Rotating LED Display with Wireless Control

H.A.Z.Sameen Shahgir
1805053

Khondker Salman Sayeed
1805050

Asif Ihtemadul Haque
1805048

September 2022

1 Introduction

Spinning LED Display consists of a panel containing a micro-controller, rows of LEDs, a Bluetooth module, a proximity sensor and a clock module mounted on a motor at its center. The idea is, that panel will be rotated by the motor, and micro-controller will blink the LED rows in specific intervals giving the illusion of an LED display. Our project can display an analogue clock, a digital clock and any text received via Bluetooth. We use the proximity sensor as a guide to identify which rotation phase our panel is in. We also use a RTC clock module to keep time.

2 Diagrams

2.1 Circuit

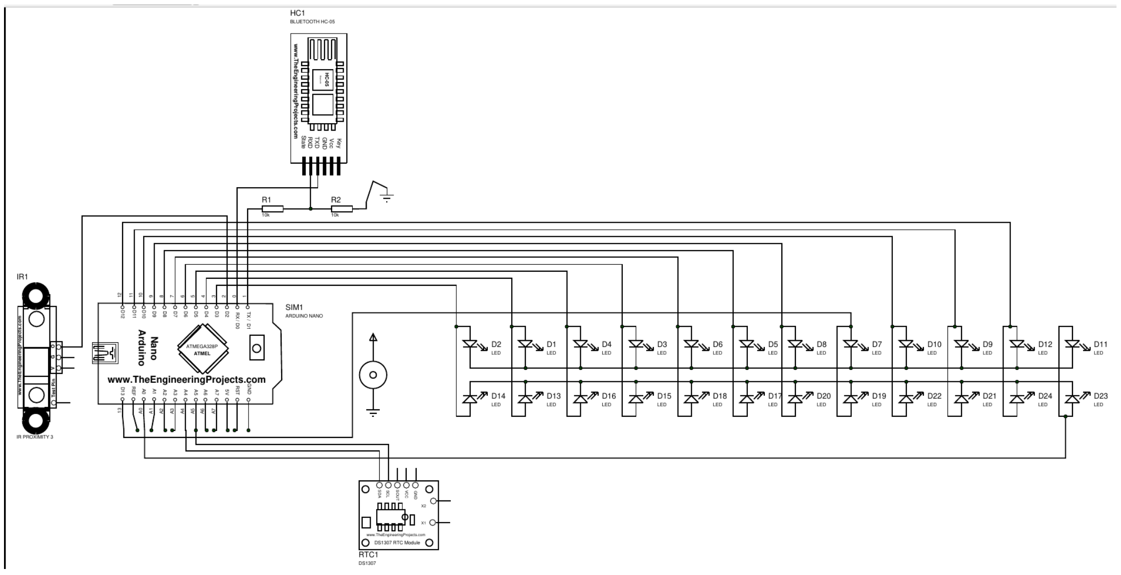

Figure 1: Circuit Diagram

2.2 Block

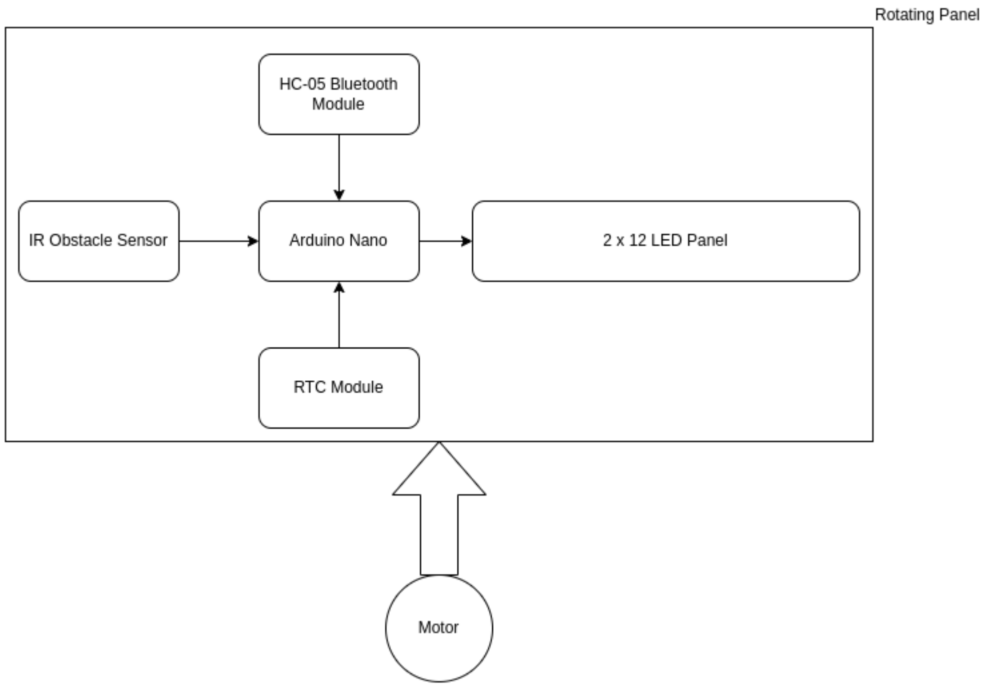

Figure 2: Block Diagram

3 Instruments

Instrument	Count
Arduino Nano	1
HC-05 Bluetooth Module	1
IR Obstacle Sensor	1
DS3231 RTC	1
LEDs	24
DC Motor	1
3.7V Lithium Ion Battery	1
Charging Circuit	1
5V Wall Charger	1

