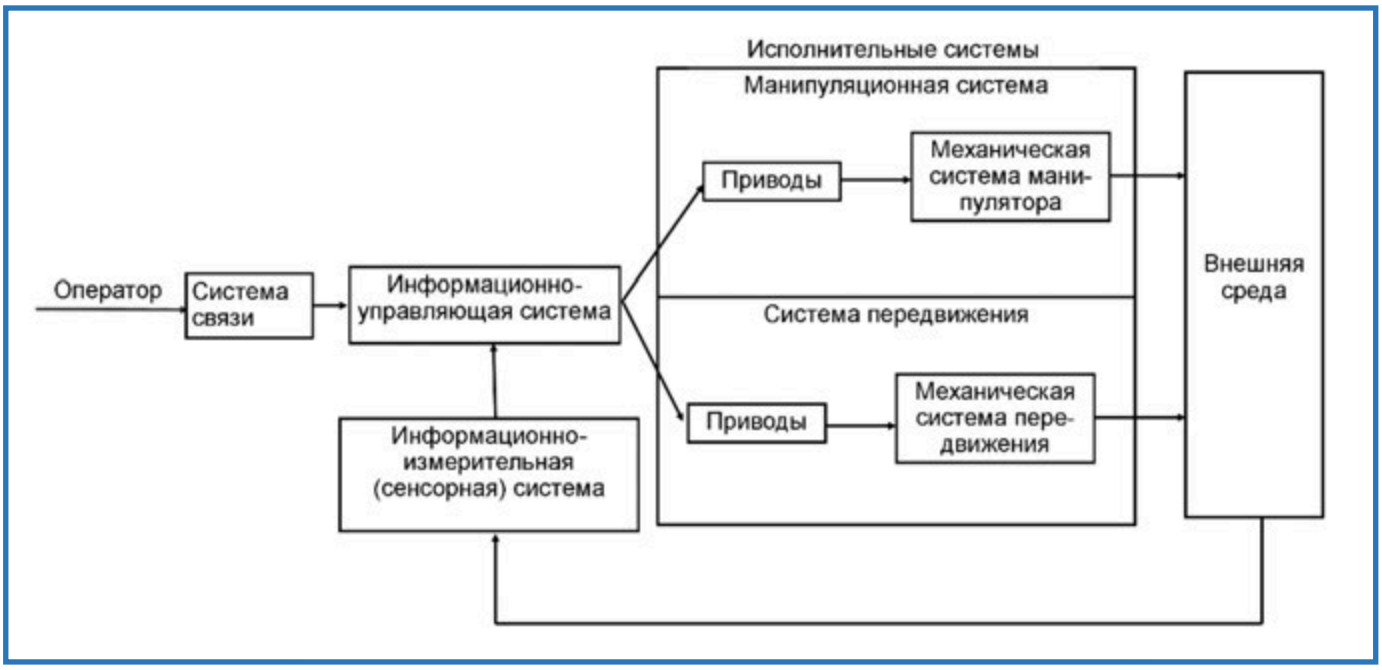
ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

РОБОТ. ЭЛЕМЕНТЫ РОБОТА. ПУЛЬТ ДИСТАНЦИОННОГО УПРАВЛЕНИЯ. ВЕТВЛЕНИЯ В С

Мы уже неоднократно использовали термин «робот», но еще ни разу не обсуждали определение этого понятия. Как и в случае со словом «технология», основная трудность здесь обусловлена тем, что робототехника как область человеческих интересов очень молода. К тому же сам термин пришел в технику из фантастической литературы. Все это привело к тому, что пока еще очень сложно отделить определение «робот» от близких и родственных понятий. В своем понимании вопроса мы будем опираться на функциональную схему робота, предложенную Е.И. Юревичем[1].

Давайте рассмотрим схему функционирования робота:



Робот состоит из трех основных частей. Это:

- информационно-измерительная система (ИИС);
- информационно-управляющая система (ИУС);
- исполнительная система (ИС).

