WINC Devices – Integrated Serial Flash Memory Download Procedure

Introduction

ATWINC1500/ATWINC3400 features an on-chip microcontroller and integrated SPI Flash memory for system firmware. The serial flash memory also stores the root certificate required for TLS/SSL connection and the gain table values used by transceiver. This application note explains in detail downloading procedure of firmware, certificate, and gain values into WINC serial flash through different supported serial interfaces like UART/I2C. This document also covers some useful troubleshooting tips for downloading failures.

Features

- Firmware download procedure
- Root certificate download procedure
- Gain table values download procedure
- Troubleshooting tips
- Common download procedure for WINC1500 and WINC3400

Contents

1. **Introduction** 3
2. **Firmware update project** 3
   - **2.2 Hardware Setup** 4
   - **2.3 Project Overview** 5
3. **Serial Bridge Application** 7
   - **3.1 Serial Flash Download Using SAM Xplained Pro Board** 7
   - **3.1.1.2 Serial Flash Download Using Custom HostMCU** 10
4. **General Information on Firmware Update** 10
   - **4.1 ATWINC1500/ATWINC3400 Scripts** 10
   - **4.2 ATWINC1500/ATWINC3400 Binary Tools** 11
   - **4.2.1 Building Firmware Image** 11
   - **4.2.2 Programming Firmware Image** 17
   - **4.2.2.1 winc_programmer_uart Tool Usage** 17
5. **Download Failure Troubleshooting** 22
   - **5.1 The script Failed To Find Any COM Port** 22
   - **5.2 The script Found More Than One Matching Tool** 22
   - **5.3 The script Listing More Than One COM Port** 22
   - **5.4 Failed To Initialize Programmer: Invalid Chip ID** 23
   - **5.5 Failed To Initialize Programmer: Waiting For Chip Permission** 23

1. Introduction

The WINC1500 or WINC3400 firmware update project can be retrieved through the Microchip® Advanced Software Framework (ASF). The latest firmware update project contains the new firmware images as well as the batch script files used to download the firmware, certificate, provisioning webpage and gain values into Wi-Fi Network Controller (WINC) through SPI/UART.

The document guide information:

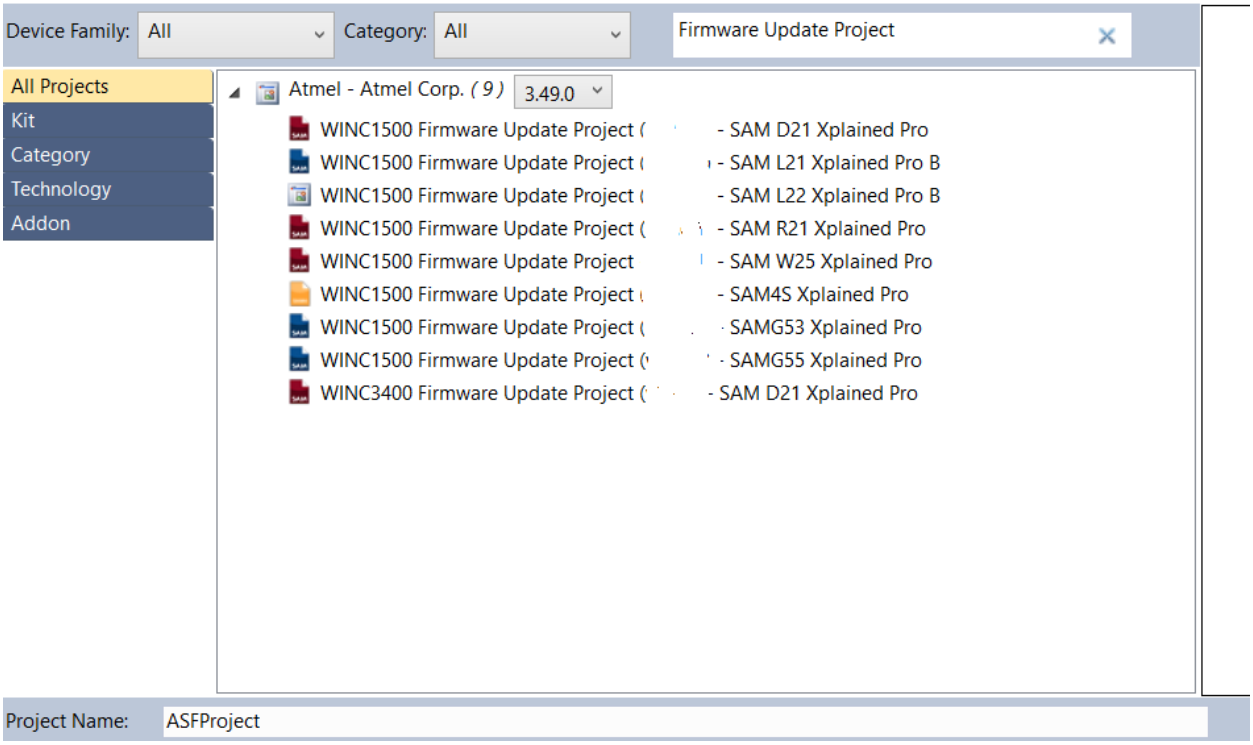
- Firmware update project:
    - User can find .bat file for required HostMCU in /src folder of “WINCXXXX_FIRMWARE_UPDATE_PROJECT” project.
    - For supported HostMCU user can follow the instruction available in section **Firmware update project**.
- Serial Bridge Application:
    - When user could not find the required HostMCU support batch file in /src folder of “WINCXXXX_FIRMWARE_UPDATE_PROJECT” project.
    - User can run the serial bridge application for the required HostMCU and add the generated elf into “WINCXXXX_FIRMWARE_UPDATE_PROJECT” project.
    - User can follow the instruction available in section **Serial Bridge Application**.
- General Information on Firmware Update
    - User can refer to this section for understanding how the binary tools can be used for building and programming the firmware image.
    - User can follow the instruction available in section **General Information on Firmware Update**.
- Download Failure Troubleshooting
    - When user faces issue while programming / updating the firmware, user can refer this section **Download Failure Troubleshooting**.

2. Firmware update project

The ATWINC1500 and ATWINC3400 WiFi devices require firmware to be loaded into flash memory. The ATWINC1500 and ATWINC3400 devices are preloaded with the firmware, however it would be useful to update the latest firmware to take advantage of fixes and new features.

2.1 Import Firmware Update Project

Pre-requisites: Install Microchip® Advanced Software Framework (ASF).The latest version of the Microchip® Advanced Software Framework (ASF) can be found on the Microchip Gallery or using Microchip Studio Extension manager.
1. Search for "Firmware Update Project" from the "New Example Project" of ASF menu in Microchip Studio.



