Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования «Московский государственный технический университет им. Н. Э. Баумана»

Среда проектирования виртуальных приборов LabVIEW

Методические указания к лабораторным работам

по курсу

«Расчет и синтез приборов и систем на ЭВМ»

Составил: А.В. Полынков

Москва 2015 год

Лабораторная работа №1.
Пример построения виртуальных приборов

Цель работы – изучение возможностей среды проектирования виртуальных приборов LabVIEW .

Назначение среды проектирования – LabVIEW — интегрированная среда разработчика для создания интерактивных программ сбора, обработки данных и управления периферийными устройствами.

Состав лабораторной установки:

- Два персональных компьютера соединенных нуль-модемным кабелем RS-232
- Программное обеспечение LabVIEW 7.0

Основные теоретические сведения

Средства автоматизации сбора, анализа и обработки экспериментальных данных

Автоматизация сбора, анализа и обработки экспериментальных данных требует решения трех основных задач:
- выбора типа вычислительно-управляющего устройства для работы в составе измерительно-вычислительного комплекса;
- установление связи с этим устройством той части оборудования, которая по условиям работы должна взаимодействовать с ним в ходе эксперимента;
- программирование работы вычислительно-управляющего устройства для выполнения всем комплексом определенных функций в ходе измерений.

Для решения проблем, связанных с созданием экспериментальных комплексов, необходимо иметь представление о современных технических и программных средствах, используемых при их организации.

Измерительная аппаратура эксперимента определяется конкретной задачей, для решения которой создается измерительная система. В каждой области исследования обычно существует свой базовый набор измерительной аппаратуры. Так при проведении экспериментальных работ в гироскопической и навигационной технике часто используют различные устройства аналого-цифрового и цифро-аналогового преобразования, измерительные приборы, предназначенные для генерации сигналов с различными характеристиками и для определения характеристик сигналов.

Вычислительные устройства в составе экспериментальных комплексов выполняют не только вычислительные операции, но могут также выполнять определенные действия по управлению ходом эксперимента и моделированию работы устройств. В качестве устройств, выполняющих вычислительные и управляющие функции, в автоматизированных системах используют различного типа ЭВМ или специальные программируемые микроконтроллеры. Решение вопроса о выборе того или иного вычислительного средства для использования в конкретном эксперименте зависит от тех функций, которые возлагаются на это устройство в ходе измерений. К числу основных функций вычислительных и управляющих средств, используемых в режиме реального

времени, относятся накопление экспериментальных данных, их предварительная или полная обработка, представление результатов обработки в процессе эксперимента в удобном для экспериментатора виде управление экспериментом по заданному алгоритму с учетом особенностей, возникающих непосредственно в процессе измерений. Принципиально не существует непреодолимых технических трудностей для использования любой ЭВМ в составе экспериментальных установок. В первую очередь основными критериями пригодности определенного типа ЭВМ для этой цели являются:

- соответствие вычислительной мощности сложности алгоритма и скорости обработки информации в ходе измерений,
- наличие необходимого объема запоминающих устройств, предназначенных для накопления и хранения экспериментальных данных,
- достаточная гибкость взаимодействия с нестандартными по отношению к данной ЭВМ внешними устройствами (измерительной аппаратурой, используемой в эксперименте).

По целому ряду причин наибольшее признание и применение при создании современных экспериментальных систем получили персональные ЭВМ.

Разработка специальных программируемых устройств, предназначенных для выполнения ограниченных вычислительных и управляющих функций в ходе исследований, связана с несколькими обстоятельствами. Универсальные промышленные ЭВМ, включаемые в состав экспериментальных установок, в ряде случаев обладают значительной избыточностью в отношении аппаратных и программных средств, которые необходимы для обеспечения конкретных измерений. Такая ситуация возникает обычно при использовании стандартных ЭВМ в так называемых рутинных измерениях, т.е. когда в течение длительного промежутка времени проводятся однотипные измерения с обработкой информации по сравнительно несложному алгоритму. Иногда, например, при работе в экстремальных условиях, применение стандартных ЭВМ связано с техническими трудностями. В таких случаях целесообразно использовать в составе экспериментальных установок специальные программируемые устройства с микропроцессорами - микроконтроллеры. Подобные устройства значительно превосходят промышленные ПЭВМ по надежности в условиях эксплуатации вне лаборатории, компактности, но уступают им по вычислительной мощности и возможности использования с ними стандартных для ПЭВМ внешних устройств.

Как правило, специальные программируемые устройства содержат все основные узлы, входящие в состав обычных ПЭВМ: процессор, ОЗУ, ПЗУ и интерфейсы связи с некоторыми внешними устройствами. Особое место занимают специализированные процессоры - устройства, которые с максимально необходимой скоростью могут осуществлять предварительную обработку или отбор событий по заданному алгоритму. От универсальных процессоров спецпроцессоры отличаются минимизацией функциональных возможностей с целью обеспечения максимального быстродействия при обработке данных в конкретных физических измерениях - векторные и матричные процессоры, процессоры быстрого преобразования Фурье (БПФ) и другие.

Наибольшее распространение из средств, связывающих вычислительные устройства с измерительной аппаратурой, получили магистральные средства сопряжения - интерфейсы измерительных систем: приборный интерфейс (КОП, IEEE-488, HP-IB), интерфейс КАМАК, интерфейсы расширения ПЭВМ (EISA, PCI, PCMCIA, VXI/MXI), последовательные интерфейсы (RS232C).

LabVIEW предоставляет все необходимые средства, объединенные единой методологией, для программирования законченной системы. Среда LabVIEW предоставляет доступ к библиотекам виртуальных инструментов (VI) для управления и получения данных через интерфейс IEEE 488, шину VXI, RS–232 и встраиваемые платы

сбора данных. LabVIEW предлагает более 600 драйверов для приборов от более чем 50 мировых изготовителей измерительной аппаратуры. Таким образом, исключается необходимость программирования приборов на низком уровне. После сбора данных можно использовать библиотеку виртуальных инструментов (VI) анализа для получения из потока данных необходимого результата. Можно воспользоваться цифровой обработкой сигналов (DSP), цифровой фильтрацией, статистикой и численным анализом. Наконец, можно управлять системой с помощью собственной программы и визуализировать результаты, используя интерактивные лицевые панели. С помощью этих панелей создается стандартный, легко узнаваемый интерфейс независимо от аппаратного обеспечения системы. Кроме того, имеются широкие возможности по манипулированию данными — запись/чтение с диска, передача по сети и печать на принтере или плоттере.

