PHOTOCELL MUSIC BOARD

By

Sean Li

Alonzo Marsh

Final Report for ECE 445, Senior Design, Spring 2018

TA: Kexin Hui

2 May 2018

Project no. 39

Abstract

The paper discusses the design and implementation of an electronic music board. The music board detects light intensity with a square array of 256 photoresistors. The incident light level on each photoresistor is measured and transmitted to a computer. The data is manipulated with audio processing programs written in SuperCollider. The photoresistor circuits and user interface functioned correctly. However, communication between microcontrollers had significant time delays, and the microcontrollers were unable to accurately read photoresistor data due to a design flaw with the analog to digital converters' reference voltage.

Contents

1. Introduction

1.1 Objective

The invention of unique musical instruments and techniques for playing them creates new opportunities for musical expression. The realm of electronic music still has a lot of unexplored potential for nontraditional music controllers. A niche exists for a music controller that uses electronic sensors and modern digital signal processing (DSP) algorithms to provide a new level of customization and control over the produced sound.

We designed a modular music board to fill this niche. This board can support multiple types of sensors, such as photoresistors, flex sensors, touch sensors, and any other resistive sensor. Sensors and supporting circuitry soldered onto modular circuit boards connected with arbitrary number of other boards create a music board sensor array. The version we implemented, based on Dr. Eli Fieldsteel's 2017 design, uses photoresistors distributed over an 18 by 18 inch region.

The music board interfaces via serial with a computer running a SuperCollider music synthesis program, "a platform for audio synthesis and algorithmic composition, used by musicians, artists, and researchers working with sound." [1] The program processes the data sent from the music board into audio based on shadows detected by the photoresistor array.

1.2 Background

Dr. Eli Fieldsteel, Director of the University of Illinois Experimental Music Studios [2], designed and built a prototype music board that uses photoresistors to detect drops in light intensity. His aimed to design a new and innovative way to generate musical patterns. In the original design, each photoresistor mapped to a sound, and the computer generates music based on the change in intensity at each photocell. With this mapping, a musician can use the music board to play music by casting shadows on the photoresistors.

For our project, we extended Fieldsteel's original work to create a more robust and modular version of his design. His original prototype uses multiple solderless breadboards and insulated wire to connect the photoresistors to a microprocessor. The prototype exhibits structural issues during transport and is difficult and time-consuming to build and debug. The prototype sequentially polls all the photoresistors, which may limit the board's responsiveness when reading hundreds of sensors. Our design addresses these issues by implementing a modular design, a user interface with debugging tools, and uses printed circuit boards to increase structural strength.

1.3 High Level Requirements

1. The music board must sense changes in light intensity at a frequency greater than 1000 Hz and transmit that data through a serial connection to a laptop computer.

2. The music board must have a modular photocell array - so in the event of a board failure, a single board can be easily replaced.

3. Design must be robust, capable of handling the stress of travel to national conventions.

2 Design

2.1 Block Diagram

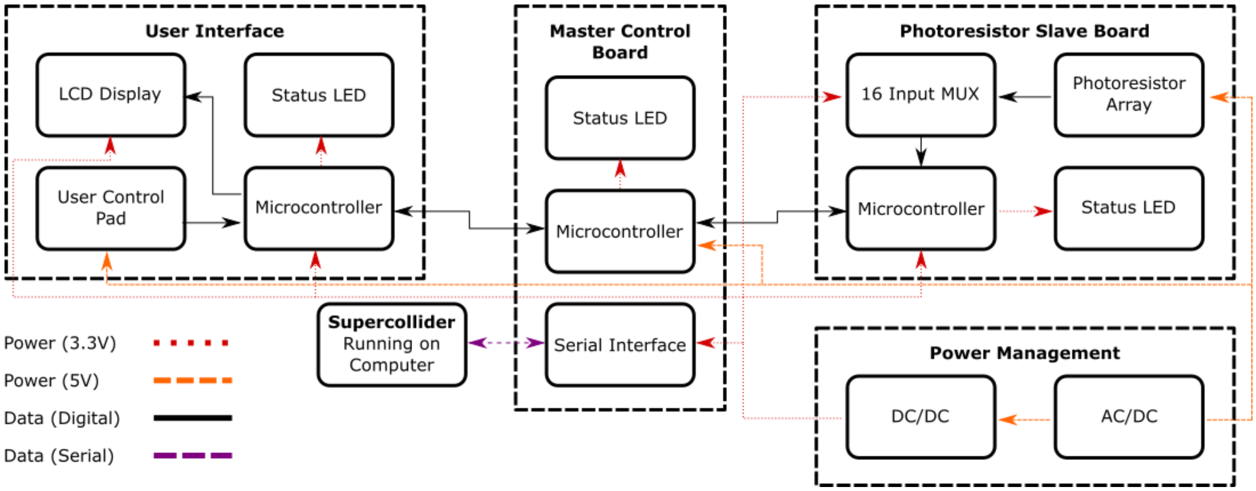

Figure 1: System block diagram. The full system contains 16 identical photoresistor slave boards connected to the master control board.

The design shown in the block diagram (Figure 1) satisfies the high-level requirements. During implementation, the design satisfied the modularity and robustness requirements (Requirements 2 and 3), but not the speed requirement (Requirement 1).

The design was not able to transmit photoresistor data faster than 1000 Hz (Requirement 1). During testing, we found that the clock syncing signal had a max rate of around 100 Hz, which made the polling rate of the entire music board around 0.038 Hz. For more information on program design, see Section 2.4.

The music board was built using printed circuit boards. This satisfies Requirement 3 since printed circuit boards are more resistant to vibration and regular wear than solderless breadboards.

The design of the photoresistor slave boards satisfies Requirement 2. Each slave board is connected to the other boards with only physical wires, and each board is individually mounted to the base. The modularity of the slave board design simplifies the repair process.

2.2 Physical and PCB Design

The music board is shown in Figure 2. The photocell circuit boards were mounted in a wooden case. Measuring 18 inches at the base and 20 inches deep with a height of 2 inches, the music board easily fits on a standard table and can be easily transported. A pane of clear plastic (not pictured) sits above the circuit boards to protect the circuit boards from damage and users from shock hazards.

