ASC CSE IoT 2.0 Kit



**Board Specifications:**

* Based on ESP WROOM32 Wi-Fi (802.11 b/g/n) + BLE Module (BLE 4.2)
* Based on the need, module can be changed
* On-board Reset and Boot Push Buttons.
* Supports LwIP and FreeRTOS
* On-board User LEDs (2 Nos), one Push Button, One 10K Analog POT, LDR. ( Connected to default IOs through Jumpers and can be connected to any IOs for assessments purposes)
* CH340 based USB to UART Converter.
* On-board Type C Connector for powering the Board, Programming the Module, Serial Port.
* On-board DC power jack to power the Board from External DC Power Source.
* On-board Connectors for 5V. 3.3V, Gnd to power additional components
* On-board connectors for UART, I2C, SPI and CAN Modules. ( Connected to default IOs through Jumpers and can be connected to any alternate IOs for assessments purposes)
* On-board Connectors for direct plugging of 3-Axis Gyroscope and Accelerometer MPU6050, CAN Transceiver TJA1050, USB2UART FT232 Module, 8 MB SPI based Data Flash module W25Q64.
* PCB : High Quality double sided Glass Epoxy. Size ; 60mm \* 105 mm
* Best Suitable for Training and Prototyping IoT Products.



# ASC-CSE-IoT 2.0 Layout

# On Board peripheral details:

# 

# USB Type C Connector and DC Power Supply Jack Socket

# A close up of a circuit board Description automatically generated

# Switch S1 is Reset (RST) and Switch S2 is Boot Button

# A close up of a circuit board Description automatically generated

# On Board LEDs: LED D2 -> RxD, LED D3 –> TxD, LED D4 –> Board Power .

# 

# GPIO Pins of ESP32

# ESP32weserverpinout

|  |  |  |  |
| --- | --- | --- | --- |
| GPIO Pin | Spl Notes | FXN 1 | FXN 2 |
| 0 | General | ADC 11 | TOUCH 1 |
| 1 | General | TX 0 |  |
| 2 | Bootstrap | ADC 12 | TOUCH 2 |
| 3 | General | RX 0 |  |
| 4 | Bootstrap | ADC 10 | TOUCH 0 |
| 5 | Bootstrap | VSPI SS |  |
| 6 | Internal use | FLASH CK |  |
| 7 | Internal use | FLASH D0 |  |
| 8 | Internal use | FLASH D1 |  |
| 9 | Internal use | FLASH D2 | RX 1 |
| 10 | Internal use | FLASH D3 | TX 1 |
| 11 | Internal use | FLASH CMD |  |
| 12 | Bootstrap | ADC 15 | TOUCH 5 |
| 13 | General | ADC 14 | TOUH 14 |
| 14 | General | ADC 16 | TOUCH 6 |
| 15 | Bootstrap | ADC 13 | TOUCH 13 |

|  |  |  |  |
| --- | --- | --- | --- |
| GPIO Pin | Spl Notes | GPIO Pin | Spl Notes |
| 16 | General | RX 2 |  |
| 17 | General | TX 2 |  |
| 18 | General | VSPI SCK |  |
| 19 | General | VSPI MISO |  |
| 21 | General | I2C SDA |  |
| 22 | General | I2C SCL |  |
| 23 | General | VSPI MOSI |  |
| 25 | General | ADC 18 | DAC 1 |
| 26 | General | ADC 19 | DAC 2 |
| 27 | General | ADC 17 | TOUCH 7 |
| 32 | General | ADC 04 | Touch 9 |
| 33 | General | ADC 05 | Touch 8 |
| 34 | General | ADC 06 |  |
| 35 | General | ADC 07 |  |
| 36 | General | ADC 00 |  |
| 39 | General | ADC 03 |  |

# 

# Users can take 3.3V , 5V and Ground from the Connector P10 for other peripheral devices. User can give external 5 V to the Board through VIN pin. (Note: Take care while powering the board other than USB port. VIN- Max 5V and the board can be also powered through DC Jack Socket with a maximum of 9V DC source)

