ПЕРВОЕ ВЫСШЕЕ ТЕХНИЧЕСКОЕ УЧЕБНОЕ ЗАВЕДЕНИЕ РОССИИ



МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение высшего образования

**«САНКТ-ПЕТЕРБУРГСКИЙ ГОРНЫЙ УНИВЕРСИТЕТ**

**ИМПЕРАТРИЦЫ ЕКАТЕРИНЫ II»**

Кафедра автоматизации технологических процессов и производств

**Лабораторная работа №2**

|  |  |
| --- | --- |
| По дисциплине: | Программное обеспечение систем управления и сбора данных |
|  | (наименование учебной дисциплины согласно учебному плану) |

|  |  |
| --- | --- |
| Тема работы: | Основы программирования виртуальных приборов |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Выполнил: студент гр. | | |  | АПГ-22 |  |  |  | Скрябнев А.В. | |
|  | | |  | (шифр группы) |  | (подпись) | |  | (Ф.И.О.) |
|  |  |

|  |  |  |
| --- | --- | --- |
| Дата ­­­\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |  |
|  |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Проверил  руководитель работы: |  |  | доцент |  |  |  | Лебедик Е.А. |
|  |  |  | (должность) |  | (подпись) |  | (Ф.И.О.) |

Санкт-Петербург

2025

1 Цель работы

Создание простейшего виртуального прибора, имитирующего сбор данных и вывод информации в числовой и графической форме.

2 Основные теоретические положения

Labview является высокоуровневым языком программирования и реализует концепцию графического программирования G. LabVIEW включает основные конструкции управления:

* переменные (локальные или глобальные)
* ветвление (case structure)
* For – циклы с проверкой завершения и без.
* While – циклы
* Группировка операций.

Для выполнения программы n количества раз или для перехода по условию в Labview используются циклы (рисунок 1):

1. **Цикл по условию** (*While Loop*) – осуществляет многократное повторение операции на потоком данных, пока не выполнится логическое условие выхода. Цикл While расположен на палитре функций в разделе Структуры (Programming→Structures). По умолчанию, терминал условия выхода указывает, что цикл будет выполняться до поступления значения Ложь (False). Слева располагается счетчик итераций, в котором можно задать необходимое количество циклов.
2. **Цикл с фиксированным числом итераций** (*For Loop*) – выполняет повторяющиеся операции над потоком данных определенное количество раз. В верхнем левом углу располагается терминал N в котором указывается максимально число итераций. Цикл For Loop отличается от цикла по условию тем, что завершает работу, выполнив заданное максимальное количество итераций N.

Для ветвления программы применяется **Структура Вариант** (*Case Structure*), которая аналогична операторам *case* или *if then else* в текстовых языках программирования. По умолчанию структура Вариант является логической и имеет два варианта - ИСТИНА (TRUE) и ЛОЖЬ (FALSE), выбираемые с помощью терминала селектора структуры варианта. Структура автоматически преобразуется в числовую или строковую при подключении соответственно числовой или строковой переменной.

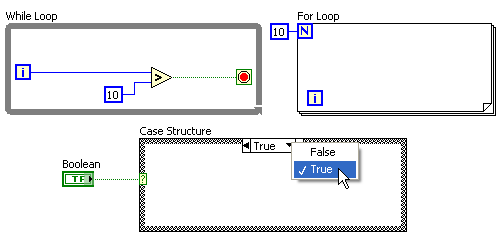


Рисунок 1 - while / for циклы и if / then / else структура

Для хранения данных, накопленных во время работы циклов, применяются **Массивы** (*Array*) (рисунок – 2). Элементом массива может быть любой тип данных, за исключением массива, таблицы или графика. Доступ к элементам осуществляется с помощью индексов. Значения индексов лежат в диапазоне от 0 до N-1, где N - количество элементов массива. Для создания массива элементов управления или индикации данных необходимо выбрать шаблон массива из палитры Controls **→**Array & Cluster и поместить его на лицевую панель.

