**Programming ATmega8 using Arduino IDE**

Installing Arduino bootloader on ATmega8 AVR microcontroller and programming it using Arduino IDE.

**Part one: Installing Arduino Bootloader**

In this part, we add ATmega8 support to Arduino IDE. Then we set up the breadboard and connect ATmega8 to Arduino UNO board.

**Step One: Adding ATmega8 Support to Arduino IDE Using Board Manager**

There are different types of cores available on MCUDude’s GitHub repository. MiniCore is used for ATmega8 microcontroller. Supported microcontrollers in this core are:

ATmega8

ATmega48

ATmega88

ATmega168

ATmega328

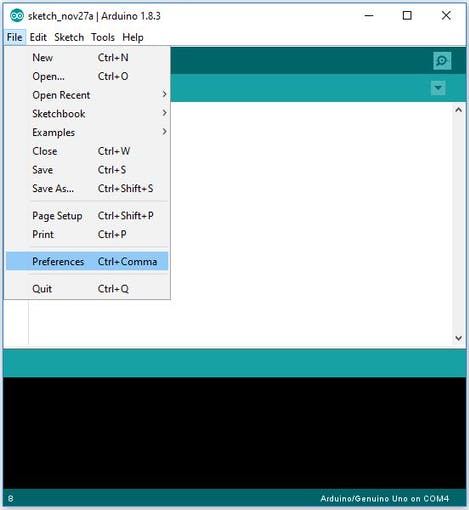
Follow steps below to install MiniCore in your Arduino IDE.

* In File menu, click on Preferences.
* Now in Additional Boards Manger URLs, enter the following URL:

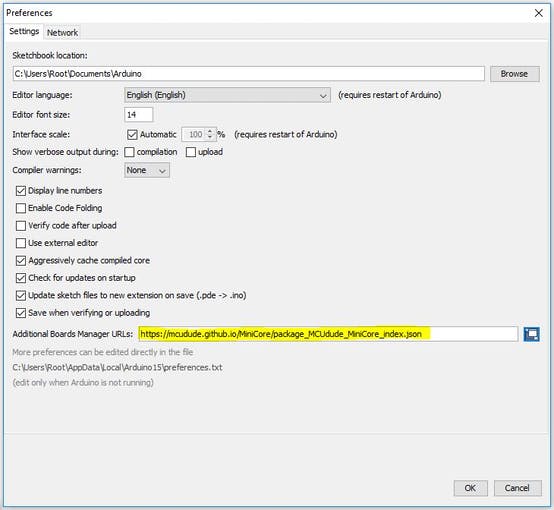
<https://mcudude.github.io/MiniCore/package_MCUdude_MiniCore_index.json>

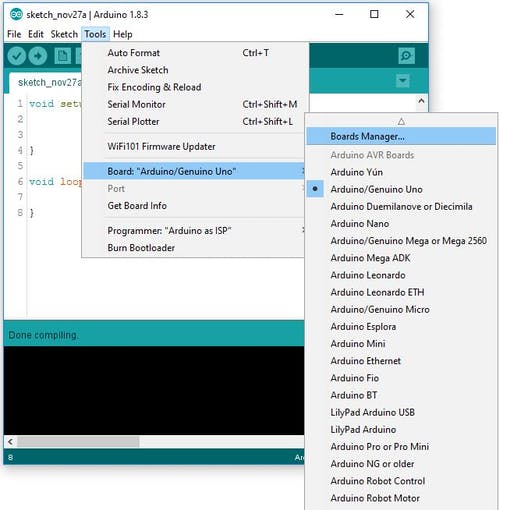
* Go to Tools menu and then select Board > Boards Manager
* In Boards Manager window, search for MiniCore and then install the latest version.

