Current Affairs

Elvis Alvarez-Carcamo, Yuina Barzdukas, Chance M Woosley

Abstract

The Light Show, seen in Figure 1, uses InfraRed (IR) sensors to detect nearby objects or body parts and illuminate an LED array based on this detection. The IR sensors measure the level of light detected and nearby objects block light which results in lower output voltages. These voltage values are sent to a microcontroller (STM32 Nucleo-64) which analyzes the voltages to determine which LEDs should be turned on. This data is sent to an integrated circuit (IC) which selects the desired LEDs using a set of muxes. This project showcases a simple, cost-effective approach to proximity sensing and has potential applications in backup cameras, touchless light switches, and education.

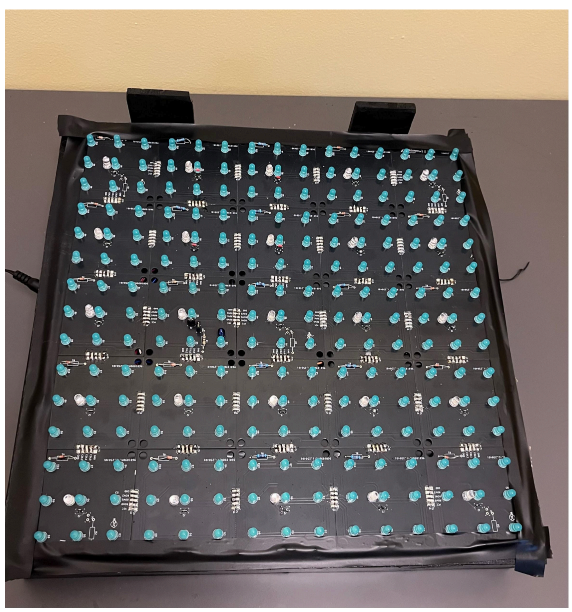

Figure 1: The Light Show Project

Hardware

As our capstone centers around LEDs, the primary hardware consideration is the LEDs we use to illuminate the screen. We decided on the 151051BS04000 diodes, which are blue through-hole LEDs [5]. These diodes were chosen because of their low price point and high supply on Digi-Key [5]. To properly turn on this LED, a forward voltage of 3.2 V and forward current of 20mA are required [5]. A generic wall power supply (SWI3-5-N-MUB) was used to power the design. This power supply was chosen as it is capable of supplying 5V DC and 3 Watts of continuous power, which is more than enough to supply all of our power consumption needs [2].

Our method of proximity detection used IR sensors, specifically the PT333-3C. This sensor is a phototransistor, which will increase the amount of current from its collector to emitter. Using a resistor, we can turn the current into a voltage, which a microcontroller can later read [4]. This phototransistor stood out from the rest since it's good at detecting wavelengths at 940 nm [4].

Our LED matrix required careful consideration to minimize current draw and control the screen's power consumption. An LED driver IC was initially considered for current and logic management which would have allowed full LED brightness control. However, the IC chosen needed complex synchronization and timing logic which were impractical for this design. Instead, additional shift register ICs were utilized to control the state of the LEDs. Although the shift registers worked, they were limited to a supply of 12mA per channel, while the LEDs required 20mA for full brightness according to the datasheet [3]. This meant that the LEDs were only operating at about half the current required for maximum brightness. Despite this limitation, the shift registers were functional and provided enough power to operate the LEDs within the project's time constraints.

Software: ***https://github.com/chancew7/LED_Matrix.git***

The software was written for the STM32 Nucleo-64 microcontroller using the STM32 cube IDE. The purpose of the software is to convert IR sensor voltages to output, dictating the state of the LED array. IR sensor data is read using the Analog to Digital Converter (ADC) of the STM32 device featured in the HAL library [1]. This sensor data is converted to a set of desired states of the LEDs. After this, output signals will be generated to map the desired states onto the actual LED array.

