Ярмолинский Леонид Маркович

УРОК 3. ЗАДАЧА ПОЗИЦИОНИРОВАНИЯ МОТОРА LEGO NXT

ВВЕДЕНИЕ

Мы уже научились управлять моторами через контроллер NXT. На этом занятии мы рассмотрим такую важную задачу автоматического управления как позиционирование мотора.

Для этого занятия нам также понадобится робот и кабель для записи программы на NXT.

ФИКСАЦИЯ ПОЛОЖЕНИЯ МОТОРА

В автоматическом управлении очень часто встречается задача четкой фиксации положения мотора. Для начала мы рассмотрим частную задачу, когда мотор надо зафиксировать в том положении, которое было при запуске нашей программы.

Начинаем с создания шаблона программы, с которым мы познакомились на прошлом уроке (рис. 1).

Поскольку после включения программы NXT не может правильно определить положение энкодера, в начале программы мы должны принять текущее положение энкодера как ноль.

Добавляем на панель диаграмм функцию чтения датчика **NXT I/O → Read Sensor**, выбираем в выпадающем меню под иконкой функции **Reset Motor** и вставляем ее перед циклом, но после функции **Specify NXT**.

Около верхнего входного терминала функции создаем константу и указываем в ней «порт А» (рис. 2).

Теперь энкодер готов к работе. Добавляем еще одну функцию чтения датчика **NXT I/O → Read Sensor** и вставляем ее внутрь цикла (рис. 3).

Рис. 1

Рис. 2

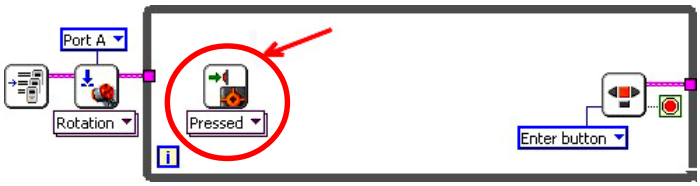


Рис. 3

Выбираем в выпадающем меню тип функции **Read Rotation** (прочитать показания энкодера) и указываем константой «порт А» (рис. 4).

На выходном терминале этой функции мы будем получать актуальное положение мотора.

Создаем числовую константу перед циклом и заводим ее значение в цикл через туннель (рис. 5).

В этой константе будет указано положение в градусах, которое мотор должен удерживать. Поскольку по условию задачи нам необходимо удерживать начальное положение после запуска программы, то мы оставим нулевое значение.

Далее нам нужно сравнить, насколько реальное положение мотора отличается от заданного положения. Для этого вычтем одно из другого. Эту разность в дальнейшем мы будем называть ошибкой по положению (рис. 6).

Теперь ошибку по положению умножаем на 0,5.

Добавляем функцию управления мотором **Functions → NXT I/O → Motor control** и выдаем результат умножения как задание мощности на этот мотор (рис. 7).

После цикла вставляем функцию останова двигателя А (**Functions → NXT I/O → Motor control**, в меню под иконкой выбираем **motor off → brake**) (рис. 8).

