Realtek Ameba 1 Firmware Image

This document introduces firmware image output and how to update image over the air.

Table of Contents

1 Introduction

Over-the-air programming (OTA) provides a methodology of updating device firmware remotely via TCP/IP network connection.

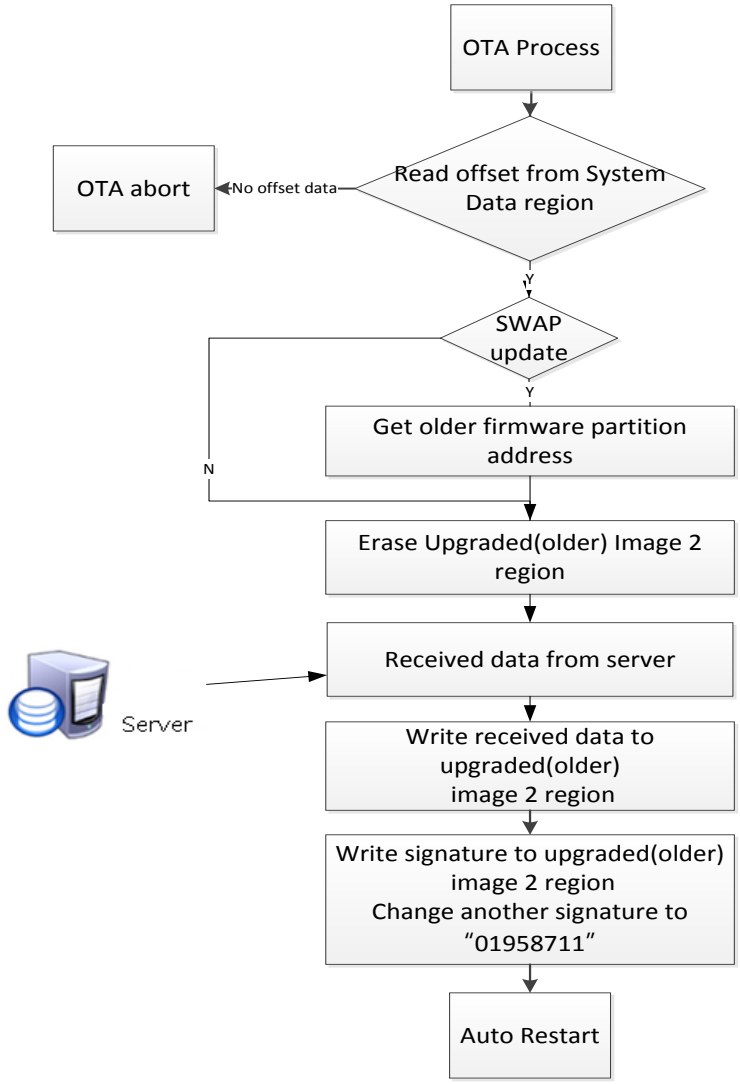
2 Firmware Image Output

After building project source files, there are 2 files will be generated automatically. The first is *ram_all.bin* that is containing boot loader and application image. And the second is *ota.bin* that is application only image. Those two images can be found at *SDK_folder/project/project_name/EWARM-RELEASE/Debug/Exe.*

