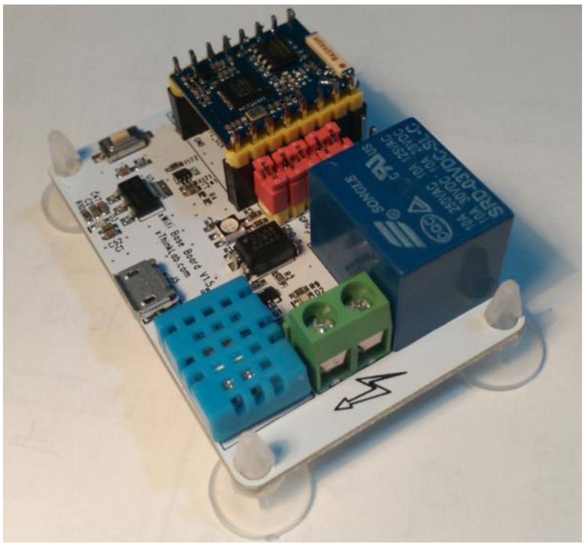
xWifi Simple Guide V1.0-MT7681



The purpose of this document is to give an user guide of xWifi kit.

1.Compiling firmware code

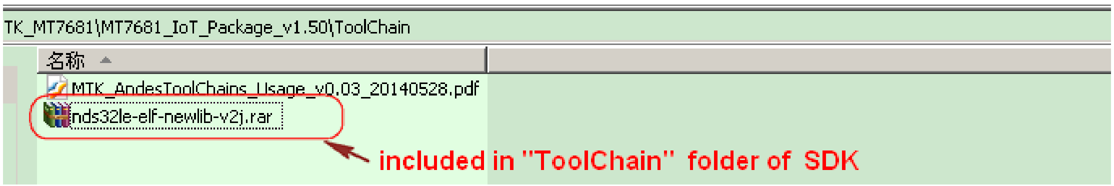
There are several ways to compile firmware source code ,we introduce two ways below based on Windows XP system .We can take either **MinGW** or **AndeSight** develop tools as the compiling environment.

There are some compile commands as follows:

```
make   b=0   clean   //clean object files
make   b=0           //create recovery   bin
make   b=1   clean   // clean object files
make   b=1           //create sta   bin
make   b=2   clean   // clean object files
make   b=2           //create ap   bin
```

1.1 Based on MinGW Toolchain

Step 1: Find the special Toolchain for MT7681



TK_MT7681\MT7681_IoT_Package_v1.50\ToolChain

名称

MTK_AndesToolChains_Usage_v0.03_20140528.pdf

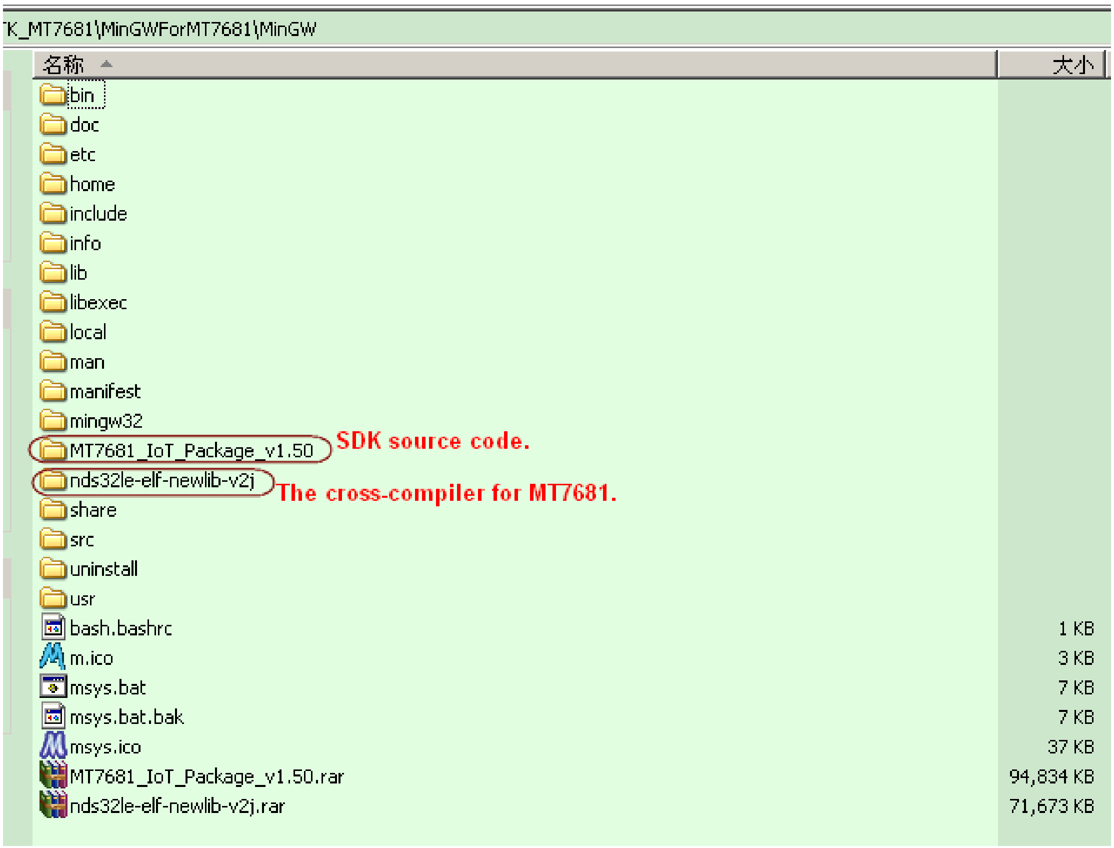
nds32le-elf-newlib-v2j.rar

included in "ToolChain" folder of SDK

Step 2: Access to link below and Download MinGW toolchain we have verified

http://pan.baidu.com/s/1jGqWG1g

Step 3: Copy Toolchain and SDK file into “MinGW” directory and extract it.



K_MT7681\MinGWForMT7681\MinGW

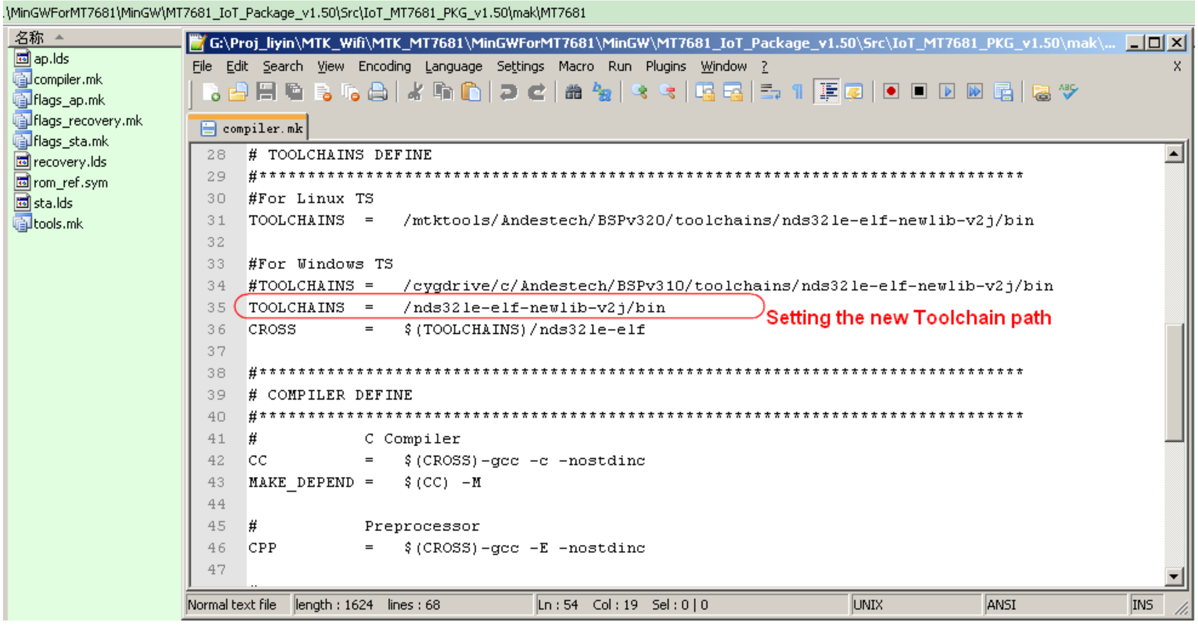
名称	大小
bin	
doc	
etc	
home	
include	
info	
lib	
libexec	
local	
man	
manifest	
mingw32	
MT7681_IoT_Package_v1.50	
nds32le-elf-newlib-v2j	
share	
src	
uninstall	
usr	
bash.bashrc	1 KB
m.ico	3 KB
msys.bat	7 KB
msys.bat.bak	7 KB
msys.ico	37 KB
MT7681_IoT_Package_v1.50.rar	94,834 KB
nds32le-elf-newlib-v2j.rar	71,673 KB

SDK source code.

The cross-compiler for MT7681.

Step 4:

change the directory to "\MinGW\MT7681_IoT_Package_v1.50\Src\IoT_MT7681_PKG_v1.50\mak\MT7681" ,open "compiler.mk" file and Set the new toolchain path like below .



.\MinGWForMT7681\MinGW\MT7681_IoT_Package_v1.50\Src\IoT_MT7681_PKG_v1.50\mak\MT7681

