This section serves a reference guide for using Talaria TWO boot arguments (bootargs) and its usage in different scenarios using different feature and protocols. It is to be used as a reference guide for using bootargs/set of bootargs customized as per the application requirement.

This section provides bootargs use case or examples for all layers from hardware interface to Wi-Fi configurations and it needs to be used as reference only.

# Introduction

Boot arguments are a list of key value pairs of the form <key>=<value> passed to the booted image

as an array of null-terminated strings (using the boot protocol). This is intended as a simple way to pass configuration to an application during development.

# Generic Syntax for Passing Boot Arguments

1. Used with Talaria TWO Download Tool (*freertos\_sdk\_x.y/pc\_tools/Download\_Tool/bin*).

The bootargs, if applicable by an application, should be entered in the Boot Arguments field, in a comma-separated form, as shown in Figure 1.

Table

Description automatically generated with medium confidence

Figure : Boot arguments for Download Tool

1. Used in CLI/command line

In general, bootargs, if applicable, are appended to the string of commands with path to the files (as illustrated with [boot-args]), in a space-separated format.

Following are the two scripts used for CLI operations:

* 1. boot.py – Script used to program RAM/Flash of Talaria TWO module and create .img file from .elf file.
  2. flash.py – Script to read and write to flash, which requires gordon.elf application.

**Note**:

1. Arguments in [ ] are optional and those in <> are mandatory.
2. Boot arguments are separated by a space “ “ in command line and by a comma “,” in the Download Tool GUI.
3. Program RAM:

|  |
| --- |
| <path\_script>/boot.py [device] <path\_app\_elf>/app.elf [boot\_args] |

1. Program Flash:

|  |
| --- |
| <path\_script>/boot.py [device] --reset=evk42\_bl\_bl --flash=all [additional commands] <path\_app\_elf>/app.elf [boot\_args] |

Based on the OS of the host machine and the interface in which the above commands are sent from, the [device] keyword can take different forms.

The [device] can be one of the following:

1. On Linux machine: --device /dev/ttyUSBn
2. On Windows machine:
   1. If libusbK installed (recommended): --device ftdi://ftdi:4232/3
   2. If libusbK not installed: --device COMn

Refer the following document for more details: RG\_Talaria\_TWO\_CLI\_Commands.pdf (*freertos\_sdk\_x.y\doc\ reference\_guides\cli\_reference\_guide*).

