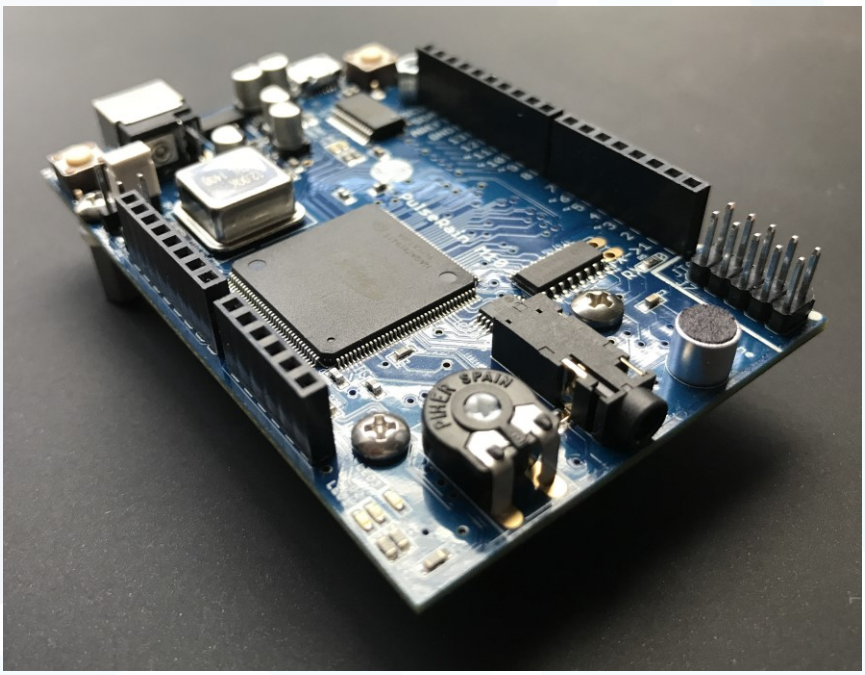
10555 Scripps Trl, San Diego, CA 92131
858-877-3485 858-408-9550
http://www.pulserain.com

Doc# QSG-0922-0039, Rev 1.1

PulseRain M10

Quick Start Guide



Sep, 2017

This page is intentionally left blank.

Table of Contents

1 Introduction

1.1 Hardware Overview

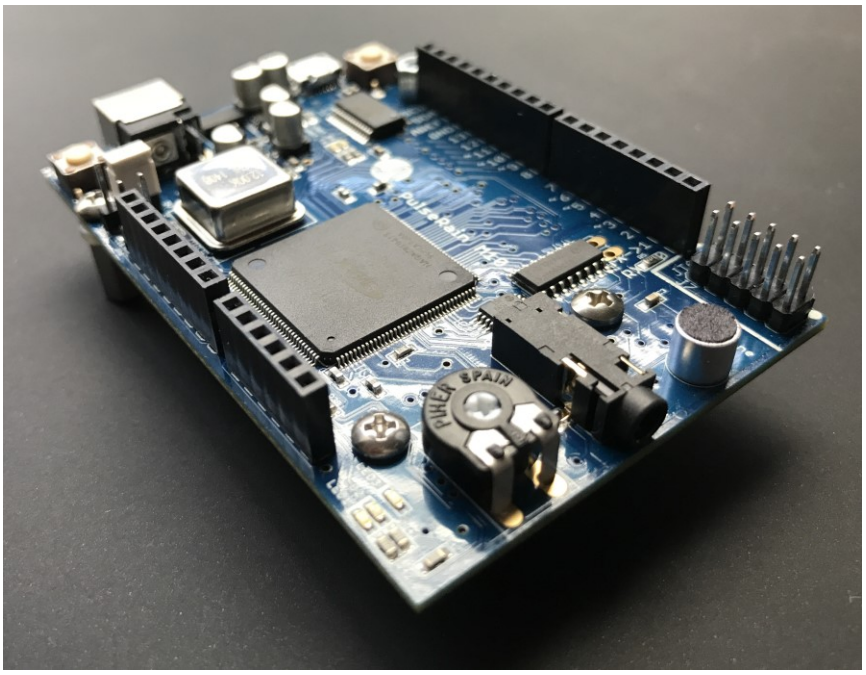


Figure 1-1 The Close View of M10

As shown Figure 1-1, the M10 board has a form factor that is compatible with Arduino UNO R3. But unlike Arduino who uses a hardcore MCU, the M10 takes a distinctive technical approach by embedding an open source soft-core MCU (96MHz) into an Intel MAX10 FPGA, while maintaining an Arduino compatible software interface. On top of that, it also carries a rich set of onboard peripherals that Arduino does not have, with dual IO voltage support (3.3V and 5V). In fact, you can now completely replace your Arduino with M10!

The major components that M10 carries are the following:

- On the top side, as illustrated in Figure 1-2:
  - Intel MAX10 FPGA – 10M08SAE144C8G
  - Onboard Microphone for Voice Record, and 3.5mm Audio Jack for Speaker
  - 12MHz Oscillator
  - Potentiometer
  - 10 pin JTAG connector for FPGA
  - SRAM (23LC1024-I/ST)
  - Voice Codec (Si3000-C-FS)
  - USB / UART (FT232RL)
  - Two Push Buttons
  - Jumpers (JP1 and JP6) for IO Voltage Configuration and Flash Program respectively