4 Display Procedure

The heart of our display was the synchronized blinking of LEDs that make letters, digits, clock arrows, ticks as the panel rotates. We found multiple codes on the internet where the shape of alphabets and digits were encoded as arrays of bits. As our panel would rotate, the LEDs would blink according to the these bit patterns, giving the shape of numbers, alphabets. So we wrote a procedure that blinks the LEDs using these bit patterns. However, the patterns were there - but the time between the display of each row of bit patterns, and the time between each letters became crucial for displaying the text correctly.

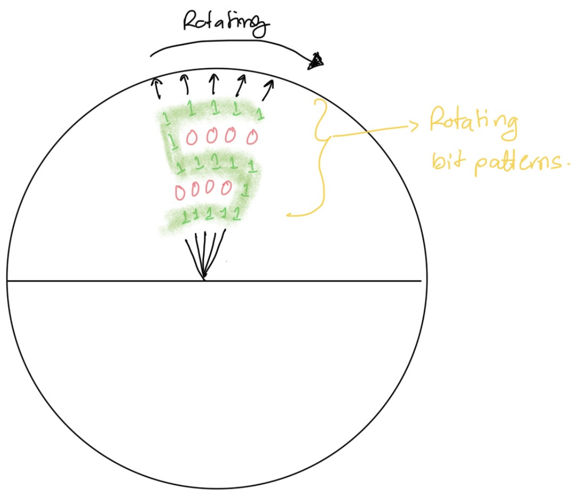

Figure 3: Displaying 'S' through blinking LEDS according to bit patterns

5 Problems and Solutions

5.1 Bluetooth Baud rate

The serial baud rate for our project had to be set to 38400, because upon multiple trials the usual value of 9600 was not working. We realized that setting the baud rate to 38400 actually made the arduino work at 9600, as observed from the serial monitor tool. This observation was repeated on a separate arduino nano. A probable reason for this could be some misrepresentation of the baud rates for cloned arduino nanos available in the markets.

5.2 Interrupt vs Timing

Our IR obstacle sensor provides us with the information to detect a cycle. We could use that information to synchronize our display in different ways. The most obvious way would be to calculate the average period of the IR sensor being, and call our *display procedure* from section 4 periodically using our calculated period. However, that approach turned our to be too sensitive to the period. If the period calculation was off by even the slightest, the display would continuously rotate, without staying steady.

We then decided that, the IR sensor should trigger an interrupt whenever it detects our *anchor* object, and that interrupt handler should perform the display on one cycle. After execution of that handler, the IR sensor again triggers an interrupt after completion of another cycle. It was crucial that the interrupt handler completed execution almost precisely within one cycle, never over a cycle. Otherwise the display would overlap, via multiple simultaneous interrupt handlers. Using this interrupt strategy, we were able to display in a steadily, without image rotation.

5.3 Letter Timing and Spacing

The bit patterns we had encoded the LEDs that needed to light up in each rotation angle. The time each column in rotation needs to be high enough to let the LED glow properly, low enough to make the character shape clear. The letter spacing time needs to be enough to separately identify each shape to be a character. We were able to set a suitable time period for our motor and power supply combination but that it liable to change if different components are used.

5.4 Phase Sensor

Initially we thought of using a Hall Sensor to synchronize the phase of our rotating panel. We planned to have a magnet on a point near which the panel rotates. When the panel is near that sensor, we would detect it in our microcontroller and know we have completed a cycle. How ever upon trying out a hall sensor we found out the signal it provides is jittery, and not as precise as our project demands.

So the next natural trial for this guiding module was the IR proximity sensor. We would have an object at the same point as that magnet, that would trigger some signal to the microcontroller. We have found IR obstacle sensor provides a satisfactory signal for our application.

5.5 Analogue Clock Arrows

The analogue clock must be aware of the current rotation angle to properly display hour, minute and second hands. This is done via keeping a counter variable, that increments after each $\frac{1}{60} - th$ of a rotation. This way, the hour hand is lit up when the variable increments to the desired hour position in the clock. The same idea is true for minute and second hands.

5.6 Interrupt and Serial Communication

Since our display procedure is entirely interrupt based, we could not have it be interfered by the received bluetooth string. Therefore it was necessary for us to detach the interrupt procedure when we have some data on our serial ports, and re-attach it after we have successfully read the incoming bluetooth data.

6 Conclusion

In this project, we made a rotating LED display based around an Arduino Nano, making use of several components such as Bluetooth and RTC. The most challenging aspects of this project were setting the correct timing for the LEDs,the balancing of the moving component over the motor and debugging the baud rate issue. Overall, the project was not overtly challenging and we were able to implement almost all the features we set out to.