RobotC Reference

Prepared for ENGR 120
Spring Term, 2011
University of Victoria

Jonah Wall

Introduction

RobotC is an *extended subset* of C programming language, which means that there are some C libraries that will NOT work in RobotC. There are also some functions available in RobotC that are not part of standard C.

The RobotC API features a sidebar containing the most commonly used structures from C that are also supported in RobotC, along with the library that is specific to RobotC. Many of the functions listed will not be used in this course, and have been omitted from this guide.

Sensors and Motors

motor[motor_name] = power;

To use this function, first you need to set up your motor.

- Go to **Robot > Motor and Sensors setup**
- Select the **Motors** tab
- Enter a name for your motor (Ex: elev_motor)
- Note the PORT NUMBER of the motor
    - Connect your motor to the corresponding port in the bank of ports on the VEX controller outlined in green and marked "motors"
- Hit OK

Valid power levels range between -127 and 127. Negative values put the motor in reverse, positive values drive it forwards, and 0 stops the motor

Example:

```
motor[my_motor] = 100;
```

Note: *motors automatically engage an internal brake when no power is applied to them*

SensorValue(sensor_input)

As with the motor function, you first need to set up your sensors before using this function.

- go to **Robot > Motor and Sensors setup**
- Select the **A/D input 0-8** tab
- Enter a name for your sensor
- From the pull-down sensor, select the type of sensor you are connecting
    - For the round buttons, use **touch sensor**

Timing functions

VEX has 4 internal timers: T1, T2, T3, and T4. The following functions should be used in place of any C functions as the four VEX timers are NOT the same as the internal clocks on a PC processor.

wait1Msec(wait_time)

Delays the execution of the next statement by **wait_time** milliseconds.

- useful if you want to ensure that your robot does not move for a preset amount of time
    - for example, when an elevator arrives at a floor, it must wait a predefined amount of time before leaving the floor to make sure people have to enter and leave elevator
- ***hint:*** your system is *not monitoring inputs* throughout **wait_time**

time10[timer]

Returns an integer representing the current value of **timer** (specified by T1, T2, T3, or T4) in *hundredths of seconds.*

Example:

```
/* store the value of T1 in the variable hundredth_seconds */
hundredth_seconds = time10[T1];
```

time100[timer]

Returns an integer representing the current value of **timer** in *tenths of seconds.*

Example:

```
/* store the value of T1 in the variable tenth_seconds */
tenth_seconds = time10[T1];
```

ClearTimer(timer)

Resets **timer** to zero.

Example:

```
/* Reset T1 to zero */
ClearTimer(T1);
```

Misc.

random(value)

Generates a random integer between 0 and **value**

Example:

```
#define MAXVALUE          10;

/* pick a random integer between 0 and 10 */
random(MAXVALUE);
```

bool

Boolean type – defined with two possible values: true (1), or false (0)

See Also

VEX Support Page
http://www.robotc.net/support/vex/

Similar RobotC Page
http://www.robotc.net/education/curriculum/vex/pdfs/reference_guide.pdf

RobotC forums
http://www.robotc.net/forums/

Make sure you only use content in the **VEX** and **IFI forums** for this course! NXT, RXC, and MindStorms are different platforms, and while there is some overlap, not instructions cross over to IFI, which we are using.