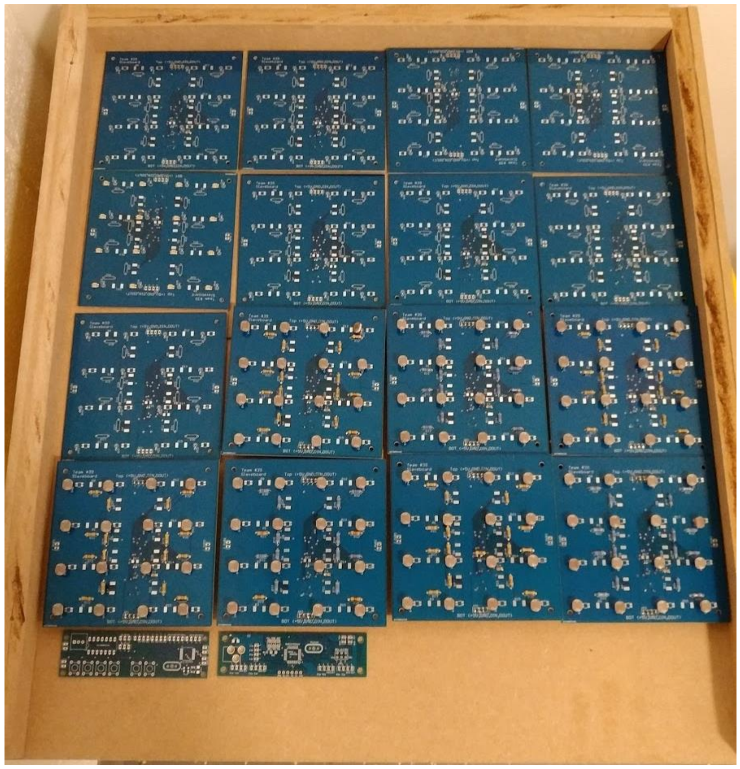

Figure 2: Final music board with all of the photoresistor slave boards arranged in a 4 by 4 grid. The bottom left board is the user interface board, and master board sits directly right of it.

The photoresistor slave board is pictured in Figures 3 and 4. It features an array of 16 photoresistors spaced on a 4-by-4 grid with 1 inch separation between each photoresistor. The board measures 4 inches by 4 inches with connectors on each of the four sides to allow it to connect to other boards. The board has four screw holes near the corners that mount the board to the casing.

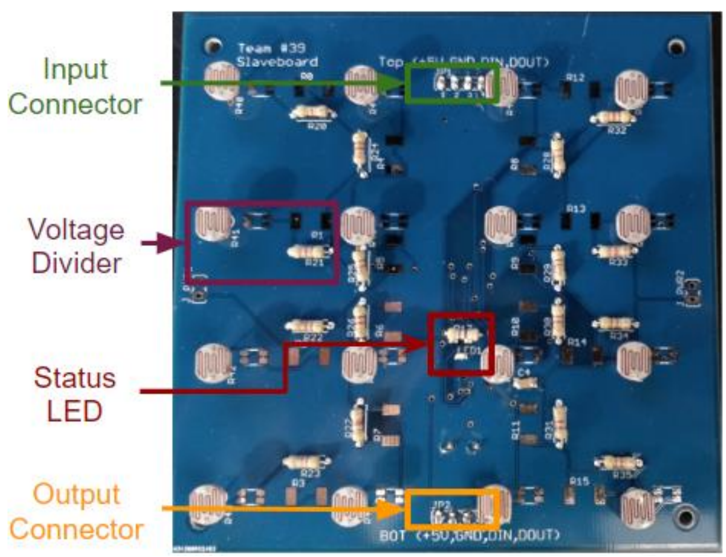

Figure 3: Top side of photoresistor slave board.

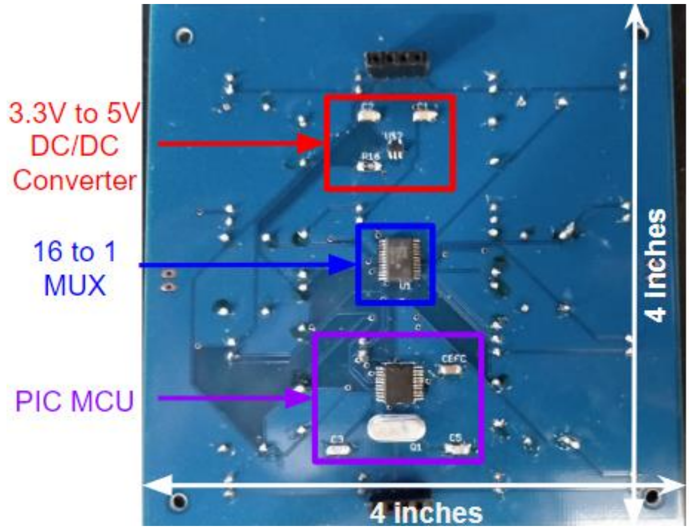

Figure 4: Bottom side of photoresistor slave board.

A single master board (Figures 2.5 and 2.6), mounted on the edge of the baseplate, connects to four slave boards and connects to the user interface board. Female header pin connectors attached to the bottom of the board allow for easy connection to other boards.

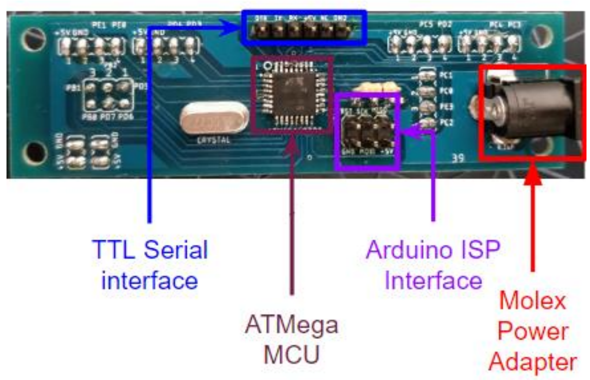

Figure 5: Top side of master control board.

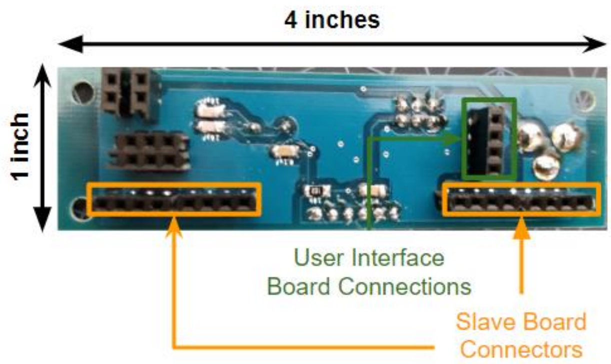

Figure 6: Bottom side of master control board.

Finally, the system includes a user interface board (Figures 7 and 8). It is connected to a small LCD screen (Figure 9) and various buttons. This board is wired to the master control board.

