Introduction

In the pursuit for small form factor, flexibility and functionality comes **Seeeduino Mega** – derived from Arduino Mega it harnesses the power of ATmega1280 to spice up your project building experience.

Model:ARD121D2P

Media:aa.pdf

Features

- ATmega 1280 @ 16MHz
- Selectable 5V/3.3V operation
- 70 Digital IO
- 16 Analog inputs
- 14 PWM outputs
- 4 Hardware serial ports (UART)
- Compatible with most Arduino Duemilanove and Diecimila Shields
- Small form factor, 30% smaller than Arduino Mega
- Easy to program, no additional hardware is required to load firmware – just plug to a USB port and you’re good to go.
- ICSP Header
- Can be powered through a battery or through a AC to DC adaptor

Application Ideas

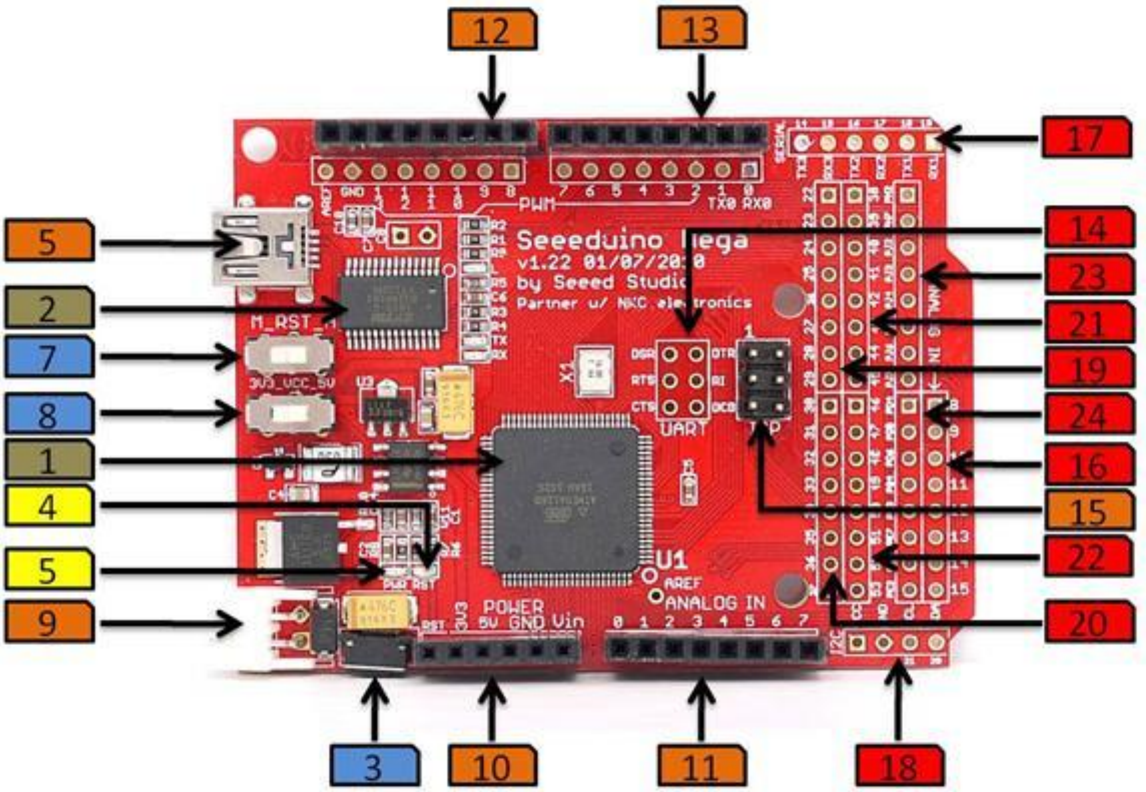
- Led display/LCD controller
- Pulse width modulation driver
- Robot controller
- Data acquisition systems
- Alarm systems
- Programmable logic controllers
- Embedded Webservers
- Control Systems

Cautions

Seeeduino Mega and other products of Seeed Studio Inc are not designed, intended or authorized for use in life support devices or systems. Life support devices or systems include, but are not limited to, surgical implants, medical systems, and other safety-critical systems in which failure of Seeed Studio Inc products could cause personal injury or loss of life. Should buyer purchase and use Seeed Studio Inc products in such an unauthorized and unintended manner, Buyer agrees to indemnify and hold harmless Seeed Studio Inc, its officers, employees, suppliers, affiliates, and distributors from any and all claims arising from such use, even if such claim alleges that Seeed Studio Inc was negligent in the design or manufacture of its product.

Schematic

The following diagram illustrates the placement of important integrated circuits, indicator LEDs, connectors common to all Arduino based development platforms and the ones specific to Seeeduino Mega.



NOTES

1. **Label [5]** The Seeeduino Mega uses a 4 Position mini-Type B Jack while the rest of the Arduino family uses standard 4 Position Type B Jack. The Arduino uses circular Power Jack.
2. **Label[9]** The Seeeduino Mega uses a 2 Pin JST connector for the power supply while the rest of the Arduino family uses coaxial power plugs.
3. **Label[3]** For the Reset button the Seeeduino Mega uses a an edge mounted push button while other Arduino based platforms uses a four legged tack switch .
4. Yellow Labels are indicator LEDs.
5. **Blue Labels** are mode selector and reset switches.
6. **Brown Labels** are items that are common to the Arduino family; this makes the Seeeduino Mega compatible with most Arduino shields.
7. **Red Labels** are the things that add the MEGA in the Seeeduino, the set includes additional ADC input pins and a bunch of bidirectional IO pins.

