Week 10 Lab Report

11/04/2019

Shubham Bhosale | Chinmay Patil || Gully Gang Boys

Section 01: Layout Design

Drawbacks in our board:

1. *No ground plane on bottom*

    i) After showing our fabricated board to Prof. Bogatin, we understood the real meaning of continuous return plane. Initially we had a misconception that grounding a set of pins altogether in a particular section, meant a continuous return plane.

    ii) Obviously, what we did was not a good layout practice as each signal path did not have a short return path.

    iii) We know that the next design is going to be a 4-Layer board, hence before starting our layout, we would be setting a rule in Altium that one layer would always be a continuous return plane.

2. *Long gaps in the ground plane without ground straps over gaps*

    i) This was an unforced consequence of the error mentioned above.

    ii) It was not a good layout practice either.

    iii) In our next design, as we will be having a continuous return plane, we will keep in mind to have short traces going through that layer and avoid long gaps in the ground plane.

Extra Credit:

1. We added a second row of return pins on the outside of standard header sockets, to make our testing process simpler.

Grades earned for this section:

- (-2) for design errors.
- (+1) for justification of errors.
- (+1) for extra credit.
- **TOTAL: (10/10)**

Section 02: Functioning of Board

We got our board to blink the ‘sck’ LED!

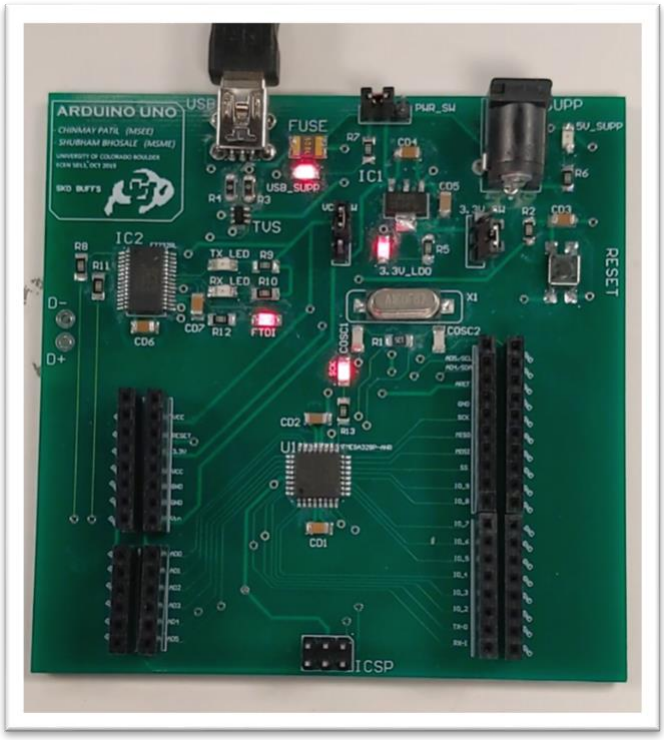


Figure 1: LED in its ON-state

Figure 2: LED in its OFF-state.

Section 03: Final Report

1. *The POR:*

- Understanding the circuit step by step made it easier for us to understand this complex Arduino schematic, as we had no prior experience in either designing or working on an Arduino UNO board.
- We estimated the total budget of the components to be $15. Along with shipping the cost of the whole board was estimated to be around $30.
- We made sure we had ESD protection through TSV Diode and a resettable fuse in our circuit to protect the USB port of our laptop.

2. *The BOM:*

- Ordering components in a set of either 10 or 20, is sometimes cheaper than buying components in non-regular quantities (e.g. 22uF Capacitors)
- We had prior successful experience in using a 3.3V LDO on our previous board. Hence, we used the same part in this design as well.
- Similarly, for the 22uF capacitor, 1K resistors and LEDs we used the ones we had experience working with before.

3. *The Schematic:*

- As we had a well scoped out POR, it made easier to create the schematic as we knew the board functionality pretty well and we also didn't miss out on any passive components. Also referring the data sheets for the ICs made it easier to map out the pin configuration on the schematic.
- As Arduino is an open source project, there are number of schematic available online for reference. We referred to these online schematics, making sure we don't blindly copy the schematic whereas we understood each and every component and connections and only then use it in our design.
- We also did peer reviews of our schematic with other teams to recheck our connection.

4. *The Layout:*

- The component on the PCB were well spaced out (within the std. board size), making it easier to route the traces in between the components. Similarly, soldering was also simplified as there was minimum interference of the components with the soldering iron.
- We followed the standard pin out spacing from the commercial Arduino boards. So, any commercial shield can be readily installed on our board.
- A well-planned project schedule during the POR phase, made it possible to order the PCB well in time.
- We provided polarity dots for correctly orienting the LEDs and the ICs without the need to refer datasheets while assembly.