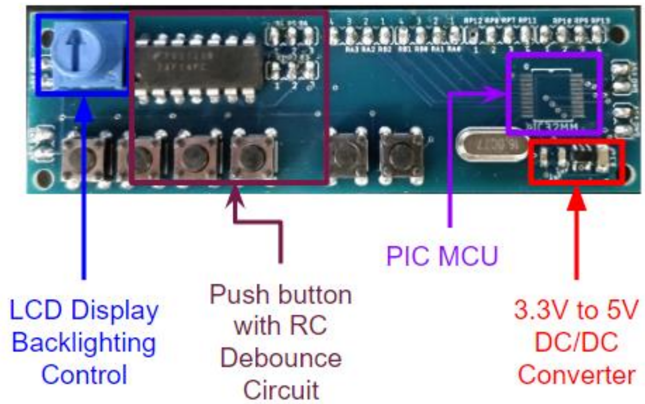

Figure 7: Top side of user interface board.

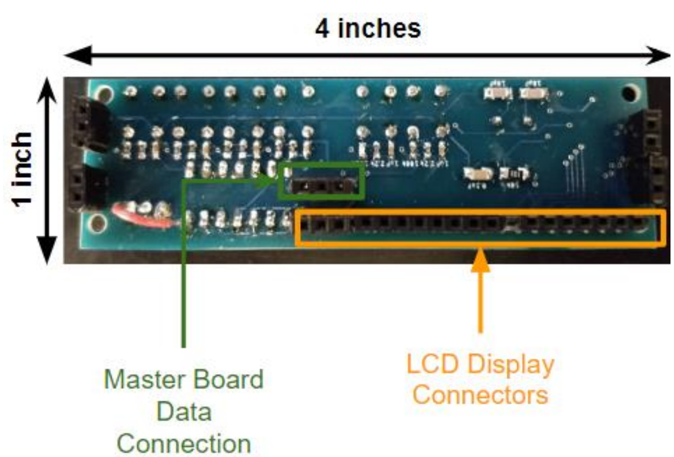

Figure 8: Bottom side of user interface board.

Figure 9 shows the LCD screen [3]. This screen is controlled by the user interface board through its microcontroller.

Figure 9: LCD Screen by Crystalfontz [3].

2.3 Block Design

2.3.1 User Interface (UI) Board

2.3.1.1 LCD Screen

The LCD screen displays the current selected preset, and if an error has occurred, the error number. The screen receives display data from the user interface microcontroller. While the music board is active, this display will be off to limit interference unless it is actively being used by the musician or if an error message needs to be displayed.

2.3.1.2 User Control Pad

The pad has debounced physical buttons for selecting preset configurations in Supercollider and generating user input signals. Each button maps to a specific preset, though more functionality could be added in the future. We elected to limit the button functionality to just selecting a specific preset to maximize performer efficiency.

After a button press, digital voltage signals from the button are processed by the UI microcontroller. Then, the UI microcontroller sends control signals to the LCD display to change the displayed information. When a user selects a new configuration option or new preset using the physical buttons, the UI microcontroller sends a corresponding digital control signal to the master control board.

The button debouce circuit uses an RC filter and an inverting Schmitt trigger to debounce a pushbutton (see Figure 10). The values of R1, R2, and C1 are chosen to be 130 kΩ, 22 kΩ, and 1 µF respectively, by consulting [4]; these values create a capacitor charging time constant of 20 ms when the button is open. The R1, R2, and C1 values are calculated to debounce buttons with the use of a 7414 hex inverting Schmitt trigger.

After prototyping the button debounce circuit, we found that the R1, R2, and C1 values shown above did not work. The input node to the inverter was around 1.0 V, which registered as a logic HIGH to the inverter. To bring the node voltage lower, we changed the R1, R2, and C1 values to 100 kΩ, 2.2 kΩ, and 1 µF respectively. Afterwards, the debounce circuit worked.

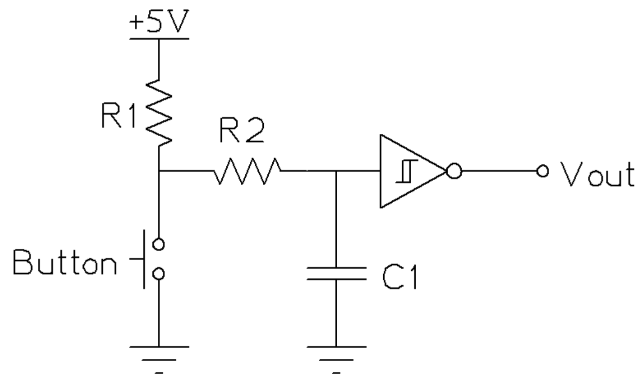

Figure 10: RC debounce circuit for pushbuttons.

2.3.1.3 User Interface Microcontroller
The UI microcontroller controls the LCD display board based on user input from the User Control Pad. It sends user preset data to and receives error codes from the master control board. It outputs error messages to the LCD display if an error occurs. The microcontroller also uses the status LED to indicate if a connection to the master board has been established.

2.3.1.4 User Interface Status LED
On power up, this LED will blink to indicate the board has power, then once communication has been established with the master board, it turns off. The LED stays off during operation to avoid distracting people near the music board and generating interference. In the event a fault occurs (ex. The board detects an open circuit, or the board loses connection to master board), the LED will begin blinking rapidly.

2.3.2 Master Control Board
2.3.2.1 Master Board Microcontroller
This microcontroller pre-processes and packages data for serial transmission to a computer. It sends integer values via serial as the four-digit ASCII representation of the number. It reads photoresistor data and error codes from the slave boards. It also reads data from the user interface board and sends error data to the user interface board.

2.3.2.2 Serial I/O
Relays data from the photoresistors and user control pad to a computer via a TTL to FTDI serial connection.

2.3.2.3 Master Board Status LED
This LED indicates when the master board connects to the slave board subsystem. The LED turns off during board operation. It indicates if there a communication failure occurs between the master board and any of the slave boards or the user interface board by blinking rapidly.

2.3 Photoresistor Slave Boards

2.3.3.1 Photoresistor Array

The photoresistor array detects the light intensity of incident light. Each array will contain 16 photoresistors arranged in a 4-by-4 grid with 1-inch spacing between each photoresistor. Each photoresistor connects in series with a 4.7 kΩ resistor to form a voltage divider circuit. The board measures the voltage across the 4.7 kΩ resistor. See figure 11.