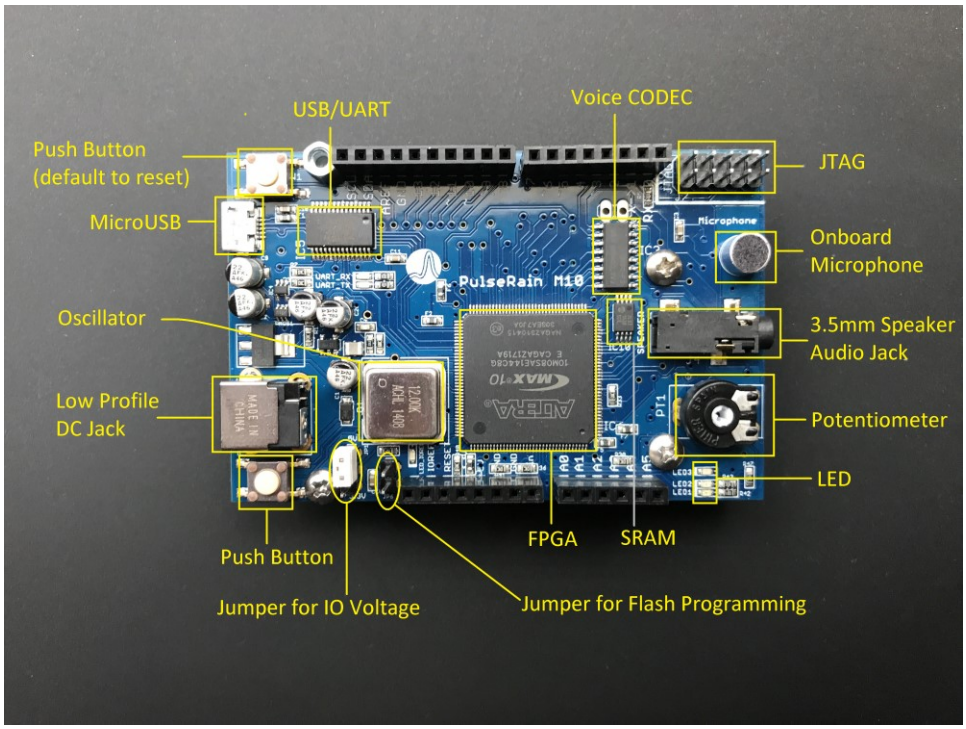


Figure 1-2 The Top View of M10

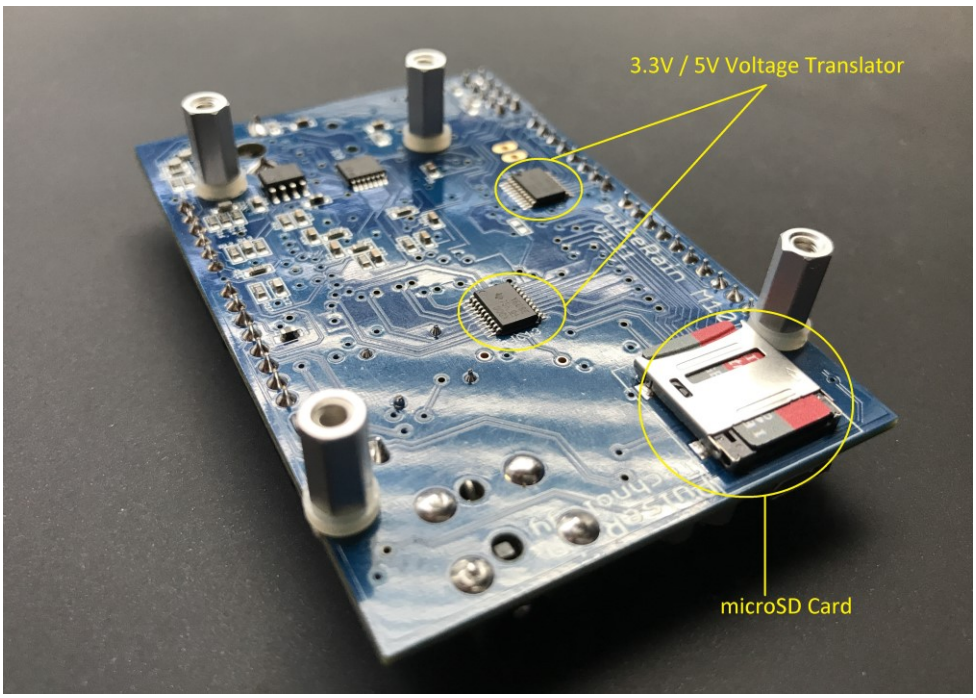


Figure 1-3 The Bottom View of M10

- On the bottom side, as illustrated in Figure 1-3:
  - Voltage Level Shifter to support both 3.3 V and 5V IO (TXS0108EPWR)
  - Socket for microSD card

1.2 Supply of Power

The Power to M10 board can be supplied either from the low profile DC jack or from the microUSB port. If DC jack is used, a DC supply of 7V – 15V can be used (Figure 1-4). Otherwise, power can come from the microUSB port's 5V rail (Figure 1-5). In fact, the microUSB port serves dual purposes: In addition to being a power source, the microUSB is also used for communication between host PC and the M10 board.

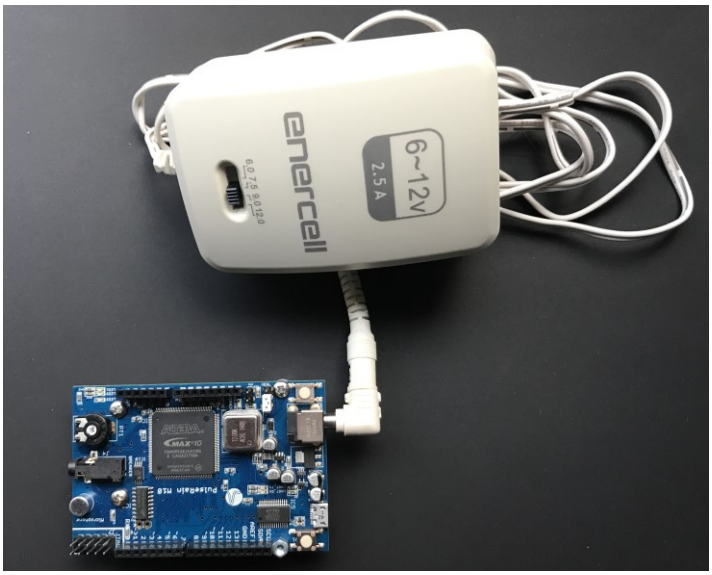


Figure 1-4 Supply Power through DC Jack (7V – 12V)

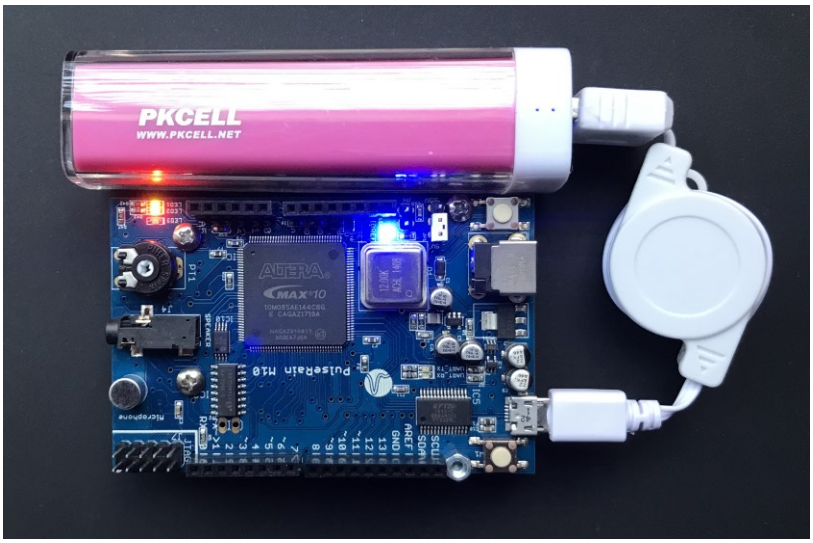


Figure 1-5 Supply Power through microUSB

1.3 Design Flow

As mentioned early, the factory FPGA image that comes with the M10 board already has a soft-core MCU, which supports an Arduino compatible software interface. If you are familiar with Arduino, and want to get things going quickly, all you have to do is: (as shown in the right part of Figure 1-6)

1) Connect the M10 board to PC through a microUSB cable
2) Set up the Arduino IDE
3) Write your sketch just like you would do with Arduino
4) To access the onboard peripherals, explore the M10 software libraries
5) When the debug is done, you can program your sketch to the MAX10 FPGA and make it available after power on.

On the other hand, if you would like to customize the hardware, modify the existing design or come up with your own, you can do the following: (as shown in the left part of Figure 1-6)

1) Write your RTL code and simulate them
2) Synthesize your code in Intel Quartus Prime and convert the .sof file into binary files for device programming (It is suggested to use the project of M10 as a template, as it contains .tcl scripts to automatically convert .sof to binary format.)
3) Program the FPGA through M10_config_gui utility (open source tool, free download from GitHub)

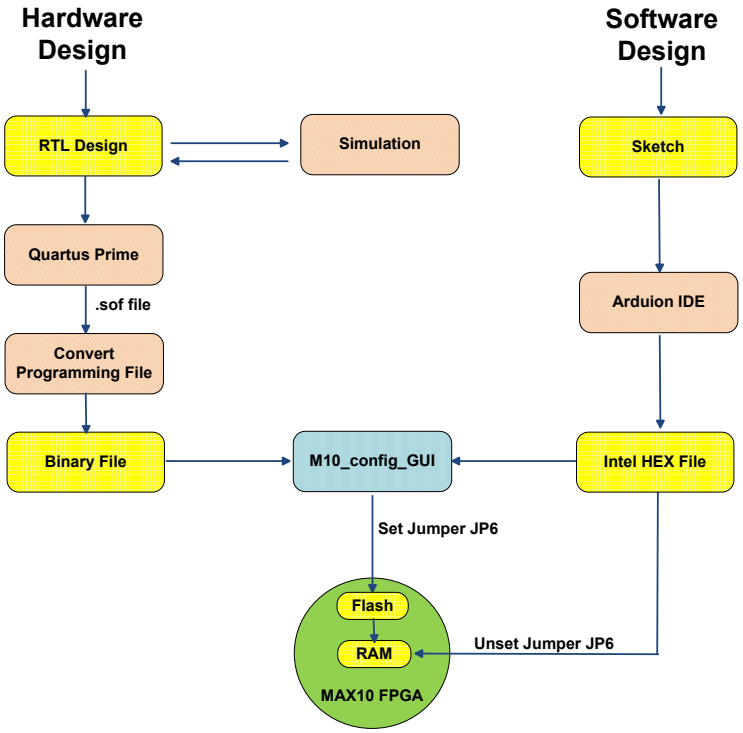


Figure 1-6 Design Flow for PulseRain M10

2 Getting Started with Software Design

This section is the detailed elaboration of the software design flow illustrated in Figure 1-6.

