Software Manual

for eYFi-Mega Development Board

Produced by: e-Yantra Team

https://e-yantra.org/products/eyfi-mega

Version: 0.1.1

Dated: December 15, 2022

About this Document

This document is created for all developers wanting to work with the **eYFi-Mega** development board. It provides the detailed description and specification of the board.

Revision History

For revision history of this document, please refer to the Appendix D.

Disclaimer and Copyright Notice

Table of Contents

List of Figures

List of Tables

1 Default State of eYFi-Mega

The eYFi-Mega board comes out of the box with *preflashed* ***bootloaders***, ***partition table***, ***OTA(Over-the-air) application*** *and LED blink* ***user application*** on both AVR and ESP32 microcontrollers. On powering up the board with the micro-USB connector, the AVR application should blink the RGB LED onboard and the ESP32 application should blink the WIFI status LED.

1.1 Default SSID and Password of eYFi-Mega Wireless Access Point

Wi-Fi SSID: **eYFi-Mega**
Wi-Fi Password: **eyantra123**

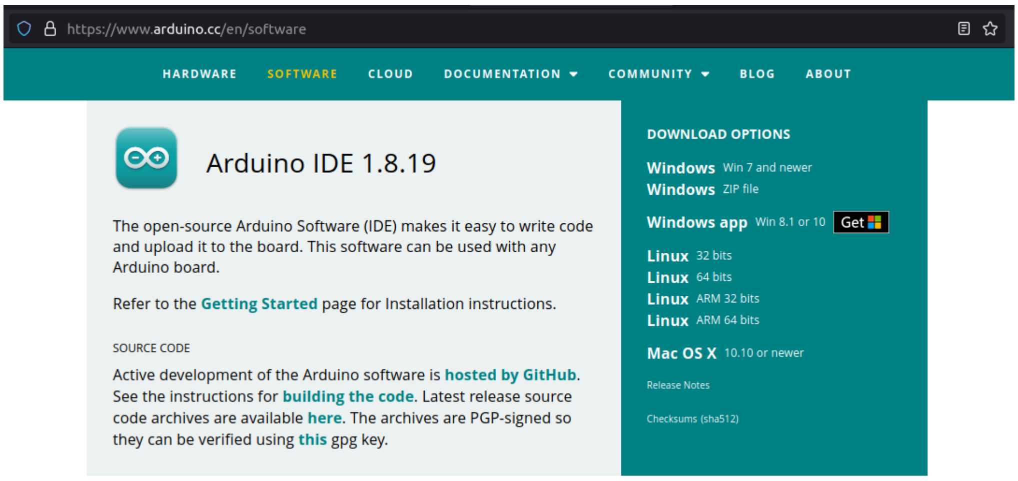
In case the *config.txt* file is missing from your file server or the format of the *config.txt* file is not proper then, this default SSID and Password will be set for your Wireless Access Point.

2 Programming eYFi-Mega

The eYFi-Mega supports **Arduino IDE** as a programming environment. After installing packages as instructed in upcoming sections, both ESP32 and ATmega2560 of eYFi-Mega can be programmed and flashed directly using Arduino IDE.

2.1 Installing Arduino IDE packages for eYFi-Mega

- Install Arduino IDE version 1.8.19.



Arduino IDE 1.8.19

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. This software can be used with any Arduino board.

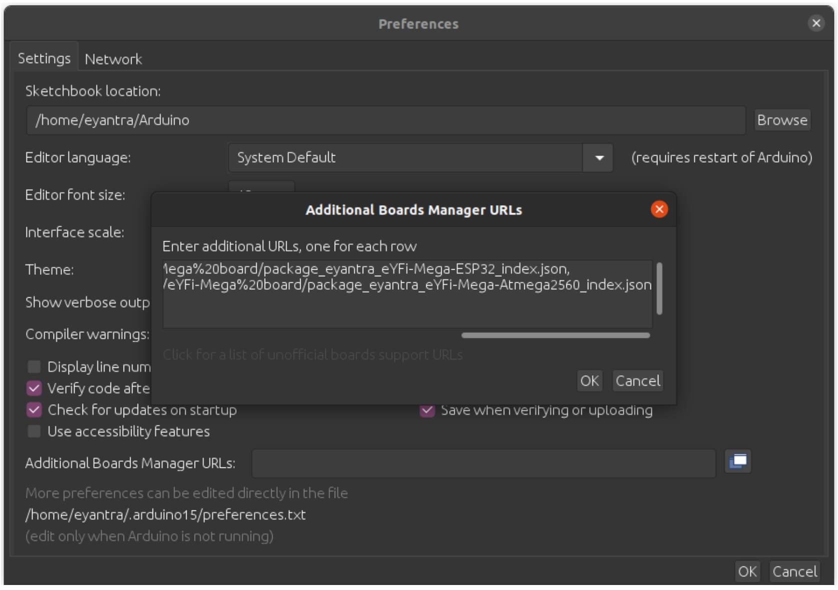
Refer to the **Getting Started** page for Installation instructions.

