

## Структура case - ветвление

### Введение

На этом уроке Вы научитесь:

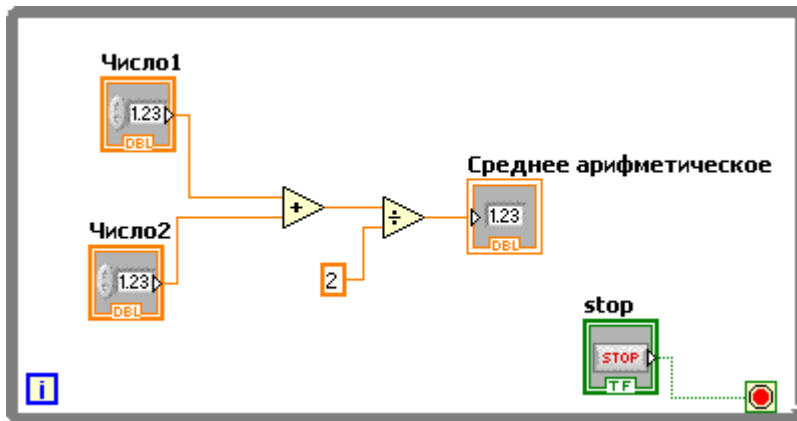
1. Использовать структуры ветвления (case) с логическим и числовым параметром-селектором
2. Обрабатывать нажатие и переключение тумблеров, кнопок, выпадающих меню

### Структура case с логическим параметром-селектором

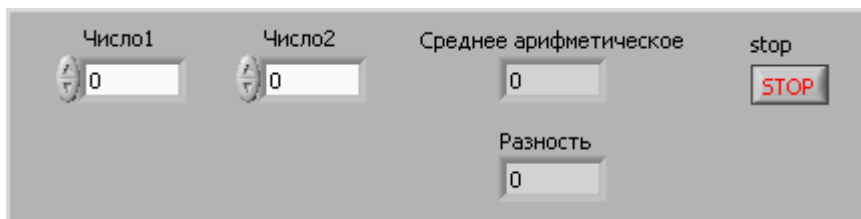
Реализация даже элементарных алгоритмов, как правило, не обходится без операций логического ветвления программы в зависимости от определенных условий. Для этих целей используются так называемые **Case**-структуры. Такие структуры позволяют осуществлять выбор по условию или по значению параметра-селектора и переходить на выполнение соответствующих действий.

Создадим простую программу, которая, в зависимости от положения тумблера, будет считать среднее арифметическое двух чисел или находить разность этих же чисел.

Открываем программу написанную на предыдущем занятии.

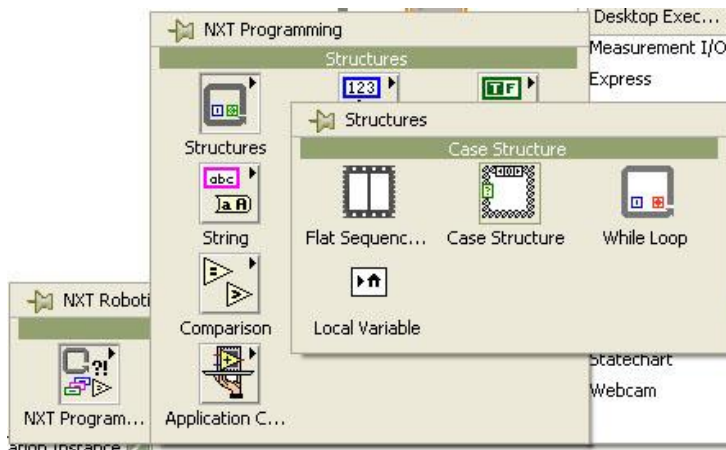


Нам понадобится еще один индикатор для вывода разности значений «число1» и «число2». Переходим на лицевую панель, добавляем еще один числовой индикатор и переименовываем его в «Разность».

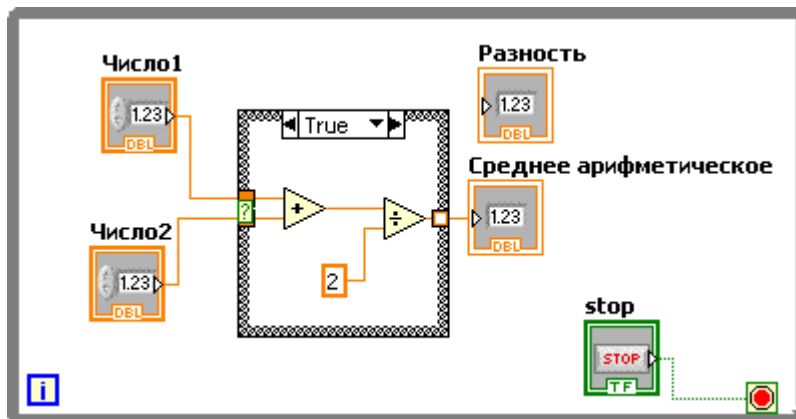


Возвращаемся на панель редактирования диаграмм.

