Talaria TWO^{TM} (INP2045)

Low Power Multi-Protocol Wireless Platform SoC

IEEE 802.11 b/g/n, BLE 5.0

Application Note

Firmware Over-The-Air Upgrade

Release: 08-13-2021

InnoPhase, Inc.

6815 Flanders Drive

San Diego, CA 92121

innophaseinc.com

Revision History

Version Number	Date	Comments
1.0	04-29-2021	First release
1.1	06-29-2021	Procedure for flashing components updated
2.0	08-13-2021	Updated for SDK 2.3 release

Contents

1 Figures

2 Terms & Definitions

BLE	Bluetooth Low Energy
EVB	Evaluation Board
FOTA	Firmware-Over-the-Air
SDK	Software Development Kit
SSBL	Second Stage Boot Loader
TLS	Transport Layer Security

3 Introduction

Firmware-Over-the-Air (FOTA) allows for wireless delivery of firmware updates and/or configurations to embedded devices.

This document describes the FOTA process for the Talaria TWO EVB using the Talaria TWO SDK with details on how to implement or trigger FOTA in a customer provided application.

4 Overview

This implementation of FOTA provides the following facilities:

1. Check for the availability of new upgrades.
2. Securely download the image into flash.
3. Check the validity of the downloaded image.
4. Set the new image as the boot image.

In conjunction with SSBL, it enables booting the latest image downloaded. The firmware is downloaded into the application image partition in the Flash.

5 Features & Limitations

Following are the FOTA application features:

1. FOTA over HTTP/HTTPS.
2. Image download from Cloud or any HTTP/web server.
3. Two copy solution. Backup copy of the correct firmware always exists.
4. Image integrity check using sha256 hash.
5. Error handling and recovery
    a. If any error occurs during downloading the image or updating the configuration files (`part.json/boot.json/fota_config.json`), the device will remain in the current image.
    b. If a reboot occurs (due to issues like power failure) during image download or configuration files upgrade, the device will boot with the current image.
6. JSON based configuration

Limitations: Upgrading the Certificates is not supported as of now.

6 Dependent Module Information

This section provides a list of modules in Talaria TWO on which FOTA is dependent. It is important to understand these concepts before proceeding with the design aspects of the FOTA.

6.1 Flash Layout

About Talaria TWO Flash:

1. Size: 2MB
2. 512 sectors
3. 4096 bytes/sector
4. 256-byte page

Flash is divided into eight partitions. Partition table information is stored in the `Boot sector`. Each partition has a starting sector and a sector count, along with a type, and some control bits. No two partitions overlap. The reason for using sector addressing is so that partitions can be independently erased.

Figure 1 provides the proposed layout of Flash memory when using SSBL. To use SSBL, Flash must at least contain SSBL, filesystem, and one application.

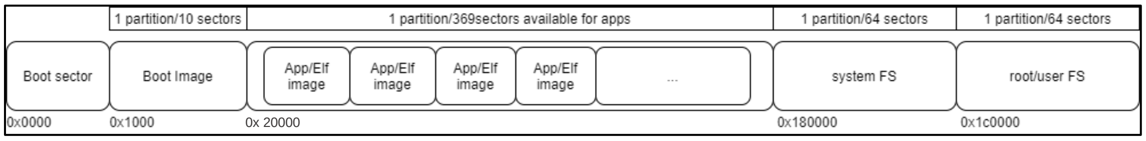


Figure 1: Flash layout when using the SSBL

The `Boot Image` is the default application that Talaria TWO's boot ROM would look for when a Talaria TWO device is powered ON. To support FOTA, SSBL shall run as `Boot image`. SSBL is a special application that determines the final application to load. In a nutshell, on power cycle, the boot ROM boots the SSBL application which in turn loads the final application

For detailed documentation on Flash layout, refer:
`sdk_x.y/apps/ssbl/doc/Application_for_using_SSBL.pdf`

Note: x.y refer to the SDK release version.

6.2 Partition Table File (part.json)

This is a json file that provides the partition information of the application images in the Flash. The file is stored in root/user FS. This file mainly contains an array of image information (represented by the name **image:**).

Each of the image information entry in the array gives image name, version, starting sector and other information about the application. Following is the basic content:

```
{
  "image"   : [
    {
      "name"  : "fota",
      "version" : "1.0",
      "start_sector"  : 32,
      "bootargs_start": 1,
      "ssid" : "inno_test",
      "passphrase" : "1234567890",
      "bootargs_end" : 1
    },
    {
      "name"  : "test_app",
      "version" : "1.0",
      "start_sector"  : 154,
      "bootargs_start": 1,
      "ssid" : "inno_test",
      "passphrase" : "1234567890",
      "bootargs_end" : 1
    },
    {
      "name"  : "test_app",
      "version" : "0.0",
      "start_sector"  : 230,

      "bootargs_start": 1,

      "ssid" : "inno_test",

      "passphrase" : "1234567890",

      "bootargs_end" : 1

    }

    ],

    "baudrate"    : 2560000,

    "timeout" : 0,

    "verbose" : 1

}
```

6.3 Boot Index File (boot.json)

This is a json file stored in root/user FS. It contains the image index. This is the index in the image information array present in `part.json` file. SSBL gets the index of the image to be loaded from this file.

Following is the content:

```
{
    image : 0
}
```

6.4 FOTA Configuration File (fota_config.json)

The FOTA configuration file `fota_config.json` is a json file. This file is stored in the root/user FS in Flash. The FOTA module gets all the information required to download the Firmware or a file.

