March 3, 2012: Draft
March 21, 2012: Release

Microcontroller Workshop
Build your own products

The University of Tokushima
Akinori Tsuji

Contact Information：
2-1, Minamijyosanjima-cho, Tokushima 770-8506, Japan
TEL/FAX： +81-88-656-7485
E-mail:：a-tsuji@is.tokushima-u.ac.jp

Agenda

1. Why the microcontrollers

2. How to start

3. How it works

4. Programming
   I/O Port, A/D Converter, Timer, Interrupt,
   Serial Communication

5. Sensors and Actuator
   Light, Touch, Temperature, Motor

6. Building a Robot (* if possible)

Set up development environment

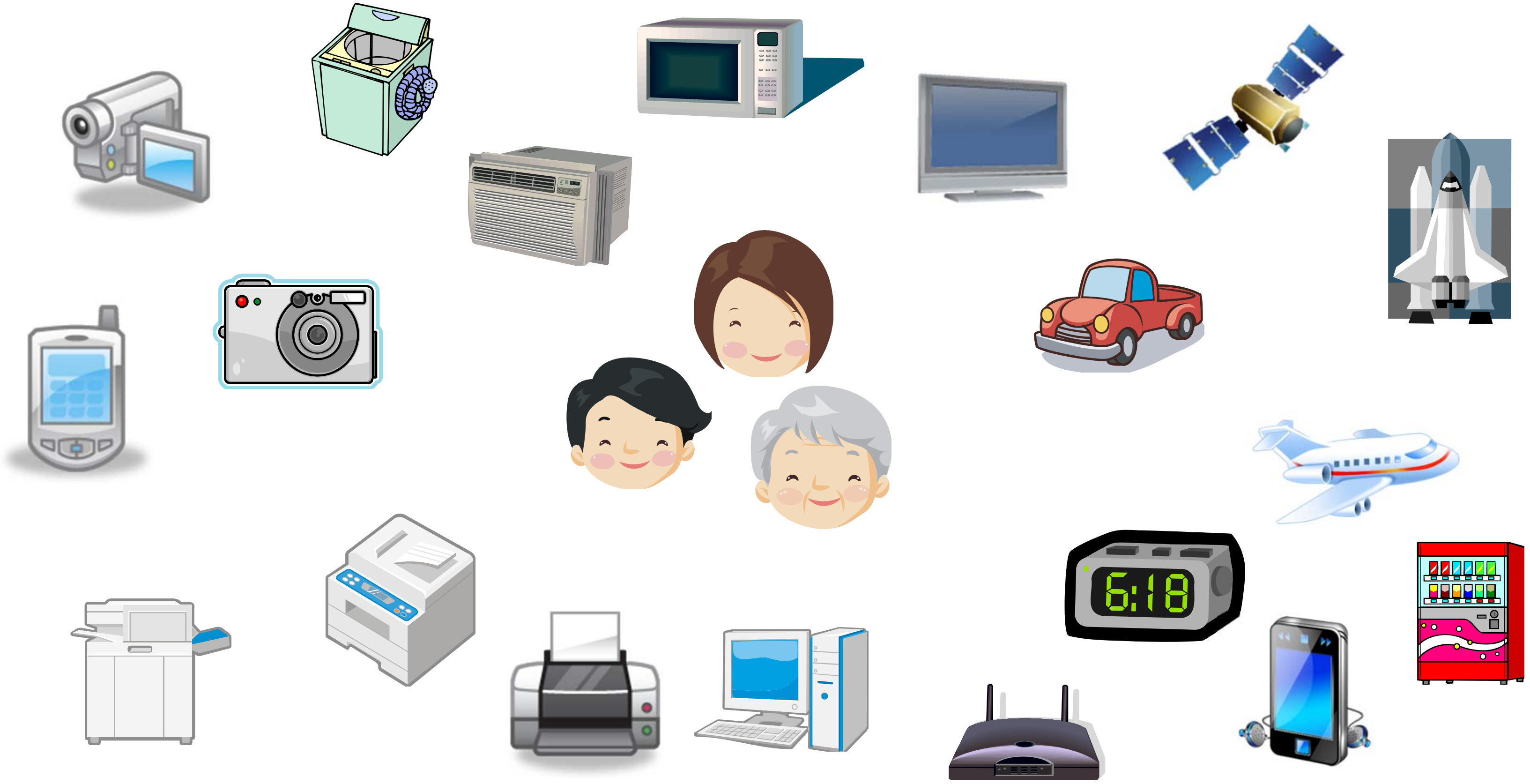
Day 1

Estimate: 2 hours

2012/3/21(Wed) 10:00—12:00

1. Why the Microcontrollers

A microcontroller is everywhere around you, but everyone does not notice them



1.1 Motivation

A microcontroller covers a lot of fields and applications, use your imaginative power

Easy to make a prototype

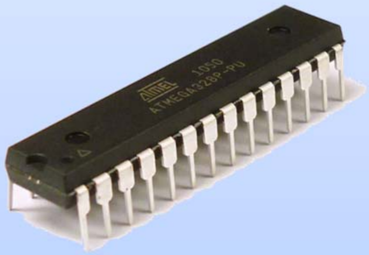
Full scratch

Get a lot of knowledge

Many applications

Essential of our life

Make original products



Long-life technology

Study Electrical and Electronics

Realize your idea

Fusion of software and hardware

Invent new one

Useful for the research

Inexpensive

Know how products work

Build your own instruments

Embedded system programming

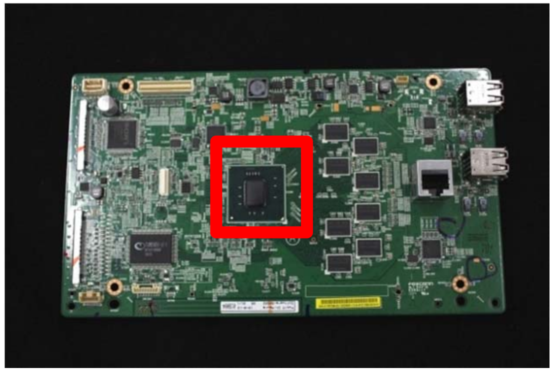
Scientific approach

Interface to the nature

and much more

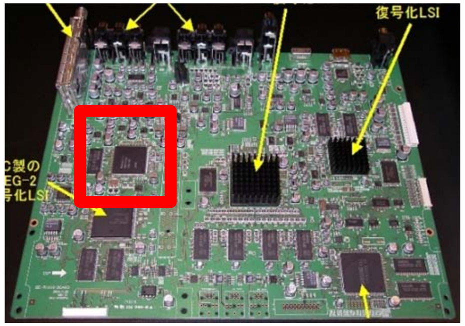
1.2 Inside of the Products

We open and found it !

