ARC DESIGN CONTEST 2021

SYNOPSYS® Silicon to Software™

# Tutorial 4 – WE-I Introduction and Hands-on (Lab 1-3)

ARC DESIGN CONTEST 2021

SYNOPSYS®
Silicon to Software™

# WE-I Introduction

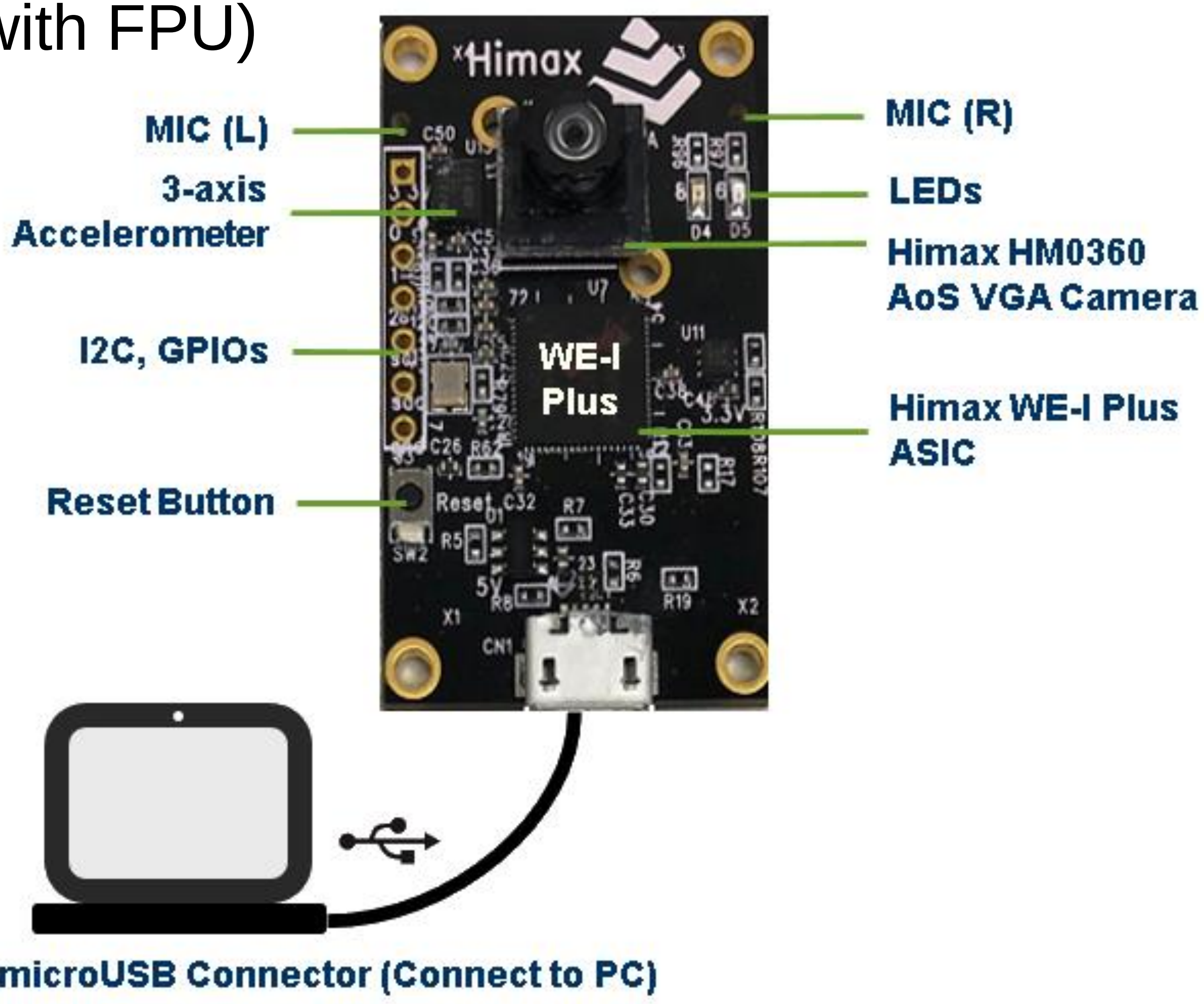
# WE-I Introduction

- CPU: HX6537 (ARC EM9D DSP with FPU)
- Frequency: 400MHz
- SPI program flash: 2MB
- Ram size: 2MB

320kB program ICCM

320kB data DCCM/XCCM/YCCM

1472kB system memory



MIC (L)

3-axis Accelerometer

I2C, GPIOs

Reset Button

MIC (R)

LEDs

Himax HM0360 AoS VGA Camera

Himax WE-I Plus ASIC

microUSB Connector (Connect to PC)

# WE-I Introduction

- UART: 1

  Use for USB VCP

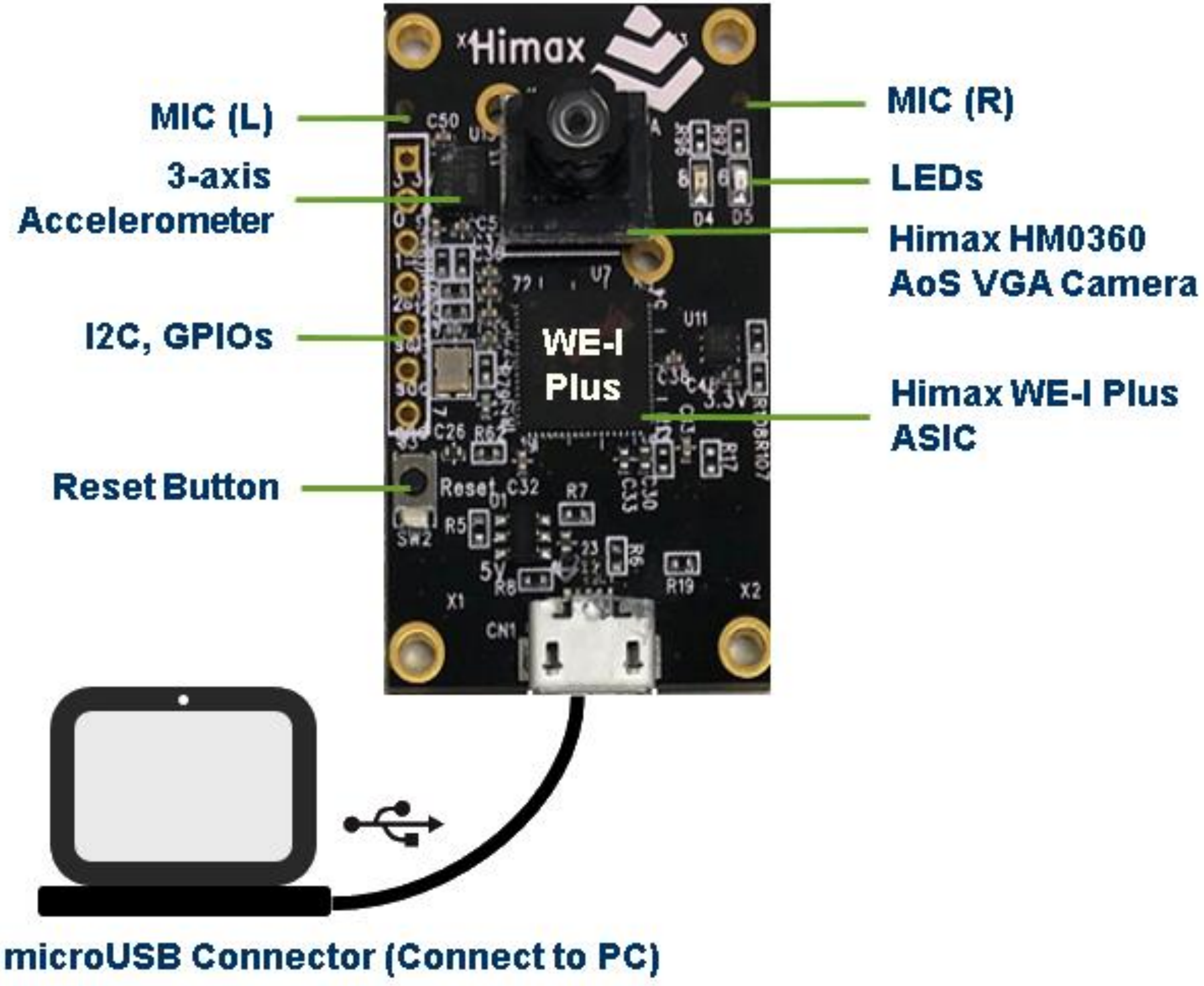
- SPI: 1

  Use for USB VCP

- I2C Master: 2

  1 for IMU, 1 for user use

- GPIO: 3



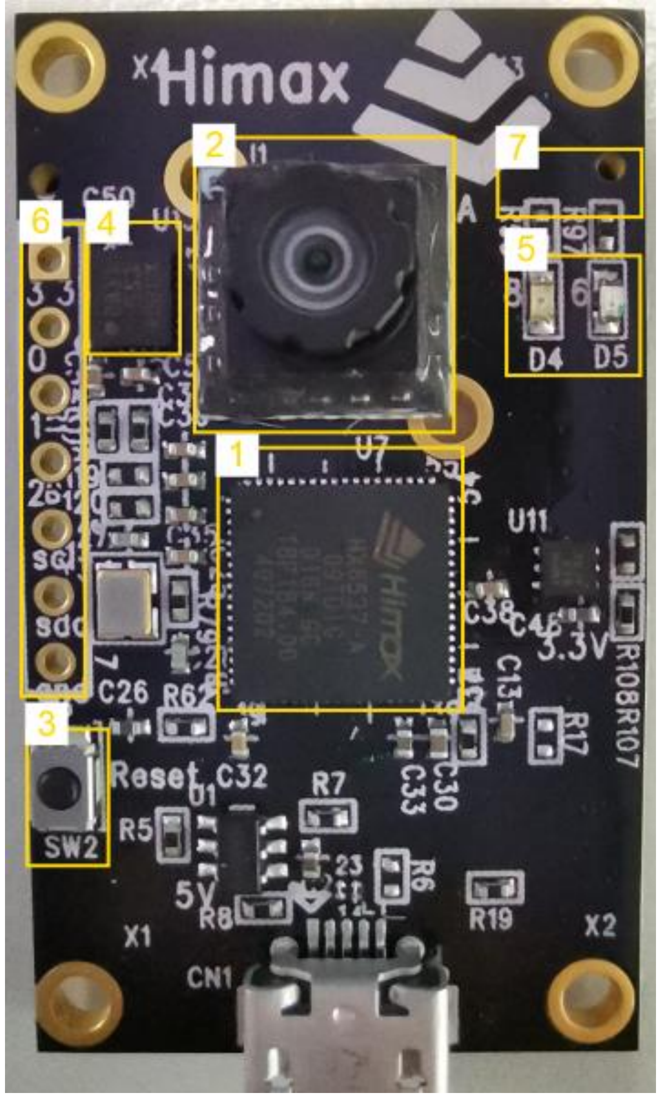
# WE-I Introduction

1. CPU HX6537
2. HM0360 AoSTM VGA camera

   Back Side Illuminated (BSI) CMOS Image Sensor

   Active Pixel Array: 656 x 496

   Frame Rate: QQVGA 1FPS to VGA 60FPS
3. Reset Button



# WE-I Introduction

4. LSM9DS1 IMU sensor

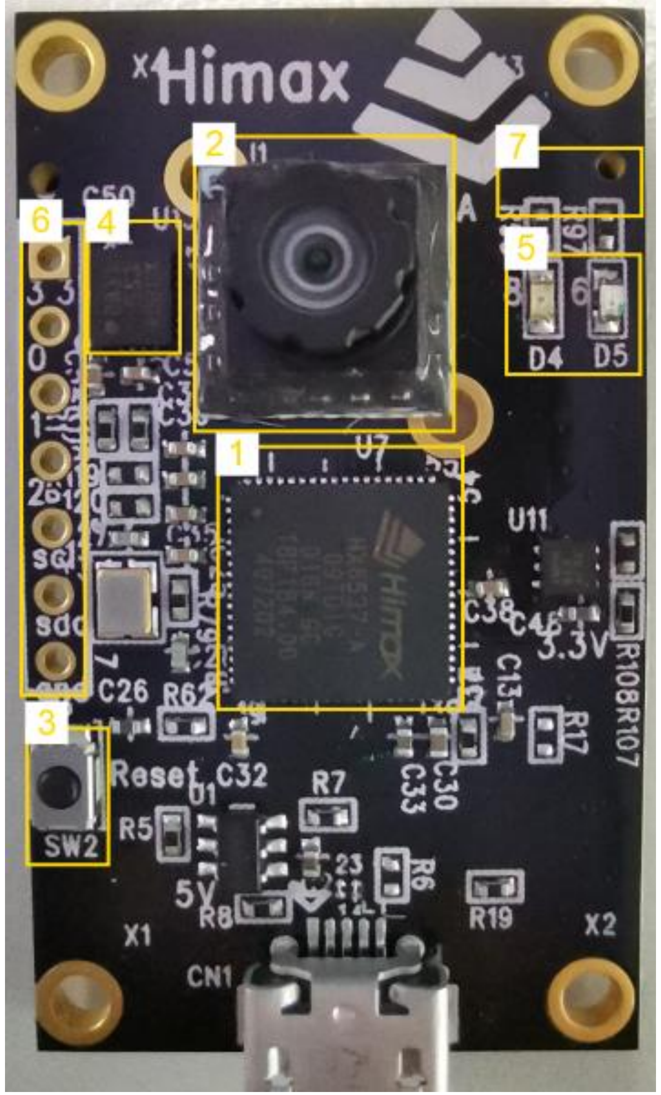
   3 acceleration: ±2 / ±4 / ±8 / ±16 g

   3 angular rate : ±245 / ±500 / ±2000 dps

   3 magnetic field: ±4 / ±8 / ±12 / ±16 gauss

   16-bit data output

5. Green & red LED



# WE-I Introduction

6. Header 7*1 pitch=2.54

   Pin1: 3V3

   Pin2: GPIO0

   Pin3: GPIO1

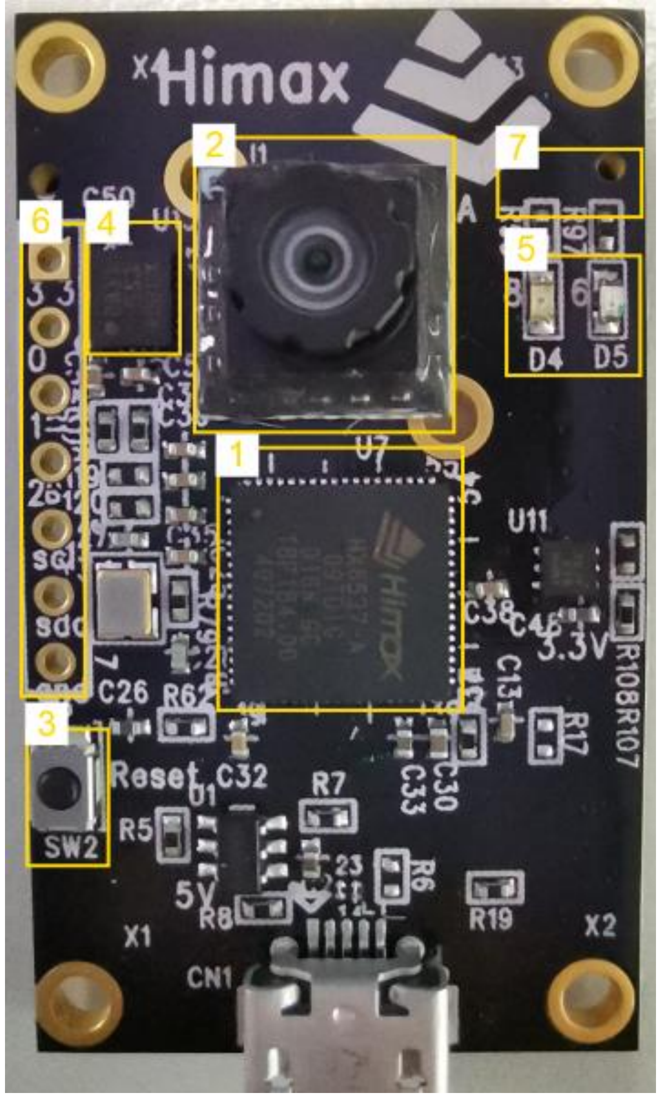
   Pin4: GPIO2

   Pin5: I2C_M1_SCL

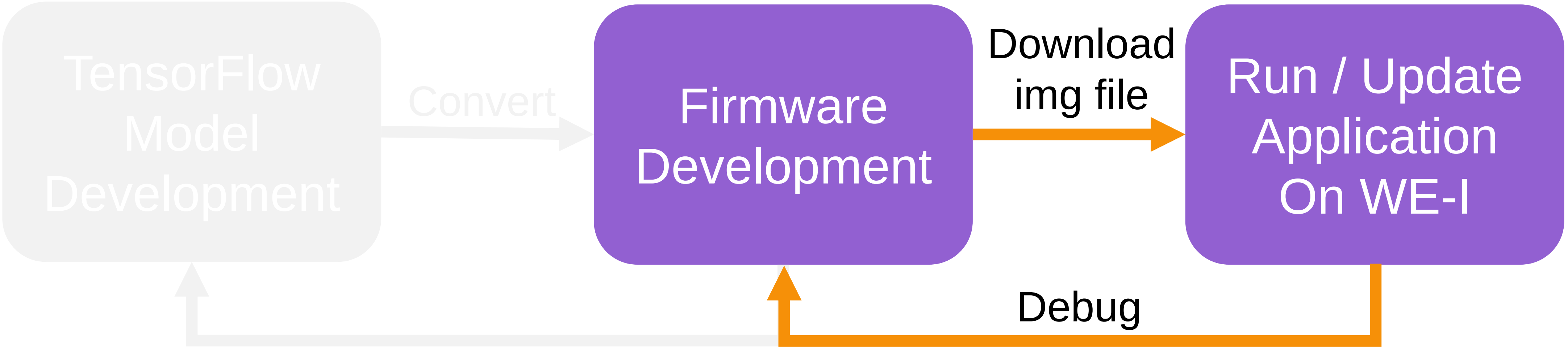
   Pin6: I2C_M1_SDA

   Pin7: GND

7. Microphones (L/R) at back side



# WE-I Project Development Flow



TensorFlow Model Development

Convert

Firmware Development

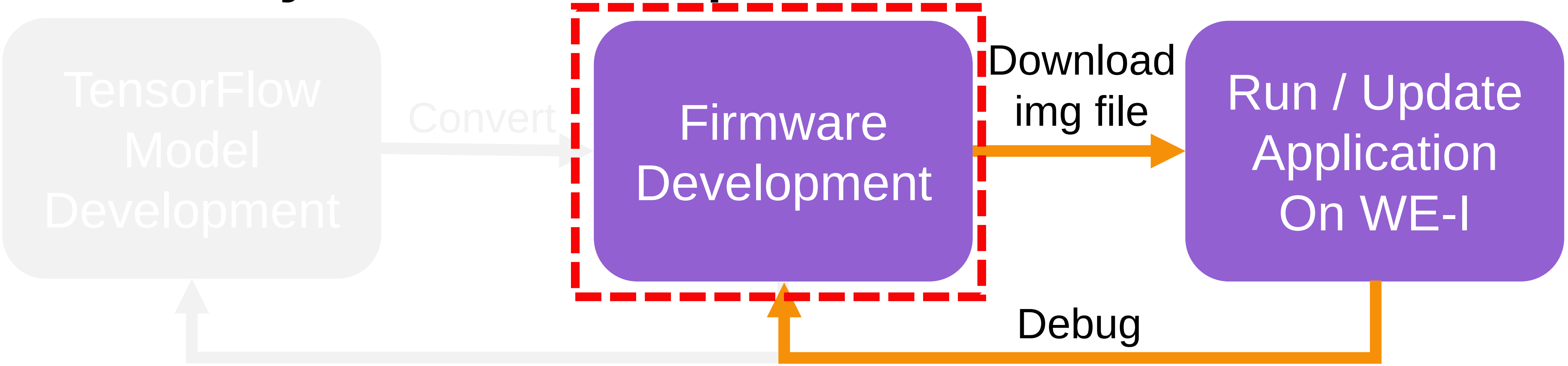
Download img file

Run / Update Application On WE-I

Debug

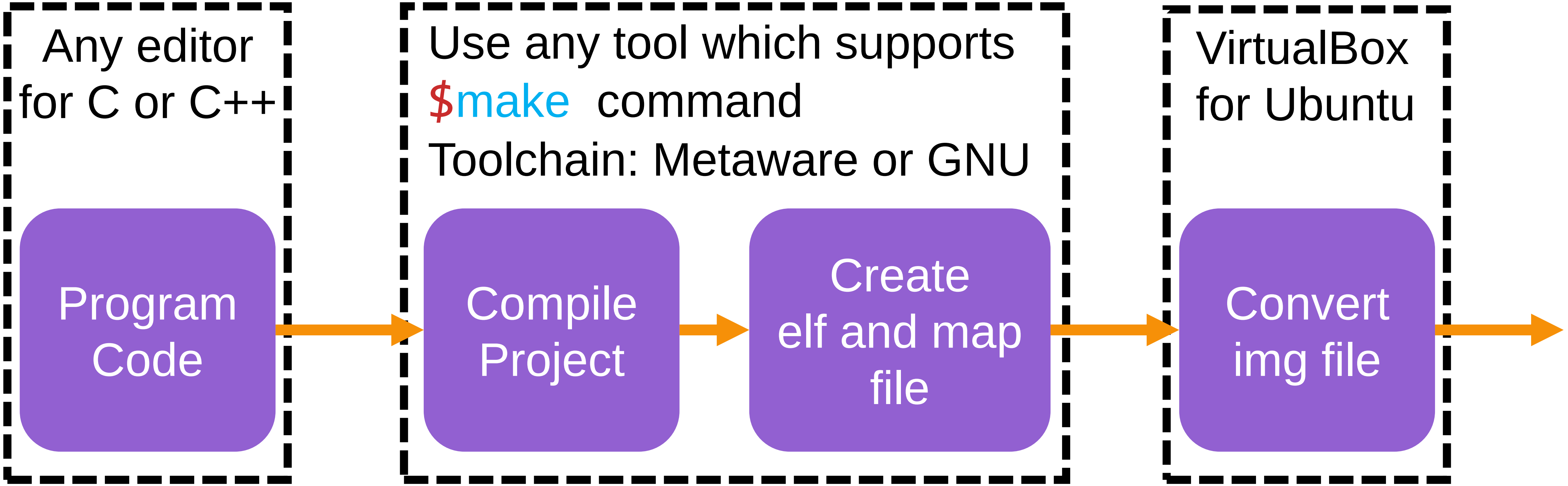
| **Stage** | **TensorFlow Model Development** | **Firmware Development** | **Run / Update Application On WE-I** |
|---|---|---|---|
| Tool | Anaconda<br>Cygwin | Cygwin<br>Metaware or ARC GNU<br>VirtualBox (Ubuntu 20.04) | Tera Term<br>USB Micro |
| Language | Python 3 | C language<br>C++ language | |

# WE-I Project Development Flow


TensorFlow Model Development
Convert
Firmware Development
Download img file
Run / Update Application On WE-I
Debug

| **Stage** | **TensorFlow Model Development** | **Firmware Development** | **Run / Update Application On WE-I** |
|---|---|---|---|
| Tool | Anaconda<br>Cygwin | Cygwin<br>Metaware or ARC GNU<br>VirtualBox (Ubuntu 20.04) | Tera Term<br>USB Micro |
| Language | Python 3 | C language<br>C++ language | |

# Firmware Development



Any editor for C or C++

Program Code

Use any tool which supports $make command

Toolchain: Metaware or GNU

Compile Project

Create elf and map file

VirtualBox for Ubuntu

Convert img file

# Example Project Download (Already done in Tutorial-2)

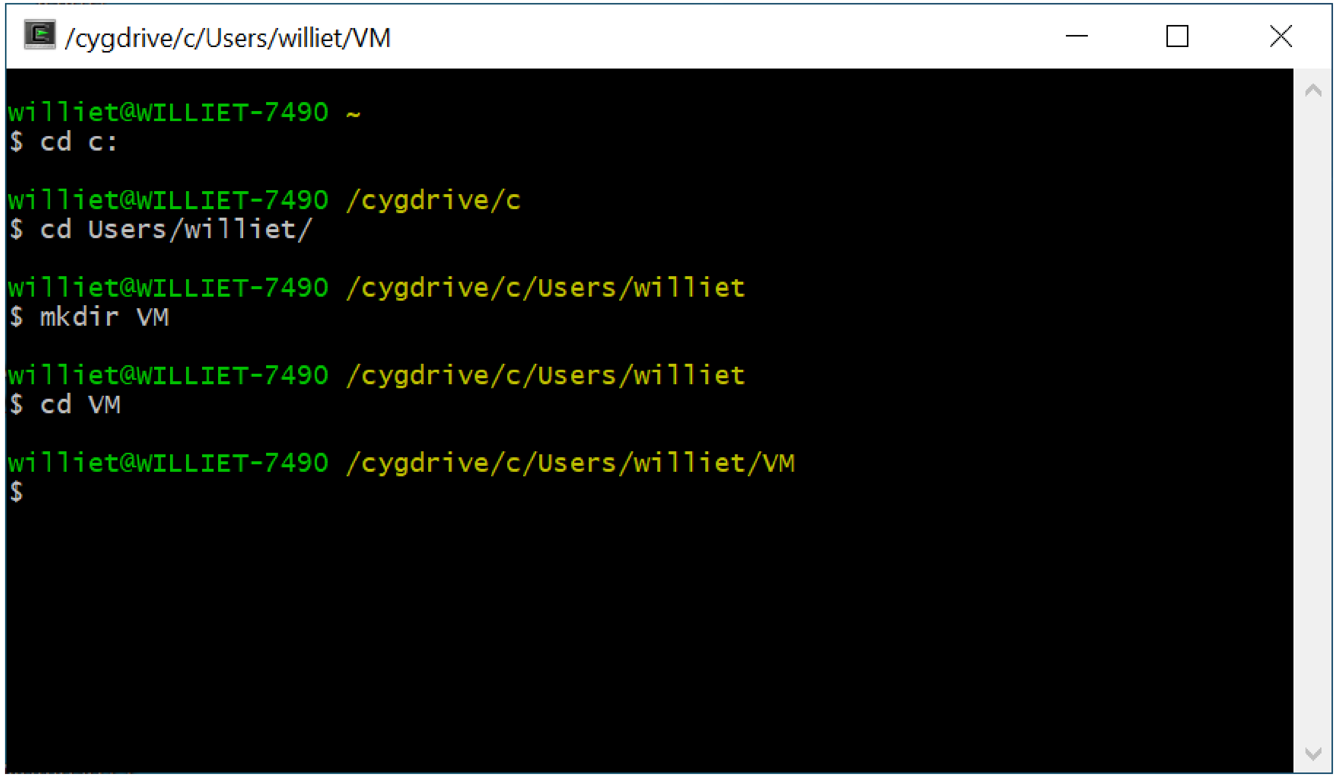
1. Open Cygwin64 Terminal

   $ cd c:

   $ cd Users/{username}/ (to your working file path)

   $ mkdir VM (Suggest create a new folder named “VM”)

   $ cd VM



/cygdrive/c/Users/williet/VM

```
williet@WILLIET-7490 ~
$ cd c:

williet@WILLIET-7490 /cygdrive/c
$ cd Users/williet/

williet@WILLIET-7490 /cygdrive/c/Users/williet
$ mkdir VM

williet@WILLIET-7490 /cygdrive/c/Users/williet
$ cd VM

williet@WILLIET-7490 /cygdrive/c/Users/williet/VM
$
```

# Example Project Download (Already done in Tutorial-2)

Commands in cygwin64 terminal

2. Download SDK from Synopsys Github

```
$ git clone https://github.com/foss-for-synopsys-dwc-arc-processors/arc_contest.git
$ cd arc_contest
$ git submodule init
$ git submodule update
$ cd himax_tflm
$ make download
```

# Example Project Download (Already done in Tutorial-2)

After these steps, your file structure will be like:

```
arc_contest
|
---- bsp_tflu
---- doc_tutorial
---- himax_tflm
        |
        ---- himax_we1_sdk
        ---- image_gen_linux
        ---- tensorflow
        ---- third_party
---- Synopsys_SDK
        |
        ---- Example_Project
        ---- User_Project
```

# Himax SDK

- “……/*arc_contest/*himax_tflm/himax_we1_sdk/hx_drv_tflm.h”

# Synopsys SDK

- “……/*arc_contest/*Synopsys_SDK/Example_Project”
  Example Project from Lab1~6, you can copy or reference it.