2.1 Connect the Board

Out of the box, the M10 board carries a FPGA image of a soft-core MCU running at 96MHz. And it also supports an Arduino compatible software interface. Assuming a Windows PC platform is used, you need to do the following to connect your PC to the M10 board

1) Power up Your PC

2) Use a micro USB cable to connect the M10 board to your PC. The factory image of M10 board already has some firmware code loaded. And you should see the blue LED flashing after you connect M10 board to PC.

3) The first time you connect your M10 board to the PC, Windows may prompt a message dialog for driver installation, like the one shown in Figure 2-1.

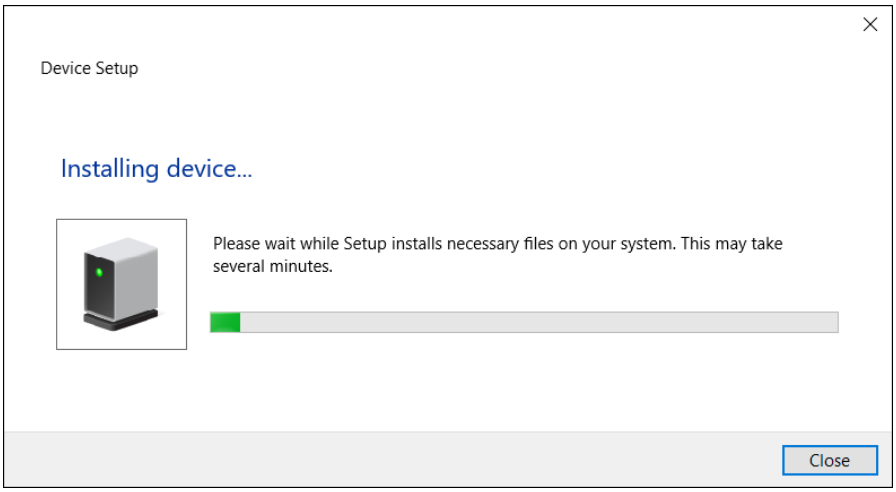


Figure 2-1 Windows Driver Installation

If for some reason, Windows could not automatically locate the driver for you, you might have to install it manually. The PulseRain M10 boards uses FTDI FT232RL chip for USB/COM port, and its driver can be found at FTDI website:
http://www.ftdichip.com/FTDrivers.htm

4) If the Windows driver is successfully installed, you should be able to find a new COM port in Windows Device Manager, as shown in Figure 2-2.

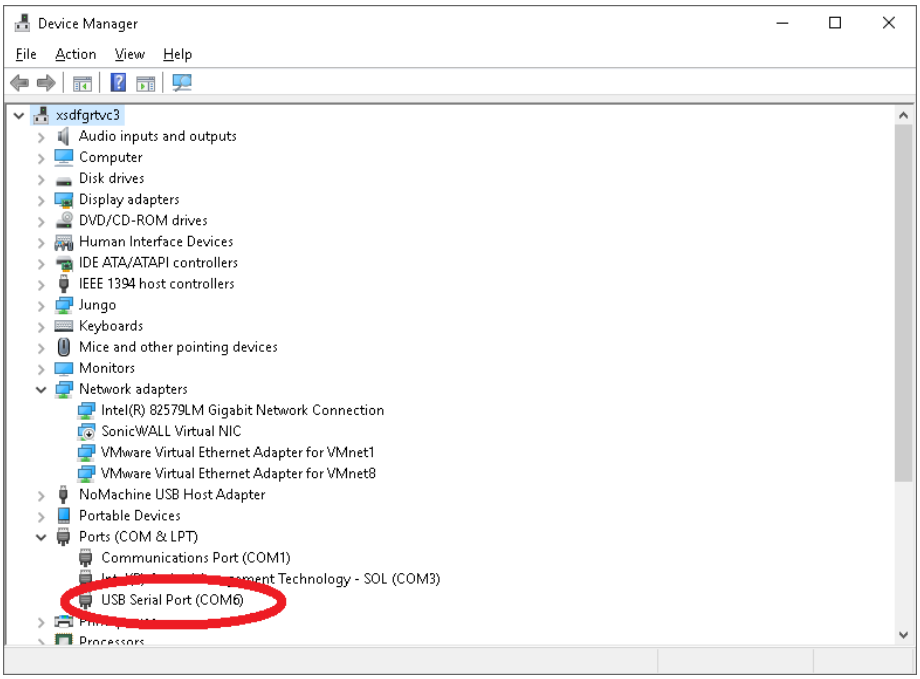


Figure 2-2 USB Serial Port in Windows Device Manager

2.2 Install Arduino IDE

1) After the Windows driver is successfully installed, the next step is to install Arduino IDE. On Windows 10, the Arduino IDE can be installed as an App directly from Windows App store. Otherwise, the Windows installer for Arduino IDE can be found at Arduino Website: https://www.arduino.cc/en/Main/Software

2) After the Arduino IDE is installed, launch it and click the menu File / Preferences, as shown in Figure 2-3:

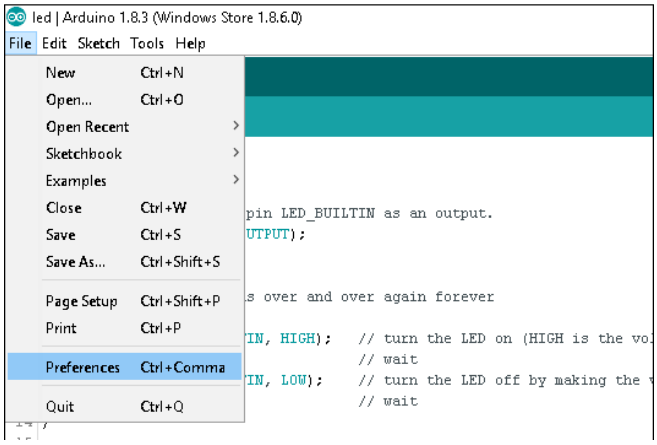


Figure 2-3 Arduino IDE, Preferences Menu

3) The File / Preferences menu will bring out a dialog like the one shown in Figure 2-4. Please set the "Additional Boards Managers URL" to https://raw.githubusercontent.com/PulseRain/Arduino_M10_IDE/master/package_M10_index.json (If this input box is not empty, use semicolon to separate multiple URLs.) And click OK to close the dialogue.

