TL25: KEY DOCUMENT

Design

Changelog

Date/Time	Updater	Description
2018-10-14T14:40:49Z	Sanket	
2018-10-14T14:51:31Z	Rowan	Document creation and initial population
2018-10-16T09:59:32Z	Rowan 11986156	• Add Design Overview • Add LED Matrix Design
2018-11-23T13:51:20Z	Jimmy	Added Communication between Tiles Section
2018-11-24T11:28:15Z	Sanket	Added Challenges Section and underneath added ambient light subsection
2018-11-25T11:39:45Z	Rowan	• Added Updated IR Sensor Matrix Section • Added Citations Section
2018-11-25T13:00:24Z	Jimmy	Update Communication Design (Formerly Communication between Tiles)
2018-11-25T14:00:00Z	Parth	Added section on Power systems design
2018-11-25T14:15:38Z	Candice 16550155	Added hardware calculations to IR sensor section Added to power system design section Major revisions of IR sensor and power system design sections

2018-11-25T19:32:10Z	Rowan 11986156	Milestone 2 prep
2019-02-05-6T5:03:00Z	Rowan 11986156	Add glossary of terms, update sections 1-3
2019-02-10T14:00:00Z	Jimmy 40243140	Added Section 4.3 Position Mapping
2019-02-10-6T17:16:00Z	Sanket	Added Section 6 Wi-Fi Connectivity
2019-02-10-6T18:36:00Z	Candice 16550155	Revised IR sensor resistor justification
2019-02-10-6T23:34:00Z	Parth 45738135	Updated section 5, grammar edits

Table of Contents

1 - Design Overview....1
2 - LED Matrix Design....3
3 - Infrared Sensor Matrix Design....4
4 - Communication Design....7
4.1 - Modularity....8
4.2 - Data Transfer....8
4.3 - Position Mapping....9
5 - Power System Design....12
5.1 - Power Requirements....12
5.2 - Power Design Features....12
5.2.1 - Connection....12
5.2.2 - Efficient Power conversion....13
6 - Wifi Connectivity Design....15
Citations....17

Table of Figures

Figure 1: Nine Tile display mirroring a hand gesture....1
Figure 2: Exploded model view of Tile PCB and faceplate....2
Figure 3: APA102 connection schematic....3
Figure 4: Tile LED and sensor matrix layout....4
Figure 5: Faceplate cross section with emitter and sensor....6
Figure 6: Dynamic Addressing Flowchart....8
Figure 7: Visual Representation of I2C Bus....9
Figure 8: Master Tiles Internal Layout Map....10
Figure 9: Pogo connectors on Tiles....12
Figure 10: Power Cord Connector Model....13
Figure 11: Visual of Power Distribution....14
Figure 12: Wi-Fi Connectivity Communication Pathway....15

Table of Tables

Table 1: IR sensor power consumption and output voltage....5
Table 2: Serial communication protocol comparison....7

1 - Design Overview

The product is an interactive modular display composed of discrete square shaped Tiles that can be rearranged “on the fly” to change the shape and size of the display. Each Tile can respond to gestures directly in front of it and render text and imagery.

Each Tile has a matrix of Light Emitting Diodes (LED) and sensors mounted onto a Printed Circuit Board (PCB). The Tiles can sense movement and display it back onto their LED matrices. [Fig 1] is an artistic representation of a single display of nine Tiles mirroring a hand movement.

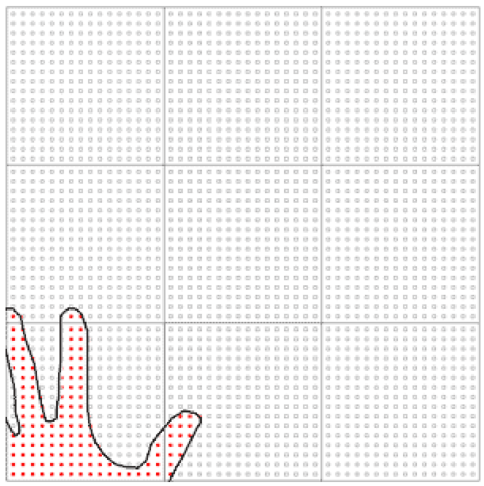

Figure 1: Nine Tile display mirroring a hand gesture

The Tiles connect to each other via a magnetic pogo pin connector. The LEDs matrices are positioned from the edges of the Tile to create a seamless display when multiple Tiles are connected. The display can be powered from any of its Tiles with a magnetic power plug.

The components are protected by a face plate, which is a 3D printed piece that gives the Tile a flat surface and shields the sensors from direct sunlight. An exploded view of the Tile assembly including the PCB, pogo pins, and faceplate can be seen in [Fig 2].