- “……/*arc_contest/*Synopsys_SDK/User_Project”
  Please develop your project here.
  Make sure your project file structure is the same as Example_Project.

- For example: “*arc_contest/*Synopsys_SDK/Example_Project/Lab1_uart”
  You will see folder “src” and “inc”
  “src” folder: always keep your .c and .cc file in here.
  “inc” folder: always keep your .h file in here.
  (c file: c language)
  (cc file: c++ language)

# Make Project and Flash File

There are some commands can be used,
$ make : compile and link your project, then create .elf and .map file
$ make flash : combine .elf and .map file to .img file
$ make clean : remove all .o file of this project
$ make clean_all : remove all .o file of this project and third party

You can add a command for changing toolchain
(default toolchain is gnu, define in makefile)
“ARC_TOOLCHAIN=mwdt”: compile with MetaWare
“ARC_TOOLCHAIN=gnu”: compile with ARC GNU Toolchain
Please use $ make clean_all before you change toolchain.

# Hands-on (Lab 1): UART

# Lab1: UART

- UART initial

```
extern HX_DRV_ERROR_E hx_drv_uart_initial(HX_DRV_UART_BAUDRATE_E baud_rate);

// UART initial API, should be called first before you use UART

// (HX_DRV_UART_BAUDRATE_E baud_rate) options are bellow
    UART_BR_9600 = 0,        /**< UART bard rate 9600bps */
    UART_BR_14400 = 1,       /**< UART bard rate 14400bps */
    UART_BR_19200 = 2,       /**< UART bard rate 19200bps */
    UART_BR_38400 = 3,       /**< UART bard rate 38400bps */
    UART_BR_57600 = 4,       /**< UART bard rate 57600bps */
    UART_BR_115200 = 5,      /**< UART bard rate 115200bps */
    UART_BR_230400 = 6,      /**< UART bard rate 230400bps */
    UART_BR_460800 = 7,      /**< UART bard rate 460800bps */
    UART_BR_921600 = 8,      /**< UART bard rate 921600bps */

Ex: hx_drv_uart_initial(UART_BR_115200); //Initial and baud is 115200
```

# Lab1: UART

- UART send data

```
extern HX_DRV_ERROR_E hx_drv_uart_print(const char*fmt, ...);
// Print message to UART port
// It will show on terminal. (Need to set correct port and baud)

Ex: hx_drv_uart_print("URAT_GET_STRING_START\n");
Ex: hx_drv_uart_print("String cnt: %d\n\n", uart_rx_cnt);  //Not support %f
Ex: hx_drv_uart_print("Echo string: %s\n", uart_rx_str);
```

# Lab1: UART

- UART get data

```
extern HX_DRV_ERROR_E hx_drv_uart_getchar(uint8_t *pch);

// return HX_DRV_LIB_PASS: UART RX FIFO has data, and put 1 byte to *pch

// return HX_DRV_LIB_ERROR: Operation fail

// return HX_DRV_LIB_NODATA: Nothing get back


Ex: hx_drv_uart_getchar(&data_buf);
```

# Lab1: UART

## Conclusion

- UART should initialize before send and get UART data.

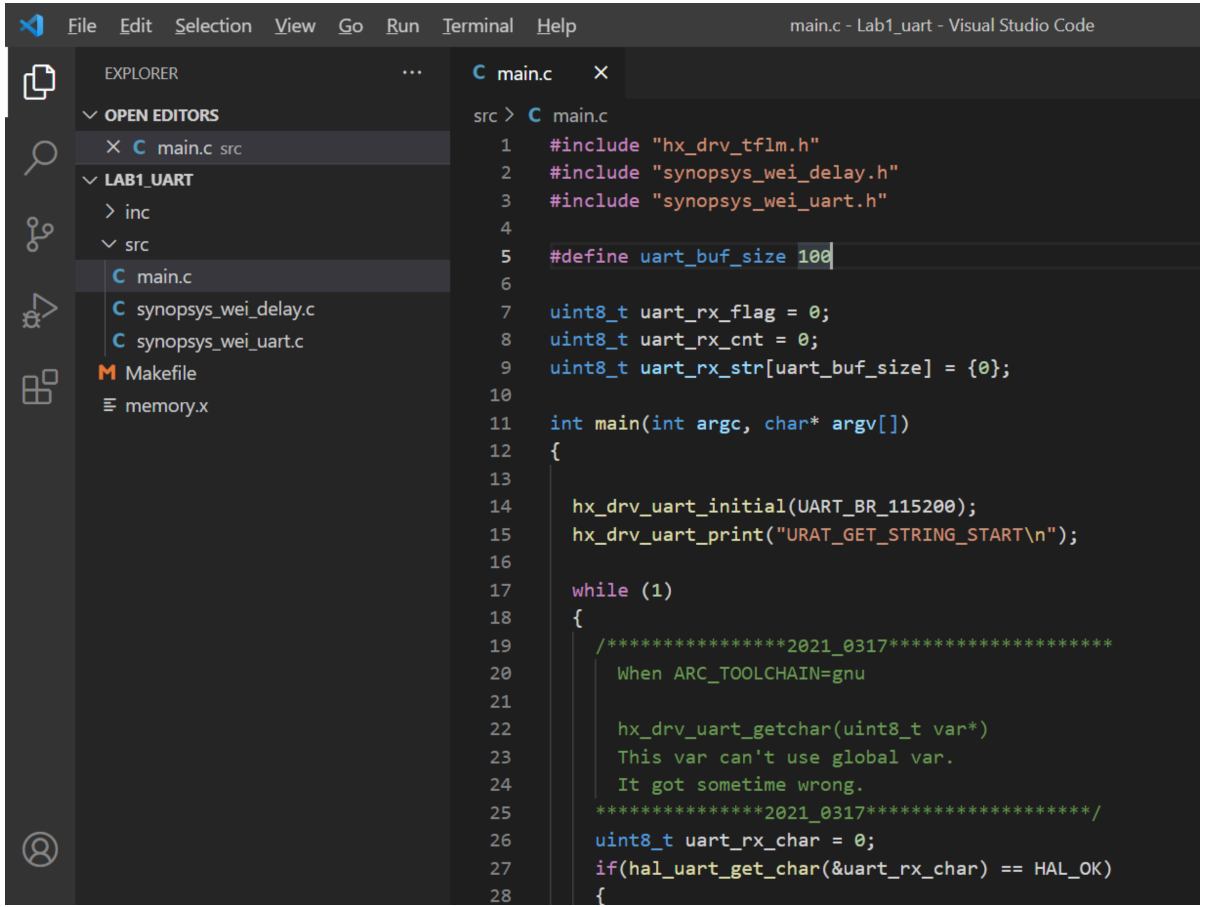
```
HX_DRV_ERROR_E hx_drv_uart_initial(HX_DRV_UART_BAUDRATE_E baud_rate);
```

- After you initialize UART, you can send and get UART data.

```
HX_DRV_ERROR_E hx_drv_uart_print(const char*fmt, ...);
HX_DRV_ERROR_E hx_drv_uart_getchar(uint8_t *pch);
```

# Lab1: UART

- Open Visual Studio Code & open folder
  "....../arc_contest/Synopsys_SDK/Example_Project/Lab1_uart"
- Open "src/main.c"

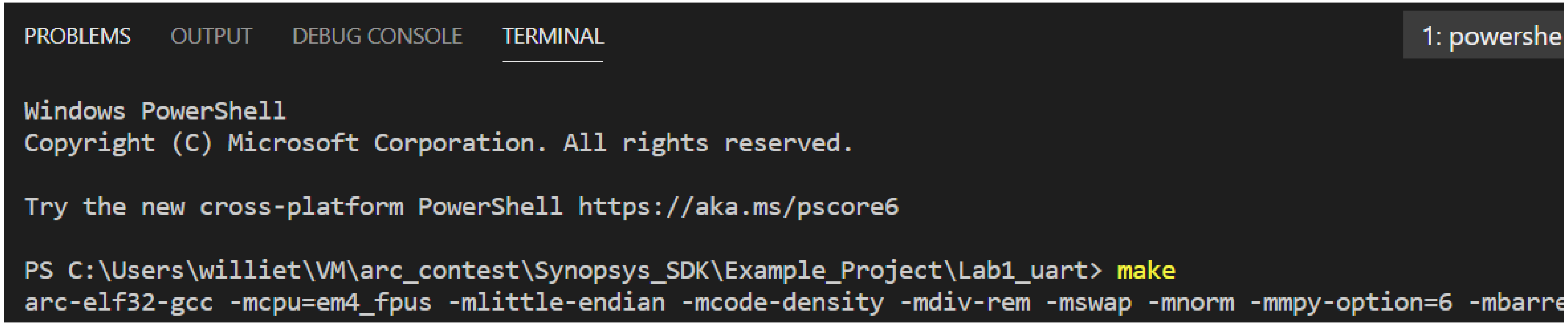


```c
#include "hx_drv_tflm.h"
#include "synopsys_wei_delay.h"
#include "synopsys_wei_uart.h"

#define uart_buf_size 100

uint8_t uart_rx_flag = 0;
uint8_t uart_rx_cnt = 0;
uint8_t uart_rx_str[uart_buf_size] = {0};

int main(int argc, char* argv[])
{

  hx_drv_uart_initial(UART_BR_115200);
  hx_drv_uart_print("URAT_GET_STRING_START\n");

  while (1)
  {
    /****************2021_0317********************
      When ARC_TOOLCHAIN=gnu

      hx_drv_uart_getchar(uint8_t var*)
      This var can't use global var.
      It got sometime wrong.
    ***************2021_0317********************/
    uint8_t uart_rx_char = 0;
    if(hal_uart_get_char(&uart_rx_char) == HAL_OK)
    {
```

