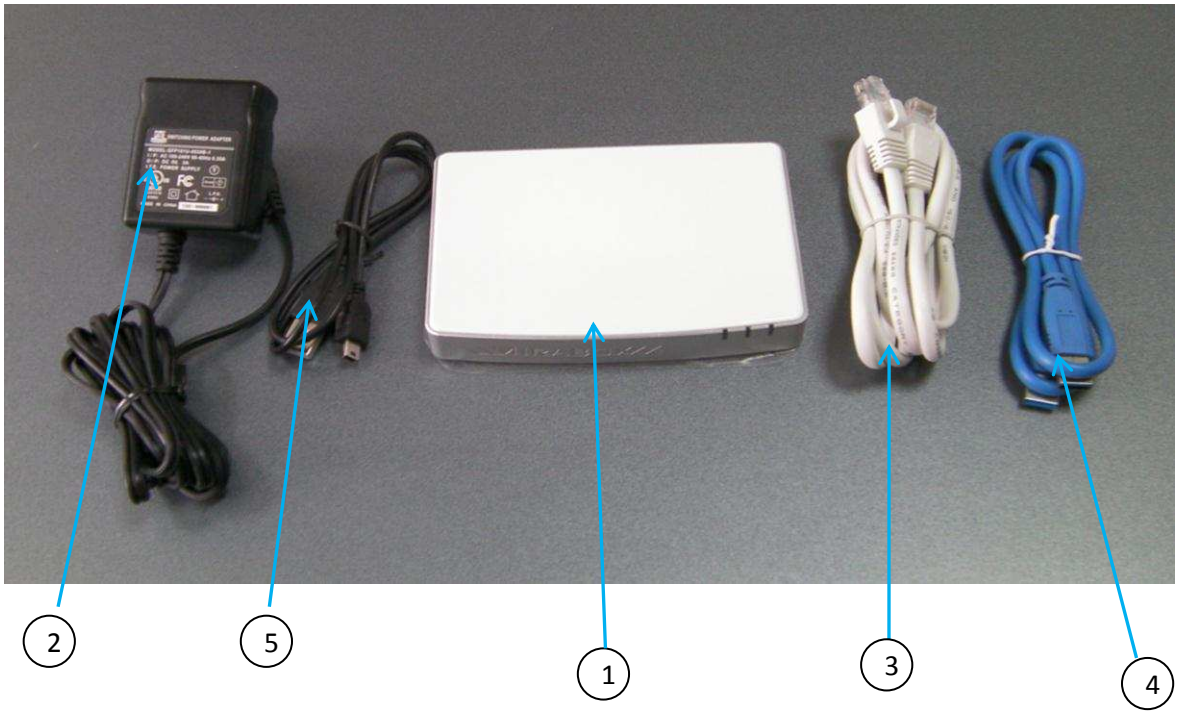
MiraBox User Guide -Sep18, 2012

Table of Contents

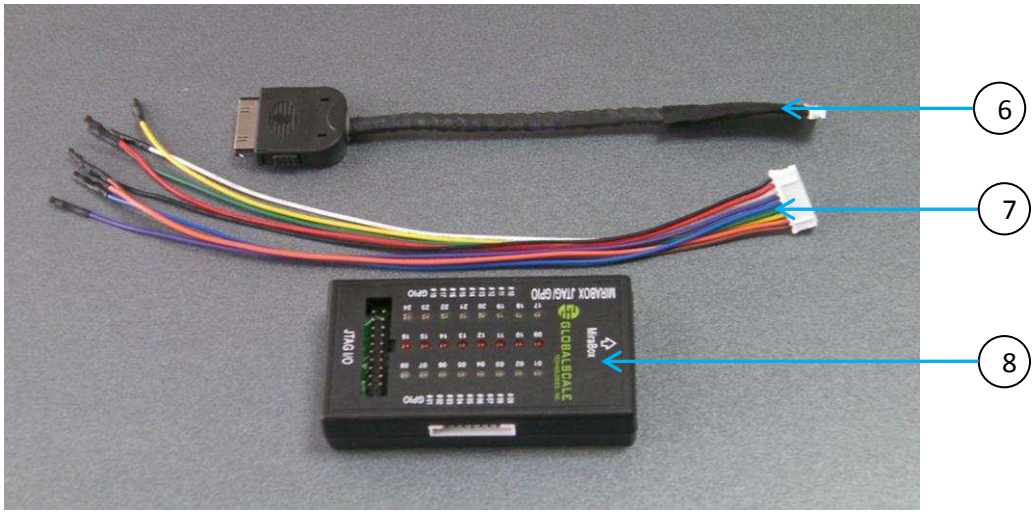
A. Package contents

1 · Standard package contents



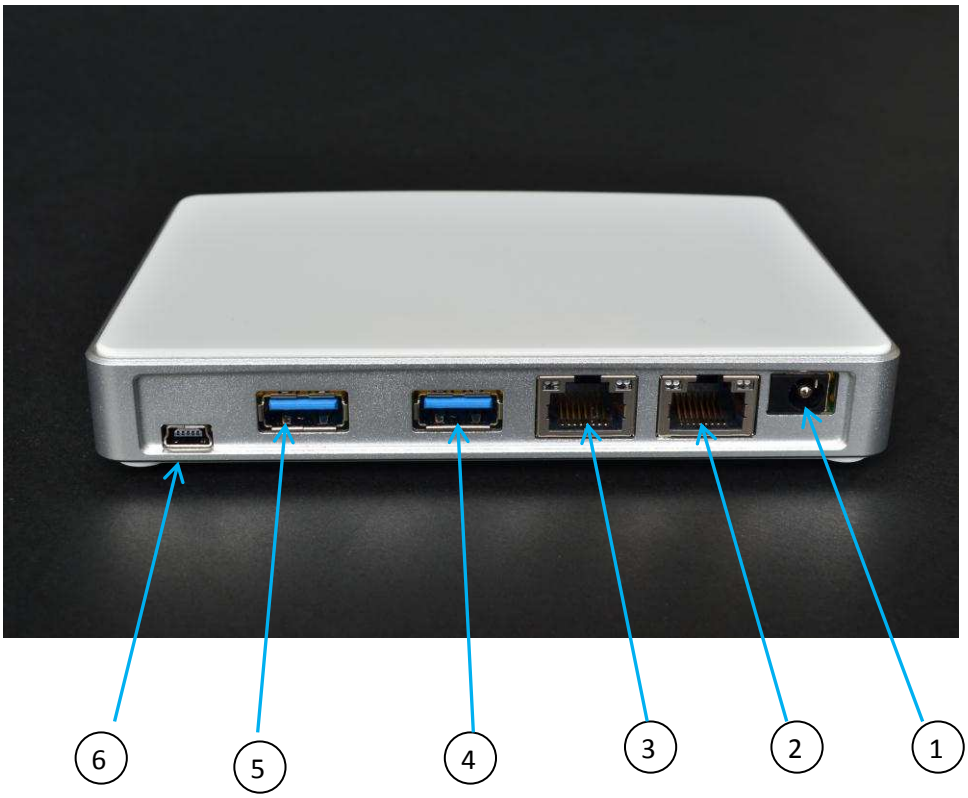
	MiraBoxstandard content List		Remark
1	MiraBox	1 unit	Mirabox main unit
2	AC-DC Power Adapter	1 pc	Input 90-240VAC / output 5V,3A DC
3	Ethernet Cable	1 pc	Cat 5e
4	USB3.0 Cable	1 pc	
5	Mini-USB Cable	1 pc	For debug console use
6	Quick reference card	1 pc	
7	Warranty card	1 pc	

2. Optional package contents



	MiraBox Optional Content List		
6	Multi I/O Cable	1 pc	Connect from MiraBox to JTAG/GPIO box
7	GPIO Cable	1 pc	For GPIO port connection
8	MiraBox JTAG/GPIO box	1 pc	External JTAG/GPIO box for debugging

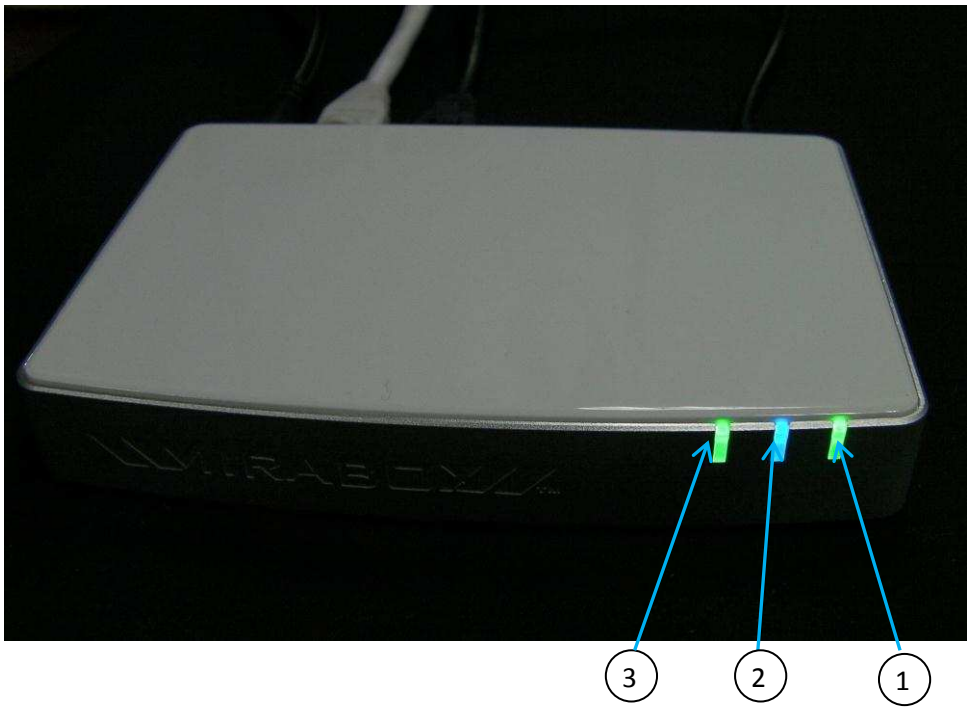
B. MiraBox appearance and connecting ports



