USER'S MANUAL

Agent200

Linux

Software User's Manual

AXIOMTEK

www.axiomtek.com

Disclaimers

Trademarks Acknowledgments

Table of Contents

Section 1 Introduction

The extreme compact AGENT200 supports the low power RISC-based module (i.MX6UL) processor with extended temperature range of -40°C to +70°C for using in wide range operating environments. Multiple built-in serial ports, high-speed LANs and USB 2.0 ports enable fast and efficient data computation, communication and acquisition. Its digital I/O feature provides users with the convenience of digital devices connection. Besides, its compact size with Din-rail mounting allows for easy installation into control.

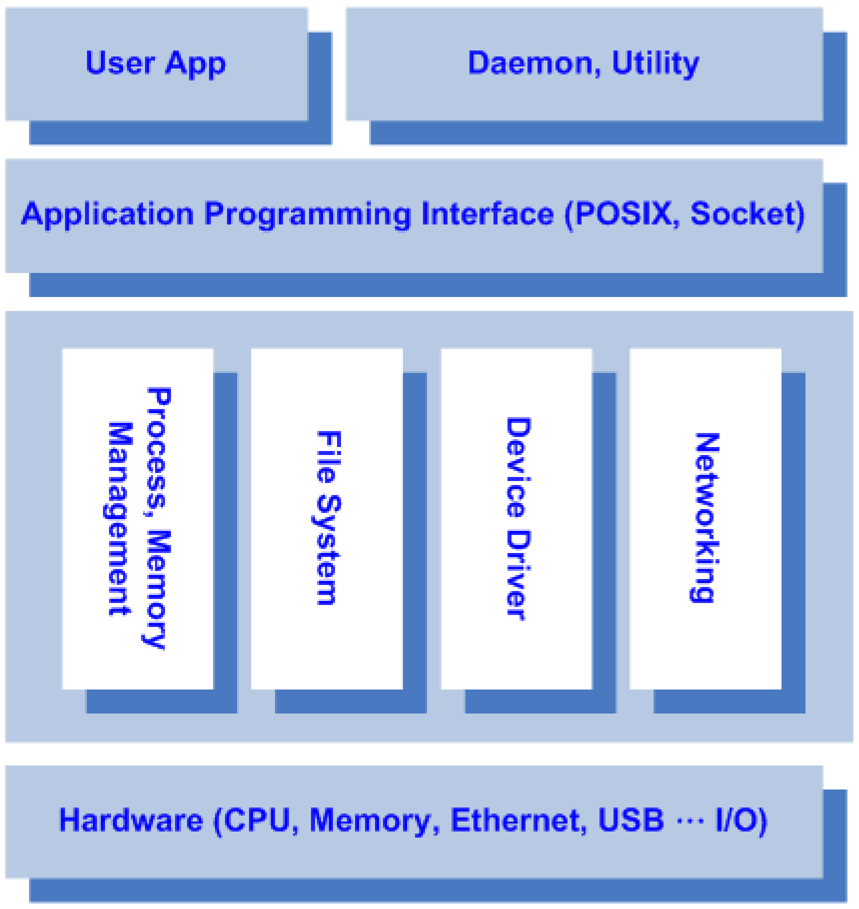
This user's manual is for the embedded Linux preinstalled in AGENT200. The embedded Linux is derived from Linux Yocto Board Support Package, which is based on Linux Kernel 4.14.98 and our hardware patches to suit AGENT200.

Software structure
The preinstalled embedded Linux image is located in eMMC Flash memory which is partitioned and formatted to accommodate boot loader, kernel and root filesystem. It follows standard Linux architecture to allow user to easily develop and deploy application software that follows Portable Operating System Interface (POSIX).

To facilitate user program in monitoring and controlling I/O device such as DIO, Watchdog Timer, CAN, COM, LED, USB power, buzzer, the AGENT200 includes 'ax_service' install file.

For connectivity, this image includes most popular internet protocols, some servers and utilities not only making it easy for downloading/uploading files (Linux kernel, application program) or for debugging, but also communicating to outside world via Ethernet, WiFi and 4G.

For the convenience of manipulating embedded Linux, this image includes lots of popular packages such as busybox, udev, etc.


User App
Daemon, Utility
Application Programming Interface (POSIX, Socket)
Process, Memory Management
File System
Device Driver
Networking
Hardware (CPU, Memory, Ethernet, USB ··· I/O)

1.1 Specifications

OS	Yocto Project 2.5.3 Sumo
Kernel	Version : 4.14.98 (with NXP and Axiomtek hardware modified patch)
Busybox	Vesion : 1.29.3, a collection of standard Linux command-line utilities
storage format	Support FAT32 /FAT/EXT2/EXT3/EXT4
Shell	Bash
BSP	AGENT200-LINUX-bsp ● Image ● Toolchain
Protocol type	Support ICMP, TCP/IP, UDP, DHCP, Telnet, HTTP, HTTPS, SSL, SMTP, NTP, DNS, PPP, PPPoE, FTP, TFTP, NFS
Daemons	
Telnetd	Telnet server daemon
Ftpd	FTP server daemon
Pppd	Point-to-point protocol
Utilities	
Telnet	Telnet client program

FTP	FTP client program
TFTP	Trivial File Transfer Protocol client
Udev	A device manager for Linux kernel
Dosfstools	Utilities for making and checking MS-DOS FAT file system
E2fsprogs	A set of utilities for maintaining the ex2,ext3 and ext4 file systems
Ethtool	A Linux command for displaying or modifying the Network Interface Controller (NIC) parameters
I2c-tools	A heterogeneous set of i2c tools for Linux
Procps	Utilities to report on the state of the system, including the states of running processes and amount of memory
Iperf3	Network performance measurement tool
Xinetd	Manages internet-based connectivity
Openssh	Based on SSH protocol for remotely controlling or transferring files
Openssh-sftp	Secure File Transfer Protocol
Ntp	Network Time Protocol, used to synchronize time
Wvdial	Point-to-Point Protocol dialer
Curl	Transfer data tool
Python	Version : 2.7, Python development environment Version : 3.5, Python development environment
Development Environment	
Host OS/development	Host OS/ development OS: Ubuntu 14.04 LTS 64bit
Kernel	Version : 4.2.0-42
Toolchain	ARM, gcc-4.8.4 (Yocto project 2.5.3 Sumo)
Machine running Ubuntu: the minimum hard disk space required is about 50 GB for the X11 backend. It is recommended that at least 120 GB is provided in order to have sufficient space to compile all backends together.	
Hardware's Service	
Comport	RS-232/422/485 mode setting(Default RS232)
Digital I/O	Read digital input Write digital output
LED	Control User definition LED
CAN	Set CAN Bus Bitrate
USB Power	Disable/Enable USB Power
Watch Dog Timer	Enable Watch Dog Timer Set Timer
WiFi (Optional)	Use Wi-Fi module WPET-236ACN(BT)
4G (Optional)	Use 4G module Quectel EC25-AU

Note

1. ***All specifications and images are subject to change without notice.***
2. ***Command definition:***

Command	Definition	Example
=>	**U-Boot**	Ex: **=>** setenv ipaddr 192.168.1.103 Meaning: U-Boot setenv ipaddr 192.168.1.103
~$	**Host PC**	Ex: **~$** sudo apt-get install subversion Meaning: To command sudo apt-get install subverhsion on host PC
~#	**Target (AGENT200):**	Ex: ~# /etc/run_rescue Meaning: To command /etc/run_rescue on AGENT200

Section 2
Getting Started

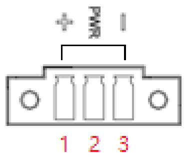
2.1 Connecting the AGENT200

The power

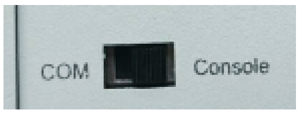
Please check you power as below:

1. DC input range 9~48V
2. DC Terminal Block

Pin	DC Signal Name
1	Power+
2	N/A
3	Power-



The console
Connect your computer to the AGENT200 via a serial cable and change the switch to Console mode (as below)



You can connect the AGENT200 to personal computer (PC) in two ways:

- Serial RS-232 console
- SSH over Ethernet

Note

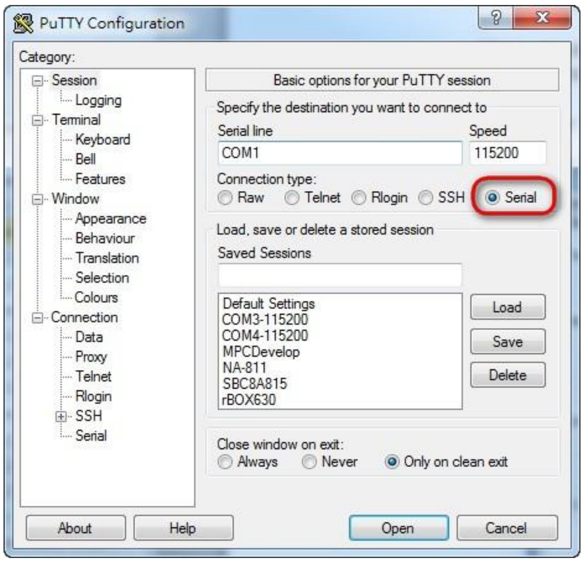
2.1.1 Serial Console

The serial console is a convenient interface for connecting AGENT200 to PC. First of all, it is very important to make sure that your desktop connects to AGENT200 by console cable. Please set the system as follows:

Baudrate: 115200 bps
Parity: None
Data bits: 8
Stop bit: 1
Flow Control: None

Here we use PuTTY to setup and link to the AGENT200. Learn how to do it with these step by step instructions:

1. Open PuTTY and choose 'Serial' as the connection type.


PuTTY Configuration

Basic options for your PuTTY session

Specify the destination you want to connect to

Serial line: COM1
Speed: 115200

Connection type: Raw Telnet Rlogin SSH [✓ Serial]

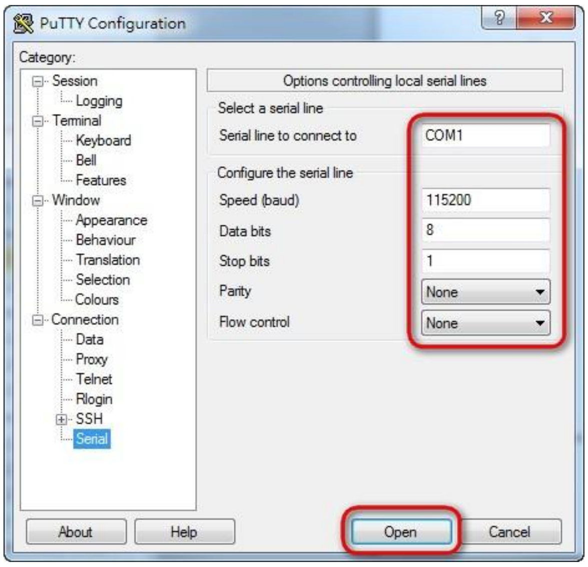
Load, save or delete a stored session

Saved Sessions

Default Settings
COM3-115200
COM4-115200
MPCDevelop
NA-811
SBC8A815
rBOX630

Close window on exit: Always Never [✓ Only on clean exit]

2. Configure the serial port correctly (see image below). Click Open and power on the AGENT200.


PuTTY Configuration

Options controlling local serial lines

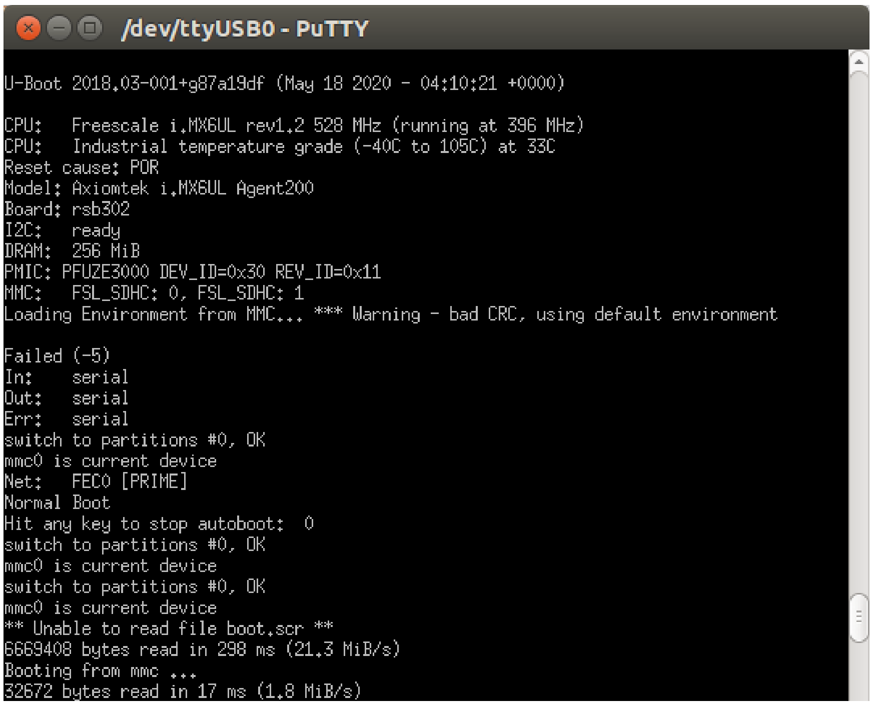
Select a serial line

Serial line to connect to: COM1

Configure the serial line

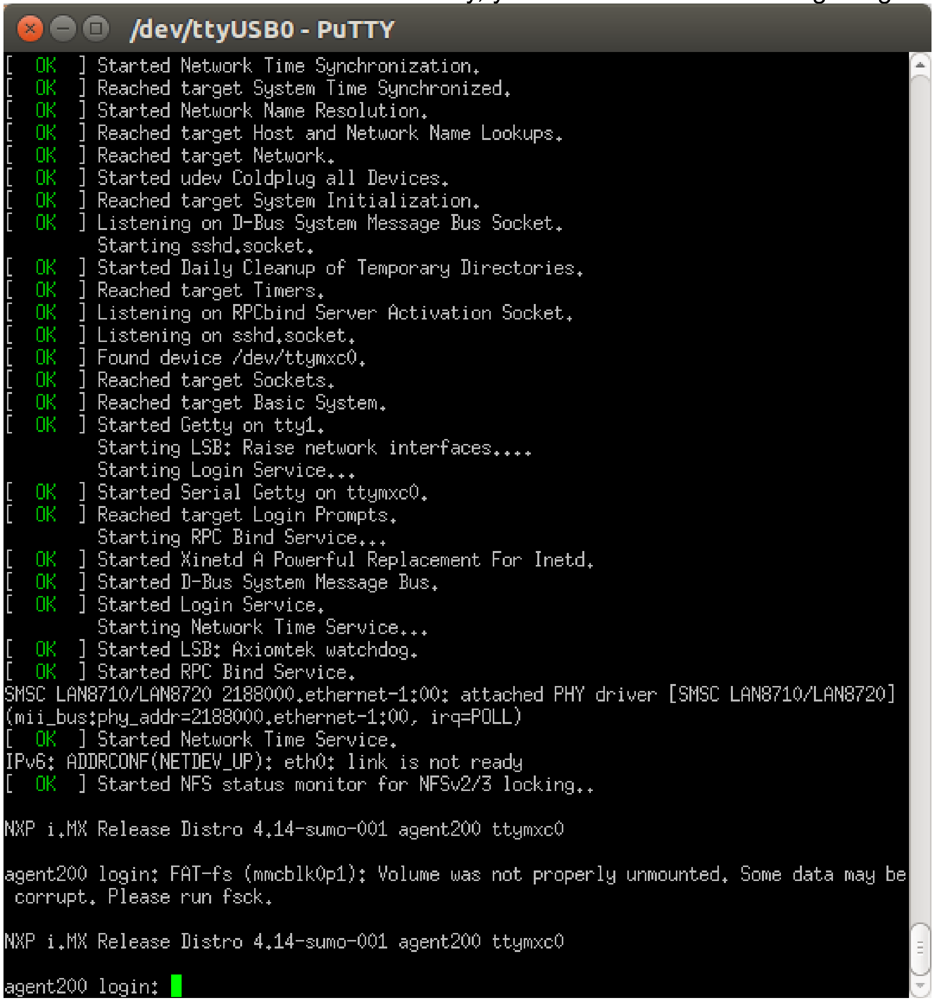
Speed (baud): 115200
Data bits: 8
Stop bits: 1
Parity: None
Flow control: None