- In our next design, we will be keeping a continuous return plane with shorter gaps and return paths.

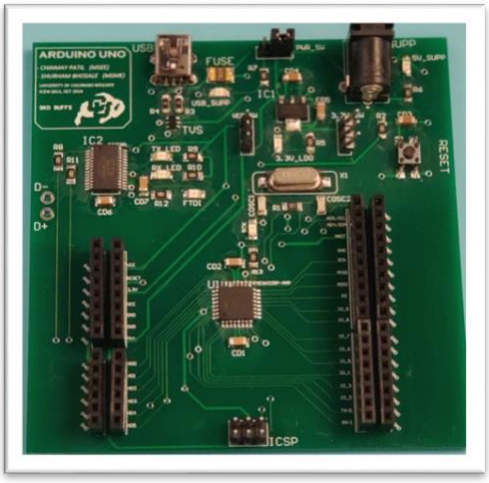
5. *Assembly:*

- As there were polarity indicators on the silkscreen of the board, we did not spend unnecessary time figuring out the orientation of ICs and LEDs.
- Before starting our assembly, we verified that there were no shorts between power-ground rails in the bare board.
- We checked for the footprints to verify that we had got all the right parts with no manufacturing defect.
- For the whole board, we soldered the small SMT components first and then the big through hole components. This allowed us to make easy movement of solder iron around the board.

6. *Bring up and Final Test:*

- Having the isolation switches on the board, made it easier to isolate different parts of the board and test them separately.
- Also, having a set of return pins next to each and every signal pin, made probing of signals much easier and avoided unnecessary crosstalk between probes.
- Setting the oscilloscope trigger mode to auto, sometimes makes the signal shaky and difficult to read through. Instead using the single shot triggering on oscilloscope, makes the signal clear and distinct for easy reading.
- Also, the probe needs to be compensated correctly each and every time, before making any of the readings on oscilloscope.
- While testing in future, first step would always be to compensate the probe well, check in the oscilloscope if the appropriate probe has been detected (1x or 10x) as it causes ambiguity in the reading. Wherever necessary the single shot triggering mode of oscilloscope will be used. And, the isolation switches will be added to the circuit while bringing up the board for the ease of debugging.

7. *Pictures of our Arduino UNO:*



8. *Comparison between Targeted Board and Commercial Board:*

i) *Without 50Ω Resistor:*

(a) *Quite Low:*

Figure 03: Quite Low on Commercial Arduino

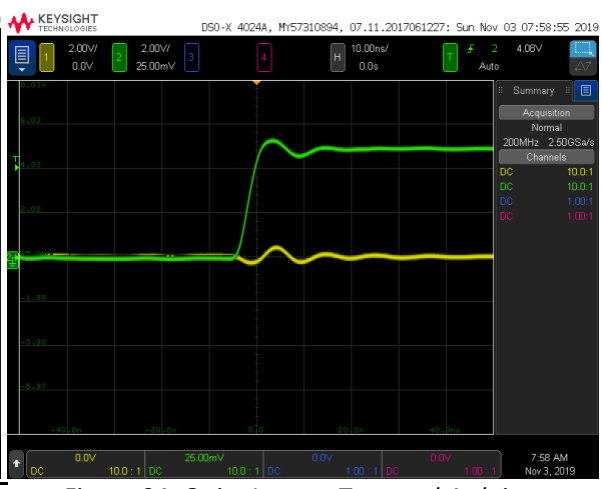


Figure 04: Quite Low on Targeted Arduino

- We can see that the commercial Arduino board has a little noise in its waveforms while switching.
- Our target board also sees considerable amount of noise at switching because of the lack of continuous return plane on the bottom layer.
- But if we had a continuous return plane, we wouldn't have seen any noise on the targeted board at switching instance.

(NOTE: For all other conditions, refer appendix A)

9. *Extra Credit:*

- To do something special with our newly created board, we tried the 'Fade' feature from the Arduino IDE sketch examples.
- We connected a Green LED to pin number 9 and grounded the LED through a 220Ω resistor.
- The initial brightness of the LED in the code was set to 0 and the incremental fade amount was set to +5. Once the brightness reached a level of 255, the fade amount is changed to -5, thereby decrementing the brightness of LED from 255 to 0.
- Also, after each increment a delay of 30 msec is provided, for the fading effect to be noticeable.

