Lab 1: Arduino Basics

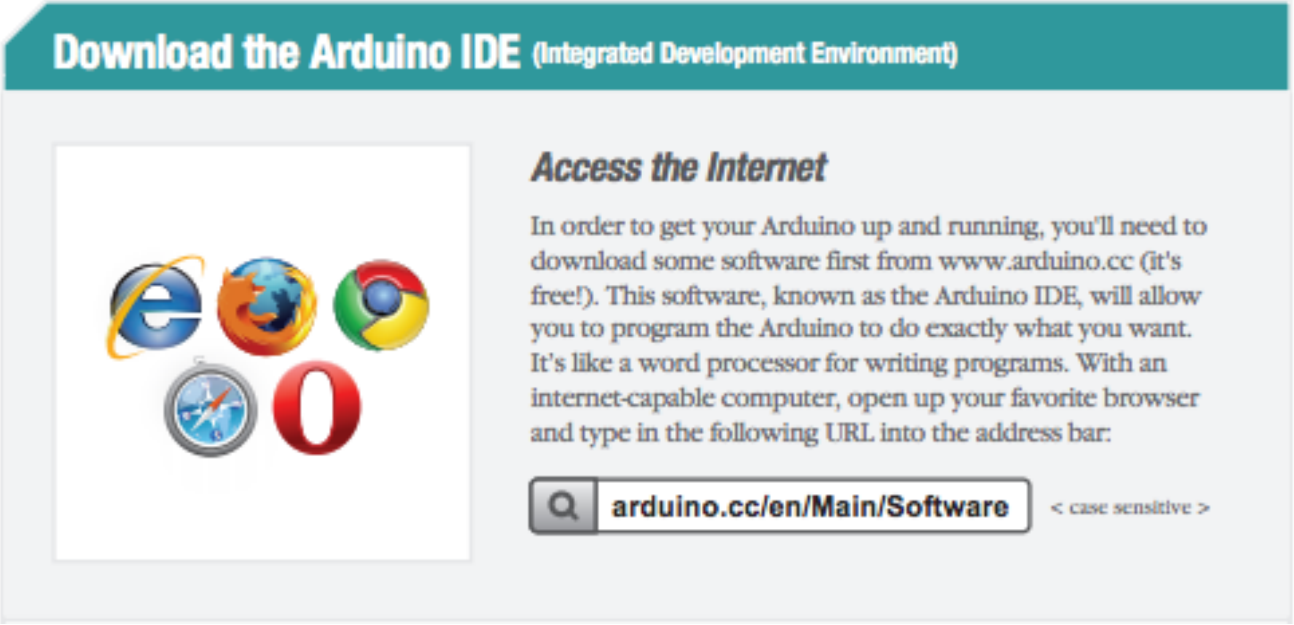
Marco Zennaro and Antoine Bagula
ICTP and UWC
Italy and South Africa

Goals of this Lab

- Learn how the programming takes place
- Excercises about:
  - installing the IDE
  - setting the clock
  - measuring the temperature
  - timestamping the temperature reading

Installing the IDE

- IDE= Integrated development environment
- Arduino IDE is Open Source



Download the Arduino IDE (Integrated Development Environment)

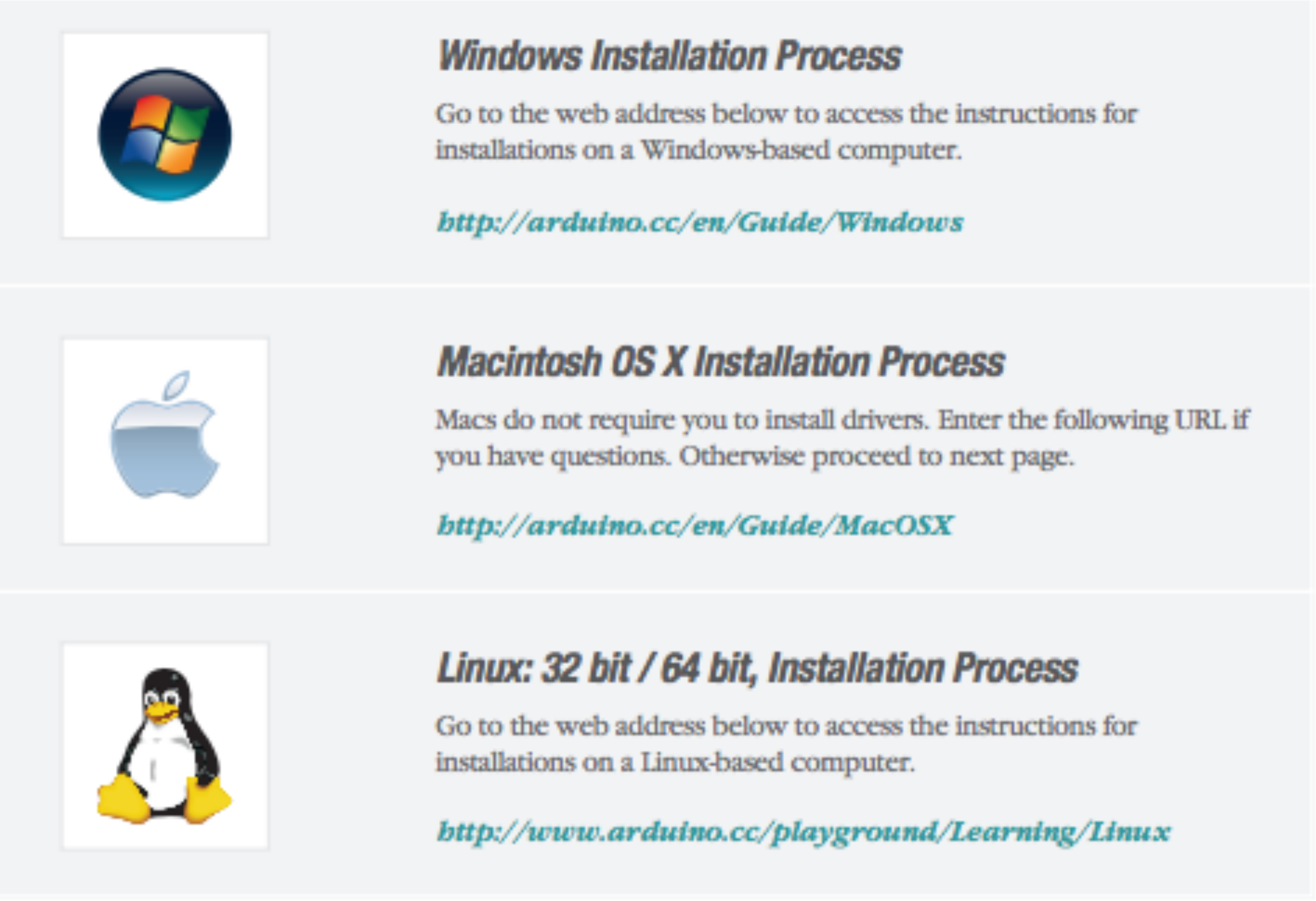
Access the Internet

In order to get your Arduino up and running, you'll need to download some software first from www.arduino.cc (it's free!). This software, known as the Arduino IDE, will allow you to program the Arduino to do exactly what you want. It's like a word processor for writing programs. With an internet-capable computer, open up your favorite browser and type in the following URL into the address bar:

arduino.cc/en/Main/Software

< case sensitive >

Installing the IDE



Windows Installation Process

Go to the web address below to access the instructions for installations on a Windows-based computer.

http://arduino.cc/en/Guide/Windows

Macintosh OS X Installation Process

Macs do not require you to install drivers. Enter the following URL if you have questions. Otherwise proceed to next page.

