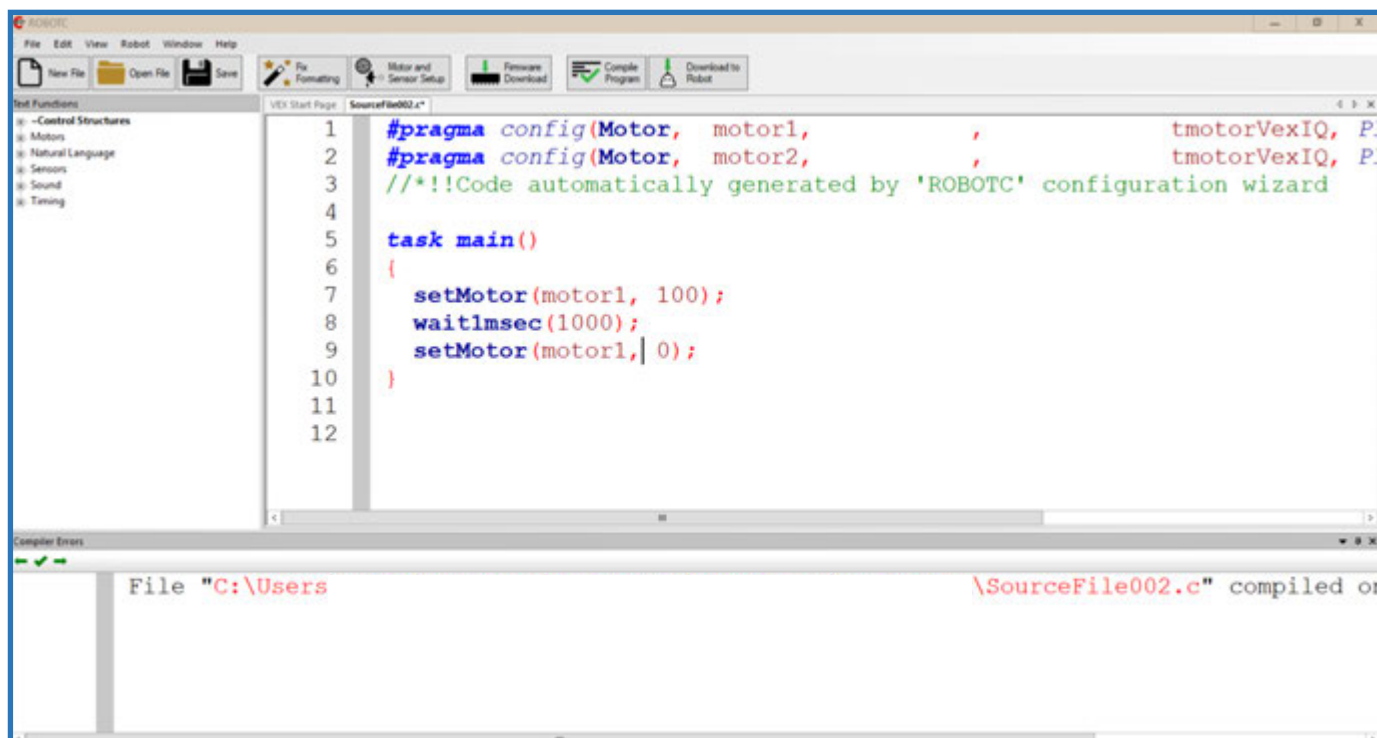


ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ КОНСТРУКЦИЯ ПОЛНОПРИВОДНОГО РОБОТА VEX IQ. ПРОГРАММИРОВАНИЕ ПОСТУПАТЕЛЬНОГО И ВРАЩАТЕЛЬНОГО ДВИЖЕНИЯ

На прошлом занятии мы начали знакомиться с языком программирования C, который, как мы убедились, является языком функций. Кроме того, мы разобрали несколько специальных функций для языка программирования роботов RobotC. Но в этом языке есть не только специальные функции, позволяющие выводить информацию на экран робота. Одной из возможностей RobotC является управление двигателями. Для этого существует специальная функция `setMotor(motorPort, speed);`.



```
1 #pragma config(Motor, motor1, , tmotorVexIQ, P
2 #pragma config(Motor, motor2, , tmotorVexIQ, P
3 /*!!Code automatically generated by 'ROBOTC' configuration wizard
4
5 task main()
6 {
7     setMotor(motor1, 100);
8     wait1msec(1000);
9     setMotor(motor1, 0);
10 }
11
12
```

File "C:\Users \SourceFile002.c" compiled o

У этой функции два аргумента: первый - это номер порта, в который подключен двигатель, второй - скорость вращения этого двигателя в относительных единицах. Значения аргумента скорости находятся в интервале от **-100** до **100**. Знак минус изменит направление вращения, а при скорости вращения, равной **0**, двигатель будет остановлен.

Важно! Для того чтобы корректно работать с любыми подключенными устройствами, необходимо их инициализировать в меню **Motors and Sensors setup**.