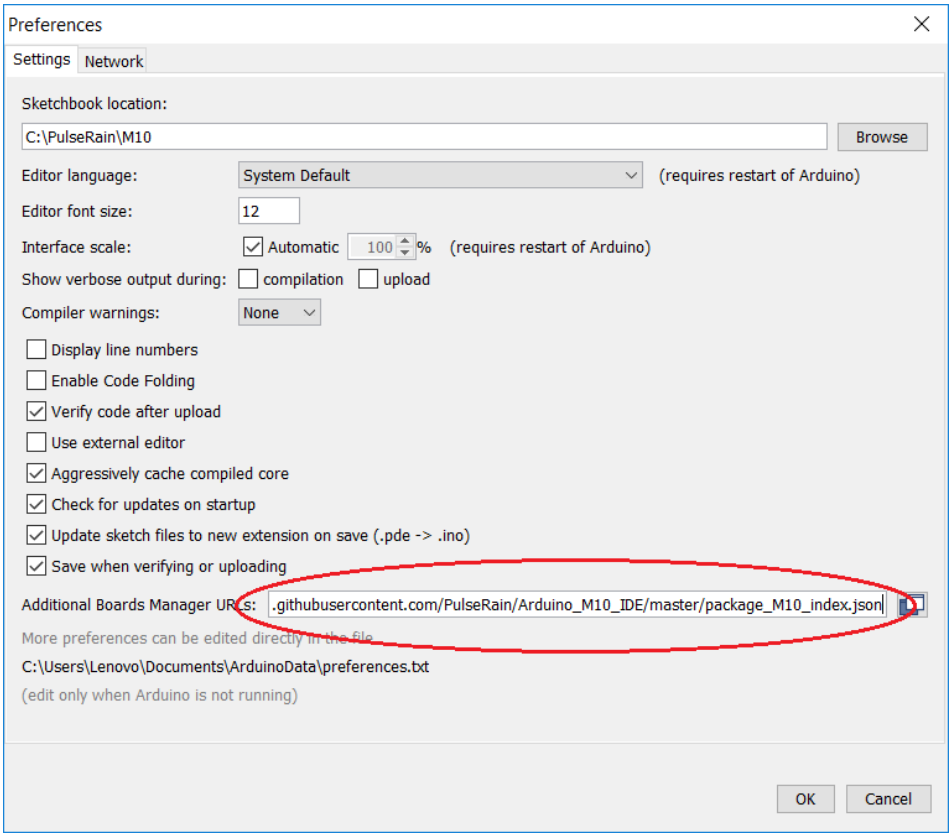


Figure 2-4 Arduino IDE, Preferences Dialogue

4) Now click the menu Tools / Boards / Boards Manager, as shown in Figure 2-5.

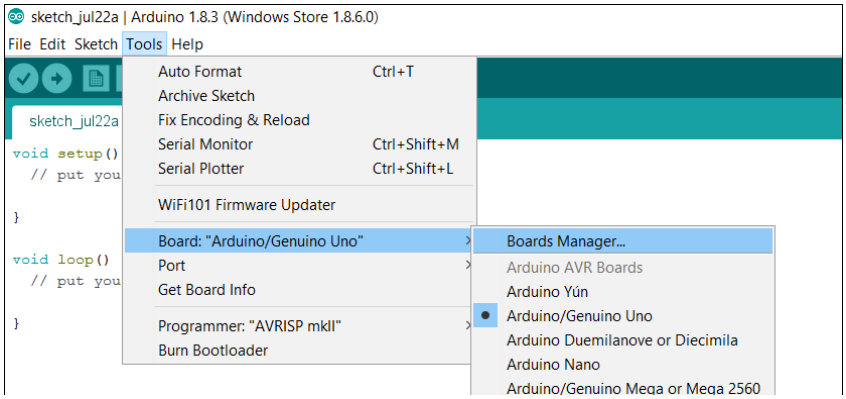


Figure 2-5 Arduino IDE, Boards Manager Menu

5) The "Boards Manager" will bring out a dialogue like the one shown in Figure 2-6. Type in "M10" in the search box to find the board support package for PulseRain M10 board, as illustrated in Figure 2-7. Click "Install" to download and install the board support package.

Figure 2-6 Arduino IDE, Boards Manager Dialogue

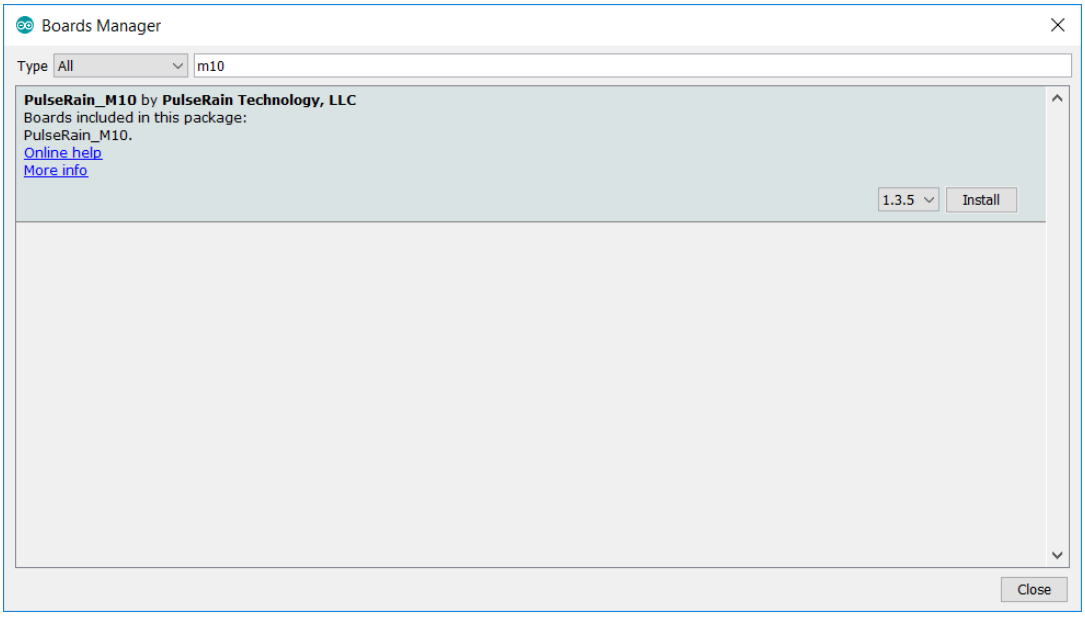


Figure 2-7 Arduino IDE, Boards Support Package for PulseRain M10

2.3 Write Sketches

Just like Arduino, you need to select the correct COM port and Board Name before you can start writing sketches.

To select the COM port in Arduino IDE, click the menu Tool / Port, as shown in Figure 2-8. The COM port for M10 board is usually the one that has the biggest index number, but not always. If you have trouble determining which COM port corresponds to the M10 board, you can always open the device manager to check, as illustrated in Figure 2-2.

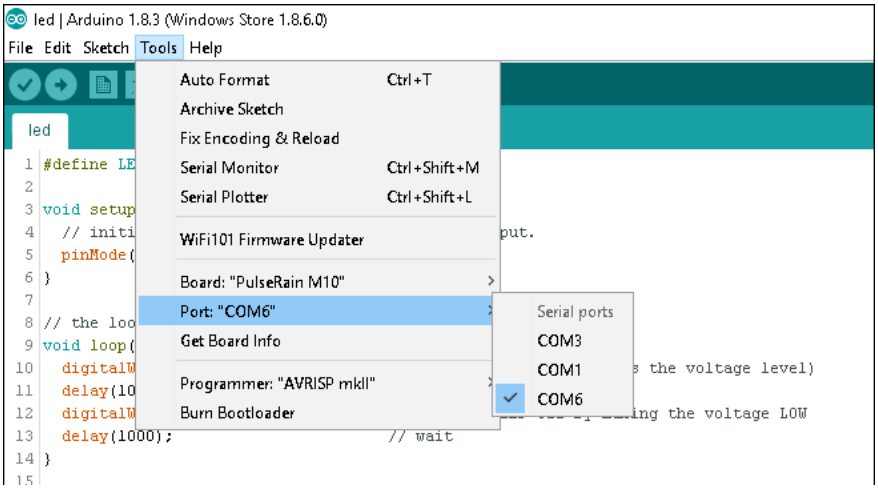


Figure 2-8 Arduino IDE, Select COM port

To select the board name as "PulseRain M10", click the menu Tool / Board. If the previous steps were done right, you should see the name "PulseRain M10" somewhere close to the bottom of the menu, as illustrated in Figure 2-9.