Specification

Key Specification

Operating Voltage	5V / 3.3V
Operating Temperature	-20~70℃
Input Voltage	6-20V

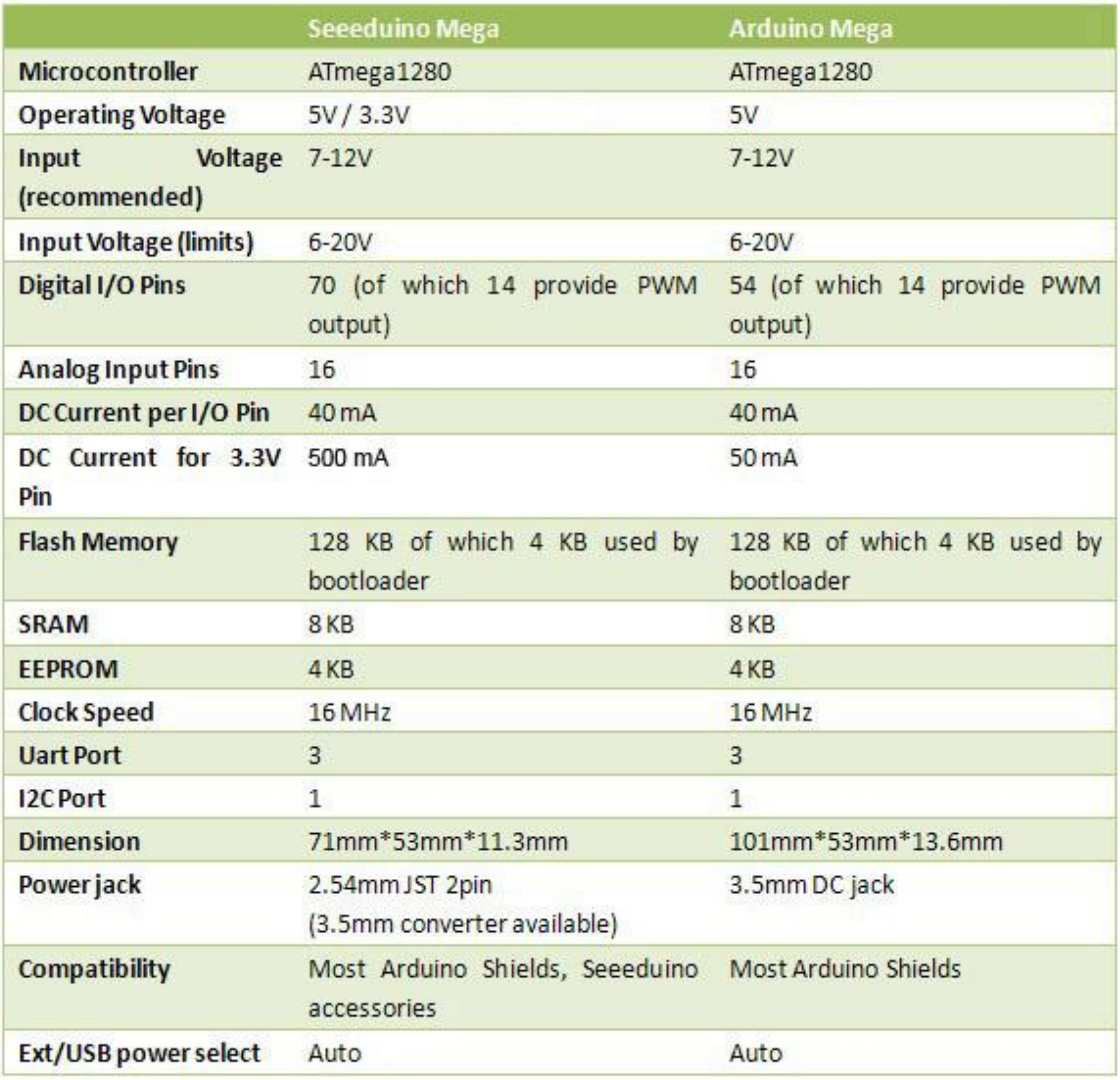
DC Current per IO pin	40mA
Flash Memory	128k
SRAM	8k
EEPROM	4k
Dimensions	71*53*11.3 mm
Power Jack	2.54mm, 2pin JST

Peripheral Features

Peripheral Features are multiplexed with General Purpose Input/Output [GPIO] Pins, for multiplexing and assignment please check Pin Definitions. For more details on the alternate pin functions please refer to the Atmega1280 datasheet.

Item	Min	Norm	Max
8-Bit Timers/Counters	2		
16-Bit Timers/Counters	4		
8-Bit Pulse Width Modulator (PWM)	4		
Programmable PWM Channels	12		
Programmable PWM Resolution (Bits)	2		16
Analog to Digital Input Channels (ADC)	16		
ADC Resolution (Bits)	10		
Programmable USART Channels	4		
On-Chip Analog Comparator	1		
Master/Slave SPI Interface	1		
Byte Oriented 2 -Wire Serial Interface	1		

