



Many NearSys kits are programmed through a three pin header soldered to the PCB. Since a three pin receptacle is not a common termination for a serial cable, this kit contains the parts to make one. In addition, most people want to program their NearSys product as soon as it's completed. This kit therefore also contains the parts necessary to make a LED traffic light, piezo speaker, temperature sensor, and light sensor. This kit enables you to observe the results of your first PICAXE code by programming a model traffic light, producing tones over a piezo speaker, measuring the temperature, and detect changing light levels.

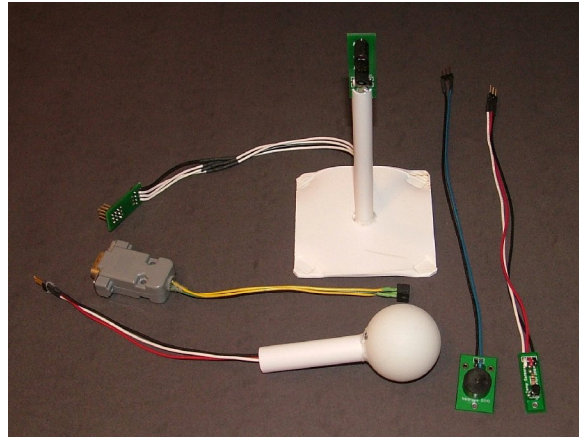


Figure 1. The completed programming kit

List of Parts

- Female DB-9 connector with solder cups
- DB-9 housing kit
- Three 680 ohm resistors (blue, gray, brown, gold)
- 22k ohm resistor (red, red, orange, gold)
- 1k ohm resistor (brown, black, red, gold)
- Piezo speaker
- Photocell
- LM335 temperature sensor
- Three light-emitting diodes (red, green, and yellow)
- Three 1 by 3-pin headers
- 3 by 3-pin header
- 2 by 3 pin receptacle
- AWG 24 stranded wires
- Heat shrink tubing
- Nylon flange bearing
- 3 inch by 3 inch Syntra sheet
- 4 inches of ½ inch diameter plastic tubing

4 inches of 3/8 inch diameter plastic tubing
1-1/4 inches of 1/4 inch diameter plastic tubing
Ping pong ball
Four PCBs

Required Tools

Wire cutters
Wire strippers
Soldering iron
Hot air gun
Small Phillips screwdriver
Hot glue gun
Pliers

Programming Cable Procedure

First, build the programming cable. You'll need the DB-9 connector, DB-9 housing, wires, 2 by 3 receptacle, and heat shrink tubing. The DB-9 connector is a solder cup style. This means there are open cups on the back of the DB-9 where wires are soldered to the connector. The three wires solder to the cups labeled with numbers 2, 3, and 5. Five is ground, three is serial-in to the PICAXE, and two is serial-out to the PC. Look at the back of the DB-9 connector and identify the proper solder cups.

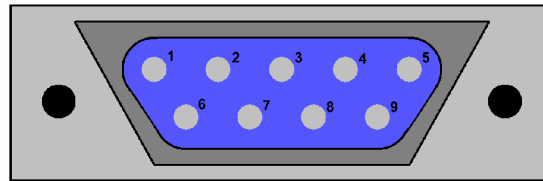


Figure 2. The back of a female DB-9 connector looks like this. Be sure you can identify the numbers printed next to the solder cups before soldering any wires.

Electrically it won't matter which wire is soldered to which cup. However, the orientation of the programming receptacle does matter when you try to program a microcontroller. Therefore, we will use the single black (or green) wire as the ground indicator.

- Strip 1/4 inch of insulation from both ends of all three wires
- Twist the bare ends of the wires tightly and tin them
- Prop the DB-9 under a weight, in a helping hands, or have someone hold it steady
- Insert one end of the black wire into solder cup 5, hold it steady, and solder it
- Continue holding the wire steady until the solder cools
- Repeat with the other two wires in cups 2 and 3
- Review the soldering to make sure no cups were shorted out

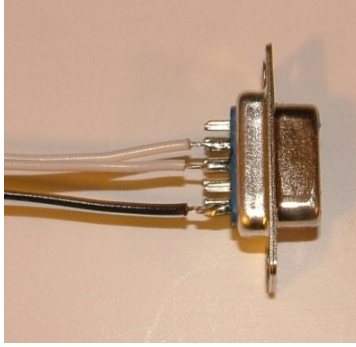


Figure 3. Three wires soldered to the DB-9 connector. Note that the black wire is solder to position #5.

The 2 by 3 pin receptacle is cut from a wider receptacle, so it has one or two rough edges. Sand the rough edges smooth with sand paper laid flat on a table. Although not absolutely necessary, it does make the programming cable look more attractive. Rather than solder to just three pins in the receptacle, we'll solder to all six. That way it's not important which row of the receptacle you plug into your microcontroller project.