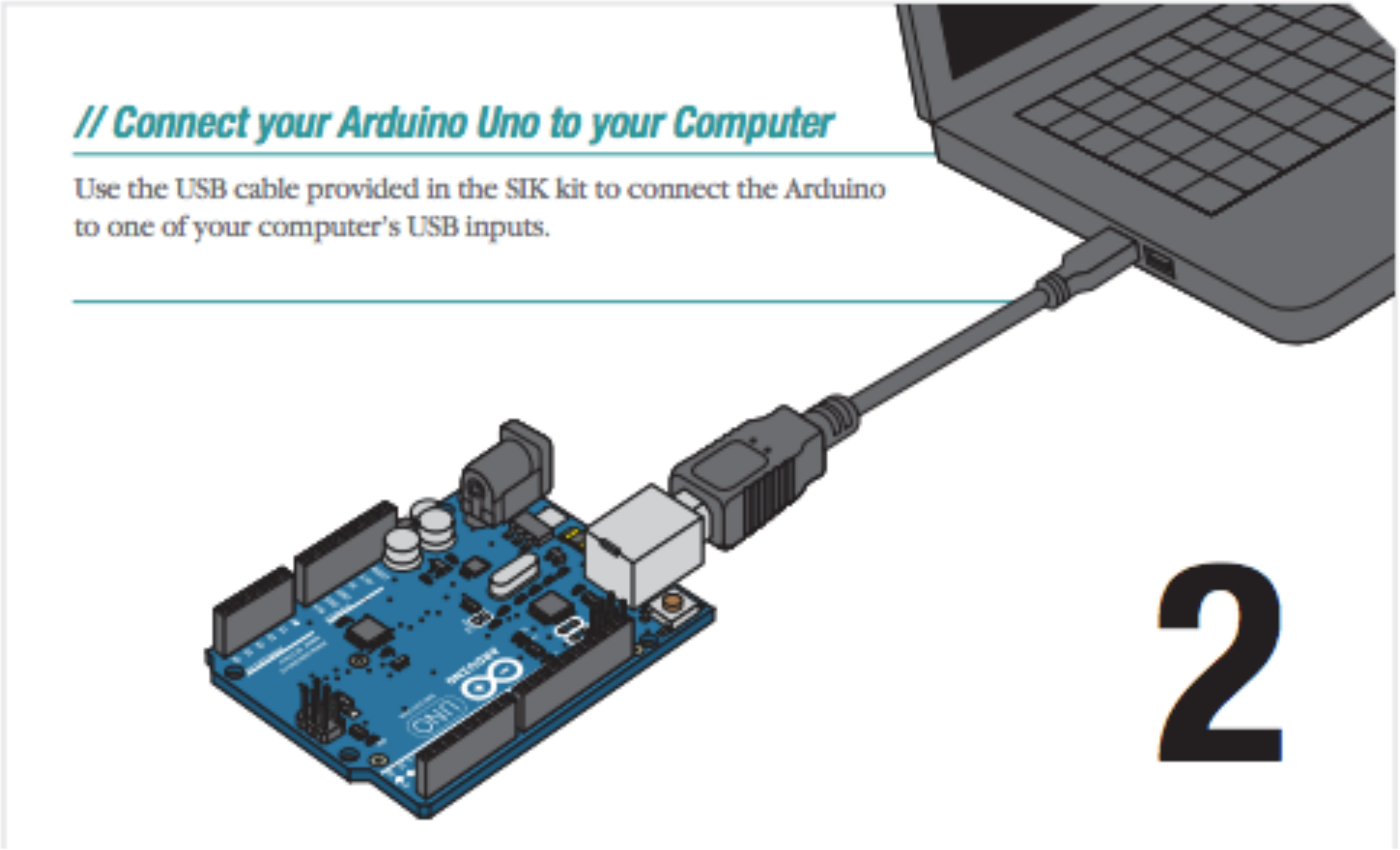
http://arduino.cc/en/Guide/MacOSX

Linux: 32 bit / 64 bit, Installation Process

Go to the web address below to access the instructions for installations on a Linux-based computer.

http://www.arduino.cc/playground/Learning/Linux

Connect your Arduino



// Connect your Arduino Uno to your Computer

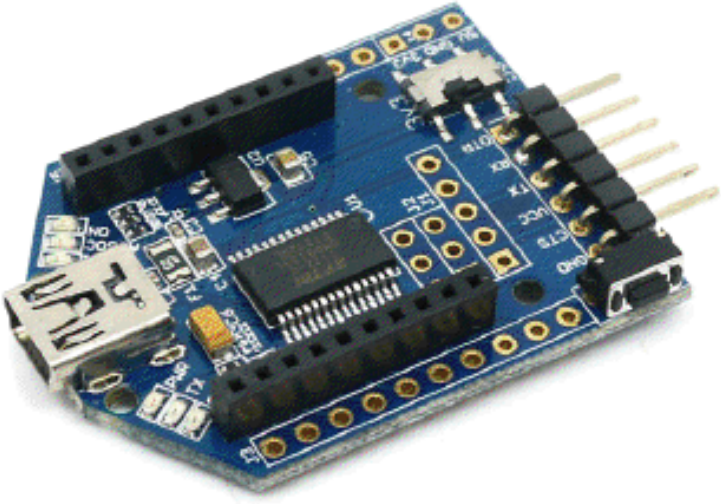
Use the USB cable provided in the SIK kit to connect the Arduino to one of your computer's USB inputs.

2

Note: we will use a slightly different board, which you connect through one additional small board, the programmer

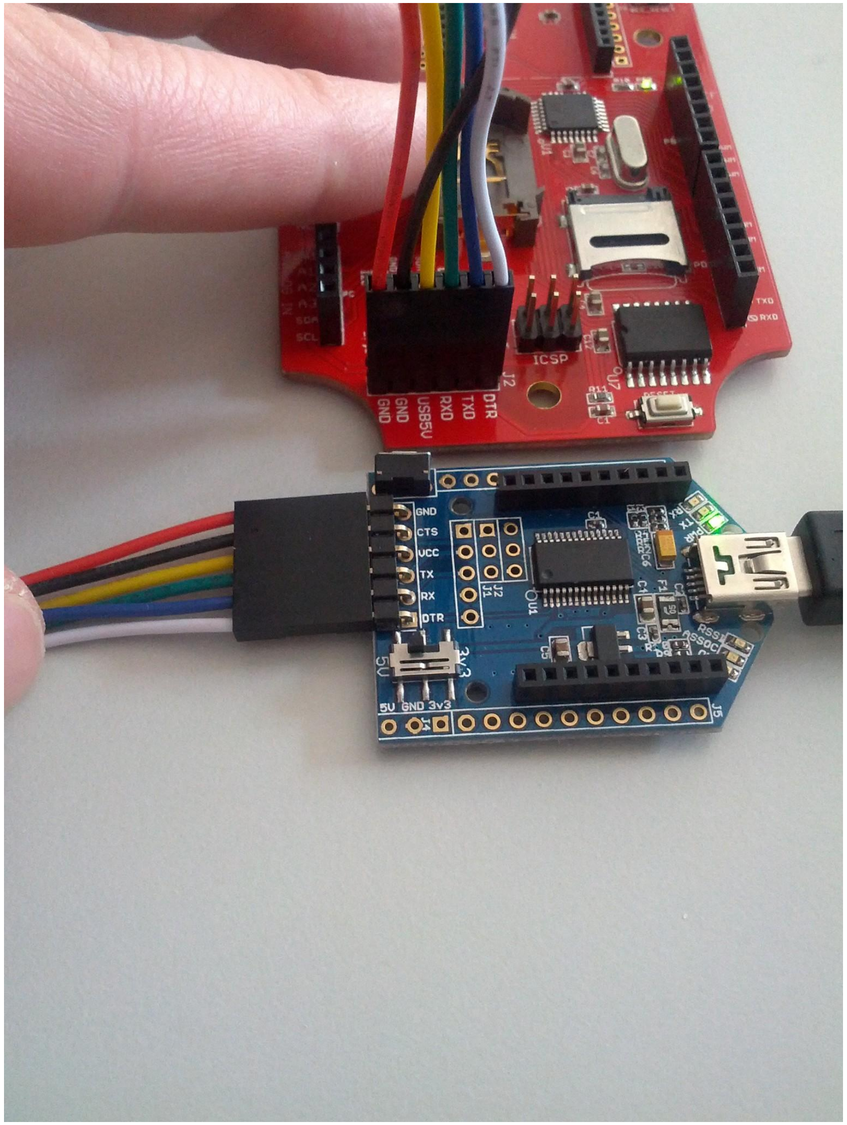
Connect your Seeduino

This is the USB-to-serial adapter - it will be between your computer and the Seeeduino board



