

## Experiment: Calbot (1)

Go to the Calbot page

<http://www-inst.eecs.berkeley.edu/~ee40/calbot/webpage/index.htm>

download all the files in [CalBOT project files](#) to a directory named “Calbot” in your home network directory on the H: drive.

Once the files have been downloaded to your directory open up Keil  $\mu$ Vision2

Open the calbot project file by clicking on **Project** ► **Open Project** and selecting **Calbot.uv2**. Look over the code and make sure you understand it. Everything is well defined in the manual.

### Downloading the Program to the KitCon Board

You need to plug the ac adaptor and serial cable into the board (make sure the battery is plugged in as well). There should be a serial cable already at all the stations in the lab so you don't need to use your own. After the board is powered, you should see a red status LED. If that is not on, something is wrong. If the green program LED is on push the reset button next to it. Make the executable file by going to **Project** ► **Rebuild all target files**. Download the executable by going to **Debug** ► **Start/Stop Debug Session**. You can monitor the status of the download by the “percentage completed” bar at the bottom of the window. If you get an error message indicating a problem making a Connection to Target System make sure your serial cable, ac adaptor, and battery are all plugged in, hit the reset push button on the board and hit the Try Again button in Keil. After the code is downloaded properly, run the program from **Debug** ► **Go**. This should turn on all the LEDs.

### Adding LED Functionality

Now edit the code to have the board display your K.I.T.T. code from prelab. You should add your code into the main function inside the while loop.

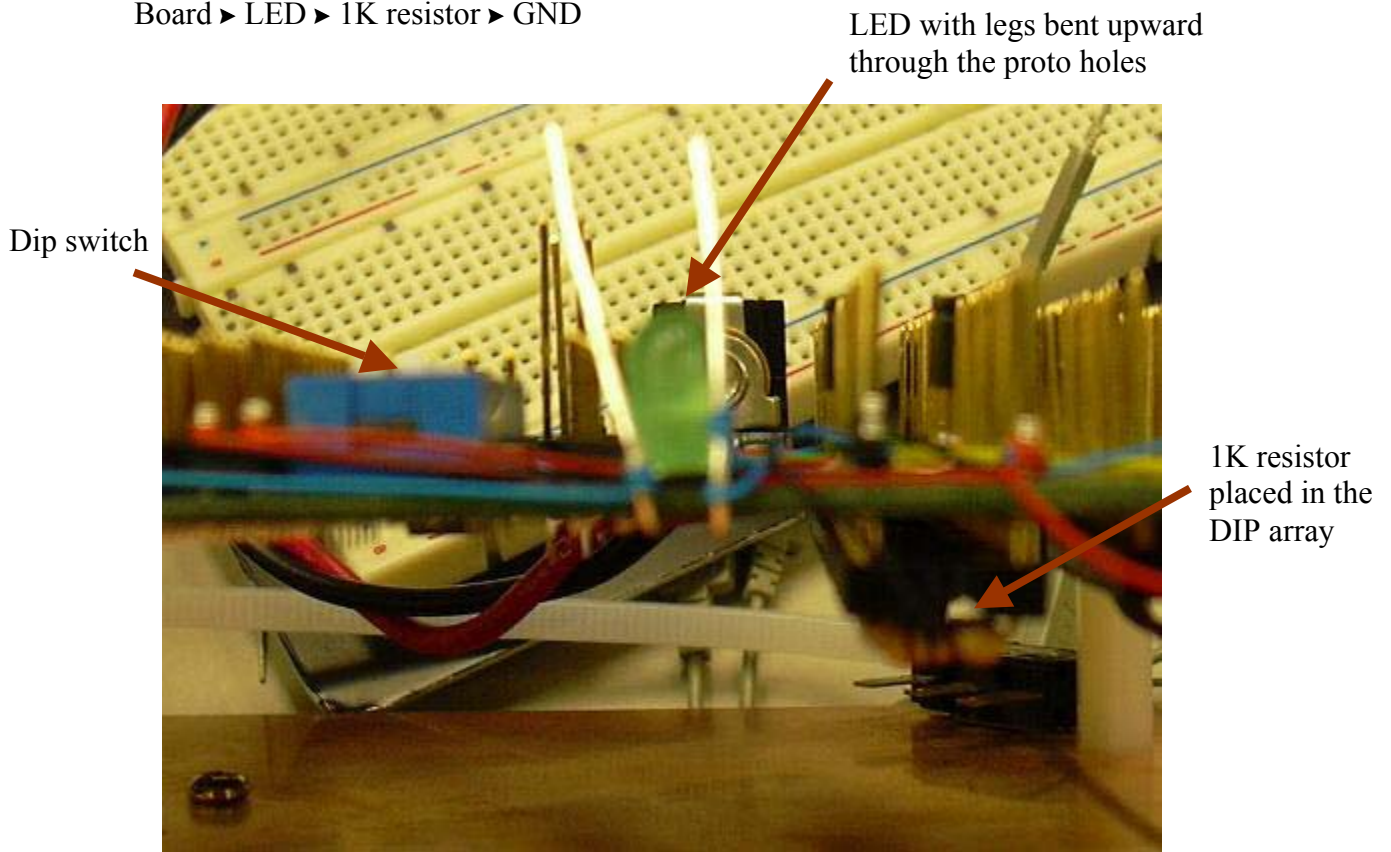
```
main(void) {
    Initialize()
    while(1) {
        /*****
        your code here
        *****/
    }
}
```