Each object in this file shall give information about the file to be downloaded. Each object will have the following tokens:

1. type: Type of the file. It can be `firmware` or `file`
2. name: Name of the firmware image/ file
3. server_domain_name: Fully Qualified domain name of the server
4. server_ip: If the Domain name is not provided, server IP will be considered.
5. Port: Server port
6. Uri: This is the location of the firmware/file in the cloud
7. Secured: Value for this token will be 1 if the connection is secure, else 0
8. root_cert: Certificate file name
9. hash: Hash used for checking the integrity of the firmware/file

Following is the basic content of the file:

```
{
  "package_version"    : "1.0",
  "files"   : [
     {
        "type" : "configuration",
        "name" : "fota.config",
        "protocol" : "http",
        "hostname" : "innotestota.s3.us-east-2.amazonaws.com",
        "port" : 443,
        "secured" : 1,
        "uri" : "/fota_config.json"
    },
    {
        "type" : "firmware",
        "name" : "test_app",
        "version" : "2.1",
        "protocol" : "http",
        "hostname" : "innotestota.s3.us-east-2.amazonaws.com",
        "port" : 443,
        "secured" : 1,
        "uri" : "/test_app.elf"
    }
    ]
}
```

The group of Firmware and files, the information of which are present in this file is considered as a package. Each `fota_config.json` file will have a package version at the top. The array of objects will provide information about firmware and files considered as one package.

The `package_version` provides the version of the package. There will be a fota_config.json file in the Cloud. If the `package_version` of the `fota_config.json` file is greater than that of the file currently present in the device, FOTA needs to be done.

The first object shall give the information about the `fota_config.json` file available on Cloud. Device can fetch the file and see if a package with a higher version is available. The Firmware will be downloaded in the application partition and files will be stored in root/user FS.

6.5 Secure Secondary Boot Loader (SSBL)

SSBL is an application that facilitates booting a specific image from the flash. On boot, the boot-ROM loads & starts SSBL. SSBL reads the image index from the `boot.json` file. It parses the `part.json` file and picks the image info in the image info array at the index read from `boot.json` file. The SSBL then loads and runs the image at the sector provided by this image information.

For detailed information about the SSBL design, refer: `sdk_x.y/apps/ssbl/doc/Application_for_using_SSBL.pdf`.

7 Design

FOTA process involves the following components:

1. Parsing the FOTA configuration file
2. Checking for the new updates
3. Selecting image area
4. Secured connection
5. Downloading the Firmware
6. Error handling

7.1 Checking for New Updates

For checking for new updates, module fetches the `fota_config.json` file from the cloud. The package version of the downloaded file is compared against the `fota_config.json` file already present in the device. If the version is higher, FOTA needs to be done.

This functionality is optional, and the step can be skipped if an external application like Mobile Application does the check and provisions the device to trigger the FOTA. The functionality is provided through API for the applications to be used for polling.

7.2 Selecting Image Area

This logic will parse the `part.json` file and selects the image area in flash for downloading the image.

Each application that can be upgraded using FOTA will have a unique name in the image information table. Multiple image information entries for the same application will have the same name. That is, each such application will have at-least two slots in the table.

For example, if there is an application called `app_image`, there will be two entries in the image information table with the same name. There will be a minimum of two entries for an application which can be upgraded using FOTA.

The version field in the image information shall represent the FOTA version and not the application release version. The selection logic will go through all the entries for a given application and selects area (image information) with least version number.

For example, if one entry for `app_image` has version `1` and its starting sector is 66 and other entry for the same application has the version `0` and its starting sector is 166, the first entry will be selected for FOTA image download. The new image will be downloaded at sector 66.

Each time after FOTA succeeds, the version number for the selected image information is changed to one more than the highest currently available version, so that the newer version will always have the highest version number.

7.3 Secured Connection

The `fota_config.json` file provides the following information for connection and download:

1. Server IP/ DNS
2. Port number
3. Firmware location on the server (URI)
4. Root CA certificate to authenticate the server at the time of SSL connection

If the DNS name is provided, DNS will be resolved. The root CA certificate as indicated in the `fota_config.json` file will be present in the root/user FS. HTTPS connection will be established with the server. The connection will be secured using Transport Layer Security (TLS1.2).

7.4 Downloading the Firmware

Once the HTTPS connection is successfully established, the image is downloaded using HTTP GET. The URI of the Firmware as provided in the `fota_config.json` file is used during the GET. The image is downloaded into flash at the location selected as detailed in 7.2.

After successful download, image is authenticated using the certificate indicated by `auth_cert` field in `fota_config.json` file. This will also ensure that the integrity of the image is intact. This certificate will be present in the root/user FS.

7.5 Setting the new image for boot and reload

If the image integrity of the downloaded image is found to be intact, version number of the selected image information in `part.json` file will be increased by one more than the highest version currently in use. Finally, image index in `boot.json` file will be updated with the index of the selected image information and the device is reset. After reboot, SSBL will automatically load the newly downloaded image.

7.6 Error handling

The FOTA alternates the image download between two application image area in flash. At any point of time there will at-least one proper application image (currently running). This acts as a backup/fallback image in case FOTA fails. The boot image index in `boot.json` file is changed to point to the new image only at the last step of FOTA after the integrity of the downloaded image is found to be intact.

At any point of time if the error occurs, the procedures can be retried. The procedure will be retried for `FOTA_MAX_RETRIES` multiple times before giving up. If FOTA is not successful, the currently available stable image will run.