Click here to see the fading LED in action

- Code:

```
int led = 9;           // the PWM pin the LED is attached to
int brightness = 0;    // how bright the LED is
int fadeAmount = 5;    // how many points to fade the LED by

// the setup routine runs once when you press reset:
void setup() {
  // declare pin 9 to be an output:
  pinMode(led, OUTPUT);
}

// the loop routine runs over and over again forever:
void loop() {
  // set the brightness of pin 9:
  analogWrite(led, brightness);

  // change the brightness for next time through the loop:
  brightness = brightness + fadeAmount;

  // reverse the direction of the fading at the ends of the fade:
  if (brightness <= 0 || brightness >= 255) {
    fadeAmount = -fadeAmount;
  }
  // wait for 30 milliseconds to see the dimming effect
  delay(30);
}
```

APPENDIX A:

1. One I/O pin switching:

a. Quite Low (Without 50Ω resistor)

Figure 03: Quite Low on Commercial Arduino

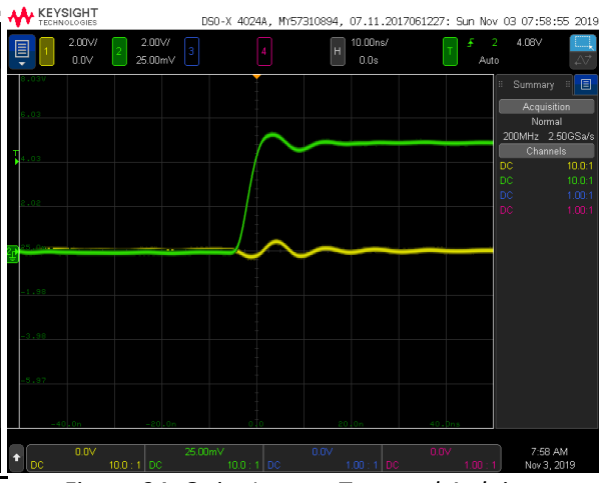


Figure 04: Quite Low on Targeted Arduino

b. Quite High (Without 50Ω resistor)

Figure 05: Quite High on Commercial Arduino

Figure 06: Quite High on Targeted Arduino

c. Quite Low (With 50Ω resistor)

Figure 07: Quite Low on Commercial Arduino

Figure 08: Quite Low on Targeted Arduino

d. Quite High (With 50Ω resistor)

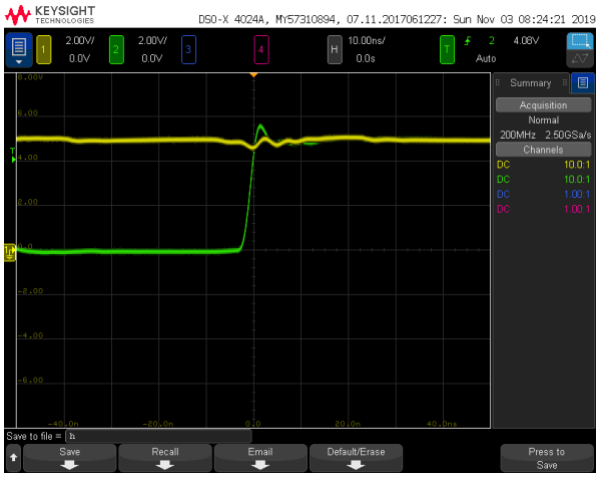


Figure 09: Quite High on Commercial Arduino

Figure 10: Quite High on Targeted Arduino

2. Two I/O pins switching:

a. Quite Low (Without 50Ω resistor)

Figure 11: Quite Low on Commercial Arduino

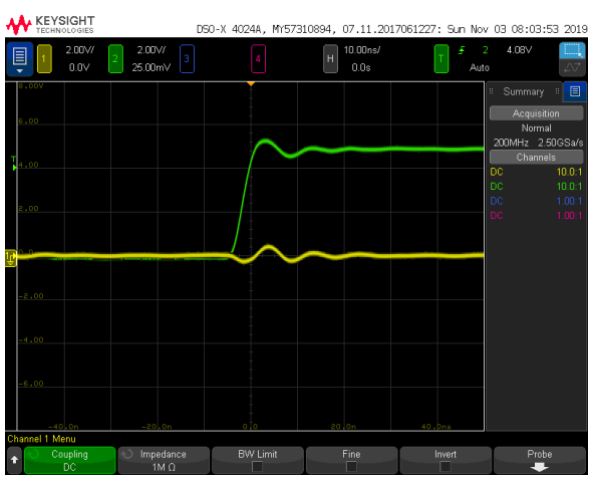


Figure 12: Quite Low on Targeted Arduino

b. Quite High (Without 50Ω resistor)

