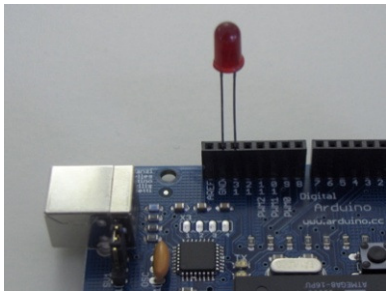
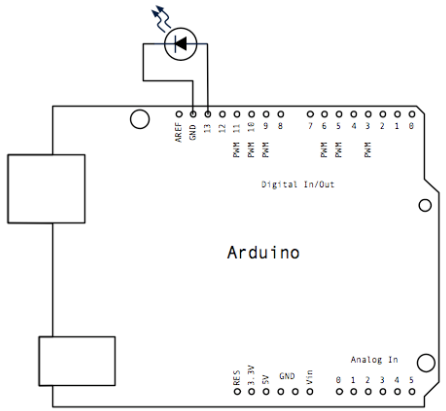
Digital I/O

Quick Recap

Blink : Hardware

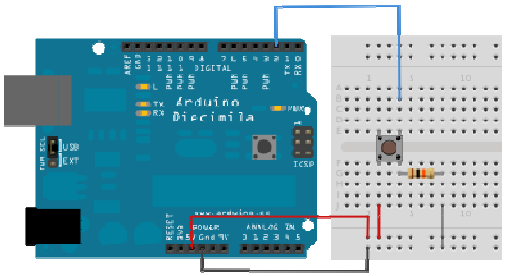
```
int ledPin =  13;     // LED connected to digital pin 13

// The setup() method runs once, when the sketch starts

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

// the loop() method runs over and over again,
// as long as the Arduino has power

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

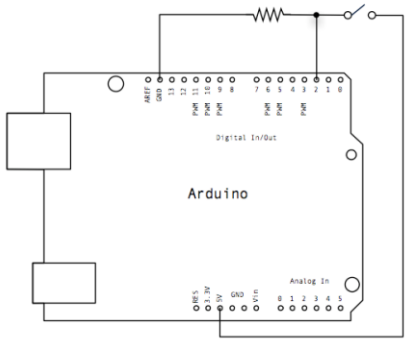




Arduino : Button

```
// constants won't change. They're used here to
// set pin numbers:
const int buttonPin = 2;     // the number of the pushbutton pin
const int ledPin =  13;      // the number of the LED pin

// variables will change:
int buttonState = 0;         // variable for reading the pushbutton status

void setup() {
  // initialize the LED pin as an output:
  pinMode(ledPin, OUTPUT);
  // initialize the pushbutton pin as an input:
  pinMode(buttonPin, INPUT);
}

void loop(){
  // read the state of the pushbutton value:
  buttonState = digitalRead(buttonPin);