7.7 Flow Diagram

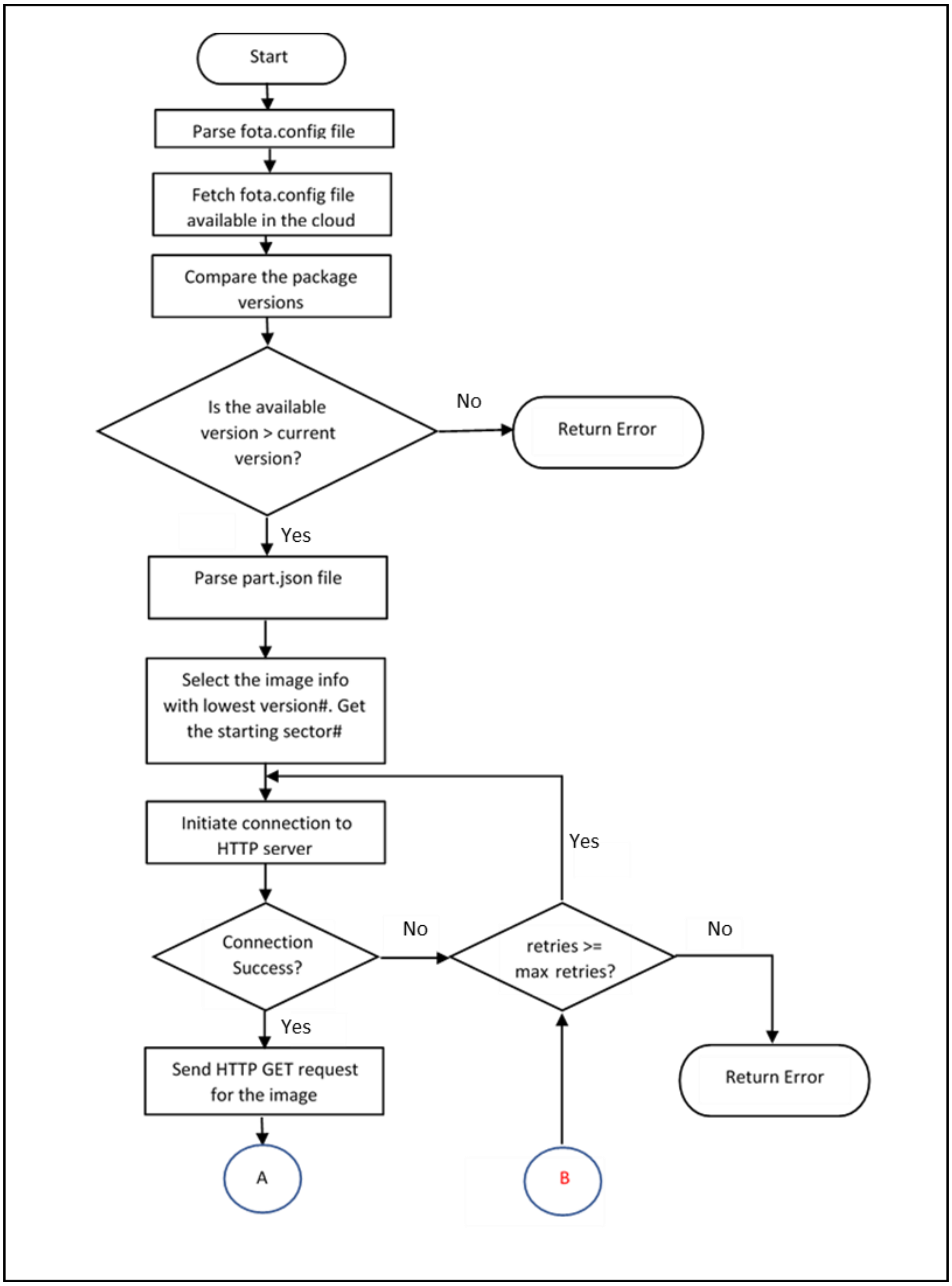

Start
Parse fota.config file
Fetch fota.config file available in the cloud
Compare the package versions
Is the available version > current version?
No
Return Error
Yes
Parse part.json file
Select the image info with lowest version#. Get the starting sector#
Initiate connection to HTTP server
Yes
Connection Success?
No
retries >= max retries?
No
Yes
Send HTTP GET request for the image
Return Error
A
B

Figure 2: Flow Diagram

Continued from the previous flow diagram:

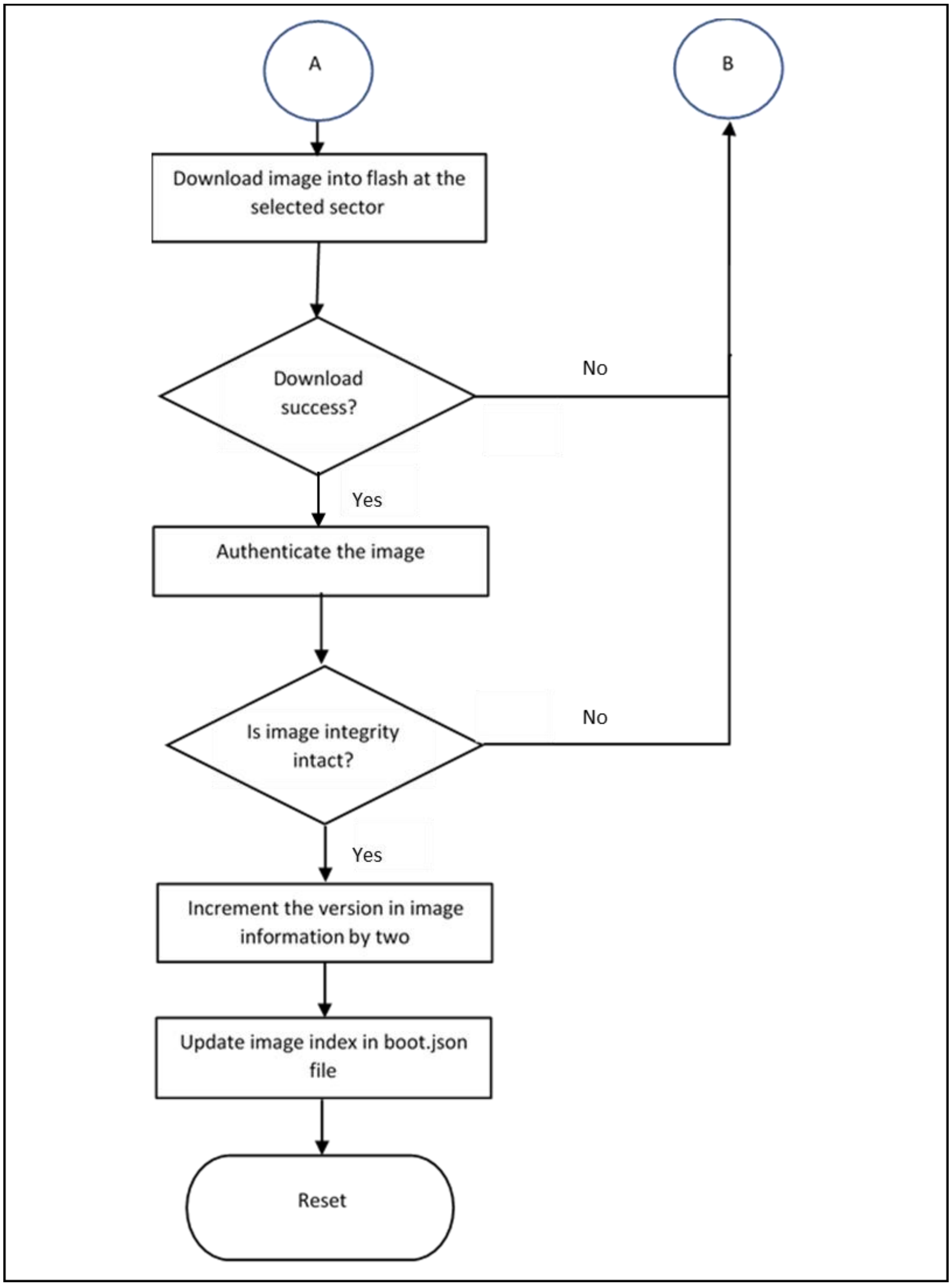

A
B
Download image into flash at the selected sector
Download success?
No
Yes
Authenticate the image
Is image integrity intact?
No
Yes
Increment the version in image information by two
Update image index in boot.json file
Reset

Figure 3: Flow Diagram - continued

8 Build SSBL & FOTA Application

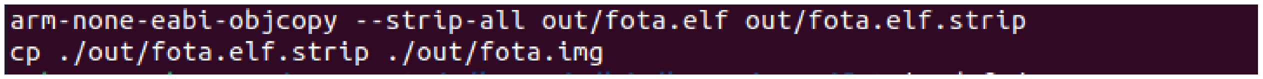
The following commands are run with the Talaria TWO SDK as the current working directory.

8.1 Build fota.img file

```
sdk/apps/fota$

make
```

Expected output:



```
arm-none-eabi-objcopy --strip-all out/fota.elf out/fota.elf.strip
cp ./out/fota.elf.strip ./out/fota.img
```

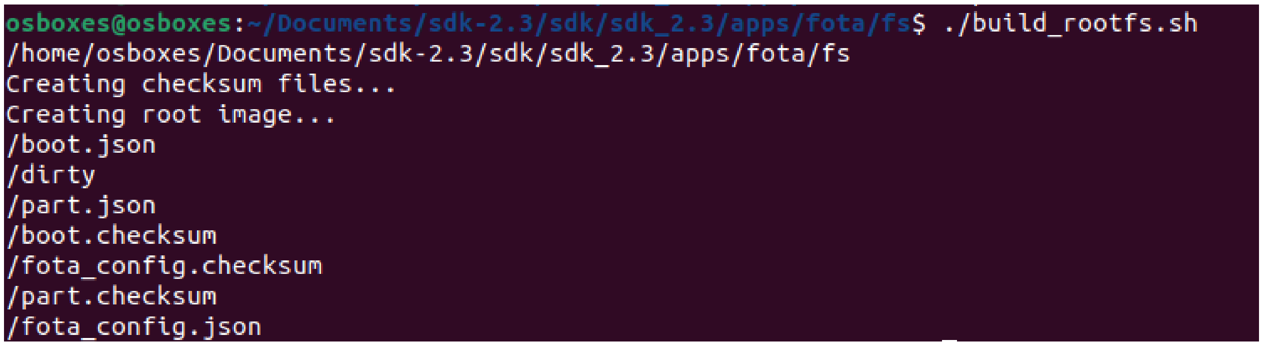
Figure 4: Build fota.img file - output

8.2 Create File System (root.img) file

```
sdk/apps/fota/fs$

./build_rootfs.sh
```

Expected output:



```
osboxes@osboxes:~/Documents/sdk-2.3/sdk/sdk_2.3/apps/fota/fs$ ./build_rootfs.sh
/home/osboxes/Documents/sdk-2.3/sdk/sdk_2.3/apps/fota/fs
Creating checksum files...
Creating root image...
/boot.json
/dirty
/part.json
/boot.checksum
/fota_config.checksum
/part.checksum
/fota_config.json
```

Figure 5: Create File System (root.img) file – output

The `sdk/apps/fota/fs` has the reference `part.json` file and `fota_config.json`. Customer specific changes need to be done here before building the root fs.

9 Flash SSBL & FOTA

This can be done in two ways:

1. Use `script.sh` present in the `sdk/apps/fota`
   OR
2. Execute the following instructions to flash the different components into Talaria TWO EVB under the SDK directory

Load Flash Helper

```
./script/boot.py --device /dev/ttyUSB2 --reset=evk42_bl ./apps/gordon/bin/gordon.elf
```

Invalidate the boot Image

```
dd if=/dev/zero of=./empty.img bs=1K count=1

./script/flash.py --device /dev/ttyUSB2 write 0x1000 ./empty.img
```

Write Partition

```
./script/flash.py --device /dev/ttyUSB2 from_json ./tools/partition_files/ssbl_part_table.json
```

Download root fs image

```
./script/flash.py --device /dev/ttyUSB2 write 0x1C0000 ./apps/fota/out/root.img
```

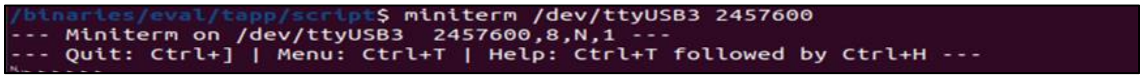
Download SSBL

```
./script/flash.py --device /dev/ttyUSB2 write 0x1000 ./apps/ssbl/fast_ssbl.img
```