Программирование осуществляется на уровне функциональных блок-диаграмм. Сочетание графического языка программирования и современного компилятора позволяет значительно сократить время разработки сложных систем при сохранении высокой скорости выполнения программ. Библиотеки современных алгоритмов обработки и анализа данных превращают LabVIEW в универсальный инструмент создания интегрированных систем на базе IBM PC совместимых компьютеров, Macintosh, рабочих станций SUN SparcStation и Hewlett Packard.

В LabVIEW вместо написания программы строятся виртуальные инструменты (VI). Легко создаваемая лицевая панель пользовательского интерфейса дает возможность интерактивного управления программной системой. Для описания функционирования системы строится блок-диаграмму, которая является привычным элементом для любой технической разработки. Но в LabVIEW блок-диаграмма является кроме всего исходным кодом программы. Таким образом, решается требующая немало времени и усилий при обычном подходе задача трансформации идеи разработчика в код программы. Виртуальные инструменты, с их графическим представлением, очень легко модифицируются, отлаживаются и полностью документированы.

Не менее важно, что созданные блоки можно встраивать как пиктограммы (subVI) в диаграммы верхнего уровня для построения сложных программных комплексов.

Для построения виртуального инструмента, в первую очередь, создается лицевая панель с необходимым набором кнопок, переключателей, регуляторов, экранов и т. п. Лицевая панель работает как интерактивный интерфейс ввода и вывода для измерительной системы или системы управления. В LabVIEW конструирование лицевой панели сводится к рисованию картинки, для чего предоставляются различные индикаторы и управляющие элементы. Остается только выбрать их из меню и расставить на панели. Кроме того, можно изменить цвет, размер, метку каждого элемента, его тип данных и диапазон значений. Можно импортировать любое изображение для создания специфического элемента задачи. Когда виртуальный инструмент будет закончен, можно использовать элементы лицевой панели для управления системой даже во время выполнения программы, меняя положение переключателей и регуляторов, поворачивая ручки управления и вводя значения с клавиатуры.

Блок-диаграммы LabVIEW

Создавая блок-диаграмму виртуального инструмента разработчик освобождается от многих синтаксических деталей обычного программирования:

- функциональные блоки выбираются из меню и соединяются с помощью проводников для обеспечения передачи данных от одного блока другому. Это могут быть как блоки элементарных алгебраических операций, так и сложные функции сбора и анализа данных, сетевые операции и файловый ввод/вывод, обмен данными с жестким диском в ASCII, бинарном формате и в формате табличного процессора;

- LabVIEW имеет обширный набор средств для разработки, тестирования и отладки разработанной системы. Окно подсказки (Help Window) описывает каждый блок и его соединения. LabVIEW немедленно проинформирует разработчика о неправильных соединениях и списке ошибок в окне Error Window. В ассортимент отладочных средств входят: подсветка выполнения блок-диаграммы, пошаговый режим, прерывания и индикация значений. Таким образом, можно производить трассировку и исследование выполнения программы непосредственно на блок-диаграмме.

Методы программирования

Порядок выполнения программы в LabVIEW устанавливается течением данных между блоками, а не последовательностью строк текста. Поэтому можно создавать диаграммы, которые имеют несколько параллельных потоков прохождения данных и несколько одновременно выполняемых операций.

В то время как потоки данных предпочтительны для параллельных операций, разработчик может задавать и специальный порядок выполнения. LabVIEW имеет такие программные структуры, как итерактивный цикл (FOR), последовательный цикл (WHILE) и оператор выбора (CASE), для последовательных, повторяющихся или разделяющих операций. Эти структуры представлены как графические рамки, окаймляющие управляемые блоки на блок-диаграмме.

LabVIEW является модульной средой по своей структуре. Любой VI может использоваться в блок-диаграмме другого виртуального инструмента как subVI. Разбив свою программную систему на subVI, разработчик может независимо разработать и интерактивно протестировать эти subVI, и тут же использовать их как узлы для построения виртуального инструмента более сложного уровня. Использование модульной иерархии позволяет эффективно разрабатывать, модифицировать, заменять и комбинировать виртуальные инструменты для удовлетворения изменяющихся требований конкретного приложения.

Возможности значительно расширяет иерархия VI. Создавая пиктограмму для собственного VI и используя ее в диаграмме другого виртуального инструмента, разработчик скрывает сложность низкоуровневой диаграммы, однако сохраняет доступ к общим переменным через панели нижнего уровня. Можно даже конфигурировать эти панели для автоматического открытия, создания анимаций и контекстно-зависимого интерфейса пользователя.

Во многих приложениях, скорость выполнения является критичной. LabVIEW - графическая среда программирования с компилятором, который генерирует оптимизированный код. Скорость выполнения LabVIEW близка к скорости выполнения компилированных Си программ.

Готовые виртуальные инструменты (VI) работают в системе разработчика LabVIEW, а также в LabVIEW Run-Time System. Это компактная, недорогая версия LabVIEW может только загружать и запускать VI, но не позволяет редактировать или показывать их диаграмму. Это свойство защищает исходный код VI.

С помощью дополнительной программы Application Builder выполняется преобразование VI в обычную исполняемую *.exe программу, которая запускается и выполняется самостоятельно, как любая Windows программа.

LabVIEW предоставляет доступ к стандартным библиотекам DLL MS Windows.

Вне зависимости от задачи, скорость выполнения программы является важнейшим фактором анализа данных. Библиотеки анализа LabVIEW используют максимум вычислительных возможностей вашего компьютера. Виртуальные инструменты оптимизированы для использования математического сопроцессора. Кроме того, существуют специализированные библиотеки, использующие вычислительные

возможности цифровых DSP процессоров, установленных на встраиваемых в компьютер платах.

Пример программирования в среде LabVIEW

В качестве примера организации ввода измерительной информации и последующей обработки представлен комплекс, состоящий из двух персональных компьютеров, соединенных между собой через последовательный интерфейс RS-232C. На одном компьютере программа LabVIEW имитирует генератор сигналов. На втором компьютере - программа LabVIEW имитирует управляющее устройство приема и обработки сигналов.

