RedBot

An introduction to robotics using the SparkFun RedBot Board

Derek Runberg, Jeff Branson

Overview

- . 9:00-9:15- Introductions
- . 9:15-9:30- Arduino/RedBot
- 9:30-10:00-Install Arduino/Libraries/Blink
- . 10:00-10:30- Build the chassis
- 10:30-11:00-Drive in a straight line, PWM slow and fast, backwards, pivot
- 11:00-12;00-control motor speed with pot, serial write motor speed to the com port, add line sensor

Overview continued

- . 12:00-1:00- Lunch
- . 1:00-1:30- Sharp Infrared sensor
- . 1:30-2:00-add servo to Infrared
- . 2:00-3:00- Xbee
- . 3:00-5:00- Openswim,

What's Arduino?

- . Open Source, Free
- . Supports lots of hardware
- . Outgrowth of Processing
- . Ardublock, Minibloq
- . Named after a bar.
- . We like 1.0.5



RedBot library Download

- The libraries to run the motor and sensor functions can be down loaded from two places;
- The RedBot product page <u>https://www.sparkfun.com/products/12097</u>

Or

SparkBox ,log into SparkBox and browse the files, download the redbot arduino library file.

Installing the library

- The library will be unzipped to the Arduino sketchbook in the documents folder on your computer.
- The file path will look something like;
- Documents>Arduino>libraries
- The location of your sketchbook location can be found under file>preferences in the Arduino environment as well.

Example code for this class

- Go to SparkBox and down load the folder that is labeled "redbot example code".
- These examples will help navigate through the basics of the RedBot library.



The RedBot Mainboard

.Arduino Uno - Atmega 328

- 5V Logic
 6-9V Input Voltage via Barrel
 Jack or 2-Pin Header
- . TB6612FNG Dual DC Motor Driver
- . USB Programmable via FTDI
- . XBee Port with HW/SW Serial Switch
- . 2x 3-Pin Female Motor Port
- . 2x 6-Pin Male Optional Servo Header
- 4x 6-Pin Male Optional Sensor Header





Blink on the RedBot board

- . Let's open Arduino and get started programming.
- . We'll use the following file path;
- . file>examples>basics>blink
- we should see something like the following slide



```
- 0 X
💿 Blink | Arduino 1.0.3
File Edit Sketch Tools Help
ø
                                                                                                      •
 Blink
/*
  Blink
  Turns on an LED on for one second, then off for one second, repeatedly.
 This example code is in the public domain.
 */
// Pin 13 has an LED connected on most Arduino boards.
// give it a name:
int led = 13;
// the setup routine runs once when you press reset:
void setup() {
  // initialize the digital pin as an output.
pinMode(led, OUTPUT);
.
```

🚱 🚞 🌻 🔮 🖉 💽 💿

Arduino Uno on COM21



Now to select the board

ds.



// Pin 13 has an LED conne
// give it a name:
int led = 13;

// the setup routine runs
void setup() {
 // initialize the digita
 pinMode(led, OUTPUT);



ATtiny45 (internal 1 MHz clock) ATtiny45 (internal 8 MHz clock) ATtiny45 (external 20 MHz clock) ATtinv85 (internal 1 MHz clock) ATtinv85 (internal 8 MHz clock) ATtiny85 (external 20 MHz clock) ATtiny44 (internal 1 MHz clock) ATtinv44 (internal 8 MHz clock) ATtinv44 (external 20 MHz clock) ATtinv84 (internal 1 MHz clock) ATtiny84 (internal 8 MHz clock) ATtiny84 (external 20 MHz clock) Arduino Uno Arduino Duemilanove w/ ATmega328 Arduino Diecimila or Duemilanove w/ ATmega168 Arduino Nano w/ ATmega328 Arduino Nano w/ ATmega168 Arduino Mega 2560 or Mega ADK Arduino Mega (ATmega1280) Arduino Leonardo Arduino Esplora Arduino Micro Arduino Mini w/ ATmega328 Arduino Mini w/ ATmega168 Arduino Ethernet Arduino Fio Arduino BT w/ ATmega328 Arduino BT w/ ATmega168 LilvPad Arduino USB LilyPad Arduino w/ ATmega328 LilyPad Arduino w/ ATmega168 Arduino Pro or Pro Mini (5V, 16 MHz) w/ ATmega328 Arduino Pro or Pro Mini (5V, 16 MHz) w/ ATmega168 Arduino Pro or Pro Mini (3.3V, 8 MHz) w/ ATmega328 Arduino Pro or Pro Mini (3.3V, 8 MHz) w/ ATmega168 Arduino NG or older w/ ATmega168

ne second, repeatedly.