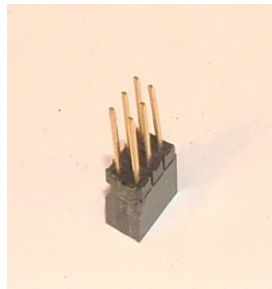


Figure 4. The receptacle has two rows of three pins on the back. The pins are fairly long and must be cut shorter.

- Cut one row of pins to half their length
- Bend the shorter pins until they contact the long row of pins

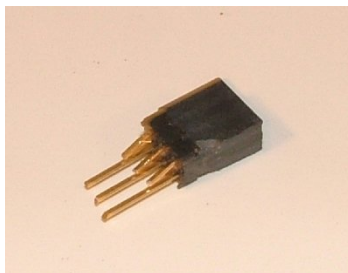


Figure 5. The short pins after being bent to touch the longer row. In this image, the longer row of pins were also trimmed slightly.

- Solder each contacting pin together

- Tin the pins of the receptacle
- Cut the heat shrink tubing into three equal pieces
- Slide heat shrink over each wire and slide it towards the DB-9 connector
- Solder a wire to each long pin of the receptacle in their **proper** order

Note: The order in which the wires are soldered to the receptacle is important. Solder the wire from solder cup #2 wire to one of the end receptacle pins of the 2 by 3 header, the wire from solder cup #3 to the middle pin of the 2 by 3 header, and the wire from solder cup #5 to the other end pin of the receptacle. Figure 6 shows how the wires are properly soldered to the 2 by 3 header.

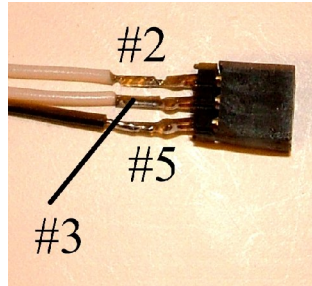


Figure 6. The proper order for the wires. Which side is left and which is right is not important. What is important is that the #3 wire is the middle wire.

- Slide the heat shrink over the soldered pins and shrink

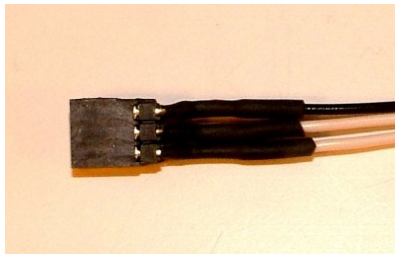


Figure 7. The receptacle pins covered in heat shrink tubing.

- Squirt a thin coating of hot glue around the solder cups on the back of the DB-9 connector

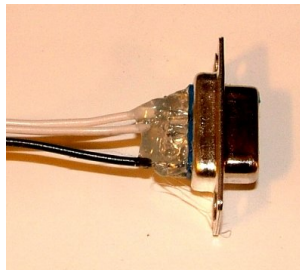


Figure 8. A coating of hot glue will prevent shorts and strengthen the connection between the DB-9 and wires.

- Fill the one side of the DB-9 housing with hot glue
- Before the glue can cool, place the DB-9 inside

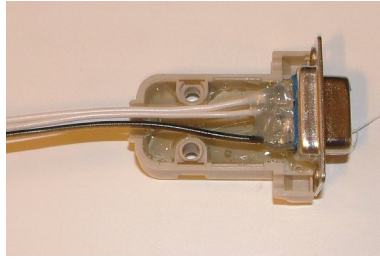


Figure 9. The DB-9 connector inside a glue filled housing half.

- Fill the other half of the housing with hot glue and before it can cool, close it over the other housing half and DB-9 connector
- Bolt the housing halves together
- Fill the opening in the back of the closed housing with hot glue

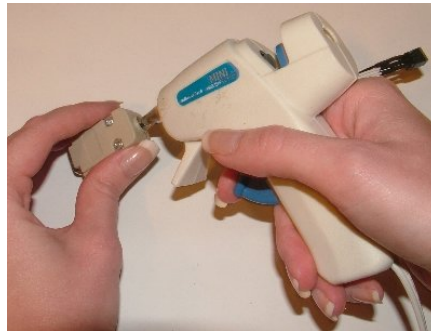


Figure 10. Finish filling the DB-9 housing by squirting glue into the opening in the back of the housing.

After the glue cools, the programming cable is ready to use.

Traffic Light

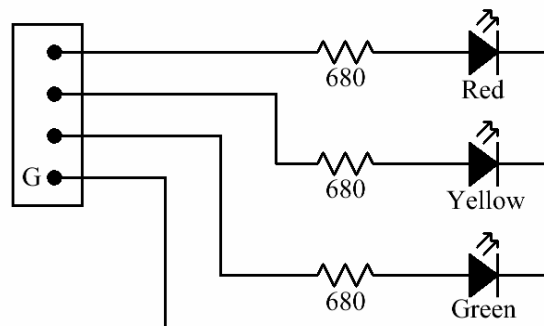


Figure 11. Traffic light schematic.