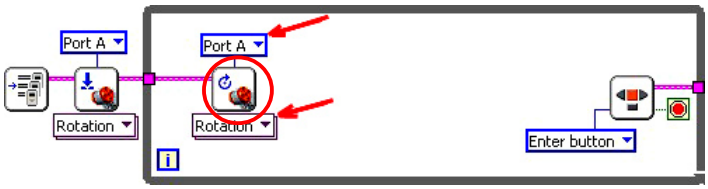


Рис. 4

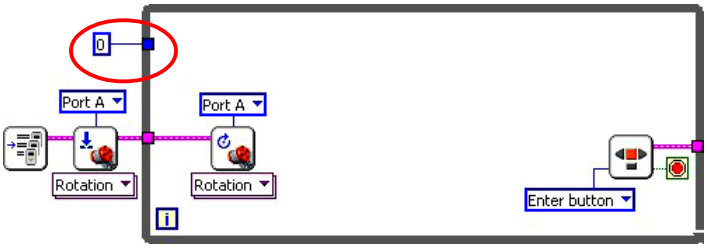


Рис. 5

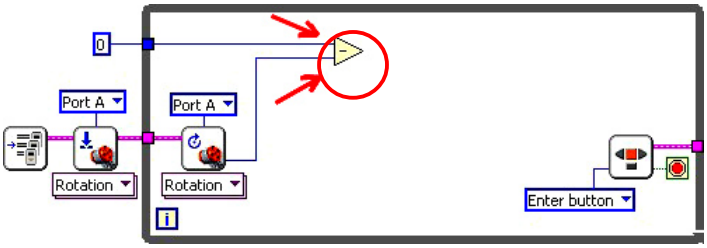


Рис. 6

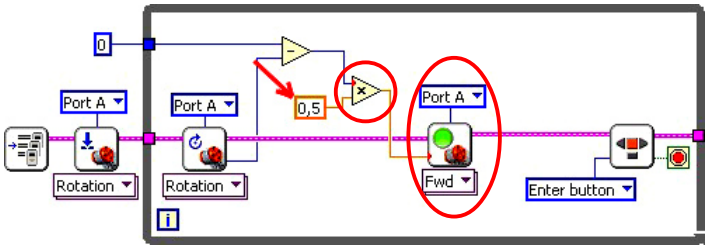


Рис. 7

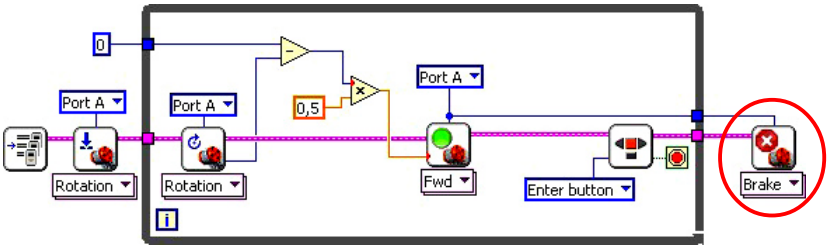


Рис. 8

Наша программа готова. Загружаем ее на NXT и проверяем ее работу.

Робот при запущенной программе ничем крутить не будет, но если вы попробуете повернуть ось двигателя A, то вы не сможете это сделать, так как программа автоматически будет противодействовать вашим попыткам повернуть ось мотора.

Поэкспериментируйте с константой 0,5 и сделайте выводы, как ее величина влияет на поведение робота при попытке повернуть ось мотора A.

РАБОТА ПРОГРАММЫ НА КОМПЬЮТЕРЕ С ИСПОЛЬЗОВАНИЕМ NXT

Мы уже научились писать программы, которые выполняются только на компьютере или только на NXT (рис. 9).

При выполнении программы на компьютере мы можем использовать лицевую панель для добавления элементов управления, контролов и индикаторов для управления программой и для отслеживания ее состояния, а также проводить отладку программы на панели редактирования диаграмм в процессе выполнения программы.

Когда мы записываем программу на NXT, то мы лишаемся возможности использовать лицевую панель программы, но зато программа может вести управление моторами и снимать показания с датчиков, подключенных к NXT.

Совместим плюсы обоих способов и организуем систему следующим образом (см. рис. 10).

Программа будет выполняться на компьютере, но при этом она будет управлять моторами и получать информацию с датчиков при помощи прямого подключения NXT к компьютеру через USB кабель.

Сохраняем нашу программу, написанную ранее, под другим именем.

Заходим в главное меню и меняем режим привязки программы на **Target to computer** (рис. 11).

На лицевой панели создаем контрол типа **Dial**, **NXT Robotics → Numeric → Dial**.

Переименовываем в «Задание положения в градусах» и меняем диапазон значений на диапазон, соответствующий 10 оборотам двигателя в градусах, то есть от 0 до 3600 (**properties → scale → scale range [minimum; maximum]**) (рис. 12).

