IoT for Fun!

Chen-Hanson Ting

SVFIG

2017 Maker Faire

2017 Maker Faire Workshop

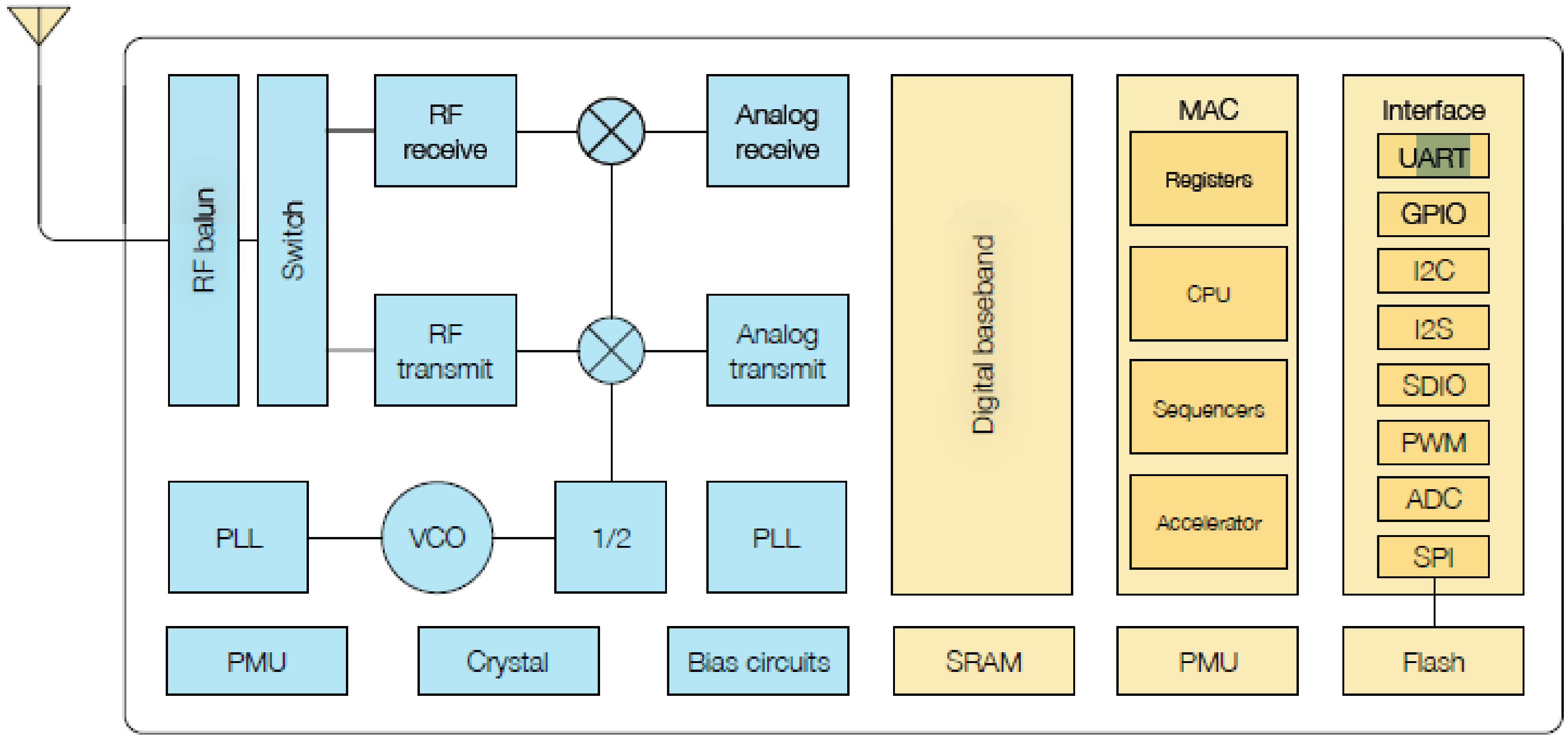
ESP8266

- 32-bit Xtensa LX106 at 160 MHz.
- 64 KB program RAM, 96 KB of data RAM, 4 MB flash.
- IEEE 802.11 b/g/n Wi-Fi.
- GPIO, SPI, I²C, UART, ADC.
- About $1 in bulk.

ESP 12E

Ai
WiFi
MODEL ESP8266MOD
VENDOR AI-THINKER
FC
ISM 2.4GHz
PA +25dBm
802.11b/g/n
M
MOORE
DUINO

ESP8266


RF balun
Switch
RF receive
RF transmit
Analog receive
Analog transmit
PLL
VCO
1/2
PLL
PMU
Crystal
Bias circuits
Digital baseband
SRAM
MAC
Registers
CPU
Sequencers
Accelerator
PMU
Interface
UART
GPIO
I2C
I2S
SDIO
PWM
ADC
SPI
Flash

NodeMCU Board

NodeMCU Board

- It looks that NodeMCU at $3 will take over Arduino Uno, with its WIFI capability, 32-bit processor, and large memories.
- In this workshop, we will explore ways to make use of the wonderful kit for IoT applications.

The Challenge

- Turn LED on NodeMCU board on and off, REMOTELY.
- You can use any tool and language.
- Supplied tools are MicroPython, Lua, and Arduino.
- If succeed, you get a NodeMCU!

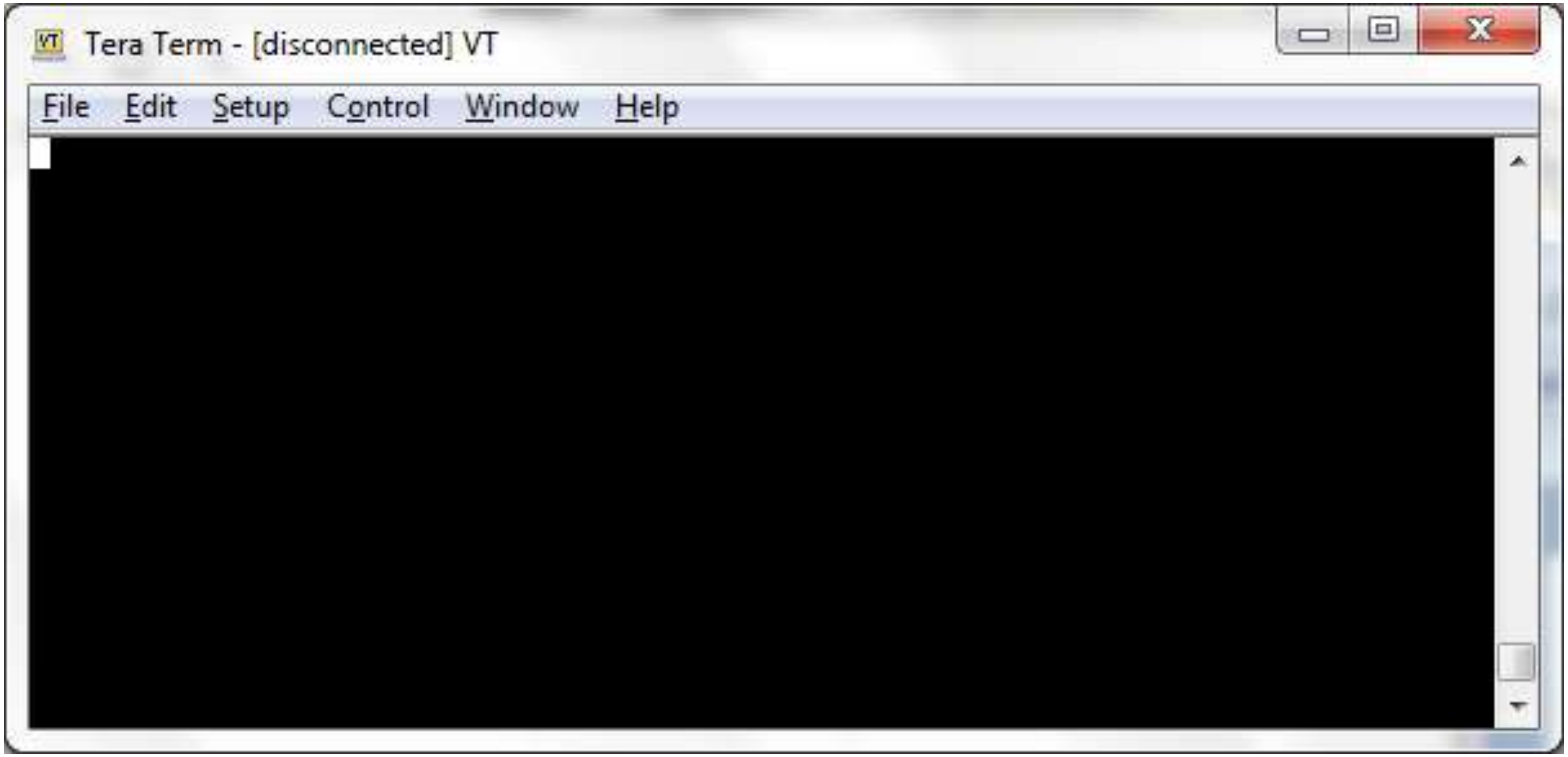
Suggested Steps

- Flash language/tool to flash memory in ESP8266.
- Control LED through USB-Serial Monitor.
- Control LED through WiFi.

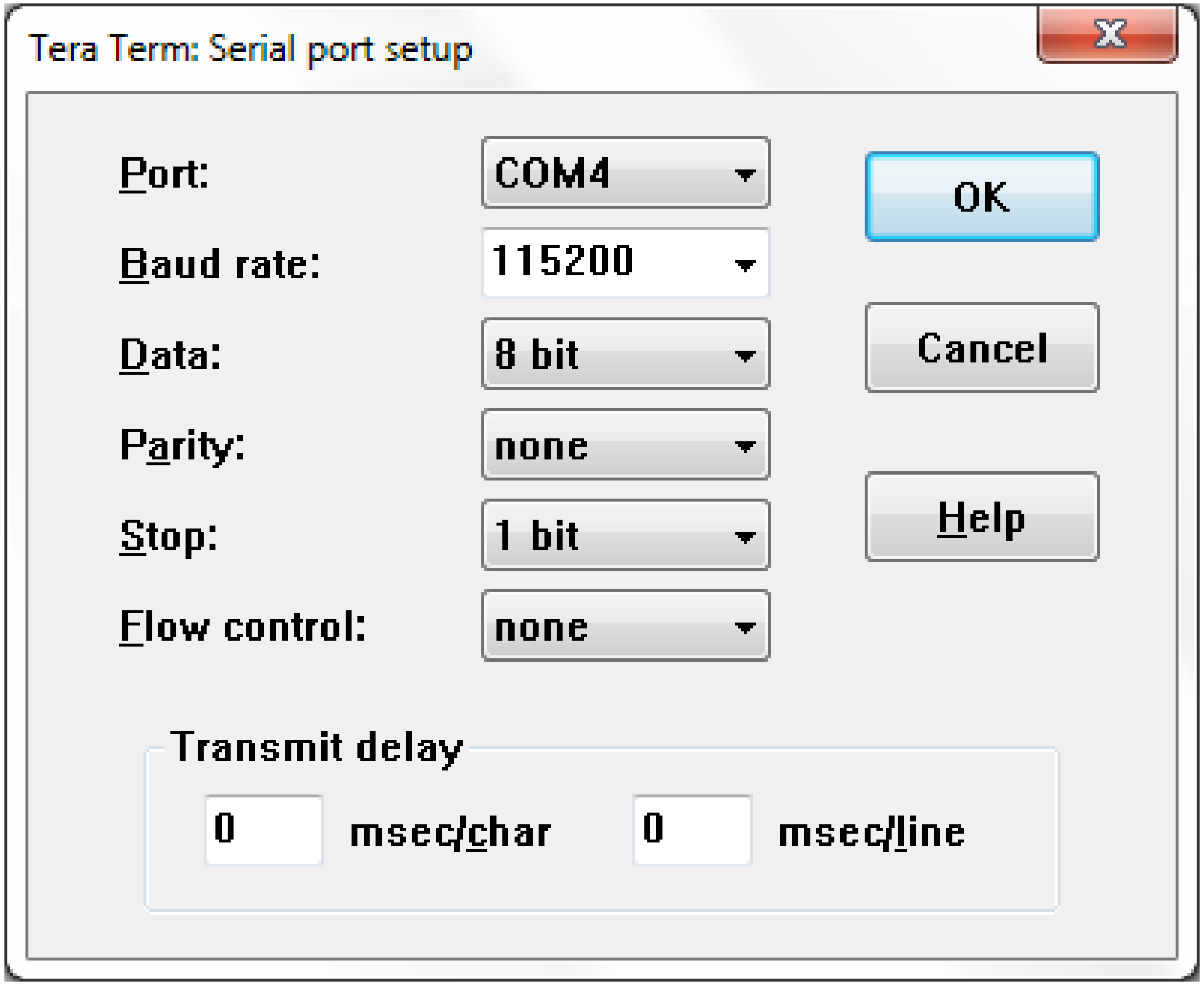
Tools You May Need

1. TeraTerm Terminal Emulator
2. ESP8266Flasher
3. Arduino IDE
4. Hercules UDP Sender

1. TeraTerm Terminal


Tera Term - [disconnected] VT
File Edit Setup Control Window Help

1. TeraTerm Terminal

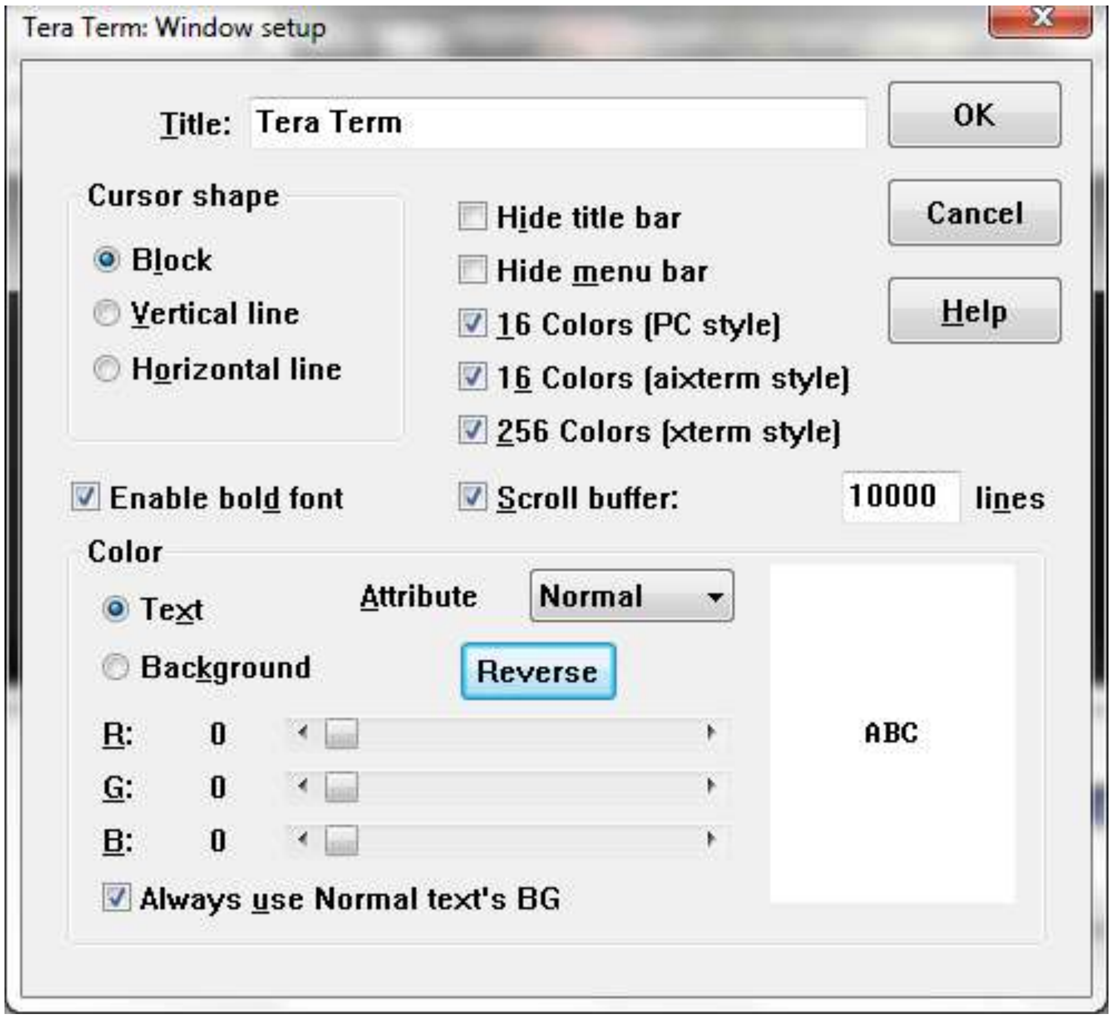

Tera Term: Serial port setup

Port:	COM4
Baud rate:	115200
Data:	8 bit
Parity:	none
Stop:	1 bit
Flow control:	none

Transmit delay

0 msec/char 0 msec/line

1. TeraTerm Terminal



Tera Term: Window setup

Title: Tera Term

Cursor shape
- [✓ Block]
- Vertical line
- Horizontal line

- [ ] Hide title bar
- [ ] Hide menu bar
- [x] 16 Colors (PC style)
- [x] 16 Colors (aixterm style)
- [x] 256 Colors (xterm style)

