Department of Electrical Engineering and Computer Science
EECS 395/495 – Engineering System Design
Winter 2017

# Final Report

Name: Karan Shah
Submitted to: Professor Mikhelson
Report Due: 3/15/2017

# Contents

# 1.Introduction

We developed a portable webcam that streams images to a website. It needs 5V power supply and must be connected to any Wi-Fi network in the vicinity. It can also be wall-mounted. The three main components of the system are:

- The camera module - OV2640DS,
- The Microcontroller - ATMEL SAM4S8B, and
- The Wi-Fi chip - AMW004.

Figure 1 shows the final circuit board. Figure 2 shows the completed webcam in its 3D printed case.

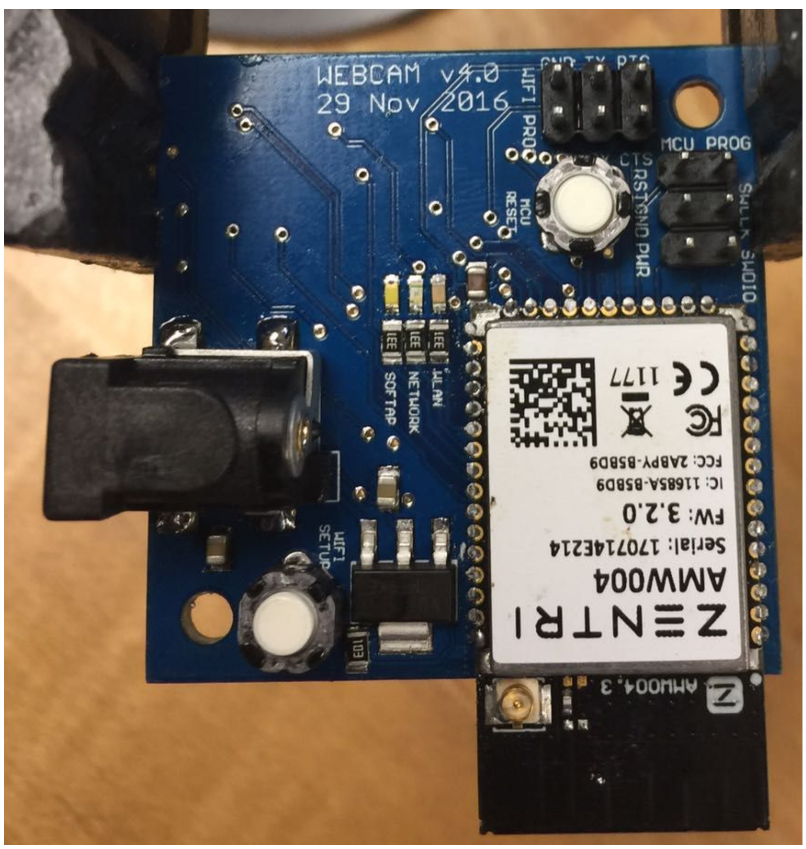



Figure 1: Circuit Board

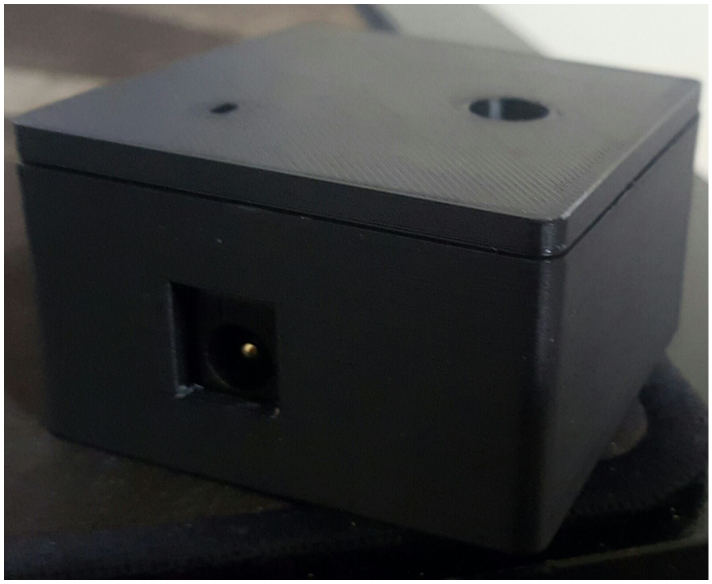


Figure 2: Completed Webcam

The camera module and the microcontroller communicate via the I2C interface. A picture taken by the camera is transferred via the Parallel Capture Mode of the MCU onto its Internal RAM. The MCU communicates with the Wi-Fi chip using its UART interface and transfers the image to the Wi-Fi chip to be stored in a *.jpg* file. On request by the user, the Wi-Fi chip displays the image on the website. This process takes place every two to three seconds when the button to start the webcam on the website is clicked. Thus, the image gets updated on the website with a timestamp every two to three seconds.

The final PCB design has a button to reset the system and a button to initiate web setup for the Wi-Fi chip to connect to an unknown Wi-Fi network using SoftAP. Furthermore, there are LED indicators placed for Power, Network, W-Lan, and SoftAP. Programming headers are placed for programming or debugging the MCU and the Wi-Fi chip.

In this report, the approaches taken and the challenges faced during the processes of PCB design, embedded system programming, website design, and 3D CAD design are discussed in brief. Also, the overall learning experience obtained from the course and suggestions to improvement of class format is discussed.

# 2.Design Process

## 2.1 PCB Design

The PCB Design was done in Eagle PCB design software. We designed our board with dimensions 39 x 50 mm and used both sides of the board as ground planes. We made the appropriate connections for the I2C and UART interfaces between the camera, MCU and the Wi-Fi chip and included push buttons to reset the system and to initiate web setup for the Wi-Fi chip.

The main challenge faced was proper placement of components with adequate spacing between them to make soldering easier and for the user to program the Wi-Fi and MCU easily. Another challenge faced was grounding the required pins and making sure all the connections were proper and connected to required pins without any overlaps. Figure 3 and Figure 4 shows eagle design schematic and layout respectively.

