ECE 445

SENIOR DESIGN LABORATORY

DESIGN DOCUMENT

Digitally-controlled LED Rotating Display System

Team #10

CHENTAI YUAN
(chentai2@illinois.edu)
GUANSHUJIE FU
(gf9@illinois.edu)
KEYI SHEN
(keyis2@illinois.edu)
YICHI JIN
(yichij2@illinois.edu)

Supervisor: Prof. Chushan Li
TA: Adeel Ahmed

March 22, 2023

Contents

1 Introduction

1.1 Problem and Solution

By applying Persistence of Vision (POV), many spinning LED message systems are developed to display messages and images with few LEDs [1]. A higher frequency in displaying an image enhances the resolution of the image [2] and when the frequency is over 30 frames per seconds (FPS) [3], human can observe images with a great quality. With a high speed motor, many existing systems can display messages clearly.

However, there are some common limitations to be improved. First, many of them only support limited patterns of text messages [1] and the images or messages to be displayed are pre-defined and cannot be changed in a real-time manner during the display. Second, the wired connection between some components may limit the rotation of LED array, and reduce the quality of display. Some economical wireless communication technologies [4] and recent advances in component connections can be applied to achieve a better display and real-time image update.

We aim at developing a digitally-controlled LED rotating display system. In this system, a FPGA board would be the core component which handles the data input and output, processes image data, and communicates with other components to work cooperatively. Compared with other popular micro-controller platforms, FPGA has more advantages in performing such functions. It easily handles complex calculations with more powerful computing resources and parallel computation among different computing units and it provides more flexibility for developers.

Specifically, the FPGA board receives users' input from keyboard or camera to update messages or images in a real-time manner. A DC motor is controlled by PWM signal from FPGA to drive the stick with one row of LEDs to do 360° circular rotation in a high frequency. The central FPGA board also communicate with a micro-controller by Bluetooth to control the status of LEDs. Moreover, the connection between LEDs, motor and other components should be simple but firm enough to suppose good display and high-speed rotation.

1.2 Visual Aid

As shown in the figure 1 below, user can use input peripherals connected to the FPGA board such as keyboard and camera to transmit text or image data to FPGA board. Then the display subsystem is able to display the input text or image with rotation of LEDs. At the same time, the expected displayed pattern is also displayed on a VGA monitor for checking. Ideally, by connecting to a camera with video feature, the system can achieve real-time video stream display.

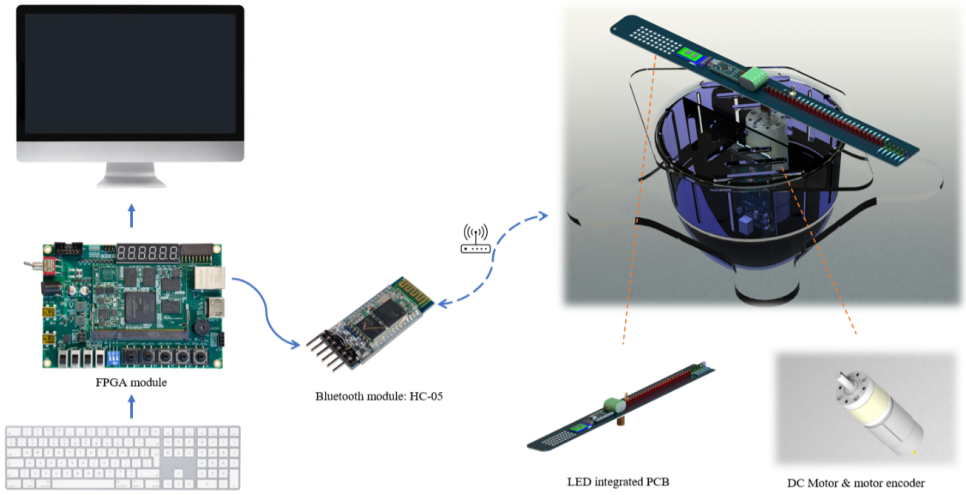


Figure 1: Visual Aid

1.3 High-level Requirements

Compared to other existing LED spinning LED message systems, our system has three highlight components: fancy LED array, advanced image processing technology and high-speed wireless communication.

- The display system can present output with an sufficient luminance and high fidelity for both the color and shape despite the possible distortion from rotation in indoor environment.
- The wireless communication between FPGA board and microcontroller can transmit control signal for LEDs with the speed over 18 Mbps and low latency under 1 second continuously and stably.
- The image processing algorithm that transform the Cartesian coordination-based image data to polar coordination-based image data should guarantee the precision of angle calculation. Also, the overall process should have low latency smaller than 30ms to optimize the delay and guarantee real time video display.

2 Design

This sections contains design diagrams, subsystem descriptions and requirements & verifications.

2.1 Diagrams

2.1.1 Physical diagram

The figure 2 below refers to the basic structure design of CAD models. The upper left sub-figure shows the motor, which is powered by a 12V power supply. The upper right sub-figure shows the LED integrated PCB board. All these components will be fixed and covered by an outer shield, which is shown on the lower sub-figure.

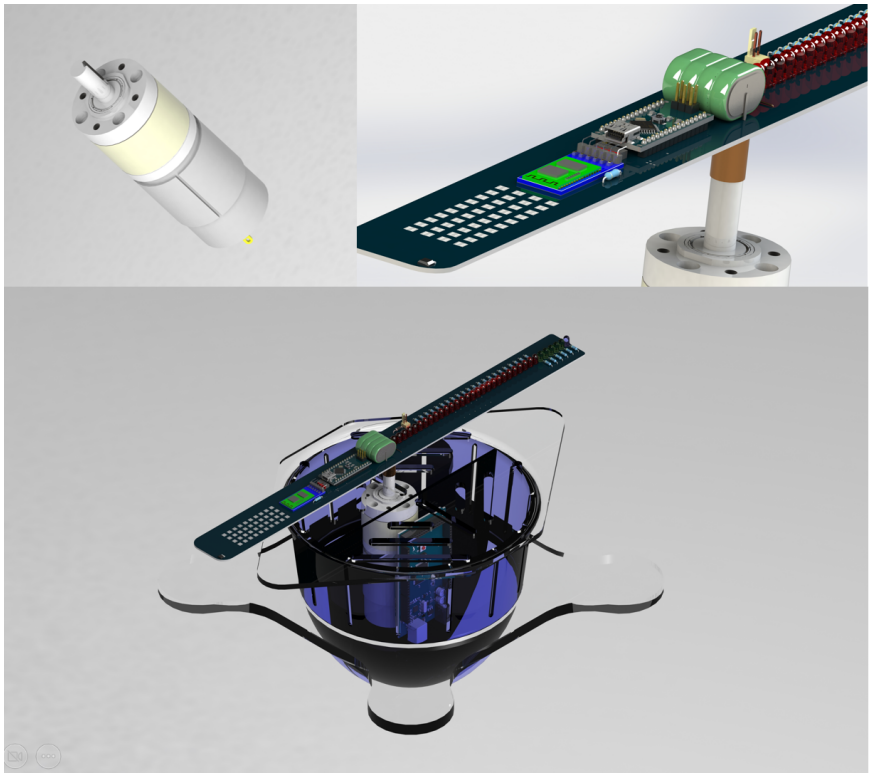


Figure 2: Design Overview

In our design, the goal is to build an integrated, compacted, and wireless LED display system, which is able to realize a real-time communication with FPGA board. Thus, we set a concept CAD model 3 with the total volume in 25,000 cm^3. The LED array is able to display a real-time image in the circle with a radius of 16 cm. The outer shield is assembled by 3D printed parts, which can give protection to motor. With a reasonable arrangement of the interior space of the shield, it will contain a 385 DC motor and a 12V power supplier.

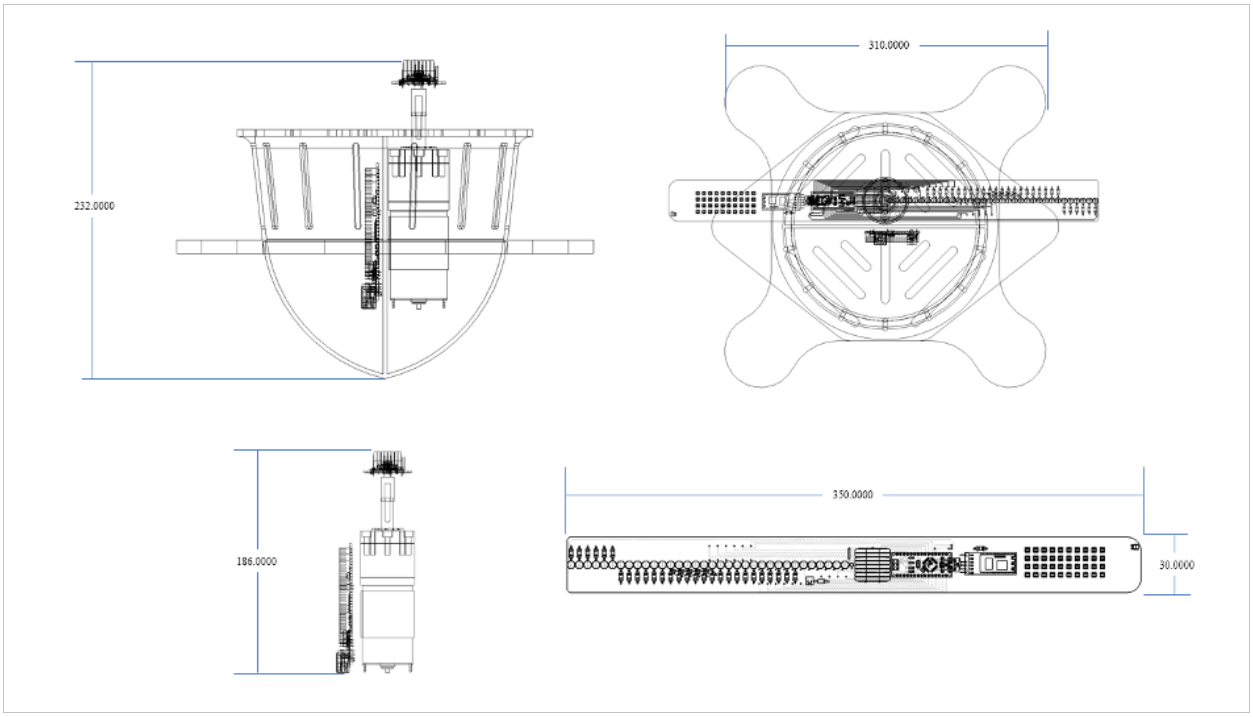


Figure 3: Design Dimensions

2.1.2 Block Diagram

The block diagram of the system is shown in Figure4. There are 3 subsystems with multiple components. We will introduce them and their connections in the next subsection.

2.2 Subsystem Descriptions and Components

This subsection gives the description of all subsystem functions and interaction with other subsystems

2.2.1 Display and Control Subsystem

The Display and Control Subsystem carries out the function of showing the image and video on the surface of rotating PCB board. The Display and Control Subsystem consists of a 385 motor, flanged joint with an integrated PCB. 160 LEDs distributed on the center line of PCB. With the rotation of motor, the LED array can show the image with 30 frame per second (FPS), forming a disk which has the diameter of 170 mm. All the LEDs are controlled by our main micro controller, ESP32-WROOM-32E, which attached at the center on the back of PCB. Powered by +3.3V voltage input through a voltage stabilizer from wireless charging module, ESP32-WROOM-32E can communicate with Logic and Interface Subsystem through Bluetooth, realizing the timely-control.

