BETA RELEASE

Report

Student: Leonel Gomez Flores
Course: ENCE-4231
Due: 05/04/2023

For: Dr. Martins

Introduction

The following report details the various design steps undertaken during the creation and testing of a Kitchen Timer project in its Beta Release version. It includes various aspects such as PCB design, component selection, and housing along with reasoning for any design choices.

Project Requirements

The Beta Release version expanded and added additional functionalities in regards to both hardware and software. The functionalities are described below:

(1) Replace Arduino Development Board & Shield with a custom PCB design and components.
(2) Voltage Regulation
(3) Power Source Decoupling
(4) Additional Crystal for Timer Capabilities
(5) Control Project from a Webpage (IoT)

These additional functionalities allowed the student to have the tools necessary to program the project with concurrent functions.

System Design

The system diagram below describes the general connections between the Microprocessor and project peripherals. This diagram does not show the ESD and fuse protections for the USB input.

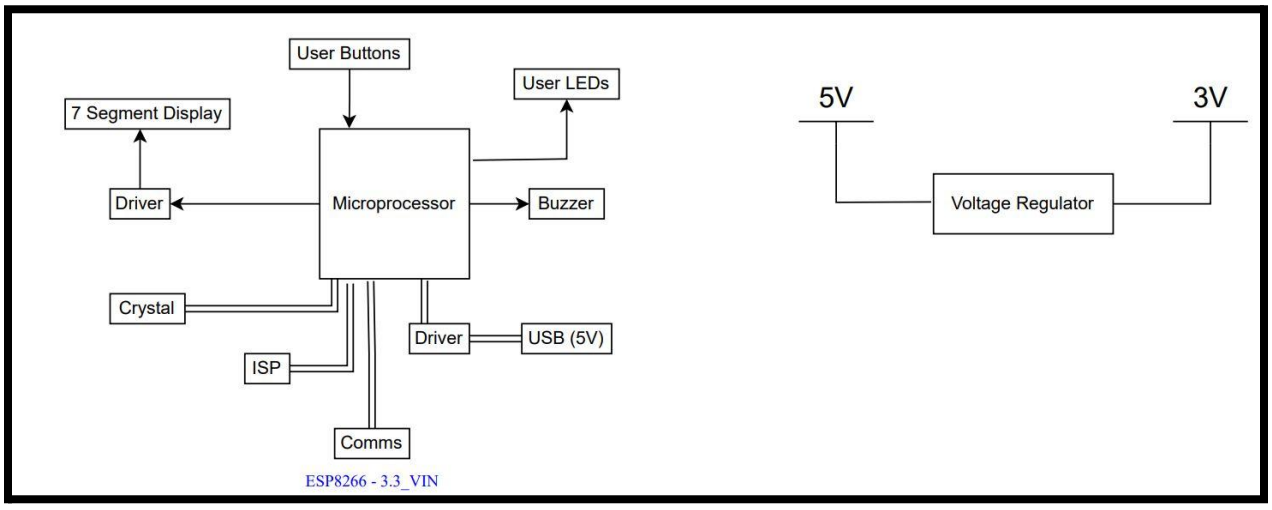

Figure 1 - System Block Diagram
(Phase_B_Sys_Diagram, ENCE-4321 Repository, Leonel Gomez)

Components Selection

The following components were selected for the initial prototype build (shield):

x1 Arduino Uno
x1 7-Seg Display
x1 Shift Register 8-Bit (SN74HC595)
x2 Buttons
x2 LEDs
x1 Buzzer
Resistors and Capacitors

The following components were selected for the Beta Release Version:

x1 ATMEGA32U4
x1 7-Seg Display
x1 Shift Register 8-Bit (SN74HC595)
x1 Voltage Regulator (LP2965)
x1 ISP Programmer (AVR ISP)
x1 ESP8266
x3 Buttons
x3 LEDs
x1 Buzzer
x1 Fuse
x1 16 MHz Crystal
x1 SOT-23 Transistor (ESD)
x1 USB Mini B
Resistors and Capacitors

As shown above, the number of components greatly increased from the initial prototype build. This introduced new capabilities such as IoT communication, thus components capable of supporting this facet were needed.

Build Prototype

This initial build prototype was fabricated with assistance and templates from Dr. Martins and originates from Phase A release of this project, therefore its more minor details lie outside the scope of this report.

This build prototype allowed students to program the 7-Segment display in conjunction with various libraries using the Arduino IDE. This prototype uses an Arduino Uno Development Board and a custom PCB shield derived from Dr. Martins' earlier development.