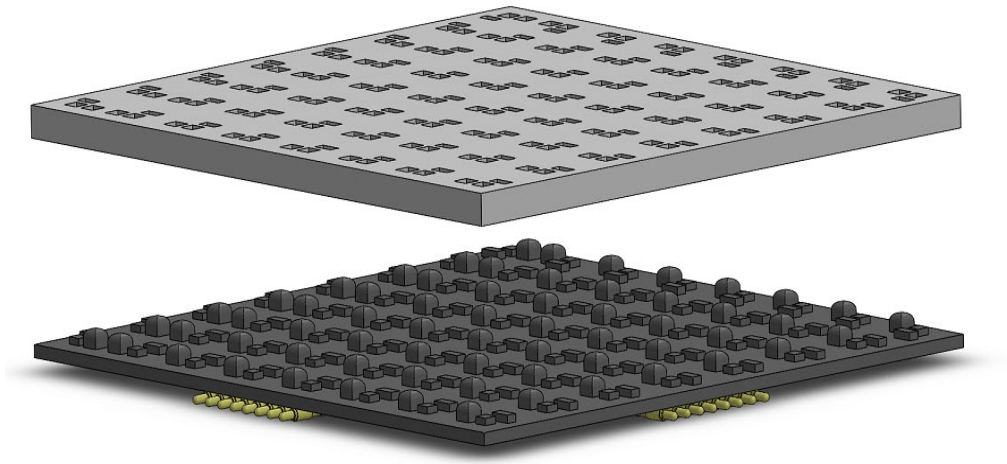

Figure 2: Exploded model view of Tile PCB and faceplate

2 - LED Matrix Design

Each Tile has an 8x8 LED matrix to display text and mirror gestures. This matrix can display at least one 5x7 pixel character of text at a time as per requirement [6.4,1]. The matrix is square with a pixel pitch of 12mm. This gives an overall Tile size of 96mm by 96mm, which satisfies the requirement [6.3, 1]. The LEDs used are an APA102 2020 addressable rgb LED. These LEDs are manufactured by the Adafruit company.

Each LED in the matrix is connected serially via a data line and clock line. Each LED is powered in parallel with 3.3V. See [Fig 3] for the LED connection schematic.

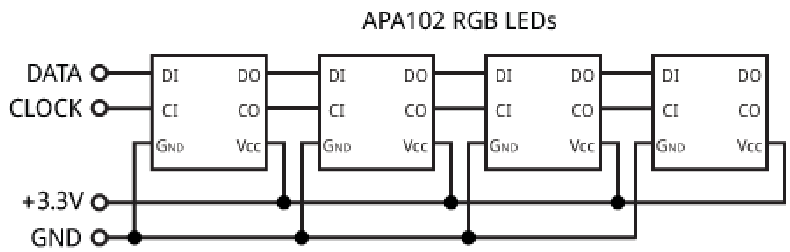

Figure 3: APA102 connection schematic

3 - Infrared Sensor Matrix Design

Each Tile has an 8x8 infrared (IR) sensor IR emitter pair matrix. The sensors are placed horizontally adjacent to the LEDs while the emitters are placed vertically adjacent to the LEDs. The sensor-emitter pairs are spaced by 12mm in a square grid between the LEDs except for the first row and column of the matrices. This ensures that all components fit within the edges of the PCB while maintaining a unique sensor for each LED, which is necessary to meet the sensor resolution requirement as per [6.4.2, 1]. A graphical representation of the LED, sensor, emitter matrix layout can be seen in [Fig 4].

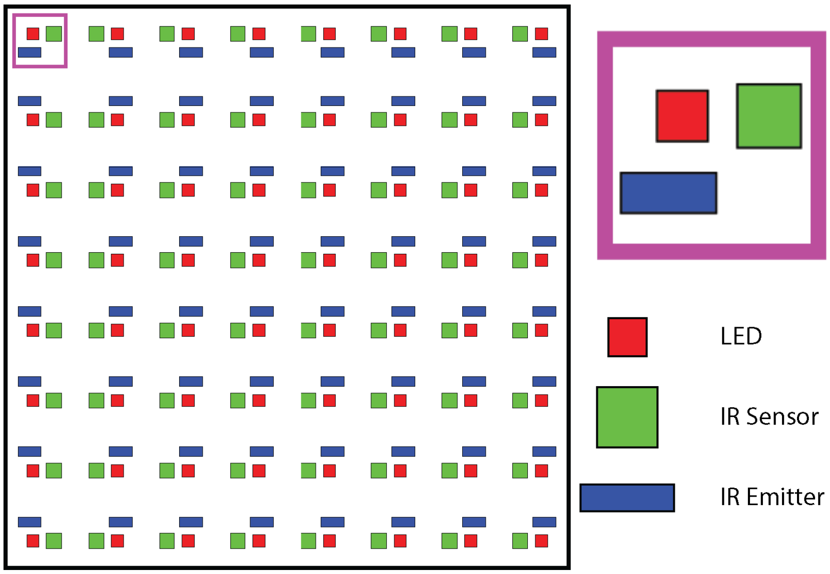

Figure 4: Tile LED and sensor matrix layout

The IR sensor selected is the VEMT2020x01. The IR sensor can be modelled as a BJT. To produce a readable voltage at the output, a resistive load is connected to the emitter of the sensor, as seen in the figure below [Fig 5]. The value of the resistive load is computed as follows with assumptions $I_C \approx I_E$ for $\beta \geq 100$ and $V_{CE(sat)} = .4V$.

$$R_L > \frac{V_{o(max)}}{I_{c(max)}} = \frac{3.3V - .4V}{9mA} = 322\Omega$$

In the interest of decreasing the rise time of the output signal, a smaller resistor is recommended, as it decreases the RC delay generated by the IR sensor circuit. However, there is also an interest in generating enough output voltage difference for a change in distance of the object detected, as this will allow the microcontroller's analog to digital converter (ADC) to detect changes in distance more easily. Thus, the next most common resistor value that satisfies the above requirement, 470Ω, was chosen.

