

## Уроки по LabVIEW

На этом уроке Вы создадите свое первое приложение, освоите начала технологии графического программирования, научитесь изменять и редактировать свойства графических элементов управления и индикации, использовать циклы типа *While-Do* и *For-Loop* в теле программы, сгенерируете массив данных и познакомитесь с последовательностью действий по организации файлового сохранения полученных данных на диске



Давайте создадим первое элементарное приложение шаг за шагом. Надеемся, что это поможет Вам почувствовать вкус программирования в среде LabVIEW.

- Запускаем программу.
- В появившемся окне выбираем опцию **New VI**.



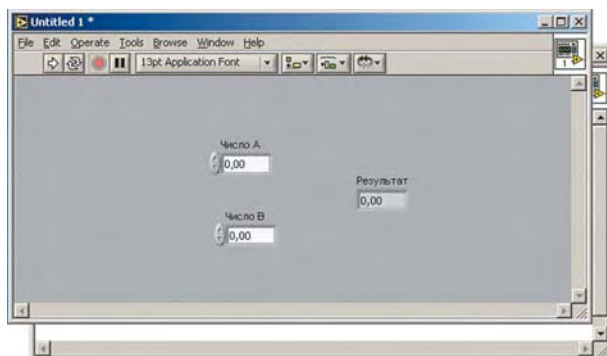
- Если панель управления неактивна, то ее следует вызвать через основное меню **Window >> Show Control Palette**.



- Используя указатель мыши в виде стрелки (переключение между инструментами производится клавишей **Tab**), установим его на интерфейсную панель **Digital Control**, который находится на панели управления (**Controls**) >> **Numeric**.



- Подпишем его, как "Число А". Для этой цели на панели инструментов Tools (вызов панели производится через **Window >> Show Tools Palette**), следует выбрать инструмент **Edit Text**, подвести указатель к метке, кликнуть и просто набрать необходимый текст.



- Аналогично предыдущим двум шагам устанавливаем и подписываем, как "Число В" еще один **Digital Control**. Это будут поля ввода наших параметров.
- Для отображения результата поместим на интерфейсную панель **Digital Indicator**, который также находится на панели управления (**Controls**) >> **Numeric**. Подпишем его, как "Результат". Должно получиться приблизительно так, как показано на картинке.

Теперь перейдем к основной части работы, а именно к графическому программированию. В отличие от других языков графического программирования, таких как, например, Borland Delphi или Microsoft Visual C++, нам не придется писать ни единой строки текстового кода, реализующего определенный алгоритм.

Создав визуальный интерфейс с двумя полями ввода чисел и одним цифровым индикатором, поставим и реализуем задачу, например, суммирования этих чисел. Для этого необходимо перейти в так называемое окно построения диаграмм, где мы видим три иконки (терминала), которые соответствуют полям ввода чисел и индикатору. Реализация простого или сложного алгоритма будет сводиться к элементарной последовательности действий, - а именно, к установке необходимых иконок, которые выполняют ту или иную функцию и связи (соединения) их между собой.

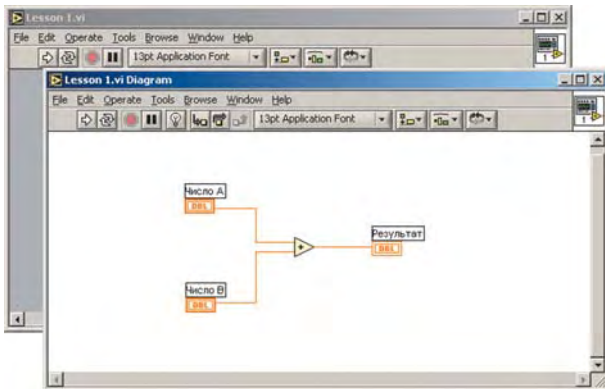


- Для суммирования чисел необходимо вызвать функциональную панель (**Functions**) и перетянуть треугольную иконку, соответствующую операции суммирования в окно редактирования диаграмм. Она находится в **Numeric >> Add**.

**Совет:** Для выбора необходимой функциональной иконки, которая находится в подменю любой сложности, используйте правую клавишу мыши, а для непосредственного выбора и перетягивания - левую.

Теперь только остается соединить необходимые контакты соединительной катушкой (**Wiring Tool**), которая размещена на панели инструментов (**Tools**). Подводим катушку к пиктограмме первого числа, нажимаем левую клавишу мыши, и не отпуская ее соединяем второй конец линии с одним из контактов пиктограммы суммирования. Для изменения направления связи потребуется еще один промежуточный щелчок левой клавишей мыши.

- Повторяем эти действия и для второго числа. Аналогично соединяем выход иконки суммирования со входом цифрового индикатора. Должна получиться функциональная диаграмма ("текст" программы) похожая на изображенную на рисунке.



Все, программа написана. Теперь остается запустить ее на выполнение и убедиться в ее работоспособности.