Test to make sure your code works the way you expect. Did you turn on only one LED at a time? Fix any errors. You may need to review hexadecimal.

## External LED

Now you will wire up an external LED and control it using general I/O.

Board ► LED ► 1K resistor ► GND



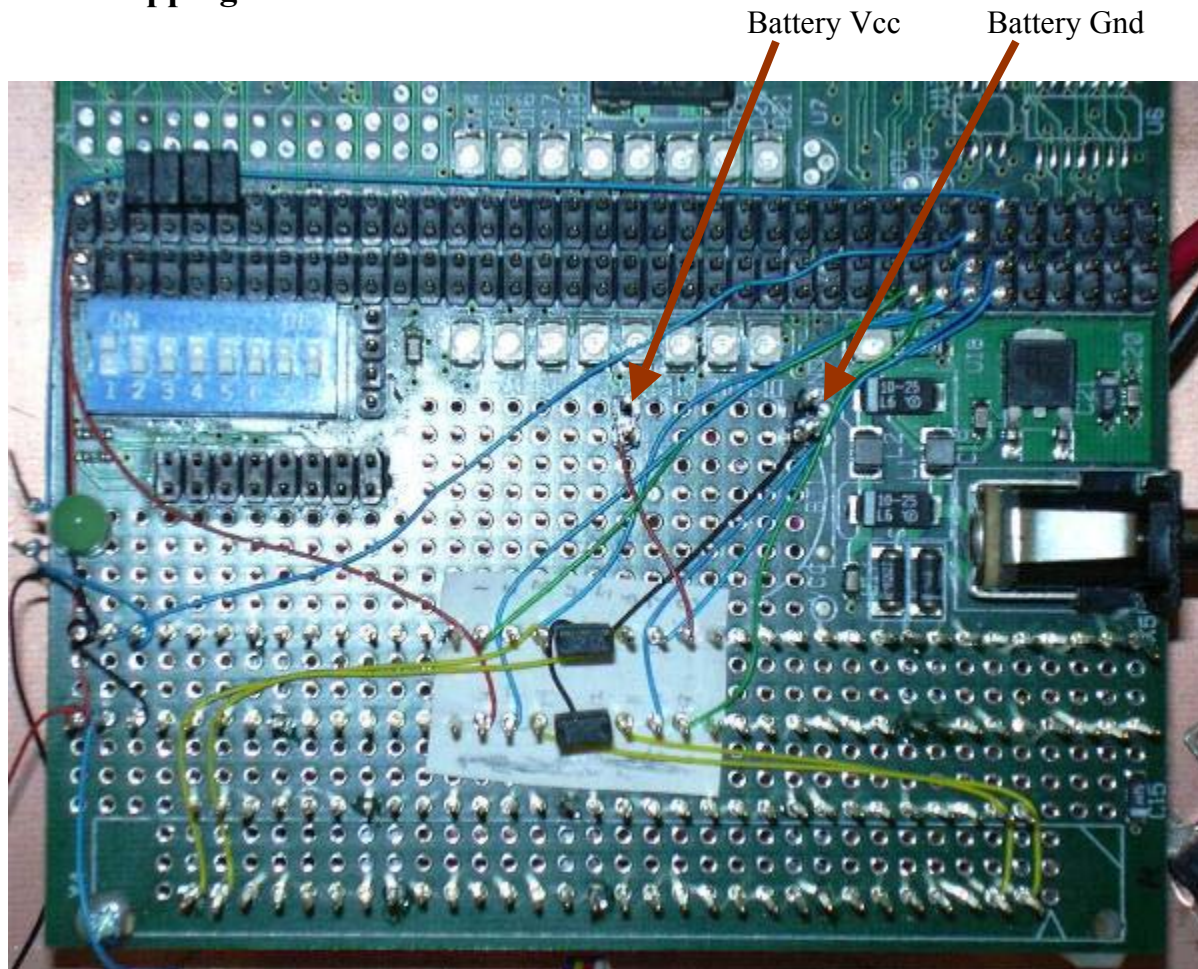
**Figure 1.** Side view of the KitCon board