Mirabox ports description

	Connection port	Description	Remark
1	Power Port	DC 5V/3A port	
2	RJ45 #1	Gigabit Ethernet port1	
3	RJ45 #2	Gigabit Ethernet port2	
4	USB 3.0 port#1	USB 3.0 high speed host	
5	USB 3.0 port#2	USB 3.0 high speed host	
6	Mini USB console port	Debug console	Connect to PC USB port
7	Multi-I/O port	JTAG and GPIO port	Connect to external JTAG/GPIO box for system development.
8	Micro SD slot	External Micro-SD slot	
9	Reset button hole	System reset button	Reset through GPIO

C. LED indication



LED indication table

	LED	Color/ Pattern	Description
1	Power on LED	Solid green	Upon power on, this LED lights up
2	WiFi AP	Blinking blue	Indicate WiFi AP mode is activated as default after boot up
		Off	WiFi AP mode is not activated
3	WiFi client	Blinking green	WiFi client mode is activated
		Off	WiFi client mode is not activated

D. System console and debugging

1. Driver and tool installation.

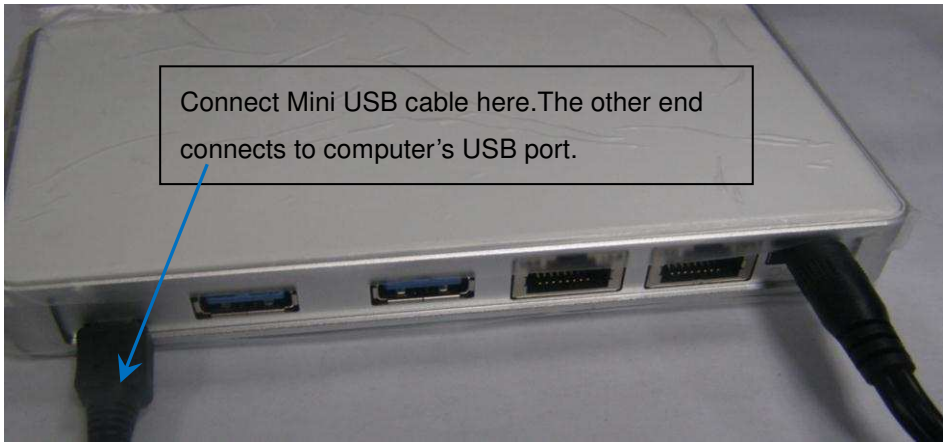
(1) Prepare one Windows PC

(2) Download the serial communication tool “putty.exe”.

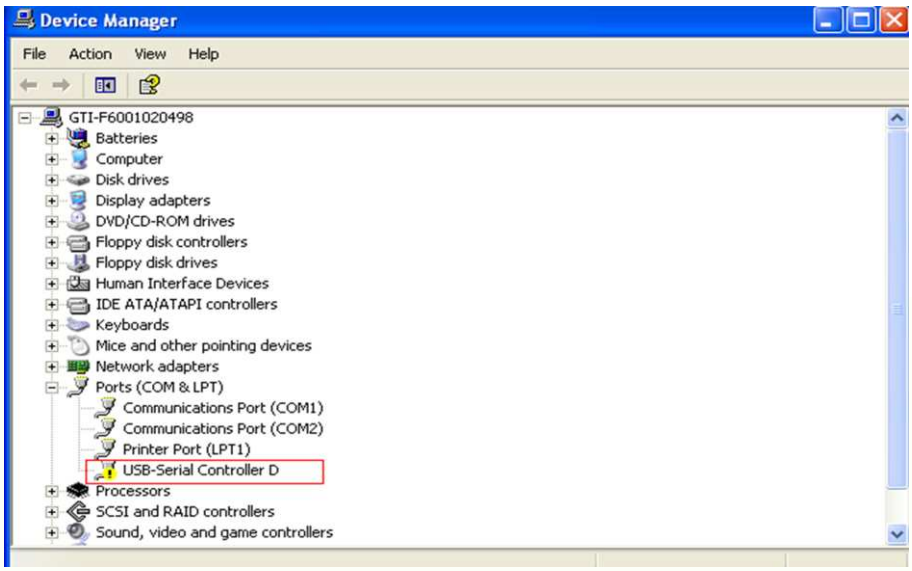
(3) Download the driver “2KXPVDock.exe”forProlific-USB-to-Serial-Comm-Portat our Website.
http://www.globalscaletechnologies.com/t-downloads.aspx

(4) Install driver on your PC (only for the first time)

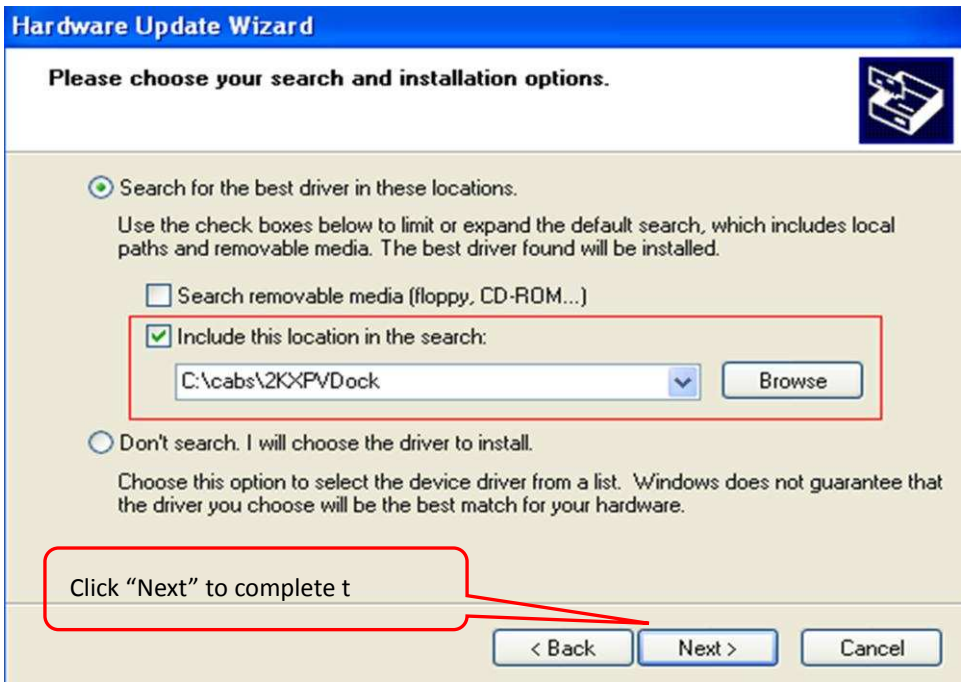
(4-1) Connect Mirabox to your Windows PC



(4-2) The first time when you connect Mirabox to the Windows computer you will be asked to install the driver



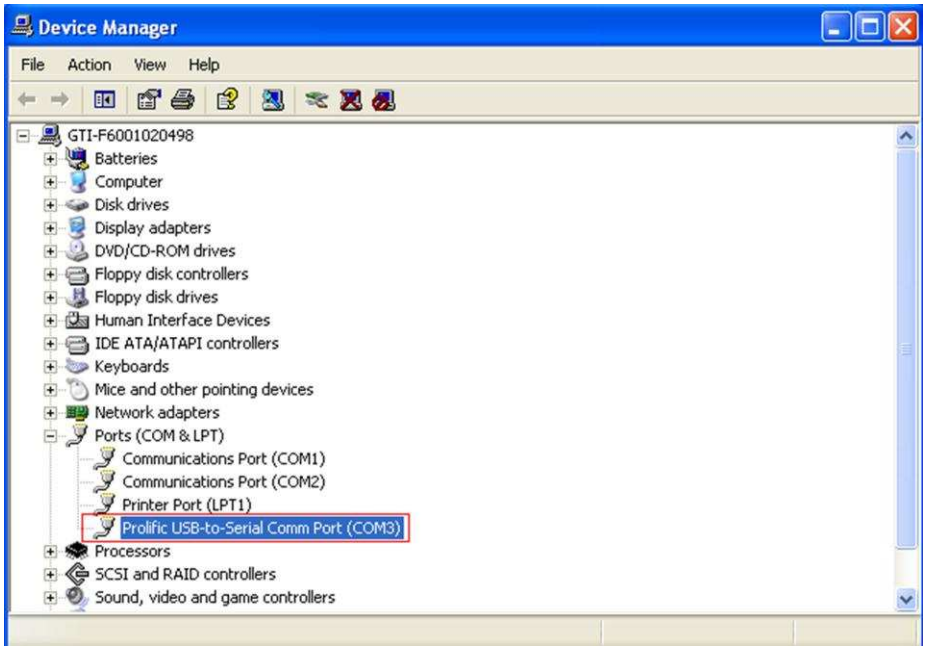
(4-3) Run file “2KXPVDock.exe”and the driver will be installed to the PATH C:\cabs\2KXPVDock”.