comparison with Seeeduino Mega and Arduino Mega



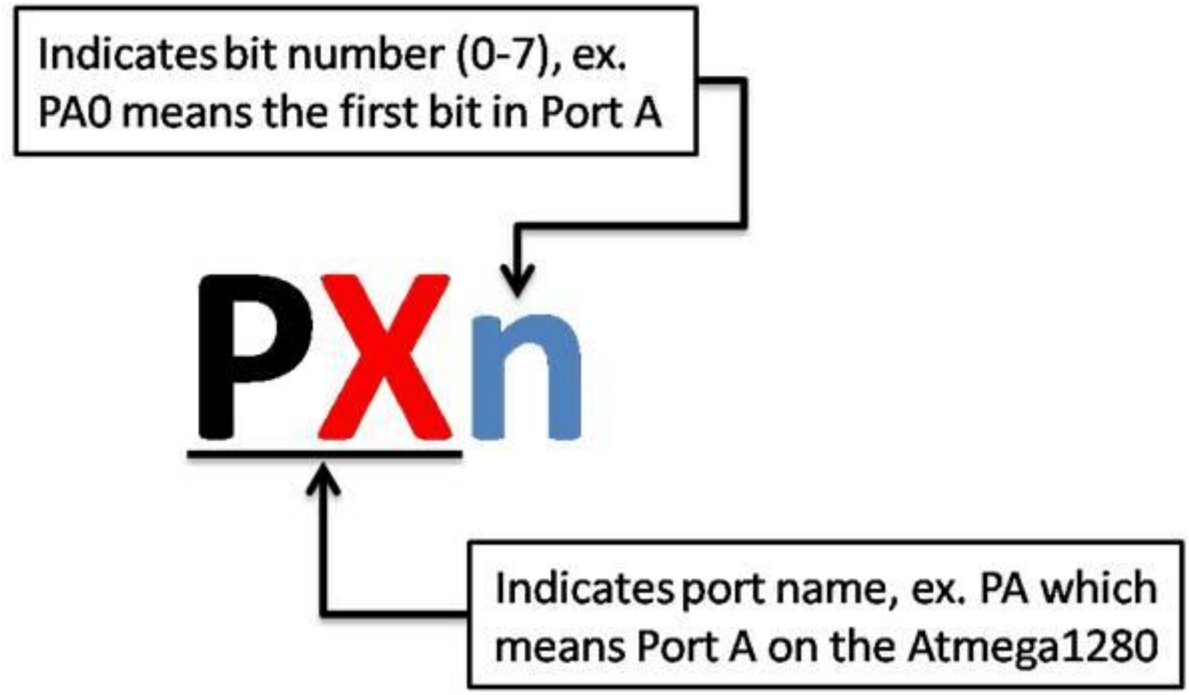
	Seeeduino Mega	Arduino Mega
Microcontroller	ATmega1280	ATmega1280
Operating Voltage	5V / 3.3V	5V
Input Voltage (recommended)	7-12V	7-12V
Input Voltage (limits)	6-20V	6-20V
Digital I/O Pins	70 (of which 14 provide PWM output)	54 (of which 14 provide PWM output)
Analog Input Pins	16	16
DC Current per I/O Pin	40 mA	40 mA
DC Current for 3.3V Pin	500 mA	50 mA
Flash Memory	128 KB of which 4 KB used by bootloader	128 KB of which 4 KB used by bootloader
SRAM	8 KB	8 KB
EEPROM	4 KB	4 KB
Clock Speed	16 MHz	16 MHz
Uart Port	3	3
I2C Port	1	1
Dimension	71mm*53mm*11.3mm	101mm*53mm*13.6mm
Power jack	2.54mm JST 2pin (3.5mm converter available)	3.5mm DC jack
Compatibility	Most Arduino Shields, Seeeduino accessories	Most Arduino Shields
Ext/USB power select	Auto	Auto

Pin Function Definition and Rating

Pin	Name	Function and Note
1	Reset	A switch that would reset the SeeeduinoMega
2	3.3V	3.3V Source
3	5V	5V Source
4	Gnd	Ground
5	Vin	A connection to the main source, this is used when the shield's supply is to be taken from the main power source
0~7	ADC / GPIO:PF0-PF7	Analog to Digital channels multiplexed with Port-F, used to interface with analog sensors like potentiometers, voltage , current, temperature, pressure, humidity sensors as well as analog gyroscopes and accelerometers
8~9	GPIO:PH5-PH6	General Purpose Input Output Pins
10~13	GPIO:PB4-PB7	General Purpose Input Output Pins
14	GND	A connection to the ground
15	AREF	The analog reference used as reference for the Seeeduino Mega’s ADC channels, Analog reference is decoupled to the ground using a capacitor for stability purposes.
0	GPIO:PE0/RX0	Receive channel for USART0
1	GPIO:PE1/TX0	Transmit channel for USART0
2~3	GPIO:PE4-PE5	General Purpose Input Output Pins
4	GPIO:PG5	General Purpose Input Output Pin

5	GPIO:PE3	General Purpose Input Output Pin
6~7	GPIO:PH3-PH4	General Purpose Input Output Pins
	ICSP	
8~15	ADC / GPIO:PK0-PK7	8 Analog to Digital channels multiplexed with Port-K
1	RXD1 / GPIO:PD2	Receive channel for USART1
2	TXD1 / GPIO:PD3	Transmit channel for USART1
3	RXD2 / GPIO:PH0	Receive channel for USART2
4	TXD2 / GPIO:PH1	Transmit channel for USART2
5	RXD3 / GPIO:PJ0	Receive channel for USART3
6	TXD3 / GPIO:PJ1	Transmit channel for USART3
I2C		Also known as the Two Wire Interface, I2C is an industry standard communication protocol that is used to communicate with ADCs, EEPROMs, DACs, sensors, and microcontrollers.
1	Vcc	
2	GND	
3	SCL / GPIO:PD0	I2C-Clock
4	SDA / GPIO:PD1	I2C-Serial Data
22~29	GPIO:PA0-PA7	General Purpose Input Output Pins
30-37	GPIO:PC0-PC7	General Purpose Input Output Pins
38	GPIO:PD7	General Purpose Input Output Pin
39~41	GPIO:PG2 - PG0	General Purpose Input Output Pins
42~45	GPIO:PL7 - PL4	General Purpose Input Output Pins
46~49	GPIO:PL3 - PL0	General Purpose Input Output Pins
SPI		
50	MISO / GPIO:PB3	SPI - Master In Slave Out
51	MOSI / GPIO:PB2	SPI - Master Out Slave In
52	SCK / GPIO:PB1	SPI - Clock
53	GPIO:PB0	General Purpose Input Output Pin
PH2	GPIO:PH2	General Purpose Input Output Pin
PH7	GPIO:PH7	General Purpose Input Output Pin
PJ2~PJ7	GPIO:PJ2-PJ7	General Purpose Input Output Pins
PD4~PD6	GPIO:PD4-PD6	General Purpose Input Output Pins
PG4~PG3	GPIO:PG4-PG3	General Purpose Input Output Pins
PE7	GPIO:PE7	General Purpose Input Output Pin
PE6	GPIO:PE6	General Purpose Input Output Pin
PE2	GPIO:PE2	General Purpose Input Output Pin

