



*Ярмолинский Леонид Маркович*

## **УРОКИ LABVIEW – ПЕРВЫЕ ШАГИ К ПРОФЕССИОНАЛЬНОМУ ПРОГРАММИРОВАНИЮ НА ЗАНЯТИЯХ ПО РОБОТОТЕХНИКЕ**

Мы начинаем цикл уроков по программированию робототехнических устройств в среде LabVIEW (Laboratory Virtual Instrument Engineering Workbench), самом популярном в мире программном продукте для систем сбора данных, их анализа, обработки и визуализации от компании National Instruments [1]. Уже на протяжении более 20 лет LabVIEW уверенно позиционирует себя на рынке программных продуктов как высокоэффективное средство для создания гибких и масштабируемых интерактивных приложений для измерений, управления и тестирования. Приложения, написанные в LabVIEW [2], находят применение в разнообразных областях, таких как автомобилестроение, аэрокосмические технологии, разработка и производство электроники, телекоммуникации, управление технологическими процессами, биомедицина и т. д. Благодаря своим качествам, LabVIEW используется на всех этапах технологического процесса от моделирования и разработки прототипов продуктов до широкомасштабных испытаний и поточных производств. Это среда, которую используют технические специалисты, инженеры, преподаватели и ученые по всему миру для быстрого создания комплексных приложений для измерения физических величин, тестирования готовой продукции, управления технологическими процессами и пр. Особенно широкое распространение среда нашла в системах автоматизации научного эксперимента (АСНИ),

благодаря большому числу встроенных функций и библиотек, тесной интеграции с новейшими аппаратными и программными решениями, совместимости с операционными системами Windows 2000/NT/XP, Mac OS X, Linux и Solaris.

Популярность LabVIEW объясняется его высокой наглядностью при хорошей читаемости кода и простоте его модификации. Графика среды позволяет создавать эргономичные, красивые и функциональные интерфейсы лицевой панели со шкалами и манипуляторами управления. Большое количество готовых средств визуализации позволяет легко представлять собранные данные в виде графиков, таблиц, с помощью компьютерной анимации, звука.

Концепция языка среды – визуальное графическое программирование [3], в котором, вместо традиционного символьного кода, используются графические образы (иконки), а сам процесс программирования сводится к построению структурных схем из графических элементов. Алгоритмическая основа среды – язык графического программирования «G», относящийся к классу языков, описывающих архитектуру потоков данных. Особенность таких языков – формирование динамических данных, которые связаны между собой, образуют единую структуру и способны инициировать выполнение операторов программы. Это позволяет описывать процессы, идущие параллельно, запускать независимо друг от друга в произ-

вольном порядке различные ветви программы, управлять информационными потоками как из внешней среды, так и от внутренних источников.

Именно этим объясняется востребованность графических языков в новом техническом направлении – робототехнике. Управление робототехническими устройствами сегодня осуществляется программами, реализованными в различных графических средах [4]. Известно, что стандартное учебное оборудование по робототехнике, используемое в образовательных учреждениях, опирается на среду программирования Robolab [5], основанную на специальной урезанной версии LabVIEW, с ограниченными функциями и интерфейсом. Поэтому программирование сложных технических задач, невозможное в Robolab, может быть реализовано на более высоком уровне непосредственно в LabVIEW. Любая профессиональная среда, востребованная в области АСНИ и САПР, ставит вопрос о коммерческом характере своего распространения. В нашем случае можно ориентироваться на специальную версию языка для школ.

Описанные достоинства среды делают изучение LabVIEW уже в школе интересным и перспективным направлением, а имеющееся учебное робототехническое оборудование создает хорошую программно-техничес-

кую базу для построения курса по основам языка, доступного для понимания школьникам. Мы считаем, что после освоения учебной среды Robolab ребенок оказывается достаточно подготовлен к пониманию более сложных структур программирования и написанию простых программ в среде LabVIEW.

