Arduino Workshop

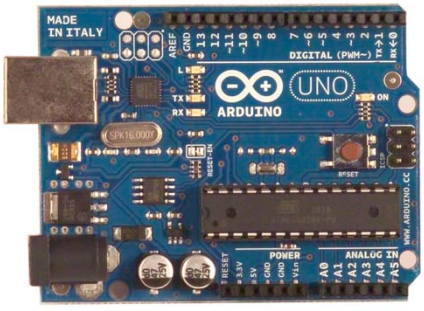
Geert Langereis
HG2.52

Arduino Workshop

- Background on microcontrollers
- Writing a program
- Connecting hardware

Action:

- Installing the software
- Making a LED blink
- Read a button
- Send text to computer

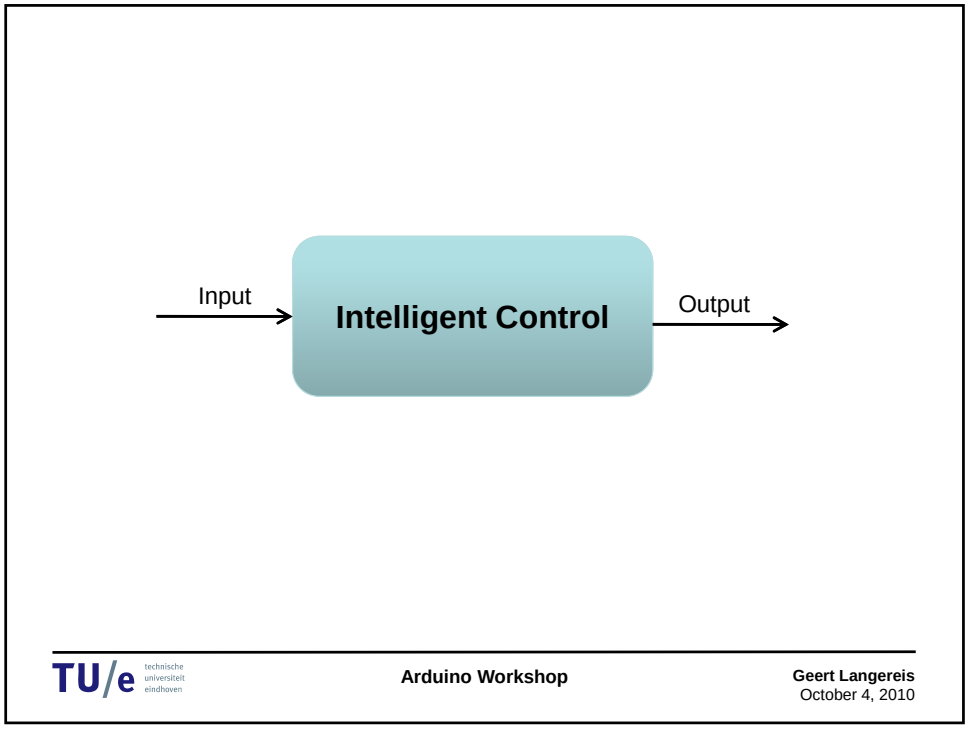


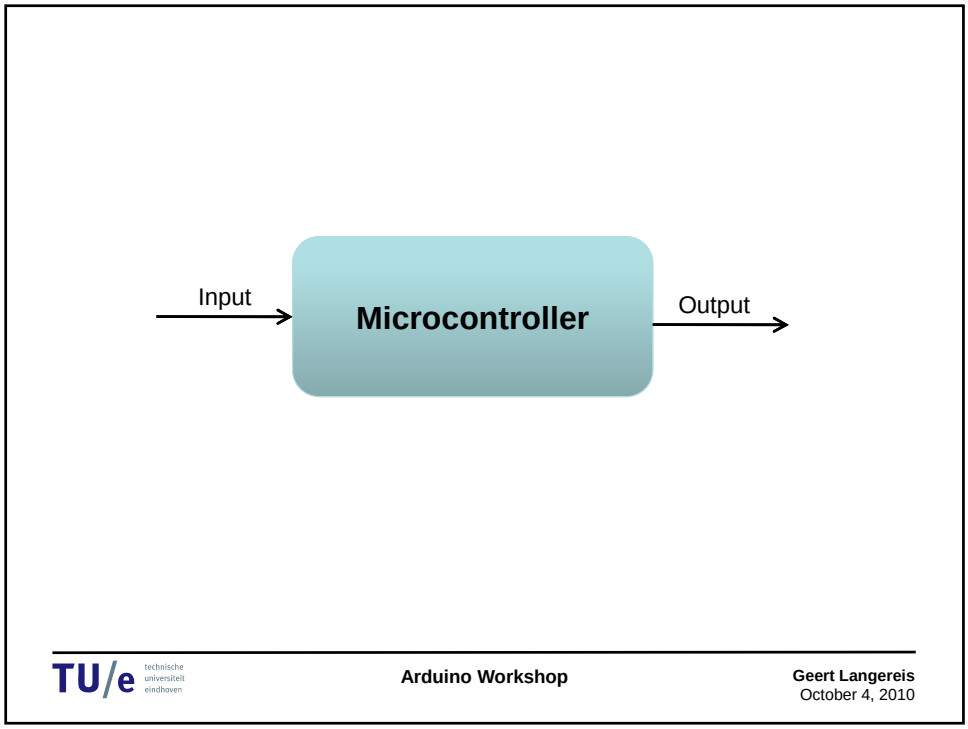
“Microcontrollers are not scary”





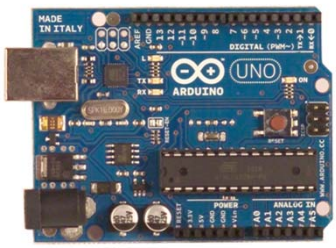

Intelligent products


Input