The traffic light consists of three LEDs and three resistors that limit the current flowing through each LED. By applying a voltage to one of the LEDs, it illuminates. Voltage to illuminate an LED comes from an I/O pin of the robot controller and each LED has a separate I/O pin through the 3 by 3 pin traffic light plug. Only four pins in the plug are used, the other five are soldered to the plug, but are not electrically connected to the LEDs. The robot controller uses the HIGH command to apply a voltage on the LED and the LOW command to stop applying a voltage. Applying a voltage lets current flow to the LED. After flowing through a resistor and its LED, current returns to ground through the fourth pin in the traffic light plug.

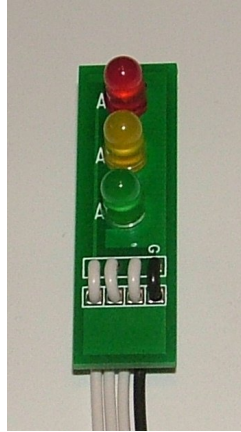


Figure 12. The head of the traffic light with its three LEDs and four wires.

The traffic light has two PCBs. The first is the traffic light head that contains the three colored LEDs. A cable from the traffic light head connects it to the traffic light plug.

- Solder the red LED to the head PCB's top circle

Note: Solder the LEDs flush to the PCB. The LEDs should not rise above the PCB on their leads.

Note: Solder the LED's anode lead to the pad closest to the letter A. The anode lead of an LED is usually the longest lead and is the one on the rounded side of the LED body. The other lead is the LED's cathode and it's next to the flat side of the LED body.

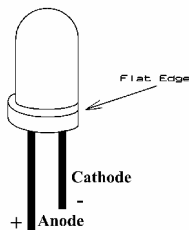


Figure 13. The cathode is on the flat side and the anode is on the rounded side of the LED.

- Solder the yellow LED to the middle circle
- Solder the green LED to the bottom circle
- Cut three wires to about 12 inches long

Note: Use one wire black so the ground wire can be identified.

- Strip ¼ inch of insulation from one end of each wire
- Pass the black wire through the strain relief hole closest to the G printed on the PCB
- Bend the black (ground) wire over and solder to its pad
- Repeat for the other three wires

Note: The finished head will look like figure 12.

- Clean up the cut edges of the 3/8 inch diameter plastic tube
- Cut two slots in the top of the tube

Note: The slot is ¼ inch deep and centered on the plastic tube.

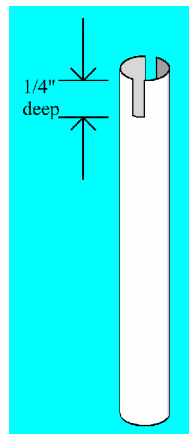


Figure 14. Cut a slot in the plastic tube large enough to slide the traffic light head into.

- Test fit the traffic head PCB; it should fit snugly into the slot

Note: the wires from the traffic light head slide into the hollow tube and eventually exit the tube near the bottom.

- Remove the traffic light head
- Mark a spot one inch from the other end of the tube.

Note: This is where the four wires from the head will exit. The wires will exit the tube on the side opposite of the LEDs. See figure 15.

- Drill a 3/16 inch diameter hole
- Push the traffic head's wire down the tube and out this hole

- Slide the traffic head into place, with the LEDs pointed in the direction opposite the wires' exit.
- Squirt a little hot glue into the top of the tube to hold the traffic light head in place



Figure 15. The completed traffic light head glued into the traffic light tube.

- Mark the center of the three inch square of Syntra plastic
- Drill a 3/8 inch hole through the center

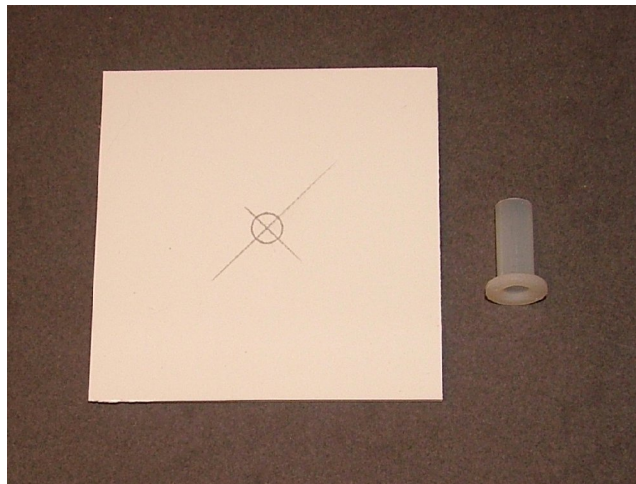


Figure 16. The Syntra plastic and nylon flange bearing of the traffic light base.