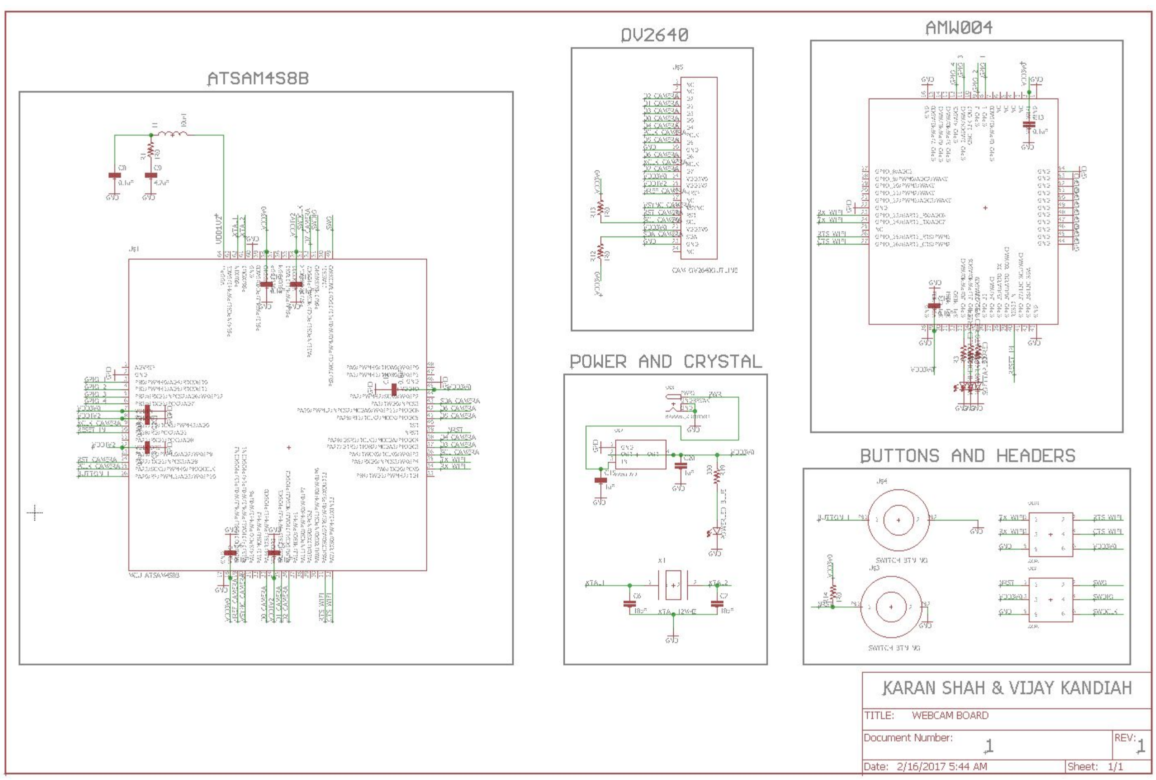



Figure 3: Eagle Schematic for PCB Design

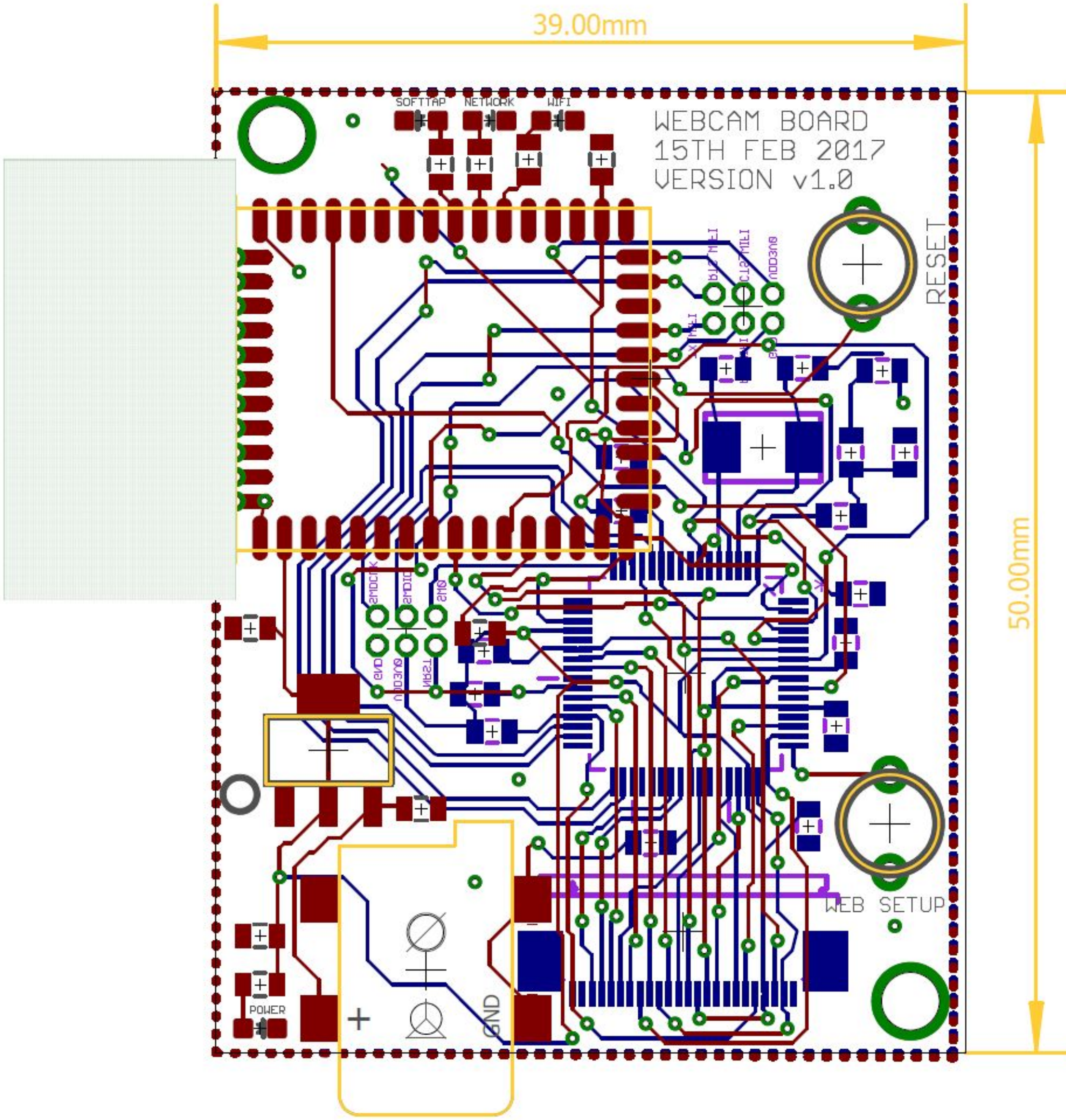



Figure 4: Eagle Layout for PCB Design

It was only after we received the manufactured board that we found out that our voltage regulator was not grounded properly as there was an isolated ground plane which was disconnected from the main ground plane. Also, the spacing of holes for the header pins were not enough to fit the programming headers. Upon realizing that our board would require some modifications and would be difficult to program and debug, we used the instructor's board for our final project.

During the surface mount soldering of the components onto the PCB, we faced many challenges. When soldering the MCU, some pins would get short and we had to either remove the short or resolder the MCU again. The same was the case with the Wi-Fi chip. After soldering each of these main components, the proper connections between the pads and pins were checked using a multimeter to detect shorts. Another challenge was soldering the camera connector because when the hot air gun is used to solder other components on the chip, the plastic of the connector melts and we need to de-solder it and solder in a new camera connector. But in the end, we soldered all the components properly and got the board working.