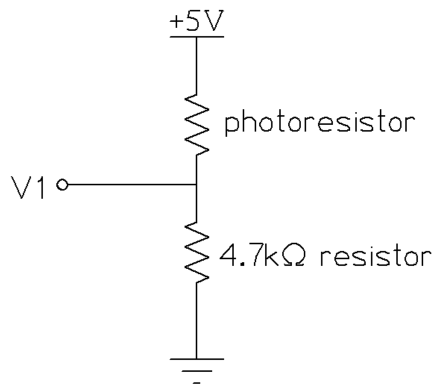

Figure 11: Photoresistor Circuit

Various series resistor values were considered. Table 1 shows the voltage drop across the series resistor for various resistance values.

Table 1: Output Voltage vs. Series Resistance

Series Resistor Value	Low Light V1 (V)	Normal Light V1 (V1)	High Light V1 (V)
1.0 kΩ	0.97	2.82	4.73
2.2 kΩ	2.0	3.75	4.5
3.3 kΩ	2.30	4.02	4.67
4.7 kΩ	2.8	4.35	4.8
10 kΩ	3.50	4.65	4.87
22 kΩ	4.23	4.80	4.92

The lower series resistance values provide high sensitivity to both low-light and high-light conditions. However, this resulted in increased power consumption. Resistances above 5 kΩ had very poor sensitivity to high-light conditions – with almost identical normal and high-light voltages. We determined that a series resistance of 4.7 kΩ provided an acceptable balance between sensitivity and power consumption.

The 256 photoresistor circuits should consume at most 1 W of power, which corresponds to 4 mW per photoresistor circuit. This leaves 4 W of power for the microcontrollers and LCD screen to use from the 5 W power supply. See Section 3.3.1 for verification of the photoresistor array power consumption.

2.3.3.2 Slave Board Microcontroller

The PIC32 microcontroller on each slave board controls the multiplexer to read the photoresistor voltage divider circuits. The microcontroller stores the photoresistor data by selecting a specific multiplexor input and reading the analog output of the multiplexor. It acts as the supervisor board for slave boards connected to it and reads the data these slave boards. It then compiles its own data and the slave board data into a single data array. The board microcontroller also responds to supervisor requests by sending all stored data and an error code. Lastly, it sends control signals to status LED.

2.3.3.3 Multiplexer

The analog 16-to-1 multiplexer reads the analog voltage from each of the 16 photoresistors and send the data to the board's microcontroller.

2.3.3.4 Slave Board Status LED

This LED indicates that a working connection has been established between this board and a supervisor board. On power up, this LED blinks to indicate the board is receiving power. Once communication has been established with a supervisor board, the LED turns off. The LED stays off during operation to minimize interference. In the event a fault occurs (examples: an open circuit is detected, connection to master board lost), the LED blinks rapidly until the problem is resolved.

2.3.4 Power Management

2.3.4.1 AC/DC Converter

A commercially available AC to DC power converter that plugs into a standard US wall outlet powers the music board. It converts 120V AC to 5V DC to power the ATmega32 [5]. The ATmega32 requires a supply voltage between 4.5 and 5.5 V.

According to Figure 149 on Page 297 of the ATmega32 datasheet [5], the master microcontroller typically consumes 23 mA current when operating at 16 MHz with a 5 V supply voltage. This corresponds to a power draw of 115 mW.

We ended up using the ATmega328PB chip instead of the ATmega32 chip. However, the difference in power consumption between both chips is too small to make a difference in the total power consumption of the board.

According to Page 208 of the PIC32 datasheet [6], the PIC32 microcontroller draws a maximum current of 9.2 mA when operating at 25 MHz with a 3.3 V supply voltage. Each PIC32 consumes 30.4 mW of power. There are 17 PIC32 microcontrollers, corresponding to a total power consumption of 520 mW.

Each photoresistor circuit on the music board will draw a maximum power of 4 mW (see section 2.3.3.1). For 256 photoresistor circuits, the total power consumption is 1055 mW.

The total predicted power consumption of the microcontrollers and photoresistor circuits is 1.70 W. To create a large margin for error, the power converter should supply more than double that power (3.5 W). We chose an AC to DC power rated for 5 W.

2.3.4.2 DC/DC Converter
This block consists of multiple 5 V to 3.3 V linear regulators, one for each slave board and for the user interface board. Each regulator must output a voltage between 3.0 and 3.6 V and at least 100 mA of current to power a single PIC32 microcontroller. We chose the LP5907 voltage regulator made by Texas Instruments [7] for its adequate 250 mA of current output.

2.4 Board to Board Communication Protocol Design
We originally planned to use an existing digital communication protocol to transmit data between our different boards, however we could not find a program compatible with the hardware we selected for our project. We decided to design and implement our own, though when we implemented it we found that the algorithm was too slow for our purposes.

Using the I^2C protocol as inspiration, our protocol uses two digital data lines to send information in one direction. A supervisor board reads data by generating a clock signal that a slave board reads. Based on the input clock signal, the slave board writes a single bit to a data output line. See Figure 12.

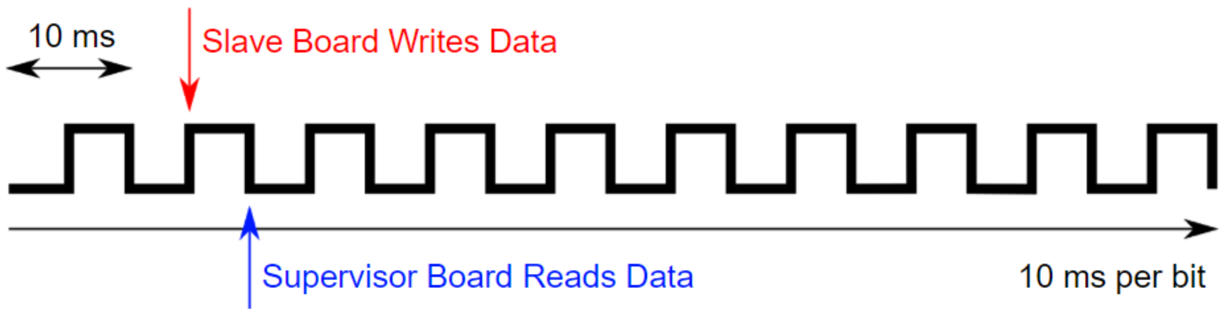

Figure 12: Visual explanation of our communication protocol indicating when the supervisor board reads the data (blue arrow) and when the slave board writes data (red arrow) based on the clock signal (black) generated by the supervisor board. In testing, we managed to transmit data with this algorithm at 1 bit per 10 milliseconds.