A DMA approach was used for the ADC reads. This designated an array for the storage of input values and enabled the ADC to continuously read input and write to the array once started. The main while loop of the code looped through each sensor value to determine whether an object was nearby or not. Once determined, the global arrays which stored the boolean row and column values were updated to represent the desired state of the LED array. This was done by mapping each LED to one of the sensors and turning that LED on when its corresponding sensor detected a nearby object and off otherwise. Finally, once the desired states were set, the outputs had to be created to communicate the states to the muxes. Four GPIO pins were initialized for the output: row_pin, col_pin, Sclk, and manualRclk. The code looped through every row and column of the array and wrote 1s or 0s corresponding to the desired led state. After every write, Sclk was set high to signal that these updated values should be stored. After every LED value was written, Rclk was set high which caused the LEDs to assume their stored values by turning on or off.

Mechanical:

The box that contains our product is made of wood and measures approximately 13 x 13 x 2 inches. The LED screen PCB rests on top of the case and is glued into place for stability. The LEDs are spaced 0.5 inches apart, and all wiring for the sensors and data transfer is enclosed inside the wooden case. One side of the box features a hole for the power connector, which is used to supply power to the PCBs and microcontroller. The STM32 is positioned approximately 0.5 inches beneath the PCB, allowing room for wiring and airflow. Wiring from the STM32 connects directly to the back of the LED screen, keeping all connections neatly tucked behind the display.

Future Work

In the future, we would purchase addressable LEDs, which allow full control over brightness and color, instead of using shift registers for LED control. Data can be sent to a chain of addressable LEDs which process their portion of the data and pass the remaining data to the next LED, all the way down the chain. This approach would provide much better control over the individual LEDs in our matrix, as each LED can be addressed and modified independently. Additionally, it would offer improved performance, as the shift registers we used were not designed for high-speed applications, and addressable LEDs are optimized for faster data transfer and more precise control. Overall, addressable LEDs would offer better scalability, more modular control, increased energy efficiency and simplicity in wiring and data transfer.

Future iterations of the project would also opt for a higher quality microcontroller than the STM32. One limitation we encountered with the STM32 was slow read speed, which affected the system's ability to handle the data from the 25 sensors in real time. The STM32 struggled to efficiently process and handle the sensor readings in parallel with other tasks, leading to delays in data acquisition and system performance. We would instead use the MSP430FR6989, which offers superior speed and more ADC inputs. According to the MSP430FR6989 datasheet, this microcontroller supports an 8-bit ADC with a conversion speed of up to 200 ksps (kilosamples per second), which is faster than the STM32 and would provide quicker sensor readings [6]. Additionally, the MSP430FR6989 is equipped with more ADC channels, allowing us to easily accommodate the 25 sensors in the design. This improved performance would eliminate the slowdowns we encountered with the STM32, ensuring faster data processing and a more efficient system [6].

References

[1] STMicroelectronics, *Description of STM32F4 HAL and low-layer drivers*, UM1725. [Online]. Available: https://www.st.com/resource/en/user_manual/um1725-description-of-stm32f4-hal-and-lowlayer-drivers-stmicroelectronics.pdf.

[2] DigiKey Electronics, *SWI3-5-N-MUB – AC/DC wall adapter 5V 0.6A 3W*. [Online]. Available: https://www.digikey.com/en/products/detail/cui-inc/SWI3-5-N-MUB/7784529.

[3] DigiKey Electronics, *CD74HC595E – 8-bit shift register 3-state output*. [Online]. Available: https://www.digikey.com/en/products/detail/texas-instruments/CD74HC595E/1507396.

[4] DigiKey Electronics, *PT333-3C – Phototransistor, through-hole, NPN*. [Online]. Available: https://www.digikey.in/en/products/detail/everlight-electronics-co-ltd/PT333-3C/2675648.

[5] Würth Elektronik, *151051BS04000 wire-to-board connector*. Digi-Key Electronics. [Online]. Available: https://www.digikey.com/en/products/detail/w%C3%BCrth-elektronik/151051BS04000/4490009. [Accessed: Mar. 4, 2025].

[6] Texas Instruments, "MSP430FR6989: 16-bit Microcontroller with 128KB FRAM and 25 ADC Inputs," *Texas Instruments*, 2021. [Online]. Available: https://www.ti.com/product/MSP430FR6989. [Accessed: May 9, 2025].