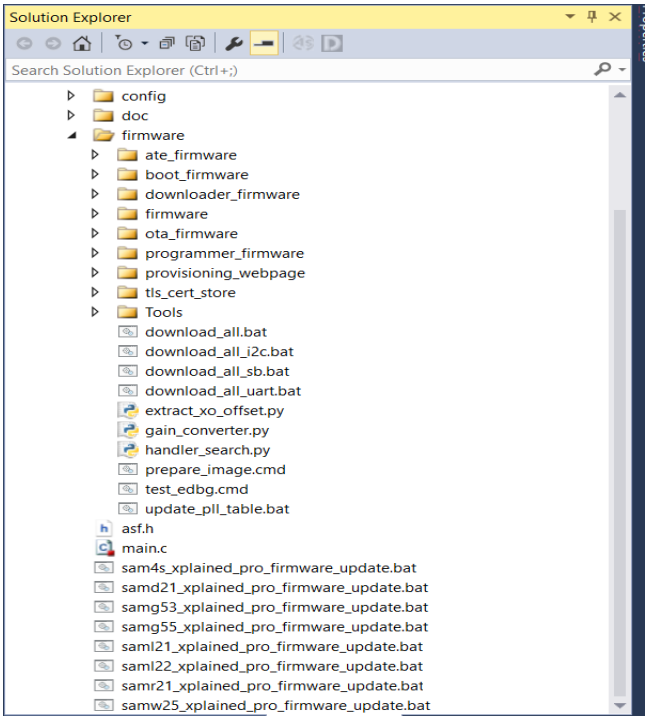
2. Select the appropriate "WINC Firmware Update Project (vxx.x.x)" project corresponding to the intended host MCU Xplained Pro board and then press OK button to import firmware update project and related documentation.

2.2 Hardware Setup

- SAM D21 Xplained Pro Evaluation Kit (ATSAMD21-XPRO) - SAMD21 is used as a HostMCU
- ATWINC1500 / WINC3400 Xplained PRO Evaluation Kit (ATWINC1500-XPRO / ATWINC3400-XPRO) - Wi-Fi SPI slave device connected to SAMD21 HostMCU device
- The ATWINC1500 / ATWINC3400 device is attached to EXT1 of the SAMD21 Xplained Pro kit.
- Plug a micro-USB cable from Windows computer to the debug USB port of the SAM D21 Xplained Pro Evaluation kit.

2.3 Project Overview

The firmware update project appears as a regular Microchip Studio ASF project.



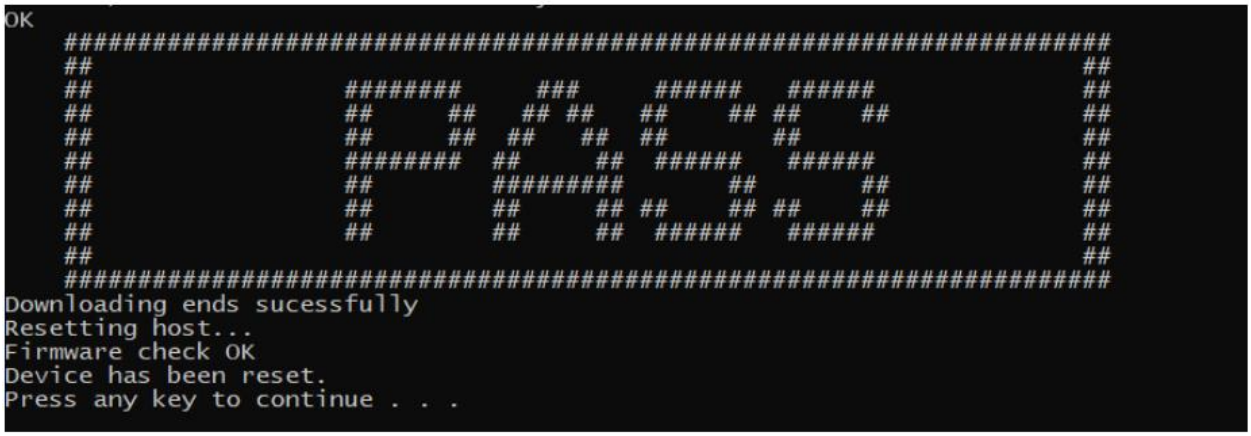
/src/firmware folder – Contains the new WINC firmware as well as:

- The download_all.bat script - To download the WINC firmware/certificate/gain values.
- /src folder – Contains update scripts to download the WINC firmware, certificate and gain values at one.

Ensure that the SAM Xplained Pro board is connected to PC via debug USB port.

Run the samxxx_xplained_pro_firmware_update.bat script that corresponds to connected Xplained board.

The script will print the following message in the successful case.



3. Serial Bridge Application

As the WINC device is connected to host MCU through SPI interface, upgrading the WINC serial flash via the host MCU would be an easier solution. Since, WINC provides transparent access to host MCU, the WINC serial flash can be read/written from host MCU. The host MCU can program the serial (SPI) flash without the need for operational firmware in the WINC. The host MCU running the serial bridge firmware is connected between computer and WINC SPI to download the firmware to WINC serial flash.

winc_programmer_UART (PC) <—-> samd21_xplained_pro_serial_bridge.elf(Host) <—-> WINC SPI device

3.1 Serial Flash Download Using SAM Xplained Pro Board

- The /src/firmware/Tools/serial_bridge shall contain the serial bridge binary images for available SAM based host MCU's.
- This serial bridge firmware uses UART interface available on SAM Xplained Pro boards.
- The batch script files available in the firmware update project /src folder contains the scripts to program the platform specific to serial bridge binary image on the host MCU before it starts the WINC serial flash download.
- EDBG on SAM Xplained Pro board is used for programming serial bridge image.
- The script uses the Microchip Studio atprogram.exe commands for programming the host MCU via EDBG of SAM Xplained Pro boards.

The same batch (.bat) script files in "src" folder inside "WINCXXXX_FIRMWARE_UPDATE_PROJECT" will trigger the WINC serial flash download.

- Ensure that the SAM Xplained Pro board is connected to PC via debug USB port.
- The virtual EDBG COM port of the board is now listed in the device manager.
- Run the samxxx_xplained_pro_firmware_update.bat script that corresponds to connected Xplained board.

A list of batch (.bat) script files in the /src folder of "WINCXXXX_FIRMWARE_UPDATE_PROJECT" shall be used to trigger a WINC serial flash download.

1. Ensure that the SAM Xplained Pro board is connected to PC via debug USB port. The virtual EDBG COM port of the board is now listed in the device manager.
2. Run the samxxx_xplained_pro_firmware_update.bat script that corresponds to connected Xplained board.

