Arduino IDE (Integrated Development Environment) Installation, Understanding and Programming

What is Arduino?

Arduino is an open-source electronics platform with easy-to-use hardware and software interface. Though it also uses microcontroller to act as the brain, it stands unique in the way it can be programmed easily with a big bunch of pre-installed libraries. Arduino also simplifies the process of working with microcontrollers, but it offers some advantage for teachers, students, and interested amateurs over other systems.

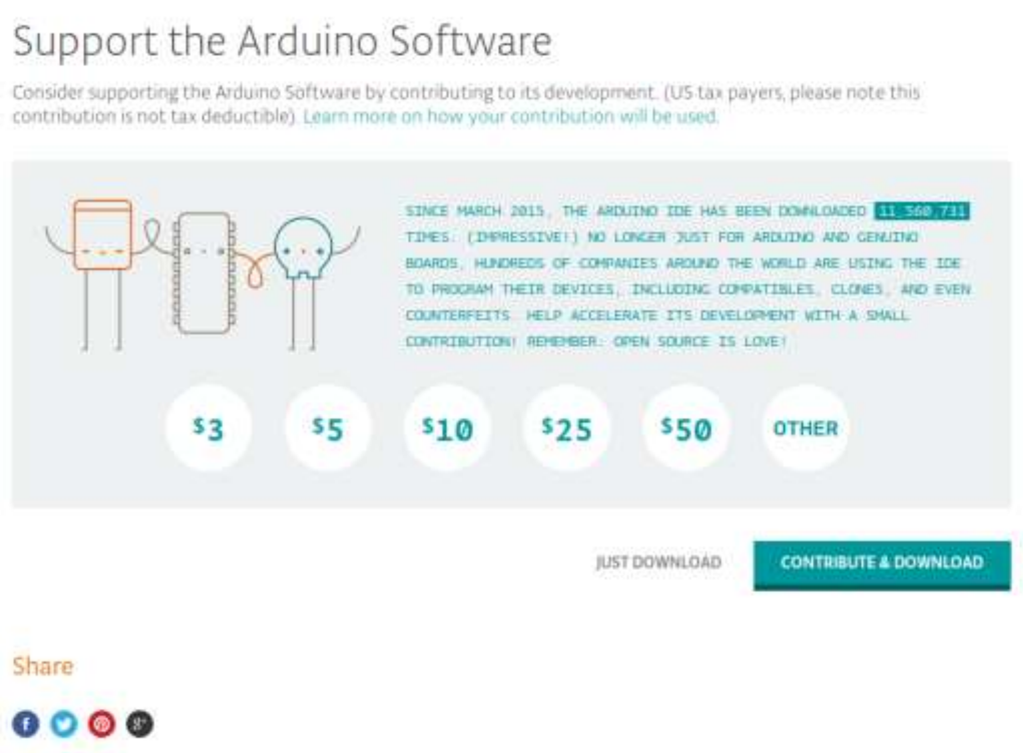
Why Arduino now?

The main reason for choosing Arduino IDE to program our hardware is that it has ROS library through which node creation and handling becomes easy and understandable. Adding ROS libraries (contains all the necessary files for using ROS) implies that we have to use these in programming ATmega2560.

From the above points, we can infer that we are not in real need with the Arduino development board but we are with Arduino IDE.

Arduino IDE Installation:

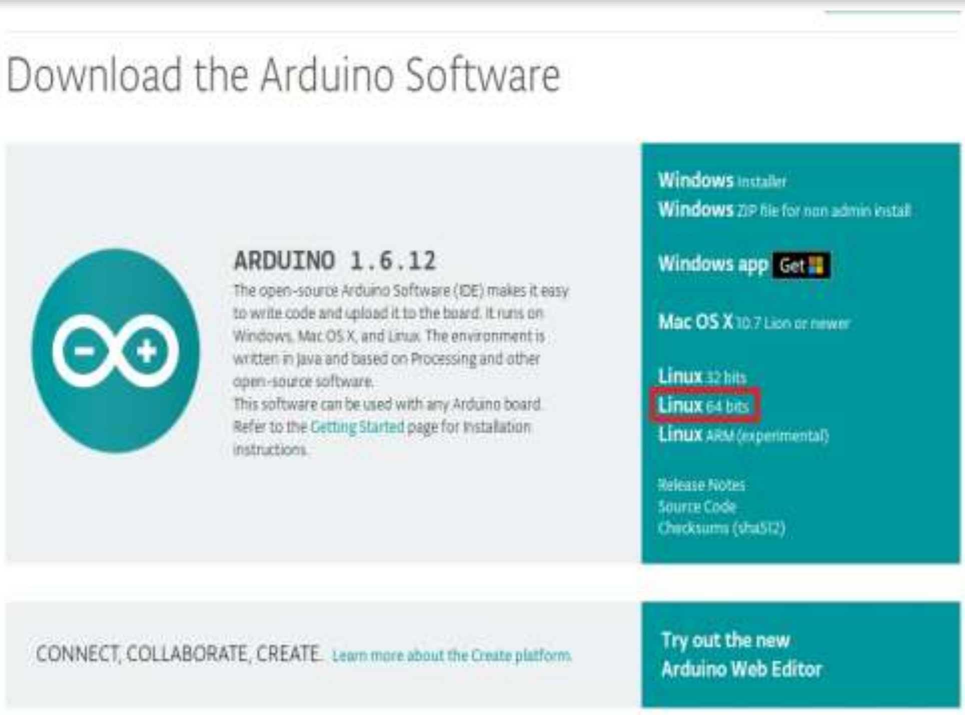
1. Open https://www.arduino.cc/en/Main/Donate . You will see a page with the following part.
2. Click on JUST DOWNLOAD



Support the Arduino Software

Consider supporting the Arduino Software by contributing to its development. (US tax payers, please note this contribution is not tax deductible). Learn more on how your contribution will be used.

SINCE MARCH 2015, THE ARDUINO IDE HAS BEEN DOWNLOADED 11,560,731 TIMES. (IMPRESSIVE!) NO LONGER JUST FOR ARDUINO AND GENUINO BOARDS, HUNDREDS OF COMPANIES AROUND THE WORLD ARE USING THE IDE TO PROGRAM THEIR DEVICES, INCLUDING COMPATIBLES, CLONES, AND EVEN COUNTERFEITS. HELP ACCELERATE ITS DEVELOPMENT WITH A SMALL CONTRIBUTION! REMEMBER: OPEN SOURCE IS LOVE!


Download the Arduino Software

ARDUINO 1.6.12

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. It runs on Windows, Mac OS X, and Linux. The environment is written in Java and based on Processing and other open-source software.
This software can be used with any Arduino board. Refer to the Getting Started page for installation instructions.

Windows Installer
Windows ZIP file for non admin install
Windows app Get
Mac OS X 10.7 Lion or newer
Linux 32 bits
Linux 64 bits
Linux ARM (experimental)
Release Notes
Source Code
Checksums (sha512)
CONNECT, COLLABORATE, CREATE. Learn more about the Create platform.
Try out the new Arduino Web Editor

