Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное
образовательное учреждение высшего образования
«Северо-Кавказский федеральный университет»

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

к лабораторным работам
по дисциплине «Программирование в среде LabVIEW»
для студентов направления подготовки
23.04.03 Эксплуатация транспортно-технологических машин и комплексов
(Направленность (профиль) «Техническая эксплуатация автомобилей»)

Основная задача методических указаний – предоставить студентам, изучающим дисциплину «Программирование в среде LabVIEW» методический материал для разработки программных моделей автоматических систем измерения в среде NI LabVIEW. Вторая, и не менее важная задача, – это популяризация самой среды как системы разработки программ.

Предназначен для студентов, обучающихся по направлению подготовки 23.04.03 Эксплуатация транспортно-технологических машин и комплексов, направленность (профиль) – Техническая эксплуатация автомобилей, квалификация выпускника – Магистр.

Содержание

Лабораторная работа № 1

ИЗУЧЕНИЕ ОСНОВНЫХ ПОНЯТИЙ ПРОГРАММНОЙ СРЕДЫ LabVIEW И ВИРТУАЛЬНОГО ПРИБОРА

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

Запуск среды программирования LabVIEW осуществляется либо двойным кликом мыши на ярлыке LabVIEW, который находится на рабочем столе, либо из раздела Пуск-Программы – National Instruments LabVIEW. При входе в главное меню LabVIEW пользователю предлагается создание нового виртуального инструмента (*New VI*) или открытие уже существующего (*Open VI*).

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

Разработка VI (ВП) осуществляется на двух панелях, находящихся в двух окнах, – передней (лицевая панель) и функциональной (блок-диаграмма). Лицевая панель – интерфейс пользователя 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.

После этого на блок-диаграмме ВП осуществляется программирование с использованием палитры Функций (*Functions*), которая включает графическое представление функций для управления объектами на лицевой панели.

Структура панелей одинакова. Основным элементом каждой панели является рабочая область, снабженная горизонтальным и вертикальным скролингами, в которой и размещаются элементы. Также на панелях имеются верхнее меню и набор управляющих кнопок (рис. 1.1):

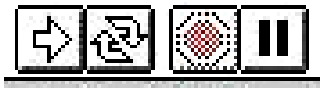


Рис. 1.1. Управляющие кнопки

- кнопка «стрелка» – пуск выполнения программы; если в программе имеются ошибки, то данная кнопка расколота на две части;
- кнопка «стрелки в цикле» – запуск программы в циклическом режиме;
- кнопка «красный круг» – остановка выполнения программы;
- кнопка «две вертикальные черты» – пауза в выполнении программы.

Рис. 1.2. Панель Tools

Для обеих панелей доступна панель *Tools Palette* (рис. 1.2), включающая набор управляющих кнопок для изменения режима редактирования. Перечислим некоторые из них:

- кнопка «указательный палец» служит для изменения позиций выключателей и кнопок, управления значениями цифровых регуляторов, настройки виртуальных осциллографов и др.;
- кнопка «стрелка» – выделение, перемещение объектов, изменение их размера;
- кнопка «А» – открытие и редактирование текстового окна;
- кнопка «катушка» служит для соединения объектов на функциональной панели;
- кнопка «кисть» – раскрашивание объектов или фона;
- кнопка «рука» – перемещение рабочей области панели в окне;
- кнопка «пипетка» – выбор текущего цвета из имеющихся на панели;
- кнопка «красный круг» – для размещения и снятия точек остановки выполнения программы на функциональной панели;
- кнопка «Р» – для размещения на функциональной панели локальных окон для отображения текущих значений данных, передаваемых в ходе выполнения программы.

Рис. 1.3. Панель Controls

При активной передней панели становится доступной панель *Controls* (рис. 1.3), которая вызывается либо щелчком правой кнопки мыши в рабочем пространстве лицевой панели, либо необходимо выбрать в пункте главного меню *Window* → *Show Controls Palette*. С ее помощью осуществляется визуальное размещение *элементов управления* и *элементов отображения* на лицевой панели ВП. В панели Controls они распределены по отдельным группам по некоторым признакам – числовые, логические, строковые, массивы, диалоговые, ActivX, Internet и др.

Рассмотрим основные подпанели панели Controls:

- *Numeric* (числовые значения). Состоит из элементов управления и элементов отображения для числовых данных;
- *Boolean* (булевы значения). Состоит из элементов управления и элементов отображения для булевых величин;
- *String&Table* (строковые значения и таблицы). Состоит из элементов управления и элементов отображения для ASCII строк и таблиц;
- *List & Ring* (списки и закольцованные списки). Состоит из элементов управления и элементов отображения для меню, выполненных в форме списков и закольцованных списков;

- *Array & Cluster* (массивы и кластеры). Состоит из элементов управления и элементов отображения для группировки наборов типов данных;
- *Graph* (виртуальные осциллографы). Состоит из элементов отображения для построения графиков данных в графах или диаграммах в реальном масштабе времени;
- *Path & Refnum* (пути и ссылки). Состоит из элементов управления и элементов отображения для путей и ссылок;
- *Decorations* (оформление). Состоит из элементов управления и элементов отображения графических объектов для настройки дисплеев лицевой панели;
- *Select Control* (выбор регулятора). Отображает диалоговое окно для загрузки самодельных элементов управления;
- *User Controls* (средства управления пользователя). Состоит из специальных средств управления, которые формирует сам пользователь;
- *ActiveX* (объекты ActiveX). Состоит из средств управления, позволяющих внедрить объекты ActiveX на лицевую панель;
- *Dialog* (диалоговая панель). Состоит из стандартных объектов для формирования диалога с пользователем;
- *IMAQ Vision* (обработка изображений). Состоит из средств обработки и анализа изображений;
- *Internet Toolkit* (работа с Internet). Состоит из средств управления, располагаемых на передней панели, позволяющих организовывать работу виртуальных инструментов в сети Internet (ftp, электронная почта, telnet, CGI и другие).

После помещения *элементов управления* или *отображения* данных на *лицевую панель* они получают свое графическое отображение (в виде терминала данных) на блок-диаграмме. Символы на терминале соответствуют типу данных терминала. Например, *DBL* – терминал представляет данные в виде вещественных чисел с двойной точностью, *TF* – логический терминал, *I16* – терминал 16-битных целых и др.

Рис. 1.4. Панель Functions

При активировании функциональной панели становится доступной палитра *Functions* (рис. 1.4), которая аналогично панели *Controls* включает систематизированные наборы стандартных элементов в виде отдельных пиктограмм, из которых осуществляется составление блок-схемы ВП. Палитра *Functions* вызывается либо щелчком правой кнопки мыши в рабочем пространстве блок-схемы, либо путем выбора в пункте главного меню *Window* → *Show Function Palette*.

Рассмотрим основные подпанели панели Functions:

- *Structures* (структуры). Состоит из управляющих структур программы, таких как циклы For Loop, While Loop и др.;
- *Numeric* (числовые функции). Состоит из тригонометрических, логарифмических и других функций;
- *Boolean* (булевы функции). Состоит из логических и булевых функций;
- *String* (строковые функции). Состоит из функций для работы со строковыми величинами;
- *Array* (массивы). Состоит из функций для обработки массивов;
- *Cluster* (кластеры). Состоит из функций для обработки кластеров;
- *Comparison* (сравнение). Состоит из функций для сравнения переменных;
- *Time & Dialog* (время и диалог). Состоит из функций для диалоговых окон, синхронизации и обработки ошибок;
- *File I/O* (ввода/вывода файла). Состоит из функций для осуществления операций по вводу/выводу файлов;
- *Instrument I/O* (инструменты ввода/вывода). Состоит из ВП для связи и управления приборами различной архитектуры;
- *Instrument Drivers* (драйверы приборов). Состоит из ВП, способных управлять внешними приборами, осциллоскопами, генераторами и т.д., через последовательный порт или интерфейс GPIB;
- *Data Acquisition* (сбор данных). Состоит из ВП для использования плат сбора данных;
- *Signal Processing* (обработка сигналов). Состоит из ВП для генерации и обработки сигналов;
- *Mathematics* (математические). Состоит из оптимизационных, алгебраических, интегральных, дифференциальных и других функций;
- *Graphics & Sound* (графика и звук). Состоит из ВП для работы с трехмерной графикой, изображениями и звуком;
- *Communication* (связи). Состоит из виртуальных приборов для работы с сетями TCP, DDE и др.;
- *Application Control* (управление приложением). Состоит из ВП, управляющих виртуальными приборами;
- *Advanced* (расширенная). Состоит из разных функций типа функции библиотечного запроса, манипуляции данных и др.;
- *Report Generation* (генерация отчета). Состоит из ВП, используемых для подготовки отчетных документов;
- *Tutorial* (обучающие программы). Состоит из ВП, используемых в обучающей программе LabVIEW;
- *User Libraries* (пользовательские библиотеки). С помощью нее организуется быстрый доступ к нужному vi;
- *Select VI* (выбор ВП). Состоит из диалогового окна для внедрения подпрограмм в текущий ВП;
- *IMAQ Vision* (обработка изображений). Состоит из ВП, используемых для обработки и анализа изображений;
- *Image Acquisition* (получение изображения). Состоит из ВП, используемых для получения и обработки изображений;
- *Internet Toolkit* (работа с Internet). Состоит из ВП, используемых для работы в сети Internet (ftp, электронная почта, telnet, CGI и др.);
- *SQL* (SQL запросы). Состоит из ВП, используемых для организации связи с SQL сервером и обработки запросов.

Объекты блок-диаграммы включают графическое отображение элементов лицевой панели, операторов, функций, подпрограмм ВП, констант, структур и *проводников данных*, по которым производится обмен данными между объектами блок-диаграммы.

Проводники данных между терминалами аналогичны переменным на обычных языках. Данные идут в только одном направлении, с исходного терминала на один или более терминалов адресата. Провода имеют различную толщину и цвет. Синий цвет соответствует целым числам, оранжевый – вещественным числам, зеленый – логическим, лиловый – строковым данным и т.д.

При нажатии правой кнопки мыши на регуляторе/индикаторе (как на передней, так и на функциональной панели) появляется *контекстное меню*, с помощью которого возможно осуществить:

- замену элемента управления (регулятора) на элемент отображения (индикатора) и наоборот (Change to Control, Change to Indicator);
- быстрый поиск терминала на функциональной панели (Find Terminal) и регулятора/индикатора на передней панели (Find Control, Find Indicator);
- демонстрацию или отказ от названия для описания регулятора/индикатора (Show → Label, Show → Caption);
- настройку параметров регулятора/индикатора (Data Operations);
- замену на другой регулятор/индикатор (Replace);
- получение справки по используемой функции (Online Help);
- открытие для функций соответствующих им констант, индикаторов и регуляторов (Create Constant, Create Indicator, Create Control) и др.

Задание 1.1. ВП *Частотный анализ*

1. Запустите LabVIEW Пуск → Программы → NationaI 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.Ilb и дважды щелкните на Frequency Response.VI (Частотная характеристика). Появится лицевая панель ВП «Частотный анализ» (рис. 1.5).

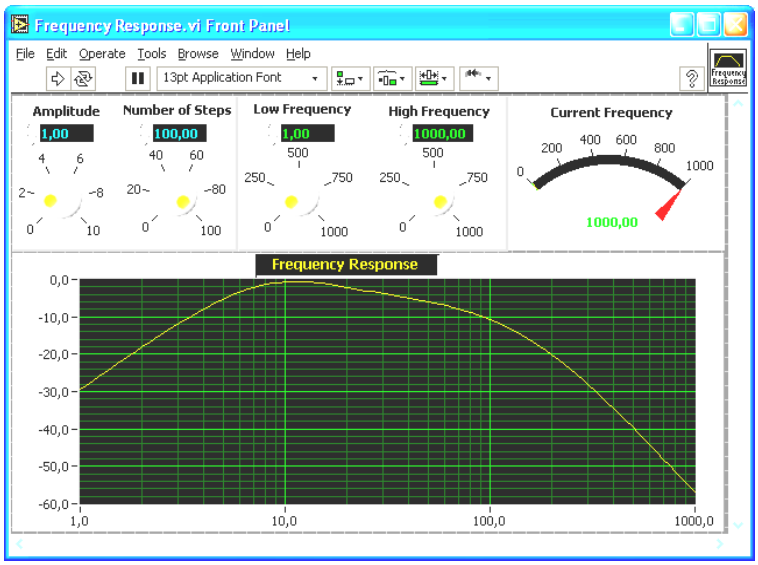


Рис. 1.5. ВП «Частотный анализ»

4. 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.

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

Блок-диаграмма

6. Перейдите на блок-диаграмму. Для этого выберите в главном меню Window → Show Diagram или введите <Ctrl–E> с клавиатуры.
7. Откройте окно контекстной справки, выбрав в пункте главного меню Help → Show Context Help. Получить информацию об объекте в окне контекстной справки Context Help можно, наведя на них курсор;

а) Поместите инструмент ПЕРЕМЕЩЕНИЕ над функцией Logarithm Base 10, расположенной под меткой Bode Plot. В окне контекстной справки Context Help появится описание функции;

б) В окне контекстной справки Context Help нажмите кнопку More Help  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. Здесь можно получить подробное описание и других функций;

в) Наведите инструмент СОЕДИНЕНИЕ  на поля ввода/вывода данных функции Logarithm Base 10. Соответствующие поля в окне контекстной справки Context Help начнут мигать;

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

8. Чтобы зафиксировать текущее окно Context Help (контекстной справки), необходимо либо нажать кнопку  , либо выбрать в пункте главного меню Help → 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.

Контрольные вопросы

1. Из каких основных компонентов состоит ВП?
2. Что понимается под интерфейсом пользователя ВП?
3. Какие палитры доступны для лицевой панели?
4. Какие палитры доступны для блок-диаграммы?
5. Что представляет собой лицевая панель?
6. Каково назначение блок-диаграммы?
7. Из каких подпалитр состоит палитра Controls (Элементов)?
8. Из каких подпалитр состоит палитра Functions (Функций)?
9. На каких панелях осуществляется разработка ВП?
10. Назовите назначение управляющих кнопок на блок-диаграмме.
11. Назовите назначение управляющих кнопок на лицевой панели.
12. Что такое элемент управления и элемент отображения?
13. Назовите основные типы данных.
14. Что такое проводник данных?
15. Каким образом осуществляется вызов контекстной справки?
16. Как можно зафиксировать текущее окно контекстной справки?
17. Назовите назначение контекстного меню.

Лабораторная работа № 2

СОЗДАНИЕ, РЕДАКТИРОВАНИЕ И ОТЛАДКА ВИРТУАЛЬНОГО ПРИБОРА

Цель занятия:

- изучить компоненты ВП;
- создать ВП (преобразовать °C в °F);
- изучить типы данных и проводники данных;
- отредактировать ВП;
- приобрести практические навыки отладки ВП.

ТЕОРЕТИЧЕСКИЕ УКАЗАНИЯ

 Объекты лицевой панели на блок-диаграмме отображаются в виде терминалов данных (графическое изображение прямоугольной формы с буквенно-численными обозначениями). 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.

 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 (Вариант), цикл While (цикл по условию) и т.д. Узлы Add (Сложение) и Subtract (Вычитание) – узлы функций.

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 два элемента отображения. Компонентами, определяющими совместимость соединения, являются: тип данных элемента управления и/или отображения и тип данных поля ввода/вывода.

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 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.

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.

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

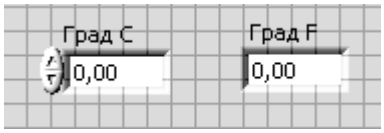
Соединение объектов проводниками данных вручную. Соединение объектов проводниками данных вручную производится с помощью инструмента СОЕДИНЕНИЕ. После наведения инструмента СОЕДИНЕНИЕ на поле ввода или вывода данных на экране появляется подсказка, которую можно использовать для уточнения места подключения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

З а д а н и е 2.1. Преобразование °C в °F

Ниже приведена последовательность действий для создания ВП, который будет преобразовывать значение температуры из градусов Цельсия (°C) в температуру по Фаренгейту (°F).

Лицевая панель

1. Выберите пункт главного меню File → New → VI, чтобы открыть новую лицевую панель.
2. Поместите цифровой элемент управления на лицевую панель. В поле собственной метки элемента управления напечатайте «Град. C».
3. Поместите элемент отображения данных на лицевую панель. Он будет использован для отображения значений температуры в °F. В поле собственной метки элемента управления напечатайте «Град. F» и щелкните мышью в свободном пространстве лицевой панели или нажмите кнопку Enter.

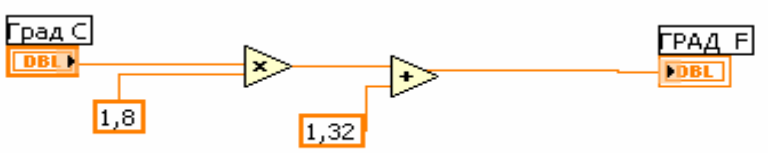


На блок-диаграмме LabVIEW создаст терминалы данных, соответствующие элементам управления и отображения. Терминалы данных представляют тип данных соответствующих элементов. Например, терминал данных DBL представляет тип числовых данных двойной точности с плавающей запятой.

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

Блок-диаграмма

4. Перейдите на блок-диаграмму, выбрав пункты главного меню Window → Show Diagram.
5. Создайте блок-диаграмму, показанную ниже:



6. Выберите функцию Multiply (Умножение) из палитры Функций в разделе Functions → Numeric (Арифметические функции). Поместите ее на блок-диаграмму.

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

8. Выберите числовую константу из палитры Функций в разделе Functions → 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.

9. Соедините объекты блок-диаграммы с помощью инструмента СОЕДИНЕНИЕ.

10. Перейдите на лицевую панель, выбрав в главном меню пункт Window → Show Panel.

11. Сохраните ВП, он будет использоваться позднее.

Запуск ВП

1. Введите число в элемент управления и запустите ВП:

а) для ввода числа в элемент управления следует использовать инструмент УПРАВЛЕНИЕ или инструмент ВВОД ТЕКСТА;

б) нажмите кнопку Run, чтобы запустить ВП;

в) введите несколько разных значений температуры и запустите ВП.

