Instructions for Setting up

Google Cloud IoT for the Allwinner R18

developer Kit

Allwinner BU1-PD4

June 07, 2017

Version #	Implemented By	Revision Date	Revised By	Revision Date	Notes
0.1	Shawn Wong	05/02/2017			Please use FTP to download the Tina SDK
0.2	Joshua Cha	08/14/2017			

Contents

Disclaimer

Allwinner Technology Co.,Ltd (Allwinner) does not make any warranties, expressed or implied, including those of merchantability and fitness for a particular purpose, with respect to any information, data, statements, or services made available to our collaborators.

While Allwinner makes its best efforts to produce valid instructions, neither Allwinner nor Google, endorses commercial products or certifies that they meet standards set by government agencies or private organizations.

This publication provides informative material based on "Cloud IoT Alpha User Guide" document from Google and professional testing results from Allwinner. It is confidential until Google makes their announcement or provides authorization. This information is intended to be as accurate as possible at the time of publication, but Allwinner assumes no responsibility for any losses or damages that might result because of reliance on this material.

Chapter 1

Executive Summary

1 *Definitions*

1) Google Cloud IoT: Cloud IoT is a fully managed service on Google Cloud Platform to securely connect and manage a few or millions of Internet of Things (IoT) and connected devices. Cloud IoT service makes it easy for users to ingest data from large number of connected devices and build rich applications by integrating with the data analytics services of Google Cloud Platform[1].

a) MQTT: The protocol supported by Google Cloud IoT at launch.

b) JWT: JSON Web Token. Used to securely transmit information via MQTT bridge.

2) M64: A developer board which is designed and assembled by Banana Pi.

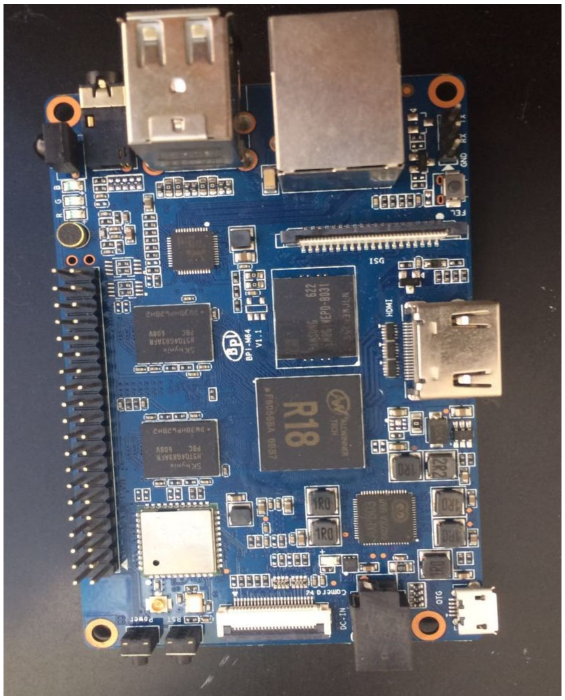


Figure 1.1 M64 Board

3) Tina: An embedded Linux operating system based on OpenWrt which is designed by Allwinner, widely used on embedded devices including smart speakers, robots, robotic sweepers and smart home devices.

4) sys_config.fex: The file defines system configuration includes GPIO pins and sets up DRAM, PMU, sensors, etc parameters on Tina SDK for Allwinner's developer boards. Different boards may have different sys_config.fex parameters.

5) AP6212: Broadcom Wi-Fi module which is used by Banana Pi boards.

6) RTL8723: Realtek Wi-Fi module which is used by M64 boards.

2 *Prerequisites*

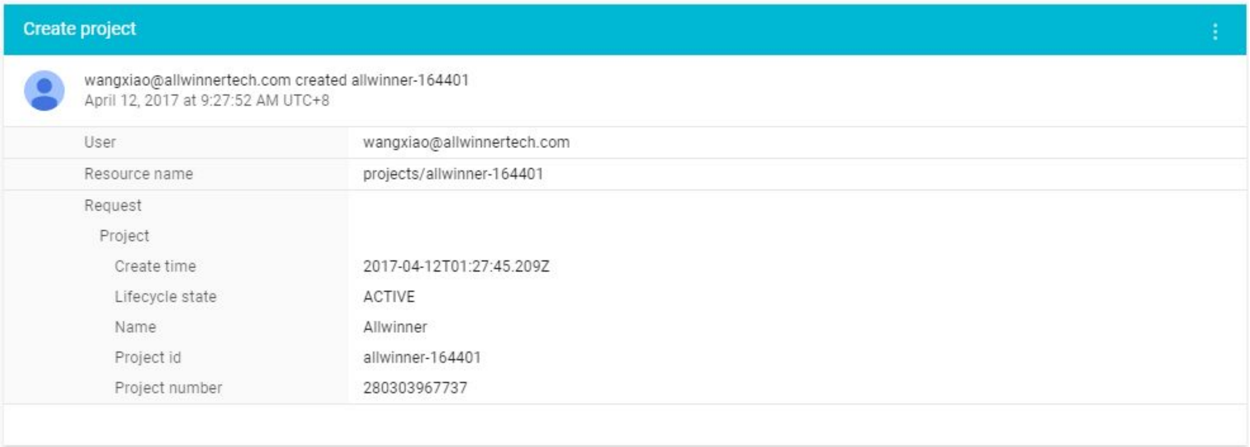
1 Prepare firmware image

1.2 Option 1: Download the Tina Linux SDK to build firmware image: Details in Chapter 2.1.

1.3 Option 2: Use prebuilt firmware: Download prebuilt firmware from Allwinner's FTP site, details in Chapter 2.2.

2 Download Google Cloud IoT on to development PC: Details are found in the in Cloud IoT Alpha-User Guide

3 Create a Google Cloud IoT project and Cloud Pub/Sub topic on website console or programmatically on command line tool: Details in Cloud IoT Alpha UserGuide



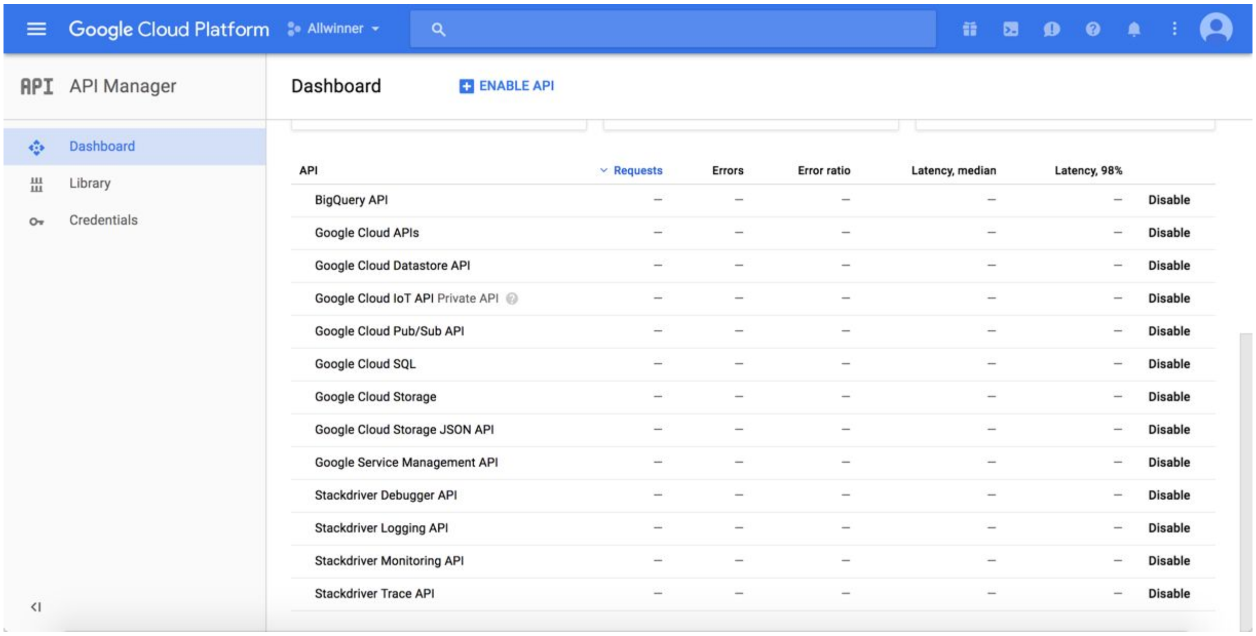
Create project

wangxiao@allwinnertech.com created allwinner-164401
April 12, 2017 at 9:27:52 AM UTC+8

User	wangxiao@allwinnertech.com
Resource name	projects/allwinner-164401
Request	
Project	
Create time	2017-04-12T01:27:45.209Z
Lifecycle state	ACTIVE
Name	Allwinner
Project id	allwinner-164401
Project number	280303967737

Figure 1.2 Project created, check this info on Google Cloud Platform console.

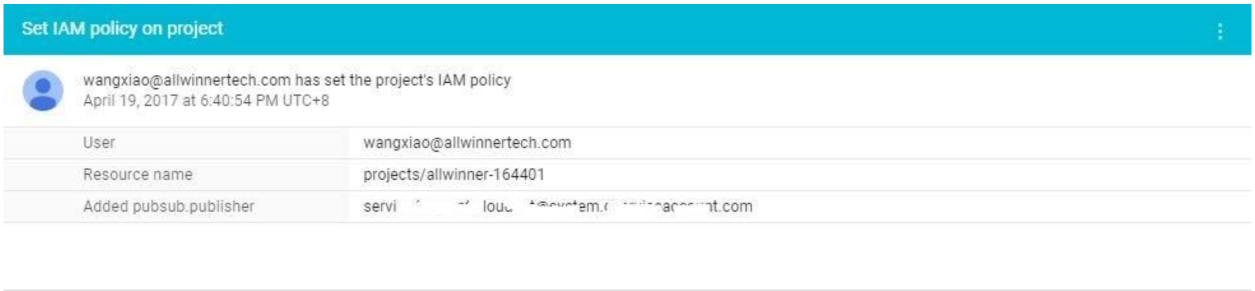
4 Enable the Cloud IoT API and the Cloud Pub/Sub API: Details in Cloud IoT Alpha User Guide



API	Requests	Errors	Error ratio	Latency, median	Latency, 98%	
BigQuery API	–	–	–	–	–	Disable
Google Cloud APIs	–	–	–	–	–	Disable
Google Cloud Datastore API	–	–	–	–	–	Disable
Google Cloud IoT API Private API	–	–	–	–	–	Disable
Google Cloud Pub/Sub API	–	–	–	–	–	Disable
Google Cloud SQL	–	–	–	–	–	Disable
Google Cloud Storage	–	–	–	–	–	Disable
Google Cloud Storage JSON API	–	–	–	–	–	Disable
Google Service Management API	–	–	–	–	–	Disable
Stackdriver Debugger API	–	–	–	–	–	Disable
Stackdriver Logging API	–	–	–	–	–	Disable
Stackdriver Monitoring API	–	–	–	–	–	Disable
Stackdriver Trace API	–	–	–	–	–	Disable