## 2.2 Embedded System Programming

The MCU was programmed using Atmel Studio software with drivers from Atmel Software Framework (ASF) incorporated into the project folder. The main flow of our program running on the MCU upon start is as follows.

- Initialise and configure all pins
- Wait for a network connection
- Poll all streams with a delay of 500ms till a websocket stream is opened
- Start image capture and find the length of the image to verify a valid image
- Write the image to a .jpg file on the Wi-Fi chip
- Display the image on the website and repeat from step 3.

The *web setup* button is configured as an interrupt to initiate web setup on the Wi-Fi chip whenever the button is pressed.

Flow control is enabled for UART communication between Wi-Fi and MCU and the responses of the Wi-Fi chip for the commands issued by the MCU is parsed and verified to keep a check on the proper flow of instructions in the code. The main function of our code is shown below.

The main function of our code is shown below.

```
int main (void)
{
	sysclk_init();
	board_init();
	ioport_init();
	wdt_disable(WDT);
	pmc_enable_pllack(7, 0x1, 1); /* PLLA work at 96 Mhz */
	configure_usart_wifi();
	wifi_web_setup_flag=false;
	configure_wifi_comm_pin();
	configure_wifi_web_setup_pin();
	configure_tc();
	ioport_set_pin_dir(USART0_RTS_IDX,IOPORT_DIR_OUTPUT);
	ioport_set_pin_level(USART0_RTS_IDX,false);
	ioport_set_pin_level(OV_RESET,false);
	delay_ms(1);
	ioport_set_pin_level(OV_RESET,true);
	init_camera();
	uint8_t find_len = false;
	uint8_t cap = false;
```

```
        /* Reset Wifi and wait for network */
        ioport_set_pin_level(WIFI_RESET, false);
        delay_ms(100);
        ioport_set_pin_level(WIFI_RESET, true);

        while (!ioport_get_pin_level(NETWORK)){}
        write_wifi_command("set system.cmd.prompt_enabled off\r\n", 1);

        write_wifi_command("set system.cmd.echo off\r\nh\r", 1);

        while(1)
        {
                while (!ioport_get_pin_level(NETWORK)){}
                while(!stream_flag){
                write_wifi_command("poll all\r\n", 1);

                delay_ms(500);
                }
                stream_flag = false;
                 cap = start_capture();
                while(!cap){}

                find_len = find_image_len();
                cap = false;
                while(!find_len) {}

                write_image_to_file();
                find_len = false;
                write_wifi_command("stream_write 0 1\r\n ", 1);
        }
}
```

