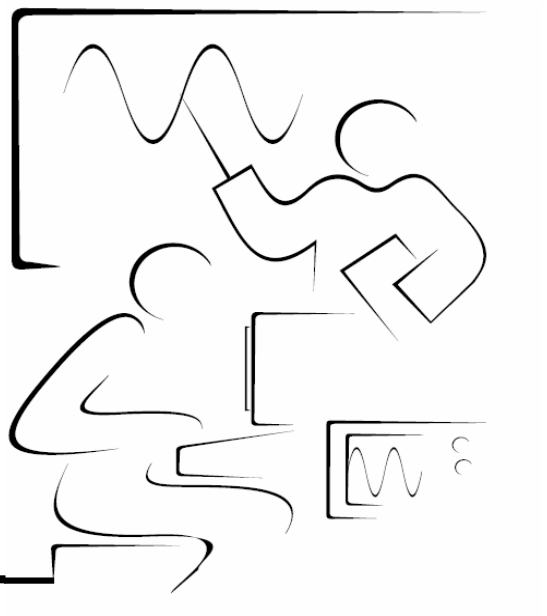
Учебный курс

LabVIEW™ Основы I



Программное обеспечение: LabVIEW 8.2

Издание: Март 2007

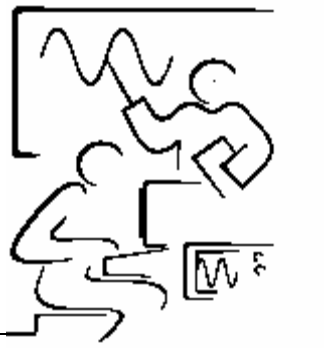
Авторский коллектив
Михеев П.М., Крылова С.И., Лукьянченко В.А., Урюпина Д.С.
Учебно-технический центр "Системы автоматизации научных исследований"
Международный учебно-научный лазерный центр
Московский государственный университет им. М.В. Ломоносова
Сайт: http://labview.ilc.edu.ru

Техническая поддержка и информация о продуктах
ni.com

Штаб-квартира корпорации National Instruments 11500 North Mopac Expressway Austin, Texas 78759-3504 USA Tel: 512 683 0100

Российское представительство корпорации National Instruments: 495 7836851

Оглавление



Рекомендации слушателю курса

Благодарим Вас за приобретение комплекта учебного курса LabVIEW Основы I. После выполнения упражнений этого учебника Вы сможете самостоятельно разрабатывать собственные программные приложения. Руководство курса и программное обеспечение, сопровождающее курс, используются в трехдневном учебном курсе LabVIEW Основы I.

A. Об учебнике

Этот учебник показывает, как использовать среду LabVIEW при разработке приложений тестирования, измерения и сбора данных, управления измерительными приборами, архивирования данных, анализа данных измерений и генерации отчета. Материал, изложенный в этом учебнике, предполагает наличие у слушателя знаний об одной из перечисленных операционных систем - Windows, MacOS или UNIX, а также наличие практических навыков разработки алгоритмов в виде блок-схем или блок-диаграмм.

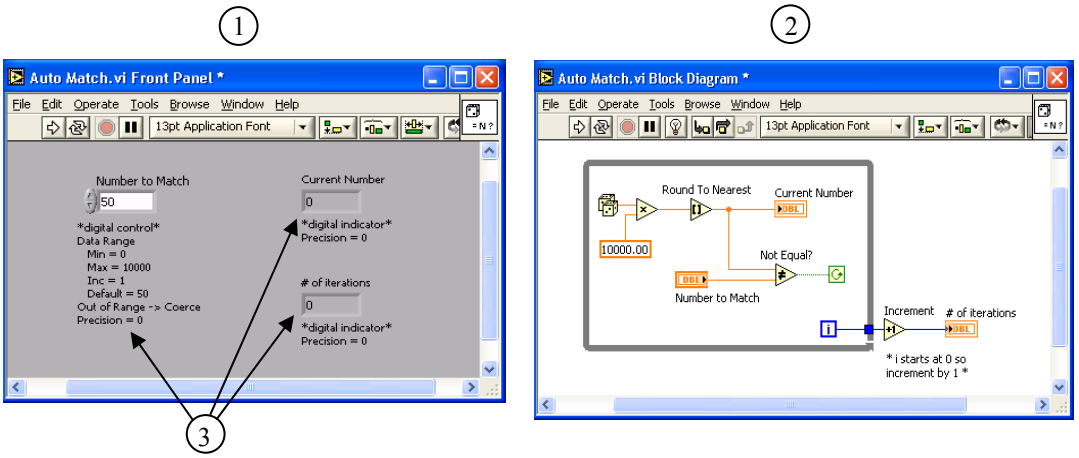
Учебный курс разделен на уроки, каждый из которых описывает одну или несколько тем. Каждый урок содержит следующую информацию:

- Описание темы и цели урока;
- Описание основных разделов урока;
- Набор упражнений, закрепляющих знания, полученные в уроке;
- Набор дополнительных упражнений;
- Краткое изложение пройденного материала, советы и секреты.

Некоторые упражнения учебного курса требуют использования аппаратных средств фирмы National Instruments. К ним относятся многофункциональные устройства сбора данных (DAQ) с подключенной панелью сигналов с температурным датчиком, генератором сигнала с индикаторами-светодиодами и плата GPIB интерфейса с подключенным симулятором измерительного прибора (NI Instrument Simulator).

При выполнении большинства упражнений, в случае отсутствия необходимых аппаратных средств, допустимо использование демонстрационных версий ВП. Но упражнения, обозначенные иконкой, показанной слева, требуют присутствия аппаратных средств. Необходимо отметить, что возможна замена требуемых для упражнения аппаратных средств на эквивалентные. Например, можно использовать GPIB измерительный прибор вместо симулятора измерительного прибора (NI Instrument Simulator) или другие DAQ устройства, соединенные с источником сигналов, таких как генератор сигналов.

В каждом упражнении показан окончательный вид лицевой панели и блок-диаграммы перед запуском ВП на выполнение. Пример лицевой панели и блок-диаграммы показан на рисунке. После примера блок-диаграммы следует описание каждого объекта приведенного на ней.



1 лицевая панель	2 блок-диаграмма	3 комментарии

B. Что необходимо для начала занятий

Для выполнения учебного курса понадобятся:

- **(Windows)** Windows 98 или более поздние версии, инсталлированные на компьютере; **(MacOS)** Power Macintosh под управлением MacOS 7.6.1 или более поздние версии; **(UNIX)** Sun workstation под управлением Solaris 2.5 или более поздние версии и системное программное обеспечение XWindows или PC компьютер под управлением Linux 2.0.*x* или более поздние версии для архитектуры Intel *x*86;.
- **(Windows)** Многофункциональное DAQ устройств, сконфигурированное утилитой Measurement & Automation Explorer как устройство 1 (device 1); **(MacOS)** Многофункциональное DAQ устройство, размещенное в слоте 1 (Slot 1);
- Дополнительные принадлежности к DAQ устройству, соединительные провода и кабели;
- **(Windows and UNIX)** GPIB интерфейс; **(MacOS)** GPIB интерфейс в слоте 2 (Slot 2);
- Симулятор измерительного прибора (NI Instrument Simulator) и источник питания;
- LabVIEW Full или Professional Development System 8.2 или более поздняя версия;
- Нуль-модемный кабель;
- GPIB кабель;
- (Дополнительно) Текстовый редактор, к примеру **(Windows)** Notepad, WordPad; **(MacOS)** SimpleText; **(UNIX)** Text Editor, vi, или vuepad;
- Компакт-диск с учебным курсом LabVIEW Основы I, включающий следующие файлы:
-

Имя файла	Описание
`Exercises`	Каталог для сохранения ВП, созданных в процессе выполнения упражнений учебного курса; также включает подпрограммы ВП, необходимые для ряда упражнений
`nidevsim.zip`	Zip файл, включающий LabVIEW-драйвер для симулятора измерительного прибора (NI Instrument Simulator)
`bas1soln.exe`	Самораспаковывающийся архив, включающий решения всех упражнений учебного курса

Примечание. При решении некоторых упражнения вместо **ВП Термометр (Thermometer VI)** используется ВП **(Demo) Thermometer VI** из каталога решений.

C. Установка программного обеспечения учебного курса

Выполните следующие шаги для установки программного обеспечения учебного курса LabVIEW Основы I.

Windows

1. Скопировать содержимое папки **nidevsim** в каталог **labview\instr.lib**. После запуска LabVIEW **NI Devsim** драйвер будет размещен на палитре **Функций** в разделе **Functions»Instrument I/O»Instrument Drivers.**
2. Скопировать каталог Exercises в корневую директорию диска **c:**.
3. Скопировать каталог Solutions в корневую директорию диска **c:**.

D. Цели учебного курса и вопросы, которые курс не рассматривает

Целью учебного курса является обучение:

- использованию среды LabVIEW для создания приложений;
- технике редактирования и отладки приложений;
- пониманию назначения лицевой панели, блок-диаграммы, соединительных панелей и иконок;
- использованию встроенных подпрограмм ВП и функций;
- созданию и сохранению ВП, для его последующего использования в качестве подпрограммы ВП;
- созданию приложений, использующих последовательную связь и GPIB-интерфейс;
- созданию приложений, использующих встроенные DAQ устройства;

Этот учебный курс не рассматривает следующие вопросы:

- теория программирования;
- использование каждой встроенной подпрограммы ВП и функций;
- команды и операции GPIB-интерфейса;
- команды и операции последовательного COM порта;
- теория аналого-цифрового преобразования;
- разработка драйверов устройств;
- создание завершенных приложений.

E. Условные обозначения

В учебном курсе использованы следующие условные обозначения:

» Символ " » " обозначает последовательность вызова пунктов вложенных меню или разделов диалоговых окон. Например, последовательность **File»Page Setup»Options** указывает, что сначала необходимо выбрать меню **File**, затем пункт меню **Page Setup** и далее раздел **Options** в диалоговом окне.

 Иконка обозначает дополнительную справочную информацию.

 Иконка обозначает важную информацию для запоминания.

 Иконка обозначает предостережение от действий, которые могут привести к потере данных или сбоям операционной системы.

 Иконка обозначает то, что выполняемое упражнение требует наличия встроенных аппаратных средств: GPIB-интерфейса или DAQ устройства.

Жирный текст Жирный текст выделяет названия пунктов меню или разделов диалоговых окон и палитр, названия используемых палитр, названия и типы данных элементов на лицевой панели или объектов на блок-диаграмме, численные значения, а также названия используемых ВП, функций и аппаратных средств.

Курсив Курсивом обозначены ссылки на дополнительную информацию.

Ариель Текст в этом шрифте обозначает текст или символы, секции исходных кодов и синтаксис управляющих команд, примеры программирования. Этот шрифт также используется для выделения текста, вводимого с клавиатуры.

(MacOS) Жирный текст в скобках, указывает на тип операционной системы, в которой используется дальше указанные действия или операции.

Щелчок **(Windows)** щелкнуть правой кнопкой мыши для выполнения последующих действий, **(MacOS)** Нажать кнопку <Command>

<Enter> **(Macos)** Нажать кнопку <Return>

<Ctrl> **(Macos)** нажать кнопку <Command>, **(Sun)** нажать кнопку <Meta>,**(Linux)** нажать кнопку <Alt>.

Урок 1.
Введение в LabVIEW

В этом уроке представлены основы программной среды LabVIEW.
Чтобы быстро начать работу с LabVIEW, получить основные сведения о технике редактирования и отладки, типах палитр, используемых LabVIEW, «горячих» клавишах и Web-ресурсах – необходимо обратиться к справочнику ***LabVIEW Quick Reference Card.***
Для просмотра PDF версии справочника, необходимо в пункте главного меню **Помощь** выбрать раздел книги LabVIEW (**Help»Search the LabVIEW Bookshelf)**. Далее нажать клавишу **<Page Down>** и выбрать **LabVIEW Quick Reference Card.**

В этом уроке изложены вопросы:

A. Программная среда LabVIEW.

B. Что такое виртуальный прибор (ВП).

C. Последовательность обработки данных.

D. Организация программной среды LabVIEW (окна, меню, инструменты).

E. Использование проектов в LabVIEW.

F. Справочная система среды LabVIEW и руководство пользователя.

A. Программная среда LabVIEW

Программа, написанная в среде LabVIEW, называется виртуальным прибором (ВП). ВП симулируют реальные физические приборы, например, осциллограф или мультиметр. LabVIEW содержит полный набор инструментов для сбора, анализа, представления и хранения данных.

В LabVIEW интерфейс пользователя — лицевая панель создается с помощью элементов управления (кнопки, переключатели и др.) и отображения (графики, светодиоды и др.). После этого на блок-диаграмме ВП осуществляется программирование с использованием графических представлений функций для управления объектами на лицевой панели.

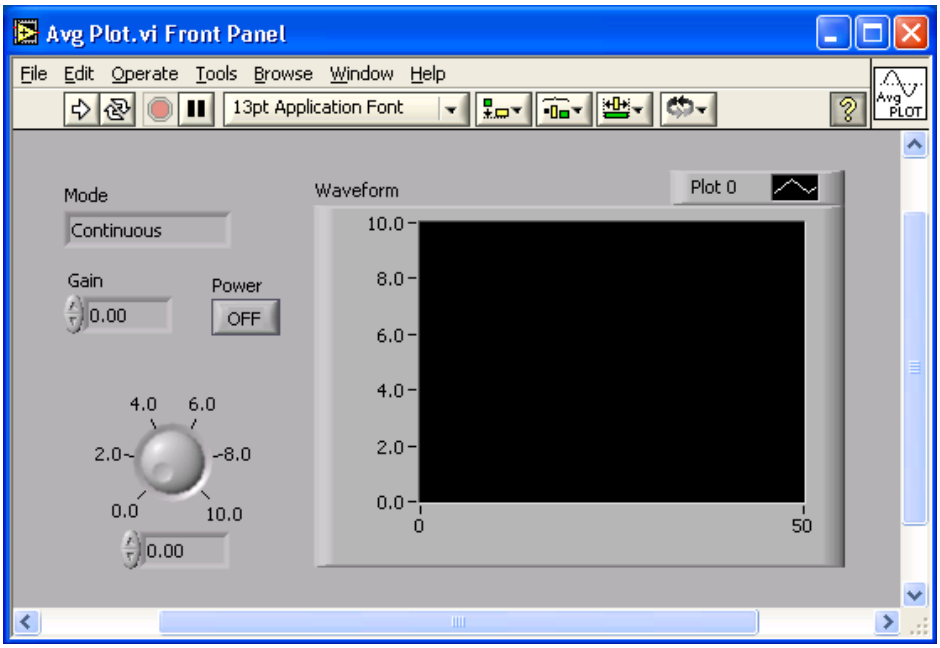
LabVIEW используется для программирования различных **DAQ**-устройств, систем контроля изображения и движения, аппаратных средств, имеющих интерфейсы типа **GPIB**, **VXI**, **PXI**, **RS-232** и **RS-485**. LabVIEW имеет встроенные возможности для работы в компьютерных сетях Интернет, используя LabVIEW Web Server и программные стандарты TCP/IP и Active X.

С помощью программной среды LabVIEW можно разрабатывать программно-аппаратные комплексы для тестирования, измерения, ввода данных, анализа и управления внешним оборудованием. LabVIEW – это 32-х разрядный компилятор, который создает как автономные модули (**.EXE**), так и совместно используемые динамические библиотеки (**.DLL**).

B. Виртуальные приборы (ВП)

ВП состоит из четырех основных компонентов – лицевой панели, блок-диаграммы, иконки и соединительной панели.

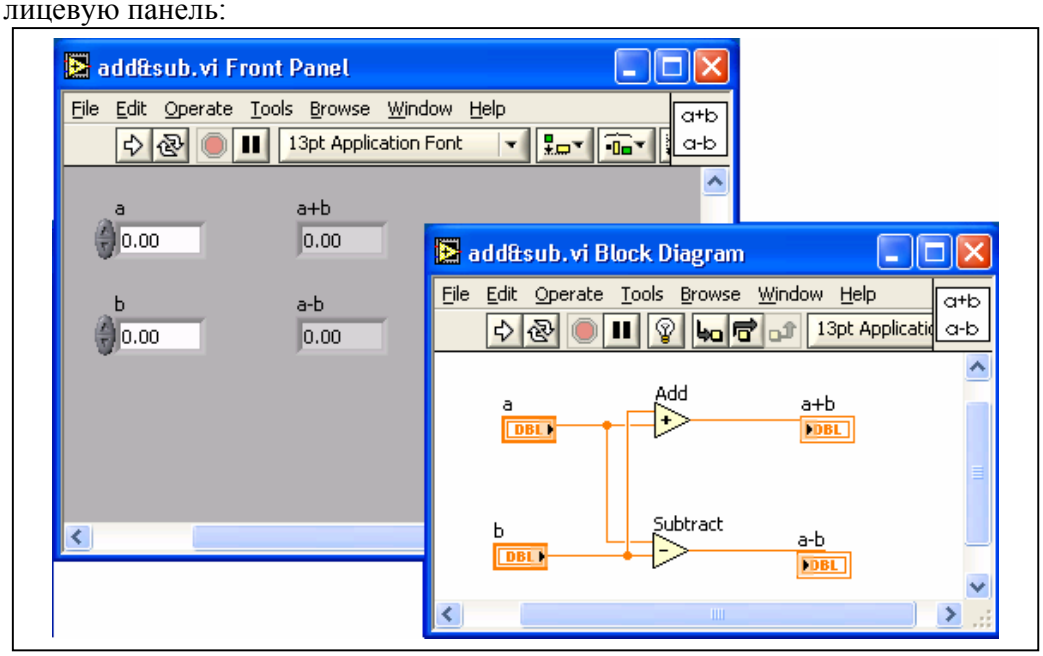
Лицевая панель – это интерфейс пользователя ВП. Пример лицевой панели представлен ниже.



Лицевая панель создается с использованием палитры **Элементов** (**Controls**). Эти элементы могут быть либо средствами ввода данных – элементы **Управления**, либо средствами отображения данных – элементы **Отображения**. Элементы **Управления** – кнопки, переключатели, ползунки и другие элементы ввода. Элементы **Отображения** – графики, цифровые табло, светодиоды и т.д. Данные, вводимые на лицевой панели ВП, поступают на блок-диаграмму, где ВП производит с ними необходимые операции. Результат вычислений передается на элементы отображения информации на лицевой панели ВП.

После помещения элементов **Управления** или **Отображения** данных на Лицевую панель, они получают свое графическое отображение на блок-диаграмме. Объекты блок-диаграммы включают графическое отображение элементов лицевой панели, операторов, функций, подпрограмм ВП, констант, структур и проводников данных, по которым производится передача данных между объектами блок-диаграммы.

Следующий пример показывает блок-диаграмму и соответствующую ей лицевую панель:



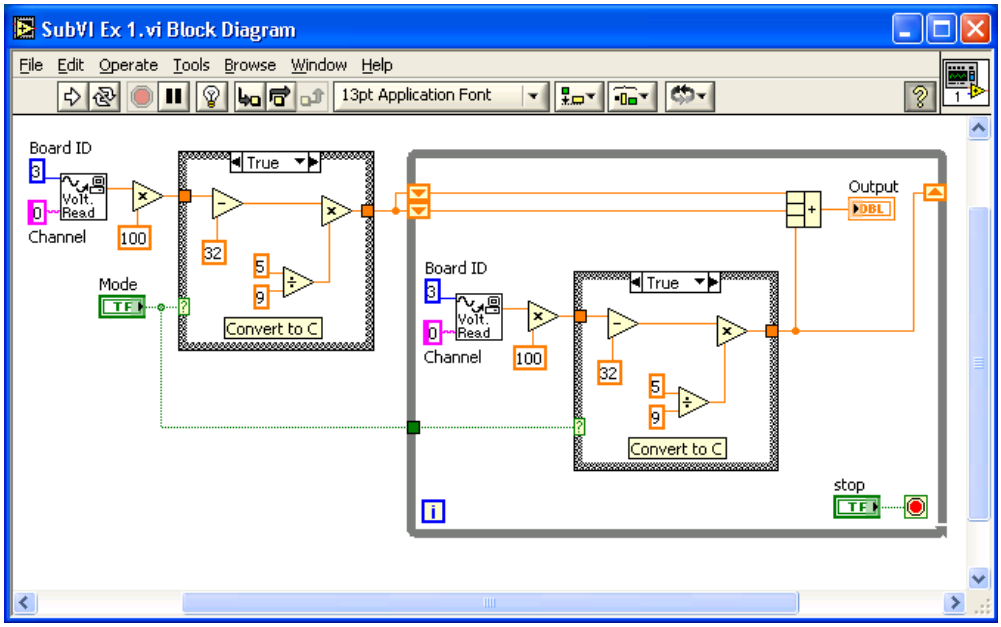
Для использования созданного виртуального прибора внутри другого ВП в качестве подпрограммы, после создания лицевой панели и блок-диаграммы, необходимо оформить иконку и настроить соединительную панель (область полей ввода/вывода данных). Подпрограмма ВП соответствует подпрограмме в текстовых языках программирования. Каждый ВП имеет показанную слева иконку в верхнем правом углу лицевой панели и блок-диаграммы. Иконка – графическое представление ВП. Она может содержать текст и/или рисунок. Если ВП используется в качестве подпрограммы, иконка идентифицирует его на блок-диаграмме другого ВП.

Необходимо также настроить показанную слева соединительную панель (область полей ввода/вывода данных), чтобы использовать ВП в качестве подпрограммы. Соединительная панель – это набор полей, соответствующий элементам ввода/вывода данных этого ВП. Поля ввода/вывода аналогичны списку параметров вызываемой функции в текстовых языках программирования. Область полей ввода/вывода данных позволяет использовать ВП в качестве подпрограммы. ВП получает данные через поля ввода данных и передает их на блок-диаграмму через элементы **Управления** лицевой панели. Результаты отображаются в его полях вывода данных посредством элементов **Отображения** лицевой панели.

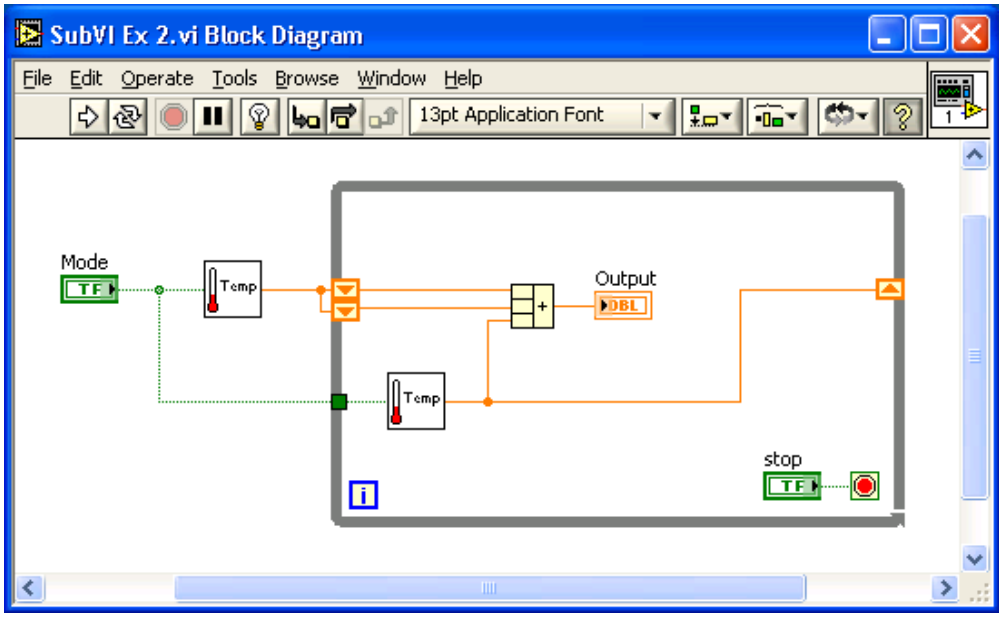
Преимущество LabVIEW заключается в иерархической структуре ВП. Созданный виртуальный прибор можно использовать в качестве

подпрограммы на блок-диаграмме ВП более высокого уровня. Количество уровней в иерархии не ограничено. Использование подпрограммы ВП помогает быстро изменять и отлаживать блок-диаграмму.

При создании ВП следует обратить внимание на то, что некоторые операции многократно повторяются. Для выполнения таких операций необходимо использовать подпрограммы ВП или циклы. Урок 4 *Многократные повторения и циклы* содержит дополнительную информацию об использовании циклов. Например, приведенная ниже блок-диаграмма содержит две идентичные операции.



Можно создать подпрограмму ВП, которая выполнит эту операцию, и можно вызвать эту подпрограмму дважды. Возможно многократное использование подпрограммы ВП в другом виртуальном приборе. Следующий пример демонстрирует использование **Temperature VI** в качестве подпрограммы на блок-диаграмме.



Автосохранение

При неправильном выключении или при повреждении системы LabVIEW производит сохранение во временный файл открытых файлов с расширениями *(.vi), (.vit), (.ctl), (.ctt)*. LabVIEW ***не сохраняет*** проекты *(.lvproj),* библиотеки проектов *(.lvlib)*, XControls *(.xctl)*, или классы *(.lvclass)*.

Если LabVIEW удалось сохранить файлы поеред тем, как произошло незапланированное выключение или ошибка системы, то при следующем запуске LabVIEW появится окно **Select Files to Recover**. Выберите файлы, которые вы хотите перезаписать, и нажмите кнопку **Recover**. Если Вы не хотите перезаписывать файлы, то, ничего не выбирая, нажмите кнопку **Discard**. При нажатии кнопки **Cancel** выбранные файлы будут помещены в папку *LVAutoSave\archives*.

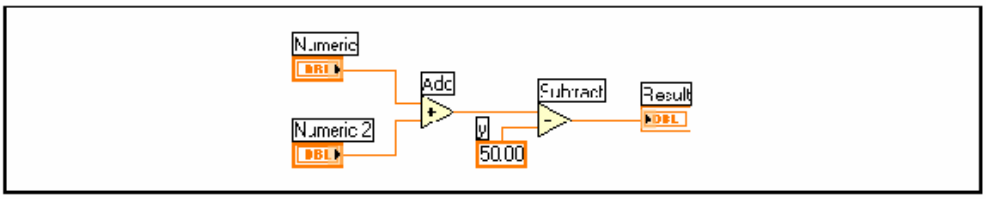
Чтобы настроить функцию автосохранения в LabVIEW, выберите на линейке инструментов в верхней части окна пункт **Tools»Options**, затем их списка **Category**, находящегося в левой части окна выберите пункт **Environment**. После этого Вы сможете включить или отключить функцию автосохранения, а также установить интервал времени, через который происходит автосохранение.

C. Последовательность обработки данных

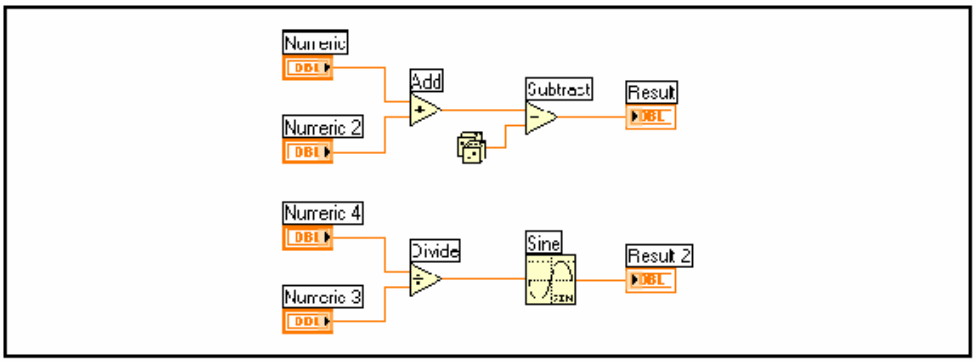
В Visual Basic, C++, Java и большинстве других текстовых языков программирования порядок выполнения всей программы определяется расположением функций программы.

В среде LabVIEW используется потоковая модель обработки данных. Узлы блок-диаграммы выполняют заложенные в них функции, если данные поступили на все необходимые поля ввода/вывода. По окончании выполнения операции одним узлом результаты операции по проводникам данных передаются следующему узлу и т.д. Другими словами, готовность входных данных определяет последовательность выполнения узлов блок-диаграммы.

В качестве примера можно рассмотреть блок-диаграмму, которая складывает два числа и затем вычитает из получившейся суммы «50.0». В этом случае блок-диаграмма выполняется слева направо не потому, что объекты помещены в этом порядке, а потому, что одно из полей ввода функции **Subtract** (Вычитание) не определено, пока не выполнилась функция **Add** (Сложение) и не передала данные к функции **Subtract** (Вычитание). Не следует забывать, что узел выполняется только тогда, когда определены его поля ввода данных.

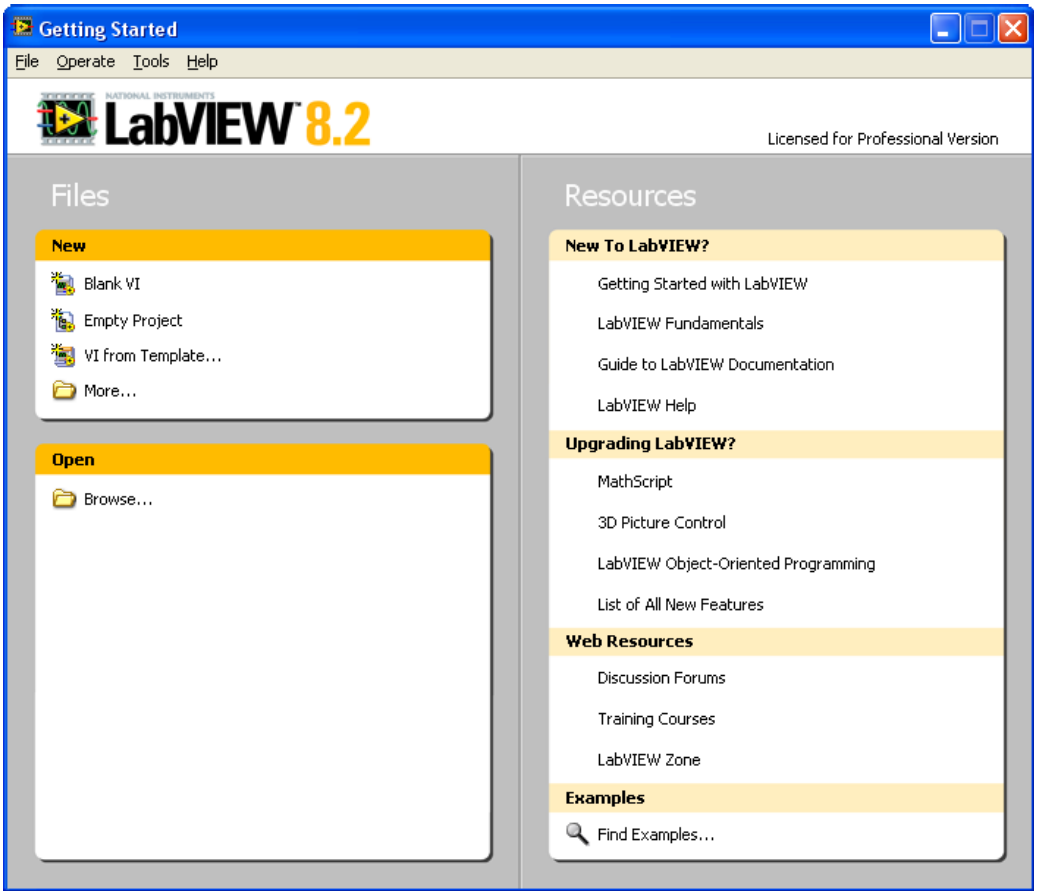


В следующем примере рассмотрена последовательность выполнения функций **Add** (Сложение), **Random Number** (Генератор случайных чисел)и **Divide** (Деление). Как видно, в данном случае последовательность выполнения функций не определена, так как поля ввода данных функций **Add** (Сложение) и **Divide** (Деление) инициализируются одновременно, а **Random Number** (Генератор случайных чисел) не имеет полей ввода данных. Когда необходимо выполнить одну часть кода блок-диаграммы раньше другой и нет зависимости данных между функциями, для установки порядка выполнения следует использовать методы программирования.



D. Организация программной среды LabVIEW

При запуске LabVIEW появляется окно запуска для работы с системой, с помощью которого можно создать новый ВП, проект, открыть уже существующий ВП, найти примеры или получить доступ к подсказке *LabVIEW Help*. Окно запуска появляется также при закрытии всех лицевых панелей и блок-диаграмм.



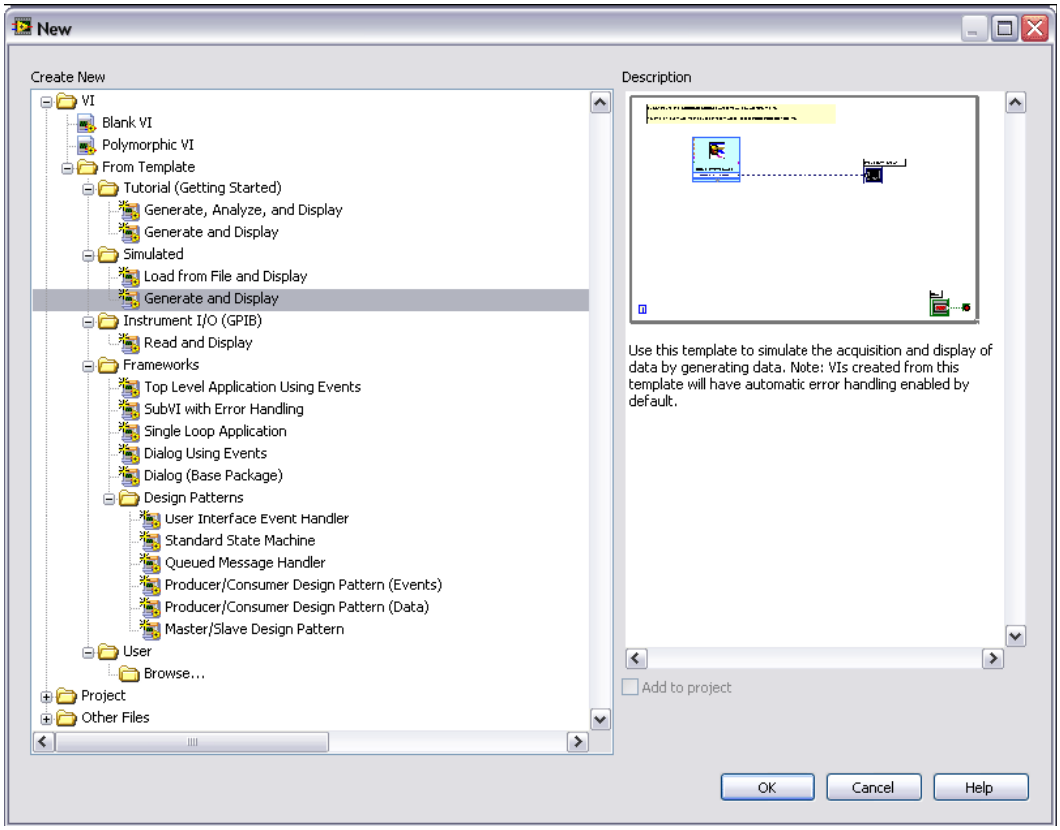
Окно запуска содержит следующие компоненты:

- Панель меню со стандартными пунктами, например, **File»Exit**
- Окно **Files**, позволяющее открыть или создать ВП. В данном окне находятся вкладки **New** и **Open**. Используя вкладку **New,** Вы можете создать новый ВП, новый проект или загрузить шаблон ВП. Используя вкладку **Open**, можно открыть созданный ранее ВП.
- Окно **Resources** позволяет обратиться за помощью или дополнительной информацией к различным текстовым и Internet ресурсам, а также посмотреть встроенные примеры.

Создание нового ВП или открытие уже существующего.

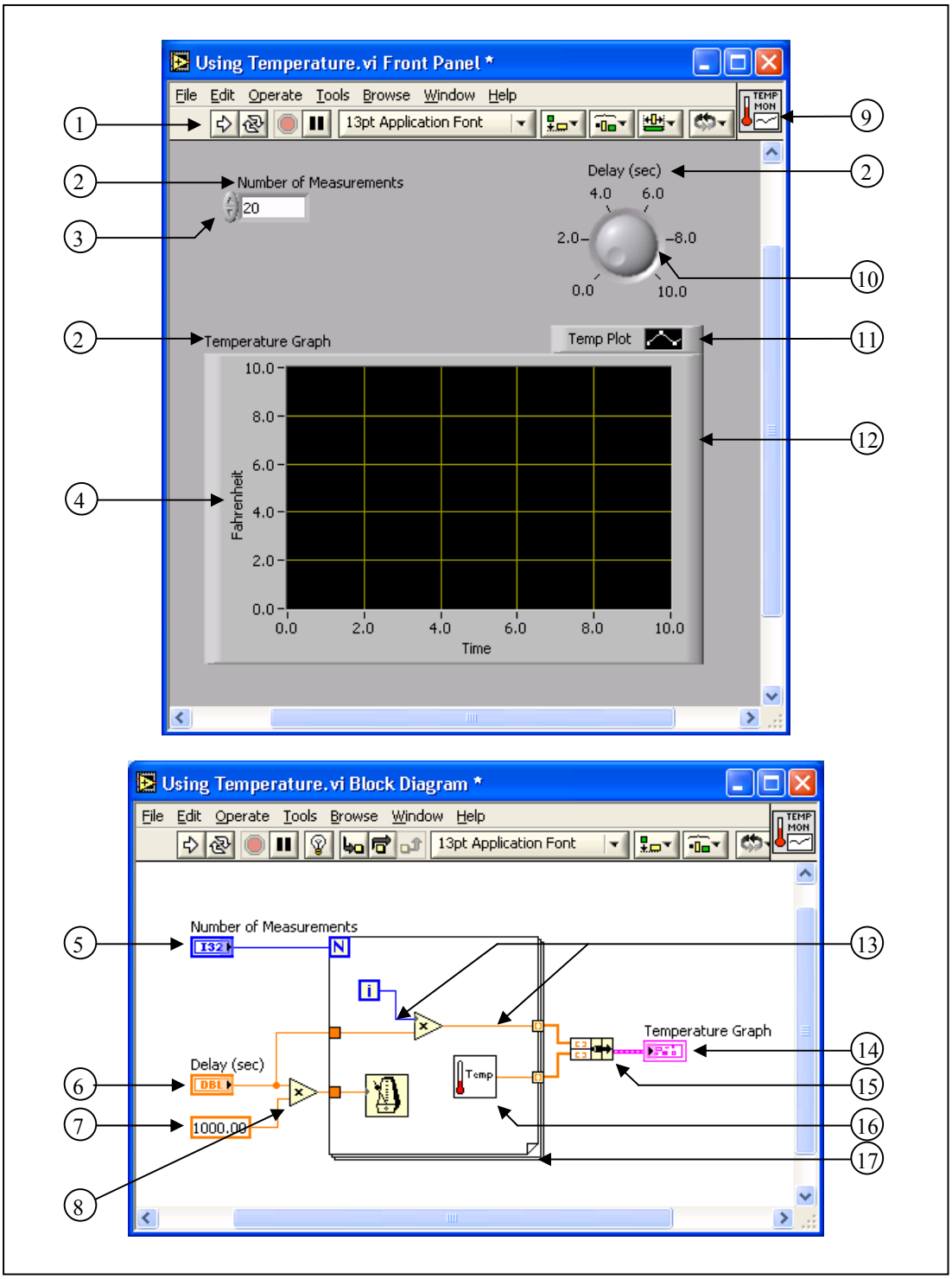
После нажатия на строку **Blank VI** во вкладке **New**, на экране появится лицевая панель и блок-диаграмма пустого ВП.

Чтобы открыть шаблон ВП, нажмите левой кнопкой мыши во вкладке **New** на строку **VI from Template…** После этого на экране появится диалоговое окно **New**. После выбора шаблона из спискового окна **Create New** (создание нового ВП), в секции **Description** отобразится описание шаблона и блок-диаграмма ВП**.** Для создания ВП нажмите кнопку **OK.**



Лицевая панель и окно блок-диаграммы

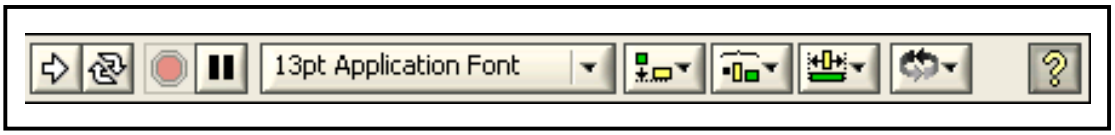
После нажатия кнопки **Blank VI** появляется окно лицевой панели. Это одно из двух окон LabVIEW, используемых для создания ВП. Другое окно содержит блок-диаграмму. На задний план лицевой панели виртуального прибора Вы можете импортировать какое-либо изображение. Для этого нажмите правой кнопкой мыши на полосе прокрутки лицевой панели и из контекстного меню выберите **Properties**. Затем, диалоговом окне **Pane Properties** выберите изображение из списка **Background**. LabVIEW поддерживает форматы BMP, JPEG и PNG. Следующая иллюстрация демонстрирует лицевую панель и соответствующую ей блок-диаграмму:



1. Инструментальная панель
2. Собственная метка
3. Цифровой элемент управления
4. Свободная метка
5. Терминал данных цифрового элемента управления
6. Терминал данных кнопки
7. Числовая константа
8. Функция Умножение
9. Иконка
10. Ручка управления
11. Панель управления графиком
12. Двукоординатный график осциллограмм
13. Проводники данных
14. Терминал данных графика осциллограмм
15. Функция Объединение (Bundle)
16. Подпрограмма ВП
17. Цикл с фиксированным числом итераций (For)

Инструментальная панель лицевой панели

Инструментальная панель используется для запуска и редактирования ВП. Пример инструментальной панели показан ниже.



Кнопка запуска **Run –** запускает ВП

Во время работы ВП кнопка **Run** меняет свой вид, как показано слева, если этот виртуальный прибор высокого уровня.

Если ВП работает в качестве подпрограммы, то кнопка **Run** выглядит, как показано слева.

Кнопка **Run** выглядит в виде «сломанной» стрелки, как показано слева, во время создания или редактирования ВП. В таком виде кнопка показывает, что ВП не может быть запущен на выполнение. После нажатия этой кнопки появляется окно **Error list,** в котором перечислены допущенные ошибки.

Кнопка непрерывного запуска **Run Continuously** – ВП выполняется до момента принудительной остановки.

Во время выполнения ВП появляется кнопка **Abort Execution**. Эта кнопка используется для немедленной остановки выполнения ВП.

Примечание. По возможности следует избегать использования кнопки **Abort Execution** для остановки ВП. Следует позволить ВП закончить передачу данных или выполнить остановку программным способом, гарантируя остановку ВП в определенном состоянии. Например, можно установить на лицевой панели кнопку, по нажатию которой ВП останавливается

Кнопка **Pause** приостанавливает выполнение ВП. После нажатия кнопки **Pause** LabVIEW подсвечивает на блок-диаграмме место остановки выполнения. Повторное нажатие – продолжение работы ВП.

Text Settings – выпадающее меню установок текста, включая размер, стиль и цвет.

В меню **Align Objects** производится выравнивание объектов по осям (по вертикали, по осям и т.д.).

В меню **Distribute Objects** производится выравнивание объектов в пространстве (промежутки, сжатие и т.д.).

В меню **Resize Objects** производится приведение к одному размеру многократно используемых объектов лицевой панели.

Меню **Reorder** используется при работе с несколькими объектами, которые накладываются друг на друга. Выделив один из объектов с помощью инструмента ПЕРЕМЕЩЕНИЕ, в меню **Reorder** следует выбрать его порядок отображения на лицевой панели.

Кнопка **Context Help** выводит на экран окно **Context Help** (контекстной справки)

Инструментальная панель блок-диаграммы

При запуске ВП на блок-диаграмме появляется показанная ниже инструментальная панель:

Кнопка **Highlight Execution** предназначена для просмотра потока данных через блок-диаграмму (режим отладки). Повторное нажатие кнопки отключает этот режим.

Кнопка **Retain Wire Values** предназначена для сохранения данных прошедших по проводникам. Включив ее, можно посмотреть значения данных в любом проводнике ВП в любой момент времени.

Кнопка **Step Into** используется при пошаговом выполнении цикла от узла к узлу, подпрограммы ВП и т.д. При этом узел мигает, обозначая готовность к выполнению.

Кнопка **Step Over** позволяет пропустить в пошаговом режиме цикл, подпрограмму и т.д.

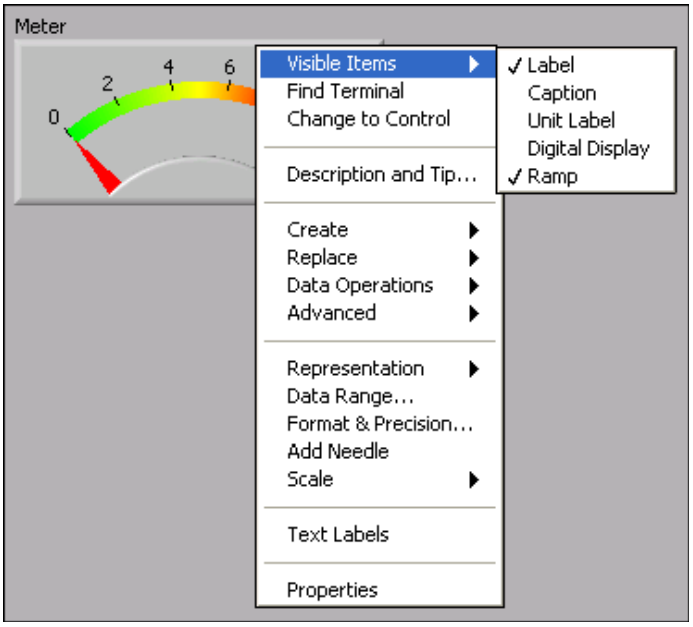
Кнопка **Step Out** позволяет выйти из цикла, подпрограммы и т.д.

Выход из узла предполагает завершение выполнения этого узла в пошаговом режиме и переход в следующий.

Кнопка **Warning** появляется, когда есть потенциальная проблема с блок-диаграммой, но она не запрещает выполнение ВП. Кнопку **Warning** можно активизировать, войдя в пункт главного меню **Инструменты**, далее – **Опции, Отладка** (**Tools»Options»Debugging)**.

Контекстное меню

Контекстное меню используется наиболее часто. Все объекты LabVIEW, свободное рабочее пространство лицевой панели и блок-диаграммы имеют свои контекстные меню. Контекстное меню используется для изменения поведения объектов блок-диаграммы и лицевой панели. Контекстное меню вызывается щелчком правой кнопкой мыши на объекте, лицевой панели или блок-диаграмме. Пример контекстного меню показан ниже.



(MacOS) Нажмите клавишу <Command> и щелкните на объекте лицевой панели или блок-диаграммы.

Главное меню

Главное меню в верхней части окна ВП содержит пункты общие с другими приложениями, такие как **Open, Save, Copy, Paste,** а также специфические пункты меню LabVIEW. Некоторые пункты главного меню содержат сведения о «горячих» клавишах вызова этих пунктов. **(MacOS)** Меню появляется в верхней части экрана.

Внимание. Во время выполнения ВП некоторые пункты главного меню недоступны.

- Пункт меню **File** используется для открытия, закрытия, сохранения и печати ВП.
- Пункт меню **Edit** используется для поиска и внесения изменений в компоненты ВП.
- Пункт меню **View** используется для отображения различных палитр, иерархии ВП и открытия различных окон, позволяющих работать в LabVIEW.
- Пункт меню **Project** используется для работы с файловой системой проекта.
- Пункт меню **Operate** используется для запуска, прерывания выполнения и изменения других опций ВП.
- Пункт меню **Tools** используется для связи с приборами и **DAQ** устройствами, сравнения ВП, формирования приложений и конфигурации LabVIEW.

- Пункт меню **Window** используется для отображения окон LabVIEW и палитр.
- Пункт меню **Help** используется для получения информации о палитрах, меню, инструментах, ВП и функциях, для получения пошаговой инструкции использования LabVIEW и информации о компьютерной памяти.

Палитры

LabVIEW имеет три вспомогательные палитры, используемые для создания и выполнения ВП: **Tools Palette** (Палитра Инструментов), **Controls Palette** (Палитра Элементов) и **Functions Palette** (Палитра Функций). Эти палитры можно поместить в любом месте экрана.

Палитра Инструментов

Создавать, редактировать и отлаживать ВП можно с помощью **Tools Palette** (Палитры Инструментов). Палитра Инструментов доступна как на лицевой панели, так и на блок-диаграмме. Термин инструмент подразумевает специальный операционный режим курсора мыши. При выборе определенного инструмента значок курсора изменяется на значок данного инструмента. Палитра Инструментов доступна через пункт главного меню **Window»Show Tools Palette**. Палитру Инструментов можно размещать в любой области рабочего пространства блок-диаграммы и лицевой панели.

Примечание. Удерживая нажатой клавишу **<Shift>** и щелкнув правой клавишей мыши, можно вывести на экран временную версию **Tools Palette** (Палитры Инструментов).

Если включен автоматический выбор инструмента, то при наведении курсора на объект лицевой панели или блок-диаграммы LabVIEW автоматически выбирает соответствующий инструмент из палитры **Tools** (Инструментов). Автоматический выбор инструментов включается нажатием на кнопку **Automatic Tool Selection** палитры **Tools** (Инструментов) или нажатием клавиш **<Shift-Tab>.**

Инструмент УПРАВЛЕНИЕ используется для изменения значения элементов управления или ввода текста. При наведении курсора на такой элемент как строковый элемент управления, значок инструмента меняется:

Инструмент ПЕРЕМЕЩЕНИЕ используется для выбора, перемещения или изменения размеров объектов. При наведении инструмента на объект изменяемого размера значок инструмента меняется:

Инструмент ВВОД ТЕКСТА используется для редактирования текста и создания свободных меток. При создании свободных меток значок инструмента меняется:

Инструмент СОЕДИНЕНИЕ создает проводники данных, соединяя объекты на блок-диаграмме.

Инструмент ВЫЗОВ КОНТЕКСТНОГО МЕНЮ вызывает контекстное меню соответствующего объекта по щелчку левой кнопки мыши.

Инструмент БЫСТРАЯ ПРОКРУТКА ЭКРАНА используется для просмотра окна без использования полосы прокрутки.

Инструмент ВВОД КОНТРОЛЬНОЙ ТОЧКИ позволяет расставлять контрольные точки на ВП, функциях, узлах, проводниках данных, структурах и приостанавливать в них выполнение программы.

Инструмент УСТАНОВКА ОТЛАДОЧНЫХ ИНДИКАТОРОВ дает возможность исследовать поток данных в проводниках блок-диаграммы. Используется для просмотра промежуточных значений при наличии сомнительных или неожиданных результатов работы ВП.

Инструмент КОПИРОВАНИЕ ЦВЕТА предназначен для копирования цвета с последующей вставкой с помощью инструмента РАСКРАШИВАНИЕ.

Инструмента РАСКРАШИВАНИЕ позволяет изменить цвет объекта. Он также отображает текущий передний план и параметры настройки цвета фона.

Если автоматический выбор инструмента выключен, можно менять инструменты палитры **Tools** (Инструментов) с помощью клавиши **<Tab>.** Для переключения между инструментом ПЕРЕМЕЩЕНИЕ и СОЕДИНЕНИЕ на блок-диаграмме или между инструментом ПЕРЕМЕЩЕНИЕ и УПРАВЛЕНИЕ на лицевой панели – достаточно нажать пробел.

Палитра Элементов и палитра Функций

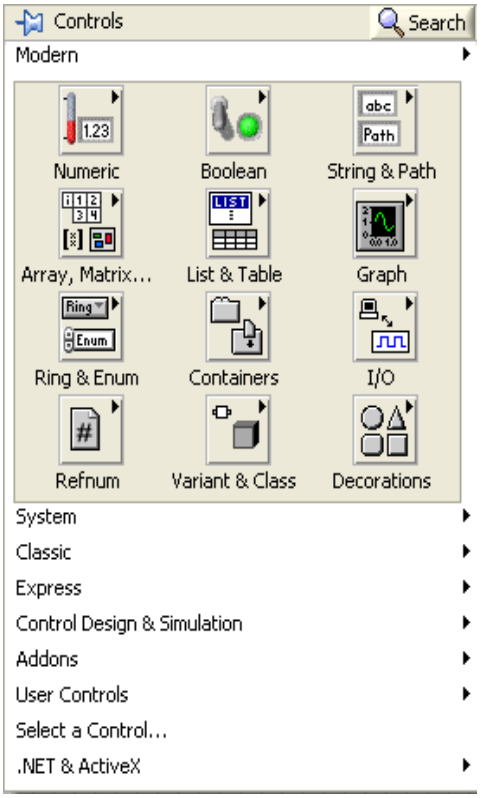
Палитра **Элементов (Controls)** и палитра **Функций (Functions)** содержат разделы, в которых размещены объекты для создания ВП. При нажатии на значок раздела, на экран выводится окно, содержащее его объекты. Для

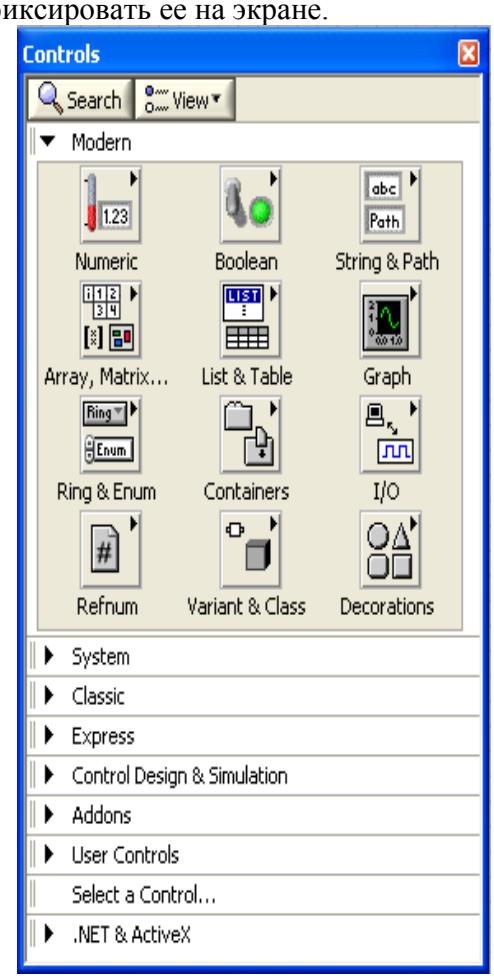
использования объекта палитры следует щелкнуть на нем мышью и поместить выбранный объект на лицевую панель или блок-диаграмму.

Для перемещения по разделам палитры, выбора элементов, ВП и функций следует использовать кнопки навигации. Для открытия ВП можно также щелкнуть правой кнопкой мыши иконку ВП на палитре и выбрать **Open VI** из контекстного меню.

Палитра Элементов

Палитра **Элементов** используется для размещения элементов управления и отображения на лицевой панели. Она доступна только на лицевой панели. Чтобы отобразить палитру **Элементов**, следует либо выбрать в пункте главного меню **Window»Show Controls Palette**, либо щелкнуть правой кнопкой мыши в рабочем пространстве лицевой панели. Используя кнопку в верхнем левом углу палитры, можно зафиксировать ее на экране.

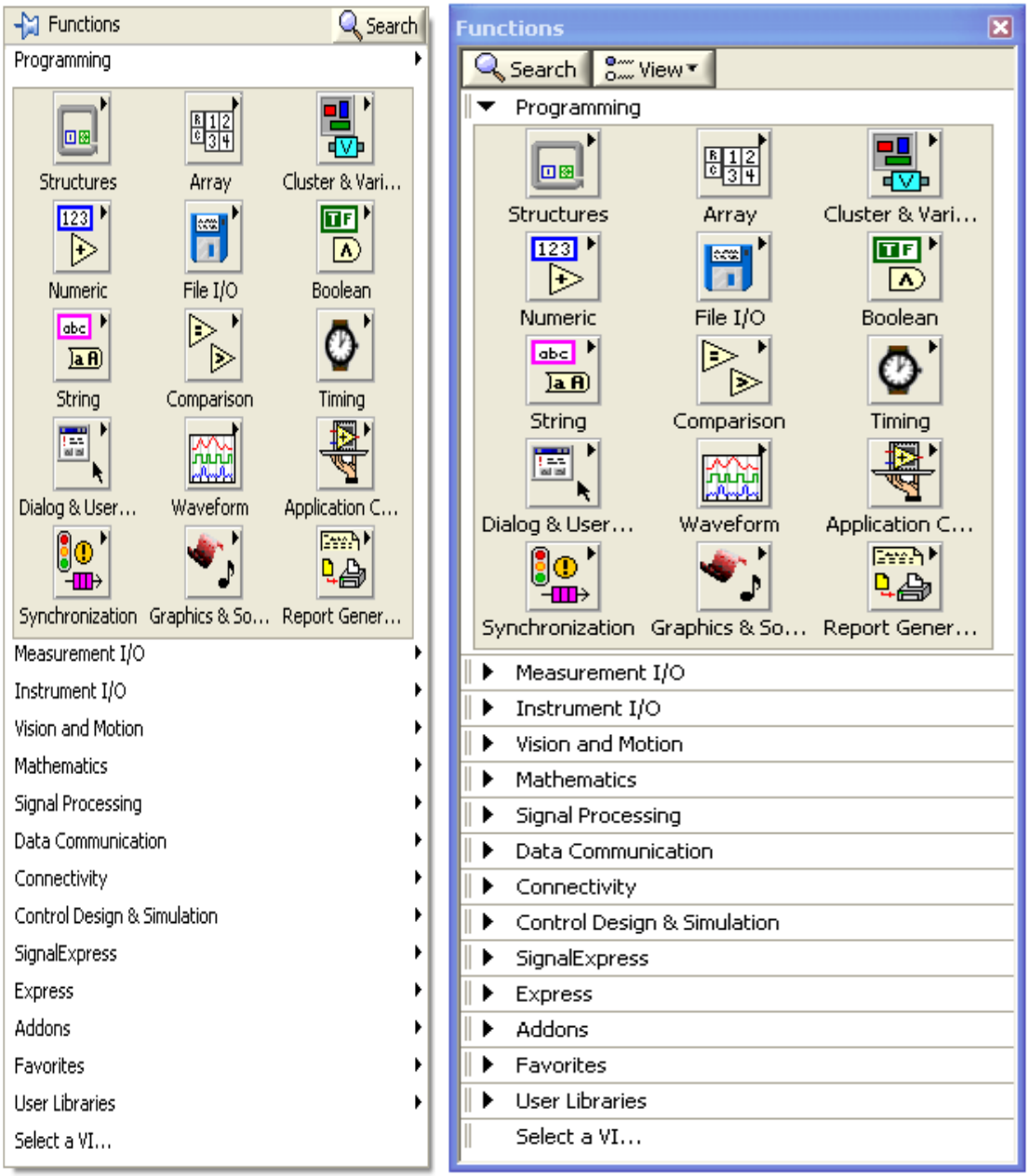




Палитра Функций

Палитра **Функций** используется для создания блок-диаграммы. Она доступна только на блок-диаграмме. Чтобы отобразить палитру **Функций**, следует либо выбрать в пункте главного меню **Window»Show Functions Palette,** либо щелкнуть правой кнопкой мыши в рабочем пространстве блок-

диаграммы. Используя кнопку в верхнем левом углу палитры можно зафиксировать ее на экране.



• Пункты на палитрах организованы в соответствии с категориями. Вы можете выбрать способ отображения различных категорий на палитре (**Category(Standard), Category (Icons and Text), Icons, Icons and Text, Text, Tree**), используя кнопку **View** в верхней части палитры.

• При способе отображения пунктов на палитре **Category(Standard)** и **Category (Icons and Text)** Вы можете изменить порядок расположения пунктов на палитре, используя в контекстном меню к категории опции **Move this Category Up** и **Move this Category Down.** Кроме того, Вы можете перетащить категорию, кликнув на двойную линию,

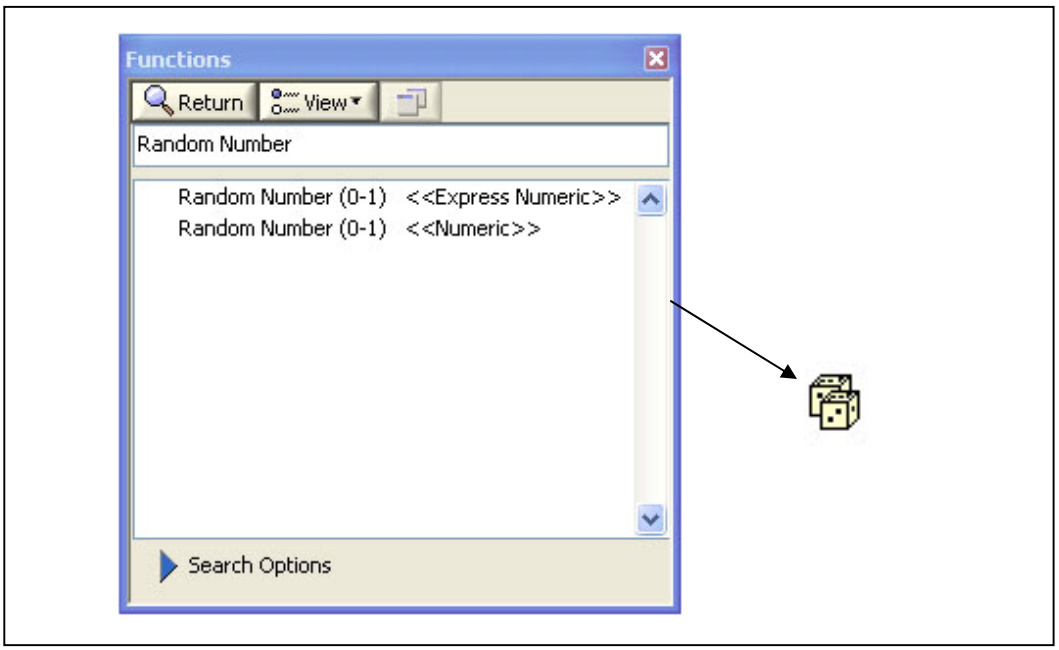
расположенную слева от категории.

• При способе отображения пунктов на палитре **Text** и **Tree** Вы можете упорядочить их в алфавитном порядке. Для этого выберите кнопку **View» Sort Alphabetically.**

• При нажатии на кнопку **Search** Вы можете перейти в режим поиска какой-либо функции, узла или ВП по названию.

Поиск объектов на палитрах Controls (Элементов) и Functions (Функций)

Например, чтобы найти функцию **Random Number** (Генератор случайных чисел)**,** следует нажать кнопку **Search** на палитре **Functions** (Функций) и ввести в поле ввода текста «Random Number». LabVIEW выведет на экран список узлов и функций, в названии которых встречается введенный текст. Выбрав в результатах поиска искомую функцию, можно перенести ее на блок-диаграмму с помощью мыши.



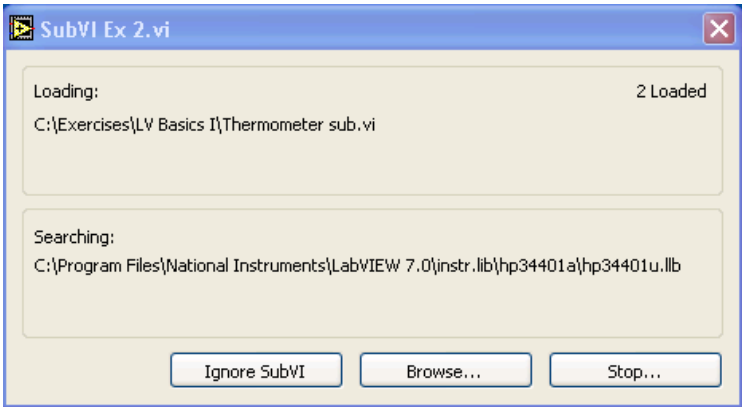
Двойной щелчок кнопкой мыши на искомой функции подсветит ее местоположение на палитре.

Загрузка ВП

С помощью пунктов главного меню **File»Open** открывается диалоговое окно, позволяющее выбрать ВП и загрузить его в память компьютера**.**

Путь к редактируемым ВП – в упражнениях учебного курса **c:\exercises\LV Basics I.**

Ниже представлен вид диалогового окна, появляющегося во время загрузки ВП.



В этом окне перечисляются все подпрограммы выбранного ВП по порядку их загрузки в память. Остановить загрузку можно в любое время, нажав кнопку **Stop**.

Если LabVIEW не может сразу найти подпрограмму, то поиск продолжается по всем директориям, прописанным в пути поиска файлов **Search Path**. Пути поиска файлов можно редактировать, используя пункты меню Инструменты (**Tools » 0ptions » Paths**). Можно сделать и так, чтобы LabVIEW игнорировал подпрограмму, нажав кнопку **Ignore Sub VI**, или использовать ручной поиск подпрограммы, нажав кнопку **Browse** (Обзор).

E. Использование проектов в LabVIEW.

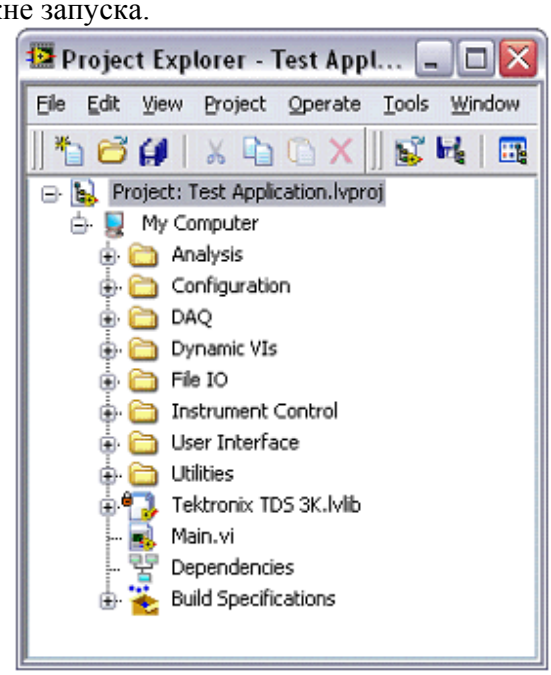
Используйте проект для того, чтобы сгруппировать между собой файлы, созданные в LabVIEW, файлы, созданные в других программах (non-LabVIEW files) и спецификацию к программе. А также для того, чтобы загрузить или использовать файлы в **Target**. **Target** это устройство или механизм, на котором выполняется данный виртуальный прибор.

При сохранении проекта LabVIEW создаст проект-файл с расширением *(.lvproj)*, который включает в себя информацию о конфигурации, структуре, распаковке виртуального прибора, ссылки на файлы, используемые в проекте и т.п. Проект нужно использовать для создания законченного автономного приложения или общей библиотеки. Вы также можете использовать проект для работы с модулями LabVIEW Real-Time, FPGA, or PDA.

Окно проекта Project Explorer Window

Для работы с проектом используйте окно проекта **Project Explorer Window**. Для отображения на экране окна проекта выберите в меню окна запуска **File»New Project**. Вы также можете отобразить окно проекта, выбрав **Project»New Project** или выбрав последовательно **File»New** и затем **Empty**

Project. Помимо этого можно воспользоваться диалоговым окном **New,** находящимся в окне запуска.



По умолчанию окно проекта содержит следующие пункты:

- **Project root (корень проекта)**—содержит все части проекта. В заголовке корня проекта находится название проекта.
- **My Computer (мой компьютер)**—представляет компьютер в качестве устройства **(target)**, на котором выполняется проект.
- **Dependencies (зависимости)**—включают в себя пункты, которые требуются для работы виртуального инструмента на данном устройстве.
- **Build Specifications (настройки для создания приложения)**—включают в себя конфигурацию для исходных файлов и других приложений доступных среди инструментов и модулей LabVIEW. Спецификация для настройки содержит все установки для создания, подобно тому какие файлы включать, директории создавать и установки для директорий виртуального инструмента. Если у Вас инсталлирована LabVIEW Professional Development System или Application Builder, то Вы можете, используя **Build Specifications,** сконфигурировать самостоятельное приложение (EXEs), динамическую библиотеку (DLLs), и zip файл. **(Windows)** Вы также можете использовать **Build Specifications** для того, чтобы сконфигурировать инсталлятор.

При добавлении в проект еще одного устройства (target) LabVIEW создаст пункт в окне проекта (**Project Explorer)** для отображения дополнительного устройства. Каждое дополнительное устройство (target) включает в себя **Dependencies** и **Build Specifications.** Вы можете добавлять файлы к каждому устройству. Существует также возможность перетаскивать виртуальные

инструменты VIs из окна проекта на блок диаграмму другого открытого виртуального прибора для использования этого VI в качестве подпрограммы subVI.

Использование проекта с существующими файлами

Вы можете добавить группу уже существующих файлов в проект. Для добавления файлов в проект можно пойти по одному из следующих путей:

• Если существующие файлы являются частью самостоятельного приложения, то выберите **Tools»Convert Build Script** для того, чтобы сконвертировать приложение в проект.

• Если существующие файлы не являются частью самостоятельного приложения, то создайте новый проект и добавьте файлы в окно проекта.

Инструментальная линейка окна проекта

Используйте кнопки, находящиеся на линейке инструментов для проведения различных операций в проекте. Линейка инструментов расположена в верхней части окна проекта. При желании Вы можете изменять размер окна проекта для того, чтобы увидеть всю линейку инструментов. Вы также можете скрыть или отобразить линейку инструментов, нажав **View»Toolbars** и выбрав там линейку инструментов, которую Вы хотите скрыть или сделать видимой. Кроме того, скрыть или отобразить линейку инструментов можно, нажав правой кнопкой мыши на линейку инструментов и воспользовавшись контекстным меню.

Добавление пунктов в проект

Используя окно проекта, Вы можете добавить в проект файлы, созданные в LabVIEW: такие как VIs, LLBs, либо различные другие файлы, например, текстовые файлы или таблицы. Кроме того, Вы можете создать собственную структуру для организации различных пунктов в проекте.

Добавление файлов в проект

- Для добавления уже существующего файла в проект выполните следующую последовательность действий: в окне проекта нажмите правой кнопкой мыши на строке отображающей устройство, на котором выполняется проект (в нашем случае это **My Computer**), или на папке, находящейся в **My Computer**. В появившемся контекстном меню выберите **Add File** и выберите файл, который Вы желаете добавить. Вы также можете воспользоваться пунктом **Project**, находящимся в линейке инструментов окна проекта. Для добавления файла в проект нажмите **Project»Add To Project»Add File**.

- Для добавления нового файла в проект выберите **New»VI** из контекстного меню к пункту **My Computer**, либо, используя линейку инструментов

окна проекта, выберите **File»New VI** или **Project»Add To Project»New VI**.

- Если виртуальный инструмент, который Вы хотите добавить в проект, уже открыт, то Вы можете подвести курсор мыши к иконке, расположенной в правом верхнем углу окна и, нажав левую кнопку мыши, перетащить иконку к **My Computer.**

Добавление папок в проект

Для добавления в проект папок, расположенных на диске

- нажмите правой кнопкой мыши на строке отображающей устройство, на котором выполняется проект (в нашем случае это **My Computer**), и из контекстного меню выберите **Add Folder.** После этого в проекте появится папка, содержание которой соответствует содержанию внешней папки, включая файлы и вложенные папки. Для добавления папки можно также воспользоваться линейкой инструментов окна проекта. Выберите **Project»Add To Project»Add Folder**.

Если Вы изменили содержание папки на диске, то LabVIEW автоматически не внесет обновления в проект. Если Вы хотите, чтобы содержание папки в проекте соответствовало содержанию внешней папки, то это нужно сделать в ручном режиме.

- Вы также можете создать новую папку в проекте для того, чтобы организовать различные пункты в проекте. Для этого нажмите правой кнопкой мыши на **My Computer** и из контекстного меню выберите **New»Folder.** Если Вы нажмете правой кнопкой мыши на корень проекта, то Вы можете организовать устройства **(Targets)** находящиеся в проекте.

Добавление библиотек LLBs в проект

Вы можете добавить в проект библиотеку LLB в качестве папки или файла. Если Вы добавляете библиотеку в качестве папки, то LabVIEW назовет папку именем библиотеки и добавит виртуальные приборы VIs находящиеся в библиотеке в качестве пунктов внутри папки. Для добавления библиотеки в проект нажмите правой кнопкой мыши на строке отображающей устройство, на котором выполняется проект (в нашем случае это **My Computer**), и из контекстного меню выберите **Add Folder.** Затем, используя диалоговое окно, добавьте библиотеку LLB в проект.

Добавление или удаление пунктов из папки, принадлежащей проекту, не вносит изменений в положение файла с расширением .llb на диске.

Вы также можете добавить библиотеку LLB в проект в качестве файла. Если Вы добавляете библиотеку как файл, то виртуальные приборы, находящиеся внутри библиотеки не будут отображаться в окне проекта. Для добавления

библиотеки в виде файла нажмите правой кнопкой мыши на строке отображающей устройство, на котором выполняется проект (в нашем случае это **My Computer**), или на папку внутри проекта и из контекстного меню выберите **Add File.** Затем, используя диалоговое окно, выберите нужный Вам файл.

Удаление пунктов из проекта

Удалить какой-нибудь пункт следующими способами:

- Нажмите правой кнопкой мыши на пункт, который Вы желаете удалить, и из контекстного меню выберите **Remove.**
- Выберите пункт, который Вы хотите удалить и нажмите клавишу <Delete> на клавиатуре.
- Выберите пункт, который Вы хотите удалить и нажмите кнопку **Delete** на панели инструментов, расположенной в верхней части окна проекта.

Удаление пункта из проекта не приводит к удалению соответствующего файла на диске.

Зависимости проекта (Dependencies)

Используйте папку зависимости (**Dependencies)** для того, чтобы увидеть, какие пункты (файлы и т.п) требуются устройству (**Target**). Каждое устройство включает в себя папку **Dependencies**. Вы можете не добавлять самостоятельно какие-либо пункты в **Dependencies**, LabVIEW сделает это автоматически, если нажать правой кнопкой мыши на строку **Dependencies** и из контекстного меню выбрать **Refresh**. Например, если Вы добавили в проект виртуальный прибор, который содержит подпрограммы (subVIs), то LabVIEW добавит данные зависимости в **Dependencies**, когда Вы обновите папку, используя **Refresh**. Однако, если Вы добавили зависимый файл в проект самостоятельно, то LabVIEW не добавит этот файл в папку **Dependencies.** Например, если Вы добавили виртуальный прибор VI и subVI, то LabVIEW не добавит subVI в **Dependencies**, при нажатии **Refresh.** Зависимости включают в себя VIs, DLLs и библиотеки проекта, которые виртуальный прибор вызывает статически.

Файлы, которые виртуальный прибор вызывает динамически, не появляются в папке **Dependencies**. Вы должны добавить эти пункты в проект в папку, отображающую устройство **(target)**, для того, чтобы управлять ими в проекте.

LabVIEW отслеживает подпрограммы subVIs, но не отслеживает динамические библиотеки DLLs. Например, если виртуальный прибор a.vi вызывает динамическую библиотеку b.dll статически, b.dll вызывает статически c.dll, то LabVIEW будет рассматривать в качестве зависимого пункта только библиотеку b.dll. Для управления библиотекой c.dll Вы должны добавить ее в папку отображающую устройство (Target).

Если зависимый пункт является частью библиотеки проекта, то LabVIEW добавит всю библиотеку в пункт **Dependencies.**

Нельзя поместить в папку **Dependencies** некий новый пункт. Вы также не сможете перенести какой-либо пункт из другого места в окне проекта в папку **Dependencies.**

Для удаления какого-либо пункта из папки **Dependencies** нажмите правой кнопкой мыши на данный пункт и из контекстного меню выберите **Remove.**

Если Вы удалили какой-то пункт из папки **Dependencies**, то данный пункт появится там снова, если Вы обновите папку **Dependencies**, используя **Refresh.**

При сохранении проекта LabVIEW не сохраняет папку **Dependencies** в качестве части проекта. То есть когда Вы вновь открываете проект, то Вы обновить папку **Dependencies,** используя в контектстном меню опцию **Refresh.**

Добавление устройств (Targets) в проект

Для добавления в проект устройства (**Target)** используйте диалоговое окно **Add Targets and Devices.**

Для того, чтобы устройство отображалось в диалоговом окне должны быть подключены все необходимые модули и инсталлированы все нужные драйвера.

Чтобы отобразить диалоговое окно **Add Targets and Devices** нажмите правой кнопкой мыши на корень проекта и выберите из контекстного меню **New»Targets and Devices**.

Если устройство в проекте поддерживает другие устройства, то Вы можете добавить это устройство в проект в качестве подустройства. Например, если у Вас на компьютере присутствует NI PCI устройство, то Вы можете добавить это устройство в качестве папки **My Computer.**

Использование файлов устройством

Для того чтобы устройство, подключенное в проект, использовало файлы проекта, откройте и запустите виртуальный прибор, находящийся внутри папки **Target (My Computer)**.

Работа с приложениями

LabVIEW создает приложение (приложение LabVIEW) для каждого устройства, подключённого к проекту. Когда Вы открываете виртуальный прибор, находящийся в окне проекта, то виртуальный прибор открывает приложение для устройства. Используйте имя приложения, которое появляется левом нижнем углу на лицевой панели на панели блик-диаграмм виртуального прибора, для идентификации к какому приложению принадлежит виртуальный прибор. Если у Вас одновременно открыто

несколько проектов, то имя приложения включает в себя имя проекта и название устройства, например: **Project 1.lvproj/My Computer**. Если открыт только один проект, то имя приложения включает в себя только название устройства.

Если Вы удалили виртуальный прибор из окна проекта в тот момент, когда этот виртуальный прибор открыт, то он все равно останется в приложении. Если Вы потом закроете виртуальный прибор и откроете его снова, то он опять появится в приложении.

Редактирование виртуальных приборов в нескольких приложениях

Вы можете открыть один и тот же виртуальный прибор, расположенный на диске, в нескольких приложениях одновременно. Например, вы можете открыть виртуальный прибор, используемый в двух различных проектах, или двумя различными устройствами внутри проекта. Если Вы редактируете виртуальный прибор, который открыт в одном приложении, а затем открываете этот виртуальный прибор в другом приложении, то он будет содержать в себе внесенные изменения. Однако, если редактируемый виртуальный прибор открыт более, чем в одном приложении, то LabVIEW не перенесет изменения в другое приложение автоматически. Кроме того, Вы не сможете редактировать, запустить или сохранить виртуальный прибор в других приложениях до тех пор пока он не будет одинаковым во всех приложениях. Для того чтобы виртуальный прибор имел одну и ту же форму во всех приложениях, используйте один из следующих способов:

- Нажмите кнопку **Synchronize with Other Application Instances,** расположенную на панели инструментов виртуального прибора.
- Сохраните виртуальный прибор в приложении, которое содержит изменения. После того, как Вы сохранили приложение, LabVIEW автоматически внесёт изменения в другие приложения.
- Отмените изменения, которые Вы внесли в виртуальный прибор.

Вы не сможете отменить изменения, после того как Вы сохранили или синхронизовали виртуальный прибор.

Вы не сможете редактировать виртуальный прибор, пока он выполняется или занят другим приложением.

Если виртуальный прибор является частью библиотеки проекта и запущен в каком-то приложении, то LabVIEW временно заблокирует библиотеку во всех приложениях.

Использование Веб-сервера для виртуальных приборов в проекте.

Вы можете использовать Веб-сервер для просмотра лицевой панели виртуального прибора. Для этого выберите на линейке инструментов окна проекта **Operate»Connect to Remote Panel** и впишите название виртуального

прибора в строку **VI name**. Название должно включать в себя имя проекта, библитеку, название устройства и название самого виртуального прибора. Например, если виртуальный прибор *MyVI.vi* принадлежит проекту *MyProject.lvproj* и выполняется устройством My Computer, то введите имя файла следующим образом: *MyProject.lvproj/My Computer/MyVI.vi.* Если виртуальный прибор находится ещё в библиотеке MyLibrary, то имя библиотеки тоже надо ввести в стороку **VI name**: *MyProject.lvproj/My Computer/MyLibrary:MyVI.vi.* Если виртуальный прибор не находится в проекте, то введите его имя без дополнительной информации.

Библиотеки проекта в LabVIEW

Библиотека проекта это собрание различных виртуальных приборов, определений типов, общих переменных (shared variables), файлов палитр и др. файлов, включая другие библиотеки проекта. Когда Вы создаете и сохраняете новую библиотеку проекта, LabVIEW создает файл библиотеки (*.lvlib*), который включает в себя свойства библиотеки и ссылки на файлы, включённые в библиотеку. Библиотека проекта может быть полезной, если Вы хотите создать единую иерархию файлов внутри проекта, избежать дублирования названий виртуальных приборов, ограничить доступ к каким-либо файлам, ограничить возможность редактирования каких-то файлов и установить одинаковый тип меню для группы виртуальных приборов. Структуру библиотеки проекта можно увидеть в окне проекта или в отдельном окне библиотеки. Вы можете создать библиотеку проекта в проекте LabVIEW следующим образом:

- Нажмите правой кнопкой мыши на строке, отображающей устройство (**My Computer)** и из контекстного меню выберите **New»Library.** После этого LabVIEW создаст библиотеку проекта, которая появится в окне проекта.

- Вы также можете сделать библиотеку из какой-нибудь папки проекта. Для этого нажмите правой кнопкой мыши на нужную папку и выберите из контекстного меню **Convert to Library**.

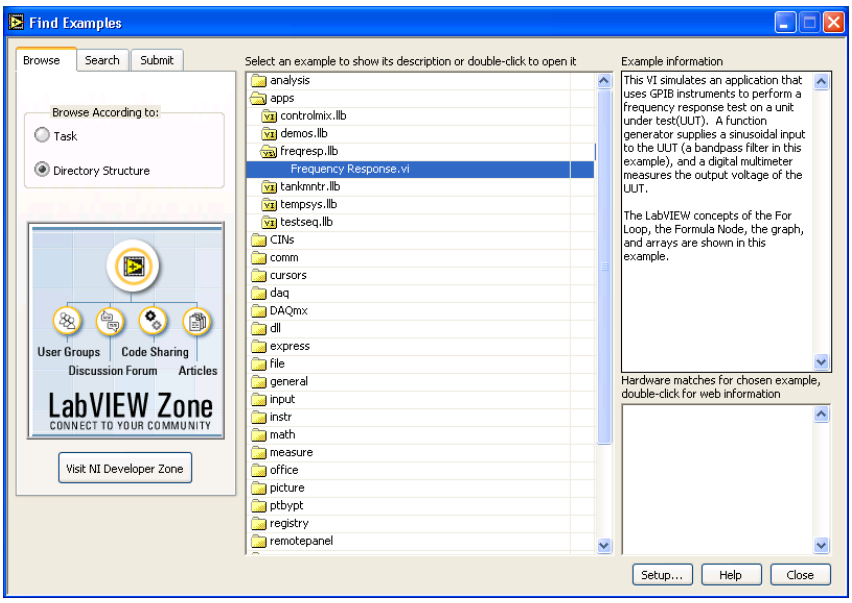
Используйте библиотеки проекта для того, чтобы организовать логическую иерархию пунктов проекта. Библиотека проекта не содержит файлы в действительности, в отличие от LLB библиотеки, которая является физическим файлом и содержит виртуальные приборы. Файлы, которые содержит библиотека проекта, находятся на диске независимо. То есть структура библиотеки проекта может отличаться от структуры файлов на диске.

Если Вы хотите переименовать библиотеку проекта, то нажмите на ней правой кнопкой мыши и выберите из контекстного меню **Rename**. Будьте внимательны: если Вы переименуете библиотеку проекта вне среды LabVIEW, то Вы можете испортить библиотеку проекта.

Упражнение 1-1. ВП Частотный анализ

Цель: Открыть и запустить ВП

1. Запустите LabVIEW (**Start»Programs»National Instruments» LabVIEW 7.0» LabVIEW**). Появится диалоговое окно **LabVIEW**.
2. Выберите **Help»Find Examples**. На экране появится диалоговое окно поиска примеров ВП, разбитых по категориям.

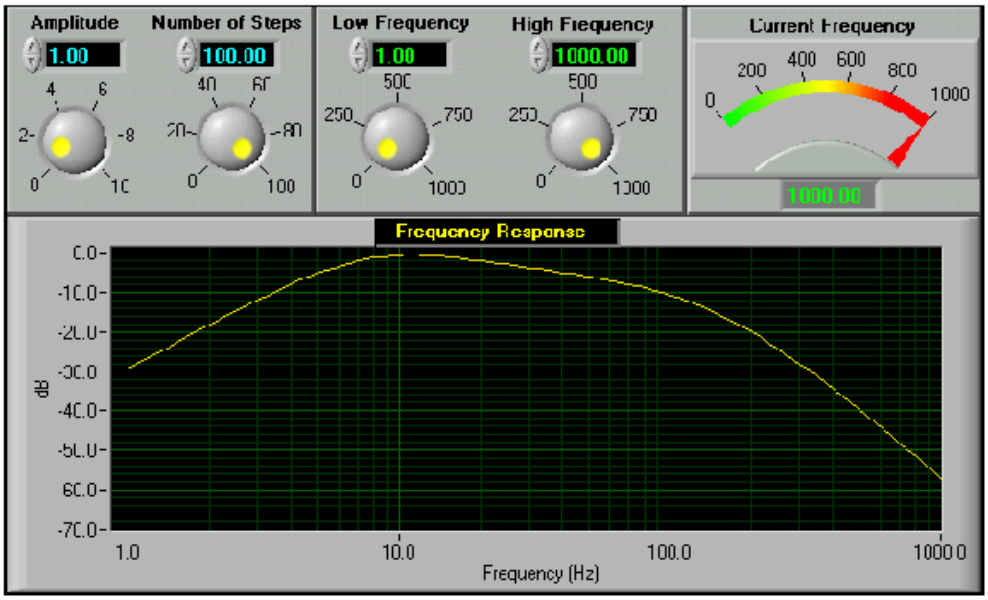


3. Перейдите на закладку **Browse** (Обзор). Отметьте пункт **Directory Structure**. Выберите **Apps**, **Freqresp.llb**, и дважды щелкните на **Frequency Response VI**. Появится лицевая панель ВП Частотный анализ.

Примечание. Открыть этот ВП можно, нажав кнопку **Open VI** и перейдя в директорию **labview\examples\apps\freqresp.llb\Frequency Response.vi.**

Лицевая панель

4. На инструментальной панели нажмите кнопку **Run**, показанную слева, и запустите ВП. Данный ВП моделирует посылку сигнала к измерительному прибору и регистрацию его отклика. Реакция прибора в частотной области показана на графике лицевой панели.



5. С помощью инструмента УПРАВЛЕНИЕ измените значение установки амплитуды **Amplitude**. Изменить значение можно либо, переместив указатель кнопки в нужное положение, либо используя стрелки изменения значений элемента управления, либо введя число непосредственно в дисплей элемента.

Если число введено непосредственно в дисплей элемента, то необходимо нажать кнопку **Enter**, показанную слева, появившуюся на инструментальной панели. Иначе число не будет введено.

(MacOS and Sun) Нажать кнопку **<Return>.**

6. Нажать кнопку **Run** и запустить ВП. Изменяя значения других средств управления, находящихся на панели, исследовать работу ВП.

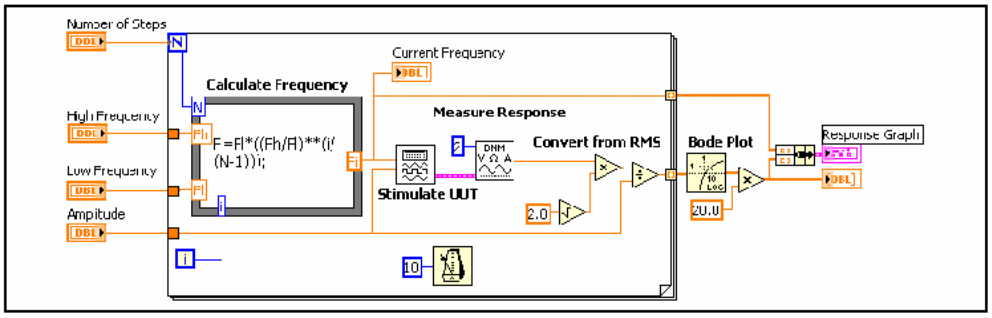
Блок-диаграмма

7. Перейдите на блок-диаграмму. Для этого выберите в главном меню **Window»Show Diagram** или введите **<Ctrl-E>** с клавиатуры.

   (MacOS) Нажмите кнопки **<Command-E>**.

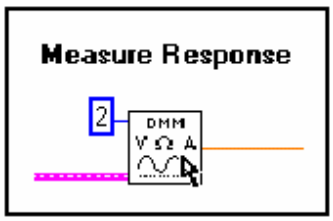
   (Sun) Нажмите кнопки **<Meta-E>.**

   (Linux) Нажмите кнопки **<Alt-E>.**



Блок-диаграмма содержит несколько основных объектов, включая подпрограммы ВП, функции и структуры, о которых пойдет речь позднее.

8. С помощью инструмента УПРАВЛЕНИЕ дважды щелкните по иконке **DMM**.



Эта иконка – графическое представление подпрограммы **Demo Fluke 8840A VI.** После двойного щелчка откроется подпрограмма и на экране появится ее лицевая панель.

Дизайн лицевой панели напоминает мультиметр. Вот почему программы LabVIEW называются виртуальными приборами.

Создавая модульные приложения LabVIEW, можно изменять только части приложения и/или многократно использовать эти части в других приложениях. Например, эта подпрограмма моделирует действие комбинированного прибора **Demo Fluke**, но пользователь может внести в него изменения, чтобы получить новые функции.

9. Выберите в главном меню пункты **File»Close** и закройте **Demo Fluke 8840AVI.**

10. Не закрывайте ВП Частотный анализ. Этот ВП будет использоваться в упражнении 1-2.

Конец упражнения 1-1

F. Встроенная Помощь среды LabVIEW (LabVIEW Help) и руководство пользователя

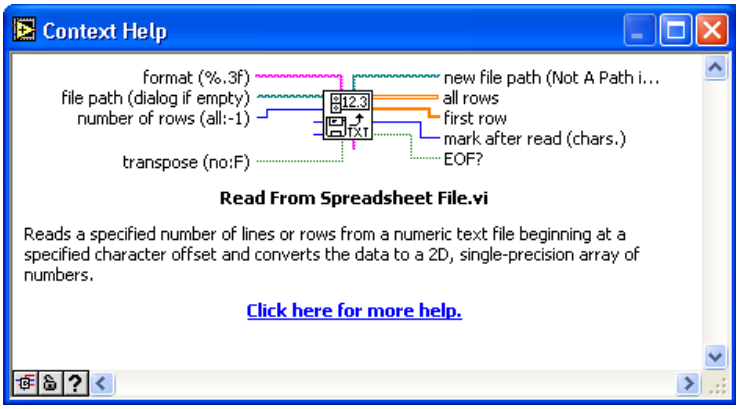
Окно контекстной справки **Context Help** помогает при создании и редактировании ВП. Более подробная информация расположена в **LabVIEW Help** (Встроенной Помощи).

Окно контекстной справки

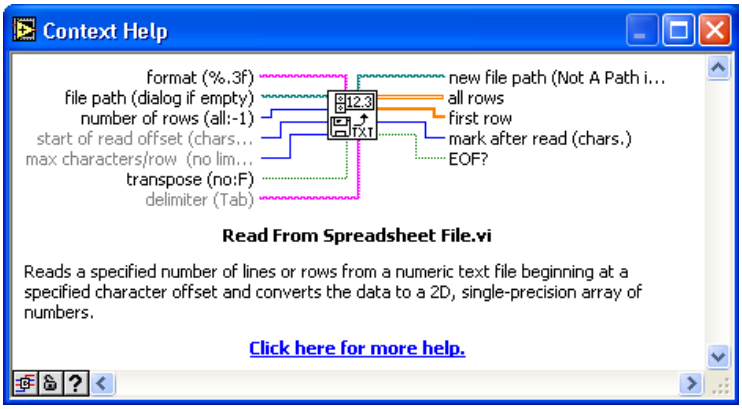
Окно **Context Help** (контекстной справки) выводится на экран из пункта главного меню **Help»Show Context Help** или вводом **<Ctrl-H>** с клавиатуры.

(MacOS) Нажмите на кнопки **<Command-H>. (Sun)** Нажмите на кнопки **<Meta-H>. (Linux)** Нажмите на кнопки **<Alt-H>.**

При наведении курсора на объект лицевой панели или блок-диаграммы в окне **Context Help** (контекстной справки) появляются иконка подпрограммы ВП, функции, константы, элементов управления или отображения данных с указанием всех полей ввода/вывода данных. При наведении курсора на опции диалогового окна в окне **Context Help** (контекстной справки) появляется описание этих опций. При этом поля, обязательные для соединения, выделены жирным шрифтом, рекомендуемые для соединения поля представлены обычным шрифтом, а дополнительные (необязательные) поля – выделены серым или вообще не показаны. Ниже приведен пример окна контекстной справки **Context Help**.



Для переключения между кратким и подробным представлением окна контекстной справки следует нажать кнопку **Simple/Detailed Context Help,** расположенную в нижнем левом углу окна **Context Help**. В кратком режиме представлены основные поля ввода/вывода данных. Дополнительные поля не описаны. Подробный режим показывает все поля ввода/вывода данных, как показано ниже:



Чтобы зафиксировать текущее окно **Context Help** (контекстной справки), необходимо нажать кнопку **Lock Context Help**. Когда текущее окно **Context Help** (контекстной справки) зафиксировано, то его содержимое не меняется после наведения курсора на другой объект. Для отмены фиксации следует нажать кнопку второй раз. Описание опций можно получить из меню **Help**.