Выбираем новую структуру, **case, NXT programming -> Structures -> Case structure.**

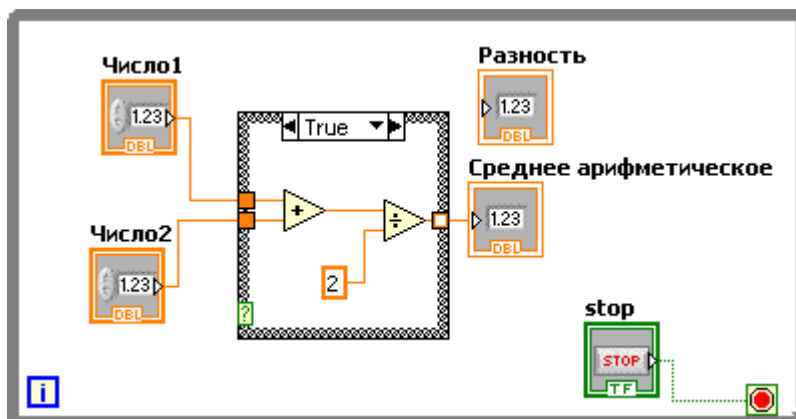


И обводим ей функции сложения (**add**) и деления (**divide**)



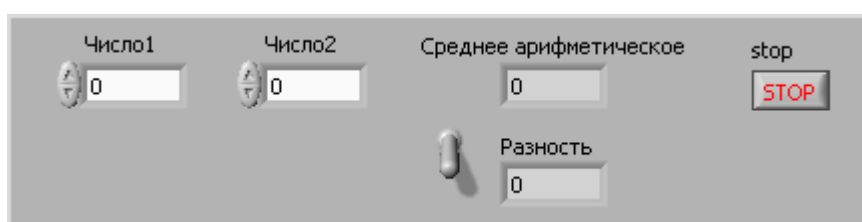
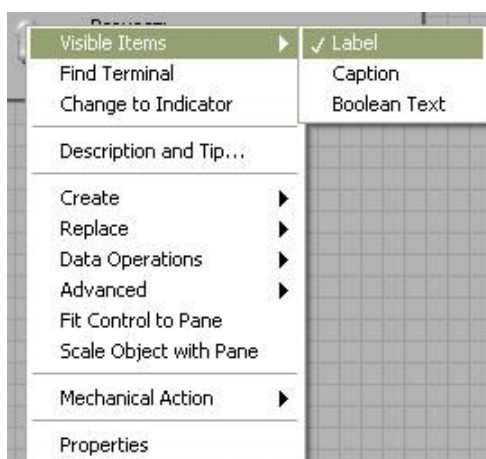
В верхней части данная структура имеет выпадающее меню, , для выбора отображения определенной ветки по значению параметра-селектора. Слева, посередине границы структуры **case**, есть терминал, со знаком вопроса  для подключения самого параметра-селектора, по которому будет вестись ветвление.

Захватом при помощи мышки, перетаскивай терминал параметра-селектора и перемещаем его ближе к нижнему углу границы структуры, чтобы он не перекрывал входные туннели контролов «число 1» и «число 2».



Возвращаемся на лицевую панель и создаем переключатель в виде тумблера **Boolean** -> **vertical toggle switch**. Нажимаем на него правой кнопкой мышки, и снимаем галочку с пункта

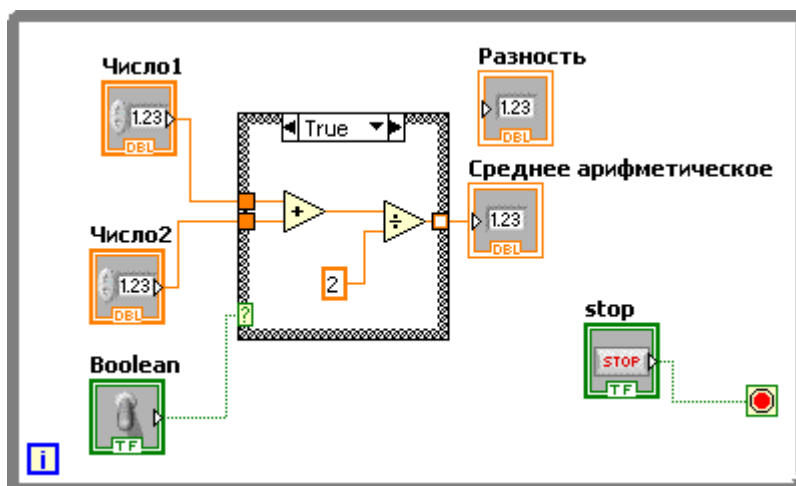
меню **Visible Items** -> **label**, таким образом мы выключим отображение надписи над контролом, которая в данном случае нам не нужна на лицевой панели.



Заранее запоминаем что при нижнем положении тумблера, мы должны считать разность чисел, а при положении тумблера вверх, среднее арифметическое.

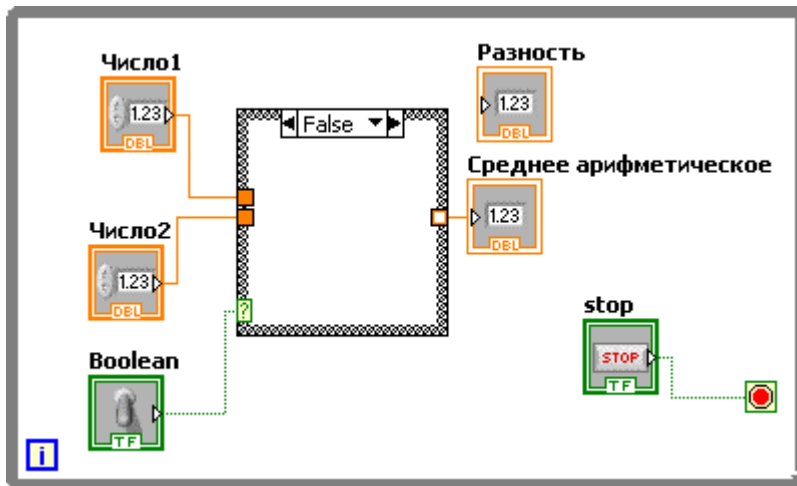
Переходим на панель редактирования диаграмм.

