TahoMeter)

Параметр (*param*)	Данные (*data*)
0	Передача измеренного значения частоты вращения
1	Передача измеренного значения общего количества оборотов с момента последнего сброса счетчика оборотов

11.3.4.17 Торсиограф(Torsiograph)

Параметр (*param*)	Данные (*data*)
0	Передача измеренного значения перемещения

11.3.4.18 Энкодер(Encoder)

Параметр (*param*)	Данные (*data*)
0	Передача измеренного значения перемещения

11.3.4.19 Термометр ТС(ThermoMeter)

Параметр (*param*)	Данные (*data*)
0	Передача измеренного значения температуры

11.3.4.20 Термометр ТП(ThermoPara)

Параметр (*param*)	Данные (*data*)
0	Передача измеренного значения температуры

11.3.4.21 Тензометр(TenzoMeter)

Параметр (*param*)	Данные (*data*)
0	Передача измеренного значения нагрузки

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

11.3.4.22 Виброметр (VibroMeter)

Параметр (*param*)	Данные (*data*)
0	Передача массива, состоящего из 3 элементов: 0 – значение ускорения 1 – значение скорости 2 – значение перемещения

11.3.4.23 Мультиметр Agilent_HP34401A (Agilent_HP34401A)

Параметр (*param*)	Данные (*data*)
0	Передача измеренного значения величины

11.3.4.24 Блок питания LPS305 (LPS305)

Параметр (*param*)	Данные (*data*)
0	Передача массива, состоящего из 17 элементов: 0 – текущее напряжение по первому каналу 1 – текущее напряжение по второму каналу 2 – текущий ток по первому каналу 3 – текущий ток по второму каналу 4 – режим ограничения по первому каналу: 0 – ограничение по напряжению (CV) 1 – ограничение по току (CC) 5 – режим ограничения по первому каналу: 0 – ограничение по напряжению (CV) 1 – ограничение по току (CC) 6 – режим слежения: 0 – независимая работа каналов 1 – слежение за первым каналом 7 – состояние питания цифрового выхода: 0 – цифровой выход отключен 1 – питание 3.3 В 2 – питание 5 В 8 – состояние выходов: 0 – выходы выключены 1 – выходы включены 9 – состояние перегрузки цифрового выхода: 0 – цифровой выход не перегружен 1 – цифровой выход перегружен 10 – состояние кулера: 0 – кулер выключен 1 – кулер включен 11 – состояние звукового сигнала: 0 – звуковой сигнал выключен 1 – звуковой сигнал включен 12 – состояние работы компенсации выхода по току

Параметр (*param*)	Данные (*data*)
	0 – компенсация выхода по току выключена 1 – компенсация выхода по току включена 13 – заданное ограничение по напряжению на первом канале 14 – заданное ограничение по напряжению на втором канале 15 – заданное ограничение по току на первом канале 16 – заданное ограничение по току на втором канале

11.3.4.25 Блок питания PPE3323 (PPE3323)

Параметр (*param*)	Данные (*data*)
0	Передача: – значения ошибки при работе с прибором: 256 – устройство отсутствует – массива, состоящего из 22 элементов: 0 – текущее напряжение по первому каналу 1 – текущее напряжение по второму каналу 2 – текущий ток по первому каналу 3 – текущий ток по второму каналу 4 – режим стабилизации по напряжению 0 – режим стабилизации по напряжению выключен 1 – режим стабилизации по напряжению включен 5 – режим стабилизации по току 0 – режим стабилизации по току выключен 1 – режим стабилизации по току включен 6 – режим слежения: 0 – независимая работа каналов 1 – слежение за первым каналом 7 – состояние питания цифрового выхода: 1 – питание 3.3 В 2 – питание 5 В 8 – состояние выходов: 0 – выходы выключены 1 – выходы включены 9 – последовательное соединение выходов: 0 – выбран первый канал 1 – выбран второй канал 2 – последовательное соединение выходов 10 – 0 (не используется) 11 – 0 (не используется) 12 – состояние защиты от перегрузки 0 – защита от перегрузки выключена 1 – защита от перегрузки включена 13 – заданное ограничение по напряжению на первом канале 14 – заданное ограничение по напряжению на втором канале 15 – заданное ограничение по току на первом канале 16 – заданное ограничение по току на втором канале 17 – заданный уровень защиты по напряжению на первом канале

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

Параметр (*param*)	Данные (*data*)
	18 – заданный уровень защиты по напряжению на втором канале 19 – заданное ограничение по напряжению при последовательном соединении выходов 20 – заданное ограничение по току при последовательном соединении выходов 21 – заданный уровень защиты по напряжению при последовательном соединении выходов

11.3.4.26 Блок питания PSM2010 (PSM2010)

Параметр (*param*)	Данные (*data*)
0	Передача массива, состоящего из 13 элементов: 0 – текущее напряжение 1 – текущее напряжение 2 – состояние выходов: 0 – выходы выключены 1 – выходы включены 3 – состояние защиты от перегрузки (OCP) 0 – защита от перегрузки выключена 1 – защита от перегрузки включена 4 – состояние защиты от перенапряжения (OVP) 0 – защита от перенапряжения выключена 1 – защита от перенапряжения включена 5 – текущий рабочий диапазон: 0 – 8V, 20A 1 – 20V, 10A 6 – заданное ограничение по напряжению 7 – заданное ограничение по току 8 – заданный уровень защиты по напряжению 9 – заданный уровень защиты от перегрузки 10 – индикация события перенапряжения 0 – перенапряжение отсутствует 1 – существует перенапряжение 11 – индикация события перегрузки 0 – перегрузка отсутствует 1 – существует перегрузка 12 – код ошибки