- [x] Enable bold font
- [x] Scroll buffer: 10000 lines

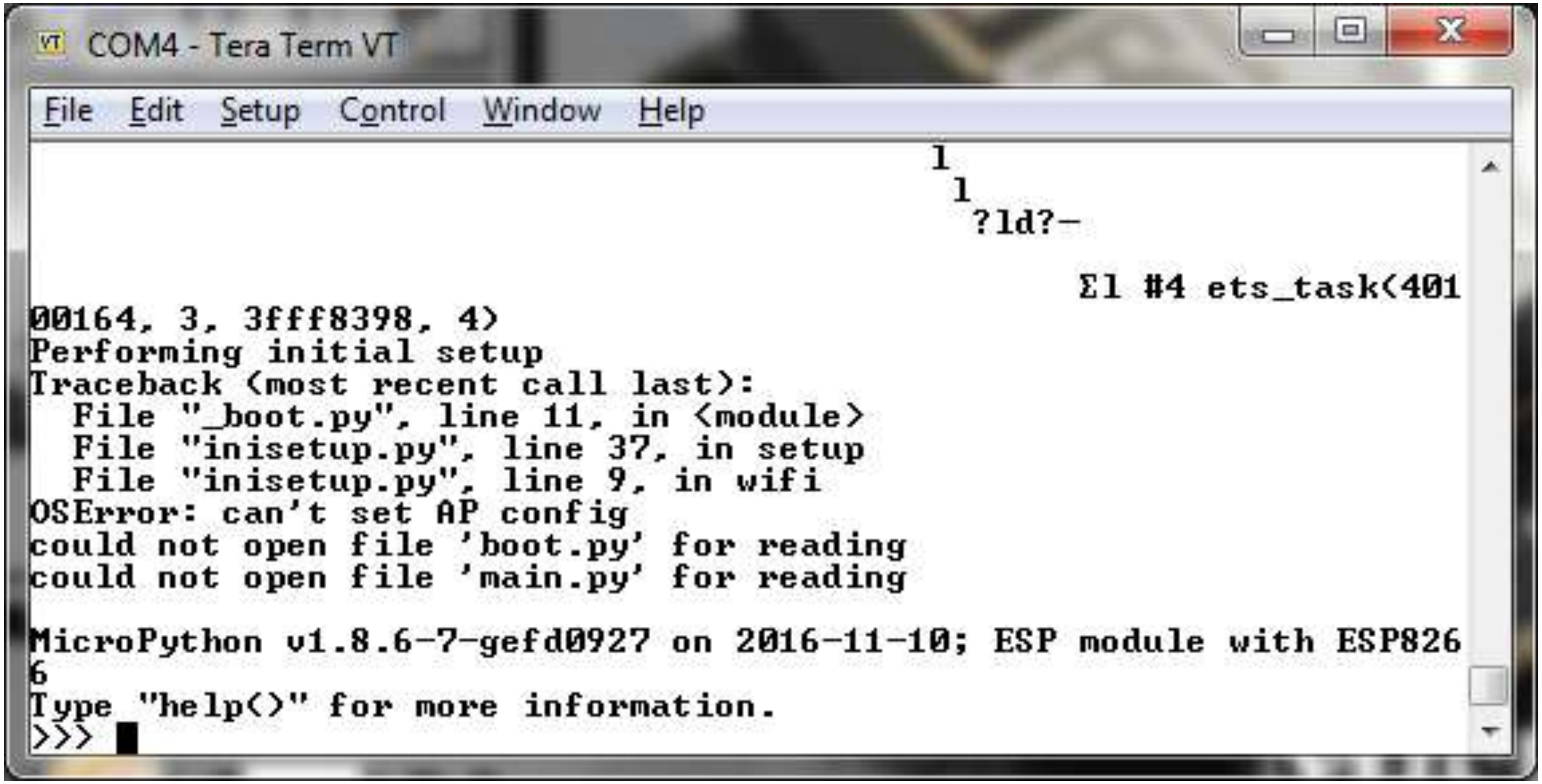
Color
- [✓ Text]
- Background

Attribute: Normal

R: 0
G: 0
B: 0

- [x] Always use Normal text's BG

1. TeraTerm Terminal

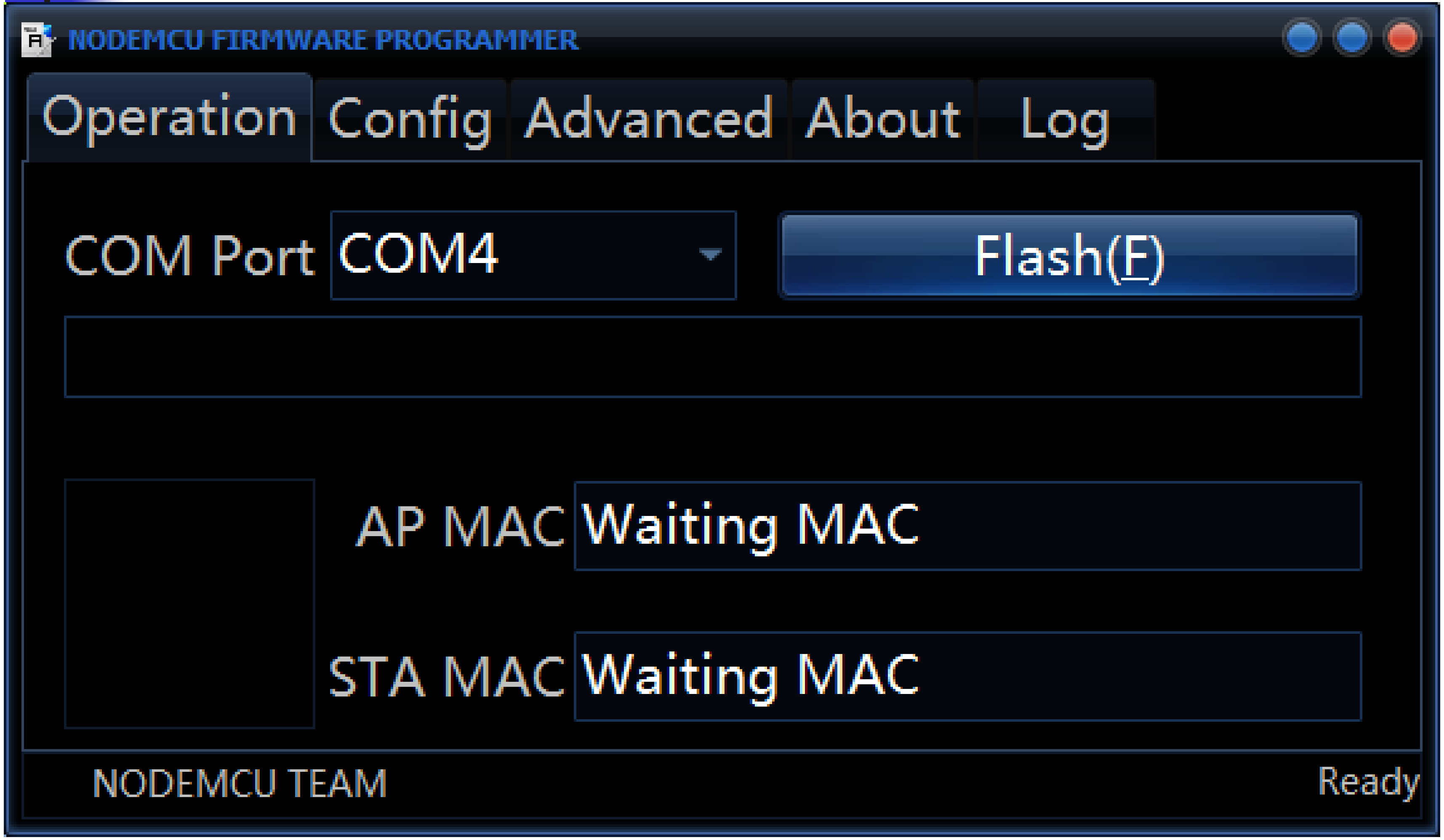


```
                                                   l
                                                    l
                                                     ?ld?-

                                                          El #4 ets_task(401
00164, 3, 3fff8398, 4)
Performing initial setup
Traceback (most recent call last):
  File "_boot.py", line 11, in <module>
  File "inisetup.py", line 37, in setup
  File "inisetup.py", line 9, in wifi
OSError: can't set AP config
could not open file 'boot.py' for reading
could not open file 'main.py' for reading

MicroPython v1.8.6-7-gefd0927 on 2016-11-10; ESP module with ESP826
6
Type "help()" for more information.
>>> █
```

2. ESP8266Flasher

- Not required if you use Arduino IDE.
- Select MicroPython binary image or Lua binary image.
- Write to flash memory of ESP8266.

3. ESP8266Flasher



COM Port: COM4

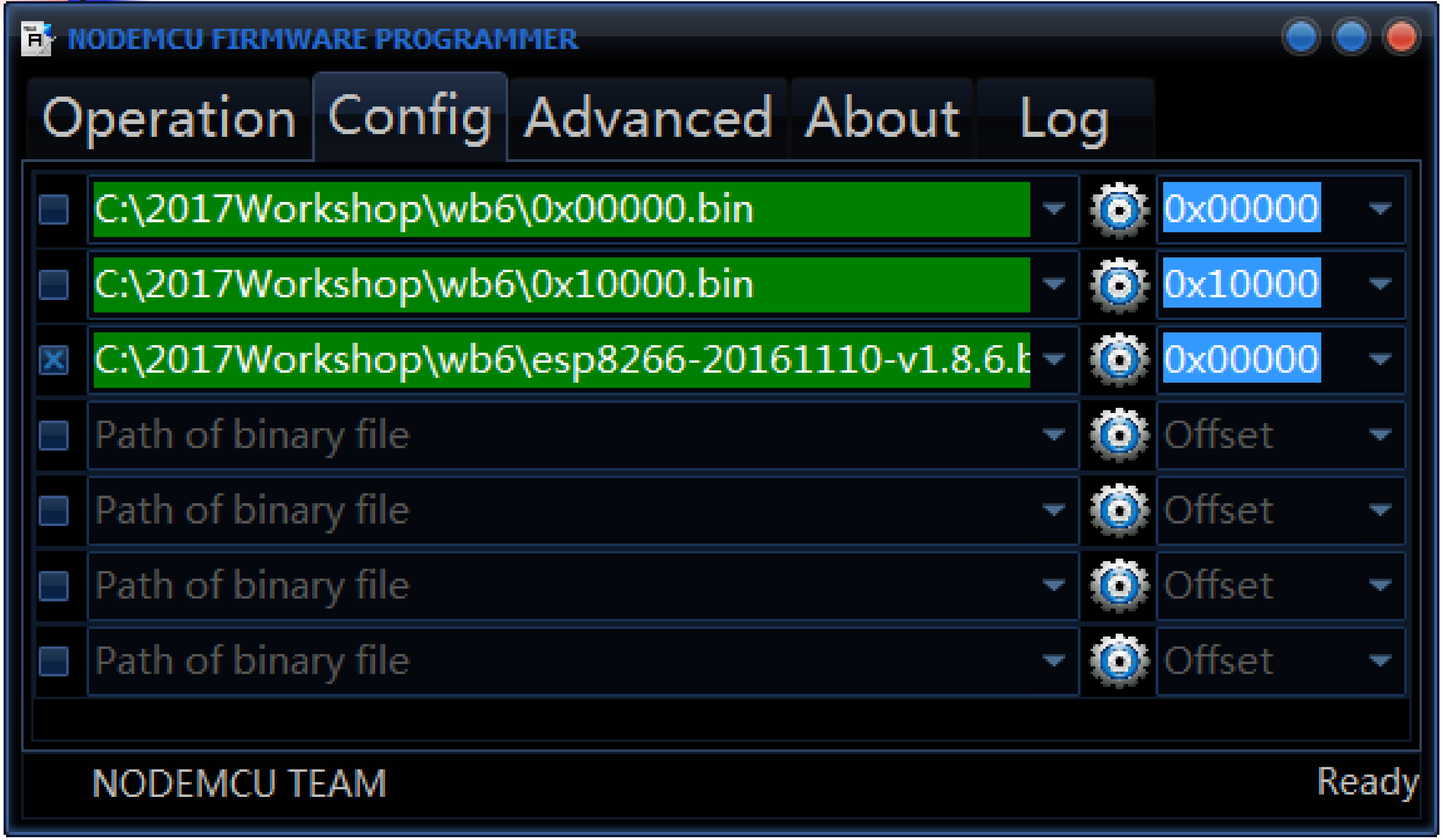
AP MAC: Waiting MAC

STA MAC: Waiting MAC

NODEMCU TEAM

Ready

3. ESP8266Flasher

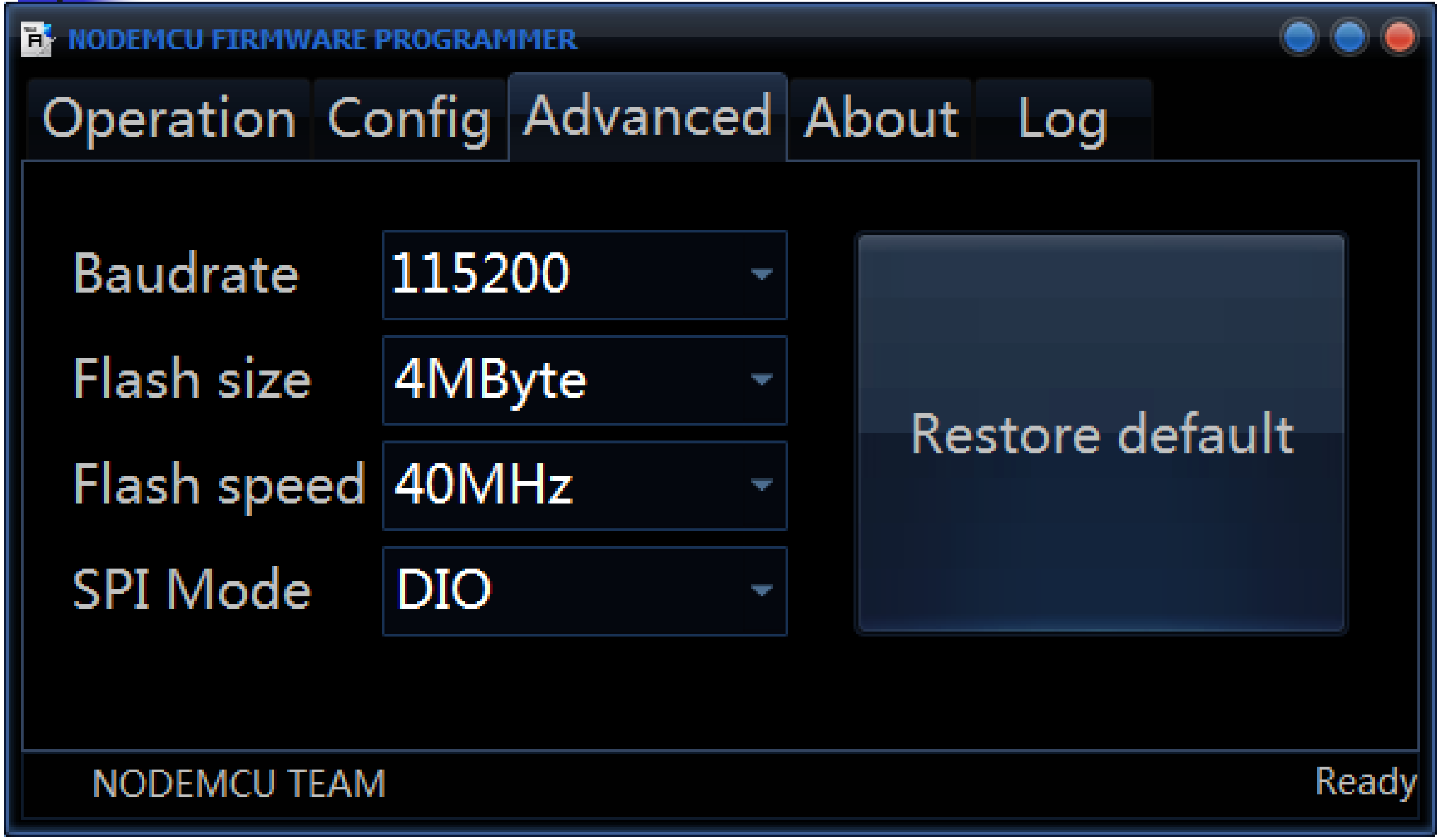

NODEMCU FIRMWARE PROGRAMMER

☐	C:\2017Workshop\wb6\0x00000.bin	0x00000
☐	C:\2017Workshop\wb6\0x10000.bin	0x10000
☒	C:\2017Workshop\wb6\esp8266-20161110-v1.8.6.b	0x00000
☐	Path of binary file	Offset
☐	Path of binary file	Offset
☐	Path of binary file	Offset
☐	Path of binary file	Offset