Figure 1.3 Enabled APIs

5 Set IAM policy on project: Details in Cloud IoT Alpha User Guide.



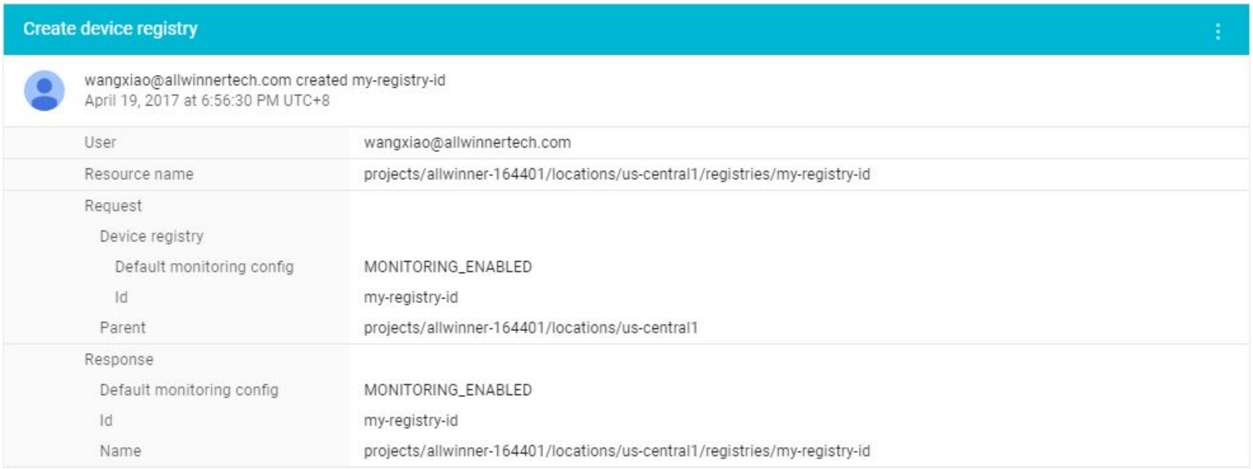
Set IAM policy on project

wangxiao@allwinnertech.com has set the project's IAM policy
April 19, 2017 at 6:40:54 PM UTC+8

User	wangxiao@allwinnertech.com
Resource name	projects/allwinner-164401
Added pubsub.publisher	servi [redacted] t.com

Figure 1.4 Set IAM policy on project

6 Create device registry: Details in Cloud IoT Alpha User Guide



Create device registry

wangxiao@allwinnertech.com created my-registry-id
April 19, 2017 at 6:56:30 PM UTC+8

User	wangxiao@allwinnertech.com
Resource name	projects/allwinner-164401/locations/us-central1/registries/my-registry-id
Request	
Device registry	
Default monitoring config	MONITORING_ENABLED
Id	my-registry-id
Parent	projects/allwinner-164401/locations/us-central1
Response	
Default monitoring config	MONITORING_ENABLED
Id	my-registry-id
Name	projects/allwinner-164401/locations/us-central1/registries/my-registry-id

Figure 1.5 Created device registry

7 Create device: Details in Cloud IoT Alpha User Guide

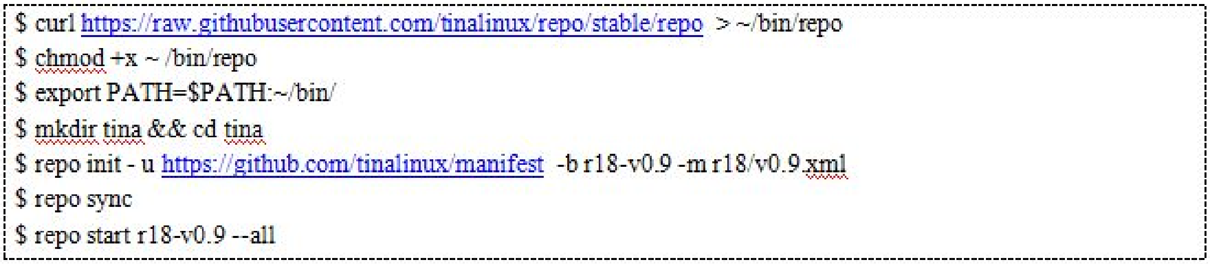
8 Openssl: To generate public/private key pairs. The Clout IoT Alpha User Guide has information on this step

Before running Google Cloud IoT demo, I recommend you to read Cloud Iot Alpha User Guide thoroughly and make sure the you meet the above prerequisites.

Chapter 2: Developer Guide

1 Download Tina Linux from GitHub or FTP server

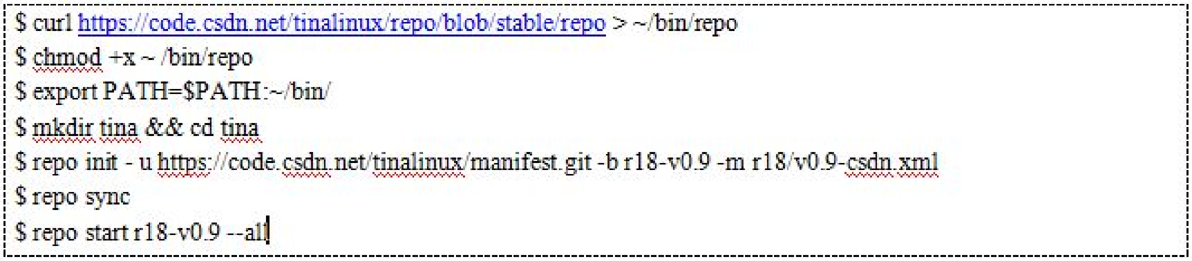
Tina Linux for R18 on GitHub is a demo version, Allwinner Team will keep updating the source code in the future. It's a completely public and free software. You can download it by the following commands:



```
$ curl https://raw.githubusercontent.com/tinalinux/repo/stable/repo  > ~/bin/repo
$ chmod +x ~ /bin/repo
$ export PATH=$PATH:~/bin/
$ mkdir tina && cd tina
$ repo init - u https://github.com/tinalinux/manifest  -b r18-v0.9 -m r18/v0.9.xml
$ repo sync
$ repo start r18-v0.9 --all
```

Figure 2.1 Commands for downloading Tina Linux

For customers in China, we recommend you download the SDK from CSDN by the following these commands:



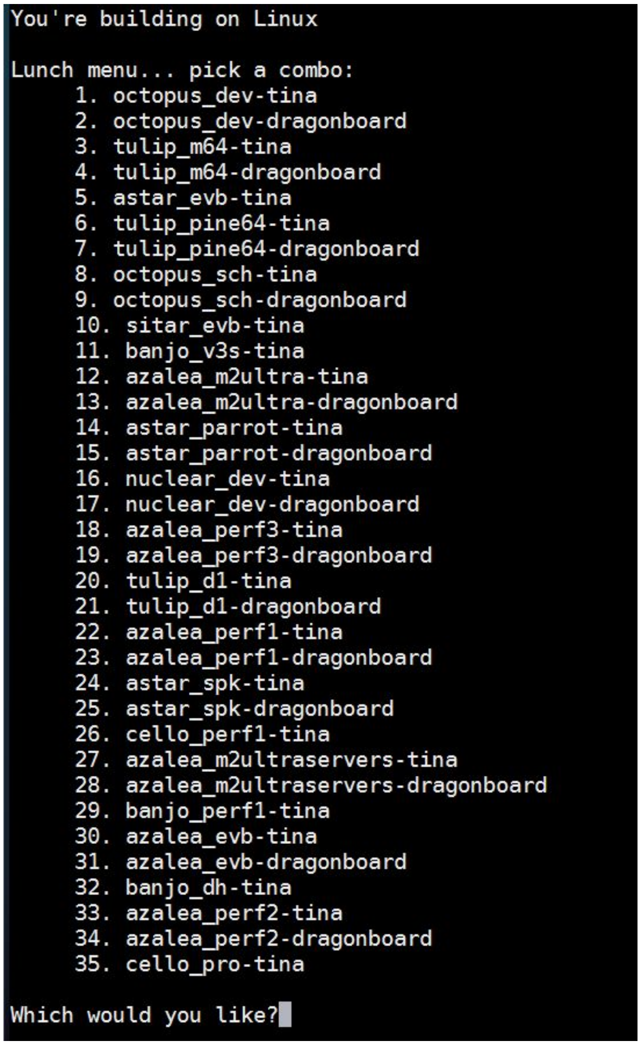
```
$ curl https://code.csdn.net/tinalinux/repo/blob/stable/repo > ~/bin/repo
$ chmod +x ~ /bin/repo
$ export PATH=$PATH:~/bin/
$ mkdir tina && cd tina
$ repo init - u https://code.csdn.net/tinalinux/manifest.git -b r18-v0.9 -m r18/v0.9-csdn.xml
$ repo sync
$ repo start r18-v0.9 --all
```

Figure 2.2 Download Tina Linux from CSDN for Chinese customers

2 Necessary modules needed by Google Cloud IoT demo on Tina SDK

2.1 Prepare for compiling Tina Linux

On Tina Linux SDK, run source build/envsetup.sh && lunch and select tulip_m64-tina (option #16 in the example below) or M64 board, this operation will load vendor setup scripts and set up some environmental parameters. Run make menuconfig command under root directory of Tina Linux SDK like many Linux OS's to choose the features or modules of Tina that will be compiled.



```
You're building on Linux

Lunch menu... pick a combo:
     1. octopus_dev-tina
     2. octopus_dev-dragonboard
     3. tulip_m64-tina
     4. tulip_m64-dragonboard
     5. astar_evb-tina
     6. tulip_pine64-tina
     7. tulip_pine64-dragonboard
     8. octopus_sch-tina
     9. octopus_sch-dragonboard
     10. sitar_evb-tina
     11. banjo_v3s-tina
     12. azalea_m2ultra-tina
     13. azalea_m2ultra-dragonboard
     14. astar_parrot-tina
     15. astar_parrot-dragonboard
     16. nuclear_dev-tina
     17. nuclear_dev-dragonboard
     18. azalea_perf3-tina
     19. azalea_perf3-dragonboard
     20. tulip_d1-tina
     21. tulip_d1-dragonboard
     22. azalea_perf1-tina
     23. azalea_perf1-dragonboard
     24. astar_spk-tina
     25. astar_spk-dragonboard
     26. cello_perf1-tina
     27. azalea_m2ultraservers-tina
     28. azalea_m2ultraservers-dragonboard
     29. banjo_perf1-tina
     30. azalea_evb-tina
     31. azalea_evb-dragonboard
     32. banjo_dh-tina
     33. azalea_perf2-tina
     34. azalea_perf2-dragonboard
     35. cello_pro-tina

Which would you like?
```