3. The Bootloader default booting system from eMMC.



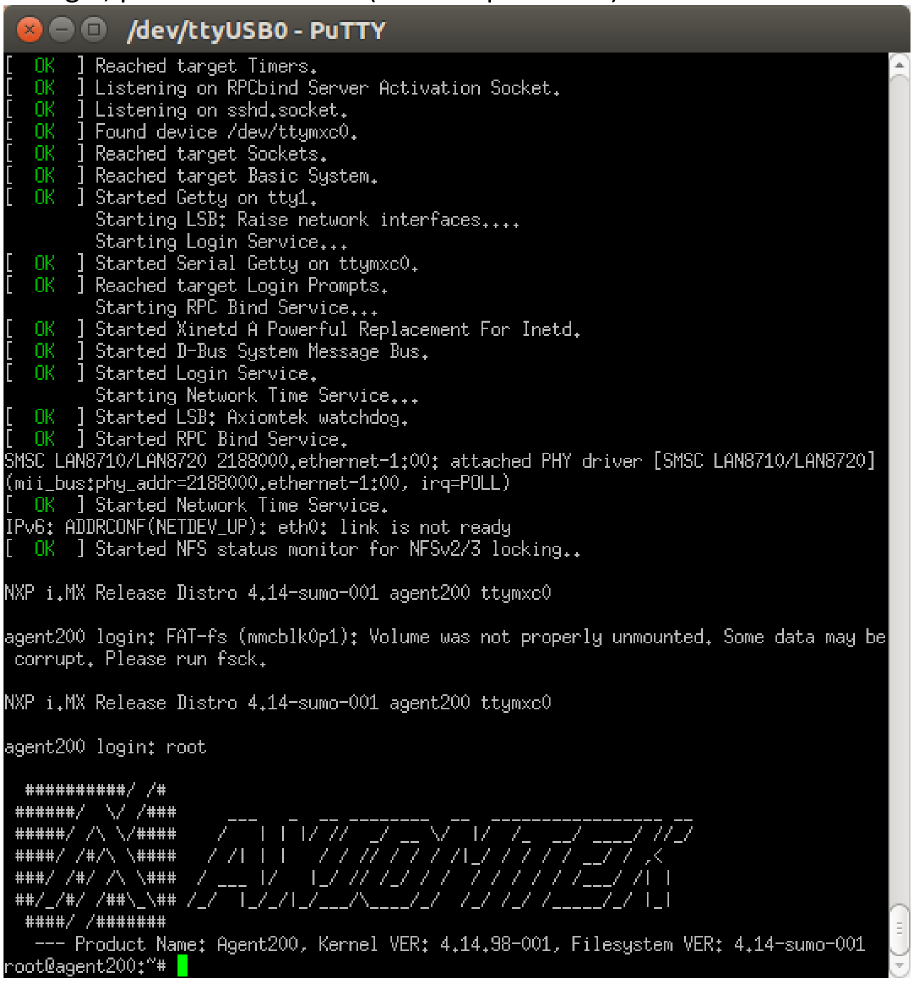
```
U-Boot 2018.03-001+g87a19df (May 18 2020 - 04:10:21 +0000)

CPU:   Freescale i.MX6UL rev1.2 528 MHz (running at 396 MHz)
CPU:   Industrial temperature grade (-40C to 105C) at 33C
Reset cause: POR
Model: Axiomtek i.MX6UL Agent200
Board: rsb302
I2C:   ready
DRAM:  256 MiB
PMIC: PFUZE3000 DEV_ID=0x30 REV_ID=0x11
MMC:   FSL_SDHC: 0, FSL_SDHC: 1
Loading Environment from MMC... *** Warning - bad CRC, using default environment

Failed (-5)
In:    serial
Out:   serial
Err:   serial
switch to partitions #0, OK
mmc0 is current device
Net:   FEC0 [PRIME]
Normal Boot
Hit any key to stop autoboot:  0
switch to partitions #0, OK
mmc0 is current device
switch to partitions #0, OK
mmc0 is current device
** Unable to read file boot.scr **
6669408 bytes read in 298 ms (21.3 MiB/s)
Booting from mmc ...
32672 bytes read in 17 ms (1.8 MiB/s)
```

4. If connection is established successfully, you should see the following image.



```
[  OK  ] Started Network Time Synchronization.
[  OK  ] Reached target System Time Synchronized.
[  OK  ] Started Network Name Resolution.
[  OK  ] Reached target Host and Network Name Lookups.
[  OK  ] Reached target Network.
[  OK  ] Started udev Coldplug all Devices.
[  OK  ] Reached target System Initialization.
[  OK  ] Listening on D-Bus System Message Bus Socket.
         Starting sshd.socket.
[  OK  ] Started Daily Cleanup of Temporary Directories.
[  OK  ] Reached target Timers.
[  OK  ] Listening on RPCbind Server Activation Socket.
[  OK  ] Listening on sshd.socket.
[  OK  ] Found device /dev/ttymxc0.
[  OK  ] Reached target Sockets.
[  OK  ] Reached target Basic System.
[  OK  ] Started Getty on tty1.
         Starting LSB: Raise network interfaces....
         Starting Login Service...
[  OK  ] Started Serial Getty on ttymxc0.
[  OK  ] Reached target Login Prompts.
         Starting RPC Bind Service...
[  OK  ] Started Xinetd A Powerful Replacement For Inetd.
[  OK  ] Started D-Bus System Message Bus.
[  OK  ] Started Login Service.
         Starting Network Time Service...
[  OK  ] Started LSB: Axiomtek watchdog.
[  OK  ] Started RPC Bind Service.
SMSC LAN8710/LAN8720 2188000.ethernet-1:00: attached PHY driver [SMSC LAN8710/LAN8720]
(mii_bus:phy_addr=2188000.ethernet-1:00, irq=POLL)
[  OK  ] Started Network Time Service.
IPv6: ADDRCONF(NETDEV_UP): eth0: link is not ready
[  OK  ] Started NFS status monitor for NFSv2/3 locking..

NXP i.MX Release Distro 4.14-sumo-001 agent200 ttymxc0

agent200 login: FAT-fs (mmcblk0p1): Volume was not properly unmounted. Some data may be
 corrupt. Please run fsck.

NXP i.MX Release Distro 4.14-sumo-001 agent200 ttymxc0

agent200 login:
```

5. To login, please enter 'root' (without password).

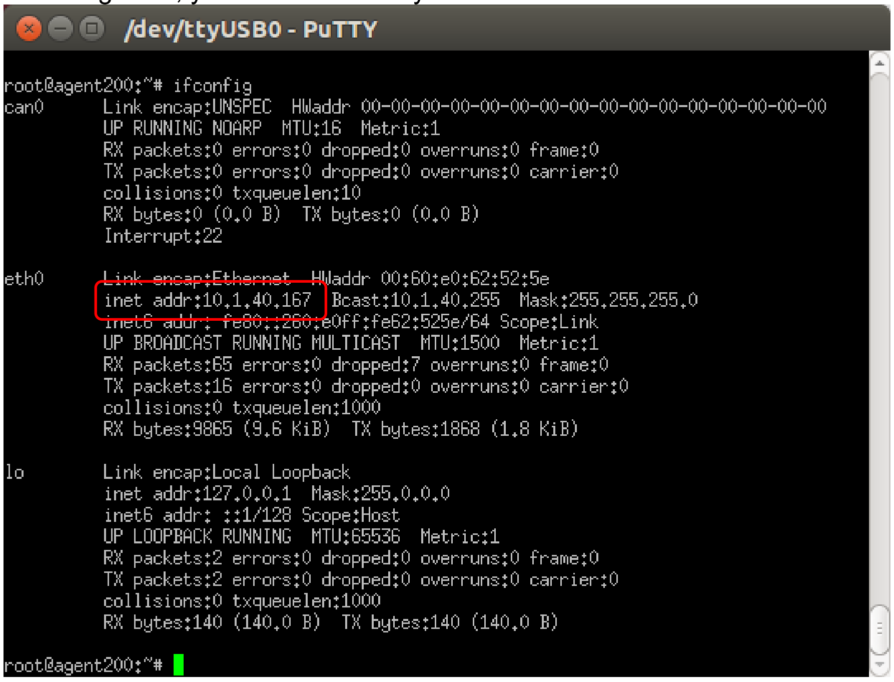


```
[  OK  ] Reached target Timers.
[  OK  ] Listening on RPCbind Server Activation Socket.
[  OK  ] Listening on sshd.socket.
[  OK  ] Found device /dev/ttymxc0.
[  OK  ] Reached target Sockets.
[  OK  ] Reached target Basic System.
[  OK  ] Started Getty on tty1.
         Starting LSB: Raise network interfaces....
         Starting Login Service...
[  OK  ] Started Serial Getty on ttymxc0.
[  OK  ] Reached target Login Prompts.
         Starting RPC Bind Service...
[  OK  ] Started Xinetd A Powerful Replacement For Inetd.
[  OK  ] Started D-Bus System Message Bus.
[  OK  ] Started Login Service.
         Starting Network Time Service...
[  OK  ] Started LSB: Axiomtek watchdog.
[  OK  ] Started RPC Bind Service.
SMSC LAN8710/LAN8720 2188000.ethernet-1:00: attached PHY driver [SMSC LAN8710/LAN8720]
(mii_bus:phy_addr=2188000.ethernet-1:00, irq=POLL)
[  OK  ] Started Network Time Service.
IPv6: ADDRCONF(NETDEV_UP): eth0: link is not ready
[  OK  ] Started NFS status monitor for NFSv2/3 locking..

NXP i.MX Release Distro 4.14-sumo-001 agent200 ttymxc0

agent200 login: FAT-fs (mmcblk0p1): Volume was not properly unmounted. Some data may be
 corrupt. Please run fsck.

NXP i.MX Release Distro 4.14-sumo-001 agent200 ttymxc0

agent200 login: root

  ##########/ /#
 ######/  \/ /###     ___   _  __ ________  __  _________________ __
 #####/ /\ \/####    /   | | |/ //  _/ __ \/  |/  /_  __/ ____/ //_/
 ####/ /#/\ \####   / /| | |   / / // / / / /|_/ / / / / __/ / ,<
 ###/ /#/ /\ \###  / ___ |/   |_/ // /_/ / /  / / / / / /___/ /| |
 ##/_/#/ /##\_\## /_/  |_/_/|_/___/\____/_/  /_/ /_/ /_____/_/ |_|
  ####/ /#######
   --- Product Name: Agent200, Kernel VER: 4.14.98-001, Filesystem VER: 4.14-sumo-001
root@agent200:~#
```

2.1.2 SSH over Ethernet

Now, we are going to connect the AGENT200 to PC over Ethernet. The following illustrations show how to do it under Windows® and Linux environment.

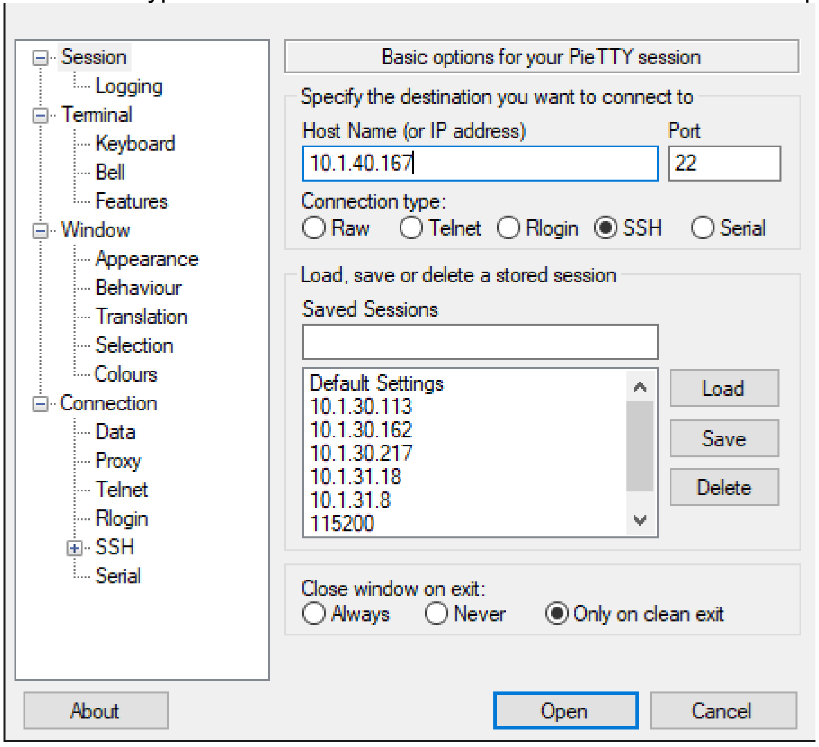
Before starting SSH, you have to check your LAN1 IP address.



```
root@agent200:~# ifconfig
can0      Link encap:UNSPEC  HWaddr 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00
          UP RUNNING NOARP  MTU:16  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:10
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
          Interrupt:22

eth0      Link encap:Ethernet  HWaddr 00:60:e0:62:52:5e
          inet addr:10.1.40.167  Bcast:10.1.40.255  Mask:255.255.255.0
          inet6 addr: fe80::260:e0ff:fe62:525e/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:65 errors:0 dropped:7 overruns:0 frame:0
          TX packets:16 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:9865 (9.6 KiB)  TX bytes:1868 (1.8 KiB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:2 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:140 (140.0 B)  TX bytes:140 (140.0 B)

root@agent200:~#
```

For Windows® users:

1. Here we also use PuTTY to setup and link. Open PuTTY and choose 'SSH' as the connection type. Then set the IP address to 10.1.40.167 and click Open.



Basic options for your PieTTY session

Specify the destination you want to connect to

Host Name (or IP address): 10.1.40.167 Port: 22

Connection type: Raw Telnet Rlogin [✓ SSH] Serial

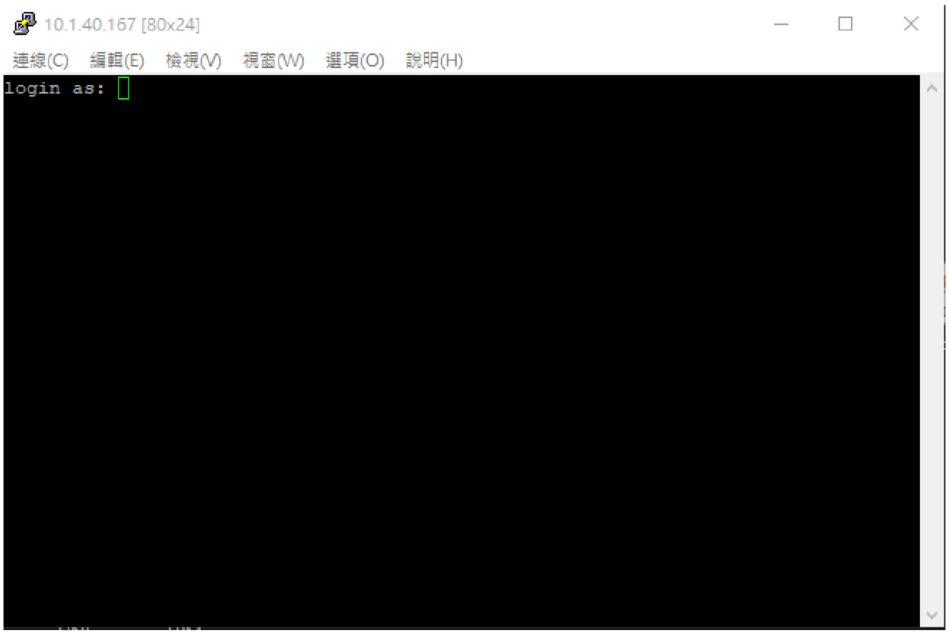
Load, save or delete a stored session

Saved Sessions

Default Settings
10.1.30.113
10.1.30.162
10.1.30.217
10.1.31.18
10.1.31.8
115200

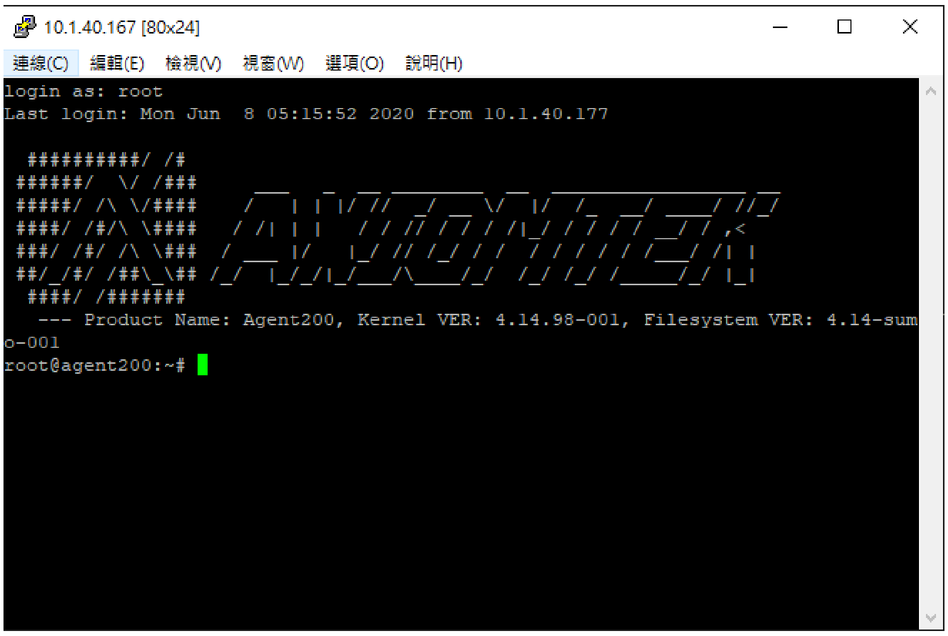
Close window on exit: Always Never [✓ Only on clean exit]

