ECEN 3730-
PCB Design,
Connor Sorrell

PCB Design Board 3 Report: Golden Arduino

Introduction

This board focused on building a custom “Golden Arduino”. This is a fully functional “Arduino”, Atmega328P-based microcontroller board, which was built from the ground up with performance, signal integrity, and noise mitigation in mind

While this board challenged me far more than Board 1 or 2, I walked away with a much deeper sense of engineering judgment. The real learning happened in the failures. This report documents both what worked and also the hard earned lessons

Design Objectives

- Using the ATmega328P microcontroller, make a working Arduino.
- Add better power delivery and decoupling for stable operation
- Implement USB-to-serial communication using the CH340G
- Provide reset switch (with debounce circuit)
- Integrate test points/LEDs for debugging and bring-up
- Burn the Arduino bootloader and validate USB upload capability
- Capture and analyze inrush current and switching noise
- Compare noise performance against a commercial Arduino board
- Include NUMEROUS test points for easy debugging/verification
- Sense resistor for measurement of inrush current

Bill of Materials (BOM)

- ATmega328P (TQFP package)
- CH340G USB-to-Serial chip
- 12 MHz Crystal & 16 MHz Crystal
- TVS diode

- 1206 SMT Resistors and Capacitors (various)
- 1206 Inductor
- LEDs, pushbuttons,
- USB mini connector
- Decoupling capacitors: 100nF local caps + bulk 10µF
- Test points (labeled throughout schematic)

1. Schematic Design

The schematic began with a rough block diagram laying out the main functional blocks: power, USB interface, crystal circuit, microcontroller core, and test/debugging. In Altium, I then carefully implemented each block. Every power pin had a corresponding decoupling cap placed directly adjacent. All critical signals had clearly labeled test points.

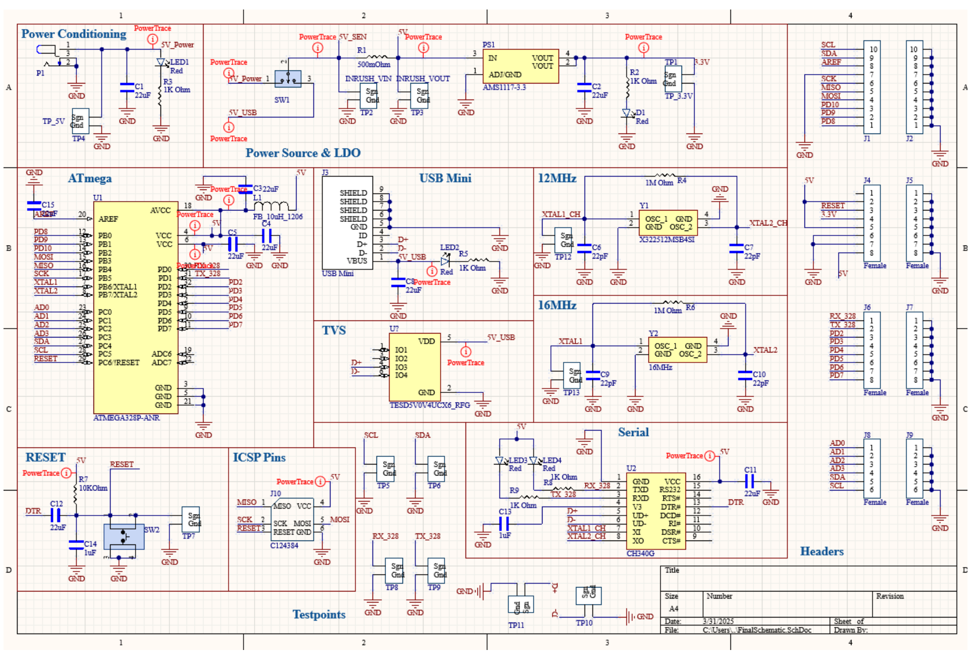
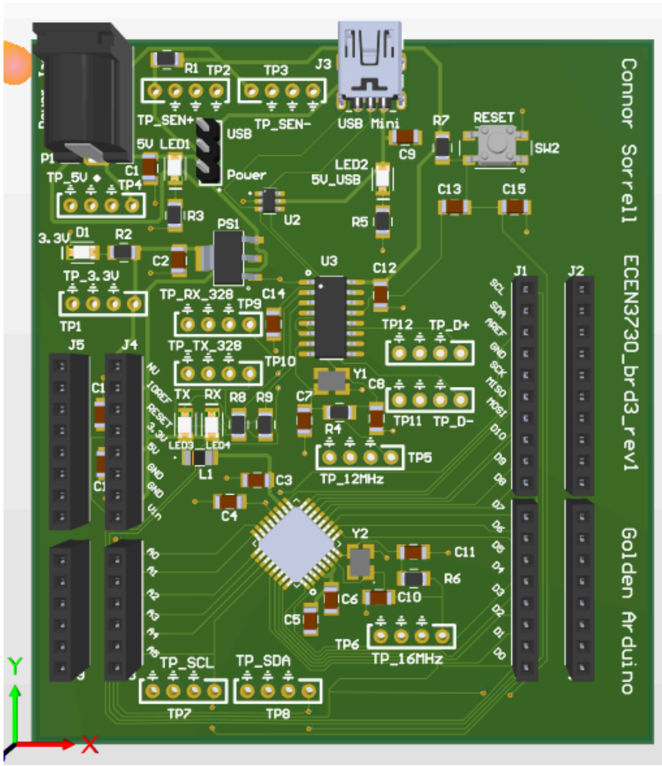
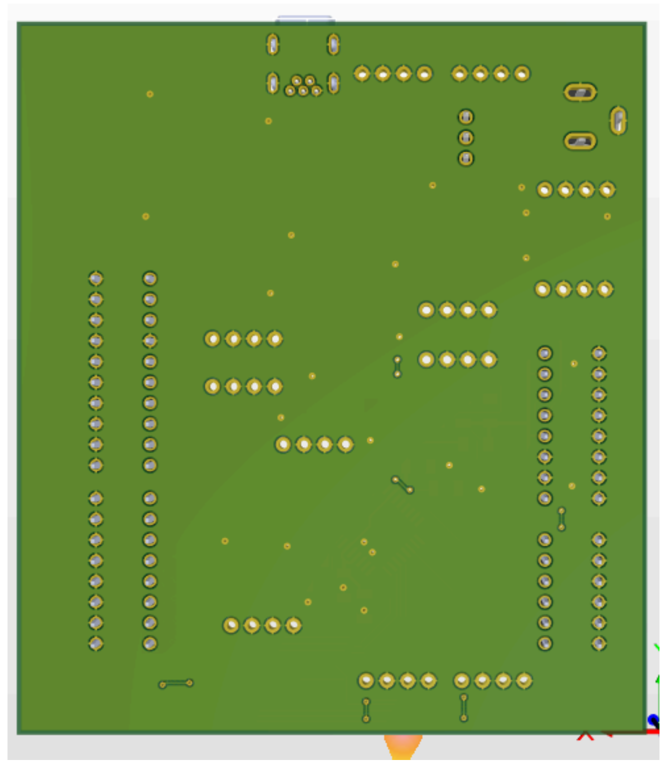