Download fota.img

```
./script/flash.py --device /dev/ttyUSB2 write 0x20000 ./apps/fota/out/fota.img
```

Open a miniterm at baud rate of 2457600 and reset the EVB:



```
/binaries/eval/tapp/script$ miniterm /dev/ttyUSB3 2457600
--- Miniterm on /dev/ttyUSB3  2457600,8,N,1 ---
--- Quit: Ctrl+] | Menu: Ctrl+T | Help: Ctrl+T followed by Ctrl+H ---
```

Figure 6: Miniterm console output

Reset the board either by giving the following command or by pressing the reset button on the EVB:

```
./script/boot.py --device /dev/ttyUSB2 --reset=evk42
```

10 Expected Output

On successful execution of the steps in section 9, reset the Talaria TWO EVB. The following observation is made:

1. Talaria TWO loads SSBL
2. SSBL loads FOTA test application
3. FOTA test application modifies files in the filesystem to trigger FOTA, then reboots
4. Talaria TWO reboots and loads SSBL, SSBL loads the FOTA application
5. FOTA application downloads and flashes application from server and reboots
6. Talaria TWO loads SSBL, SSBL loads the downloaded application

Console output:

```
Y-BOOT 208ef13 2019-07-22 12:26:54 -0500 790da1-b-7

ROM yoda-h0-rom-16-0-gd5a8e586

FLASH:PWAE

WWWWWAE[0.018,733] heapsize is less than requested 29952 < 30000

Build $Id: git-f92bee540 $

krn.heapsize=30000

$App:git-2c93b72

SDK Ver: SDK_2.3


SSBL No Secureboot krg



**Checking Sanity of : /root/fota_config.json **



Integrity Check : /root/fota_config.json

checksum (calculated) = a72e03cc

checksum (stored)     = a72e03cc

Integrity Check : /root/fota_config_backup.json
```

```
Error: Failed opening /root/fota_config_backup.json


/root/fota_config.json : Master copy is intact. Backup copy Not present

/root/fota_config_backup.json : Is is not intact or take_backup = 1

utils_file_clone: /root/fota_config.json to /root/fota_config_backup.json

utils_file_clone: /root/fota_config.checksum to

/root/fota_config_backukp.checksum


**Checking Sanity of : /root/part.json **



Integrity Check : /root/part.json

checksum (calculated) = 37238cd7

checksum (stored)     = 37238cd7

Integrity Check : /root/part_backup.json

Error: Failed opening /root/part_backup.json


/root/part.json : Master copy is intact. Backup copy Not present

/root/part_backup.json : Is is not intact or take_backup = 1

utils_file_clone: /root/part.json to /root/part_backup.json

utils_file_clone: /root/part.checksum to /root/part_backup.checksum


**Checking Sanity of : /root/boot.json **



Integrity Check : /root/boot.json

checksum (calculated) = ae8acfbb

checksum (stored)     = ae8acfbb
```

```
Integrity Check : /root/boot_backup.json

Error: Failed opening /root/boot_backup.json


/root/boot.json : Master copy is intact. Backup copy Not present

/root/boot_backup.json : Is is not intact or take_backup = 1

utils_file_clone: /root/boot.json to /root/boot_backup.json

utils_file_clone: /root/boot.checksum to /root/boot_backup.checksum

Sanity check OK.. removed dirty bit file

Note: Max size of json token = 80

Key= image

Key= name

Val(str)= fota

Key= version

Val(str)= 1.0

Key= start_sector

Val(NUM)= 32

Key= bootargs_start

bootargs_start

Val(NUM)= 1

Key= ssid

Val(str)= innotest

Key= passphrase

Val(str)= innophase123

Key= bootargs_end

bootargs_end

Val(NUM)= 1

Key= name

Val(str)= test_app
```

```
Key= version

Val(str)= 1.0

Key= start_sector

Val(NUM)= 154

Key= bootargs_start

bootargs_start

Val(NUM)= 1

Key= ssid

Val(str)= innotest

Key= passphrase

Val(str)= innophase123

Key= bootargs_end

bootargs_end

Val(NUM)= 1

Key= name

Val(str)= test_app

Key= version

Val(str)= 0.0

Key= start_sector

Val(NUM)= 230

Key= bootargs_start

bootargs_start

Val(NUM)= 1

Key= ssid

Val(str)= innotest

Key= passphrase

Val(str)= innophase123

Key= bootargs_end
```

```
bootargs_end

Val(NUM)= 1

Key= baudrate

Val(NUM)= 2560000

Key= timeout

Val(NUM)= 0

Key= verbose

Val(NUM)= 1

key =  image

num val : 0


Boot indx = 0

ssbl_load_elf : Read elf @ 20000

Elf OK.

Reading section names @ offset = 72380


.text

.data

.bss

.virt

.hwreg.hw_ver

.hwreg.hw_pmu

.hwreg.hw_wfe

.hwreg.hw_evh

.hwreg.hw_timer

.hwreg.hw_boff

.hwreg.hw_gpio

.hwreg.hw_rnd
```

```
.hwreg.hw_qspi

.hwreg.hw_uart

.hwreg.hw_freq

.hwreg.hw_wdg

.hwreg.hw_tb

.hwreg.hw_sup

.hwreg.hw_trxdma

.hwreg.hw_sspi

.hwreg.hw_sdio

.hwreg.hw_cipher

.hwreg.hw_clone

.hwreg.hw_clone_bt

.hwreg.hw_clone_sdio

.hwreg.hw_i2c

.hwreg.hw_rxtdc

.hwreg.hw_pwm

.hwreg.hw_mmu

.hwreg.hw_afe

.hwreg.hw_core

.hwreg.hw_pdm

.hwreg.hw_rstclk

.hwreg.hw_bbrx_ofdm

.hwreg.hw_macif

.hwreg.hw_frame

.hwreg.hw_txbb

.hwreg.hw_ble

.hwreg.hw_bbrx_11b

.hwreg.hw_frontend_rx
```

```
.hwreg.hw_frontend

.hwreg.hw_fpga

.hwreg.hw_tx_dummy

.hwreg.hw_tap

.hwreg.hw_dpll

.ARM.attributes

.comment

.note.innophase

.shstrtab

Elf Load OK...

vm_flash location: 262400

Starting image...

Boot-args:

      vm.flash_location=0x00040100

      passphrase=innophase123

      ssid=innotest


Resetting...

=====================================================




Build $Id: git-f92bee540 $

vm.flash_location=0x00040100 passphrase=innophase123 ssid=innotest




Application Information:

------------------------

Name       : FOTA application
```

```
Version    : 1.0

Build Date : Aug  5 2021

Build Time : 13:25:43

Heap Available: 286 KB (293272 Bytes)

addr e0:69:3a:00:01:3d

Connecting to ...innotest - innophase123[1.316,383] CONNECT:e4:c3:2a:31:0c:ae

Channel:3 rssi:-16 dBm

[3.312,221] MYIP 192.168.0.14

[3.312,496] IPv6 [fe80::e269:3aff:fe00:13d]-link


N/w Connection done..

fota_json_init: /root/fota_config.json  f = 0x000b4f88

Parsing rootfs FOTA config file***

package_version = 1.0

Package version = 1.0

type = configuration

name = fota.config

Error: key not found

protocol = http

hostname = innotestota.s3.us-east-2.amazonaws.com

port = 443

secured = 1

uri = /fota_config.json

Error: key not found

Error: key not found

configuration

      fota.config

      http
      innotestota.s3.us-east-2.amazonaws.com

      443

      1

      /fota_config.json

      <null>

      <null>

type = firmware

name = test_app

version = 2.1

protocol = http

hostname = innotestota.s3.us-east-2.amazonaws.com

port = 443

secured = 1

uri = /test_app.elf

Error: key not found

Error: key not found

firmware

      test_app

      http

      innotestota.s3.us-east-2.amazonaws.com

      443

      1

      /test_app.elf

      <null>

      2.1

 Fota Init Success: b4f28

fota_http_connect:host=innotestota.s3.us-east-2.amazonaws.com port=443

Calling http_client_open()  . Checking input configurations...
```

```
  . Seeding the random number generator...

  . Connecting to tcp innotestota.s3.us-east-2.amazonaws.com:443...