  // check if the pushbutton is pressed.
  // if it is, the buttonState is HIGH:
  if (buttonState == HIGH) {
    // turn LED on:
    digitalWrite(lepPin, HIGH);
  }
  else {
    // turn LED off:
    digitalWrite(lepPin, LOW);
  }
}
```





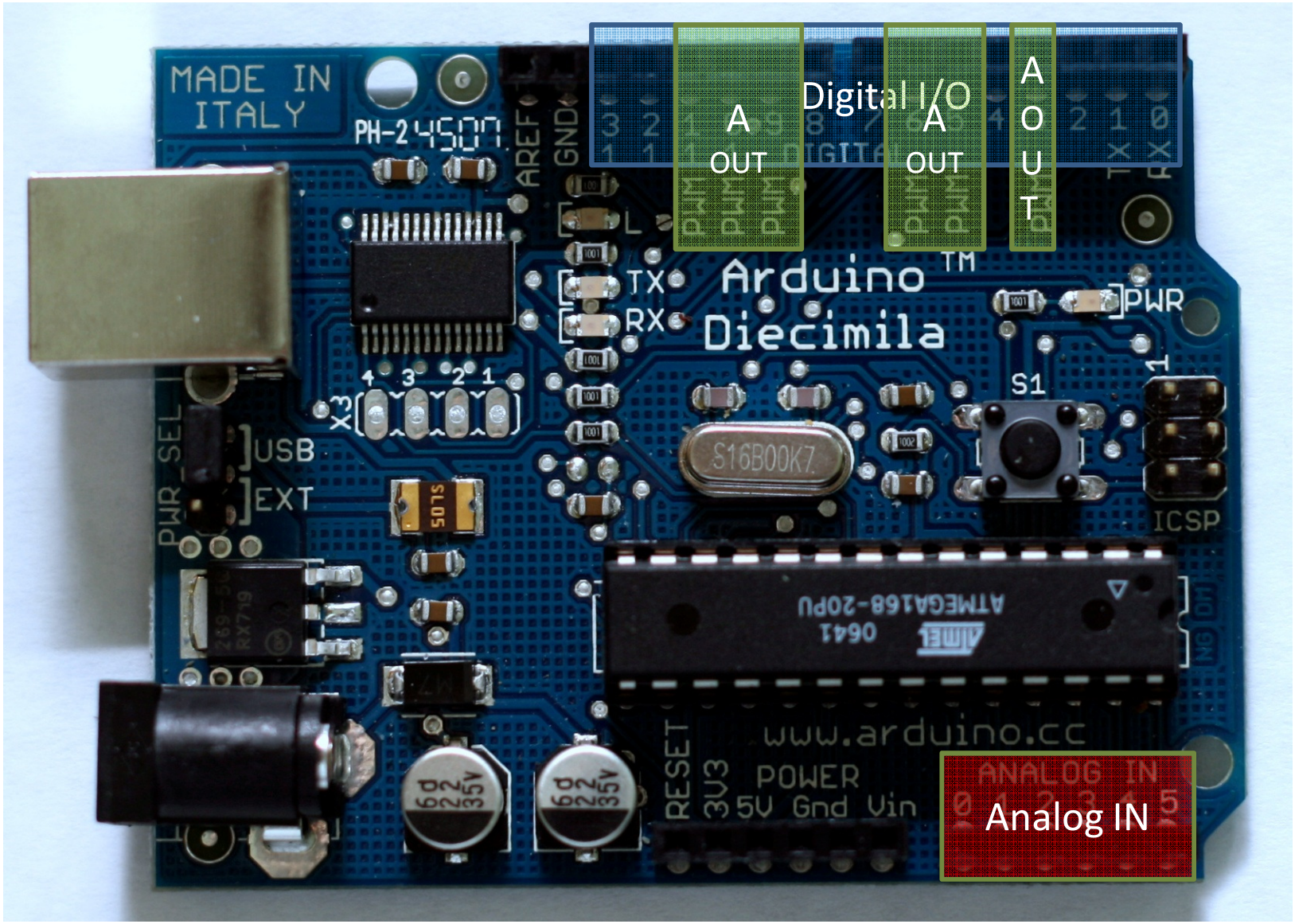
Digital I/O

- pinMode(Pin#, INPUT/OUTPUT); // Assign PIN
- digitalRead(Pin#); // Digital Input
- digitalWrite(Pin#, State); // Digital Output

Analog I/O

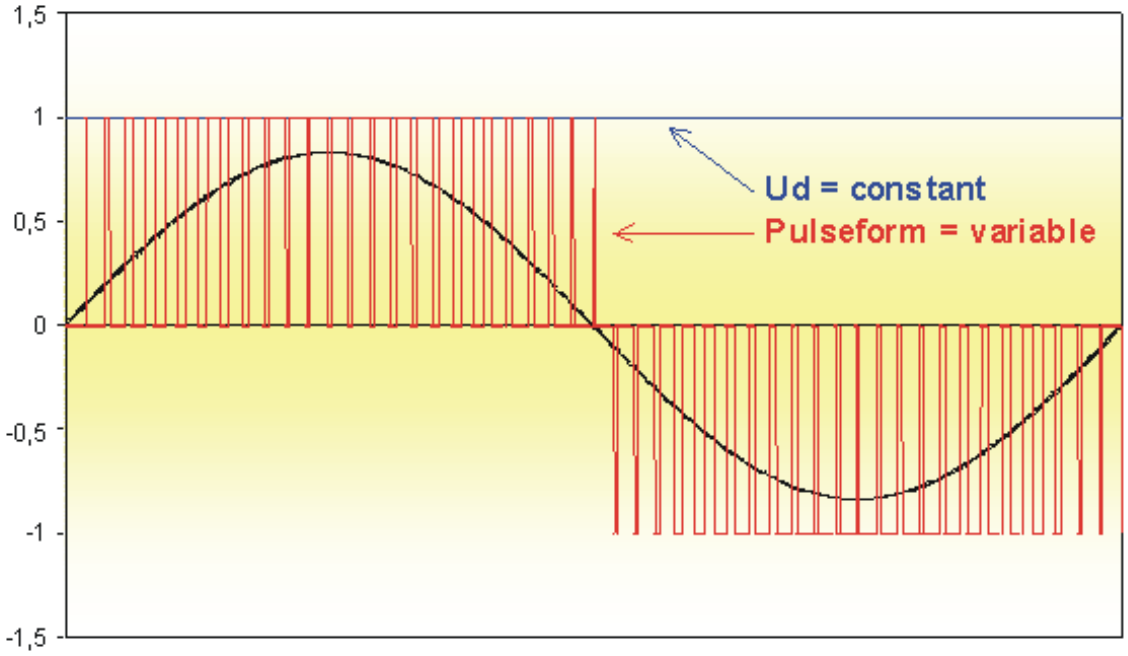
- pinMode(Pin#, OUTPUT); // Assign PIN