Figure 1: Shows the completed schematic in Altium

2. PCB Layout





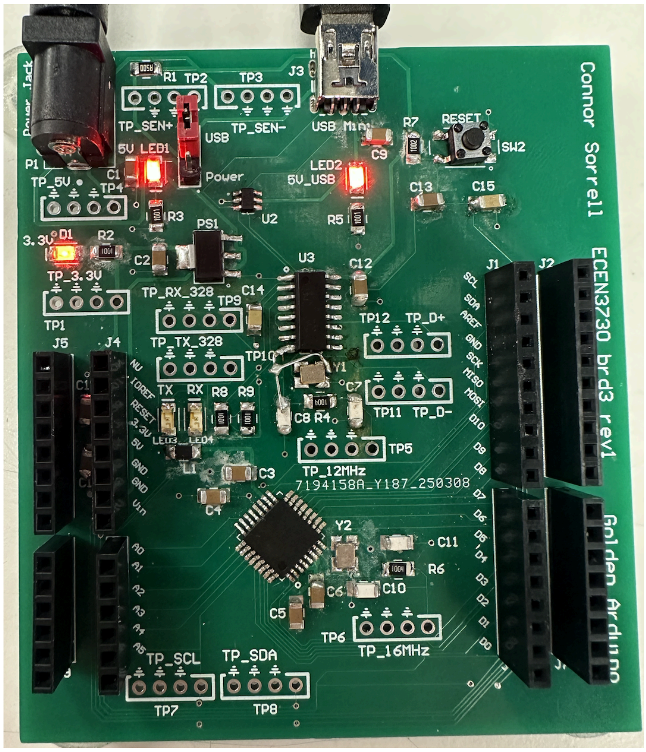
Figures 2 and 3: Shows the front and back of the PCB layout in Altium

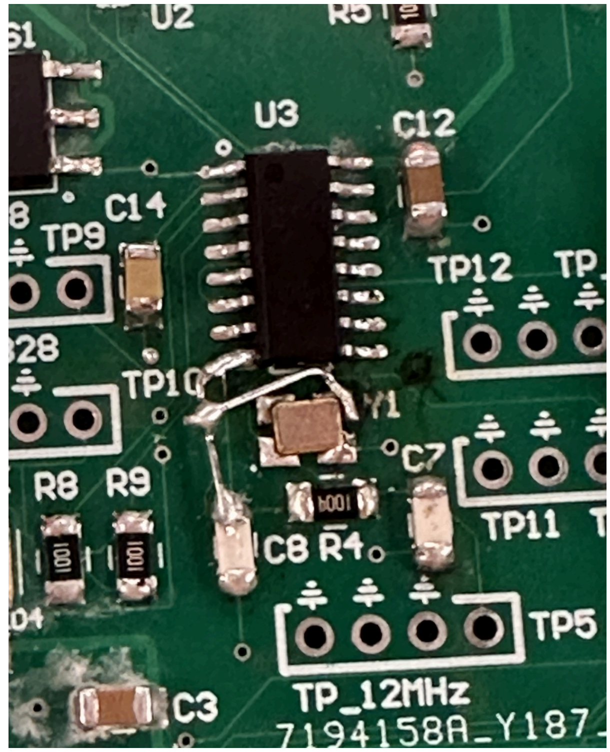
Layout decisions were driven by signal integrity and testability:

- Crystal and CH340G were traced for minimal loop area and trace symmetry (as close together as possible)
- Decoupling capacitors placed as close as possible to each power pin
- Power rails (VCC, AVCC, USB 5V) routed thick (20mm)
- Ferrite bead isolates AVCC from digital switching noise
- Continuous return plane present
- Test points labeled and accessible

3. Assembly & Fabrication

I assembled the board by hand using hot air and a soldering iron. The AtMega was the trickiest component to solder, especially due to pad size.