  . Setting up the SSL/TLS structure...

        >setting configurations..

        >auth mode = 0 (0- skip, 1- optional, 2- required

        >max fragment len = 0

  . Performing the SSL/TLS handshake...

 ok

  . Verifying peer X.509 certificate...

Parsing Remote FOTA config file***

package_version = 2.1

Package version = 2.1

type = configuration

name = fota.config

Error: key not found

protocol = http

hostname = innotestota.s3.us-east-2.amazonaws.com

port = 443

secured = 0

uri = /fota_config.json

Error: key not found

Error: key not found

configuration

      fota.config

      http

      innotestota.s3.us-east-2.amazonaws.com

      443

      0
      /fota_config.json

      <null>

      <null>

type = firmware

name = test_app

version = 2.1

protocol = http

hostname = innotestota.s3.us-east-2.amazonaws.com

port = 443

secured = 1

uri = /test_app.elf

Error: key not found

Error: key not found

firmware

      test_app

      http

      innotestota.s3.us-east-2.amazonaws.com

      443

      1

      /test_app.elf

      <null>

      2.1

utils_num_str_cmp

 2

 1

 1

 0

deci1 = 2, fracn1 = 1, deci2 = 1, fracn2 = 0
```

```
Perform Fota
fota_debug_print_file_info_list:
configuration
      fota.config
firmware
      test_app
 type = configuration
 type = firmware
fota_json_init: /root/part.json  f = 0x000b2fb0
Image array size = 3
name = fota
name = test_app
version = 1.0
start_sector = 154
1.0 :154
name = test_app
version = 0.0
start_sector = 230
0.0 :230
utils_num_str_cmp
 1
 0
 0
 0
deci1 = 1, fracn1 = 0, deci2 = 0, fracn2 = 0


image_info_l->index = 2
```

```
Download the new f/w @ sector = 230


fota_http_connect:host=innotestota.s3.us-east-2.amazonaws.com port=443

Calling http_client_open()  . Checking input configurations...

  . Seeding the random number generator...

  . Connecting to tcp innotestota.s3.us-east-2.amazonaws.com:443...

  . Setting up the SSL/TLS structure...

        >setting configurations..

        >auth mode = 0 (0- skip, 1- optional, 2- required

        >max fragment len = 0

  . Performing the SSL/TLS handshake...

 ok

  . Verifying peer X.509 certificate....

      fota_http_cb: resp->resp_len = 0, resp->resp_total_len = 235928

total_rcvd_len= 0

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 1460

.

      fota_http_cb: resp->resp_len = 219, resp->resp_total_len = 235928

total_rcvd_len= 1679

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 3139

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 4599

.
```

```
      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 6059

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 7519

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 8979

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 10439

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 11899

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 13359

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 14819

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 16279

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 17739

.
```

```
      fota_http_cb: resp->resp_len = 324, resp->resp_total_len = 235928

total_rcvd_len= 18063

.

      fota_http_cb: resp->resp_len = 1024, resp->resp_total_len = 235928

total_rcvd_len= 19087

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 20547

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 22007

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 23467

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 24927

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 26387

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 27847

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 29307

.
```

```
      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 30767

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 32227

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 33687

.

      fota_http_cb: resp->resp_len = 760, resp->resp_total_len = 235928

total_rcvd_len= 34447

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 35907

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 37367

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 38827

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 40287

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 41747

.
```

```
      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 43207

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 44667

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 46127

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 47587

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 49047

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 50507

.

      fota_http_cb: resp->resp_len = 324, resp->resp_total_len = 235928

total_rcvd_len= 50831

.

      fota_http_cb: resp->resp_len = 1024, resp->resp_total_len = 235928

total_rcvd_len= 51855

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 53315

.
```

```
      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 54775

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 56235

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 57695

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 59155

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 60615

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 62075

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 63535

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 64995

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 66455

.
```

```
      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 67915

.

      fota_http_cb: resp->resp_len = 324, resp->resp_total_len = 235928

total_rcvd_len= 68239

.

      fota_http_cb: resp->resp_len = 1024, resp->resp_total_len = 235928

total_rcvd_len= 69263

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 70723

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 72183

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 73643

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 75103

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 76563

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 78023

.
```

```
      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 79483

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 80943

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 82403

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 83863

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 85323

.

      fota_http_cb: resp->resp_len = 324, resp->resp_total_len = 235928

total_rcvd_len= 85647

.

      fota_http_cb: resp->resp_len = 1024, resp->resp_total_len = 235928

total_rcvd_len= 86671

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 88131