Some of the important functions in our code are discussed below.

```
uint8_t find_image_len(void)
{
        uint8_t len_found = false;

        for( uint32_t i = 0; i< array_size; i++)
        {
        if ((image_buf[i] == 255)&& (image_buf[i+1] == 216))// Find Start frame FFD8
                {       im_start = i;
                        break;
                }
        }

        for( uint32_t j = im_start; j< array_size; j++)
        {
        if ((image_buf[j] == 255)&& (image_buf[j+1] == 217))// Find End frame FFD9
                {       im_length = j+2 - im_start;
                        len_found = true;
                        break;
                }
        }
```

```
	return len_found;

}
```

This function finds the start of image frame *FFD8* and the end of image frame *FFD9* and calculates the length of the image by subtracting their locations in the array. This length is fed into the write image to file function shown below.

```
void write_image_to_file(void)
{

	usart_write_line(BOARD_USART, "fde img.jpg\r\n");
	delay_ms(10);
	uint8_t file_create_string[100];
	sprintf(file_create_string, "fcr img.jpg %d\r\n", im_length);
	usart_write_line(BOARD_USART, file_create_string);
	delay_ms(10);
	for (uint32_t jj=im_start;jj<(im_start + im_length);jj++)
	{
	usart_putchar(BOARD_USART, image_buf[jj]);
	}
	delay_ms(10);
	while(!process_flag){}
	process_flag = false;

}
```

This function issues a command to delete the already stored image on the Wi-Fi chip and then creates the .jpg file with length found from the previous function. The function then proceeds to fill the file with the data of the image and waits for responses *'File Created'* and '*Success'* from the Wi-Fi chip before returning.

The main challenges faced in embedded system programming is the initialization and configuration of all the pins and interfaces properly. While writing the code with references from a few examples took a reasonable amount of time, an extensive amount of man hours was spent on debugging. Almost all the errors in our base code were due to improper pin configurations and initializations of interfaces like the USART and TWI.

## 2.3 Website Design

Our website has three created pages and a link to open the Wi-Fi Chip's setup page in a new tab. Figure 5 shows our homepage which is opened when a user types in the system's IP address in a web browser. Figure 6 shows our webcam page which displays the image stream with a timestamp and a log to monitor web socket connections. There is also an info page which shows our information. Each webpage has its own .html and .css file and the webcam page has a .js file to run the scripts for the timestamp, to start and stop the webcam, and to display the log.