Разработчиками языка процесс освоения LabVIEW хорошо подкреплён интерактивной обучающей системой, контекстно-зависимой помощью и множеством примеров использования приемов программирования. Одним из интересных примеров разработки для детей уроков по Labview является интернет-площадка для учителей [6]. Однако существуют проблемы с локализацией. Все это методическое богатство на сегодняшний день русифицировано лишь в небольшом объеме (например [6, 7]). Поэтому надеемся, что предлагаемый материал будет интересен и полезен заинтересованному читателю, увлекающемуся построением и программированием роботов, а данная серия публикаций будет одним из кирпичиков методического сопровождения процесса освоения начального уровня прикладного программирования в среде LabVIEW.

Пробная версия Labview for Educational скачивается по ссылке: <http://joule.ni.com/nidu/cds/view/p/id/2635/lang/ru>

## УРОК 1. СЕКРЕТЫ ИНТЕРФЕЙСА LABVIEW

### ИНТЕРФЕЙСНЫЕ ПАНЕЛИ FRONT PANEL И BLOCK DIAGRAM

После запуска LabVIEW появляется диалоговое окно, которое предлагает создать новую программу (New) или открыть уже существующий проект (Open). Следует выбрать Blank VI для создания нового приложения. Перед нами раскрываются два окна (см. рис. 1 и 2): *интерфейсная панель* (Front Panel) и *окно редактирования диаграмм* (Block Diagram), которое по своей сути является графическим окном написания программы.

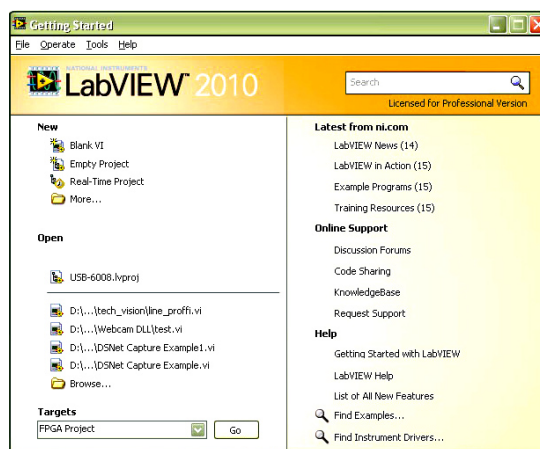


Рис. 1. Интерфейсная панель (Front Panel)

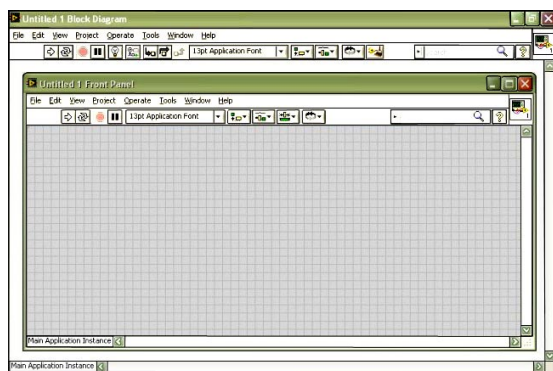


Рис. 2. Окно редактирования диаграмм (Block Diagram)

Интерфейсная панель (Front Panel) – это интерфейс пользователя. В этом окне устанавливаются графические элементы разных типов: индикаторы ввода и вывода, элементы управления. Последние мы будем называть «контролями» (controls). Для удобства пользователя контроли реализованы в виде разнообразных ручек, регуляторов, кнопок, ползунковых устройств и др.

Установленные на переднюю панель контроли и индикаторы отображаются соответствующими иконками (терминалами) во втором окне – окне редактирования диаграмм. То есть каждому установленному элементу на интерфейсной панели соответствует иконка в окне редактирования. В этом окне и «пишется» программа в виде графического кода.

Обе панели связаны друг с другом (см. рис. 3 и 4). Удалив, например, контрол или индикатор в интерфейсном окне, мы тем самым удаляем соответствующую иконку (терминал) в окне редактирования диаграмм.

## ЛИНЕЙКА ИНСТРУМЕНТОВ

Оба окна – как интерфейсное, так и окно редактирования диаграмм – имеют линейки инструментов, которые содержат служебные кнопки и индикаторы состояния, предназначенные для контроля виртуальных инструментов. Одна из линеек инструментов всегда доступна, и ее вид зависит от того, в каком окне вы находитесь (см. рис. 5). Назначение кнопок линейки инструментов мы будем разбирать по ходу наших занятий.