3 Firmware Update Over the Air

3.1 Overview

3.1.1 OTA operation flow



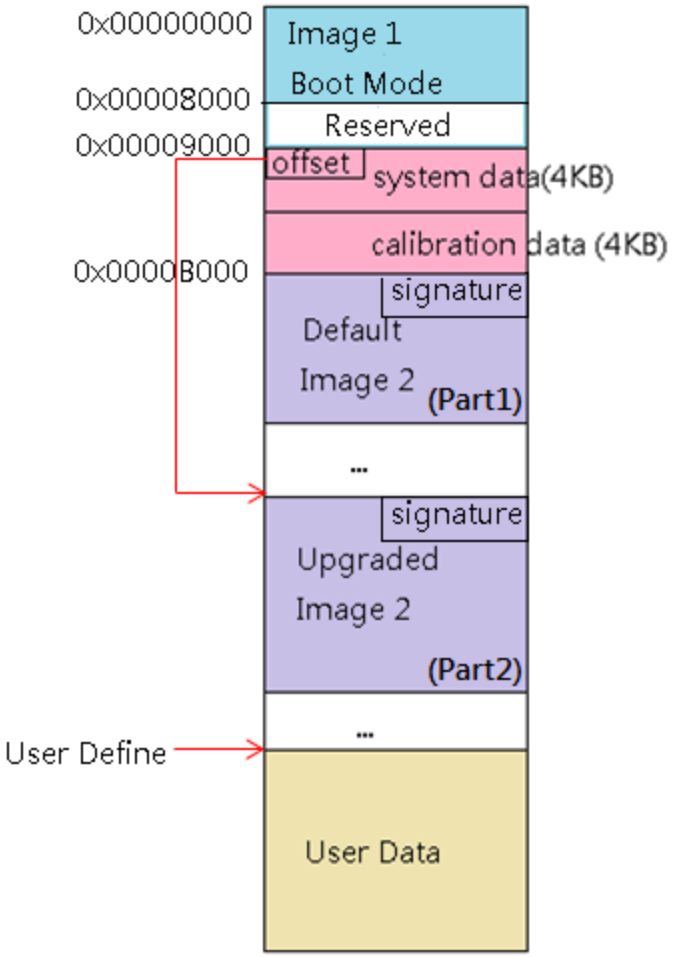
Note: During the step of "Erase Upgraded (Older) Image2 region", the signature is set to 0xffffffff, which is invalid signature.

Note: OTA updater will change signature of another region from "81958711" to "01958711" when writing signature. Firmware with signature "01958711" is previous version, and with "81958711" is latest updated version.

3.1.2 Boot process flow

Boot loader will select latest (signature == “81958711”) updated image2 and load it to SRAM.

3.1.3 Upgraded Partition



In most case, we suggest only updating Upgraded Image 2.

Default Image2 can be updated if set SWAP_UPDATE in ota_8195a.h. Please note that using SWAP update methods will delete factory default firmware.

NOTE: Signature “81958711” is mean latest updated version. “01958711” is previous version. Boot loader will load latest version by default.

3.2 Implement OTA over Wi-Fi

3.2.1 OTA using local download server base on socket

The example shows how device updates image from a local download server. The local download server send image to device based on network socket.

Make sure both device and PC are connecting to the same local network.

3.2.1.1 Build OTA Application image

Turn on OTA command

The flag defined in \project\realtek_ameba1_va0_example\inc\platform_opts.h

```
//Config in platform_opts.h
#define CONFIG_OTA_UPDATE      1
```

Define SWAP_UDPATE in ota_8195a.h file

```
//Config in ota_8195a.h
#define SWAP_UPDATE 1
```

Enable this will update OTA image to Default (Part1) Image2 region. This behavior may damage another region data, please use this at your own risk.

Write the address of the upgraded image 2 to system data.

Use the following sample code to write the upgraded image 2 address to system data flash section.

```
Sample code:
#include "flash_api.h"
#define WRITE_OTA_ADDR 1

flash_t flash;
//address:0x00080000
Uint32_t ota_addr = 0x00080000;
//boundary check
if((ota_addr > IMAGE_3) && ((ota_addr < (IMAGE_3+Img3Len))) ||
        (ota_addr < IMAGE_3) || ((ota_addr & 0xfff) != 0)|| (ota_addr == ~0x0)){
                printf("\n\r[%s] illegal ota addr 0x%x", __FUNCTION__, ota_addr);
                goto update_ota_exit;
         }else
           write_ota_addr_to_system_data( &flash, ota_addr);
```

Read upgraded image 2 address from system data and verify this address

```
//Config in ota_8195a.c
uint32_t update_ota_prepare_addr(void)
{
   …
   //Get upgraded image 2 addr from offset
   device_mutex_lock(RT_DEV_LOCK_FLASH);
   flash_read_word(&flash, OFFSET_DATA, &NewImg2Addr);
   device_mutex_unlock(RT_DEV_LOCK_FLASH);
   if((NewImg2Addr > IMAGE_3) && ((NewImg2Addr < (IMAGE_3+Img3Len))) ||
      (NewImg2Addr < IMAGE_3) ||((NewImg2Addr & 0xfff) != 0)|| (NewImg2Addr == ~0x0)){
         printf("\n\r[%s] Invalid OTA Address 0x%x", __FUNCTION__, NewImg2Addr);
         return -1;
   }

                    goto update_ota_exit;
```

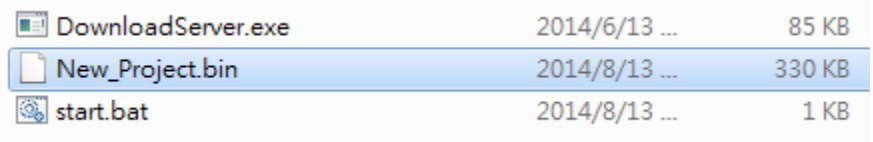
The address of OFFSET_DATA is 0x9000, and the address of upgraded image 2 is the first 4 byte from this address. If the address was not qualified, then the OTA process will be stopped.