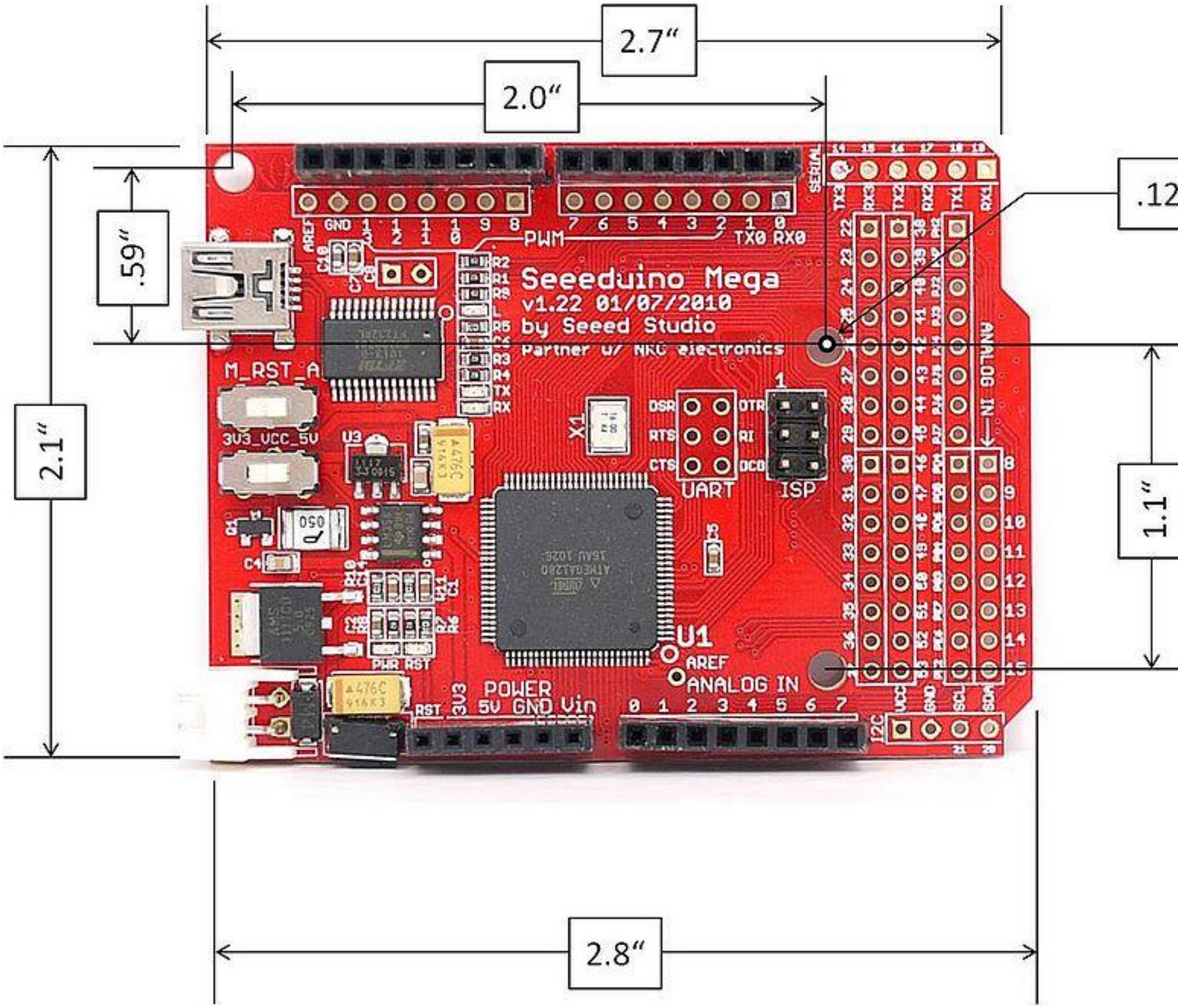
Port Naming Conventions



Indicates bit number (0-7), ex. PA0 means the first bit in Port A
PXn
Indicates port name, ex. PA which means Port A on the Atmega1280

Note: Port names and numbering in the Seeeduino Mega matches the port naming and numbering convension of Atmega1280’s datasheet.

