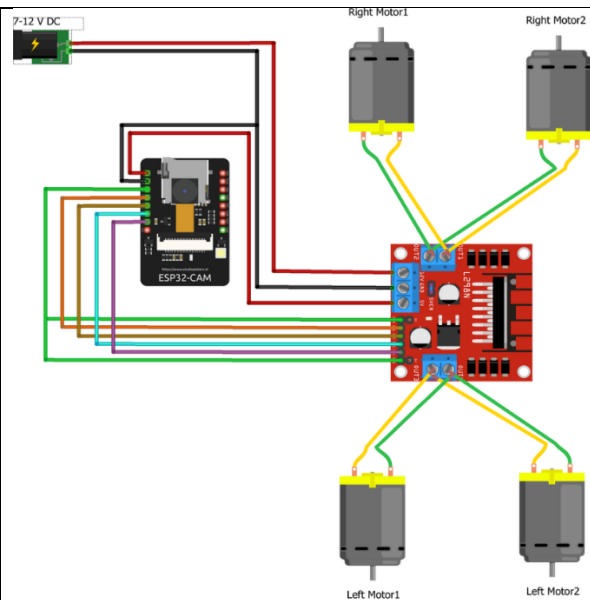
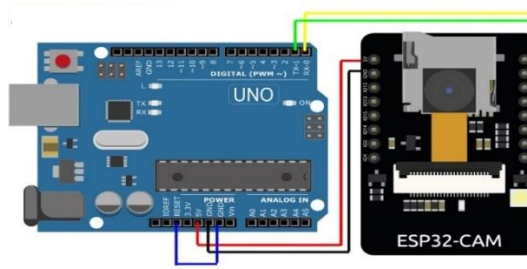


Title: ESP32 Controlled Surveillance RC Car

Motivation: To get familiarized with ESP32 Microcontrollers

Aim/Objective: To remotely survey non-accessible areas through wireless means such as Wi-Fi or Bluetooth through an in-built camera module

Circuit Diagram:



Components Required: 4 wheels, 4 TT Gear motors, Arduino UNO, ESP32 Cam module, Chassis, L298N motor driver, 2 3.3V Batteries and their holders, Screwdrivers, Jumper wires, tapes, Glue, Soldering materials.

Product Designed:



Advantage: i) Easily accessible due to real time data accessing techniques
ii) Connect to a mobile phone to send the data through Wi-Fi
iii) Connects through a custom IP Address

Limitations: Only available for communication up to 300 feet distances.

What I have Learnt: i) To program a ESP32 Cam module using Arduino UNO board.
ii) To create a custom IP Address for connecting both devices.
iii) Voltage limitations of specific devices
iv) How to work in a team
v) Improved Communication Skills

Basic Principles Used:

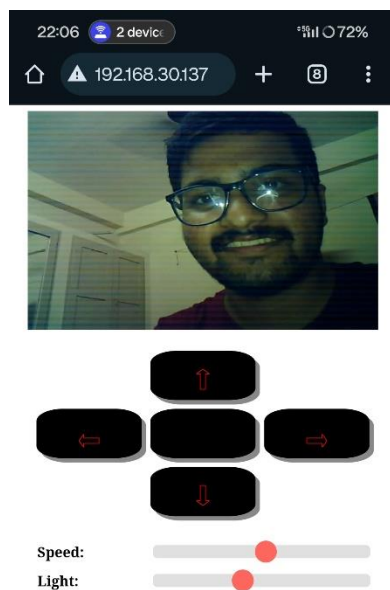
4MB PSRAM (pseudostatic random-access memory) of ESP32 Cam is used for image buffering from the camera into video streaming and permits to utilize higher quality within the images without any crash

Potential Application:

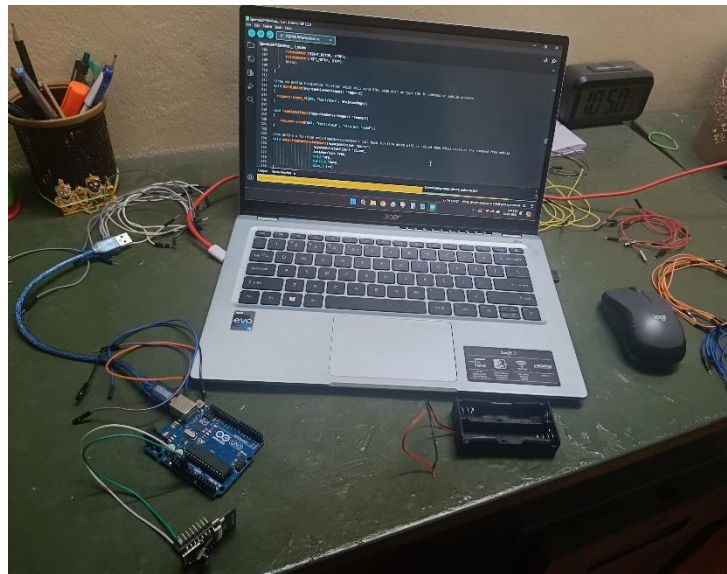
- i) For defence organisations, to use a familiar technology in order to carry out surveillances
- ii) For accessing areas where human face difficulty to personally access

What issues did I face:

Passing of over voltage through ESP32 Cam module led to breakdown of the Module, and it became unusable thereafter.

Some more pictures:

(Successfully connecting mobile device with ESP32 Cam module in a custom IP address)



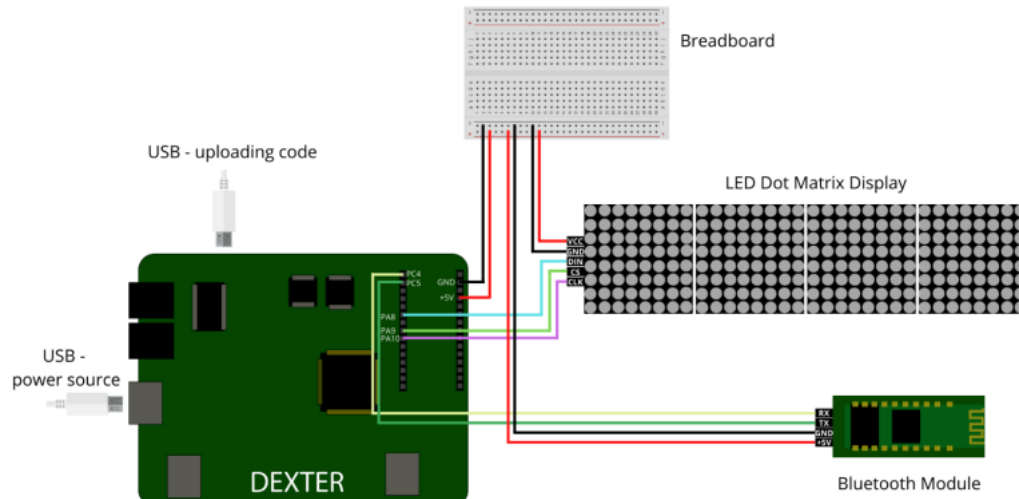
(Workspace photo where Arduino and ESP32 Cam connections can be seen)

Title: Rolling LED Display

Motivation: To use Dexter Base, a custom-built embedded processor and get familiarized to it (STM32L452RC).