- Переходим на интерфейсную панель, запускаем программу на выполнение в циклическом режиме, нажав левой клавишей мыши на кнопке циклического запуска.
- Меняем значения полей ввода чисел, используя клавиатуру или мышь.

Для останова выполнения программы следует воспользоваться кнопкой линейки управления "**Abort Execution**".

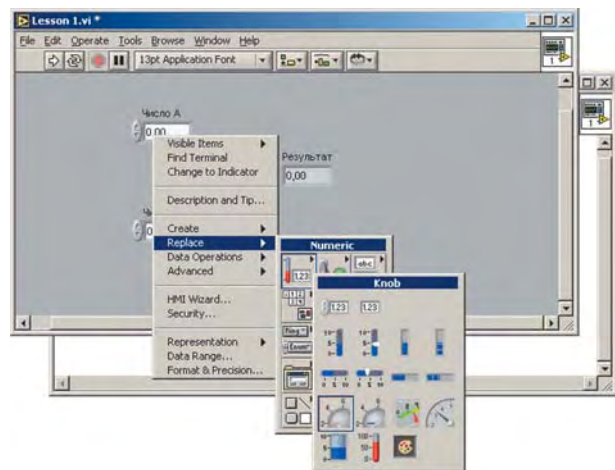
Сделаем еще один шаг и сохраним наше первое элементарное приложение на диске. Сохранение LabVIEW-программы аналогично записи, например, документа в Microsoft Word или Excel.

- Для первого сохранения программы необходимо выбрать в меню **File** пункт **Save**.
- В появившемся диалоговом окне необходимо выбрать или создать желаемую директорию (папку), ввести имя файла и подтвердить ввод.

Записанный нами файл сохранился с расширением **vi** (Virtual Instrument - виртуальный инструмент), и будет иметь вид **<имя файла>.vi**.

**Примечание:** *Файлы с расширением **vi** переносимы между различными платформами, будь-то Windows 9x/NT или Unix/Linux.*

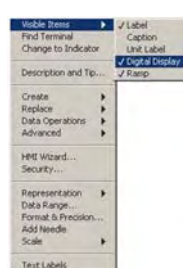
- Изменим внешний вид наших графических объектов. Для этого подводим указатель в виде стрелки на объект, соответствующий "числу А", и нажимаем правую кнопку мыши.
- В появившемся контекстном меню выбираем опцию замены (**Replace**). Далее входим в подменю **Numeric** и там выбираем шарообразную ручку управления (**Knob**).



**Примечание:** *В дальнейшем, любые манипуляции с графическими объектами (кнопками, ручками, дисплеями и т.п.), такие, как изменение их свойств, внешнего вида, режимов работы, формата, точности и других параметров, следует осуществлять вышеуказанным способом, т.е. используя нажатие правой кнопкой мыши на необходимом объекте.*

- Изменим размер ручки. Изменяя положение указателя, увидим, что в четырех точках он меняет вид со стрелки на окружности. В этот момент, нажав и удерживая левую кнопку мыши, изменяем вид ручки.

- Теперь разместим метку "Число А", выделив и переместив ее в необходимую позицию.

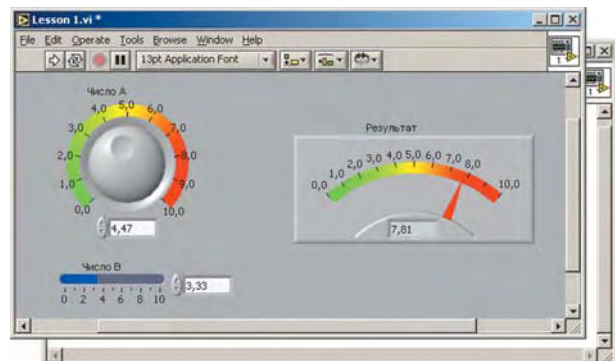


- Изменим атрибуты ручки "Число А". В выпадающем меню (нажав правую клавишу мыши) выберем изменение визуальных свойств объекта (**Visible Items**), а в них - **Ramp**.

- Для точного позиционирования ручки или отображения значения выберем еще и свойство **Digital Display**, которое также находится в **Visible Items**.

Разместите его на панели по-вашему усмотрению.

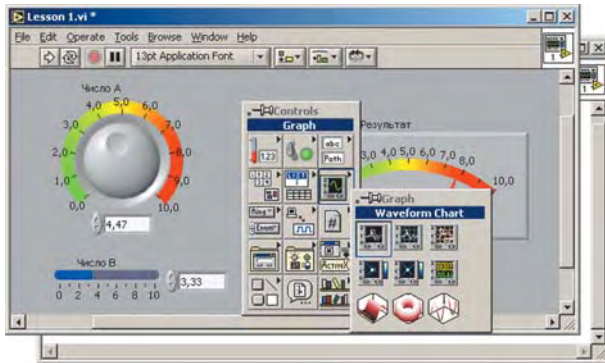
Проделайте аналогичные шаги для остальных элементов интерфейса. В результате получим похожий рисунок



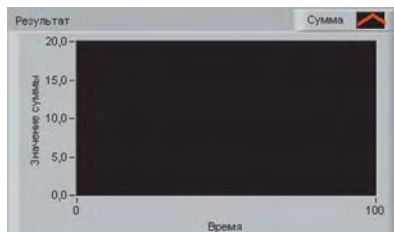
**Примечание:** *Для изменения диапазона вводимого или выводимого значения необходимо в инструментах (**Tools**) выбрать возможность редактирования текста (**Edit Text**), подвести курсор к начальному или конечному диапазону шкалы, нажать левую клавишу мыши и изменить значение на необходимое.*

Немного усложним задачу.