# Lab1: UART

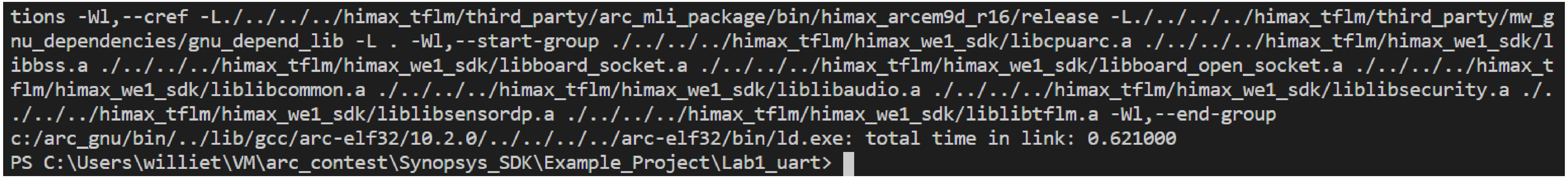
- Open Terminal and key-in

make



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\williet\VM\arc_contest\Synopsys_SDK\Example_Project\Lab1_uart> make
arc-elf32-gcc -mcpu=em4_fpus -mlittle-endian -mcode-density -mdiv-rem -mswap -mnorm -mmpy-option=6 -mbarre
```

- Make sure there are no error message



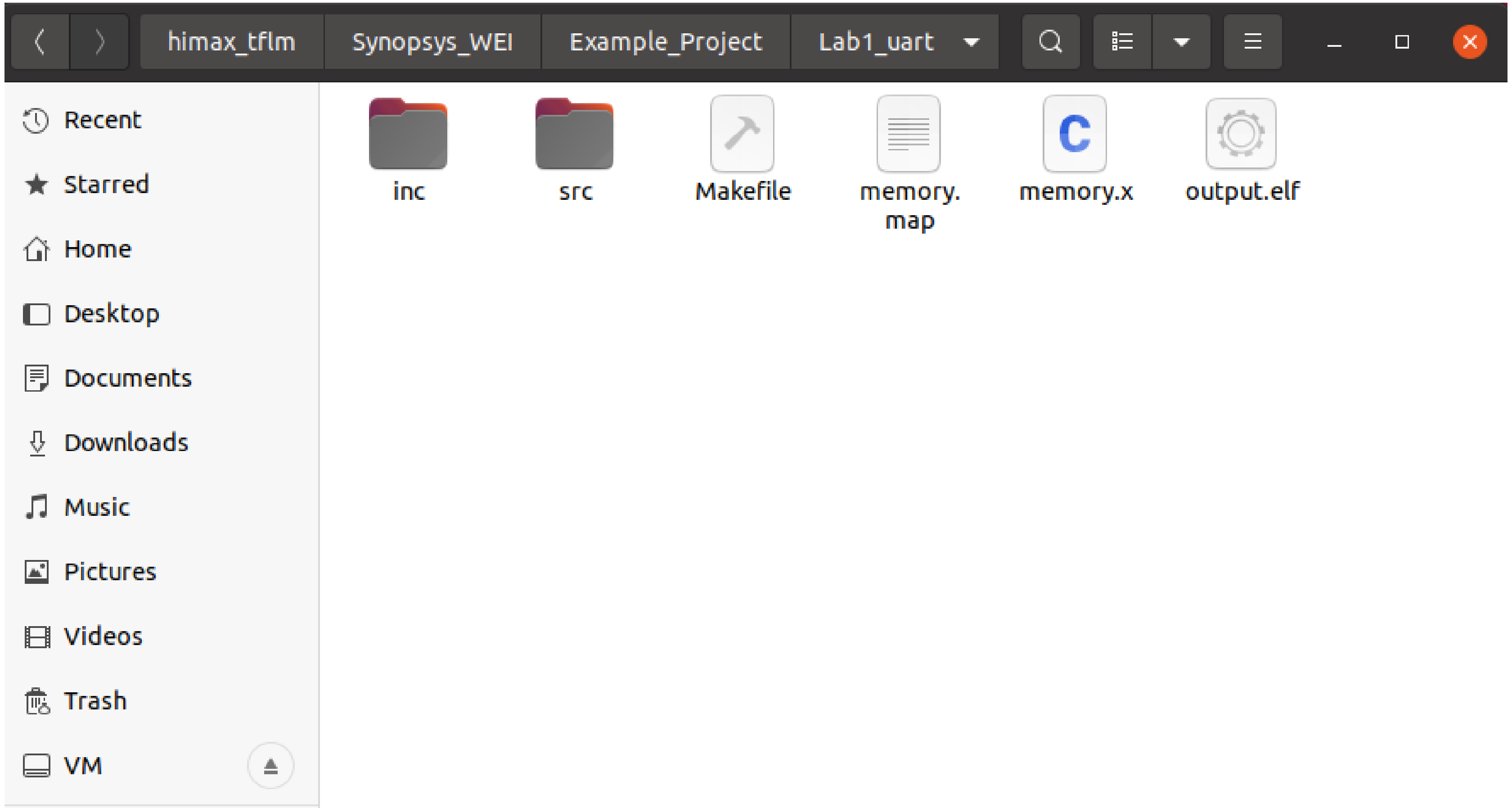
```
tions -Wl,--cref -L./../../../himax_tflm/third_party/arc_mli_package/bin/himax_arcem9d_r16/release -L./../../../himax_tflm/third_party/mw_g
nu_dependencies/gnu_depend_lib -L . -Wl,--start-group ./../../../himax_tflm/himax_we1_sdk/libcpuarc.a ./../../../himax_tflm/himax_we1_sdk/l
ibbss.a ./../../../himax_tflm/himax_we1_sdk/libboard_socket.a ./../../../himax_tflm/himax_we1_sdk/libboard_open_socket.a ./../../../himax_t
flm/himax_we1_sdk/liblibcommon.a ./../../../himax_tflm/himax_we1_sdk/liblibaudio.a ./../../../himax_tflm/himax_we1_sdk/liblibsecurity.a ./.
./../../himax_tflm/himax_we1_sdk/liblibsensordp.a ./../../../himax_tflm/himax_we1_sdk/liblibtflm.a -Wl,--end-group
c:/arc_gnu/bin/../lib/gcc/arc-elf32/10.2.0/../../../../arc-elf32/bin/ld.exe: total time in link: 0.621000
PS C:\Users\williet\VM\arc_contest\Synopsys_SDK\Example_Project\Lab1_uart>
```

- You will get output.elf and memory.map to convert image file

# Lab1: UART

- Open Virtual Machine Ubuntu and go to same project path

  {Share Folder…}\arc_contest\Synopsys_SDK\Example_Project\Lab1_uart

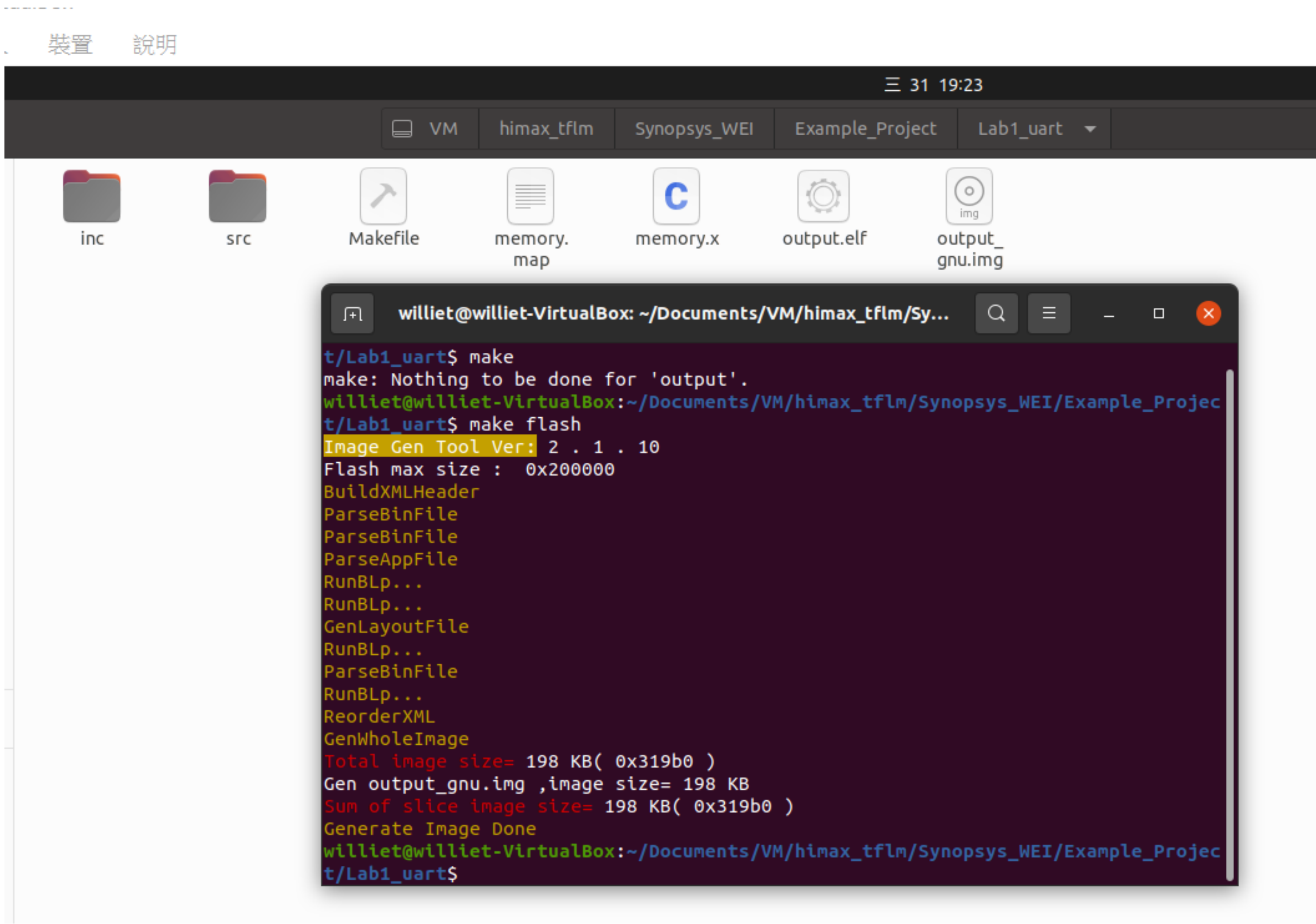


# Lab1: UART

- Open Terminal and key-in

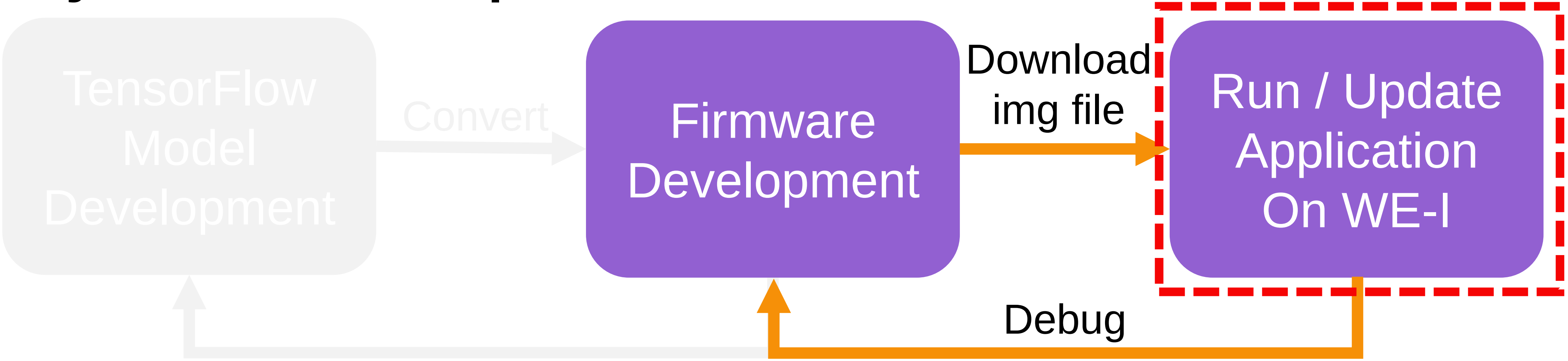
  make flash

- You will get output_gnu.img



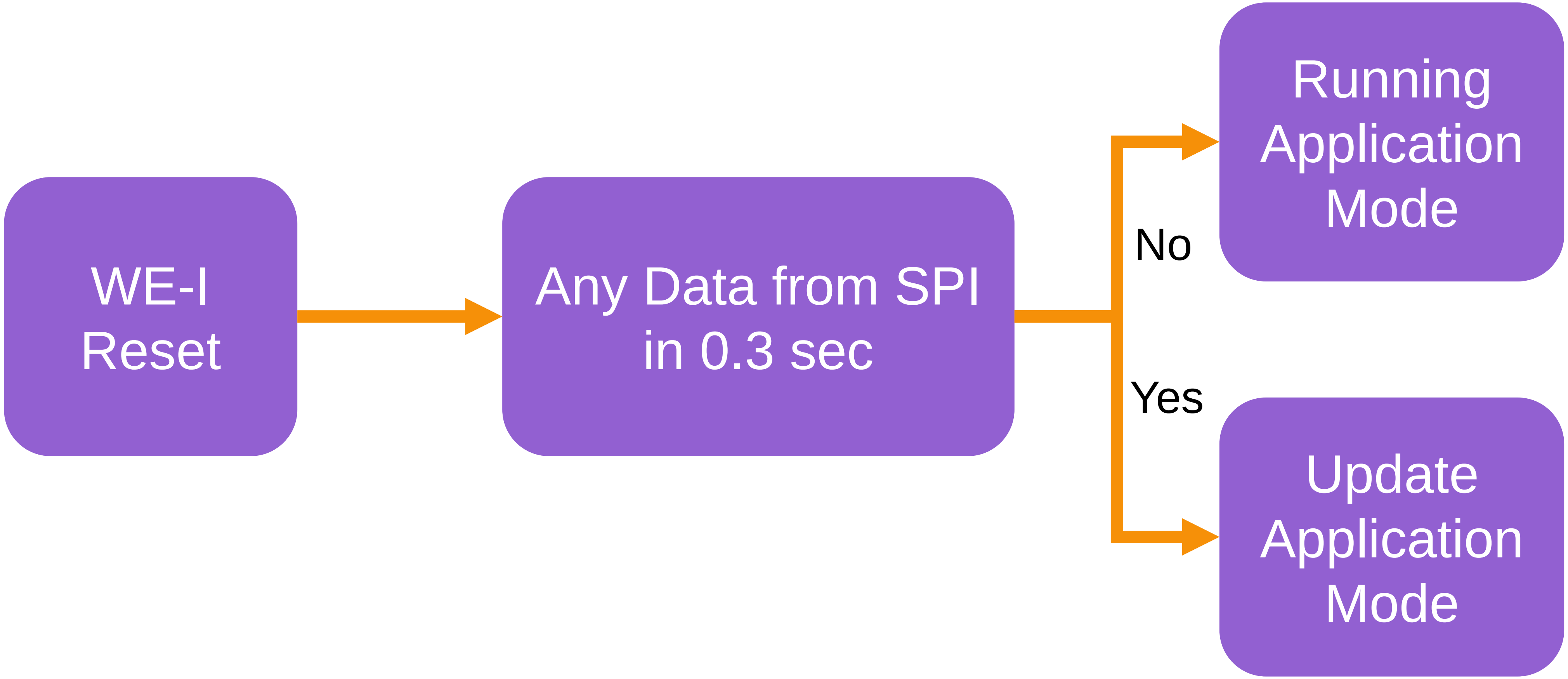
```
t/Lab1_uart$ make
make: Nothing to be done for 'output'.
williet@williet-VirtualBox:~/Documents/VM/himax_tflm/Synopsys_WEI/Example_Projec
t/Lab1_uart$ make flash
Image Gen Tool Ver: 2 . 1 . 10
Flash max size :  0x200000
BuildXMLHeader
ParseBinFile
ParseBinFile
ParseAppFile
RunBLp...
RunBLp...
GenLayoutFile
RunBLp...
ParseBinFile
RunBLp...
ReorderXML
GenWholeImage
Total image size= 198 KB( 0x319b0 )
Gen output_gnu.img ,image size= 198 KB
Sum of slice image size= 198 KB( 0x319b0 )
Generate Image Done
williet@williet-VirtualBox:~/Documents/VM/himax_tflm/Synopsys_WEI/Example_Projec
t/Lab1_uart$
```

# Project Development Flow

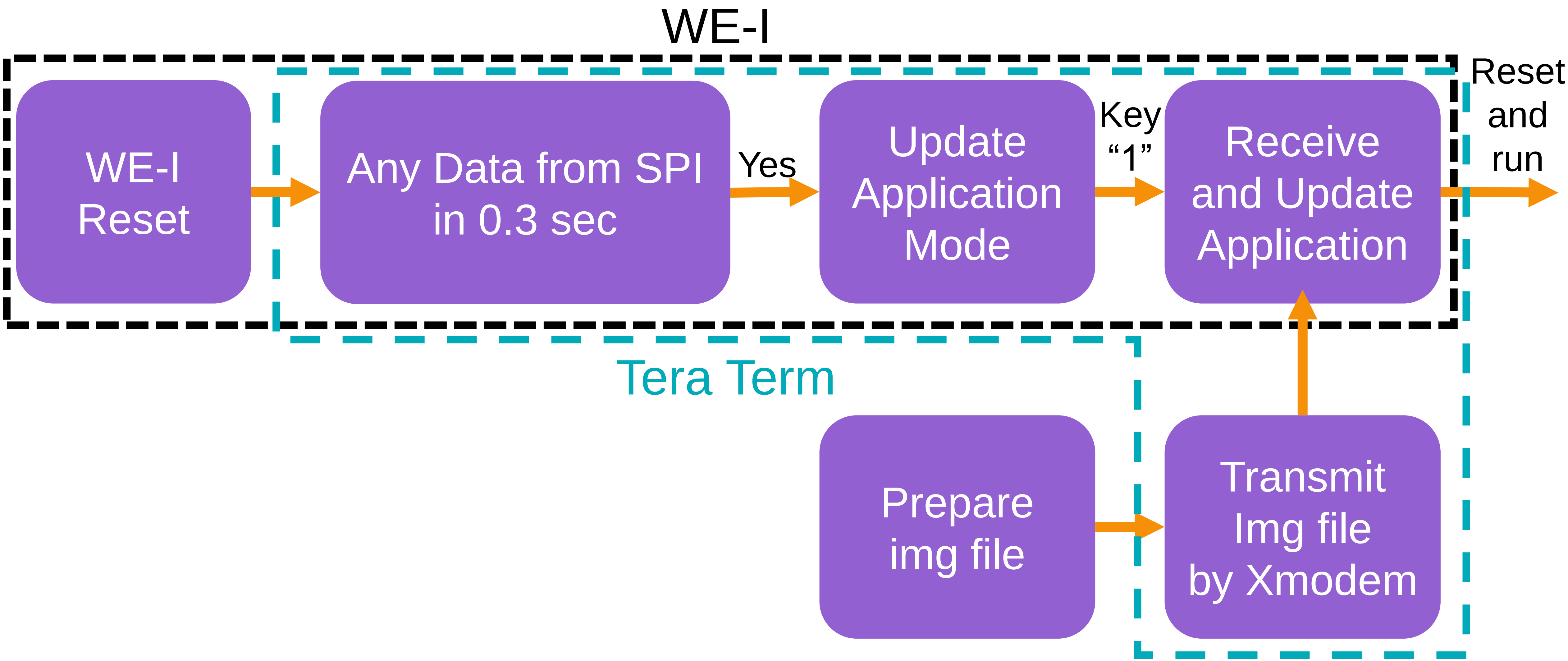

TensorFlow Model Development
Convert
Firmware Development
Download img file
Run / Update Application On WE-I
Debug

| **Stage** | **TensorFlow Model Development** | **Firmware Development** | **Run / Update Application On WE-I** |
|---|---|---|---|
| Tool | Anaconda<br>Cygwin | Cygwin<br>Metaware or ARC GNU<br>VirtualBox (Ubuntu 20.04) | Tera Term<br>USB Micro |
| Language | Python 3 | C language<br>C++ language | |

# Run / Update Application On WE-I

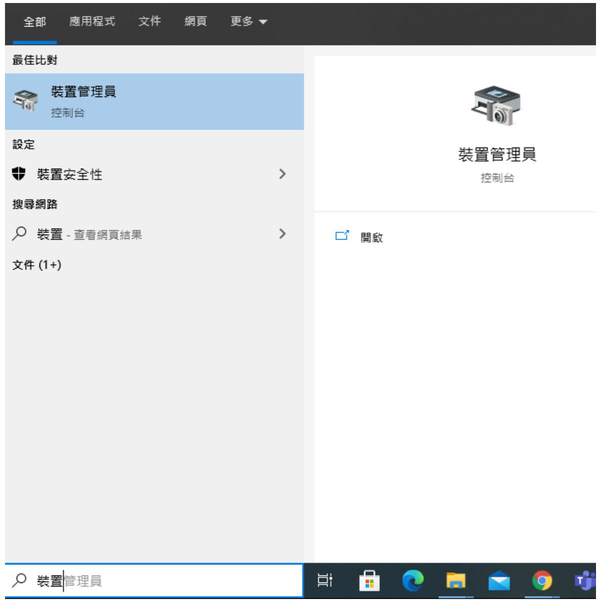


# Update Application On WE-I

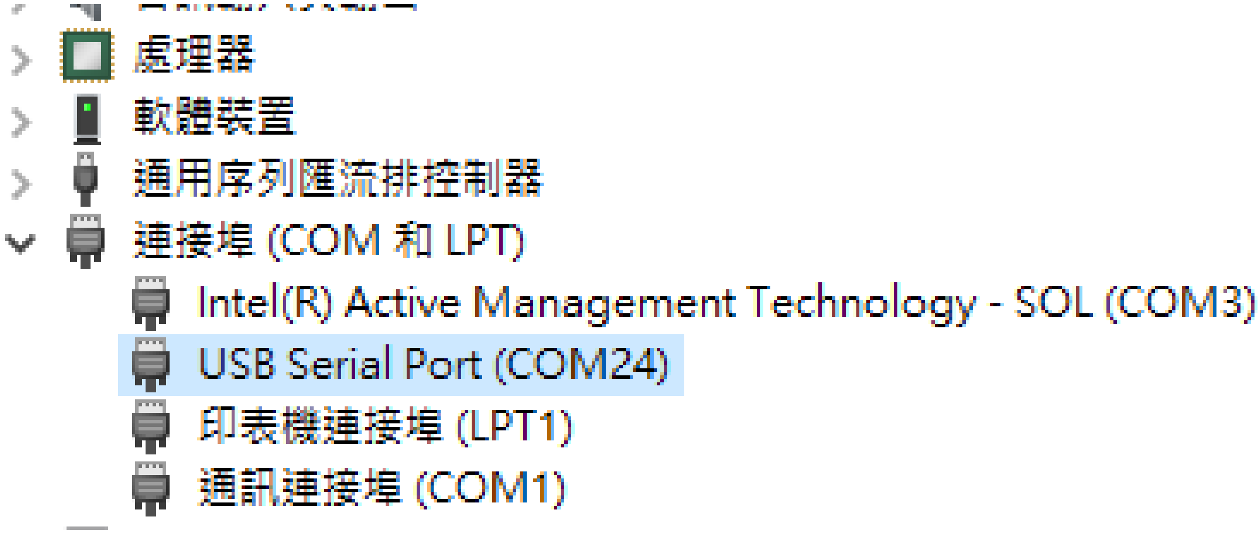

WE-I
WE-I Reset
Any Data from SPI in 0.3 sec
Yes
Update Application Mode
Key "1"
Receive and Update Application
Reset and run
Tera Term
Prepare img file
Transmit Img file by Xmodem

# Lab1: UART

- Connect WE-I and PC by USB Cable
- Check your WE-I usb port number
  裝置管理員> 連接埠(COM & LPT) > USB Serial Port (COMx)
  x: This is your WE-I usb port number
  (If USB Serial Port is not shown here, please refer to Appendix-2)

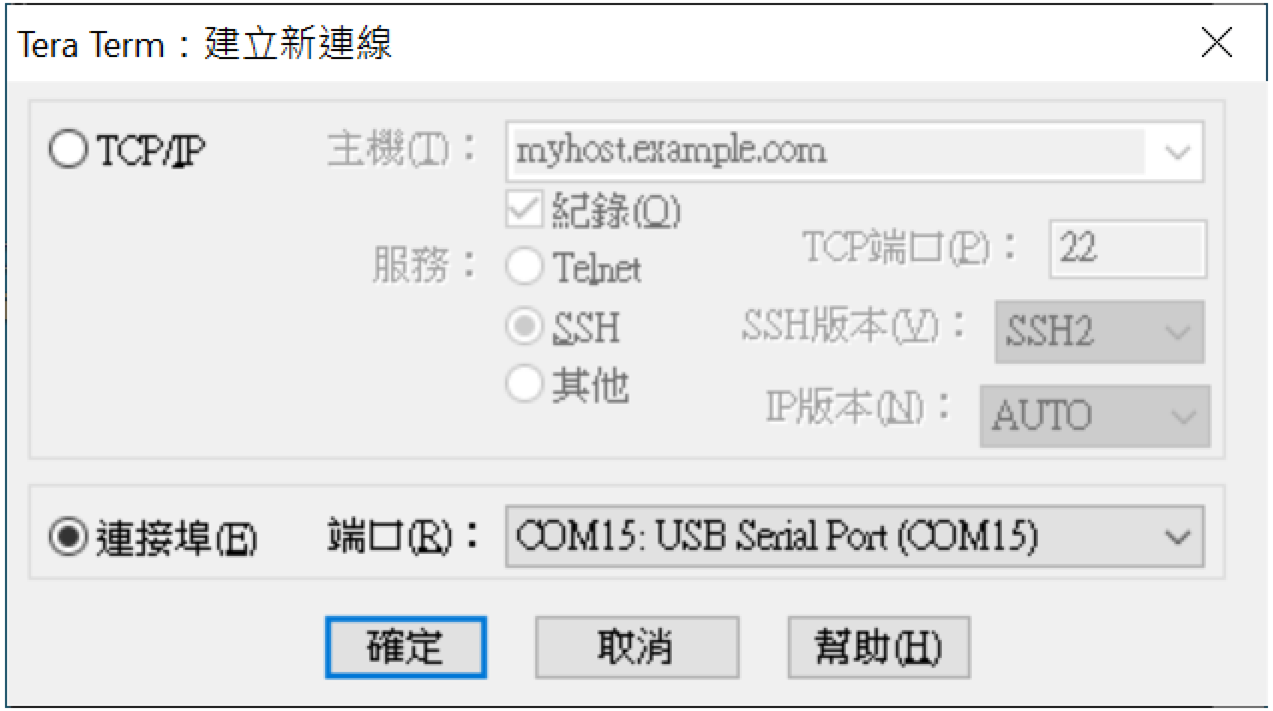


裝置管理員


處理器
軟體裝置
通用序列匯流排控制器
連接埠 (COM 和 LPT)
Intel(R) Active Management Technology - SOL (COM3)
USB Serial Port (COM24)
印表機連接埠 (LPT1)
通訊連接埠 (COM1)

# Lab1: UART

- Open TeraTerm, set your COM port



Tera Term：建立新連線

○ TCP/IP 主機(T)：myhost.example.com

☑ 紀錄(O)

服務：○ Telnet ● SSH ○ 其他

TCP端口(P)：22

SSH版本(V)：SSH2

IP版本(N)：AUTO

● 連接埠(E) 端口(R)：COM15: USB Serial Port (COM15)

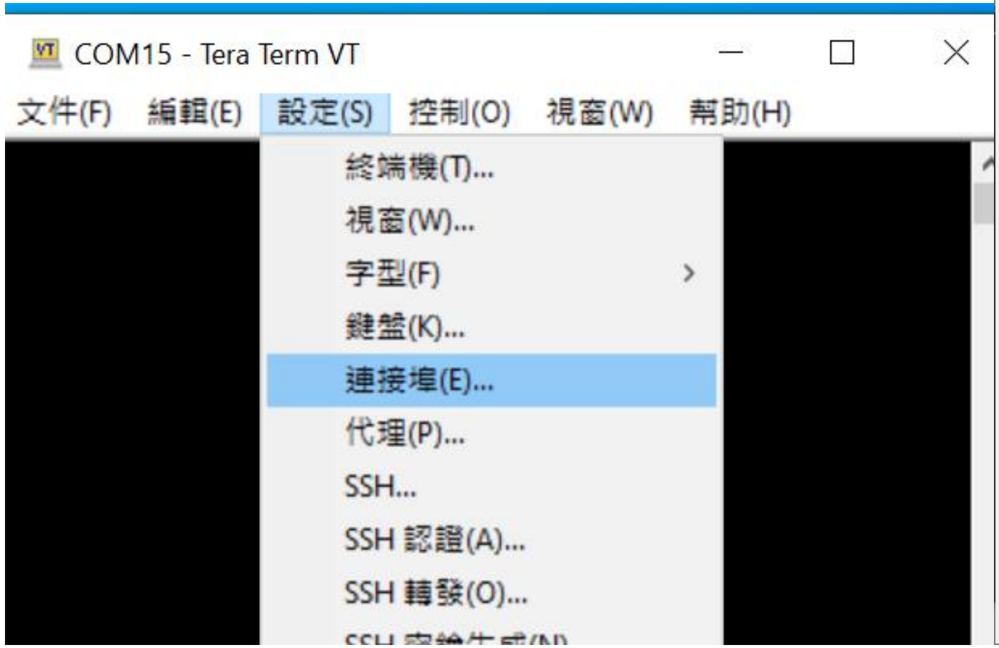
# Lab1: UART

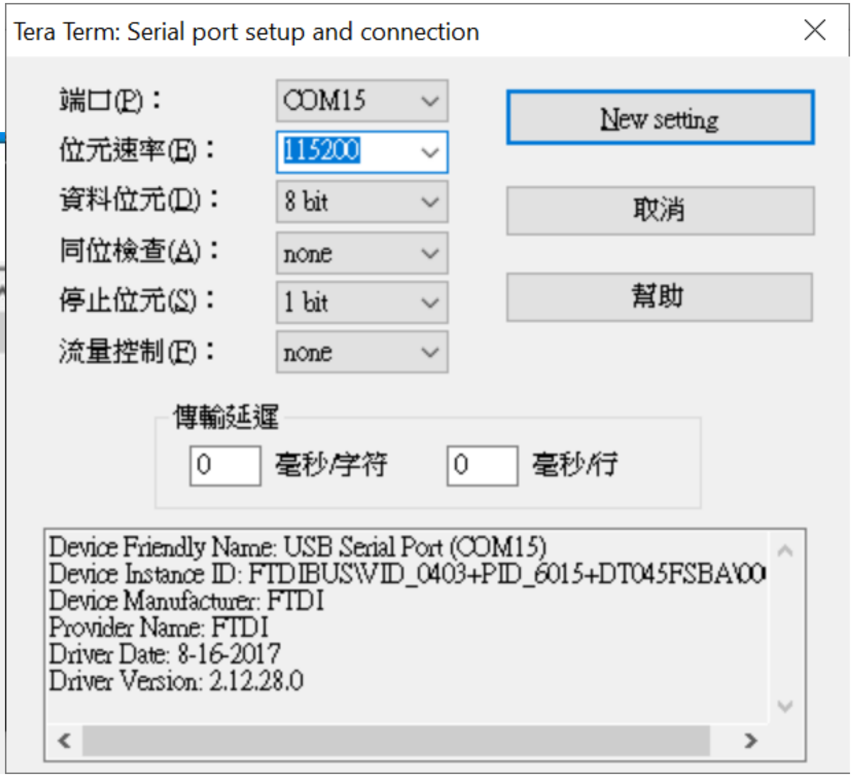
- Set your baud rate

  When send image file, baud rate always use 115200

  For your program run, you can select any baud rate.

  We suggest you always use 115200.




Tera Term: Serial port setup and connection

端口(P)： COM15

位元速率(B)： 115200

資料位元(D)： 8 bit

同位檢查(A)： none

停止位元(S)： 1 bit

流量控制(F)： none

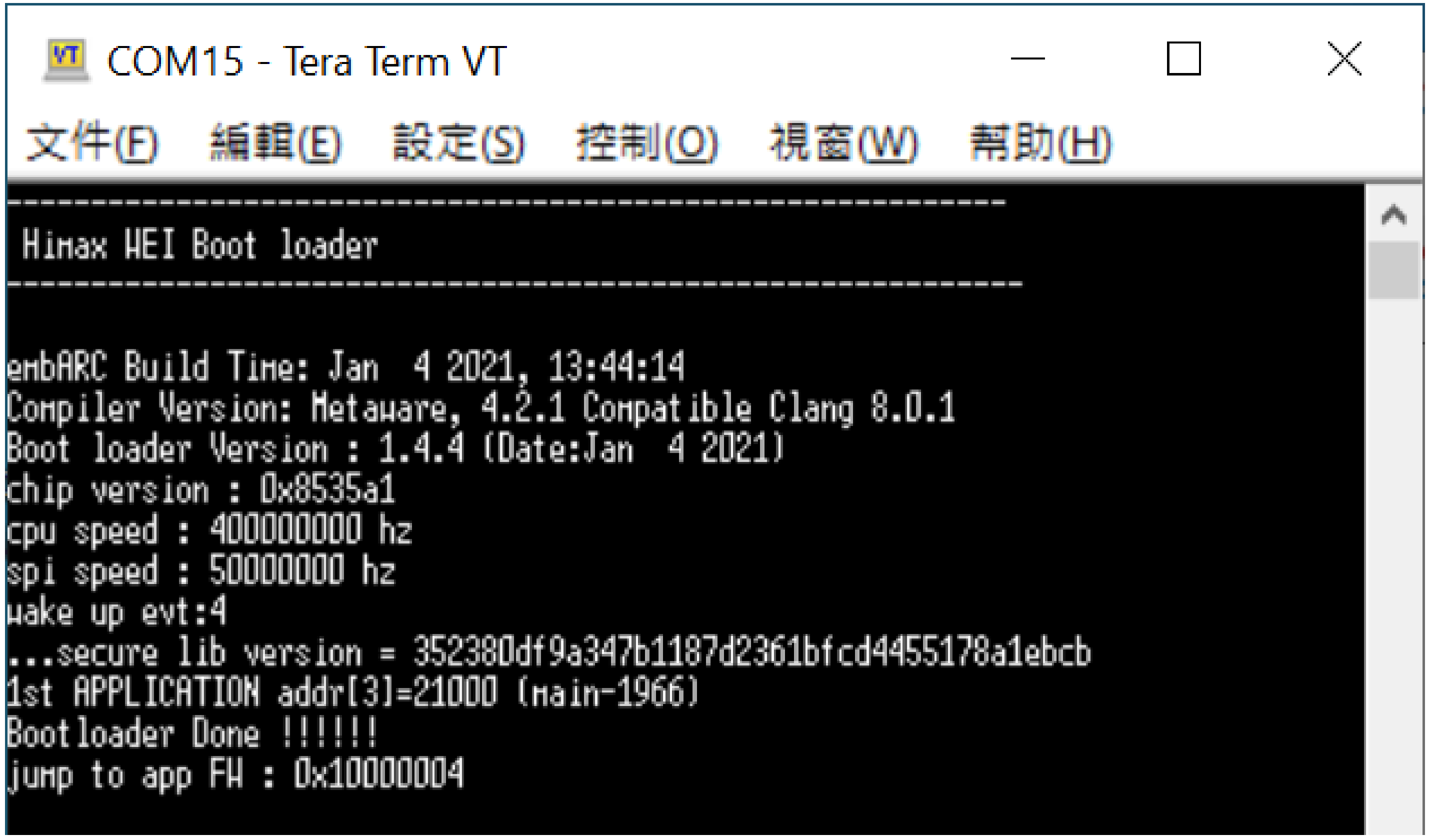
傳輸延遲

0 毫秒/字符 0 毫秒/行

```
Device Friendly Name: USB Serial Port (COM15)
Device Instance ID: FTDIBUS\VID_0403+PID_6015+DT045FSBA\00
Device Manufacturer: FTDI
Provider Name: FTDI
Driver Date: 8-16-2017
Driver Version: 2.12.28.0
```

# Lab1: UART

- Push WE-I reset button, you will see Boot message

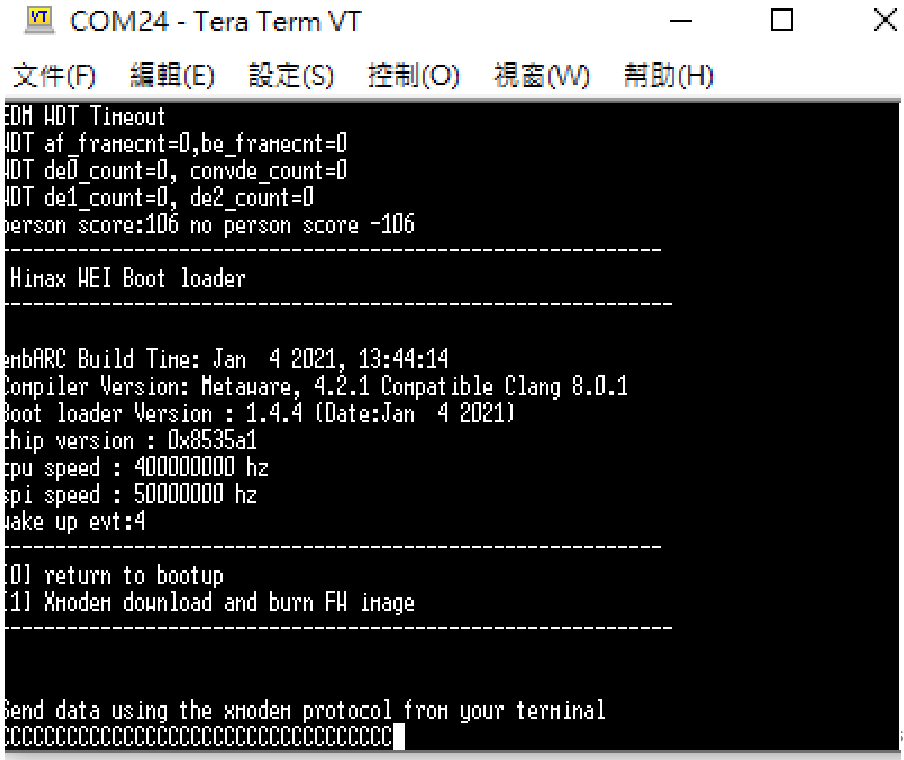


```
-----------------------------------------------------------
 Himax WEI Boot loader
------------------------------------------------------------

embARC Build Time: Jan  4 2021, 13:44:14
Compiler Version: Metaware, 4.2.1 Compatible Clang 8.0.1
Boot loader Version : 1.4.4 (Date:Jan  4 2021)
chip version : 0x8535a1
cpu speed : 400000000 hz
spi speed : 50000000 hz
wake up evt:4
...secure lib version = 352380df9a347b1187d2361bfcd4455178a1ebcb
1st APPLICATION addr[3]=21000 (main-1966)
Bootloader Done !!!!!!
jump to app FW : 0x10000004
```

# Lab1: UART

## Update Application

1. Finish to connect WE-I with Tera Term
2. Click on any display area
3. Keep to press key “1” on the keyboard, and reset WE-I
4. WE-I will start to receive img file by Xmodem

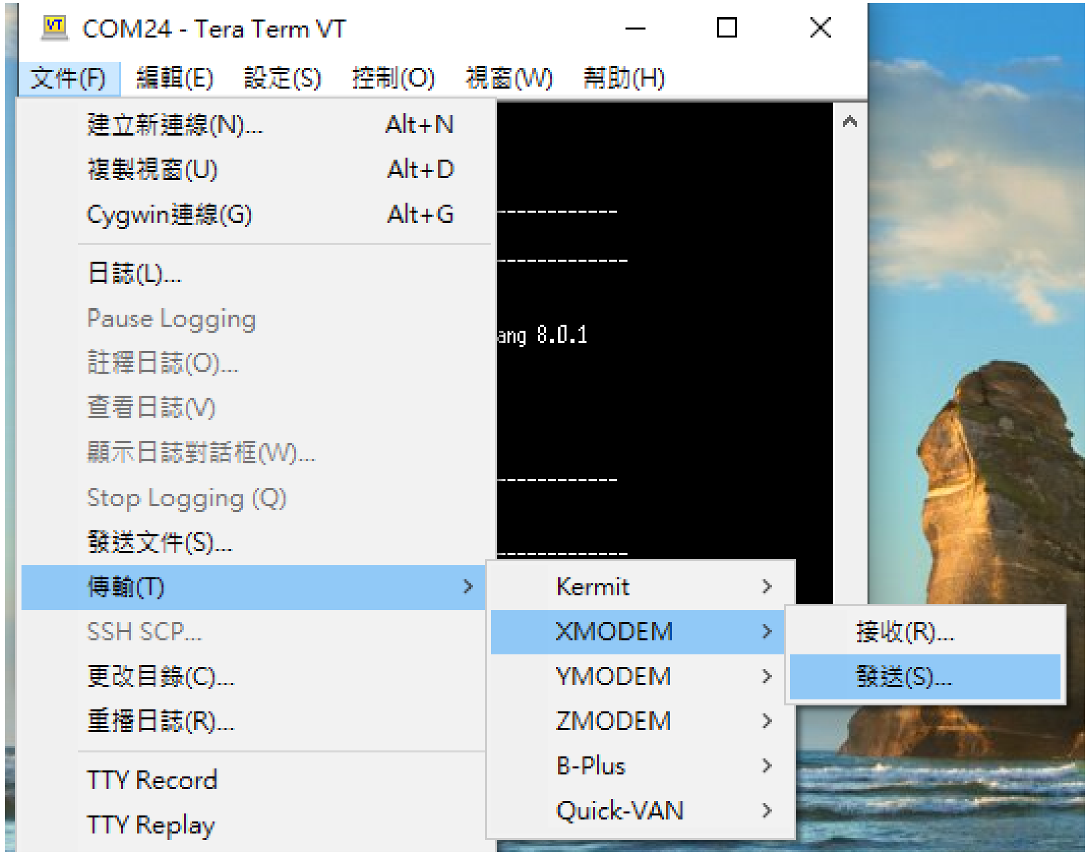


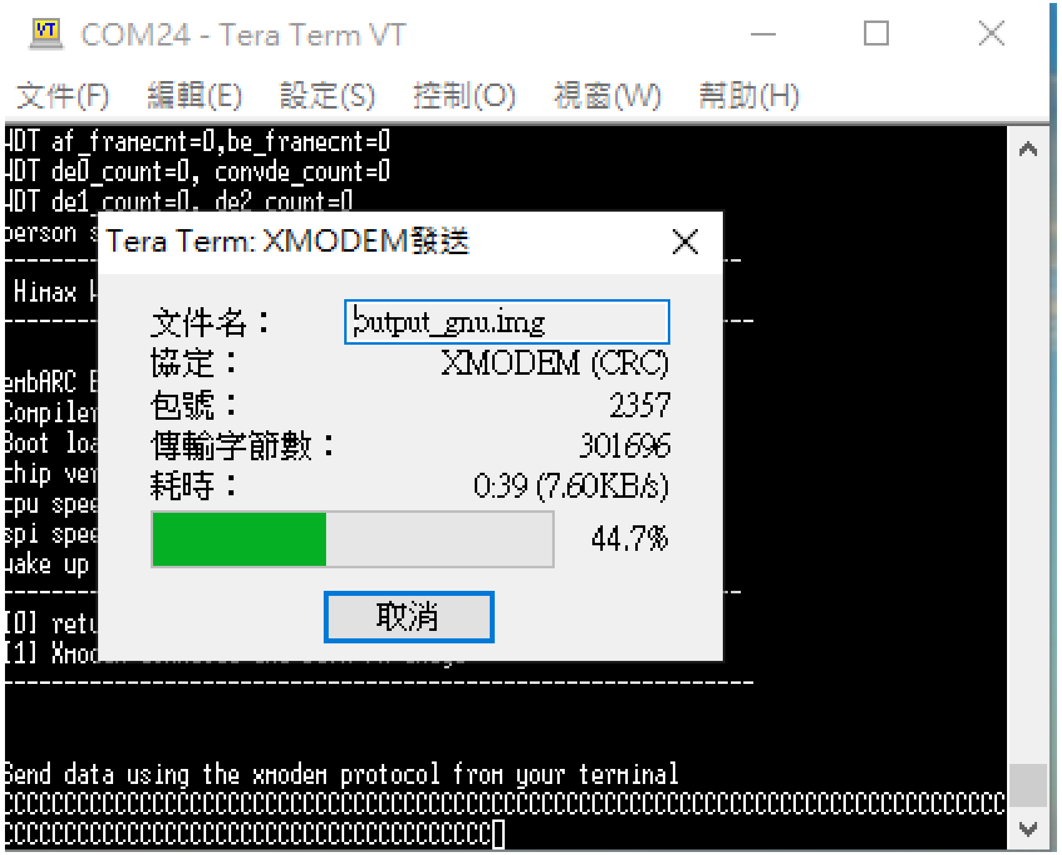
```
EOM WDT Timeout
WDT af_framecnt=0,be_framecnt=0
WDT de0_count=0, convde_count=0
WDT de1_count=0, de2_count=0
person score:106 no person score -106
-------------------------------------------------------------
 Himax WEI Boot loader
--------------------------------------------------------------


embARC Build Time: Jan  4 2021, 13:44:14
Compiler Version: Metaware, 4.2.1 Compatible Clang 8.0.1
Boot loader Version : 1.4.4 (Date:Jan  4 2021)
chip version : 0x8535a1
cpu speed : 400000000 hz
spi speed : 50000000 hz
wake up evt:4
-------------------------------------------------------------
[0] return to bootup
[1] Xmodem download and burn FW image
--------------------------------------------------------------