Mechanic Dimensions

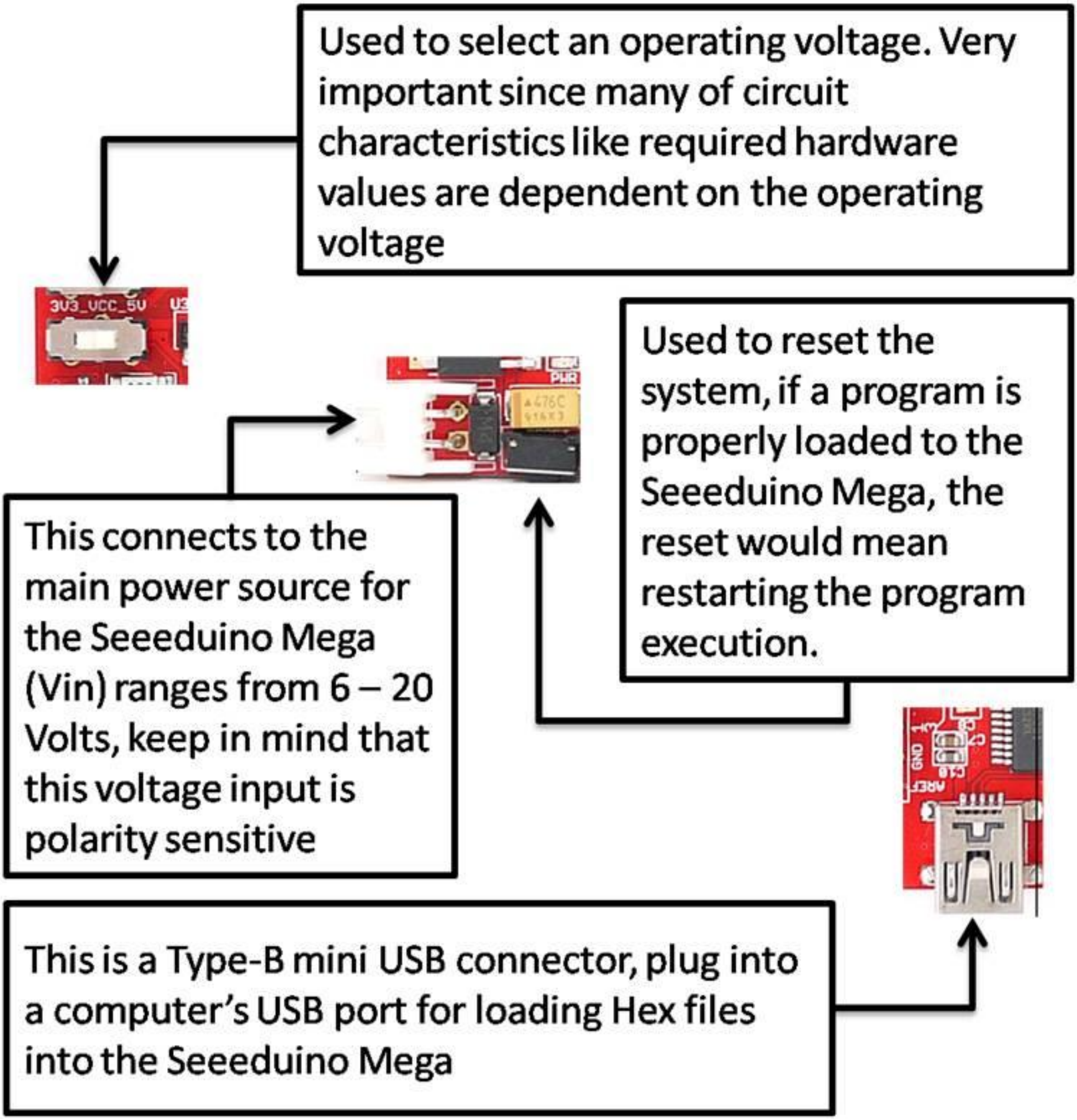


Notes
1. The dimensions given above are in terms of inches.
2. The given dimensions are compatible with most Arduino shields.
3. Mounting holes are designed to match the Seeeduino Harness

Usage

The Seeeduino Mega is a very powerful embedded development platform based on the Arduino project. It is capable of communicating on both analog and digital domains. Robot controllers, light dimmers, display controllers, digital interface translators, data acquisition interfaces and motor speed controllers are just some of the applications the Seeeduino Mega can do. The capabilities of the Atmega1280 is harnessed while eliminating the need for an external program loader, this makes the Seeeduino Mega not just powerful but also easy to use. Making prototyping fast, efficient and easy.

Hardware Installation



Used to select an operating voltage. Very important since many of circuit characteristics like required hardware values are dependent on the operating voltage

Used to reset the system, if a program is properly loaded to the Seeeduino Mega, the reset would mean restarting the program execution.

This connects to the main power source for the Seeeduino Mega (Vin) ranges from 6 – 20 Volts, keep in mind that this voltage input is polarity sensitive

This is a Type-B mini USB connector, plug into a computer's USB port for loading Hex files into the Seeeduino Mega

Programming

Programming the Seeeduino Mega is virtually easy, basic C programming knowledge would be more than sufficient to program. As mentioned before the Seeeduino Mega is based on the Arduino Project, which makes both program development and program loading a breeze. Before plugging the Seeeduino to a computer's USB port for programming, one must first code logic for the Seeeduino to execute. Coding is done on Integrated Development Environments (IDE) of which the most commonly used is the Arduino Development Environment. The Arduino

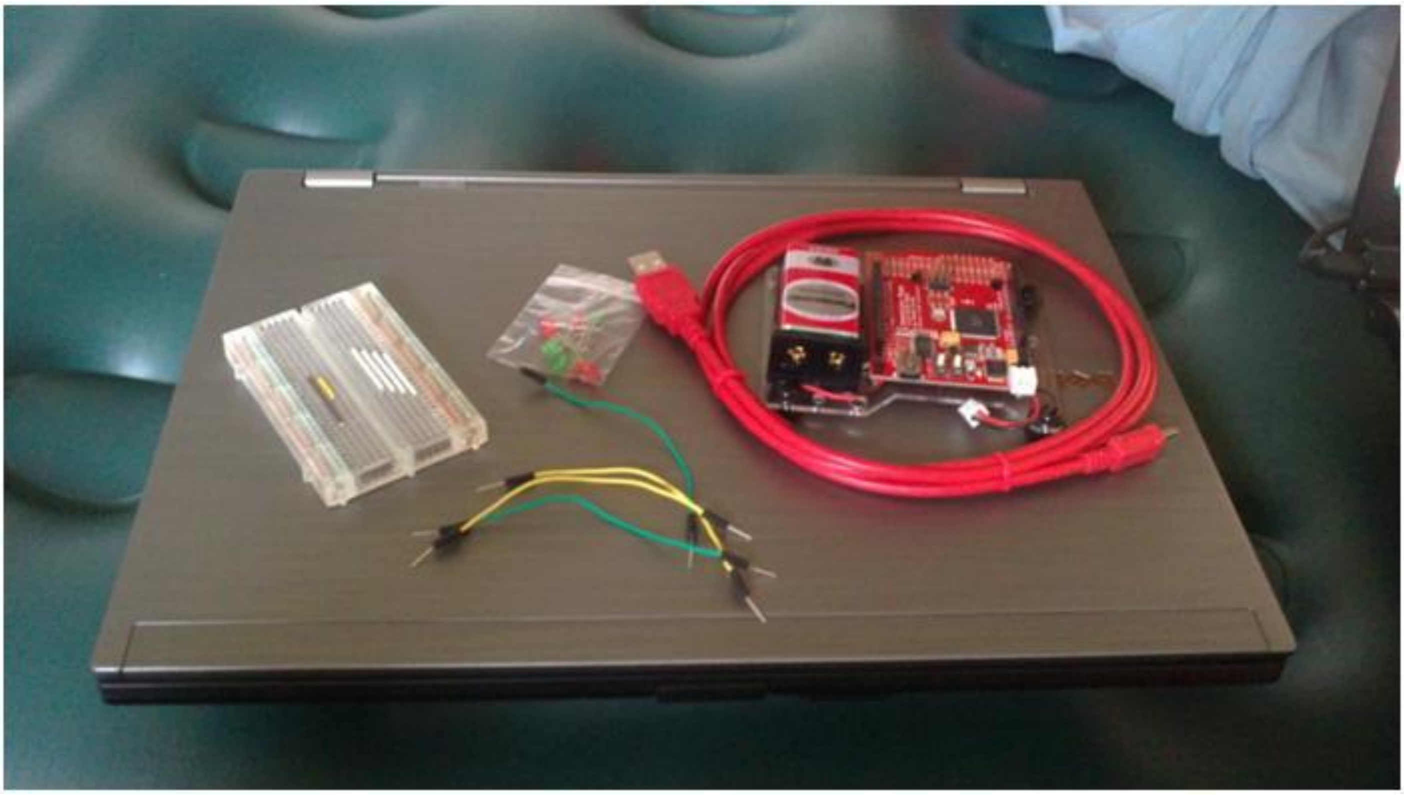
Development Environment is a java program that is very portable and has versions available for almost all operating systems.