Intelligent Control
Output


Input
Microcontroller
Output

Why microcontrollers for ID?

- They enable you to turn your concepts into working prototypes
- Can be developed everywhere in your career at low cost
- You can re-use blocks of code and hardware, or re-use work of others

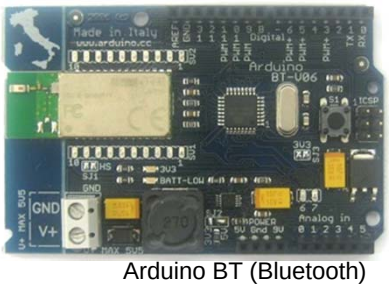


Arduino Uno

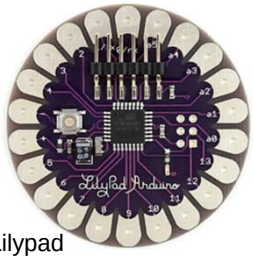
ARDUINO

Arduino Pro Mini

Arduino Nano

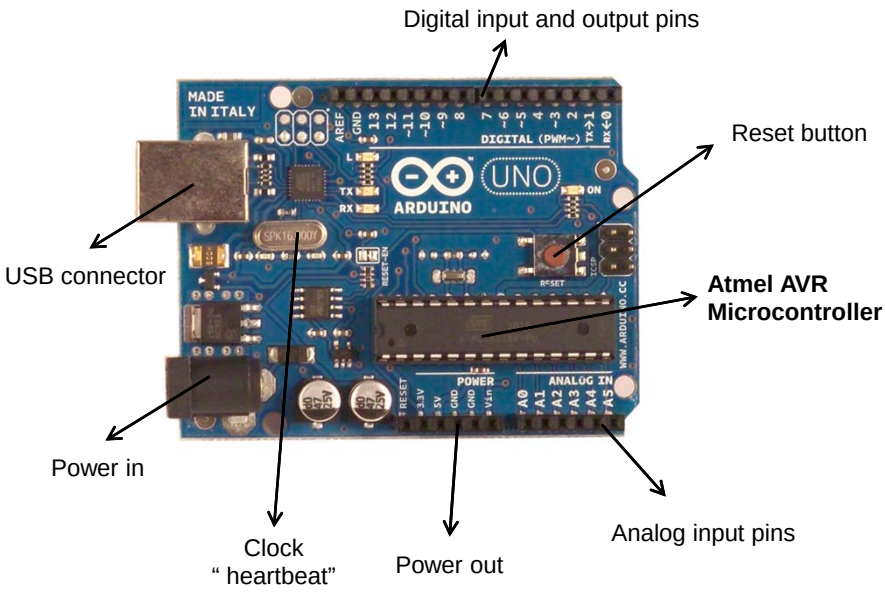


Arduino BT (Bluetooth)