Send data using the xmodem protocol from your terminal
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
```

# Lab1: UART

## Update Application

5. Tera term File > Transmit > XMODEM > Transmit > select img file
6. Wait for Transmit





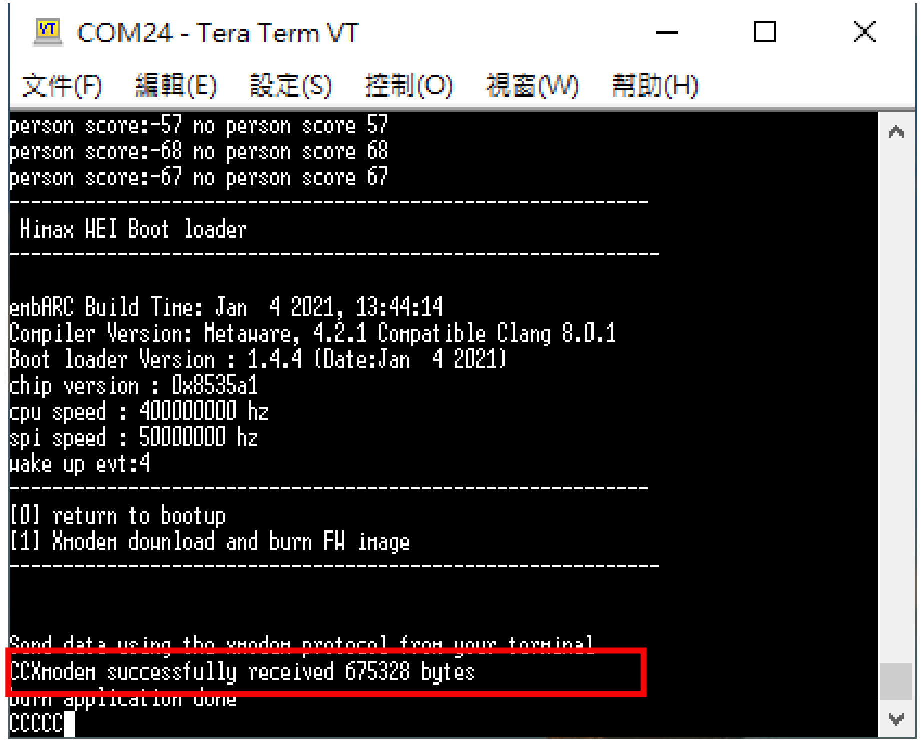
文件名： output_gnu.img
協定： XMODEM (CRC)
包號： 2357
傳輸字節數： 301696
耗時： 0:39 (7.60KB/s)
44.7%

```
HDT af_framecnt=0,be_framecnt=0
HDT de0_count=0, convde_count=0
HDT de1_count=0, de2_count=0
person s
--------
 Himax H
--------

embARC B
Compiler
Boot loa
chip ver
cpu spee
spi spee
wake up
--------
[0] retu
[1] Xmod
----------------------------------------------------------------


Send data using the xmodem protocol from your terminal
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
```

# Lab1: UART

## Update Application

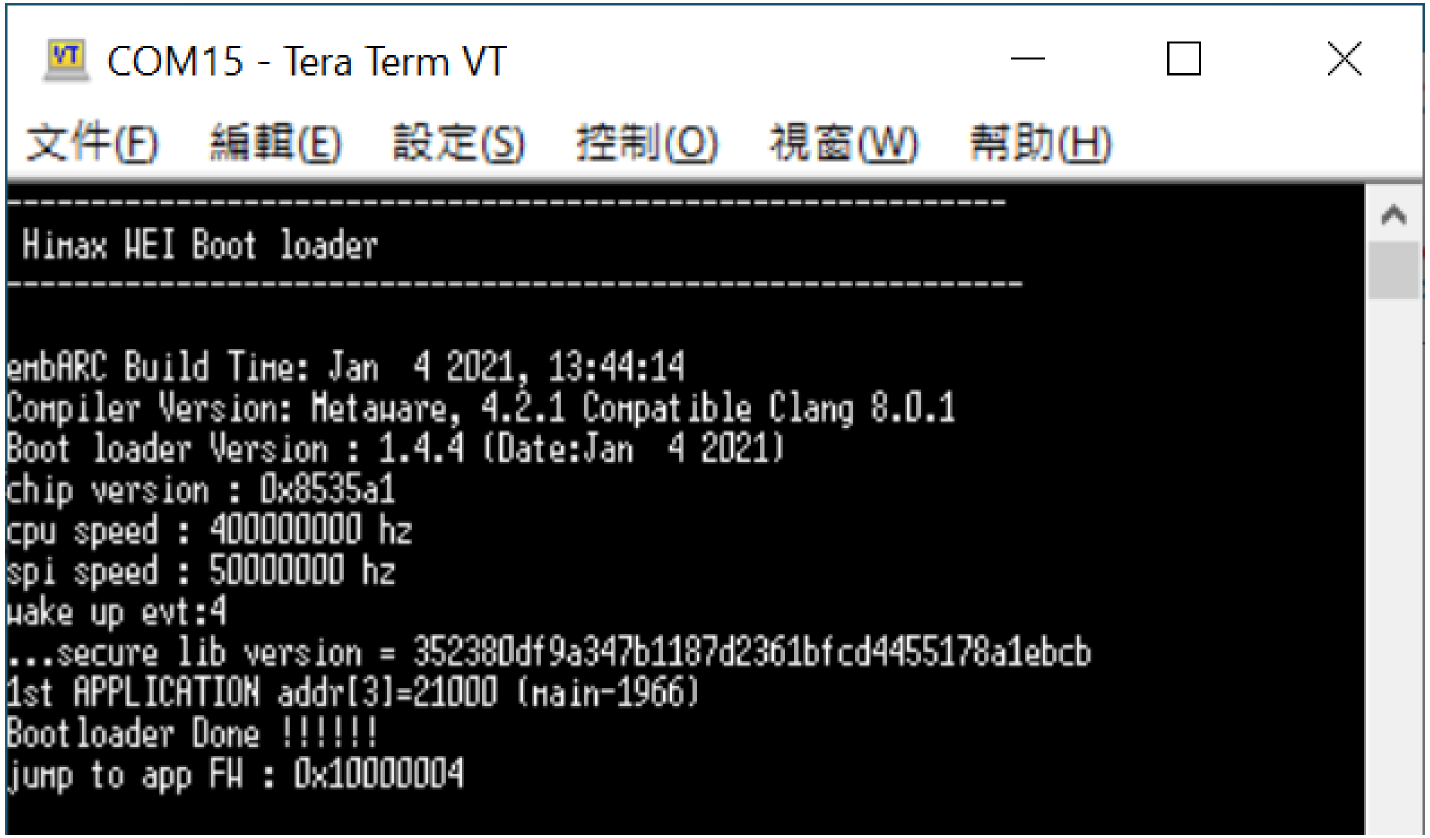
7. Terminal will show “Xmodem successfully received xxx bytes” after transmission
8. Press reset button to run your application



```
person score:-57 no person score 57
person score:-68 no person score 68
person score:-67 no person score 67
------------------------------------------------------------
 Himax WEI Boot loader
-------------------------------------------------------------


embARC Build Time: Jan  4 2021, 13:44:14
Compiler Version: Metaware, 4.2.1 Compatible Clang 8.0.1
Boot loader Version : 1.4.4 (Date:Jan  4 2021)
chip version : 0x8535a1
cpu speed : 400000000 hz
spi speed : 50000000 hz
wake up evt:4
------------------------------------------------------------
[0] return to bootup
[1] Xmodem download and burn FW image
-------------------------------------------------------------



Send data using the xmodem protocol from your terminal
CCXmodem successfully received 675328 bytes
burn application done
CCCCC
```

# Lab1: UART

- Push WE-I reset button, you will see Boot message

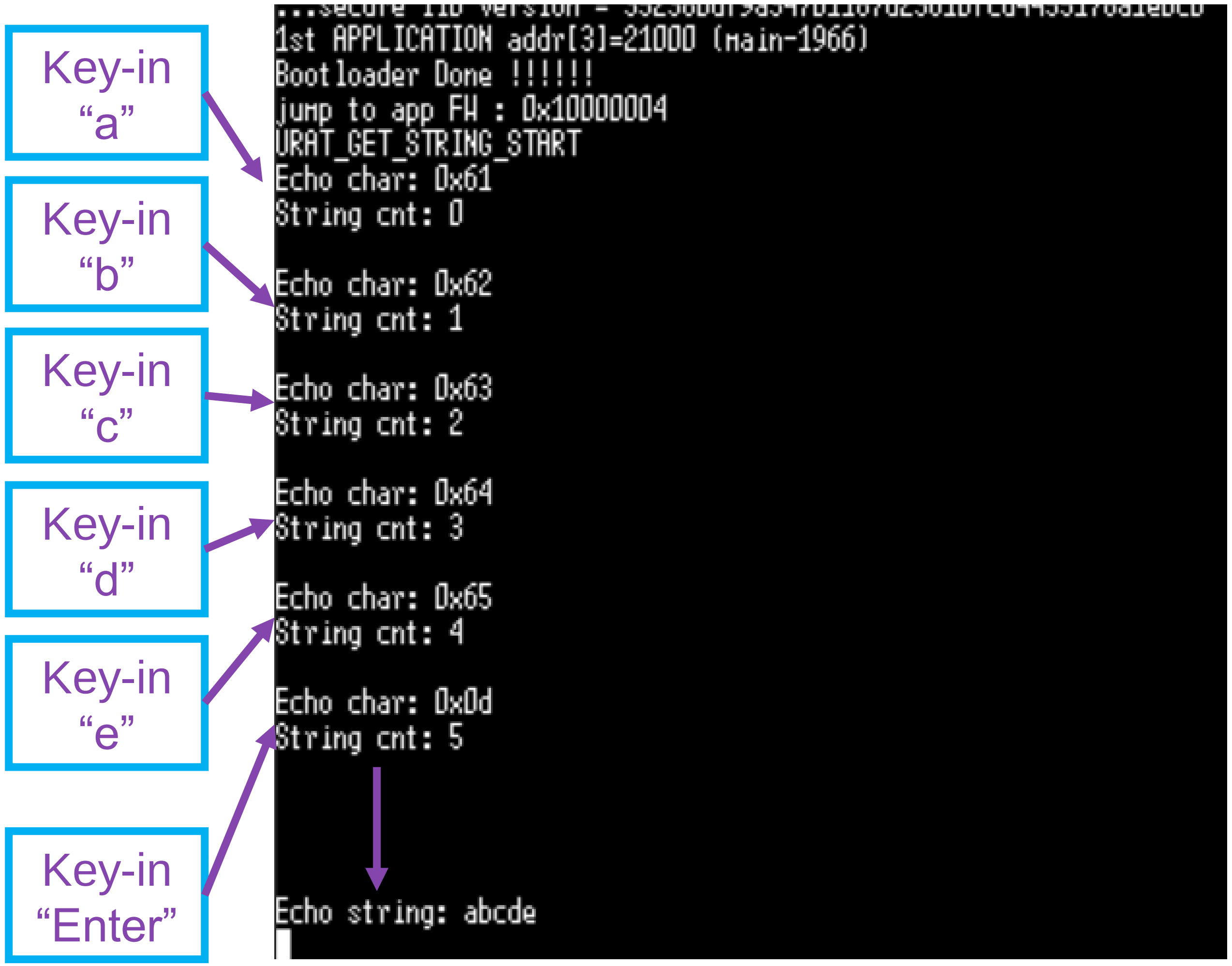


```
-------------------------------------------------------------
 Himax WEI Boot loader
-------------------------------------------------------------

embARC Build Time: Jan  4 2021, 13:44:14
Compiler Version: Metaware, 4.2.1 Compatible Clang 8.0.1
Boot loader Version : 1.4.4 (Date:Jan  4 2021)
chip version : 0x8535a1
cpu speed : 400000000 hz
spi speed : 50000000 hz
wake up evt:4
...secure lib version = 352380df9a347b1187d2361bfcd4455178a1ebcb
1st APPLICATION addr[3]=21000 (main-1966)
Bootloader Done !!!!!!
jump to app FW : 0x10000004
```

# Lab1: UART

- This example project will echo every word you key-in
- After you key-in “Enter”

  It will return the string you key-in



```
...secure lib version = [illegible]
1st APPLICATION addr[3]=21000 (main-1966)
Bootloader Done !!!!!!
jump to app FW : 0x10000004
URAT_GET_STRING_START
Echo char: 0x61
String cnt: 0

Echo char: 0x62
String cnt: 1

Echo char: 0x63
String cnt: 2

Echo char: 0x64
String cnt: 3

Echo char: 0x65
String cnt: 4

Echo char: 0x0d
String cnt: 5




Echo string: abcde
```

# Hands-on (Lab 1): GPIO

# Lab1: GPIO

- GPIO initial

```
extern HX_DRV_ERROR_E hx_drv_gpio_initial(hx_drv_gpio_config_t *pgpio_config);

// GPIO initial API, should be called first before you use GPIO

// (hx_drv_gpio_config_t *pgpio_config) options are bellow
   pgpio_config.gpio_pin = HX_DRV_PGPIO_0     /**< Select GPIO number 0  */
   pgpio_config.gpio_pin = HX_DRV_PGPIO_1     /**< Select GPIO number 1  */
   pgpio_config.gpio_pin = HX_DRV_PGPIO_2     /**< Select GPIO number 2  */
   pgpio_config.gpio_pin = HX_DRV_LED_RED     /**< Select GPIO LED RED   */
   pgpio_config.gpio_pin = HX_DRV_LED_GREEN   /**< Select GPIO LED GREEN */

   pgpio_config.gpio_direction = HX_DRV_GPIO_INPUT  /**< Select GPIO as input */
   pgpio_config.gpio_direction = HX_DRV_GPIO_OUTPUT /**< Select GPIO as output */

   pgpio_config.gpio_data = 0    /**< GPIO output LOW  (only in output mode)*/
   pgpio_config.gpio_data = 1    /**< GPIO output HIGH (only in output mode)*/
```

# Lab1: GPIO

- GPIO initial

```
Ex: hx_drv_gpio_config_t hal_gpio;

    hal_gpio.gpio_pin = HX_DRV_LED_GREEN;
    hal_gpio.gpio_direction = HX_DRV_GPIO_OUTPUT;
    hal_gpio.gpio_data = 1;

    hx_drv_gpio_initial(&hal_gpio);
```

# Lab1: GPIO

- GPIO set output stage

```
extern HX_DRV_ERROR_E hx_drv_gpio_set(hx_drv_gpio_config_t *pgpio_config);

Ex: hal_gpio.gpio_data = 1;
    hx_drv_gpio_set(&hal_gpio);
    /*delay 100ms ....*/

    hal_gpio.gpio_data = 0;
    hx_drv_gpio_set(&hal_gpio);
    /*delay 100ms ....*/
```

# Lab1: GPIO

- GPIO get input stage

```
extern HX_DRV_ERROR_E hx_drv_gpio_get(hx_drv_gpio_config_t *pgpio_config);

Ex: hx_drv_gpio_get(&hal_gpio);
    /* you can print hal_gpio.gpio_data */
```

# Lab1: GPIO

## Conclusion

- GPIO should initialize to set input or output mode.

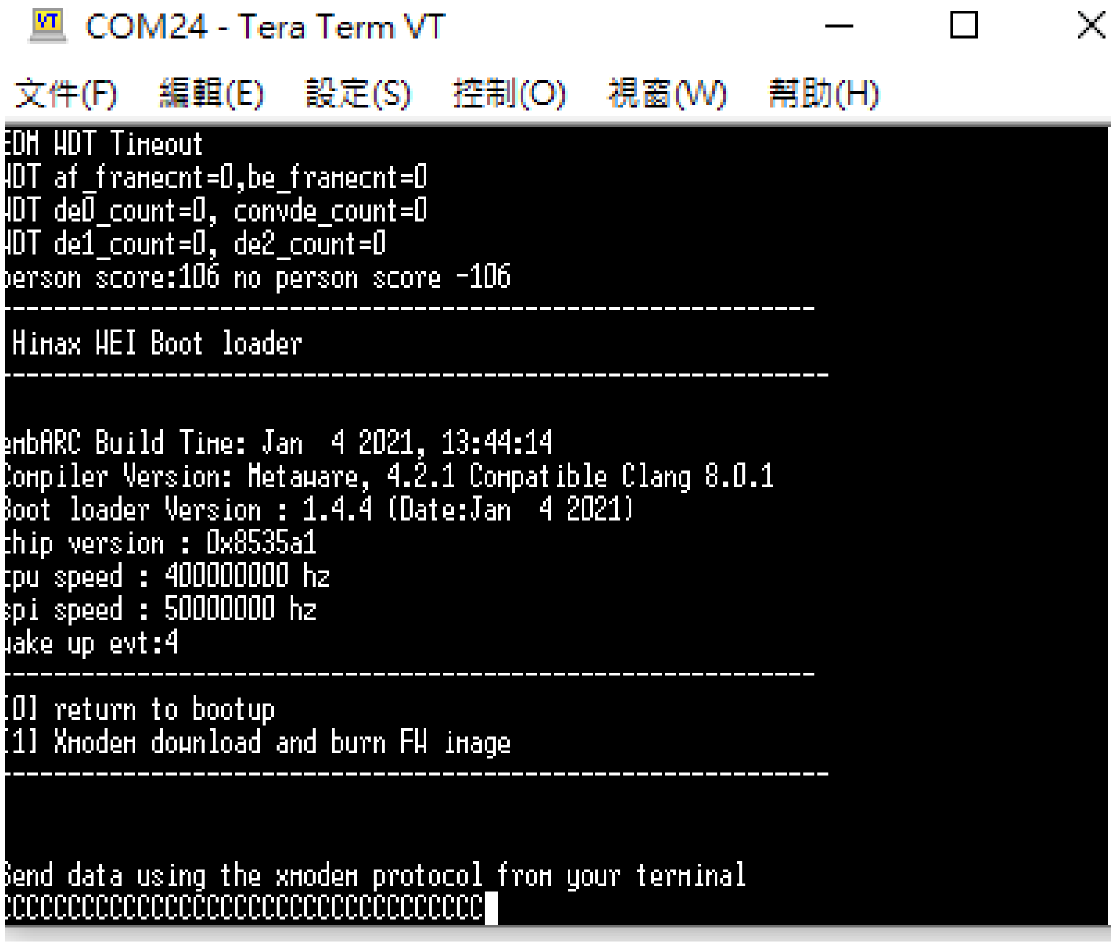
```
HX_DRV_ERROR_E hx_drv_gpio_initial(hx_drv_gpio_config_t *pgpio_config);
```

- After you initialize GPIO, you can get or set GPIO stage.

```
HX_DRV_ERROR_E hx_drv_gpio_set(hx_drv_gpio_config_t *pgpio_config);

HX_DRV_ERROR_E hx_drv_gpio_get(hx_drv_gpio_config_t *pgpio_config);
```

# Lab1: GPIO

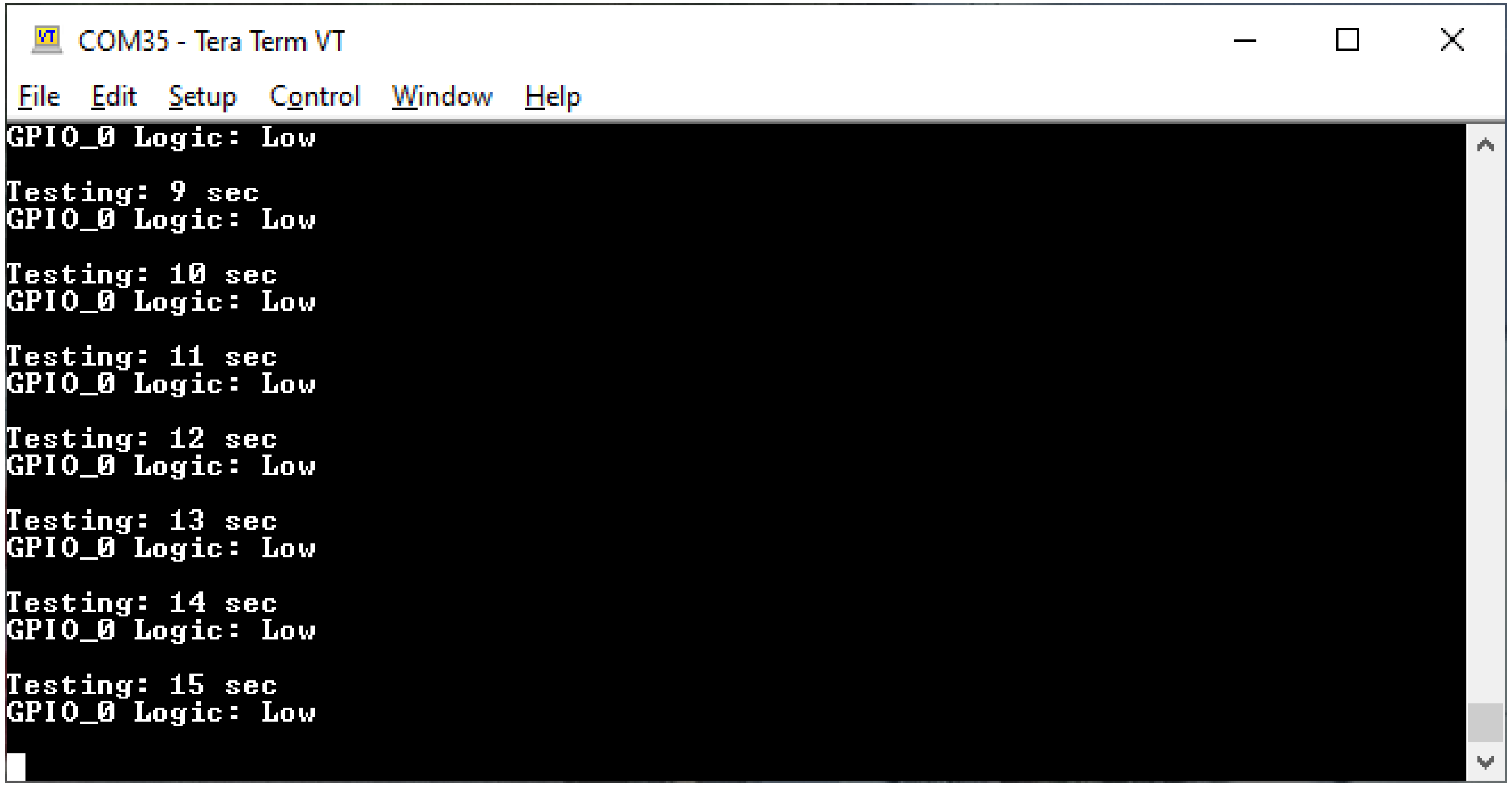
- Push WE-I reset button, and download image file to WE-I



```
EDM WDT Timeout
WDT af_framecnt=0,be_framecnt=0
WDT de0_count=0, convde_count=0
WDT de1_count=0, de2_count=0
person score:106 no person score -106
-----------------------------------------------------------
 Himax WEI Boot loader
------------------------------------------------------------


embARC Build Time: Jan  4 2021, 13:44:14
Compiler Version: Metaware, 4.2.1 Compatible Clang 8.0.1
Boot loader Version : 1.4.4 (Date:Jan  4 2021)
chip version : 0x8535a1
cpu speed : 400000000 hz
spi speed : 50000000 hz
wake up evt:4
-----------------------------------------------------------
[0] return to bootup
[1] Xmodem download and burn FW image
------------------------------------------------------------



Send data using the xmodem protocol from your terminal
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
```

# Lab1: GPIO

- This example project will print GPIO_0 input stage
- Red and green LED will toggle.