Порядок обмена информацией между генератором сигналов и управляющим устройством осуществляется специальным протоколом обмена – последовательностью и видом информационных посылок, которые производят управляющее устройство и генератор сигналов. Как правило, управляющее устройство имеет статус ведущего устройства. Т.е. управляющее устройство первым посылает запрос ведомому устройству (в нашем случае генератору сигналов), а последнее должно ответить на информационный запрос ведущего устройства, переслав ему ответную запрашиваемую информацию. В лабораторной работе обмен между управляющей программой и программой генератора сигналов осуществляется в пакетном режиме. Управляющая программа, работающая на первом персональном компьютере, посылает командный пакет, а программа генератора сигналов, работающая на втором персональном компьютере, отвечает на соответствующий запрос информационным пакетом. Формат командного пакета приведен в таблице 1. В лабораторной работе принят упрощенный вариант формата пакетов. В реальных устройствах длина командного и ответного пакетов может быть от нескольких байт до 255 байт.

Формат командного пакета Таблица 1.

1	2	3	4	5	6
Стартовый символ	Идент. номер	Код команды	Данные	Контрольная сумма пакета	Символ окончания пакета
0x10	ID	Code	Data	FCS	0x16

Стартовый символ пакета (поле 1 командного пакета) – определяет начало командного пакета в потоке данных. В лабораторной работе в качестве стартового символа используется 16-тиричный код (HEX код) 0x10. В реальных измерительных приборах чаще всего при обмене информацией используются только коды символов латинского алфавита и коды десятичных цифр. В этом случае, в качестве символа начала пакета может использоваться один из редко используемых символов (например, символы !, @, $, & и другие).

Идентификационный номер (ID) – условный уникальный номер, который присваивается устройству, работающему в данной системе. В лабораторной работе генератор сигналов имеет номер 01.

Код команды (Code) – условный номер, который имеет соответствующая команда, посылаемая управляемому устройству. Обычно число команд, которые может выполнять управляемое устройство, не превышает 128. В таблице 2 приведен перечень команд генератора сигналов, который реализован в лабораторной работе.

Перечень команд генератора сигналов Таблица 2.

№	Code	Наименование	Действие
1	00h	Random	Генерация случайного сигнала
2	01h	Harmonic	Генерация гармонического сигнала
3	02h	Randon & harmonic	Генерация суммы случайного и гармонического сигналов

В перечень команд обычно входят, как обязательные, следующие команды:

- нет операции (по этой команде управляемое устройство не выполняет никаких действий, но должно ответить управляющему устройству);
- идентификация устройства (по этой команде управляемое устройство должно возвратить идентификационную информацию о себе);
- состояние устройства (по этой команде управляемое устройство должно возвратить исчерпывающую информацию о своем состоянии).

Данные (Data) – информация, которую должно получить управляемое устройство от ведущего устройства. Для измерительных приборов это информация, которая определяет рабочие параметры управляемого устройства (например, диапазон частот и амплитуду сигнала генератора).

Контрольная сумма пакета (FCS) – код, который служит для проверки правильности приема информации ведомым устройством от ведущего устройства. При пакетном обмене чаще всего в качестве контрольной суммы пакета используется побайтная сумма по модулю два (логическое XOR) полей 1… 4 пакета. Кроме этого для контроля правильности приема информации используется циклическая контрольная сумма (CRC8 или CRC16), представляющая специальным образом вычисляемые полиномы из байтов пакета.

Символ окончания пакета – код, который является формальным признаком окончания командного пакета. Если в пакете передаются только коды символов, то в качестве окончания пакета выбираются символы перевода строки и возврата каретки.

В ответ на командный пакет от управляющего устройства ведомое устройство должно переслать ответный информационный пакет. Обычно структура ответного пакета повторяет структуру командного пакета и приведена в таблице 3.

Формат ответного пакета (информационного пакета) Таблица. 3

1	2	3	4	5	6
Стартовый символ	Идент. номер датчика	Код выполнения команды	Данные	Контрольная сумма пакета	Символ окончания пакета
0x10	ID	ResponseCode	Data	FCS	0x16

В ответном пакете вместо кода команды имеется код выполнения команды (ResponseCode), который формируется управляемым устройством в зависимости от его возможности выполнить полученную команду. В таблице 4 приведен примерный перечень кодов выполнения команды. Код выполнения команды позволяет управляющему устройству иметь информацию о работе управляемого устройства.

Перечень кодов выполнения команды Таблица 4.

№	ResponseCode	Результат выполнения команды
1	00h	Команда выполнена
2	02h	Ошибка в длине данных пакета
3	03h	Ошибка символа окончания пакета
4	04h	Принятый код FCS не совпадает с расчетным FCS
5	06h	Код команды не поддерживается устройством

В случае успешного выполнения команды в ответном пакете генератор сигналов передает двух байтное целое число (Data), соответствующее сгенерированному в данном запросе сигналу.

На Рис. 1 показано окно **Panel** передней панели программы Генератор.vi.