11.3.4.27 Многоканальный осциллограф (OscGraph)

Параметр (*param*)	Данные (*data*)
0	Передача: – размера массива данных (если была установка на получение размера массива данных параметром 7) – массива данных (если была установка на получение массива данных параметром 8)

11.3.4.28 XYZ-осциллограф (XYOscGraph)

Параметр (*param*)	Данные (*data*)
0	Передача: – размера массива данных (если была установка на получение размера массива данных параметром 7) – массива данных (если была установка на получение массива данных параметром 8)

11.3.4.29 Генератор сигналов (DAC_OCX)

Параметр (*param*)	Данные (*data*)
0	Передача значения текущей частоты сигнала линейной или логарифмической частотной модуляции

11.3.4.30 Многоканальный генератор (ManyChanDac)

Параметр (*param*)	Данные (*data*)
В качестве параметра передается идентификатор виртуального канала первого … десятого канала генератора, заданного параметром 18 … 108	0

11.3.4.31 Синхронный генератор (SynchroChanDac)

Программа не передает в вызывающую программу никаких данных.

11.3.4.32 Генератор с обратной связью (синус) (DAC_OS_sin)

Параметр (*param*)	Данные (*data*)
0	Передается текущее значение уровня сигнала в режиме установки заданного значения уровня
1	Передается текущее значение уровня сигнала в режиме точной (< 2%) установки заданного значения уровня
2	Передается текущее значение уровня сигнала при обрыве в случае, если уровень шума, измеренный селективным вольтметром, больше 50% значения, измеренного селективным вольтметром.
3	Передается текущее значение уровня сигнала при обрыве в случае, если уровень сигнала, измеренный селективным вольтметром, меньше 50% значения, измеренного интегральным вольтметром
4	Передается текущее значение уровня сигнала при перегрузке по уровню генератора
6	Передается текущее значение уровня сигнала при выключении генератора с обратной связью

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

11.3.4.33 Запись сигналов (writer)

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.)

11.3.4.34 Воспроизведение сигналов (reader)

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

11.3.4.35 Многоканальный самописец (miltiSWvm)

Параметр (*param*)	Данные (*data*)
0	Передача: – массива текущих значений по всем измеряемым параметрам (если была установка на получение массива текущих значений параметром 16) – размера массива данных (если была установка на получение размера массива данных параметром 16)

11.3.4.36 Регулятор (Regulator)

Параметр (*param*)	Данные (*data*)
0	Передача измеренного значения контролируемого сигнала

11.3.4.37 Арифмометр (ArithmoMeter)

Параметр (*param*)	Данные (*data*)
0	Передача измеренного значения результирующего сигнала

11.3.4.38 Адаптивный фильтр 50 Гц (filtr50)

Параметр (*param*)	Данные (*data*)
0	Передача массива, состоящего из 2 элементов: 0 – значение частоты сигнала 1 – значение амплитуды сигнала

11.3.4.39 Фильтрация сигналов (filtrdiff)

Параметр (*param*)	Данные (*data*)
В качестве параметра передается идентификатор виртуального канала первого … сотого канала фильтра, заданного параметром 18 … 1008	0

11.3.4.40 Формула (ZETFormula)

Программа не передает в вызывающую программу никаких данных.

11.3.4.41 Синхронизация по GPS (GPSSync)

Параметр (*param*)	Данные (*data*)
0	Передача массива, состоящего из 11 элементов: 0 – год 1 – месяц 2 – день 3 – час 4 – минута 5 – секунда 6 – широта 7 – долгота 8 – скорость 9 – количество спутников 10 – сдвиг по времени

11.3.4.42 Управление реле (ReleComm)

Программа не передает в вызывающую программу никаких данных.

11.3.4.43 Управление релейными ключами (ReleyComm)

Программа не передает в вызывающую программу никаких данных.

11.3.4.44 Подключение устройств по Ethernet (NetWizard)

Программа не передает в вызывающую программу никаких данных.

11.3.4.45 Подключение устройств по BlueTooth (BthWizard)

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: 0 – активация прошла успешно 1 – не удалось активировать модуль

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

	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: 0 – деактивация прошла успешно 1 – не удалось деактивировать модуль 2 – деактивация недоступна

11.3.4.46 Контроль конфигурации (ConfigControl)

Параметр (*param*)	Данные (*data*)
0	Передача массива, состоящего из (количество устройств + 1) элементов. Каждый член массива, кроме последнего соответствует устройству по порядку: = 0 – в конфигурации *i-ого* устройства нет изменений ≠ 0 – в конфигурации *i-ого* устройства есть изменения, при этом каждый взведенный бит *data[i]* отвечает за: 1 – изменились настройки измерительных каналов устройства 2 – изменились настройки линий цифрового порта устройства 3 – изменилась частота дискретизации АЦП устройства 4 – изменилась частота дискретизации ЦАП устройства 5 – изменилось количество включенных каналов устройства Каждый взведенный бит последнего элемента массива отвечает за: 1 – изменился порядок устройств в системе 2 – в системе появились новые устройства 3 – в системе пропали существовавшие устройства