Для отображения подробного описания объекта в соответствующем разделе **LabVIEW Help** (Встроенной Помощи) следует нажать кнопку **More Help**.

Встроенная Помощь LabVIEW

Для отображения **LabVIEW Help** (Встроенной Помощи) можно нажать кнопку **More Help** в окне **Context Help** (контекстной справки) и выбрать в пункте главного меню помощь – **Help»VI, Function, & How-To Help** или в окне **Context Help** (контекстной справки) щелкнуть на **Click here for more help**.

Встроенная Помощь LabVIEW содержит детальные описания большинства палитр, меню, инструментов, ВП и функций, включает в себя пошаговую инструкцию использования особенностей LabVIEW и связана с ***LabVIEW Tutorial*** *(руководством пользователя)***,** PDF версией учебника LabVIEW и технической поддержкой на Web-сайте National Instruments.

Упражнение 1-2. Использование справочной системы LabVIEW и руководства пользователя

Цель: Использование справочной системы LabVIEW для получения информации об объектах лицевой панели и блок-диаграммы и особенностях их использования

Часть A. Окно контекстной справки

1. ВП Частотный анализ должен быть открыт. Если нет, откройте его, как описано в упражнении 1-1.
2. Выберите в главном меню пункты **Window»Show Diagram,** чтобы перейти на блок-диаграмму.
3. Выберите в главном меню пункты **Help»Show Context Help** или нажмите клавиши **<Ctrl-H>,** чтобы отобразить окно контекстной справки **Context Help**.

   (MacOS) Нажмите на кнопки **<Command-H>. (Sun)** Нажмите на кнопки **<Meta-H>. (Linux)** Нажмите на кнопки **<Alt-H>.**
4. Получить информацию об объекте в окне контекстной справки **Context Help** можно, наведя на них курсор.

   a. Поместите инструмент ПЕРЕМЕЩЕНИЕ, показанный слева, над функцией **Logarithm Base 10**, расположенной под меткой **Bode Plot.** В окне контекстной справки **Context Help** появится описание функции.

   b. В окне контекстной справки **Context Help** нажмите кнопку **More Help**, показанную слева, для перехода в соответствующий раздел **LabVIEW Help** (Встроенной Помощи)**.** Можно также щелкнуть по ссылке **Click here for more help** окна контекстной справки **Context Help**.

      LabVIEW Help (Встроенная Помощь) содержит подробное описание палитр, меню, инструментов, ВП и функций. Получите подробное описание других функций.

   c. Наведите инструмент СОЕДИНЕНИЕ, показанный слева, на поля ввода/вывода данных функции **Logarithm Base 10.** Соответствующие поля в окне контекстной справки **Context Help** начнут мигать.

   d. Переместите инструмент СОЕДИНЕНИЕ на проводник данных. В окне контекстной справки **Context Help** появится описание типа данных в проводнике.

Часть B. Справочная система LabVIEW

5. Выберите в главном меню пункты **Help»VI, Function, & How-To Help,** чтобы открыть **LabVIEW Help** (Встроенную Помощь среды LabVIEW). **LabVIEW Help** (Встроенная помощь среды LabVIEW) включает в себя пошаговую инструкцию использования особенностей LabVIEW и

связана с ***LabVIEW Tutorial*** *(руководством пользователя),* PDF версией учебника LabVIEW и технической поддержкой на Web-сайте National Instruments.

6. Использование оглавления справочной системы LabVIEW.
   a. Нажмите закладку **Index** для вывода оглавления справочной системы LabVIEW.
   b. В окне ввода текста введите Frequency Response. В оглавлении появятся два близких по смыслу ответа на запрос.
   c. Просмотрите каждый. Справочная система LabVIEW выведет соответствующий раздел.
   d. Нажмите закладку **Contents** для вывода содержания раздела справочной системы LabVIEW.
   e. Снова нажмите закладку **Index**.
   f. В окне ввода текста напечатайте GPIB, потому что ВП Частотный анализ – симулятор приложения GPIB.
   g. Войдите в подраздел **«functions»** для просмотра описания работы функций GPIB в LabVIEW.
7. Выполните поиск по всей справочной системе LabVIEW.
   a. Нажмите закладку **Search**.
   b. В окне ввода текста напечатайте displaying frequency. В окне результата поиска нажмите **«Frequency Information Displayed».**
8. Если на компьютере установлен **«Adobe Acrobat Reader»**, нажмите на закладку **Contents** и откройте PDF версию руководства пользователя LabVIEW справочной системы LabVIEW.
   a. Выберите папку **Related Documentation** в окне **Contents**. Справа появится соответствующий раздел.
   b. Нажмите ссылку **LabVIEW User Manual**, чтобы открыть PDF версию руководства пользователя LabVIEW.
   c. Нажмите кнопку **Back,** чтобы вернуться в раздел **Related Documentation.**
9. Если компьютер подключен к Интернету, обратиться к технической поддержке на *Web-сайте National Instruments*.
   a. Выберите папку **Technical Support Resources** в основном разделе **Contents** (Содержания).
   b. Войдите в папку и выберите страницу **Technical Support Resources**. Появится раздел ресурсов технической поддержки.
   c. Выберите ссылку **Technical Support** для перехода в секцию **Техническ**ой **поддер**жки на *ni.com.*
   d. Вернитесь в раздел **Technical Support Resources**.

e. Выберите ссылку **NI Developer Zone** и войдите в **Developer Zone** (Зону Разработчика) National Instruments.

f. В окне ввода текста напечатайте **Frequency Response** и нажмите **GO**.
Появившиеся разделы демонстрируют различные решения с помощью продуктов фирмы National Instruments.

Часть C. Книжная полка LabVIEW

10. Если на компьютере установлен Acrobat Reader, выберите в меню **Help» Search the LabVIEW Bookshelf** и перейдите в **LabVIEW Bookshelf** (Книжную Полку LabVIEW)**.**
11. Нажмите ссылку **Search**, появится диалоговое окно **Adobe Acrobat Search**.
12. В поле ввода текста напечатайте **«Frequency Response»** и нажмите кнопку **Search**. На экране появятся все доступные руководства LabVIEW.
13. Дважды щелкните на первом результате поиска. На экране появится точное местоположение **«Frequency Response»** в документе.
14. Снова выведите результат поиска, выбрав пункты меню **Edit»Search»Results**.
15. Просмотрите другие результаты поиска и выйдите из Acrobat Reader.
16. На лицевой панели выберите пункты меню **File»Close** и закройте ВП Частотный анализ. Изменения не сохраняйте.

Конец упражнения 1-2

Краткое изложение пройденного материала, советы и секреты

- ВП состоит из четырех основных компонентов – лицевой панели, блок-диаграммы, иконки и соединительной панели.
- Лицевая панель – интерфейс пользователя ВП.
- Блок-диаграмма – графический исходный текст программы, состоящий из узлов, терминалов данных и проводников.
- Палитра инструментов предназначена для создания и редактирования ВП. Удерживая нажатой клавишу **<Shift>** и щелкнув правой клавишей мыши, можно вывести на экран временную версию Палитры Инструментов.
- Палитра **Controls** (Элементов) предназначена для создания интерфейса лицевой панели. Для вывода на экран палитры **Controls** (Элементов) следует щелкнуть правой кнопкой мыши в открытом пространстве лицевой панели.
- Палитра **Functions** (Функций) предназначена для создания блок-диаграммы. Для вывода на экран палитры **Functions** (Функций) следует щелкнуть правой кнопкой мыши в открытом пространстве блок-диаграммы.
- Все объекты LabVIEW, свободное рабочее пространство лицевой панели и блок-диаграммы имеют свое контекстное меню. Обращение к всплывающему меню производится щелчком правой кнопкой мыши на объекте, лицевой панели или блок-диаграмме.

  (MacOS) Нажмите на клавишу **<Command>** и щелкните по объекту лицевой панели или блок-диаграммы.
- Меню **Help** используется для вывода на экран окна **Context Help** (контекстной справки) и **LabVIEW Help** (Встроенной Помощи среды LabVIEW), которая описывает палитры, меню, инструменты, ВП и функции, включая пошаговую инструкцию использования особенностей LabVIEW.
- Выберите меню **Help»Search the LabVIEW Bookshelf** и перейдите в **LabVIEW Bookshelf (Книжную Полку LabVIEW)** для использования PDF версии справочной системы и руководства пользователя LabVIEW.

Примечания

Урок 2. Создание ВП

В этом уроке представлены основы создания ВП.

В этом уроке изложены вопросы:

A. Компоненты ВП
B. Создание ВП
C. Типы данных и проводники данных
D. Редактирование ВП
E. Отладка ВП

A. Компоненты ВП

ВП состоит из четырех основных компонентов – лицевой панели, блок-диаграммы, иконки и соединительной панели. Подробная информация о создании иконки и соединительной панели – в уроке 3 «Создание подпрограмм ВП».

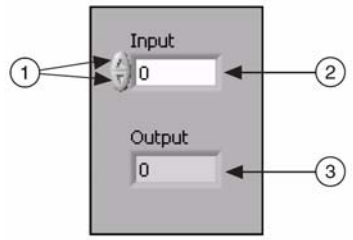
Лицевая панель

На лицевой панели создаются элементы управления и отображения, которые являются интерактивными средствами ввода и вывода данных этого ВП. Элементы **Управления** – кнопки, переключатели и другие устройства ввода данных. Элементы **Отображения** – графики, светодиоды и другие индикаторы. Элементы **Управления** моделируют устройства ввода данных и передают данные на блок-диаграмму ВП. Элементы отображения моделируют устройства вывода и отображения данных, которые получает или генерирует блок-диаграмма.

Для размещения элементов **Управления** и **Отображения** данных на лицевой панели используется палитра **Controls** (Элементов). Палитра **Controls** (Элементов) доступна только с лицевой панели. Для вывода на экран палитры **Controls** (Элементов) следует выбрать пункты главного меню **Window»Show Controls Palette** или щелкнуть правой кнопкой мыши в рабочем пространстве лицевой панели.

Числовые элементы управления и отображения данных

Чаще других используют два числовых объекта – это числовой элемент управления и числовой элемент отображения данных:



1 Кнопки изменения значений

2 Цифровой элемент управления

3 Цифровой элемент отображения данных

Ввод или изменение значения элемента управления осуществляется либо с помощью кнопок приращения значений, либо нужное значение просто вводится в элемент с помощью инструмента ВВОД ТЕКСТА, после чего следует нажать кнопку **<Enter>. (MacOS and Sun)** Нажать кнопку **<Return>.**

Логические элементы управления и отображения данных

Логические элементы управления и отображения используются для ввода и отображения значения логической переменной (TRUE/FALSE —

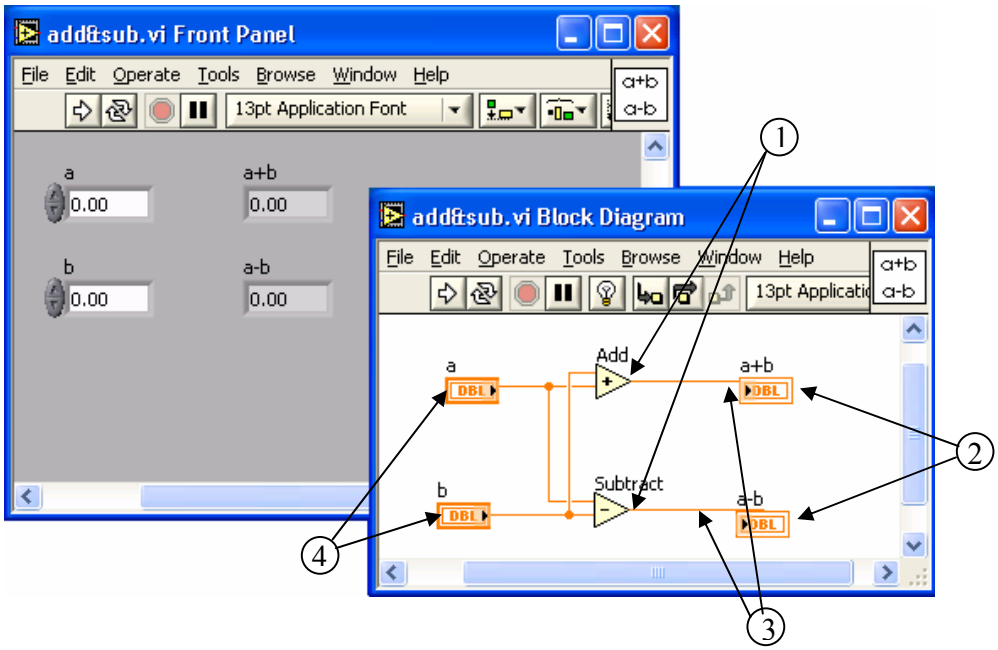
ИСТИНА/ЛОЖЬ). Логические объекты моделируют выключатели, кнопки и светодиоды. Вертикальный переключатель и круглый светодиод показаны ниже:

Редактирование элементов управления и отображения данных

Почти все элементы управления и отображения данных можно редактировать, используя их контекстное меню. Для вызова контекстного меню следует щелкнуть правой кнопкой мыши на объекте. Например, для редактирования метки – щелкнуть правой кнопкой мыши на метке.

Блок-диаграмма

Блок-диаграмма состоит из узлов, терминалов и проводников данных, как показано ниже:



1 Узлы	2 Терминалы данных элементов отображения	3 Проводники данных	4 Терминалы данных элементов управления

Рис. 2-1. Пример блок-диаграммы.

Терминалы данных

Объекты лицевой панели на блок-диаграмме отображаются в виде терминалов данных (графическое изображение прямоугольной формы с

буквенно-численными обозначениями). Буквенно-численное обозначение на терминале данных определяет тип данных, который может использоваться в элементах управления или отображения. Например, DBL-терминал, показанный слева, определяет, что данный элемент управления использует числа двойной точности с плавающей запятой.

Терминал данных может отображаться в виде иконки, показанной слева. Для этого достаточно щелкнуть правой кнопкой мыши в поле терминала данных и выбрать **View as Icon** (отображать в виде иконки) из контекстного меню. Снять метку для отображения в стандартном виде. Отображение терминала данных в стандартном виде позволяет сохранить место на блок-диаграмме.

Терминалы данных обеспечивают обмен данными между лицевой панелью и блок-диаграммой; они подобны переменным и константам текстовых языков программирования. Различают терминалы данных следующих типов – терминалы элементов управления и отображения данных, терминалы узлов. Терминалы элементов управления и отображения относятся к средствам управления и отображения данных на лицевой панели. Данные, введенные в элементы управления на лицевой панели (a и b на рисунке 2-1), поступают на блок-диаграмму через эти терминалы. Когда функции **Add** (Сложение) и **Subtract** (Вычитание) завершают свои вычисления, то на выходе выдают новое значение данных. Эти значения поступают на терминалы элементов отображения данных и передаются на лицевую панель.

Терминалы данных предыдущей блок-диаграммы принадлежат четырем элементам лицевой панели. Соединительная панель функций **Add** (Сложение) **Subtract** (Вычитание), показанная слева, содержит три поля ввода/вывода данных. Для отображения соединительной панели следует щелкнуть правой кнопкой мыши на функции и в контекстном меню выбрать **Visible Items»Terminals**.

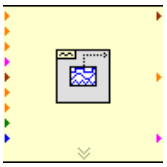
Узлы

Узлы – это объекты на блок-диаграмме, которые имеют одно или более полей ввода/вывода данных и выполняют алгоритмические операции ВП. Они аналогичны операторам, функциям и подпрограммам текстовых языков программирования. Узлы включают в себя функции, подпрограммы ВП и структуры. Подпрограмма ВП – виртуальный прибор, который можно использовать на блок-диаграмме другого ВП в качестве подпрограммы. Структуры – это элементы управления процессом, такие как структура **Case** (Варианта), цикл **While** (цикл по условию) и т.д. Узлы **Add** (Сложение) и **Subtract** (Вычитание)**,** показанные на предыдущей блок-диаграмме, - узлы функций.

Иконки в сравнении с раскрывающимися узлами

ВП и экспресс-ВП могут отображаться в виде иконок или раскрывающихся узлов. Раскрывающиеся узлы представляют собой иконки, окруженные цветным полем. Подпрограммы ВП отображаются на желтом поле, экспресс-ВП — на голубом. Использование иконок, таких как иконка ВП Function Generator, показанная слева, позволяет сохранить место на блок-

диаграмме.



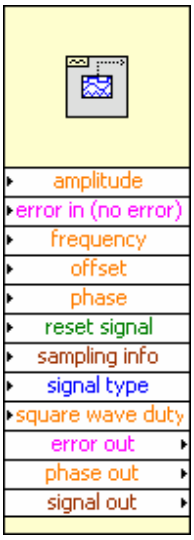
Использование раскрывающихся узлов, таких как раскрывающийся узел ВП Function Generator VI, показанный слева, позволяет упростить процесс соединения и документирования блок-диаграммы. По умолчанию подпрограммы ВП отображаются в виде иконок, экспресс-ВП — в виде раскрывающихся узлов.

Для отображения подпрограммы ВП или экспресс-ВП в виде иконок достаточно щелкнуть по ним правой кнопкой мыши и выбрать в контекстном меню пункт **View as Icon**. Для отображения подпрограммы ВП или экспресс-ВП в виде раскрывающихся узлов снимите выделение с этого пункта.

Предусмотрена возможность изменения размеров раскрывающихся узлов с целью облегчения соединений, но при этом на блок-диаграмме расходуется много места. Для изменения размера узла следует выполнить следующее:

1. Установите инструмент ПЕРЕМЕЩЕНИЕ на узел. На верхнем и нижнем основаниях узла появятся метки изменения размера.
2. Переместите курсор на ручку изменения размера для перевода курсора в режим изменения размера.
3. Переместите курсор, удерживая нажатой левую кнопку мыши, до достижения размера, необходимого для размещения дополнительных терминалов.
4. Отпустите левую кнопку мыши.

На следующем примере показан ВП **Function Generator** в виде раскрывающегося узла с измененным размером.



Внимание. При отображении подпрограммы ВП или **Экспресс-ВП** в виде раскрывающегося узла, поля ввода/вывода этого узла не отображаются.

B. Создание ВП

Для создания ВП откройте новый ВП или шаблон и сохраните его. После этого можно конструировать лицевую панель и блок-диаграмму.

Открытие ВП и Шаблоны

Диалоговое окно **New** используется для создания различных компонент в среде LabVIEW при построении приложений. Можно начинать с пустого ВП или с шаблона для упрощения программирования. Диалоговое окно New содержит следующие компоненты:

- **Create New —** отображает шаблоны, с помощью которых можно создавать ВП или другие документы LabVIEW. Для этого достаточно выбрать шаблон и нажать кнопку OK.
  - **VI** — содержит различные ВП.
    - **Blank VI** — открывает пустые лицевую панель и блок-диаграмму.
    - **VI from Template** — открывает лицевую панель и блок-диаграмму, содержащие компоненты для построения различных видов ВП.
      - **Frameworks** — открывает лицевую панель и блок-диаграмму, содержащие компоненты для построения ВП, включающих специальные виды выполняемых функций
      - **Instrument I/O** — открывает лицевую панель и блок-диаграмму, содержащие компоненты, необходимые для связи с внешними устройствами, подсоединенными к компьютеру
      - **Simulated** — открывает лицевую панель и блок-диаграмму, содержащие компоненты, необходимые для моделирования получения данных с устройства
      - **Tutorial (Getting Started)** — открывает лицевую панель и блок-диаграмму, содержащие компоненты, необходимые для построения ВП, предназначенных для выполнения упражнений руководства Getting Started.
      - **User** – открывет лицевую панель и блок-диаграмму ВП, созданного ранее.
    - **Project** — открывает окно проекта в LabVIEW**.**
    - **Other Files –** позволяют создать классы, глобальные переменные, библиотеки и т.д.
- **Description** — отображает блок-диаграмму и описание выбранного из списка **Create New** шаблона ВП, в случае если шаблон имеет описание

Сохранение ВП

Выбрав из пункта главного меню **File** подпункт **Save, Save All** или **Save As,** можно сохранить ВП либо как отдельный файл, либо как группу из нескольких ВП в файл библиотеки ВП LabVIEW. Файл библиотеки ВП имеет расширение *.llb. National Instruments рекомендует сохранять ВП в

виде отдельных файлов, организованных в каталоги, особенно если над одним и тем же проектом работают несколько разработчиков.

LabVIEW использует диалоги загрузки и сохранения файлов, заданные по умолчанию. Эту функцию можно отключить с помощью пунктов главного меню **Tools»Options**, выбрав из выпадающего меню пункт **Miscellaneous**.

Перемещение ВП между платформами

Предусмотрена возможность портирования ВП из одной платформы в другую, например из MacOS в Windows. На новой платформе LabVIEW автоматически перекомпилирует ВП.

Поскольку ВП являются файлами, можно использовать любой способ или утилиты для перемещения файла ВП между платформами. Можно передавать ВП по сетям, использующим протокол **FTP**, **Z** и **XModem** протоколы. Такой способ передачи файлов устраняет необходимость в дополнительном программном обеспечении. Если ВП передается с использованием магнитных носителей, типа гибких дисков или внешнего жесткого диска, необходима универсальная программа средств передачи файлов следующего типа:

- **(Windows)** MacDisk и TransferPro для передачи файлов MacOS в формат PC и наоборот.
- **(MacOS)** DOS Mounter, MacLink и Apple File Exchange преобразует PC файлы в формат MacOS и наоборот.
- **(Sun)** PC File System (PCFS) преобразует PC файлы в формат Sun и наоборот.

Примечание. Некоторые специфические ВП не переносятся между платформами, такие как DDE (Динамический обмен данными) ВП, ActiveX ВП и AppleEvents.

Для получения дополнительной информации о перемещении ВП между платформами следует выбрать в главном меню пункт **Help»Search the LabVIEW Bookshelf** – *Porting and Localizing LabVIEW VIs***.**

C. Типы и проводники данных

В среде LabVIEW проводники данных используются для соединения многочисленных терминалов данных. Поля ввода/вывода должны быть совместимыми с типами данных, передаваемыми им по проводникам. Например, нельзя соединять поле вывода массива с полем ввода данных численного типа. Кроме того, характер соединения должен быть корректным. Проводники должны быть подсоединены лишь к одному источнику данных и, по крайней мере, к одному полю ввода данных. Например, нельзя соединять 2 элемента отображения. Компонентами, определяющими совместимость соединения, являются тип данных элемента управления и/или отображения и тип данных поля ввода/вывода.

Типы данных

В данном курсе используются следующие типы данных:

- **Numeric** (численный тип)
  - **Floating point** — число с плавающей запятой, отображается в виде оранжевых терминалов. Может быть представлено в виде **single** (32 bit), **double** (64-bit) или **extended** (128-bit) **precision** (с одиночной, двойной или расширенной точностью). Число с плавающей запятой может быть комплексным.
  - **Integer —** целочисленный тип, отображается в виде голубых терминалов. Возможны три представления целых чисел: 8, 16 и 32 бита. Один бит может использоваться для знака числа, если это число является знаковым целым.
- **Boolean —** логический тип, отображается в виде зеленых терминалов. Логический тип может принимать только два значения: 0 (FALSE) или 1 (TRUE).
- **String** — строковый тип, отображается в виде розовых терминалов. Строковый тип данных содержит текст в ASCII формате.
- **Path** — путь к файлу, отображается в виде терминалов. Путь к файлу близок строковому типу, однако, LabVIEW форматирует его, используя стандартный синтаксис для используемой платформы.
- **Array** — массивы включают типы данных составляющих элементов и принимают соответствующий им цвет.
- **Cluster** — кластеры включают различные типы данных. Кластерный тип данных отображается коричневым цветом, если все его элементы численные, если же элементы кластера являются данными различных типов, он отображается розовым.
- **Waveform** — сигнальный тип данных является кластером элементов, содержащим данные, начальное значение времени и интервал времени между измерениями.
- **Dynamic** — динамический тип, отображается в виде темно-синих терминалов. Кроме данных сигнала, динамический тип содержит дополнительную информацию, например, название сигнала или дату и время его получения. Большинство экспресс-ВП принимают и/или возвращают данные динамического типа. Данные динамического типа

можно направлять к любому элементу отображения или полю ввода, принимающему данные численного, логического или сигнального типа.

Проводники данных

Данные между объектами блок-диаграммы передаются по соединительным линиям – проводникам данных. Проводник данных аналогичен переменным в текстовых языках программирования. Каждый проводник данных имеет единственный источник данных, но может передавать их ко многим ВП и функциям. Проводники данных различаются цветом, стилем и толщиной линии, в зависимости от типа передаваемых данных. Примеры основных типов проводников данных представлены в таблице.

Тип проводника данных	Одно значение	Одномерный (1D) массив	Двумерный (2D) массив	Цвет
Численный				Оранжевый (с плавающей точкой), Голубой (целочисленный)
Логический				Зеленый
Строковый				Розовый

Автоматическое соединение объектов проводниками данных

В среде LabVIEW объекты соединяются проводниками данных после их помещения на блок-диаграмму. В автоматическом режиме среда LabVIEW подключает те поля ввода/вывода данных, которые наиболее совместимы, несовместимые поля остаются несоединенными.

Если выбранный объект помещается на блок-диаграмме недалеко от другого объекта, среда LabVIEW показывает пунктирные временные проводники данных, намечающие области возможного соединения. Следует обратить внимание, что при отпускании кнопки мыши LabVIEW автоматически подключает проводник данных к полю ввода/вывода данных, выбранного объекта.

Корректировка параметров автоматического подключения проводников осуществляется через пункты главного меню **Tools>>Options>>Block Diagram**.

Соединение объектов проводниками данных вручную

- Соединение объектов проводниками данных вручную производится с помощью инструмента СОЕДИНЕНИЕ. После наведения инструмента СОЕДИНЕНИЕ на поле ввода или вывода данных на экране появляется подсказка, которую можно использовать для уточнения места подключения проводника.

Упражнение 2-1. Преобразование °C в °F

Цель: Создать ВП

Ниже приведена последовательность действий для создания ВП, который будет преобразовывать значение температуры из градусов Цельсия в градусы Фаренгейта.

На иллюстрациях, демонстрирующих соединение, стрелка показывает, где произвести щелчок мышью, а число на стрелке – сколько раз необходимо щелкнуть.

Лицевая панель

1. Выберите пункт главного меню **File»New»VI**, чтобы открыть новую лицевую панель.

2. (Дополнительно) Выбрать пункт главного меню **Window»Tile Left and Right** для вывода на экран рядом друг с другом лицевой панели и блок-диаграммы.
3. Создайте цифровой элемент управления. Он будет использован для ввода значений температуры в °C.
   a. Выберите цифровой элемент управления в разделе палитры **Элементов** в подразделе **Controls»Modern»Numeric** (Числовые элементы). Для вывода на экран палитры **Controls** (Элементов) следует щелкнуть правой кнопкой мыши по рабочему пространству лицевой панели.

   b. Поместите цифровой элемент управления на лицевую панель.

   c. В поле собственной метки элемента управления напечатайте **«Град C»** и щелкните мышью в свободном пространстве лицевой панели или нажмите кнопку **Enter**, показанную слева, на инструментальной панели. Если сразу после создания элемента не присвоить имя его собственной метке, то LabVIEW присвоит имя, заданное по умолчанию. Собственная метка в любое время доступна для редактирования, оно производится с помощью инструмента ВВОД ТЕКСТА, показанного слева.

4. Создайте цифровой элемент отображения данных. Он будет использован для отображения значений температуры в °F.

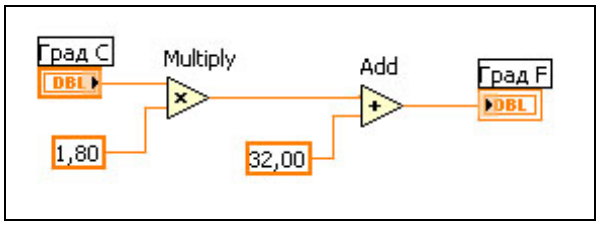
   a. Выберите цифровой элемент отображения в палитре **Элементов** в подразделе **Controls»Modern»Numeric** (Числовые элементы).
   b. Поместите элемент отображения данных на лицевую панель.
   c. В поле собственной метки элемента управления напечатайте «Град F» и щелкните мышью в свободном пространстве лицевой панели или нажмите кнопку **Enter**.

На блок-диаграмме LabVIEW создаст терминалы данных, соответствующие элементам управления и отображения. Терминалы данных представляют тип данных соответствующих элементов. Например, терминал данных DBL, показанный слева, представляет тип числовых данных двойной точности с плавающей запятой.

Внимание. Терминалы данных, соответствующие элементам управления, имеют более широкий обводной контур по сравнению с терминалами данных, соответствующими элементам отображения.

Блок-диаграмма

5. Перейдите на блок-диаграмму, выбрав пункты главного меню **Window» Show Diagram**.



6. Выберите функцию **Multiply** (Умножение) из палитры **Функций** в разделе **Functions»Programming»Numeric** (Арифметические функции). Поместите ее на блок-диаграмму. Для вывода на экран палитры **Functions** (Функций) следует щелкнуть правой кнопкой мыши в рабочем пространстве блок-диаграммы.

7. Выберите функцию **Add** (Сложение) из палитры **Функций** в разделе **Functions»Programming»Numeric** (Арифметические функции). Поместите ее на блок-диаграмму.

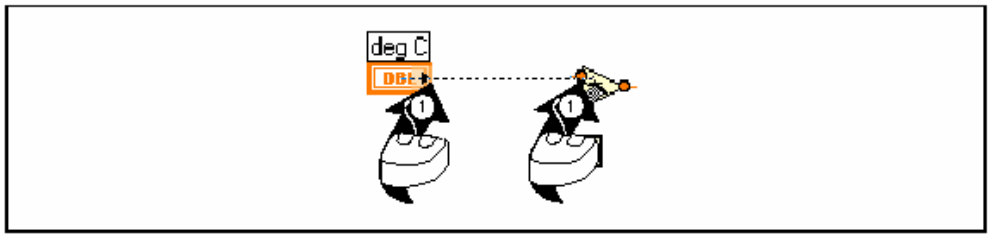
8. Выберите числовую константу из палитры **Функций** в разделе **Functions»Programming»Numeric** (Арифметические функции). Поместите две числовые константы на блок-диаграмму. После размещения числовой константы на блок-диаграмме поле ввода ее значений подсвечивается и готово для редактирования.

   - Одной константе присвойте значение 1,8 , другой 32,0.

- Если значение в константу не введено сразу после ее размещения на блок-диаграмме, следует использовать инструмент ВВОД ТЕКСТА.

9. Соедините объекты блок-диаграммы с помощью инструмента СОЕДИНЕНИЕ, показанного слева:

   - Для соединения двух объектов на блок-диаграмме инструментом СОЕДИНЕНИЕ следует нажать левой кнопкой мыши на одном из объектов и, не отпуская кнопку, перевести инструмент на другой объект, как показано на иллюстрации ниже. Последовательность соединения не имеет значения.



   - Проводник данных можно изгибать, закрепив его щелчком мыши и переместив курсор в перпендикулярном направлении. Нажав пробел, можно переключить направление движения проводника.
   - Для идентификации полей ввода/вывода данных узла или функции следует щелкнуть на выбранном объекте правой кнопкой мыши и в контекстном меню выбрать пункт **Visible Items»Terminals**. Чтобы вернуться к прежнему виду объекта, следует повторить ту же последовательность действий.
   - При наведении инструмента СОЕДИНЕНИЕ на поле ввода/вывода данных поле начинает мигать, показывая, что щелчок мыши подключит к нему проводник данных. При этом появляется всплывающая подсказка с указанием имени поля ввода/вывода.
   - Для отмены начала соединения следует нажать **<Esc>,** либо щелкнуть правой кнопкой мыши или щелкнуть инструментом СОЕДИНЕНИЕ на поле - источник соединения.

10. Перейдите на лицевую панель, выбрав в главном меню пункт **Window»Show Panel**.
11. Сохраните ВП, он будет использоваться позднее.
    a. Выберите пункт главного меню **File»Save**.
    b. Укажите каталог **c:\exercises\LV Basics I.**

Совет. Все ВП этого курса следует сохранять в каталоге **c:\exercises\LV Basics I.**

c. В диалоговом окне введите *Преобразование C в F (начало).vi*

d. Нажмите кнопку **Save**.

Запуск ВП

12. Введите число в элемент управления и запустите ВП.

a. Для ввода числа в элемент управления следует использовать инструмент УПРАВЛЕНИЕ, показанный слева, или инструмент ВВОД ТЕКСТА.

b. Нажмите кнопку **Run**, показанную слева, чтобы запустить ВП.

c. Введите несколько разных значений температуры и запустите ВП.

13. Закройте ВП, выбрав пункт главного меню **File»Close**.

Конец упражнения 2-1

D. Редактирование ВП

Существует несколько методов редактирования объектов лицевой панели и блок-диаграммы.

Создание объектов

В дополнение к созданию объектов лицевой панели с помощью палитры **Элементов (Controls)** предусмотрена возможность создания элементов управления и отображения данных, констант по щелчку правой копкой мыши на узле. Для этого в контекстном меню следует выбрать пункт **Create**

- **Constant** — создание констант, отображающихся только на блок-диаграмме.
- **Control** — создание элемента управления на лицевой панели ВП.
- **Indicator —** создание элемента отображения данных на лицевой панели.

Выделение объектов

Выделение объектов на лицевой панели и блок-диаграмме производится с помощью инструмента ПЕРЕМЕЩЕНИЕ.

Когда объект выделен, его окружает пунктирная линия. Для выбора нескольких объектов, следует во время их выделения нажать и удерживать клавишу **<Shift>**.

Можно также выделить несколько объектов, щелкнув мышью в свободном пространстве и обведя их курсором.

Перемещение объектов

Перемещение объектов осуществляется при помощи инструмента ПЕРЕМЕЩЕНИЕ. Перемещать объекты можно также при помощи стрелок на клавиатуре. Для перемещения объекта с шагом в несколько пикселей в момент перемещения следует нажать и удерживать клавишу **<Shift>.**

Можно ограничить направление движения выбранного объекта только по горизонтали или только по вертикали, если в момент его перемещения удерживать клавишу **<Shift>**. Первоначально выбранное направление движения (горизонтальное или вертикальное) определяет направление перемещение объекта.

Удаление объектов

Чтобы удалить объект, следует выделить его с помощью инструмента ПЕРЕМЕЩЕНИЕ, после чего нажать на клавиатуре клавишу **<Delete>** или выбрать пункты главного меню **Edit»Clear**.

Отмена и восстановление действий

Если в процессе редактирования ВП была допущена ошибка, можно отменить или восстановить действия, выбрав **Undo** (Отменить) или **Redo** (Восстановить) в пункте главного меню **Edit** (Редактирование). Установка количества действий, подлежащих отмене или восстановлению, производится в пункте главного меню **Tools»Options**. Для этого из выпадающего меню следует выбрать раздел **Block Diagram**. Установка небольшого числа повторений сохраняет ресурсы памяти компьютера.

Копирование объектов

Большинство объектов можно копировать, перемещая выделенный объект и одновременно удерживая клавишу **<Ctrl>.**

(MacOS) Нажать кнопку **<Option>**. **(Sun)** Нажать кнопку **<Meta>. (Linux)** Нажать кнопку **<Alt>.**

После переноса выбранного объекта на новое место, отпускается сначала кнопка мыши, а затем клавиша **<Ctrl>.** В этом месте появляется копия объекта, а первоначальный объект остается на старом месте. Этот процесс называется копированием либо клонированием.

Можно копировать объекты и стандартным способом, выбирая пункты главного меню **Edit>>Copy** и затем **Edit>>Paste.**

Метки объектов

Метки используются для идентификации объектов. Среда LabVIEW имеет два вида меток: свободные и собственные. Собственные метки принадлежат объекту, описывают только его и двигаются вместе с ним. Собственную метку можно перемещать независимо от объекта, но при перемещении объекта метка перемещается вместе с ним. Свободные метки не принадлежат объектам, их можно создавать, перемещать, вращать или удалять независимо. Они используются для описания объектов, ввода комментариев на лицевой панели и блок-диаграмме.

Для создания свободной метки используется инструмент ВВОД ТЕКСТА. Выбрав этот инструмент, необходимо щелкнуть в свободном пространстве одной из панелей и ввести текст. После ввода текста метки поместить курсор в пространство вне метки или нажать кнопку **<Enter>** на инструментальной панели.

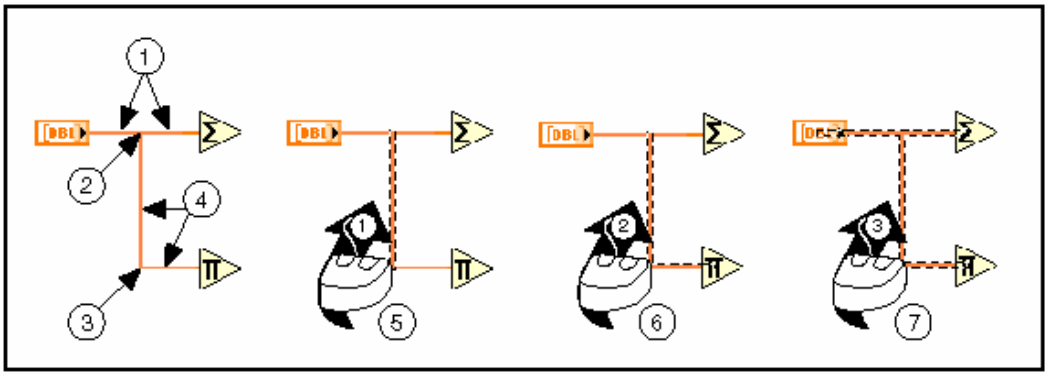
Совет. По умолчанию нажатие на клавиатуре клавиши **<Enter>** добавляет новую строку. Чтобы закончить ввод текста с клавиатуры, следует нажать **<Shift-Enter>.** Можно закончить ввод текста с клавиатуры нажатием клавиши **<Enter>,** для этого в пункте главного меню следует выбрать **Tools»Options**, далее, в выпадающем меню найти **Front Panel** и отметить пункт **End text entry with Return key**.

Специальный вид свободной метки используется для ввода комментариев на блок-диаграмму. Эта свободная метка находится на палитре **Functions»Programming»Structures»Decorations.**

Выделение и удаление проводников данных

Сегмент проводника данных – это отдельная горизонтальная или вертикальная его часть. Место соединения двух сегментов – излом проводника данных. Точка, в которой встречаются два, три или четыре проводника данных называется точкой соединения.

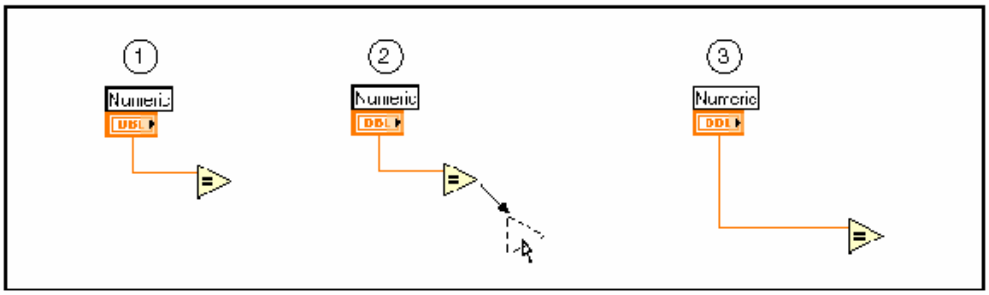
Проводник данных содержит все сегменты между точками соединения, между терминалом данных и точкой соединения, между терминалами данных, если нет точек соединений. Для выделения сегмента используется инструмент ПЕРЕМЕЩЕНИЕ. Двойной щелчок мыши выделяет проводник данных, тройной щелчок – выделяет множество проводников данных:



1 Сегмент
2 Точка соединения
3 Излом проводника данных
4 Проводник данных
5 Выделенный сегмент
6 Выделенный проводник данных
7 Выделенное множество проводников данных

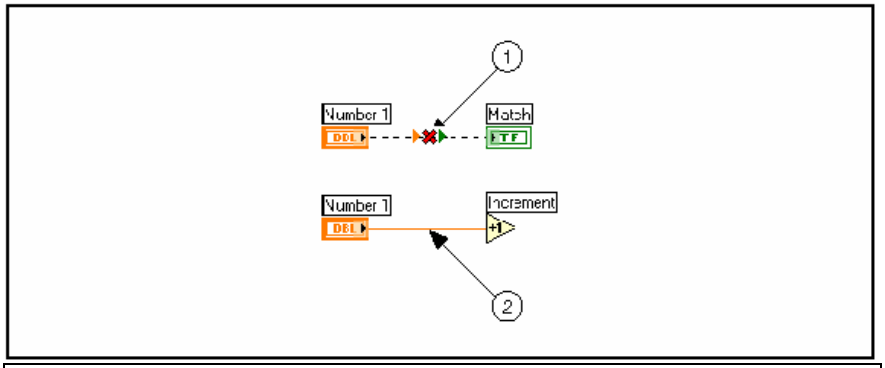
Автомасштабирование проводников данных

Как показано на иллюстрации, перемещение объектов не приводит к нарушению проводника данных.



Разорванные проводники данных

Разорванный проводник данных выглядит, как черная штриховая линия с красным крестом посередине, как показано ниже. Разрыв проводников данных происходит по причинам разного рода. Например, при попытке соединения объектов с несовместимыми типами данных:



1 Нарушенный проводник 2 Правильный проводник

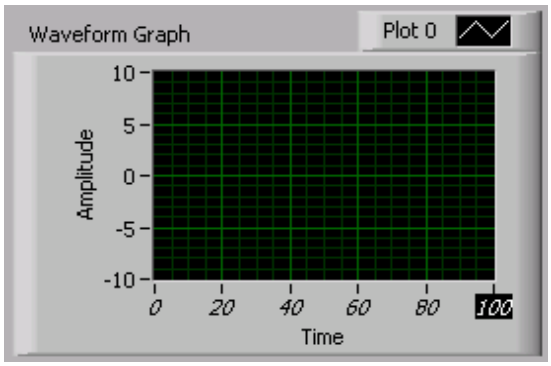
Описание причины разрыва проводника данных появляется в окне всплывающей подсказки после наведения на проводник инструмента СОЕДИНЕНИЕ. Тройной щелчок инструментом ПЕРЕМЕЩЕНИЕ на проводнике и последующее нажатие клавиши **<Delete>** удаляет выделенный проводник. Удаление всех разорванных проводников производится через пункт главного меню **Edit»Remove Broken Wires**.

Внимание. Использование пункта главного меню **Remove Broken Wires** требует определенной осторожности. Иногда проводник является разорванным потому, что еще не закончено создание блок-диаграммы.

Редактирование текста (изменение шрифта, стиля и размера)

Выбрав пункт меню **Text Settings** на инструментальной панели, можно изменить шрифт, стиль, размер и провести выравнивание любого текста внутри меток или на дисплеях элементов управления и отображения.

На некоторых элементах управления и отображения данных, текст может быть помещен более чем в одном месте, например оси графиков. В этом случае текст в каждом поле можно изменять независимо. Текст выделяется инструментом ВВОД ТЕКСТА, как показано на рисунке, и на инструментальной панели выбирается пункт меню **Text Settings**.



Изменение размеров объектов

Большинство объектов лицевой панели допускают изменение размеров. Чтобы подготовить объект к изменению размера, необходимо навести на него инструмент ПЕРЕМЕЩЕНИЕ. По углам объекта появляются маркеры, показанные слева. Затем курсор следует установить на один из маркеров и, удерживая нажатой левую кнопку мыши, переместить маркер, размер шрифта при этом не меняется. Промежуточные границы изменяемого размера обозначаются штриховой линией. Когда нужный размер элемента достигнут, кнопку мыши следует отпустить. Удержание клавиши **<Shift>** во время перемещения маркеров сохраняет пропорции объекта.

Можно изменять размеры и объектов блок-диаграммы, таких как структуры и константы.

Выравнивание и распределение объектов в пространстве

Выравнивание группы объектов по оси производится с помощью опций в пункте инструментальной панели **Align Objects**. Для равномерного распределения объектов в пространстве следует воспользоваться пунктом **Distribute Objects**.

Установка порядка размещения объектов, объединение объектов в группу и закрепление местоположения объектов на рабочем пространстве лицевой панели

В случае, когда объекты перекрывают друг друга, можно установить порядок размещения объектов - один впереди другого. Для этого объект следует выделить с помощью инструмента ПЕРЕМЕЩЕНИЕ и в пункте меню **Reorder** инструментальной панели выбрать необходимые установки: **Move Forward** (Поместить на передний план), **Move Backward** (Поместить на задний план), **Move To Front** (Передвинуть вперед), **Move To Back** (Передвинуть назад).

Для объединения объектов в группу и закрепления их местоположения на рабочем пространстве лицевой панели следует выбрать необходимые установки в пункте меню **Reorder** инструментальной панели: **Group** (Группировать), **Ungroup** (Разгруппировать), **Lock** (Блокировать), **Unlock** (Разблокировать).

Приведение нескольких объектов к одному размеру

Приведение нескольких объектов к одному виду производится с помощью выпадающего меню **Resize Objects (**Изменение размеров объектов). Предусмотрена возможность изменения размера всех выбранных объектов по ширине или высоте до ширины/высоты наименьшего или наибольшего объекта, также имеется возможность задать размер всех выбранных объектов в пикселях.

Отдельные объекты допускают изменения размера лишь по вертикали или горизонтали, например, числовые элементы управления и отображения;

некоторые объекты сохраняют пропорции при изменении размера. Например, если среди объектов, выбранных для изменения размера по высоте, присутствует числовая константа, LabVIEW не изменит ее размер, изменив размер остальных объектов, допускающих изменение размера.

Копирование объектов между ВП или между другими приложениями

Копировать и вставлять объекты из одного ВП в другой можно выбором пунктов главного меню **Edit»Copy** и затем **Edit»Paste**. Возможно копирование изображения или текста из других приложений и их использование на лицевой панели или блок-диаграмме. Если оба ВП открыты, можно копировать выбранные объекты, перемещая их с одного ВП на другой.

Окрашивание объектов

Можно изменять цвет большинства объектов ВП, но не всех. Например, терминалы данных и проводники данных блок-диаграммы используют только определенные цвета, соответствующие типу представленных данных.

Изменение цвета объекта или фона рабочего пространства производится с помощью инструмента РАСКРАШИВАНИЕ. Для этого следует щелкнуть правой кнопкой мыши на выбранном элементе или рабочем пространстве любой из панелей. Можно изменить заданные по умолчанию цвета большинства объектов, выбирая пункты меню **Tools>>Options** и затем **Colors.**

Можно также сделать объект прозрачным, выбрав **T** в меню **Colors.**

Упражнение 2-2. ВП Редактирование

Цель: Отредактировать ВП

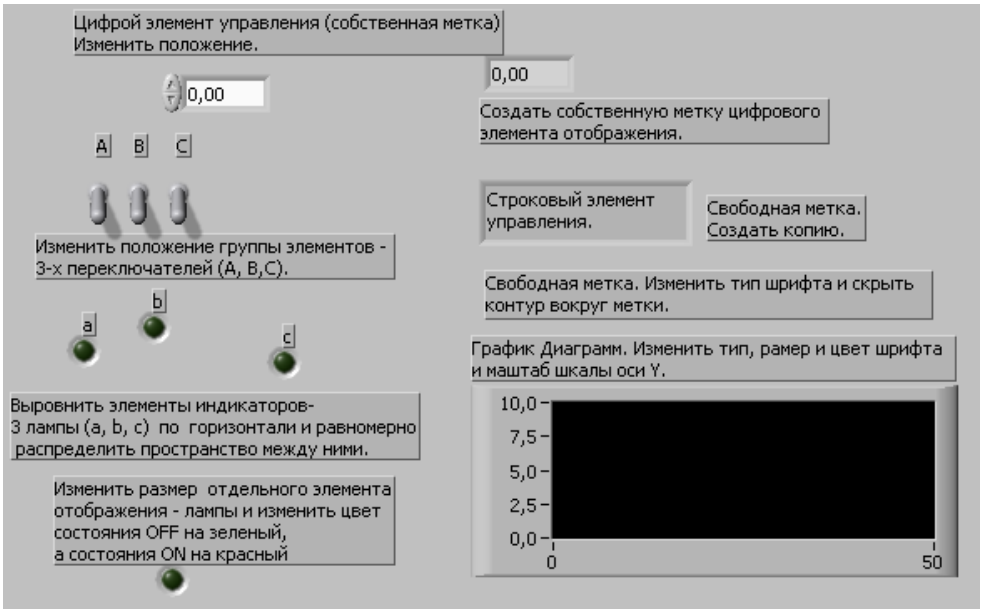
Ниже приведена последовательность действий для изменения существующего ВП Редактирование таким образом, чтобы его лицевая панель выглядела, как показано ниже. Также необходимо соединенить объекты на блок-диаграмме для приведения ВП в рабочее состояние.

Примечание. В случае совершения ошибки можно воспользоваться пунктом меню **Edit>>Undo**

Лицевая панель

1. Выберите пункт главного меню **File>>Open**, укажите папку **c:\exercises\LV basics I** и откройте *Редактирование.vi*

   В случае если все ВП закрыты, следует воспользоваться кнопкой **Open VI** диалогового окна **LabVIEW**.



2. Измените положение элемента **Цифровой элемент управления**.

   a. С помощью инструмента ПЕРЕМЕЩЕНИЕ, показанного слева, выделите элемент и переместите его на новое место. Собственная метка элемента переместится вместе с ним.

   b. Щелкните левой кнопкой мыши по свободному пространству лицевой панели, чтобы снять выделение с элемента.

   c. Выделите собственную метку элемента и переместите ее на новое место. Элемент останется неподвижным. Собственную метку элемента можно перемещать независимо от элемента. Собственная метка следует за своим элементом в случае его

перемещения.

3. Измените положение трех вертикальных переключателей одновременно.

   a. С помощью инструмента ПЕРЕМЕЩЕНИЕ выделите область лицевой панели, охватив все три переключателя.

   b. Нажмите мышью на одном из них и переместите на новое место. При этом переместятся все три переключателя.

4. Выровняйте три светодиода по горизонтали, равномерно распределите пространство между ними, сгруппируйте и закрепите их местоположение на лицевой панели.

   a. С помощью инструмента ПЕРЕМЕЩЕНИЕ выделите область лицевой панели, охватив все три светодиода.

   b. Чтобы выровнять элементы по горизонтали, в пункте меню инструментальной панели **Align Objects** выберите опцию **Vertical Centers**, показанную слева.

   c. Чтобы равномерно распределить пространство между элементами, в пункте меню инструментальной панели **Distribute Objects** выберите опцию **Horizontal Centers**, показанную слева.

   d. Чтобы сгруппировать элементы, в пункте меню инструментальной панели **Reorder** выберите опцию **Group**.

   e. Чтобы закрепить местоположение элементов на лицевой панели, в пункте меню инструментальной панели **Reorder** выберите опцию **Lock**.

   f. Теперь попробуйте переместить один из светодиодов. Следует обратить внимание, что светодиоды выделяются как группа и переместить их невозможно, пока их местоположение закреплено.

5. Измените размер отстоящего круглого светодиода.

   a. Наведите на светодиод инструмент ПЕРЕМЕЩЕНИЕ. По краям светодиода появятся круглые маркеры.

   b. Захватите курсором маркер и измените размер светодиода. Если при этом нажать и удерживать клавишу <**Shift**>, светодиод будет изменять свои размеры, сохраняя пропорции.

6. Измените цвет этого светодиода.

   a. По умолчанию цвет светодиода в режиме **OFF** является темно-зеленым (значение FALSE)**.**
   С помощью инструмента УПРАВЛЕНИЕ, показанного слева, переведите светодиод в режим ON. Его цвет изменится на светло-зеленый (значение TRUE).

   b. С помощью инструмента ОКРАШИВАНИЕ, показанного слева, щелкните правой кнопкой мыши по светодиоду для вывода цветовой палитры.

c. Для режима **ON** установите красный цвет.

7. Для цифрового элемента управления, расположенного в правой верхней части лицевой панели, выведите собственную метку.

a. С помощью инструмента ВВОД ТЕКСТА, показанного слева, щелкните правой кнопкой мыши на элементе и выберите из контекстного меню пункт **Visible Items»Label**.

Появится маленькое окно ввода текста с курсором в левой части, готовое к вводу текста.

b. Напечатайте **Цифровой элемент отображения данных**.

c. Щелкните мышью в пространстве вне метки или нажмите кнопку **Enter**, показанную слева, для завершения ввода текста.

8. Удалите строковый элемент управления.

a. С помощью инструмента ПЕРЕМЕЩЕНИЕ выделите строковый элемент управления.

b. Нажмите клавишу **<Delete>** или выберите пункт главного меню **Edit»Clear.**

9. Размножьте свободную метку.

a. Нажмите и удерживайте клавишу **<Ctrl>,** с помощью инструмента ПЕРЕМЕЩЕНИЕ передвиньте метку.

(MacOS) Нажмите клавишу **<Option>. (Sun)** Нажмите клавишу **<Meta>.** (**Linux)** Нажмите клавишу **<Alt>.**

b. Поместите копию метки на новое место.

10. Измените шрифт свободной метки и скройте поле вокруг нее.

a. С помощью инструмента ПЕРЕМЕЩЕНИЕ выделите свободную метку.

b. Измените текст, выбрав показанный слева пункт **Text Settings** инструментальной панели.

c. С помощью инструмента ОКРАШИВАНИЕ щелкните правой кнопкой мыши на метке и в цветовой палитре выберите **T.**

11. Измените текст и цвет оси Y.

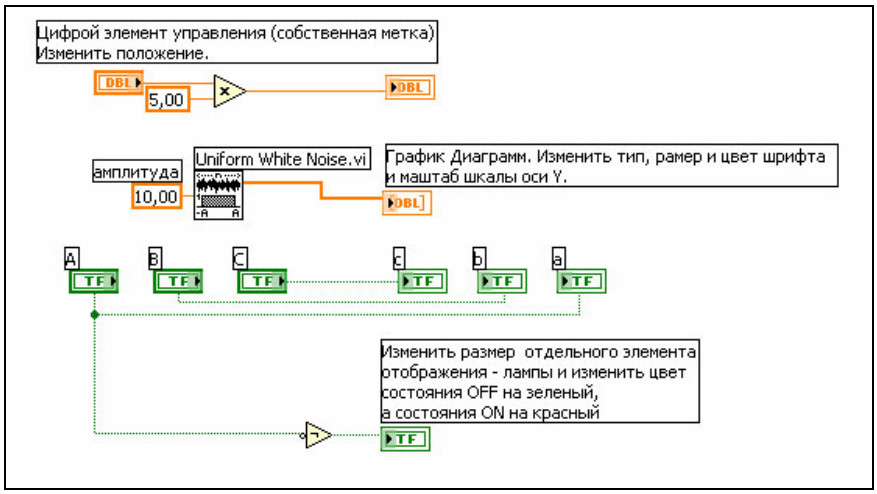
a. С помощью инструмента ВВОД ТЕКСТА выделите **10,0** по оси Y.

b. Выбрав пункт инструментальной панели **Text Settings**, измените размер и цвет текста.

12. Двойным щелчком левой кнопкой мыши выделите **0,0** и введите значение **-10.0**, чтобы изменить диапазон оси Y.

Блок-диаграмма

13. Перейдите на блок-диаграмму, выбрав пункт главного меню **Window»Show Diagram**. Соедините терминалы данных блок-диаграммы, как показано ниже:



- Функция **Умножение (Multiply)** умножает числовую константу «5.00» на значение, введенное в числовой элемент управления.

- Узел **Uniform White Noise VI** генерирует белый шум в диапазоне [–*a*:*a*], где *a* – абсолютное значение амплитуды шума (**amplitude**) «10.00», и передает данные на график осциллограмм.

- Функция **Not** (Логическое нет) инвертирует значение логического выключателя **A** и передает значение на светодиод.

14. Для создания константы щелкните правой кнопкой мыши по нижнему полю ввода данных функции **Multiply** (Умножение) и выберите из контекстного меню пункт **Create>>Constant**.

    - Введите значение «5.0» в поле ввода текста и нажмите кнопку **Enter** на инструментальной панели.

15. Используя инструмент СОЕДИНЕНИЕ, соедините проводниками данных объекты блок-диаграммы. Для этого:

    - Войдите в пункт главного меню **Help»Show Context Help** и выведите на экран окно **Context Help** (контекстной справки). Окно **Context Help** (контекстной справки) поможет определить поля ввода данных, необходимые для соединения. Обязательные для соединения поля выделены жирным текстом, рекомендуемые для соединения поля обозначены обычным текстом, а дополнительные (необязательные) помечены светло-серым.

    - Для идентификации полей ввода/вывода данных следует

щелкнуть правой кнопкой мыши по терминалу функции, и в контекстном меню выбрать пункт **Visible Items»Terminal**. Когда проводники данных подключены, войдите в пункт контекстного меню **Visible Items»Terminal,** чтобы вернуть терминалу функции прежний вид.

- Для добавления ответвления к проводнику данных, щелкните в выбранном месте проводника.
- Для отмены начала соединения нажмите клавишу **<Esc>**, щелкните правой кнопкой мыши или нажмите на источник соединения.

16. Выберите пункт главного меню **File»Save**, чтобы сохранить ВП.
17. Перейдите на лицевую панель, выбрав пункт главного меню **Window»Show Panel**.
18. С помощью инструмента УПРАВЛЕНИЕ измените значения элементов управления.
19. Нажмите кнопку **Run** на инструментальной панели, чтобы запустить ВП.
20. Выберите пункт главного меню **File»Close**, чтобы закрыть ВП.

Конец упражнения 2-2

E. Отладка ВП

Если ВП не запускается, это означает, что он не готов к работе. В процессе создания или редактирования ВП кнопка **Run** принимает вид разорванной стрелки, как показано слева. Если после завершения редактирования блок-диаграммы стрелка все еще имеет разорванный вид, то ВП работать не будет.

Поиск ошибок

Нажмите кнопку **Run** или выберите пункт главного меню **Windows»Show Error List**, чтобы вывести на экран окно **Список ошибок**, в котором перечислены все допущенные ошибки. После двойного щелчка левой кнопкой мыши на описании ошибки выделится объект, содержащий эту ошибку.

Режим анимации выполнения ВП

Режим анимации выполнения блок-диаграммы активируется щелчком правой кнопки мыши по кнопке **Highlight Execution**, показанной слева. После нажатия кнопки «лампочка» загорится – режим активирован. Выполнение ВП в этом режиме сопровождается подсветкой движения данных по блок-диаграмме от одного узла к другому. При этом числовые значения передаваемых данных будут отображаться на экране в виде всплывающих окон. Этот режим используется для пошаговой отладки ВП и наблюдения за выполнением блок–диаграммы.

Примечание. Режим анимации замедляет скорость выполнения ВП.

Режим пошаговой отладки ВП

Режим пошаговой отладки ВП используется для просмотра выполнения ВП на блок-диаграмме. Активация пошагового режима осуществляется нажатием кнопок **Step Over** или **Step Into** на инструментальной панели. Чтобы увидеть подсказку, следует поместить курсор поверх кнопок **Step Over, Step Into** или **Step Out.** Подсказка описывает событие, которое последует после нажатия этой кнопки. Пошаговый режим можно использовать и для просмотра выполнения подпрограммы ВП.

При использовании пошагового режима отладки ВП в режиме анимации на иконке подпрограммы ВП появится зеленая стрелка, как показано слева. Зеленая стрелка показывает, что данная подпрограмма ВП в данный момент времени выполняется.

Отладочные индикаторы

Инструмент УСТАНОВКА ОТЛАДОЧНЫХ ИНДИКАТОРОВ, показанный слева, предназначен для проверки промежуточного значения данных в проводнике данных в процессе выполнения ВП. В режиме пошагового выполнения или при остановке в контрольной точке с помощью отладочных индикаторов можно визуализировать значения данных в проводнике или поле ввода/вывода узла данных, если узел уже получил свое значение.

Можно установить несколько локальных отладочных индикаторов для одновременного наблюдения за данными в разных точках блок-диаграммы. Чтобы создать локальный отладочный индикатор, следует щелкнуть правой кнопкой мыши на выбранном проводнике данных и выбрать в контекстном меню пункт **Custom Probe**.

При отладке ВП Вы можете сохранять в памяти значения данных, прошедших по проводникам. То есть, когда Вы помещаете ОТЛАДОЧНЫЙ ИНДИКАТОР на блок-диаграмму, в нем сразу же будет отображаться значение данных прошедших по проводу в момент последнего выполнения. Чтобы сделать функцию доступной, нажмите на панели инструментов в верхней части окна блок-диаграммы кнопку **Retain Wire Values**. После этого, LabVIEW будет сохранять значения данных в каждой точке потока

Примечание. Вы должны успешно запустить ВП по крайней мере один раз, для того, чтобы на всех проводах были какие-то данные.

Контрольные точки

Инструмент ВВОД КОНТРОЛЬНОЙ ТОЧКИ, показанный слева, предназначен для размещения контрольных точек в узлах или проводниках данных блок-диаграммы. В месте установки контрольной точки в момент прохождения через нее данных возникает пауза в выполнении программы. Когда ВП приостановил свое выполнение в контрольной точке, LabVIEW подсвечивает узел или проводник данных в месте установки контрольной точки. LabVIEW обводит красной границей узел и блок-диаграмму и отмечает красным маркером проводник данных. После наведения курсора на контрольную точку, черное поле инструмента ВВОД КОНТРОЛЬНОЙ ТОЧКИ становится белым. Для удаления существующей контрольной точки по ней следует щелкнуть инструментом ВВОД КОНТРОЛЬНОЙ ТОЧКИ.

Упражнение 2-3. Отладка ВП

Цель: Приобретение практических навыков отладки ВП

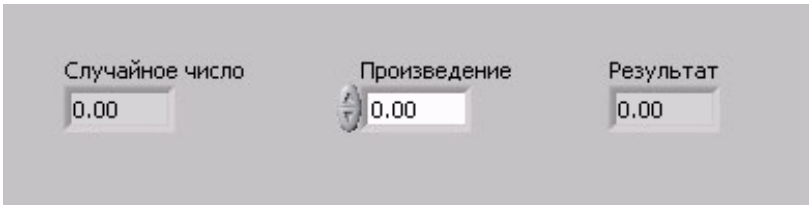
Ниже приведены последовательность действий для загрузки содержащего ошибки ВП, использование режимов анимации и пошаговой отладки для исправления ошибок.

Лицевая панель

1. Выберите пункт главного меню **File»Open** и перейдите в каталог c:\exercises\LV Basics I, чтобы открыть *Отладка ВП (Главная).vi*

   Если все ВП закрыты, следует воспользоваться кнопкой **Open VI** в диалоговом окне **LabVIEW**.

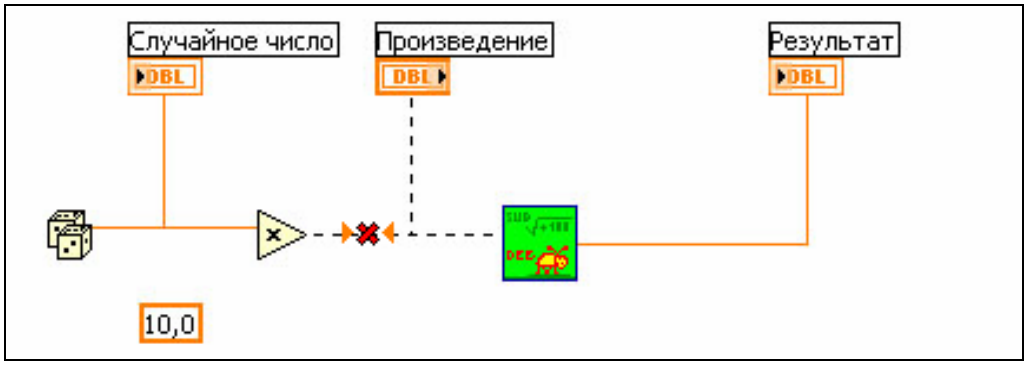
   Появится следующая лицевая панель:



Кнопка **Run**, показанная слева, имеет вид разорванной стрелки. Это говорит о том, что ВП не может работать.

Блок-диаграмма

2. Выберите пункт главного меню **Window»Show Diagram**, чтобы перейти на блок-диаграмму.



- Функция **Random Number (0-1)** (Генератор случайных чисел) - генерирует случайные числа от 0 до 1.

- Функция **Multiply** (Умножение) - умножает случайное число на 10,0.

- В этой числовой константе введено значение, на которое

умножается случайное число.

- ВП *Отладка ВП (Подпрограмма)* - добавляет 100.0 и вычисляет квадратный корень из полученного значения.

3. Найдите все ошибки.
   a. Нажмите кнопку **Run**. В окне **Error list** (Список ошибок) перечислены все допущенные ошибки.
   b. Для получения информации об ошибках нажмите на описание каждой из них.
   c. Нажмите кнопку **Show Error,** чтобы перейти на место текущей ошибки.
   d. Подробная информация об ошибке находится в окне **Details.**
4. Выберите пункт главного меню **File»Save**, чтобы сохранить ВП.
5. Перейдите на лицевую панель, выбрав пункт главного меню **Window»Show Panel**.

Запуск ВП

6. Несколько раз запустите ВП, нажимая кнопу **Run**.
7. Выберите пункт главного меню **Window»Show Diagram**, чтобы перейти на блок-диаграмму
8. Включите режим анимации выполнения ВП.

   a. Нажмите кнопку **Highlight Execution**, показанную слева, чтобы включить режим анимации выполнения.

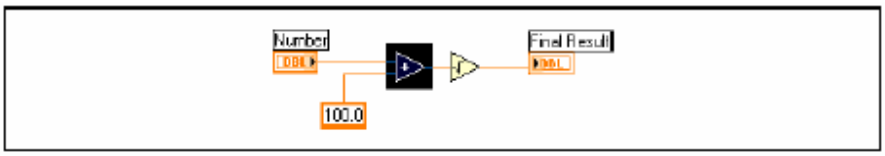
   b. Нажмите кнопку **Step Into,** показанную слева, чтобы включить режим пошаговой отладки. Режим анимации показывает передвижение данных на блок-диаграмме от узла к узлу с помощью подвижных точек на проводниках данных. Узлы мигают, показывая готовность к выполнению.

   c. Нажимать кнопку **Step Over**, показанную слева, следует после каждого узла для передвижения по всей блок-диаграмме. Каждый раз после нажатия кнопки **Step Over**, узел выполняется и наступает пауза перед выполнением следующего, готового к выполнению узла. Во время выполнения ВП на лицевой панели отображаются данные. ВП генерирует случайные числа и умножает их на 10,0. Подпрограмма ВП прибавляет 100,0 и вычисляет из результата корень квадратный.

   d. Когда начнет мигать контур блок-диаграммы, нажмите кнопку **Step Out**, показанную слева, чтобы выключить режим пошаговой отладки ВП *Отладка ВП (Главная).vi*

9. В режиме пошаговой отладки просмотрите подпрограмму ВП.
   a. Нажмите кнопку **Step Into,** чтобы включить режим пошаговой отладки.

b. Когда ВП *Отладка ВП (подпрограмма).vi* начнет мигать, нажмите кнопку **Step Into**. Появится следующая блок-диаграмма:



c. Перейдите на блок-диаграмму ВП *Отладка ВП (Главная).vi*, нажав на нее.

Зеленая стрелка на иконке ВП *Отладка ВП (Главная.)vi*, показанная слева, указывает на режим пошаговой отладки.

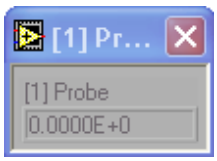
d. Перейдите на блок-диаграмму ВП *Отладка ВП (подпрограмма) .vi*

e. Дважды нажмите кнопку **Step Out** для завершения режима пошаговой отладки для подпрограммы ВП. Блок-диаграмма ВП *Отладка ВП (Главная).vi* останется активной.

f. Нажмите кнопку **Step Out** для завершения режима пошаговой отладки.

10. С помощью отладочных индикаторов посмотрите текущее значение данных в проводнике.

a. Инструментом УСТАНОВКА ОТЛАДОЧНЫХ ИНДИКАТОРОВ, показанным слева, нажмите на любом объекте. Появится следующее окно.



Номер в заголовке окна **Probe** соответствует номеру отладочного индикатора, поставленного на блок-диаграмме.

b. Снова включите режим пошаговой отладки. Окно **Probe** покажет данные по мере их прохождения через отладочный индикатор.

11. Установите контрольные точки для приостановки выполнения ВП в этом месте.

a. Инструментом ВВОД КОНТРОЛЬНОЙ ТОЧКИ щелкните на узле или проводнике данных. Щелчок по рабочему пространству блок-диаграммы означает остановку выполнения в самом начале выполнения ВП.

b. Нажмите кнопку **Run,** чтобы запустить ВП. ВП остановится на контрольной точке.

c. Нажмите кнопку **Continue**, показанную слева, чтобы продолжить выполнение ВП.

d. Инструментом ВВОД КОНТРОЛЬНОЙ ТОЧКИ щелкните по установленной контрольной точке, чтобы удалить ее.

12. Нажмите кнопку **Highlight Execution**, чтобы отключить режим анимации выполнения ВП.
13. Войдите в пункт главного меню **File»Close**, чтобы закрыть ВП и все открытые окна.

Конец упражнения 2-3

Краткое изложение пройденного материала, советы и секреты

- Лицевая панель создается с помощью элементов управления и отображения данных, которые являются интерактивными полями ввода и выводы данных ВП соответственно.
- Терминалы данных элементов управления имеют более широкий обводной контур, чем терминалы данных элементов отображения. Чтобы изменить терминал данных элемента управления на терминал данных элемента отображения и обратно, необходимо на выбранном объекте щелкнуть правой кнопкой мыши и в контекстном меню указать **Change to Indicator** или **Change to Control**.
- Блок-диаграмма состоит из узлов, терминалов данных и проводников.
- Инструмент УПРАВЛЕНИЕ используется для выбора конфигурации элементов лицевой панели. Инструмент ПЕРЕМЕЩЕНИЕ используется для выделения, перемещения и изменения размеров объектов. Инструмент СОЕДИНЕНИЕ используется для соединения объектов блок-диаграммы проводниками данных.
- Кнопка **Search** на палитрах **Controls** (Элементов) и **Functions** (Функций) предназначена для поиска элементов, ВП и функций.
- Кнопка **Run** в виде разорванной стрелки появляется на инструментальной панели, когда ВП не готов к работе. После нажатия кнопки **Run** на экран выводится окно **Error list** (Список ошибок), в котором перечислены все ошибки.
- Режимы анимации выполнения ВП и пошаговой отладки, отладочные индикаторы и контрольные точки предназначены для отладки ВП в режиме просмотра потока данных на блок-диаграмме.

Советы и секреты

В большинстве советов и секретов рекомендуется использовать клавишу <Ctrl>. **(MacOS)** Клавиша <Option> вместо <Ctrl>. **(Sun)** Клавиша <Meta>. **(Linux)** Клавиша <Alt>.

Действие

- Часто используемые пункты меню имеют эквивалентные «горячие» клавиши.
- Например, чтобы сохранить ВП, необходимо войти в пункт главного меню **File»Save** или нажать клавиши **<Ctrl-S>** на клавиатуре. Основные «горячие» клавиши:

– **<Ctrl-R>** Запуск ВП.

– **<Ctrl-E>** Переключение между лицевой панелью и блок- диаграммой.

– **<Ctrl-H>** Вывод на экран и скрытие окна **Context Help** (контекстной справки).

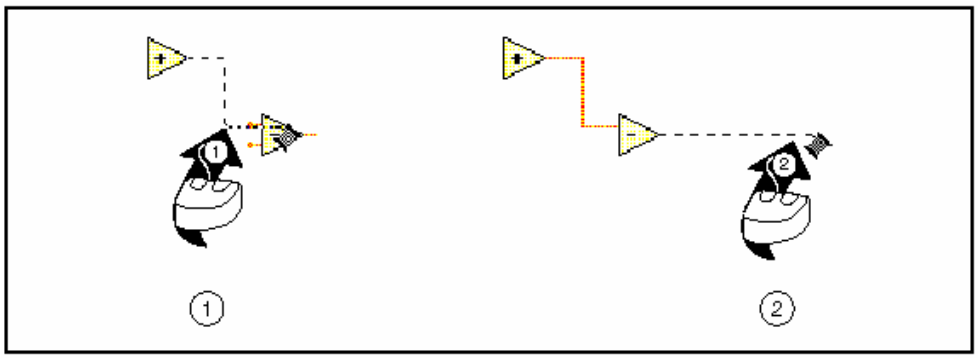
– **<Ctrl-B>** Удаление всех разорванных проводников данных.

– **<Ctrl-F>** Поиск ВП, глобальных переменных, функций, текста или других объектов, находящихся в памяти ВП.

- Для выбора инструментов на палитре **Tools** (Инструментов) используется клавиша **<Tab>.**
  На блок-диаграмме, для переключения между инструментом ПЕРЕМЕЩЕНИЕ и инструментом СОЕДИНЕНИЕ, а на лицевой панели – между инструментом ПЕРЕМЕЩЕНИЕ и инструментом УПРАВЛЕНИЕ используется клавиша пробел.
- Для быстрого изменения значений элементов управления инструментом УПРАВЛЕНИЕ или ВВОД ТЕКСТА курсор помещается в поле элемента и, удерживая нажатой клавишу **<Shift>,** нажимается одна из стрелок приращения значений.
- Можно отключить средства отладки и тем самым уменьшить требования к оперативной памяти. Для этого необходимо выбрать пункт главного меню **File»VI Properties** и войти в раздел **Execution** выпадающего меню, после чего снять метку в **Allow Debugging**.

Проводники данных

- Выберите пункт главного меню **Help»Show Context Help,** чтобы вывести на экран окно **Context Help** (Контекстной справки). Окно **Context Help** (контекстной справки) поможет определить поля ввода/вывода данных, обязательные для соединения. Поля ввода/вывода данных, рекомендованные для соединения, представлены обычным текстом, дополнительные (необязательные) поля выделены серым цветом.
- Пробел переключает направление движения проводника данных.
- Проводник данных можно изогнуть, зафиксировав излом щелчком мыши. Фиксация и разрыв проводника производится двойным щелчком мыши:



1	Фиксация излома проводника	2	Фиксация и разрыв проводника двойным щелчком мыши

- Перемещение объекта на один пиксель проводится стрелками на клавиатуре. Для быстрого перемещения объекта сразу на несколько пикселей необходимо нажать и удерживать клавишу **<Shift>.**
- Для отмены начала соединения следует нажать **<Esc>,** щелкнуть правой кнопкой мыши или вернуться на источник соединения.
- После наведения инструмента СОЕДИНЕНИЕ на поле ввода или вывода данных на экране появляется быстрая подсказка, которую можно использовать для уточнения места закрепления проводника.
- Отображение соединительной панели терминала объекта осуществляется пунктом контекстного меню **Visible Items»Terminals**.

Редактирование

- Ускоренное создание констант, элементов управления и отображения данных:
  - Щелкните правой кнопкой мыши на поле ввода или вывода данных функции и выберите **Create»Constant**, **Create»Control** или **Create»Indicator** в контекстном меню.
  - Для создания константы с помощью мыши переместите элемент управления или отображения данных с лицевой панели на блок-диаграмму.
  - Для создания элемента управления с помощью мыши переместите константу на лицевую панель.
- Для копирования объектов нажмите и удерживайте клавишу **<Ctrl>** во время перемещения выделенного объекта инструментом ПЕРЕМЕЩЕНИЕ.
- Чтобы ограничить направление движения объекта по горизонтали или по вертикали, во время перемещения нажмите и удерживайте клавишу **<Shift>.**
- Чтобы сохранить пропорции во время изменения размера объекта, нажмите и удерживайте клавишу **<Shift>.**
- Чтобы изменить размер элемента во время размещения его на лицевой панели, нажмите и, удерживая клавишу **<Ctrl>**, передвиньте маркеры изменения размера.
- Для замены узлов выберите пункт **Replace** в контекстном меню.
- Чтобы перейти на блок-диаграмму подпрограммы ВП, удерживая нажатой клавишу **<Ctrl>**, дважды щелкните по иконке подпрограммы инструментом ПЕРЕМЕЩЕНИЕ или УПРАВЛЕНИЕ.
- Чтобы перейти на лицевую панель подпрограммы ВП, нужно дважды щелкнуть на иконке подпрограммы инструментом ПЕРЕМЕЩЕНИЕ

или УПРАВЛЕНИЕ. Другой способ - выберите пункт главного меню **Browse»This VI's SubVIs**.

- Для окончания ввода текста следует нажать **<Shift-Enter>**.
- Для быстрого добавления варианта в элемент **ring** (Список выбора) и структуру **Case** (Варианта), нажмите клавиши **<Shift-Enter>.** Нажатие клавиш **<Shift-Enter>** добавляет вариант и позиционирует курсор в месте добавления следующего варианта. Подробная информация о работе структуры **Case** (Варианта) представлена в Уроке 8 «*Принятие решений в ВП*».
- Для копирования цвета объекта и передачи этого цвета другому объекту следует воспользоваться инструментом КОПИРОВАНИЕ ЦВЕТА. Щелкните инструментом на объекте, цвет которого необходимо копировать, затем щелкните на объект, которому требуется присвоить выбранный цвет.
- Если в процессе редактирования ВП допущена ошибка, следует выбрать пункт главного меню **Edit»Undo**.
- Для расширения свободного пространства блок-диаграммы следует нажать клавишу **<Ctrl>** и, удерживая ее, инструментом ПЕРЕМЕЩЕНИЕ на свободном пространстве обвести область расширения.

Отладка

- «Горячие» клавиши для режима пошаговой отладки:
  - **<Ctrl-down arrow>** - войти в узел.
  - **<Ctrl-right arrow>** - пропустить узел.
  - **<Ctrl-up arrow>** - выйти из узла.

Дополнительные упражнения

2-4. Создайте ВП, который сравнивает два числа. Если первое число больше или равно второму, то должен включаться светодиод.

Совет. Следует использовать функцию **Greater Or Equal?** (Больше или равно?), которая расположена на палитре функций **Functions»Programming»Comparison**

Сохраните ВП под именем *Сравнение чисел.vi*

2-5. Создайте ВП, который генерирует случайные числа в диапазоне от «0.0» до «10.0» и делит результат на число, введенное на лицевой панели. Если производится попытка деления на ноль, должен включаться светодиод.

Сохраните ВП под именем *Деление на ноль.vi*

2-6. Создайте ВП, который генерирует случайное число X в диапазоне от 0 до 100 и сравнивает его с введенным числом Y из того же диапазона. Если X больше или равно Y, то должен включаться светодиод.

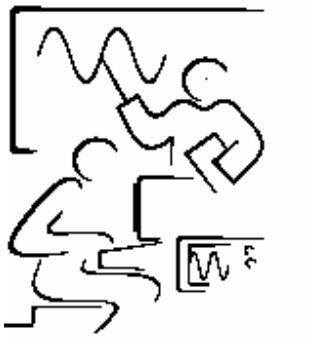
Сохраните ВП под именем *Сравнение чисел 2.vi*

2-7. Создайте ВП, преобразующий переменную типа Double в две переменные для целой (переменная типа Integer) и дробной части (переменная типа Double).

Совет. Для получения целой части используйте функцию **Round To –Infinity**, расположенную в палитре **Programming»Numeric**. Для получения остатка от деления используйте функцию **Quotient & Remainder**, тоже находящуюся в палитре **Programming»Numeric.**

Сохраните ВП под именем *Целая и дробная части.vi*

Примечания



Урок 3.
Создание подпрограмм ВП

В этом уроке представлена последовательность действий по редактированию иконки ВП, а также настройки соединительной панели (области полей ввода/вывода данных), что позволяет использовать виртуальный прибор как подпрограмму в других ВП.

В этом уроке изложены вопросы:

A. Подпрограмма ВП.

B. Создание иконки ВП и настройка соединительной панели.

C. Использование виртуального прибора в качестве подпрограммы ВП.

D. Преобразование экспресс-ВП в подпрограмму ВП

E. Превращение выделенной секции блок-диаграммы ВП в подпрограмму ВП.

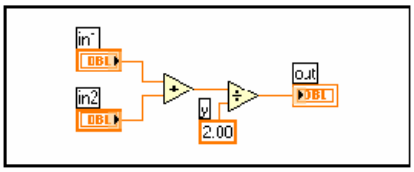
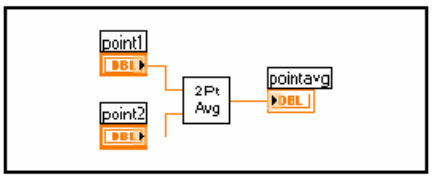
A. Подпрограммы ВП

После того как ВП сформирован, создана его иконка и настроена соединительная панель, виртуальный прибор можно использовать как подпрограмму в других ВП. Виртуальный прибор, используемый внутри другого виртуального прибора, называется подпрограммой ВП. Подпрограмма ВП соответствует подпрограмме в текстовых языках программирования. Узел подпрограммы ВП соответствует вызову подпрограммы в текстовых языках программирования. Узел – это графическое представление подпрограммы ВП, а не собственно исполняемый код подпрограммы ВП, так же как вызов подпрограммы в текстовых языках программирования не есть сам исполняемый код подпрограммы. Использование подпрограмм ВП помогает быстро управлять изменениями и отладкой блок-диаграмм. Более подробная информация о разработке приложений изложена в курсе **LabVIEW Основы II.**

Для демонстрации аналогии между подпрограммой ВП и подпрограммой текстовых языков программирования ниже представлены текстовый аналог кода и блок-диаграмма:

Исходный текст	**Вызов подпрограммы**

`function average (in1,in2,out)`
`{`
`out = (in1 + in2) / 2.0;`
`}`
	`main`
	`{`
	`average(point1,point2,pointavg);`
	`}`

Блок-диаграмма подпрограммы ВП	**Вызов подпрограммы ВП**
	

B. Создание иконки ВП и настройка соединительной панели

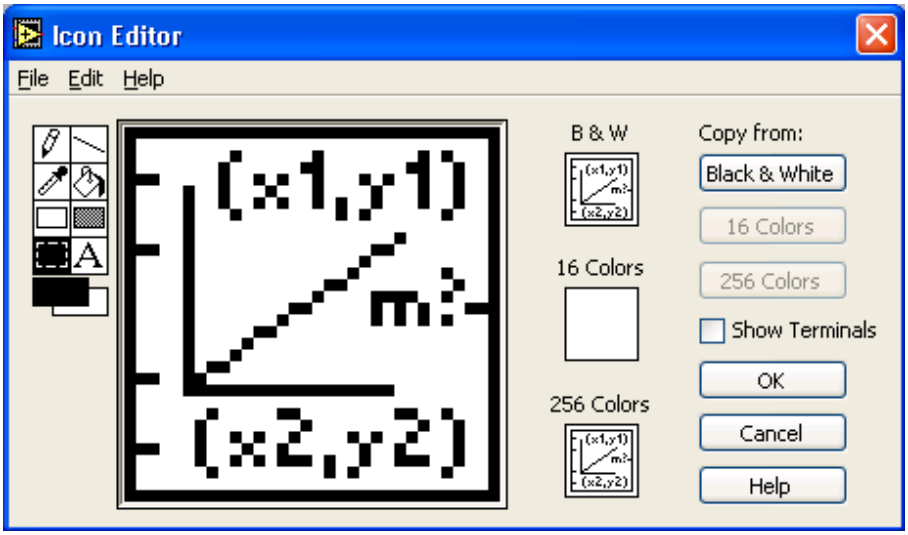
Следующий шаг после создания блок-диаграммы и формирования лицевой панели ВП – создание иконки ВП и настройка соединительной панели для использования виртуального прибора в качестве подпрограммы ВП.

Создание иконки ВП

Каждый виртуальный прибор в правом верхнем углу лицевой панели и в окне блок-диаграммы отображает иконку, показанную слева. Иконка – графическое представление прибора. Она может содержать текст, рисунок или и то и другое одновременно. Если ВП используется в качестве подпрограммы, то иконка идентифицирует его на блок-диаграмме другого ВП.

Установленная по умолчанию иконка ВП содержит номер, который указывает, сколько новых приборов открылись после запуска LabVIEW. Создать собственную иконку, отличную от заданной по умолчанию, можно, щелкнув правой кнопкой мыши по иконке в правом верхнем углу лицевой панели или блок-диаграммы. Затем выбрать пункт **Edit Icon** (Редактирование иконки) из контекстного меню. **Icon Editor** (Редактор иконки) можно также вызвать двойным щелчком левой кнопки мыши в верхнем правом углу одной из панелей. Редактирование иконки доступно также из пункта главного меню **File**, далее **VI Properties** (Свойства ВП), где в диалоговом окне **Category** (Категория) следует выбрать пункт **General** (Общие) и нажать кнопку **Edit Icon** (Редактирование иконки).

Проектирование иконки выполняется в области редактирования, расположенной в центре окна **Icon Editor** (Редактора иконки), при помощи инструментов, расположенных слева от области редактирования. Вид иконки и доступный на блок-диаграмме и в правом верхнем углу обеих панелей размер иконки появляется справа от области редактирования, в соответствующем поле, как показано ниже.



В зависимости от типа монитора, иконка может быть создана для черно-белого, 16-цветного или 256-цветного режима. Для печати, в случае отсутствия цветного принтера, LabVIEW использует черно-белую иконку. По умолчанию установлен 256-цветный режим.

Меню **Edit** (редактирование) используется для вырезания, копирования и вставки картинок из иконки или в нее. При выборе фрагмента иконки для вставки картинки LabVIEW изменяет размер картинки для соответствия размеру выбранной области.

Предусмотрена возможность перемещения графических символов из файловой системы в верхний правый угол лицевой панели или блок-диаграммы. LabVIEW автоматически преобразует изображение в иконку размером 32×32 точки.

Для копирования цветной иконки в черно-белую (или наоборот) достаточно выбрать опцию **Copy from**, находящуюся в правой части диалогового окна **Icon Editor.** Нажать кнопку **OK** для окончательной замены.

Внимание. В случае если сплошная граница вокруг иконки не нарисована, фон иконки будет прозрачным. При выборе иконки на блок-диаграмме маркеры выбора появляются вокруг каждого графического элемента иконки.

Набор инструментов для редактирования иконки расположен в левой части окна **Icon Editor** и выполняет следующие задачи:

Инструмент КАРАНДАШ позволяет рисовать или стирать по одной точке.

Инструмент ЛИНИЯ позволяет рисовать прямые линии. Для рисования вертикальных, горизонтальных и диагональных линий необходимо во время рисования нажать и удерживать клавишу **<Shift>.**

Инструмент КОПИРОВАНИЕ ЦВЕТА предназначен для копирования цвета символа в поле редактирования иконки.

Инструмент ЗАПОЛНЕНИЕ ЦВЕТОМ предназначен для заполнения ограниченной области заданным цветом переднего плана.

Инструмент ПРЯМОУГОЛЬНИК выводит в область редактирования прямоугольную границу заданным цветом переднего плана.

Двойной щелчок левой кнопкой мыши на ПРЯМОУГОЛЬНИК обводит иконку рамкой заданным цветом переднего плана.

Инструмент ЗАПОЛНЕННЫЙ ЦВЕТОМ ФОНА ПРЯМОУГОЛЬНИК выводит в область редактирования прямоугольную границу заданным цветом переднего плана, заполненную цветом фона. Двойной щелчок левой кнопкой мыши на ЗАПОЛНЕННОМ ЦВЕТОМ ФОНА ПРЯМОУГОЛЬНИК обводит иконку рамкой цвета символа и заполняет цветом фона.

Инструмент ВЫБОР предназначен для выделения фрагмента иконки, что позволяет вырезать, копировать, перемещать или вносить другие изменения в выделенный фрагмент. Чтобы очистить область редактирования иконки достаточно дважды щелкнуть левой кнопкой мыши на инструменте ВЫБОР и нажать кнопку **<Delete>.**

Инструмент ВВОД ТЕКСТА позволяет вводить текст в область редактирования иконки. Выбор шрифта производится двойным щелчком левой кнопкой мыши на инструменте ВВОД ТЕКСТА. **(Windows)** Доступна опция «**Small Fonts**».

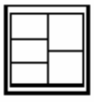
Инструмент ПЕРЕДНИЙ ПЛАН/ФОН отображает цвета фона и переднего плана (символа). При нажатии на каждый прямоугольник появляется палитра выбора цвета.

Опции в правой части **Icon Editor** предназначены для выполнения следующих задач:

- **Show Terminals -** выводит в область редактирования поля ввода/вывода данных.
- **OK -** сохраняет внесенные в иконку изменения
- **Cancel -** закрывает **Icon Editor** без сохранения.

Строка меню в окне **Icon Editor** содержит опции редактирования, такие как **Undo** (Отмена), **Redo** (Повтор), **Cut** (Вырезать), **Copy** (Копировать), **Paste** (Вставить) и **Clear** (Очистить).

Настройка соединительной панели

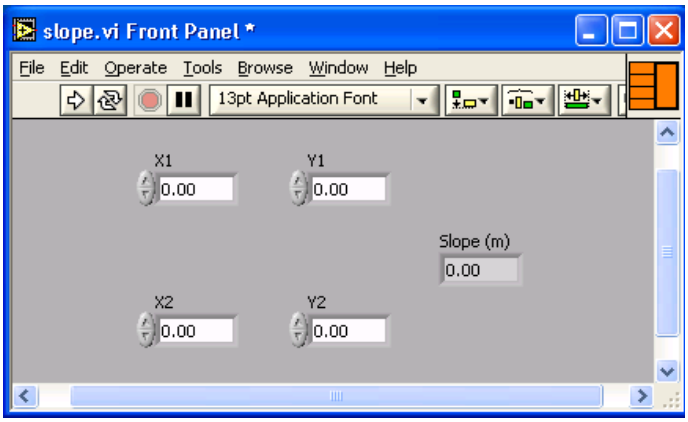


Для использования ВП в качестве подпрограммы ВП необходимо настроить соединительную панель, показанную слева.

Соединительная панель является совокупностью полей ввода/вывода данных, соответствующих элементам управления и отображения этого ВП, подобно набору параметров вызова функции в текстовых языках программирования. Соединительная панель определяет поля входных и выходных данных ВП. Таким образом, ВП можно использовать в качестве подпрограммы.

Каждому полю ввода или вывода данных назначается свой элемент лицевой панели. Для редактирования соединительной панели необходимо щелкнуть правой кнопкой мыши на иконке ВП и выбрать из контекстного меню пункт

Show Connector (Показать поля ввода/вывода данных). Вместо иконки появится соединительная панель, в которой каждый прямоугольник соответствует полю ввода или вывода данных. Количество отображаемых LabVIEW полей ввода/вывода данных соответствует количеству элементов на лицевой панели. Ниже показана лицевая панель, содержащая четыре элемента управления и один элемент отображения. Таким образом, в соединительной панели LabVIEW отображает четыре поля ввода и одно поле вывода данных.



Выбор и редактирование шаблона соединительной панели

Выбор шаблона осуществляется щелчком правой кнопки мыши на соединительной панели и выбором пункта **Patterns** (Шаблон) из контекстного меню. В шаблоне некоторые из полей ввода/вывода данных можно оставить без соединения и задействовать позднее при необходимости. Такая гибкость дает возможность вносить изменения с минимальным отражением на иерархии ВП. Причем не все элементы лицевой панели должны быть обязательно задействованы в соединительной панели.

Задействованные поля выделены цветом, соответствующим типу данных элемента. Максимально возможное количество полей ввода/вывода данных ограничено 28.

Наиболее часто используемый шаблон показан слева. Данный шаблон является стандартным для упрощения соединения. Верхние поля ввода/вывода обычно используются для ссылок, нижние — для обработки ошибок. Подробная информация об обработке ошибок находится в Уроке 6, *Кластеры*

Внимание. Следует избегать необходимости использования более 16 полей ввода/вывода данных. Наличие более 16 полей снижает удобочитаемость.

Предусмотрена возможность изменять пространственное положение полей ввода-вывода соединительной панели с помощью соответствующего пункта контекстного меню: **Flip Horizontal** (отражение по горизонтали), **Flip Vertical** (вертикали) или **Rotate 90 Degrees** (поворот на 90^{o}).

Привязка полей ввода/вывода данных к элементам лицевой панели

После выбора шаблона соединительной панели необходимо каждому полю назначить свой элемент лицевой панели. Для упрощения использования подпрограммы ВП следует поля ввода данных размещать слева, а поля, связанные с элементами отображения, - справа на соединительной панели.

Чтобы назначить поля ввода или вывода данных, следует щелкнуть по выбранному полю левой кнопкой мыши, затем щелкнуть мышью на элементе, который необходимо связать с этим полем, после этого вывести курсор в свободное пространство лицевой панели и снова щелкнуть мышью. Задействованные поля примут цвет, определенный типом данных соответствующего элемента.

Можно также сначала щелкнуть левой кнопкой мыши по элементу, а потом по полю ввод/вывода данных.

Внимание. Во время назначения полей ввода/вывода данных используется инструмент СОЕДИНЕНИЕ, однако между элементом лицевой панели и соответствующим ему полем проводник не появляется.

Упражнение 3-1. ВП Преобразования °C в °F

Цель: Создать иконку и настроить соединительную панель для возможности использования ВП в качестве подпрограммы ВП

В этом уроке представлена последовательность действий по созданию иконки и настройке соединительной панели для созданного ВП, который переводит значение измеренной температуры из °C в °F.

Лицевая панель

1. Выберите пункт главного меню **File»Open**, укажите папку **c:\exercises\LV Basics I** и выберите файл *Преобразование C в F (начало).vi*

   Если закрыты все ВП, следует нажать кнопку **Open VI** (Открыть ВП) в диалоговом окне **LabVIEW**.

