

# Talaria TWO™ (INP2045)

Low Power Multi-Protocol Wireless Platform SoC

IEEE 802.11 b/g/n, BLE 5.0

## Application Note

Second Stage Boot Loader (SSBL)

Release: 08-02-2021

InnoPhase, Inc.

6815 Flanders Drive

San Diego, CA 92121

[innophaseinc.com](http://innophaseinc.com)

## Revision History

Version Number	Date	Comments
1.0	04-29-2020	First release.
1.1	06-29-2021	Procedure for flashing components and ELF paths updated.
2.0	08-02-2021	Updated for SDK 2.3 release.

## Contents

1	Figures.....	3
2	Tables .....	3
3	Terms & Definitions .....	3
4	Introduction .....	4
5	Description of operation.....	4
5.1	Memory Layout.....	5
5.2	Layout of Flash .....	7
5.3	SSBL Configuration.....	8
5.4	SSBL Boot Arguments .....	11
6	Building Components.....	12
6.1	Create File System (root.img) file .....	12
7	Flashing Components.....	13
7.1	Flashing .....	13
7.2	Expected Output.....	15
8	Support .....	22
9	Disclaimers .....	23

## 1 Figures

Figure 1: Memory when using the SSBL.....	5
Figure 2: Flash layout when using the SSBL .....	7
Figure 3: SSBL Bootargs stored in memory .....	11
Figure 4: Miniterm Console Ouptut .....	14
Figure 5: iPerf3 Client .....	21

## 2 Tables

Table 1: SSBL Configuration Files .....	8
---	---

## 3 Terms & Definitions

ECDSA	Elliptic Curve Digital Signature Algorithm
ELF	Executable and Linkable Format
OTA	Over The Air
OTP	One Time Programmable
PUF	Physical Unclonable Function
SDK	Software Development Kit
SHA256	Secure Hash Algorithm
SRAM	Static Random-Access Memory
SSBL	Second Stage Boot Loader

## 4 Introduction

The Second Stage Boot Loader (SSBL) is a piece of firmware that can be installed in Talaria TWO to enhance the flexibility of booting the applications on the device. The SSBL enables the following features on Talaria TWO:

1. Selective boot of one of the applications loaded into flash
2. Over the Air (OTA) update of applications (requires additional OTA application)

**Note:** All the Download Tool & Console Output screenshots in this document are from SDK 2.2 release.

## 5 Description of operation

The Second Stage Boot Loader (SSBL) is a special application that is written to Talaria TWO's flash. The chip's ROM boots the SSBL, which then determines the final application to boot. This is accomplished with a configuration file present in a filesystem also residing in flash. Applications supported by the SSBL are stripped ELF files written to flash memory.

## 5.1 Memory Layout

Figure 1 shows the memory layout when using the SSBL.

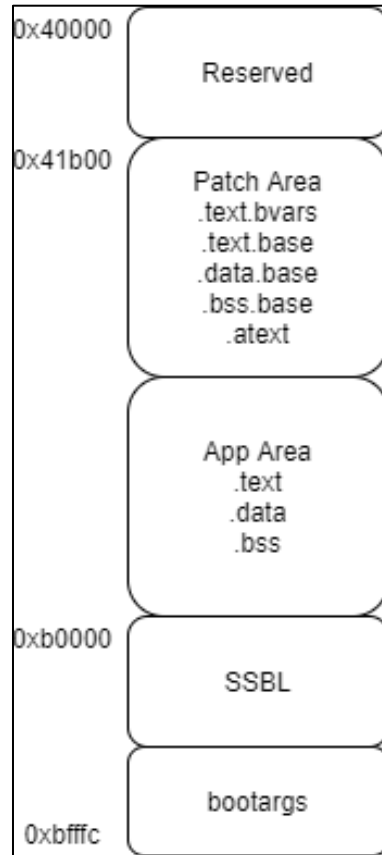


Figure 1: Memory when using the SSBL

1. 512 KB RAM
  - a. 0x40,000 – 0xC0,000
2. ROM patch area
  - a. 0x41b00
  - b. SSBL `.text.bvars` is replaced by application `.text.bvars`
  - c. SSBL and application must be used built using the same ROM patch
3. User Application area
  - a. Contains application `.text` and `.data` sections
4. Bootargs
  - a. 0xbfffc – grows backwards

### 5. SSBL area

a. 0xb0000

## 5.2 Layout of Flash

Figure 2 shows the layout of flash memory when using the SSBL. To use the SSBL, flash must contain at minimum the SSBL, the filesystem, and one application.



Figure 2: Flash layout when using the SSBL



## 5.3 SSBL Configuration

The SSBL is configured with JSON files present in the flash-based filesystem. Table 1 provides a description of the relevant files and their purpose. The contents of these files can be updated at installation time or by a running application to modify the behavior of the SSBL.

File	Purpose
part.json	<ol style="list-style-type: none"> <li>1. Image table which is a json array of applications image information. Each element in the image array gives information like image name starting sector of the elf, boot arguments etc.</li> <li>2. Application boot arguments</li> <li>3. Additional SSBL options</li> </ol>
boot.json	This file gives which application to boot. This holds the index of the application to boot. This index is the index in the image table in part.json file.