2. Закройте ВП, выбрав пункт главного меню File → Close.

З а д а н и е 2.2. Создать ВП согласно Вашему варианту

№ варианта	Содержание задания
1	ВП преобразует значение температуры из градусов Цельсия в температуру по шкале Кельвина (°К = °С + 273°)
2	ВП преобразует значение температуры из градусов Цельсия в температуру по шкале Реомюра (°R = °С · 4/5)
3	ВП преобразует значение температуры по шкале Кельвина в градусы Цельсия (°С = ° К – 273°)
4	ВП преобразует значение температуры по шкале Реомюра в градусы Цельсия (°С = °R · 5/4)
5	ВП преобразует значение температуры по Фаренгейту в градусы Цельсия (°F = 9/5 · °С + 32°)
6	ВП преобразует значение температуры по шкале Реомюра в температуру по Фаренгейту (°F = 9/4 · °R + 32°)
7	ВП преобразует значение напряжения и силы тока по закону Ома в сопротивление ($R = U/I$)
8	ВП преобразует значение температуры по Кельвину в температуру по Реомюру (°R = (°К – 273°) · 4/5)
9	ВП преобразует по закону Ома значение напряжения и сопротивления в силу тока ($I = U/R$)
10	ВП преобразует значение динамической вязкости μ в кинематическую ν вязкость ($\nu = -\mu/\rho$)
11	ВП преобразует значения напряжения (мВ) и силы тока (мкА) в мощность (Вт) ($P = I \cdot U$)
12	ВП преобразует значения напряжения (В) и силы тока (А) в мощность (Вт) ($P = I \cdot U$)
13	ВП преобразует значения массы (кг) и времени (с) в массовый расход (кг/с) ($G = m/t$)
14	ВП преобразует значения массы (кг) и объема ($м^3$) в плотность ($кг/м^3$) ($\rho = m/V$)

15	ВП преобразует значения объема ($м^3$) и времени (с) в объемный расход ($м^3/с$) ($G = V/t$)

Контрольные вопросы

1. Из каких основных компонентов состоит Ваш ВП?
2. Какие узлы на блок-диаграмме ВП Вы знаете?
3. Как отображаются терминалы данных и какие функции они выполняют?
4. Что такое проводник данных?
5. Какие проводники данных Вы знаете?
6. Какие типы данных Вы знаете?
7. Как соединяются объекты проводниками данных?
8. Из каких подпалитр состоит палитра Controls (Элементов)?
9. Из каких подпалитр состоит палитра Functions (Функций)?
10. Как осуществляется запуск разработанного ВП?
11. Назовите назначение управляющих кнопок на блок-диаграмме.
12. Назовите назначение управляющих кнопок на лицевой панели.
13. Что такое элемент управления и элемент отображения?
14. Назовите основные типы данных.
15. Как ввести число в элемент управления?

Лабораторная работа № 3

СОЗДАНИЕ ПОДПРОГРАММ ВИРТУАЛЬНОГО ПРИБОРА

Цель занятия:

- изучить подпрограмму ВП;
- создать иконку и настроить соединительную панель для возможности использования ВП в качестве подпрограммы ВП.

ТЕОРЕТИЧЕСКИЕ УКАЗАНИЯ

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*. Подпрограмма ВП соответствует подпрограмме в текстовых языках программирования. Использование подпрограмм ВП помогает быстро управлять изменениями и отладкой блок-диаграмм.

Любой ВП может быть использован как подпрограмма при создании в последующем других виртуальных инструментов. Для объединения нескольких функциональных блоков разрабатываемой блок-диаграммы в подпрограмму достаточно выделить их мышкой на диаграмме, удерживая клавишу Shift, и затем выбрать в верхнем меню пункт Edit – Create SubVI. При этом они объединятся в новую подпрограмму с новым значком (иконкой) на функциональной панели. Двойной клик на данном значке позволит вызвать созданную подпрограмму, настроить ее должным образом и сохранить с заданным именем. В последующем данный модуль может быть многократно использован в различных ВП.

Каждый виртуальный прибор в правом верхнем углу лицевой панели и в окне блок-диаграммы отображает иконку  . 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.

Для редактирования иконки создаваемых ВП и подпрограмм достаточно кликнуть правой кнопкой мыши на пиктограмме ВП в правом верхнем углу и выбрать пункт Edit Icon … (рис. 3.1).

С помощью простейших функций графического редактора можно создать собственный вариант иконки. Настройка входов/выходов (терминалов) подпрограмм осуществляется следующим образом. Необходимо нажать правой кнопкой мыши на пиктограмме vi в правом верхнем углу и выбрать пункт Show Connector… (рис. 3.2). При этом пиктограмма разделится на несколько прямоугольников, общий набор и вид которых можно редактировать с помощью всплывающего меню пиктограммы (добавить/удалить терминал – Add Terminal/Remove Terminal,

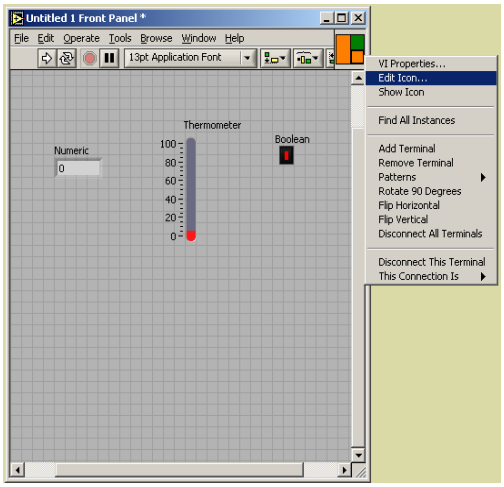


Рис 3.1. Выбор пункта Edit Icon

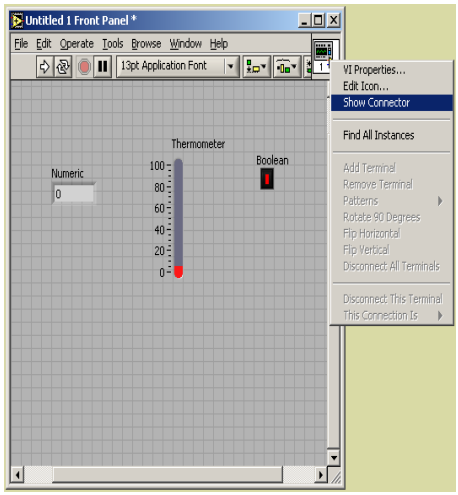


Рис 3.2. Выбор пункта Show Connector

поворот на 90 градусов – Rotate 90 Degrees, другой вид – Patterns… и др.). Для того, что бы сопоставить каждый терминал с определенными данными, необходимо левой кнопкой мыши кликнуть на нужном терминале, а затем – на том элементе управления или отображения на лицевой панели, которой он будет соответствовать. При этом терминал окрасится в цвет, соответствующий типу данных указанного элемента управления или отображения. В результате все терминалы будут связаны с определенными входными или выходными данными.

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.

З а д а н и е 3.1. ВП *Преобразования °C в °F*

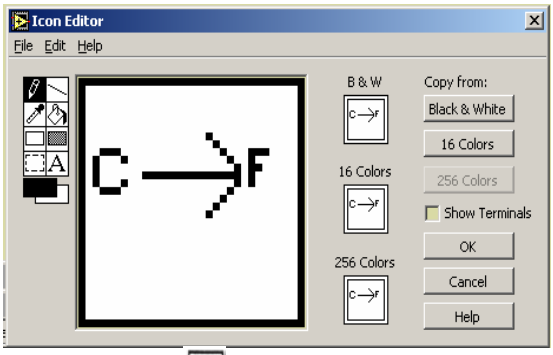
Ниже представлена последовательность действий по созданию иконки и настройке соединительной панели для созданного ВП, который переводит значение измеренной температуры из градусов Цельсия в градусы по Фаренгейту.

Лицевая панель

1. Откройте файл с ранее созданным ВП *Преобразование C в F* (*начало*).*vi*.

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 .
4. Нажав кнопку <Delete>, очистите область редактирования иконки.
5. Дважды щелкните по инструменту ПРЯМОУГОЛЬНИК , чтобы обвести область редактирования границей выбранного цвета.
6. Создайте следующую иконку:



а) введите текст инструментом ВВОД ТЕКСТА ;

б) напечатайте «C» и «F»;

в) для выбора размера шрифта дважды щелкните левой кнопкой мыши по инструменту ВВОД ТЕКСТА;

г) чтобы нарисовать стрелку, воспользуйтесь инструментом КАРАНДАШ ;

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

д) для передвижения текста и стрелки по полю редактирования иконки используйте инструмент ВЫБОР и стрелки на клавиатуре;

е) в разделе *Copy from* (Копировать из) 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;

ж) в разделе *Copy from* (Копировать из) выберите *16 Colors* и *256 Colors*;

з) 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.

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

а) в пункте главного меню *Help* (Помощь) выберите *Show Context Help* (показать контекстную справку) и выведите на экран окно *Context Help* (контекстной справки) для просмотра соединений;

б) щелкните левой кнопкой мышки на левом поле соединительной панели. Инструмент УПРАВЛЕНИЕ автоматически поменяется на инструмент СОЕДИНЕНИЕ, а выбранное поле окрасится в черный цвет;

в) щелкните левой кнопкой мыши по элементу *Град C*. Левое поле станет оранжевым и выделится маркером;

г) щелкните курсором по свободному пространству. Маркер исчезнет, и поле окрасится в цвет данных типа соответствующего элемента управления;

д) щелкните левой кнопкой мыши по правому полю соединительной панели и элементу *Град. F*. Правое поле станет оранжевым;

е) щелкните курсором по свободному пространству. Оба поля останутся оранжевыми;

ж) 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*.

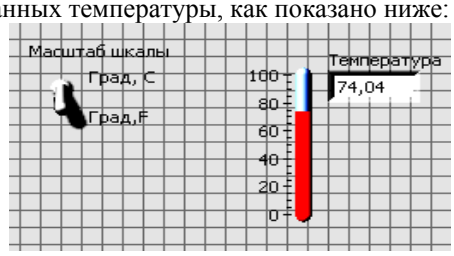
10. Выберите пункт главного меню *File* → *Close*. Закройте ВП.

З а д а н и е 3.2. ВП *Термометр*

Ниже приведена последовательность действий для создания ВП, который измеряет температуру и отображает значение температуры в градусах Цельсия или температуру по Фаренгейту.

Лицевая панель

1. Создайте элемент отображения данных температуры, как показано ниже:



а) выберите элемент отображения данных, расположенный на палитре Controls в разделе *Numeric* (Числовые элементы) ;

б) напечатайте «Температура» внутри собственной метки и нажмите кнопку Enter на инструментальной панели ;

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

2. Создайте элемент управления в виде вертикального переключателя :

а) выберите вертикальный переключатель, расположенный в палитре Controls раздела Boolean (Логические элементы) ;

б) введите имя собственной метки переключателя Масштаб шкалы и нажмите кнопку Enter на инструментальной панели;

в) используя инструмент ВВОД ТЕКСТА, создайте на лицевой панели свободную метку °C, как показано выше;

г) с помощью инструмента ВВОД ТЕКСТА создайте на лицевой панели свободную метку °F, как показано выше.

3. Создайте описание ВП, которое появляется в окне контекстной справки Context Help после наведения курсора на иконку ВП:

а) выберите пункт главного меню File → VI Properties;

б) выберите пункт Documentation (Описание) в разделе Category (Категория) из выпадающего меню;

в) в поле ввода текста напечатайте следующее:

Этот ВП измеряет температуру, используя ВП Demo Read Voltage VI.

4. Создайте описание элементов управления и отображения данных, которое появляется в окне контекстной справки *Context Help* после наведения на них курсора:

а) щелкните правой кнопкой мыши по элементу отображения и выберите пункт контекстного меню Description and Tip (Описание и предупреждения);

б) в поле ввода текста напечатайте следующее:

Выводит на экран значения измеренной температуры;

в) введите в поле Tip значение Температура;

г) нажмите кнопку ОК;

д) щелкните правой кнопкой мыши по элементу управления и выберите пункт контекстного меню Description and Tip (Описание и предупреждения);

е) в поле ввода текста напечатайте следующее:

Определяет шкалу (по Фаренгейту или Цельсию), используемую для измерения температуры;

ж) введите в поле Tip значение шкала – °C или °F и нажмите кнопке ОК.

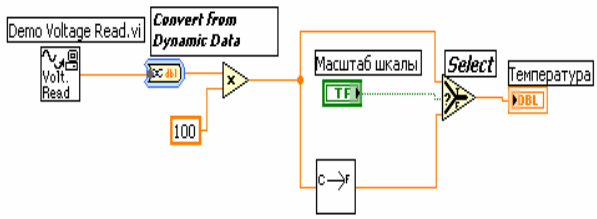
5. Отобразите окно контекстной справки *Context Help*, которое доступно из пункта главного меню Help → Show Context Help.

6. Наведите курсор на один из объектов для просмотра описания их работы в окне Context Help.

Блок-диаграмма

7. Перейдите на блок-диаграмму, выбрав *Window → Show Diagram.*

8. Создайте блок-диаграмму, показанную ниже.



 Поместите на блок-диаграмму ВП Demo Read Voltage VI, расположенный в каталоге I:\Texts\AICiI\for_LabVIEW, который служит для имитации считывания напряжения, пропорционального температуре. Например, если температура составляет 20 °C, то напряжение на выходе датчика будет равно 20 В.

 Поместите на блок-диаграмму ВП Convert from Dynamic Data (преобразовать динамические данные), расположенный в палитре *Functions → Express → Signal Manipulation*. Этот ВП преобразует динамический тип данных. В конфигурационном диалоговом окне выберите пункт Single Scalar in списка Resulting data type.

 Выберите функцию Multiply (Умножение), расположенную в палитре *Functions → 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 у функции Multiply (Умножение) и в контекстном меню выберите пункт *Create → Constant* (Создать константу). Константе присвойте значение «100» и нажмите клавишу <Enter>.

 В палитре Functions (Функций) в разделе Select a VI (Выбор ВП) выберите ВП Преобразование °C в °F, созданный в задании 3.1. Поместите его на блок-диаграмму. Этот ВП переведет градусы Цельсия в градусы Фаренгейта.

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

Лицевая панель

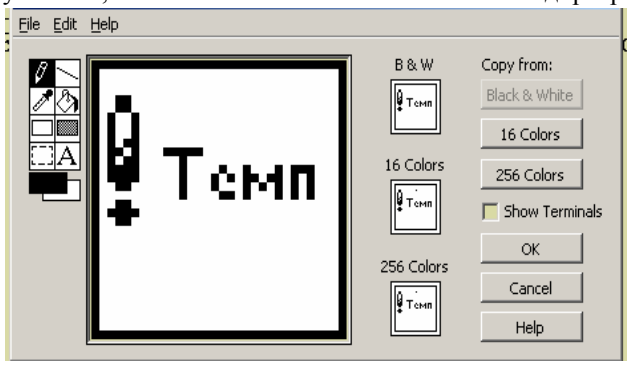
9. Перейдите на лицевую панель.

10. Нажмите на кнопку непрерывного запуска, показанную слева.

11. Нажмите на кнопку непрерывного запуска еще раз для остановки ВП.

Иконка и соединительная панель

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



13. 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).

14. Сохраните ВП под именем Термометр, он будет использоваться позднее.

15. Закройте ВП, выбрав пункт главного меню File → Close.

З а д а н и е 3.3. Аналогично заданию 3.2 создать ВП, используя в качестве подпрограммы прибор, созданный в задании 2.2

№ варианта	Содержание задания
1	ВП измеряет температуру и отображает значение температуры по шкале Кельвина или в градусах Цельсия
2	ВП измеряет температуру и отображает значение температуры по шкале Реомюра или в градусах Цельсия
3	ВП измеряет температуру и отображает значение температуры в градусах Цельсия или по шкале Кельвина
4	ВП измеряет температуру и отображает значение температуры в градусах Цельсия или по шкале Реомюра
5	ВП измеряет температуру и отображает значение температуры в градусах Цельсия или по шкале Фаренгейта
6	ВП измеряет температуру и отображает значение температуры по Реомюру или по шкале Фаренгейта
7	ВП измеряет напряжение; на выходе – значение напряжения или сопротивления (рассчитанного по закону Ома) в зависимости от состояния переключателя
8	ВП измеряет температуру и отображает значение температуры по Кельвину или по шкале Реомюра
9	ВП измеряет напряжение; на выходе – значение напряжения или силы тока (рассчитанной по закону Ома) в зависимости от состояния переключателя
10	ВП измеряет вязкость и отображает значение динамической μ или кинематической вязкости ν
11	ВП измеряет напряжение и силу тока и отображает напряжения (мВ) или мощность (Вт)
12	ВП измеряет напряжение и силу тока и отображает напряжение (В) или мощность (Вт)
13	ВП измеряет значения массы и объема и отображает массу (кг) или расход (кг/с)
14	ВП измеряет объем и массу и отображает объем ($м^3$) или плотность ($кг/м^3$)
15	ВП измеряет объем и время и отображает значение объема ($м^3$) или объемного расхода ($м^3/с$)

Примечания: 1) Для имитации считывания напряжения, пропорционального температуре в вариантах № 1 – 9, 11, 12, следует использовать ВП Demo Read Voltage VI, расположенный в каталоге i:\Text\AICI\for_LabVIEW; 2) в вариантах № 10, 13 – 15 для имитации считывания значений вязкости, массы, объема, времени и силы тока – функцию Random Number (0-1), расположенную в палитре Functions → Numeric. Эта функция будет генерировать случайные числа в пределах от 0 до 1.

Контрольные вопросы

1. Из каких основных компонентов состоит Ваш ВП?
2. Что называется иконкой ВП?
3. Как создать иконку ВП?
4. Что такое соединительная панель ВП?
5. Как настроить соединительную панель ВП?
6. Как редактируется иконка ВП?
7. Как поместить подпрограмму ВП на блок-диаграмму?
8. Зачем нужна функция Select?
9. Как осуществляется непрерывный пуск ВП?
10. Поясните назначение каждого инструмента, используемого для редактирования иконки в Icon Editor (Редакторе иконки).