- Press the nylon flange bearing through the hole

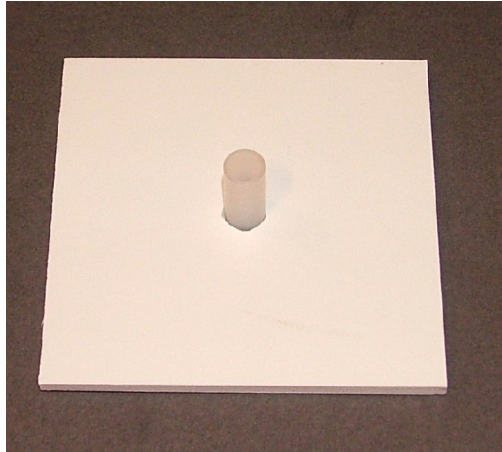


Figure 17. The flange bearing after it is pressed through the Syntra base.

Note: We want a snug fit; you may need to wrap the flange bearing with a strip of tape if the hole in the Syntra is a little too large.

- Slide the traffic light tube over the flange

Note: This must be a snug fit also. You may want to wrap the flange in tape or use a little hot glue.

- Press the four felt pads on the bottom corners of the Syntra base.

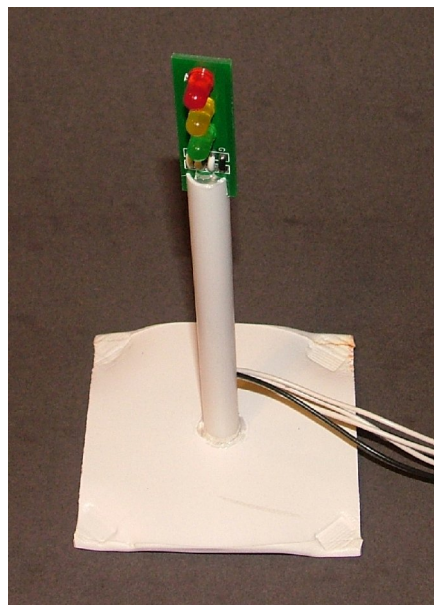


Figure 18. The completed traffic light base. These corners are bent into feet; your traffic light uses felt pads instead.

To simplify the connection to the robot controller, a second PCB is included with the traffic light. The traffic light plug terminates the four traffic light wires, limits current to the LEDs, and connects to three I/O ports.

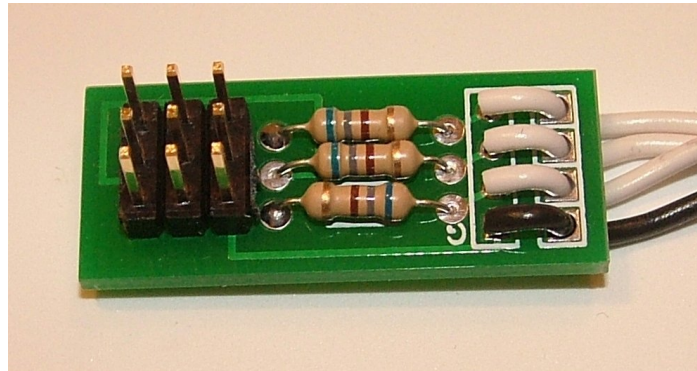


Figure 19. The Traffic Light Plug. This is how the traffic light cable connects to a robot controller.

- Bend the leads of a 680 ohm resistor and solder to the PCB
- Repeat for the other two resistors
- Solder the three by three pin header to the PCB

Note: Solder the short pins to the PCB. The longer pins plug into the robot controller's I/O port.

- Cut the four wires in the traffic light cable to the same length
- Strip ¼ inch of insulation from the end of each wire
- Pass the ground wire through its strain relief hole in the PCB
- Solder the ground wire to its pad

Note: The traffic lights plug's ground pad is mark with the letter G. The ground wire must be soldered to the proper pad of the traffic light will not work.

- Pass the remaining three wires in the cable through their strain relief holes
- Solder each wire to the pad in the PCB

Note: The pads to which the wires are soldered to is not important. The traffic light program will be written to create the proper lighting sequence in the traffic light.

Light Detector

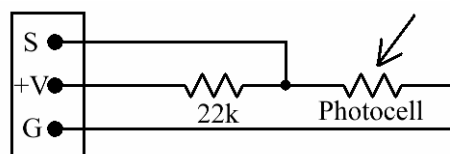


Figure 20. The schematic for the light sensor.

A photocell is a light-sensitive resistor. Soldering it in series to a fixed resistor creates a light-sensitive voltage divider. This means it produces a voltage that changes in response to changing light levels. The robot controller detects this voltage with the **READADC** or **READADC10** command. Alternatively, the robot controller can just check to see if the light intensity is high or low by checking if the voltage from the light detector is high or low.

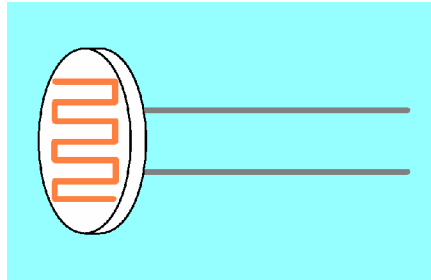


Figure 21. Like a fixed resistor, the photocell has no positive or negative lead. Its resistance decreases as the intensity of light on it increases.