Подсоединяем терминал тумблера к терминалу параметра селектора структуры **case**.




Тумблер – это контрол логического типа (зеленый цвет соединений и терминала), соответственно он имеет 2 значения **true** (истина), **false** (ложь). **False** – тумблер в нижнем положении, **true** – тумблер в верхнем положении.

В выпадающем меню наверху структуры **case**, сейчас стоит значение **true**, это значит что те функции которые мы видим внутри структуры будут выполняться когда значение параметра-селектора, будет равно **true**, то есть в нашем случае когда тумблер будет в верхнем положении. Выбираем в выпадающем меню значение **false**.





Поле структуры пустое, то есть при нижнем положении тумблера, никаких функций выполняться не будет.


Обращаем наше внимание на то, что сейчас кнопка запуска программы со значком разорванной стрелки , это признак того что в нашей программе есть ошибка.

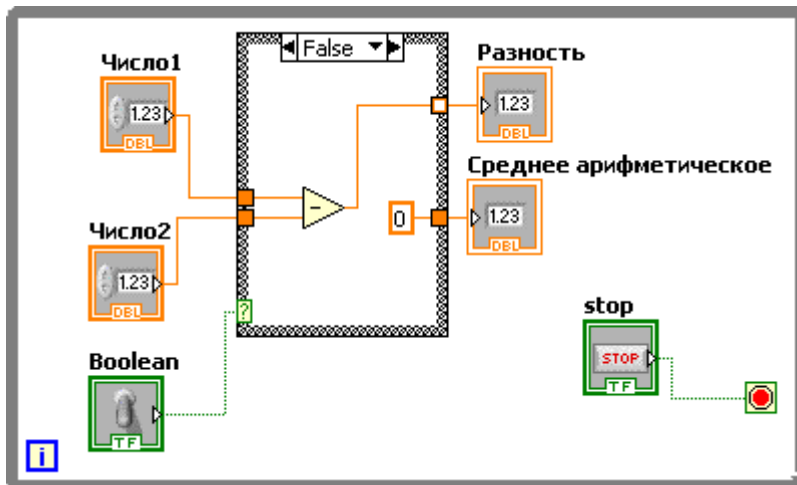
Нажимаем на кнопку. Перед нами откроется окно со списком всех ошибок в программе




Окно содержит 3 поля, верхнее с указанием того, в какой программе ошибки, среднее – список ошибок и нижнее описание выбранной ошибки

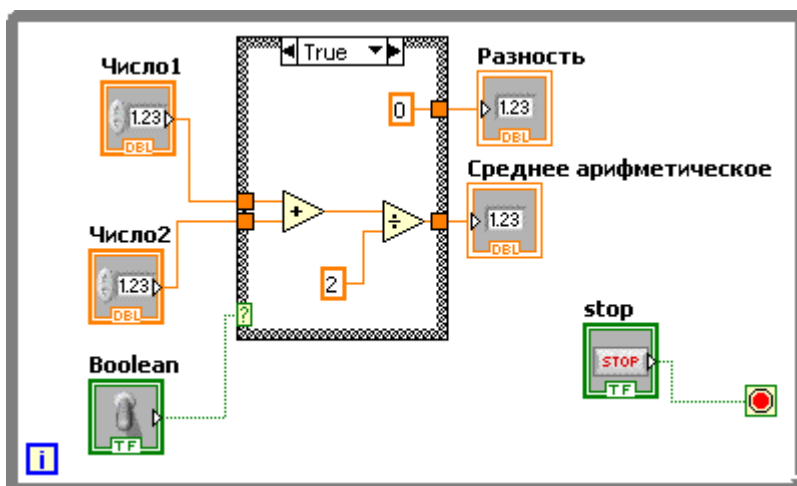
Кликаем двойным щелчком на выделенную ошибку в среднем поле окна, после этого окно закроется и Labview выделит тот участок программы, в котором обнаружена ошибка. В нашем случае выходной туннель структуры case не имеет подключения в каком-то из ветвлений. То есть компьютер не знает, какое значение передать на данный выход структуры. Наш выходной туннель с неопределенными выходными значениями выглядит так , а «хороший» выходной туннель выглядит так . Таким образом можно визуальнo контролировать правильность использования выходных туннелей структуры case.

Кликаем на выходной туннель правой кнопкой мышки и создаем константу. Оставляем в ней значение «0». Добавляем в поле структуры функцию **Subtract**  **Functions -> Numeric -> Subtract**, и присоединяем к ее входам через входные туннели структуры значения «число 1» и «число 2». Выход выводим из структуры и присоединяем к выходному терминалу «Разность», автоматически создается выходной туннель из структуры case.