11.3.4.47 Высокочастотный осциллограф (ZetScope)

Параметр (*param*)	Данные (*data*)
0	Передача массива, состоящего из: – значений по первому каналу осциллографа – значений по второму каналу осциллографа
1	Передача размера массива значений по первому и второму каналу осциллографа
2	Передача значения кода ошибки: 1 – невозможно подключиться к прибору 2 – прибор не отвечает на обмен данными 3 – прибор не запускается 4 – не найдено ни одного устройства

11.3.4.48 Автономный регистратор (Registrator)

Параметр (*param*)	Данные (*data*)
0	Передача порядкового номера устройства в системе с функцией регистратора, с которым будет производиться дальнейшая работа. Пример (param, value): 0,0 – выбираем первое устройство;

Параметр (*param*)	Данные (*data*)
	0,2 – выбираем третье устройство;
1	Передача запроса на получение количества файлов, сохраненных на флеш карте. Ответ приходит в виде числа в параметр 1.
2	Передача запроса на получение информации о файле. Операнд value – номер файла. Ответ приходит в виде строки wchar_t, т.е. полученный массив float-ов нужно привести к данному типу. Приходит в параметр 2.
3	Передача запроса на получение времени и даты с устройства. Ответ приходит в виде массива в параметр 3. Структура массива: 0 – день 1 – месяц 2 – год 3 – час 4 – минута 5 – секунда 6 – миллисекунда
4	Синхронизировать время устройства со временем компьютера.
5	Выбрать файл для скачивания. Укажите номер файла на устройстве в операнде value. Для того, чтобы получить список файлов предварительно воспользуйтесь командой с параметром «1» (см. выше). Пример (param, value): 4,4 – выбираем пятый файл; 4,7 – выбираем восьмой файл;
6	Скачать выбранный файл с флеш карты устройства. Для того, чтобы выбрать файл предварительно воспользуйтесь командой с параметром «4» (см. выше). Для скачивания требуется указать строку с полным адресом директории, в которую вы хотите скачать файл. Пример (param, StringVal, timer): 5,L”C:\\test\\”, 0 – Сохраняем файл в директорию C:\test
7	Закачать файл на флеш карту устройства. Укажите полное имя файла в операнде StringVal. Пример (param, StringVal, timer): 6,L”C:\\test\\sig0001.anp”, 0 – Записываем на флеш карту устройства файл C:\test\sig0001.anp.
8	Закрыть программу.

Примечание: во время скачивания файла с флеш карты или загрузки на нее, доступ через программу Unit блокируется, на все запросы отправляется ответ со значением -1 в параметр 0. После завершения процедуры скачивания/загрузки доступ возобновляется.

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

12 Описание функций доступа к драйверам ZET через библиотеку Zadc.dll

12.1 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.

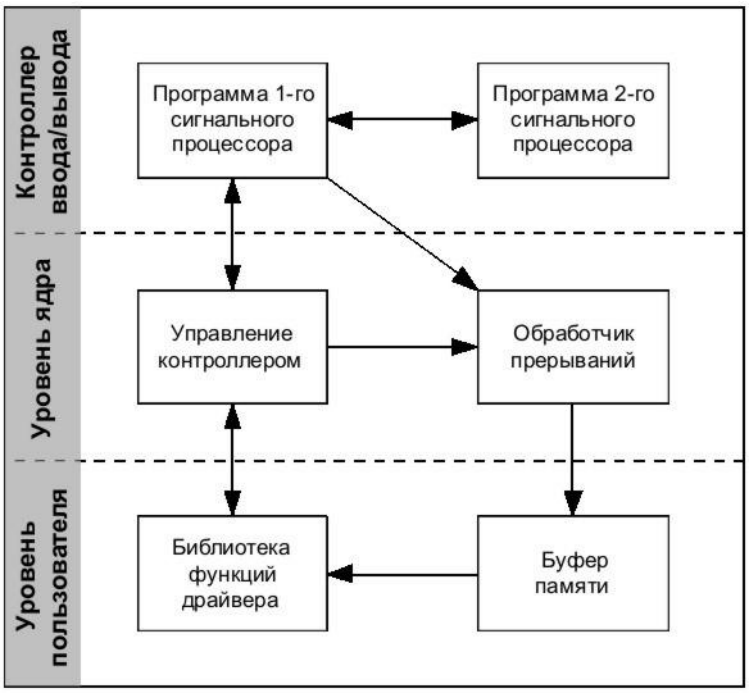


Рисунок 1.

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.

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.

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*).

12.2 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.

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

12.2.1 Подключение к драйверу

long ZOpen(long typeDevice, long numberDSP) *(Zadc.dll)*

функция для подключения к драйверу. При работе с платами АЦП и ЦАП – это первая функция, к которой необходимо произвести обращение.

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 (ZET019-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, 0x0900, 0x0A00, 0x0B00 – ошибка открытия драйвера,
0x0C00 – неподдерживаемая версия драйвера.
Функция не меняет признак *modify* в драйвере.

12.2.2 Отключение от драйвера

long ZClose(long typeDevice, long numberDSP) *(Zadc.dll)*

функция для отключения от сигнального процессора. Эта функция – последнее обращение к драйверу перед выходом из программы.