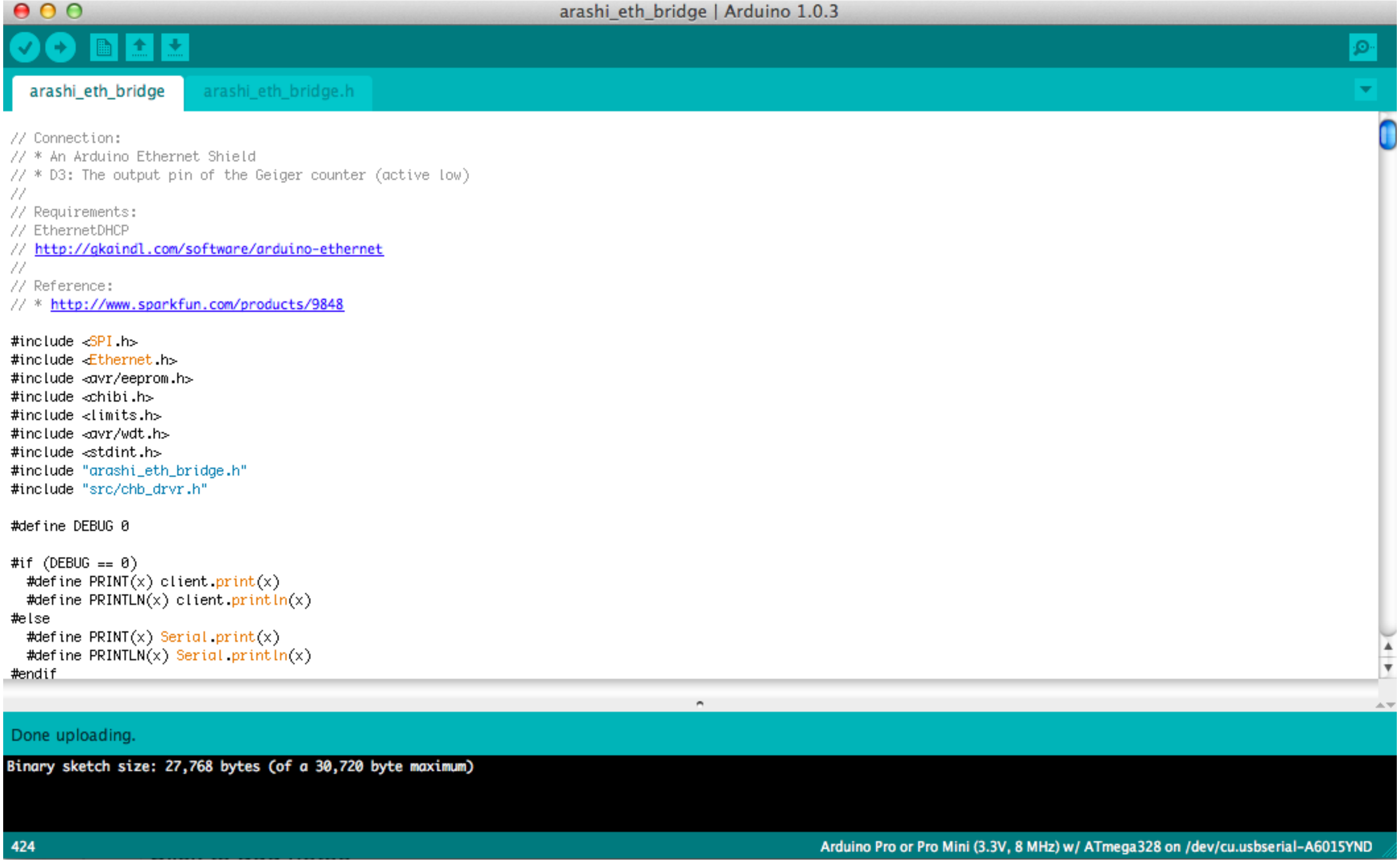
Connect the multi-colored cable so that the GND pins on the Seeduino and on the USB device have the same color (white, for example).

Connect your Seeduino



The other end of this USB cable goes to your computer.

The Arduino IDE



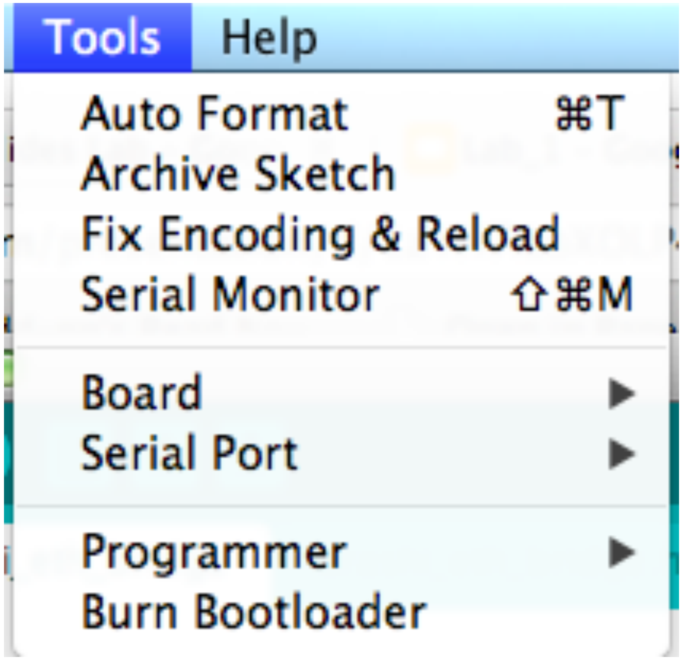
```
// Connection:
// * An Arduino Ethernet Shield
// * D3: The output pin of the Geiger counter (active low)
//
// Requirements:
// EthernetDHCP
// http://gkaindl.com/software/arduino-ethernet
//
// Reference:
// * http://www.sparkfun.com/products/9848

#include <SPI.h>
#include <Ethernet.h>
#include <avr/eeprom.h>
#include <chibi.h>
#include <limits.h>
#include <avr/wdt.h>
#include <stdint.h>
#include "arashi_eth_bridge.h"
#include "src/chb_drvr.h"

#define DEBUG 0

#if (DEBUG == 0)
  #define PRINT(x) client.print(x)
  #define PRINTLN(x) client.println(x)
#else
  #define PRINT(x) Serial.print(x)
  #define PRINTLN(x) Serial.println(x)
#endif
```

Done uploading.

```
Binary sketch size: 27,768 bytes (of a 30,720 byte maximum)
```