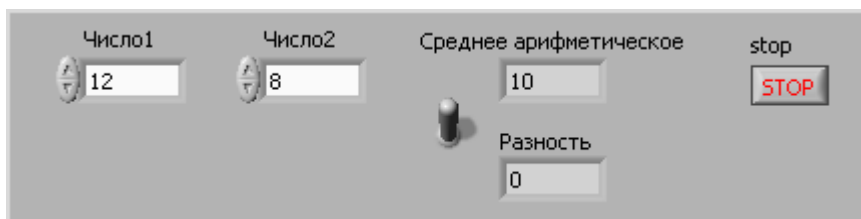
Новый туннель тоже имеет неопределенное значение . Через выпадающее меню наверху структуры case, меняем отображаемое ветвление, на **True**.

У неопределенного туннеля опять создаем константу и оставляем значение «0»

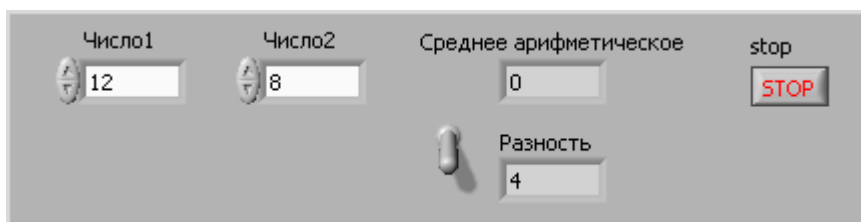


Переходим на лицевую панель и запускаем программу.

В зависимости от положения тумблера наша программа либо считает среднее арифметическое и обнуляет индикатор «Разность»,



Либо наоборот, обнуляет индикатор «Среднее арифметическое» и вычитает «число2» из «числа1».



## Структура case с числовым параметром-селектором


Мы разобрали работу со структурой case с логическим параметром выбора. У данного типа есть всего 2 ветки. Далее мы разберем case с числовым параметром.

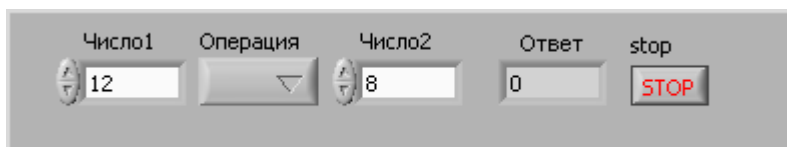
Сохраняем программу под новым именем.

Удаляем тумблер

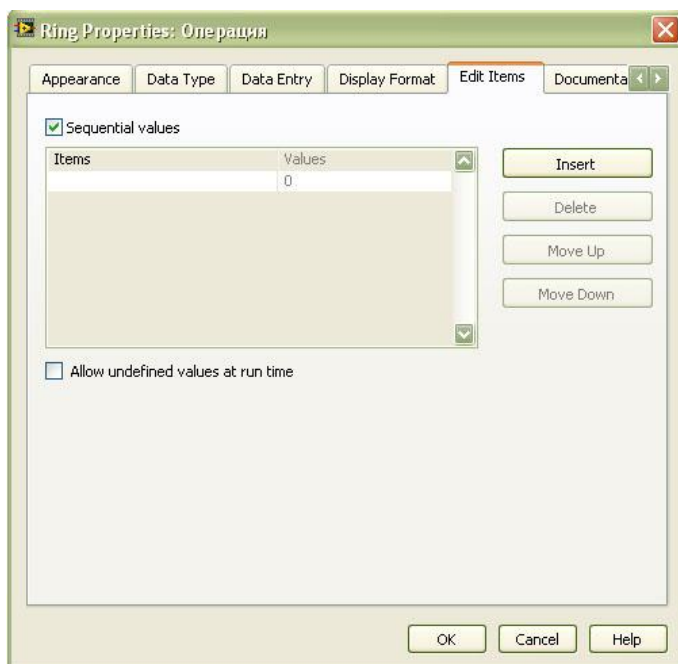
Удаляем индикатор «Разность»

Переименовываем «Среднее арифметическое» в «Ответ»

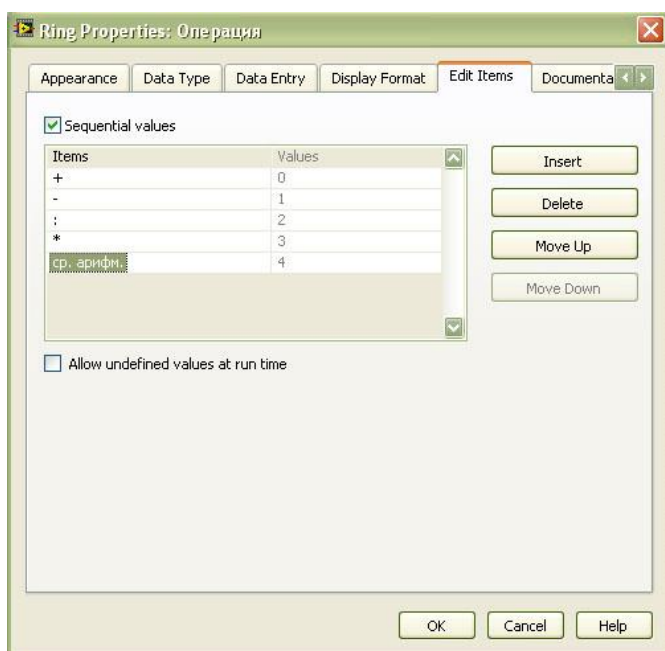
Добавляем новый контрол **NXT Robotics -> Rings & Enum -> Menu ring** , и размещаем его между контролами «число1» и «число2» и даем ему название «Операция».