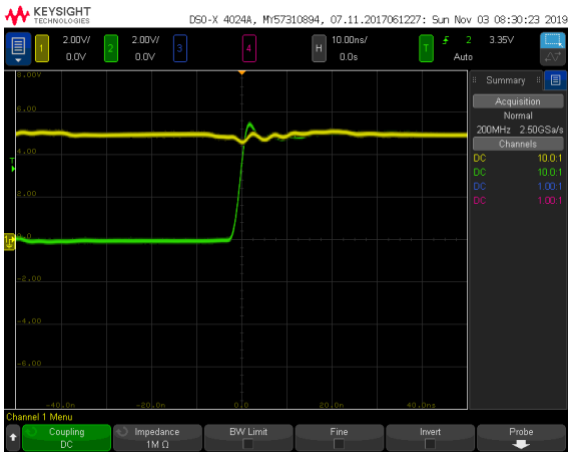


Figure 13: Quite High on Commercial Arduino

Figure 14: Quite High on Targeted Arduino

c. Quite Low (With 50Ω resistor)

Figure 15: Quite Low on Commercial Arduino

Figure 16: Quite Low on Targeted Arduino

d. Quite High (With 50Ω resistor)

Figure 17: Quite High on Commercial Arduino

Figure 18: Quite High on Targeted Arduino

3. Three I/O pins switching:

a. Quite Low (Without 50Ω resistor)

Figure 19: Quite Low on Commercial Arduino

Figure 20: Quite Low on Targeted Arduino

b. Quite High (Without 50Ω resistor)

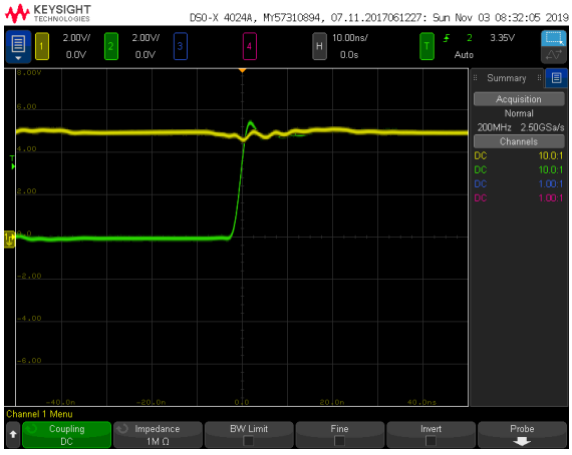


Figure 21: Quite High on Commercial Arduino

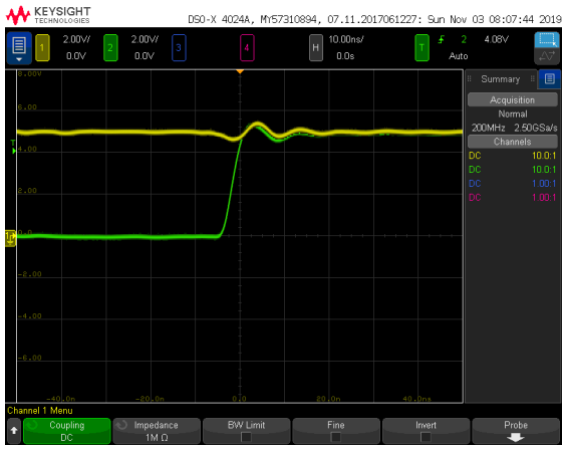


Figure 22: Quite High on Targeted Arduino

c. Quite Low (With 50Ω resistor)

Figure 23: Quite Low on Commercial Arduino

Figure 24: Quite Low on Targeted Arduino

d. Quite High (With 50Ω resistor)

Figure 25: Quite High on Commercial Arduino

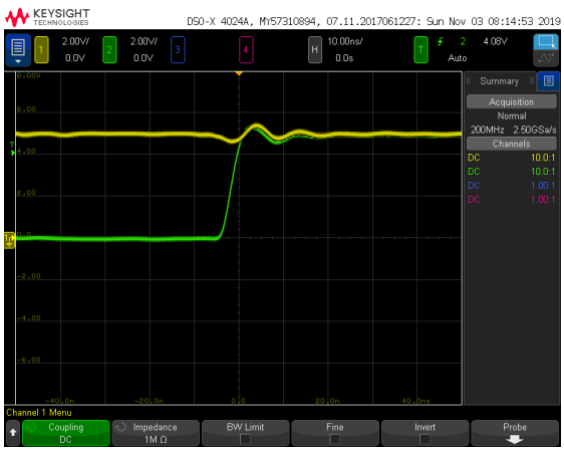


Figure 26: Quite High on Targeted Arduino