Совет. Нажатие стрелки рядом с кнопкой **Open VI** (Открыть ВП) в диалоговом окне **LabVIEW** позволит обратиться к недавно использовавшимся ВП, таким как *Преобразование C в F(начало).vi*

Появится следующая лицевая панель:

Иконка и соединительная панель

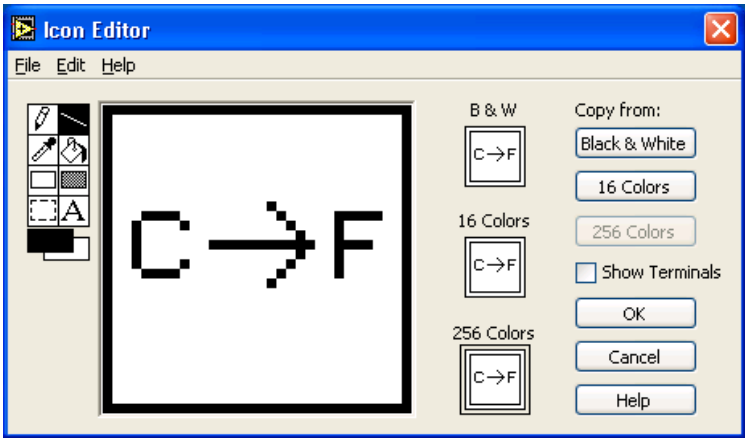
2. Щелкните правой кнопкой мыши по иконке ВП и в контекстном меню выберите пункт **Edit Icon** (Редактирование иконки). Появится диалоговое окно редактора иконки **Icon Editor**.

3. Дважды щелкните правой кнопкой мыши по инструменту ВЫБОР (показан слева).

4. Нажав кнопку **<Delete>**, очистите область редактирования иконки.

5. Дважды щелкните по инструменту ПРЯМОУГОЛЬНИК (показан слева), чтобы обвести область редактирования границей выбранного цвета.

6. Создайте следующую иконку:



a. Введите текст инструментом ВВОД ТЕКСТА, который показан слева.

b. Напечатайте «C» и «F».

c. Для выбора размера шрифта дважды щелкните левой кнопкой мыши по инструменту ВВОД ТЕКСТА.

d. Чтобы нарисовать стрелку, воспользуйтесь инструментом КАРАНДАШ.

Внимание. Для рисования вертикальных, горизонтальных и диагональных линий требуется во время рисования нажать и удерживать клавишу **<Shift>.**

e. Для передвижения текста и стрелки по полю редактирования иконки используйте инструмент ВЫБОР и стрелки на клавиатуре.

f. В разделе **Copy from** (Копировать) выберите **B & W** (черно-белую) иконку и **256 Colors** (256-цветный режим) для создания черно-белой иконки, которую LabVIEW использует в случае отсутствия цветного принтера.

g. В разделе **Copy from** (Копировать) выберите **16 Colors** и **256 Colors**.

h. После завершения редактирования иконки нажмите кнопку **OK** и закройте **Icon Editor**. Новая иконка появится в правом верхнем углу обеих панелей.

7. Перейдите на лицевую панель, щелкните правой кнопкой мыши на иконке и выберите пункт **Show Connector** (Показать поля ввода/вывода данных) из контекстного меню. Количество отображаемых LabVIEW полей ввода/вывода данных соответствует количеству элементов на лицевой панели. Например, лицевая панель этого ВП имеет два элемента **Град C** и **Град F** и LabVIEW выводит в соединительной панели два поля, показанные слева.

8. Элементам управления и отображения данных назначьте соответственно поля ввода и вывода данных.

   a. В пункте главного меню **Help** (Помощь) выберите **Show Context**

Help (контекстную подсказку) и выведите на экран окно **Context Help** (контекстной справки) для просмотра соединений.

b. Щелкните левой кнопкой мышки на левом поле соединительной панели. Инструмент УПРАВЛЕНИЕ автоматически поменяется на инструмент СОЕДИНЕНИЕ, а выбранное поле окрасится в черный цвет.

c. Щелкните левой кнопкой мыши по элементу **Град C**. Левое поле станет оранжевым и выделится маркером.

d. Щелкните курсором по свободному пространству. Маркер исчезнет, и поле окрасится в цвет данных типа соответствующего элемента управления.

e. Щелкните левой кнопкой мыши по правому полю соединительной панели и элементу **Град F**. Правое поле станет оранжевым.

f. Щелкните курсором по свободному пространству. Оба поля останутся оранжевыми.

g. Наведите курсор на область полей ввода/вывода данных. Окно **Context Help** (контекстной справки) покажет, что оба поля соответствуют типу данных двойной точности с плавающей запятой.

9. Выберите пункт главного меню **File»Save**. Сохраните ВП под именем *Преобразование C в F.vi*, он будет использоваться позднее.

10. Выберите пункт главного меню **File»Close**. Закройте ВП.

Конец упражнения 3-1

C. Использование подпрограмм ВП

После создания ВП, оформления его иконки и настройки соединительной панели ВП может использоваться в качестве подпрограммы. Чтобы поместить подпрограмму ВП на блок-диаграмму, следует выбрать на палитре **Functions** (Функций) подраздел **Select a VI** (Выбор ВП). Указать ВП и перенести его на блок-диаграмму.

Открытый ВП можно поместить на блок-диаграмму другого ВП, переместив на нее иконку этого ВП с помощью инструмента ПЕРЕМЕЩЕНИЕ.

Редактирование подпрограммы ВП

Вызов лицевой панели подпрограммы ВП из блок-диаграммы другого ВП производится двойным щелчком на нем инструментом УПРАВЛЕНИЕ или ПЕРЕМЕЩЕНИЕ. Это же можно сделать с помощью главного меню, выбрав в пункте **Browse** (Обзор) подпункт **This VI's SubVIs** (Подпрограммы этого ВП). Для вызова блок-диаграммы подпрограммы ВП следует, удерживая клавишу **<Ctrl>**, дважды щелкнуть на нем левой кнопкой мыши.

(MacOS) Нажать клавишу **<Option>. (Sun)** Нажать клавишу **<Meta>. (Linux)** Нажать клавишу **<Alt>.**

Изменения, внесенные в подпрограмму ВП, доступны вызывающим его программам только после предварительного их сохранения.

Установка значимости полей ввода/вывода данных: обязательные, рекомендуемые и дополнительные (не обязательные)

В окне контекстной справки **Context Help**, которое доступно из пункта главного меню **Help»Show Context Help**, обязательные для соединения поля обозначены жирным шрифтом, рекомендуемые – нормальным, а дополнительные (не обязательные) – светло-серым шрифтом при условии, что используется режим подробного просмотра **Detailed**. В **Simple** (Кратком) просмотре окна контекстной справки **Context Help** эта информация недоступна.

При создании подпрограммы ВП необходимо указать обязательные для соединения поля (также рекомендуемые и дополнительные) с целью предупреждения пользователя от ошибки.

Для указания значимости полей следует щелкнуть правой кнопкой мыши по соединительной панели, в контекстном меню выбрать пункт **This Connection Is** (Это поле…), установить метку на требуемую позицию: **Required** (Обязательное), **Recommended** (Рекомендуется) или **Optional** (Дополнительное).

Если поле ввода или вывода данных обязательно для соединения, то ВП не будет выполняться до тех пор, пока поле не будет правильно инициализировано. Если поле, рекомендованное для соединения, не задействовано, то ВП будет работать, но LabVIEW выдаст предупреждение в

окне **Error List** (Список ошибок), если в диалоговом окне **Error List** (Список ошибок) стоит метка в поле **Show Warnings** (Выдать предупреждение). LabVIEW не сообщает о незадействованных и не обязательных для соединения полях.

По умолчанию LabVIEW устанавливает значимость созданного поля в позицию **Recommended** (Рекомендуется). Установка **Required** (Обязательно) необходима для указания соединений, без которых ВП работать не будет. В качестве примера можно рассмотреть **File I/O** (подпрограммы работы с файлами), расположенные на палитре **Functions** (Функций).

D. Преобразование экспресс-ВП в подпрограмму ВП

Экспресс-ВП называются настраиваемые с помощью диалогового окна узлы функций. Они используются для выполнения стандартных измерений, уменьшая количество соединений проводников данных. Подробнее об экспресс-ВП можно почитать в руководстве *Getting Started with LabVIEW*.

Предусмотрена возможность создания подпрограммы ВП из сконфигурированного экспресс-ВП. Для этого достаточно щелкнуть правой кнопкой мыши по экспресс-ВП и выбрать пункт **Open Front Panel** (открыть лицевую панель) в контекстном меню.

Для создания подпрограммы ВП из сконфигурированного экспресс-ВП необходимо выполнить следующую последовательность действий:

1. сконфигурировать экспресс-ВП;
2. щелкнуть правой кнопкой мыши по экспресс-ВП и выбрать пункт **Open Front Panel** (открыть лицевую панель) в контекстном меню;
3. нажать на кнопку **Convert** (преобразовать) в появившемся диалоговом окне с предупреждением, после этого появится лицевая панель ВП;
4. отредактировать ВП;
5. выбрать пункт **Operate»Make Current Values Default** или выделять мышью каждый элемент управления и выбирать пункт контекстного меню **Make Current Values Default** для сохранения значений каждого элемента управления;
6. сохранить ВП, новая подпрограмма ВП, отображенная в виде раскрывающегося узла, заменит экспресс-ВП на блок-диаграмме.

После создания ВП из экспресс-ВП подпрограмма ВП не преобразовывается обратно в экспресс-ВП.

Упражнение 3-2. ВП Снятие напряжения

Цель: Создание подпрограммы ВП из экспресс-ВП

В упражнении будет создана подпрограмма ВП из сконфигурированного экспресс-ВП **DAQmx Assistant**, которая будет в дальнейшем использоваться на протяжении курса. Подробная информация о конфигурации и использовании средств DAQ приведена в Уроке10 "Сбор и отображение данных".

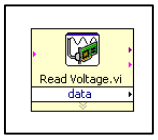
Ниже приведена последовательность действий для создания ВП, который снимает показания с температурного датчика сигнальной панели DAQ. Датчик регистрирует напряжение, пропорциональное температуре. Например, если температура составляет 23°C, то напряжение на выходе датчика будет равно 23 В. Измерение напряжения производится с помощью встроенной в компьютер DAQ платы. Датчик температуры должен быть присоединен к каналу 0 DAQ платы

Блок-диаграмма

1. Создайте новый ВП.
2. Перейдите на блок-диаграмму, выбрав **Window»Show Diagram.**

Поместите на блок-диаграмму ВП **DAQmx Assistant**, расположенный в палитре **Functions»Express»Input**. После размещения ВП на блок-диаграмме появится конфигурационное окно MAX.

3. Выберите тип измерений **Analog Input»Voltage**
4. Выберите канал измерений **Dev1»ai0**
5. Появится диалоговое окно **Analog Voltage Task Configuration.** В разделе **Task Timing** выберите тип измерений **Acquire 1 Point**. Нажмите на кнопку **OK**.
6. Щелкните правой кнопкой мыши по ВП **DAQmx Assistant** и выберите в контекстном меню пункт **Open Front Panel** (открыть лицевую панель).
7. Нажмите на кнопку **Convert** (преобразовать) в диалоговом окне с предупреждением. Появится лицевая панель подпрограммы ВП.
8. Выберите пункт **Operate»Make Current Values Default** или выделите мышью каждый элемент управления и выберите пункт контекстного меню **Make Current Values Default** для сохранения значений каждого элемента управления.
9. Сохраните ВП под именем *Снятие напряжения.vi* в каталоге **c:\exercises\LV Basics I.**
10. Закройте ВП **Снятие напряжения**.
11. Теперь исходная блок-диаграмма содержит новую подпрограмму ВП, отображенную в виде раскрывающегося узла, как показано на примере ниже. Сохраните этот ВП под именем «Термометр.vi» в каталоге **c:\exercises\LV Basics I.** Не закрывайте ВП, он будет использоваться в следующем упражнении.



Внимание. Если аппаратные средства DAQ недоступны, для имитации считывания напряжения следует использовать **(Демо) Снятие напряжения VI**, расположенный в каталоге **c:\exercises\LV Basics I.**

Конец упражнения 3-2

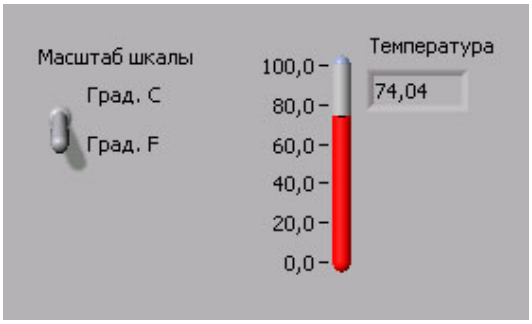
Упражнение 3-3. ВП Термометр

Цель: Создание ВП, иконки и настройка соединительной панели для обеспечения возможности использования ВП в качестве подпрограммы

Ниже приведена последовательность действий для создания ВП, который измеряет температуру с помощью температурного датчика сигнальной панели DAQ и отображает значение температуры в градусах Цельсия или в градусах Фаренгейта.

Лицевая панель

1. Откройте ВП **Термометр**, созданный в последнем упражнении, в случае если он не открыт.
2. Создайте элемент отображения данных температуры, как показано ниже:



   a. Выберите элемент отображения данных, расположенный на палитре **Controls»Modern** в разделе **Numeric** (Числовые элементы)

   b. Напечатайте **Температура** внутри собственной метки и нажмите кнопку **Enter** на инструментальной панели.

   c. Щелкните правой кнопкой мыши по элементу и выберите пункт контекстного меню **Visible Items** (Отображаемые элементы), **Digital Display** (Цифровой индикатор).

3. Создайте элемент управления в виде вертикального переключателя.

   a. Выберите вертикальный переключатель, расположенный в палитре **Controls»Modern** раздела **Boolean** (Логические элементы).

   b. Введите имя собственной метки переключателя **Масштаб шкалы** и нажмите кнопку **Enter** на инструментальной панели.

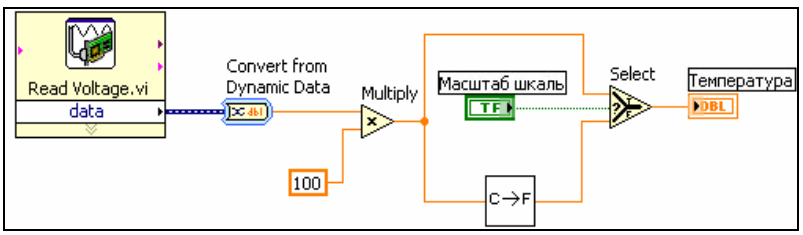
   c. Используя инструмент ВВОД ТЕКСТА, создайте на лицевой панели свободную метку **°C**, как показано выше.

   d. С помощью инструмента ВВОД ТЕКСТА, создайте на лицевой панели свободную метку **°F**, как показано выше.

4. Создайте описание ВП, которое появляется в окне контекстной справки **Context Help** после наведения курсора на иконку ВП.
   a. Выберите пункт главного меню **File»VI Properties**.
   b. Выберите пункт **Documentation** (Описание) в разделе **Category** (Категория) из выпадающего меню.
   c. В поле ввода текста напечатайте следующее:
   Этот ВП измеряет температуру, используя температурный датчик сигнальной панели DAQ.
5. Создайте описание элементов управления и отображения данных, которое появляется в окне контекстной справки **Context Help** после наведения на них курсора.
   a. Щелкните правой кнопкой мыши по элементу отображения и выберите пункт контекстного меню **Description and Tip** (Описание и предупреждения).
   b. В поле ввода текста напечатайте следующее:
   Выводит на экран значения измеренной температуры.
   c. Введите в поле **Tip** значение **Температура**.
   d. Нажмите кнопку **OK**.
   e. Щелкните правой кнопкой мыши по элементу управления и выберите пункт контекстного меню **Description and Tip** (Описание и предупреждения).
   f. В поле ввода текста напечатайте следующее:
   Определяет шкалу (по Фаренгейту или Цельсию), используемую для измерения температуры.
   g. Введите в поле **Tip** значение **шкала - °C или °F**.
   h. Нажмите кнопку **OK.**
6. Отобразите окно контекстной справки **Context Help**, которое доступно из пункта главного меню **Help»Show Context Help**.
7. Наведите курсор на один из объектов для просмотра описания их работы в окне **Context Help**.

Блок-диаграмма

8. Перейдите на блок-диаграмму, выбрав **Window»Show Diagram**.
9. Создайте блок-диаграмму, показанную ниже:



Внимание. Если аппаратные средства DAQ недоступны, для имитации считывания напряжения следует использовать **(Demo) Read Voltage VI**, расположенный в каталоге **c:\exercises\LV Basics I.**

Поместите на блок-диаграмму ВП **Convert from Dynamic Data**, расположенный в палитре **Functions»Express»Signal Manipulation**. Этот ВП преобразует динамический тип данных. В конфигурационном диалоговом окне выбрать пункт **Single Scalar** из списка **Resulting data type**.

Выберите функцию **Multiply** (Умножение), расположенную в палитре **Functions»Programming»Numeric**. Эта функция умножает считанное ВП **«Read Voltage VI»** напряжение на «100.0» для представления температуры в градусах Цельсия.

Щелкните правой кнопкой мыши по полю ввода данных *y* функции **Multiply** (Умножение) и в контекстном меню выберите пункт **Create»Constant**. Константе присвойте значение «100» и нажмите клавишу **<Enter>.**

В палитре **Functions** (Функций) в разделе **Select a VI** (Выбор ВП) в директории **c:\exercises\LV Basics I** выберите ВП **Преобразование C в F**, созданный в упражнении 3-1. Поместите его на блок-диаграмму. Этот ВП переведет градусы Цельсия в градусы Фаренгейта.

Выберите функцию **Select** (Выбор), расположенную в палитре **Functions»Programming»Comparison**. Эта функция выдает значения °C или °F в зависимости от состояния переключателя **Масштаб шкалы**.

Используйте инструмент ПЕРЕМЕЩЕНИЕ для перемещения объектов в положение, показанное на предыдущей блок-диаграмме, и инструмент СОЕДИНЕНИЕ для их соединения.

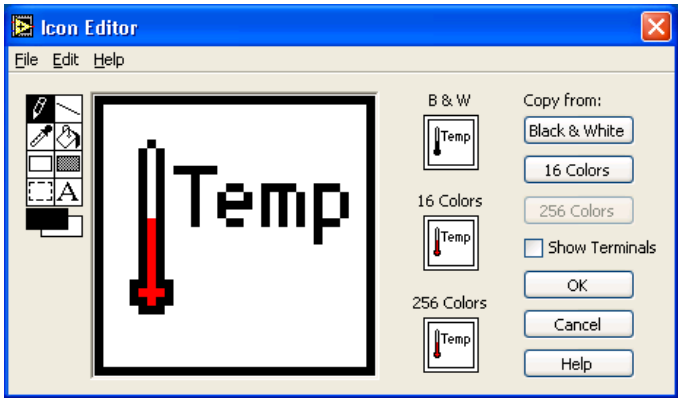
Совет. Для идентификации полей ввода и вывода данных узлов следует щелкнуть правой кнопкой мыши на узле и в контекстном меню выбрать пункт **Visible Items»Terminal**.

Лицевая панель

10. Перейдите на лицевую панель.
11. Нажмите на кнопку непрерывного запуска, показанную слева.
12. Приложите свой палец к датчику температуры и наблюдайте ее увеличение.
13. Нажмите на кнопку непрерывного запуска еще раз для остановки ВП.

Иконка и соединительная панель

14. Создайте иконку, показанную ниже, чтобы использовать ВП в качестве подпрограммы



a. Щелкните правой кнопкой мыши по иконке ВП и в контекстном меню выберите пункт **Edit Icon** (Редактирование иконки). Появится диалоговое окно **Icon Editor** (Редактор иконки).

b. Дважды щелкните по инструменту ВЫБОР (показан слева).

c. Нажмите клавишу **<Delete>**, очистите область редактирования иконки.

d. Дважды щелкните по инструменту ПРЯМОУГОЛЬНИК (показан слева), чтобы обвести область редактирования границей выбранного цвета.

e. Чтобы оформить иконку, как показано выше, воспользуйтесь инструментом КАРАНДАШ.

f. Выберите цвет инструмента ПЕРЕДНИЙ ПЛАН с помощью инструмента ЗАПОЛНЕНИЕ ЦВЕТОМ и раскрасьте термометр в красный цвет.

Внимание. Для рисования вертикальных, горизонтальных и диагональных линий необходимо во время рисования нажать и удерживать клавишу **<Shift>.**

g. Для выбора размера шрифта дважды щелкните по инструменту ВВОД ТЕКСТА.

h. Из пункта **Copy from** (Копировать) выберите **B & W** (черно-белую) иконку и **256 Colors** (256-цветный) режим печати для создания черно-белой иконки, которую LabVIEW использует в случае отсутствия цветного принтера.

i. После завершения редактирования иконки нажмите кнопку **OK** и закройте **Icon Editor** (Редактор иконки). Новая иконка появится в правом верхнем углу обеих панелей.

15. Элементам управления и отображения данных поставьте в соответствие поля ввода и вывода данных, щелкнув правой кнопкой мыши по иконке и выбрав пункт контекстного меню **Show Connector** (Показать поля ввода/вывода данных).

    a. Щелкните по левому полю соединительной панели.

b. Щелкните по элементу **Масштаб шкалы**. Левое поле ввода данных станет зеленым.

c. Щелкните по правому полю соединительной панели.

d. Щелкните по элементу **Температура**. Правое поле ввода данных станет оранжевым.

e. Щелкните по свободному пространству панели.

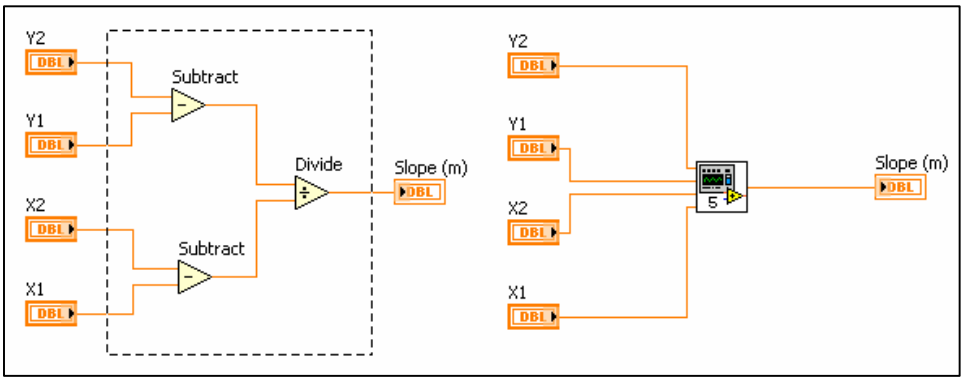
16. Сохраните ВП под именем *Термометр.vi* в директории **c:\exercises\LV Basics I**, он будет использоваться позднее.

17. Закройте ВП, выбрав пункт главного меню **File»Close**.

Конец упражнения 3-3

E. Превращение выделенной секции блок-диаграммы ВП в подпрограмму ВП

Можно упростить блок-диаграмму ВП, создав из часто выполняемых операций подпрограмму ВП. Для этого с помощью инструмента ПЕРЕМЕЩЕНИЕ необходимо выделить интересующую секцию блок-диаграммы и выбрать из пункта главного меню **Edit** (Редактирование) пункт **Create SubVI** (Создать подпрограмму ВП). Выделенная секция сменится иконкой новой подпрограммы ВП. LabVIEW создаст элементы управления и отображения данных для новой подпрограммы ВП и соединит поля ввода/вывода данных с существующими проводниками, как показано на примере ниже:



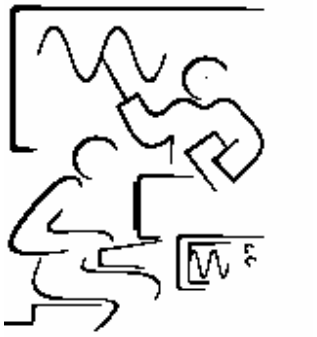
По умолчанию новая подпрограмма ВП использует шаблон для создания соединительной панели и иконку. Дважды щелкните правой кнопкой мыши по иконке подпрограммы ВП для редактирования соединительной панели и иконки и для сохранения ВП.

Внимание. Нельзя создать подпрограмму ВП из секции с количеством входов и выходов более 28, так как 28 – максимальное количество возможных полей ввода/вывода данных подпрограммы ВП.

Краткое изложение пройденного материала, советы и секреты

- Виртуальный прибор, используемый внутри другого виртуального прибора, называется подпрограммой ВП. Использование подпрограмм ВП помогает легко управлять изменениями и отладкой блок-диаграмм.
- После того как ВП сформирован, создана его иконка и настроена соединительная панель, виртуальный прибор можно использовать как подпрограмму в других ВП.
- Соединительная панель – набор полей ввода/вывода данных, соответствующих элементам управления и отображения этого ВП. Каждому полю ввода или вывода данных ставится в соответствие свой элемент лицевой панели.
- Создать собственную иконку, отличную от заданной по умолчанию, можно, дважды щелкнув левой кнопкой мыши по иконке в правом верхнем углу лицевой панели.
- Выбор шрифта в **Icon Editor** (Редакторе иконки) производится двойным щелчком левой кнопкой мыши по инструменту ВВОД ТЕКСТА.
- Можно указать обязательные для соединения поля (также рекомендуемые и дополнительные) с целью предупреждения пользователя от ошибки. Для указания значимости полей следует щелкнуть правой кнопкой мыши по соединительной панели и в пункте контекстного меню **This Connection Is** (Это поле…) установить метку на требуемую позицию.
- Описание ВП: в пункте главного меню **File**»**VI Properties** необходимо выбрать подраздел **Documentation** (Описание) в разделе **Category** (Категория) из выпадающего меню. После наведения курсора на иконку ВП в окне контекстной справки **Context Help** появляется описание ВП и индикация обязательных, рекомендуемых и не обязательных для соединения полей ввода/вывода данных.
- Описание для элементов управления и отображения данных: необходимо щелкнуть правой кнопкой мыши по элементу и выбрать из контекстного меню пункт **Description and Tip** (Описание и предупреждения). Описание появляется в окне контекстной справки **Context Help** после наведения курсора на элемент.
- Можно создать из часто выполняемых операций подпрограмму ВП. Для этого с помощью инструмента ПЕРЕМЕЩЕНИЕ («стрелка») необходимо выделить интересующую секцию блок-диаграммы и выбрать из пункта главного меню **Edit** (Редактирование) пункт **Create SubVI** (Создать подпрограммы ВП).

Примечания



Урок 4.
Многократные повторения и Циклы

Структуры являются графическим представлением операторов цикла и операторов **Case** (Варианта), используемых в текстовых языках программирования. Структуры на блок-диаграмме используются для выполнения повторяющихся операций над потоком данных, операций в определенном порядке и наложения условий на выполнение операций. Среда LabVIEW содержит следующие структуры: цикл **While** (по условию), цикл **For** (с фиксированным числом итераций), структура **Case** (Вариант), структура **Sequence** (Последовательность), структура **Event** (Событие), а также **Formula Node** (узел Формулы), **MathScript Node** (математический узел) и др.

В этом уроке рассмотрены структуры – Цикл **While** (по условию), Цикл **For** (с фиксированным числом итераций), а также функции, часто используемые с этими структурами, такие как **Shift Register** (сдвиговый регистр) и **Feedback Node** (узел обратной связи).

В этом уроке изложены вопросы:

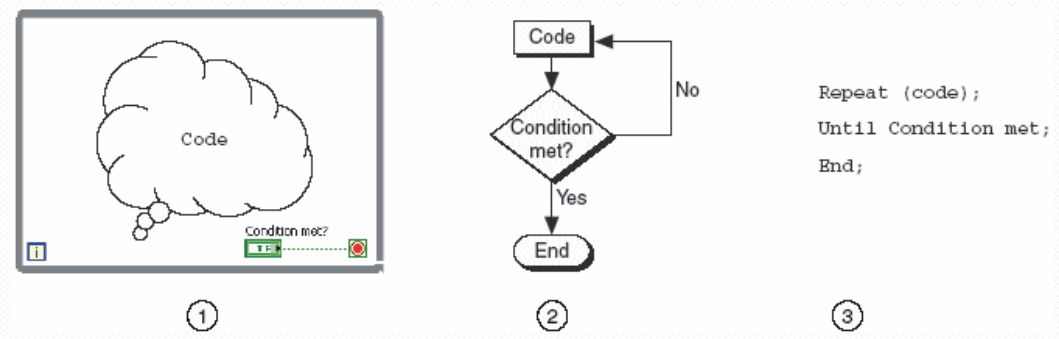
A. Использование цикла **While** (по условию).

B. Использование цикла **For** (с фиксированным числом итераций).

C. Организация доступа к значениям предыдущих итераций цикла.

A. Цикл While (по Условию)

Цикл **While** (по условию) работает до тех пор, пока не выполнится логическое условие выхода из цикла. Цикл **While** аналогичен циклам **Do** и **Repeat Until,** используемым в текстовых языках программирования. Следующая иллюстрация демонстрирует (1) цикл **While** в среде LabVIEW, (2) эквивалентную блок-схему работы цикла **While,** (3) пример текстового аналога кода работы цикла **While.**



1. LabVIEW цикл While 2. Блок-схема 3. Текстовый аналог когда

Цикл **While** находится в палитре **Functions»Pogramming»Structures.** После того как цикл выбран в палитре **Functions** (Функций), следует с помощью курсора выделить часть блок-диаграммы, которую необходимо поместить в цикл. После отпускания кнопки мыши, выделенная область блок-диаграммы помещается в тело цикла.

Добавление объектов блок-диаграммы в тело цикла осуществляется помещением или перетаскиванием объектов.

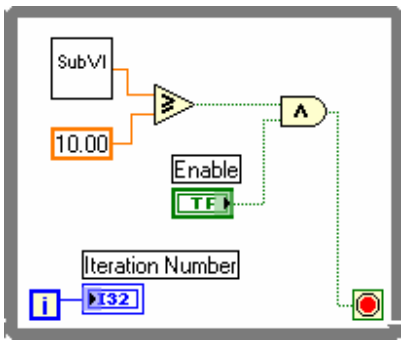
Совет. Цикл **While** выполняется всегда, по крайней мере, 1 раз.

Блок-диаграмма цикла **While** выполняется до тех пор, пока не выполнится условие выхода из цикла. По умолчанию, терминал условия выхода имеет вид, показанный слева. Это значит, что цикл будет выполняться до поступления на терминал условия выхода значения TRUE. В этом случае терминал условия выхода называется терминалом **Stop If True** (Остановка если Истина).

Терминал счетчика итераций, показанный слева, содержит значение количества выполненных итераций. Начальное значение терминала всегда равно нулю.

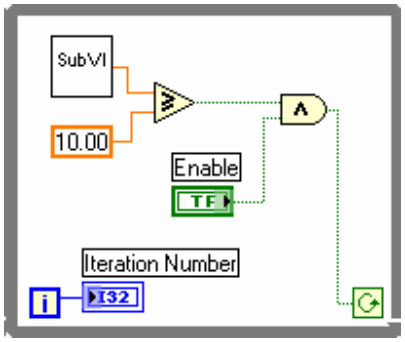
На блок-диаграмме, показанной на рисунке, условие выхода из цикла **While** определяется значением выходного параметра подпрограммы ВП большего или равного 10,00 и состоянием терминала элемента управления **Enable**. Функция **And** (Логическое «И») на выходе выдает значение TRUE**,** если оба на поля ввода данных функции поступают значения TRUE. В

противном случае функция на выходе выдает значение FALSE и работа цикла завершается**.**



В предыдущем примере велика вероятность получения бесконечно выполняемого цикла. Обычно стремятся получить единственное условие выхода из цикла, нежели одновременное выполнение двух условий.

Предусмотрена возможность изменения условия выхода и соответствующего ему изображения терминала условия выхода. Щелчком правой кнопки мыши по терминалу условия выхода или по границе цикла необходимо вызвать контекстное меню и выбрать пункт **Continue If True** (Продолжение если Истина). Также можно воспользоваться инструментом УПРАВЛЕНИЕ, щелкнув им по терминалу условия выхода. Изображение терминала условия выхода поменяется на показанное слева **Continue If True** (Продолжение Если Истина). В результате условием выхода из цикла становится поступающее на терминал условия выхода значение FALSE, как показано на следующей блок-диаграмме.



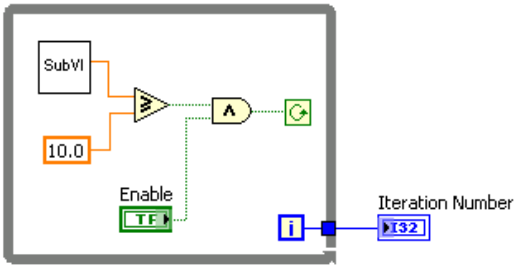
Цикл **While** выполняется до тех пор, пока выходные данные подпрограммы ВП остаются меньше «10».

Терминалы входных/выходных данных цикла

Данные могут поступать в цикл **While** (или выходить из него) через терминалы входных/выходных данных цикла. Терминалы входных/выходных данных цикла передают данные из структур и в структуры. Терминалы входных/выходных данных цикла отображаются в виде сплошных прямоугольников на границе области цикла **While.** Прямоугольник принимает цвет типа данных, передаваемых по терминалу. Данные выходят из цикла по его завершении. В случае если данные

поступают в цикл **While** через терминал входных/выходных данных цикла, выполнение цикла начинается при поступлении данных в терминал.

На следующей блок-диаграмме терминал счетчика итераций присоединен к терминалу выхода цикла. Значения из терминала выхода цикла не поступают к элементу отображения номера итерации до завершения цикла **While.**



Лишь последнее значение итерации отображается элементом отображения номера итерации.

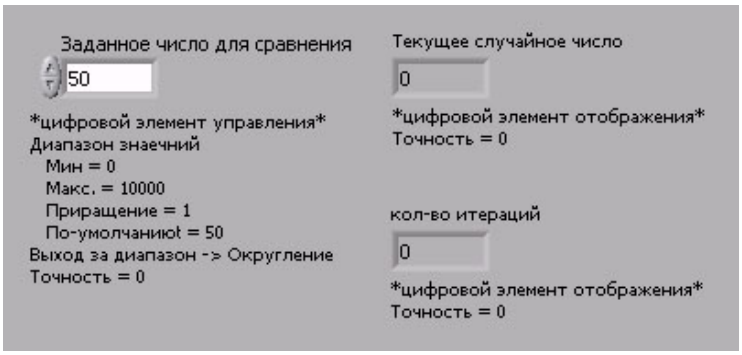
Упражнение 4-1. ВП Подсчет итераций

Цель: использование терминала выходных данных цикла While

Создайте ВП, который генерирует случайные числа до тех пор, пока одно из них не окажется равным значению, введенному в элемент управления. При этом должно отображаться количество итераций, выполненное циклом.

Лицевая панель

1. Откройте новую лицевую панель. Создайте лицевую панель, разместив на ней элементы управления и отображения, как показано ниже на рисунке.



   a. Поместите на лицевую панель числовой элемент управления, находящийся на палитре **Controls»Modern»Numeric**. Назовите элемент **Заданное число для сравнения**. Этот элемент задает число, с которым будет проводиться сравнение.

   b. Поместите на лицевую панель числовой элемент отображения, находящийся на палитре **Controls»Modern»Numeric**. Назовите элемент **Текущее случайное число**. Этот элемент отображает текущее значение, выданное функцией **Генератор случайных чисел**.

   c. Поместите еще один числовой элемент отображения на лицевую панель. Назовите элемент **Кол-во итераций**. Этот элемент показывает номер текущей итерации.

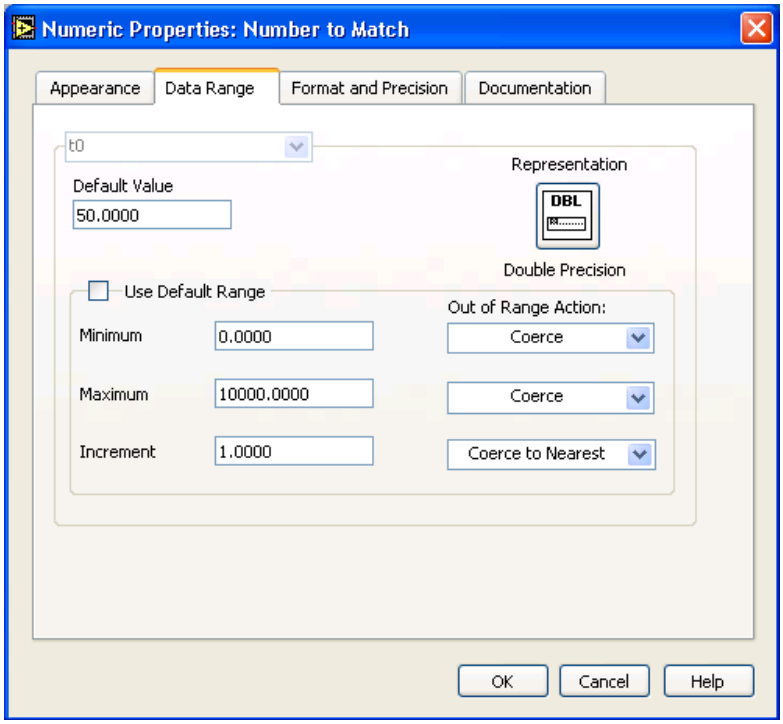
Установка диапазона данных

Чтобы значения элемента **Заданное число для сравнения** не выходили за рамки диапазона значений, выдаваемых функцией **Генератор случайных чисел**, следует использовать диалоговое окно **Data Range**. Выполните следующие шаги для настройки диапазона выходных значений элемента **Заданное число для сравнения** от 0 до 100000 с шагом изменения 1 и значением по умолчанию равным 50.

2. Щелкните правой кнопкой мыши по элементу **Заданное число для**

сравнения. Из контекстного меню выберите пункт **Data Range**. Появится диалоговое окно, показанное ниже.

3. Снимите выделение с пункта **Use Defaults** (использовать значения по умолчанию).
4. Выберите пункты, показанные в этом примере диалогового окна:



   a. Установите **Default Value** (значение по умолчанию) равным 50.
   b. Установите **Minimum Value** (минимальное значение) равным 0 и выберите **Coerce**.
   c. Установите **Maximum Value** (максимальное значение) равным 10000 и выберите **Coerce.**
   d. Установите **Increment** (значение приращения) равным 1 и выберите **Coerce to Nearest.**
5. Выберите раздел **Format and Precision** (формат и точность).

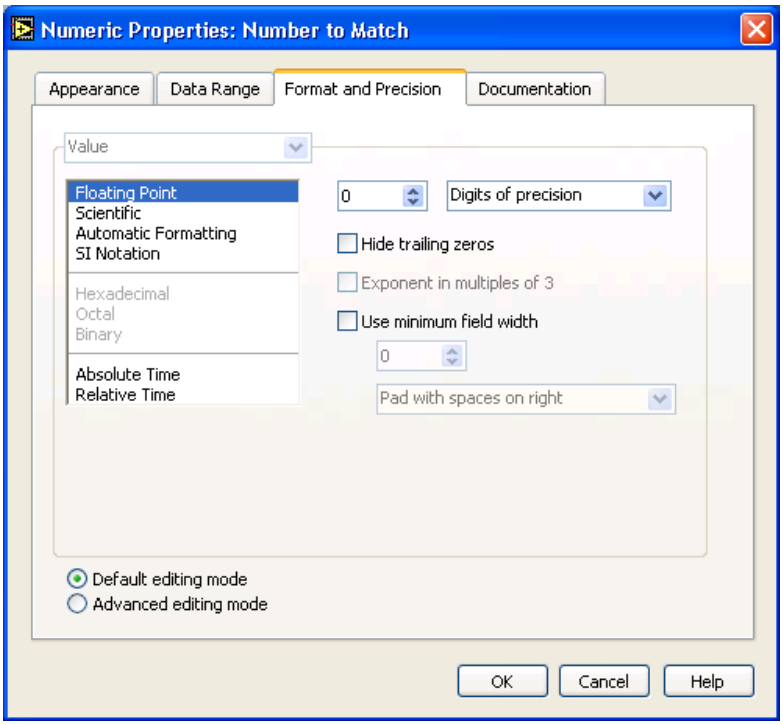
Установка количества знаков после запятой

По умолчанию, LabVIEW отображает числовые элементы управления и отображения в виде десятичных чисел с точностью до двух знаков после запятой (3,14). С помощью опции **Format&Precision** можно изменить точность и вид представления значений элементов (научная нотация, инженерная нотация, формат времени).

6. Щелкните правой кнопкой мыши по элементу **Текущее случайное число** и выберите в контекстном меню пункт **Format&Precision**.

Появится следующее диалоговое окно **Format&Precision**.

7. Сделайте настройки, показанные ниже.

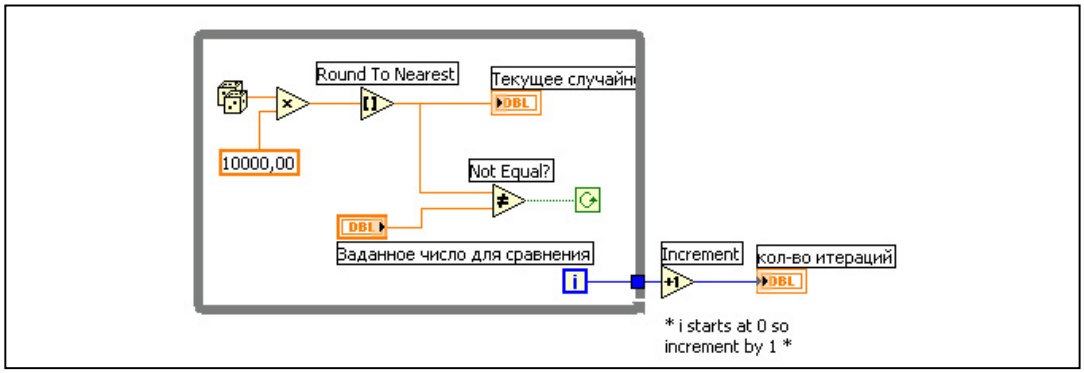


В поле ввода **Digits of Precision** следует ввести значение 0.

8. Повторите шаги 6 и 7 для элементов отображения **Текущее случайное число** и **Кол-во итераций**.

Блок-диаграмма

9. Создайте блок-диаграмму, как показано на рисунке



Поместите на блок-диаграмму функцию **Random Number** (Генератор случайных чисел), расположенную на палитре **Функций** в разделе **Functions»Programming»Numeric.** Эта функция генерирует случайные числа в пределах от 0 до 1.

Поместите на блок-диаграмму функцию **Multiply** (Умножение), расположенную в палитре **Функций** в разделе **Functions»Programming»Numeric**. Эта функция умножает текущее значение с выхода функции **Random Number** (Генератор случайных чисел) на 10000.

Создайте константу. Для этого следует навести курсор на поле ввода данных функции **Multiply** (Умножение), щелкнуть по нему правой кнопкой мыши и выбрать в контекстном меню пункт **Create»Constant**. С помощью инструмента ВВОД ТЕКСТА присвойте ей значение 10000.

Поместите на блок-диаграмму функцию **Round To Nearest** (Округление до ближайшего целого), расположенную в палитре **Функций** в разделе **Functions»Programming»Numeric.** Эта функция будет округлять полученное в пределах от 0 до 10000 случайное число до ближайшего целого числа.

Поместите на блок-диаграмму функцию **Not Equal? (≠),** расположенную в палитре **Функций** в разделе **Functions»Programming»Comparison.** Эта функция предназначена для сравнения случайного числа с числом, введенным в элемент управления **Заданное число для сравнения**. Если значения не равны, функция выдает значение TRUE.

Поместите на блок-диаграмму цикл **While**, расположенный в палитре **Функций** в разделе **Functions»Programming»Structures.** Наведите курсор на терминал условия выхода, щелкните по нему правой кнопкой мыши и выберите пункт **Continue if True** (Продолжение если Истина).

Подсоедините терминал счетчика итераций к границе области цикла **While**. На границе цикла появится синий прямоугольник. Терминал выходных данных цикла присоединен к функции приращения. При выполнении цикла счетчик итераций получает приращение равное 1. После завершения цикла значение счетчика итераций передается на выход через терминал выхода цикла. Вне тела цикла значение счетчика итераций увеличивается на единицу для отображения количества выполненных итераций.

Поместите на блок-диаграмму функцию **Increment** (Приращение на 1), расположенную в палитре **Функций** в разделе **Functions»Programming»Numeric**. Эта функция добавляет 1 к значению счетчика итераций после завершения выполнения цикла. Следует обратить внимание, что на терминале элемента **Кол-во итераций** имеется серая точка, означающая, что LabVIEW автоматически осуществляет преобразование типа данных счетчика итераций к типу данных терминала элемента **Кол-во итераций**. Подробнее о приведении типов данных можно прочитать в разделе **В "**Цикл **For** (с фиксированным числом итераций)".

10. Сохраните ВП под именем *Подсчет итераций.vi*

Запуск ВП

11. Перейдите на лицевую панель и измените значение элемента **Заданное число для сравнения**.

12. Запустите ВП. Измените значение элемента **Заданное число для сравнения** и запустите ВП снова. При этом элемент **Текущее случайное число** обновляется после каждой итерации цикла, потому что его терминал данных расположен внутри тела цикла. Значение же элемента **Кол-во итераций** обновляется после завершения цикла, потому что терминал данных этого элемента расположен вне тела цикла.

13. Чтобы посмотреть, как ВП обновляет значения элементов отображения информации, необходимо запустить ВП в режиме анимации. Для этого следует нажать на инструментальной панели кнопку **Highlight Execution**, показанную слева. Режим отладки анимирует поток данных, проходящих по блок-диаграмме. Таким образом, имеется возможность наблюдать изменения значений на каждом этапе их генерации.

14. Измените значение элемента **Заданное число для сравнения** таким образом, чтобы оно с увеличением на 1 выходило за установленный диапазон значений от 0 до 10000.

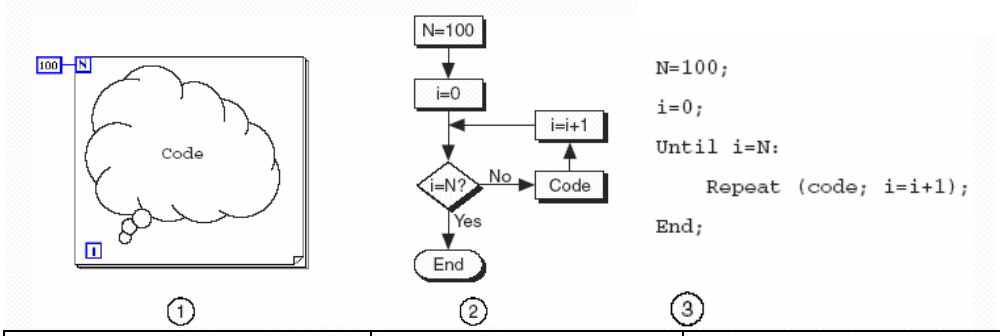
15. Запустите ВП. LabVIEW автоматически приведет новое значение к ближайшему значению в указанном диапазоне входных данных элемента.

16. Закройте ВП.

Конец упражнения 4-1

B. Цикл For (с фиксированным числом итераций)

Цикл **For** (с фиксированным числом итераций) выполняет повторяющиеся операции над потоком данных определенное количество раз. Следующая иллюстрация демонстрирует (1) цикл **For** в среде LabVIEW, (2) эквивалентную блок-схему работы цикла **For,** (3) пример текстового аналога кода работы цикла **For.**



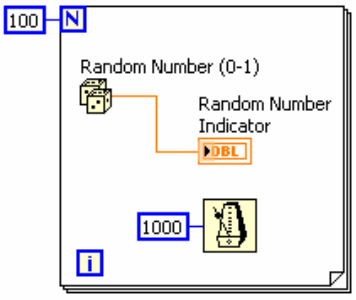
1. LabVIEW	2. Блок-схема	3. Текстовый аналог кода

Цикл **For**, расположен в палитре **Функций** в разделе **Functions»Programming»Structures**. Значение, присвоенное **терминалу максимального числа итераций N** цикла, показанного слева, определяет максимальное количество повторений операций над потоком данных.

Терминал счетчика итераций, показанный слева, содержит значение количества выполненных итераций. Начальное значение счетчика итераций всегда равно 0.

Цикл **For** отличается от цикла **While** тем, что завершает работу, выполнив заданное максимальное число итераций **N**. Цикл **While** завершает работу при выполнении заданного условия выхода из цикла.

Цикл **For**, показанный на рисунке ниже, генерирует случайное число каждую секунду 60 раз и отображает их в элементе отображения данных.

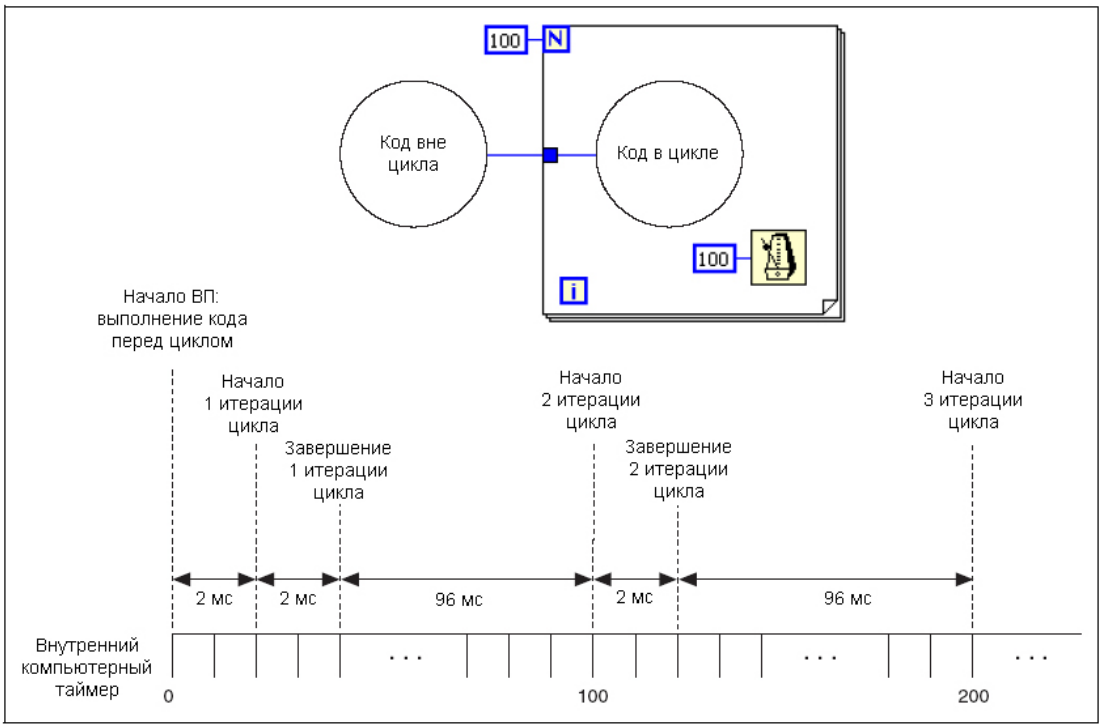


Функции ожидания

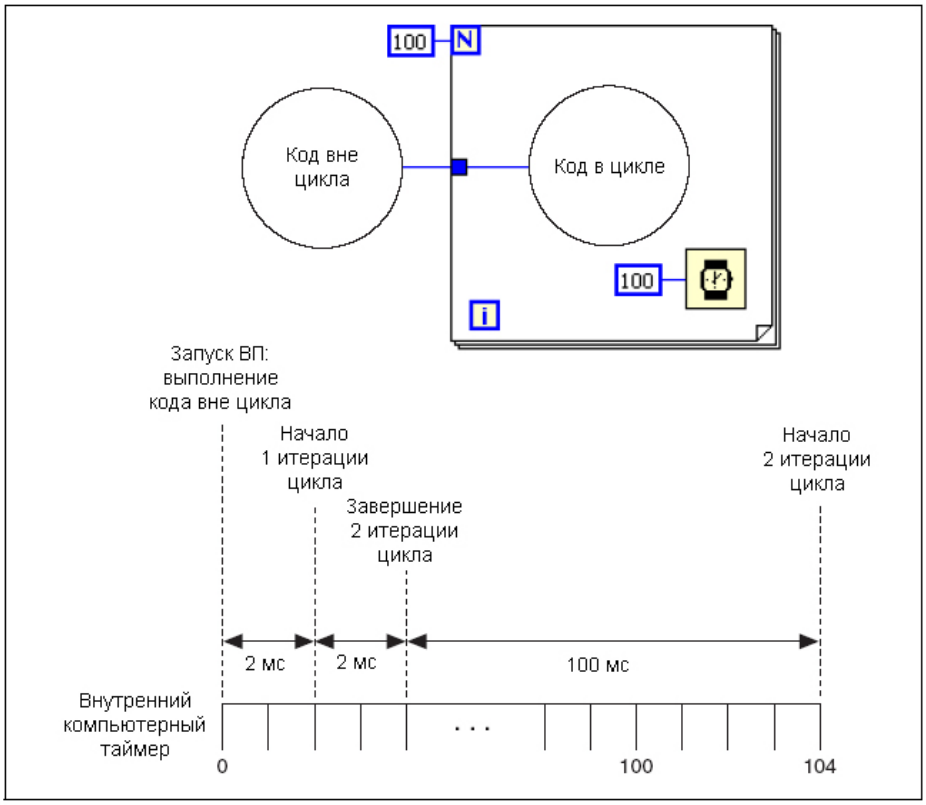
Функция **Wait Until Next ms Multiple,** показанная слева, обеспечивает интервал между итерациями, равный интервалу времени, необходимому для того, чтобы миллисекундный счетчик достиг значения, кратного введенному пользователем. Эта функция используется для синхронизации действий. Функцию **Wait Until Next ms Multiple** вызывают внутри цикла для контроля

скорости выполнения цикла.

Функция **Wait Until Next ms Multiple** обеспечивает интервал между итерациями, равный интервалу времени, необходимому внутреннему таймеру компьютера для достижения указанного кратного значения. Существует вероятность, что первый период цикла будет коротким, как показано ниже.



Функция **Wait(ms),** показанная слева, добавляет время ожидания ко времени выполнения программы, как показано ниже. Это может вызвать затруднения, если время выполнения программы является переменным.



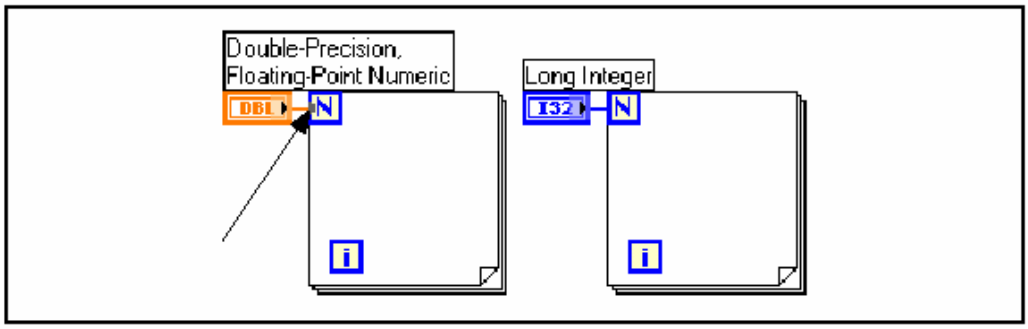
Примечание. ВП **Express Time Delay**, находящийся в палитре **Functions»Express»Execution Control**, подобен функции **Wait(ms).** Однако он обладает встроенным кластером ошибок. Подробнее о кластерах ошибок можно прочитать в Уроке 6 "*Кластеры"*.

Функции ожидания находятся в палитре **Functions»Programming»Timing.**

Преобразование типов данных

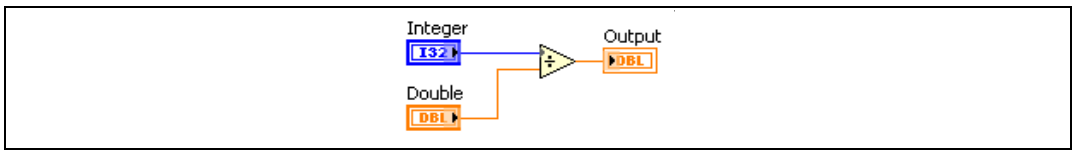
LabVIEW может оперировать с такими типами данных, как целочисленный тип (integer): **byte**, **word, long,** число с плавающей запятой: **single, double, extended precision**, комплексное число: **single, double, extended precision**. Когда в поле ввода данных функции поступают операнды разных типов, то значение на выходе функции принимает формат данных более широкого диапазона. При этом LabVIEW автоматически осуществляет преобразование типов и в месте соединения проводника с терминалом появляется изображение серой точки.

Например, **терминал максимального числа итераций N** цикла **For** имеет **целочисленный тип двойной точности (long integer).** На него поступают данные в формате числа двойной точности с плавающей запятой. На терминале числа итераций появляется серая точка, как показано ниже на рисунке.



Если в поля ввода данных функции, работающей с данными одного типа, поступают данные двух разных типов, LabVIEW приводит тип данных одного из терминалов к типу данных другого терминала. LabVIEW выбирает тип данных, занимающий большее количество бит. Если типы эквивалентны по количеству занимаемых бит, LabVIEW предпочитает беззнаковый тип данных.

В следующем примере данные типов I32 (signed 32 bit integer / 32-битный целочисленный со знаком) и DBL (double 64 bit floating-point / 64-битное с плавающей запятой двойной точности) поступают в поля ввода функции деления. LabVIEW на выходе функции формирует данные в представлении DBL, поскольку тип данных I32 занимает меньшее количество бит.



Для изменения типа представления данных на объектах блок-диаграммы необходимо щелкнуть по ним правой кнопкой мыши и из контекстного меню выбрать пункт **Representation**.

Когда LabVIEW проводит преобразование данных из формата числа двойной точности с плавающей запятой в целочисленный формат, то значение x,5 округляется до ближайшего целого четного. Например, LabVIEW округляет 2,5 до 2, а 3,5 до 4.

Подробнее о типах данных можно прочитать в Уроке 2, "*Создание ВП"* или в *LabVIEW Help*.

Упражнение 4-2. ВП Измерение температуры во времени

Цель: измерение температуры раз в секунду в течение одной минуты.

Ниже приведена последовательность действий для создания ВП, который использует ВП Термометр для измерения температуры раз в секунду в течение одной минуты.

Лицевая панель

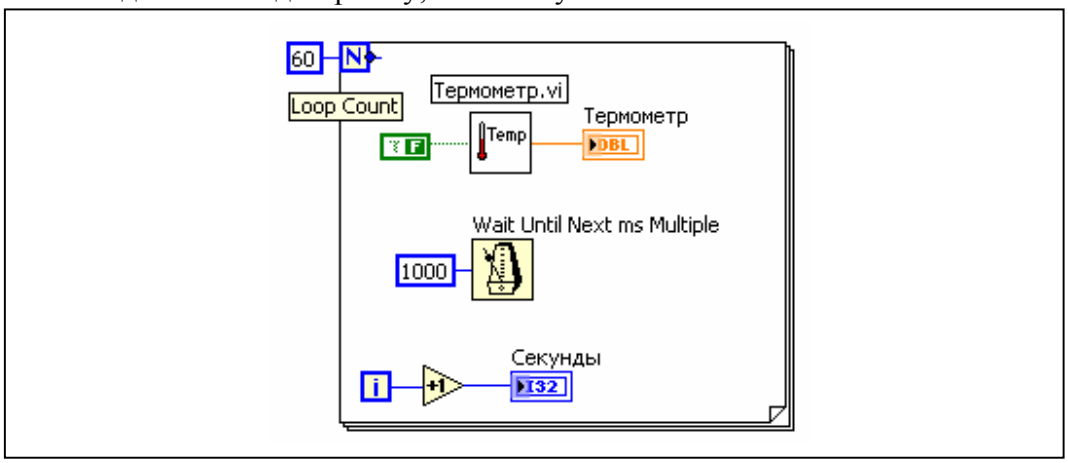
1. Откройте новый ВП и создать лицевую панель, как показано ниже на рисунке:

Поместите Термометр, расположенный на палитре **Controls»Modern»Numeric**, на блок-диаграмму для отображения на экране измерений температуры.

Поместите на лицевую панель цифровой элемент отображения данных, расположенный в палитре **Controls»Modern»Numeric**. Назовите его **Секунды**.

Блок-диаграмма

2. Создайте блок-диаграмму, показанную ниже:



Поместите ВП Термометр на блок-диаграмму. Для этого выберите **Functions»Select a VI** и укажите папку **c:\exercises\LV Basics I**. Этот ВП, снимает показания с устройства DAQ. Если устройства DAQ с датчиком

температуры, присоединенным к каналу 0 DAQ платы, недоступны, следует использовать *(Демо) Термометр.vi*

Щелкните правой кнопкой мыши по полю ввода данных **Temp Scale** и в контекстном меню выберите пункт **Create»Constant**. Константе присвойте значение FALSE — для градусов Фаренгейта и TRUE — для градусов Цельсия.

Поместите на блок-диаграмму функцию **Wait Until Next ms Multiple**, находящуюся в палитре **Functions»Programming»Timing**. Щелкните правой кнопкой мыши по полю ввода данных и выберите пункт **Create»Constant.** Созданной константе присвойте значение 1000. Теперь каждая итерация цикла выполняется с интервалом времени 1000 мс (раз в секунду).

Поместите на блок-диаграмму функцию **Increment** (приращение), находящуюся в палитре **Functions»Programming»Numeric**. Эта функция добавляет 1 к значению счетчика итераций после завершения выполнения цикла.

3. Сохраните ВП под именем *Измерение температуры во времени.vi* в папке **c: \exercises\LV\Basics I.**
4. Запустите ВП. Снятие первого показания может занять больше одной секунды в случае, если компьютер будет конфигурировать устройство DAQ.
5. Закройте ВП.

Дополнительно

6. Создайте ВП, генерирующий случайные числа в цикле **While**. Организуйте выход из цикла по нажатию кнопки на лицевой панели ВП.
7. Сохраните ВП под именем *Цикл While.vi* в папке **c:\exercises\LV Basics I.**
8. Перейдите к выполнению упражнений повышенной сложности или закройте ВП.

Упражнения повышенной сложности

9. Переделайте ВП Цикл **While** таким образом, чтобы выход из цикла осуществлялся по нажатию кнопки на лицевой панели или по достижению циклом **While** числа итераций, заданного элементом управления на лицевой панели.
10. Сохраните ВП под именем *Цикл While-For.vi* в папке **c:\exercises\LV Basics I.**
11. Закройте ВП.

Конец упражнения 4-2

C. Организация доступа к значениям предыдущих итераций цикла

При работе с циклами зачастую необходим доступ к значениям предыдущих итераций цикла. Например, в случае ВП, измеряющего температуру и отображающего ее на графике, для отображения текущего среднего значения температуры, необходимо использовать значения, полученные в предыдущих итерациях. Есть два пути доступа к этим данным: **Shift Register** (сдвиговый регистр) и **Feedback Node** (узел обратной связи).

Сдвиговые регистры

Сдвиговые регистры используются при работе с циклами для передачи значений от текущей итерации цикла к следующей. Сдвиговые регистры аналогичны статическим переменным в текстовых языках программирования

Сдвиговый регистр выглядит как пара терминалов, показанных слева. Они расположены непосредственно друг против друга на противоположных вертикальных сторонах границы цикла. Правый терминал содержит стрелку «вверх» и сохраняет данные по завершению текущей итерации. LabVIEW передает данные с этого регистра в следующую итерацию цикла. Сдвиговый регистр создается щелчком правой кнопки мыши по границе цикла и выбором из контекстного меню пункта **Add Shift Register**.

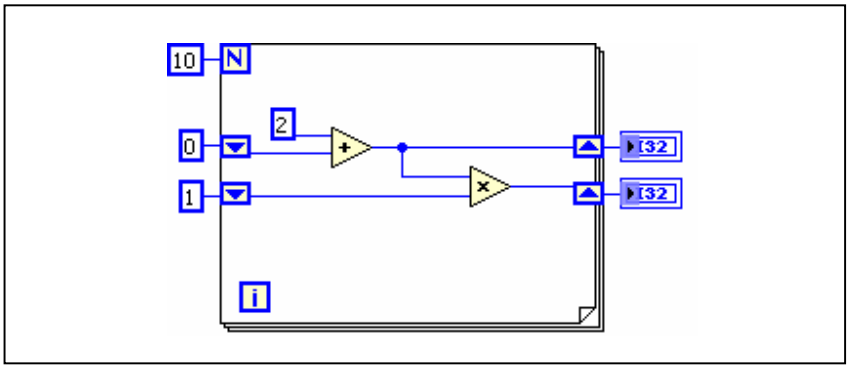
Сдвиговый регистр передает данные любого типа, он автоматически принимает тип первых поступивших на него данных. Данные, передаваемые на терминалы сдвигового регистра, должны быть одного типа.

Чтобы инициализировать сдвиговый регистр, необходимо передать на его левый терминал любое значение извне цикла. Если не инициализировать сдвиговый регистр, он использует значение, записанное в регистр во время последнего выполнения цикла или значение, используемое по умолчанию для данного типа данных, если цикл никогда не выполнялся.

Цикл с неинициализированным сдвиговым регистром используется при неоднократном запуске ВП для присвоения выходному значению сдвигового регистра значения, взятого с последнего выполнения ВП. Чтобы сохранить информацию о состоянии между последующими запусками ВП, следует оставить вход левого терминала сдвигового регистра не определенным. После завершения выполнения цикла последнее значение, записанное в регистр, останется на правом терминале. При последующей передаче данных из цикла через правый терминал будет передано последнее значение, записанное в регистр.

Предусмотрена возможность создания нескольких сдвиговых регистров в одной структуре цикла. Если в одном цикле выполняется несколько

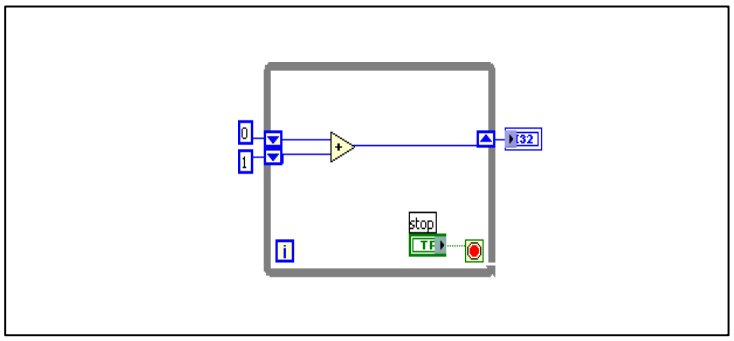
операций, следует использовать сдвиговый регистр с несколькими терминалами для хранения данных, полученных в результате выполнения различных операций цикла. На следующем рисунке показано использование двух инициализированных сдвиговых регистров.



Стек сдвиговых регистров.

Для создания стека сдвиговых регистров достаточно щелкнуть правой кнопкой мыши по левому терминалу и выбрать пункт контекстного меню **Add Element**. Стек сдвиговых регистров осуществляет доступ к значениям предыдущих итераций цикла. Стек сдвиговых регистров сохраняет данные предыдущей итерации и передает эти значения к следующей итерации.

Стек сдвиговых регистров может находиться только в левой части цикла, так как правый терминал лишь передает данные из текущей итерации в следующую.



При добавлении еще двух сдвиговых регистров к левому терминалу данные последних трех итераций переносятся на следующую итерацию, при этом значение последней итерации сохраняется в самом верхнем сдвиговом регистре. Второй терминал сохранят данные, переданные ему с предыдущей итерации, нижний терминал хранит данные, полученные две итерации назад.

Узлы обратной связи

Узел обратной связи, показанный слева, автоматически появляется в циклах **While** или **For** при соединении поля вывода данных подпрограммы ВП, функции или группы подпрограмм ВП и функций с полем ввода данных тех же самых подпрограмм ВП, функций или их групп. Как и сдвиговый регистр, узел обратной связи сохраняет данные любого типа по завершению

текущей итерации и передает эти значения в следующую итерацию. Использование узлов обратной связи позволяет избежать большого количества проводников данных и соединений.

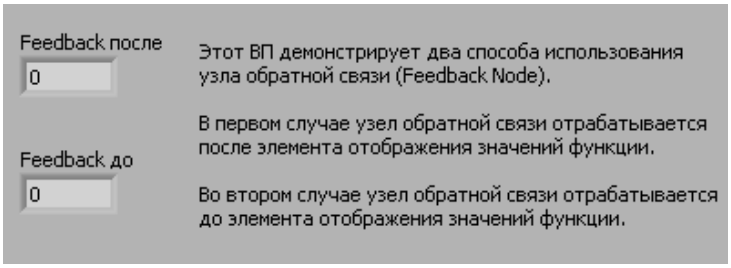
Можно поместить узел обратной связи внутри цикла **While** или **For,** выбрав **Feedback Node** в палитре **Structures.** При помещении узла обратной связи на проводник данных до ответвления, передающего данные на выходной терминал цикла, узел обратной связи передает все значения на выходной терминал цикла. При помещении узла обратной связи на проводник после ответвления, передающего данные на выходной терминал цикла, узел обратной связи передаст все значения обратно на поле ввода данных ВП или функции, а затем передаст последнее значение на выходной терминал цикла. Следующее упражнение содержит пример работы узла обратной связи.

Упражнение 4-3. Доступ к данным предыдущих итераций

Цель: Использование сдвиговых регистров и узлов обратной связи для организации доступа к значениям на предыдущих итерациях цикла For.

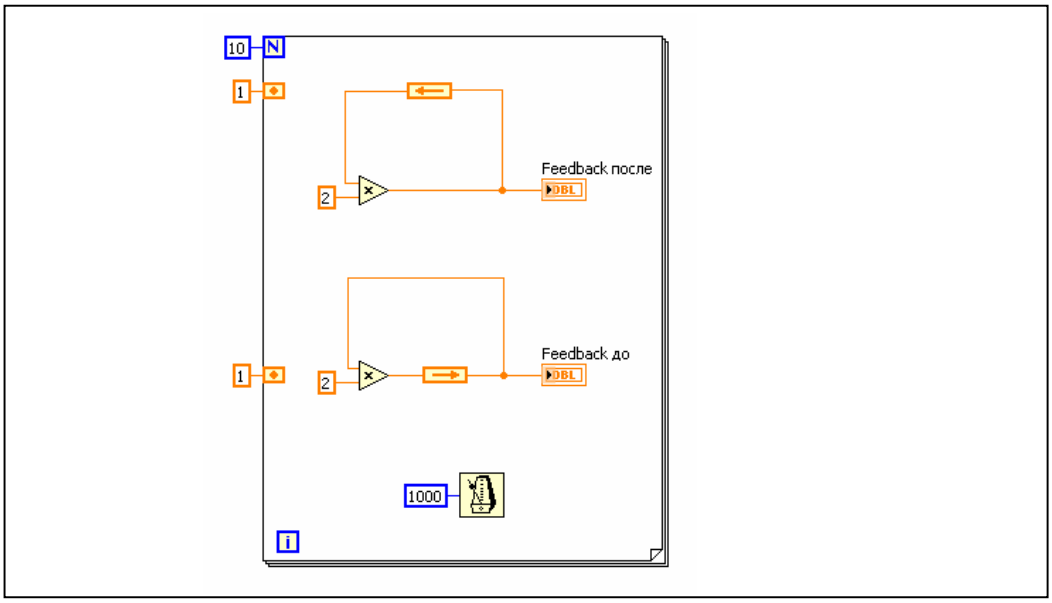
Лицевая панель

1. Откройте ВП **Узел обратной связи VI**, находящийся в каталоге **c:\exercises\LV Basics I.** Лицевая панель этого ВП, показанная ниже, уже создана



Блок-диаграмма

2. Отобразите блок-диаграмму, показанную ниже. Разместите лицевую панель и блок-диаграмму на экране так, чтобы они не перекрывали друг друга. Переместите палитры **Tools** и **Functions**, если это необходимо.



Единица, соединенная с левым терминалом цикла **For**, инициализирует узел обратной связи. Таймер **Wait Until Next ms Timer** замедляет процесс выполнения программы. Для замедления выполнения процесса

можно также использовать режим **Highlight Execution** (анимации выполнения блок-диаграммы).

На данной блок-диаграмме один и тот же процесс выполняется дважды, при этом узел обратной связи помещен в различных местах соединения.

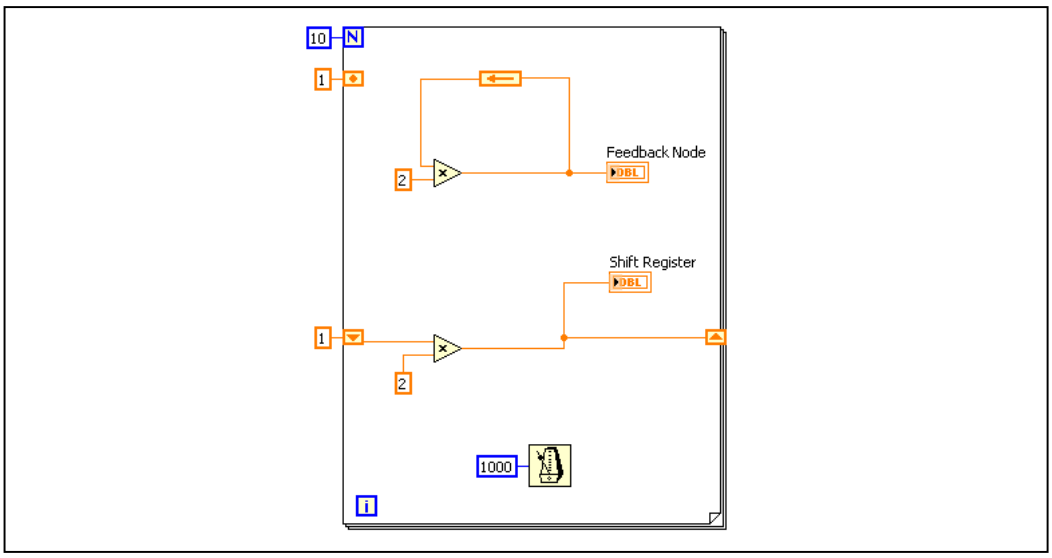
Запуск ВП

3. Запустите ВП. Программа в верхней части сначала считывает значение узла обратной связи, инициализированного значением 1. Затем это значение передается функции **Multiply** (умножение).

Программа в нижней части сначала считывает значения узла обратной связи, инициализированного значением 1. Затем это значение передается на цифровой элемент отображения. Функция **Multiply** (умножение) не будет выполняться до следующей итерации цикла.

4. Активируйте режим анимации выполнения блок-диаграммы, нажав на показанную слева кнопку **Highlight Execution**. Запустите ВП еще раз для наблюдения порядка выполнения программы. Отключите режим анимации для работы ВП в нормальном режиме.

5. Замените узел обратной связи сдвиговым регистром, как показано на следующей блок-диаграмме:



a. Выделите нижний узел обратной связи и нажмите клавишу <Delete>, чтобы удалить его.

b. Щелкните правой кнопкой мыши по границе цикла и выберите пункт контекстного меню **Add Shift Register**.

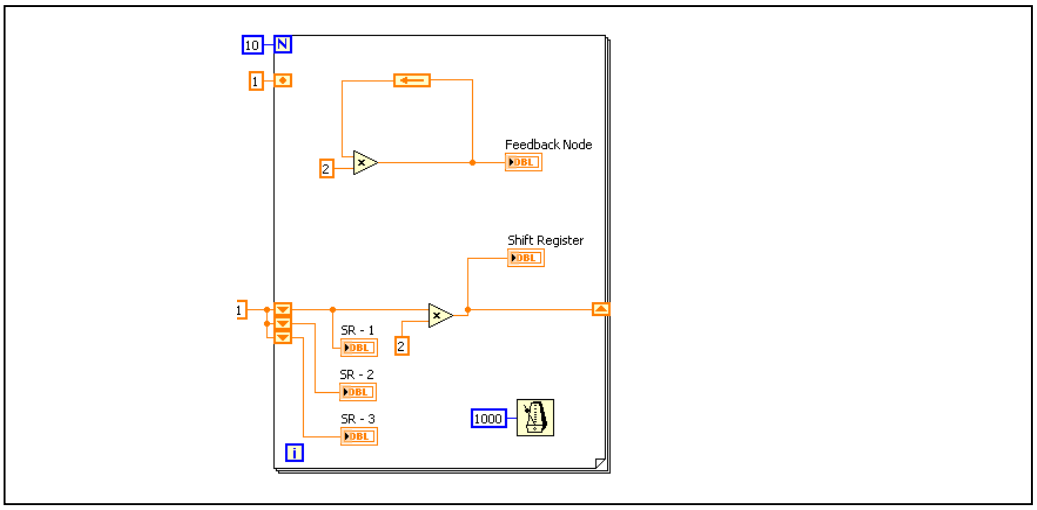
c. Инициализируйте сдвиговый регистр значением 1.

d. Переименуйте верхний и нижний элементы отображения соответственно **Узел обратной связи** и **Сдвиговый регистр**.

6. Запустите ВП. Обратите внимание, что узел обратной связи и сдвиговый регистр выполняют одинаковые функции.
7. Не закрывайте ВП, перейдите к выполнению дополнительных упражнений или закройте ВП, не сохраняя его.

Дополнительно

8. Модифицируйте сдвиговый регистр, чтобы он отображал последние три итерации цикла **For**, как показано на следующей блок-диаграмме:



   a. Измените размер левого сдвигового регистра до трех элементов.
   b. Инициализируйте все элементы сдвигового регистра значением 1.
   c. Щелкните правой кнопкой мыши по каждому элементу и выберите пункт контекстного меню **Create»Indicator.** Назовите каждый элемент отображения.
9. Запустите ВП.
10. Закройте ВП, не сохраняя его.

Конец упражнения 4-3

Краткое изложение пройденного материала, советы и секреты

- Структуры на блок-диаграмме используются для повторяющихся операций над потоками данных, для выполнения операций над данными в определенном порядке и наложения условий выполнения операций.
- Цикл **While** выполняется до тех пор, пока на его терминал условия выхода не поступит логическое значение выхода из цикла. По умолчанию, цикл **While** выполняется, пока на терминал условия выхода из цикла не поступит значение TRUE**.**
- Цикл **For** выполняется определенное количество раз.
- Для создания цикла достаточно с помощью курсора выделить часть блок-диаграммы, которую необходимо поместить в цикл, и отпустить кнопку мыши или перенести эту часть блок-диаграммы в тело цикла с помощью курсора.
- Функция **Wait Until Next ms Multiple** обеспечивает заданный интервал времени выполнения итераций цикла.
- Функция **Wait** ожидает определенное время.
- Серые точки появляются в местах соединения проводника с терминалом данных, когда LabVIEW проводит автоматическое приведение типов данных.
- Для передачи данных из одной итерации цикла в другую используются сдвиговые регистры.
- Для создания сдвигового регистра необходимо щелкнуть правой кнопкой мыши по границе цикла и выбрать из контекстного меню пункт **Add Shift Register**.
- Для передачи значений из нескольких предыдущих итераций цикла в последующую необходимо сдвиговому регистру добавить дополнительные терминалы данных. Для этого необходимо щелкнуть по нему правой кнопкой мыши и выбрать из контекстного меню пункт **Add Element.**
- Узел обратной связи сохраняет данные любого типа по завершению текущей итерации, передает эти значения в следующую итерацию.
- Узел обратной связи используется для уменьшения количества соединений.

Дополнительные упражнения

4-4. Создать ВП – калькулятор модуля разности, суммы, произведения, либо отношения целого числа со знаком и действительного числа (без использования встроенной функции модуль). Прекращение работы – по кнопке на передней панели.

Сохраните ВП под именем *Калькулятор.vi*

4-5. Создайте ВП, вычисляющий корни квадратного уравнения $ax^2 + bx + c = 0$ в комплексной форме. Значения коэффициентов a,b,c вводить с лицевой панели. Ограничить диапазон их возможных значений +/-10. Если корень действительный (то есть дискриминант больше нуля), то на лицевой панели должен зажигаться светодиод.

Сохраните ВП под именем *Вычисление корней квадратного уравнения.vi*

4-6. Создайте ВП, вычисляющий сумму, разность, произведение или частное двух чисел, в зависимости от положения селектора и прекращающий работу по нажатию на кнопку «STOP». Сообщать пользователю о некорректности деления на ноль.

Совет. Используйте функцию **One Button Dialog**, расположенную в палитре **Programming»Dialog&User Interface**, для выдачи сообщения о делении на ноль.

Сохраните ВП под именем *Калькулятор 2.vi*

4-7. Создайте ВП, который угадывает загаданное Вами число (от 1 до 100) максимум за семь шагов, «задавая» Вам вопросы типа: «загаданное Вами число >50?» и т.д. и, получая от Вас утвердительные или отрицательные ответы.

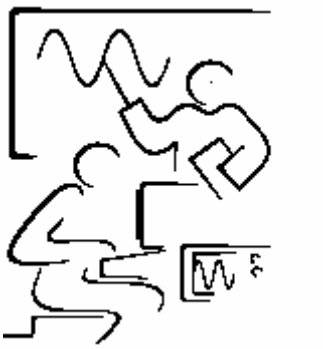
Совет. Используйте функцию **Two Button Dialog**, расположенную в палитре **Programming»Dialog&User Interface**, для задания вопроса.

Сохраните ВП под именем *Угадай число.vi*

4-8. Создйте ВП, вычисляющий n!, и оформите его в качестве подпрограммы (создайте пиктограмму и соединительную панель). Вычислите с помощью этой подпрограммы количество сочетаний $C_n^m = \frac{n!}{(n-m)!\,m!}$

Сохраните ВП под именем *Число сочетаний.vi*

Примечания



Урок 5
Массивы

В этом уроке рассказывается об объединении элементов одного типа данных в массивы.

В этом уроке изложены вопросы:

A. Что такое массив.

B. Создание массивов с помощью цикла.

C. Использование функций работы с массивами.

D. Полиморфизм.

A. Массивы

Массивы объединяют элементы одного типа данных. Массив – это набор элементов определенной размерности. Элементами массива называют группу составляющих его объектов. Размерность массива - это совокупность столбцов (длина) и строк (высота), а также глубина массива. Массив может иметь одну и более размерностей, и до 2^{31}–1 элементов в каждом направлении, насколько позволяет оперативная память.

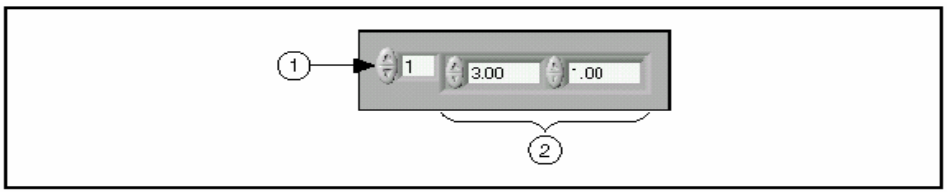
Данные, составляющие массив, могут быть любого типа: целочисленного, логического или строкового. Массив также может содержать элементы графического представления данных и кластеры. Использовать массивы удобно при работе с группами данных одного типа и при накоплении данных после повторяющихся вычислений. Массивы идеально подходят для хранения данных, полученных с графиков, или накопленных во время работы циклов, причем одна итерация цикла создает один элемент массива.

Нельзя создать массив, состоящий из массивов. Однако можно создать массив кластеров, где каждый кластер будет состоять из одного или более массивов. Более подробную информацию по этому вопросу можно найти в Уроке 6, *Кластеры*.

Все элементы массива упорядочены. Чтобы к ним было легко обращаться, каждому элементу присвоен индекс. Нумерация элементов массива всегда начинается с 0. Таким образом, индексы массива находятся в диапазоне от 0 до (n-1), где n - число элементов в массиве. Например, в массиве из девяти планет солнечной системы n=9, следовательно, значение индекса находится в пределах от 0 до 8. Земля является третьей планетой от Солнца, поэтому ее индекс равен 2.

Создание массива элементов управления и отображения

Для создания массива элементов управления или отображения данных, как показано в примере, необходимо выбрать шаблон массива из палитры **Controls»Modern»Array, Matrix & Cluster** и поместить его на лицевую панель. Затем поместить в шаблон массива элемент управления либо отображения данных. Поместить в шаблон массива запрещенный элемент управления или отображения, например, двухкоординатный график осциллограмм **(XY graph)** не удастся.



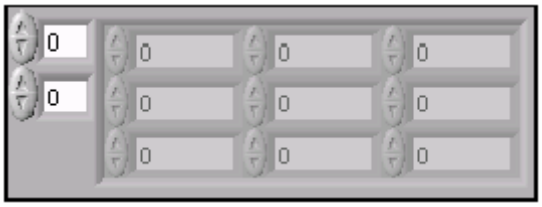
1. Элемент индекса массива
2. Элементы значений массива

Поместить объект в шаблон массива следует до того, как он будет использоваться на блок-диаграмме. Если этого не сделать, то шаблон массива не будет инициализирован, и использовать массив будет нельзя.

Двумерные массивы

В двумерном (2D) массиве элементы хранятся в виде матрицы. Таким образом, для размещения элемента требуется указание индекса столбца и строки. На иллюстрации показан двумерный массив, состоящий из 6 столбцов (длина) и 4 строк (высота). Количество элементов в массиве = 24 (6×4=24).

Для увеличения размерности массива необходимо щелкнуть правой кнопкой мыши по элементу индекса и выбрать из контекстного меню пункт **Add Dimension**. С этой целью также можно использовать инструмент ПЕРЕМЕЩЕНИЕ. Для этого надо просто изменить размер элемента индекса. Ниже приведен пример неинициализированного двумерного массива элементов управления.

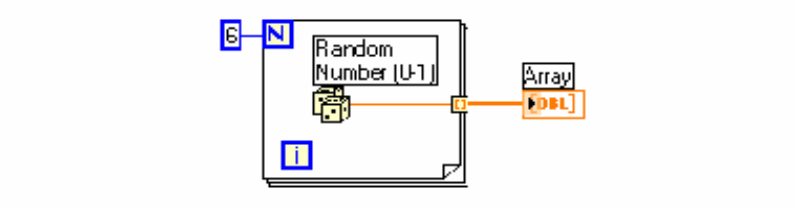


Создание массива констант

Создать массив констант на блок-диаграмме можно, выбрав в палитре **Functions»Programming»Array** шаблон **Array Constant** и поместив в него числовую константу. Массив констант удобно использовать для передачи данных в подпрограммы ВП.

B. Автоматическая индексация

Цикл **For** и цикл **While** могут автоматически накапливать массивы и проводить их индексацию на своих границах. Это свойство называется автоиндексацией. После соединения терминала данных массива с терминалом выхода из цикла каждая итерация цикла создает новый элемент массива. На иллюстрации видно, что проводник данных, соединяющий терминал данных массива с терминалом выхода из цикла стал толще, а сам терминал выхода из цикла окрашен в цвет терминала данных массива.

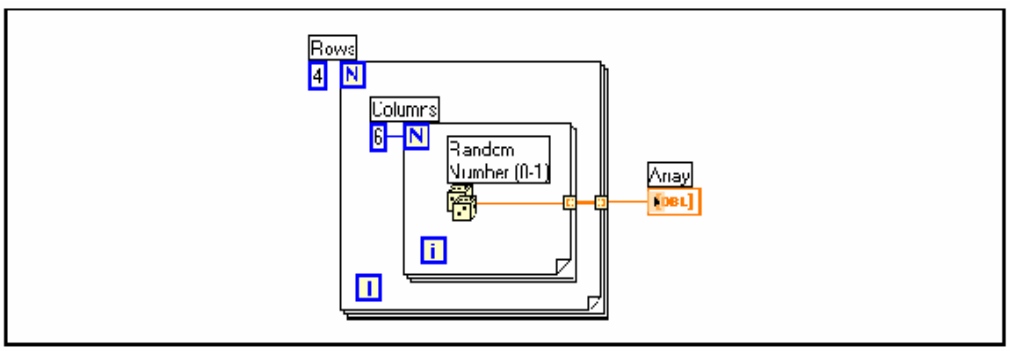


Автоиндексация отключается щелчком правой кнопки мыши по терминалу входа/выхода из цикла и выбором пункта контекстного меню **Disable Indexing**. Автоиндексацию следует отключать, например, в случае, когда нужно знать только последнее значение.

Ввиду того, что цикл **For** часто используется при работе с циклами, для него в LabVIEW автоиндексация включена по умолчанию. Для цикла **While** автоиндексация по умолчанию отключена. Для того, чтобы включить автоиндексацию, необходимо щелкнуть правой кнопкой мыши по терминалу входа/выхода из цикла и выбрать в контекстном меню пункт **Enable Indexing**.

Создание двумерных (2D) массивов

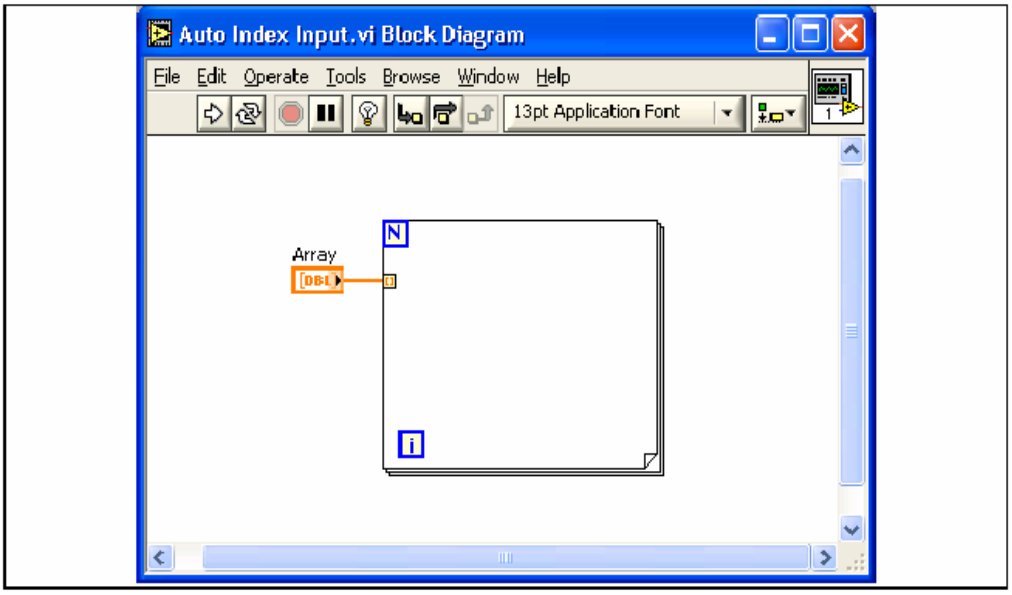
Для создания двумерных массивов необходимо использовать два цикла **For**, один внутри другого. Как показано на иллюстрации, внешний цикл создает элементы массива в строке, а внутренний цикл создает элементы массива в столбце.



Использование автоиндексации для установки значения терминала количества итераций цикла

При включенной автоиндексации массива, подключенного к терминалу входа в цикл **For**, LabVIEW автоматически устанавливает значение терминала количества итераций цикла **N** равным размерности массива. Таким образом, отпадает необходимость задания значения терминалу **N**.

В следующем примере цикл **For** будет выполнен ровно столько раз, сколько элементов в массиве. Как правило, стрелка на кнопке **Run** сломана, если терминал количества итераций цикла не подключен, однако в этом примере стрелка цела, что говорит о возможности запуска ВП.



Если автоиндексация установлена более чем для одного терминала входа в цикл или явно задано значение терминала количества итераций цикла **N**, то значением терминала **N** станет меньшая из величин. Например, если соединить массив из 10 элементов с терминалом входа в цикл, а значение терминала количества итераций установить равным 15, то цикл выполнит 10 итераций.

C. Функции работы с массивами

Для создания и управления массивами используются функции, расположенные в палитре **Functions»Programming»Array**. Наиболее часто используемые функции работы с массивами включают в себя:

- **Array Size -** показывает количество элементов массива каждой размерности. Если массив n-мерный, на выходе функции **Array Size** будет массив из n элементов. Например, для приведенного ниже массива функция **Array Size** выдаст значение 3.

7	4	2

- **Initialize Array -** создает n-мерный массив, в котором каждый элемент инициализирован значением поля ввода данных **element.** Для увеличения размерности массива достаточно добавить поля ввода данных, растянув узел функции. Например, если для функции **Initialize Array** заданы следующие значения параметров: на поле **element** подается значение 4, а на поле **dimension size** (если оно одно) - значение 3, то на выходе получится массив, показанный ниже.

4	4	4

- **Build Array -** объединяет несколько массивов или добавляет элемент в n-мерный массив. Изменение размера функции увеличивает количество полей ввода данных, что позволяет увеличить количество добавляемых элементов. Например, если объединить два предыдущих массива, то функция **Build Array** выдаст на выходе следующий массив.

7	4	2
4	4	4

Для объединения входных данных в более длинный массив той же размерности, как показано ниже, достаточно щелкнуть правой кнопкой мыши на функции и выбрать из контекстного меню пункт **Concatenate Inputs**.

7	4	2	4	4	4

- **Array Subset -** выдает часть массива, начиная с индекса, поступившего на поле **index,** и длиной, указанной в поле **length**. Например, если подать предыдущий массив на поле ввода функции **Array Subset**, значение 2 – на поле **index** и 3 – на поле **length**, то на поле вывода данных будет следующее

подмножество

2	4	4

- **Index Array -** выдает элемент, соответствующий индексу, значение которого подается на поле ввода **index.** Например, при использовании предыдущего массива, функция **Index Array** выдаст значение 2, если на поле ввода данных **index** подать значение 0.

  Функцию **Index Array** можно использовать для выделения строки или столбца из двумерного массива и дальнейшего отображения в виде подмассива. Для этого двумерный массив надо подать в поле ввода данных функции. Функция **Index Array** должна иметь два поля **index.** Верхнее поле **index** указывает строку, а нижнее – столбец. Можно задействовать оба поля **index** для выбора отдельного элемента или только одно, для выбора строки или столбца. Например, в поле ввода данных функции подается массив, показанный ниже.