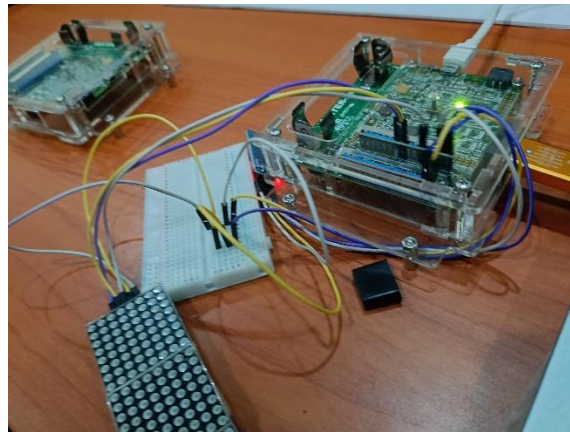
Aim/Objective: Build an LED matrix display and send messages to it wirelessly from our phone using Bluetooth. This includes controlling the Direction and Speed of scrolling as well as Brightness of the display remotely from our phone.

Circuit Diagram:

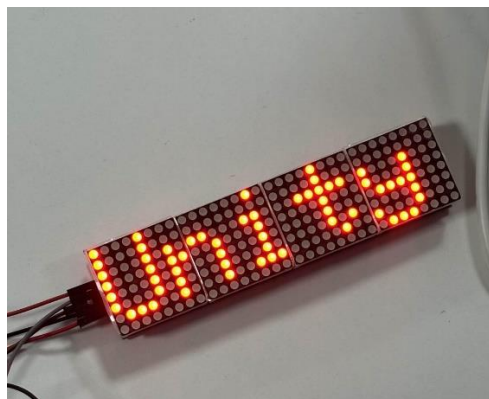


Components Required: Dexter board, LED Dot Matrix display - MAX 7219 board with four 8x8 displays mounted, Bluetooth module - HC05 module, Breadboard, Jumper wires, USB Cables

Product Designed:



Sample Output:



Advantages: i) Ultra-low-power with Flex Power Control
ii) Alert notifications through Buzzer and LED
iii) Availability of all required ports

Limitations: i) frequency up to 80MHz
ii) 256MB SPI flash memory, scalable up to 1GB

What I have Learnt: i) C Programming into real life applications
ii) To use a custom-built processor
iii) STM 32 IDE Software familiarization

Basic Principles Used:

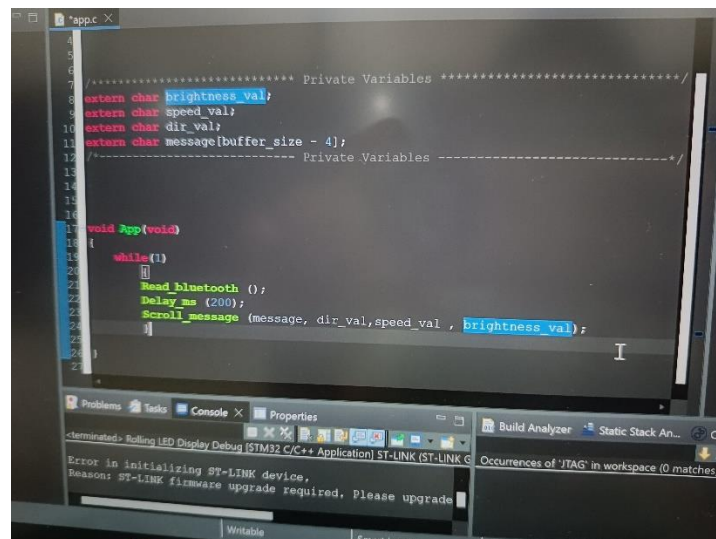
converting an electric current into light by projecting an electric current through a semiconductor with a crystalline structure, such as Silicon.

Potential Application:

What issues did I face:

Unstable/ Connection Loosing tendencies between the devices in a frequent manner.