Define custom signature

```
//Configuration in ota_8195a.h
1. turn on the marco as follows:
#define CONFIG_CUSTOM_SIGNATURE 1
2. Define your own signature.
//Define in ota_8195a.c

#if CONFIG_CUSTOM_SIGNATURE
/* -------------------------------------------------
 *  Customized Signature
 * -------------------------------------------------*/
// This signature can be used to verify the correctness of the image
// It will be located in fixed location in application image
#pragma location=".custom.validate.rodata"
const unsigned char cus_sig[32] = "Customer Signature-modelxxx";
#endif
3. Compare it while complete flashing.
int update_ota_checksum(_file_checksum *file_checksum, uint32_t flash_checksum, uint32_t
NewImg2Addr)
{
…
#if CONFIG_CUSTOM_SIGNATURE
        && !strcmp(read_custom_sig,custom_sig)
#endif
…
}
```

3.2.1.2 Local download server

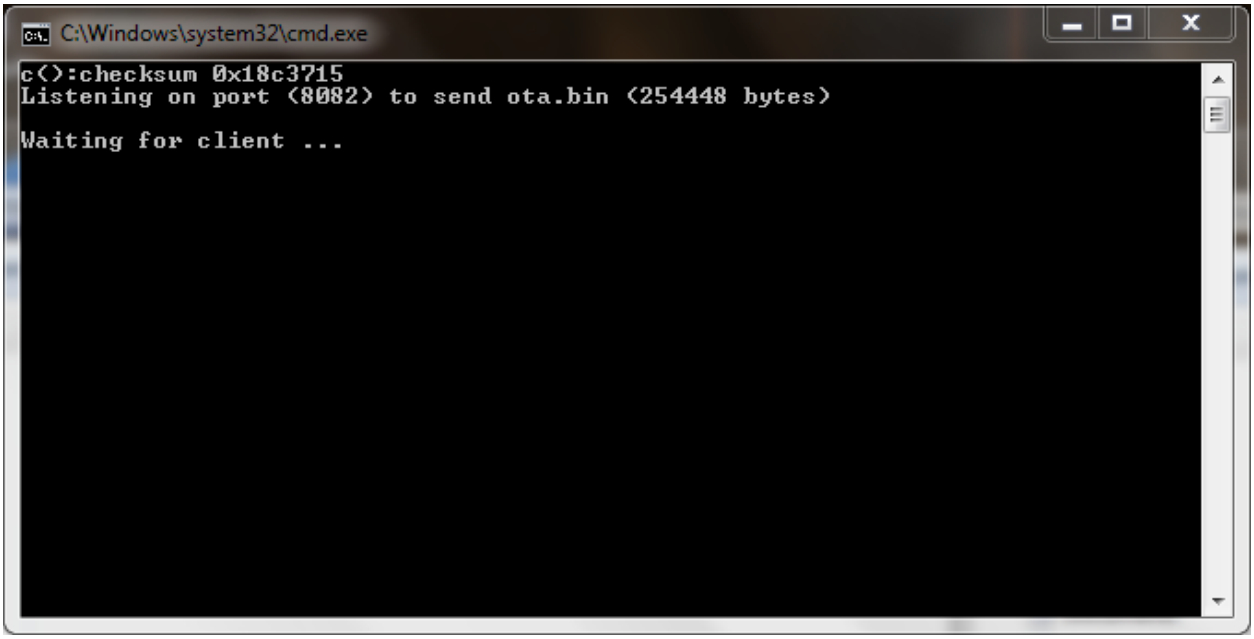
Build new image New_Project.bin in DownloadServer folder (path: tools\DownloadServer\).