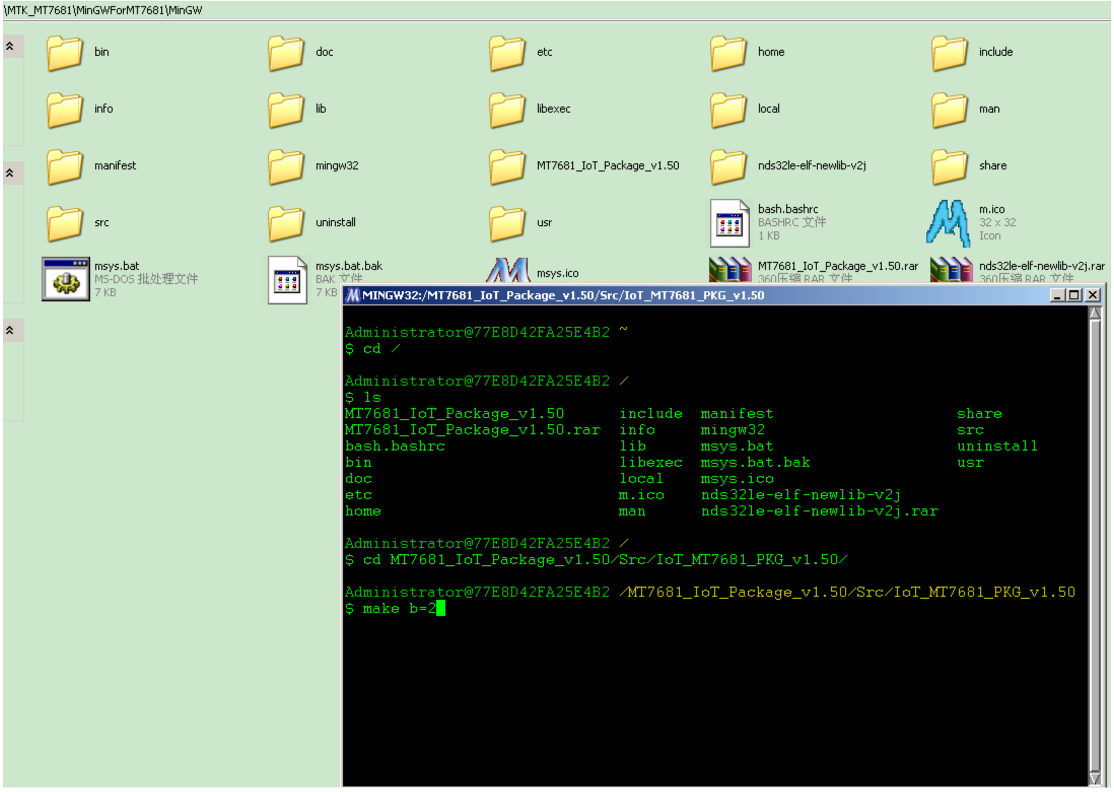
G:\Proj_liyin\MTK_Wifi\MTK_MT7681\MinGWForMT7681\MinGW\MT7681_IoT_Package_v1.50\Src\IoT_MT7681_PKG_v1.50\mak\...

```
# TOOLCHAINS DEFINE
#***************************************************************************
#For Linux TS
TOOLCHAINS  =   /mtktools/Andestech/BSPv320/toolchains/nds32le-elf-newlib-v2j/bin

#For Windows TS
#TOOLCHAINS =   /cygdrive/c/Andestech/BSPv310/toolchains/nds32le-elf-newlib-v2j/bin
TOOLCHAINS  =   /nds32le-elf-newlib-v2j/bin
CROSS       =   $(TOOLCHAINS)/nds32le-elf

#***************************************************************************
# COMPILER DEFINE
#***************************************************************************
#           C Compiler
CC          =   $(CROSS)-gcc -c -nostdinc
MAKE_DEPEND =   $(CC) -M

#           Preprocessor
CPP         =   $(CROSS)-gcc -E -nostdinc
```

Setting the new Toolchain path

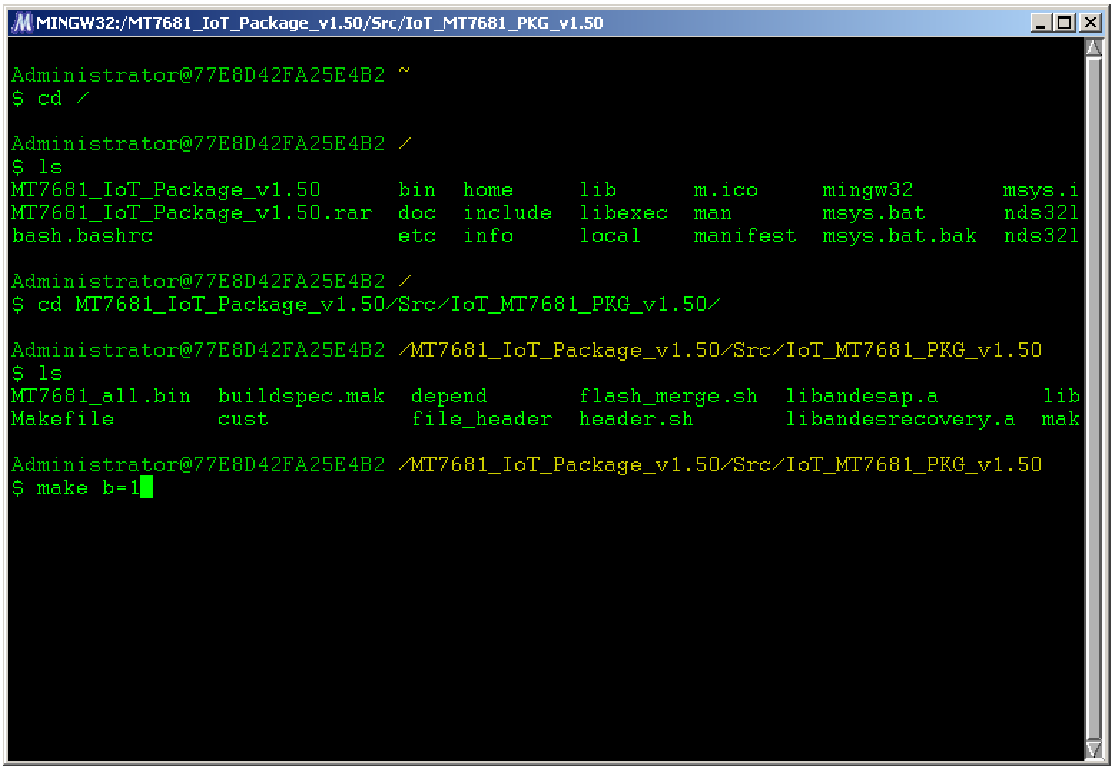
Normal text file | length : 1624 lines : 68 | Ln : 54 Col : 19 Sel : 0 | 0 | UNIX | ANSI | INS

Step 5: Run "msys.bat" file in "MinGW" directory:

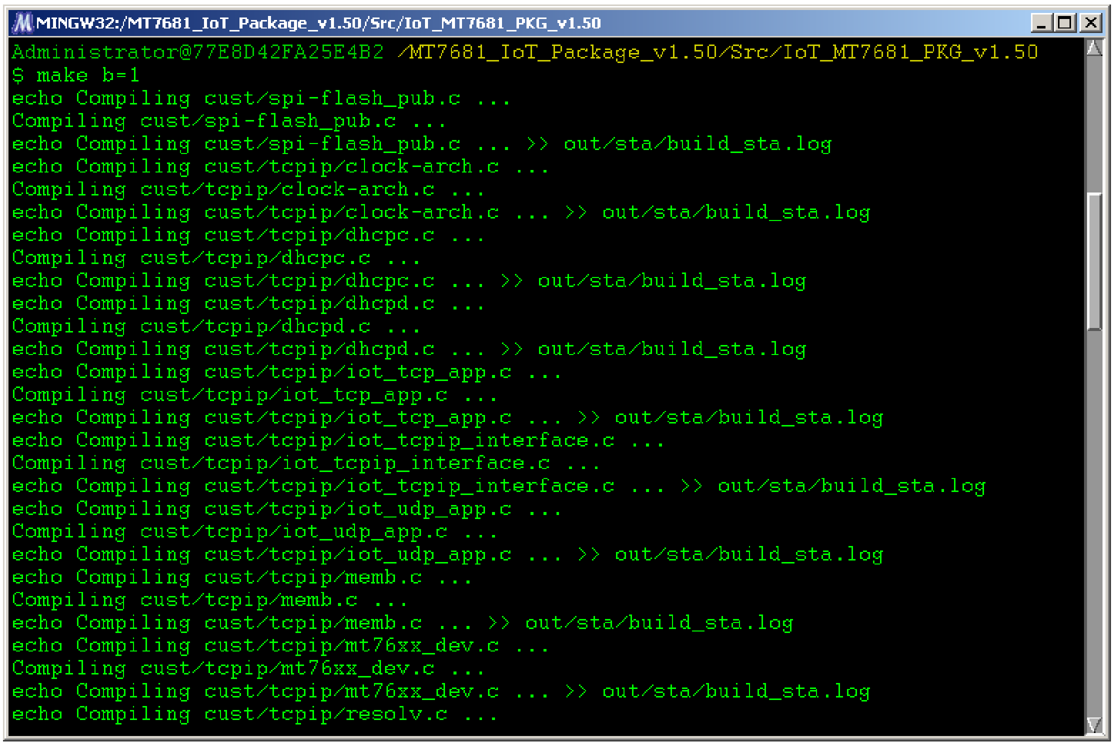


```
Administrator@77E8D42FA25E4B2 ~
$ cd /

Administrator@77E8D42FA25E4B2 /
$ ls
MT7681_IoT_Package_v1.50      include  manifest                    share
MT7681_IoT_Package_v1.50.rar  info     mingw32                     src
bash.bashrc                   lib      msys.bat                    uninstall
bin                           libexec  msys.bat.bak                usr
doc                           local    msys.ico
etc                           m.ico    nds32le-elf-newlib-v2j
home                          man      nds32le-elf-newlib-v2j.rar

Administrator@77E8D42FA25E4B2 /
$ cd MT7681_IoT_Package_v1.50/Src/IoT_MT7681_PKG_v1.50/

Administrator@77E8D42FA25E4B2 /MT7681_IoT_Package_v1.50/Src/IoT_MT7681_PKG_v1.50
$ make b=2
```

Step 6: Enter into "\MT7681_IoT_Package_v1.50\Src\IoT_MT7681_PKG_v1.50" directory ,compiling source code .

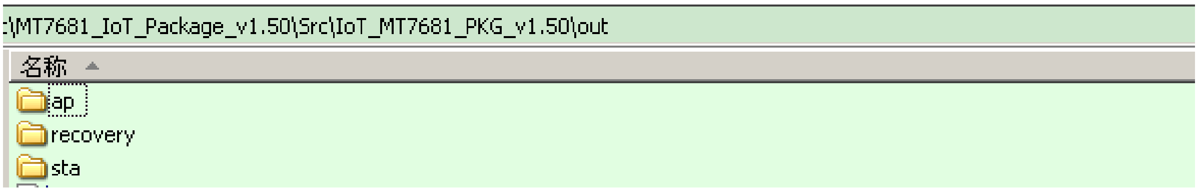


```
Administrator@77E8D42FA25E4B2 ~
$ cd /

Administrator@77E8D42FA25E4B2 /
$ ls
MT7681_IoT_Package_v1.50      bin  home     lib      m.ico     mingw32       msys.i
MT7681_IoT_Package_v1.50.rar  doc  include  libexec  man       msys.bat      nds321
bash.bashrc                   etc  info     local    manifest  msys.bat.bak  nds321

Administrator@77E8D42FA25E4B2 /
$ cd MT7681_IoT_Package_v1.50/Src/IoT_MT7681_PKG_v1.50/

Administrator@77E8D42FA25E4B2 /MT7681_IoT_Package_v1.50/Src/IoT_MT7681_PKG_v1.50
$ ls
MT7681_all.bin  buildspec.mak  depend       flash_merge.sh  libandesap.a          lib
Makefile        cust           file_header  header.sh       libandesrecovery.a    mak

Administrator@77E8D42FA25E4B2 /MT7681_IoT_Package_v1.50/Src/IoT_MT7681_PKG_v1.50
$ make b=1
```



```
Administrator@77E8D42FA25E4B2 /MT7681_IoT_Package_v1.50/Src/IoT_MT7681_PKG_v1.50
$ make b=1
echo Compiling cust/spi-flash_pub.c ...
Compiling cust/spi-flash_pub.c ...
echo Compiling cust/spi-flash_pub.c ... >> out/sta/build_sta.log
echo Compiling cust/tcpip/clock-arch.c ...
Compiling cust/tcpip/clock-arch.c ...
echo Compiling cust/tcpip/clock-arch.c ... >> out/sta/build_sta.log
echo Compiling cust/tcpip/dhcpc.c ...
Compiling cust/tcpip/dhcpc.c ...
echo Compiling cust/tcpip/dhcpc.c ... >> out/sta/build_sta.log
echo Compiling cust/tcpip/dhcpd.c ...
Compiling cust/tcpip/dhcpd.c ...
echo Compiling cust/tcpip/dhcpd.c ... >> out/sta/build_sta.log
echo Compiling cust/tcpip/iot_tcp_app.c ...
Compiling cust/tcpip/iot_tcp_app.c ...
echo Compiling cust/tcpip/iot_tcp_app.c ... >> out/sta/build_sta.log
echo Compiling cust/tcpip/iot_tcpip_interface.c ...
Compiling cust/tcpip/iot_tcpip_interface.c ...
echo Compiling cust/tcpip/iot_tcpip_interface.c ... >> out/sta/build_sta.log
echo Compiling cust/tcpip/iot_udp_app.c ...
Compiling cust/tcpip/iot_udp_app.c ...
echo Compiling cust/tcpip/iot_udp_app.c ... >> out/sta/build_sta.log
echo Compiling cust/tcpip/memb.c ...
Compiling cust/tcpip/memb.c ...
echo Compiling cust/tcpip/memb.c ... >> out/sta/build_sta.log
echo Compiling cust/tcpip/mt76xx_dev.c ...
Compiling cust/tcpip/mt76xx_dev.c ...
echo Compiling cust/tcpip/mt76xx_dev.c ... >> out/sta/build_sta.log
echo Compiling cust/tcpip/resolv.c ...
```