NODEMCU TEAM

Ready

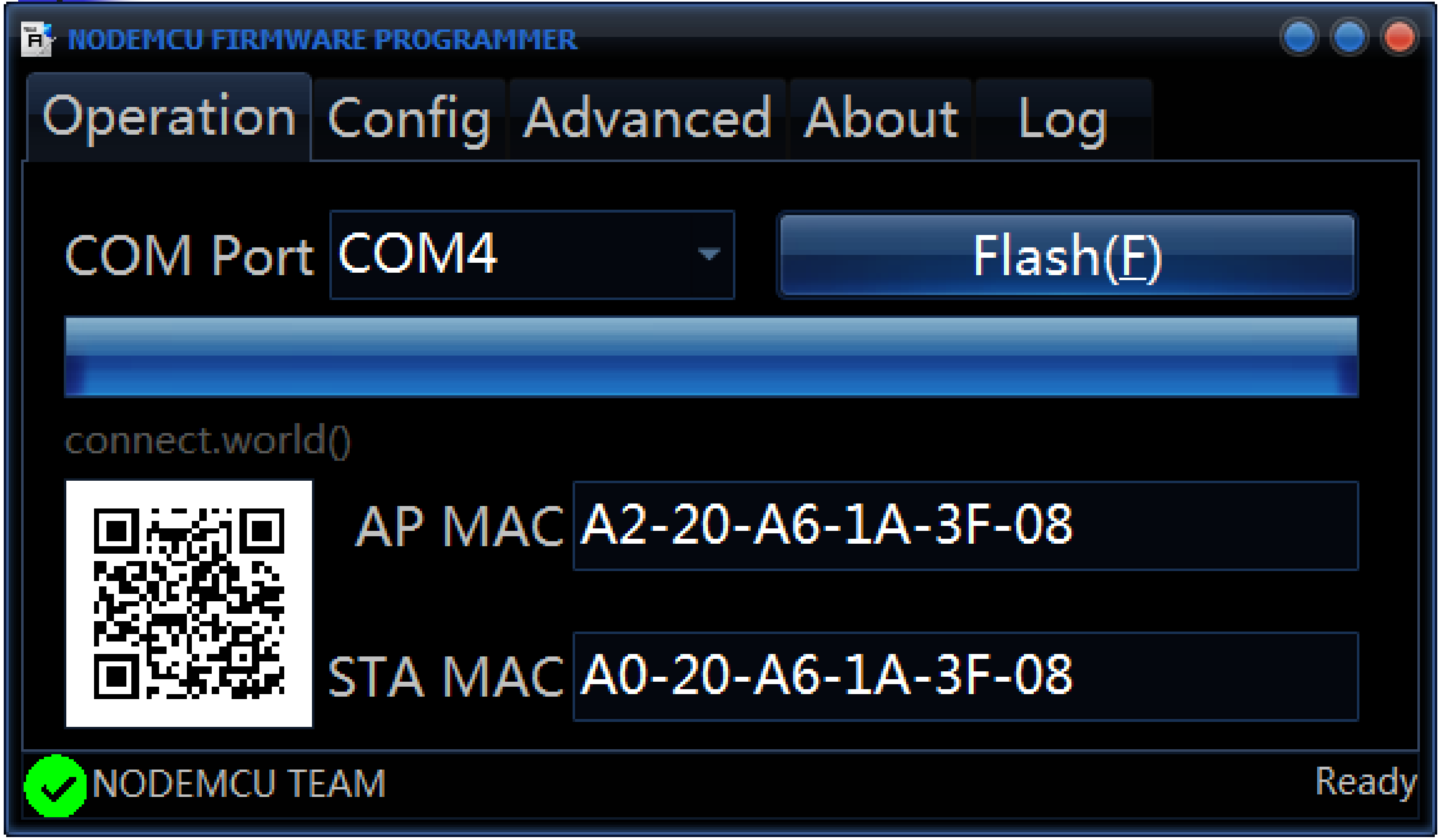
3. ESP8266Flasher



Baudrate 115200
Flash size 4MByte
Flash speed 40MHz
SPI Mode DIO

NODEMCU TEAM Ready

3. ESP8266Flasher



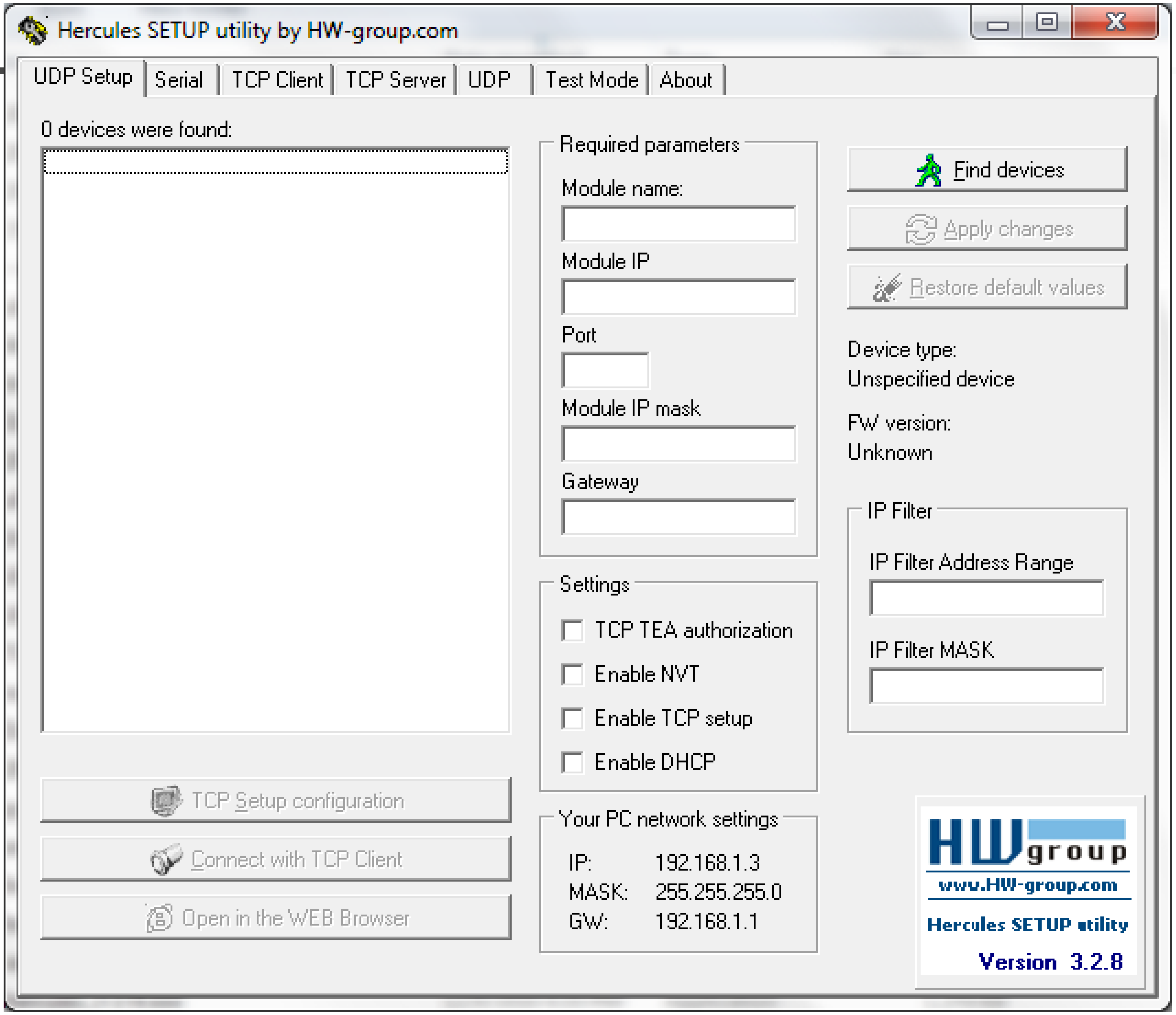
COM Port COM4

connect.world()

AP MAC A2-20-A6-1A-3F-08

STA MAC A0-20-A6-1A-3F-08

4. Hercules Setup Utiltiy



Hercules SETUP utility by HW-group.com

0 devices were found:

Required parameters

Module name:

Module IP

Port

Module IP mask

Gateway

Settings

- [ ] TCP TEA authorization
- [ ] Enable NVT
- [ ] Enable TCP setup
- [ ] Enable DHCP

Your PC network settings

```
IP:     192.168.1.3
MASK:   255.255.255.0
GW:     192.168.1.1
```

Device type:
Unspecified device

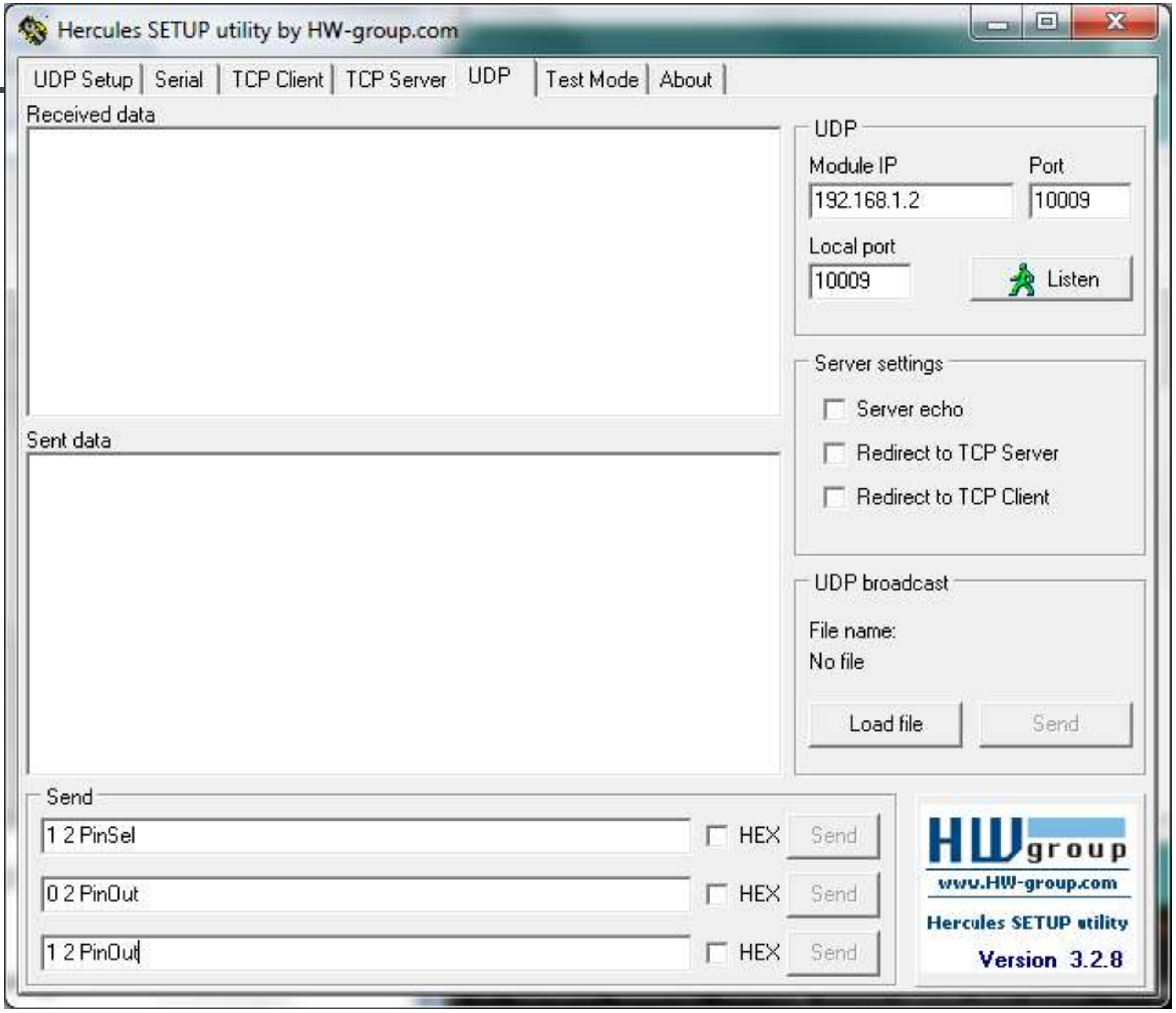
FW version:
Unknown

IP Filter

IP Filter Address Range

IP Filter MASK

4. Hercules UDP Sender



Received data

Sent data

UDP

Module IP: 192.168.1.2

Port: 10009

Local port: 10009

Server settings

- [ ] Server echo
- [ ] Redirect to TCP Server
- [ ] Redirect to TCP Client

UDP broadcast

File name:
No file

Send

1 2 PinSel	[ ] HEX	Send
0 2 PinOut	[ ] HEX	Send
1 2 PinOut	[ ] HEX	Send

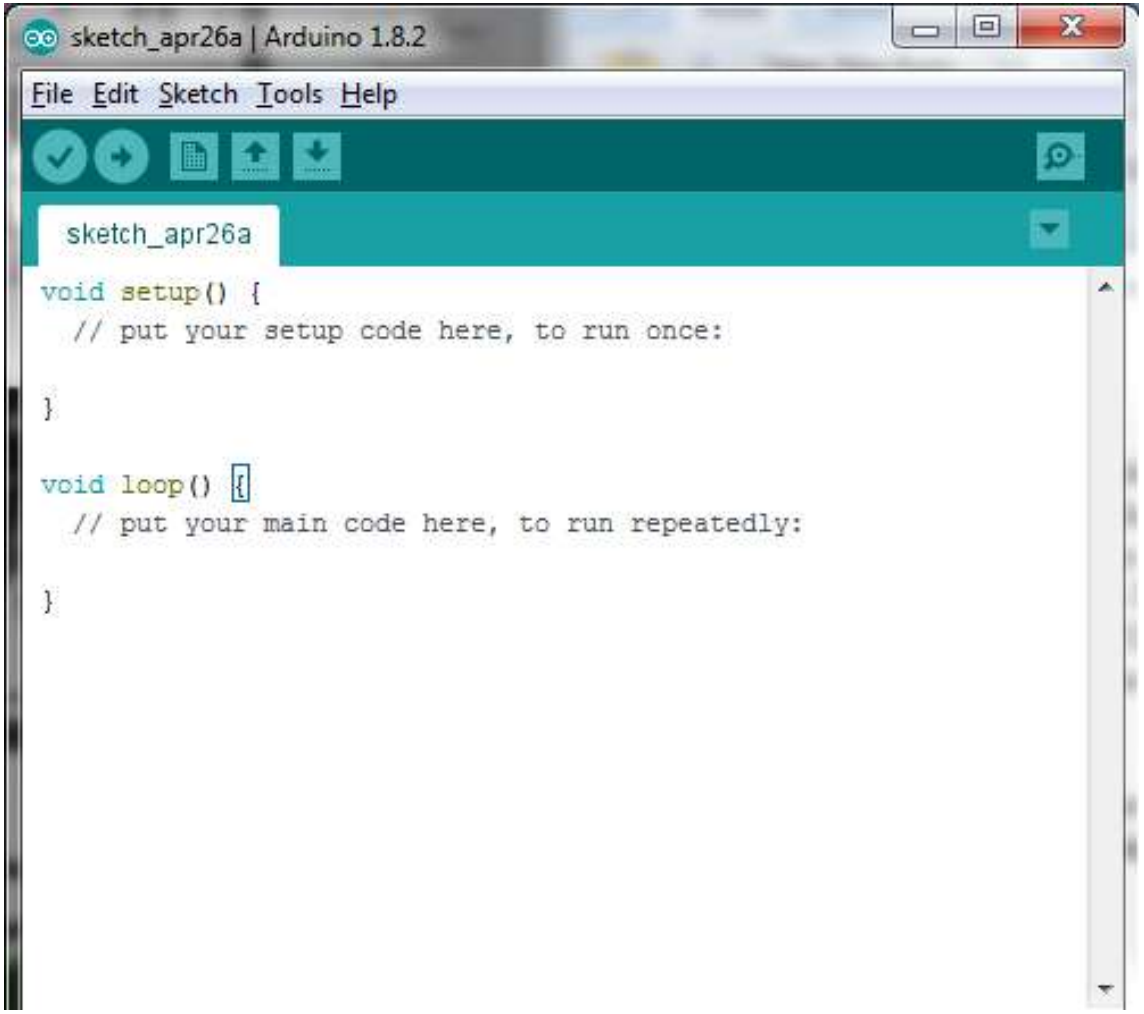
Experiments

1. espForth
2. Arduino UDP Server
3. MicroPython and WebREPL
4. MicroPython UDP Server
5. Lua UDP Server

1. espForth

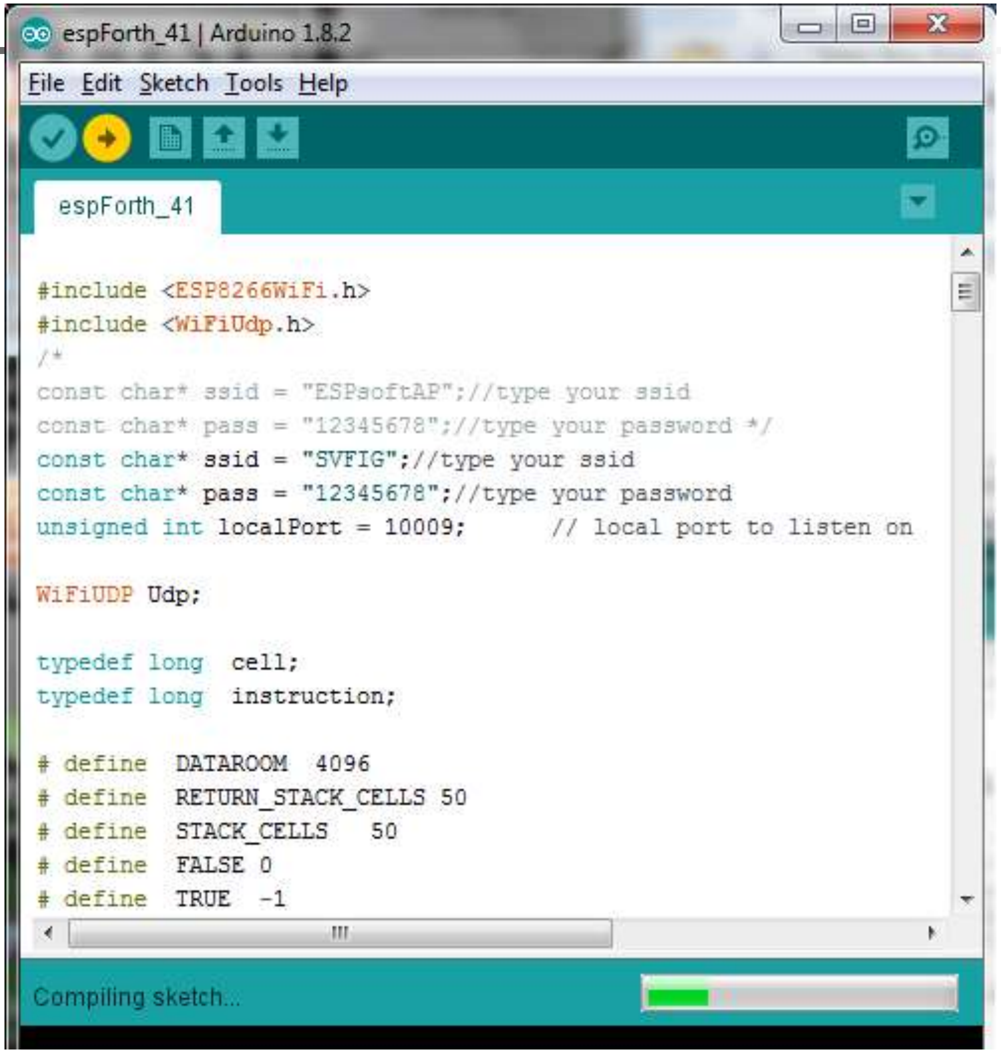
1. Open Arduino IDE
2. Compile espForth_41.ino
3. Open Serial Monitor
4. Open Hercules UDP Sender
5. Send UDP Packets to Control LED remotely