4. Extract *arduino-1.6.12.tar.gz*

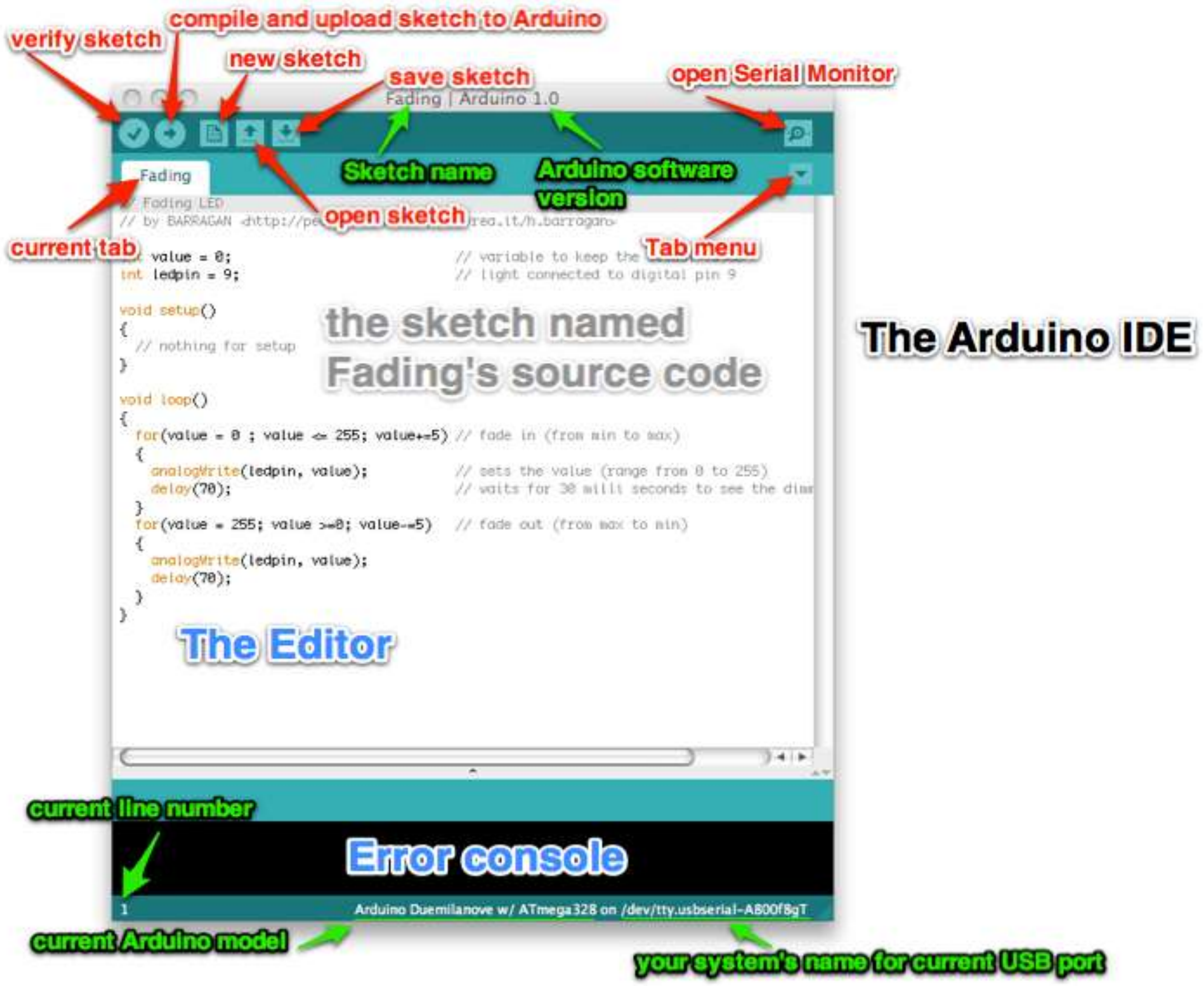
5. Open terminal and change directory to extracted folder by the following command: (From now on, we assume that you have downloaded it in Downloads directory, else change the path accordingly)

   cd Downloads/arduino-1.6.12

6. Run the executable file by the following command:

   ./install.sh

Getting familiarized with Arduino IDE:



```
// Fading LED
// by BARRAGAN <http://pe              rea.it/h.barragan>

    value = 0;                           // variable to keep the
int ledpin = 9;                          // light connected to digital pin 9

void setup()
{
  // nothing for setup
}

void loop()
{
  for(value = 0 ; value <= 255; value+=5) // fade in (from min to max)
  {
    analogWrite(ledpin, value);           // sets the value (range from 0 to 255)
    delay(70);                            // waits for 30 milli seconds to see the dim
  }
  for(value = 255; value >=0; value-=5)   // fade out (from max to min)
  {
    analogWrite(ledpin, value);
    delay(70);
  }
}
```

```
1                 Arduino Duemilanove w/ ATmega328 on /dev/tty.usbserial-A800f8gT
```

Source: http://www.hacdc.org/summer-school-2013/

Programming with Arduino IDE:

If you have basics in C then you are good to go with programming in Arduino. Before going to directly program Arduino go through this link http://playground.arduino.cc/uploads/Main/arduino_notebook_v1-1.pdf

Start only if you are confident enough in key concepts of programming in Arduino.

Compiling and Debugging:

Use *compile* (verify check) option in the IDE of Arduino to compile the program. Go through this to know about common errors and troubleshooting them. Try reading the error message which in most cases explains about the cause of the error. If not, try to resolve the error with extensive web based sources.

If your code gets compiled successfully it shows a message as “Build succeeded”. Then you can program your microcontroller.