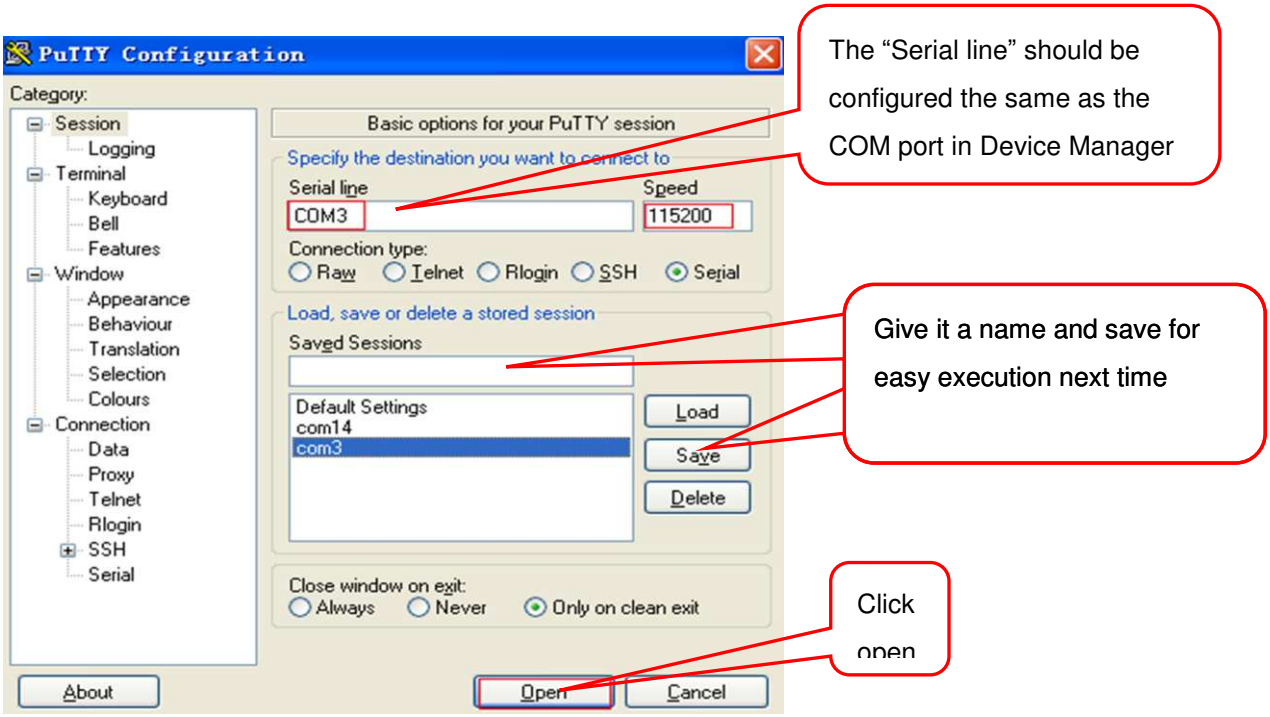
(5) Find out the com port of your debugger on your windows PC.

(5-1) Right click [My Computer] → [Properties] → [Hardware] → [Device Manager]

(5-2) Find out your com port as below (here is com3 for example)

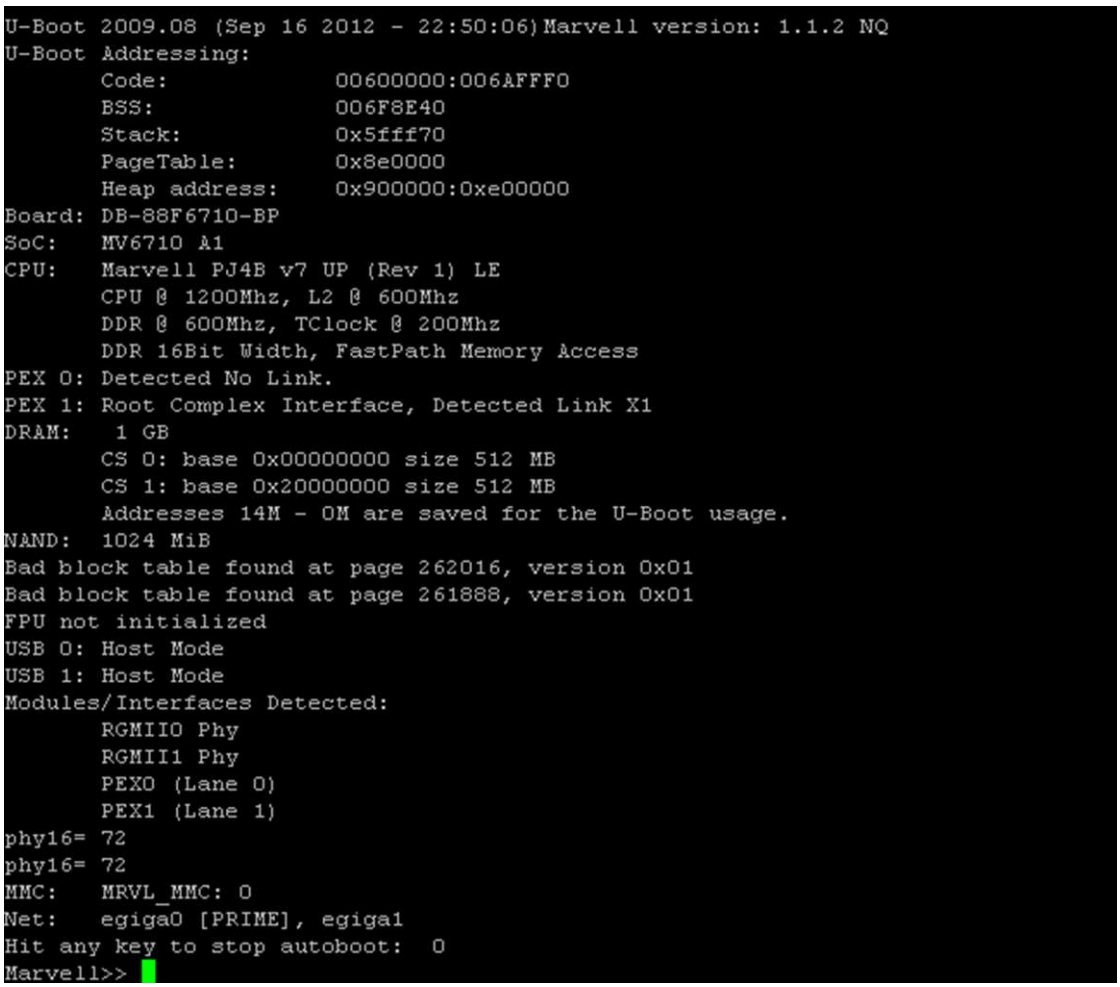


(6) Run “putty.exe” and configure as below on your Windows PC.



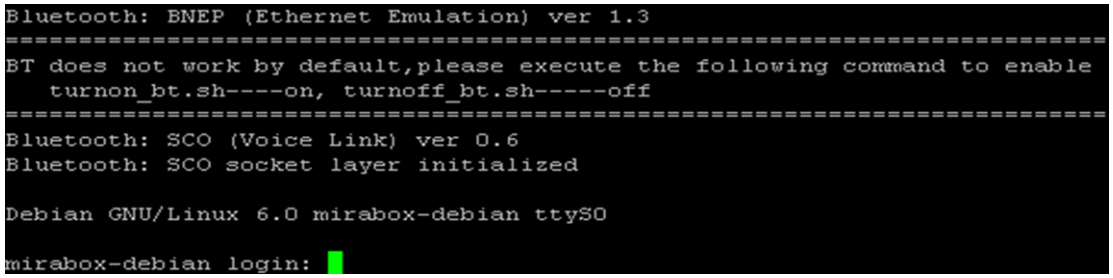
2. Debugging console.

(1) Power on the MiraBox and you will see messages on screen as below



(2) You can press any key to stop auto-boot when you see the bootdelay timer is counting down.
After entering the uboot prompt, you can also change the uboot environment variables such as bootdelay time, Ipaddr, serverip and so on.

(3) If no key was pressed to interrupt the uboot, it will continue running to the login screen where urges you to input the login name and password, here is the default login information.
Login :root
Password: nosoup4u



(4) Now you are the root user and have the full control of the mirabox

3. WiFi AP mode testing

MiraBox Server has a built-in WiFi module which is in compliance with 802.11 b/g/n standard. The WiFi can work as client or AP mode but only one at a time. The default mode is AP mode every time when it powers on, and the indication light D6 is blinking blue.