Edit start.bat file. Port = 8082, file = New_Project.bin

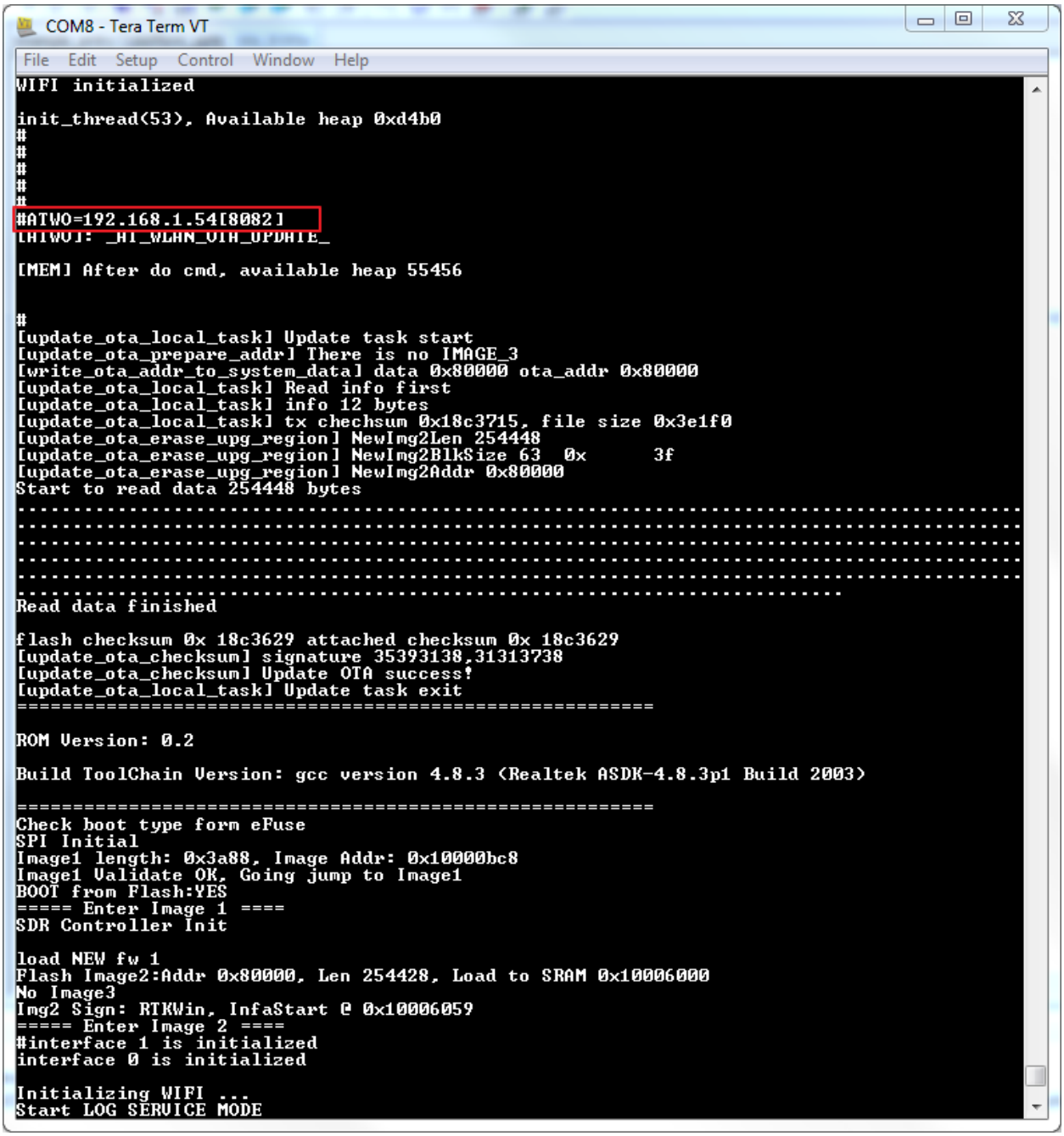
```
1  @echo off
2  DownloadServer 8082 New_Project.bin
3  set /p DUMMY=Press Enter to Continue ...
```

