A picture containing text, electronics, circuit

Description automatically generated

Cylon

ESP32-S3 Development Board

Quick Start Guide

**Introduction**

This Document is a brief introduction to the Cylon development board and a guide to getting started developing with the board by following some basic online tutorials.

**Features**

* **ESP32-S3-WROOM-1 Espressif Wifi Module**
* **Micro USB for programming and powering the board**
* **+5V to +3v3 volt linear regulator**
* **11 individually addressable WS2812B-2020 LEDs**
* **RS-485 Transceiver**
* **1 x Programmable push button**
* **2 x Push Buttons for pulling EN and GPIO0 low**
* **Optional Coin cell battery holders on bottom of board**
* **Switch to turn Coin cell battery power ON/OFF**
* **Breakout header for accessing MCU pins**

[**ESP32-S3-WROOM-1**](https://www.espressif.com/sites/default/files/documentation/esp32-s3-wroom-1_wroom-1u_datasheet_en.pdf) **Module**

This module features:

* CPU and On-Chip Memory
* Wifi
* Bluetooth
* Peripherals – GPIO, SPI, LCD interface, Camera Interface, UART, I2C, etc. (see module [datasheet](https://www.espressif.com/sites/default/files/documentation/esp32-s3-wroom-1_wroom-1u_datasheet_en.pdf))
* Integrated Components
* Antenna
* Voltage 3.0 ~ 3.6 V
* FCC and CE Certified (see full [list](https://www.espressif.com/en/support/documents/certificates?keys=ESP32-S3-WROOM-1&field_product_value%5B%5D=ESP32-S3-WROOM-1) of certifications)

**Micro USB port**

The board can be powered by 5V over the micro USB port. This port is also connected to the USB OTG pins on the Wifi module which allows the CPU to be flashed and debugged over the usb port as well. This is the recommended method for programming since a USB-UART bridge is not included on the development board.

**5V to +3V3 volt linear regulator**

The 5V over the USB port as well as the 6V if coin cell batteries are used is used for powering the WS2812B-2020 LEDs directly as well as being regulated to +3V3 to power the WROOM-1 module and RS485.

When +3V3 is present the Red LED labeled LED12 will be lit.

**WS2812B-2020 LEDs**

The 11 LEDs on the side of the board are individually addressable RGB leds. They communicate over a single serial line connected to GPIO5 that can control brightness and color.

**RS-485 Transceiver**

The board is equipped with a single RS485 transceiver that can be connected to using the header pins on connecter labeled J1. The transceiver is connected to the MCU UART RXD0 and TXD0 pins.

**1 x Programmable push button**

The push button labeled Key3 has no pre-determined function. It is connected to GPIO18 and is available to be programmed by the user.

**Push Buttons for pulling EN and GPIO0 low**

The push buttons labeled key 1 and key 2 are intended for resetting the MCU and setting the bootmode. Key2 is connected to the EN pin on the WROOM-1 module and can be used to reset the MCU. Key1 is connected to GPIO1 and can be used to set the bootmode (setting the bootmode shouldn’t be necessary in the tutorial listed in this guide).

**Optional Coin Cell Batteries**

The Cylon development board can be powered by coin cell batteries if the battery holder footprints on the bottom of the board are populated. This board is not designed for low power consumption applications, but should be capable of them.

**Slide Switch**

**The switch labeled SW1 is only switches power to the board from the coin cell batteries. It does not have a function when batteries are not used.**

**Breakout header pins**

The header pins labeled H1 are connected to all the pins on the WROOM-1 module. These are solely to gain easy access to the MCU.

