Using Robots to Teach Technology Engineering

- Chris Beaton Technology Engineering Teacher Ashland High School
- Griffin Whittredge- Ashland High School Class of 2011
- Barry Biletch- Ashland High School Class of 2013
- Tim Abraham- Ashland High School Class of 2011

Using Robots to Teach Technology Engineering

- Objectives
  - Understand the use of hardware and software, specifically the iRobot Create, Botball CBC KISS C as a tool in solving problems using the Engineering Design Process
  - Provide an opportunity for you to think about exciting students about STEM

Kiss C

- C-Based Programming Language developed to support Botball CBC
  - Built in Library Functions ensure ease of use
  - Has a built in simulator and compiler for debugging

Using the Hardware

- iRobot Create

- Botball CBC v.2

iRobot Create

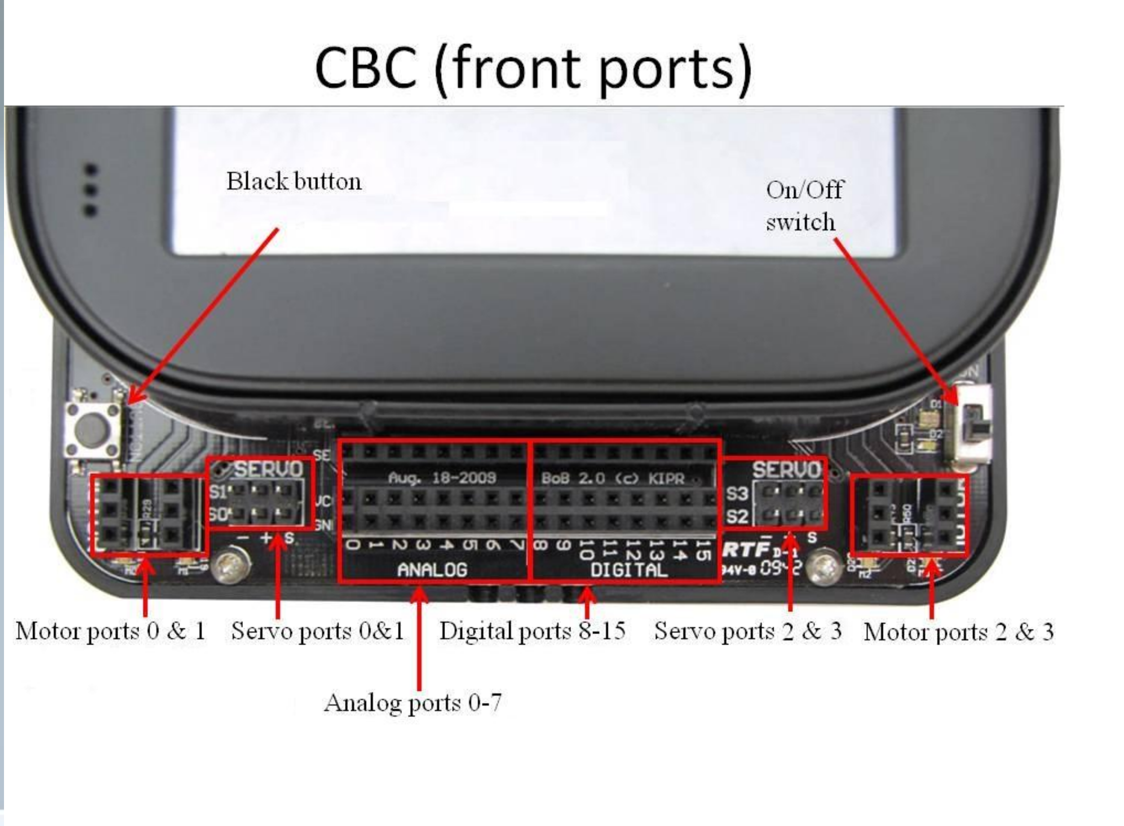
- Sturdy and reliable
- Easy to use
- Fast to get up and running
- Inexpensive (Relatively speaking)

CBC Processor

- Has a touch screen which can be very useful to students and teachers
- Has built in functionality
- Is simple to use.
- Software and updates are free!!!