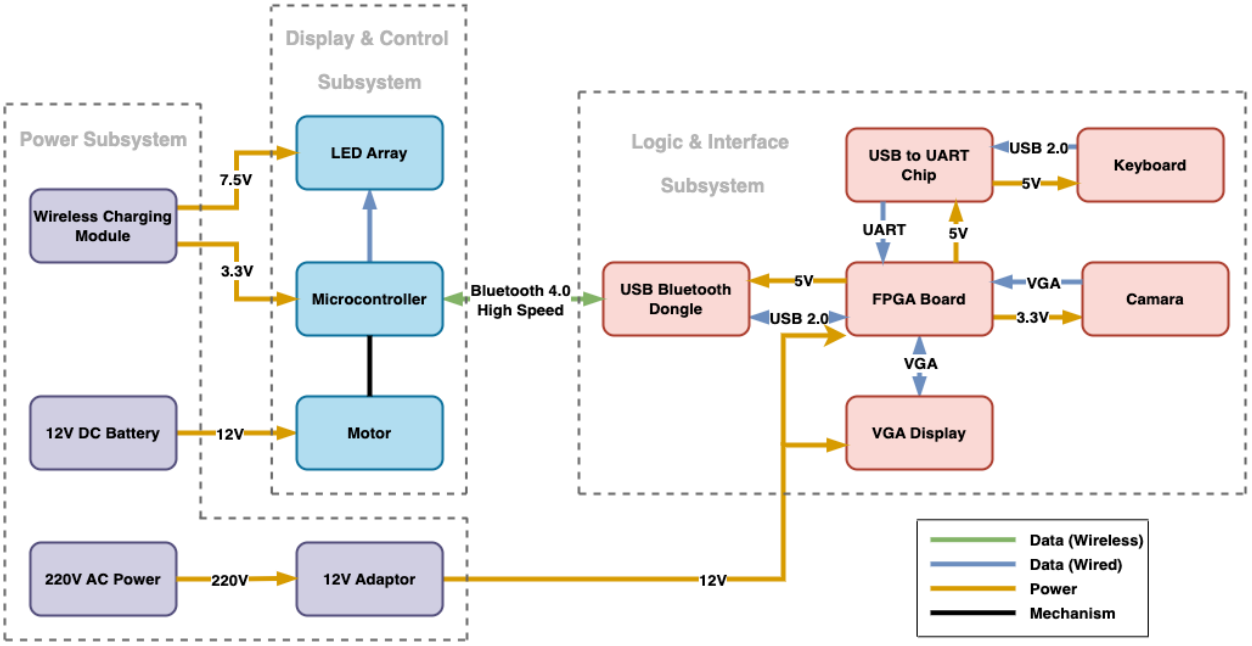


Figure 4: Block Diagram

2.2.2 Logic and Interface Subsystem

Logic and Interface Subsystem receives users' input from keyboard and camera, process the input text, image and video with low latency, send control signals to ESP32 micorcontroller with high speed Bluetooth communication, and offers Graphical User Interface for visualization. Since the communication with Display and Control Subsystem is wireless, it is physically independent with Display and Control Subsystem. Besides, it receive the power from the Power Subsystem with wired connection.

2.2.3 Power Subsystem

Power Subsystem supplies power to multiple components in other subsystems. There is a wireless charging module which supply 7.5V power to ESP32 microcontroller and LED Array with a pull-up resistor. More information about wireless charging module will be introduced in next subsection. The 12V DC Battery supplies power to the DC motor. Besides, a 220V AC power and 12V Adapter supply power to the FPGA board and VGA monitor. In short, different power supplies are independent and form the power subsystem logically.

2.3 Subsystem Requirements and Verifications

2.3.1 Display and Control Subsystem

Microcontroller
We use ESP32-WROOM-32E with 4 MB SPI flash as the microcontroller controls LED array on PCB. It is supplied by 3.3V DC power and can receive TransFlash (TF) card data and Bluetooth signal input. Running the pre-burned program, ESP32 converts input data to control signal for LED array with a clock signal.

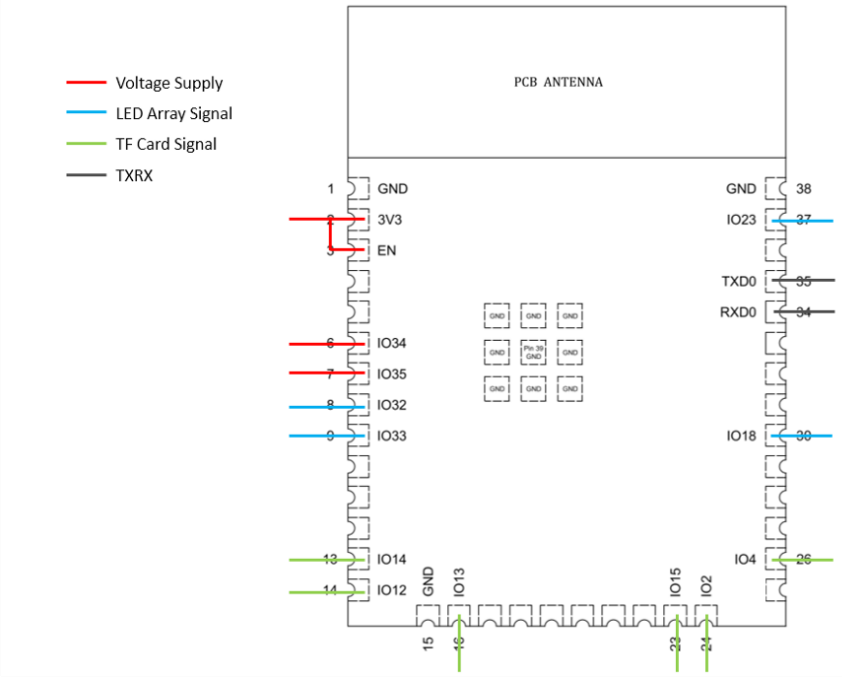


Figure 5: ESP32-WROOM-32E Pin Layout

In order to have function of LED array control (high frequency crystal oscillator output), Bluetooth, TF card reading and a small size, we choose ESP32 as our microcontroller. The ESP32-WROOM-32E microcontroller will be powered by a +3.3V source through a voltage stabilizer (AMS1117-3.3). The voltage source is a wireless voltage supply, which will be introduced in the part of Power supply. Refer to Datasheet, pin layout and connection to other components are shown on figure 5 and table 1. Microcontroller is used to professing all data input (from FPGA through Bluetooth module communication, and from TF card data) and output the data for LED array. The 520 KB SRAM need to contain all the instruction data, which will be pre-burned. Physical dimensions of ESP32-WROOM-32E on PCB layout is 18.0 mm $\times$ 25.5 mm $\times$ 3.1 mm.

LED array
Our LED array is made up of 160 square LC8822-2020 LEDs, whose diagram is shown

Table 1: ESP32-WROOM-32E Pin connection

Pin	Function	Connection
2, 3	Voltage input, keep High on Pin3 to enable the chip	With voltage stabilizer (AMS1117-3.3)
6, 7	RTC_GPIO, awaken chip from Deep-sleep mode	VCC, through a Hall switch (SOT23-3)
8, 9 & 30, 37	CI/DI, as clock/data input for LED array	With LED array (Left and right)
13, 14, 16, 23, 24, 26	Receive data from TF card	Through TF card reader (HYC77-TF09-200) connect with TF card
34, 35	TX RX, receive and transmit data	Through programmer writer connect with computer
Built-in Bluetooth	Receive and transmit data with IEEE 802.11 wireless standard	FPGA Bluetooth module

in figure 6. The LEDs have an integrated circuit inside and with a single sequency of data signal we can control a serial of LEDs, this greatly reduce the complexity of our PCB configuration 2. LED array supplied by 7.5V DC Power and controlled by ESP32-WROOM-32E through pins 8, 9 and 30, 37. As it is shown in figure 7, for each LED chip, data and clock of input pins is provided by microcontroller, while the data and clock output pins are connected with the next LED input.

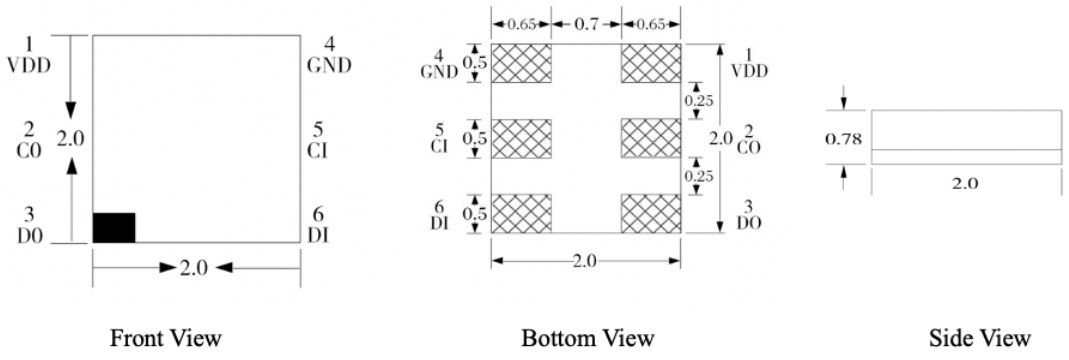


Figure 6: Three-view drawing of LED

A series of LEDs will be a connected in a line and their pin connection is shown is figure 8. To control LED array, our input data signal is arrange in a way shown in figure 9. For a single LED, the control signal is formatted as 24-bit RGB, 5-bit brightness (D[0:4]) and 3-bit padding. For the whole LED array, the control signal starts with 32-bit starting ”0”s,

Table 2: Pin Configuration of LED

Pin	Function	Connection
VDD	Supply Voltage	VDD input
CO	Clock output	Next LED's CI
DO	Data output	Next LED's DI
GND	Ground	Common ground
CI	Clock input	From clock signal/previous LED's CO
DI	Data input	From input signal/Previous LED's DO

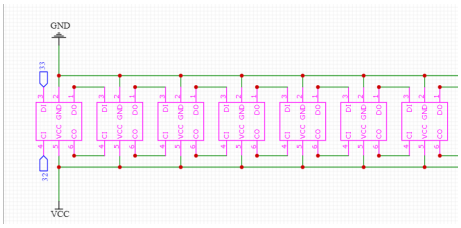


Figure 7: PCB Sketch of LED Array

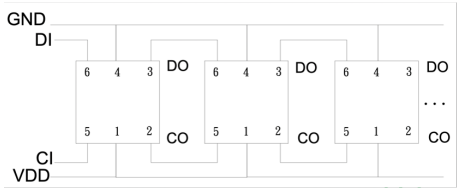


Figure 8: LEDs layouts

follows with 160 32-bit control signal for every LED and end with 32-bit ending ”1”s.