2. If connection is established successfully, you should see the following image.



```
login as: 
```

3. To login AGENT200, please enter 'root' (with no password).

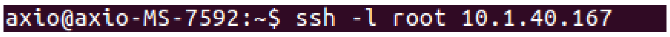


```
login as: root
Last login: Mon Jun  8 05:15:52 2020 from 10.1.40.177

  ##########/ /#
 ######/  \/ /###     ____  _  __ _______  __  _______________ __
 #####/ /\ \/####    /    | | |/ //  _/ __ \/  |/  /_  __/ ____/ //_/
 ####/ /#/\ \####   / /| | |   / / // / / / /|_/ / / / / __/ / ,<
 ###/ /#/ /\ \###  / ___ |/   |_/ // /_/ / /  / / / / / /___/ /| |
 ##/_/#/ /##\_\## /_/  |_/_/|_/___/\____/_/  /_/ /_/ /_____/_/ |_|
  ####/ /#######
   --- Product Name: Agent200, Kernel VER: 4.14.98-001, Filesystem VER: 4.14-sum
o-001
root@agent200:~# 
```

For Linux users:

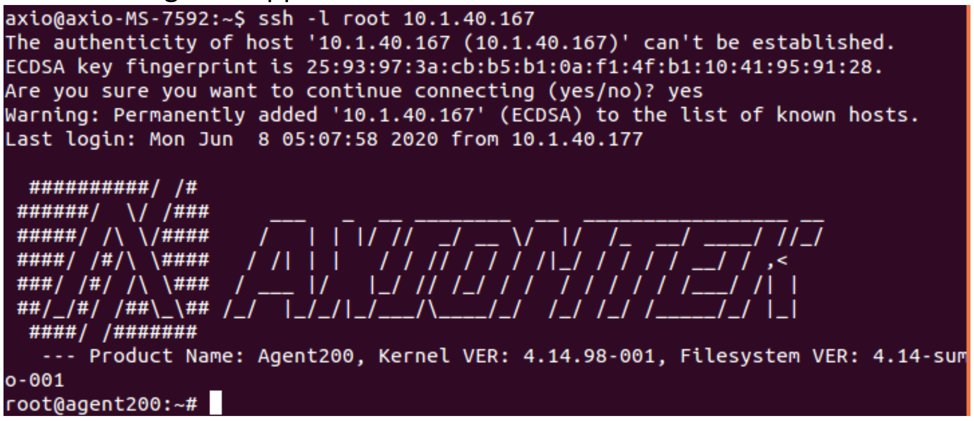
1. Open terminal and keyin 'ssh' command.

   ~$ ssh -l root 10.1.40.167



```
axio@axio-MS-7592:~$ ssh -l root 10.1.40.167
```

2. The following data appears after the connection is established successful.



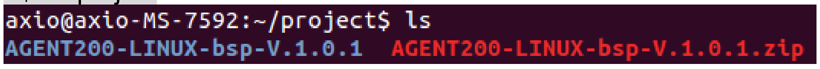
```
axio@axio-MS-7592:~$ ssh -l root 10.1.40.167
The authenticity of host '10.1.40.167 (10.1.40.167)' can't be established.
ECDSA key fingerprint is 25:93:97:3a:cb:b5:b1:0a:f1:4f:b1:10:41:95:91:28.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.1.40.167' (ECDSA) to the list of known hosts.
Last login: Mon Jun  8 05:07:58 2020 from 10.1.40.177

  ##########/ /#
 ######/  \/ /###     ___   _ __ ________ __  _________________ __
 #####/ /\ \/####    /   | | |/ //  _/ __ \/  |/  /_  __/ ____/ //_/
 ####/ /#/\ \####   / /| | |   / / // / / / /|_/ / / / / / __/ / ,<
 ###/ /#/ /\ \###  / ___ |/   |_/ // /_/ / /  / / / / / / /___/ /| |
 ##/_/#/ /##\_\## /_/  |_/_/|_/___/\____/_/  /_/ /_/ /_____/_/ |_|
  ####/ /#######
   --- Product Name: Agent200, Kernel VER: 4.14.98-001, Filesystem VER: 4.14-sum
o-001
root@agent200:~#
```

2.2 How to Develop a Sample Program

In this section, learn how to develop a sample program for AGENT200 with the following step by step instructions. The sample program is named 'hello.c'.

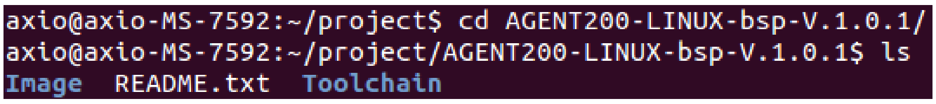
1. To Create a directory for AGENT200 BSP, and copy AGENT200-Linux-bsp-x.x.x.zip to here

   ~$ mkdir project
   ~$ cd project



```
axio@axio-MS-7592:~/project$ ls
AGENT200-LINUX-bsp-V.1.0.1  AGENT200-LINUX-bsp-V.1.0.1.zip
```

2. After extracted the file, you will find a directory AGENT200 Linux V.x.x.x



```
axio@axio-MS-7592:~/project$ cd AGENT200-LINUX-bsp-V.1.0.1/
axio@axio-MS-7592:~/project/AGENT200-LINUX-bsp-V.1.0.1$ ls
Image  README.txt  Toolchain
```

Note

Image : This directory include kernel, rootfilesystem
Toolchain : This directory include cross compiler toolchain build from Yocto Project 2.5.3
README.txt : This BSP's documentation file

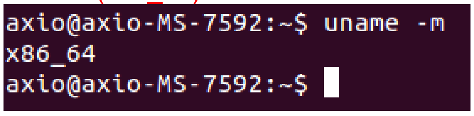
2.2.1 Install Yocto Toolchain

Before you develop and compile sample program, you should install Yocto toolchain into development PC. You can follow below step to install Yocto toolchain or refer to Chapter 5 Board Support Package to build the toolchain for AGENT200.

.

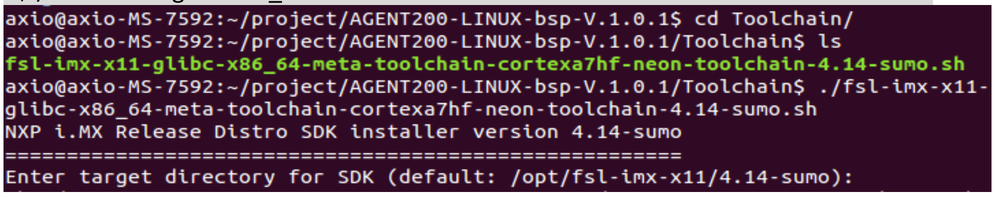
1. To check your Ubuntu version on your host PC.
   ~$ uname -m

Ubuntu **64-bit** (x86_64):



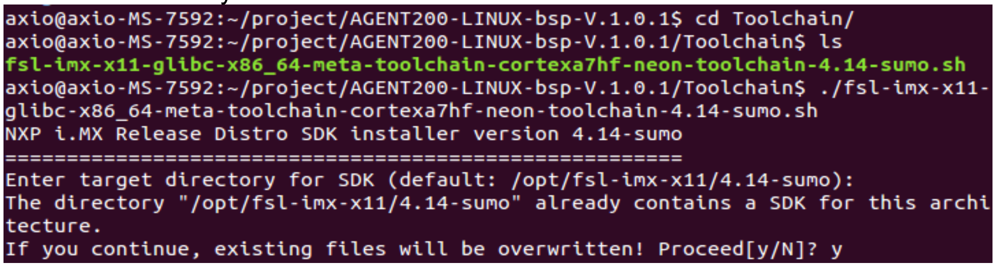
```
axio@axio-MS-7592:~$ uname -m
x86_64
axio@axio-MS-7592:~$
```

2. Copy the toolchain script to home directory, and execute the toolchain script and press Enter to install to default directory.
   64-bit machines:
   ~$./fsl-imx-x11-glibc-x86_64-meta-toolchain-cortexa7hf-neno-toolchain-4.14-sumo.sh



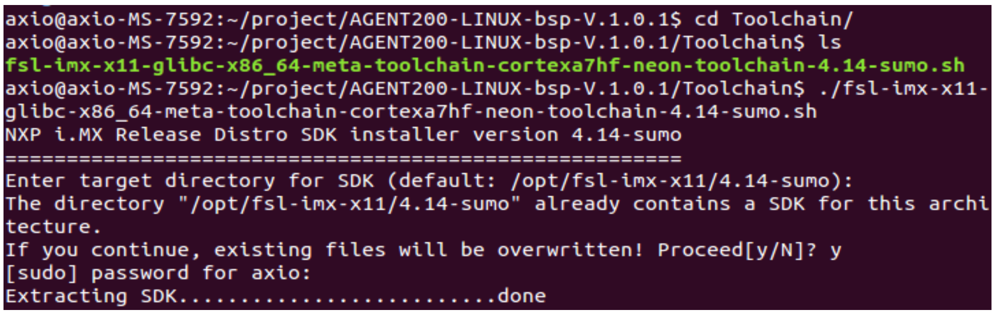
```
axio@axio-MS-7592:~/project/AGENT200-LINUX-bsp-V.1.0.1$ cd Toolchain/
axio@axio-MS-7592:~/project/AGENT200-LINUX-bsp-V.1.0.1/Toolchain$ ls
fsl-imx-x11-glibc-x86_64-meta-toolchain-cortexa7hf-neon-toolchain-4.14-sumo.sh
axio@axio-MS-7592:~/project/AGENT200-LINUX-bsp-V.1.0.1/Toolchain$ ./fsl-imx-x11-
glibc-x86_64-meta-toolchain-cortexa7hf-neon-toolchain-4.14-sumo.sh
NXP i.MX Release Distro SDK installer version 4.14-sumo
=======================================================
Enter target directory for SDK (default: /opt/fsl-imx-x11/4.14-sumo):
```

3. Check the directory.



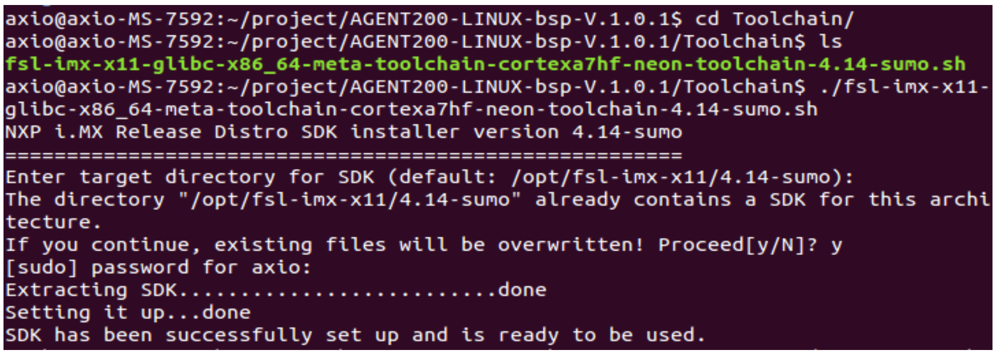
```
axio@axio-MS-7592:~/project/AGENT200-LINUX-bsp-V.1.0.1$ cd Toolchain/
axio@axio-MS-7592:~/project/AGENT200-LINUX-bsp-V.1.0.1/Toolchain$ ls
fsl-imx-x11-glibc-x86_64-meta-toolchain-cortexa7hf-neon-toolchain-4.14-sumo.sh
axio@axio-MS-7592:~/project/AGENT200-LINUX-bsp-V.1.0.1/Toolchain$ ./fsl-imx-x11-
glibc-x86_64-meta-toolchain-cortexa7hf-neon-toolchain-4.14-sumo.sh
NXP i.MX Release Distro SDK installer version 4.14-sumo
=======================================================
Enter target directory for SDK (default: /opt/fsl-imx-x11/4.14-sumo):
The directory "/opt/fsl-imx-x11/4.14-sumo" already contains a SDK for this archi
tecture.
If you continue, existing files will be overwritten! Proceed[y/N]? y
```

4. Wait to installation



```
axio@axio-MS-7592:~/project/AGENT200-LINUX-bsp-V.1.0.1$ cd Toolchain/
axio@axio-MS-7592:~/project/AGENT200-LINUX-bsp-V.1.0.1/Toolchain$ ls
fsl-imx-x11-glibc-x86_64-meta-toolchain-cortexa7hf-neon-toolchain-4.14-sumo.sh
axio@axio-MS-7592:~/project/AGENT200-LINUX-bsp-V.1.0.1/Toolchain$ ./fsl-imx-x11-
glibc-x86_64-meta-toolchain-cortexa7hf-neon-toolchain-4.14-sumo.sh
NXP i.MX Release Distro SDK installer version 4.14-sumo
=======================================================
Enter target directory for SDK (default: /opt/fsl-imx-x11/4.14-sumo):
The directory "/opt/fsl-imx-x11/4.14-sumo" already contains a SDK for this archi
tecture.
If you continue, existing files will be overwritten! Proceed[y/N]? y
[sudo] password for axio:
Extracting SDK.........................done
```

5. Install finish

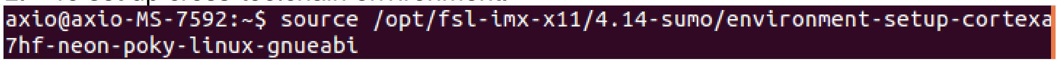


```
axio@axio-MS-7592:~/project/AGENT200-LINUX-bsp-V.1.0.1$ cd Toolchain/
axio@axio-MS-7592:~/project/AGENT200-LINUX-bsp-V.1.0.1/Toolchain$ ls
fsl-imx-x11-glibc-x86_64-meta-toolchain-cortexa7hf-neon-toolchain-4.14-sumo.sh
axio@axio-MS-7592:~/project/AGENT200-LINUX-bsp-V.1.0.1/Toolchain$ ./fsl-imx-x11-
glibc-x86_64-meta-toolchain-cortexa7hf-neon-toolchain-4.14-sumo.sh
NXP i.MX Release Distro SDK installer version 4.14-sumo
=======================================================
Enter target directory for SDK (default: /opt/fsl-imx-x11/4.14-sumo):
The directory "/opt/fsl-imx-x11/4.14-sumo" already contains a SDK for this archi
tecture.
If you continue, existing files will be overwritten! Proceed[y/N]? y
[sudo] password for axio:
Extracting SDK.........................done
Setting it up...done
SDK has been successfully set up and is ready to be used.
```

2.2.2 Setting Up the Cross-Development Environment

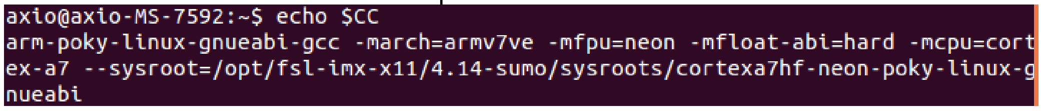
Before you can develop using the cross-toolchain, you need to set up the cross-development environment, and then you can find this script in the directory you chose for installation.

1. To set up cross-toolchain environment.



```
axio@axio-MS-7592:~$ source /opt/fsl-imx-x11/4.14-sumo/environment-setup-cortexa
7hf-neon-poky-linux-gnueabi
```

2. Check whether the Cross-Development Environment is successfully set up. You will find the information below if setup is succesful.