SOURCE CODE

Active development of the Arduino software is **hosted by GitHub**. See the instructions for **building the code**. Latest release source code archives are available **here**. The archives are PGP-signed so they can be verified using **this** gpg key.

Figure 1: Installing Arduino IDE 1.8.19

- Add package URLs in Arduino IDE -> **Files -> Preferences**
  For ATmega2560 : package URL
  For ESP32 : package URL



Preferences

Settings | Network

Sketchbook location: /home/eyantra/Arduino

Editor language: System Default (requires restart of Arduino)

Editor font size:

Interface scale:

Theme:

Show verbose outp

Compiler warnings:

Display line num

[✓] Verify code afte

[✓] Check for updates on startup

[✓] Save when verifying or uploading

Use accessibility features

Additional Boards Manager URLs:

More preferences can be edited directly in the file

/home/eyantra/.arduino15/preferences.txt

(edit only when Arduino is not running)

Additional Boards Manager URLs

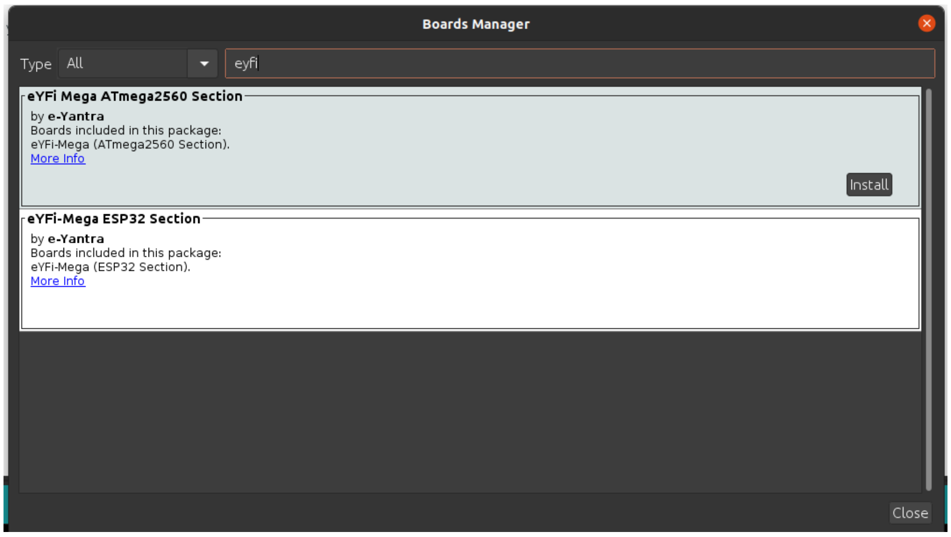
Enter additional URLs, one for each row

```
lega%20board/package_eyantra_eYFi-Mega-ESP32_index.json,
/eYFi-Mega%20board/package_eyantra_eYFi-Mega-Atmega2560_index.json
```

Click for a list of unofficial boards support URLs

Figure 2: Adding eYFi-Mega package URLs

- Install board packages in **Tools -> Boards Manager**. Search *"eyfi"* in the boards manager.



Boards Manager

Type All | eyfi

eYFi Mega ATmega2560 Section
by **e-Yantra**
Boards included in this package:
eYFi-Mega (ATmega2560 Section).
More Info

eYFi-Mega ESP32 Section
by **e-Yantra**
Boards included in this package:
eYFi-Mega (ESP32 Section).
More Info