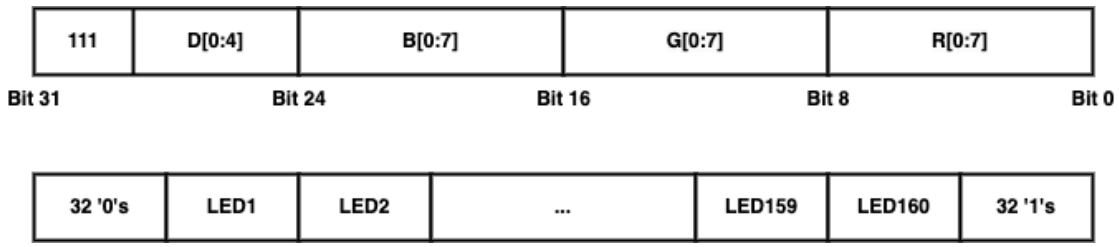


Figure 9: Signal format for a single LED and a serial of LEDs

When all the 160 LEDs shine on and give white light at the highest intensity the power consumption will be

$$160 \times 7.5V \times 18mA = 21.6W.$$

The control signal will consume

$$160 \times 5V \times 0.9mA = 0.72W.$$

Considering power dissipation for all the 2 serials of LEDs which is

$$2 \times 350mW = 0.7W,$$

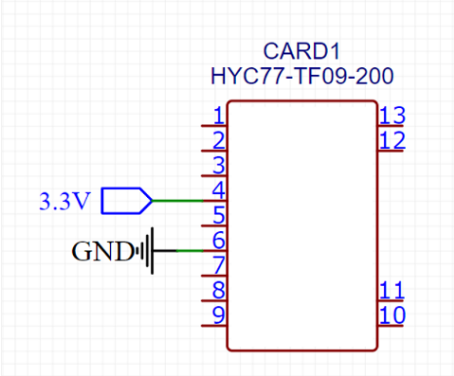
LED module will consume power up to

$$21.6W + 0.72W + 0.7W = 23.02W$$

For a single LED state, we will need 32 bits of data. Our system will require 30×180 frames per second. The input will also include starting bits and the ending bits. The total transfer rate will be

$$30s^(- 1) \times 180 \times (160 + 2) \times 32 \times \frac{1}{8 \times 1024 \times 1024} Mb = 3.337 Mb/s.$$

TF card reader
The HYC77-TF09-200 TF card reader on PCB is supplied by 3.3V power, receieve TF card push and output TF card data.



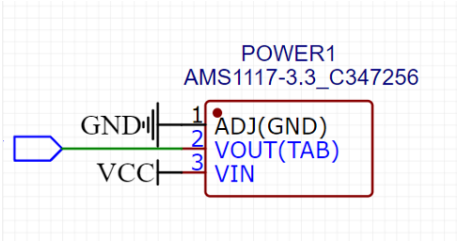


Figure 10: PCB Sketch of TF card reader Figure 11: PCB Sketch of Voltage stabilizer

HYC77-TF09-200 is a common TF card reader attached on PCB. It can realize the function of reading data stored in TF card. We will transfer these data to microcontroller (ESP32-WROOM32E). Shown on figure 10, HYC77-TF09-200 will be powered by a +3.3V source through voltage stabilizer (AMS1117-3.3). The pin connection table is shown in table 3.

Voltage Stabilizer
The AMS1117-3.3 Voltage Stabilizer on PCB accepts unstable power supply in range of 3V and +8V from the wireless voltage supply module which will be introduced in power subsystem, then output a 7.5V stable power supply. The PCB sketch is shown in firgure 11

Switch/Connector
A self-locking switch is placed as a cutting off for the +12V voltage source. Detailed function for connectors is shown in table 4 below.

Table 3: HYC77-TF09-200 Pin connection

Pin	Function	Connection
1	SD_DATA2	GPIO 12, ESP32-WROOM-32E Pin 14
2	SD_DATA3	GPIO 13, ESP32-WROOM-32E Pin 16
3	SD_CMD	GPIO 15, ESP32-WROOM-32E Pin 23
4	+3.3 V voltage input	/
5	SD_CLK	GPIO 14, ESP32-WROOM-32E Pin 13
6	GND	/
7	SD_DATA0	GPIO 2, ESP32-WROOM-32E Pin 24
8	SD_DATA1	GPIO 4, ESP32-WROOM-32E Pin 26

Table 4: Switch and Connector Functions

Device	Requirement	Function
LA22-A2 Switch	Size: 22mm diameter; Type: Self-locking; electric parameter: bearing current should larger than 700mA (motor rated current)	Turn ON/OFF the connection of +12 V voltage source. Connect with +12V Li-PO battery and motor.
HDR-M-2.54_1x2	Connector bearing +8 V voltage	Connect with ESP32 (for TX/RX).
HDR-F-2.54_1x2	Connector bearing +8 V voltage	Connect with LED array (for LED test).

LED Display Overall
The overall PCB design of the is shown in appendix C.

Motor
To achieve the image refresh rate of over 30 FPS, motor need to reach at least 1800 RPM (revolutions per minute) when carrying PCB. Then we choose RS-385A motor, which has 5000 RPM without load. We list some other parameters in table 5 and more detailed analysis is given in tolerance analysis 2.4.

The following two tables 6 and 7 show the requirements and verification procedures of display and control part.

Table 5: Motor Requirements

Function	Requirement
Physical dimensions	Shell diameter 29 mm. Output shaft diameter 2.3 mm, 11 mm length
Voltage supply	+12V DC
Rated current	550 mA
No load RPM	5000 RPM
Stall condition	Torque = 750 g·cm

Table 6: R&V Table of display part

Requirements	Verifications
Colors are bright and there's no obvious distortion (subjectively)	1. Turn off all the light in the room. Use test input signal provided (pure colors). Turn on the system and wait till the PCB is rotating constantly. 2. Use a photometer to measure the intensity and wavelength of the light 1m perpendicularly from the display system. 3. See if the difference of measured wavelength and the theoretical value is within 20nm.
The display is balanced, the whole surface is evenly luminous or light sources is evenly distributed.	1. Turn off all the light in the room. Use test input signal provided (white light at half the maximum intensity). Turn on the system and wait till the PCB is rotating constantly. 2. Turn on the photometer and. place the test probe every 42.5mm from the center (including center point and the edge) read and record the result. 3. Compare all the result we get from step 2, calculate if the maximum difference of intensity is within 10% of the average intensity.

2.3.2 Logic and Interface Subsystem

Video Capture with FPGA
To achieve the expected display quality of 30 fps, we set rotation speed of motor as 30 circles per second and except to use to a camera able to capture continuous images in about 30 fps. Considering a VGA monitor is used to display original and processed video,

Table 7: R&V Table of control part

Requirements	Verifications
The motor will output a rotation speed 1800 RPM under a 12V power supply and torque of 480 g·cm	1. The motor will connect with +12V power supply, and it will rotate with the speed of 5000 RPM. 2. An empty PCB will attach to the motor using flange. Applied a weight of 10 gram at the distance of 6 mm place on the PCB (It is a high-load test). 3. Use slow motion camera and record the rotation speed. It will rotate more than 30 circles per second.
The PCB have a displacement within 1.5 mm with the rotation speed of 1800 RPM	1. Set a horizontal line (Line1) at the back of the PCB, which horizontal to the PCB. Set a line (Line2) parallel to the horizontal line, but 1.5 mm lower. 2. Repeat step 1 and step 2 above. 3. Use slow-motion camera and capture the movement of the upper PCB. The maximum displacement point (the edge of PCB) will within the Line2, which shows the maximum displacement less than 1.5 mm.
The Hall switch (SOT23-3) changes the status under the influence of magnet. (Type for magnet is 2 mm height, 6mm diameter. Distance with hall switch is 20-40mm)	1. Connect hall switch with ESP32 chip on the firing tool, check the status is working for the chip. 2. Put the magnet at 40mm distance with the hall switch, the hall switch will not change the status. 3. Move the magnet 1 mm as a step, closer to the hall switch. Record the status for each step the switch will change the status at the distance of 28 ± 0.5 mm

the VGA-based camera would be the first choice. Then we get OV7670 which has an image array capable of operating at up to 30 fps in VGA with complete user control over image quality, formatting and output data transfer. The connection to DE2-115 board in shown in figure 12 and table 8 refer to [5].

It support a 640x480 VGA resolution and multiple output formats such as Raw RGB, RGB565, RGB555 and RGB444 which are sufficient and compatible with other components.

To reduce the overhead of image processing and data transmission, we simply use RGB444 format and keep the same when image processing and data transmission. In this case, the

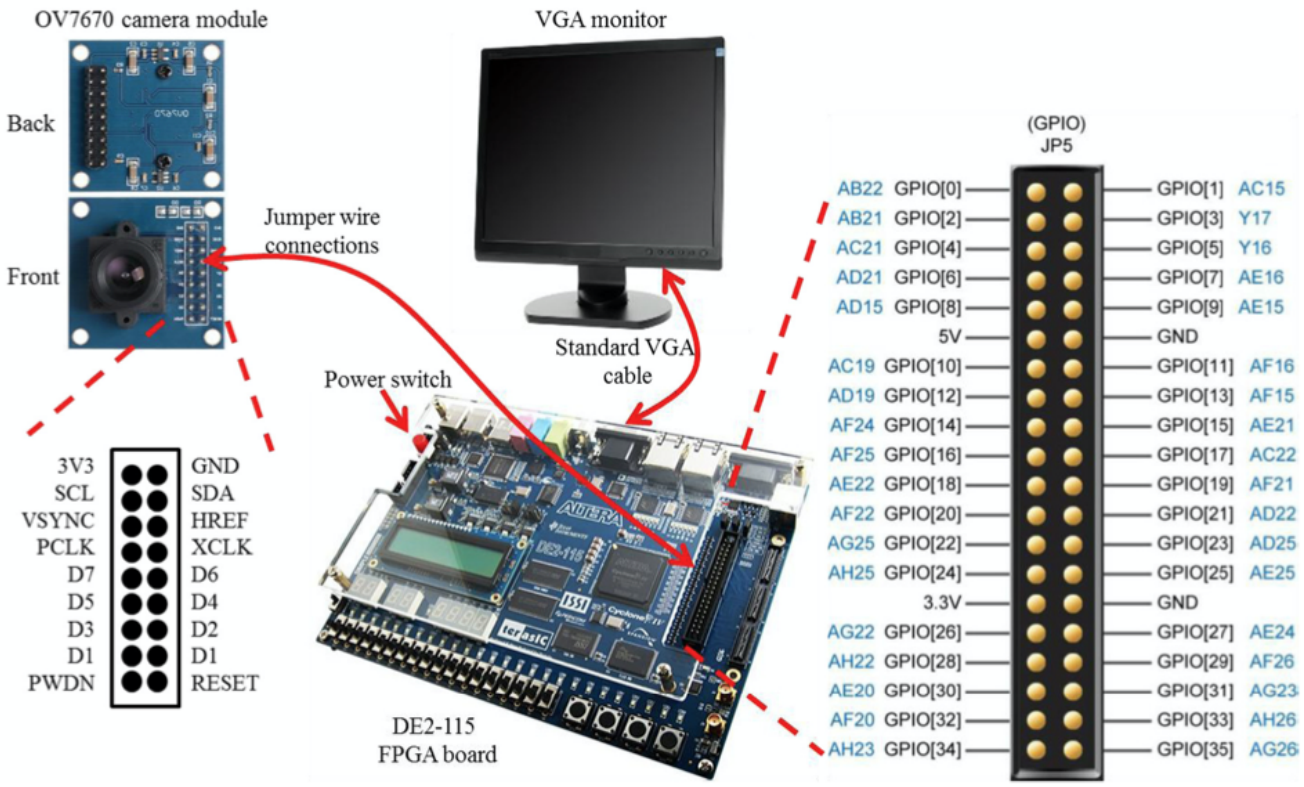


Figure 12: Connections between OV7670 and DE2-115 and VGA display

Table 8: OV7670 Pin connection to DE2-115

OV7670 Pin	DE2-115 Pin	OV7670 Pin	DE2-115 Pin
3V3	3.3V	GND	GND
SCL	AF24	SDA	AE21
RESET	AF15	VSYNC	AF25
HREF	AC22	PCLK	AC19
XCLK	AF16	PWDN	AD19
D7	AE16	D6	AD21
D5	Y16	D4	AC21
D3	Y17	D2	AB21
D1	AC15	D0	AB22

color of every LED is determined by a 12-bit RGB444 signal.