- Установим на панель графический экран, который будет отображать графическую зависимость значения суммы чисел от времени. Для этого на панели управления (**Controls**) выберем иконку **Graph**, а в появившемся подменю - **Waveform Chart**.

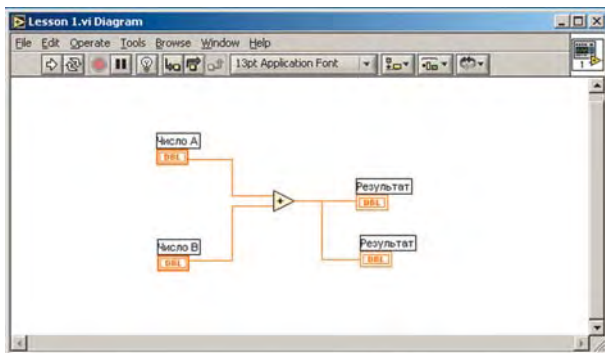


- Устанавливаем графический экран на панель и меняем его атрибуты и свойства так, как показано на рисунке:



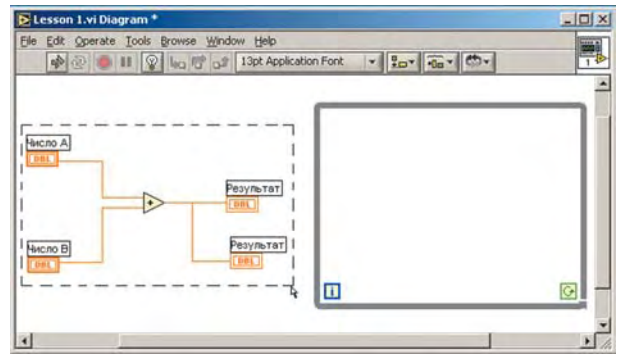
Теперь продолжим разработку нашего программного кода.

- Установим связь между значением суммы и диаграмм и соединим с помощью катушки (**Tools >> Wiring Tool**) необходимый контакт с соответствующей иконкой графического окна.



До этого времени мы запускали программу на выполнение в циклическом режиме. Теперь же мы "поставим" фрагмент нашей диаграммы в цикл, например, который является аналогом оператора "While". Условием выхода из цикла будет нажатие на кнопку останова. Для реализации поставленной задачи необходимо выполнить следующие шаги.

- В функциональном меню (**Functions**) выбрать структуры (**Structures**), а в них цикл **While-Loop**.
- Перетянуть его в окно редактирования диаграмм, увеличить до размера чуть большего уже созданной нами структурной схемы так, как показано на рисунке:



- Поместим базовую структуру в цикл. Для этого указателем в виде стрелки выделяем всю схему (удерживая нажатой левую клавишу мыши выделяем прямоугольную область куда попадают под выделение все необходимые элементы схемы), а затем перетягиваем выделенные элементы вовнутрь цикла.

**Примечание:** Процесс отладки и проверки исполнения приложения происходит во время составления диаграммы или во время формирования интерфейса, поэтому если кнопка запуска приложения является неактивной, приложение не запустится и будет выдано сообщение об ошибке. Это связано с тем, что отсутствует условие выхода программы из цикла.

Следуя вышеупомянутым замечаниям и поставленной задаче, выполним следующие действия чтобы все выполнялось корректно.

- Перейдем в окно редактирования диаграмм.
- Выберем на панели инструментов иконку соответствующую указательному пальцу (инструмент **Operation Tool**).
- Изменим вид иконки условия выхода из цикла нажатием левой кнопки мыши на соответствующей пиктограмме аналогично тому, как показано на рисунке:



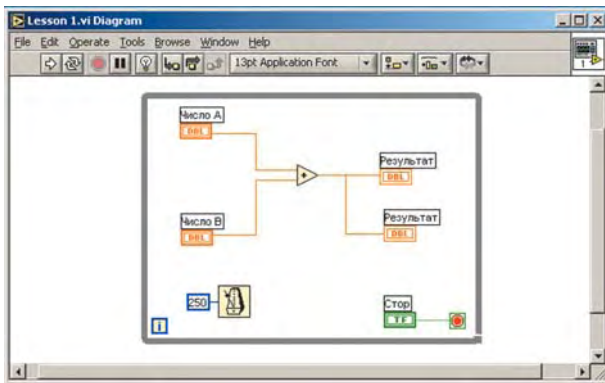
- Теперь нажмем на той же иконке только правой кнопкой мыши. В выпадающем меню следует выбрать пункт создания управляющего элемента (**Create Control**). В результате появится пиктограмма, соответствующая кнопке останова.