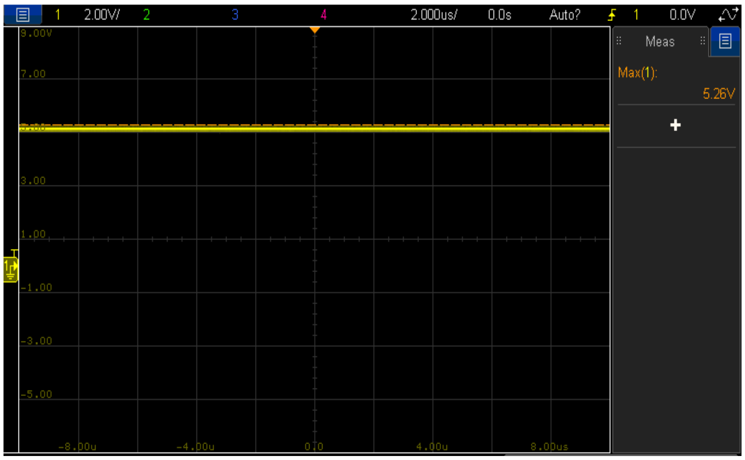


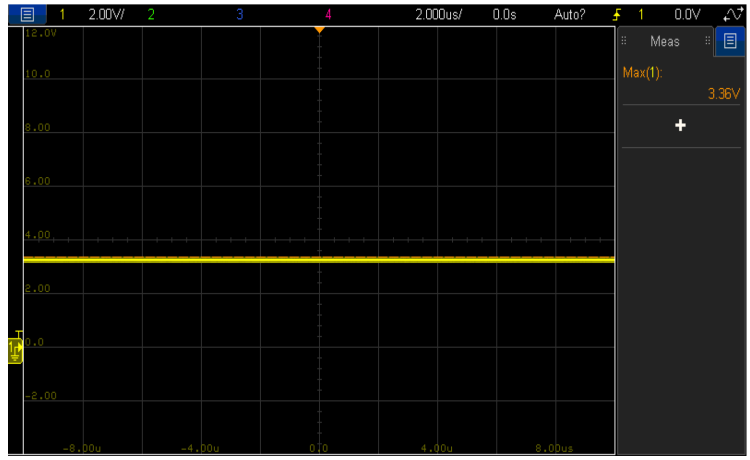
Figures 4 and 5: Show the completed, powered-on PCB, as well as the soldering "hack" I had to use to connect the paths of the 12MHz crystal (more about this in part 6: Problems)

I took special care to avoid solder bridges on the USB mini, CH340 and AtMega pads. Debugging the CH340 later taught me just how fragile these connections can be.

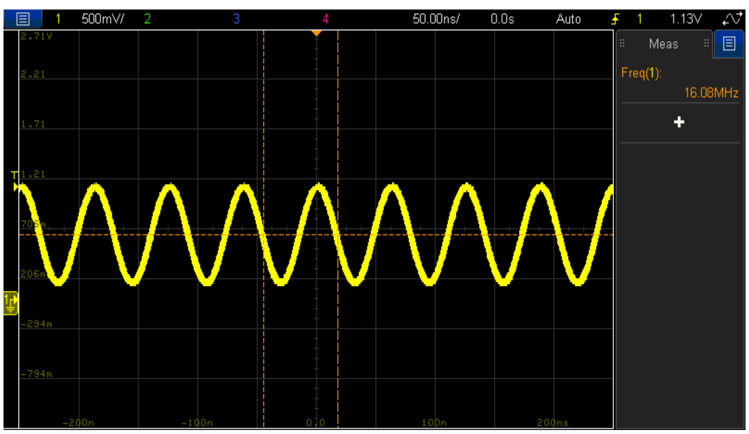
4. Testing & Measurements

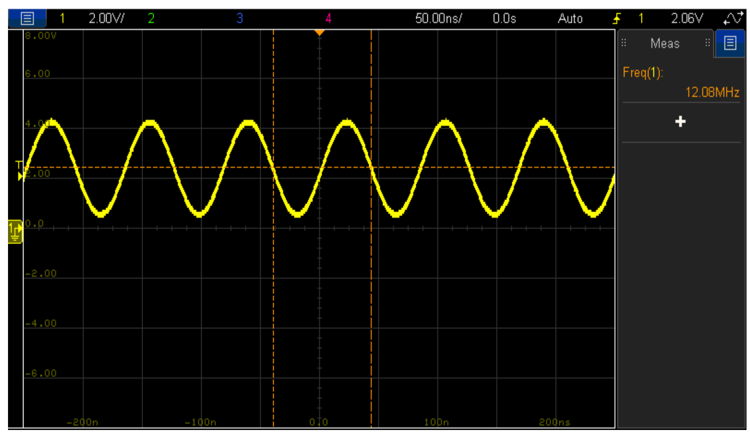
4.1 Baseline sanity checks for Board power, USB communication, and AtMega





Figures 6 and 7: Confirm proper power delivery on the board and LDO is working as expected to convert 5V to 3.3V





Figures 8 and 9: Confirm both the 12MHz and 16MHz crystals are working properly

The 5V and 3.3V working as intended mean that the PDN, for the most part, works as intended. The 16MHz crystal confirms a successful bootload, and a working 12MHz crystal is a strong hint towards successful USB communication.

4.2 USB mini check

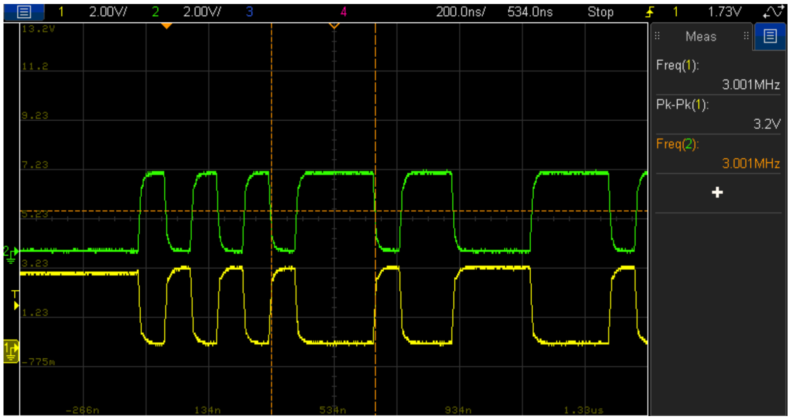
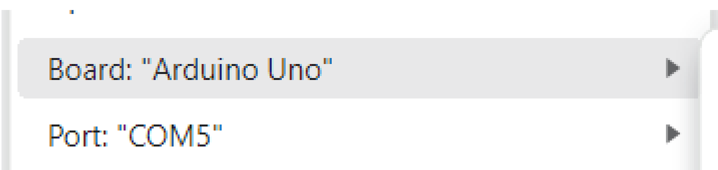


Figure 10: Shows the USB D+ and D- lines during communication. The signals toggle at 3MHz as expected and showcase a 3.3v peak to peak swing as expected.

- This verifies the USB section of the board is working properly and that the USB interface is alive and communicating

4.3 Bootloader Flashing & Uploading Blink Sketch



- After burning the arduino and connecting the USB, I can detect the arduino on my computer, proving full Arduino compatibility. This is the defining test of the board’s functionality.
- I then uploaded a simple sketch on Arduino which is designed to blink pin 13 at 5 Hz.