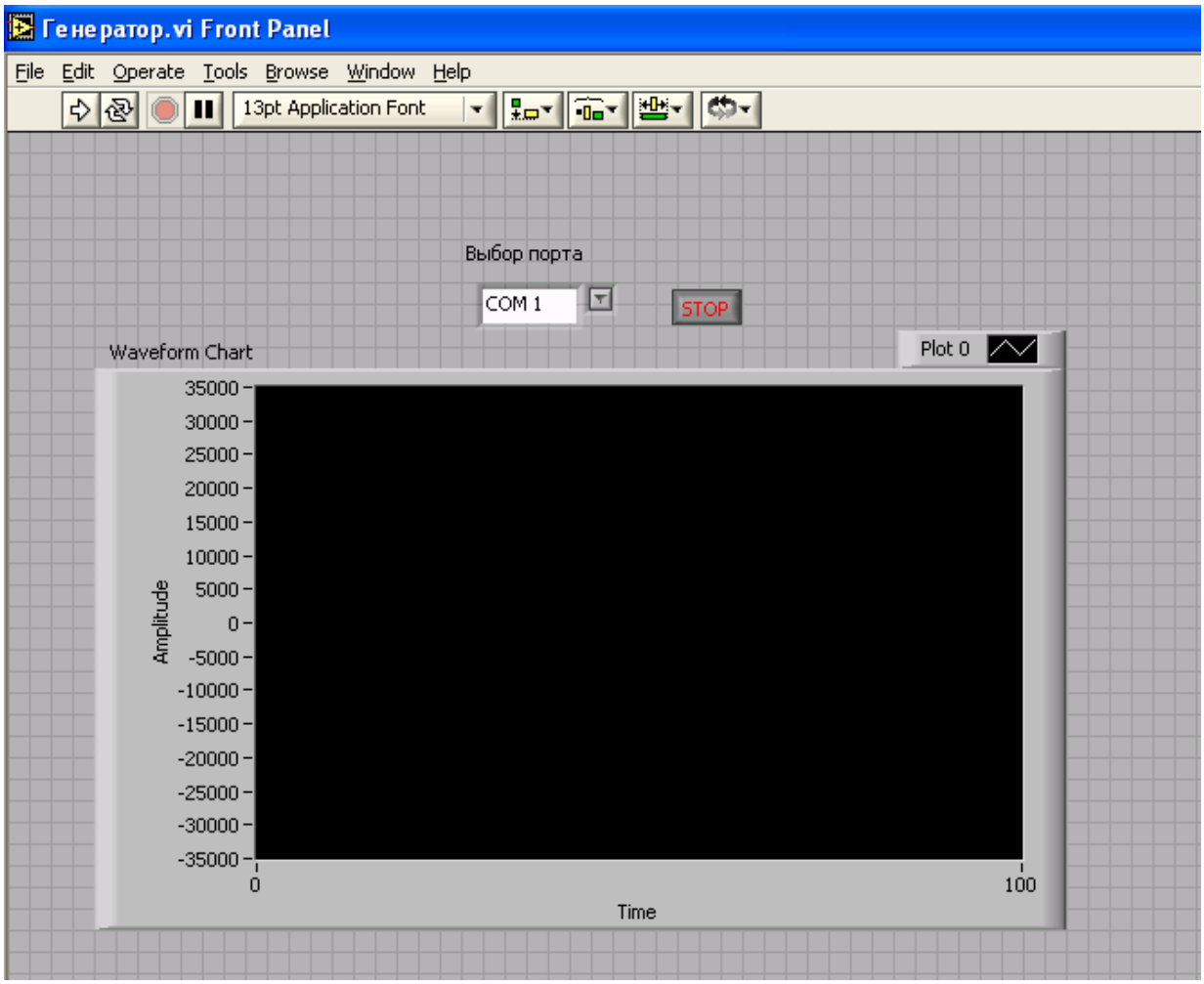

Рис. 1. Окно передней панели программы **Генератор.vi**.

На передней панели имеется управляющий элемент ***Выбор порта***, позволяющий выбрать порт для обмена информацией с управляющим устройством (COM1), кнопка остановки **STOP** и окно отображения передаваемой информации **Waveform Chart**.

На Рис. 2 показано окно **Block Diagram**, в котором осуществляется программирование алгоритма функционирования устройства.

Работа генератора начинается с определения настроек порта (функция VISA OPEN), через который осуществляется обмен данными с управляющим устройством. При настройке порта используются данные, вводимые с управляющего элемента ***Выбор порта,*** имеющегося на передней панели генератора. Другие настройки порта позволяют установить скорость обмена (9600 Бод), время ожидания (10000 мс), установить возможность прерывания при приеме заданного символа (**T**/F). К выходу блока открытия порта подключается функциональный блок определения ошибки. В случае возникновения ошибки при открытии порта этот блок вырабатывает логический сигнал True/False. Сигнал ошибки используется для определения дальнейших операций устройства. Далее генератор должен циклически принимать команду от управляющего устройства и посылать в ответ запрашиваемые данные. Однако при ошибке открытия порта дальнейшие

действия по приему и передачи данных являются невозможными. В алгоритме предусматривается пропуск этих действий и переход к выполнению функции закрытия порта (VISA Close), которая необходима для корректного завершения программы. Это состояние и показано на рис. 2.

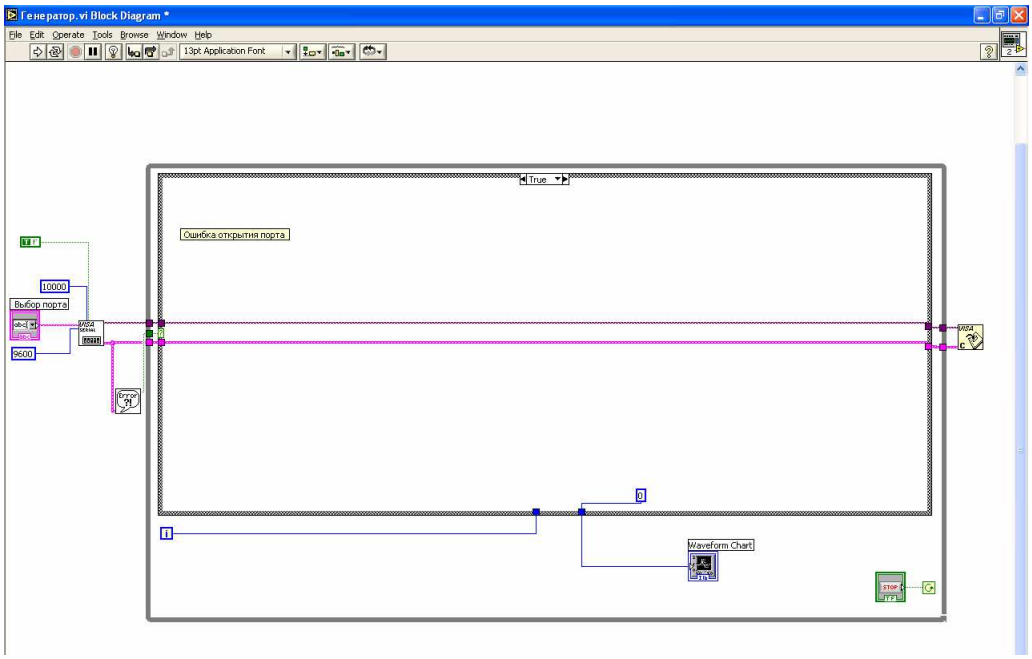

Рис. 2. Окно блок-диаграммы программы **Генератор.vi**. при наличии ошибки открытия порта

На рис. 3 …6 показано состояние диаграммы при отсутствии ошибки открытия порта. В цикле производится анализ последовательности символов, принятых коммуникационным портом. Процедура анализа последовательности данных оформлена в виде отдельной программы. Она будет рассмотрена далее. Цикл завершается в случае обнаружения ошибки работы порта или обнаружении командной последовательности. В случае появления ошибки генерация сигнала не производится и на графический индикатор выводится нулевое значение. При приеме командного пакета анализ номера команды осуществляется с помощью структуры выбора.