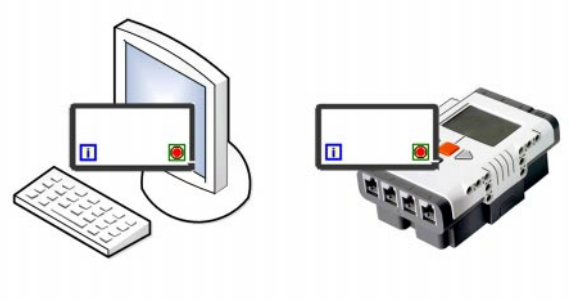


Рис. 9

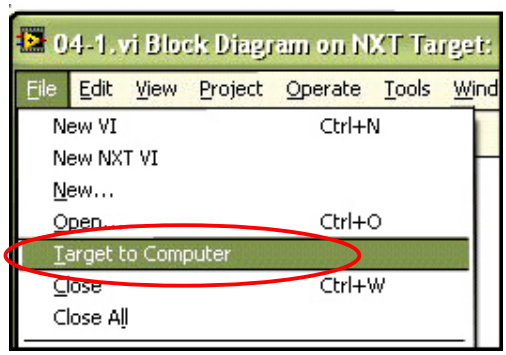


Рис. 11

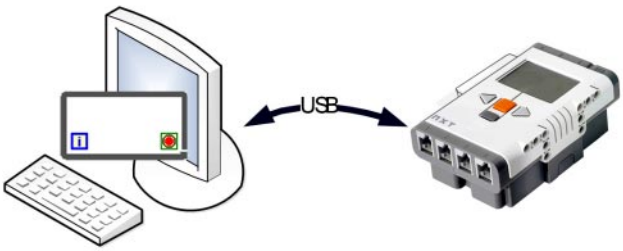


Рис. 10

Рис. 12

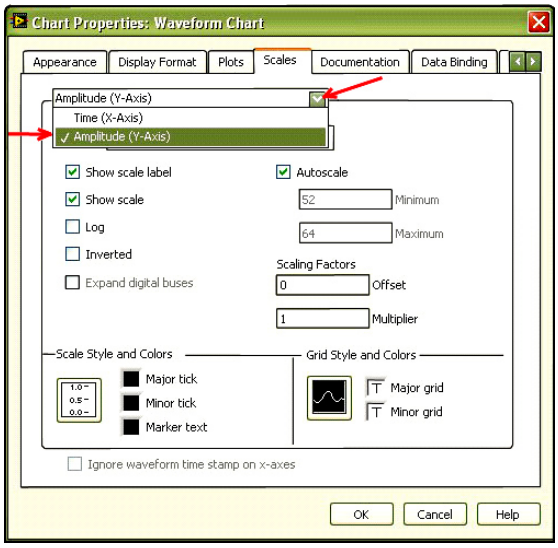


Рис. 14

Еще добавляем на лицевую панель новый для нас тип индикатора **Тренд (Chart) Graph → Waveform Chart** (рис. 13).

На этом индикаторе будут в виде графиков отображаться задание положения и реальное текущее положение мотора, поэтому необходимо настроить шкалу отображения по оси «Amplitude», чтобы мы могли наблюдать изменения значений во всем диапазоне, от 0 до 3600.

Кликаем по Waveform Chart правой кнопкой мышки **properties → scales**, далее в выпадающем меню выбираем **ось-Y** (**Amplitude (Y-Axe)**) (рис. 14).

Снимаем галочку **Autoscale**, а в графы **minimum**, **maximum** устанавливаем 0 и 3600. Нажимаем «Ок» (рис. 15).

Переходим на панель редактирования диаграмм.