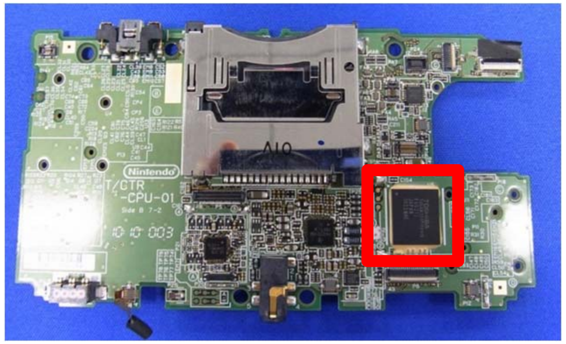


Digital TV

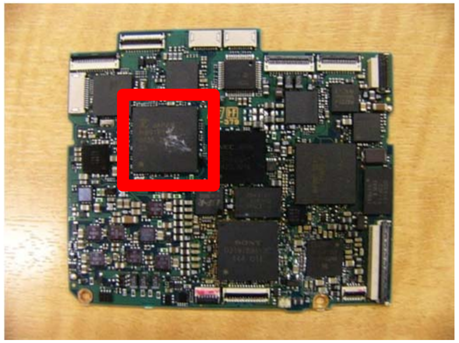
Mini Projector



Blu-ray Recorder



Portable Game Machine

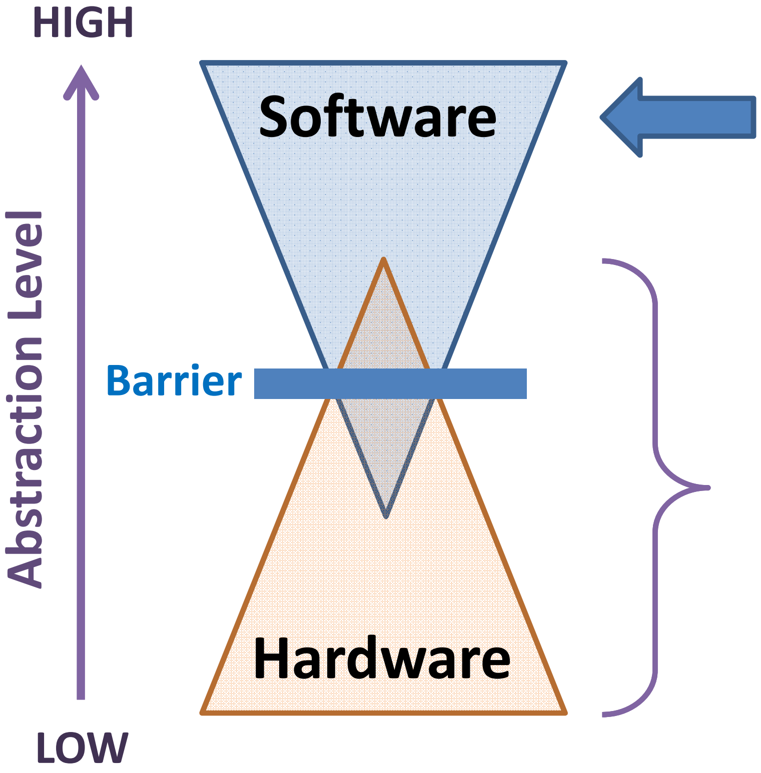


Digital Video Camera

Courtesy of NikkeiBP

1.3 Embedded System

An Embedded System includes the aspect of hardware and software



Embedded System

You are here

GUI applications, Operating System
APIs, Libraries, Drivers

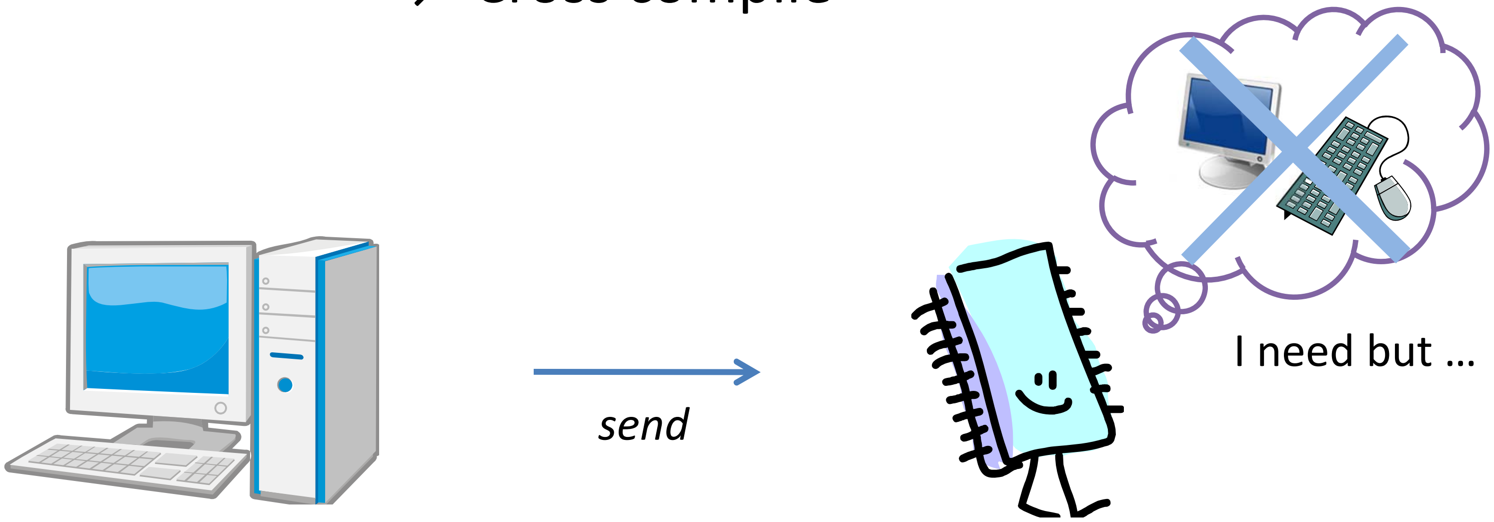
Development tools for microcontroller include;
- **Cross** Compiler, Linker, Assembler,
- Debugger

Development board includes;
- Microcontroller and other ICs
- Programmer, is a tool for writing program

2. How to start

A microcontroller does not have so much resources to develop on itself

→ Cross compile



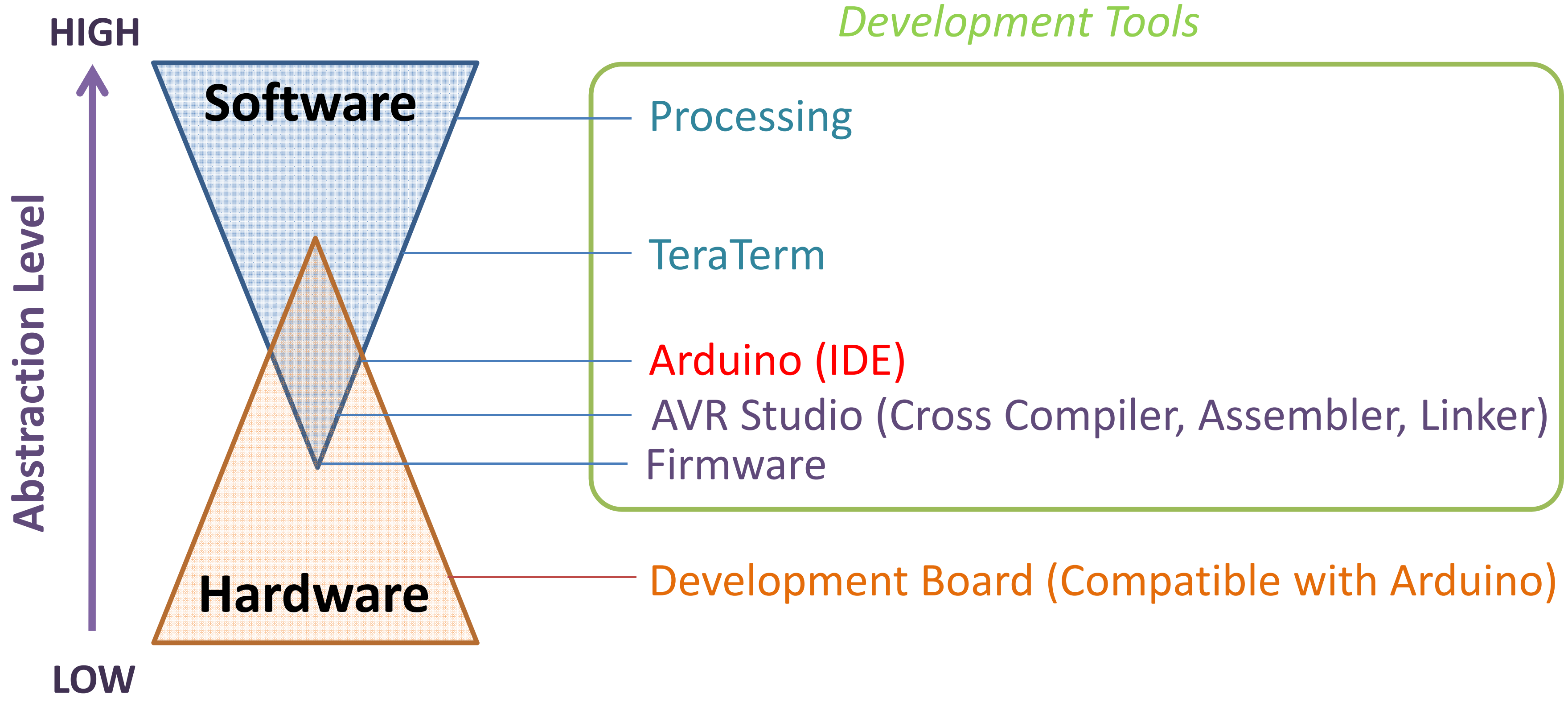
send

I need but …

Build a program

Microcontroller

2.1 Development Tools



Development Tools

- Processing
- TeraTerm
- Arduino (IDE)
- AVR Studio (Cross Compiler, Assembler, Linker)
- Firmware