**Примечание:** Создание любого элемента управления или индикатора в окне редактирования диаграмм влечет за собой создание соответствующего графического компонента на главной интерфейсной панели.

- Чтобы изменения значений суммы, выводимой в виде графика выполнялось с определенной задержкой во времени (для большей наглядности анализа работы) установим из функциональной панели иконку **Functions >> Time & Dialog >> Wait Until Next ms Multiple**.

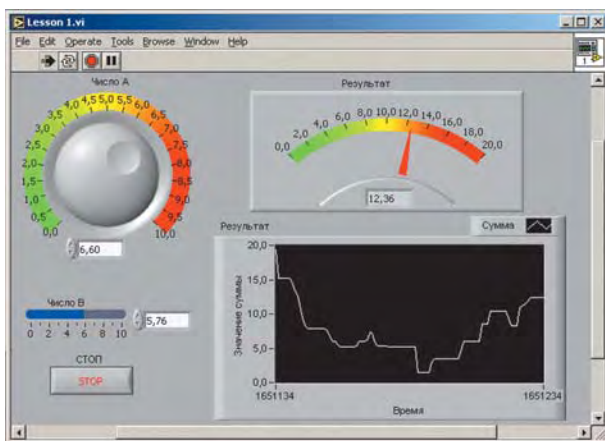
- Создадим для нее константу, соответствующую этой задержке. Для этого подведем указатель в виде катушки к левой части иконки, нажмем правую клавишу мыши и выберем пункт создания константы (**Create >> Constant**).

- 250 Введем значение с клавиатуры, равное, например, "250". В результате проделанной работы, диаграмма должна выглядеть следующим образом:



Теперь остается желаемым образом сформировать переднюю интерфейсную панель. Как и для любых визуальных приборов, регуляторов, ползунков, меню и т.д., для **Waveform Chart** также можно менять различные визуальные параметры и свойства.

- Запускаем программу на исполнение.
- В итоге, визуально, должен получиться похожий результат:



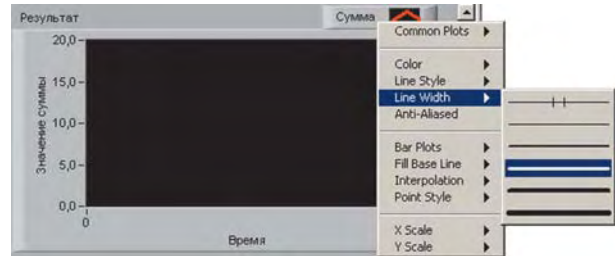
- Нажав на созданную нами кнопку "СТОП", мы остановим программу.
- Еще один штрих. Изменим константу, определяющую задержку, на один из визуальных элементов управления.
  - Выбираем на панели управления понравившийся регулятор, например **Horizontal Pointer Slider**.
  - Подписываем его необходимым образом.
  - Меняем размер и диапазон значений от 0 до 1000.
  - Переходим в окно редактирования диаграммы.
  - Удаляем соответствующую константу и оборвавшуюся связь следующим образом: выделяем константу и связь и нажимаем на клавиатуре клавишу "Delete".
  - Соединяем с помощью катушки контакт для установки задержки и задатчик.

Переходим на панель отображения и делаем перекомпоновку элементов на панели, меняем толщину и цвет линии прорисовки графика.

**Совет:** Изменение атрибутов цвета любого элемента на интерфейсной панели программируемого приложения удобнее всего осуществляется путем использования инструмента **Get Color**



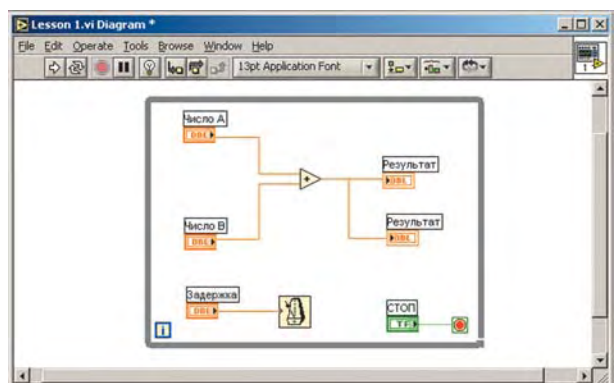
Редактирование свойств графического экрана **Waveform Chart** (в частности, цвет и толщина линии) осуществляется через обобщенное меню его свойств, вызов которого производится простым нажатием мыши инструментом **Operation Tool** в области верхней надписи:



В результате проделанных манипуляций должен получиться следующий графический интерфейс и код исполнения (диаграмма) программы.



