



IoT for Fun!

Chen-Hanson Ting

SVFIG

2017 Maker Faire

2017 Maker Faire Workshop



SILICON
VALLEY


FIG



IoT for FUN!
WiFi Workshop

- espForth
- Arduino Server
- MicroPython/WebREPL
- MicroPython Server
- Lua/ESPlorer Server

High	
High Side	
High Side / Low	2.5
High Side / Low	2.5
High Side / Low Side	2.5
High Side / Low Side	2.5
High Side / Low Side	2.5
High Side / Low Side	2.5



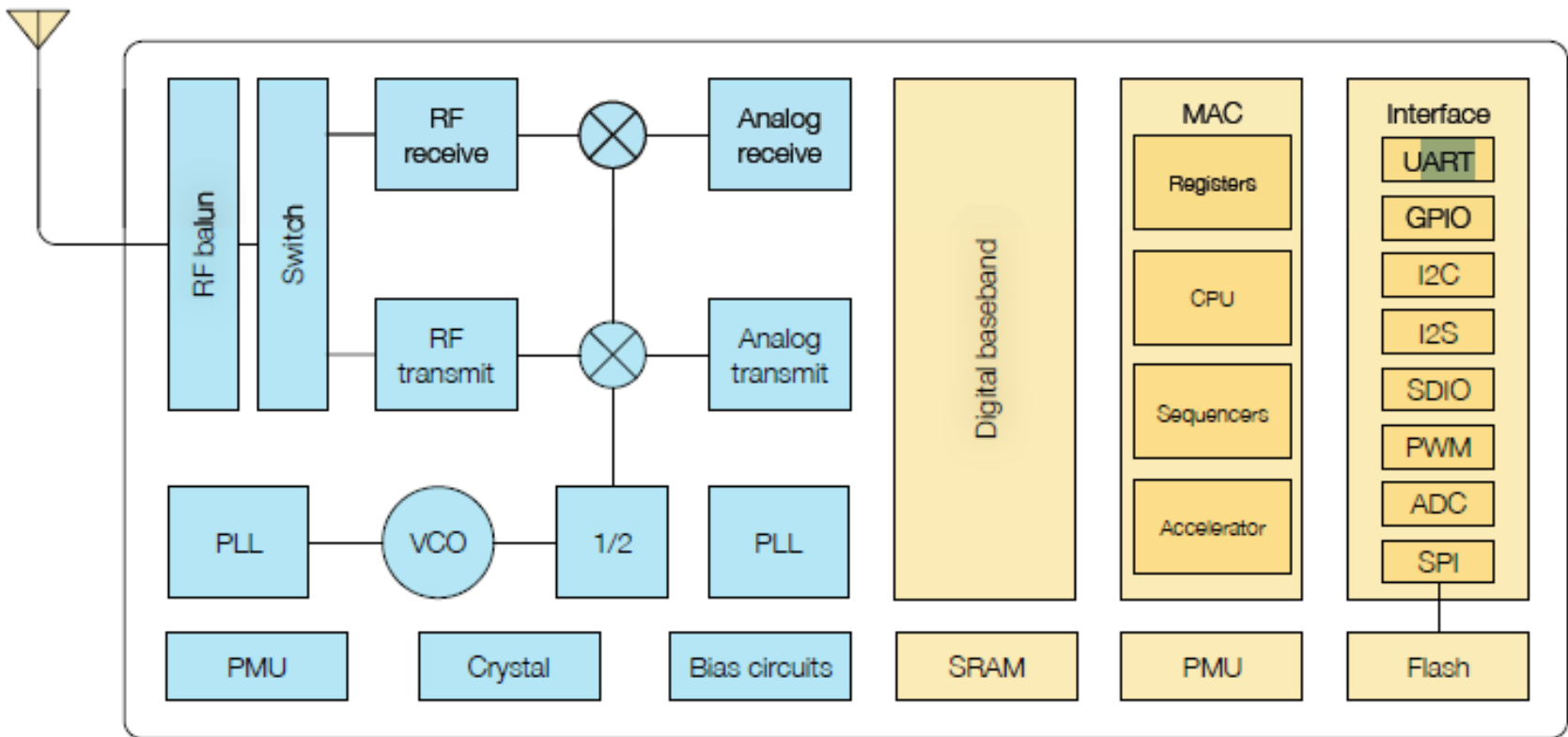
ESP8266

- 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



ESP8266



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.**

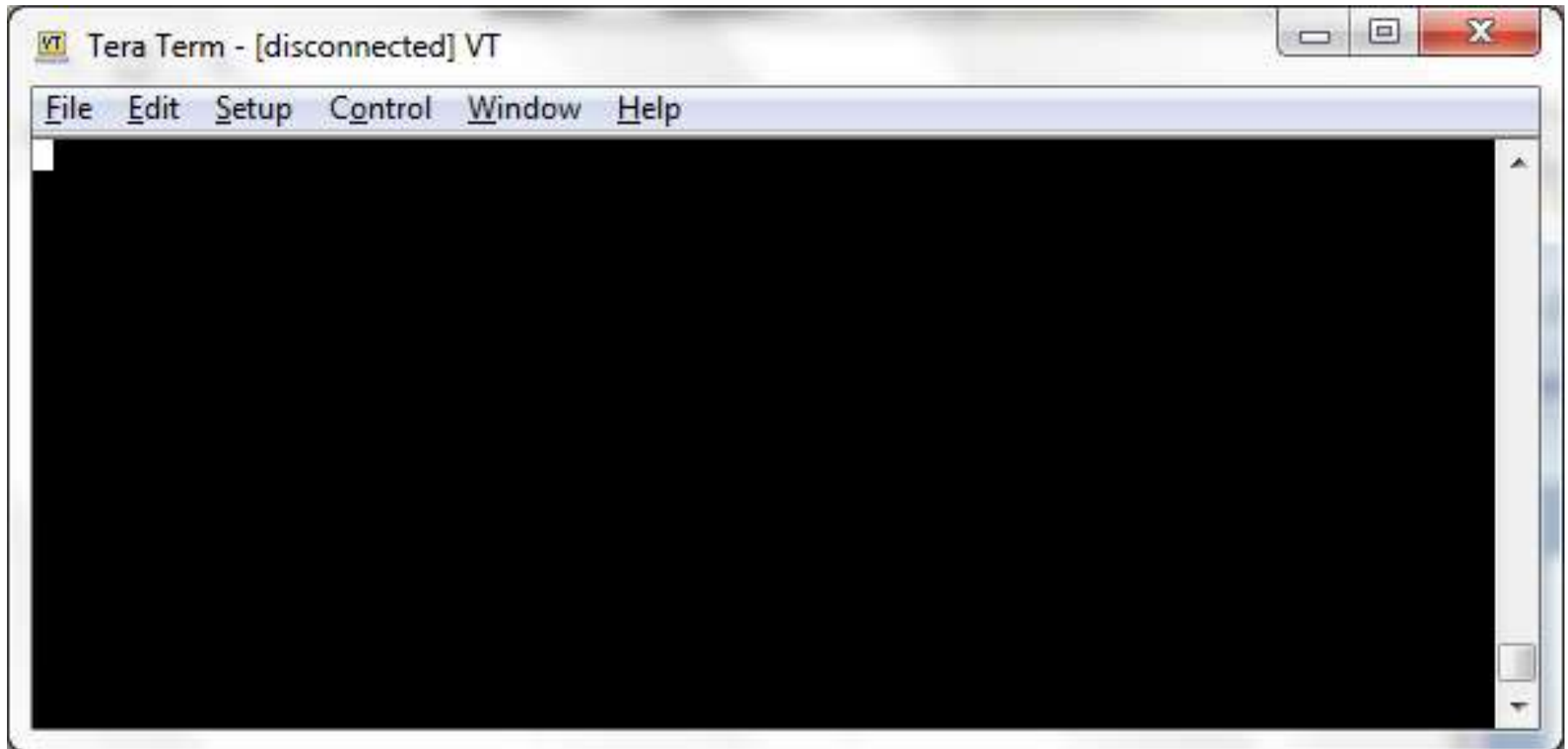


Tools You May Need

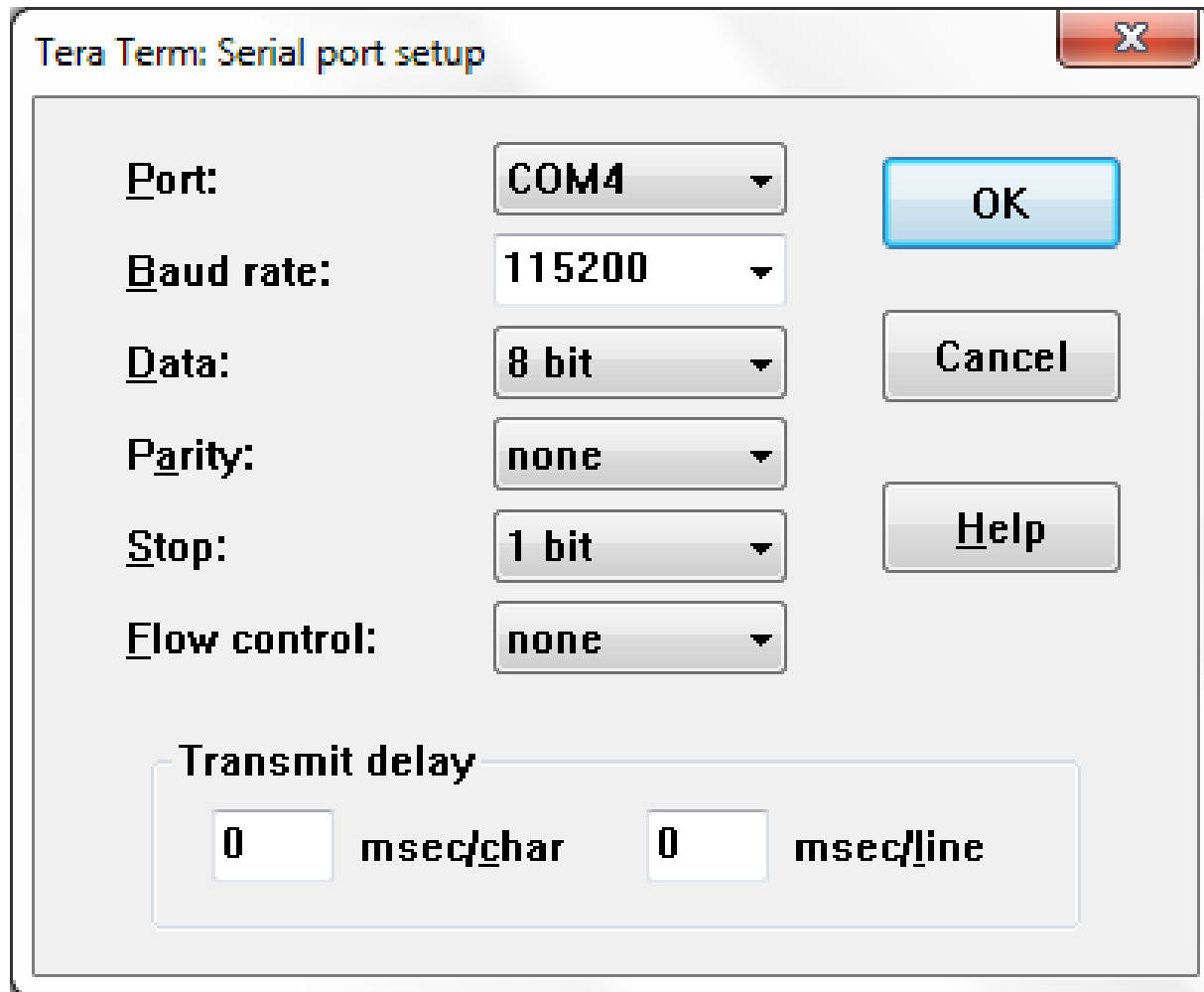
- 1. TeraTerm Terminal Emulator**
- 2. ESP8266Flasher**
- 3. Arduino IDE**
- 4. Hercules UDP Sender**



1. TeraTerm Terminal



1. TeraTerm Terminal



The image shows a screenshot of the 'Tera Term: Serial port setup' dialog box. The dialog has a title bar with a close button (X) in the top right corner. The main area contains several settings, each with a label and a dropdown menu:

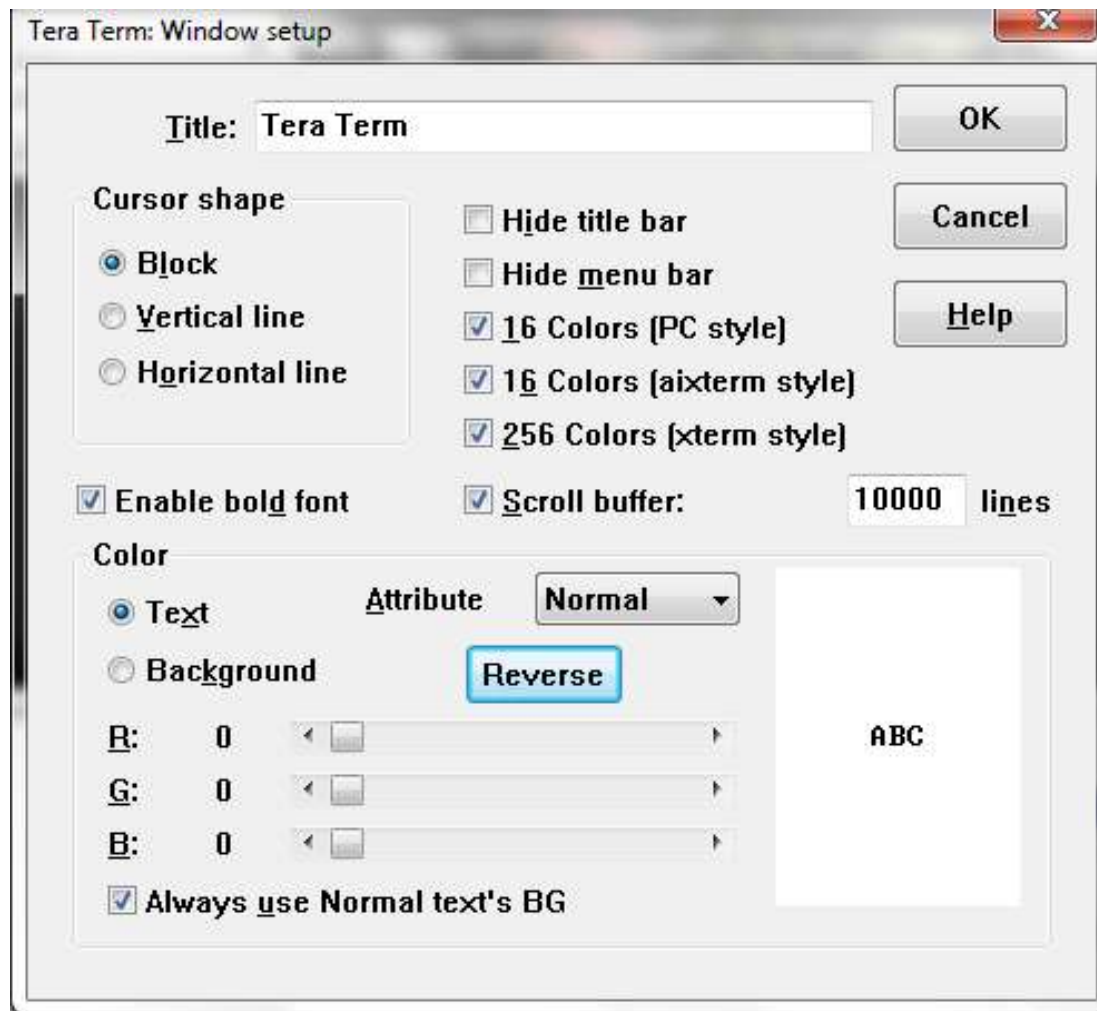
- Port:** COM4
- Baud rate:** 115200
- Data:** 8 bit
- Parity:** none
- Stop:** 1 bit
- Flow control:** none

On the right side of the dialog, there are four buttons: 'OK' (highlighted in blue), 'Cancel', and 'Help' (with an underline under the 'H').

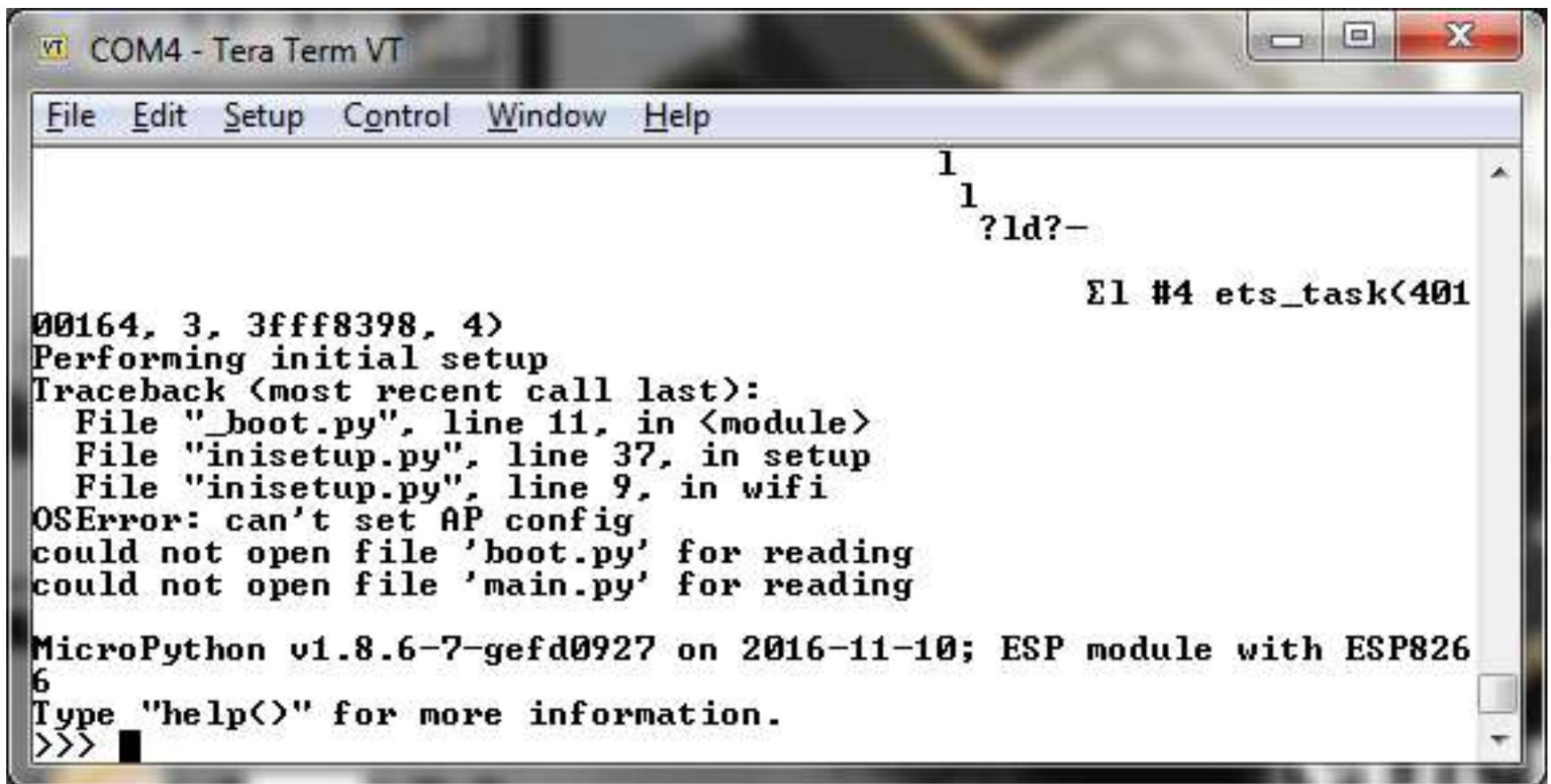
At the bottom, there is a section for 'Transmit delay' with two input fields:

- 0 msec/char
- 0 msec/line

1. TeraTerm Terminal



1. TeraTerm Terminal



The image shows a screenshot of a TeraTerm terminal window. The window title is "COM4 - Tera Term VT". The menu bar includes "File", "Edit", "Setup", "Control", "Window", and "Help". The terminal output shows a MicroPython error during the initial setup of an ESP8266 module. The error message is: "OSError: can't set AP config", followed by "could not open file 'boot.py' for reading" and "could not open file 'main.py' for reading". The terminal also displays the MicroPython version "v1.8.6-7-gefd0927" and the date "2016-11-10". The prompt "Type 'help()' for more information." is shown, followed by the input "6" and the prompt "6 >>>".

```
VT COM4 - Tera Term VT
File Edit Setup Control Window Help
                                     1
                                     1
                                     ?ld?-
                                     Σ1 #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

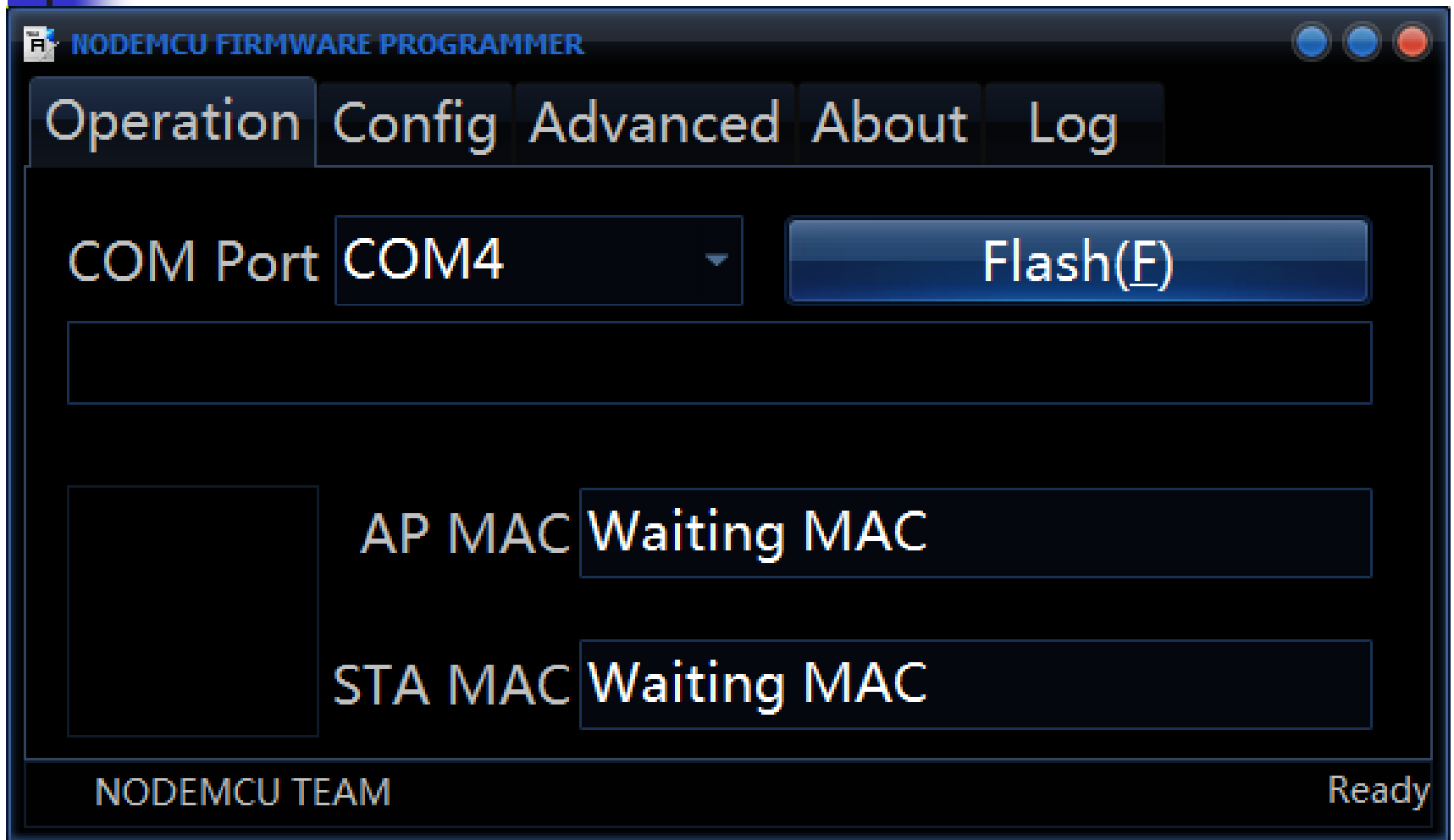
MicroPython v1.8.6-7-gefd0927 on 2016-11-10; ESP module with ESP8266
6
Type "help()" for more information.
6 >>>
```