```
GPIO_0 Logic: Low

Testing: 9 sec
GPIO_0 Logic: Low

Testing: 10 sec
GPIO_0 Logic: Low

Testing: 11 sec
GPIO_0 Logic: Low

Testing: 12 sec
GPIO_0 Logic: Low

Testing: 13 sec
GPIO_0 Logic: Low

Testing: 14 sec
GPIO_0 Logic: Low

Testing: 15 sec
GPIO_0 Logic: Low
```

# Hands-on (Lab 1): I2C

# Lab1: I2C

- I2C master initial

```
extern HX_DRV_ERROR_E hx_drv_share_switch(HX_DRV_SHARE_MODE_E mode);
// I2C initial API, should be called first before you use I2C
// I2C and SPI use the same output pin, this API will switch output pin function
```

Ex: hx_drv_share_switch(SHARE_MODE_I2CM);

# Lab1: I2C

- I2C master send package

```
extern HX_DRV_ERROR_E hx_drv_i2cm_set_data(uint8_t slave_addr_sft, uint8_t *addr,
uint32_t addr_len, uint8_t *data, uint32_t data_len);
// I2C master send package API, variable descriptions are bellow:
   slave_addr_sft: i2c 7-bit slave address    /**< Align right */
   *addr: Get package pointer       /*Not use in sending mode*/
    addr_len: Get package length /*Not use in sending mode*/
   *data: Send package pointer
    data_len: Send package length
```

# Lab1: I2C

- I2C master send package

Ex:
```
uint8_t data_write[2];
uint8_t data_read[2];
data_write[0] = SSD1306_DATA_CONTINUE;
data_write[1] = cmd;

hx_drv_i2cm_set_data(SSD1306_ADDRESS, data_read, 0, data_write, 2);
```

# Lab1: I2C

- I2C master get package

```
extern HX_DRV_ERROR_E hx_drv_i2cm_get_data(uint8_t slave_addr_sft, uint8_t *addr,
uint32_t addr_len, uint8_t *data, uint32_t data_len);
// I2C master get package API, variable descriptions are bellow:
   slave_addr_sft: i2c 7-bit slave address    /**< Align right */
   *addr: Get package pointer
    addr_len: Gen package length
   *data: Send package pointer
    data_len: Send package length
```

# Lab1: I2C

## Conclusion

- I2C should initialize before use it.

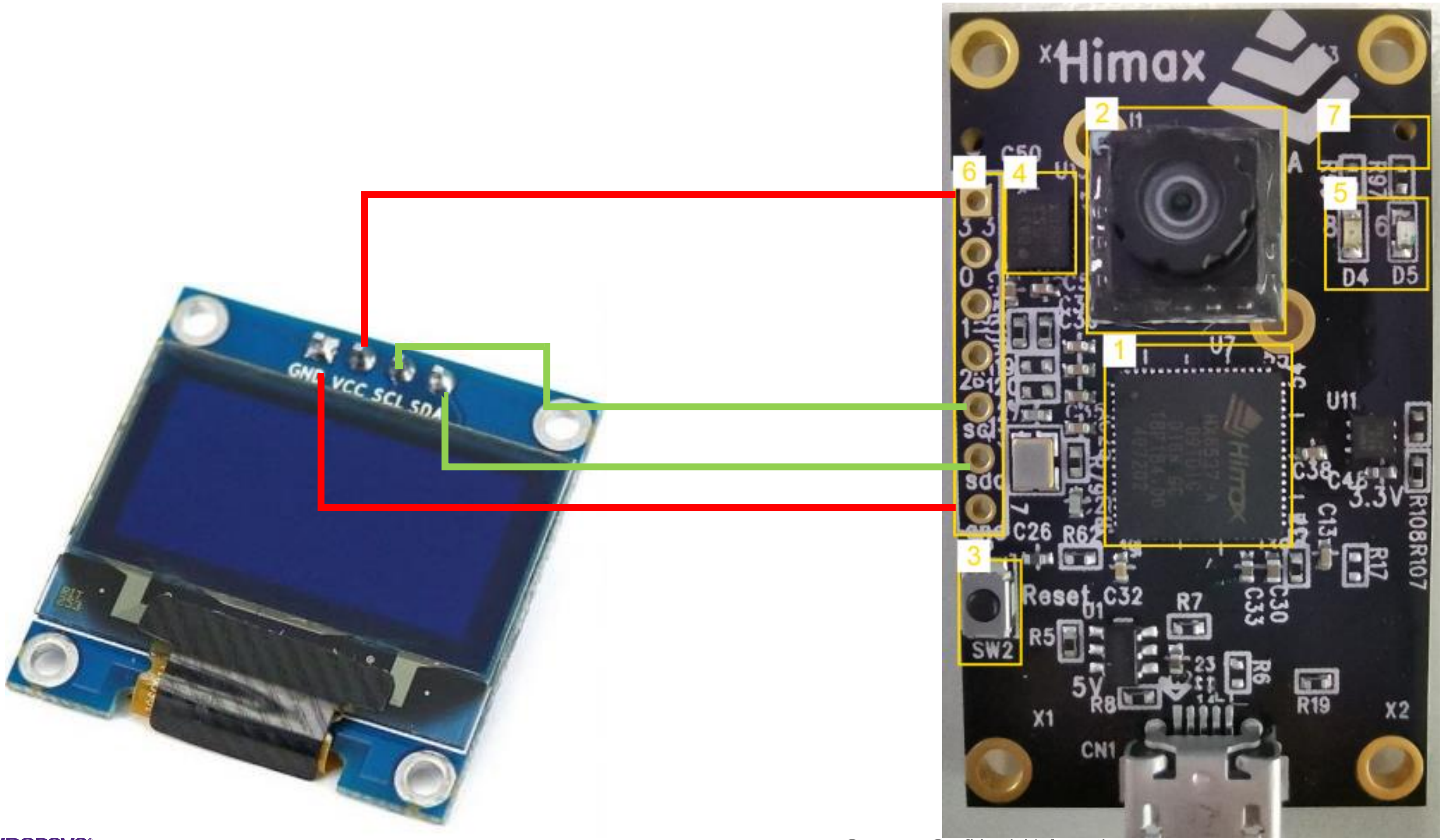
```
HX_DRV_ERROR_E hx_drv_share_switch(HX_DRV_SHARE_MODE_E mode);
```

- After you initialize I2C, you can get or send I2C package3

```
HX_DRV_ERROR_E hx_drv_i2cm_set_data(uint8_t slave_addr_sft, uint8_t *addr,
uint32_t addr_len, uint8_t *data, uint32_t data_len);
HX_DRV_ERROR_E hx_drv_i2cm_get_data(uint8_t slave_addr_sft, uint8_t *addr,
uint32_t addr_len, uint8_t *data, uint32_t data_len);
```

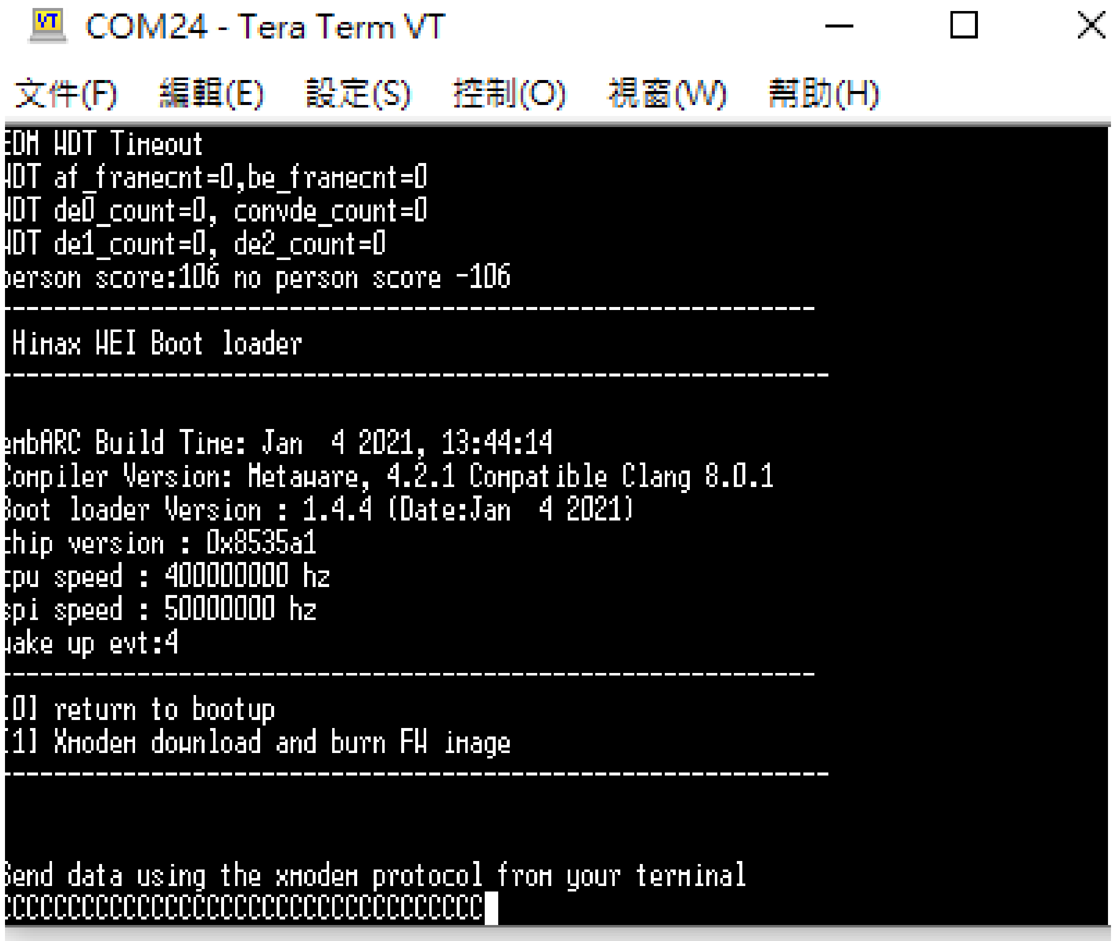
# Lab1: I2C

- Connect OLED1306 and WE-I by 2.54 header



# Lab1: I2C

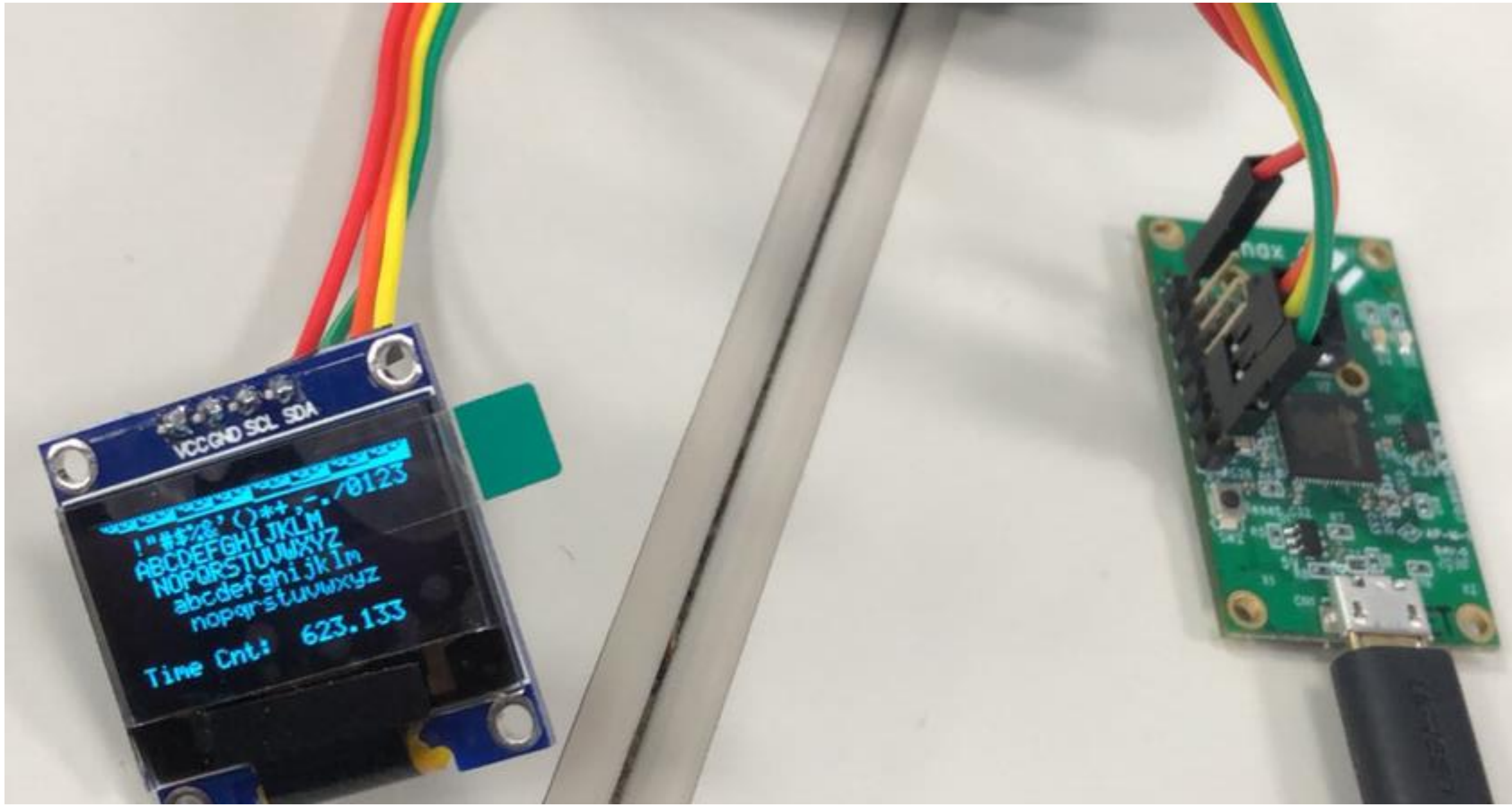
- Push WE-I reset button, and download image file to WE-I



```
EDM WDT Timeout
WDT af_framecnt=0,be_framecnt=0
WDT de0_count=0, convde_count=0
WDT de1_count=0, de2_count=0
person score:106 no person score -106
-----------------------------------------------------------
 Himax WEI Boot loader
------------------------------------------------------------


embARC Build Time: Jan  4 2021, 13:44:14
Compiler Version: Metaware, 4.2.1 Compatible Clang 8.0.1
Boot loader Version : 1.4.4 (Date:Jan  4 2021)
chip version : 0x8535a1
cpu speed : 400000000 hz
spi speed : 50000000 hz
wake up evt:4
-----------------------------------------------------------
[0] return to bootup
[1] Xmodem download and burn FW image
------------------------------------------------------------




Send data using the xmodem protocol from your terminal
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
```

# Lab1: I2C

- This example project will print string on OLED1306.



# Hands-on (Lab 2): Accelerometer

# Lab2: Accelerometer

- Accelerometer initial

```
extern HX_DRV_ERROR_E hx_drv_accelerometer_initial();
// 3-axis accelerometer initialization, it start to retrieve data after initial
// It will initial accelerometer with sampling rate 119 Hz, bandwidth 50 Hz,
// scale selection 4g at continuous mode.

Ex: hx_drv_accelerometer_initial();
```

# Lab2: Accelerometer

- Accelerometer FIFO count get

```
extern HX_DRV_ERROR_E hx_drv_accelerometer_available_count();
// Check how many data in the accelerometer FIFO.
// Each count represent 1 set of x-axis,y-axis,z-axis data.

Ex: available_count = hx_drv_accelerometer_available_count();
```

# Lab2: Accelerometer

- Get 1 package from Accelerometer FIFO

```
extern HX_DRV_ERROR_E hx_drv_accelerometer_receive(float *x, float *y, float *z);

// Receive data from 3-axis accelerometer.


Ex: float x, y, z;
    hx_drv_accelerometer_receive(&x, &y, &z);
```

# Lab2: Accelerometer

## Conclusion

- Accelerometer should initialize before get accelerometer data.

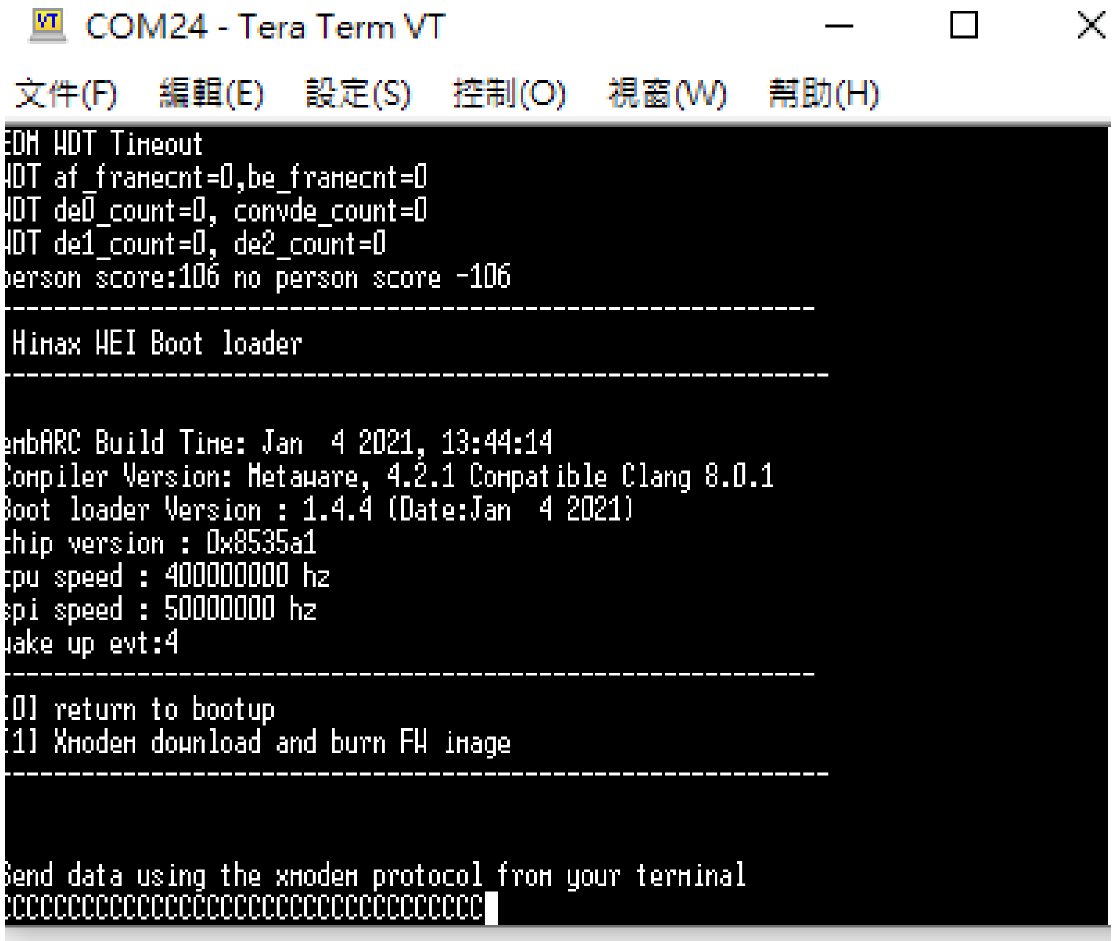
```
HX_DRV_ERROR_E hx_drv_accelerometer_initial();
```

- After you initialize accelerometer, you can get data from FIFO.

```
HX_DRV_ERROR_E hx_drv_accelerometer_available_count(float *x, float *y, float *z);
HX_DRV_ERROR_E hx_drv_accelerometer_receive(float *x, float *y, float *z);
```

# Lab2: Accelerometer

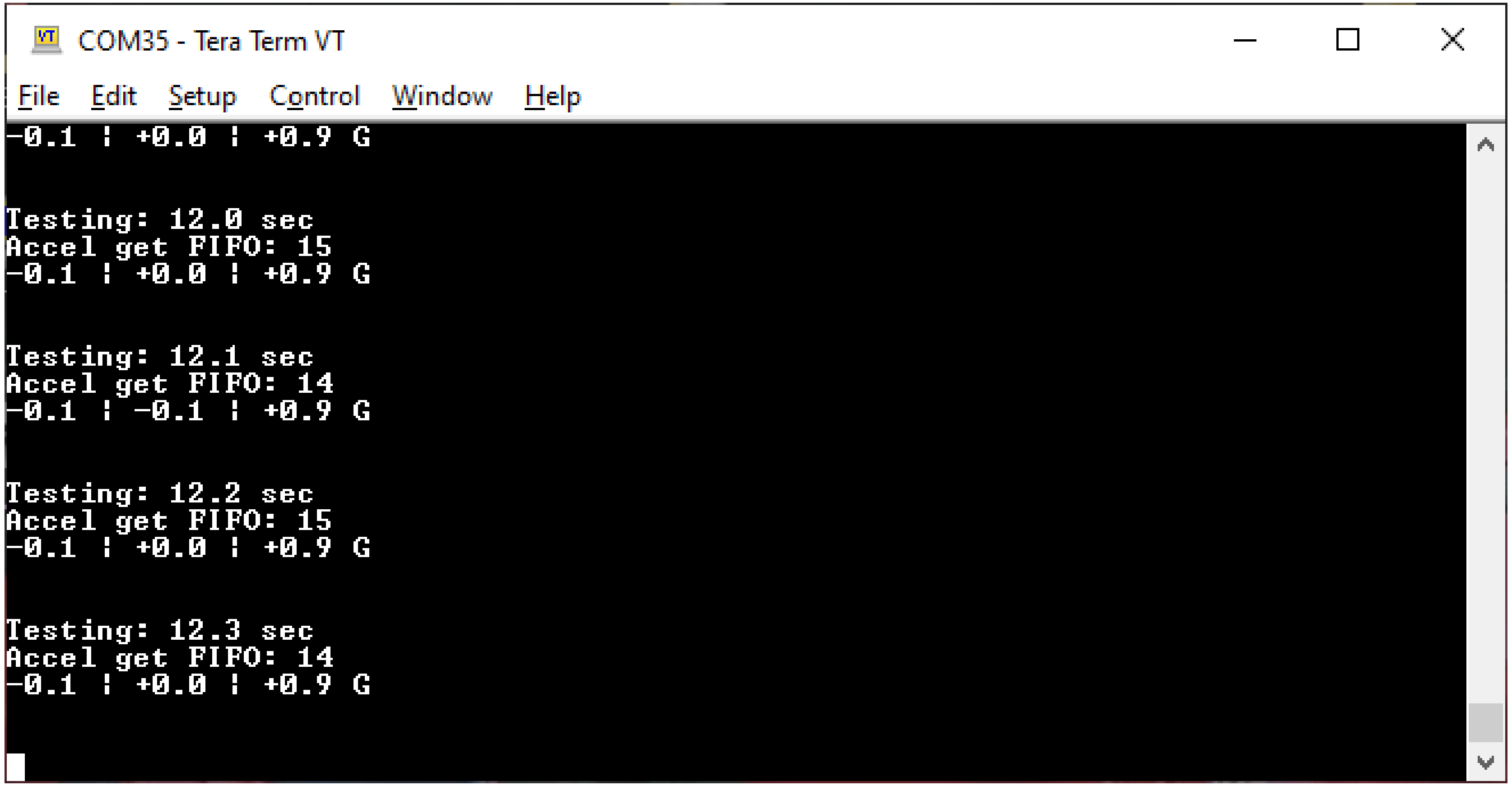
- Push WE-I reset button, and download image file to WE-I



```
EDM WDT Timeout
WDT af_framecnt=0,be_framecnt=0
WDT de0_count=0, convde_count=0
WDT de1_count=0, de2_count=0
person score:106 no person score -106
-----------------------------------------------------------
 Himax WEI Boot loader
------------------------------------------------------------


embARC Build Time: Jan  4 2021, 13:44:14
Compiler Version: Metaware, 4.2.1 Compatible Clang 8.0.1
Boot loader Version : 1.4.4 (Date:Jan  4 2021)
chip version : 0x8535a1
cpu speed : 400000000 hz
spi speed : 50000000 hz
wake up evt:4
-----------------------------------------------------------
[0] return to bootup
[1] Xmodem download and burn FW image
------------------------------------------------------------



Send data using the xmodem protocol from your terminal
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
```

# Lab2: Accelerometer

- This example project will print accelerometer data on terminal



```
-0.1 ¦ +0.0 ¦ +0.9 G


Testing: 12.0 sec
Accel get FIFO: 15
-0.1 ¦ +0.0 ¦ +0.9 G


Testing: 12.1 sec
Accel get FIFO: 14
-0.1 ¦ -0.1 ¦ +0.9 G


Testing: 12.2 sec
Accel get FIFO: 15
-0.1 ¦ +0.0 ¦ +0.9 G


Testing: 12.3 sec
Accel get FIFO: 14
-0.1 ¦ +0.0 ¦ +0.9 G
```

# Hands-on (Lab 2): Microphone

# Lab2: Microphone

- Microphone initial

```
extern HX_DRV_ERROR_E hx_drv_mic_initial();
// Capture Single channel audio data from Microphone.
//  Each sample for mono PDM is 16bits little-endian signed data.
//  During each millisecond, there will be 16 samples(32 bytes) of
//  audio data storage to target address.
//  Please use API "hx_drv_mic_capture_dual" for normal case.
```

Ex: hx_drv_mic_initial();

# Lab2: Microphone

- Microphone enable

```
extern HX_DRV_ERROR_E hx_drv_mic_on();

// Turn on microphone, it will start to record audio.

//  Please call hx_drv_mic_initial() first to initial microphone.
```

Ex: hx_drv_mic_on();

# Lab2: Microphone

- Microphone disable

```
extern HX_DRV_ERROR_E hx_drv_mic_off();
// Turn off microphone.
```

Ex: hx_drv_mic_off();

# Lab2: Microphone

- Microphone time stamp get

```
extern HX_DRV_ERROR_E hx_drv_mic_timestamp_get(int32_t *time);

// Get current time-stamp from audio buffer in driver

//  For current Himax mic driver, time stamp will be updated every 100ms


Ex: hx_drv_mic_timestamp_get(&time_cur);
    if(time_cur != time_prev) //time stamp is changed
    {
        ………
    }
    time_prev = time_cur;
```

# Lab2: Microphone

- Microphone dual channel data get

```
extern HX_DRV_ERROR_E hx_drv_mic_capture_dual(hx_drv_mic_data_config_t
*pmic_config);
// Received data will be assigned by driver with address and
//  size count in bytes about samples
//  For example, if data_size is 6400,
//  that means 1600 samples(100ms) of audio data in address
//  For current Himax mic driver, time stamp will be updated every 100ms
// (hx_drv_mic_data_config_t *pmic_config) options are bellow
   pmic_config.data_address /**< microphone data array address, assigned by driver
   pmic_config.data_size    /**< microphone data size in bytes, assigned by driver
```

# Lab2: Microphone

- Microphone dual channel data get

```
Ex: #define mic_sample_rate 16000
    #define AUD_BLK_100MS_SZ (mic_sample_rate / 10)  //100ms

    typedef struct
    {
        int16_t left;
        int16_t right;
    } META_AUDIO_t;

    Hx_drv_mic_data_config_t slt_audio_config;
    META_AUDIO_t audio_clip[AUD_BLK_100MS_SZ];
```

# Lab2: Microphone

- Microphone dual channel data get

```
Ex(continue):
   //After you check time stamp is change, then………
   hx_drv_mic_capture_dual(&slt_audio_config);

   //Copy dual channel data to array.
   memcpy((void*) &audio_clip[0], (void*)slt_audio_config.data_address,
slt_audio_config.data_size*sizeof(uint8_t));

   //Microphone data will in audio_clip[i].left and audio_clip[i].right
```

# Lab2: Microphone

## Conclusion

- Microphone should initialize and enable before get microphone data.

```
HX_DRV_ERROR_E hx_drv_mic_initial();
HX_DRV_ERROR_E hx_drv_mic_on();
```

- Before you get microphone data, please check time stamp change or not.

```
HX_DRV_ERROR_E hx_drv_mic_timestamp_get(int32_t *time);
```

- After time stamp is changed, get dual channel data. (every 100ms)

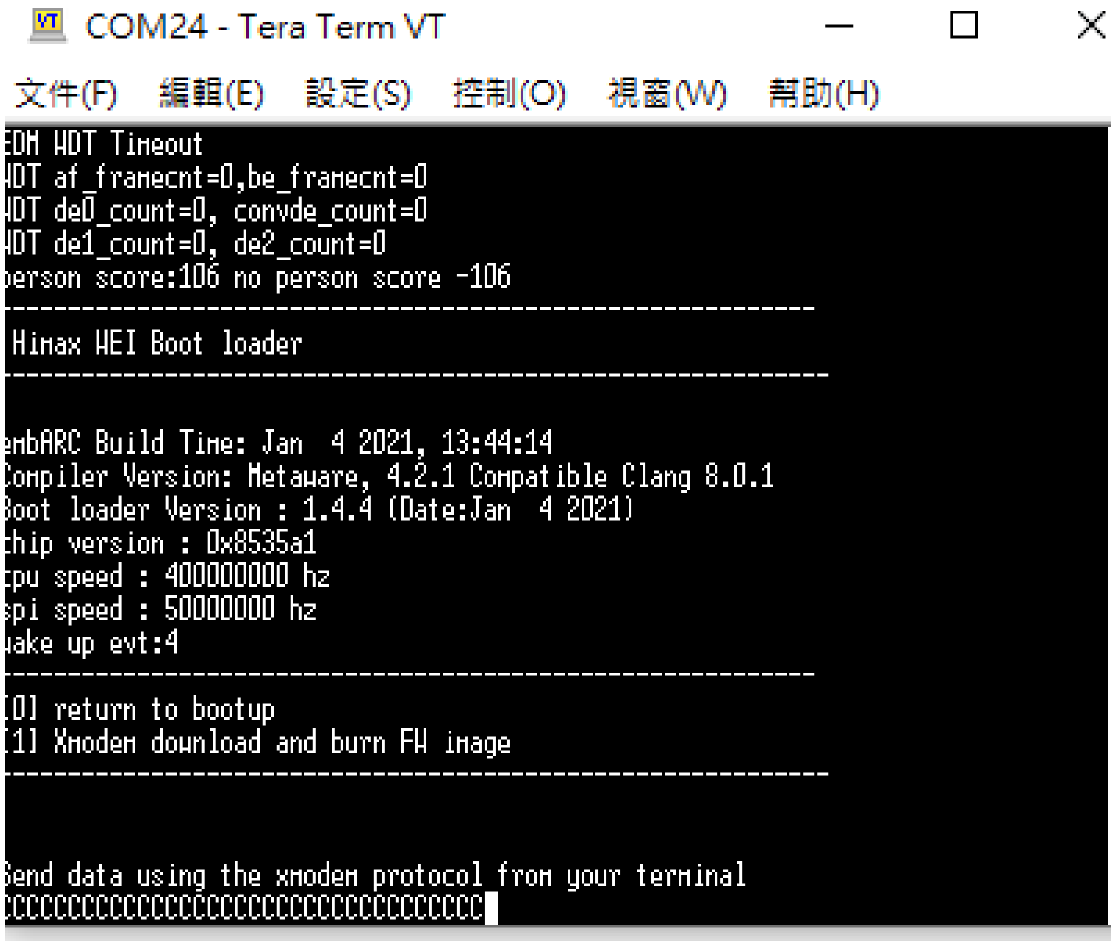
```
HX_DRV_ERROR_E hx_drv_mic_capture_dual(hx_drv_mic_data_config_t *pmic_config);
```

- Use memcpy to copy dual channel data to array.