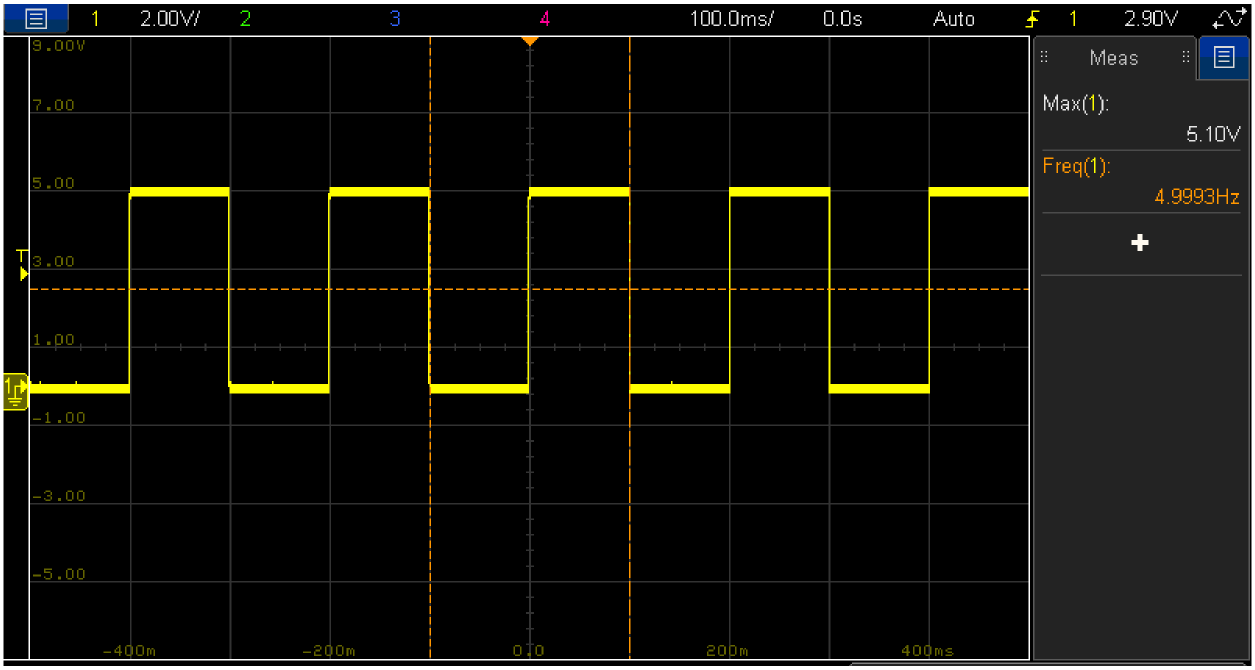
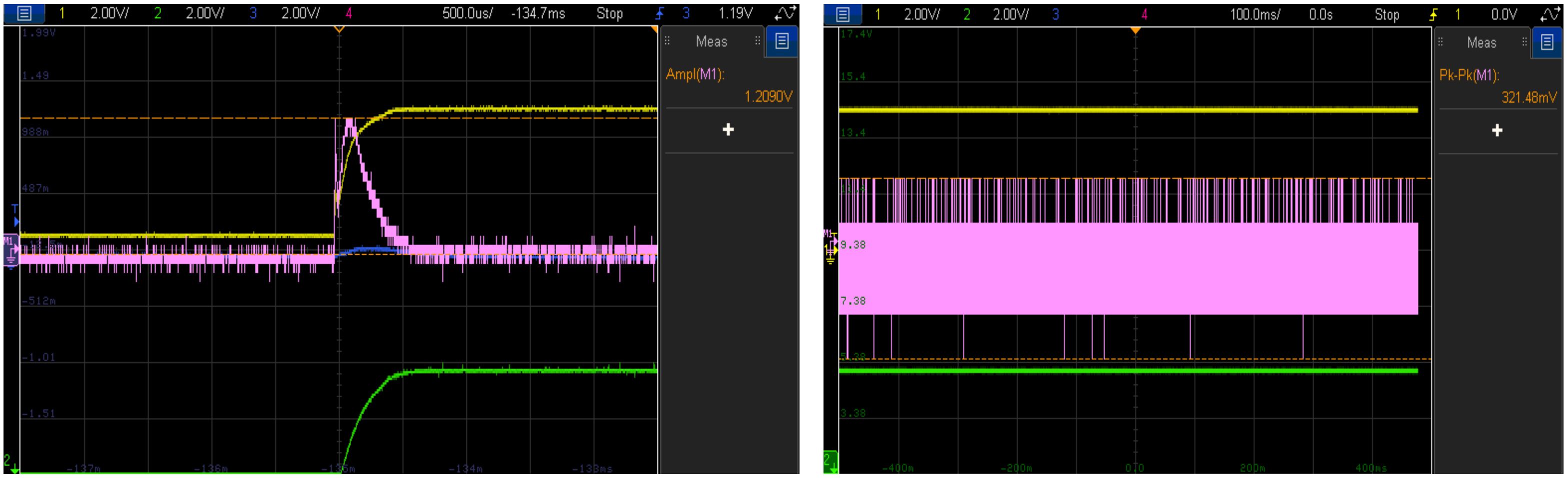


Figure 11: Shows the output of pin 13 after uploading the blink sketch.

- This shows that the ATmega328P is not only bootloaded but also fully programmable. It closes the loop between schematic, layout, assembly, USB communication, and software upload.