The Arduino development environment contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions, and a series of menus. It connects to the Seeeduino Mega to upload programs and communicate with them. It is freely available and can be downloaded free of charge, it does not require installation, one just needs the friendly java runtime environment to be able to run the IDE.

To start our Mega-rific experience with the Seeeduino Mega here is a crash course on the Arduino, program codes written using the arduino is called sketches. One must write a program/sketch, compile the code to verify its validity next would be to connect the Seeeduino Mega to the USB port and start loading the compiled program and the last step would be to test the actual performance of the Seeeduino. Now we're ready to rock with the Seeeduino Mega!

Example

What do you need

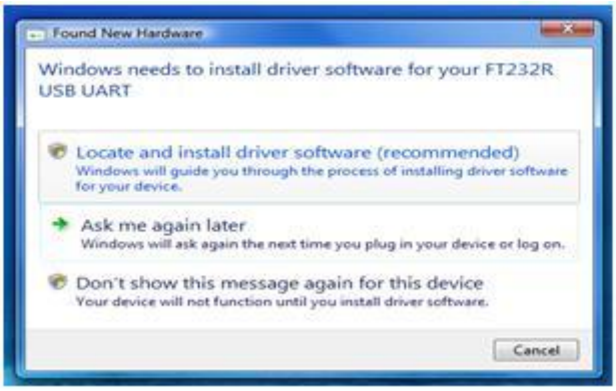


- Computer with a USB port
- Jumper wires
- Mini-USB cable
- Breadboard
- LEDs
- Resistors - for current limiting purposes (not used in this example)
- **SeeeduinoMega!**

The Set Up

*Connect the mini-USB cable to the Seeeduino Mega.

- Connect the other side of the mini-USB connector to the copmters's USB port.
- Wait for the new found hardware prompt.
- You can ethier browse for the drivers in your local file system - if you aleady downloaded drivers from the FTDI website or you could use the windows update option.



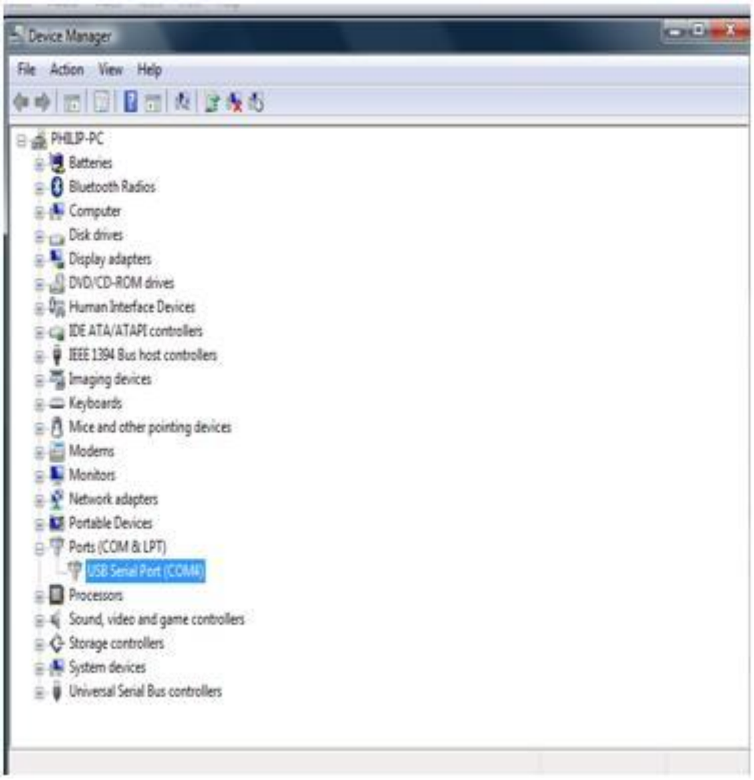
Found New Hardware

Windows needs to install driver software for your FT232R USB UART

Locate and install driver software (recommended)
Windows will guide you through the process of installing driver software for your device.

Ask me again later
Windows will ask again the next time you plug in your device or log on.

Don't show this message again for this device
Your device will not function until you install driver software.



*After the driver installation, go to **My Computer > Properties > Device Manager** then look for **Ports (COM and LPT) > USB Serial Port (COMn)** the n may vary from one PC to another, in this case we have 4. Take note of this since it will be used in the Arduino IDE configuration.

- Run the Arduino IDE! A splash screen would then pop up.