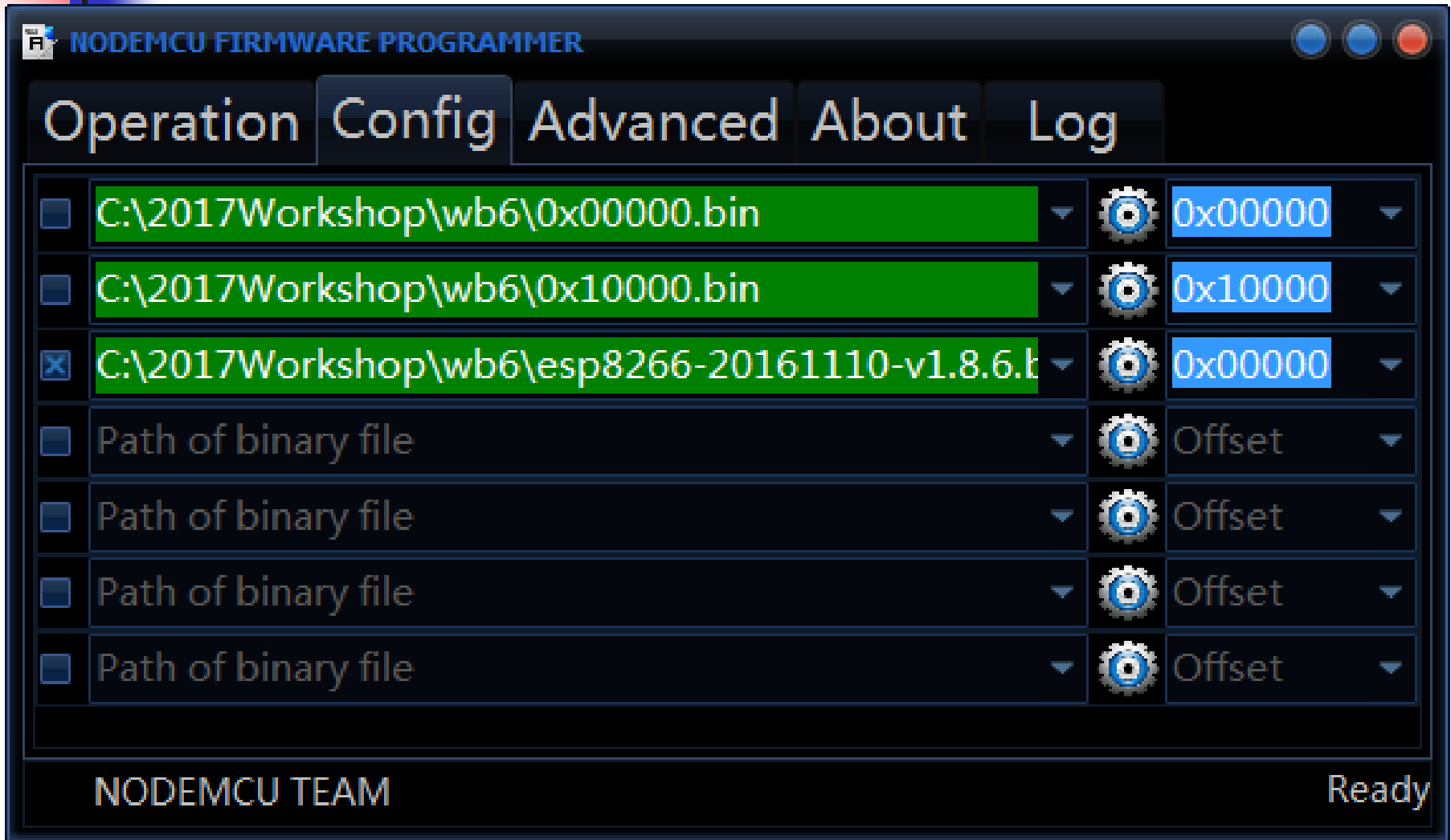
2. ESP8266Flasher

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

3. ESP8266Flasher



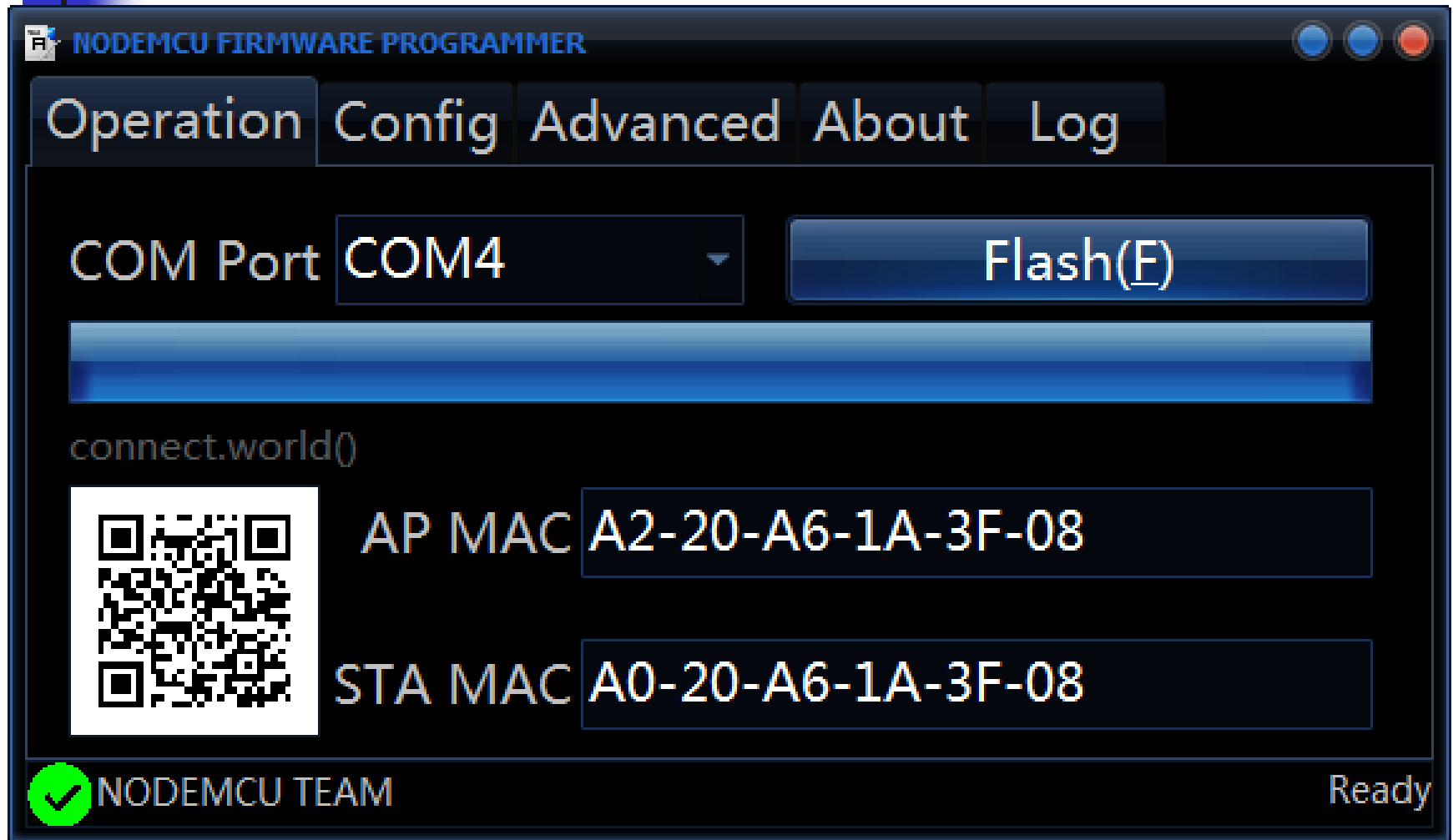
3. ESP8266Flasher



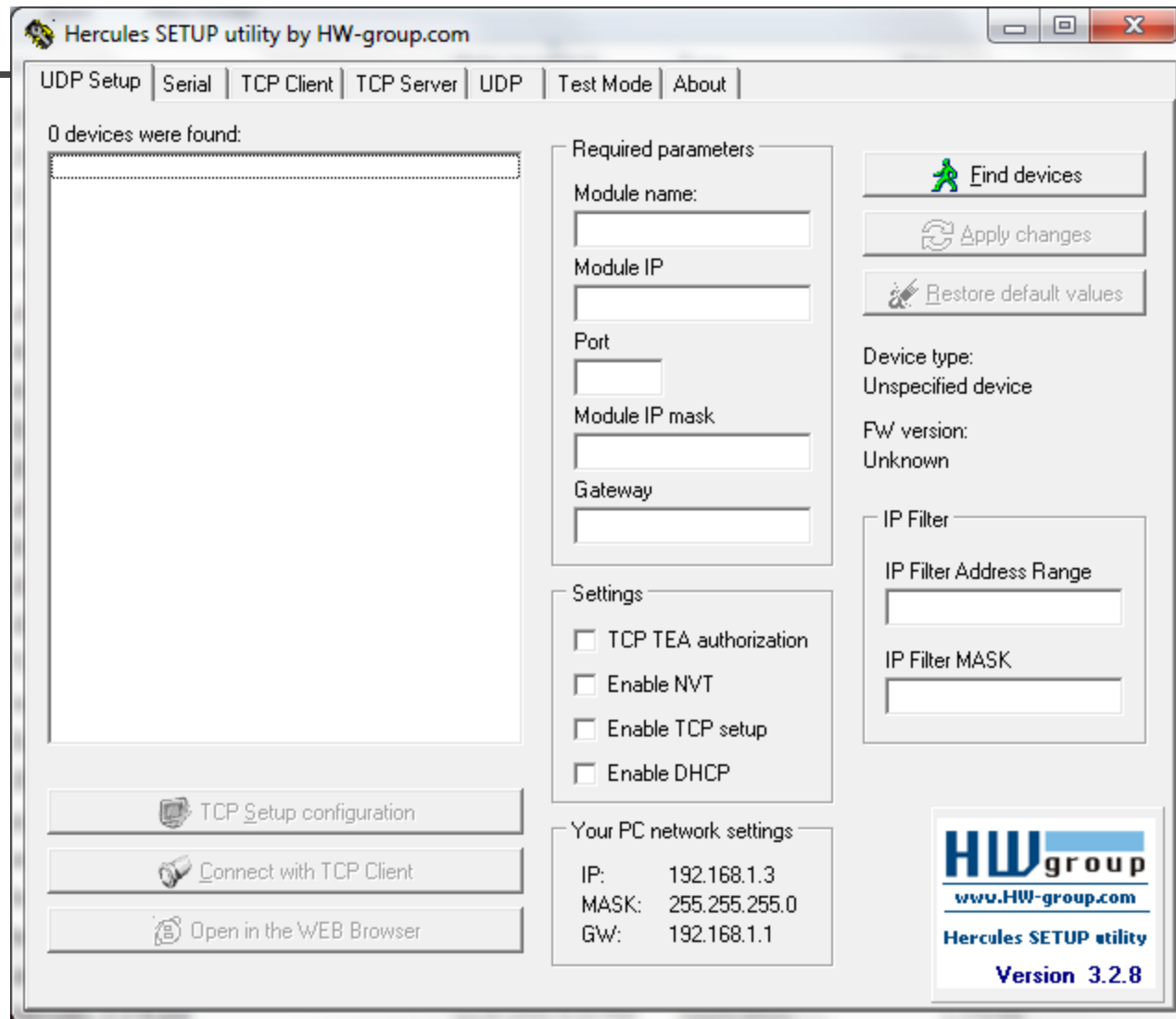
3. ESP8266Flasher



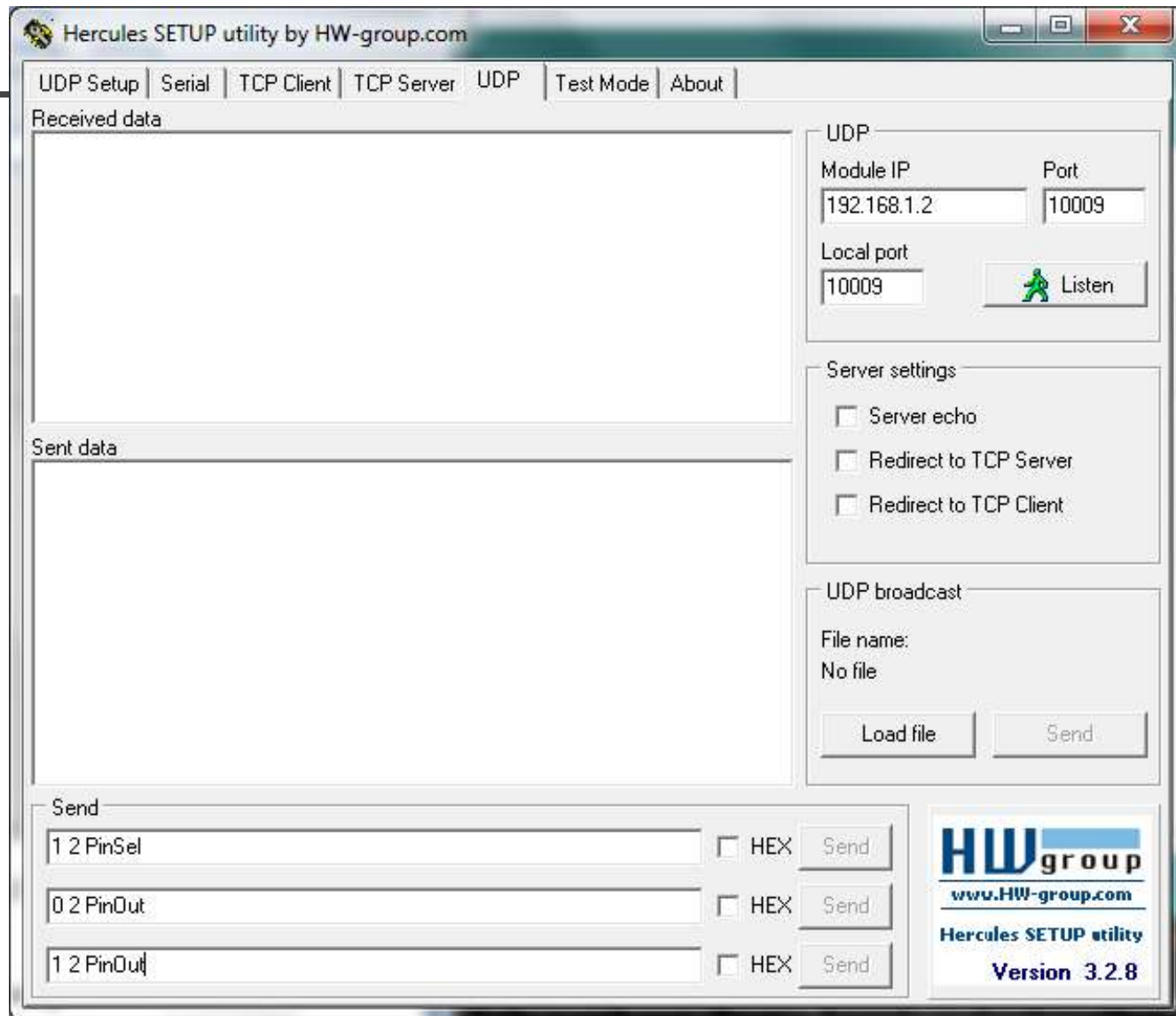
3. ESP8266Flasher



4. Hercules Setup Utility



4. Hercules UDP Sender





Experiments

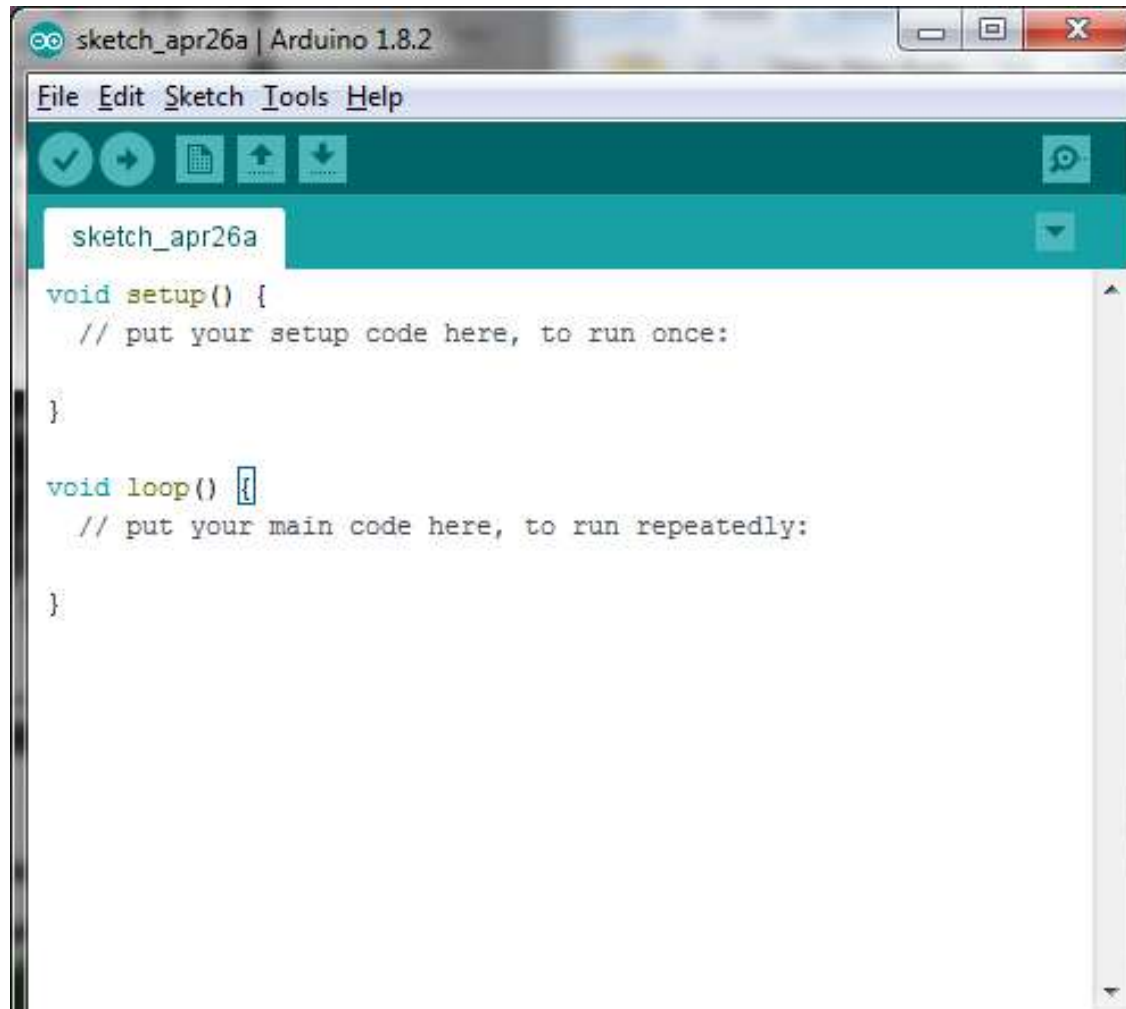
- 1. espForth**
- 2. Arduino UDP Server**
- 3. MicroPython and WebREPL**
- 4. MicroPython UDP Server**