Here are steps for testing:

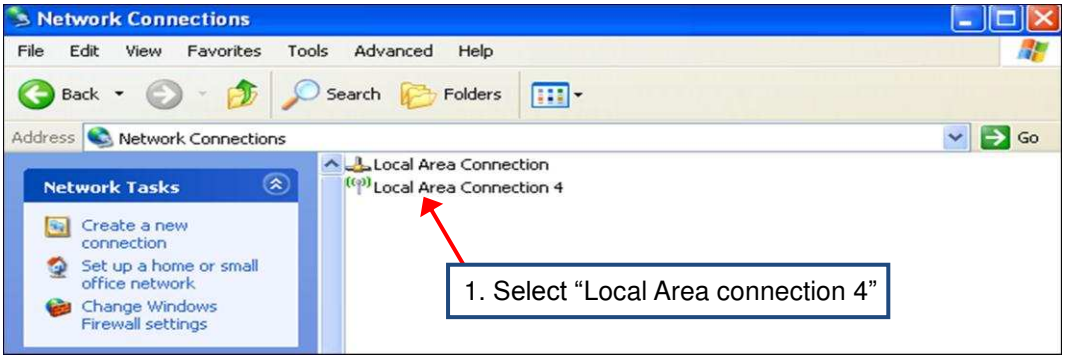
(1) Enter command and you will see message for uap0 device.

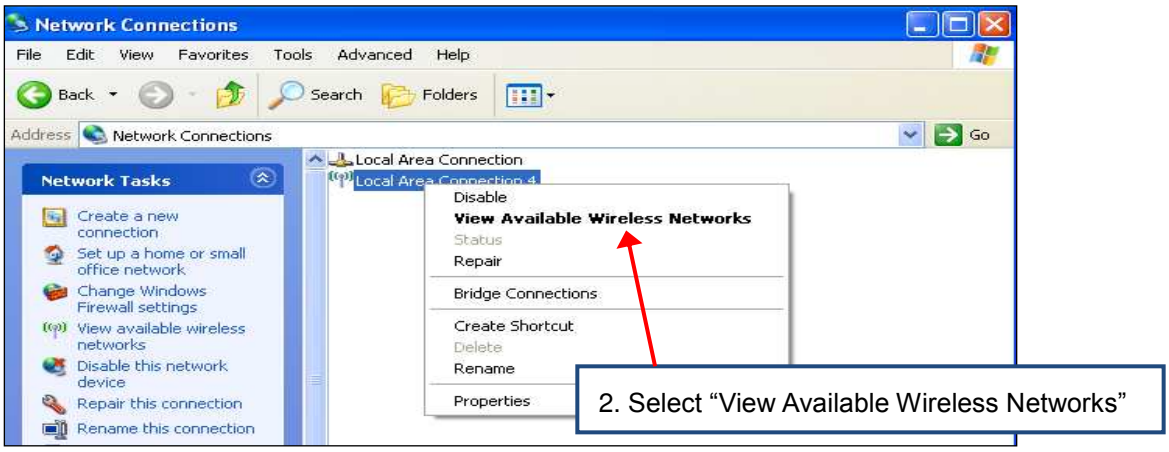
ifconfig

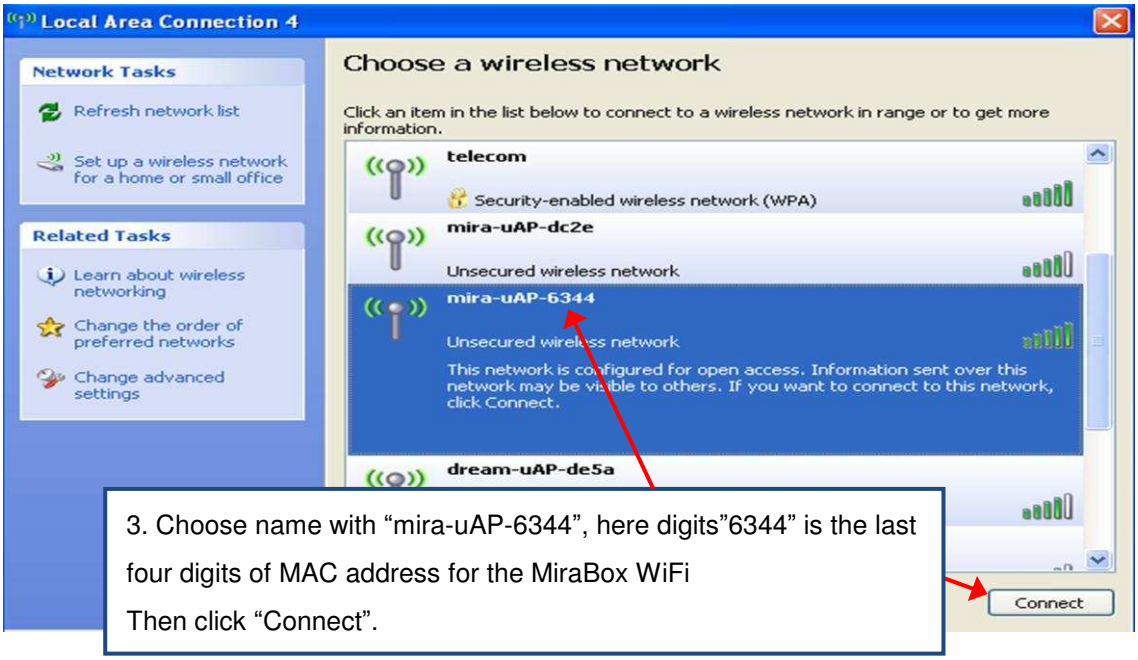
```
root@mirabox-debian:~# ifconfig
eth0      Link encap:Ethernet  HWaddr f0:ad:4e:01:a3:ea
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:512
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
          Interrupt:8

eth1      Link encap:Ethernet  HWaddr f0:ad:4e:01:a3:eb
          UP BROADCAST MULTICAST  MTU:1500  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:512
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
          Interrupt:10

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:16436  Metric:1
          RX packets:8 errors:0 dropped:0 overruns:0 frame:0
          TX packets:8 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:480 (480.0 B)  TX bytes:480 (480.0 B)

uap0      Link encap:Ethernet  HWaddr 00:90:a2:4a:63:44
          inet addr:192.168.1.1  Bcast:192.168.1.255  Mask:255.255.255.0
          UP BROADCAST MULTICAST  MTU:1500  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:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
```

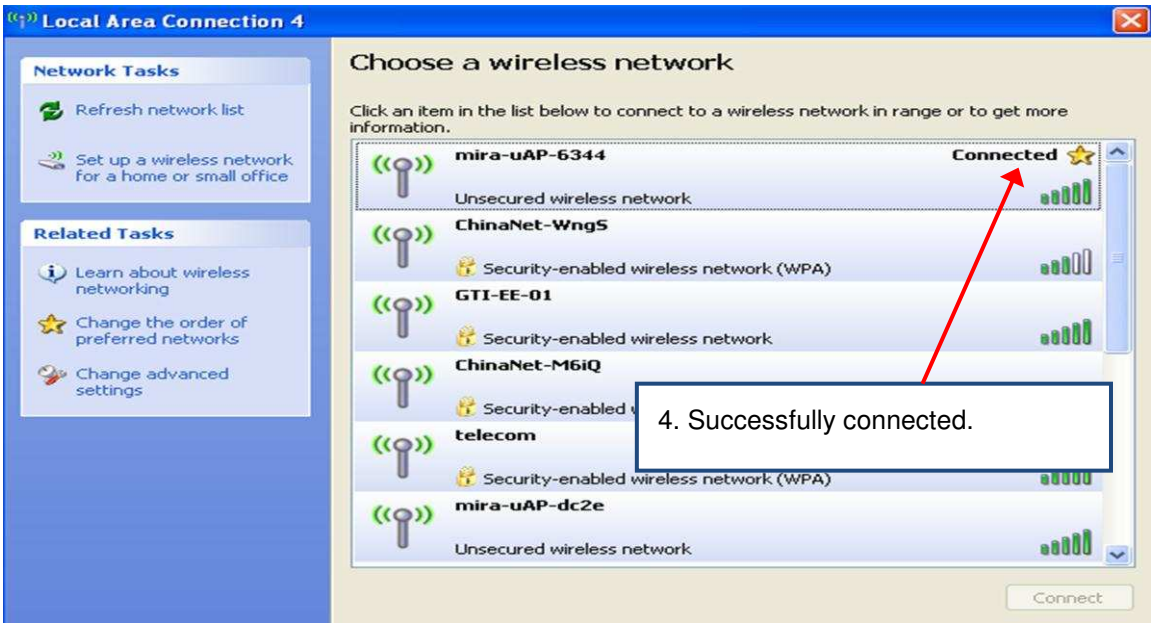
(2) Prepare one computer installed with a Wi-Fi Lan card, here we use a computer with a Windows XP operating system for example.

(3) Go to “Network Connections” as shown below..




Network Connections
File Edit View Favorites Tools Advanced Help
Back Search Folders
Address Network Connections Go
Network Tasks
Create a new connection
Set up a home or small office network
Change Windows Firewall settings
View available wireless networks
Disable this network device
Repair this connection
Rename this connection
Local Area Connection
Local Area Connection 4
Disable
View Available Wireless Networks
Status
Repair
Bridge Connections
Create Shortcut
Delete
Rename
Properties
2. Select “View Available Wireless Networks”


Local Area Connection 4
Network Tasks
Refresh network list
Set up a wireless network for a home or small office
Related Tasks
Learn about wireless networking
Change the order of preferred networks
Change advanced settings
Choose a wireless network
Click an item in the list below to connect to a wireless network in range or to get more information.
telecom
Security-enabled wireless network (WPA)
mira-uAP-dc2e
Unsecured wireless network
mira-uAP-6344
Unsecured wireless network
This network is configured for open access. Information sent over this network may be visible to others. If you want to connect to this network, click Connect.
dream-uAP-de5a
Connect
3. Choose name with “mira-uAP-6344”, here digits”6344” is the last four digits of MAC address for the MiraBox WiFi
Then click “Connect”.