Image Storage on FPGA
The DE2-115 FPGA board contains multiple storage and memory components according to the Intel manual[6], and it's important to choose the appropriate one to store the images that will be processed by Verilog logic. As mentioned earlier, we'll be using RGB444 pixel formats for both video and image data in image processing and data transmission. Therefore, the selected component should have sufficient space and high read/write bandwidths. After considering the capacity and I/O bandwidth of various memory components, we have decided to use synchronous dynamic RAM (SDRAM) to store raw or pre-processed image data. According to the Intel DE2-115 Computer man-

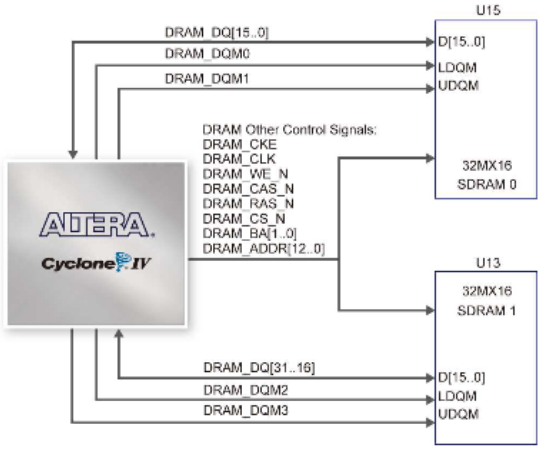


Figure 13: Connection between FPGA and SDRAM

ual[6], the DE2-115 has 128MB of SDRAM, which is made up of two 64MB SDRAM devices, each containing a 16-bit data line to the FPGA as shown in Figure13. If the SDRAM controller operates at a performance frequency of 100MHz, the theoretical bandwidth can be calculated as follows:

$$BW = 100MHz \times 2 \times 32bits/8/1024/1024 = 381.47MB/s$$

Let's consider the scenario when processing VGA stream data with a resolution of 640x480 and RGB444 pixel format. The data size per fps can be calculated as:

$$640 \times 480 \times 12bits/8/1024/1024 = 0.439MB/fps$$

Using SDRAM, we have enough space to buffer the video data and transfer more than 30 frames per second to the FPGA logic for image processing. With the help of SDRAM, the memory I/O bandwidth will not be a bottleneck during the entire process.

Image processing algorithm on FPGA
To display an image in rotating LEDs, the core of the image processing would be converting a Cartesian coordination-based image data into polar coordination-based image data. The basic philosophy of the algorithm is then the transformation from Cartesian coordination to polar coordination. As explain in figure 14 below.

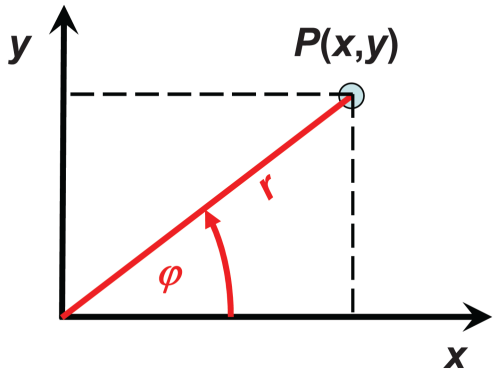


Figure 14: Coordination Transform

Suppose we have a pixel with position info (x, y), to turn it into a polarized position, the following formulas will be utilized:

$$r = \sqrt{x^2 + y^2}$$

$$\phi = \arcsin \frac{y}{r}$$

Regarding to above formulas, there are mainly 2 steps involved in the processing algorithm: calculating square root for r and perform $\arcsin$ operation for ϕ using Verilog logics. Thanks to the development of mathematics and computing, we already have a well developed algorithm for computing square roots on hardware logic. For the details of this part, please refer to appendix B. As for the $\arcsin$ computation, since Verilog does not directly support trigonometric functions, we will need an alternative approach to perform this. By applying Maclaurin expansion, we can write $\arcsin$ in another form:

$$\arcsin(x) = x + \frac{x^3}{6} + \frac{3x^5}{40} + \frac{5x^7}{112} + ... + \frac{1}{2^{2n}} \binom{2n}{n} \frac{x^{2n+1}}{2n+1}$$

In this case, we can calculate $\arcsin$ in a method feasible for Verilog logic though we will lose some precision. The lost of precision is mainly related to the number of terms we keep in the Maclaurin expansion. However, this kind of precision lost can be tolerated upon calculation and the detailed analysis will be discussed below.

Control signal to ESP32

Since data are usually transmit in multiple of 8 bits, we pad the signal to 16 bits by add exact 4 brightness bits to simplify transmission and get 16-level brightness. We expect it will achieve the best display quality compared to the format of 15-bit RGB555 + 1-bit brightness and 16-bit RGB565 when we fix the length of signal as 16 bits. So the unit control signal to ESP32 is arrange as in figure 15 Then the control signals for different LED are transmitted continuously. Recall we plan to refresh 160 LEDs totally 5400 times per LED per second and every LED is controlled by 16-bit signal. Then the expected transmission speed is up to

$$5,400 \times 160 \times 16 = 13.2 Mbps.$$

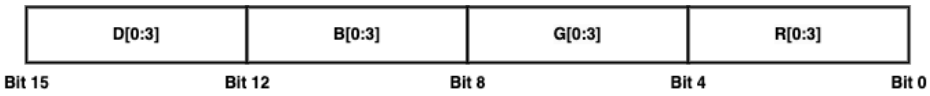


Figure 15: 16-bit unit control signal

But to follow the specific data transmission protocol, the actual transmission speed should be larger than 13.2 Mbps.

UART on FPGA Board
DE2-115 board offers a serial port that implements a UART (Universal Asynchronous Receiver/Transmitter) and is connected to an RS232 chip. And we can use that to implement Bluetooth communication with ESP32. About more details about UART, please refer to appendix A. In UART, Baud rate is a critical parameter to determine the maximum transmission speed. Baud rate is the rate at which the number of signal elements or changes to the signal occurs per second when it passes through a transmission medium. It is related to gross bit rate which is the number of binary bits (1s or 0s) transmitted per second.

$$Bit\ rate = Baud\ rate \times bits\ per\ signal$$

However, Baud rate has sometimes incorrectly been used to mean gross bit rate, since these rates are the same in old modems and the simplest digital communication links using only one bit per symbol. [7] In serial port, Baud rate is usually equal to the gross bit rate and then we can easily get the maximum transmission speed of UART.

Take existing UART implementation as an example, it is configured for 8-bit data, one stop bit, odd parity, and operates at a baud rate of 115,200. So the maximum transmission speed is

$$115,200/1024/1024 = 0.11Mbps.$$

Considering the whenever we transmit an 8-bit data, two extra bits are used as stop bit and parity. The valid maximum transmission speed would be

$$0.11/10 \times 8 = 0.088Mbps.$$

Due to the large gap between the transmission speed we need and maximum speed supported by UART, we are unable to implement Bluetooth with UART to transmit control signals to ESP32.

Alternatively, DE2-115 board offers a USB port which support USB 2.0 protocol. It can achieve a transmission speed of about 30 MBps and meets our requirement pretty well so that we can extend control signals to ESP32 if necessary.

$$30MBps/13.2Mbps = 18.2$$

And we can use official accessories USB Bluetooth Dongle or USB WiFi Dongle to implement Bluetooth or WiFi communication. USB Bluetooth Dongle supports Bluetooth 4.0 + EDR with transmission speed up to 24 Mbps and transmission distance up to 50 meters.

USB WiFi Dongle supports 802.11b/g/n WiFi Transmission Protocol with transmission speed up to 150 Mbps. So USB WiFi Dongle has a larger maximum transmission speed but might be limited by practical WiFi connection. Ideally, speed of 24 Mbps with Bluetooth 4.0 is sufficient for transmitting continues control signals to ESP32 even we consider protocol overhead and practical performance reduction.

$$24Mbps/13.2Mbps = 1.82$$

We can try to use Bluetooth communication first, if it does not fit the requirement, then we use WiFi communication to supply larger possible transmission speed.

And on the ESP32 side, it integrates Bluetooth module with Bluetooth 4.2 BR/EDR and Bluetooth LE specification which is backward compatible with Bluetooth 4.0 and WiFi module with 802.11b/g/n. Both two alternatives are available and can work theoretically.

Keyboard input with FPGA
Since we use USB port to support Bluetooth communication with ESP32, we do not have extra USB port in DE2-115 to connect to keyboard directly. Alternatively, we will use a USB to UART chip to support communication with USB keyboard via the serial port on DE2-115. Actually we can also use Bluetooth module and Bluetooth keyboard to build connection. But we find that Bluetooth keyboards are much more expensive than USB keyboards.

One of the of keyboard we noticed is the polling rate which is the frequency of how often the keyboard sends data. We can understand the polling rate in two ways. First, it defines the maximum number of key presses it can detect and send. Second, the approximate response time or delay of keyboard can be simply calculate as

$$T_{response} = \frac{1}{R_{polling}}$$

Common keyboards currently have polling rate of 125 Hz, 1000 Hz and even larger. Take 125 Hz as an example, the keyboard can detect up to 125 key presses per second which much large than a person can do and have a delay up to 8 ms which is pretty short to observe and does matter in our system. The polling rate of 125 Hz is sufficient to be used.

We aim to accept all ASCII code characters as input. Since ASCII is an 8-bit code, the USB-UART communication should be able to transmit up to 1000 Bits valid payload so that it can handle input in real-time manner. That is, the lower bound of transmission speed of USB and UART should be larger than about 1 Kbps with a range of tolerance.

$$125 \times 8 = 1000Bits$$

We know USB 2.0 supports up to 30 MBps raw data speed and built-in UART implementation on DE2-115 supports about 0.088 Mbps (88 Kbps) valid data speed. Both of them are much larger than 1 Kbps. And considering there is only one built-in serial port as

USB port, we can leave the built-in serial port for future use like enabling closed-loop control of motor and implement a simpler UART with a lower baud rate to connect with keyboard.