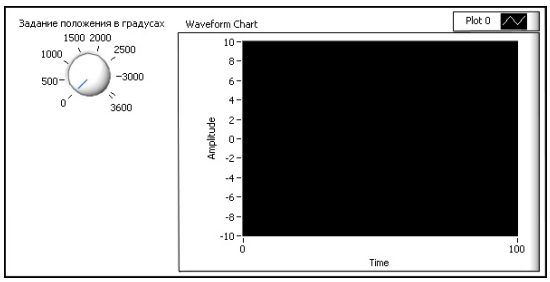


Рис. 13

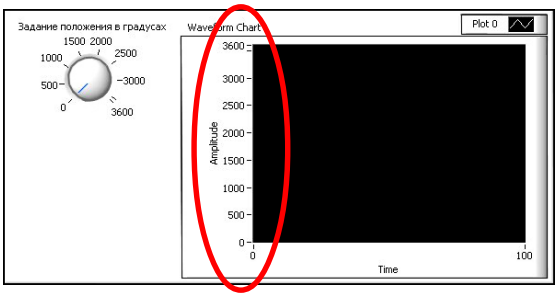


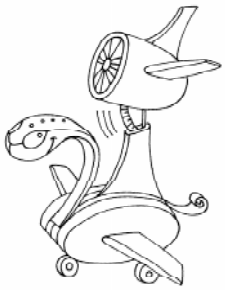
Рис. 15

Удаляем константу с заданием положения и туннель, ведущий от нее к функции вычитания. Присоединяем к освободившемуся входу функции вычитания, терминал «задание положения в градусах» (рис. 16).

Добавляем функцию **Bundle** (создать массив) **NXT robotics →NXT programming → Cluster → Bundle** (рис. 17).

Присоединяем к первому входу «Задание положения в градусах», ко второму – текущее значение энкодера и полученный кластер подаем на вход терминала **Waveform Chart** (рис. 18).