An open project written, debugged and supported by Massimo Banzi, David Cuartielles, Tom Igoe Gianluca Martino and David Mellis

Based on processing by Casey Reas and Ben Fry

The Program

Now let’s try the "Hello World" of microcontrollers the blinking LEDs, Go to **File > Examples > 1.Basics > Blink**. The program is fairly simple - as the name suggests it will make a LED blink at 0.5Hz. Turns a LED ON and OFF in 1 second intervals.

```
int ledPin = 13;                  // LED connected to digital pin 13
void setup()
{
  pinMode(ledPin, OUTPUT);       // sets the digital pin as output
}

void loop()
{
  digitalWrite(ledPin, HIGH);    // sets the LED on
  delay(1000);                   // waits for a second
  digitalWrite(ledPin, LOW);     // sets the LED off
  delay(1000);                   // waits for a second
}
```

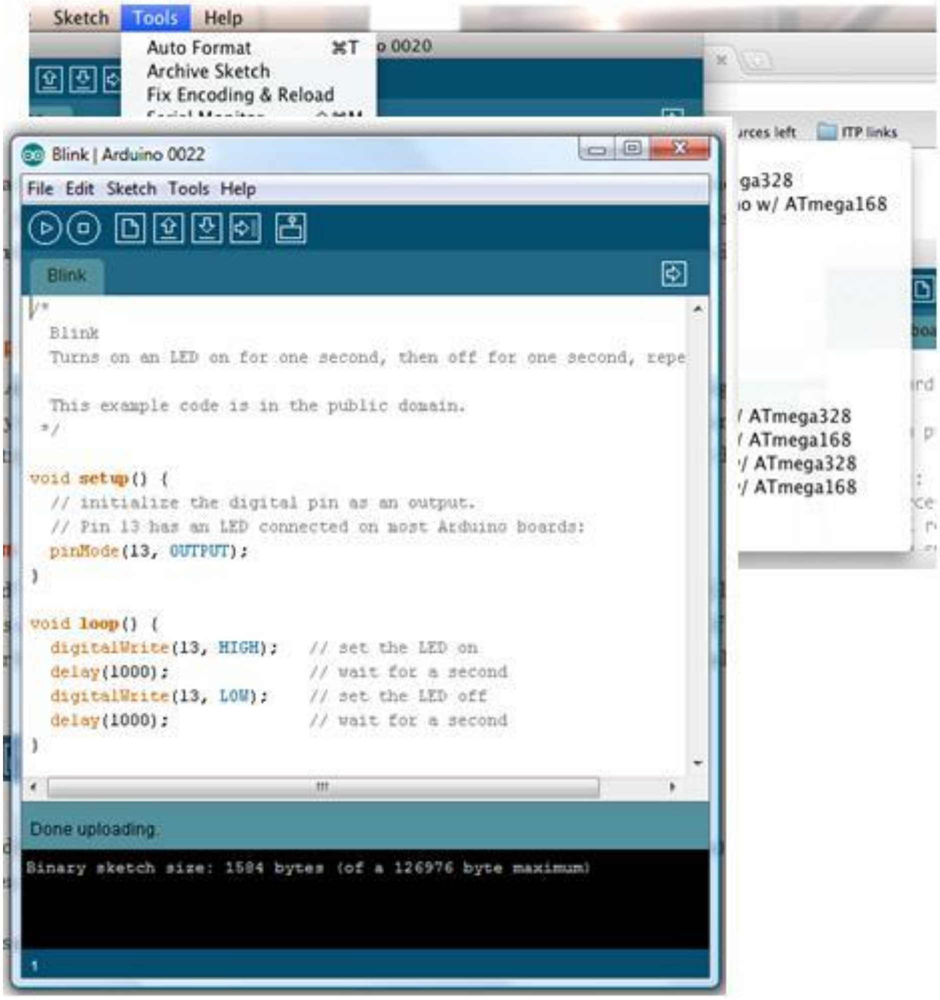
- **pinMode(13, OUTPUT)** - This line indicates, declares that Pin 13 on the Seeeduino Mega is an output Pin.
- **digitalWrite(13, HIGH)** - Outputs a logic high on Pin 13, Logic 1 voltage is dependent on setting.
- **delay(1000)** - This line delays the excecution by 1 second, the parameter is in milliseconds.
- **digitalWrite(13, LOW)** - Ouputs a logic low on Pin 13

Arduino IO Notes:
Some of you might have noticed that we made the setup IO pin 13 as OUTPUT. But how about if we want to use a certain IO pin for digital input? What should we do? As starters, when SeeeduinoMega or as a matter of fact any Arduino powers up all digital pins are set to INPUT mode in default - this sets each digital IO pin to a high impedance mode as default, thus we do not need to set to INPUT using the pinMode() function. A simple program that uses a 1 pin as input and another as output is shown below.

```
int ledPin = 13;             // LED connected to digital pin 13
int inPin = 7;               // pushbutton connected to digital pin 7
int val = 0;                 // variable to store the read value
void setup()
{
pinMode(ledPin, OUTPUT);     // sets the digital pin 13 as output
pinMode(inPin, INPUT);       // sets the digital pin 7 as input
}
void loop()
{
val = digitalRead(inPin);    // read the input pin
digitalWrite(ledPin, val);  // sets the LED to the button's value
}
```

In the example above, we initially set the ledPin (Pin13) to output and the inPin (Pin7) as input. The program simply reflects the status of the push button to the LED connected to Pin13. But note! you you must always put to mind that the INPUT pin must be at a defined state when we will attempt to read digital value from it - when it is left hanging it could either be a HIGH or a LOW at random. To remedy this we must put a pull up resistor on Pin7 this will ensure that it has a logic state (HIGH) and we must also connect the pushbutton such that it is normally open with respect to the ground. This way we will be able to explicitly define the state of our INPUT pin. And when the pushbutton is pressed the Pin will read LOW and when it is released the Pin's logic state will be pulled up by the pull up resistor giving us a HIGH.

Now that we have the blinking LED program its now time to load it into the Seeeduino Mega.



```
/*
  Blink
  Turns on an LED on for one second, then off for one second, repe

  This example code is in the public domain.
 */

void setup() {
  // initialize the digital pin as an output.
  // Pin 13 has an LED connected on most Arduino boards:
  pinMode(13, OUTPUT);
}

void loop() {
  digitalWrite(13, HIGH);   // set the LED on
  delay(1000);              // wait for a second
  digitalWrite(13, LOW);    // set the LED off
  delay(1000);              // wait for a second
}
```

```
Done uploading.
```

```
Binary sketch size: 1584 bytes (of a 126976 byte maximum)
```

First we must pick the type of Arduino board we are using, there are a LOT of Arduino versions out there - pick **Arduino Mega (AtMega1280)**.