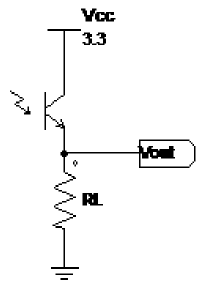

Figure 5: NPN IR phototransistor circuit schematic

For further justification of the IR sensor selection, the output voltage and power dissipation at several input conditions is displayed in the table below.

Table 1: IR sensor power consumption and output voltage

Base Current Condition	$I_{c(dark)} = 100nA$	$I_{c(min)} = 3mA$	$I_{c(max)} = 9mA$
$V_o\ (V)$	47µ	1.01	3.3
$P_{diss}\ (mW)$	.0005	3.03	3

Each IR sensor/emitter pair will be multiplexed. An interrupt subroutine will sequentially turn on emitters and read the sensors using the following logic:

if first sensor:

Turn on current emitter

Turn on next emitter

Wait for sensor rise tim e

Read current sensor

Store sensor data

else if not last sensor:

Turn on next emitter

Turn off prev Emitter

Read current sensor

Store sensor data

else:

Read current Sensor

Turn off current emitter

Store sensor data

Power consumption will be kept to a minimum, as only a couple of IR emitters will be on simultaneously. Furthermore, because only one IR sensor-emitter pair will be powered at once, the IR reflected will be local for each sensor. Thus, there is no need for blinding the IR sensor/emitter pair from each other.

The faceplate will create a physical barrier between the IR sensor and emitter so only light reflected from the object is detected. A graphic depicting this concept can be seen in [Fig 6].

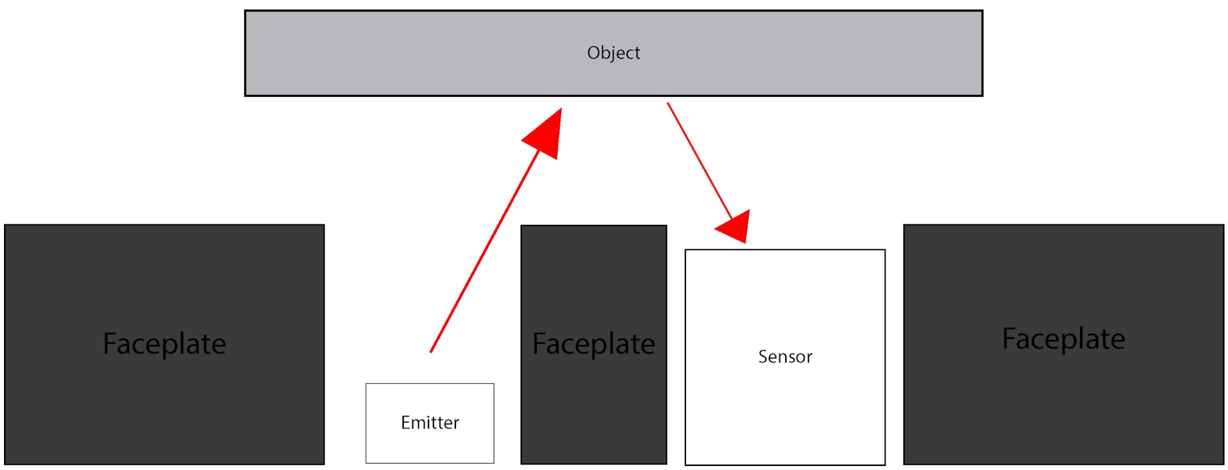

Figure 5: Faceplate cross section with emitter and sensor

The IR emitter selected is the SFH 4056. See ["IR Emitter and Sensor Selection Justification", 2] for justification.

4 - Communication Design

The inter-Tile communication in a single display is implemented with a serial communication protocol. The Inter-Integrated Circuit (I^2C) serial communication protocol is the best choice for the project as it allows us to:

- Implement the functional requirement of being modular [4.2, 1]
- Satisfy the specification of being able to add and remove devices dynamically [4.2.1, 1]
- Satisfy the specification of creating a unified display by allowing devices to exchange data [4.2.2, 1]
- Satisfy the 8-month constraint of the project

During the planning phase we removed from consideration UART, SPI and CAN Bus for the following reasons:

- Number of total devices on the bus
- Pins required
- Data Transfer Limits

The limits of each serial communication protocol are summarized in the table below.