Figure 3: Installing board packages

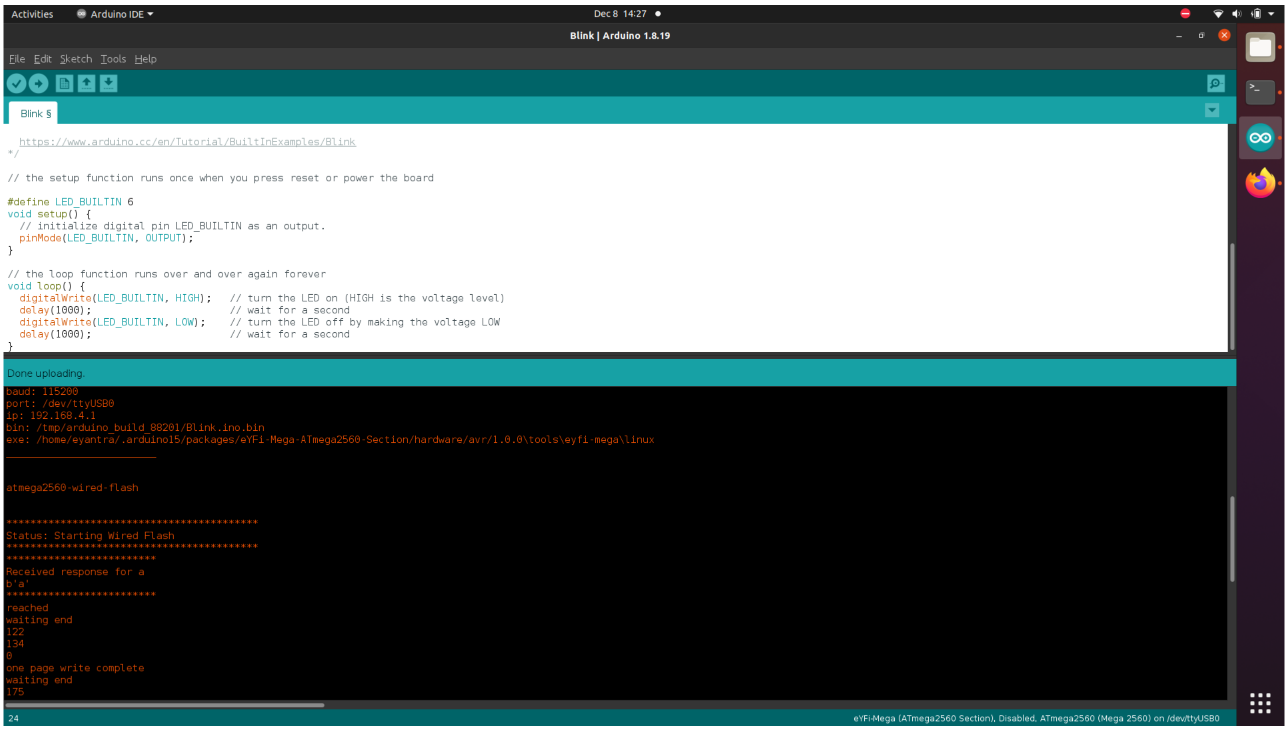
- eYFI board sections should be visible in **Tools -> Board** Selector dropdown
- On LINUX : change ESP & AVR flasher permissions : Find **.arduino15** folder in your $HOME directory

```
chmod u+x ~/.arduino15/packages/eYFi-Mega-ATmega2560-Section/
hardware/avr/1.0.0/tools/eyfi-mega/linux/eyfi*
```

2.2 Uploading program on ATmega2560/ESP32 using Arduino IDE

- Select board from **Tools -> Board** Selector dropdown

- Select USB port connected to eYFi-Mega board from **Tools -> Port** Selector drop-down
- **Compile** & **Upload**



```
  https://www.arduino.cc/en/Tutorial/BuiltInExamples/Blink
*/

// the setup function runs once when you press reset or power the board

#define LED_BUILTIN 6
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);                       // wait for a second
  digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);                       // wait for a second
}
```

Done uploading.