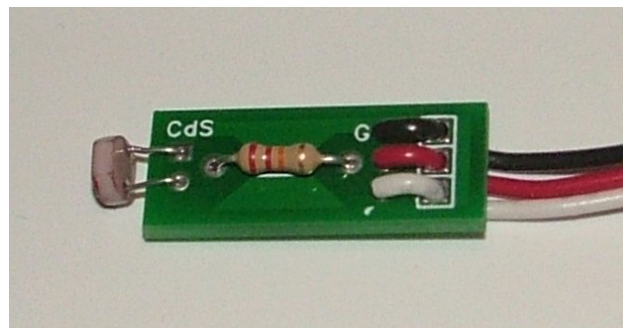


Figure 22. A completed light sensor. Notice the photocell is bent towards the top of the printed circuit board.

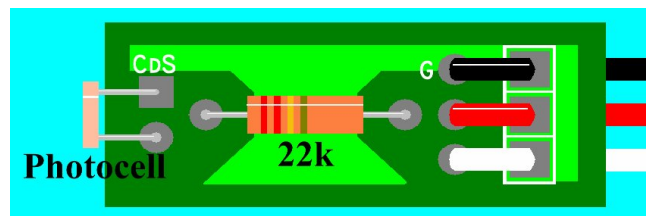


Figure 23. The placement of parts for the light sensor.

- Bend the leads of a 22k-ohm (red, red, orange, gold) resistor and solder it into the PCB.
- Bend the leads of the photocell about 1/8 inches behind the photocell and solder into the PCB.
- Cut three pieces of wire one foot long (use three different colors, with red for +5V and black or green for ground)
- Strip ¼ inches of insulation from one end of all three wires

- Insert the wires into the PCB using the strain relief holes

Note: The ground pad is marked with a G

- Strip $\frac{1}{4}$ inches of insulation from the other ends of all three wires
- Tin the stripped ends
- Insert the long pins of a three pin header into the I/O socket of a robot controller
- Tin the short pins
- Cut three pieces of thin heat shrink $\frac{1}{2}$ inch long each
- Slide the heat shrink over the tinned wires
- Hold the ground against a tinned pin on the edge of the three pin header and heat with a soldering iron

Note: It doesn't matter which pin you solder the ground wire to, as long as it is not the center pin.

- Solder the +5V wire to the center pin
- Solder the signal wire to the remaining pin
- Slide heat shrink over the soldered pins and shrink

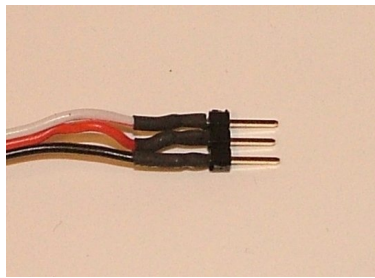


Figure 24. The light sensor cable will look similar to this when completed.

So that the light sensor is less sensitive to the direction it points and able to detect light from all directions, a diffuser is added to the light sensor. The diffuser is a white ping pong ball that the photocell sits inside.

- Draw a circle on the ping pong ball $\frac{1}{2}$ inches in diameter

Note: This is easiest if you can use a circle template. If not, use the $\frac{1}{2}$ inch diameter plastic tube.

Note: While not critical, you may want to draw the hole in the middle of the writing on the ping pong ball (the black lettering).

- Use an Exacto knife and carefully cut out the hole

Note: Cuticle scissors are another good method to cut open a ping pong ball.

- Slide the light sensor PCB into one end of the plastic tube
- Slide the cable out the other end of the tube
- Leave the photocell at the opening of the tube

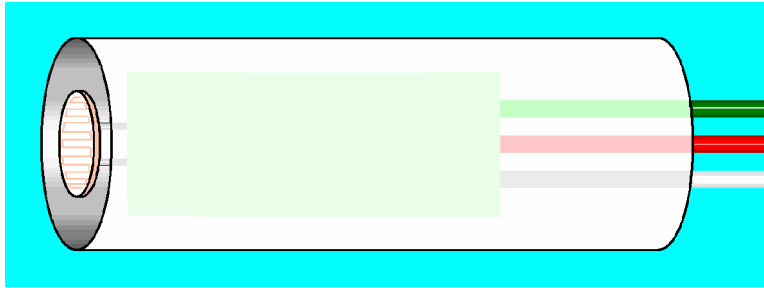


Figure 25. The photocell should be at the opening of the tube. It's better to have the photocell protrude from the plastic tube than it is for the photocell recessed inside.