# List of Bootargs

This section describes the list of boot arguments available to configure various software/hardware modules. Such as hardware/host interface, pin configuration, serial console, Wi-Fi operations, miscellaneous etc.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Argument name** | **Bootarg** | **Type** | **Default** | **Description** |
| Host interface baud rate | hio.baudrate | int | 921600 | Select the baudrate for the HIO interface |
| Host interface max message size | hio.maxsize | int | 1600 | Configure the maximum size of a host interface message sent to Talaria TWO |
| Host interface transport type | hio.transport | int | NA | Select the HIO interface as UART/SPI/SDIO:   * hio.transport=uart |
| Default MAC address | hwaddr | string | (random) | Default MAC address for the device |
| Console baud rate | krn.console.baudrate | int | 2457600 | Baudrate for the serial interface for console output |
| Console buffer | krn.console.bufsz | int | 1024 | Select the size of the console output buffer. The value should be an even power-of-two (if not it will be rounded down to the nearest lower power of two) |
| Core dump | krn.coredump | bool | 0 | Enable support for download of coredump using UART after a crash |
| Pin configuration | krn.gpio | string | ------------------------ | Configure the state of the 23 GPIO pins. Set 23 characters (starting with GPIO 0) with the following meaning: '-' = No change, 'k' = Power save keep, 'K' = Power save keep with pull up, 'H' = Output high, 'L' = Output low, 'i' = Input, 'I' = Input with pull up, 'p' = No pull up, 'P' = Pull up |
| Initialize system time | krn.systime | int | 0 | Initialize system time to this number in seconds. |
| Set antenna gain | rf.antenna\_gain | Positive decimal number | 0 | Set antenna gain in dBi, e.g. "2.15". Is used in calculation of output power to comply with regulatory domain settings |
| Virtual memory flash location | vm.flash\_location | int | None | The recommended way to place virtual memory image on flash is to add a dedicated partition. This setting makes it possible to use virtual memory without such a partition |
| Virtual memory cache size | vm.pages | int | 256 | Set the virtual memory cache size in number of pages. Each page is 256 bytes, so the default results in 64 kilobytes of cache |
| Wi-Fi Announce AMSDU capability | wifi.amsdu | bool | 1 | Announce that the device can receive AMSDUs |
| Gratuitous ARP period | wifi.arp\_grat\_period | int | 60 | Period in seconds for gratuitous ARP announcements. Set to zero to disable |
| Wi-Fi Channels | wifi.channels | string | 1-13 | Set which Wi-Fi channels to use. Set "1-11,13" to use channels 1 to 11 and 13. |
| Wi-Fi Cloud Power Mode | wifi.cloud\_pm | bool | 0 | Reduced power consumption for units only communicating with the cloud. Internal LAN traffic may not work. |
| Wi-Fi Dynamic listen interval | wifi.dyn\_listen\_int | bool | 0 | Listen to all beacons if there has been traffic recently |
| Wi-Fi Enable/disable HT (802.11n) | wifi.ht | bool | 1 | Enable or disable HT mode (802.11n). When disabled 802.11bg is enabled |
| Wake time after keep alive | wifi.keep\_alive\_wake\_time | int | 8 | Time in milliseconds to stay awake after a keep alive NULL frame |
| Wi-Fi Beacon listen interval | wifi.listen\_interval | int | 0 | Configure how often the device shall wake up to listen for beacons. Set to 0 to listen to every DTIM beacon. Set to 1 to listen to every beacon. |
| Wi-Fi Max idle period | wifi.max\_idle\_period | int | 1800 | This parameter specifies how many seconds the device can be idle before it must send a keep alive to the AP |
| Wi-Fi MSDU lifetime | wifi.msdu\_lifetime | int | 500 | This parameter specifies the time in milliseconds for which the WiFi MAC layer will try to send the frame before it is discarded |
| Wi-Fi Nap scan | wifi.nap\_scan | int | 1 | Turn off receiver for uninteresting frames during scan. When the receiver has started to receive a frame with high signal strength, and this frame is not a beacon or probe response, it can turn off the receiver ("take a nap") for the duration of this frame to save power |
| Wi-Fi Nap scan RSSI threshold | wifi.nap\_scan\_rssi | int | -70 | Threshold to determine if a frame is strong or not when using the wifi.nap\_scan function |
| Wi-Fi Nap in station mode (associated) | wifi.nap\_sta | int | 0 | Turn off receiver for uninteresting frames in associated station mode |
| Wi-Fi Don’t receive multicast frames which are not broadcast | wifi.only\_broadcast | int | 0 | "Take a nap" for multicast frames which are not broadcast (address ff:ff:ff:ff:ff:ff) |
| Wi-Fi PS poll | wifi.ps\_poll | bool | 0 | Use Power Save poll to fetch incoming packets when a beacon was missed |
| Wi-Fi Scan max listen time | wifi.scan\_max\_listen\_time | int | 24 | Maximum time in milliseconds to listen on a channel during scan. The time to send the probe requests is not included in this time |
| Wi-Fi Scan max responses | wifi.scan\_max\_responses | int | 0 | When this number of responses has been reached, the scan will be stopped |
| Wi-Fi Scan min listen time | wifi.scan\_min\_listen\_time | int | 8 | Minimum time in milliseconds to listen on a channel during scan |
| Wi-Fi Scan number of probes | wifi.scan\_num\_probes | int | 2 | Number of probe requests to send on each channel during scan. Set to 1 or 2 |
| Wi-Fi Scan wait time between channels | wifi.scan\_wait\_time | int | 0 | Wait time in milliseconds between channel during scan |
| Wi-Fi Traffic timeout | wifi.traffic\_timeout | int | 12 | Set power save traffic timeout. The device will stay awake and listen for incoming frames this time (in ms) after other frames has been sent or received |
| Wi-Fi Tx power save | wifi.tx\_ps | bool | 0 | Outgoing frames will be sent with the power save bit set in the frame |
| Wi-Fi DTIM only | wifi.dtim\_only | bool | 0 | Only listens to DTIMs (no ps-poll at beacon miss). |