The directory of generated binary files is in


:\MT7681_IoT_Package_v1.50\Src\IoT_MT7681_PKG_v1.50\out

1.2 Based on AndeSight Toolchain

Step 1:

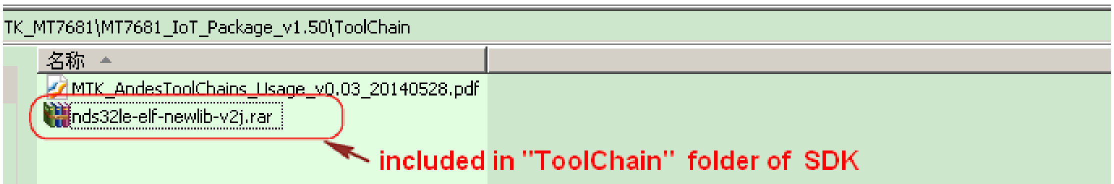
Access to the link below ,register and request a Evaluation Version of AndeSight

http://www.andestech.com/en/download/andesight-download.htm

when you register successfully ,you will receive a Email with download information from Andes.Download it and install it .

Step 2:

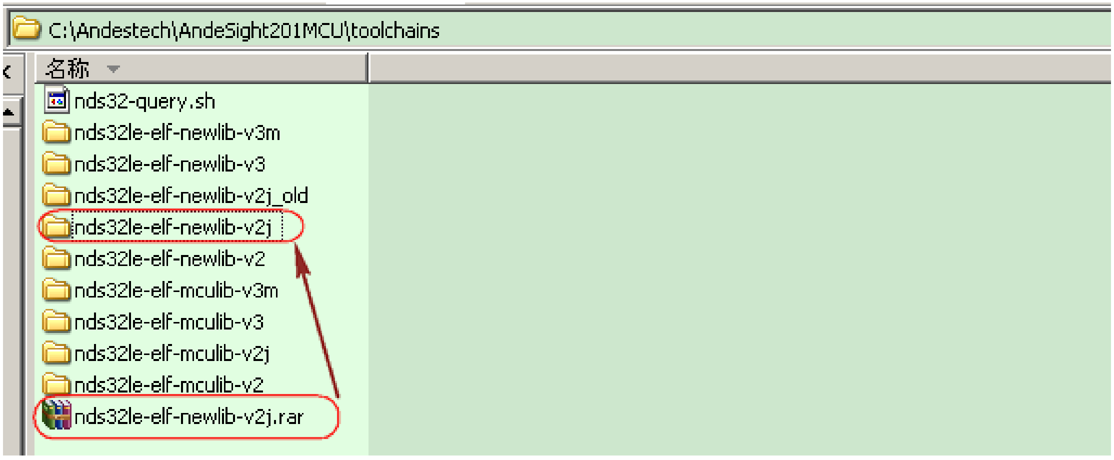
copy Toolchain file into "C:\Andestech\AndeSight201MCU\toolchains"


TK_MT7681\MT7681_IoT_Package_v1.50\ToolChain

名称
MTK_AndesToolChains_Usage_v0.03_20140528.pdf
nds32le-elf-newlib-v2j.rar

included in "ToolChain" folder of SDK

Backup "nds32le-elf-newlib-v2j" folder into other name or delete it ,then uncompress "nds32le-elf-newlib-v2j.rar".



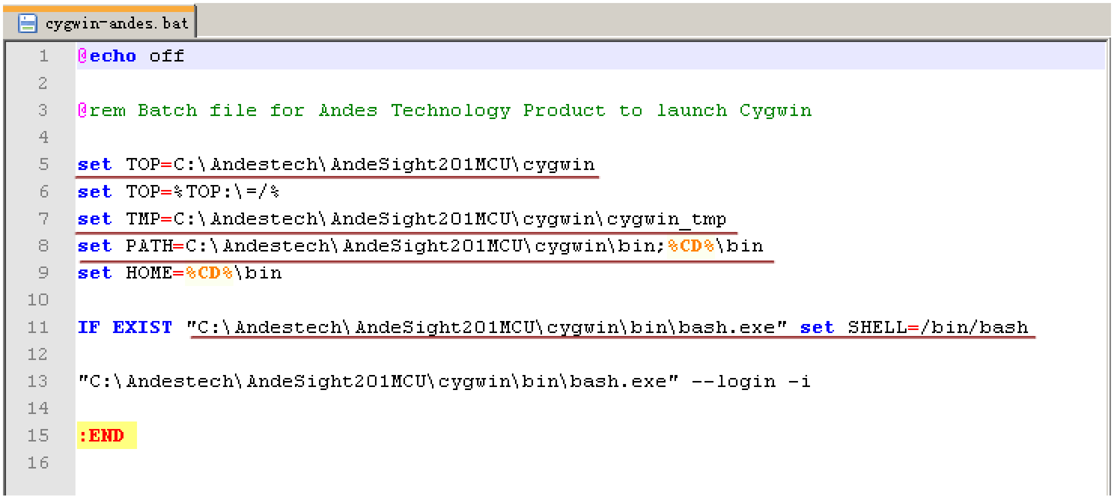
C:\Andestech\AndeSight201MCU\toolchains

名称

nds32-query.sh
nds32le-elf-newlib-v3m
nds32le-elf-newlib-v3
nds32le-elf-newlib-v2j_old
nds32le-elf-newlib-v2j
nds32le-elf-newlib-v2
nds32le-elf-mculib-v3m
nds32le-elf-mculib-v3
nds32le-elf-mculib-v2j
nds32le-elf-mculib-v2
nds32le-elf-newlib-v2j.rar

Step 3:

Change the directory to "C:\Andestech\AndeSight201MCU\toolchains\nds32le-elf-newlib-v2j",and edit "cygwin-andes.bat" like below .

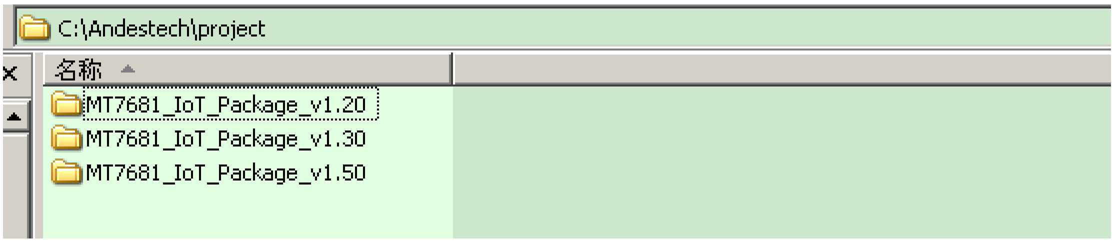


cygwin-andes.bat

```
@echo off

@rem Batch file for Andes Technology Product to launch Cygwin

set TOP=C:\Andestech\AndeSight201MCU\cygwin
set TOP=%TOP:\=/%
set TMP=C:\Andestech\AndeSight201MCU\cygwin\cygwin_tmp
set PATH=C:\Andestech\AndeSight201MCU\cygwin\bin;%CD%\bin
set HOME=%CD%\bin

IF EXIST "C:\Andestech\AndeSight201MCU\cygwin\bin\bash.exe" set SHELL=/bin/bash

"C:\Andestech\AndeSight201MCU\cygwin\bin\bash.exe" --login -i

:END

```

Step 4:

Extract SDK into a directory (such as C:\Andestech\project\)



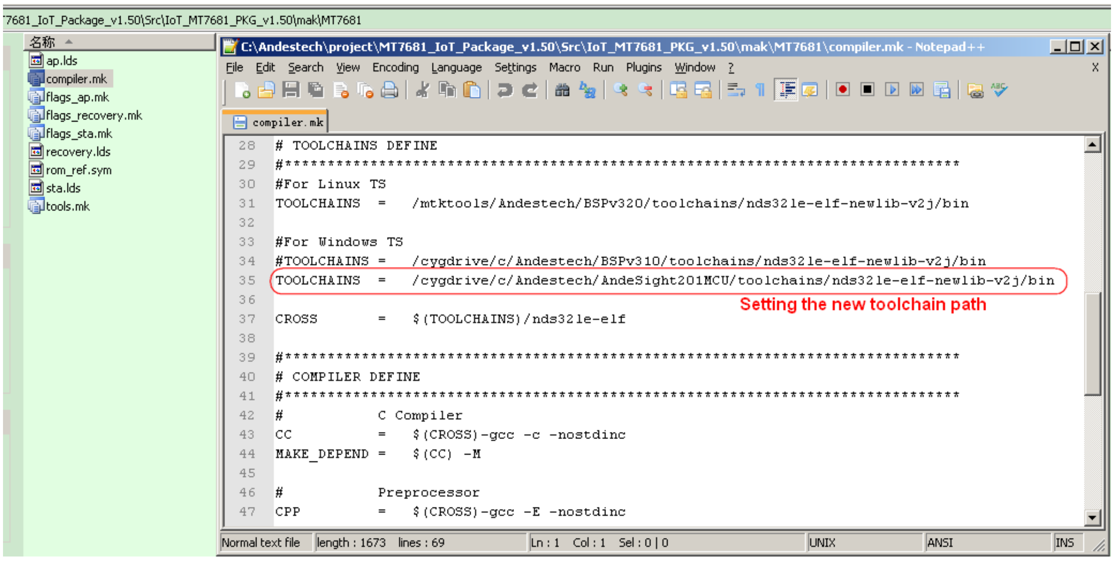
C:\Andestech\project

名称

MT7681_IoT_Package_v1.20
MT7681_IoT_Package_v1.30
MT7681_IoT_Package_v1.50

Step 5:

change the directory to "C:\Andestech\project\MT7681_IoT_Package_v1.50\Src\IoT_MT7681_PKG_v1.50\mak\MT7681",open "compiler.mk" file and Set the new toolchain path like below .