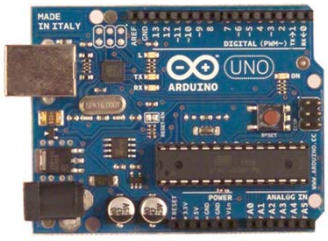


Arduino Lilypad

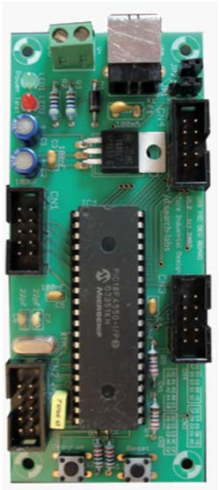
TU/e technische universiteit eindhoven
Arduino Workshop
Geert Langereis
October 4, 2010



Two microcontrollers supported by e-Atelier



Arduino



Microchip PIC

Two microcontrollers supported by e-Atelier

Arduino

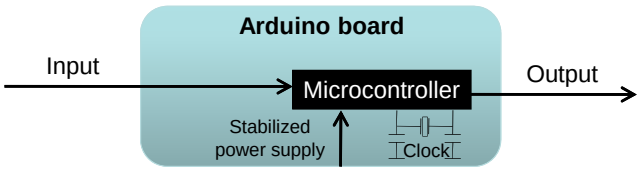
- €24,50
- Programming in Arduino Programming Environment (C language) or AVR-Studio
- Easy to start with
- “Phidgets” available
- Huge community
- Slow and less flexible
- Software development on Mac OS-X and Windows

Microchip PIC

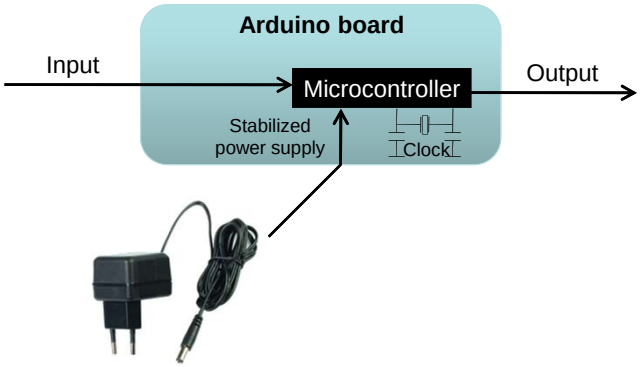
- €25,-
- Programming in MP-Lab (C language)
- More complex
- Fast and flexible
- Windows only software development

The development system

The development system



The development system



The development system

Computer

USB port

Arduino board

Input

Microcontroller

Output

Stabilized power supply

Clock

The development system

Computer

USB port

Arduino board

Input

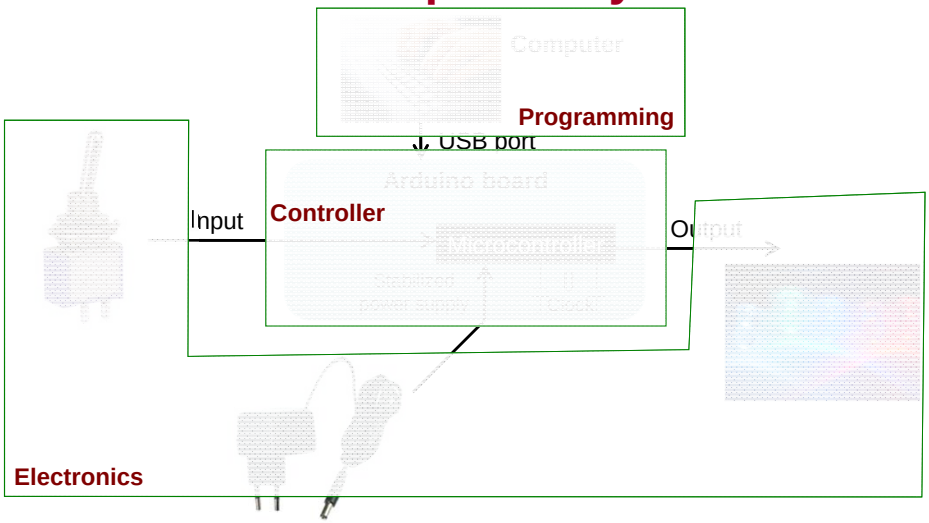
Microcontroller

Output

Stabilized power supply

Clock

The development system



A microcontroller . .