**Board Tour**

|  |  |
| --- | --- |
| **GND** | **GND** |
| **+3V3** | **IO1** |
| **EN** | **IO2** |
| **IO4** | **UART0\_TX** |
| **IO5** | **UART0\_RX** |
| **IO6** | **IO42** |
| **IO7** | **IO41** |
| **IO15** | **IO40** |
| **IO16** | **IO39** |
| **IO17** | **IO38** |
| **S3** | **IO37** |
| **IO8** | **IO36** |
| **IO19** | **IO35** |
| **IO20** | **IO0** |
| **IO3** | **IO45** |
| **IO46** | **IO48** |
| **IO9** | **IO47** |
| **IO10** | **IO21** |
| **IO11** | **IO14** |
| **IO12** | **IO13** |

|  |  |  |  |
| --- | --- | --- | --- |
| **NC** | **RS485\_B** | **RS485\_A** | **NC** |
| **GND** | **NC** | **+5V** | **+3V3** |

**+5V to +3V3 Regulator**

**Micro USB**

**+3V3 LED**

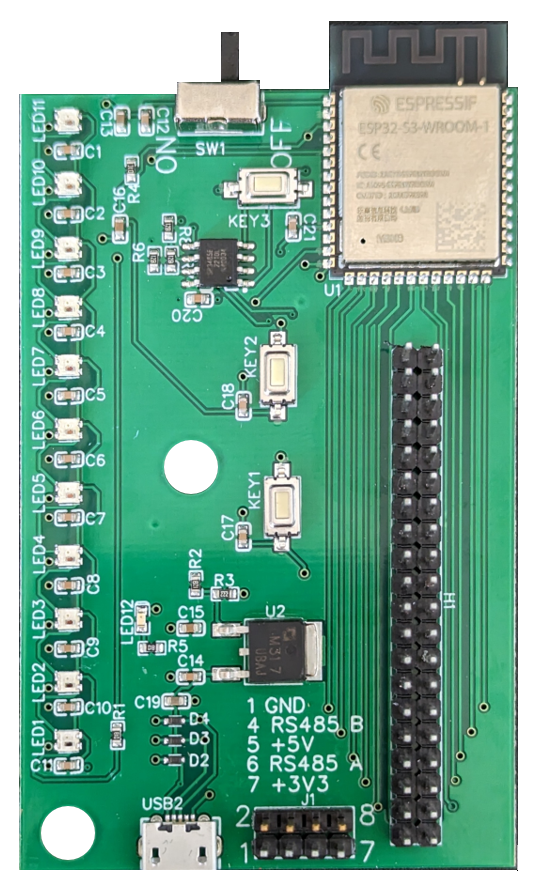
**Addressable LEDs**

**RS485 Transceiver**

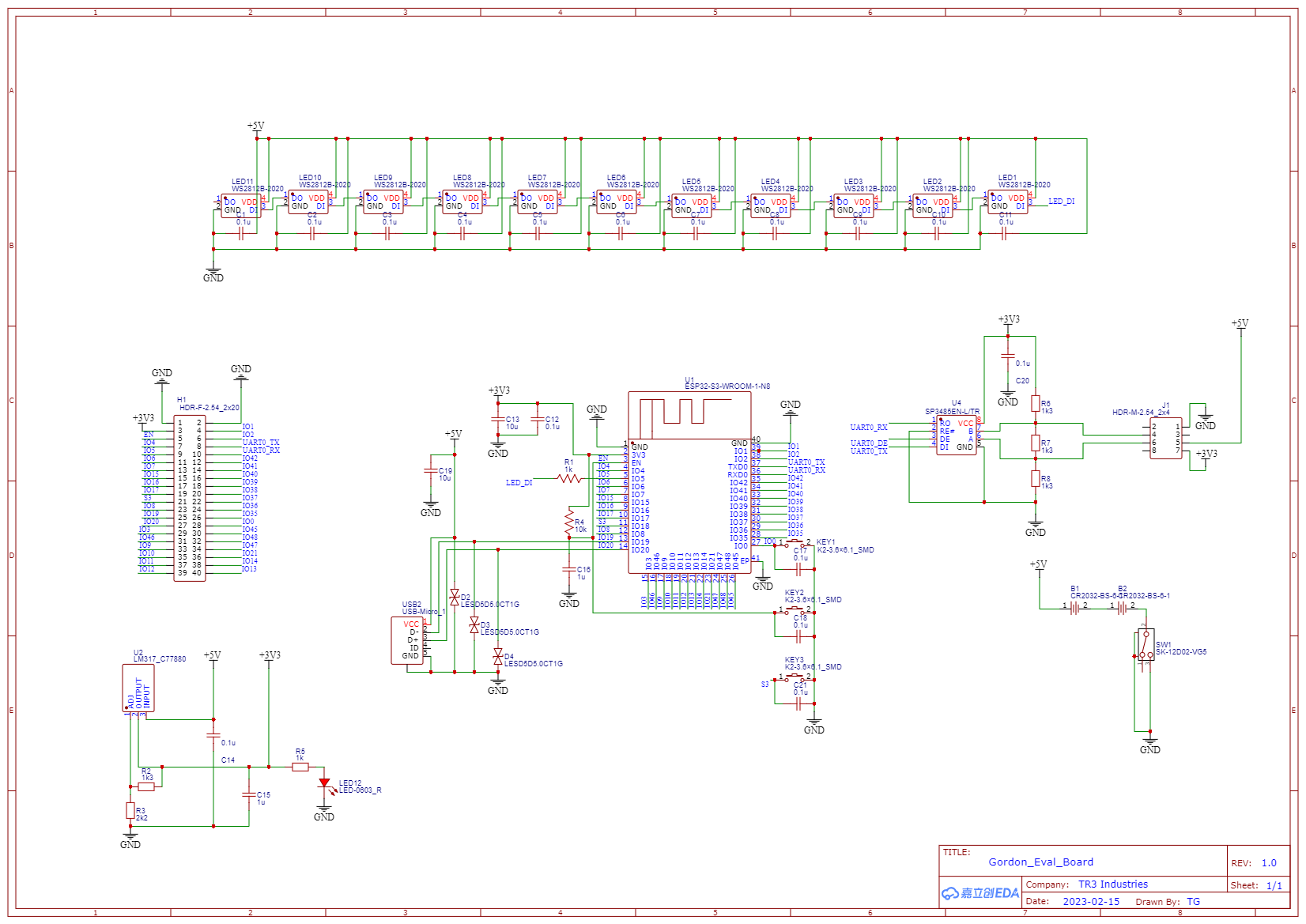
**Battery Switch**

**Programmable Button**

**ESP32-S3-WROOM-1**



|  |
| --- |
| **Key1 – GPIO0** |
| **Key2 – EN (reset)** |

**Schematic**

**Programming the Cylon**

The Cylon Development board can be programmed and powered over the micro USB port. I used VS-code with the IDF-ESP plugin for my IDE. And used the IDF-ESP command prompt to flash and monitor my board since I couldn’t get the terminal within VS code to recognize idf.py command. The steps I took follow:

1. Download VScode
2. Follow Espressif’s guide to getting started with VS Code IDE to get the plugin and required tools/dependencies. (I used Eclipse for this so didn’t actually go through this step in VS code which may be why I had issues running idf.py commands in the built in vs code terminal)

<https://docs.espressif.com/projects/esp-idf/en/v4.2.3/esp32/get-started/vscode-setup.html>

1. I was never able to get the terminal to run idf.py commands so I had to use the IDF-ESP command prompt to navigate to the project folder to flash and run my project. Using:

idf.py -p COM# flash monitor (this flashes the program and then opens the monitor tool)  
Text

Description automatically generated

1. I followed this guide to setting up a webserver and controlling an LED on a gpio

<https://esp32tutorials.com/esp32-web-server-esp-idf/>

1. Then I followed the esp-idf example project to blink an LED in which an addressable LED like the ones used on the Cylon board can be controlled using the LED\_STRIP.h library.
2. By combining the two guides I was able to turn on and off LED 1 by clicking the butons on the webserver.