Ports on the CBC

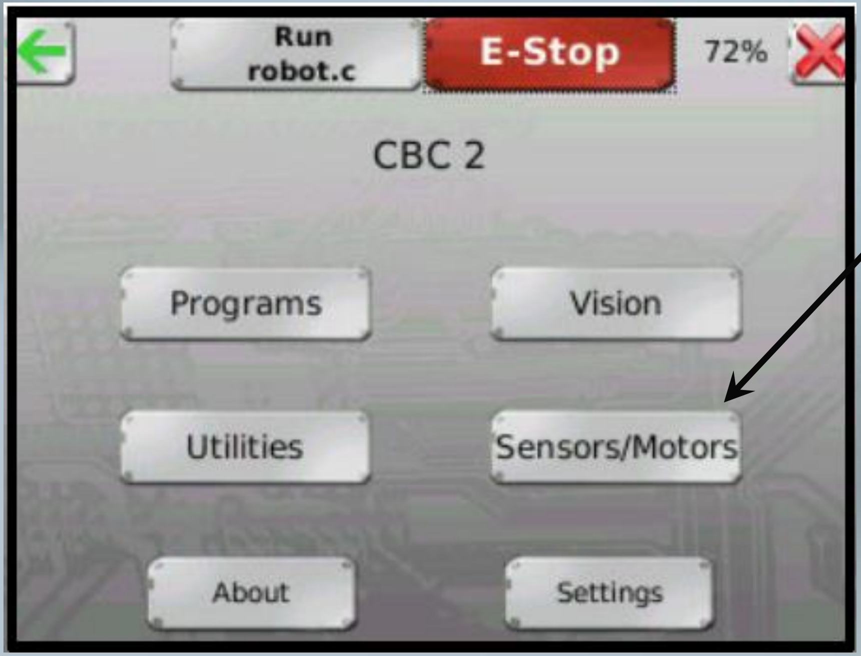
- If trouble arises with sensors, check the programmed port number with the physical port number



CBC (front ports)

CBC Sensors and Touch Screen

CBC Opening Screen

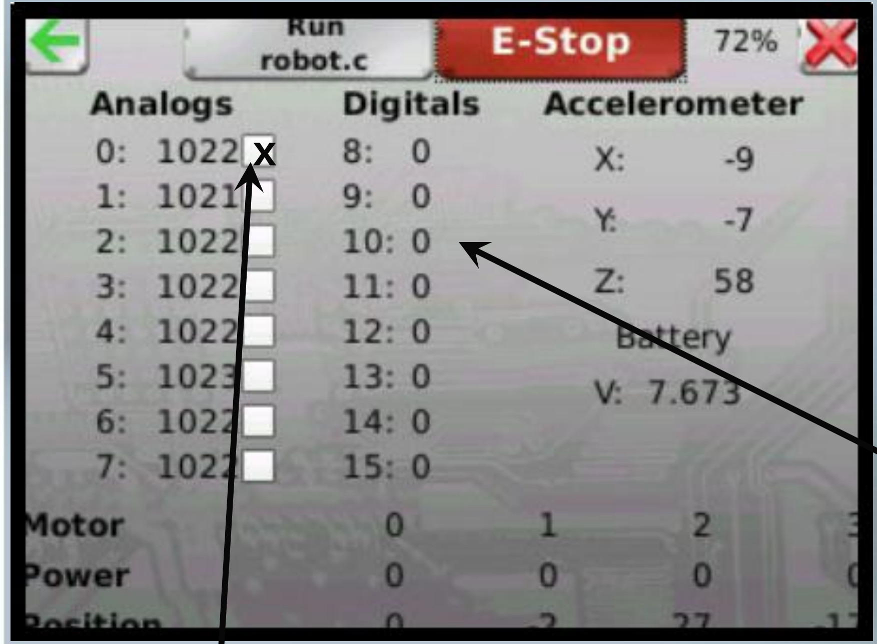


Sensors and Motor Button

TOUCH this button

Select the sensors option on the next screen

CBC Sensors and Touch Screen (cont.)