We programmed two versions of this protocol, one for communication between the master board and the user interface board, and another for communication from each slave board up to a supervisor board. The master board and UI board communication protocol included transmitted an error code from the master board to the UI board and a preset value from the UI board to the master board. To achieve this, we wrote a program that transmitted four bits from the UI board to the master, then transmitted four bits from the master board to the UI board. For slave board communication, the slave boards send an end of transmission signal along with an error code signal after transmitting all the data at a rate controlled by the supervisor board.

3. Design Requirements and Verification

3.1 User Interface

3.1.1 LCD Screen

Table 5 shows the requirements for the LCD screen. During implementation, we were not able to use the PIC32 microcontroller or Arduino development board to display text on the LCD screen. The SBN1661G digital controller on the screen required us to write our own software library, but time constraints forced us to switch to an alternative display.

We satisfied the requirements for the LCD screen using an OLED display controlled by an Arduino microcontroller. The OLED display was powered on at all times and refreshed with no discernable delay (well above 10 Hz). When the user selected a preset or an error occurred, the screen displayed the appropriate preset selected message or error messages respectively.

3.1.2 User Control Pad

Table 6 shows the requirements for the user control pad. The design met both requirements.

We used an oscilloscope to measure the debounced button signals. The signals showed very fast rising edges and falling edges when we pressed and released the buttons, respectively. Even when pressing a button up to 8 times a second, the output signal showed signs of proper debouncing.

3.1.3 User Interface Microcontroller

Table 7 shows the requirements for the user interface microcontroller. Our implementation did not meet the requirements. Instead, the requirements were partially met using an Arduino.

Requirement 1 was not met. When buttons were pressed on the user control pad, the UI microcontroller successfully sent the button data to the master microcontroller. However, the I^2C style communication protocol was too slow. The delay between pressing the button and the data being sent to the computer was larger than 500 ms. With a faster communication protocol, this requirement would be satisfied.

Requirement 2 was met. The microcontroller displayed which presets were selected onto the OLED display and sent the button data to the master microcontroller.

3.1.4 User Interface Status LEDs

Table 8 shows the requirements for the user interface status LED. The design did not meet these requirements. The User Interface printed circuit board did not feature an LED circuit, and a lack of digital pins on the Arduino prevented the user interface LED from begin implemented.

3.2 Master Control Board

3.2.1 Master Microcontroller

Table 9 shows the requirements for the master microcontroller.

For Requirement 1, we did not test the music board with the Supercollider programs, so we could not test interrupts from SuperCollider.

The design partially met Requirement 2. The recursive algorithm for reading photoresistor data was successfully implemented. However, the read time for a single slave board far exceeded the requirement threshold. The time to read all sixteen slave boards exceeded 16 seconds, much higher than the 20 ms threshold listed in the requirement. With a faster data transmission protocol, the design would meet this requirement.

The design did meet Requirement 3. When we generate a connection error between a slave board and the master board, an error message is transmitted to the computer and to the user interface microcontroller. The error transmission happens within an I^2C clock cycle, which is around 10 ms long.

3.2.2 Serial Communication

Table 10 shows the requirements for serial communication. Both requirements were met. We were able to reprogram the Atmega328 microcontroller with an In-Circuit Programmer connection after soldering the microcontroller to the circuit board. Also, the board transmitted photoresistor data and user interface data to a computer using a USB to serial connection.

3.2.3 Master Board Status LED

Table 11 shows the requirements for this system, which were all met. When a slave board disconnected from the master board, the status LED blinked to indicate the error.

3.3 Photoresistor Slave Boards

3.3.1 Photoresistor Array

Table 12 shows the requirements for the photoresistor array. The design met all requirements.

The array measures light intensity with the photoresistors. Table 2 shows the resistance and output voltage of different lighting conditions. We used a series resistor with a resistance of 4.7 kΩ.

Table 2: Photoresistor Resistance and Output Voltage

Lighting	Photoresistor Resistance	V1 Node
Direct Light	0 Ω - 1 kΩ	> 4.5 V
Normal Light	~ 2 kΩ	~ 3.5 V
No Light (covered photocell)	> 50 kΩ	< 2.5 V

We calculated the power consumption of each voltage divider circuit with equation (1).

$$P = \frac{(5\,V)^2}{R_{photoresistor} + 4.7\,k\Omega} \tag{1}$$

Table 3 shows the power consumption for different lighting conditions.

Table 3: Photoresistor Circuit Power Consumption

Lighting	Photoresistor Resistance	P (mW)
Direct Light	0 Ω to 1 kΩ	5.3 to 4.4
Normal Light	2 kΩ	3.7
No Light (covered photocell)	50 kΩ	0.46

Each photoresistor circuit used less than 4 mW of power, except in the direct light case. This does not meet the requirements in all cases. Powering 256 photoresistor circuits, each consuming 5.3 mW of power, is a total of 1.36 W of power. This is well within the 5 W of available power from the power supply block; the rest of the circuitry does not need more than 2 W of power. Although use of the 4.7 kΩ resistor does not meet this requirement, it does not compromise the operation of the music board.

3.3.2 Slave Board Microcontroller

Table 13 shows the requirements for the slave board microcontroller. The design met all the requirements.

During testing, we were not able to accurately read the photoresistor data using a PIC32 microcontroller. This is because the PIC32 uses an analog to digital converter with a 3.3 V reference; the PIC32 cannot detect voltages above 3.3 V. However, the photoresistor circuits use a 5 V supply, and normal lighting conditions create voltages above 3.3 V. This design issue makes it impossible to achieve accurate readings with the PIC32.

When we used an Arduino microcontroller to read the photoresistor data, it could detect the full range of values and process the data into an integer array.

The slave board microcontroller could detect interrupts from other slave boards. However, the time delay was greater than 10 ms.

3.3.3 Multiplexer

Table 14 shows the requirements for the multiplexer. The design met all the requirements because the slave board microcontroller could use the multiplexer to read data values. We tested this by covering up specific photoresistors and checking that the correct value read in changed.

3.3.4 Slave Board Status LED

Table 15 shows the requirements for the slave board status LED. The design met all the requirements during operation.

3.4 Power Supply

3.4.1 AC/DC Converter

Table 16 shows the requirements for the AC to DC converter. The design met these requirements.

The converter uses standard 110 V AC from wall sockets and converts it to 5 V DC. We tested this with a digital multimeter. Input voltages above 130 V AC were not tested. The max power output of the converter was also not tested However, since the converter is a commercial product, we assumed that the converter could accept input voltages between 100 V and 240 V AC and output between 3.5 and 5 W of power

3.4.2 DC/DC Converter

Table 17 shows the requirements for the DC to DC converter. The design met all its requirements.