7681_IoT_Package_v1.50\Src\IoT_MT7681_PKG_v1.50\mak\MT7681
名称
ap.lds
compiler.mk
flags_ap.mk
flags_recovery.mk
flags_sta.mk
recovery.lds
rom_ref.sym
sta.lds
tools.mk
C:\Andestech\project\MT7681_IoT_Package_v1.50\Src\IoT_MT7681_PKG_v1.50\mak\MT7681\compiler.mk - Notepad++
File Edit Search View Encoding Language Settings Macro Run Plugins Window ?
compiler.mk

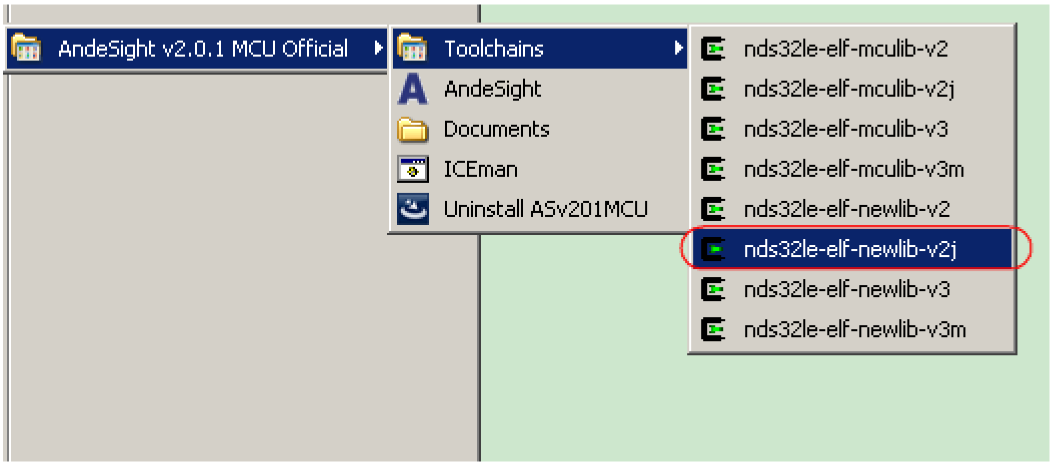
```
# TOOLCHAINS DEFINE
#****************************************************************************
#For Linux TS
TOOLCHAINS  =   /mtktools/Andestech/BSPv320/toolchains/nds32le-elf-newlib-v2j/bin

#For Windows TS
#TOOLCHAINS =   /cygdrive/c/Andestech/BSPv310/toolchains/nds32le-elf-newlib-v2j/bin
TOOLCHAINS  =   /cygdrive/c/Andestech/AndeSight201MCU/toolchains/nds32le-elf-newlib-v2j/bin

CROSS       =   $(TOOLCHAINS)/nds32le-elf

#****************************************************************************
# COMPILER DEFINE
#****************************************************************************
#           C Compiler
CC          =   $(CROSS)-gcc -c -nostdinc
MAKE_DEPEND =   $(CC) -M

#           Preprocessor
CPP         =   $(CROSS)-gcc -E -nostdinc
```

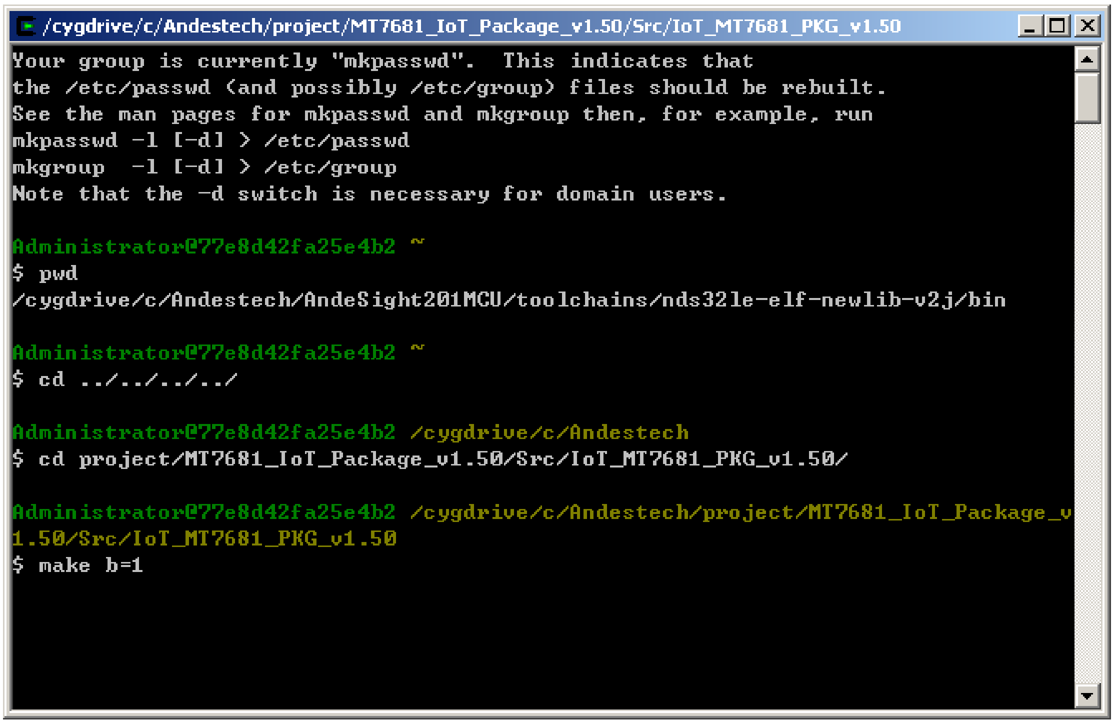
Setting the new toolchain path

Step 6:

Run "nds32le-elf-newlib-v2j" program,

Enter into "\MT7681_IoT_Package_v1.50\Src\IoT_MT7681_PKG_v1.50" directory ,compile source code .

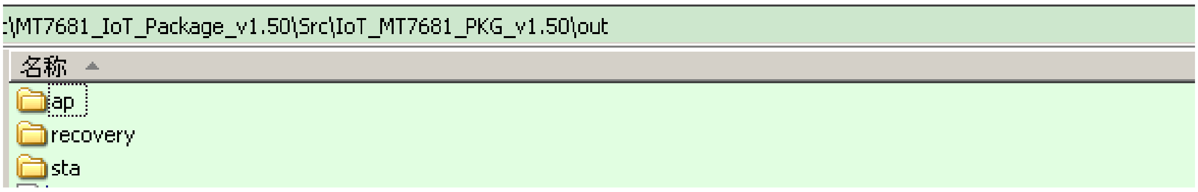




/cygdrive/c/Andestech/project/MT7681_IoT_Package_v1.50/Src/IoT_MT7681_PKG_v1.50

```
Your group is currently "mkpasswd".  This indicates that
the /etc/passwd (and possibly /etc/group) files should be rebuilt.
See the man pages for mkpasswd and mkgroup then, for example, run
mkpasswd -l [-d] > /etc/passwd
mkgroup  -l [-d] > /etc/group
Note that the -d switch is necessary for domain users.

Administrator@77e8d42fa25e4b2 ~
$ pwd
/cygdrive/c/Andestech/AndeSight201MCU/toolchains/nds32le-elf-newlib-v2j/bin

Administrator@77e8d42fa25e4b2 ~
$ cd ../../../../

Administrator@77e8d42fa25e4b2 /cygdrive/c/Andestech
$ cd project/MT7681_IoT_Package_v1.50/Src/IoT_MT7681_PKG_v1.50/

Administrator@77e8d42fa25e4b2 /cygdrive/c/Andestech/project/MT7681_IoT_Package_v
1.50/Src/IoT_MT7681_PKG_v1.50
$ make b=1
```

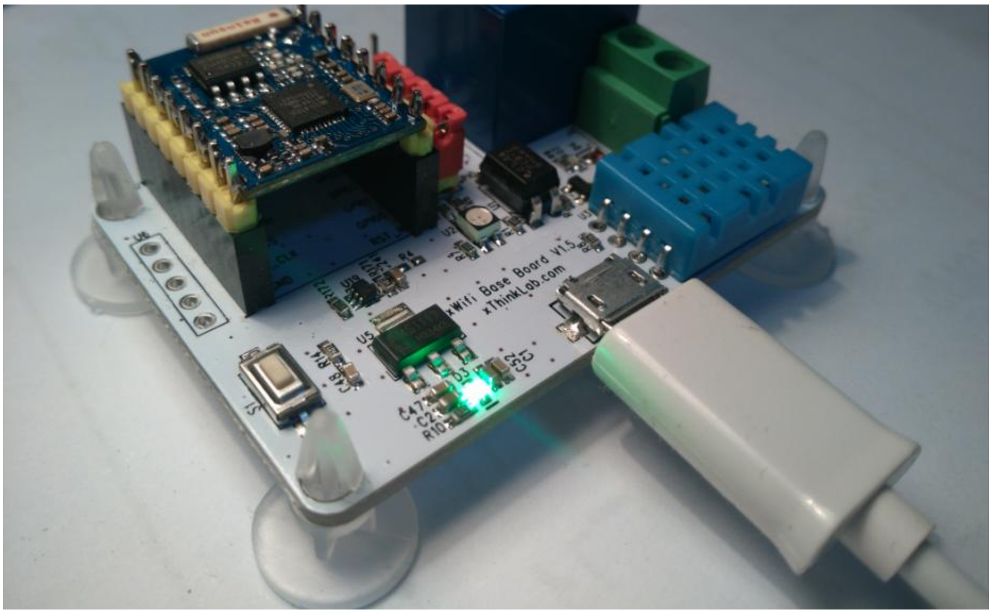
The directory of generated binary files is in