4.4 Inrush Current Measurement

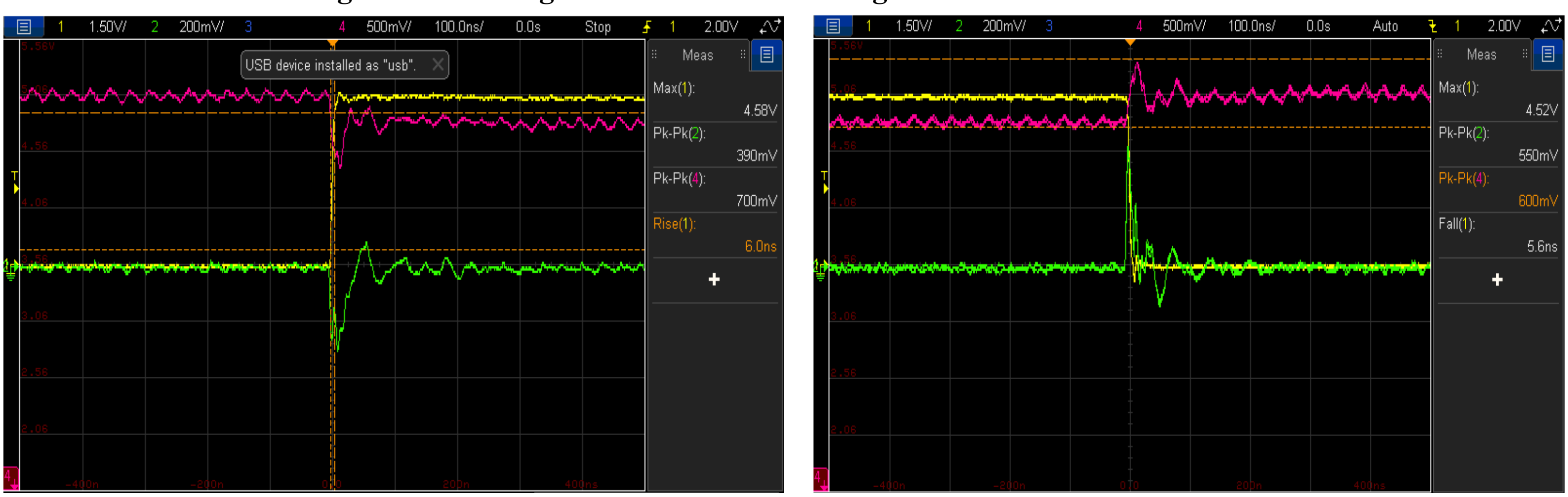


Figures 12 and 13: Shows the inrush current (both steady state and transient)

- In steady state, we see a 320mV drop over the 1 Ohm sense resistor, meaning we see a steady-state current draw of about 320mA.
- When the power supply is initially plugged in, we see a 1.2V drop, corresponding to an inrush current draw of 1.2 A.
- Measuring inrush current teaches us how much stress our power rail will see at plug-in. It tells us if we need additional filtering or protectional components.

Now, we compare my Arduino to the commercial Arduino:

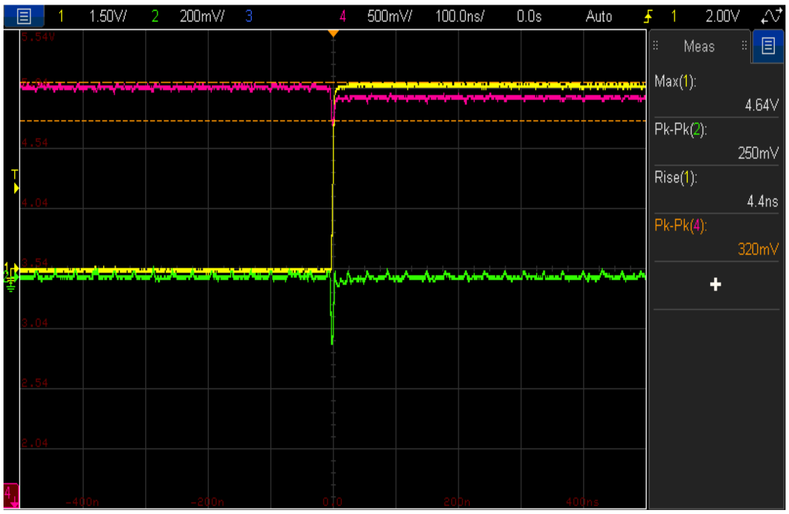
4.5 Switching Noise and Edge Times when the AtMega switches states

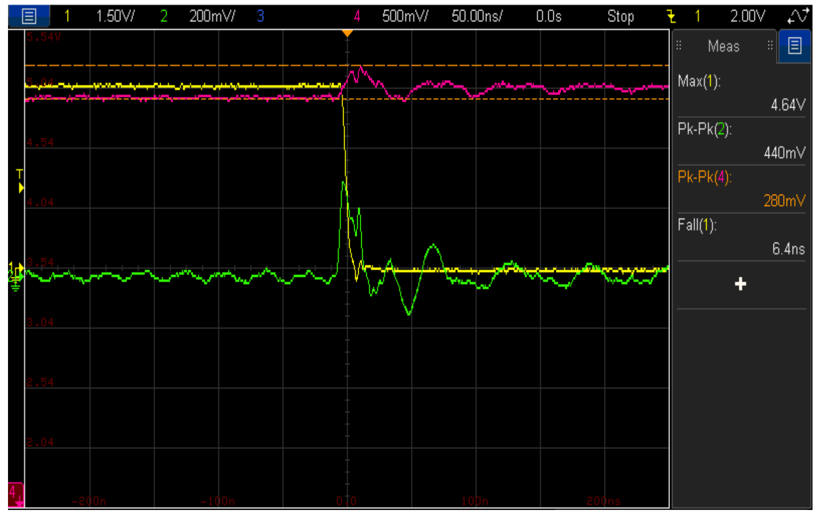


Figures 14 and 15: Shows the rise/fall times of Pin 13 (yellow waveform), shows the quiet high noise (pink waveform) and the ground bounce on quiet low (green waveform)

From the screenshots, we observe that on the **Commercial Arduino**, we see rising edge voltage collapse of 700mV on the quiet high pin, with 390mV of ground bounce on the quiet low pin, and a 6ns rise time.

On the falling edge, we observe 600mV of power rail collapse (quite high), and 550mV of ground bounce (quiet low), with a 5.6ns fall time.





