MK14ESP8266keys

A Web server to program the MK14 computer

This provides an ESP8266 based web server which can “program” the MK14 using “hex” files loaded into the ESP8266 file store, and it’s all done on the web.

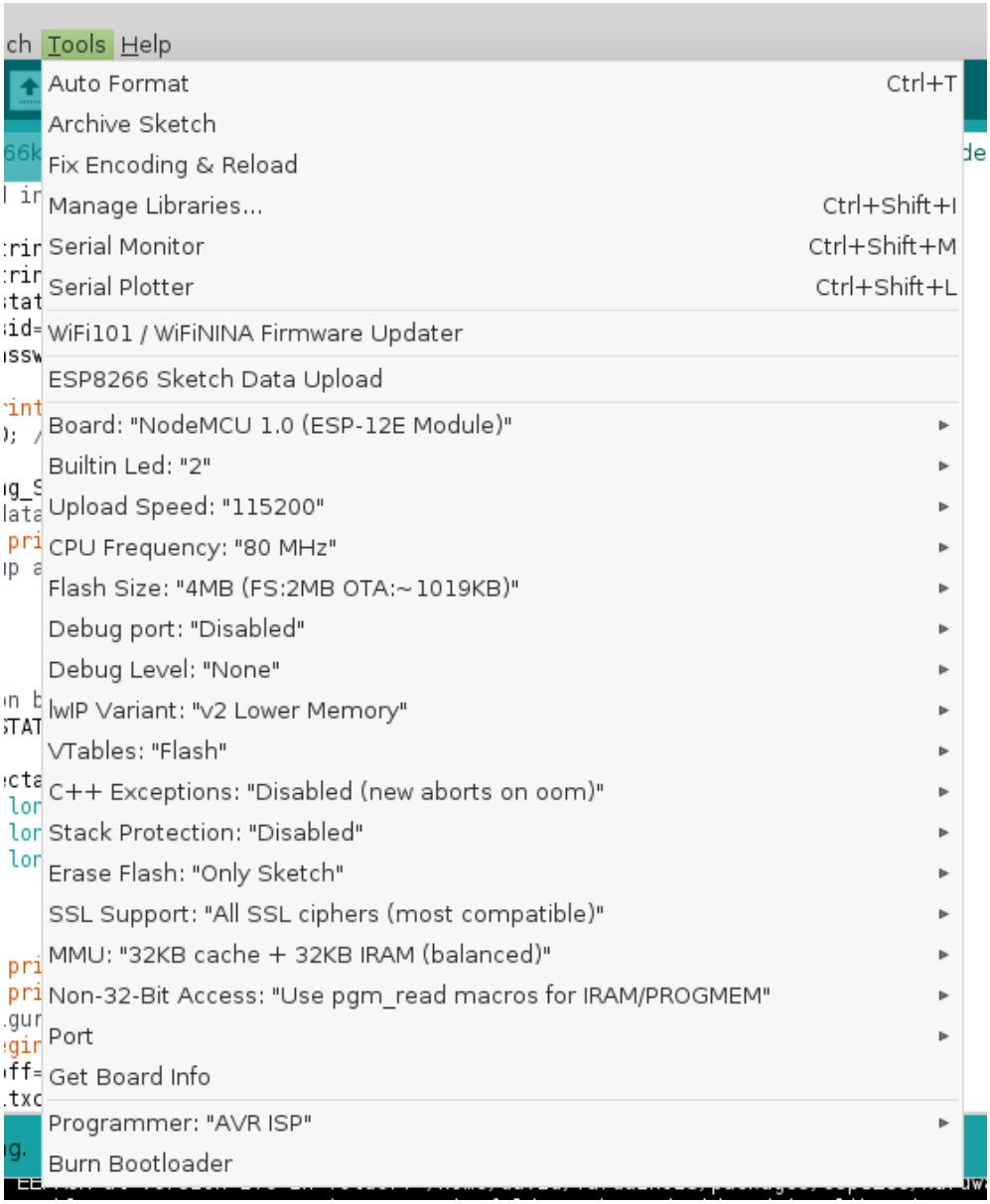
Setup

Having built the mk14keyscp board you will need to program the ESP8266 module.

The main sketch file, MK14ESP8266keys.ino, must be in a folder called MK14ESP8266keys and have the other files in the same folder. These are to split out the code and hopefully make it easier to read and modify.

File name	Description
MK14ESP8266keys.ino	The initial code file
htmlcode.ino	the basic HTML code
htmlcodeprocess.ino	the more complex html code
mk14hexread.ino	decodes the data in a hex file
mk14keys.ino	sends “keys” to the MK14
wificode.ino	handles the setup and html pages for the wifi connections
wifidata.h	Some wifi related data
zimage.ino	Handles sending the favicon.ico

The sketch should be loaded into the Arduino IDE and the correct board selected in the Tools menu. Here is a copy of the options I used when programming my ESP8266 LOLIN v3.



You then need to put the ESP8266 into flash mode. Some development boards do this automatically. I found that when running under Windows it seemed to work, were as running under linux (Debian 9) it did not.

The other issue I had under Linux was that the standard drivers for the CH340 USB to TTL chip used on the development board did not work. I downloaded the drivers, CH341SER_LINUX.ZIP, from http://www.wch-ic.com/downloads/CH341SER_LINUX_ZIP.html and followed the instructions to compile and then load it. My current issue is that it needs to be loaded after each reboot. Luckily reboots don't happen very often – I must fix the issue. I think if you are using a more modern linux build the problem may have been solved.

Once you have it loaded then I would suggest you select the "format the file store option" to setup the SPIFFS flash file system. If you had used one of the other file options before it maynot be compatible with SPIFFS.

There is a jumper attached to Arduino A7 and the ESP8266 A0 which can be used to tell the firmware which version of the MK14 OS it is talking to. Add jumper to use the original OS that starts with ---- --, leave open for the later versions.

When running the sketch outputs debug information to the serial port at 115200 baud. Among other things this will show the IP address of the ESP8266.

You can then connect using that IP address, for the very first time you will need to connect to the WAP SSID ESP8266MK14key, password mynetwork and then load http://192.168.8.1 in the browser.

From there you use the system, or you can go to the set SSID page and set the SSID and password. Once this has been set you can reboot the ESP8266 and if all is well it will connect to your wifi. Be careful not to “double” boot the ESP8266 else it will revert to WAP mode.

There is a video of the basic operations at https://www.youtube.com/watch?v=Oq6K6SKH3e0 maybe I’ll get some more details documented later.

For now you can

- Load .hex files into the ESP8266 memory
- Send them to the MK14 via the “mk14keyscp PCB”
- Delete any of the hex files
- Recover any deleted files
- and format the file system.

Hopefully we can add a further option to send key strokes directly to the MK14, but I’ll need to think about that.

There is a /dir option which will list all the files in the SPIFFS file system.

Stay safe
David Allday
November 2021