- Apply a little hot glue around the PCB to hold the light sensor in place
- Insert the photocell end of the plastic tube into the hole cut into the ping pong ball
- Slide the tube into the ping pong ball until it's roughly centered
- Apply hot glue to the plastic tube and ping pong ball to hold the tube in place

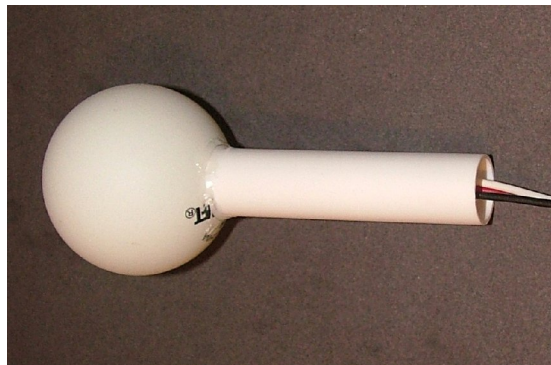


Figure 26. The light sensor with its ping pong ball diffuser.

Temperature Sensor

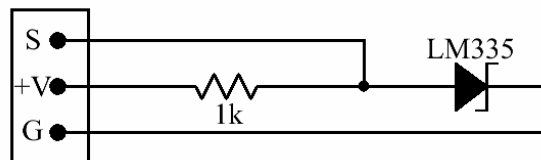


Figure 27. The schematic of the temperature sensor.

The heart of the temperature sensor is the LM335. It looks like a three lead transistor, but is really a temperature sensitive zener diode. Zeners are diodes that are connected into a circuit backwards compared to regular diodes. The zener is designed to breakdown, that

is, to let current flow backwards through them. Normally this destroys diodes, but the zener is designed to let current flow backwards once the reversed voltage on the zener exceeds its limit. This property of a zener makes it useful as a source of constant voltage. The voltage across the zener climbs higher as the voltage gets higher until the breakdown voltage is exceeded. Then the voltage across the zener never gets higher than the zener's breakdown voltage.

What makes the LM335 so special as a zener is that its breakdown voltage depends on its temperature. The breakdown voltage of the zener increases by 0.01 (1/100th) of a volt for every degree Celsius increased in its temperature. The breakdown voltage and hence the LM335's temperature is measured with the **READADC** or **READADC10** command.

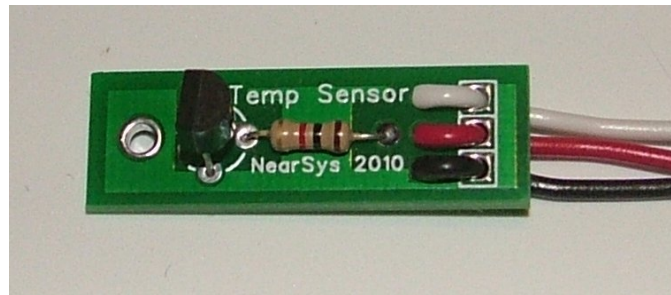


Figure 28. The completed temperature sensor.

- Bend the leads of a 1k-ohm (brown, black, red, gold) resistor and solder it into the PCB.
- Solder the LM335 (looks like a transistor) to the PCB, observing the drawing in the top silk to align the flat face of the LM335..
- Cut three pieces of wire one foot long (use three different colors, with red for +5V and black or green for ground)
- Strip ¼ inches of insulation from one end of all three wires
- Insert the wires into the PCB using the strain relief holes

Note: The ground pad is the closest to NearSys 2010

- Strip ¼ inches of insulation from the other ends of all three wires
- Tin the stripped ends
- Insert the long pins of a three pin header into the I/O socket of a robot controller
- Tin the short pins
- Cut three pieces of thin heat shrink ½ inch long each
- Slide the heat shrink over the tinned wires
- Hold the ground against a tinned pin on one of the end pins

Note: It doesn't matter which pin you solder the ground wire to, as long as it is not the center pin.

- Solder the +5V wire to the center pin

- Solder the signal wire to the remaining pin
- Slide heat shrink over the soldered pins and shrink

Piezo Speaker

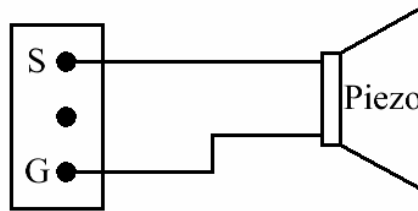


Figure 29. Schematic for the Piezo speaker.

The piezo is a simple speaker that doesn't use a magnet to move a cone. Instead, it uses a voltage to cause a thinly cut crystal change its shape, or deform. If the voltage changes, or even switches off and on, the crystal changes between two slightly different shapes. The movement of the crystal acts as a speaker and the piezo produces a tone to match the changing voltage applied to it. The piezo element is therefore a durable, simple speaker.



Figure 30. The completed piezo speaker.

- Remove the paper sticker covering the top of the piezo speaker.
- Look on the side of the piezo speaker to identify the positive lead.