Table 2: Serial communication protocol comparison

I^2C	SPI	UART	CAN Bus
Up to 127 devices using 7 bit addressing	Up to number of unused GPIO pins	One-to-one device communication	Typically, 30 Nodes [4.12, 3]
1 SCLK (clock) and SDA (data) line	1 MOSI, 1 MISO, 1 SCLK, 1 Slave Select (per slave)	1 Tx, 1 Rx pin	1 Can Hi, 1 Can Lo,
400 Kbits/s	18 Mbits/s	4.5 Mbits/s	1 Mbits/s 8 bytes/message [Figure 3, 3]

4.1 - Modularity

A single master device controls the data flow between all the slave Tiles in the display. Each slave Tile only needs to synchronize with the master and not any other Tiles. UART only supports a one-to-one device communication which does not meet our constraint. Additionally, the SPI interface uses a physical slave select line for each individual slave removing the ability to add or remove slaves dynamically.

One concern with I^2C for this modular method is that devices are usually assigned a fixed address in the code. To address this, we will modify the I^2C software to dynamically allocate addresses as outlined below [Fig 6]. The blue and red boxes represent the slave and master Tiles respectively.

1. Slave device connects to the I^2C bus with default address
2. Master detects a device on default address and will check for the next available address.
3. Master marks the address as unavailable and then sends it to the slave device.
4. Slave receives this address on the I^2C bus
5. Slave re-initialize I^2C hardware with new address.

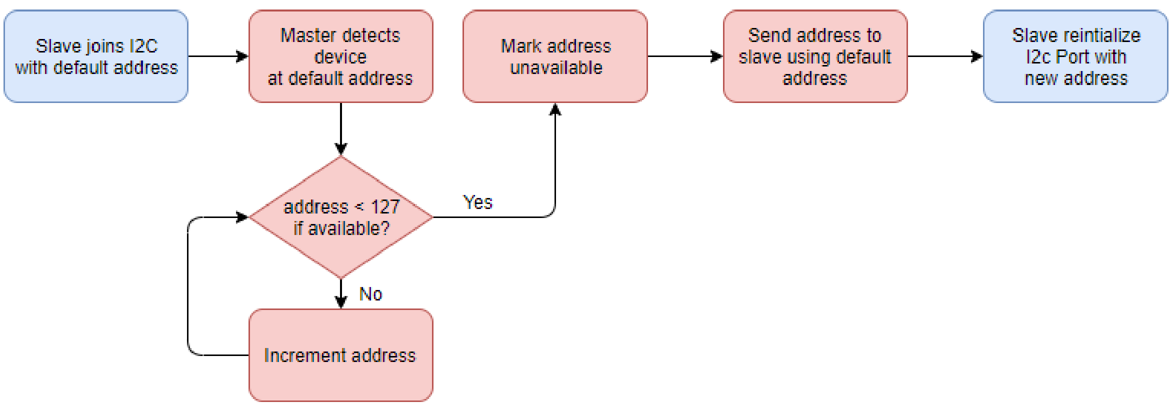

Figure 6: Dynamic Addressing Flowchart

With the use of dynamic addressing, we can have all slave devices run the same code, simplifying the process of configuring multiple devices. If the master assigns an address as each slave appears it minimizes the risk of having an address belonging to multiple devices.

4.2 - Data Transfer

The decision made between I^2C and Can Bus was based on the data transfer format we expect to use. To find our transfer rate, we used the following sample message to justify our decision.

Sample Data Packet

- 8 bits - position data (4 bits to select 16 horizontal or vertical lines)
- 8 bits - per character data
- 16 bits - RGB data (around 4 bits each for 16 levels of brightness for each colour)

Each character is about 5 by 8 pixels so on a 16 by 16 display we can fit 3 horizontally and 2 vertically. We also have the case where characters are partially on the screen so we can fit a total of 4 characters horizontally and 3 vertically for a total of 12 characters. This means our data packet can vary from 4 bytes to 15 bytes. The varying message length causes an issue on the CAN Bus since the maximum number of bytes contained in a single message is limited to 8 bytes. CAN Bus would require us to possibly send more than one message at a time complicating the communication protocol.

In I^2C, given that our maximum data length is 16 bytes (15 bytes + 1 additional byte for address data), we can then calculate the expected speed requirements. NanoLeaf light panels support a maximum of 30 panels [4] and be the number used for analysis. For scrolling text, we will use our targeted frame rate of 24 fps.

$$30\, Tiles\ \times 24\, fps\ \times 16\, bytes\ = \mathbf{11520}\, bytes\ or\ \mathbf{92160}\, bits/s$$

This value is much lower than the maximum data transfer rate of the STM32F103C8 I^2C bus of 400 Kbits/s [5]. Therefore, the master can transmit 16 bytes of data to any slave on the I^2C bus without concern for speed [Fig 7].