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 (ZET019-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.

12.3 Сброс и инициализация

12.3.1 Сброс и останов сигнального процессора

long ZResetDSP(long typeDevice, long numberDSP) *(Zadc.dll)*

Функция сбрасывает все сигнальные процессоры одного устройства. 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.

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;

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

15 – тип устройства ZET017-U2 (ZET019-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 открытия драйвера,
Функция меняет признак *modify* в драйвере.

12.3.2 Инициализация сигнального процессора

*long ZInitDSP(long typeDevice, long numberDSP, char *fileName)* *(Zadc.dll)*

Функция 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.

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;
        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;
}
```

12.4 Сервисные функции

12.4.1 Опрос версии программ и драйвера

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

(Zadc.dll)

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

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.

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

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

12.4.2 Чтение кода ошибки

*long ZGetError(long typeDevice, long numberDSP, long *error)* *(Zadc.dll)*

Функция возвращает код последней внутренней ошибки драйвера (обработчика прерываний)

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.

12.4.3 Опрос изменения режима работы сигнального процессора

*long ZGetModify (long typeDevice, long numberDSP, long *modify)* *(Zadc.dll)*

Функция 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.

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,
0x0800 – ошибка открытия драйвера,

Функция не меняет признак *modify* в драйвере.

12.5 Установка режима работы сигнального процессора.

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

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.

12.5.1 Установка работы с модулем АЦП

long ZSetTypeADC(long typeDevice, long numberDSP) *(Zadc.dll)*
установка сигнального процессора в режиме работы с модулем АЦП.

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.

12.5.2 Установка работы с модулем ЦАП

long ZSetTypeDAC(long typeDevice, long numberDSP) *(Zadc.dll)*
установка сигнального процессора в режиме работы с модулем ЦАП.

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.

12.6 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.

12.6.1 Опрос возможности работы сигнального процессора с модулем АЦП

*long ZGetEnableADC(long typeDevice, long numberDSP, long *enable)* *(Zadc.dll)*
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.

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,
0x0800 – ошибка открытия драйвера,
Функция не меняет признак *modify* в драйвере.

12.6.2 Опрос возможности работы сигнального процессора с модулем ЦАП

*long ZGetEnableDAC(long typeDevice, long numberDSP, long *enable) (Zadc.dll)*
опрос возможности работы сигнального процессора с модулем ЦАП.
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.

12.6.3 Опрос максимального количества каналов модуля АЦП/ЦАП

*long ZGetQuantityChannelADC (long typeDevice, long numberDSP, long *quantityChannel)*
*long ZGetQuantityChannelDAC (long typeDevice, long numberDSP, long *quantityChannel)*
(Zadc.dll)
опрос максимального количества каналов модуля АЦП/ЦАП.
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.

12.6.4 Опрос веса младшего разряда АЦП/ЦАП

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

опрос веса младшего разряда АЦП/ЦАП. 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).

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

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.

12.6.5 Опрос количества двоичных разрядов АЦП/ЦАП

*long ZGetBitsADC (long typeDevice, long numberDSP, long *numberBits)* *(Zadc.dll)*
*long ZGetBitsDAC (long typeDevice, long numberDSP, long *numberBits)*
опрос разрядности. 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.

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.

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

*long ZGetWordsADC (long typeDevice, long numberDSP, long *numberWords)* *(Zadc.dll)*
*long ZGetWordsDAC (long typeDevice, long numberDSP, long *numberWords)*
опрос размера каждого отсчета АЦП/ЦАП в 16-разрядных словах. 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++,
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

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.

12.7 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 Z*GetListFreqADC() (*Z*GetListFreqDAC())* можно получить список возможных частот дискретизации. А при помощи функции Z*SetNextFreqADC() (*Z*SetNextFreqDAC())* можно поменять текущее значение частоты дискретизации в большую или меньшую сторону на один шаг.
Задается требуемая частота дискретизации, драйвер через функцию Z*SetFreqADC() (*Z*SetFreqDAC())* устанавливает ближайшую возможную частоту дискретизации и возвращает точное значение установленной частоты дискретизации.

12.7.1 Получение списка возможных частот дискретизации АЦП/ЦАП

*long ZGetListFreqADC (long typeDevice, long numberDSP, long next, double *freq) (Zadc.dll)*
*long ZGetListFreqDAC (long typeDevice, long numberDSP, long next, double *freq)*
получение списка возможных частот дискретизации. При многократном обращении к этой функции, она возвращает возможные частоты дискретизации.
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

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