Table 1: SSBL Configuration Files

part.json

```
{
  "image" : [
    {
      "name" : "iperf_vm",
      "version" : "1.0",
      "start_sector" : 32,
      "bootargs_start": 1,
      "ssid" : "innotest",
      "passphrase" : "123467890",
      "bootargs_end" : 1
    },
    {
      "name" : "hello_world",
      "version" : "1.0",
      "start_sector" : 232,

```

```
"bootargs_start": 1,  
  "ssid" : "innotest",  
  "passphrase" : "123467890",  
  "bootargs_end" : 1  
}  
  
],  
  
  "baudrate"      : 2560000,  
  "timeout" : 0,  
  "verbose" : 1  
}
```

### 1. General parameters:

- a. baud – baud rate used by SSBL when using hio
- b. timeout – timeout used by SSBL when using hio
- c. verbose – verbosity mode
- d. image []: image information

### 2. Image information:

- a. name: name of application
- b. version: version number of applications
- c. sector: start sector of image in flash
- d. bootargs\_start: The following objects will be boot params
- e. bootargs\_end: end of boot arguments

boot.json

```
boot.json
{
  image : 0
}
```

where,

image – The image to boot from `part.json`

### 5.4 SSBL Boot Arguments

SSBL can pass bootargs to an application by utilizing the filesystem. SSBL reads the bootargs from the part.json file and stores the bootargs at memory location 0xbfffc where it grows backwards. The size occupied by the bootargs is dependent on the length and count of the bootargs read from the file system. Figure 3 shows how they are stored in memory.



Figure 3: SSBL Bootargs stored in memory

## 6 Building Components

This section describes building the required components for the SSBL.

### 6.1 Create File System (root.img) file

The root folder at `root_fs` contains the files that will be put into the filesystem image to be flashed to Talaria TWO. Before building the filesystem image for the first time, the configuration files need to be updated based on the applications you will load and your particular use of the SSBL (refer section 5.3).

Once you have updated the SSBL configuration files, run the following command from within the root folder to build the filesystem image:

```
~/sdk_2.3/root_fs$ ./build.sh
```

## 7 Flashing Components

After the SSBL, filesystem, and applications have been built, follow the instructions in this section to flash the components to Talaria TWO.

**Note:** If Talaria TWO has been flashed before, connect GPIO17 to ground on the peripheral header of the EVK, then press and release reset before following the instructions here. This will inhibit flash boot and allow the flash helper to be loaded, provided fuses have not already been blown.

### 7.1 Flashing

The following commands will write the SSBL and other components to flash. Run the commands from the `sdk_2.3` directory:

Load flash helper

```
./script/boot.py --device /dev/ttyUSB2 --reset=evk42_b1 ./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
```

Flash SSBL

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

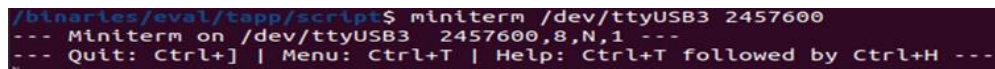
### Flash filesystem

```
./script/flash.py --device /dev/ttyUSB2 write 0x1C0000 ./root_fs/root.img
```

### Flash apps

```
./script/flash.py --device /dev/ttyUSB2 write 0x20000 ./apps/iperf3/bin/iperf3.elf  
./script/flash.py --device /dev/ttyUSB2 write 0xE8000 ./apps/helloworld/bin/helloworld.elf
```

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 4: Miniterm Console Ouput

Reset the board either by giving the command below or by pressing the reset button on the evb

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

## 7.2 Expected Output

```
Y-BOOT 208ef13 2019-07-22 12:26:54 -0500 790da1-b-7
ROM yoda-h0-rom-16-0-gd5a8e586
FLASH:PWAE
WWWWWAE[0.018,754] heapsize is less than requested 29952 < 30000
Build $Id: git-f92bee540 $
krn.heapsize=30000
$App:git-34f13ba
SDK Ver: SDK_2.3
SSBL No Secureboot krg
**Checking Sanity of : /root/fota_config.json **
Integrity Check : /root/fota_config.json
checksum (calculated) = 2b12fcd3
checksum (stored) = 2b12fcd3
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_backupp.checksum
**Checking Sanity of : /root/part.json **
Integrity Check : /root/part.json
checksum (calculated) = 3c2156c8
checksum (stored) = 3c2156c8
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)= iperf_vm
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)= hello_world
Key= version
Val(str)= 1.0
Key= start_sector
Val(NUM)= 232
Key= bootargs_start
bootargs_start
Val(NUM)= 1
Key= ssid
Val(str)= innotest
Key= passphrase
Val(str)= 123467890
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 = 3da32
.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
.note.innophase
.comment
.ARM.attributes
.shstrtab
Elf Load OK...
```

```
vm_flash location: 241920
Starting image...
Boot-args:
vm.flash_location=0x0003b100
passphrase=innophase123
ssid=innotest
Resetting...

=====

Build $Id: git-f92bee540 $
vm.flash_location=0x0003b100 passphrase=innophase123 ssid=innotest
addr 40:f2:01:61:14:aa
[1.384,942] CONNECT:e4:c3:2a:31:0c:ae Channel:7 rssi:-17 dBm
[3.246,031] MYIP 192.168.0.6
[3.246,077] IPv6 [fe80::42f2:1ff:fe61:14aa]-link
IPerf3 server @ 192.168.0.6

-----

Iperf3 TCP/UDP server listening on 5201

-----

Accepted connection from 192.168.0.11 port 45594
[ 1] local 192.168.0.6 port 5201 connected to 192.168.0.11 port 45596

-----

[ ID] Interval Transfer Bitrate
[ 1] iperf3[S-RX-tcp]: 0.0-30 sec 60.8 MBytes 17.0 Mbits/sec
User: 19398894 (64%)
IRQ: 2289883 (7%)
Idle: 8402587 (27%)

-----
```

```
Iperf3 TCP/UDP server listening on 5201
```