And now you can start writing your sketches. It is suggested that you try the sketch in List 2-1 first to turn on/off the LED:

```
#define LED_BUILTIN 13

void setup()  {
  pinMode(LED_BUILTIN, OUTPUT);
}

void loop()  {
  digitalWrite(LED_BUILTIN, HIGH);
  delay(1000);
  digitalWrite(LED_BUILTIN, LOW);
  delay(1000);
}
```

List 2-1 Sketch to Test LED

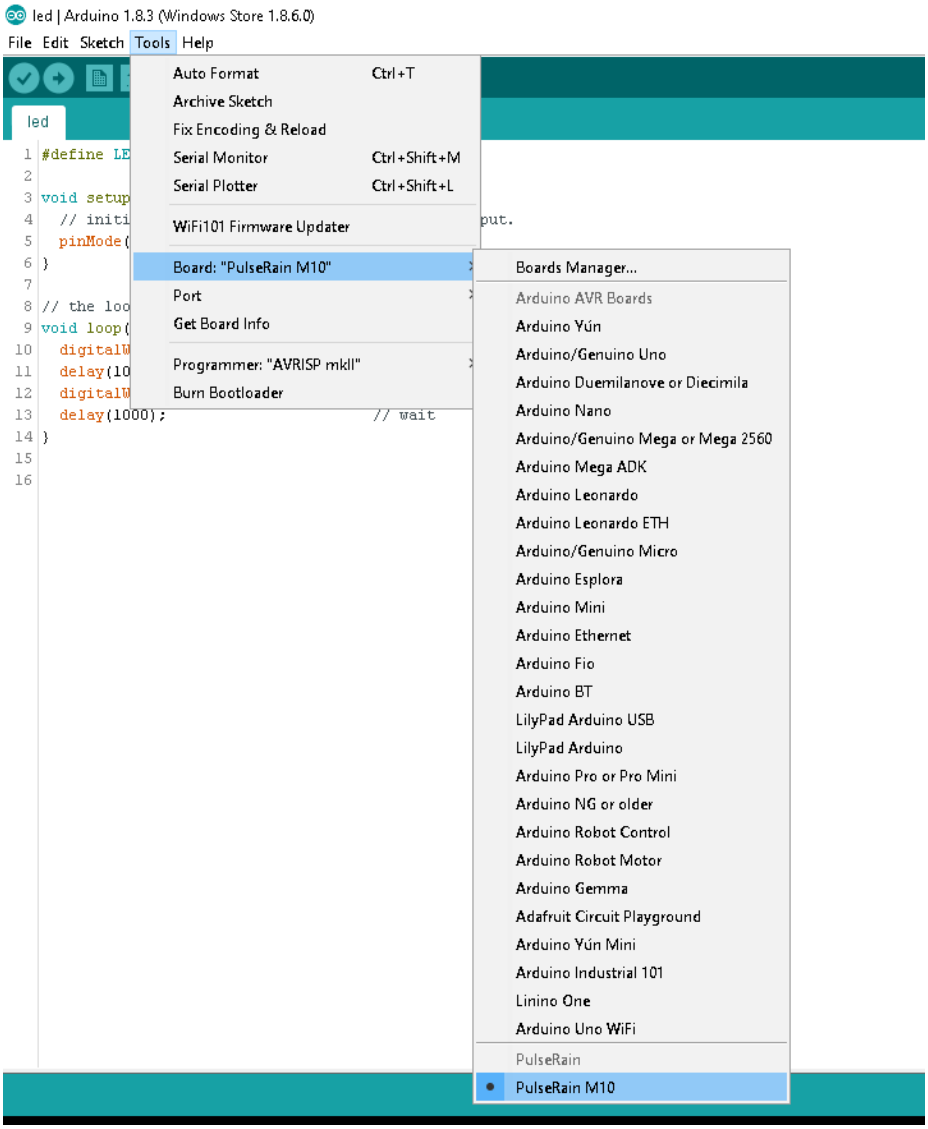


Figure 2-9 Arduino IDE, Select PulseRain M10 board

After you copy/paste List 2-1 into Arduino IDE, you can type in "ctrl-U" (or menu Sketch/Upload) to compile and upload the sketch to M10 board. And it is also recommended to turn on the option of "Show Verbose Output" in Preferences Dialogue (Menu File / Preferences), as illustrated in Figure 2-10. In this way, the path of the .hex file can be located through the verbose output. And the .hex file is needed if you want to program your sketch into MAX10 FPGA's onchip flash memory.

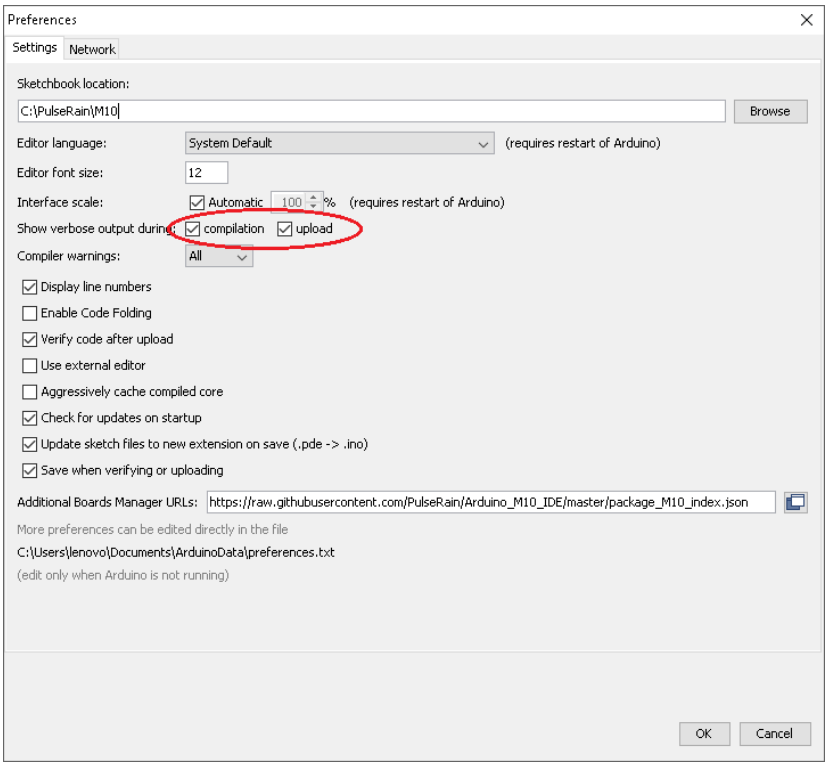


Figure 2-10 Turn on Verbose Option

2.4 Program the sketch into Flash Memory

If the previous steps were done right, you should be able to see the green LED flashing on and off after you upload the sketch in List 2-1. And at the bottom of the Arduino IDE windows, you are supposed to see something like the following (List 2-2), assuming you've turned on the option of "Show Verbose Output" (Figure 2-10).

```
Sketch uses 6340 bytes (19%) of program storage space. Maximum is 32768 bytes.
Global variables use 618 bytes (7%) of dynamic memory, leaving 7574 bytes for local variables. Maximum is 8192 bytes.
C:\Users\PulseRain\Documents\ArduinoData\packages\PulseRain_M10\tools\M10_upload\1.2.0/FP51_upload -pFP51 -cUART -PCOM6 -b921600 -UC:\Users\PulseRain\AppData\Local\Temp\arduino_build_469503/led.ino.eep
================================================================================
# Copyright (c) 2017, PulseRain Technology LLC
# FP51 Code Upload Utility, Version 1.2
================================================================================
baud_rate  =  921600
com_port   =  COM6
image file =  C:\Users\PulseRain\AppData\Local\Temp\arduino_build_469503/led.ino.eep
================================================================================
CPU paused
CPU reset ...
```

```
Loading... C:\Users\PulseRain\AppData\Local\Temp\arduino_build_469503/led.ino.eep
Writing | ########################## | 100% 1.22s

CPU reset ...
Done:  6209  Byte(s)
CPU is running
```

List 2-2 Arduino IDE Verbose Output

If you are satisfied with the outcome, you could choose to program your sketch into MAX10 FPGA's onchip flash, so that your sketch can function right after power up, without connecting to PC.