Key Differences from the Beta Release include:

- No IoT Communication
- Lacks Coupled Capacitors for Noise Reduction
- Lacks Additional LEDs and Reset Button
- Lacks Additional 16 MHz Crystal
- Needs to Interface w/ Arduino UNO

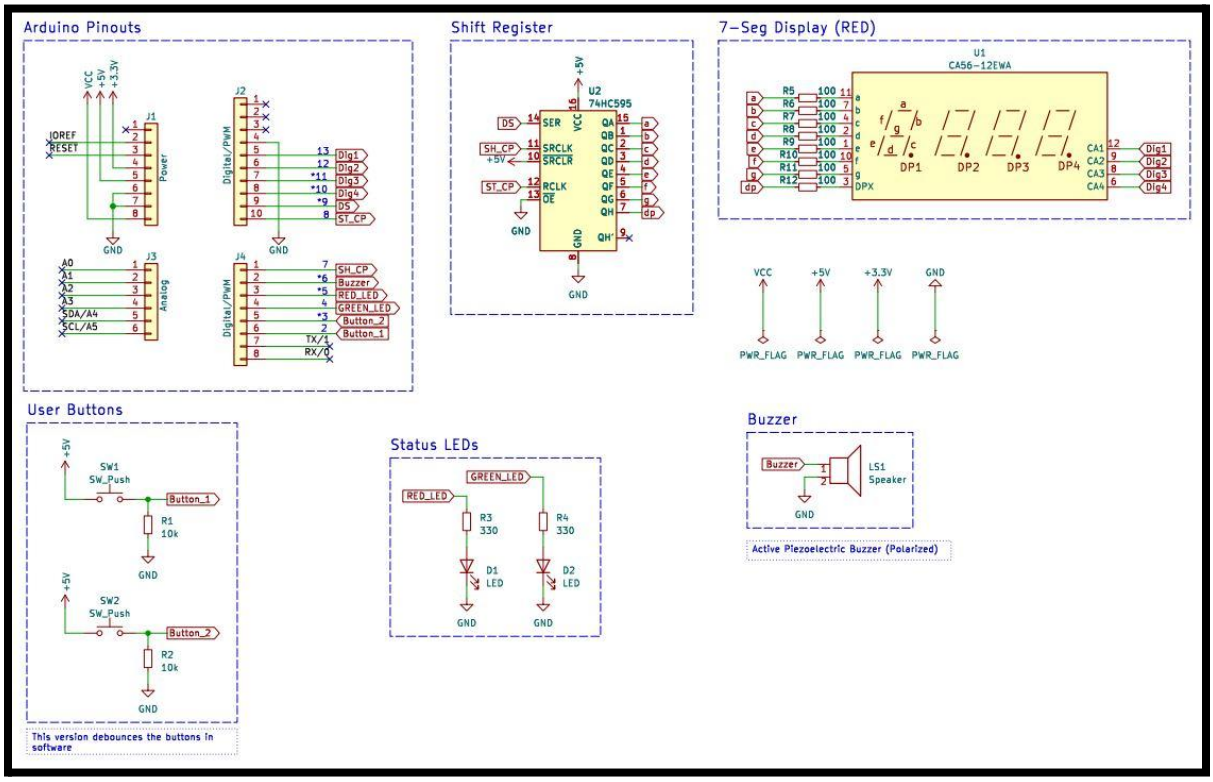

Figure 2 - Initial Prototype Schematic
(Phase A Schematic, ENCE-4321 Repository, Leonel Gomez)