- analogRead(Pin#); // Analog Input
- analogWrite(Pin#, duty Cycles); // Analog Output

Closer look at the board



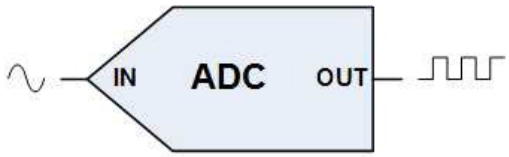
Analog Digital Convertors (ADCs)

- Arduino has a built in 10-bit ADC which keeps checking the value of the voltage input and returns back the values.



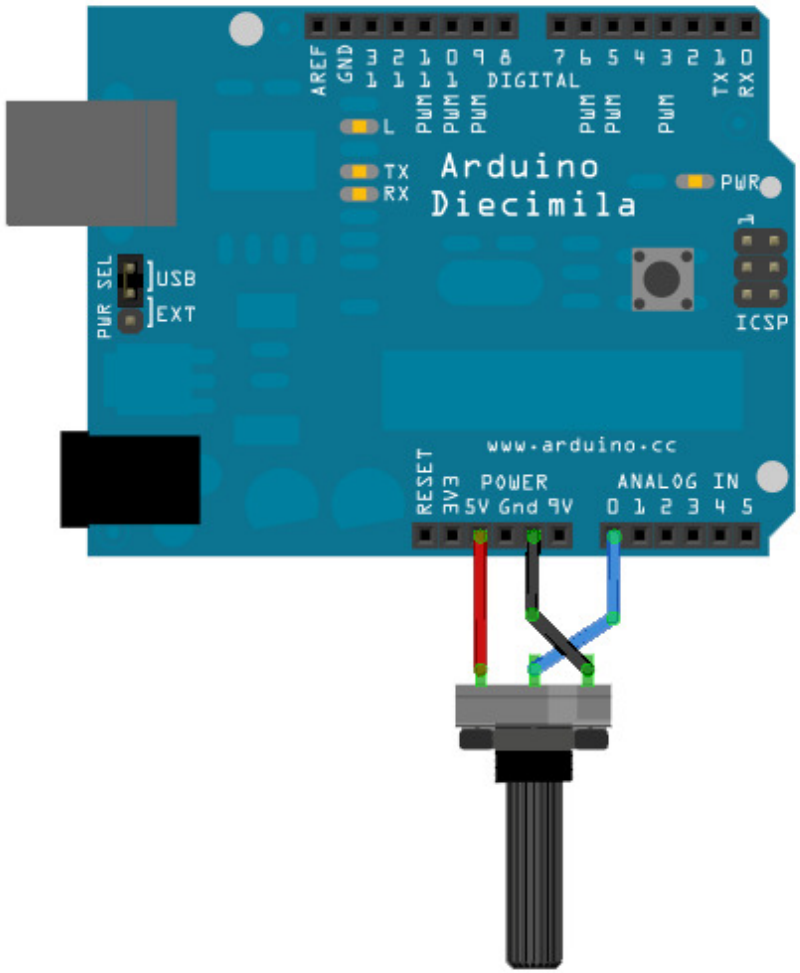
It takes (0.0001 s) per reading.

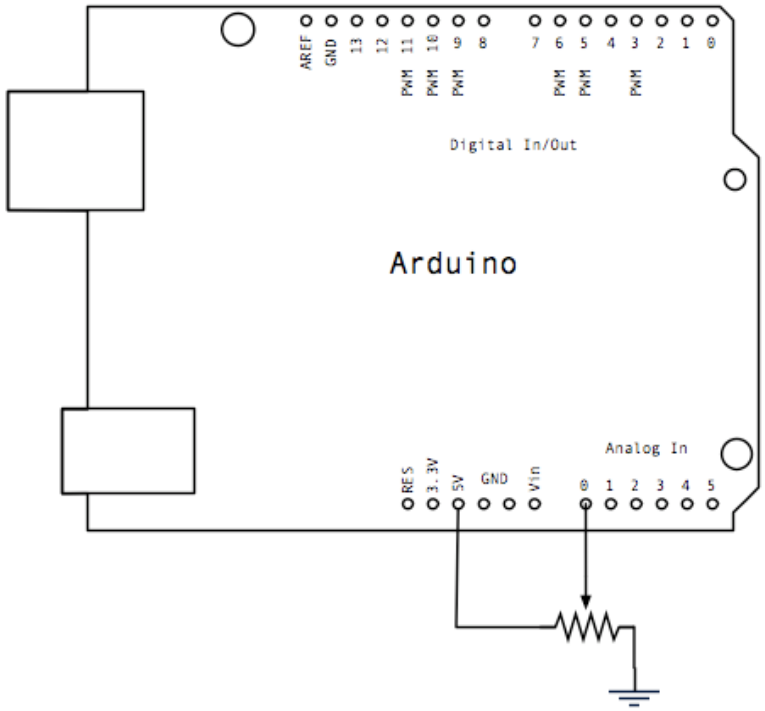
10,000 readings / sec.



ELECTRICAL SYMBOL FOR ANALOG TO DIGITAL CONVERTER (ADC)