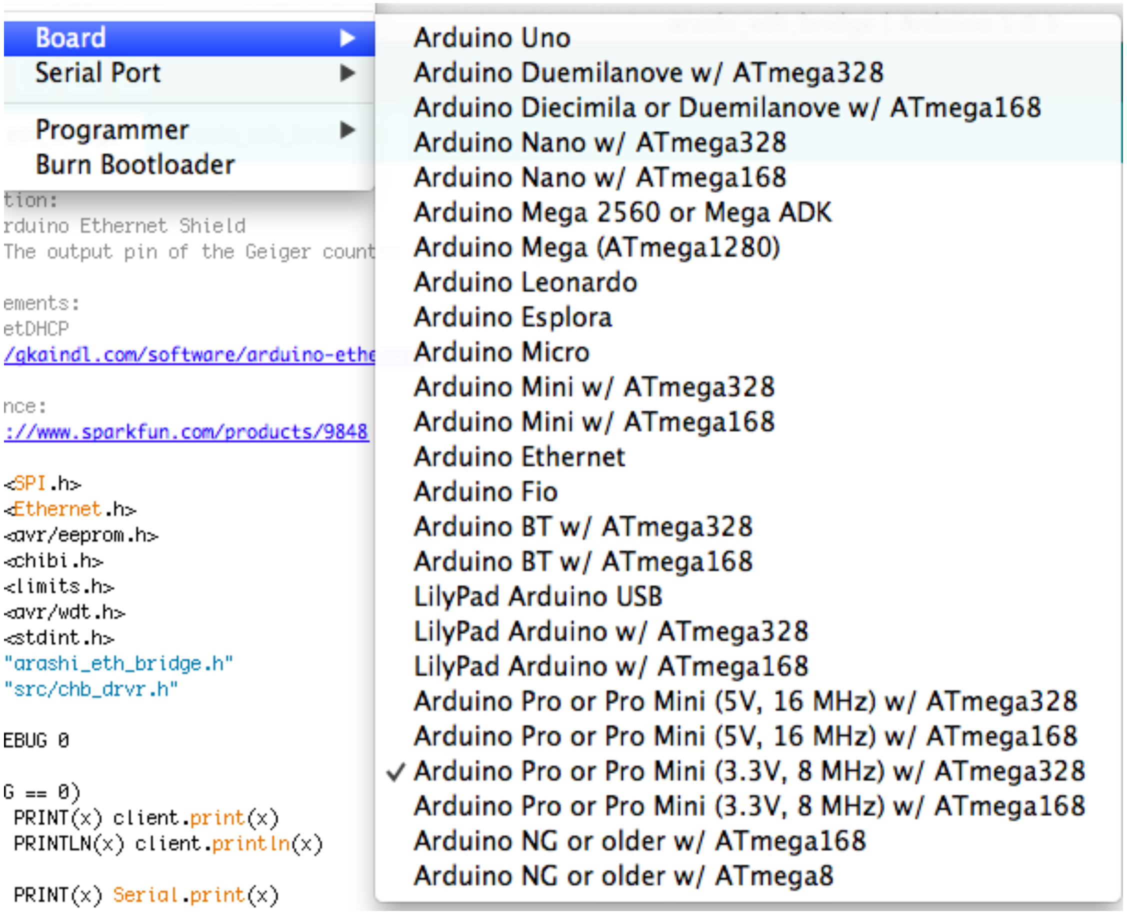
424 Arduino Pro or Pro Mini (3.3V, 8 MHz) w/ ATmega328 on /dev/cu.usbserial-A6015YND

1: Select serial port



Select the Serial Port:
tty/USBx on Linux,
COMx on Windows

2: Select Arduino model


Board
Serial Port
Programmer
Burn Bootloader
Arduino Uno
Arduino Duemilanove w/ ATmega328
Arduino Diecimila or Duemilanove w/ ATmega168
Arduino Nano w/ ATmega328
Arduino Nano w/ ATmega168
Arduino Mega 2560 or Mega ADK
Arduino Mega (ATmega1280)
Arduino Leonardo
Arduino Esplora
Arduino Micro
Arduino Mini w/ ATmega328
Arduino Mini w/ ATmega168
Arduino Ethernet
Arduino Fio
Arduino BT w/ ATmega328
Arduino BT w/ ATmega168
LilyPad Arduino USB
LilyPad Arduino w/ ATmega328
LilyPad Arduino w/ ATmega168
Arduino Pro or Pro Mini (5V, 16 MHz) w/ ATmega328
Arduino Pro or Pro Mini (5V, 16 MHz) w/ ATmega168
✓ Arduino Pro or Pro Mini (3.3V, 8 MHz) w/ ATmega328
Arduino Pro or Pro Mini (3.3V, 8 MHz) w/ ATmega168
Arduino NG or older w/ ATmega168
Arduino NG or older w/ ATmega8

```
tion:
rduino Ethernet Shield
The output pin of the Geiger count

ements:
etDHCP
/gkaindl.com/software/arduino-ethe

nce:
://www.sparkfun.com/products/9848

<SPI.h>
<Ethernet.h>
<avr/eeprom.h>
<chibi.h>
<limits.h>
<avr/wdt.h>
<stdint.h>
"arashi_eth_bridge.h"
"src/chb_drvr.h"

EBUG 0

G == 0)
 PRINT(x) client.print(x)
 PRINTLN(x) client.println(x)

 PRINT(x) Serial.print(x)
```

Select Board:
Arduino Pro
3.3V, ATmega328

Programming an Arduino

From the File menu, choose Open and select the code you want to open.

The source code will appear in the IDE window.

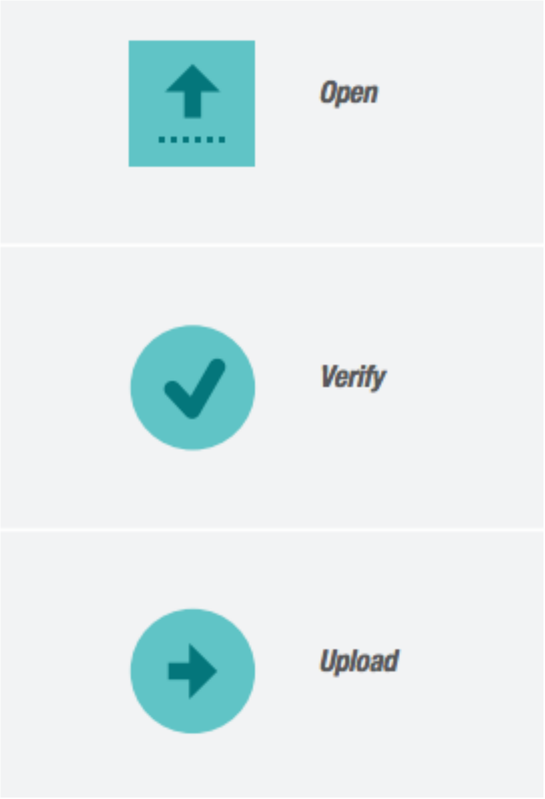
Lab Examples

From the Workshop's webpage,
download the zip file with all the examples
for this Lab 1 Session.

Open the folder called Example_1
and open the Example_1.ino file

Programming workflow

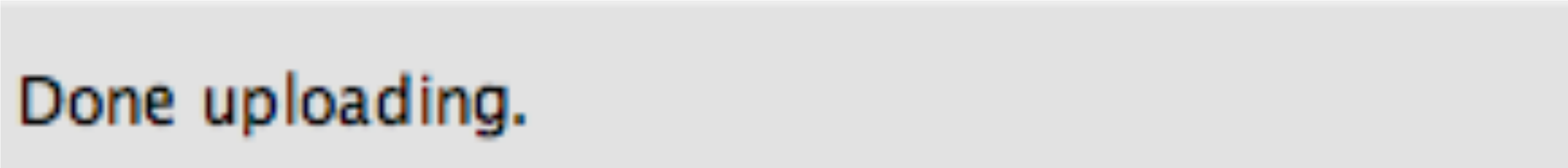
1. Opening
2. Verifying
3. Uploading



Programming an Arduino

Click on the upload button and wait until the code has been compiled and uploaded.

At the end you will see in the bottom right corner:



Done uploading.

Programming an Arduino

This is the template of a basic Arduino program:

```
void setup()
{
  Initialize variables, open USB, open WiFi, etc
}
```

SETUP
(once)

```
void loop()
{
   Perform some action

   Wait for a certain number of msecs or wait for an alarm
}
```

LOOP
(forever)

Lab session

This lab session will be like this:

```
For (i=1;i<=3;i++) {
    Simple example (me)  /* 2 min */
    Extended example (you) /* 20 min */
}

Real-world exercise /* 1 hour */
```

