Национальный исследовательский университет «МЭИ»

Кафедра Радиотехнических систем

Информационные технологии

Лабораторная работа №1

*«Знакомство с LabView»*

Студент: Жеребин В.Р.

Группа: ЭР-15-15

Москва

2017

**1. Введение**

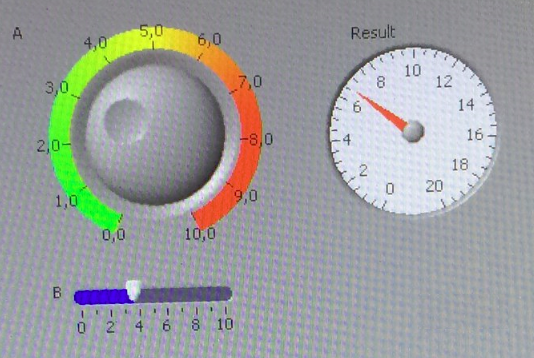
LabVIEW — язык графического программирования, в котором для создания приложений используются графические образы (иконки) вместо традиционного текстового кода. От пользователя пакета не требуется знаний языков программирования, но понятие об алгоритме, цикле, выходе по условию и т.п. конечно иметь нужно. Все действия сводятся к графической работе в двух окнах – окне сборки структурной схемы (начинки), и окне «лицевой панели» прибора. Все объекты собираются из уже готовых библиотечных объектов, называемых *Виртуальными Инструментами* (*VI*).

Процесс освоения LabVIEW существенно облегчается благодаря наличию интерактивной обучающей системы, разветвленной контекстнозависимой помощи и множества примеров использования приемов программирования (но все это на английском, немецком и даже китайском, но не на русском языке). В работе рассмотрен пакет LabVIEW на примере версии 6.1, который, возможно, станет для Вас одним из средств разработки прикладного программного обеспечения.

**Лабораторное задание**

**2.1. Простая арифметика**

Создадим первое элементарное приложение, состоящие из 3 элементов группы *Numeric*, панели *Controls.*

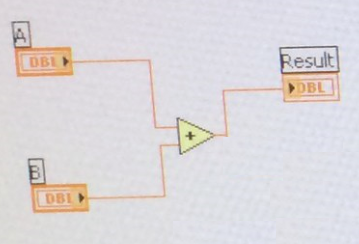


Где А и В – элементы Digital Control, которые позволяют вводить значения,

С – элемент Digital Indicator, который выводит значения на интерфейсную панель.

На структурной схеме соединили элементы «проводами» с помощью инструмента «Катушка». Преимущество графического программирования в том, что не придется писать ни одной строки кода.

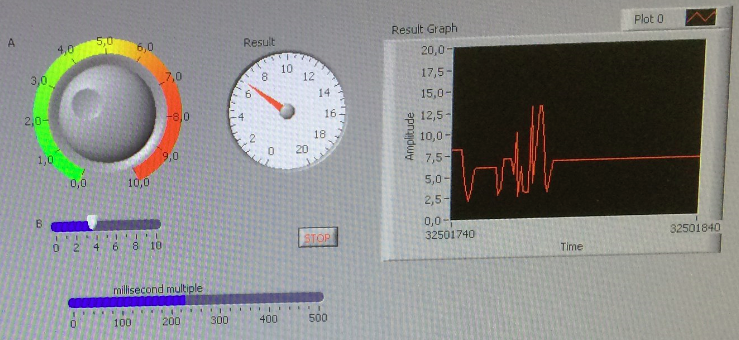
Так же добавим на схему элемент *Add* из такой же группы *Numeric* панели *Functions.*



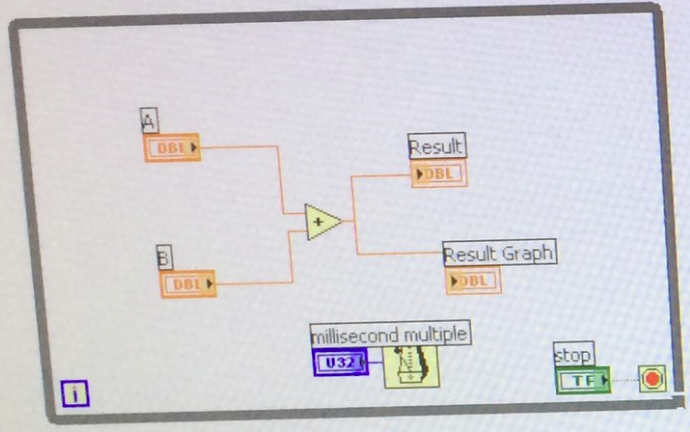
**2.2. Работа с циклами**

На данном этапе наша программа запускается на бесконечное выполнение, останов производится вручную. Усложним задачу, ограничив условия ее выполнения, и добавим новые элементы. Сперва вернем на место функцию суммы и пределы [0-10] для чисел A и B.