- 5. Lua UDP Server**



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**

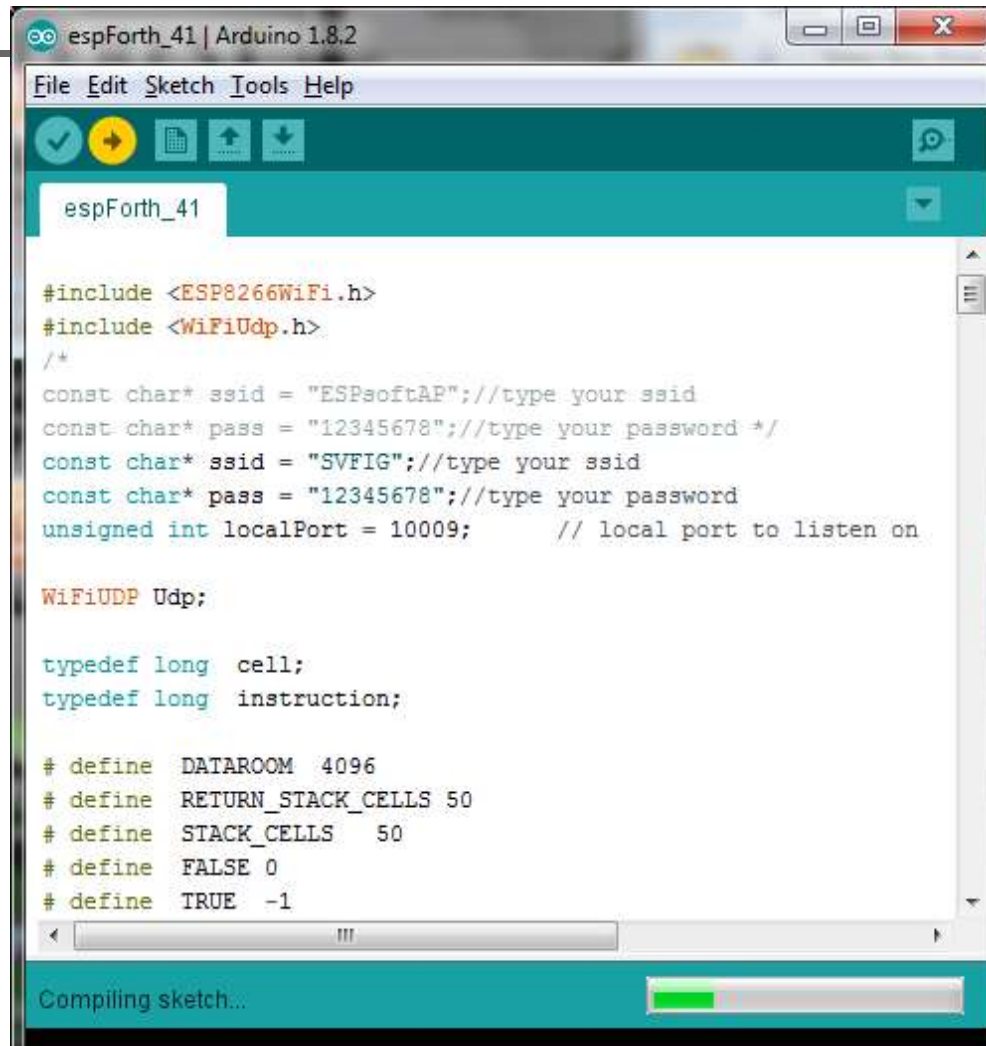
1. Arduino IDE



```
sketch_apr26a | Arduino 1.8.2
File Edit Sketch Tools Help
[Icons: Checkmark, Arrow, File, Up Arrow, Down Arrow, Search]
sketch_apr26a
void setup() {
  // put your setup code here, to run once:
}

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


1. espForth



```
espForth_41 | Arduino 1.8.2
File Edit Sketch Tools Help
espForth_41
#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