Кликаем на контролл «операция» правой кнопкой мышки и выбираем пункт меню Edit items. Открывается окно следующего вида:

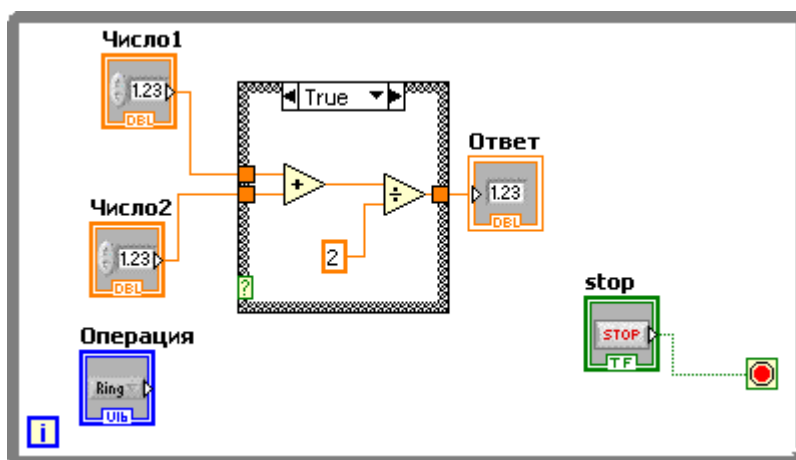


В табличке нажимаем на поле под заголовком Items и через энтер набираем +, -, :, \*, ср. арифм. Таблица должна принять вид:

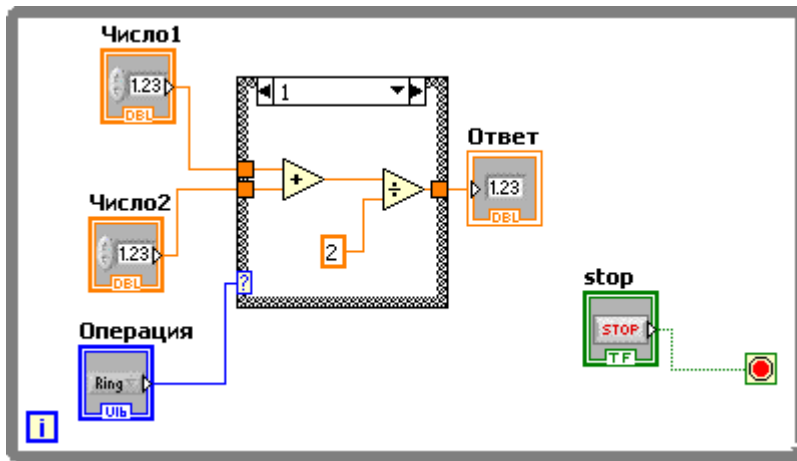


Нажимаем «ок» и переходим на панель редактирования диаграмм.

Удаляем испорченные связи и выходной туннель который вел к терминалу индикатора «Разность», и приводим нашу программу к виду:

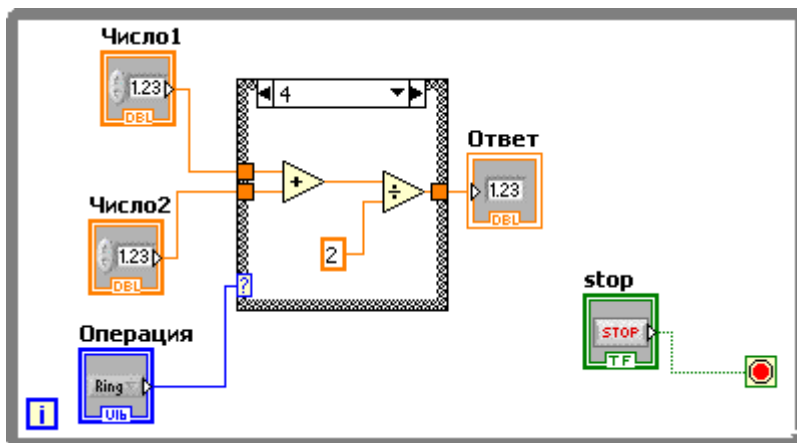


Соединяем выходной терминал «Операция» с терминалом параметра-селектора структуры case, он изменит свой цвет с зеленого на синий, а значение параметра в меню поменяется с True на 1.

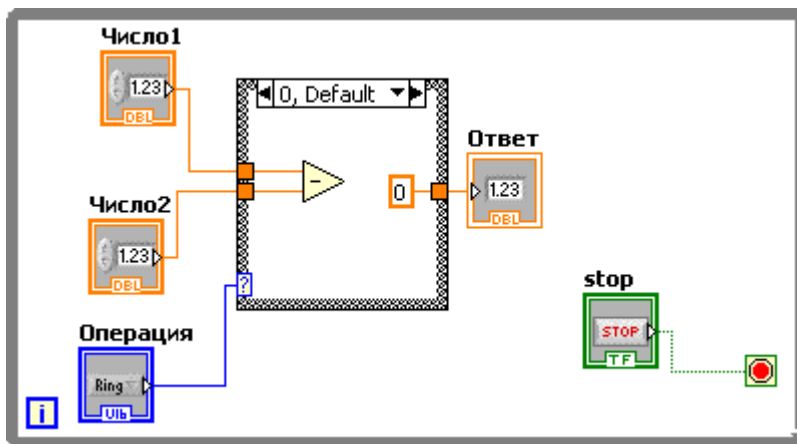


Теперь определенная ветка структуры case выбирается в соответствии с числом которое приходит от контрола «Операция».