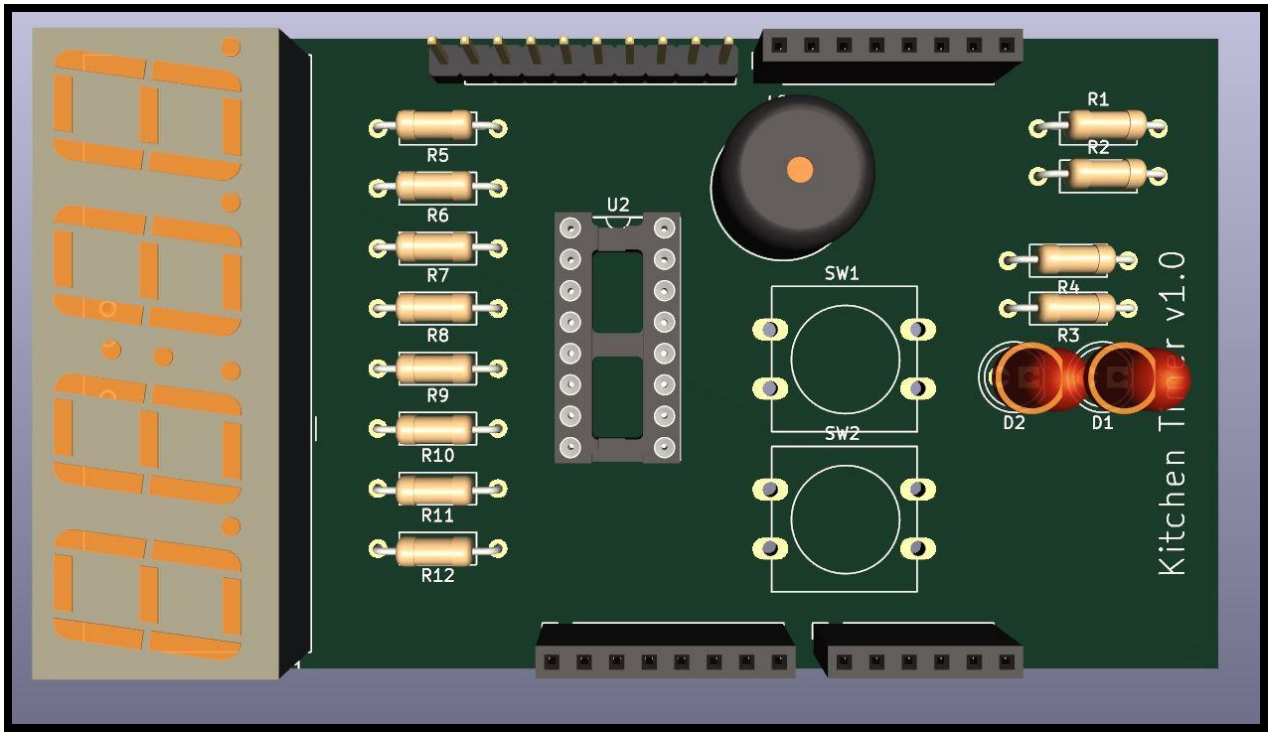

Figure 3 - 3D Render Layout for Prototype
(3D Render Plot, ENCE-4321 Repository, Leonel Gomez)

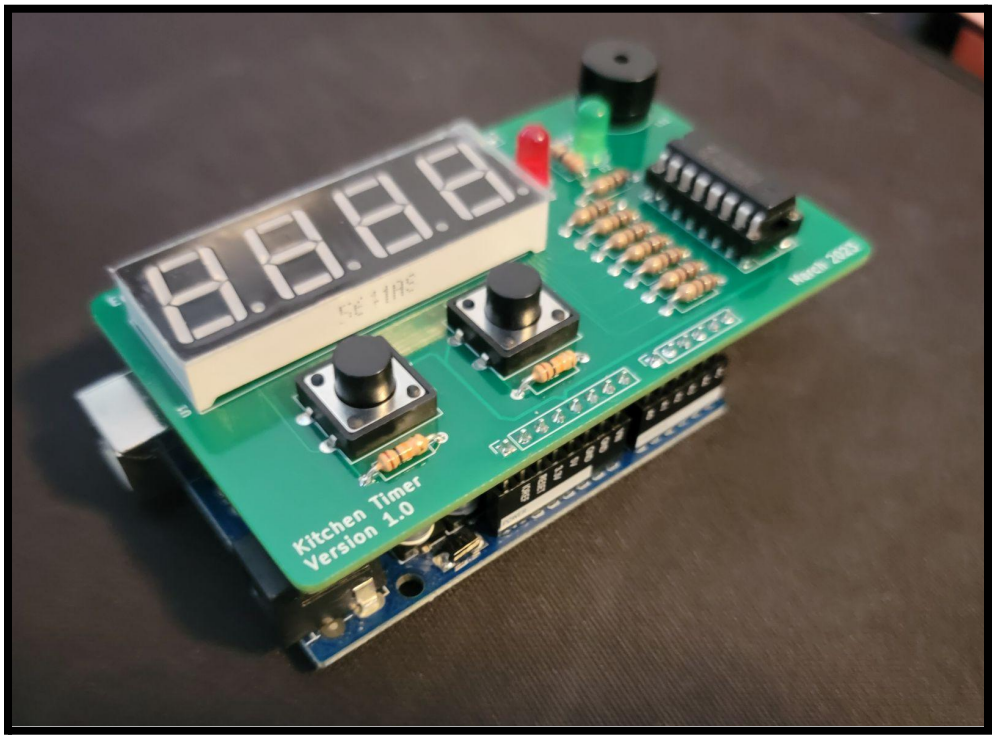

Figure 4 - Prototype Image
(PCB Image, ENCE-4321 Repository, Leonel Gomez)

PCB Design

The Beta Release version of this project included a much more complex board layout and demanded careful planning in order to fit and route components on a 10cm x 10cm area. As opposed to the prototype, this design is all on one board and fitted only with the components required. This allows the board to be lighter, more efficient, low level, and more cost effective because there is no need to purchase an additional microcontroller to interface with. Despite this, development boards allow one to prototype quickly with existing libraries and their importance shall not be diminished.