Run iperf3 client for this application.

```
osboxes@osboxes:~/Pictures/SDK_new/sdk_2.2$ iperf3 -c 192.168.0.107
Connecting to host 192.168.0.107, port 5201
[ 5] local 10.0.2.15 port 54572 connected to 192.168.0.107 port 5201
[ ID] Interval           Transfer     Bitrate      Retr  Cwnd
[ 5]  0.00-1.00      sec    887 KBytes  7.26 Mbits/sec    0   94.1 KBytes
[ 5]  1.00-2.00      sec    1.07 MBytes  9.00 Mbits/sec    0   94.1 KBytes
[ 5]  2.00-3.00      sec    1.01 MBytes  8.48 Mbits/sec    0   94.1 KBytes
[ 5]  3.00-4.00      sec    1.26 MBytes  10.5 Mbits/sec   0   94.1 KBytes
[ 5]  4.00-5.00      sec    1.47 MBytes  12.3 Mbits/sec   0   94.1 KBytes
[ 5]  5.00-6.00      sec    1.41 MBytes  11.8 Mbits/sec   0   94.1 KBytes
[ 5]  6.00-7.00      sec    1.07 MBytes  9.00 Mbits/sec    0   94.1 KBytes
[ 5]  7.00-8.00      sec    1.47 MBytes  12.3 Mbits/sec   0   94.1 KBytes
[ 5]  8.00-9.00      sec    1.04 MBytes  8.74 Mbits/sec    0   95.5 KBytes
[ 5]  9.00-10.00     sec    1.07 MBytes  8.99 Mbits/sec    0   94.1 KBytes
-----
[ ID] Interval           Transfer     Bitrate      Retr
[ 5]  0.00-10.00     sec    11.7 MBytes  9.85 Mbits/sec    0
[ 5]  0.00-10.00     sec    11.3 MBytes  9.48 Mbits/sec    0
sender
receiver
iperf Done.
```

Figure 5: iPerf3 Client

To run the helloworld application, make the changes in sdk\_2.3/root\_fs/boot.json and execute ./build.sh from sdk\_2.3/root\_fs to generate the root image.

Repeat the steps in section 7.1 and reset the board. On reboot, the new application will be loaded.

### 8 Support

1. Sales Support: Contact an InnoPhase sales representative via email – [sales@innophaseinc.com](mailto: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](mailto:support@innophaseinc.com)

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

## 9 Disclaimers

**Limited warranty and liability** — Information in this document is believed to be accurate and reliable. However, InnoPhase Incorporated does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and assumes no liability associated with the use of such information. InnoPhase Incorporated takes no responsibility for the content in this document if provided by an information source outside of InnoPhase Incorporated.

InnoPhase Incorporated disclaims liability for any indirect, incidental, punitive, special or consequential damages associated with the use of this document, applications and any products associated with information in this document, whether or not such damages are based on tort (including negligence), warranty, including warranty of merchantability, warranty of fitness for a particular purpose, breach of contract or any other legal theory. Further, InnoPhase Incorporated accepts no liability and makes no warranty, express or implied, for any assistance given with respect to any applications described herein or customer product design, or the application or use by any customer's third-party customer(s).

Notwithstanding any damages that a customer might incur for any reason whatsoever, InnoPhase Incorporated' aggregate and cumulative liability for the products described herein shall be limited in accordance with the Terms and Conditions of identified in the commercial sale documentation for such InnoPhase Incorporated products.

**Right to make changes** — InnoPhase Incorporated reserves the right to make changes to information published in this document, including, without limitation, changes to any specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

**Suitability for use** — InnoPhase Incorporated products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an InnoPhase Incorporated product can reasonably be expected to result in personal injury, death or severe property or environmental damage. InnoPhase Incorporated and its suppliers accept no liability for inclusion and/or use of InnoPhase Incorporated products in such equipment or applications and such inclusion and/or use is at the customer's own risk.

All trademarks, trade names and registered trademarks mentioned in this document are property of their respective owners and are hereby acknowledged.