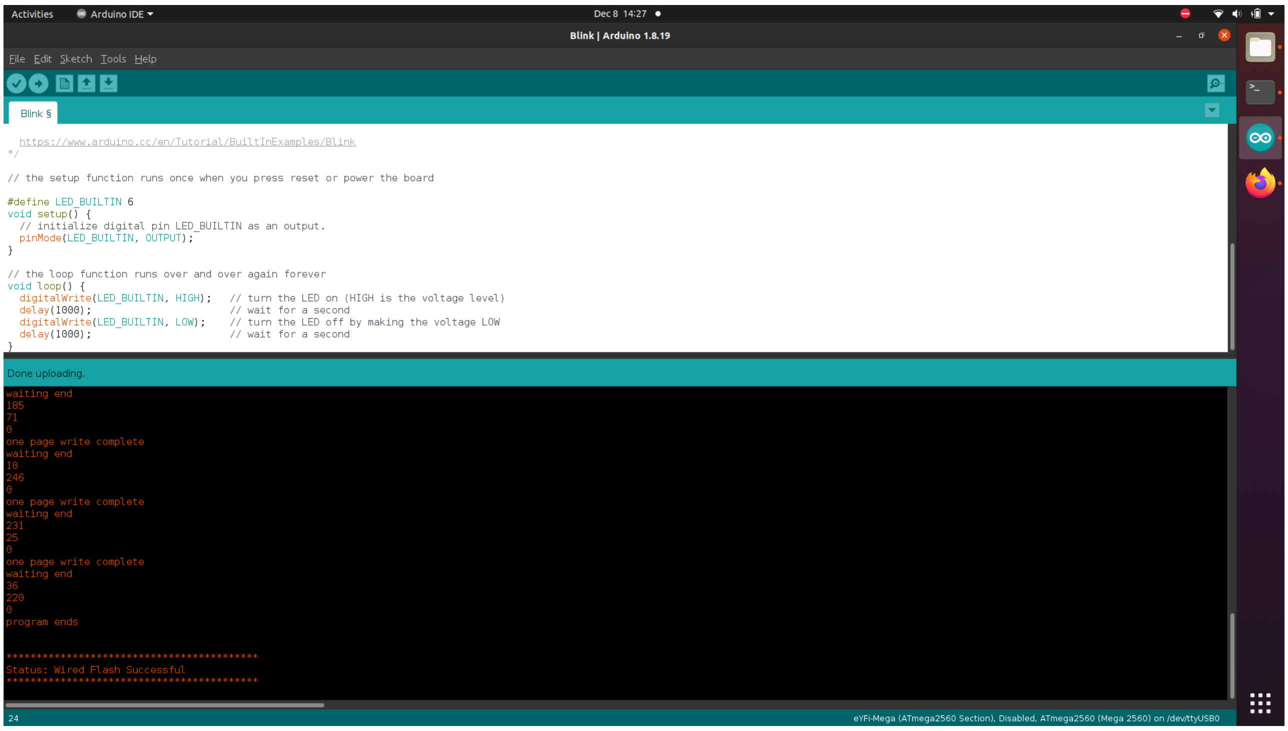
```
baud: 115200
port: /dev/ttyUSB0
ip: 192.168.4.1
bin: /tmp/arduino_build_88201/Blink.ino.bin
exe: /home/eyantra/.arduino15/packages/eYFi-Mega-ATmega2560-Section/hardware/avr/1.0.0\tools\eyfi-mega\linux
_________________________


atmega2560-wired-flash


******************************************
Status: Starting Wired Flash
******************************************
*************************
Received response for a
b'a'
*************************
reached
waiting end
122
134
0
one page write complete
waiting end
175
```

24   eYFi-Mega (ATmega2560 Section), Disabled, ATmega2560 (Mega 2560) on /dev/ttyUSB0

Figure 4: Flashing started for ATmega2560



```
  https://www.arduino.cc/en/Tutorial/BuiltInExamples/Blink
*/

// the setup function runs once when you press reset or power the board

#define LED_BUILTIN 6
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);                       // wait for a second
  digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);                       // wait for a second
}
```

Done uploading.

```
waiting end
185
71
0
one page write complete
waiting end
10
246
0
one page write complete
waiting end
231
25
0
one page write complete
waiting end
36
220
0
program ends


******************************************
Status: Wired Flash Successful
******************************************
```