Is a processor with

- Low power
- Low cost
- Dedicated for a single task
- On-board program memory
- On-board data memory
- I/O pins
- Analog and digital out
- Timer/counter circuits
- Bus protocols (serial bus, USB, I^2C, SPI, …)

Registers

Registers are memory locations in the microcontroller which give you access to all the functions.

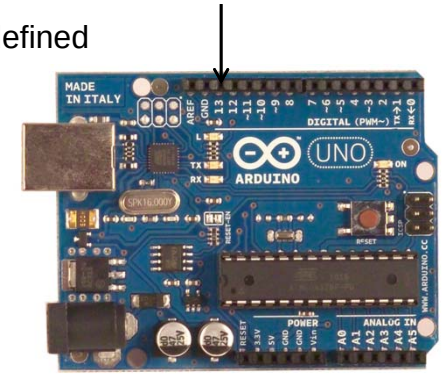
Example: If you write a logical “1” to register “PORTB0”, the output pin “B0” goes high (=5 Volt)

This connects the software world to electronics and hardware

Electronics: Digital Inputs

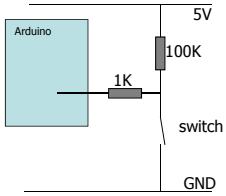
Digital input:

- 0 Volt: Low
- 5 Volt: High
- 2.5 Volt or floating? : undefined

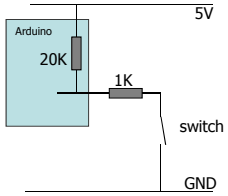


Electronics: Digital Inputs

External pull up
```
pinMode(13, INPUT);
digitalWrite(13, LOW);
int a = digitalRead(13);
```

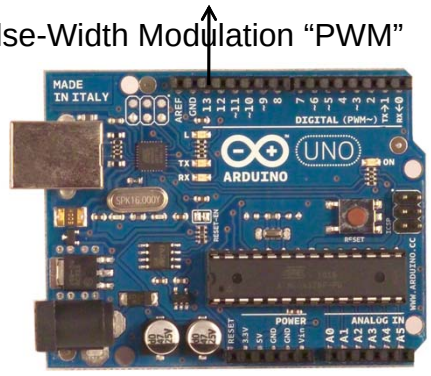


internal pull up
```
pinMode(13, INPUT);
digitalWrite(13, HIGH);
int a = digitalRead(13);
```



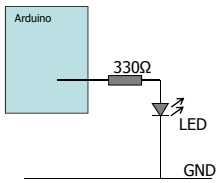
Electronics: Digital Outputs

Digital output:
- 0 Volt: Low
- 5 Volt: High
- 2.5 Volt? You can use Pulse-Width Modulation “PWM”



Electronics: Digital Outputs

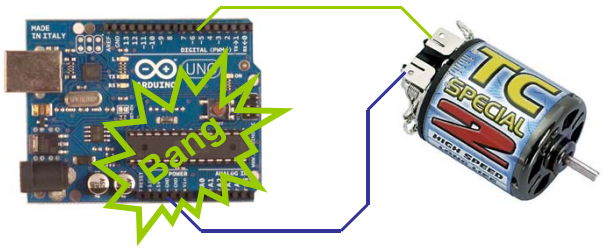
```
pinMode(13, OUTPUT);
digitalWrite(13, LOW);
digitalWrite(13, HIGH);
```



Electronics: Digital Outputs

Digital output:

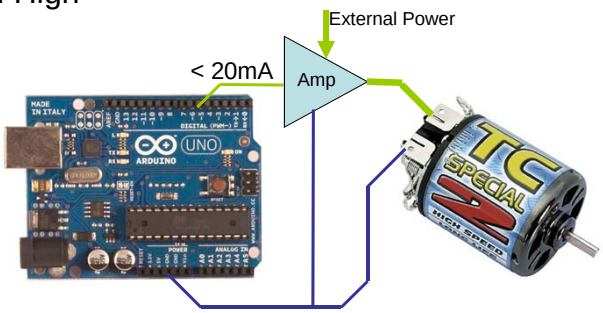
- 0 Volt: Low
- 5 Volt: High



Electronics: Digital Outputs

Digital output:

- 0 Volt: Low
- 5 Volt: High



The development system

Computer

USB port

Arduino board

Input

Microcontroller

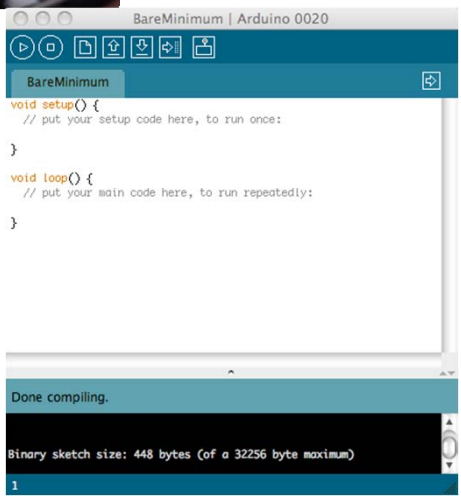
Output

Stabilized power supply

Clock

Computer

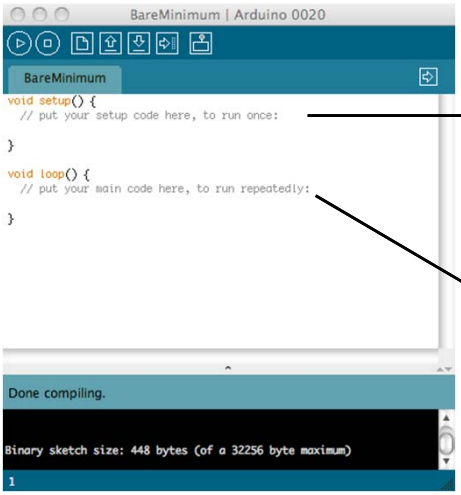
The program



USB port → **Arduino board**

A compiled “hex”-file is Sent to the program memory of the board

The program



First, tell Arduino what it is:
- Declare pins as input or output
- Set global variables

Then tell it what to do:
- Read inputs
- Do calculations
- Set outputs

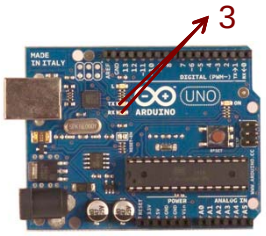
This is done continuously, in a loop

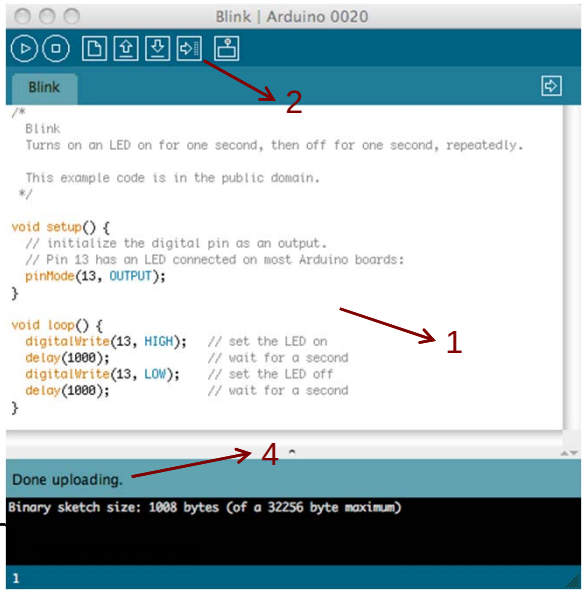
The program

```
/*  Configuration:
First, tell Arduino what it is */

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

/*  Then loop:
Tell Arduino what to do */

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
}
```


Compiling and uploading code

1. Type code in text window
2. Push 'upload' button
3. Check if TX and RX Leds are blinking rapidly
4. If the 'Done uploading' message displays, the Arduino is ready.