```
#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
                                                                  //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;
}
```

12.7.2 Установка большей или меньшей частоты дискретизации АЦП/ЦАП

*long ZSetNextFreqADC(long typeDevice, long numberDSP, long next, double *freq)* *(Zadc.dll)*
*long ZSetNextFreqDAC(long typeDevice, long numberDSP, long next, double *freq)*
функция устанавливает большую или меньшую частоту дискретизации по сравнению с текущей частотой дискретизации.

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.

12.7.3 Опрос текущей частоты дискретизации АЦП/ЦАП

*long ZGetFreqADC(long typeDevice, long numberDSP, double *freq)* *(Zadc.dll)*
*long ZGetFreqDAC(long typeDevice, long numberDSP, double *freq)*
опрос текущего значения частоты дискретизации
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.

12.7.4 Установка частоты дискретизации АЦП/ЦАП

*long ZSetFreqADC(long typeDevice, long numberDSP, double freqIn, double *freqOut)* *(Zadc.dll)*
*long ZSetFreqDAC(long typeDevice, long numberDSP, double freqIn, double *freqOut)*
функция устанавливает частоту дискретизации, наиболее близкую к требуемой, и возвращает значение установленной частоты дискретизации. Эти частоты могут различаться.

typeDevice – тип устройства.

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

12.7.5 Опрос текущей опорной частоты АЦП/ЦАП

*long ZGetExtFreqADC(long typeDevice, long numberDSP, double *oporFreq)* *(Zadc.dll)*
*long ZGetExtFreqDAC(long typeDevice, long numberDSP, double *oporFreq)*
опрос значения текущей опорной частоты.
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.

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

Функция не меняет признак *modify* в драйвере.

12.7.6 Установка значения внешней опорной частоты АЦП/ЦАП

long ZSetExtFreqADC(long typeDevice, long numberDSP, double extFreq) *(Zadc.dll)*
long ZSetExtFreqDAC(long typeDevice, long numberDSP, double extFreq)
установка значения внешней опорной частоты.

typeDevice – тип устройства.

numberDSP – номер сигнального процессора (значение от 0 до 136),
extFreq – 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 меняет признак *modify* в драйвере.

12.7.7 Опрос режима работы от внешней опорной частоты

*long ZGetEnableExtFreq(long typeDevice, long numberDSP, long *enable)* *(Zadc.dll)*
опрос режима работы от внешней опорной частоты.

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.

12.7.8 Установка режима работы от внешней опорной частоты АЦП

long ZSetEnableExtFreq(long typeDevice, long numberDSP, long enable) *(Zadc.dll)*
установка/сброс режима работы от внешней опорной частоты. Влияет и на АЦП и на ЦАП.

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.

12.7.9 Опрос разрешения внешнего запуска накопления данных

*long ZGetEnableExtStart(long typeDevice, long numberDSP, long *enable)* *(Zadc.dll)*

Функция опрашивает разрешение/запрещение начала накопления данных от внешнего сигнала запуска.

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.

12.7.10 Установка внешнего запуска накопления данных

long ZSetEnableExtStart(long typeDevice, long numberDSP, long enable) (Zadc.dll)
Функция 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.

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.

12.8 Управление каналами ввода (вывода) АЦП/ЦАП

12.8.1 Опрос количества включенных каналов АЦП/ЦАП

*long ZGetNumberInputADC(long typeDevice, long numberDSP, long *workChannel) (Zadc.dll)*
*long ZGetNumberOutputDAC(long typeDevice, long numberDSP, long *workChannel)*
опрос количества включенных каналов для ввода данных.

typeDevice – тип устройства.

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

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.

12.8.2 Опрос разрешения канала для ввода (вывода) АЦП/ЦАП

*long ZGetInputADC(long typeDevice, long numberDSP, long numberChannel, long *enable)*
*long ZGetOutputDAC(long typeDevice, long numberDSP, long numberChannel, long *enable)*
(Zadc.dll)

Опрос разрешения канала для ввода
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:
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.

12.8.3 Включение канала для ввода (вывода) АЦП/ЦАП

long ZSetInputADC(long typeDevice, long numberDSP, long numberChannel, long enable)
long ZSetOutputDAC(long typeDevice, long numberDSP, long numberChannel, long enable)
(Zadc.dll)

Выбор или отмена канала для ввода.
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.

12.8.4 Опрос типа канала ввода АЦП

*long ZGetInputDiffADC(long typeDevice, long numberDSP, long numberChannel, long *enable)*

(Zadc.dll)

Опрос типа канала для ввода (синфазный или дифференциальный).

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:

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.

12.8.5 Установка типа канала ввода АЦП

long ZSetInputDiffADC(long typeDevice, long numberDSP, long numberChannel, long enable)

(Zadc.dll)

Опрос типа канала для ввода (синфазный или дифференциальный).

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.

12.9 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.

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

12.9.1 Получение списка возможных коэффициентов усиления АЦП

*long ZGetListAmplifyADC(long typeDevice, long numberDSP, long next, double *amplify)* *(Zadc.dll)*

получение списка возможных коэффициентов усиления. При многократном обращении к этой функции, она возвращает возможные коэффициенты усиления.

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:

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.

12.9.2 Установка большего или меньшего коэффициента усиления выбранного канала АЦП

*long ZSetNextAmplifyADC(long typeDevice, long numberDSP, long numberChannel, long next, double *amplify)* *(Zadc.dll)*

установка большего или меньшего коэффициента усиления выбранного канала.

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.)
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.

12.9.3 Опрос коэффициента усиления выбранного канала АЦП

*long ZGetAmplifyADC(long typeDevice, long numberDSP, long numberChannel, double *amplify)* *(Zadc.dll)*

опрос коэффициента усиления выбранного канала.

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.

Функция меняет признак *modify* в драйвере.

12.9.4 Установка коэффициента усиления выбранного канала АЦП

*long ZSetAmplifyADC(long typeDevice, long numberDSP, long numberChannel, double amplifyIn, double *amplifyOut) (Zadc.dll)*

установка коэффициента усиления выбранного канала.

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:

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;

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


ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