Local Area Connection 4
Network Tasks
Refresh network list
Set up a wireless network for a home or small office
Related Tasks
Learn about wireless networking
Change the order of preferred networks
Change advanced settings
Choose a wireless network
Click an item in the list below to connect to a wireless network in range or to get more information.
mira-uAP-6344
Connected
Unsecured wireless network
ChinaNet-WngS
Security-enabled wireless network (WPA)
GTI-EE-01
Security-enabled wireless network
ChinaNet-M6iQ
telecom
Security-enabled wireless network (WPA)
mira-uAP-dc2e
Unsecured wireless network
Connect
4. Successfully connected.

4. WIFI Client mode testing

(1) Switch to WiFi client mode by giving command as following.

wlan.sh

```
root@mirabox-debian:~# wlan.sh
Stopping very small Busybox based DHCP server: Stopped /usr/sbin/udhcpd (pid 1911).
udhcpd.
Stopping DNS forwarder and DHCP server: dnsmasq.
WLAN FW already running! Skip FW download
WLAN FW is active
ADDRCONF(NETDEV_UP): mlan0: link is not ready
```

When done successfully, the LED(D7) is blinking green and mlan0 is activated.

(2) To check mlan0 with ifconfig command

ifconfig

```
root@mirabox-debian:~# ifconfig
eth0      Link encap:Ethernet  HWaddr f0:ad:4e:01:a3:ea
          UP BROADCAST MULTICAST  MTU:1500  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:512
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
          Interrupt:8

eth1      Link encap:Ethernet  HWaddr f0:ad:4e:01:a3:eb
          UP BROADCAST MULTICAST  MTU:1500  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:512
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
          Interrupt:10

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:16436  Metric:1
          RX packets:8 errors:0 dropped:0 overruns:0 frame:0
          TX packets:8 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:480 (480.0 B)  TX bytes:480 (480.0 B)

mlan0     Link encap:Ethernet  HWaddr 00:90:a2:4a:63:44
          UP BROADCAST MULTICAST  MTU:1500  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:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
```

(3) Scan for WiFi AP near-by

iwlist mlan0 scan

```
root@mirabox-debian:~# iwlist mlan0 scan
mlan0     Scan completed :
          Cell 01 - Address: 08:18:1A:90:97:E9
                    ESSID:"ChinaNet-WngS"
                    Mode:Master
                    Frequency:2.412 GHz (Channel 1)
                    Quality:3/5  Signal level:-68 dBm  Noise level:-96 dBm
                    Encryption key:on
                    Bit Rates:1 Mb/s; 2 Mb/s; 5.5 Mb/s; 11 Mb/s; 6 Mb/s
                              9 Mb/s; 12 Mb/s; 18 Mb/s; 24 Mb/s; 36 Mb/s
                              48 Mb/s; 54 Mb/s
                    Extra:Beacon interval=100
                    IE: WPA Version 1
                        Group Cipher : CCMP
                        Pairwise Ciphers (1) : CCMP
                        Authentication Suites (1) : PSK
                    IE: Unknown: DD180050F2020101000003A4000027A4000042435E0062322F00
                    IE: Unknown: DD1E00904C330C181EFF000000000000000000000000000000000000000000000000
                    IE: Unknown: DD1A00904C3401000000000000000000000000000000000000000000
                    IE: Unknown: DD0600E04C020160
                    Extra:band=bg
          Cell 02 - Address: 40:16:9F:44:CB:2C
                    ESSID:"GTI-EE-01" [2]
                    Mode:Master
                    Frequency:2.412 GHz (Channel 1)
                    Quality:5/5  Signal level:-42 dBm  Noise level:-96 dBm
                    Encryption key:on
                    Bit Rates:1 Mb/s; 2 Mb/s; 5.5 Mb/s; 11 Mb/s; 6 Mb/s
                              9 Mb/s; 12 Mb/s; 18 Mb/s; 24 Mb/s; 36 Mb/s
                              48 Mb/s; 54 Mb/s
                    Extra:Beacon interval=100
                    IE: IEEE 802.11i/WPA2 Version 1
                        Group Cipher : TKIP
                        Pairwise Ciphers (2) : CCMP TKIP
                        Authentication Suites (1) : PSK
                    IE: Unknown: DD180050F2020101020003A4000027A4000042435E0062322F00
                    IE: Unknown: DD0900037F01010000FF7F
                    IE: Unknown: DD810050F204104A0001101044000102103B0001031047001000000000000
00000040169F44CB2C1021000754502D4C494E4B10230009544C2D57523734304E1024000331 2E3010420003312E30
4000800060050F2040001101100195769726 56C65737320526F7574657220544C2D57523734304E1008000200861030
0101
                    Extra:band=bg
          Cell 03 - Address: C8:3A:35:48:A2:28
                    ESSID:"Tenda_48A228" [3]
                    Mode:Master
```

(4) Connect to WiFi AP

iwconfig mlan0 essid<AP's ID>

dhclient mlan0

ifconfig mlan0

#ping 192.168.1.1

The below screenshot is an example for how to connect a mlan device. If you can see IP address (for example: 192.168.1.102) means you have already connected and got an IP.

```
root@mirabox-debian:~# iwconfig mlan0 essid "dream-uAP-de5a"
ADDRCONF(NETDEV_CHANGE): mlan0: link becomes ready
root@mirabox-debian:~# dhclient mlan0
Reloading /etc/samba/smb.conf: smbd only.
root@mirabox-debian:~# ifconfig mlan0
mlan0     Link encap:Ethernet  HWaddr 00:90:a2:4a:63:44
          inet addr:192.168.1.102  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::290:a2ff:fe4a:6344/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:5 errors:0 dropped:0 overruns:0 frame:0
          TX packets:31 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:794 (794.0 B)  TX bytes:3750 (3.6 KiB)

root@mirabox-debian:~# ping 192.168.1.1
PING 192.168.1.1 (192.168.1.1) 56(84) bytes of data.
64 bytes from 192.168.1.1: icmp_req=1 ttl=64 time=4.74 ms
64 bytes from 192.168.1.1: icmp_req=2 ttl=64 time=5.03 ms
64 bytes from 192.168.1.1: icmp_req=3 ttl=64 time=4.47 ms
64 bytes from 192.168.1.1: icmp_req=4 ttl=64 time=4.49 ms
64 bytes from 192.168.1.1: icmp_req=5 ttl=64 time=3.65 ms
^C
--- 192.168.1.1 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4040ms
rtt min/avg/max/mdev = 3.655/4.478/5.032/0.462 ms
```

(5) Switch back to WiFi AP mode

There are two useful script files:

Client_ap.sh – switch from wifi client mode to wifi AP mode

Wlan.sh – switch from wifi AP mode to wifi client mode

```
root@mirabox-debian:~# client_ap.sh
WLAN FW already running! Skip FW download
WLAN FW is active
ADDRCONF(NETDEV_UP): uap0: link is not ready
SSID setting successful
BSS started!
Starting very small Busybox based DHCP server: Starting /usr/sbin/udhcpd...
udhcpd.
Starting DNS forwarder and DHCP server: dnsmasq.
root@mirabox-debian:~# wlan.sh
Stopping very small Busybox based DHCP server: Stopped /usr/sbin/udhcpd (pid 10278).
udhcpd.
Stopping DNS forwarder and DHCP server: dnsmasq.
WLAN FW already running! Skip FW download
WLAN FW is active
ADDRCONF(NETDEV_UP): mlan0: link is not ready
```

5. Bluetooth testing

There is bluetooth chip built-in Mirabox but disabled as default.

(1) Enable the Bluetooth function.

turnon_bt.sh

hcitool -i hci0 dev

```
root@mirabox-debian:~# turnon_bt.sh
BT FW is active(0)
BT FW already downloaded!
root@mirabox-debian:~# hcitool -i hci0 dev
Devices:
        hci0    00:90:A2:4A:63:45
```

If you see the message like this means your Bluetooth is up and run

(2) Scan near-by Bluetooth devices

a. Please turn on the Bluetooth device such as your mobile phone, BT earphone or laptop computer then put it into “to be discovered” mode

b. Giving command as following on your Mirabox

hciconfig

hcitool-i hci0 scan --flush

Normally, the Bluetooth devices near-by will be searched and shown as below.

```
root@mirabox-debian:~# hciconfig
hci0:   Type: BR/EDR  Bus: SDIO
        BD Address: 00:90:A2:4A:63:71  ACL MTU: 1021:7  SCO MTU: 120:6
        UP RUNNING PSCAN
        RX bytes:1348 acl:0 sco:0 events:37 errors:0
        TX bytes:1527 acl:0 sco:0 commands:37 errors:0

root@mirabox-debian:~# hcitool -i hci0 scan --flush
Scanning ...
        00:25:66:9E:3E:35       GT-S5230C
root@mirabox-debian:~# sdptool browse
Inquiring ...
```

This is a mobile phone with Bluetooth enabled that is found

sdptool browse

This command shows the protocol of Bluetooth device found.

```
root@mirabox-debian:~# sdptool browse
Inquiring ...
Browsing 00:25:66:9E:3E:35 ...
Service RecHandle: 0x10000
Service Class ID List:
  "PnP Information" (0x1200)

Service Name: HSP PS Audio Gateway
Service RecHandle: 0x10001
Service Class ID List:
  "Headset Audio Gateway" (0x1112)
  "Generic Audio" (0x1203)
```

6. Gigabit Ethernet ports

(1) Connect the two Gigabit Ethernet ports to Gigabit switch by Network cables. Normally it will get an IP address assigned by DHCP.