Execute start.bat

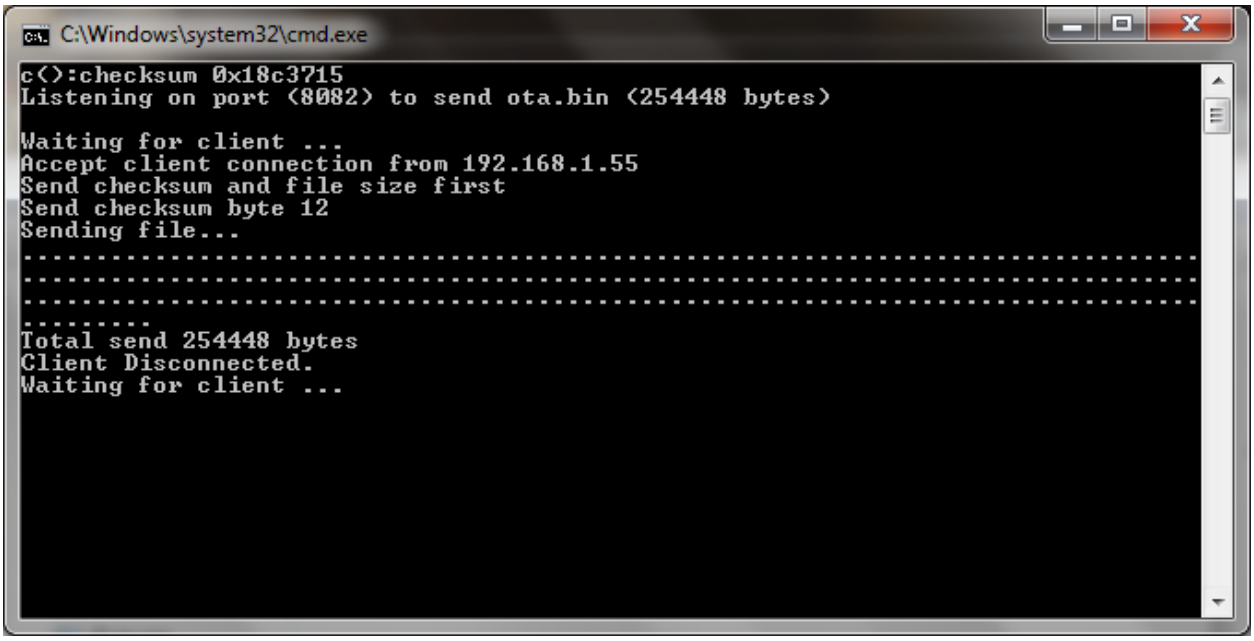


Reboot device and connect to AP.

Enter command: ATWO=IP[PORT].



Local download server success message:



After finishing downloading image, device will be auto-rebooted, and the bootloader will load new image 2 if it exist.

3.2.2 OTA using local download server base on HTTP

This example shows how device updates image from a local http download server. The local http download server will send the http response which data part is ota.bin after receiving the http request.

Make sure both device and PC are connecting to the same local network.

3.2.2.1 Build OTA Application image

Turn on OTA HTTP example

The example flag defined in \project\realtek_ameba1_va0_example\inc\platform_opts.h

The http ota flag defined in \component\soc\realtek\8195a\misc\platform\ota_8195a.h

```
//Config in platform_opts.h
#define CONFIG_OTA_UPDATE      1
//Defined in ota_8195a.h
#define HTTP_OTA_UPDATE
```

Define Server IP and PORT in example_ota.c file (In \component\common\example\ota_update\example_ota.c)

```
//Defined in example_ota.c
#define PORT            8082
#define IP              "192.168.1.54"
#define RESOURCE        "ota.bin"
```

Example: SERVER: http://m-apps.oss-cn-shenzhen.aliyuncs.com/051103061600.bin

Setting: #define PORT 80

#define HOST "m-apps.oss-cn-shenzhen.aliyuncs.com"

#define RESOURCE "051103061600.bin"

Define SWAP_UDPATE in ota_8195a.h file

```
//Config in ota_8195a.h
#define SWAP_UPDATE 1
```

Enable this will update OTA image to Default (Part1) Image2 region. This behavior may damage another region data, please use this at your own risk.

Write the address of the upgraded image 2 to system data.