To program the sketch into flash memory, do the following:

1) Power off the M10 board by removing the USB connection
2) Put a jumper on JP6, as illustrated in Figure 2-11.

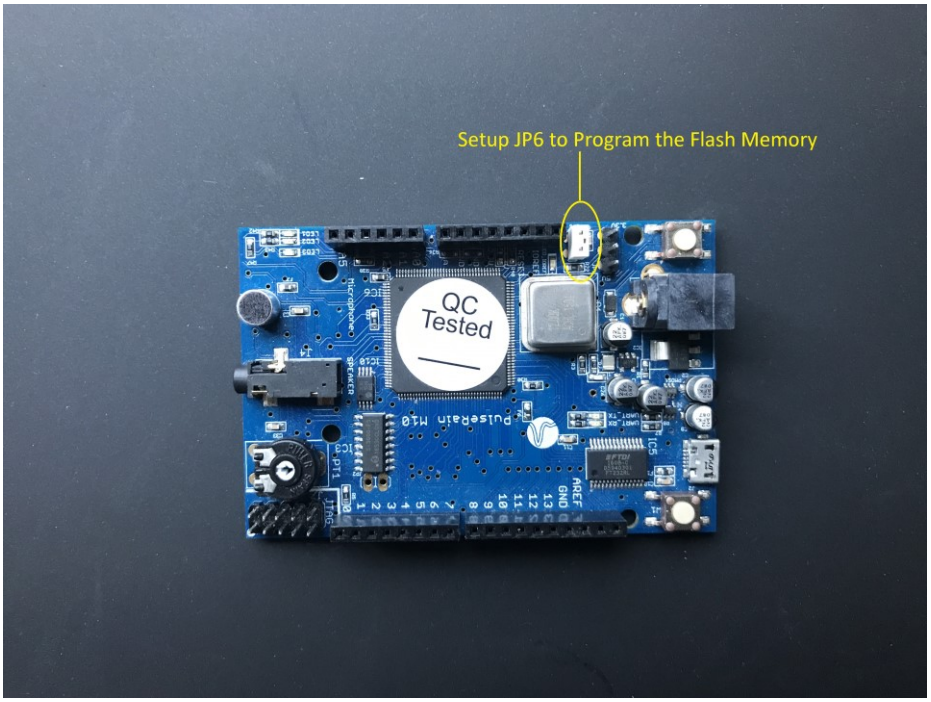


Figure 2-11 Set JP6 to Program the Flash Memory

3) Reconnect the USB cable to PC, and open the utility "M10 High Speed Config Utility"(M10_config_gui). The source code of this utility can be found in
https://github.com/PulseRain/M10_high_speed_config_software
And its latest binary can be downloaded from
https://github.com/PulseRain/M10_high_speed_config_software/raw/master/bin/M10_config_gui.exe

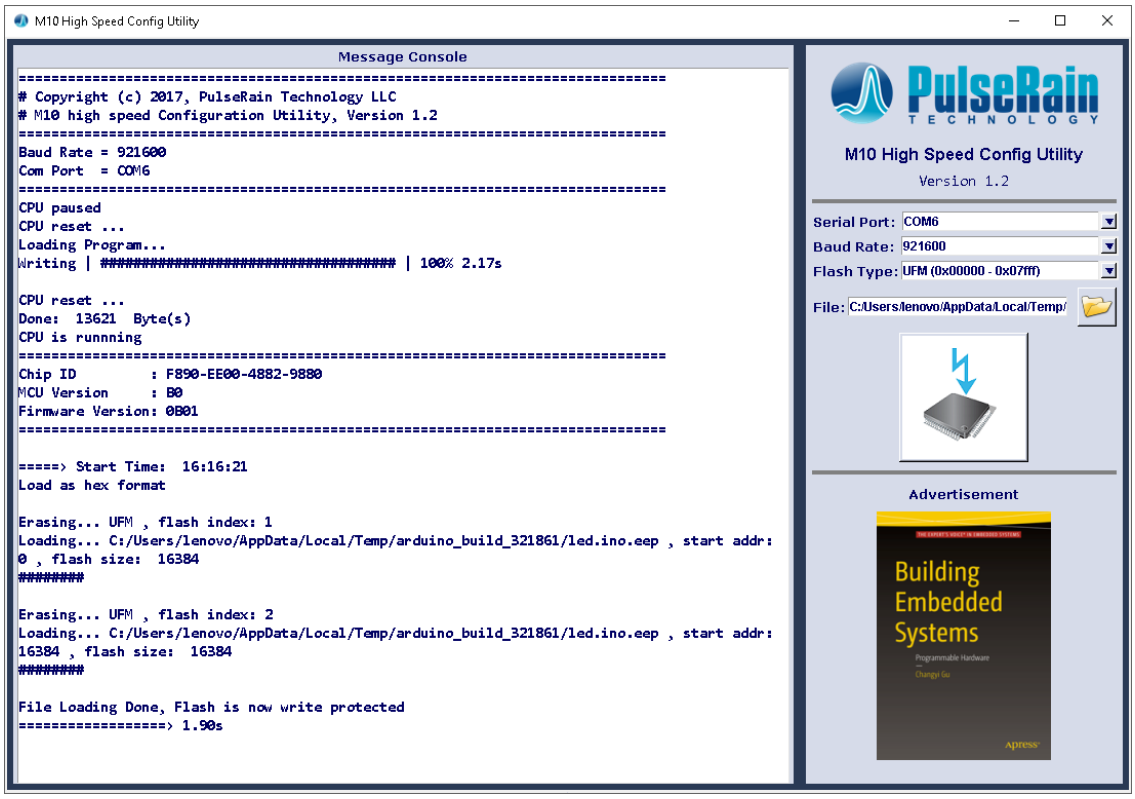


Figure 2-12 M10 High Speed Config Utility

4) After the utility is launched, setup the Serial Port, baud rate accordingly, and point File to the hex file produced by Arduino IDE. Such information can be found in the verbose output in Arduino IDE. In List 2-2, it something like

    baud_rate = 921600
    com_port = COM6
    image file = C:\Users\PulseRain\AppData\Local\Temp\arduino_build_469503/led.ino.eep

    And set the Flash Type to be UFM.

5) Click the big square button on the right-hand side (with the picture of lightning strike). If everything is done right, you see a message like "***File Loading Done, Flash is now write protected***." at the end of flash programming, as illustrated in Figure 2-12.

6) Power off the M10 board again by removing the USB connection. And remove the jumper on JP6.

7) Power up the M10 board again. And your sketch is supposed to be loaded and executed automatically at this point.

2.5 Install and Use M10 Library

For those who are interested in writing more advanced sketches, PulseRain Technology provides the M10 library to smooth the design process. To install those libraries, click Menu Sketch / Include Library / Manage Libraries ..., as illustrated in Figure 2-13.

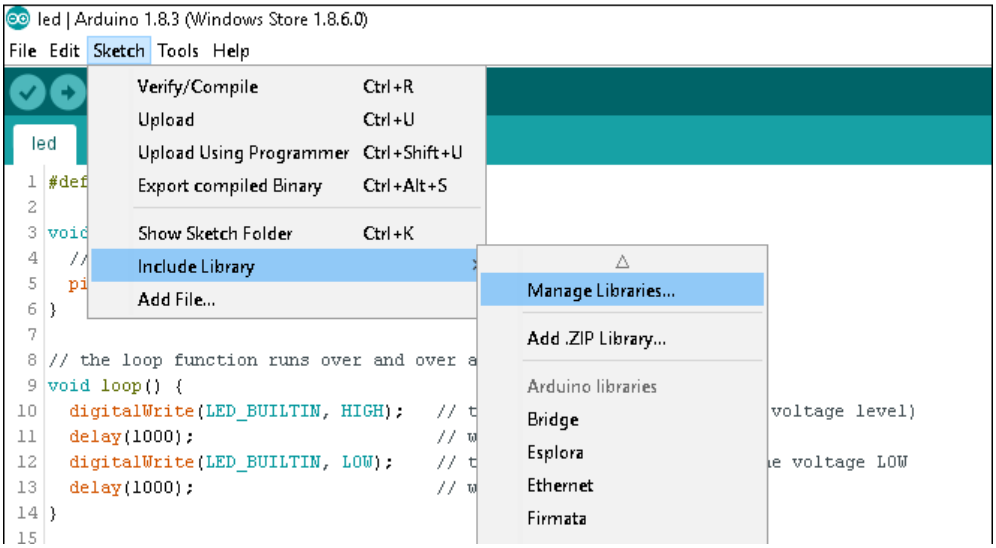


Figure 2-13 Arduino Manage Libraries Menu

This will bring out a dialogue like the one shown in Figure 2-14. Type in "PulseRain M10" in the search box, and install those libraries as you see fit (Figure 2-15).