## ПАЛИТРА ИНСТРУМЕНТОВ

Следующим важным компонентом работы в Labview является панель инструментов (Tools Palette), пример которой вы видите на рис. 6. Вызывается она из основного меню **View** → **Show Tools Palette**.

В данный момент нам важна только одна кнопка этой панели, а именно Автоматический выбор инструмента (Automatic tool selection), которая позволяет автоматически подбирать инструмент под то действие, ко-

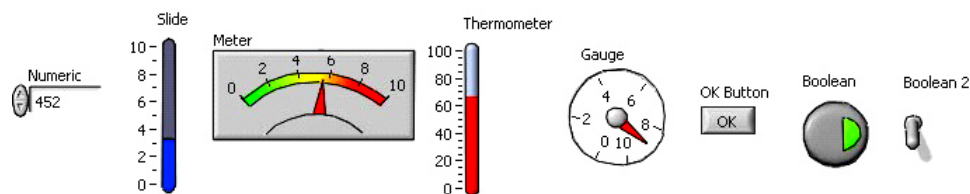


Рис. 3. Пример контролов на передней панели



Рис. 4. Пример соответствующих контролам иконок на интерфейсной панели

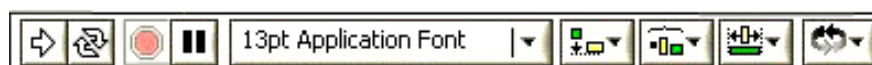


Рис. 5. Вид линейки инструментов



Рис. 6.  
Вид палитры инструментов

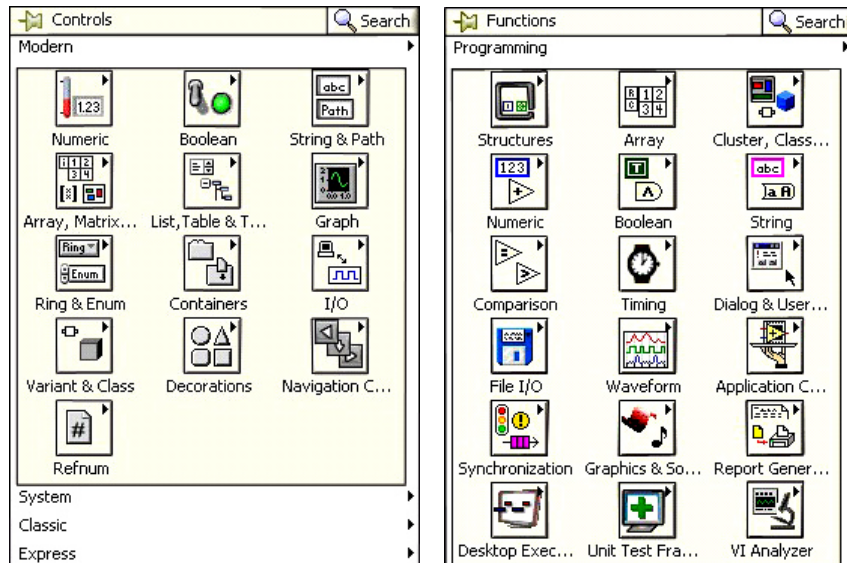


Рис. 7. Примеры палитр индикаторов и функций

торое мы хотим совершить. Если эта кнопка не нажата, то необходимо ее нажать, тем самым разрешить автоматический подбор инструментов.

### ПАЛИТРЫ ИНДИКАТОРОВ И ФУНКЦИИ

Палитры индикаторов и функций представляют собой структурированный набор иконных меню, предназначенных для доступа к библиотекам элементов интерфейса и функций (см. рис. 7). Вызов необходимой

панели осуществляется автоматически, при переключении между окном редактирования и интерфейсной панелью. Вызвать палитры можно либо из основного меню **View** → **Show Controls Palette**, **View** → **Show Functions Palette**, либо нажатием правой кнопки мышки на поле лицевой панели или панели редактирования диаграмм.