Some Pictures:

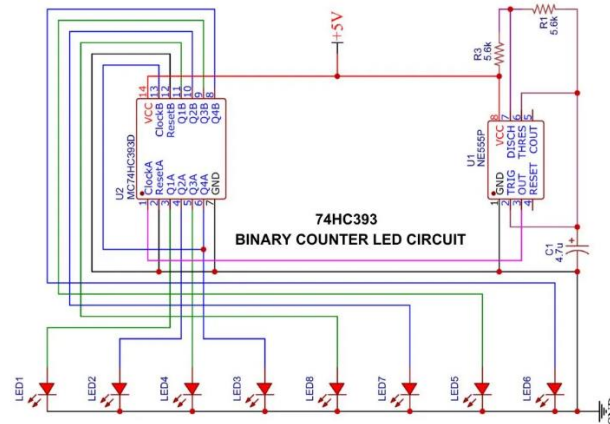


Title: 8-bit Binary Counter using HC393

Motivation: Familiarize with 555 Timer IC and HC393

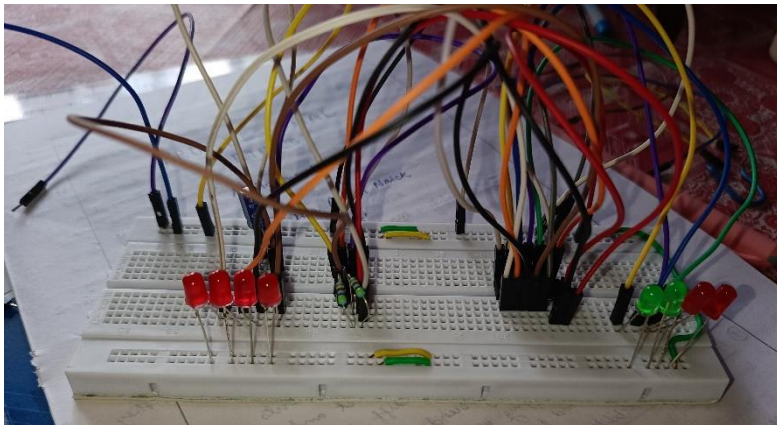
Aim/Objective: To build a typical binary counter using 555 Timer IC

Circuit Diagram:



Components Required: 74HC393 Binary Counter IC, NE555 Timer IC, LED, 2 5.6K Resistor, 4.7uF Capacitor, 5V power supply, Connecting Wires

Product Designed:



Sample Output: not available due to constantly changing samples

Advantage: HC393 have parallel outputs from each counter stage so that any submultiple of the input count frequency is available for system timing signals

Limitations: counts only up to 256

What I have Learnt: i) To design a circuit using 555Timer IC.

ii) Learn some basic features of HC393

iii) Gone through the internal circuitry of 555 Timer IC.

Basic Principles Used: 555 Timer IC uses internal comparators to monitor the voltage across an external capacitor, triggering a flip-flop to switch its output state when the capacitor voltage reaches specific thresholds ($1/3$ and $2/3$ of the supply voltage)

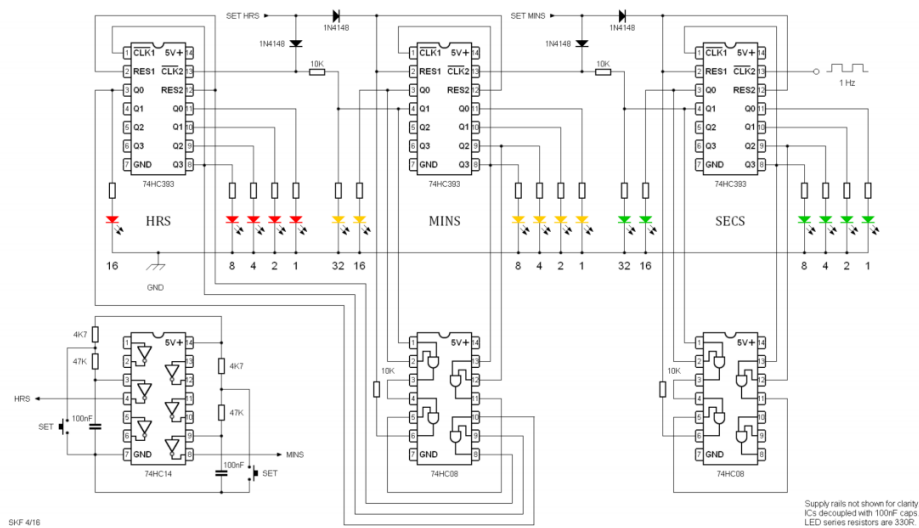
Potential Application:

Title: Binary Clock

Motivation: To get familiarized with improvement in wiring of complex circuits

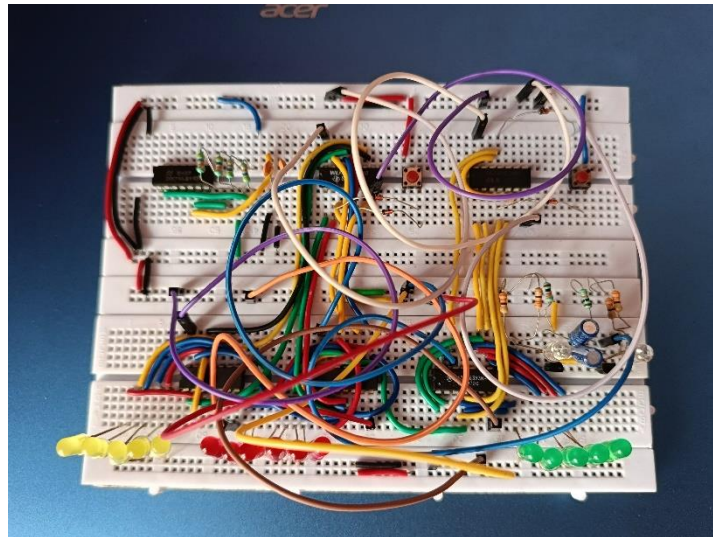
Aim/Objective: To Designing a Binary Clock using logic gates

Circuit Diagram:



Components Required: 3 × 74HC393 IC, 2 × 74HC08 IC, 1 × 74HC14 IC, 4 × 1N4148 IC, 2 × Switch, 2 × 100 nF Capacitor, 4 × 47 kΩ Resistors, 17 × 330 Ω Resistors, 3 × 10 kΩ Resistors

Product Designed:



Sample Output: not available due to constantly changing samples

Advantage: i) Works as a normal clock

ii) Useful in applications requiring sequential data storage

iii) Fast data transfer and clock-driven synchronization

iv) Operates under minimal power.

Limitations: The time shown is not easily understandable for a non-technical human

What I have Learnt:

i) Generating a pulse not using an IC, but by designing an astable multivibrator

ii) Divide and work in a team.

iii) Improved Communication Skills

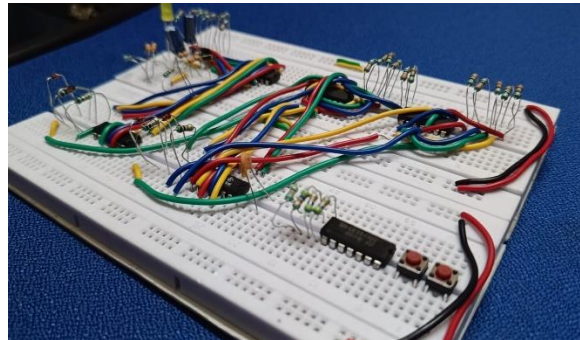
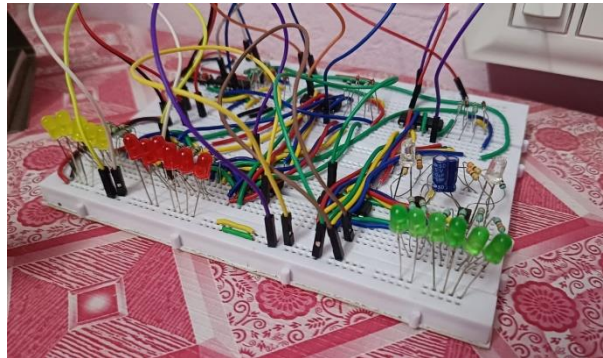
iv) Improved my wiring skills