Use the following sample code to write the upgraded image 2 address to system data flash section.

```
Sample code:
#include "flash_api.h"
#define WRITE_OTA_ADDR 1

flash_t flash;
//address:0x00080000
Uint32_t ota_addr = 0x00080000;
//boundary check
if((ota_addr > IMAGE_3) && ((ota_addr < (IMAGE_3+Img3Len))) ||
        (ota_addr < IMAGE_3) || ((ota_addr & 0xfff) != 0)|| (ota_addr == ~0x0)){
                printf("\n\r[%s] illegal ota addr 0x%x", __FUNCTION__, ota_addr);
                goto update_ota_exit;
         }else
           write_ota_addr_to_system_data( &flash, ota_addr);
```

Read upgraded image 2 address from system data and verify this address

```
//Config in ota_8195a.c
uint32_t update_ota_prepare_addr(void)
{
   …
   //Get upgraded image 2 addr from offset
   device_mutex_lock(RT_DEV_LOCK_FLASH);
   flash_read_word(&flash, OFFSET_DATA, &NewImg2Addr);
   device_mutex_unlock(RT_DEV_LOCK_FLASH);
   if((NewImg2Addr > IMAGE_3) && ((NewImg2Addr < (IMAGE_3+Img3Len))) ||
      (NewImg2Addr < IMAGE_3) ||((NewImg2Addr & 0xfff) != 0)|| (NewImg2Addr == ~0x0)){
         printf("\n\r[%s] Invalid OTA Address 0x%x", __FUNCTION__, NewImg2Addr);
         return -1;
   }
```

The address of OFFSET_DATA is 0x9000, and the address of upgraded image 2 is the first 4 byte from this address. If the address was not qualified, then the OTA process will be stopped.

Define custom signature

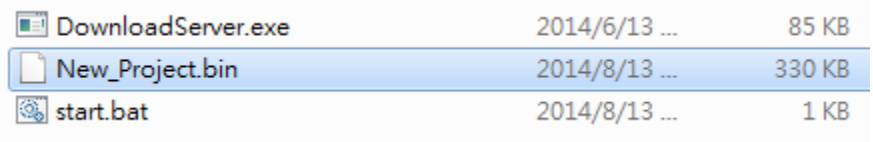
```
//Configuration in ota_8195a.h
1. turn on the marco as follows:
#define CONFIG_CUSTOM_SIGNATURE 1
2. Define your own signature.
//Define in ota_8195a.c

#if CONFIG_CUSTOM_SIGNATURE
/* -------------------------------------------------
 *  Customized Signature
 * -------------------------------------------------*/
// This signature can be used to verify the correctness of the image
// It will be located in fixed location in application image
#pragma location=".custom.validate.rodata"
const unsigned char cus_sig[32] = "Customer Signature-modelxxx";
#endif
3. Compare it while complete flashing.
int update_ota_checksum(_file_checksum *file_checksum, uint32_t flash_checksum, uint32_t
NewImg2Addr)
{
…
#if CONFIG_CUSTOM_SIGNATURE
        && !strcmp(read_custom_sig,custom_sig)
#endif
…
}
```

3.2.2.2 Communication with Local HTTP download server

1. In http_update_ota_local_task(), after connecting with server, Ameba will send a HTTP request to server : "GET /RESOURCE HTTP/1.1\r\nHost: host\r\n\r\n"
2. The local HTTP download server will send the HTTP response after receiving the request. The response header contains the "Content-Length" which is the length of the ota.bin The response data part is just ota.bin
3. After ameba receiving the HTTP response, it will parse the http response header to get the content length to judge if the receiving ota.bin is completed.