# 

# On-Board User Button (Switch), LED, POT and LDR

# 

|  |  |
| --- | --- |
| **User Peripherals** | **GPIO Pin** |
| **Switch SW (S3)** | **32** |
| **LED 1 (D7)** | **33** |
| **LED 2 (D8)** | **4** |
| **POT (RP1)** | **36** |
| **LDR (R12)** | **39** |

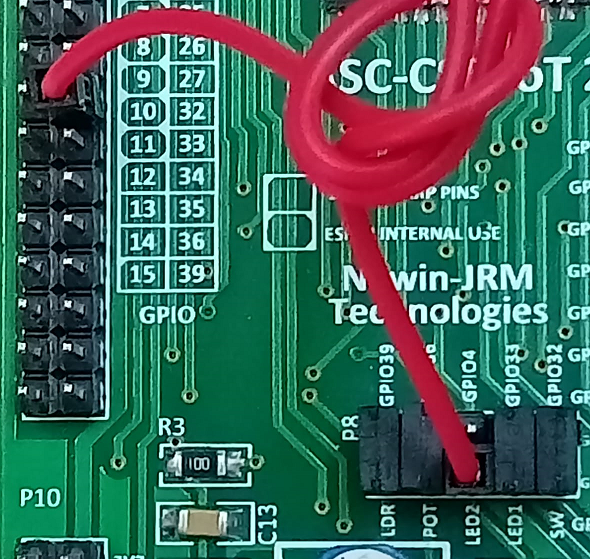
By placing the individual respective jumpers, on board User Button (SW) , USER LED1 and 2, POT and LDR will be connected to the particular GPIO mentioned in the above table.

By removing the Jumper, they can be connected to any available GPIO pins as per design.

For example, suppose, if you want to connect USER LED 2 to GPIO 26, you have to remove the respective jumper connecting USER LED2 to GPIO4 and a jumper wire is used to connect one end of the USER LED 2 to GPIO 26 as illustrated in the following image.

**PLs Note that BOOT Button at GPIO 0 can also be used as user button during program running time.**

**So, users can have two buttons, two LEDs and two Analog inputs POT and LDR in the board itself.**

****

**Sample Programs.**

[**https://github.com/asc-es-iot/asc-esp32-iot-board**](https://github.com/asc-es-iot/asc-esp32-iot-board)

**LED Blink at** [**https://github.com/asc-es-iot/asc-esp32-iot-board/tree/main/basics/led\_blink**](https://github.com/asc-es-iot/asc-esp32-iot-board/tree/main/basics/led_blink)

**Button and LED at** [**https://github.com/asc-es-iot/asc-esp32-iot-board/tree/main/basics/button**](https://github.com/asc-es-iot/asc-esp32-iot-board/tree/main/basics/button)

**Analog Input (POT) and Serial Output at**

[**https://github.com/asc-es-iot/asc-esp32-iot-board/tree/main/basics/potentiometer**](https://github.com/asc-es-iot/asc-esp32-iot-board/tree/main/basics/potentiometer)

**Light Sensor LDR and Serial Output at**

[**https://github.com/asc-es-iot/asc-esp32-iot-board/tree/main/basics/ldr**](https://github.com/asc-es-iot/asc-esp32-iot-board/tree/main/basics/ldr)

**Physical Computing using Arduino (ESP32)**

**Pre-Workshop Notes**

Installation of latest Arduino IDE:

**1. Visit**[Software | Arduino](https://www.arduino.cc/en/software) (<https://www.arduino.cc/en/software>) and download

A screenshot of a computer

Description automatically generatedArduino IDE 2.2.1

**A screenshot of a computer

Description automatically generated**

A screenshot of a computer program

Description automatically generated

Click Just DOWNLOAD. Arduino-ide\_2.2.1\_windows\_64bit.zip will be downloaded. Extract this zip.

