## CG2271 Real-Time Operating Systems

## App Controlled ESP32 WebServer and KL25Z

We have developed a basic App to control the LED connected to the ESP32 through a WebServer. We will now extend the design to control the KL25Z board.

- Serial Interface of ESP32

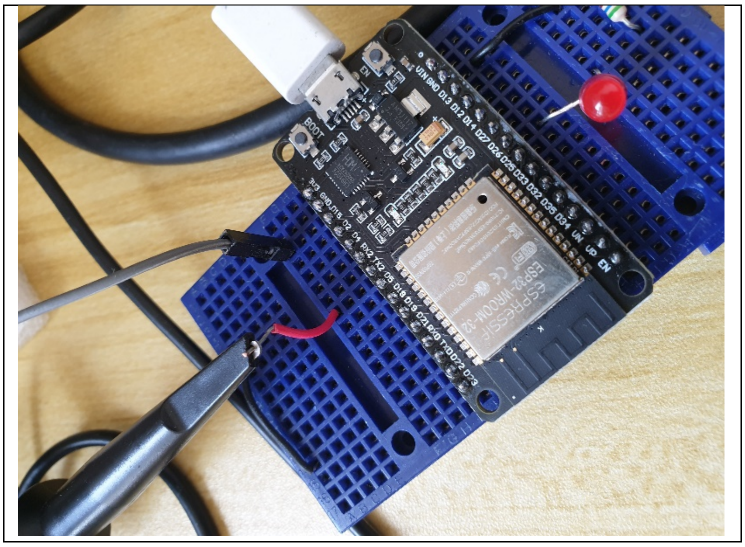
We will first enable the Serial Interface on the ESP32 board to send out different data packets, depending on what action we want the KL25Z board to perform. In this example, I have set aside the first 4 bits to indicate the peripheral/subsystem, and the remaining 4 bits to represent the state. The following table shows the bit patterns and their corresponding action.

*** This is just an example. You are free to design the protocol in any way you want. ***

| Hex Value | Binary Value | Action |
|---|---|---|
| **0x30** | 0b00110000 | OFF Red LED |
| **0x31** | 0b00110001 | ON Red LED |
| **0x32** | 0b00110010 | OFF Green LED |
| **0x33** | 0b00110011 | ON Green LED |
| **0x34** | 0b00110100 | OFF Blue LED |
| **0x35** | 0b00110101 | ON Blue LED |

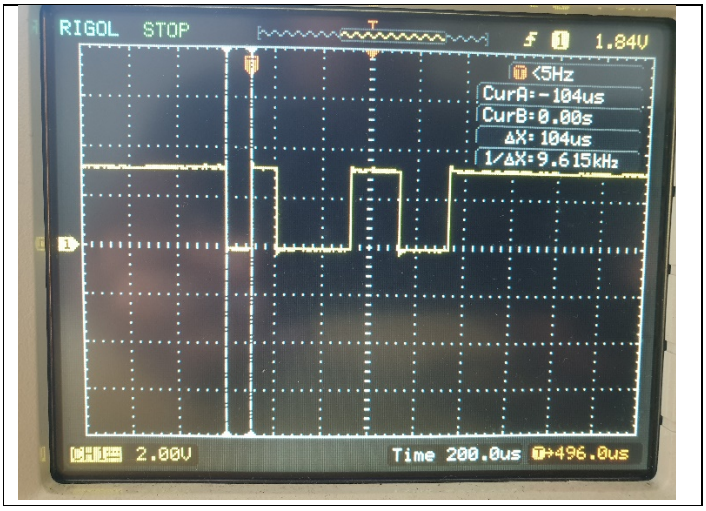
When you examine the ESP32 code, you will observe that the code that checks for the client requests already transmits some data on the Serial Port using Serial2.write(). Serial2 refers to the UART module 2 for the ESP32 and it is mapped to GPIO 16 and 17 on the board. On the board, it is already labelled as TX2 and RX2.

Connect the oscilloscope to TX2 of the ESP32.