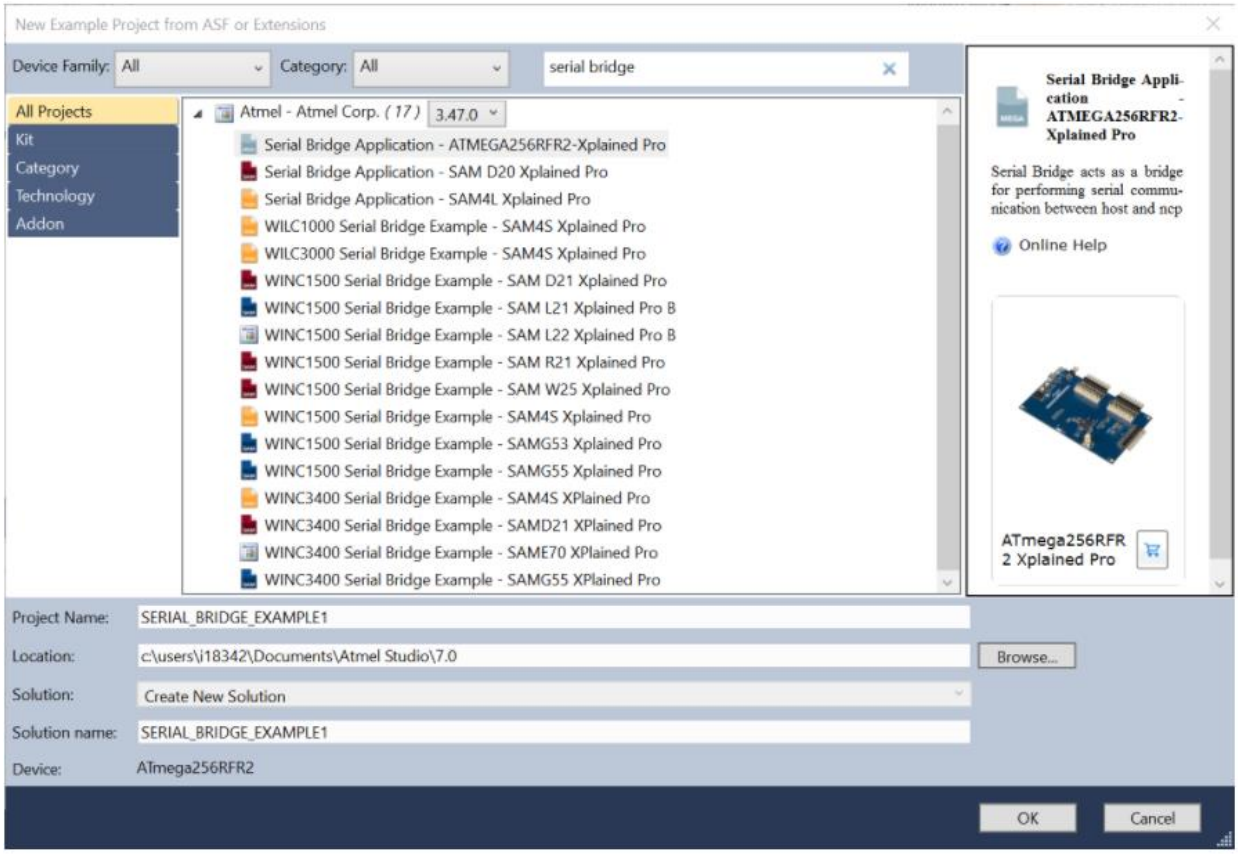
The batch script will program a serial bridge binary on the host MCU to redirect firmware data from the computer (EDBG virtual COM port) to the WINC chip (via SPI). The serial bridge application also performs the WINC power up sequence, thus ensuring that the WINC bootloader is in the appropriate state to start a download.

```
Erasing host...
Firmware check OK
Chiperase completed successfully
Programming host...
Firmware check OK
Programming completed successfully.
Resetting host...
Firmware check OK
Device has been reset.
```

The binary elf files which are available in the /src/firmware/Tools/serial_bridge directory is taken from the Serial bridge application.

3.1.1 Serial Bridge Application for Microchip Studio

- For Microchip Studio, Serial Bridge application can be imported in the same way how firmware update project is imported.



- Build the project.
- Copy the elf binary file to the /src/firmware/Tools/serial_bridge path inside the "WINCXXXX_FIRMWARE_UPDATE_PROJECT" project.
- Create a copy of batch file samxxx_xplained_pro_firmware_update.bat and rename according to your HostMCU name.
- In the created batch file replace the samxxx_xplained_pro_serial_bridge.elf with your generated elf file and change the HostMCU name.

- Run the newly created batch file.

3.1.1.1 Serial Flash Download via Built-in UART

The serial flash download can be done using the built-in UART of ATWINC1500 device. Prior to running any update script, ensure that the hardware is setup as required.

Note: ATWINC3400 does not support download through built-in UART at present.

Hardware Setup

Power On Sequence

- To perform a serial flash download using the ATWINC1500 built-in UART, it is mandatory that the ATWINC1500 chip is in the right bootloader state.
- To do so, the HostMCU must power up the ATWINC1500 chip and then perform the reset sequence as defined in the ATWINC1500 datasheet.
- This can be done very easily from the HostMCU by calling the m2m_bsp_init() function.
- Please find the below example code as reference.

```
int main(void)
{
/* Initialize the board. */
system_init();
/* Initialize the BSP. */
nm_bsp_init();
while(1) {
}
}
```

UART Pin Assignment

- Pin assignment of WINC1500 module UART are described in the following table.
- On ATWINC1500 Xplained Pro, TX and RX are available on through holes labeled "DEBUG_UART" for easy identification.

ATWINC1500 module pin name	ATWINC1500 Xplained Pro pin name	Function
J14	UART_TX	TXD
J19	UART_RXD	RXD

Ensure that the HostMCU is powered up and ATWINC1500 built-in UART is connected to PC via a serial to USB converter.

- In a Windows shell, run the command download_all.bat UART to start the download.
- During the download process, the batch script will output the firmware version being programmed onto the ATWINC1500 as well as the previously installed firmware version.
- The download_all.bat batch script shall be located in the src/firmware folder of the "WINCXXXX_FIRMWARE_UPDATE_PROJECT" triggers the download through built-in UART.

3.1.1.2 Serial Flash Download Using Custom HostMCU

The serial bridge example application for any supported HostMCU can be taken as a reference for implementing serial bridge for custom HostMCU.

- Integrate the serial bridge application in your development environment.
- Build the project.
- Copy the built elf binary file to the /src/firmware/Tools/serial_bridge path inside the "WINCXXXX_FIRMWARE_UPDATE_PROJECT" project.
- Create a copy of batch file samxxx_xplained_pro_firmware_update.bat and rename according to your HostMCU name.
- In the created batch file replace the samxxx_xplained_pro_serial_bridge.elf with your generated elf file.
- Run the newly created batch file.