Analog In


AREF GND 13 12 11 10 9 8 7 6 5 4 3 2 1 0
DIGITAL
PWM PWM PWM PWM PWM PWM
TX RX
L
TX
RX
Arduino
Diecimila
PWR
PWR SEL
USB
EXT
ICSP
www.arduino.cc
RESET 3V3 5V Gnd 9V
POWER
ANALOG IN 0 1 2 3 4 5

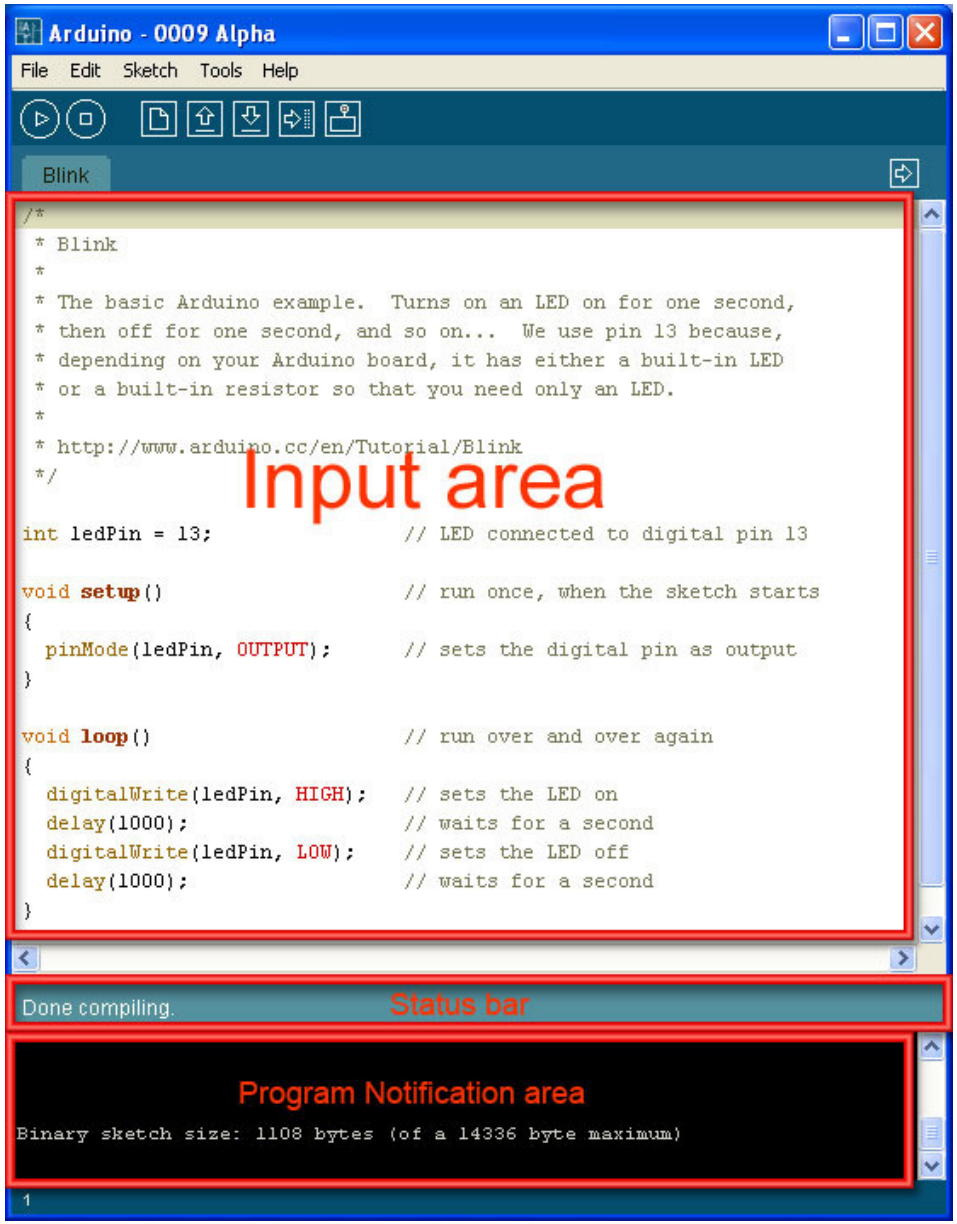

AREF GND 13 12 11 10 9 8 7 6 5 4 3 2 1 0
PWM PWM PWM PWM PWM PWM
Digital In/Out
Arduino
Analog In
RES 3.3V 5V GND Vin 0 1 2 3 4 5

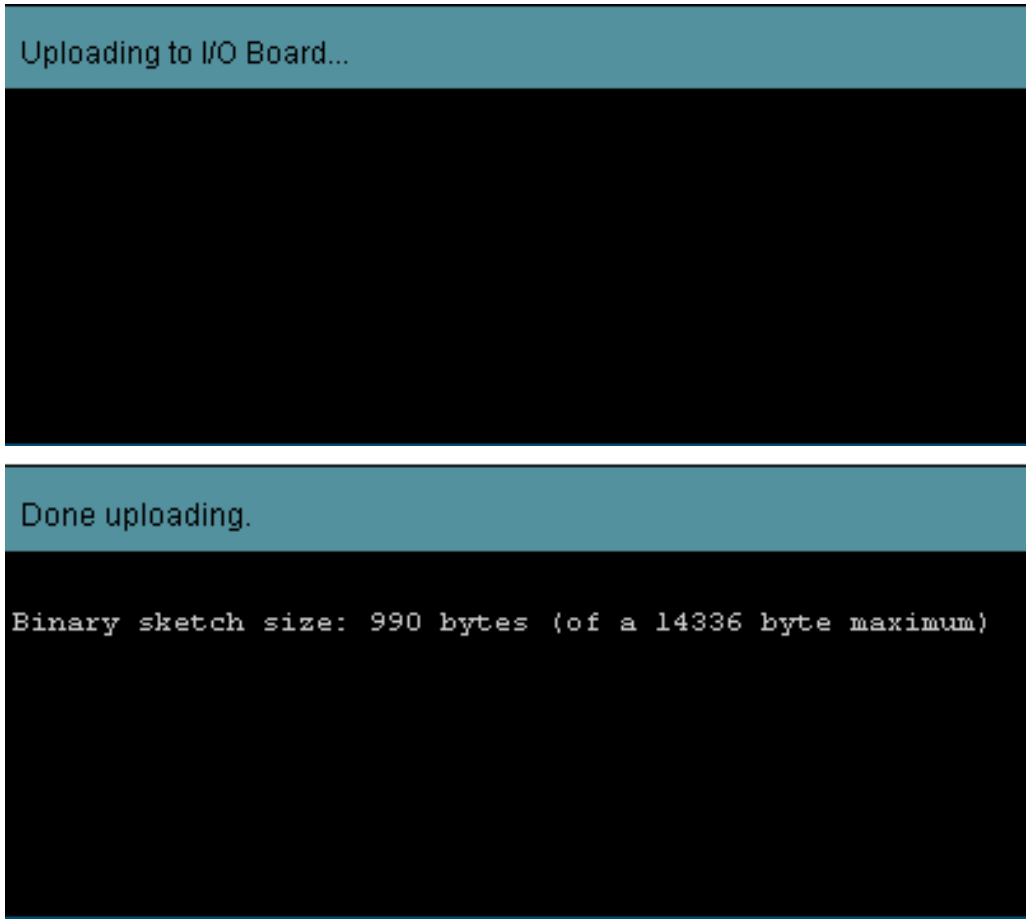
Analog In: Code

FILE> Examples> Analog> AnalogInput