Figure 5: Home page

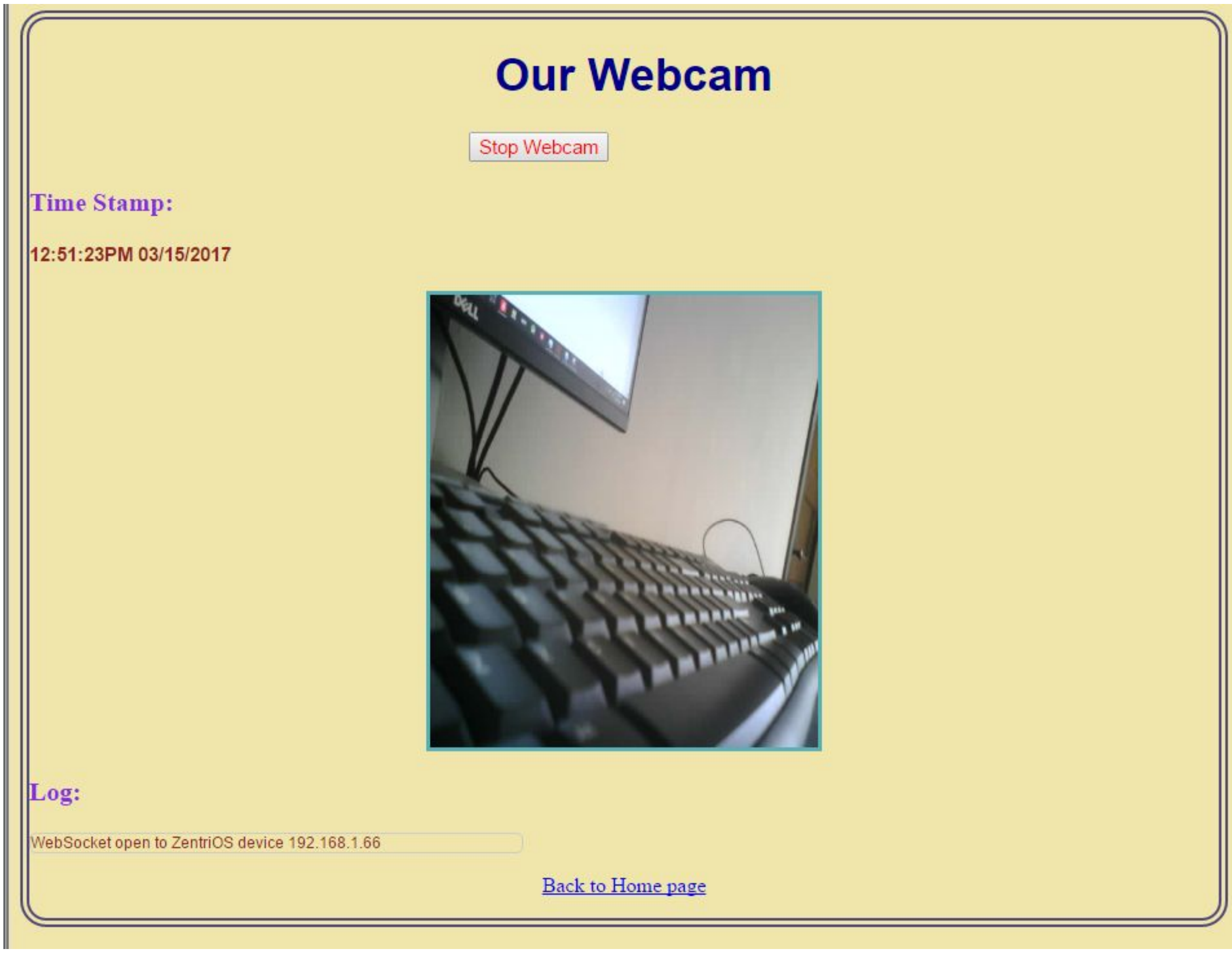



Figure 6: Webcam Page

One of the challenges faced while designing the webpage was that the IP address of the system is not the same for every Wi-Fi network it connects to. This challenge was overcome by using the variable *'window.location.hostname'* to denote the IP address of the system in our webcam script file.

## 2.4 3D CAD Design

We used Onshape CAD for design of our case. We designed the case for our own board. But after obtaining the board and finding the voltage regulator was not grounded properly, we decided to use the instructor's board. For the instructors board, we printed 4 iterations of the case because each time we had to make certain changes for the board to fit perfectly. Figure 7 shows the CAD design of the case. After design of case, we used Z-suite application for creating a .zcode file to print in the 3-D printer. The printed case is shown in Figure 8.