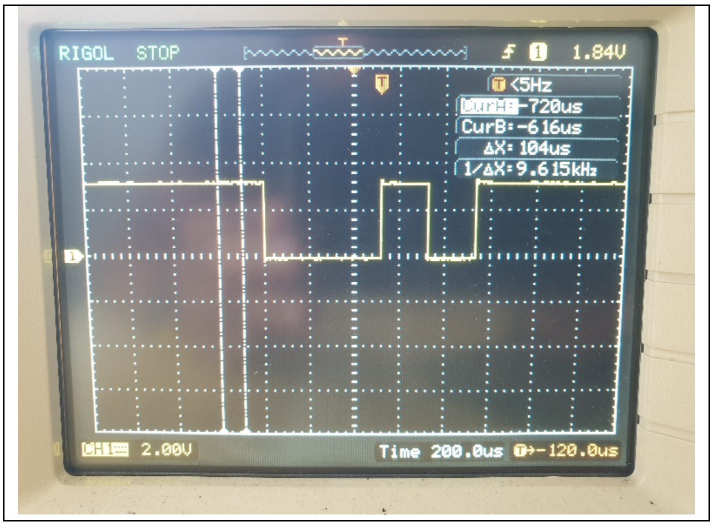


Observe the signal being transmitted whenever you send the following requests through the browser.

http://192.168.43.221/onRed **-> 0x31**



http://192.168.43.221/offRed **-> 0x30**

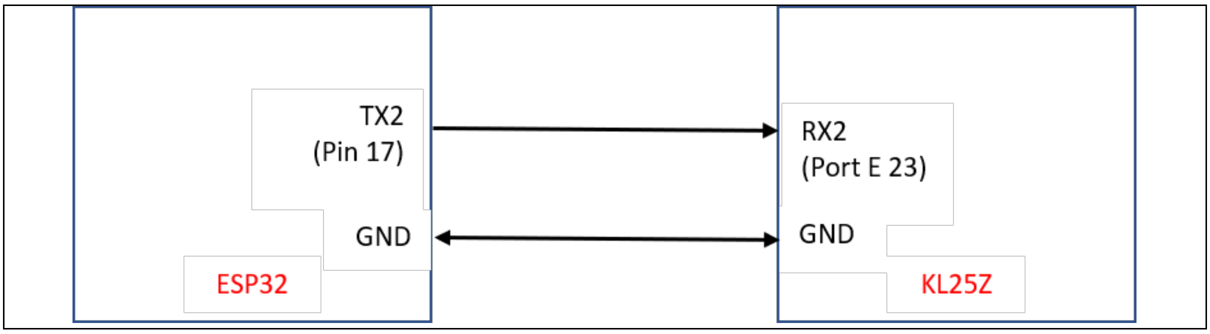


- **Connection between ESP32 and KL25Z**

In order for the ESP32 to relay the command from the App to the KL25Z, we need to connect the Tx of the ESP32 to the Rx of the KL25Z.

*** This step should only be performed after you have developed the code for the KL25Z Serial module in either Polling or Interrupt mode. ***

The HW connection is as such:



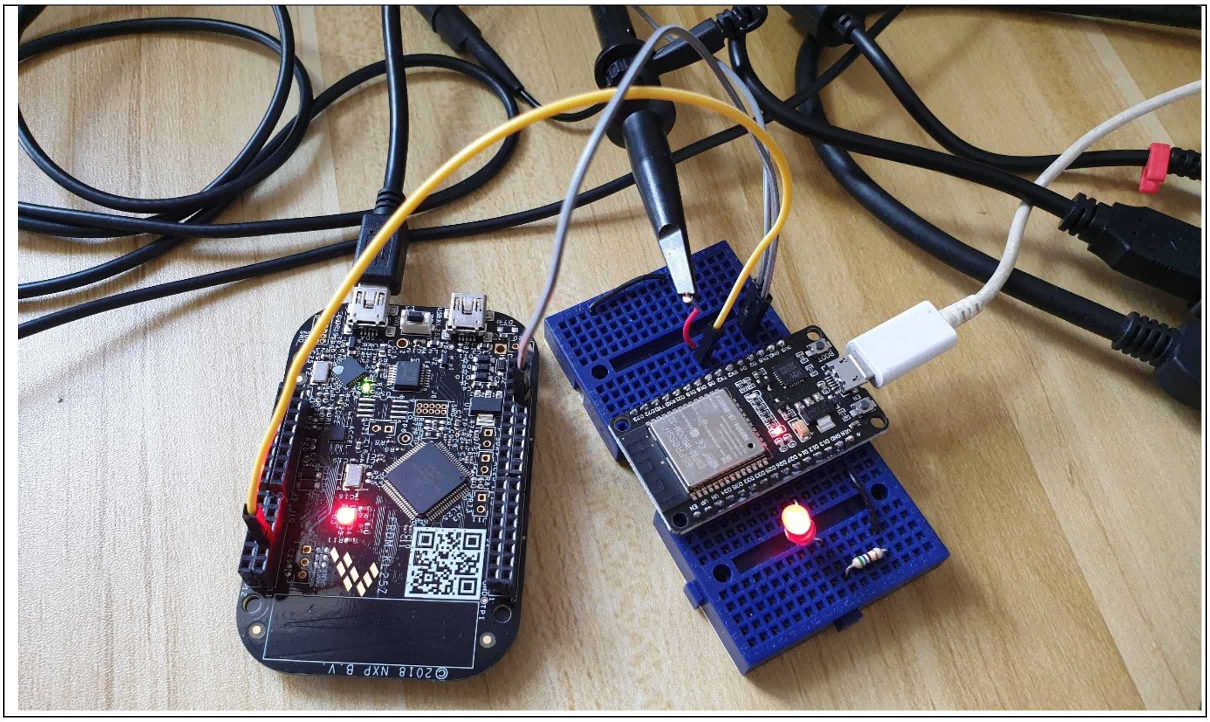
In the Lecture slides, we already went through the process to setup and configure the UART module in the KL25Z. The code implemented the polling approach and this is sufficient for the testing.

- Ensure that the baud rate is 9600bps
- Perform the HW connection as shown in the figure above.
- Configure your code in the KL25Z.

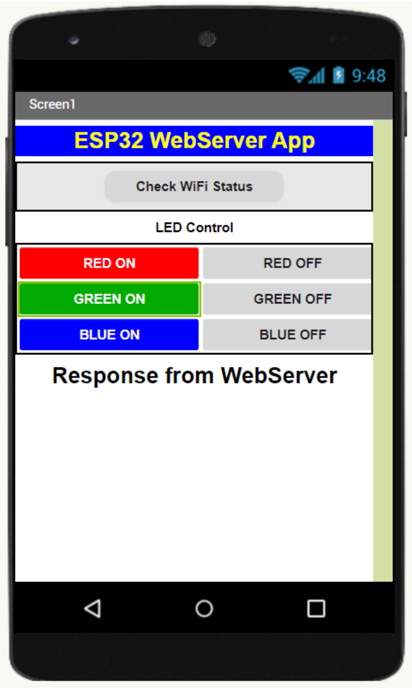
```
main()
{
     // setup code for UART, RGD LED, etc.

     while(1)
     {
          rx_data = UART2_Receive_Poll();

          /*
           Perform necessary if-else checks to compare rx_data with expected values.
           Take appropriate action if any expected value is received
           */

     }
}
```

*** Recall that in your ESP32 code, we are transmitting 0x30 and 0x31 for RED LED Off and On.***

Once the code is ready, compile and download it onto the KL25Z board. Using the Android App, you should be able to control both the ESP32 LED and the KL25Z RGB LED at the same time.



Once this is done, you can update the code to control the other LED’s.



Congratulations! You are done with the entire communication protocol that links up the App with the ESP32 and the KL25Z. Using this framework, you can extend it to control anything on the ESP32 and the KL25Z.

THE END