You will get a folder named Arduino-ide\_2.2.1\_windows\_64bit

**A screenshot of a computer

Description automatically generated**

In this folder , right click Arduino IDE ( Application file ) to create a shortcut and place the Short Cut in the Desktop.

**A screenshot of a phone

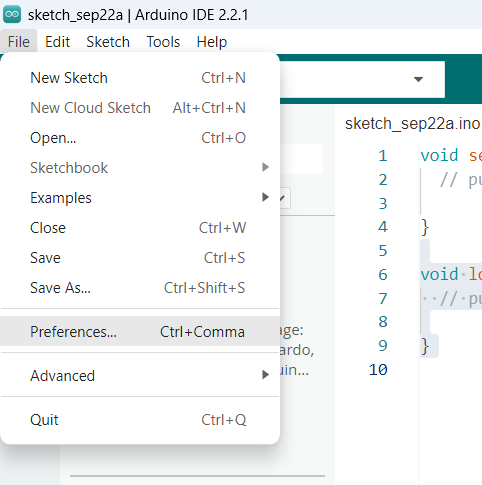
Description automatically generated**

A screenshot of a computer

Description automatically generated

Arduino Installation was successfully completed.

Next we need to install ESP32 Board under the Boards Manager. Click preferences under file menu.

****

In the Additional boards manager, paste

<https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json>

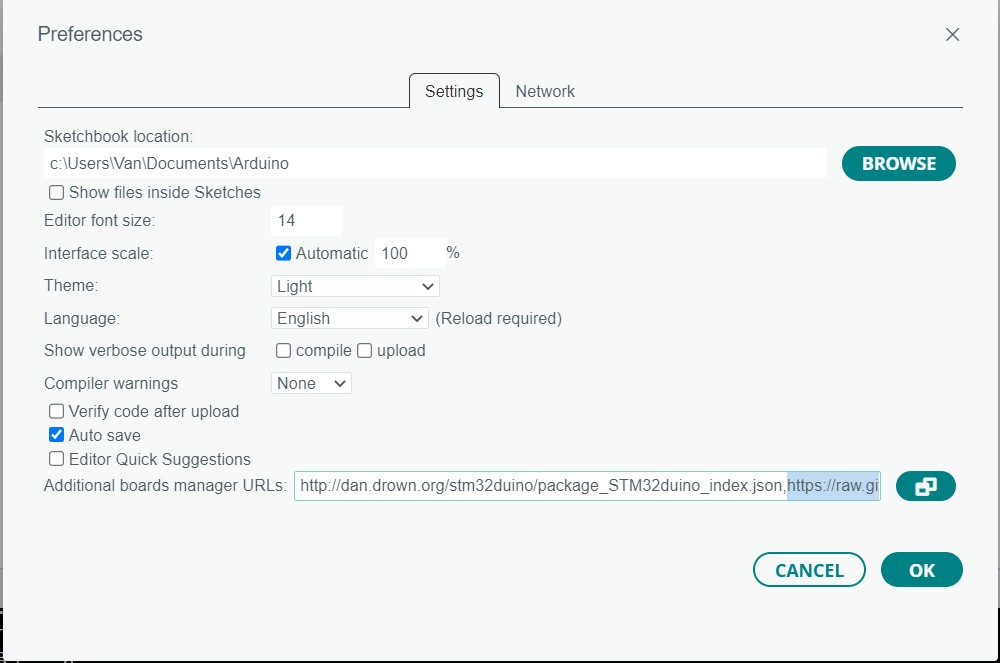
If some URLs are already there, say

<http://dan.drown.org/stm32duino/package_STM32duino_index.json>

Just put a comma and then paste the URL for ESP32 Installation files.