Place the LED through the proto holes in the board by the dip switch, make sure to note which side is the anode (+) and the cathode (-). Put a 1K resistor into the DIP array on the bottom of the board, you will need to cut the legs of the resistor to fit into the DIP socket. Wire one end of the resistor to ground and the other end to the cathode of the LED. Then wire P8.7 to the anode of the LED. Add the code to turn on the LED to your project and verify its operation.

```
DP8 = 0xff;           //set all port 8 to output
P8=0x80;              //set P8.7 high
```

You should be able to change P8.7 and control if the LED is on or off.

## Wire Wrapping the Motor Driver



**Figure 2.** Top view of the KitCon board with wire wraps

Now that you are familiar with the board, you can start setting up the motor driver. Wire wrapping the motor driver is probably the most difficult part of the project. If you aren't organized and careful, you can end up with a real mess in wiring, which will be more work trying to debug than actually taking the time to make it nice originally.

Open up the PWM chapter of the manual ([Controlling the motor - programming the Pulse Width Modulation unit](#)) and find the pinout of the motor driver on page 14. Using the KitCon pinout wire up the PWM inputs, P7.0 and P7.1, and the control inputs, P8.0-P8.3. Battery GND and Battery Vcc are two sets of pins sticking up from the board. Battery Vcc is different from board Vcc. If you wire those wrong, you could burn out your motor driver – so be very careful!

### Some tips on wire wrapping

It's a good idea to put some paper down over the pins and number the pins the pins of the motor driver on the paper. Remember that you are putting your motor driver into the DIP socket on the backside of the board; thus on the top of the board, the pins are the mirror image of the pins from the datasheet. When you design any circuit for the board, remember to design in with the normal

view of the circuit and a mirror image for when it's underneath the board. Also, it is also a good idea to use different color wires for the different signals.

Wire color	Signal
Red	Vcc
Black	Ground
Green	PWM
Blue	I/O – control input

Hopefully you've gotten this far during the lab time. If you're really adventurous you can read the whole PWM chapter and adjust the code to see if you can send all the appropriate signals to the motor driver. Please don't hook up the motor until a TA has checked your wiring and made sure you're PWM signal is correct.