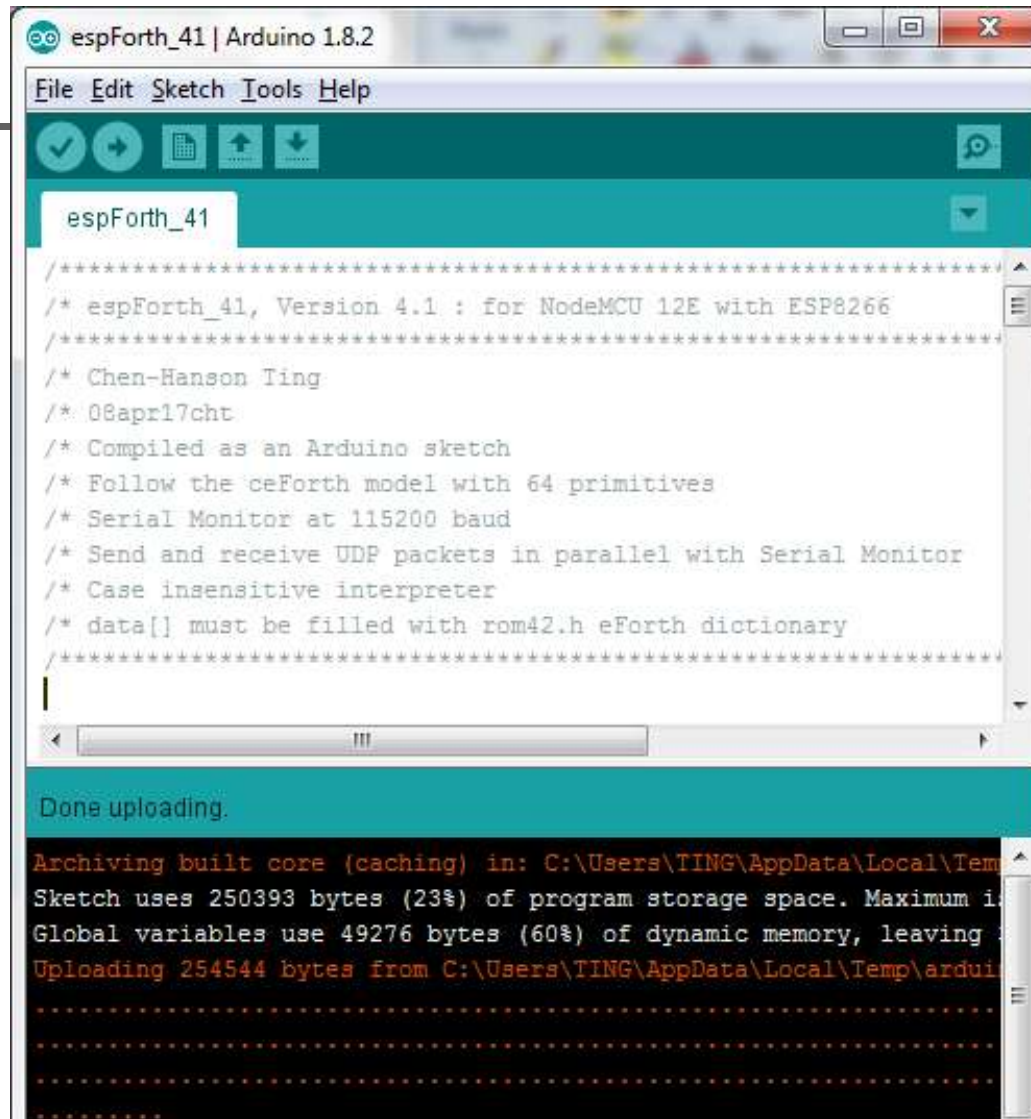
WiFiUDP Udp;

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

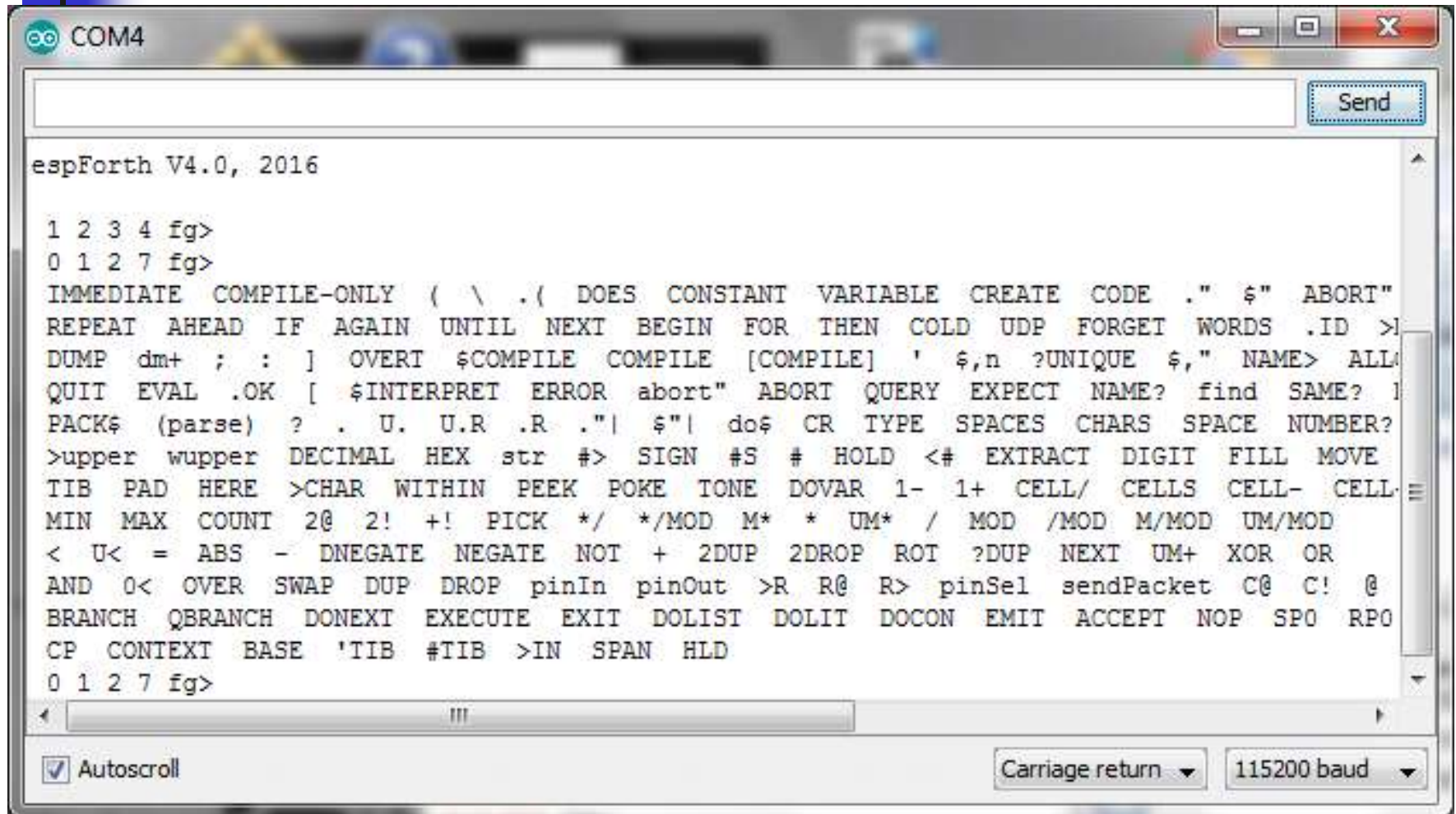


```
espForth_41 | Arduino 1.8.2
File Edit Sketch Tools Help
espForth_41
/*****
/* espForth_41, Version 4.1 : for NodeMCU 12E with ESP8266
/*****
/* Chen-Hanson Ting
/* 08apr17cht
/* Compiled as an Arduino sketch
/* Follow the ceForth model with 64 primitives
/* Serial Monitor at 115200 baud
/* Send and receive UDP packets in parallel with Serial Monitor
/* Case insensitive interpreter
/* data[] must be filled with rom42.h eForth dictionary
/*****
|
Done uploading.
Archiving built core (caching) in: C:\Users\TING\AppData\Local\Temp
Sketch uses 250393 bytes (23%) of program storage space. Maximum is
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. espForth



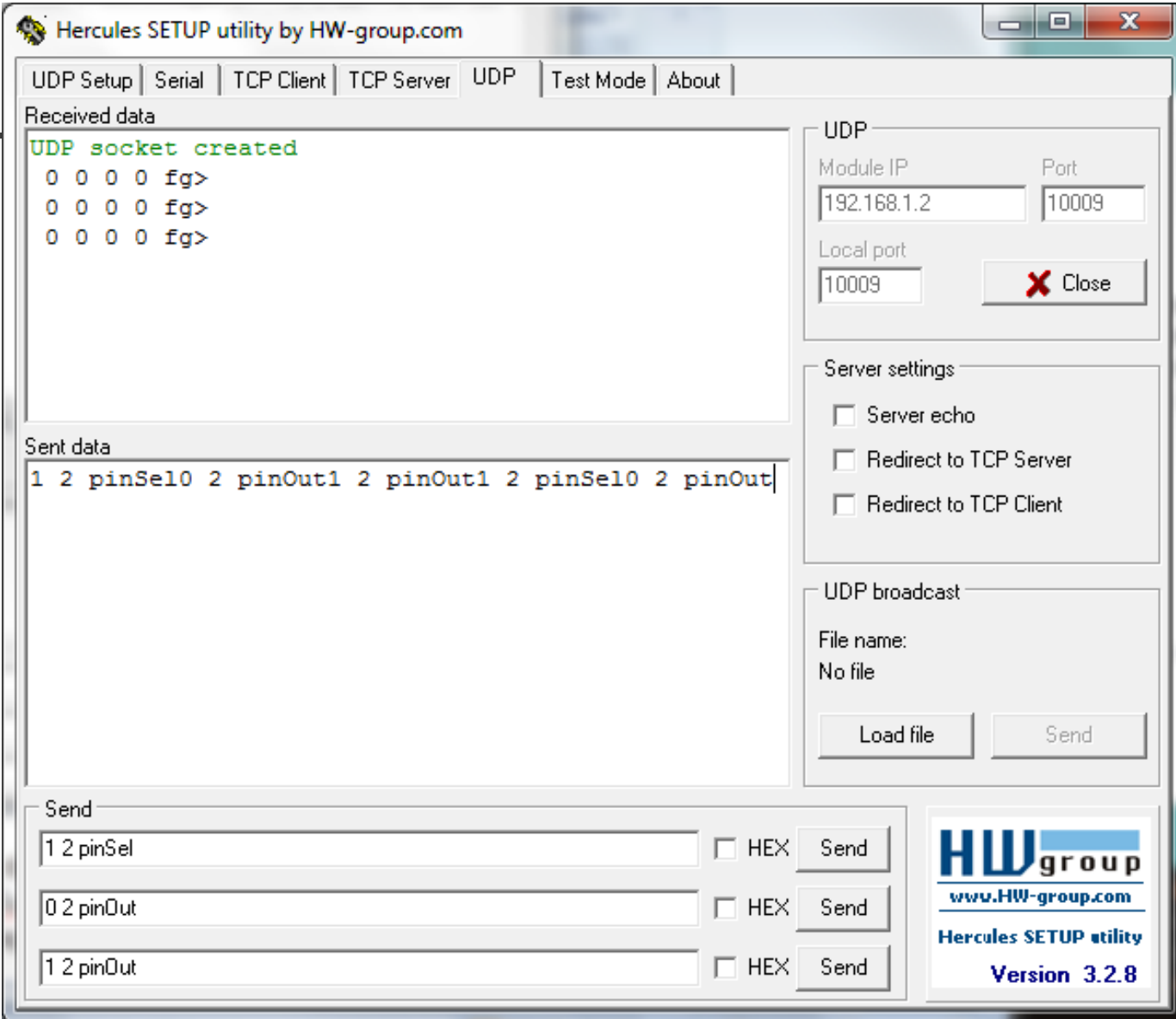
```
COM4