Everything you need is www.arduino.cc

- Installation
- Download
- Help
- Forum
- Tutorials
- Hardware examples
- Software examples
- Reference

TU/e technische universiteit eindhoven

Installation 1: The Environment

Download

Arduino 0020 (release notes), hosted by Google Code:

- Windows
- Mac OS X

Arduino 0019:

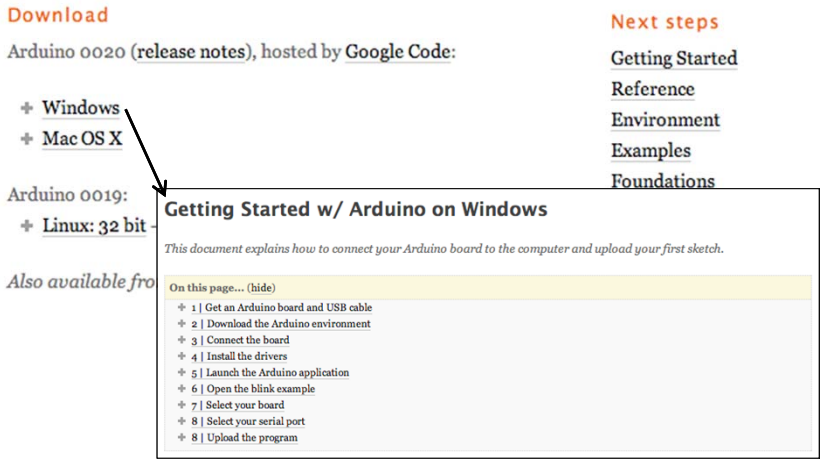
- Linux: 32 bit - check here for compatibility

Also available from Arduino.cc: Windows, Mac OS X, Linux (32bit)

Next steps

Getting Started
Reference
Environment
Examples
Foundations
FAQ

Installation 1: The environment



Installation 2: The COM port

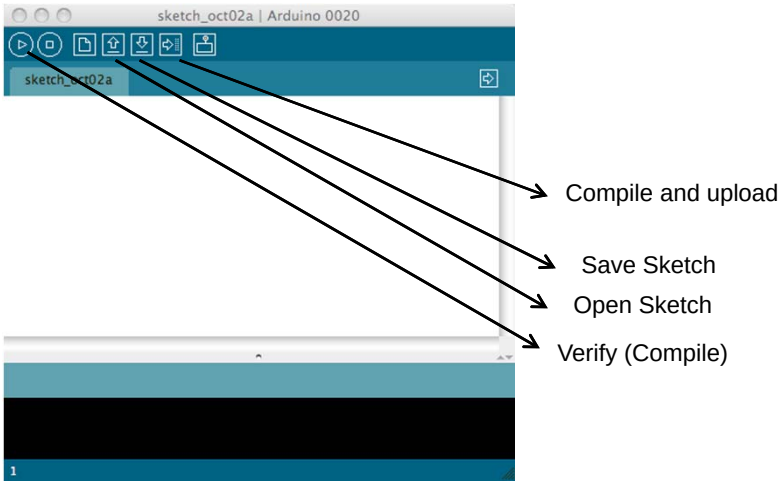
A “Virtual COM port” is needed to communicate with the board.

For the Arduino UNO: plug in the Arduino
- On Mac OS-X, the VCP driver is installed automatically
- On Windows, you have to point to the .inf file in the “drivers” directory of the Arduino software

For the Arduino Duemilanove, you have to point to the FTDI install directory which is in the “drivers” directory as well

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

The Arduino Programming Environment

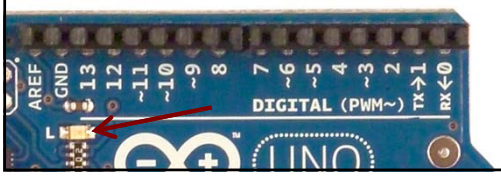


Settings

- Let the Arduino programming environment know which board you have
  - "Tools" menu ➔ "Board"
- Let the Arduino programming environment know to which port the bard is connected
  - "Tools" menu ➔ "Serial port"

Blink

```
/*  comment */

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}
}
```