Установим графический экран для фиксирования результатов суммы во времени. На лицевую панель прибора добавим элемент *Graph >> Waveform Chart.* Зададим максимальное значение по оси Y, равное сумме максимальных возможных чисел А и B = 20, изменим цвет линии правым щелчком мыши на ломаной кривой *(Color)*, подпишем график, чтобы не путаться на структурной схеме, и соединим катушкой с результатом суммирования.



Теперь поместим наш ВП в цикл. Циклы в LabView представлены в виде рамок на структурной схеме. Все, что буквально содержится внутри рамки, выполняется внутри цикла. Добавим цикл *While Loop* из группы *Structures* на структурную схему.

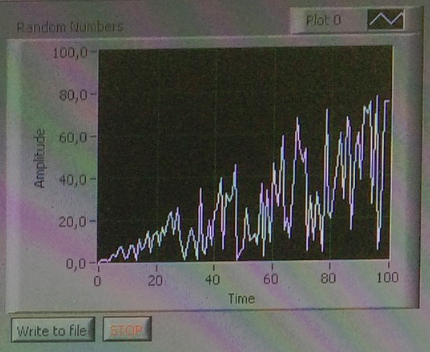
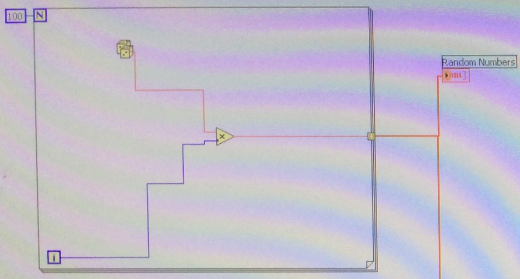


Условие работы цикла задается двоичной переменной типа Истина/Ложь и подключается в виде кнопки к пиктограмме круговой стрелки. По умолчанию замкнутая на себя стрелка означает *«Продолжать, если Истина»*. Нажав по стрелке «Пальцем», изменим условие на *«Останов, если Истина»*. Для автоматического добавления кнопки нажмем ПКМ на появившейся красной кнопке *>> Create Control.* Такую операцию можно выполнять для многих элементов. В результате на лицевую панель добавится кнопка STOP. Поскольку каждое новое значение суммы рассчитывается очень быстро и тут же выводится на график в виде постоянного уровня, добавим в цикл задержку. Она позволит сохранить все предыдущие значения в виде кривой. Задержка (метроном) расположена в *Functions >> Time & Dialog >> Wait Until Next ms Multiple.* Добавим регулятор задержки и установим пределы 0-500.

Рассмотрим второй наиболее часто используемый тип цикла – *цикл For Loop.* Цикл *For*, в отличие от цикла *While*, имеет фиксированное число итераций.