Arduino Uno on COM21

🔺 🏴 🙀 .atl 🏟

7:47 PM

5/7/2013

ø.

Select the COM port



Now to upload to the board

. We'll click on the right arrow button



Watch the "RX" and "TX" lights on the board to see your code moving to the board.

Let's make some changes to the code

- // the setup routine runs onc
 void setup() {
- // initialize the digital r
 pinMode(led, OUTPUT);
 }

```
// the loop routine runs over
void loop() {
  digitalWrite(led, HIGH);
  delay(1000);
  digitalWrite(led, LOW);
  delay(1000); Let's change this
}
```

The Chassis



Let's build the chassis

Modifications to the chassis-1

• You'll need to thread the 3/4" standoffs through the chassis as shown;



Adding the servo plates

We'll add two 3/8' standoffs to mount the servo plate



Mounting the line following sensors

• We'll need to mount two plastic standoffs on the chassis for the line follow sensors





Mount Sharp IR Sensor

• We need to add two standoffs for the Sharp distance sensor



Modifications to the chassis-2

• You'll need to splice the right side and left side motors together. The red wires will go together and the black wires will go together



Adding header pins to the Wires

- . We'll need to solder header pins to the wires
- . Don't forget shrink wrap



We'll add the board to the chassis



Adding the barrel jack to the battery pack

- Slip the jacket over the wires before soldering the wires to the barrel jack
- This is a center positive connector, the red wire solders to the center terminal and the black wire goes to the outside terminal.



Time to drive!

- Open the code in the redbot examples folder called "redbot first robot "
- . Load the code to your RedBot and
- see what happens.
- What happens when we change the delays?



Let's add the sensors



Working with motor.drive

- The motor.drive function can be passed numbers 0-255
- What happens when we change the numbers?
- If we copy the code block and put in different numbers, what bappens?



Some basics

- . The microcontroller is the brain
- . The sensors are the eyes ears and touch of the robot
- . The motors are the arms and legs
- . The motors are controlled by the microcontroller and can react to the

sensors.



Driving a motor

. The Arduino chip isn't powerful enough to drive a motor, so we use transistors



But what about turning in both directions?

 To switch the flow of current we have to reverse the direction the current flows through the motor. To do this we use an array of transistors, an H-bridge.



Let's see what happens when we change speed

- Open redbot second robot and see what the code does.
- The value that's being written to the motors is changing, why?
- How can we influence the rate of change using math operators?
- How do we know where the math stats and stops?
- How do we see the numbers that are being used?

Turning

- Let's use the pivot function from the library to get the bot to turn
- We'll open the redbot_third_robot sketch and load it to the board
- We can pass the motor.pivot command numbers between -100 and +100
- Positive numbers turn in one direction and negative numbers in the opposite direction

Reading a sensor

- We can read input from the world in two forms, digital and analog
- We'll use the infrared sensor as an input, it emits infrared light and looks for the value of the light reflected from an object
- The sensor pins on the board are pre-wired
- . to power the emitter and the detector

Analog Values

- Let's open the redbot_fourth_robot sketch and load the code
- The range of values we see is 0-1023, this corresponds to a voltage. 0=no voltage, 1023= 5volts
- We'll read the sensor on pin A0, after loading the code open the serial monitor, the magnifying glass in the upper right hand corner



Using an input and output

Lets build this circuit



Adding code to the circuit

- Let's open the redbot_fifth_robot sketch
- . We'll load the code
- The potentiometer will control the motor speed
- Why have we used a math operator on the analog read?



Wiring the sensors

- The sensors will wire to A0 and A1
- Make sure the VCC connection on the sensor goes to 5V on the board
- The GND connection will go to GND



Calibrating the line following IR sensor

- In order to get the line follower sensing the line we'll need to do some calibration.
- Mount the IR sensors to the bottom of the chassis
- Read each sensor through the serial terminal by passing them over the tape line on the courses

Adding the Servo pt.1

• We'll mount the servo to the frame using the plate and the layout in the photo







Adding the Servo pt.2

- . Mount the servo horn to the servo
- The sensor will mount to the servo horn with hot glue, see the picture, wire and tape are also an option





Adding wires to the sensor

- We'll need to add some wires to plug into the board, or jump through the breadboard.
- We try to keep colors coordinated, but wire color makes no difference.

Servo Code

- Let's calibrate the code and start a program to use the infrared sensor
- We can start by adding a Serial.print function and read the values when the bot approaches the wall
- From there we need an if statement to decide what to do

Free Swim

- Questions, specific task and project time
- . Xbee
- Processing and robotics
- Bump switches