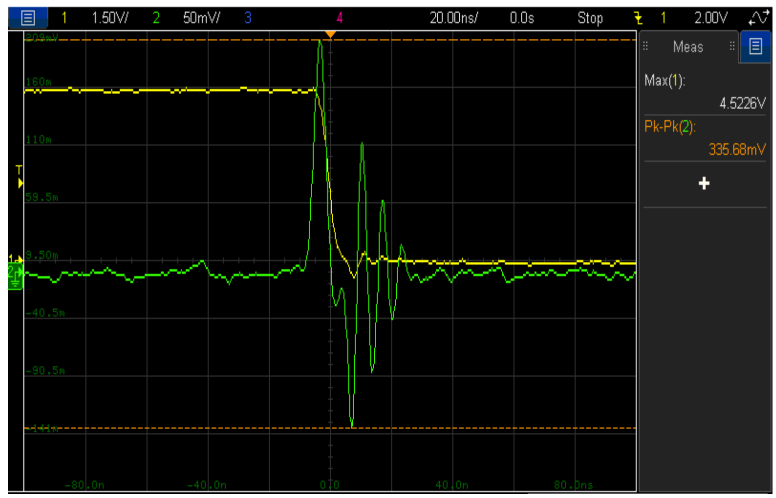
Figures 16 and 17: GOLDEN ARDUINO– Shows the rise/fall times of Pin 13 (yellow waveform), shows the quiet high noise (pink waveform) and the ground bounce on quiet low (green waveform)

On my Golden Arduino, we see a rising edge voltage collapse of 320mV on the quiet high pin, with 250mV of ground bounce on the quiet low pin, and a 4.4ns rise time. On the falling edge, we observe 280mV of power rail collapse (quite high), and 440mV of ground bounce (quiet low), with a 6.4ns fall time.

This comparison confirms the effect of a good layout, decoupling, and PDN structure. The cleaner (less noise) quiet LOW shows that inductive transients are successfully managed. The cleaner quiet HIGH shows that the power rail is stable and the added decoupling capacitors are effectively serving their purpose.

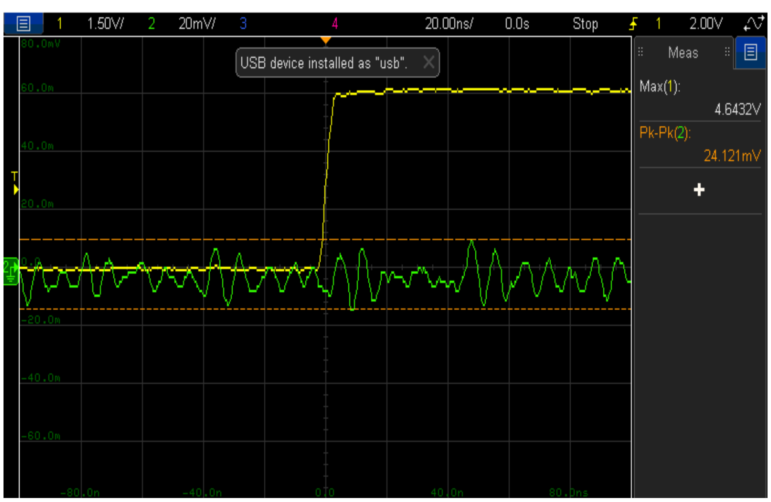
4.6 Near Field Emissions

These emissions directly correlate with the quality of the layout. A tight power delivery and smartly managed return paths result in lower emissions. With pin 13 triggering the scope, I shorted a probe tip to the ground return, forming a loop to pick up the near field emissions from the board



Figures 18 and 19: Shows the near field emissions on the ***Commercial Arduino***

- The rising edge shows an amplitude of 250mV of emissions
- The falling edge shows 336mV of emissions



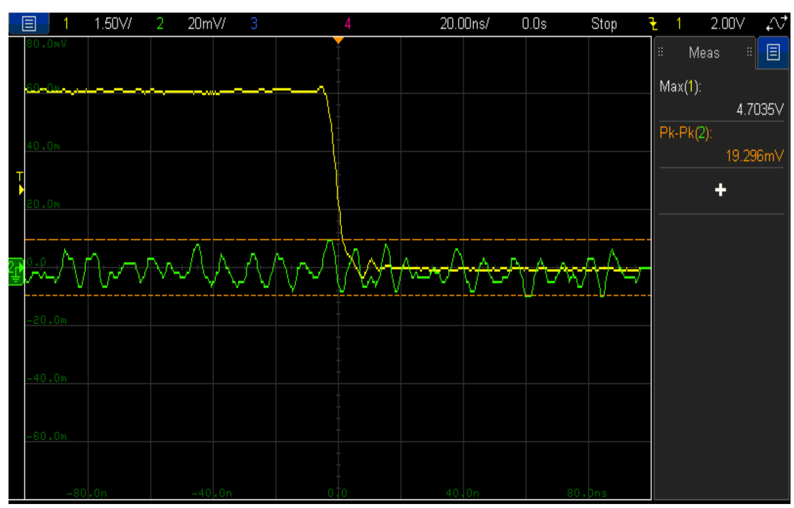
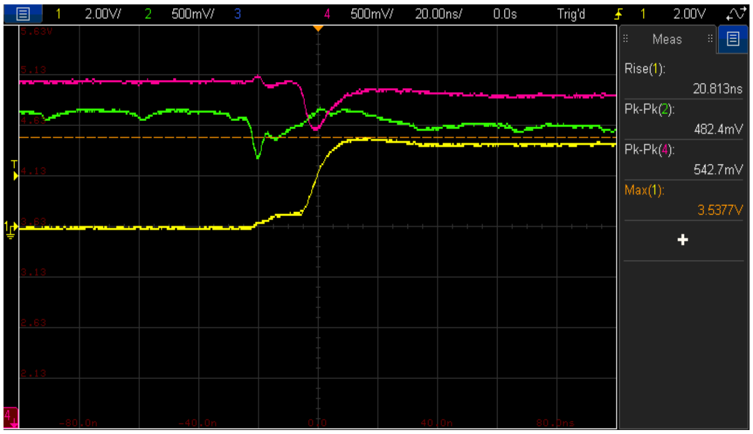
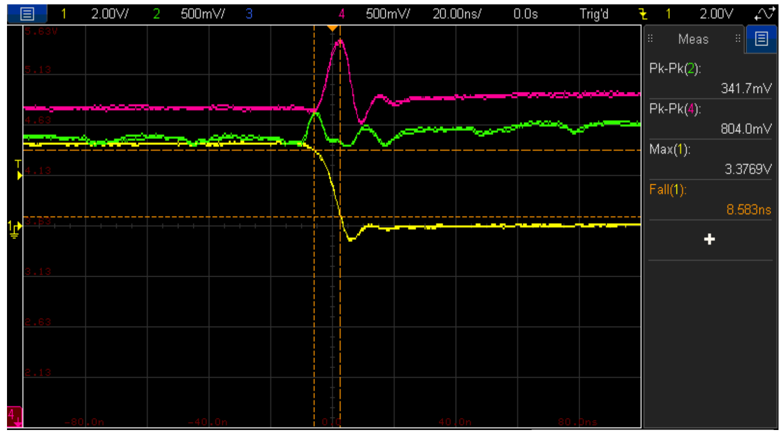


Figure 20 and 21: Shows the near field emissions on the ***Commercial Arduino***

- On my **Golden Arduino**, I measured 24mV of rising edge NFE and 19mV of falling edge NFE, over 10x better.