### ТЕРМИНАЛЫ, ФУНКЦИИ, СВЯЗИ

Терминалы осуществляют связь между интерфейсной панелью и диаграммой (см. рис. 8)



*В этом окне... индикаторы ввода и вывода, элементы управления... в виде разнообразных ручек, регуляторов, кнопок...*

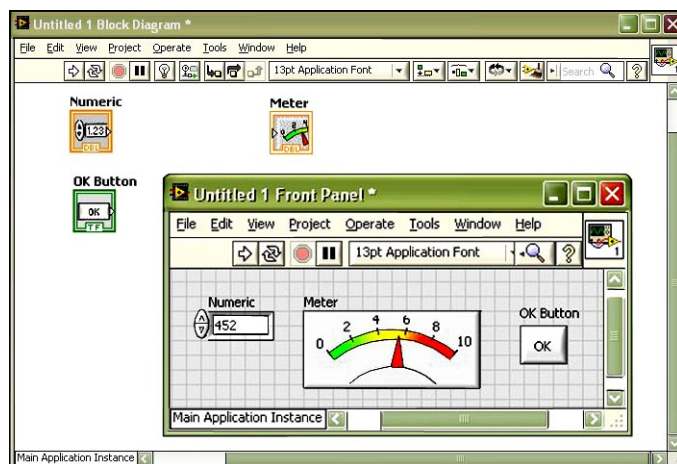


Рис. 8. Фрагмент программы с обозначения терминалов, функций и связей

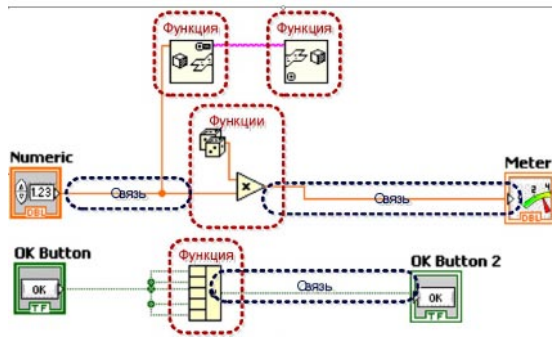


Рис. 9. Терминальные связи между интерфейсной панелью и диаграммой

*Функции* – это объекты окна редактирования диаграмм, которые могут иметь один и/или несколько входов и/или выходов. Функции LabVIEW аналогичны выражениям, операторам, процедурам и функциям текстовых языков программирования. Примеры функций изображены на рис. 9.

*Связи* – это графическое отображение потоков данных в виде соединительных линий между иконками (терминалами). Важно запомнить, что данные могут передаваться только в одном направлении: от терминала-источника к одному или нескольким терминалам-приемникам.

### ТИПЫ ДАННЫХ

LabVIEW поддерживает работу с данными разных типов (целые, дробные, логические, строковые). Связи возможны между терминалами, относящимися к одному типу

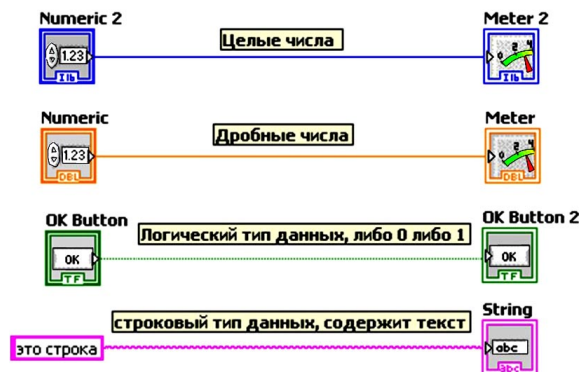


Рис. 10. Объявление типов данных

данных. Различный вид и цвет соединений соответствует различным типам передаваемых данных (см. рис. 10). Неправильная связь терминалов или незаконченное соединение изображаются штриховой линией (см. рис. 11).