```
int sensorPin = 0;          // select the input pin for the potentiometer
int ledPin = 13;            // select the pin for the LED
int sensorValue = 0;        // variable to store the value coming from the sensor

void setup() {
  // declare the ledPin as an OUTPUT:
  pinMode(ledPin, OUTPUT);
}

void loop() {
  // read the value from the sensor:
  sensorValue = analogRead(sensorPin);
  // turn the ledPin on
  digitalWrite(ledPin, HIGH);
  // stop the program for <sensorValue> milliseconds:
  delay(sensorValue);
  // turn the ledPin off:
  digitalWrite(ledPin, LOW);
  // stop the program for for <sensorValue> milliseconds:
  delay(sensorValue);
}
```

Arduino platform





Memory

There are three pools of memory in the microcontroller used on Arduino boards (ATmega168):

- **Flash memory** (program space), is where the Arduino sketch is stored.
- **SRAM** (static random access memory) is where the sketch creates and manipulates variables when it runs.
- **EEPROM** (Electronically Erasable Programmable Read Only Memory) is memory space that programmers can use to store long-term information.

The ATmega168 chip has the following amounts of memory:

- Flash 16k bytes (of which 2k is used for the bootloader)
- SRAM 1024 bytes
- EEPROM 512 bytes

Exercise

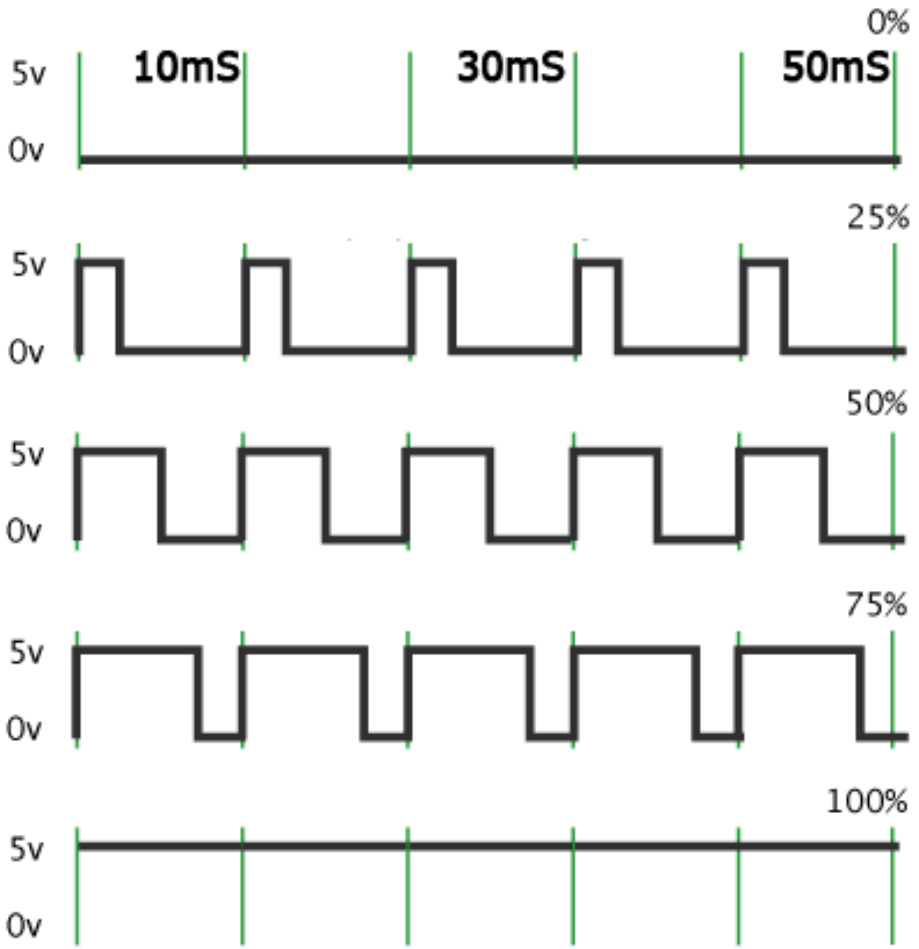
- File > Examples > Digital > Blink

Digital : ON / OFF

High intensity / No intensity

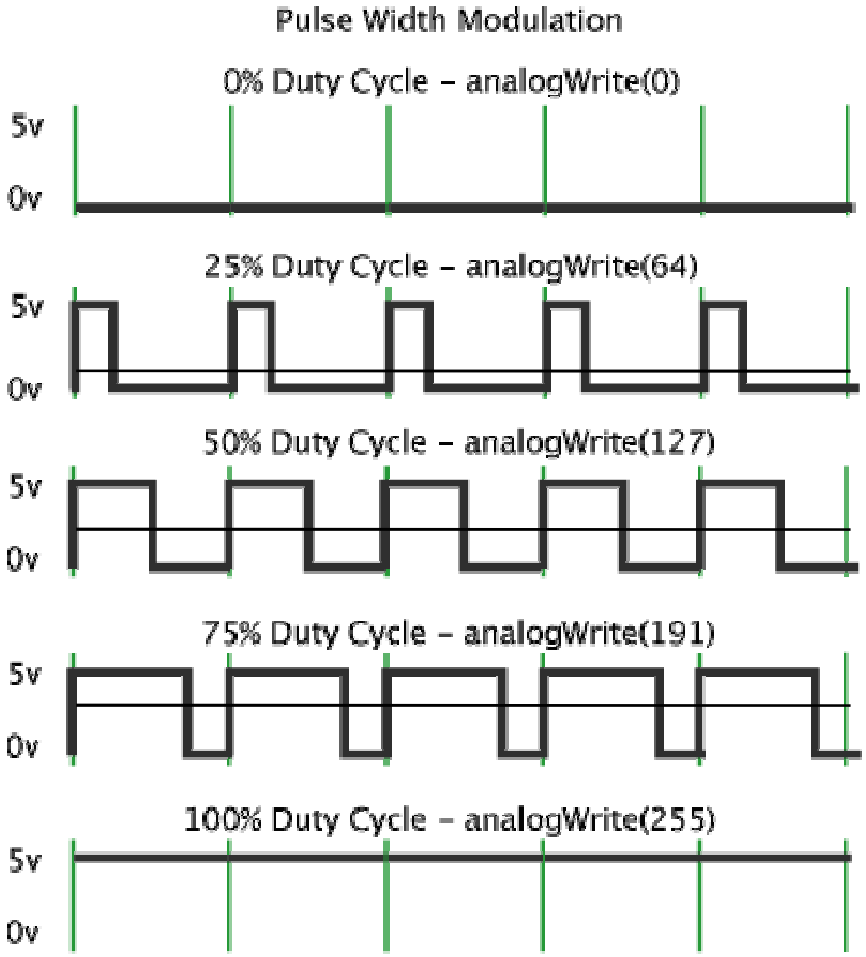
Analog is in between.