To test the output current of the voltage regulator, a 30 Ω resistor was connected between its output and ground. We measured the voltage across the resistor at 3.3 V. Therefore, the output current was around 110 mA, which satisfies the requirement.

3.5 Board to Board Communication

3.5.1 Slave Board to Supervisor Board

The speed requirement implicit in high level requirement for operating the master board read cycle at 1000 Hz was not met, but data could be transmitted accurately.

To verify that data could be transmitted accurately with our protocol, we set the data values of the slave board to just be incrementing numbers and transmitted the data. The data that the supervisor board read in was transmitted through a serial interface and displayed on a computer. We determined the max speed that the protocol could transmit data by continually decreasing the clock duration and checking that the supervisor board could still read data accurately. The shortest clock duration we were able to use in testing was 10 milliseconds, far slower than our design constraint.

3.5.2 Master Board to User Interface Board

The speed requirement implicit in high level requirement for operating the master board read cycle at 1000 Hz was not met, but data could be transmitted accurately.

To test that data transmission accuracy, we set different values for the user preset value and flag value and printed the data read in by each board through the serial connection of the master board and on the user interface display, respectively. At a clock rate greater than 10 milliseconds, the boards accurately sent and received data. This does not meet the speed requirements of our design.

4. Costs

4.1 Parts

Table 4 shows the list of parts and cost of parts.

Table 4: Parts Costs

Part	Manufacturer	Quantity	Unit Cost ($)	Total Cost ($)
LCD Screen	Crystalfontz	1	$13.02	$13.02
10 kΩ potentiometer	Suntan	1	$0.95	$0.95
16 MHz crystal	Citizen Finedevice	18	$0.45	$8.10
ATMEGA328PB	Atmel	1	$1.51	$1.51
PIC32MM0064GPL020-I/SS	Microchip	17	$1.41	$23.97
CD74HC 16-1 mux	Texas Instruments	16	$0.775	$12.40
LP5907MFX voltage regulator	Texas Instruments	17	$0.475	$8.08
AC to DC power converter	XP Power	1	$6.50	$6.50
2.1mm DC Jack	CUI, Inc.	1	$0.60	$0.60
Photoresistor	Luna Optoelectronics	256	$0.748	$191.49
4.7 kΩ ¼ W resistor	Stackpole Electronics or other manufacturer	256	Est. $0.02	Est. $5.12
18 pF 1206 SMD capacitor	Samsung Electro-Mechanics	36	$0.12	$4.32
0.1 uF 1206 SMD capacitor	Samsung Electro-Mechanics	20	$0.08	$1.60
1 uF 1206 SMD capacitor	Yageo	34	$0.106	$3.60
10 uF 1206 SMD capacitor	Samsung Electro-Mechanics	17	$0.16	$2.72
10 kΩ 1206 SMD resistor	Yageo	18	$0.03	$0.54
2.2 kΩ 0603 SMD resistor	Yageo	6	$0.011	$0.07
100 kΩ 0603 SMD resistor	Yageo	6	$0.011	$0.07
1 uF 0603 SMD capacitor	Samsung Electro-Mechanics	8	$0.036	$0.29
Header pins and wires				Est. $10
			Grand Total	**$294.95**

4.2 Labor

Engineering Design labor costs are billed out at a higher rate. We spent engineering hours working on designing the PCB, writing code, creating design docs, and other design related work. Not every week is an engineering design period, and we estimate about 9 weeks (2 for initial design, 3 for system design, 2 for PCB testing, 2 for documenting project at the end of the semester) were spent on design.

Engineering Design:

$$2 \cdot \frac{\$45}{hour} \cdot 10\,\frac{hours}{week} \cdot 9\ weeks \cdot 2.5 = \$20{,}250 \tag{2}$$

Soldering labor accounts for time spent by group members soldering components to the PCBs. We spent around one week soldering components onto the 18 boards.

Soldering Hours:

$$2 \cdot \frac{\$20}{hour} \cdot 25\,\frac{hours}{week} \cdot 1\ week \cdot 2.5 = \$2{,}500 \tag{3}$$

Total Labor Cost: $22,750

4.3 Total Cost

Cost of parts: $294.95

Cost of labor: $22,750

Total cost: $23,044.95

5. Conclusion

5.1 Accomplishments

Through the course of this project, we designed 3 separate functioning PCB designs. Our master board design implemented an ATMega328PB microcontroller with an In-Circuit Programming interface and a TTL Serial to FTDI interface. The master board also interfaced with our commercial AC to DC power converter and propagated this power to other boards. Our photoresistor slave boards implemented 16 voltage divider circuits and incorporated a 16-to-1 multiplexer. The slave board microcontroller was powered using a 3.3 V linear regulator. Finally, our user interface board included six properly debounced push buttons and a functioning microcontroller.

We also designed and implemented our own board to board digital communication protocol based on I^2C protocols. This enabled our boards to communicate data packets with each other. We also successfully connected our master board to a computer via serial and transmitted data from the entire subsystem through this serial connection.

5.2 Uncertainties and Unaccomplished Goals

Our first major decision issue arose when we attempted to implement our slave board microcontroller code on the PIC32 microprocessors. While selecting a microprocessor for the slave boards, we decided to choose the lowest power 32-bit microprocessor available. To limit the power consumption, we decided to use a 3.3V chip rather than a more conventional 5V chip that would consume more power. Unfortunately, we overlooked the fact that a microprocessor with internal analog to digital converters (like the PIC32) use their supply voltage as a reference voltage. This created an issue since we used a 5V supply for our voltage divider circuits on our slave boards. This caused the microprocessor to constantly max out while trying to read our voltage divider circuits, making the board completely unable to sense normal lighting conditions and brighter lights.

Additionally, our board to board digital communication protocol failed to meet our design specifications. We had hoped to achieve a full master board read speed of about 20 Hz, but since the board to board communication protocol worked 1000 times slower than we expected, we were only able to achieve a master board read speed of 0.038 Hz. This rate is much too slow for human interaction.

We also ran into problems using the LCD screen originally selected for our design. This screen lacked the prebuilt libraries that we had planned to use to write characters to the screen. Therefore, we decided to use a better supported OLED display for our final prototype.

5.3 Ethical considerations

Throughout our work on this project, we sought to uphold the IEEE Code of Ethics [8].

An important ethical issue in the field of music is plagiarism, which falls under number seven in the Code of Ethics (“credit properly the contributions of others” [8]). Anyone can perform plagiarism when composing a piece of music and play the plagiarized music on the music board. We will not include a method to detect and disallow plagiarism within the music board. Since analyzing a piece of music for plagiarized sections is difficult, time-consuming, and often a matter of interpretation and intent; it is not feasible to include such a solution within a musical instrument.