Кластер является структурой, группирующей данные, он может группировать дан-



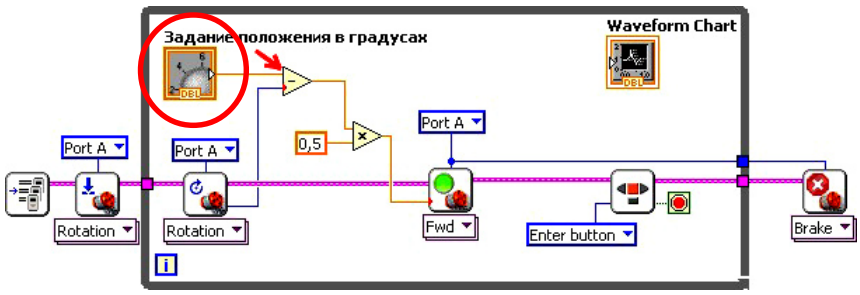


Рис. 16

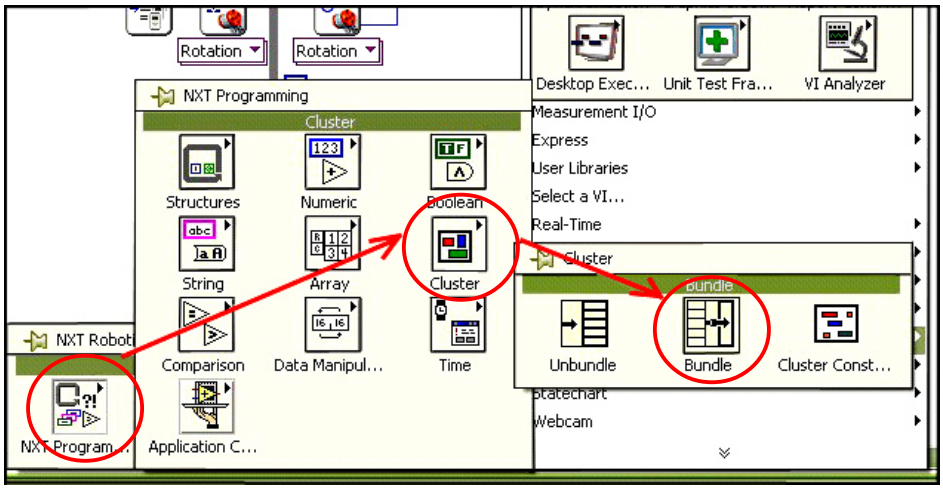


Рис. 17

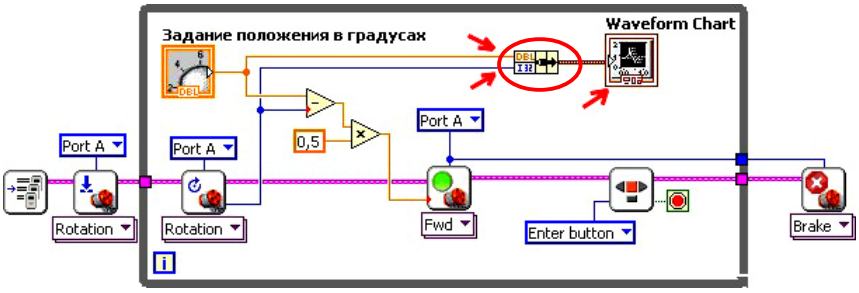


Рис. 18

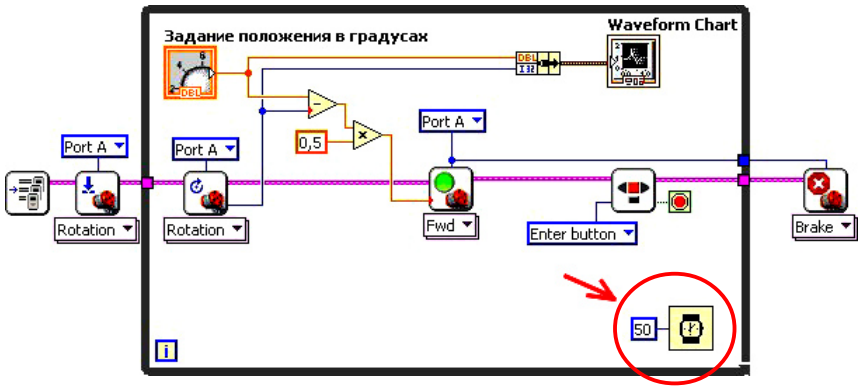


Рис. 19

ные разных типов, по аналогии, например, с реальными проводами и объединением их в кабель.

Теперь добавляем временную задержку в цикл для разгрузки центрального процессора, величину задержки выставим в 50 мс (рис. 19).

Запускаем программу. Изменяя задание контролом на лицевой панели, мы можем наблюдать на графике, как и насколько быстро мотор переходит в заданное положение.

ЭКСПЕРИМЕНТ

Проведем эксперимент, как влияет константа равная 0,5 на скорость перехода в требуемое положение и на точность этого перехода.

Рис. 20

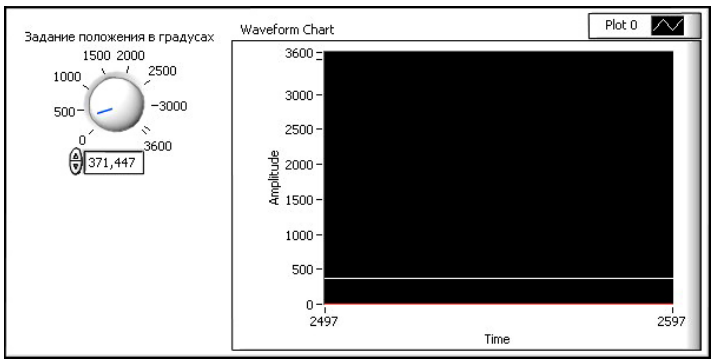


Рис. 21

На контроле «Задание положение в градусах» кликаем правой кнопкой мышки и выбираем **Visible Items → Digital display**. Рядом с ручкой контрола появится дисплей (рис. 20), на котором будет дублироваться значение, но в численном виде, также через него можно задавать значение, как мы это делали через обычный **Numeric control**.

Для удобства перемещаем дисплей под рукоятку управления (рис. 21).

Теперь мы можем мгновенно менять задание, изменяя значение в окошке дисплея.

Эксперимент будем проводить циклически для каждого значения константы из табл. 1.

Алгоритм проведения эксперимента:

1. Устанавливаем новое значение константы.

2. При помощи дисплея на лицевой панели устанавливаем задание, равное 0.