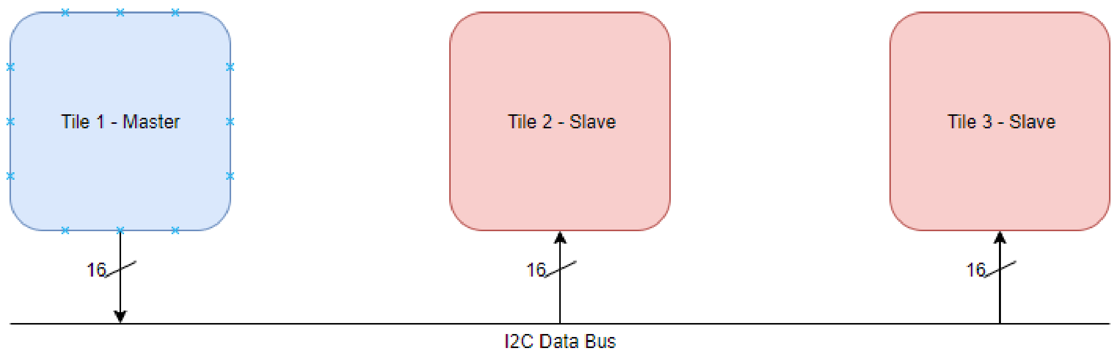

Figure 7: Visual Representation of I2C Bus

4.3 - Position Mapping

The master needs to know the position of each Tile in order to be able to determine what data a Tile gets based on the position the Tile is currently at. Thus, the master will need a method to correctly determine the position of each Tile. We are using a 2D integer array to achieve this functionality.

The master Tile is always located in the center of this 2D array. The array size will also depend on the maximum number of Tiles supported. If X represents the maximum number of Tiles, the array size will be (2X - 1) by (2X - 1). The reason for this size is that for X number of Tiles the maximum length it can extend is X in any one direction.

For example, if we have 4 Tiles our array size will be 7 by 7. The coordinates of the master Tile will then be the center at (4, 4). The 3 other Tiles, represented by T1, T2, and T3, will be located at (5, 4), (6, 4) and (7, 4) if we were only connecting them to the right of the master. All configurations possible with four adjacent Tiles are shown below as white boxes in the following figure [Fig 9].

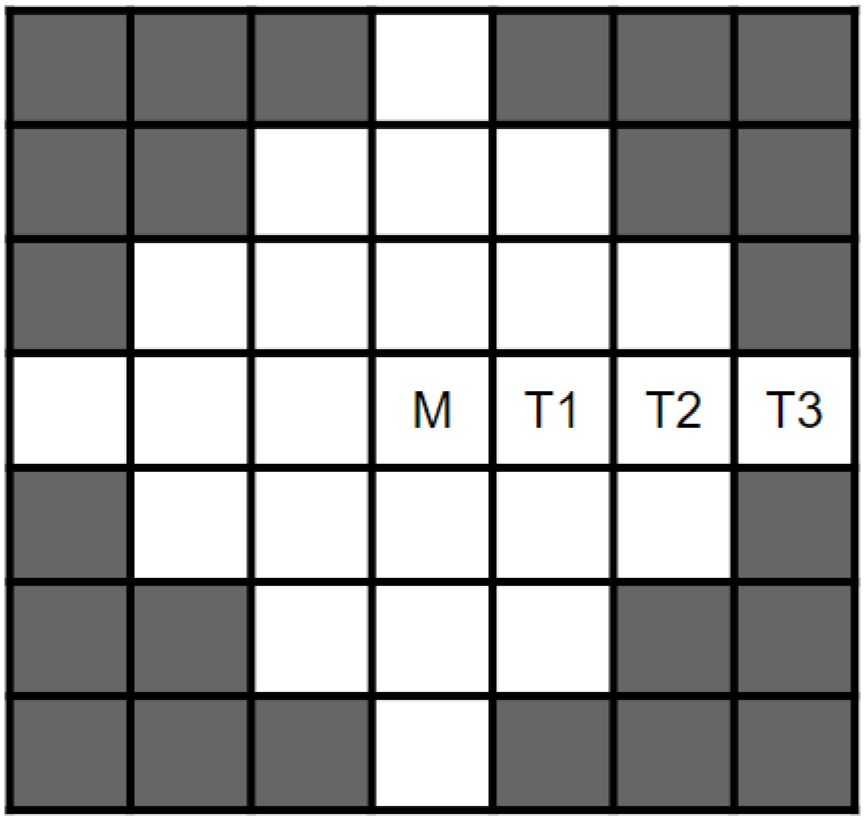

Figure 8: Master Tiles Internal Layout Map

The STM32F103C8 microcontroller we are using has a maximum of 20480 bytes available for variables. Thus, we have a maximum of 5120 four-byte integers for the array. This translates to a 71 x 71 matrix or a maximum of 36 Tiles constraint, which does not restrict our goal of 30 Tiles. The calculation for the maximum number of Tiles is shown below:

$$20480\ bytes \times \frac{1\ integer}{4\ bytes} = 5120\ integers \approx 71.55\ \times 71.55\ integers$$

$$\therefore (2X - 1) = 71.55 \Rightarrow X\ \approx 36.2\ Tiles$$

As mentioned in section 4.1, the slave Tiles will start with the default address and dynamically assign new addresses as they join the bus. Each address assigned also has an identification number associated with it. The master must place these identification numbers at the correct coordinates in the 2D array. To achieve this, each Tile has four directional input pins corresponding to up, down, left and right. This will help determine the relative location of a connected Tile. The algorithm we will use is summarized below.

