**Lab 4**

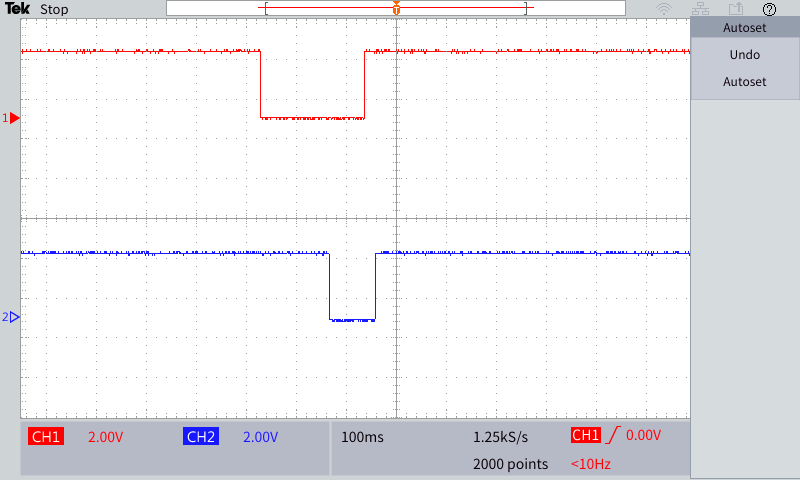
**Serial Communication**

Clayton Davidson | 000860643

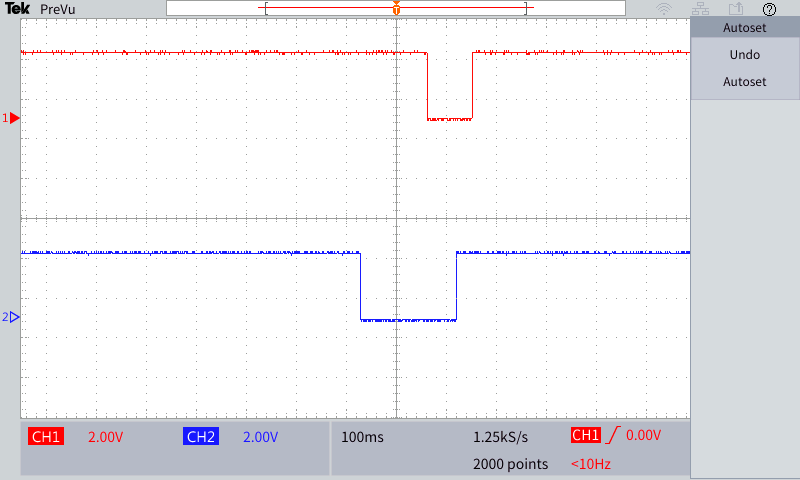
March 30, 2022

**P1**

Clockwise Turn - A(Clock) contacts first

****

Counter-Clockwise Turn - B(Data) contacts first



**P2**

Text

Description automatically generated

**YouTube Link:**

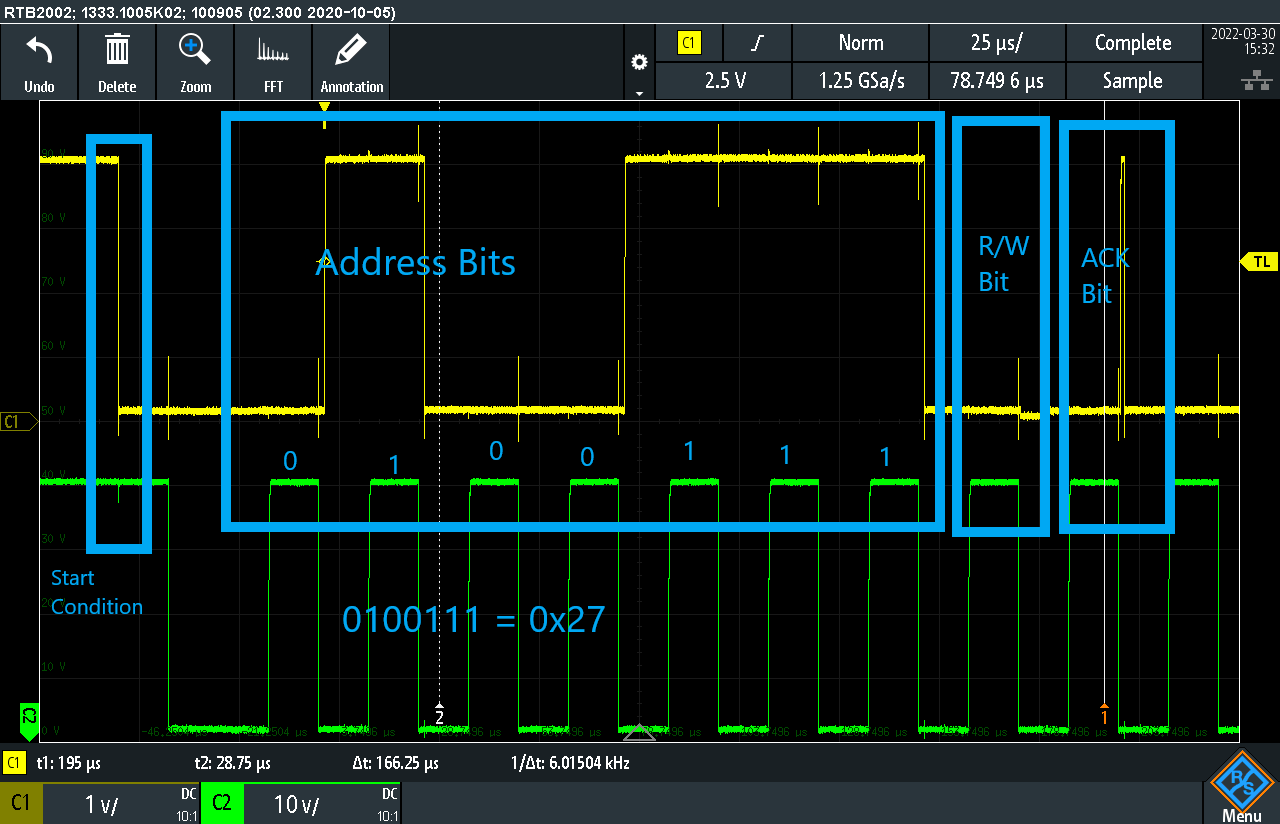
**P3**

**I2C Address :** 27

Text

Description automatically generated

**Correct Address:**

****

**Incorrect Address: A picture containing timeline

Description automatically generated**

**P4**

Text

Description automatically generated

****Text

Description automatically generated****

**P5**

**YouTube Link:** <https://youtu.be/utH49olhOnI>

**\*\*Please See Attached p5.py file for Commented Code\*\***

**Functionality:** This is somewhat explained in my video as well. When the program is run. A Main Menu screen is shown, indicating that you can scroll with the rotary encoder to view and select menu options.

There is a Menu with four options:

1. Temperature - which reads the local temperature from a DHT-11 sensor,
2. Humidity - which reads the local humidity of the air from the same DHT-11 sensor,
3. Voltage - which reads converted voltage value given with an ADS7830 ADC and a 3.3V 10kΩ potentiometer.
4. Exit - Closes the menu and ends the program.

**Select Functionality:** When at the Main Menu, if the rotary encoder is pressed instead of rotated, it takes you to that options Submenu which shows the values associated with that option, along with the unit of measurement. This value is refreshed every 0.5s, which provides time for the other program functionality to run smoothly and minimizes errors in the LCD display, but also produces a noticeable refresh delay on the values. If the encoder is pressed while in the Submenu, it will return to the Main Menu.

If Option 4 - Exit is selected, it will present an exit message, clear the LCD, and end the program.

**Critical Reflection**

Serial Communication is essential to IoT and is often overlooked as a vector for security risks. The first step of the lab was to understand on a hardware level how a change in voltage over a timescale (provided by turning the rotary encoder), when combined with a second connection, could be used to indicate information(rotational direction in this case).