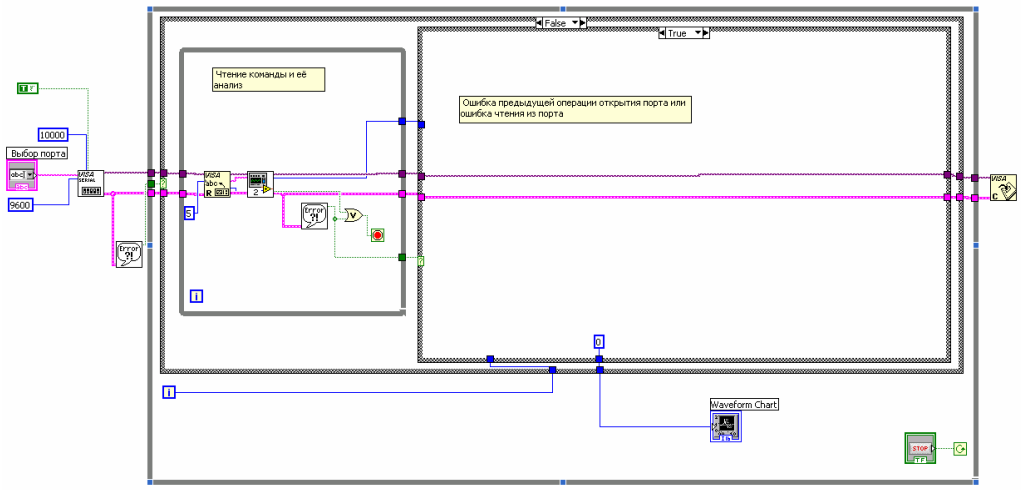

Рис. 3. Окно блок-диаграммы программы **Генератор.vi**. при отсутствии ошибки открытия порта

На рис. 4 показана диаграмма действий при приеме команды с номером 0. В этом случае производится формирование значения случайного сигнала, формирование пакета данных и запись его в коммуникационный порт. При формировании пакета данных осуществляется объединение стартового символа, номера устройства, кода ответа, символов полученных данных, подсчитанной контрольной суммы и символа окончания пакета.

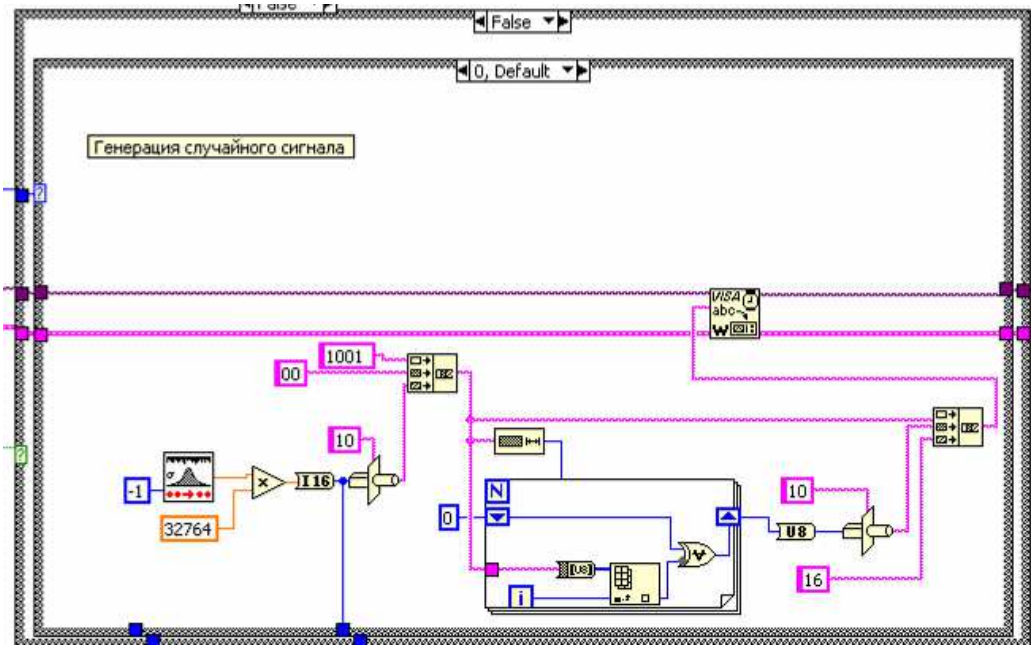

Рис. 4. Окно блок-диаграммы программы **Генератор.vi** при формировании случайного сигнала

На рис. 5 показана диаграмма действий при приеме команды с номером 1. В этом случае производится формирование синусоидального сигнала, формирование пакета данных и запись его в коммуникационный порт. Параметры синусоидального сигнала задаются соответствующими константами.

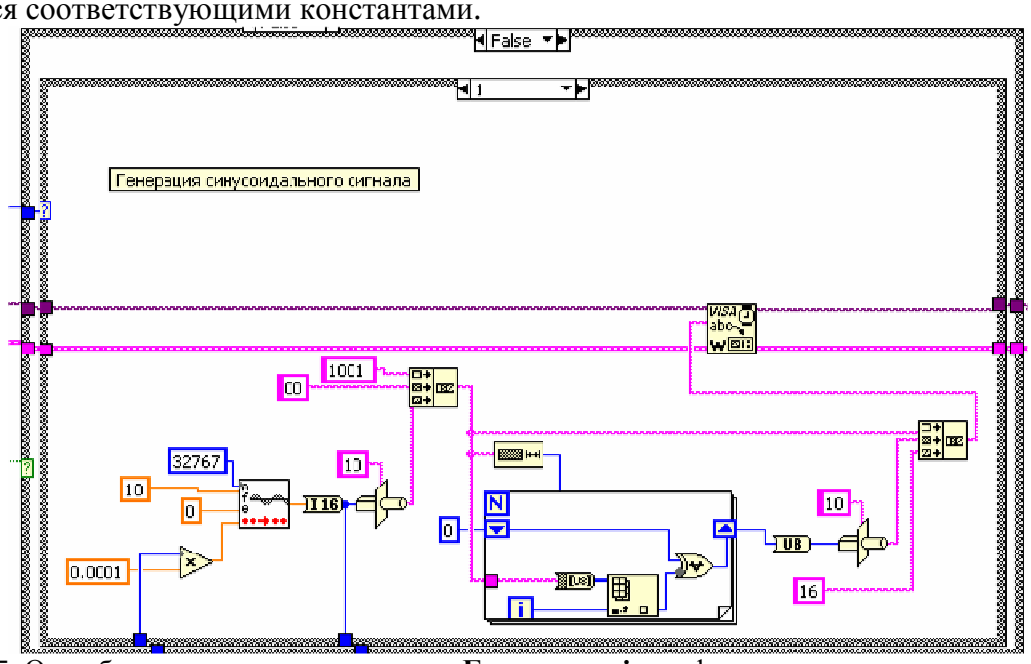

Рис. 5. Окно блок-диаграммы программы **Генератор.vi** при формировании синусоидального сигнала

На рис. 6 показана диаграмма действий при приеме команды с номером 2. В этом случае производится формирование суммы синусоидального и случайного сигналов. Параметры сигналов задаются константами.