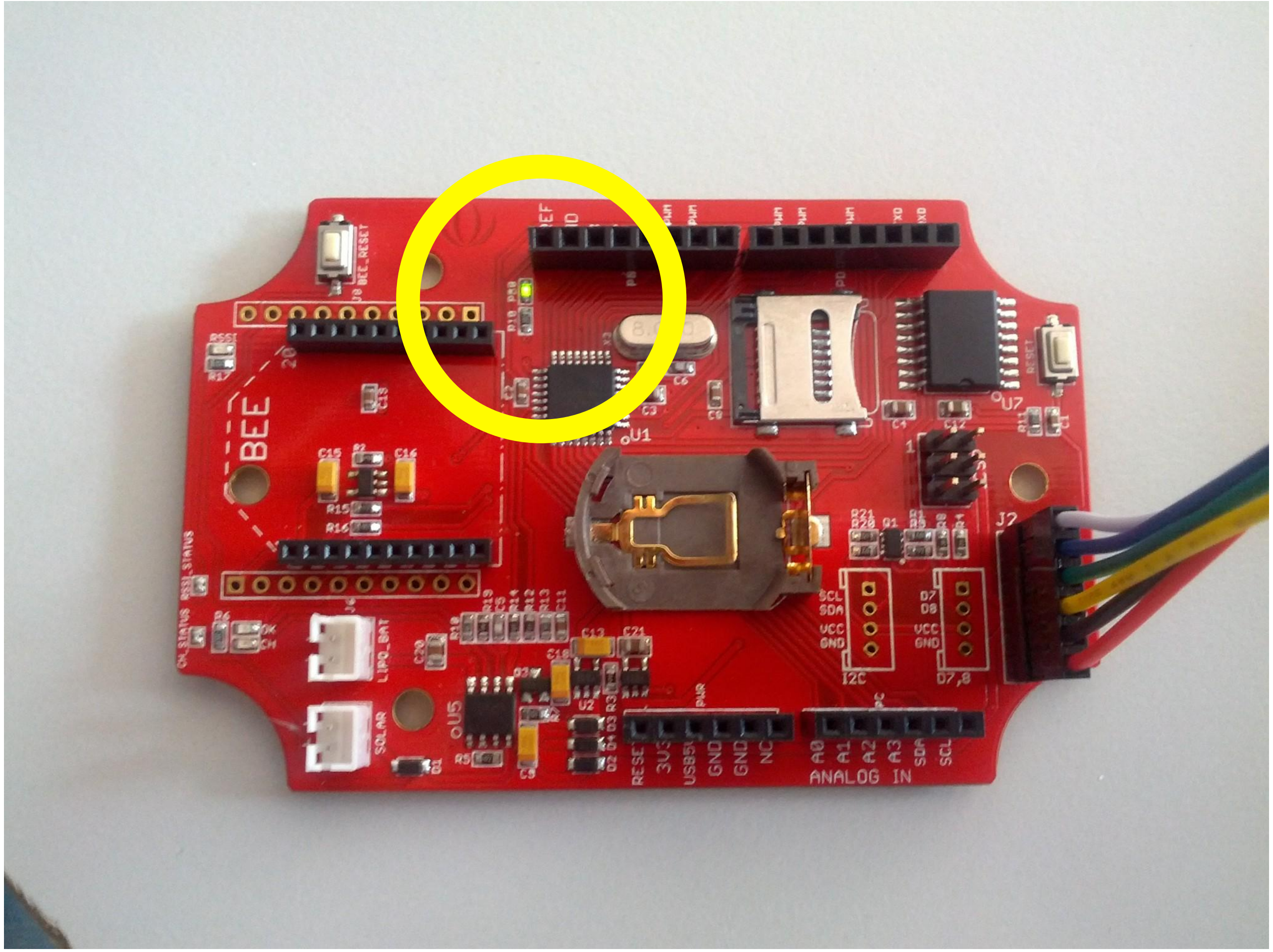
Start!