3.2.2.3 Local HTTP download server

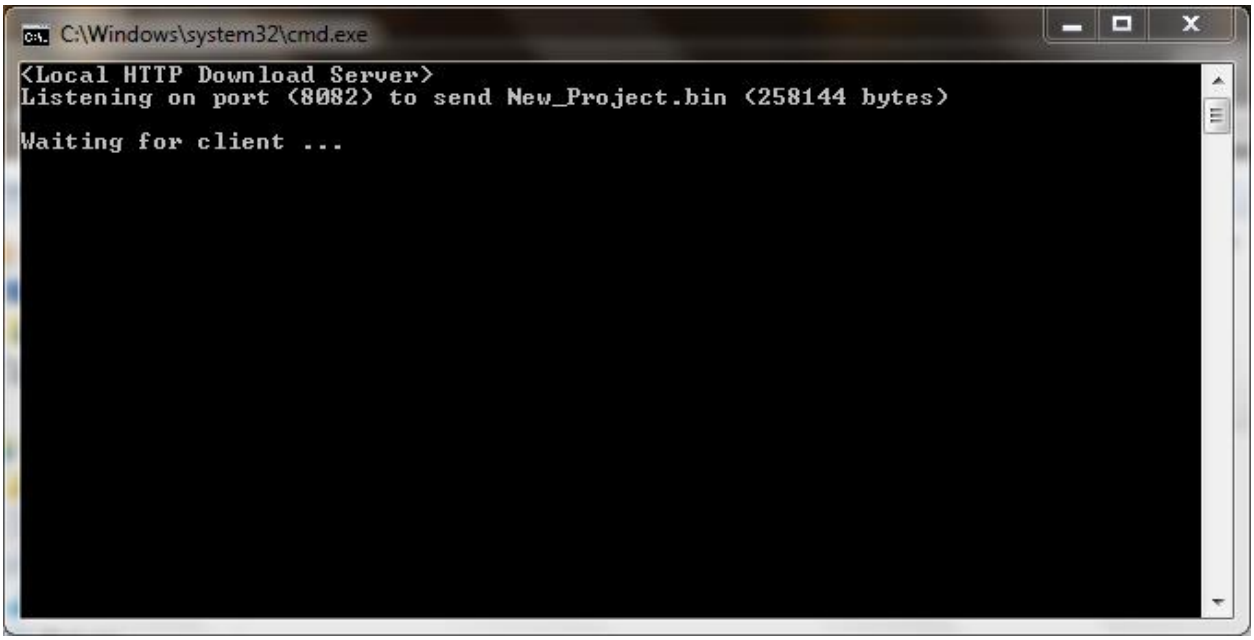
Build new image New_Project.bin in DownloadServer(HTTP) folder.



Edit start.bat file. Port = 8082, file = New_Project.bin

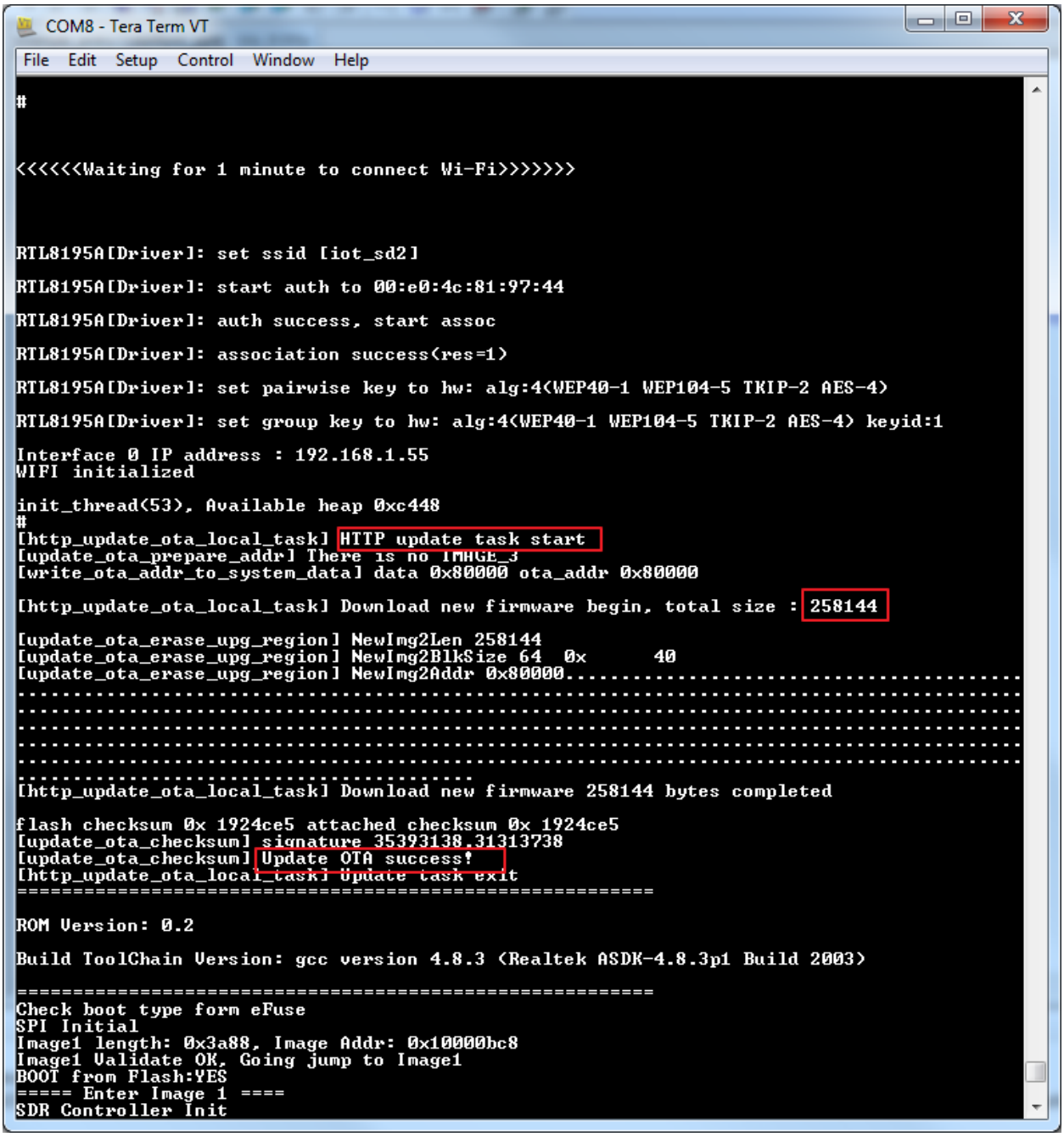
```
1  @echo off
2  DownloadServer 8082 New_Project.bin
3  set /p DUMMY=Press Enter to Continue ...
```