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 89591

.
```

```
      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 91051

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 92511

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 93971

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 95431

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 96891

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 98351

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 99811

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 101271

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 102731

.
```

```
      fota_http_cb: resp->resp_len = 324, resp->resp_total_len = 235928

total_rcvd_len= 103055

.

      fota_http_cb: resp->resp_len = 1024, resp->resp_total_len = 235928

total_rcvd_len= 104079

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 105539

.


      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 106999

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 108459

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 109919

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 111379

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 112839

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 114299

.
```

```
      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 115759

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 117219

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 118679

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 120139

.

      fota_http_cb: resp->resp_len = 324, resp->resp_total_len = 235928

total_rcvd_len= 120463

.

      fota_http_cb: resp->resp_len = 1024, resp->resp_total_len = 235928

total_rcvd_len= 121487

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 122947

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 124407

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 125867

.
```

```
      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 127327

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 128787

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 130247

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 131707

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 133167

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 134627

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 136087

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 137547

.

      fota_http_cb: resp->resp_len = 324, resp->resp_total_len = 235928

total_rcvd_len= 137871

.
```

```
      fota_http_cb: resp->resp_len = 1024, resp->resp_total_len = 235928

total_rcvd_len= 138895

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 140355

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 141815

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 143275

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 144735

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 146195

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 147655

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 149115

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 150575

.
```

```
      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 152035

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 153495

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 154955

.

      fota_http_cb: resp->resp_len = 324, resp->resp_total_len = 235928

total_rcvd_len= 155279

.

      fota_http_cb: resp->resp_len = 1024, resp->resp_total_len = 235928

total_rcvd_len= 156303

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 157763

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 159223

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 160683

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 162143

.
```

```
      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928
total_rcvd_len= 163603
.
      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928
total_rcvd_len= 165063
.
      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928
total_rcvd_len= 166523
.
      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928
total_rcvd_len= 167983
.
      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928
total_rcvd_len= 169443
.
      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928
total_rcvd_len= 170903
.
      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928
total_rcvd_len= 172363
.
      fota_http_cb: resp->resp_len = 324, resp->resp_total_len = 235928
total_rcvd_len= 172687
.
      fota_http_cb: resp->resp_len = 1024, resp->resp_total_len = 235928
total_rcvd_len= 173711
.
```

```
      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 175171

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 176631

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 178091

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 179551

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 181011

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 182471

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 183931

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 185391

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 186851

.
```

```
      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 188311

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 189771

.

      fota_http_cb: resp->resp_len = 324, resp->resp_total_len = 235928

total_rcvd_len= 190095

.

      fota_http_cb: resp->resp_len = 1024, resp->resp_total_len = 235928

total_rcvd_len= 191119

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 192579

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 194039

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 195499

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 196959

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 198419

.
```

```
      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 199879

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 201339

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 202799

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 204259

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 205719

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 207179

.

      fota_http_cb: resp->resp_len = 324, resp->resp_total_len = 235928

total_rcvd_len= 207503

.

      fota_http_cb: resp->resp_len = 1024, resp->resp_total_len = 235928

total_rcvd_len= 208527

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 209987

.
```

```
      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 211447

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 212907

.



      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 214367

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 215827