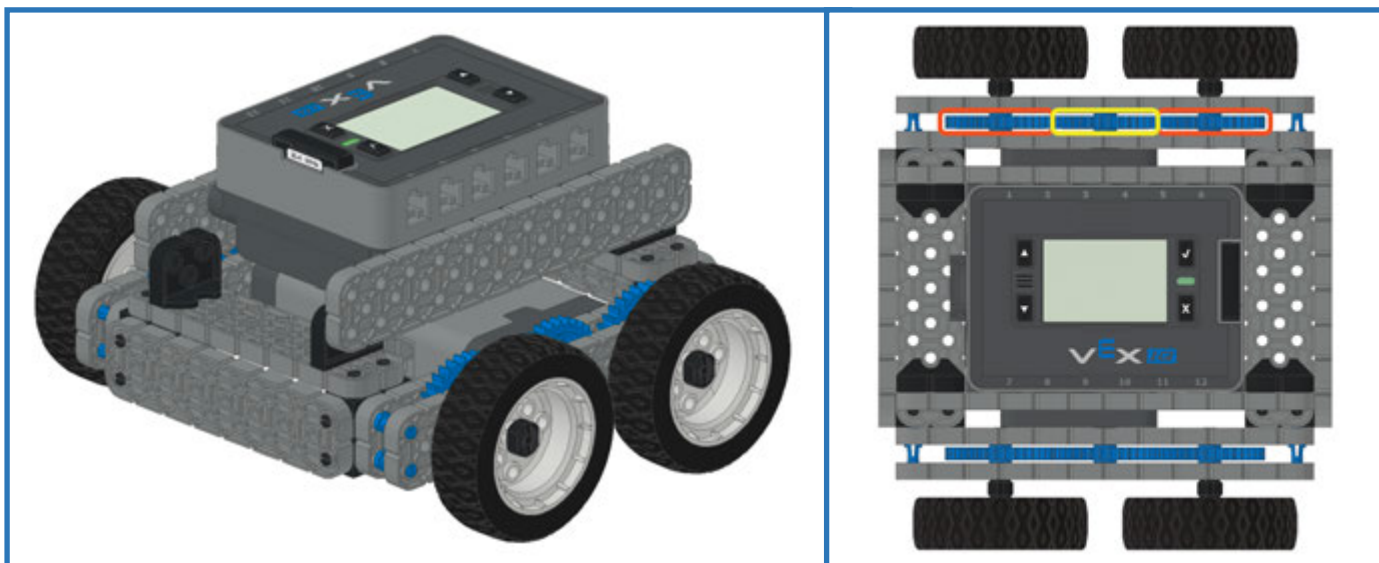
Согласно приведенной выше программе робот выполнит следующие инструкции:

- в 7 строке двигатель, подключенный в **port1**, будет запущен по часовой стрелке;
- в 8 строке новая для нас функция **wait1Msec(1000) ;;**
- в 9 строке осуществляется остановка двигателя.

wait1Msec(1000) ; - это функция ожидания; ее аргумент - это время в миллисекундах, от которого зависит время ожидания. В нашем случае это 1000 миллисекунд или одна секунда. Все это время двигатель, подключенный к **port1**, будет вращаться.

Настало время применить полученные знания из области конструирования, механики и программирования для решения задач, связанных с движением мобильного робота.

Для наших целей подойдет полноприводный четырехколесный робот VEX IQ:

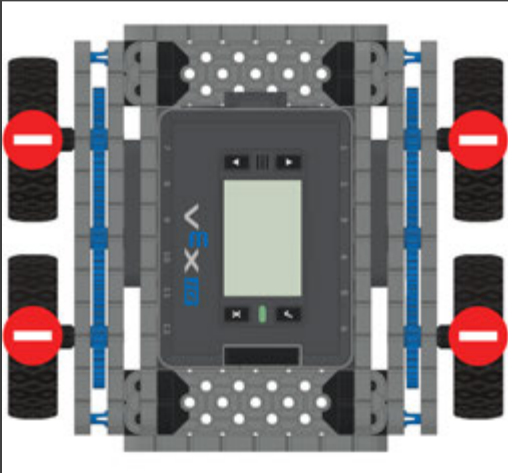
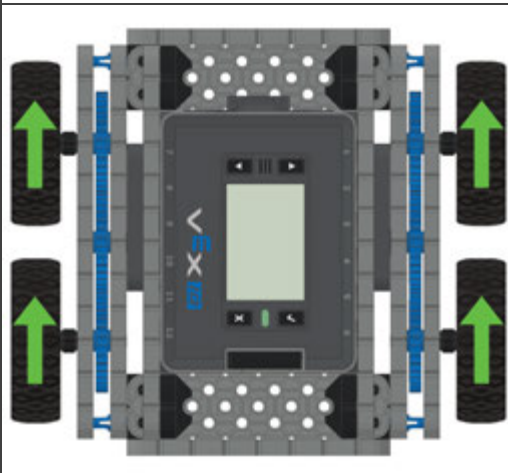
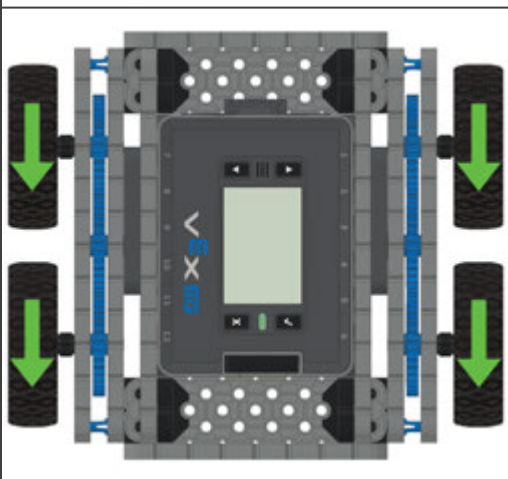