Table : List of bootargs

# Boot Arguments

## Host Interface

### Host Interface Transport Type

Specify the Host Interface (HIO) transport type, using the UART/SPI/SDIO interface.

#### Syntax

|  |
| --- |
| hio.transport=<transport\_tpye>  <transport\_type>:  Type: int  Range: 0-2  Value: 0-UART (default),1-SPI, 2-SDIO  **Note**: SDIO is not supported in the current release. |

#### Example

Configuring host interface transport type as UART.

**Command**:

|  |
| --- |
| ./script/boot.py --device /dev/ttyUSB2 --reset=evk42\_bl\_bl ./apps/hello\_world/bin/hello\_world.elf **hio.transport=0** |

**Note**: Single line command.

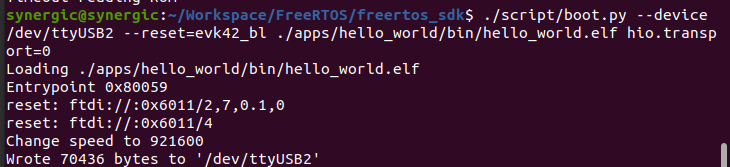


Figure : Configuring host interface transport type as UART

### Host Interface Baudrate

Set the baudrate for the HIO interface

#### Syntax

|  |
| --- |
| hio.baudrate=<baud\_rate>  <baud\_rate>:  Type: int  Range: 300-921600  Value: 921600(default)  Valid baudrate values: 300, 600, 1200, 2400, 4800, 9600, 14400, 19200, 38400, 57600, 115200, 230400, 460800, 921600 |

#### Example

Set the baudrate for the HIO interface at 921600.

**Command**:

|  |
| --- |
| ./script/boot.py --device /dev/ttyUSB2 --reset=evk42\_bl\_bl ./apps/hello\_world/bin/hello\_world.elf **krn.console.baudrate=921600** |

**Note**: Single line command.

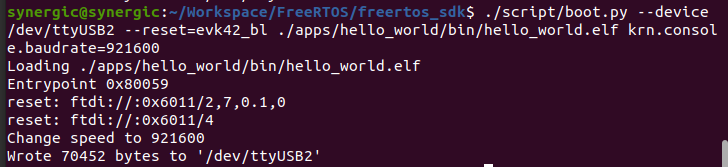


Figure : Setting host interface baudrate

### Host Interface Max Message Size

Configure the maximum size of a host interface message sent to Talaria TWO

#### Syntax

|  |
| --- |
| hio.maxsize=<size\_bytes>  <size\_bytes>:  Type: int  Value: 1600 (default) |

#### Example

Set the maximum size of a host message 1024 (bytes).

**Command**:

|  |
| --- |
| ./script/boot.py --device /dev/ttyUSB2 --reset=evk42\_bl\_bl --speed 2457600 ./apps/gordon.elf **hio.maxsize=1024** |

**Note**: Single line command.



Figure : Set maximum size of host message

### Default MAC Address for the Device

Set default MAC address for the device.

#### Syntax

|  |
| --- |
| hwaddr =<mac\_addr>  <mac\_addr>:  Type: string  Value: no default. |

#### Example

Set the default MAC address of the device as 01:02:03:04:05:06.