Enter command as below to check:

ifconfig

```
Mon Sep 17 06:20:41 2012  -0.252600 seconds
root@mirabox-debian:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:50:43:9d:2b:2d
          inet addr:10.10.10.152  Bcast:10.10.10.255  Mask:255.255.255.0
          inet6 addr: fe80::250:43ff:fe9d:2b2d/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1309 errors:0 dropped:0 overruns:0 frame:0
          TX packets:138 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:512
          RX bytes:136008 (132.8 KiB)  TX bytes:15562 (15.1 KiB)
          Interrupt:8

eth1      Link encap:Ethernet  HWaddr 00:50:43:81:2b:2d
          inet addr:10.10.10.189  Bcast:10.10.10.255  Mask:255.255.255.0
          inet6 addr: fe80::250:43ff:fe81:2b2d/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1359 errors:0 dropped:0 overruns:0 frame:0
          TX packets:11 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:512
          RX bytes:141431 (138.1 KiB)  TX bytes:1278 (1.2 KiB)
          Interrupt:10

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:16436  Metric:1
          RX packets:8 errors:0 dropped:0 overruns:0 frame:0
          TX packets:8 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:480 (480.0 B)  TX bytes:480 (480.0 B)

uap0      Link encap:Ethernet  HWaddr 94:db:c9:d2:ae:ec
          inet addr:192.168.1.1  Bcast:192.168.1.255  Mask:255.255.255.0
          UP BROADCAST MULTICAST  MTU:1500  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:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
```

(2) Testing the speed of Gigabit Ethernet ports

ethtool eth0

Normally you will see the information as below:

```
root@mirabox-debian:~# ethtool eth0
Settings for eth0:
        Supported ports: [ TP MII ]
        Supported link modes:   10baseT/Half 10baseT/Full
                                100baseT/Half 100baseT/Full
                                1000baseT/Full
        Supports auto-negotiation: Yes
        Advertised link modes:  10baseT/Half 10baseT/Full
                                100baseT/Half 100baseT/Full
                                1000baseT/Half 1000baseT/Full
        Advertised pause frame use: No
        Advertised auto-negotiation: No
        Link partner advertised link modes:  10baseT/Half 10baseT/Full
                                             100baseT/Half 100baseT/Full
                                             1000baseT/Half 1000baseT/Full
        Link partner advertised pause frame use: No
        Link partner advertised auto-negotiation: Yes
        Speed: 1000Mb/s
        Duplex: Full
        Port: MII
        PHYAD: 0
        Transceiver: internal
        Auto-negotiation: on
        Link detected: yes
```

7. USB 3.0 port

(1) Plug in the usb3.0 hard disk or flash disk to the USB port then you can see some driver messages as below of this device

```
root@mirabox-debian:~# hub 3-0:1.0: unable to enumerate USB device on port 2
usb 3-4: new SuperSpeed USB device using xhci_hcd and address 3
xhci_hcd 0000:01:01.0: WARN: short transfer on control ep
xhci_hcd 0000:01:01.0: WARN: short transfer on control ep
xhci_hcd 0000:01:01.0: WARN: short transfer on control ep
xhci_hcd 0000:01:01.0: WARN: short transfer on control ep
scsi4 : usb-storage 3-4:1.0
scsi 4:0:0:0: Direct-Access     Seagate  Portable         SF06 PQ: 0 ANSI: 4
sd 4:0:0:0: [sdc] 976773164 512-byte logical blocks: (500 GB/465 GiB)
sd 4:0:0:0: [sdc] Write Protect is off
sd 4:0:0:0: [sdc] Assuming drive cache: write through
sd 4:0:0:0: [sdc] Assuming drive cache: write through
 sdc: sdc1
sd 4:0:0:0: [sdc] Assuming drive cache: write through
sd 4:0:0:0: [sdc] Attached SCSI disk
```

(2) View the usb3.0 hard disk

fdisk-l

you will see below messages of the usb3.0 device which is usually shown as /dev/sdb*、/dev/sdc*.

```
Disk /dev/sdc: 500.1 GB, 500107859968 bytes
255 heads, 63 sectors/track, 60801 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x165060a4

   Device Boot      Start         End      Blocks   Id  System
/dev/sdc1               1       60802   488384032+   7  HPFS/NTFS
```

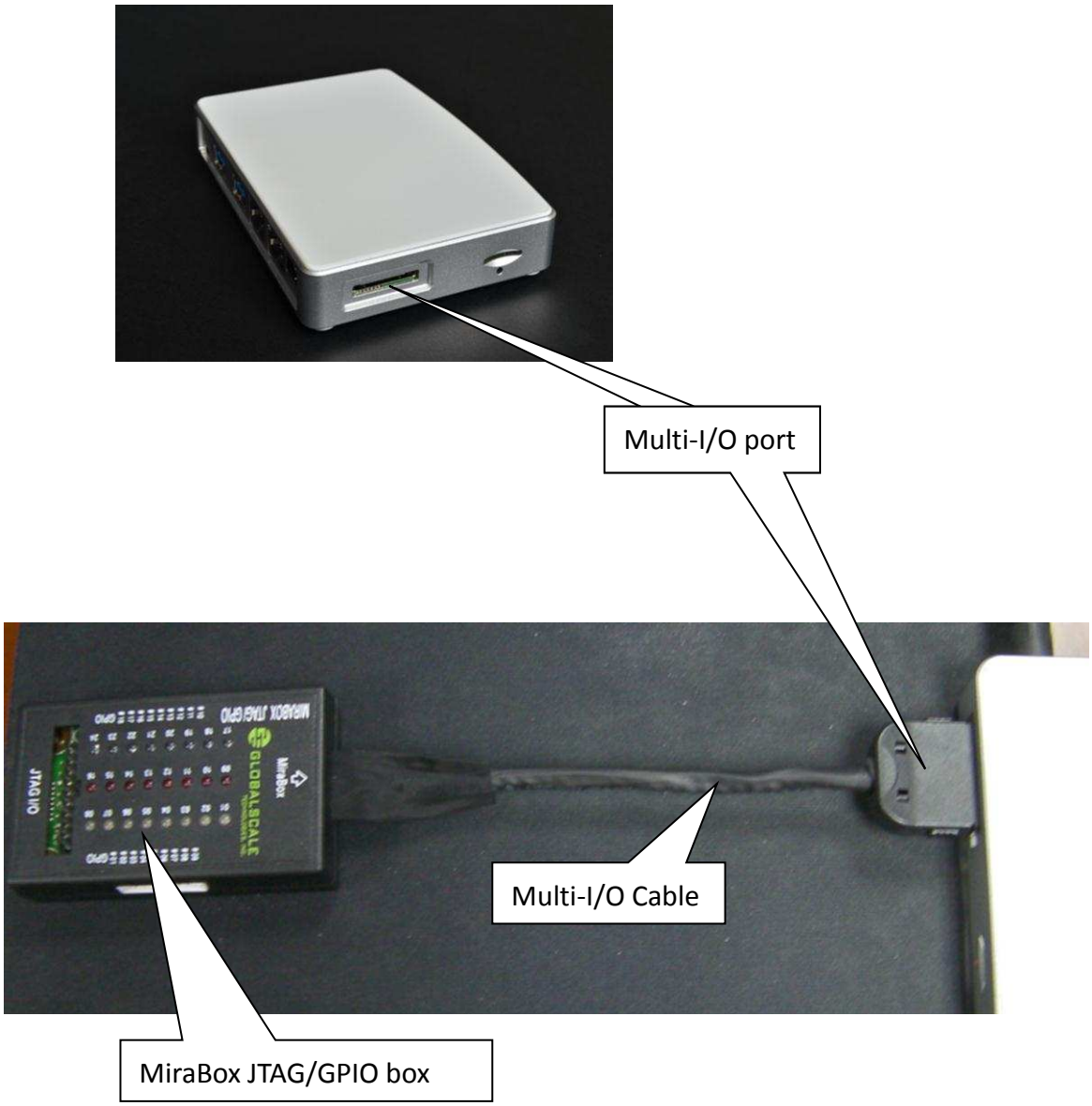
(3) Testing the access speed of the usb3.0 ports

hdparm-t /dev/sdc1

```
root@mirabox-debian:~# hdparm -t /dev/sdc1

/dev/sdc1:
 Timing buffered disk reads: 118 MB in  3.02 seconds =  39.10 MB/sec
```