Note that the .hex file will be generated in the path /tmp/arduino_build_xxxxx. Arduino_build_xxxxx is a folder in which xxxxx corresponds to a number which will be different at different times. The .hex has the filename of the form <sketchname>.ino.hex. Remember that the .hex file which has to be programmed doesn't contain "with_bootloader" in its file name.

Go through NEX_AVR_USB_ISP_STK500V2.pdf manual when programming for the first time.

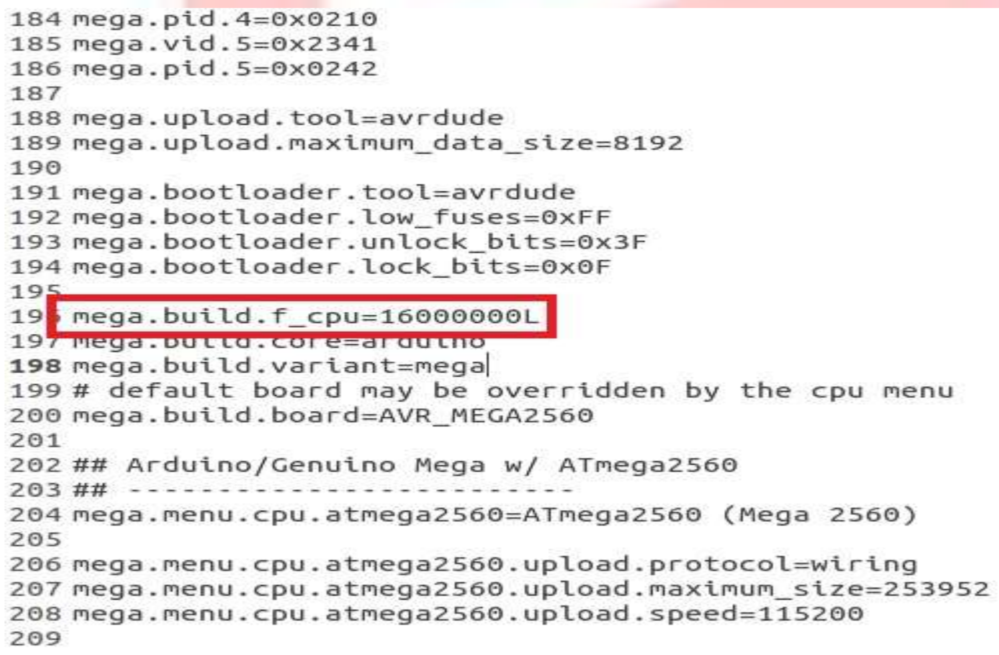
Setting up frequency:

It has to be noted that we are using Atmega2560 which operates at a frequency of 14745600 Hz. Unless you change the frequency to 14745600, Arduino IDE considers the frequency as 16MHz in compiling the code making it unfeasible for ATmega2560 programming.

To change the frequency to 14745600 perform the following steps:

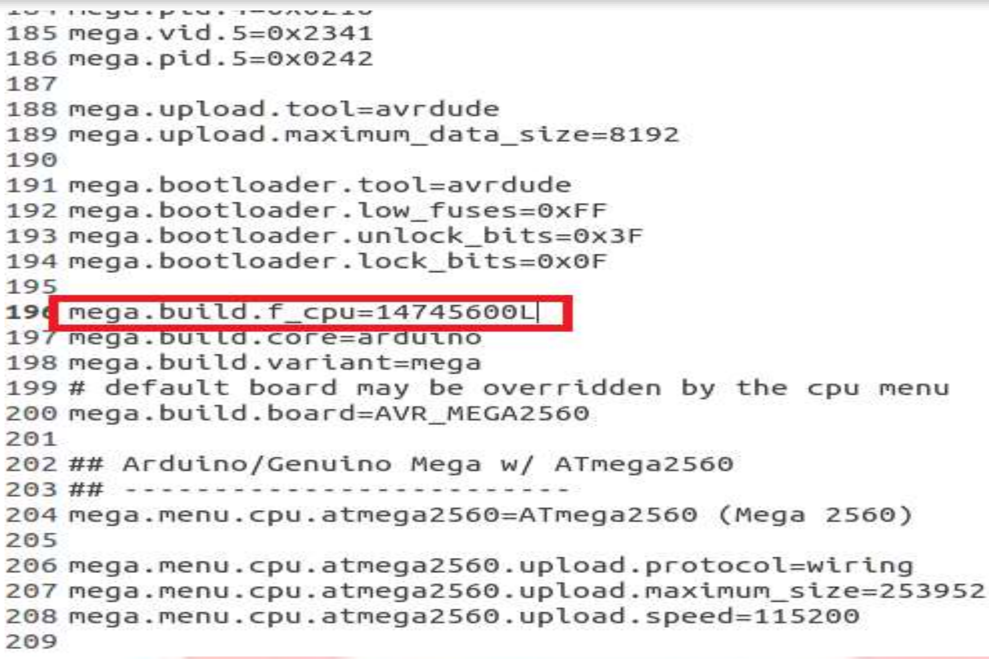
- Go to the location where you have saved the Arduino folder.
- Open the path ~/Downloads/hardware/arduino/avr
- Open boards.txt