```
void * memcpy ( void * destination, const void * source, size_t num );
```

# Lab2: Microphone

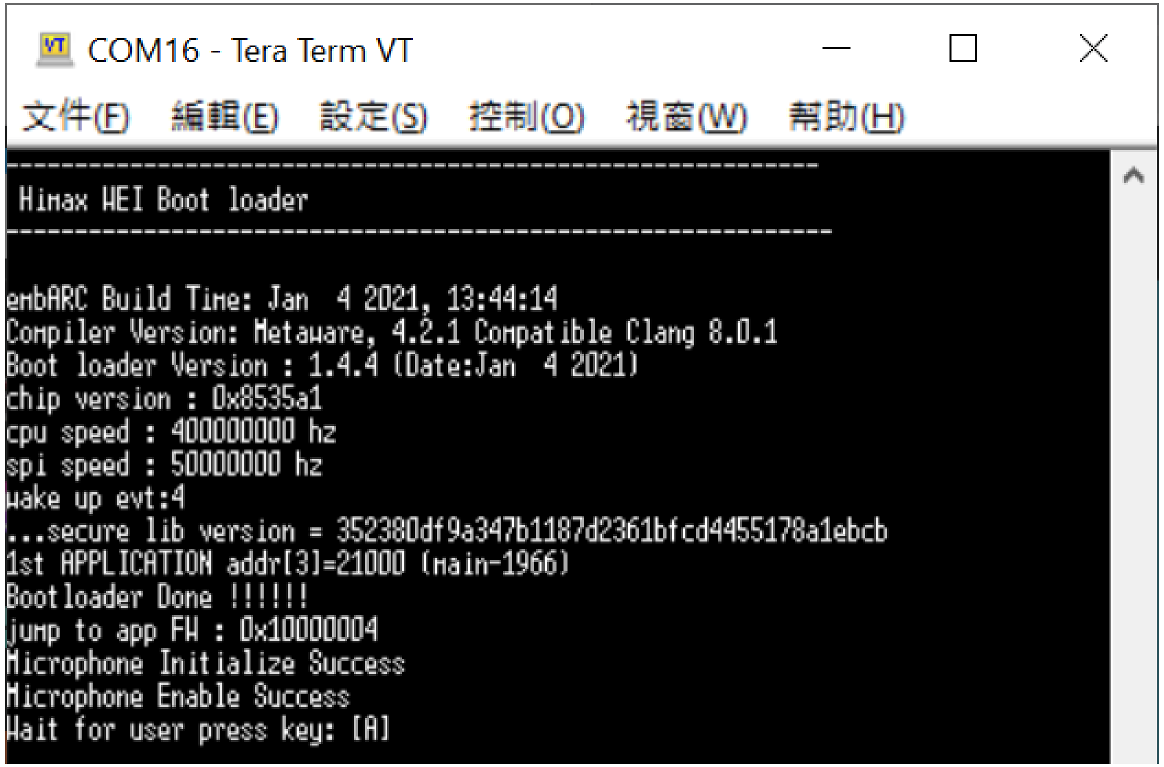
- Push WE-I reset button, and download image file to WE-I



```
EDM WDT Timeout
WDT af_framecnt=0,be_framecnt=0
WDT de0_count=0, convde_count=0
WDT de1_count=0, de2_count=0
person score:106 no person score -106
-----------------------------------------------------------
 Himax WEI Boot loader
------------------------------------------------------------

embARC Build Time: Jan  4 2021, 13:44:14
Compiler Version: Metaware, 4.2.1 Compatible Clang 8.0.1
Boot loader Version : 1.4.4 (Date:Jan  4 2021)
chip version : 0x8535a1
cpu speed : 400000000 hz
spi speed : 50000000 hz
wake up evt:4
-----------------------------------------------------------
[0] return to bootup
[1] Xmodem download and burn FW image
------------------------------------------------------------


Send data using the xmodem protocol from your terminal
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
```

# Lab2: Microphone

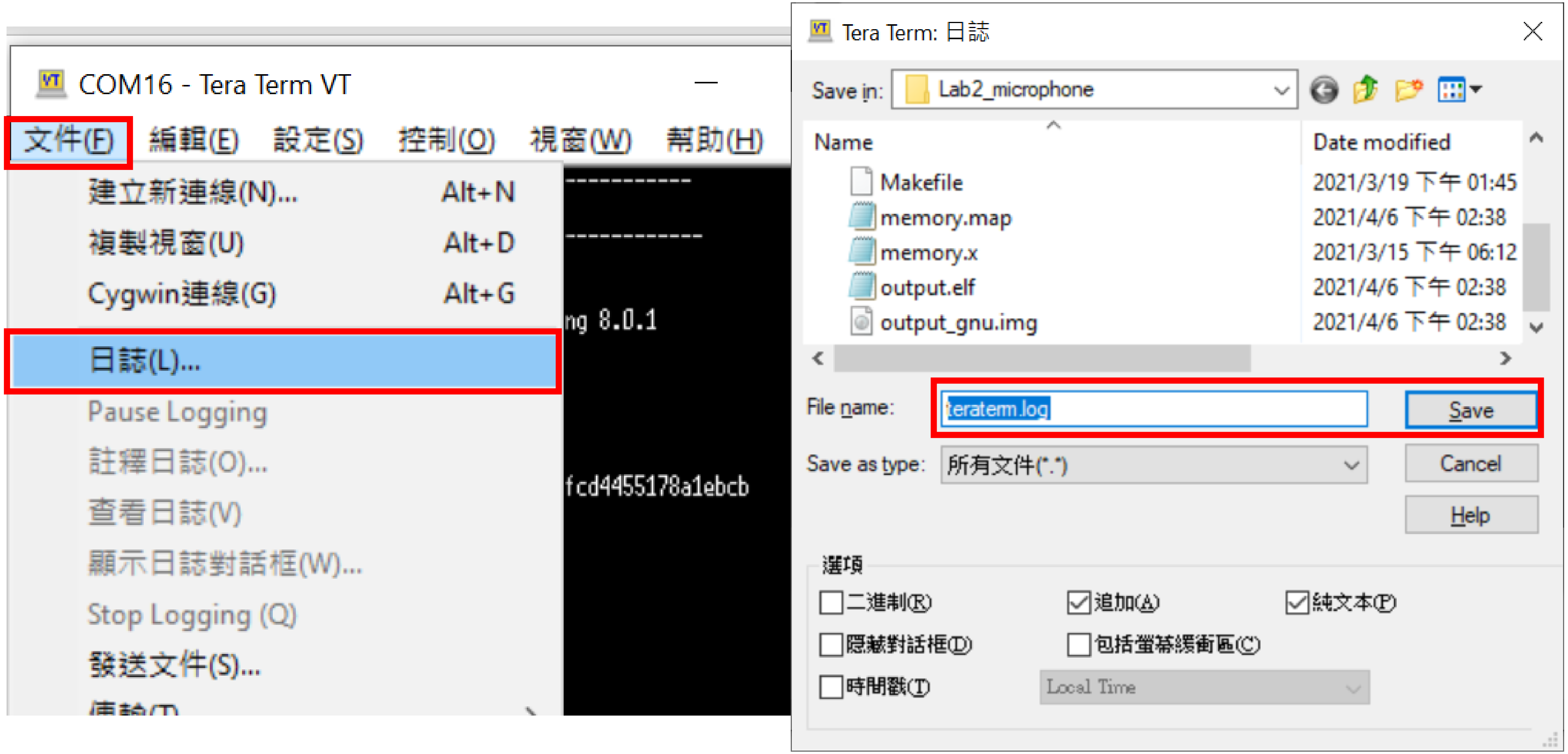
- This example project will wait user key-in “A”, and then recode and send 14 seconds dual channel audio data
- You can save dual channel audio data by terminal log function



```
------------------------------------------------------------
 Himax WEI Boot loader
-------------------------------------------------------------


embARC Build Time: Jan  4 2021, 13:44:14
Compiler Version: Metaware, 4.2.1 Compatible Clang 8.0.1
Boot loader Version : 1.4.4 (Date:Jan  4 2021)
chip version : 0x8535a1
cpu speed : 400000000 hz
spi speed : 50000000 hz
wake up evt:4
...secure lib version = 352380df9a347b1187d2361bfcd4455178a1ebcb
1st APPLICATION addr[3]=21000 (main-1966)
Bootloader Done !!!!!!
jump to app FW : 0x10000004
Microphone Initialize Success
Microphone Enable Success
Wait for user press key: [A]
```

# Lab2: Microphone

- You can save terminal text to a log file



```
------------
------------
ng 8.0.1
fcd4455178a1ebcb
```

Save in: Lab2_microphone

| Name | Date modified |
|---|---|
| Makefile | 2021/3/19 下午 01:45 |
| memory.map | 2021/4/6 下午 02:38 |
| memory.x | 2021/3/15 下午 06:12 |
| output.elf | 2021/4/6 下午 02:38 |
| output_gnu.img | 2021/4/6 下午 02:38 |

File name: teraterm.log

Save as type: 所有文件(*.*)

選項

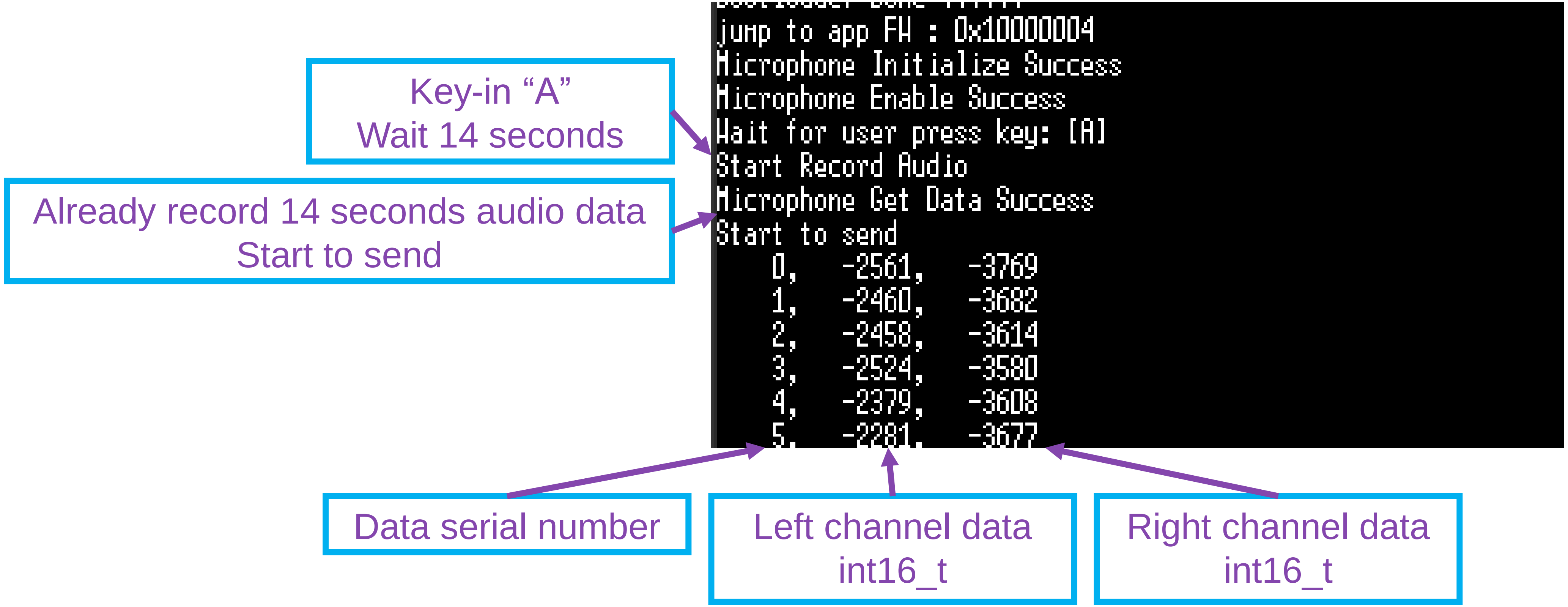
□ 二進制(R) ☑ 追加(A) ☑ 純文本(P)

□ 隱藏對話框(D) □ 包括當前緩衝區(C)

□ 時間戳(T) Local Time

# Lab2: Microphone

- Now, it is recoding and saving log file.
- After key-in “A”, WE-I start to recode 14 seconds audio data and send.



```
bootloader done !!!!!!
jump to app FW : 0x10000004
Microphone Initialize Success
Microphone Enable Success
Wait for user press key: [A]
Start Record Audio
Microphone Get Data Success
Start to send
    0,   -2561,   -3769
    1,   -2460,   -3682
    2,   -2458,   -3614
    3,   -2524,   -3580
    4,   -2379,   -3608
    5,   -2281,   -3677
```

Key-in “A”
Wait 14 seconds

Already record 14 seconds audio data
Start to send

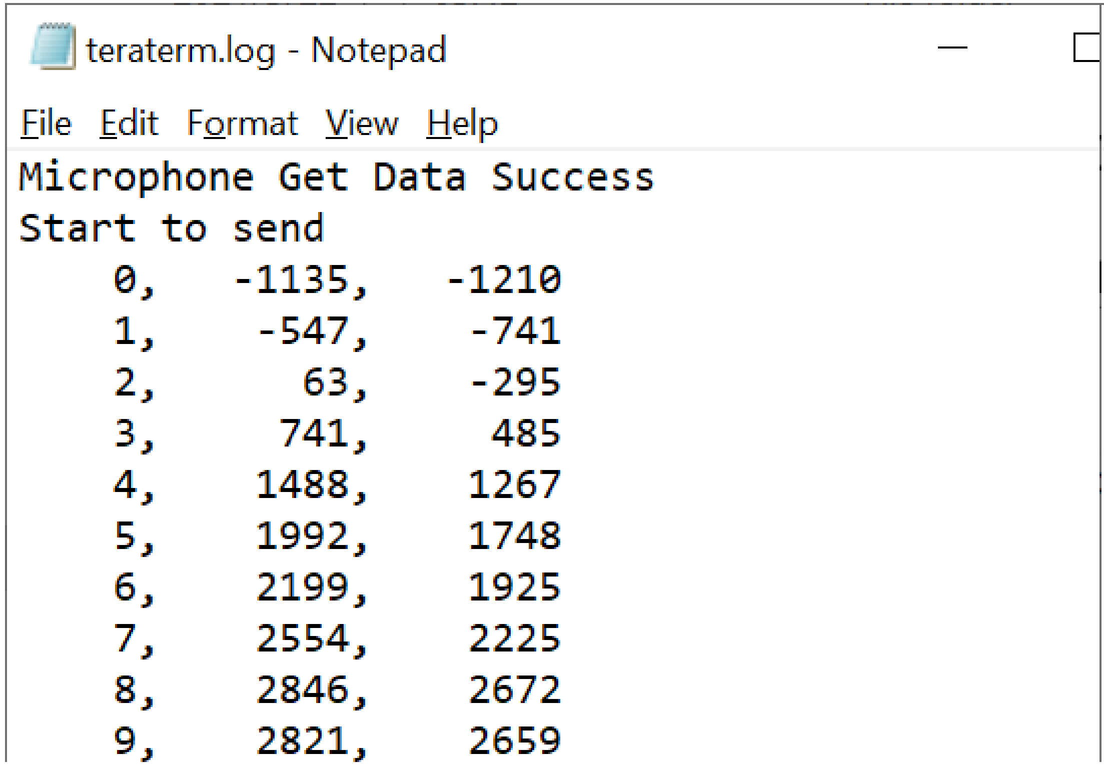
Data serial number

Left channel data
int16_t

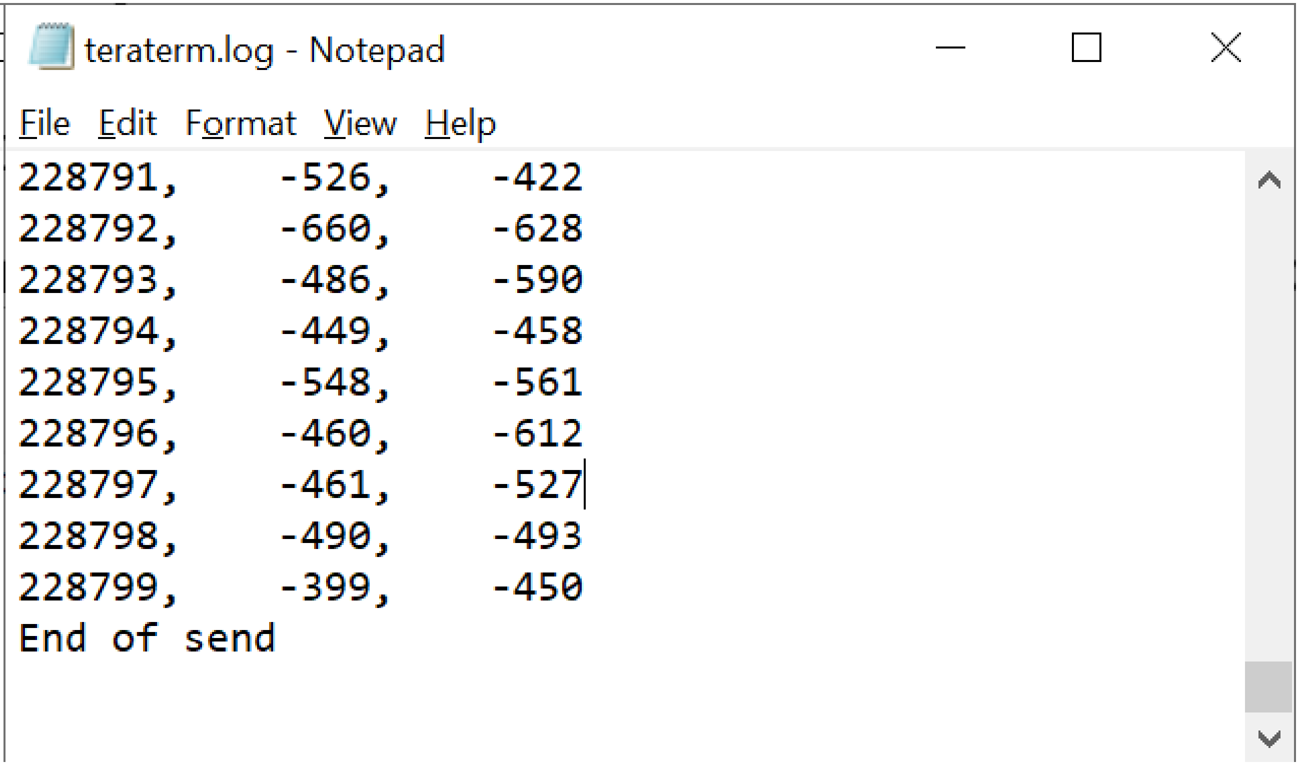
Right channel data
int16_t

# Lab2: Microphone

- It will take a lot of time to send data.
- After WE-I finished sending, close terminal log function and you can open log file.



```
Microphone Get Data Success
Start to send
    0,   -1135,   -1210
    1,    -547,    -741
    2,      63,    -295
    3,     741,     485
    4,    1488,    1267
    5,    1992,    1748
    6,    2199,    1925
    7,    2554,    2225
    8,    2846,    2672
    9,    2821,    2659
```

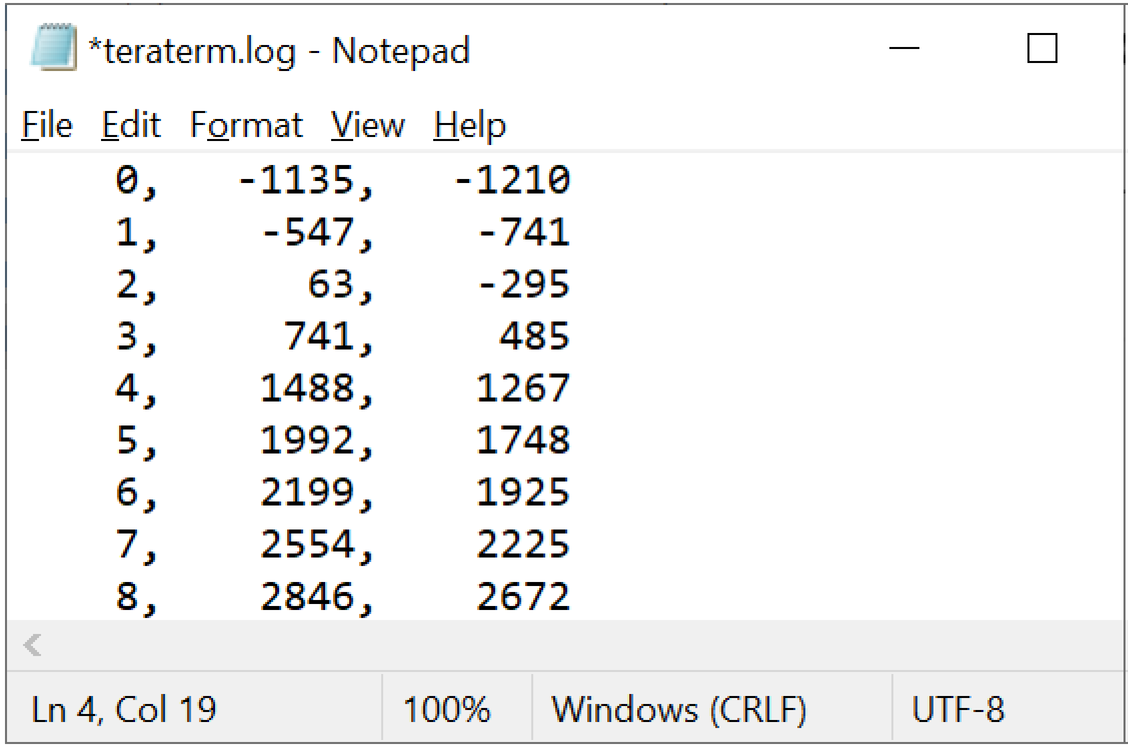


```
228791,    -526,    -422
228792,    -660,    -628
228793,    -486,    -590
228794,    -449,    -458
228795,    -548,    -561
228796,    -460,    -612
228797,    -461,    -527
228798,    -490,    -493
228799,    -399,    -450
End of send
```

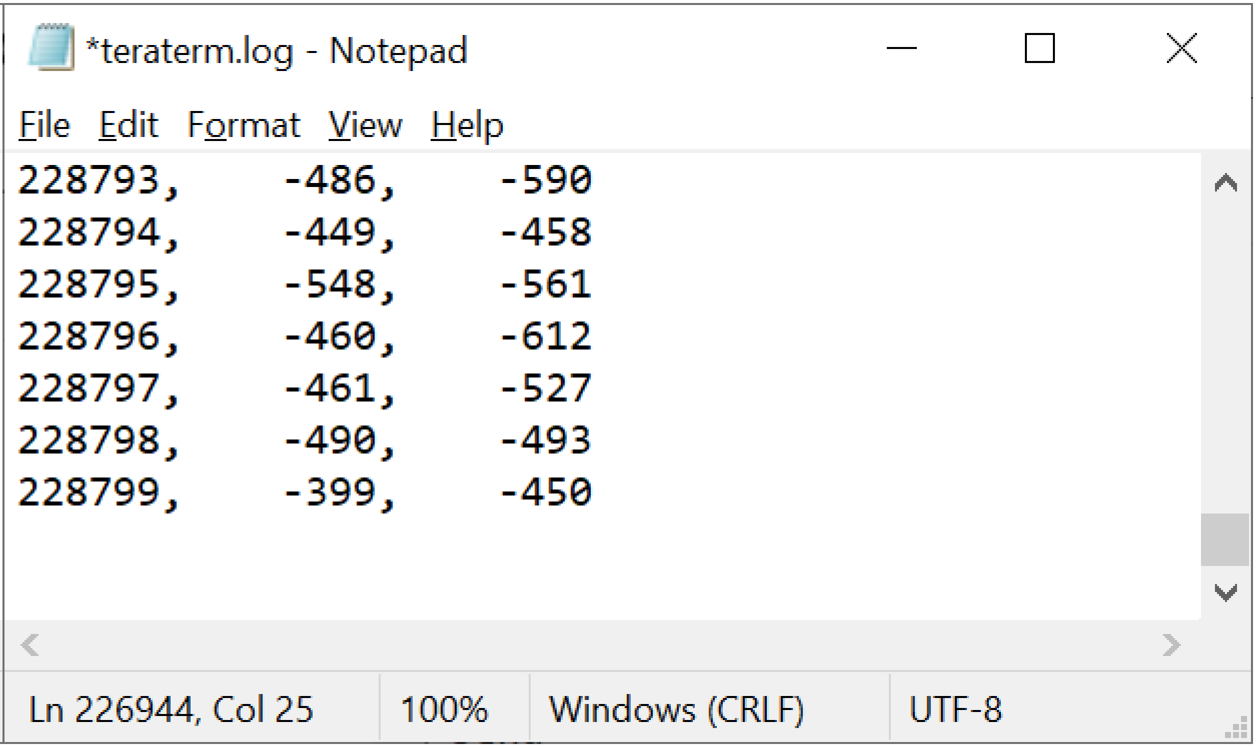
# Lab2: Microphone

- We also provide python code, help you convert log file to wav file

1. Delete first, second and last log message



