ESP32 Tool Installation steps: Windows Systems

Various types of ESP32 devices are available in the market (for better results use original Espressif devices and also use high-quality USB cables to connect the esp32 to computer for uploading /flashing the code.)

Step1: Esp-idf Environment Setup

1- Download the appropriate version of esp-idf for windows system
Here we are using esp-idf version 4.3.x

https://dl.espressif.com/dl/esp-idf/?idf=4.3

2- After installation two icons ESP-IDF CMD and ESP-IDF Power Shell will be in your desktop

3- Then open any of the two icons configure the environment by typing the following commands

For Power Shell:

./install.ps1
./export.ps1

For CMD:
./install.bat
./export.bat

4- Then we need to download and install USB to UART driver (CP2102) in order to detect our ESP32 devices

https://www.pololu.com/docs/0J7/all#2

or

https://www.silabs.com/developers/usb-to-uart-bridge-vcp-drivers

Now we can connect the ESP32 to our windows system and open the device manager to see the virtual COM port number

This is required to flash the code to the COM port number

After this esp-idf environment will be ready for application development

Now you can try hello_world example in the example folder

Step2: WiFi Sensing module configuration

1- Create a folder in C: drive

>mkdir esp

2- Download the wifi module from the following link and extract it put it in the *esp* folder

https://stevenmhernandez.github.io/ESP32-CSI-Tool/

For ACESS POINT (AP):

3- Then go to the esp folder (through power shell or CMD) and follow the following command

>cd active_ap (this is for access point)
>idf.py set-target esp32
>idf.py menuconfig

Change the following in the menuconfig window: The following configurations are important for this project:

1. Serial flasher config > 'idf.py monitor' baud rate > Custom Baud Rate
2. Serial flasher config > Custom baud rate value > 921600 This allows more data to be transmitted on the Serial port
3. Component config > Common ESP32-related > Channel for console output > Custom UART
4. Component config > Common ESP32-related > UART console baud rate > 921600
5. Component config > Wi-Fi > WiFi CSI(Channel State Information) (Press space to select)
6. Component config > FreeRTOS > Tick rate (Hz) > 1000
7. ESP32 CSI Tool Config > **** all options in this menu can be specified per your experiment requirements.

NOTE: For some systems, other baud rates may be required. Good options to try are 921600, 1000000, 1152000, 1500000, and 1552000

>idf.py build
>idf.py –p COMX flash (you need to check the com port number in the device)
>idf.py –p COMX monitor (not required for AP, monitor at the receiver side(STA))

For STA / RECEIVER:

4- Then go to the esp folder (through power shell or CMD) and follow the following command

>cd active_sta (this is for access point)
>idf.py set-target esp32
>idf.py menuconfig

Change the following in the menuconfig window: The following configurations are important for this project:

1. Serial flasher config > 'idf.py monitor' baud rate > Custom Baud Rate
2. Serial flasher config > Custom baud rate value > 921600 This allows more data to be transmitted on the Serial port
3. Component config > Common ESP32-related > Channel for console output > Custom UART
4. Component config > Common ESP32-related > UART console baud rate > 921600
5. Component config > Wi-Fi > WiFi CSI(Channel State Information) (Press space to select)
6. Component config > FreeRTOS > Tick rate (Hz) > 1000
7. ESP32 CSI Tool Config > **** all options in this menu can be specified per your experiment requirements.

NOTE: For some systems, other baud rates may be required. Good options to try are 921600, 1000000, 1152000, 1500000, and 1552000.

>idf.py build
>idf.py –p COMX flash (you need to check the com port number in device)
>idf.py –p COMX monitor

To write the data to a file the following commands

>idf.py monitor | findstr "CSI_DATA" > my-experiment-file.csv

For visualization of CSI data

>pip install numpy matplotlib
>idf.py monitor | python ../python_utils/serial_plot_csi_live.py

If you have linux system similar procedure can be followed.