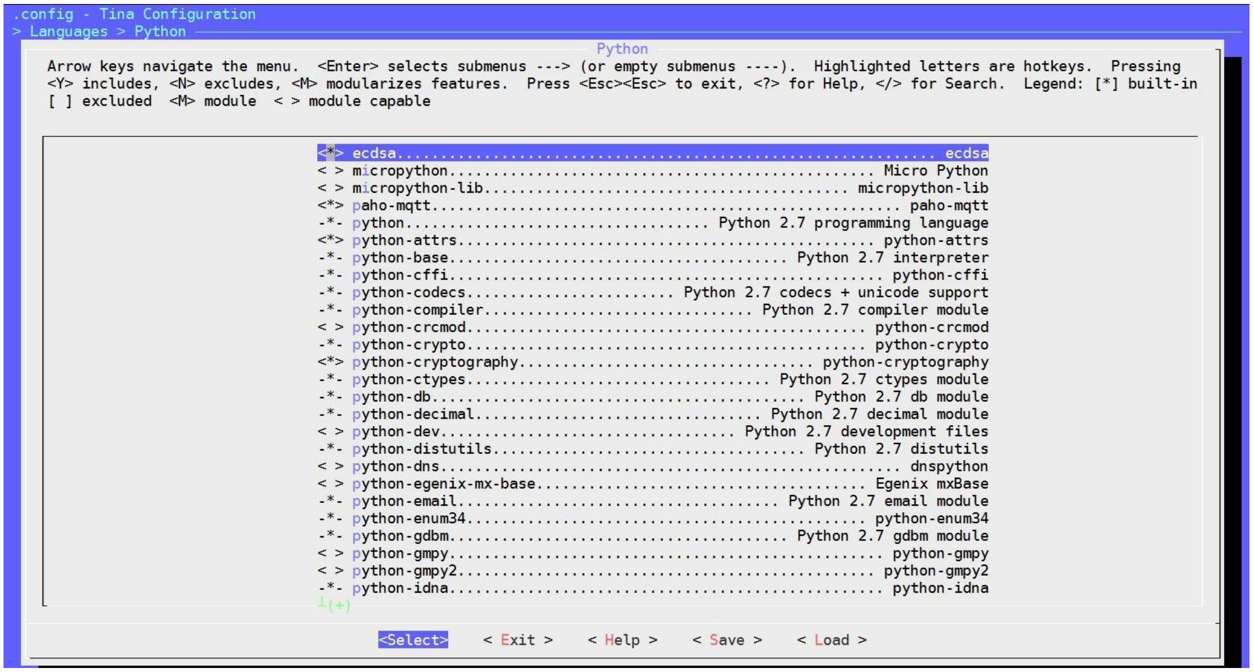
Figure 2.4 lunch result

Note: If you are using a Pine64 board, select tulip-pine64-tina

2.2 Select Python packages needed

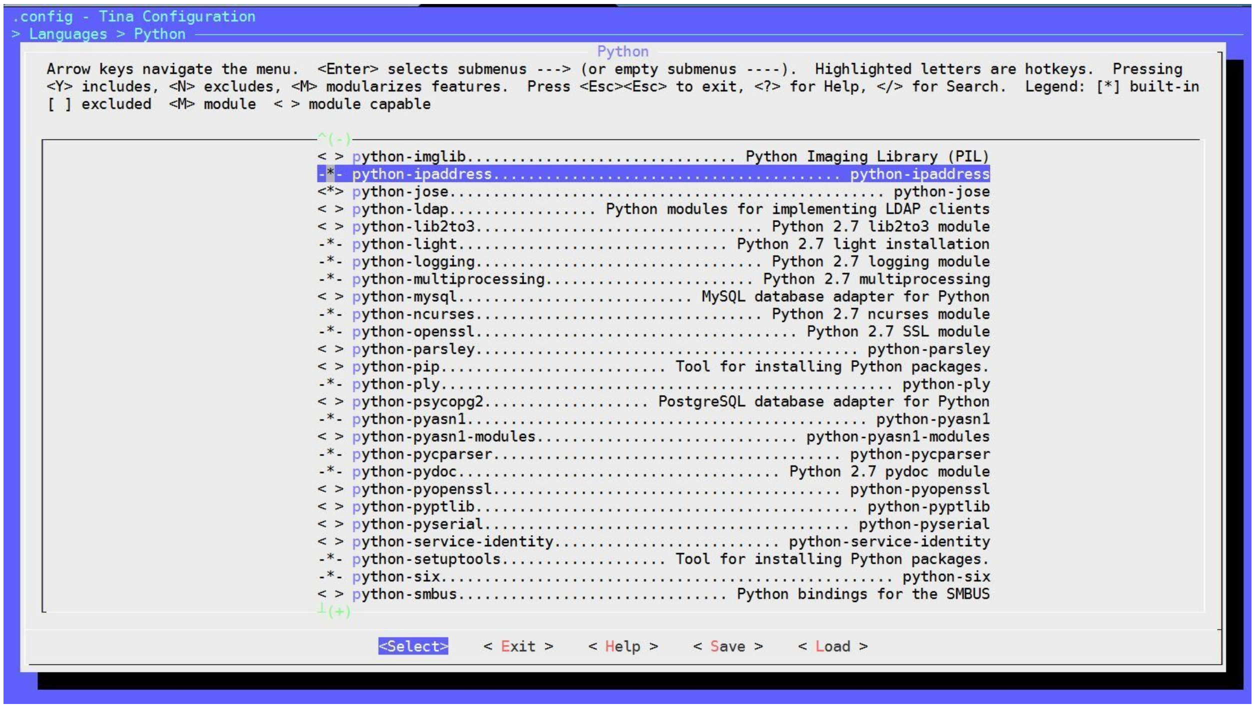
make menuconfig will pop up a menu-driven user interface, select the following package after entering Languages--------------> Python menu(The path will be shown on the top left corner on every screen shot in this document),(M) means compile it as a module and (*) means compile it as built-in.

Please be aware that ecdsa, paho-mqtt, python-cryptography, python-jose, python-openssl are particularly important for Google Cloud IoT demo.



```
.config - Tina Configuration
> Languages > Python
                                        Python
  Arrow keys navigate the menu.  <Enter> selects submenus ---> (or empty submenus ----).  Highlighted letters are hotkeys.  Pressing
  <Y> includes, <N> excludes, <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help, </> for Search.  Legend: [*] built-in
  [ ] excluded  <M> module  < > module capable

                               <*> ecdsa.............................................................. ecdsa
                               < > micropython................................................ Micro Python
                               < > micropython-lib......................................... micropython-lib
                               <*> paho-mqtt....................................................... paho-mqtt
                               -*- python................................... Python 2.7 programming language
                               <*> python-attrs............................................... python-attrs
                               -*- python-base........................................ Python 2.7 interpreter
                               -*- python-cffi................................................. python-cffi
                               -*- python-codecs......................... Python 2.7 codecs + unicode support
                               -*- python-compiler............................... Python 2.7 compiler module
                               < > python-crcmod............................................. python-crcmod
                               -*- python-crypto............................................. python-crypto
                               <*> python-cryptography................................. python-cryptography
                               -*- python-ctypes.................................... Python 2.7 ctypes module
                               -*- python-db............................................ Python 2.7 db module
                               -*- python-decimal.................................. Python 2.7 decimal module
                               < > python-dev................................. Python 2.7 development files
                               -*- python-distutils.................................. Python 2.7 distutils
                               < > python-dns...................................................... dnspython
                               < > python-egenix-mx-base.................................... Egenix mxBase
                               -*- python-email..................................... Python 2.7 email module
                               -*- python-enum34............................................. python-enum34
                               -*- python-gdbm....................................... Python 2.7 gdbm module
                               < > python-gmpy................................................. python-gmpy
                               < > python-gmpy2............................................... python-gmpy2
                               -*- python-idna................................................. python-idna
                               └(+)

                                 <Select>    < Exit >    < Help >    < Save >    < Load >
```

Figure 2.5 Necessary Python packages(1)



```
.config - Tina Configuration
> Languages > Python ─────────────────────────────────────────────────────────
                                       Python
  Arrow keys navigate the menu.  <Enter> selects submenus ---> (or empty submenus ----).  Highlighted letters are hotkeys.  Pressing
  <Y> includes, <N> excludes, <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help, </> for Search.  Legend: [*] built-in
  [ ] excluded  <M> module  < > module capable

                                 ^(-)
                                 < > python-imglib............................. Python Imaging Library (PIL)
                                 -*- python-ipaddress........................................ python-ipaddress
                                 <*> python-jose................................................. python-jose
                                 < > python-ldap................ Python modules for implementing LDAP clients
                                 < > python-lib2to3............................... Python 2.7 lib2to3 module
                                 -*- python-light.............................. Python 2.7 light installation
                                 -*- python-logging............................... Python 2.7 logging module
                                 -*- python-multiprocessing....................... Python 2.7 multiprocessing
                                 < > python-mysql.......................... MySQL database adapter for Python
                                 -*- python-ncurses............................... Python 2.7 ncurses module
                                 -*- python-openssl................................... Python 2.7 SSL module
                                 < > python-parsley......................................... python-parsley
                                 < > python-pip......................... Tool for installing Python packages.
                                 -*- python-ply................................................. python-ply
                                 < > python-psycopg2.................. PostgreSQL database adapter for Python
                                 -*- python-pyasn1............................................ python-pyasn1
                                 < > python-pyasn1-modules............................ python-pyasn1-modules
                                 -*- python-pycparser...................................... python-pycparser
                                 -*- python-pydoc.................................... Python 2.7 pydoc module
                                 < > python-pyopenssl...................................... python-pyopenssl
                                 < > python-pyptlib........................................ python-pyptlib
                                 < > python-pyserial........................................ python-pyserial
                                 < > python-service-identity.......................... python-service-identity
                                 -*- python-setuptools.................. Tool for installing Python packages.
                                 -*- python-six................................................. python-six
                                 < > python-smbus............................. Python bindings for the SMBUS
                                 ┴(+)

                                   <Select>    < Exit >    < Help >    < Save >    < Load >
```

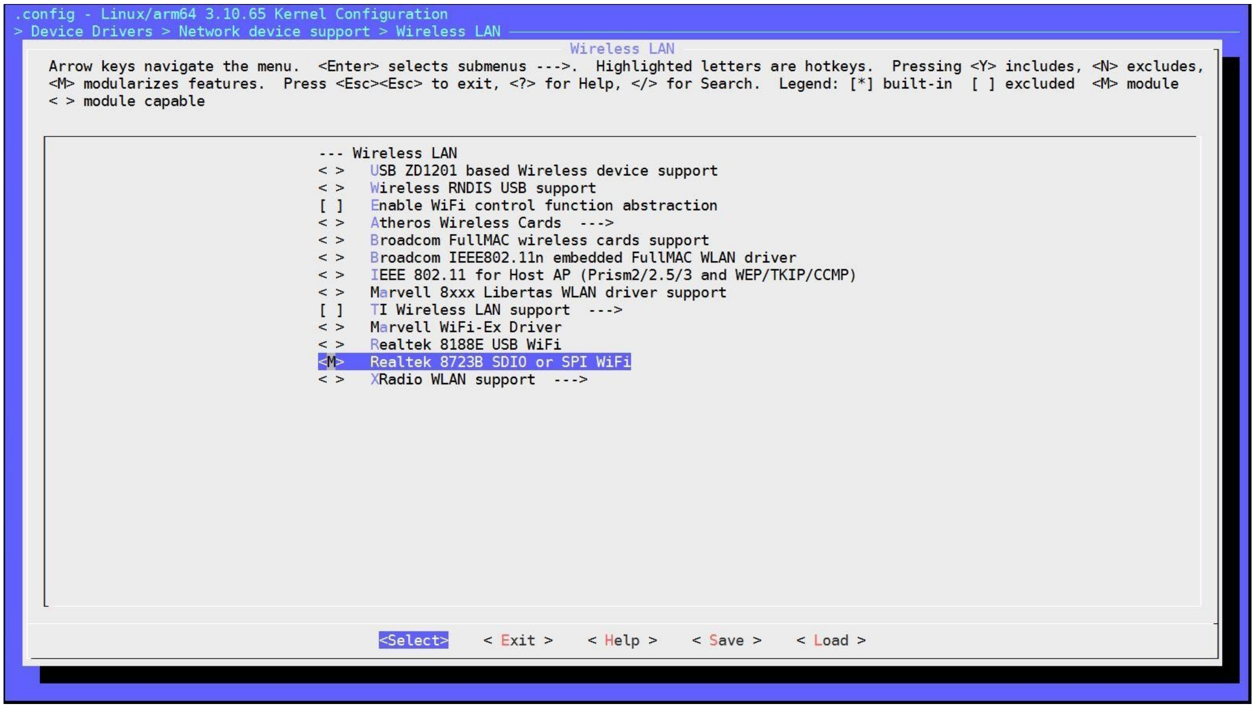
Figure 2.6 Necessary Python packages(2)

2.3 Enable wireless function on Tina

There are two Wi-Fi modules used by these three developer boards: AP6212 and RTL8723. This section will show you how to enable these modules on Tina.