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 217287

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 218747

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 220207

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 221667

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 223127

.
```

```
      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 224587

.

      fota_http_cb: resp->resp_len = 324, resp->resp_total_len = 235928

total_rcvd_len= 224911

.

      fota_http_cb: resp->resp_len = 1024, resp->resp_total_len = 235928

total_rcvd_len= 225935

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 227395

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 228855

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 230315

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 231775

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 233235

.

      fota_http_cb: resp->resp_len = 1460, resp->resp_total_len = 235928

total_rcvd_len= 234695

.
```

```
      fota_http_cb: resp->resp_len = 1233, resp->resp_total_len = 235928

total_rcvd_len= 235928


sector_cache_flush_all

Fw download complete

image hash: bac472d642128151c7373a4dc59a1dace6b3acd16781c1a29790e375700965f

!!! New version = 2.1, next index = 2

fota_commit

utils_create_checksum_file : /root/fota_config.json: /root/fota_config.checksum

new checksum = 038ae603

utils_create_checksum_file : /root/part.json: /root/part.checksum

new checksum = 51e8b58c

fota_json_init: /root/boot.json  f = 0x000b3020

Setting next boot index = 2

utils_create_checksum_file : /root/boot.json: /root/boot.checksum

new checksum = c37563ff

Y-BOOT 208ef13 2019-07-22 12:26:54 -0500 790da1-b-7

ROM yoda-h0-rom-16-0-gd5a8e586

FLASH:PWAE

WWWWWAE[0.018,742] heapsize is less than requested 29952 < 30000

Build $Id: git-f92bee540 $

krn.heapsize=30000

$App:git-2c93b72

SDK Ver: SDK_2.3


SSBL No Secureboot krg
```

```
**Checking Sanity of : /root/fota_config.json **



Integrity Check : /root/fota_config.json
checksum (calculated) = 038ae603
checksum (stored)     = 038ae603
Integrity Check : /root/fota_config_backup.json
checksum (calculated) = a72e03cc
checksum (stored)     = a72e03cc
/root/fota_config.json : Master copy is intact. Backup copy Not present
/root/fota_config_backup.json : Is is not intact or take_backup = 1
utils_file_clone: /root/fota_config.json to /root/fota_config_backup.json
utils_file_clone: /root/fota_config.checksum to
/root/fota_config_backukp.checksum

**Checking Sanity of : /root/part.json **



Integrity Check : /root/part.json
checksum (calculated) = 51e8b58c
checksum (stored)     = 51e8b58c
Integrity Check : /root/part_backup.json
checksum (calculated) = 37238cd7
checksum (stored)     = 37238cd7
/root/part.json : Master copy is intact. Backup copy Not present
/root/part_backup.json : Is is not intact or take_backup = 1
utils_file_clone: /root/part.json to /root/part_backup.json
utils_file_clone: /root/part.checksum to /root/part_backup.checksum
```

```
**Checking Sanity of : /root/boot.json **



Integrity Check : /root/boot.json
checksum (calculated) = c37563ff
checksum (stored)     = c37563ff
Integrity Check : /root/boot_backup.json
checksum (calculated) = ae8acfbb
checksum (stored)     = ae8acfbb
/root/boot.json : Master copy is intact. Backup copy Not present
/root/boot_backup.json : Is is not intact or take_backup = 1
utils_file_clone: /root/boot.json to /root/boot_backup.json
utils_file_clone: /root/boot.checksum to /root/boot_backup.checksum
Sanity check OK.. removed dirty bit file
Note: Max size of json token = 80
Key= image
Key= name
Val(str)= fota
Key= version
Val(str)= 1.0
Key= start_sector
Val(NUM)= 32
Key= bootargs_start
bootargs_start
Val(NUM)= 1
Key= ssid
Val(str)= innotest
```

```
Key= passphrase

Val(str)= innophase123

Key= bootargs_end

bootargs_end

Val(NUM)= 1

Key= name

Val(str)= test_app

Key= version

Val(str)= 1.0

Key= start_sector

Val(NUM)= 154

Key= bootargs_start

bootargs_start

Val(NUM)= 1

Key= ssid

Val(str)= innotest

Key= passphrase

Val(str)= innophase123

Key= bootargs_end

bootargs_end

Val(NUM)= 1

Key= name

Val(str)= test_app

Key= version

Val(str)= 2.1

Key= start_sector

Val(NUM)= 230

Key= bootargs_start
```

```
bootargs_start

Val(NUM)= 1

Key= ssid

Val(str)= innotest

Key= passphrase

Val(str)= innophase123

Key= bootargs_end

bootargs_end

Val(NUM)= 1

Key= baudrate

Val(NUM)= 2560000

Key= timeout

Val(NUM)= 0

Key= verbose

Val(NUM)= 1

key =  image

num val : 2


Boot indx = 2

ssbl_load_elf : Read elf @ e6000

Elf OK.

Reading section names @ offset = 38ef8


.text

.data

.bss

.virt

.hwreg.hw_ver
```

```
.hwreg.hw_pmu

.hwreg.hw_wfe

.hwreg.hw_evh

.hwreg.hw_timer

.hwreg.hw_boff

.hwreg.hw_gpio

.hwreg.hw_rnd

.hwreg.hw_qspi

.hwreg.hw_uart

.hwreg.hw_freq

.hwreg.hw_wdg

.hwreg.hw_tb

.hwreg.hw_sup

.hwreg.hw_trxdma

.hwreg.hw_sspi

.hwreg.hw_sdio

.hwreg.hw_cipher

.hwreg.hw_clone

.hwreg.hw_clone_bt

.hwreg.hw_clone_sdio

.hwreg.hw_i2c

.hwreg.hw_rxtdc

.hwreg.hw_pwm

.hwreg.hw_mmu

.hwreg.hw_afe

.hwreg.hw_core

.hwreg.hw_pdm

.hwreg.hw_rstclk
```

```
.hwreg.hw_bbrx_ofdm

.hwreg.hw_macif

.hwreg.hw_frame

.hwreg.hw_txbb

.hwreg.hw_ble

.hwreg.hw_bbrx_11b

.hwreg.hw_frontend_rx

.hwreg.hw_frontend

.hwreg.hw_fpga

.hwreg.hw_tx_dummy

.hwreg.hw_tap

.hwreg.hw_dpll

.comment

.ARM.attributes

.note.innophase

.shstrtab

Elf Load OK...

vm_flash location: 1049856

Starting image...

Boot-args:

      vm.flash_location=0x00100500

      passphrase=innophase123

      ssid=innotest


Resetting...

=====================================================
Build $Id: git-4b1aff048 $

vm.flash_location=0x00100500 passphrase=innophase123 ssid=innotest

$App:git-34f13ba

SDK Ver: SDK_2.3

Wifi Connect Demo App


Bootargs: ssid = innotest, passphrase = innophase123[0.956,952] WPA3/SAE is not

built in!

addr e0:69:3a:00:01:3d


Adding network: ssid = innotest  : passphrase = innophase123

Connecting to added network : innotest[1.607,738] CONNECT:e4:c3:2a:31:0c:ae

Channel:3 rssi:-17 dBm

wcm_notify_cb to App Layer - WCM_NOTIFY_MSG_LINK_UP

wcm_notify_cb to App Layer - WCM_NOTIFY_MSG_ADDRESS

[1.782,000] MYIP 192.168.0.14

[1.782,313] IPv6 [fe80::e269:3aff:fe00:13d]-link


Connected to < innotest > network


      ------------ Program Exit -------------------
```

11 Support

1. Sales Support: Contact an InnoPhase sales representative via email – sales@innophaseinc.com
2. Technical Support:
    a. Visit: https://innophaseinc.com/contact/
    b. Also Visit: https://innophaseinc.com/talaria-two-modules
    c. Contact: support@innophaseinc.com

InnoPhase is working diligently to provide outstanding support to all customers.

12 Disclaimers