Execute start.bat

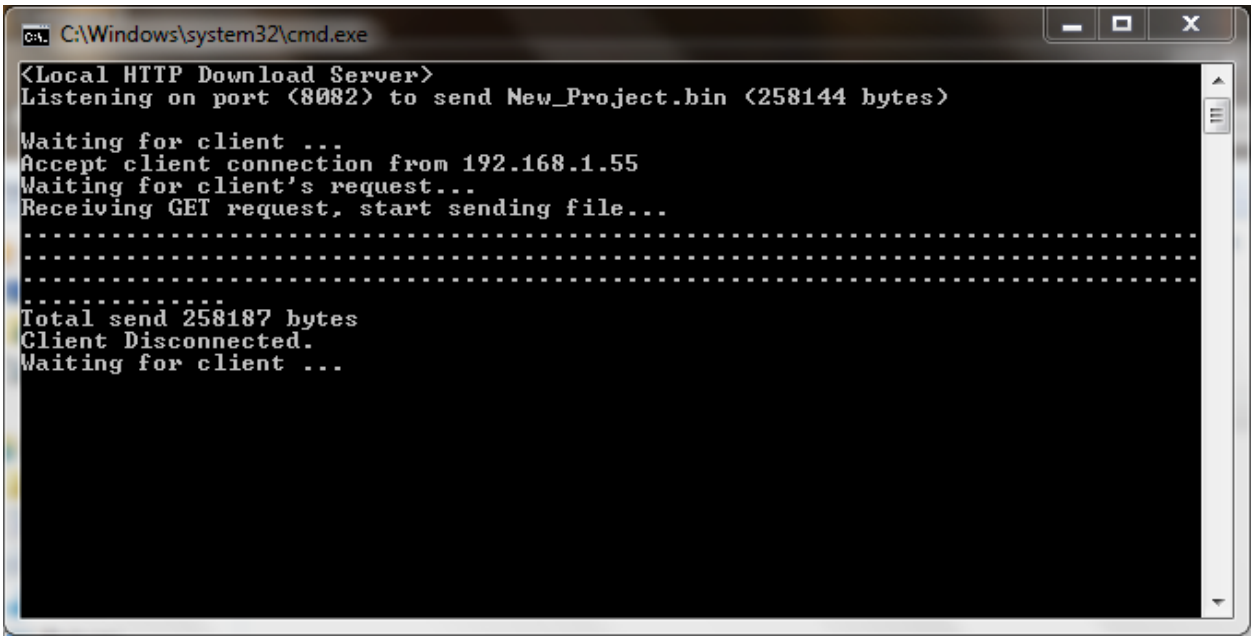


Reboot device and connect to AP.

After 1 minute, the OTA update through HTTP protocol will start.



Local download server success message:



After finishing downloading image, device will be auto-rebooted, and the bootloader will load the new image 2 if it exists.

3.3 OTA Signature

To Clear or Recover OTA signature for verification via UART at command, please refer to AN0025.