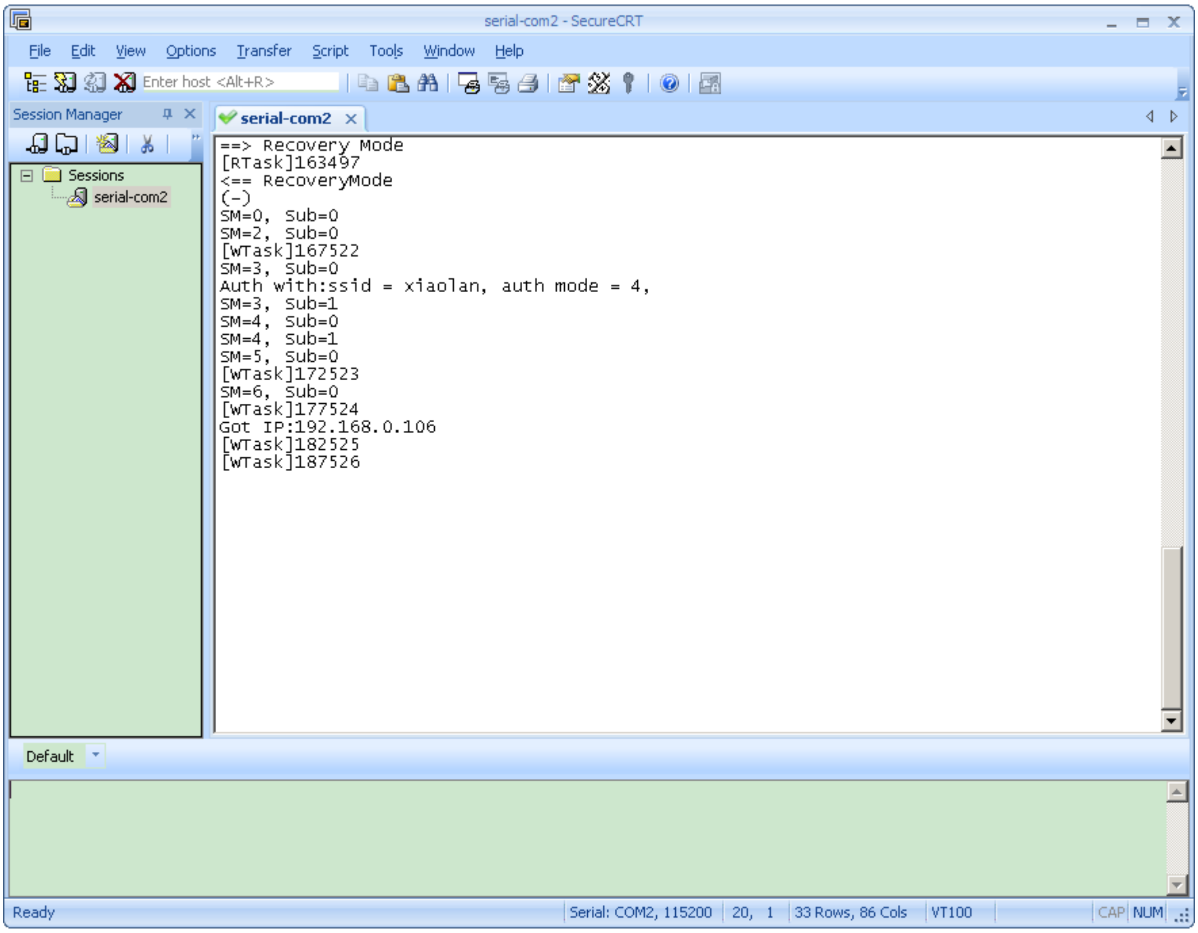
```
:\MT7681_IoT_Package_v1.50\Src\IoT_MT7681_PKG_v1.50\out
名称
ap
recovery
sta
```

2.Upgrade firmware via UART

Step 1: Power on xWifi module .

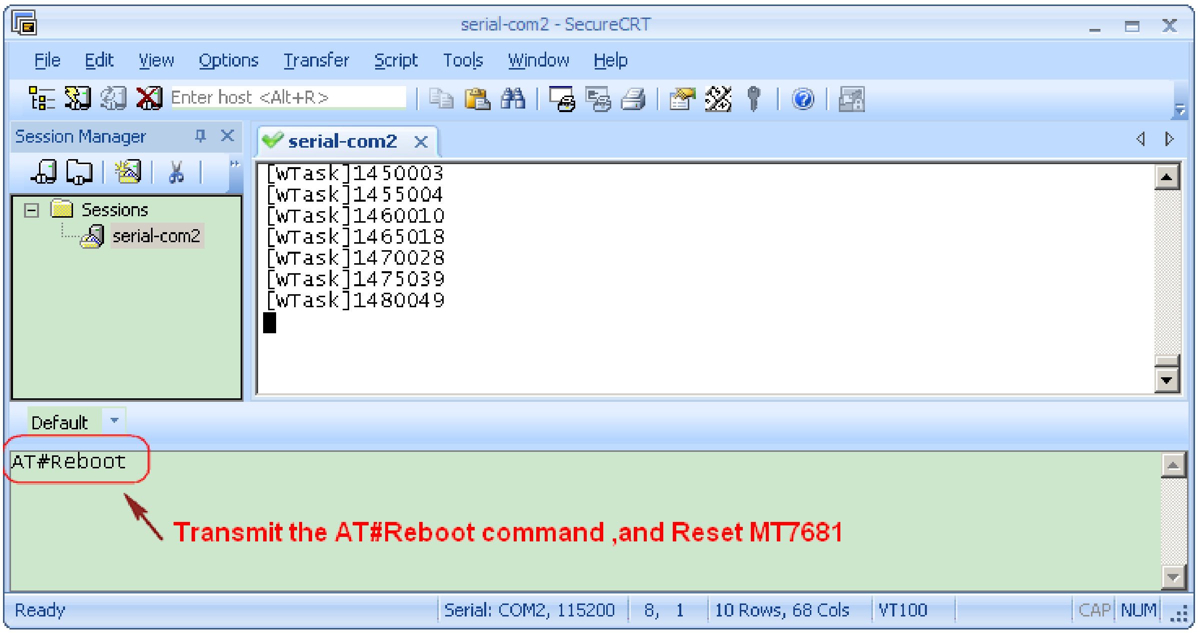


Step 2: Need a USB to UART Bridge to set up a connection between Computer and xWifi boar.



```
==> Recovery Mode
[RTask]163497
<== RecoveryMode
(-)
SM=0, Sub=0
SM=2, Sub=0
[WTask]167522
SM=3, Sub=0
Auth with:ssid = xiaolan, auth mode = 4,
SM=3, Sub=1
SM=4, Sub=0
SM=4, Sub=1
SM=5, Sub=0
[WTask]172523
SM=6, Sub=0
[WTask]177524
Got IP:192.168.0.106
[WTask]182525
[WTask]187526
```

Step 3: Sending “AT#Reboot” command to reset MT7681

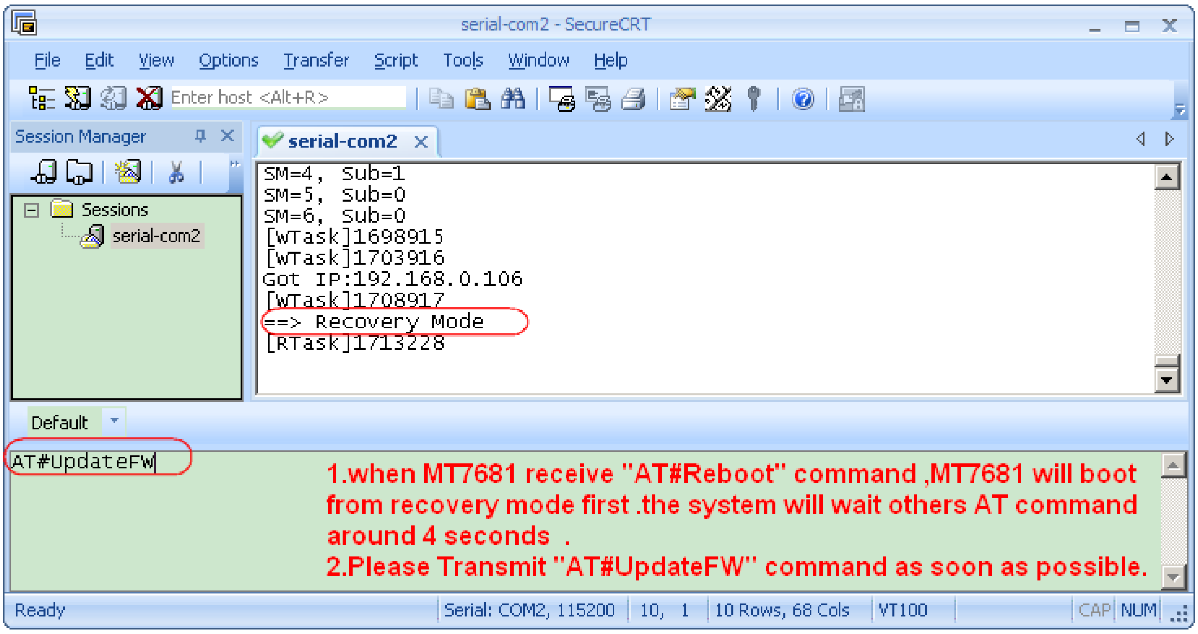


```
[WTask]1450003
[WTask]1455004
[WTask]1460010
[WTask]1465018
[WTask]1470028
[WTask]1475039
[WTask]1480049
```

```
AT#Reboot
```

Transmit the AT#Reboot command ,and Reset MT7681

Step 4: Sending “AT#UpdateFW” command within 4 seconds

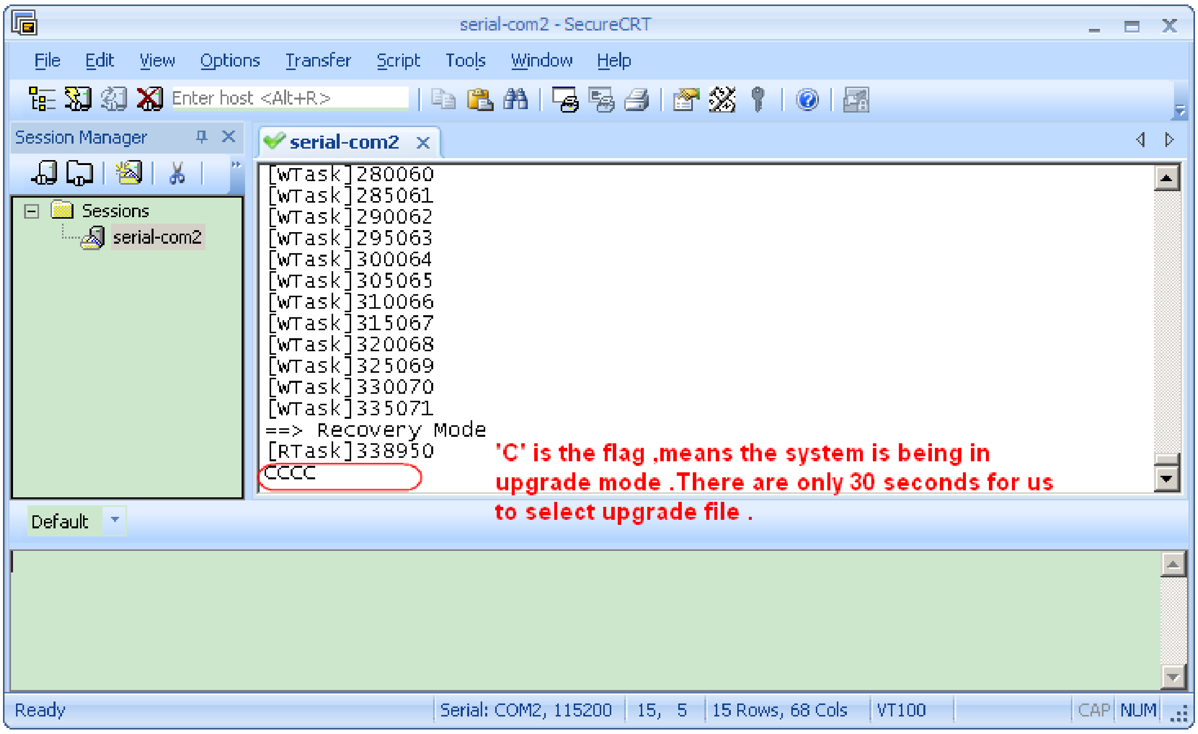


```
SM=4, Sub=1
SM=5, Sub=0
SM=6, Sub=0
[WTask]1698915
[WTask]1703916
Got IP:192.168.0.106
[WTask]1708917
==> Recovery Mode
[RTask]1713228
```

```
AT#UpdateFW
```