[1] Юрьевич Е.И. Основы робототехники: учеб. пособие. - 4-е изд., перераб. и доп. - СПб.: БХВ-Петербург, 2017. - С. 1.

Как видно, функционирование робота построено циклическим образом. То есть ИИС считывает данные из внешней среды и передает их в ИУС. Там в свою очередь эти данные обрабатываются и принимается решение об управляющем воздействии на внешнюю среду. Управляющее воздействие осуществляется ИС, и оно заключается в изменении внешней среды. За этим изменением следит в свою очередь ИИС. Именно за счет такого построения и достигается «самостоятельное» автономное поведение робота.

Обратите внимание, что на рисунке в ИУС направлена еще одна стрелка из системы связи. С точки зрения робота, оператор – это внешняя среда, а система связи – ИИС. Стрелка, которая соединяет внешнюю среду и ИИС, называется обратной связью.

По вышеописанному принципу функционируют и устройства, называемые автоматами. Дискуссия о разнице между роботами и автоматами не входит в нашу задачу, но может стать темой для интересных рассуждений.

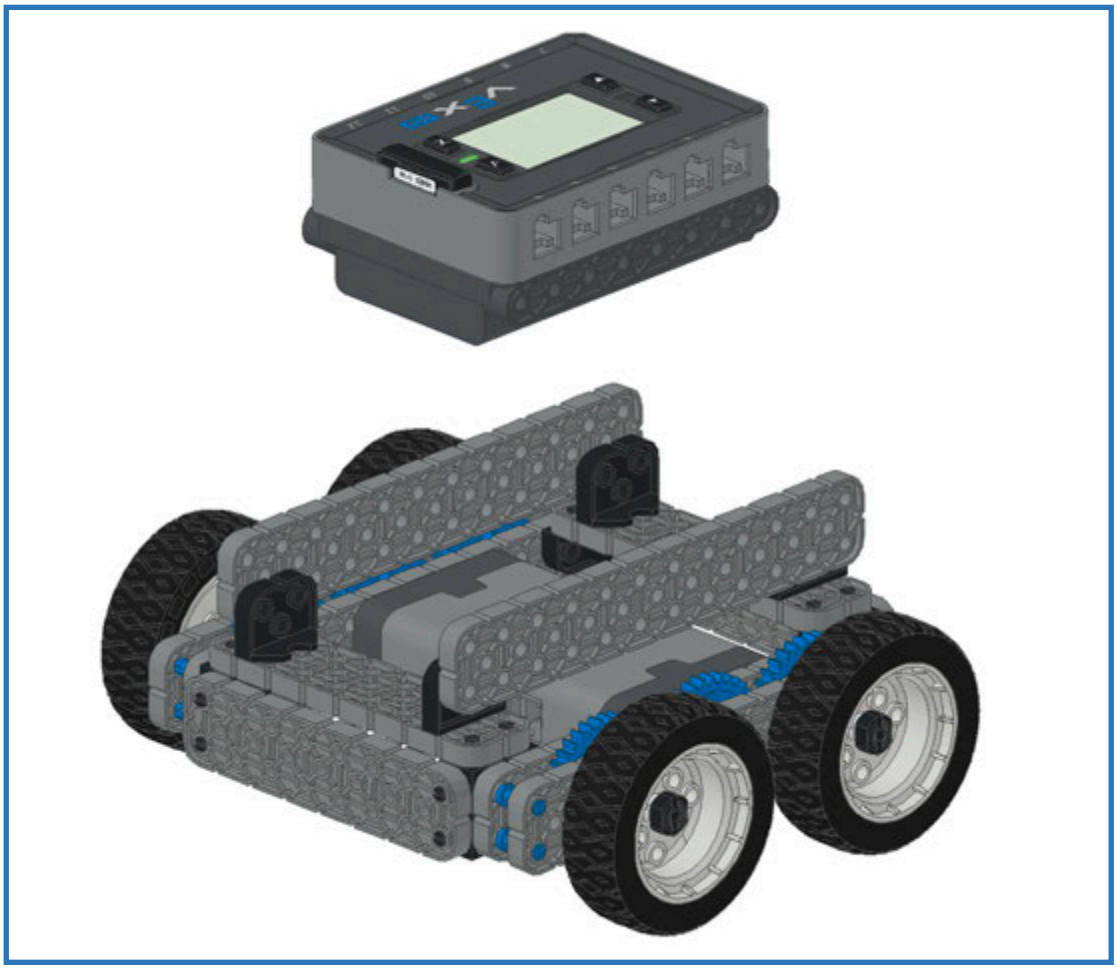
Для наглядности рассмотрим робота, который функционирует по нашей схеме.

