# Interrupt Driven LED Control using DWIN Display

**Saikishen P V**

The system uses a hardware UART interrupt to capture incoming data asynchronously, ensuring minimal latency and optimal CPU utilization. Received data is processed in a structured manner to control peripheral outputs based on validated command frames.

## Interrupt Service Routine Workflow

The UART ISR follows the steps outlined below:

1. The UART receive interrupt is triggered when new data arrives.
2. The ISR checks whether data is available in the UART RX buffer.
3. Incoming bytes are read one at a time.
4. Each byte is stored sequentially in a predefined receive buffer.
5. A buffer index is incremented after every byte.
6. Once the expected frame size is reached:
    - A frame completion flag is set.
    - The buffer index is reset.
7. The ISR exits immediately, returning control to the main program.

## Main Loop Processing

The main program loop continuously monitors a flag that indicates the availability of a complete data frame. When this flag is set:

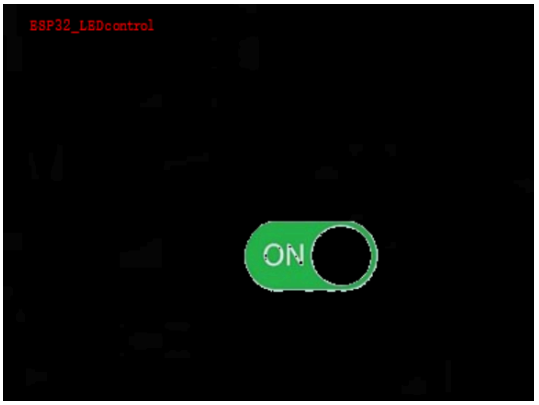
1. The received data is copied to a local buffer to avoid race conditions.
2. The frame completion flag is cleared.
3. The frame header and control fields are validated.
4. Based on the received command value, the corresponding action is executed.

## Frame Validation Logic

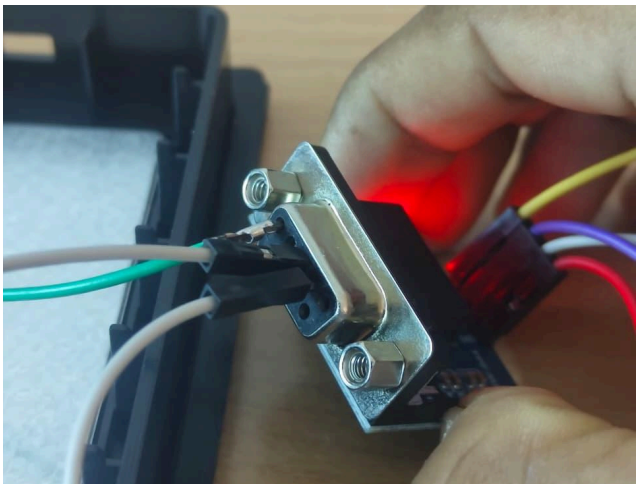
To ensure data integrity, the following validation checks are performed:

- Verification of frame start byte
- Verification of command identifier bytes
- Extraction of payload data for decision making

## HMI Interface:

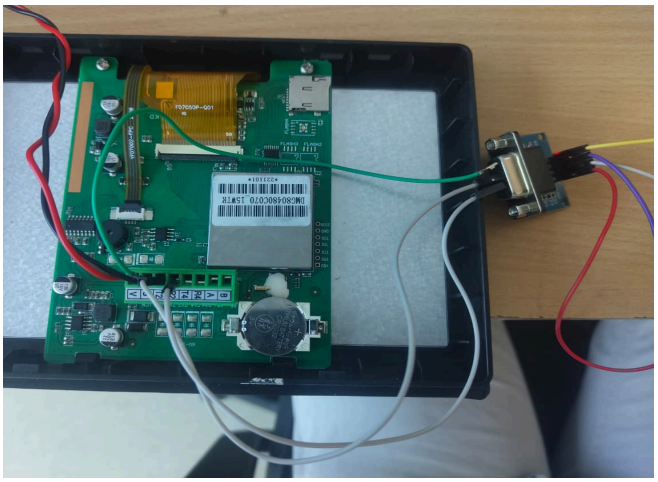

## Circuit Diagram



**RX2->RX of rs232**

**TX2->TX of rs232**

**Gnd ->Gnd**



MAX3232 TO ESP32:
TXD -> TX2
RXD -> RX2