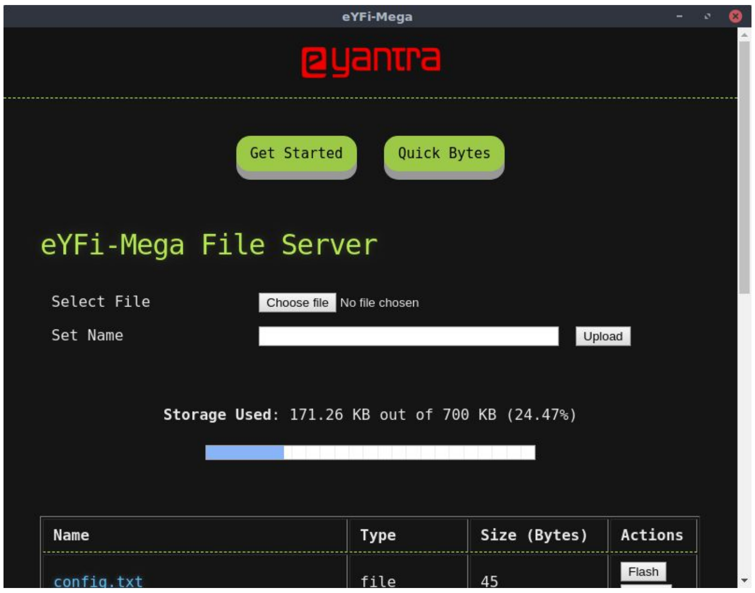
24   eYFi-Mega (ATmega2560 Section), Disabled, ATmega2560 (Mega 2560) on /dev/ttyUSB0

Figure 5: Flashing completed for ATmega2560

3 eYFi-Mega OTA Application

The eYFi-Mega OTA Application which resides in the factory partition of ESP32, allows user to flash the firmware of ATmega 2560 or ESP32 wirelessly and gives access to the file storage of eYFi-Mega.

3.1 eYFi-Mega File Server



eYFi-Mega File Server

Select File No file chosen

Set Name

Storage Used: 171.26 KB out of 700 KB (24.47%)

Name	Type	Size (Bytes)	Actions
config.txt	file	45	Flash

Figure 6: eYFI-Mega Fileserver

eYFi-Mega File Server is like a file explorer for the 700 KB onboard storage. To access this file server you need to first connect to eYFi-Mega's Wireless Access Point. After connecting to the Access Point you can use any web-browser to open the file server window at **192.168.4.1** IP address.

You can use this storage to store any files like you can use this storage to configure your WiFi-SSID and password (explained in the next section), log data from sensors, you can even store **.bin** or **.hex** files of your firmware and flash it in any of the two controllers through this file server.

NOTE: When you are setting the name of your file which is to be uploaded, keep the name of the file including the name of the extension less than 31 characters in length.

3.2 Set Wi-Fi SSID and Password

To set your own Wi-Fi SSID and Password of the board, upload a **config.txt** file to your eYFi-Mega File Server with your SSID and Password in the following format in the file. After the upload is done, reset the ESP32 by pressing the **ESP_RESET** button, and you are all set!

Filename: **config.txt**
WiFi-SSID [your-wifi-ssid]
WiFi-Pass [your-wifi-pass]

For example,
WiFi-SSID [eYFi-Mega]
WiFi-Pass [eyantra123]
will set the SSID to "eYFi-Mega" and Password to "eyantra123".

3.3 Wi-Fi Status LED Indication

When you are connected to your eYFi-Mega over a Wi-Fi link, it's important to know the different states in which your board is. These states are indicated by the Blue **Wi-Fi Status** LED, which is right next to Red **ESP_ON** LED. Below is the description of different LED Patterns associated with different states of the board.

Table 1: Wifi-Status LED indications for different states

State	**LED Pattern**
Wi-Fi Client is connected to ESP32	ON
Wi-Fi Client is disconnected from ESP32	OFF
File Upload Start	OFF
File Upload End	Blink Fast for 100 ms
Firmware Flash Start	Blink Fast
Firmware Flash End for AVR	Blink Slow for 5s
Firmware Flash End for ESP32	Blink Slow till application Switch #1(S1) is toggled

4 eYFi-Mega Wireless Serial Terminal

We have developed an ESP32 application which allows ATMega 2560 to transmit its data on UART #0 over Wi-Fi. So with the help of this application, you can receive serial data over Wi-Fi at your side.