### СТРУКТУРЫ

*Структуры* – это циклы, ветвления, операторы выбора. Их графическое представление имеет вид прямоугольной области (контейнера), куда помещаются элементы и связи, составляющие тело цикла или ветви разветвляющегося алгоритма. Условия завершения цикла, счетчики цикла или условия ветвлений также интуитивно понятно отображаются графически (см. рис. 12). Содержание ветвей разветвляющихся алгоритмов в виде совокупности пиктограмм и связей помещаются в один и тот же контейнер на разные его страницы, но в строгом соответствии с условием, по которому организуется ветвление. В процессе программирования состояние условий можно принудительно менять и получать доступ к разным страницам контейнера. Таким образом, видимое содержимое контейнеров на экране не постоянно и может меняться в зависимости от условий, управляющих циклами или ветвлениями.

Терминалы, функции, связи и структуры – это весь основной синтаксис языка программирования LabVIEW. Все очень просто! Теперь можно сочинить первую программу.

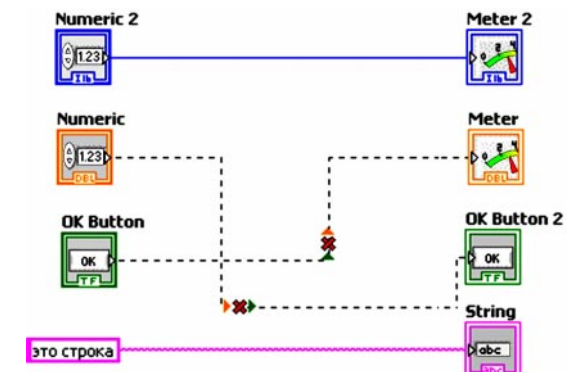


Рис. 11. Пример неправильной связи

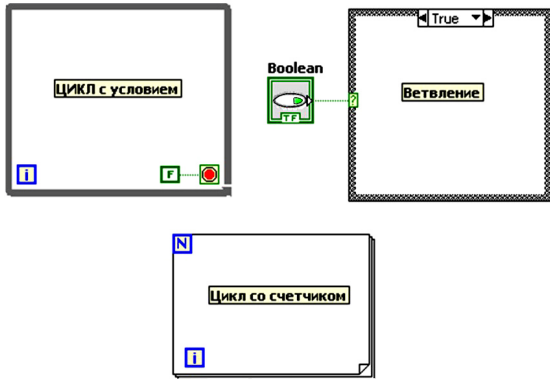


Рис. 12. Графическое изображение структур

*Структуры — это циклы, ветвления, операторы выбора.*

### ПРОГРАММА СУММИРОВАНИЯ ДВУХ ЧИСЕЛ

Создание нашего первого приложения начинаем со стандартной операции: открываем лицевую панель, вызываем палитру индикаторов.

В палитре индикаторов заходим во вкладку **Numeric** (см. рис. 13), выбираем самый первый элемент управления **Numeric Control** и добавляем его на нашу лицевую панель. Одновременно на панели диаграмм появится терминал с числовым выходом оранжевого цвета (см. рис. 14).

Возвращаемся на лицевую панель, таким же образом добавляем еще один **Numeric Control** и из той же вкладки **Numeric** добавляем элемент **Numeric Indicator**.

На панели блока диаграмм появятся 3 иконки, две из которых будут являться терминалами с числовым выходом — желтый треугольник с левого края иконки, а одна

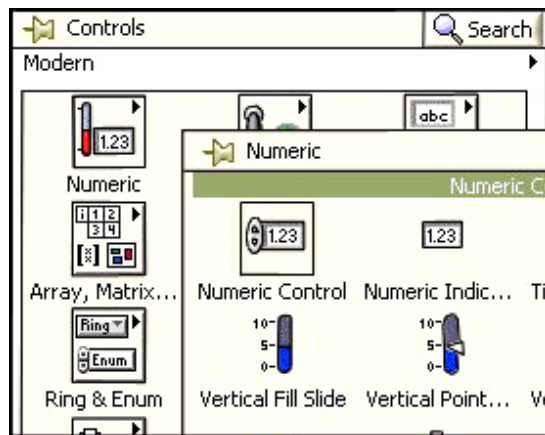


Рис. 13. Вид вкладки Numeric

с числовым входом — желтый треугольник с левого края иконки (см. рис. 15).