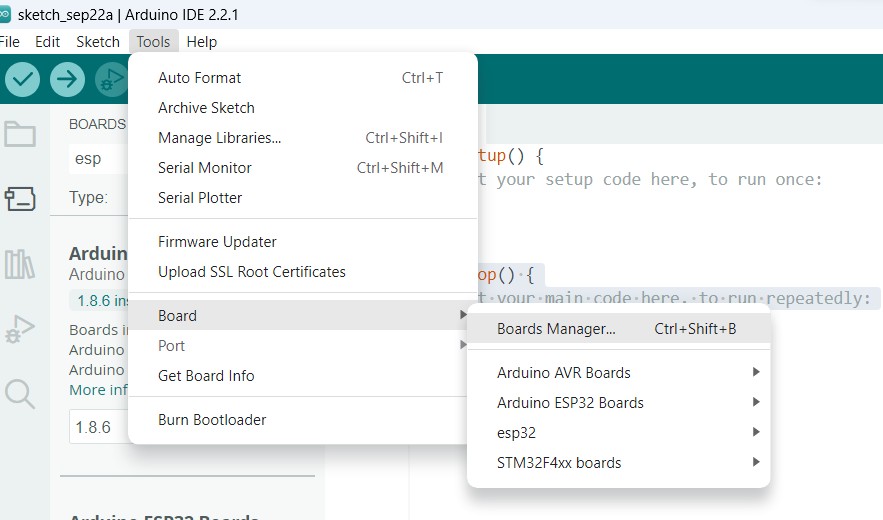
http://dan.drown.org/stm32duino/package\_STM32duino\_index.json,https://raw.githubusercontent.c