Some challenges included routing the 7-Segment display to necessary resistors, routing/placement of the MPU in a central, accessible location to the USB hardware, and various DRC issues that appear during design.

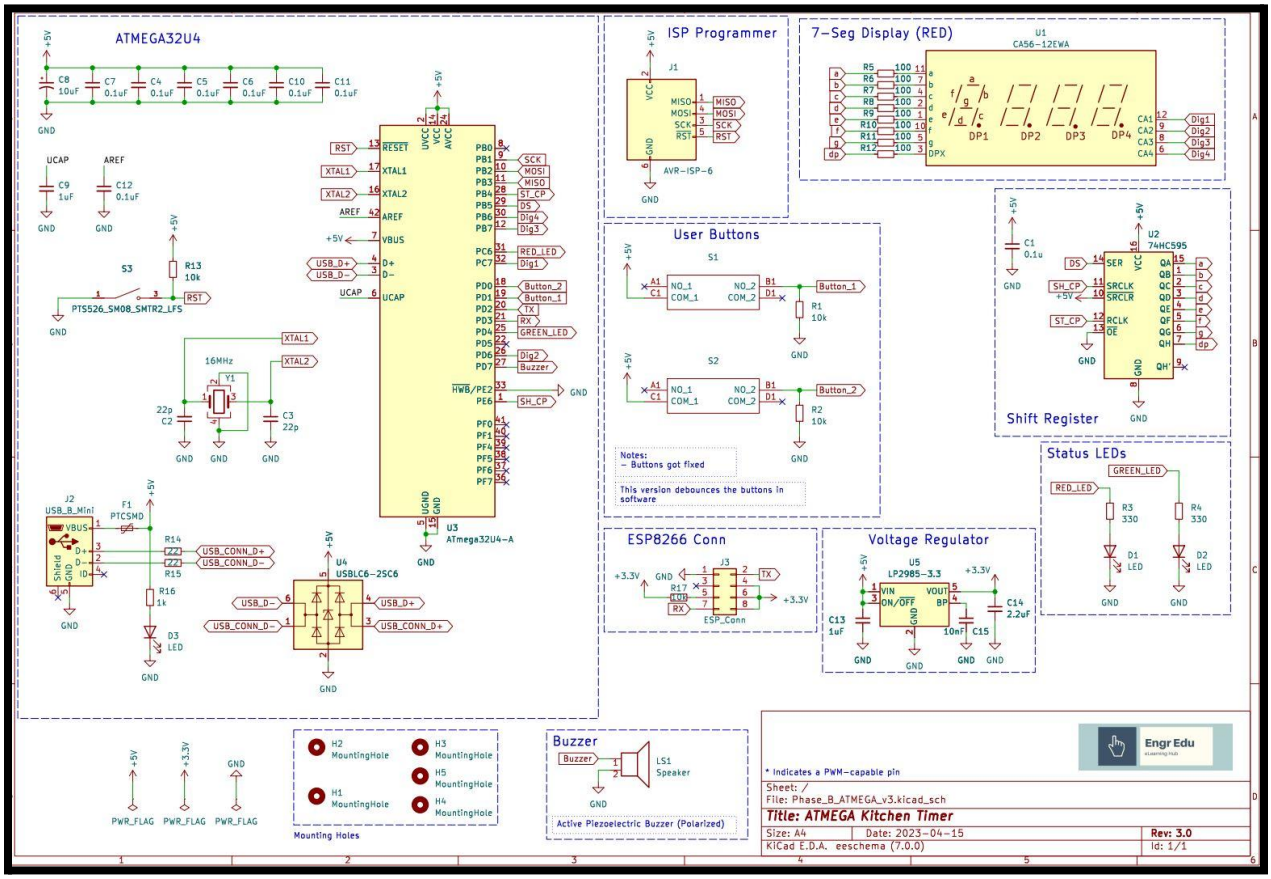

Figure 5 - Beta Release Schematic
(Phase_B_ATMEGA_v3.pdf, ENCE-4321 Repository, Leonel Gomez)