Лабораторная работа № 4

МНОГОКРАТНЫЕ ПОВТОРЕНИЯ И ЦИКЛЫ ПРИ СОЗДАНИИ ВИРТУАЛЬНОГО ПРИБОРА В СРЕДЕ LabVIEW

Цель занятия:

- изучение основных структур среды LabView;
- измерение температуры с интервалом 1 секунда в течение одной минуты;
- использование сдвиговых регистров и узлов обратной связи для организации доступа к значениям на предыдущих итерациях цикла For (с фиксированным числом итераций);
- изучение организации доступа к значениям предыдущих итераций цикла.

ТЕОРЕТИЧЕСКИЕ УКАЗАНИЯ

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 пять структур: Цикл While (по условию), Цикл For (с фиксированным числом итераций), структура Case (Вариант), структура Sequence (‘сиквенс) (Последовательность), структура Event (Событие), а также Formula Node (узел Формулы).

Цикл While (по условию). Цикл While (по условию) работает до тех пор, пока не выполнится логическое условие выхода из цикла.

 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.

 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).

Цикл For (с фиксированным числом итераций). Цикл For (с фиксированным числом итераций) выполняет повторяющиеся операции над потоком данных определенное количество раз.

 Цикл For расположен в палитре Функций в разделе Functions → → 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.

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 (Добавить сдвиговый регистр).

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.

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 рис. 4.1 показано использование двух инициализированных сдвиговых регистров.

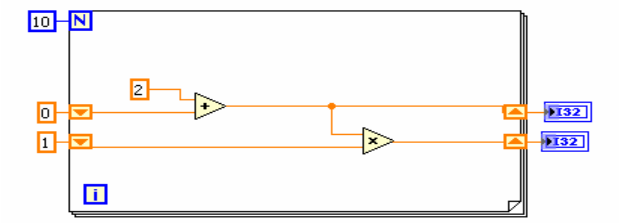


Рис. 4.1. Использование сдвиговых регистров в цикле For

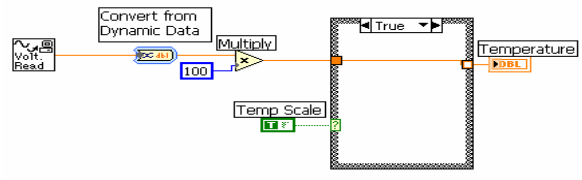
Узлы обратной связи.  Узел обратной связи, показанный слева, автоматически появляется в циклах While или For при соединении поля вывода данных подпрограммы ВП, функции или группы подпрограмм ВП и функций с полем ввода данных тех же самых подпрограмм ВП, функций или их групп. Как и сдвиговый регистр, 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 (Узел обратной связи) в палитре Structures (Структуры). 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.

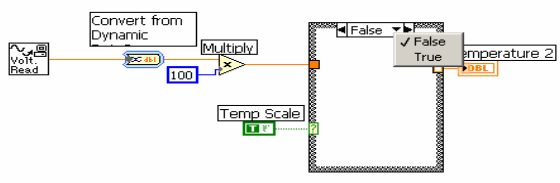
Структура выбора Case. В структуре выбора Case имеются две или более встроенных блок-схемы. Выбор одной из них определяется в зависимости от значения, поданного на вход данной структуры. Структура Case включает:

- Терминал выбора ?. Значение, подаваемое на него, может быть целочисленным, логическим или строковым.
- Переключатель блок-схем (True \ False \ и т.д.). Позволяет переходить от одной блок-схемы к другой. Содержит по умолчанию два окна True и False. При необходимости количество блок-схем выбора может быть увеличено. Кроме True и False, в качестве значений переключателя могут использоваться целые числа или строковые значения.

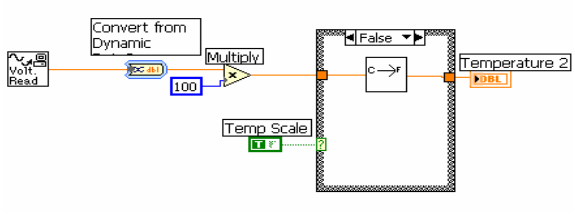
В качестве примера использования структуры Case вместо функции Select приведена измененная блок-диаграмма ВП Термометр (рис. 4.2). На переднем плане структуры Case показан логический вариант TRUE (рис. 4.2, *а*).



а)



б)



в)

Рис. 4.2. Применение структуры Case в ВП Термометр:
а – логический вариант TRUE; *б* – выбор варианта;
в – логический вариант FALSE

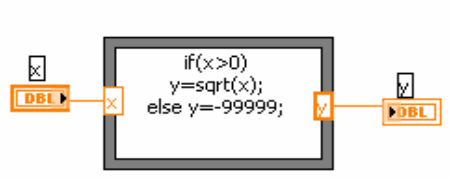
Определение варианта осуществляется либо выбором значения на селекторе структуры Case, либо вводом значения с помощью инструмента ВВОД ТЕКСТА (рис. 4.2, *б*).

Выбранный вариант появляется на переднем плане, как показано на блок-диаграмме (рис. 4.2, *в*).

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.

Формульный блок Formula Node. Формульный блок Formula Node позволяет вводить формулы в обычном виде прямо в блок-схему. Особенно это удобно, когда выражение имеет много переменных и сложный вид. Формулы вводятся как простой текст. При этом создаются терминалы на границе блока (контекстное меню Add Input или Add Output), куда вписываются имена переменных. Каждое выражение заканчивается разделителем «;».

Узел Формулы может также использоваться для принятия решений. На следующей блок-диаграмме показан способ применения операторов if- then в узле Формулы.



Последовательная структура Sequence Structure. Последовательная структура Sequence Structure выполняет встроенные в нее блок-схемы последовательно в определенном порядке. Количество встроенных блок-схем определяется числом фреймов данной структуры. Их количество увеличивается при помощи контекстного меню – Add Frame After, Add Frame Before. Для передачи значений переменных из фрейма в фрейм используются локальные переменные структуры (контекстное меню – Add Sequence Local variable), создаваемые на границе фрейма. Данные, связанные с такой переменной, доступны во всех последующих фреймах и не доступны в предыдущих.

Задание 4.1. ВП *Измерение температуры во времени*

Ниже приведена последовательность действий для создания ВП, который использует ВП «Термометр» для измерения температуры 1 раз в секунду в течение одной минуты.

Лицевая панель

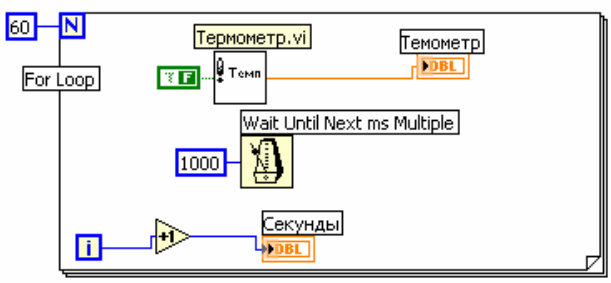
1. Откройте новый ВП и создайте лицевую панель, как показано ниже на рисунке:

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

Поместите на лицевую панель цифровой элемент отображения данных, расположенный в палитре *ControIs* → *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, в которой находится прибор, созданный в задании 3.2.

Щелкните правой кнопкой мыши по полю ввода данных *Temp Scale* (Шкала температур), 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* → *Time and Dialog*. Функция *Wait Until Next ms Multiple* (ждать кратного значения), показанная слева, обеспечивает интервал между итерациями, равный интервалу времени, необходимому для того, чтобы миллисекундный счетчик достиг значения, кратного введенному пользователем. 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* → *Numeric*. Эта функция добавляет 1 к значению счетчика итераций после завершения выполнения цикла.

3. Сохраните ВП под именем *Измерение температуры во времени*.
4. Запустите ВП.
5. Закройте ВП.

Задание 4.2. Создать ВП, согласно Вашему варианту

№ варианта	Содержание задания
1	Создайте ВП, генерирующий случайные числа в цикле While. Организуйте выход из цикла по нажатию кнопки на лицевой панели ВП
2	Посчитайте значение выражения $Y = X^2 + Z^3 - XZ + 10$ с помощью блока Formula Node и ВП Formula Express, расположенных в палитре *Functions* → *Arith*/*Compare* (арифметика/сравнение)
3	Создайте ВП, который с помощью Formula Node считает значение выражения $y = \sin(x)$, если y – положительное число, то $z = y + A$, иначе $z = y - A$
4	Создайте ВП, который с помощью Formula Node считает значение выражения $Y = x + \cos(x) - 10$, и если $Y >= 0$, то $Z = \text{sqrt}(Y)$
5	Используя структуру Case, создайте ВП, который считает разность 2-х чисел, и если полученное число ≥ 0 , то вычисляется значение корня, иначе выдается сообщение об ошибке
6	ВП осуществляет поочередное включение индикаторов на лицевой панели; промежутки между включениями индикаторов 2, 3 и 7 с, соответственно. Используйте последовательность Sequence Structure и функцию Time Delay, расположенную в палитре *Functions* → *Time and Dialog*
7	Создайте ВП, который измеряет температуру в течение минуты с помощью термометра, созданного в задании 3.2, и считает среднее значение температуры. Подсчет среднего значения осуществить с помощью сдвигового регистра
8	Посчитайте значение выражения $Y = X^5 + \cos^2(Z) - XZ + 10$ с помощью блока Formula Node и ВП Formula Express, расположенных в палитре *Functions* → *Arith*/*Compare* (арифметика/сравнение)

9	Используя структуру Case, создайте ВП, который считает значение выражения $y = ax + 14$, где a = const, и если $y \geq 0$, то вычисляется значение корня, иначе выдается сообщение об ошибке
10	Создайте ВП, который генерирует 70 случайных чисел и считает среднее значение. Подсчет среднего значения осуществите с помощью сдвигового регистра
11	Создайте ВП, генерирующий 70 случайных чисел в цикле FOR
12	ВП осуществляет поочередное включение индикаторов на лицевой панели; промежутки между включениями индикаторов 5, 8 и 12 с, соответственно. Используйте последовательность Sequence Structure и функцию Time Delay, расположенную в палитре *Functions* → *Time and Dialog*
13	Посчитайте значение выражения $Y = 10X^5 + \sin^2(Z) - XZV$ с помощью блока Formula Node и ВП Formula Express, расположенных в палитре *Functions* → *Arith*/*Compare* (арифметика/сравнение)
14	Создайте ВП, генерирующий случайные числа в цикле While. Организуйте выход из цикла по нажатию кнопки на лицевой панели ВП
15	Создайте ВП, который генерирует случайные числа до тех пор, пока одно из них не окажется равным значению, введенному в элемент управления

Контрольные вопросы

1. Из каких основных компонентов состоит Ваш ВП?
2. Для чего предназначена структура последовательности Sequence Structure? Как добавить фрейм в Sequence Structure?
3. Какие приемы использования цикла While Вы знаете?
4. Как измерить температуру с интервалом 30 с в течение 2 мин?
5. Как и зачем используются сдвиговые регистры в ВП?
6. Зачем нужны узлы обратной связи?
7. Как добавить 1 к значению счетчика итераций после завершения выполнения цикла?
8. Для чего предназначена структура выбора Case? Какие типы данных можно подавать на терминал выбора структуры Case?
9. Для чего предназначен формульный блок Formula Node? Как создать терминал на границе блока Formula Node?
10. Назовите назначение терминалов в цикле For.

Лабораторная работа № 5

РАБОТА С МАССИВАМИ В СРЕДЕ LabVIEW

Цель занятия:

- изучить типовые приемы создания массива элементов управления и отображения;
- изучить типовые приемы создания массива констант;
- изучить функции работы с массивами;
- освоить приемы работы с массивами.

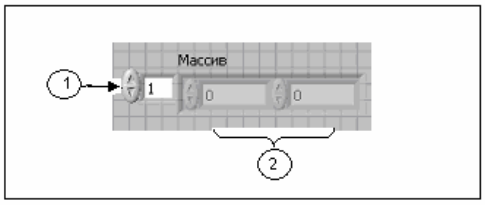
ТЕОРЕТИЧЕСКИЕ УКАЗАНИЯ

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 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.

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.

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 → Array & Cluster и поместить его на лицевую панель. Затем поместить в шаблон массива элемент управления либо отображения данных.



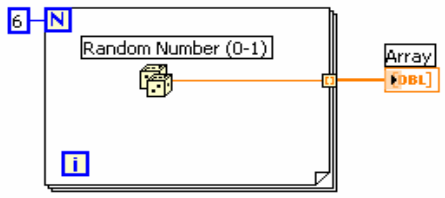
1. Элемент индекса массива 2. Элементы значений массива

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

Двумерные массивы. В двумерном (2D) массиве элементы хранятся в виде матрицы. Таким образом, для размещения элемента требуется указание индекса столбца и строки. Ниже 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 (Добавить размер). С этой целью также можно использовать инструмент ПЕРЕМЕЩЕНИЕ. Для этого надо просто изменить размер элемента индекса.

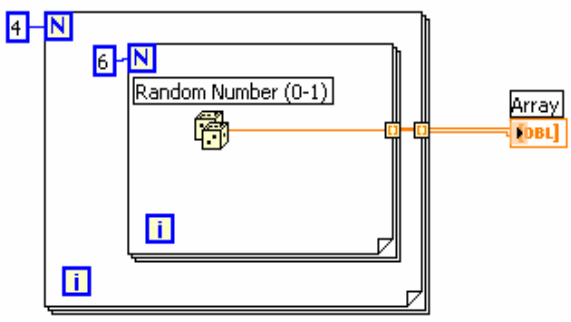
Автоматическая индексация. Цикл For и цикл While могут автоматически накапливать массивы и проводить их индексацию на своих границах. Это свойство называется автоиндексацией. После соединения терминала данных массива с терминалом выхода из цикла каждая итерация цикла создает новый элемент массива. На экране 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 (запретить автоиндексацию). 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 (разрешить автоиндексацию).

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.



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

 *Array Size* (Размер массива) – 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* (задать массив) – 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* (создать массив) – 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 (объединение входных данных).

7	4	2	4	4	4

 *Array Subset* (подмножество массива) – выдает часть массива, начиная с индекса, поступившего на поле *index*, и длиной, указанной в поле *length* (длина). Например, если подать предыдущий массив на поле ввода функции Array Subset, значение 2 – на поле index и 3 – на поле Подмножество:

2	4	4

 *Index Array* (индекс массива) – 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

Задание 5.1. ВП *Работа с массивами*

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:

а) в палитре *ControlIs* → *Array & Cluster* выберите шаблон массива;

б) созданному массиву присвойте имя *Массив случайных чисел*;

в) поместите внутрь шаблона массива цифровой элемент отображения, расположенный в палитре *Controls* → *Numeric*;

г) с помощью инструмента ПЕРЕМЕЩЕНИЕ измените размер массива таким образом, чтобы он содержал 10 элементов;

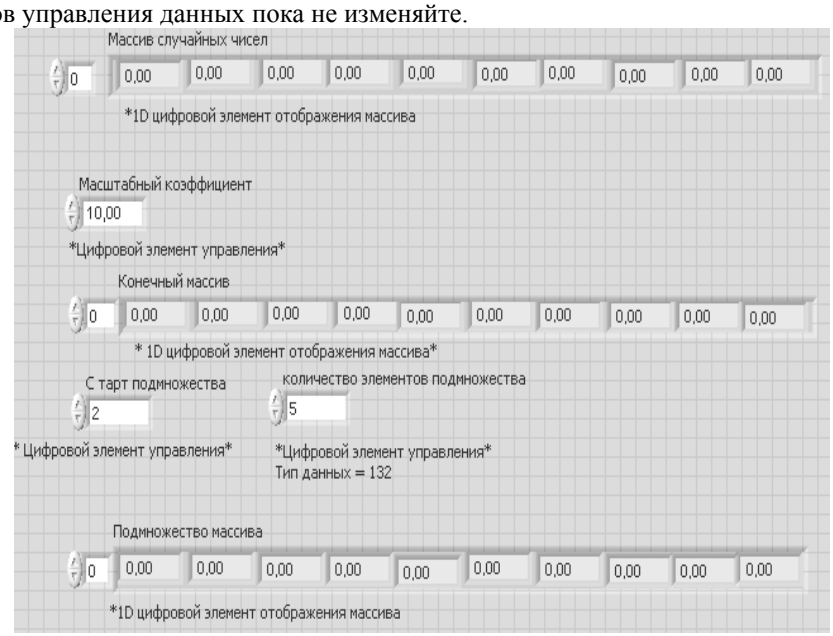
д) нажмите и удерживайте клавишу <CtrI> и, перемещая элемент *Массив случайных чисел*, создать две его копии;

е) копиям присвойте имена *Конечный массив* и *Подмножество массива*;

ж) создайте три цифровых элемента управления и присвойте им имена *Масштабный коэффициент*, *Старт подмножества, Количество элементов подмножества*;

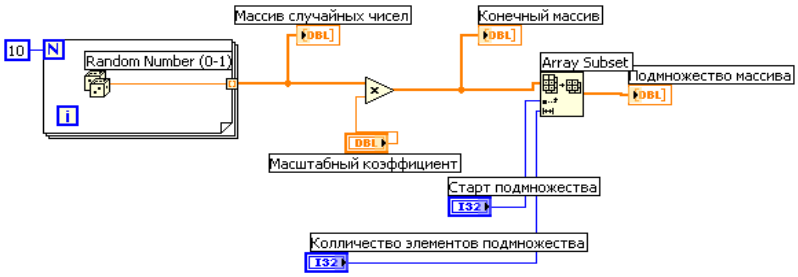
з) щелкните правой кнопкой мыши по элементам *Старт подмножества* и *Количество элементов подмножества*, в контекстном меню выберите пункт *Representation*, затем пункт *132*;

к) 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* → *Numeric*. Эта функция будет генерировать случайное число в пределах от 0 до 1.

 Выберите цикл *For*, расположенный в палитре *Functions* → → *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* → *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*.