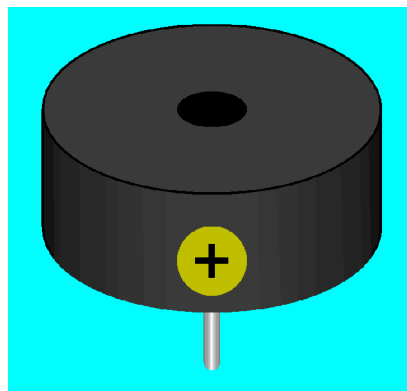


Figure 31. The positive lead of the piezo is the pin closest to the positive mark.

- Solder the piezo speaker to the PCB, making sure the positive lead is in the pad marked By the plus (+)
- Cut two pieces of wire one foot long (use two different colors with black or green for ground)
- Strip ¼ inches of insulation from one end of both wires
- Insert the wires into the PCB using the strain relief holes

Note: The ground pad is the square pad closest to the capital letter P

- Strip ¼ inches of insulation from the other ends of both wires
- Tin the stripped ends
- Insert the long pins of a three pin header into the I/O socket of a robot controller
- Tin the short pins
- Cut two pieces of thin heat shrink ½ inch long each
- Slide the heat shrink over the tinned wires
- Hold the ground against a tinned pin on one of the end pins

Note: It doesn't matter which pin you solder the ground wire to, as long as it is not the center pin.

- Solder the signal wire to pin on the opposite side of the header

Note: No wire is soldered to the center pin of the piezo speaker.

- Slide heat shrink over the soldered pins and shrink

Using the Programming Kit

The Programming Cable

The black (or green) wire indicates which side of the receptacle plugs into the ground pin of the microcontroller programming header.

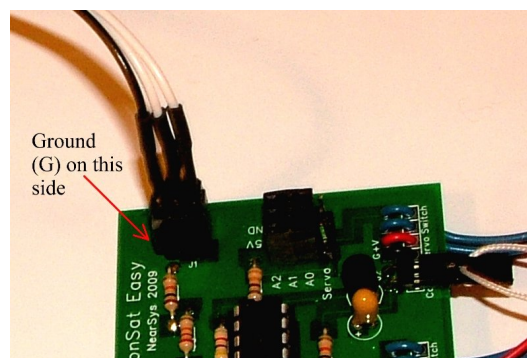


Figure 32. A programming cable connected to a BalloonSat Easy. Look for the G next to the programming pin's ground connection

The other end of the cable plugs into a PC's comm port or USB to serial adapter. After writing code, click on the blue arrow on the editor's menu (the RUN button).

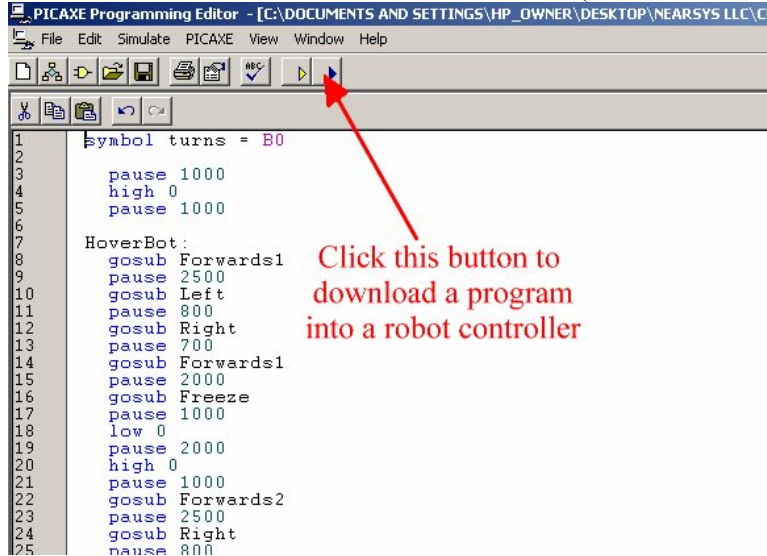


Figure 33. The RUN button in the PICAXE Editor.

The Traffic Light

The traffic light is designed to plug into Input numbers 0, 1, and 2. This may not make sense, since these are inputs and it requires an output to turn on an LED connected to ground. However, the PICAXE-14M, the microcontroller in the CheapBot-14, can be reconfigured to make three of the input pins on PortC into output pins. The command to HIGH a PortC pin does two things. First, it converts the pin to an output and second it connects the pin to the +5 volt supply inside the PICAXE-14M.

The command to high the pin is

HIGH PORTC x

The command to low the pin is

LOW PORTC x

Where X can be either 3, 4, or 5

This table is the PORTC names for the three inputs you can modify on the CheapBot-14

INPUT 0 is PORTC 3
INPUT 1 is PORTC 4
INPUT 2 is PORTC 5

Whether you use a HIGH command on a “normal” output pin or use PORTC, the pause commands work the same.

The rest of the code is pretty simple. It's just an endless loop where one LED at a time is turned on while the other LEDs remain off. Each LED takes turns shining before it is switched off and another LED is turned on. At the end of the code, send the program back to the beginning of the program with a GOTO and a label. Your code will operate like this example below.