Это **сигвей** - электрическое самобалансирующееся транспортное средство с двумя колесами, расположенными по обе стороны от водителя. В качестве ИИС в данном устройстве используется гироскопический датчик (не путать с гироскопом!), вернее, несколько гироскопических датчиков. ИС - это двигатели. Внешний мир для этого робота - это наклон робота. ИУС формирует такие управляющие воздействия, чтобы робот не находился в состоянии падения. То есть при заваливании вперед колеса вращаются таким образом, чтобы робот начал заваливаться назад, но при этом двигался вперед. И наоборот, при заваливании назад колеса стремятся опрокинуть робота вперед, робот начинает двигаться вперед.

Мы полагаем, что любое занятие по робототехнике целесообразно начинать с решения теоретических задач, таких как, например:

- климат-контроль;
- круиз-контроль;
- квадрокоптер;
- автофокусировка фотоаппарата.

Обратите внимание, что используемый на занятиях «мобильный робот» не является роботом в прямом смысле. В нем есть только две составляющие из трех: ИИС и ИС.



Важно понимать, что, начиная с этого занятия, мы различными способами будем осуществлять дистанционное управление. Однако до тех пор, пока не будет осуществлена обратная связь, использование термина «робот», строго говоря, будет некорректным. То, что мы называем роботом, правильнее было бы называть исполнителем.

Рассмотрим пульт дистанционного управления VEX IQ.

На нем имеются два стика. Для описания перемещения левого стика выбраны направления A и B, правого – D и C. На лицевой стороне расположены пары кнопок E и F. Левые и правые пары курки дистанционного управления обозначаются L и R. Обратите внимание, что в каждой паре одна из кнопок находится выше другой, а следовательно, в каждой паре кнопок есть верхняя и нижняя.

Любая из кнопок может принимать значение `0` (кнопка отпущена) или `1` (кнопка нажата), что соответствует значениям `true` (истина) и `false` (ложь). Стики же принимают значения от `-100` до `100` в любом из направлений.

В языке программирования RobotC состояние любой из кнопок и стиков пульта управления возвращает функция:

```
getJoystickValue();
```

Аргумент функции - это название кнопки или стика. Возвращаемое значение - 0 или 1 для кнопок, или интервал от -100 до 100 для направлений смещения стиков.

Осталось разобраться с тем, что такое название кнопки или стика, понятное языку RobotC.

Название направления перемещения стика всегда начинается с Ch, а затем добавляется большая буква, которая написана на пульте. Например, название направления отклонения левого джойстика от пользователя или к нему - это ChA.

				
Название	ChA	ChB	ChD	ChC
Возможные значения	[-100,100]	[-100,100]	[-100,100]	[-100,100]