4. General Information on Firmware Update

The "WINCXXXX_FIRMWARE_UPDATE_PROJECT" contains,

- ATWINC1500/ATWINC3400 Scripts
- ATWINC1500/ATWINC3400 Binary Tools

4.1 ATWINC1500/ATWINC3400 Scripts

The "src" folder inside "WINCXXXX_FIRMWARE_UPDATE_PROJECT" contains a list of batch (.bat) files which internally calls the following scripts.

- The samxxx_xplained_pro_firmware_update.bat script performs the following actions:
    - Program the Serial Bridge application
    - Prepare compound programmable binary image
    - Program the prepared binary image in to WINC device.

Execution flow of the scripts:

- The samxxx_xplained_pro_firmware_update.bat -> download_all_sb -> download_all -> update_pll_table

- The download_all_sb script performs the following actions:
    - Erase the HostMCU
    - Program the Serial bridge application
    - Reset the HostMCU
    - Find COM port where the device is connected

    - Calls download_all batch file
- The download_all script performs the following actions:
    - Calls update_pll_table to prepare the image
    - Program the ATWINC1500 device using winc_programmer_uart.exe utility
    - The winc_programmer_uart program the built firmware binary image to the WILC device.
- The update_pll_table script performs the following actions:
    - Calculating PLL lookup table using the xo offset from efuse at flash time.
    - Prepare compound binary image using image_tool utility
    - It uses the binary image_tool to buildd firmware binary image

An additional script “prepare_image” binary is available inside “src\firmware” directory which can performs the following actions:

- Prepare programmable image along with gain table using image_tool utility
- Prepare OTA image along with gain table using image_tool utility

The differences between prepare_image and update_pll_table are:

Prepare_image	Update_pll_table
Does not update PLL table based on xo offset	Updates PLL lookup table using the xo offset from efuse at flash time.
Prepares OTA image	Does not prepare OTA image
Includes Gain table to create binary image	Does not includes Gain table to create binary image
Not for production line	Created for production level purpose

4.2 ATWINC1500/ATWINC3400 Binary Tools

The script files internally use the following tools to build and program the image.

1. image_tool - Builds firmware binary image
2. winc_programmer_uart/i2c – Program the built firmware binary image to the WILC device using UART or I2C interface.

4.2.1 Building Firmware Image

- image_tool located in src/firmware/firmware is used to build binary images for ATWINC devices.
- It collects all the binaries for each section and combine it in to one firmware called m2m_image_XXXX.bin.
- The image_tool arrange the section according to the flash memory orgianization of ATWINC1500/ATWINC3400 as shown in the below diagram.
- The image_tool collects the information from the flash_image XML file.
- Refer flash_image.config XML for how the flash memory is divided.

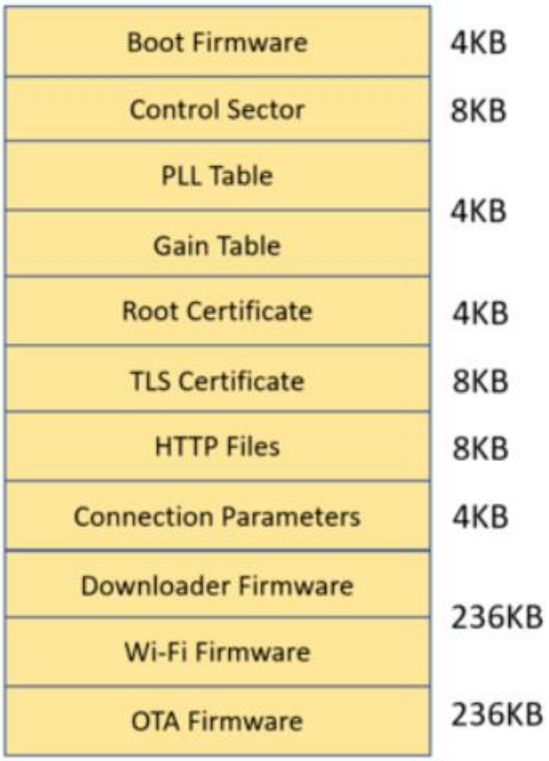
- User needs to update the flash_image.config for updating the root certificate or adding custom provisioning pages.

  [root certificates]
  type is root certificates
  schema is 1
  file is Tools/root_certificate_downloader/binary/BaltimoreCyberTrustRoot.cer
  file is Tools/root_certificate_downloader/binary/DigiCert.cer
  file is Tools/root_certificate_downloader/binary/DigiCertSHA2.cer
  file is Tools/root_certificate_downloader/binary/EnTrust.cer
  file is Tools/root_certificate_downloader/binary/GeoTrust.cer
  file is Tools/root_certificate_downloader/binary/GlobalSignRoot.cer
  file is Tools/root_certificate_downloader/binary/NMA_Root.cer
  file is Tools/root_certificate_downloader/binary/PROWL_Root.cer
  file is Tools/root_certificate_downloader/binary/QuoVadis_Root.cer
  file is Tools/root_certificate_downloader/binary/VeriSign.cer

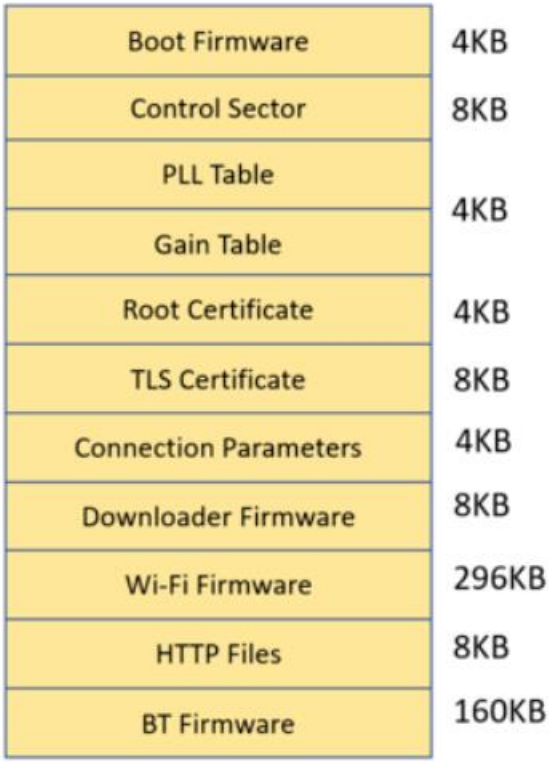
  [http files]
  type is http files
  schema is 2
  filename length is 32
  file is provisioning_webpage/default.html
  file is provisioning_webpage/style.css
  file is provisioning_webpage/favicon.ico
  file is provisioning_webpage/logo.png
  file is provisioning_webpage/error.json
  file is provisioning_webpage/scanresults.json
  file is provisioning_webpage/ok.json

- image_tool and the configuration XML file(flash_image.config) both can be found under “src\firmware\firmware” directory inside “WINCXXXX_FIRMWARE_UPDATE_PROJECT”

ATWINC1500 Flash Memory Segmentation:



ATWINC3400 Flash Memory Segmentation:



4.2.1.1 image_tool Usage

Usage	Command Example
To create firmware image	image_tool.exe -c flash_image.config -o m2m_image_3a0.bin -of prog
Writing to a specific region(eg: Root certificate)	image_tool.exe -c flash_image.config -o m2m_image_3a0.bin -of prog -r “root certificates”
To create OTA firmware image	image_tool.exe -c flash_image.config -c c Tools\gain_builder\gain_sheets\new_gain.config -o ota_firmware\m2m_ota_3A0.bin -of winc_ota -s ota

Arguments	Details
-c	Configuration files Microchip recommends to use the default configuration files which is flash_image.config
-o	Output name of the binary image files ATWINC1500: m2m_image_3a0.bin </br> ATWINC3400: m2m_image_3400.binit
-of	The image_tool supports 4 output formats 1. raw - Raw binary image. 2. winc_ota - WINC OTA format. 3. prog - Format suitable for programming. 4. log - Textual log information.
-r	Specifies a region to process. More than one region can be specified with repeated use of this option. If used only the regions specified will be processed.

For more information, enter image_tool help command:image_tool -h

Commands logs

Creating Firmware Image

Expected output log for the command:

image_tool.exe -c flash_image.config -o m2m_image_3a0.bin -of prog

```
Device Image Creation Tool 1.0.2 [r708] (Jul 28 2020)
Copyright (C) Microchip Technology Inc. 2020

processing region '[boot firmware]'
WINCFirmwareImageBuild: opening firmware file 'boot_firmware/release3A0/boot_firmware.bin'
written 1304 of 4096 bytes to image (32%)
processing region '[control sector]'
WINC1500ControlSectorBuild: creating control sector
written 64 of 8192 bytes to image (1%)
processing region '[pll table]'
Creating WiFi channel lookup table for PLL with xo_offset = 0.0000.
written 456 of 1024 bytes to image (45%)
processing region '[gain table]'
WINC1500GainBuildv2: creating gain tables
written 1600 of 3072 bytes to image (53%)
processing region '[root certificates]'
found certificate: Baltimore CyberTrust Root
found certificate: DigiCert High Assurance EV Root CA
found certificate: DigiCert SHA2 High Assurance Server CA
found certificate: Entrust Root Certification Authority
found certificate: GeoTrust Global CA
found certificate: GlobalSign Root CA
found certificate: AddTrust External CA Root
found certificate:
found certificate: QuoVadis Root CA 2
found certificate: VeriSign Class 3 Public Primary Certification Authority - G5
written 3188 of 4096 bytes to image (78%)
processing region '[tls certificates]'
written 0 of 8192 bytes to image (0%)
processing region '[http files]'
HTTPFileSystemAddFile: opening HTTP file 'provisioning_webpage/default.html'
HTTPFileSystemAddFile: opening HTTP file 'provisioning_webpage/style.css'
HTTPFileSystemAddFile: opening HTTP file 'provisioning_webpage/favicon.ico'
HTTPFileSystemAddFile: opening HTTP file 'provisioning_webpage/logo.png'
HTTPFileSystemAddFile: opening HTTP file 'provisioning_webpage/error.json'
HTTPFileSystemAddFile: opening HTTP file 'provisioning_webpage/scanresults.json'
HTTPFileSystemAddFile: opening HTTP file 'provisioning_webpage/ok.json'
written 7760 of 8192 bytes to image (95%)
processing region '[connection parameters]'
written 0 of 4096 bytes to image (0%)
processing region '[downloader firmware]'
WINCFirmwareImageBuild: opening firmware file 'downloader_firmware/release3A0/downloader_firmware.bin'
written 4628 of 241664 bytes to image (2%)
processing region '[wifi firmware]'
WINCFirmwareImageBuild: opening firmware file 'firmware/wifi_v111/ASIC_3A0/wifi_firmware.bin'
written 235578 of 237036 bytes to image (100%)
processing region '[ate firmware]'
WINCFirmwareImageBuild: opening firmware file 'ate_firmware/burst_tx_firmware_winc1500.bin'
written 76736 of 765952 bytes to image (11%)
```

Writing to a Specific Region

Expected output log for the command:

image_tool.exe -c flash_image.config -o m2m_image_3a0.bin -of prog -r “root certificates”

```
Device Image Creation Tool 1.0.2 [r708] (Jul 28 2020)
Copyright (C) Microchip Technology Inc. 2020

processing region '[boot firmware]'
ignoring region as not in specified region list
written 0 of 4096 bytes to image (0%)
processing region '[control sector]'
ignoring region as not in specified region list
written 0 of 8192 bytes to image (0%)
processing region '[pll table]'
ignoring region as not in specified region list
written 0 of 1024 bytes to image (0%)
processing region '[gain table]'
ignoring region as not in specified region list
written 0 of 3072 bytes to image (0%)
processing region '[root certificates]'
found certificate: Baltimore CyberTrust Root
found certificate: DigiCert High Assurance EV Root CA
found certificate: DigiCert SHA2 High Assurance Server CA
found certificate: Entrust Root Certification Authority
found certificate: GeoTrust Global CA
found certificate: GlobalSign Root CA
found certificate: AddTrust External CA Root
found certificate:
found certificate: QuoVadis Root CA 2
found certificate: VeriSign Class 3 Public Primary Certification Authority - G5
written 3188 of 4096 bytes to image (78%)
processing region '[tls certificates]'
ignoring region as not in specified region list
written 0 of 8192 bytes to image (0%)
processing region '[http files]'
ignoring region as not in specified region list
written 0 of 8192 bytes to image (0%)
processing region '[connection parameters]'
ignoring region as not in specified region list
written 0 of 4096 bytes to image (0%)
processing region '[downloader firmware]'
ignoring region as not in specified region list
written 0 of 241664 bytes to image (0%)
processing region '[wifi firmware]'
ignoring region as not in specified region list
written 0 of 241664 bytes to image (0%)
processing region '[ate firmware]'
ignoring region as not in specified region list
written 0 of 765952 bytes to image (0%)
```

Creating OTA Firmware Image

Expected output log for the command:

image_tool.exe -c flash_image.config -c c Tools\gain_builder\gain_sheets\new_gain.config -o ota_firmware\m2m_ota_3A0.bin -of winc_ota -s ota

```
Device Image Creation Tool 1.0.2 [r708] (Jul 28 2020)
Copyright (C) Microchip Technology Inc. 2020

processing region '[downloader firmware]'
WINCFirmwareImageBuild: opening firmware file 'downloader_firmware/release3A0/downloader_firmware.bin'
written 4628 of 241664 bytes to image (2%)
processing region '[wifi firmware]'
WINCFirmwareImageBuild: opening firmware file 'firmware/wifi_v111/ASIC_3A0/wifi_firmware.bin'
written 235578 of 237036 bytes to image (100%)
```

4.2.2 Programming Firmware Image

winc_programmer_uart/i2c tool located in src/firmware/firmware is used to program the binary images for ATWINC1500/ATWINC3400 devices using UART or I2C interface. it does the following primary jobs:

- Erase the ATWINC1500/ATWINC3400 memory
- Read the firmware from ATWINC1500/ATWINC3400
- Write the firmware to ATWINC1500/ATWINC3400
- Verify the written firmware.