The make kernel_menuconfig is a command widely used on OpenWrt systems, run it under root directory is the same as going to kernel directory and run make ARCH=arm64 menuconfig. It will pop up with a menu driven interface that lets users select kernel features and drivers that will be compiled.

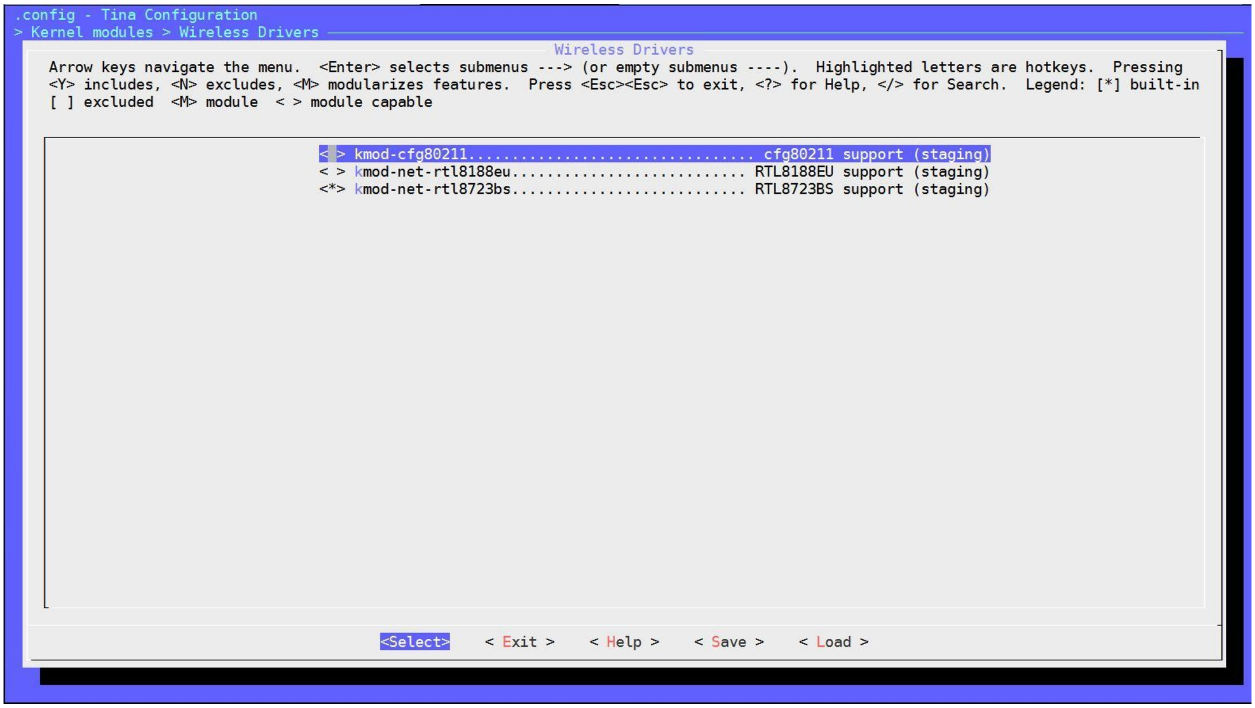
For RTL8723, step 1: select RTL8723 driver as a module from make kernel_menuconfig:



```
.config - Linux/arm64 3.10.65 Kernel Configuration
> Device Drivers > Network device support > Wireless LAN ──────────────────────────────────────────────────
                                          Wireless LAN
  Arrow keys navigate the menu.  <Enter> selects submenus --->.  Highlighted letters are hotkeys.  Pressing <Y> includes, <N> excludes,
  <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help, </> for Search.  Legend: [*] built-in  [ ] excluded  <M> module
  < > module capable

                               --- Wireless LAN
                               < >   USB ZD1201 based Wireless device support
                               < >   Wireless RNDIS USB support
                               [ ]   Enable WiFi control function abstraction
                               < >   Atheros Wireless Cards  --->
                               < >   Broadcom FullMAC wireless cards support
                               < >   Broadcom IEEE802.11n embedded FullMAC WLAN driver
                               < >   IEEE 802.11 for Host AP (Prism2/2.5/3 and WEP/TKIP/CCMP)
                               < >   Marvell 8xxx Libertas WLAN driver support
                               [ ]   TI Wireless LAN support  --->
                               < >   Marvell WiFi-Ex Driver
                               < >   Realtek 8188E USB WiFi
                               <M>   Realtek 8723B SDIO or SPI WiFi
                               < >   XRadio WLAN support  --->

                                   <Select>    < Exit >    < Help >    < Save >    < Load >
```

Figure 2.7 Select Realtek 8723 module

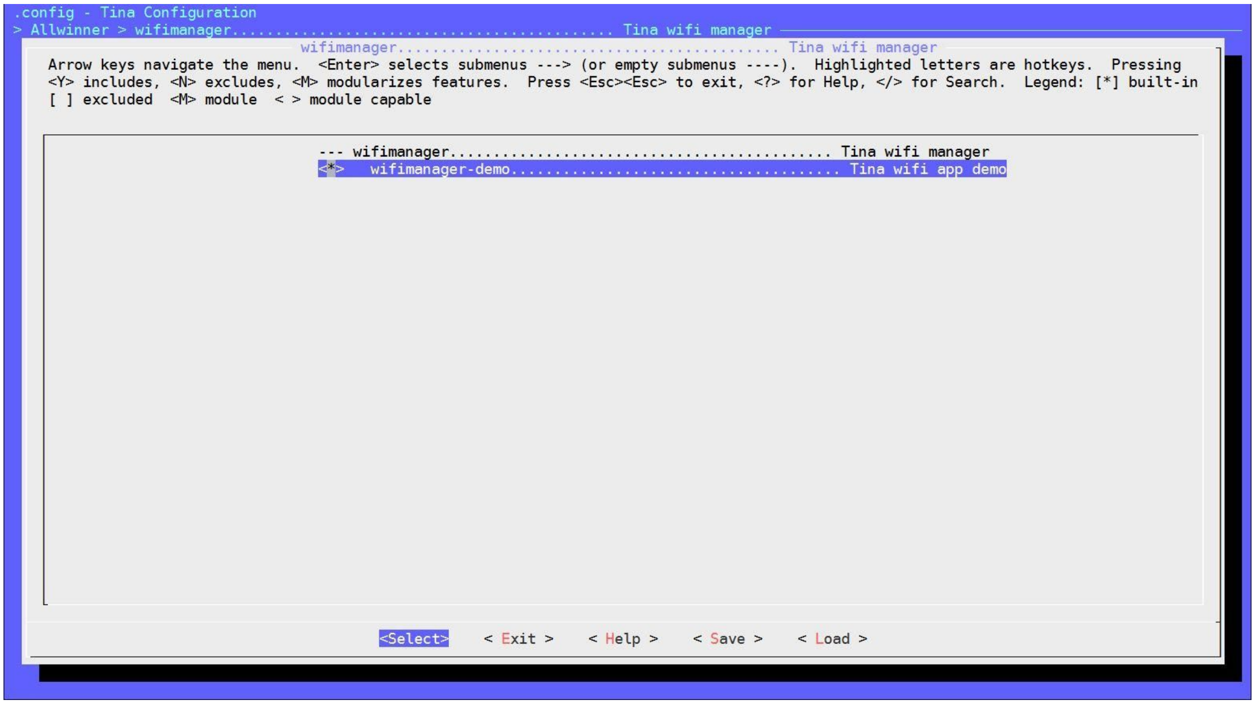
Step 2: Enter make menuconfig and select RTL8723 driver as a kernel_module.



```
.config - Tina Configuration
> Kernel modules > Wireless Drivers ──────────────────────────────────────────
                              Wireless Drivers
  Arrow keys navigate the menu.  <Enter> selects submenus ---> (or empty submenus ----).  Highlighted letters are hotkeys.  Pressing
  <Y> includes, <N> excludes, <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help, </> for Search.  Legend: [*] built-in
  [ ] excluded  <M> module  < > module capable

                              < > kmod-cfg80211................................ cfg80211 support (staging)
                              < > kmod-net-rtl8188eu.......................... RTL8188EU support (staging)
                              <*> kmod-net-rtl8723bs.......................... RTL8723BS support (staging)

                       <Select>    < Exit >    < Help >    < Save >    < Load >
```

Figure 2.8 Select wifi driver as a kernel module

Step 3: Select Wi-Fi_manager demo which is the user space application to connect to Wi-Fi via Wi-Fi module built above:

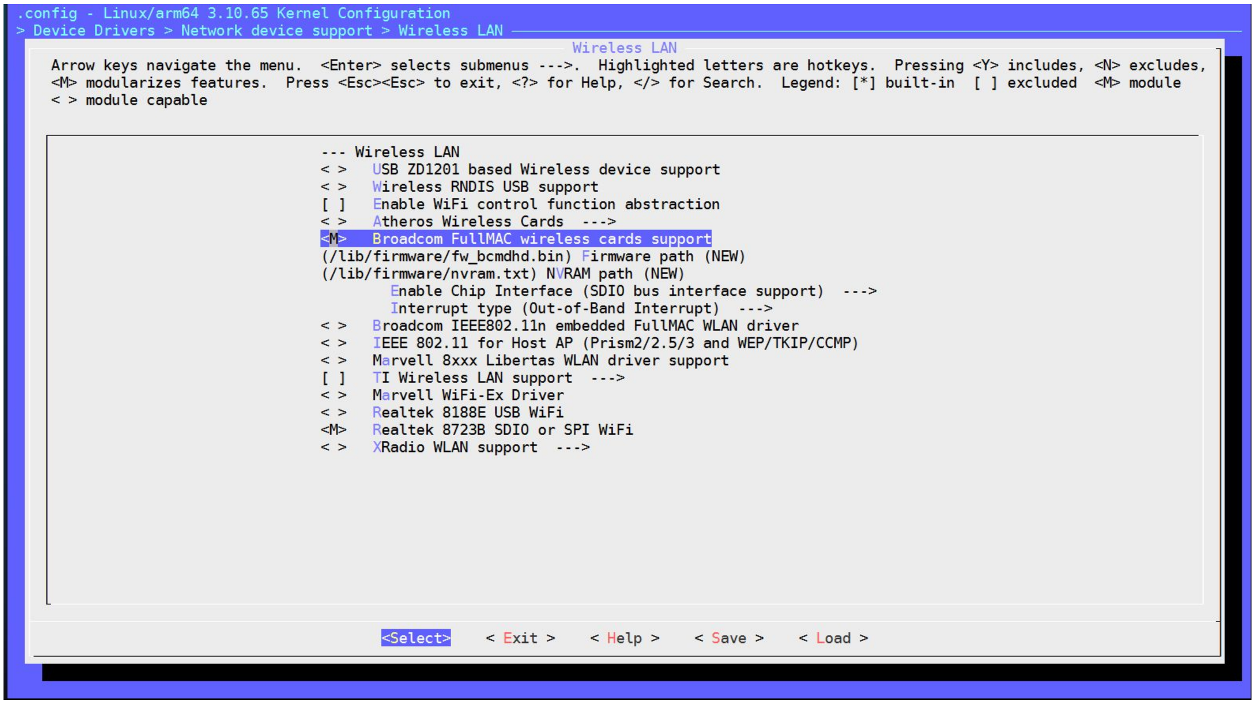


```
.config - Tina Configuration
> Allwinner > wifimanager........................................... Tina wifi manager
                        wifimanager........................................... Tina wifi manager
  Arrow keys navigate the menu.  <Enter> selects submenus ---> (or empty submenus ----).  Highlighted letters are hotkeys.  Pressing
  <Y> includes, <N> excludes, <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help, </> for Search.  Legend: [*] built-in
  [ ] excluded  <M> module  < > module capable

                              --- wifimanager.......................................... Tina wifi manager
                              <*>   wifimanager-demo.................................... Tina wifi app demo

                                  <Select>    < Exit >    < Help >    < Save >    < Load >
```

Figure 2.9 Select wifi_manager_demo

For Banana Pi M64 board, the Wi-Fi module is AP6212, please follow below instructions to compile the WLAN firmware.

Step 1: Compile "Broadcom FullMAC wireless card support" as module (M) from make kernel_menuconfig.

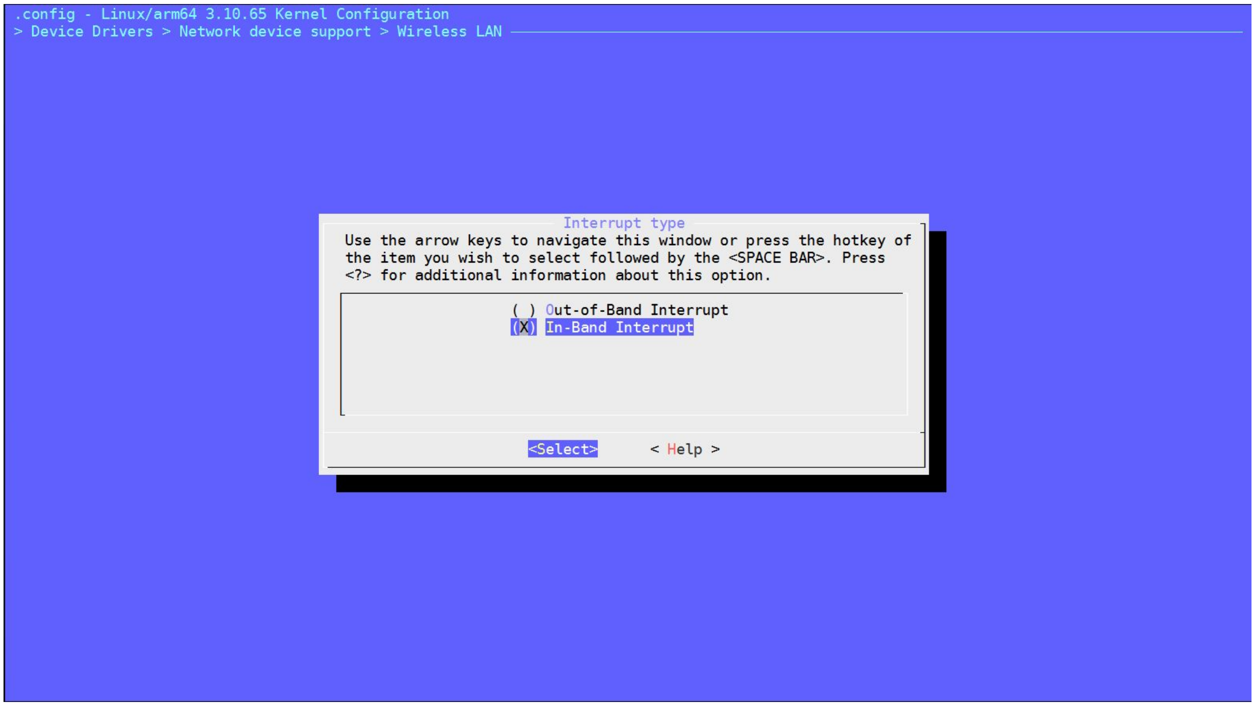


```
.config - Linux/arm64 3.10.65 Kernel Configuration
> Device Drivers > Network device support > Wireless LAN ─────────────────────────────────────────────
                                             Wireless LAN
  Arrow keys navigate the menu.  <Enter> selects submenus --->.  Highlighted letters are hotkeys.  Pressing <Y> includes, <N> excludes,
  <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help, </> for Search.  Legend: [*] built-in  [ ] excluded  <M> module
  < > module capable

                              --- Wireless LAN
                              < >   USB ZD1201 based Wireless device support
                              < >   Wireless RNDIS USB support
                              [ ]   Enable WiFi control function abstraction
                              < >   Atheros Wireless Cards  --->
                              <M>   Broadcom FullMAC wireless cards support
                              (/lib/firmware/fw_bcmdhd.bin) Firmware path (NEW)
                              (/lib/firmware/nvram.txt) NVRAM path (NEW)
                                      Enable Chip Interface (SDIO bus interface support)  --->
                                      Interrupt type (Out-of-Band Interrupt)  --->
                              < >   Broadcom IEEE802.11n embedded FullMAC WLAN driver
                              < >   IEEE 802.11 for Host AP (Prism2/2.5/3 and WEP/TKIP/CCMP)
                              < >   Marvell 8xxx Libertas WLAN driver support
                              [ ]   TI Wireless LAN support  --->
                              < >   Marvell WiFi-Ex Driver
                              < >   Realtek 8188E USB WiFi
                              <M>   Realtek 8723B SDIO or SPI WiFi
                              < >   XRadio WLAN support  --->

                  <Select>    < Exit >    < Help >    < Save >    < Load >
```

Figure 2.10 Select Broadcom FullMAC wireless cards support

Make sure that the “Firmware path” and “NVRAM path” are the same as the one on this figure.

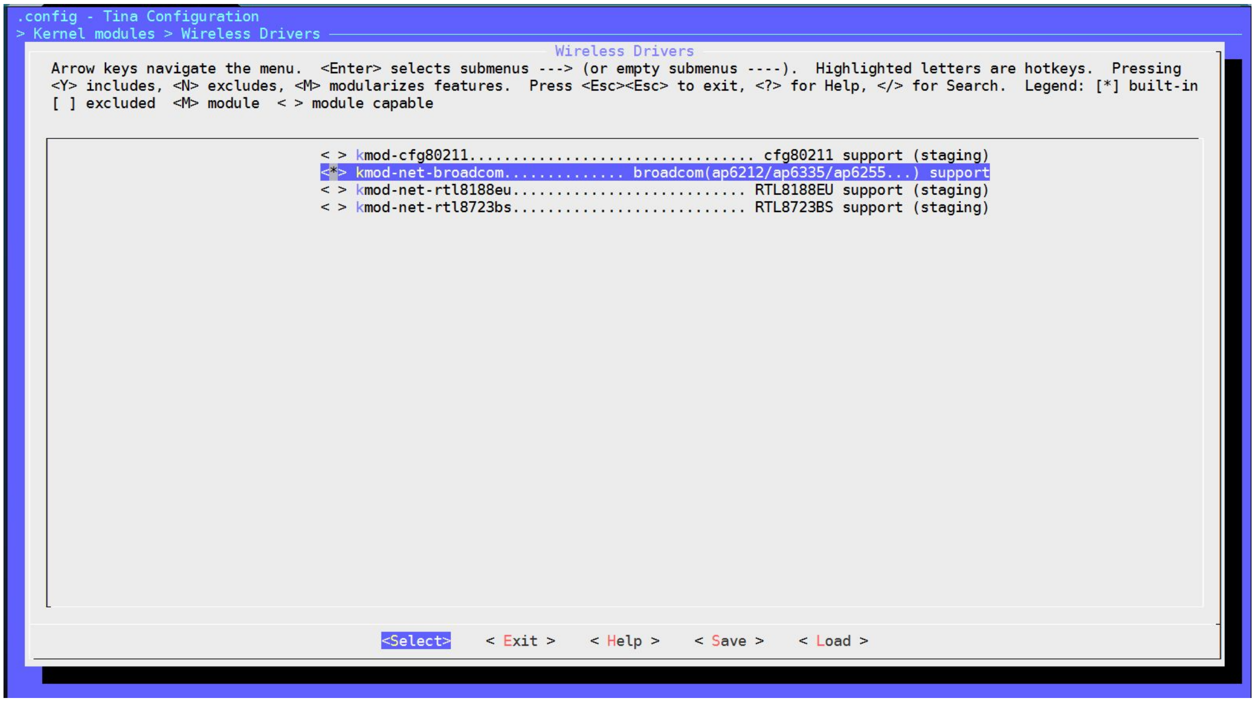
Step 2: Configure the “Interrupt type” as In-Band Interrupt.



```
.config - Linux/arm64 3.10.65 Kernel Configuration
> Device Drivers > Network device support > Wireless LAN

                              Interrupt type
  Use the arrow keys to navigate this window or press the hotkey of
  the item you wish to select followed by the <SPACE BAR>. Press
  <?> for additional information about this option.

                    ( ) Out-of-Band Interrupt
                    (X) In-Band Interrupt

                    <Select>      < Help >
```

Figure 2.11 Choose In-Band Interrupt

Step 3: Select kernel module broadcom from make menuconfig as a built-in(*)