om/espressif/arduino-esp32/gh-pages/package\_esp32\_index.json

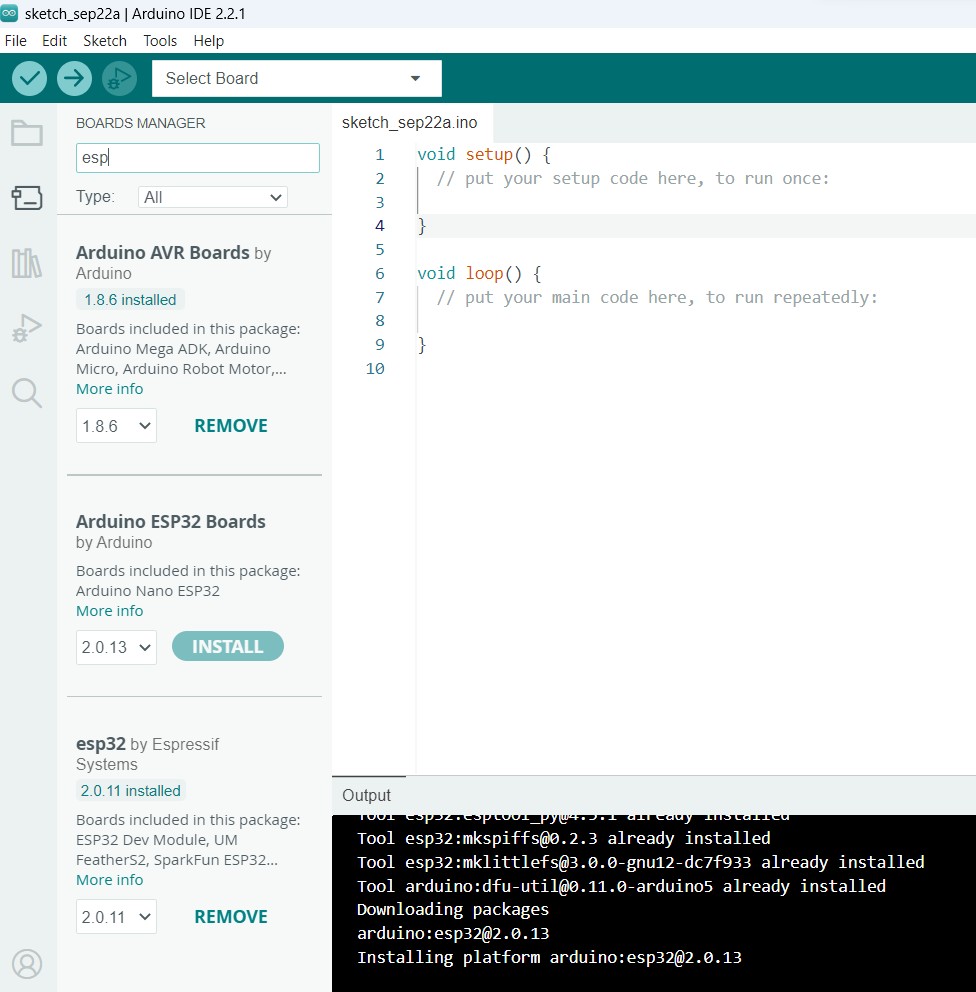
****

Click OK to proceed Further.

Under Tools menu, Select Board and then Boards Manager.



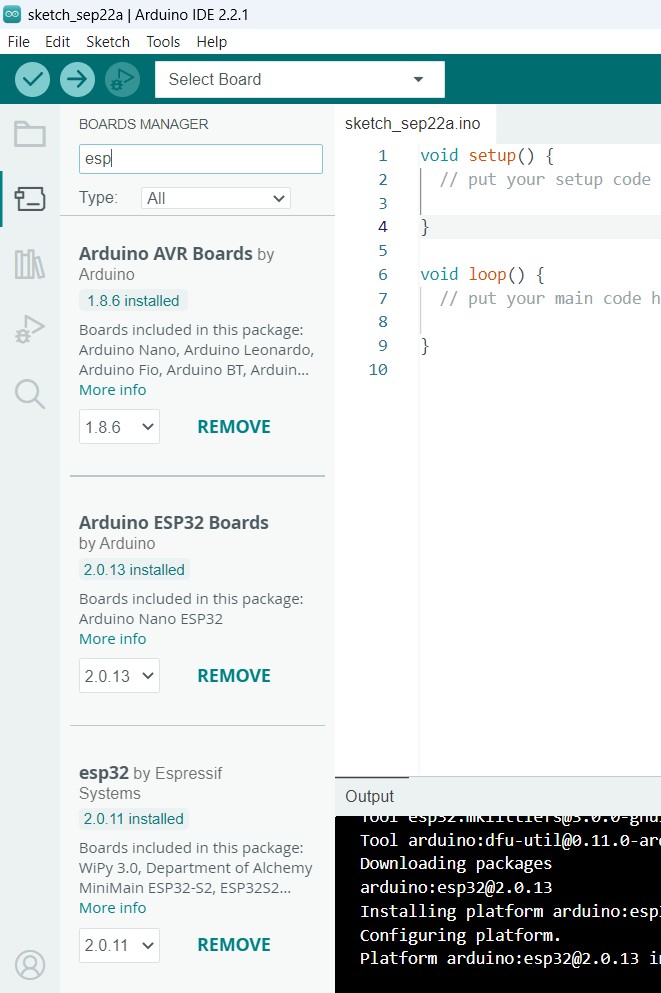
In the following screen, type esp in the text box provided for Boards Manager.



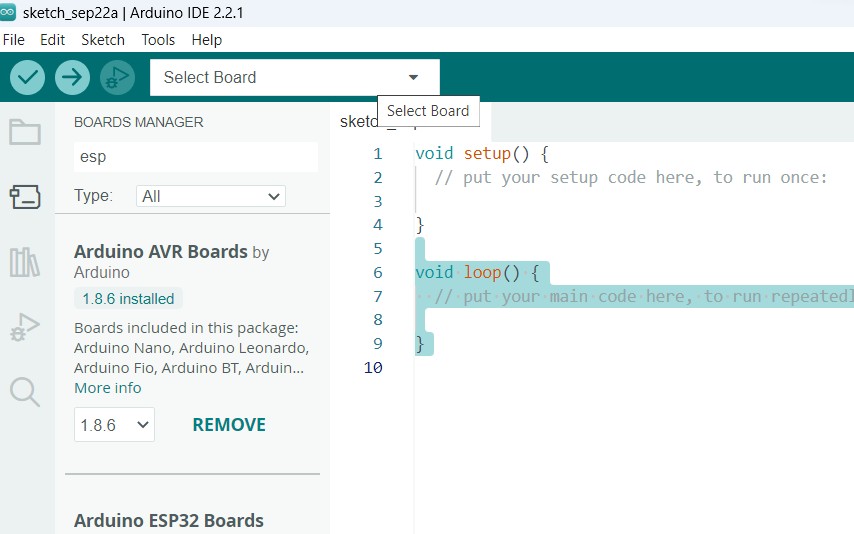
You will have a list of related board packages.