Number one in the Code of Ethics also states to “hold paramount the safety, health, and welfare of the public” [8]. The project is associated with the following safety concerns.

By creating a tabletop instrument with exposed circuitry that requires users to interact with at close distances creates risk of injury. To limit contact with the circuit, we included a clear piece of plastic cover over the top of the music board to prevent objects and people from directly touching the electronic components. This solution reduces the chances of an individual receiving an electric shock or a cut.

Also, the music board uses a wall adapter to receive power. The power supply must follow strict safety standards to protect people and equipment from AC voltage. To protect people and devices from electric shocks, we used a commercial AC to DC power converters that follow UL standards [9] and is RoHS compliant. Besides the power supply, we also designed and built the power adapter section to safely supply the music board with power. We also measured the power draw of the music board and ensured we did not exceed our 5 W design limit.

All in all, we not only designed a functional instrument, but also to created board that is safe for anyone to interact with.

5.4 Future work

We have already designed and ordered a new version of our printed circuit board. Improvements can also be made to increase the speed of our board communication algorithm and increase the functionality of the user interface.

The next iteration of our design features a simplified design. To eliminate the data propagation delays outlined above, we have decided to eliminate the microcontrollers from all but the master board. Instead of preprocessing the data on the photoresistor slave boards, all the data will be read by the master control board. To accommodate this change, we have included a new 16-to-1 multiplexer on the master board to enable serial reading of each slave board one at a time. We believe this will decrease our data reading loop latency and allow us to poll data at a rate greater than 20 Hz.

References

[1] "SuperCollider Github page", SuperCollider website. [Online]. Available: https://supercollider.github.io/. [Accessed: 23- Feb- 2018].

[2] "Eli Fieldsteel", Music at Illinois, 2018. [Online]. Available: https://music.illinois.edu/faculty/eli-fieldsteel. [Accessed: 23- Feb- 2018].

[3] "122x32 Transflective Graphic LCD (CFAG12232DYYHVA)", Crystalfontz.com. [Online]. Available: https://www.crystalfontz.com/product/cfag12232dyyhva-transflective-graphic-lcd-122x32. [Accessed: 03- May- 2018].

[4] J. Ganssle, A Guide to Debouncing, 3rd ed. Baltimore, MD: The Ganssle Group, 2004, pp. 12-16. Available: http://www.eng.utah.edu/~cs5780/debouncing.pdf. [Accessed: 23- Feb- 2018].

[5] "ATmega32/L Datasheet", Microchip, 2018. [Online]. Available: http://ww1.microchip.com/downloads/en/DeviceDoc/doc2503.pdf. [Accessed: 23- Feb- 2018].

[6] "PIC32MM0064GPL036 Family Datasheet", Microchip, 2016. [Online]. Available: http://ww1.microchip.com/downloads/en/DeviceDoc/60001324b.pdf. [Accessed: 23- Feb- 2018].

[7] "LP5907 250-mA Ultra-Low-Noise, Low-IQ LDO datasheet (Rev. M)", TI.com, 2012. [Online]. Available: http://www.ti.com/lit/ds/symlink/lp5907.pdf. [Accessed: 23- Feb- 2018].

[8] "IEEE Code of Ethics", IEEE.org, 2018. [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html. [Accessed: 23- Feb- 2018].

[9] "UL Standards", UL Standards, 2018. [Online]. Available: https://ulstandards.ul.com/. [Accessed: 23- Feb- 2018].

Appendix A Requirement and Verification Tables

Table 5: LCD Screen Requirements and Verification

Requirements	Verifications	Verification status (Y or N)
Must be powered on and display information at all times when the music board is on.	1. Connect the master board to the UI Board 2. Test that microcontroller can print data to LCD screen using diagnostic mode.	Y
Refresh display at a frequency of at least 10 Hz when a new preset is selected or when a photoresistor slave board reports an error	1. Initialize diagnostic tests. 2. Using the internal clock, refresh the screen incrementing a value on screen by 1 to ensure operation at at least 10 Hz.	Y
Display system diagnostic messages pushed for other boards in system. Diagnostic messages will be pushed from the master board mapped to strings by an array of possible interrupt values and one byte of data.	1. Build test rig that on an input signal (button) generates and error code 2. Let error message recurse through system. 3. Check that the proper error data is displayed on UI board. 4. In final system, build an incomplete slave board (missing one or more photoresistors). 5. Check that the proper error data is displayed on UI board.	Y

Table 6: User Control Pad Requirements and Verification

Requirements	Verifications	Verification status (Y or N)

All buttons should be debounced: pressing a button down should not result in multiple rising or falling edges in the voltage across the button.	1. Build debouncer circuit. Connect the button to the circuit and the output of the button to a microcontroller analog input. 2. Connect the positive terminal of an oscilloscope probe to the Schmitt trigger output (Vout). Connect the ground terminal to ground. 3. Press the pushbutton down repeatedly. Ensure that the button is debounced.	Y
Enable users to select desired preset within 5 seconds to allow changing the configuration.	1. After building UI menus, give test users a brief explanation of the interface. 2. Ask them to navigate to a specific preset and time the duration of their first attempt. 3. Record the time and repeat tests 3 more times determining the benefit of familiarity with the system.	Y

Table 7: User Interface Microcontroller Requirements and Verification

Requirements	Verifications	Verification status (Y or N)
Appear to have immediate response to user requests and send data to Supercollider (main computer) within 500 milliseconds.	1. Initialize diagnostic mode. 2. Test transmission of data from UI board, through master control to computer then back (through a series of interrupt signals). 3. Use the UI board's internal clock to determine time elapsed. 4. Since the path travelled in the test case is twice as long as the requirement, the response time should be less than 1000ms.	N

Interface with the master control board to communicate any user interface selections and receive system diagnostic information.	1. Initialize Diagnostic mode. 2. Test print values through serial interface to check that UI preset selections are encoded with data.	Y

Table 8: User Interface Status LED Requirements and Verification

Requirements	Verifications	Verification status (Y or N)
Respond within 100 milliseconds when errors occur on this board.	1. Short one pin of the user control pad. 2. Status LED should start flashing red without any discernible delay.	N
Turn off when board is in use.	1. Once system is in operating mode (all slave boards are connected to master boards, and master board is communicating with Supercollider), check that the LED is off.	N