Development Board (Compatible with Arduino)

2.2 Setup the development tools

Download and install development tools:

AVR Studio 4 and AVR Toolchain is a development environment
- http://www.atmel.com/tools/AVRSTUDIO4.aspx

AVR Writer is to write a program (USBasp) of a microcontroller
- USBasp Writer

Tera Term is to communicate with a microcontroller
- http://sourceforge.jp/projects/ttssh2/releases/

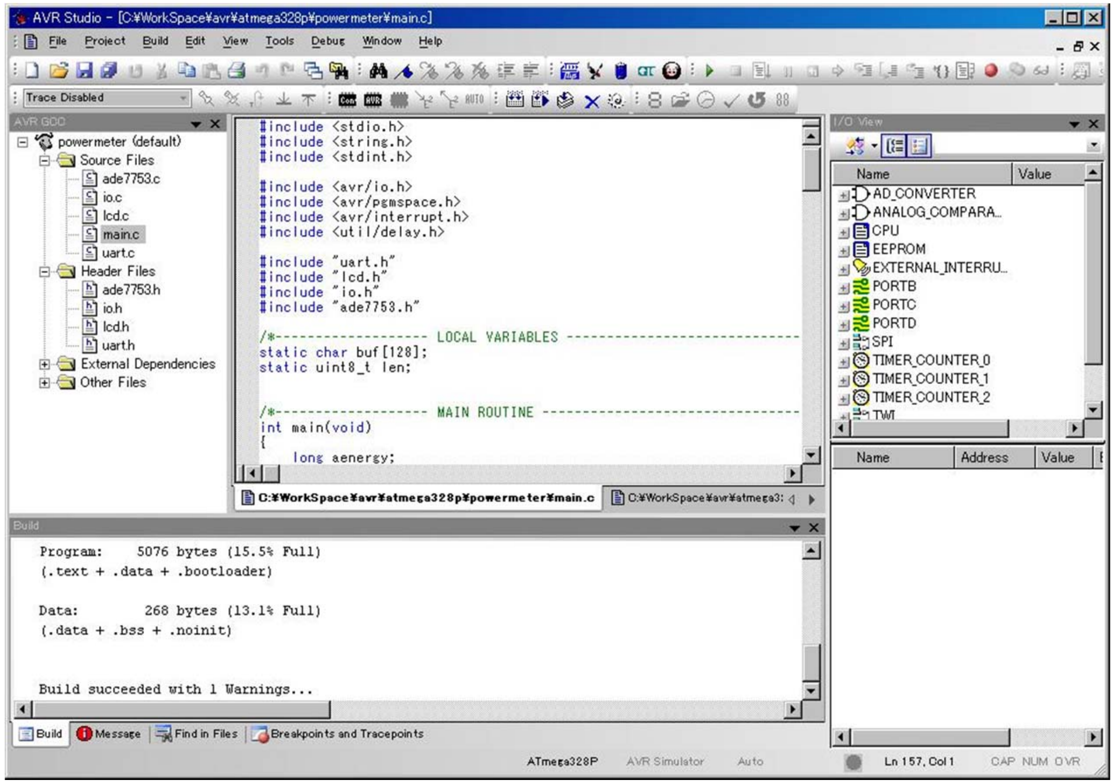
Arduino is software for easy to program on a microcontroller
- http://arduino.cc/hu/Main/Software/

Processing is software for easy to develop a GUI interface
- http://www.processing.org/download/

These are free software without warranty

2.2.1 Install AVR Studio & Toolchain

1. Double click “avr-toolchain-installer-3.3.0.710-win32.x86.exe”
2. Follow the installation wizard
3. Double click on “AVRStudio4Setup.exe”
4. Follow the installation wizard

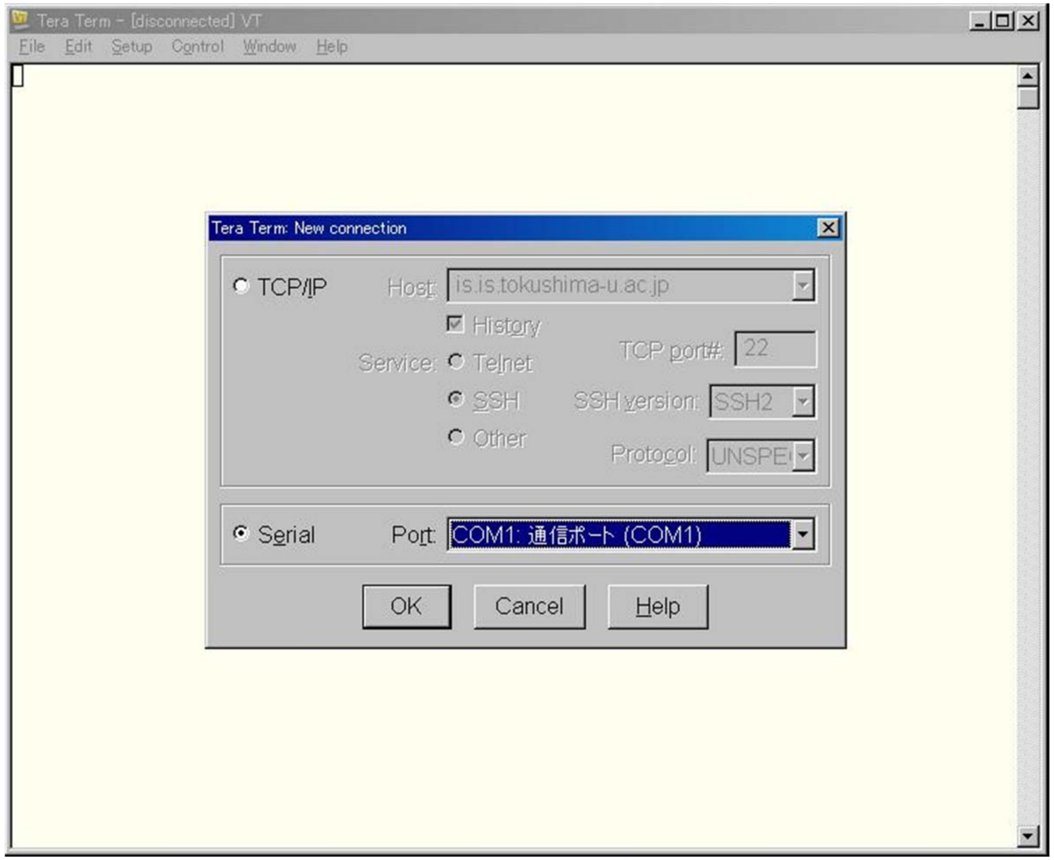


```
#include <stdio.h>
#include <string.h>
#include <stdint.h>

#include <avr/io.h>
#include <avr/pgmspace.h>
#include <avr/interrupt.h>
#include <util/delay.h>

#include "uart.h"
#include "lcd.h"
#include "io.h"
#include "ade7753.h"

/*------------------ LOCAL VARIABLES ---------------------------
static char buf[128];
static uint8_t len;


/*------------------ MAIN ROUTINE ------------------------------
int main(void)
{
    long aenergy;
```

```
Program:    5076 bytes (15.5% Full)
(.text + .data + .bootloader)

Data:        268 bytes (13.1% Full)
(.data + .bss + .noinit)


Build succeeded with 1 Warnings...
```