4.2.2.1 winc_programmer_uart Tool Usage

Usage	Command Example
Erase ATWINC1500/ATWINC3400 flash memory	winc_programmer_uart.exe -p \.\COM16 -d winc1500 -e -pfw ..\programmer_firmware\release3A0\programmer_release_text.bin
Write the created binary image to ATWINC1500/ATWINC3400 flash memory	winc_programmer_uart.exe -p \.\COM16 -d winc1500 -i m2m_image_3A0.bin -if prog -w -pfw ..\programmer_firmware\release3A0\programmer_release_text.bin
Read back the written image from ATWINC1500/ATWINC3400 flash memory	winc_programmer_uart.exe -p \.\COM16 -d winc1500 -r -pfw ..\programmer_firmware\release3A0\programmer_release_text.bin
Verify the written image in ATWINC1500/ATWINC3400 device	winc_programmer_uart.exe -p \.\COM16 -d winc1500 -i m2m_image_3A0.bin -if prog -r -pfw ..\programmer_firmware\release3A0\programmer_release_text.bin
Single command (including all the above operations)	winc_programmer_UART.exe -p \.\COM16 -d winc1500 -e -i m2m_image_3A0.bin -if prog -w -r -pfw ..\programmer_firmware\release3A0\programmer_release_text.bin

Arguments	Details
-p	Port number of the connected HostMCU COM Port Command to find the Port number: test_edbg
-d	ATWINC device: winc1500 or winc3400

-e	To erase the ATWINC1500/ATWINC3400 device flash memory before writing the firmware image
-w	To write the firmware image
-r	To read the firmware image
-if	Input format. winc_ota - WINC OTA format. raw - A raw binary image. prog - Format suitable for programming.
-pfw	programming firmware : WINC firmware used to program the device.

For more information enter winc_programmer_uart help command:

winc_programmer_uart.exe -h

Commands logs

Erase WINC memory

Expected output log for the command:

winc_programmer_uart.exe -p .\COM16 -d winc1500 -e -pfw
..\programmer_firmware\release3A0\programmer_release_text.bin

```
WINC Programming Tool 1.0.3 [r708] (Jul 28 2020)
Copyright (C) Microchip Technology Inc. 2020

software WINC serial bridge found, baud rate changes supported
chip ID is 0x001503a0
programming firmware file: ..\programmer_firmware\release3A0\programmer_release_text.bin
starting device
reinitialise serial bridge to 500000
waiting for firmware to run
flash ID 0x001440ef
flash size is 8 Mb

begin erase operation

 0x000000:[eeeeeeee] 0x008000:[eeeeeeee] 0x010000:[eeeeeeee] 0x018000:[eeeeeeee]
 0x020000:[eeeeeeee] 0x028000:[eeeeeeee] 0x030000:[eeeeeeee] 0x038000:[eeeeeeee]
 0x040000:[eeeeeeee] 0x048000:[eeeeeeee] 0x050000:[eeeeeeee] 0x058000:[eeeeeeee]
 0x060000:[eeeeeeee] 0x068000:[eeeeeeee] 0x070000:[eeeeeeee] 0x078000:[eeeeeeee]
 0x080000:[eeeeeeee] 0x088000:[eeeeeeee] 0x090000:[eeeeeeee] 0x098000:[eeeeeeee]
 0x0a0000:[eeeeeeee] 0x0a8000:[eeeeeeee] 0x0b0000:[eeeeeeee] 0x0b8000:[eeeeeeee]
 0x0c0000:[eeeeeeee] 0x0c8000:[eeeeeeee] 0x0d0000:[eeeeeeee] 0x0d8000:[eeeeeeee]
 0x0e0000:[eeeeeeee] 0x0e8000:[eeeeeeee] 0x0f0000:[eeeeeeee] 0x0f8000:[eeeeeeee]
```

Write Firmware Image to WINC

Expected output log for the command:

winc_programmer_uart.exe -p .\COM16 -d winc1500 -i m2m_image_3A0.bin -if prog -w -pfw programmer_firmware\release3A0\programmer_release_text.bin

```
WINC Programming Tool 1.0.3 [r708] (Jul 28 2020)
Copyright (C) Microchip Technology Inc. 2020

software WINC serial bridge found, baud rate changes supported
chip ID is 0x001503a0
programming firmware file: ..\programmer_firmware\release3A0\programmer_release_text.bin
starting device
reinitialise serial bridge to 500000
waiting for firmware to run
flash ID 0x001440ef
flash size is 8 Mb

begin write operation

 0x000000:[ww.ww..w] 0x008000:[w.wwwww] 0x010000:[wwwwwwww] 0x018000:[wwwwwwww]
 0x020000:[wwwwwwww] 0x028000:[wwwwwwww] 0x030000:[wwwwwwww] 0x038000:[wwwwwwww]
 0x040000:[wwwwwwww] 0x048000:[wwwwwwww] 0x050000:[wwwwwwww] 0x058000:[........]
 0x060000:[........] 0x068000:[........] 0x070000:[........] 0x078000:[........]
 0x080000:[........] 0x088000:[........] 0x090000:[........] 0x098000:[........]
 0x0a0000:[........] 0x0a8000:[........] 0x0b0000:[........] 0x0b8000:[........]
 0x0c0000:[........] 0x0c8000:[........] 0x0d0000:[........] 0x0d8000:[........]
 0x0e0000:[........] 0x0e8000:[........] 0x0f0000:[........] 0x0f8000:[........]
```

Read Firmware Image from WINC memory

Expected output log for the Command:

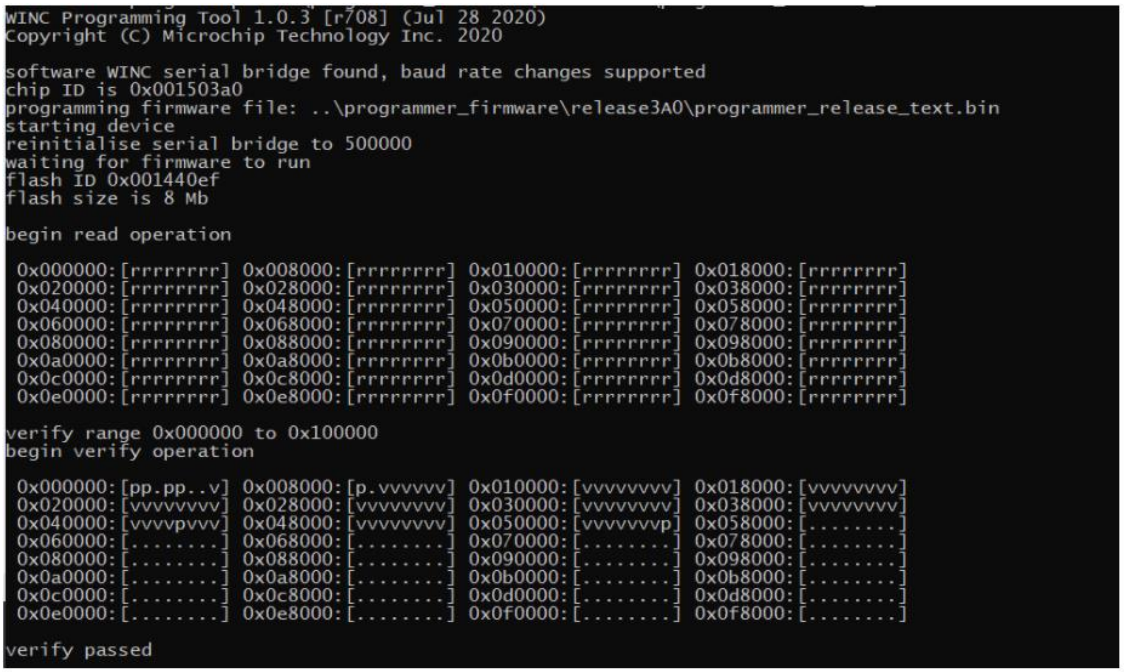
winc_programmer_uart.exe -p .\COM16 -d winc1500 -r -pfw ..\programmer_firmware\release3A0\programmer_release_text.bin

```
WINC Programming Tool 1.0.3 [r708] (Jul 28 2020)
Copyright (C) Microchip Technology Inc. 2020

software WINC serial bridge found, baud rate changes supported
chip ID is 0x001503a0
programming firmware file: ..\programmer_firmware\release3A0\programmer_release_text.bin
starting device
reinitialise serial bridge to 500000
waiting for firmware to run
flash ID 0x001440ef
flash size is 8 Mb

begin read operation

 0x000000:[rrrrrrrr] 0x008000:[rrrrrrrr] 0x010000:[rrrrrrrr] 0x018000:[rrrrrrrr]
 0x020000:[rrrrrrrr] 0x028000:[rrrrrrrr] 0x030000:[rrrrrrrr] 0x038000:[rrrrrrrr]
 0x040000:[rrrrrrrr] 0x048000:[rrrrrrrr] 0x050000:[rrrrrrrr] 0x058000:[rrrrrrrr]
 0x060000:[rrrrrrrr] 0x068000:[rrrrrrrr] 0x070000:[rrrrrrrr] 0x078000:[rrrrrrrr]
 0x080000:[rrrrrrrr] 0x088000:[rrrrrrrr] 0x090000:[rrrrrrrr] 0x098000:[rrrrrrrr]
 0x0a0000:[rrrrrrrr] 0x0a8000:[rrrrrrrr] 0x0b0000:[rrrrrrrr] 0x0b8000:[rrrrrrrr]
 0x0c0000:[rrrrrrrr] 0x0c8000:[rrrrrrrr] 0x0d0000:[rrrrrrrr] 0x0d8000:[rrrrrrrr]
 0x0e0000:[rrrrrrrr] 0x0e8000:[rrrrrrrr] 0x0f0000:[rrrrrrrr] 0x0f8000:[rrrrrrrr]
```

Verify the Written Image

Expected output log for the command:

winc_programmer_uart.exe -p .\COM16 -d winc1500 -i m2m_image_3A0.bin -if prog -r -pfw ..\programmer_firmware\release3A0\programmer_release_text.bin



Consolidated Single Command

This command does all the all the above operations in a single command.

Expected output log for the command:

winc_programmer_UART.exe -p .\COM16 -d winc1500 -e -i m2m_image_3A0.bin -if prog -w -r -pfw ..\programmer_firmware\release3A0\programmer_release_text.bin

```
WINC Programming Tool 1.0.3 [r708] (Jul 28 2020)
Copyright (C) Microchip Technology Inc. 2020

software WINC serial bridge found, baud rate changes supported
chip ID is 0x001503a0
programming firmware file: ..\programmer_firmware\release3A0\programmer_release_text.bin
starting device
reinitialise serial bridge to 500000
waiting for firmware to run
flash ID 0x001440ef
flash size is 8 Mb

begin erase operation

 0x000000:[eeeeeeee] 0x008000:[eeeeeeee] 0x010000:[eeeeeeee] 0x018000:[eeeeeeee]
 0x020000:[eeeeeeee] 0x028000:[eeeeeeee] 0x030000:[eeeeeeee] 0x038000:[eeeeeeee]
 0x040000:[eeeeeeee] 0x048000:[eeeeeeee] 0x050000:[eeeeeeee] 0x058000:[eeeeeeee]
 0x060000:[eeeeeeee] 0x068000:[eeeeeeee] 0x070000:[eeeeeeee] 0x078000:[eeeeeeee]
 0x080000:[eeeeeeee] 0x088000:[eeeeeeee] 0x090000:[eeeeeeee] 0x098000:[eeeeeeee]
 0x0a0000:[eeeeeeee] 0x0a8000:[eeeeeeee] 0x0b0000:[eeeeeeee] 0x0b8000:[eeeeeeee]
 0x0c0000:[eeeeeeee] 0x0c8000:[eeeeeeee] 0x0d0000:[eeeeeeee] 0x0d8000:[eeeeeeee]
 0x0e0000:[eeeeeeee] 0x0e8000:[eeeeeeee] 0x0f0000:[eeeeeeee] 0x0f8000:[eeeeeeee]

begin write operation

 0x000000:[ww.ww..w] 0x008000:[w.wwwwww] 0x010000:[wwwwwwww] 0x018000:[wwwwwwww]
 0x020000:[wwwwwwww] 0x028000:[wwwwwwww] 0x030000:[wwwwwwww] 0x038000:[wwwwwwww]
 0x040000:[wwwwwwww] 0x048000:[wwwwwwww] 0x050000:[wwwwwwww] 0x058000:[........]
 0x060000:[........] 0x068000:[........] 0x070000:[........] 0x078000:[........]
 0x080000:[........] 0x088000:[........] 0x090000:[........] 0x098000:[........]
 0x0a0000:[........] 0x0a8000:[........] 0x0b0000:[........] 0x0b8000:[........]
 0x0c0000:[........] 0x0c8000:[........] 0x0d0000:[........] 0x0d8000:[........]
 0x0e0000:[........] 0x0e8000:[........] 0x0f0000:[........] 0x0f8000:[........]

begin read operation

 0x000000:[rrrrrrrr] 0x008000:[rrrrrrrr] 0x010000:[rrrrrrrr] 0x018000:[rrrrrrrr]
 0x020000:[rrrrrrrr] 0x028000:[rrrrrrrr] 0x030000:[rrrrrrrr] 0x038000:[rrrrrrrr]
 0x040000:[rrrrrrrr] 0x048000:[rrrrrrrr] 0x050000:[rrrrrrrr] 0x058000:[rrrrrrrr]
 0x060000:[rrrrrrrr] 0x068000:[rrrrrrrr] 0x070000:[rrrrrrrr] 0x078000:[rrrrrrrr]
 0x080000:[rrrrrrrr] 0x088000:[rrrrrrrr] 0x090000:[rrrrrrrr] 0x098000:[rrrrrrrr]
 0x0a0000:[rrrrrrrr] 0x0a8000:[rrrrrrrr] 0x0b0000:[rrrrrrrr] 0x0b8000:[rrrrrrrr]
 0x0c0000:[rrrrrrrr] 0x0c8000:[rrrrrrrr] 0x0d0000:[rrrrrrrr] 0x0d8000:[rrrrrrrr]
 0x0e0000:[rrrrrrrr] 0x0e8000:[rrrrrrrr] 0x0f0000:[rrrrrrrr] 0x0f8000:[rrrrrrrr]

verify range 0x000000 to 0x100000
begin verify operation

 0x000000:[pp.pp..v] 0x008000:[p.vvvvvv] 0x010000:[vvvvvvvv] 0x018000:[vvvvvvvv]
 0x020000:[vvvvvvvv] 0x028000:[vvvvvvvv] 0x030000:[vvvvvvvv] 0x038000:[vvvvvvvv]
 0x040000:[vvvvpvvv] 0x048000:[vvvvvvvv] 0x050000:[vvvvvvvp] 0x058000:[........]
 0x060000:[........] 0x068000:[........] 0x070000:[........] 0x078000:[........]
 0x080000:[........] 0x088000:[........] 0x090000:[........] 0x098000:[........]
 0x0a0000:[........] 0x0a8000:[........] 0x0b0000:[........] 0x0b8000:[........]
 0x0c0000:[........] 0x0c8000:[........] 0x0d0000:[........] 0x0d8000:[........]
 0x0e0000:[........] 0x0e8000:[........] 0x0f0000:[........] 0x0f8000:[........]

verify passed
```