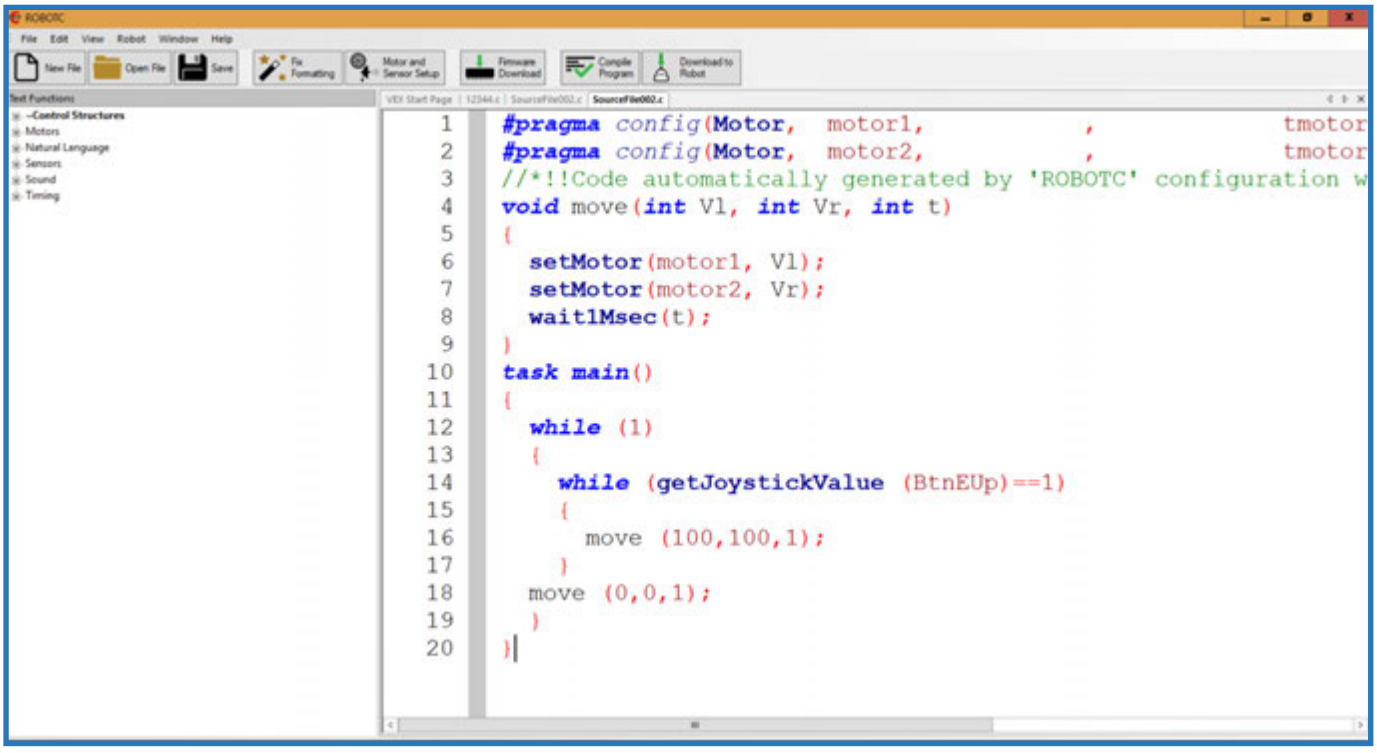
Теперь перейдем к кнопкам:

В случае кнопок название формируется следующим образом: сначала **Btn**, затем название пары кнопки с заглавной буквы, затем указание, верхняя это или нижняя кнопка из пары, тоже с заглавной буквы. Например, **BtnLUp** - это кнопка, левый курок, верхняя.

				
Название	**BtnEUp**	**BtnEDown**	**BtnFUp**	**BtnFDown**
Возможные значения	1/0	1/0	1/0	1/0

				
Название	**BtnLUp**	**BtnLDown**	**BtnRUp**	**BtnRDown**
Возможные значения	1/0	1/0	1/0	1/0

Но вернемся к нашему роботу (вернее, как мы выяснили, пока еще только исполнителю). Ниже представлен код, по которому робот будет при нажатии на кнопку **BtnEup** двигаться вперед, при отпускании - стоять на месте.