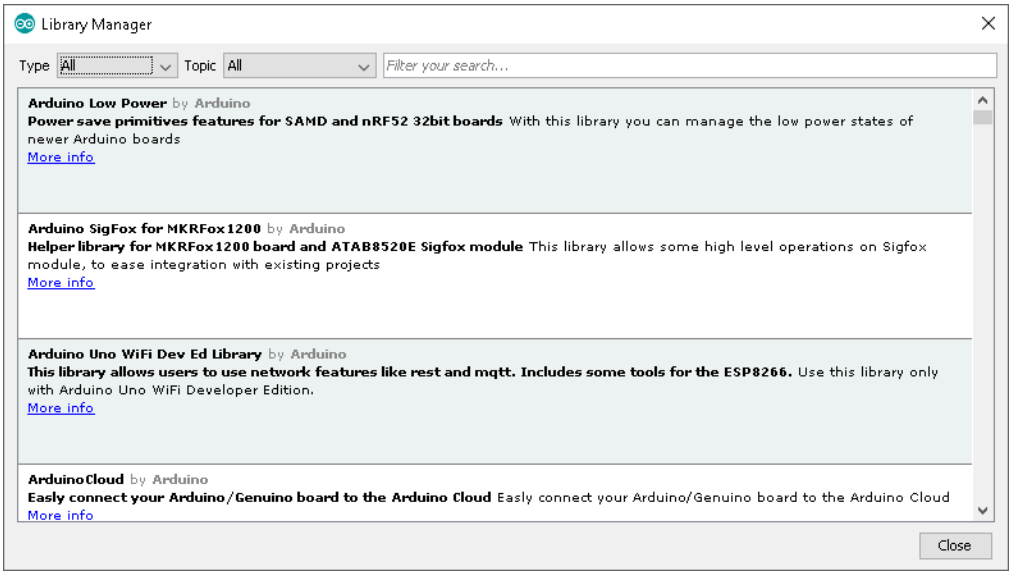


Figure 2-14 Arduino Library Manager

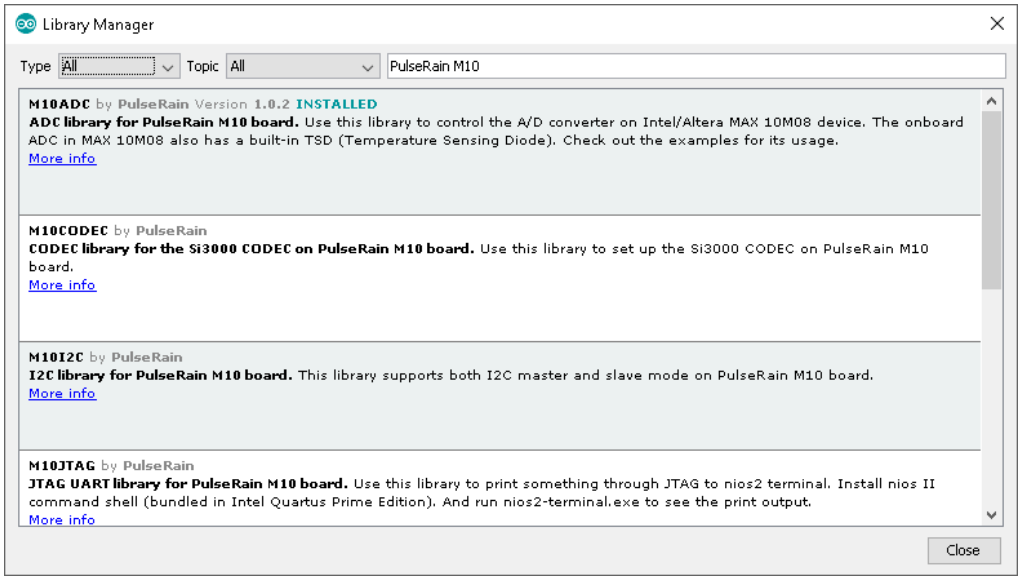


Figure 2-15 Search for M10 Library in Arduino Library Manager

Just like Arduino, to use those libraries in your sketch, simply include the correspondent head file. For example, to include PWM library, just add ***#include "M10PWM.h"*** to the top of your sketch, and then start calling M10PWM APIs in your sketch.

3 Getting Started with Hardware Design

This section is the detailed elaboration of the hardware design flow illustrated in Figure 1-6.

3.1 Repository for PulseRain M10 RTL Design

The phase one release of the M10 board contains a soft-core MCU (enhanced 1T 8051 processor core, with a rich set of peripherals) that runs at 96MHz. And its correspondent source code can be found on Github: https://github.com/PulseRain/Mustang

The project is code-named Mustang (named after the legendary FP-51 Mustang Fighter-bomber.). To get the source code, do the following:

```
>> git clone https://github.com/PulseRain/Mustang.git

>> cd Mustang

>> git submodule update --init –recursive
```

3.2 Simulate the RTL Design

The Mustang project supports RTL simulation with Modelsim. And the following assumes Modelsim – Intel FPGA Starter Edition 10.5b is used. The Modelsim Starter Edition is part of the Intel FPGA Lite Edition, which can be downloaded from https://www.altera.com/downloads/download-center.html

To simulate the Mustang project, open the Modelsim Starter Edition, and set the active path to the Mustang working copy that was cloned in the previous section.

Now enter the sim directory, and build the whole Mustang project by executing the do file "build_soc.do".

```
ModelSim> cd sim
ModelSim> do build_soc.do
```

To start the simulation, execute the "run_soc.do", setup the wave windows through "wave.do", and run for 5ms.

```
ModelSim> do run_soc.do
ModelSim> do wave.do
ModelSim> run 5ms
```

If everything is done right, the wave window may look like the one in Figure 3-1.