1. Arduino IDE



```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

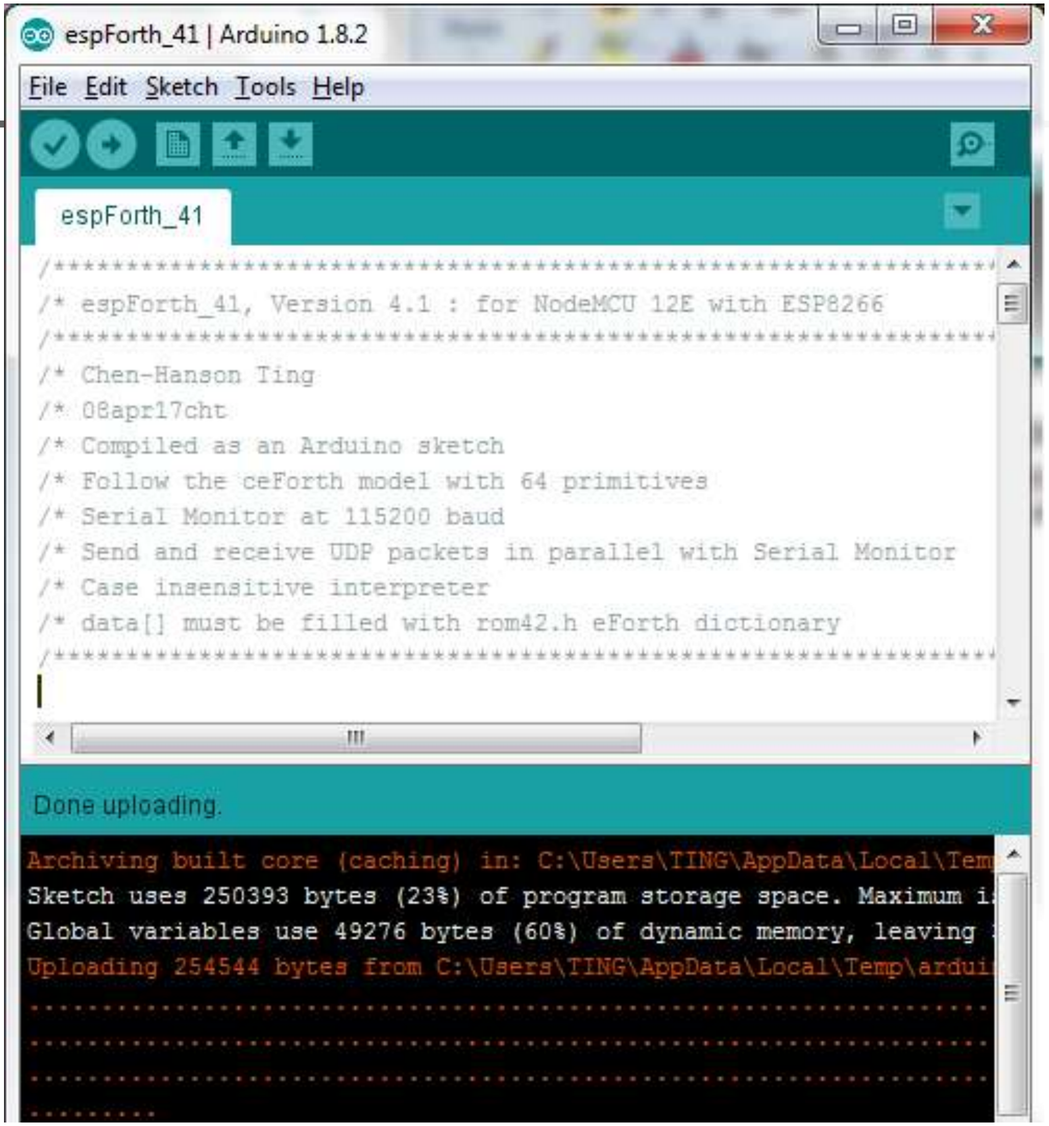
1. espForth



```
#include <ESP8266WiFi.h>
#include <WiFiUdp.h>
/*
const char* ssid = "ESPsoftAP";//type your ssid
const char* pass = "12345678";//type your password */
const char* ssid = "SVFIG";//type your ssid
const char* pass = "12345678";//type your password
unsigned int localPort = 10009;      // local port to listen on

WiFiUDP Udp;

typedef long  cell;
typedef long  instruction;

# define  DATAROOM  4096
# define  RETURN_STACK_CELLS 50
# define  STACK_CELLS   50
# define  FALSE 0
# define  TRUE  -1
```

Compiling sketch...

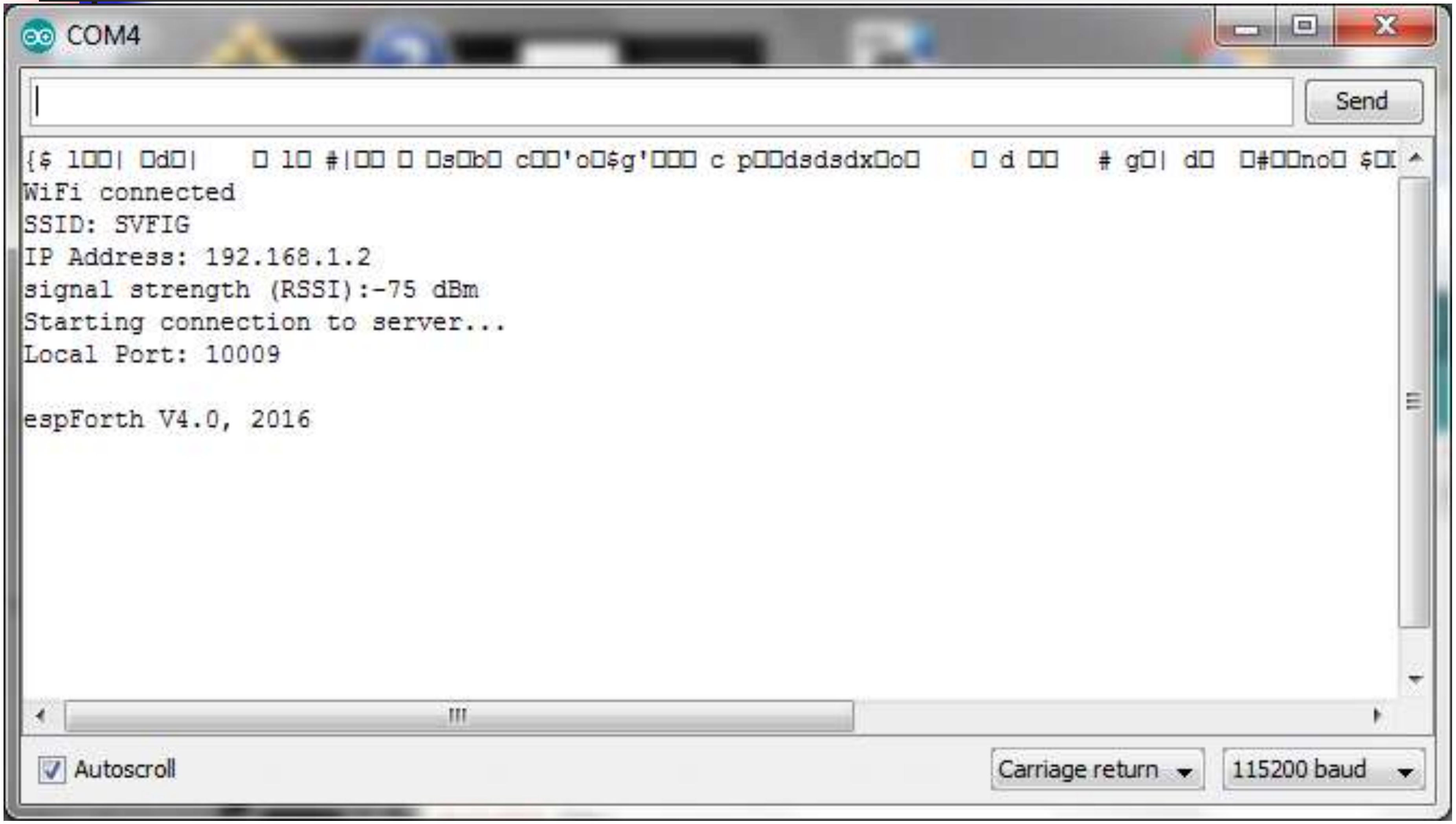
1. espForth



```
/*****************************************************************
/* espForth_41, Version 4.1 : for NodeMCU 12E with ESP8266
/*****************************************************************
/* Chen-Hanson Ting
/* 08apr17cht
/* Compiled as an Arduino sketch
/* Follow the ceForth model with 64 primitives
/* Serial Monitor at 115200 baud
/* Send and receive UDP packets in parallel with Serial Monitor
/* Case insensitive interpreter
/* data[] must be filled with rom42.h eForth dictionary
/*****************************************************************
```

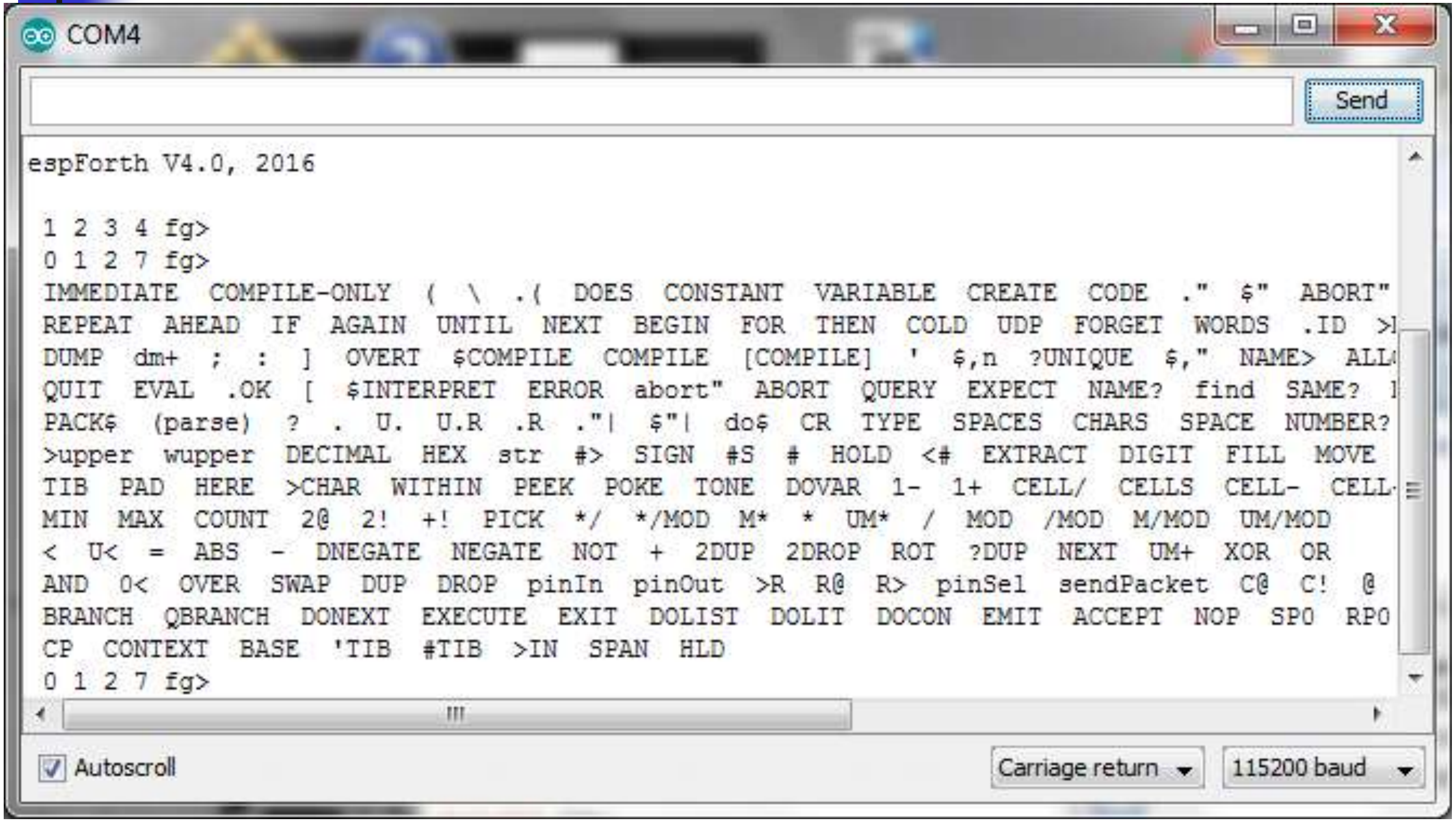
```
Archiving built core (caching) in: C:\Users\TING\AppData\Local\Tem
Sketch uses 250393 bytes (23%) of program storage space. Maximum i
Global variables use 49276 bytes (60%) of dynamic memory, leaving
Uploading 254544 bytes from C:\Users\TING\AppData\Local\Temp\ardui
....................................................................
....................................................................
....................................................................
.........
```

1. Serial Monitor



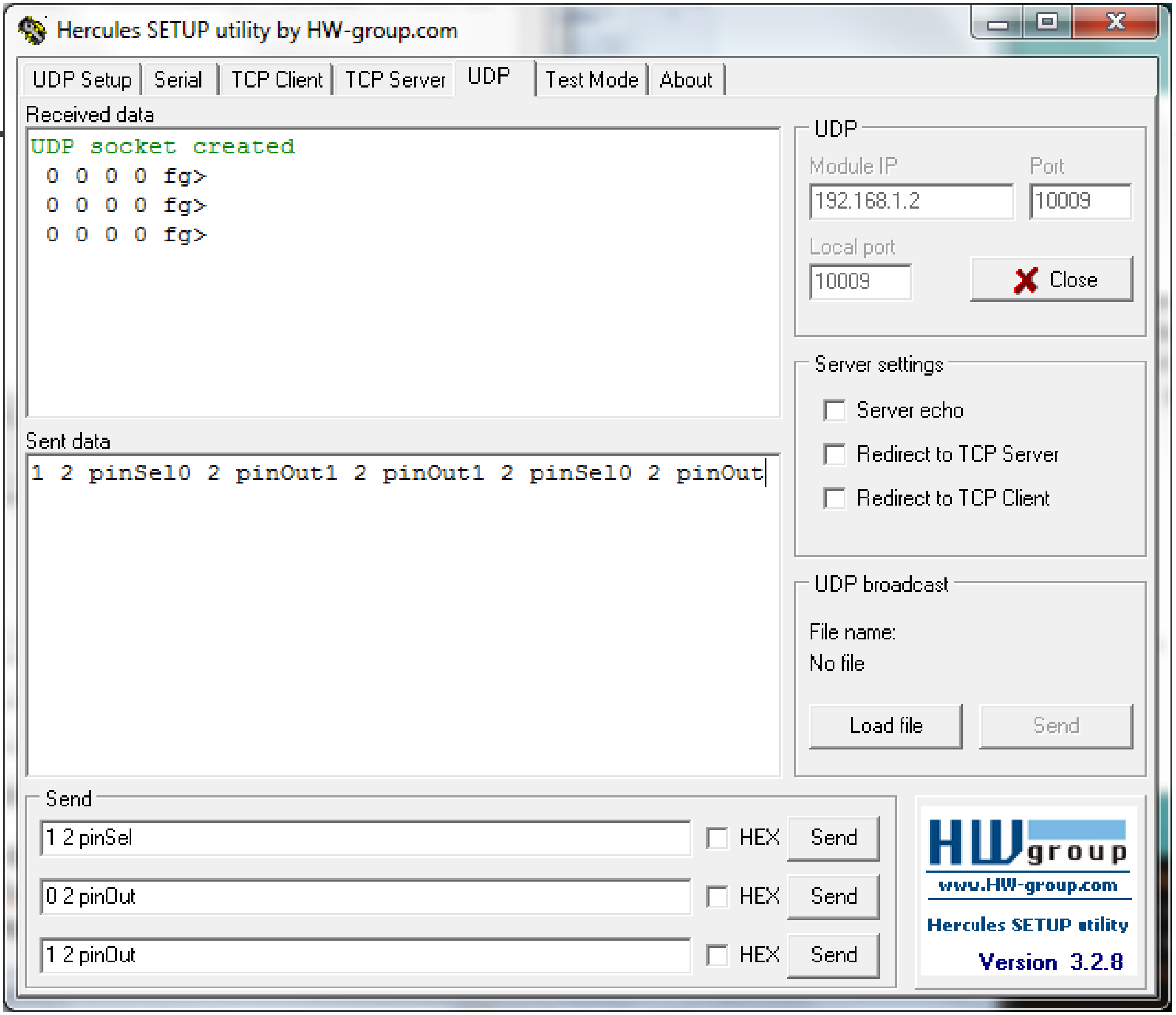
```
{$ 1□□| □d□|    □ 1□ #|□□ □ □s□b□ c□□'o□$g'□□□ c p□□dsdsdx□o□    □ d □□   # g□| d□  □#□□no□ $□
WiFi connected
SSID: SVFIG
IP Address: 192.168.1.2
signal strength (RSSI):-75 dBm
Starting connection to server...
Local Port: 10009

espForth V4.0, 2016
```

1. espForth



```
espForth V4.0, 2016

 1 2 3 4 fg>
 0 1 2 7 fg>
 IMMEDIATE  COMPILE-ONLY  (  \  .(  DOES  CONSTANT  VARIABLE  CREATE  CODE  ."  $"  ABORT"
 REPEAT  AHEAD  IF  AGAIN  UNTIL  NEXT  BEGIN  FOR  THEN  COLD  UDP  FORGET  WORDS  .ID  >
 DUMP  dm+  ;  :  ]  OVERT  $COMPILE  COMPILE  [COMPILE]  '  $,n  ?UNIQUE  $,"  NAME>  ALL
 QUIT  EVAL  .OK  [  $INTERPRET  ERROR  abort"  ABORT  QUERY  EXPECT  NAME?  find  SAME?  
 PACK$  (parse)  ?  .  U.  U.R  .R  ."|  $"|  do$  CR  TYPE  SPACES  CHARS  SPACE  NUMBER?
 >upper  wupper  DECIMAL  HEX  str  #>  SIGN  #S  #  HOLD  <#  EXTRACT  DIGIT  FILL  MOVE
 TIB  PAD  HERE  >CHAR  WITHIN  PEEK  POKE  TONE  DOVAR  1-  1+  CELL/  CELLS  CELL-  CELL
 MIN  MAX  COUNT  2@  2!  +!  PICK  */  */MOD  M*  *  UM*  /  MOD  /MOD  M/MOD  UM/MOD
 <  U<  =  ABS  -  DNEGATE  NEGATE  NOT  +  2DUP  2DROP  ROT  ?DUP  NEXT  UM+  XOR  OR
 AND  0<  OVER  SWAP  DUP  DROP  pinIn  pinOut  >R  R@  R>  pinSel  sendPacket  C@  C!  @
 BRANCH  QBRANCH  DONEXT  EXECUTE  EXIT  DOLIST  DOLIT  DOCON  EMIT  ACCEPT  NOP  SP0  RP0
 CP  CONTEXT  BASE  'TIB  #TIB  >IN  SPAN  HLD
 0 1 2 7 fg>
```

1. UDP Packet Sender



Received data