**Command**:

|  |
| --- |
| ./script/boot.py --device /dev/ttyUSB2 --reset=evk42\_bl apps/hello\_world/bin/hello\_world.elf **hwaddr=01:02:03:04:05:06** |

**Note**: Single line command.

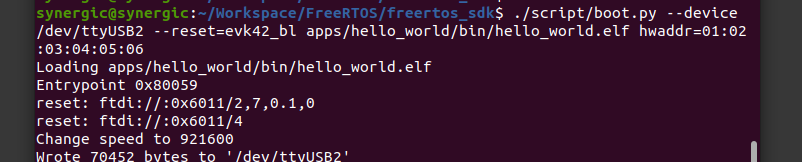


Figure : Set default MAC address of the device

### Console Baudrate

Set the baudrate for the serial interface for console output.

#### Syntax

|  |
| --- |
| krn.console.baudrate=<baudrate>  <baudrate>:  Type: int  Value: 2457600 (default)  Valid baudrate values: 1200, 2400, 4800, 9600, 14400, 19200, 38400, 57600, 115200, 230400, 460800, 921600, 2457600 |

#### Example

Set the baudrate for device console output at 115200.

**Command**:

|  |
| --- |
| ./script/boot.py --device /dev/ttyUSB2 --reset=evk42\_bl apps/hello\_world/bin/hello\_world.elf **krn.console.baudrate=115200** |

**Note**: Single line command.

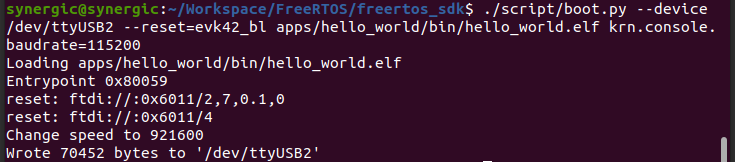


Figure : Set baudrate for device console output

### Coredump

Enable support for download of coredump using UART after a crash.

#### Syntax

|  |
| --- |
| krn.coredump=<ena/dis>  <ena/dis>:  Type: int  Range: 0-1  Value: 0-disabled (default), 1-enabled |

#### Example

Enable the coredump via UART after a crash.

**Command:**

|  |
| --- |
| ./script/boot.py --device /dev/ttyUSB2 --reset=evk42\_bl apps/hello\_world/bin/hello\_world.elf **krn.coredump=1** |

**Note**: Single line command.

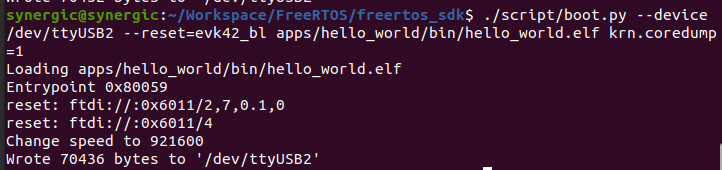


Figure : Enable coredump via UART after crash

### PIN Configuration

Configure the state of the 23 GPIO PINS.

#### Syntax

|  |
| --- |
| krn.gpio=-----------------------  <----------------------->:  Type: String of 23 characters  Options for each character:   * '-' : No change * 'k' : Power save keep * 'K' : Power save keep with pull up * 'H' : Output high * 'L' : Output low * 'i' : Input * 'I' : Input with pull up * 'p' : No pull up * 'P' : Pull up   No default |

#### Example

Set the GPIO 13 pin as input with pull up:

**Command:**

|  |
| --- |
| ./script/boot.py --device /dev/ttyUSB2 --reset=evk42\_bl apps/hello\_world/bin/hello\_world.elf **krn.gpio=-------------I---------** |

**Note**: Single line command.

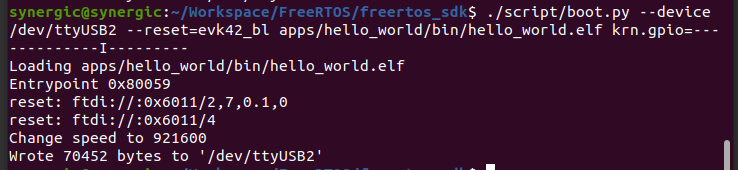


Figure : Set the GPIO 13 pin as input with pull up

### Gratuitous ARP Period

Set the ARP period in seconds for gratuitous ARP announcements. Set to zero to disable.