Example_1

Example_1 will blink a light (Hello World! in the WSN world) on the Seeduino.

```
int led = 13;

void setup() {
  // initialize the digital pin as an output.
  pinMode(led, OUTPUT);
}

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

Example_1 output



Example_1 - extended

Get acquainted with the IDE

Make the LED blink for a different amount of time

Make the LED blink as fast as possible!

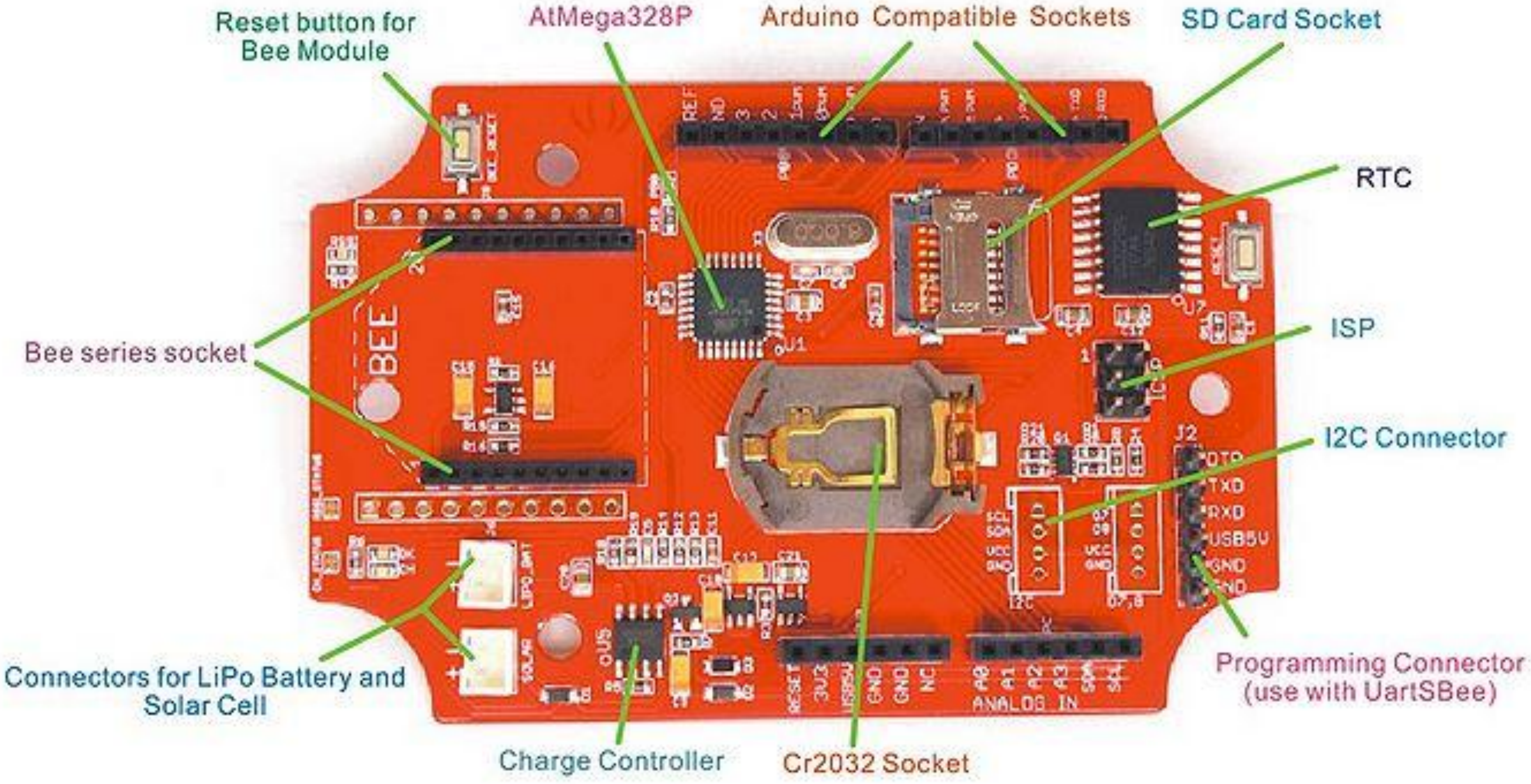
Seeeduino

Seeeduino is an Arduino board designed for WSN applications.

Special features:

- RTC (Real Time Clock) to timestamp the data
- microSD card to store data
- socket for Xbee modules
- solar charger on board
- very low deep sleep mode (1000uA)

Seeeduino


Reset button for Bee Module
AtMega328P
Arduino Compatible Sockets
SD Card Socket
RTC
ISP
I2C Connector
Bee series socket
Programming Connector (use with UartSBee)
Connectors for LiPo Battery and Solar Cell
Charge Controller
Cr2032 Socket

Seeeduino RTC

Having a RTC is useful for two reasons:

1. to time stamp the collected data (for example: temperature is 27.4C at 10:02:30 of 6/7/2020)
2. to be able to set alarms to wake up the mote from sleeping mode (for example: wake up on Tuesday 15th of August at 10:30:00).

Insert the coin battery in the Seeeduino.

Example_2

Example_2 will set the time of the Seeduino using the RTC.

To program the RTC you need to download some libraries first. In the zip file you will find the DS3231 library and examples.

Place the files in folder labelled “ds3231_library” into your Arduino libraries folder.

Linux: /

Windows:

Restart the IDE!

Example_2

Example_2 will set the time of the Seeduino using the RTC.

Change the line:

DateTime dt(2011, 11, 10, 15, 18, 0, 5);

to adjust to today's date and time.

Format is: year, month, date, hour, min, sec and week-day (starts from 0 (Sunday) and goes to 6 (Saturday))

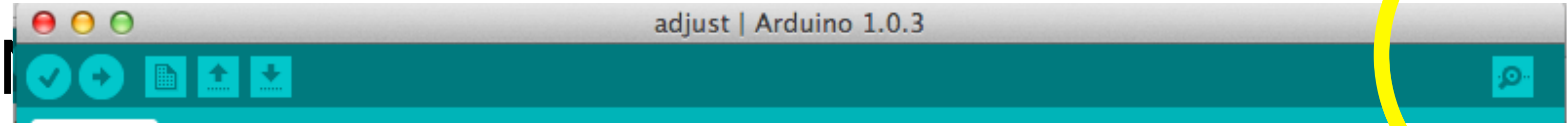
Example_2

How do you see the output of your code?

You need to check the output coming from the USB port.

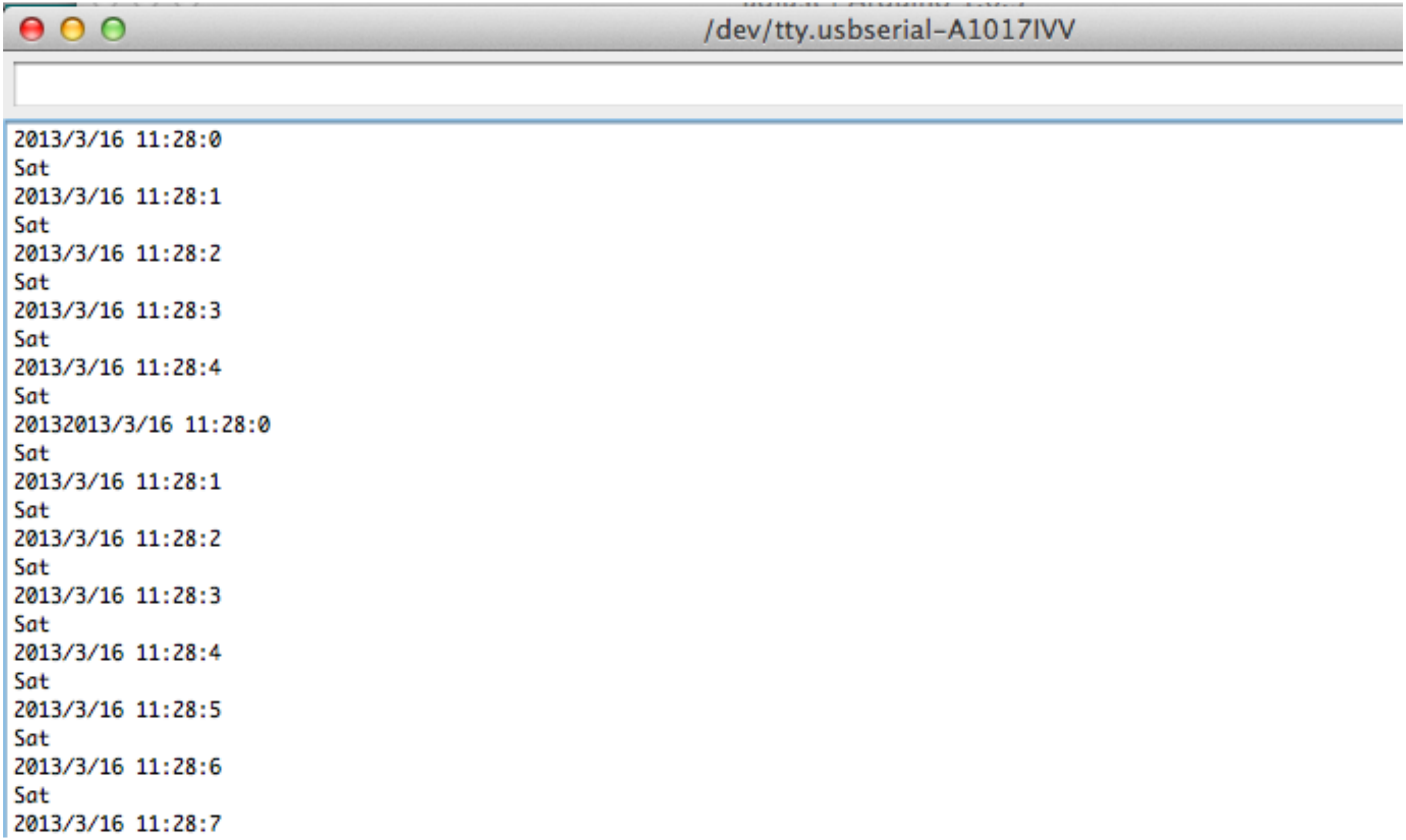
Select Serial Monitor:

I e!



Your Serial Monitor needs to match that!

Example_2 output



```
2013/3/16 11:28:0
Sat
2013/3/16 11:28:1
Sat
2013/3/16 11:28:2
Sat
2013/3/16 11:28:3
Sat
2013/3/16 11:28:4
Sat
20132013/3/16 11:28:0
Sat
2013/3/16 11:28:1
Sat
2013/3/16 11:28:2
Sat
2013/3/16 11:28:3
Sat
2013/3/16 11:28:4
Sat
2013/3/16 11:28:5
Sat
2013/3/16 11:28:6
Sat
2013/3/16 11:28:7
```

Example_2 - extended

Comment the line where you set the time. Is the time OK?

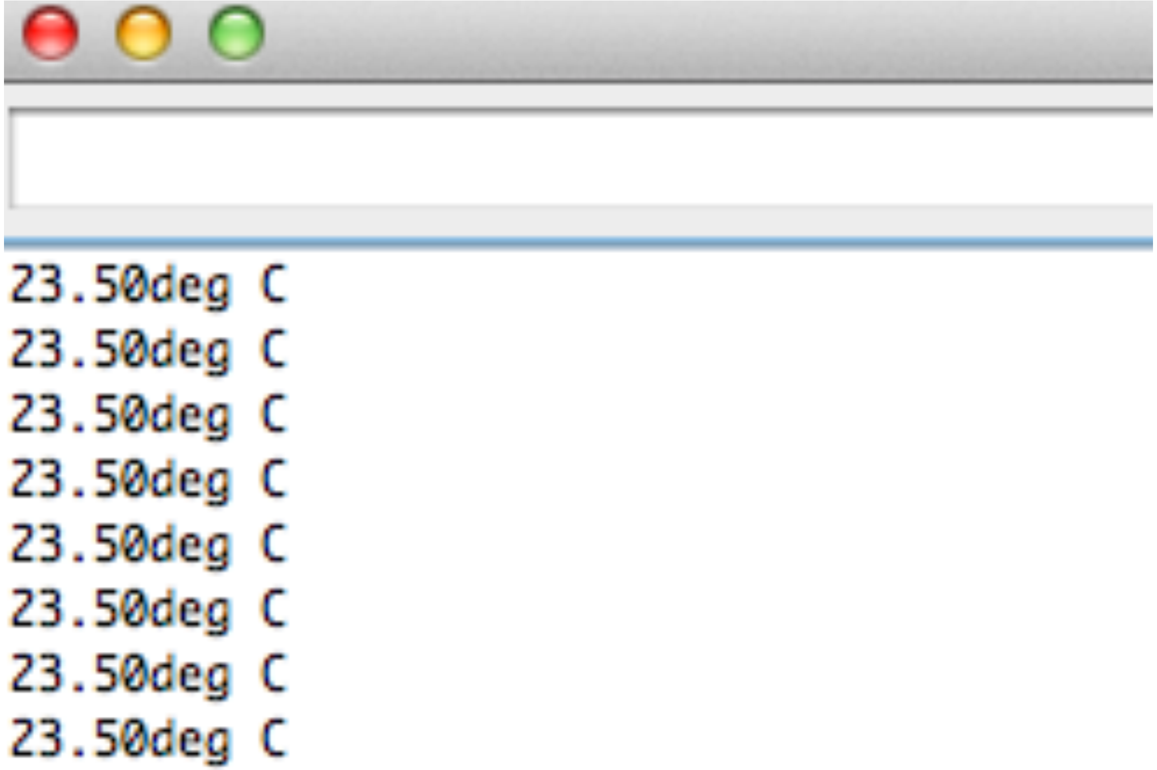
Disconnect the Seeduino from the USB. Connect it again. Are the dats and time OK?

Example_3

The RTC has a temperature sensor to keep the clock calibrated.