```
digitalWrite(ledPin, HIGH);
delay(0);
digitalWrite(ledPin, LOW);
delay(0);
```



PWM

- Pulse Width Modulation
- Digital way of getting an analog current supply.
- Pulse: sudden increase and decrease in current flow.
- High = 256 (0-255)



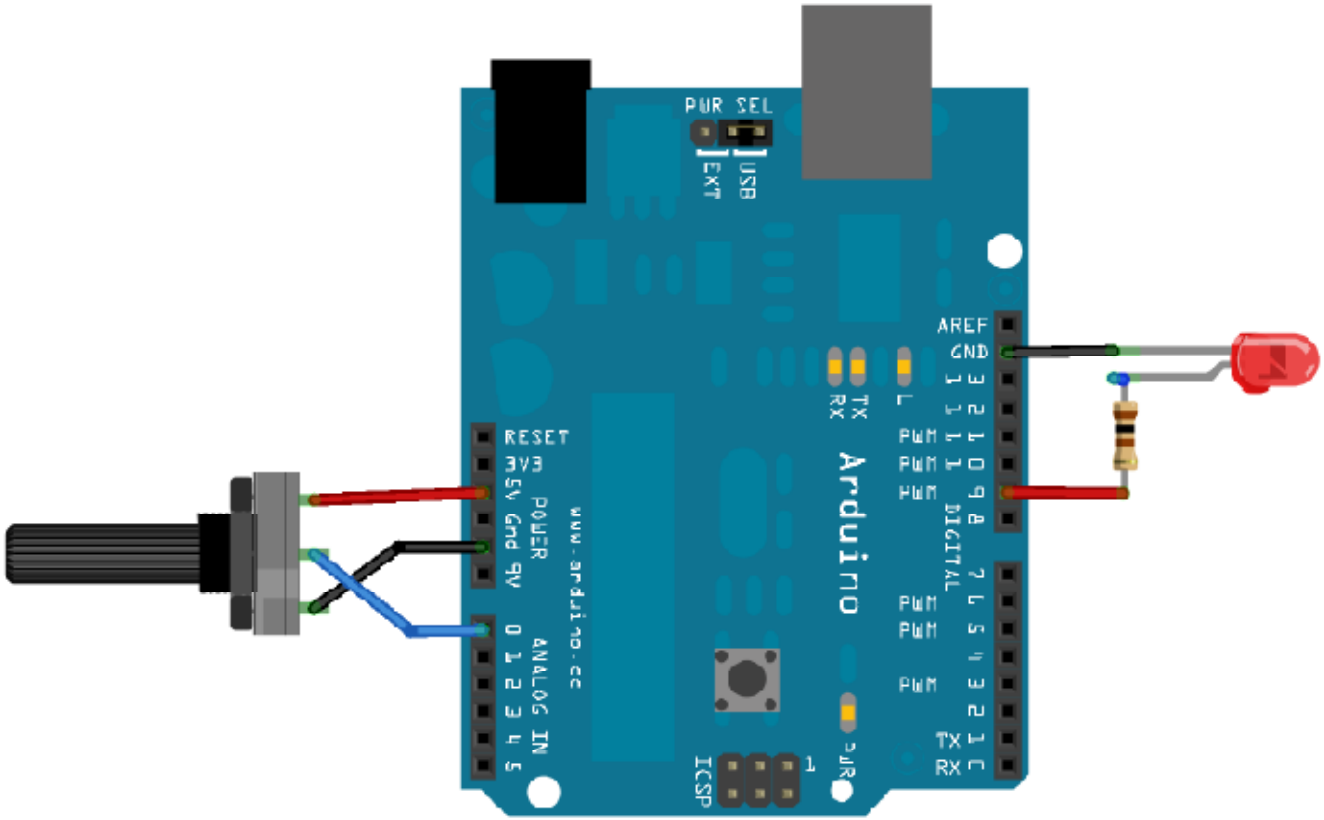
Analog I/O: LED

FILE> Examples> Analog> AnalogInput