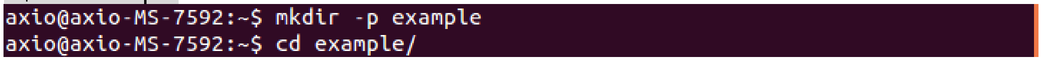


```
axio@axio-MS-7592:~$ echo $CC
arm-poky-linux-gnueabi-gcc -march=armv7ve -mfpu=neon -mfloat-abi=hard -mcpu=cort
ex-a7 --sysroot=/opt/fsl-imx-x11/4.14-sumo/sysroots/cortexa7hf-neon-poky-linux-g
nueabi
```

2.2.3 Write and Compile Sample Program

1. Create a directory on your host PC

```
~$ mkdir -p example
~$ cd example
```

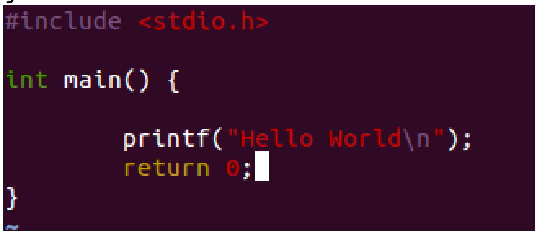


```
axio@axio-MS-7592:~$ mkdir -p example
axio@axio-MS-7592:~$ cd example/
```

2. Use vi to edit hello.c.

```
~$ vi hello.c
```

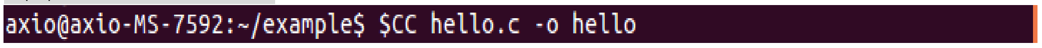
```
#include<stdio.h>
int main()
{
    printf("Hello World\n");
    return 0;
}
```



```
#include <stdio.h>

int main() {

        printf("Hello World\n");
        return 0;
}
~
```

3. To compile the program, enter the commands:

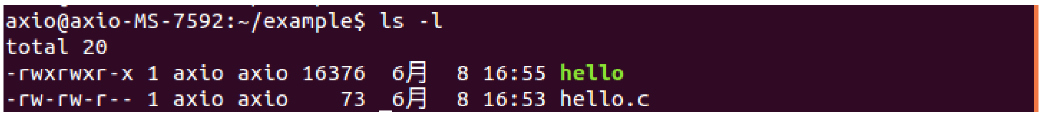
```
~$ $CC hello.c -o hello
```



```
axio@axio-MS-7592:~/example$ $CC hello.c -o hello
```

4. After compiling, enter the following command and you can see the 'hello' execution file.

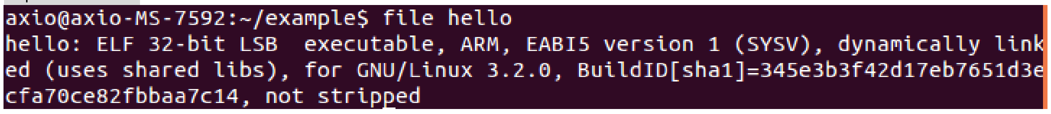
```
~$ ls -l
```



```
axio@axio-MS-7592:~/example$ ls -l
total 20
-rwxrwxr-x 1 axio axio 16376  6月  8 16:55 hello
-rw-rw-r-- 1 axio axio    73  6月  8 16:53 hello.c
```

5. Check whether the ARM executable format is created successfully. You will see the information below if the format is successfully created

```
~$ file hello
```



```
axio@axio-MS-7592:~/example$ file hello
hello: ELF 32-bit LSB  executable, ARM, EABI5 version 1 (SYSV), dynamically link
ed (uses shared libs), for GNU/Linux 3.2.0, BuildID[sha1]=345e3b3f42d17eb7651d3e
cfa70ce82fbbaa7c14, not stripped
```

2.3 How to Put and Run a Sample Program

This section shows how to put the 'hello' program into the AGENT200 and execute the program via FTP, a USB flash drive, and TFTP.

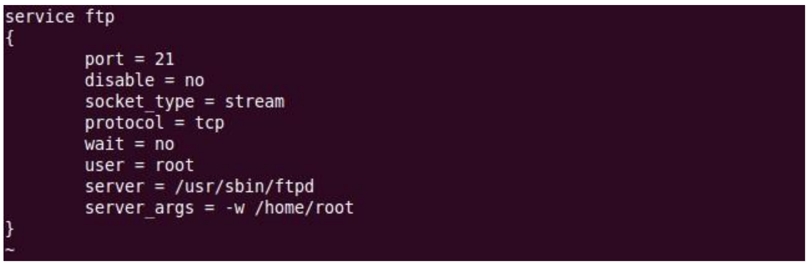
2.3.1 Via FTP

The AGENT200 has a built-in FTP server. Users can put 'hello' program to AGENT200 via FTP by following the steps below.

1. Enable FTPD daemon on AGENT200
   Use vi to open /etc/xinetd.d/ftpd file

```
~# vi /etc/xinetd.d/ftpd
```

```
service ftp
{
        port = 21
        disable = no
        socket_type = stream
        protocol = tcp
        wait = no
        user = root
        server = /usr/sbin/ftpd
        server_args = -w /home/root
}
```



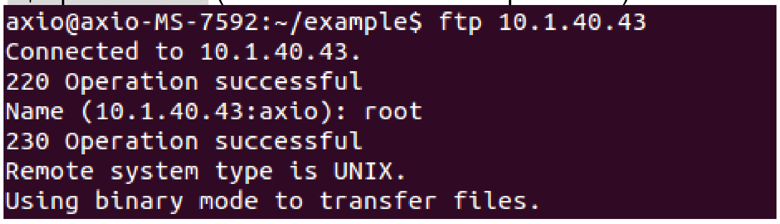
```
service ftp
{
        port = 21
        disable = no
        socket_type = stream
        protocol = tcp
        wait = no
        user = root
        server = /usr/sbin/ftpd
        server_args = -w /home/root
}
~
```

2. Restart FTP server on AGENT200

```
~# systemctl restart xinetd
```

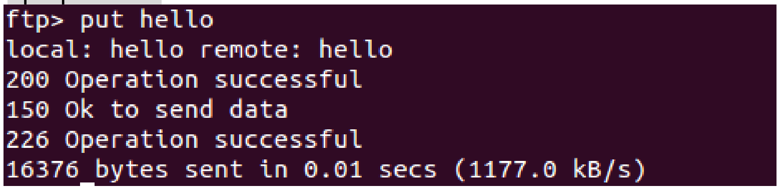
```
root@agent200:~# systemctl restart xinetd
```

3. To connect your host PC to AGENT200.
~$ ftp 10.1.40.43 (username 'root' without password)



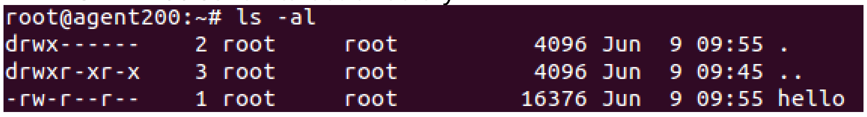
```
axio@axio-MS-7592:~/example$ ftp 10.1.40.43
Connected to 10.1.40.43.
220 Operation successful
Name (10.1.40.43:axio): root
230 Operation successful
Remote system type is UNIX.
Using binary mode to transfer files.
```

4. Upload "hello" program to AGENT200 from your host PC
ftp> put hello



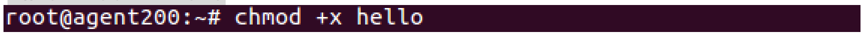
```
ftp> put hello
local: hello remote: hello
200 Operation successful
150 Ok to send data
226 Operation successful
16376 bytes sent in 0.01 secs (1177.0 kB/s)
```

5. If the operation is successful on the AGENT200, you can see 'hello' program on AGENT200's */home/root* directory.



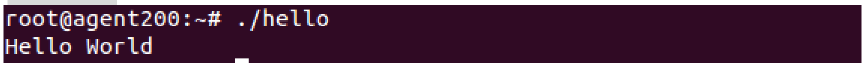
```
root@agent200:~# ls -al
drwx------    2 root     root          4096 Jun  9 09:55 .
drwxr-xr-x    3 root     root          4096 Jun  9 09:45 ..
-rw-r--r--    1 root     root         16376 Jun  9 09:55 hello
```

6. To change file permission for executable on AGENT200.
~# chmod +x hello



```
root@agent200:~# chmod +x hello
```

7. Run the 'hello' program on AGENT200.
~# ./hello



```
root@agent200:~# ./hello
Hello World
```

2.3.2 Via USB Flash Drive

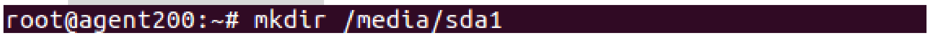
Another method of putting 'hello' program into AGENT200 is via USB flash drive. Please follow the instructions below.

AGENT200 supports storage format FAT32 /FAT/EXT2/EXT3/EXT4

1. From the host PC, copy 'hello' program to USB flash drive.

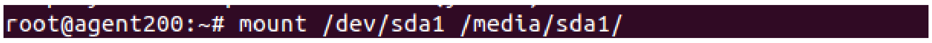
2. Attach USB flash drive to AGENT200.

3. ~# mkdir /media/sda1



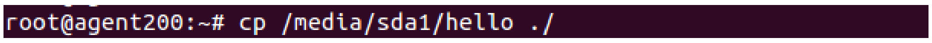
```
root@agent200:~# mkdir /media/sda1
```

4. ~# mount /dev/sda1 /media/sda1



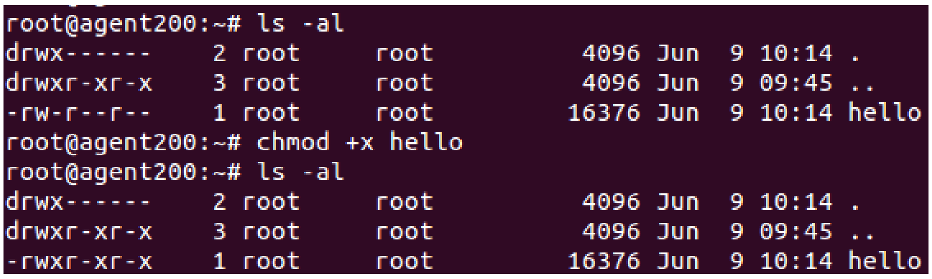
```
root@agent200:~# mount /dev/sda1 /media/sda1/
```

5. ~# cp /media/sda1/hello ./



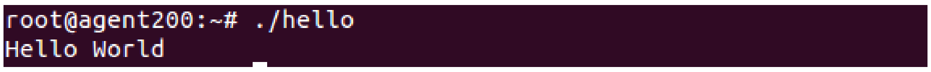
```
root@agent200:~# cp /media/sda1/hello ./
```

6. ~# chmod +x hello



```
root@agent200:~# ls -al
drwx------    2 root     root          4096 Jun  9 10:14 .
drwxr-xr-x    3 root     root          4096 Jun  9 09:45 ..
-rw-r--r--    1 root     root         16376 Jun  9 10:14 hello
root@agent200:~# chmod +x hello
root@agent200:~# ls -al
drwx------    2 root     root          4096 Jun  9 10:14 .
drwxr-xr-x    3 root     root          4096 Jun  9 09:45 ..
-rwxr-xr-x    1 root     root         16376 Jun  9 10:14 hello
```

7. ~# ./hello



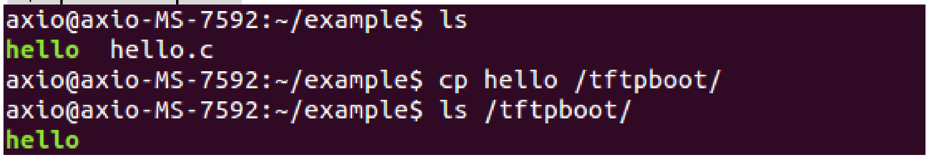
```
root@agent200:~# ./hello
Hello World
```

2.3.3 Via TFTP

Originally the Host Development System Installation already has TFTP server installed. You can put the 'hello' program into AGENT200 via TFTP. Please follow the instructions below.

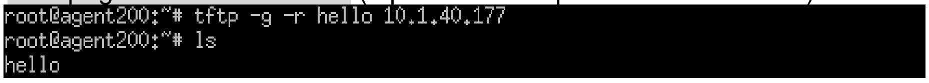
1. Refer to section 5.1.1 step 4. Install and configure TFTP server for install and setup your TFTP:

2. To copy "hello" program to "tftpboot" folder in host PC.
~$ cp hello /tftpboot



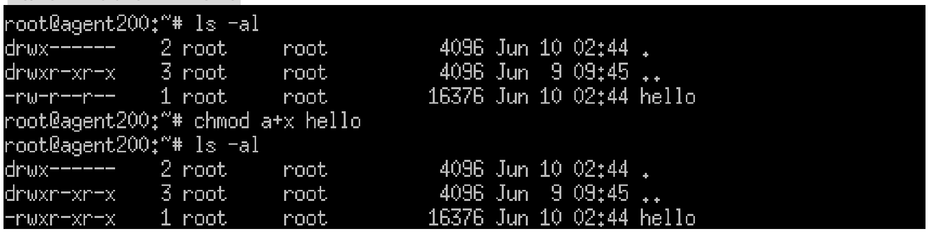
```
axio@axio-MS-7592:~/example$ ls
hello  hello.c
axio@axio-MS-7592:~/example$ cp hello /tftpboot/
axio@axio-MS-7592:~/example$ ls /tftpboot/
hello
```

3. To enter the following command on AGENT200.
~# tftp -g -r hello 10.1.40.177 (tftp server IP depend on host PC's IP)



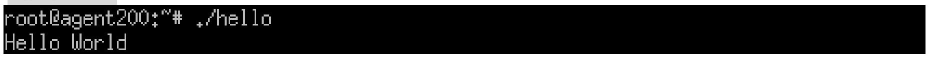
```
root@agent200:~# tftp -g -r hello 10.1.40.177
root@agent200:~# ls
hello
```

4. To enter the following command on AGENT200.
~# chmod a+x hello



```
root@agent200:~# ls -al
drwx------    2 root     root          4096 Jun 10 02:44 .
drwxr-xr-x    3 root     root          4096 Jun  9 09:45 ..
-rw-r--r--    1 root     root         16376 Jun 10 02:44 hello
root@agent200:~# chmod a+x hello
root@agent200:~# ls -al
drwx------    2 root     root          4096 Jun 10 02:44 .
drwxr-xr-x    3 root     root          4096 Jun  9 09:45 ..
-rwxr-xr-x    1 root     root         16376 Jun 10 02:44 hello
```

5. Run the 'hello' program on AGENT200.
~# ./hello



```
root@agent200:~# ./hello
Hello World
```

Section 3
The Embedded Linux

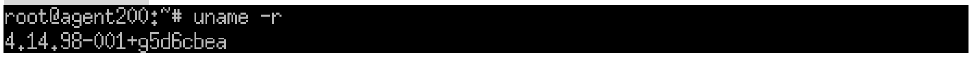
3.1 Embedded Linux Image Managing

3.1.1 System Version

This section describes how to determine system version information including kernel and root filesystem version on AGENT200.

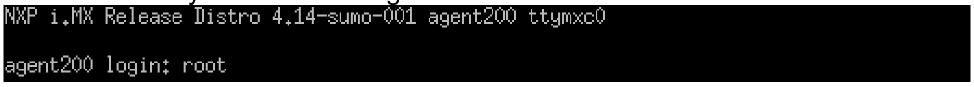
Check kernel version with the following command:

~# uname -r



```
root@agent200:~# uname -r
4.14.98-001+g5d6cbea
```

Check root filesystem with the login screen:

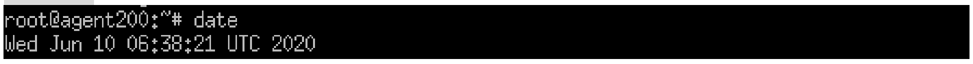


```
NXP i.MX Release Distro 4.14-sumo-001 agent200 ttymxc0

agent200 login: root
```

3.1.2 System Time

System time is the time value loaded from RTC each time the system boots up. Read system time with the following command on AGENT200:

~# date



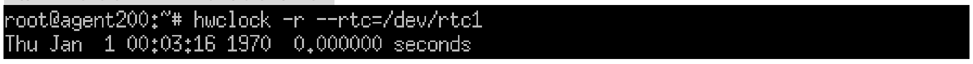
```
root@agent200:~# date
Wed Jun 10 06:38:21 UTC 2020
```

3.1.3 Internal RTC Time

The internal RTC time is read from i.MX processor internal RTC. Note that this time value is not saved, when system power is removed.

Read internal RTC time with the following command on AGENT200:

~# hwclock -r --rtc=/dev/rtc1



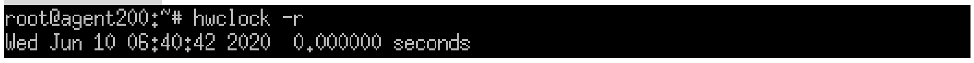
```
root@agent200:~# hwclock -r --rtc=/dev/rtc1
Thu Jan  1 00:03:16 1970  0.000000 seconds
```

3.1.4 External RTC Time

The external RTC time is read from RS5C372 external RTC. When system power is removed, this time value is kept as RS5C372 is powered by battery.

