Beacon Board: Assembly Plans

Before you begin, take a moment to look over the *Bill of Materials* for our beacon boards. Verify that all of the parts mentioned on the Bill of Materials are present in your kit.

Part 1: PC Interface & Controller

First, you'll assemble the *microcontroller* and *interfacing logic*.

1. Populate JP1, the USB header. For this part, you'll need to ensure you solder the five pads which are slightly occluded by the USB header, as well as the four "structural" pads on the header itself. It's better to put too little solder on the five small pads rather than too much: you can always add more later. If you need help, you can ask for assistance from the undergraduate course assistants.
2. Populate IC1, the AVR microcontroller. Be sure to place the *dot* on the microcontroller in the same direction as the dot on the silkscreen.
3. Populate the resistors and capacitors that support the microcontroller:

   (a) Populate R2 and R3, located next to the USB port.
   (b) Populate C2 and C5, the two capacitors on either side of the USB connectors.
   (c) Populate C4, to the left side of the microcontroller.
   (d) Populate R1 and R4, to the top of the microcontroller.
   (e) Populate C3 and C1, to the right of the microcontroller—above and below the slot for the system clock.

4. Populate Y1, the 16MHz crystal oscillator.

After your microcontroller and interfacing logic have been assembled, carefully inspect your each of your components; ensure that there are no bridges or shorts. Several magnifying lenses are available in the student shop, which may help with your analysis.

Once you're sure that your components are correctly soldered, return to the lab and attach your beacon board via USB. It should now show up as a USB device on your computer:

Testing on Linux

If you're connecting your board to one of the lab computers (recommended), or another Linux computer, you can check to see if the device is recognized by running the following terminal command:

```
lsusb
```

If your board is working correctly, the list will feature a device described as *Atmel Corporation:*

```
03eb:2ffB Atmel Corporation
```

This is your microcontroller! If this text appears, you're ready to move on to the next part.

Testing on Windows

When you first attach the board, Windows will announce that a new *Atmel DFU* device has been connected. If this pop-up appears, your microcontroller is correctly soldered— and you're ready to move on to the next part!

If not, open the Device Manager:

- On *Windows XP,* select "Run" from the start menu, and enter the command "`devicemgr`". Press enter to launch the device manager.

- On *Windows Vista/7/8,* type "Device Manager" into the start menu.

In the device manager, look for a category labeled *Atmel DFU.* If this category is present, your microcontroller has been successfully soldered!

Troubleshooting

If your device isn't working, you'll want to thoroughly check your soldered board against the schematic. A basic debugging process might go as follows:

- **Verify that your device is correctly receiving power.**
  A good initial rule for debugging a board: verify your known voltages! In this case, a good first step is to find a microcontroller pin that should be connected to VCC, and verify that it is correctly receiving the USB power supply of +5V, with respect to a GND pin. As you verify voltages, it may help to check off the nets printed on your schematics.
- **Verify that all of the connections that *should* be present are actually implemented.**
  Using the continuity mode of your multimeter, check to make sure that all of the appropriate connections exist. As an example, you might place one lead on the top of R3, and verify that it's making a good connection to D- pin on your microcontroller.
- **Check your design for "obvious" shorts.**
  Using the continuity mode of your multimeter, check to make sure there *isn't* continuity between VCC and ground.
- **Verify known resistance values.**
  Using the ohm mode of your multimeter, verify that signal paths have the appropriate resistance values. As an example, you might check the resistance between the bottom of R2, and the D+ pin on your microcontroller; verifying that this value is close to 22 ohms.
- **Verify that your clock is functioning.**
  Using the oscilloscope, probe the voltage *across* your 16MHz clock crystal. Verify that the signal that appears has a frequency of 16MHz— though note that its precise shape doesn't matter.

If you've verified the above and still need help, ask for assistance from the TAs, UCAs, or instructor.

Part 2: The Beacon LEDs

Next, you'll implement the Beacon LEDs, and their supporting hardware:

1. Populate R13-R16, the current limiting resistors for the white LEDs.
2. Populate R18-R21, the current limiting resistors for the green LEDs.
3. Populate R23-R26, the current limiting resistors for the red LEDs.
4. Populate R12, R17, and R22, the input resistors for the LED drive transistors.
5. Populate Q3-Q5, the LED drive transistors.
6. Populate the white LEDs, as described below— you can discern the white LEDs by their shorter legs; if you look into the top of these LED's lens, you should notice a yellow substrate. Be sure to observe the polarity of these LEDs: the flat side of the LED goes towards the flat side of the silkscreen.

   (a) Populate the LED next to R15 normally, with the LED facing straight up from the board. Note that the student LEDs have smaller legs than the competition LEDs, and thus fit loosely into the through-holes. As you fill this hole with solder, the LEDs will become held tightly in the hole.
   (b) Populate the LED next to R14 on the reverse side of the board: the LED's head should face down, towards the table. Be sure you're still observing correct key polarity.
   (c) Populate the LED next to R13 by bending its leads by 90 degrees, so the LED faces the bottom edge of the board.

(d) Populate the LED next to R16 by bending its leads by 90 degrees in the opposite direction, so the LED faces the top edge of the board.
(e) Once the LEDs are fully soldered, trim their leads using the provided flush cutters.

7. Populate the four *red* LEDs in the same way. You can discern the red LEDs by looking directly into the lens of the LEDs: their substrate is surrounded by a red ring.
8. Populate the four *green* LEDs in the same way. You should have eight LEDs remaining; the green LEDs will be the four larger ones.

Part 3: The IR Communications hardware.

Finally, you'll implement the IR communications hardware:

1. Populate R5, R6, R28, and R29; the base transistors for the IR hardware.
2. Populate R7-R10, the current limiting resistors for the IR transmitters.
3. Populate R27, the pull-up resistor for the IR receiver NAND gate.
4. Populate the four remaining LEDs in the same way as you did the four beacon LEDs. You should be done!

Once you've finished the instructions above, your device is *assembled*— and ready for testing and characterization during our third week.