Во время работы программы попробуйте изменить положение ползунка регулятора задержки. Видим, что так можно управлять скоростью отображения результата "вручную".

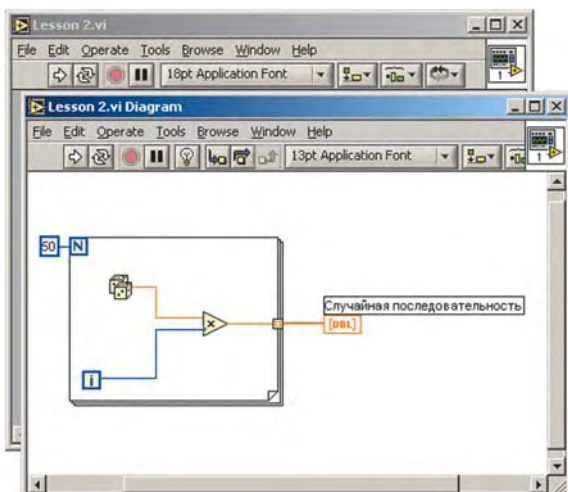


Надеемся, что Ваше первое приложение успешно функционирует.

При разработке реальных приложений приходится решать задачу, связанную с накоплением, организацией доступа и дальнейшей обработкой данных. Не существует универсальной и единой программы сбора и обработки данных, в которой были бы реализованы именно Ваши требования по сохранению и чтению данных. LabVIEW дает возможность реализовать тот или иной алгоритм не прибегая к кропотливому изучению процедур и функций,

как, например, в Visual/Borland C++ или Delphi. Все сводится к доступным методам графического построения диаграммы решаемой задачи.

В этом уроке мы затронем только малую долю всех возможностей графического программирования, связанных с записью и чтением данных. Но приобретая определенные навыки, Вы сможете в дальнейшем реализовывать самые замысловатые алгоритмы. Сначала реализуем элементарную задачу генерации массива случайных чисел. Для ее решения Вам понадобится составить диаграмму, которая показана на приведенном ниже рисунке. Следует обратить внимание на то, что вместо привычного цикла **While-Loop**, здесь используется цикл **For-Loop**. Условием завершения работы или выхода из такого цикла является равенство переменной цикла и числа, определяющего количество итераций. Другими словами, в LabVIEW для работы цикла типа **For-Loop** необходимо соединить пиктограмму **N** с константой, указывающей количество повторений. Для начала, результат будем выводить в виде графической зависимости случайных чисел от текущего значения числа итераций.

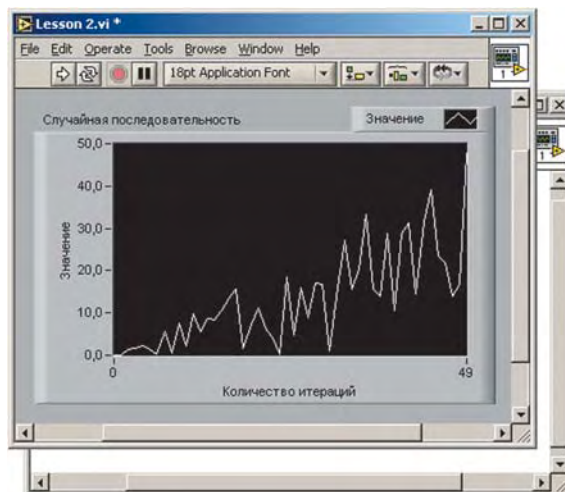


Для реализации поставленной задачи нужно выполнить следующие шаги:

- Создать новый VI: **File >> New VI**.
- Установить на интерфейсную панель элемент **Waveform Graph: Controls >> Graph >> Waveform Chart**. Изменить его внешний вид и свойства можно по своему усмотрению.
- Перейти в окно редактирования диаграмм.
- Перетянуть в окно редактирования вышеупомянутый цикл: **Functions >> Structures >> For Loop**. Изменить его размеры и местоположение (если нужно), чтобы была возможность устанавливать дополнительные компоненты вовнутрь. Следует сказать, что "i" — это переменная цикла.
- Щелчком правой кнопки мыши на пиктограмме цикла **N** выбираем создание константы (**Create Constant**) и вводим с клавиатуры значение, например 50.
- Создадим простейший генератор случайных чисел. Перетянем в середину цикла пиктограмму, которая позволяет генерировать случайные числа в диапазоне от 0 до 1: **Functions >> Numeric >> Random Number (0-1)**.

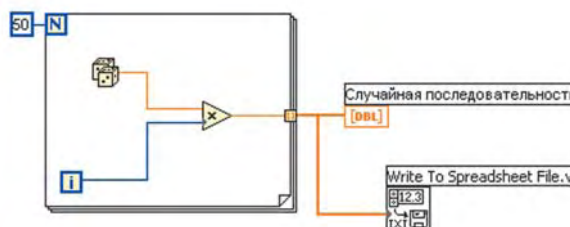


Соединим все компоненты так, как показано на рисунке, изображающем диаграмму. Запустим программу на выполнение и посмотрим на получившийся результат.