1. Each Tile polls the status of its four directional pins.
2. Each Tile continuously monitors a change in the value of the pins raises a flag for that direction when it detects change.
3. The master polls every Tile to see if the flag has been raised. If there is, the new Tile's position can be determined using the coordinates of the currently polled Tile.
4. The master checks the I^2C bus for a slave using the default address. If detected, the master sends the next available address for use to that slave Tile.
5. The master fills the array using the coordinates determined in step 3 and the identification number of the assigned address in step 4 upon successful communication.

5 - Power System Design

5.1 - Power Requirements

The display consists of a variable number of Tiles, which, as of now we have restricted to 30 per display. An individual Tile requires a relatively small amount of continuous power (.875W - 1W). However, a whole display may have variable power requirements depending on number of Tiles and the content displayed. Hence, due to a maximum Tile configuration of 30 Tiles (30W), the maximum available power supply needs to be high and at the same time be able to regulate to lower power with minimum power loss as other forms of energy, such as heat.

The generally accepted rating when considering inrush current is approximately 20 times [6], but because our system is primarily low power and will not have many large reactive elements, in consideration of the lifetime of the LEDs 10 times the estimated maximum power consumption would be ideal. Although, in the scope of this capstone project, a power adapter rated 60W will be used.

5.2 - Power Design Features

5.2.1 - Connection

The display can be powered from any Tile. A Tile will consist of a connection port on each of the 4 sides with 2 male connection ports on the top and right, and 2 female ports on the bottom and left sides which also connect to the power cord [Fig 11]. They will be held together on each side by neodymium magnets.

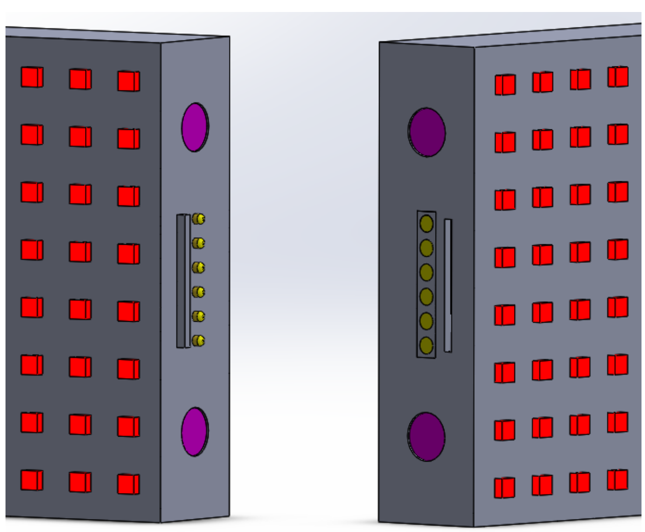

Figure 9: Pogo connectors on Tiles

Magnetic pogo pin connectors are used which have a current rating of 2 Amps(A). Besides the 2 data pins, each port will consist of 4 pogo pins for power flow as opposed to just 2 since that allows lowering the voltage supply to half, while supplying the same power. With 2 voltage pins we can supply maximum 4A. To extend the life of pogo pins if we pass half the current limit through each of them (=1A), to provide 30W of power, we would need:

$$V = \frac{P}{I} \Rightarrow V > \frac{30W}{1A} = 15V$$

The voltage pins in each connector will be installed symmetrically to allow for 2 connection configurations, as shown in [Fig 11].

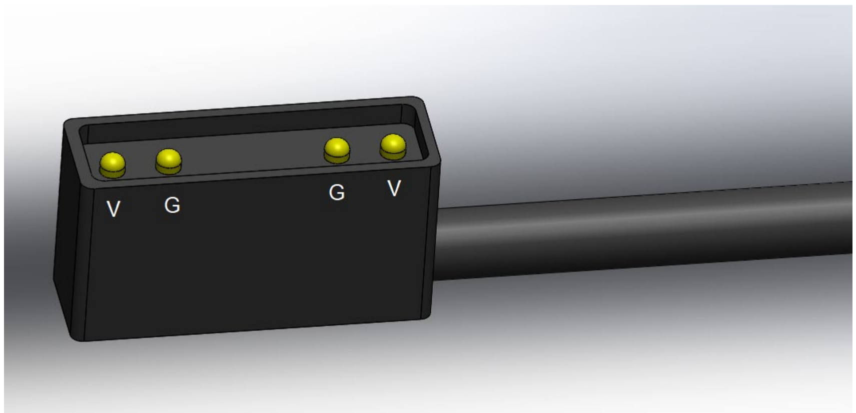

Figure 10: Power Cord Connector Model

A power adaptor of **18V/ 3.61A (65W power)** has been chosen to allow for enough room in power while considering current constraints on pins. This will consist of an AC/DC converter and will hence feed DC power into the connected Tile. This 65W DC power line will run across each Tile, allowing the Tile to use a small amount of power from the bus and remaining power to flow into the connected Tiles.