```
        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;
}
```

12.10 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 Z*GetListPreAmplifyADC()* можно получить список возможных коэффициентов усиления. А при помощи функции Z*SetNextPreAmplifyADC()* можно поменять текущее значение коэффициента усиления в большую или меньшую сторону на один шаг.

Задается требуемый коэффициента усиления, драйвер через функцию Z*SetPreAmplifyADC()* устанавливает ближайший возможный коэффициент усиления и возвращает точное значение установленного коэффициента усиления.

12.10.1 Получение списка возможных коэффициентов усиления предварительного усилителя

*long ZGetListPreAmplifyADC(long typeDevice, long numberDSP, long next, double *amplify)*
(Zadc.dll)

получение списка возможных коэффициентов усиления предварительного усилителя. При многократном обращении к этой функции, она возвращает возможные коэффициенты усиления предварительного усилителя.

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:
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.

12.10.2 Установка большего или меньшего коэффициента усиления предварительного усилителя

*long ZSetNextPreAmplifyADC(long typeDevice, long numberDSP, long numberChannel, long next, double *amplify)* *(Zadc.dll)*
установка большего или меньшего коэффициента усиления предварительного усилителя выбранного канала.
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:
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.

12.10.3 Установка коэффициента усиления предварительного усилителя выбранного канала

*long ZSetPreAmplifyADC(long typeDevice, long numberDSP, long numberChannel, double amplifyIn, double *amplifyOut)* *(Zadc.dll)*
установка коэффициента усиления предварительного усилителя выбранного канала.
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,

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

amplifyOut – возвращаемое значение установленного коэффициента усиления предварительного усилителя.
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 меняет признак *modify* в драйвере.

12.10.4 Опрос коэффициента усиления предварительного усилителя выбранного канала

*long ZGetPreAmplifyADC(long typeDevice, long numberDSP, long numberChannel, double *amplify)* *(Zadc.dll)*
опрос коэффициента усиления предварительного усилителя выбранного канала.
typeDevice – тип устройства.

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

12.11 Управление коэффициентами ослабления аттенюатора ЦАП

12.11.1 Опрос коэффициента ослабления аттенюатора выбранного канала

*long ZGetAttenDAC(long typeDevice, long numberDSP, long numberChannel, double *reduction)* *(Zadc.dll)*
опрос коэффициента ослабления аттенюатора выбранного канала
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.

12.11.2 Установка коэффициента ослабления аттенюатора выбранного канала

*long ZSetAttenDAC(long typeDevice, long numberDSP, long numberChannel, double reductionIn, double *reductionOut)* *(Zadc.dll)*
установка коэффициента ослабления аттенюатора заданного канала аналогового вывода.
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.
Функция меняет признак *modify* в драйвере.

12.12 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

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

8	6	1
9	7	1
10	1	2
11	3	2
12	5	2
13	6	2
14	7	2

12.12.1 Установка режима внешней подкачки данных или внутренней генерации сигналов

long ZSetExtCycleDAC(long typeDevice, long numberDSP, long enable) *(Zadc.dll)*

установка режима работы ЦАП. 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.

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:
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.

12.12.2 Опрос размера буфера для перекачки данных АЦП/ЦАП

*long ZGetInterruptADC(long typeDevice, long numberDSP, long * size)* *(Zadc.dll)*
*long ZGetInterruptDAC(long typeDevice, long numberDSP, long * size)*
опрос размера буфера для перекачки данных за одно прерывание
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),
0x0100 – *numberDSP* задан неправильно,
0x0800 – ошибка открытия драйвера,
0xFF00 – для данного типа устройства функция не поддерживается.
Функция не меняет признак *modify* в драйвере.

12.12.3 Опрос максимального размера буфера для перекачки данных АЦП/ЦАП

*long ZGetMaxInterruptADC(long typeDevice, long numberDSP, long * size)* *(Zadc.dll)*