You have to install both esp32 by Espressif and Arduino ESP32 boards by Arduino

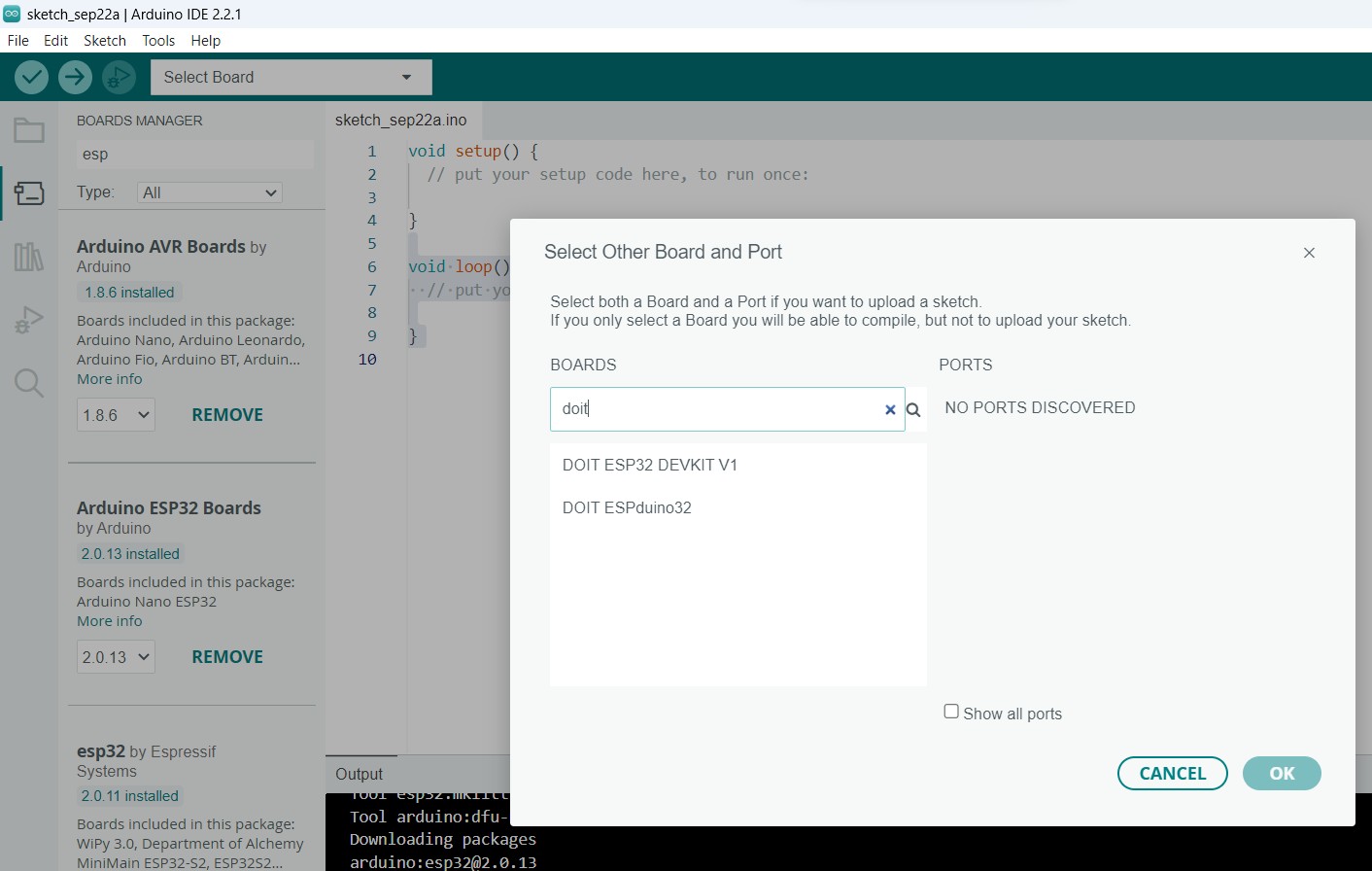
After clicking install button, it will take few minutes to install and after successful installation, Install Button will be changed to Remove Button.