5.2.2 - Efficient Power conversion

Stepping down from high power supply a high voltage bus to a much smaller one for each module requires stepping voltage down while still being able to pull required current.

If each Tile's: $P = 1W$ at $V = 3.3V$,

Then, $I_{in} = \frac{1W}{3.3V} = 0.303A$ (input current per Tile)

DC/DC buck converter - The purpose of this device is to convert the 18V output of the AC/DC converter to 3.3V for use by the microcontroller, LEDs, IR sensors and emitters. Buck convertors operate at about 96% efficiency and can step up the input current to supply enough current to the Tiles' components. This allows us to pass more current in our Tiles' components than the AC/DC converter would provide.

The buck converter provides 1W using an 18V input and minimum required current:

$$P_{in} = I_{in}V_{in} \Rightarrow I_{in} = \frac{1W}{18V} = 55.56mA \text{ (input current per Tile)}$$

This is **1.667A** of total current draw for max configuration of 10 Tiles and hence the supply of 3.6A will be enough. The buck convertor will subsequently provide power (in parallel) to the components within the Tile as seen in [Fig 12].

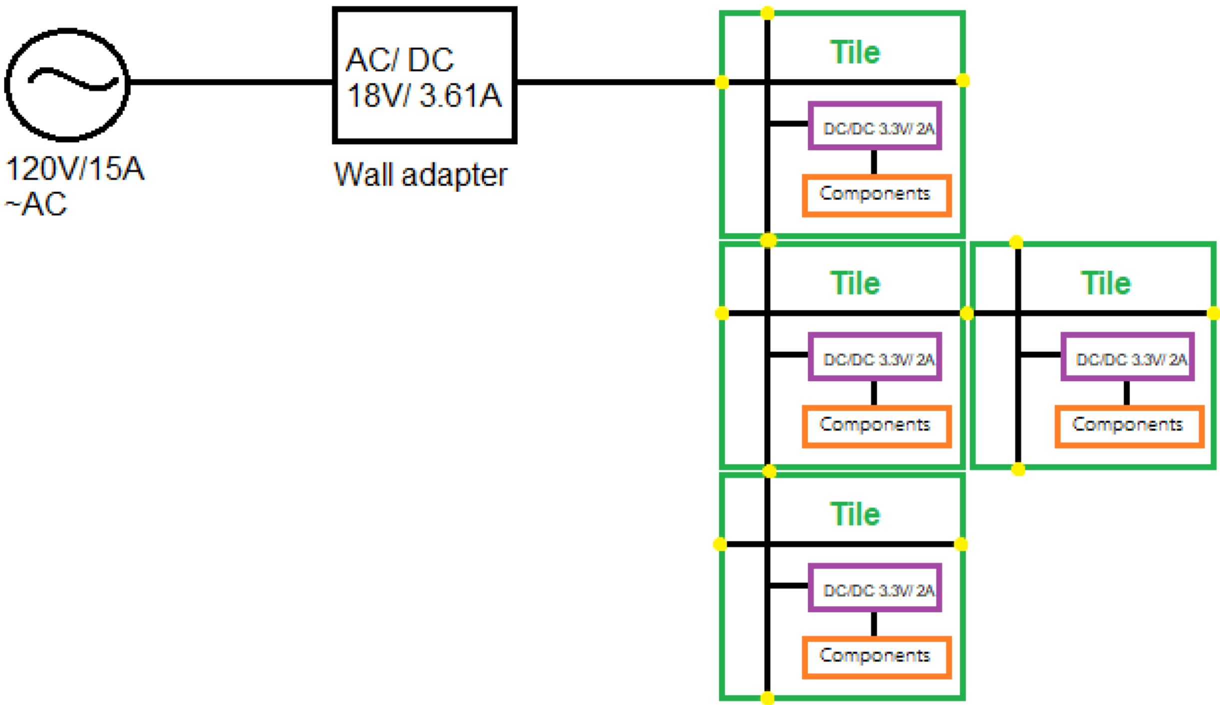

Figure 11: Visual of Power Distribution

6 - Wi-Fi Connectivity Design

A key aspect of the product is giving the user freedom to control the display wirelessly. This is achieved by using the Wi-Fi communication protocol connected to a basic Android application. The below figure [Fig 13] visualizes the communication pathway taken by the data from the app to the Tile.