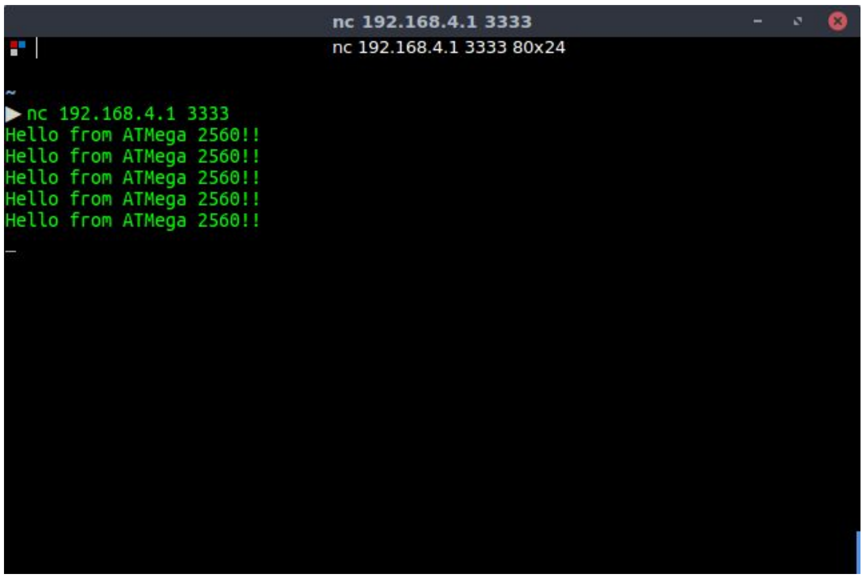
- To use this first, you would have to flash this application which you can download from eYFi-Mega's website in the user-app partition of ESP32.
- After that make sure the S2 switch is towards the Wi-Fi symbol. This will connect UART #0 of ATMega 2560 with UART #1 of ESP32.
- To start this application switch S1 away from the Wi-Fi symbol and then press **ESP_RESET** button, to start the application.
- In your ATMega 2560 application make sure that you are sending data to UART #0 at a baud rate of 115200.
- To receive data on your Linux system you can use **netcat** utility. Run the following command in the terminal to receive data from the UART #0 of ATMega 2560 over Wi-Fi,

```
nc 192.168.4.1 3333
```

You can also run this command on your Android Device. To run terminal commands on your Android Device use **Termux App** which you can download from Google Play Store. Termux App Download URL

Before you run this command, make sure that you are connected to the Wireless Access Point of eYFi-Mega. For this application the Wi-Fi SSID is **eYFi-Wireless-Serial** and there is no password.

In the Quick Bytes page of eYFi-Mega which you can find at eYFi-Mega's website, you can find the example code for using this Wireless Serial Terminal.



```
nc 192.168.4.1 3333
nc 192.168.4.1 3333 80x24

~
▶ nc 192.168.4.1 3333
Hello from ATMega 2560!!
Hello from ATMega 2560!!
Hello from ATMega 2560!!
Hello from ATMega 2560!!
Hello from ATMega 2560!!
_
```

Figure 7: eYFI Wireless Serial Terminal Output

5 eYFi-Mega ESP32 Partition Table

We have partitioned the 4 MB flash of onboard ESP32 to accommodate 1 MB OTA App, 2 MB for User App and 700 KB for File Storage. This partition table can be modified by you as per your requirement.
For example, if you want to increase or decrease the size of the 700 KB File Storage you can update the partition table. If you decide to modify the partition table for your project, you might not be able to use the OTA App because our OTA App is compatible with the partition table described below. In case you accidentally erase the partition you can always download our Partition Table and our OTA App from our website. The current partition table is described below,

Table 2: eYFi-Mega ESP32 Partitions

Partition Name	**Start Address**	**Size**
reserved area (nvs, otadata, phy_init	-	300 KB
factory (ota-app)	0x10000	1 MB
app1 (user-app)	0x110000	2 MB
storage (spiffs)	0x310000	700 KB

A Know-hows for Linux systems

A.1 Kill Any Command

While any command is being executed either in your terminal and you want to kill it, just press **CTRL+C** in the terminal.

A.2 Linux and Blank Spaces

Linux systems are not good in handling blank spaces so, we recommend not to have any blank spaces in your folder name or filename. Use **camelCase** or **snake_case** instead of having a blank space between two words.

B Flashing eYFi-Mega ESP32

In case you accidentally erase any of the important firmware which is required for eYFi-Mega to function properly, you can always download them from our website and flash them again. In this section, we will discuss how you can do that.

To flash firmware in ESP32 you would need **esptool**. The **esptool** and the firmware described in the following sections can be downloaded from eYFi-Mega's website which can be found at e-Yantra's products page.