*long ZGetMaxInterruptDAC(long typeDevice, long numberDSP, long * size)*
опрос размера буфера для перекачки данных за одно прерывание
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.

12.12.4 Установка размера буфера для перекачки данных АЦП/ЦАП

long ZSetInterruptADC(long typeDevice, long numberDSP, long size) *(Zadc.dll)*
long ZSetInterruptDAC(long typeDevice, long numberDSP, long size)
установка размера буфера памяти сигнального процессора для перекачки данных из памяти сигнального процессора в память компьютера во время каждого прерывания. 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

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

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.

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.

12.12.5 Установка размера буфера памяти накопления АЦП/ЦАП

long ZSetBufferSizeADC(long typeDevice, long numberDSP, long size) (Zadc.dll)
long ZSetBufferSizeDAC(long typeDevice, long numberDSP, long size)

устанавливает размер буфера накопления в памяти процессора. 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.

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:

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.

12.12.6 Отображение буфера памяти накопления АЦП/ЦАП

*long ZGetBufferADC(long typeDevice, long numberDSP, void **buffer, long *size) (Zadc.dll)*
*long ZGetBufferDAC(long typeDevice, long numberDSP, void **buffer, long *size)*

Отображение буфера данных АЦП/ЦАП, опрос адреса и размера буфера данных АЦП/ЦАП. 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()).*

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:
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.

12.12.7 Освобождение буфера памяти накопления АЦП/ЦАП

*long ZRemBufferADC(long typeDevice, long numberDSP, void **buffer)* *(Zadc.dll)*
*long ZRemBufferDAC(long typeDevice, long numberDSP, void **buffer)*

Освобождает дескрипторы отображения памяти для доступа к буферу данных драйвера. Эта команда подается в конце программы для того, чтобы освободить ресурсы операционной системы. Операционная система может поддерживать открытый доступ для ~ 2000 буферов.

typeDevice – тип устройства.

numberDSP – номер сигнального процессора (значение от 0 до 136),
buffer – адрес буфера в памяти процессора.
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.

12.12.8 Установка циклического или одноразового накопления АЦП/ЦАП

long ZSetCycleSampleADC(long typeDevice, long numberDSP, long enable) *(Zadc.dll)*
long ZSetCycleSampleDAC(long typeDevice, long numberDSP, long enable)

установка циклического или одноразового накопления в буфер данных центрального процессора.

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.

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

12.12.9Пересылка последних накопленных данных АЦП

*long ZGetLastDataADC(long typeDevice, long numberDSP, long numberChannel, void *buffer, long size)* *(Zadc.dll)*

пересылает последние полученные отсчеты по заданному каналу в буфер данных пользователя.

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.

12.12.10 Опрос указателя накопления в буфер данных АЦП/ЦАП

*long ZGetPointerADC(long typeDevice, long numberDSP, long *pointer)* *(Zadc.dll)*
*long ZGetPointerDAC(long typeDevice, long numberDSP, long *pointer)*
опрос указателя накопления в буфер данных центрального процессора.

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.

Функция не меняет признак *modify* в драйвере.

12.12.11 Опрос флага прерываний

*long ZGetFlag (long typeDevice, long numberDSP, unsigned long *flag)* *(Zadc.dll)*
Функция опрашивает флаг прерываний. 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.

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.

12.12.12 Опрос состояния накопления данных АЦП/ЦАП

*long ZGetStartADC(long typeDevice, long numberDSP, long *status)* *(Zadc.dll)*
*long ZGetStartDAC(long typeDevice, long numberDSP, long *status)*

Функция проверяет состояние сигнального процессора. Эта функция полезна при однократном накоплении данных.

typeDevice – тип устройства.

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

12.12.13 Старт накопления данных АЦП/ЦАП

long ZStartADC(long typeDevice, long numberDSP) *(Zadc.dll)*
long ZStartDAC(long typeDevice, long numberDSP)
Функция запускает процесс накопления данных в память буфера накопления данных.