```
void setup ()
{
   Serial.begin(57600);
   Wire.begin();
   RTC.begin();
}
void loop ()
{
   RTC.convertTemperature();
   Serial.print(RTC.getTemperature());
   Serial.println("deg C");
   delay(1000);
}
```

Example_3 output



```
23.50deg C
23.50deg C
23.50deg C
23.50deg C
23.50deg C
23.50deg C
23.50deg C
23.50deg C
```

Example_3 - extended

Put the Seeduino near the window / light: does the temperature change?

Convert the temperature to Fahrenheit and show values in both C and F.

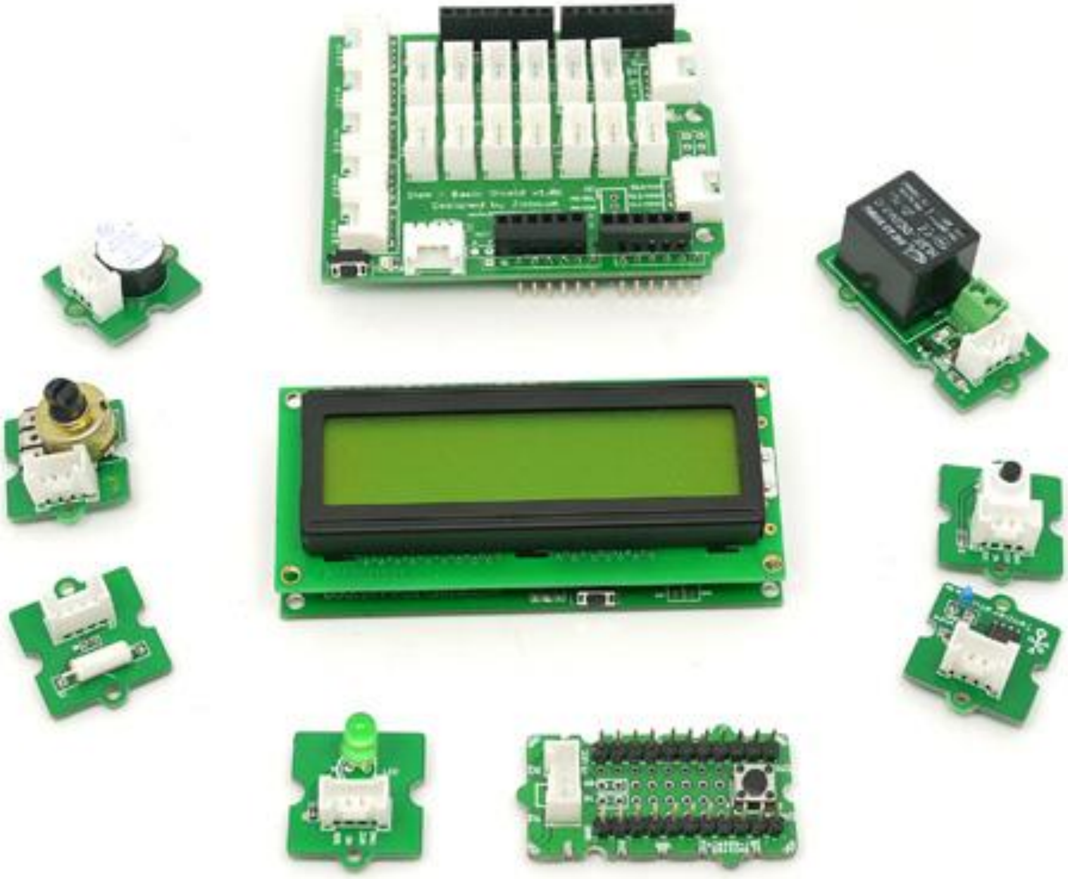
Exercise

I own a chalet in Switzerland and want to monitor its temperature every 2 minutes. I want to visualize date, time and temperature.

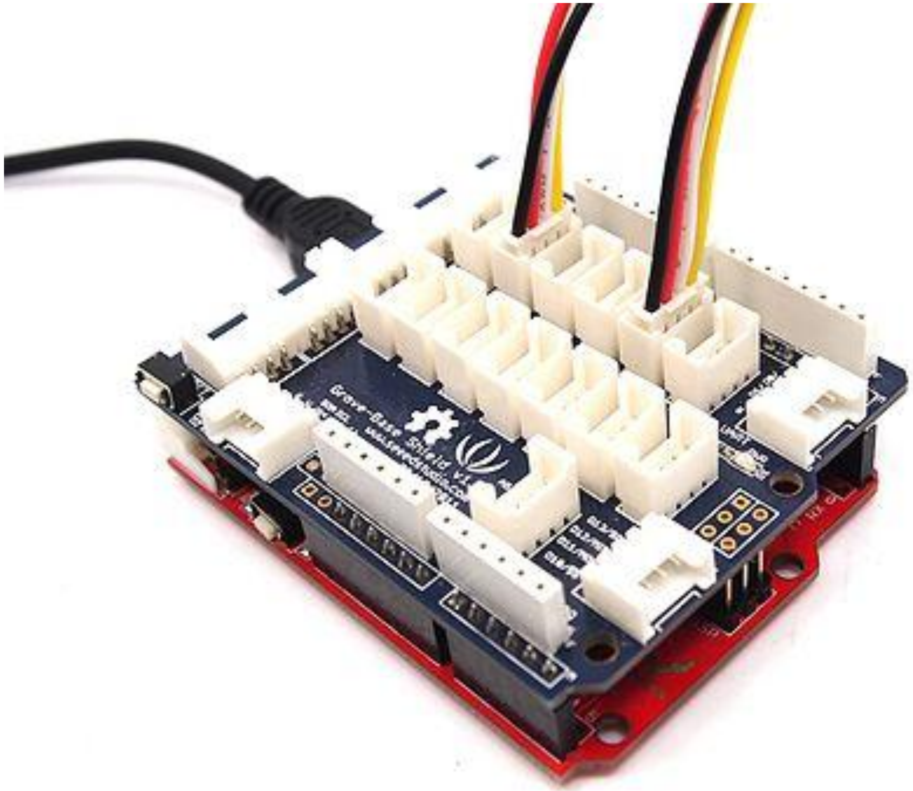
Seeeduino Grove