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.

Задание 5.2. Создать ВП согласно Вашему варианту

№ варианта	Содержание задания
1	Создайте ВП, который полностью изменяет порядок элементов в массиве, содержащем 10 случайных чисел. Например, элемент массива с индексом 0 становится элементом массива с индексом 9, а элемент массива с индексом 1 становится элементом массива с индексом 8, и так далее. Для изменения порядка данных в массиве следует использовать функцию Reverse ID Array, расположенную в палитре *Functions* → *Array*
2	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}. Используйте функцию Decimate ID Array, расположенную в палитре *Functions* → *Array*
3	Создайте ВП, который генерирует одномерный массив, содержащий 80 случайных чисел, и выдает часть массива, начиная с индекса 15 до индекса 60. На лицевую панель вывести массив случайных чисел и полученный массив
4	Создайте ВП, который генерирует одномерный массив случайных чисел до тех пор, пока не нажата кнопка на лицевой панели. На лицевую панель вывести полученный массив и его размерность
5	Создайте ВП, который генерирует двумерный массив случайных чисел, содержащий 3 строки и 10 столбцов
6	Создайте ВП, который генерирует одномерный массив случайных чисел и сортирует полученный массив в порядке возрастания. На лицевую панель вывести массив случайных чисел и отсортированный массив. Для сортировки элементов в массиве следует использовать функцию Sort 1D Array, расположенную в палитре *Functions* → *Array*
7	Создайте ВП, который генерирует одномерный массив случайных чисел и выводит максимальное значение полученного массива и его порядковый номер. Использовать функцию Array Max & Min, расположенную в палитре *Functions* → *Array*
8	Создайте ВП, который генерирует одномерный массив случайных чисел и выводит минимальное значение полученного массива и его порядковый номер. Использовать функцию Array Max & Min, расположенную в палитре *Functions* → *Array*
9	Создайте ВП, который генерирует двумерный массив случайных чисел, содержащий 4 строки и 5 столбцов
10	Создайте ВП, который генерирует два одномерных массива случайных чисел и объединяет эти массивы в

	двумерный массив чисел. На лицевую панель вывести два исходных массива случайных чисел и двумерный массив, состоящий из элементов исходных массивов
11	Создайте ВП, который генерирует двумерный массив случайных чисел размерностью 5 × 6, и выдает часть этого массива размерностью 4 × 5. На лицевую панель вывести исходный массив случайных чисел и полученный массив
12	Создайте ВП, который генерирует двумерный массив случайных чисел и осуществляет транспонирование полученного массива. На лицевую панель вывести массив случайных чисел и транспонированный массив. Для транспонирования массива используйте функцию Transpose 2D Array, расположенную в палитре *Functions* → → *Array*
13	Создайте ВП, который полностью изменяет порядок элементов в массиве, содержащем 20 случайных чисел. Например, элемент массива с индексом 0 становится элементом массива с индексом 9, а элемент массива с индексом 1 становится элементом массива с индексом 8, и так далее. Для изменения порядка данных в массиве следует использовать функцию Reverse ID Array, расположенную в палитре *Functions* → *Array*
14	Создайте ВП, который генерирует одномерный массив случайных чисел до тех пор, пока минимальный элемент массива не станет равным числу, введенному в элемент управления на лицевой панели
15	Создайте ВП, который создает двумерный массив чисел размерностью 5 × 10 с помощью функции Initialize Array, расположенную в палитре *Functions* → *Array*

Контрольные вопросы

1. Из каких основных компонентов состоит Ваш ВП?
2. Какие типовые приемы создания массива констант Вы знаете?
3. Какие функции работы с массивами Вы знаете?
4. Что такое полиморфные функции?
5. Как создать многомерный массив?
6. Какие функции создания массивов Вы знаете?
7. Как создать двумерный массив в цикле For?
8. Как объединить два одномерных массива в двумерный массив?
9. Как объединить два одномерных массива в более длинный массив той же размерности?
10. Каково назначение элемента *Старт подмножества* в подпрограмме ВП?
11. Какая функция генерирует случайное число в пределах от 0 до 1?
12. Как выбирается количество элементов подмножества в подпрограмме ВП?

Лабораторная работа №6

СОЗДАНИЕ КЛАСТЕРОВ ИЗ ЭЛЕМЕНТОВ УПРАВЛЕНИЯ И ОТОБРАЖЕНИЯ ДАННЫХ. РАБОТА С КЛАСТЕРАМИ. МАСШТАБИРОВАНИЕ КЛАСТЕРА

Цель занятия:

- изучение типовых приемов создания кластеров и функций отображения кластеров;
- создание кластеров на лицевой панели;
- сборка и демонтирование кластеров с использованием функций обработки кластеров;
- создание ВП, использующего полиморфизм в кластерах.

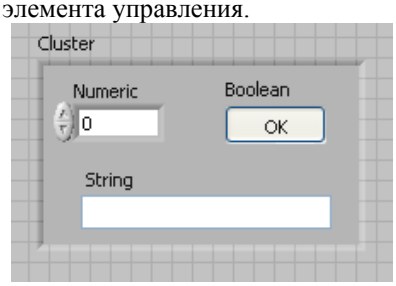
ТЕОРЕТИЧЕСКИЕ УКАЗАНИЯ

Как и массив, кластер является структурой, группирующей данные. Однако в отличие от массива кластер может группировать данные различных типов (логические, числовые и т.д.). 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 вво-

да/вывода данных. Как и массив, кластер может быть элементом управления или отображения данных, однако кластер не может одновременно содержать элементы управления и отображения данных.

Создание кластеров из элементов управления и отображения данных. *Для создания кластеров из элементов управления и отображения данных следует выбрать шаблон кластера в палитре* Controls → → Array & 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.

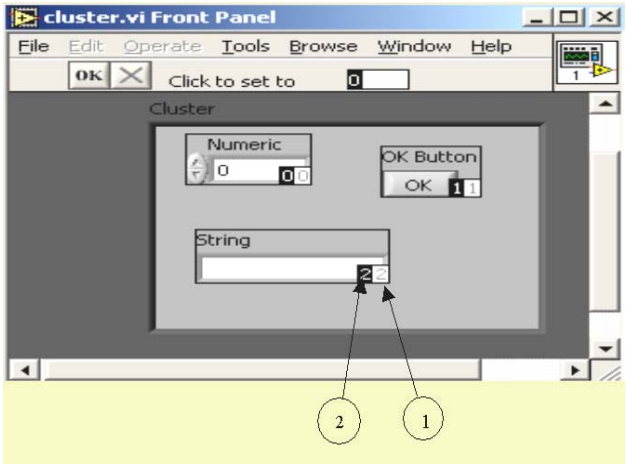


Порядок элементов в кластере. Каждый элемент кластера имеет свой логический порядковый номер, не связанный с положением элемента в шаблоне. Первому помещенному в кластер элементу автоматически присваивается номер 0, второму элементу – 1 и так далее. При удалении элемента порядковые номера автоматически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

В белом поле (1) указан текущий порядковый номер элемента, в черном (2) – 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 *OK* (подтвердить) на панели инструментов.



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

Функции работы с кластерами. Для создания и управления кластерами используются функции, расположенные в палитре *Functions → CIuster*. Функции *Bundle* (Связать) и *Bundle by Name* (Связать по названию) используются для сборки и управления кластерами. Функции *Unbundle* (Разделить) и *Unbundle by Name* (Разделить по названию) 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* (Инструменты кластеров). 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.

Иногда удобно поменять массивы на кластеры и наоборот, поскольку LabVIEW включает в себя намного больше функций, работающих с массивами, чем с кластерами. Для преобразования кластера в массив служит функция *Кластер в массив* (*Cluster to Array*). Обратная операция осуществляется с помощью функции *Массив в кластер* (*Array to Cluster*). Функция *Кластер в массив* конвертирует кластер с количеством элементов *N* одного типа данных в массив с количеством элементов *N* того же типа данных. Индекс массива соответствует порядковому номеру в кластере (т.е. нулевой элемент кластера становится значением массива с индексом 0). Следует обратить внимание, что при использовании этой функции все элементы в кластере должны быть одного типа.

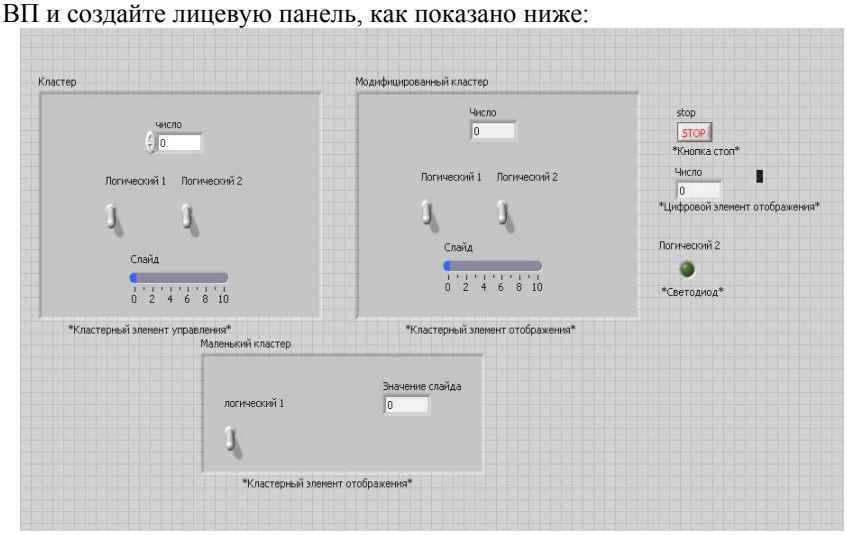
Функция *Массив в кластер* преобразует одномерный массив с числом элементов *N* в кластер с числом элементов *N* того же типа данных. Для включения этой функции необходимо щелкнуть правой кнопкой мыши по терминалу *Массив в Кластер* и выбрать опцию *Размер кластера* (*Cluster Size*) для установления размера выходного кластера, поскольку класте-

ры, в отличие от массивов, не устанавливают свой размер автоматически. Размер кластера по умолчанию равен 9. Если массив имеет меньшее количество элементов, чем это определено размером кластера, LabVIEW автоматически создаст дополнительные элементы кластера со значениями по умолчанию для типа данных кластера. Однако, если количество элементов входного массива больше величины, установленной в окне размера кластера, то проводник блок-диаграммы, идущий к выходному кластеру, будет разорванным, пока не будет отрегулирован его размер.

Задание 6.1. ВП *Работа с кластерами*

Лицевая панель

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



а) поместите на лицевую панель кнопку «Стоп» и круглый светодиод;

б) из палитры *Controls* → *Array & Cluster* выберите шаблон кластера;

в) объекты лицевой панели, показанные на иллюстрации, поместите в шаблон кластера;

г) 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*;

д) повторите пункт «г» для создания элемента *Маленький кластер*; измените его, как показано на рисунке.

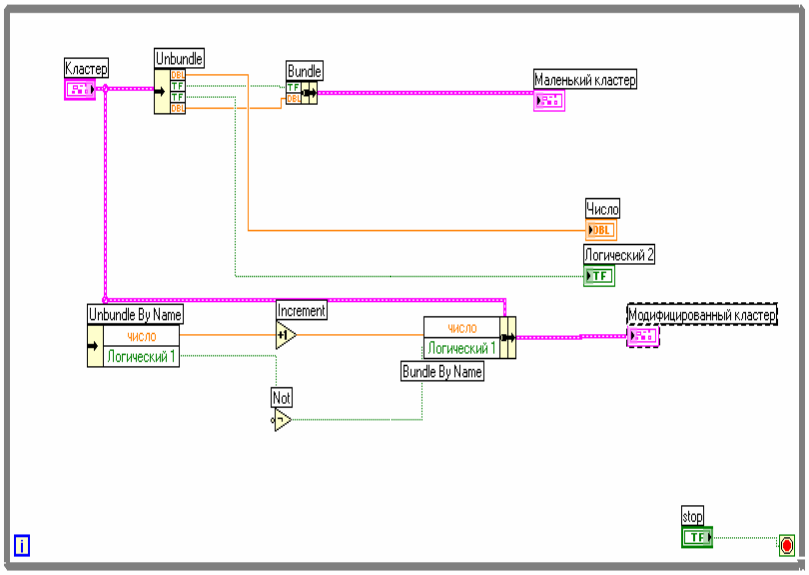
2. Проверьте порядковые номера элементов в кластерах *Кластер* и *Маленький кластер.* Порядковые номера элементов кластера *Модифицированный кластер* и *Кластер* должны совпадать:

а) щелкните правой кнопкой мыши по границе шаблона каждого кластера, из контекстного меню выберите пункт *Reorder Controls in Cluster*;

б) порядковые номера элементов установите, как показано ниже.

Блок-диаграмма

3. Создайте блок-диаграмму, как показано ниже.



 Из палитры *Functions* → *Cluster* 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* → *Cluster* выберите функцию *Bundle*. Эта функция объединит элементы в кластер *Маленький кластер*.

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

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

Из палитры *Functions* → *Cluster* выберите функцию *Bundle by Name*. Эта функция изменит значения элементов *Число* и *Логический1* 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.*
5. Перейдите на лицевую панель и запустите ВП.
6. Поменяйте значения элементов в кластере *Кластер* и запусти-те ВП.
7. Закройте ВП.

Контрольные вопросы

1. Из каких основных компонентов состоит Ваш ВП?
2. Что понимается под термином «Кластер»?
3. Какие типовые приемы создания кластеров Вы знаете?
4. Какие функции отображения кластеров Вы знаете?
5. Как создать кластер на лицевой панели?
6. Как собираются и демонтируются кластеры?
7. Какие функции обработки кластеров Вы знаете?
8. Что такое полиморфизм в кластерах?
9. Как создать модифицированный кластер?
10. Каково основное отличие кластера от массива?
11. Каков порядок размещения элементов в кластере?
12. Как посмотреть и изменить порядковый номер объекта, помещенного в кластер?
13. Как изменить количество полей ввода/вывода в кластере?
14. Как устанавливают размер кластера?
15. Каков размер кластера по умолчанию?
16. Как создать и переименовать копию элемента «Кластер»?

Лабораторная работа № 7
ГРАФИЧЕСКОЕ ОТОБРАЖЕНИЕ ДАННЫХ

Цель занятия:

- приобрести навыки по использованию цикла For и графика Диаграмм для получения и отображения данных;
- создать массив, используя свойство автоиндексации цикла For и вывести данные массива на график Осциллограмм;
- отобразить данные на графике и проанализировать их.

ТЕОРЕТИЧЕСКИЕ УКАЗАНИЯ

График Диаграмм. График Диаграмм (*Waveform Chart*) (волнообразный график) – специальный элемент отображения данных в виде одного и более графиков. График Диаграмм расположен на палитре *Controls* → *Graph*. На рис. 7.1 показан пример Графика Диаграмм с двумя графиками: экспериментальные данные и их бегущее среднее значение.

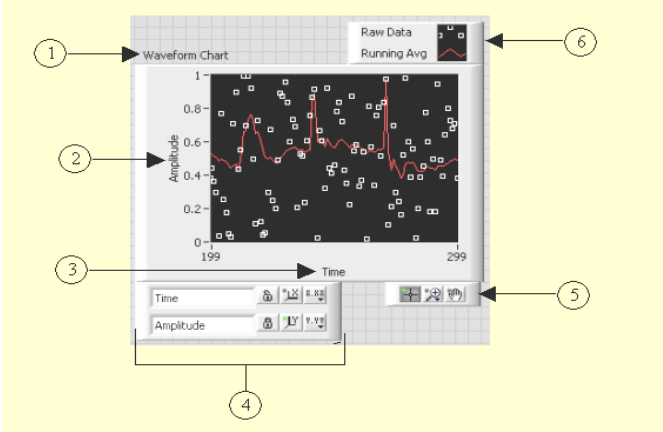
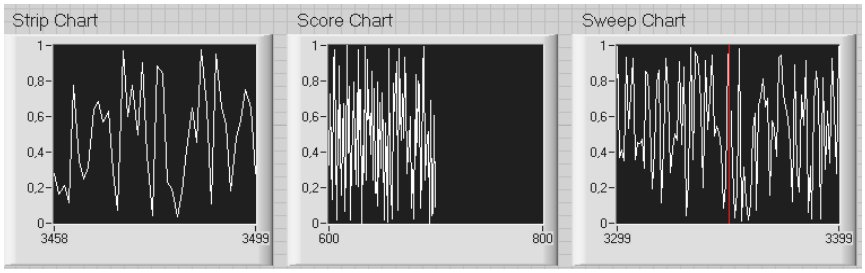


Рис. 7.1. График Диаграмм:
1 – название (Label); *2* – шкала *Y* (*Y* – scal); *3* – шкала *X* (*X* – scale); *4* – панель управления шкалами (Scale legend); *5* – палитра инструментов для работы с графиком (Graph palette); *6* – панель управления графиком (Plot legend)

График Диаграмм использует три различных режима отображения данных: *strip chart* (ленточный), *scope chart* (предельный) и *sweep chart* (амплитудный). 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. 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.



Для создания диаграмм достаточно соединить поле вывода скалярной величины с терминалом данных *графика Диаграмм*.

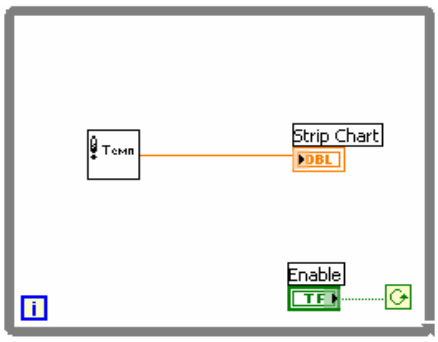


График Диаграмм может отображать несколько графиков. Для объединения отображаемых данных используется функция *Bundle* (объединение), расположенная в палитре *Functions* → *CIuster*. Например, блок-диаграмма, показанная ниже, с

помощью функции *Bundle* объединяет выходные данные трех подпрограмм ВП для последующего отображения на графике Диаграмм.