2.2.2 Install TeraTerm

1. Double Click on “teraterm-4.7.3.exe”
2. Choose language English
3. Follow the installation wizard



○ TCP/IP Host: is.is.tokushima-u.ac.jp

☑ History

Service: ○ Telnet ● SSH ○ Other

TCP port#: 22

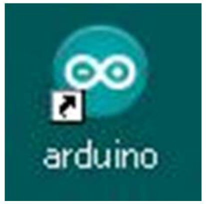
SSH version: SSH2

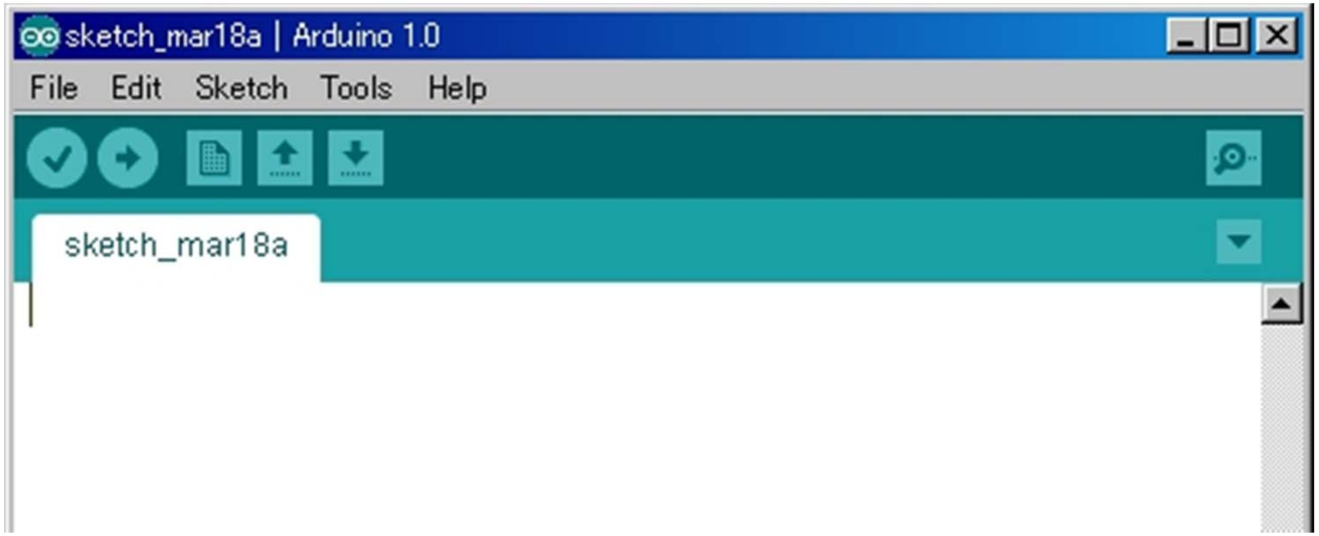
Protocol: UNSPE(

● Serial Port: COM1: 通信ポート (COM1)

2.2.3 Install Arduino

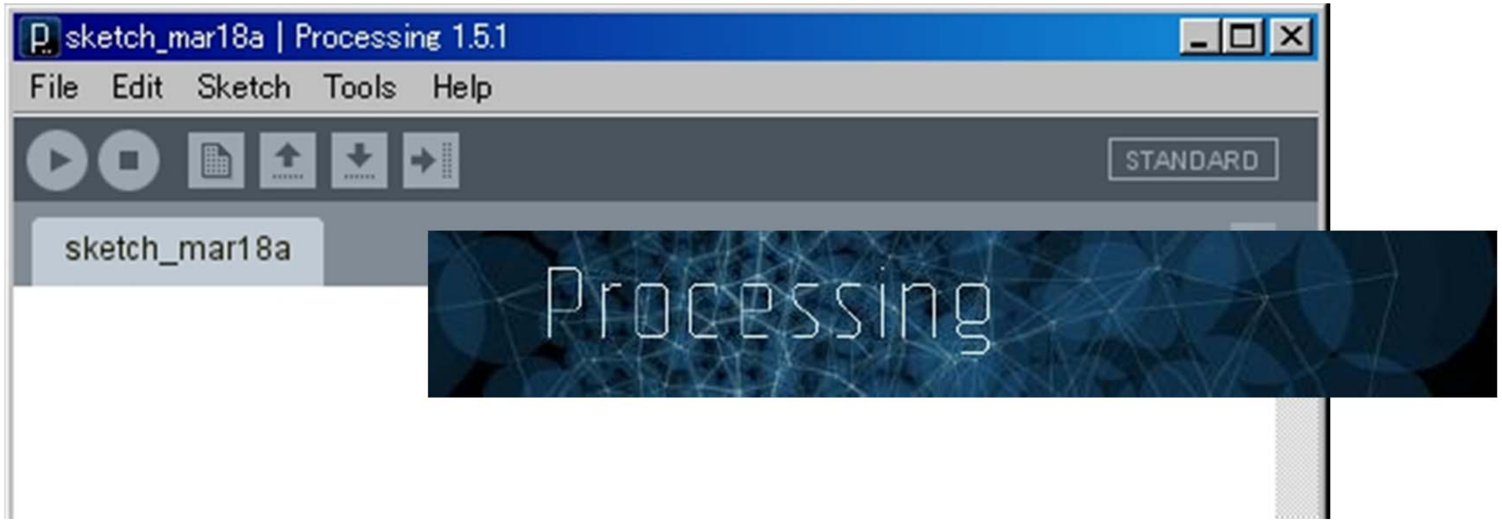
1. Extract the archive “arduino-1.0-windows.zip”
2. Move arduino-1.0 to C:¥
3. Make a short cut of C:¥arduino-1.0¥arduino.exe
   - Right click on arduino.exe
   - Create a shortcut
4. Move the short cut file to the Desktop and rename “arduino”





2.2.4 Install Processing

1. Extract the archive “processing-1.5.1.zip”
2. Move processing-1.5.1 to C:¥
3. Make a short cut of C:¥processing-1.5.1¥processing.exe
   - Right click on processing.exe
   - Create a shortcut
4. Move the short cut file to the Desktop and rename “processing”

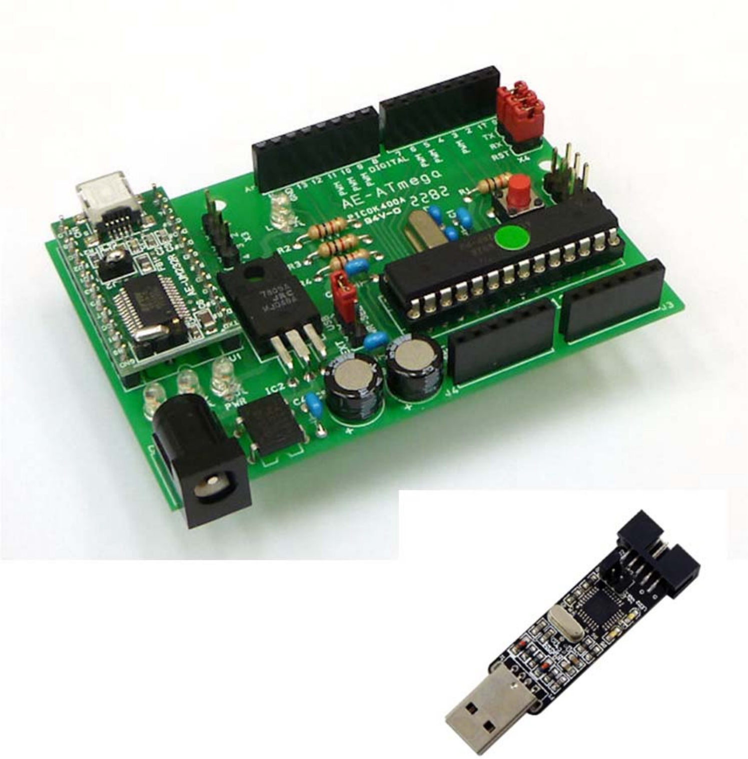


2.3 Setup the development board

Development board:

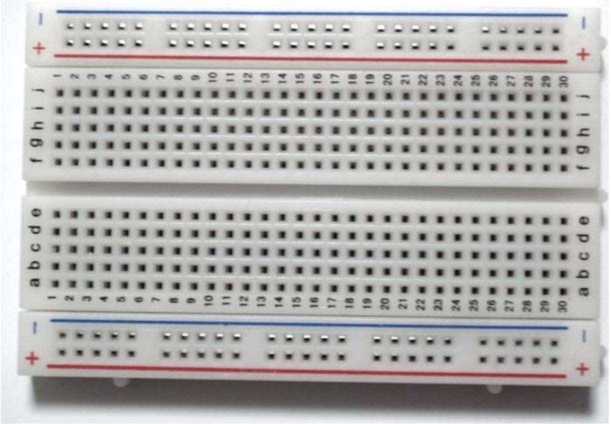
-Compatible with Arduino,

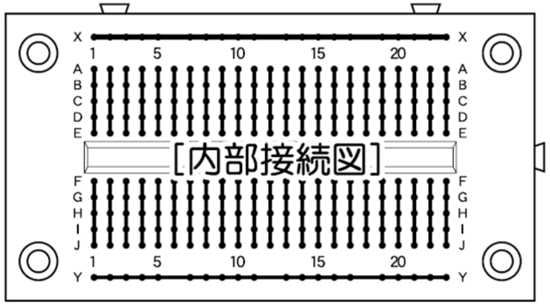
-Programmer



Bread board:

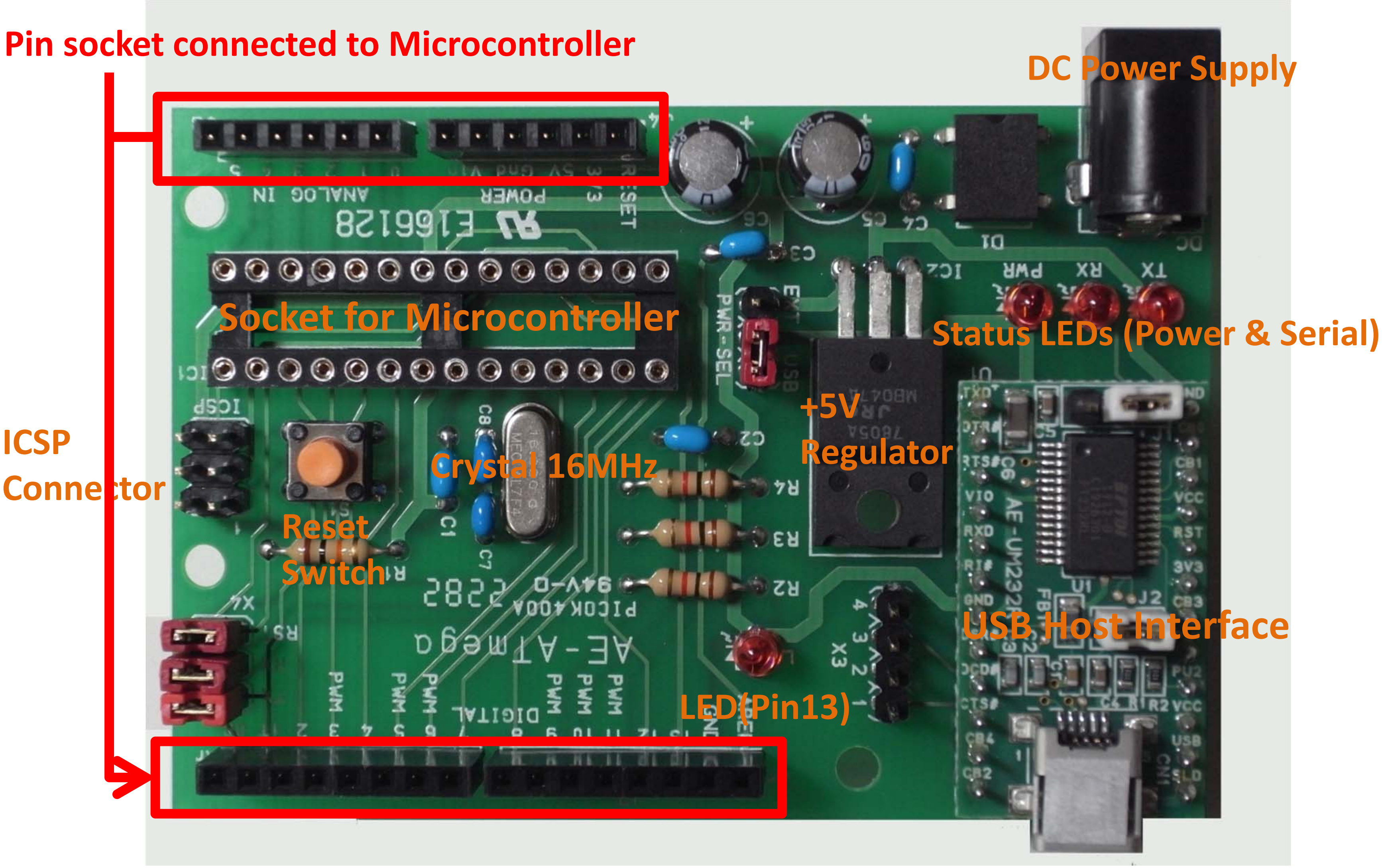
- Rapid prototyping
- Solderless



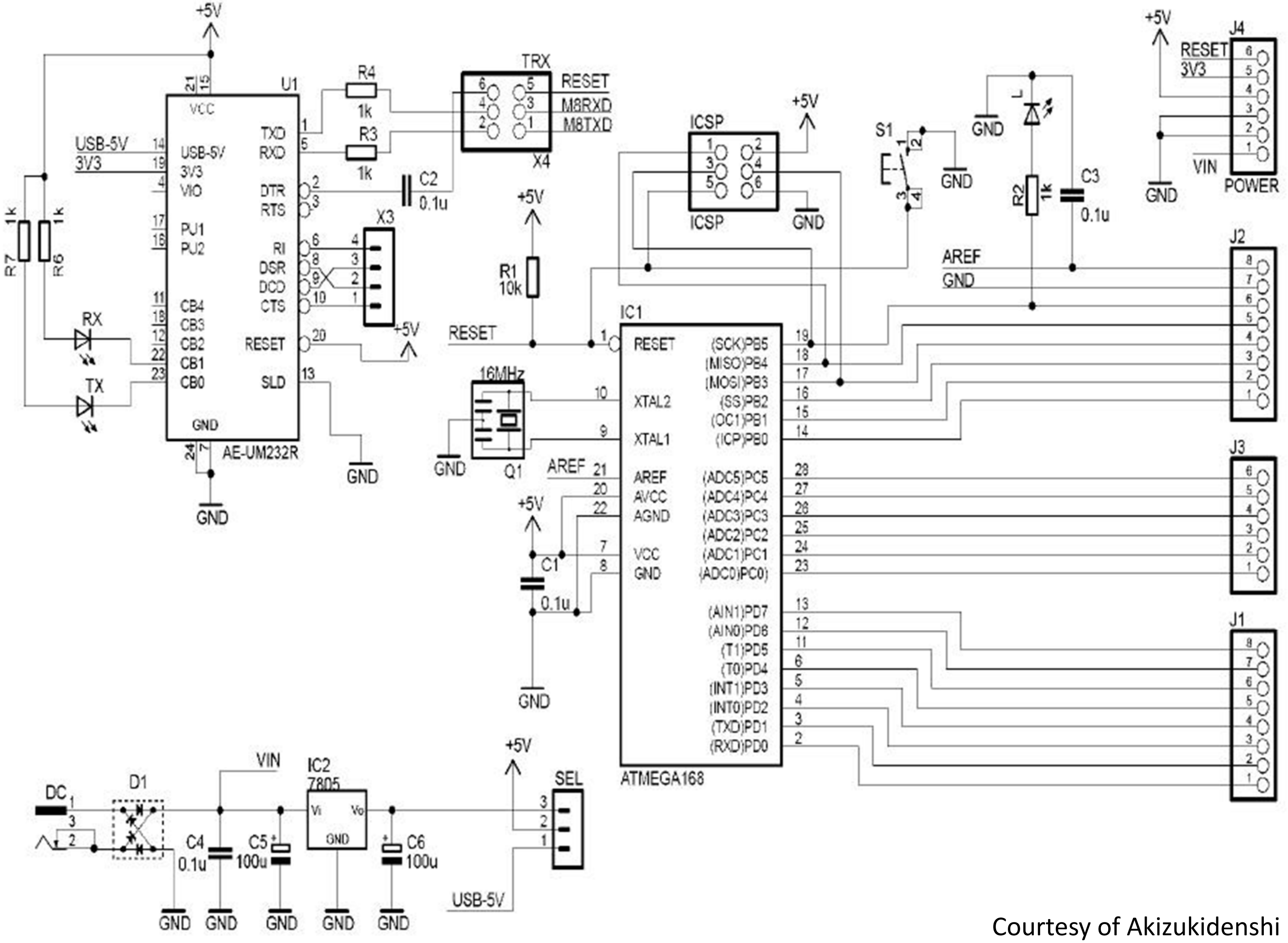


Courtesy of Akizukidenshi

2.3.1 Development Board


Pin socket connected to Microcontroller
DC Power Supply
Socket for Microcontroller
Status LEDs (Power & Serial)
ICSP
Connector
+5V
Regulator
Crystal 16MHz
Reset
Switch
USB Host Interface
LED(Pin13)

2.3.1 Development Board (continued)

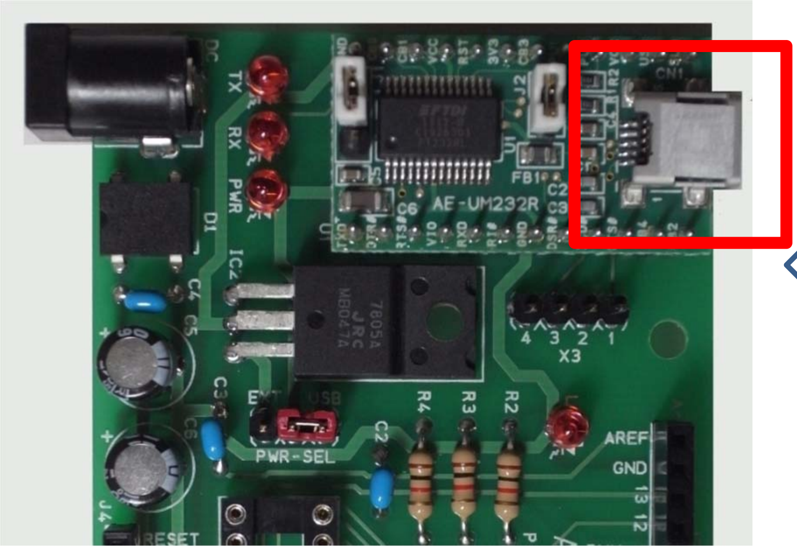


Courtesy of Akizukidenshi

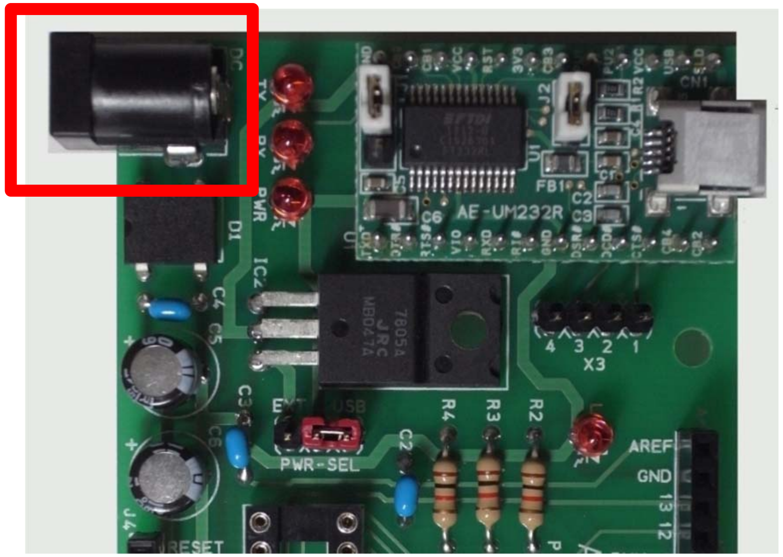
2.3.2 Power Supplies

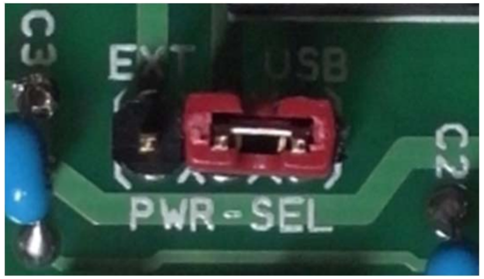
From PC via the USB Port (USB)

MAX: 500mA



From Adapter or Battery (EXT)





Short Pin to change power source, EXT or USB

2.3.3 Arduino Bootloader

Write the Arduino Bootloader (Only first time)

1. Connect a USB cable to the board
2. Connect the USBasp Writer to the ICU connector on the board
3. Check Serial Port Number

   System -> Control Panel -> Device Manager -> Port (COM and LPT)
4. Run Arduino
5. Tools -> Serial Port -> COMx
6. Tools -> Programmer -> USBasp (Set Writer)
7. Tools -> Boards -> Duemilanove w/ Atmega328 (Set Board definition)
8. Tools -> Run Bootloader