#### Syntax

|  |
| --- |
| wifi.arp\_grat\_period=<arp\_period>  <arp\_period>:  Type: int  Value: 0-disabled, 60-default, N-arp period in seconds |

#### Example

Set the ARP period to 90 (s).

**Command**:

|  |
| --- |
| ./script/boot.py --device /dev/ttyUSB2 --reset=evk42\_bl examples/using\_wifi/bin/wifi\_connect.elf np\_conf\_path=/data/nprofile.json **wifi.arp\_grat\_period=90** |

**Note**: Single line command.

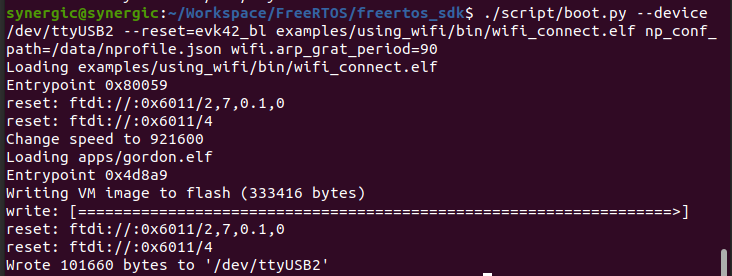


Figure : Set the ARP period to 90 (s)

### Wi-Fi Channels

Set which Wi-Fi channels to use.

#### Syntax

|  |
| --- |
| wifi.channels=<channels>  <channels>:  Type: string  Range: 1-13  Value: enter as a range and/or individual channels |

#### Example

Set the Wi-Fi to use channels 1 to 6 and 11.

**Command:**

|  |
| --- |
| ./script/boot.py --device /dev/ttyUSB2 --reset=evk42\_bl examples/using\_wifi/bin/wifi\_connect.elf np\_conf\_path=/data/nprofile.json **wifi.channels=1-6,11** |

**Note**: Single line command.

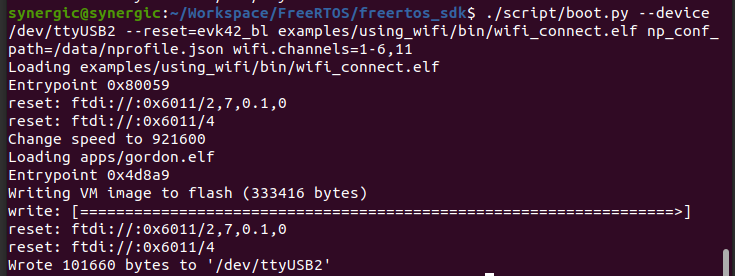


Figure : Set the Wi-Fi to use channels 1 to 6 and 11

### Wi-Fi Dynamic Listen Interval

Configure the dynamic listening interval

#### Syntax

|  |
| --- |
| wifi.dyn\_listen\_int=<dyn\_listen\_int>  <dyn\_listen\_int>:  Type: int  Range: 0-1  Value: 0-disabled,1-listen to all beacons if there has been traffic recently |

#### Example

Turn dynamic listening ON.