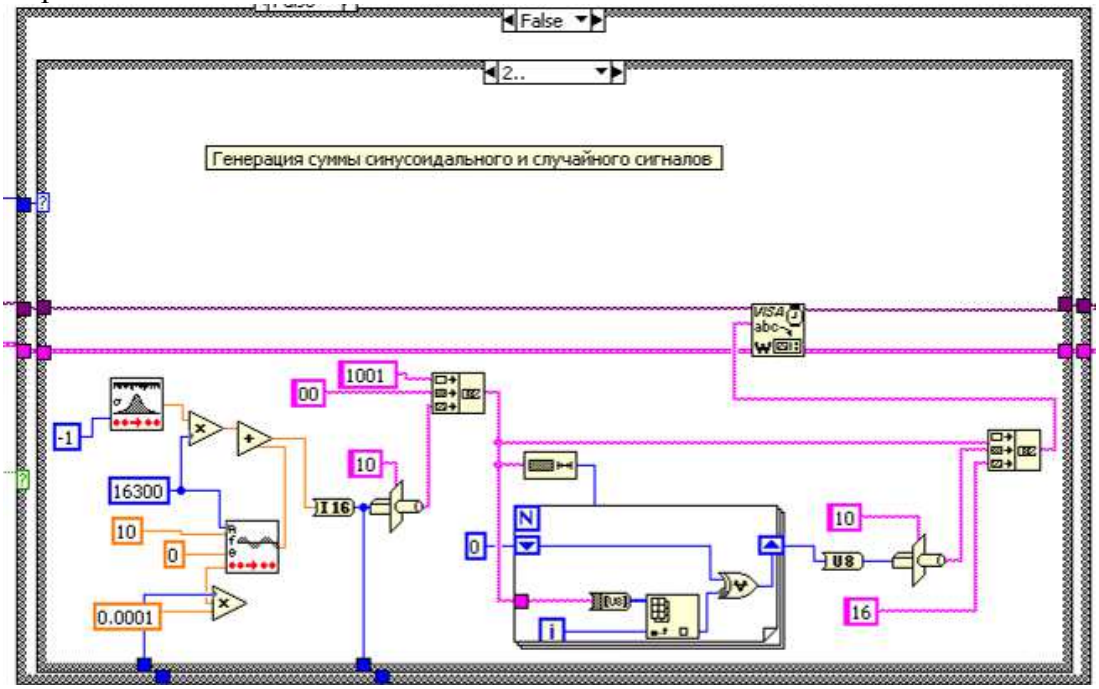

Рис. 6. Окно блок-диаграммы программы **Генератор.vi** при формировании суммы случайного и синусоидального сигналов

Во всех случаях для формирования сигналов используются стандартные функции генерирования сигналов. Для контроля результаты формирования различных сигналов подаются для визуализации на графический индикатор, который по точкам отображает результат работы устройства.

На рис. 7 показана диаграмма виртуального прибора анализа командного пакета данных. Входными параметрами являются количество принятых байт и последовательность принятых символов. Из последовательности символов выделяется команда, которая должна выполниться устройством и логическая переменная, принимающая значение false при отсутствии ошибки и true при ошибке выделения команды.

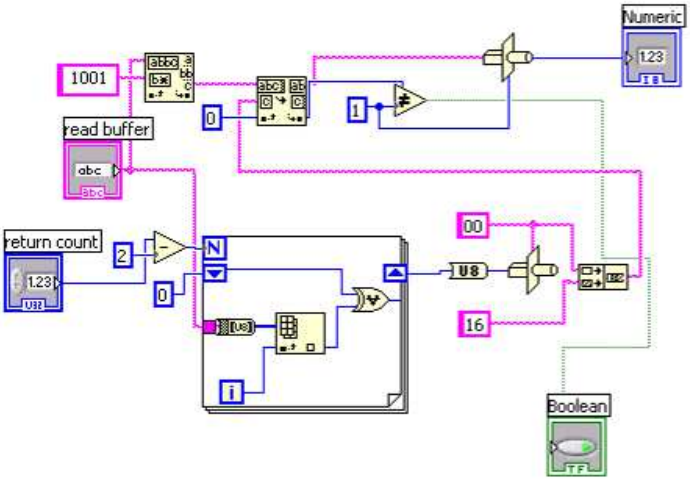

Рис. 7. Окно блок-диаграммы подпрограммы **Выделение данных.vi**.

Программа **Управляющее устройство.vi** управляет процессом генерирования сигнала программой **Генератор.vi**. На рис. 8 показана передняя панель устройства. На ней расположены управляющие элементы для выбора номера коммуникационного порта (COM1), номера команды и кнопка STOP. На панели расположен экран графического индикатора, который должен отображать числовые данные, принятые от управляемого генератора.