Read external RTC time with the following command:

~# hwclock -r

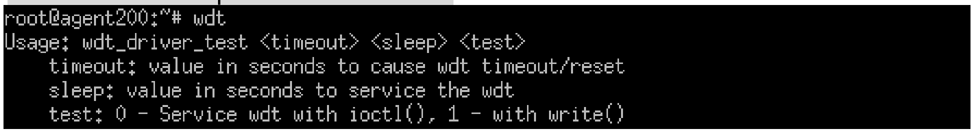


```
root@agent200:~# hwclock -r
Wed Jun 10 06:40:42 2020  0.000000 seconds
```

3.1.5 Watchdog timer

Function: wdt_driver_test.out
Description: When <sleep> parameters is more than <timeout> parameters, watchdog timer will be trigger

Note: The AGENT200 has been enabled for default settings, and the default parameters are ***10 5 0***
Commands example: ~# wdt 10 5 0 &

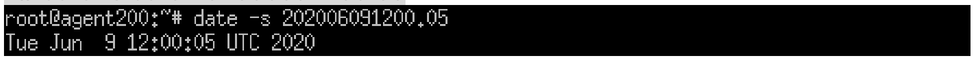


```
root@agent200:~# wdt
Usage: wdt_driver_test <timeout> <sleep> <test>
    timeout: value in seconds to cause wdt timeout/reset
    sleep: value in seconds to service the wdt
    test: 0 - Service wdt with ioctl(), 1 - with write()
```

3.1.6 Adjusting System Time

1. Manually set up the system time.

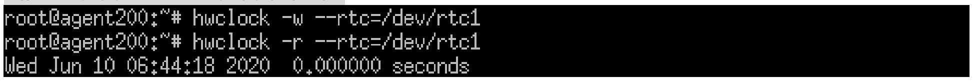
Format: YYYYMMDDHHmm.SS
~# date -s date 202006091200.05



```
root@agent200:~# date -s 202006091200.05
Tue Jun  9 12:00:05 UTC 2020
```

2. Write sync time to internal RTC

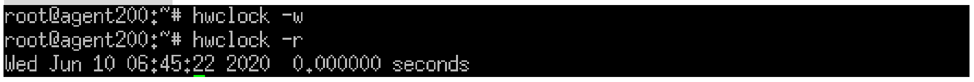
~# hwclock -w --rtc=/dev/rtc1



```
root@agent200:~# hwclock -w --rtc=/dev/rtc1
root@agent200:~# hwclock -r --rtc=/dev/rtc1
Wed Jun 10 06:44:18 2020  0.000000 seconds
```

3. Write sync time to external RTC

~# hwclock -w



```
root@agent200:~# hwclock -w
root@agent200:~# hwclock -r
Wed Jun 10 06:45:22 2020  0.000000 seconds
```

3.2 Networking

3.2.1 FTP – File Transfer Protocol

FTP is a standard network protocol used to transfer files from one host to another host over TCP-based network.

The AGENT200 comes with a built-in FTP server. Section 2.1 shows the steps to put 'hello' program in the AGENT200 via FTP.

3.2.2 TFTP – Trivial File Transfer Protocol

TFTP is a lightweight protocol of transfer files between a TFTP server and TFTP client over Ethernet. To support TFTP, this embedded Linux image has built-in TFTP client, so does its accompanying bootloader U-boot.

In Chapter 5, there are descriptions of TFTP server installation and kernel boot up process via TFTP. Section 2.3.3 shows you how to transfer file between server and client.

3.2.3 NFS – Network File System

NFS enables you to export a directory on an NFS server and mount that directory on remote client machine as if it were a local file system. Using NFS on target machine, we can have access to a huge number of files, libraries, and utilities during development and debugging, as well as booting up kernel.

This embedded Linux kernel is compiled with support for NFS, including server-side, client-side functionality and 'Root file system on NFS'.

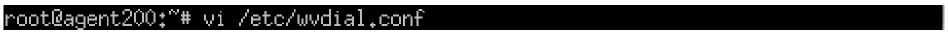
3.2.4 How to use 4G module

4G module connection to the Internet with wvdial Tool

If your 4G module is Quectel EC25, you can follow the instructions as bellows.

To create a wvdial config

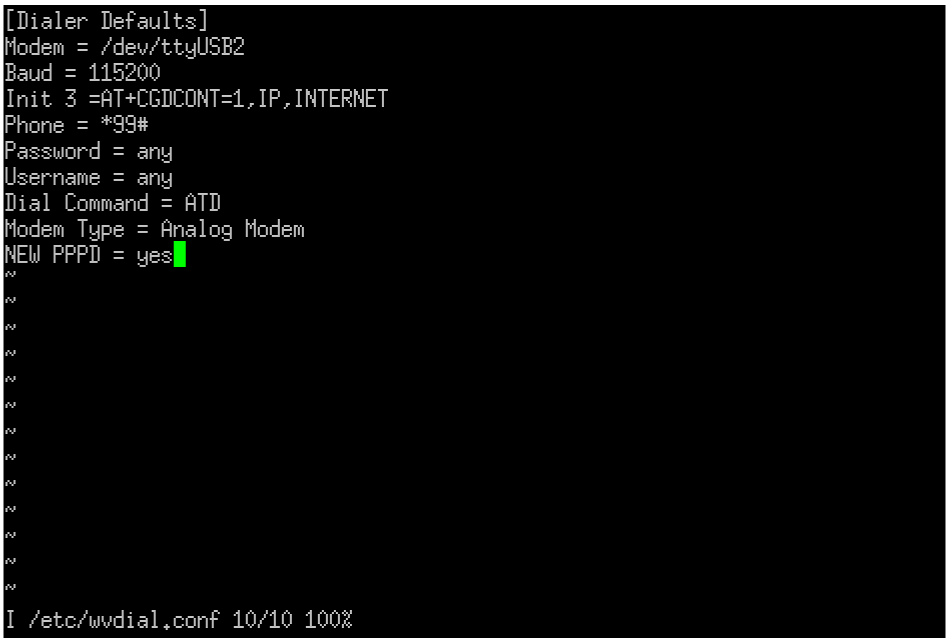
```
~# vi /etc/wvdial.conf
```



```
root@agent200:~# vi /etc/wvdial.conf
```

Please enter your information as below.

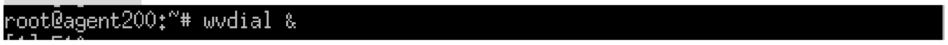
```
[Dialer Defaults]
Modem = /dev/ttyUSB2
Baud = 115200
Init 3 = AT+CGDCOUNT=1, “IP”,”INTERNET”
Phone = *99#
Password = any
Username = any
Dial Command = ATD
Modem Type = Analog Modem
NEW PPPD = yes
```



```
[Dialer Defaults]
Modem = /dev/ttyUSB2
Baud = 115200
Init 3 =AT+CGDCONT=1,IP,INTERNET
Phone = *99#
Password = any
Username = any
Dial Command = ATD
Modem Type = Analog Modem
NEW PPPD = yes
~
~
~
~
~
~
~
~
~
~
~
~
~
I /etc/wvdial.conf 10/10 100%
```

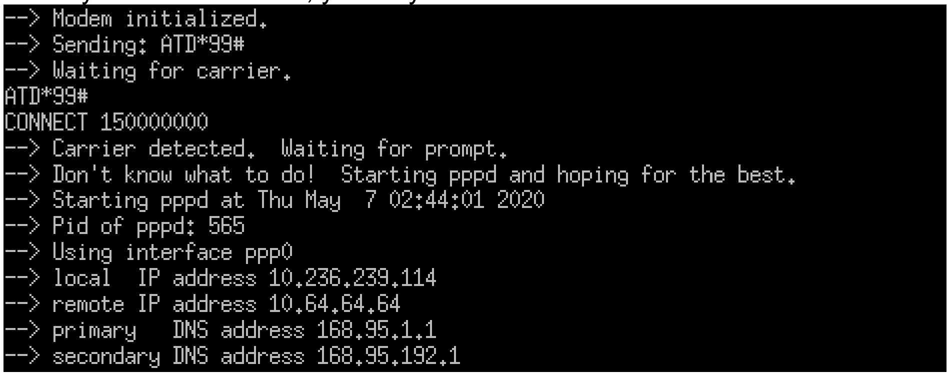
Please execute wvdial for internet connection

~# wvdial &



```
root@agent200:~# wvdial &
```

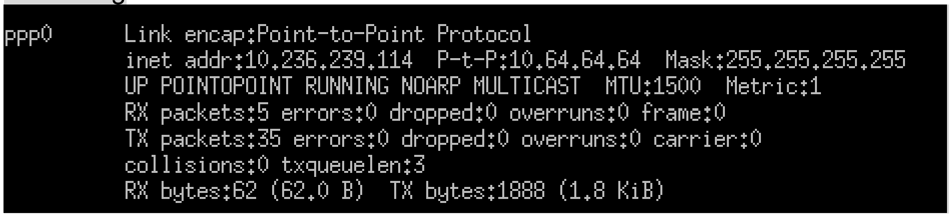
When you execute wvdial, you may find the information as below.



```
--> Modem initialized.
--> Sending: ATD*99#
--> Waiting for carrier.
ATD*99#
CONNECT 150000000
--> Carrier detected.  Waiting for prompt.
--> Don't know what to do!  Starting pppd and hoping for the best.
--> Starting pppd at Thu May  7 02:44:01 2020
--> Pid of pppd: 565
--> Using interface ppp0
--> local  IP address 10.236.239.114
--> remote IP address 10.64.64.64
--> primary   DNS address 168.95.1.1
--> secondary DNS address 168.95.192.1
```

You can execute command, **ifconfig** to examine PPP0 connection，ppp0 will be shown successful connection.

~# ifconfig



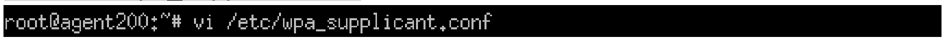
```
ppp0      Link encap:Point-to-Point Protocol
          inet addr:10.236.239.114  P-t-P:10.64.64.64  Mask:255.255.255.255
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric:1
          RX packets:5 errors:0 dropped:0 overruns:0 frame:0
          TX packets:35 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:3
          RX bytes:62 (62.0 B)  TX bytes:1888 (1.8 KiB)
```

3.2.5 How to use Wi-Fi module (Optional)

If your Wi-Fi module is WPEQ-160ACN, you can follow the instructions below.

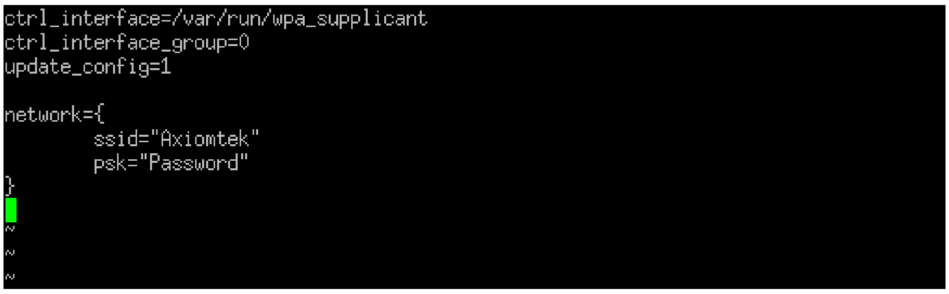
Editor /etc/wpa_supplicant.conf file

~# vi /etc/wpa_supplicant.conf



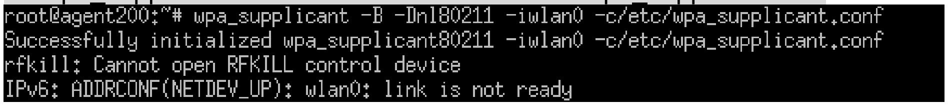
```
root@agent200:~# vi /etc/wpa_supplicant.conf
```

Enter your router's SSID and Password



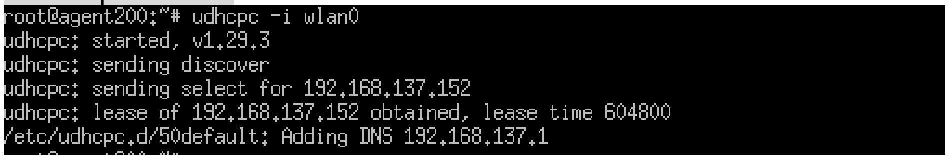
```
ctrl_interface=/var/run/wpa_supplicant
ctrl_interface_group=0
update_config=1

network={
        ssid="Axiomtek"
        psk="Password"
}

~
~
~
```

Please execute wpa_supplicant for internet connection
~# wpa_supplicant –B –Dnl80211 –iwlan0 –c/etc/wpa_supplicant.conf



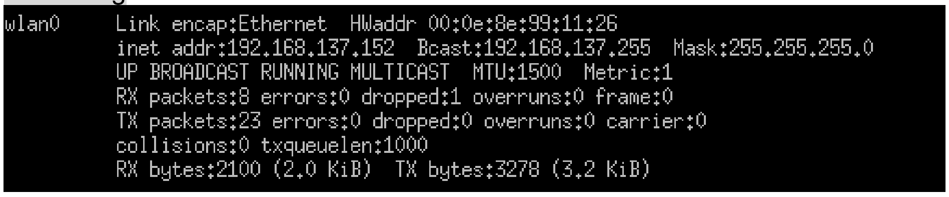
```
root@agent200:~# wpa_supplicant -B -Dnl80211 -iwlan0 -c/etc/wpa_supplicant.conf
Successfully initialized wpa_supplicant80211 -iwlan0 -c/etc/wpa_supplicant.conf
rfkill: Cannot open RFKILL control device
IPv6: ADDRCONF(NETDEV_UP): wlan0: link is not ready
```

Then, execute command "udhcpc" to get IP.
~# udhcpc –i wlan0



```
root@agent200:~# udhcpc -i wlan0
udhcpc: started, v1.29.3
udhcpc: sending discover
udhcpc: sending select for 192.168.137.152
udhcpc: lease of 192.168.137.152 obtained, lease time 604800
/etc/udhcpc.d/50default: Adding DNS 192.168.137.1
```

Final, you can execute command "ifconfig" to check connection.
~# ifconfig



```
wlan0     Link encap:Ethernet  HWaddr 00:0e:8e:99:11:26
          inet addr:192.168.137.152  Bcast:192.168.137.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:8 errors:0 dropped:1 overruns:0 frame:0
          TX packets:23 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:2100 (2.0 KiB)  TX bytes:3278 (3.2 KiB)
```

This page is intentionally left blank.

Section 4
System service

This chapter separates into three parts. First is introducing Agent200 service function. Second is sample code for C (programming language). Three is explaining how to operate services in Linux.

4.1 Agent200 system service

Please refer below introduction for Agent 200 services and it will teach how to operate hardware function which included erial、wdt、can bus、power usb、dio and led.

Summary table of available service

No.	Service	Description
1	ax_commode	Initialize COM port communication type.
2	ax_wdt	Initialize WDT state.
3	ax_can	Initialize CAN bus bitrate.
4	ax_power_use	Startup USB power
5	ax_dio_init	Initialize DIO status
6	ax_led_init	Initialize LED status.

Service: ax_commode

Service	ax_commode
Description	Initialize COM port communication type.
Config file	/etc/init.d/ax_commode.config
Arguments	com1_mode : setting com1 mode type mode type： 1：RS232 2：RS485 3：RS422 Example: com1_mode=1 (com1 as rs232 mode)
Usage	/etc/init.d/ax_commode start /etc/init.d/ax_commode.config
Other	None

Service: ax_wdt

Service	ax_wdt
Description	Initialize WDT state.
Config file	/etc/init.d/ax_wdt.config
Arguments	wdt_timeout: value in seconds to cause wdt timeout/reset. wdt_sleep: value in seconds to service the wdt wdt_test: 0—service wdt with ioctl(), 1—with write() Example: wdt_timeout=10 wdt_sleep=5 wdt_test=1
Usage	/etc/init.d/ax_wdt start /etc/init.d/ax_wdt.config
Other	None

Service: ax_can

Service	ax_can
Description	Initialize CAN bus bitrate.
Config file	/etc/init.d/ax_can.config
Arguments	bitrate：Setting value correspond bitrate of can bus. 1：10k 2：20k 3：50k 4：100k 5：125k 6：250k 7：500k 8：800k 9：1000k Example: bitrate=3 (bitrate as 50k)
Usage	/etc/init.d/ax_can start /etc/init.d/ax_can.config
Other	None

Service: ax_power_usb

Service	ax_power_usb {start}
Description	Startup USB power
Usage	/etc/init.d/ax_power_usb start
Other	None

Service: ax_dio

Service	ax_dio
Description	Initialize DIO status
Config file	/etc/init.d/ax_dio.config
Arguments	do0_status: setting DO0 value do1_status: setting DO1 value do2_status: setting DO2 value do3_status: setting DO3 value Digital output value： 0：LOW 1：HIGH Example: do0_status=0 (digital output as low) do1_status=0 do2_status=0 do3_status=0
Usage	/etc/init.d/ax_dio start /etc/init.d/ax_dio.config
Other	None

Service: ax_led

Service	ax_led
Description	Initialize LED status
Arguments	led1_status: setting LED1 value led2_status: setting LED2 value led3_status: setting LED3 value LED value： 0：LOW 1：HIGH Example: led1_status=0 (LED output as low) led2_status=0 led3_status=0
Usage	/etc/init.d/ax_led start /etc/init.d/ax_led.config
Other	None

4.2 Sample code for C

Please refer sample code for C and understand how to operate hardware function.