7	4	2
4	4	4

  Функция **Index Array** в поле вывода данных выдаст следующий массив в случае, если на поле **index** (строка) подается значение 0.

7	4	2

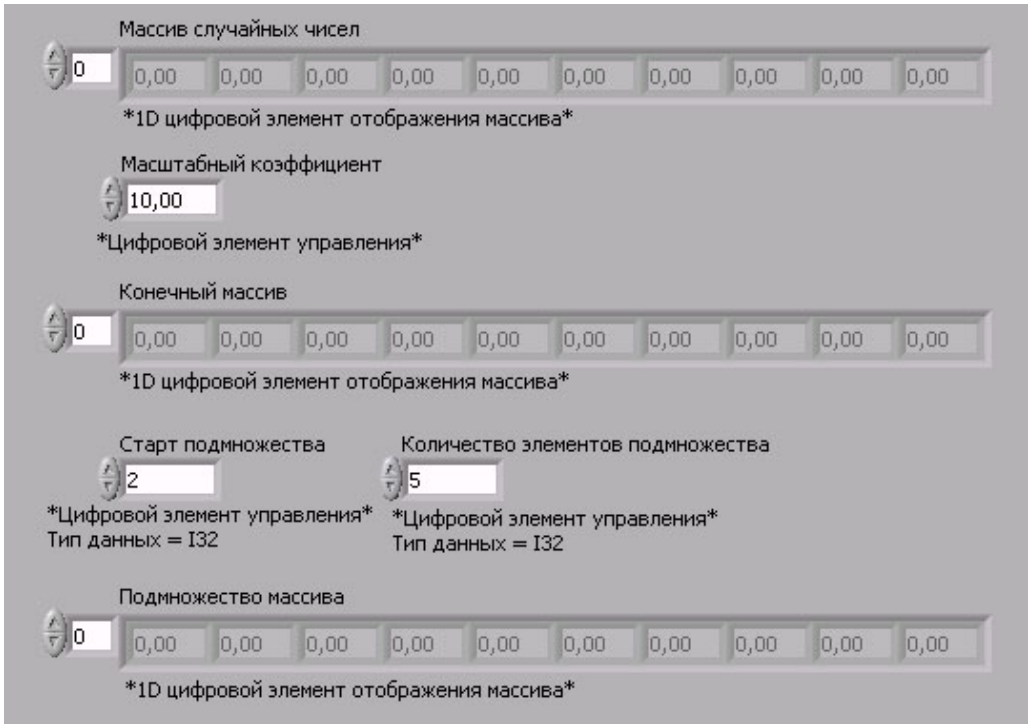
Упражнение 5-1 ВП Работа с массивами

Цель: Создание массивов и знакомство с функциями обработки массивов

Выполните следующие шаги для создания ВП, который формирует массив случайных чисел, масштабирует полученный массив и выделяет из него подмножество.

Лицевая панель

1. Откройте новый ВП и создайте лицевую панель, как показано ниже.



   a. В палитре **Controls»Modern»Array, Matrix & Cluster** выберите шаблон массива.

   b. Созданному массиву присвойте имя **Массив случайных чисел**.

   c. Поместите внутрь шаблона массива цифровой элемент отображения, расположенный в палитре **Controls»Modern»Numeric**.

   d. С помощью инструмента ПЕРЕМЕЩЕНИЕ измените размер массива таким образом, чтобы он содержал 10 элементов.

   e. Нажмите и удерживайте клавишу <**Ctrl**> и, перемещая элемент **Массив случайных чисел**, создать две его копии.

   f. Копиям присвойте имена **Конечный Массив** и **Подмножество Массива**.

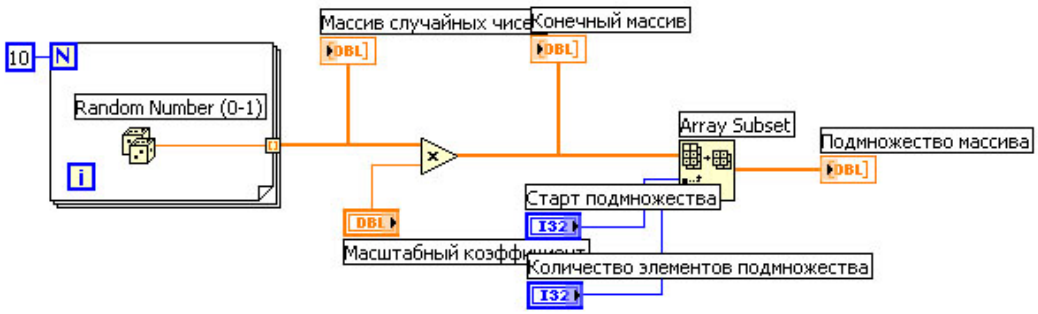
g. Создайте три цифровых элемента управления и присвойте им имена **Масштабный коэффициент**, **Старт подмножества** и **Количество элементов подмножества**.

h. Щелкните правой кнопкой мыши по элементам **Старт подмножества** и **Количество элементов подмножества**, в контекстном меню выберите пункт **Representation,** затем пункт **I32.**

i. Значения элементов управления данных пока не изменяйте.

Блок-диаграмма

2. Постройте блок-диаграмму, как показано ниже.



Выберите функцию **Random Number (0-1),** расположенную в палитре **Functions»Programming»Numeric**. Эта функция будет генерировать случайное число в пределах от 0 до 1.

Выберите цикл **For,** расположенный в палитре **Functions»Programming»Structures**. Этот цикл на терминале выхода накапливает массив из 10 случайных чисел. Терминалу количества итераций присвойте значение 10.

Выберите функцию **Array Subset,** расположенную в палитре **Functions»Programming»Array**. Эта функция выдает подмножество массива, начиная со значения, введенного в элементе **Старт подмножества** и будет содержать количество элементов, указанное в элементе **Количество элементов подмножества**.

3. Сохраните ВП под именем *Работа с массивами.vi*

Запуск ВП

4. Перейдите на лицевую панель, измените значения элементов управления и запустите ВП.

   Цикл **For** совершит 10 итераций. Каждая итерация создаст случайное число и сохранит его в терминале выхода из цикла. В элементе **Массив случайных чисел** отобразится массив из 10 случайных чисел. ВП умножит каждое значение этого массива на

число, введенное в элемент управления **Масштабный коэффициент**, для создания массива, отображаемого в индикаторе **Конечный массив**. ВП выделит подмножество из получившегося массива, начиная со значения в элементе **Старт подмножества**, длиной, указанной в элементе **Количество элементов подмножества**, и отобразит это подмножество в индикаторе **Подмножество массива**.

5. Закройте ВП.

Конец упражнения 5-1

D. Полиморфизм

Арифметические функции, расположенные в палитре **Functions»Programming»Numeric**, являются полиморфными. Это означает, что на поля ввода этих функций могут поступать данные различных типов (скалярные величины, массивы). Например, можно использовать функцию **Add** для прибавления скалярной величины к массиву или сложения двух массивов. Если на одно поле ввода данных функции **Add** подать скалярную величину 2, а другое соединить с массивом, показанным ниже,

1	3	2

то функция прибавит 2 к каждому элементу массива, и массив будет иметь вид:

3	5	4

Если на вход функции **Add** подать два предыдущих массива, функция сложит каждый элемент первого массива с соответствующим элементом второго и выдаст результат в виде массива, показанного ниже.

4	8	6

Если с помощью функции **Add** сложить два массива разной размерности, например, таких, как предыдущий и показанный ниже,

3	1	2	3

то функция сложит каждый элемент первого массива с соответствующим элементом второго и выдаст результат в виде массива размерностью меньшей из двух исходных.

7	9	8

С кластерами арифметические функции работают таким же образом. Подробнее о работе с кластерами можно узнать на Уроке 6, *Кластеры*.

Краткое изложение пройденного материала, советы и секреты

- Массивы объединяют группу элементов одного типа данных. Данные, составляющие массив, могут быть любого типа: целочисленные, логические или строковые. Массив также может содержать сигнальные данные и кластеры.
- Индекс первого элемента массива всегда равен 0. Таким образом, индексы массива находятся в диапазоне от 0 до (n-1), где n – число элементов в массиве.
- Поместить объект в шаблон массива следует до того, как использовать этот массив на блок-диаграмме. Иначе на блок-диаграмме шаблон массива не будет инициализирован.
- Для создания массива элементов управления или отображения необходимо выбрать его шаблон из палитры **Controls»Modern»Array, Matrix & Cluster** и поместить его на лицевую панель. Затем поместить в шаблон массива элемент управления либо отображения.
- Если массив соединить с терминалом входа цикла **For** или цикла **While**, то при включенной автоиндексации можно читать и обрабатывать каждый элемент массива.
- Для создания и управления массивами используются функции, расположенные на палитре **Functions»Programming»Array**.
- LabVIEW осуществляет автоиндексацию автоматически. Автоиндексация для цикла **For** включена по умолчанию. Для цикла **While** автоиндексация по умолчанию отключена.
- Полиморфизм означает, что поля ввода данных функций могут различаться по структуре данных (скалярные величины, массивы).

Дополнительные упражнения

5-2. Создайте ВП, который полностью изменяет порядок элементов в массиве, содержащем 100 случайных чисел. Например, элемент массива с индексом 0 становится элементом массива с индексом 99, а элемент массива с индексом 1 становится элементом массива с индексом 98, и так далее.

Совет. Для изменения порядка данных в массиве следует использовать функцию **Reverse 1D Array**, расположенную на палитре **Functions»Programming»Array**.

Сохраните ВП под именем *Перевернутый массив случайных чисел.vi*

5-3. Создайте ВП, определяющий и отображающий текущее среднее, максимум и минимум для последовательности данных с генератора случайных чисел. Прекращение работы по кнопке на передней панели.

Сохраните ВП под именем *Работа генератора случайных чисел.vi*

5-4. Создайте ВП, который накапливает массив значений температур с помощью *Термометр.vi* (упражнение 3-3). Размер массива определяйте с помощью элемента управления на лицевой панели. С помощью функции **Initialize Array** создайте массив такого же размера, в котором все элементы имеют значение 10. Сложите два массива, найдите размер конечного массива и вычислите его среднее значение. На лицевую панель выведите массив значений температуры, инициализированный массив, конечный массив и среднее значение.

Сохраните ВП под именем *Найти среднее значение.vi*

5-5. Создайте ВП, который генерирует двумерный массив случайных чисел, содержащий 3 строки и 10 столбцов. После этого ВП индексирует все строки и выводит осциллограмму каждой из них на отдельном графике. На лицевой панели должно быть три графика.

Сохраните ВП под именем *Выделение 2D массива.vi*

5-6. Создайте ВП, который имитирует бросок игральной кости с гранями 1 - 6 и записывает в массив выпавшие значения после каждого броска. На входе – количество бросков кости, а на выходе – количество выпадений каждой грани. Использовать только один сдвиговый регистр (**shift register**).

Сохраните ВП под именем *Игра в кости.vi*

5-7 Создайте ВП, который генерирует одномерный массив и затем попарно перемножает элементы, начиная с элементов с индексами 0 и 1 и т.д., а затем выводит результаты в массив элементов отображения данных. Например, входной массив имеет значение {1, 23, 10, 5, 7, 11}, а в результате получается массив {23, 50, 77}.

Совет. Используйте функцию **Decimate ID Array**, расположенную в палитре **Functions»Programming»Array**.

Сохраните ВП под именем *Попарное перемножение элементов массива.vi*

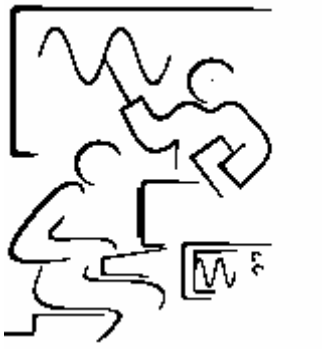
5-8 Создайте ВП, перемножающий поэлементно две матрицы размером 3x3.

Сохраните ВП под именем *Перемножение элементов матриц.vi*

5-9 Создайте ВП, генерирующий массив из N=100 случайных чисел, вычисляющий их среднее, стандартное отклонение и автокорреляционную функцию $y(m)=\sum_{n=0}^{N-m}x(n)x(n+m)$ без использования встроенных функций.

Сохраните ВП под именем *характеристики генератора случайных чисел.vi*

Примечания



Урок 6
Кластеры

На этом уроке рассказывается об объединении элементов различных типов данных в кластеры.

В этом уроке изложены вопросы:

A. Что такое кластеры.

B. Использование функций работы с кластерами.

C. Кластеры ошибок.

A. Кластеры

Кластеры объединяют элементы разных типов данных, подобно пучку проводов телефонного кабеля, где каждый провод представляет собой отдельный элемент кластера. Кластеры играют ту же роль, что и структуры в языках программирования.

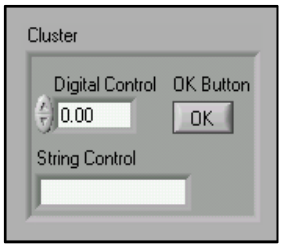
Объединение нескольких групп данных в кластер устраняет беспорядок на блок-диаграмме и уменьшает количество полей ввода/вывода данных, необходимых подпрограмме ВП. Максимально возможное количество полей ввода/вывода данных ВП равно 28. Если лицевая панель содержит более 28 элементов, которые необходимо использовать в ВП, можно некоторые из них объединить в кластер и связать кластер с полем ввода/вывода данных. Как и массив, кластер может быть элементом управления или отображения данных, однако кластер не может одновременно содержать элементы управления и отображения данных.

В кластере, как и в массиве, все элементы упорядочены, но обратиться по индексу к ним нельзя, необходимо сначала разделить их. Для этого предназначена функция **Unbundle By Name**, которая обеспечивает доступ к определенным элементам кластера по их имени.

Создание кластеров из элементов управления и отображения данных

Для создания кластеров из элементов управления и отображения данных следует выбрать шаблон кластера на палитре **Controls»Modern»Array, Matrix & Cluster** и поместить его на лицевую панель. После этого шаблон кластера следует заполнить элементами. Изменить размер кластера можно с помощью курсора.

Ниже показан кластер, содержащий три элемента управления.

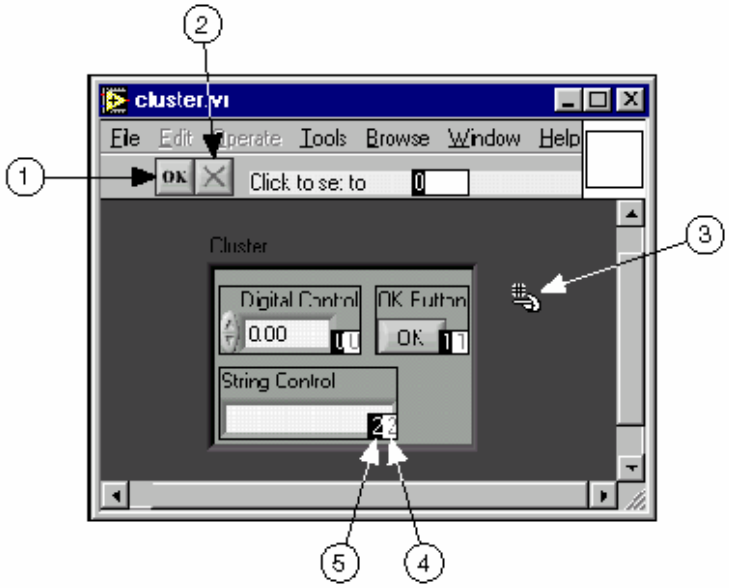


Порядок элементов в кластере

Каждый элемент кластера имеет свой логический порядковый номер, не связанный с положением элемента в шаблоне. Первому помещенному в кластер элементу автоматически присваивается номер 0, второму элементу – 1 и так далее. При удалении элемента порядковые номера автоматически изменяются.

Порядок элементов в кластере определяет то, как элементы кластера будут распределены по терминалам функций **Bundle** (объединения) и **Unbundle** (разделения) на блок-диаграмме.

Посмотреть и изменить порядковый номер объекта, помещенного в кластер, можно, щелкнув правой кнопкой мыши по краю кластера и выбрав из контекстного меню пункт **Reorder Controls In Cluster**. Панель инструментов и кластер примут вид, показанный ниже на рисунке.



1. Кнопка подтверждения (Confirm button)
2. Кнопка отмены (Cancel button)
3. Курсор определения порядка (Cluster order cursor)
4. Текущий порядковый номер (Current order)
5. Новый порядковый номер (New order)

В белом поле (4) указан текущий порядковый номер элемента, в черном (5) – новый порядковый номер. Для установки порядкового номера элемента нужно в поле ввода текста **Click to set to** ввести число и нажать на элемент. Порядковый номер элемента изменится. При этом корректируются порядковые номера других элементов. Сохранить изменения можно, нажав кнопку **Confirm** на панели инструментов. Вернуть первоначальные установки можно, нажав кнопку **Cancel**.

Соответствующие элементы, определенные в кластерах одинаковыми порядковыми номерами, должны иметь совместимые типы данных. Например, в одном кластере элемент 0 является числовым элементом управления, а элемент 1 – строковым элементом управления. Во втором кластере элемент 0 – числовой элемент отображения данных и элемент 1 – строковый элемент отображения данных. Кластер элементов управления корректно соединится с кластером элементов отображения данных.

Однако если изменить порядковые номера элементов в одном из кластеров, проводник данных между кластерами будет разорван, так как типы данных элементов кластеров не будут соответствовать друг другу.

Создание кластера констант

На блок-диаграмме можно создать кластер констант, выбрав в палитре **Functions»Programming»Cluster & Variant** шаблон **Cluster Constant** и поместив в него числовую константу или другой объект данных, логический или строковый.

Если на лицевой панели кластер уже существует, то кластер констант на блок-диаграмме, содержащий те же элементы, можно создать, просто перетащив кластер с лицевой панели на блок-диаграмму или, щелкнув правой кнопкой мыши на кластере, выбрать из контекстного меню пункт **Create»Constant**.

B. Функции работы с кластерами

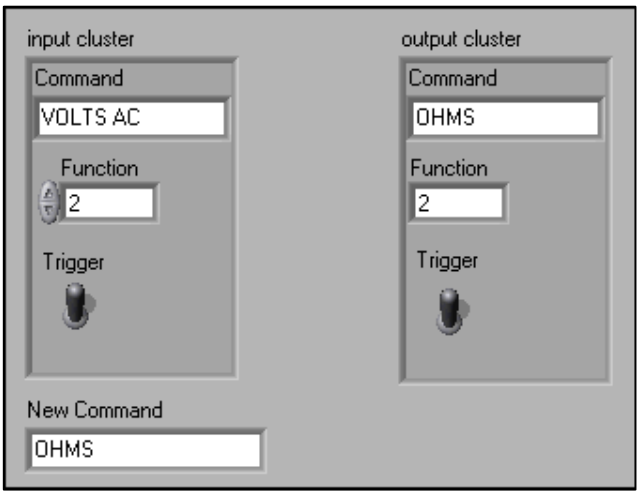
Для создания и управления кластерами используются функции, расположенные на палитре **Functions»Programming»Cluster & Variant**. Функции **Bundle** и **Bundle by Name** используются для сборки и управления кластерами. Функции **Unbundle** и **Unbundle by Name** используются для разборки кластеров.

Эти функции также можно вызвать, щелкнув правой кнопкой мыши по терминалу данных кластера и выбрав из контекстного меню подменю **Cluster Tools**. Функции **Bundle** и **Unbundle** автоматически содержат правильное количество полей ввода/вывода данных. Функции **Bundle by Name** и **Unbundle by Name** в полях ввода/вывода данных содержат имя первого элемента кластера.

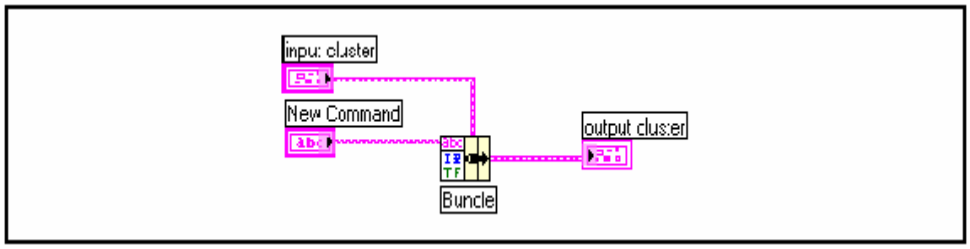
Сборка кластеров

Для сборки отдельных элементов в кластер используется функция **Bundle**. Эта же функция используется для изменения данных в элементе уже существующего кластера. Инструмент ПЕРЕМЕЩЕНИЕ используется для добавления полей ввода данных, для этого также можно щелкнуть правой кнопкой по полю ввода данных и выбрать из контекстного меню пункт **Add Input**. При соединении кластера с полем ввода данных **cluster** количество полей ввода данных функции должно соответствовать количеству элементов во входящем кластере.

На поле ввода данных **cluster** можно подать только одну требующую замены компоненту. Например, ниже показан кластер, имеющий три элемента управления.



Если известен логический порядок элементов, можно использовать функцию **Bundle** для изменения значения элемента **Command**, соединив элементы, как показано ниже.

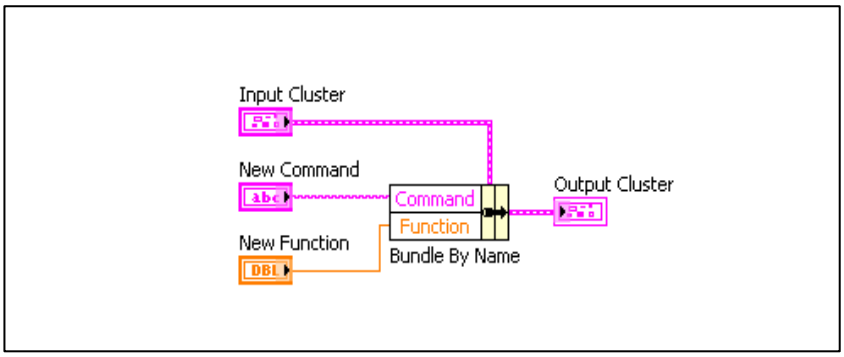


Замена или доступ к элементам кластера

Для замены элемента в уже существующем кластере используется функция **Bundle by Name**. Функция **Bundle by Name** работает так же как функция **Bundle**, но вместо обращения к элементу кластера по его порядковому номеру обращается к нему по его собственной метке (имени). При этом можно получить доступ только к элементам, имеющим собственную метку. Количество полей ввода данных не требует соответствия с количеством элементов в кластере.

С помощью элемента УПРАВЛЕНИЕ можно щелкнуть по полю ввода данных терминала и выбрать желаемый элемент из выпадающего меню. Можно также щелкнуть правой кнопкой мыши по полю ввода данных и выбрать элемент в разделе контекстного меню **Select Item**.

Ниже показано, как можно использовать функцию **Bundle by Name** для изменения значений элементов **Command** и **Function**.



Использовать функцию **Bundle by Name** следует при работе со структурами данных, которые могут меняться в процессе работы. Чтобы добавить новый элемент в кластер или изменить порядковый номер элемента, нет необходимости вновь подключать функцию **Bundle by Name**, так как имя элемента все еще действительно.

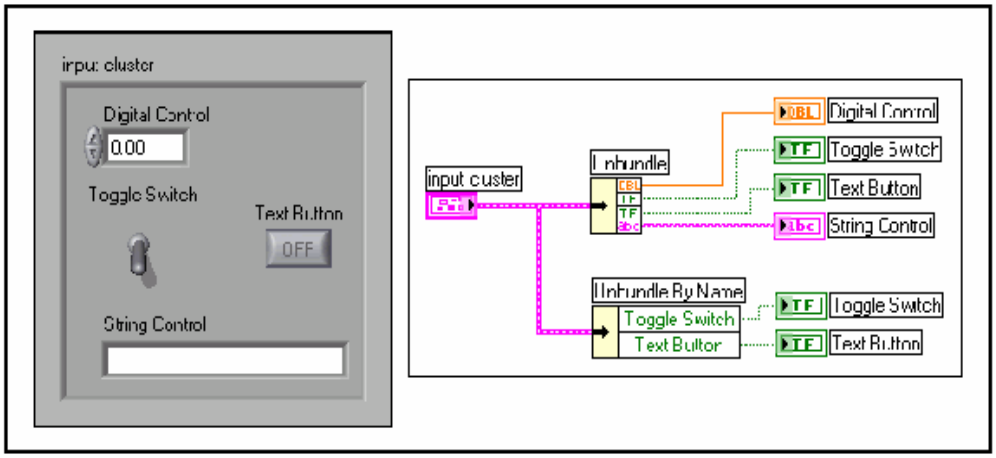
Разделение кластера

Функция **Unbundle** используется для разбиения кластеров на отдельные элементы.

Функция **Unbundle by Name** используется для выделения из кластера элементов по определенному имени. Количество полей вывода данных не зависит от количества элементов в кластере.

С помощью инструмента УПРАВЛЕНИЕ можно щелкнуть по полю вывода данных и выбрать желаемый элемент из контекстного меню. Можно также щелкнуть правой кнопкой мыши по полю вывода данных и выбрать из контекстного меню пункт **Select Item**.

Например, функция **Unbundle**, при использовании кластера, показанного ниже, имеет четыре поля вывода данных, которые соотносятся с четырьмя элементами кластера. Необходимо знать порядок элементов в кластере для корректного сопоставления логического элемента соответствующему вертикальному переключателю в кластере. В этом примере элементы упорядочены сверху вниз, начиная с 0. Если использовать функцию **Unbundle by Name**, то полей вывода данных может быть произвольное количество, и обращаться к отдельным элементам можно в произвольном порядке.

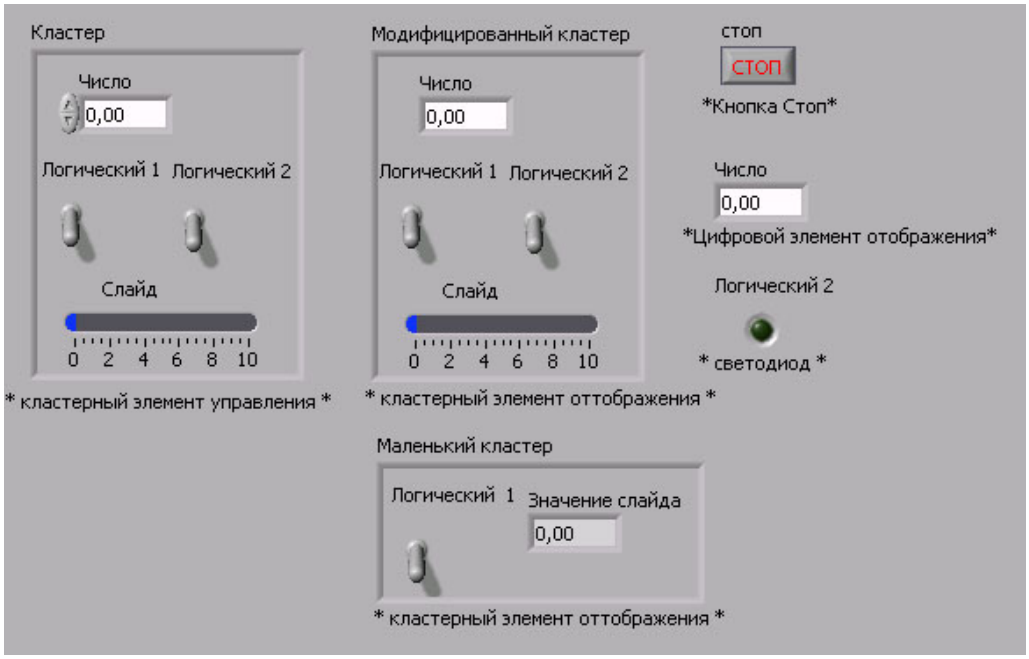


Упражнение 6-1 ВП Работа с кластерами

Цель: Создание кластеров на лицевой панели. Используя функции обработки кластеров, собирать и демонтировать кластеры

Лицевая панель

1. Откройте новый ВП и создайте лицевую панель, как показано ниже.



   a. Поместите на лицевую панель кнопку «Стоп» и круглый светодиод.

   b. Из палитры **Controls»Modern»Array, Matrix & Cluster** выберите шаблон кластера.

   c. Объекты лицевой панели, показанные на иллюстрации, поместите в шаблон кластера.

   d. Создайте и переименуйте копию элемента **Кластер** в **Модифицированный Кластер**. После этого щелкните правой кнопкой мыши по границе шаблона кластера **Модифицированный Кластер** и выберите из контекстного меню пункт **Change to Indicator**.

   e. Повторите пункт d для создания элемента **Маленький кластер**. Измените его, как показано на рисунке.

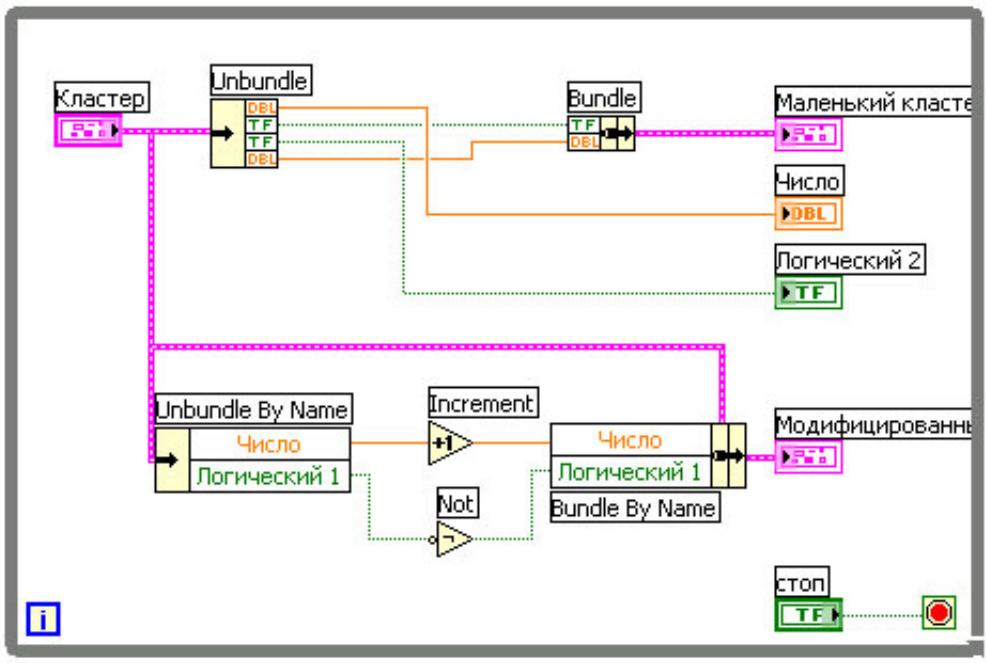
2. Проверьте порядковые номера элементов в кластерах **Кластер** и **Маленький кластер**. Порядковые номера элементов кластера **Модифицированный кластер** и **Кластер** должны совпадать.

   a. Щелкните правой кнопкой мыши по границе шаблона каждого кластера, из контекстного меню выберите пункт **Reorder Controls in Cluster**.

b. Порядковые номера элементов установите, как показано ниже на рисунке.

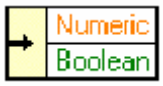
Блок-диаграмма

3. Создайте блок-диаграмму, как показано ниже.



Из палитры **Functions»Programming»Cluster & Variant** выберите функцию **Unbundle**. Эта функция разъединяет кластер **Кластер**. Измените размер этой функции до четырех полей ввода данных или соедините терминал данных кластера с функцией для автоматического добавления полей ввода данных.

Из палитры **Functions»Programming»Cluster & Variant** выберите функцию **Bundle**. Эта функция объединит элементы в кластер **Маленький кластер**.



Из палитры **Functions»Programming»Cluster & Variant** выберите функцию **Unbundle by Name**. Эта функция выделит два элемента из кластера **Кластер**. Измените размер функции до двух полей вывода данных. Если имена в полях вывода данных отличаются от показанных на иллюстрации, следует щелкнуть правой кнопкой мыши по имени элемента и в контекстном меню войти в раздел **Select Item**.

Из палитры **Functions»Programming»Numeric** выберите функцию **Increment.** Эта функция добавит 1 к значению элемента **Число**.

Из палитры **Functions»Programming»Boolean** выберите функцию **Not**. Эта функция выдаст логическое отрицание элемента **Логический 1**.

Из палитры **Functions»Programming»Cluster & Variant** выберите функцию **Bundle by Name**. Эта функция изменит значения элементов **Число** и **Логический** в кластере **Кластер** и создаст кластер **Модифицированный кластер**. Измените размер этой функции на два поля ввода данных. Если имена в полях вывода данных отличаются от показанных на иллюстрации, следует щелкнуть правой кнопкой мыши по имени элемента и в контекстном меню войти в раздел **Select Item**.

4. Сохраните ВП под именем *Работа с кластерами. vi*

Запуск ВП

5. Перейдите на лицевую панель и запустите ВП.
6. Поменяйте значения элементов в кластере **Кластер** и запустите ВП.
7. Закройте ВП.

Конец упражнения 6-1

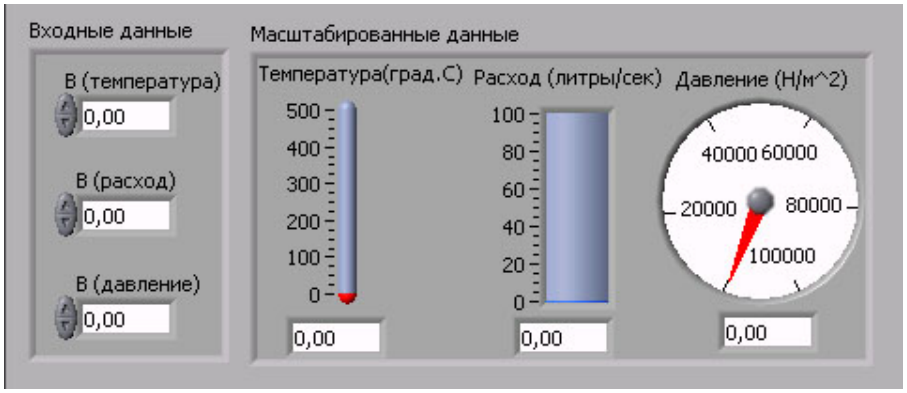
Упражнение 6-2 ВП Масштабирование кластера (дополнительное)

Цель: Создать ВП, использующий полиморфизм в кластерах

Выполните следующие шаги для создания ВП, в котором масштабируются данные, хранящиеся в кластере. Каждый элемент в кластере имеет свой масштабный коэффициент. Предположим, что исходные данные, значения давления, скорости потока и температуры были получены с соответствующих датчиков напряжения. Затем ВП масштабирует эти значения и выдает фактические значения физической величины.

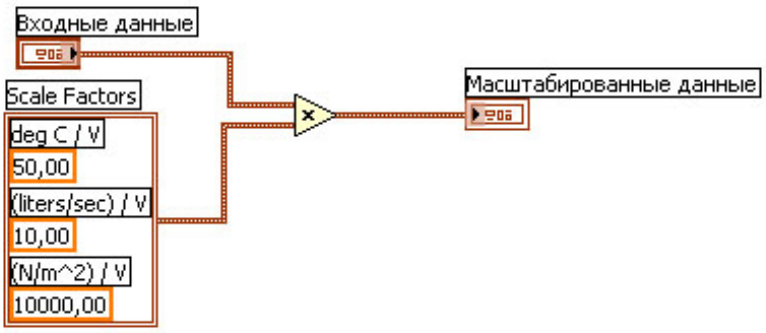
Лицевая панель

1. Откройте ВП *Масштабирование кластера.vi*, расположенный в папке *C:\Exercises\LV Basics I*. Лицевая панель уже подготовлена.
2. Измените значения элементов управления на лицевой панели.



Блок-диаграмма

3. Создайте блок-диаграмму, как показано ниже. Убедитесь, что значения элементов кластера **Входные данные** введены правильно.



4. Сохраните ВП.
5. Перейдите на лицевую панель и запустите ВП.

6. Поменяйте значения элементов управления и снова запустите ВП.
7. Закройте ВП.

Конец упражнения 6-2

C. Кластеры ошибок

Даже в самой отлаженной программе встречаются ошибки, поэтому никогда нельзя предусмотреть все проблемы, которые могут возникнуть у пользователя. Без механизма проверки ошибок, о ВП можно сказать только то, что он не работает. Проверка ошибок позволяет узнать, в каком месте и почему произошел сбой.

При программировании любых операций ввода/вывода стоит подумать о возможном появлении ошибок. Почти все операции ввода/вывода возвращают информацию об ошибке. Чтобы правильно обрабатывать ошибки, в ВП нужно особо тщательно выполнять проверку для таких операций ввода/вывода, как файловые и последовательные операции, операции работы с приборами, операции получения данных, а также процессы передачи информации.

Проверка на ошибки в ВП может помочь определить следующие проблемы:

- Неправильная инициализация связи с внешним устройством или запись в него некорректной информации.
- Внешнее устройство не включено или не работает.
- При переустановке системного программного обеспечения или по другим причинам был изменен путь к необходимым файлам.

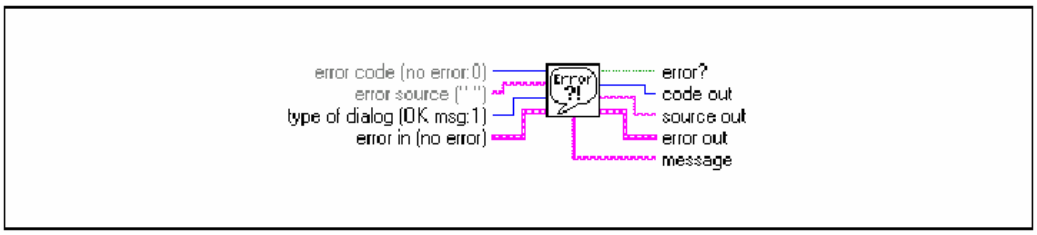
Обработка ошибок

В LabVIEW не реализована автоматическая обработка ошибок. Это сделано для того, чтобы можно было самостоятельно выбирать метод, которым обрабатываются ошибки. Например, если для ВП истекло время ожидания ввода/вывода, можно сделать так, чтобы не прекращалась работа всего приложения. Можно также заставить ВП повторить попытку через некоторое время. Процесс обработки ошибок в LabVIEW происходит на блок-диаграмме.

Существует два способа возврата ошибок в ВП и функциях: с помощью числа, обозначающего код ошибки и с помощью кластера ошибок. Как правило, функции используют число - код ошибки, а ВП принимают на вход и выдают на выходе информацию об ошибках в виде кластера.

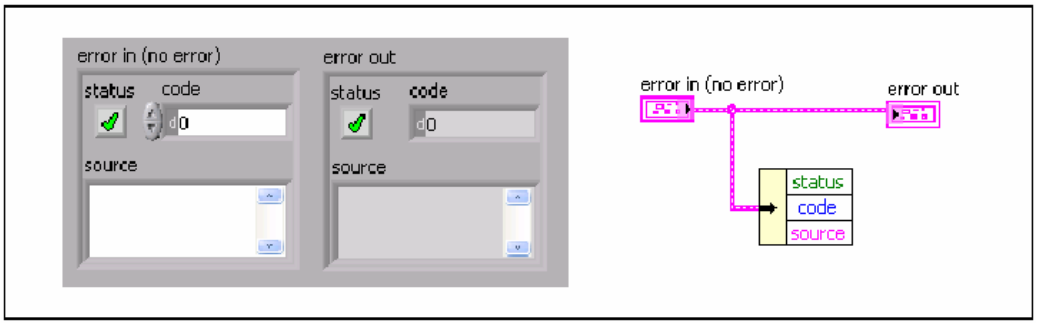
Обработка ошибок в LabVIEW также построена на модели поточного программирования. Как и другие данные, информация об ошибках проходит через ВП. Для передачи информации об ошибках через ВП необходимо использовать входной и выходной кластеры ошибок, а также включить в конце ВП обработчик ошибок для определения того, были ли сбои в процессе работы ВП.

При выполнении ВП LabVIEW следит за появлением ошибок, и, как только где-нибудь происходит сбой, составляющие части ВП перестают выполняться и только передают ошибку дальше, на выход. Для обработки появляющихся в ВП ошибок в конце потока выполнения обычно используется приведенный ниже простой обработчик ошибок **Simple Error Handler**. **Simple Error Handler** находится на палитре **Functions»Programming»Dialog & User Interface**. Подсоедините кластер ошибок к полю входных данных «**Error In**» (по умолчанию ошибки нет).



Кластеры ошибок

Ниже приведены компоненты кластеров ошибок, расположенных на палитре **Controls»Modern»Array, Matrix & Cluster**. Кластер ошибок и проводники, соединяющие кластеры ошибок, обозначаются желтым цветом.



- **status** является логической величиной, принимающей значение TRUE в случае возникновения ошибки. Большинство ВП, функций и структур, которые принимают логические данные, используют этот параметр. При возникновении ошибки кластер ошибок передает функции значение TRUE.
- **code** является целым 32-х битным числом со знаком, которое соответствует ошибке. В случае если **status** имеет значение FALSE, а **code** отличен от нуля, то, скорее всего, это предупреждение, а не фатальная ошибка.
- **source** является строкой, которая определяет место возникновения ошибки.

Для создания входа и выхода ошибок в подпрограммах ВП используются кластеры ошибок из элементов управления и отображения.

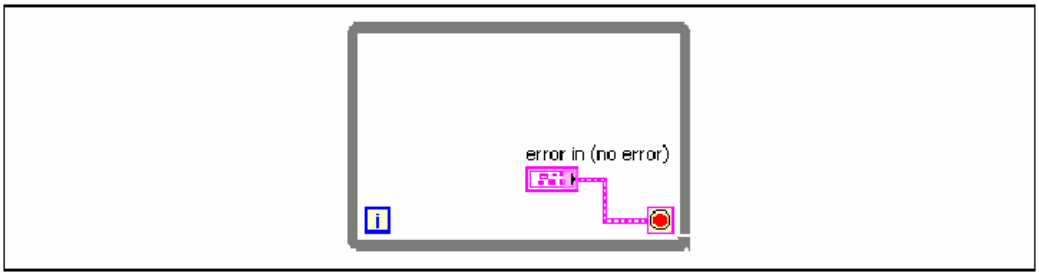
Объяснение ошибки

При появлении ошибки можно щелкнуть правой кнопкой мыши внутри кластера и из контекстного меню выбрать пункт **Explain Error**. Появится диалоговое окно **Explain Error**, содержащее информацию об ошибке. В контекстном меню также есть пункт **Explain Warning**, если в ВП нет ошибок, но есть предупреждения.

Диалоговое окно **Explain Error** также можно вызвать из меню **Help**.

Использование цикла While при обработке ошибок

Кластер ошибок может быть подсоединен к терминалу условия цикла **While** для остановки цикла. Когда кластер ошибок подсоединен к терминалу условия, на терминал подаются только значения параметра **status** – TRUE или FALSE. При возникновении ошибки выполнение цикла **While** прекращается.

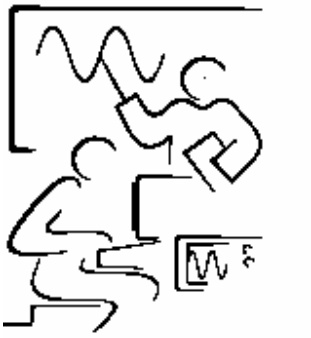


Если к терминалу условия подсоединен кластер ошибок, пункты контекстного меню меняются с **Stop if True** и **Continue if True** на **Stop on Error** и **Continue while Error**.

Краткое изложение пройденного материала, советы и секреты

- Кластеры объединяют элементы разных типов данных. Кластер не может одновременно содержать элементы управления и отображения данных.
- Если лицевая панель содержит более 28 элементов, которые необходимо использовать в ВП, то некоторые из них можно объединить в кластер и назначить кластер полем ввода/вывода данных.
- Для создания кластеров из элементов управления и отображения данных следует выбрать шаблон кластера на палитре **Controls»Modern»Array, Matrix & Cluster** и поместить его на лицевую панель. После этого шаблон кластера следует заполнить элементами.
- Для создания и управления кластерами используются функции, расположенные на палитре **Functions»Programming»Cluster & Variant**.
- Проверка ошибок позволяет узнать, в каком месте и почему произошел сбой.
- Кластер ошибок содержит элементы **status** (статус), **code** (код) и **source** (источник) ошибки.
- Для создания входа и выхода ошибок в подпрограммах ВП используются кластеры ошибок из элементов управления и отображения.

Примечания



Урок 7
Графическое отображение данных

В этом уроке рассмотрены способы визуализации данных с помощью графика Диаграмм (**Waveform Chart**), графика Осциллограмм (**Waveform Graph**), двухкоординатного графика Осциллограмм (**XY graph**) и графика интенсивности (**Intensity Graph**).

В этом уроке изложены вопросы:

A. Использование графика Диаграмм для отображения потока данных.

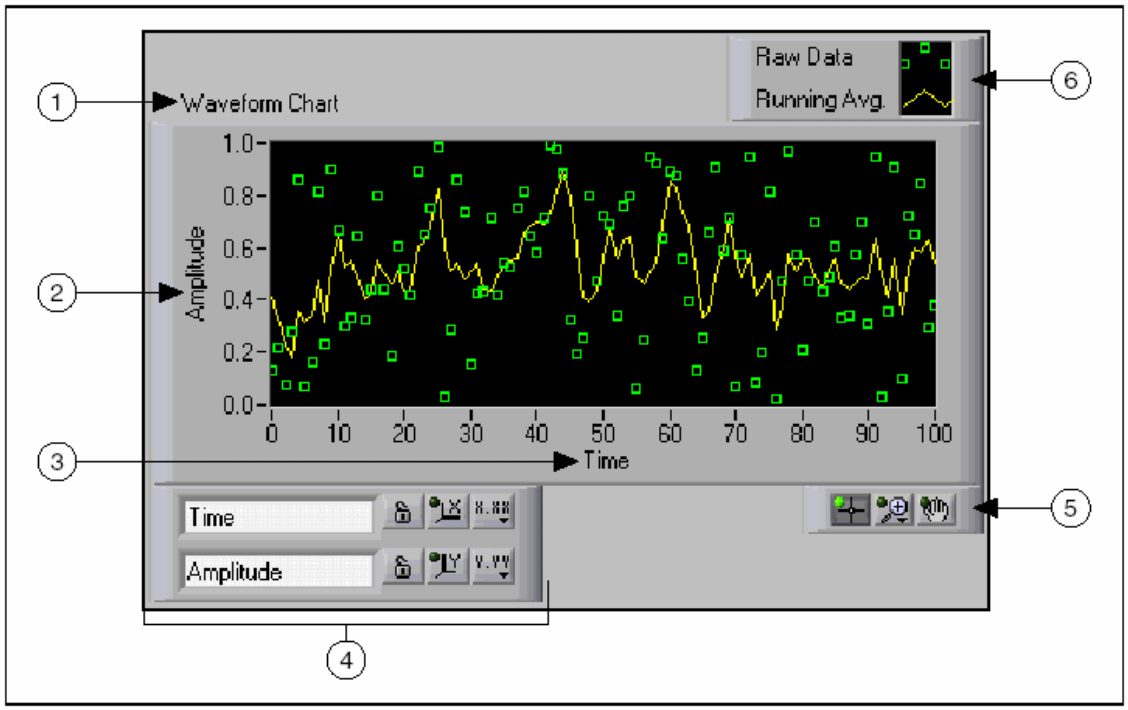
B. Использование графика Осциллограмм и двухкоординатного графика Осциллограмм для отображения данных.

C. График интенсивности (дополнительно).

D. Создание трехмерных сцен (дополнительно).

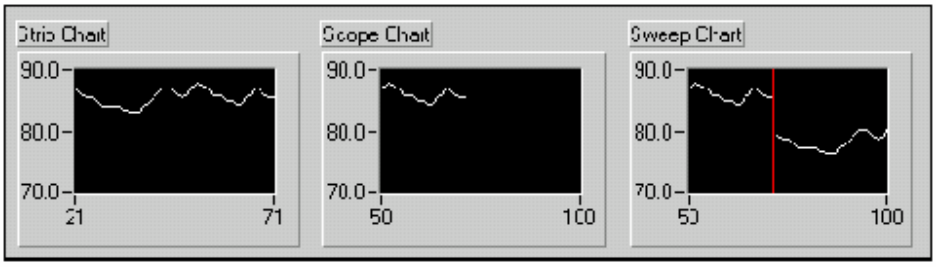
A. График Диаграмм

График Диаграмм (**Waveform Chart**)– специальный элемент отображения данных в виде одного и более графиков. График Диаграмм расположен на палитре **Controls»Modern»Graph**. На рисунке показан пример Графика Диаграмм с двумя графиками: экспериментальные данные и их бегущее среднее значение.



1. Название (Label)
2. Шкала Y (Y-scale)
3. Шкала X (X-scale)
4. Панель управления шкалами (Scale legend)
5. Палитра инструментов для работы с графиком (Graph palette)
6. Панель управления графиком (Plot legend)

График Диаграмм использует три различных режима отображения данных: **strip chart**, **scope chart** и **sweep chart**. Режим по умолчанию – **strip chart.**



Задание режима осуществляется щелчком правой клавишей мыши по диаграмме и выбором пункта **Advanced»Update Mode** из контекстного меню.

Режим **strip chart** представляет собой экран, прокручиваемый слева направо, подобно бумажной ленте. Режимы **scope chart** и **sweep chart** подобны экрану осциллографа и отличаются большей скоростью отображения данных по сравнению с **strip chart.** В режиме **scope chart** по достижении правой границы поле графика очищается, и заполнение диаграммы начинается с левой границы. Режим **sweep chart,** в отличие от режима **scope chart,** не очищает поле графика, а отделяет новые данные от старых вертикальной линией – маркером.

Соединение графиков

Для создания диаграмм достаточно соединить поле вывода скалярной величины с терминалом данных графика Диаграмм. В следующем примере тип данных на терминале графика Диаграмм, соответствует входному типу данных.

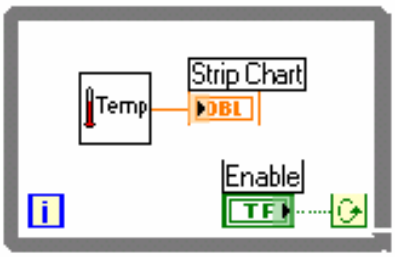
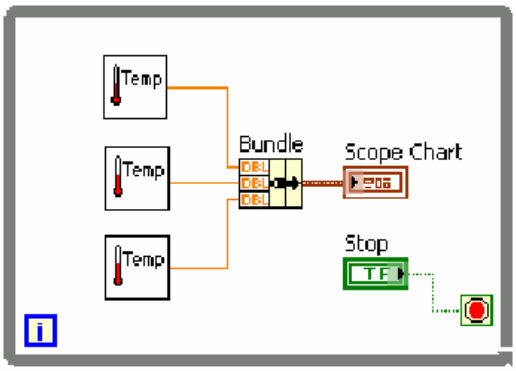


График Диаграмм может отображать несколько графиков. Для объединения отображаемых данных используется функция **Bundle,** расположенная в палитре **Functions»Programming»Cluster & Variant**. Например, блок-диаграмма, показанная ниже, с помощью функции **Bundle** объединяет выходные данные трех подпрограмм ВП для последующего отображения на графике Диаграмм.



Терминал данных графика Диаграмм имеет кластерный тип данных в соответствии с полем вывода функции **Bundle.** Для увеличения количества полей ввода данных функции **Bundle** необходимо с помощью инструмента ПЕРЕМЕЩЕНИЕ изменить ее терминала.

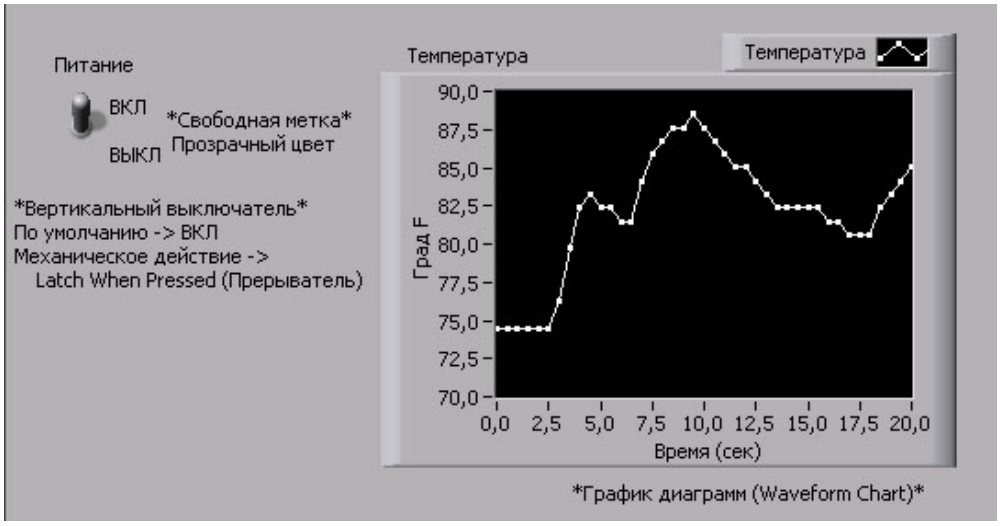
Упражнение 7-1 ВП Мониторинг температуры

Цель: Приобретение навыков по использованию цикла While и графика Диаграмм для получения и отображения данных.

Ниже приведена последовательность действий для создания ВП, который измеряет температуру и отображает результат в виде диаграммы.

Лицевая панель

1. Откройте новый ВП и создайте лицевую панель, как показано ниже на рисунке.



Выберите логический элемент управления – вертикальный переключатель из палитры **Controls»Modern»Boolean** и поместите его на лицевую панель, назвав его **Питание**. Выключатель будет использоваться для остановки выполнения ВП программным способом.

Выберите график Диаграмм (**Waveform Chart**) из палитры **Controls»Modern»Graph** и поместите его на лицевую панель. На графике Диаграмм будет отображаться значение температуры в реальном масштабе времени. Введите текст **Температура** в поле собственной метки графика.

Обратите внимание на то, что на панели управления графиком (**chart legend**) введен текст **Plot 0**. Измените текст на **Температура** с помощью инструмента ВВОД ТЕКСТА.

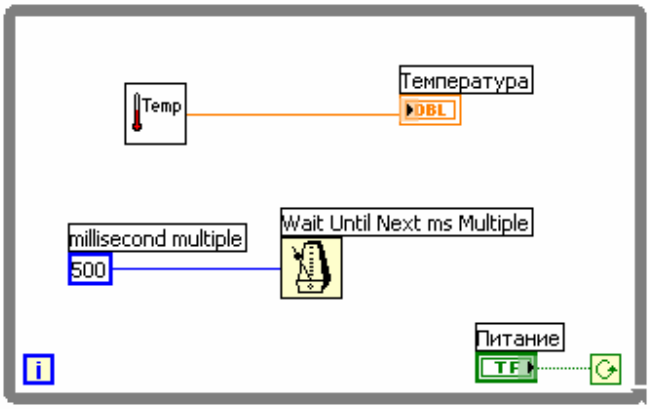
Датчик температуры измеряет комнатную температуру, поэтому с помощью инструмента ВВОД ТЕКСТА измените верхнее значение оси Y с **10** на **90** в °F. Для этого выделите значение **10,0** двойным щелчком инструмента и введите значение **90**.

Измените нижнее значение масштаба оси Y с **-10** на **70** в °F.

Введите названия осей Y – **Град. F**, X – **Время (сек)**.

Блок-диаграмма

2. Перейдите на блок-диаграмму, выбрав пункт главного меню **Window»Show Diagram**.
3. Поместите терминалы данных в тело цикла **While,** как показано ниже.



Наведите курсор на терминал условия выхода из цикла и нажмите правую кнопку мыши. Выберите пункт контекстного меню **Continue if TRUE**.

4. Соедините элементы, как показано выше.

В палитре **Functions»Select a VI** выберите ВП *Термометр.vi* (C*:\exercises\LV Basics I*) и поместите его на блок-диаграмму. ВП на выходе выдает измеренное датчиком значение температуры.

Поместите на блок-диаграмму функцию **Wait Until Next ms Multiple,** размещенную в палитре **Functions»Programming»Timing**.

Щелкните правой кнопкой мышки по полю ввода **millisecond multiple** функции **Wait Until Next ms Multiple** и в контекстном меню выберите пункт **Create»Constant**. Созданной константе присвойте значение **500**. Теперь каждая итерация цикла будет выполняться с интервалом времени 500 мс (один раз каждые полсекунды).

Примечание Для измерения температуры в °C необходимо поле ввода данных **шкала температуры** ВП **Термометр** соединить с логической константой, размещенной в палитре **Functions»Programming»Boolean.** Установите значение константы TRUE**.** Измените значение диапазона оси Y с **70** на **20** и с **32** на **90**.

5. Сохраните ВП под именем *Мониторинг температуры.vi*.

Запуск ВП

6. Перейдите на лицевую панель, нажав клавиши «**Ctrl+E**» или выбрав пункт главного меню **Window»Show Panel**.
7. С помощью инструмента УПРАВЛЕНИЕ переведите вертикальный переключатель в положение **ВКЛ**.
8. Запустите ВП.

   Блок-диаграмма внутри границы цикла **While** выполняется до тех пор, пока значение терминала условия выхода равно TRUE**.** Например, пока элемент управления **Питание** находится в положение **ВКЛ** (TRUE), ВП **Термометр** осуществляет сбор и отображение текущих значений температуры в виде диаграммы.
9. Остановите выполнение цикла переводом переключателя **Питание** в положение **ВЫКЛ**. Условие выхода примет значение FALSE и выполнение цикла прекратится.

Лицевая панель

10. Настройте формат и масштаб шкал X и Y:
    a. Щелкните правой кнопкой мыши по графику и выберите пункт **Properties**. Появится диалоговое окно **Chart Properties** (свойства графика). Перейдите на закладку **Format and Precision** и установите значение параметра **Digits of precision** (порядок точности) равное 1.

    b. Нажмите на закладку **Plots** и просмотрите различные стили оси Y.

    c. Перейдите на закладку **Scales** и выберите пункт **Время (сек) (X-axis)** из верхнего меню**.** Установите параметры, как показано в диалоговом окне ниже. Установите значение параметра **Multiplier** равное **0,5** , так как теперь из-за функции ожидания **Wait** итерации считывания значения температуры происходят через 0,5 секунды.

    d. На закладке **Scales** выберите пункт **Град. F (Y-axis)** из выпадающего меню**.** Установите параметры, как показано в диалоговом окне ниже.

    e. Установив нужные опции, нажмите кнопку **OK**.
11. Щелкните правой кнопкой мыши по графику и выберите из контекстного меню пункт **Data Operations»Clear Chart** для очистки буфера и окна отображения графика Диаграмм.

Совет Если ВП выполняется, можно выбрать пункт контекстного меню **Clear Chart**, вызываемого щелчком правой кнопки мыши.

12. В данный момент вертикальному переключателю сопоставлено такое механическое действие, что его необходимо возвращать в начальное положение каждый раз перед запуском ВП. Существует возможность изменить механическое действие переключателя так, чтобы он сам возвращался в исходное положение.

    a. Остановите ВП, если он запущен.

    b. С помощью инструмента УПРАВЛЕНИЕ переведите вертикальный переключатель в положение **ВКЛ**.

    c. Щелкните правой кнопкой мыши по переключателю и выберите из контекстного меню пункт **Data Operations»Make Current Value Default**. Это делает положение **ВКЛ** положением переключателя по умолчанию.

    d. Щелкните правой кнопкой мыши по переключателю и выберите из контекстного меню пункт **Mechanical Actions»Latch When Pressed.** Эта установка изменяет значение логического элемента после нажатия левой клавиши мышки и сохраняет его до первого обращения к нему ВП. После обращения ВП значение логического элемента возвращается в исходное положение. Это действие похоже на разрыв цепи и применяется для остановки цикла **While** или для однократного выполнения какой-либо операции.

Запуск ВП

13. Запустите ВП.

14. Инструментом УПРАВЛЕНИЕ нажмите на вертикальный переключатель для остановки процесса считывания температуры. Переключатель перейдет в положение **ВЫКЛ** и, после того как значение попадет на терминал условия выхода из цикла, переключатель вновь перейдет в положение **ВКЛ**.

15. Сохраните ВП. Этот ВП будет использован в упражнении 7-2.

Конец упражнения 7-1

Упражнение 7-2 ВП Расчет средней температуры

Цель: Научиться использовать сдвиговые регистры для реализации алгоритма «бегущее среднее»

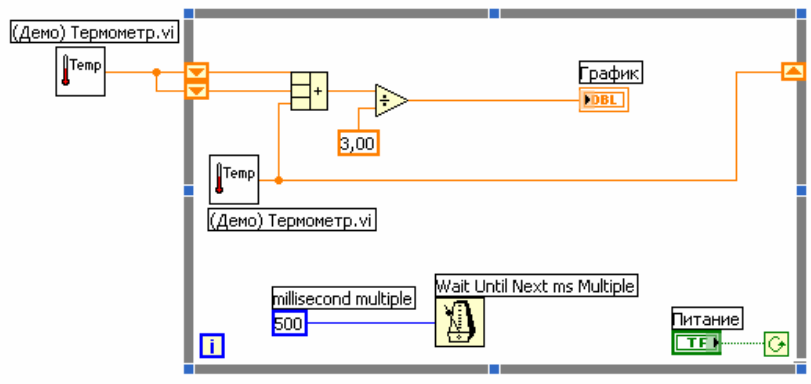
Ниже приведена последовательность действий для изменения **ВП Мониторинг температуры** путем добавления усреднения трех последних измерений температуры и отображения этого среднего значения на диаграмме.

Лицевая панель

1. Откройте **ВП Мониторинг температуры**, созданный в упражнении 7-1.
2. Выберите пункт главного меню **File»Save As** и переименуйте ВП в *Расчет средней температуры.vi*

Блок-диаграмма

3. Перейдите на блок-диаграмму.
4. Щелкните правой кнопкой мыши по границе цикла и выберите пункт **Add Shift Register** в контекстном меню для создания правого и левого терминалов сдвигового регистра.
5. Щелкните правой кнопкой мыши на левом терминале сдвигового регистра и выберите пункт контекстного меню **Add Element** для добавления дополнительных терминалов к сдвиговому регистру.
6. Измените блок-диаграмму, как показано ниже на рисунке.



Нажав и удерживая клавишу <**Ctrl**>, переместите подпрограмму ВП **Термометр** за границу цикла, чтобы создать его копию.

(**Macintosh**) Нажмите клавишу **<0ption>**. (**Sun**) Нажмите клавишу **<Meta>**. (**Linux**) Нажмите клавишу **<Alt>**.

ВП **Термометр** на выходе выдает одно измеренное датчиком значение

температуры и инициализирует левые терминалы сдвигового регистра перед началом выполнения цикла.

Поместите на блок-диаграмму функцию **Compound Arithmetic**, расположенную в палитре **Function»Programming»Numeric**. Эта функция возвращает сумму текущей температуры и двух предыдущих ее значений. С помощью инструмента ПЕРЕМЕЩЕНИЕ измените размеры функции таким образом, чтобы получить три поля ввода данных, показанные слева.

Поместите на блок-диаграмму функцию **Divide**, расположенную в палитре **Function»Programming»Numeric**. Эта функция возвращает среднее значение последних трех измерений температуры.

Щелкните правой кнопкой мыши по полю ввода данных функции **Divide** и выберите пункт **Create»Constant** контекстного меню. Созданной константе присвойте значение **3**.

7. Сохраните ВП.

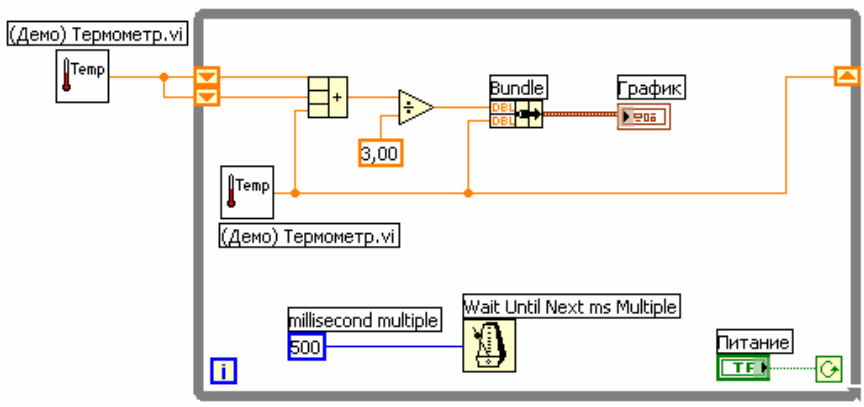
Запуск ВП

8. Запустите ВП.

В каждой итерации происходит измерение температуры **ВП Термометр**. ВП добавляет это значение к двум предыдущим, хранящимся в левых терминалах сдвигового регистра. Для нахождения среднего значения этих измерений их сумма делится на 3. Далее среднее значение выводится на график. Заметьте, что ВП инициализирует сдвиговый регистр значениями температуры.

Блок-диаграмма

9. Измените блок-диаграмму, как показано ниже.



Поместите на блок-диаграмму функцию **Bundle**, расположенную в палитре **Function»Programming»Cluster & Variant**, для отображения полученного среднего значения и значения текущего измерения

температуры на одном и том же графика Диаграмм. Эта функция связывает значения средней и текущей температуры для вывода на график.

10. Сохраните ВП, так как он будет использоваться в дальнейшем.

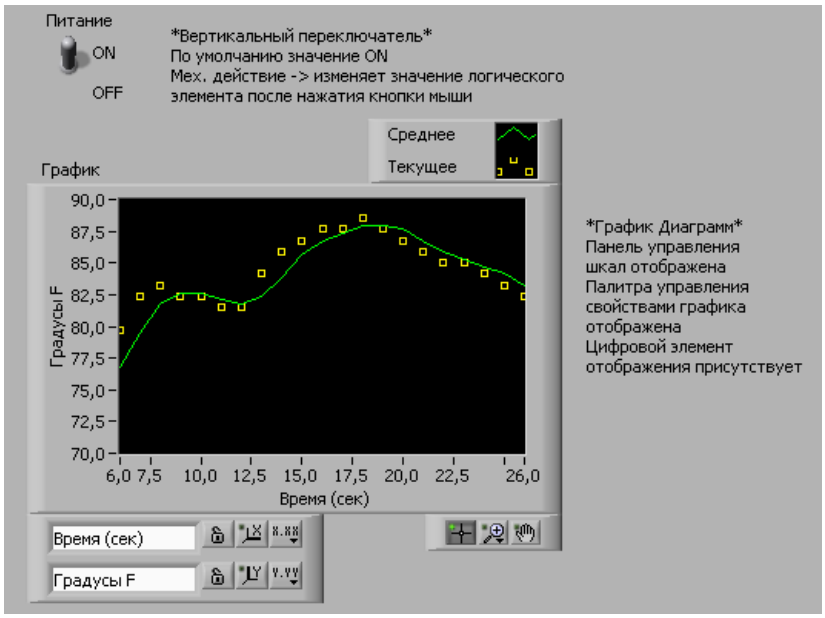
Запуск ВП

11. Запустите ВП. На график Диаграмм выводятся два накладывающихся графика, то есть два графика имеют одну шкалу Y.

12. При желании выполните дополнительную часть упражнения. В противном случае, закройте ВП.

Дополнительная часть

Далее приведена последовательность действий для настройки графика Диаграмм, показанного ниже. На лицевую панель можно вывести панель управления графиком Диаграмм, панель управления масштабом шкалы, палитру элементов управления диаграммой, цифровой элемент отображения, полосу прокрутки и содержимое буфера. По умолчанию, вместе с элементом график Диаграмм отображается панель управления свойствами графика.



13. Настройте ось Y.

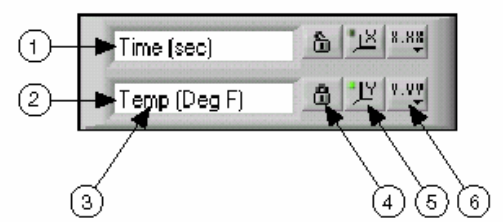
a. С помощью инструмента ВВОД ТЕКСТА измените значение с **70,0** по оси Y на **75,0**

b. С помощью инструмента ВВОД ТЕКСТА измените второе значение на оси Y, напечатав **80,0**. Это число определяет шаг по оси Y.

Например, если следующее значение после **75,0** равно **77,5**, то интервал значений по оси Y составит **2,5** Если изменить значение **77,5** на **80,0** , то интервал значений по оси Y будет равен **5,0** (**75,0** , **80,0** , **85,0** и так далее).

Примечание Размер графика Диаграмм напрямую связан с масштабом осей. При проблемах с настройкой осей попробуйте увеличить размер графика.

14. Щелкните правой кнопкой мыши по полю графика Диаграмм и выберите пункт контекстного меню **Visible Items»Scale Legend**, чтобы отобразить панель **Scale Legend**, как показано на следующем рисунке. Панель управления масштабом шкалы может быть перемещена в любое место лицевой панели.

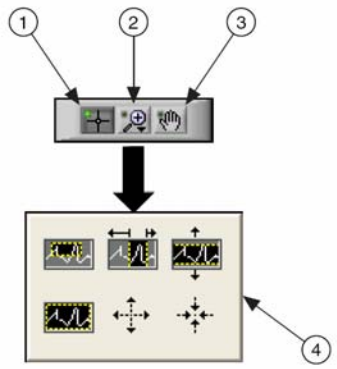


1. Ось X	4. Кнопка блокировки (Scale Lock)
2. Оси Y	5. Кнопка автомасштабирования (Autoscale)
3. Метки курсоров	6. Кнопка формат оси (Scale Format)

15. Настройте каждую ось, используя панель управления свойствами диаграммы **plot legend**.
    a. Убедитесь, что кнопка **Autoscale** окрашена в зеленый цвет, а кнопка **Scale Lock** выключена. Таким образом, ось Y автоматически изменяет масштаб вывода на экран текущих значений.
    b. Нажмите кнопку **Scale Format** для изменения формата, точности, режима отображения, масштаба и вида размерной сетки для каждой оси.
16. С помощью панели **plot legend** настройте вид диаграмм, отображаемых на элементе график Диаграмм**.**
    a. С помощью инструмента ПЕРЕМЕЩЕНИЕ измените размер **plot legend** для отображения двух графиков.
    b. С помощью инструмента ВВОД ТЕКСТА измените метку **Температура** на **Среднее значение** и **Plot 1** на **Текущая температура**. Если текст получается слишком длинным, то следует увеличить размер **plot legend**, используя инструмент ПЕРЕМЕЩЕНИЕ.

c. Щелкните правой кнопкой мыши по **plot legend** и вызовите контекстное меню с разделами редактирования цвета, фона, представления линий и точек.

17. Щелкните правой кнопкой мыши по элементу график Диаграмм и выберите пункт контекстного меню **Visible Items»Graph Palette** для отображения палитры элементов управления графиком **Graph Palette**, как показано ниже на рисунке. Палитра **Graph Palette** может быть перенесена в любую часть лицевой панели.



1 Инструмент перемещения курсора (**Cursor Movement)**.	3 Инструмент панорамирования (**Panning tool**).
2 Кнопка масштабирования (**Zoom).**	4 Панель управления масштабом шкалы (**Zoom pull-down menu**).

Для изменения масштаба изображения следует использовать кнопку **Zoom.** Для быстрого перемещения по графику следует использовать инструмент **Panning**. Инструмент **Cursor Movement** позволяет перемещать курсор в поле графика.

18. Сохраните и запустите ВП. Во время работы ВП измените настройки графика Диаграммы с помощью панели управления масштабом шкалы **Scale Legend** и палитры элементов управления графиком **Graph Palette**.

Примечание При редактировании меток осей графика Диаграмм следует обратить внимание на то, что размер элемента должен быть больше, чем максимальный размер окна отображения диаграмм, заданный по умолчанию.

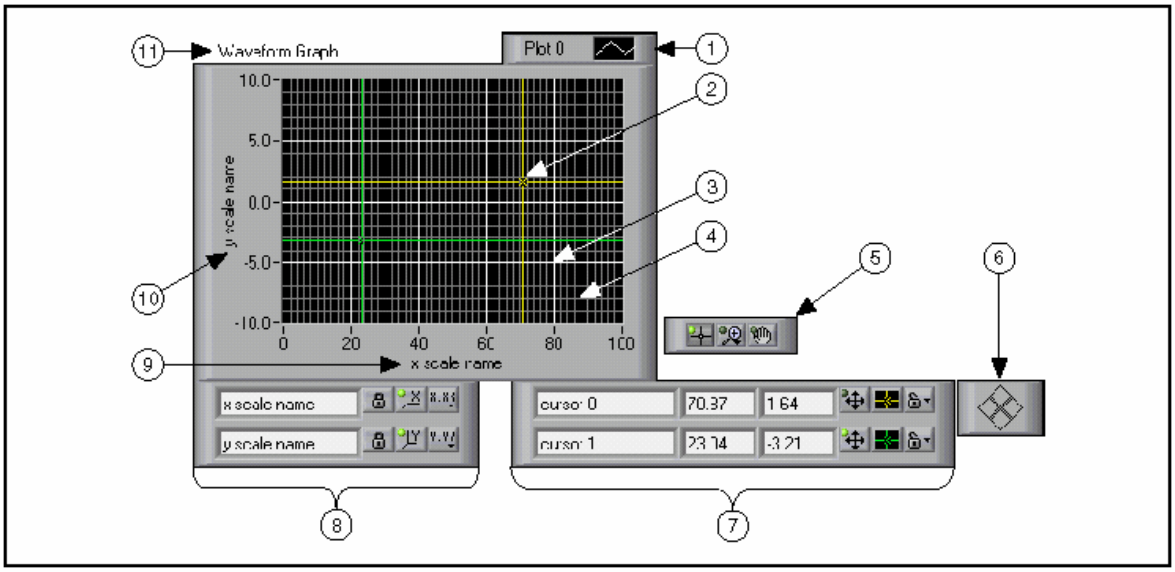
19. С помощью инструмента УПРАВЛЕНИЕ переведите вертикальный выключатель **Питание** в положение **ВЫКЛ** для завершения работы ВП.

20. Сохраните изменения и закройте ВП.

Конец упражнения 7-2

B. График Осциллограмм и двухкоординатный график Осциллограмм

С помощью графиков в виде осциллограмм ВП обычно отображает накопленные в массив данные. На следующей иллюстрации показаны элементы графика.



1. Панель управления свойствами осциллограмм (Plot legend)	5. Палитра элементов управления графиком (Graph palette)	9. Шкала X (X-scale)
2. Курсор (Cursor)	6. Панель перемещения курсора (Cursor mover)	10. Шкала Y (Y-scale)
3. Основная размерная сетка (Grid mark)	7. Панель управления свойствами курсора (Cursor legend)	11. Собственная метка графика (Label)
4. Дополнительная размерная сетка (Mini-grid mark)	8. Панель управления шкалой (Scale legend)	

График Осциллограмм (**Waveform Graph**) и двухкоординатный график Осциллограмм (**X-Y Graph**) расположены на палитре **Controls»Modern»Graph**. График Осциллограмм отображает только однозначные функции, такие как y=f(x), с точками, равномерно распределенными по оси X. Двухкоординатный график Осциллограмм отображает любой набор точек, будь то равномерно распределенная выборка во времени или нет.

Для отображения множества осциллограмм необходимо изменить размер панели **Plot legend.** График множества Осциллограмм используется с целью экономии пространства на лицевой панели и для сравнения осциллограмм

данных между собой**.** График Осциллограмм и двухкоординатный график Осциллограмм автоматически поддерживают режим отображения множества осциллограмм.

Одиночный график Осциллограмм

Одиночный график Осциллограмм работает с одномерными массивами и представляет данные массива в виде точек на графике, с приращением по оси X равным **1** и началом в точке x=0. Графики также отображают кластеры, с установленным начальным значением x, Δx и массивом данных по шкале y. В качестве примера можно рассмотреть ВП **Waveform Graph VI** в разделе библиотеки примеров (*examples\general\ graphs\gengraph.llb*).

График множества Осциллограмм

График множества Осциллограмм работает с двумерными массивами данных, где каждая строка массива есть одиночная осциллограмма данных и представляет данные массива в виде точек на графике, с приращением по оси X равным **1** и началом в точке x=0.

Для представления каждого столбца двумерного массива данных в виде осциллограммы на графике необходимо соединить терминал данных массива с входным терминалом данных графика, затем щелкнуть правой кнопкой мыши по полю графика и выбрать пункт контекстного меню **Transpose Array** (транспонирование массива).

В качестве примера можно рассмотреть график **(Y) Multi Plot 1**, открыв ВП **Waveform Graph VI** из библиотеки примеров (*examples\general\ graphs\gengraph llb*).

Графики множества Осциллограмм отображают кластеры, состоящие из начального значения x, Δx и двумерного массива данных по шкале y. График представляет данные по шкале y в виде точек с приращением Δx по оси x и началом в точке x=0. В качестве примера можно рассмотреть график **(Xo, dX, Y) Multi Plot 3**, открыв ВП **Waveform Graph VI** из библиотеки примеров (*examples\general\graphs\gengraph llb*).

Графики множества Осциллограмм отображают также и кластеры с установленным начальным значением x, Δx и массивом данных, содержащим кластеры. Каждый кластер содержит массив точек, отображающих данные по шкале Y. Для создания массива кластеров следует использовать функцию **Bundle**, которая объединяет массивы в кластеры. Далее, с помощью функции **Build Array** создается массив кластеров. Можно также использовать функцию **Build Cluster Array**, которая создает массив кластеров с определенными полями ввода данных. В качестве примера можно рассмотреть график **(Xo, dX, Y) Multi Plot 2,** открыв ВП **Waveform Graph VI** из библиотеки примеров (*examples\general\ graphs\gengraph llb*).

Одиночные двухкоординатные графики Осциллограмм

Одиночный двухкоординатный график Осциллограмм работает с кластерами, содержащими массивы *x* и *y*. Двухкоординатный график Осциллограмм также воспринимает массивы точек, где каждая точка является кластером, содержащим значения по шкалам *x* и *y*. В качестве примера можно рассмотреть ВП **XY Graph VI** из библиотеки примеров (*examples\general\ graphs\gengraph llb*).

Двухкоординатные графики множества Осциллограмм

Двухкоординатные графики множества Осциллограмм работают с массивами осциллограмм, в которых осциллограмма данных является кластером, содержащим массивы значений *x* и *y*. Двухкоординатные графики множества Осциллограмм воспринимают также массивы множества осциллограмм, где каждая осциллограмма представляет собой массив точек. Каждая точка – это группа данных, содержащая значения по *x* и *y*. В качестве примера можно рассмотреть ВП **XY Graph VI** из библиотеки примеров (*examples\general\ graphs\gengraph llb*).

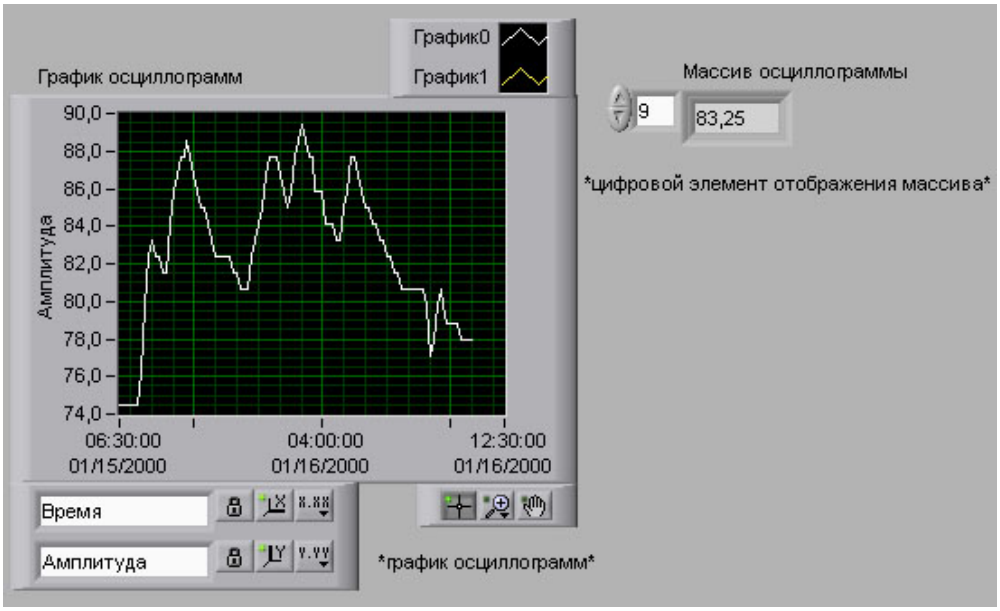
Упражнение 7-3 ВП Вывод массива данных на график

Цель: Создать массив, используя свойство автоиндексации цикла For, и вывести данные массива на график Осциллограмм

Ниже приведена последовательность действий по созданию ВП, который отображает массив данных на графике Осциллограмм, и изменению ВП для отображения множества осциллограмм.

Лицевая панель

1. Откройте новый ВП и оформите лицевую панель, как показано ниже.



   a. Выберите шаблон массива, расположенный в палитре **Controls»Modern»Array, Matrix & Cluster**.

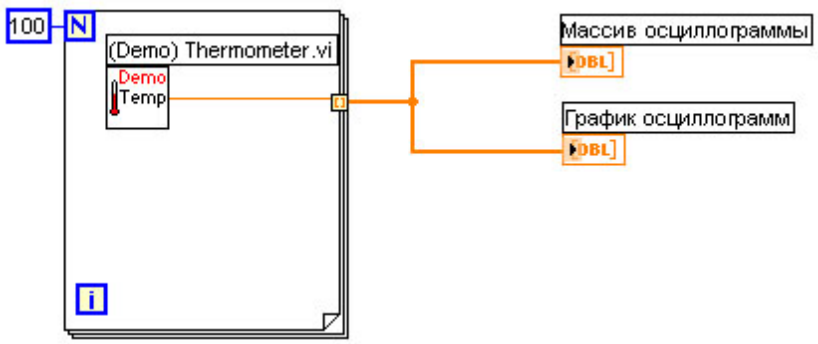
   b. Собственной метке массива присвойте имя **Массив осциллограмм**.

   c. Выберите цифровой элемент отображения данных, расположенный в палитре **Controls»Modern»Numeric**, и поместите его в шаблон массива.

   d. Поместите на лицевую панель график Осциллограмм, расположенный в палитре **Controls»Modern»Graph.**

Блок-диаграмма

2. Создайте блок-диаграмму, как показано ниже.

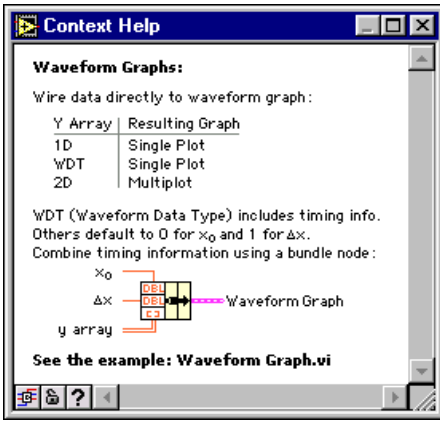


В палитре **Functions»Select a VI** выберите ВП *Термометр.vi* (*c:\exercises\LV Basics I*), который был создан в упражнении 3-2. Поместите выбранный ВП на блок-диаграмму. В этом ВП каждая итерация цикла **For** выдает одно значение температуры.

В палитре **Functions»Programming»Structures** выберите цикл **For**. На каждой итерации цикла создается значение температуры и передается на терминал выхода из цикла. Терминалу максимального количества итерации цикла присвойте значение **100**.

Совет При передаче данных на диаграмму или график для определения порядка полей ввода данных и т.д. следует использовать окно контекстной справки **Context Help**, чтобы понять, какую функцию следует использовать – **Build Array** или **Bundle**. Для включения окна контекстной справки следует нажать <**Ctrl+H**>. В общем случае, графики Диаграмм следует использовать для вывода на экран скалярных точек, а графики Осциллограмм – массивов данных по *y*. Двухкоординатные графики Осциллограмм следует использовать для отображения массивов значений *x* и y.

Например, если на блок-диаграмме навести курсор на терминал данных графика Осциллограмм, в окне **Context Help** появится информация, показанная ниже. Для просмотра примера использования графика Осциллограмм ВП **Waveform Graph** следует выбрать в главном меню **Help»Find Examples** и далее **Fundamentals»Graphs and Charts**. Более подробную информацию о графическом типе данных можно получить в Уроке 10, *Сбор и отображение данных.*



3. Сохраните ВП под именем *Вывод массива данных на график.vi*

Запуск ВП

4. Перейдите на лицевую панель и запустите ВП. ВП построит осциллограмму данных массива на графике Осциллограмм**.**

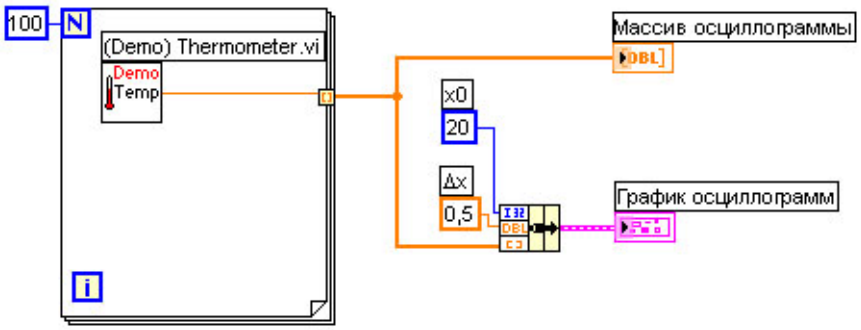
5. Введите любое число в элемент **Массив осциллограммы** для просмотра соответствующего ему элемента массива. Введите число, превосходящее размер формируемого массива данных (100), значения этих элементов неопределенны.

6. С помощью инструмента ПЕРЕМЕЩЕНИЕ измените размер **Массива осциллограмм** для просмотра большего количества элементов. Элемент отображения данных располагает элементы по возрастанию значения индекса, начиная с введенного индекса.

Блок-диаграмма

На блок-диаграмме используются начальные значения *x* и Δ*x* для графика Осциллограмм, заданные по умолчанию. В случае, когда необходимо установить значения *x* и Δ*x,* отличные от значений заданных по умолчанию, следует использовать функцию **Bundle**.

7. Измените блок-диаграмму, как показано ниже.



В палитре **Functions»Programming»Cluster & Variant** выберите функцию **Bundle**. Эта функция объединяет элементы в одномерный кластер. Элементы состоят из начального значения *x* =20, Δ*x*=0,5 и массива данных y.

Создайте две числовые константы, выбрав их из палитры **Functions»Programming»Numeric**, для установки начального значения *x* и Δx.

Собственную метку константы Δ*x* назовите **Dx**. С помощью инструмента ВВОД ТЕКСТА выделите символ **D** и в выпадающем меню панели инструментов **Text Settings** выберите шрифт **Symbol**. Символ **D** превратится в Δ.

8. Сохраните ВП.

Запуск ВП

9. Перейдите на лицевую панель и запустите ВП.

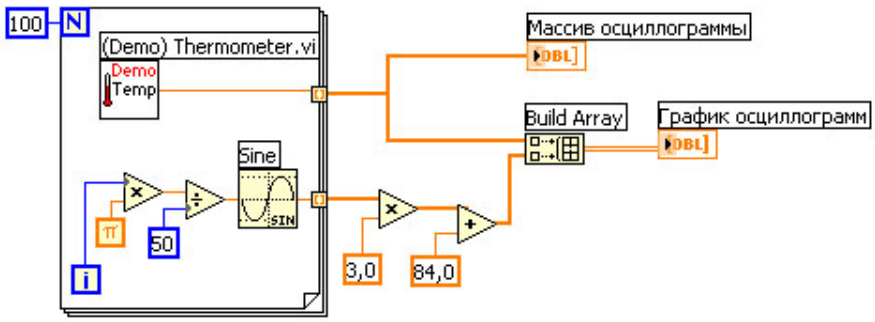
   График отобразит те же 100 точек с началом в точке 20 и Δ*x*=0,5 для каждой точки по оси *x*. Это соответствует проведению измерений с 20-ти секундной отметки в течение 50 секунд.

Примечание Изменять начальное значение *x* и Δ*x* следует только в одном месте - или в функции **Bundle**, или в диалоговом окне свойств графика**.**

10. При желании, выполните дополнительную часть упражнения. В противном случае закройте ВП.

Дополнительная часть

11. Щелкните правой кнопкой мыши в поле графика Осциллограмм и из контекстного меню выберите пункт **Visible Items»Graph Palette**. Нажмите кнопку **Zoom** для более детального рассмотрения данных на графике.
12. Щелкните правой кнопкой мыши по полю графика и из контекстного меню выберите пункт **Visible Items»Scale Legend**.
13. Перейдите на блок-диаграмму. Выполните следующие пункты для создания графика множества Осциллограмм. Для этого нужно создать двумерный массив элементов типа данных, которые обычно подаются на одиночный график Осциллограмм. Измените блок-диаграмму как показано ниже.



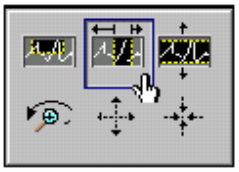
Выбнрите функцию **Sine,** расположенную в палитре **Functions» Mathematics»Elementary & Special Function**. Эта функция будет использоваться для формирования массива данных, описывающих один цикл волны синуса.

Выберите функцию **Build Array**, расположенную в палитре **Functions»Programming»Array**. Эта функция создает структуру данных для отображения двух массивов на графике Осциллограмм.

Выберите константу π, расположенную в палитре **Functions»Programming»Numeric»Math Constants**.

14. Сохраните ВП.
15. Перейдите на лицевую панель и запустите ВП. Два графика отображаются на одном графике Осциллограмм**.**
16. Перейдите на блок-диаграмму.
17. Щелкните правой кнопкой мыши по проводнику данных к **Массиву осциллограмм**, из контекстного меню выберите пункт **Custom Probe»Graph»Waveform Graph** для создания графического отладочного индикатора.
18. Перейдите на лицевую панель и запустите ВП. Отладочный индикатор показывает только массив данных. График синуса отсутствует потому, что отладочный индикатор не был помещен на проводник данных, который связан с синусом.
19. Закройте окно отладочного индикатора.
20. Измените масштаб части графика.

a. Нажмите на кнопку **Zoom** в палитре **graph palette**, показанную слева, для вывода на экран выпадающего меню, показанного ниже.



b. Выберите пункт **Zoom by X Rectangle**, как показано выше.

c. Удерживая кнопку мыши в нажатом состоянии, обведите курсором часть осциллограммы. После отпускания кнопки мыши выделенная область изменит масштаб.

d. Можно также выбрать **Zoom by Y Rectangle** или **Zoom by Selected Area**.

e. Для отмены изменения масштаба следует выбрать **Undo Zoom** из нижнего левого угла выпадающего меню или нажать на кнопку **x-axis** на панели **scale legend**, показанной слева.

21. Используя инструмент ПАНОРАМИРОВАНИЕ, показанный слева, переместите экран графика. Для возвращения экрана графика в его первоначальное положение следует нажать на кнопки **x-axis** и **y-axis** панели редактирования шкалы.

22. С помощью инструмента ПЕРЕМЕЩЕНИЕ КУРСОРА, показанного слева, верните курсор в стандартный режим.

23. Сохраните изменения и закройте ВП.

Конец упражнения 7-3

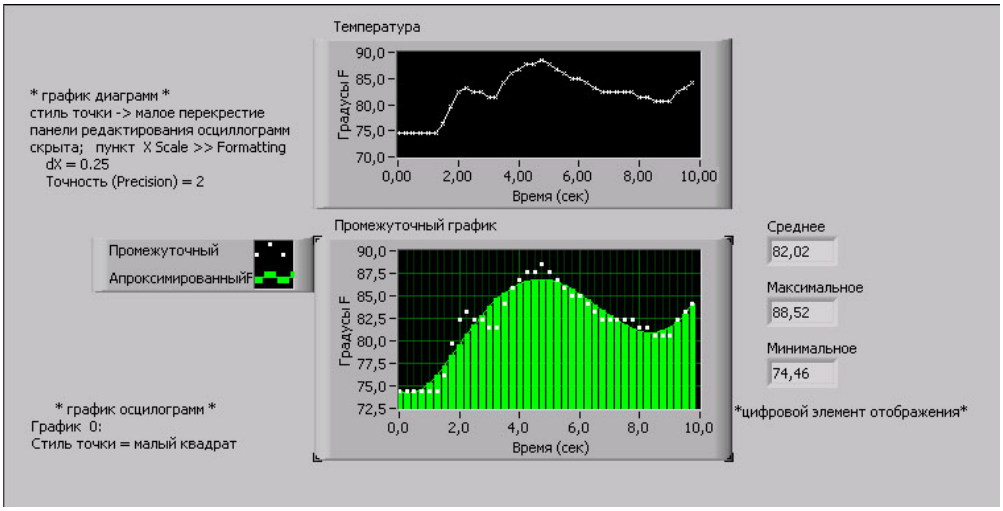
Упражнение 7-4 ВП Температурный анализ

Цель: Отобразить на графике и проанализировать данные

Ниже приведена последовательность действий для создания ВП, который измеряет температуру каждые 0,25 секунды в течение 10 секунд. В процессе измерения ВП в реальном масштабе времени отображает данные на графике Диаграмм**.** После завершения измерений, ВП выводит данные на график Осциллограмм и рассчитывает минимальную, максимальную и среднюю температуру. Кроме того, ВП отображает аппроксимацию осциллограммы температуры.

Лицевая панель

1. Откройте новый ВП. Используя советы, оформите лицевую панель, как показано ниже на рисунке.