**Command:**

|  |
| --- |
| ./script/boot.py --device /dev/ttyUSB2 --reset=evk42\_bl examples/using\_wifi/bin/wifi\_connect.elf np\_conf\_path=/data/nprofile.json **wifi.dyn\_listen\_int=1** |

**Note**: Single line command.

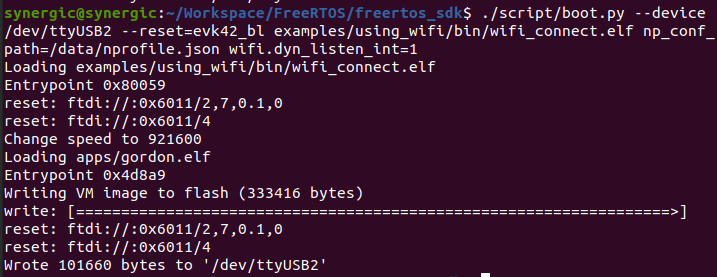


Figure : Turning dynamic listening ON

### Wi-Fi Beacon Listen Interval

Configure how often the device should wake up to listen for beacons.

#### Syntax

|  |
| --- |
| wifi.listen\_interval=<listen\_interval>  <listen\_interval>:  Type: int  Range: 0-255  Value: 0-disabled, 1-listen to every beacon, N-listen to every N beacon elapsed |

#### Example

Set the beacon listening interval to 10.

**Command:**

|  |
| --- |
| ./script/boot.py --device /dev/ttyUSB2 --reset=evk42\_bl examples/using\_wifi/bin/wifi\_connect.elf np\_conf\_path=/data/nprofile.json **wifi.listen\_interval=10** |

**Note**: Single line command.

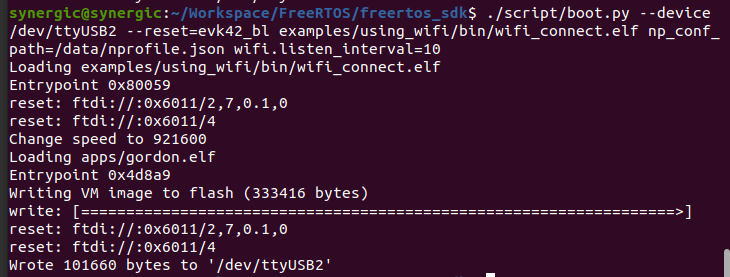


Figure : Set the beacon listening interval to 10

### Wi-Fi Max Idle Period

Specifies how many seconds the device can be idle for before it must send a keep alive to the AP.

#### Syntax

|  |
| --- |
| wifi.max\_idle\_period=<max\_idle>  <max\_idle>:  Type: int  Value: 1800 (default), N-max idle time in seconds |

#### Example

Set the max idle time to 300 (s).

**Command:**

|  |
| --- |
| ./script/boot.py --device /dev/ttyUSB2 --reset=evk42\_bl examples/using\_wifi/bin/wifi\_connect.elf np\_conf\_path=/data/nprofile.json **wifi.max\_idle\_period=300** |

**Note**: Single line command.

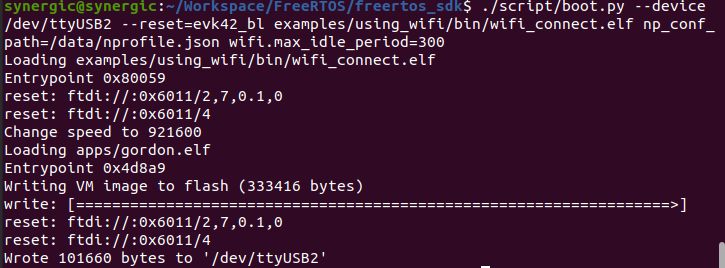
****

Figure : Set the max idle time to 300 (s)

### Wi-Fi Traffic Timeout

Sets the time in milliseconds in which the device will stay awake after incoming or outgoing traffic.

It is possible to increase the default value to decrease latency to the expense of higher power consumption. A traffic timeout value of 0 will keep the device awake forever.

#### Syntax

|  |
| --- |
| wifi.traffic\_timeout=<timeout>  <timeout>:  Type: int  Value: 12 (default), 0 (Disables sleep) |

#### Example

Set the traffic timeout value to 24 (ms).