Find the line in boards.txt which is highlighted **(mega.build.f_cpu=16000000L)** in the figure below:



```
mega.pid.4=0x0210
mega.vid.5=0x2341
mega.pid.5=0x0242

mega.upload.tool=avrdude
mega.upload.maximum_data_size=8192

mega.bootloader.tool=avrdude
mega.bootloader.low_fuses=0xFF
mega.bootloader.unlock_bits=0x3F
mega.bootloader.lock_bits=0x0F

mega.build.f_cpu=16000000L
mega.build.core=arduino
mega.build.variant=mega
# default board may be overridden by the cpu menu
mega.build.board=AVR_MEGA2560

## Arduino/Genuino Mega w/ ATmega2560
## -------------------------
mega.menu.cpu.atmega2560=ATmega2560 (Mega 2560)

mega.menu.cpu.atmega2560.upload.protocol=wiring
mega.menu.cpu.atmega2560.upload.maximum_size=253952
mega.menu.cpu.atmega2560.upload.speed=115200
```

Change 16000000L to 14745600L (as shown in figure below) and save the file.



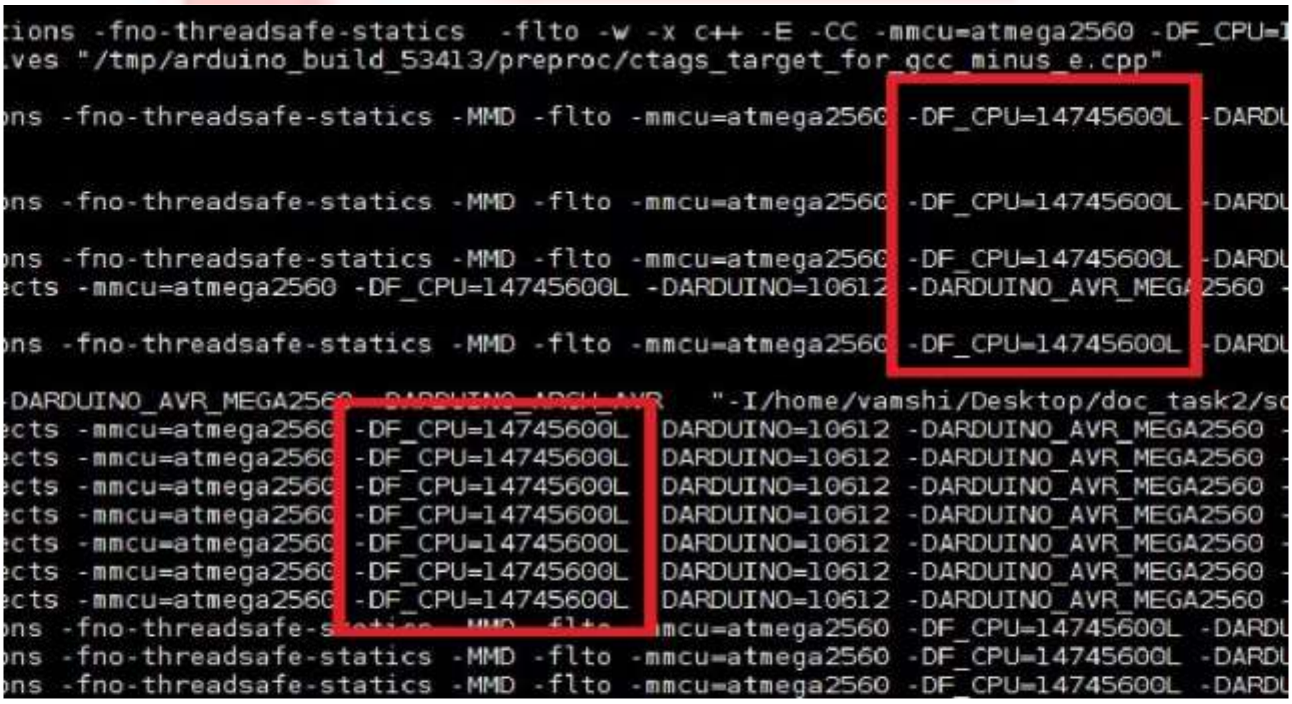
```
mega.vid.5=0x2341
mega.pid.5=0x0242

mega.upload.tool=avrdude
mega.upload.maximum_data_size=8192

mega.bootloader.tool=avrdude
mega.bootloader.low_fuses=0xFF
mega.bootloader.unlock_bits=0x3F
mega.bootloader.lock_bits=0x0F

mega.build.f_cpu=14745600L
mega.build.core=arduino
mega.build.variant=mega
# default board may be overridden by the cpu menu
mega.build.board=AVR_MEGA2560

## Arduino/Genuino Mega w/ ATmega2560
## -------------------------
mega.menu.cpu.atmega2560=ATmega2560 (Mega 2560)

mega.menu.cpu.atmega2560.upload.protocol=wiring
mega.menu.cpu.atmega2560.upload.maximum_size=253952
mega.menu.cpu.atmega2560.upload.speed=115200
```