1. HIGH one of the PORTC pins to turn on the colored LED in the traffic light
2. LOW the remaining PORTC pins to turn off the other colors.
3. PAUSE the code for a while to leave the LED turned on.
4. LOW the LED turned on to shut it off
5. HIGH one of the other PORTC pins to turn on a different colored LED
6. PAUSE the code for a while to leave the LED turned on.
7. LOW the LED turned on to shut it off
8. HIGH one of the other PORTC pins to turn on a different colored LED
9. PAUSE the code for a while to leave the LED turned on.
10. LOW the LED turned on to shut it off
11. GOTO the beginning of the program to repeat

The Light Sensor

The light sensor attaches to a pillar on the CheapBot with four nylon wire clamps and two nylon bolts and nuts. Use the ½ inch nylon wire clamp to attach to the light sensor post and the ¼ inch nylon wire clamp to attach to the CheapBot pillar. The nylon nut and bolt attach the wire clamps together as illustrated below.



Figure 34. Two nylon wire clamps attach to the pillar of the light sensor. They connect to each other, back to back, with a nylon nut and bolt.

Wrap two ½ inch nylon wire clamps on the pillar of the light sensor and wrap two ¼ inch nylon wire clamps around a pillar of a CheapBot riser. The nylon wire clamps are loose at this point, so they can slide up and down the pillars. Align the wire clamps with each other and bolt them together with nylon nuts and bolts. After tightening the nylon nuts, the nylon wire clamps will be tight enough that they will resist sliding up and down any further.

Plug the light sensor into an Input of the robot controller and use the following command.

READADC 1, B0

Where 1 is the Input port on the robot controller (you may plug it into a different port) and B0 is a one byte variable to store the result from the **READADC** command (you can store the results into any used one byte variable).

ADC is short for analog-to-digital conversion. The **READADC** command takes the analog voltage of the light sensor and converts it into a number between 0 and 255. The analog voltage of the light sensor increases in bright light and decreases in dimmer light. By comparing the value of the conversion, the robot controller can determine if the light level is bright, dim, increasing, or decreasing. It's probably best if you first determine the values you will get when your robot is in bright light and in dim or no light. Get the values with the **DEBUG** command and place your robot in bright and dark locations. The code looks like this.

```
Light:  
READADC 1,B0  
DEBUG  
GOTO Light
```

Leave the robot connected to the PC while running this program and the Debug Window will display the value from the light sensor in byte B0. Once you know typical values, you can write comparison tests into your robotic program. For example,

```
Robot:  
READADC 1,B0  
IF B0 > 158 THEN Bright  
IF B0 < 75 THEN Dark
```

In_Between:

Bright:

Dark:

You can start your robot code by looking at the light level at the beginning, storing the value in memory, and then later comparing the new light levels to the initial value to determine changes.

Using the Temperature Sensor

The temperature sensor produces an analog voltage that is related to its temperature. The temperature scale the LM335 uses is the Kelvin scale. Degrees Kelvin are the same units as degrees Celsius, except that zero begins at absolute zero (-459.4⁰ Fahrenheit). Use the **READADC** command to convert the LM335's analog voltage into a digital value the robot controller can use.

Convert the digital reading into degrees Kelvin by multiplying it by two (actually, this introduces a small error, but it can be ignored for what we need the robot to do with the temperature). If you subtract 273 from the temperature in Kelvins, the temperature converts to Celsius. What about converting the temperature in Fahrenheit? For this, use the Rankin scale, the Fahrenheit equivalent to Kelvins. In the Fahrenheit scale, a value from the **READADC** command of 130 is 0° F. Every 1 increase in the temperature reading is equal to an increase of 3.5° F. So use the following code to convert the temperature reading into degrees Fahrenheit.

```
READADC 1,B0  
B0 = B0 - 130  
W0 = B0 * 35  
B0 = W0 / 10
```

Now this code required the use of a word-sized variable (W0). That's because when B0 is multiplied by 35, the result will be greater than 255, the largest variable that can be stored inside a byte-sized variable (B0). The resulting temperature can be stored inside a byte-sized variable, so the result is stored back in B0. For this code snippet to work properly, W0 cannot be used elsewhere in the program (or else the value stored in W0 will be changed unpredictably). Note that W0 consists of B0 and B1. If either of these variables store important data, then use a different variable for the conversion.

Using the Piezo Speaker

The command for the piezo speaker to produce a tone is **SOUND**. The piezo speaker must be plugged into an Output on the robot controller. Use the **SOUND** command as illustrated below.

```
SOUND 1,(185,30)
```

This code snippet will produce a tone if the piezo is plugged into Output channel 1. The tone is #185 and lasts for 300 milliseconds. Valid tones are values from 1 to 127. The duration of the tone is the second number in parenthesis multiplied by 10 milliseconds. The higher the number of the tone, the higher its pitch. Numbers above 127 (128 to 255) are white noise and not very interesting.

1 October 2010