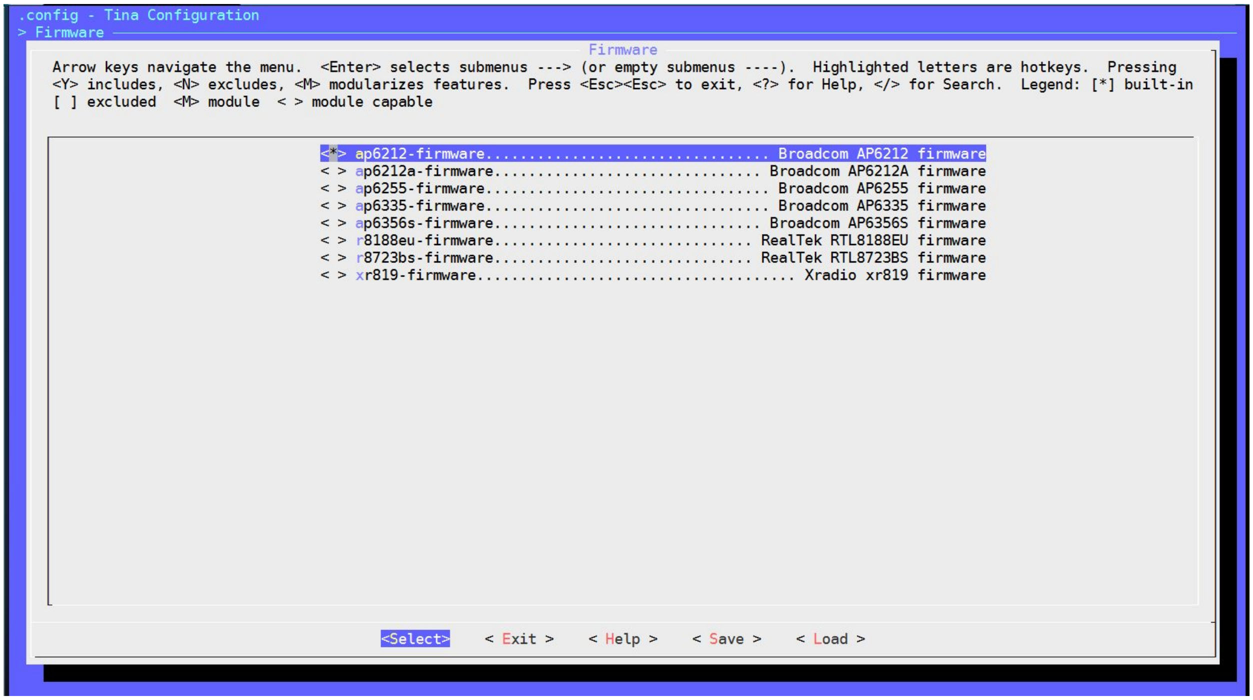


```
.config - Tina Configuration
> Kernel modules > Wireless Drivers ─────────────────────────────────────────────────────────
                                 Wireless Drivers
  Arrow keys navigate the menu.  <Enter> selects submenus ---> (or empty submenus ----).  Highlighted letters are hotkeys.  Pressing
  <Y> includes, <N> excludes, <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help, </> for Search.  Legend: [*] built-in
  [ ] excluded  <M> module  < > module capable

                                < > kmod-cfg80211................................ cfg80211 support (staging)
                                <*> kmod-net-broadcom.............. broadcom(ap6212/ap6335/ap6255...) support
                                < > kmod-net-rtl8188eu.......................... RTL8188EU support (staging)
                                < > kmod-net-rtl8723bs.......................... RTL8723BS support (staging)

                           <Select>    < Exit >    < Help >    < Save >    < Load >
```

Figure 2.12 Select Kernel module

Please only select this one if you are using AP6212, otherwise it won't work properly.

Step 4: Select AP6212-firmware as a built-in from make menuconfig



```
.config - Tina Configuration
> Firmware ─────────────────────────────────────────────────────────────────────────────────────────────────────────
                                                 Firmware
  Arrow keys navigate the menu.  <Enter> selects submenus ---> (or empty submenus ----).  Highlighted letters are hotkeys.  Pressing
  <Y> includes, <N> excludes, <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help, </> for Search.  Legend: [*] built-in
  [ ] excluded  <M> module  < > module capable

                              <*> ap6212-firmware.................................. Broadcom AP6212 firmware
                              < > ap6212a-firmware............................... Broadcom AP6212A firmware
                              < > ap6255-firmware................................ Broadcom AP6255 firmware
                              < > ap6335-firmware................................ Broadcom AP6335 firmware
                              < > ap6356s-firmware............................... Broadcom AP6356S firmware
                              < > r8188eu-firmware.............................. RealTek RTL8188EU firmware
                              < > r8723bs-firmware.............................. RealTek RTL8723BS firmware
                              < > xr819-firmware......................................... Xradio xr819 firmware

                                   <Select>    < Exit >    < Help >    < Save >    < Load >
```

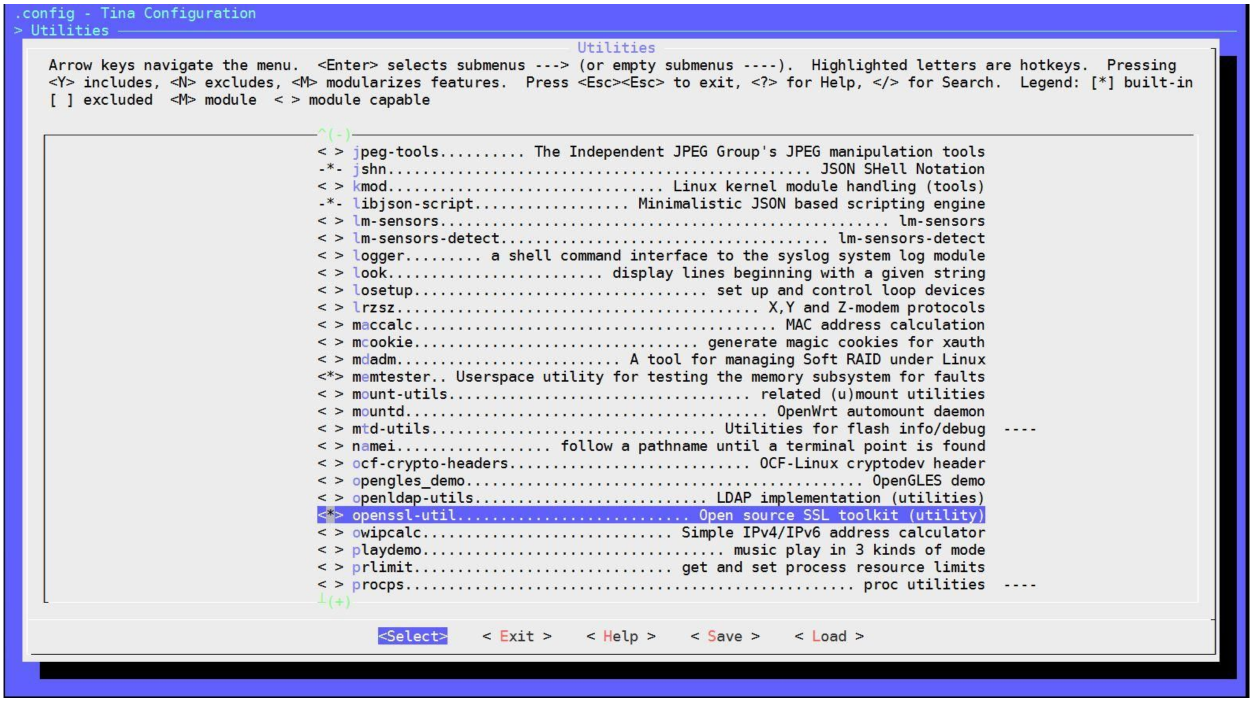
Figure 2.13 Select AP6212 firmware

Beware you have to select Wi-Fi_manager_demo(introduced in Figure 2.9) to connect to Internet with the proper commands.

2.4 Enable openssl command line tools

Google Cloud IoT requires a public/private key pair for security. You can either generate it on your PC via openssl or generate it on Tina, this section shows you where to enable the openssl command line tools.

Enter make menuconfig and select the openssl-util as a built-in(*).



```
.config - Tina Configuration
> Utilities
                                          Utilities
  Arrow keys navigate the menu.  <Enter> selects submenus ---> (or empty submenus ----).  Highlighted letters are hotkeys.  Pressing
  <Y> includes, <N> excludes, <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help, </> for Search.  Legend: [*] built-in
  [ ] excluded  <M> module  < > module capable

                                ^(-)
                                < > jpeg-tools.......... The Independent JPEG Group's JPEG manipulation tools
                                -*- jshn............................................... JSON SHell Notation
                                < > kmod................................ Linux kernel module handling (tools)
                                -*- libjson-script................. Minimalistic JSON based scripting engine
                                < > lm-sensors.................................................. lm-sensors
                                < > lm-sensors-detect.................................... lm-sensors-detect
                                < > logger......... a shell command interface to the syslog system log module
                                < > look......................... display lines beginning with a given string
                                < > losetup................................. set up and control loop devices
                                < > lrzsz......................................... X,Y and Z-modem protocols
                                < > maccalc........................................ MAC address calculation
                                < > mcookie................................ generate magic cookies for xauth
                                < > mdadm......................... A tool for managing Soft RAID under Linux
                                <*> memtester.. Userspace utility for testing the memory subsystem for faults
                                < > mount-utils................................. related (u)mount utilities
                                < > mountd........................................ OpenWrt automount daemon
                                < > mtd-utils................................ Utilities for flash info/debug  ----
                                < > namei.................. follow a pathname until a terminal point is found
                                < > ocf-crypto-headers............................ OCF-Linux cryptodev header
                                < > opengles_demo............................................ OpenGLES demo
                                < > openldap-utils........................... LDAP implementation (utilities)
                                <*> openssl-util........................... Open source SSL toolkit (utility)
                                < > owipcalc............................ Simple IPv4/IPv6 address calculator
                                < > playdemo.................................. music play in 3 kinds of mode
                                < > prlimit............................. get and set process resource limits
                                < > procps................................................. proc utilities  ----
                                └(+)

                      <Select>    < Exit >    < Help >    < Save >    < Load >
```

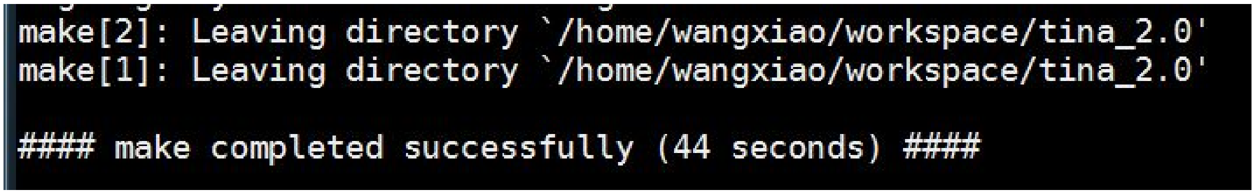
Figure 2.14 Select openssl-util

3 Compile the Tina Linux and flash the firmware on board

Now all the necessary modules for Google Cloud IoT on Tina have been selected, we can start compiling the Tina OS.

Step 1: run make -j1

Once you have successfully compiled the system, you should get the following info:



```
make[2]: Leaving directory `/home/wangxiao/workspace/tina_2.0'
make[1]: Leaving directory `/home/wangxiao/workspace/tina_2.0'

#### make completed successfully (44 seconds) ####
```

Figure 2.11 successfully compiled logging info

Step 2: run pack or pack -d command to pack the firmware.

The difference between pack and pack -d is that you use uart port or Allwinner's TF card debugger head for serial port debugging. They will pack a tina_tulip-d1_uart0.img or tina_tulip-d1_card0.img respectively.