Next would be the serial port, remember the port we found on the device manager earlier? Select that port on the **Tools>Serial Port** option.

Fatal error: Allowed memory size of 134217728 bytes exhausted (tried to allocate 433103136 bytes) in **Unknown** on line **0**

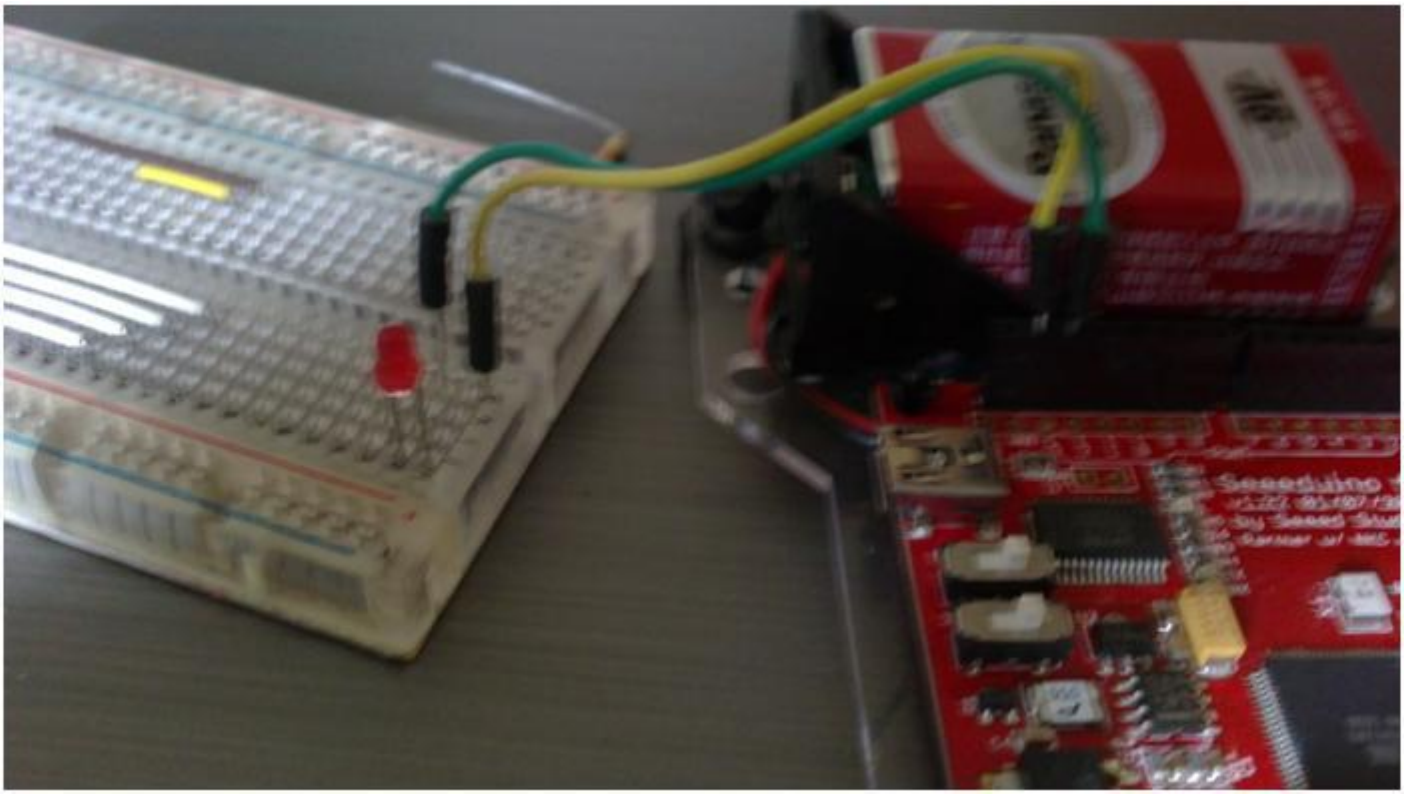
The Arduino IDE

The figure on the left shows the Arduino IDE with the Blinking LEDs program loaded. To upload the program to the Seeeduino Mega we need to press the upload button:

After the upload the "Done uploading" message would appear. Simultaneaously the L LED (yellow green LED) on the board will start to blink - since the L LED is connected to Pin 13.

The picture on the right shows the blinking yellow-green LED on the Seeeduino Mega.

We will now take the Blinking LEDs further. Lets connect an external LED on Pin 13 to demonstrate how shields connect to the Seeeduino Mega.



Green is Pin13 while Yellow in Ground, in this set up I used a 3.5mm LED the maximum current that should be forced to it must be below 10mA. However to have a brighter blink I omitted the series resistor - thus in order to dim the output just add a resistor in series of the LED. Also take note that an LED's legs can't be and shouldn't be interchanged. See http://en.wikipedia.org/wiki/Light-emitting_diode for more details.

Bill of Materials (BOM) /parts list

All the components used to produce the product.

FAQ

Please list your questions here:

Support

If you have questions or other better design ideas, you can go to our forum or wish to discuss.

Version Tracker

Revision	Descriptions	Release Date
Seeeduino Mega V1.21		Dec 1, 2010
Seeeduino Mega V1.23	Stronger mini USB connector	Jun 21, 2011

Bug Tracker

Bug Tracker is the place you can publish any bugs you think you might have found during use. Please write down what you have to say, your answers will help us improve our products.

Additional Idea

The Additional Idea is the place to write your project ideas about this product, or other usages you've found. Or you can write them on Projects page.

Resources

- Seeeduino Mega v1.23 Eagle format Source files

See Also

- Seeeduino V2.2
- Seeeduino Stalker
- Seeeduino v2.21
- Seeeduino Film
- Seeeduino Motion Frame

Licensing

External Links

Here is some useful links provided for you:

- Where to get Arduino IDE: http://arduino.cc/
- Wherecan I get the USB -to-Rs232 drivers: http://www.ftdichip.com/Drivers/VCP.htm
- Where can I find the Pin conventions: http://www.arduino.cc/playground/Main/ShieldPinUsage