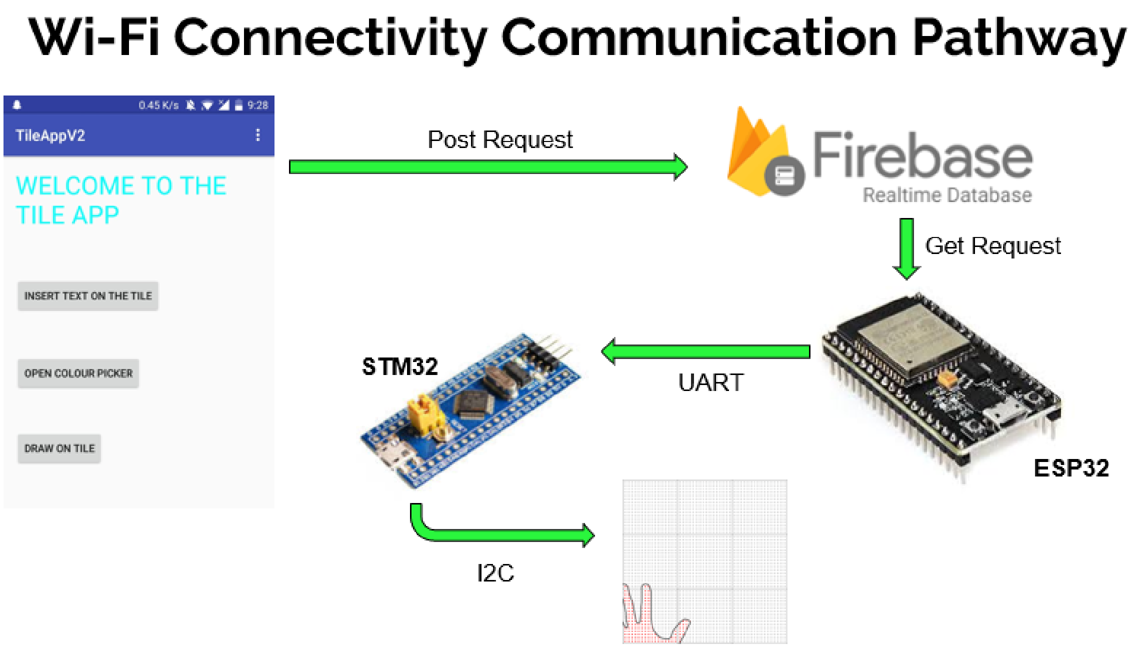

Figure 12: Wi-Fi Connectivity Communication Pathway

The communication starts from the app when performing any of the three following functions colour change, display text or draw on Tile. The app transfers data to the Firebase Database where the information is stored for future use and passed along to the ESP32. The ESP32 is a Wi-Fi enabled microcontroller that acts as a local access point allowing wireless data transfer to occur between the database and itself. Once the data has arrived at the ESP32, it transfers through the UART communication protocol to the STM32. The STM32 is directly wired to the LEDs and through I²C protocol gives it the necessary commands to perform the functions asked of it.

Glossary of Terms

API	An Application Programming Interface is a set of functions and procedures allowing the creation of applications that access the features or data of an operating system, application, or other service.
BJT	A bipolar junction transistor (BJT) is a type of transistor that uses both electron and hole charge carriers, as opposed to field effect transistors which use only one type of charge carrier.
CAN bus	Controller Area Network (CAN bus) is a robust bus standard for microcontroller communication without a host computer.
SPI	Serial Peripheral Interface (SPI) is a synchronous serial communication interface specification for short distance communication in embedded systems.
IR	Infrared (IR) is a spectrum of non-visible light spanning in wavelength from Red (700nm) to Microwaves (1mm).
I^{2}C	I2C is a serial protocol for two-wire interface to connect low speed devices, such as microcontrollers, in embedded systems.
Master/slave	The Master/slave model of communication has a single master device with unidirectional control over one or more slave devices. Slave devices only send or receive data upon a request by the master.
PCB	A Printed Circuit Board (PCB) is a sheet of layered copper and non-conductive substrate that mechanically supports and electrically connects electronic components via embedded copper tracks.
Phototransistor	A semiconductor junction that alters the level current flowing through it depending on the level of light the junction is exposed to.
Pogo Pin	A spring-loaded conducting device used to connect two PCBs.
LED	A Light Emitting Diode is a semiconductor junction that emits light when current flows through it.
UART	Universal Asynchronous Receiver/Transmitter is a physical circuit in microcontrollers that uses two wires (TX and RX) for serial data transfer.

Citations

[1] Team 25, “TL25: KEY DOCUMENT Requirements”, Capstone Key Document, Team 25, November 2018

[2] Team 25, “TL25: KEY DOCUMENT Validation”, Capstone Key Document, November 2018

[3] Texas Instruments, "Controller Area Network Physical Layer Requirements", Application Report. SLLA270, Jan. 2008

[4] NanoLeaf, “LED Lighting Products: Technical Specifications”, 2018 [Online] Available: https://NanoLeaf.me/en-ca/consumer-led-lighting/for-owners/get-support/technical-specifications/. [Accessed Oct 16, 2018].

[5] STM32F103C8 datasheet, https://www.st.com/, Oct 10, 2018

[6] SunPower UK. (2018). What is Inrush Current? - SunPower UK. [online] Available at: https://www.sunpower-uk.com/glossary/what-is-inrush-current/ [Accessed 26 Nov. 2018].

[7] Renesas, “Linear vs. Switching Regulators”, 2018 [Online] Available: https://www.renesas.com. [Accessed: Nov 2, 2018]