espForth V4.0, 2016

1 2 3 4 fg>
0 1 2 7 fg>
IMMEDIATE COMPILER-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 COMPILER [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 SPO RPO
CP CONTEXT BASE 'TIB #TIB >IN SPAN HLD
0 1 2 7 fg>
```

Autoscroll Carriage return ▼ 115200 baud ▼

1. UDP Packet Sender



The screenshot displays the Hercules SETUP utility interface, specifically the UDP configuration window. The window title is "Hercules SETUP utility by HW-group.com". The interface includes a menu bar with options: UDP Setup, Serial, TCP Client, TCP Server, UDP, Test Mode, and About. The main area is divided into several sections:

- Received data:** A text area showing the message "UDP socket created" in green, followed by three lines of data: "0 0 0 0 fg>".
- Sent data:** A text area showing the command "1 2 pinSel 0 2 pinOut 1 2 pinOut 1 2 pinSel 0 2 pinOut".
- UDP Configuration:** Fields for "Module IP" (192.168.1.2) and "Port" (10009). A "Local port" field is also set to 10009. A "Close" button with a red 'X' icon is present.
- Server settings:** Three checkboxes: "Server echo", "Redirect to TCP Server", and "Redirect to TCP Client", all of which are currently unchecked.
- UDP broadcast:** A "File name:" field containing "No file". "Load file" and "Send" buttons are located below this section.
- Send Section:** Three input fields for sending data, each with a "HEX" checkbox and a "Send" button. The first field contains "1 2 pinSel", the second "0 2 pinOut", and the third "1 2 pinOut".

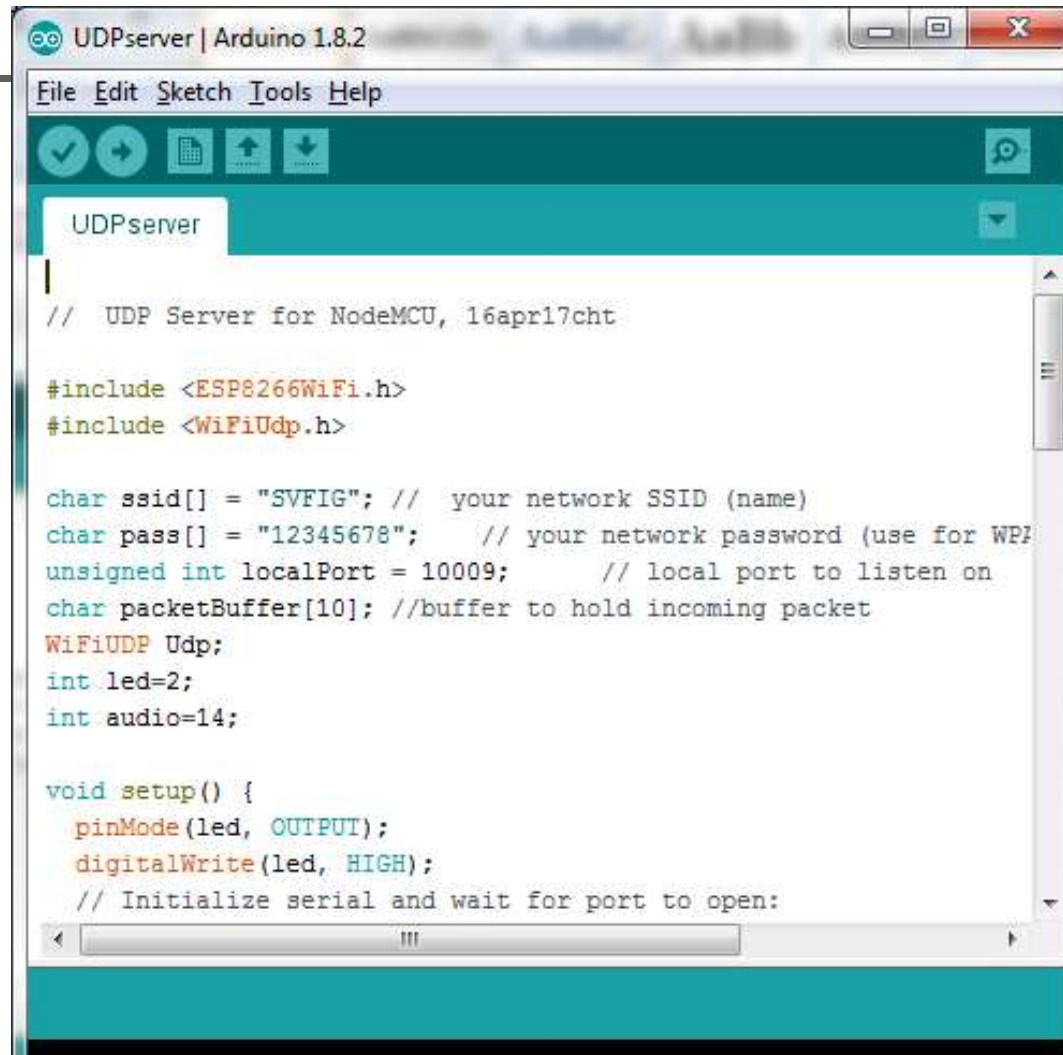
In the bottom right corner, the HW group logo is displayed along with the website "www.HW-group.com", the text "Hercules SETUP utility", and the version number "Version 3.2.8".



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



The image shows a screenshot of the Arduino IDE interface. The window title is "UDPserver | Arduino 1.8.2". The menu bar includes "File", "Edit", "Sketch", "Tools", and "Help". The toolbar contains icons for a checkmark, a right arrow, a document, an upload arrow, a download arrow, and a search icon. The sketch name is "UDPserver". The code in the editor is as follows:

```
// UDP Server for NodeMCU, 16apri17cht

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

char ssid[] = "SVFIG"; // your network SSID (name)
char pass[] = "12345678"; // your network password (use for WPA2)
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:
```

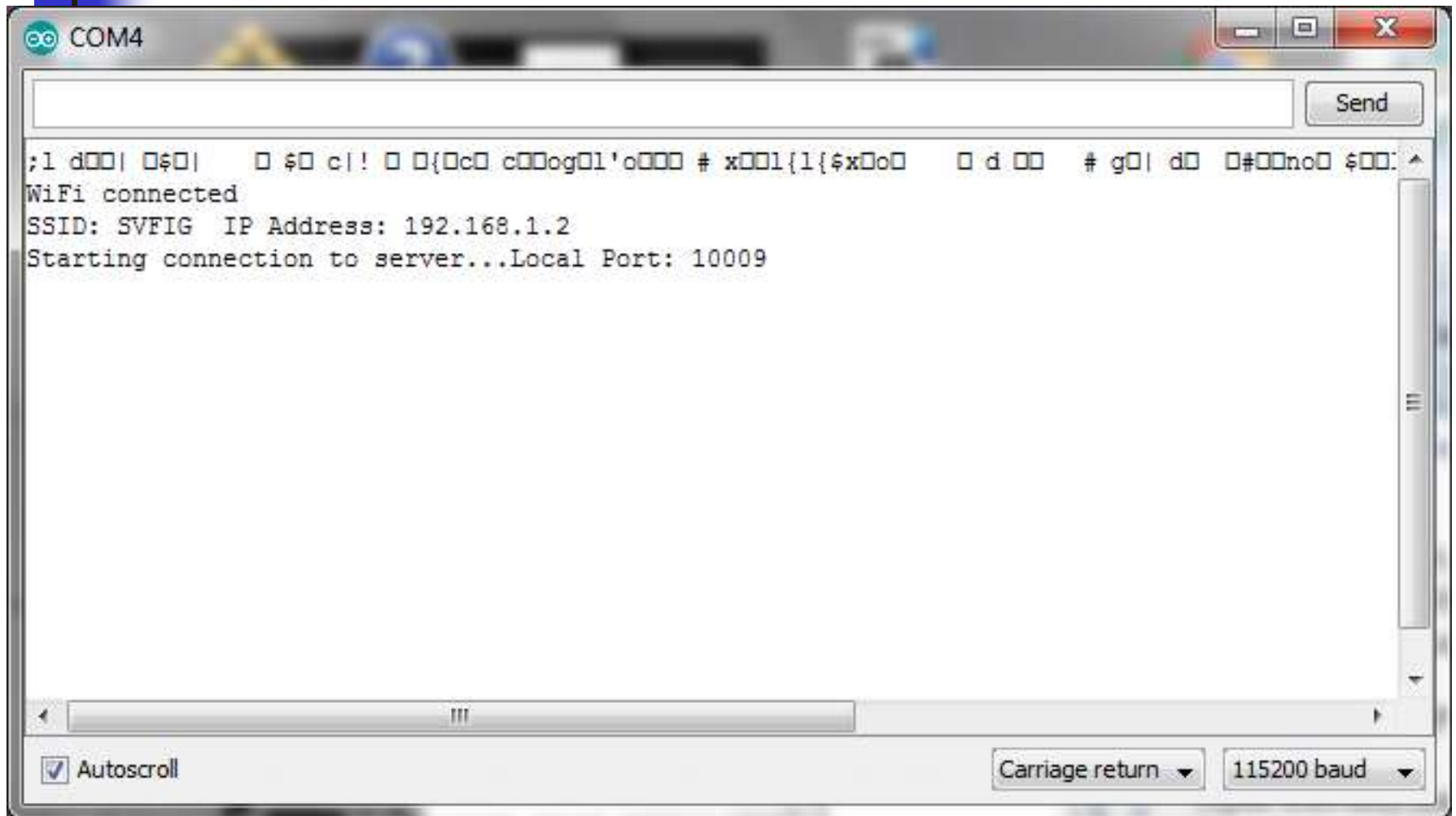


2. Arduino UDP Server

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% ]
```


2. IP on Serial Monitor



The screenshot shows a serial monitor window titled "COM4". The window contains a text area with the following output:

```
;1 d00| 0$0|  0 $0 c|! 0 0{0c0 c00og01'o000 # x001{1{$x0o0  0 d 00  # g0| d0 0#00no0 $00: ^  
WiFi connected  
SSID: SVFIG  IP Address: 192.168.1.2  
Starting connection to server...Local Port: 10009
```

At the bottom of the window, there is a status bar with the following controls:

- Autoscroll
- Carriage return ▾
- 115200 baud ▾

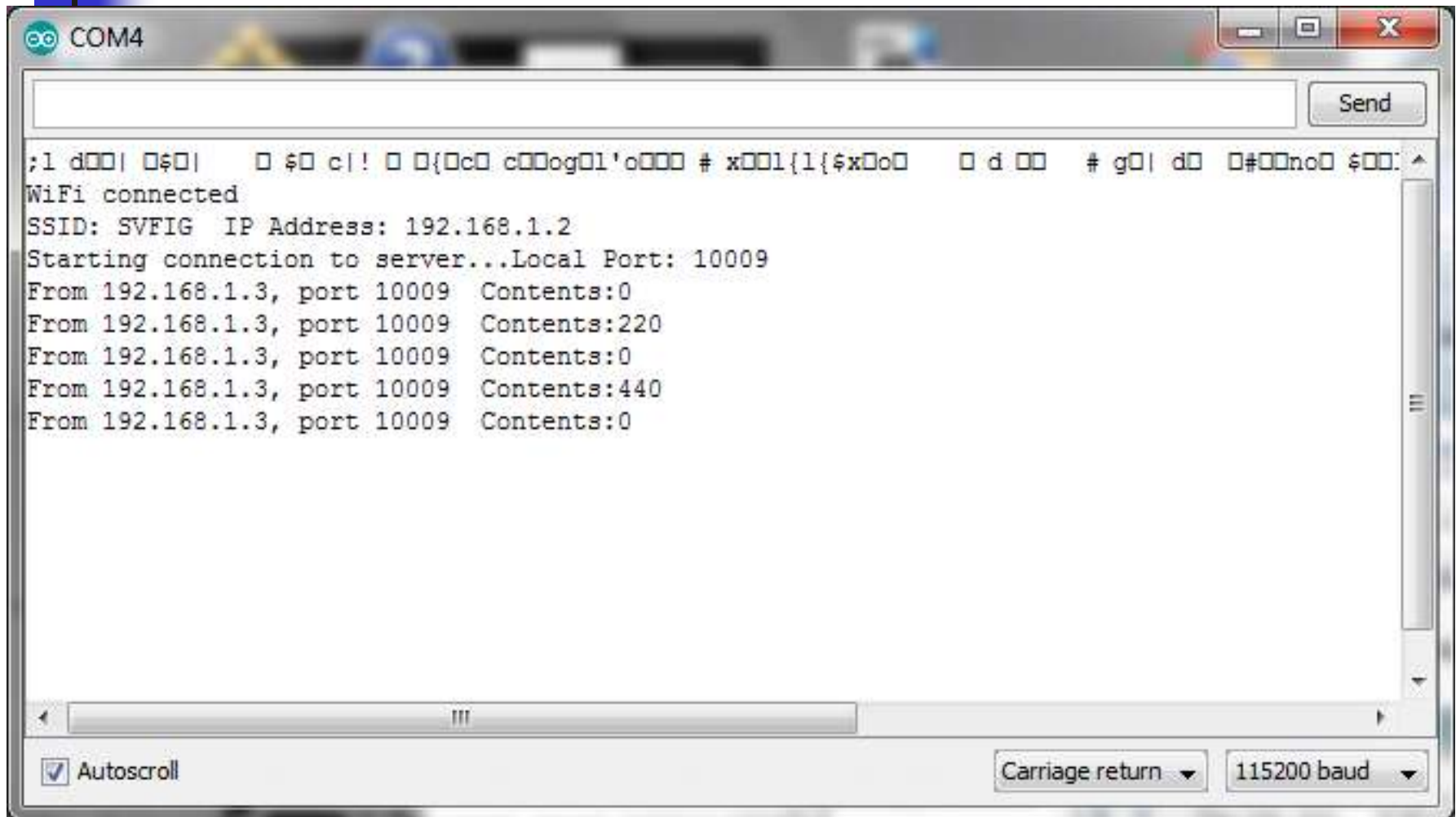
2. Hercules UDP Sender

The screenshot displays the Hercules SETUP utility window, titled "Hercules SETUP utility by HW-group.com". The interface includes a menu bar with options: UDP Setup, Serial, TCP Client, TCP Server, UDP, Test Mode, and About. The main area is divided into several sections:

- Received data:** A text area showing the message "UDP socket created".
- Sent data:** A text area showing the hexadecimal value "440022004400".
- UDP Configuration:** Fields for "Module IP" (192.168.1.2) and "Port" (10009). A "Local port" field is also set to 10009, with a "Close" button next to it.
- Server settings:** Three checkboxes: "Server echo", "Redirect to TCP Server", and "Redirect to TCP Client", all of which are currently unchecked.
- UDP broadcast:** A "File name:" field containing "No file", with "Load file" and "Send" buttons below it.
- Send Section:** Three input fields for data to be sent, each with a "HEX" checkbox and a "Send" button. The first field contains "440", the second "0", and the third "220".