```
UDP socket created
 0  0  0  0  fg>
 0  0  0  0  fg>
 0  0  0  0  fg>
```

Sent data

```
1 2 pinSel0 2 pinOut1 2 pinOut1 2 pinSel0 2 pinOut
```

UDP

Module IP: 192.168.1.2
Port: 10009
Local port: 10009

Server settings

- [ ] Server echo
- [ ] Redirect to TCP Server
- [ ] Redirect to TCP Client

UDP broadcast

File name:
No file

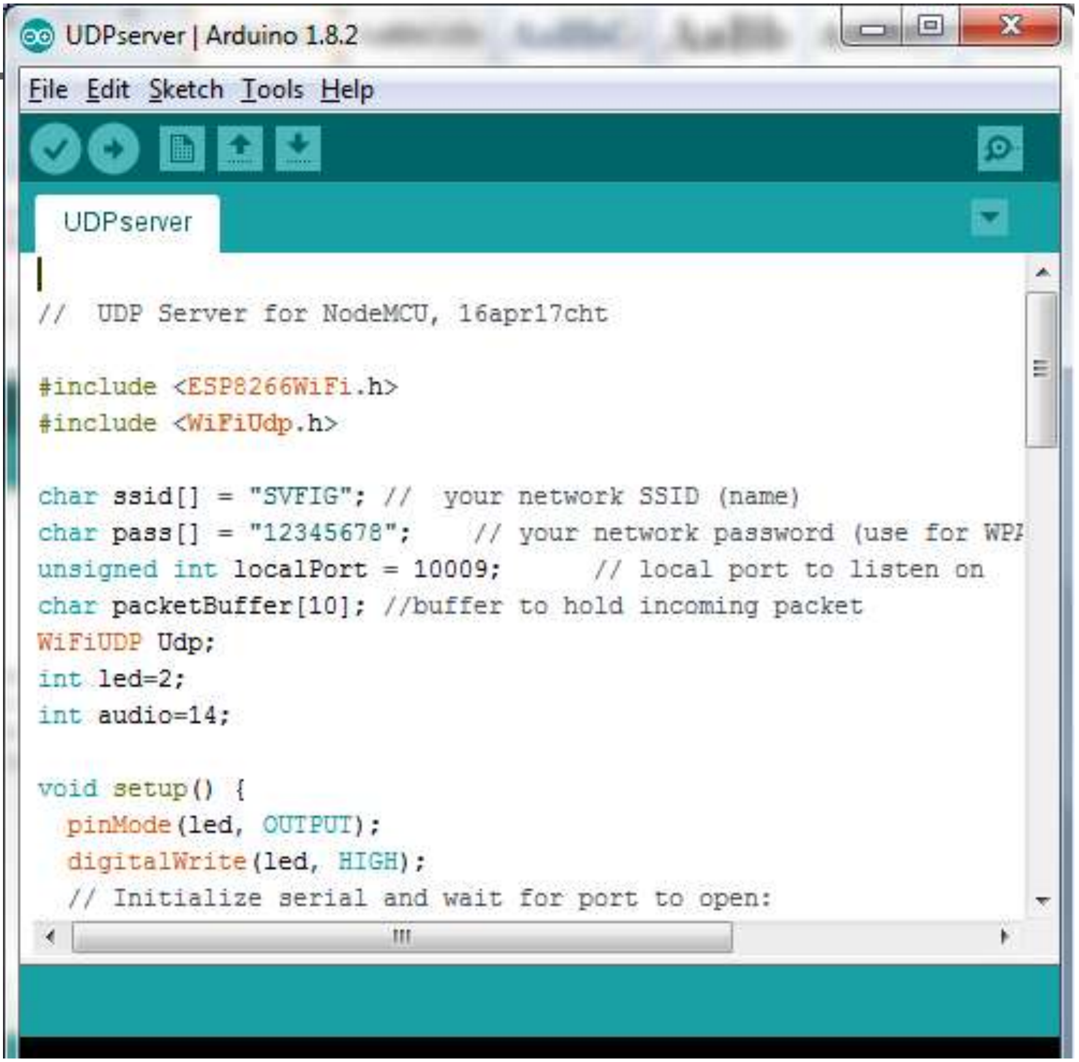
Send

1 2 pinSel	☐ HEX	Send
0 2 pinOut	☐ HEX	Send
1 2 pinOut	☐ HEX	Send

2. Arduino IDE UDP Server

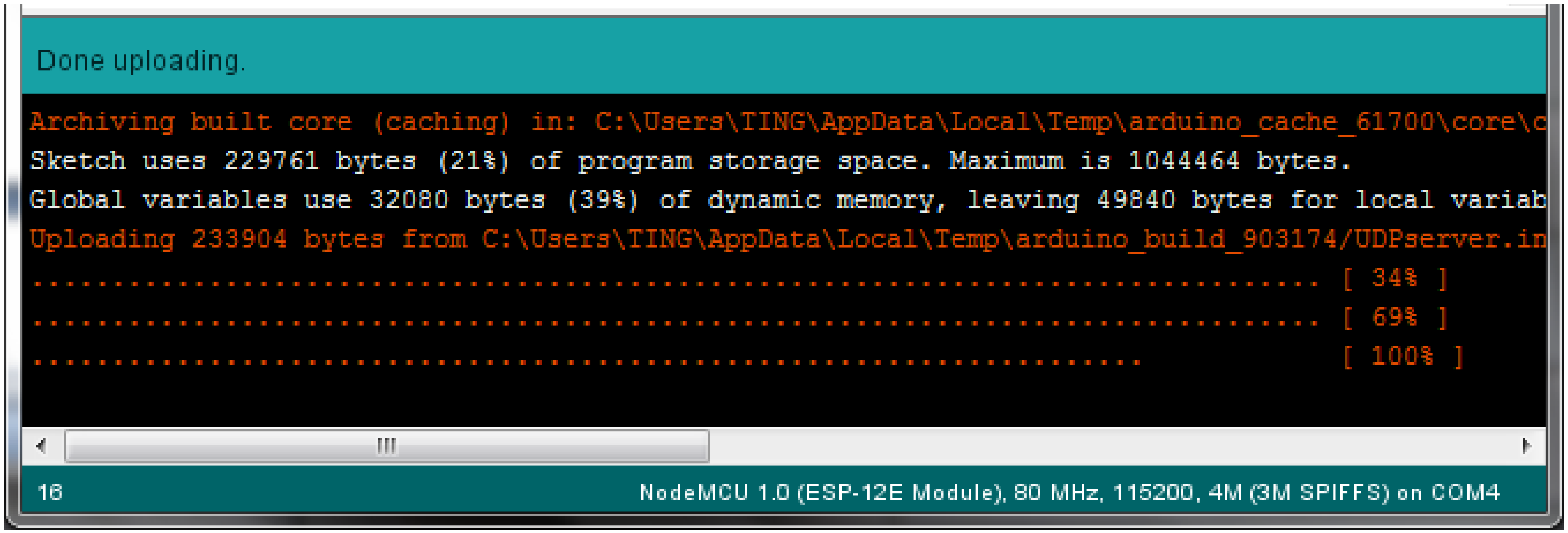
- Open Arduino IDE.
- Open UDPserver.ino project.
- Compile and upload UDPserver.
- Open Serial Monitor to see IP.
- Open Hercules to send UDP packets.

2. Arduino UDP Server



```
//  UDP Server for NodeMCU, 16apr17cht

#include <ESP8266WiFi.h>
#include <WiFiUdp.h>

char ssid[] = "SVFIG"; //  your network SSID (name)
char pass[] = "12345678";    // your network password (use for WPA
unsigned int localPort = 10009;      // local port to listen on
char packetBuffer[10]; //buffer to hold incoming packet
WiFiUDP Udp;
int led=2;
int audio=14;

void setup() {
  pinMode(led, OUTPUT);
  digitalWrite(led, HIGH);
  // Initialize serial and wait for port to open:
```

2. Arduino UDP Server

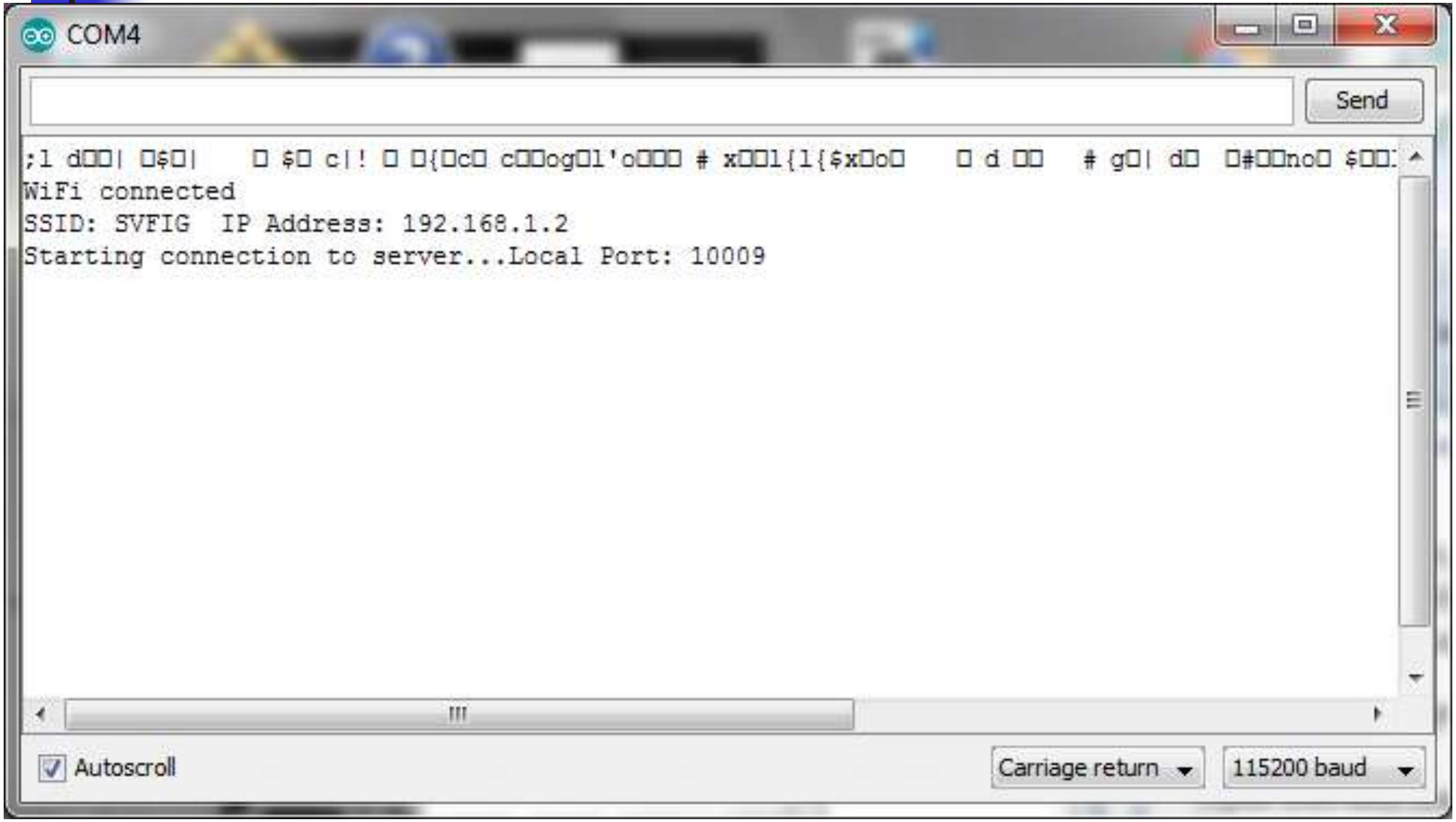


Done uploading.

```
Archiving built core (caching) in: C:\Users\TING\AppData\Local\Temp\arduino_cache_61700\core\c
Sketch uses 229761 bytes (21%) of program storage space. Maximum is 1044464 bytes.
Global variables use 32080 bytes (39%) of dynamic memory, leaving 49840 bytes for local variab
Uploading 233904 bytes from C:\Users\TING\AppData\Local\Temp\arduino_build_903174/UDPserver.in
................................................................................ [ 34% ]
................................................................................ [ 69% ]
.......................................................................          [ 100% ]
```

16   NodeMCU 1.0 (ESP-12E Module), 80 MHz, 115200, 4M (3M SPIFFS) on COM4

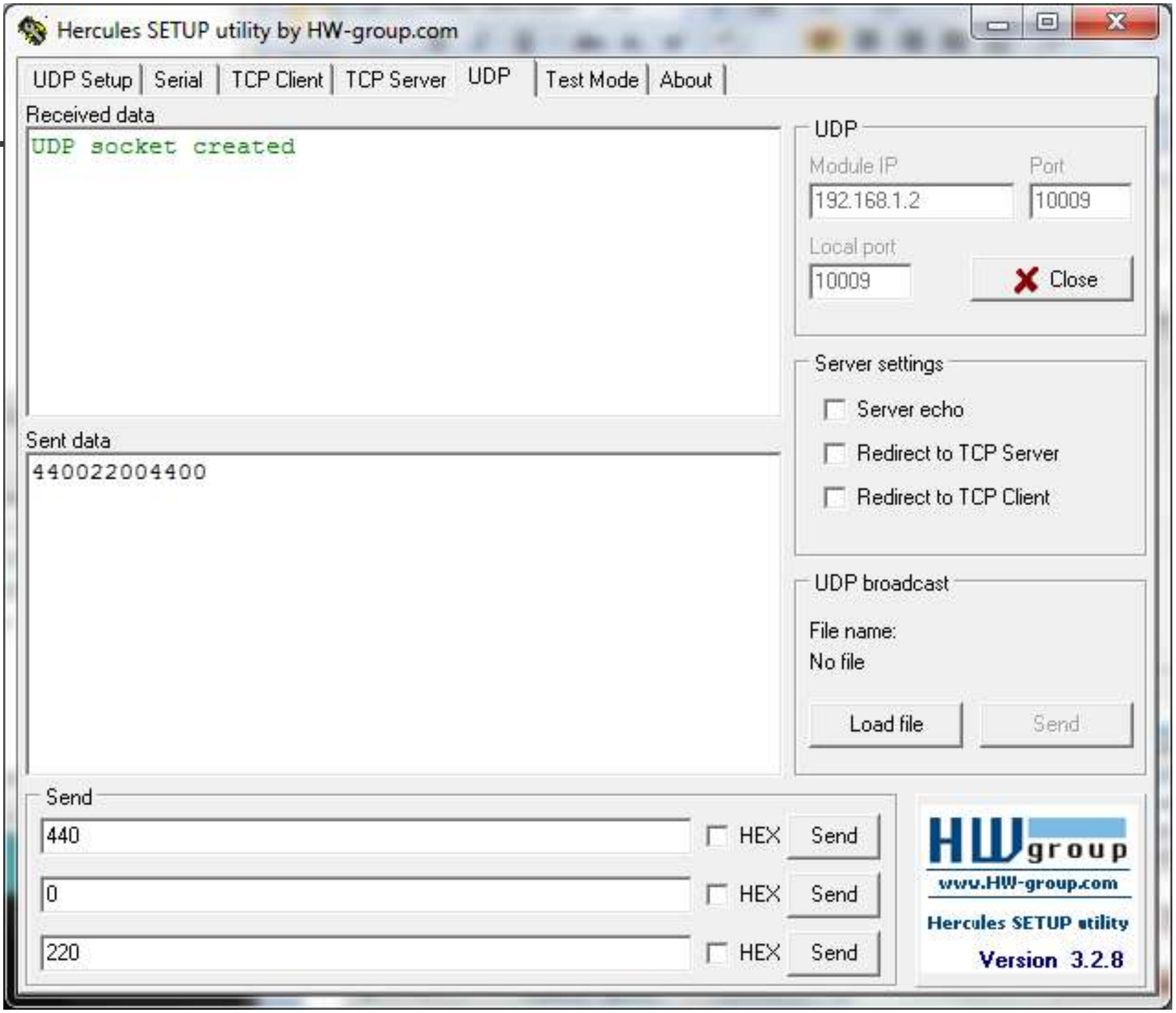
2. IP on Serial Monitor



```
;l d□□| □$□|    □ $□ c|! □ □{□c□ c□□og□l'o□□□ # x□□l{l{$x□o□    □ d □□    # g□| d□  □#□□no□ $□□
WiFi connected
SSID: SVFIG  IP Address: 192.168.1.2
Starting connection to server...Local Port: 10009
```

[✓] Autoscroll   Carriage return   115200 baud

2. Hercules UDP Sender


Hercules SETUP utility by HW-group.com

Received data