Для реализации такого робота необходимо левую и правую пары колес сделать зеркально. Привод на колеса осуществляется через зубчатую передачу, состоящую из трех зубчатых колес, как показано на рисунке. Обратите внимание на то, что ведущее колесо в зубчатой передаче является средним, а ведомые колеса соединены с колесами робота.

Предлагаемая конструкция робота является жесткой и прочной за счет того, что все поперечные и продольные пластины соединены в нескольких местах.

Важной характеристикой сконструированного робота является и расположение его центра масс между всеми четырьмя колесами, что позволяет совершать точные маневры.

Для такого робота все возможные варианты движения представлены в таблице ниже. Зеленые стрелки – это направления движения верхних точек колес. Если эти направления сонаправлены с положением робота, то считаем, что двигатель включен вперед, если противоположны – назад.

	<p>Оба двигателя остановлены.</p>	<p>Робот не двигается.</p>
	<p>Оба двигателя включены в одну сторону.</p>	<p>Робот перемещается вперед.</p>
	<p>Оба двигателя включены в одну сторону.</p>	<p>Робот перемещается назад</p>

	<p>Двигатели включены в разные стороны.</p>	<p>Робот разворачивается на месте против часовой стрелки. Ось разворота пересекает оси вращения колес и находится на равном расстоянии от них.</p>
	<p>Двигатели включены в разные стороны.</p>	<p>Робот разворачивается на месте по часовой стрелке. Ось разворота пересекает оси вращения колес и находится на равном расстоянии от них.</p>
	<p>Левый двигатель включен вперед, правый остановлен.</p>	<p>Робот поворачивает вокруг правого колеса по часовой стрелке.</p>
	<p>Правый двигатель включен вперед, левый остановлен.</p>	<p>Робот поворачивает вокруг левого колеса против часовой стрелки.</p>

	<p>Правый двигатель включен назад, левый остановлен.</p>	<p>Робот поворачивает вокруг левого колеса по часовой стрелке.</p>
	<p>Левый двигатель включен назад, правый остановлен.</p>	<p>Робот поворачивает вокруг правого колеса против часовой стрелки.</p>
	<p>Скорость левого двигателя меньше, чем скорость правого двигателя. Оба двигателя включены вперед.</p>	<p>Робот движется по дуге вперед налево.</p>
	<p>Скорость правого двигателя меньше, чем скорость левого двигателя. Оба двигателя включены вперед.</p>	<p>Робот движется по дуге вперед направо.</p>

	<p>Скорость правого двигателя меньше, чем скорость левого двигателя. Оба двигателя включены назад.</p>	<p>Робот движется по дуге назад направо.</p>
	<p>Скорость левого двигателя меньше, чем скорость правого двигателя. Оба двигателя включены назад.</p>	<p>Робот движется по дуге назад налево.</p>

Из таблицы видно, что для движения вперед двигатели робота необходимо запустить в одну сторону. Как это сделать, показано в коде, приведенном ниже.

```

1  #pragma config(Motor,  motor1,  ,  ,  tmotorVexIQ, P
2  #pragma config(Motor,  motor2,  ,  ,  tmotorVexIQ, P
3  /**!!Code automatically generated by 'ROBOTC' configuration wizard
4
5  task main()
6  {
7      setMotor(motor1, 100);
8      setMotor(motor2, 100);
9      wait1msec(1000);
10 }
11
12

```

File "C:\Users\... SourceFile002.c" compiled ok

Однако, если запустить данный код, робот вместо передвижения вперед начнет вращаться на месте. Это связано с тем, что двигатели в нашей конструкции расположены

зеркально относительно друг друга. Для того чтобы наш код позволял роботу двигаться вперед, необходимо в меню **Motors and Sensors setup** поставить галочку в поле **reverse** для **motor2**.

Итак, использование различных аргументов функции `setMotor()` позволяет реализовать все возможные варианты движения робота, представленные в таблице выше: вперед, назад, направо по часовой стрелке и направо против часовой стрелки, налево по часовой стрелке и налево против часовой стрелки, вперед направо, вперед налево, назад направо, назад налево, на месте по часовой стрелке и на месте против часовой стрелки.