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.

12.12.14 Останов накопления данных АЦП/ЦАП

long ZStopADC(long typeDevice, long numberDSP) *(Zadc.dll)*
long ZStopDAC(long typeDevice, long numberDSP)
Функция останавливает процесс накопления данных АЦП/ЦАП.
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.

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

12.13 Управление модулем НСР

12.13.1 Опрос поддержки и подключения модуля НСР

*long ZFindHCPADC(long typeDevice, long numberDSP, long *present)* *(Zadc.dll)*

Функция опрашивает поддерживается ли модуль НСР для питания датчиков стандарта ICP.

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:

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.

12.13.2 Опрос режима работы заданного канала модуля НСР

*long ZGetHCPADC(long typeDevice, long numberDSP, long numberChannel long *enable)* *(Zadc.dll)*

Функция опрашивает, включено ли питание постоянным током по заданному каналу модуля НСР.

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 – 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:

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.

12.13.3 Установка режима работы заданного канала модуля НСР

long ZSetHCPADC(long typeDevice, long numberDSP, long numberChannel, long enable) *(Zadc.dll)*

Функция включает-выключает питание постоянным током по заданному каналу модуля НСР.

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,
0xFF00 – для данного типа устройства функция не поддерживается (не поддерживается НСР).
Функция меняет признак *modify* в драйвере.

12.14 Управление цифровым портом

12.14.1 Опрос маски выходов цифрового порта

long ZGetDigOutEnable(long typeDevice, long numberDSP,
*unsigned long *digitalOutEnableMask)* *(Zadc.dll)*
Функция опрашивает битовую маску разрешения работы на выход входов-выходов цифрового порта.
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.

12.14.2 Установка маски выходов цифрового порта

long ZSetDigOutEnable(long typeDevice, long numberDSP,
unsigned long digitalOutEnableMask) *(Zadc.dll)*
Функция устанавливает входы-выходы цифрового порта в режим входа согласно битовой маске.
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.

12.14.3 Чтение данных с входов цифрового порта

*long ZGetDigInput(long typeDevice, long numberDSP, unsigned long *digitalInput)* *(Zadc.dll)*
Функция производит чтение цифрового порта.
typeDevice – тип устройства.

ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

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.

12.14.4 Чтение данных, выдаваемых на выходы цифрового порта

*long ZGetDigOutput(long typeDevice, long numberDSP, unsigned long *digitalOutput) (Zadc.dll)*
Функция считывает данные, которые выдаются на выходы цифрового порта.
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 не меняет признак *modify* в драйвере.

12.14.5 Запись данных в цифровой порт

long ZSetDigOutput(long typeDevice, long numberDSP, unsigned long digitalOutput) (Zadc.dll)
Функция записывает данные по выходам цифрового порта.
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 не меняет признак *modify* в драйвере.

13 Описание примеров программирования

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

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

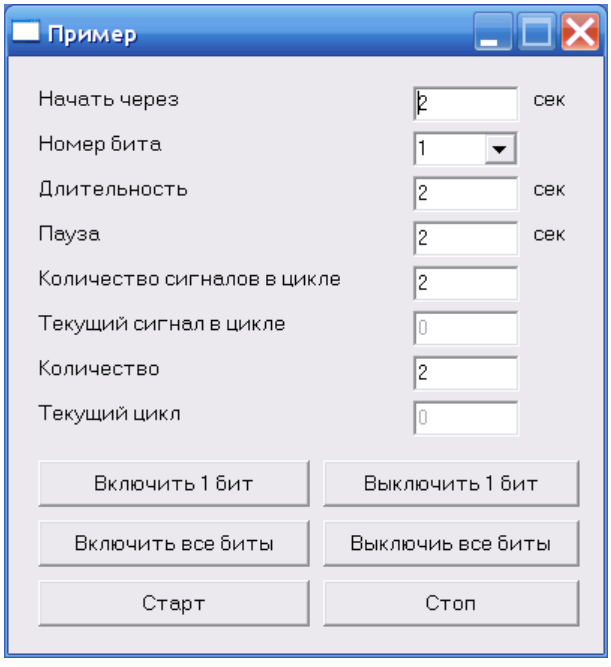
Программа – простой цифровой автомат. В зависимости от уровня входного сигнала включается и выключается заданный выходной бит по цифровому порту. В программе выбирается входной аналоговый канал, устанавливаются верхний и нижний предел срабатывания, выбирается выходной цифровой бит, по которому выполняется 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.



Sample_XIV

Программа суммирования сигналов. Программа создает виртуальный канал суммы или разности двух сигналов. Программа создана в среде VB6.0.

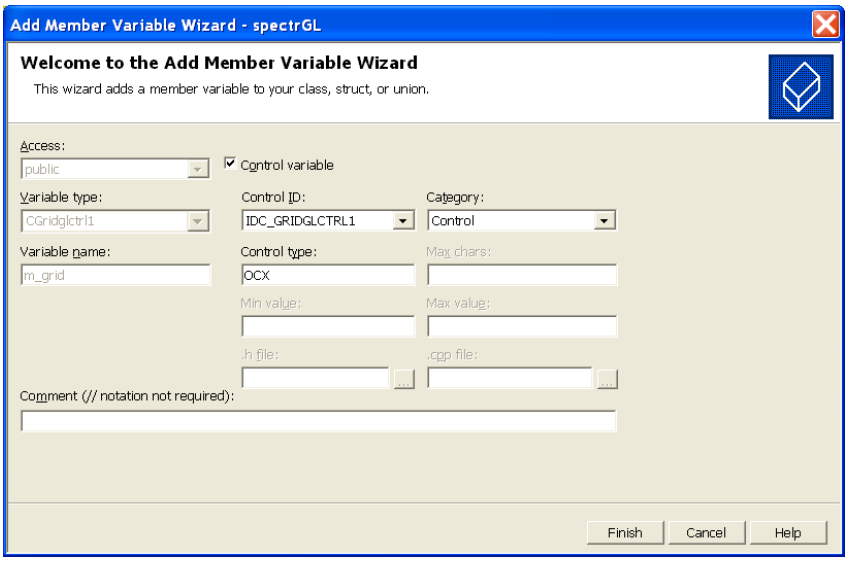
Sample_XV

Программа коррекции уровня сигнала. Эта программа предназначена для снятия показаний с датчиков с нелинейной характеристикой, например, тензометрических датчиков, датчиков давления, температуры.

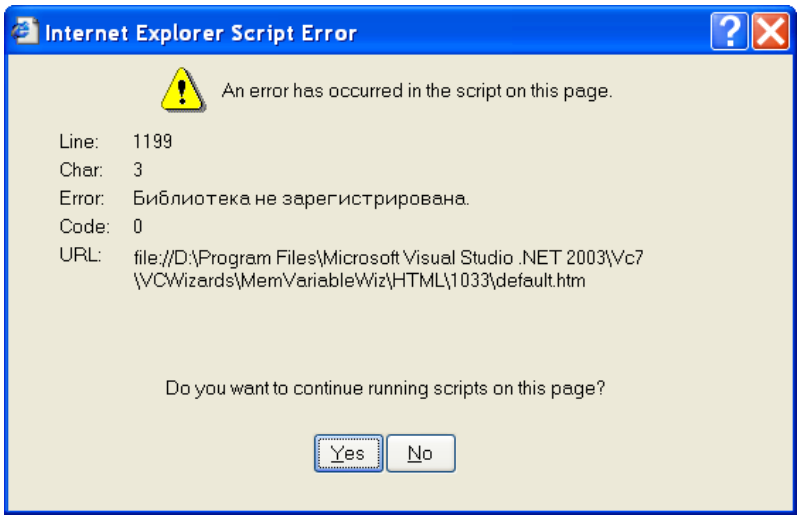
ЗАО «Электронные технологии и метрологические системы»
тел./факс: (495) 739-39-19, www.zetlab.ru

14 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:



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.