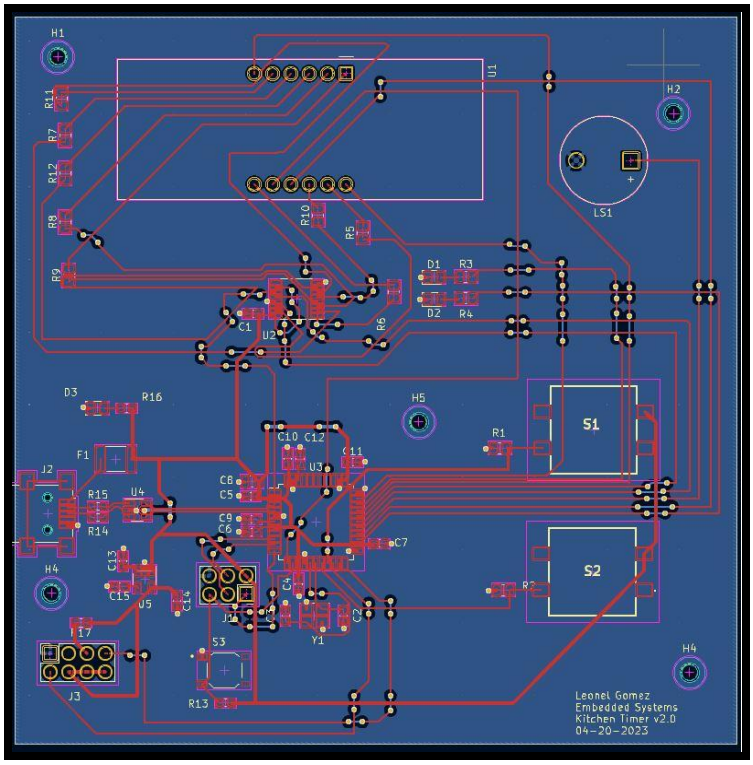

Figure 6 - Beta Release PCB Layout
(pcb_layout.jpeg, ENCE-4321 Repository, Leonel Gomez)

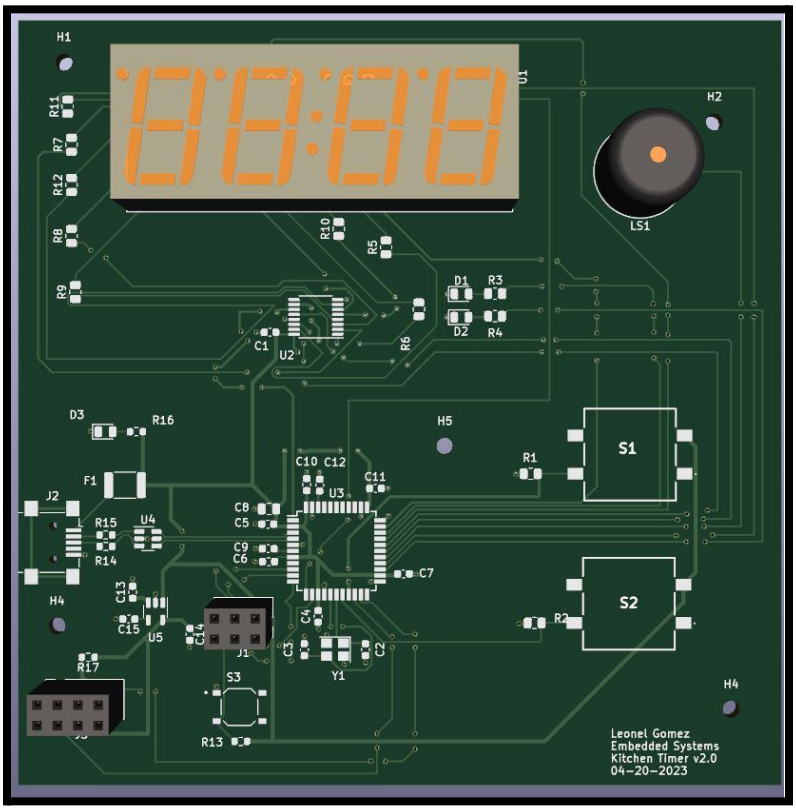

Figure 7 - Beta Release Model Front

(model_front_view.jpeg, ENCE-4321 Repository, Leonel Gomez)