The second step was to bring the hardware logic gained from Step 1 and turn it into software logic(python code).

Part 3 involved using the oscilloscope to detect and interpret more intricate data being sent through I2C to an attached LCD display. This data had a particular pattern that when compared with a second standard pattern (serial clock), can be used to interpret the data being sent, unencrypted, between components.

This data always started with:

* Start Condition - Indicated by Data channel moving from HV(high value) to LV(low value) and Clock channel staying at HV,
* Address Bits(7) - Indicated by either HV(1) or LV(0), when measured during HV Serial Clock,
* Read/Write Bit - Indicated by HV(Read, 1) or LV(Write, 0), when measure during HV Serial Clock,
* Acknowledge/Not-Acknowledge Bit - Indicated by a small spike of HV(ACK) or a longer LV followed by an end condition(NACK)

In the event of an ACK signal, the rest of the unencrypted communication would follow, which is a major security risk.

Part 4 introduced an LCD display that was used to write strings and custom characters to. Custom characters can be created with bitmaps as each “block” essentially functions like a 5x8 LED.  
Part 5 was by far the most complicated. This step required the making of a Menu on the LDC with at least 3 options (Temp, Humidity, Voltage) that, when selected, displayed the associated values.

There were several problems I encountered throughout the process. Using the oscilloscopes to properly display the bits would have been beyond me without extensive instruction and many questions. I also had issues with creating bitmaps of the symbols I was trying to create, but Aris shared an Excel document that made creating the bitmaps much easier. My largest issue was getting the RFID to work, which was abandoned after several hours for the ADC/potentiometer showing voltage as the third option. Something I would change for next time in Part 5, would be to discover a way to keep all function calls outside of the “while True:” loop. I could not figure out another way to have the selected values refresh constantly.

There are a few security concerns with Serial Communication directly, but most of them can be greatly mitigated by encrypting data that is sent over these connections. The other step to mitigate threats to IoT communication security is to minimize physical access to the devices.