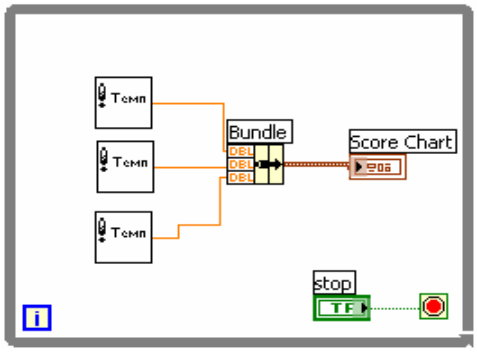


График Осциллограмм и двухкоординатный график Осциллограмм. С помощью графиков в виде осциллограмм ВП обычно отображает накопленные в массив данные. На рис. 7.2 показаны элементы графика.

График Осциллограмм (Waveform Graph) и *двухкоординатный график Осциллограмм* (*XY* Graph) расположены в палитре *Controls* → *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.

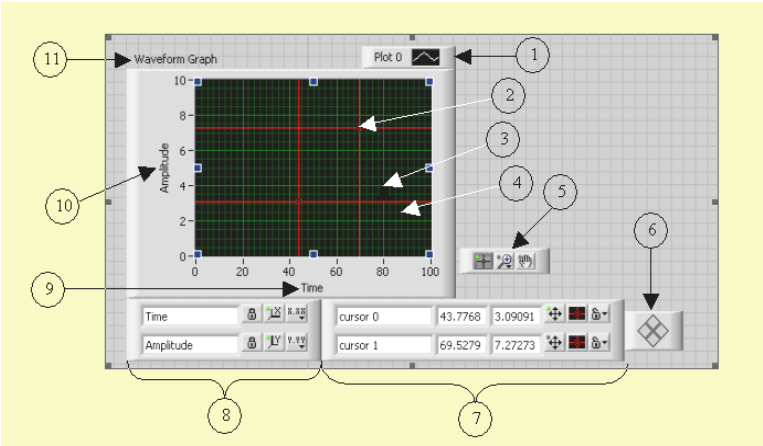


Рис. 7.2. График Осциллограмм:
1 – панель управления свойствами осциллограмм (Plot legend); *2* – курсор (Cursor); *3* – основная размерная сетка (Grid mark); *4* – дополнительная размерная сетка (Mini-grid mark); *5* – палитра элементов управления графиком (Graph palette); *6* – панель перемещения курсора (Cursor mover); *7* – панель управления свойствами курсора (Cursor legend); *8* – панель управления шкалой (Scale legend); *9* – шкала *X* (*X* – scale); *10* – шкала *Y* (*Y* – scale); *11* – собственная метка графика (Label)

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_0, dx и массивом данных по шкале *y*.

Для отображения множества осциллограмм необходимо изменить размер панели *Plot legend* (легенда графика). График множества Осциллограмм используется с целью экономии пространства на лицевой панели и для сравнения осциллограмм данных между собой.

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 $x = 0$.

Графики множества Осциллограмм отображают также и кластеры с установленным начальным значением x_0, dx 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.

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*.

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*.

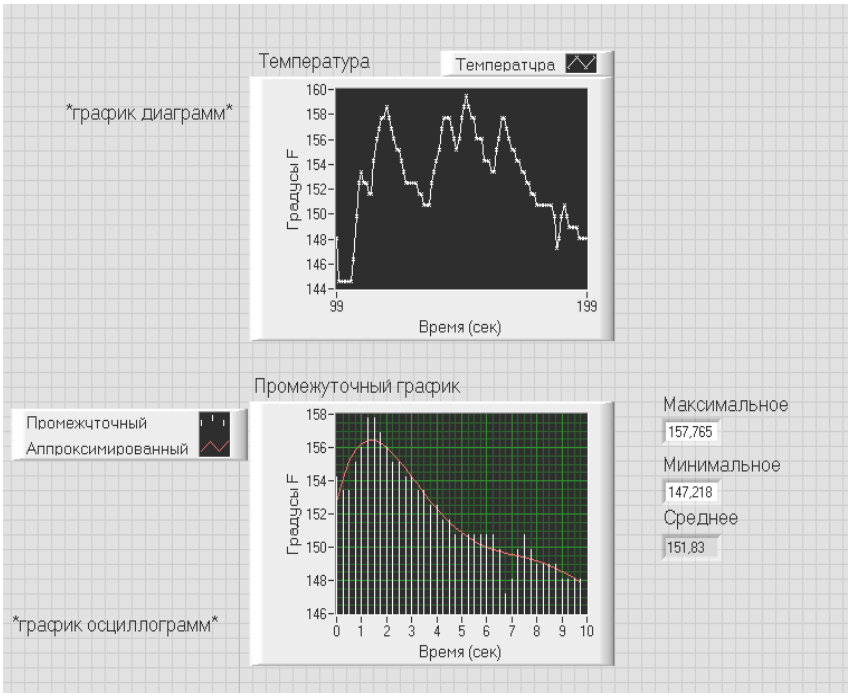
Задание 7.1. ВП *Температурный анализ*

Ниже приведена последовательность действий для создания ВП, который измеряет температуру каждые 0,25 с в течение 10 с. В процессе измерения ВП в реальном масштабе времени отображает данные на графике Диаграмм. После заверше-

ния 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 и создайте лицевую панель, как показано ниже на рисунке.



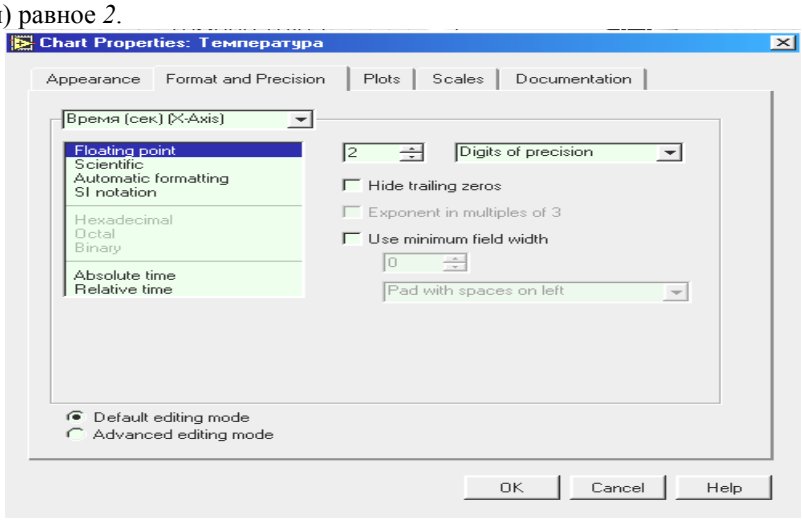
а) Выберите график Диаграмм (*Waveform Chart*) из палитры *Controls* → → *Graph* и поместите его на лицевую панель. На графике Диаграмм будет отображаться значение температуры в реальном масштабе времени. Введите текст *Температура* в поле собственной метки графика.

Обратите внимание на то, что на панели управления графиком (*chart legend*) введен текст *Plot 0*. Измените текст на *Температура* с помощью инструмента ВВОД ТЕКСТА

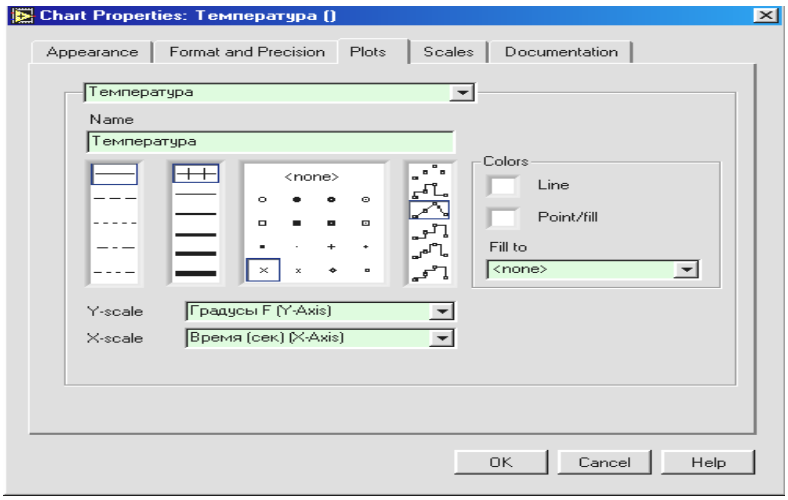
Введите текст *Градусы F* в *поле собственной метки оси ординат графика.*

Введите текст *Время* (с) в поле собственной метки оси абсцисс графика.

б) 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 *2*.



в) Нажмите на закладку *Plots* и установите стиль точек на графике диаграмм в виде *x*.



г) Выберите график Осциллограмм (*Waveform Graph*) из палитры *Controls* → *Graph* и поместите его на лицевую панель. На графике Осциллограмм будут отображаться значения температуры по завершении работы ВП и их аппроксимация. С помощью инструмента ПЕРЕМЕЩЕНИЕ измените размер панели *plot legend*. С помощью инструмента ВВОД ТЕКСТА переименуйте *График 0* в *Промежуточный*, а График 1 – в *Аппроксимированный*.

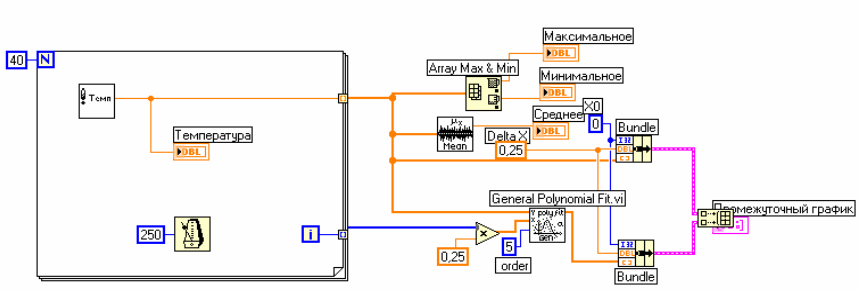
Измените названия осей и собственные метки графика осциллограмм согласно рисунку лицевой панели.

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 *Термометр.vi*, созданный в задании 3.2, и поместите его на блок-диаграмму. Этот ВП выдает одно измеренное значение температуры.


В палитре *Functions* → *Time & Dialog* 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 с.


В палитре *Functions* → *Array* выберите функцию *Array Max & Min*. Эта функция определяет минимум и максимум температуры.

В палитре *Functions* → *Analyze* → *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 *Максимальное, Минимальное и Среднее*.


В палитре *Functions* → *Analyze* → *Mathematics* → *Curve Fitting* (сглаживание) выберите ВП *General Polynomial Fit* (основная полиномиальная аппроксимация). Этот ВП проведет аппроксимацию осциллограммы температуры.


В палитре *Functions* → *Cluster* 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= 0, $Д_x$ и массив значений температуры по *y*. Значение *dx* = 0,25 необходимо для того, чтобы ВП выводил значения температуры на график Осциллограмм каждые 0,25 секунды.


В палитре *Functions* → *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».

Запуск ВП

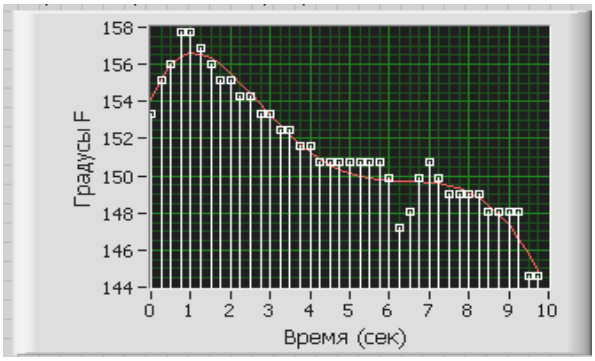
4. Перейдите на лицевую панель и запустите ВП.

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:

а) Щелкните правой кнопкой мыши по надписи *Промежуточный* на панели *Plot legend* и выберите в контекстном меню *Common Plots* → *Scatter Plot* (экспериментальные точки);

б) 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.

Задание 7.2. Создать ВП согласно Вашему варианту

№ варианта	Содержание задания
1	Создайте ВП, который генерирует двумерный массив случайных чисел размерностью 5 × 10 и отображает на графике строки полученного массива
2	Создайте ВП, который генерирует одномерный массив случайных чисел и полученные значения выводит на график. Также отображает аппроксимацию полученных данных
3	Создайте ВП, который генерирует двумерный массив случайных чисел размерностью 11 × 4 и отображает на графике столбцы полученного массива
4	Создайте ВП, который генерирует одномерный массив случайных чисел в диапазоне от 0 до 100 (состоящий из 7 элементов) до тех пор, пока минимальное значение массива не станет равным числу, введенному в элемент управления на лицевой панели. На лицевую панель вывести полученный массив, его график, минимальный элемент и число итераций
5	Создайте ВП, который генерирует два одномерных массива случайных чисел, затем объединяет эти массивы в двумерный массив чисел и отображает на графике строки полученного массива
6	Создайте ВП, который генерирует одномерный массив случайных чисел до тех пор, пока не нажата кнопка на лицевой панели. На лицевую панель вывести полученный массив, его размерность и график
7	Создайте ВП, который на ЛП содержит кластер и массив элементов отображения (Подмножество массива). Кластер состоит из 2 регуляторов (старт подмножества и количество элементов подмножества) и массива числовых данных – Массив 1. Из Массива 1 выделить подмножество, начиная с элемента Старт подмножества, содержащее количество элементов, заданное с регулятора Количество элементов подмножества, и поместить в массив – Подмножество массива. На графике Осциллограмм отображаются элементы Массива 1 и Подмножество массива
8	Создайте ВП, который измеряет температуру каждые 20 с в течение 2 мин и отображает значения температуры в реальном масштабе времени

9	Создайте ВП, который на ЛП содержит кластер, состоящий из 10 числовых элементов управления. На графике отобразите значения, введенные с элементов управления
10	Создайте ВП, который на ЛП содержит кластер, состоящий из 10 числовых элементов управления. На графике отобразите значения, введенные с элементов управления
11	Создайте ВП, который генерирует двумерный массив случайных чисел размерностью 5 × 6 и выдает часть этого массива размерностью 4 × 5. На лицевую панель вывести исходный массив случайных чисел, полученный массив и их графики
12	Создайте ВП, который на ЛП содержит кластер, состоящий из 2-х массивов числовых элементов управления. Элементы этих массивов отобразите на графике
13	Создайте ВП, который генерирует двумерный массив случайных чисел размерностью 3 × 7 и отображает на графике строки полученного массива
14	Создайте ВП, который генерирует одномерный массив случайных чисел в диапазоне от 0 до 100 (состоящий из 7 элементов) до тех пор, пока максимальное значение массива не станет равным числу, введенному в элемент управления на лицевой панели. На лицевую панель вывести полученный массив, его график, максимальный элемент и число итераций
15	Создайте ВП, который генерирует одномерный массив случайных чисел и сортирует полученный массив в порядке возрастания. На лицевую панель вывести массив случайных чисел, отсортированный массив и их графики

Контрольные вопросы

1. Назовите основные элементы графика Диаграмм. Для чего он предназначен?
2. Назовите основные элементы графика Осциллограмм. Для чего он предназначен?
3. Какие режимы отображения данных на графике Диаграмм Вы знаете?
4. С какими данными работает график множества Осциллограмм?
5. Какая функция определяет среднее значение массива?
6. Какая функция определяет минимальное и максимальное значение массива?
7. Какой ВП служит для аппроксимации графика Осциллограмм в Вашем ВП?
8. Как установить определенный стиль точек, цвет и тип линии на графике?

Лабораторная работа № 8
СТРОКИ И ТАБЛИЦЫ

Цель занятия:

- изучить создание строковых элементов управления и отображения данных;
- приобрести навыки по использованию функций обработки строк;
- изучить создание элемента управления и отображения Таблица.

ТЕОРЕТИЧЕСКИЕ УКАЗАНИЯ

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.
- Передачу 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* → → *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* (полоса прокрутки).

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
Режим отображения с обратным слэшем непечатаемых управляющих кодов (‘\’ Codes Display)	Выводит \ для всех непечатаемых управляющих кодов	There\sare\sfour\s display \stypes
Режим скрытого отображения текста (Password Display)	Выводит * для всех кодов текстового пространства	********** **********
Режим отображения 16-тиричных ASCII кодов (Hex Display)	Выводит значение ASCII кода для каждого символа	5468 6572 6520 6172 6520 666F

Функции работы со строками. **Для редактирования и управления строками на блок-диаграмме следует пользоваться функциями обработки строк, расположенными в палитре *Functions* → *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 (Быстрая чернобурка).

- *Concatenate Strings* (связать строки) – 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 (Быстрая чернобурка перепрыгнула через ленивую собаку).

- *String Subset* (подстрока) – 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* (похожая структура) – 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.**

Примечание. **При недостатке места на блок-диаграмме лучше использовать функцию *Format Into String*.**

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.

Функция *Format Into String* преобразует параметры любого формата, такие как числовые данные, в строку. Для увеличения количества параметров следует изменить размер функции.

Для преобразования строки в числовые данные следует использовать функцию *Scan From String*. 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 (внутренний поиск положения строки). 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.

Таблицы. Элемент управления Таблица, расположенный в палитре *Controls* → *List & Table*, предназначен для создания таблиц на лицевой панели. Каждая ячейка находится в строке и столбце таблицы. Поэтому таблица отображает двумерный массив строк. На рис. 8.1 показана таблица и ее составные части.