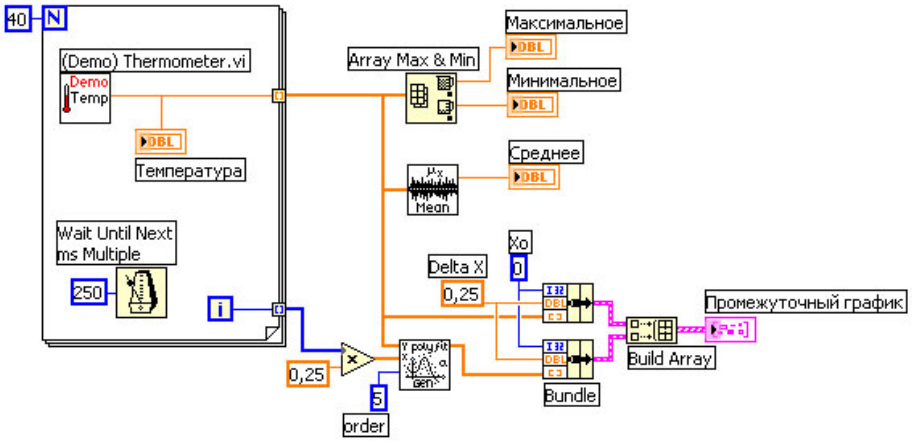


- Установите стиль точек на графике диаграмм в виде **x**.
- Скройте панель управления свойствами диаграммы.
- Щелкните правой кнопкой мыши по графику Диаграмм и из контекстного меню выберите пункт **X Scale>>Formatting**, установите значение **dX** равное **0,25** , а **Digits of Precision** (количество точек после запятой) равное **2**.
- С помощью инструмента ПЕРЕМЕЩЕНИЕ измените размер панели **plot legend**.
- С помощью инструмента ВВОД ТЕКСТА переименуйте **График 0** в **Промежуточный**, а **График 1** - в **Аппроксимированный**.
- Установите стиль точек осциллограммы **Промежуточный** в виде маленького квадрата.

- Пока не создавайте элементы отображения данных **Среднее**, **Макс.** и **Мин.**

Блок-диаграмма

2. Постройте блок-диаграмму, как показано ниже.



Выберите в палитре **Functions»Select a VI** и перейдите в директорию *c:\exercises\LV Basics I*. Выберите *Термометр.vi*, созданный в упражнении 3-2, и поместите его на блок-диаграмму. Этот ВП выдает одно измеренное значение температуры.

В палитре **Functions»Programming»Timing** выберите функцию **Wait Until Next ms Multiple**. С помощью числовой константы на поле ввода функции подайте значение 250, что заставит цикл **For** выполняться каждые 0,25.

В палитре **Functions»Programming»Array** выберите функцию **Array Max & Min**. Эта функция определяет минимум и максимум температуры.

В палитре **Functions»Mathematics»Probability and Statistics** выберите ВП **Mean VI**. Этот ВП определяет среднее значение измеренной температуры.

Щелкните правой кнопкой мыши по полю вывода данных **Array Max & Min** и ВП **Mean VI** и выберите в контекстном меню пункт **Create»Indicator** для создания элементов **Макс.**, **Мин.** и **Среднее**.

В палитре **Functions»Mathematics»Fitting** выберите ВП **General Polynomial Fit**. Этот ВП проведет аппроксимацию осциллограммы температуры.

В палитре **Functions»Programming»Cluster & Variant** выберите функцию **Bundle**. Нажмите и удерживайте клавишу <**Ctrl**> во время перемещения функции для создания ее копии. Эта функция

объединяет элементы в одномерный кластер. Элементы содержат начальное значение x=0, Δx=0,25 и массив значений температуры по y. Значение Δx=0,25 необходимо для того, чтобы ВП выводил значения температуры на график Осциллограмм каждые 0,25 секунды.

В палитре **Functions»Programming»Array** выберите функцию **Build Array**. Эта функция создает массив кластеров из группы измеренных данных температуры и их аппроксимации.

3. Сохраните ВП под именем *Анализ температуры.vi*

Запуск ВП

4. Перейдите на лицевую панель и запустите ВП.

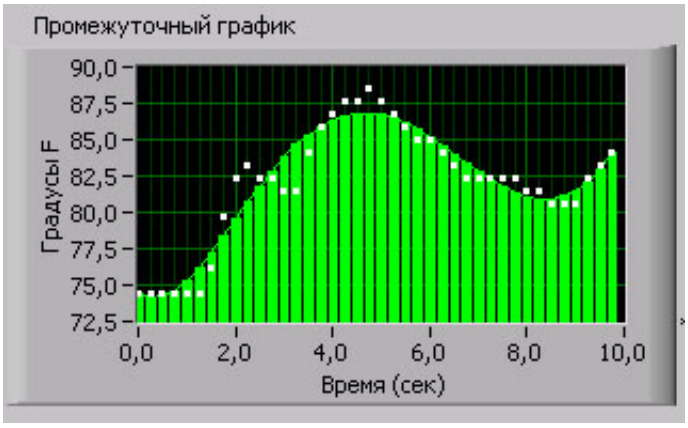
   На графике Осциллограмм одновременно появятся осциллограммы данных температуры и их аппроксимации.

5. Поменяйте значения константы порядка аппроксимации на блок-диаграмме и снова запустите ВП.

6. Измените вид представления осциллограмм.

   a. Щелкните правой кнопкой мыши по надписи **Промежуточный** на панели **Plot legend** и выберите в контекстном меню **Common Plots»Scatter Plot** (экспериментальные точки).

   b. Щелкните правой кнопкой мыши по надписи **Аппроксимированный** на панели **Plot legend** и в разделе **Bar Plots** контекстного меню выберите вторую иконку в средней строке. Получившийся график осциллограмм должен оказаться подобным изображенному ниже.



7. Сохраните и закройте ВП.

Конец упражнения 7-4

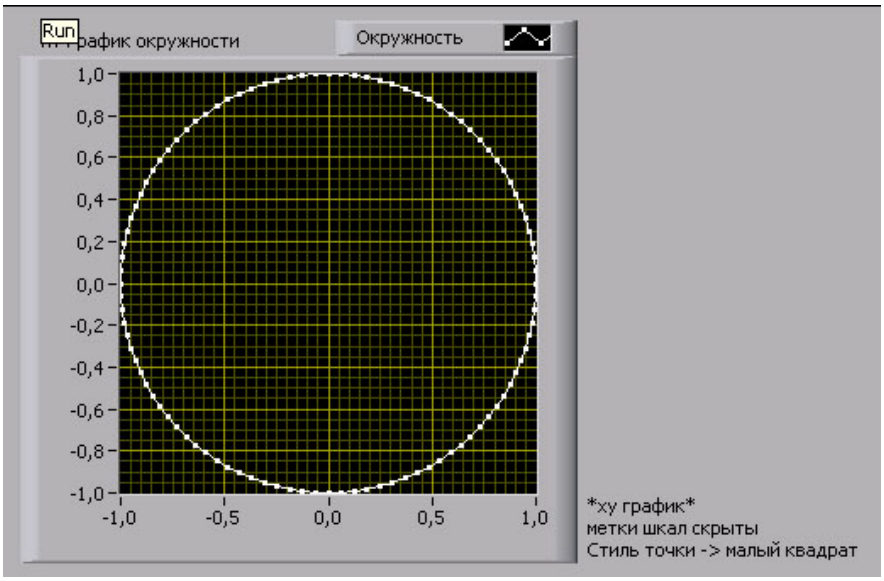
Упражнение 7-5 ВП График окружности

Цель: Построить осциллограмму окружности, используя двухкоординатный график Осциллограмм.

Выполните следующие шаги для создания ВП, который с помощью двух независимых массивов X и Y построит осциллограмму в форме окружности.

Лицевая панель

1. Откройте новый ВП и создайте лицевую панель, показанную ниже на рисунке.



   a. В палитре **Controls»Graph** выберите **XY Graph**.

   b. Присвойте графику имя **XY график окружности**.

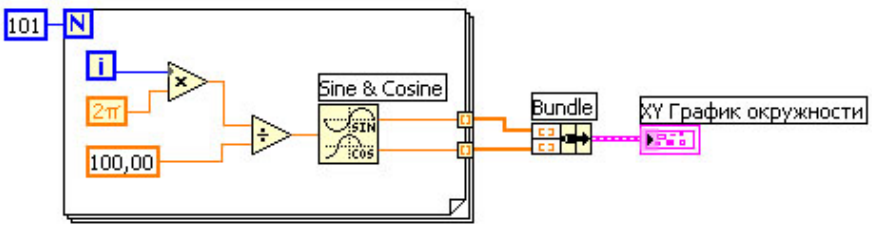
   c. Переименуйте надпись **График 0** на панели **Plot legend** в **Окружность**.

   d. Щелкните правой кнопкой мыши на панели **Plot legend** и выберите из контекстного меню в разделе **Point Style** точки в форме маленьких квадратов.

   e. Переименуйте и измените диапазон шкал, как показано на рисунке.

Блок-диаграмма

2. Постройте блок-диаграмму, как показано ниже.



В палитре **Functions»Mathematics»Elementary»Trigonometric** выберите функцию **Sine & Cosine.** Эта функция создает массив данных, содержащий один период функций синуса и косинуса.

В палитре **Functions»Programming»Cluster & Variant** выберите функцию **Bundle**. Эта функция объединяет массивы синуса и косинуса в кластер.

В палитре **Functions»Programming»Numeric»Math Constants** выберите константу 2π.

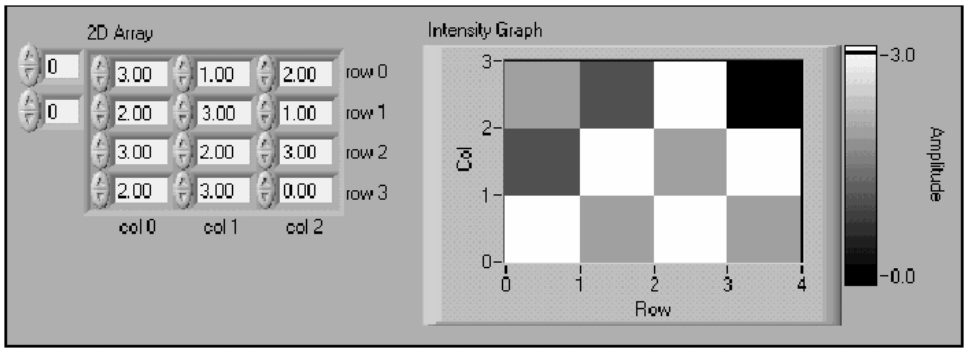
3. Сохраните ВП под именем *График окружности.vi*
4. Перейдите на лицевую панель и запустите ВП.
5. Закройте ВП.

Конец упражнения 7-5

C. Графики интенсивности (дополнительно)

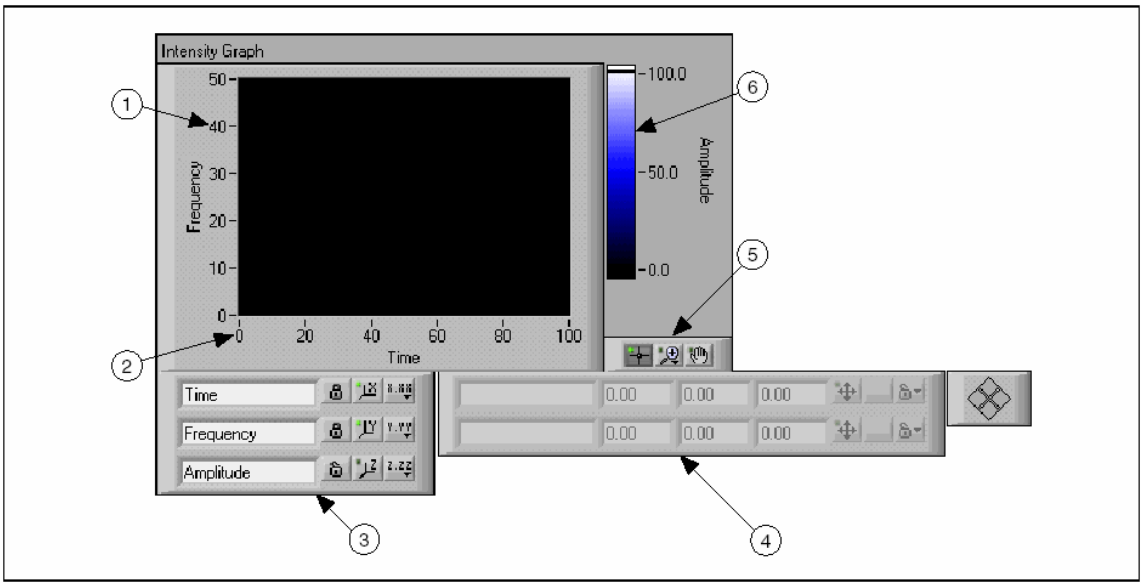
Графики и таблицы интенсивности (**Intensity graphs and charts**) удобны для отображения двумерных данных. Например, для представления топографии местности, где амплитудой является высота над уровнем моря. Как и в случае с графиками Диаграмм и Осциллограмм, график интенсивности имеет постоянный размер дисплея, а дисплей таблицы интенсивности обладает возможностью прокрутки. Графики и таблицы интенсивности принимают на вход двумерный массив данных, где каждое число соответствует определенному цвету. Положение данного цвета на графике определяется индексами элемента в массиве. Графики и таблицы интенсивности имеют возможность отображать до 256 различных цветов.

На следующей иллюстрации изображен массив размера 4x3, визуализированный на графике интенсивности. График отображает транспонированный массив.



Настройки графиков и таблиц интенсивности

Графики и таблицы интенсивности имеют много общих свойств с графиками Диаграмм и Осциллограмм, которые можно отобразить или спрятать, выбрав пункт контекстного меню **Visible Items**. Так как в графиках и таблицах интенсивности появляется третье измерение, то необходим дополнительный элемент – элемент управления цветовой шкалой, который определяет диапазон и способ цветового отображения данных. Ниже показаны составляющие части графика интенсивности.



1. Шкала Y (Y scale)	4. Панель управления курсорами (Scale legend)
2. Шкала X (X scale)	5. Палитра инструментов для работы с графиком (Graph Palette)
3. Панель управления шкалами (Scale legend)	6. Шкала Z (цветовая шкала) (Z scale (color ramp))

Для того чтобы поменять цвет, ассоциированный с маркером, нужно выбрать пункт **Marker Color** в контекстном меню и выбрать цвет в окне выбора цвета. Контекстное меню вызывается инструментами УПРАВЛЕНИЕ или ПЕРЕМЕЩЕНИЕ нажатием правой кнопки мыши по маркеру, расположенному около цветовой шкалы. Для добавления маркера к цветовой шкале необходимо нажать правой кнопкой мыши на цветовую палитру и выбрать пункт **Add Marker** из контекстного меню. Чтобы изменить значение какого-либо маркера на цветовой шкале нужно переместить маркер к требуемому значению инструментом УПРАВЛЕНИЕ или использовать инструмент ВВОД ТЕКСТА для ввода нового значения в текстовое поле маркера.

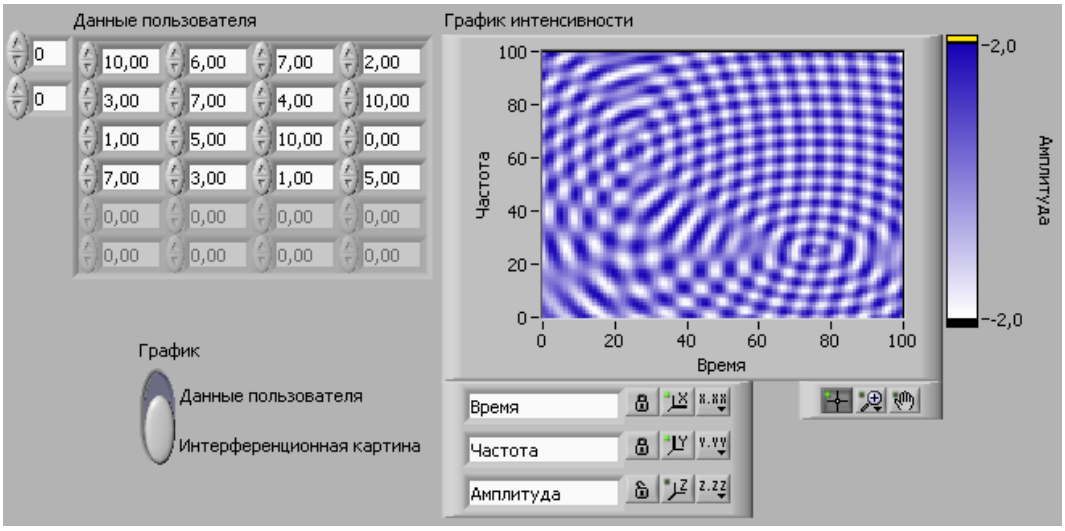
Упражнение 7-6 ВП Пример графика интенсивности (дополнительное)

Цель: Использовать график интенсивности.

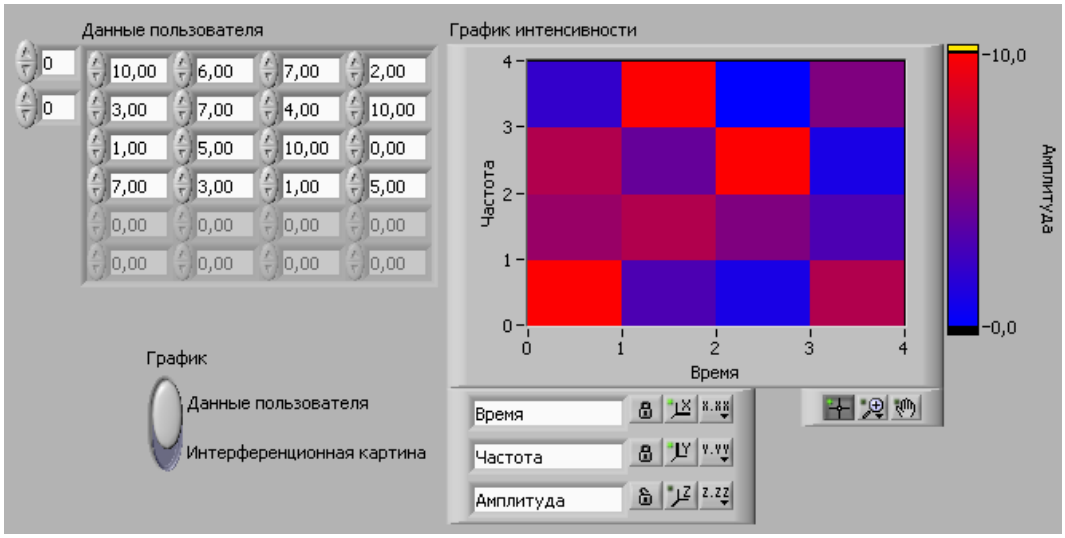
В этом упражнении используется ВП, отображающий интерференционную картину на графике интенсивности. Этот же ВП используется для отображения двумерного массива данных на графике.

Лицевая панель

1. Откройте и запустите ВП *Пример графика интенсивности.vi* По умолчанию, ВП выводит на график картину интерференции волн. Узел атрибутов на блок-диаграмме задает диапазон применяемых цветов, который может быть изменен редактированием массива констант **Color Array**.



2. Переведите переключатель **График** в положение **Данные пользователя** и введите значения от 0 до 10 в массив. Запустите ВП. Обратить внимание, как значение каждого элемента преобразуются в цвет на графике интенсивности.



3. Закройте ВП, не сохраняя изменений.

Конец упражнения 7-6

D. Создание трехмерных сцен (дополнительно)

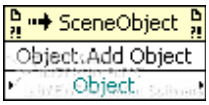
Создание трехмерной сцены может использоваться с целью визуализации какого-либо физического процесса. Вы можете создавать трехмерные объекты, задавать их форму, размер, текстуру, способ движения, способ появления и способ взаимоотношения с другими объектами внутри сцены.
Для работы с 3D объектами используйте виртуальные приборы, находящиеся на палитре **Functions»Programming»Graphics & Sound»3D Picture Controls.** Данная палитра содержит следующие вложенные палитры:

• **Object VIs:**

Create Object VI – создает ссылку на 3D объект.

Find Object VI – находит ссылку на 3D объект по ссылке на 3D сцену и имени 3D объекта.

Примечание. Вы также можете использовать свойства SceneObject properties и методы SceneObject methods для того, чтобы создать ссылку на 3D объект.



• **File Loading VIs** () **-** Используйте виртуальные приборы, находящиеся в данной закладке для добавления существующих файлов, содержащих модели или сцены на 3D сцену.

• **Geometries VIs** – на данной палитре находятся ВП с помощью которых можно задать геометрическую форму для 3D объекта:

Create Cone VI – создает конус с заданными параметрами.

Create Cylinder VI – создает цилиндр с заданными параметрами.

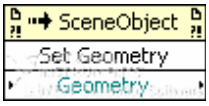
Create Box VI – создает параллелепипед с заданными параметрами.

Create Sphere VI – создает сферу с заданными параметрами.

Create Height Field VI – конвертирует двумерный массив данных в 3-х мерный объект.

Create Mesh VI – создает 3-х мерную координатную сетку.

Примечание. Вы также можете использовать свойства SceneGeometry properties и методы SceneGeometry methods для того, чтобы задать геометрическую форму 3D объекта.



• **Transformations VIs –** позволяют задать положение 3D объекта на сцене, вращать его, перемещать и масштабировать.

• **Helpers VIs** – позволяют производить различные операции со сценой в целом. Используя виртуальные приборы, находящиеся в этой закладке, Вы можете сконфигурировать отдельное окно для сцены (**Setup Window VI**), задать плоскость, разрезающую сцену (**Create Clip Plane VI**), добавить источники света (**New Light VI**), добавить текстуру для 3D объекта (**Texture VI**) и создать кластер RGBA кластер для задания цвета объекта (**Color Change VI**).

Для конфигурирования 3D сцены Вы также можете использовать следующие свойства и методы:
- SceneWindow properties – для выполнения сцены в отдельном окне, конфигурирования окна и установки способа взаимодействия камеры со сценой.
- SceneClipPlane properties – для установки плоскостей разреза на сцене.
- SceneLight properties – для установки на сцену источника света.
- SceneTexture properties и SceneTexture methods – для задания текстуры для 3D объекта.

Для большей информации о 3D picture control Вы можете обратиться к *LabVIEW Help*. Найдите по содержанию пункт **Fundamentals»Graphics and Sound VIs**. Кроме того, можно посмотреть пример виртуального прибора, моделирующего движение солнца, луны и земли. Он находится в папке *labview\examples\picture\3D Picture Control*.

Краткое изложение пройденного материала, советы и секреты

- График Диаграмм (**Waveform Chart**)– это специальный цифровой элемент отображения, предназначенный для отображения одной и более диаграмм.
- График Диаграмм имеет три режима отображения данных:
  - **strip chart** представляет собой экран, прокручиваемый слева направо подобно бумажной ленте.
  - **scope chart**, по достижении правой границы окно диаграммы очищается, и заполнение диаграммы начинается с левой границы.
  - **sweep chart**, в отличие от режима **scope chart**, окно диаграммы не очищается, а новые данные отделяются от старых вертикальной линией – маркером.
- График Осциллограмм (**Waveform Graph**) и двухкоординатный график Осциллограмм (**XY graph**) отображают данные из массивов.
- Для редактирования графика или изменения настроек графика Диаграмм необходимо щелкнуть по нему правой кнопкой мыши.
- На один график можно вывести более одной осциллограммы с помощью функции **Build Array**, расположенной в палитре **Functions»Programming»Array**. На двухкоординатный график Осциллограмм данные вводятся с помощью функции **Bundle**, расположенной в палитре **Functions»Programming»Cluster & Variant**.
- Для визуализации трехмерных данных можно воспользоваться графиками и таблицами интенсивности. Третья координата отображается цветом в соответствии с определенной цветовой схемой. Графики и таблицы интенсивности часто применяются в задачах спектрального анализа, визуализации температуры и обработки изображений.
- При выводе данных на диаграмму или график удобно использовать окно контекстной справки **Context Help.**

Дополнительные упражнения

7-7 Создайте ВП, строящий два графика: график произвольных чисел и бегущее среднее по четырем точкам на графике Диаграмм в режиме **sweep chart.** Воспользуйтесь следующими советами:

⇒ Используйте цикл **For** (N=200) вместо цикла **While**.

⇒ Используйте три левых терминала сдвигового регистра для усреднения последних четырех значений.

⇒ Для генерации данных используйте функцию **Random Number (0-1)**, расположенную в палитре **Functions»Programming»Numeric**.

⇒ Для объединения произвольных чисел с их средним значением для построения на одном графике используйте функцию **Bundle**, расположенную в палитре **Functions»Programming»Cluster & Variant**.

Сохраните ВП под именем *Бегущее среднее.vi*

7-8 Создайте ВП для непрерывного измерения температуры. Данные выводите на график Диаграмм в режиме **scope chart** с задержкой в 1 секунду. Если температура превысит предельное значение, введенное в соответствующий элемент управления, на лицевой панели должен загореться красный светодиод. График Диаграмм должен отображать текущие и предельные значения измерений температуры. Необходимо предусмотреть возможность изменения предельных значений температуры с помощью элементов отображения.

Сохраните ВП под именем *Предел температуры.vi*

7-9 Измените ВП, созданный в упражнении 7-8 так, чтобы он отображал только максимальные и минимальные значение текущих измерений температуры.

Совет Используйте сдвиговый регистр и две функции **Max & Min,** расположенных в палитре **Functions»Programming»Comparison.**

Сохраните ВП под именем файла *Предел температуры (Max-Min).vi*

7-10 Постройте на графике интенсивности функцию $z(x, y) = sin\, c^2\left(x^2 + y^2\right)$. С помощью курсора в виде горизонтальной линии указать сечение графика, которое должно отображаться на отдельном графике.

Сохраните ВП под именем *Профиль интенсивности.vi*

7-11 Создайте ВП, отображающий графики функций ($y = ax^2 + b,\ y = ax^2 + bx + c,\ y = a\,exp(bx)$) с помощью **XY graph** и масштабирующий оси графика в соответствии с заданными

параметрами.

Совет Для задания масштаба графика используйте элемент Property node.

Сохраните ВП под именем *Масштабирование графика.vi*

Примечания

Урок 8
Принятие решений в ВП и структуры

В этом уроке излагаются методы принятия решений в ВП, а также рассказывается о работе со структурами, помогающими упростить выполнение различных математических операций. В частности, рассматриваются функция **Select**, структура **Case**, узел Формулы (**Formula Node**) и узел Математики (**MathScript Node**). В уроке также описываются возможности ВП **Formula Express VI**.

В этом уроке изложены вопросы:

A. Функция **Select** и принятие решений.

B. Использование структуры **Case**.

C. Использование узла Формулы.

D. Использование узла Математики.

A. Функция Select и принятие решений

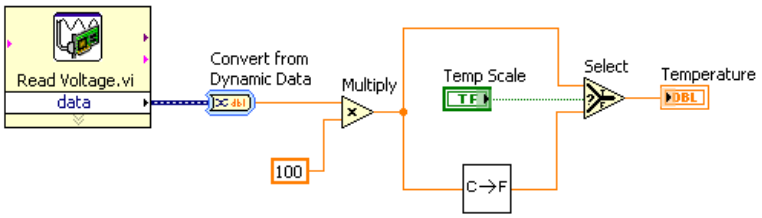
Каждый ВП до этого места курса выполнялся в порядке, определяемом потоком данных. Однако, бывают случаи, когда по ходу программы должно быть принято решение. Например, если происходит событие A, то необходимо сделать B, а если происходит C – то D.

В программах, написанных на текстовых языках программирования, эта задача решается операторами if – else, операторами case, switch и т.д. В LabVIEW реализовано много различных способов принятия решений. Самый простой из них - функция **Select**.

Функция Select

Функция **Select**, расположенная в палитре **Functions»Programming» Comparison**, в зависимости от значения на логическом входе выбирает одно из двух значений. Если на логическом входе будет значение TRUE, то выходе функция выдаст значение, поданное на вход **t**, если же на логическом входе FALSE, то возвращается значение с поля **f**.

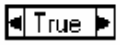
Функция **Select** использовалась в упражнении 3-3, **ВП Термометр**, для определения температурной шкалы. Блок-диаграмма упомянутого ВП приведена ниже.



При необходимости принимать более сложные решения может понадобиться структура **Case**.

B. Структура Case.

Структура Case, показанная слева, имеет две или более поддиаграммы вариантов. Только одна поддиаграмма варианта видима в данный момент времени и только одна поддиаграмма варианта работает при выполнении данной структуры. Входное значение терминала селектора структуры определяет, какая поддиаграмма будет выполняться в данный момент времени. Структура **Case** аналогична операторам case или логическим операторам (if...then...else) в текстовых языках программирования.



Селектор структуры **Case**, расположенный сверху графического изображения Структуры, показанный слева, состоит из указателя значения варианта в центре и стрелок прокрутки по сторонам. Эти стрелки используются для просмотра возможных Вариантов.

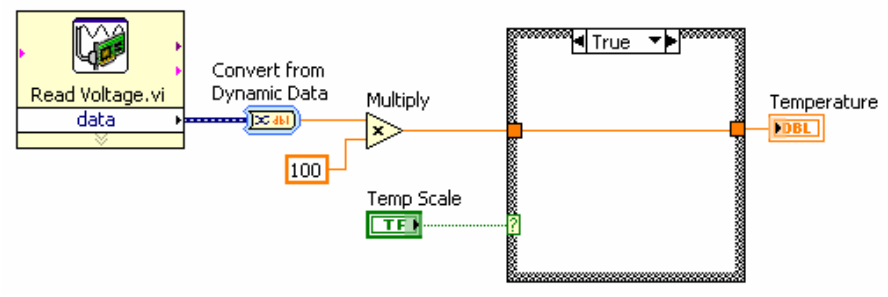
Значение, подаваемое на терминал селектора варианта, показанный слева, определяет, какая поддиаграмма структуры, или вариант, будет выполняться. Допустимо использовать целочисленный, логический, строковый типы, а также тип перечисления в качестве значения, подаваемого на терминал варианта. Терминал варианта может располагаться в любом месте левой границы структуры **Case**. Если терминал Варианта логического типа, то структура состоит из двух логических вариантов TRUE и FALSE. Если терминал варианта имеет один из следующих типов: целочисленный, строковый или перечисления, то количество вариантов может достигать 2^{31}-1 вариантов.

Для использования структуры **Case** необходимо отметить вариант по умолчанию**.** Вариант по умолчанию или поддиаграмма по умолчанию выполняется, если значение терминала варианта выходит за пределы диапазона или не существуют вариантов для возможных значений терминала варианта.

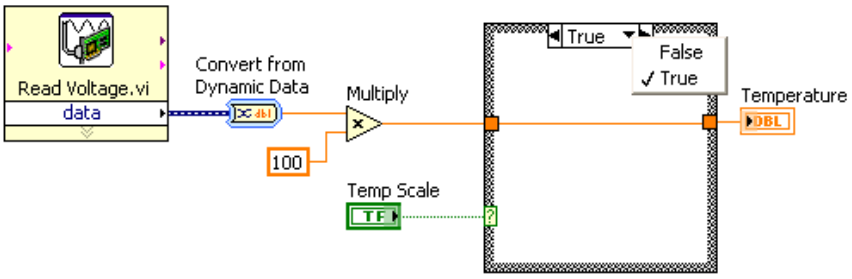
Щелчок правой кнопки мыши на границе структуры **Case** позволяет добавлять, дублировать, перемещать и удалять варианты (поддиаграммы), а также отмечать вариант по умолчанию.

Выбор варианта

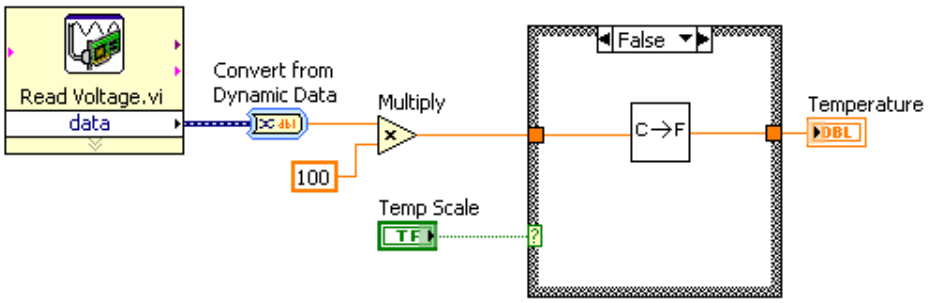
В качестве примера использования структуры **Case** вместо функции **Select** приведена измененная блок-диаграмма **ВП Термометр**. На переднем плане структуры **Case** показан логический вариант TRUE.



Определение варианта осуществляется либо выбором значения на селекторе структуры **Case**, либо вводом значения с помощью инструмента ВВОД ТЕКСТА.



При выборе какого-либо варианта, он появляется на переднем плане, как показано на следующей блок-диаграмме.



Значения селектора варианта должны быть того же типа, что и тип данных, подаваемых на терминал селектора варианта. Значение селектора варианта, окрашенное красным цветом, показывает, что его необходимо удалить или отредактировать, иначе ВП не будет выполняться. Нельзя подавать числа с плавающей точкой на терминал селектора варианта, так как возможны ошибки округления и возникновение ситуации неопределенности. Если подать число с плавающей точкой на терминал селектора варианта, LabVIEW округлит это значение до ближайшего четного целого. Если число с плавающей точкой введено непосредственно в селектор варианта, то оно окрашивается в красный цвет и должно быть удалено или отредактировано.

Терминалы входа и выхода

Структура **Case** допускает использование входных и выходных терминалов данных. Терминалы входных данных доступны во всех поддиаграммах, но их использование поддиаграммой структуры необязательно. Создание выходного терминала на одной поддиаграмме приводит к его появлению на других поддиаграммах в том же самом месте границы структуры. Если хотя бы в одной поддиаграмме выходной терминал не определен, то поле этого терминала окрашивается в белый цвет, что говорит об ошибке создания структуры. Необходимо определять значения выходных терминалов во всех вариантах (поддиаграммах). Кроме того, выходные терминалы должны иметь значения совместимых типов.

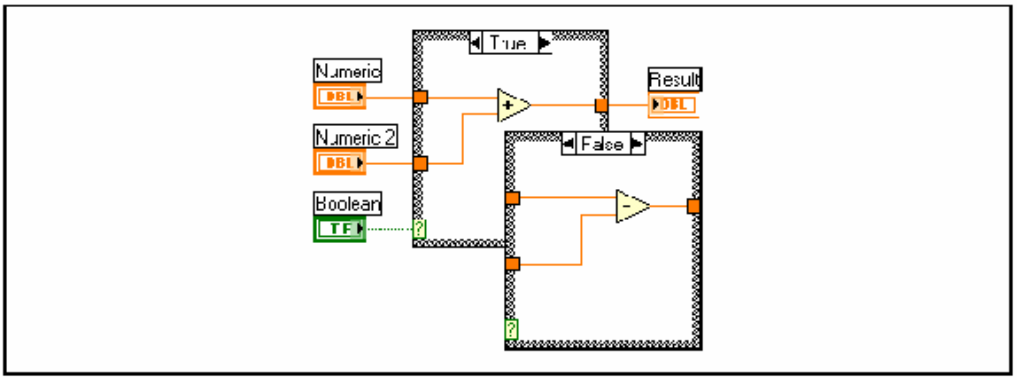
Для определения значения выходного терминала следует правым щелчком мыши по терминалу вызвать контекстное меню и выбрать пункты: **Create»Constant** или **Create»Control**.

Примеры

Следующие примеры показывают, как значения входных терминалов структуры **Case** складываются или вычитаются в зависимости от значения терминала варианта.

Логическая структура Case

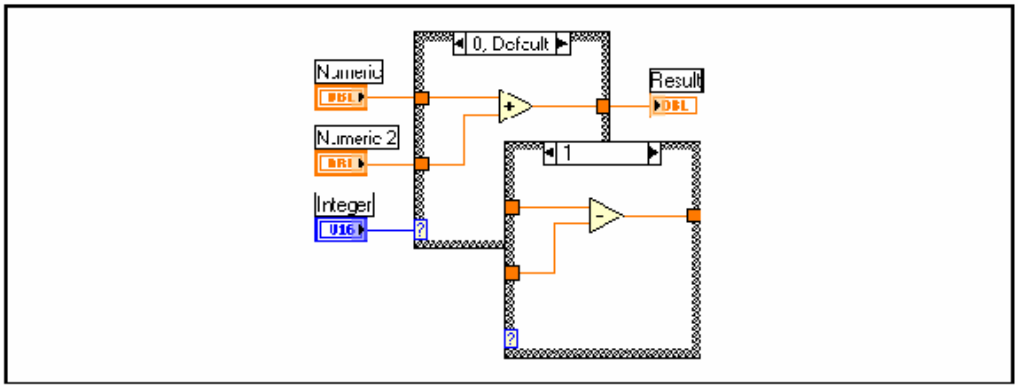
Ниже на рисунке приведен пример логической структуры **Case**. Варианты структуры наложены друг на друга для упрощения иллюстрации.



Если в терминал логического элемента управления, соединенный проводником данных с терминалом селектора варианта, введено значение TRUE, то выполняется сложение; если введено значение FALSE, то выполняется вычитание значений числовых элементов управления.

Целочисленная структура Case

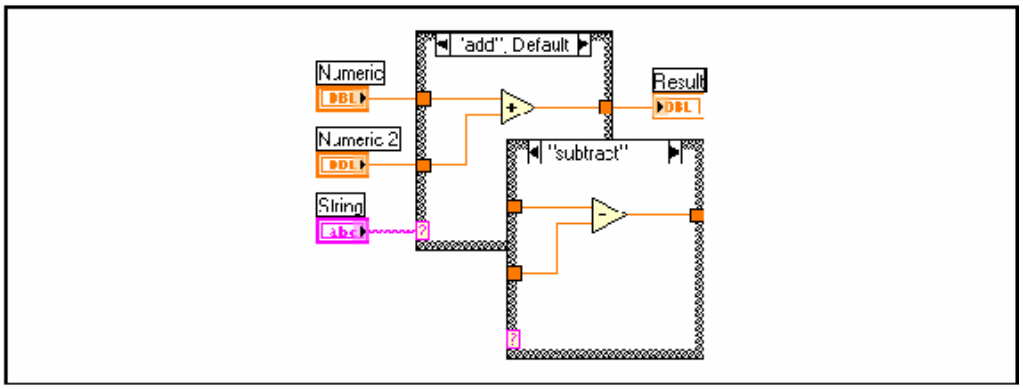
Ниже на рисунке показан пример целочисленной структуры **Case.**



Терминал **Integer** соответствует элементу управления **ring control** (списка с циклическим перебором значений), расположенного в палитре **Controls»Modern»Ring & Enum**. Если значение элемента управления **ring control** равно **0** (сложить), то ВП складывает числа; если равно **1** (вычесть), то ВП производит вычитание чисел. Если значение элемента управления отлично от **0** (сложить) и **1** (вычесть), то ВП складывает числа, т.к. этот вариант выполняется по умолчанию.

Строковая структура Case

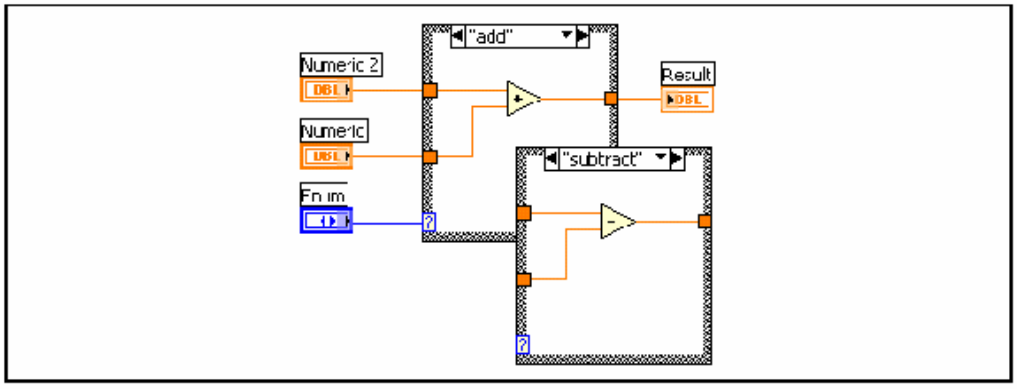
Ниже на рисунке показан пример строковой структуры **Case**.



Если в поле элемента управления введена строка **add**, то ВП производит сложение чисел и вычитает их, если введено значение **subtract**.

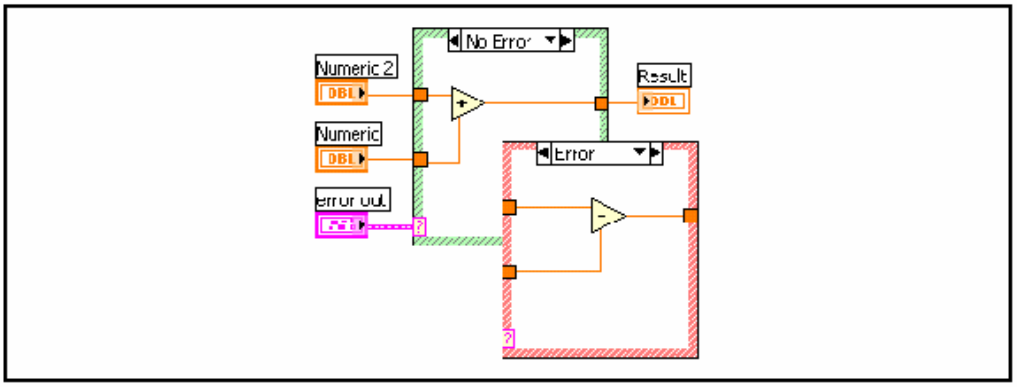
Структура Case по перечислениям

Ниже на рисунке показан пример структуры **Case** по перечислениям.



Структура Case для кластера ошибок

Ниже на рисунке показан пример структуры **Case** для кластера ошибок**.**



В этом примере на терминал селектора структуры **Case** подается кластер ошибок **error out**. В этом случае есть только два варианта структуры: **Ошибка** и **Нет ошибки**, для которых граница структуры имеет красный и зеленый цвет соответственно. Структура **Case** выполняет вариант, основываясь на информации о наличии ошибки.

Структура **Case** реагирует только на логическую переменную **status** кластера ошибок.

Упражнение 8-1 ВП Извлечение квадратного корня

Цель: Изучить структуру Case.

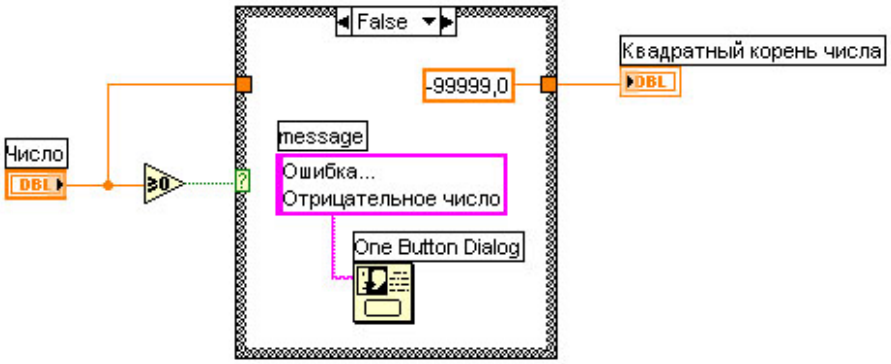
Выполните следующие шаги для построения ВП, который проверяет входное число на знак, вычисляет его квадратный корень или выдает сообщение об ошибке, если число отрицательное.

Лицевая панель

1. Откройте новый ВП и создайте лицевую панель, как показано ниже.

Блок-диаграмма

2. Создайте блок-диаграмму, показанную ниже на рисунке:



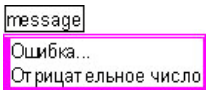
Поместите на блок-диаграмму структуру **Case**, расположенную в палитре **Functions»Programming»Structures.**

Используйте стрелки уменьшения или увеличения селектора структуры для выбора варианта FALSE.

Поместите на блок-диаграмму функцию **Greater or Equal to 0?**, расположенную в палитре **Functions»Programming»Comparison**. Функция возвращает значение TRUE, если число больше или равно 0.

Щелкните правой кнопкой мыши по численной константе и в контекстном меню выберите пункт **Format & Precision**. Установите **Digits of Precision** равным 1, выберите вид представления **Floating Point Notation** и нажмите кнопку **OK**.

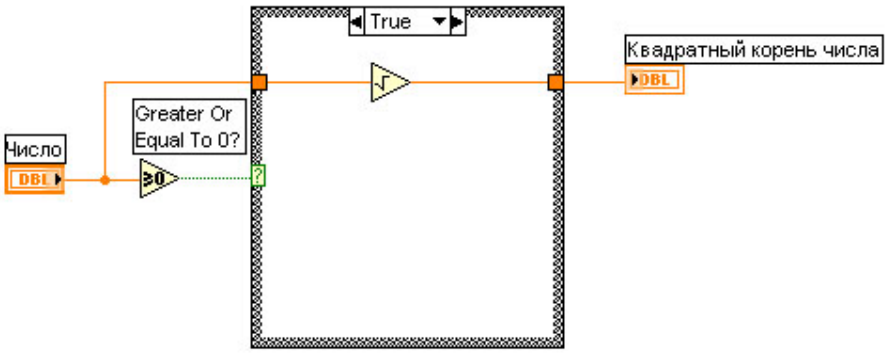
Поместите на блок-диаграмму **One Button Dialog**, расположенный в палитре **Functions»Programming»Dialog & User Interface**. Это диалоговое окно будет отображать сообщение «Ошибка… Отрицательное число».



Щелкните правой кнопкой мыши по полю ввода/вывода **message** функции **One Button Dialog** и в контекстном меню выберите пункт **Create»Constant**. Введите текст «Ошибка… Отрицательное число». Для получения более подробной информации смотрите Урок 9, Строки и файловый ввод/вывод.

3. Выберите вариант TRUE.

Поместите функцию **Square Root** на блок-диаграмму, как показано ниже. Функция размещена в палитре **Functions»Programming» Numeric.** Она возвращает квадратный корень входного числа.



4. Сохраните созданный ВП под именем *Извлечение квадратного корня.vi*

Запуск ВП

5. Отобразите лицевую панель и запустите ВП.

Внимание Не запускайте ВП кнопкой непрерывного запуска, так как при определенных обстоятельствах запуск этого ВП в непрерывном режиме может привести к бесконечному циклу.

Если входное значение элемента управления **Число** положительно, то выполнится поддиаграмма варианта TRUE и вычисляется значение квадратного корня. Если значение элемента **Число** является отрицательным, то выполнится поддиаграмма варианта FALSE, которая возвращает –99999,0 и отображает диалоговое окно с сообщением «Ошибка… Отрицательное число».

6. Закройте ВП.

Конец упражнения 8-1

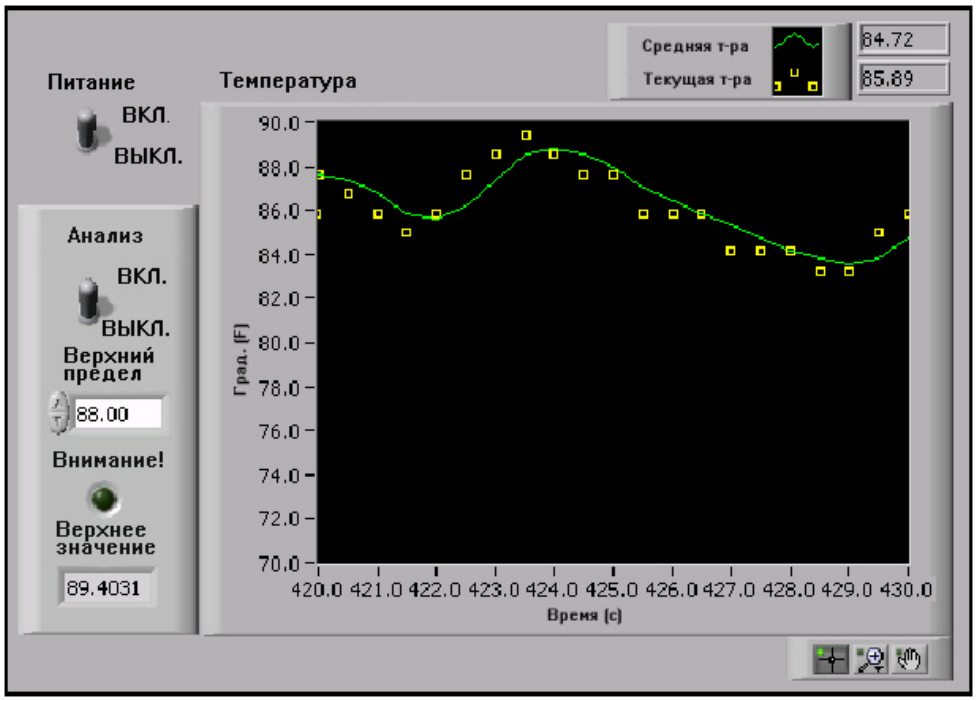
Упражнение 8-2 ВП Контроль температуры

Цель: Изучить работу структуры Case.

Выполните следующие шаги для создания ВП, определяющего моменты выхода температуры за границы заданного диапазона, а также наибольшее значение температуры.

Лицевая панель

1. Откройте ВП *Расчет средней температуры.vi*, созданный в упражнении 7-2.
2. Измените лицевую панель, как показано на рисунке ниже.



Поместите на лицевую панель объект **Horizontal Smooth Box**, расположенный в палитре **Controls»Modern»Decorations**. Этот объект будет использоваться для визуальной группировки элементов анализа.

Создайте копию переключателя **Питание** и назовите его **Анализ**. Щелкните по нему правой кнопкой мыши и выберите из контекстного меню пункт **Mechanical Action»Switch When Pressed**.

Поместите на лицевую панель числовой элемент управления, расположенный в палитре **Controls»Modern»Numeric**, и назовите его **Верхний предел**.

Поместите на лицевую панель индикатор **Round LED**, расположенный в палитре **Controls»Modern»Boolean**, и назовите его **Внимание!**

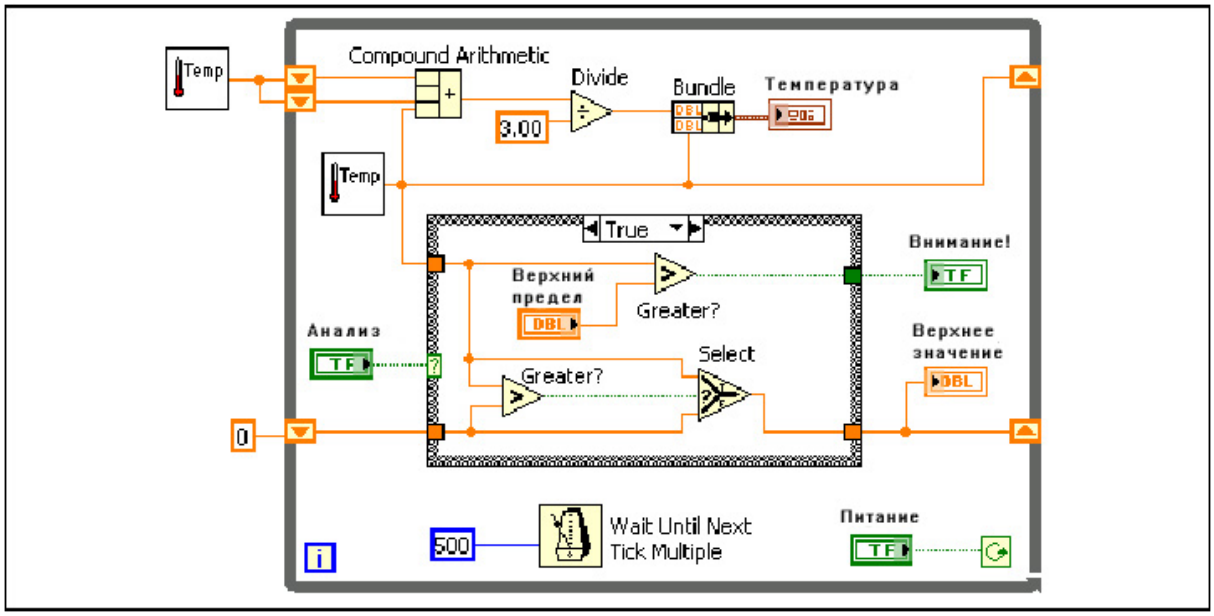
Поместите на лицевую панель числовой элемент отображения данных, расположенный в палитре **Controls»Modern»Numeric**, и назовите его **Максимальное значение**.

Щелкните правой кнопкой мыши по диаграмме для отображения контекстного меню. Далее, выберите пункт **Visible Items»Digital Display** для появления цифрового элемента отображения.

3. Сохраните ВП под именем *Контроль температуры.vi*

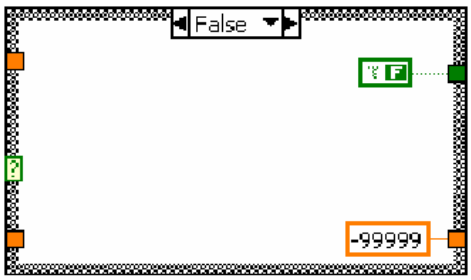
Блок-диаграмма

4. Измените блок-диаграмму, как показано ниже на рисунке. При необходимости растяните цикл **While** так, чтобы внутрь него поместились все элементы.



Поместите на блок-диаграмму функцию **Greater?**, размещенную в палитре **Functions»Programming»Comparison**. Эта функция возвращает значение TRUE, если температура превышает величину в элементе **Верхний предел**. В противном случае функция возвращает значение FALSE.

5. Заполните вариант структуры FALSE, как показано ниже. Если переключатель **Анализ** находится в выключенном состоянии, то индикатор не горит, и в элемент отображения данных **Верхнее значение** выводится число -99999.



6. Сохраните ВП, потому что он будет использоваться в дальнейшем при изучении курса.

Запуск ВП

7. Перейдите на лицевую панель, введите число 80 в элемент управления **Верхний предел** и запустите ВП.

   Если переключатель **Анализ** выключен, то индикатор не горит, и в элементе **Верхнее значение** отображается число -99999. При включенном переключателе **Анализ** в элементе **Верхнее значение** отображается максимальная температура на данный момент. Если температура превышает значение, введенное в элементе **Верхний предел**, то загорается индикатор светодиода **Внимание!**

8. Закройте ВП.

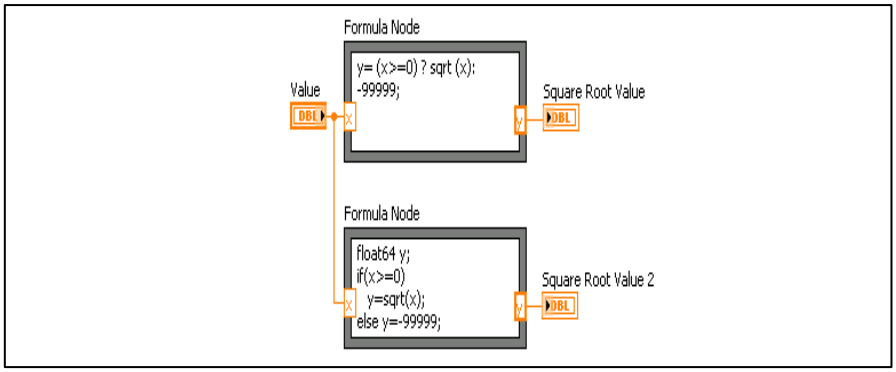
Конец упражнения 8-2

C. Узел Формулы

Узел Формулы (**Formula Node**) используется для выполнения математических операций в текстовом виде на блок-диаграмме. Использовать узел Формулы удобно, когда выражения имеют много переменных, или они достаточно сложные. Для ускорения процесса создания алгоритма можно копировать и вставлять имеющиеся текстовые математические коды в узел Формулы вместо их воссоздания на блок-диаграмме.

Создание терминалов входных и выходных данных узла Формулы осуществляется щелчком правой кнопки мыши по границе узла. В контекстном меню необходимо выбрать пункты **Add Input** или **Add Output**, а затем ввести переменные для входа и выхода**.** Далее вводится уравнение в рабочую область структуры. Каждое выражение должно заканчиваться разделителем (**;**).

Узел Формулы может также использоваться для принятия решений. На следующей блок-диаграмме показаны два эквивалентных способа применения операторов if – then в узле Формулы.



Узел Формулы позволяет производить разнообразные математические операции. Для получения более подробной информации о функциях, операциях и синтаксисе узла Формулы используйте справку **LabVIEW Help**.

Замечание ВП **Formula Express**, расположенный в палитре **Functions»Express»Arith/Compare**, является, по сути, встроенным в LabVIEW научным калькулятором. Он может выполнять большинство операций, выполняемых узлом Формулы, однако только по одной операции за раз. Для получения более подробной информации о ВП **Formula Express** используйте справку **LabVIEW Help**.

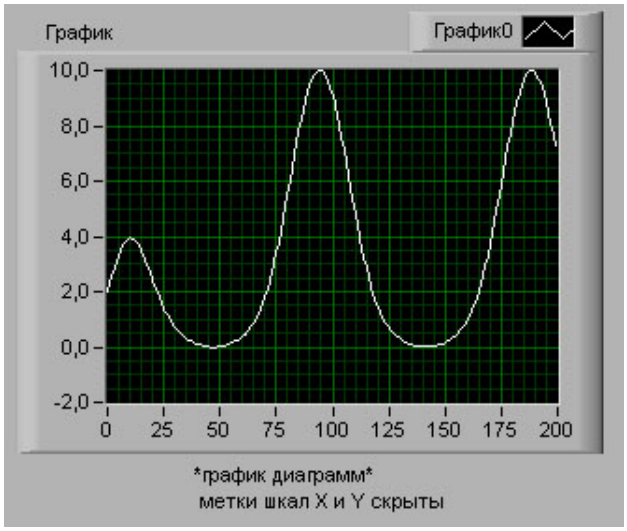
Упражнение 8-3 ВП Узел Формулы

Цель: Изучить работу структуры узел Формулы.

Выполните следующие пункты для построения ВП, который использует узел Формулы для выполнения нескольких математических операций и выводит результаты в виде графика.

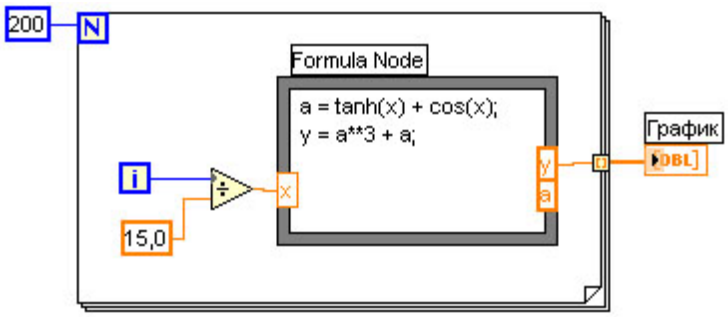
Лицевая панель

1. Откройте новый ВП и постройте лицевую панель, как показано ниже на рисунке.



Блок-диаграмма

2. Постройте блок-диаграмму, показанную на рисунке:



Поместите на блок-диаграмму узел Формулы (**Formula Node**), расположенный в палитре **Functions»Programming»Structures**.

3. Создайте входной терминал **x**, щелкнув правой кнопкой мыши по левой границе структуры и выбрав пункт **Add Input** из контекстного меню. Введите значение **x** в появившееся окно.

4. Создайте терминалы выходных данных **y** и **a**, щелкнув правой кнопкой мыши по левой границе структуры и выбрав пункт **Add**

Output из контекстного меню. Допустимо открытие выходного терминала для промежуточных переменных, например **a**.

Замечание При открытии входных или выходных терминалов имя переменной должно точно соответствовать имени переменной, используемой в арифметическом выражении. Также необходимо учитывать то, что прописные и заглавные буквы различаются.

5. Введите следующие уравнения в узел Формулы (** - это оператор степени). Для получения информации о синтаксисе узла Формулы используйте справку **LabVIEW Help**.

   a = tanh(x) + cos(x);

   *y = a**3 + a;*

6. Сохраните ВП под именем *Узел Формулы.vi*

Запуск ВП

7. Отобразите лицевую панель и запустите ВП. На графике отобразится кривая, соответствующая функции $y = f(x)^3 + f(x)$, где $f(x) = \tanh(x) + \cos(x)$.

   В каждой итерации ВП делит значения терминала текущей итерации на 15,0. Частное является аргументом **x** узла Формулы, по которому рассчитывается значение функции **y**. Далее ВП строит график массива **y**.

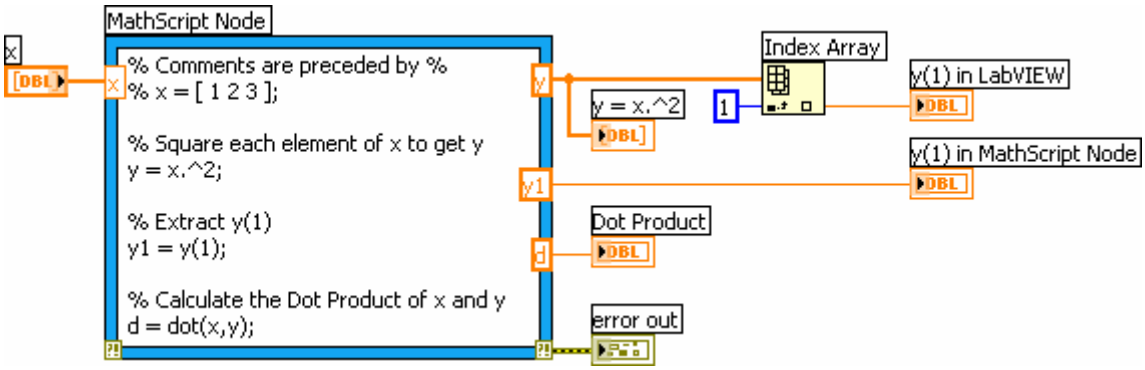
8. Закройте ВП.

Конец упражнения 8-3

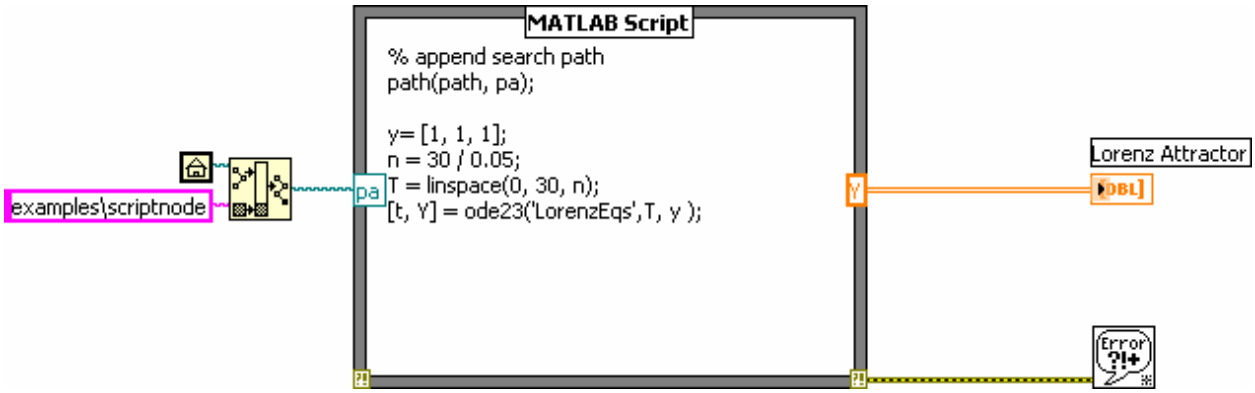
D. Узел Математики (MathScript Node)

Подобно узлу Формул узел Математики используется для выполнения математических операций в текстовом виде на блок-диаграмме. Как и при использовании узла Формул, Вы можете передавать и получать данные из узла кода. Однако, в отличие от узла Формул, узел Математики обладает более расширенными возможностями. Используя узел Математики, Вы сможете импортировать в LabVIEW код, написанный в **LabVIEW MathScript, MATLAB** или **Xmath**.

На рисунке приведен пример окна Математики в LabVIEW:



Вы можете использовать узел **MathScript Node** для создания, загрузки и редактирования кодов, написанных на MATLAB, даже, если MATLAB не инсталлирован на Вашем компьютере. Однако, надо иметь в ввиду, что MathScript поддерживает не все функции, поддерживаемые MATLAB. Если же у Вас на компьютере инсталлирована Xmath или MATLAB версии 6.5 и выше, то Вы можете использовать любые части кода, написанные на Xmath или MATLAB.



MATLAB и Xmath инсталлируют свои программы обработки кода, LabVIEW общается с данными программами с помощью соответствующего узла кода. Вы можете заменить, используемую программу обработки кода, например, конвертировать Xmath Script Node в MATLAB script node. Для этого нажмите правой кнопкой мыши на границу узла кода в выберите из контекстного меню **Choose Script Server»Xmath Script Node** или **Choose Script Server»MATLAB Script Node.** Для **MathScript Node** изменить программу обработки кода нельзя.

Узлы **MATLAB Script Node** и **Xmath Script Node** находятся на палитре **Functions»Mathematics»Scripts & Formulas»Script Nodes**. Узел **MathScript Node** находится на палитре **Functions»Mathematics»Scripts & Formulas**, либо на палитре **Functions»Programming»Structures.**

Примечание MathScript Node, MATLAB script node и **Xmath Script Node** доступны только при использовании операционной системы Windows.

Примечание MathScript может использоваться только с LabVIEW Professional Development System.

Использование окна LabVIEW MathScript

Для создания математического кода в LabVIEW Вы также можете использовать **LabVIEW MathScript Window**. Чтобы открыть окно **LabVIEW MathScript Window** выберите на линейке инструментов пункт **Tools»MathScript Window**. **LabVIEW MathScript Window** генерирует значения на выходе, содержит историю вызываемых команд, список переменных, которые Вы определяете, и отображает выбранные переменные.

Вы можете сохранить скрипты, созданные в **LabVIEW MathScript Window**, и загрузить их в MathScript Node. И наоборот, скрипты созданные и сохраненные в **MathScript Node**, можно загрузить в **LabVIEW MathScript Window.**

Для большей информации о LabVIEW MathScript обратитесь к *LabVIEW Help.* В содержании найдите пункт **Fundumentals»Formulas and Equations**.

Краткое изложение пройденного материала, советы и секреты

- Функция **Select** возвращает одно из двух входных значений в зависимости от значения на третьем логическом входе.
- Структура **Case** может иметь два или более вариантов (поддиаграмм). Только одна поддиаграмма видна на блок-диаграмме, и только одна поддиаграмма может выполняться в момент времени.
- Если терминал варианта логического типа, то структура состоит из двух логических вариантов: TRUE и FALSE. Если терминал варианта имеет типы: целочисленный, строковый или перечисления, то количество вариантов может достигать значений 2^{31}-1.
- Для того чтобы избежать выполнения подпрограммы ВП в случае ошибки необходимо подсоединить входной кластер ошибок к терминалу селектора и поместить весь код подпрограммы ВП в Вариант **Нет ошибки** структуры **Case**.
- Использование узла Формулы полезно, когда арифметическое выражение имеет много переменных или достаточно сложное, а также для уже имеющегося текстового математического кода. Каждое выражение должно заканчиваться разделителем (**;**).

Дополнительные упражнения

8-4. Создайте ВП, который использует узел Формулы для расчета следующих уравнений:

$y_1 = x^3 + x^2 + 5$

$y_2 = mx + b$

Введите оба уравнения в узел Формулы, используйте разделитель (**;**) после каждого уравнения.

Сохраните ВП как файл *Уравнения.vi*

8-5. Создайте ВП, функционирующий как калькулятор. Используйте элементы цифрового управления для ввода численных значений и элемент цифрового отображения для вывода результата арифметических операций: сложение, вычитание, деление, умножение, которые ВП выполняет над вводимыми числами. Для выбора арифметической операции рекомендуется использовать элемент управления **slide control**.

Сохраните ВП под именем *Калькулятор.vi*

8-6. Измените ВП *Извлечение квадратного корня.vi*, созданный в упражнении 8-1, так чтобы ВП выполнял все арифметические и логические операции с помощью узла Формулы.

Сохраните ВП под именем *Извлечение квадратного корня 2.vi*.

8-7. Создайте ВП, который имеет два элемента управления с именами **Порог** и **Входной Массив**, а также один элемент отображения **Выходной массив**. **Выходной массив** должен состоять из элементов **Входного Массива**, значения которых превысят значение **Порога**.

Сохраните ВП под именем *Превышение над порогом.vi*

Создайте другой ВП, который генерирует массив случайных чисел в диапазоне от 0 до 1 и использует ВП *Превышение над порогом.vi* в качестве подпрограммы для создания выходного массива значений элементов превышающих порог 0,5.

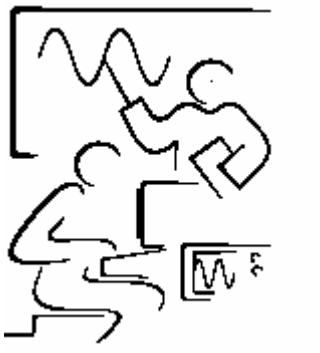
Сохраните ВП под именем *Использование ВП Превышение над порогом.vi*

8-8. Создайте ВП, который генерирует два массива данных: случайные числа и гармонический сигнал. Отобразите их сумму на графике. На втором графике отобразите спектральную плотность мощности суммарного сигнала, вычислите основную частоту.

Сохраните ВП под именем *Зашумленный гармонический сигнал.vi*

Совет Для вычисления спектральной плотности мощности используйте экспресс ВП **Spectral Measurements**, находящийся в палитре **Functions»Express**.

Примечания



Урок 9
Строки и файловый ввод/вывод

Строки объединяют последовательности и являются массивами ASCII символов. Подпрограммы ВП работы с файлами обеспечивают ввод/вывод данных в/из файл(а).

В этом уроке изложены вопросы:

A. Создание строковых элементов управления и отображения данных.

B. Использование некоторых функций обработки строк.

C. Использование файловых функций ввода/вывода.

D. Форматирование текстовых файлов для использования в таблице символов.

E. Использование файлового ввода/вывода высокого уровня.

A. Строки

Строки – это последовательность отображаемых и неотображаемых ASCII символов. Строки обеспечивают независимый от платформы формат обмена данными. Некоторые из наиболее распространенных строковых приложений включают в себя:

- Создание простых текстовых сообщений.
- Передача числовых данных в приборы в виде строк символов и преобразование строк в числовые данные.
- Сохранение числовых данных на диск. Чтобы сохранять числовые данные в виде файла ASCII, необходимо перед записью преобразовать их в строки.
- Диалоговые окна инструкций и подсказок.

На лицевой панели строки появляются в виде таблиц, полей ввода текста и меток.

Создание строковых элементов управления и отображения данных

Для работы с текстом и метками используются строковые элементы управления и отображения данных, расположенные в палитре **Controls»Modern»String & Path**. Создание и редактирование текста в строке производится с помощью инструментов УПРАВЛЕНИЕ и ВВОД ТЕКСТА. Для изменения размера строкового объекта на лицевой панели используется инструмент ПЕРЕМЕЩЕНИЕ. Для экономии места на лицевой панели можно использовать полосу прокрутки. Для этого необходимо щелкнуть правой кнопкой мыши по строковому объекту и выбрать в контекстном меню пункт **Visible Items»Scrollbar**.

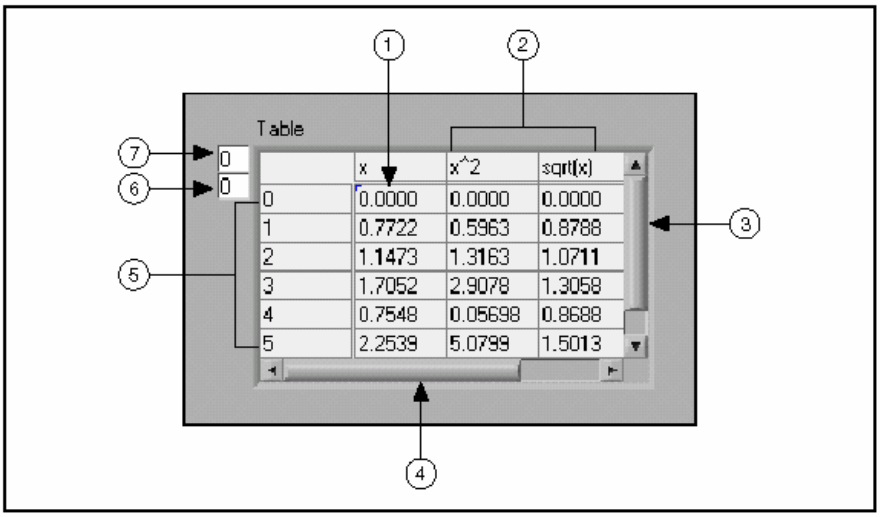
Тип отображения строкового объекта выбирается в его контекстном меню. Типы отображения строки и примеры заполнения поля ввода текста показаны в таблице ниже.

Тип отображения	**Описание**	**Пример текста**
Режим стандартного отображения (**Normal Display**)	Отображает стандартные ASCII коды, используя шрифт элемента управления. Управляющие коды для печати выводятся на экран в виде квадратов.	`There are four display types. \ is a backslash`
Режим отображения с обратным слэшем непечатаемых управляющих кодов (**‘\’ Codes Display**)	Выводит \ для всех непечатаемых управляющих кодов	`There\sare\sfour\s display\stypes.\n\ \\sis\sa\sbackslas h`
Режим скрытого отображения текста (**Password Display**)	Выводит * для всех кодов текстового пространства	`********** ********** **********`
Режим отображения 16-тиричных ASCII	Выводит значение ASCII кода для каждого символа	`5468 6572 6520 6172 6520 666F`

кодов (**Hex Display**)		7572 2064 6973 706C 6179 2074 7970 6573 2E0A 5C20 6973 2061 2062 6163 6B73 6C61 7368 2E

Таблицы

Элемент управления Таблица, расположенный в палитре **Controls»Modern»List & Table** предназначен для создания таблиц на лицевой панели. Каждая ячейка находится в строке и столбце таблицы. Поэтому таблица отображает двумерный массив строк. Ниже показана таблица и ее составные части.



1. Ячейка таблицы
2. Заголовок столбца
3. Вертикальная полоса прокрутки
4. Горизонтальная полоса прокрутки
5. Заголовок строки
6. Индекс по горизонтали
7. Индекс по вертикали

Для инициализации значений ячеек таблицы используется инструмент УПРАВЛЕНИЕ или ВВОД ТЕКСТА, с помощью которых достаточно ввести текст в выделенную ячейку.

Таблица – это двумерный массив строк. Таким образом, для использования таблицы в качестве элемента отображения данных, необходимо двумерный массив чисел преобразовать в двумерный массив строк. Заголовки строк и столбцов таблицы, как в таблице символов, автоматически не отображаются. Необходимо создать одномерный массив строк, содержащий заголовки строк и столбцов таблицы.

B. Функции работы со строками

Для редактирования и управления строками на блок-диаграмме следует пользоваться функциями обработки строк, расположенными в палитре **Functions»Programming»String**. Некоторые из функций работы со строками рассмотрены ниже:

- **String Length** – выдает количество символов в строке, включая пробелы. Например, функция **String Length** выдает значение 19 для приведенного ниже текста: *The quick brown fox*
- **Concatenate Strings** – объединяет строки и одномерные массивы строк в отдельную строку. Для увеличения полей ввода данных функции следует изменить ее размер. Например, объединив предыдущую строку со следующим массивом строк,

jumped	over	the	lazy	dog

  функция **Concatenate Strings** на выходе выдает следующую строку**:** *The quick brown fox jumped over the lazy dog*
- **String Subset** – выдает подстроку определенной длины **length**, начиная со значения **offset** (смещение). Смещение первого элемента в строке равно 0. Например, если на поле ввода данных функции подать предыдущую строку, то функция **String Subset** при **offset**=4 и **length**=5 выдаст значение: *quick*.
- **Match Pattern** – ищет повторяющуюся последовательность, поданную на поле ввода данных **regular expression**, в строке начиная со значения смещения **offset**, и, если находит соответствие, разбивает строку на три подстроки. Если соответствие не найдено, поле вывода данных **match substring** является пустым, а значение поля вывода данных **offset past match** (смещение повторяющейся последовательности в строке) равно `-1`. Например, на поле **regular expression** (шаблон подстроки) подается значение **:**, а строка на входе *VOLTS DC: +1.22863E+1***.**

  Функция **Match Pattern** выдаст величины **before substring** (перед подстрокой) *VOLTS DC*, **match substring** (шаблон подстроки) **:** и **after substring** (после подстроки) *+1.22863E+1*, а также **offset past match** равный 9.

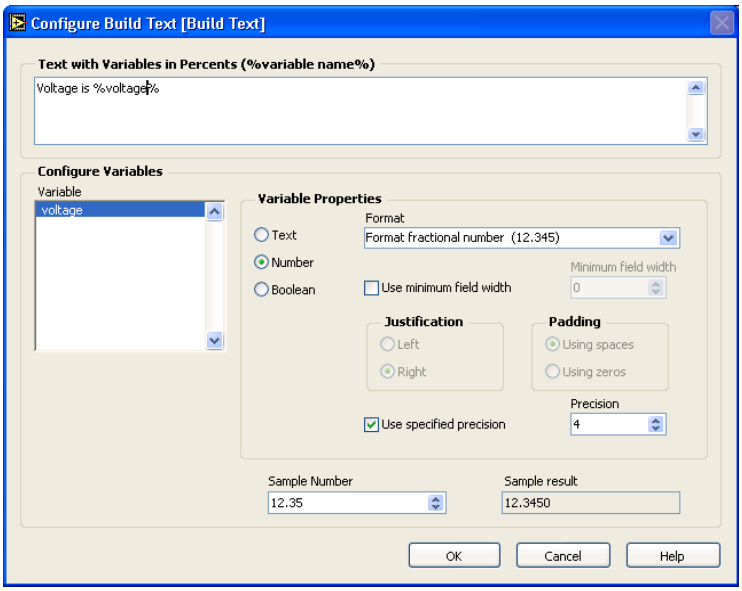
Преобразование числовых данных в строку

Для преобразования числовых данных в строковые используются ВП **Build Text Express** и функция **Format Into String** (конвертирование в строку). Обе эти функции имеют входные и выходные кластеры ошибок. При недостатке места на блок-диаграмме лучше использовать функцию **Format Into String**.

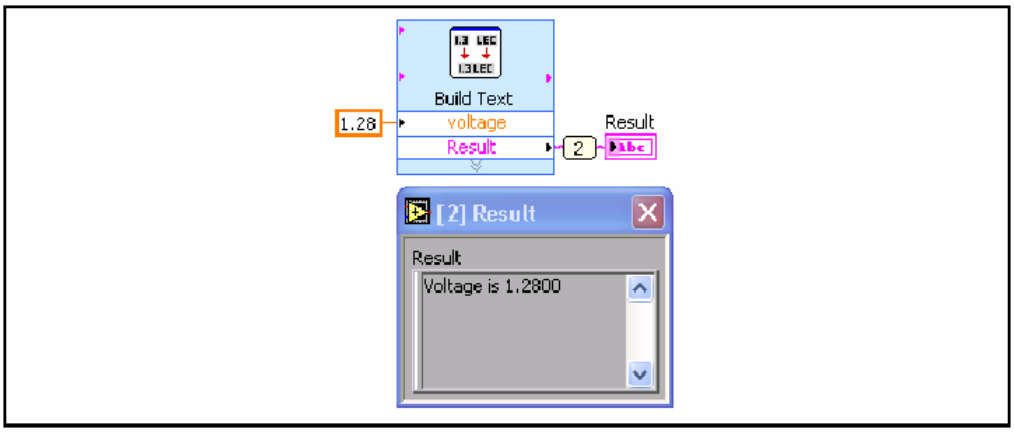
Экспресс-ВП Build Text Express VI

Экспресс-ВП **Build Text**, расположенный в палитре **Functions»Express»Output** производит объединение входных строк. Если входные величины имеют не строковый тип данных, то они преобразуются в строку в соответствии с настройками этого экспресс-ВП.

При помещении Экспресс-ВП **Build Text** на блок-диаграмму появляется диалоговое окно настроек **Configure Build Text**. В следующем примере значение напряжения подается на вход экспресс-ВП и преобразуется к формату данных с плавающей запятой с 4-мя числами после запятой. Затем это значение добавляется к концу строки **Voltage is** (Напряжение равно).



При такой настройке экспресс-ВП выглядит следующим образом. Для наблюдения за выходной строкой используется отладочный индикатор. Любые значения, подаваемые на поле ввода данных **Beginning Text**, будут присоединяться к началу текста из диалогового окна настроек.

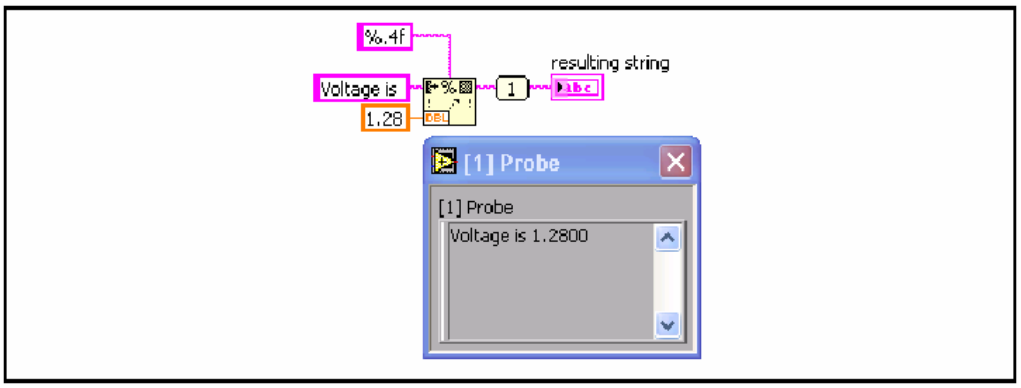


Совет Операция объединения строк называется конкатенацией, т.е. в данном случае при конкатенации две строки сливаются в одну.

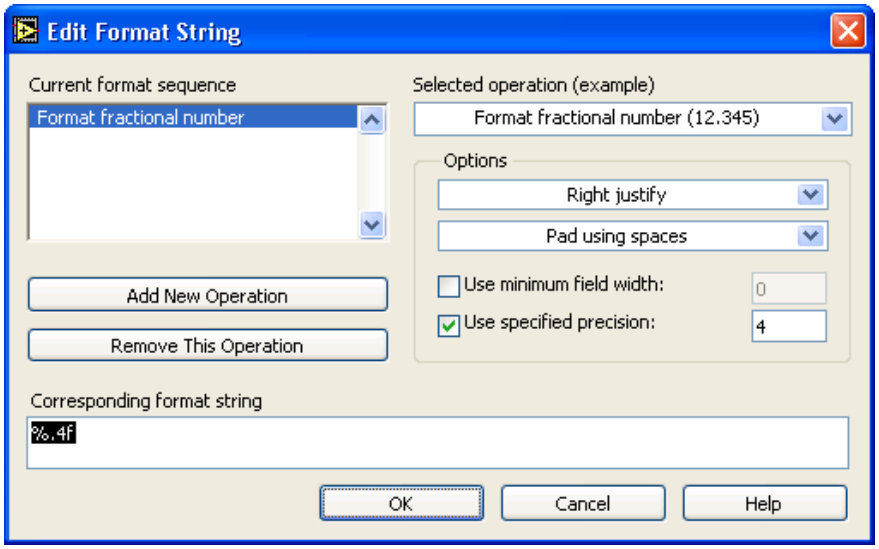
Функция Format Into String

Функция **Format Into String** преобразует параметры любого формата, такие как числовые данные, в строку. Для увеличения количества параметров следует изменить размер функции.

В приведенном ниже примере функция **Format Into String** выдает указанную на отладочном индикаторе строку при значениях полей **format string** (формате строки) %.4f, **input string** (входной строке) *Voltage is* (учитывая пробел в конце), и параметре 1.28**.**



В формате строки % – указывает начало формата строки, **.** – число после точки определяет точность представления числа, 4 – показывает количество знаков после запятой, а f – указывает тип данных с плавающей запятой. Для создания и редактирования формата строки следует щелкнуть правой кнопкой мыши по функции и выбрать пункт контекстного меню **Edit Format String**. Рисунок ниже показывает вид диалогового окна **Edit Format String** из предыдущего примера.



Для получения более подробной информации о синтаксисе форматов, следует обратиться к встроенной в LabVIEW справочной информации (**LabVIEW Help).**

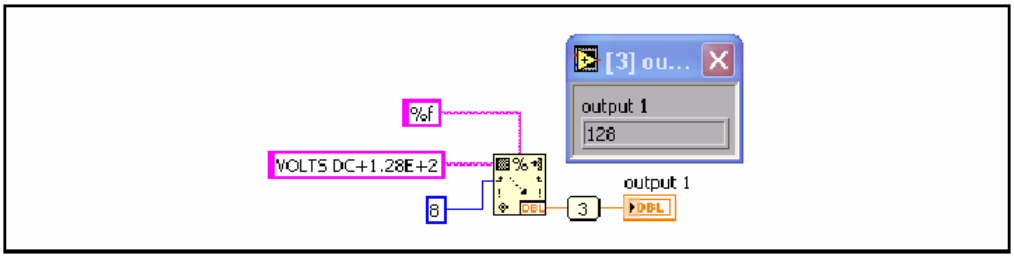
Преобразование строк в числовые данные

Для преобразования строки в числовые данные следует использовать функцию **Scan From String**.

Просмотр и конвертирование строки

Функция **Scan From String** преобразует строку, содержащую допустимые числовые символы, такие как `0-9`, `+`, `-`, `e`, `E` и разделитель `.`, в данные числового формата. Функция начинает просмотр строки, подаваемой на поле ввода данных **input string** с номера символа, задаваемого на поле **initial search location**. Функция может просматривать входящую строку различных типов данных, таких как числовые или логические данные, основываясь на формате строки. Для увеличения количества полей вывода данных следует изменить размер функции**.**

Например, при значениях на полях ввода данных **format string** – %f**,** **initial search location** – 8, **input string** – *VOLTS DC+1.28E+2* функция выдает результат 128.00, как показано ниже.



В формате строки % – указывает начало формата строки, f – указывает тип данных с плавающей запятой. Для создания и редактирования формата строки следует щелкнуть правой кнопкой мыши по функции и выбрать пункт контекстного меню **Edit Scan String**.

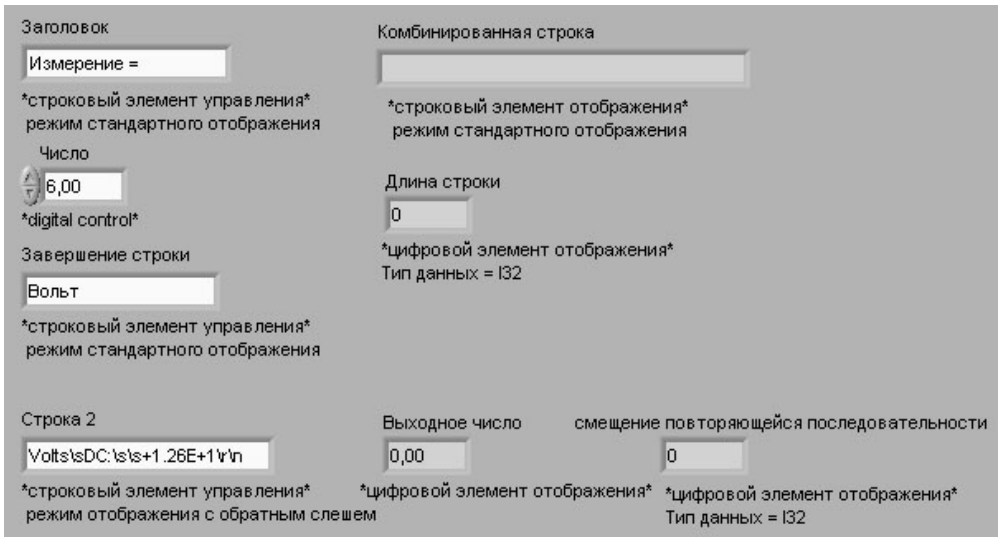
Упражнение 9-1 ВП Компоновка строки

Цель: Приобретение навыков по использованию функций обработки строк.

Ниже приведена последовательность действий для создания ВП, который преобразует числовые данные в строку и объединяет строку с другими строками в одну. Затем после поиска по шаблону полученная часть строки переводится в числовой формат.

Лицевая панель

1. Откройте новый ВП и оформите лицевую панель, как показано ниже на рисунке. Воспроизводить комментарии и подписи к элементам не обязательно.

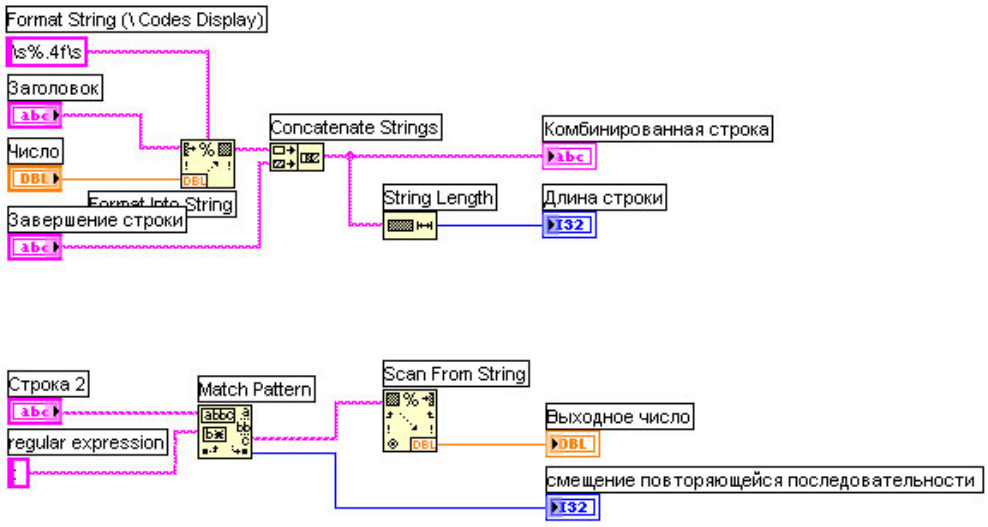


   a. Щелкните правой кнопкой мыши по элементу **Строка 2** и выберите из контекстного меню режим отображения **‘\’ Codes Display**.

   b. Для элементов **Длина строки** и **Смещение повторяющейся последовательности** установите тип представления данных **I32**.

Блок-диаграмма

2. Постройте блок-диаграмму, как показано ниже:



Выберите функцию **Format Into String,** расположенную в палитре **Functions»Programming»String.** Эта функция преобразует число в строку.

a. Щелкните правой кнопкой мыши по функции **Format Into String** и выберите пункт **Edit Format String** для вызова соответствующего диалогового окна.

b. Выделите опцию **Use specified precision** и в поле ввода текста введите значение 4 для преобразования элемента **Число** в строку с четырьмя знаками после запятой.

c. Нажмите на кнопку **OK**. LabVIEW создаст формат строки %.4f , используя выбранную опцию.

d. С помощью инструмента ВВОД ТЕКСТА, введите пробел с обеих сторон **%.4f** и нажмите клавиши <**Shift+Enter**>. Таким образом, на элементе **Комбинированная строка** числовые данные появятся с пробелами с обеих сторон.

e. Щелкните правой кнопкой мыши по константе и выберите режим отображения **'\' Codes Display** из контекстного меню. Введенные пробелы заменятся на \.

Выберите функцию **Concatenate Strings**, расположенную в палитре **Functions»Programming»String**. Эта функция объединит входящие в нее строки в одну.

Выберите функцию **String Length**, расположенную в палитре **Functions»Programming»String.** Эта функция выдаст значение количества символов в объединенной строке **Комбинированная строка**.

Выберите функцию **Match Pattern**, расположенную в палитре **Functions»Programming»String**. Эта функция осуществляет поиск в элементе **Строка 2** по шаблону **:** (двоеточие).

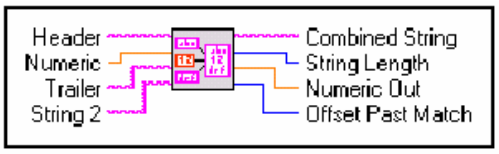
f. Щелкните правой кнопкой мыши по полю **regular expression** и

выберите пункт контекстного меню **Create»Constant**, введите двоеточие и нажмите на клавиши **<Shift+Enter>.**

Выберите функцию **Scan from String**, расположенную в палитре **Functions»Programming»String**. Эта функция преобразует строку после двоеточия в числовые данные.

Иконка ВП и соединительная панель

3. Перейдите на лицевую панель и создайте иконку и соединительную панель для использования созданного ВП в качестве подпрограммы в других ВП. Подробная информация о методах оформления иконки и соединительной панели находится в Уроке 3, *Подпрограммы ВП*.



4. Сохраните ВП под именем *Компоновка строки.vi*. Этот ВП будет использоваться позднее.

Запуск ВП

5. Измените значение элементов на лицевой панели и запустите ВП.

   ВП объединит элементы: **Заголовок**, **Число** и **Завершение строки** в строку **Комбинированная строка** и выдаст значение длины строки.

   ВП также найдет месторасположение подстроки **:** в элементе **Строка 2**. При выполнении ВП преобразует строку после двоеточия в число **Выходное число** и выводит на экран индекс первого элемента после двоеточия в элемент **Смещение повторяющейся последовательности**.

6. Сохраните и закройте ВП.