B.1 Flashing eYFi-Mega ESP32 Bootloader

Copy the bootloader file in the esptool folder and execute the following command to flash the bootloader.

```
$ python esptool.py -b 921600 --port /dev/ttyUSB0 write_flash 0x1000
eyfi-mega_esp32_bootloader.bin
```

B.2 Flashing eYFi-Mega ESP32 Partition Table

Copy the partition table file in the esptool folder and execute the following command to flash the partition table.

```
$ python esptool.py -b 921600 --port /dev/ttyUSB0 write_flash 0x8000
eyfi-mega_esp32_partitions.bin
```

B.3 Flashing eYFi-Mega ESP32 OTA Application

Copy the OTA Application file in the esptool folder and execute the following command to flash the OTA Application.

```
$ python esptool.py -b 921600 --port /dev/ttyUSB0 write_flash 0x10000
eyfi-mega_esp32_ota_app.bin
```

B.4 Flashing eYFi-Mega ESP32 User Application and Wireless Serial Terminal Application

Copy the Wireless Serial Terminal Application or User Application file in the esptool folder and execute the following command to flash the file.

```
$ python esptool.py -b 921600 --port /dev/ttyUSB0 write_flash 0x110000
eyfi-mega_esp32_user_app.bin
```

C

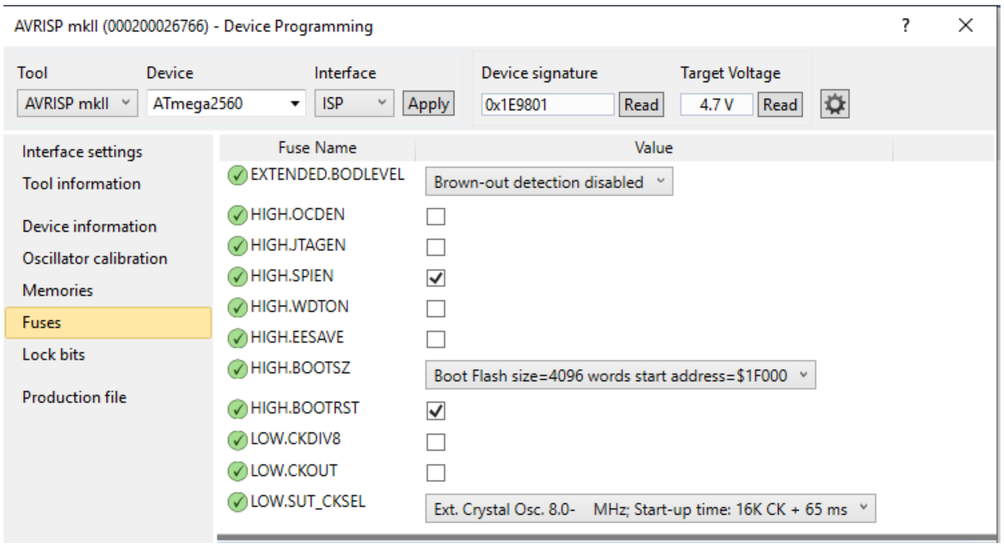
Flashing Boot-loader on AVR

Note: Refer to this section only if you have accidentally overwritten the Boot-loader firmware of ATmega2560 on eYFi-Mega board.

Follow each of the steps given below carefully:

1. Components and Software required to get started:
   (a) eYFi-Mega board
   (b) USB micro-B plug to USB-A plug cable
   (c) USB A-Male to B-Male cable
   (d) Atmel AVRISP mkII (for more details, refer to link)
   (e) Atmel Studio 7 software (only on Windows platform)
   (f) Boot-loader firmware of ATmega2560 micro-controller on eYFi-Mega board (can be downloaded from **Downloads** section on **eYFi-Mega** page, link: `https://e-yantra.org/products/eyfi-mega`).
2. Connect the USB A-Male to B-Male cable to the PC and Atmel AVRISP mkII.
3. Connect the 6-pin AVR ISP FRC cable of Atmel AVRISP mkII to the ICSP header on eYFi-Mega board.
   Note: The notch on the FRC cable should match with the notch provided around the ICSP header.
4. Connect the USB micro-B plug to USB-A plug cable to the PC and Micro-USB connector on eYFi-Mega board.
5. Once the above connections are done, check the LED status on Atmel AVRISP mkII. If the LED color is **Green**, this means that everything is working fine and you can proceed with the next step. But, if the LED color is **Orange** or **Red**, this means that the connections between the Atmel AVRISP mkII and eYFi-Mega board is weak.