The Grove system is a modular, safe and easy to use group of items that allow you to minimise the effort required to get started with microcontroller-based experimentation and learning.

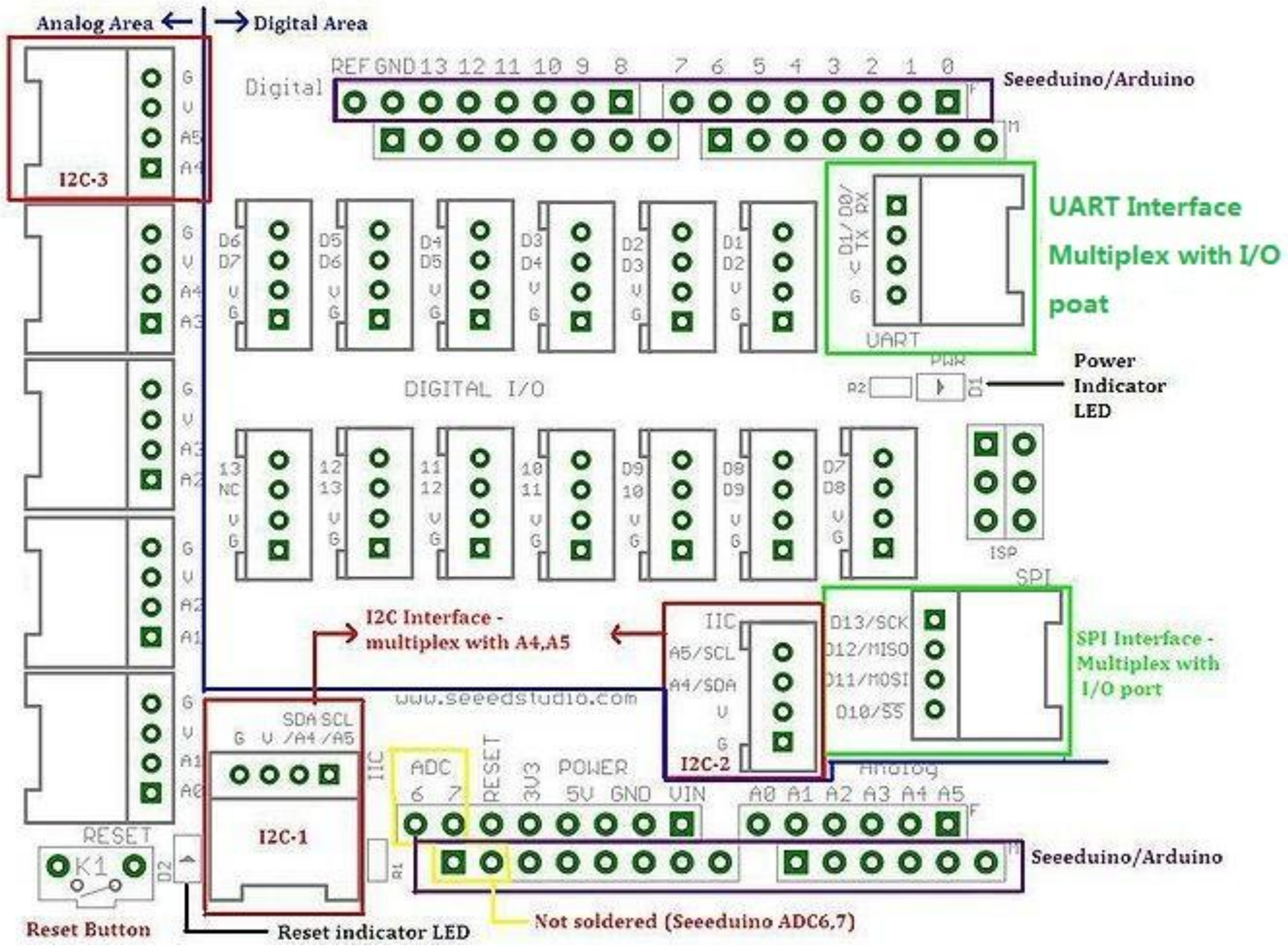
Seeeduino Grove



Seeeduino Grove base shield



Seeeduino Grove base shield



Analog Area ← → Digital Area

Seeeduino/Arduino

UART Interface Multiplex with I/O poat

Power Indicator LED

I2C Interface - multiplex with A4,A5

SPI Interface - Multiplex with I/O port

Seeeduino/Arduino

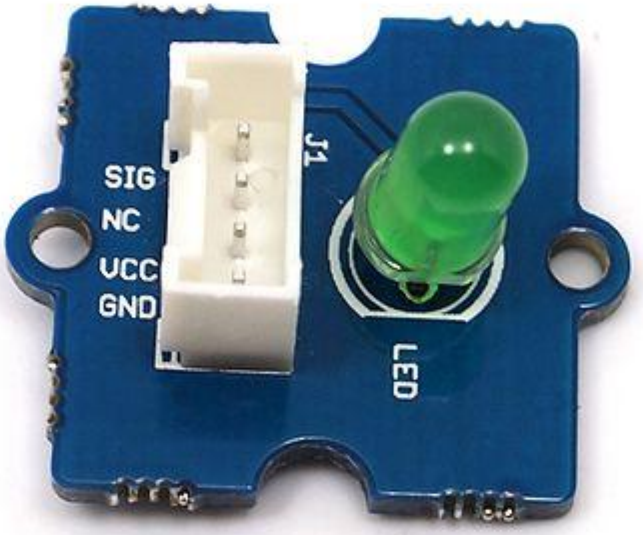
Reset Button

Reset indicator LED

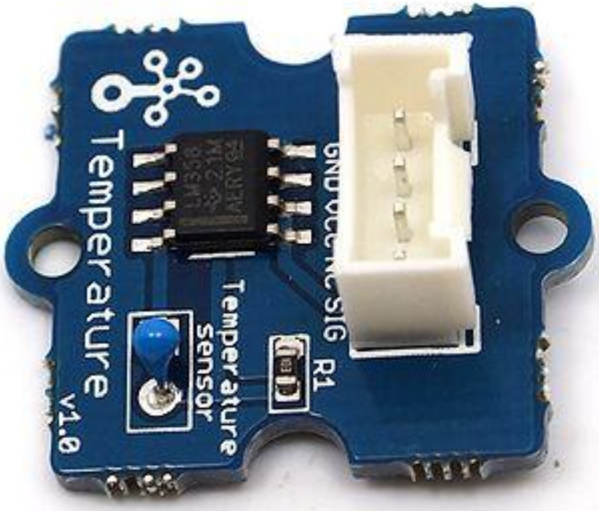
Not soldered (Seeeduino ADC6,7)

Seeeduino Grove units: button

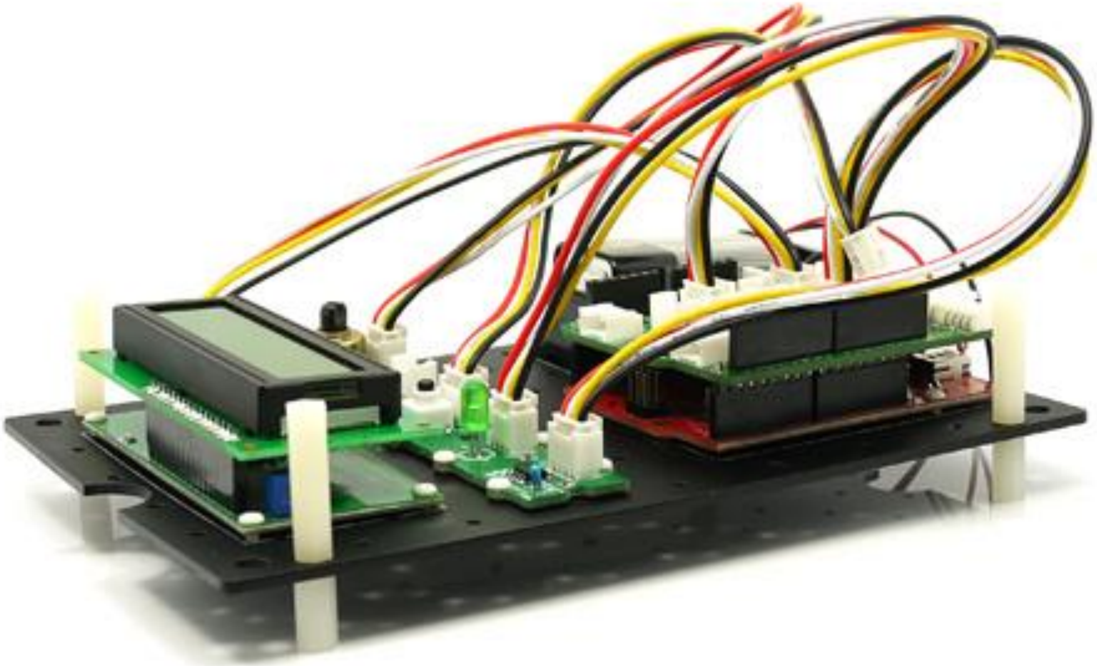
Seeeduino Grove units: LED



Seeeduino Grove units: Temperature



Seeeduino Grove units



Exercise

Feel free to take as many Grove units as you want and experiment with them!