To implement UART on FPGA, we need to program three modules, which are Receiver, Transmitter and Baud generator modules as shown in figure 16.

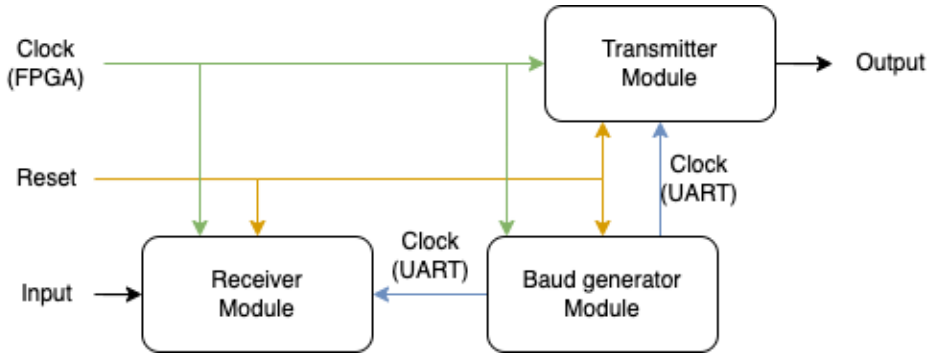


Figure 16: UART module

Since the clock frequency of FPGA is asynchronous with the sample clock frequency in UART, the Baud generator module is used to generate a clock in Receiver and Transmitter modules. The sample clock is related to the specified baud rate and should be 16 times of the baud rate. A With the classic baud rate of 9600 Hz, UART can transmit vaild data in the speed of

$$9600 \times \frac{8}{10}/1024 = 7.5Kbps$$

and sample clock frequency is

$$9600 \times 16 = 153,600Hz.$$

The frequency of FPGA clock is 50 MHz. So Baud generator module should generate a pulse whenever it receives 325 clock pulses.

$$50M/153,600 = 325.5$$

Successful extraction of the data from a frame requires that, over 10.5 bit periods, the drift of the receiver clock relative to the transmitter clock be less than 0.5 bit periods in order to correctly detect the stop bit. One bit period counts

$$325.5 \times 16 = 5208\ pulses.$$

Then the number of pulses in 10.5 bit periods is

$$5208 \times 10.5 = 54684.$$

In the implemented 10.5 bit periods, the number of pulses is

$$325 \times 16 \times 10.5 = 54600.$$

The drift is

$$54684 - 54600 = 84\ pulses,$$

which corresponds to

$$\frac{84}{5208} = 0.016\ bit\ periods$$

Hence, the tolerance from 325 and 325.5 is small enough for the Receiver and Transmitter modules to receive and transmit correct data.

Receiver module has Finite State Machine (FSM) with IDLE and READ states. It starts in IDLE state and samples input data in clock frequency given by Baud generator. When there is a high-low transition in input, it converts to READ state and waits 24 clock period (1.5 bit period) to sample the center of the first bits of data. Now Receiver samples data at bit-period intervals (16 clock periods) until it has read the remaining 7 data bits and the stop bit, then returns to IDLE state and repeats.

Transmitter module works in a similar way. There are IDLE and WRITE states in FSM. It starts in IDLE state and samples the signal (not shown in figure 16). When it detects data to be send, it converts to WRITE state and sends data at bit-period intervals. It also works in the center of bit, so there is a short delay of 8 clock periods.

Graphical User Interface design
We offer a simple GUI on standard VGA monitor to visualize input. Since it is not the critical part of our system, we only give conceptual scheme here and will refine and implement it later. GUI has four pages. The main page 17 shows three supported modes, text, image, video. User can choose any one to enter the specific mode. In the text mode 18, user can type string in the text-box and use ”Enter” key to send the string and start display. In the image mode 19, GUI shows some pre-stored images and choose one to display. In the video mode 20, it shows two streaming videos, one is the original video captured by OV7670 camera and the another one is the processed video by FPGA.

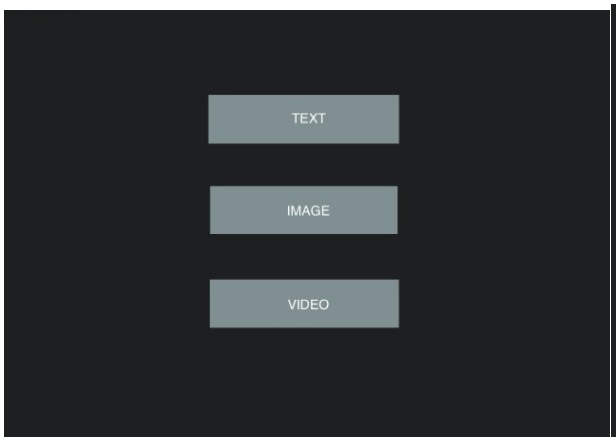


Figure 17: PCB Sketch of TF card reader

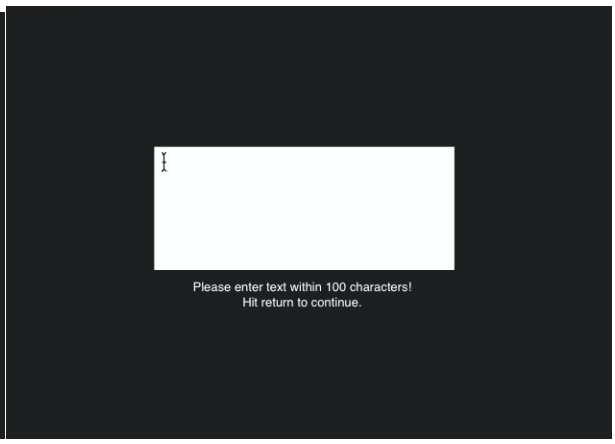


Figure 18: GUI text mode

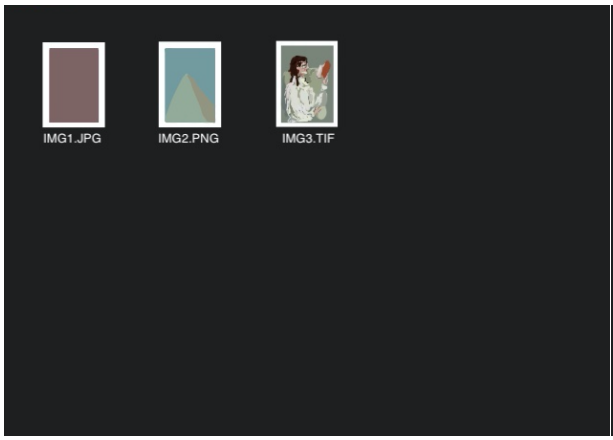


Figure 19: GUI image mode

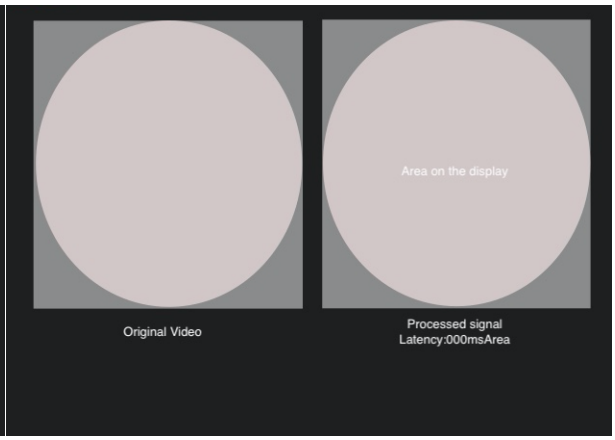


Figure 20: GUI video mode

The following table 9 shows the requirements and verification procedures of Logic and Interface Subsystem. We use test programs to verify requirements. They should be created by us during development.

2.3.3 Power Subsystem

Since other components are simple and easy to meet requirements of power supply, here we only introduce the wireless charging module.

Wireless Charging Module

The wireless charging module is used for transfer power from the base part to the upper PCB that carry voltage stabilizer, signal receiving module and the LED array. It will be located at the rotation axis of the base motor part and PCB. The voltage stabilizer requires a current input larger than 700mA. The wireless will supply a current larger than 1000mA in 500ms once it is on. Detailed function for connectors is shown in figure 21. The power

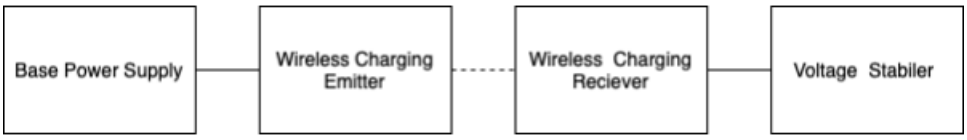


Figure 21: Wireless Charging Connection

is transferred from electricity to magnetic field, and then to electricity again, part of the power is dissipated at a rate of 16.7%.

The emitter part will transfer the input DC power to an AC power to its coil and a magnetic field is produced. In the following formula N corresponds to the number of turns of the coil and r is the distance where we measure from the coil that produce the field.

$$B = \frac{N\mu_0 I}{2\pi r}$$

Table 9: R&V Table of Logic and Interface Subsystem

Requirements	Verifications
The transmission speed of USB and Bluetooth is larger than 18 Mbps under 25 % tolerance2.4 to support uninterrupted control to LEDs.	1. Reset or activate DE2-115 board and ESP32. 2. Connect USB Bluetooth Dongle to Pin U28 (USB port) on DE2-115 board. 3. Build Bluetooth connection between Dongle and ESP32 by push Button 3 on DE2-115, if connection succeeds, LED 3 on DE2-115 will be on. 4. Run the Bluetooth test program on ESP32 and wait for input. 5. Turn on Switch 15 on DE2-115, then it will send a specific control signal with duration of 10 seconds to ESP32. To simulate the 25 % tolerance, it mandatorily adds 1 invalid data frame within every 4 data frames. 6. The test program will count the total number of valid bits received, calculate and display the bit rate of valid data and the LEDs should blink continuously.
The overall delay of image processing algorithm should be less than 30ms for a single frame to guarantee the continuous display of video stream.	1. Activate DE2-115 board. 2. Implement a clock counter module to record the number of cycles spend to perform image processing. 3. Execute the image data processing module together with the SDRAM read module. 4. Run the simulation testbench program that instantiate the above modules. Config the clock frequency manually in the testbench. The clock counter module will output the result and display on the console. We can then check if the clock cycles cost for a single frame processing.

The power is transfer back to electricity from the magnetic field follows the following formula:

$$E = n\frac{\Delta\Phi}{\Delta t}$$

The receiver will be connected to the voltage stabilizer and its pin configuration is shown in table 10.