Blink with button

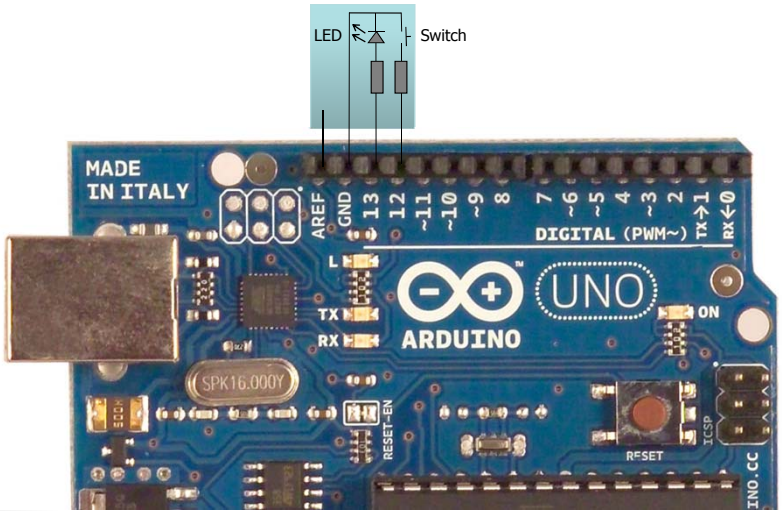
- The period of “Blink” is 2 x 1000ms = 2 sec
- Now we want: if we push a button, the period should become 0.5 sec

- To do:
  - Make a button
  - Read a button
  - Change the “1000” in “delay” depending on the button

Blink and read button

```
/*  comment */

void setup() {
        // initialize the digital pin as an output.
        // Pin 13 has an LED connected on most Arduino boards:
        pinMode(13, OUTPUT);
        pinMode(12, INPUT);
        digitalWrite(12, HIGH);     // activate pull-up resistor
}

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
}
```


Some hardware for Blink with button



```
void setup() {
        // initialize the digital pin as an output.
        // Pin 13 has an LED connected on most Arduino boards:
        pinMode(13, OUTPUT);
        pinMode(12, INPUT);
        digitalWrite(12, HIGH);    // activate pull-up resistor
}

void loop() {
        if (digitalRead(12) == HIGH) {
                // Do something
        }
        else {
                // Do something else
        }
        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
}
```



```
int wait = 1000;

void setup() {
        // initialize the digital pin as an output.
        // Pin 13 has an LED connected on most Arduino boards:
        pinMode(13, OUTPUT);
        pinMode(12, INPUT);
        digitalWrite(12, HIGH);    // activate pull-up resistor
}

void loop() {
        if (digitalRead(12) == HIGH) {
                // Do something
        }
        else {
                // Do something else
        }
        digitalWrite(13, HIGH);    // set the LED on
        delay(wait);               // wait for a second
        digitalWrite(13, LOW);     // set the LED off
        delay(wait);               // wait for a second
}
```

```
int wait = 1000;

void setup() {
        // initialize the digital pin as an output.
        // Pin 13 has an LED connected on most Arduino boards:
        pinMode(13, OUTPUT);
        pinMode(12, INPUT);
        digitalWrite(12, HIGH);    // activate pull-up resistor
}

void loop() {
        if (digitalRead(12) == HIGH) {
                wait = 1000; // Wait 1 sec
        }
        else {
                wait = 250;  // Wait 0.25 sec
        }
        digitalWrite(13, HIGH);    // set the LED on
        delay(wait);               // wait for a second
        digitalWrite(13, LOW);     // set the LED off
        delay(wait);               // wait for a second
}
```


Create some interaction: The "Physical Pixel" example

```
const int ledPin = 13; // the pin that the LED is attached to
int incomingByte;      // a variable to read incoming serial data into

void setup() {
  Serial.begin(9600); // initialize serial communication
  Serial.println("Good morning"); // Send text to computer
  pinMode(ledPin, OUTPUT); // initialize the LED pin as an output
}

void loop() {
    if (Serial.available() > 0) {
       incomingByte = Serial.read();
       if (incomingByte == 'H') digitalWrite(ledPin, HIGH);
       if (incomingByte == 'L') digitalWrite(ledPin, LOW);
    }
  }
}
```

How to proceed?

- Play with more pre-installed Sketches (all descriptions on http://arduino.cc/en/Tutorial/HomePage)
- Find new functions in the reference (http://arduino.cc/en/Reference/HomePage)

- DG233 “Introducing Microcontrollers – Arduino and Beyond”
- Don’t be scared by “Making things talk”: the book is about connecting to the internet and less to interfacing with electronics