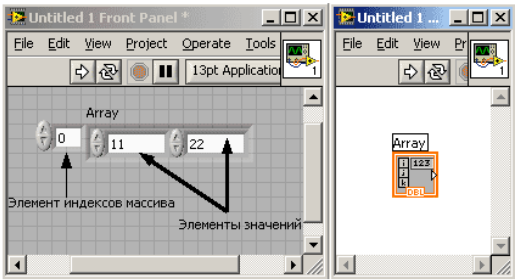


Рисунок 2 – Массив числовых элементов управления

3 Ход работы

1. Отображение графического и числового индикатора на «Front Panel» и придание логики генерации числа в диапазоне от 4 до 5 в «Block Diagram» (реализация имитации измерительного сигнала) (Рисунок 3).

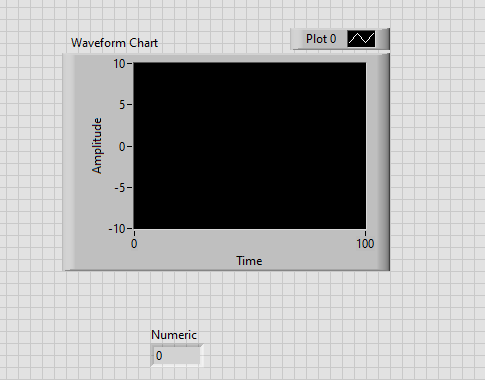
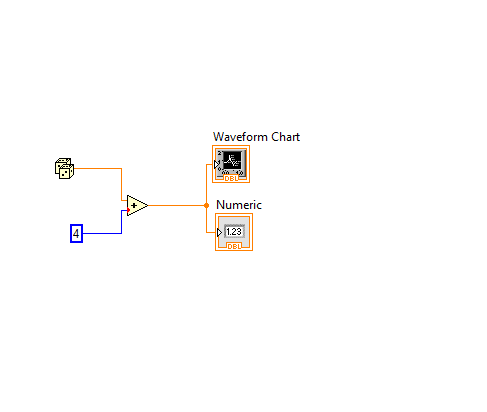


Рисунок 3 – Отображение наполнения «Front Panel» и «Block Diagram»

При нажатии на кнопку «Run Continuously ВП» в режиме просмотра потока данных, программа заработает циклично и непрерывно (Рисунок 4).

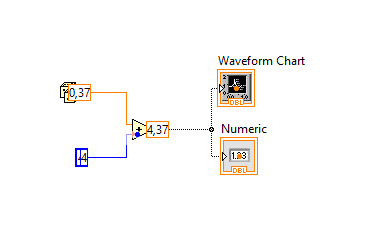
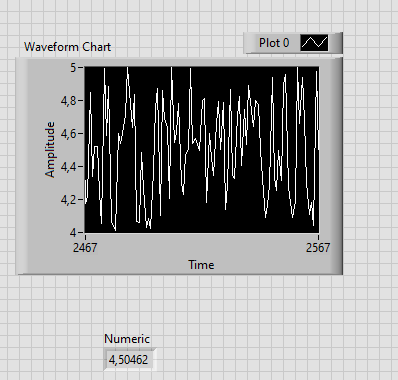


Рисунок 4 – Программа при нажатии на кнопку «Run Continuously ВП»

1. Однако у данного способа нет условия выхода из цикла (только в случае ошибки кода), поэтому лучше использовать цикл «while». Для этого в «Block Diagram» в «Express>> Execution Control (v7)» и выберем «while» и выделим кусок кода, который хотим зациклить (Рисунок 5).