Table 10: Connection for the wireless charging module

Device I/O	Requirement	Function
Input Vcc on the emitter	Input Anode. Supply 24V, 1A input to the emission part	Connected to a 24V, 1A power supply in the base part
Input GND on the emitter	Input cathode	Connected to the ground
Output high on the receiver	Output 7.5V, 3-4A	Connected to the voltage stabilizer high input
Output low on the receiver	Output low	Connected to the voltage stabilizer low input

The module we use includes an integrated protection circuit and a light will shine on if there is anything wrong with the emitter, but we still need to verify that the output is within the desired range when an input at the possible range is given. A 7.5V$\pm$ 0.5V, 3A $\pm$0.5A input is what our voltage stabilizer can withstand and a power of 28W input is required to supply to the PCB.

The following table 11 shows the requirements and verification procedures of Wireless Charging Module.

Table 11: R&V Table of Logic and Interface Subsystem

Requirements	Verifications
Input voltage 23V$\pm$1V Input current 1A $\pm$0.05A Output voltage 7.5V$\pm$0.5V Output current 3A $\pm$0.5A	*Verification process for input:* 1. Attach variable power supply to Vin. 2. Sweep from 10.0 V to 12.0 V and ensure input current remains within the desired value. 3. Ensure the emitter will not give any warning and work normally. Its temperature is below 70°C. *Verification process for output:* 1. Attach voltmeter and amperemeter to receiver output pin. 2. Ensure the output voltage and current is within the tolerance range

2.4 Tolerance Analysis

Regarding to high-level requirements we proposed, we prove functions of critical components through mathematical analysis or numerical simulation. As the visualized and critical part of our system, we do want to do a better visual effect analysis, but it is more subjective and harder to analyze mathematically and simulate. We will improve our analysis after finding more formal materials and do experiments after we get all components.

2.4.1 Visual effect analysis

To reach our final goal of displaying desired patterns, our display subsystem must possess a certain hardware prosperity, namely, an appropriated luminance and the ability of anti-distortion.

To be more specific about the luminance, our display should never be too bright to look at, nor it should be too dim to cause any loss of expressiveness. We have all the data about the luminous intensity. Despite that, the values are just theoretical numbers of a single LED. In a real case of application, our LEDs seldom shine on at the highest intensity continuously. The output of each color channel should be balanced. Considering a case that our display will show a serial fluctuation of pure colors, each channel will have an average output of half the maximum intensity.

Because our design uses a long- radial LED array, the center part will look much denser than the rear parts. We decided to adjust PWM for each LED so that their total luminous intensity can get proportional to their distance from the axis:

$$Pause\ width \propto Distance\ to\ \ rotational\ axis$$

The environment also affects the result a lot. A display of 1600 nit on cell phone can barely be seen clearly under a strong sunlight, but a laptop using 1000-nit-display is far too bright to be looked at at night even in a bright study room. A contrast of environment may also cause a distortion in color visually, i.e., a color becoming too cool or too warm. Some company has even introduced a real-time adjustment for color display, making displayed color consistent to human eyes. We considered making innovations in both hardware and software, but it is too costly.

Our display subsystem is made up of 160 LEDs, which have a maximum luminous intensity of 530mcd. The total luminous intensity will be apportioned by an area of $170^2\pi mm^2$. Considering an ideal model that all the light gives out the maximum intensity on average, we can get an approximate value in nit:

$$530 \times \frac{1}{1000} cd \times 1/2/(170^2 \times \pi \times 10^(-6) m^2) = 2.92 nit$$

The result is far too worse than the common displays we use in life, though it is already the best our power system could support. Human eyes can distinguish color at 1 nit and our system barely reaches this goal since there should be plenty of room left for adjustment PWM and balancing color. Nevertheless, the design is feasible. We will work on the current module but also update our design if it is allowed.

2.4.2 Motor load analysis

In this analysis, we want to figure out the maximum load for the motor when we set the system vertically and horizontally and keeping a 30 FPS display of images.

With parameter got from datasheet, we know that the motor has a rotation rate of 5000 RPM when no load. And it will be stalled under a torque of 750 g·cm. Based on the relationship between rotation rate and torque, we will have an idealized function:

$$N = -\frac{20}{3}\tau + 5000$$

As we are aimed at forming a display with image refresh rate ¿ 30 fps, which means,

$$N_{aim} \geq 30\frac{circle}{second} = 1800RPM$$

Constrains has shown below in figure 22. By substitution, we will have a maximum

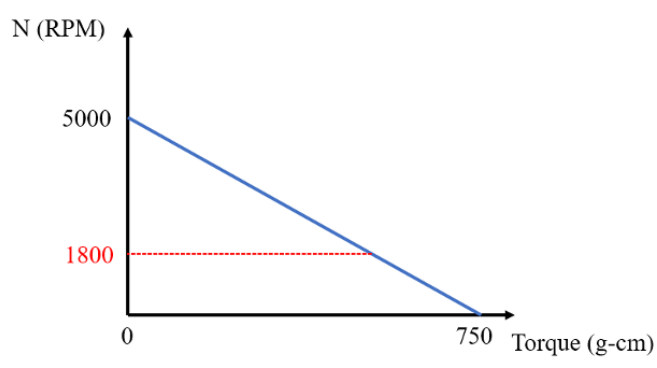


Figure 22: Motor N-Torque Constrains

torque shown below.

$$\tau_{aim} \leq 480g \cdot cm$$

When we place the system vertically, we can calculate the force on the output shaft of motor by the equation,

$$F_{shaft} - mg = F_{centripetal}$$

We have the equation for centripetal force,

$$F_{centripetal} = m \times w^2 \times r$$

where m is the equivalent weight for whole PCB, w is the angular velocity, and r is the mass offset on PCB.

The angular velocity w and rotation speed N have the relationship,

$$w = \frac{N \times \pi}{30}$$

With a minimum rotation speed 1800 RPM, we can calculate the relationship between mass offset on PCB and the PCB weight, shown below.

$$188.55^2[\frac{rad}{s}] \times r^2 + 9.81[\frac{N}{kg}] \times r - \frac{47[N \times m]}{m} \leq 0$$

Also shown on the figure 23 below, where m is PCB weight and r is mass offset on PCB. For example, we designed the overall PCB weight $m = 100g$, and by substitution, the

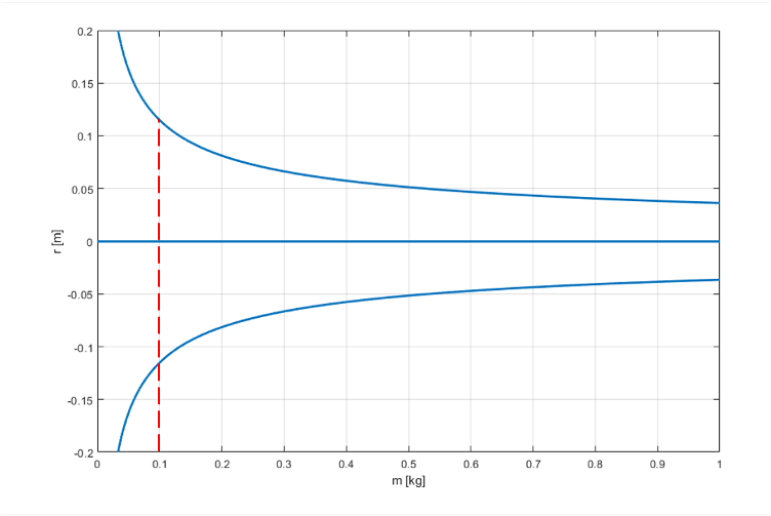


Figure 23: Relationship between mass offset and the PCB weight

mass offset on PCB can be calculated (Shown in red line on figure 23), which is,

$$-0.1151m \leq r_{offset} \leq 0.1148m$$

With a mechanical simulation, Finite Element Analysis is placed on our main PCB. During the rotation, the board will be bearing a torque of 480 g · cm for maximum, together with a pressure from all the components, which weight 30 g for maximum. The result is shown in figure 24 below. The maximum displacement at two edges of the PCB will within 1.5mm, which meets the requirement.

2.4.3 Bluetooth transmission speed analysis

We have calculated the expected transmission speed to control LEDs is about 13.2 Mbps. Here we analyze the format of data packet in Bluetooth 4.0. There are two forms of Bluetooth wireless technology systems: Basic Rate (BR) and Low Energy (LE). The Basic Rate system offers synchronous and asynchronous connections with data rates of 721.2 kbps for Basic Rate, 2.1 Mbps for Enhanced Data Rate and high speed operation up to 24 Mbps with the 802.11 AMP. Specifically, 802.11 AMP supports 802.11a, b, g Standards and 802.11g offers fastest maximum speed and good signal range. There are three types of 802.11 frames: management, control, and data. To verify the expected maximum speed of 18 Mbps does meet our requirement, we simply focus on the data frame. Refer to IEEE

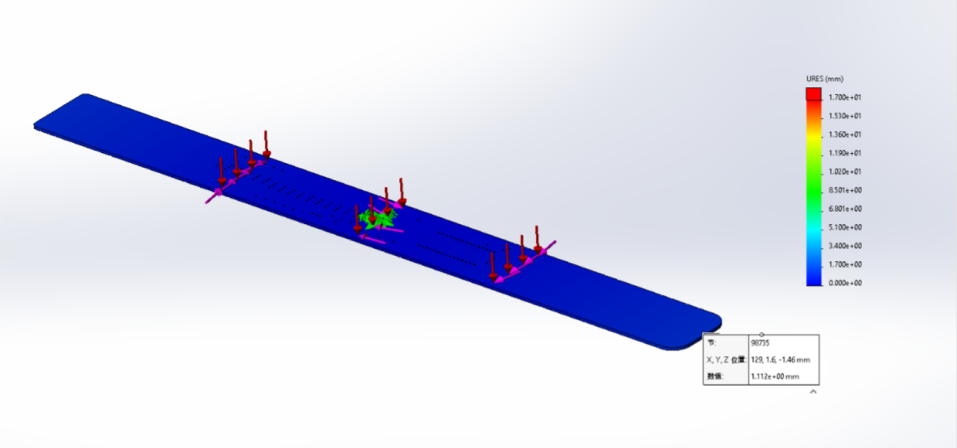


Figure 24: FEA Result for the main PCB

802.11-2007 Standard [8], the data frame is formatted in figure 25. The fixed length of data except payload in the data frame L_{fixed} is

$$2 + 2 + 6 + 6 + 6 + 2 + 6 + 2 + 4 = 36Bytes$$

Then the valid payload rate of one data frame is defined as

$$\frac{L_{body}}{L_{body} + L_{fixed}}$$

where $L_{body} \leq 2312$ is the length of frame body in Bytes. The maximum valid payload rate of one data frame is

$$\frac{2312}{2312 + 36} \times 100\% = 98.47\%$$

Then ideally the speed of

$$13.2/98.47\% = 13.41Mbps$$

and then the tolerance of limited overhead of management and control frames and potential transmission loss is

$$\frac{18 - 13.41}{18} \times 100\% = 25.5\%$$

And we still have large space to reach the theoretical maximum speed which is 24 Mbps.

$$\frac{24 - 18}{24} \times 100\% = 25\%$$

Since 18 Mbps is 25.5% larger than what we need and 25% smaller than theoretical maximum, it is achievable and fits critical overall feature.