8. Multi-IO port

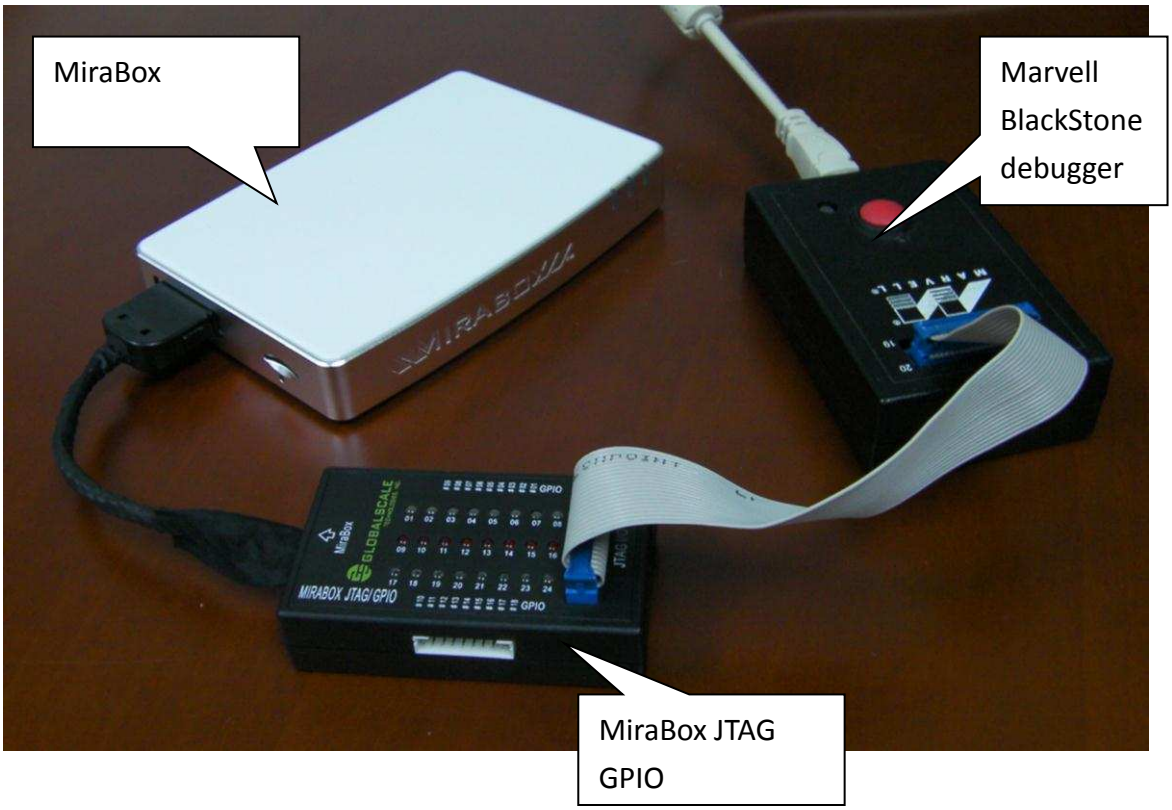
Connect the MIRABOX JTAG/GPIO box to the Multi-IO port via Multi-IO cable.



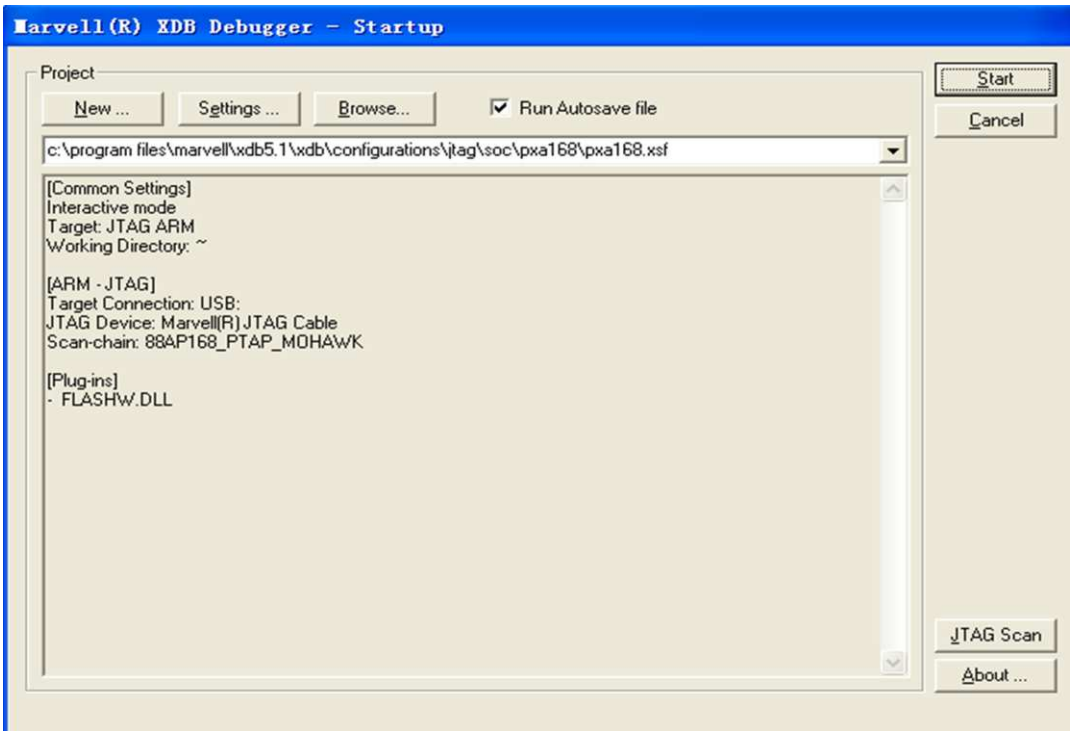
9. Multi-IO port- JTAG interface

Please connect the debugger to the 20-pin JTAG slot as shown in the picture, here we use the Marvell BlackStone debugger and XDB(Marvell eXtreme Debugger 5.1) software on Windows PC for example.

Debugger connection



Run XDB on Windows PC



10. GPIO interface and control

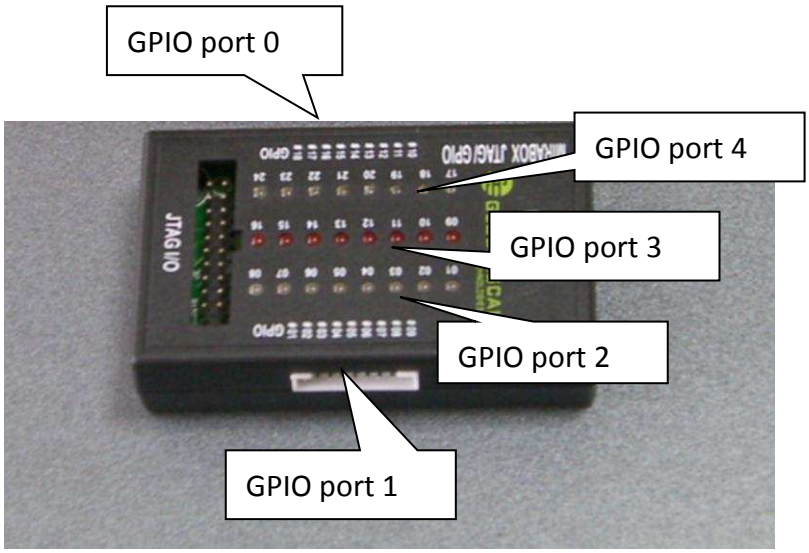
(1) LED demo program

There is a demo program to show the GPIO LEDs by entering command as below:

ledtest1

This will light up all the LEDs

(2) Write a simple program to control the LED



In this GPIO box we use NXP PCA9505 I/O port expansion IC to provide 40-bit parallel input/output GPIOs. Please download the datasheet to get the detailed description for this chip.

a. Prepare a pc with linux Fedora 14 installed, and download the cross tool chain **"arm-marvell-linux-gnueabi-vfp.tar.bz2"** from our website
http://www.plugcomputer.org/downloads/d2plug/
or
http://www.globalscaletechnologies.com

b. Install and configure the tool chain

Copy "**arm-marvell-linux-gnueabi-vfp.tar.bz2**" to directory /opt

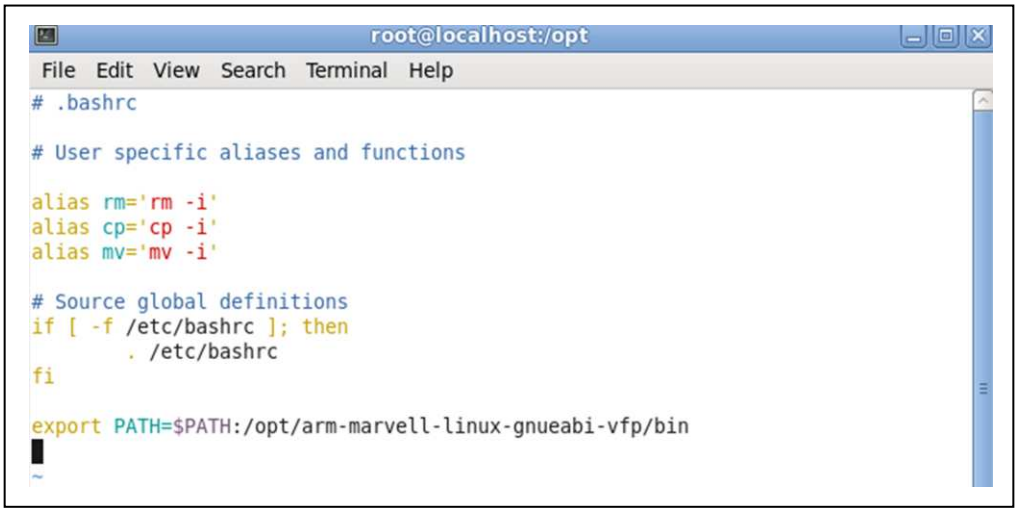
cd /opt

tar –jxf arm-marvell-linux-gnueabi-vfp.tar.bz2

ls

vim /root/.bashrc

Add "**export PATH=$PATH:/opt/arm-marvell-linux-gnueabi-vfp/bin"** to last line as below, then save and exit.