```
    0,   -1135,   -1210
    1,    -547,    -741
    2,      63,    -295
    3,     741,     485
    4,    1488,    1267
    5,    1992,    1748
    6,    2199,    1925
    7,    2554,    2225
    8,    2846,    2672
```



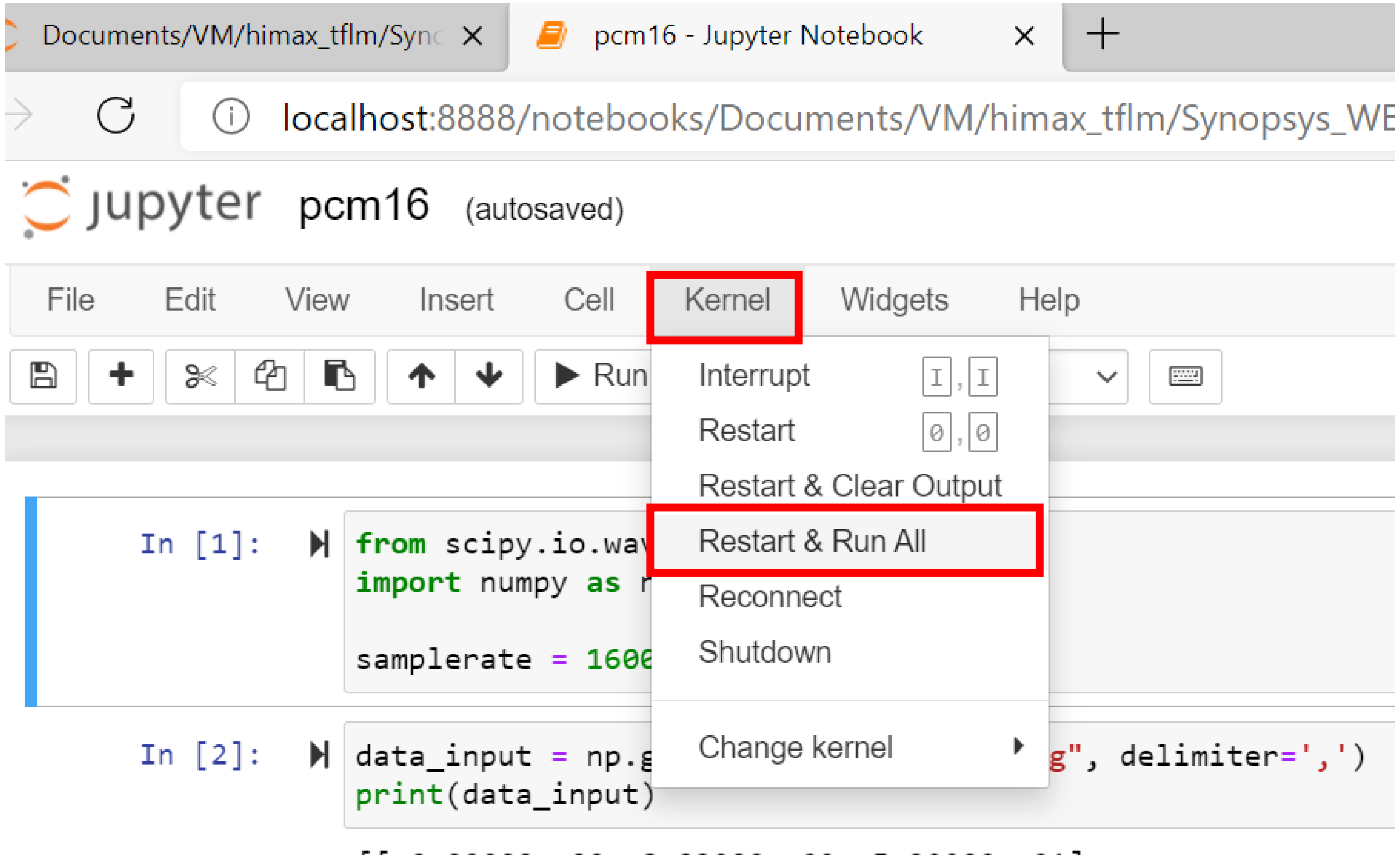
```
228793,     -486,     -590
228794,     -449,     -458
228795,     -548,     -561
228796,     -460,     -612
228797,     -461,     -527
228798,     -490,     -493
228799,     -399,     -450
```

# Lab2: Microphone

2. Copy log file to “./py_pcm16_wav/”
3. Rename log file to “pcm_dual.log”
4. Open Jupyter Notebook
5. Open “./py_pcm16_wav/pcm16.ipynb”

# Lab2: Microphone

6. Press “Restart and Run All”, then wav file will be saved path: “py_pcm16_wav/pcm16_example.wav”

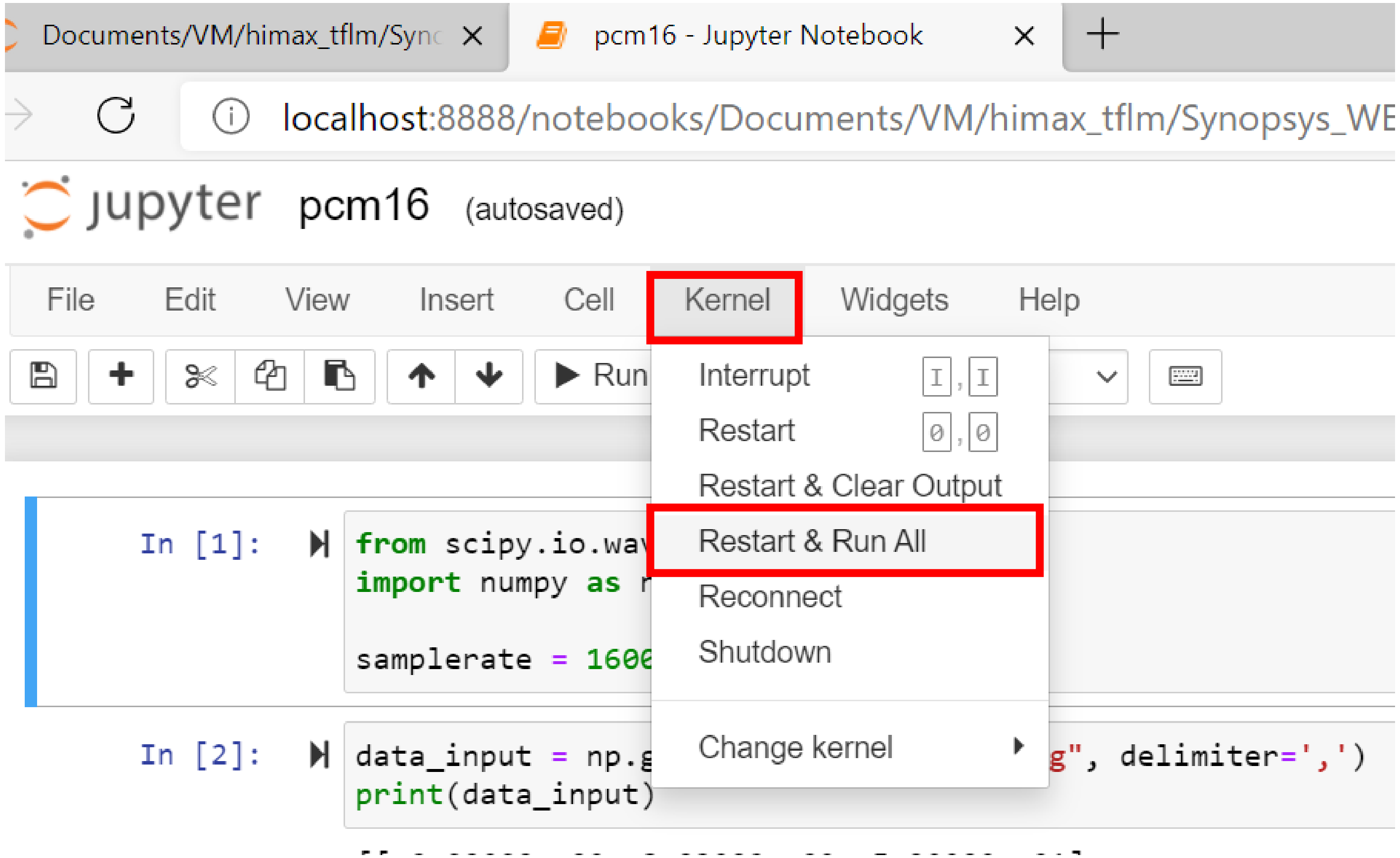


```
In [1]:  from scipy.io.wav
         import numpy as r

         samplerate = 160

In [2]:  data_input = np.g                g", delimiter=',')
         print(data_input)
```

# Lab2: Microphone

7. Now can play “pcm16_example.wav”

   You will know what microphone recorded in 14 seconds



```
In [1]:  from scipy.io.wav
         import numpy as r

         samplerate = 1600

In [2]:  data_input = np.g                    g", delimiter=',')
         print(data_input)
```

# Hands-on (Lab 3): Camera

# Lab3: Camera

- Camera initial

```
extern HX_DRV_ERROR_E hx_drv_sensor_initial(hx_drv_sensor_image_config_t
*pimg_config);
// Image sensor initialization, query one JPEG and one RAW frame to target address.
// Current image sensor use is HM0360, image resolution is 640x480.
// Each pixel data is 8-bit.
// (hx_drv_sensor_image_config_t *pimg_config) options are bellow
    pimg_config.img_width     /**< image width, assigned by driver */
    pimg_config.img_height    /**< image height, assigned by driver */
    pimg_config.jpeg_address  /**< JPEG image address, assigned by driver */
    pimg_config.jpeg_size     /**< JPEG image size, assigned by driver */
    pimg_config.raw_address   /**< RAW image address, assigned by driver */
    pimg_config.raw_size      /**< RAW image size, assigned by driver */
```

# Lab3: Camera

- Camera initial

Ex:
```
hx_drv_sensor_image_config_t pimg_config;
hx_drv_mic_initial();
```

# Lab3: Camera

- Camera image capture

```
extern HX_DRV_ERROR_E hx_drv_sensor_capture(hx_drv_sensor_image_config_t
*pimg_config);
// Query Image sensor and capture one JPEG frame and one RAW frame,
// sensor back to standby mode then.
// both RAW frame and JPEG frame will be provided to target address.
```

Ex: hx_drv_sensor_capture(&pimg_config);

# Lab3: Camera

- Camera image send by SPI (methon-1)

```
extern HX_DRV_ERROR_E hx_drv_spim_send(uint32_t addr, uint32_t size,
HX_DRV_SPI_TYPE data_type);
//  SPI master send data from dedicated memory address
//  SPI master and I2C master share the same output pin,
//  we need to switch to needed output mode.
//  You can send JPG or RAW by this API.
```

```
Ex: hx_drv_share_switch(SHARE_MODE_SPIM);
    hx_drv_spim_send(pimg_config.jpeg_address, pimg_config.jpeg_size, SPI_TYPE_JPG);

Ex: hx_drv_share_switch(SHARE_MODE_SPIM);
    hx_drv_spim_send(pimg_config.raw_address, pimg_config.raw_size, SPI_TYPE_RAW);
```

# Lab3: Camera

- Camera image send by UART (methon-2)

```
Ex: uint8_t * img_ptr;
    img_ptr = (uint8_t *) pimg_config.raw_address;

    for(uint32_t heigth_cnt = 0; heigth_cnt < pimg_config.img_height; heigth_cnt ++) {
        for(uint32_t width_cnt = 0; width_cnt < pimg_config.img_width; width_cnt ++) {
            hx_drv_uart_print("%3d", *img_ptr);
            if(width_cnt != (pimg_config.img_width - 1))
                hx_drv_uart_print(", ");
            else
                hx_drv_uart_print("\n");

            img_ptr = img_ptr + 1;
        }
    }
```

# Lab3: Camera

## Conclusion

- Camera should initialize before capture image.

```
HX_DRV_ERROR_E hx_drv_sensor_initial(hx_drv_sensor_image_config_t *pimg_config);
```

- Capture image.

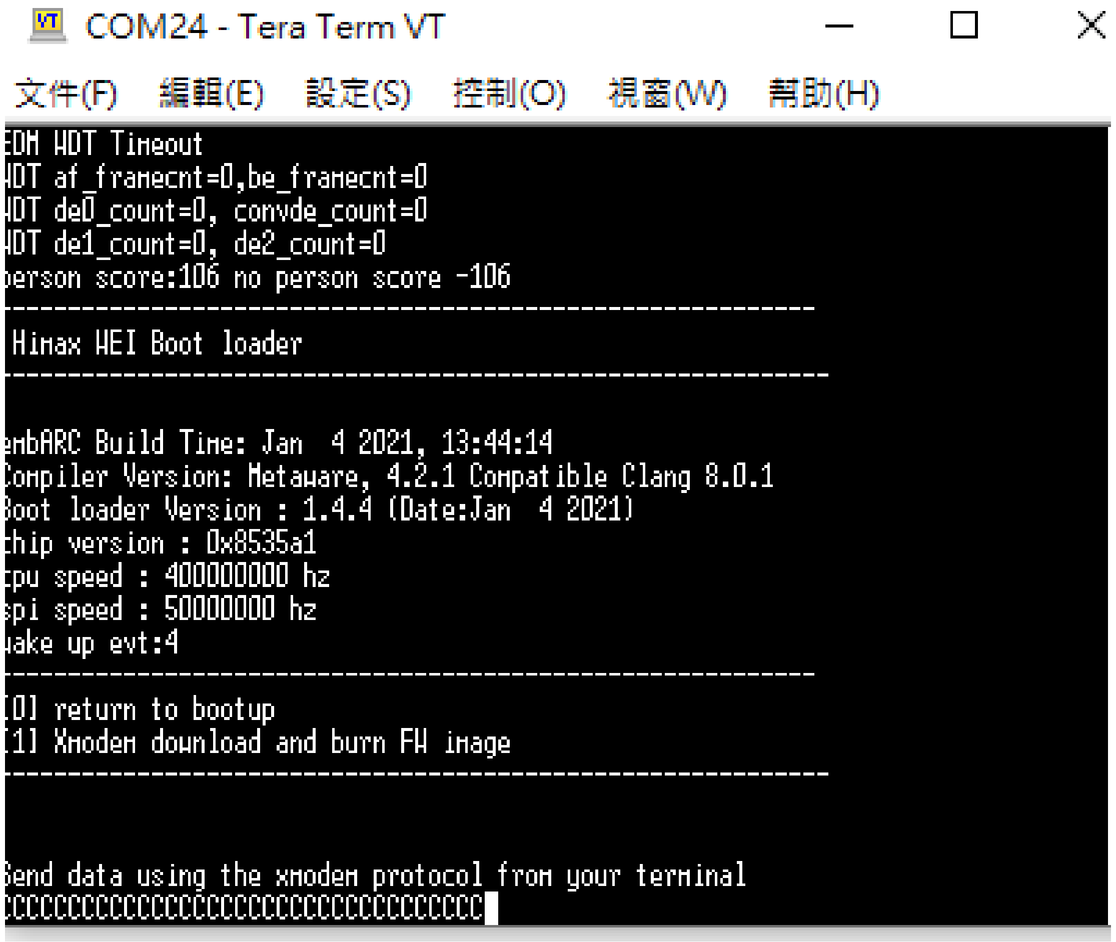
```
HX_DRV_ERROR_E hx_drv_sensor_capture(hx_drv_sensor_image_config_t *pimg_config);
```

- Send JPG or RAW image data by SPI. (Need to select output pin function)

```
HX_DRV_ERROR_E hx_drv_share_switch(HX_DRV_SHARE_MODE_E mode);
HX_DRV_ERROR_E hx_drv_spim_send(uint32_t addr, uint32_t size, HX_DRV_SPI_TYPE
data_type);
```

# Lab3: Camera

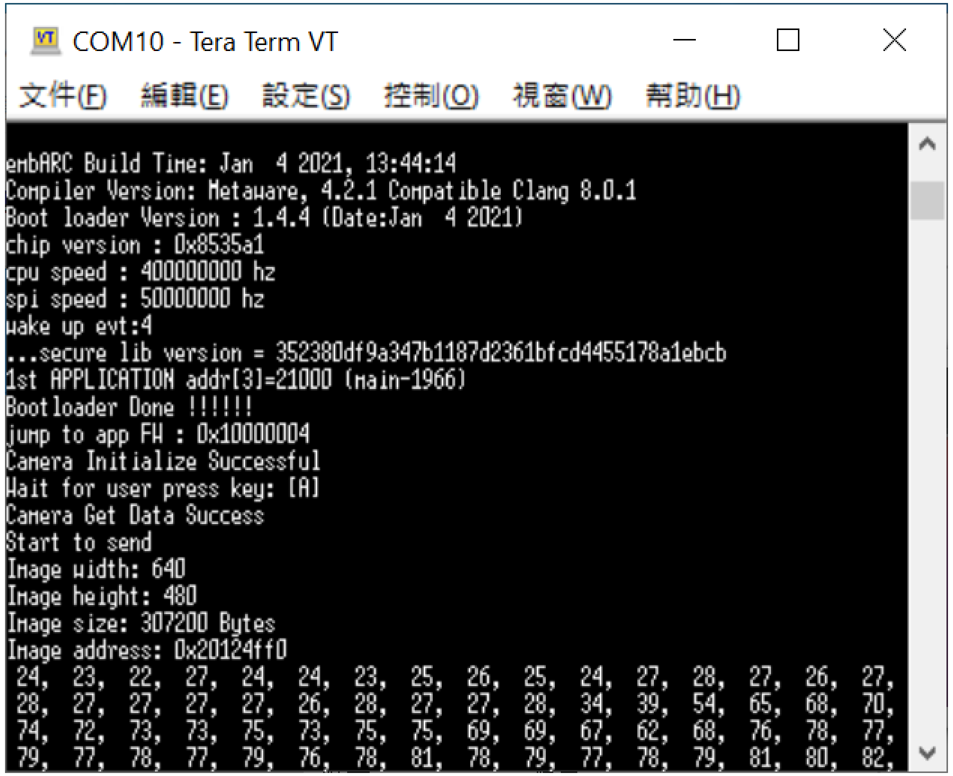
- Push WE-I reset button, and download image file to WE-I



```
EDM WDT Timeout
WDT af_framecnt=0,be_framecnt=0
WDT de0_count=0, convde_count=0
WDT de1_count=0, de2_count=0
person score:106 no person score -106
------------------------------------------------------------
 Himax WEI Boot loader
-------------------------------------------------------------

embARC Build Time: Jan  4 2021, 13:44:14
Compiler Version: Metaware, 4.2.1 Compatible Clang 8.0.1
Boot loader Version : 1.4.4 (Date:Jan  4 2021)
chip version : 0x8535a1
cpu speed : 400000000 hz
spi speed : 50000000 hz
wake up evt:4
------------------------------------------------------------
[0] return to bootup
[1] Xmodem download and burn FW image
-------------------------------------------------------------



Send data using the xmodem protocol from your terminal
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
```

# Lab3: Camera

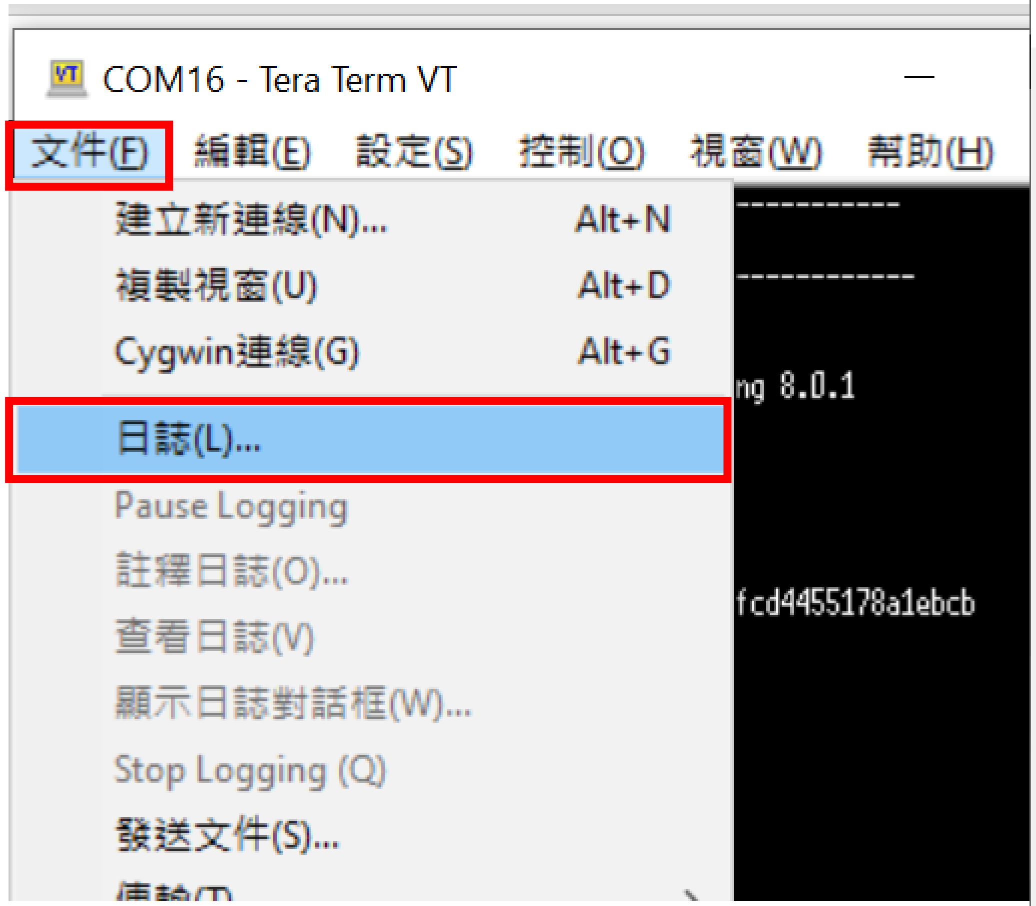
- This example project will wait user key-in “A”, and then capture and send RAW image data.
- You can save RAW image data data by terminal log function



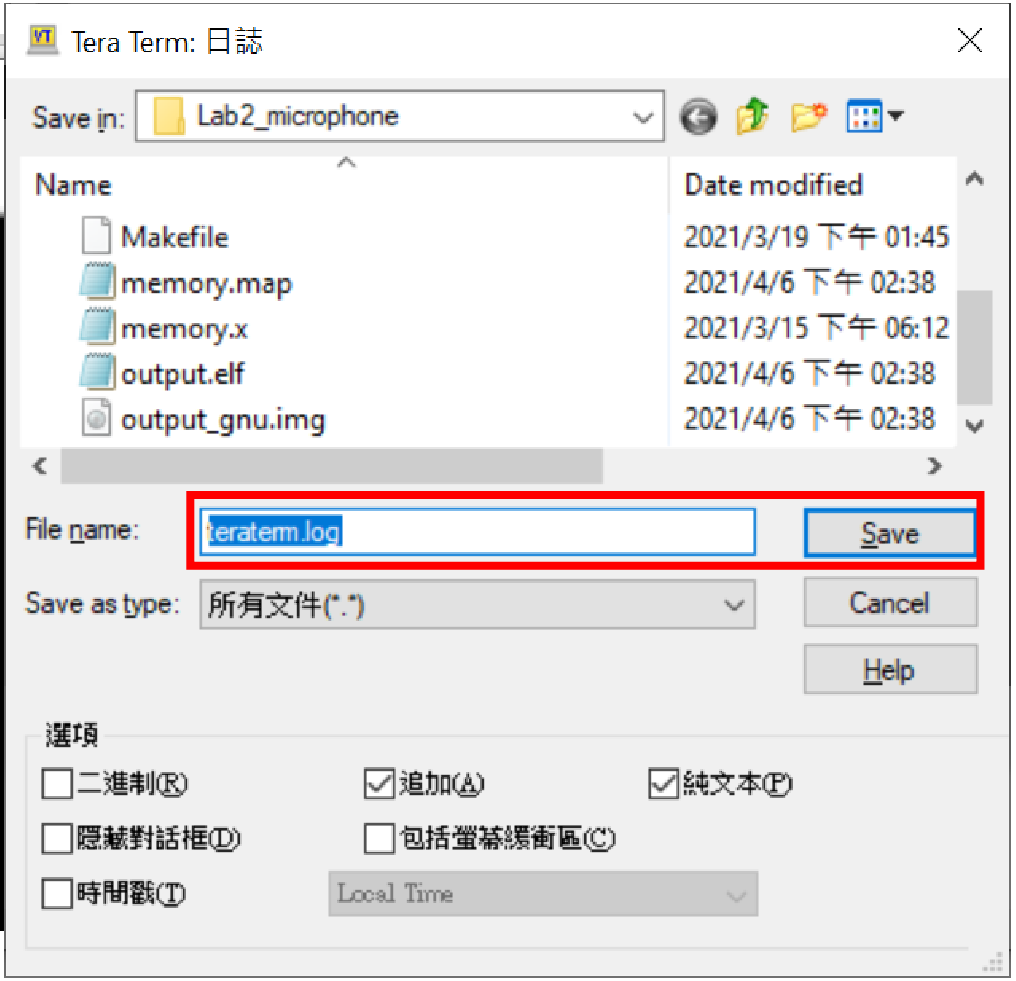
```
embARC Build Time: Jan  4 2021, 13:44:14
Compiler Version: Metaware, 4.2.1 Compatible Clang 8.0.1
Boot loader Version : 1.4.4 (Date:Jan  4 2021)
chip version : 0x8535a1
cpu speed : 400000000 hz
spi speed : 50000000 hz
wake up evt:4
...secure lib version = 352380df9a347b1187d2361bfcd4455178a1ebcb
1st APPLICATION addr[3]=21000 (main-1966)
Bootloader Done !!!!!!
jump to app FW : 0x10000004
Camera Initialize Successful
Wait for user press key: [A]
Camera Get Data Success
Start to send
Image width: 640
Image height: 480
Image size: 307200 Bytes
Image address: 0x20124ff0
 24,  23,  22,  27,  24,  24,  23,  25,  26,  25,  24,  27,  28,  27,  26,  27,
 28,  27,  27,  27,  27,  26,  28,  27,  27,  28,  34,  39,  54,  65,  68,  70,
 74,  72,  73,  73,  75,  73,  75,  75,  69,  69,  67,  62,  68,  76,  78,  77,
 79,  77,  78,  77,  79,  76,  78,  81,  78,  79,  77,  78,  79,  81,  80,  82,
```

# Lab3: Camera

- You can save terminal text to a log file



```
-----------
------------
ng 8.0.1
fcd4455178a1ebcb
```



Save in: Lab2_microphone

| Name | Date modified |
|---|---|
| Makefile | 2021/3/19 下午 01:45 |
| memory.map | 2021/4/6 下午 02:38 |
| memory.x | 2021/3/15 下午 06:12 |
| output.elf | 2021/4/6 下午 02:38 |
| output_gnu.img | 2021/4/6 下午 02:38 |

File name: teraterm.log

Save as type: 所有文件(*.*)

選項

□ 二進制(R) ☑ 追加(A) ☑ 純文本(P)

□ 隱藏對話框(D) □ 包括當前緩衝區(C)

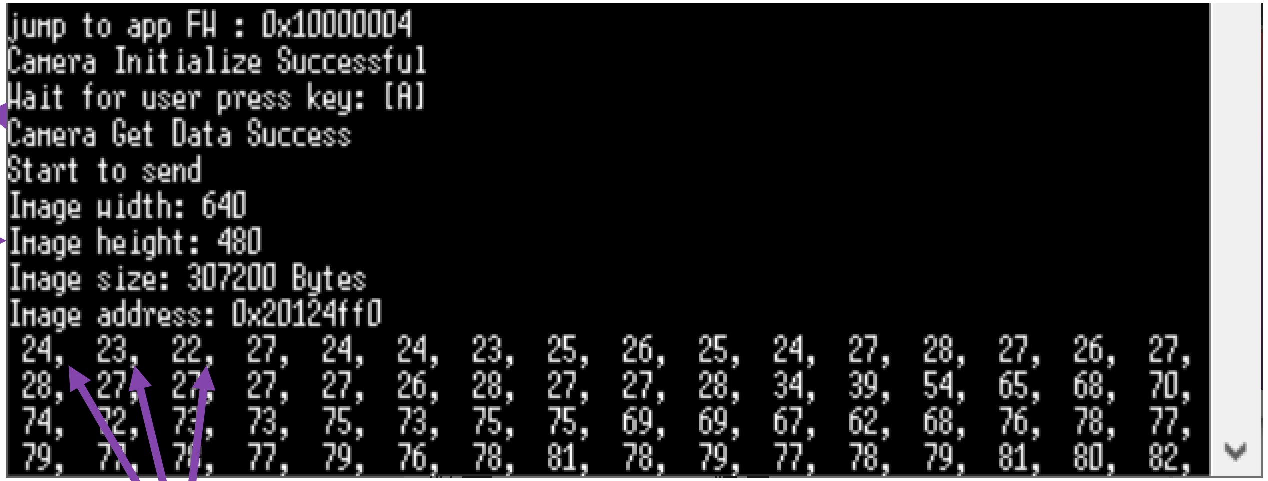
□ 時間戳(T) Local Time

# Lab3: Camera

- Now, it is recoding and saving log file.
- After key-in “A”, WE-I start to capture and send RAW image data.

Key-in “A”
Capture and Send

RAW image
data size

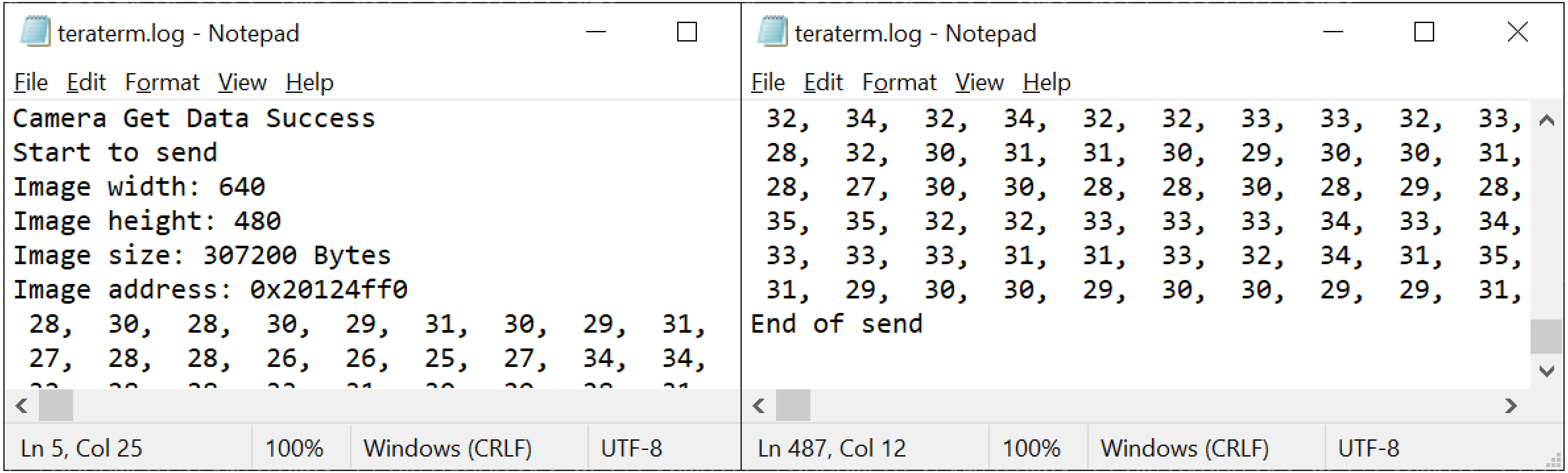