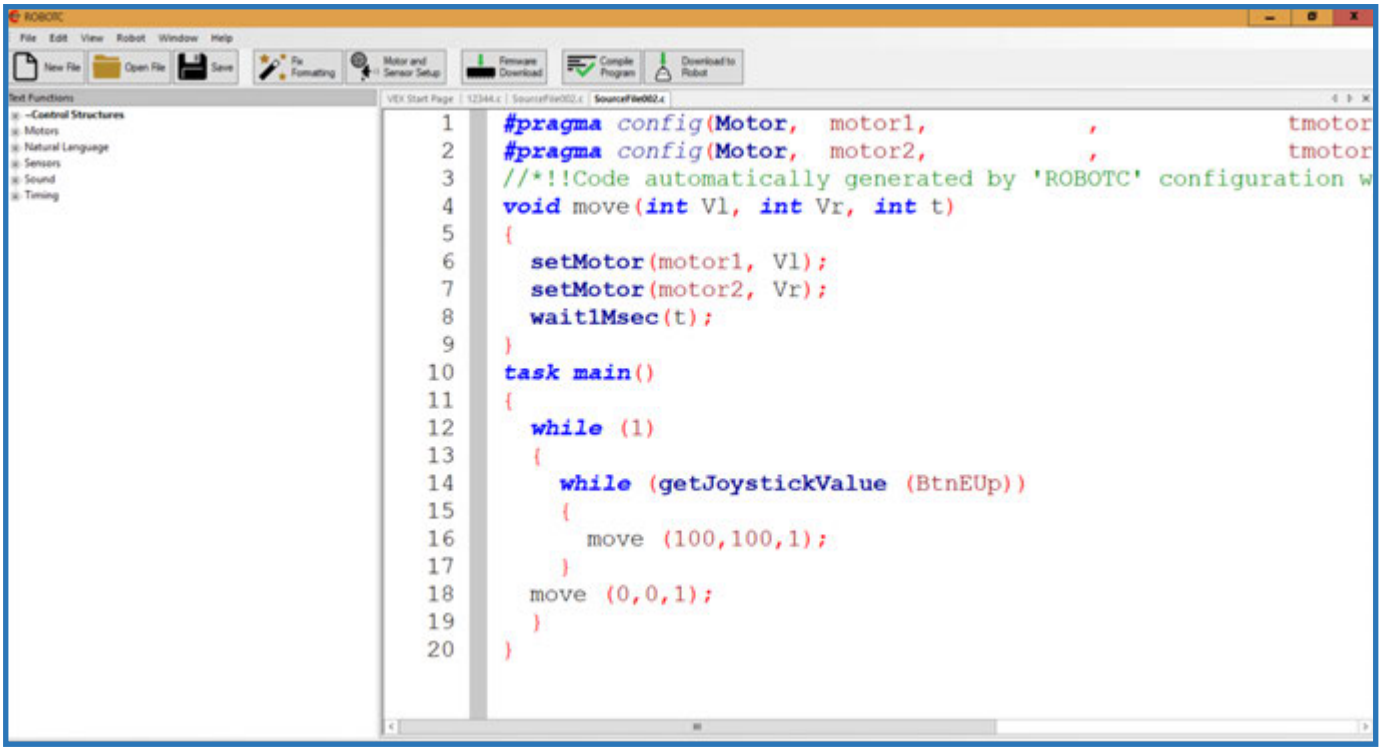


```
1  #pragma config(Motor,  motor1,           ,             tmotor
2  #pragma config(Motor,  motor2,           ,             tmotor
3  //*!!Code automatically generated by 'ROBOTC' configuration w
4  void move(int Vl, int Vr, int t)
5  {
6    setMotor(motor1, Vl);
7    setMotor(motor2, Vr);
8    wait1Msec(t);
9  }
10 task main()
11 {
12   while (1)
13   {
14     while (getJoystickValue (BtnEUp)==1)
15     {
16       move (100,100,1);
17     }
18   move (0,0,1);
19   }
20 }
```

Разберем код подробнее. В 12 строке объявляется бесконечный цикл, его аргумент всегда равен единице. Тело этого цикла описано между 13 и 19 строками, где находятся открывающаяся и закрывающаяся фигурные скобки.