****

After installation, click select board



If you are able to see the below screen, your software installation was successfully completed.

****

Installation of CH340 USB-UART Bridge driver

**CH340 is a TTL (serial) to USB converter and vice versa. This chip has been used in some boards such as Arduino boards (non-original Arduinos), ESP8266, etc. The boards using the CH340 chip, don’t need a programmer in order to access the processor or to program them. But there is a downside. An extra driver must be installed before starting to work with boards having this IC. In this section, you will learn how to install the CH340 driver.**

Installing CH340 Driver on Windows

If you connect your board to the computer before installing the driver, your computer will not recognize the board correctly and you will see following image in Device Manager.

To open Device Manager, search for it in the Windows Start menu.

**A screenshot of a computer

Description automatically generated**A screenshot of a computer

Description automatically generated

Follow the steps below to install the CH340 driver:

**Step 1: CH340 driver download**

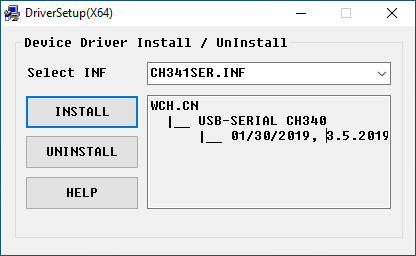
First, download the CH340 driver from this link or any alternate link.

<http://www.wch.cn/download/CH341SER_ZIP.html>

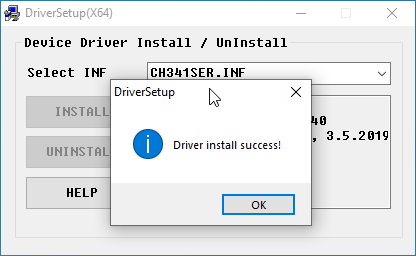
[**https://electropeak.com/learn/download/ch341ser-zip/**](https://electropeak.com/learn/download/ch341ser-zip/)

### Step 2: Installing the driver

After downloading the CH340driver, open it and click **Install**.



After successful installation you should see this message.



**A screenshot of a computer

Description automatically generated**

### Step 4: Checking Correct Board and Driver Installation in Arduino IDE

### A screenshot of a computer Description automatically generated

### **Type DOIT I the search text box for boards and select DOIT ESP32 DEVKIT V1 as well as select COM4 Serial Port (USB)**

### **Note: It should match with the port number for USB-Serial CH340 listed in device Manager. Depends on the system, this COM Port may have various number from 3 on wards. Check your device manager for the COM port through which your IoT Kit is connected to your system / laptop.**

### A screenshot of a computer Description automatically generated

A screenshot of a computer

Description automatically generated

**Testing a sample program**

A screenshot of a computer

Description automatically generated

void setup() {

  // initialize digital pin LED\_BUILTIN as an output.

  pinMode(LED\_BUILTIN, OUTPUT);