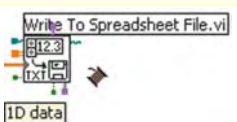


Выполнив первый шаг, связанный с генерацией и визуализацией случайной последовательности, перейдем к основной задаче - задаче записи данных на диск. Сначала мы используем самый простой, но, в то же время, элегантный подход для записи последовательности. Дословно он называется "Записью в Крупноформатную таблицу" (**Write to Spreadsheet File**). Используя соответствующую диаграмму (**VI** — виртуальный инструмент), без указания дополнительных параметров, например таких, как формат записываемого числа, создадим файл текстового формата, где через пробел последовательно будет записана сгенерированная последовательность.

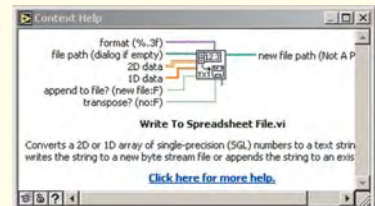
Выполним вышенамеченный план. Для этого перетянем из функционального меню **Functions**, пиктограмму, которая называется **Write to Spreadsheet File.vi: File I/O >> Write to Spreadsheet File.vi**, и соединим блоки так, как показано на рисунке:



**Совет:** Если подпрограмма LabVIEW (иконка **VI**) имеет несколько входов/выходов, то правильно подключиться поможет следующий метод. Выберете из инструментальной панели инструмент для соединения элементов (катушка) и "обследуйте" им иконку. При этом, будут активизированы и названы соответствующие входы/выходы. Чтобы видеть все "контакты" VI сразу, удобно воспользоваться окном контекстной помощи (вызов **Help >> Show Context Help**). Например окно контекстной помощи для **Write to Spreadsheet File.vi** выглядит так:



Format (%.3F)  
file path (dialog if empty)  
2D data  
ID data  
append to file? (new file:F)  
transpose? (no:T)

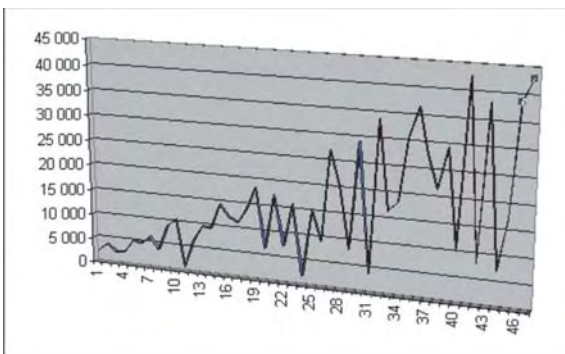


Так как кроме входа "ввод одномерного массива" (1D data), другие входы (формат, путь и т.п.) нами не подсоединялись, то LabVIEW сформирует следующую последовательность действий:

- Сначала будет создан массив из 50-и чисел;
- Результат будет отображен в виде графика;
- Появится стандартное диалоговое окно, в котором будет предложено выбрать директорию и имя файла, для сохранения данных;
- Если файл до этого не существовал, то он будет создан и в него будет записан массив чисел, а если он уже был на диске, то будет предложено добавить данные в файл или заменить уже существующие;
- После записи данных на диск программа сама завершит свое выполнение.

Вы можете легко убедиться в этом, когда перейдете в окно интерфейсной панели и запустите программу на исполнение.

Вводим имя файла: **random.xls**. Расширение имени файла **.xls** было выбрано не случайно, а с целью продемонстрировать возможность того, что файл созданный в LabVIEW можно с легкостью использовать и для работы с другими приложениями, например, с **Microsoft Excel**. В этом нет ничего удивительного, т.к. данные были сохранены в обычном текстовом формате. Поэтому не составит труда построить график в Excel, аналогичный графику в LabVIEW, используя Мастер Диаграмм Microsoft Excel. В результате должна получиться похожая картинка:



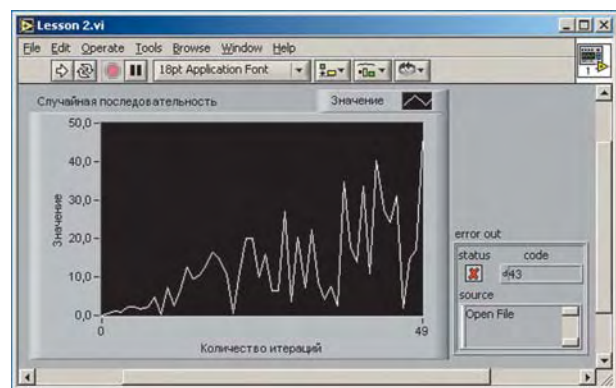
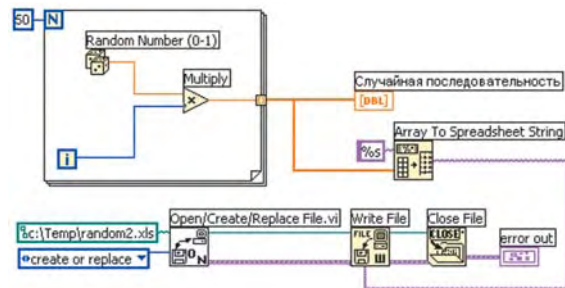
До этого времени мы использовали высокоуровневую функцию записи данных на диск. Для реализации более гибкого алгоритма можно использовать функции более низкого уровня. Основные отличия заключаются в том, что функции создания или открытия файла, записи или чтения данных и закрытия файла выполнены как отдельные функциональные блоки. Составим диаграмму, которая как раз и будет использовать виртуальные инструменты низкоуровневого ввода/вывода.