```
int sensorPin = 0;                // select the input pin for the potentiometer
int ledPin = 9;                   // select the PWM pin for the LED
int sensorValue = 0;              // variable to store the value coming from the sensor

void setup() {
  // declare the ledPin as an OUTPUT:
  pinMode(ledPin, OUTPUT);
}

void loop() {
  // read the value from the sensor:
  sensorValue = analogRead(sensorPin);
  // turn the ledPin on according to potentiometer's modulation
  analogWrite(ledPin, sensorValue);
}
```

http://led.linear1.org/1led.wiz

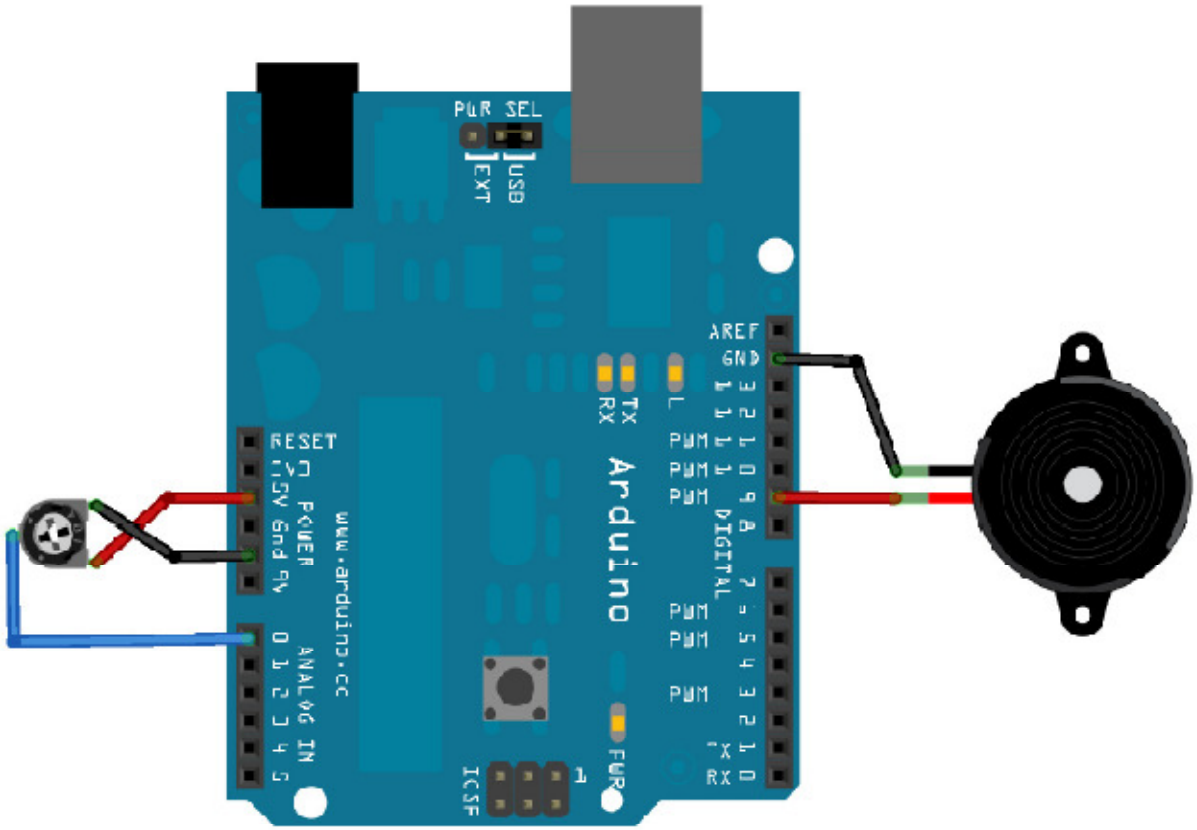
Analog I/O: LED



Analog I/O: LED