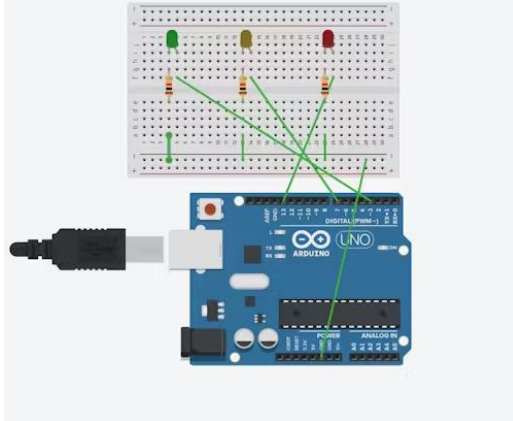
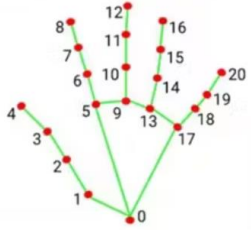
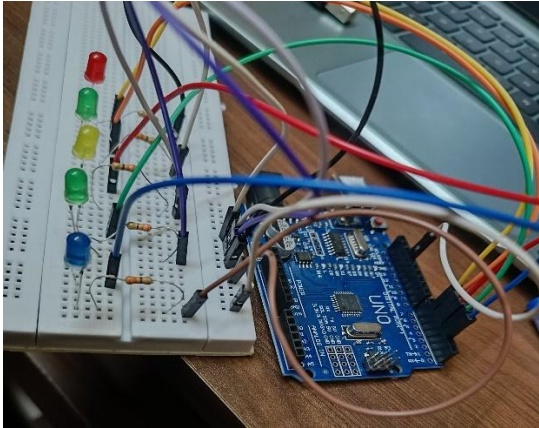
Basic Principles Used: D flip-flops that latch data on the rising edge of the clock, with additional asynchronous set and reset inputs to force specific output states.

Potential Application:

- i) Used in embedded devices, where low-cost, simple, and reliable timekeeping is required
- ii) great tool for teaching and learning about binary numbers, logic gates, and digital circuits in educational institutions.
- iii) In communication systems where data transmission and time synchronization are critical

Some Pictures:



| | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|----------|-----------------------|--------------|-----------------------|--------------|---------------------|-------------|---------------------|--------------|---------------------|---------------------|---------------------|---------------------|---------------|---------------------|---------------|---------------------|---------------|----------------------|---------------|-----------------------|--|
| Title: <u>Finger Gesture controlled LEDs</u> | | | | | | | | | | | | | | | | | | | | | | | |
| Motivation: Familiarize with a Machine Learning and get used to OpenCV | | | | | | | | | | | | | | | | | | | | | | | |
| Aim/Objective: To implement a switching mechanism of LEDs placed in a sequence and turn them ON/OFF only using our fingers. | | | | | | | | | | | | | | | | | | | | | | | |
| Circuit Diagram: | | | | | | | | | | | | | | | | | | | | | | | |
|  <p>(in our case, we had used 5 LEDs instead of 3)</p> |  <table border="0"> <tbody> <tr> <td>0. WRIST</td><td>11. MIDDLE_FINGER_DIP</td></tr> <tr> <td>1. THUMB_CMC</td><td>12. MIDDLE_FINGER_TIP</td></tr> <tr> <td>2. THUMB_MCP</td><td>13. RING_FINGER_MCP</td></tr> <tr> <td>3. THUMB_IP</td><td>14. RING_FINGER_PIP</td></tr> <tr> <td>4. THUMB_TIP</td><td>15. RING_FINGER_DIP</td></tr> <tr> <td>5. INDEX_FINGER_MCP</td><td>16. RING_FINGER_TIP</td></tr> <tr> <td>6. INDEX_FINGER_PIP</td><td>17. PINKY_MCP</td></tr> <tr> <td>7. INDEX_FINGER_DIP</td><td>18. PINKY_PIP</td></tr> <tr> <td>8. INDEX_FINGER_TIP</td><td>19. PINKY_DIP</td></tr> <tr> <td>9. MIDDLE_FINGER_MCP</td><td>20. PINKY_TIP</td></tr> <tr> <td>10. MIDDLE_FINGER_PIP</td><td></td></tr> </tbody> </table> | 0. WRIST | 11. MIDDLE_FINGER_DIP | 1. THUMB_CMC | 12. MIDDLE_FINGER_TIP | 2. THUMB_MCP | 13. RING_FINGER_MCP | 3. THUMB_IP | 14. RING_FINGER_PIP | 4. THUMB_TIP | 15. RING_FINGER_DIP | 5. INDEX_FINGER_MCP | 16. RING_FINGER_TIP | 6. INDEX_FINGER_PIP | 17. PINKY_MCP | 7. INDEX_FINGER_DIP | 18. PINKY_PIP | 8. INDEX_FINGER_TIP | 19. PINKY_DIP | 9. MIDDLE_FINGER_MCP | 20. PINKY_TIP | 10. MIDDLE_FINGER_PIP | |
| 0. WRIST | 11. MIDDLE_FINGER_DIP | | | | | | | | | | | | | | | | | | | | | | |
| 1. THUMB_CMC | 12. MIDDLE_FINGER_TIP | | | | | | | | | | | | | | | | | | | | | | |
| 2. THUMB_MCP | 13. RING_FINGER_MCP | | | | | | | | | | | | | | | | | | | | | | |
| 3. THUMB_IP | 14. RING_FINGER_PIP | | | | | | | | | | | | | | | | | | | | | | |
| 4. THUMB_TIP | 15. RING_FINGER_DIP | | | | | | | | | | | | | | | | | | | | | | |
| 5. INDEX_FINGER_MCP | 16. RING_FINGER_TIP | | | | | | | | | | | | | | | | | | | | | | |
| 6. INDEX_FINGER_PIP | 17. PINKY_MCP | | | | | | | | | | | | | | | | | | | | | | |
| 7. INDEX_FINGER_DIP | 18. PINKY_PIP | | | | | | | | | | | | | | | | | | | | | | |
| 8. INDEX_FINGER_TIP | 19. PINKY_DIP | | | | | | | | | | | | | | | | | | | | | | |
| 9. MIDDLE_FINGER_MCP | 20. PINKY_TIP | | | | | | | | | | | | | | | | | | | | | | |
| 10. MIDDLE_FINGER_PIP | | | | | | | | | | | | | | | | | | | | | | | |
| Components Required: Arduino UNO, breadboard, resistors, LEDs, jumper wires | | | | | | | | | | | | | | | | | | | | | | | |
| Product Designed: | | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | | |
| Sample Output: | | | | | | | | | | | | | | | | | | | | | | | |
| Advantage: i) Does not require mechanical switches ii) Hands-Free Operation (without any physical contact) iii) Customizable and Versatile iv) Enhanced user experience | | | | | | | | | | | | | | | | | | | | | | | |
| Limitations: Works perfectly up to 5 LEDs, after this number there is seen a fluctuation seen in power supply. | | | | | | | | | | | | | | | | | | | | | | | |
| What I have Learnt: | | | | | | | | | | | | | | | | | | | | | | | |
| Basic Principles Used: | | | | | | | | | | | | | | | | | | | | | | | |
| Potential Application: | | | | | | | | | | | | | | | | | | | | | | | |

| |
|-------------------------------|
| <u>Title</u> |
| Motivation: |
| Aim/Objective: |
| Circuit Diagram: |
| Components Required: |
| Inputs: |
| Sample Output: |
| Advantage: |
| Limitations: |
| What I have Learnt: |
| Basic Principles Used: |
| Potential Application: |