- Модифицируем предыдущую программу или создаем новый файл.
- Устанавливаем в окне редактирования диаграмм все необходимые блоки, как показано на рисунке: **Functions >> File I/O >> Open/Create/Replace File.vi, Write File, Close File**.
- Небольшим отличием в программе будет то, что формат записываемых данных будет изменен: **Functions >> String >> Array To Spreadsheet String**.
- Соединяем блоки между собой. Для **Close File** создаем индикатор сообщения об ошибках записи. Для этого указатель мыши в виде катушки подводим к контакту **Error Out**, нажимаем правую кнопку



мыши, выбираем **Create >> Indicator**. При этом на интерфейсной панели появится его графический эквивалент.

- По аналогии создайте константы и для других блоков. Для модуля преобразования массива данных в таблицу строк, используйте константу форматирования **%s** (строка).



После запуска программы на выполнение данные автоматически запишутся в заданный файл. Поскольку файл данных будет уже существовать, после попытки повторного запуска программы появится диалоговое окно, где нужно будет выбрать между заменой существующего файла на новый или отменой записи в целом.

**Задание:** Используя текстовый редактор, просмотрите содержимое созданного файла и сравните его с содержимым файла, записанного в результате работы предыдущей программы.

0,000000	0,518394	0,025902	0,578916	2,992766	2,043003	5,174921
0,528943	0,293538	2,402827	7,360899	2,344953	13,678138	9,598111
0,251507	11,193800	17,929595	2,396556	11,877293	4,737813	9,598111
236082	6,411871	18,205469	25,409592	18,429205	5,279468	10,321269
172	14,260775	18,156509	25,452578	17,613483	13,056872	26,082162
7	37,762342	34,730999	17,946161	7,404865	36,481578	43,837788

Надеемся, что полученный Вами результат повторяет наши данные.

На этом можно было бы и закончить первый урок. Но многие из Вас, проникшись духом LabVIEW, не станут дожидаться следующего выпуска журнала и наверняка продолжат самостоятельно осваивать его. Приведенный ниже справочный материал будет Вам хорошим помощником.

**Авторы - сотрудники "ХОЛИТ Дэйта Системс"**  
т. (044) 241-87-39, 241-67-54  
e-mail: info@holit.com.ua

## Терминалы и связи в LabVIEW

Программирование потоков передачи данных в LabVIEW осуществляется при помощи механизма графического связывания терминалов. Терминалы показывают типы данных элементов управления или индикаторов. LabVIEW оперирует различными видами терминалов управляющих элементов, индикаторов, узлов, констант, специализированных структур. Различия во внешнем виде терминалов (цвет и символ) характеризуют тип представляемых данных:

Элементы управления	Индикатор	Тип данных
		Число с плавающей запятой обычной точности (Single-precision floating-point numeric)
		Число с плавающей запятой двойной точности (Double-precision floating-point numeric)
		Число с плавающей запятой повышенной точности (Extended-precision floating-point numeric)
		Комплексное число с плавающей запятой обычной точности (Complex single-precision floating-point numeric)
		Комплексное число с плавающей запятой двойной точности (Complex double-precision floating-point numeric)
		Комплексное число с плавающей запятой повышенной точности (Complex extended-precision floating-point numeric)
		Целое 8-ми битовое число со знаком (Signed 8-bit integer numeric)
		Целое 16-ти битовое число со знаком (Signed 16-bit integer numeric)
		Целое 32-х битовое число со знаком (Signed 32-bit integer numeric)
		Целое 8-ми битовое число без знака (Unsigned 8-bit integer numeric)
		Целое 16-ти битовое число без знака (Unsigned 16-bit integer numeric)
		Целое 32-х битовое число без знака (Unsigned 32-bit integer numeric)
		Перечисляемый тип (Enumerated type)
		Дискретный (Boolean)
		Строка (String)
		Массивы (Array)
		Кластеры разных типов данных (Cluster)
		Путь (Path)
		Временная диаграмма (Waveform)
		Номер ссылки (Reference number)
		Универсальный тип данных (Variant)
		Полиморфный (Polymorphic)
		Имя устройства ввода/вывода (I/O name)
		Рисунок (Picture)