Конец упражнения 9-1

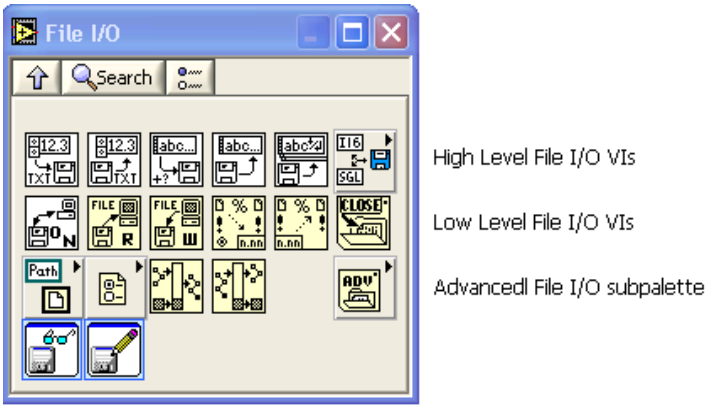
C. Функции файлового ввода/вывода

Функции файлового ввода/вывода производят файловые операции записи и считывания данных. Функции файлового ввода/вывода расположены в палитре **Functions»Programming»File I/O** и предназначены для:

- Открытия и закрытия файла данных.
- Считывания и записи данных из/в файл(а).
- Считывания и записи данных из/в файл(а) в виде таблицы символов.
- Перемещения и переименования файлов и каталогов.
- Изменения характеристик файла.
- Создания, изменения и считывания файлов конфигурации.

Функции файлового ввода/вывода

Палитра функций файлового ввода вывода, показанная ниже, разделена на три части: функции высокого уровня (**high level File I/O**), функции низкого уровня (**low level File I/O**) и подпалитра функций расширенных возможностей (**advanced File I/O**).



Функции файлового ввода/вывода высокого уровня

Функции файлового ввода/вывода высокого уровня расположены в верхней строке палитры **Functions»Programming»File I/O**. Они предназначены для выполнения основных операций по вводу/выводу данных. Более подробную информацию можно получить в разделе E *Использование файлового ввода/вывода высокого уровня.*

Использование функций файлового ввода/вывода высокого уровня позволяет сократить время и усилия программистов при записи и считывании данных в/из файл(а). Функции файлового ввода/вывода высокого уровня выполняют запись и считывание данных и операции закрытия и открытия файла. При наличии ошибок функции файлового ввода/вывода высокого уровня отображают диалоговое окно с описанием ошибок и предлагают на выбор: продолжить выполнение программы или остановить ее. Однако из-за того, что функции данного класса

объединяют весь процесс работы с файлами в один ВП, переделать их под определенную задачу бывает трудно. Для специфических задач следует использовать функции файлового ввода/вывода низкого уровня.

Функции файлового ввода/вывода низкого уровня

Функции файлового ввода/вывода низкого уровня расположены в средней строке палитры **Functions»Programming»File I/O**. Дополнительные функции работы с файлами (**Advanced File I/O**) расположены в палитре **Functions»Programming»File I/O»Advanced File Functions** и предназначены для управления отдельными операциями над файлами.

Функции файлового ввода/вывода низкого уровня используются для создания нового или обращения к ранее созданному файлу, записи и считывания данных и закрытия файла. Функции низкого уровня работы с файлами поддерживают все операции, необходимые при работе с файлами. Информация о дополнительных возможностях функций работы с файлами изложена в учебном курсе LabVIEW Основы II.

Основы файлового ввода/вывода

Стандартные операции ввода/вывода данных в/из файла состоят из следующей последовательности действий:

1. Создание или открытие файла. Указание месторасположения существующего файла или пути для создания нового файла с помощью диалогового окна LabVIEW. После открытия файл LabVIEW создает ссылку на него. Более подробную информацию о ссылке на файл можно найти в разделе *Сохранение данных в новом или уже существующем файле*.
2. Произведение операций считывания или записи данных в/из фал(а).
3. Закрытие файла.
4. Обработка ошибок.

Для осуществления основных операций файлового ввода/вывода используются следующие ВП и функции:

- **Open/Create/Replace File –** открывает, перезаписывает существующий файл, или создает новый. Если **file path** (путь размещения файла) не указан, ВП выводит на экран диалоговое окно, в котором можно создать новый или выбрать уже существующий файл.

- **Read File** – считывает данные из файла, определяемого по ссылке **refnum,** и выдает данные на поле вывода **data**, на поле **count** подается значение количества считываемых данных. Считывание данных начинается с места, определяемого элементами **pos mode** и **pos offset,** и зависит от формата файла.

- **Write File** – записывает данные в файл, определяемый по ссылке **refnum**. Запись начинается с места, определяемого полями ввода данных **pos mode** и **pos offset** для файла потока байтовых данных, и

указателем конца файла для файла протоколированных данных.

- **Close File** – закрывает указанный в ссылке **refnum** файл.

Обработка ошибок

Подпрограммы ВП и функции низкого уровня содержат информацию об ошибках. Для их обработки используются подпрограммы обработки ошибок, такие как **Simple Error Handler VI** (ВП Простой обработчик ошибок), расположенный в палитре **Functions»Programming»Dialog & User Interface**. Поля ввода **error in** и вывода **error out** информации об ошибках используются в каждом ВП для обмена информацией об ошибках между ВП.

Во время работы ВП LabVIEW проверяет наличие ошибок в каждом узле. Если LabVIEW не находит ошибок, то узел выполняется нормально. Если LabVIEW обнаруживает ошибку в одном узле, то его выполнение прерывается, а информация об ошибке передается следующему узлу. Следующий узел поступает так же, и в конце выполнения LabVIEW сообщает об ошибках.

Сохранение данных в новом или уже существующем файле

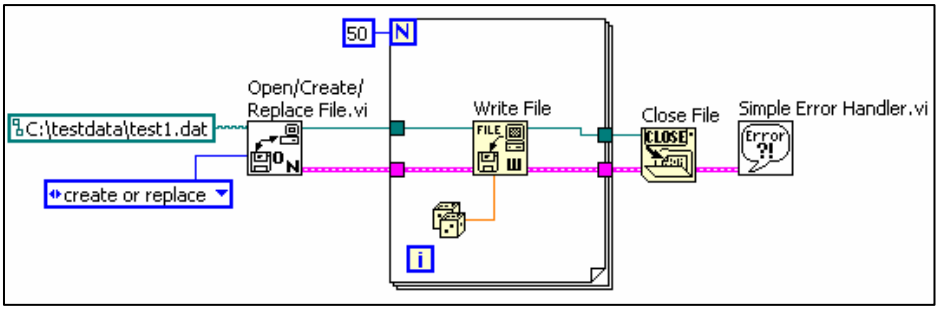
В файл, созданный (или открытый) с помощью функций файлового ввода/вывода, можно записать данные любого типа. При необходимости доступа к файлу со стороны других приложений или пользователей, следует записывать данные в виде строки ASCII символов. Более полную информацию о записи бинарных файлов и файлов регистрации данных можно найти в учебном курсе LabVIEW Основы II.

Доступ к файлу можно осуществить программным путем или с использованием диалогового окна. Для доступа к файлу с помощью диалогового окна на поле ввода **file path** подпрограммы ВП **Open/Create/Replace File VI** не следует подавать данные. Программный доступ к файлу экономит время. Приведенная ниже таблица демонстрирует организацию путей к файлам.

Платформа	Путь к файлу
Windows	Состоит из имени дисковода, двоеточия, обратного слэша, разделяющего директории, и имени файла. Например, *c:\testdata\test1.dat* - путь к файлу *test1.dat* в папке *testdata*
UNIX	Состоит из прямого слэша, разделяющего директории, и имени файла. Например, */home/testdata/test1.dat* – путь к файлу *test1.dat* в папке *testdata* в каталоге *home*. Имя фала и имя директории чувствительны к регистру символов.
MacOS	Состоит из имени диска, двоеточия, имен папок, разделенных двоеточиями, и имени файла. Например, *Hard Disk:testdata:test1.dat* - путь к файлу

	`test1.dat` в папке `testdata` на диске `Hard Disk`

В приведенном ниже примере показано, как записать строку данных в файл при программном указании пути и имени файла. Если файл уже существует, то он перезаписывается, если нет - то создается новый файл.



Подпрограмма ВП **Open/Create/Replace File VI** открывает файл `test1.dat`. ВП также создает ссылку на файл и кластер ошибок.

Совет Ссылка (**refnum**) является уникальным идентификатором для таких объектов как файл, прибор и сетевое соединение.

При открытии файла, устройства или сетевого соединения LabVIEW создает ссылку на объект. Все операции с открытыми объектами выполняются с использованием ссылок.

Кластер ошибок и ссылка на файл последовательно передаются от узла к узлу. Поскольку узел не может выполниться, пока не определены все его входные поля данных, эти два параметра заставляют узлы работать в определенном порядке. Подпрограмма ВП **Open/Create/Replace File VI** передает ссылку на файл и кластер ошибок функции **Write File**, которая производит запись файла на диск. Функция **Close File** закрывает файл после получения кластера ошибок и ссылки на файл из функции **Write File.**

Подпрограмма ВП **Simple Error Handler VI** проверяет наличие ошибок и выводит информацию о них в диалоговом окне. Если в одном из узлов допущена ошибка, последующие узлы не выполняются, и кластер ошибок передается в подпрограмму ВП **Simple Error Handler VI**.

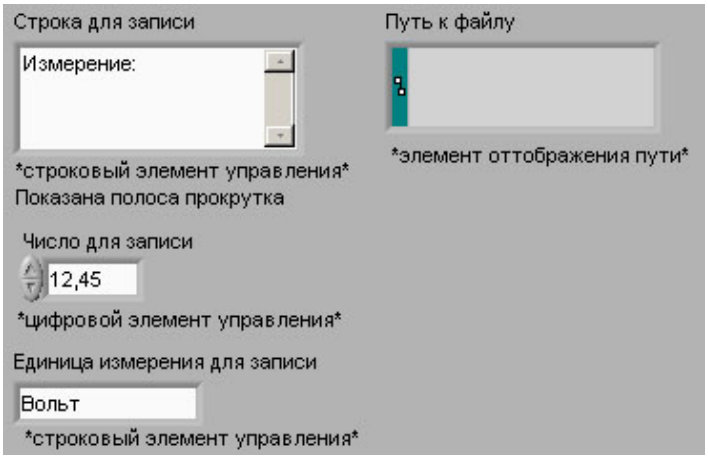
Упражнение 9-2 ВП Запись файла

Цель: Записать данные в файл.

Ниже приведена последовательность действий для создания ВП, который объединяет строку, числовые данные и модуль строки в файл. В упражнении 9-3 будет создан ВП, который считывает и отображает его содержимое.

Лицевая панель

1. Откройте новый ВП и оформите лицевую панель, как показано ниже:

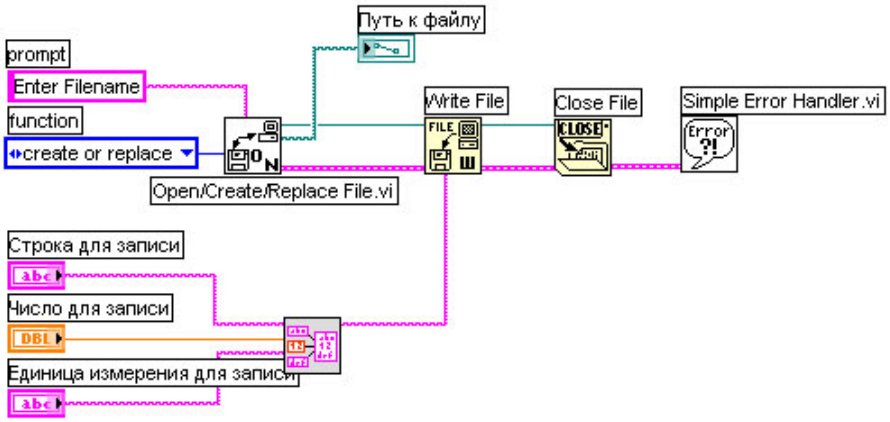


   a. В палитре **Controls»Modern»String & Path** выберите элемент отображения пути. Этот элемент отобразит путь к созданному файлу данных.

   b. Щелкните правой кнопкой мыши по элементу **Строка для записи** и в контекстном меню выберите пункт **Visible Items»Scrollbar**.

Блок-диаграмма

2. Постройте следующую блок-диаграмму:



Выберите в разделе **Functions»Select a VI** ВП *Компоновка строки.vi*, созданный в упражнении 9-1, и поместите его на блок-диаграмму. Этот ВП объединяет три строки в одну.

Поместите на блок-диаграмму подпрограмму ВП **Open/Create/Replace File VI**, расположенную в палитре **Functions»Programming»File I/O.** Этот ВП выводит на экран диалоговое окно для создания файла.

a. Щелкните правой кнопкой мыши по полю **prompt** и в контекстном меню выберите пункт **Create»Constant** для создания константы **Введите имя файла**. При запуске ВП появится окно выбора файла, которое будет называться **Введите имя файла**.

b. Щелкните правой кнопкой мыши по входному полю **function** и в контекстном меню выберите пункт **Create»Constant**. Для выбора пункта выпадающего меню **create or replace** следует использовать инструмент УПРАВЛЕНИЕ.

Выберите функцию **Write File**, расположенную в палитре **Functions»Programming»File I/O**. Эта функция записывает объединенную строку в файл.

Выберите функцию **Close File**, расположенную в палитре **Functions»Programming»File I/O**. Эта функция закрывает файл.

Выберите подпрограмму ВП **Simple Error Handler VI**, расположенную в палитре **Functions»Programming»Dialog & User Interface**. Этот ВП проверяет кластер ошибок и выводит диалоговое окно при возникновении ошибки.

3. Сохраните ВП под именем *Запись файла.vi*.

Запуск ВП

4. Поменяйте значения элементов управления на лицевой панели и запустите ВП. Появится диалоговое окно **Введите имя файла**.

5. Введите в диалоговое окно название файла *демофайл.txt* и нажмите на кнопку **Save** или **OK**.

   ВП запишет в файл данные из элементов **Строка для записи**, **Число для записи** и **Единица измерения для записи**.

6. Закройте ВП.

Конец упражнения 9-2

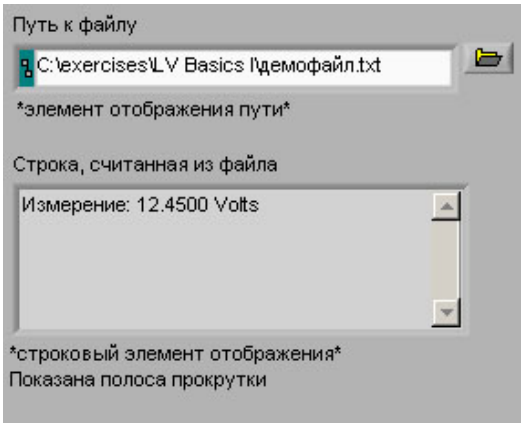
Упражнение 9-3 ВП Чтение файла

Цель: Считать данные из файла.

Ниже приведена последовательность действий для создания ВП, который читает файл, созданный в упражнении 9-2, и выводит данные в строковом элементе отображения.

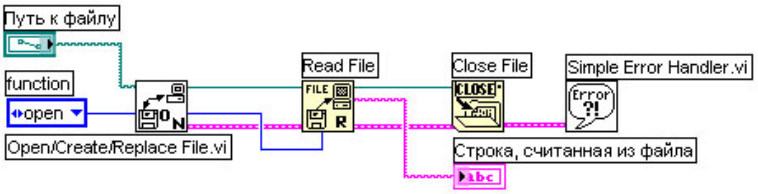
Лицевая панель

7. Откройте новый ВП и создайте лицевую панель, используя элемент управления путем к файлу и строковый элемент отображения в палитре **Controls»Modern»String & Path.**



Блок-диаграмма

8. Постройте следующую блок-диаграмму:



В палитре **Functions»Programming»File I/O** выберите подпрограмму **Open/Create/Replace File VI.** Этот ВП выведет на экран диалоговое окно, которое используется для создания и открытия файла.

a. Щелкните правой кнопкой мыши по входному полю **prompt** и из контекстного меню выберите **Create»Constant** для создания константы **Выбрать имя файла**.

b. Щелкните правой кнопкой мыши по полю **function** и выберите в контекстном меню пункт **Create»Constant** для создания константы. С помощью инструмента УПРАВЛЕНИЕ выберите пункт выпадающего меню **open**.

В палитре **Functions»Programming»File I/O** выберите функцию **Read File.** Эта функция читает количество байт данных с начала файла, определяемое значением поля **count**.

В палитре **Functions»Programming»File I/O** выберите функцию **Close File**. Эта функция закроет файл.

В палитре **Functions»Programming»Dialog & User Interface** выберите подпрограмму **Simple Error Handler VI**. Этот ВП проверяет кластер ошибок и, в случае появления ошибки, выводит на экран диалоговое окно.

9. Сохраните ВП под именем *Чтение файла.vi*.

Запуск ВП

10. Перейдите на лицевую панель и с помощью инструмента УПРАВЛЕНИЕ выберите кнопку **Browse** (обзор) в элементе управления **Путь к файлу**.
11. Выберите файл *демофайл.txt* и нажмите на кнопку **Open** или **OK**.
12. Запустите ВП. Строка, считанная из файла, отобразится на лицевой панели ВП.
13. При желании выполните часть упражнения повышенной сложности. В противном случае сохраните и закройте ВП.

14. Измените ВП таким образом, чтобы числовые данные распознавались в строке и отображались в цифровом элементе отображения. По окончании сохраните и закройте ВП.

Совет Для поиска первого числового символа используйте функцию **Match Pattern.**

Конец упражнения 9-3

D. Форматирование строк таблицы символов

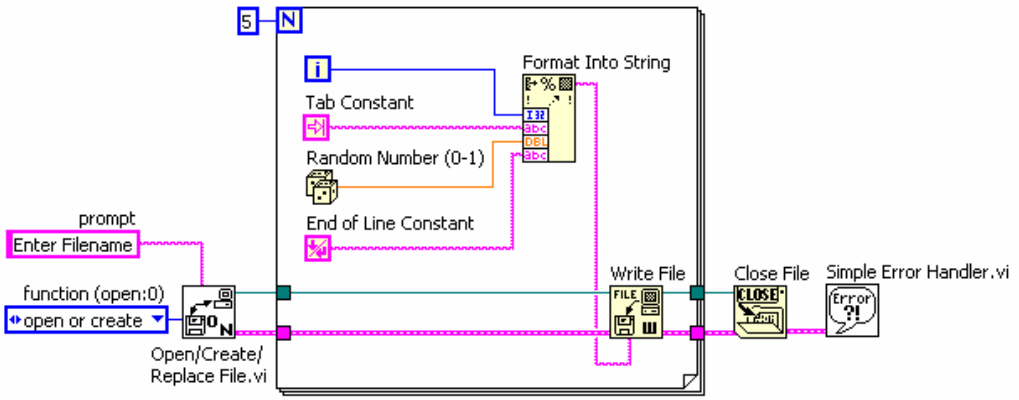
Для того чтобы записать данные в файл формата электронной таблицы, необходимо переформатировать строковые данные в строку таблицы, содержащую разделители, такие как символ табуляции. Во многих приложениях символ табуляции разделяет столбцы, а символ **end of line** (конец строки) разделяет строки.

Примечание Для обеспечения совместимости между различными платформами следует использовать константу **end of line constant**, расположенную в палитре **Functions»Programming»String**. **(Windows)** Константа осуществляет перевод каретки и перевод строки. **(MacOS)** Константа осуществляет перевод каретки. **(UNIX)** Константа осуществляет перевод строки.

Функция **Format Into File** предназначена для форматирования строк, путей к файлам, числовых и логических данных в текст, а также для записи текста в файл. Часто эта функция используется вместо двух операций – форматирования строки с помощью функции **Format Into String** или ВП **Build Text Express VI** и записи результата с помощью функций **Write Characters To File** или **Write File**.

Функция **Format Into File** предназначена для определения порядка, в котором данные записываются в тестовый файл. Однако ее нельзя применять для добавления данных в файл или перезаписи существующего файла. Для этих операций используется функция **Format Into String** совместно с функцией **Write File**. Путь к файлу или ссылку на него можно подать на поле **input file** или оставить это поле без соединения, чтобы указать имя файла в диалоговом окне.

Ниже представлена блок-диаграмма, на которой подпрограмма ВП **Open/Create/Replace File VI** открывает файл. Цикл **For** выполняется пять раз. Функция **Format Into String** преобразует значения счетчика итераций и случайное число в строку. Также указываются символы **Tab constant** (табуляции) и **End of Line Constant** (конца строки) для создания двух столбцов и одной строки таблицы символов. По окончании пяти итераций цикла файл закрывается и ВП проверяет наличие ошибок.



Этот ВП создает следующий текстовый файл, в котором стрелка (→) указывает символ табуляции, а символ параграфа (¶) указывает конец строки:

0→0.798141¶

1→0.659364¶

2→0.581409¶

3→0.526433¶

4→0.171062¶

Можно открыть данный текстовый файл в любом редакторе электронных таблиц для отображения на экране следующей таблицы.

	A	B
1	0	0.798141
2	1	0.659364
3	2	0.581409
4	3	0.526433
5	4	0.171062

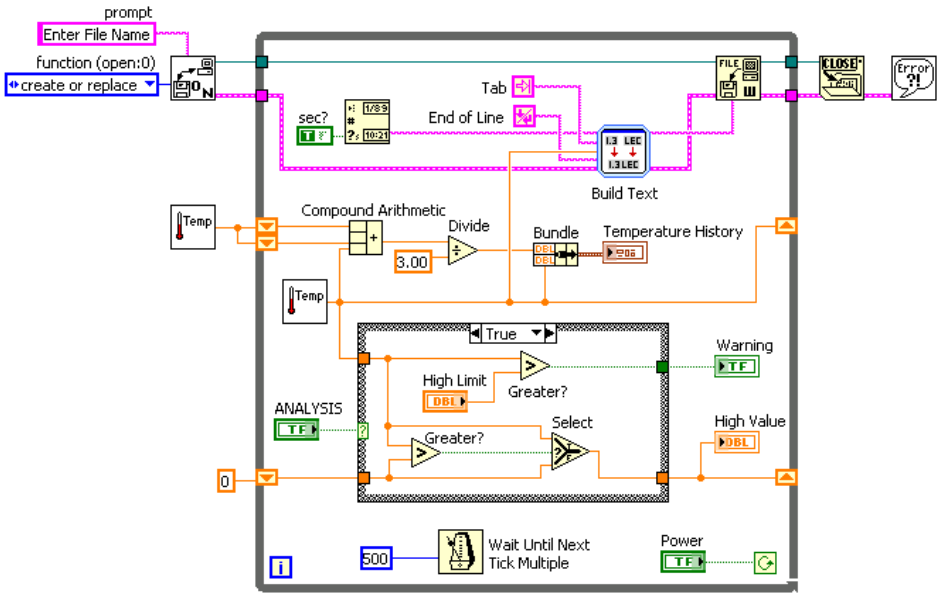
Упражнение 9-4 ВП Регистратор температуры

Цель: Сохранить данные в файл в форме, доступной для редактора электронных таблиц или текстового редактора.

Выполните следующие шаги для сохранения времени и текущего значения температуры в файл.

Блок-диаграмма

1. Откройте ВП *Контроль температуры.vi*, созданный в упражнении 8-2, и сохраните его под именем *Регистратор температуры.vi*. Изменять лицевую панель нет необходимости.
2. Измените блок-диаграмму, как показано ниже:



В палитре **Functions»Programming»File I/O** выберите **Open/Create/Replace File VI**. Этот ВП выводит диалоговое окно для открытия или создания файла.

В палитре **Functions»Programming»Timing** выберите функцию **Get Date/Time String.** Эта функция выдает время проведения измерения температуры в строковом формате.

Щелкните правой кнопкой мыши по терминалу **want seconds?** (нужны секунды?) и в контекстном меню выберите **Create»Constant**. С помощью инструмента УПРАВЛЕНИЕ измените значение логической константы с FALSE на значение TRUE. Это значение константы заставит функцию включить секунды в строку.

Поместите на блок-диаграмму экспресс-ВП **Build Text Express VI**, расположенный в палитре **Functions»Express»Output**. Этот экспресс-ВП преобразует несколько входных строк в одну.

a. При помещении экспресс-ВП **Build Text Express VI** на блок-диаграмму появляется диалоговое окно. Для определения трех переменных введите строку **%tab%temp%end%** в поле **Text with Variables in Percents**. Первая переменная – константа табуляции, вторая – значение температуры, последняя – константа конца строки. Для времени создавать переменную не нужно, оно подается на поле ввода **Beginning Text**.

b. Выберите переменную **temp** в разделе **Configure variables**. Отметьте пункты **Number** и формат дробного числа **Format fractional number**. Переменные **tab** и **end** настраивать не нужно, оставьте их в состоянии по умолчанию.

c. После выполнения настроек нажмите на кнопку **OK**.

d. Для экономии места на блок-диаграмме щелкните правой кнопкой мыши по ВП **Build Text Express VI** и выберите пункт **View as Icon**.

В палитре **Functions»Programming»String** выберите константу **Tab constant** и константу **End of Line constant**.

Поместите на блок-диаграмму функцию **Write File** из палитры **Functions»Programming»File I/O**. Эта функция записывает данные в файл, указанный в ссылке **refnum**.

В палитре **Functions»Programming»File I/O** выберите функцию **Close File.** Эта функция закрывает файл.

В палитре **Functions»Programming»Dialog & User Interface** выберите подпрограмму ВП **Simple Error Handler VI**. Эта подпрограмма проверяет кластер ошибок ВП и при наличии ошибок выводит на экран диалоговое окно.

3. Сохраните ВП, он будет использоваться позднее.

Запуск ВП

4. Перейдите на лицевую панель и запустите ВП. Появится диалоговое окно **Enter Filename**.

5. В диалоговом окне введите имя файла *temp.txt* и нажмите на кнопку **Save** или **OK**.

   ВП создаст файл с именем *temp.txt*. ВП считывает данные каждые полсекунды и сохраняет значения измеренной температуры и время измерения в файл, пока не нажата кнопка **Power**. Когда ВП заканчивает работу, файл закрывается.

6. Откройте текстовый редактор, например, (**Windows)** Notepad или WordPad, (**MacOS)** SimpleText или (**UNIX)** Text Editor.

7. Откройте файл *temp.txt*. Время измерения отображено в первом столбце, а данные температуры – во втором.

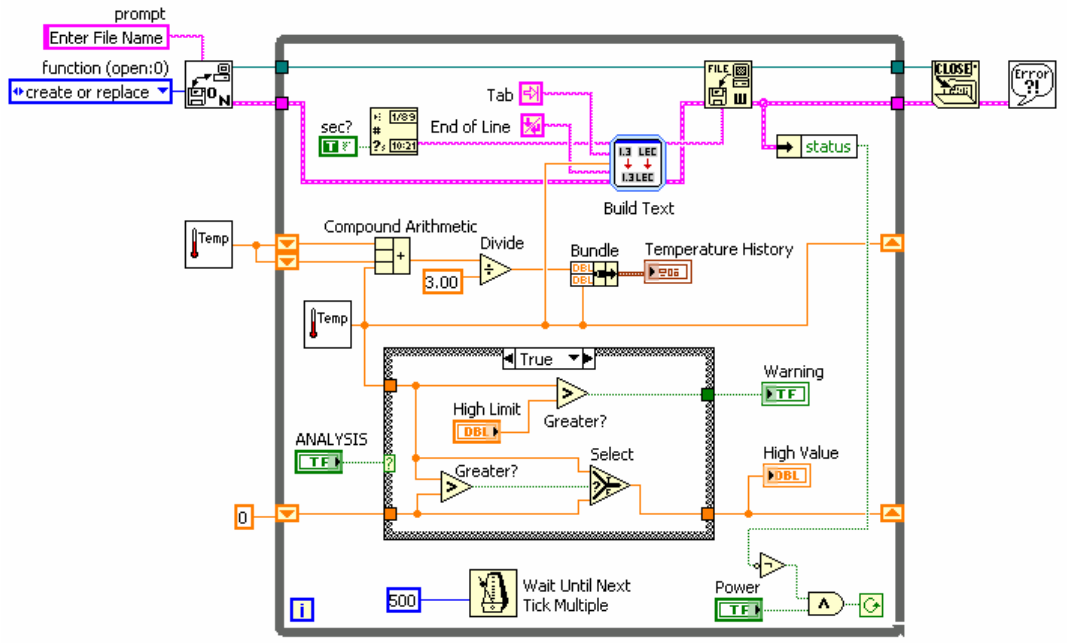
8. Закройте текстовый редактор и вернитесь в LabVIEW.

9. При желании выполните дополнительную часть упражнения. В противном случае закройте ВП.

Дополнительная часть

При использовании системы обработки ошибок в ВП, как только возникает ошибка, цикл **While** должен остановиться. Проделайте следующие шаги для изменения ВП так, чтобы он прекращал работу не только от кнопки **Power**, но и в случае возникновения ошибки.

10. Измените блок-диаграмму, как показано на рисунке.



В палитре **Functions»Programming»Cluster & Variant** выберите функцию **Unbundle by Name.** Эта функция выделяет параметр **status** из кластера ошибок.

В палитре **Functions»Programming»Boolean** выберите функции **Not** и **And**. Функции **Not** и **And** формируют условие выхода из цикла. Пока элемент **Power** имеет значение TRUE**,** и ошибки отсутствуют, ВП продолжает выполняться.

11. Сохраните и запустите ВП.

12. Проверьте обработку ошибок, удалив проводник ссылки на файл между функцией **Write File** и левой границей цикла. Щелкните правой кнопкой мыши по полю ввода **refnum** и выберите пункт **Create»Constant**.

13. Снова запустите ВП. ВП должен запросить путь к файлу, а затем остановиться из-за ошибки. Если бы обработка ошибок не была включена в этот ВП, то сообщение об ошибке появилось бы только после остановки ВП пользователем.

14. При желании выполните часть упражнения повышенной сложности. В противном случае закройте ВП, не сохраняя изменений.

15. Замените ВП **Build Text Express VI** функцией **Format Into String**.
16. Запустите ВП.
17. Закройте ВП, не сохраняя изменений.

Конец упражнения 9-4

E. Функции файлового ввода/вывода высокого уровня и дополнительные функции файлового ввода/вывода

Функции файлового ввода/вывода высокого уровня расположены в верхней строке палитры **Functions»Programming»File I/O**. Они предназначены для выполнения действий файлового ввода или вывода данных следующих типов:

- Символов в/из текстовых файлов.
- Строк из текстовых файлов.
- Одномерных или двумерных массивов числовых данных одинарной точности в/из файла электронной таблицы.
- Одномерных или двумерных массивов числовых данных одинарной точности или целочисленных 16-разрядных в/из бинарного файла.

Функции файлового ввода/вывода высокого уровня включают в себя:

Write to Spreadsheet File – преобразует 2D или 1D массив числовых данных одинарной точности в текстовую строку и записывает строку в новый или добавляет в уже существующий файл. При этом можно также транспонировать данные. ВП открывает или создает файл перед записью и после всех операций закрывает его. Этот ВП используется для создания текстовых файлов, читаемых большинством текстовых редакторов и редакторов электронных таблиц.

Read From Spreadsheet File – считывает определенное число строк от начального смещения **start of read offset** и преобразует данные в 2D массив числовых данных одинарной точности. ВП открывает файл перед чтением и после всех операций закрывает его. Этот ВП можно использовать для чтения таблицы символов, сохраненной в текстовом формате.

Read from Text File – считывает заданное количество символов или строк из файла. ВП открывает файл перед чтением и после всех операций закрывает его.

Write to Text File – записывает заданное количество символов или строк из файла. ВП открывает файл перед чтением и после всех операций закрывает его.

Read from Binary File – читает файл, записанный в бинарном формате. Данные могут быть целочисленного типа или числовыми данными одинарной точности с плавающей точкой.

Write to Binary File – записывает файл в бинарном формате. Данные могут быть целочисленного типа или числовыми данными одинарной точности с плавающей точкой.

New Zip File – Создает новый пустой файл с расширением *.zip.

Add File to Zip – Добавляет файлы в zip файл.

Close Zip File – Закрывает файл с расширением *.zip.

Дополнительные функции работы с файлами (**Advanced File I/O**) расположены в палитре **Functions»Programming»File I/O»Advanced File Functions**.

Get File Position – Возвращает текущее значение положения метки в файле по ссылке на файл.

Get File Size – Возвращает размер файла в байтах.

Set File Position - Устанавливает значение положения метки в файле по ссылке на файл.

Set File Size - Устанавливает размер файла в байтах.

Get Type and Creator – Возвращает тип файла поданного на вход ВП и название программы в которой файл создан. В случае, если ВП не удалось распознать файл, то он возвращает ????.

Set Type and Creator - Устанавливает тип файла поданного на вход ВП и название программы в которой файл создан.

Move – Перемещает файл или папку из одного места на диске в другое.

Copy - Копирует файл или папку из одного места на диске в другое.

Delete - Удаляет файл или папку на диске.

File/Directory Info – Возвращает информацию о файле или папке, поданной на вход, включая размер файла или папки, дату последнего изменения и т.п.

Create Folder – Программно создает папку. Если файл или папка с подобным именем уже существует на диске, то ВП возвращает ошибку.

Get Volume Info – Возвращает информацию об объеме свободного и занятого места на диске.

List Folder – Возвращает два массива строк перечисляющих все файлы и папки, обнаруженные по указанному пути.

Check if File or Folder Exists VI – Проверяет существует ли файл или папка на диске по указанному пути.

Compare Two Paths VI – сравнивает между собой два пути и возвращает общую часть пути, различающуюся часть пути и переменную логического типа, говорящую одинаковые пути или нет.

Generate Temporary File Path VI – генерирует путь к файлу с расширением .tmp. Данный виртуальный прибор только создает путь, но не создает файл.

Get File Extension VI – возвращает расширение файла, поданного на вход.

MD5Checksum File VI – Вычисляет для файла MD5 message-digest. MD5 message-digest это 128 битное число, отображающееся прописными символами в шестнадцатеричном формате.

Recursive File List VI – Возвращает список содержимого папки или библиотеки LLB.

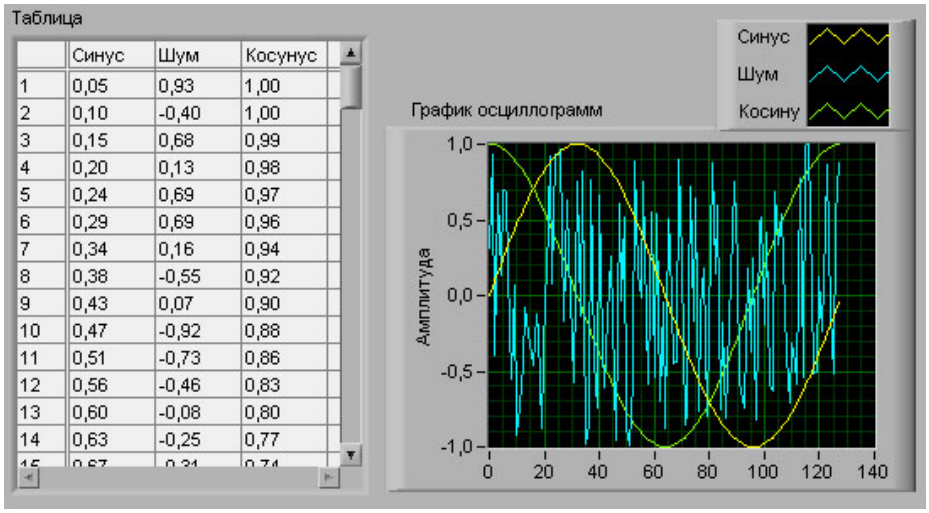
Упражнение 9-5 ВП Запись таблицы символов

Цель: Сохранить 2D массив в текстовый файл в виде таблицы так, чтобы он был доступен редактору электронной таблицы.

В упражнении 9-4 строка была отформатирована таким образом, что позиция табулятора разделял столбцы, а символ конца строки разделял строки. Выполните следующие шаги для изучения ВП, который сохраняет числовой массив в файл в формате, доступном для редактора электронных таблиц.

Лицевая панель

1. Откройте ВП *Запись таблицы символов.vi* из папки *C:\Exercises\LV Basics I*. Лицевая панель уже создана.



Запуск ВП

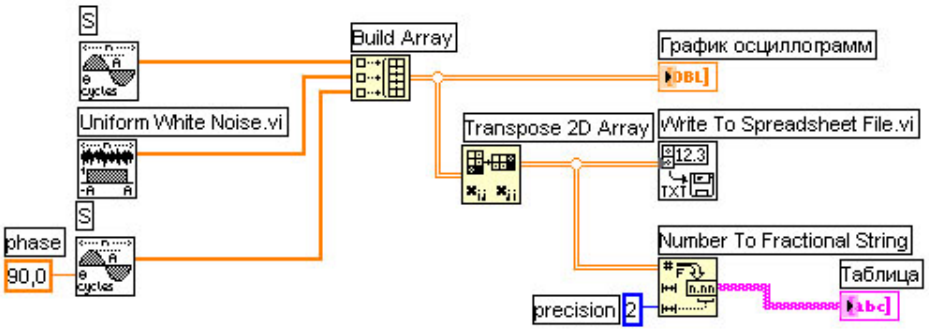
2. Запустите ВП.

   ВП генерирует 2D массив из 128 строк и 3 столбцов. Первый столбец содержит данные осциллограммы синуса, второй – осциллограммы шума, третий столбец – данные осциллограммы косинуса. ВП выводит осциллограммы каждого столбца и отображает данные в виде таблицы. После вывода данных на экран появляется диалоговое окно.

3. Введите имя файла *осциллограмма.txt* и нажмите на кнопку **Save** или **OK**. Позднее этот файл будет использоваться.

Блок-диаграмма

4. Перейдите на блок-диаграмму:



- ВП **Sine Pattern VI,** расположенный в палитре **Functions»Signal Processing»Signal Generation** выдает числовой массив из 128 элементов, содержащий значения синусоидального сигнала. Константа 90,0 определяет фазу гармонического сигнала.

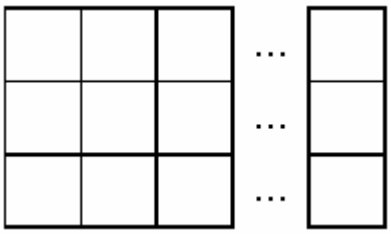
- ВП **Uniform White Noise VI**, расположенный в палитре **Functions»Signal Processing»Signal Generation** выдает числовой массив из 128 элементов, содержащий значения шумового сигнала.

- Функция **Build Array**, расположенная в палитре **Functions»Programming»Array,** компонует 2D массив из массивов синуса, шума и косинуса.

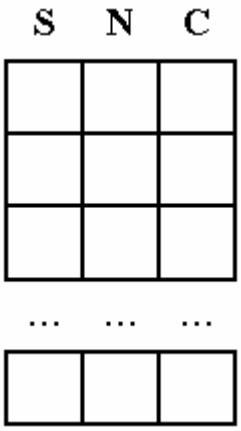
Массив синуса

Массив шума

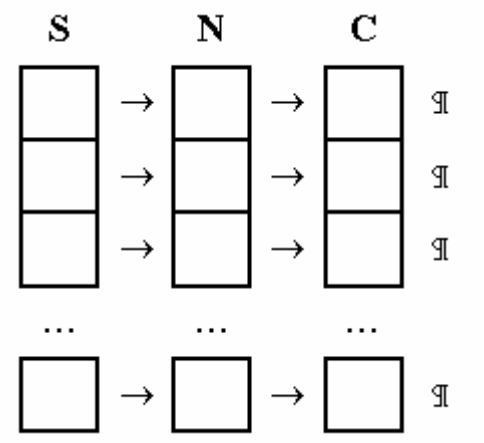
Массив косинуса



- Функция **Transpose 2D Array**, расположенная в палитре **Functions»Programming»Array**, перестраивает элементы 2D массива так, что элемент [i,j] становится элементом [j,i], как показано ниже:



- ВП **Write To Spreadsheet File VI,** расположенный в палитре **Functions»Programming»File I/O**, форматирует 2D массив в строку таблицы символов и записывает строку в файл. Формат строки показан на рисунке, где стрелка → показывает позицию табулятора, а символ параграфа ¶ показывает окончание строки символов.



- Функция **Number To Fractional String**, расположенная в палитре **Functions»Programming»String»String/Number Conversion**, преобразует массив чисел в массив строк, которые отображаются в таблице.

5. Закройте ВП.

Примечание В этом примере сохраняются в файл только три массива. Для добавления большего количества массивов следует увеличить количество полей ввода данных функции **Build Array**.

Дополнительная часть

Откройте файл текстовым редактором или редактором электронных таблиц и просмотрите его содержимое.

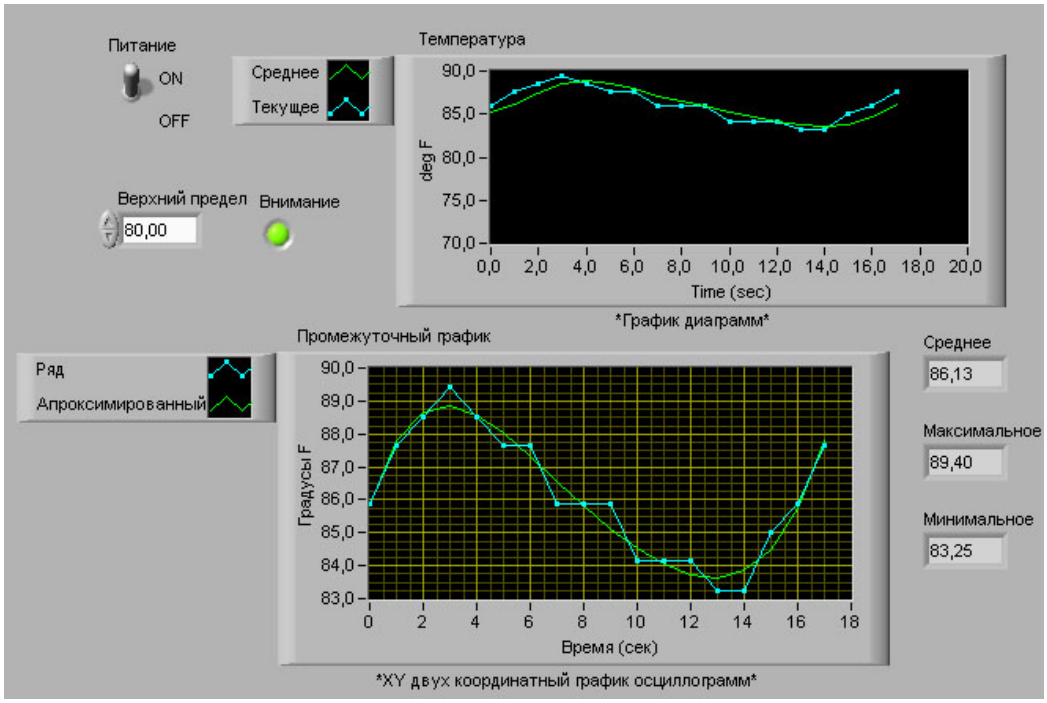
6. Откройте текстовый редактор, такой как в **(Windows)** Notepad или WordPad, **(MacOS)** SimpleText, **(UNIX)** Text Editor.
7. Откройте файл *осциллограмма.txt*. Данные синуса представлены в первом столбце, шум – во втором, косинус – в третьем.
8. Выйдите из текстового редактора или редактора электронных таблиц и вернитесь в LabVIEW.

Конец упражнения 9-5

Упражнение 9-6 ВП Приложение анализа температуры

Цель: Применить все знания, полученные в этом курсе: структуры, сдвиговые регистры, графики диаграмм, массивы, графики, подпрограммы работы с файлами и так далее.

Создайте ВП, выполняющий перечисленные ниже действия. Создайте лицевую панель, как показано ниже, и сохраните ВП под именем *Приложение анализа температуры.vi*.



1. Необходимо производить измерение температуры с частотой 1 Гц до момента остановки выполнения ВП переключателем или появления ошибки.
2. На график Диаграмм необходимо одновременно выводить текущее значение температуры и среднее значение последних трех измерений.
3. Если температура превысит верхний допустимый предел, необходимо зажечь светодиод.
4. После каждого измерения необходимо регистрировать дату, время, включая секунды, температуру, среднее значение последних измерений и одно текстовое сообщение, описывающее, была ли температура в норме или превышала допустимый предел. Данные необходимо регистрировать таким образом, чтобы каждый элемент отображался в отдельном столбце таблицы, как показано ниже.

	A	B	C	D	E
1	Date	Time	Temp	Avg	Comment
2	9/26/00	12:45:17 AM	74.46	74.46	Normal
3	9/26/00	12:45:18 AM	74.46	74.46	Normal
4	9/26/00	12:45:19 AM	74.46	74.46	Normal
5	9/26/00	12:45:20 AM	74.46	74.46	Normal

5. После прекращения измерений необходимо отобразить на двухкоординатном графике данные измерений температуры и кривую аппроксимации. Выведите на экран среднее значение температуры, ее максимальное и минимальное значения.

Совет. Начните работу с модернизации **ВП Регистратор температуры**, созданного в упражнении 9-4. Для выполнения пункта 5 используйте часть **ВП Анализ Температуры**, созданного в упражнении 7-4.

Конец упражнения 9-6

Краткое изложение пройденного материала, советы и секреты

- Строки являются последовательностями ASCII кодов. Для создания окон ввода текста и меток используются строковые элементы управления и отображения данных, расположенные в палитре **Controls»Modern»String & Path.**
- С помощью полосы прокрутки можно минимизировать пространство, занимаемое строковым объектом. Для этого необходимо щелкнуть правой кнопкой мыши по объекту и выбрать в контекстном меню пункт **Visible Items»Scrollbar**.
- Для редактирования и управления строками на блок-диаграмме используются функции обработки строк, расположенные в палитре **Functions»Programming»String**.
- Для преобразования числовых данных в строки используются функция **Format Into String** и экспресс-ВП **Build Text Express VI**.
- Для преобразования строк в числовые данные используется функция **Scan From String**.
- Для создания или редактирования формата строки необходимо щелкнуть правой кнопкой мыши по функции **Format Into String** или **Scan From String** и выбрать пункт контекстного меню **Edit Format String** или **Edit Scan String**.
- Для проведения операций над файлами используются функции файлового ввода/вывода, расположенные в палитре **Functions»Programming»File I/O**.
- Функции файлового ввода/вывода высокого уровня расположены в верхней строке палитры **Functions»Programming»File I/O.** Они предназначены для выполнения стандартных действий по файловому вводу/выводу данных.
- Функции файлового ввода-вывода низкого уровня расположены в средней строке палитры **Functions»Programming»File I/O**. Дополнительные функции работы с файлами расположены в палитре **Functions»Programming»File I/O»Advanced File Functions** и предназначены для управления отдельными операциями над файлами.
- При осуществлении записи в файл производятся следующие операции: открытие файла, создание или перезапись файла, запись данных и закрытие файла. Аналогичные операции производятся и при чтении файла: открытие существующего файла, считывание данных и закрытие файлов.

- Для получения доступа к файлу с помощью диалогового окна поле **file path** подпрограммы ВП **Open/Create/Replace File VI** следует оставить неподключенным.

- Чтобы записать таблицу символов в файл, следует отформатировать строку в строку таблицы символов, которая содержит разделители, такие как табуляция. Для форматирования строки, числовых, логических данных и путей в текстовый файл используется функция **Format Into File**.

Дополнительные упражнения

9-7. Постройте ВП, который генерирует 2D массив, состоящий из трех строк и 100 столбцов случайных чисел, помещает данные в таблицу символов и записывает данные таблицы в файл. Добавьте заголовок каждому столбцу. Используйте функции файлового ввода/вывода высокого уровня, расположенные в верхней строке палитры **Functions»Programming»File I/O**.

Совет Для записи заголовка и числовых данных в тот же файл следует использовать подпрограммы ВП **Write to Text File VI** и **Write To Spreadsheet File VI** соответственно.

Сохраните ВП под именем *Запись осциллограмм в таблицу.vi*.

9-8. Постройте ВП, который преобразует строки таблицы символов, разграниченные позицией табуляции, в строки с разделителями - запятыми. Причем столбцы должны быть разделены запятыми, а строки - символами **end of line**. Строки таблицы символов, разграниченные позицией табуляции, и строки с разделителями-запятыми необходимо отобразить на лицевой панели.

Совет Следует использовать функцию **Search and Replace String**.

Сохраните ВП под именем *Преобразование таблицы.vi*.

9-9 Измените **ВП Регистратор температуры**, созданный в упражнении 9-4, так, чтобы ВП не создавал файл каждый раз при запуске. Добавьте данные в конец уже существующего файла *temp.dat*. Запустите ВП несколько раз и с помощью текстового редактора убедитесь, что ВП добавляет в файл новые значения.

Совет Следует удалить функцию **Format Into File**, заменив ее функциями **Format Into String** и **Write File**. Для перемещения текущей метки файла, следует использовать параметры **pos mode** и **pos offset** функции **Write File.**

Сохраните ВП под именем *Регистратор температуры 2.vi*.

9-10 Создайте ВП, осуществляющий поиск текстового файла с заданным текстом в указанной папке. При отсутствии файла выдается сообщение и создается файл с названием «file DD-MM-YY.txt», содержащим текущую дату.

Совет Для получения даты используйте ВП **Get Date/Time String**, находящуюся в палитре **Functions»Programming»Timing**.

Сохраните ВП под именем *Поиск файла с текстом.vi*.

9-11 Создайте ВП, считывающий из файла числовые данные посимвольно и преобразующий их в значения типа Double (разделителем полей считать пробел).

Сохраните ВП под именем *Считывание чисел из текстового файла.vi*.

9-12 Создайте ВП, анализирующий произвольную строку и разбивающий её на слова (разделитель вводится отдельно). Каждое слово преобразуется в массив ASCII кодов. Результат представьте виде массива строк и двумерного массива ASCII кодов на лицевой панели.

Сохраните ВП под именем *Преобразование слов из файла в ASCII коды.vi*.

Примечания

Урок 10.
Сбор и отображение данных

На этом уроке рассматривается использование встроенных **Устройств сбора данных (DAQ)** в LabVIEW. Для получения дополнительной информации о сборе данных в среде LabVIEW используйте руководство **Measurements Manual**.

В этом уроке изложены вопросы:

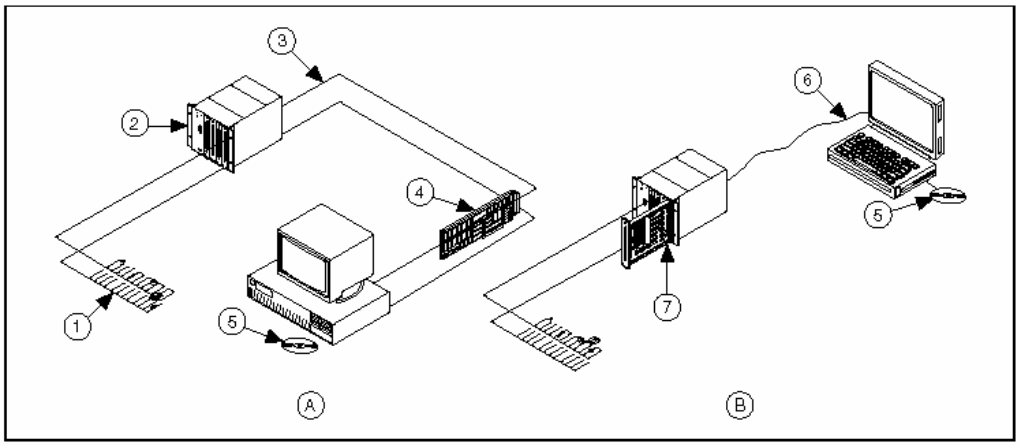
A. Введение и конфигурация.
B. Сбор данных в LabVIEW.
C. Выполнение операций аналогового ввода.
D. Запись полученных данных в файл.
E. Выполнение операций аналогового вывода.
F. Информация о счетчиках.
G. Информация о цифровых линиях ввода-вывода.

A. Введение и конфигурация

Среда LabVIEW включает в себя набор подпрограмм ВП, позволяющих конфигурировать, собирать и посылать данные на **DAQ**-устройства. Часто **DAQ**-устройства могут выполнять разнообразные функции: **аналого-цифровое преобразование (A/D)**, **цифро-аналоговое преобразование (D/A), цифровой ввод/вывод (I/O)** и управление счетчиком/таймером. Каждое устройство имеет свой набор возможностей и скорость обработки данных. Кроме этого, **DAQ**-устройства разрабатываются с учетом аппаратной специфики платформ и операционных систем. Для получения дополнительной информации о **DAQ**-устройствах используйте документ National Instruments Product Catalog на web-сайте **ni.com/catalog.**

Компоненты DAQ-системы

На иллюстрации продемонстрированы два варианта компоновки **DAQ**-системы. В варианте «A» **DAQ**-устройство встроено в компьютер, а в варианте «B» **DAQ**-устройство является внешним. С внешним устройством можно построить DAQ-систему на базе компьютера без доступных слотов расширения, например, с использованием портативных компьютеров. Компьютер и DAQ-модуль связываются между собой через аппаратные интерфейсы, такие как параллельный порт, последовательный порт и сетевые карты (Ethernet). Практически эта система является примером удаленного управления **DAQ**-устройством.



1. Датчики
2. Модуль согласования сигналов
3. Согласованные сигналы
4. Встроенное DAQ-устройство
5. Программное обеспечение
6. Связь с параллельным портом
7. Внешний DAQ-модуль

Основной задачей, решаемой **DAQ**-системами, является задача измерения или генерации физических сигналов в реальном времени. Перед тем как компьютерная система измерит физический сигнал, датчик или усилитель должен преобразовать физический сигнал в электрический, например, ток или напряжение. Встроенное **DAQ**-устройство часто рассматривается как полная **DAQ**-система, хотя практически это только один из компонент системы. В отличие от самостоятельных устройств измерения, не всегда возможно соединение напрямую источника сигналов с встроенным **DAQ**-устройством. В этих случаях необходимо использовать дополнительные

модули согласования сигналов перед тем как **DAQ**-устройство преобразует их в цифровой формат. Программные средства **DAQ**-систем включают в себя: сбор данных, анализ данных и представление результатов.
DAQ-устройства производства компании NI поставляются в комплекте с драйверами NI-DAQ. NI-DAQ взаимодействует и управляет измерительными устройствами National Instruments, включая такие DAQ-устройства как многофункциональные устройства ввода-вывода сигналов (MIO) серии E, SCXI модули согласования сигналов и модули переключения сигналов. NI-DAQ является расширенной библиотекой функций, которые можно вызывать из среды создания приложений, например, LabVIEW, для программирования всех возможностей измерительного устройства NI.

Программирование измерительного устройства NI возможно как в программных пакетах National Instruments: LabVIEW, LabWindows/CVI и Measurement Studio, так и в любой среде программирования, поддерживающей вызовы динамических библиотек (DDL) с использованием ANSI C интерфейса. Использование любого программного обеспечения NI существенно уменьшает время, затраченное на создание приложений сбора данных:

- LabVIEW обеспечивает сбор данных с помощью LabVIEW DAQ - комплекса виртуальных приборов для программирования измерительных устройств NI.
- LabWindows/CVI имеет полную встроенную поддержку ANSI C окружения, программирование измерительных устройств NI производится с помощью библиотеки сбора данных LabWindows/CVI Data Acquisition library.
- Инструменты программирования Measurement Studio предназначены для создания тестовых программ и приложений сбора данных в среде Mirosoft Visual Studio .NET. Measurement Studio имеет поддержку Visual C#, Visual Basic .NET и Visual C++ .NET

Комплекс разработки приложений сбора данных состоит из среды программирования, MAX и NI-DAQ. MAX является высокоуровневым приложением, которое используется для тестирования и настройки DAQ-устройств. NI-DAQ состоит из следующих программных интерфейсов:

- стандартный NI-DAQ
- NI-DAQmx
- NI-SWITCH

Стандартный NI-DAQ

Стандартный NI-DAQ является обновлением предыдущей версии 6.9.x NI-DAQ. **Стандартный NI-DAQ** включает в себя те же ВП/функции и работает аналогично NI-DAQ версии 6.9.x, однако внесенные изменения позволяют использовать NI-DAQ и NI-DAQmx совместно в разрабатываемом приложении. В **Стандартном NI-DAQ** исключена поддержка некоторых измерительных устройств по сравнению с версией 6.9.x. Список устройств, поддерживаемых **Стандартным NI-DAQ**, приведен в документации NI-DAQ.

NI-DAQmx

NI-DAQmx является следующим поколением драйверов NI-DAQ. Он обладает новыми функциями и инструментами управления измерительными устройствами. **NI-DAQmx** имеет много новых особенностей и преимуществ по сравнению с предыдущей версией NI-DAQ:

- DAQ Configuration Assistant - помощник настройки DAQ-устройств с помощью нового графического интерфейса позволяет конфигурировать настройки, каналы и задания измерения DAQ-устройств в LabVIEW, LabWindows/CVI и Measurement Studio.
- Увеличилось быстродействие ряда операций, в частности однократного аналогового ввода-вывода, более эффективно организована многозадачность.
- Программный интерфейс для создания DAQ-приложений стал более простым и интуитивно понятным.
- Появились дополнительные возможности программного интерфейса **NI-DAQmx** для LabVIEW, включая узлы Атрибутов для сбора данных и улучшенную поддержку типов данных waveform для операций аналогового и цифрового ввода-вывода.
- Разработаны единые программные интерфейсы и функциональность для ANSI C, LabWindows/CVI и Measurement Studio, включая интерфейсы .NET и C++.

NI-SWITCH

NI-SWITCH является IVI-совместимым драйвером устройств, поддерживающим все модули переключения сигналов компании NI. NI-SWITCH имеет интерактивную программную лицевую панель, которую можно использовать для поиска неисправности приложений.

В этом курсе описан программный интерфейс NI-DAQmx.

Настройка аппаратных средств DAQ-устройств

Перед использованием ВП **Сбора Данных** необходимо настроить **DAQ**-устройство, установленное в компьютер.

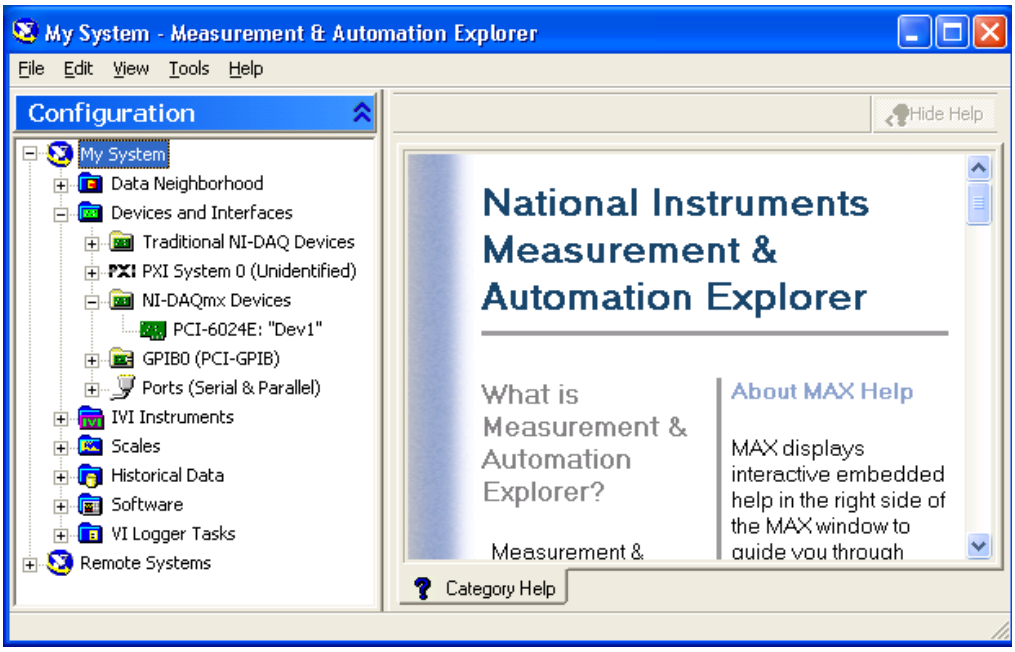
Windows

Windows Configuration Manager хранит информацию обо всем установленном на компьютер аппаратном обеспечении, включая устройства National Instruments DAQ. Если у Вас есть Plug & Play (PnP) устройства, например, карта MIO серии E, Windows Configuration Manager автоматически его определит и настроит. Если устройство не поддерживает PnP, необходимо его настроить вручную с помощью раздела Add New Hardware (Установка и удаление устройств) в Control Panel (Панели Управления).

Аппаратную конфигурацию Windows можно проверить с помощью Device Manager (Менеджера Устройств), который находится в **Start»Settings»Control Panel»System»Device Manager** (Пуск»Настройки»Панель Управления»Система»Диспетчер Устройств).
В разделе **Data Acquisition Devices** отображаются все DAQ-устройства, установленные на компьютер. Дважды щелкните мышью по устройству для отображения диалогового окна с закладками. На закладке **General** (Общие) приводится общая информация об устройстве. На закладке **Resources** (Ресурсы) перечислены системные ресурсы, используемые устройством: номера прерываний (**IRQ**), каналы прямого доступа к памяти (**DMA**) и базовые адреса ввода-вывода для программно конфигурируемых устройств. На закладке **NI-DAQ Information** указан тип шины данного DAQ-устройства. Производитель и номер версии драйвера DAQ-устройства приведены на закладке **Driver** (Драйвер).

Среда LabVIEW устанавливает утилиту конфигурации **Measurement & Automation Explorer** для детальной настройки параметров конфигурации каналов устройств. Эту утилиту необходимо запускать после установки **DAQ**-устройства на компьютер. Утилита конфигурации считывает информацию из реестра Windows, записанную **Диспетчером устройств (Device Manager),** и присваивает логическое имя для каждого **DAQ**-устройства. По логическому имени среда LabVIEW распознает **DAQ**-устройство. Запуск конфигурационной утилиты происходит двойным щелчком левой кнопки мыши по ее иконке на рабочем столе операционной системы или выбором пункта главного меню **Tools»Measurement & Automation Explorer** непосредственно в среде

LabVIEW. Начальное окно конфигурационной утилиты показано ниже на рисунке. **Measurement & Automation Explorer** также используется для конфигурации устройств стандарта **SCXI**.



Конфигурационная утилита определяет все аппаратные средства фирмы National Instruments, включая **GPIB** интерфейс. Дополнительная информация о **GPIB** интерфейсе приведена в Уроке 11 *Управление измерительными приборами.*

Параметры устройства можно также установить с помощью утилит-конфигураций, входящих в комплект поставки устройств. **Measurement & Automation Explorer** позволяет сохранить логическое имя устройства и параметры конфигураций в реестр Windows.

Windows автоматически находит и настраивает **DAQ**-устройства, удовлетворяющие стандарту **PnP**, например, карту **PCI-6024E**.

Настройка каналов и заданий

В Стандартном NI-DAQ можно создавать *виртуальные каналы*, которые включают в себя совокупность настроек физического канала DAQ, типа измерений и информацию о нормировке значений. В Стандартном NI-DAQ и NI-DAQ более ранних версий понятие виртуальных каналов используется в качестве соответствия физических каналов проводимым измерениям. Понятие *NI-DAQmx channels* аналогично виртуальным каналам Стандартного NI-DAQ.

NI-DAQmx также включает в себя *заданий*, являющихся частью программного интерфейса API. *Задание* - это совокупность настроек одного или нескольких каналов, синхронизации, временных и других параметров. *Задание* описывает измерение или генерацию сигналов, которые необходимо выполнить. Каналы, созданные в рамках задания, являются локальными. Каналы, определенные вне заданий, являются глобальными, и могут быть использованы отдельно. Настройка виртуальных каналов является необязательной в Стандартном NI-DAQ и

более ранних версий, но в NI-DAQmx выполнение этой процедуры необходимо для проведения измерений. В Стандартном NI-DAQ конфигурирование виртуальных каналов производилось в MAX. В NI-DAQmx настройка виртуальных каналов возможна как в MAX, так и в самом приложении, причем как в рамках *задания*, так и отдельно.

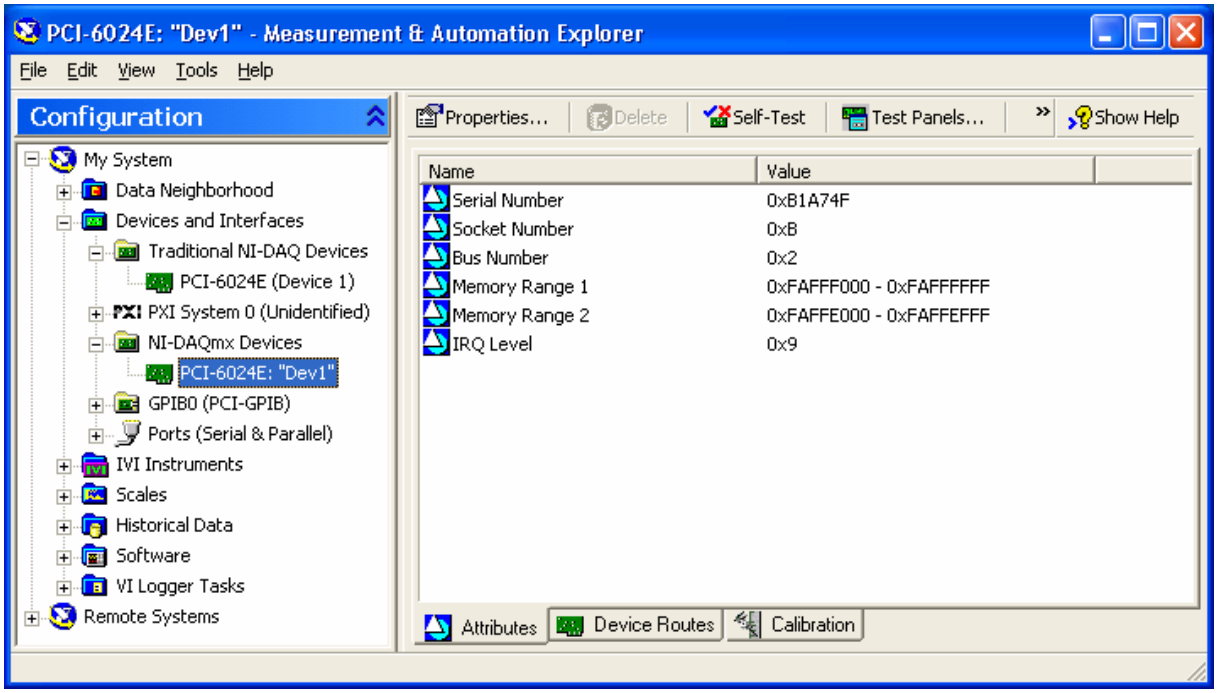
Упражнение 10-1. MAX (только для Windows)

Цель: Использование конфигурационной утилиты Measurement & Automation Explorer для проверки текущих установок DAQ-устройства, его тестирования и создания двух заданий.

Выполните следующие шаги по использованию конфигурационной утилиты **Measurement & Automation Explorer** для проверки конфигурации **DAQ**-устройства, установленного на компьютере, и тестирования возможностей устройства, а также установки двух заданий для работы с сигнальной панелью DAQ.

Часть A. Проверка настроек DAQ-устройства

1) Двойным щелчком левой кнопки мыши по иконке на рабочем столе или выбором пункта главного меню **Tools»Measurement & Automation Explore**r в среде LabVIEW запустите конфигурационную утилиту. Утилита производит поиск в системе установленных аппаратных средств фирмы National Instruments и отображает найденную информацию.
2) Откройте секцию **Devices and Interfaces** (Устройства и Интерфейсы) для просмотра установленных устройств фирмы National Instruments. В примере, показанном на рисунке, обнаружены устройства **PCI-6024E** и **PCI-GPIB**.

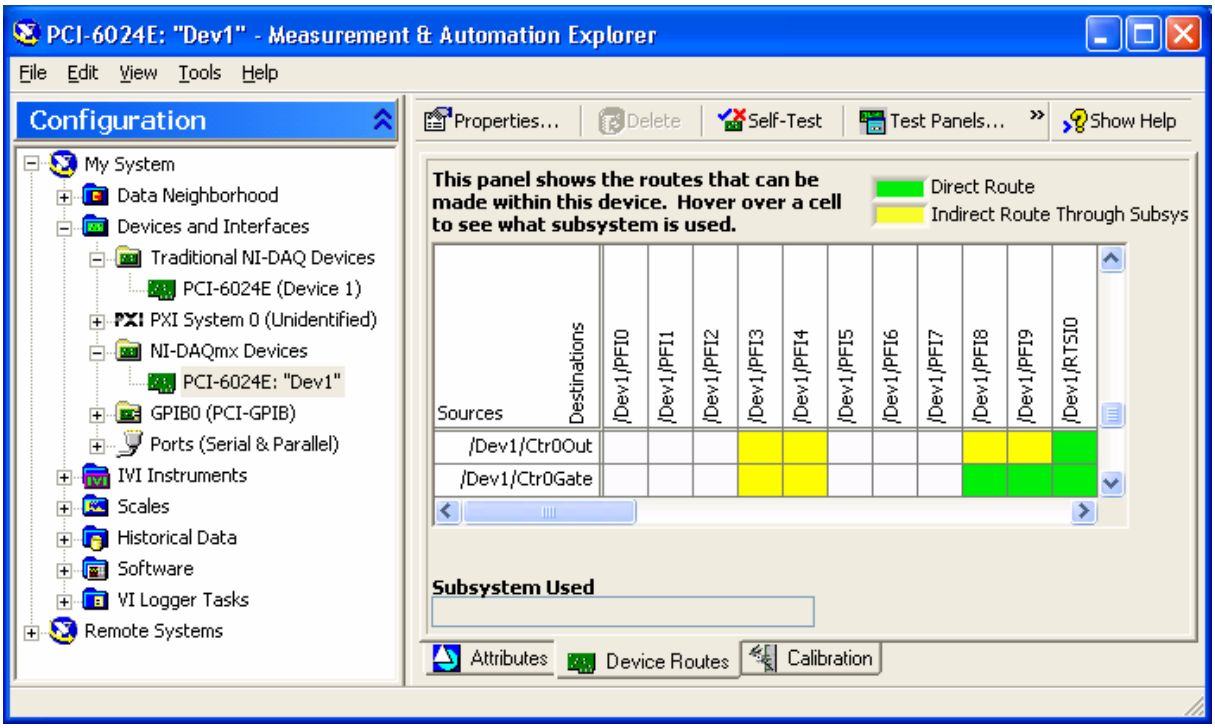


Помимо аппаратных средств, конфигурационная утилита отображает программные средства, установленные в системе. Номер устройства отображен после его названия. ВП Сбора Данных использует номер устройства для определения, какое устройство выполняет операцию

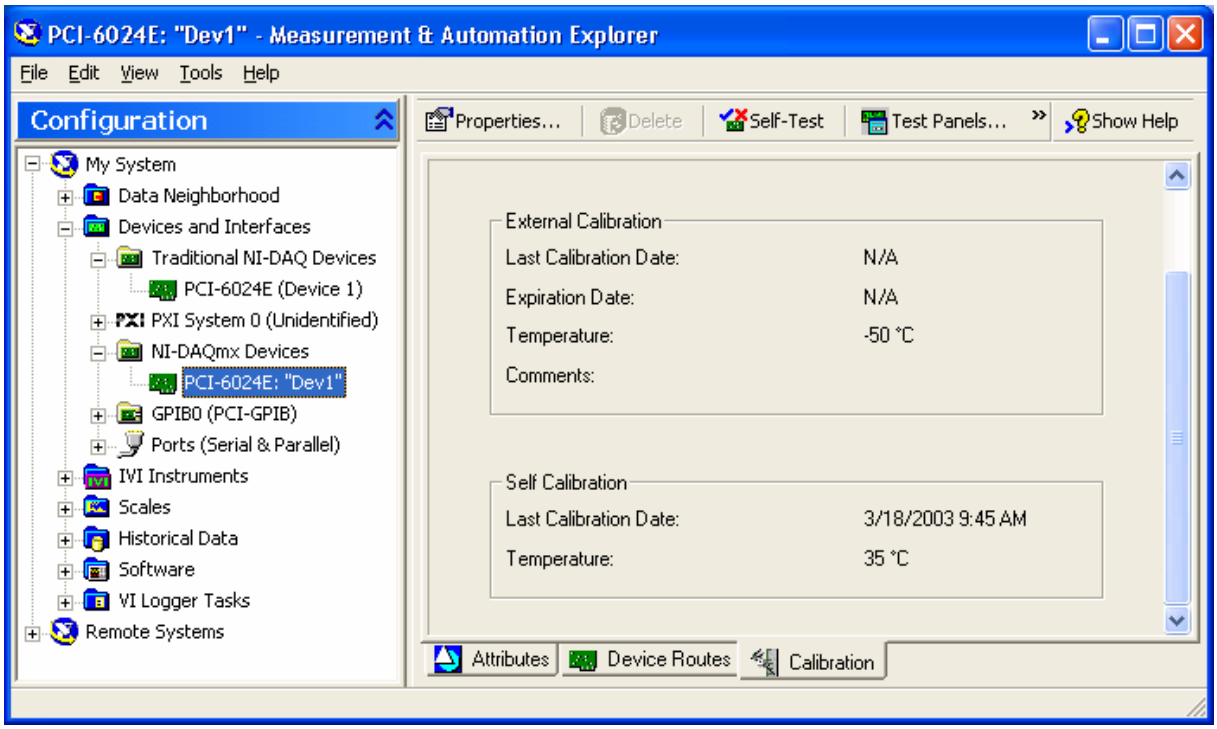
сбора данных. MAX также отображает параметры устройства, например, используемые системные ресурсы.

Примечание. Возможно у вас установлены другие DAQ-устройства, и некоторые настройки могут различными. Нажмите на кнопку **Show/Hide** в верхнем правом углу окна конфигурационной утилиты для скрытия контекстной подсказки и показа информации о **DAQ**-устройствах.

3) На закладке **Device Routes** (Разводка сигналов устройства) приведена подробная информация о внутренних сигналах устройства, которые могут быть различным образом перенаправлены. Эта панель является мощным инструментом, позволяющим осуществлять синхронизацию внутренних компонент и внешних устройств.



4) На закладке **Calibration** (Калибровка) указаны даты проведения внешней и внутренней калибровки устройства.



5) Щелкните правой кнопкой мыши по устройству в разделе NI-DAQmx и выберите пункт **Self Calibrate** (Внутренняя калибровка) для осуществления калибровки DAQ-устройства с помощью высокоточного источника напряжения и обновления калибровочных констант. По окончанию калибровки информация о ней появится на закладке **Calibration** в разделе **Self Calibration**.

Часть B. Тестирование компонент DAQ-устройства

6) Нажмите на кнопку **Self-Test** для проведения тестирования устройства. Проводится тестирование системных ресурсов, используемых устройством. Устройство должно успешно пройти все тесты, поскольку оно уже настроено.

7) Нажмите на кнопку **Test Panel** (Тестовая панель) для тестирования отдельных функций DAQ-устройства, таких как аналоговый вход и выход. На экране появится диалоговое окно **Test Panel** (Тестовая панель).

   a) Используйте закладку **Analog Input** (Аналоговый Вход) для тестирования различных каналов аналогового входа DAQ-устройства. Канал Dev1/ai0 подключен к датчику температуры сигнальной панели. Нажмите на кнопку **Start** для начала сбора данных. Поместите палец на датчик температуры, в результате напряжение возрастает. Измените положение переключателя **Noise** (Шум) на **On** (Включено) и проследите за изменением сигнала на закладке. По завершению нажмите на кнопку **Stop**.

   b) Перейдите на закладку **Analog Output** (Аналоговый выход) для установки значения постоянного напряжения или гармонического сигнала на одном из аналоговых выходов DAQ-устройства. Установите **Output Mode** (Режим генерации) в положение **Sine**

Generator (Синусоидальный сигнал) и нажмите на кнопку **Start Sine Generator** (Генерация гармонического сигнала). LabVIEW генерирует непрерывный синусоидальный сигнал на аналоговом выходе канала «0».

c) На внешней соединительной сигнальной панели DAQ-устройства подключите Аналоговый выход канала «0» к Аналоговому входу канала «1».

d) Откройте закладку **Analog Input** (Аналоговый вход) и выберите канал Dev1/ai1. Нажмите на кнопку **Start** для начала сбора данных со входного канала «1». На графике отобразится синусоидальный сигнал, который генерируется на выходе канала «0».

e) Перейдите на закладку **Digital I/O** (Цифровой ввод/вывод) для проверки работы цифровых линий **DAQ**-устройства.

f) Настройте цифровые каналы с «0» по «3» в качестве каналов вывода и отметьте или уберите выделение с пунктов **Logic Level** (Логический уровень). При этом погаснут или загорятся соответствующие светодиоды на сигнальной панели. Светодиоды работают по принципу отрицательной логики: выключенный светодиод соответствует положительному уровню цифрового канала.

g) Нажмите на кнопку **Close** для закрытия тестовой панели и возврата в главное окно **MAX**.

8) Перейдите на закладку **Counter I/O** (Счетчики ввода-вывода) для проверки работы счетчиков и таймеров DAQ-устройства. Для проверки операций счетчик/таймер перевести режим **Counter Mode** (Режим Счетчика) в положение **Simple Event Counting** (Простой подсчет событий) и нажать на кнопку **Start.** Значение счетчика начнет быстро возрастать. Нажмите на кнопку **Stop** для остановки теста.

9) Закройте MAX, выбрав в меню пункт **File»Exit**.

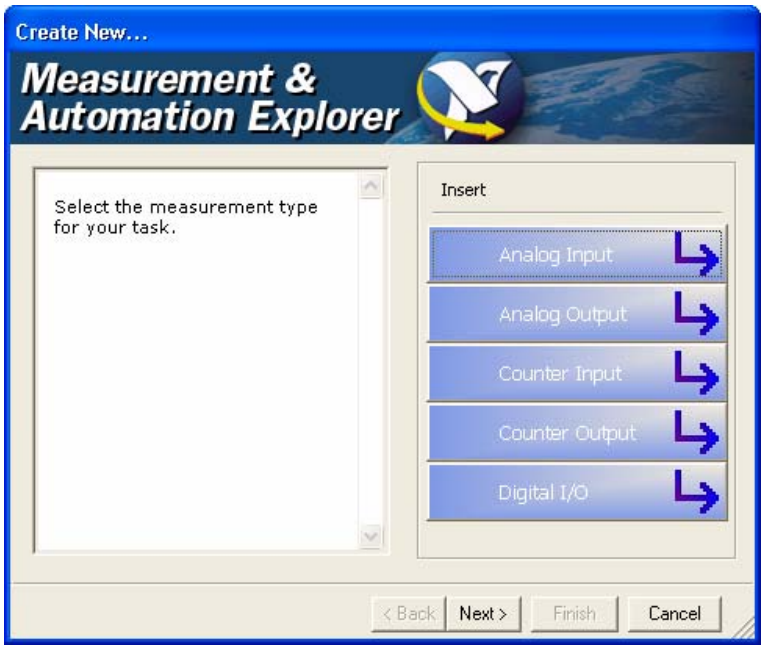
Конец упражнения 10-1

B. Сбор данных в LabVIEW

Подпрограммы сбора данных размещены в палитрах **Functions» Measurements I/O»Data Acquisition** и **Functions»Measurements I/O»DAQmx - Data Acqusition.** В разделе **Data Acquisition** собраны стандартные виртуальные приборы сбора данных, а в **DAQmx - Data Acqusition -** ВП для работы с NI-DAQmx.

В палитре **DAQmx - Data Acqusition** содержатся все необходимые подпрограммы для осуществления операций аналогового и цифрового ввода-вывода и работы со счетчиками/таймерами. Виртуальные приборы собраны таким образом, что большинство задач могут быть решены с их использованием. Можно настроить ВП для выполнения специфического действия с помощью узла Атрибутов. Многие задания, не требующие расширенных возможностей синхронизации, могут быть выполнены с помощью экспресс-ВП **DAQmx Assistant**. В этом курсе описано, как использовать экспресс-ВП **DAQmx Assistant** для выполнения операций сбора данных. Для получения дополнительной информации об использовании NI-DAQmx обратитесь к справочному пособию NI-DAQmx или изучите расширенный курс *LabVIEW Data Acquisition and Signal Conditioning*.

Экспресс-ВП **DAQmx Assistant** позволяют просто осуществить настройку DAQ-устройства. При добавлении экспресс-ВП **DAQmx Assistant** на блок-диаграмму появляется диалоговое окно, в котором осуществляется конфигурация задания - провести определенные измерения. В процессе создания локального задания указывается необходимый тип измерения.

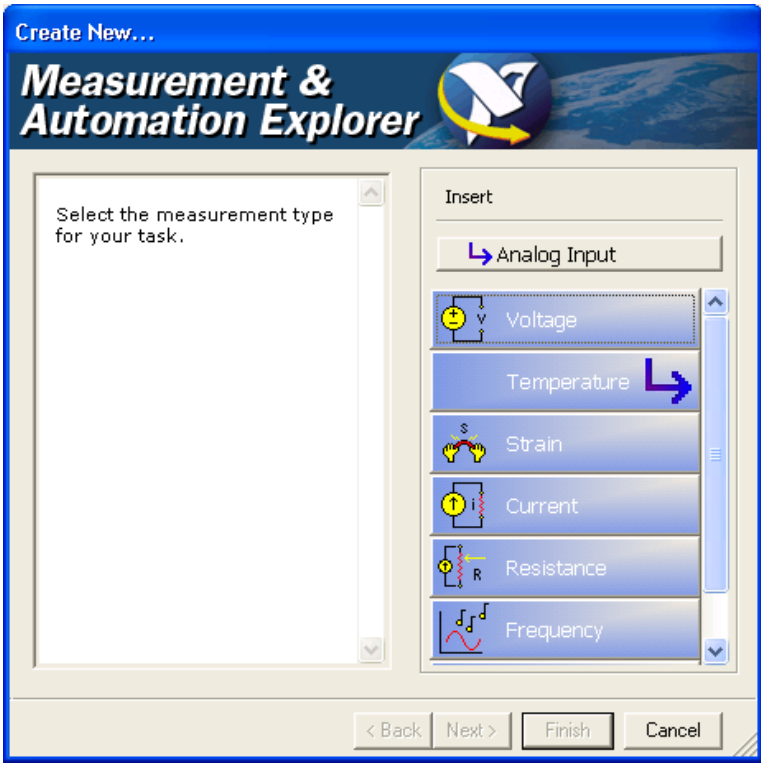


Один раз создав задание, информация о нем сохраняется в экспресс-ВП **DAQmx Assistant**. Впоследствии можно сконфигурировать экспресс-ВП **DAQmx Assistant** заново, дважды щелкнув по нему мышью.

C. Выполнение операций аналогового ввода

Используйте операции аналогового ввода для осуществления аналого-цифрового преобразования.

Существует несколько типов измерений входного сигнала: напряжение, температура, деформация, ток, сопротивление или частота.



Каждый тип измерений имеет собственные параметры, например, величина сопротивления для измерения тока или калибровка датчика деформаций для их измерений.

Установление временного такта выполнения заданий

При выполнении операций аналогового ввода данных задание может формулироваться по-разному: получение 1 значения, получение n значений или непрерывный сбор данных.

Получение одного значения

Получение одного значения является операцией по вызову. Другими словами NI-DAQmx оцифровывает одно значение с входного канала и немедленно возвращает его величину. Выполнение этой операции не требует наличия буфера и аппаратного контроля временного такта. Например, для периодического контроля уровня жидкости в резервуаре необходимо получать по одному значению. Вы можете подключить датчик, генерирующий разное напряжение в зависимости от уровня жидкости, к одному из каналов DAQ-устройства и производить контроль,

периодически оцифровывая по одному значению.

Получение n значений

Один из методов получения n значений - n раз получить по одному значению. Однако получение по одному значению из одного или нескольких каналов снова и снова является неэффективным и занимает много времени. Более того, отсутствует контроль времени между последовательными операциями получения значений. Вместо этого необходимо использовать аппаратное задание временного такта выполнения операций, в процессе которого используется буферизация полученных данных в компьютерной памяти, что приводит к более эффективному процессу сбора данных. С программной точки зрения необходимо включить режим аппаратного задания временного такта выполнения операций и задать частоту оцифровки **sample rate** и ограниченный по времени режим работы **sample mode (finite)**. Возможна оцифровка нескольких значений из одного канала или из нескольких.

С помощью NI-DAQmx можно осуществлять сбор данных с нескольких каналов. Например, необходимо контролировать уровень жидкости в резервуаре и ее температуру. В этом случае нужно иметь два датчика, подключенных к разным каналам DAQ-устройства.

Непрерывный сбор данных

Если необходимо отображать, обрабатывать и производить запись данных по мере их поступления, лучше использовать режим непрерывного сбора данных. Для этого устанавливается режим работы **sample mode (continuous)**.

Синхронизация заданий

Когда устройство, управляемое NI-DAQmx, работает, оно производит операции. Два наиболее часто встречающихся действия - обработка значения и начало сбора данных. Каждое производимое действие NI-DAQmx вызвано чем-либо, либо имеет причину. Такие причины называются синхронным запуском.

Start Trigger

Сигнал, запускающий сбор данных.

Reference Trigger

Сигнал, устанавливающий реперную точку в наборе входных значений. Данные, полученные до этой точки называются **pretrigger data**. Данные после реперной точки являются **posttrigger data**.

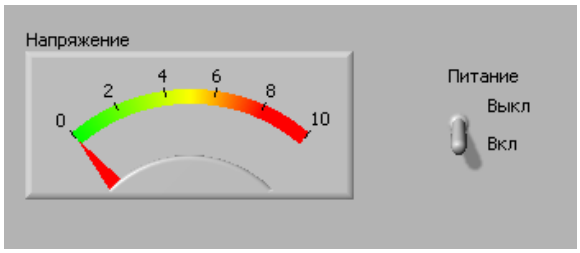
Упражнение 10-2. ВП Вольтметр

Цель: Ввод аналогового сигнала с помощью DAQ-устройства

Выполните следующие шаги для построения ВП, измеряющего напряжение с датчика температуры на сигнальной панели DAQ-устройства. Напряжение на выходе датчика пропорционально температуре. Убедиться в том, что датчик подключен к каналу «0» DAQ-устройства.

Лицевая панель

1. Откройте новый ВП и создайте лицевую панель, показанную на рисунке:



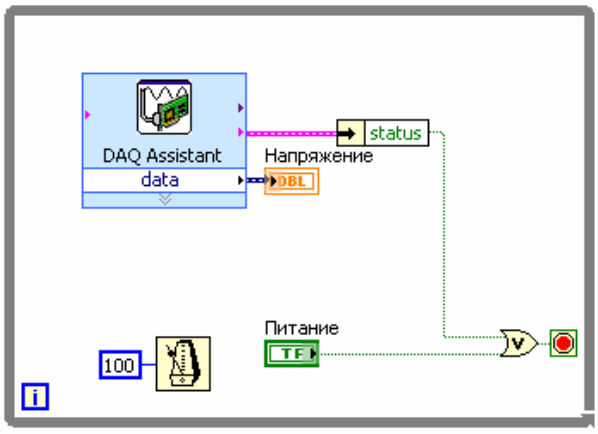
   a) Поместите элемент отображения **Meter** (Стрелочный индикатор)**,** размещенный на палитре **Элементов** в разделе **Controls»Modern»Numeric.** Настройте шкалу на диапазон «0.0 - 0.4». Для этого дважды щелкните с помощью инструмента ввод текста по полю значения числа «10.0» и введите значение «0.4». Возможно придется увеличить размер элемента, чтобы получить изображение, аналогичное приведенному выше.

   b) Поместите на лицевую панель вертикальный переключатель **vertical toggle switch**, расположенный в палитре **Controls»Modern»Boolean**. Установите его значение TRUE и назначьте тип механического действия **Latch When Pressed** (Изменение значения выключателя в нажатом состоянии).

   c) Создайте две свободный метки и назовите их **OFF** и **ON** с помощью инструмента ввод текста.

Блок-диаграмма

2. Создайте блок-диаграмму, как показано на рисунке:



Поместите на блок-диаграмму экспресс-ВП **DAQmx Assistant,** размещенную на палитре **Functions»Measurements I/O»DAQmx-Data Acquisition**. Настройте ВП на чтение аналогового сигнала с входного канала и возвращение значения напряжения.

a. Выберите тип измерения **Analog Input»Voltage**

b. Выберите физический канал **Dev1»ai0**

c. Появится конфигурационное диалоговое окно **Analog Input Voltage Task Configuration**. Установите метод измерений **Task Timing** в положение **Acquire 1 Sample** (измерение одной точки).

d. Нажмите на кнопку **OK** и закройте диалоговое окно. Все установки для этого задания будут сохранены локально в экспресс-ВП **DAQmx Assistant**.

Поместите на блок-диаграмму функцию **Wait Until Next ms Multiple** из палитры **Functions»Programming»Timing**. Функция синхронизирует выполнение цикла через каждые 100 мс.

Поместите на блок-диаграмму функцию **Unbundle by Name** из палитры **Functions»Programming»Cluster & Variant**. С помощью этой функции получите доступ к статусу ошибки **status** из кластера ошибок.

Добавьте на блок-диаграмму функцию **OR** из палитры **Functions»Programming»Boolean**. Значение на выходе этой функции остановит выполнение цикла, если произойдет ошибка, или пользователь переведет переключатель на лицевой панели в положение **OFF**.

3. Сохраните ВП под именем файла *Вольтметр.vi* ВП будет использоваться позднее в курсе.

4. Отобразите лицевую панель и запустите ВП.

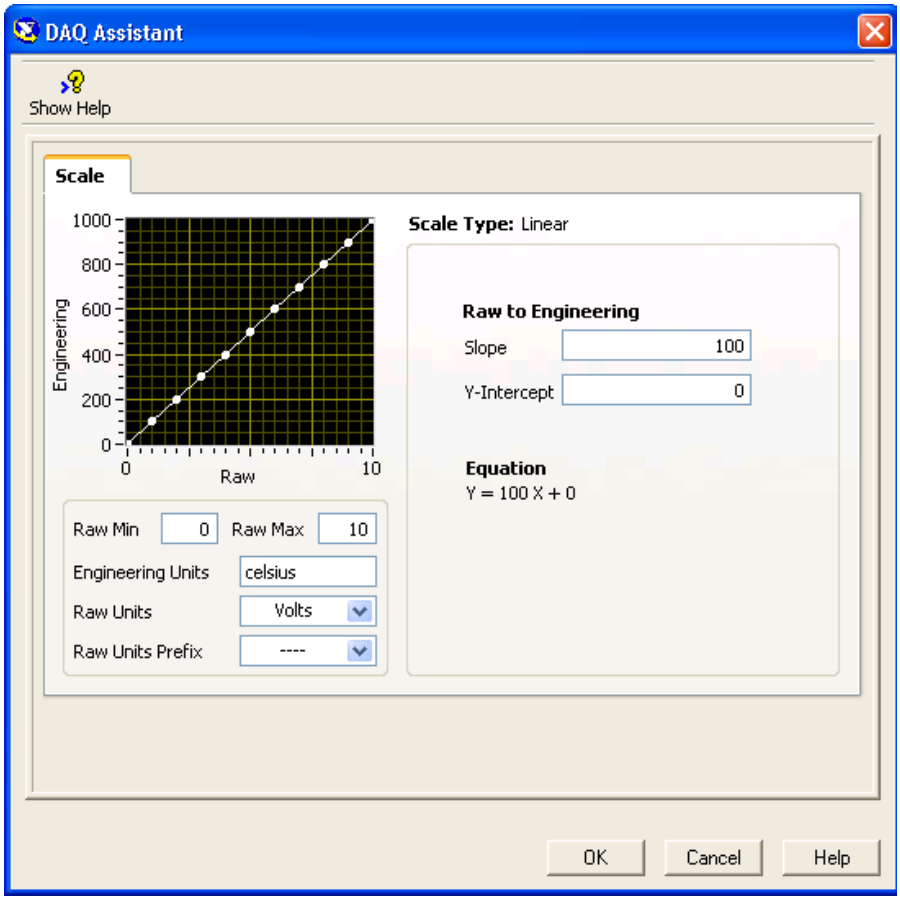
   На стрелочном элементе отображения будет показано значение напряжения датчика температуры. Возьмите датчик температуры в руку и зарегистрируйте повышение значения напряжения.

5. Закройте ВП.

Масштабирование

Температурный датчик сигнальной панели DAQ выдает значение напряжения равное градусам Цельсия, деленным на 100. Соответственно, для получения значений температуры необходимо полученные величины умножить на 100. Это можно с помощью функции умножения, либо настроить экспресс-ВП **DAQmx Assistant** на выполнение автоматического масштабирования значений. Более того, использование таких внутренних возможностей уменьшит сложность блок-диаграммы.

6. Дважды щелкните левой кнопкой мыши по иконке экспресс-ВП **DAQmx Assistant** для отображения диалогового окна **Analog Input Voltage Task Configuration.**
7. Выберите **Create New** и в элементе с циклическим перебором значений **Custom Scaling** выберите пункт **Scale Wizard**.
8. Выберите линейную зависимость **Linear** и назовите шкалу `temperature`. Нажмите на кнопку **Finish**.
9. Помощник масштабирования **Scale Wizard** откроет диалоговое окно, в котором можно отмасштабировать данные с помощью множителя и смещения. Установите наклон прямой **slope** равный 100, задайте единицы измерения **Engineering Units** равные градусам Цельсия **celsius**.



10. Нажмите на кнопку **ОК**.
11. В диалоговом окне **Analog Input Voltage Task Configuration** измените диапазон значений 0 – 100 и нажмите на кнопку **OK** для возвращения на блок-диаграмму.
12. Запустите ВП. Теперь элемент отображения показывает значение температуры в 100 раз большее значений напряжения. Измените масштаб элемента для правильного отображения величин.
13. Остановите ВП. Не закрывайте ВП, он понадобится в следующем упражнении.