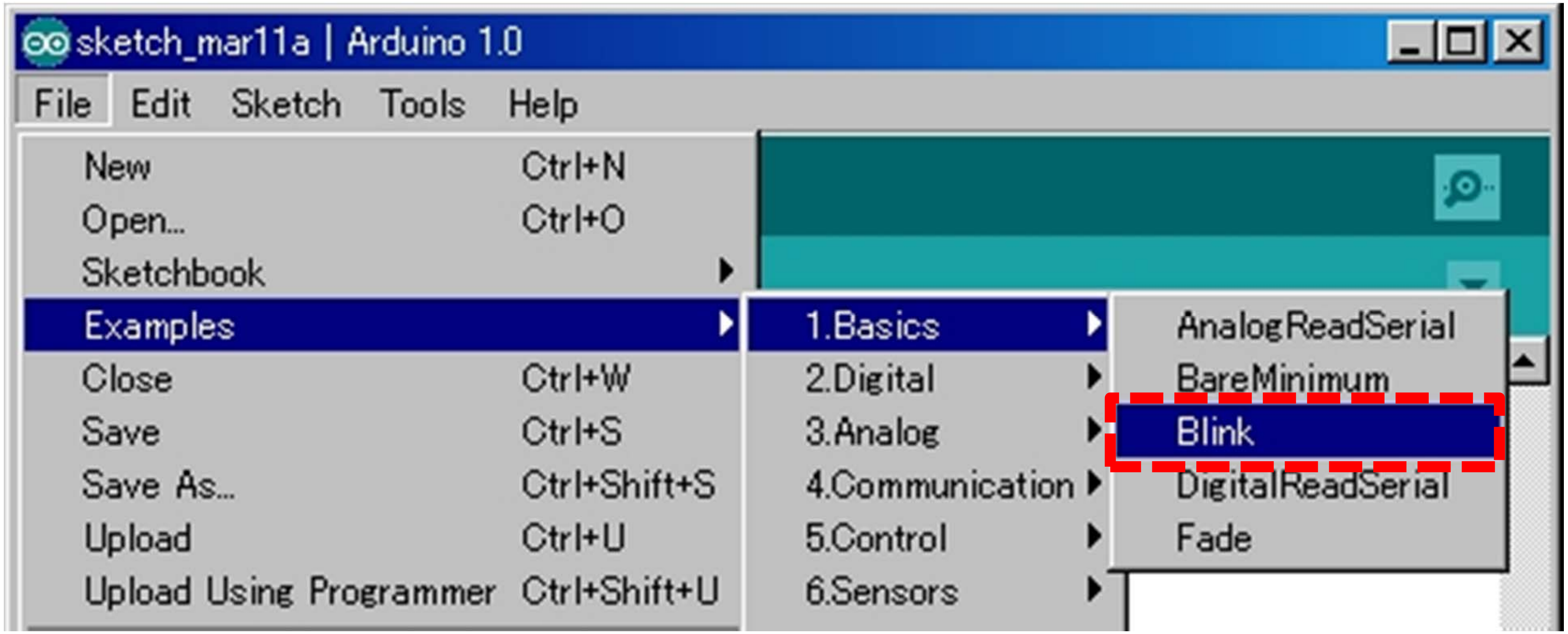
   It takes about one minutes to complete the process

9. Remove the USB cable
10. Remove the USBasp Writer

2.3.4 Board Test (1/5)

Run a test program, Blink a LED on the board

1. Run Arduino
2. File -> Examples -> 1:Basics -> Blink

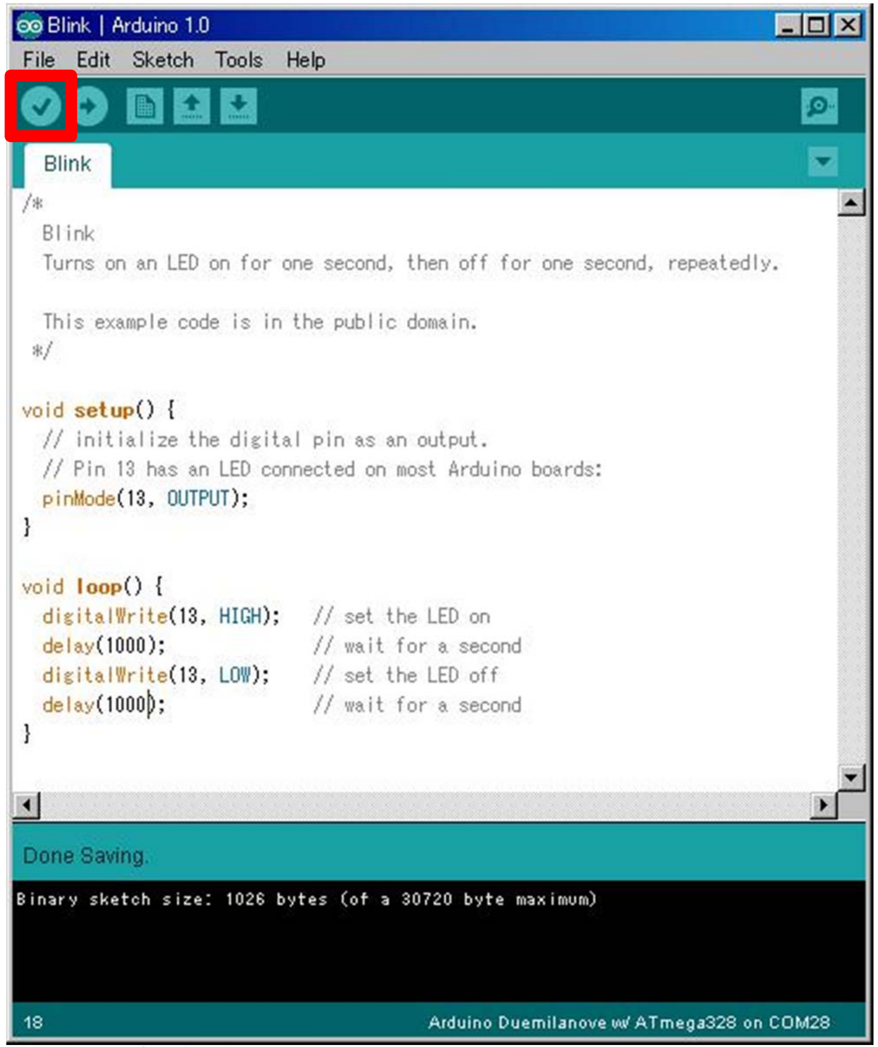


2.3.4 Board Test (2/5)

3. Click on the Verify Icon
to **compile the program**

Source code called **sketch**

Status window



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

  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
}
```

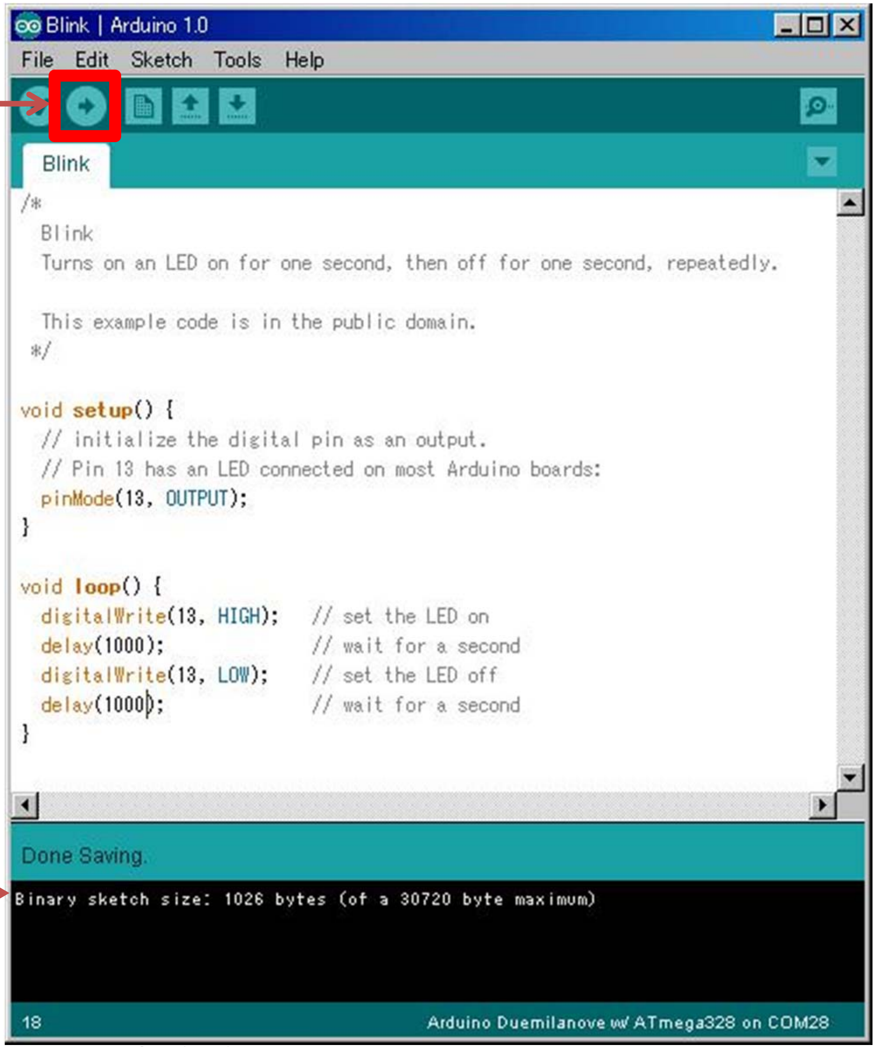
Done Saving.