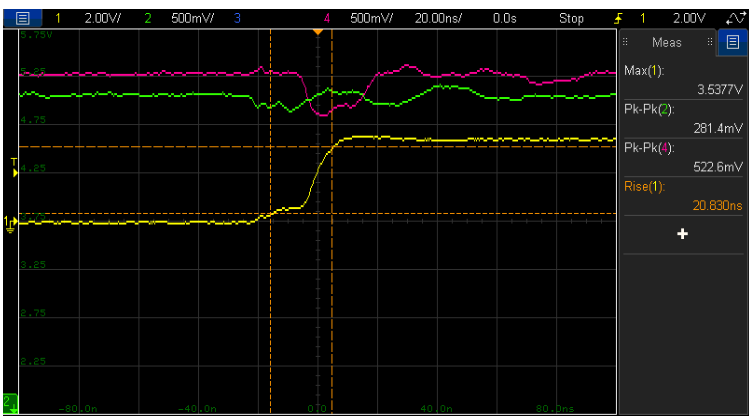
4.7 Quiet HIGH power rail noise when the slammer circuit switches on

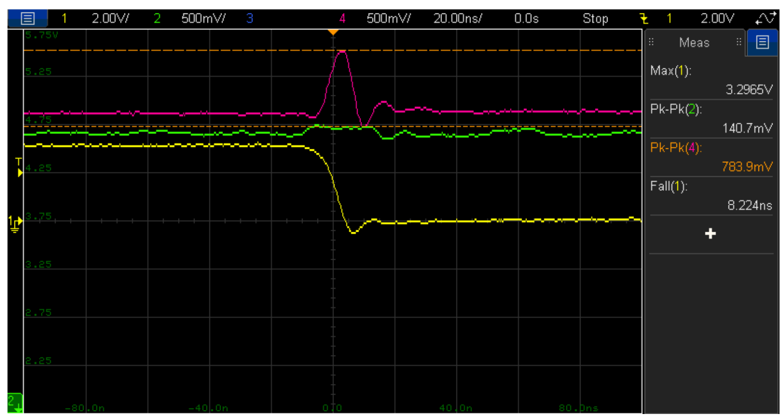




Figures 22 and 23: Show the on die power rail switching noise (slammer circuit)

- On the rising edge of the **commercial arduino**, we see 482 mV of noise on the die rail (quiet high) (green waveform)
- On the falling edge, we see 342mV of noise on the 5V die (quiet high) (green waveform)





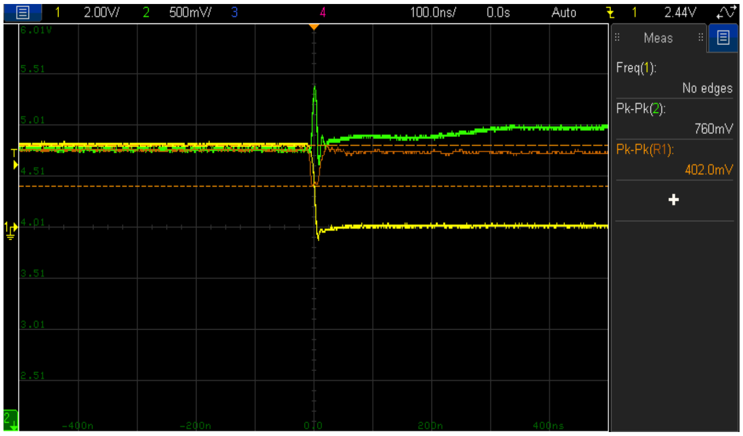
Figures 24 and 25: Show the on die power rail switching noise (slammer circuit)

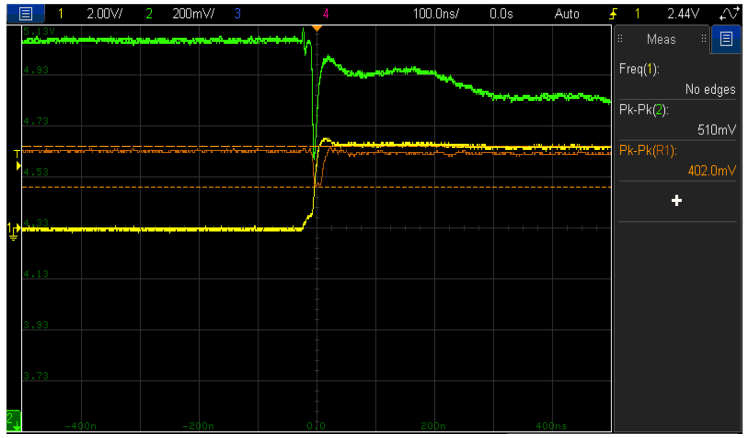
(MY ARDUINO)

- On the rising edge of **my Golden Arduino**, we see 281mV of noise on the 5V die (green)
- On the falling edge, we see 140mV of noise on the 5V die.

As shown, we see over a 2x decrease in noise on the 5V die (quiet high) pin. This noise reduction on the quiet high rail shows that my board was developed with better practices; closer/more decoupling capacitors, lower impedance return plane, which causes the switching transients to have less of an effect on noise present in the power rail.