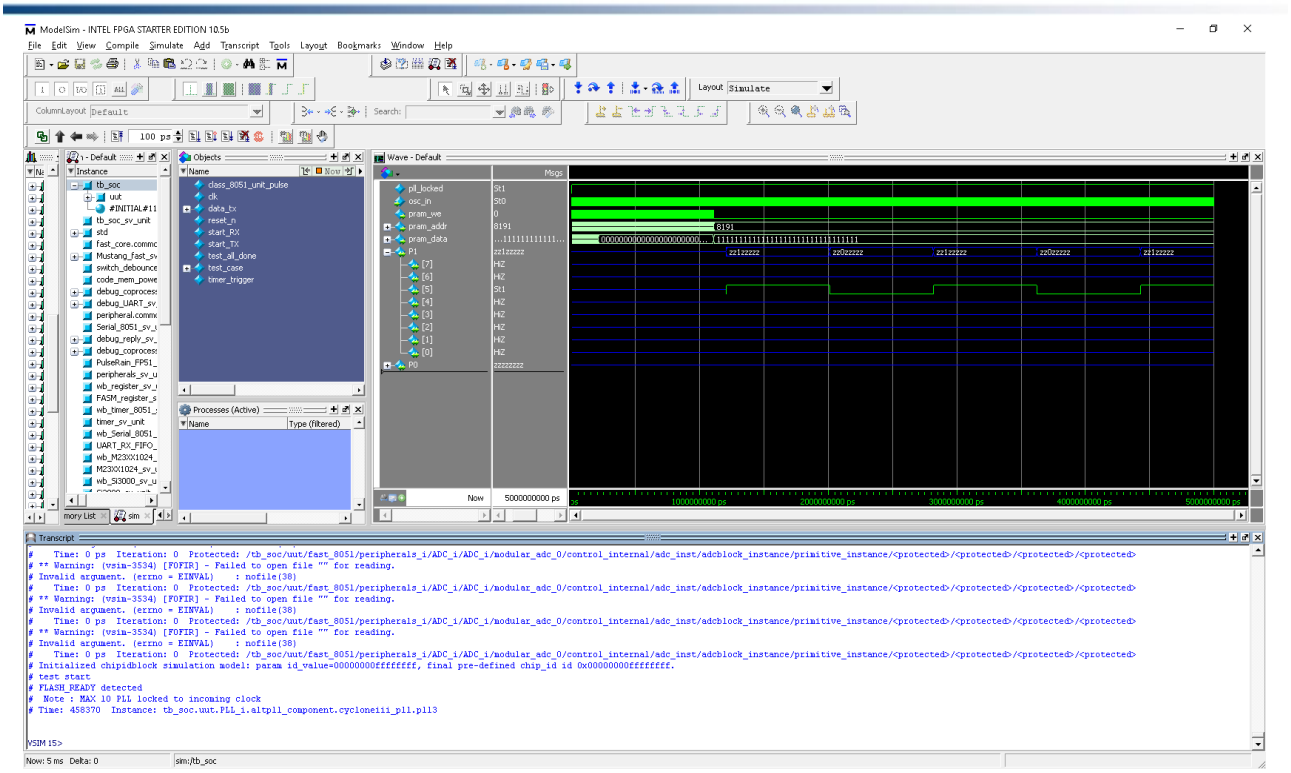


Figure 3-1 Mustang Simulation

The default test bench will simulate the case after power on reset, where the following will happen:

1) It takes some time (less than 1 micro-second) for the PLL to get locked

2) The onchip_flash core will load the firmware image into the code RAM for execution. In simulation, the onchip_flash core will look for a file named " **altera_onchip_flash.dat**" as its flash content (32KB). This file can be generated by the Python script "**hex2data.py**", which can also be found in the sim folder.

   To generate your own "altera_onchip_flash.dat", run the "hex2data.py" as following:

   ```
   python hex2data.py hex_file_name 8192
   ```

   The hex file can be the one generated by Arduino IDE (See Section 2.4 for locating the hex file), and the number 8192 above actually means the size of the onchip UFM (user flash memory), which is 32768 KB for MAX10 10M08SAE144C8G device.

3) The process of loading code from UFM to RAM can be monitored by observing the **pram_we**, **pram_addr** and **pram_data** signals.

4) After the code loading process is done, the code will be executed from address zero. In the default simulation, the sketch that is executed is the same as the one shown in List 2-1, except the delay is reduced from 1000 millisecond to 1 millisecond. The decrease of delay is simply to speed up the simulation so that signal toggle can be observed within 5ms. The source code and hex file for the simulated sketch can be found in **sim/led.ino** and **sim/led.hex**

5) In the simulation, the PIN 13 on Arduino connector is actually mapped to Bit 5 of Port P1. It's toggling can be observed after 1.2 ms

3.3 Build the FPGA Image

To generate the FPGA image, Intel Quartus Prime software is needed. The following assumes Intel FPGA 16.1.0.196 Lite Edition is used.

To build the FPGA image, simply open the **synth/Mustang_fast.qpf** project in Quartus Prime, and click "Start a new compilation", as illustrated in Figure 3-2.

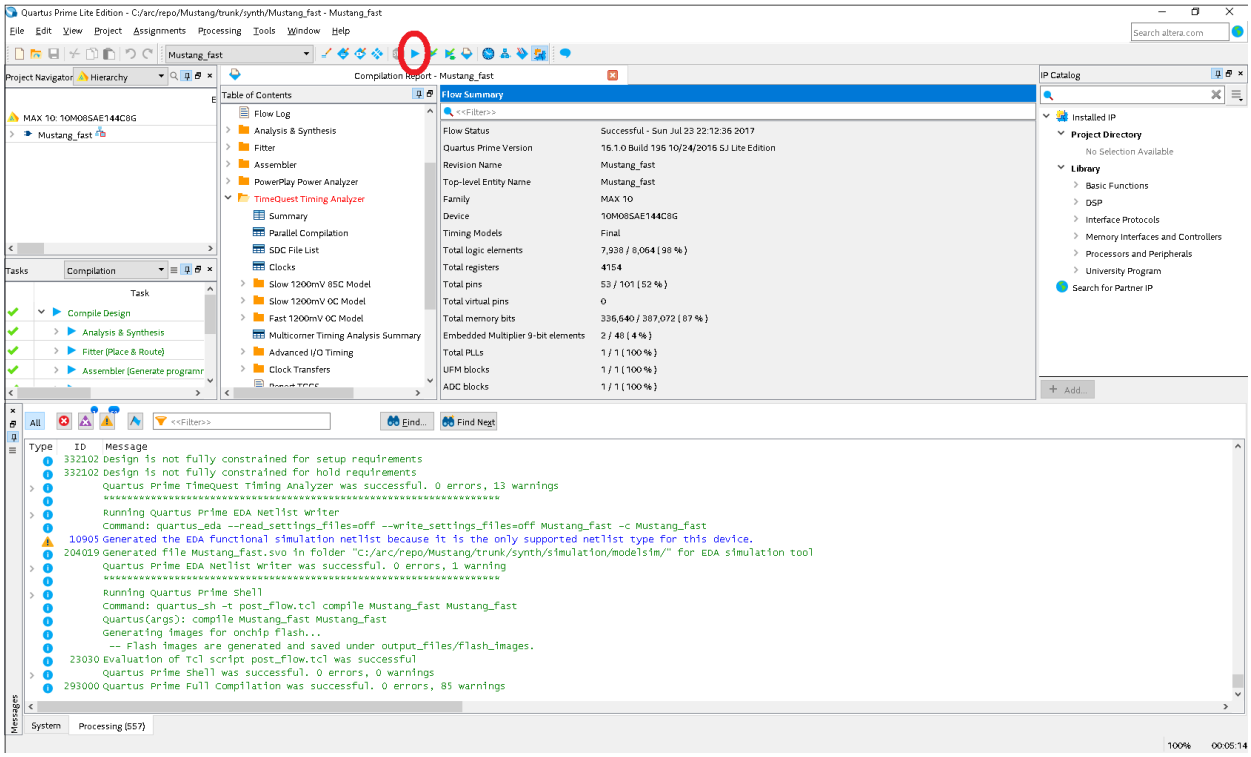


Figure 3-2 Building Mustang Project

The Mustang project has built-in script (post_flow.tcl) for file conversion. After the build is done, you can find the converted binary file and .pof file in **synth/output_files/flash_images**

Like all projects in Quartus Prime, the Mustang project uses a seed for random number generation. And these random numbers will be used to guide the Place and Route process during fitting. It is discovered that the random number generators are different between Windows and Linux platforms. If you have trouble closing time on your platform, you can try different seeds in Menu

Assignments / Setting / Compiler Settings / Advanced Settings (Fitter) / Fitter Initial Placement Seed

3.4 Program the FPGA image

The procedure to program the FPGA image is almost the same as that mentioned in Section 2.4. And a copy of the M10_config_gui can also be found in synth/output_files.

To program the FPGA image, please set the Flash Type to **CFM**, point the File location to **synth/output_files/flash_images/cfm.bin**, and click the big button on the right-hand side to start the program process.