}

// the loop function runs over and over again forever

void loop() {

  digitalWrite(LED\_BUILTIN, HIGH);  // turn the LED on (HIGH is the voltage level)

  delay(1000);                      // wait for a second

  digitalWrite(LED\_BUILTIN, LOW);   // turn the LED off by making the voltage LOW

  delay(1000);                      // wait for a second

}

**In our ASC-IoT Kit, User LED 1 is connected to GPIO Pin 33 and so, LED\_BUILTIN is changed to 33.**

void setup() {

  // initialize digital pin LED\_BUILTIN as an output.

  pinMode(33, OUTPUT);

}

// the loop function runs over and over again forever

void loop() {

  digitalWrite(33, HIGH);  // turn the LED on (HIGH is the voltage level)

  delay(1000);                      // wait for a second

  digitalWrite(33, LOW);   // turn the LED off by making the voltage LOW

  delay(1000);                      // wait for a second

}

A screenshot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

Click Upload Button

A screenshot of a computer

Description automatically generated

Sketch uses 237117 bytes (18%) of program storage space. Maximum is 1310720 bytes.

Global variables use 21048 bytes (6%) of dynamic memory, leaving 306632 bytes for local variables. Maximum is 327680 bytes.

esptool.py v4.5.1

Serial port COM4

Connecting.......

Chip is ESP32-D0WD-V3 (revision v3.0)

Features: WiFi, BT, Dual Core, 240MHz, VRef calibration in efuse, Coding Scheme None

Crystal is 40MHz

MAC: c4:de:e2:10:70:ac

Uploading stub...

Running stub...

Stub running...

Changing baud rate to 921600

Changed.

Configuring flash size...

Flash will be erased from 0x00001000 to 0x00005fff...

Flash will be erased from 0x00008000 to 0x00008fff...

Flash will be erased from 0x0000e000 to 0x0000ffff...

Flash will be erased from 0x00010000 to 0x00049fff...

Compressed 17568 bytes to 12204...

Writing at 0x00001000... (100 %)

Wrote 17568 bytes (12204 compressed) at 0x00001000 in 0.5 seconds (effective 264.9 kbit/s)...

Hash of data verified.

Compressed 3072 bytes to 146...

Writing at 0x00008000... (100 %)

Wrote 3072 bytes (146 compressed) at 0x00008000 in 0.1 seconds (effective 301.9 kbit/s)...

Hash of data verified.

Compressed 8192 bytes to 47...

Writing at 0x0000e000... (100 %)

Wrote 8192 bytes (47 compressed) at 0x0000e000 in 0.2 seconds (effective 428.1 kbit/s)...

Hash of data verified.

Compressed 237488 bytes to 130712...

Writing at 0x00010000... (12 %)

Writing at 0x0001d273... (25 %)

Writing at 0x00024369... (37 %)

Writing at 0x0002953b... (50 %)

Writing at 0x0002eb93... (62 %)

Writing at 0x00037102... (75 %)

Writing at 0x0003f146... (87 %)

Writing at 0x000447c5... (100 %)

Wrote 237488 bytes (130712 compressed) at 0x00010000 in 2.6 seconds (effective 720.4 kbit/s)...

Hash of data verified.

Leaving...

Hard resetting via RTS pin...

After RESET, User LED User LED 1 at GPIO 33 blinks at the rate of 1 Second.

A green circuit board with a red light

Description automatically generated

**Designed by**

**Co-CPS and IoT TAG,**

**Amrita School of Computing, Coimbatore Amrita Vishwa Vidyapeetham**

**Manufactured and Marketed by JRM Technologies.**

**For more details, Contact**

Er A Balakrishna Veni +91 9361140290 [info@jrmtech.in](mailto:info@jrmtech.in) [asbkveni@gmail.com](mailto:asbkveni@gmail.com) JRM Tehnologies, (www.jrmtech.in)

Co-Workspace, CIR Building, Amrita Vishwa Vidyapeetham,

Amrita Nagar POST, Coimbatore – 641112.

**All the Best**

# \*\*\*\*\*\*\*\*\*