```
int sensorPin = 0;                // select the input pin for the potentiometer
int ledPin = 9;                   // select the PWM pin for the LED
int sensorValue = 0;              // variable to store the value coming from the sensor
Int outputData = 0;               // Assigning integer value to the variable outputData

void setup() {
  // declare the ledPin as an OUTPUT:
  pinMode(ledPin, OUTPUT);
}

void loop() {
  // read the value from the sensor:
  sensorValue = analogRead(sensorPin);
  // map it to the range of the analog out:
  outputData = map(sensorValue, 0, 1023, 0, 255);
  // turn the ledPin on according to potentiometer's modulation
  analogWrite(ledPin, sensorValue);
}
```

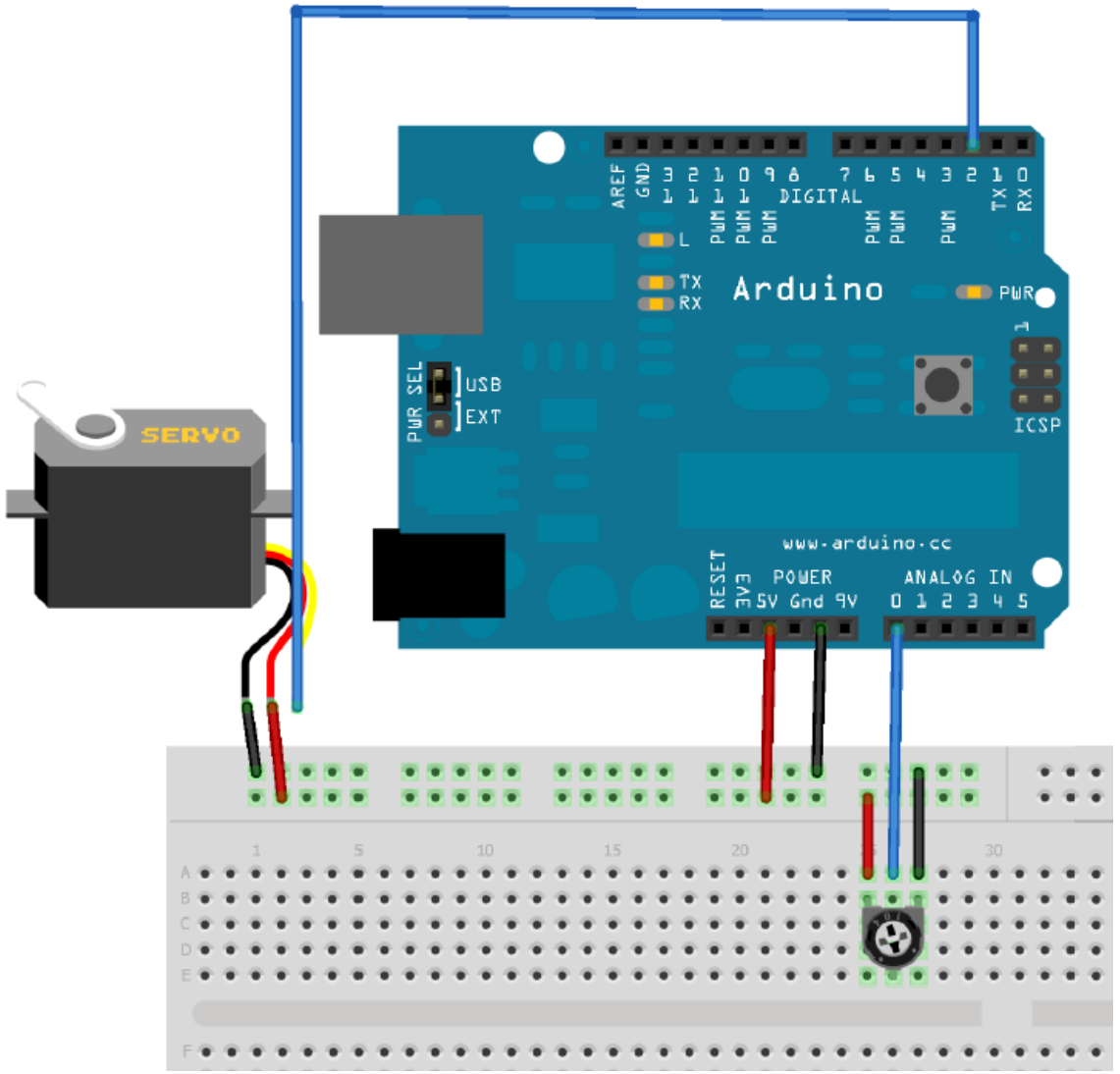
Analog I/O : Sound



Analog I/O: Sound

```
int sensorPin = 0;                // select the input pin for the potentiometer
int pPin = 9;                     // select the PWM pin for the speaker
int sensorValue = 0;              // variable to store the value coming from the sensor

void setup() {
  // declare the ledPin as an OUTPUT:
  pinMode(ledPin, OUTPUT);
}

void loop() {
  // read the value from the sensor:
  sensorValue = analogRead(sensorPin);
  // turn the ledPin on according to potentiometer's modulation
  analogWrite(pPin, sensorValue);
}
```

Arduino I/O : Servo



Analog I/O: Servo

http://www.arduino.cc/playground/ComponentLib/Servo

```
#include <SoftwareServo.h>

SoftwareServo myservo;  // create servo object to control a servo

int potpin = 0;  // analog pin used to connect the potentiometer
int val;    // variable to read the value from the analog pin

void setup()
{
  myservo.attach(2);  // attaches the servo on pin 2 to the servo object
}

void loop()
{
  val = analogRead(potpin);          // reads the value of the potentiometer (value between 0 and 1023)
  val = map(val, 0, 1023, 0, 179);     // scale it to use it with the servo (value between 0 and 180)
  myservo.write(val);                  // sets the servo position according to the scaled value
  delay(15);                           // waits for the servo to get there

  SoftwareServo::refresh();
}
```

Coding

- Variables
- Data types
- Control Structures
- Operators
- Functions

http://arduino.cc/en/Reference/HomePage