6. Open the Atmel Studio 7 software. Select the option **Device Programming** or use the shortcut **Ctrl + Shift + P**. A dialog box will open as shown in Figure 8.
7. Under **Tool** option, select **AVRISP mkII**.
8. Under **Device** option, select **ATmega2560**.
9. Under **Interface** option, select **ISP**.
10. Click on **Apply**.
11. Click on **Read** to read the Device Signature.
12. Check whether the **Device Signature** value is **0x1E9801** and the **Target Voltage** is around **5V**.
13. Go to **Fuses** and select the Fuse settings as shown in the Figure 8.



AVRISP mkII (000200026766) - Device Programming

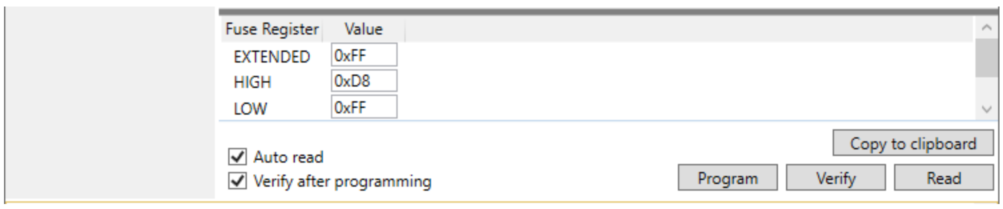
Tool	Device	Interface		Device signature		Target Voltage	
AVRISP mkII	ATmega2560	ISP	Apply	0x1E9801	Read	4.7 V	Read

Interface settings
Tool information
Device information
Oscillator calibration
Memories
Fuses
Lock bits
Production file

Fuse Name	Value
EXTENDED.BODLEVEL	Brown-out detection disabled
HIGH.OCDEN	☐
HIGH.JTAGEN	☐
HIGH.SPIEN	☑
HIGH.WDTON	☐
HIGH.EESAVE	☐
HIGH.BOOTSZ	Boot Flash size=4096 words start address=$1F000
HIGH.BOOTRST	☑
LOW.CKDIV8	☐
LOW.CKOUT	☐
LOW.SUT_CKSEL	Ext. Crystal Osc. 8.0- MHz; Start-up time: 16K CK + 65 ms

Figure 8: Device Programming dialog box in Atmel Studio 7

14. Make sure that the option of **Auto read** and **Verify after programming** are selected as shown in Figure 8.



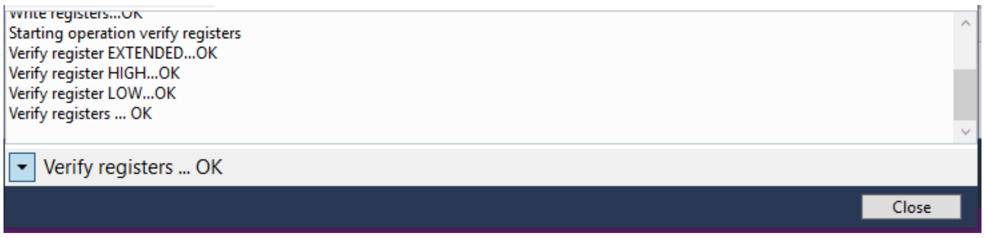
Fuse Register	Value
EXTENDED	0xFF
HIGH	0xD8
LOW	0xFF

☑ Auto read
☑ Verify after programming

Figure 9: Fuse settings to be done

15. Once you are done with the above settings, click on **Program**.

16. After the programming is complete, you will see the message as shown in the below Figure 8.



```
Write registers...OK
Starting operation verify registers
Verify register EXTENDED...OK
Verify register HIGH...OK
Verify register LOW...OK
Verify registers ... OK
```

Verify registers ... OK

Figure 10: All OK message after programming the Fuses

17. Go to **Memories**, select **Erase Now** to erase the ATmega2560 chip before flashing the Boot-loader firmware on to it.

18. Make sure the options **Erase device before programming** and **Verify flash device after programming** are selected.

19. Select the Boot-loader firmware downloaded under the **Flash (256 KB)** section and press **Program**.

20. Once the programming is complete, you will get the message: **Verifying Flash....OK**. The LED color on Atmel AVRISP mkII will turn to **Orange** and will blink continuously.

21. Congrats! Your eYFi-Mega board is updated with the Boot-loader firmware of ATmega2560.

D Revision History

Table 3: Revision History of the document

Date	**Version**	**Release notes**
December 14, 2019	v0.1	First release of the document
December 15, 2022	v0.1.1	Added Arduino IDE support Added default state of eYFi-Mega board Discarded VScode section