```
UDP socket created
```

Sent data

```
440022004400
```

UDP

Module IP: 192.168.1.2
Port: 10009
Local port: 10009

Server settings

- [ ] Server echo
- [ ] Redirect to TCP Server
- [ ] Redirect to TCP Client

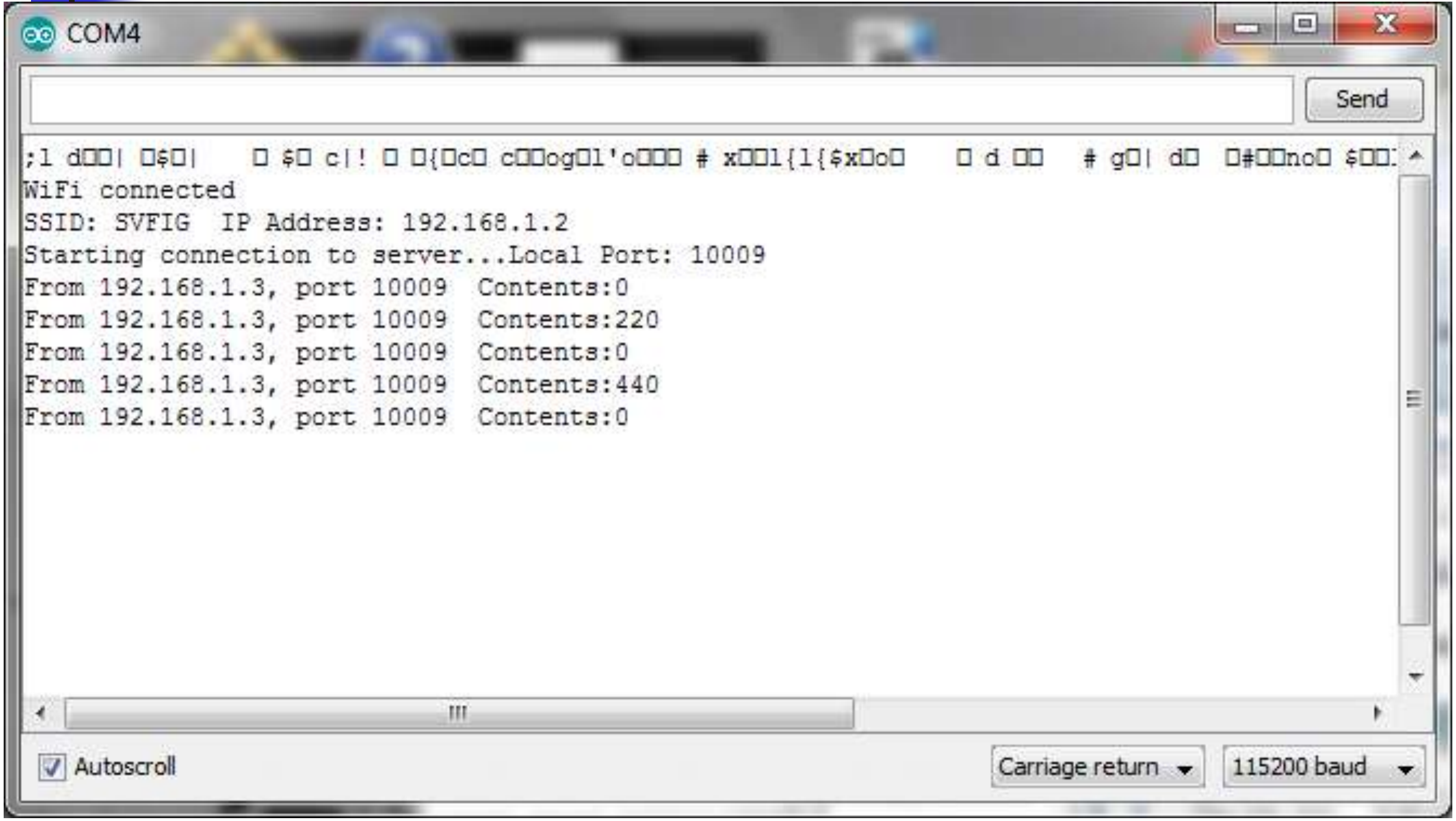
UDP broadcast

File name:
No file

Send

440	[ ] HEX	Send
0	[ ] HEX	Send
220	[ ] HEX	Send

2. Serial Monitor Logging

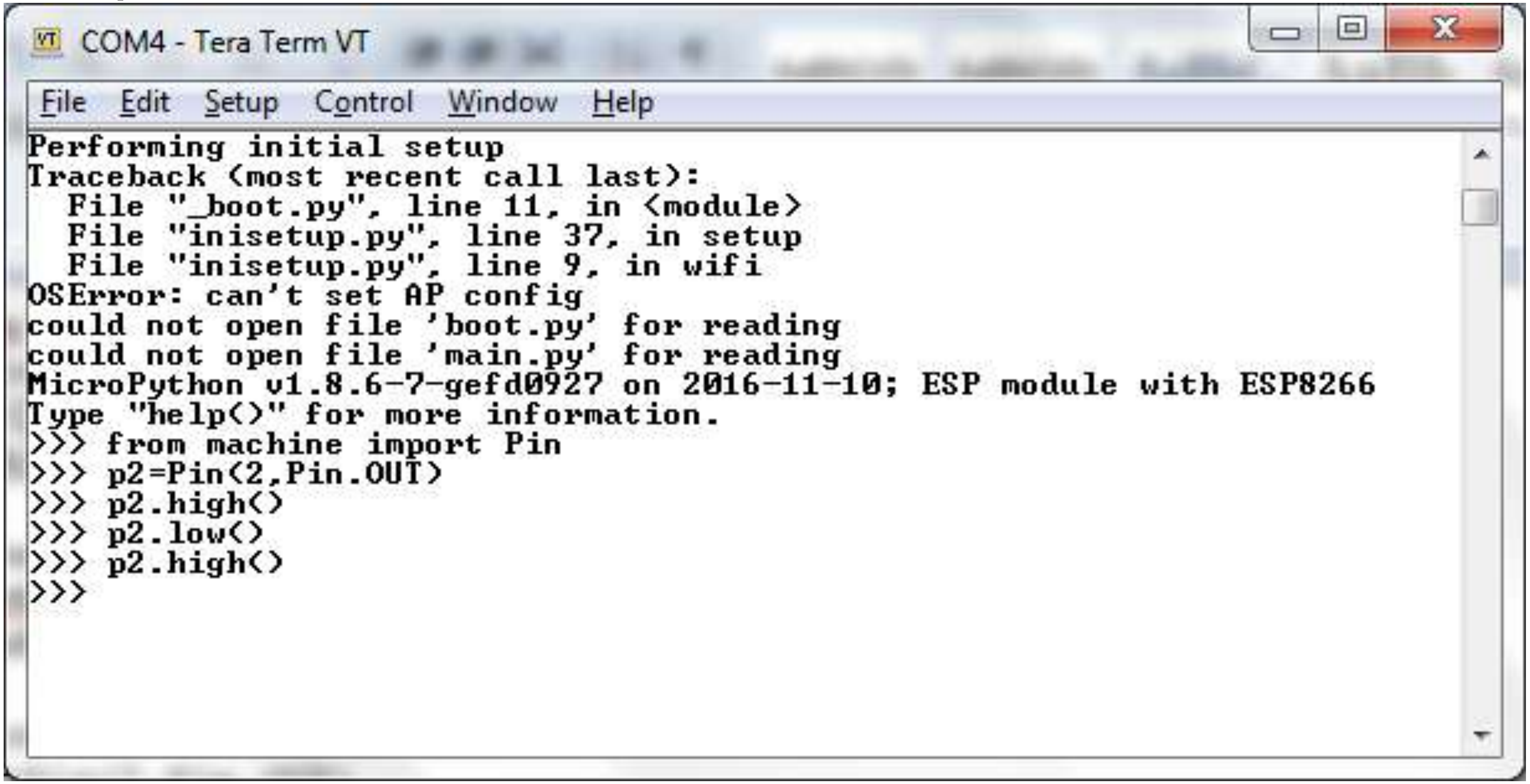


```
;1 d□□| □$□|    □ $□ c|! □ □{□c□ c□□og□1'o□□□ # x□□1{1{$x□o□    □ d □□   # g□| d□  □#□□no□ $□□
WiFi connected
SSID: SVFIG  IP Address: 192.168.1.2
Starting connection to server...Local Port: 10009
From 192.168.1.3, port 10009  Contents:0
From 192.168.1.3, port 10009  Contents:220
From 192.168.1.3, port 10009  Contents:0
From 192.168.1.3, port 10009  Contents:440
From 192.168.1.3, port 10009  Contents:0
```

3. MicroPython/WebREPL

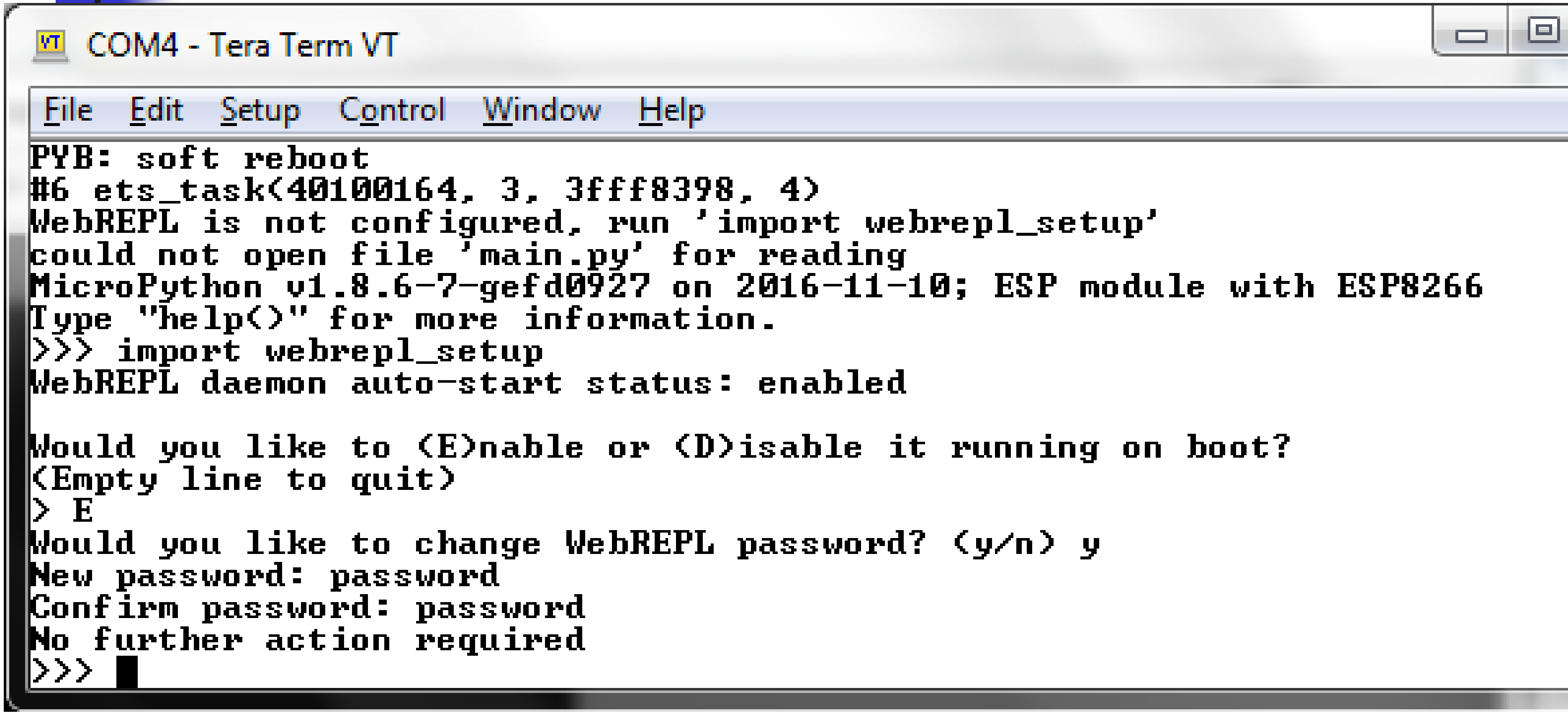
- Flash MicroPython
- Open TeraTerm Terminal
- Control LED with REPL
- import webrepl_setup
- Control LED with WebREPL

3. MicroPython REPL



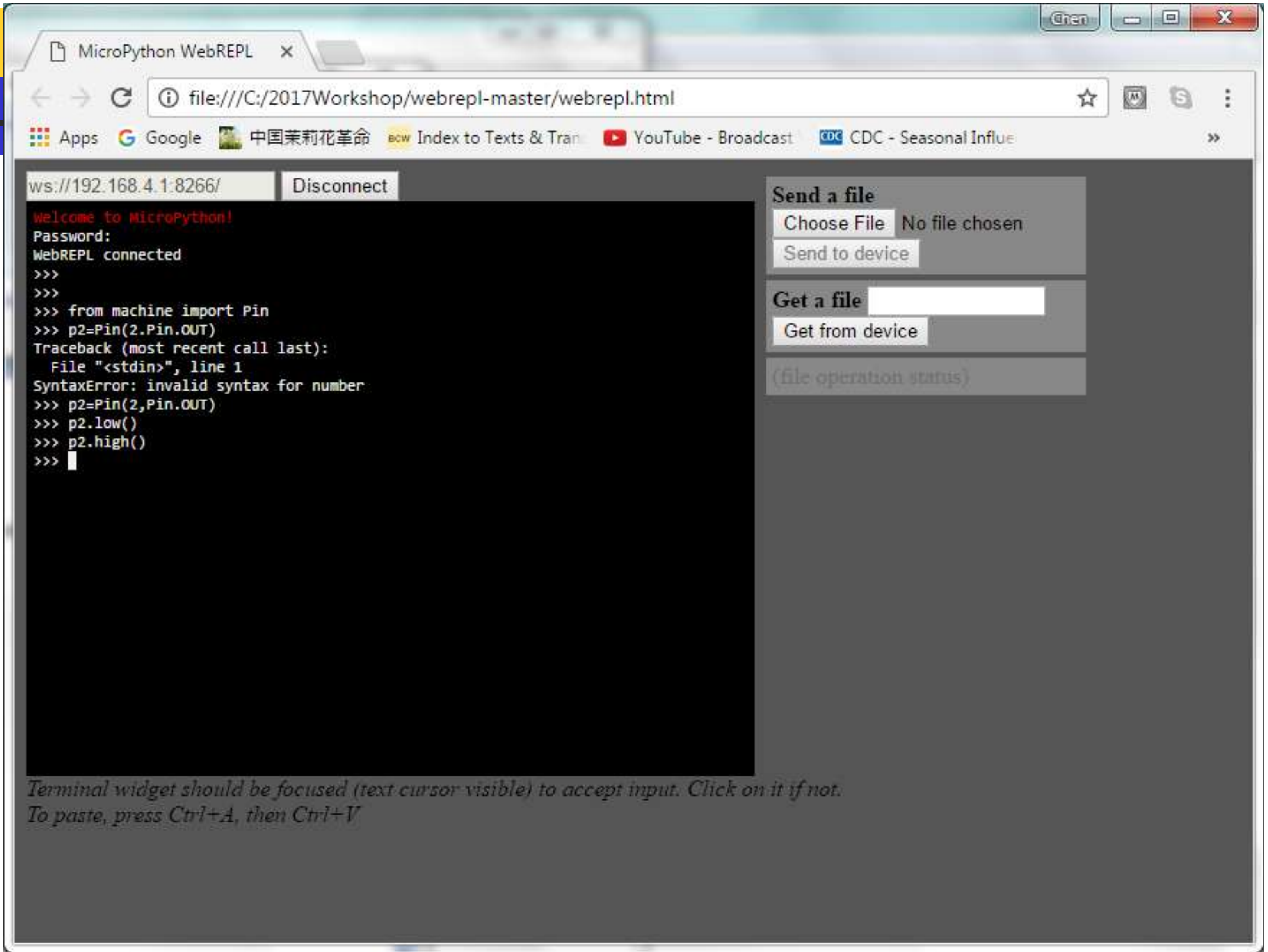
```
Performing initial setup
Traceback (most recent call last):
  File "_boot.py", line 11, in <module>
  File "inisetup.py", line 37, in setup
  File "inisetup.py", line 9, in wifi
OSError: can't set AP config
could not open file 'boot.py' for reading
could not open file 'main.py' for reading
MicroPython v1.8.6-7-gefd0927 on 2016-11-10; ESP module with ESP8266
Type "help()" for more information.
>>> from machine import Pin
>>> p2=Pin(2,Pin.OUT)
>>> p2.high()
>>> p2.low()
>>> p2.high()
>>>
```

3. Install WebREPL



```
PYB: soft reboot
#6 ets_task(40100164, 3, 3fff8398, 4)
WebREPL is not configured, run 'import webrepl_setup'
could not open file 'main.py' for reading
MicroPython v1.8.6-7-gefd0927 on 2016-11-10; ESP module with ESP8266
Type "help()" for more information.
>>> import webrepl_setup
WebREPL daemon auto-start status: enabled