Analogs		Digitals	Accelerometer
0: 1022	☒	8: 0	X: -9
1: 1021	□	9: 0	Y: -7
2: 1022	□	10: 0	Z: 58
3: 1022	□	11: 0	Battery
4: 1022	□	12: 0	V: 7.673
5: 1023	□	13: 0	
6: 1022	□	14: 0	
7: 1022	□	15: 0	

Motor	0	1	2	3
Power	0	0	0	0
Position	0	-2	27	-17

Provides user with “real time” feedback of sensor values

To view digital values, push the touch sensor. The value of the port(s) should change to 1. In this case digital 8

To view analog values, check (touch) the white checkbox which corresponds to the port(s) being used. In this case Analog 0

Using KISS C

- Click on KISS C icon
- Select CBC2
- Select Cancel on the port screen
- You should now be at the editor screen

Coding example:

Copy and paste this code into the KISS-C software

```
/*This code is designed to demonstrate the function of a while loop in KISS-C

while(condition)
{
	code to execute
}

A while loop will loop through the code in it's brackets until the condition in the parenthesis becomes false

msleep(time)
this command will let the robot do nothing for the specified time(milliseconds), if a motor is moving, this command
does not tell the motors to stop, they will continue to spin

This code will beep continually until the black button on the CBC is pushed*/

int main()
{
	while(!black_button())//loops while the black button is not being pushed
	{
		beep();//a beep will be heard
		msleep(500);//this is so that the different beeps can be heard separately
	}
	return 0;
}
```

Using an external digital sensor

```
/*
This code uses the command digital(port number) this command
    checks the port number specified for sensors that
only return an on or off value, a 1 or a 0
*/

int main()
{
    while(!digital(8))//loops while the joystick sensor is not being
    pushed (if it is not working, try holding it, and check the port
    number)
    {
            beep();//a beep will be heard
            msleep(500);//this is so that the different beeps can be heard
    separately
    }
    return 0;
}
```

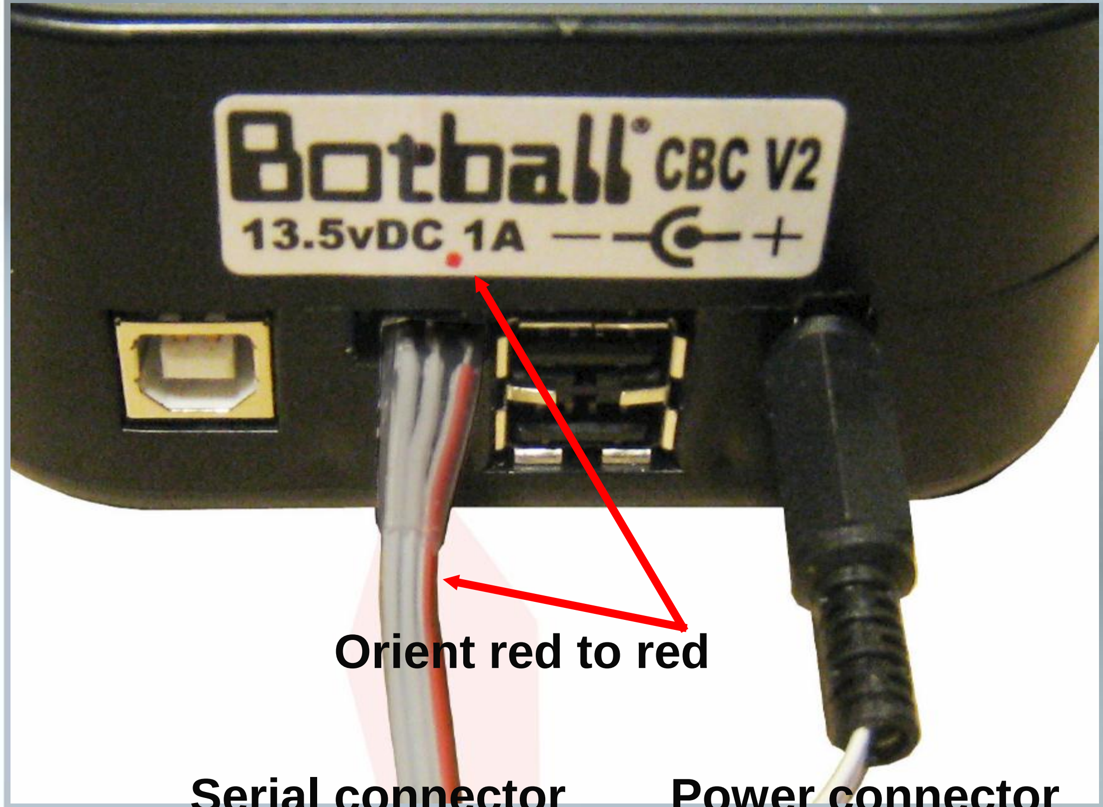
Displaying messages on the CBC

```
/*
This code will beep continually and display: Please push the black button!!!! repeatedly,
      then, when the
black button on the CBC is pushed it will display: Thank You wait 2 seconds and clear the
      display
*/

int main()
{
      while(!black_button())//loops while the black button is not being pushed
      {
                 printf("Please push the black button!!!!\n");//this command displays the
      message in the quotations, the  \n goes to the next line of display space
                 beep();//a beep will be heard
                 msleep(500);//this is so that the different beeps can be heard separately
                 cbc_display_clear();//this command clears the display
      }
      printf("Thank you");
      msleep(2000);
      cbc_display_clear();
      return 0;
}
```