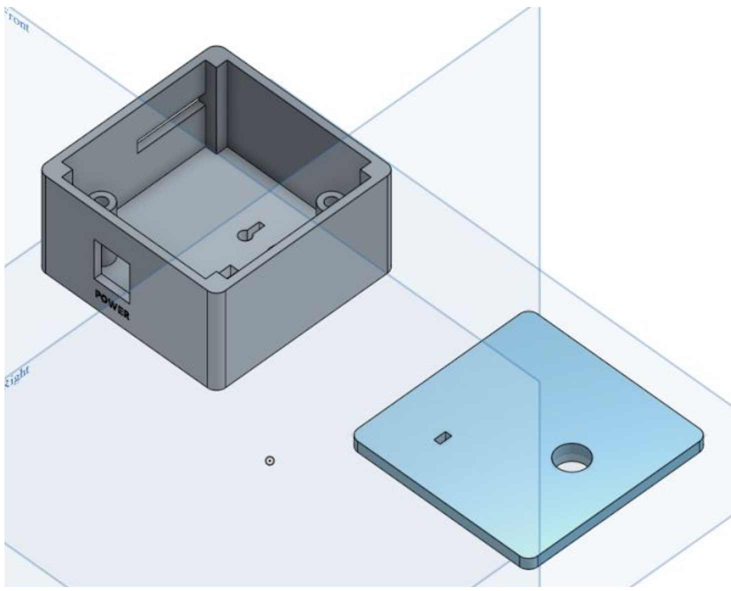
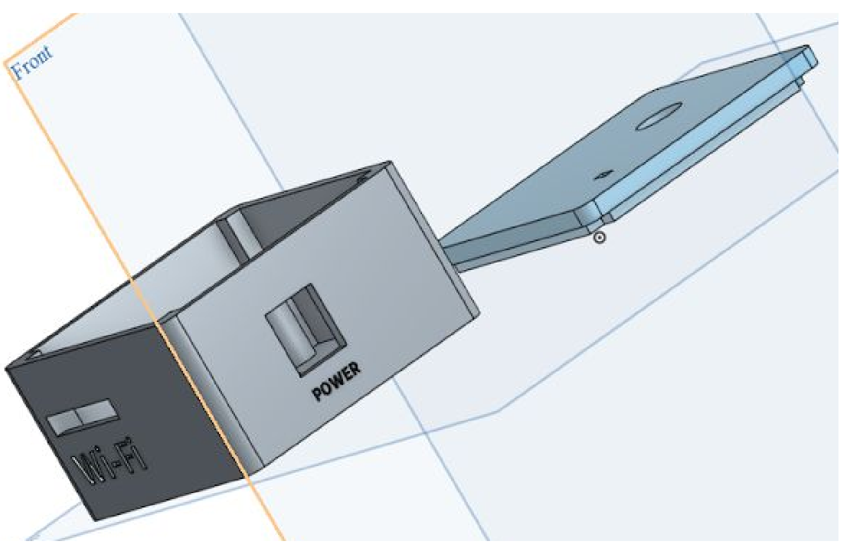



Figure 7: CAD Design



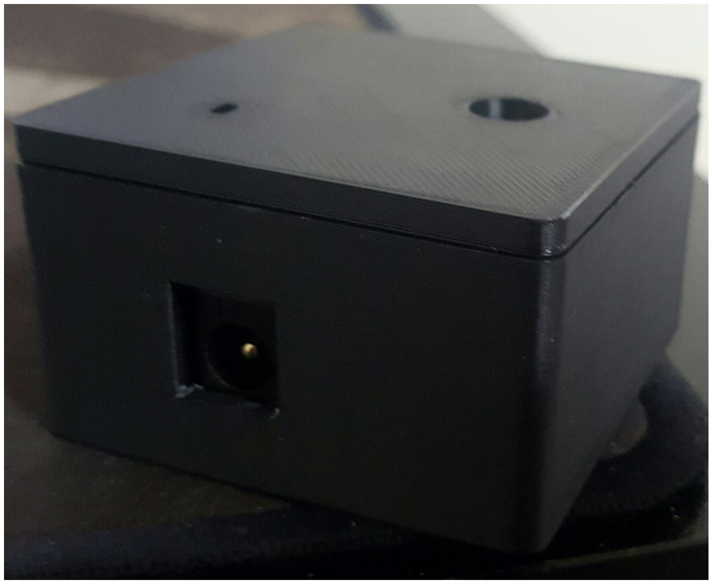


Figure 8: 3-D Printed Case

The main challenge for designing the 3D printed case was the proper placement of the holes for power jack, Wi-Fi antenna, LED's and buttons. They were always offset by a few millimeters. Another challenge was determining the height of the case so that the screws fit exactly into the printed holes of the case and the PCB.

# 3.Learning

I took this class as the description mentioned on the course website made me more intrigued about embedded system and it's design. I hoped to learn embedded system design and programming. I learned a lot more in all the different sections from design to full implementation of the circuit along with development of website and 3-D printing. I hoped I would also learn android programming by controlling the webcam from phone. The class structure is the best but if more emphasis is given to java-script would be better for non - computer science students. The workload was manageable but during the week for submission of program and design of board, the workload was higher as programming the controller took a bit longer than expected.

# 4. Conclusion

The project was a thoughtful concept. It's a good idea to design a portable webcam which can be used as a security camera. The last 3 lectures were pretty good as we learned a lot about the different aspects after design and manufacturing. My overall experience was pretty good and i would recommend to other students as well. I feel this course makes you more comfortable to work in group and builds your team building aspect.