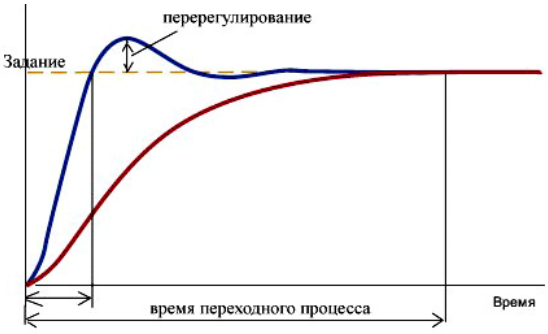


Рис. 22

3. Запускаем программу.

4. Набираем в дисплее 1800 градусов – это соответствует 5 оборотам двигателя.

5. Останавливаем программу, когда двигатель остановится.

6. Измеряем время, когда было выдано задание и когда мотор переместился в соответствии с этим заданием.

7. Находим разность этих значений и записываем в ячейку таблицы «Время переходного процесса».

8. Измеряем по графику (рис. 22) перерегулирование (то, насколько мотор проскочил задание).

Синий график (верхний) – перерегулирование есть, измеряем и записываем в таблицу, красный график (нижний) – перерегулирования нет, в таблицу записываем 0.

Возвращаемся к 1 пункту и повторяем все действия.

ЗАКЛЮЧЕНИЕ

В заключение этого урока мы немного «причешем» программу.

Терминалы контролов и индикаторов на панели редактирования диаграмм изначально после создания имеют большой размер,

Табл. 1

Пропорциональный коэффициент усиления	0,2	0,5	0,8	1	1,5	2	5	8	8	10
Время переходного процесса										
Перерегулирование										

для экономии места их можно уменьшить. Кликаем правой кнопкой на терминал и снимаем галочку **View As Icon**, после этой операции терминал уменьшится в размерах

Любая константа в программе должна иметь комментарий, поясняющий ее назначение. В нашей программе только одна константа, кликаем на ней правой кнопкой мышки и в меню выбираем **Visible Items →** **Label**. Рядом с константой появится окошечко с ее названием, в котором мы пишем латинскими буквами «Kp» Kp 0,5. «K» – означает «коэффициент усиления», а «p» – «пропорциональный». Это следует запомнить, так как в дальнейшем мы будем очень часто называть константы. Перетаскиваем название константы к левой стенке Kp 0,5.

Теперь наша программа приняла следующий вид (рис. 23).

На этом мы заканчиваем урок.

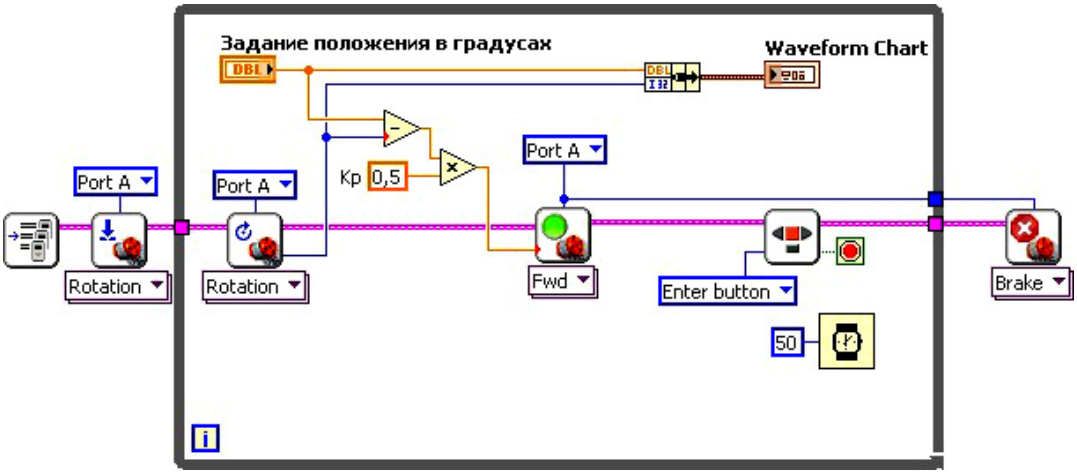


Рис. 23

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