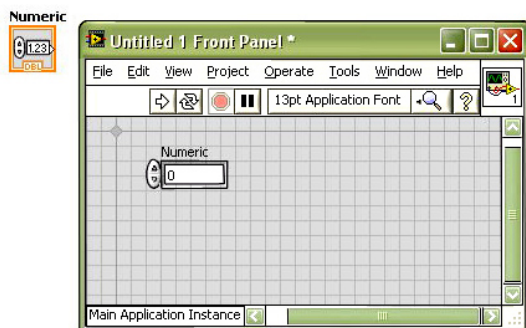


Рис. 14. Терминал с числовым выходом на панели диаграмм

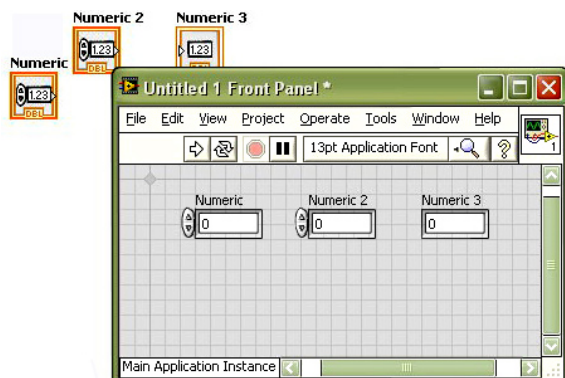


Рис. 15. Соответствие элементов управления и терминалов

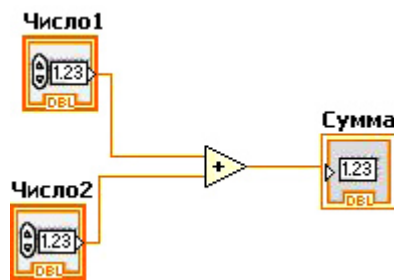



Рис. 16. Вид функции Add


Возвращаемся на лицевую панель, кликаем двойным щелчком на название любого элемента управления и переименовываем его, соответственно: «Число1», «Число2», «Сумма».

Теперь располагаем наши контролы последовательно в линию, после чего переходим на панель редактирования диаграмм, чтобы определить функцию суммы.

Открываем палитру функций, заходим во вкладку **Numeric**, выбираем самую первую функцию **Add** (см. рис. 16) – сложение и добавляем ее на блок диаграмм (см. рис. 17).

Функция суммы имеет вид треугольника, помеченного «плюсиком» с 3 терминалами для присоединения связей (2 – входа и 1 – выход). Подключаем наши два выходных терминала «Число1» и «Число2» к входам функции **Add**, а выход функции соединяем с нашим терминалом выхода «Сумма».

Переходим на лицевую панель, вводим любые значения в элементы управления «Число1» и «Число2» и жмем на кнопку запуска программы .

*Внимание! Если программа не может быть запущена на выполнение по какой-либо причине, то кнопка запуска примет следующий вид:* 

После запуска программы в индикаторе «Сумма» появится результат сложения двух чисел, введенных через терминалы «Число1» и «Число2» (см. рис. 18).



Рис. 18. Иллюстрация выполнения программы

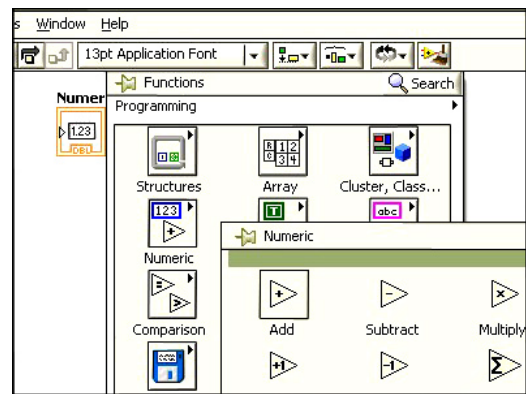




Рис. 17. Вид панели Functions

Теперь нажимаем на кнопку запуска программы в цикле  и пытаемся менять значения элементов управления «Число1» и «Число2» в реальном времени.

Наигравшись, останавливаем программу, при помощи кнопки .

### ПРОГРАММА ВЫЧИСЛЕНИЯ СРЕДНЕГО ДВУХ ЧИСЕЛ

Модифицируем программу и заставим ее считать среднее арифметическое двух чисел. Для этого переходим на панель редактирования диаграмм, добавляем на нее функцию **Devide**  из вкладки **Numeric** и присоединяем ее следующим образом (см. рис. 19).