  

Совет:	
Поиск на блок-схеме терминалов, локальных переменных, ссылок и узлов, связанных с элементами управления передней панели.	Щелкнуть правой кнопкой мыши на элементе управления передней панели и выбрать в меню опцию <b>Find</b> . Такой же подход работает при поиске элементов передней панели с помощью терминалов блок-схемы. Кроме того, те же результаты приносит двойной щелчок левой кнопкой мыши на элементе передней панели или блок-схемы.
Поиск текста или объекта в памяти.	Выбрать опцию <b>Find</b> пункта главного меню <b>Edit</b> . Можно воспользоваться комбинацией клавиш <b>Ctrl + F</b> .
Поиск <b>VI</b> , глобальных переменных, функций или текста.	В окне <b>VI</b> иерархии вызов через <b>Browse &gt;&gt; Show VI Hierarchy</b> набрать <b>Edit &gt;&gt; Find</b> . Затем выбрать тип объекта или набрать текст.
Открыть переднюю панель подпрограммы ( <b>subVI</b> ).	Произвести двойной щелчок левой кнопкой мыши на иконке подпрограммы.
Открыть блок-схему подпрограммы.	Произвести двойной щелчок левой кнопкой мыши на иконке подпрограммы при нажатой кнопке <b>Ctrl</b> .

Линии связи между терминалами отображают потоки данных в разрабатываемом приложении. Данные могут передаваться только в одном направлении: от источника к приемнику сигнала. При этом цвет и толщина линий связи характеризуют типы передаваемых данных:

Переменная	Скалярная величина	Одномерный массив	Двумерный массив
Аналоговая			
Дискретная			
Строка			

## Средства отладки приложений LabVIEW

Обнаружение ошибок	
	Когда созданный виртуальный инструмент содержит ошибки и не может быть исполнен, кнопка запуска в линейке инструментов принимает "разрушенный" вид (Broken Run button). Список обнаруженных ошибок выдается в ответ на щелчок мышью в области индикатора. После выбора в списке ошибки и нажатия кнопки Find программа выделит содержащий ошибку объект либо связь.
<b>Broken Run button</b>	
Исполнение с подсвечиванием	
	Эта функция позволяет анимировать исполнение блок-схемы программы при нажатии на кнопку исполнения с подсвечиванием (Execution highlighting button). Такой режим используется совместно с режимом пошагового исполнения и позволяет визуализировать процесс передачи данных между элементами исполняемого кода программы.
<b>Execution highlighting button</b>	
Пошаговый режим	
	Запуск пошагового режима исполнения программы (последовательное исполнение от узла к узлу) производится нажатием на кнопку Step Into button или Step Over button. При этом начинает мигать первый узел программы, что означает его готовность к выполнению.
<b>Step Into button</b>	
	Для выполнения шага достаточно нажать на кнопку Step Into button или Step Over button. Если следующий узел является структурой или виртуальным инструментом, то Step Over button приводит к выполнению всего узла без реализации пошагового режима внутри. Для выполнения пошагового режима внутри структур или VI необходимо выбирать Step Into button.
<b>Step Over button</b>	
	Нажатие на кнопку Step Out button приводит к прекращению исполнения блок-схемы узла и выходу из шагового режима.
<b>Step Out button</b>	
Пошаговый режим со входом в подпрограммы	
	При комбинации пошагового режима и исполнения с подсвечиванием, когда подпрограмма LabVIEW (subVI) исполняется, вид ее иконки на блок-схеме главной программы дополняется изображением зеленой стрелки. При этом на передний план выдвигается исполняемый код подпрограммы и Вы можете либо осуществлять ее пошаговую отладку, либо вернуться назад в тело вызывающей ее главной программы.
<b>SubVI исполняется</b>	
Использование пробника	
	Использование инструмента Probe tool позволяет просматривать значение переменных при исполнении настраиваемого приложения в требуемом месте блок-схемы. Для этого необходимо выбрать "пробник" в панели инструментов (Tools) и щелкнуть на линии связи.
<b>Probe tool</b>	
Использование точек останова	
	В процессе отладки Вам может понадобиться остановить выполнение программы в том или другом месте, например, чтобы зафиксировать данные посредством "пробника". Используя инструмент Breakpoint tool можно создать точку останова в любом месте блок-схемы исполняемого приложения, будь-то узел или линия связи.
<b>Breakpoint tool</b>	

## ИЕРАРХИЯ виртуального инструмента

Использование окна иерархии позволяет наглядно представить как происходит расчет алгоритма программы в целом и какие подпрограммы используются. Окно иерархии вызывается из главного меню LabVIEW путем выбора пунктов **Browse >> Show VI Hierarchy**. Оно представляет собой графическую интерпретацию дерева используемых в приложении подпрограмм. Причем материнские **VI** помещаются сверху, а вызываемые ими дочерние **subVI** - снизу.