Отображаемая сейчас ветка выполняет операцию по вычислению среднего арифметического. Вспоминаем нашу таблицу которую мы создавали для контрола «операция», в ней данная операция соответствует номеру 4, а у нас сейчас эта ветка выполняется при значении 1. Дважды кликаем на 1 в меню ветки структуры case и набираем вместо 1, 4.



Открываем меню и выбираем значение 0, default



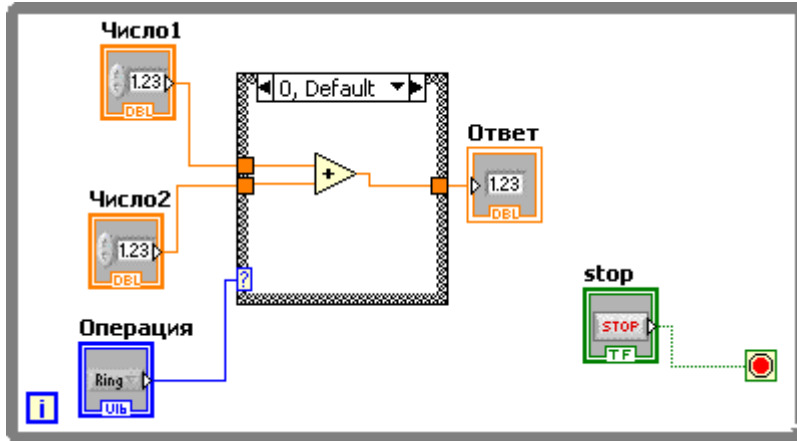
Эта ветка будет выполняться когда значение контрола «Операция» будет 0 или иметь значение не подходящее ни одной другой ветке.



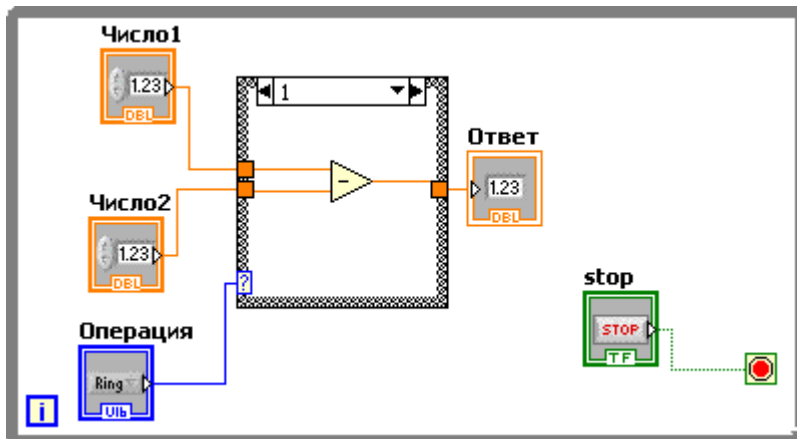
Значению 0 в нашей таблице соответствует операция сложения.

Удаляем функцию вычитания и константу.

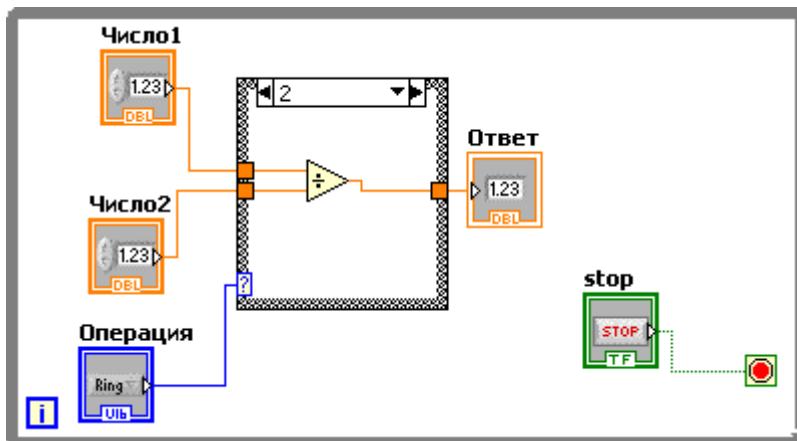
Добавляем функцию сложения и соединяем ее входы и выходы с контролами «Число1», «Число2» и индикатором «Ответ».

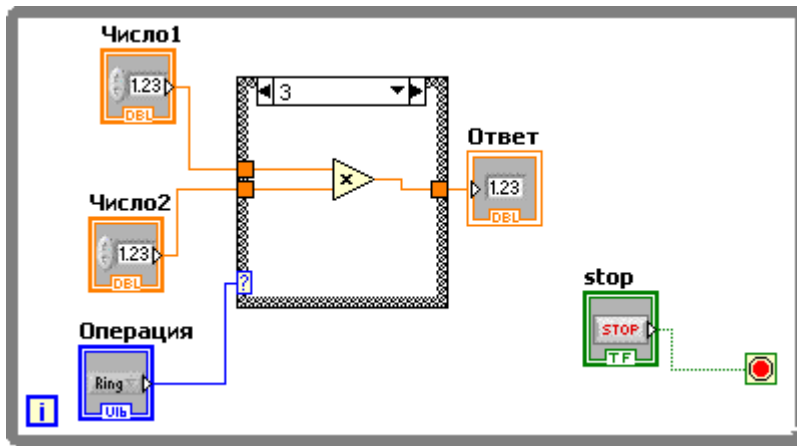


Кликаем на рамке структуры case и выбираем пункт меню Add case after (Добавить ветку после текущей). Вводим 1, как значение параметра для этой ветки. Данному значению соответствует вычитание, добавляем в поле структуры функцию вычитания, и подсоединяем ее.