Добавим цикл For из *Functions >> Structures >> For Loop.* В цикле

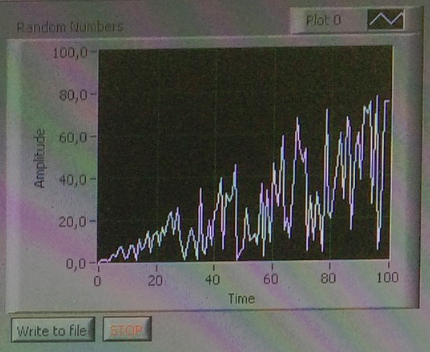
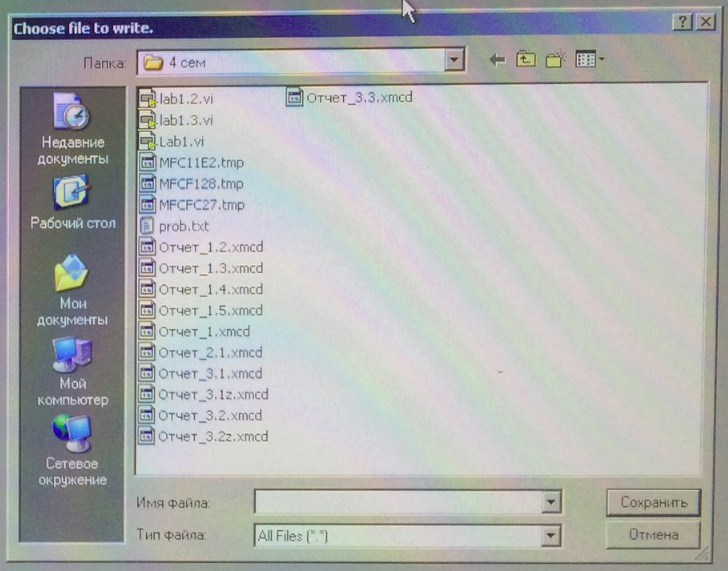
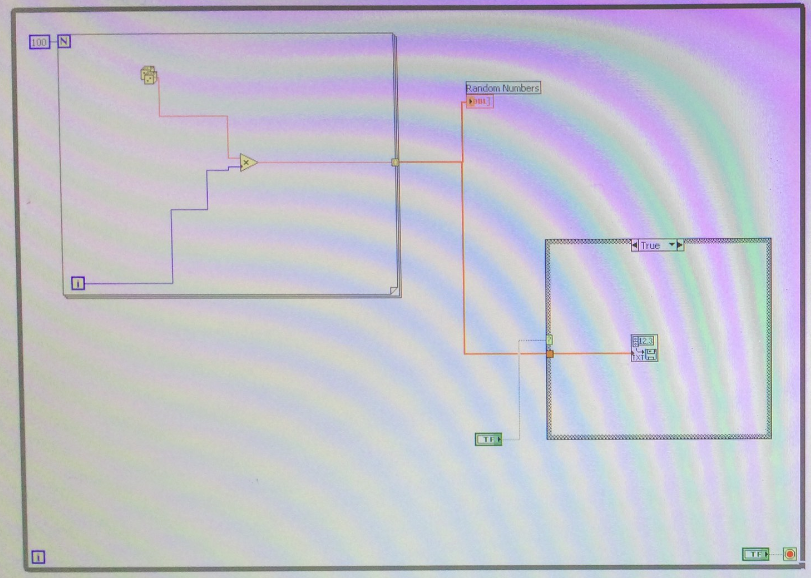
присутствует индекс текущей итерации *i* и общее число итераций *N.* Щелчком ПКМ на *N >> Create Constant* задаем число итераций 100. Добавим внутрь цикла элемент «Произвольное число в интервале [0-1]» из *Functions >> Numeric >> Random Number (0-1)*. Добавим умножение и подключим к первому входу «Кубики», ко второму – индекс итерации *i*. Добавим на лицевую панель график *Controls >> Graph >> Waveform Graph.* Зададим максимальное значение по оси Y = 100. Выключим авто масштаб по оси Y, нажав на графике ПКМ *>> Y scale >> AutoScale Y*.



**2.3. Запись в файл.**

Научимся сохранять выборки данных в текстовый файл. Для этого модифицируем наш ВП.

Добавим на схему элемент «Если, то» из *Functions >> Structures >> Case Structure.* Данный блок принимает на вход [?] переменную типа Истина/Ложь и, в зависимости от ее значения, выполняет действия внутри блока под вкладкой True/False соответственно. Добавим внутрь блока *Case* во вкладку *True* элемент, осуществляющий запись в файл, из *Functions >> File I/O >> Write To Spreadsheet File.vi.* Нажатием ПКМ на символе [?] блока *Case >> Create Control* добавим кнопку записи. Двойным щелчком на этой кнопке перейдем к ее представлению на лицевой панели. Настроим механизм срабатывания: нажмем ПКМ *>> Mechanical Action >> Latch When Released.* Соединим провод, идущий от «Кубиков» к графику, с блоком записи в файл. Добавим уже известный внешний цикл While с кнопкой останова.



Запись файла на диск: