ФЕДЕРАЛЬНОЕ АГЕНТСТВО ПО ОБРАЗОВАНИЮ
ОРЛОВСКИЙ ГОСУДАРСТВЕННЫЙ
ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
ФАКУЛЬТЕТ ЭЛЕКТРОНИКИ
И ПРИБОРОСТРОЕНИЯ

Кафедра «Прикладная математика и информатика»

О.В. Пилипенко, Н.Б. Горбачев, М.А. Музалевская

ОСНОВЫ ПРОГРАММИРОВАНИЯ, МАТЕМАТИЧЕСКОГО МОДЕЛИРОВАНИЯ И ОБРАБОТКИ ДАННЫХ В СРЕДЕ LABVIEW

Практикум

Дисциплина – «Информатика»
Специальности: 150301 «Динамика и прочность машин»
150204 «Машины и технологии литейного производства»
150201 «Машины и технологии обработки металлов давлением»
140610 «Электрооборудование и электрохозяйство предприятий, организаций и учреждений»
220301 «Автоматизация технологических процессов и производств»
220200.62 «Автоматизация и управление»
150400.62 «Технологические машины и оборудование»
150600.62 «Материаловедение и технология новых материалов»

Печатается по решению редакционно-издательского совета ОрелГТУ

Орел 2008

Авторы: канд. техн. наук, доц. каф. ПМиИ О.В. Пилипенко
канд. техн. наук, доц. каф. ПМиИ Н.Б. Горбачев
канд. эконом. наук, доц. каф. ПМиИ М.А. Музалевская

Рецензент: д-р техн. наук, проф. каф. ПМиИ А.А. Батенков

В методических указаниях рассмотрены основы работы в среде LabVIEW. Они отражают основные направления совершенствования современных методов обучения - использование высоких информационных технологий для моделирования изучаемых законов и явлений. На основе практических заданий к лабораторным работам и требованиям по их выполнению и оформлению студенты закрепляют теоретические знания по демонстрации структур и основным принципам разработки и отладки компьютерных программ, созданию инженерного интерфейса и методов обработки данных.

Предназначены для изучения дисциплины «Информатика» студентами очной формы обучения специальностей 150301 «Динамика и прочность машин», 150204 «Машины и технологии литейного производства», 150201 «Машины и технологии обработки металлов давлением», 140610 «Электрооборудование и электрохозяйство предприятий, организаций и учреждений», 220301 «Автоматизация технологических процессов и производств», 220200.62 «Автоматизация и управление», 150400.62 «Технологические машины и оборудование», 150600.62 «Материаловедение и технология новых материалов».

Редактор Ю.А.Демина
Технический редактор В.В.Лях

Орловский государственный технический университет
Лицензия ИД №00670 от 05.01.2000 г.

Подписано к печати 30.09.2008 г. Формат 60x84 1/16.
Печать офсетная. Уч.-изд. л. 5,0. Усл. печ. л. 4,0. Тираж 15 экз.
Заказ №_____

Отпечатано с готового оригинал-макета
в ООО «СтройИндустрияИнвест»,
302020, г. Орел, Наугорское шоссе, 29.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Практикум по основам программирования в среде LabVIEW предназначен для студентов 1-2 курсов, обучающихся по инженерным специальностям. Он предваряет внедрение современных методов изучения естественнонаучных и инженерных дисциплин, связанное с использованием высоких информационных технологий для визуализации изучаемых законов и явлений. При этом учебные лабораторные и демонстрационные работы выполняются как многофакторные исследования, осуществляются в динамике и дополняются элементами математического моделирования.

На практике такая постановка требует одновременного и непрерывного измерения различных физических и технологических параметров, таких как перемещение, скорость движения, температура, статическое и динамическое давление, расходы и уровни жидкости, локальные скорости воздушного потока и т.п. Одним из лучших инструментов для реализации этих задач являются среда графического программирования LabVIEW и технические средства компании National Instruments (США) - мирового лидера в области автоматизации измерений и управления технологическими процессами. Основной идеей построения автоматизированных систем сбора, обработки и визуализации экспериментальных данных в среде LabVIEW служит возможность модификации обычного персонального компьютера до уровня многоканальной информационно - измерительной системы с высокими метрологическими характеристиками (рисунок 1).

Первый уровень платформы технических и программных средств, используемых для автоматизации лабораторного оборудования - это объекты исследования, то есть отдельные лабораторные установки, оснащенные измерительными датчиками - преобразователями физических величин в электрические сигналы.

Следующий уровень - платы автоматизированного сбора данных, преобразующие аналоговые сигналы датчиков в цифровой код; компьютер и измерительные сервисы. Последние представляют собой программные модули, обеспечивающие осуществление измерений доступным для пользователя способом. В их состав входят драйверы измерительных и вспомогательных приборов и инструменты для их калибровки.

Рисунок 1 - Платформа технических и программных средств, использованная для автоматизации измерений и обработки данных

Третий уровень - представляет программная среда, в которой создаются и работают модули, автоматизирующие процессы измерений и обработки данных.

Необработанные данные редко содержат полезную информацию. Вначале они должны быть преобразованы к виду, удобному для анализа. Для этого необходимо убрать шумовые искажения, скорректировать аппаратные ошибки, компенсировать возмущающие воздействия. Затем должны быть разработаны специальные программы управления экспериментом и управления полученными информационными массивами, определить способ представления данных для каждой лабораторной работы. Это четвертый уровень платформы.

При поочередном подключении автоматизированные лабораторные установки обслуживаются одним компьютером. Измерительная система, используемая в настоящем практикуме, позволяет одновременно измерять 8 различных параметров и осуществлять управление экспериментом по 2 каналам. Электрические сигналы с датчиков подаются на 12 разрядный аналого-цифровой преобразователь с USB выходом. Плата сбора данных преобразует токи и измеряемые напряжения в цифровой код с точностью до 0,5 % от действующего значения.

Полученный цифровой сигнал обрабатывается в среде LabVIEW. Результаты измерений выводятся на лицевую панель монитора компьютера или мультимедийный экран в виде показаний обычных стрелочных или цифровых приборов, осциллографов и самописцев. Непосредственно во время эксперимента строятся графики изменения параметров во времени, а по его окончанию их зависимости друг от друга. При этом уникальное лабораторное оборудование, имеющееся в вузах, может быть коренным образом модернизировано и приведено в соответствие с современным уровнем образовательных технологий.

Благодаря возможности визуального наблюдения процессов, в том числе и скрытых от непосредственного наблюдения, значительно повышается информативность выполняемых лабораторных работ. Это позволяет отказаться от использования традиционной измерительной техники, при которой, по-прежнему, остаются рутинные операции считывания результатов измерений, преобразования их в цифровые величины, ввода полученных массивов в стандартные программы статистической обработки и т.д. В них обычно теряется часть полезной информации, появляются дополнительные погрешности, непродуктивно используется время занятий, многие эксперименты вообще неосуществимы.

В процессе обучения на компьютеризированном лабораторном оборудовании возникают дополнительно положительные моменты, которые в большей мере используются уже в курсе «Контроль и регулирование технологических процессов с применением ЭВМ». Это демонстрация современных методов мониторинга и управления производством, составления баз данных, документирования отчетности, анализа аварийных ситуаций и т.д.

В заключение следует отметить, что процесс обучения связан в первую очередь с развитием способностей студентов самостоятельно познавать новые сложные явления и использовать эти знания на производстве. Этому в значительной мере способствует совершенствование лабораторной базы и методик лабораторного эксперимента. Современные компьютерные технологии позволяют решать эту задачу наиболее эффективным образом.

Создание, редактирование и отладка программ в среде LabVIEW

1. Создание, копирование и удаление объектов

В LabVIEW предусмотрена возможность создания элементов управления и отображения данных, констант по щелчку правой кнопкой мыши (ПКМ) на узле или предполагаемом месте их расположения. Для этого в контекстном меню следует выбрать пункт Create → Constant для создания констант, отображающихся только на блок-диаграмме; Control -для создания элемента управления на лицевой панели и блок-диаграмме; Indicator - для создания элемента отображения данных на блок-диаграмме и лицевой панели.

Объекты можно копировать, перемещая выделенный объект и одновременно удерживая клавишу Ctrl. После переноса выбранного объекта на новое место отпускается сначала кнопка мыши, а затем клавиша Ctrl. В этом месте появляется копия объекта, а первоначальный объект остается на старом месте. Можно копировать объекты стандартным способом, выбирая пункты главного меню Edit→ Copy и затем Edit→ Paste.

Для удаления объекта следует выделить его с помощью инструмента активизации объекта (палитра Tools) затем нажать на клавиатуре клавишу Delete или выбрать пункты главного меню Edit→ Clear.

2. Отмена и восстановление действий

В процессе редактирования виртуального прибора может быть допущена ошибка. Можно отменить и восстановить действия, выбрав Undo (Отменить) или Redo (Восстановить) в пункте главного меню Edit (Редактирование). Установка количества действий, подлежащих отмене или восстановлению, производится в пункте главного меню Tools→ Options. Для этого из выпадающего меню следует выбрать раздел Block Diagram. Установка небольшого числа повторений сохраняет ресурсы памяти компьютера.

3. Идентификация объектов

Для идентификации объектов используются метки. Среда LabVIEW имеет два вида меток - свободные и собственные. Собственные метки принадлежат объекту, описывают только его и двигаются вместе с ним. Собственную метку можно перемещать независимо от объекта, но при перемещении самого объекта его собственная метка пе-

ремещается вместе с ним. Свободные метки не принадлежат объектам. Их можно создавать, перемещать, вращать или удалять независимо. Они используются для описания объектов на русском языке, ввода комментариев на лицевой панели и блок-диаграмме.

Для создания свободной метки используется инструмент «Редактирование» на палитре Tools. Выбрав его, необходимо щелкнуть на свободном пространстве одной панели и ввести текст. После ввода текста метки поместить курсор в пространство вне метки или нажать кнопку Enter на инструментальной панели.

4. Выделение и удаление проводников данных

Сегмент проводника данных - это отдельная его горизонтальная или вертикальная часть. Место соединения двух сегментов - излом проводника данных. Точка, в которой встречаются два, три или четыре проводника данных, называется точкой соединения. Проводник данных содержит все сегменты между точками соединения и между терминалами данных, если нет точек соединений. Для выделения сегмента используется инструмент активизации на палитре Tools. Один щелчок мыши по выбранному сегменту проводника выделяет этот сегмент, двойной выделяет излом проводника данных, тройной щелчок – все проводники, соединяющиеся с выделенным сегментом. Разорванный проводник данных выглядит как черная штриховая линия с красным крестом посередине. Разрыв образуется при попытке соединения объектов с несовместимыми с ними типами данных. Описание причины разрыва проводника данных появляется в окне всплывающей подсказки после наведения на проводник инструмента «Соединение». Удаление всех разорванных проводников производится через пункт главного меню Edit→ Remove Broken Wires.

5. Редактирование текста

Выбрав пункт меню Text Setting на инструментальной панели, можно изменить шрифт, стиль, размер и провести выравнивание любого текста внутри меток или на дисплеях элементов управления и отображения.

6. Изменение размеров объектов

Большинство объектов лицевой панели допускают изменение размеров. Чтобы подготовить объект к изменению размера, необходимо навести на него инструмент активизации. При этом по углам

объекта появляются маркеры. Следует установить курсор на один из маркеров и, удерживая нажатой левую кнопку мыши, переместить маркер. Размер шрифта при этом не меняется. Промежуточные границы изменяемого размера обозначаются штриховой линией. Когда нужный размер элемента достигнут, кнопку мыши следует отпустить. Удержание клавиши Shift во время перемещения маркеров сохраняет пропорции объекта.

Можно изменять размеры и объектов блок-диаграммы, таких как структуры и константы.

ЛАБОРАТОРНАЯ РАБОТА № 1

ОСНОВЫ ПРОГРАММИРОВАНИЯ В СРЕДЕ LABVIEW

Цель работы:

- ознакомление с организацией программной среды LabVIEW: изучение компонент диалогового окна LabVIEW, лицевой панели и блок-диаграммы, изучение палитры инструментов (Tools Palette), палитр элементов контроля (Controls Palette) и функций (Function Palette);

- приобретение практических навыков создания, редактирования и отладки компьютерных приборов.

Задание 1. Создать и запустить программу генератора случайных чисел и запустить режим анимации потоков данных.

Задание 2. Найти в библиотеке примеров LabVIEW программу исследования функций, построение графиков самой функции и ее производной, нахождение их нулей и экстремальных значений.

Общие положения

Для создания собственных программ в среде LabVIEW используются следующие инструменты: Лицевая панель, Блок-диаграмма, палитры элементов управления и отображения данных и палитры функций. При запуске LabVIEW из меню стартового диалогового окна командами New →Blank VI открываются два окна - Лицевая панель и Блок-диаграмма (рисунки 1.1 и 1.2).

В правом верхнем углу каждого окна находится пиктограмма для архивирования созданной программы в качестве нового компьютерного прибора. Здесь же размещена традиционная для приложений Windows полоса главного меню с одинаковыми для обоих окон пунктами: File, Edit, Operate, Tools, Browse, Windows, Help. Краткое описание функций пунктов главного меню приведено в таблице 1.1.

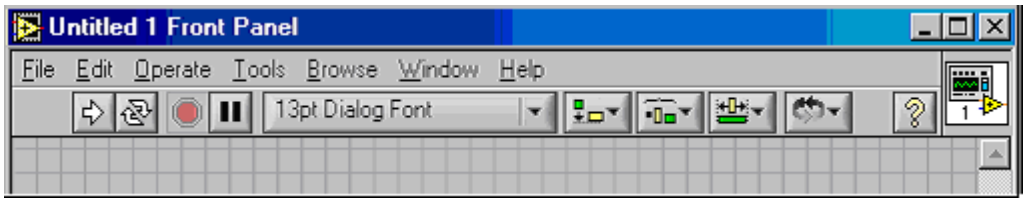


Рисунок 1.1 - Лицевая панель

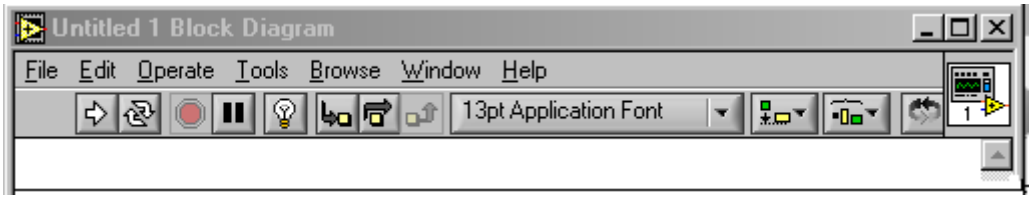


Рисунок 1.2 - Панель блок-диаграмм

Таблица 1.1 - Краткое описание функций главного меню

Пункты меню	Перевод	Функции
File	Файл	Открытие, закрытие, сохранение и печать программ
Edit	Правка	Редактирование панелей, поиск объектов
Operate	Управление	Запуск и прерывание выполнения программ
Tools	Инструменты	Управление библиотеками программ
Browse	Просмотр	Просмотр иерархий программ
Windows	Окно	Отображение окон и палитр LabVIEW
Help	Справка	Дополнительная информация об элементах и функциях LabVIEW

Ниже полос главного меню расположены линейки инструментов, которые различны для Лицевой панели и Блок-диаграммы за счет дополнительных кнопок для отладки программ (таблица 1.2).

Таблица 1.2 - Назначение кнопок инструментальных панелей

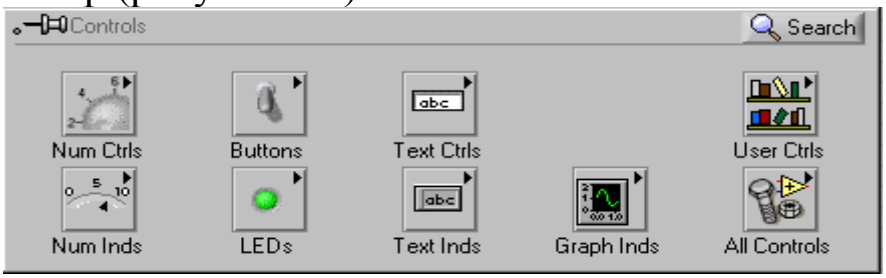
Пиктограмма	Назначение кнопок инструментальных панелей
	Кнопка **Запуск** (Run) при правильно составленной программе
	Вид кнопки **Запуск** (Run) при наличии ошибок в программе
	Вид кнопки **Запуск** (Run) в процессе выполнения программы
	Вид кнопки **Запуск** (Run) в процессе выполнения подпрограммы
	Кнопка **Непрерывный (повторяющийся) Запуск** (Run Continuosly)
	Кнопка Останова выполнения программы (Abort Execution)

Продолжение таблицы 1.2

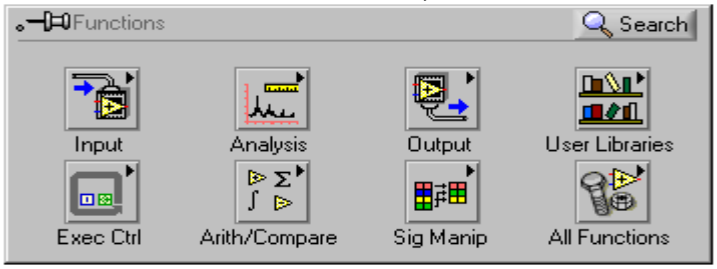
Пикто-грамма	Назначение кнопок инструментальных панелей
	Кнопка временной паузы выполнения программы (Pause)
	Анимация потоков данных при отладке программ
	Начало пошагового выполнения отладки программ
	Пошаговое выполнение
	Выход из пошагового выполнения программ
	Редактирование текста (шрифт, размер, стиль и текст)

Свободное пространство каждой панели образует рабочую область, снабженную горизонтальной и вертикальной полосами прокрутки. При разработке программ в рабочей области Лицевой панели размещаются визуальные элементы управления и индикации, формирующие интерфейс пользователя, а на панели Блок-диаграммы составляется графический код создаваемого приложения. Для одновременного отображения на экране монитора обоих панелей целесообразно использовать команду: Windows→Tile Left and Right.

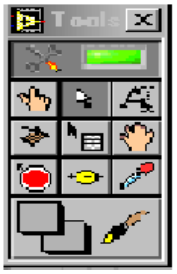
Разработка программ осуществляется с помощью трех вспомогательных палитр (рисунок 1.3):



а)



б)



в)

Рисунок 1.3 - Вспомогательные палитры: а) палитра элементов контроля и индикации, б) палитра функций, в) палитра инструментов

- Палитры элементов управления и индикации (Controls Palette) на Лицевой панели;

- Палитры функций (Functions Palette) на Блок-диаграмме;

- Палитры инструментов (Tools Palette), доступной на обеих панелях.

Инструменты имеют следующее назначение:

-  - инструмент УПРАВЛЕНИЕ - для изменения значения элементов управления или ввода текста;

-  - ПЕРЕМЕЩЕНИЕ - для активизации, перемещения или изменения размеров объектов;

-  - ВВОД ТЕКСТА – для редактирования текста и создания свободных меток;

-  - СОЕДИНЕНИЕ - создает проводники данных, соединяя объекты на блок-диаграмме;

-  - ВЫЗОВ КОНТЕКСТНОГО МЕНЮ - вызывает контекстное меню соответствующего объекта по щелчку левой кнопки мыши.

-  - БЫСТРАЯ ПРОКРУТКА ЭКРАНА – для просмотра окна без использования полосы прокрутки;

-  - ВВОД КОНТРОЛЬНОЙ ТОЧКИ - позволяет расставлять контрольные точки в функциях, узлах, проводниках данных, структурах и приостанавливать в них выполнение программы;

-  - УСТАНОВКА ОТЛАДОЧНЫХ ИНДИКАТОРОВ – показывает текущее значение переменных в проводниках блок-диаграммы, используется при отладке программ для просмотра промежуточных значений;

-  - КОПИРОВАНИЕ ЦВЕТА - предназначен для копирования и последующей вставки цвета;

-  - РАСКРАШИВАНИЕ - позволяет изменить цвет объекта и отображает текущий фон.

Типы и проводники данных

В среде LabVIEW используются различные типы данных (рисунок 1.4, таблица 1.3).

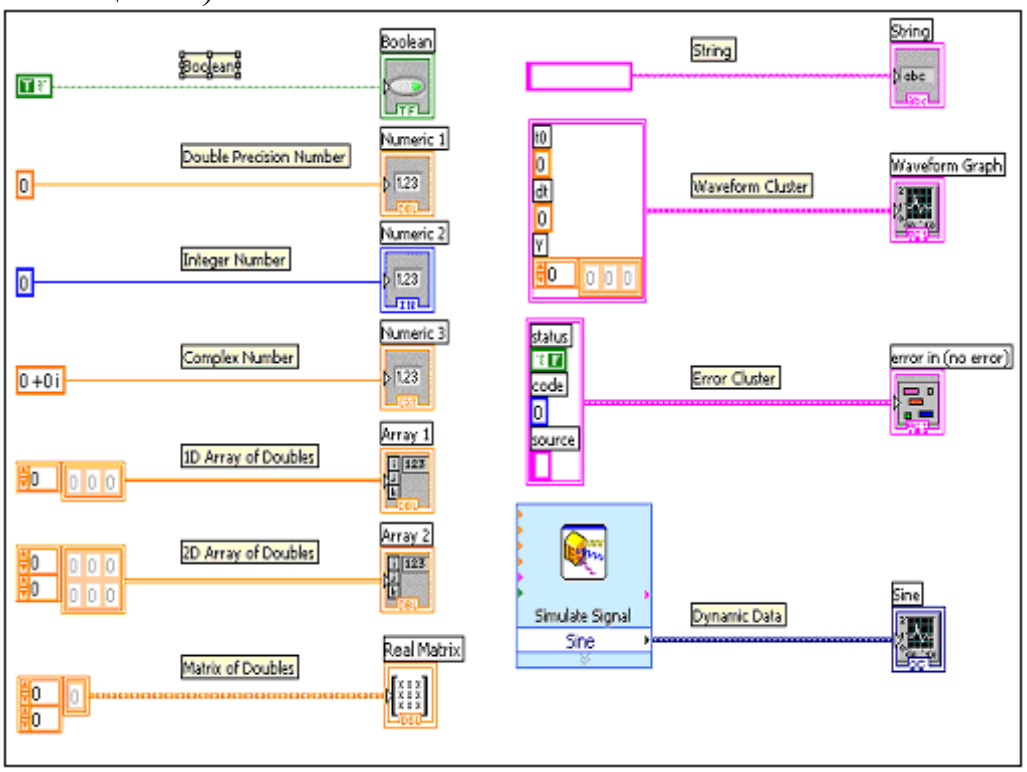


Рисунок 1.4 - Типы данных в LabVIEW

Таблица 1.3 - Типы данных в LabVIEW

Тип данных	Цвет	Значение по умолчанию
логический	зеленый	ложь
число с плавающей запятой	оранжевый	0,0
комплексное число	оранжевый	0,0+i0,0
целое число	синий	0
строка	розовый	пустая
кластер (включает разные типы данных)	розовый	-
динамический (информация о сигнале - имя, дата и время получения данных)	фиолетовый	-
массив (включает тип данных в скобки и принимает цвет данных этого типа)	различный	-

Для организации повторяющихся вычислений используются структуры цикла с заданным числом итераций и цикла, прекращаю-

щего свою работу при наступлении того или иного события в тех случаях, когда число итераций заранее не известно (рисунок 1.5).

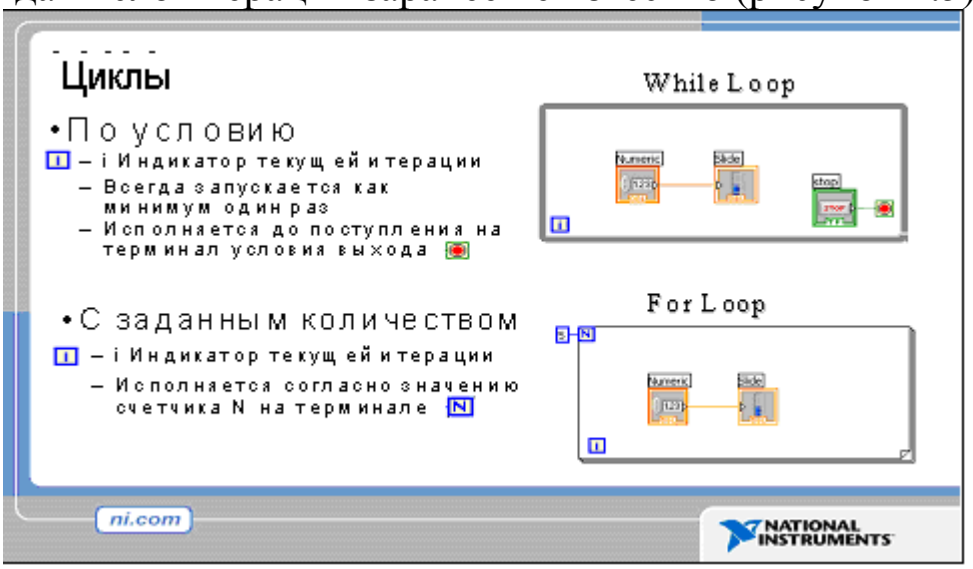


Рисунок 1.5 - Структуры цикла в LabVIEW

Порядок выполнения задания 1

1.1 Осуществите запуск среды LabVIEW из каталога D:\LABV. В появившемся главном окне программы выберите команды: *New→Blank VI* для создания нового файла. Далее выберите меню: *Window→The Left and Right* для одновременного отображения на экране двух окон программы: серой и белой панелей. Серая *Лицевая панель* (обычно располагается слева) – инструмент пользователя, который предназначен для размещения элементов ввода и вывода данных в виде привычных технических устройств, таких как: цифровые указатели, ползунковые реостаты, регуляторы громкости, осциллографы, самописцы, графопостроители и т.д.. Белая (обычно располагается справа) - *Блок-диаграмма*, на которой вызываются пиктограммы различных функций и структур и составляется графический код программы.

Для совершения различных операций с помощью курсора необходимо вызвать *Палитру инструментов* с помощью меню: *Window→Show Tools Palette* на *Лицевой панели* или на *Блок-диаграмм.*

1.2 Щелчком ПКМ на Лицевой панели вызываем палитру элементов контроля и управления и закрепляем ее в стационарном положении с помощью инструмента «кнопка» в левом верхнем углу палитры. В ней активизируем элементы контроля – первая пиктограмма

в первом ряду – для задания исходных параметров. Выделяем курсором поочередно «цифровой регулятор», «реостат», «ручку регулятора громкости» и переносим их на верхнюю часть Лицевой панели.

Создадим пять элементов индикации работы этих приборов: «стрелочный амперметр», «манометр», «термометр», «линейный индикатор» и «осциллограф». Для этого активизируем пиктограмму «элементы индикации», выбираем в ней соответствующие приборы и переносим их на свободную часть Лицевой панели. Обратим внимание, что при появлении любого нового элемента на Лицевой панели одновременно появляется его модифицированное изображение на блок-диаграмме. Дальнейшее программирование в среде LabVIEW практически сводится к соединению элементов блок-диаграммы проводниками данных. При этом вид проводника автоматически выбирается соответствующим типу данных (рисунок 1.4).

1.3 Для работы с блок-диаграммой нужны дополнительные инструменты, которые вызываются из главного меню как Палитра инструментов (Tools Palette), доступная на обоих панелях - *Window→ Show Tools Palette.*

1.4 Подадим выходные сигналы управляющих элементов на входы произвольных индикаторов, соединяя их проводниками данных с помощью инструмента «катушка». Поскольку управляющих элементов меньше, чем индикаторов, разделим выход одного из них на два за счет присоединения дополнительного проводника к любой из линий передачи данных.

1.5 При белой стрелке Run () включаем периодический запуск () работы составленных программ. Изменяя на лицевой панели значения исходных величин, проследим отображение этих изменений на показывающих приборах. Обратим внимание на соответствие шкал управляющих элементов и показывающих приборов. При необходимости скорректируйте их с помощью инструмента «редактирование текста».

1.6 Щелчком ПКМ на блок-диаграмме вызовем панель «Все функции» и закрепим ее. В ней находим палитру «арифметические действия», открываем и переносим на блок-диаграмму два элемента: «суммирование» -  (Add) и «генератор случайных чисел» -  (Random Num). Для этого выбираем: Functions→ Arith/Compare→ Numeric (рисунок 1.6). Выделяем щелчком ЛКМ проводник, соеди-

няющий выбранный регулятор с осциллографом и удаляем проводник.

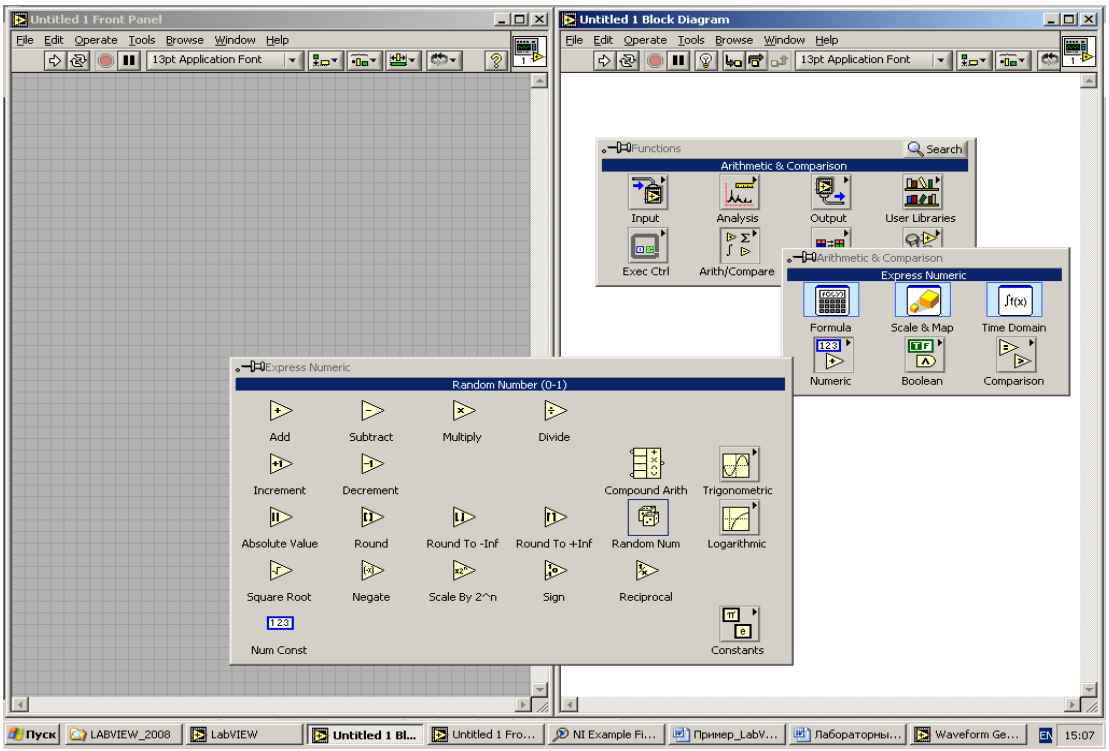


Рисунок 1.6 - Палитра арифметических действий

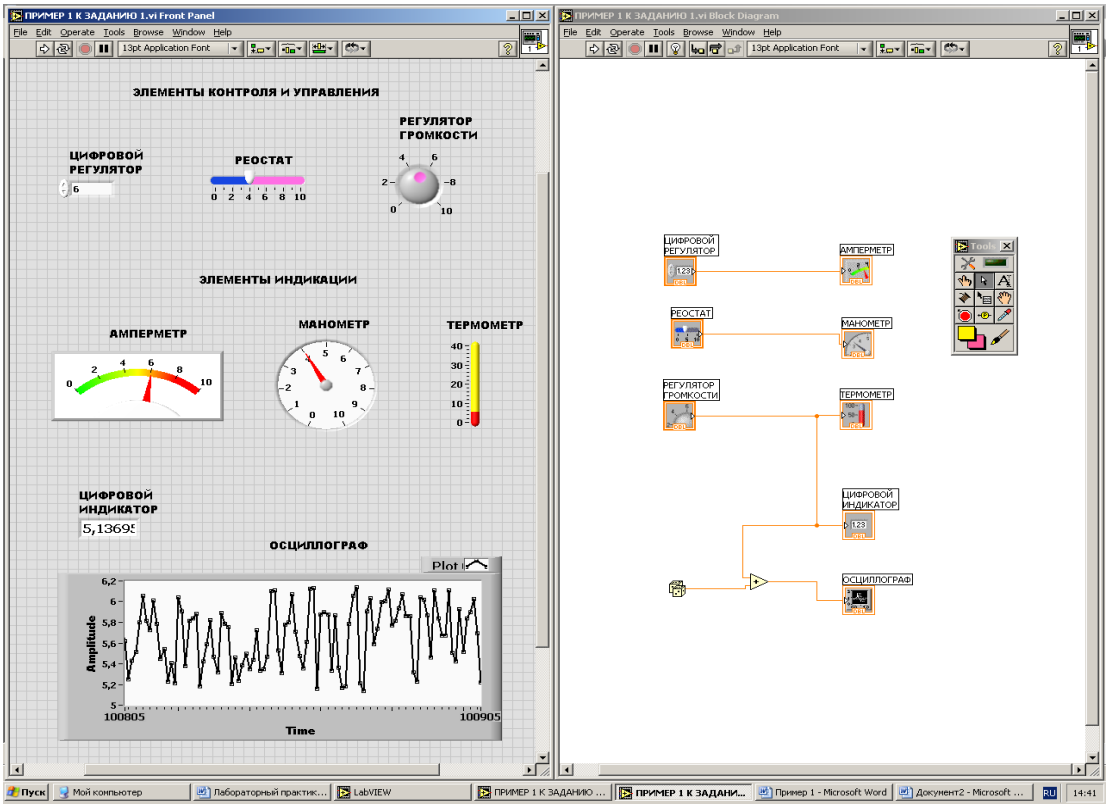


Рисунок 1.7 - Лицевая панель и блок-диаграмма задания 1

1.7 Выделяем элемент «суммирование» и вызываем справку Help, которая показывает схему его подключения. В соответствии с

этой схемой, подводим к одному из входов сумматора сигнал с выбранного регулятора, а к другому - генератор случайных чисел.

Результат суммирования подаем на вход осциллографа и при белой стрелке Run ( ) включаем периодический запуск ( ). При работающей программе рассмотрите и перепишите в отчет различные формы представления результатов вычисления на графике - в виде сплошной линии, отдельных точек, отрезков прямых, соединяющих соседние точки и т.д.

1.8 Остановите программу кнопкой «стоп» ( ). С помощью инструмента «Лампочка» и кнопки «периодический запуск» включите режим анимации потоков данных, используемый при отладке программ. Проследите движение данных по проводникам и их преобразование на элементах блок-диаграммы (рисунок 1.7).

Порядок выполнения задания 2

2.1 В LabVIEW существует большая библиотека примеров использования этой среды в различных областях знаний и практического использования в управлении технологическими процессами. Особенностями этих примеров является возможность использования предлагаемых решений в целом или частично в качестве программ или подпрограмм в самостоятельных разработках.

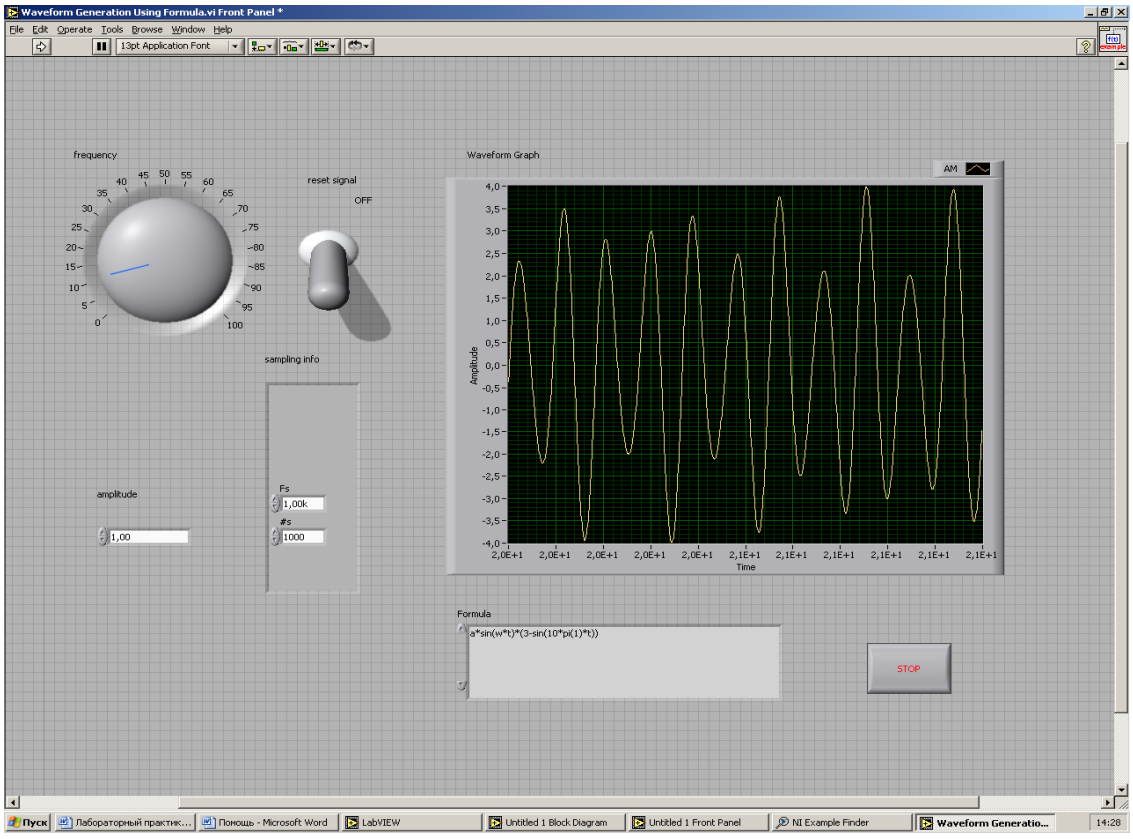


Рисунок 1.8 - Пример лицевой панели
для исследования функций

2.2 В качестве настоящего задания предлагается найти примеры использования среды для исследования функций, то есть построения графиков самой функции и ее производной, нахождения нулей и экстремальных значений. Для этого необходимо выбрать меню: Help→ Find Examples→ Analyzing and Processing Signals→Signal Processing→Waveform Generation Using Formula.vi и найти в нем подходящий аналог решаемой задачи (рисунки 1.8 и 1.9)

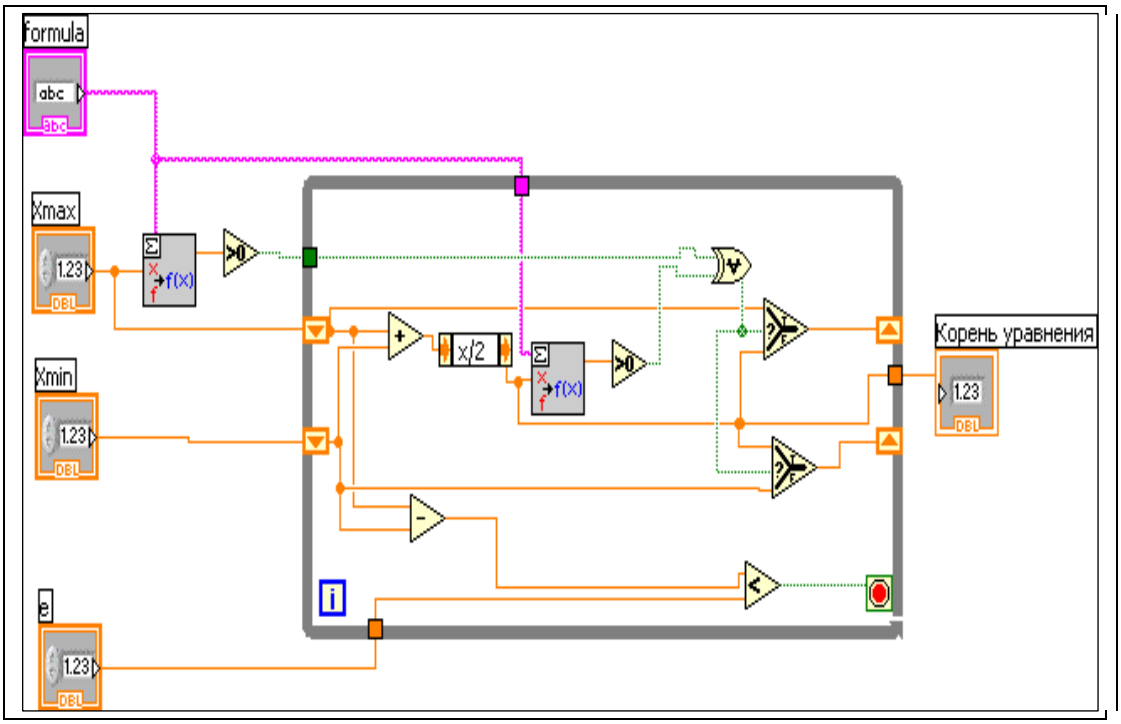


Рисунок 1.9 - Блок-диаграмма с графическим кодом задачи для решения нелинейного уравнения методом бисекции

2.3 Для выполнения настоящего задания выберите одну из функций, приведенных в таблице 1.4, в соответствии с порядковым номером компьютера, установленного на Вашем столе. Введите ее в окно для записи функций на лицевой панели задачи, запустите программу и перенесите в отчет условие задачи, полученный график функции, значения ее нулей и экстремумов и их положение.

Таблица 1.4 - Функции для исследования

Номер ПК	Функция	Значение параметра	Интервал
1	y= sin(a) - b*a	b= 0,1	0; 10

2	$y= a*x^3+ b*x^2+ c*x+ d$	a= 1, b= 3,7 c=0,293, d=-1,96	-5; 5

Продолжение таблицы 1.4

Номер ПК	Функция	Значение параметра	Интервал
3	$y= (\sin(a))/a - b*a$	b= 0,05	0; 10
4	$y= \exp(x) - a*\cos(x) -1$	a= 1,1	0; 1,0
5	$y= A* e^{-a*x} *\sin(x+1)$	A= 2, a= 0,03	0; 10
6	$y= a* e^{b*x+c*x\wedge 2} - 10$	a= 2,b=1, c=0,01	-5; 5
7	$y= a*x^2 + b*x - c$	a=1, b=2,c=-4	-4; 4
8	$y= x^4 - 16$	-	-16; 16
9	$y= a* e^{b*x+c*x\wedge 2} - 16$	a=2, b=0,1, c=0,01	-20; 20
10	$y= a^3 / (x^2 + a^2) - 5$	a=2	-3; 3
11	$y= A* e^{-a*x} *\cos(x+1)$	A=2, a=0,03	0; 10
12	$y= \pm\sqrt{a*x^2 + b*x + c}$	a=2, b=4, c=4	-3; 3

Контрольные вопросы:

1. Какие команды используются в среде LabVIEW для отладки программ?

2. Имеет ли какое-либо значение порядок подключения проводников к элементам суммирования и вычитания, умножения и деления?

3. Какая форма графического представления результатов работы программы в наибольшей мере отражает дискретный принцип работы ПК? В каких случаях целесообразнее использовать другие графики?

4. Опишите назначение каждого из элементов блок диаграммы, приведенной на рисунке 1.9.

5. Откройте на блок-диаграмме палитру всех функций и запишите в отчет ее основные элементы.

ЛАБОРАТОРНАЯ РАБОТА № 2

ИССЛЕДОВАНИЕ ФУНКЦИЙ И ПОСТРОЕНИЕ СЛОЖНЫХ КРИВЫХ В СРЕДЕ LABVIEW

Цель работы: изучение возможностей среды LabVIEW. Основные приемы программирования и отладки программ.

Задание 1. Разработать программу вычисления координат и построения графика окружности, заданной параметрическим способом:

X=X0+R*COS(A),

Y=Y0+R*SIN(A),

где R – радиус окружности, R = 3;

X0 – координата центра окружности, X0 = -1;

Y0 – координата центра окружности, Y0 = 2;

A – параметр, A є [0; 360°] .

Задание 2. Разработать программу выбора вариантов заданий курсовой работы с использованием генератора случайных чисел.

Основные сведения

Инженерная среда программирования LabVIEW создана для разработки компьютерных систем измерения и управления экспериментальными средствами и технологическими процессами. Ее главной особенностью является графический код, который позволяет быстро проектировать и отлаживать сложные программы. Последняя версия среды LabVIEW 8.5 содержит 14 экспресс-программ, которые позволяют в считанные часы превратить обычный персональный компьютер в мощную информационно-измерительную систему мирового уровня.

Кроме того, LabVIEW может использоваться для проведения различных расчетов, вычисления аналитических функций и построения их графиков; анализа и архивирования данных и генерации отчетов. Среда обладает уникальными возможностями быстрой отладки программ и наблюдения за потоком данных в анимационном режиме.

Порядок выполнения работы по заданию 1

1.1 Осуществите запуск среды LabVIEW из каталога D:\LABV. В появившемся главном окне программы выберите команды: *New→Blank VI* для создания нового файла. Далее выберите меню: *Window→The Left and Right* для одновременного отображения на экране двух окон программы: серой и белой панелей. Серая *Лицевая панель* (обычно располагается слева) – инструмент пользователя, который предназначен для размещения элементов ввода и вывода данных в виде привычных технических устройств, таких как: цифровые указатели, ползунковые реостаты, регуляторы громкости, осциллографы, самописцы, графопостроители и т.д.. Белая (обычно располагается справа) – *Блок-диаграмма*, на которой вызываются пиктограммы различных функций и структур и составляется графический код программы.

Для совершения различных операций с помощью курсора необходимо вызвать *Палитру инструментов* с помощью меню: *Window→Show Tools Palette* на *Лицевой панели* или на *Блок-диаграмм.*

1.2 Создайте на *Лицевой панели* четыре цифровых элемента управления для исходных данных задачи: X0, Y0, R, A, как показано на рисунке 1. Для этого щелчком ПКМ (правой кнопки мыши) по серой панели вызовите *Палитру элементов управления* (Controls) и закрепите ее, активизировав кнопку в левом верхнем углу палитры.

Откройте пункт меню *Num Ctrl*, выберите в нем первый элемент в верхнем ряду. На открывшейся *Палитре элементов управления* выделить элемент *Num Ctrl* со спаренной кнопкой изменения значения параметров (рисунок 2.1). Переместите четыре элемента поочередно перетаскиванием на *Лицевую панель* и расположите их горизонтально в одну строку.

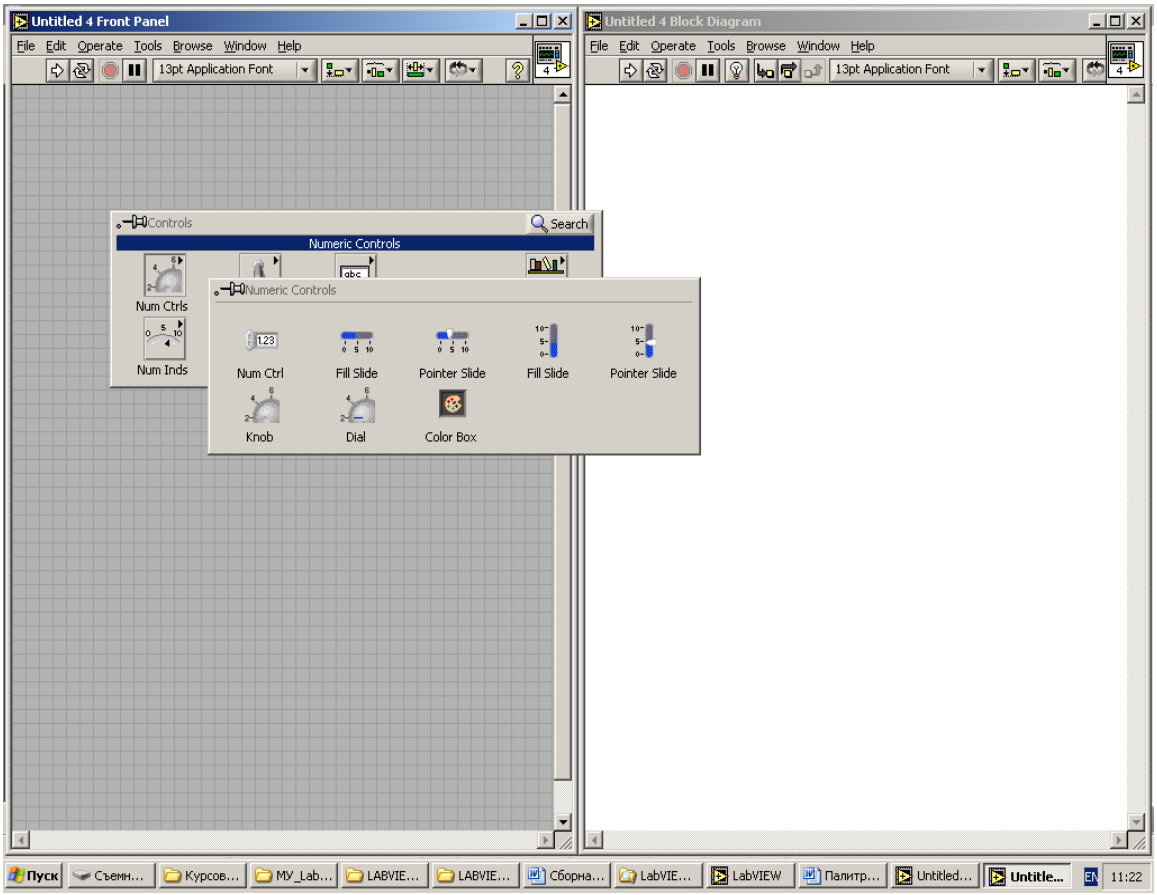


Рисунок 2.1 - Палитра элементов управления

Измените собственные метки вызванных регуляторов, подписав вместо *Numeric* новые обозначения: X0, Y0, R, A. Установите в окошках регуляторов соответствующие значения исходных данных. Значения X0=-1, Y0=2, R=3 набираются с помощью клавиш указателей, а число A, равное 360, с помощью инструмента ВВОД ТЕКСТА.

Создайте на *Лицевой панели* два прибора для отображения полученных данных - двухлучевой запоминающий осциллограф, работающий в режиме реального времени, и двухкоординатный самописец. Для этого вернитесь на главную панель *Палитры элементов управления* (Controls), откройте графические индикаторы (Graph Inds), из которых на *Лицевую панель* выносится первый (Waveform Graph) и третий элемент (XY-Graph) (рисунок 2.2).

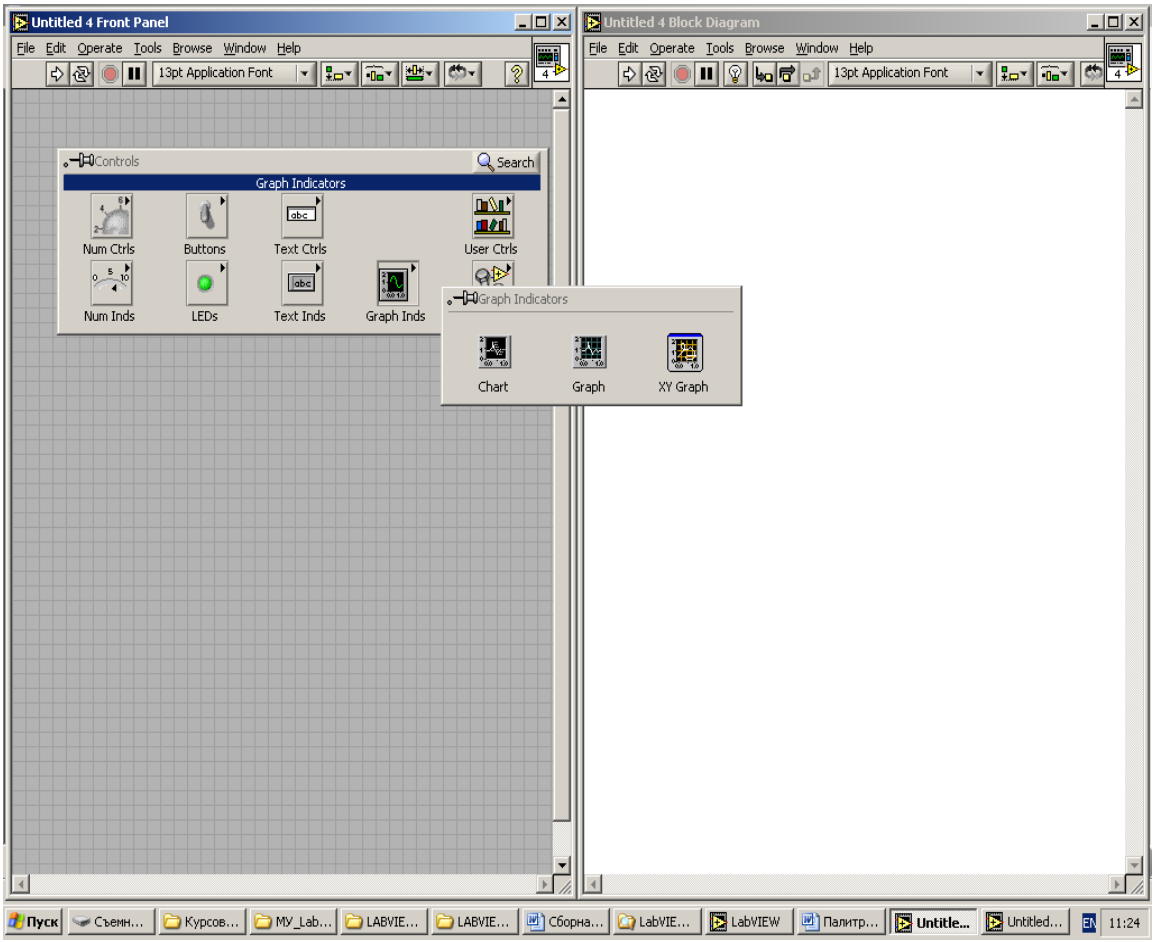


Рисунок 2.2 - Графические индикаторы

1.3 Обратите внимание на то, что при появлении элементов на *Лицевой панели,* их пиктограммы сразу же появляются на *Блок-диаграмме*. Освободите среднюю часть *Блок-диаграммы* для построения графического кода программы путем перетаскивания имеющихся элементов в верхнюю и нижнюю части экрана.

Далее щелчком ПКМ на *Блок-диаграмме* вызовите *Палитру функций* (Functions→ All functions). Используя кнопку в верхнем левом углу палитры, нужно зафиксировать ее на экране. В *Палитре всех функций* вызываем первый элемент «Структуры» в первом ряду в виде квадрата с утолщенными сторонами, далее в нем выбираем «цикл, повторяющий вычисления с заданным числом итераций - For loop». Перетаскиваем его на блок-диаграмму и растягиваем на большую часть экрана (рисунок 2.3).

Вернитесь к *Палитре всех функций*, откройте панель для операций с числами (Functions→ Arith/Compare→ Numeric) и зафиксируйте ее на экране. Далее реализуйте формулы, заданные в условии задачи, по пунктам 1.4 - 1.6. с помощью узлов подпалитры Function→ All function→ Numeric (рисунок 2.4).

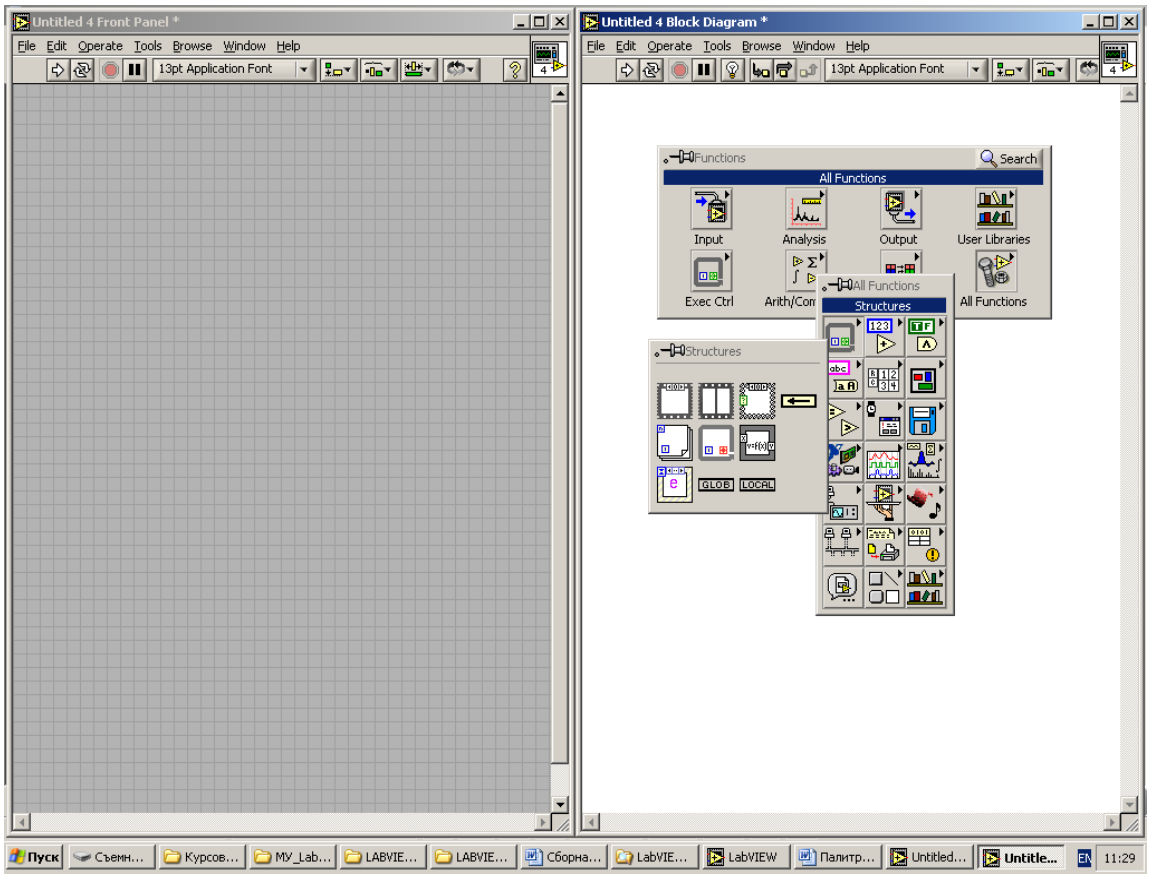


Рисунок 2.3 - Палитра функций для выбора циклических структур

1.4 Поскольку здесь, как и во многих других средах тригонометрические функции работают с радианной мерой угла, переведите градусы в радианы. Для этого в палитре операций с числами выберите число 2π в Numeric→ Constants→ 2π и перетащите его внутрь цикла на *Панели блок-диаграмм*.

Вернувшись в Numeric, перенесите на *Блок диаграмму* два арифметических узла: деление и сложение. С помощью "катушки" выведите значение элемента цифрового управления A на границу цикла. Дополнительным щелчком создайте здесь терминал (заштрихованный квадратик) ввода данных в цикл. Соедините выход узла 2π с верхним терминалом узла деления, а к нижнему с помощью проводника подведите значение A. Входы каждой пиктограммы располагаются на ее левой стороне, а выходы – на правой.

Выход узла деления соединяем с одним из входов узла умножения, а на другой подаем значение текущей итерации от узла "i" внутри цикла. Полученное на его выходе значение равно текущему значению A в радианной мере. При выполнении цикла "i" последовательно принимает значения от 0 до 359, а величина A меняется от 0 до 2π.

1.5 Щелчком ПКМ на панели Numeric найдите тригонометрические функции и пиктограммы функций SIN и COS (Functions→ Arith/Compare→ Numeric→ Trigonometric→ SIN или COS). Перенесите их внутрь цикла и с помощью инструмента "катушка" подайте на их входы значение A.

1.6 Вернитесь на панель арифметических действий и поставьте за каждой функцией SIN и COS пиктограмму умножения "*" и сложения "+" (Functions→ Arith/Compare→ Numeric→ элемент умножения "*" или сложения "+").

Умножьте R на значение SIN(A) и COS(A). Выходы узлов умножения просуммируйте с Y0 и X0 соответственно. Выходы сумматоров выведите на правую границу цикла.

1.7 Соедините образовавшиеся массивы данных X и Y с соответствующими входами XY-самописца для получения графика окружности. Обратите внимание на «сломанную стрелку» запуска программы. Активизируйте ее ЛКМ (левой кнопкой мыши), в появившемся окне появится перечень ошибок, допущенных при составлении программы. Одна из них: не заполненный терминал счетчика итераций, который подает команду на завершение циклических вычислений. Исправьте эту ошибку, продублировав соединение элемента управления A с внешним входом счетчика итераций «N». Остальные ошибки, если они присутствуют, исправьте самостоятельно.

1.8 После исправления ошибок, запустите программу и убедитесь, что построенная кривая является окружностью с радиусом R= 3 и центром окружности, расположенным в точке X0=-1, Y0=2.

Запустите программу в режиме анимации потоков данных. Для этого активизируйте в строке команд на *Блок-диаграмме* элемент в виде "лампочки", а затем стрелку запуска программы. Изучите движение данных и их трансформацию в различных узлах *Блок- диаграммы* и сделайте выводы о возможности использования этого режима при отладке сложных программ.

1.9 Для отображения изменения координат окружности в зависимости от параметра внесите внутрь цикла двухлучевой осциллограф. Для преобразования его в многолучевой прибор на *Палитре функций* (Functions→All functions) откройте третью пиктограмму во второй строке (Cluster), а в ней пиктограмму – Bundle (объединение), захватите ее и перенесите внутрь цикла, к входу осциллографа. К двум ее входам подключите X и Y, взятые от проводников соответ-

ствующих данных. Выход объединительной панели соедините с осциллографом.

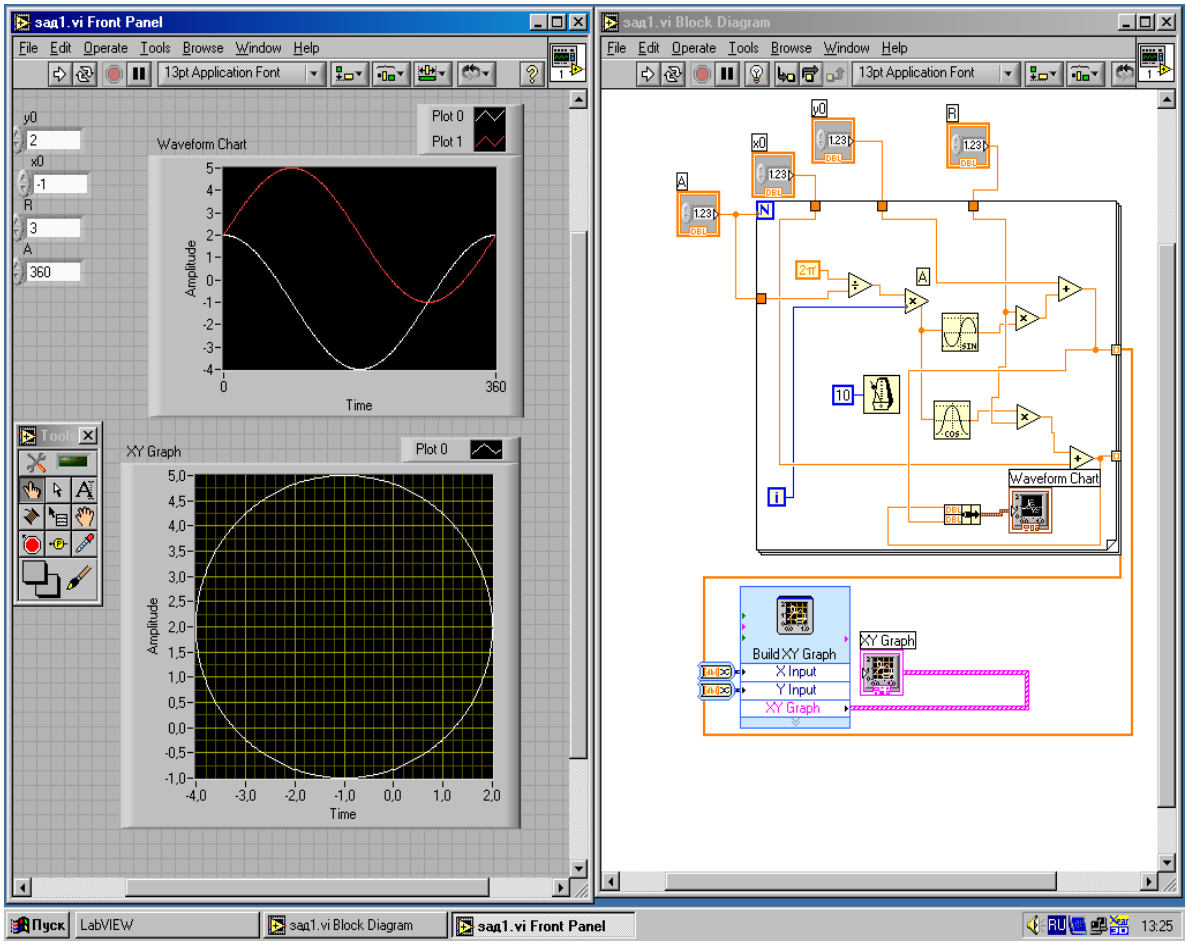


Рисунок 2.4 - Лицевая панель и блок-диаграмма расчета координат и построения кривой окружности

1.10 Для того, чтобы замедлить вычисления и наблюдать построение кривых координат окружностей во времени, с *Палитры функций* (ПКМ→Functions→All functions) вызывается пиктограмма Time&Dicelog, в виде метронома, который переносится внутрь цикла. По умолчанию, цикл рассчитывает всю программу со скоростью 1000 раз в секунду. Подведите инструмент "катушка" с левой стороны "метронома" к терминалу. Выберите команды: ПКМ→ Create→ Constant и с клавиатуры наберите время задержки 20 миллисекунд. При этом процесс расчета и отображения координат X и Y замедлится в 20 раз.

Запустите программу и просмотрите процесс построения графиков. Обратите внимание, что первые графики строятся в реальном

времени, а сама кривая в координатах X-Y на самописце только после окончания всего цикла вычислений.

1.11 Перенесите в отчет по лабораторной работе блок-диаграмму вычислений и построения графиков в соответствии с рисунком 2.4. Сделайте выводы по результатам выполненной работы.

Порядок выполнения работы по заданию 2

2.1 Создайте бланк для новой программы по пункту 1.1.

2.2 Разместите на *Лицевой панели* массив индикаторов номеров заданий Вашей курсовой работы. Для этого в *Палитре* элементов контроля и индикации, вызванной щелчком ПКМ по *Лицевой панели,* открыть все функции и во второй строке найти элемент массива (Function→ Array → Array constant). В появившийся шаблон вставить цифровой элемент индикации с той же *Палитры*, как показано на рисунке 2.5. С помощью курсора растяните шаблон вниз на три позиции с тем, чтобы число индикаторов было равно количеству заданий курсовой работы (3). Напишите название задачи в верхней части *Лицевой панели*.

2.3 Выбор варианта осуществляется с помощью цикла While. Он вызывается с *Панели всех функций* на *Блок-диаграмме* (ПКМ→ Functions→ All functions). В *Палитре* «Структуры» выбираем «цикл по условию»: While, перетаскиваем его на блок-диаграмму и растягиваем на большую часть экрана.

2.4 Внутрь цикла помещают: генератор случайных чисел, мультипликатор, узлы «прибавление единицы» и «целая часть числа» и соединяют их как показано на рисунке 2.5. Результаты вычислений выводятся на границу цикла и соединяются с массивом. Необходимо обратить внимание, что в цикле While, по умолчанию, появляется закрашенный терминал, который выводит не весь массив данных, а только результат последнего вычисления. Для изменения терминала на вывод всего массива, необходимо щелкнуть по нему ПКМ и изменить форму терминала (переставить галочку на верхнюю строчку выпадающего меню).

2.5 Условие завершения цикла здесь можно сконструировать самостоятельно. Для этого внести в нижнюю часть Блок-диаграммы элементы: «прибавление единицы» и логический элемент «больше». К нижнему терминалу этого элемента подключаем константу 3, как

это сделано в пункте 1.10 (ПКМ→Create→Constant). При первой, второй и третьей итерациях на выходе логического элемента появляется логический сигнал «ложь» и цикл продолжает свою работу. В том случае, когда неравенство выполняется, цикл завершается.

2.6 **Запустите программу**. Запишите в отчет по лабораторной работе строку «Варианты задач курсовой работы студента …(ФИО) группы …. Подпишите под ней полученную последовательность случайных чисел, поставьте число, свою подпись и подпишите у преподавателя.

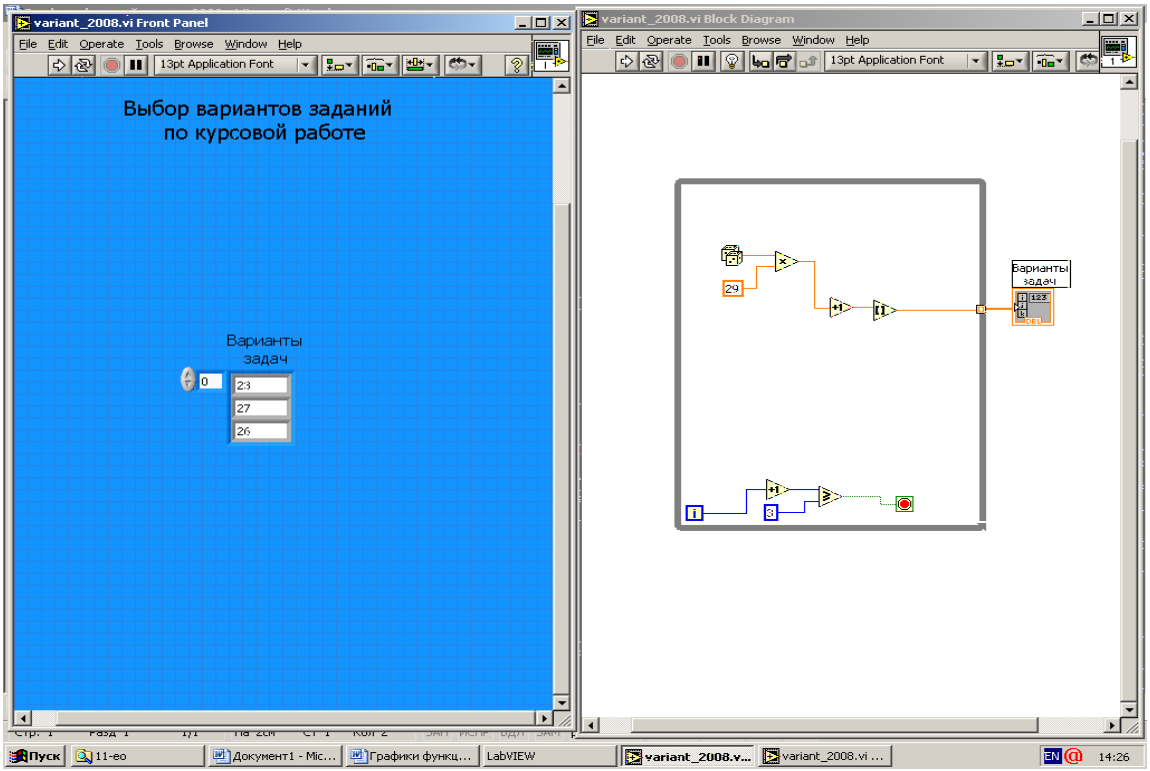


Рисунок 2.5 - Программа выбора вариантов заданий курсовой работы

Выводы

1. Рассмотрены основы программирования в среде LabVIEW. Установлены ее преимущества перед другими средами по скорости программирования, методам отладки сложных программ и способам графического представления данных.

2. Самостоятельно разработана программа вычисления координат и построения графика окружности, заданной параметрическим способом. Результаты вычислений представлены в виде графика зависимости координат от параметра и кривой окружности в координатах X-Y.

3. Разработана программа, в которой с помощью генератора случайных чисел выбраны непересекающиеся варианты заданий для курсовой работы.

Контрольные вопросы:

1. Назовите области возможного применения в процессе дальнейшего обучения.
2. Каковы отличительные черты интерфейса среды?
3. В чем заключаются принципиальные отличия разработки программ в графической и текстовой средах программирования?
4. Как увеличить число каналов компьютерного осциллографа?
5. Какие инструменты служат для отладки программ и обнаружения ошибок? Для каких целей используется режим анимации потоков данных?
6. Чем отличается работа узлов и индикаторов, расположенных внутри и снаружи цикла?
7. Чем отличаются циклы For и While?
8. В каких случаях применяется функция задержки времени?

ЛАБОРАТОРНАЯ РАБОТА № 3

МОДЕЛИРОВАНИЕ ФИЗИЧЕСКИХ ПРОЦЕССОВ В ИНЖЕНЕРНОЙ СРЕДЕ LABVIEW

Цель: изучение возможностей среды для математического моделирования физических явлений и процессов, создания подпрограмм обработки данных.

Задание. Разработать программу моделирования политропного процесса сжатия воздуха в цилиндре объемом V0=1 л с начальным давлением P0=100 кПа и температурой T0=300K при степени сжатия λ=V0/VK=5. Результаты вычислений отобразить в виде индикаторов традиционных приборов, служащих для измерения параметров состояния V, P, T, графиков их изменения по времени и P-V диаграммы исследованного процесса.

Основные положения

В настоящее время решение большинства научно-технических задач базируется на использовании компьютерного моделирования, когда этапам проведения физических экспериментов, разработки новых изделий и технологий предшествуют сложные математические расчеты, создание имитационных моделей и т.п.

В качестве примера в настоящей работе рассматриваются возможности инженерной среды графического программирования для моделирования термодинамических процессов идеального газа. Процессом называется любое изменение параметров его состояния. Обычно изменяются все три параметра, связанные между собой уравнением PV=RT. Существует ряд процессов, в течение которых сохраняется постоянное отношение выполненной работы и количества тепла, участвующего в теплообмене с внешней средой. Такие процессы называются политропными. Для них выполняется дополнительное соотношение PV^n=const.

Если в политропном процессе воздух, являющийся идеальным газом, сжимается очень быстро, то при уменьшении объема в 15 раз, температура его повышается до 650 °С. В него можно впрыснуть дизельное топливо и оно самовоспламеняется. Таким способом может быть реализован один из процессов, с помощью которого приводится в действие дизельный двигатель.

При той же степени сжатия, осуществляемого очень медленно, температура остается без изменений. Это связано с тем, что в медленном процессе тепловая энергия, которая образуется при сжатии газа, успевает рассеяться в окружающей среде. Таким образом, характер изменения параметров состояния фактически зависит от скорости процесса. В первом случае показатель политропы n равен коэффициенту адиабаты n=1,4 (адиабатический процесс), во втором n=1 (изотермический процесс).

Порядок выполнения задания

1.1 Осуществите запуск среды LabVIEW из каталога D:\LABV. В появившемся главном окне программы выберите команды: *New→Blank VI* для создания нового файла. Далее выберите меню: *Window→The Left and Right* для одновременного отображения на экране двух окон программы - серой *Лицевой панели* и белой панели *Блок-диаграмм.* Для дальнейшей работы необходимо вызвать *Палитру инструментов* с помощью меню: *Window→Show Tools Palette* на *Лицевой панели* или на *Блок-диаграмме.*

1.2 Создайте на *Лицевой панели* четыре цифровых элемента управления для исходных данных задачи: V0, P0, T0, n как показано на рисунках 3.1 и 3.2. Для этого щелчком ПКМ (правой кнопки мыши) по серой панели вызовите *Палитру элементов управления* (Controls) и закрепите ее, активизировав кнопку в левом верхнем углу палитры.

Откройте пункт меню *Num Ctrl*, выберите в нем первый элемент в верхнем ряду. На открывшейся *Палитре элементов управления* выделить элемент *Num Ctrl*. Переместите четыре элемента поочередно перетаскиванием на *Лицевую панель* и расположите их горизонтально в одну строку.

Измените собственные метки управляющих элементов, подписав вместо *Numeric* новые обозначения: V0, P0, T0, n. Установите в окошках регуляторов соответствующие значения исходных данных (n=1) с помощью инструмента «ввод текста (**A**)».

Для отображения полученных данных V, P, T создайте на *Лицевой панели* три прибора - мерную емкость, манометр и индикатор температуры. Подпишите названия этих приборов на русском языке и измените верхние пределы их шкал - для объема - 1л, давления - 2000 кПа, температуры 1000 К. Для наблюдения за ходом процесса создайте на *Лицевой панели* трехлучевой запоминающий осциллограф и X-Y-самописец для построения P-V диаграммы процесса → *Палитры элементов управления* (Controls), графические индикаторы (Graph Inds), первый (Waveform Graph) и третий элемент (XY-Graph). Измените легенды шкал, как показано на рисунках 3.1 и 3.2.

1.3 Освободите среднюю часть *Блок-диаграммы* для построения графического кода программы. Щелчком ПКМ на *Блок-диаграмме* вызовите *Палитру функций* (Functions→All functions). Используя кнопку в верхнем левом углу палитры, зафиксируйте ее на экране.

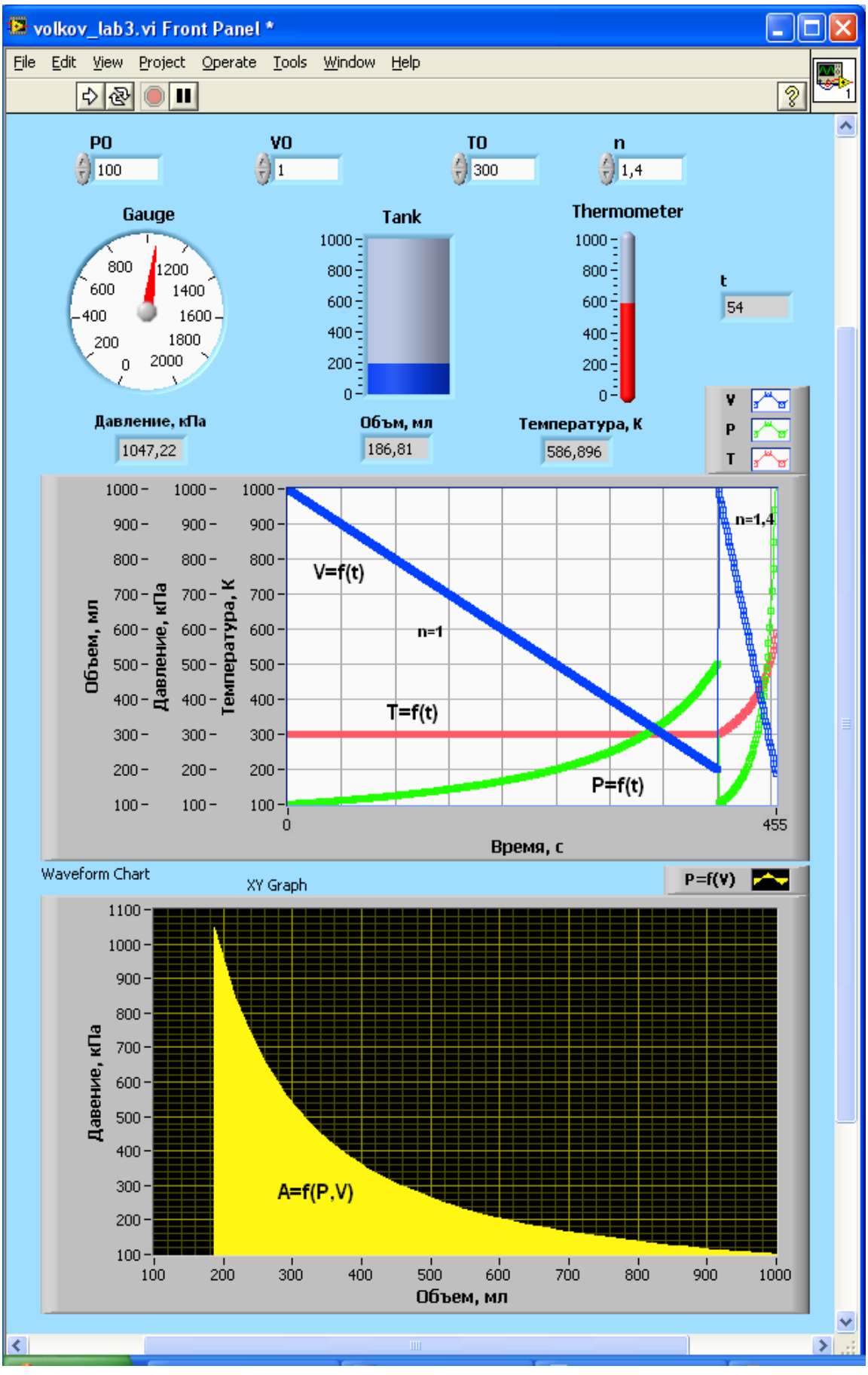


Рисунок 3.1 - Лицевая панель моделирования процессов сжатия

В *Палитре всех функций* вызвать первый элемент в первом ряду в виде квадрата с утолщенными сторонами, далее в нем выбираем цикл по условию *While,* перетаскиваем его на блок-диаграмму и растягиваем на большую часть экрана. Вернитесь к «Структурам», выберите формульный узел и перенесите его внутрь цикла (рисунок 3.2)

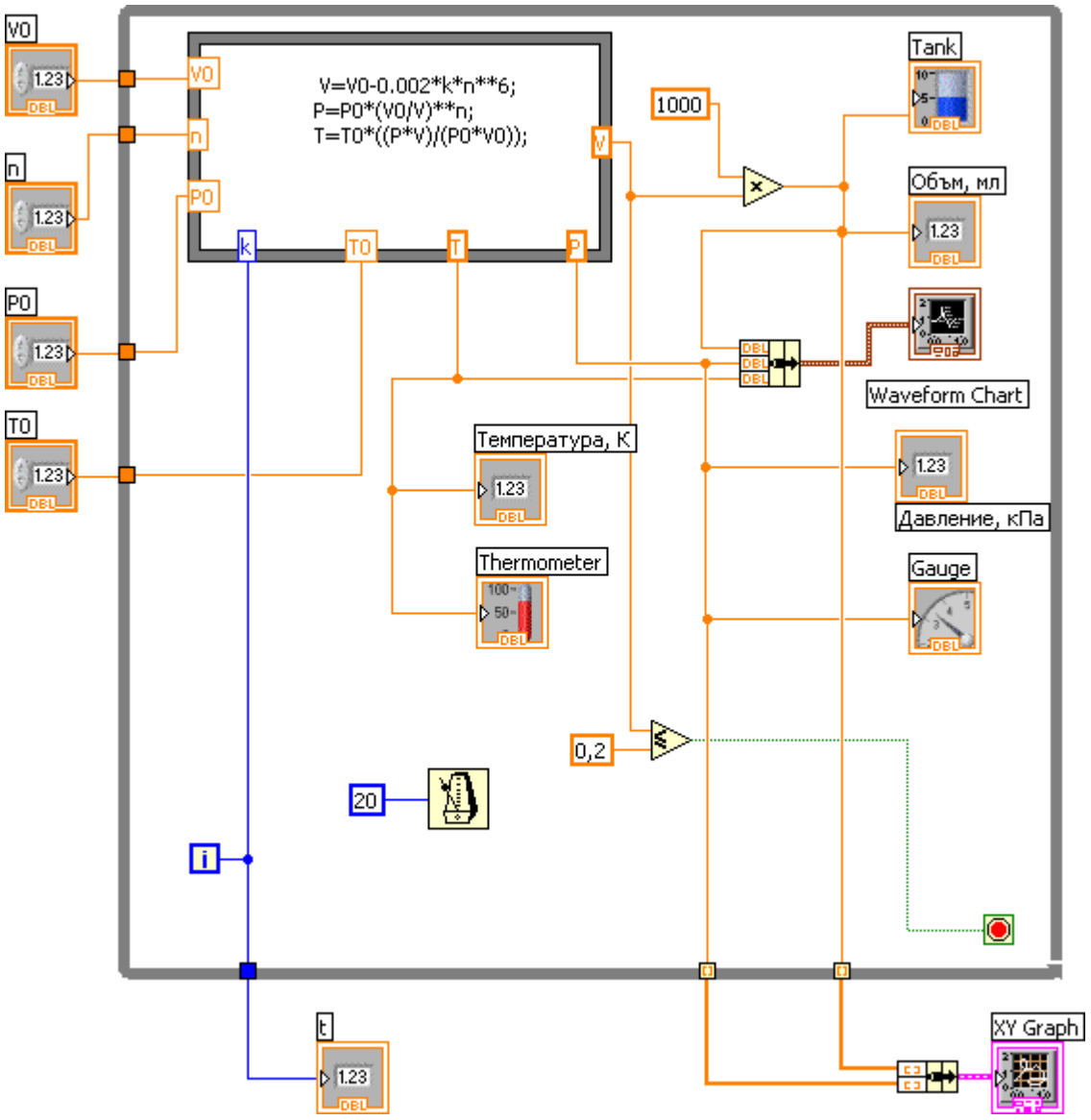


Рисунок 3.2 - Блок-схема программы расчета параметров сжимаемого газа

В нашем случае определяющим процессом в моделировании является движение поршня и соответствующее изменение объема сжатого воздуха. Свяжем скорость этого процесса с показателем политропы следующим образом:

$$V = V0 - 0{,}002 * k * n^5,$$

где k=i.

Таким образом, при моделировании время процесса принимается равным i - номеру текущей итерации цикла.

Это означает, что за каждый цикл моделирования процесса объем сжимаемого газа линейно уменьшается на ($2*n^5$) мл. Введем эту формулу в подготовленный узел и создадим на его границе три терминала ввода необходимых для расчета V данных - V0, k, n, а также один терминал для вывода V. Для этого щелчком ПКМ на границе узла вызываем всплывающее меню, в нем - три раза "Add Input" и один раз "Add Output".

Впишем в появившиеся терминалы буквенные обозначения переменных и с помощью инструмента «проводник» соединим их соответственно с пиктограммами V0, n и узлом счетчика итераций, считая, что k =i.

1.4 Выведем полученное значение V на границу цикла, внесем в цикл пиктограмму мерной емкости, осциллограф и подадим значение V на их входы.

Так как условие λ=V0/VK=5 (VK меньше или равно 0,2 литра), используется для завершения цикла, находим в *Палитре всех функций* подпалитру *Логические функции*, в ней узел «≤». Подводим текущее значение V к верхнему терминалу логического узла, а к нижнему подводим константу 0,2. Логический результат *Falshe* или *True* - присоединяем к терминалу завершения цикла. Обратим внимание на цвет проводника, соответствующий логическому типу данных. Так как по умолчанию каждый цикл рассчитывается всего за одну миллисекунду, то для отслеживания динамики процесса установим задержку цикла, равную 20 мс. Для этого на *Панели всех функций* выбираем пиктограмму прибора времени, в ней метроном. Помещаем его внутри цикла, находим входной терминал и щелчком ПКМ вызываем всплывающее меню, в нем Create→ Const. В появившемся прямоугольнике с клавиатуры набираем число 20.

Убедитесь, что стрелка запуска цикла имеет правильную, не изломанную форму. Это означает, что программа составлена правильно и готова к запуску. В противном случае - щелчком ПКМ по стрелке вызываем контекстное меню с распечаткой допущенных ошибок. Устраняем их и запускаем программу. При этом объем сжатого воздуха в мерной емкости за 10-15 секунд уменьшается от 1 до 0,2 литров, а на осциллографе появляется график уменьшения объема в виде прямой наклонной линии. На этом создание и отладка программного управления изменением объема сжимаемого газа завершается.

1.5 Соответствующее рассмотренному процессу изменение давления в цилиндре описывается формулой $P=P0*(V0/V)^n$.

Вносим это соотношение в формульный узел. Добавляем дополнительный вход для ввода P0 и с помощью проводника связываем его с соответствующим элементом управления. На правой границе формульного узла создаем выход P и соединяем его с пиктограммой манометра, внесенной внутрь цикла.

Для одновременного отображения графиков изменения давления и объема воздуха на одном и том же приборе преобразуем однолучевой осциллограф в двулучевой. Для этого в *Палитре всех функций* выбираем массивы и кластеры, в них элемент Bunde – «объединение». Активизируем проводник, соединяющий осциллограф с выходом V и убираем его. К нижнему входу элемента «объединение» подводим значение P, а у верхнего – восстанавливаем соединение с V. Выход элемента «объединение» соединяем с входом компьютерного осциллографа, который с этого момента становится двулучевым.

Работа по созданию подпрограммы моделирования изменения давления при сжатии может считаться завершенной, если стрелка запуска имеет правильный вид. Для проверки правильности работы этой программы необходимо очистить предыдущий график щелчком ПКМ по экрану осциллографа набором команд во всплывающем меню Data Operation и Clear Car.

Запустить программу и убедиться в том, что за время процесса давление в цилиндре изменяется от 100 до 500 кПа, а график его изменения по времени представляет собой возрастающую экспоненциальную функцию.

1.6 Изменение температуры в исследуемом процессе определяется соотношением: $T=T0*((P*V)/(P0*V0))$. Введем его в формульный узел, добавим дополнительный вход T0 и выход для полученного значения T. Соединим выход T и пиктограмму термометра, помещенную внутри цикла. Добавим еще один канал соединения с осциллографом. Для этого с помощью курсора в виде стрелки активизируем элемент «объединение» и растянем его вниз на одну новую позицию. Подведем к образовавшемуся новому входу сигнал T и соединим общий выход с осциллографом.

Далее необходимо убедиться в правильности составленной подпрограммы, очисть прежний график и запустить программу целиком. При n=1 показания термометра остаются на том же уровне, так как этот изотермический процесс характеризуется как раз постоянным

значением температуры. Давление и объем изменяются, как и в предыдущем случае.

1.7 Предусмотреть вывод конечных значений параметров моделируемого процесса на лицевую панель работы (рисунок 3.3). Для этого щелчком ПКМ по серой панели вызвать 5 цифровых индикаторов и расположить их в следующей последовательности в соответствии с таблицей 3.1.

Таблица 3.1 - Цифровые индикаторы

n	t, мс	Vк, л	Рк, кПа	Тк, К

Вывести значения этих параметров на правую границу цикла и соединить их с соответствующими индикаторами. Повторить запуски программы при n=1,2 и n=1,4. Перенести измеренные значения в таблицу 3.2.

Таблица 3.2 - Результаты моделирования

№ п/п	Параметры				
	n	t, мс	V, л	P, кПа	T, К
1	1				
2	1,2				
3	1,4				

1.8 Полученные значения параметров состояния могут быть использованы для автоматического построения P-V диаграммы исследуемого процесса. Для этого необходимо вывести на границу цикла текущее значение P и V. По умолчанию выходные терминалы в цикле While сохраняют только последние значения цикла как в пункте 1.7. Для того, чтобы при моделировании были бы сохранены все значения, необходимо для P и V создать еще по одному параллельному выходу и изменить их вид. Для этого щелчком ПКМ по терминалу вызвать контекстное меню и поменять выходы. Далее соединить выходные терминалы V и P соответственно с X -Y входами двухкоординатного самописца. Обратить внимание, что толщина проводников для массивов чисел, передаваемых из выходных терминалов цикла, больше чем у проводников одиночных скалярных величин (рисунок 3.4).

В очередной раз очистить графики и запустить программу. Проанализировать вид P-V диаграммы. Изменить форму представления

данных на выделение области, лежащей под кривой P(V) , и дать ее физическую интерпретацию

$$A=\Sigma P*dv$$

1.9 Провести численное моделирование процесса при n = 1; 1,2 и 1,4. Полученные данные перенести в таблицу 3.2 и проанализировать их.

Выводы

1. Разработана универсальная программа моделирования политропного процесса сжатия воздуха в цилиндре, объем, начальное давление и температура в котором могут быть заданы произвольным образом.

2. Установлено, что в медленном изотермическом процессе степень повышения давления равна степени сжатия воздуха. При быстром адиабатическом сжатии давление воздуха в несколько раз превышает изотермическое и при степени сжатия 5 увеличивается более, чем в 10 раз. Температура при этом достигает 560 К.

3. Построены P-V диаграммы исследуемых процессов. Показано, что площадь под кривой сжатия на P-V диаграмме численно равна механической работе затраченной на сжатие воздуха.

Контрольные вопросы

1. Приведите примеры других задач, которые могут решаться с помощью компьютерного моделирования.

2. Объясните, как работает цикл While.

3. Для чего на границах цикла формируются массивы данных P и V?

4. Что дает представление полученных зависимостей в виде площади под кривой процесса?

5. Назовите основные типы данных, использованных при выполнении задания.

ЛАБОРАТОРНАЯ РАБОТА № 4

АВТОМАТИЗАЦИЯ ЭКСПЕРИМЕНТАЛЬНЫХ ИССЛЕДОВАНИЙ В СРЕДЕ LABVIEW

Цель: приобретение навыков работы с автоматизированными системами сбора экспериментальных данных; использования формульных узлов и сдвиговых регистров для обработки данных в режиме реального времени.

Задание 1. Разработать подпрограмму автоматизации обработки данных физического эксперимента по определению электрической емкости конденсатора методом суммирования количества зарядов, стекающих с его обкладок. Запустить программу и исследовать зависимость времени разрядки конденсатора при сопротивлении нагрузки 100, 500 и 1000 кОм.

Задание 2. Изучить программу автоматизации реального эксперимента, выявить место и роль автоматизированной системы сбора экспериментальных данных и разработанной по заданию 1 подпрограммы. Провести эксперимент и сделать выводы о целесообразности использования моделирования в качестве предварительного этапа исследований.

Основные положения

Электрическая емкость уединенного проводника характеризуется величиной накопленного на нем заряда к потенциалу этого проводника. Единица электроемкости является фарад (Ф),

$$1\ Ф = 1\ Кл/В$$

Увеличить величину накапливаемого электрического заряда, а следовательно и энергию электростатического поля можно использованием системы заряженных проводников, представляющих собой плоские пластины, концентрические цилиндрические или сферические поверхности.

Электрическая емкость конденсатора - это физическая величина, равная отношению заряда на одной из его поверхностей к разности потенциалов между соседними поверхностями.

$$\tilde{N} = \frac{Q}{U},$$

где Q – величина заряда, U – разность потенциалов.

Две проводящих поверхности, образующие конденсатор, называются его обкладками. При равных по модулю и противоположных по знаку зарядов этих поверхностей электрическое поле конденсатора практически целиком заключено в пространстве между его обкладками. Характер силовых линий в конденсаторах различного типа показан на рисунке 4.1.

Величину заряда, находящегося на обкладках конденсатора можно вычислить, измеряя ток при разрядке конденсатора. Действительно сила тока характеризуется количеством элементарных зарядов протекающих через проводник в единицу времени.

$$1\ \text{А}=1\ \text{Кл/с},$$

$$Q = n \cdot e,$$

где n – число элементарных зарядов, e = $1{,}6 \cdot 10^{-19}$ Кл.

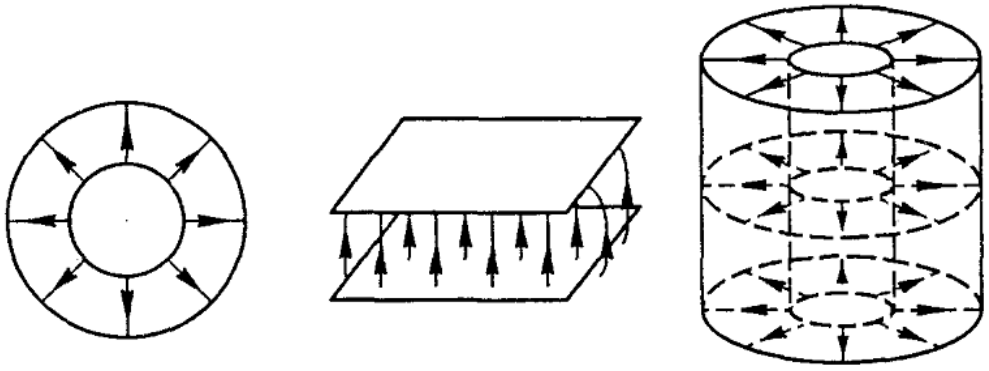


Рисунок 4.1 - Электрическое поле в сферическом, плоском и цилиндрическом конденсаторах

Суммируя произведения силы тока за малые интервалы времени за все время разрядки, можно получить величину заряда, находившуюся на обкладках конденсатора по формуле:

$$Q = I_1 \cdot \Delta t_1 + I_2 \cdot \Delta t_2 + ... + I_n \cdot \Delta t_n,$$

где Q – величина заряда, I – сила тока, Δt_n – интервал времени.

Разделив полученную величину на начальное напряжение на конденсаторе, получим значение электроемкости конденсатора.

Описание экспериментальной установки

Лабораторная установка состоит из конденсатора определяемой емкости С от 400 до 4000 мкФ, источника напряжения с ЭДС=3В, сопротивления нагрузки R от 500 Ом до 10 кОм, ключа K и компьютерной измерительной лаборатории в соответствии с рисунком 4.2. При горизонтальном положении ключа конденсатор оказывается заряженным до 3 В. При вертикальном положении - источник питания отключается, и конденсатор начинает разряжаться через сопротивление нагрузки. Для оставшейся активной цепи справедливо $I = U/R$.

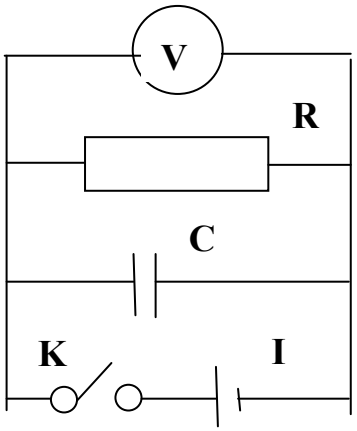


Рисунок 4.2 - Схема экспериментальной установки

Порядок выполнения задания 1

1.1 Выйти в главное диалоговое окно LabVIEW.

1.2 Создадим на лицевой панели: стрелочный прибор для контроля падения напряжения на конденсаторе, три цифровых элемента управления для ввода данных ЭДС, сопротивления R и остаточного напряжения U_k, три цифровых индикатора для отображения величины заряда Q, емкости конденсатора С и времени разрядки t, как показано на рисунке 4.3.

Для наблюдения за процессом поместим на *Лицевую панель* три осциллографа для регистрации текущих значений напряжения на обкладках конденсатора, силы тока на сопротивлении нагрузки и количества зарядов, стекающих с конденсатора. Щелчком ПКМ по серой панели вызываем соответствующие элементы контроля и индикации. Изменяем собственные метки индикаторов и элементов контроля и вводим значения ЭДС=3В, R=1000 Ом, Uk=0,1 В.

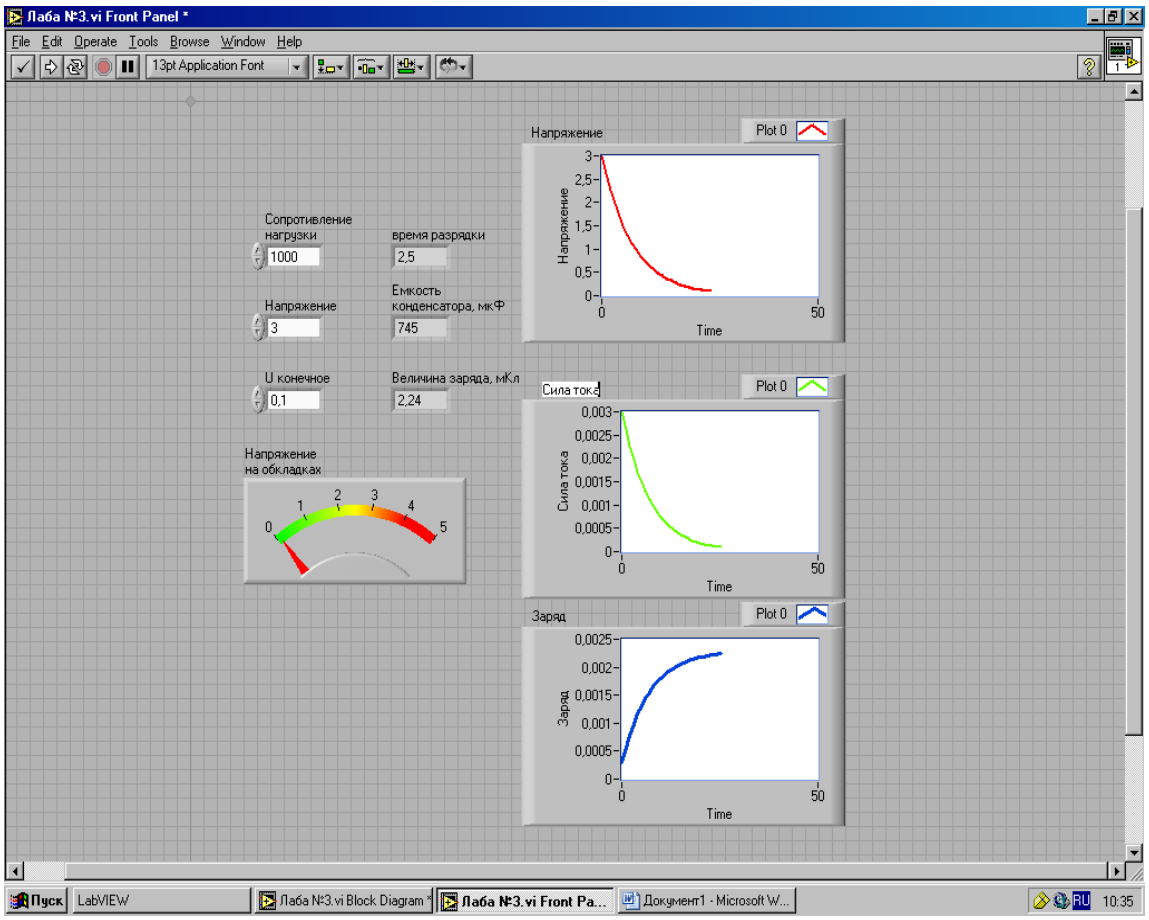


Рисунок 4. 3 - Лицевая панель моделирования разрядки конденсатора

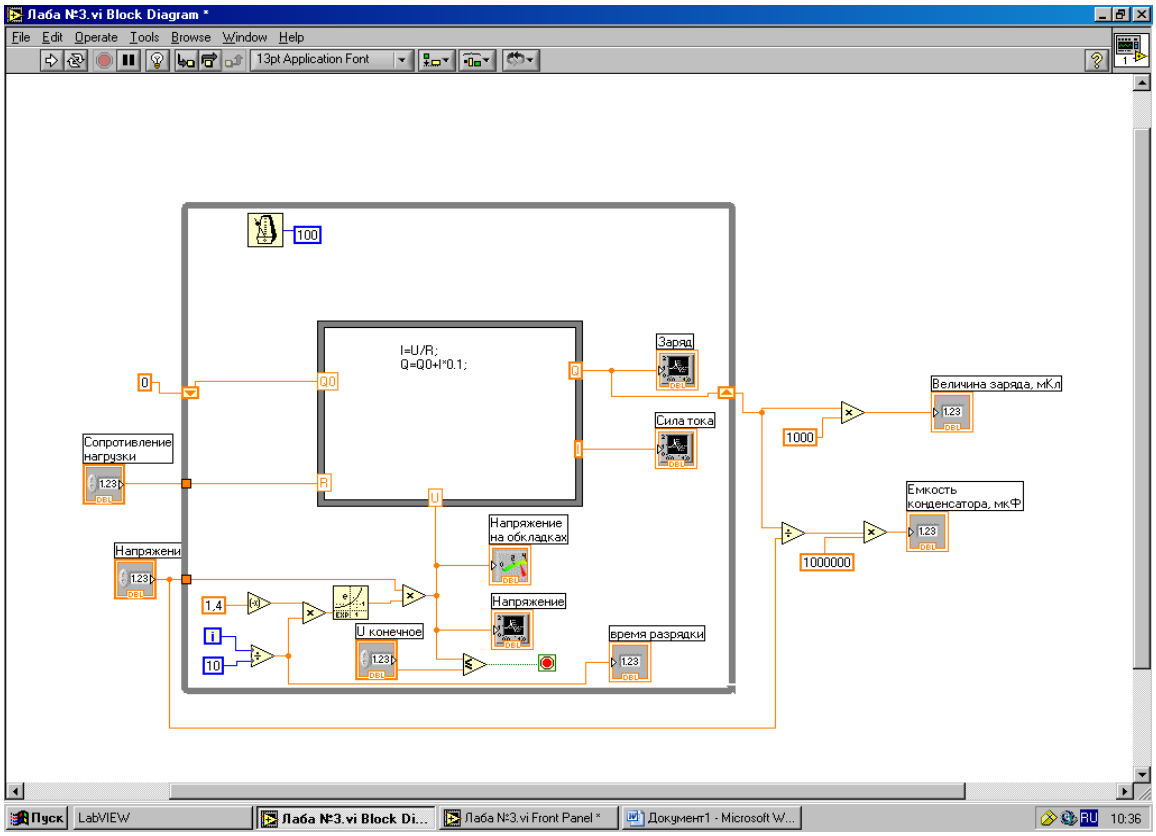


Рисунок 4.4 - Блок-схема моделирования разрядки конденсатора

1.3 Вызываем на блок-диаграмме все функции, в них структуры, далее цикл While, переносим его пиктограмму на белую панель и растягиваем на большую часть экрана. Вновь возвращаемся в структуры, активизируем узел формулы, переносим его внутрь цикла и растягиваем его в соответствии с рисунком 4.4.

Вписываем в узел формулы для определения силы тока разрядки I=U/R и суммирования величины зарядов, стекающих с обкладок конденсатора Q=Q0+I*dt. При этом вторая формула предусматривает организацию процесса численного интегрирования, в котором на каждой новой итерации используется предыдущее значение Q, которое в формуле всякий раз учитывается как новое значение Q0. Для запоминания вычисленного значения Q и возвращения его в цикл используется «сдвиговый регистр», который устанавливается щелчком ПКМ на правой границе цикла → по команде Add Register. При этом на левой и правой границе появляются два терминала – регистры с разнонаправленными стрелками. Правый регистр указывает, что поступившие значения выводятся за границы цикла, а левый - что выведенное значение вновь возвращается в цикл. Для исключения неопределенности при выполнении первой итерации, необходимо задать начальное значение входной регистр равное нулю. Для этого щелчком ПКМ по нему вызвать всплывающее меню, в нем Create→Const, в появившемся прямоугольнике с клавиатуры наберите 0.

1.4 Для работы формульного узла необходимо ввести в него следующие данные Q0, R и U. Щелчком ПКМ на границах узла создаем три входа. В один из них подводим значение сопротивления, в другой – сигнал с левого регистра. В эксперименте сигнал, соответствующий напряжению на обкладках конденсатора поступает с платы автоматизированного сбора данных, подключенной к лабораторной установке. В задании 1 студенты моделируют виртуальное падение напряжения с обкладок конденсатора, а демонстрационный вариант работы с заданием 2, платой автоматического сбора данных и лабораторной установкой находится у преподавателя.

Для моделирования экспериментального сигнала из Палитры арифметических функций выбираем пиктограмму экспоненциальной функции e^x и переносим ее внутрь цикла. В качестве показателя экспоненциальной функции принимаем число i, инвертированное до отрицательного числа и деленное на 10. Выход экспоненты умножаем на величину ЭДС и вводим формульный узел в качестве текущего значения U.

1.5. Создать на правой границе узла два выхода I и Q, сигналы с которых вывести на входы второго и третьего осциллографов. Для обеспечения операции суммирования заряда продублировать соединение Q с правым регистром. То же выполнить для напряжения, подведя соответствующий сигнал к пиктограмме вольтметра и первому осциллографу.

1.6. Текущее значение U используется в качестве входного параметра системой автоматического отключения работы цикла. Условие выключения $-U \leq U_k$. Для его реализации вызывать с *Палитры всех функций* подпалитру *Логические элементы*, в ней – узел «≤», и ввести на верхний терминал узла текущее значение U, а на нижнее – заданное значение остаточного напряжения U_k от элемента управления, созданного ранее на *Лицевой панели*. Результат *Falshe* или *True* присоединить к терминалу завершения цикла проводником зеленого цвета, который окрашивается автоматически при появлении логического сигнала.

Так как по умолчанию каждый цикл рассчитывается за одну миллисекунду, то для отслеживания динамики процесса установить задержку цикла, равную 100 мс. Для этого на *Панели всех функций* выбирать приборы времени, в ней пиктограмму метронома. Поместить его внутри цикла, найти входной терминал и щелчком ПКМ вызвать всплывающее меню. В нем выбрать команды Create→Const. В появившемся прямоугольнике с клавиатуры набирать число 100.

1.7 Значения Q и t, сохраненные в цикле после выполнения последней итерации, выводятся на цифровые индикаторы величины исходного заряда и времени разрядки. Кроме того, вне цикла, после его выполнения, рассчитывается емкость конденсатора: C=Q/U0.

1.8 Убедитесь, что стрелка запуска цикла имеет правильную, не изломанную форму. Это означает, что программа составлена правильно и готова к запуску. В противном случае – щелчком ПКМ по стрелке вызываем контекстное меню с распечаткой допущенных ошибок. Устраняем их и запускаем программу.

При этом на первых двух осциллографах строятся кривые падения напряжения на обкладках конденсатора и соответствующего уменьшения тока через сопротивление как показано на рисунке 4.3. На третьем осциллографе демонстрируется подсчет по времени величины суммарного заряда, сошедшего с обкладок конденсатора. В конце цикла на цифровых индикаторах появляются значения заряда - Q, емкости конденсатора – C и времени разрядки - t. Провести моде-

лирование процесса при различных значениях сопротивления резистора в цепи разрядки, занести эти значения в таблицу 4.1 и построить график зависимости T=f(R).

Таблица 4.1 - Результаты измерений

N п/п	R, Ом	t, с	Q, Кл	C, мкФ
1	1000			
2	2000			
3	3000			
4	500			

Порядок выполнения задания 2

1. Рассмотреть лицевую панель и блок-диаграмму экспериментальной работы (рисунки 4.5 и 4.6) и отметить общие узлы, структуры и наиболее существенные отличия от моделирующей программы.
2. Собрать установку, как показано на рисунке 4.2, соединив источник питания, ключ, конденсатор и резистор. Подключить источник питания и плату автоматизированного сбора данных.
3. Подключить датчик напряжения к контактам конденсатора и плате сбора данных.
4. Подключить плату сбора данных к USB разъему компьютера.
5. Включить компьютер и вызвать программу лабораторной работы «Condensator».
6. Перевести ключ в положение «зарядка конденсатора» и запустить программу измерений.
7. Через 3-5 секунд перевести ключ в режим разрядки конденсатора. Обратить внимание, что при разрядке напряжение на обкладках конденсатора и ток через сопротивление резистора падают, а прошедший через сопротивление нагрузки суммируемый заряд увеличивается.
8. Программа измерения отключается автоматически при достижении нулевого уровня (≤0,1 В) напряжения на обкладках конденсатора. При этом на индикаторах отображаются величина заряда конденсатора, его емкость и время разрядки, которые заносятся в таблицу 4.2.

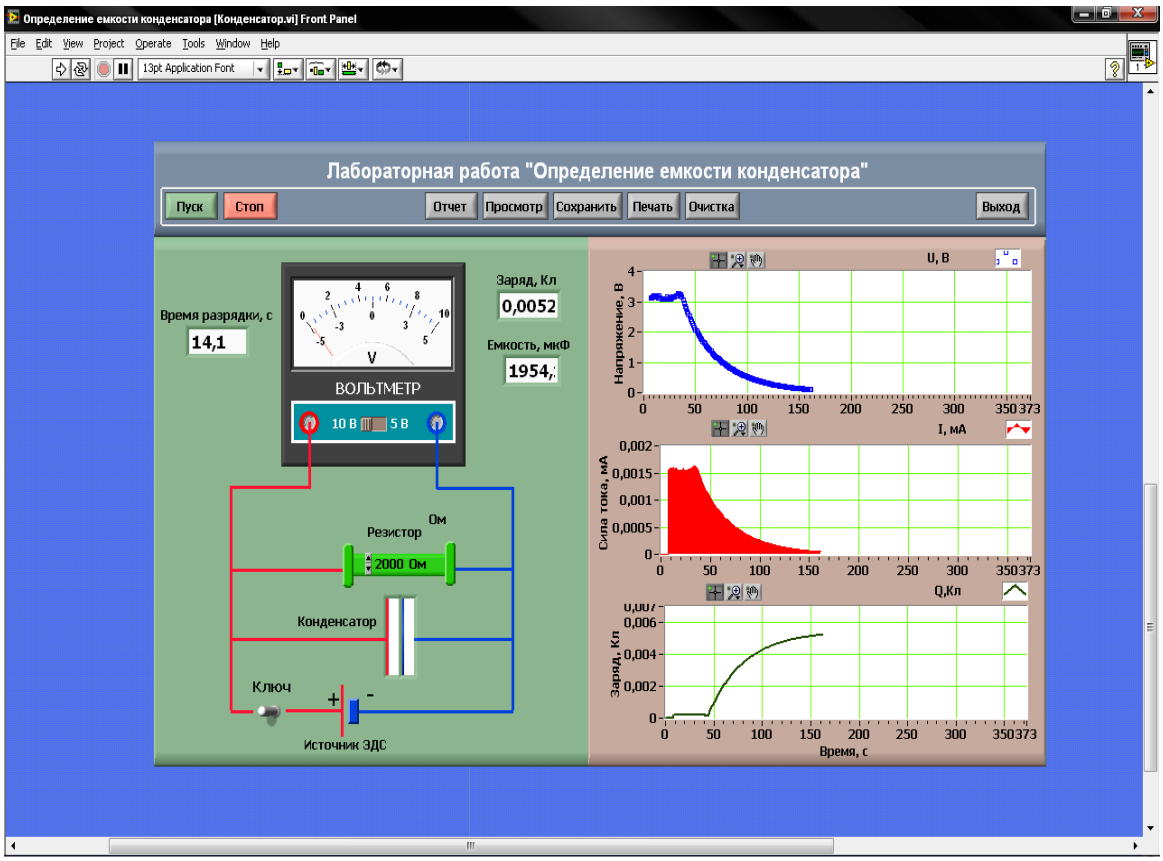


Рисунок 4.5 - Лицевая панель компьютерной системы измерения

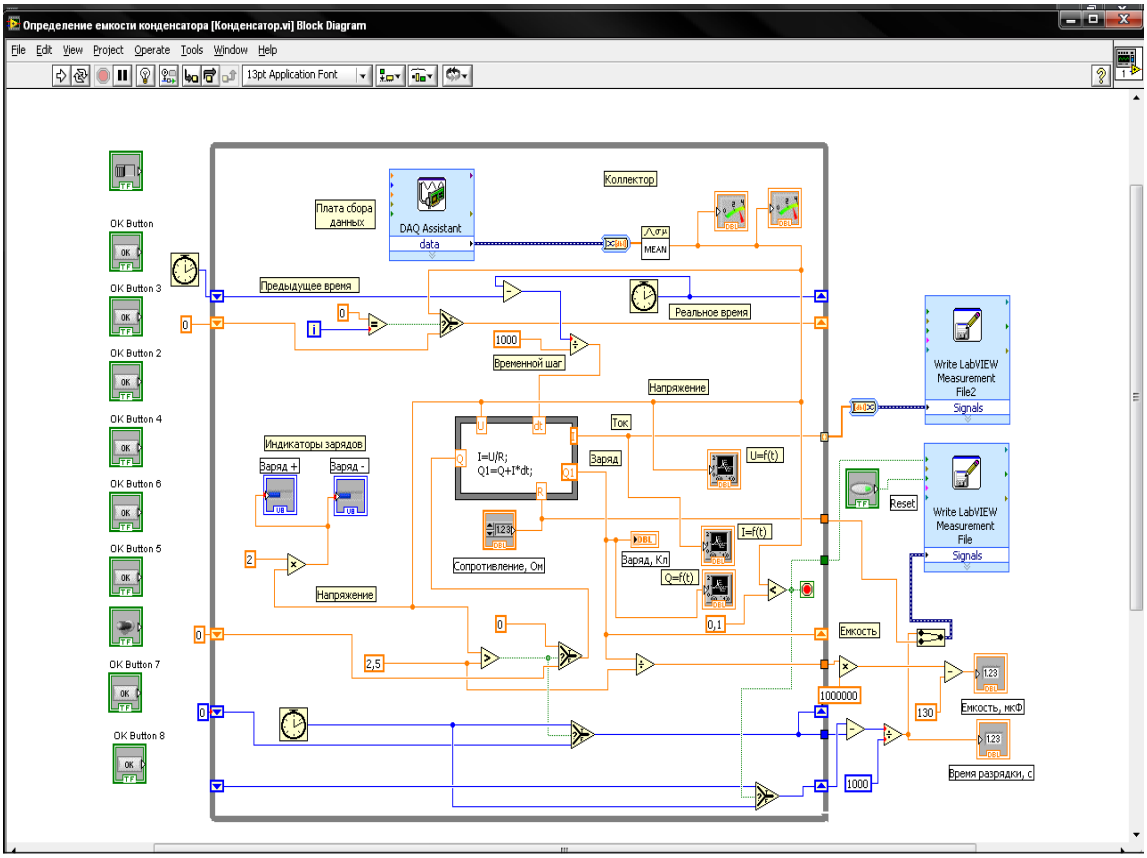


Рисунок 4.6 - Графический код программы по определению емкости конденсатора

9. После автоматического выключения измерений сравнить полученные данные с результатами компьютерного моделирования.

10. Ответить на контрольные вопросы и сделать самостоятельные выводы по проведенной работе.

Таблица 4.2 - Результаты измерений

№ опыта	Сопротивление резистора, Ом	Заряд, Кл	Емкость конденсатора, мкФ	Время разрядки, с
1				
2				
3				
4				

Выводы

1. Составлена программа моделирования лабораторной работы по определению емкости конденсатора методом суммирования количества элементарных зарядов, стекающих с его обкладок.

2. В качестве структур, обеспечивающих выполнение работы, использован узел формул и сдвиговый регистр, позволяющий интегрировать значения силы тока в цепи разрядки конденсатора по времени.

3. Для повышения точности определения емкости конденсатора, предусмотрено автоматическое выключение измерений в тот момент, когда напряжение на его обкладках снизится до 0,1 В.

4. Проведено сравнение результатов выполненных вычислений с экспериментальными данными. Показано, что моделирование позволяет установить общие закономерности процесса разрядки, а разработанная программа обработки данных может полностью использоваться в качестве самостоятельной подпрограммы автоматизации реального физического эксперимента.

Контрольные вопросы

1. Какие новые элементы и структуры среды LabVIEW использованы в настоящей работ?

2. Для чего выполнение программы измерений и интегрирования зарядов прекращается при остаточном напряжении на обкладках конденсатора 0,1 В? Каким будет результат подсчета суммарного заряда при уменьшении порога отключения до 0,01 В?

3. В чем состоят основные отличия программ моделирования и автоматизации реального эксперимента?

4. Каковы характеристики использованной в эксперименте платы сбора данных?

5. Какие дополнительные возможности кроме автоматизации сбора и обработки данных обеспечивает использованная плата?

6. Назовите величину элементарного заряда?

7. В каком случае два проводящих тела образуют конденсатор?

8. Что называется зарядом конденсатора?

9. От каких величин зависит время разрядки конденсатора?

10. Определите энергию электростатического поля испытанного конденсатора, заряженного 3,5 В.

ЛАБОРАТОРНАЯ РАБОТА № 5

ФУНКЦИИ ГЕНЕРАЦИИ, ВВОДА И ОБРАБОТКИ ДАННЫХ В LABVIEW

Цель работы: Изучение функций LabVIEW для ввода и обработки данных во временной и частотной области

Задание 1. Создать генератор сигналов заданной длительности с с числом выборок 50, частотой 10 Гц, амплитудой равной 2, постоянным смещением, равным 1 и начальным значением фазы равным $\pi/2$. Подать выходной сигнал на однолучевой осциллограф, сохранить в Excel, построить график и описать свойства сигнала.

Задание 2. Использовать генератор, созданный по заданию 1 для моделирования входных сигналов системы регулирования. Создать систему и программу ее тестирования. Добавить к входному сигналу случайный гауссовский шум с амплитудой равной 1, запустить программу и проанализировать изменение спектра выходного сигнала.

Основные положения

LabVIEW предоставляет широкий набор функциональных возможностей для отладки сложных программ, тестирования реальных систем измерения и регулирования и развернутого анализа получаемых данных. Так функции генерации сигналов и шумов используют-

ся для формирования детерминированных и случайных сигналов с заданным набором параметров (рисунок 5.1).

Первые два прибора в верхнем ряду представляют многофункциональные программно регулируемые генераторы сигналов с широким набором контролируемых параметров. Приборы, размещенные во второй и третьей строках, предназначены для генерации наиболее широко применяемых детерминированных периодических сигналов, а находящиеся в четвертой и пятой строках - для генерации шумов с различными законами амплитудного и спектрального распределения.

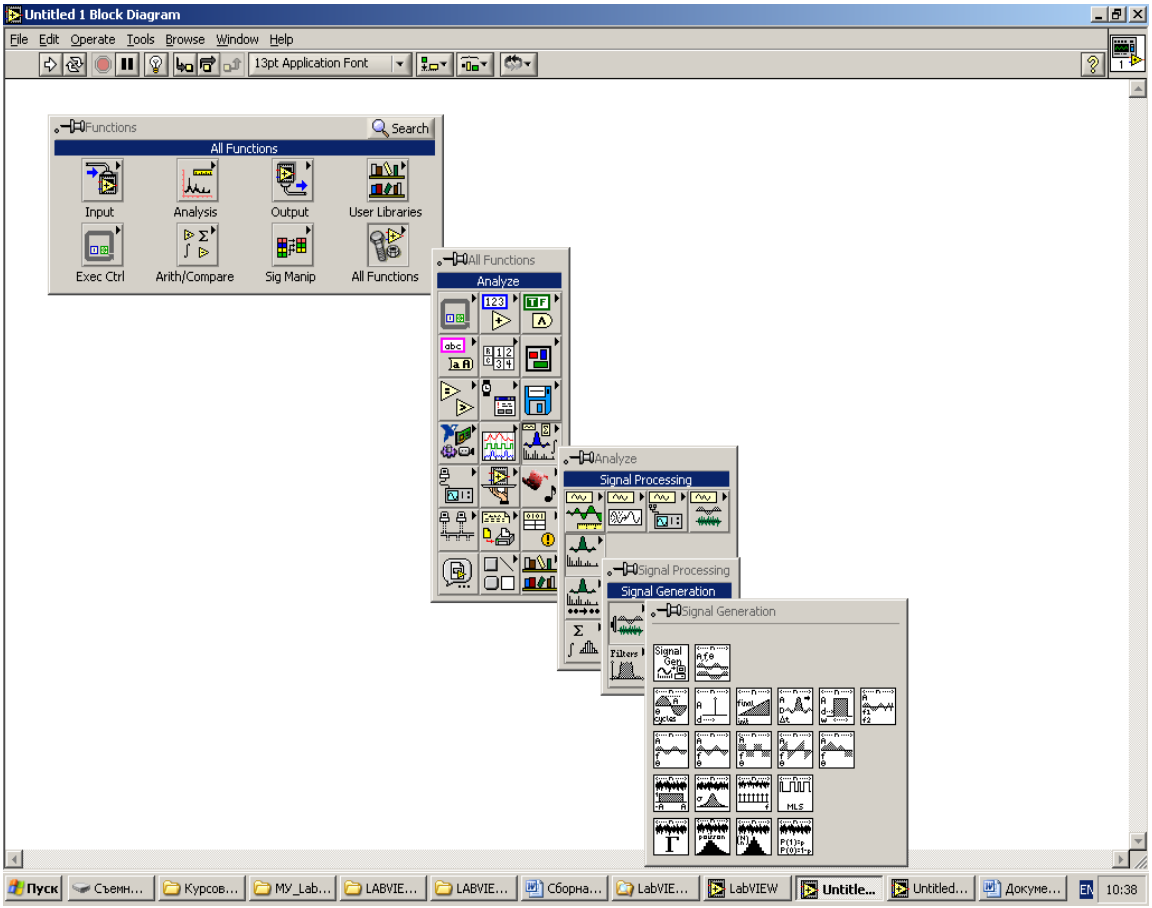


Рисунок 5.1 - Палитра функций генерации сигналов и шумов

Среди них:

- генератор с заданной длительностью сигналов;
- гармонические колебания и шум;
- отрезки синусоидального, импульсного, пилообразного, sin(x)/y, прямоугольного и частотно-модулированного сигналов;
- синусоидальные, треугольные, прямоугольные, пилообразные и произвольные колебания любой длительности;
- равномерный, гауссовский, периодический случайный шумы и двоичная последовательность максимальной длины;

- гамма-шум, пуассоновский, биноминальный шумы, шум Бернулли.

Порядок выполнения задания 1

1.1 В *Палитре всех функций* на блок-диаграмме находим *«Генератор сигналов с заданной длительностью»*: Function→ All Function→ Analyze→ Signal Processing→ Signal Generation. Для получения справки по какому-либо виртуальному прибору надо активизировать его изображение при помощи левой кнопки мыши и выбрать пункт меню Help (рисунок 5.2).

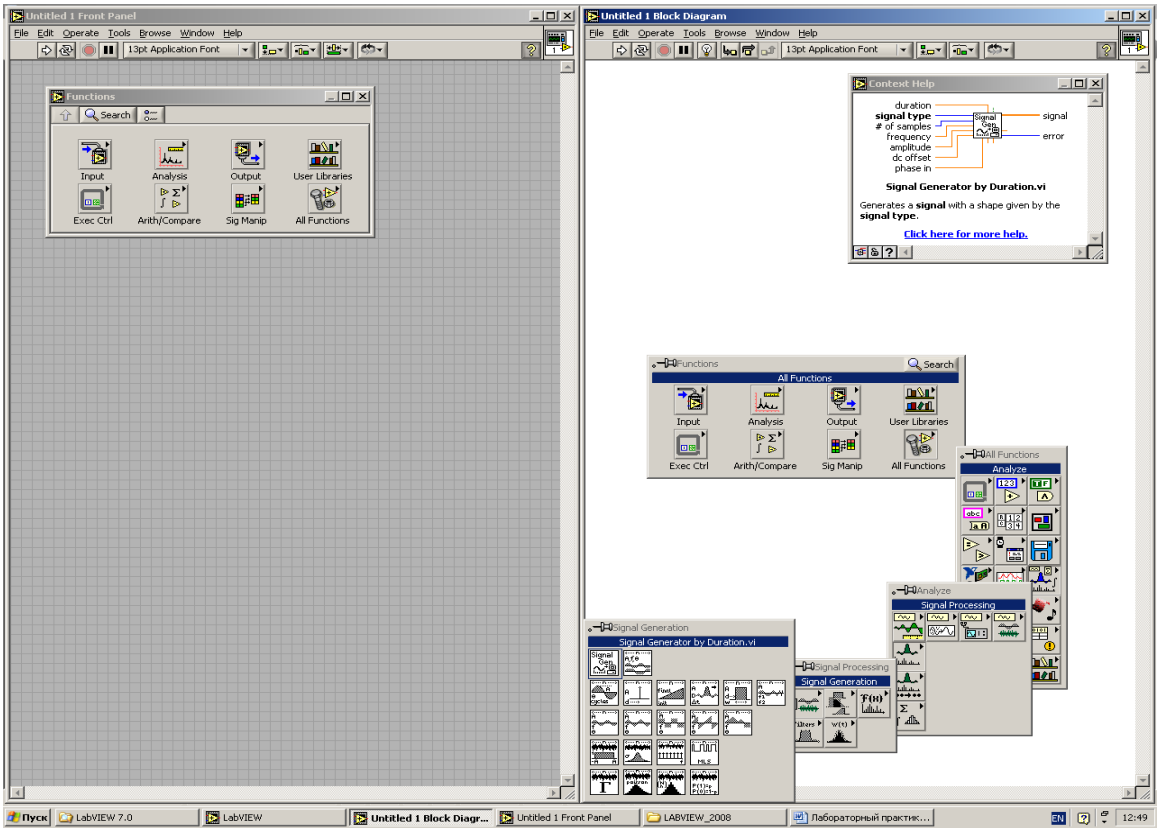


Рисунок 5.2 - Выбор генератора сигналов с заданной длительностью

Рассматриваемый виртуальный прибор (ВП) имеет девять входов и три выхода. Входы: сбросить фазу, длительность, тип сигнала, число выборок, частота, амплитуда, постоянная составляющая, вход фазы, заполнение цикла прямоугольного колебания (%). Выходы: сигнал, частота выборок, выход фазы (рисунок 5.3).

ВП генерирует сигнал (signal), имеющий форму, задаваемую на входе *тип сигнала* (signal type). Вход *сбросить фазу* (reset phase) определяет начальную фазу выходного сигнала. По умолчанию на

входе установлено состояние ИСТИНА. При этом начальная фаза сигнала устанавливается в соответствии со значением на входе *вход фазы* (phase in). Если на входе *сбросить фазу* установлено состояние ЛОЖЬ, то начальная фаза устанавливается равной значению фазы на *выходе фазы* (phase out) при последнем выполнении этого ВП.

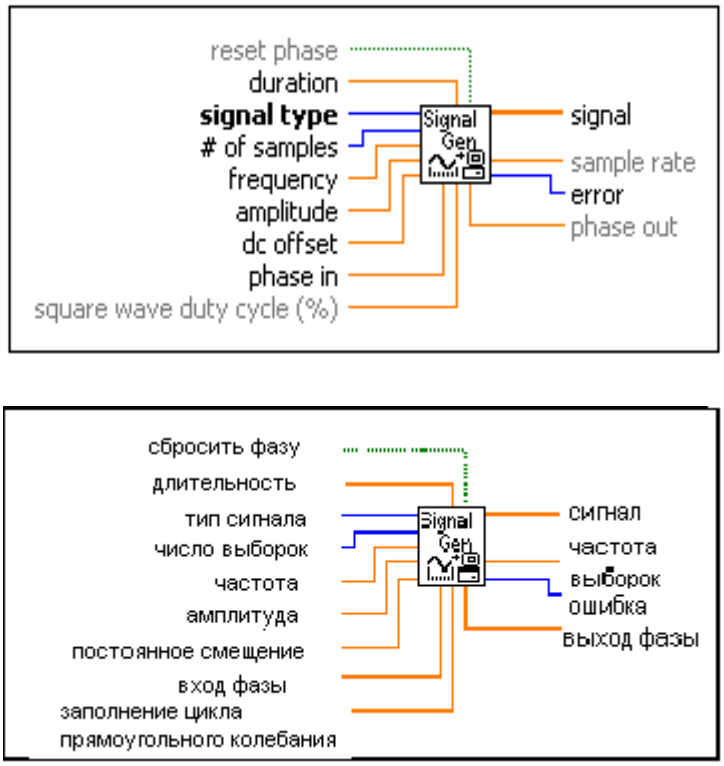


Рисунок 5.3 - Генератор сигналов с заданной длительностью

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

Вход *тип сигнала* задает следующие типы генерируемого сигнала: 0-синусоидальный, 1-косинусоидальный, 2-треугольный, 3-прямоугольный, 4-пилообразный, 5-линейно нарастающий, 6-линейно спадающий.

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

Вход *заполнение цикла прямоугольного колебания* (square wave duty cycle) определяет время (в % 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* представляет сгенерированный массив выборок сигнала.

Выход *частота выборок* (sample rate) отображает частоту дискретизации выходного сигнала. *Частота выборок* равна отношению числа выборок к длительности. *Выход фазы* указывает значения фазы (в градусах) последней выборки выходного сигнала.

1.2 Подать выходной сигнал на однолучевой осциллограф. Запустить программу и описать свойства сигнала.

1.3 Добавить шум.

1.4 Посмотреть спектр шума и сигнала. Определить смещение (рисунки 5.4, 5.5).

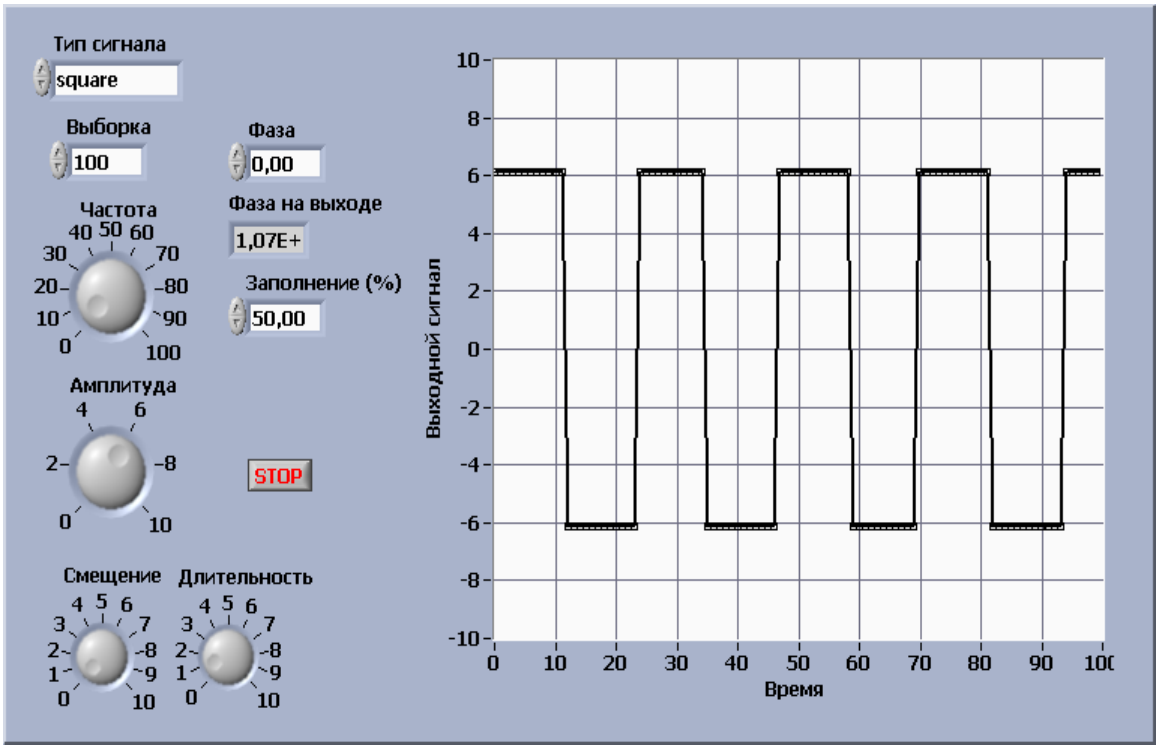


Рисунок 5.4 – Лицевая панель генератора сигналов

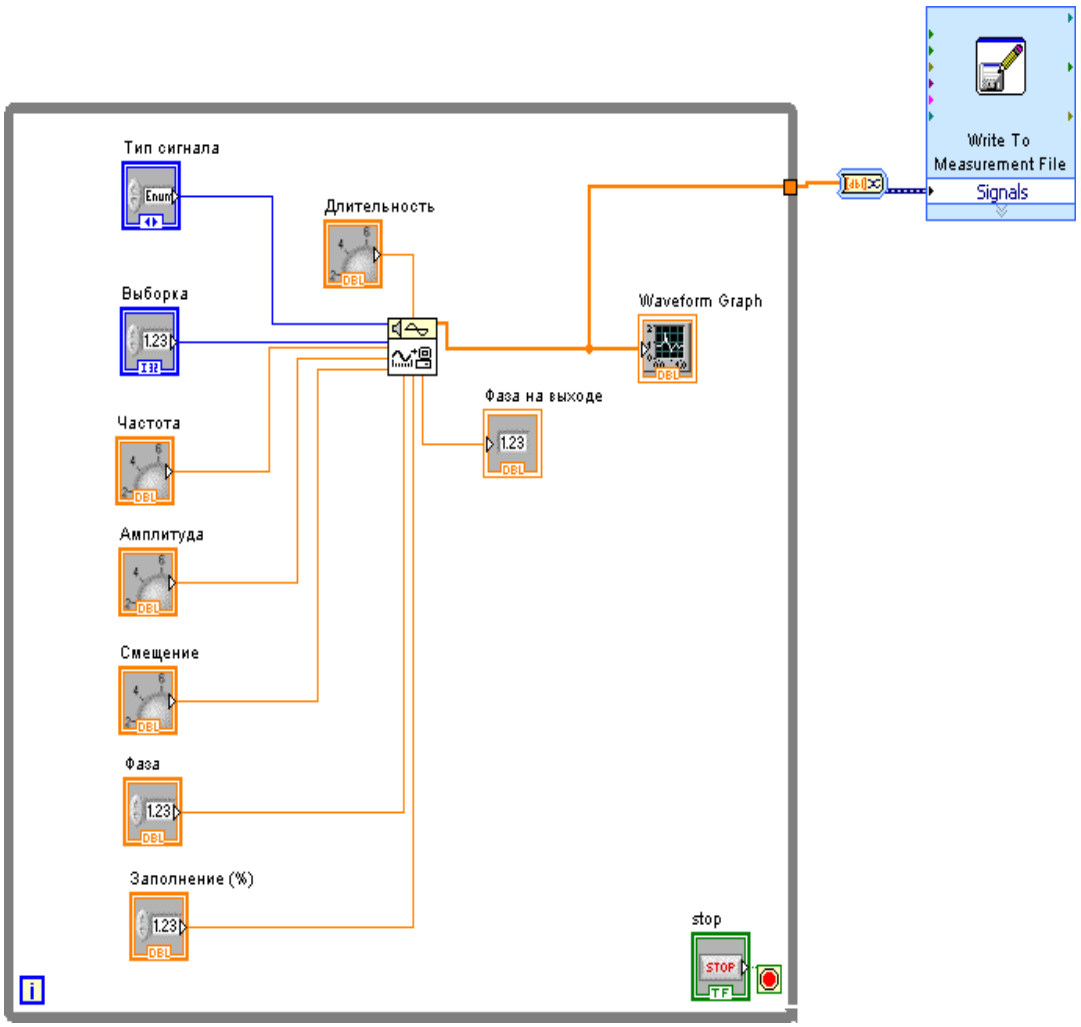


Рисунок 5.5 – Блок-диаграмма программного управления режимом работы генератора

1.5 Сохранить полученные данные в Excel, построить график и описать свойства сигнала.

Порядок выполнения задания 2

2.1 Создать систему регулирования, формирующую управляющий сигнал на отключение оборудования при достижении значения генерируемой функции 0,75 ее максимума.

Для выполнения задания надо выбрать *Палитру функций обработки сигналов в частотной области* (Function→ All Function→ Analyze→ Signal Processing→ Frequency Domain→ Transfer Function), которая содержит наборы функций, позволяющих выполнить прямое и обратное преобразования Фурье, Гильберта, Хартли и Уолша-Адамара, а также прямое и обратное вейвлет-преобразование (рисунок 5.6).

На базе функций преобразования Фурье разработан ряд высокоуровневых приборов для оценки взаимного спектра мощности, импульсной и частотной передаточной характеристик цепей, частотной функции когерентности и измеритель гармонических искажений сигнала, размещенные на данной палитре.

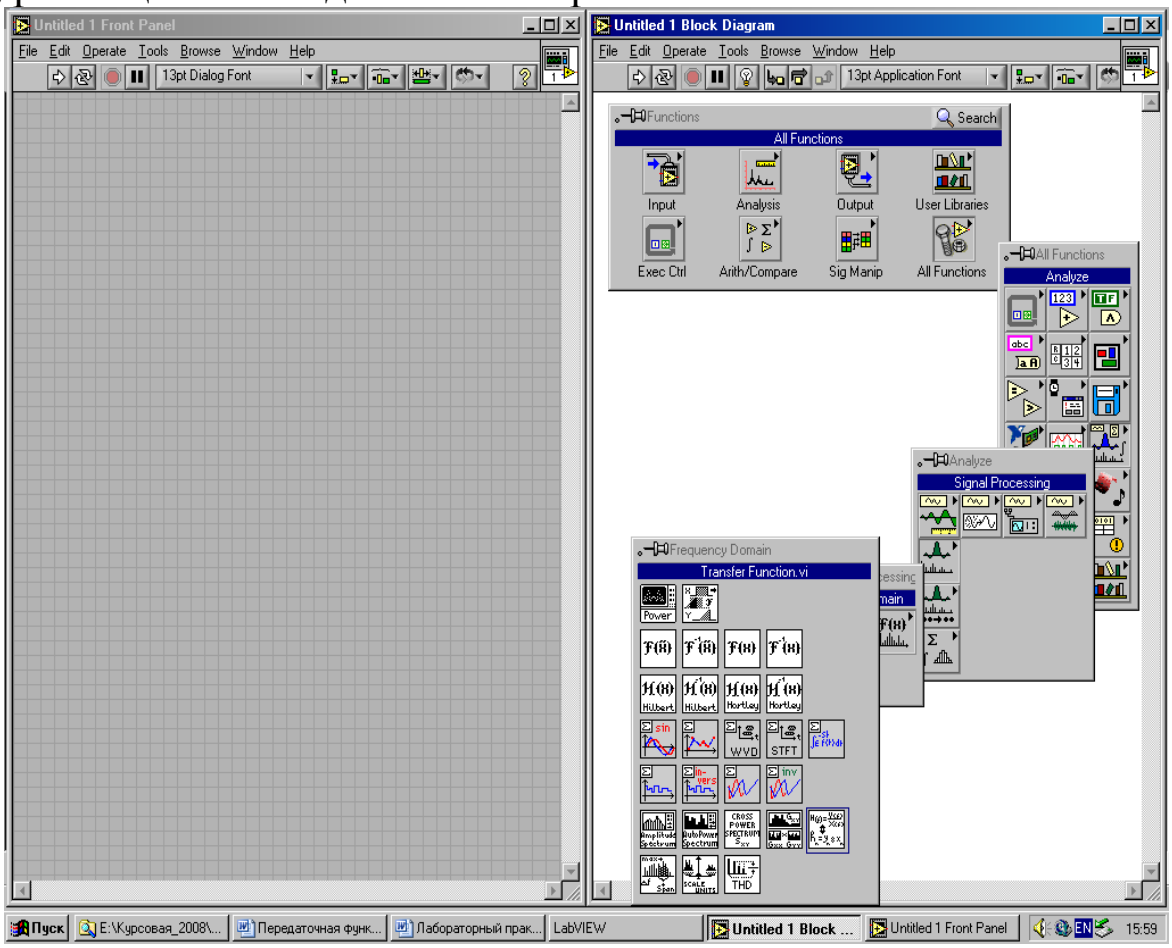


Рисунок 5.6 - Палитра функций обработки сигналов в частотной области

Виртуальный прибор (ВП) *Передаточная функция* (Transfer function) производит расчет односторонней передаточной функции, также известной как частотная передаточная функция, на основе анализа заданных во временной области тестирующего сигнала (Stimulus Signal) и выходного сигнала тестируемого объекта (Response Signal) на входе и выходе тестируемой электрической цепи. Блок-диаграмма приведена на рисунке 5.7.

2.2 Добавить к схеме генератора гауссовский шум со стандартными параметрами. Для этого на предыдущей блок-диаграмме вызвать Виртуальный прибор -  - Гауссовский белый шум (Gaussian White Noise), который генерирует псевдослучайную последователь-

ность с гауссовским (нормальным) распределением с параметрами: *выборка (samples), стандартное отклонение (standart deviation), начальное значение (seed).*

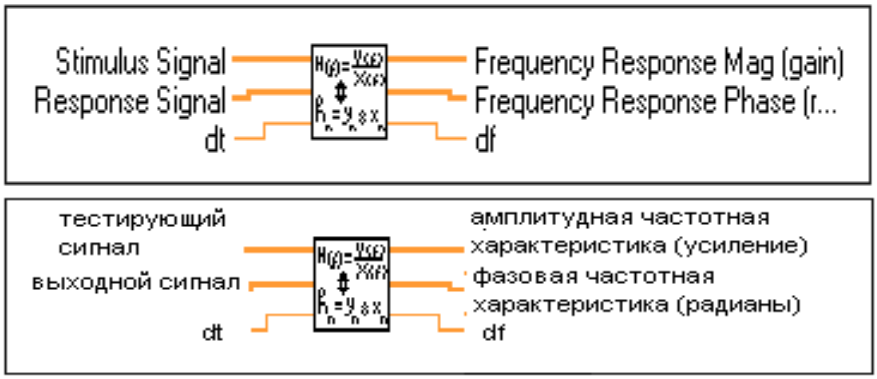


Рисунок 5.7 - Входы и выходы пиктограммы расчета передаточной функции

Гауссовский шум описывается следующей функцией плотности вероятностей:

$$f(x) = \frac{1}{\sqrt{2*\pi}} * e^{-\frac{1}{2}*\left(\frac{x}{s}\right)^2}$$

По умолчанию значение стандартного отклонения равно 1,0.

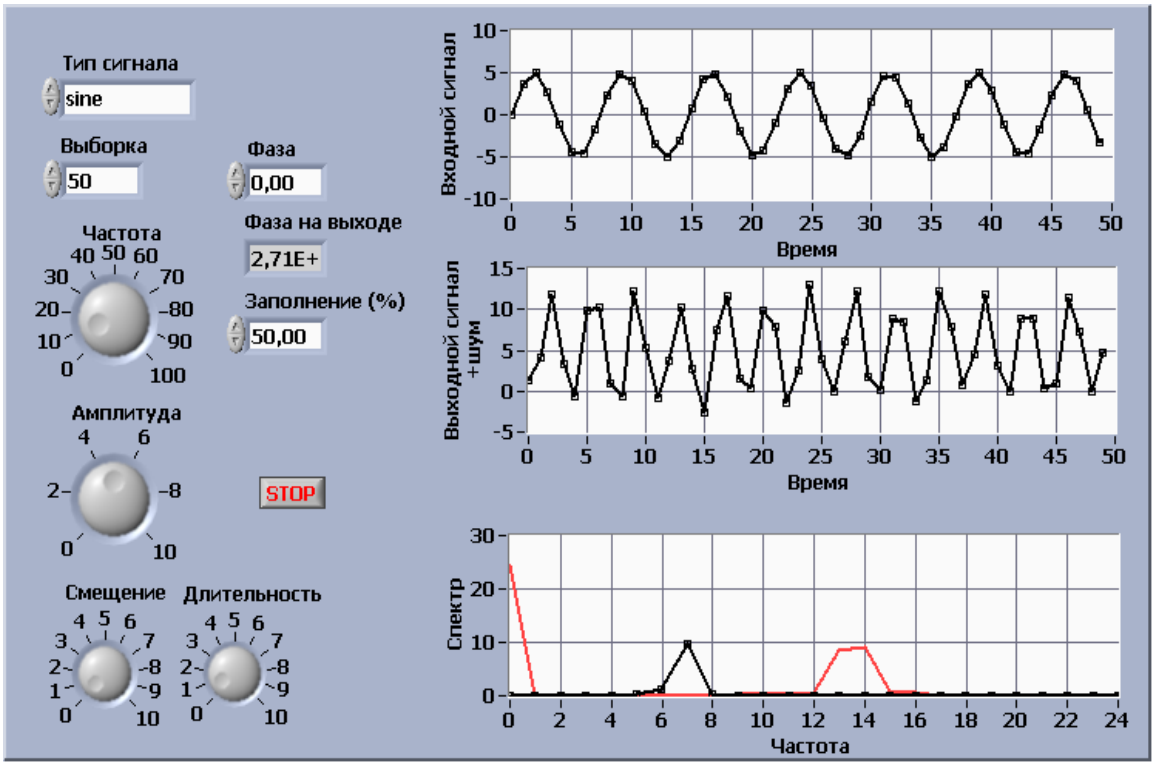


Рисунок 5.8 - Лицевая панель системы регулирования и блок-диаграмма тестирования

2.3 Создать программу его тестирования. Прибавить к сигналу генератора случайный гауссовский шум с амплитудой равной 1. Подать результирующий сигнал на вход анализатора спектра. Запустить программу и проанализировать реакцию системы регулирования (рисунки 5.8, 5.9).

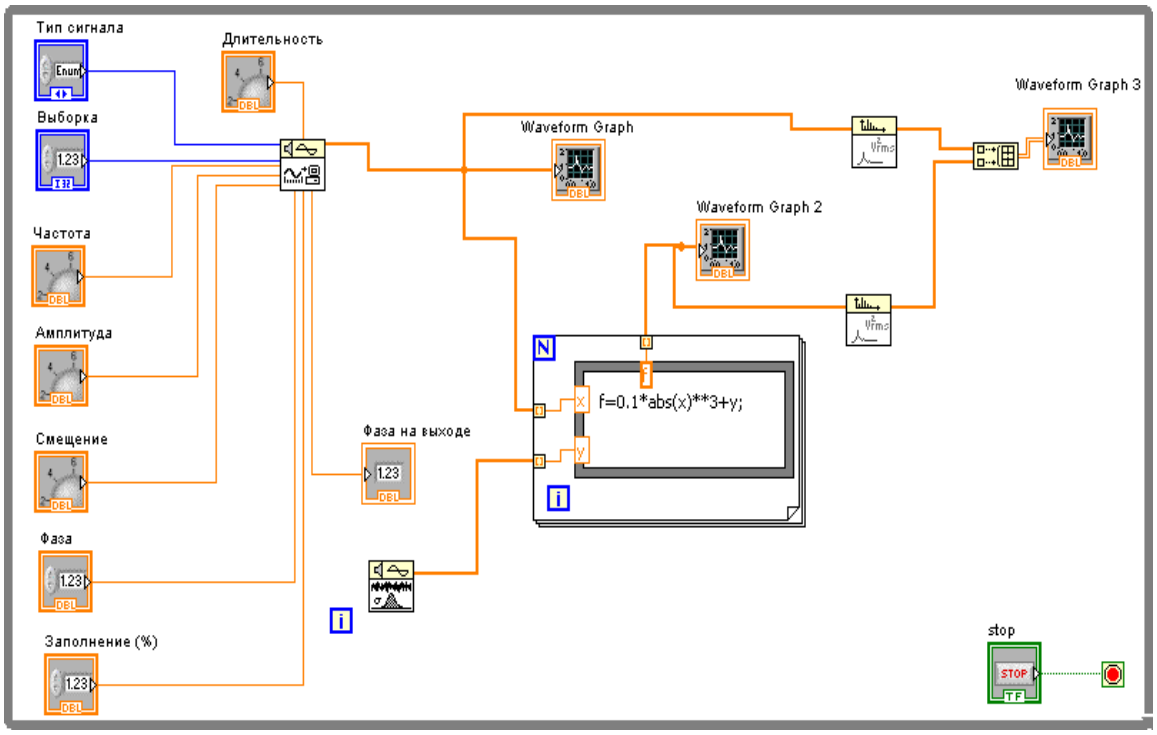


Рисунок 5.9 - Блок-диаграмма тестирования выходного сигнала

Контрольные вопросы:

1. Для каких целей в LabVIEW используются функции генерации шумов и сигналов?
2. Какие свойства тестируемого объекта характеризует передаточная функция?
3. С помощью контекстной справки определите обязательные, рекомендуемые и опциональные входы и выходы генератора сигнала с заданной длительностью.
4. Опишите спектр выходного сигнала.

ДОПОЛНИТЕЛЬНЫЕ СВЕДЕНИЯ

NI LabVIEW - единая платформа для управления, измерений и моделирования

Уже почти 20 лет инженеры и ученые используют среду графического программирования National Instruments LabVIEW для создания автоматизированных систем сбора данных и управления приборами, которые нашли применение как в исследовательских и испытательных лабораториях, так и на технологических производственных линиях. Все это время среда LabVIEW постоянно совершенствовалась - благодаря регулярному выходу новых версий, а также выпуску специализированных модулей, библиотек и дополнений, обусловленных пожеланиями пользователей и исследовательской работой коллектива разработчиков LabVIEW, и фактически стала стандартом в ряде областей науки и техники. Согласно своей фундаментальной идее, LabVIEW позволила инженерам, не имеющим глубоких знаний и опыта в традиционном программировании, быстро создавать сложные автоматизированные системы измерений и управления. Но в своем развитии LabVIEW стала больше, чем просто языком программирования. LabVIEW предоставляет пользователю широкую гамму инструментов, которые образуют графическую платформу разработки для моделирования, управления и тестирования. В данной статье кратко рассматриваются инструментальные средства и библиотеки, которые продвигают платформу LabVIEW в новые, все более перспективные отрасли промышленности и на новые сегменты рынка высоких технологий.

Проектирование и разработка на единой платформе

В связи с бурным развитием технологий за последние 20 лет, включая увеличение производительности полупроводниковых приборов и уменьшение их размеров в соответствии с законом Мура, повсеместным внедрением компьютеров и микропроцессоров, развитием стандартов связи и сетевых технологий, инженеры были вынуждены в равной степени увеличивать сложность процессов разработки, производства и тестирования новых продуктов.

LabVIEW имеет множество преимуществ в различных областях разработки приложений и отраслях промышленности. Кроме этого, компания National Instruments дополнила среду программирования внедрением целого семейства дополнительных модулей и библиотек для расширения круга решаемых задач. Данная платформа полностью перекрывает потребности трех базовых областей применений:

- автоматизированные системы измерения и тестирования;
- промышленные системы контроля и управления;
- проектирование и отладка встраиваемых систем.

Автоматизированные системы измерения и тестирования на платформе LabVIEW

Для приложений автоматизированного тестирования LabVIEW предоставляет широкий набор средств для ввода и вывода сигналов с различного аппаратного обеспечения, а также функции специализированного анализа, необходимые для проведения измерений в различных областях. Кроме этого, платформа содержит целый спектр инструментов для задач автоматизации и обработки данных:

- Интерактивные измерения. С помощью пакета NI Signal Express вы можете интерактивно использовать виртуальные приборы (встраиваемые либо автономные приборы, управляемые с компьютера) для проведения необходимых вам измерений и анализа сигналов. Для проведения оценочных, быстрых и простых измерений, когда даже простейшее программирование избыточно, NI Signal Express помогает за считанные минуты сконфигурировать процедуру измерения, сравнить результаты с данными моделирования и сохранить их на компьютере.

- Автоматизированные системы испытаний. С помощью пакета NI TestStand вы можете разработать структурированную последовательность испытаний, представляющих собой отдельные программы LabVIEW (либо модули других систем программирования), со сложной логикой принятия решений «тест прошел/не прошел» для управления общим ходом испытания. Кроме того, NI TestStand легко интегрируется в единые информационные системы предприятия (ERP) для предоставления результатов в базы данных либо для отслеживания испытуемых изделий через автоматизированные системы управления производством (MES).

Для проведения автоматизированных измерений LabVIEW содержит пакеты анализа, оптимизированные для различных измерительных задач:

- Тестирование линий связи – средства обработки и генерации сложных модулированных сигналов и усовершенствованные функции для проведения спектральных измерений, расширяющие возможности библиотек, содержащихся в базовом комплекте LabVIEW.

- Измерение виброакустических сигналов – модули для исследования динамических акустических сигналов с целью оценки качества звука, или проведения структурных испытаний.

- Мониторинг состояния машин и механизмов – специализированные алгоритмы порядкового анализа вращающихся частей механизмов (вэйвлет-анализ, совместный частотно-временной анализ).

- Обработка изображений – средства для автоматизированного визуального контроля и приложений машинного зрения.

Платформа для промышленных систем измерения и управления на базе LabVIEW

Для создания приложений управления LabVIEW содержит отдельный набор специализированных библиотек, дополняющих графическую платформу методами управления, функциями распределенного мониторинга и управления, АСУТП, а также возможностями управления в реальном времени.

С помощью LabVIEW вы можете использовать единую платформу для разработки и развертывания собственных концепций управления, применяя различные подходы и технологии, такие как:

ПИД-регулирование - Используйте преимущества этой технологии для относительно простого создания промышленных приложений управления.

Расширенное управление - LabVIEW содержит средства разработки алгоритмов оптимального управления на основе признанных моделей контроллер-объект, либо управление на базе принятых/выданных сигналов с учетом усовершенствованной идентификации системы. Кроме того, к LabVIEW поставляется дополнительная библиотека для непрерывного контроля работы динамических систем, позволяющая использовать указанные модели совместно с традиционными функциями управления, такими как передаточные функции, интеграторы, дифференциаторы и цепи обратной связи.

Управление движением – Используется для управления электроприводами и промышленными механизмами.

Повторное воспроизведение опытных сигналов - Это уникальная возможность воспроизводить ранее сохраненные измерительные данные (ход нагрузок в автомобильной технике, сигналы переходных процессов в электронике и связи и т.д.) для проведения модельных испытаний или отладки прототипов изделий и устройств

Модуль LabVIEW Real-Time для промышленных платформ является идеальным выбором для реализации алгоритмов управления в производственных системах. Тем не менее, ряд пользователей из отраслей проектирования машин и промышленного управления считают, что расширение возможностей LabVIEW для программирования ПЛИСов, интегрированных в узлы ввода/вывода, является ещё более надежным методом внедрения управляющих алгоритмов. Если вы запрограммируете функциональность оборудования через встроенную ПЛИС, то это оборудование оказывается гораздо более защищенным и надежным в производственном процессе. Обеспечение тесной взаимосвязи между программированием встраиваемых ПЛИС и приложениями промышленного управления с помощью интуитивно понятного подхода графического программирования является уникальным преимуществом использования LabVIEW.

Для разработки распределенных систем мониторинга и управления LabVIEW имеет системные возможности более высокого уровня, такие как занесение информации в базу данных, алгоритмы принятия решений, обеспечение безопасности.

Платформа для разработки и отладки встраиваемых систем на базе LabVIEW

Разработчики в основном знакомы с LabVIEW как со средством проведения измерений. Однако LabVIEW продолжает приобретать популярность и как инструмент создания универсальных алгоритмов для инженеров и ученых во многих сферах деятельности. Сочетание развитых библиотек для обработки сигналов и управляющих алгоритмов с готовыми к использованию инструментальными средствами позволяет быстро проектировать, создавать прототипы и разворачивать системы с помощью LabVIEW. Ниже приведены некоторые ключевые свойства платформы LabVIEW, используемые при проектировании систем.

Обширная библиотека анализа и математических функций. LabVIEW содержит сотни математических функций, охватывающих широкий спектр традиционных алгоритмов в областях математического анализа, обработки сигналов, вероятности и статистики, систем управления, представляющих собой основу любого пользовательского алгоритма.

Естественная интеграция с устройствами ввода/вывода – Поскольку реальные физические данные очень легко получить с помощью LabVIEW, вы, несомненно, оцените удобство проверки и отладки созданных алгоритмов на примере реальных данных.

Аппаратные платформы для создания систем реального времени - алгоритмы LabVIEW можно выполнять на платформах реального времени с интегрированным вводом/выводом. С помощью модульных аппаратных средств National Instruments CompactRIO и PXI можно быстро создавать прототипы встраиваемых систем, использующих процессоры, ПЛИС для встроенной логики и широкий спектр оригинальных устройств ввода/вывода.

Одна платформа, множество приложений

Во многих областях современной промышленности, начиная от исследовательских лабораторий до конструкторских бюро, создающих распределенные и встраиваемые системы, платформа графической разработки LabVIEW увеличивает производительность труда инженеров и ученых. Сочетание интуитивно понятного графического языка программирования, поддержки широкого набора устройств ввода/вывода и растущего сообщества пользователей, участвующих в развитии платформы LabVIEW, делает успешным создание принципиально новых приложений. Переходя на более эффективный графический принцип разработки, однако продолжая использовать открытую среду программирования LabVIEW для воплощения разработанных алгоритмов и обмена данными со средствами моделирования, можно модернизировать средства разработки и сократить временные затраты на всех этапах жизненного цикла изделий.

Чтобы узнать, как уже сегодня начать разрабатывать приложения для распределенных интеллектуальных систем в LabVIEW 8, позвоните в филиал корпорации National Instruments Russia по телефону +7 (095) 783-6851 или посетите **ni.com/russia**.

Единая платформа LabVIEW 8: возможности распределенного интеллекта для решения задач управления, измерений и проектирования

Что первое приходит Вам на ум, когда Вы слышите о *распределенной* системе? Многопроцессорная система, параллельно обрабатывающая несколько задач? Или электронная платежная система для работы с заказами со всего мира? Сеть беспроводных датчиков, отслеживающих состояние «умного дома»? Все эти примеры объединяет общая идея – распределение ресурсов системы для решения поставленной задачи. Благодаря повышению производительности современных микроэлектронных устройств при одновременном снижении их стоимости, инженеры и ученые нашли эффективные решения сложнейших задач путем добавления процессоров и «интеллектуальных» микропроцессорных компонентов в свои системы. И, как следствие, современные контрольно-измерительные устройства и системы все больше становятся распределенными.

Тем не менее, разработка измерительных и управляющих систем, включающих несколько вычислительных узлов, не так проста, как может показаться на первый взгляд. При программировании распределенных систем специалисты столкнулись с целым классом новых проблем, к решению которых существующие средства разработки оказались не вполне пригодны. National Instruments LabVIEW 8 представляет новую технологию «распределенного интеллекта», ориентированную именно на этот класс задач, и включающую в себя следующие средства:

- Возможность программирования нескольких целевых платформ, таких как персональные, промышленные, портативные и встраиваемые компьютеры;
- LabVIEW Project – новая интерактивная среда управления распределенными системами из единой программной оболочки;
- LabVIEW Shared Variable – новый коммуникационный интерфейс, позволяющий упростить обмен данными между различными устройствами и программами, входящими в распределенную систему;
- Средства синхронизации и тактирования как устройств, входящих в распределенную систему, так и нескольких распределенных систем.

Распределенные системы проектирования, управления и измерений

Создание распределенных систем требует новых, оригинальных подходов к программированию. Например, беспроводные датчики (wireless sensors) образуют самоорганизующуюся сеть, узлы которой самостоятельно устанавливают связь друг с другом. Очевидно, что специалисты, работающие с такой технологией, столкнутся с совершенно новыми проблемами в области программного обеспечения. И хотя некоторые проблемы возникают только при реализации конкретных систем, многие инженеры и ученые уже сейчас начинают испытывать схожие трудности при программировании распределенных систем. В качестве примеров можно привести системы испытания автомобильной электроники, смартфоны, комплексы технического зрения и промышленного мониторинга, а также комплексы синхронизированных автоматизированных тестовых станций.

Проблемы разработки распределенных систем

Вы встретите распределенные системы в самых различных отраслях промышленности, на различных фазах жизненного цикла изделий, тем не менее, всем приложениям, использующим такого рода системы присущи схожие сложности:

- Программирование приложений, использующих многопроцессорную архитектуру, в том числе и смешанную – с микропроцессорами, ПЛИС и цифровыми сигнальными процессорами;
- Эффективный обмен данными между несколькими процессорами, расположенными как непосредственно на одной печатной плате, так и внутри единого инструментального шасси или объединенными через сеть;
- Объединение всех узлов в завершенную систему, с решением задач тактирования и синхронизации составляющих ее узлов;
- Интеграция в единой системе различных типов ввода-вывода, таких как высокоскоростной аналоговый или цифровой, а также техническое зрение и управление движением;
- Добавление иных сервисных функций по обмену данными между узлами, включая протоколирование, выдачу сигналов тревог и взаимодействие с информационными системами корпоративного уровня.

Использование новых возможностей NI LabVIEW 8 позволяет разрешить большинство из вышеперечисленных проблем.

Программирование распределенных систем с вычислительными узлами разного типа

Распределенные системы обычно состоят из узлов, выполняющих различные функции – датчиков, приборов, автономных подсистем. Все эти узлы так или иначе взаимодействуют с главной системой, которая осуществляет управление, мониторинг и протоколирование данных. В настоящее время разработчикам распределенных систем приходится пользоваться разными средствами для программирования различных узлов. Более того, доступное на рынке стандартное оборудование не всегда может удовлетворить специфическим требованиям к системе. Поэтому для реализации особых алгоритмов применяется конфигурируемое аппаратное обеспечение, чаще всего на основе ПЛИС, что влечет за собой ощутимое усложнение разработки и требует более высокой квалификации разработчика в области специальных средств и языков программирования.

LabVIEW 8 призвана разрешить данную проблему, предоставляя единую универсальную среду разработки для программирования разнотипных узлов. Используя LabVIEW, Вы создаете код, который может выполняться на таких вычислительных платформах, как персональные компьютеры, устройства реального времени, устройства и подсистемы на базе ПЛИС. В единой оболочке LabVIEW сочетает специфические функции для решения совершенно разнотипных задач, например, функции распознавания образов и классификации объектов для систем автоматизированного видеоконтроля, построение траектории движения для управления электроприводами, измерение аналоговых и цифровых сигналов. Традиционно каждая из этих задач требовала применения отдельных специализированных программных продуктов. LabVIEW также содержит библиотеку расширенного анализа сигналов, а также развитые средства коммуникации с Интернет для удаленного управления и мониторинга.

Способность универсального программного средства преодолеть ограничения стандартной функциональности узла позволяет резко снизить сложность разработки и в той же мере повысить производительность труда инженера-разработчика распределенных систем.

Коммуникации и обмен данными

При создании распределенных измерительно-управляющих систем как правило используются различные средства и протоколы обмена данными. Реализация процедур обмена данными между процессорами, особенно работающими в режиме реального времени и во встраиваемых системах, без снижения производительности их работы, часто представляет собой трудную задачу. И хотя существует множество стандартов и протоколов обмена – например, TCP/IP, Modbus, UDP и OPC – ни один из них сам по себе не в состоянии удовлетворить всем требованиям различных задач. Кроме того, программные вызовы функций (API) различных протоколов отличаются между собой. Поэтому разработчики и системные интеграторы при создании комплексной системы автоматизации вынуждены использовать несколько коммуникационных протоколов. Для обеспечения детерминированного обмена данными между узлами системы часто приходится прибегать к таким дорогостоящим решениям, как использование аппаратно-реализованной «зеркальной памяти» (reflective memory). Одним из способов решения данного класса задач является устранение жесткой привязки определенного транспортного уровня и протокола к его программному вызову (API) в среде разработки. В этом случае вы можете использовать множество протоколов в рамках одного и того же программного кода, тем самым значительно сокращая время разработки и отладки приложения.

Технология распределенного интеллекта LabVIEW 8 призвана разрешить эти трудности за счет унификации процедур обмена данными через единый, гибкий и открытый коммуникационный протокол, поддерживающий различные процессоры, устройства реального времени, а также изделия сторонних разработчиков. Новые Переменные Общего Доступа (Shared Variables) в LabVIEW 8 являются уровнем абстракции транспортного протокола, адаптированы к передаче сложных типов данных, характерных для расширенных приложений с распределенных системах, и легко масштабируются до использования в функциях высокого уровня – протоколирования и тревожной сигнализации. Переменные Общего Доступа позволяют обмениваться данными между всеми узлами распределенной системы, включая узлы, работающие под управлением ОС жесткого реального времени, а также предоставляют доступ к историческим базам данных и операторским консолям с Web-интерфейсом. Вы можете легко сконфигу-

рировать переменные при помощи интерактивных диалогов, осуществляя привязку пользовательских элементов управления и индикации к источникам данных в узлах распределенной системы.

Разработка, отладка и загрузка кода на узлы распределенной системы

Обмен данными и командами между различными узлами – это только одна из трудностей разработки распределенных систем. Управление исходным программным кодом для каждого из узлов и загрузка исполняемого кода на все распределенные узлы также представляет собой серьезную задачу для разработчиков. Только в простейшем случае система состоит из однотипных вычислительных узлов, исходный программный код располагается на центральном компьютере и синхронно переносится на все узлы. В реальном, более сложном случае, в системе присутствуют узлы различного типа (смешанная архитектура), исполняемый код которых различен, причем не все узлы одновременно могут быть доступны для управления и перепрограммирования.

Новая оболочка управления проектами в LabVIEW 8 (LabVIEW 8 Project) хранит исходные коды и настройки всех узлов распределенной системы, включая ПК, контроллеры реального времени, системы на базе ПЛИС, портативные (карманные) компьютеры. Проект также предоставляет множество новых средств для совместной разработки и управления крупным приложением коллективом разработчиков, такие как:

- Интегрированные средства управления исходным кодом, совместимые с ведущими программными продуктами подобного назначения, например, Visual SourceSafe, Perforce, Rational ClearCase, PVCS, MKS и CVS
- Библиотеки Проектов (Project Libraries), содержащие исходные коды в виде модульных, унифицированных функций, которые можно многократно вызывать из различных подсистем
- Средства для хранения настроек устройств управления и ввода/вывода данных, входящих в состав каждого из узлов распределенной системы
- Создание спецификаций, определяющих и хранящих многочисленные опции и настройки дистрибутивов исходного кода, от-

ладки и компиляции исполняемого кода, а также описание процессов окончательной загрузки приложений на удаленные узлы

Используя возможности распределенного интеллекта в LabVIEW 8, Вы значительно облегчаете процесс разработки распределенных систем. Все узлы и устройства – процессоры реального времени, ПЛИС, традиционные приборы, программируемые контроллеры автоматизации с OPC, карманные компьютеры – отображаются в окне Проекта LabVIEW, что упрощает конфигурирование и управление системой. Вы можете добавлять в Проект LabVIEW платформы исполнения, даже если они в данный момент времени работают в автономном режиме или недоступны – это также ускоряет проектирование и разработку системы с временно отсутствующими компонентами. Из простой и дружественной оболочки Проекта LabVIEW, Вы можете наблюдать, редактировать, загружать, выполнять и отлаживать программный код, работающий на любом узле системы. Вы можете также в реальном масштабе времени отслеживать взаимодействие между различными узлами системы. Эта возможность позволяет улучшить синхронизацию и коммуникации в системе на всех этапах ее создания – проектирования, разработки и отладки, тем самым значительно сокращая полное время разработки.

Синхронизация отдельных узлов системы с множеством процессоров и платформ исполнения

Важной составляющей частью разработки распределенной системы является организация совместной работы интеллектуальных узлов – координация и синхронизация их действий. Во многих распределенных системах такое взаимодействие осуществляется через операции ввода-вывода при помощи датчиков, исполнительных устройств и непосредственной генерации специальных сигналов синхронизации. Зачастую инженеры-разработчики прибегают к аппаратной реализации процедур синхронизации и тактирования узлов через ПЛИС и служебные сигнальные линии, интегрированные в системные шины устройств. LabVIEW 8 предлагает новое детерминированное Ethernet-решение для надежной синхронизации узлов в распределенных системах. Новая Переменная Общего Доступа LabVIEW (LabVIEW Shared Variable) может иметь жесткую привязку ко времени ее обновления и поэтому может быть использована для построения сложных распределенных систем управления с коррелированны-

ми измерительными и управляющими каналами, расположенными в различных узлах. Вместо применения дорогостоящих карт «зеркальной памяти», LabVIEW 8 обеспечивает простое, недорогое и стандартизированное решение по тактированию и синхронизации узлов системы в сети с периодом синхронизации 100 мкс и точностью ±5 мкс.

LabVIEW 8 представляет возможности распределенного интеллекта

Современные тенденции показывают, что разрозненные контрольно-измерительные системы предприятий объединяются в распределенные системы более высокого уровня с полной интеграцией вычислительных и управляющих ресурсов. LabVIEW 8 является высокоэффективной, но простой в использовании оболочкой для проектирования, управления, запуска и синхронизации распределенных систем. Для удовлетворения ваших текущих и перспективных потребностей LabVIEW обеспечивает:

- Поддержку различных архитектур и платформ исполнения, таких как персональные, промышленные, портативные и встраиваемые компьютеры, в том числе многопроцессорные системы с ПЛИС и цифровыми сигнальными процессорами, а также системы, работающие под управлением ОС жесткого реального времени
- Мониторинг и управление распределенными узлами системы из единой интерактивной оболочки Проекта LabVIEW (LabVIEW Projet)
- Упрощение передачи данных между различными вычислительными узлами при помощи новой Переменной Общего Доступа LabVIEW 8 (LabVIEW Shared Variable)
- Поддержку множества вариантов синхронизации и тактирования узлов распределенных систем через новую технологию детерминированного Ethernet

Графическая платформа разработки приложений LabVIEW способствует повышению производительности труда инженеров и ученых – от разработки простых лабораторных стендов до создания сложнейших распределенных систем с интеллектуальными узлами. Уникальное сочетание простых графических средств разработки, поддержки широкого спектра устройств ввода-вывода, возможностей программирования распределенных систем и быстрорастущего сообщества пользователей делает платформу LabVIEW передовым про-

дуктом, используемым для решения задач проектирования, управления и измерений.

Чтобы узнать, как уже сегодня начать разрабатывать приложения для распределенных интеллектуальных систем в LabVIEW 8, позвоните в филиал корпорации National Instruments Russia по телефону +7 (095) 783-6851 или посетите **ni.com/russia**.

СПИСОК ЛИТЕРАТУРЫ

1. Бутырин, П.А. Автоматизация физических исследований и эксперимента: компьютерные измерения и виртуальные приборы на основе LabVIEW 7 Express / П.А. Бутырин, Т.А. Васьковская, В.В. Каратаев, С.В. Материкин – М.: ДМК Пресс, 2005. - 264 с.
2. Суранов, С.Я. LabVIEW 7: справочник по функциям / С.Я. Суранов. - М.: ДМК Пресс, 2005. - 512 с.
3. Суранов, С.Я. LabVIEW 8.5: справочник по функциям / С.Я. Суранов. - М.: ДМК Пресс, 2007. - 572 с.
4. Визильтер, Ю.В. Обработка и анализ цифровых изображений с примерами на LabVIEW IMAQ Vision / С.Ю. Желтов, В.А. Князь, А.Н. Ходарев, А.В. Моржин. - М.: ДМК Пресс, 2007. - 464 с.
5. Батоврин, В.К. LabVIEW: практикум по электронике и измерительной технике / В.К. Батоврин, А.С. Бессонов, В.В. Мошкин. - М.: ДМК Пресс, 2007.
6. Батоврин, В.К. LabVIEW: практикум по основам измерительных технологий / В.К. Батоврин, А.С. Бессонов, В.В. Мошкин. - М.: ДМК Пресс, 2007.
7. Евдокимов, Ю.К. LabVIEW для радиоинженера: от виртуальной модели до реального прибора / Ю.К. Евдокимов, В.Р. Линдваль, Г.И. Щербаков. - М.: ДМК Пресс, 2007.
8. Федосов, В.П. Цифровая обработка сигналов в LabVIEW / В.П. Федосов, А.К. Нестеренко. - М.: ДМК Пресс, 2007.
9. Орешина, М.Н. Контроль и регулирование технологических процессов пищевых производств с применением ЭВМ: методические указания по выполнению лабораторных работ. - Орел: ОрелГТУ, 2006. - 57 с.