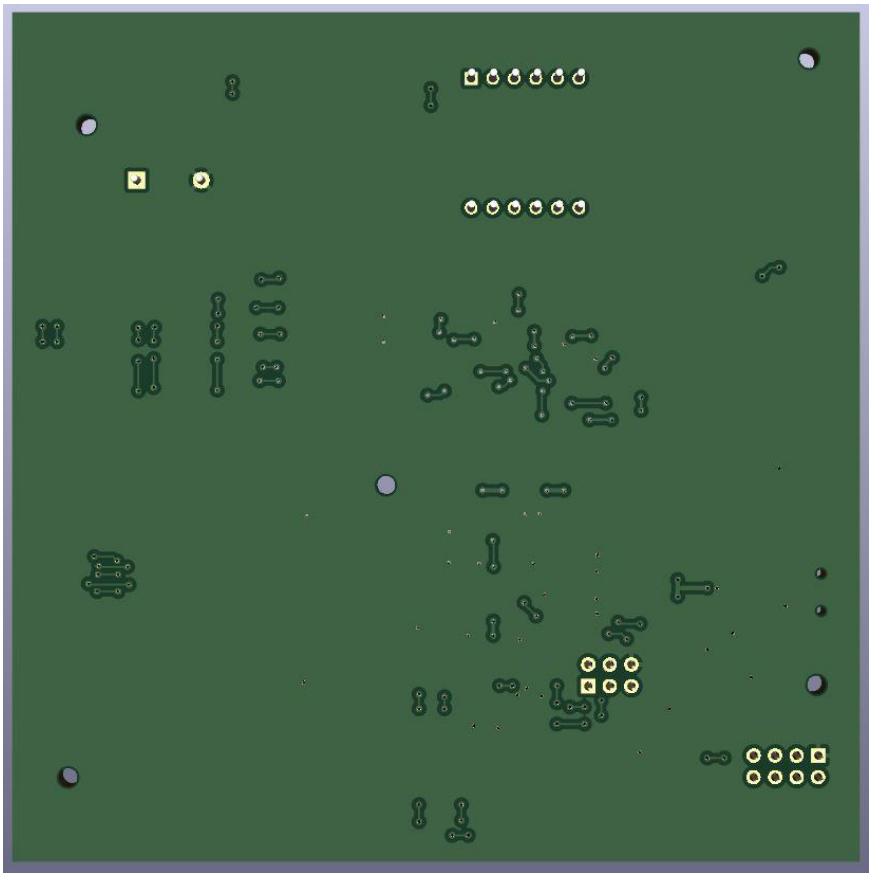

Figure 8 - Beta Release Model Back
(model_back_view.jpeg, ENCE-4321 Repository, Leonel Gomez)

Assemble Stage

At this point in time, the Assemble Stage is not included in the report. The custom PCB design is currently being fabricated and delivered to the author of this report for testing and component soldering. This report will be updated to include this section in the future.

Software Development

The following program has been altered from a provided file for the purpose of testing the Beta Release PCB.