Check if everything is going right:

From the File menu, go to Examples and load Led Blink (File-> Examples -> 01.Basics -> Blink). Compile the code using verify sketch icon.

After building the code, make sure that there is no error and warning.

Scroll up a bit in error console to make sure that the frequency under operation is 14745600L as shown in figure below.



```
tions -fno-threadsafe-statics  -flto -w -x c++ -E -CC -mmcu=atmega2560 -DF_CPU=1
ves "/tmp/arduino_build_53413/preproc/ctags_target_for_gcc_minus_e.cpp"

ns -fno-threadsafe-statics -MMD -flto -mmcu=atmega2560 -DF_CPU=14745600L -DARDU


ns -fno-threadsafe-statics -MMD -flto -mmcu=atmega2560 -DF_CPU=14745600L -DARDU

ns -fno-threadsafe-statics -MMD -flto -mmcu=atmega2560 -DF_CPU=14745600L -DARDU
ects -mmcu=atmega2560 -DF_CPU=14745600L -DARDUINO=10612 -DARDUINO_AVR_MEGA2560 -

ns -fno-threadsafe-statics -MMD -flto -mmcu=atmega2560 -DF_CPU=14745600L -DARDU

-DARDUINO_AVR_MEGA2560 -DARDUINO_ARCH_AVR   "-I/home/vamshi/Desktop/doc_task2/sc
ects -mmcu=atmega2560 -DF_CPU=14745600L DARDUINO=10612 -DARDUINO_AVR_MEGA2560 -
ects -mmcu=atmega2560 -DF_CPU=14745600L DARDUINO=10612 -DARDUINO_AVR_MEGA2560 -
ects -mmcu=atmega2560 -DF_CPU=14745600L DARDUINO=10612 -DARDUINO_AVR_MEGA2560 -
ects -mmcu=atmega2560 -DF_CPU=14745600L DARDUINO=10612 -DARDUINO_AVR_MEGA2560 -
ects -mmcu=atmega2560 -DF_CPU=14745600L DARDUINO=10612 -DARDUINO_AVR_MEGA2560 -
ects -mmcu=atmega2560 -DF_CPU=14745600L DARDUINO=10612 -DARDUINO_AVR_MEGA2560 -
ects -mmcu=atmega2560 -DF_CPU=14745600L DARDUINO=10612 -DARDUINO_AVR_MEGA2560 -
ns -fno-threadsafe-statics -MMD -flto -mmcu=atmega2560 -DF_CPU=14745600L -DARDU
ns -fno-threadsafe-statics -MMD -flto -mmcu=atmega2560 -DF_CPU=14745600L -DARDU
ns -fno-threadsafe-statics -MMD -flto -mmcu=atmega2560 -DF_CPU=14745600L -DARDU
```