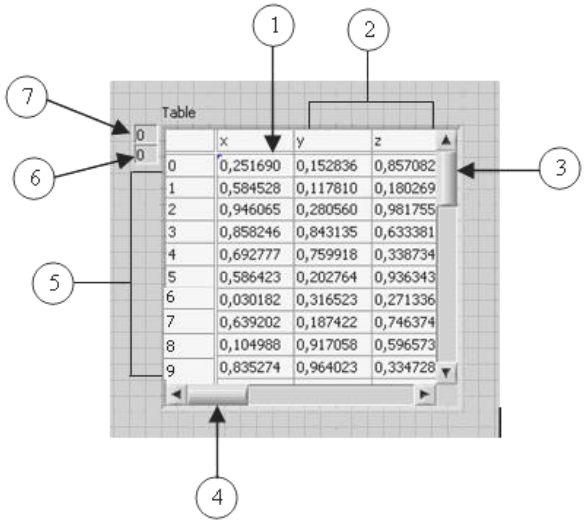


Рис. 8.1. Таблица:
1 – ячейка таблицы; *2* – заголовок столбца; *3* – вертикальная полоса прокрутки; *4* – горизонтальная полоса прокрутки; *5* – заголовок строки; *6* – индекс по горизонтали; *7* – индекс по вертикали

Для инициализации значений ячеек таблицы используется инструмент УПРАВЛЕНИЕ или ВВОД ТЕКСТА, с помощью которых достаточно ввести текст в выделенную ячейку.

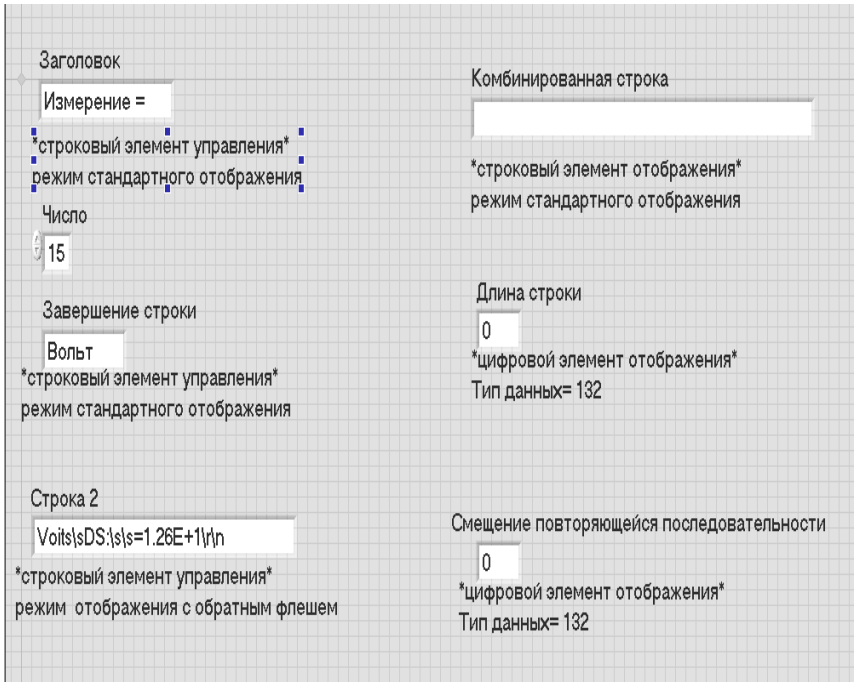
Таблица – это двумерный массив строк. Таким образом, для использования таблицы в качестве элемента отображения данных необходимо двумерный массив чисел преобразовать в двумерный массив строк с помощью функции *Number To Fractional String,* расположенной в палитре *Functions* → *String* → *Number Conversion*. Заголовки строк и столбцов таблицы автоматически не отображаются. Необходимо создать одномерный массив строк, содержащий заголовки строк и столбцов таблицы.

Задание 8.1. ВП *Компоновка строки*

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 выводится на экран индекс первого элемента после указанного символа.

Лицевая панель

1. Откройте новый ВП и оформите лицевую панель, как показано ниже на рисунке. Воспроизводить комментарии и подписи к элементам не обязательно.

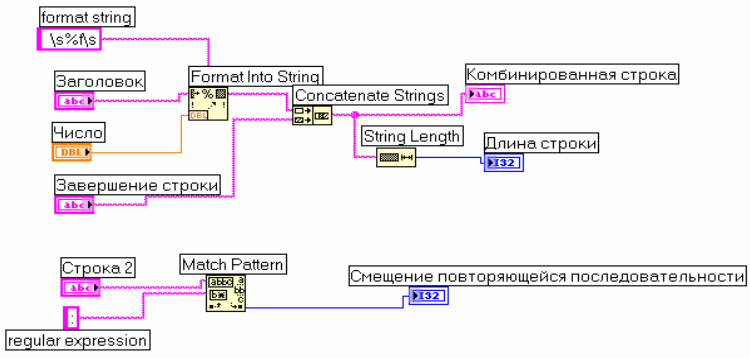


а) Щелкните правой кнопкой мыши по элементу *Строка 2* и выберите из контекстного меню режим отображения '\' Codes Display.

б) Для элементов *Длина строки* и *Смещение повторяющейся последовательности* установите тип представления данных *132*.

Блок-диаграмма

2. Постройте блок-диаграмму, как показано ниже:




Выберите функцию *Format Into String*, расположенную в палитре *Functions* → *String*. Эта функция преобразует число в строку.

а) Щелкните правой кнопкой мыши по функции *Format Into String* и выберите пункт *Edit Format String* для вызова соответствующего диалогового окна.

б) Выделите опцию *Use specified precision* (использовать определенную точность) и в поле ввода текста введите значение 4 для преобразования элемента *Число* в строку с четырьмя знаками после запятой.

в) Нажмите на кнопку OK. LabVIEW создаст формат строки %.4f, используя выбранную опцию.

г) 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.

д) 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* → *String*. Эта функция объединит входящие в нее строки в одну.

 Выберите функцию *String Length*, расположенную в палитре *Functions* → *String*. Эта функция выдаст значение количества символов в объединенной строке *Комбинированная строка*.

 Выберите функцию *Match Pattern*, расположенную в палитре *Functions* → *String*. Эта функция осуществляет поиск в элементе *Строка 2* по шаблону : (двоеточие).

е) Щелкните правой кнопкой мыши по полю *regular expression* (постоянное выражение) и выберите пункт контекстного меню *Create* → *Constant*, введите двоеточие и нажмите на клавиши <»Shift + Enter>.

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.

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 выводит на экран индекс первого элемента после двоеточия в элемент *Смещение повторяющейся последовательности*.

6. Сохраните и закройте ВП.

Задание 8.2. Создать ВП, согласно Вашему варианту

№ варианта	Содержание задания
1	Создайте ВП, который генерирует двумерный массив случайных чисел размерностью 5×40 и записывает полученный массив в таблицу. Таблица должна содержать заголовки для каждого столбца
2	Создайте ВП, который генерирует двумерный массив случайных чисел размерностью 3×10 и записывает полученный массив в таблицу. Таблица должна содержать заголовки для каждого столбца
3	Создайте ВП, который объединяет 3 произвольные строки в одну строку, считает количество символов полученной строки и выдает подстроку определенной длины, указанной с ЛП
4	Создайте ВП, который генерирует двумерный массив случайных чисел размерностью 5×6, транспонирует его и записывает в таблицу
5	Создайте ВП, который объединяет 2 одномерных массива строк в одну отдельную строку
6	Создайте ВП, который объединяет 2 произвольные строки в одну строку и выдает подстроку определенной длины, указанной с ЛП
7	Создайте ВП, который объединяет 2 одномерных массива строк в одну отдельную строку
8	Создайте ВП, который объединяет 4 строки, содержащие числовые данные, в строку
9	Создайте ВП, который считает количество символов строки, введенной на ЛП
10	Создайте ВП, который генерирует 2 одномерных массива случайных чисел, объединяет эти массивы в двумерный массив и помещает его в таблицу

11	Создайте ВП, который объединяет 3 одномерных массива строк в одну отдельную строку
12	Создайте ВП, который объединяет 4 произвольные строки в одну строку и выдает подстроку определенной длины, указанной с ЛП
13	Создайте ВП, который генерирует двумерный массив случайных чисел размерностью 5×8 и записывает полученный массив в таблицу. Таблица должна содержать заголовки для каждого столбца
14	Создайте ВП, который объединяет 2 строки (1 – Автоматизация измерений, 2 – контроля и испытаний) в одну строку и считает количество символов полученной строки
15	Создайте ВП, который генерирует двумерный массив случайных чисел размерностью 6×8, транспонирует его и записывает в таблицу

Контрольные вопросы

1. Как создать строковый элемент управления и отображения данных?
2. Назовите типы отображения строкового объекта.
3. Назовите основные функции работы со строками.
4. С помощью какой функции можно преобразовать числовые данные в строку?
5. Какая функция позволяет подсчитывать количество символов в строке?
6. Какая функция позволяет объединять строки и одномерные массивы строк в отдельную строку?
7. Как создать элемент управления Таблица? Для чего он предназначен?
8. Какая функция позволяет преобразовывать двумерный массив чисел в двумерный массив строк?

Лабораторная работа № 9
ФАЙЛОВЫЙ ВВОД/ВЫВОД

Цель занятия:

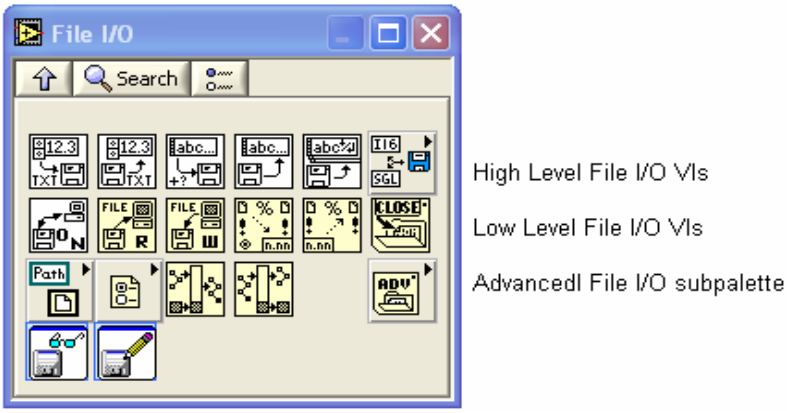
- изучить использование файловых функций ввода/вывода, записать данные в файл, считать данные из файла;
- сохранить данные в файл в форме, доступной для редактора электронных таблиц или текстового редактора;
- изучить использование файлового ввода/вывода высокого уровня.

ТЕОРЕТИЧЕСКИЕ УКАЗАНИЯ

Функции файлового ввода/вывода производят файловые операции записи и считывания данных. Функции файлового ввода/вывода расположены в палитре *Functions* → *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.

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* → *File I/O*. Они предназначены для выполнения основных операций по вводу/выводу данных. 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.

Функции файлового ввода/вывода высокого уровня включают в себя:

- *Write to Spreadsheet File* (запись в крупноформатный файл) – преобразует 2D или ID 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* (чтение из крупномасштабного файла) – 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.
- *Write Characters to File* – записывает строку символов в новый файл или добавляет ее в уже существующий. ВП открывает или создает файл перед записью и после всех операций закрывает его.
- *Read Characters From File* – считывает количество символов number of characters от начального смещения start of read offset. ВП открывает файл перед чтением и после всех операций закрывает его.
- *Read Lines From File* – считывает определенное число строк из текстового или бинарного файла с положения start of read offset. ВП открывает файл перед чтением и закрывает его после.
- *Binary File* – читает и записывает файл в бинарном формате. Данные могут быть целочисленного типа или числовыми данными одинарной точности с плавающей точкой.

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* → *FiIe I/O*. 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.

Функции файлового ввода/вывода низкого уровня включают в себя:

Open/*Create*/*Replace File* (открыть/создать/переместить файл) – 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 Time & Dialog*. 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.

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. Путь к файлу состоит из имени дисковода, двоеточия, обратного слэша, разделяющего директории, и имени файла. Например, H:\Laboratoria\lab1.vi в папке Laboratoria.

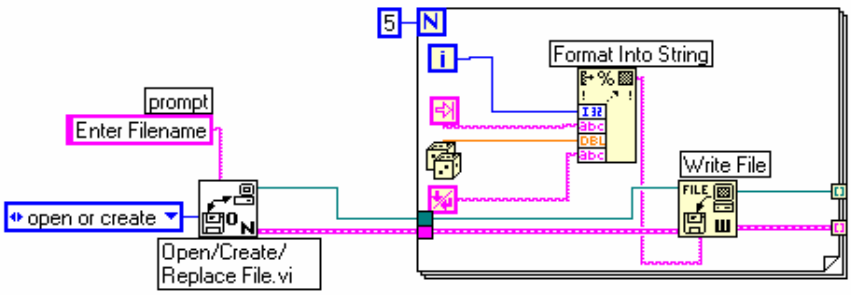
Форматирование строк таблицы символов.

Для того чтобы записать данные в файл формата электронной таблицы, необходимо переформатировать строковые данные в строку таблицы, содержащую разделители, такие как символ табуляции. Символ табуляции *Tab constant* разделяет столбцы, а символ *end of line* разделяет строки. Оба символа расположены в палитре *Functions* → *String*.

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* (записать в файл) или *Write File* (записать файл).

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*.

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.



Можно открыть данный текстовый файл в любом редакторе электронных таблиц для отображения на экране следующей таблицы:

	A	B
1	0	0,874272
2	1	0,882184
3	2	0,62185
4	3	0,211246
5	4	0,459975

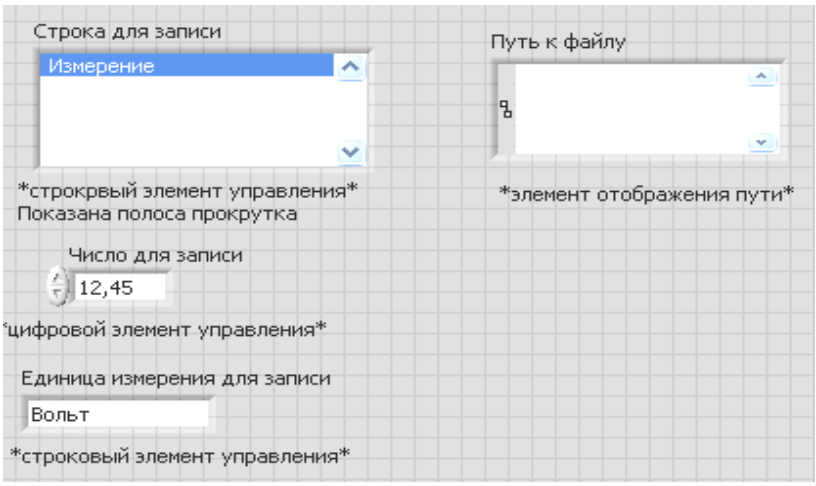
Дополнительные функции работы с файлами (Advanced File I/O) расположены в палитре *Functions* → *File I/O* → *Advanced File Functions* и предназначены для управления отдельными операциями над файлами.

Задание 9.1. ВП *Запись файла*

Ниже приведена последовательность действий для создания ВП, который объединяет строку, числовые данные и модуль строки в файл.

Лицевая панель

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

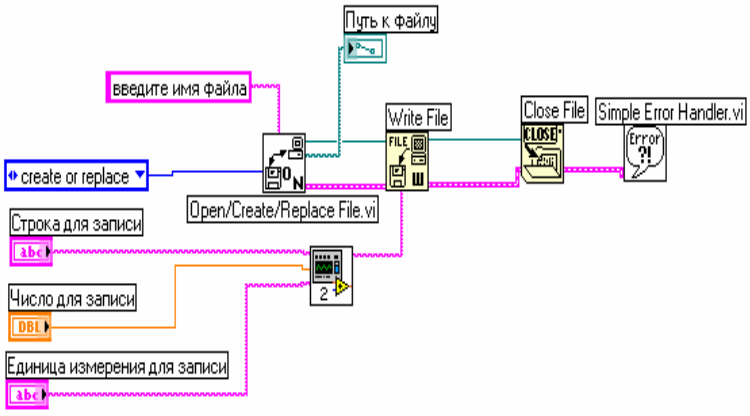


а) В палитре *Controls* → *String & Path* выберите элемент отображения пути. Этот элемент отобразит путь к созданному файлу данных;

б) Щелкните правой кнопкой мыши по элементу *Строка для записи* и в контекстном меню выберите пункт *Visible Items* → *Scrollbar* (полоса прокрутки).

Блок-диаграмма

2. Постройте следующую блок-диаграмму:



 Выберите в разделе *Functions* → *Select a VI* ВП *Компоновка строки.vi*, созданный в задании 8.1, 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* → *File I/O*. Этот ВП выводит на экран диалоговое окно для создания файла.

а) Щелкните правой кнопкой мыши по полю *prompt* (подсказка) 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*;

б) 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* → *File I/O*. Эта функция записывает объединенную строку в файл.

 Выберите функцию *Close File*, расположенную в палитре *Functions* → *File I/O*. Эта функция закрывает файл.

 Выберите подпрограмму ВП *Simple Error Handler VI*, расположенную в палитре *Functions* → *Time & Dialog*. 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 ОК.

ВП запишет в файл данные из элементов *Строка для записи*, *Число для записи* и *Единица измерения для записи*.

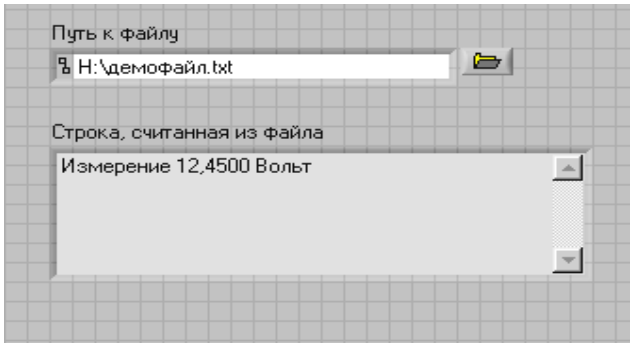
6. Закройте ВП.

Задание 9.2. ВП *Чтение файла*

Ниже приведена последовательность действий для создания ВП, который читает файл, созданный в задании 9.1, и выводит данные в строковом элементе отображения.

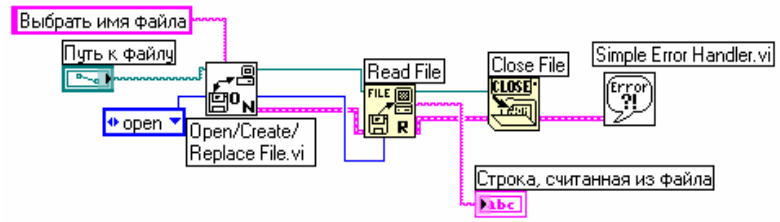
Лицевая панель

7. Откройте новый ВП и создайте лицевую панель, используя элемент управления путем к файлу и строковый элемент отображения в палитре *Controls* → *String & Path*.