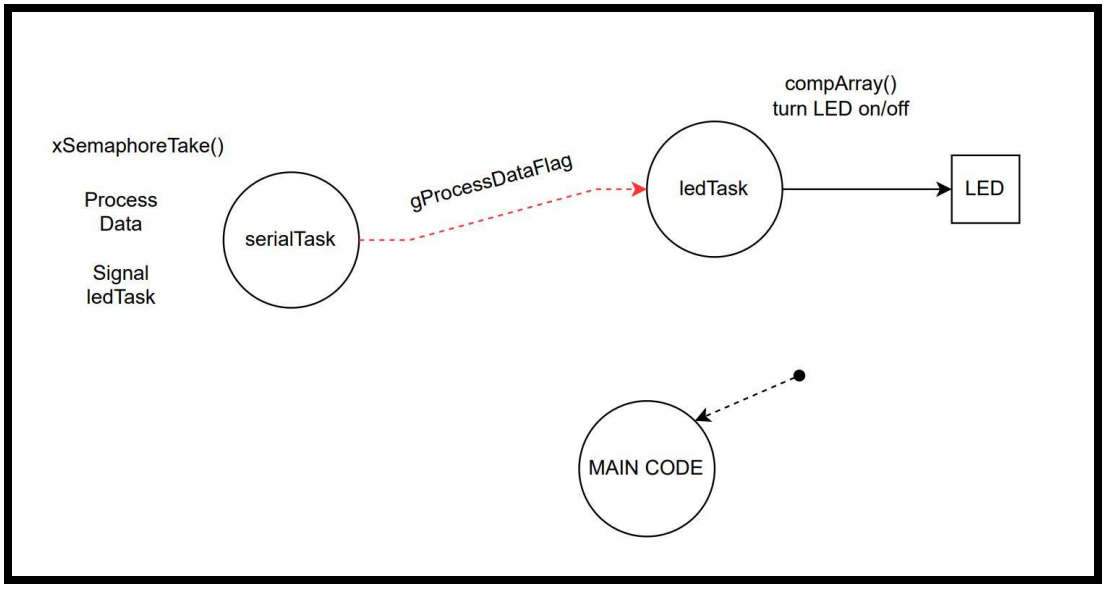

Figure 9 - Software Block Diagram
(soft_block_diag.pdf, ENCE-4321 Repository, Leonel Gomez)

```
if(xSemaphoreTake(gSerialSemaphore, portMAX_DELAY) == pdTRUE) { //wait till semaphore is available
  char gIncomingChar = Serial.read(); //read message from serial

  if(gPackageFlag == true) // incoming character belongs to data
  {
    gCommsMsgBuff[iBuff] = gIncomingChar; //add to buffer
    iBuff++; //increment buffer size

    // Buffer Size is at maximum
    if(iBuff == BUFF_SIZE)
    {
      gPackageFlag = false; //this prevents task from stalling if \n is never received
      gProcessDataFlag = true; //ledTask uses this flag to process the result
    }
  }
```

Figure 10 - Serial Task Code Snippet
(Phase_B_Program_WIP, ENCE-4321 Repository, Leonel Gomez)

```
// check if start of message
if(gIncomingChar == '$')
{
  gPackageFlag = true;  // beginning of message

  // Ensure buffer is clear
  for(int i=0; i<BUFF_SIZE; i++)
  {
    gCommsMsgBuff[i] = 0;
  }

  // reset index of array
  iBuff = 0;
}

// End of Message Check
if((gIncomingChar == '\n') && gPackageFlag)
{
  // Signal end of package
  gPackageFlag = false; //end reached
  gProcessDataFlag = true; //signal to other task that the message can be processed
}
```

Figure 11 - Serial Task Code Snippet
(Phase_B_Program_WIP, ENCE-4321 Repository, Leonel Gomez)

```
if(gProcessDataFlag) {

  gProcessDataFlag = false; //flag reset

  //compares first three chars with "STR"
  if(compArray(gCommsMsgBuff, "STR", 3) == 1)
  {

    //turn on LED
    digitalWrite(LED, HIGH);
  }

  //compares first three chars with "STR" using array comparison function
  if(compArray(gCommsMsgBuff, "STP", 3) == 1)
  {
    // Turn off LED
    digitalWrite(LED, LOW);
```

Figure 12 - LED Task Code Snippet
(Phase_B_Program_WIP, ENCE-4321 Repository, Leonel Gomez)

Unfortunately, I was unable to achieve concurrency with other peripherals like the 7-Segment display, external crystal, and buttons. Working with RTOS is challenging even with AI assisted tools. This serves to highlight areas to improve in/sink more time into.

However, there are key snippets from the program that are worth discussing. Figure 10 and 11 showcases the ability of the serial task to wait for an available Semaphore for a virtually indefinite amount of time. This task also increments the buffer, handles a max buffer size, and delineates between the beginning and end of a message ($ and \n) for use in the ledTask. Specifically, the gProcessDataFlag is set by the serial task and used in the logic of the ledTask

Figure 12 demonstrates the use of the compArray function to determine whether the message is ‘STR’ or ‘STP’ and set the LED accordingly. It utilizes the gProcessDataFlag to enter the logic.

Please refer to my GitHub Repository for a complete view of the program developed.

Enclosure Designs

The following housing design is fairly simple and does not currently have a faceplate paired with it. It features a cutout for the USB connection and can be easily expanded to account for mounting holes or other auxiliary features.