Конец упражнения 10-2

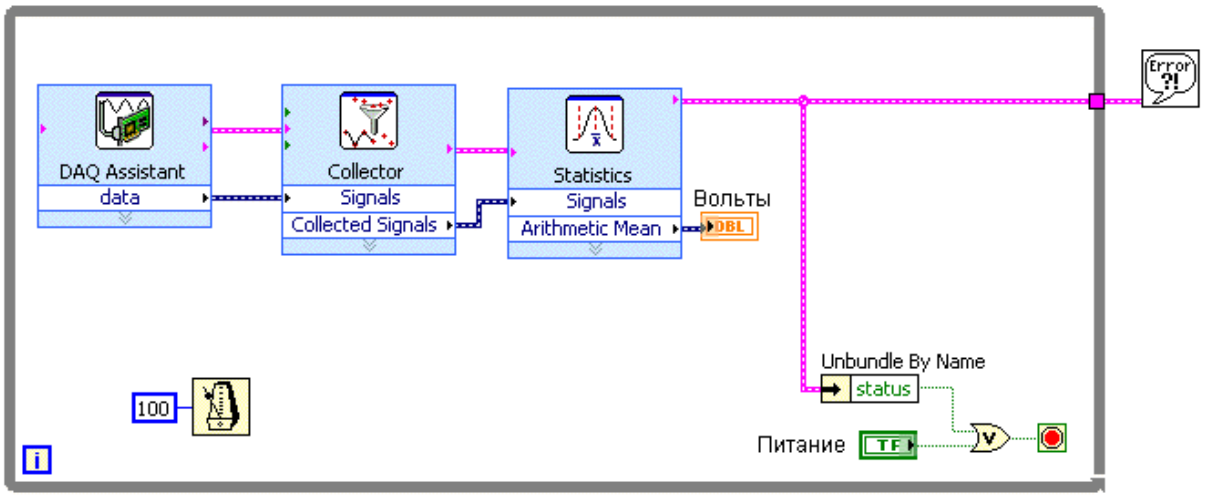
Упражнение 10-3. ВП Измерение среднего значения

Цель: Уменьшение шума аналоговых измерений с помощью процедуры усреднения.

1. Запустите ВП, созданный в предыдущем упражнении 10-2. Этот ВП будет измерять напряжение с температурного датчика и отображать его на графике.
2. Введите шум в измерения температуры, переместив переключатель **Temp Sensor Noise** (Шум Датчика) на сигнальной панели DAQ–устройства в положение **ON** (Включено)**.** Обратите внимание на появление шумовых помех в измерениях температуры.

Блок-диаграмма

3. Остановите выполнение ВП и отобразите блок-диаграмму. Измените структуру блок-диаграммы для вычисления среднего значения температуры по 100 измерениям.



4. Добавьте на блок-диаграмму экспресс-ВП **Collector**, расположенный в палитре **Functions»Express»Signal Manipulation**. Этот экспресс-ВП создает внутренний буфер для хранения отдельных значений. Когда собрано максимальное количество значений, экспресс-ВП **Collector** удаляет старые данные, добавляя на их место новые. Установите значение максимального количества собираемых данных равное 100.

5. Добавьте на блок-диаграмму экспресс-ВП **Statistics**, расположенный в палитре **Functions»Express»Signal Analysis**. Нажмите на кнопку **Arithmetic Mean** для расчета среднего значения полученных данных.
6. Запустите ВП. Отметьте уменьшение амплитуды шума при использовании усреднения.
7. Сохраните под именем *Измерение с усреднением.vi* и закройте ВП.

Конец упражнения 10-3

D. Запись полученных данных в файл

Часто бывает необходимо производить запись данных, полученных с помощью DAQ-устройства. При планировании сохранения данных в файл необходимо учесть следующие важные моменты:

- Не все приложения анализа данных используют LabVIEW. Подумайте, какое приложение будет использоваться для обработки сохраненных данных.
- Формат записи данных в файл определяет приложение, которое будет их обрабатывать. Поскольку LabVIEW обладает стандартными файловыми операциями, которые присутствуют и в других языках программирования, то существует полный доступ к формату записываемой информации.

LabVIEW может создавать LabVIEW Measurement File - текстовый ASCII файл, который может быть открыт в любом редакторе электронных таблиц или в текстовом редакторе. Файл формата LabVIEW Measurement File просто создается LabVIEW, легко открывается как в LabVIEW, так и в других приложениях.

Экспресс-ВП **Write LabVIEW Measurement File**, расположенный в палитре **Functions»Express»Output**, производит запись данных в файл формата LabVIEW Measurement File. При помещении экспресс-ВП **Write LabVIEW Measurement File** на блок-диаграмму появляется диалоговое окно, в котором указывается, как сохранить файл.

Экспресс-ВП **Read LabVIEW Measurement File**, расположенный в палитре **Functions»Express»Input**, производит чтение данных из файла формата LabVIEW Measurement File. Чтение данных производится по одному значению, поэтому необходимо помещать этот экспресс-ВП внутрь цикла.

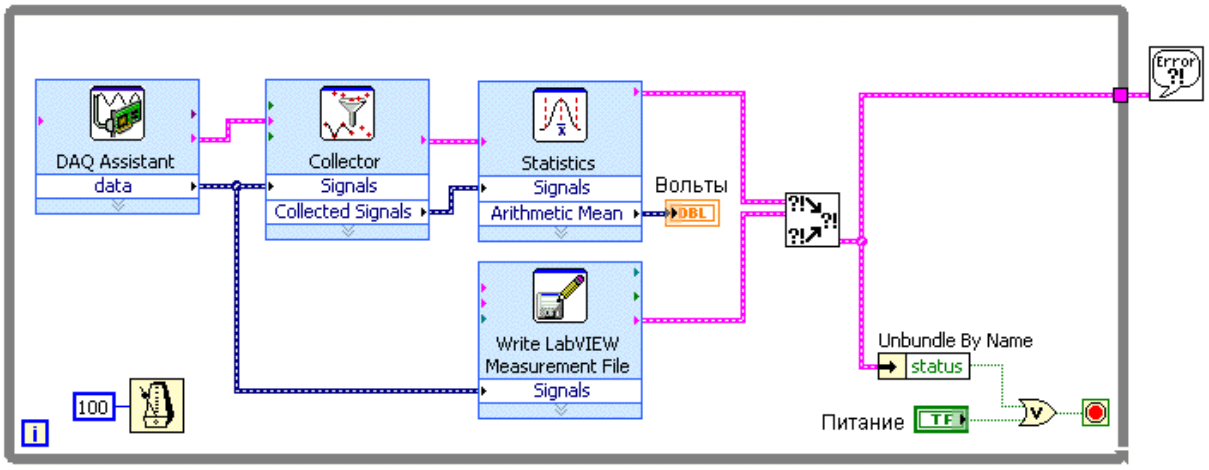
Упражнение 10-4. ВП простая запись данных в файл

Цель: изучить, как использовать LabVIEW Measurement File

Выполните изменения ВП, созданного в предыдущем упражнении 10-3, с целью добавления возможности записи и чтения измеренных данных из файла формата LabVIEW Measurement File.

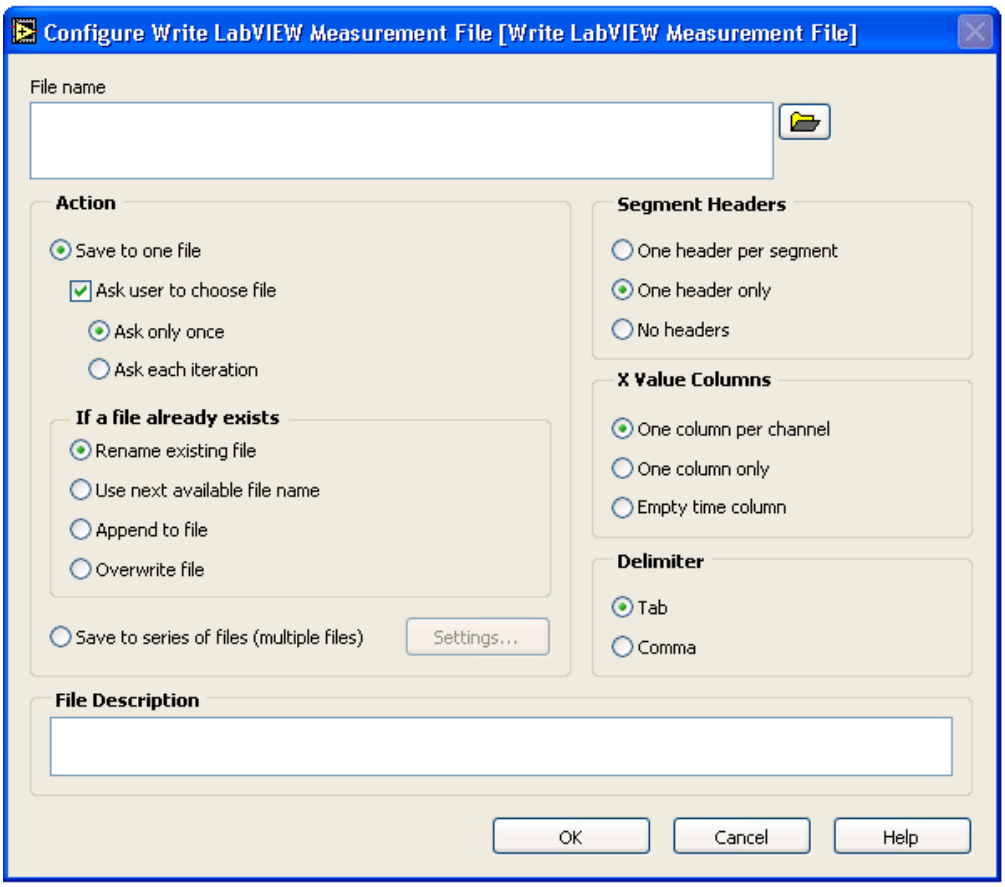
Блок-диаграмма ВП простая запись данных в файл

1. Откройте ВП, созданный в предыдущем упражнении.
2. Измените блок-диаграмму с целью добавления возможности записи и чтения измеренных данных, как это показано ниже



Поместите на блок-диаграмму экспресс-ВП **Write LabVIEW Measurement File** расположенный в палитре **Functions»Express»Output Export Waveforms to Spreadsheet File VI**. Этот экспресс-ВП записывает полученные данные в файл. В появившемся диалоговом окне **Configure LabVIEW Measurement File** сделайте следующие настройки:

a. Установите тип действия **Action** в состояние **Ask user to choose file** (спросить пользователя выбрать имя файла).
b. Установите правило записи заголовка файла **Segment Headers** в положение **One header only** для записи одного заголовка для всех данных. Заголовок содержит информацию о скорости и времени оцифровки данных.
c. Установите количество колонок в файле **X Value Columns** в положение **One column per channel** (одна колонка на канал) для записи данных в файл, который можно открыть в любом редакторе электронных таблиц или в текстовом редакторе.
d. Установите значение разделителя **Delimiter** - табуляция **Tab** для простого разделения колонок в файле.
e. Нажмите на кнопку **OK** для выхода из диалогового окна.

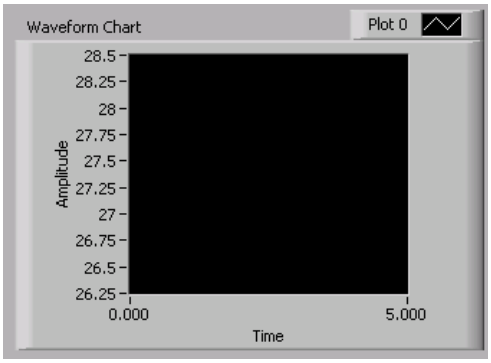


Поместите на блок-диаграмму ВП **Merge Errors**, расположенный в палитре **Functions»Programming»Dialog & User Interface**. Важно отслеживать ошибки в обоих экспресс-ВП: DAQ и файлового вывода. Поскольку код имеет две параллельные ветви выполнения, необходимо объединить ошибки обоих экспресс-ВП для определения корректности выполнения всего ВП.

3. Сохраните ВП.
4. Отобразите лицевую панель и запустите ВП. После сбора данных ВП запросит в диалоговом режиме имя файла данных. Ввести имя файла logger.lvm и нажмите кнопку **OK**.
5. Остановите и закройте ВП.

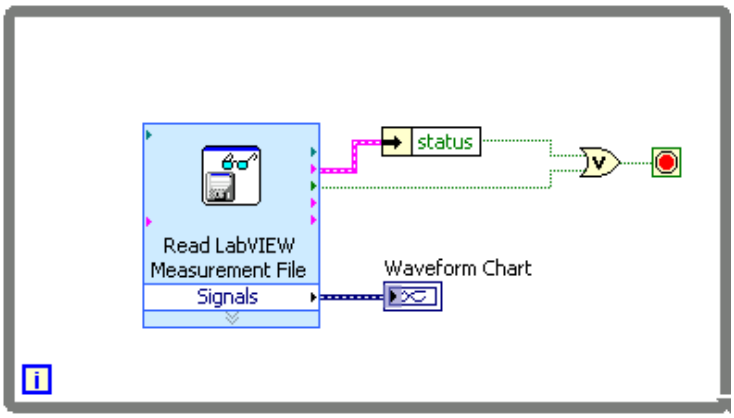
Лицевая панель ВП Простое чтение данных из файла

6. Откройте новый ВП и поместите на лицевую панель график Диаграмм, расположенный в палитре **Controls»Modern»Graph**.



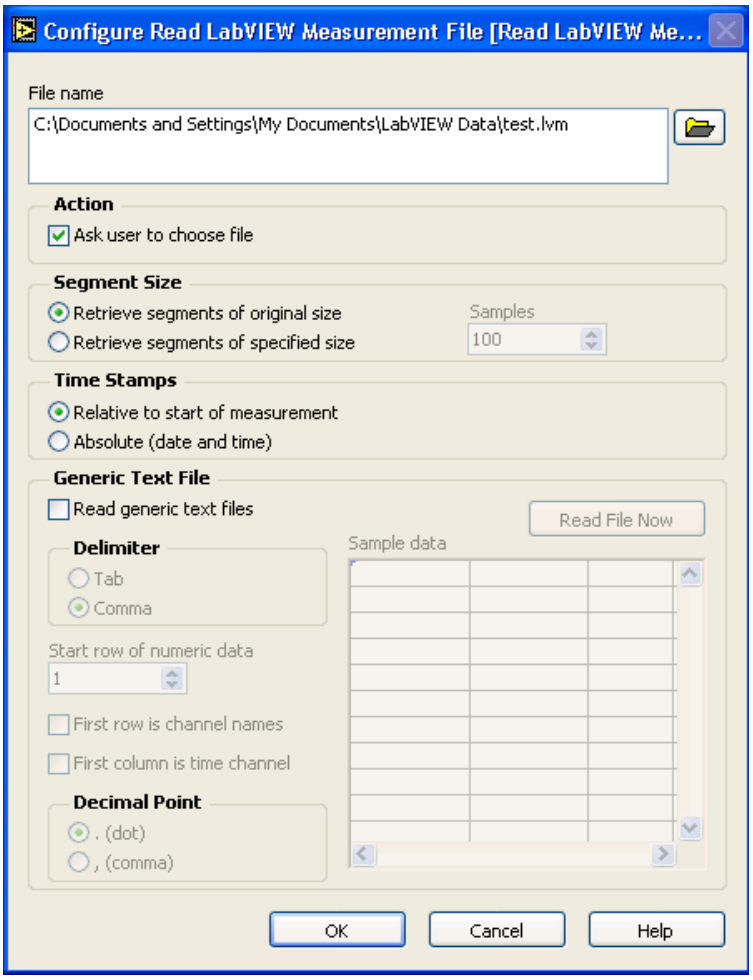
Блок-диаграмма

7. Создайте следующую блок-диаграмму



Поместите на блок-диаграмму экспресс-ВП **Read From Measurement File**, расположенный в палитре **Functions»Express»Input**. Поскольку этот экспресс-ВП считывает одно значение, он помещен внутрь цикла. В появившемся диалоговом окне **Configure LabVIEW Measurement File** сделайте следующие настройки:

a. Установите тип действия **Action** в состояние **Ask user to choose file** (спросить пользователя выбрать имя файла).
b. Установите максимальное количество считываемых значений **Segment Size** в положение **Retrieve segments of original size** (автоматическое определение).
c. Установите информацию о времени сбора данных **Time Stamps** в положение **Relative to start of measurement** (относительно начала измерений). Поскольку динамический тип данных хранит информацию о времени получения значений, то они упорядочиваются по времени измерения.
d. В разделе **Generic Text File** уберите выделение с пункта **Read generic text files** (чтение стандартного текстового файла), поскольку данные записывались в файл формата LabVIEW Measurement File.
e. Нажмите на кнопку **OK** для выхода из диалогового окна.



Соедините поле вывода данных **Data Available** (данные получены) с терминалом условия выхода из цикла. В результате выполнение цикла прекратится при прочтении всего файла формата LabVIEW Measurement File.

Поместите на блок-диаграмму функцию **Unbundle by Name** из палитры **Functions»Programming»Cluster & Variant**.

Добавьте на блок-диаграмму функцию **OR** из палитры **Functions»Programming»Boolean**.

8. Сохраните ВП под именем *Простая запись данных.vi*
9. Отобразите лицевую панель и запустите ВП. На запрос имени файла данных введите logger.lvm**,** созданного в пункте 4.
10. Данные из файла формата LabVIEW Measurement File появятся на графике Диаграмм.

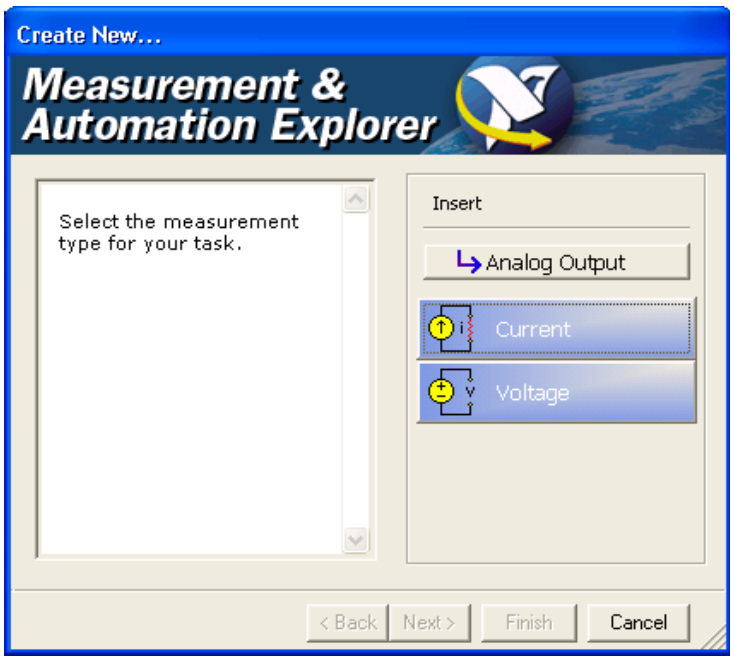
Примечание. Возможно Вам придется изменить масштаб осей графика для отображения всех данных.

11. Остановите и закройте ВП.

Конец упражнения 10-4

E. Выполнение операций аналогового вывода

Аналоговый выход используется для произведения цифро-аналогового преобразования. Величина на выходе может быть напряжением или током.



Для осуществления вывода напряжения или тока должно быть установлено соответствующее DAQ-устройство, позволяющее генерировать сигналы заданной формы.

Генерация данных

При выполнении операций аналогового вывода данных задание может формулироваться по-разному: генерация 1 значения, генерация n значений или непрерывная генерация данных.

Генерация одного значения

Генерация одного значения используется в том случае, когда уровень сигнала более важен, чем скорость его обновления. Например, используйте генерацию одного значения, если необходимо сгенерировать постоянный сигнал. В этом случае возможно использование программных средств управления временными задержками.

Выполнение этой операции не требует наличия буфера и аппаратного контроля временного такта. Например, необходимо сгенерировать известное напряжение для эмуляции работы устройства, в этом случае можно использовать метод генерации одного значения.

Генерация n значений

Один из методов генерации n значений - n раз сгенерировать по одному значению. Однако генерация по одному значению из одного или нескольких каналов снова и снова является неэффективной и занимает много времени. Более того, отсутствует контроль времени между последовательно сгенерированными значениями. Вместо этого

необходимо использовать аппаратное задание временного такта выполнения операций, в процессе которого используется буферизация генерируемых данных в компьютерной памяти, что приводит к более эффективному процессу генерации.

Возможно использование программного и аппаратного задания временного такта выполнения операций. В случае программного управления моментом генерации значений, временные задержки определяются программой и операционной системой, а не измерительным устройством. В случае аппаратного управления временным тактом генерация данных производится по TTL сигналу внутреннего таймера DAQ-устройства. Аппаратный таймер работает намного быстрее программных циклов. Также аппаратный таймер более точен по сравнению с программными циклами.

Примечание. Некоторые устройства не поддерживают аппаратное управление тактом. Обратитесь к руководству пользователя DAQ-устройства для получения подробной информации.

С программной точки зрения необходимо включить режим аппаратного задания временного такта выполнения операций и задать частоту оцифровки **sample rate** и ограниченный по времени режим работы **sample mode (finite)**. Возможна генерация значении на одном канале или нескольких.

Режим генерации n значений имеет смысл использовать при создании изменяющегося в конечный интервал времени сигнала, например, фрагмента переменного тока.

Непрерывная генерация данных

Непрерывная генерация данных аналогична генерации n значений с отличием в том, что для остановки непрерывной генерации должно произойти какое-то событие. Если необходимо непрерывно генерировать сигнал, для этого устанавливается режим работы **sample mode (continuous)**.

Синхронизация заданий

Когда устройство, управляемое NI-DAQmx, работает, оно производит операции. Два наиболее часто встречающихся действия - задание значения и начало его генерации. Каждое производимое действие NI-DAQmx вызвано чем-либо, либо имеет причину. Такие причины называются синхронным запуском.

Start Trigger

Сигнал, запускающий генерацию данных.

Reference Trigger

Этот сигнал не поддерживается для аналогового выхода.

Упражнение 10-5. ВП выходное напряжение

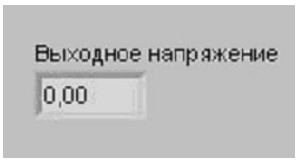
Цель: Вывести аналоговое напряжение, используя DAQ плату

Выполните следующие шаги для завершения создания ВП, который выводит напряжение от 0 до 9,5 Вольт с шагом 0,5 Вольта.

1. Соедините выход **Analog Out CH0** с входом **Analog In CH1** на сигнальной панели DAQ-устройства.

Лицевая панель

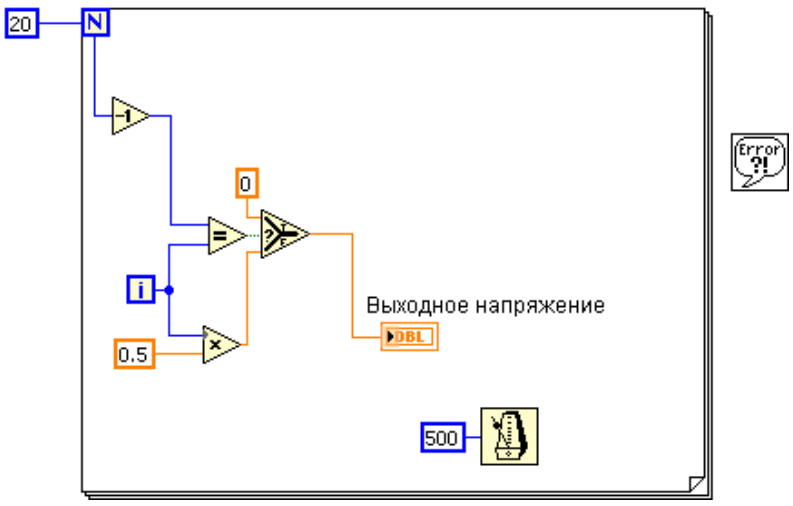
2. Откройте ВП *Вывод напряжения.vi* Откроется лицевая панель, показанная на рисунке.



Элемент отображения **Выходное напряжение** отображает текущее значение напряжения на выходе.

Блок-диаграмма

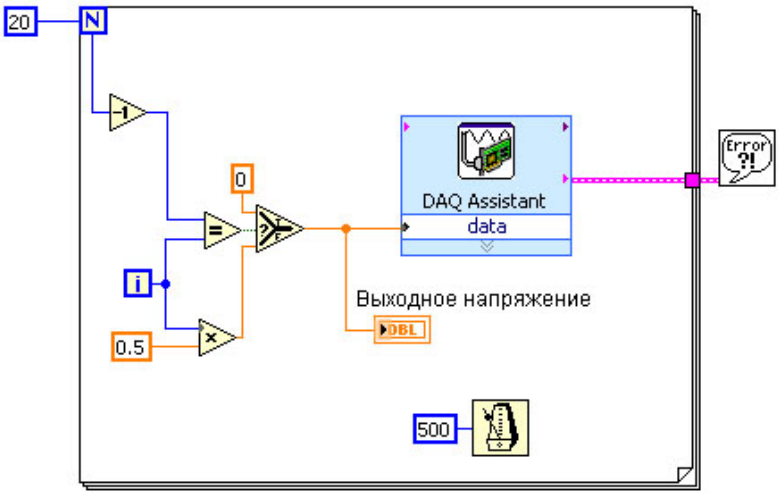
3. Отобразите блок-диаграмму и исследуйте ее.



- Функция **Wait Until Next ms Multiple**, размещенная в разделе **Functions»Programming»Timing**, вызывает выполнение итераций цикла **For Loop** через каждые 500 мс.

- ВП **Select**, расположенный в палитре **Functions»Programming»Comparison**, для проверки момента завершения работы цикла. Если работа цикла завершена, DAQ-устройство выводит нулевое значение напряжения. Это правило хорошего тона - установить значение выходного напряжения равное 0 по окончании работы.

4. Завершите создание блок-диаграммы, как это показано ниже



Поместите на блок-диаграмму в цикл **For** экспресс-ВП **DAQmx Assistant**, расположенный в палитре **Functions»Measurements I/O» DAQmx - Data Acquisition**. Завершите настройку экспресс-ВП **DAQmx Assistant** для генерации аналогового сигнала:

a. Выберите тип измерений **Analog Output»Voltage**.
b. Выберите физический канал **Dev1»ao0**.
c. В диалоговом окне **Analog Output Voltage Task Configuration** выберите режим генерации **Task Timing - Generate 1 Sample** (генерация одного значения). Измените диапазон выводимых значений напряжения 0 - 10.
d. Нажмите на кнопку **OK** для закрытия диалогового окна **Analog Output Voltage Task Configuration**. Все настройки были сохранены в экспресс-ВП **DAQmx Assistant**.

5. Сохраните ВП.
6. Закройте блок-диаграмму и откройте ВП, созданный в упражнении 10-2.
7. Настройте шкалу элемента отображения **Вольты** на диапазон от 0.0 до 10.0.
8. Откройте блок-диаграмму этого ВП и дважды щелкните левой кнопкой мыши по экспресс-ВП **DAQmx Assistant** для отображения диалогового окна **Analog Input Voltage Task Configuration**.
9. Щелкните правой кнопкой мыши по пункту **Voltage** в разделе **Channel List** и выберите **Change Physical Channel** (изменить физический канал). Выберите канал **ai1**, поскольку Вы соедините входной канал 0 с выходным каналом 1 сигнальной панели DAQ-устройства.
10. Установите отсутствие масштабирования **No Scale** в выпадающем меню **Custom Scaling**.
11. Измените диапазон напряжения от 0 до 10.
12. Нажмите на кнопку **OK**, закройте диалоговое окно.

13. Запустите ВП.
14. Для сбора и отображения напряжения запустите ВП, созданный в этом упражнении. Происходит генерация напряжения от 0 до 9,5 Вольт с шагом 0,5 Вольта. После завершения цикла **For**, ВП обнулит значение напряжение выхода.

15. Закройте оба Виртуальных Прибора.

Конец упражнения 10-5

F. Информация о счетчиках

Счетчики – это цифровые временные устройства. Обычно счетчики используют для подсчета произошедших событий, измерения периода и частоты сигнала и генерации импульсов.

Счетчик состоит из четырех основных компонентов: регистр значений счетчика, источник, сигнал управления и выходной сигнал.

- **регистр значений счетчика** - содержит текущее значение счетчика. Значение, хранимое в регистре, можно узнать программно.
- **источник** - сигнал, вызывающий изменение значения счетчика, хранимого в регистре. Счетчик реагирует на возрастающий или спадающий фронт сигнала. Какой тип фронта сигнала вызывает изменение состояния счетчика - задается программно. Программно выбранный тип фронта сигнала называется активным. Когда на вход счетчика подается активный фронт сигнала, его значение изменяется на единицу. Программно задается и знак изменения значения счетчика - увеличивается оно или уменьшается.
- **сигнал управления** - входящий сигнал, определяющий вызывает ли активный уровень изменение состояния счетчика. Счет может происходить при высоком или низком уровне этого сигнала, или при различных комбинациях возрастающих и/или спадающих фронтов сигнала управления. Сигнал управления формируется программно.
- **выходной сигнал** - сигнал, генерируемый импульсы или серию импульсов.

Увеличение значения счетчика, сконфигурированного для подсчета простых событий, происходит при поступлении на вход источника сигнала с активным фронтом. Чтобы счетчик считал события при поступлении активного фронта сигнала, он должен быть инициализирован. Разрядность счетчика определяет его разрешение, например, 24-битовый счетчик может подсчитать следующее число событий:

$$2^{(\text{Разрядность счетчика})} - 1 = 2^{24} - 1 = 16{,}777{,}215$$

Когда 24-разрядный счетчик достигает значения 16,777,215, это означает, что он достиг своего предельного значения. Следующее событие приведет к его переполнению и сбросу на 0.

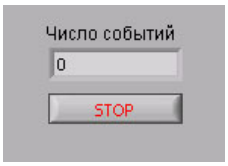
Упражнение 10-6. ВП простой подсчет событий

Цель: Создать ВП для подсчета простых событий

В этом упражнении необходимо исследовать ВП, считающий импульсы генератора сигнальной панели DAQ-устройства.

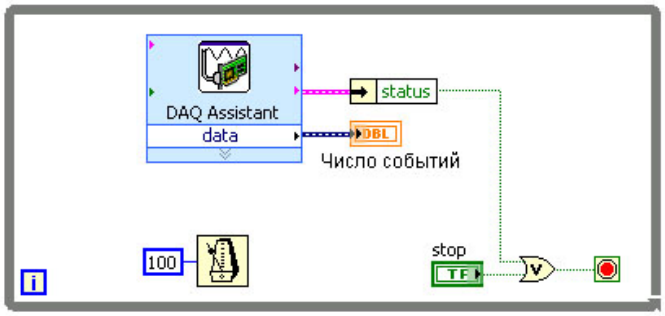
Лицевая панель

1. Откройте новый ВП и создайте следующую лицевую панель.



Блок-диаграмма

2. Создайте следующую блок-диаграмму.



Поместите на блок-диаграмму экспресс-ВП **DAQmx Assistant,** расположенный в палитре **Functions»Measurements I/O»DAQmx - Data Acquisition**. Завершите настройку счетчика для осуществления регистрации событий:

a. Выберите тип измерений **Counter Input»Edge Count**.

b. Выберите физический канал **Dev1»ctr0.**

c. В диалоговом окне **Counter Input Edge Count Task Configuration** оставьте все как есть. Значения по умолчанию определяют в качестве источника сигнала счетчика 0 программный вход Programmable Function Input (PFI) 8. На сигнальной панели DAQ вход счетчика 0 подключен к линии Programmable Function Input (PFI) 8.

d. Нажмите на кнопку **OK** для закрытия диалогового окна **Counter Input Edge Count Task Configuration**. Все настройки были сохранены в экспресс-ВП **DAQmx Assistant**.

3. Сохранение ВП под именем *Простой подсчет событий.vi*

4. На сигнальной панели DAQ-устройства соедините выход A генератора прямоугольного сигнала со входом SOURCE счетчика 0.

5. Запустите ВП, повращайте ручку **quadrature encoder knob** на сигнальной панели DAQ-системы. Обратите внимание, что значение элемента отображения **Число событий** увеличивается при вращении ручки, которое приводит к генерации прямоугольных импульсов. Счетчик подсчитывает эти импульсы.

6. Остановите ВП.

7. На сигнальной панели DAQ-устройства соедините выход фазы B генератора прямоугольного сигнала со входом Up/Down счетчика 0. Это можно использовать для определения направления поворота ручки. Дважды щелкните левой кнопкой мыши по экспресс-ВП **DAQmx Assistant** и выберите значение **Externally Controlled** из циклического списка **Count Direction**. Нажмите на кнопку **OK** для закрытия диалогового окна.

8. Запустите ВП. Повращайте ручку **quadrature encoder knob** на сигнальной панели DAQ-устройства. Обратите внимание, что значение элемента отображения **Число событий** увеличивается при вращении ручки против часовой стрелки, и уменьшается при вращении по часовой стрелке.

9. Сохраните и закройте ВП.

Конец упражнения 10-6

G. Ввод и вывод цифровых сигналов

Измерение и генерация цифровых сигналов используется в большом количестве приложений, включая мониторинг систем безопасности. В основном, генерация и измерение цифровых сигналов применяется в лабораторных исследованиях, тестировании продуктов и промышленных процессах мониторинга и контроля.

Цифровой ввод-вывод представляет собой чтение или запись значения в цифровую линию или во весь цифровой порт, состоящий из совокупности линий.

Вы можете использовать цифровые линии DAQ-устройства для сбора цифровых значений. Этот сбор данных основывается на программном задании временного такта выполнения операций. На некоторых устройствах можно настраивать цифровые линии независимо для генерации или сбора данных. Каждая цифровая линия является отдельным каналом в LabVIEW.

Вы можете использовать цифровые порты DAQ-устройства для сбора данных совокупности цифровых линий. Этот сбор данных основывается на программном задании временного такта выполнения операций. Вы можете настраивать цифровые порты независимо для генерации или сбора данных. Каждый цифровой порт является отдельным каналом в LabVIEW.

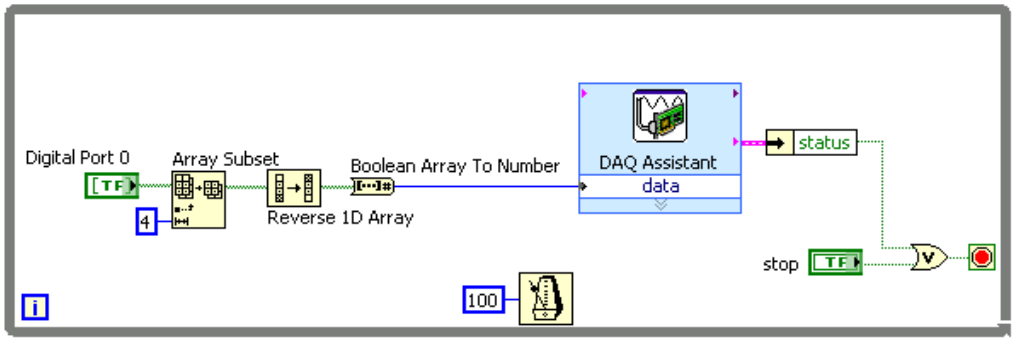
Упражнение 10-7. ВП цифровой пример

Цель: Управление цифровыми линиями DAQ-устройства

Выполните следующие действия для создания ВП, включающего светодиоды порта 0 на сигнальной панели DAQ-устройства в соответствии с цифровым значением, вводимым на лицевой панели. Каждый светодиод подключен к цифровой линии DAQ-устройства. Линии пронумерованы 0, 1, 2, 3, начиная с правого светодиода.

Примечание. Светодиоды используют отрицательную логику. Следовательно, подача сигнала по цифровой линии, подсоединенной к светодиоду, вызывает его выключение. Запись нуля в линию вызывает зажигание светодиода.

1. Откройте ВП Digital Example, расположенный в папке `C:\Exercises\LV Basics I`, и измените блок-диаграмму.



Поместите на блок-диаграмму внутрь цикла **While** экспресс-ВП **DAQmx Assistant,** расположенный в палитре **Functions»Measurements I/O»DAQmx - Data Acquisition**. Завершите настройку счетчика для осуществления регистрации событий:

a. Выберите тип измерений **Digital I/O»Port Output**.

b. Выберите физический канал **Dev1»port0**.

c. В диалоговом окне **Digital Output Port Task Configuration** отметьте пункт **Invert All Lines In Port** (инвертировать все линии цифрового порта), так как светодиоды работают согласно отрицательной логике.

d. Нажмите на кнопку **OK** для закрытия диалогового окна **Digital Output Port Task Configuration**. Все настройки были сохранены в экспресс-ВП **DAQmx Assistant**.

Кнопки на лицевой панели находятся в массиве для упрощения кода. Функция **Array Subset** выделяет только первые четыре элемента массива. Элементы выделенного массива необходимо поменять местами, поскольку элемент 0 массива является самым значительным битом. После этого массив преобразуется в число с помощью функции **Boolean Array to Number** и подается на вход экспресс-ВП **DAQmx Assistant** для записи в цифровой порт.

2. Сохраните ВП.
3. Отобразите лицевую панель и запустите ВП. Включите и выключите светодиоды, наблюдайте за изменениями на сигнальной панели DAQ-устройства.
4. Остановите и закройте ВП.

Конец упражнения 10-7

Краткое изложение пройденного материала, советы и секреты

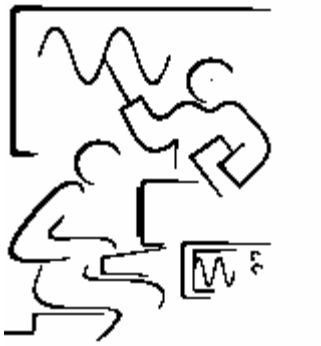
- MAX является основной конфигурационным и тестовым приложением для DAQ-устройств.
- Экспресс-ВП **DAQmx Assistant** используется для настройки DAQ-устройств и проведения измерений.
- Большинство программ могут использовать экспресс-ВП **DAQmx Assistant**. В приложениях, использующих расширенные возможности установки временного такта и синхронизации, используются Виртуальные приборы, поставляемые вместе с экспресс-ВП **DAQmx Assistant**.
- **DAQmx Assistant** может осуществлять операции аналогового и цифрового ввода-вывода и операции со счетчиками.

Дополнительные упражнения

10-8. Постройте ВП для непрерывного измерения значения температуры с частотой 2 Гц и отображения измеренных значений на графике Диаграмм. Если температура превышает установленный предел, ВП должны зажечься светодиоды **LED** на лицевой панели и **LED 0** на сигнальной панели DAQ-системы. График Диаграмм должен отображать как значения температуры, так и значения текущего предела.

Сохраните ВП под именем файла *Монитор температуры с LED.vi*

Примечания



Урок 11
Управление измерительными приборами

Урок описывает использование среды LabVIEW для управления и сбора данных приборами, работающими через интерфейс GPIB и последовательный порт. Для управления вводом/выводом данных с измерительных приборов следует использовать драйверы приборов и экспресс-ВП **Instrument I/O Assistant**.

В этом уроке изложены вопросы:

A. Управление измерительными приборами.

B. GPIB-интерфейс и его настройка.

C. Использование **Instrument I/O Assistant**.

D. Архитектура программного обеспечения виртуальных интерфейсов **VISA**.

E. Драйверы измерительных приборов.

F. Использование ВП драйвера устройства.

G. Последовательная связь.

H. Передача сигнальных данных (дополнительно).

A. Управление измерительными приборами

Среда LabVIEW не накладывает ограничения на средства управления измерительными приборами, если они удовлетворяют индустриальным технологическим стандартам управления. LabVIEW позволяет использовать различные коммуникационные интерфейсы, такие как последовательный и параллельный порты, **GPIB**, **VXI**, **PXI**, **Ethernet**, **SCSI** и **CAMAC**. Этот урок описывает два наиболее распространенных коммуникационных интерфейса: GPIB и последовательный порт.

Для выполнения урока понадобится следующая информация и перечень дополнительных аппаратных средств:

- Тип разъема у измерительного прибора.
- Нуль-модемный кабель с заданным числом контактных штырьков и типом разъемов «мама» или «папа».
- Техническая характеристика выхода измерительного прибора: уровень выходного сигнала, максимальная длина соединительного кабеля и наличие заземления.
- Используемые коммуникационные протоколы: ASCII-команды, двоичные команды и формат передаваемых данных.
- Существующие драйверы устройства.

B. GPIB-интерфейс и его настройка

Стандарт ANSI/IEEE Standard 488.1-1987, известный также как **General Purpose Interface Bus** (**GPIB**), описывает стандартный интерфейс для связи между измерительными и управляющими приборами (сканеры, пленочные регистраторы и т.д.) различных производителей. Он включает в себя информацию об электрических, механических и функциональных спецификациях интерфейса. Интерфейс **GPIB** – это цифровой 8-битный параллельный коммуникационный интерфейс со скоростью передачи данных 1 Мбайт/с и выше. Он использует 3 линии синхронизации данных. Шина **GPIB** поддерживает один системный контроллер, обычно компьютер, и может управлять дополнительно 14-ю измерительными приборами. Стандарт ANSI/IEEE Standard 488.2-1992 расширяет возможности IEEE 488.1 и описывает протокол связи, общие форматы данных и управляющих команд для стандартных устройств.

Интерфейс **GPIB** часто вводится в измерительные приборы широкого класса производителей с целью обеспечения возможности их тестирования. Интерфейс традиционно используется для самостоятельных настольных измерительных приборов с ручным управлением.

GPIB является 24-проводной параллельной шиной, состоящей из 8-ми линий данных, 5-ти линий управления шиной (**ATN**, **EOI**, **IFC**, **REN**, и **SRQ**), 3-х линий синхронизации и 8-ми заземляющих линий. В интерфейсе **GPIB** использована побайтовая асинхронная схема передачи данных, т.е. байты целиком последовательно передаются через шину на скорости, которая определяется скоростью самого медленного участника передачи. Поскольку в качестве единицы данных используется 1 байт, то передаваемые сообщения кодируются как символьные строки ASCII.

Адресация в интерфейсе GPIB

Каждый **GPIB**-измерительный прибор и **GPIB**-интерфейсная плата имеют уникальный **GPIB**-адрес в диапазоне от 0 до 30. Адрес 0 обычно присваивается **GPIB**-интерфейсу. Измерительные приборы, связанные с **GPIB**-интерфейсом, могут иметь адреса от 1 до 30. Каждое **GPIB**-устройство может быть передатчиком (**talker**) – источником сообщения, слушателем (**listener**) – устройством, принимающим данные, или контроллером (**controller**). Контроллер, обычно компьютер, управляет потоком информации, передаваемым по шине. Он определяет коммуникационные связи и посылает **GPIB**-команды измерительным приборам. Виртуальные Приборы **GPIB** автоматически оперируют адресацией и большинством других управляющих команд шины.

Остановка передачи данных

Прервать передачу **GPIB**-данных можно следующими тремя методами:

- Аппаратная линия **GPIB** (**EOI**) изменяет уровень при передаче последнего байта данных. Этот метод считается наиболее предпочтительным.
- Установка управляющего символа **end-of-string** (**EOS**) в конце строки передаваемых данных. Некоторые измерительные приборы используют этот метод вместо или в дополнение к аппаратному обозначению окончания передачи данных на линии **EOI**.
- Слушатель считывает заданное количество переданных байтов и останавливает чтение, когда счетчик достиг предельного значения. Этот метод часто используется как метод прерывания по умолчанию, потому что передача прекращается в соответствии с результатом логической операции ИЛИ параметров **EOI** и **EOS** (если этот способ задействован), логически умноженной на параметр счетчика байтов. Таким образом, количество байтов на счетчике устанавливается равным или превышающим число байтов, которые нужно прочесть.

Ограничения

Для достижения высокой скорости передачи, заложенной в **GPIB**-интерфейсе, следует ограничить количество измерительных приборов, присоединенных к шине и физическое расстояние между устройствами. Ниже перечислены типичные ограничения:

- Максимальное расстояния между двумя соседними измерительными приборами должно быть 4 м, а среднее расстояние должно составлять 2 м.
- Максимальная длина кабеля – 20 м.
- Максимальное количество измерительных приборов, соединенных с каждой шиной не должно превышать 15, при этом по крайне мере две трети приборов должны быть включены.

Для достижения максимальной скорости передачи данных необходимо выполнять следующие условия:

- Все измерительные приборы в системе должны быть включены.
- Длина кабеля должна быть как можно короче, максимум длины кабеля для каждой системы составляет 15 м.
- Измерительные приборы должны располагаться через каждый метр длины кабеля.

Если необходимо превысить перечисленные ограничения, то при увеличении длины кабеля используется устройство **bus extender**, а при увеличении количества измерительных приборов – устройство **bus expander**. Эти устройства можно заказать на сайте компании National Instruments.

Примечание Для получения дополнительной информации о **GPIB** используйте ссылку на Web-сайт: *ni.com/support/gpibsupp.htm.*

Архитектура программных средств

Архитектура программных средств LabVIEW для управления измерительными приборами посредством **GPIB**-интерфейса аналогична архитектуре программных средств **DAQ**. В **GPIB**-систему включен комплект драйверов, эти драйверы также входят в комплект поставки LabVIEW CD, и большинство из них можно скачать по адресу *ni.com/support/gpib/versions.htm*. Всегда необходимо устанавливать новейшие версии драйверов, если нет особых указаний в инструкции к конкретному **GPIB**-интерфейсу или в руководстве к версии LabVIEW.

(**Windows**) Конфигурационная утилита **MAX** (**Measurement & Automation Explorer**) используется для настройки и тестирования аппаратных средств **GPIB**. **MAX** взаимодействует с установленными вместе с драйвером средствами диагностики и настройки, а также с реестром и Диспетчером устройств ОС Windows. Программный драйвер является динамической библиотекой (**DLL**) и включает в себя все функции прямой связи с **GPIB**-интерфейсом. Виртуальные Приборы и функции раздела **Instrument I/O** работают напрямую с драйверами устройств.

Программные средства настройки (Windows)

Примечание (**MacOS и UNIX)** Для получения информации о настройке и тестировании **GPIB**-интерфейса используйте документацию интерфейса **GPIB**.

Утилита настройки **MAX** предназначена для управления аппаратными и программными средствами компании National Instruments. Утилита позволяет проводить диагностику системы, добавлять новые каналы, интерфейсы и виртуальные каналы, просматривать устройства и измерительные приборы, подсоединенные к системе.

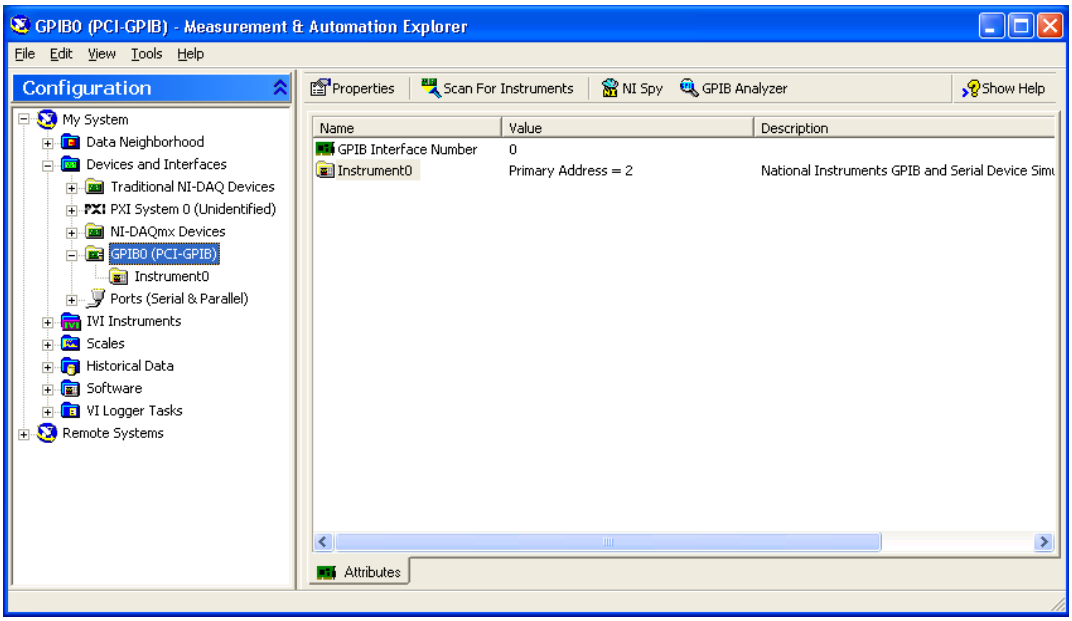
Запуск утилиты **MAX** осуществляется двойным щелчком левой кнопкой мыши по ее иконке на рабочем столе или выбором в главном меню среды LabVIEW пункта **Tools»Measurement & Automation Explorer**.

Панель **Configuration** утилиты **MAX** включает в себя несколько секций под элементом **My System**:

- **Data Neighborhood** – секция предназначена для создания и тестирования виртуальных каналов, имен и дескрипторов каналов и измерений, настраиваемых в секции **Devices and Interfaces**, аналогично проделанному в Уроке 10, *Сбор и отображение данных*.
- **Devices and Interfaces** – секция предназначена для настройки ресурсов и других физических свойств устройств и интерфейсов, а также для просмотра атрибутов, например, серийных номеров, одного или многих устройств.
- **IVI Instruments** – секция предназначена для создания имен виртуальных инструментов **IVI**, изменения их параметров и перемены мест инструментов **IVI**.

- **Scales** – секция предназначена для проведения простых операций масштабирования данных, таких как преобразование температуры из единиц напряжения Вольты, зарегистрированных температурным датчиком, в градусы шкалы Цельсия.
- **Historical Data** – секция используется для доступа к базам данных и записанным данным.
- **Software** – секция предназначена для определения установленных драйверов и приложений от National Instruments, а также их версий.
- **VI Logger Tasks** – секция используется для создания, изменения, запуска и просмотра задач **VI Logger**.

На следующей иллюстрации показана панель утилиты **MAX** после нажатия кнопки **Scan for Instruments** на инструментальной панели.

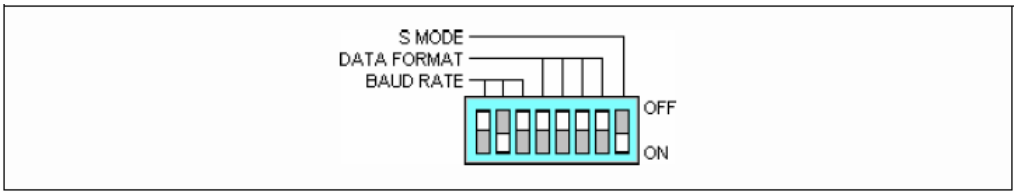


Раздел **Remote Systems** на панели **Configuration** позволяет просматривать и настраивать удаленные системы, такие как контролеры **RT Series PXI Controllers**. Настройка объектов, перечисленных в **MAX**, осуществляется щелчком правой кнопкой мыши по имени объекта и выбором пунктов контекстного меню

Упражнение 11-1 Настройка GPIB с помощью MAX (только для Windows)

Цель: Исследование настроек GPIB-интерфейса с помощью утилиты MAX, определение подключенных измерительных приборов и установка связи с измерительными приборами.

1. Выключите **NI Instrument Simulator** и настройте его для связи с **GPIB**-интерфейсом с помощью установок блока переключателей, показанных на рисунке.



2. Включите **NI Instrument Simulator** и проверьте, загорелись ли светодиоды **Power** и **Ready**.
3. Запустите утилиту **MAX** двойным щелчком левой кнопки мыши по иконке на рабочем столе или выбором пункта главного меню **Tools»Measurement & Automation Explorer** в среде LabVIEW.
4. Откройте секцию **Devices and Interfaces**, которая отобразит установленные в системе интерфейсы. Если **GPIB**-интерфейс присутствует в списке, то программные средства **NI-488.2** правильно загружены в систему.
5. Выделите **GPIB**-интерфейс и нажмите кнопку **Properties** на инструментальной панели для отображения диалогового окна **Properties.**
6. Изучите, но не изменяйте установки **GPIB**-интерфейса и нажмите на кнопку **OK**.
7. Убедитесь в том, что в секции **Devices and Interfaces** все еще выбран **GPIB**-интерфейс, и нажмите на кнопку **Scan for Instruments** инструментальной панели.
8. Откройте секцию **GPIB-board**. В секции появится измерительный прибор под именем **Instrument0**.
9. Выделите имя **Instrument0** для отображения информации в правой половине панели **MAX**.

   NI Instrument Simulator имеет первичный адрес **GPIB** (**PAD**)**,** равный **2**.

Примечание Для отображения информации об **Instrument0** необходимо закрыть контекстную подсказку щелчком кнопки **Show/Hide** в верхнем правом углу окна **MAX.**

10. Нажмите на кнопку **Communicate with Instrument** инструментальной панели. Появится интерактивное окно, в котором можно посылать запросы, производить чтение с измерительного прибора или запись в него.

11. Введите текст ***IDN?** в поле **Send String** и нажмите на кнопку **Query.** Измерительный прибор вернет номер модели в поле **String Received**. Можно использовать это окно при отладке измерительного прибора или проверки управляющих команд, описанных в документации на измерительный прибор.

12. Введите текст **MEAS:DC?** в поле **Send String** и нажмите на кнопку **Query**. **NI Instrument Simulator** вернет значение эмулированного напряжения.

13. Нажмите на кнопку **Query**, и измерительный прибор выдаст следующее значение.

14. Нажмите на кнопку **Exit**.

15. Установите имя VISA (**VISA alias**) для **NI Instrument Simulator**. Таким образом, не нужно будет запоминать первичный адрес, и использовать только имя.

    a. При выбранном в **MAX** имени **Instrument0** нажмите на кнопку **VISA Properties** для отображения диалогового окна **Properties.**

    b. Введите текст **devsim** в поле **VISA alias** и нажмите на кнопку **OK**. Теперь в этом уроке для обращения к инструменту можно использовать это имя.

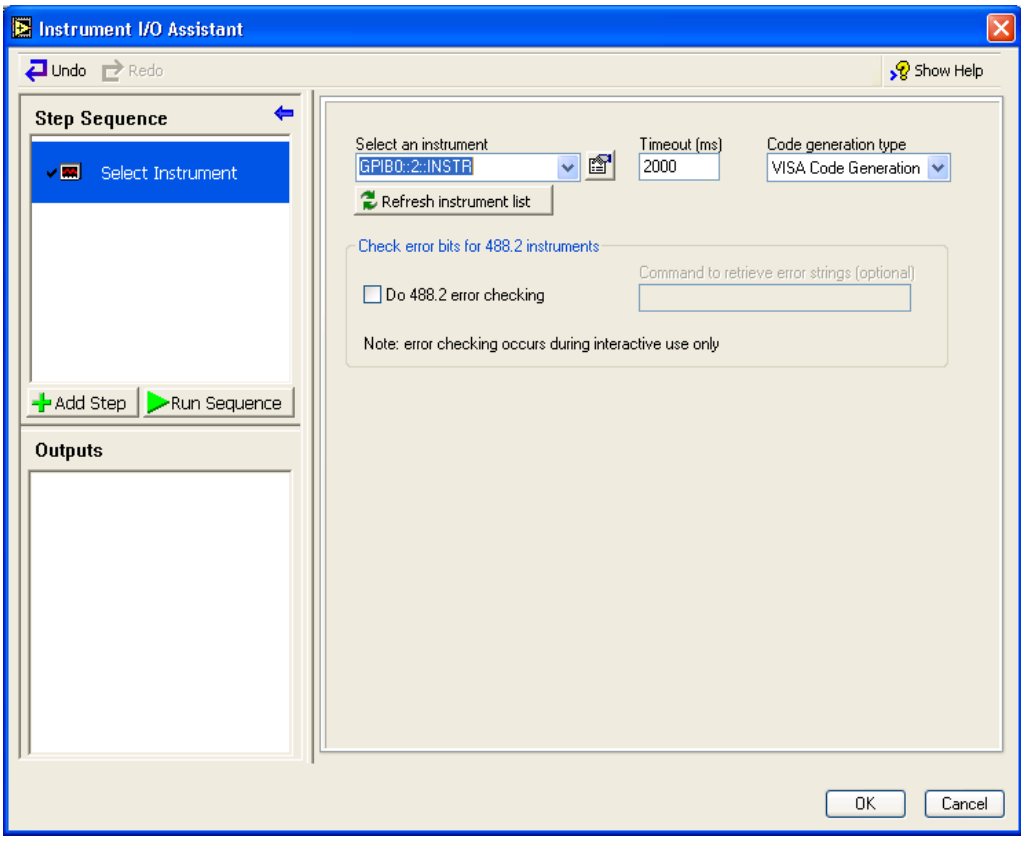
16. Выберите пункт **File»Exit** для выхода из конфигурационной утилиты **MAX**.

Конец упражнения 11-1

C. Использование Instrument I/O Assistant

ВП **Instrument I/O Assistant**, расположенный в палитрах **Functions»Express»Input** и **Functions»Instrument I/O**, является экспресс-ВП среды LabVIEW. Этот экспресс-ВП позволяет легко проверять связь с измерительными приборами, а также разрабатывать последовательности запросов, анализа и записи данных. Эти этапы могут быть сохранены как экспресс-ВП для непосредственного использования или конвертированы в подпрограмму ВП. **Instrument I/O Assistant** следует использовать, когда нет необходимых драйверов к измерительному прибору.

Для запуска **Instrument I/O Assistant** поместите этот экспресс ВП на блок-диаграмму. Появится диалоговое окно настроек **Instrument I/O Assistant**. Если оно не появилось, следует нажать два раза на иконке **Instrument I/O Assistant**.



Для настройки экспресс-ВП **Instrument I/O Assistant** следует выполнить следующие пункты.

1. Выберите измерительный прибор. Приборы, настроенные в **MAX** отображены в выпадающем списке **Select an instrument**.
2. Выберите пункт **Code generation type**. Генерация кода **VISA** обеспечивает большую гибкость и модульность, чем генерация кода **GPIB**.
3. Выберите один из следующих шагов связи с помощью кнопки **Add Step**:

- **Query and Parse** – посылает устройству запрос типа ***IDN?** и анализирует возвращенную строку.
- **Write** – посылает команду измерительному прибору.
- **Read and Parse** – считывает и анализирует данные с измерительного прибора.

4. После составления требуемой последовательности операций нажмите кнопку **Run Sequence** для проверки созданной для экспресс-ВП последовательности.
5. Нажмите кнопку **OK** для выхода из диалогового окна настройки **Instrument I/O Assistant**.

LabVIEW добавит поля ввода/вывода в ВП **Instrument I/O Assistant** на блок-диаграмме в соответствии с данными, которые будут приходить с измерительного прибора.

Для просмотра кода, созданного **Instrument I/O Assistant**, щелкните правой кнопкой мыши по иконке **Instrument I/O Assistant** и выберите из контекстного меню пункт **Open Front Panel**. Эта опция преобразует экспресс ВП в подпрограмму ВП. Для просмотра созданного кода перейдите на блок-диаграмму.

Примечание После того, как экспресс ВП был преобразован в подпрограмму ВП, преобразовать его назад невозможно.

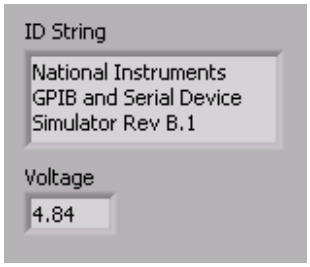
Упражнение 11-2 Использование помощника Instrument I/O Assistant

Цель: Создать ВП, который использует Instrument I/O Assistant для осуществления соединения по GPIB-интерфейсу.

Выполните следующие пункты для создания ВП, который собирает данные, поступающие от **NI Instrument Simulator**.

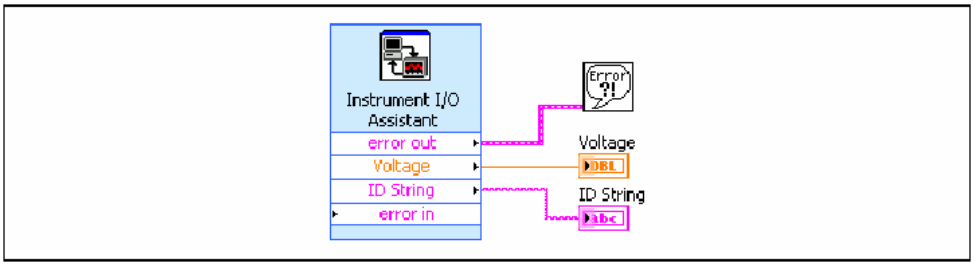
Лицевая панель

1. Откройте новый ВП.
2. После создания блок-диаграммы лицевая панель примет следующий вид.



Блок-диаграмма

3. Постройте блок-диаграмму, показанную ниже.



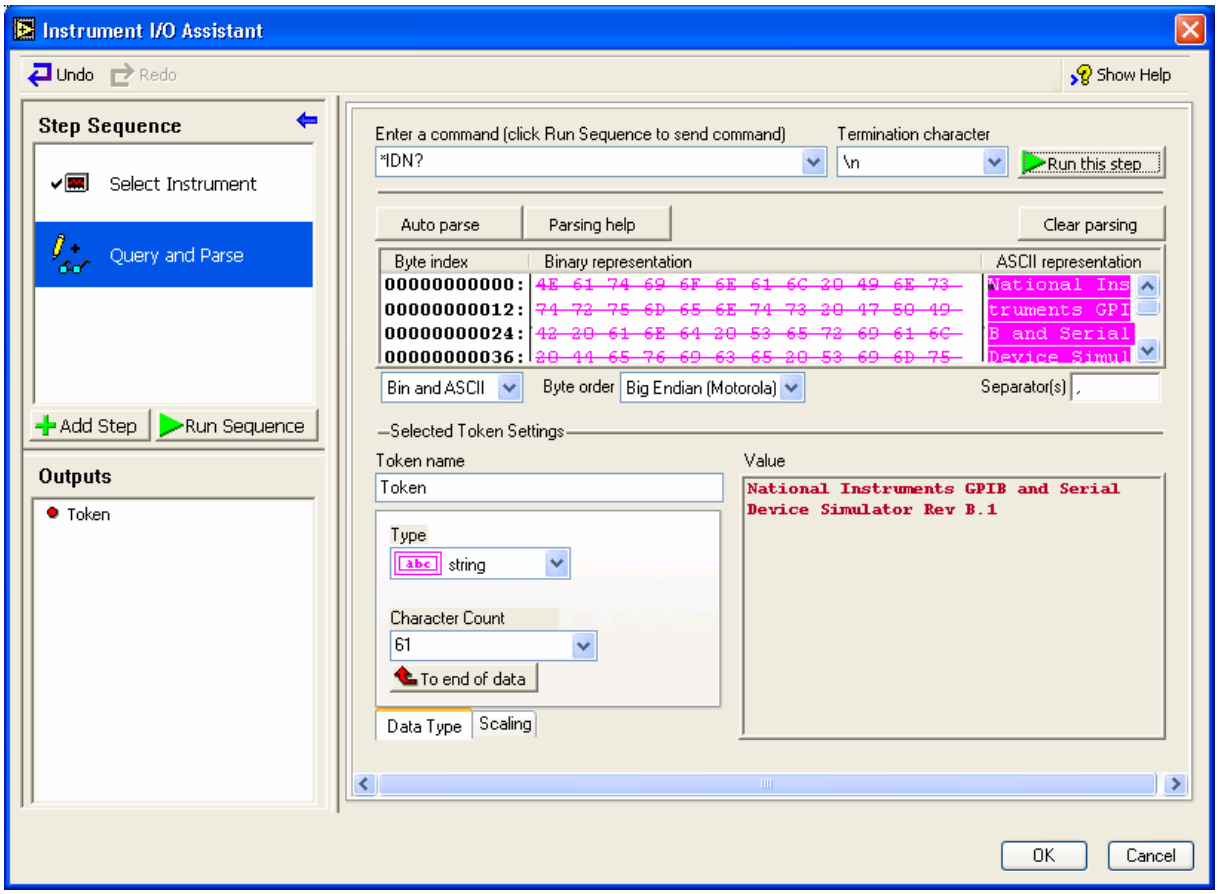
Поместите на блок-диаграмму экспресс-ВП **Instrument I/O Assistant**, расположенный в палитре **Functions»Express»Input**. Выполните следующие пункты для настройки экспресс-ВП в диалоговом окне настроек.

a. Выберите **devsim** из выпадающего меню **Select an Instrument**. Выберите пункт **VISA Code Generation** из выпадающего меню **Code Generation Type**.

b. Нажмите на кнопку **Add Step**. Нажмите на кнопку **Query and Parse** для записи и чтения из **Instrument Simulator**.

c. Напечатайте ***IDN?** в качестве команды, выберите **\n** в качестве **Termination Character** и нажмите на кнопку **Run this Step**. Если в нижней части окна не появилось предупреждения об ошибке, то этот шаг выполнился успешно.

d. Для анализа полученных данных нажмите на кнопку **Auto Parse**.



Заметьте, что в левой части диалогового окна в поле **Outputs** появился элемент **token**. Это строка, вернувшаяся после идентификационного запроса. Измените имя **token**, напечатав **ID String** в текстовом поле **Token name**.

e. Нажмите на кнопку **Add Button**. Нажмите на кнопку **Query and Parse**. Введите в качестве команды **MEAS:DC?** и нажмите на кнопку **Run this Step**.

f. Для анализа полученных данных следует нажать кнопку **Auto Parse**. Возвращенные данные являются произвольным числом. Измените имя **token** на **Voltage** в текстовом поле **Token name**.

g. Нажмите на кнопку **OK** для выхода из **I/O Assistant** и возврата на блок-диаграмму.

Щелкните правой кнопкой мыши по полю вывода **ID String** и выберите пункт контекстного меню **Create»Indicator**.

Щелкните правой кнопкой мыши по полю вывода **Voltage** и выберите пункт контекстного меню **Create»Indicator**.

Соедините поле вывода **Error Out** с ВП **Simple Error Handler VI**.

4. Перейдите на лицевую панель и запустите ВП. При необходимости измените размер строкового элемента отображения.

5. Сохраните ВП под именем *Чтение данных с прибора.vi*
6. Нажмите правой кнопкой мыши на **I/O Assistant** и выберите пункт **Show Front Panel**. Нажмите на кнопку OK в ответ на вопрос, хотите ли вы преобразовать в подпрограмму ВП.
7. Просмотрите код, сгенерированный **I/O Assistant**. Где находится команда ***IDN?**, записанная в **Instrument Simulator**? Где происходит считывание значения напряжения?
8. Выберите пункт меню **File»Exit** для выхода из подпрограммы ВП. Изменения не сохраняйте.

Конец упражнения 11-2

D. Архитектура программного обеспечения виртуальных интерфейсов (VISA)

Virtual Instrument Software Architecture – (VISA) – это низкоуровневые функции, которые используются в Виртуальных Приборах драйверов измерительных приборов для связи с программными драйверами.

Введение

В 1993 году компания National Instruments подписала соглашение с компаниями GenRad, Racal Instruments, Tektronix и Wavetek о поддержке единого стандарта **VXI*plug&play***. Цель альянса – сделать программное обеспечение для систем **VXI** совместимым независимо от производителя и сократить время разработки приложений.

Для этих целей был разработан единый стандарт для драйверов интерфейса и интерфейса ввода/вывода. Термин **VXI*plug&play*** охватывает как аппаратную, так и программную стандартизацию.

В попытке стандартизации программных средств члены альянса **VXI*plug&play*** выработали следующий блок требований:

- Максимальная легкость в использовании и исполнении.
- Долговременная поддержка совместимости базовых компонентов.
- Открытая архитектура для других фирм.
- Максимальная многоплатформенная совместимость.
- Максимальная расширяемость и модульность архитектуры.
- Максимальная многократность использования программных средств.
- Стандартизация использования программных элементов в системе.
- Рассмотрение драйверов интерфейса как части интерфейса.
- Адаптация к уже утвержденным стандартам.
- Максимальная совместная поддержка пользователей.

VISA – это язык программирования ввода/вывода **VXI*plug&play***, который явился результатом усилий альянса по стандартизации. **VISA** сам по себе не обеспечивает программную совместимость интерфейсов измерительных приборов. Это высокоуровневый блок **API,** который вызывает низкоуровневые функции драйвера. **VISA** позволяет управлять **VXI, GPIB**, последовательным портом и другими интерфейсами, основанными на компьютере. **VISA** выполняет соответствующие вызовы функции драйвера в зависимости от типа используемого интерфейса. В случае возникновения проблем отладки **VISA** необходимо вспомнить об этой иерархии. Возникновение очевидных проблем использования **VISA**, как правило, всегда связано с проблемами драйверов, функции которых **VISA** вызывает.

В среде LabVIEW **VISA** является единственной библиотекой функций, которая используется для связи с **GPIB**, последовательным портом, **VXI** и измерительными приборами, соединенными с компьютером. Нет необходимости использовать разные палитры ввода/вывода для программирования измерительного прибора. Например, некоторые измерительные приборы предоставляют выбор типа интерфейса. Драйвер, написанный с помощью функций, размещенных в палитре **Functions»Instrument I/O»GPIB**, не сможет работать с интерфейсом последовательного порта. **VISA** решает эти проблемы предоставлением единого набора функций для работы с интерфейсами любого типа. Поэтому все драйверы интерфейса в LabVIEW используют **VISA** как язык ввода/вывода.

Терминология программирования VISA

Функции, применяемые к ресурсам, называются операциями. Ресурсы имеют свойства или атрибуты, содержащие информацию, относящуюся к ресурсам. При программировании с использованием **VISA** употребляется терминология, похожая на терминологию Виртуальных Приборов драйверов измерительных приборов:

- **Resource** – любой устройство системы, включая последовательный и параллельный порты.
- **Session** – для связи с ресурсом необходимо открыть **VISA-**сессию что эквивалентно открытию канала связи. При открытии сессии для ресурса среда LabVIEW возвращает номер сессии **VISA**, который является уникальным логическим идентификатором устройства. Этот номер сессии необходимо использовать во всех последующих функциях **VISA**.
- **Instrument Descriptor** – точное имя ресурса. Дескриптор указывает тип интерфейса (**GPIB, VXI, ASRL**), адрес устройства (логический или первичный) и тип **VISA**-сессии (**INSTR** или **Event**).

Дескриптор интерфейса похож на телефонный номер, где ресурс – это абонент, а сессия – это телефонная линия. Каждый вызов использует свою собственную линию, и при пересечении таких линий происходит ошибка. Следующая таблица показывает правильный синтаксис для описания дескриптора интерфейса.

Интерфейс	**Синтаксис**
Асинхронный последовательный порт (Asynchronous serial)	`ASRL[`*`board`*`][::INSTR]`
GPIB	`GPIB[`*`board`*`]::`*`primary address`*`[::`*`secondary address`*`][::INSTR]`
VXI интерфейс через встроенный или MXIbus контроллер	`VXI[`*`board`*`]::`*`VXI logical address`*`[::INSTR]`
GPIB-VXI контроллер	`GPIB-VXI[`*`board`*`][::`*`GPIB-VXI primary address`*`]::`*`VXI logical address`*`[::INSTR]`

Можно использовать имя, заданное в конфигурационной утилите **MAX**, вместо дескриптора. **(MacOS)** Отредактируйте файл *visaconf.ini* для создания имени VISA. **(UNIX)** Используйте утилиту **visaconf**.

В случае, когда **Instrument I/O Assistant** не используется для автоматического создания кода, можно самостоятельно написать ВП для связи с измерительным прибором. Наиболее часто используемые функции **VISA** – это функции **VISA Write** и **VISA Read**. Большинству измерительных приборов необходимо получить информацию в виде запроса или команды до того, как они начнут передавать информацию. Таким образом, после функции **VISA Write** обычно стоит функция **VISA Read**.

Упражнение 11-3 Программирование с помощью VISA

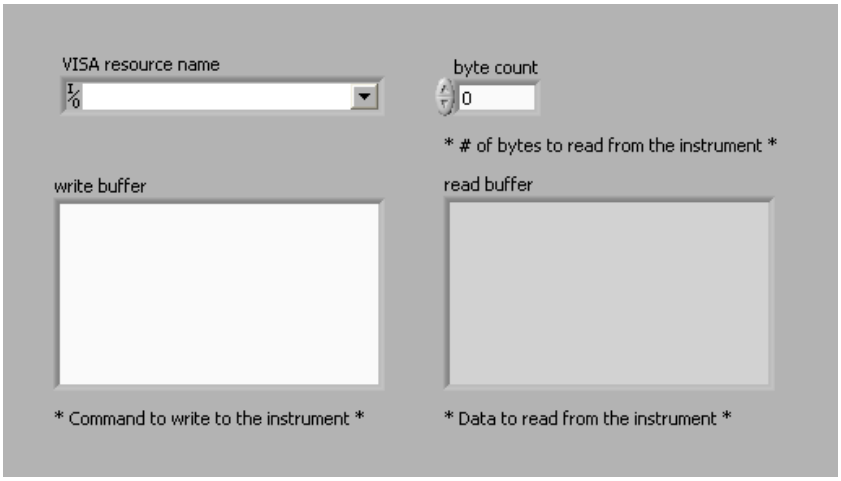
Цель: Создать ВП, который с помощью функций VISA читает и записывает информацию в NI Instrument Simulator.

Выполните следующие пункты для создания ВП, который собирает данные от **NI Instrument Simulator** с помощью функций **VISA**.

1. Убедитесь, что **Instrument Simulator** включен и соединен с интерфейсом **GPIB**.

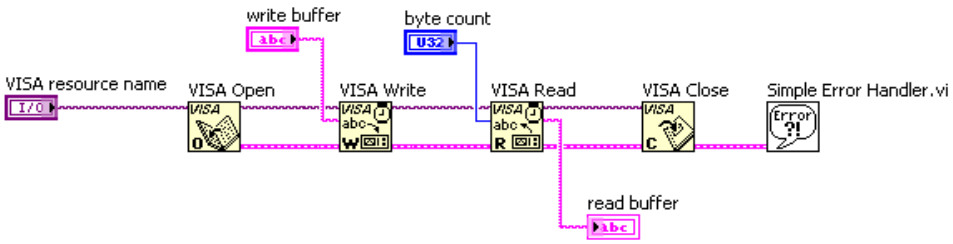
Лицевая панель

2. Откройте новый ВП. После построения блок-диаграммы лицевая панель примет следующий вид.



Блок-диаграмма

3. Постройте блок-диаграмму, показанную ниже.



Поместите на блок-диаграмму функцию **VISA Open**, расположенную в палитре **Functions»Instrument I/O»VISA»VISA Advanced**. Эта функция открывает **VISA**-сессию с измерительным прибором. Щелкните правой кнопкой мыши по входному полю **resource name** и выберите пункт контекстного меню **Create»Control**.

Поместите на блок-диаграмму функцию **VISA Write**, расположенную в палитре **Functions»Instrument I/O»VISA**. Эта функция записывает в

измерительный прибор строку. Щелкните правой кнопкой мыши по входному полю **write buffer** и выберите пункт контекстного меню **Create»Control**.

Поместите на блок-диаграмму функцию **VISA Read**, расположенную в палитре **Functions»Instrument I/O»VISA**. Эта функция считывает данные с измерительного прибора. Щелкните правой кнопкой мыши по терминалу **byte count** и выберите из контекстного меню пункт **Create»Control**. Щелкните правой кнопкой мыши по терминалу **read buffer** и выберите из контекстного меню пункт **Create»Control**.

Поместить на блок-диаграмму функцию **VISA Close**, расположенную в палитре **Functions»Instrument I/O»VISA»VISA Advanced**. Эта функция закрывает **VISA**-сессию с измерительным прибором и освобождает использованные системные ресурсы.

Поместите на блок-диаграмму ВП **Simple Error Handler VI**, расположенный в палитре **Functions»Programming»Dialog & User Interface**. Этот ВП проверяет состояние ошибки и открывает диалоговое окно с информацией об ошибке в случае ее возникновения.

4. Сохраните ВП под именем *Запись и чтение VISA.vi*

5. Перейдите на лицевую панель. Введите **devsim** в строку **resource name** и установите значение счетчика **byte count** равным **200** для того, чтобы считались все данные. Введите команду ***IDN?** в поле **write buffer** и запустите ВП.

6. В верхней части **Instrument Simulator** перечислены другие команды, воспринимаемые этим прибором. Попробуйте использовать другие команды в этом ВП.

7. По окончании закройте ВП.

Конец упражнения 11-3

E. Драйверы измерительных приборов

Драйвер измерительного прибора – это набор модульных программных функций, которые используют команды или протокол измерительного прибора для проведения стандартных операций. Драйвер измерительного прибора также вызывает нужные функции и Виртуальные Приборы. Драйверы устройств из библиотеки LabVIEW устраняют необходимость изучать сложные низкоуровневые команды программирования для каждого отдельного измерительного прибора.

Библиотека LabVIEW драйверов устройств содержит драйвера для множества программируемых приборов, использующих **GPIB**, **VXI**, **PXL** интерфейсы или последовательный порт.

Драйверы измерительных приборов принимают, анализируют и масштабируют строки отклика измерительных приборов в данные, используемые в приложениях. Драйверы измерительных приборов упрощают приложения диагностики, т.к. драйверы содержат в одной библиотеке все вводы/выводы прибора отдельно от какого-либо другого кода. При замене оборудования изменить приложения становится проще, поскольку весь код, относящийся к определенному оборудованию, заключен в драйверах.

Библиотека LabVIEW драйверов измерительных приборов расположена на LabVIEW CD. Драйверы также можно скачать с сайта компании National Instruments по адресу *www.ni.com/idnet*. Для установки драйверов следует их разархивировать поместить полученную директорию в *\labview\instr.lib*. При последующем запуске LabVIEW Виртуальные Приборы драйверов измерительных приборов будут находиться на палитре **Functions»Instrument I/O»Instrument Drivers**.

Getting Started Example (Начальный пример)

Все драйверы измерительных приборов содержат пример, который может быть использован для проверки связи с прибором. Этот пример обычно называется **Getting Started Example**. Укажите верный **GPIB**-адрес (или имя ресурса **VISA**) измерительного прибора, как он был настроен в **MAX**.

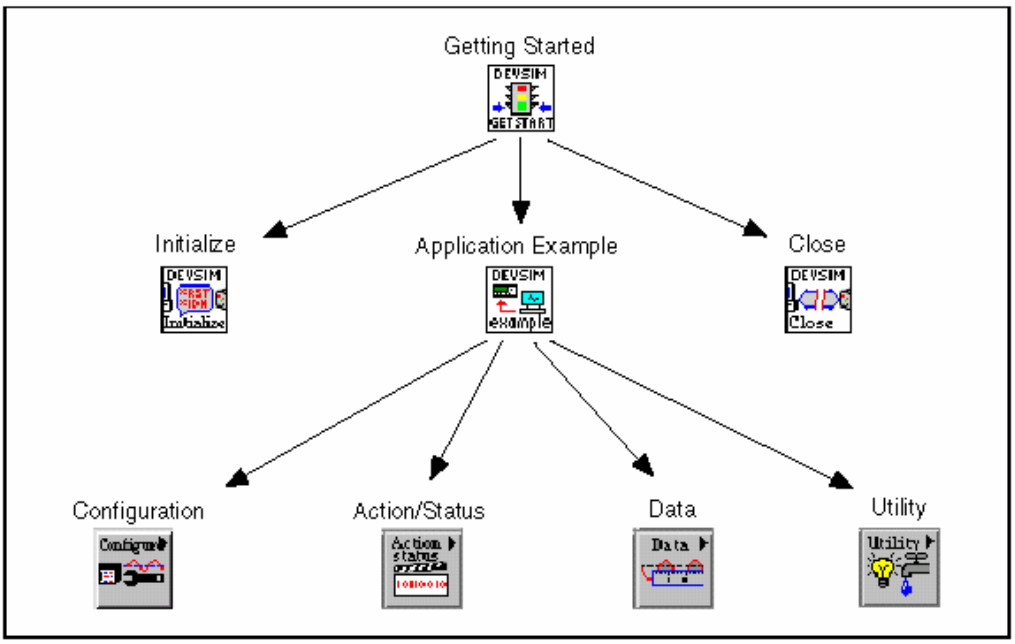
F. Использование ВП драйвера устройства

Драйверы создаются под определенные измерительные приборы и делают необязательным для пользователя точное знание команд стандарта IEEE 488.2, которые требуются для управления прибором.

Компоненты драйвера устройства

Все драйвера устройств в библиотеке имеют одинаковую базовую иерархию. Иерархия, последовательность использования ВП и обработка ошибок устроены так же, как и в ВП сбора данных, ВП файлового ввода/вывода, TCP/IP и т.д. Для получения информации об обработке ошибок следует обратиться к разделу C *Функции и ВП файлового ввода/вывода* Урока 7 *Строки и файловый ввод/вывод*.

На рисунке показана иерархия драйвера устройства.



Высокоуровневые функции созданы на основе низкоуровневых функций. Низкоуровневые функции обеспечивают расширенный контроль над интерфейсом, однако высокоуровневые функции просты в использовании и имеют удобные лицевые панели. Драйвера устройств содержат Виртуальные Приборы следующих категорий:

- **Initialize -** инициализирует каналы связи с устройством. Этот ВП также осуществляет операции идентификационного запроса, операции сброса или любые другие операции по сбросу значений установок устройства к значениям по умолчанию.
- **Configuration -** настраивает устройство для выполнения операций, таких как установка частоты триггера.
- **Action/Status –** включает в себя два типа ВП. ВП типа **Action VIs**

осуществляют начало и остановку тестирования и измерительных операций. ВП типа **Status VIs** предназначены для получения текущего состояния устройства или статуса выполняемой операции. Например, ВП типа **Action VIs** – ВП **Acquire Single Shot**. ВП типа **Status VIs**– ВП **Query Transfer Pending**.

- **Data –** выполняет операции передачи данных к/из измерительного прибора, такие как чтение измеренной осциллограммы (оцифрованный сигнал) из устройства или загрузка осциллограммы в устройство**.**
- **Utility –** выполняет широкий класс функций, таких как сброс, самотестирование, запрос ошибки или проверка состояния запроса.
- **Close –** разрывает канал связи с устройством и освобождает использованные для него ресурсы.

Все драйверы интерфейсов от компании National Instruments включают в себя следующие функции: инициализация, закрытие, сброс, самотестирование, проверка состояния запроса, обнаружение ошибки запроса и сообщение об ошибке.

Примеры приложений

Среда LabVIEW также включает примеры приложений, которые показывают, как использовать Виртуальные Приборы – компоненты драйвера для решения стандартных задач. Как правило, к этим задачам относятся: настройка, переключение и получение результатов измерения из измерительного прибора. Эти приложения не выполняют операции инициализации или закрытия драйвера устройства. Кроме того, эти ВП не претендуют на вариант полного приложения, а только демонстрируют возможности драйвера интерфейса и являются введением в разработку собственных драйверов интерфейса.

Поля ввода/вывода Виртуальных Приборов драйвера устройства

Так как иерархия ВП всех драйверов устройств одинакова, они имеют похожие поля ввода/вывода данных.

Имя ресурса или дескриптор устройства

При инициализации канала связи с устройством необходимо знать дескриптор этого устройства (**instrument descriptor**) или имя ресурса (**resource name**). Ресурс – это интерфейс или измерительный прибор, а идентификатор ресурса – точное имя и положение ресурса в данном формате:

Interface Type[board index]::Address::INSTR

Необязательные параметры показаны в квадратных скобках [ ]. Например, GPIB::2::INSTR – дескриптор GPIB прибора по адресу 2.

ВП **VISA resource name control**, размещенный в палитре **Controls»Modern»I/O**, аналогичен ВП **DAQ channel name control**, но ориентирован на управление измерительными приборами. Для получения

дополнительной информации о **VISA** смотрите раздел E, *VISA* этого урока.

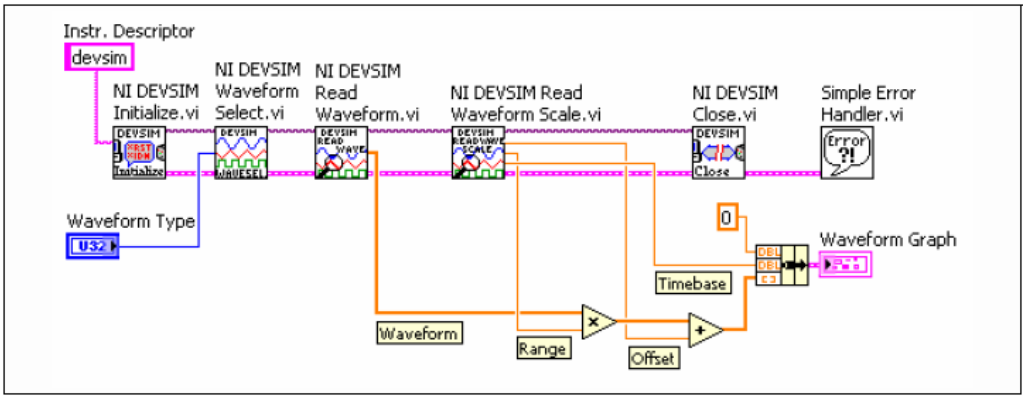
Для получения информации о доступных ресурсах и адресах устройств можно использовать конфигурационную утилиту **MAX**, как было показано в упражнении 11-1. В этом упражнении было определено имя **VISA (VISA alias).** Наличие имени упрощает связь с устройством, потому что, работая с именем, не нужно запоминать тип интерфейса и адрес устройства. Можно ввести имя вместо дескриптора устройства в поле имени ресурса **VISA**, например, можно ввести строку **devsim** вместо строки **GPIB::2::INSTR.**

Сессии VISA

После инициализации устройства ВП **Initialize VI** возвращает номер сессии **VISA**. Сессия **VISA** является связующим звеном с ресурсом, таким как измерительный прибор. Нет необходимости отображать это значение, однако каждый раз, когда происходит связь с устройством, необходимо соединять поля **VISA session** Виртуальных Приборов драйвера устройства. После завершения связи с устройством необходимо использовать ВП **Close VI** для закрытия всех ссылок и ресурсов, использованных во время связи.

Пример приложения драйвера интерфейса

Блок-диаграмма, показанная на рисунке, инициализирует измерительный прибор под именем **devsim**. С помощью ВП настройки выбирается тип осциллограммы, затем используются два ВП: для чтения осциллограммы и ее масштабирования. После этого производится закрытие измерительного прибора и проверка статуса ошибки. Каждое приложение, которое использует драйвер интерфейса, применяет подобную последовательность выполнения событий.



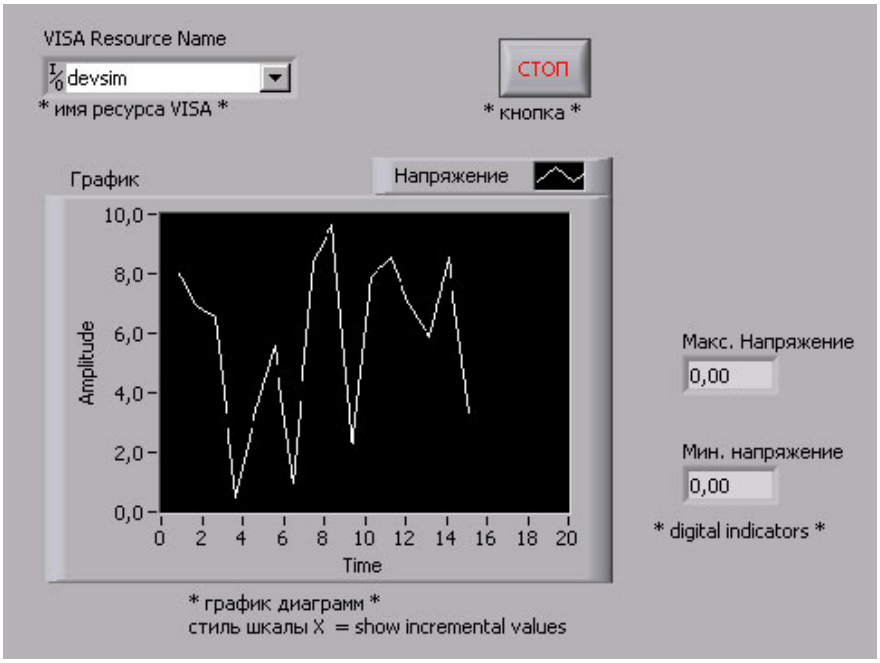
Упражнение 11-4 ВП Монитор напряжения

Цель: Построить ВП, который использует Виртуальные Приборы драйвера устройства DevSim для сбора и отображения значений напряжения

Выполните следующие шаги для создания ВП, который считывает значения напряжения постоянного тока с ВП **NI Instrument Simulator** с частотой 1 Гц и отображает их на графике Диаграмм, пока не будет нажата кнопка **СТОП.** Каждое считанное значение напряжения ВП сравнивает с предыдущими минимальными и максимальными значениями. ВП непрерывно подсчитывает и отображает минимальные и максимальные значения.

Лицевая панель

1. Выберите в главном меню пункт **File»New**, затем **Template»Frameworks»Single Loop Application** для открытия шаблона ВП **Single Loop Application VI**.
2. Создайте лицевую панель, показанную на рисунке.

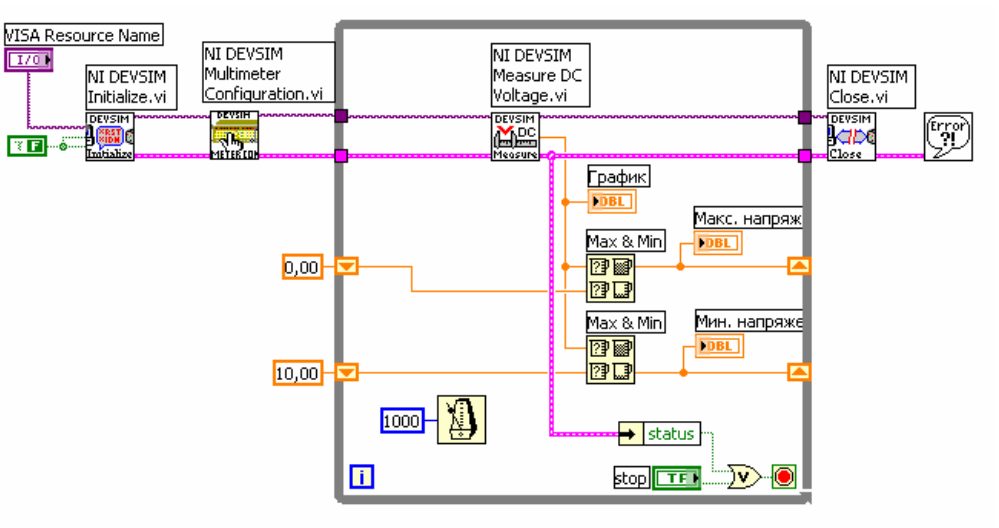


Поместите элемент управления **VISA resource name**, размещенный в палитре **Controls»Modern»I/O**.

Отредактируйте масштаб оси *x* на графике Диаграмм на диапазон от **0** до **20**.

Блок-диаграмма

3. Создайте блок-диаграмму, показанную на рисунке:



Создайте два сдвиговых регистра, щелкнув правой кнопкой мыши по правой и левой границам структуры цикла и выбрав пункт **Add Shift Register** из контекстного меню.

Поместите на блок-диаграмму ВП **NI DEVSIM Initialize VI**, размещенный в палитре **Functions»Instrument I/O»Instrument Drivers»NI Device Simulator**. Этот ВП открывает связь между средой LabVIEW и ВП **NI Instrument Simulator**.

k. Щелчком правой кнопкой мыши по полю **ID Query** вызовите всплывающее меню и выберите в нем пункт **Create»Constant**. С помощью инструмента УПРАВЛЕНИЕ измените значение константы на FALSE**.**

l. Соедините константу с входным полем **Reset** .

Поместите на блок-диаграмму ВП **NI DEVSIM Multimeter Configuration VI**, размещенный в палитре **Functions»Instrument I/O»Instrument Drivers»NI Device Simulator»Configuration**. Этот ВП устанавливает диапазон изменения напряжения, генерируемого ВП **NI Instrument Simulator**. По умолчанию диапазон значений установлен от 0,0 до 10,0 Вольт.

Поместите на блок-диаграмму ВП **NI DEVSIM Measure DC Voltage VI**, размещенный в палитре **Functions»Instrument I/O»Instrument Drivers»NI Device Simulator»Data**. Этот ВП возвращает имитированное значение измеренного напряжения с ВП **NI Instrument Simulator.**

Поместите на блок-диаграмму ВП **NI DEVSIM Close VI**, размещенный в палитре **Functions»Instrument I/O»Instrument Drivers»NI Device Simulator**. Этот ВП завершает связь между средой LabVIEW и **NI Instrument Simulator**.

Поместите на блок-диаграмму функцию **Max & Min**, размещенную в палитре **Functions»Programming»Comparison**. Используйте две копии этой функции для сравнения текущего значения с минимальным и максимальным значениями и последующим сохранением результата в терминале регистра сдвига.

Поместите на блок-диаграмму ВП **Simple Error Handler VI**, размещенный в палитре **Functions»Programming»Dialog & User Interface**. Этот ВП отображает диалоговое окно с информацией об ошибке в случае ее возникновения.

Поместите на блок-диаграмму функцию **Unbundle by Name**, размещенную в палитре **Functions»Programming»Cluster & Variant**. Эта функция возвращает значение **status** кластера ошибок**.**

Поместите на блок-диаграмму функцию **Or**, размещенную в палитре **Functions»Programming»Boolean**. Эта функция управляет условием завершения цикла **While**. Если произошла ошибка или нажата клавиша **СТОП**, выполнение цикла завершается.

Установите экспресс-ВП **Time Delay Express VI** на ожидание в течение 1 секунды.