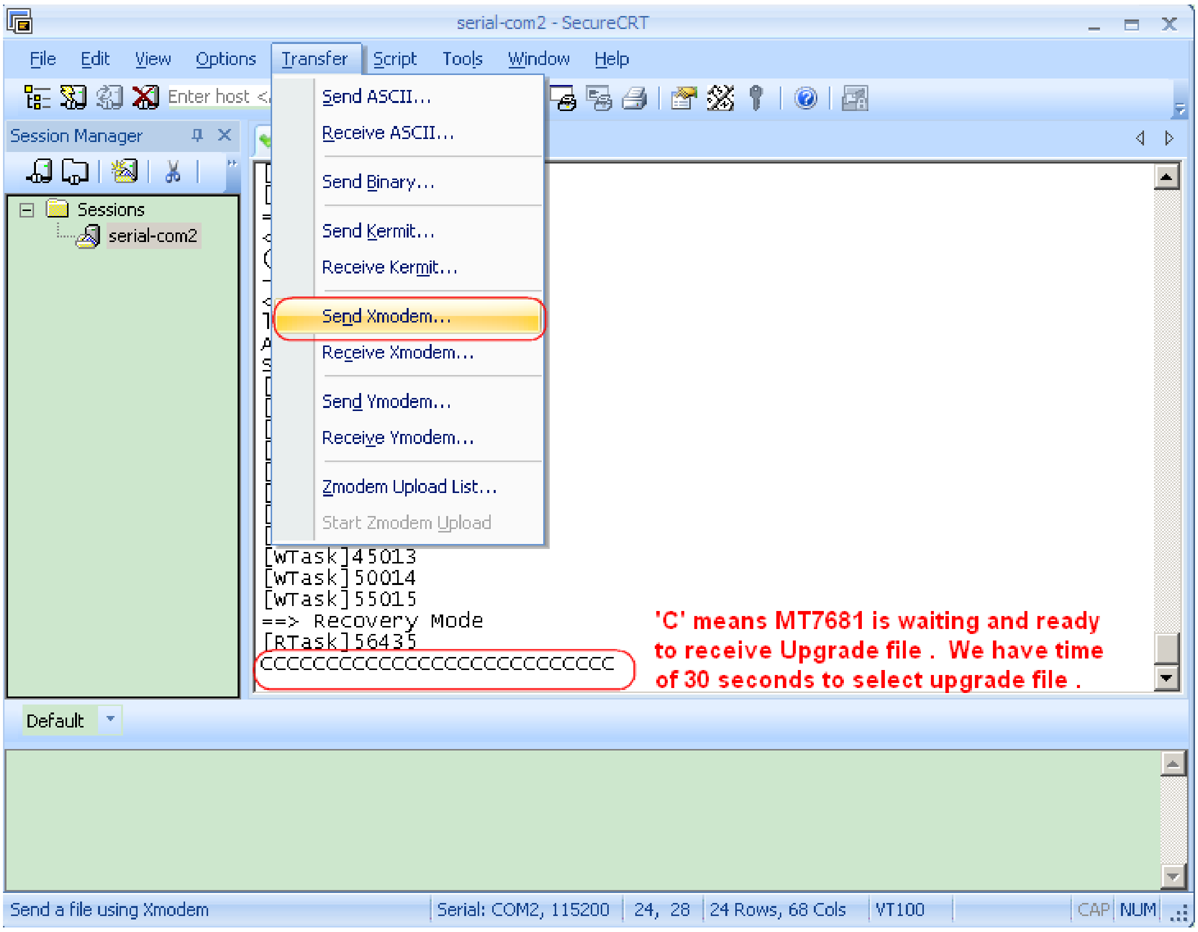
1.when MT7681 receive "AT#Reboot" command ,MT7681 will boot from recovery mode first .the system will wait others AT command around 4 seconds .
2.Please Transmit "AT#UpdateFW" command as soon as possible.

Step 5: Enter into upgrade mode



```
[WTask]280060
[WTask]285061
[WTask]290062
[WTask]295063
[WTask]300064
[WTask]305065
[WTask]310066
[WTask]315067
[WTask]320068
[WTask]325069
[WTask]330070
[WTask]335071
==> Recovery Mode
[RTask]338950
CCCC
```

'C' is the flag ,means the system is being in upgrade mode .There are only 30 seconds for us to select upgrade file .



```
[WTask]45013
[WTask]50014
[WTask]55015
==> Recovery Mode
[RTask]56435
CCCCCCCCCCCCCCCCCCCCCCCCCCCCC
```

'C' means MT7681 is waiting and ready to receive Upgrade file . We have time of 30 seconds to select upgrade file .

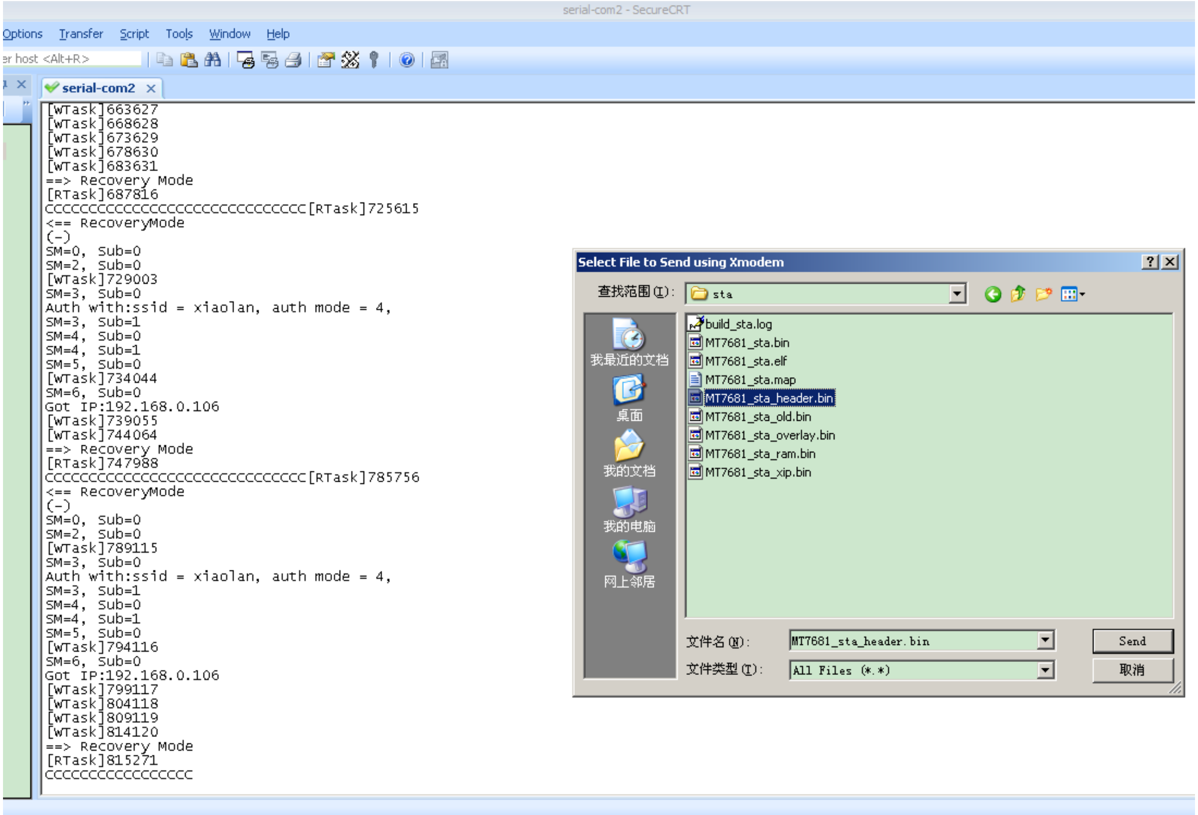
Send a file using Xmodem | Serial: COM2, 115200 | 24, 28 | 24 Rows, 68 Cols | VT100 | CAP | NUM

Step 6: Selecting the upgrade file .

The generated file include:

MT7681_sta_header.bin,MT7681_ap_header.bin,MT7681_recovery_header.bin.

Firmware marked with "****_header" which can be used as upgrading files via UART interface.



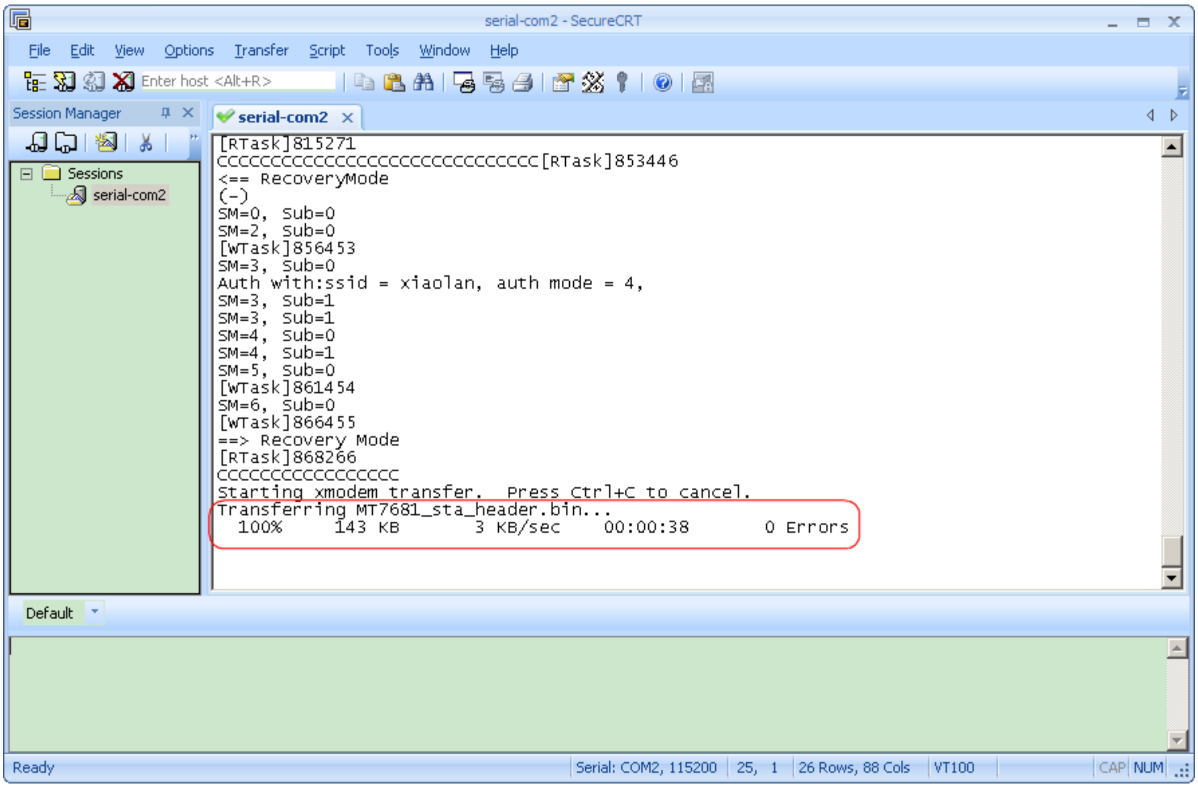
```
[WTask]663627
[WTask]668628
[WTask]673629
[WTask]678630
[WTask]683631
==> Recovery Mode
[RTask]687816
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCC[RTask]725615
<== RecoveryMode
(-)
SM=0, Sub=0
SM=2, Sub=0
[WTask]729003
SM=3, Sub=0
Auth with:ssid = xiaolan, auth mode = 4,
SM=3, Sub=1
SM=4, Sub=0
SM=4, Sub=1
SM=5, Sub=0
[WTask]734044
SM=6, Sub=0
Got IP:192.168.0.106
[WTask]739055
[WTask]744064
==> Recovery Mode
[RTask]747988
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCC[RTask]785756
<== RecoveryMode
(-)
SM=0, Sub=0
SM=2, Sub=0
[WTask]789115
SM=3, Sub=0
Auth with:ssid = xiaolan, auth mode = 4,
SM=3, Sub=1
SM=4, Sub=0
SM=4, Sub=1
SM=5, Sub=0
[WTask]794116
SM=6, Sub=0
Got IP:192.168.0.106
[WTask]799117
[WTask]804118
[WTask]809119
[WTask]814120
==> Recovery Mode
[RTask]815271
CCCCCCCCCCCCCCCCC
```