The above steps are shown graphically here:

[](javascript:openLightBox('9516a663e6',%200);)

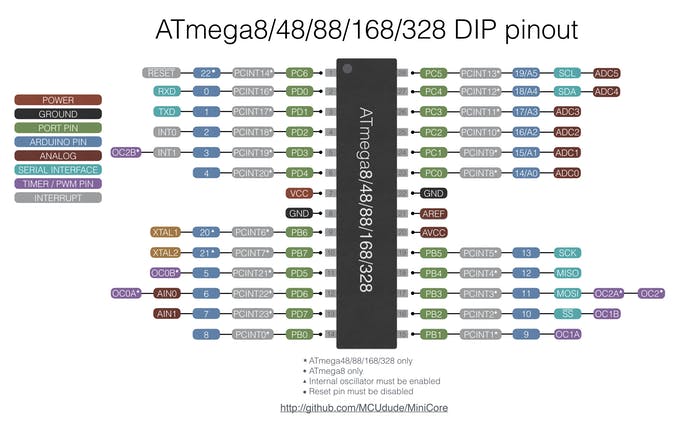
Adding ATmega8 support to Arduino IDE using board manager (installing MiniCore)

[](javascript:openLightBox('9516a663e6',%201);)

[](javascript:openLightBox('9516a663e6',%202);)

Adding ATmega8 support to Arduino IDE using board manager (installing MiniCore)

ATmega8 pinout is shown in the following figure.

[](javascript:openLightBox('7877b422df',%200);)ATmega8 DIP pinout

### Step Two: Programming Arduino as an ISP (In-system programming)

### Bootloader: The AVR Bootloader allows the programming or re-programming of the target AVR microcontroller using the PC serial port instead of a traditional programmer. Once the AVR Bootloader is programmed into the microcontroller, it remains until the chip is erased.

To burn Arduino bootloader, we need to make our Arduino UNO as an ISP. There’s a sketch named ArduinoISP in the built-in examples of Arduino IDE. Follow these steps to program your Arduino UNO as an ISP.

Open **File** > **Examples** > **11.ArduinoISP** > **ArduinoISP**

Upload this sketch to your Arduino UNO.

### Step Three: Burning Bootloader

* Connect Arduino to ATmega8 as below.

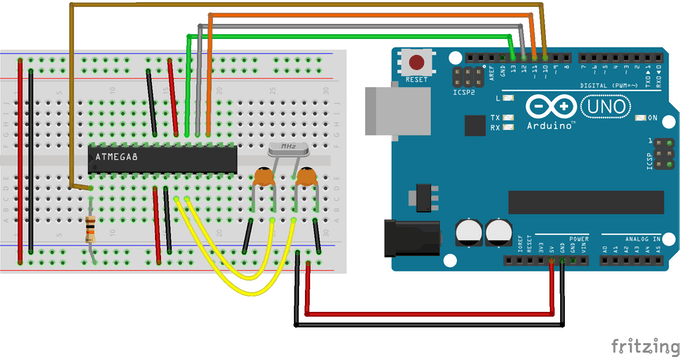
**Arduino ----------ATmega8**

SCK / Pin 13 ----------SCK / PB5

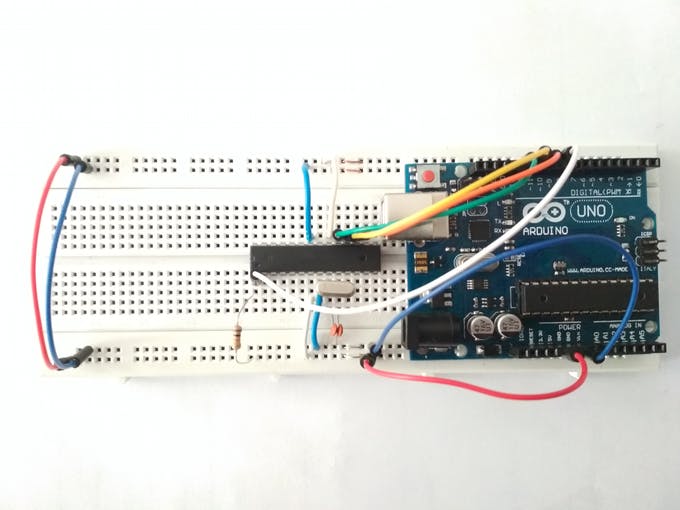
MISO / Pin 12 ----------MISO / PB4

MOSI / Pin 11 ----------MOSI / PB3

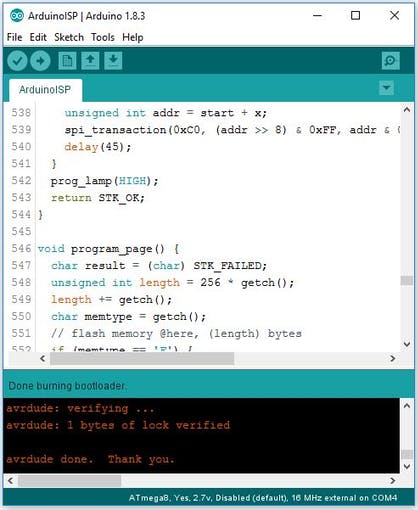
SSN / Pin 10 ----------RESET

[](javascript:openLightBox('577b43e5fa',%200);)