In the bottom right corner, the HWgroup logo is displayed along with the website "www.HW-group.com", the text "Hercules SETUP utility", and the version number "Version 3.2.8".

2. Serial Monitor Logging



The screenshot shows a serial monitor window titled "COM4" with a "Send" button in the top right. The main area contains the following text:

```
;1 d00| 0$0|  0 $0 c|! 0 0{0c0 c00og0l'o000 # x001{1{$x0o0  0 d 00 # g0| d0 0#00no0 $00: ^
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
```

At the bottom of the window, there is a scroll bar, a checked "Autoscroll" checkbox, a "Carriage return" dropdown menu, and a "115200 baud" dropdown menu.



3. MicroPython/WebREPL

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

3. MicroPython REPL

VT COM4 - Tera Term VT

File Edit Setup Control Window Help

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)
```

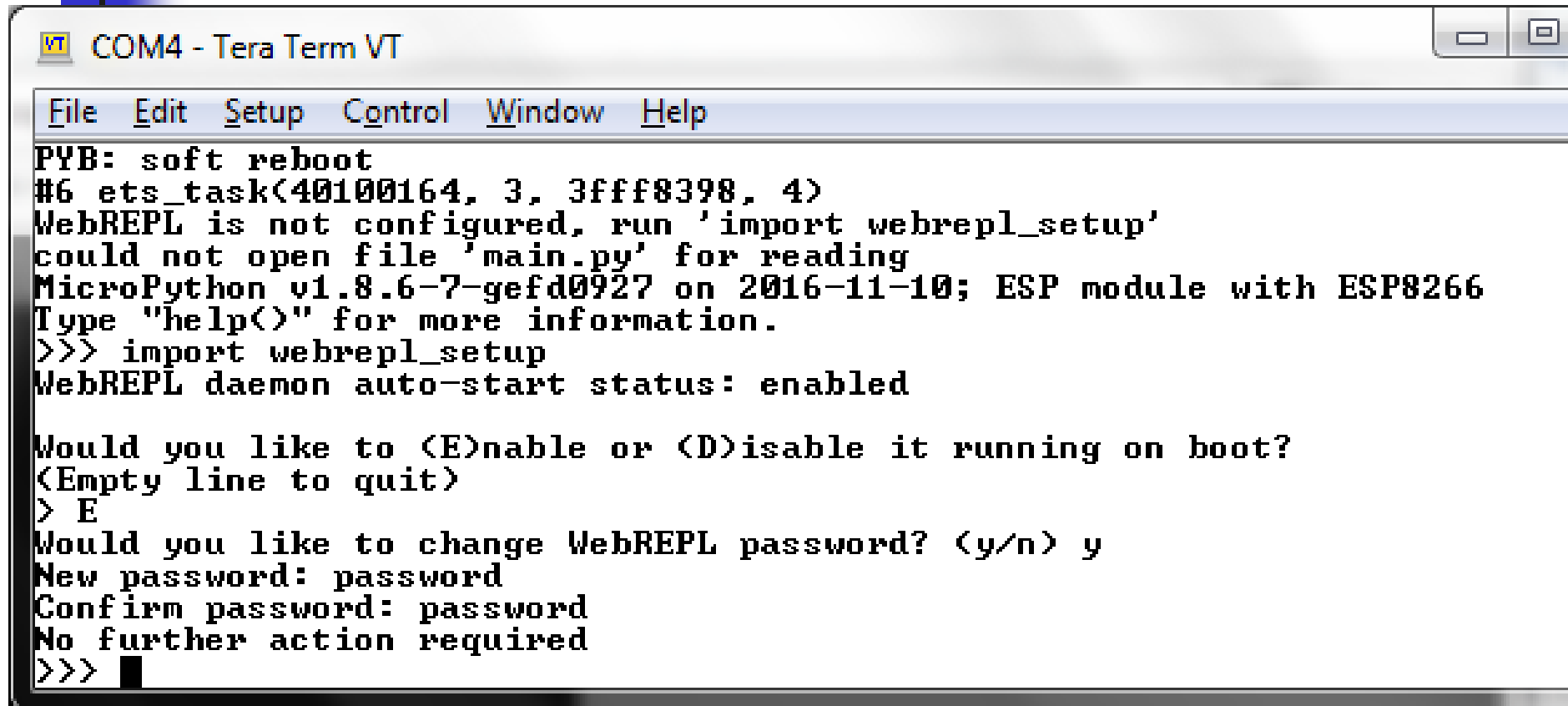
```
>>> p2.high()
```

```
>>> p2.low()
```

```
>>> p2.high()
```

```
>>>
```

3. Install WebREPL



```
COM4 - Tera Term VT
File Edit Setup Control Window Help
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

ws://192.168.4.1:8266/ Disconnect

```
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()  
>>> |
```

Send a file
Choose File No file chosen
Send to device

Get a file
Get from device

(file operation status)

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

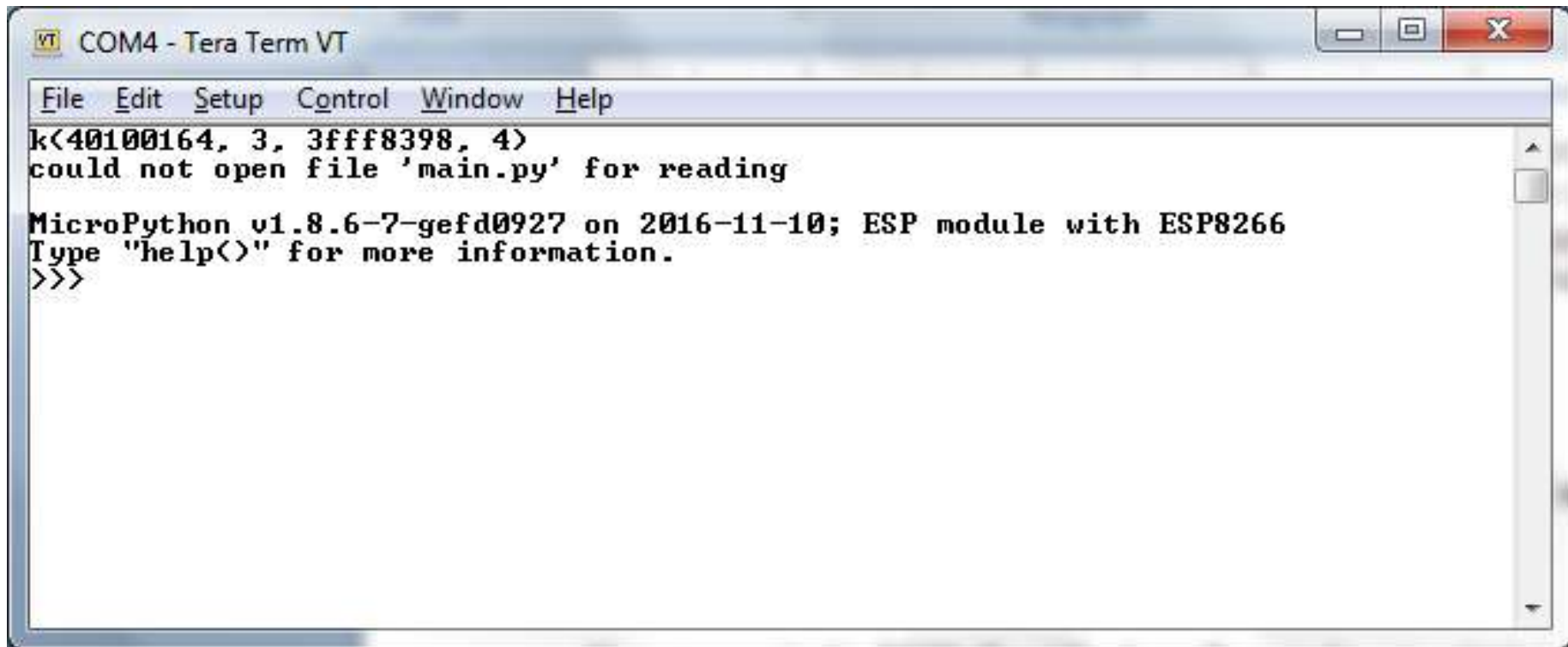


4. MicroPython UDP Server

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



4. MicroPython REPL



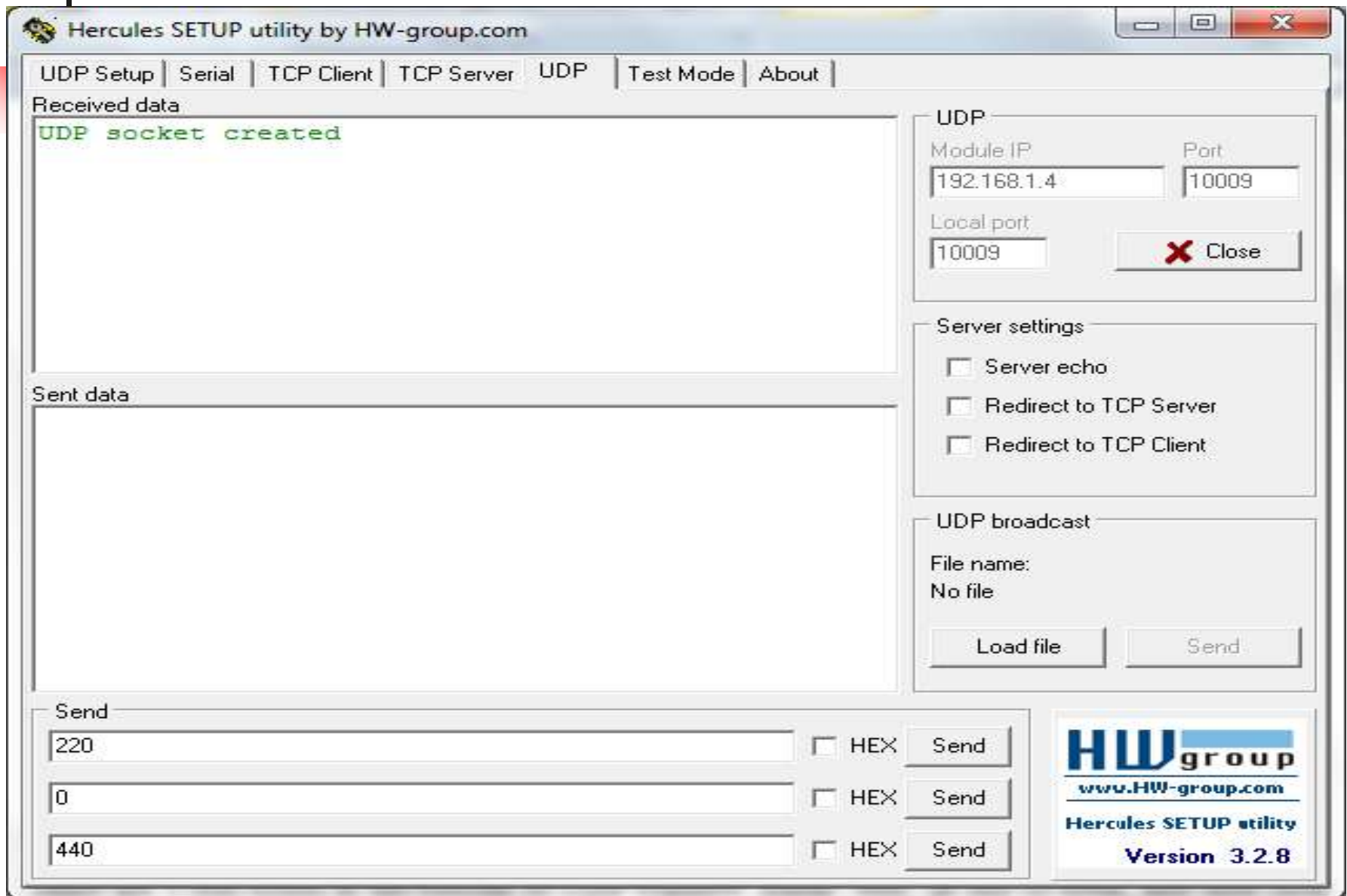
```
VT COM4 - Tera Term VT
File Edit Setup Control Window Help
k<40100164, 3, 3fff8398, 4>
could not open file 'main.py' for reading

MicroPython v1.8.6-7-gefdd0927 on 2016-11-10; ESP module with ESP8266
Type "help(<)" for more information.
>>>
```