查找范围(I): sta

build_sta.log
MT7681_sta.bin
MT7681_sta.elf
MT7681_sta.map
MT7681_sta_header.bin
MT7681_sta_old.bin
MT7681_sta_overlay.bin
MT7681_sta_ram.bin
MT7681_sta_xip.bin

文件名(N): MT7681_sta_header.bin

文件类型(T): All Files (*.*)



```
[RTask]815271
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCC[RTask]853446
<== RecoveryMode
(-)
SM=0, Sub=0
SM=2, Sub=0
[WTask]856453
SM=3, Sub=0
Auth with:ssid = xiaolan, auth mode = 4,
SM=3, Sub=1
SM=3, Sub=1
SM=4, Sub=0
SM=4, Sub=1
SM=5, Sub=0
[WTask]861454
SM=6, Sub=0
[WTask]866455
==> Recovery Mode
[RTask]868266
CCCCCCCCCCCCCCCCC
Starting xmodem transfer.  Press Ctrl+C to cancel.
Transferring MT7681_sta_header.bin...
  100%      143 KB       3 KB/sec     00:00:38       0 Errors
```

3.Switch working mode

3.1 Read BootIndex value of 0x18001 via AT#FLASH command:

AT#FLASH -r98305

[0x18001]=[0x00] //Boot as STA mode,

if [0x18001]=[0x01] //Boot as AP mode

AT#FLASH -s98305 -v1 //Switching working mode into AP mode .

AT#FLASH -s98305 -v0 //Switching working mode into STA mode .

3.2 Example:

Switching to AP Mode is simple,just modify the value of Flash Offset: 0x18001 to 1.

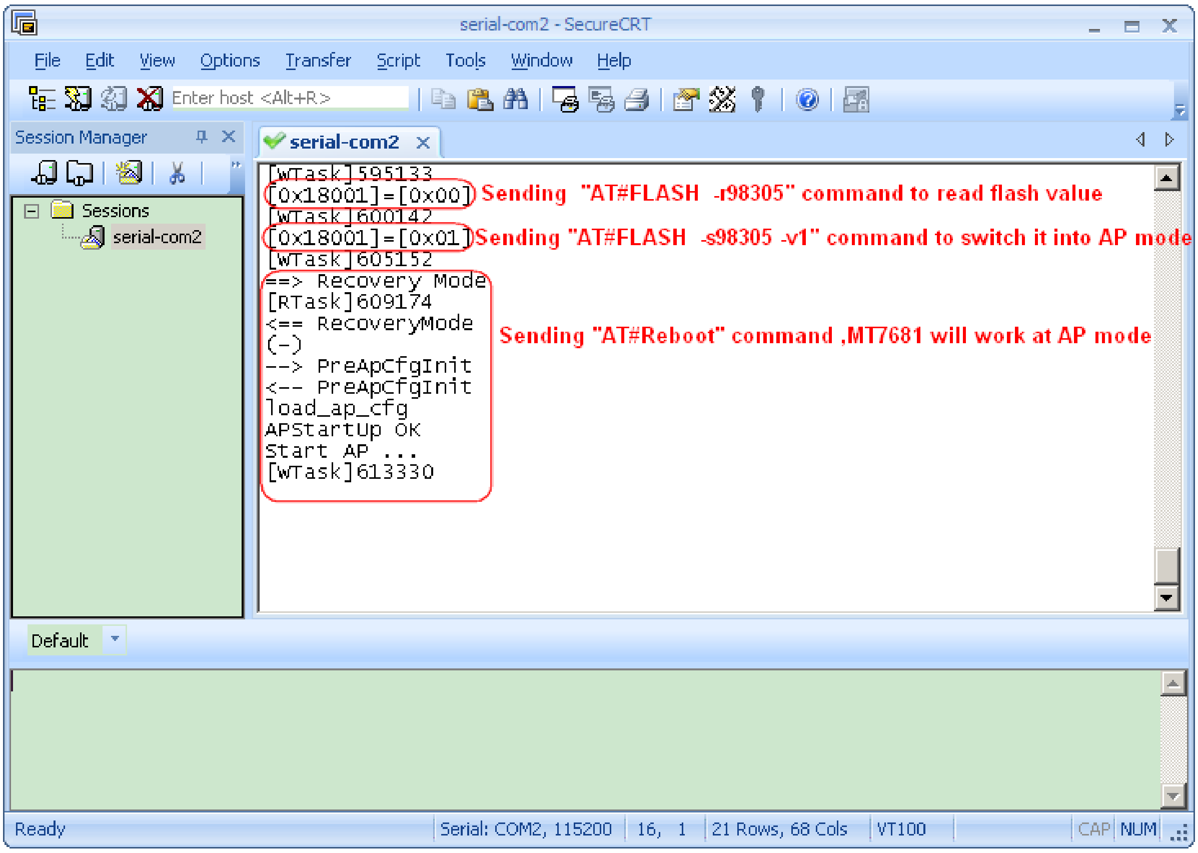
Step1:

Modify BootIndex value of 0x18001 via AT#FLASH command to 1:

AT#FLASH -s98305 -v1

Step2:

Power on MT7681 again, it will boot in AP mode，



```
[wTask]595133
[0x18001]=[0x00]
[wTask]600142
[0x18001]=[0x01]
[wTask]605152
==> Recovery Mode
[RTask]609174
<== RecoveryMode
(-)
--> PreApCfgInit
<-- PreApCfgInit
load_ap_cfg
APStartUp OK
Start AP ...
[wTask]613330
```

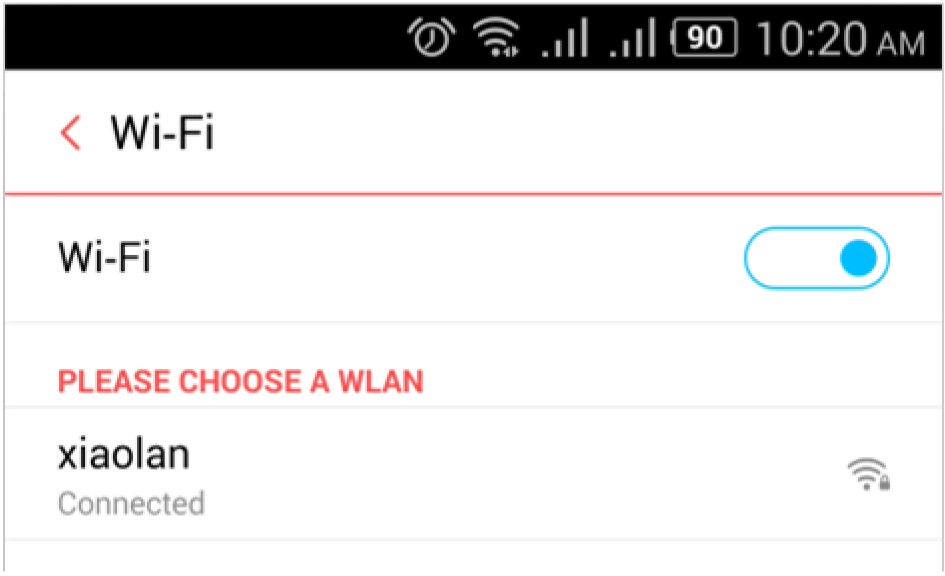
Sending "AT#FLASH -r98305" command to read flash value

Sending "AT#FLASH -s98305 -v1" command to switch it into AP mode

Sending "AT#Reboot" command ,MT7681 will work at AP mode

4.Application demo (Android)

4.1 If MT7681 is being in STA mode ,make sure your smart phone connect your wifi router (example : 'xiaolan' is my router's SSID).



90 10:20 AM

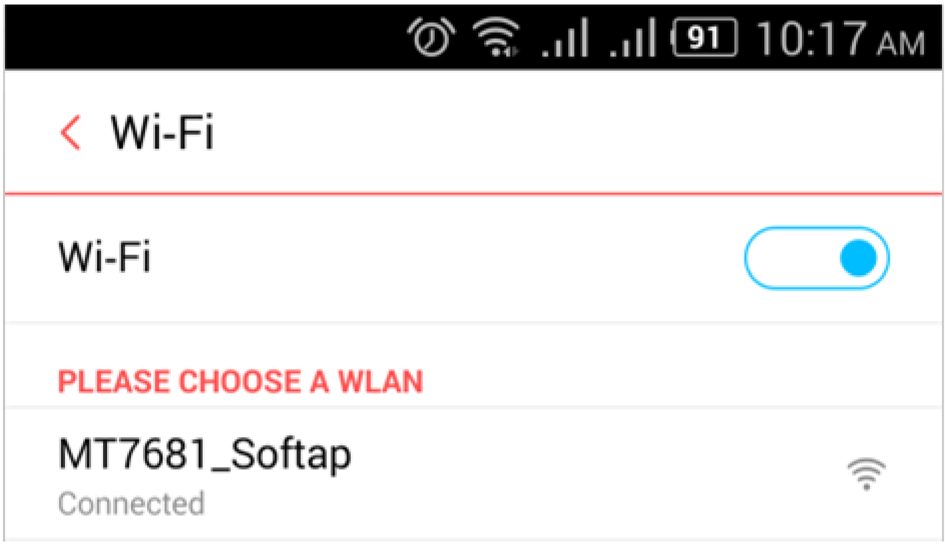
Wi-Fi

Wi-Fi

PLEASE CHOOSE A WLAN

xiaolan
Connected

If MT7681 is being in AP mode ,make sure your smart phone connect “MT7681_Softap”.