5. Download Failure Troubleshooting

Here are the troubleshooting tips for a specific error while downloading using batch script.

5.1 The script Failed To Find Any COM Port

The winc_programmer_uart.exe expects a COM port as an argument. If the expected COM port is not found, then it will provide the below error.

```
Cannot find and EDBG boards?
see  '"C:\Program Files (x86)\Atmel\Studio\7.0\atbackend\atprogram.exe" list'
```

How to fix it:

- Make sure ATWINC1500/ATWINC3400 COM port is listed in the device manager.
- Make sure ATWINC1500/ATWINC3400 COM port is not opened by any other application. For verification, try to open and close the COM port with a terminal application.
- low quality USB cable or low quality serial to USB converter (built-in UART) can introduce garbage on the UART line thus failing the detection of the ATWINC1500/ATWINC3400 COM port. Try a different cable.
- When performing a “built-in UART download”, it is expected that the ATWINC1500/ATWINC3400 bootloader is in a particular state that can only be achieved after doing a clean power up and reset sequence. Hence, before doing a download always ensure that a clean power up and reset sequence has been made.
- Make sure that no other extension board (ex: IO1…) is connected to the Xplained Pro board while performing the download.
- Make sure the project path is not exceeding Windows maximum 260 characters path length.

5.2 The script Found More Than One Matching Tool

The Found more than one matching tool error could be observed when downloading using Xplained Pro board serial bridge with sam_xplained_pro_firmware_update.bat batch script. The script will try to look for available COM ports and try to match each COM port name with “EDBG” string. This is to program the serial bridge binary image on the HostMCU. How to fix it:

- All the Xplained Pro boards are enumerated with “EDBG Virtual COM Port”. Make sure to connect one Xplained Pro board at a time on PC.

```
This batch file is unsuitable if more than one EDBG based development board is installed, found 2
Use download_all_sb.bat with options
        edbg
        ATSAMD21J18A
        Tools\serial_bridge\samd21_xplained_pro_serial_bridge.elf
        3400 or 3A0 [1500/1510]
        serial number of the dev board attached to the board you wish to program - see '"C:\Program Files (x86)\Atmel\Studio\7.0\atbackend\atprogram.exe" list'
        com port number assigned to the dev board attached to the board you wish to program by the OS
```

5.3 The script Listing More Than One COM Port

More than one COM port could be listed when downloading using download_all.bat where the HostMCU already has the serial bridge firmware or download through built-in UART. The winc_programmer_UART tool used to perform a serial bridge or a built-in UART download will try to look for available COM ports and try to match each COM port name with “EDBG” string or a port number “COM” string. If one of the two conditions is true, the program will then try to send a 0x12 char on each UART line. The other side is

then expected to answer 0x5A for a built-in UART update or 0x5B for a serial bridge update. If the expected response is received on all UART lines, the script will list all the detected COM ports.

How to fix it:

- Input COM port number of the intended device to be downloaded when “Please enter COM port number to program:” is displayed as shown in the above figures.
- Note that for each downloading option of ATWINC chip firmware, TLS/SSL root certificates, gain table the COM port number to be given. To avoid this, it is possible to force the winc_programmer_UART tool to use a specific COM port number in the beginning. For example to use COM56, run the script like this: “download_all.bat UART 56”

5.4 Failed To Initialize Programmer: Invalid Chip ID

The Failed to initialize programmer with Invalid chip ID error typically happens when there is garbage or noise on the UART line preventing from reading the correct chip ID value. How to fix it:

- Try connecting the PC and the ATWINC1500/ATWINC3400 with a different cable. A clean power up and reset sequence of the ATWINC1500/ATWINC3400 is necessary to start over with the ATWINC1500/ATWINC3400 bootloader in the appropriate state.

5.5 Failed To Initialize Programmer: Waiting For Chip Permission

After printing the correct chip ID of the ATWINC1500/ATWINC3400, the winc_programmer_UART tool programs a small binary (programmer firmware) to assist with ATWINC1500/ATWINC3400 flash programming. At this stage the winc_programmer_UART will change the UART baud rate from 115200 to 500000 to speed up the actual transfer of the firmware image. Once the baud rate change is made, the chip permission is verified to ensure the UART connection is reliable. Failing at this stage means that the current setup does not support such a high baud rate. How to fix it:

- It is recommended to try connecting the PC and the ATWINC1500/ATWINC3400 with a different cable. Also a clean power up and reset sequence of the ATWINC1500/ATWINC3400 is necessary to start over with the ATWINC1500/ATWINC3400 bootloader in the appropriate state.