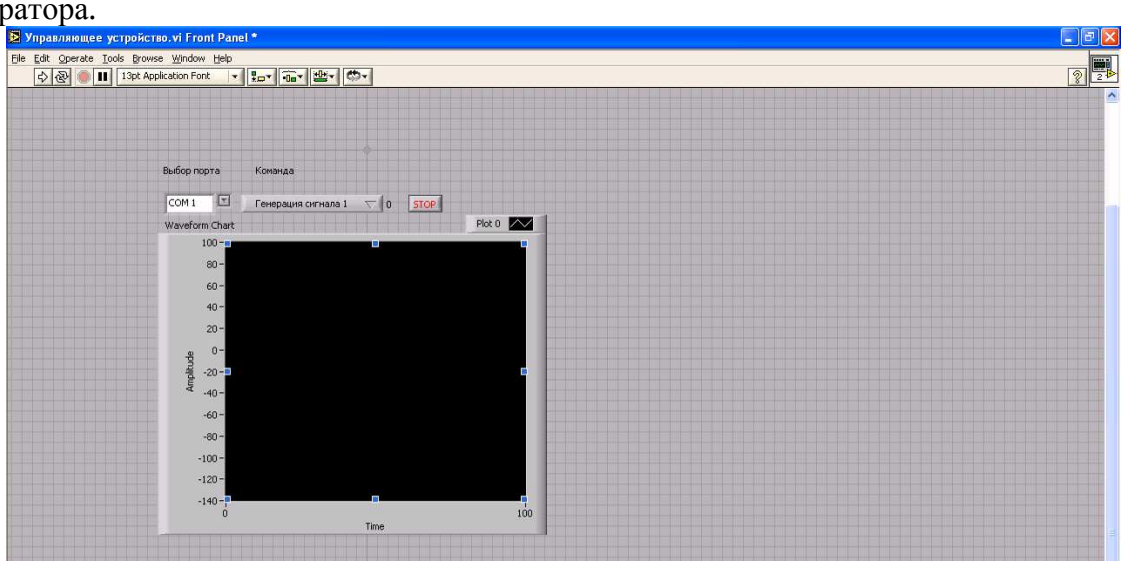

Рис. 8. Окно передней панели программы **Управляющее устройство.vi**.

На рис. 9 …12 показана блок-диаграмма управляющего устройства. Работа управляющего устройства начинается с открытия порта. Далее последовательно производится формирование команды для управления генератором и ожидание ответа от управляемого генератора. При приеме ответного пакета данных осуществляется его декодирование, а выделенные данные подаются для визуализации на графический индикатор. В отличие от программы генератора в данной программе используется структура последовательности.

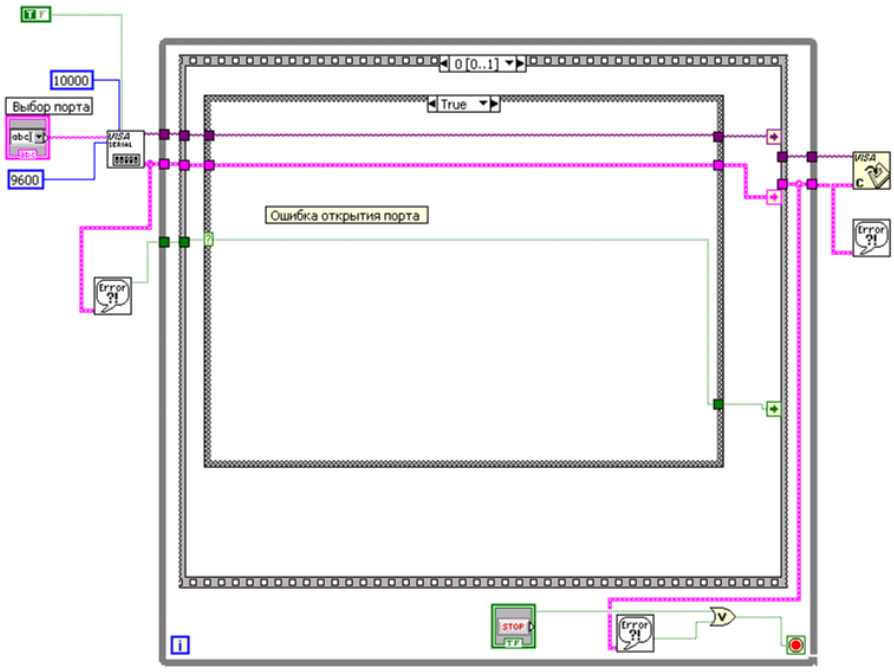

Рис. 9. Окно блок-диаграммы подпрограммы **Управляющее устройство.vi** (кадр 0 последовательности) при ошибке открытия порта

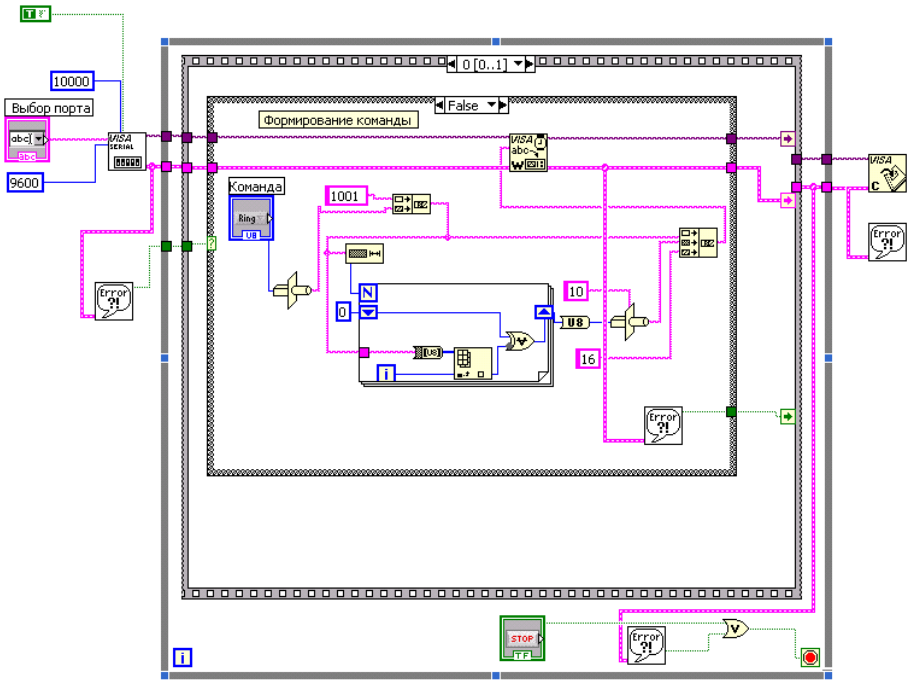

Рис. 10. Окно блок-диаграммы подпрограммы **Управляющее устройство.vi** (кадр 0 последовательности) при отсутствии ошибки открытия порта