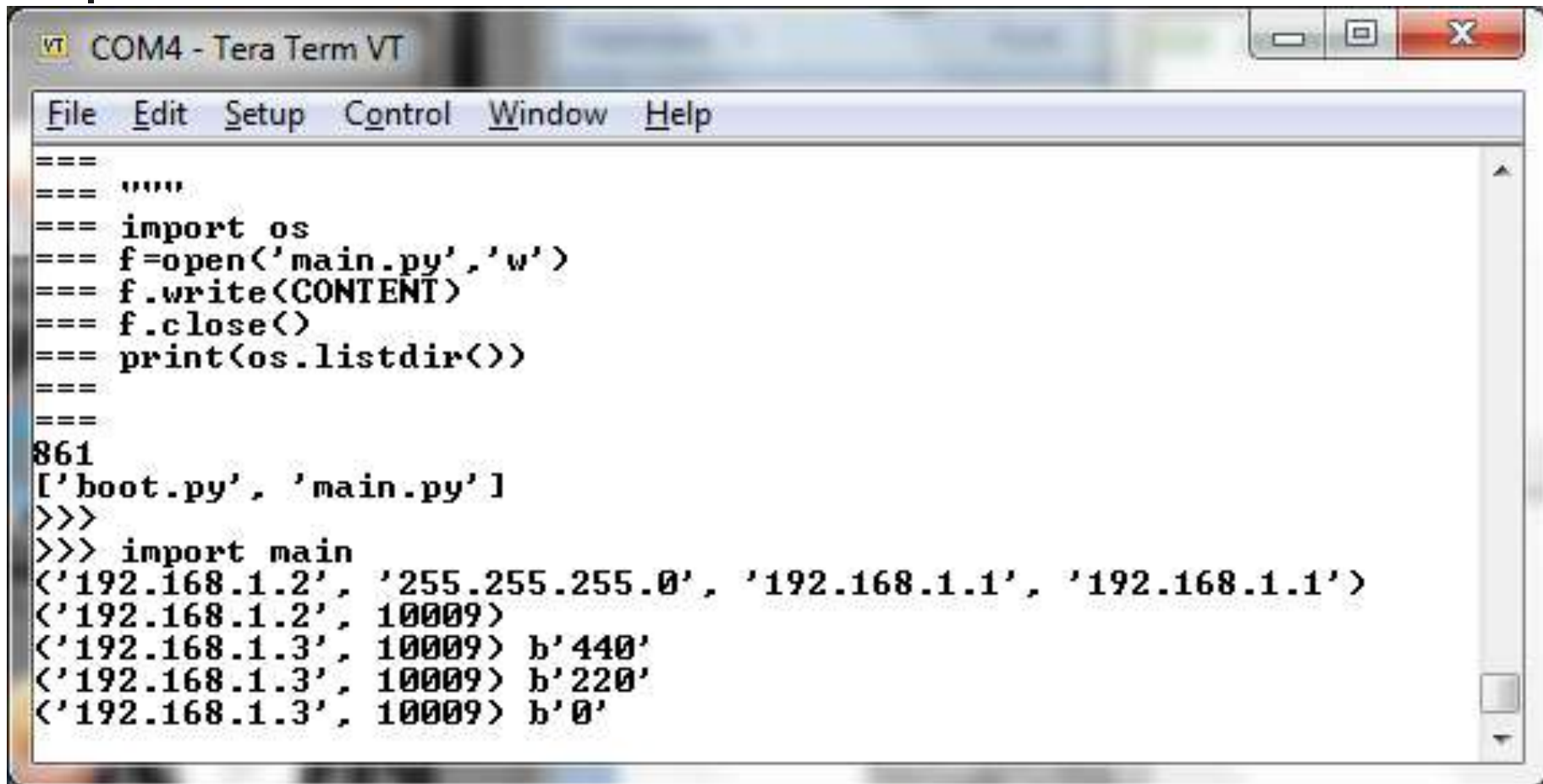
4. Load WriteFile.py

```
VT COM4 - Tera Term VT
File Edit Setup Control Window Help
=== 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('SUFIG','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



4. UDP Packet Log



```
VT COM4 - Tera Term VT
File Edit Setup Control Window Help
====
==== *****
==== 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

The image shows the ESP8266 IDE (ESPlorer) interface. The main window displays a Lua script named 'UDPserver.lua' with the following code:

```
1 led=4
2 audio=5
3 port=10009
4 print("IP: "..wifi.sta.getip()..", Port: "..port)
5 gpio.mode(led, gpio.OUTPUT)
6 gpio.write(led, gpio.LOW)
7 pwm.setup(audio,440,512)
8 srv=net.createServer(net.UDP)
9 srv:on("receive", function(srv, pl)
10     n = tonumber(pl)
11     print("Command Received: "..n)
12     if n==0 then
13         gpio.write(led, gpio.HIGH)
14         pwm.stop(audio)
15     else
16         gpio.write(led, gpio.LOW)
17         pwm.setup(audio,n,512)
18         pwm.start(audio)
19     end
```

The status bar at the bottom indicates the device is in 'IDLE' mode. The right-hand side of the interface shows the serial terminal output for COM4, with a baud rate of 115200. The output text is as follows:

```
PORT OPEN 115200
Communication with MCU..Got answer! Communi
AutoDetect firmware...
Can't autodetect firmware, because proper a
Please, reset module or continue.
,, cÄö'oß$g'æää c piçdsdsdpûgà f l c r
{$@,ÿ
WiFiBoy-NodeMCU 1.5.1-wb5a build 2016-7-23
>
```

Below the terminal, there are buttons for 'Format', 'FS Info', and 'Reload'. At the bottom of the IDE, there are buttons for 'Save&Run', 'Save&Compile', 'Save&Compile&Run...', 'Save', 'Save&Compile All', 'View on ESP', 'View on ESP', 'Save&...', 'Save to ESP', 'Send to ESP', 'Run', and 'Upload'.

5. Compile UDPserver

The screenshot displays the ESPLorer v0.2.0-rc5 IDE interface. The main window is titled "UDPserver.lua" and contains the following Lua code:

```
3 port=10009
4 print("IP: "..wifi.sta.getip()..", Port:"..port)
5 gpio.mode(led, gpio.OUTPUT)
6 gpio.write(led, gpio.LOW)
7 pwm.setup(audio,440,512)
8 srv=net.createServer(net.UDP)
9 srv:on("receive", function(srv, pl)
10     n = tonumber(pl)
11     print("Command Reveived: "..n)
12     if n==0 then
13         gpio.write(led, gpio.HIGH)
14         pwm.stop(audio)
15     else
16         gpio.write(led, gpio.LOW)
17         pwm.setup(audio,n,512)
18         pwm.start(audio)
19     end
20 end)
21 srv:listen(port)
22
```

The terminal window on the right shows the output of the script:

```
> 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
>
```

The interface also features a menu bar (File, Edit, ESP, View, Links?), a toolbar with buttons for Open, Reload, Save, Save..., Close, Undo, Redo, and Cut, and a status bar at the bottom showing "IDLE" and the file path "C:\2017Workshop\ESPLorer6\UDPserver.lua". The bottom right corner includes a "Send" button and a terminal input field containing "pwm.stop(5)".

5. Send UDP Packets

The screenshot shows the Hercules SETUP utility interface. The window title is "Hercules SETUP utility by HW-group.com". The "UDP" tab is selected in the menu bar. The "Received data" pane shows the message "UDP socket created". The "Sent data" pane is empty. The "UDP" configuration section on the right includes fields for "Module IP" (192.168.1.4) and "Port" (10009), and a "Local port" field (10009) with a "Close" button. The "Server settings" section has three unchecked checkboxes: "Server echo", "Redirect to TCP Server", and "Redirect to TCP Client". The "UDP broadcast" section has a "File name" field with "No file" and "Load file" and "Send" buttons. At the bottom, the "Send" section has three input fields with values "220", "0", and "440", each with a "HEX" checkbox and a "Send" button. The HW group logo and version "Hercules SETUP utility Version 3.2.8" are in the bottom right corner.

Hercules SETUP utility by HW-group.com

UDP Setup | Serial | TCP Client | TCP Server | **UDP** | Test Mode | About

Received data

UDP socket created

Sent data

UDP

Module IP: 192.168.1.4 Port: 10009

Local port: 10009 [Close]

Server settings

- Server echo
- Redirect to TCP Server
- Redirect to TCP Client

UDP broadcast

File name: No file

[Load file] [Send]

Send

220 [HEX] [Send]

0 [HEX] [Send]

440 [HEX] [Send]

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

5. UDP Packet Log

The screenshot displays the ESPlorer v0.2.0-rc5 IDE interface. The left pane shows a Lua script named 'UDPserver.lua' with the following code:

```
3 port=10009
4 print("IP: "..wifi.sta.getip()..", Port: "..port)
5 gpio.mode(led, gpio.OUTPUT)
6 gpio.write(led, gpio.LOW)
7 pwm.setup(audio,440,512)
8 srv=net.createServer(net.UDP)
9 srv:on("receive", function(srv, pl)
10     n = tonumber(pl)
11     print("Command Received: "..n)
12     if n==0 then
13         gpio.write(led, gpio.HIGH)
14         pwm.stop(audio)
15     else
16         gpio.write(led, gpio.LOW)
17         pwm.setup(audio,n,512)
18         pwm.start(audio)
19     end
20 end)
21 srv:listen(port)
22
```

The right pane shows the terminal output for the script:

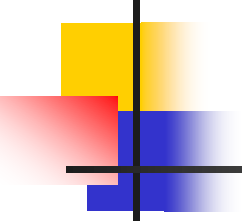
```
> w([=[ else ]==]);
> w([=[     gpio.write(led, gpio.LOW)]==]);
> w([=[     pwm.setup(audio,n,512)]==]);w([=[
> w([=[end]]==]);
> w([=[srv:listen(port) ]==]);
> file.close();
> dofile("UDPserver.lua");
IP:192.168.1.4, Port:10009
> Command Received: 440
Command Received: 0
Command Received: 220
Command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

The IDE interface includes a menu bar (File, Edit, ESP, View, Links?), a toolbar with options like 'Scripts', 'Commands', 'Snippets', and 'Settings', and a status bar at the bottom showing 'IDLE' and the file path 'C:\2017Workshop\ESPlorer6\UDPserver.lua'. The terminal window has a 'Send' button and a dropdown menu for input.

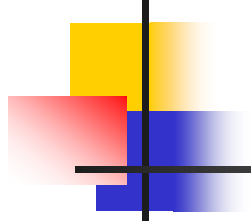


Final Thoughts

- **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.