4.2.1 Com Port SAMPLE CODE

COM receive

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <errno.h>
#include <termios.h>
#include <fcntl.h>
#include <termios.h>
#include <pthread.h>
#include "serial.h"
#include <asm-generic/ioctls.h>
#define SET_COM_TYPE 0x542A
```

```
int main(int argc, char *argv[])
{
        int ReadRet,fd,RX_len = 0,OutCount = 0;
        struct termios orig_options,options;
        struct serial_rs485 conf;
        char RecvBuf[128];
        int type = atoi(argv[1]);
        printf("Test for com2 Read(232/422/485) \n");
        printf("example : ./comRead 1 (1=232, 2=485, 3=422)\n");
        fd = open("/dev/ttymxc1", O_RDWR | O_NOCTTY);
        if(fd < 0) {
                printf("open error /dev/ttymxc1 error\n");
        }
        //setting com1 as rs485
                switch(type) {
                case 1:
                        printf("Set as RS232\n");
                        break;
                case 2:
                        printf("Set as RS485\n");
                        break;
                case 3:
                        printf("Set as RS422\n");
                        break;
        }
        //init setting
        fcntl(fd, F_SETFL, 0);
        tcgetattr(fd, &orig_options);
       memset(&options, 0, sizeof(options));
        options.c_cflag &= ~CSTOPB;
        options.c_cflag &= ~CSIZE;
        options.c_cflag |= PARENB;
        options.c_cflag &= ~PARODD;
        options.c_cflag |= CS8;
        options.c_cflag &= ~CRTSCTS;
        options.c_iflag &= ~(IXON | IXOFF | IXANY);
        options.c_lflag &= ~(ICANON | IEXTEN | ISIG | ECHO);
        options.c_oflag &= ~OPOST;
        options.c_iflag &= ~(ICRNL | INPCK | ISTRIP | IXON | BRKINT );
        options.c_cflag |= (CLOCAL | CREAD);
        options.c_cc[VMIN] = 1;
        options.c_cc[VTIME] = 0;

        usleep(100);
        ioctl(fd, SET_COM_TYPE, &type);
        cfsetispeed(&options, B115200);
        cfsetospeed(&options, B115200);
        tcsetattr(fd, TCSANOW, &options);



        while(1)
        {
                //Test Read
                memset(RecvBuf,0x00,sizeof(RecvBuf));
                ReadRet = read(fd, RecvBuf, sizeof(RecvBuf));
                if (ReadRet > 0)
                {
                        printf("Test Read : Len [%d] / Read [%s]\n",ReadRet,RecvBuf);
                }
                usleep(100000);

        }
        tcsetattr(fd, TCSANOW, &orig_options);
        close(fd); //Close the serial port
        printf("Serial port closed.\n");

        return 0;
}
```

COM send:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <errno.h>
#include <termios.h>
#include <fcntl.h>
#include <termios.h>
#include <pthread.h>
#include "serial.h"
#include <asm-generic/ioctls.h>
#define SET_COM_TYPE 0x542A


int main(int argc, char *argv[])
{
        int i,WriteRet,fd,TX_len = 0;
        struct termios orig_options,options;
        struct serial_rs485 conf;
        char SendBuf[16];
        int type = atoi(argv[1]);
        printf("Test for com1 Write(232/422/485) \n");
        printf("example : ./comWrite 1 (1=232, 2=485, 3=422)\n");
        fd = open("/dev/ttymxc1", O_RDWR | O_NOCTTY);
        if(fd < 0) {
                printf("open error /dev/ttymxc1 error\n");
        }
        //setting com1 as rs485
                switch(type) {
                case 1:
                        printf("Set as RS232\n");
                        break;
                case 2:
                        printf("Set as RS485\n");
                        break;
```

```
          case 3:
                    printf("Set as RS422\n");
                    break;
     }
     //init setting
     fcntl(fd, F_SETFL, 0);
     tcgetattr(fd, &orig_options);
    memset(&options, 0, sizeof(options));
     options.c_cflag &= ~CSTOPB;
     options.c_cflag &= ~CSIZE;
     options.c_cflag |= PARENB;
     options.c_cflag &= ~PARODD;
     options.c_cflag |= CS8;
     options.c_cflag &= ~CRTSCTS;
     options.c_iflag &= ~(IXON | IXOFF | IXANY);
     options.c_lflag &= ~(ICANON | IEXTEN | ISIG | ECHO);
     options.c_oflag &= ~OPOST;
     options.c_iflag &= ~(ICRNL | INPCK | ISTRIP | IXON | BRKINT );
     options.c_cflag |= (CLOCAL | CREAD);
     options.c_cc[VMIN] = 1;
     options.c_cc[VTIME] = 0;

     usleep(100);
     ioctl(fd, SET_COM_TYPE, &type);
     cfsetispeed(&options, B115200);
     cfsetospeed(&options, B115200);
     tcsetattr(fd, TCSANOW, &options);



     printf("start write\n");
     memset(SendBuf,0x00,16);
     sprintf(SendBuf,"hello word");

     for(i=0;i<10;i++)
     {
          //Test Write
          WriteRet = write(fd,SendBuf,strlen(SendBuf));
          if(WriteRet > 0)
```

```
            {
                    TX_len = strlen(SendBuf);
                    printf("Test Write :Len [%d] / Send [%s] \n",TX_len,SendBuf);
            }
            else
            {
                    printf("Test Write Fail \n");
            }
            usleep(500000);

        }
        tcsetattr(fd, TCSANOW, &orig_options);
        close(fd); //Close the serial port
        printf("Serial port closed.\n");

        return 0;
}
```

4.2.2 USB Power sample code

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <fcntl.h>

#define USB_PIN 111

int main(int argc, char* argv[])
{
        FILE *pUSB;
        char file_name[80];

        sprintf(file_name, "/sys/class/gpio/gpio%d/value", USB_PIN);
        printf("%s\n", file_name);
        pUSB = fopen(file_name, "w" );
        fprintf(pUSB, "%d", 0); //usb power enable value is 0

        fclose(pUSB);

        return 0;
}
```

4.2.3 LED sample code

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <fcntl.h>

#define LED1_PIN 9
#define LED2_PIN 128
#define LED3_PIN 134


int main(int argc, char* argv[])
{
        FILE *pLED;
        char file_name[80];

        sprintf(file_name, "/sys/class/gpio/gpio%d/value", LED1_PIN);
         printf("%s\n", file_name);
        pLED = fopen(file_name, "w" );
        fprintf(pLED, "%d", 1);    //LED write 1 ,output is high
        fclose(pLED);



        return 0;
}
```

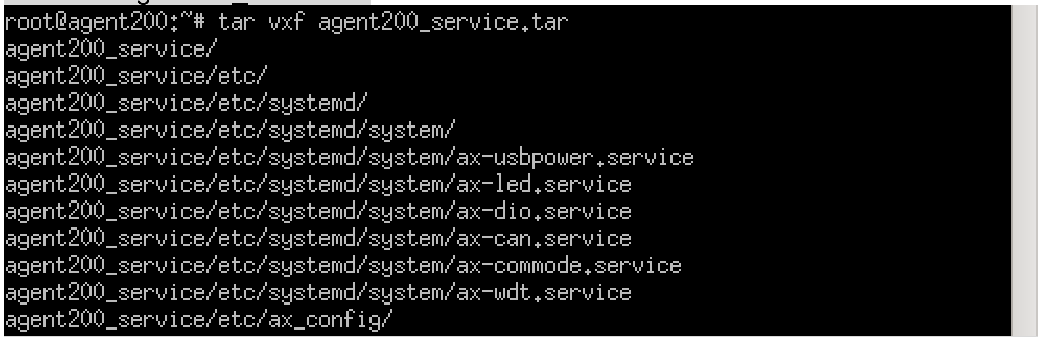
4.3 How to operate services in Linux

Please refer below to understand how to operate services and I/O.

4.3.1 Installation and activation Service

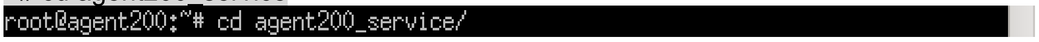
Open the file agent200_servic in "agent200_service.tar", execute "install_service.sh" and reboot system to make sure service is workable.

~# tar xvf agent200_service.tar



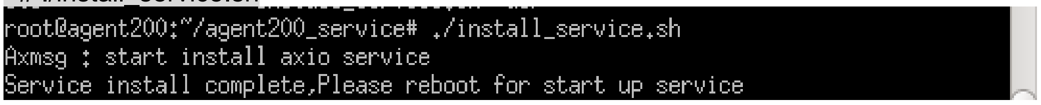
```
root@agent200:~# tar vxf agent200_service.tar
agent200_service/
agent200_service/etc/
agent200_service/etc/systemd/
agent200_service/etc/systemd/system/
agent200_service/etc/systemd/system/ax-usbpower.service
agent200_service/etc/systemd/system/ax-led.service
agent200_service/etc/systemd/system/ax-dio.service
agent200_service/etc/systemd/system/ax-can.service
agent200_service/etc/systemd/system/ax-commode.service
agent200_service/etc/systemd/system/ax-wdt.service
agent200_service/etc/ax_config/
```

~# cd agent200_service



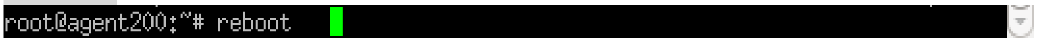
```
root@agent200:~# cd agent200_service/
```

~# ./install_service.sh

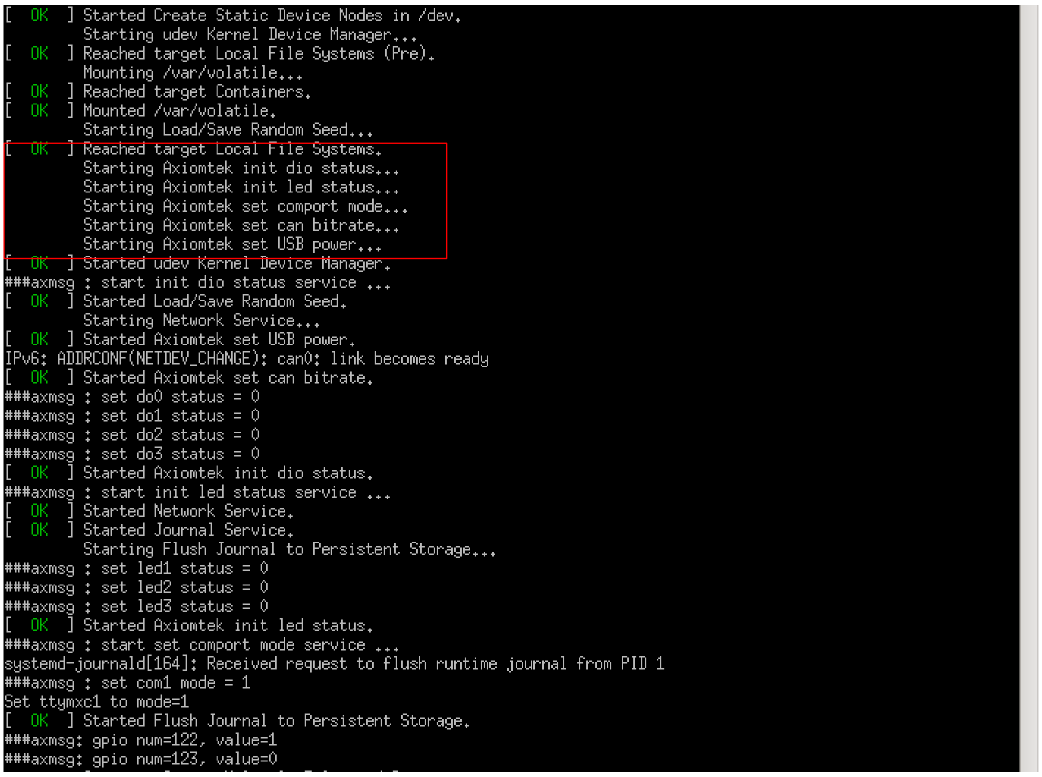


```
root@agent200:~/agent200_service# ./install_service.sh
Axmsg : start install axio service
Service install complete,Please reboot for start up service
```

~# reboot



```
root@agent200:~# reboot
```

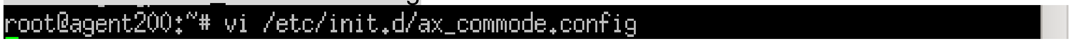


```
[  OK  ] Started Create Static Device Nodes in /dev.
         Starting udev Kernel Device Manager...
[  OK  ] Reached target Local File Systems (Pre).
         Mounting /var/volatile...
[  OK  ] Reached target Containers.
[  OK  ] Mounted /var/volatile.
         Starting Load/Save Random Seed...
[  OK  ] Reached target Local File Systems.
         Starting Axiomtek init dio status...
         Starting Axiomtek init led status...
         Starting Axiomtek set comport mode...
         Starting Axiomtek set can bitrate...
         Starting Axiomtek set USB power...
[  OK  ] Started udev Kernel Device Manager.
###axmsg : start init dio status service ...
[  OK  ] Started Load/Save Random Seed.
         Starting Network Service...
[  OK  ] Started Axiomtek set USB power.
IPv6: ADDRCONF(NETDEV_CHANGE): can0: link becomes ready
[  OK  ] Started Axiomtek set can bitrate.
###axmsg : set do0 status = 0
###axmsg : set do1 status = 0
###axmsg : set do2 status = 0
###axmsg : set do3 status = 0
[  OK  ] Started Axiomtek init dio status.
###axmsg : start init led status service ...
[  OK  ] Started Network Service.
[  OK  ] Started Journal Service.
         Starting Flush Journal to Persistent Storage...
###axmsg : set led1 status = 0
###axmsg : set led2 status = 0
###axmsg : set led3 status = 0
[  OK  ] Started Axiomtek init led status.
###axmsg : start set comport mode service ...
systemd-journald[164]: Received request to flush runtime journal from PID 1
###axmsg : set com1 mode = 1
Set ttymxc1 to mode=1
[  OK  ] Started Flush Journal to Persistent Storage.
###axmsg: gpio num=122, value=1
###axmsg: gpio num=123, value=0
```

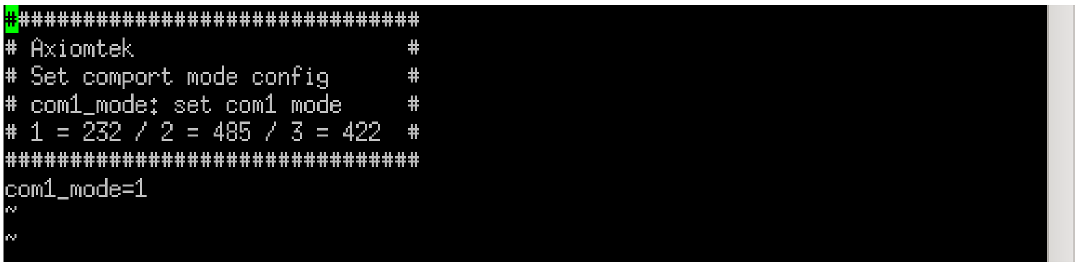
4.3.2 Change services setting

Please open the file "ax_commode.config" to revise service setting.

~# vi /etc/init.d/ax_commode.config



```
root@agent200:~# vi /etc/init.d/ax_commode.config
```

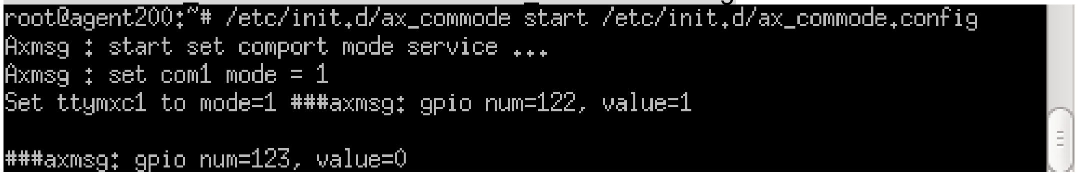


```
#################################
# Axiomtek                      #
# Set comport mode config       #
# com1_mode: set com1 mode      #
# 1 = 232 / 2 = 485 / 3 = 422   #
#################################
com1_mode=1
~
~
```

Activation Service

After finish revision, users can activate service without rebooting system.

~# /etc/init.d/ax_commode start /etc/init.d/ax_commode.config



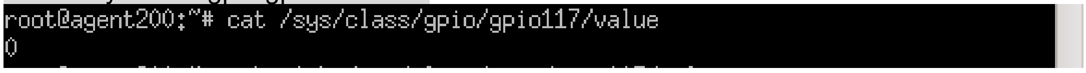
```
root@agent200:~# /etc/init.d/ax_commode start /etc/init.d/ax_commode.config
Axmsg : start set comport mode service ...
Axmsg : set com1 mode = 1
Set ttymxc1 to mode=1 ###axmsg: gpio num=122, value=1

###axmsg: gpio num=123, value=0
```

4.3.3 How to operate I/O in Linux

Please refer below operation process for DIO and LED setting in Linux.