Нажимаем на нижний терминал функции деления правой кнопкой мышки и в меню выбираем **Create** → **Constant** (см. рис. 20). Появится оранжевая рамка с нулем для ввода значения константы (рис. 21).

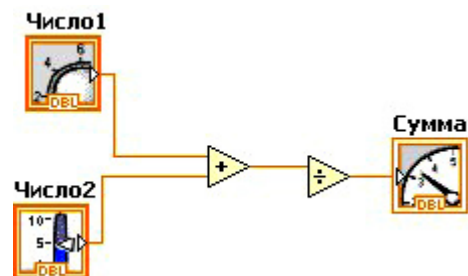


Рис. 19. Модификация программы

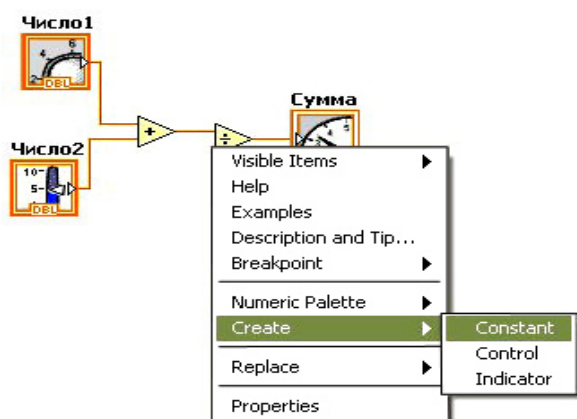


Рис. 20. Вставка константы

*Внимание! Значение, записанное в константе, можно менять только в процессе редактирования программы, когда программа выполняется, значение константы изменить нельзя!*

Двойным щелчком мышки по константе входим в режим редактирования и изменяем значение на 2. Меняем название индикатора на «Среднее арифметическое». Переключаемся на лицевую панель, меняем значения «Число1», «Число2» и проверяем, как работает наша программа (рис. 22).

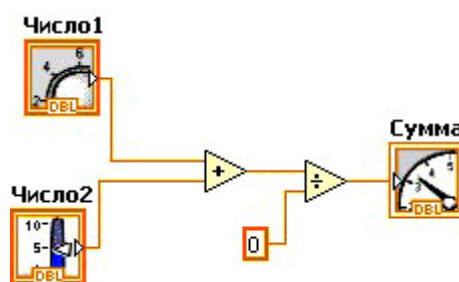


Рис. 21. Константа



Рис. 22. Примеры выполнения программы

На этом наш первый урок заканчивается, обязательно сохраняем полученную программу, она нам пригодится на следующем занятии. В следующих номерах вы сможете найти новые уроки по освоению этой замечательной инженерной среды. Желаем успехов!

## Литература

1. Официальный сайт компании National Instruments в России [электронный ресурс]: <http://russia.ni.com/>
2. Сайт поддержки внедрения LabView в России, СНГ и странах Балтии [электронный ресурс]: <http://www.labview.ru/solutions/>
3. Рак Н.А., Смех В.И., Щербак С.Б. Графическое программирование: Уч. методич. пособие по дисц. «Выч. техн. и программир.» Мн.: БГПА, 1997.
4. Пилипенко А.В., Пилипенко О.В., Пенькова А.П. Анализ технологий программирования микроконтроллеров AVR. [электронный ресурс]: <http://itnop.gu-unpk.ru/media/transfer/Material/2/252/%D1%81%D1%82.rar>
5. GET REAL ABOUT TEACHING SCIENCE AND ENGINEERING [электронный ресурс]: <http://www.k12lab.com/>
6. Белиовская Л.Г., Белиовский А.Е. Программируем микрокомпьютер NXT в LabVIEW. М.: ДМК-Пресс, 2010.

**Ярмолинский Леонид Маркович,**  
 ведущий инженер-программист  
 ООО «ПромАвтоматика»,  
 педагог дополнительного образования  
 ГБОУ СОШ № 255  
 (кружок робототехники).



Наши авторы, 2013.  
 Our authors, 2013.