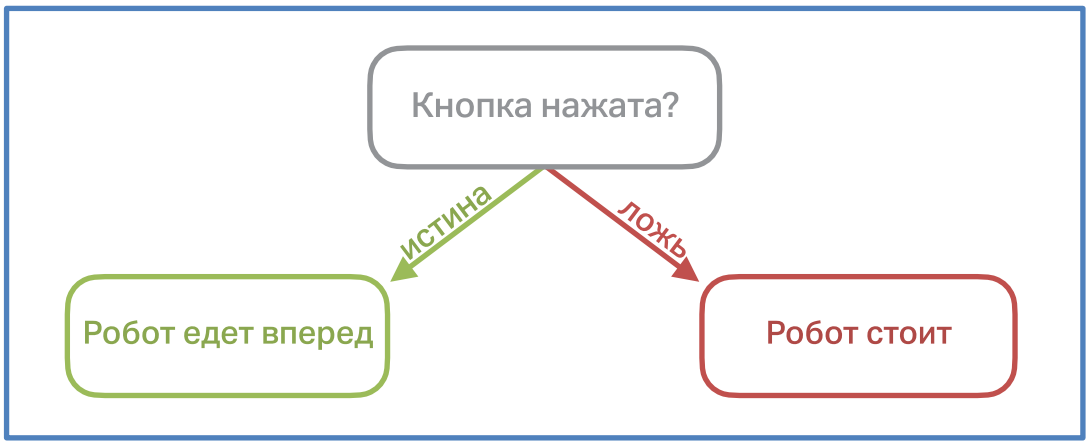
В 14 строке находится вложенный цикл. Он так называется, потому что вложен в цикл, объявленный в строке 12. Его аргумент - это результат операции сравнения значения, которое возвращает функция `getJoystickValue(BtnEUp)` с числом `1`. Как говорилось выше, эта функция возвращает `1`, если кнопка нажата, `0` – если она отпущена. То есть в случае нажатия кнопки будет выполняться строка 16, пока кнопка не будет отпущена. Если же кнопка отпущена, программа перейдет к строке 18 и робот будет стоять на месте.

Учитывая, что аргументом `while` являются `0` и любое другое значение, а `getJoystickValue(BtnEUp)` - `0` и `1`, программу можно представить следующим образом:

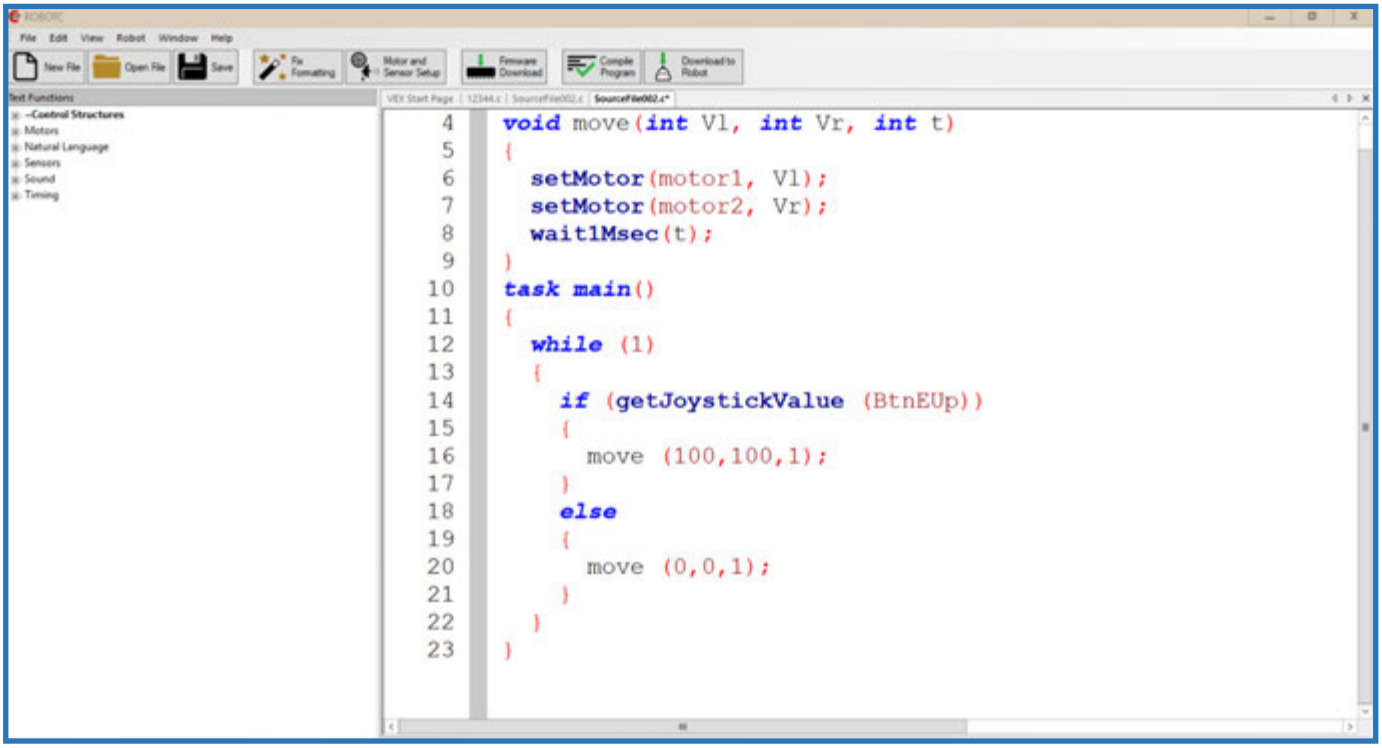


```
1  #pragma config(Motor,  motor1,           ,             tmotor
2  #pragma config(Motor,  motor2,           ,             tmotor
3  //*!!Code automatically generated by 'ROBOTC' configuration w
4  void move(int Vl, int Vr, int t)
5  {
6    setMotor(motor1, Vl);
7    setMotor(motor2, Vr);
8    wait1Msec(t);
9  }
10 task main()
11 {
12   while (1)
13   {
14     while (getJoystickValue (BtnEUp))
15     {
16       move (100,100,1);
17     }
18   move (0,0,1);
19   }
20 }
```

В нашей задаче есть два состояния: кнопка нажата (робот едет вперед) и кнопка отпущена (робот стоит). Графически это можно выразить в виде дерева:



Для задач, в которых есть два состояния, в языке С существует специальное выражение: оно называется **`if else`**, или **если иначе**. Код программы в этом случае примет следующий вид:



```
4   void move(int Vl, int Vr, int t)
5   {
6     setMotor(motor1, Vl);
7     setMotor(motor2, Vr);
8     wait1Msec(t);
9   }
10  task main()
11  {
12    while (1)
13    {
14      if (getJoystickValue (BtnEUp))
15      {
16        move (100,100,1);
17      }
18      else
19      {
20        move (0,0,1);
21      }
22    }
23  }
```

Как видно из кода выше, **`if else`** - это тоже, по сути, функция. Ее аргументами, как и в функции **`while`**, являются два возможных значения: **`false`** (ложь, или 0) или **`любое другое значение`**. В случае, если аргумент равен любому значению кроме `0`, выполняются действия между фигурными скобками, которые следуют сразу после **`if()`**, в остальных случаях выполняются действия в фигурных скобках после **`else`**.

В нашем случае, если истинно то, что кнопка нажата, робот перемещается, в обратном случае - стоит на месте.

Такой подход в программировании называется **ветвлением**, потому что его графическое представление похоже на дерево с ветками, где от каждой ветки отходят еще ветки, а на ветвях располагаются листья.

Итак, **робот - это машина автоматического действия, которая может взаимодействовать с окружающей средой на физическом (силовом) и информационном уровнях**. Если же устройство не осуществляет посредством ИУС обратную связь, а решение об управляющем воздействии на окружающую среду принимается при внешнем вмешательстве, то мы имеем дело с программируемым исполнителем. При управлении таким устройством используется пульт дистанционного управления, работа которого в большинстве случаев программируется с помощью ветвления. Ветвление помогает осуществить выбор одного из двух вариантов в зависимости от заданного условия.