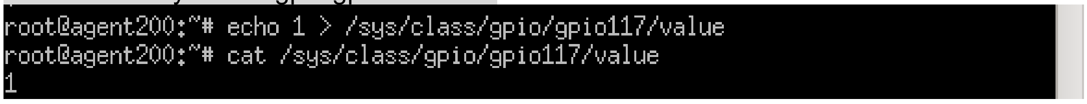
(1) Execute below command to operate DIO and LED function.

~# cat /sys/class/gpio/gpio117/value



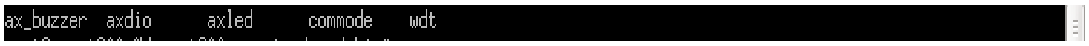
```
root@agent200:~# cat /sys/class/gpio/gpio117/value
0
```

~# echo 1 > /sys/class/gpio/gpio117/value



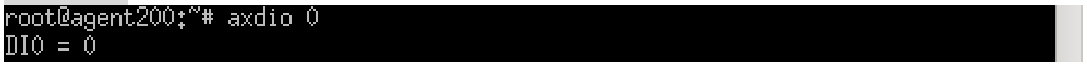
```
root@agent200:~# echo 1 > /sys/class/gpio/gpio117/value
root@agent200:~# cat /sys/class/gpio/gpio117/value
1
```

(2) Use ax tool to operate DIO and LED function.
Ax tool was installed in Linux when "install_service.sh" executing, user use directly ax tool in Linux.



```
ax_buzzer  axdio      axled      commode    wdt
```

~# axdio 0



```
root@agent200:~# axdio 0
DIO = 0
```

Section 5
Board Support Package (BSP)

5.1 Host Development System Installation

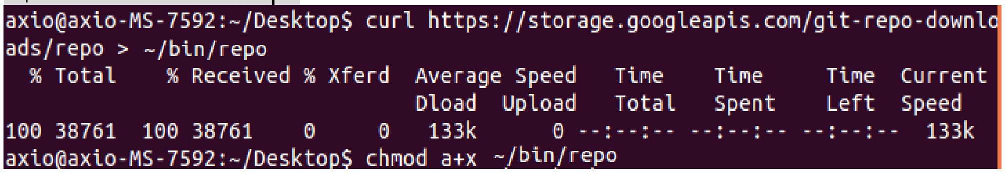
5.1.1 Install Host System

1. Download Ubuntu 14.04 LTS iso image.

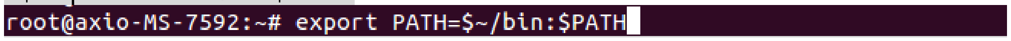
2. Install Ubuntu 14.04 LTS.

3. Install host packages needed by Yocto development as follows:
```
~$ sudo apt-get install wget git-core unzip texinfo libsdl1.2-dev gawk diffstat \
               wget git-core unzip texinfo libsdl1.2-dev gawk diffstat texi2html \
               docbook-utils python-pysqlite2 help2man make gcc g++ \
               desktop-file-utils libgl1-mesa-dev libglu1-mesa-dev mercurial \
               autoconf automake groff curl lzop asciidoc xterm chrpath
```
   i.MX layer host packages for a Ubuntu 14.04 host only are:
```
~$ sudo apt-get install u-boot-tools
```

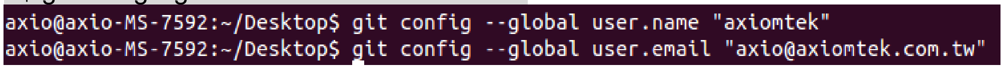
5.1.2 Install Yocto Development

1. Setting up the repo utility Create a bin folder in the home directory.
   `~$ mkdir ~/bin` (this step may not be needed if the bin folder already exists)
```
~$ curl https://storage.googleapis.com/git-repo-downloads/repo > ~/bin/repo
```
```
~$ chmod a+x ~/bin/repo
```


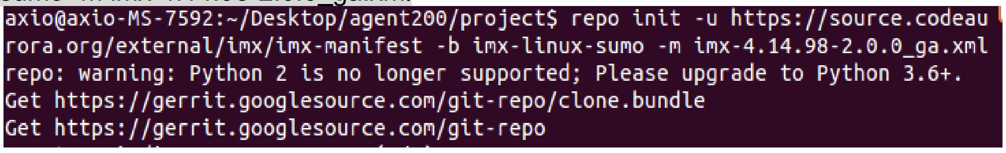
```
axio@axio-MS-7592:~/Desktop$ curl https://storage.googleapis.com/git-repo-downlo
ads/repo > ~/bin/repo
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 38761  100 38761    0     0   133k      0 --:--:-- --:--:-- --:--:--  133k
axio@axio-MS-7592:~/Desktop$ chmod a+x ~/bin/repo
```

2. Add the following line to the.bashrc file to ensure that the ~/binfolder is in your PATH variable.
```
~$ export PATH=~/bin:$PATH
```


```
root@axio-MS-7592:~# export PATH=$~/bin:$PATH
```

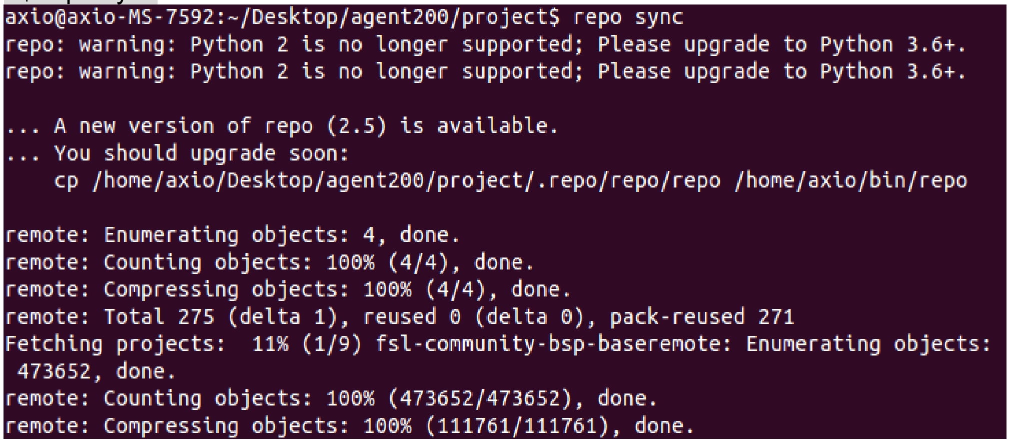
3. Setting up the Git environment
```
~$ git config --global user.name “Your Name”
~$ git config --global user.email “Your Email”
```


```
axio@axio-MS-7592:~/Desktop$ git config --global user.name "axiomtek"
axio@axio-MS-7592:~/Desktop$ git config --global user.email "axio@axiomtek.com.tw"
```

4. Download the Freescale's Yocto BSP source
```
~$ mkdir project
~$ cd project
~$ repo init -u https://source.codeaurora.org/external/imx/imx-manifest -b imx-linux-sumo -m imx-4.14.98-2.0.0_ga.xml
```


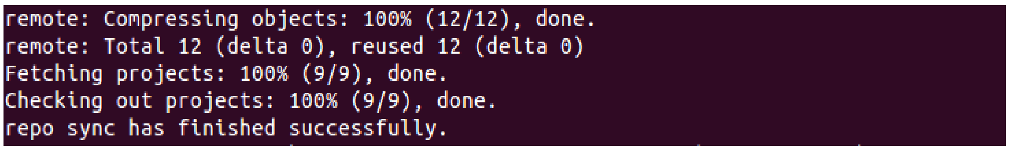
```
axio@axio-MS-7592:~/Desktop/agent200/project$ repo init -u https://source.codeau
rora.org/external/imx/imx-manifest -b imx-linux-sumo -m imx-4.14.98-2.0.0_ga.xml
repo: warning: Python 2 is no longer supported; Please upgrade to Python 3.6+.
Get https://gerrit.googlesource.com/git-repo/clone.bundle
Get https://gerrit.googlesource.com/git-repo
```

~$ repo sync



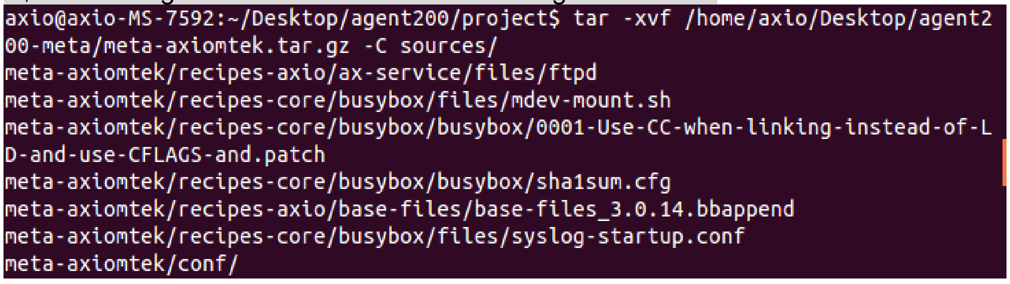
```
axio@axio-MS-7592:~/Desktop/agent200/project$ repo sync
repo: warning: Python 2 is no longer supported; Please upgrade to Python 3.6+.
repo: warning: Python 2 is no longer supported; Please upgrade to Python 3.6+.

... A new version of repo (2.5) is available.
... You should upgrade soon:
    cp /home/axio/Desktop/agent200/project/.repo/repo/repo /home/axio/bin/repo

remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 275 (delta 1), reused 0 (delta 0), pack-reused 271
Fetching projects:  11% (1/9) fsl-community-bsp-baseremote: Enumerating objects:
 473652, done.
remote: Counting objects: 100% (473652/473652), done.
remote: Compressing objects: 100% (111761/111761), done.
```

Clone Finish



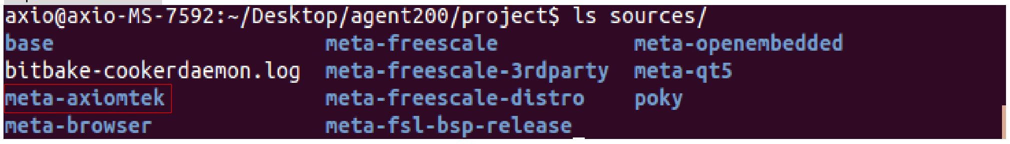
```
remote: Compressing objects: 100% (12/12), done.
remote: Total 12 (delta 0), reused 12 (delta 0)
Fetching projects: 100% (9/9), done.
Checking out projects: 100% (9/9), done.
repo sync has finished successfully.
```

5. Extract Axiomtek's Yocto BSP source (file is named **meta-axiomtek**)
   ~$ tar -xvf ./agent200-meta/meta-axiometk.tar.gz -C sources



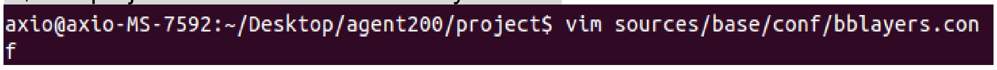
```
axio@axio-MS-7592:~/Desktop/agent200/project$ tar -xvf /home/axio/Desktop/agent2
00-meta/meta-axiomtek.tar.gz -C sources/
meta-axiomtek/recipes-axio/ax-service/files/ftpd
meta-axiomtek/recipes-core/busybox/files/mdev-mount.sh
meta-axiomtek/recipes-core/busybox/busybox/0001-Use-CC-when-linking-instead-of-L
D-and-use-CFLAGS-and.patch
meta-axiomtek/recipes-core/busybox/busybox/sha1sum.cfg
meta-axiomtek/recipes-axio/base-files/base-files_3.0.14.bbappend
meta-axiomtek/recipes-core/busybox/files/syslog-startup.conf
meta-axiomtek/conf/
```

Check **meta-axiomtek** in **sources** folder
~$ls sources/



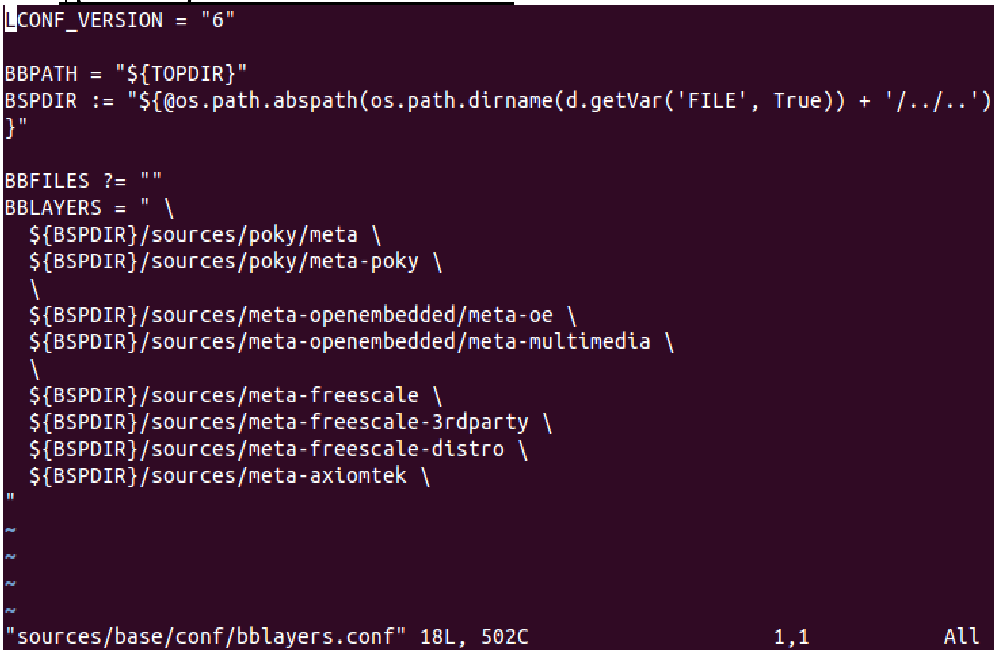
```
axio@axio-MS-7592:~/Desktop/agent200/project$ ls sources/
base                       meta-freescale           meta-openembedded
bitbake-cookerdaemon.log   meta-freescale-3rdparty  meta-qt5
meta-axiomtek              meta-freescale-distro    poky
meta-browser               meta-fsl-bsp-release
```

6. Update bblayers.conf
   ~$ vim project/sources/base/conf/bblayers.conf



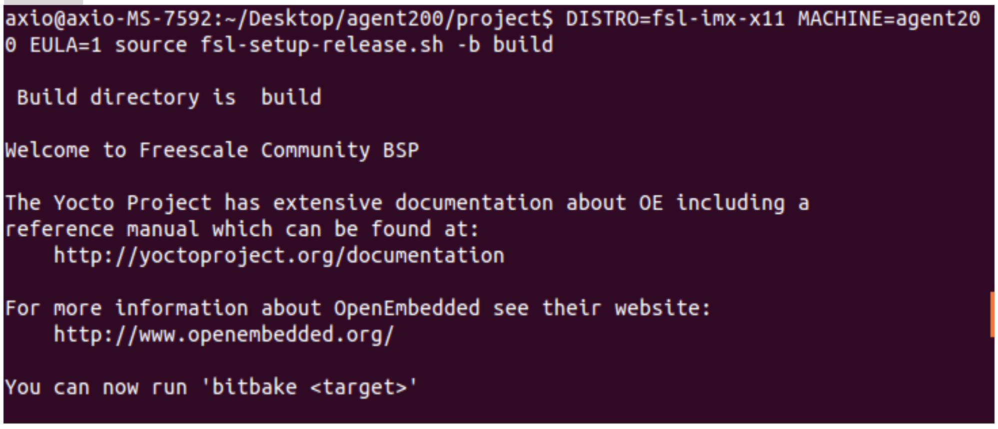
```
axio@axio-MS-7592:~/Desktop/agent200/project$ vim sources/base/conf/bblayers.con
f
```