**Command:**

|  |
| --- |
| ./script/boot.py --device /dev/ttyUSB2 --reset=evk42\_bl examples/using\_wifi/bin/wifi\_connect.elf np\_conf\_path=/data/nprofile.json **wifi.traffic\_timeout=24** |

**Note**: Single line command.

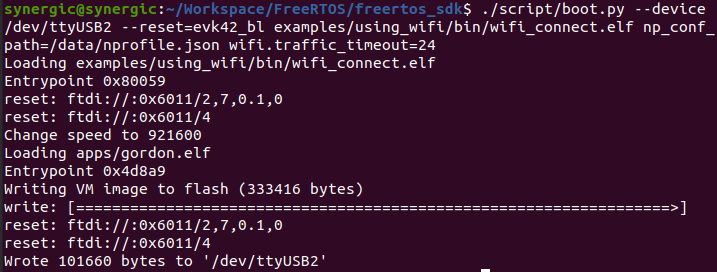


Figure : Set the traffic timeout value to 24 (ms)

### Wi-Fi Tx Power Save

When this mode is enabled, outgoing frames will be sent with the power save bit set in the frame.

#### Syntax

|  |
| --- |
| wifi.tx\_ps=<dis/ena>  <dis/ena>:  Type: int  Range: 0-1  Value: 0-disabled (default), 1–enabled |

#### Example

Turn ON the Wi-Fi Tx power save mode.

**Command:**

|  |
| --- |
| ./script/boot.py --device /dev/ttyUSB2 --reset=evk42\_bl examples/using\_wifi/bin/wifi\_connect.elf np\_conf\_path=/data/nprofile.json **wifi.tx\_ps=1** |

**Note**: Single line command.

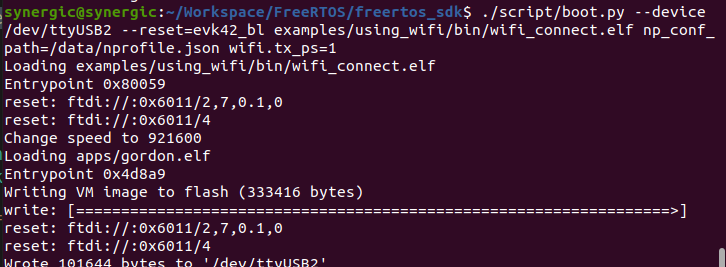


Figure : Turn ON the Wi-Fi Tx power save mode

### Wi-Fi DTIM Only Listening

When this mode is enabled, the device will not listen to every beacon in case of beacon misses. It wakes up only at the effective listen interval.

#### Syntax

|  |
| --- |
| wifi.dtim\_only = < dtim\_only >  < dtim\_only > :  Type: int  Range: 0-1  Value: 0-disabled, 1-Does not switch to listen mode for every beacon in case of beacon miss |

#### Example

Turn DTIM only ON.

**Command:**

|  |
| --- |
| ./script/boot.py --device /dev/ttyUSB2 --reset=evk42\_bl examples/using\_wifi/bin/wifi\_connect.elf np\_conf\_path=/sys/nprofile.json wifi.dtim\_only=1 |

**Note**: Single line command.

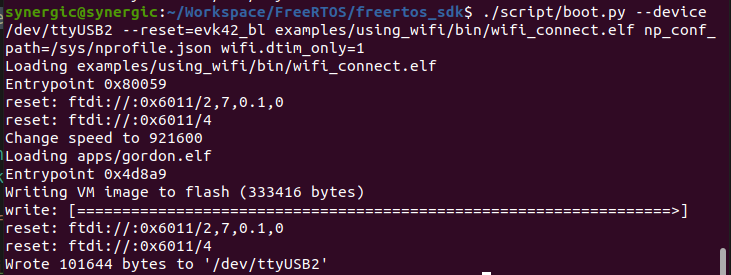


Figure : Turn ON the Wi-Fi DTIM only listening mode