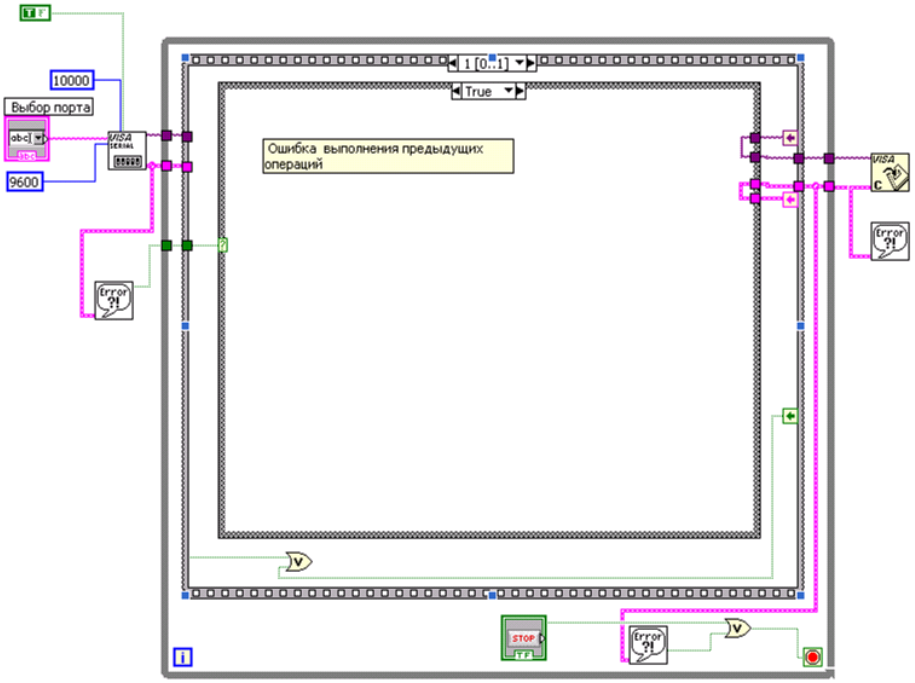

Рис. 11. Окно блок-диаграммы подпрограммы **Управляющее устройство.vi** (кадр 1 последовательности) при наличии ошибки в предыдущих действиях

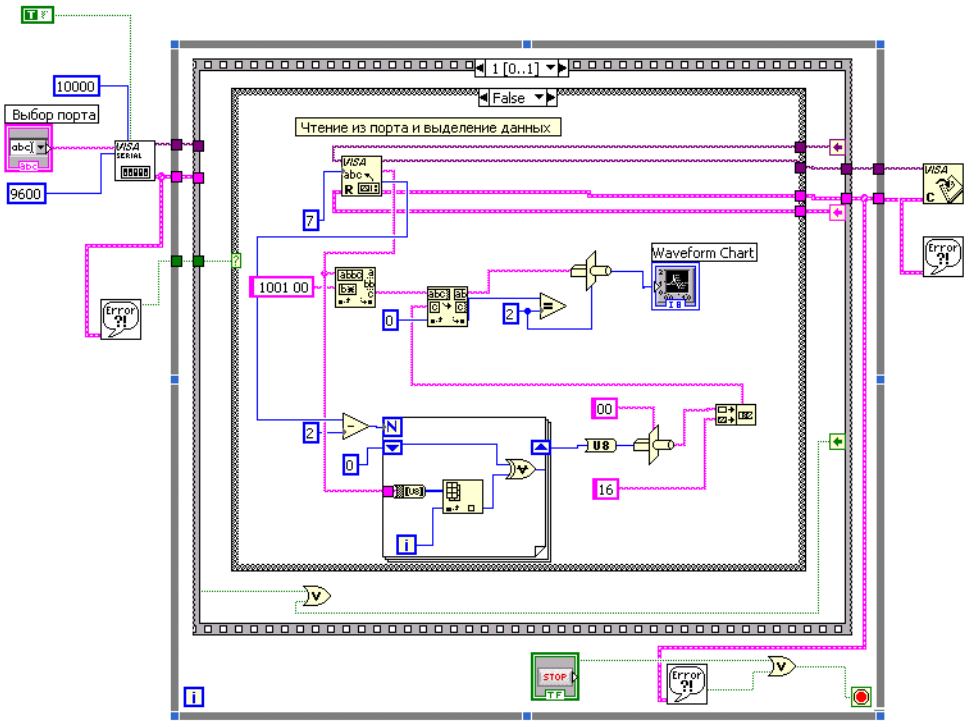

Рис. 12. Окно блок-диаграммы подпрограммы **Управляющее устройство.vi** (кадр 1 последовательности) при отсутствии ошибки в предыдущих действиях

В кадре 0 последовательности (рис. 9 и 10) производится формирование команды и её запись в коммуникационный порт. В кадре 1 последовательности (рис. 11 и 12) производится декодирование принятого ответного пакета и выделение числовых данных, сформированных генератором.

Порядок выполнения работы.

1. Запускается программа LabVIEW соответствующей иконкой в среде Windows.
2. С помощью пункта головного меню File Open загрузить файл программы генератора сигналов (Генератор.vi) или файл программы управляющего устройства (Управляющее устройство.vi).
3. Запустить соответствующую программу на выполнение и наблюдать формирование сигналов в окнах графических индикаторов соответствующих программ. Программа генератора должна запускаться на выполнение первой, так как переходит в режим ожидания команды от управляющего устройства. Максимальное время ожидания равно 10 секундам.
4. В режиме манипулирования элементами управления выбрать вид сигнала, который формируется генератором сигналов. Наблюдать изменения на графических индикаторах программ.
5. Остановите работу программы и разберитесь в этапах её работы.
6. Включите в схему генератора и управляющего устройства дополнительные блоки, обеспечивающие:

- формирования массивов данных с заданным числом измерений N;

- вывода на графический индикатор только всего массива измерений из N точек;

- анализа параметров сигнала (например, спектрального анализа, измерения среднеквадратического значения и др. по указанию преподавателя);

- записи результатов расчетов параметров сигнала в текстовый файл, доступный для просмотра программой текстового редактора;

- анализ ошибок, возникающих при неправильной последовательности запуска программ генератора сигналов и управляющего устройства.

7. Записать модифицированный файл с расширением *.VI в рабочую директорию.

8. Убедитесь в совместной работе программ генерации сигнала и управляющего устройства. Проверьте, как изменится работа программ при изменении параметров портов, через которые происходит обмен данными.

9. Формы отображаемых графиков на графических индикаторах в программах генератора и управляющего устройства должны быть одинаковыми.

10. По указанию преподавателя задайте требуемые параметры генерируемых сигналов и отметьте соответствующие изменения в результатах работы программы.

Внимание!

1. Перед началом выполнения работы создайте свой подкаталог.

2. Файл с выполненными заданиями является вашим отчетом.

3. Лицевая панель созданной вами виртуальной установки должна быть подписана (фамилия, номер группы) в режиме текстового редактирования лицевой панели.

4. Скопируйте исходный файл в свой подкаталог**. Редактируйте только копию исходного файла.**