   Add **${BSPDIR}/sources/meta-axiomtek ** in the .conf file



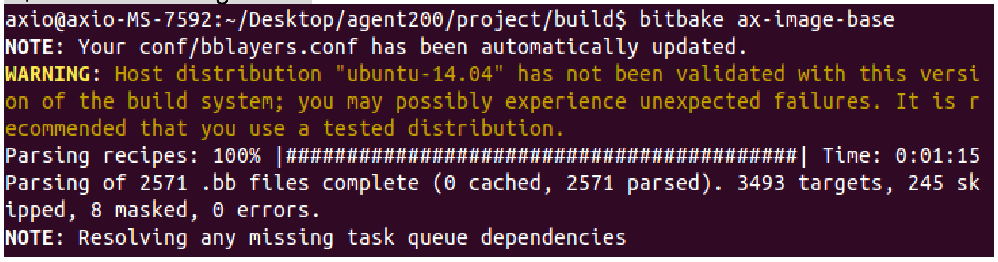
```
LCONF_VERSION = "6"

BBPATH = "${TOPDIR}"
BSPDIR := "${@os.path.abspath(os.path.dirname(d.getVar('FILE', True)) + '/../..')
}"

BBFILES ?= ""
BBLAYERS = " \
  ${BSPDIR}/sources/poky/meta \
  ${BSPDIR}/sources/poky/meta-poky \
  \
  ${BSPDIR}/sources/meta-openembedded/meta-oe \
  ${BSPDIR}/sources/meta-openembedded/meta-multimedia \
  \
  ${BSPDIR}/sources/meta-freescale \
  ${BSPDIR}/sources/meta-freescale-3rdparty \
  ${BSPDIR}/sources/meta-freescale-distro \
  ${BSPDIR}/sources/meta-axiomtek \
"
~
~
~
~
"sources/base/conf/bblayers.conf" 18L, 502C                    1,1           All
```

7. First build
   Choose your board
   ~$ DISTRO=fsl-imx-x11 MACHINE=agent200 EULA=1 source fsl-setup-release.sh -b build



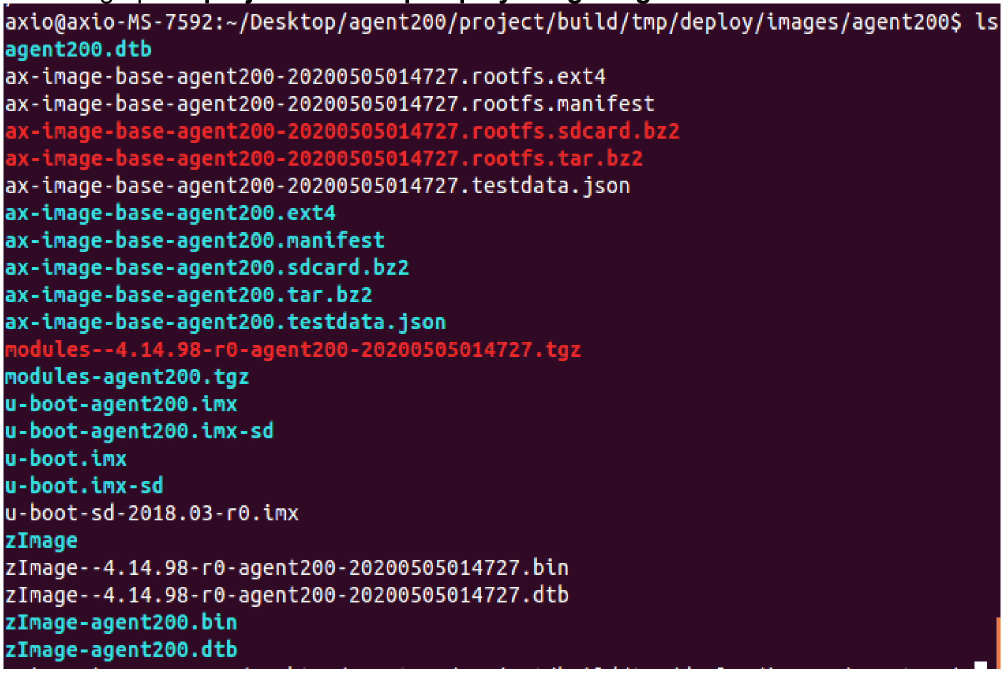
```
axio@axio-MS-7592:~/Desktop/agent200/project$ DISTRO=fsl-imx-x11 MACHINE=agent20
0 EULA=1 source fsl-setup-release.sh -b build

 Build directory is  build

Welcome to Freescale Community BSP

The Yocto Project has extensive documentation about OE including a
reference manual which can be found at:
    http://yoctoproject.org/documentation

For more information about OpenEmbedded see their website:
    http://www.openembedded.org/

You can now run 'bitbake <target>'
```

   Start to build image
   ~$ bitbake ax-image-base



```
axio@axio-MS-7592:~/Desktop/agent200/project/build$ bitbake ax-image-base
NOTE: Your conf/bblayers.conf has been automatically updated.
WARNING: Host distribution "ubuntu-14.04" has not been validated with this versi
on of the build system; you may possibly experience unexpected failures. It is r
ecommended that you use a tested distribution.
Parsing recipes: 100% |###########################################| Time: 0:01:15
Parsing of 2571 .bb files complete (0 cached, 2571 parsed). 3493 targets, 245 sk
ipped, 8 masked, 0 errors.
NOTE: Resolving any missing task queue dependencies
```

8. After build image finish.
   The image path: **project/build/tmp/deploy/images/agent200**

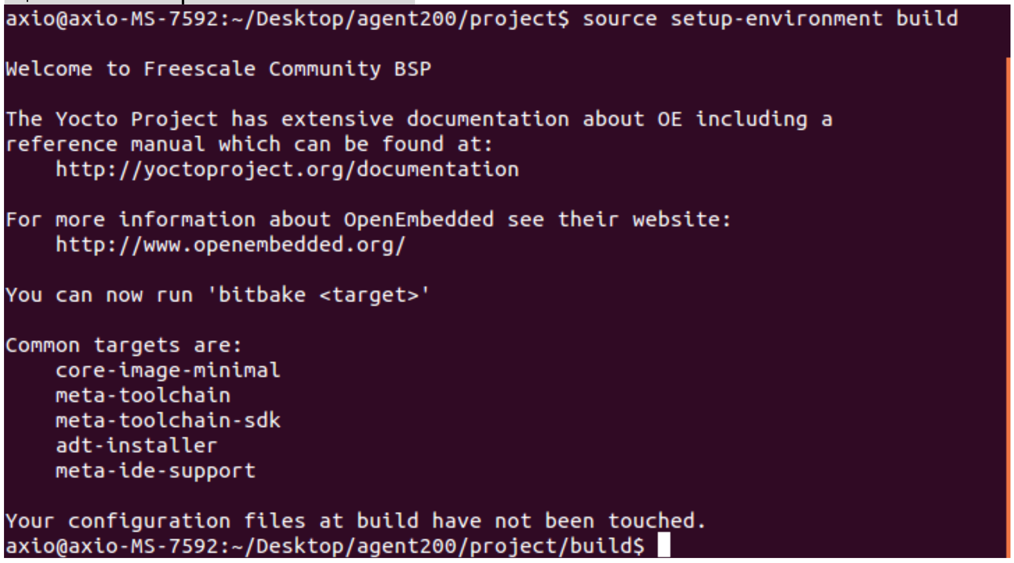


```
axio@axio-MS-7592:~/Desktop/agent200/project/build/tmp/deploy/images/agent200$ ls
agent200.dtb
ax-image-base-agent200-20200505014727.rootfs.ext4
ax-image-base-agent200-20200505014727.rootfs.manifest
ax-image-base-agent200-20200505014727.rootfs.sdcard.bz2
ax-image-base-agent200-20200505014727.rootfs.tar.bz2
ax-image-base-agent200-20200505014727.testdata.json
ax-image-base-agent200.ext4
ax-image-base-agent200.manifest
ax-image-base-agent200.sdcard.bz2
ax-image-base-agent200.tar.bz2
ax-image-base-agent200.testdata.json
modules--4.14.98-r0-agent200-20200505014727.tgz
modules-agent200.tgz
u-boot-agent200.imx
u-boot-agent200.imx-sd
u-boot.imx
u-boot.imx-sd
u-boot-sd-2018.03-r0.imx
zImage
zImage--4.14.98-r0-agent200-20200505014727.bin
zImage--4.14.98-r0-agent200-20200505014727.dtb
zImage-agent200.bin
zImage-agent200.dtb
```

5.1.3 Build and Install user's Yocto Toolchain

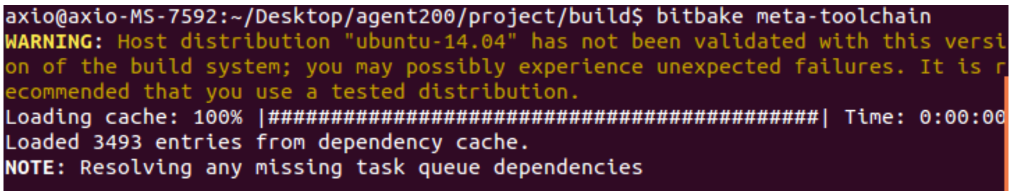
We have provided Yocto Toolchain in IRU131-SO BSP. However, if you want to build your own toolchain using Yocto development, you can follow the instructions on the host PC:

1. Change to *Yocto development* directory.
   ~$ source setup-environment build



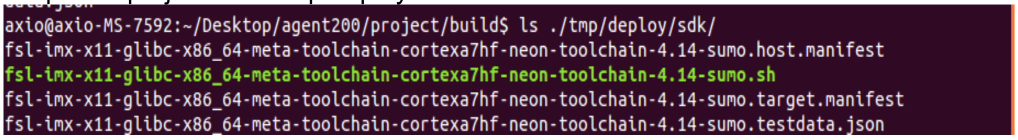
```
axio@axio-MS-7592:~/Desktop/agent200/project$ source setup-environment build

Welcome to Freescale Community BSP

The Yocto Project has extensive documentation about OE including a
reference manual which can be found at:
    http://yoctoproject.org/documentation

For more information about OpenEmbedded see their website:
    http://www.openembedded.org/

You can now run 'bitbake <target>'

Common targets are:
    core-image-minimal
    meta-toolchain
    meta-toolchain-sdk
    adt-installer
    meta-ide-support

Your configuration files at build have not been touched.
axio@axio-MS-7592:~/Desktop/agent200/project/build$
```

   ~$ bitbake meta-toolchain



```
axio@axio-MS-7592:~/Desktop/agent200/project/build$ bitbake meta-toolchain
WARNING: Host distribution "ubuntu-14.04" has not been validated with this versi
on of the build system; you may possibly experience unexpected failures. It is r
ecommended that you use a tested distribution.
Loading cache: 100% |############################################| Time: 0:00:00
Loaded 3493 entries from dependency cache.
NOTE: Resolving any missing task queue dependencies
```

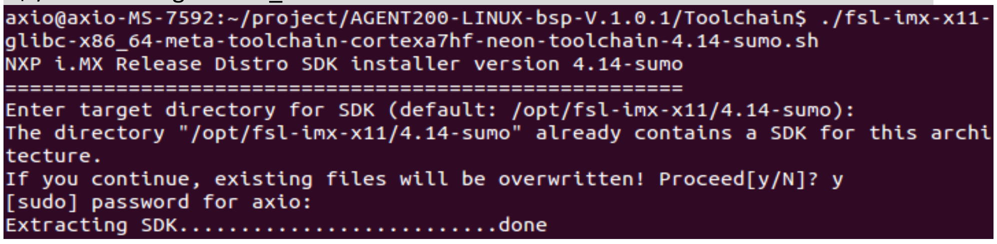
2. After these steps to generate the toolchain into the Build Directory, you can find the file path: ./project/build/tmp/deploy/sdk



```
axio@axio-MS-7592:~/Desktop/agent200/project/build$ ls ./tmp/deploy/sdk/
fsl-imx-x11-glibc-x86_64-meta-toolchain-cortexa7hf-neon-toolchain-4.14-sumo.host.manifest
fsl-imx-x11-glibc-x86_64-meta-toolchain-cortexa7hf-neon-toolchain-4.14-sumo.sh
fsl-imx-x11-glibc-x86_64-meta-toolchain-cortexa7hf-neon-toolchain-4.14-sumo.target.manifest
fsl-imx-x11-glibc-x86_64-meta-toolchain-cortexa7hf-neon-toolchain-4.14-sumo.testdata.json
```

Install the toolchain into your host system /opt directory.
Note: Installing the toolchain requires root authorization

```
~$./fsl-imx-x11-glibc-x86_64-meta-toolchain-cortexa7hf-neno-toolchain-4.14-sumo.sh
```



```
axio@axio-MS-7592:~/project/AGENT200-LINUX-bsp-V.1.0.1/Toolchain$ ./fsl-imx-x11-
glibc-x86_64-meta-toolchain-cortexa7hf-neon-toolchain-4.14-sumo.sh
NXP i.MX Release Distro SDK installer version 4.14-sumo
=======================================================
Enter target directory for SDK (default: /opt/fsl-imx-x11/4.14-sumo):
The directory "/opt/fsl-imx-x11/4.14-sumo" already contains a SDK for this archi
tecture.
If you continue, existing files will be overwritten! Proceed[y/N]? y
[sudo] password for axio:
Extracting SDK.........................done
```

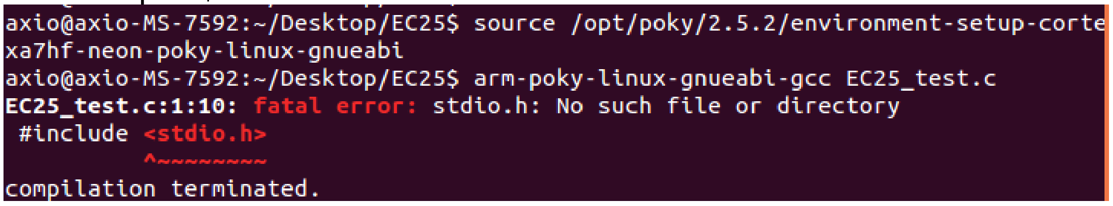
This page is intentionally left blank.

Appendix
Frequently Asked Questions

Q1. When I use toolchain to compile, I can't find " include" file.

A1: Refer to section 2.3 and 2.2.2 Setting up the Cross-Development Environment for detailed information.

For example: $CC hello.c -o hello

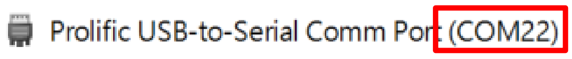


```
axio@axio-MS-7592:~/Desktop/EC25$ source /opt/poky/2.5.2/environment-setup-corte
xa7hf-neon-poky-linux-gnueabi
axio@axio-MS-7592:~/Desktop/EC25$ arm-poky-linux-gnueabi-gcc EC25_test.c
EC25_test.c:1:10: fatal error: stdio.h: No such file or directory
 #include <stdio.h>
          ^~~~~~~~~
compilation terminated.
```

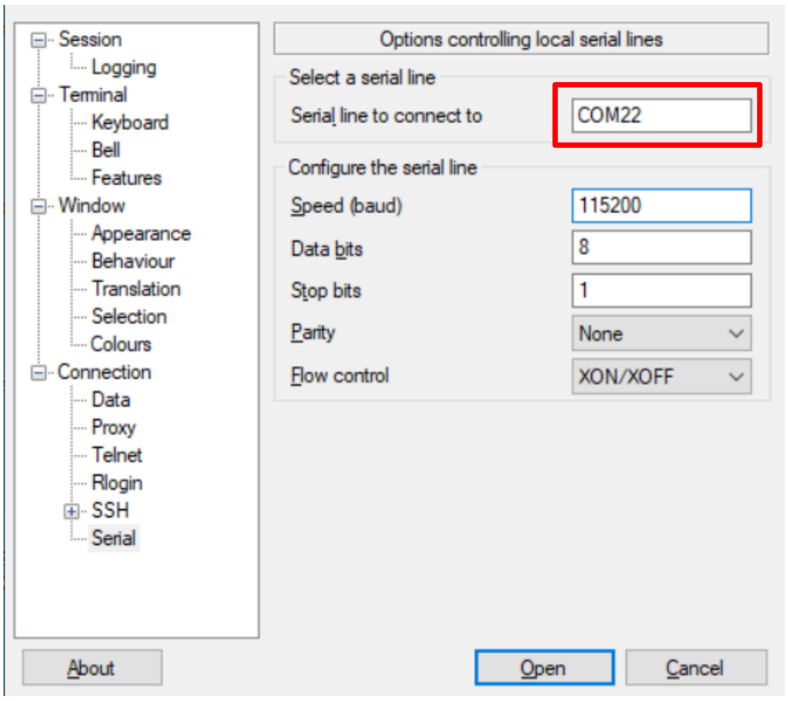
Q2. Why does the screen show nothing as below after I follow the steps described in section 2.1.1 to set up?

A2. Please follow steps as below

1. To check your power.
2. To check serial item "COM port" name and Device Manager "COM port" name are both the same as below



Prolific USB-to-Serial Comm Port (COM22)



Select a serial line

Serial line to connect to: COM22

Configure the serial line

Speed (baud): 115200

Data bits: 8

Stop bits: 1

Parity: None

Flow control: XON/XOFF

Q3. Why can’t transfer the file to FTP、TFTP、NFS after following the instructions, or disconnection.

A3: Check whether your firewall has been blocked in your host PC or router.