Table 9: Master Microcontroller Requirements and Verification

Requirements	Verifications	Verification status (Y or N)
Respond within 10 milliseconds of serial interrupts from supercollider.	1. Initialize serial diagnostic mode. 2. Run test program that uses Supercollider timing to check that Supercollider receives a response to an interrupt to master controller in less than 10 milliseconds.	N
Generate and control recursive reading algorithm (see section 2.5.4) such that a single read cycle takes less than 20 milliseconds of processing time.	1. Send serial interrupt to master control board. 2. Using serial terminal, print entire master board data transmission and check that all values are correct.	Y

	(should be all zeros if no breakout boards are connected). 3. Test complete system with Eli's supercollider code and it should generate music similar to the prototype.	
Transmit error and diagnostic messages to user interface within 10 milliseconds of error occuring. Translate 2 byte error (1 byte for code and 1 byte for data) to string message.	1. Generate test error on slave board and check that proper information is displayed on UI board. (error board number, error type) 2. Use the master control board microcontroller to time period from receiving interrupt from slave to transmitting error to UI. 3. Print time elapsed to serial.	Y

Table 10: Serial Connection Requirements and Verification

Requirements	Verifications	Verification status (Y or N)
Communicate through USB serial to load new programs to the microcontroller.	1. Use Arduino IDE to load new text program that just prints "hello" to the serial terminal to show that new code can be loaded.	Y
Transmit photoresistor data and user interface data.	1. Set up and power on the board with at least one slave board attached. 2. Use a serial monitor to send a serial interrupt and ensure that the data read back is non-zero.	Y

Table 11: Master Board Status LED Requirements and Verification

Requirements	Verifications	Verification status (Y or N)

Receive interrupts for breakout boards and receive data through digital pins.	1. Set up and power on the board with at least one slave board attached. 2. Use a serial monitor to send a serial interrupt and ensure that the data read back is non-zero.	Y
If there is an error with the master control board, this LED should blink, otherwise it should be off.	1. Set the system to diagnostic mode. 2. Send an error serial interrupt from the PC. LED should blink 3. Resume normal operation, LED should turn off to limit interference.	Y

Table 12: Photoresistor Array Requirements and Verification

Requirements	Verifications	Verification status (Y or N)
Measure light intensity using 16 photoresistors.	1. On each slave board, check that the photoresistor circuit is connected to supply voltage and ground.	Y
Draw at most 4 mW of power per photoresistor circuit connected.	1. Check that the photoresistor circuit is connected to supply voltage and ground using a multimeter. 2. Using a multimeter, measure the voltage at node V1. 3. Calculate the DC current passing through both resistors. 4. Use the current to calculate the power consumption of the photoresistor circuit.	Y

Table 13: Slave Board Microcontroller Requirements and Verification

Requirements	Verifications	Verification status (Y or N)
Connect 16 photoresistor circuits to a 16 input multiplexer and process the data into an integer array.	1. Using a multimeter to measure resistance, check that node V1 (see figure 2.3.2) photoresistor circuit are connected to the 16 multiplexer inputs and that the multiplexer output node is connected to the board microcontroller.	Y
Respond to interrupt signals from slave board within 10 ms and read packaged data from slave boards.	1. Initialize system and set to diagnostic. 2. Simulate interrupts from lowest slave boards and light LED when interrupt is received. 3. On each slave, time duration from start to receiving response from supervisor and push all data as error codes. 4. Print duration values to terminal.	Y

Table 14: Multiplexer Requirements and Verification

Requirements	Verifications	Verification status (Y or N)
Multiplexer must be connected to all 16 photoresistors on each slave board to allow microcontrollers to read photoresistor light intensity.	1. Use ruler to cover a single row of photoresistors at a time (similar to a test Eli did with his design). 2. Check that the appropriate values change. This ensures that the multiplexer connects the correct photoresistor to the microcontroller for each read cycle.	Y

Table 15: Slave Board Status LED Requirements and Verification

Requirements	Verifications	Verification status (Y or N)

Provide visual confirmation of connection to master control board.	1. Initialize Board system. 2. Check that LED blinks before going solid (indicating connection established) then turns off.	Y
Indicate presence of connection error within 10 milliseconds of the error occuring.	1. Set the system to diagnostic mode. 2. Short the connection of a single photoresistor, this should generate an error. 3. Without perceivable delay, the LED should light up to indicate an error.	Y
LEDs must remain off during normal board operation.	1. Use board for 10 minutes with at least 1 photoresistor slave board is connected to the master board and transmitting data. The LED should not turn on unless an error code is generated.	Y

Table 16: AC to DC Converter Requirements and Verification

Requirements	Verifications	Verification status (Y or N)
Accept input voltages between 100 V AC and 240 V AC from standard US wall outlets.	1. Connect the AC/DC converter to a variable transformer; connect the variable transformer to an AC wall socket. 2. Change the turns ratio of the variable transformer so that the wall adapter receives various input voltages between 100 V and 240 V AC. 3. Measure the open circuit output of the wall adapter with a multimeter or oscilloscope. The DC voltage should be within 4.5 to 5.5 V for any input voltage within the above range.	Y
Output voltages between 4.5 V DC and 5.5 V DC.	See verification process for the previous requirement.	Y
Outputs between 3.5 W and 5 W of power.	1. Connect the AC/DC converter to a powered AC wall socket.	Y

	2. Plug a 2.1mm barrel jack into a breadboard or protoboard rated for more than 1 A DC current. 3. Plug the converter's output plug into a 2.1mm barrel jack. 4. Connect a resistive load between 8Ω and 5Ω to the output pin of the barrel jack. These resistance values correspond to output power between 3.5 and 5 W. 5. Connect an ammeter in series with the load resistor. Measure the load current. Calculate the output power	
Outer casing temperature should remain stable under 65°C	1. While testing the above 3 requirements, use an IR thermometer to monitor the outer casing temperature.	Y

Table 17: DC to DC Converter Requirements and Verification

Requirements	Verifications	Verification status (Y or N)
Output voltage is between 3.0 and 3.6 V DC.	1. Connect the AC/DC converter output to the voltage regulator input pin. 2. Connect a resistive load above 30Ω to the voltage regulator output pin. 3. Measure the voltage across the resistive load with a multimeter or oscilloscope.	Y
Output at least 100 mA DC current.	1. Connect the AC/DC converter output to the voltage regulator input pin. 2. Connect a resistive load under 30Ω to the voltage regulator output pin. 3. Connect an ammeter in series with the resistive load to measure the load current. The load current should be greater than 100 mA.	Y

Outer casing temperature should remain stable under 65℃ under all conditions.	1. While testing the above requirements, use an IR thermometer to monitor the outer casing temperature.	Y