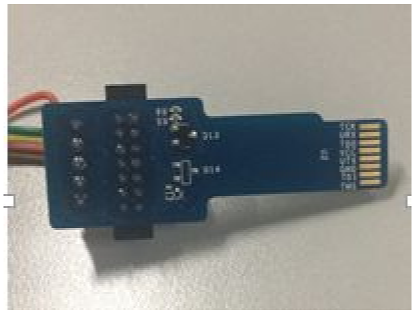
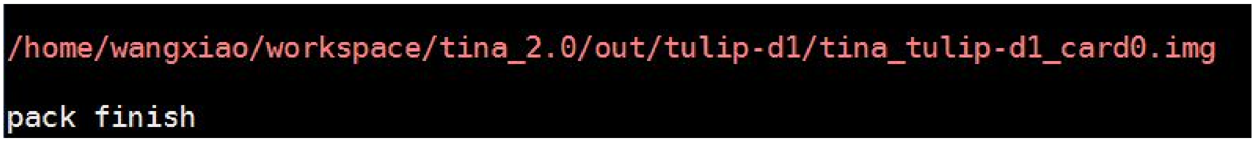


Figure 2.12 Allwinner's TF card debugger head

For example, using pack -d command which will successfully pack the firmware and will generate the log info and a firmware file under /out/tulip-d1 folder.



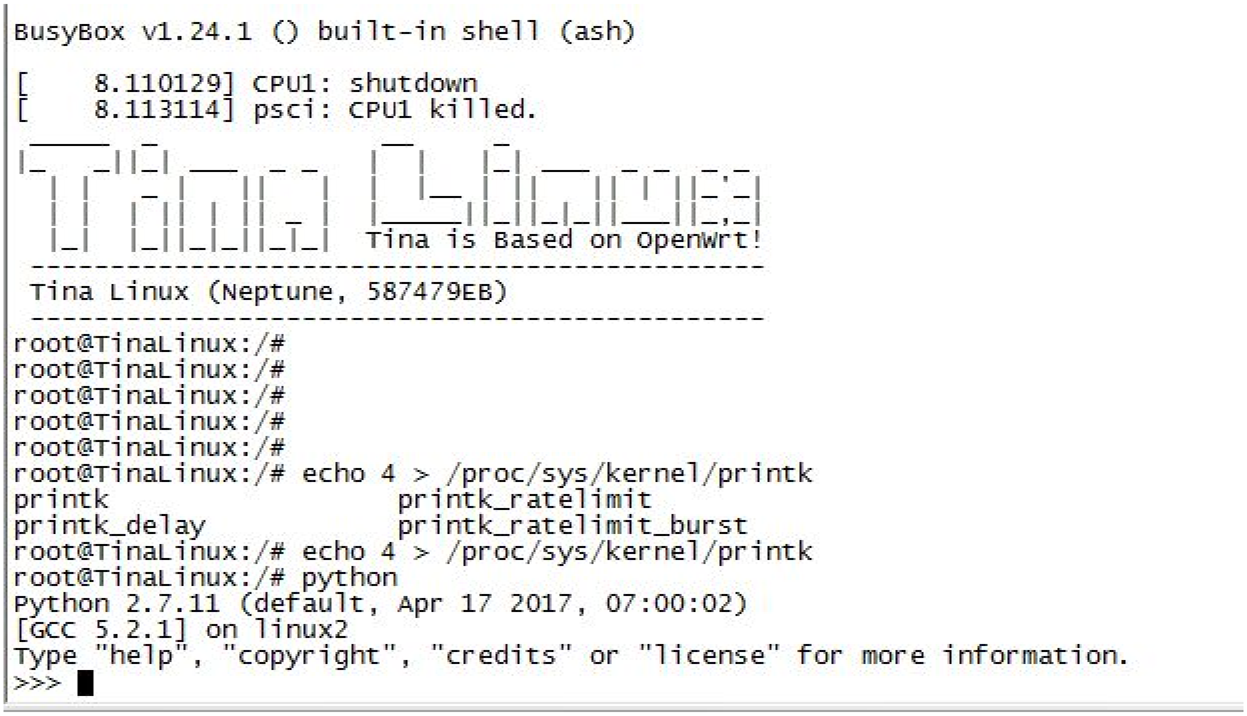
```
/home/wangxiao/workspace/tina_2.0/out/tulip-d1/tina_tulip-d1_card0.img

pack finish
```

Figure 2.13 successfully packed

To flash the firmware, please reference instructions from attached document "PhoenixSuit User Manual".

4 Run Google Cloud IoT Demo on Tina

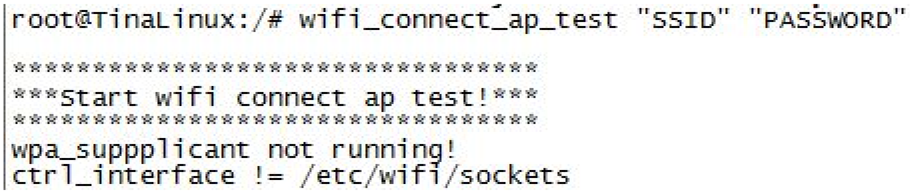
As Tina OS with Python has been brought up, in this section we'll run the google Cloud IoT demo on it.



```
BusyBox v1.24.1 () built-in shell (ash)

[    8.110129] CPU1: shutdown
[    8.113114] psci: CPU1 killed.
 _____  _              __     _
|_   _||_| ____  _ _  |  |   |_| ____  _ _  _ _
  | |   _ |    ||   | |  |__ | ||    || | ||_'_|
  | |  | || | || _  | |_____||_||_|_||___||_,_|
  |_|  |_||_|_||_|_|  Tina is Based on OpenWrt!
 ----------------------------------------------
 Tina Linux (Neptune, 587479EB)
 ----------------------------------------------
root@TinaLinux:/#
root@TinaLinux:/#
root@TinaLinux:/#
root@TinaLinux:/#
root@TinaLinux:/#
root@TinaLinux:/# echo 4 > /proc/sys/kernel/printk
printk                  printk_ratelimit
printk_delay            printk_ratelimit_burst
root@TinaLinux:/# echo 4 > /proc/sys/kernel/printk
root@TinaLinux:/# python
Python 2.7.11 (default, Apr 17 2017, 07:00:02)
[GCC 5.2.1] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> █
```

Figure 2.14 Serial console and Python output

Step 1: Connect to Internet. Run wifi_connect_ap_test "SSID""PASSWORD", fill in the corresponding Wi-Fi SSID and Password.



```
root@TinaLinux:/# wifi_connect_ap_test "SSID" "PASSWORD"

*************************************
***Start wifi connect ap test!***
*************************************
wpa_suppplicant not running!
ctrl_interface != /etc/wifi/sockets
```

Figure 2.15 Connect to Intenet

Step 2: Follow the instructions of MQTT Client samples on Page 24 CloudIotAlpha-UserGuide. Don't forget to create the pub/sub subscription to the verity that you are receiving device telemetry events.

Like Cloud IoT Alpha- User Guide , the next step assumes that the following has been created:

A project called allwinner-164401

A registry called: my-registry-id using the Pub/Sub topic
projects/allwinner-164401/topics/device-events

One device called: my-rs256-device-id

A RSA certificate and private key called rsa_cert.pem and rsa_private.pem respectively

Issue the commands:

adb push rsa_private.pem (This pushes rsa_private.pem file to the board)

adb push roots.pem (This pushes the roots.pem file to the board)

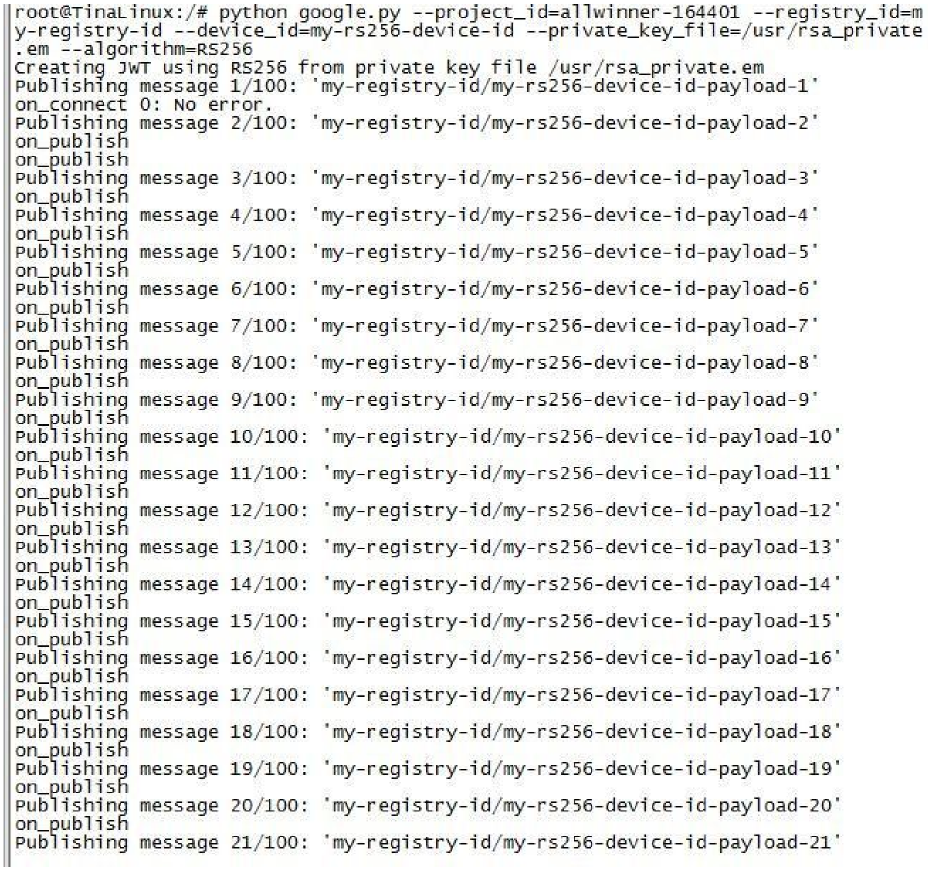
Edit the cloudiot_mqtt_example.py file:

Replace the the line "import jwt" with "from jose import jwt"

Step 3: Run the Google Cloud IoT demo, substituting in your project name, registry id and device id:

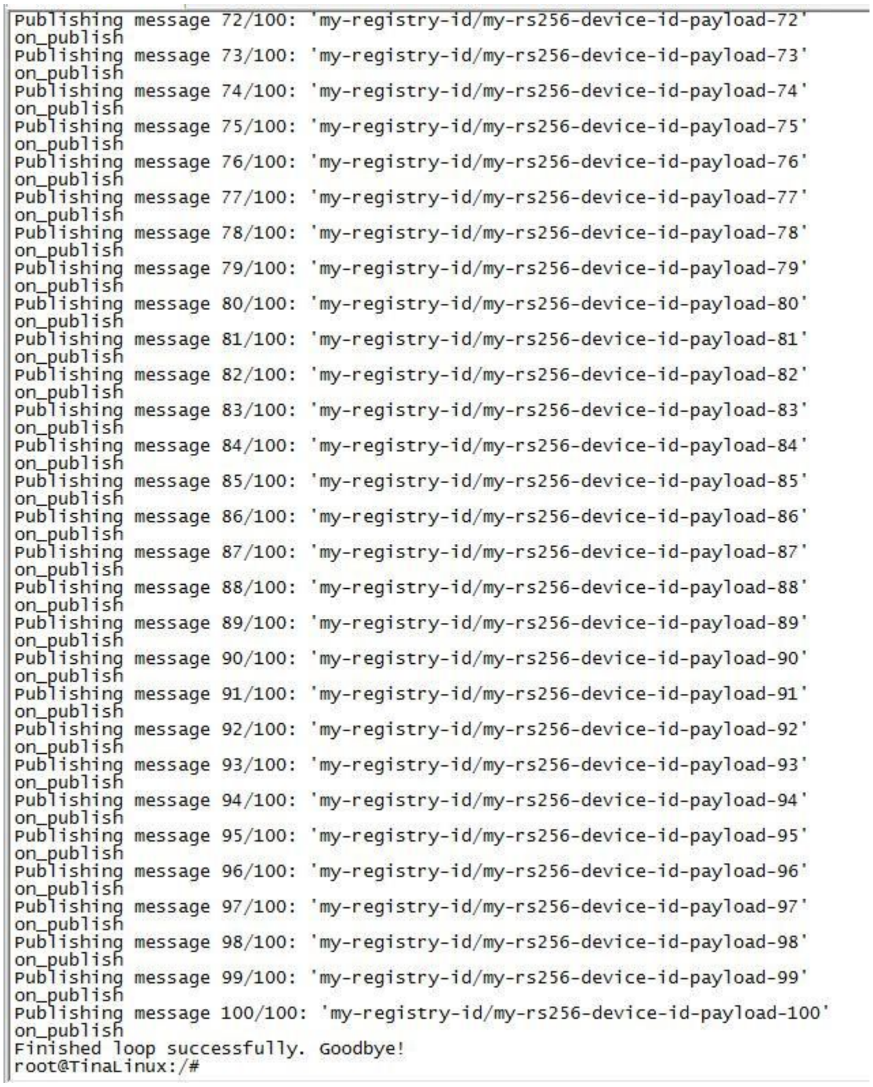
For example, in this format:

```
python cloudiot_mqtt_example.py  --project_id=my-iot-project-id \
 --registry_id=my-registry-id \
--device_id=my-device-id \
 --private_key_file=rsa_private.pem \
--algorithm=RS256 \
```



```
root@TinaLinux:/# python google.py --project_id=allwinner-164401 --registry_id=m
y-registry-id --device_id=my-rs256-device-id --private_key_file=/usr/rsa_private
.em --algorithm=RS256
Creating JWT using RS256 from private key file /usr/rsa_private.em
Publishing message 1/100: 'my-registry-id/my-rs256-device-id-payload-1'
on_connect 0: No error.
Publishing message 2/100: 'my-registry-id/my-rs256-device-id-payload-2'
on_publish
on_publish
Publishing message 3/100: 'my-registry-id/my-rs256-device-id-payload-3'
on_publish
Publishing message 4/100: 'my-registry-id/my-rs256-device-id-payload-4'
on_publish
Publishing message 5/100: 'my-registry-id/my-rs256-device-id-payload-5'
on_publish
Publishing message 6/100: 'my-registry-id/my-rs256-device-id-payload-6'
on_publish
Publishing message 7/100: 'my-registry-id/my-rs256-device-id-payload-7'
on_publish
Publishing message 8/100: 'my-registry-id/my-rs256-device-id-payload-8'
on_publish
Publishing message 9/100: 'my-registry-id/my-rs256-device-id-payload-9'
on_publish
Publishing message 10/100: 'my-registry-id/my-rs256-device-id-payload-10'
on_publish
Publishing message 11/100: 'my-registry-id/my-rs256-device-id-payload-11'
on_publish
Publishing message 12/100: 'my-registry-id/my-rs256-device-id-payload-12'
on_publish
Publishing message 13/100: 'my-registry-id/my-rs256-device-id-payload-13'
on_publish
Publishing message 14/100: 'my-registry-id/my-rs256-device-id-payload-14'
on_publish
Publishing message 15/100: 'my-registry-id/my-rs256-device-id-payload-15'
on_publish
Publishing message 16/100: 'my-registry-id/my-rs256-device-id-payload-16'
on_publish
Publishing message 17/100: 'my-registry-id/my-rs256-device-id-payload-17'
on_publish
Publishing message 18/100: 'my-registry-id/my-rs256-device-id-payload-18'
on_publish
Publishing message 19/100: 'my-registry-id/my-rs256-device-id-payload-19'
on_publish
Publishing message 20/100: 'my-registry-id/my-rs256-device-id-payload-20'
on_publish
Publishing message 21/100: 'my-registry-id/my-rs256-device-id-payload-21'
```

Figure 2.16 Demo log(1)



```
Publishing message 72/100: 'my-registry-id/my-rs256-device-id-payload-72'
on_publish
Publishing message 73/100: 'my-registry-id/my-rs256-device-id-payload-73'
on_publish
Publishing message 74/100: 'my-registry-id/my-rs256-device-id-payload-74'
on_publish
Publishing message 75/100: 'my-registry-id/my-rs256-device-id-payload-75'
on_publish
Publishing message 76/100: 'my-registry-id/my-rs256-device-id-payload-76'
on_publish
Publishing message 77/100: 'my-registry-id/my-rs256-device-id-payload-77'
on_publish
Publishing message 78/100: 'my-registry-id/my-rs256-device-id-payload-78'
on_publish
Publishing message 79/100: 'my-registry-id/my-rs256-device-id-payload-79'
on_publish
Publishing message 80/100: 'my-registry-id/my-rs256-device-id-payload-80'
on_publish
Publishing message 81/100: 'my-registry-id/my-rs256-device-id-payload-81'
on_publish
Publishing message 82/100: 'my-registry-id/my-rs256-device-id-payload-82'
on_publish
Publishing message 83/100: 'my-registry-id/my-rs256-device-id-payload-83'
on_publish
Publishing message 84/100: 'my-registry-id/my-rs256-device-id-payload-84'
on_publish
Publishing message 85/100: 'my-registry-id/my-rs256-device-id-payload-85'
on_publish
Publishing message 86/100: 'my-registry-id/my-rs256-device-id-payload-86'
on_publish
Publishing message 87/100: 'my-registry-id/my-rs256-device-id-payload-87'
on_publish
Publishing message 88/100: 'my-registry-id/my-rs256-device-id-payload-88'
on_publish
Publishing message 89/100: 'my-registry-id/my-rs256-device-id-payload-89'
on_publish
Publishing message 90/100: 'my-registry-id/my-rs256-device-id-payload-90'
on_publish
Publishing message 91/100: 'my-registry-id/my-rs256-device-id-payload-91'
on_publish
Publishing message 92/100: 'my-registry-id/my-rs256-device-id-payload-92'
on_publish
Publishing message 93/100: 'my-registry-id/my-rs256-device-id-payload-93'
on_publish
Publishing message 94/100: 'my-registry-id/my-rs256-device-id-payload-94'
on_publish
Publishing message 95/100: 'my-registry-id/my-rs256-device-id-payload-95'
on_publish
Publishing message 96/100: 'my-registry-id/my-rs256-device-id-payload-96'
on_publish
Publishing message 97/100: 'my-registry-id/my-rs256-device-id-payload-97'
on_publish
Publishing message 98/100: 'my-registry-id/my-rs256-device-id-payload-98'
on_publish
Publishing message 99/100: 'my-registry-id/my-rs256-device-id-payload-99'
on_publish
Publishing message 100/100: 'my-registry-id/my-rs256-device-id-payload-100'
on_publish
Finished loop successfully. Goodbye!
root@TinaLinux:/#
```

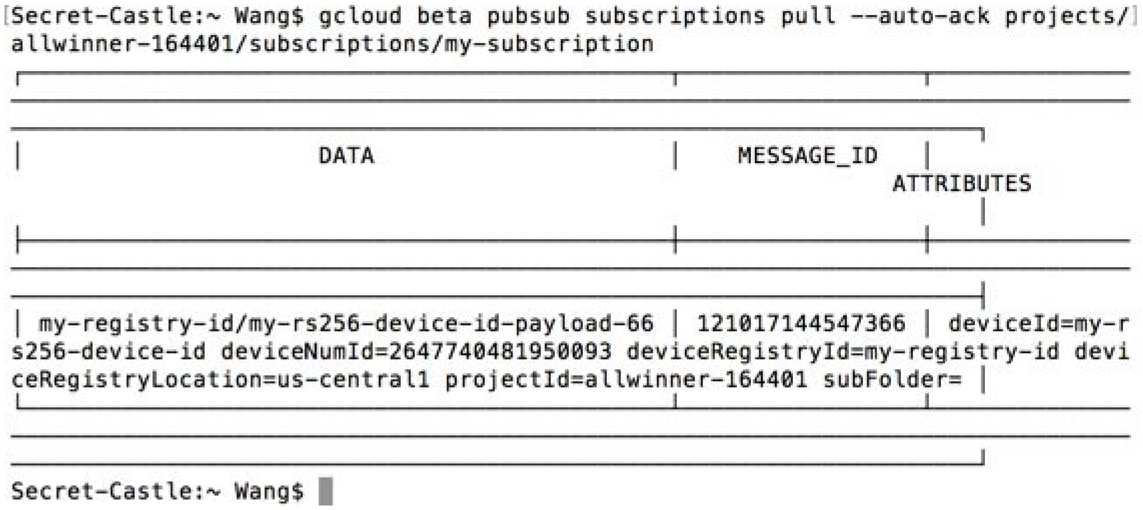
Figure 2.17 Demo Log(2)

Step 4: Once you have your MQTT client sending data to the endpoint, you can verify that the data made it to your Pub/Sub topic with

gcloud beta pubsub subscriptions pull --auto-ack \

projects/allwinner-164401/subscriptions/my-subscription \

on your Google Cloud IoT SDK.



```
[Secret-Castle:~ Wang$ gcloud beta pubsub subscriptions pull --auto-ack projects/]
 allwinner-164401/subscriptions/my-subscription
 ┌─────────────────────────────────────────┬───────────────┬─────────────
─────────────────────────────────────────────────────────────────────────
─────────────────────────────────────────────────────────────┐
 │                  DATA                   │   MESSAGE_ID   │
                                                     ATTRIBUTES
                                                             │
 ├─────────────────────────────────────────┼───────────────┼─────────────
─────────────────────────────────────────────────────────────────────────
─────────────────────────────────────────────────────────────┤
 │ my-registry-id/my-rs256-device-id-payload-66 │ 121017144547366 │ deviceId=my-r
s256-device-id deviceNumId=2647740481950093 deviceRegistryId=my-registry-id devi
ceRegistryLocation=us-central1 projectId=allwinner-164401 subFolder= │
 └─────────────────────────────────────────┴───────────────┴─────────────
─────────────────────────────────────────────────────────────────────────
─────────────────────────────────────────────────────────────┘
Secret-Castle:~ Wang$ █
```

Figure 2.18 Subscriptions captured data

References

[1] “Cloud IoT (Alpha) - User Guide”, handrei@, eschapira@, mkess@,malter@, indchak@google.com02/20/2017

roots.pem	PhoenixSuit User Manual V1.0 2(