Повторяем добавление веток еще два раза, для значения 2 – операция деления и 3 – операция умножения.





Запускаем программу и проверяем ее работу:

Число1	Операция	Число2	Ответ	stop
12	+	8	20	STOP

- сложение;

Число1	Операция	Число2	Ответ	stop
12	-	8	4	STOP

- вычитание;

Число1	Операция	Число2	Ответ	stop
12	:	8	1,5	STOP

- деление;

Число1	Операция	Число2	Ответ	stop
12	*	8	96	STOP

- умножение;

Число1	Операция	Число2	Ответ	stop
12	ср.	8	10	STOP

- среднее арифметическое

Таким образом, у нас получился калькулятор сделанный собственными руками!

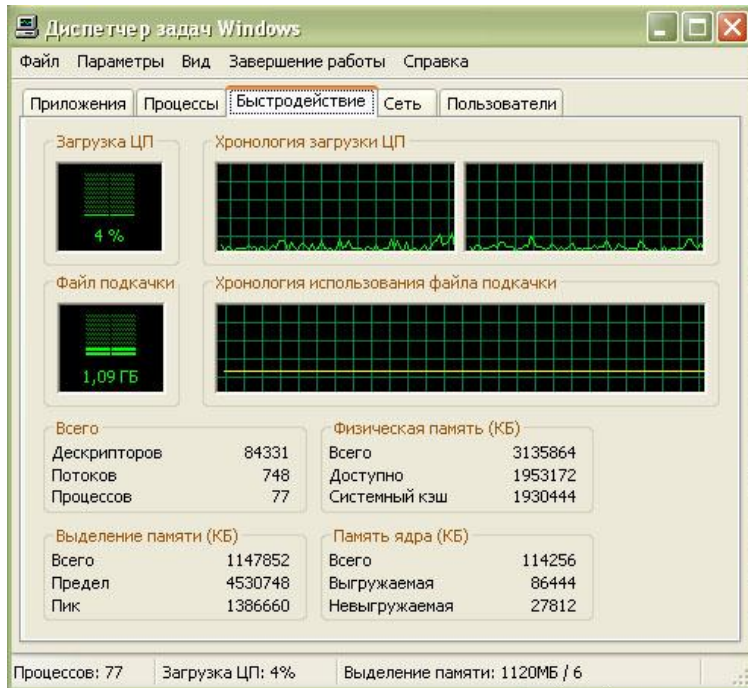
Если при работе программы выполняемая операция не соответствует то, что вы выбрали, значит в каком-то месте была совершена ошибка в выборе значения параметра-селектора для той ветки которая не верно выполняется. Следует открыть таблицу значений для контроля «Операция», и проверить соответствие веток структуры case значениям из таблицы.

В заключении этого урока мы поговорить об

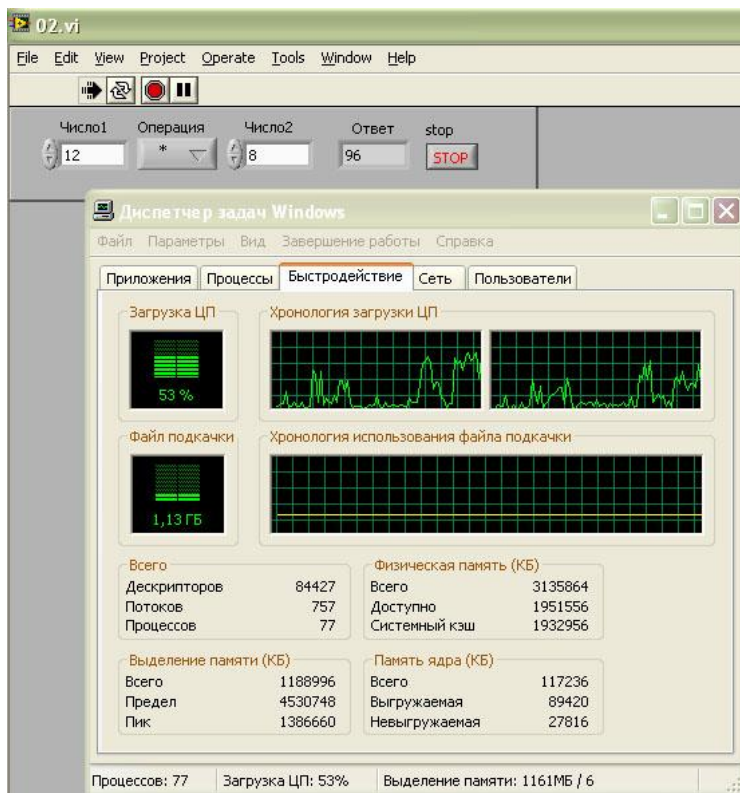
Нажимаем ctrl+alt+delete, если вы работаете под Windows XP диспетчер задач появится сразу, если под Windows Vista или Windows 7 появится меню в котором нужно будет выбрать диспетчер задач

Нас будет интересовать вкладка «быстродействие», открываем ее и видим две полоски с графиками, верхняя показывает загрузку центрального процессора (далее ЦП) компьютера, то есть на сколько сильно ему приходится «Думать» в данный момент, а вторая это сколько в данный момент используется оперативной памяти. Вторая полоска нам сейчас не понадобится, поэтому остановимся на верхней полоске с графиком загрузки ЦП.