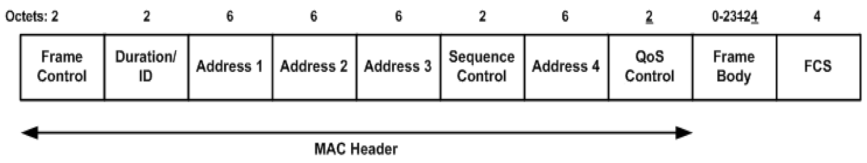


Figure 25: Data frame

2.4.4 Maclaurin Expansion Error Analysis

For calculation simplicity, suppose we have a image of size $m \times m$ after resizing to fit into our LED arrays and each pixel of the image will then be able to display on a LED. We set the pixel at $(\frac{m}{2}, \frac{m}{2})$ as the center pivot and convert all other pixels from Cartesian coordination to polar coordination. In this case, assume we use the first 5 terms of Maclaurin expansion of $\arcsin$ to calculate the estimated value, we can easily find the lost precision to be:

$$error = \Sigma_5^n (\frac{1}{2^{2n}} \binom{2n}{n} \frac{(\frac{y}{r})^{2n+1}}{2n+1}),\ -1 \leq \frac{y}{r} \leq 1$$

For the constant term in the expression, it is:

$$c(n) = \frac{1}{2^{2n}} \binom{2n}{n} \frac{1}{2n+1}$$

and $c(n)$ is monotonically decreasing respect to n. For $n = 5$, we can get $c(5) = 0.01065$. Also, for $n \geq 5$ and $-1 \leq x \leq 1$, x^{2n+1} will have a power larger than 11. Hence we can indicate that the maximum error of estimated value calculated by Maclaurin expansion is then less than 0.01, which is 0.572^o out of 360^o. Recall that in our analysis of the display system, it requires to change LED value every 2^o, we can then indicate that the use of Maclaurin expansion will not affect the correct display of images.

3 Cost & Schedule

3.1 Cost

Include a cost analysis of the project by following the outline below. Include a list of any non-standard parts, lab equipment, shop services, etc., which will be needed with an estimated cost for each. Labor: (For each partner in the project) Assume a reasonable salary ($/hour) x 2.5 x hours to complete = TOTAL Then total labor for all partners. It's a good idea to do some research into what a graduate from ECE at Illinois might typically make. Parts: Include a table listing all parts (description, manufacturer, part #, quantity and cost) and quoted machine shop labor hours that will be needed to complete the project. Sum of costs into a grand total

3.1.1 Labor

The estimated labor cost is listed in Labor Cost Analysis Table 12. We refer to the sample and past senior design projects to get the hourly salary and estimate total working hours based on our project workload and course credits.

Table 12: Labor Cost Analysis. The table shows the partner name in the project (Partner), the hourly salary of partner (Hourly Salary), the working hours in the project per person (Working Hours) and the total salary per person(Total).

Partner	Hourly Salary	Working Hours	Total
Chentai Yuan	$40	150	$\$40 \times 150 \times 2.5 = 15000$
Guanshujie FU	$40	150	$\$40 \times 150 \times 2.5 = 15000$
Keyi Shen	$40	150	$\$40 \times 150 \times 2.5 = 15000$
Yichi Jin	$40	150	$\$40 \times 150 \times 2.5 = 15000$
Sum			$60000

3.1.2 Parts

The estimated parts cost is listed in Parts Cost Analysis Table 13. The estimated cost is lower than $100.

3.1.3 Total cost

The total cost of labor and parts is

$$\$60000 + \$93.08 = \$60093.08.$$

Table 13: Parts Cost Analysis. The table shows the part name and short description (Description), the manufacturer of the part (Manufacturer), the vendor of the part (Vendor), the quantity needed in the project (Quantity), the unit cost of the part (Cost/unit) and the total cost of the the part (Total Cost).

Description	Manufacturer	Vendor	Quantity	Cost/unit	Total Cost
TF card reader (TF-SMD_HYC77-TF09-200)	Huayuchuang	LCSC	1	0.198	0.198
ESP32-WROOM-32E(4MB)	Espressif Systems	LCSC	1	7.48	7.48
HDR-F-2.54_1x2	LCSC	LCSC	2	0.0155	0.031
HDR-M-2.54_1x2	LCSC	LCSC	2	0.0262	0.0524
Voltage Stabilizer (AMS1117-3.3 SOT-89_L4.5-W2.5-P1.50-LS4.2-BR)	Youtai Semiconductor Co., Ltd.	LCSC	1	0.0731	0.0731
HALL (SOT23-3)	LCSC	LCSC	1	0.01	0.01
10k resister (R0603)	LCSC	LCSC	5	0.004	0.02
LED (LC8822 2020)	Jiufeng	LCSC	160	0.218	34.092
Switch (LA22-A2)	Zhengping	Zhengping	1	2.1	2.1
motor (motor 385)	YouxinDianzi	YouxinDianzi	1	3.8	3.8
PCB	LCSC	LCSC	1	15	15
USB to UART Chip CH9350L	WCH	Taobao	1	0.087	0.087
MK120 USB Keyboard and Mouse	Logitech	Jingdong	1	12	12
Camera OV7670	OMNIVISION	Yuanmeiguang	1	0.13	0.13
Wireless charging module 24V45C	Gosoon Electrons	Taobao	1	18	18
DE2-115 FPGA board	Terasic	ZJUI Lab	1	-	-
VGA montior	Unknown	ZJUI Lab	1	-	-
Total					93.08

3.2 Schedule

The weekly time-table is given in table 14 below.

Table 14: Weekly schedule table. The table shows the duration of weeks (Week) and indivival schedule for every group member (Chentai, Guanshujie, Keyi and Yichi).

Week	Chentai	Guanshujie	Keyi	Yichi
3/20-3/26	Order components	Explore image processing algorithm	Figure out communication protocol	Update visual effect design
3/27-4/2	solder PCB, test components	Implement SDRAM I/O module	Get parts, build connection	Test signal on the LED test series in static
4/3-4/9	Implement the Rainbow test code on ESP32, unit test on the LED array	Implement image processing algorithm	Implement UART, test Bluetooth	Test the visual effect in motion
4/10-4/16	Test the system with LED array and the motor	Test module functionality and processing output	Get expected transmission speed	improve display, implement GUI
4/17-4/23	Implement the outer shell design and render the CAD	Merge with other subsystem to test	Control signal test	Integrate display system with control part
4/24-4/20	Finish the outer shell assembly	Test and debug	Test and debug	Test and refine GUI
5/1-5/7	Integrate, finalize decoration	Integrate all	Integrate all	Integrate all
5/8-5/14	Mock Demo	Mock Demo	Mock Demo	Mock Demo
5/15-5/21	Prepare for Final	Prepare for Final	Prepare for Final	Prepare for Final
5/22-5/28	Final Demo & Report	Final Demo & Report	Final Demo & Report	Final Demo & Report

4 Ethics & Safety

4.1 Ethics

Regarding to the IEEE[9] and ACM Codes[10] of Ethics, we summarize the following ethical concerns that apply to this project:

1. Consider the impact of our project on society: We should have a responsibility to consider the impact of our work on society as a whole, including the social, economic, and environmental consequences. This requires taking a holistic view and considering not only the immediate benefits, but also the potential long-term consequences.
2. Uphold the principles of fairness and justice: We should strive to ensure that our work is fair and just, and that it does not discriminate against individuals or groups based on factors such as race, gender, religion, or nationality. This requires being aware of potential biases and taking steps to mitigate them.
3. Be honest and transparent: We should be honest and transparent in the professional conduct, including the communication with team members, teaching assistant, instructors and the public. This requires being forthright about potential risks and uncertainties, and disclosing any conflicts of interest.

4.2 Safety

Regarding to the safety guidelines provided in course website and our project development flow, we summarize the following safety concerns that should be considered:

1. Electrical safety: The power system involved in our project could pose risks of electrocution or electrical fires if proper safety measures are not taken. This includes ensuring that all wiring is properly insulated and grounded, that circuits are appropriately sized and protected, and that appropriate safety equipment is available for handling and testing electrical components.
2. Mechanical safety: The rotating motor involved in our project could pose risks of injury or damage to equipment if not properly installed or operated. This includes ensuring that the motor is securely mounted and that all moving parts are properly guarded to prevent contact with users or other objects.
3. Lab safety: The design and testing of our LED display system involves working with various tools, equipment, and materials that can pose hazards to our members. This includes ensuring that all of us are trained in proper safety procedures, that appropriate safety equipment is available, and that all testing and assembly is performed in a designated and controlled laboratory environment.

References

[1] L. Yong, M. F. Zakaria, and H. Hashim, "Design and development of spinning led message display system," Mar. 2012.

[2] H. Demirel and G. Anbarjafari, "Image resolution enhancement by using discrete and stationary wavelet decomposition," *IEEE Transactions on Image Processing*, vol. 20, no. 5, pp. 1458–1460, 2011. DOI: 10.1109/TIP.2010.2087767.

[3] T. Morris, P. Blenkhorn, and F. Zaidi, "Blink detection for real-time eye tracking," *Journal of Network and Computer Applications*, vol. 25, no. 2, pp. 129–143, 2002, ISSN: 1084-8045. DOI: https://doi.org/10.1006/jnca.2002.0130. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S108480450290130X.

[4] A. SARODE, D. SALUNKE, A. SHUKLA, and M. SONAWADEKAR, "Spinning led display using radio frequency," *International Journal of Computer Science and Informatics*, pp. 280–283, Apr. 2014. DOI: 10.47893/IJCSI.2014.1160.

[5] P. Irgens, C. Bader, T. Lé, D. Saxena, and C. Ababei, "An efficient and cost effective fpga based implementation of the viola-jones face detection algorithm," *HardwareX*, vol. 1, pp. 68–75, 2017.

[6] Intel. ""DE2-115 Computer Manual"." (2017), [Online]. Available: https://ftp.intel.com/Public/Pub/fpgaup/pub/Intel_Material/18.1/Computer_Systems/DE2-115/DE2-115_Computer_NiosII.pdf.

[7] M. A. Banks. ""BITS, BAUD RATE, AND BPS Taking the Mystery Out of Modem Speeds"." (1990), [Online]. Available: http://www.textfiles.com/apple/bitsbaud.txt (visited on 03/22/2023).

[8] IEEE. ""EEE 802.11-2007 Standard"." (2007), [Online]. Available: https://standards.ieee.org/ieee/802.11/3605/ (visited on 03/22/2023).

[9] IEEE. ""IEEE Code of Ethics"." (2016), [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html (visited on 02/08/2020).

[10] ACM. ""ACM Code of Ethics and Professional Conduct"." (), [Online]. Available: https://www.acm.org/about-acm/acm-code-of-ethics-and-professional-conduct (visited on 02/08/2020).