Burning Arduino bootloader

[](javascript:openLightBox('577b43e5fa',%201);)

* Go to **Tools** > **Board** and select **ATmega8**. You can also select your clock type and frequency in **Tools** menu.
* Select programmer type in **Tools** > **Programmer:** as “**Arduino** **as ISP**”.
* Now in **Tools** menu, click on the **Burn Bootloader**.

[](javascript:openLightBox('c6eaec4d2e',%200);)

Successful bootloader burning. If the operation was successful, it says “Done burning bootloader”.

### Part Two: Programming ATmega8 Using Arduino IDE and a USB to TTL Convertor

Here we have an ATmega8 with Arduino bootloader. As you know, there is a USB to TTL convertor on all Arduino boards. It acts as a bridge between microcontroller and Arduino IDE. We use CH340 USB to TTL convertor module to upload the sketch to ATmega8.

### Step Four: Setting up the Connections

Connect components as shown below.

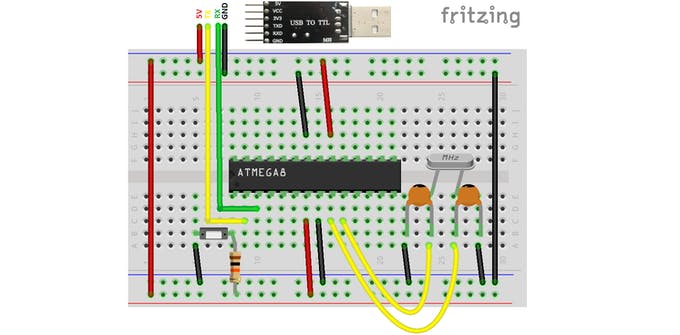
**CH340 ---------- ATmega8**

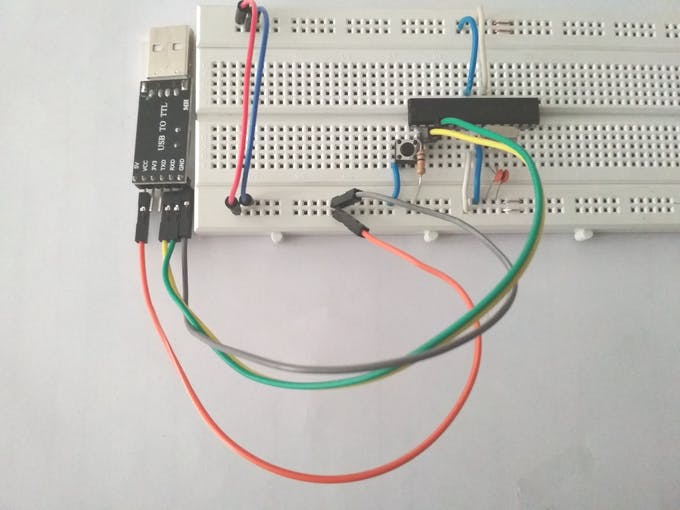
VCC ---------- VCC

GND ---------- GND

Tx ---------- RX / PD0

Rx ---------- Tx / PD1

[](javascript:openLightBox('e782627a56',%200);)Programming ATmega8 using Arduino IDE and a USB to TTL module

[](javascript:openLightBox('e782627a56',%201);)

### Step Five: Uploading the Sketch to ATmega8

Press and hold the reset pushbutton. Now click on upload in IDE. Hold the pushbutton until it says “Uploading…”on the IDE status bar. Release the button after uploading process begins. It’s because microcontroller should be in the RESET state when the uploading process starts.

Note: If the process was unsuccessful, place a 100nF capacitor at Vcc and GND of microcontroller as close as possible.

Note: Some USB to TTL convertor modules have a pin named DTR. If yours has this pin too, you can connect it to the RESET pin of ATmega8 and there’s no need to use the pushbutton to manually reset the microcontroller.

## SCHEMATICS

##### **Installing Arduino Bootloader on ATmega8**