Блок-диаграмма

8. Постройте следующую блок-диаграмму:



 В палитре *Functions* → *File I/O* выберите подпрограмму *Open/Create/Replace File VI*. Этот ВП выведет на экран диалоговое окно, которое используется для создания и открытия файла.

а) Щелкните правой кнопкой мыши по входному полю *prompt* и из контекстного меню выберите *Create* → *Constant* для создания константы *Выбрать имя файла*;

б) 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* → *File I/O* выберите функцию *Read File*. Эта функция читает количество байт данных с начала файла, определяемое значением поля *count*.

 В палитре *Functions* → *File I/O* выберите функцию *Close File*. Эта функция закроет файл.

 В палитре *Functions* → *Time& Dialog* 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 *ОК*.

12. Запустите ВП. Строка, считанная из файла, отобразится на лицевой панели ВП.

Задание 9.3. Создать ВП, согласно Вашему варианту

№ варианта	Содержание задания
1	Создайте ВП, который генерирует двумерный массив случайных чисел размерностью 5 × 40 и записывает транспонируемый массив в файл текстового формата
2	Создайте ВП, который записывает 3 произвольные строки в файл текстового формата. На ЛП должен отображаться путь к файлу
3	Создайте ВП, который генерирует двумерный массив случайных чисел размерностью 5 × 6, транспонирует его и записывает в файл формата электронной таблицы
4	Создайте ВП, который генерирует одномерный массив случайных чисел и записывает транспонируемый массив в файл любого формата
5	Создайте ВП, который записывает массив чисел, содержащийся на ЛП, в файл. На ЛП должен отображаться путь к файлу
6	Создайте ВП, который генерирует двумерный массив случайных чисел размерностью 3 × 10 и записывает полученный массив в таблицу на ЛП и в файл
7	Создайте ВП, который объединяет 2 строки (1 – Автоматизация измерений, 2 – контроля и испытаний) в одну строку и записывает эту строку в файл. На ЛП должен отображаться путь к файлу
8	Создайте ВП, который генерирует два одномерных массива случайных чисел, объединяет эти массивы в двумерный массив чисел и помещает транспонированный массив в файл
9	Создайте ВП, который записывает массив чисел в файл формата электронной таблицы. Файл должен содержать заголовки для каждого столбца
10	Создайте ВП, который объединяет два одномерных массива в одну строку и записывает ее в файл. На ЛП отображается путь к файлу
11	Создайте ВП, который генерирует массив случайных чисел и записывает его в файл формата электронной таблицы. Файл должен содержать заголовки для каждого столбца
12	Создайте ВП, который генерирует массив случайных чисел и записывает его в файл текстового формата. Файл должен содержать заголовки каждого столбца
13	Создайте ВП, который записывает 4 произвольные строки в файл текстового формата. На ЛП должен отображаться путь к файлу
14	Создайте ВП, который объединяет два одномерных массива чисел в одну строку и записывает ее в файл. На ЛП отображается путь к файлу
15	Создайте ВП, который генерирует массив случайных чисел и записывает его в файл текстового формата. Файл должен содержать заголовки каждого столбца

Контрольные вопросы

1. Каково назначение функций файлового ввода/вывода? Где они расположены?
2. Назовите назначение функций файлового ввода/вывода высокого уровня.
3. Назовите назначение функций файлового ввода/вывода низкого уровня.
4. Для чего предназначены дополнительные функции работы с файлами?
5. Для чего предназначена функция Format Into File?
6. Какие существуют способы доступа к файлу при сохранении данных?
7. Каким образом в LabVIEW осуществляется обработка ошибок при работе с функциями файлового ввода/вывода?
8. Что представляет собой палитра функций файлового ввода/вывода?
9. Как записать массив чисел в файл текстового формата, содержащий заголовки для каждого столбца?

Лабораторная работа №10

Отладка VI

Цель упражнения

Научиться пользоваться встроенными в LabVIEW отладочными средствами.

Порядок выполнения

Выполняя последующие действия, загрузите в LabVIEW неработающий VI и исправьте в нем ошибки. Для отслеживания выполнения VI воспользуйтесь пошаговым режимом и подсветкой выполнения.

1. Откройте Debug Exercise (Main) VI и ознакомьтесь с ним.
   - ❑ Выберите команду меню **File»Open**.
   - ❑ Откройте файл `Debug Exercise (Main).vi` в папке `<Exercises>\ LabVIEW Core 1\Debugging`.

   На экране появится лицевая панель, приведенная на рисунке 3-1.

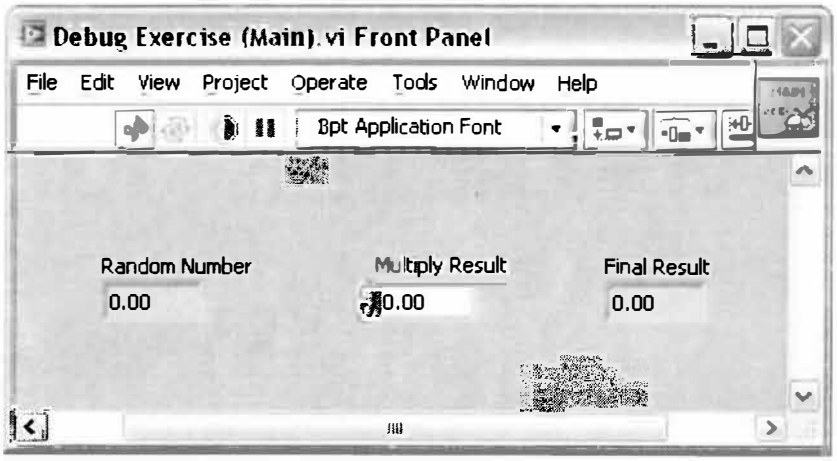


Рисунок 3-1. Лицевая панель Debug Exercise (Main) VI

   - ❑ Обратите внимание, что кнопка **Run** на панели инструментов разорвана, что указывает на то, что VI неисправен, и не может быть выполнен.

2. Откройте и рассмотрите блок-диаграмму Debug Exercise (Main) VI.
   - ❑ Выберите команду меню **Window»Show Block Diagram**, чтобы на экране появилась блок-диаграмма, приведенная на рисунке 3-2.

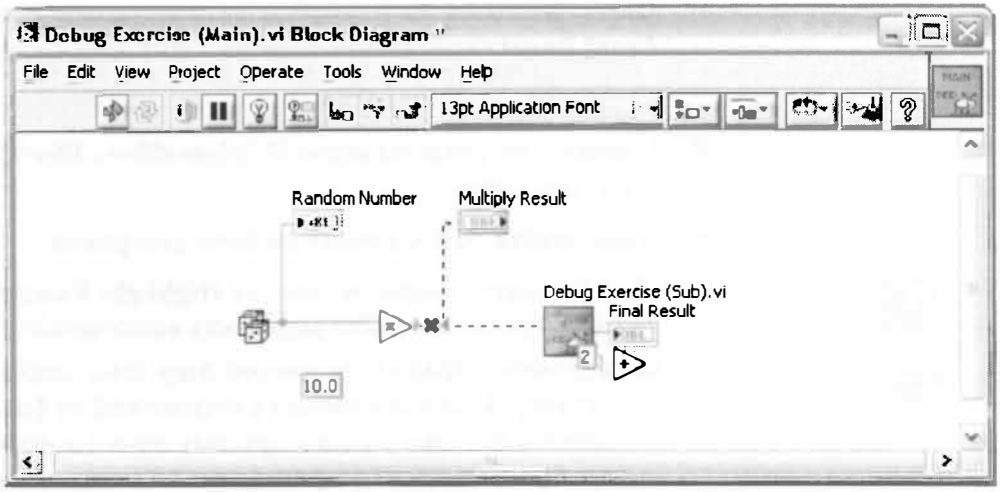


Рисунок 3-2. Блок-диаграмма Debug Exercise (Main) VI

Пояснения к элементам блок-диаграммы на рисунке 3-2:

- Функ ция Random Number (0-1) — генерирует случайное число в диапазоне от 0 до 1.

- Функция Multiply умножает случайное число на `10.0`.

- Числовая константа, на которую умножается случайное число.

- Debug Exercise (Sub) VI находится в папке `<Exercises>\LabVIEW Core 1\Debugging\Supporting Files`. Этот VI прибавляет к результату умножения 100 и извлекает из полученного значения квадратный корень.

3. Приведите в порядок ту часть блок-диаграммы, где элементы расположены как попало, чтобы улучшить воспринимаемость блок-диаграммы.
   - ❑ Щелкните мышью и протяните курсор так, чтобы выделить Debug Exercise (Sub) VI и находящиеся справа от него функцию, константу и индикатор.

   - ❑ Щелкните по кнопке **Clean Up Diagram** на линейке инструментов.
4. Найдите и исправьте каждую ошибку.
   - ❑ Щелкните по кнопке **Run**, чтобы вывести на экран окно **Error list**, в котором приведен полный список ошибок.
   - ❑ Выделите описание ошибки в окне **Error list**. В разделе **Details** приведено описание ошибки, а в некоторых случаях даются рекомендации по ее исправлению.
   - ❑ Щелкните по кнопке **Help**, чтобы вывести на экран раздел справки *LabVIEW Help*, в котором приводится детальное описание ошибки, включая пошаговые инструкции по ее исправлению.
   - ❑ Одиночным щелчком по кнопке **Show Error** или двойным щелчком по описанию ошибки подсветите область блок-диаграммы, в которой находится ошибка.
   - ❑ Пользуясь окном **Error list**, последовательно исправьте все ошибки.
5. Сохраните VI командой меню **File»Save**.

6. Щелчком мыши по лицевой панели или командой меню **Window»Show Front Panel** переместите лицевую панель на передний план.
7. Щелкните по кнопке **Run**.
8. С помощью команды меню **Window»Show Block Diagram** перейдите на блок-диаграмму.
9. Анимируйте поток данных на блок-диаграмме.

- ❑ Щелкните мышью по кнопке **Highlight Execution** в линейке инструментов, чтобы разрешить выполнение с подсветкой.

- ❑ Щелкните мышью по кнопке **Step Into**, чтобы начать пошаговую отладку. При выполнении с подсветкой на блок-диаграмме поток данных от одного узла к другому показывается кружками, продвигающихся по проводникам. Узлы, готовые к выполнению, мерцают.

- ❑ Щелкайте мышью по кнопке **Step Over** после выполнения каждого узла, чтобы пройти по всей блок-диаграмме. После каждого щелчка по кнопке **Step Over** выполняется текущий узел и выполнение останавливается на следующем узле.
- ❑ Данные появляются на лицевой панели по мере вашего продвижения по блок-диаграмме. VI генерирует случайное число и умножает его на `10.0`. Далее к полученному результату SubVI прибавляет `100.0` и затем из полученного значения извлекает квадратный корень.

- ❑ Когда появляется мерцающая рамка вокруг всей блок-диаграммы, щелкните мышью по кнопке **Step Out**, чтобы прекратить пошаговую отладку Debug Exercise (Main) VI.

10. Выполните пошаговую отладку VI с заходом внутрь subVI.
    - ❑ Щелкните мышью по кнопке **Step Into**, чтобы начать пошаговую отладку.

- ❑ Когда Debug Exercise (Sub) VI начинает мерцать, щелкните мышью по кнопке **Step Into**. Кнопка Run примет вид, указанный слева.

- ❑ Щелчком мыши по Debug Exercise (Sub) VI выведите на экран его блок-диаграмму. На иконке subVI на блок-диаграмме Debug Exercise (Main) VI появится зеленая стрелка, которая указывает на то, что subVI выполняется.
- ❑ Щелчком мыши по блок-диаграмме Debug Exercise (Sub) VI переместите блок-диаграмму на передний план.
- ❑ Дважды щелкните по кнопке **Step Out**, чтобы завершить пошаговую отладку subVI. Блок-диаграмма Debug Exercise (Main) VI становится активной.
- ❑ Щелкните мышью по кнопке **Step Out**, чтобы завершить пошаговую отладку.

11. В процессе работы VI с помощью пробника проверьте промежуточные значения в каком-нибудь проводнике.

- ❑ В палитре Tools выберите инструмент **Probe**.
- ❑ Щелкните инструментом Probe по любому проводнику – на экране появится окно **Probe Watch Window**.

В окне Probe Watch Window отображаются все пробники всех VI, которые в настоящий момент находятся в памяти. В этом окне пробники отсортированы в порядке их создания, причем они группируются ниже тех VI, которым они принадлежат.

- ❑ Выполните VI в пошаговом режиме. В окне Probe Watch Window отображаются данные, передаваемые через проводник.

12. Разместите на блок-диаграмме контрольные точки, чтобы на них приостанавливалось выполнение VI.

- ❑ Поместите контрольную точку на блок-диаграмме, щелкая инструментом Breakpoint по узлам или проводникам, чтобы выполнение VI приостанавливалось после того, как все узлы на блок-диаграмме выполнятся.
- ❑ Щелкните по кнопке **Run** для запуска VI. Когда будет достигнута контрольная точка, выполнение VI приостанавливается, а кнопка **Pause** на панели инструментов становится красного цвета.

- ❑ Щелкните по кнопке **Continue**, чтобы продолжить выполнение VI до следующей контрольной точки или до выхода из программы.
- ❑ Для удаления контрольных точек щелкните по ним инструментом Breakpoint.

13. Щелкните по кнопке **Highlight Execution**, чтобы запретить подсветку выполнения.
14. Закройте VI и другие открытые окна командой меню **File»Close**.

Лабораторная работа №11

Измерение среднего значения

Цель работы: ознакомление с функциями статистической обработки данных в темпе процесса измерения при помощи палитр функций LabView. В связи с этим необходимо решить следующие задачи:

1. Выполнить построение кода формирующего массивы случайных данных с заданным законом распределения согласно заданию.

2. Используя функции статистической обработки данных выполнить поиск требуемых численных оценок среднего значения, и стандартного отклонения.

Теоретическая часть

Функции генерации сигналов и шумов

Функции генерации и обработки сигналов, размещенные в подпалитре Обработка сигналов, позволяют формировать как отдельные значения, так и массивы отсчетов сигналов и шумов и производить их обработку с помощью различных операции в частотной и временной областях.

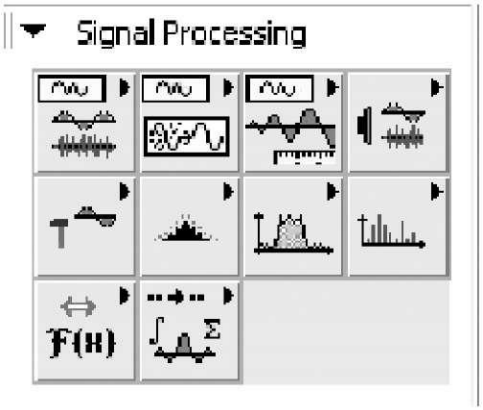


Рисунок 4.1. Вид категории функций
Обработка (Signal Processing)

ВП из палитры генерации сигналов и шумов (рис. 4.2) используются для формирования детерминированных и случайных сигналов с заданным набором параметров. Первые два ВП в верхнем ряду представляют многофункциональные генераторы сигналов с широким набором регулируемых параметров. ВП, размещенные во второй и третьей строках, предназначены для генерации наиболее широко применяемых детерминированных периодических сигналов, а ВП, находящиеся в четвертой и пятой строках, служат для расчета шумов с различными законами амплитудного и спектрального распределения.

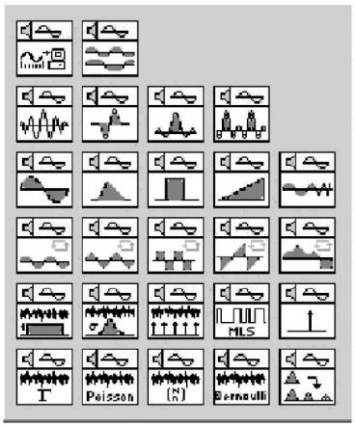


Рисунок 4.2. Палитра функций генерации сигналов и шумов

Описание функций генерации сигналов и шумов приведено далее.

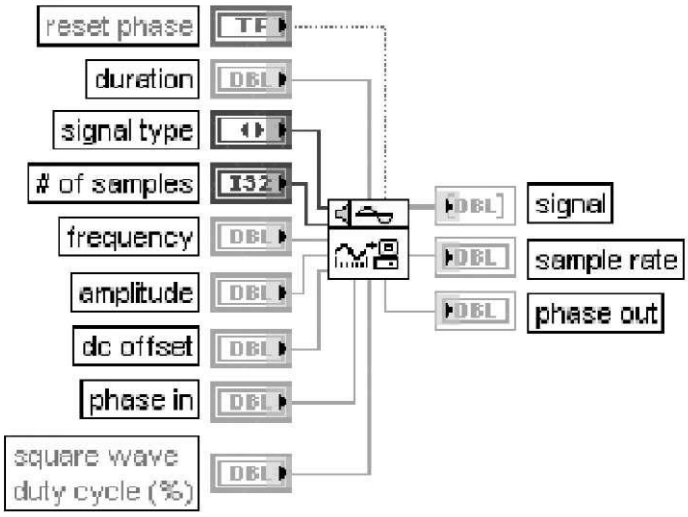


Рисунок 4.3. Signal Generator by Duration – 53

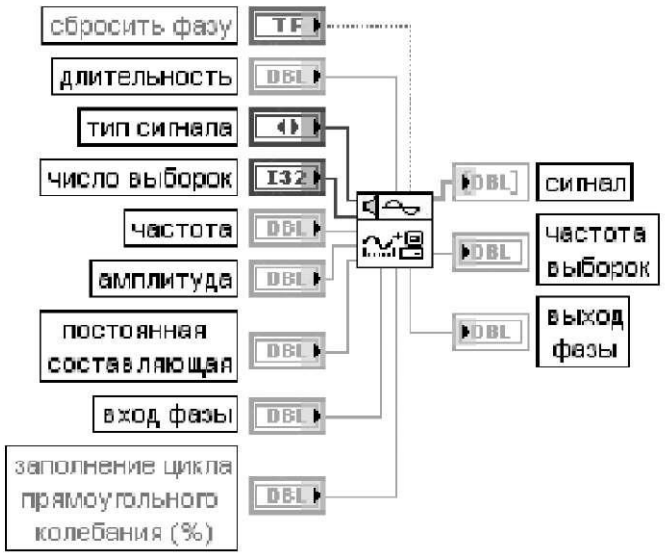


Рисунок 4.4. Генератор сигналов с заданной длительностью

ВП генерирует сигнал (signal), имеющий форму, задаваемую на входе тип сигнала тип сигнала (signal type).