91 10:17 AM

Wi-Fi

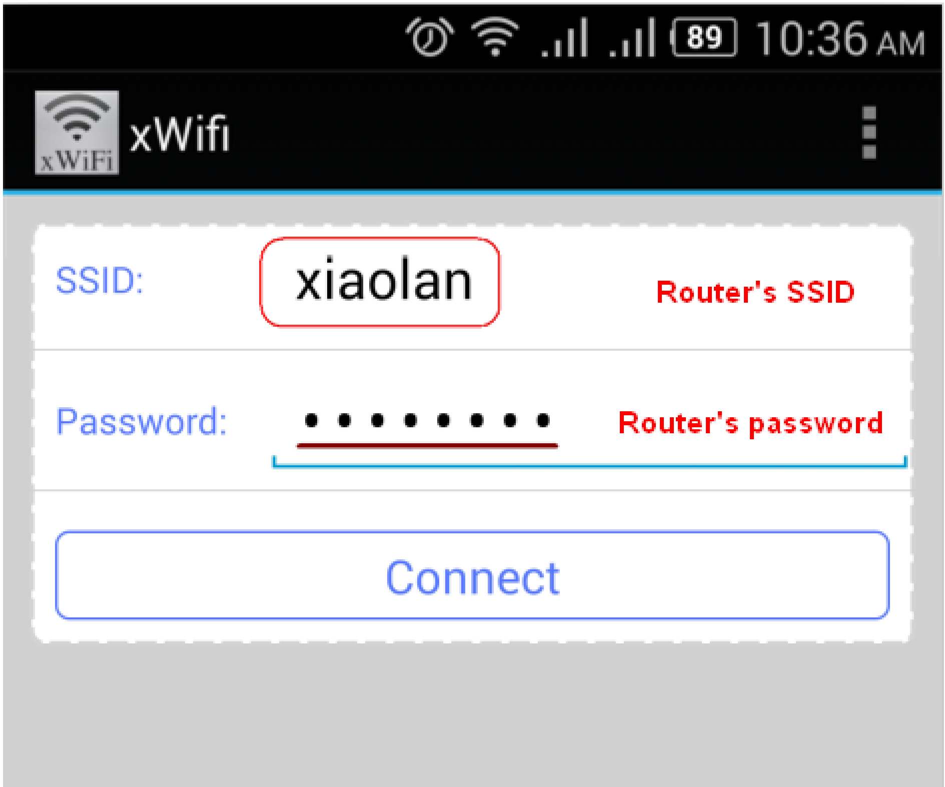
Wi-Fi

PLEASE CHOOSE A WLAN

MT7681_Softap
Connected

4.2 Install ‘xWifi _v1.0.apk’ (you can find it in APK directory of SDK)and open the app ,the screen below will show you first .

If MT7681 is being in STA mode:



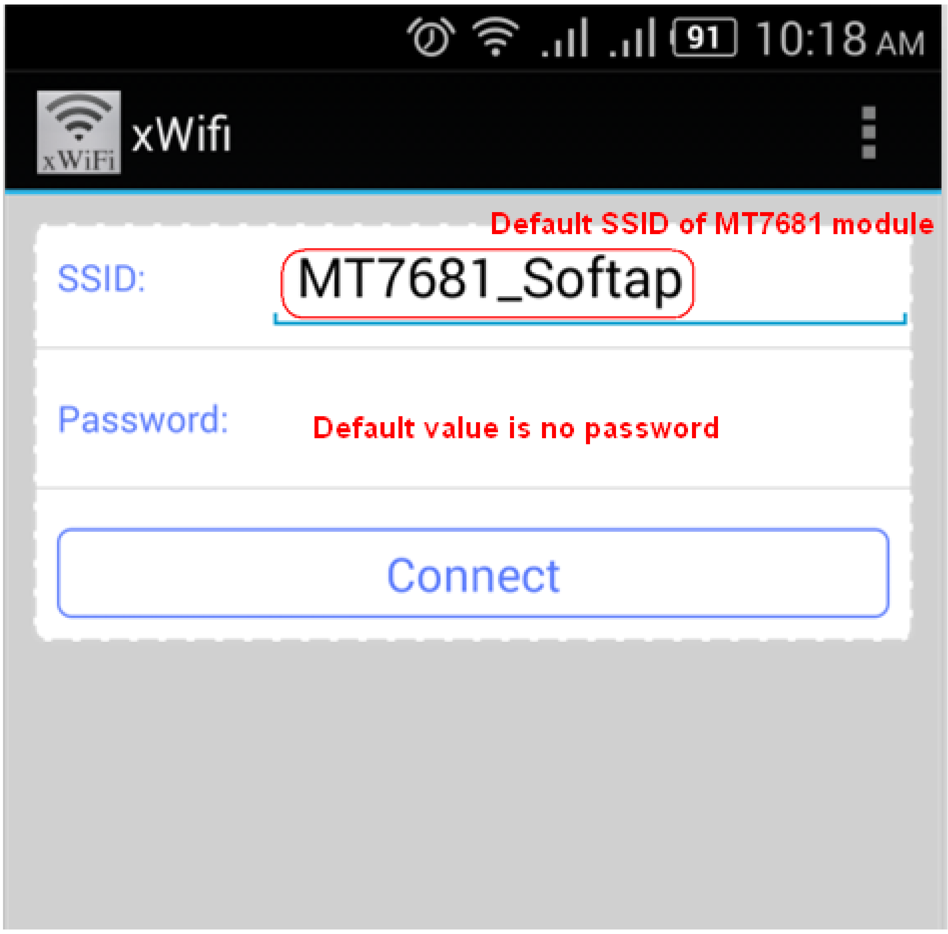
SSID: xiaolan

Router's SSID

Password: ••••••••

Router's password

If MT7681 is being in AP mode:



Default SSID of MT7681 module

SSID: MT7681_Softap

Password:

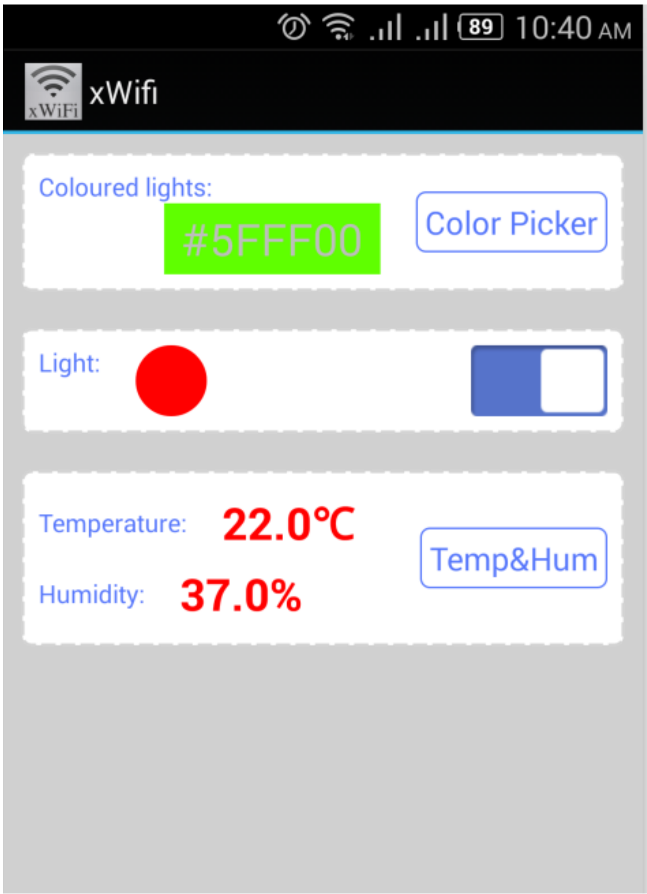
Default value is no password

4.3 we have completed the follow functions:

a.Click ‘Color Picker’ ,the RGB-LED on base board will show the color you picked

b.Click the Light slide button ,The relay will be close or open with blue indicator on or off on base board .

c.Click ‘Temp&Hum’ ,trigger a measurement of Temperature and Humidity ,the result will show you immediately .

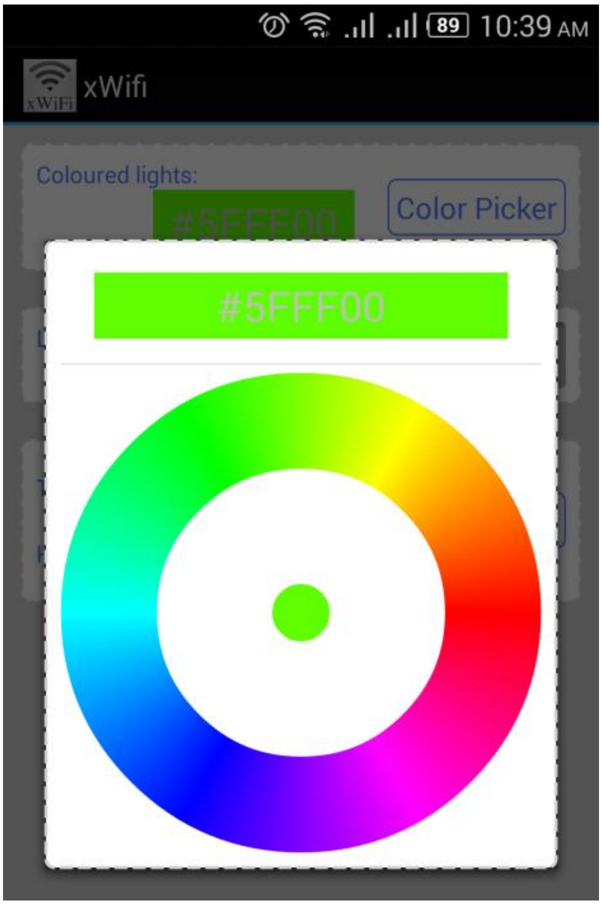


Coloured lights: #5FFF00

Light: ●

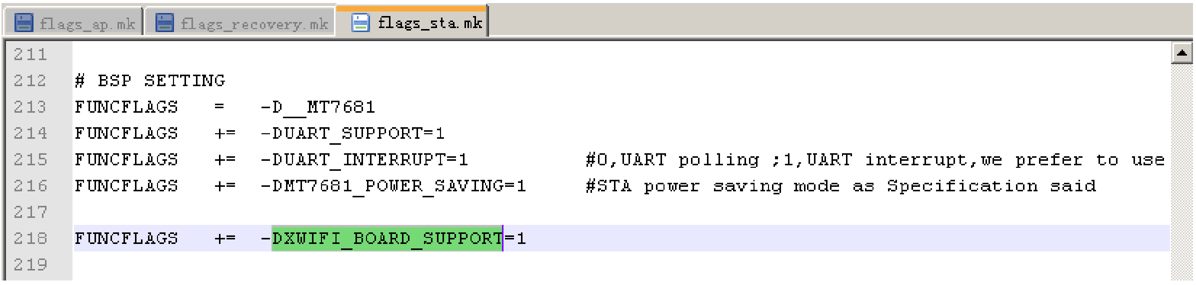
Temperature: 22.0℃

Humidity: 37.0%



5. A macros switch

We added a macros for xWifi based on original code from Mediatek in Makefile (flags_ap.mk, flags_recovery.mk, flags_sta.mk)



```

# BSP SETTING
FUNCFLAGS    =    -D__MT7681
FUNCFLAGS    +=   -DUART_SUPPORT=1
FUNCFLAGS    +=   -DUART_INTERRUPT=1          #0,UART polling ;1,UART interrupt,we prefer to use
FUNCFLAGS    +=   -DMT7681_POWER_SAVING=1     #STA power saving mode as Specification said

FUNCFLAGS    +=   -DXWIFI_BOARD_SUPPORT=1

```

If you want to disable xWifi function,pls set

FUNCFLAGS += -DXWIFI_BOARD_SUPPORT=**0**

in the above three Makefile