```
jump to app FW : 0x10000004
Camera Initialize Successful
Wait for user press key: [A]
Camera Get Data Success
Start to send
Image width: 640
Image height: 480
Image size: 307200 Bytes
Image address: 0x20124ff0
 24,  23,  22,  27,  24,  24,  23,  25,  26,  25,  24,  27,  28,  27,  26,  27,
 28,  27,  27,  27,  27,  26,  28,  27,  27,  28,  34,  39,  54,  65,  68,  70,
 74,  72,  73,  73,  75,  73,  75,  75,  69,  69,  67,  62,  68,  76,  78,  77,
 79,  77,  78,  77,  79,  76,  78,  81,  78,  79,  77,  78,  79,  81,  80,  82,
```

Each pixel

# Lab3: Camera

- It will take a lot of time to send data.
- After WE-I finished sending, close terminal log function and you can open log file.



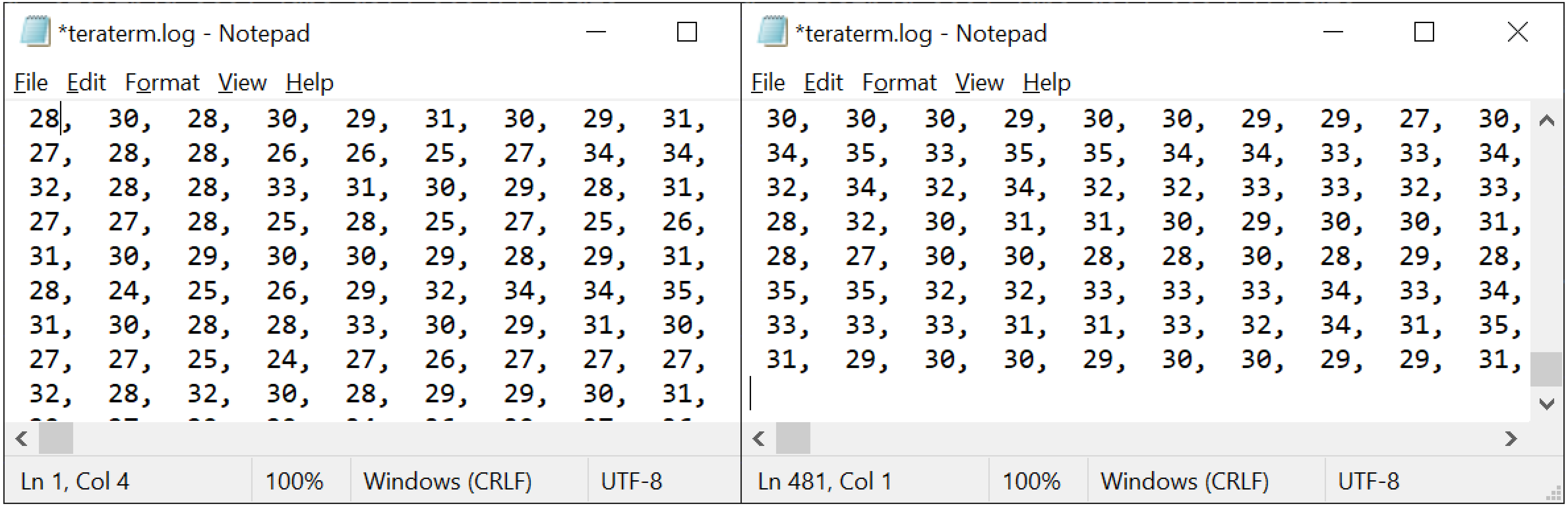
```
Camera Get Data Success
Start to send
Image width: 640
Image height: 480
Image size: 307200 Bytes
Image address: 0x20124ff0
 28,  30,  28,  30,  29,  31,  30,  29,  31,
 27,  28,  28,  26,  26,  25,  27,  34,  34,
```

```
 32,  34,  32,  34,  32,  32,  33,  33,  32,  33,
 28,  32,  30,  31,  31,  30,  29,  30,  30,  31,
 28,  27,  30,  30,  28,  28,  30,  28,  29,  28,
 35,  35,  32,  32,  33,  33,  33,  34,  33,  34,
 33,  33,  33,  31,  31,  33,  32,  34,  31,  35,
 31,  29,  30,  30,  29,  30,  30,  29,  29,  31,
End of send
```

# Lab3: Camera

- We also provide python code, help you convert log file to png file

1. Delete line 1~6 and last log message



```
 28,   30,   28,   30,   29,   31,   30,   29,   31,
 27,   28,   28,   26,   26,   25,   27,   34,   34,
 32,   28,   28,   33,   31,   30,   29,   28,   31,
 27,   27,   28,   25,   28,   25,   27,   25,   26,
 31,   30,   29,   30,   30,   29,   28,   29,   31,
 28,   24,   25,   26,   29,   32,   34,   34,   35,
 31,   30,   28,   28,   33,   30,   29,   31,   30,
 27,   27,   25,   24,   27,   26,   27,   27,   27,
 32,   28,   32,   30,   28,   29,   29,   30,   31,
```

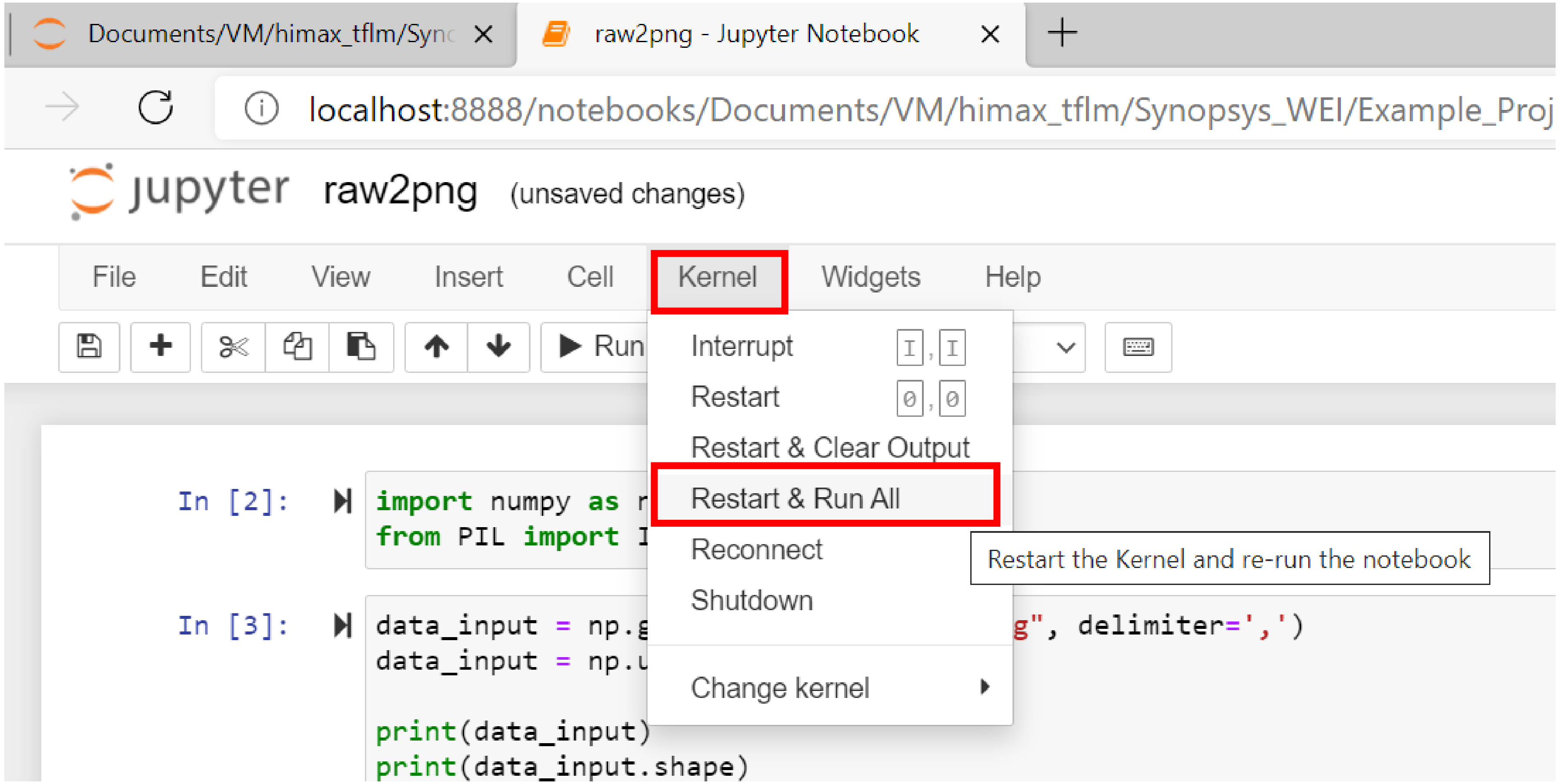
```
 30,   30,   30,   29,   30,   30,   29,   29,   27,   30,
 34,   35,   33,   35,   35,   34,   34,   33,   33,   34,
 32,   34,   32,   34,   32,   32,   33,   33,   32,   33,
 28,   32,   30,   31,   31,   30,   29,   30,   30,   31,
 28,   27,   30,   30,   28,   28,   30,   28,   29,   28,
 35,   35,   32,   32,   33,   33,   33,   34,   33,   34,
 33,   33,   33,   31,   31,   33,   32,   34,   31,   35,
 31,   29,   30,   30,   29,   30,   30,   29,   29,   31,
```

# Lab3: Camera

2. Copy log file to “./py_raw2png/”
3. Rename log file to “camera_y.log”
4. Open Jupyter Notebook
5. Open “./py_raw2png/raw2png.ipynb”

# Lab3: Camera

6. Press “Restart and Run All”, then png file will be saved path: “py_raw2png/my_gray.png”



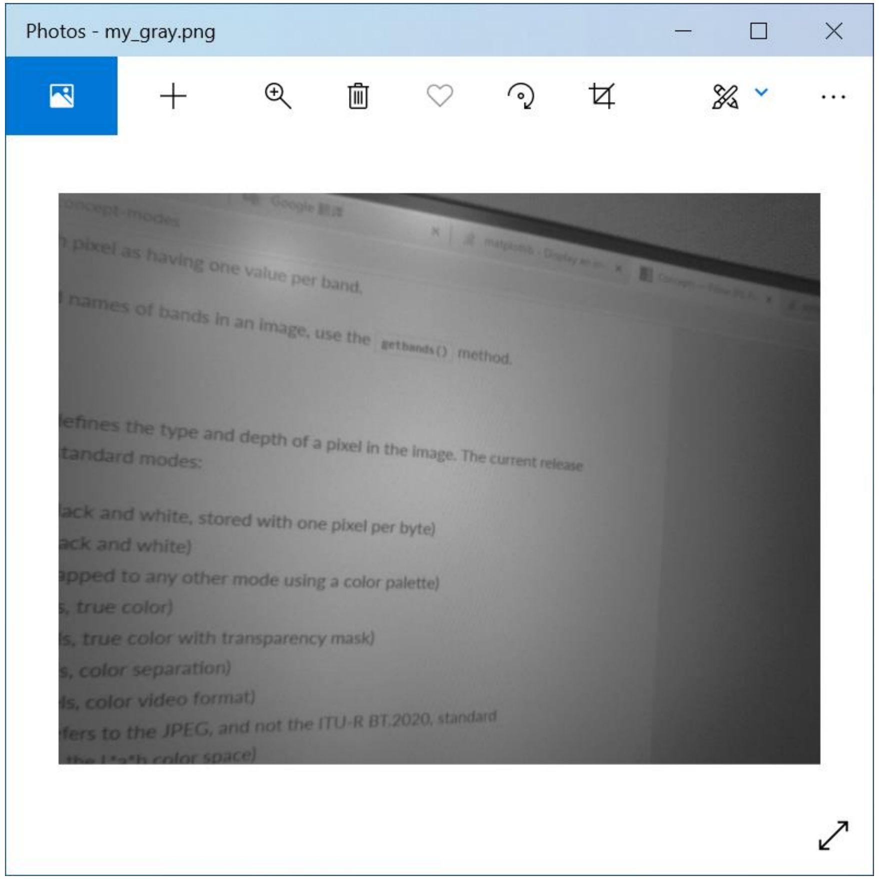
```
In [2]:  import numpy as r
         from PIL import I

In [3]:  data_input = np.g                    g", delimiter=',')
         data_input = np.u

         print(data_input)
         print(data_input.shape)
```

Documents/VM/himax_tflm/Sync
raw2png - Jupyter Notebook
localhost:8888/notebooks/Documents/VM/himax_tflm/Synopsys_WEI/Example_Proj
jupyter
raw2png
(unsaved changes)
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Run
Interrupt
I, I
Restart
0, 0
Restart & Clear Output
Restart & Run All
Reconnect
Shutdown
Change kernel
Restart the Kernel and re-run the notebook

# Lab3: Camera

7. Now can show “my_gray.png”

   You will know what camera captured

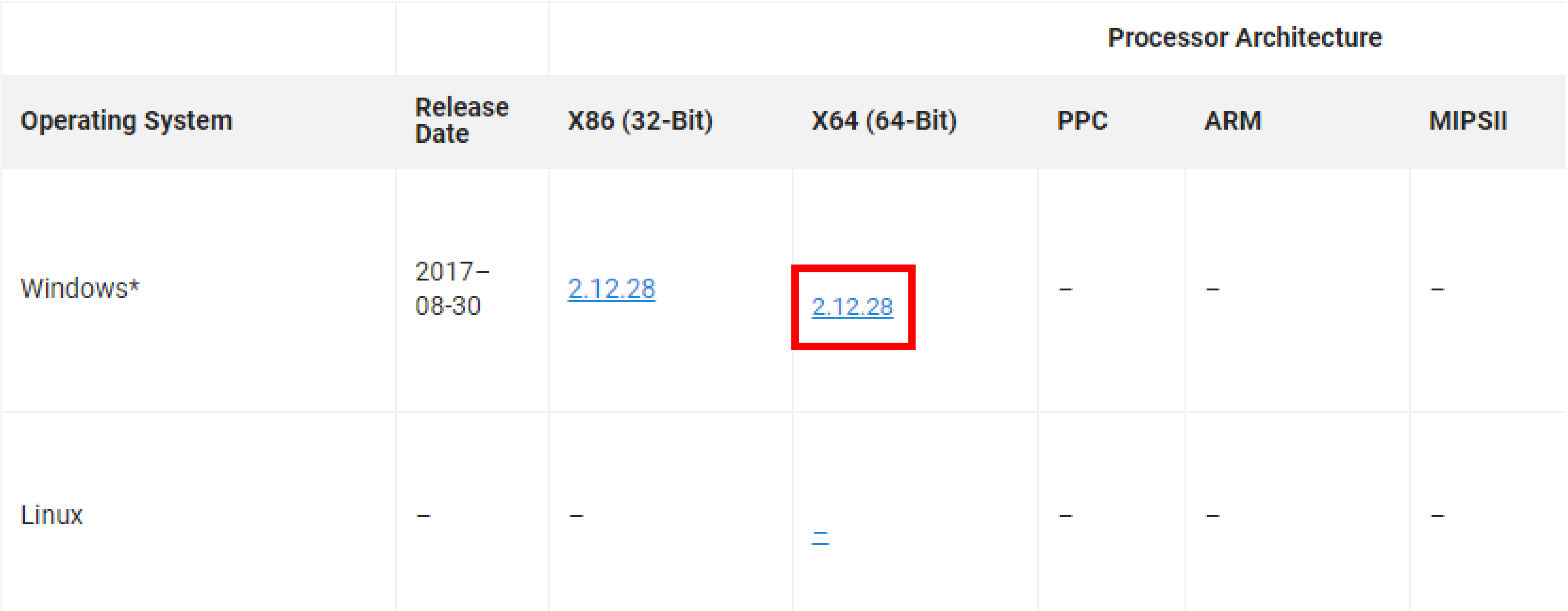

Photos - my_gray.png
pixel as having one value per band.
names of bands in an image, use the getbands() method.
efines the type and depth of a pixel in the image. The current release
tandard modes:
ack and white, stored with one pixel per byte)
ack and white)
apped to any other mode using a color palette)
s, true color)
s, true color with transparency mask)
s, color separation)
ls, color video format)
fers to the JPEG, and not the ITU-R BT.2020, standard
the L*a*b color space)

# Appendix-2: Troubleshooting - Update VCP Driver

# Troubleshooting – Update VCP driver

If the USB serial port is not shown in Ports (COM & LPT):

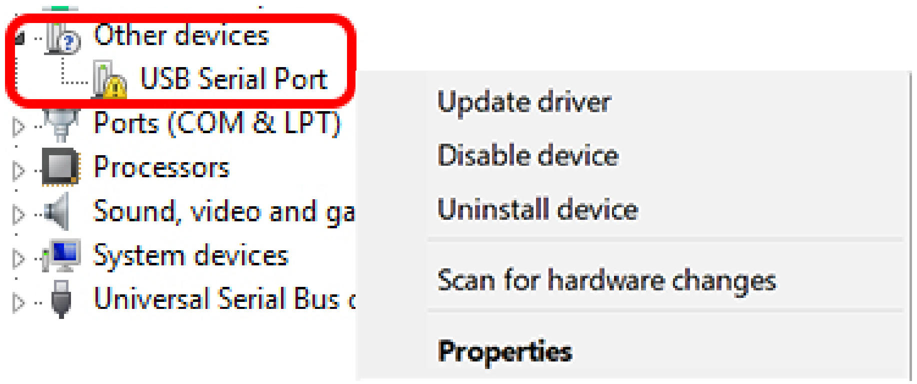
1. Download VCP driver: https://ftdichip.com/drivers/vcp-drivers/
   Select Windows/X64 version
2. Unzip the downloaded file (CDM v2.12.28 WHQL Certified)

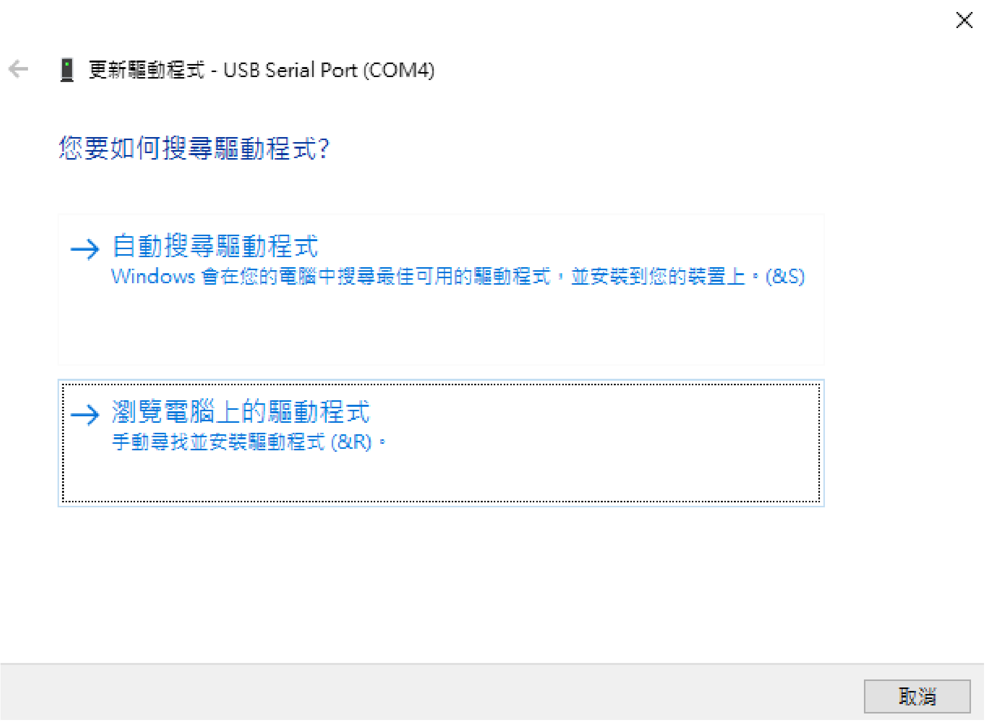


| | | Processor Architecture | | | | |
|---|---|---|---|---|---|---|
| Operating System | Release Date | X86 (32-Bit) | X64 (64-Bit) | PPC | ARM | MIPSII |
| Windows* | 2017–08-30 | 2.12.28 | 2.12.28 | – | – | – |
| Linux | – | – | – | – | – | – |

# Troubleshooting – Update VCP driver

3. Click Other devices > USB Serial Port > Update driver
4. Choose “瀏覽電腦上的驅動程式”





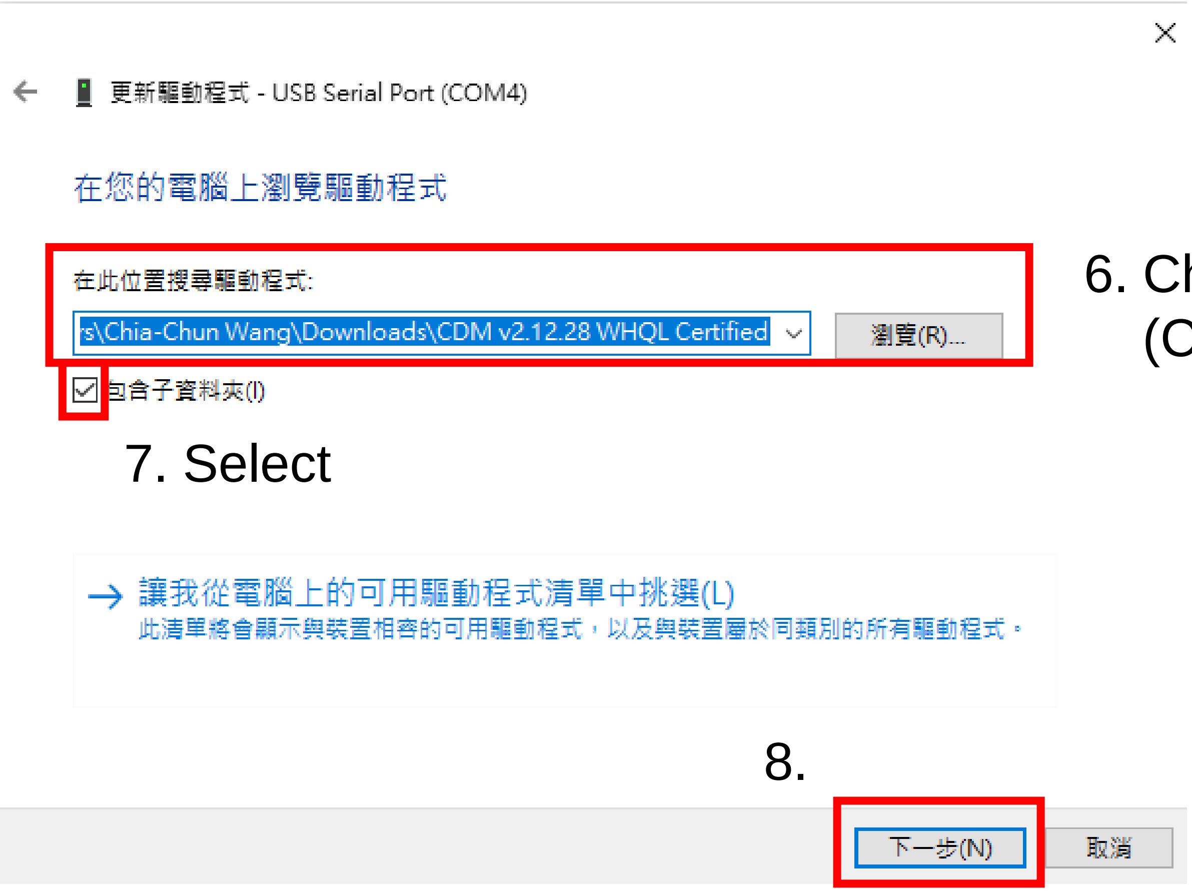
更新驅動程式 - USB Serial Port (COM4)

您要如何搜尋驅動程式?

→ 自動搜尋驅動程式
Windows 會在您的電腦中搜尋最佳可用的驅動程式，並安裝到您的裝置上。(&S)

→ 瀏覽電腦上的驅動程式
手動尋找並安裝驅動程式 (&R)。

# Troubleshooting – Update VCP driver



更新驅動程式 - USB Serial Port (COM4)

在您的電腦上瀏覽驅動程式

在此位置搜尋驅動程式:

rs\Chia-Chun Wang\Downloads\CDM v2.12.28 WHQL Certified

☑ 包含子資料夾(I)

→ 讓我從電腦上的可用驅動程式清單中挑選(L)

此清單將會顯示與裝置相容的可用驅動程式，以及與裝置屬於同類別的所有驅動程式。

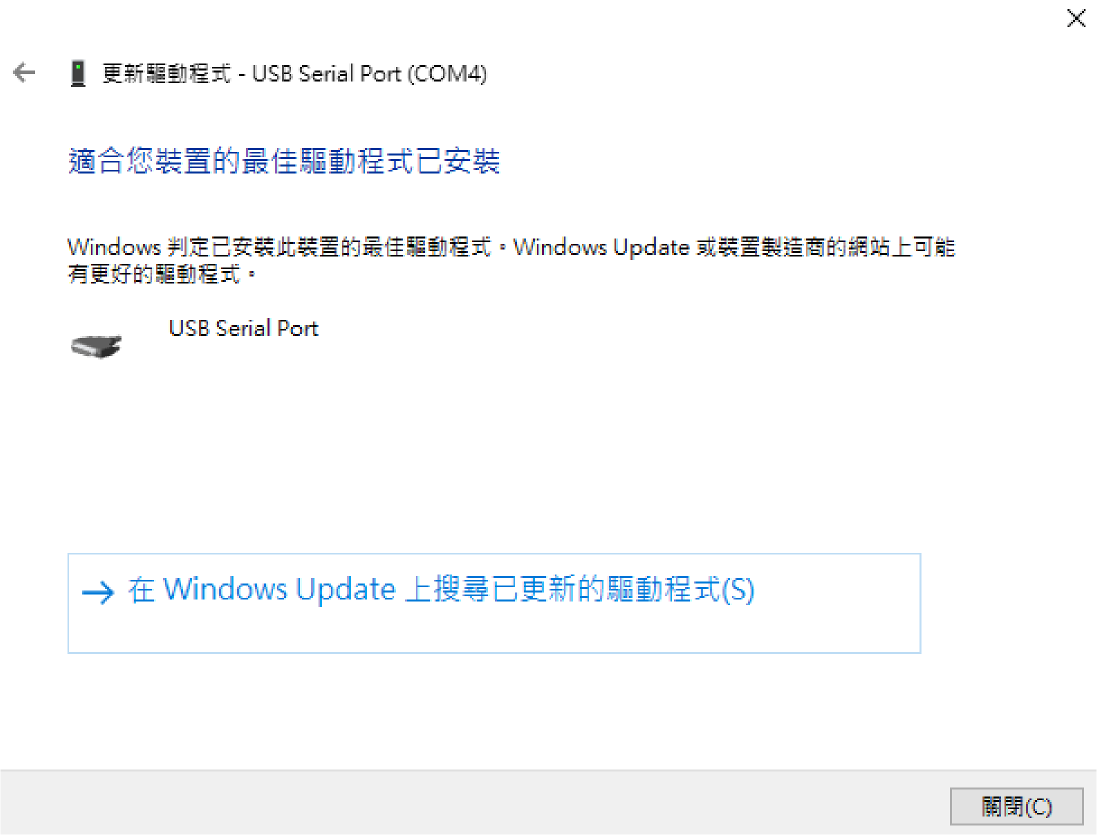
6. Choose the downloaded folder (CDM v2.12.28 WHQL Certified)

7. Select

8.

# Troubleshooting – Update VCP driver

9. Finish



更新驅動程式 - USB Serial Port (COM4)

適合您裝置的最佳驅動程式已安裝

Windows 判定已安裝此裝置的最佳驅動程式。Windows Update 或裝置製造商的網站上可能有更好的驅動程式。

USB Serial Port