Appendix A UART

Refer to http://www.dejazzer.com/eece4740/lectures/example_uart_echo.pdf, the most basic method for communication with the FPGA chip or an embedded processor is asynchronous serial. It is implemented over a symmetric pair of wires connecting two devices (referred as host and target here, though these terms are arbitrary). Whenever the host has data to send to the target, it does so by sending an encoded bit stream over its transmit (TX) wire. This data is received by the target over its receive (RX) wire. The communication is similar in the opposite direction. This simple arrangement is illustrated in figure 26 below. This mode of communications is called "asynchronous" because the host and target share no time reference (no clock signal). Instead, temporal properties are encoded in the bit stream by the transmitter and must be decoded by the receiver. A commonly

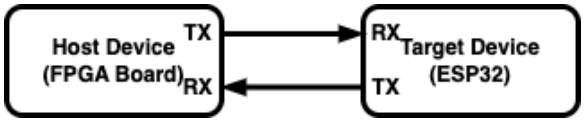


Figure 26: Basic serial communication

used device for encoding and decoding such asynchronous bit streams is a Universal Asynchronous Receiver/Transmitter (UART). UART is a circuit that sends parallel data through a serial line. UARTs are frequently used in conjunction with the RS-232 standard (or specification), which specifies the electrical, mechanical, functional, and procedural characteristics of two data communication equipment.

A UART includes a transmitter and a receiver. The transmitter is essentially a special shift register that loads data in parallel and then shifts it out bit by bit at a specific rate. The receiver, on the other hand, shifts in data bit by bit and reassembles the data. The serial line is '1' when it is idle. The transmission starts with a start-bit, which is '0', followed by data-bits and an optional parity-bit, and ends with stop-bits, which are '1'. The number of data-bits can be 6, 7, or 8. The optional parity bit is used for error detection. For odd parity, it is set to '0' when the data bits have an odd number of '1's. For even parity, it is set to '0' when the data-bits have an even number of '1's. The number of stop-bits can be 1, 1.5, or 2. The transmission with 8 data-bits, no parity, and 1 stop-bit is shown in figure 27 (note that the LSB of the data word is transmitted first). No clock information is conveyed through the serial line. Before the transmission starts, the transmitter and receiver must agree on a set of parameters in advance, which include the baud- rate (i.e., number of bits per second), the number of data bits and stop bits, and use of parity bit.

To understand how the UART's receiver extracts encoded data, assume it has a clock running at a multiple of the baud rate (e.g., 16x). Starting in the idle state (as shown in figure 28), the receiver "samples" its RX signal until it detects a high-low transition. Then, it waits 1.5 bit periods (24 clock periods) to sample its RX signal at what it estimates to be the center of data bit 0. The receiver then samples RX at bit-period intervals (16 clock periods) until it has read the remaining 7 data bits and the stop bit. From that point this process is repeated.

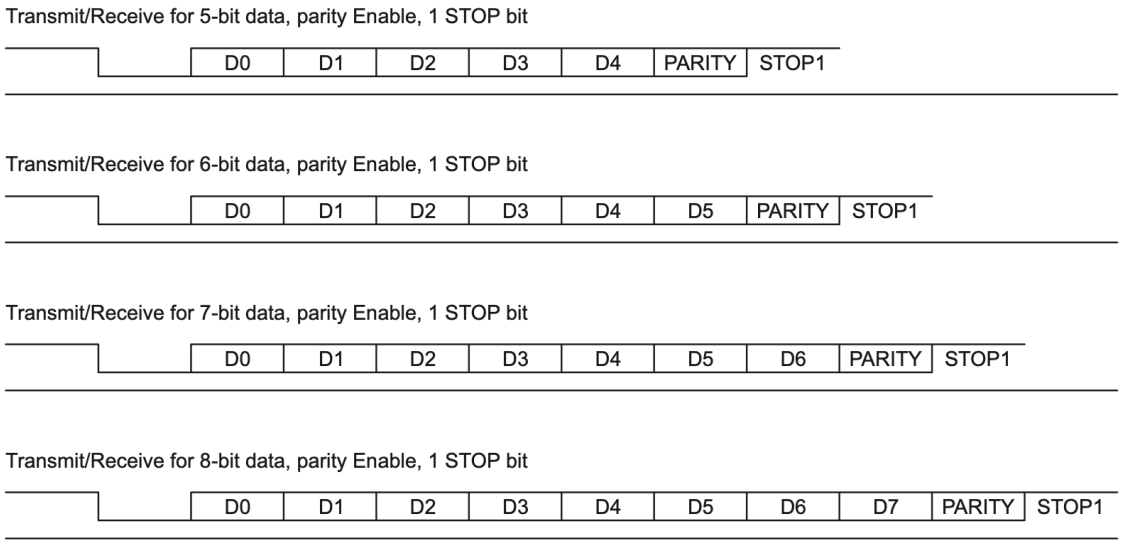


Figure 27: UART Protocol Formats

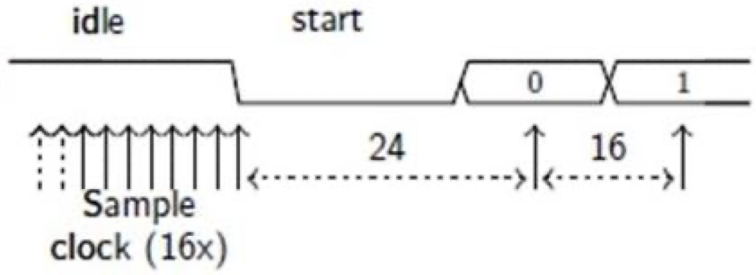


Figure 28: UART Protocol Formats

Appendix B Square Roots Algorithm

Let N be a number with n bits and let X be the square root of N, and we have the following equations:

$$N = (a_1 + a_2 + a_3 + ... + a_n)^2$$

$$X = \sqrt{N} = a_1 + a_2 + a_3 + ... + a_n$$

Note that N can be expanded to:

$$N = a_1^2 + (2a_1 + a_2)a_2 + ... + [2(\Sigma_{i=0}^{n-1} a_i) + a_n]a_n$$

In this case, it is possible to estimate the square root of N one bit at a time according to the following algorithm:

Starting from the most significant bit a_n, each estimated bit of the square root is removed from R and added to a variable P, where P is defined in the equation:

$$P_m = \Sigma_{i=0}^{m} a_i$$

So, if the $m-1$ bits $a_1, a_2, ..., a_{m-1}$ have already been estimated, each guess of the square root should satisfy the recursion of following equation:

$$X_m = X_{m-1} - Y_m, \ where \ Y_m = (2P_{m-1} + a_m)a_m$$

When $X_n = 0$ the exact square root of N have been found so far; otherwise, the sum of the ${}_a i$ gives an approximation of the square root of N.

A flow chat can of the algorithm is shown below.

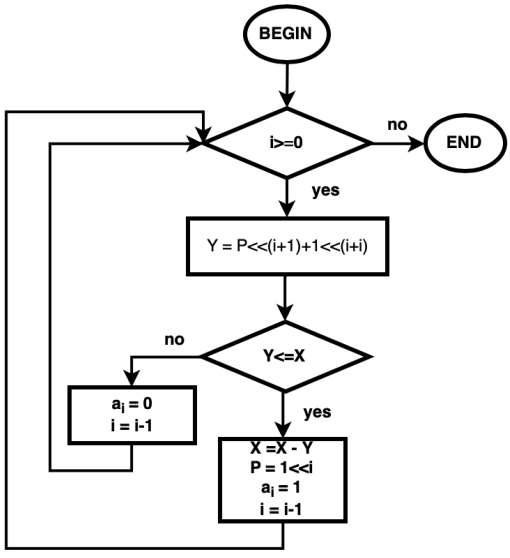


Figure 29: Square Roots Algorithm

Appendix C PCB Design (Partial)

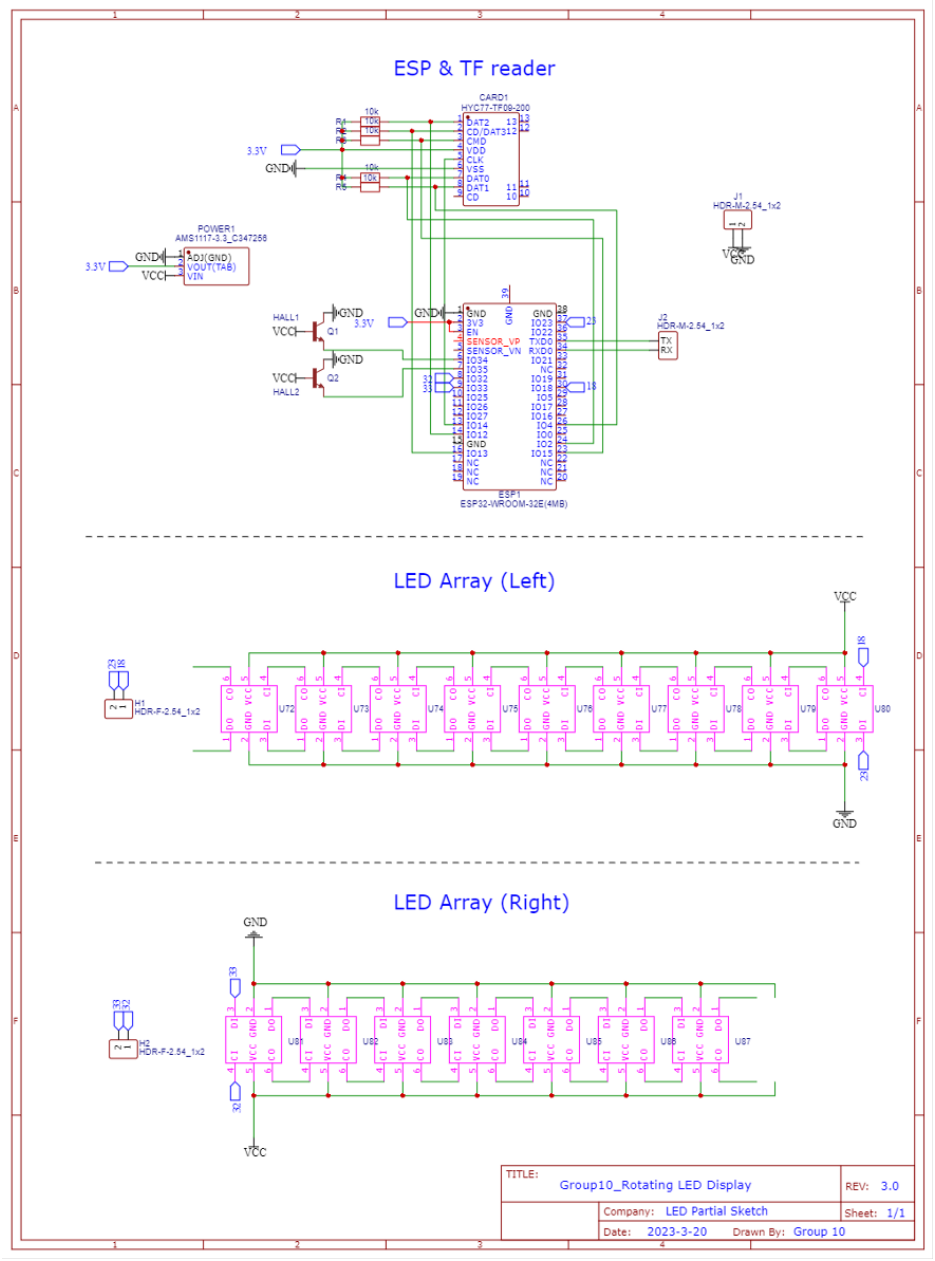


Figure 30: PCB Sketch (Partial)