Motor control and sleep commands

```
/*
This code will demonstrate how to use sleep functions and how to use a motor

motor(port number,speed)
this command will tell whatever motor is in the port specified to spin at the speed(0-1000) specified
a positive speed and negative speed will spin opposite directions

ao()
this command stops all motors (all off)

*/

int main()
{
    motor(0,100);
    msleep(1000);
    ao();
    return 0;
}
```

Connecting CBC-Create Cable



Orient red to red

Serial connector

Power connector

Once the cable is plugged into the CBC leave it be since yanking the serial connector out could damage the plug

Getting the Roomba to move

```
/*This code is designed to drive until the black button is pushed

create_connect();
this is the command to initialize communications between the cbc and the create

create_disconnect();
this is the command to end communications between the cbc and create

create_drive_straight(speed 0-500);
this is a command line that tells the create to drive at the specified speed

create_stop();
this is the command line that tells the create to stop moving
*/

int main()
{
    create_connect();
    create_drive_straight(500);//this command is never countered, so putting it in the while loop is
    unnecessary because it is only needed once
    while(!black_button())
    {
        msleep(10);
    }
    create_stop();//when there is only one command in a while loop, brackets are not needed, but
    tabbing it over is conventional
    create_disconnect();
    return 0;
}
```

Using the Roomba’s touch sensors

```
/*
gc_lbump and gc_rbump are the names for the built-in touch sensors
    on the create
create_bumpdrop() is the comand line that updates those values*/

int main()
{
    create_connect();
    create_drive_straight(500);
    create_bumpdrop();//initial update of sensor values
    while(!(gc_lbump||gc_rbump))//this will loop until the right bumper
    OR the left bumper is pushed
           create_bumpdrop();//this is used in the while loop to update
    the sensor values continually
    create_stop();
    create_disconnect();
    return 0;
}
```

Using an analog sensor with the Roomba

```
/*
This code uses an analog sensor which returns a variable value depending upon
    the function of the sensor

this code is intended for use with the tophat sensor, which measures reflectivity

the command analog(port number) is used to check the value of an analog
    sensor*/

int main()
{
    create_connect();
    create_drive_straight(500);
    while(analog(0)<225)//checks analog sensor(if does not work, check port
    number)
            msleep(10);
    create_stop();
    create_disconnect();
    return 0;
}
```

Sample line tracking

```
/*
this is simple line tracking code

create_spin_CW(speed) and create_spin_CCW(speed)
this command will spin the robot in place at the speed specified(0-500) they spin ClockWise and
CounterClockWise respectively

*/

int main()
{
    create_connect();
    //note: this code is an example for line tracking, it does not work if the bot starts in the starting box
    while(!black_button())
    {
        if(analog(0)>225)//checks if the dark line is detected by the tophat sensor
        {
                create_spin_CW(20);//spins the robot in place
        }
        else//otherwise(if no line is detected)
        {
                create_drive_straight(20);//drives forward
        }
    }
    create_stop();
    create_disconnect();
    return 0;
}
```

Solving an open ended Problem

Ω Click on the link below

https://sites.google.com/a/ashland.k12.ma.us/robotics-club/bull-in-the-ring-stream