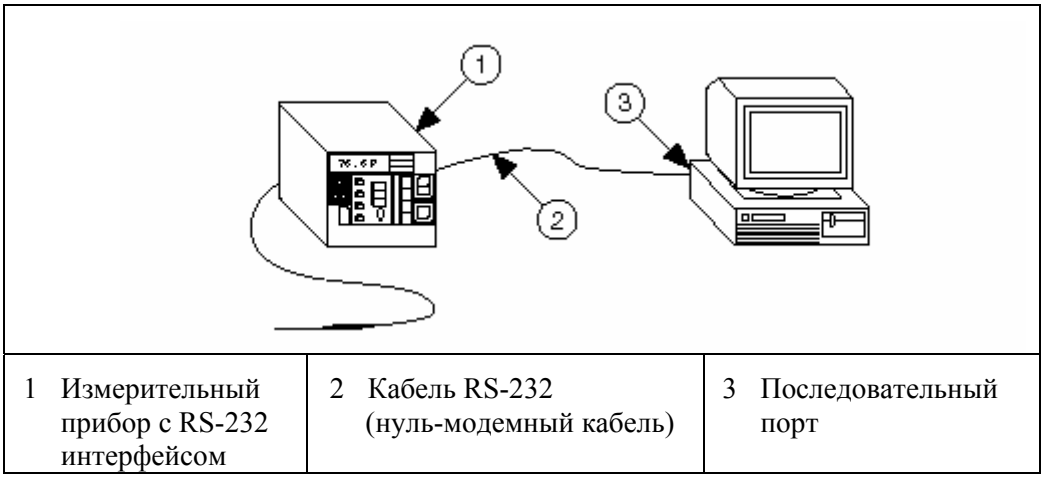
Примечание Нет необходимости подсоединять все поля ввода/вывода каждого узла. Нужно соединять только обязательные для использования поля каждого узла, такие как: **instrument descriptor**, **VISA session** и терминалы входа/выхода ошибки.

4. Сохраните ВП под именем файла *Монитор напряжения.vi*
5. Убедитесь, что ВП **NI Instrument Simulator** включен.
6. Запустите ВП. Светодиоды **Listen** (слушатель) и **Talk** (передатчик) будут мигать с частотой 1 Гц, что сигнализирует о соединении приложения LabVIEW с прибором. На диаграмме будет отображаться имитируемое напряжение, а также минимальное и максимальное значения.
7. Остановите и закройте ВП.

Конец упражнения 11-4

G. Последовательная связь

Последовательная связь – наиболее распространенное средство передачи данных между компьютером и периферийными устройствами, такими как программируемые измерительные приборы или даже другие компьютеры. Последовательная связь использует передатчик для передачи данных последовательно бит за битом по однопроводной линии к приемнику. Этот метод можно использовать, когда скорость передачи мала или необходимо передать данные на большие расстояния. Распространенность последовательной передачи определяется тем, что большинство компьютеров имеют один или более последовательных портов и не требует никаких дополнительных аппаратных средств, кроме нуль-модемного кабеля.

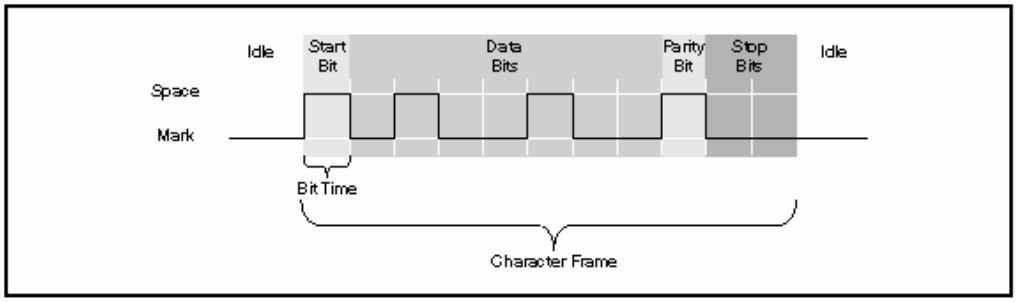


1 Измерительный прибор с RS-232 интерфейсом	2 Кабель RS-232 (нуль-модемный кабель)	3 Последовательный порт

Для использования последовательной связи требуется задание четырех параметров:

- скорость передачи (baud rate)
- количество битов данных, кодирующих передаваемый символ
- наличие бита четности (parity bit)
- количество стоп-битов (stop bits)

Каждый передаваемый символ упаковывается в кадр символа, который состоит из одиночного стартового бита (**start bit**), за ним следуют биты данных (**data bits**), бит четности (если установлен), и заданное количество стоповых битов. На рисунке показан кадр символа буквы **m**.



Скорость передачи показывает, как быстро поток данных перемещается

между устройствами, использующими последовательный порт. Последовательный порт **RS-232** использует только два уровня напряжения, которые называются **MARK** (ЛОГИЧЕСКАЯ ЕДИНИЦА) и **SPACE** (ЛОГИЧЕСКИЙ НОЛЬ). В такой двухуровневой схеме передачи скорость передачи равна максимальному количеству информационных битов, включая управляющие биты, в секунду.

MARK – отрицательное значение напряжения, **SPACE** – положительное. На предыдущем рисунке было показано, как идеальный сигнал выглядит на осциллографе. Таблица истинности для RS-232 выглядит следующим образом:

Уровень сигнала > +3 V = 0
Уровень сигнала < -3 V = 1

Выходной уровень сигнала обычно находится в диапазоне +12 – -12 В. «Мертвая» зона (+3 – -3 В) необходима для отсечки шума коммуникационной линии.

Стартовый бит сигнала является началом каждого кадра символа. Он передается изменением уровня от отрицательного (**MARK**) напряжения к положительному (**SPACE**). Его длительность характеризует скорость передачи. Если устройство передает на скорости 9600 бод, длительность стартового бита и всех последующих бит составляет 0,104 мс. Полный кадр символа (11 бит) передается примерно за 1,146 мс.

Биты данных передаются в обратном порядке с использованием отрицательной логики. Сначала передаются младшие биты (LSB – least significant bit), затем старшие биты (MSB – most significant bit). Для интерпретации битов данных в кадре символа необходимо читать кадр справа налево, причем **1** соответствует отрицательному значению напряжения, а **0** - положительному. На рисунке биты символа можно прочесть как **1101101** в двоичном формате или **6D** в шестнадцатеричном формате. Это значение по таблице ASCII соответствует символу **m**.

Если установлен бит четности, то он следует за битами данных в кадре символа. Бит четности также передается в инвертированной логике: **1** - отрицательное значение напряжения; **0** - положительное. Бит четности является простым средством обнаружения ошибок передачи. Заранее определяется, какой будет четность передачи – четной или нечетной. Если выбрана нечетная схема, то передатчик устанавливает бит четности так, чтобы сделать количество бит данных и бита четности нечетным. Приведенная в примере операция передачи данных использует нечетную схему. Количество битов данных равно 5, т.е. уже нечетное число, поэтому бит четности равен 0.

Конец кадра символа состоит из 1, 1,5 или 2 стоп битов. Эти биты всегда представлены отрицательным уровнем напряжения. Если передачи символов больше не происходит, то линия связи остается в отрицательном уровне напряжения (**MARK**). Изменение напряжения к положительному уровню (**SPACE**) показывает начало передачи следующего кадра символа.

Какая максимальная скорость передачи?

Зная структуру кадра символа и значение скорости передачи, можно подсчитать максимальную частоту передачи в символах за 1 с для заданных параметрах связи. Эта частота является отношением скорости передачи к общему количеству битов в кадре символа. В предыдущем примере общее количество битов в кадре символа было равным 11. Если скорость передачи установить равной 9600 бот, то частота передачи будет равна 9600/11 = 872 символов в секунду. Полученное значение является максимальной частотой передачи символов. Аппаратные средства интерфейса могут не достигнуть этого значения по разным причинам.

Обзор аппаратных средств

Существует много различных стандартов последовательной связи, наиболее распространенные из которых перечислены ниже.

RS-232

Стандарт RS-232 разработан **Electronic Industries Association (EIA)** и другими заинтересованными организациями в целях описания последовательного интерфейса между **Data Terminal Equipment (DTE)** и **Data Communications Equipment (DCE)**. Стандарт RS-232 включает описание характеристики электрического сигнала (уровня напряжения), механические спецификации (разъемы), функциональное описание линий (функции каждого электрического сигнала) и несколько способов соединения терминала с модемом. Наиболее часто встречающееся расширение этого стандарта называется RS-232C. Этот стандарт описывает (с различным уровнем надежности) использование последовательного порта для связи между компьютером и принтером, модемом и другими периферийными устройствами. Стандарту RS-232 соответствуют последовательные порты IBM-совместимых ПК.

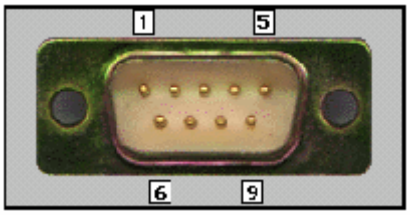
RS-449, RS-422, RS-423

Стандарты RS-449, RS-422 и RS-423 являются дополнительными стандартами **EIA** последовательной связи. Эти стандарты являются производными от RS-232. Стандарт RS-449 был издан в 1975 году и предназначался для замены стандарта RS-232, однако его поддержали мало производителей. Стандарт RS-449 включает в себя две разновидности, названные RS-422 и RS-423. Если стандарт RS-232 описывает одноуровневую передачу, при которой уровень сигнала измеряется относительно общего заземления, то стандарт RS-422, напротив, описывает уровни сигналов друг относительно друга (относительная передача). Если в стандарте RS-232C приемник оценивает, насколько значение отрицательного уровня напряжения относительно земли достаточно, чтобы быть логической **1**, то в RS-422 приемник оценивает отрицательный уровень сигнала относительно другого сигнала. Это делает стандарт RS-422 более защищенным от помех, нечувствительным к интерференции и ориентированным на

большие расстояния передачи. Последовательные порты компьютера **Macintosh** соответствуют стандарту RS-422, стандарт RS-423 является модернизацией стандарта RS-422 и предназначен для совместимости со стандартом RS-232. С помощью специального внешнего кабеля интерфейс RS-422 преобразуется к RS-423 и позволяет осуществлять связь с интерфейсом RS-422 на расстоянии более 15 м.

Кабель RS-232

Устройства, использующие последовательный порт, можно разделить на две категории: **DCE** и **DTE**. Категория **DCE** – это устройства, такие как модем, плоттер и т.д. Категория **DTE** – это последовательные порты другого компьютера. Разъемы RS-232 последовательного порта существуют в двух вариантах: 25-ти и 9-ти штырьковые **(D-Type 25-pin connector** и **D-Type 9-pin connector).** Оба варианта на компьютере являются «папами» и требуют разъема типа «мама» на приборе. На рисунке показан внешний вид 9-тиштырькового разъема, ниже в таблице приведено функциональное описание контактов.



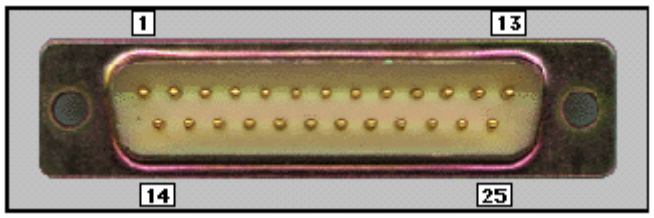
Функция (Function)	**Сигнал (Signal)**	**Номер штырька (PIN)**	**Категория DTE**	**Категория DCE**
Обмен данных (Data)	TxD RxD	3 2	Output Input	Input Output
Управление интерфейсом (Handshake)	RTS CTS DSR DCD DTR	7 8 6 1 4	Output Input Input Input Output	Input Output Output Output Input
Общий (Common)	Com	5	-	-
Другие (Other)	RI	9	Input	Output

Вариант 9-тиштырькового разъема (DB-9) обычно используется в лабораторном оборудовании небольших размеров. Он является компактной версией стандарта RS-232.

Примечание DB-9 используют для передачи и приема 3-й и 2-й штырьки соответственно. 25-тиштырьковый разъем (DB-25), напротив, использует для передачи и приема штырьки 2-й и 3-й. Необходимо обращать внимание на эти различия при определении категории прибора – **DTE** и **DCE**.

25-тиштырьковый разъем (DB-25) соответствует полной версии стандарта

RS-232. Внешний вид разъема показан на рисунке, а в таблице указано функциональное назначение выводов.



Function	Signal	PIN	DTE	DCE
Data	TxD	2	Output	Input
	RxD	3	Input	Output
Handshake	RTS	4	Output	Input
	CTS	5	Input	Output
	DSR	6	Input	Output
	DCD	8	Input	Output
	DTR	20	Output	Input
Common	Com	7	-	-

Обзор программных средств

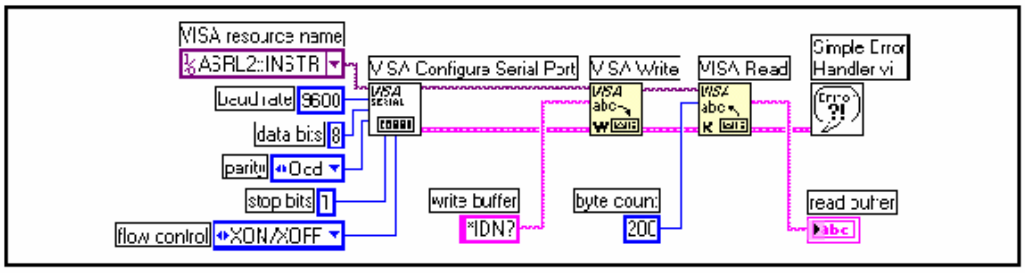
ВП и функции для работы с последовательным портом размещены на палитре **Functions»Instrument I/O»Serial**.

Функции **VISA** (**VISA Write** и **VISA Read**), которые использовались для связи по GPIB-интерфейсу, также могут быть использованы для управления последовательной связью, как и любым другим интерфейсом. Так как последовательный порт использует много параметров, помимо этих функций необходимо дополнительно использовать ВП **VISA Configure Serial Port VI**.

ВП **VISA Configure Serial Port VI** инициализирует последовательный порт, определенный полем **VISA resource name**, установленными значениями. Поле **timeout** устанавливает время простоя последовательного порта. Поля **baud rate**, **data bits**, **parity** и **flow control** определяют параметры настройки последовательного порта. На поля **error in** и **error out** подается кластер ошибок, содержащий информацию об ошибке.

Следующий пример показывает, как осуществляется посылка управляющей команды ***IDN?** измерительному прибору через последовательный порт COM2.

ВП **VISA Configure Serial Port VI** открывает связь через последовательный порт COM2 и устанавливает его параметры: 9600 бод, 8 бит данных, проверка на нечетность (**odd parity**), один стоп бит и программное управление передачей **XON/XOFF.** Функция **VISA Write** посылает управляющую команду. Функция **VISA Read** принимает 200 байт в буфер чтения, и ВП **Simple Error Handler VI** проверяет состояние кластера ошибок.



Примечание ВП и функции, размещенные на палитре **Functions»Instrument I/O»Serial**, можно использовать для связи через параллельный порт. В поле **VISA resource name** должен быть описан дескриптор параллельного порта LPT. Например, с помощью конфигурационной утилиты **MAX** можно назначить порту LPT1 имя ресурса **VISA** – ASRL10::INSTR.

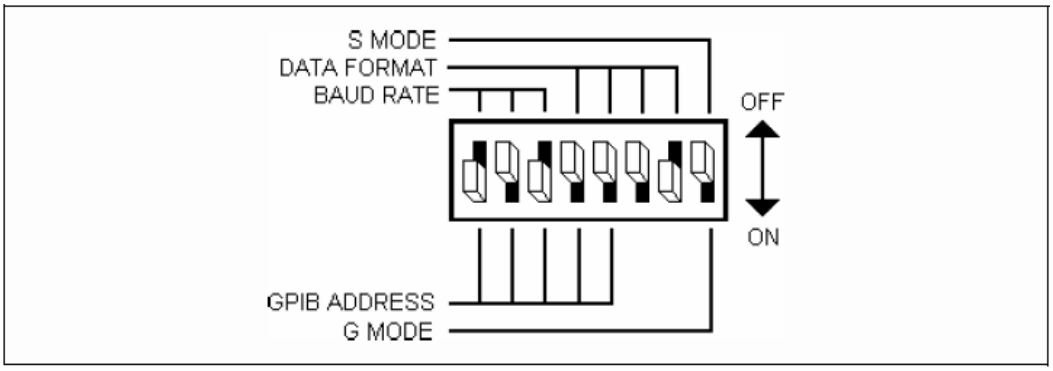
Упражнение 11-5 ВП Запись и чтение в последовательный порт

Цель: Построить ВП, который обеспечивает связь с RS-232 интерфе йсом

Выполните следующие шаги для создания ВП, который выполняет операцию связи с **NI Instrument Simulator**.

Имитатор NI Instrument Simulator

1. Выключите **NI Instrument Simulator** и установите переключатели установок последовательного порта, как показано на рисунке.



Положения переключателей на рисунке соответствуют следующим параметрам последовательного интерфейса:

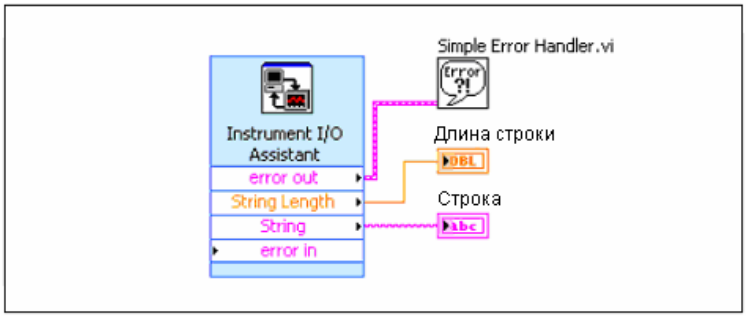
- Скорость передачи = 9600.
- Биты данных = 8.
- Четность – без проверки четности.
- Стоп биты = 1.
- Контроль потока данных – аппаратный (hardware handshaking).

Контроль (Handshaking) потока данных разделяется на аппаратный и программный. Программный контроль включает в себя выделение управляющих символов в потоке данных. Например, управляющие символы **XON** и **XOFF** обрамляют передаваемое сообщение. Аппаратный контроль потока данных заключается в использовании уровней напряжения на физических линиях. Линии **RTS** и **CTS** интерфейса RS-232 часто используются в качестве аппаратного контроля. Большинство лабораторного оборудования использует именно аппаратный контроль.

2. Проверьте физическое соединение кабеля имитатора **NI Instrument Simulator** с разъемом последовательного порта. Запомните номер порта.
3. Включите питание **NI Instrument Simulator**. Индикаторы-светодиоды **Power, Ready и Listen** загорятся, что показывает настройку прибора на режим последовательной связи.

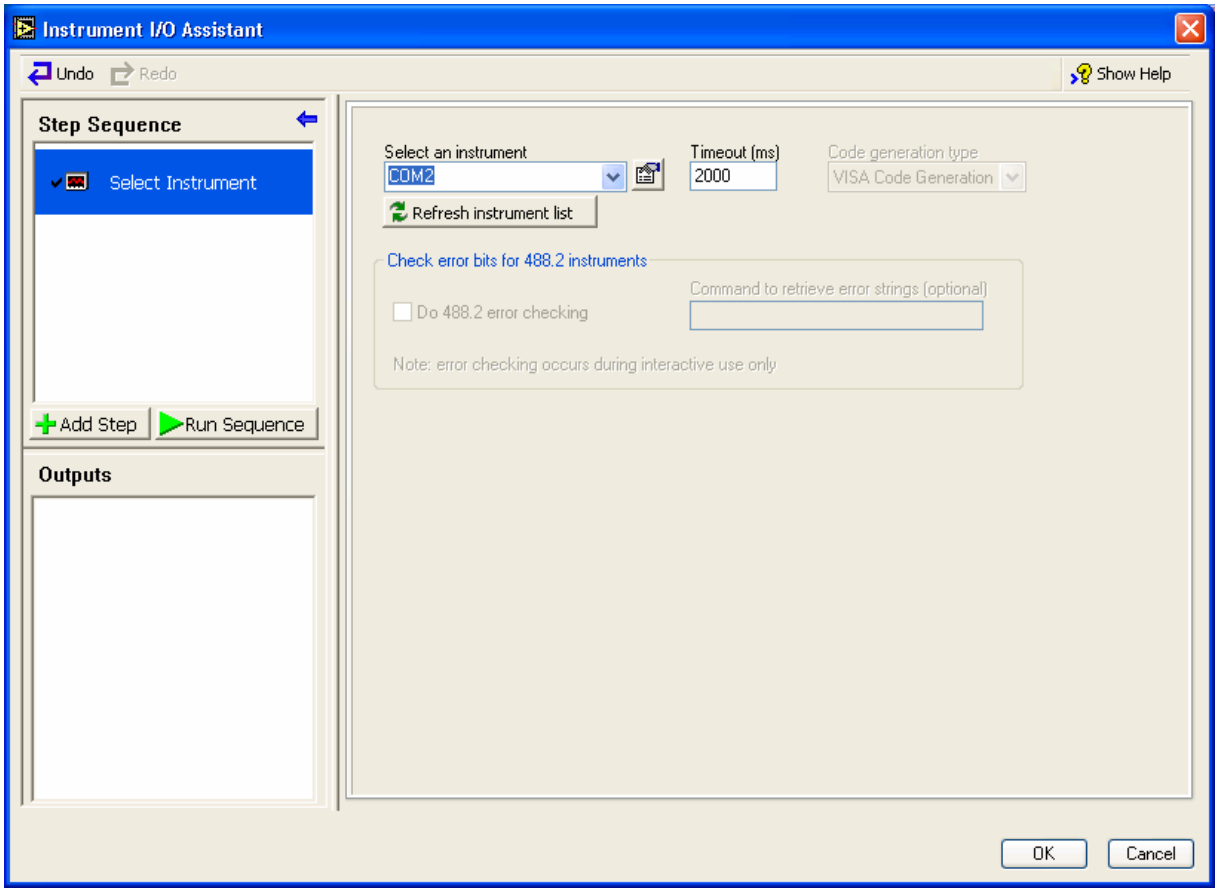
Блок-диаграмма

4. Откройте новый ВП и постройте блок-диаграмму, показанную на рисунке.



Поместите на блок-диаграмму экспресс-ВП **Instrument I/O Express VI**, расположенный в палитре **Functions»Instrument I/O**. Выполните следующие пункты для настройки экспресс-ВП в диалоговом окне **Instrument I/O Assistant**.

a. Выберите COM1 (или COM2 в зависимости от того, к какому порту подсоединен **NI Instrument Simulator**) в выпадающем меню **Select an instrument**.



b. Нажмите на кнопку **Add Step** и нажмите на кнопку **Query and Parse.** Напечатайте ***IDN?** в качестве команды, выберите **\n** в качестве **Termination Character**.

c. Нажмите на кнопку **Run this Step**, затем нажмите на кнопку

Auto Parse. Полученное значение соответствует размеру в байтах отклика на запрос.

d. Измените имя **token** на **Длина строки** в текстовом поле **Token name**.

e. Нажмите на кнопку **Add Step** и нажмите на кнопку **Query and Parse.** Напечатайте ***IDN?** в качестве команды, выберите **\n** в качестве **Termination Character**.

f. Нажмите на кнопку **Run this Step** еще раз, затем нажмите на кнопку **Auto Parse**. Полученное значение является строкой идентификации **NI Instrument Simulator**.

g. Измените имя **token** на **String** в текстовом поле **Token name**.

h. Нажмите на кнопку **Run Sequence** для проверки ошибок.

i. Нажмите на кнопку **OK** для возврата на блок-диаграмму.

Щелкните правой кнопкой мыши по полю вывода **String** и выберите пункт **Create»Indicator** из контекстного меню.

Щелкните правой кнопкой мыши по полю вывода **String Length** и выберите пункт **Create»Indicator** из контекстного меню.

Соедините поле вывода **error out** с ВП **Simple Error Handler VI**.

a. Перейдите на лицевую панель и запустите ВП.

b. Сохраните ВП под именем *Последовательная связь.vi*

c. По окончании закройте ВП.

Конец упражнения 11-5

H. Передача сигнальных данных (дополнительно)

Многие измерительные приборы возвращают сигнальные данные в виде ASCII строк или в двоичном формате. Необходимо принять во внимание то, что передача сигнальных данных в бинарном формате происходит быстрее и требует меньшего количества памяти, чем передача сигнальных данных в виде ASCII строк. Кодирование сигнальных данных в бинарный формат требует меньшего количества байт, чем кодирование в ASCII строки.

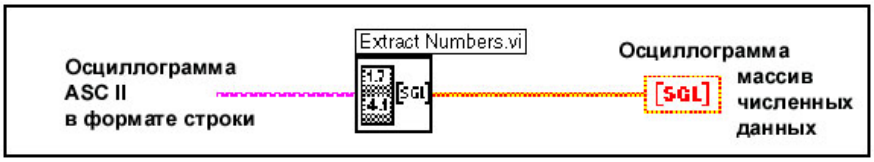
Сигнальные данные в формате ASCII

Например, рассматриваемый сигнал состоит из 1024 значений, которые находятся в диапазоне от 0 до 255. Использование ASCII кодировки требует 4 байта для представления каждого значения (3 байта для числа и 1 байт для разделителя, такого как запятая). Таким образом, потребуется 4096 (4 * 1024) байт плюс заголовок и плюс сопроводительные символы, указывающие на тип представления осциллограммы данных в виде ASCII строки.

Ниже показан пример данных в формате ASCII строки.

CURVE	**{12,28,63,...1024 значения}**	**CR**
Заголовок (6 байт)	значения (по 4 байта на каждое значение)	сопроводительный символ (2 байта)

Для преобразования данных в формате ASCII строки необходимо использовать ВП **Extract Numbers VI**, размещенный в палитре **Functions»User Libraries»Basics I Course**, как показано на блок-диаграмме.



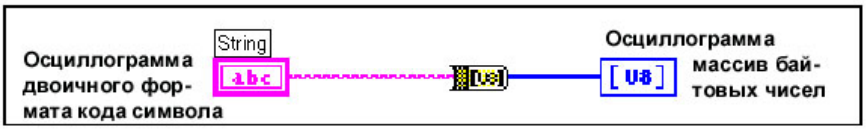
Сигнальные данные в формате однобайтового целого

В этом формате сигнальные данные требуют только 1024 байта. В сумме: (1 * 1024) плюс заголовок плюс сопроводительный символ, представленный в двоичном формате. Для преобразования сигнальных данных в двоичный формат кодов символов требуется 1 байт на значение, т.е. каждое значение представлено как беззнаковое байтовое значение. Пример демонстрирует осциллограмму данных в формате 1-байтовых целых.

CURVE %	**{MSB}{LSB}**	**{A,a...1024 значения}**	**{Chk} CR**
заголовок (7 байт)	счетчик (4 байта)	значения (по 1-у байту на каждое значение)	сопроводительный символ (3 байта)

Преобразовать двоичного формата кодов символов в численный массив

труднее. Необходимо преобразовать строку в целочисленный массив. Для преобразования использовать функцию, размещенную в палитре **Functions»Programming»String»String/Array/Path Conversion**. Перед преобразованием необходимо удалить заголовок и сопроводительные символы с принятого двоичного формата. В противном случае эта информация также будет преобразована в целочисленный массив.



Сигнальные данные в формате двухбайтового целого

Каждое значение в этом формате представлено, как двухбайтовое целое и его преобразование можно осуществить с помощью функции **Type Cast,** размещенной в палитре **Functions»Programming»Numeric»Data Manipulation.** Для получения дополнительной информации о приведении типов используйте руководство *LabVIEW Основы II*.

Например, **GPIB-**осциллограф передает сигнальные данные в формате двухбайтового целого. Данные состоят из 1024 значений, каждое из которых представлено 2 байтами. Поэтому в этом формате сигнальные данные представлены 2048 байтами. Ниже показан формат сигнальных данных, состоящий из 4-байтового заголовка, данных и 2-байтового сопроводительного символа **linefeed**.

ДАТА	**{(HB1,LB1),...1024 значения} CR**	**CF**
заголовок (4 байт)	значения (по 2 байта на каждое значение)	сопроводительный символ (2 байта)

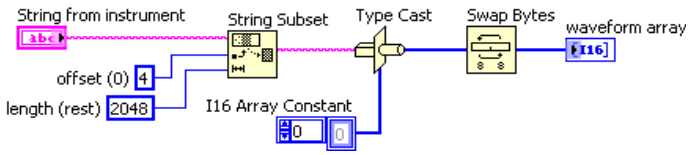
Данные измерительного прибора

«LF»«CR»«LB1024»«HB1024»...«LB2»«HB2»«LB1»«HB1»ATAD

GPIB

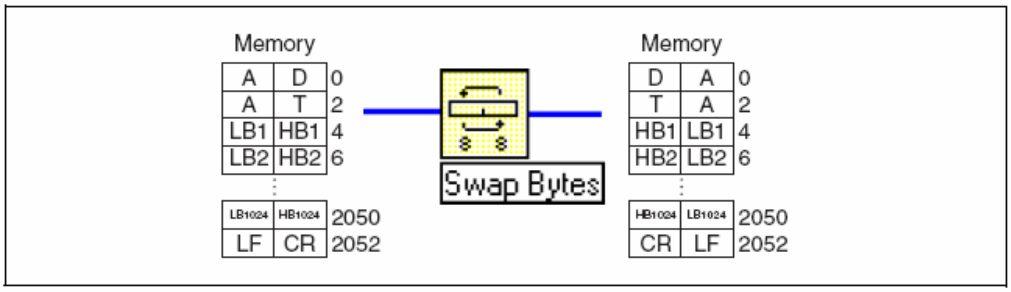
A	**D**
A	**T**
LB1	**HB1**
LB2	**HB2**
….	**….**
LB1024	**HB1024**
LF	**CR**

На примере блок-диаграммы показано использование функции **Type Cast** для преобразования сигнальных данных в формате двухбайтового целого в массив 16-битовых (двухбайтовых) целых.



Для перестановки старших и младших байтов значений необходимо использовать функцию **Swap Bytes**, размещенную на палитре **Functions»Programming»Numeric»Data Manipulation. GPIB** – 8-битовая шина и в каждый момент времени передает только один байт. От измерительного прибора данные идут в обратной последовательности, поэтому и требуется функция **Swap Bytes**. Измерительный прибор передает вначале старший байт, затем младший байт, а приемник-интерфейс старший байт принятого значения физически размещает по адресу младшего, младший байт – по адресу старшего.

Поэтому и требуется их перестановка с помощью функции **Swap Bytes**, как показано на рисунке.



Упражнение 11-6 ВП Пример осциллограммы (дополнительное)

Цель: Создать прибор – цифровой осциллограф, который отображает данные, полученные от измерительного прибора в виде ASCII строки или в двоичном формате.

Осциллограмма данных в виде ASCII строки состоит из 128 значений. Каждое значение передается в виде ASCII символа, включая разделительную запятую. Формат ASCII строки имеет следующий вид:

CURVE {12,28,63,...128 points in total...,}CR LF.

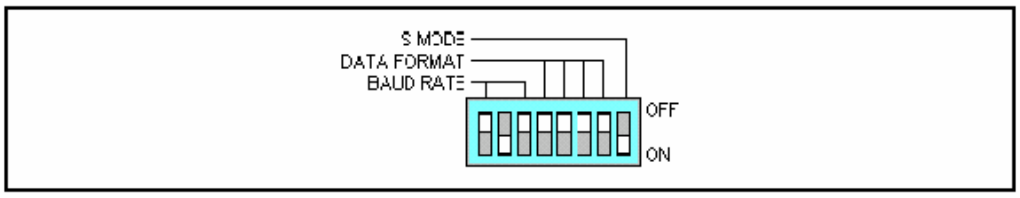
Осциллограмма данных в двоичном формате также состоит из 128 значений. Каждое значение представлено 1-байтовым кодом символа. Осциллограммы данных в двоичном формате имеют следующий вид:

CURVE % {Bin Count MSB}{Bin Count LSB}{aaAA...128 bytes in total...} {Checksum} CR LF.

Для исследования ВП, который конвертирует осциллограмму данных в массив чисел, следует выполнить следующие шаги. Осциллограмма данных принимается ВП от **NI Instrument Simulator** или из заранее подготовленного массива.

Имитатор NI Instrument Simulator

1. Выключите **NI Instrument Simulator** и установите переключатели на поддержку **GPIB-**интерфейса, как показано на рисунке.

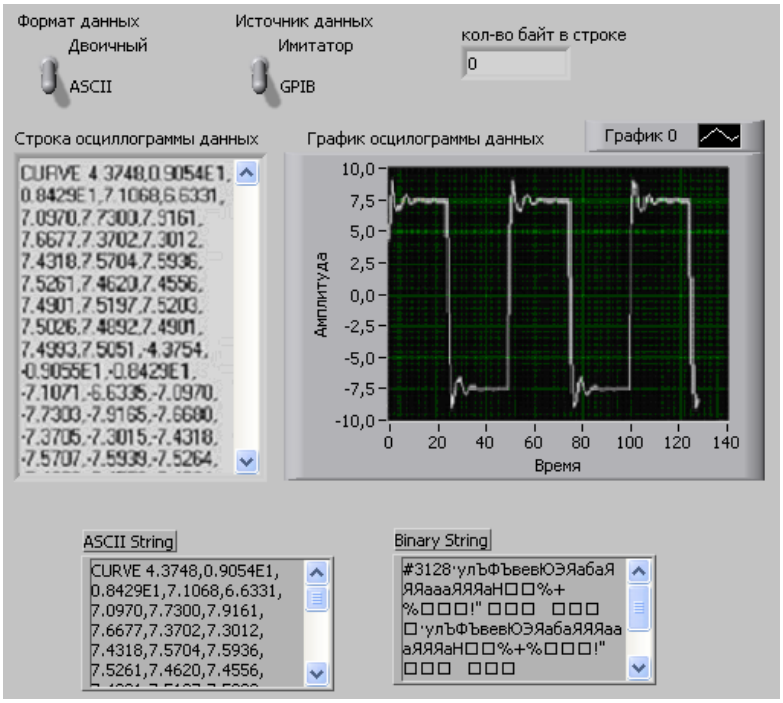


Положение переключателей соответствует имитации измерительного прибора с **GPIB-**интерфейсом с адресом **2**.

2. Включите **NI Instrument Simulator**. Индикаторы-светодиоды **Power** и **Ready** загорятся, что указывает на инициализацию **GPIB** связи.

Лицевая панель

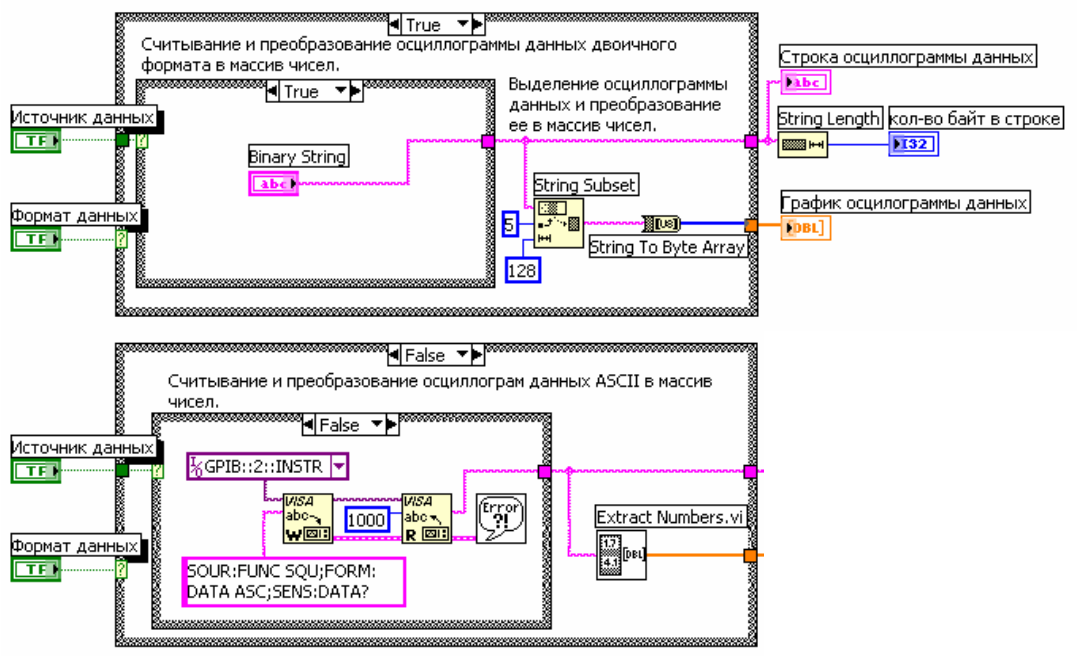
3. Откройте ВП **Waveform Example VI**. Откроется лицевая панель, показанная на рисунке.



Элемент управления **Формат осциллограмм** указывает на тип формата осциллограммы данных: ASCII или двоичный формат. Элемент управления **Источник** указывает, откуда принимаются данные: имитация принятой осциллограммы данных из файла или чтение через **GPIB** от **NI Instrument Simulator**.

Блок-диаграмма

4. Исследуйте блок-диаграмму, показанную на рисунке.



Функция **String Subset,** размещенная в палитре **Functions»Programming»String,** возвращает подстроку из 128 элементов, которая начинается с 5-го байта двоичного формата осциллограммы данных. При этом помимо заголовка исключается сопроводительный символ.

Функция **String to Byte Array**, размещенная в палитре **Functions»Programming»String»String/Array/Path Conversion,** преобразует коды символа двоичного формата в массив беззнаковых целых чисел.

Функция **String Length**, размещенная в палитре **Functions»Programming»String,** возвращает количество символов в осциллограмме данных.

ВП **Extract Numbers VI**, размещенный в палитре **Functions»User Libraries»Basics I Course**, выделяет числа из осциллограммы данных в формате **ASCII** символов и помещает их в массив чисел. Нечисловые символы, такие как запятые, разделяют числа в строке.

ВП **VISA Write** и ВП **VISA Read**, размещенные в палитре **Functions»Instrument I/O»VISA,** осуществляют запрос **NI Instrument Simulator** на генерацию осциллограммы прямоугольных импульсов в формате ASCII или 1-байтовом двоичном формате.

ВП **Simple Error Handler VI**, размещенный в палитре **Functions» Programming»Dialog & User Interface,** сигнализирует о наличии ошибки.

5. Отобразите лицевую панель и запустите ВП.

   Вариант TRUE структуры **Case** осуществляет сбор и преобразование осциллограммы данных в двоичном формате в массив чисел. Вариант FALSE осуществляет сбор и преобразование осциллограммы данных в формате ASCII строки в массив чисел.

6. Выберите положение ASCII элемента управления **Формат осциллограмм** и запустите ВП. Считанная осциллограмма данных ASCII отобразится в элементе **Строка осциллограммы**, далее ВП преобразует осциллограмму данных в массив чисел, отобразит ее длину и выведет массив чисел на график.

7. Переключите элемент **Формат осциллограмм** в положение **Двоичный** и снова запустите ВП. Считанная осциллограмма данных двоичного формата отобразится в элементе **Строка осциллограммы**, далее ВП преобразует ее в массив чисел, отобразит ее длину и выведет массив чисел на график.

Примечание. Двоичный формат данных использует меньшее количество байт для кодировки считанных значений, поэтому принимается из интерфейса быстрее, но требует дополнительных функций при преобразовании в массив чисел.

8. Закройте ВП, изменения не сохраняйте.

Конец упражнения 11-6

Краткое изложение пройденного материала, советы и секреты

- Среда LabVIEW позволяет связываться с измерительными приборами через разнообразные аппаратные интерфейсы.
- Используйте конфигурационную утилиту **MAX** для настройки и тестирования **GPIB-**интерфейса, подсоединенных измерительных приборов, последовательных и параллельных портов.
- Драйверы устройств LabVIEW позволяют избежать изучения трудных низкоуровневых команд для каждого устройства.
- Библиотека драйверов устройств LabVIEW размещена на диске LabVIEW CD. Их также можно загрузить с Web-сайта www.ni.com компании National Instruments.
- Все библиотечные драйверы устройств имеют одинаковую базовую иерархию Виртуальных Приборов.
- Функции **VISA** используются для ввода и вывода данных из интерфейса, а также управления VXI, GPIB, RS-232 или других типов интерфейсами.
- Для быстрого создания ВП для связи с измерительным прибором используйте экспресс-ВП **Instrument I/O Assistant**. Таким способом можно управлять **VXI-**, **GPIB-**, RS-232- и другими устройствами.
- Последовательная связь является самым распространенным средством передачи данных между компьютером и периферийными устройствами, такими как программируемые измерительные приборы или даже другой компьютер.
- Измерительные приборы передают осциллограммы данных в различных форматах. Осциллограмма данных в **ASCII** формате легко читается, а осциллограмма данных в двоичном формате более компактна. Среда LabVIEW включает в себя все необходимые функции и ВП, преобразующие осциллограмму данных для последующей обработки.

Дополнительные упражнения

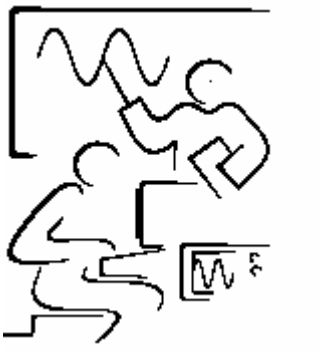
1-1 Используйте ВП **NI DEVSIM Getting Started VI**, как и в упражнении 11-2, для тестирования и исследования драйверов **NI Instrument Simulator** через последовательный порт.

1-2 Откройте **ВП Монитор Напряжения**, созданный в упражнении 11-4. Измените блок-диаграмму так, чтобы данные записывались в виде таблицы символов в файл ***напряжение.txt*** в формате, показанном на рисунке.

	A	B	C	D
1	Start Date: 6/8/00	Start Time: 1:26 PM		
2	Max Voltage: 9.151000	Min Voltage: 0.354010		
3	Data:			
4	8.965			
5	9.067			
6	0.354			
7	5.08			
8	4.511			
9	3.946			
10	6.446			
⋮	⋮			
N	2.293			

Сохраните ВП под именем *Запись напряжения в файл.vi*

Примечания



Урок 12
Настройка ВП

В этом уроке рассказывается о том, как настроить внешний вид и поведение Виртуальных Приборов и среды LabVIEW.

В этом уроке изложены вопросы:

A. Настройка внешнего вида лицевой панели.
B. Отображение лицевых панелей подпрограмм ВП во время работы.
C. Назначение и использование «горячих» клавиш.
D. Редактирование ВП с некоторыми свойствами.
E. Настройка палитр (дополнительно).
F. Обмен данными между ВП с помощью общих переменных.

A. Настройка внешнего вида лицевой панели

Для того чтобы пользователю было удобно работать с ВП, существует возможность настройки внешнего вида лицевой панели. Например, можно убрать меню и инструментальную панель т.к. при работе ВП они не нужны.

Для настройки внешнего вида и поведения ВП следует выбрать пункт меню **FileVI»Properties**. Можно также щелкнуть правой кнопкой мыши по иконке в правом верхнем углу лицевой панели и выбрать пункт **VI Properties** из контекстного меню. Диалоговое окно **VI Properties** не доступно во время работы ВП. За более подробной информацией о настройке поведения прибора обратитесь к *Учебному курсу LabVIEW Основы II*.

Для выбора настроек следует использовать выпадающее меню **Category** в диалоговом окне **VI Properties**. Можно выбрать следующие категории настроек.

- **General** – общие настройки, такие как путь, по которому ВП был сохранен, номер версии, история ВП, а также все изменения, которые были сделаны с момента последнего сохранения. На этой странице также можно отредактировать иконку ВП.
- **Documentation** – эта страница используется для добавления описания ВП и ссылки на справочный файл. Для более подробной информации о документировании ВП смотрите *Упражнение 3-3*.
- **Security** – на этой странице можно защитить с помощью пароля или заблокировать доступ к ВП.
- **Window appearance** – эта страница используется для настройки различных свойств окна.
- **Window Size** – на этой странице устанавливается размер окна.
- **Execution** – эта страница используется для настройки вариантов выполнения ВП.

Внешний вид окна

Для настройки того, как будет выглядеть окно при выполнении ВП, выберите пункт **Window appearance** из выпадающего меню **Category** в диалоговом окне **VI Properties**.

По умолчанию, название окна ВП совпадает с названием самого ВП. Название окна можно изменить, чтобы сделать его более понятным, чем имя файла, под которым сохранен ВП. Это удобно при локализации ВП, так как название окна может быть переведено на другой язык. Для того чтобы изменить название окна следует снять галочку напротив надписи **Same as VI Name** и ввести новое название в поле **Window Title**.

При настройке внешнего вида окна можно выбрать несколько стилей оформления, описанных ниже. При выборе какого-либо стиля его графическое представление отображается справа.

- **Top-level Application Window** – отображается название и меню, скрыты полосы прокрутки и инструментальная панель, позволяется пользователю закрыть окно, разрешается использование контекстных меню во время работы ВП, запрещено изменять размер окна, при вызове приложения отображается лицевая панель.
- **Dialog** – ВП ведет себя подобно диалоговому окну в операционной системе, т.е. пользователь не может взаимодействовать с другими окнами LabVIEW, пока это окно ВП открыто. Однако это не означает, что другие окна или приложения не могут отображаться поверх данного окна. (**UNIX**) Нельзя заставить окно оставаться поверх всех других окон.

  Диалоговые окна остаются поверх других окон, не имеют ни меню, ни полос прокрутки, ни инструментальной панели. Их можно закрыть, но изменить их размер нельзя. Разрешено использование контекстных меню во время работы, при вызове отображается лицевая панель. Если какой-нибудь логический элемент лицевой панели ассоциирован с клавишей <Enter> или <Return>, LabVIEW отображает его границу более темным цветом.
- **Default** – стиль окна такой же, как и у окон среды LabVIEW.
- **Custom** – стиль определяется пользователем, для определения стиля окна следует нажать кнопку **Customize**.

Размер окна

Для настройки размера лицевой панели и объектов, расположенных на ней, выберите пункт **Window appearance** из выпадающего меню **Category** в диалоговом окне **VI Properties**. На странице расположены следующие настройки:

- **Minimum Panel Size** – устанавливает минимальный размер лицевой панели. Если на странице **Window appearance** пользователю разрешено изменять размер окна, то минимальные высота и ширина задаются на этой странице.
- **Size the Front Panel to the Width and Height of the Entire Screen** – автоматически изменяет размер окна так, чтобы оно отображалось на весь экран при запуске ВП. Старые положение и размер окна не сохраняются, поэтому при переходе обратно в режим редактирования окно остается на новом месте.
- **Maintain Proportions of Window for Different Monitor Resolutions** – изменяет размер окна лицевой панели так, чтобы оно занимало примерно столько же места на мониторе при изменении разрешения. Например, если ВП разрабатывался на мониторе с

разрешением 1024*768, то для нормального отображения ВП на мониторе с разрешением 800*600 следует использовать эту опцию.

- **Scale All Objects on Front Panel as the Window Resizes** – автоматически изменяет размеры всех объектов на лицевой панели пропорционально размеру окно лицевой панели. Размер текста при этом не изменяется, т.к. размеры шрифтов фиксированы. Эту опцию следует использовать, если пользователю разрешено изменять размер окна лицевой панели.

B. Отображение лицевых панелей подпрограмм ВП во время работы ВП

Размер единственной лицевой панели накладывает ограничение на максимально возможное количество используемых элементов. Для решения этой проблемы, используемые Виртуальные Приборы следует организовать таким образом, чтобы самый верхний ВП предоставлял возможность управлять основными параметрами, а подпрограммы ВП – второстепенными.

Совет Для организации лицевой панели можно использовать табуляцию.

Как правило, при запуске подпрограммы ВП его лицевая панель не отображается. Для отображения лицевой панели при запуске отдельного экземпляра подпрограммы ВП следует воспользоваться диалоговым окном настройки узла подпрограммы ВП **SubVI Node Setup**. Для отображения лицевой панели при запуске всех экземпляров подпрограмм ВП следует воспользоваться диалоговым окном свойств ВП **VI Properties**.

Отдельный экземпляр

Для отображения лицевой панели при запуске отдельного экземпляра подпрограммы ВП следует щелкнуть правой кнопкой мыши по подпрограмме ВП и из контекстного меню выбрать пункт **SubVI Node Setup**. В открывшемся диалоговом окне отметьте пункты **Show Front Panel when called** и **Close afterwards if originally closed**. Диалоговое окно также содержит следующие элементы:

- **Open Front Panel when loaded** – отображает лицевую панель при загрузке подпрограммы ВП или ВП, который ее вызывает.
- **Show Front Panel when called** – отображает лицевую панель при запуске подпрограммы ВП.
- **Close afterwards if originally closed** – если опция **Show Front Panel when called** включена, и подпрограмма ВП была закрыта до вызова, то лицевая панель закрывается по завершению работы подпрограммы ВП.
- **Suspend when called** – приостанавливает выполнение подпрограммы ВП и ожидает взаимодействия с пользователем. Эту опцию можно также включить, выбрав **Operate»Suspend when called**.

Все экземпляры

Для отображения лицевой панели при запуске всех экземпляров подпрограмм ВП следует выбрать пункт **File»VI Properties**. В диалоговом окне **VI Properties** выберите пункт **Window appearance** из выпадающего меню **Category**, нажмите кнопку **Customize** и отметьте пункты **Show Front Panel when called** и **Close afterwards if originally closed**.

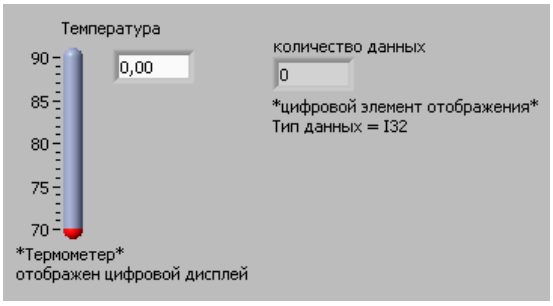
Упражнение 12-1 ВП Всплывающий график и ВП Использование всплывающего графика

Цель: Отображение лицевой панели подпрограммы ВП при выполнении ВП.

Выполните следующие пункты для создания ВП, который измеряет температуру каждые 0,5 секунды в течение 10 секунд, отображает лицевую панель подпрограммы ВП с графиком накопленных данных и не закрывает лицевую панель подпрограммы, пока не нажата кнопка.

Лицевая панель

1. Откройте новый ВП и создайте лицевую панель, как показано ниже.

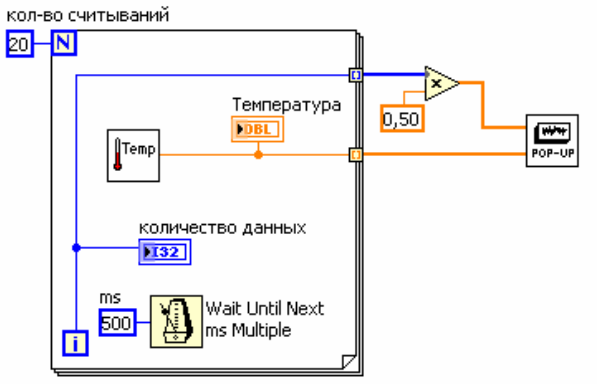


   a. Элемент отображения справа от термометра – это цифровой дисплей термометра. Для отображения численного значения температуры следует щелкнуть правой кнопкой мыши по термометру и из контекстного меню выбрать пункт **Visible Items»Digital Display**.

   b. Измените тип представления данных элемента **количество данных** на **I32**.

Блок-диаграмма

2. Создайте блок-диаграмму, как показано ниже.



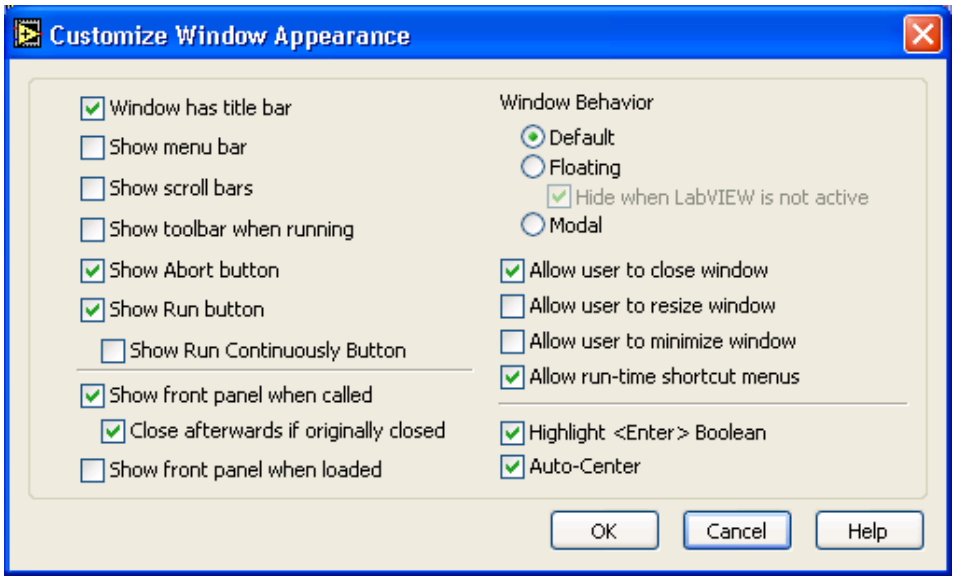
   a. Поместите на блок-диаграмму **ВП Термометр**, который был сделан в упражнении 3-3. Этот ВП измеряет значения температуры.

b. Поместите на блок-диаграмму функцию **Wait Until Next ms Multiple,** расположенную в палитре **Functions» Programming»Timing**. Эта функция заставляет цикл **For** выполняться каждые 500 мс.

c. Поместите на блок-диаграмму функцию **Multiply**, расположенную в палитре **Functions»Programming» Numeric**. Эта функция умножает на 0,5 каждый элемент выходного массива x, т.к. измерения делаются каждые полсекунды.

d. Поместите на блок-диаграмму **ВП Всплывающий график**, расположенный в папке *C:\Exercises\LV Basics I*. Этот ВП отображает данные на двухкоординатном графике.

3. Сохраните ВП под именем *Использование всплывающего графика.vi*
4. Настройте подпрограмму ВП так, чтобы она при вызове отображала на экране свою лицевую панель.
   a. Откройте лицевую панель **ВП Всплывающий график,** два раза щелкнув по нему левой кнопкой мыши.
   b. Выберите в главном меню пункт **File»VI Properties**.
   c. Выберите пункт **Window appearance** из выпадающего меню **Category**.
   d. Нажмите на кнопку **Customize**. Настройте внешний вид окна, как показано на следующем диалоговом окне.



   e. Нажмите два раза на кнопку **OK**, сохраните и закройте ВП. Если лицевая панель не будет закрыта, то она не закроется по окончании работы подпрограммы ВП.
5. Запустите **ВП Использование всплывающего графика**. После 10-ти секундного набора значений температуры появляется лицевая панель **ВП Всплывающий график**, и строится график

температуры. Нажмите на кнопку **DONE** для возврата в вызывающий ВП.

6. Измените настройки внешнего вида окна для подпрограммы ВП **Pop-up Graph VI** на **Dialog**.
7. Сохраните и закройте подпрограмму ВП.
8. Запустите **ВП Использование всплывающего графика** еще раз. Теперь окно лицевой панели ведет себя как диалоговое окно, то есть, например, находится поверх всех окон и использует системные цвета.
9. Закройте все открытые ВП.

Конец упражнения 12-1

C. «Горячие» клавиши

Для перемещения фокуса ввода во время работы ВП от одного элемента управления к другому можно воспользоваться клавишей <**Tab**>. Фокус ввода также можно установить, щелкнув правой кнопкой мыши по элементу управления. Пока элемент управления обладает фокусом ввода, значение в него можно ввести с клавиатуры. Если элемент управления цифровой или текстовый, то LabVIEW выделяет текст, который можно отредактировать. Если элемент управления логический, то поменять его значение можно нажатием клавиш пробел или <**Enter**>.

Для элементов управления можно назначать «горячие» клавиши, то есть пользователь может передвигаться по лицевой панели, используя другие клавиши. Щелкните правой кнопкой мыши по элементу управления и выберите пункт контекстного меню **Advanced»Key Navigation** для вызова диалогового окна **Key Navigation**.

Примечание Пункт контекстного меню **Advanced»Key Navigation** для элементов отображения не активен, так как вводить данные в элементы отображения невозможно.

Выберите «горячие» клавиши для нужного элемента управления в секции **Key Assignment**. Названия элементов управления на лицевой панели, перечисленные в списке **Current Assignments**, соответствуют собственным меткам элементов управления.

Для того, чтобы пользователь не имел доступа к элементу управления с помощью клавиши <**Tab**> следует отметить пункт **Skip this control when tabbing**.

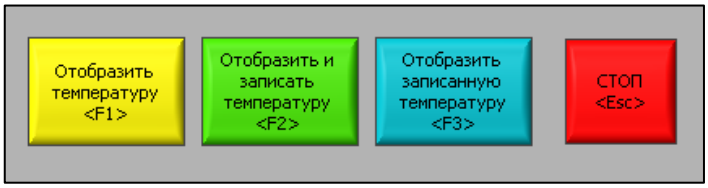
Упражнение 12-2 ВП Температурная система

Цель: Назначение «горячих» клавиш для элементов управления и отображение лицевой панели подпрограммы ВП при выполнении ВП.

Выполните следующие пункты для создания системы слежения за температурой, которая по требованию может проводить три различных теста.

Лицевая панель

1. Откройте **ВП Температурная система**, находящийся в директории *C:\Exercises\LV Basics I*. Лицевая панель уже создана.



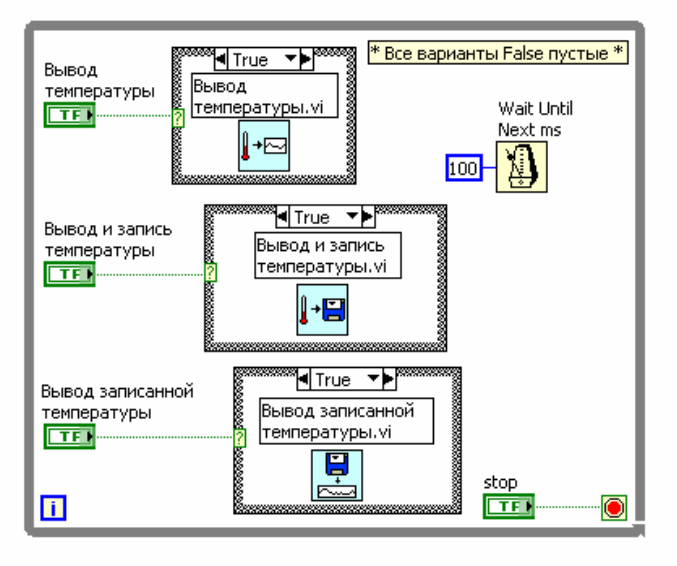
На лицевой панели расположены четыре логические кнопки. Механическое действие первых трех кнопок – **Latch when Pressed**. Эта установка изменяет значение логического элемента после нажатия левой кнопки мыши и сохраняет его до первого обращения к нему ВП. После обращения ВП значение логического элемента возвращается в исходное положение. Это действие похоже на разрыв цепи и применяется для остановки цикла **While** или для однократного выполнения какой-либо операции.

Механическое действие кнопки **STOP – Latch When Released**. Эта установка изменяет значение логического элемента только после того, как отпускается левая кнопка мыши, причем курсор должен находиться внутри графических границ элемента. После обращения ВП значение логического элемента возвращается в исходное положение. Этот режим гарантирует изменение значения как минимум один раз. Это действие похоже на действие системных кнопок и кнопок в диалоговых окнах.

2. Щелкните правой кнопкой мыши по элементу управления и выберите пункт контекстного меню **Advanced»Key Navigation** для вызова диалогового окна **Key Navigation**.
3. В секции назначения клавиш **Key Assignment** назначьте «горячие» клавиши, указанные на предыдущем изображении лицевой панели.
4. Повторите шаги 2 и 3 для каждого элемента.

Блок-диаграмма

5. Посмотрите на уже созданную блок-диаграмму.



- **ВП Вывод температуры** эмулирует измерение температуры каждые 500 мс и строит результаты на графике диаграмм в режиме **strip chart**.

- **ВП Вывод и запись температуры** эмулирует измерение температуры каждые 500 мс, строит результаты на графике диаграмм в режиме **strip chart** и записывает их в файл.

- **ВП Вывод записанной температуры** открывает выбранный файл, считывает данные и строит их на графике.

6. Настройте каждую подпрограмму ВП так, чтобы при вызове отображалась ее лицевая панель.
   a. Щелкните правой кнопкой мыши по подпрограмме ВП и из контекстного меню выберите пункт **SubVI Node Setup**.
   b. В открывшемся диалоговом окне отметьте пункты **Show Front Panel when called** и **Close afterwards if originally closed**.
   c. Повторите шаги a и b для остальных двух подпрограмм ВП.
7. Сохраните ВП. Перейдите на лицевую панель и запустите ВП.
8. Нажмите мышкой на каждую кнопку и нажмите соответствующие «горячие» клавиши. Все три прибора возвращаются в **ВП Температурная система** при нажатии клавиши <**Enter**>. Попробуйте нажать клавишу <**Enter**> для возвращения в вызывающий ВП.
9. Остановите ВП.
10. Настройте **ВП Температурная система** на автоматический запуск при открытии.
    a. Выберите в главном меню пункт **File»VI Properties**.
    b. Выберите пункт **Execution** в выпадающем меню **Category**.
    c. Отметьте пункт **Run When Opened**.

11. Настройте ВП таким образом, чтобы при работе ВП меню и инструментальная панель не отображались.
    a. Выберите пункт **Window appearance** из выпадающего меню **Category**.
    b. Нажмите на кнопку **Customize**.
    c. Снимите выделение с пунктов **Show Menu Bar** и **Show Tool Bar When Running**.
    d. Два раза нажмите на кнопку **OK**.
12. Сохраните и закройте все ВП.
13. Вновь откройте **ВП Температурная система**. ВП автоматически запустится при открытии. Нажмите мышкой на кнопки на лицевой панели или используйте соответствующие «горячие» клавиши.
14. Остановите и закройте все ВП.

Конец упражнения 12-2

D. Редактирование свойств ВП

Некоторые настройки ВП приводят к тому, что редактировать ВП становится трудно. Например, можно отметить настройку **Run When Opened** и убрать меню и инструментальную панель. Если настроить ВП так, что он будет закрываться и выходить из LabVIEW по окончании работы, то остановить и отредактировать такой ВП не получится. Редактировать такой ВП будет очень трудно.

Примечание Для выхода из среды LabVIEW можно использовать функцию **Quit LabVIEW**, расположенную в палитре **Functions»Programming» Application control**. Эта функция прекращает работу всех ВП и завершает работу текущей сессии LabVIEW. Эта функция имеет только одно поле ввода данных. Если оно подключено, то сессия LabVIEW заканчивается только когда входное значение равно TRUE. Если же поле ввода не подключено, то сессия LabVIEW заканчивается при выполнении данного узла.

Для того чтобы избежать описанных в примере ситуаций, перед редактированием свойств ВП следует создавать резервную копию ВП в новой папке. Для этого выберите из меню **File»Save with Options**.

Для сохранения ВП со всей его иерархией следует выбрать опцию **Development Distribution**. В сохраняемые файлы также можно включить файлы *vi.lib*. После создания резервной копии измените свойства ВП. При возникновении каких-либо проблем можно вернуться к резервной копии.

Внимание При выборе опции **Remove diagrams** удаляется исходный код ВП. Использовать эту опцию следует только в том случае, когда точно известно, что редактировать ВП больше не потребуется. Перед тем, как сохранять ВП без блок-диаграмм, следует сделать резервную копию с блок-диаграммами.

Если ВП уже сохранен с настройками, создающими трудности при редактировании, смотрите упражнение 12-3 для получения более подробной информации о редактировании ВП.

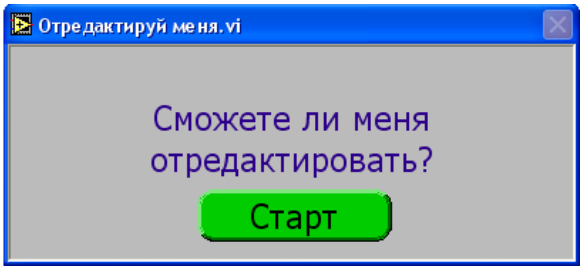
Упражнение 12-3 ВП Отредактируй меня

Цель: Отредактировать ВП со свойствами, затрудняющими редактирование.

Выполните следующие пункты для изменения ВП, настроенного на запуск при открытии и выход из LabVIEW по завершению работы.

Лицевая панель

1. Закройте все открытые ВП и откройте **ВП Отредактируй меня**, находящийся в папке *C:\Exercises\LV Basics I*. Его лицевая панель уже создана.



ВП запускается сразу во время открытия. Во время работы ВП невозможно использовать ни меню, ни инструментальную панель, ни «горячие» клавиши для редактирования или остановки ВП.

2. Нажмите на кнопку **Start**. Через 10 секунд ВП завершит свою работу и работу LabVIEW.
3. Вновь запустите LabVIEW и откройте новый ВП.
4. Выполните пункты с 5 по 13, если ВП, который нужно отредактировать, не содержит подпрограмм или неизвестно, что он содержит.

   Если же ВП, который нужно отредактировать, содержит подпрограммы, откройте одну из подпрограмм ВП и измените ее блок-диаграмму для «поломки» подпрограммы. Можно, например, поместить на блок-диаграмму функцию **Add** и не подсоединить ее поля ввода данных. Откройте ВП, который нужно отредактировать. Так как подпрограмма ВП перестала быть выполнимой, то и сам ВП не сможет выполниться. ВП откроется в режиме редактирования, и кнопка **Run** будет разорвана. Не забудьте исправить подпрограмму ВП после редактирования ВП.
5. Перейдите на блок-диаграмму нового ВП.
6. Поместите уже созданный **ВП Отредактируй меня** на блок-диаграмму. Появится лицевая панель ВП.

Теперь можно отображать блок-диаграмму **ВП Отредактируй меня**, но редактировать ее нельзя.

7. Выберите в главном меню пункт **Operate»Change to Edit Mode**. Появится диалоговое окно с сообщением, что ВП заблокирован.
8. Нажмите на кнопку **Unlock**. Теперь можно редактировать ВП. Разблокировать ВП также можно, выбрав из меню пункт **File»VI Properties** и затем из выпадающего меню **Category** пункт **Security**.
9. Выделите и удалите с блок-диаграммы функцию **Quit LabVIEW**.
10. Сохраните и закройте **ВП Отредактируй меня**. Закройте новый ВП, не сохраняя изменений.
11. Откройте **ВП Отредактируй меня**.
12. Отредактируйте ВП после окончания его работы.
13. Закройте **ВП Отредактируй меня**.

Конец упражнения 12-3

E. Настройка палитр функций и элементов управления (дополнительно)

Для удобства доступа к наиболее часто используемым ВП и функциям существует возможность настройки палитр функций и элементов управления. Можно добавлять ВП и элементы управления в палитры, убирать ВП и функции, а также изменять встроенные палитры.

Добавление ВП и элементов управления в Библиотеку пользователя и в Инструментальную библиотеку

Самый простой способ добавить ВП или элемент управления в палитры **Functions** и **Controls** – это сохранить их в папку *user.lib*. При повторном запуске LabVIEW палитры **Functions»User Libraries** и **Controls»User Controls** будут содержать подпалитры для каждой папки, библиотеки ВП (*.llb*) или файла меню (*.mnu*) в *user.lib*, и иконки для каждого файла в *user.lib*. LabVIEW автоматически обновляет палитры при добавлении или удалении файлов из соответствующей папки.

Палитра **Functions»Instrument I/O** соответствует папке *instr.lib*. Драйвера приборов следует сохранять именно в эту папку для быстрого доступа через палитру **Functions**.

При использовании этого метода добавления ВП или элемента управления в палитры **Functions** и **Controls** нельзя указать определенное положение добавляемого элемента на палитре.

Создание и редактирование видов палитры

Для указания точного местоположения на палитре добавленного элемента необходимо создать вид палитры. В LabVIEW встроены три вида палитры: вид сбор данных, дополнительный вид и экспресс-вид. Для смены текущего вида палитры следует нажать кнопку **Options** на панели инструментов палитр **Functions** и **Controls**. В появившемся диалоговом окне **Controls/Functions Palette Options** выберите вид палитры из выпадающего меню **Palette View**.

В выпадающем меню **Format** можно настроить вид палитры так, чтобы отображались только названия элементов (**text-only**) или только их иконки (**icon-only**).

Для редактирования существующего вида палитры следует выбрать в меню **Tools»Advanced»Edit Palette Views**. В появившемся диалоговом окне **Edit Controls and Functions Palette** выберите вид палитры из выпадающего меню **Palette view**.

Более подробную информацию о видах палитр можно найти в *Руководстве пользователя LabVIEW* и в справочной системе *LabVIEW Help*.

F. Обмен данными между ВП с помощью общих переменных

Разделение данных с использованием общих переменных (Shared Variables)

В версиях LabVIEW 7.x и более ранних для передачи информации между приложениями, которые нельзя соединить проводниками, используются глобальные (global variables) и локальные (local variables) переменные. Тип переменной выбирается исходя из размера создаваемого приложения.

В LabVIEW 8.0 введен новый тип переменных: общие переменные (**shared variables**). Используя общие переменные, Вы можете читать и передавать данные между виртуальными приборами внутри проекта или используя сеть. Общие переменные комбинируют в себе как функциональность встроенных в LabVIEW технологий передачи данных (таких как DataSocket), так и возможности управления данных в окне проекта. Читать или передавать данные с помощью общих переменных можно, как с лицевой панели виртуального прибора, так и с блок-диаграммы.

Создание общих переменных

Для создания общей переменной откройте проект LabVIEW. Затем в окне проекта нажмите правой кнопкой мыши на строку, отображающую устройство, либо на библиотеку проекта, либо на папку внутри библиотеки проекта и из контекстного меню выберите **New»Variable**.

Примечание. Общая переменная должна быть частью библиотеки проекта. Если Вы попытаетесь создать общую переменную для устройства или папки, которые не являются частью проекта, то LabVIEW создаст новую библиотеку проекта для расположения общей переменной.

Чтобы сконфигурировать общую переменную программно используйте ссылку на общую переменную (Shared Variable Refnum) и папку свойства. Для этого создайте индикатор для общей переменной, затем, используя контекстное меню к индикатору, создайте ссылку (Reference).

Чтение и передача данных с помощью общей переменной на лицевой панели.

Используйте привязку данных к лицевой панели виртуального прибора для того, чтобы читать и передавать данные объекту, находящемуся на лицевой панели.

Замечание Объекты, расположенные на лицевой панели, можно привязать только к общим переменным типа network-published.

Чтобы создать переменную (Control) связанную с общей переменной, перетащите с помощью мышки общую переменную из окна проекта на лицевую панель виртуального прибора.

Вы также можете привязать уже созданную переменную (Control) к общей переменной. Для этого нажмите правой кнопкой мыши на Control и из контекстного меню выберите **Properties.** Затем, используя опции на закладке **Data Binding,** привяжите Control к общей переменной.

После того, как Вы привязали общую переменную к переменной Control, изменение значений переменной на лицевой панели приведет к изменению значений общей переменной.

Чтение и передача данных с помощью общей переменной на блок-диаграмме

Узел общей переменной (shared variable node) это объект, который указывает на общую переменную, находящуюся в окне проекта. Вы можете использовать узел общей переменной чтения и передачи данных, а также для чтения временных отметок (time stamp) общей переменной.

Для создания узла общей переменной перетащите с помощью мышки общую переменную из окна проекта на блок-диаграмму виртуального прибора.

Вы также можете создать узел общей переменной, выбрав **Shared Variable** в палитре **Functions»Programming»Structures**. Чтобы соединить появившийся на блок-диаграмме узел общей переменной с общей переменной, находящийся в окне проекта, двойным нажатием на левую кнопку мыши в области **Shared Variable** вызовите диалоговое окно **Select Variable** и выберите там нужную переменную. Вы также можете нажать на правую кнопку мыши и открыть окно **Select Variable**, используя контекстное меню.

Установка очередности передачи данных общей переменной

По умолчанию, сразу несколько приложений могут передавать данные в общую переменную. Однако, для общей переменной типа network-published Вы можете задать, чтобы в данный момент времени она принимала значения только от одного устройства. Для этого в диалоговом окне **Shared Variable Properties** отметьте галочкой пункт **Single Writer**. После этого только первое устройство, передающее данные и подсоединенное к общей переменной, может передавать данные; все последующие устройства этого делать не могут. Только, когда первое устройство отсоединено, следующие устройства в порядке очереди могут передавать данные.

Буферизация для общих переменных

Если Вы используете общую переменную для распределения данных программно, то по умолчанию LabVIEW передает только самые последние значения всем устройствам и файлам, принимающим значения. Когда одно устройство передает данные быстрее, чем другое их читает, то более новые данные будут записываться поверх старых до того, как устройство их считает. В этом случае, если читающее устройство не успело прочитать данные, то они теряются. Если Вы хотите считывать все значения общей переменной, то Вы должны буферизовать данные. Для этого отметьте галочкой в диалоговом окне **Shared Variable Properties** пункт **Use Buffering**.

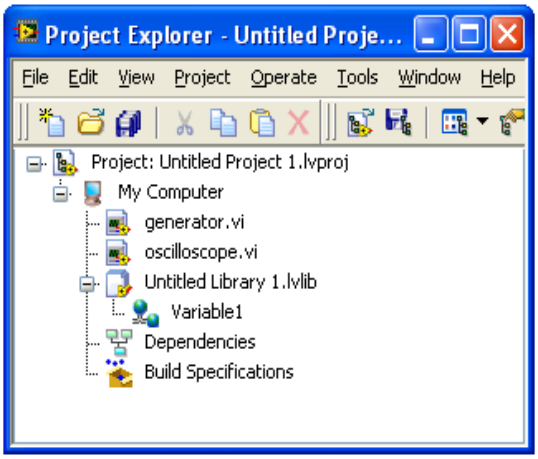
Управление общими переменными вне проекта

В LabVIEW существует возможность редактировать, создавать, отслеживать и управлять общими переменными вне проекта. Чтобы это осуществить отобразите диалоговое окно **Variable Manager,** расположенное в меню **Tools»Shared Variable»Variable Manager**.

Упражнение 12-4 Обмен данными между ВП с помощью общей переменной

Цель: Создание нового проекта, содержащего два виртуальных прибора: генератор сигнала и осциллограф. Осуществление обмена данными между виртуальными приборами с помощью общей переменной.

Ниже приведена последовательность действия для создания нового проекта, содержащего два виртуальных прибора обменивающихся между собой данными с помощью общей переменной.



1. Создайте новый проект. Для этого откройте окно нового проекта **Project Explorer**, используя либо окно запуска LabVIEW, либо пункт **File»New Project**, расположенный на линейке инструментов виртуального прибора.

2. Добавьте в проект два новых виртуальных прибора. Используйте для этого либо пункт **File»New VI**, расположенный на линейке инструментов окна проекта, либо контекстное меню к пункту **My Computer**.

3. Сохраните первый ВП с именем *Generator.vi*, а второй – с именем *Oscilloscope.vi*. Для сохранения файла, нажмите на правую кнопку мыши на строке с названием файла и из контекстного меню выберите пункт save.

4. Добавьте в проект новую общую переменную.

   a. Нажмите в окне проекта правой кнопкой мыши на пункт **My Computer** и из контекстного меню выберите **New»Variable**. После этого появится диалоговое окно **Shared Variable Properties**.

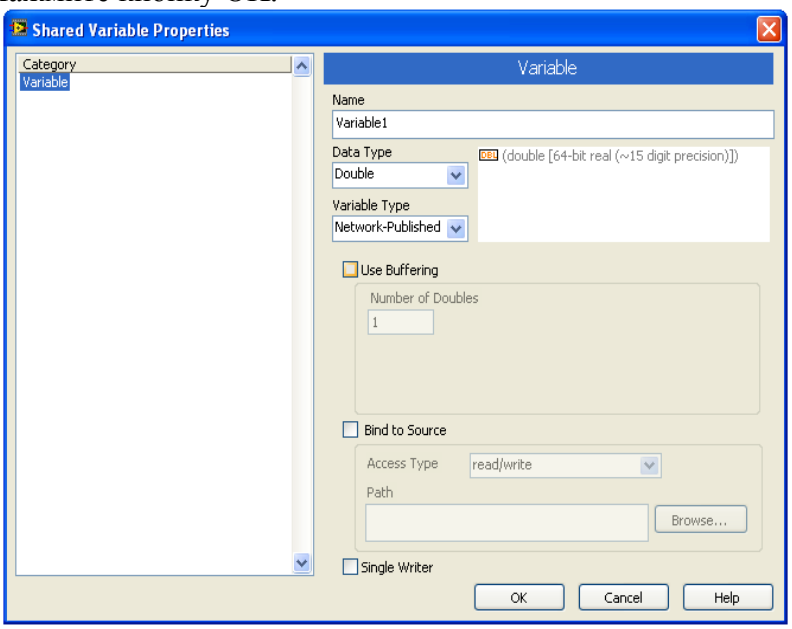
   b. Введите имя переменной в строке **Name**.

c. Установите в строке **Data Type** тип данных **Double**.

d. Установите в строке **Variable Type** тип переменной **Network-Published**.

e. Пункты **Use Buffering, Bind to Source** и **Single Writer** оставьте отключенными.

f. Нажмите кнопку OK.

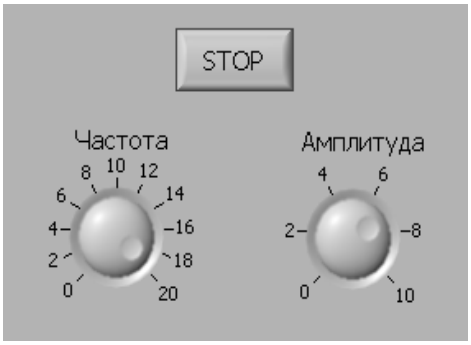


Работа с ВП Generator.vi

Лицевая панель

1. Откройте лицевую панель ВП Generator.vi

2. Добавьте на лицевую панель два элемента управления данных Частота и Амплитуда.



a. Выберите элемент управления данных **Knob** в палитре **Controls»Modern»Numeric** или в палитре **Controls»Express» Numeric Controls**.

b. Задайте для элементов управления имена **Частота** и **Амплитуда**.

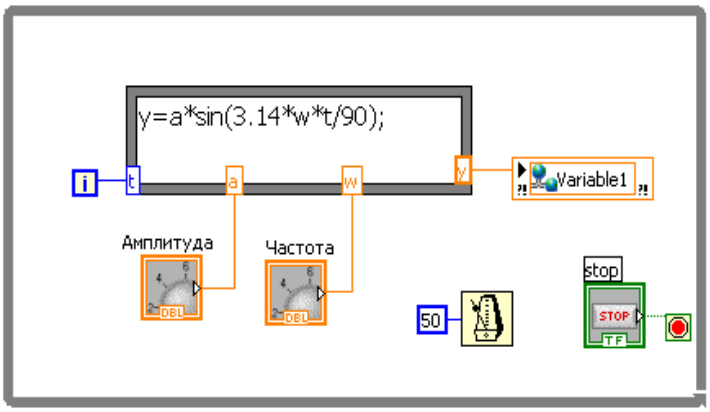
c. Установите диапазон изменения данных для элемента **Частота** от 0 до 20, а для элемента **Амплитуда** – от 0 до 10.

3. Добавьте на лицевую панель кнопку STOP. Выберите её в палитре **Controls»Modern»Boolean**. Нажмите на неё правой кнопкой мыши и из контекстного меню выберите способ действия кнопки **Mechanical Action»Latch When Released**.

Блок-диаграмма

4. Перейдите на блок-диаграмму, выбрав на линейке инструментов **Window»Show Diagram**.

5. Создайте блок-диаграмму, показанную ниже:



a. Поместите на блок-диаграмму цикл **While Loop**, расположенный в палитре **Functions»Programming»Structures**. Наведите курсор на терминал условия выхода, нажмите на нем правой кнопкой мыши и выберите пункт **Stop If True**. Затем внутрь цикла поместите все элементы управления и кнопку STOP.

b. Добавьте на блок-диаграмму узел формул **Formula Node**, находящийся в палитре **Functions»Programming»Structures**, и введите в него формулу, генерирующую синусоидальный сигнал.

c. Добавьте функцию **Wait Until Next ms Multiple**, находящуюся в палитре **Functions»Programming»Timing**. Щелкните правой кнопкой мыши по полю ввода и выберите пункт **Create»Constant**. Созданной константе присвойте значение 50.

d. Поместите на блок-диаграмму ссылку на общую переменную. Для этого выберите пункт **Shared Variable** в палитре **Functions»Programming»Structures**. После этого нажмите правой кнопкой мыши на появившуюся ссылку на переменную и из контекстного меню выберите строку **Select Variable…** В появившемся диалоговом окне выберите общую переменную,

на которую Вы хотите сделать ссылку. Нажмите кнопку OK. Затем снова нажмите правой кнопкой мыши на ссылку на переменную и из контекстного меню выберите строку **Change To Write**.

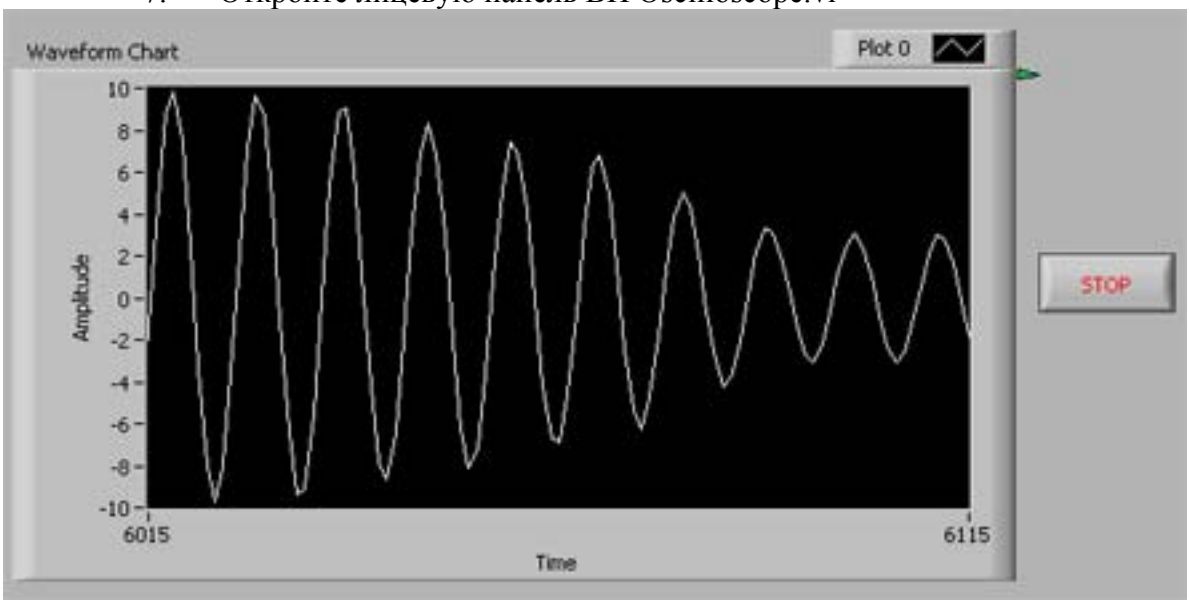
e. Используя инструменты ПЕРЕМЕЩЕНИЕ и СОЕДИНЕНИЕ завершите построение блок-диаграммы.

6. Сохраните ВП.

Работа с ВП Oscilloscope.vi

Лицевая панель

7. Откройте лицевую панель ВП Oscilloscope.vi



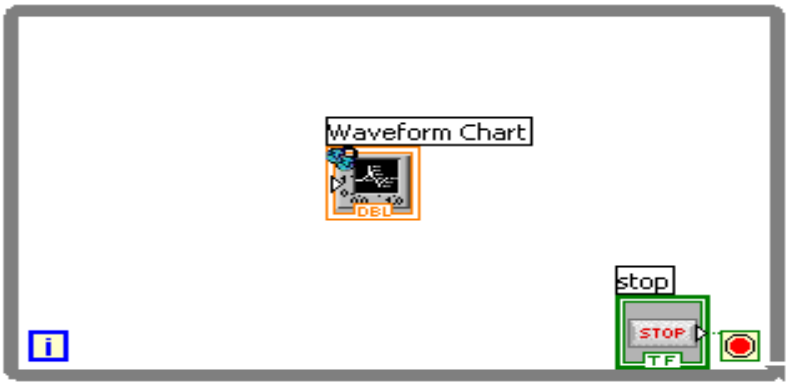
8. Поместите на лицевую панель элемент отображения графика **Waveform Chart**, находящийся в палитре **Controls»Modern»Graph**. Нажмите правой кнопкой мыши на ось Y и в контекстном меню уберите галочку с пункта AutoScale Y. Затем установите диапазон изменения данных по оси Y от -10 до 10. По оси X установите диапазон изменения данных от 0 до100.

9. Поместите на лицевую панель кнопку STOP. Выберите её в палитре **Controls»Modern»Boolean**. Нажмите на неё правой кнопкой мыши и из контекстного меню выберите способ действия кнопки **Mechanical Action»Latch When Released**.

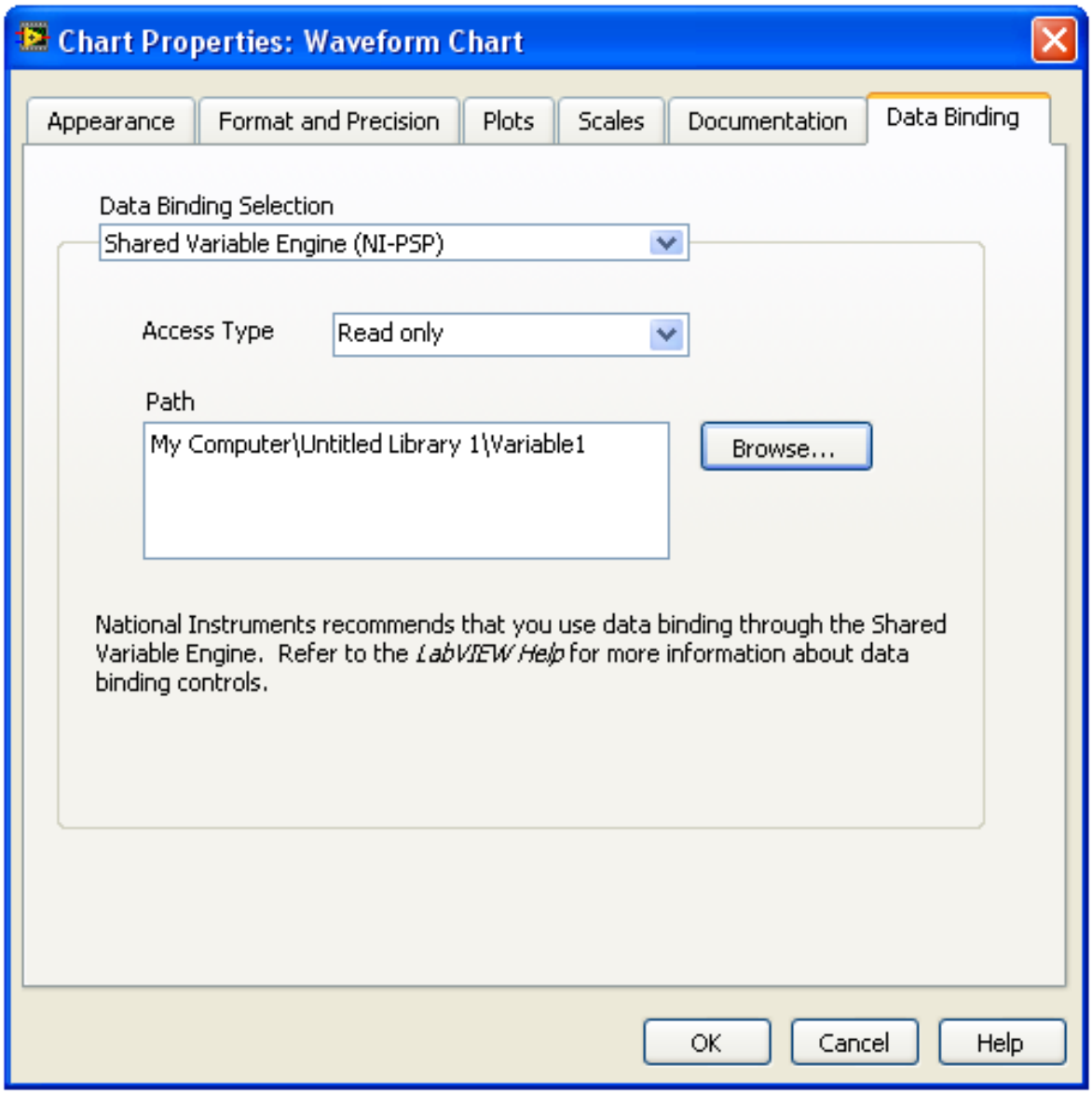
Блок-диаграмма

10. Перейдите на блок-диаграмму, выбрав на линейке инструментов **Window»Show Diagram**.

11. Создайте блок-диаграмму, показанную ниже:



a. Поместите на блок-диаграмму цикл **While Loop**, расположенный в палитре **Functions»Programming»Structures**. Наведите курсор на терминал условия выхода, нажмите на нем правой кнопкой мыши и выберите пункт **Stop If True**. Затем внутрь цикла поместите элемент отображения данных Waveform Chart и кнопку STOP.



	b. Соедините элемент отображения данных **Waveform Chart** с общей

	переменной. Для этого подведите курсор мыши к элементу отображения данных, нажмите правую кнопку мыши и из контекстного меню выберите строку **Properties**. В появившемся диалоговом окне выберите закладку **Data Binding**. Установите на этой закладке следующие параметры: в строке **Data Binding Selection** установите пункт **Shared Variable Engine (NI-PSP)**, а в строке **Access Type – Read Only**. Используя кнопку, **Browse** введите путь к переменной, с которой Вы хотите связать элемент отображения данных. Нажмите кнопку OK.
	c. Используя инструменты ПЕРЕМЕЩЕНИЕ и СОЕДИНЕНИЕ завершите построение блок-диаграммы.

12. Сохраните ВП.

Запуск ВП.

1. Перейдите на лицевую панель ВП *Oscilloscope.vi* и запустите его.

2. Перейдите на лицевую панель ВП *Generator.vi* и запустите его.

Примечание Разместите лицевые панели обоих ВП так, чтобы можно было смотреть на них одновременно.

3. Изменяя значения элементов управления Амплитуда и Частота на лицевой панели ВП *Generator.vi*, убедитесь, что отображение данных происходит на лицевой панели ВП Oscilloscope.vi.

4. Остановите выполнение ВП *Generator.vi*, нажав кнопку STOP на лицевой панели.

5. Остановите выполнение ВП *Oscilloscope.vi*, нажав кнопку STOP на лицевой панели.

Конец упражнения 12-4.

Краткое изложение пройденного материала, советы и секреты

- Для настройки внешнего вида и поведения ВП следует выбрать пункт **VI Properties** из меню **File**. Можно также щелкнуть правой кнопкой мыши по иконке в правом верхнем углу лицевой панели и выбрать пункт контекстного меню **VI Properties**.
- Для отображения лицевой панели при запуске отдельного экземпляра подпрограммы ВП следует щелкнуть правой кнопкой мыши по подпрограмме ВП и из контекстного меню выбрать пункт **SubVI Node Setup**. В открывшемся диалоговом окне отметьте пункты **Show Front Panel when called** и **Close afterwards if originally closed**.
- Для отображения лицевой панели при запуске всех экземпляров подпрограмм ВП следует выбрать в меню **File** пункт **VI Properties**. В диалоговом окне **VI Properties** выберите пункт **Window appearance** из выпадающего меню **Category**, нажмите кнопку **Customize** и отметьте пункты **Show Front Panel when called** и **Close afterwards if originally closed**.
- Назначить «горячие» клавиши для элементов управления можно, щелкнув правой кнопкой мыши по элементу управления и выбрав пункт контекстного меню **Advanced»Key Navigation**.
- Для того, чтобы избежать проблем при редактировании ВП, перед изменением свойств ВП следует создавать резервную копию ВП в новой папке. Для этого выберите из главного меню пункт **File»Save with Options**.
- Для редактирования ВП с настройками, затрудняющими редактирование можно:
  - ⇒ «Сломать» подпрограмму ВП. ВП откроется в режиме редактирования, поскольку он не сможет работать с неисправной подпрограммой.
  - ⇒ Если подпрограмм у ВП нет, то следует поместить его на блок-диаграмму нового ВП.
- Самый простой способ добавить ВП или элемент управления в палитры **Functions** и **Controls** – это сохранить их в директорию *user.lib*.
- Для смены текущего вида палитры следует нажать кнопку **Options** на панели инструментов палитр **Functions** и **Controls**. В появившемся диалоговом окне **Controls/Functions Palette Options** выберите вид палитры из выпадающего меню **Palette View**.
- В выпадающем меню **Format** можно настроить вид палитры так, чтобы отображались только названия элементов (**text-only**) или только их иконки (**icon-only**).

- Используя общие переменные (**Shared Variable**), Вы можете читать и передавать данные между виртуальными приборами внутри проекта или используя сеть. Для создания общей переменной откройте проект LabVIEW. Затем в окне проекта нажмите правой кнопкой мыши на строку, отображающую устройство, либо на библиотеку проекта, либо на папку внутри библиотеки проекта и из контекстного меню выберите **New»Variable**.
- Изменять значения общей переменной можно как на лицевой панели, так и на блок-диаграмме.

Примечания