Would you like to (E)nable or (D)isable it running on boot?
(Empty line to quit)
> E
Would you like to change WebREPL password? (y/n) y
New password: password
Confirm password: password
No further action required
>>> 
```

3. Run WebREPL



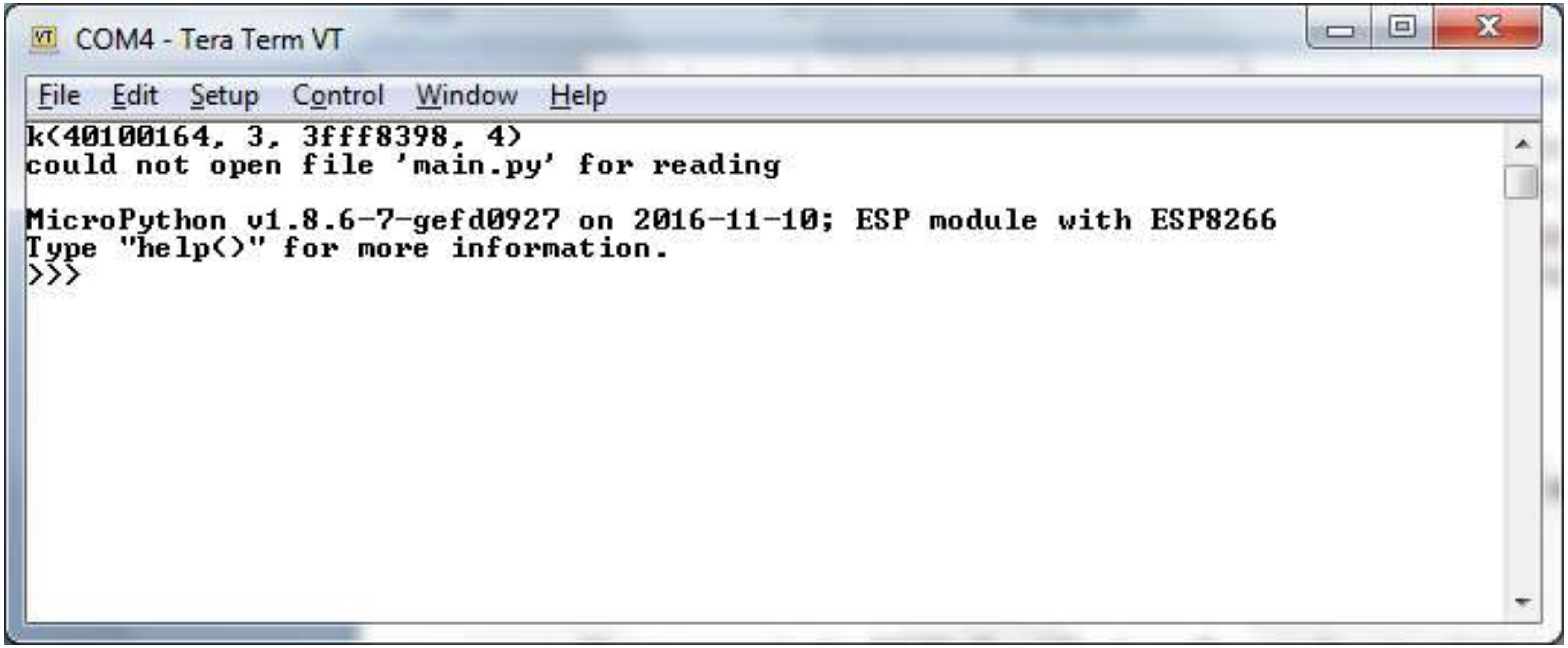
```
Welcome to MicroPython!
Password:
WebREPL connected
>>>
>>>
>>> from machine import Pin
>>> p2=Pin(2.Pin.OUT)
Traceback (most recent call last):
  File "<stdin>", line 1
SyntaxError: invalid syntax for number
>>> p2=Pin(2,Pin.OUT)
>>> p2.low()
>>> p2.high()
>>> 
```

Terminal widget should be focused (text cursor visible) to accept input. Click on it if not.
To paste, press Ctrl+A, then Ctrl+V

4. MicroPython UDP Server

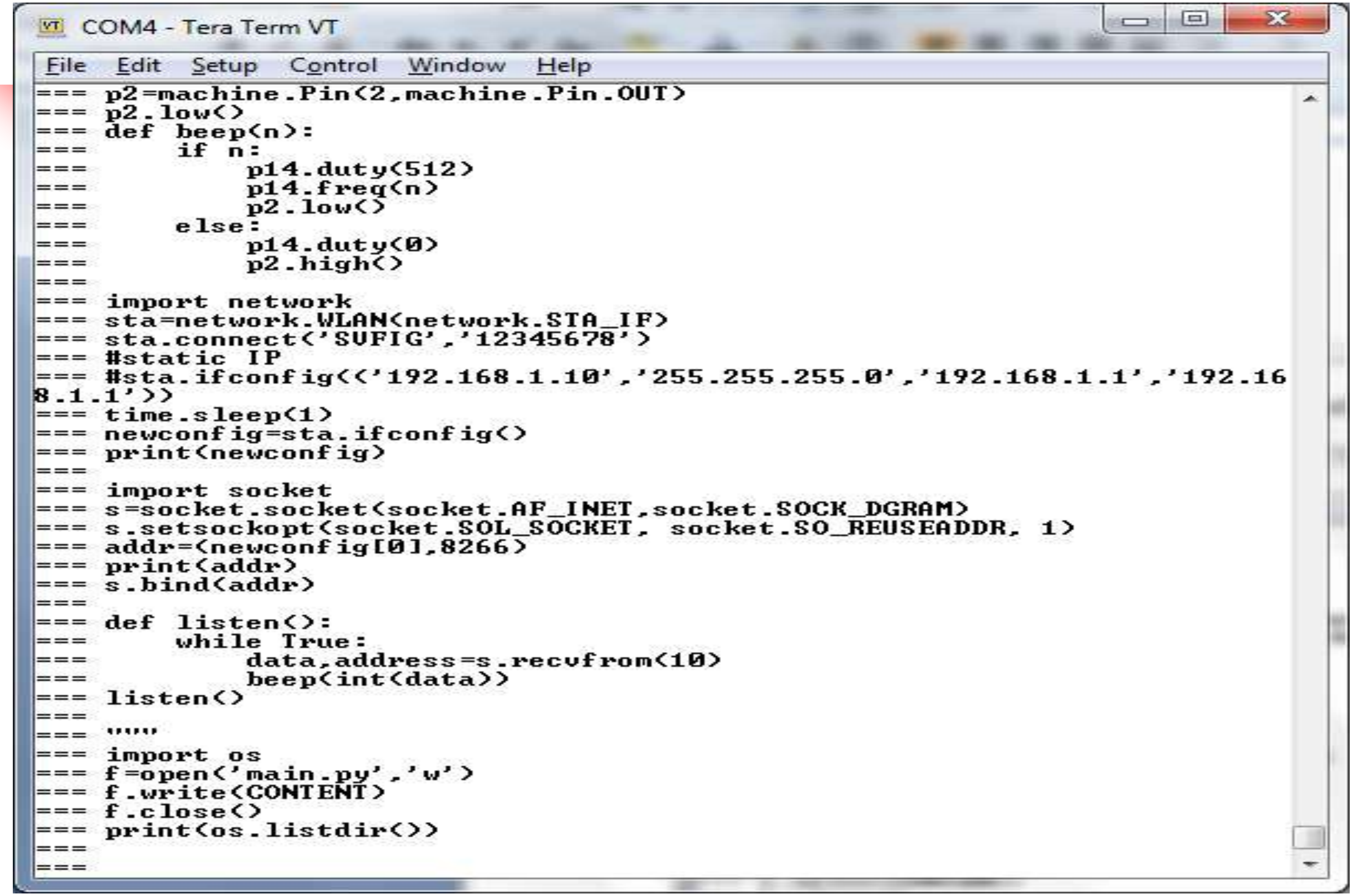
- Flash MicroPython
- Open TeraTerm Terminal
- Load WriteFile.py on NodeMCU
- Open Hercules UDP Sender
- Send UDP packets to control LED

4. MicroPython REPL



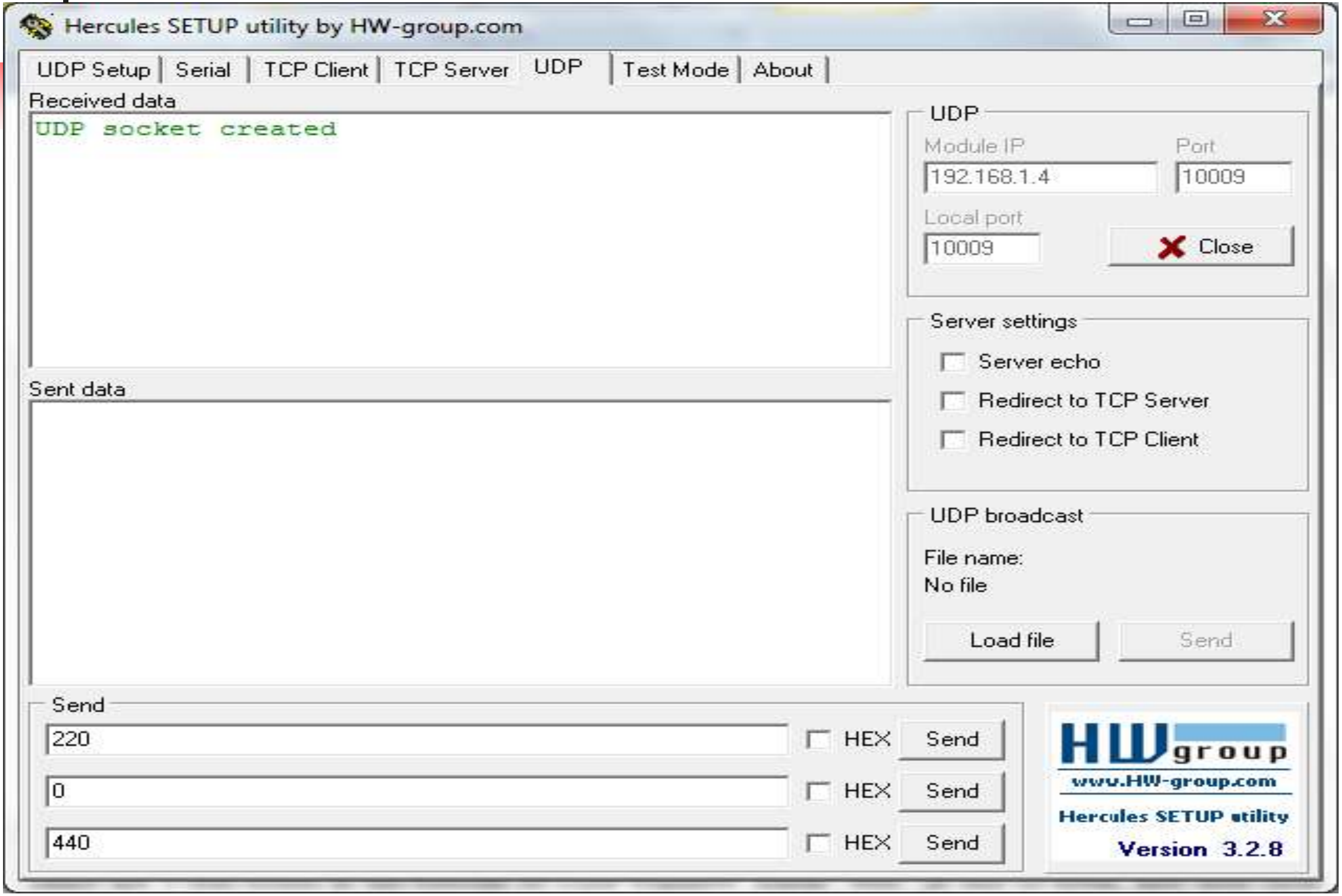
```
k(40100164, 3, 3fff8398, 4)
could not open file 'main.py' for reading

MicroPython v1.8.6-7-gefd0927 on 2016-11-10; ESP module with ESP8266
Type "help()" for more information.
>>>
```

4. Load WriteFile.py



```
=== p2=machine.Pin(2,machine.Pin.OUT)
=== p2.low()
=== def beep(n):
===     if n:
===         p14.duty(512)
===         p14.freq(n)
===         p2.low()
===     else:
===         p14.duty(0)
===         p2.high()
===
=== import network
=== sta=network.WLAN(network.STA_IF)
=== sta.connect('SVFIG','12345678')
=== #static IP
=== #sta.ifconfig(('192.168.1.10','255.255.255.0','192.168.1.1','192.16
8.1.1'))
=== time.sleep(1)
=== newconfig=sta.ifconfig()
=== print(newconfig)
===
=== import socket
=== s=socket.socket(socket.AF_INET,socket.SOCK_DGRAM)
=== s.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
=== addr=(newconfig[0],8266)
=== print(addr)
=== s.bind(addr)
===
=== def listen():
===     while True:
===         data,address=s.recvfrom(10)
===         beep(int(data))
=== listen()
===
=== """
=== import os
=== f=open('main.py','w')
=== f.write(CONTENT)
=== f.close()
=== print(os.listdir())
===
===
```

4. Send UDP Packets



Received data

```
UDP socket created
```

Sent data

UDP

Module IP: 192.168.1.4
Port: 10009
Local port: 10009

Server settings

- [ ] Server echo
- [ ] Redirect to TCP Server
- [ ] Redirect to TCP Client

UDP broadcast

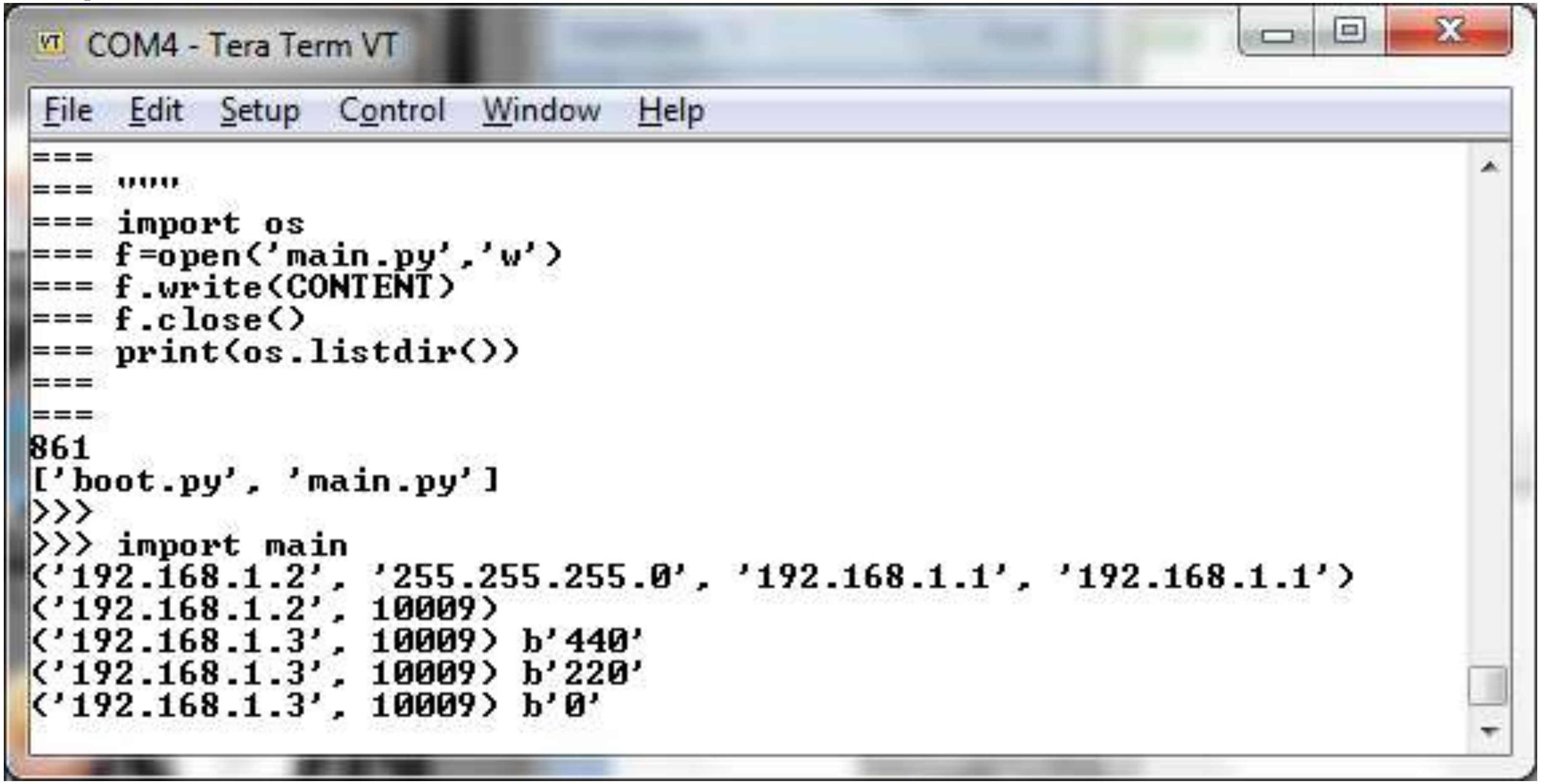
File name:
No file

Send

220	[ ] HEX	Send
0	[ ] HEX	Send
440	[ ] HEX	Send

HW group
www.HW-group.com
Hercules SETUP utility
Version 3.2.8

4. UDP Packet Log

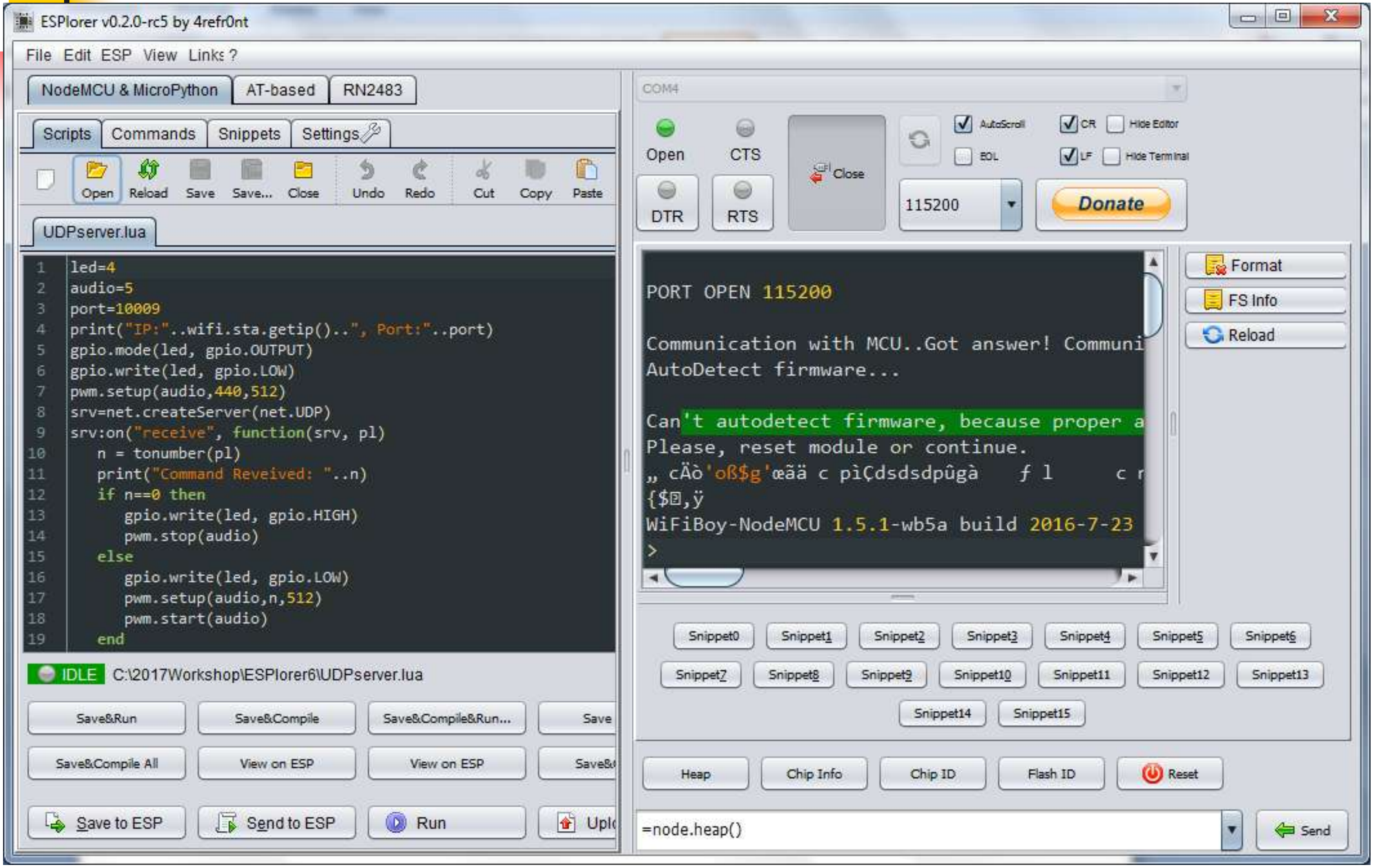