Когда у вас на компьютере ничего не запущено, то загрузка ЦП колеблется в диапазоне от 0 до 10%.




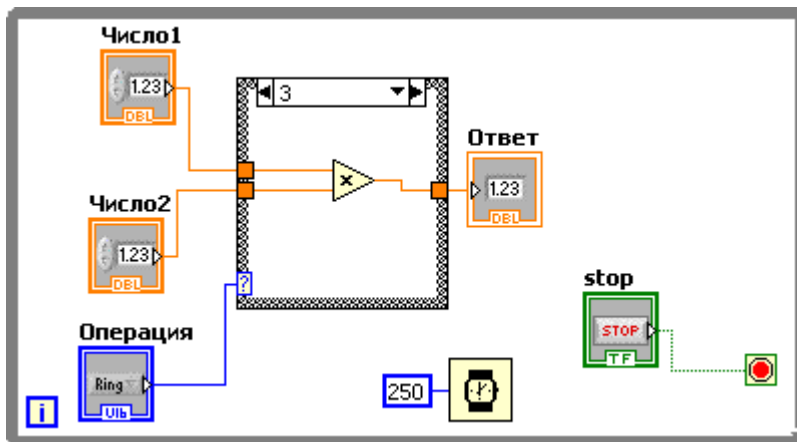
Теперь не закрывая окошко диспетчера задач запускаем нашу программу



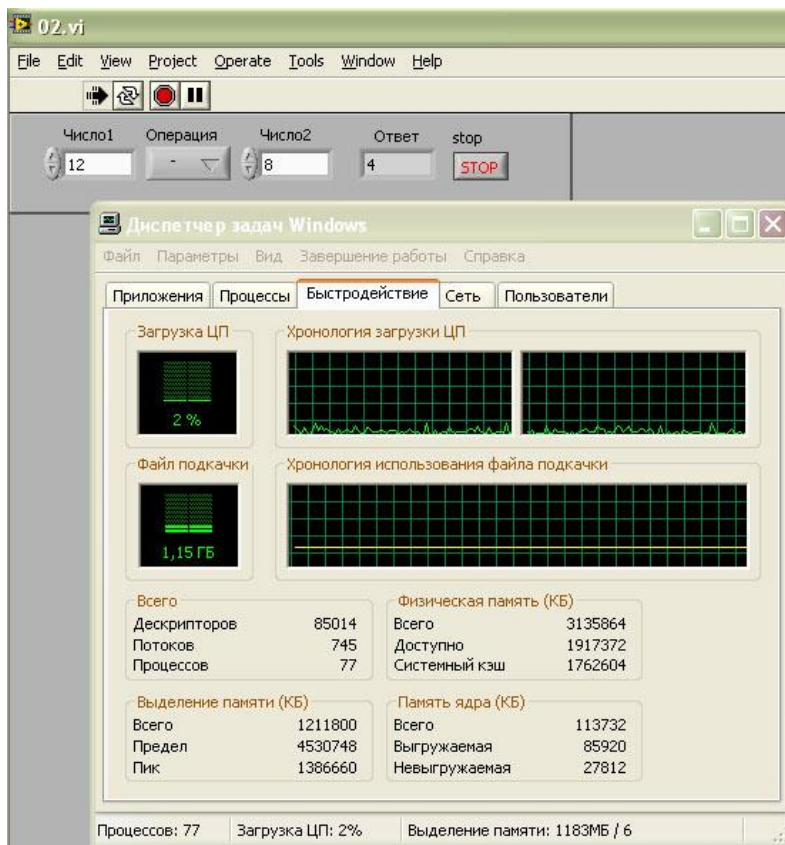
Мы видим что загрузка ЦП подскочила до 53%, а как мы помним наша программа это всего лишь простой калькулятор и занимать около 50% процессорного времени это все равно что заставить весь класс протереть доску после урока, хотя с этой задачей спокойно может справиться и один человек.

Такая нагрузка «мозга» компьютера происходит потому, что наша программы выполнены в виде бесконечного цикла, выход из которого осуществляется кнопкой стоп, то есть пока программа работает, она постоянно выполняет одни и те же операции, без остановки. За одну секунду цикл выполняется порядка 2,5 миллионов раз, для нас такая скорость не нужна, поэтому для облегчения работы программы мы поставим временную задержку в цикле. Добавляем

функцию задержки **wait**  **NXT Robotics -> Time -> Wait (ms)**, и создаем на ее входе числовую константу со значением 250 миллисекунд.



Снова проводим эксперимент с определением загрузки ЦП при работе программы



Загрузка ЦП снизилась до значения соответствующего работе компьютера без запущенных программ. Таким образом мы оптимизировали нашу программу, все что мы хотели от нее она выполняет, и при этом не загружает ЦП компьютера. При программировании роботов это позволит нам уменьшить нагрузку на ЦП блока NXT тем самым снизить потребление электричества.