Вход *сбросить фазу* (reset phase) определяет начальную фазу выходного сигнала. При установке на входе состояния ИСТИНА (по умолчанию) начальная фаза сигнала определяется входом *вход фазы* (phase in). В противном случае начальная фаза устанавливается равной значению фазы на выходе *выход фазы* (phase out) при последнем выполнении этого ВП.

Вход *длительность* (duration) задает время в секундах, равное длительности генерируемого выходного сигнала. По умолчанию значение длительности равно 1,0.

Вход *тип сигнала* (signal type) задает следующие типы генерируемого сигнала:

Синусоидальный (sine) (по умолчанию);

Косинусоидальный (cosine);

Треугольный (triangle)

Прямоугольный (square);

Пилообразный (sawtooth);

Линейно нарастающий (increasing ramp);

Линейно спадающий (decreasing ramp).

Вход *число выборок* (# of samples) задает число выборок выходного сигнала. По умолчанию это значение равно 100.

Вход *частота* (frequency) определяет частоту выходного сигнала в герцах. По умолчанию значение частоты равно 10. При задании частоты необходимо учитывать требование выполнения критерия Найквиста: частота < число выборок/(2*длительность).

Вход *амплитуда* (amplitude) задает амплитуду выходного сигнала. По умолчанию значение амплитуды равно 1,0.

Вход *постоянное смещение* (dc offset) задает постоянное смещение или значение постоянной составляющей выходного сигнала. По умолчанию значение постоянной составляющей равно 0.

Вход *Вход фазы* (phase in) определяет начальную фазу (в градусах) выходного сигнала при установке входа *сбросить фазу* (reset phase) в состояние ИСТИНА. По умолчанию значение на *входе фазы* равно 0.

Вход *заполнение цикла прямоугольного колебания* (square wave duty cycle) определяет время (в процентах от периода), в течение которого прямоугольный сигнал имеет высокий уровень. ВП использует данный параметр только для прямоугольного сигнала. По умолчанию значение на входе равно 50%.

Выход *сигнал* (signal) представляет сгенерированный массив выборок сигнала.

Выход *частота выборок* (sample rate) отображает частоту дискретизации выходного сигнала. Частота выборок равна отношению числа выборок к длительности длительности.

Выход фазы (phase out) указывает значение фазы (в градусах) последней выборки выходного сигнала

Tones and Nonise - Гармонические колебания и шум

ВП генерирует массив, содержащий сумму гармонических колебаний, шум и постоянную составляющую.

Вход сбросить сигнал сбросить сигнал сбросить сигнал сбросить сигнал сбросить сигнал (reset signal) при подаче значения ИСТИНА устанавливает фазу каждого гармонического колебания равной значению фазы (phase) из массива гармонические колебания (tones), значение начального числа при генерации шума равным значению входа начальное значение (seed) и отметку времени равной нулю. По умолчанию состояние входа - ЛОЖЬ.

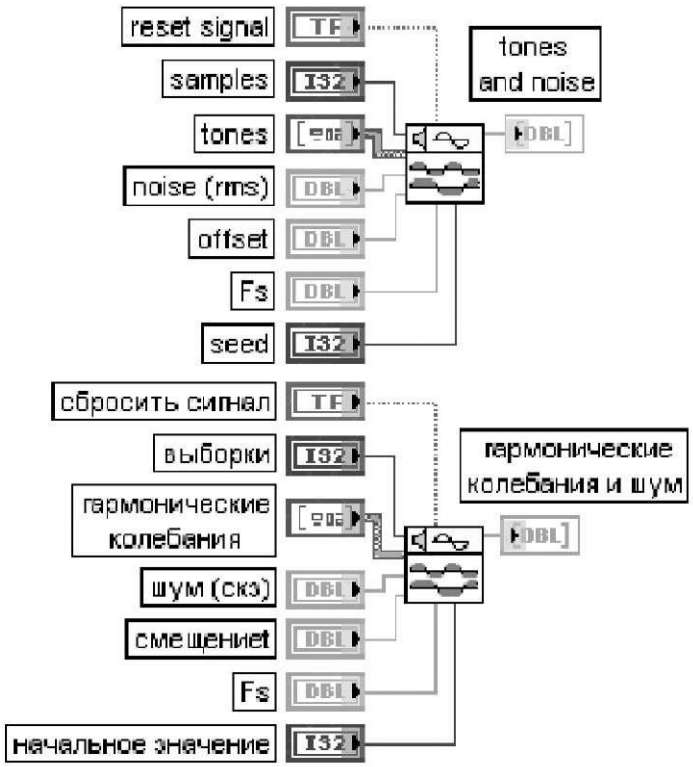


Рисунок 4.5. Tones and Nonise - Гармонические колебания и шум

Вход выборки (samples) определяет число выборок выходного массива гармонические колебания и шум (tones and noise). По умолчанию это число равно 1000.

Вход гармонические колебания гармонические колебания гармонические колебания гармонические колебания гармонические колебания (tones), являющийся кластером, содержит параметры каждого гармонического колебания:

- частота (frequency) определяет частоту синусоидального колебания в герцах;
- амплитуда (amplitude) определяет амплитуду синусоидального колебания;

• фаза (phase) определяет начальную фазу синусоидального колебания. По умолчанию фаза равна 0

Вход шум (noise) определяет среднеквадратичное значение (с.к.з.) аддитивного гауссовского шума. По умолчанию значение шума равно 0,0.

Вход смещение (offset) задает уровень постоянной составляющей сигнала. По умолчанию уровень равен 0,0.

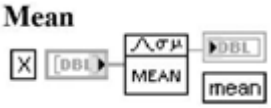
Вход Fs определяет частоту выборок в единицах число выборок/с. По умолчанию частота равна 1000.

Установка на входе начальное значение начальное значение начальное значение начальное значение начальное значение начальное значение (seed) значения >0 вызывает инициализацию генератора аддитивного шума. По умолчанию значение входа равно -1. Если начальное значение начальное значение начальное значение начальное значение начальное значение начальное значение меньше или равно 0, то генератор шума не инициализируется и продолжает генерацию шума на основе предыдущих значений.

Выход гармонические колебания и шумы гармонические колебания и шумы гармонические колебания и шумы гармонические колебания и шумы гармонические колебания и шумы (tones and noise) отображает сгенерированный выходной массив.

Функции статистической обработки данных

Функции основной палитры статистической обработки данных (рисунок 4.6а) позволяют рассчитать основные статистические параметры наборов данных: среднее, среднеквадратическое отклонение, дисперсию, среднеквадратичное значение, центральные моменты различного порядка, медиану и моду, а также рассчитать значения гистограммы набора данных. Дополнительные подпалитры (рисунок 4.6б-4.6г) содержат функции расчета вероятностей (интегральных распределений) для различных законов



распределения и функции расчета квантилей распределений по заданным значениям вероятности, проверки гипотез и дисперсионного анализа.

Ниже приведено описание функций основной палитры статистической обработки данных.

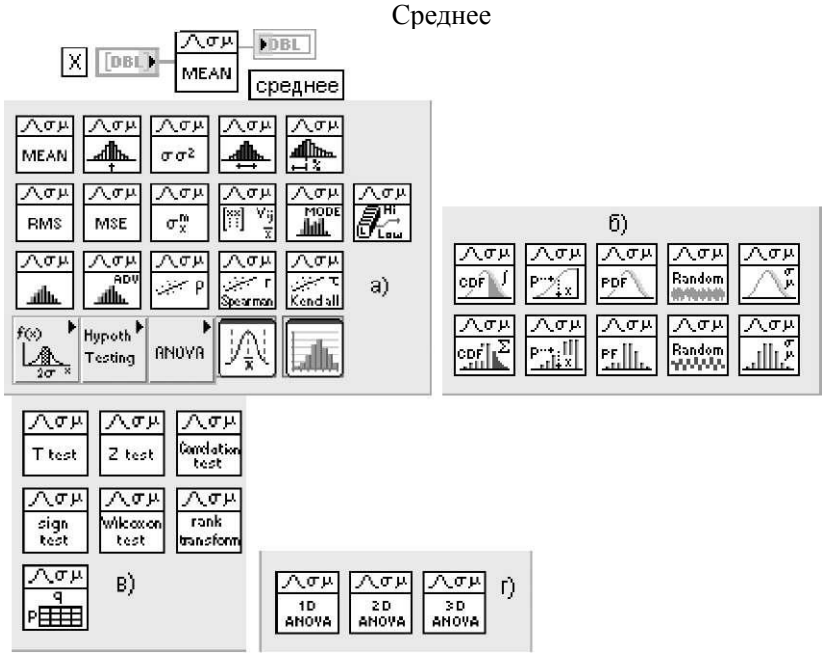


Рисунок 4.6. Вид основной палитры (а) и дополнительных подпалитр (б) - (г) функций статистической обработки данных

Задание

Необходимо создать виртуальный прибор управляющий конфигурированием, запуском и отображением генерируемых и вычисляемых величин.

Прибор должен генерировать произвольное множество значений по заданному закону распределения, и вычислять среднее значение из множества и стандартное отклонение, а также отображать в графической форме этих данные на графике в масштабе.

Так же необходимо выбрать из таблицы 4.1 согласно варианту исходное значение для генерации произвольного множества и закон распределения, задав их значения на фронтальной панели и на блок-диаграмме реализованного виртуального прибора определить среднее значение и стандартное отклонение для десяти опытов и занести значения в таблицу.

Выполнение работы

На рисунке 4.1 отображена лицевая панель виртуального прибора, несущая в себе элементы управления.

Таблица 4.1 Варианты значений коэффициентов и типа распределения для генерации исходных последовательностей данных

№ варианта		исходные значе		Тип распределения
		ния для генерации		
1-я цифра	2-я цифра	среднее	разброс	
варианта	варианта			
0	0	1	1	Нормальное (Normal)
1	1	2	2	Лапласа (Laplas)
2	2	3	3	Коши (Cauchy)
3	3	4	4	Экспоненциальное (Exp)
4	4	5	5	F-распределение (F)
5	5	6	6	Гамма (Gamma)
6	6	7	7	Парето-распределение (Pareto)
7	7	8	8	Треугольное (Triangular)
8	8	9	9	Формальное (Uniform)
9	9	10	10	Вейбула распределение (Weibull)

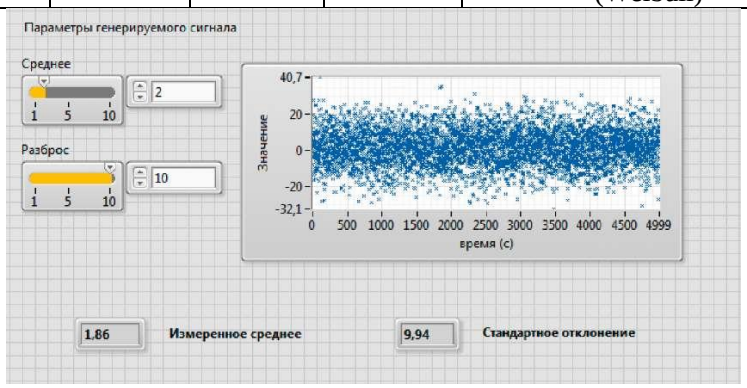


Рисунок 4.1. Фронтальная панель виртуального прибора

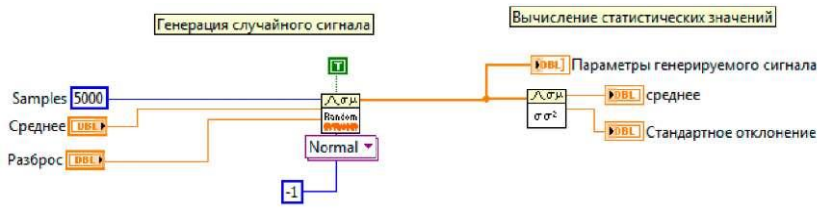


Рисунок 4.2. Блок-диаграмма виртуального прибора

Результат выполнения работы

В результате выполнения работы необходимо создать виртуальный прибор по выше предлагаемому заданию и написать отчет о лабораторной работе.

Лабораторная работа №12

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 рис. 4.1.

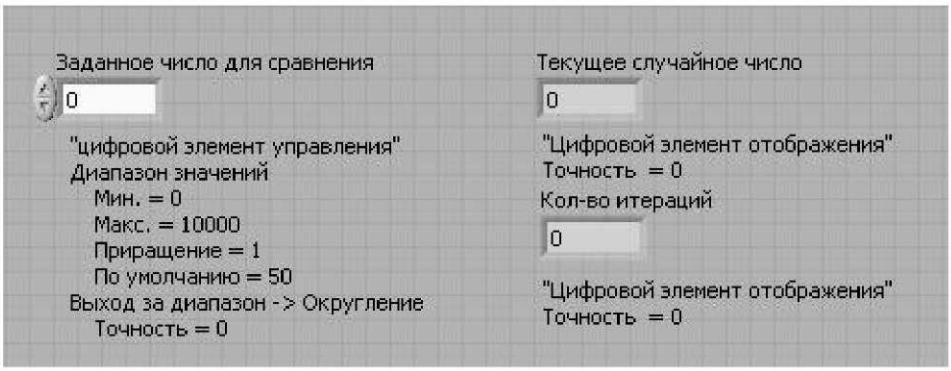


Рис. 4.1 - Лицевая панель ВП Подсчёт итераций

- **a.** Поместите на лицевую панель числовой элемент управления, находящийся на палитре **Controls»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»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 Entry.** 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.

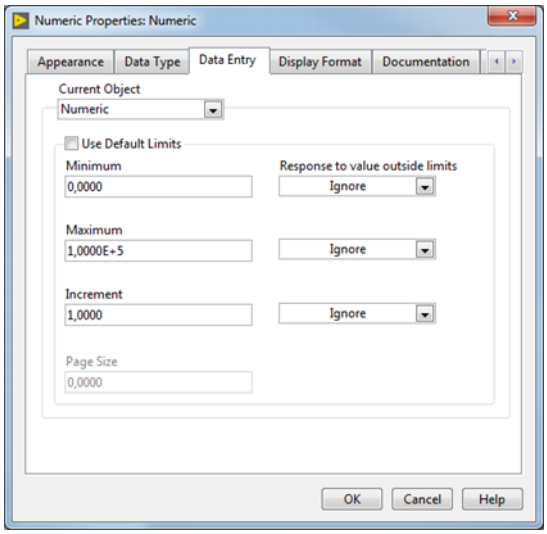


Рис. 4.2 - Диалоговое окно Data Range

3. Снимите выделение с пункта **Use Defaults** (использовать значения по умолчанию).

4. Выберите пункты, показанные в этом примере диалогового окна.

a. Установите **Minimum** (минимальное значение) равным 0 и выберите Coerce.

b. Установите **Maximum** (максимальное значение) равным 10000 и выберите Coerce.

c. Установите **Increment** (значение приращения) равным 1 и выберите Coerce to Nearest.

5. Выберите раздел **Display Format**

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 **Display Format** 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 **Display Format.** Появится следующее диалоговое окно **Display Format** (рис. 4.3).

7. Сделайте настройки, показанные ниже.

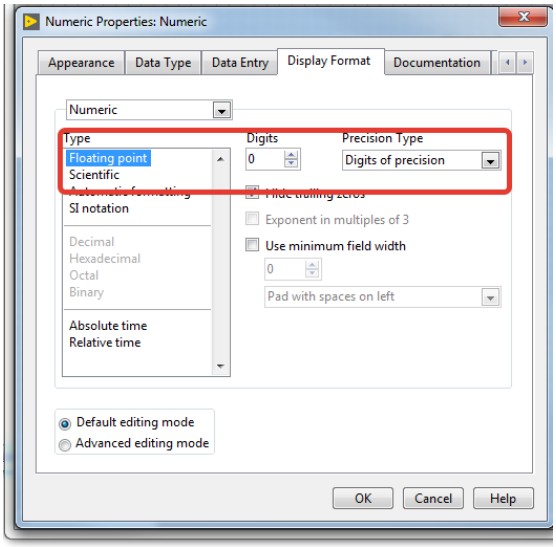


Рис. 4.3 - Диалоговое окно **Display Format**

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.

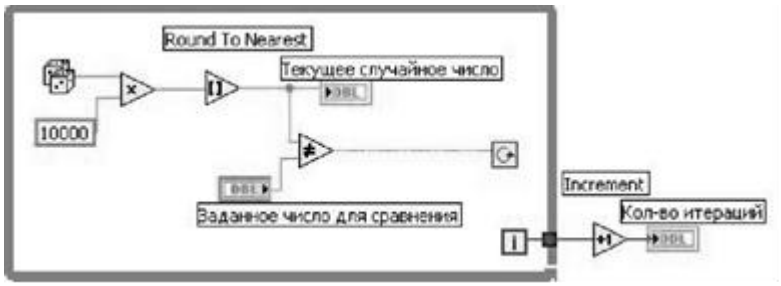


Рис. 4.4 - Блок-диаграмма ВП Подсчёт итераций

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

Function» 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 Function»Numeric. Эта функция будет округлять полученное в пределах от 0 до 10000 случайное число до ближайшего целого числа.

атор 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 Function»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 **Function» 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 **Function»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 **Function» 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. Lab VIEW автоматически приведет новое значение к ближайшему значению в указанном диапазоне входных данных элемента. **16**.Закройте ВП.