```
===
=== """
=== import os
=== f=open('main.py','w')
=== f.write(CONTENT)
=== f.close()
=== print(os.listdir())
===
===
861
['boot.py', 'main.py']
>>>
>>> import main
('192.168.1.2', '255.255.255.0', '192.168.1.1', '192.168.1.1')
('192.168.1.2', 10009)
('192.168.1.3', 10009) b'440'
('192.168.1.3', 10009) b'220'
('192.168.1.3', 10009) b'0'
```

5. Lua UDP Server

- Flash Lua on NodeMCU.
- Open ESPlorer.
- Open UDPserver.lua on Editor.
- Press Save to ESP button to compile server.
- Open Hercules to send UDP packets to NodeMCU.

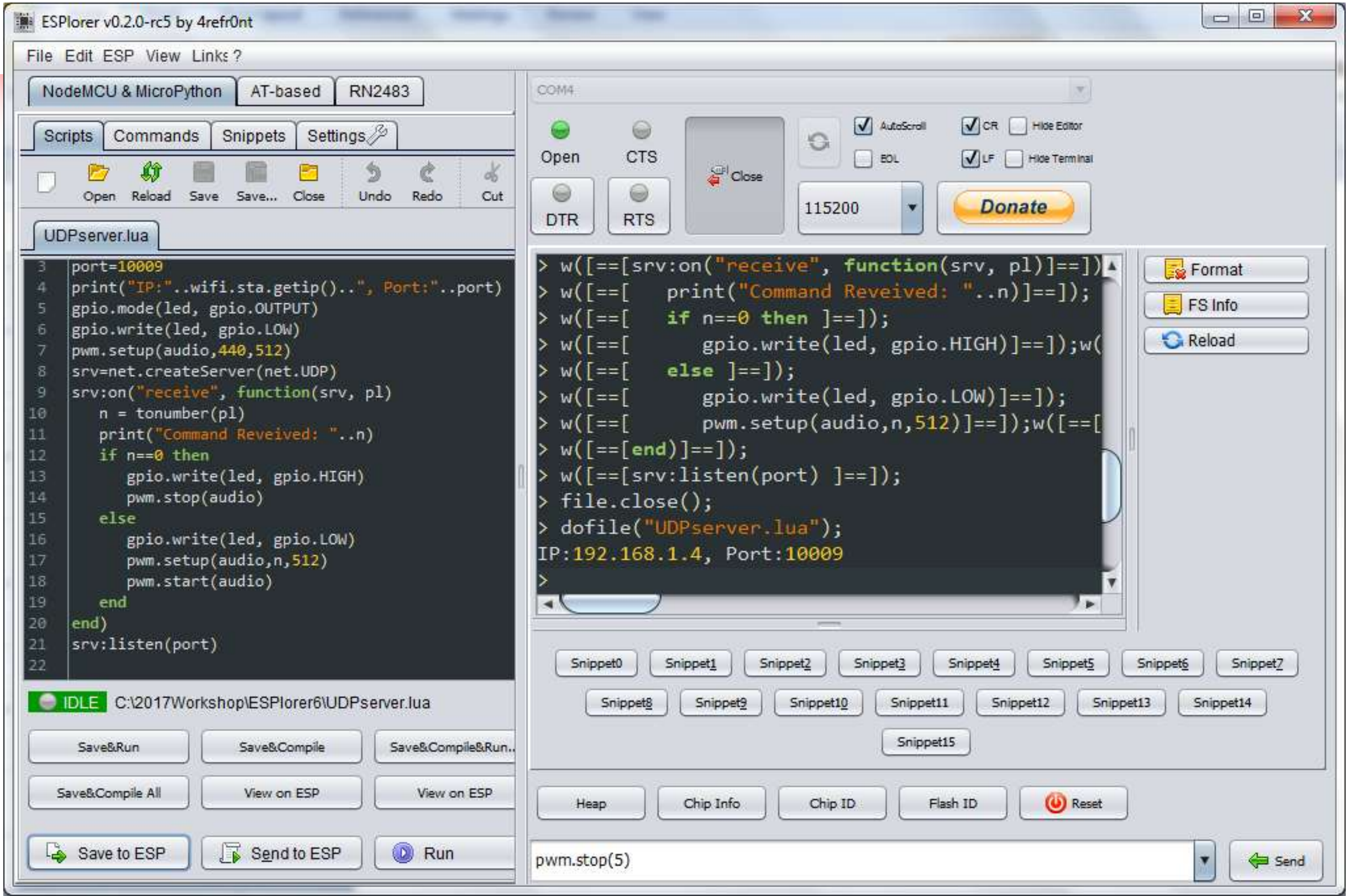
5. Open UDPserver



```lua
led=4
audio=5
port=10009
print("IP:"..wifi.sta.getip()..", Port:"..port)
gpio.mode(led, gpio.OUTPUT)
gpio.write(led, gpio.LOW)
pwm.setup(audio,440,512)
srv=net.createServer(net.UDP)
srv:on("receive", function(srv, pl)
   n = tonumber(pl)
   print("Command Reveived: "..n)
   if n==0 then
      gpio.write(led, gpio.HIGH)
      pwm.stop(audio)
   else
      gpio.write(led, gpio.LOW)
      pwm.setup(audio,n,512)
      pwm.start(audio)
   end
```

```
PORT OPEN 115200

Communication with MCU..Got answer! Communi
AutoDetect firmware...

Can't autodetect firmware, because proper a
Please, reset module or continue.
„ cÄò'oß$g'œãä c piÇdsdsdpûgà   ƒ l     c r
{$▯,ÿ
WiFiBoy-NodeMCU 1.5.1-wb5a build 2016-7-23
>
```

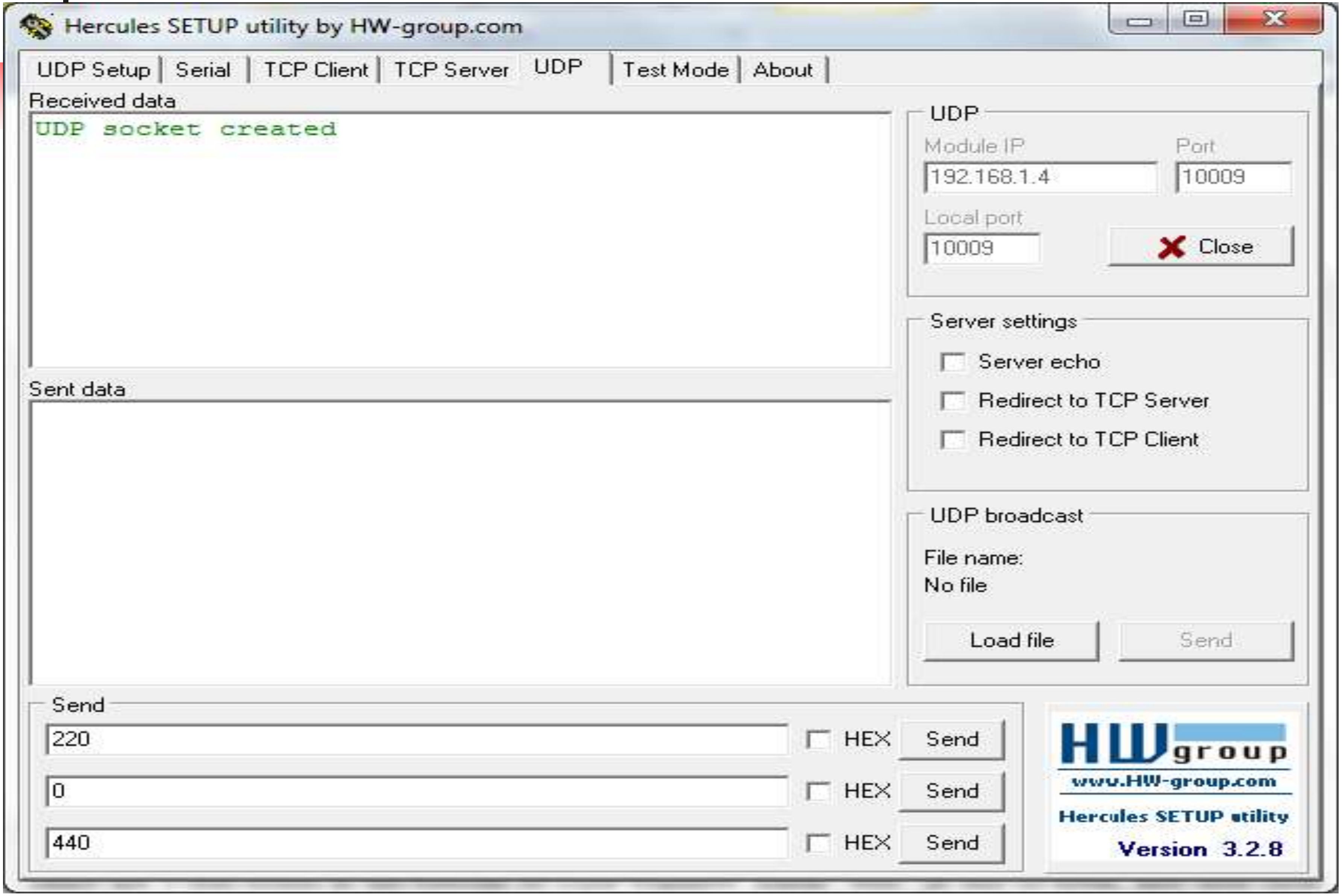
5. Compile UDPserver



```
port=10009
print("IP:"..wifi.sta.getip()..", Port:"..port)
gpio.mode(led, gpio.OUTPUT)
gpio.write(led, gpio.LOW)
pwm.setup(audio,440,512)
srv=net.createServer(net.UDP)
srv:on("receive", function(srv, pl)
   n = tonumber(pl)
   print("Command Reveived: "..n)
   if n==0 then
      gpio.write(led, gpio.HIGH)
      pwm.stop(audio)
   else
      gpio.write(led, gpio.LOW)
      pwm.setup(audio,n,512)
      pwm.start(audio)
   end
end)
srv:listen(port)
```

```
> w([==[srv:on("receive", function(srv, pl)]==])
> w([==[   print("Command Reveived: "..n)]==]);
> w([==[   if n==0 then ]==]);
> w([==[      gpio.write(led, gpio.HIGH)]==]);w(
> w([==[   else ]==]);
> w([==[      gpio.write(led, gpio.LOW)]==]);
> w([==[      pwm.setup(audio,n,512)]==]);w([==[
> w([==[end)]==]);
> w([==[srv:listen(port) ]==]);
> file.close();
> dofile("UDPserver.lua");
IP:192.168.1.4, Port:10009
>
```

5. Send UDP Packets


Hercules SETUP utility by HW-group.com
UDP Setup | Serial | TCP Client | TCP Server | UDP | Test Mode | About
Close
Load file
Send
HEX
Send
HEX
Send
HEX
Send
HW group
www.HW-group.com
Hercules SETUP utility
Version 3.2.8

Received data

```
UDP socket created
```

Sent data

UDP
- Module IP: 192.168.1.4
- Port: 10009
- Local port: 10009

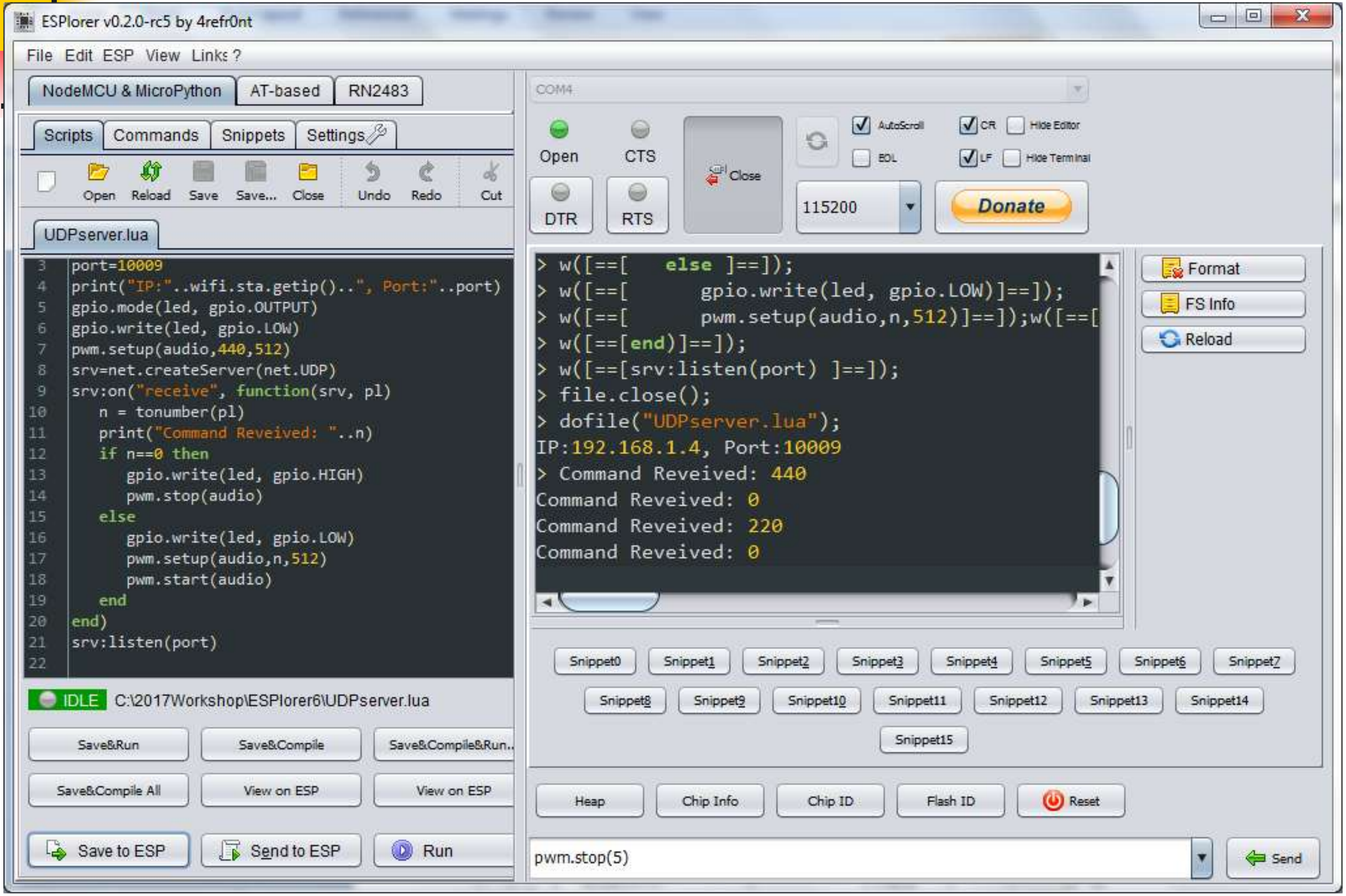
Server settings
- [ ] Server echo
- [ ] Redirect to TCP Server
- [ ] Redirect to TCP Client

UDP broadcast

File name:
No file

Send
- 220 [ ] HEX
- 0 [ ] HEX
- 440 [ ] HEX

5. UDP Packet Log



```lua
port=10009
print("IP:"..wifi.sta.getip()..", Port:"..port)
gpio.mode(led, gpio.OUTPUT)
gpio.write(led, gpio.LOW)
pwm.setup(audio,440,512)
srv=net.createServer(net.UDP)
srv:on("receive", function(srv, pl)
   n = tonumber(pl)
   print("Command Reveived: "..n)
   if n==0 then
      gpio.write(led, gpio.HIGH)
      pwm.stop(audio)
   else
      gpio.write(led, gpio.LOW)
      pwm.setup(audio,n,512)
      pwm.start(audio)
   end
end)
srv:listen(port)
```

C:\2017Workshop\ESPlorer6\UDPserver.lua

```
> w([==[   else ]==]);
> w([==[      gpio.write(led, gpio.LOW)]==]);
> w([==[      pwm.setup(audio,n,512)]==]);w([==[
> w([==[end)]==]);
> w([==[srv:listen(port) ]==]);
> file.close();
> dofile("UDPserver.lua");
IP:192.168.1.4, Port:10009
> Command Reveived: 440
Command Reveived: 0
Command Reveived: 220
Command Reveived: 0
```

pwm.stop(5)

Final Thoughts

- NodeMCU is the most powerful WiFi kit everybody can afford.
- If you can turn a LED on and off over WiFi, you can do anything in IoT world!
- Have fun!!!

Any Questions?

Thank You.