Enable the PATH

source /root/.bash_profile

c. Write a program

Here is the sample program named **"i2c_led.c"**

```
/*i2c_led.c*GTI--Globalscaletechnologies.,INC**/
        #include <stdio.h>
        #include <linux/types.h>
        #include <stdlib.h>
        #include <fcntl.h>
        #include <unistd.h>
        #include <sys/types.h>
        #include <sys/ioctl.h>
        #include <errno.h>
        #define I2C_RETRIES 0x0701
        #define I2C_TIMEOUT 0x0702
        #define I2C_RDWR 0x0707
        #define I2C_M_RD 0x0001
        #define I2C_M_NOSTART          0x4000      /* if I2C_FUNC_PROTOCOL_MANGLING */
    #define I2C_M_REV_DIR_ADDR  0x2000      /* if I2C_FUNC_PROTOCOL_MANGLING */
        #define I2C_M_IGNORE_NAK 0x1000     /* if I2C_FUNC_PROTOCOL_MANGLING */
    #define I2C_M_NO_RD_ACK          0x0800      /* if I2C_FUNC_PROTOCOL_MANGLING */
        #define I2C_M_RECV_LEN          0x0400      /* length will be first received byte */
/*********define struct i2c_rdwr_ioctl_data and struct i2c_msg，they must be consistent to kernel*******/
struct i2c_msg
        {
unsigned short addr;
unsigned short flags;
        #define I2C_M_TEN 0x0010
unsigned short len;
unsigned char *buf;
        };
struct i2c_rdwr_ioctl_data
        {
struct i2c_msg *msgs;
intnmsgs;/* the nmsgs decide the num of start signal*/
        };
/***********the main program***********/
int main()
        {
intfd,ret;
```

```
                struct i2c_rdwr_ioctl_data e2prom_data;
fd=open("/dev/i2c-0",O_RDWR);/*      /dev/i2c-0 is registered to the system */
if(fd<0)
                {
perror("open error");
                }
                e2prom_data.nmsgs=2;
                e2prom_data.msgs=(struct i2c_msg*)malloc(e2prom_data.nmsgs*sizeof(struct i2c_msg));
if(!e2prom_data.msgs)
                {
perror("malloc error");
exit(1);
                }
ioctl(fd,I2C_TIMEOUT,1);/*timeout */
ioctl(fd,I2C_RETRIES,2);/*retries times*/
/***write data to e2prom**/
                e2prom_data.nmsgs=1;
                (e2prom_data.msgs[0]).len=2;
                (e2prom_data.msgs[0]).addr=0x25;//e2prom device address
                (e2prom_data.msgs[0]).flags=0;    //write
                (e2prom_data.msgs[0]).buf=(unsigned char*)malloc(2);
   /***********    control the GPIO OP-0 ************************************/
                (e2prom_data.msgs[0]).buf[0]=0x18;// e2prom write address
                (e2prom_data.msgs[0]).buf[1]=0x0;//the data to write
                        ret=ioctl(fd,I2C_RDWR,(unsigned long)&e2prom_data);
if(ret<0)
                {
perror("ioctl error2");
                }
   /***********    control the GPIO OP-1 ************************************/
(e2prom_data.msgs[0]).buf[0]=0x19;// e2prom write address
                (e2prom_data.msgs[0]).buf[1]=0x0;//the data to write
                        ret=ioctl(fd,I2C_RDWR,(unsigned long)&e2prom_data);
if(ret<0)
                {
perror("ioctl error2");
                }
```

```
/***********    control the GPIO OP-2 ************************************/
     //turn on LED
(e2prom_data.msgs[0]).buf[0]=0x1a;// e2prom write address
                    (e2prom_data.msgs[0]).buf[1]=0x0;//the data to write
                         ret=ioctl(fd,I2C_RDWR,(unsigned long)&e2prom_data);
if(ret<0)
                    {
perror("ioctl error3");
                    }
   //turn off LED
sleep(1);                                   //delay 1 second
                    (e2prom_data.msgs[0]).buf[0]=0x1a;// e2prom write address
                    (e2prom_data.msgs[0]).buf[1]=0xFF;//the data to write
                         ret=ioctl(fd,I2C_RDWR,(unsigned long)&e2prom_data);
if(ret<0)
                    {
perror("ioctl error3");
                    }
   /***********    control the GPIO OP-3 ************************************/
   //turn on LED
sleep(1);
                    (e2prom_data.msgs[0]).buf[0]=0x1b;// e2prom write address
                    (e2prom_data.msgs[0]).buf[1]=0x0;//the data to write
                         ret=ioctl(fd,I2C_RDWR,(unsigned long)&e2prom_data);
if(ret<0)
                    {
perror("ioctl error4");
                    }
   //turn off LED
sleep(1);
(e2prom_data.msgs[0]).buf[0]=0x1b;// e2prom write address
                    (e2prom_data.msgs[0]).buf[1]=0xFF;//the data to write
                         ret=ioctl(fd,I2C_RDWR,(unsigned long)&e2prom_data);
if(ret<0)
                    {
perror("ioctl error4");
                    }
```

```
   /***********    control the GPIO OP-4 ************************************/
   //turn on LED
sleep(1);
                        (e2prom_data.msgs[0]).buf[0]=0x1c;// e2prom write address
                        (e2prom_data.msgs[0]).buf[1]=0x0;//the data to write
                             ret=ioctl(fd,I2C_RDWR,(unsigned long)&e2prom_data);
if(ret<0)
                        {
perror("ioctl error5");
                        }
   //turn off LED
sleep(1);
                        (e2prom_data.msgs[0]).buf[0]=0x1c;// e2prom write address
                        (e2prom_data.msgs[0]).buf[1]=0xFF;// the data to write
                             ret=ioctl(fd,I2C_RDWR,(unsigned long)&e2prom_data);
if(ret<0)
                        {
perror("ioctl error5");
                        }
            printf("testing ok\n");
close(fd);
return 0;
            }
```

Give this program the name **“i2c_led.c”** and save to **“/home”** directory then compile it .

```
# cd /home
# arm-marvell-linux-gnueabi-gcc -o led i2c_led.c
```

Then copy the executable file “led” to Mirabox and run it. You can see the LED is controlled by yourself.

11. Reset

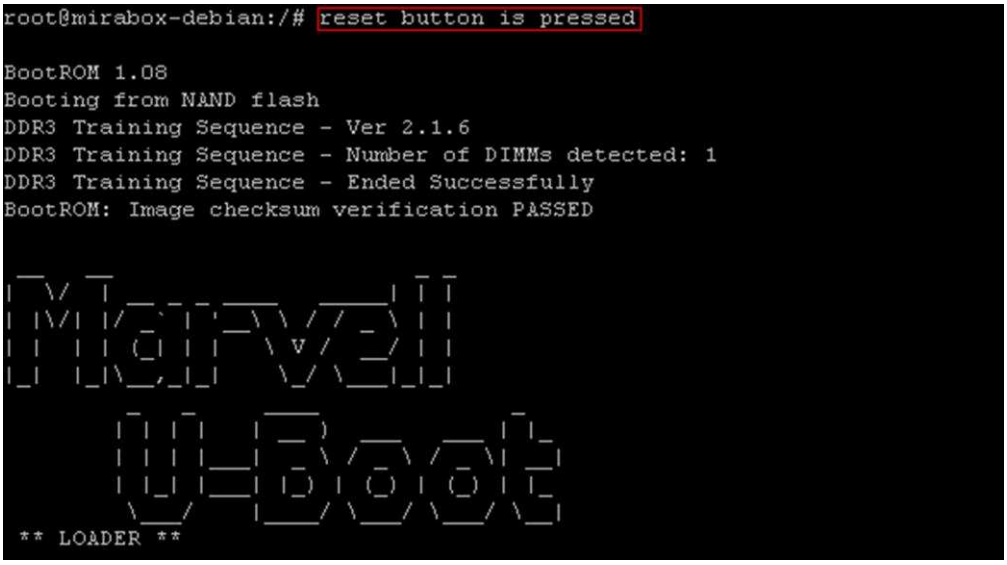
6、GPIO reset button hole.

(1) GPIO reset button

The reset button is connected to one GPIO which means controlled by software.

When you press down this button by using a sharp pin, it arouses attention of CPU then CPU asserts the master reset low signal to start the system reset session.

Below is the reboot screen after reset



Note: Since this reset function is controlled by software so it has no effect during boot-up session.

(2) More push button functions

System developer can program this button to have more functions by different press-then-release delay time.

12. Download sites

To download the files for MiraBox server, please visit:
http://www.globalscaletechnologies.com/t-downloads.aspx

Other useful resource links are:
http://www.plugcomputer.org/
http://plugcomputer.org/plugwiki/index.php/GuruPlug