4.8 Board level power rail noise when the slammer circuit switches on





Figures 26 and 27: Figure shows the board level power rail noise (slammer circuit)

- In the screenshots, the green waveform shows the measurements on the commercial Arduino.
  - We see 760mV of noise on the rising edge, and 510mV of noise on the falling edge.
- The reference waveform shows the measurements taken on my Golden Arduino.
  - In this case, we see 400mV of noise on both edges.

- The improvements in these measurements show that the power delivery on my Golden Arduino does a better job at handling transients. The design decisions like tight decoupling, minimizing loops, and the solid return plane are the reasons for these improvements on the power rail.

5. Analysis & Discussion

Did it work?

Yes. But, it took a lot of work and debugging. After identifying and correcting a missing connection between the CH340G and the crystal (pin 8 not connected to X0), the board fully came to life.

Measured vs Expected:

Bootload AtMega	See 16MHz on the crystal for AtMega	✓
USB	Com Port Appears on PC, 12 MHz is seen	✓
Blink Code Upload	See a 5Hz propagation on pin 13. Digital pins work as programmed	✓
Reset pin	Works as intended. Rise time of 150 ms after pressing before it is back to high. Confirms proper debounce circuit	✓
USB mini output	The scope shows the differential signaling at 3MHz	✓
UART output	TX lines toggle when sending/receiving data. Confirms USB to UART works properly.	✓
Proper PDN	5V can be delivered through either USB or power jack. An LDO properly converts to 3.3V	✓

Measurements–Commercial Arduino vs Golden Arduino

	Commercial Arduino	Golden Arduino
Voltage collapse multiple aggressors	700mV (rising), 600mV (falling)	320mV (rising), (> 2x better) 280mV (falling)
Ground bounce multiple aggressors	390mV (rising), 550mV (falling)	250mV (rising) (~1.5x better) 440mV (falling)
On die power rail noise (slammer)	480mV (rising), 340mV (falling)	280mV (rising), (~2x better) 140mV (falling)
Board level power rail noise (slammer)	760mV (rising), 510mV (falling)	400mV (rising), (~1.5x better) 400mV (falling)
NFE	250mV	19mV (10x better)

6. Problems & Debugging

This board taught me more about debugging than any prior build.

Problems I Faced:

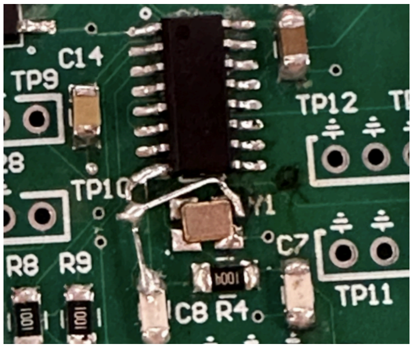
- CH340G crystal pin not connected, which left the USB device to not be detected

- Spent hours probing signals, checking solder joints, and checking connection with the DMM using continuity mode.
- Verified 16MHz crystal was working, but 12 MHz was not
- Bootloaded perfectly, but still had no connection to USB
- Reassembled the CH340 incase it was bad
- Verified 5V and 3.3V worked as intended. Entire board worked great aside from this issue
- Eventually, after looking at the Altium layout, I realized pin 8 (X0) was not routed to the crystal due to a netlist issue in Altium.
- No DRC errors flagged this since the symbol was present.
- I had to manually solder a thin jumper wire from the pin to the crystal pad.

- Tore off CH340G pin 8 leg and pad

- In the rework process, I ended up ripping the pad off the board, and disconnecting the pins leg from the IC.
- This resulted in even more troubleshooting, but I eventually worked around the issue by applying a lot of solder to pin 8 in order to reestablish the connectivity, and then finishing off by rerouting my jumper wire so that it sat comfortably inside the extra solder.

This issue though frustrating was invaluable. It taught me to always check what's *really* going on in the layout when troubleshooting. In the future, I will always look at my PCB layout before trying to debug other things like soldering connections. If only I had realized my issue sooner, I could have avoided countless hours of troubleshooting. Altium isn't always about red lines or "visual" connections. It's about *real* continuity, which takes a lot of review and verification, something I will pay even more attention to moving forward. Encountering this problem taught me more than I could have imagined. Through debugging, I learned every in and out of the board. It made me a better solderer, better problem solver, it taught me more about individual components, and most of all, gave me an invaluable skill of why to never give up, as I eventually got it working.



7. What Could Be Improved

- I didn't include ICSP headers because they added layout complexity. In the future, I'll route them through test points or on another layer.
- USB trace routing could have been tighter and matched in length.
- I could have made the board roughly half its size. I left a lot of unused space, knowing I will come back to this project in the future.
- I could have done a more intensive and stronger CDR on my Altium layout, in which I would have noticed an unconnected netlist, which would have prevented hours of troubleshooting/debugging

8. Conclusion

Three cool takeaways from this board:

- My layout decisions directly impacted my arduino's performance, which improved on the commercial arduino.
- My mistakes directly caused real failures
- My fixes directly led to the board coming alive (after a very lengthy process)

Other conclusions/takeaways/learning experiences/thoughts:

- A “working” design is not only about whether or not it functions.
  - Getting a board to turn on and perform its tasks is just the bare minimum. True indicators of success have to do with its performance, such as its ability to mitigate noise, its power distribution, etc.
  - Every single decision I made, from where I placed a decoupling cap to how I routed a trace ended up directly impacting the board’s performance
- My board showed roughly a 2x improvement in power integrity compared to the commercial board, which proves that decoupling placement, PDN, and a solid return path are of utmost importance.
- I drastically improved on many skills in this lab. I gained a ton of experience in debugging, problem solving, scope measurements, probing, using the DMM, and even hacking wires onto the board.
  - Learned to trust engineering instincts but make sure I verify them very carefully
- Learned/reinforced my knowledge of sense resistors, inrush current measurements, thevenin measurements, and current sourcing characteristics
- Gained more practical experience searching for useful information within datasheets

- Overall, extremely happy with my working board and extremely happy with the process it took to achieve it. I learned more in this process than any other board. I am looking forward to the opportunity to make improvements on my board and send it back to the fab shop. Though I saw an over 2x improvement in several different noise measurements compared to the commercial Arduino, I have identified ways to make my board even better and am looking forward to doing so.