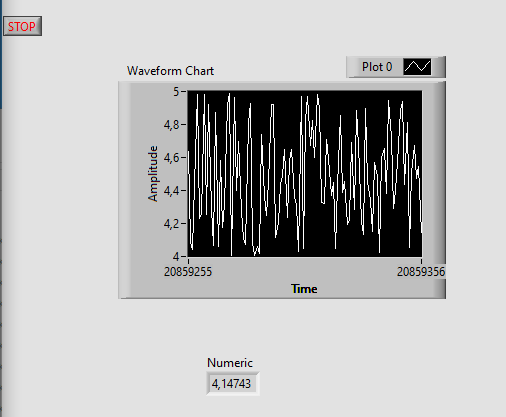
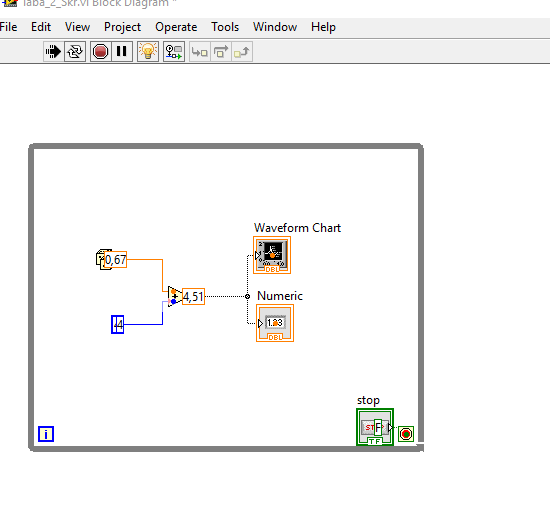
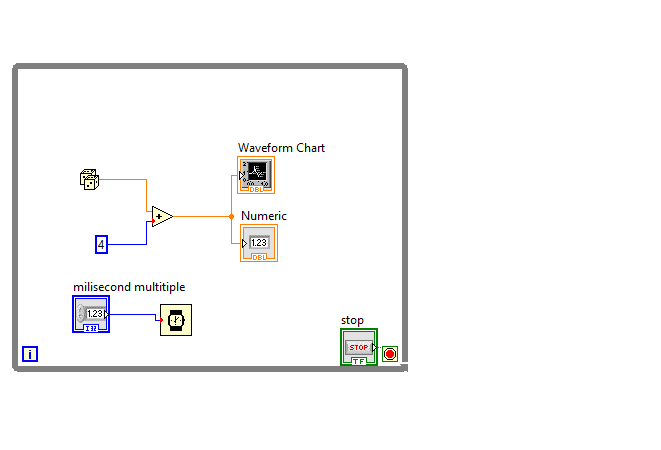


Рисунок 5 – Цикл «while»

Однако переменные генерируются в цикле слишком быстро, чтобы это исправить необходимо добавить в цикл «while» функцию wait, которая добавляет задержку исполнения цикла на заданное значение времени (в миллисекундах) (Рисунок 6).



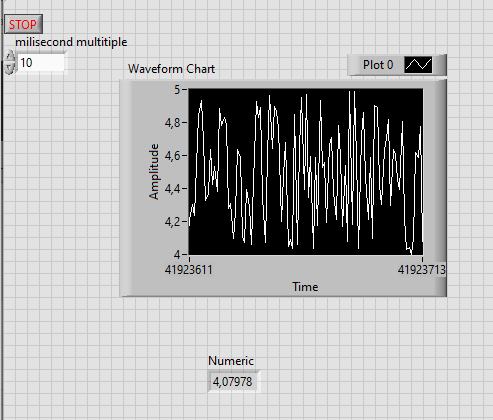


Рисунок 6 – Функция «wait»

1. Теперь добавим условия остановки цикла при генерации значения равном:

Реализация подобной логики изображена на рисунке 7.

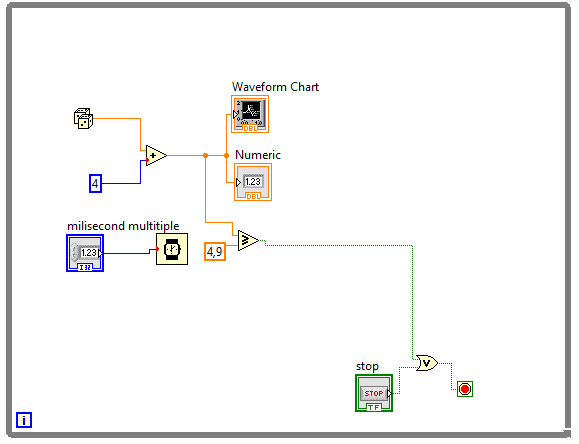


Рисунок 7 – Реализация дополнительного условия остановки цикла

4 Вывод

В ходе лабораторной работы были получены навыки реализации циклов в среде Labview, а также особенности работы с ними.