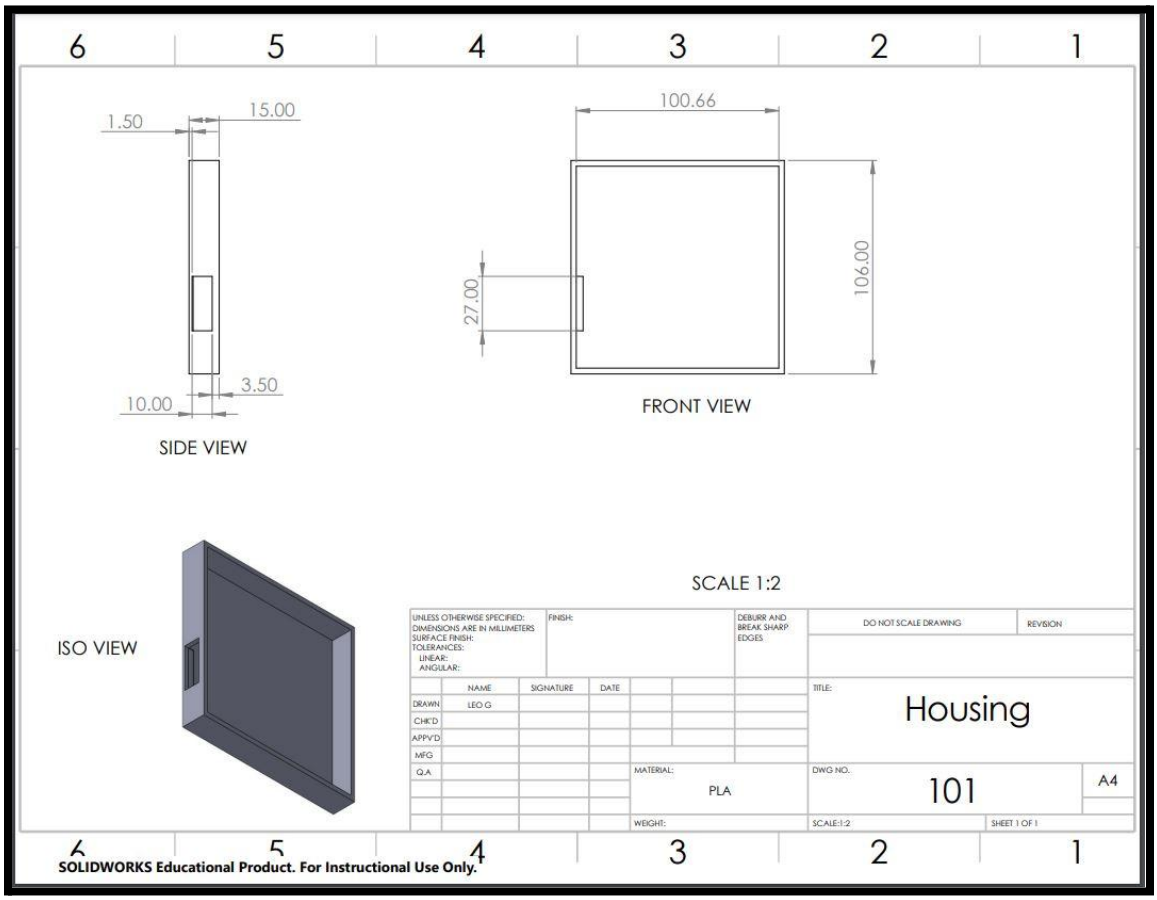

Figure 9 - Housing Drawing
(Phase_B_Housing, ENCE-4321 Repository, Leonel Gomez)

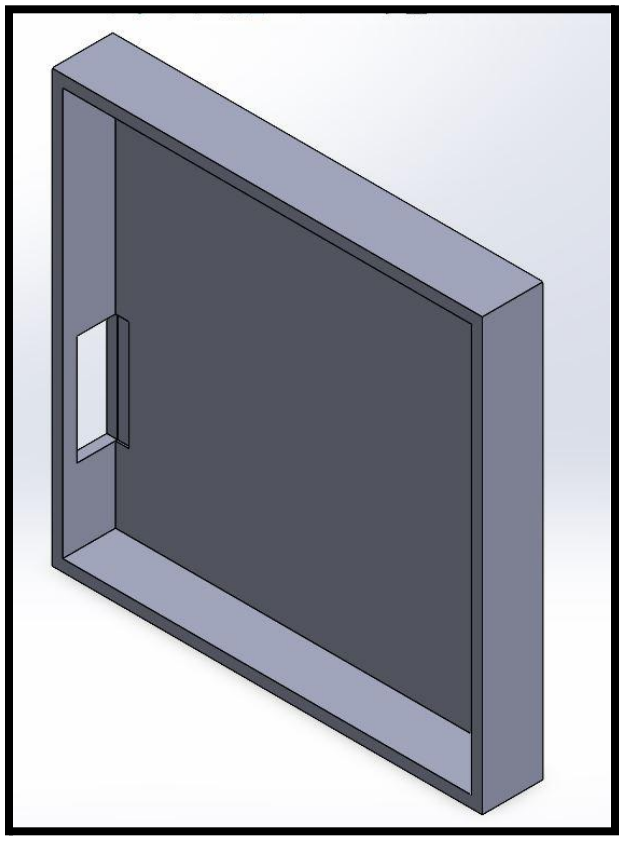

Figure 9 - Housing 3D Model
(Phase_B_Model, ENCE-4321 Repository, Leonel Gomez)

Ideally, in future releases there will be a faceplate with cutouts for the ESP, 7-Segment Display, Buttons, and LEDs in order to house the PCB more adequately.

Conclusion

The Beta Release of the Kitchen Timer project indicates a significant iteration in the engineering design process. Students were able to expand upon a previous design and increase the project's functionality significantly with only a few more key components and some software development. However, this was only one step in the engineering process and the project will see more revisions before a complete finalization.