```
Binary sketch size: 1026 bytes (of a 30720 byte maximum)
```

18 Arduino Duemilanove w/ ATmega328 on COM28

2.3.4 Board Test (3/5)

4. Click on the Upload Icon
   to **load the program to the microcontroller**



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

  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
}
```

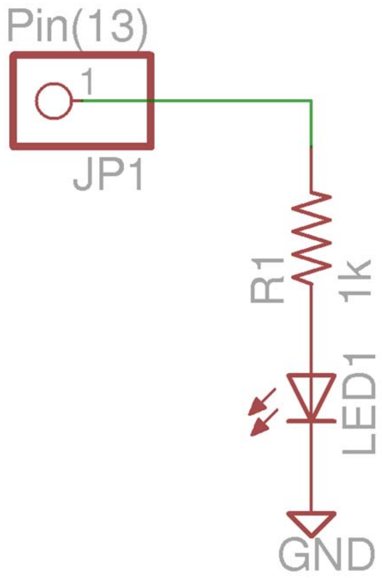
Done Saving.

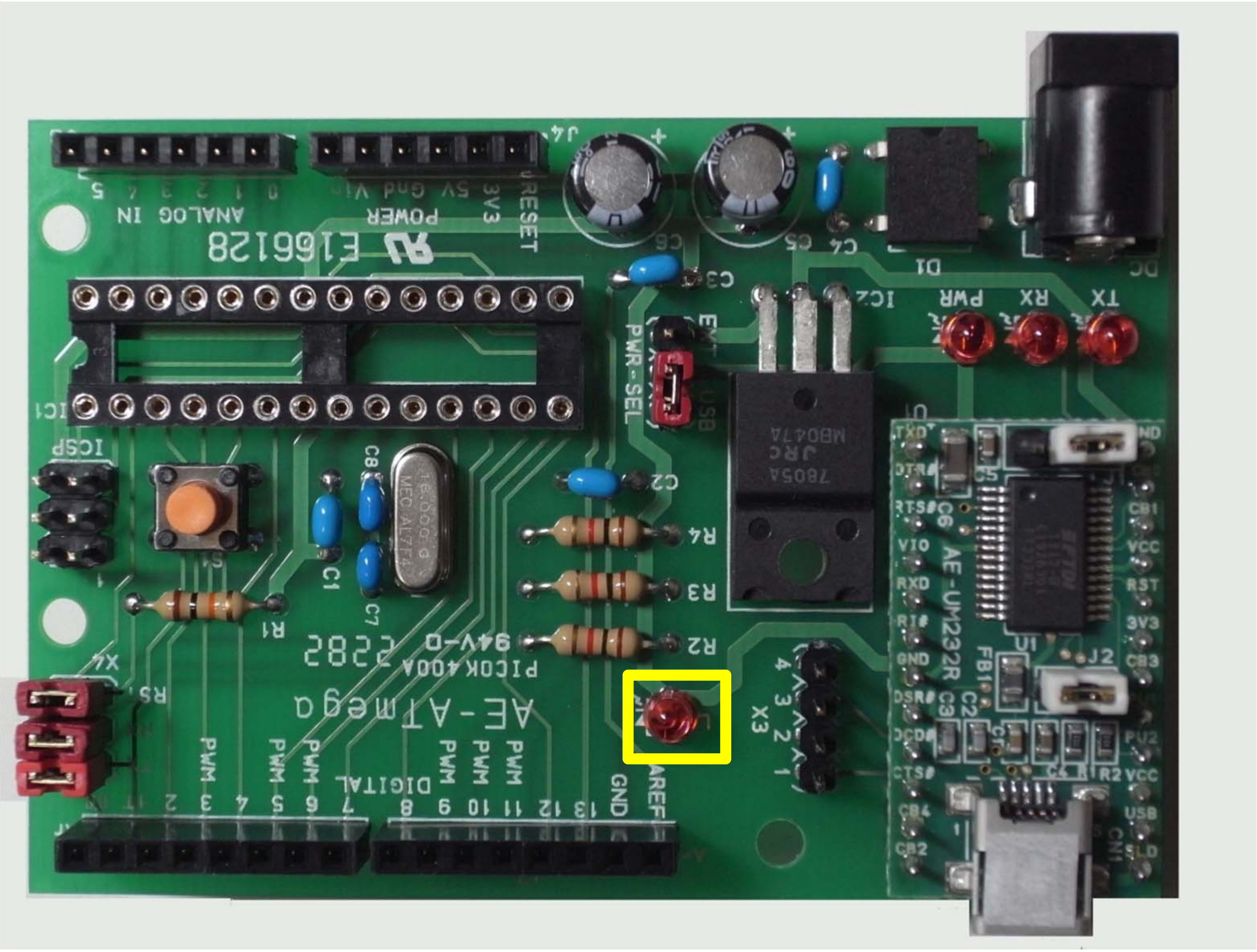
```
Binary sketch size: 1026 bytes (of a 30720 byte maximum)
```

5. Check the status window

2.3.4 Board Test （4/5)

LED connected to
Pin(13) is Blinking

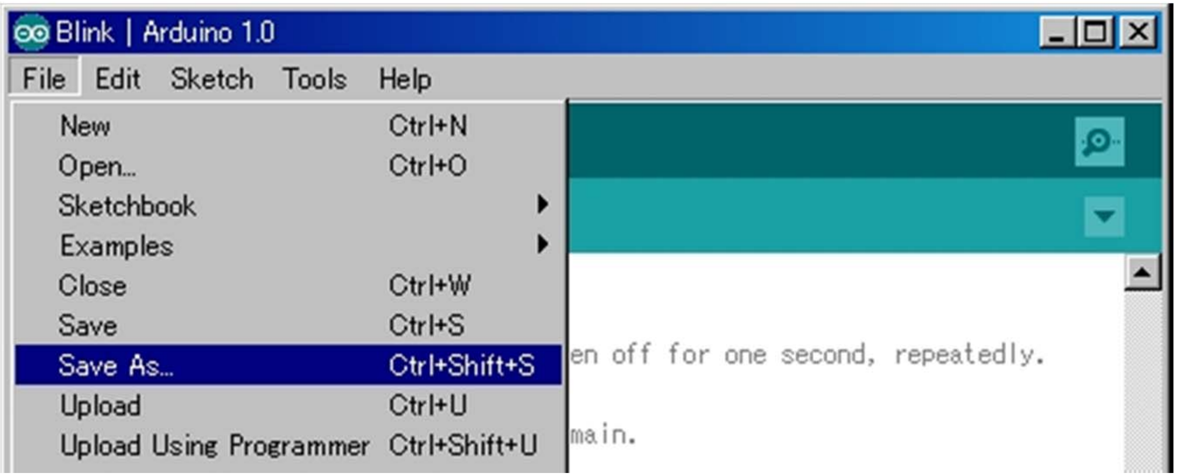




2.3.4 Board Test (5/5)

Save Project

1. File -> Save As
2. Save as Project name:
   Blink (*1)

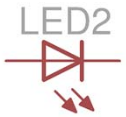


```
en off for one second, repeatedly.

main.
```

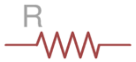
(*1) Project is saved to under the MyDocuments¥Arduino¥Blink folder
Source code is saved as Blink.ino in the Blink folder
If you want to delete the project, just remove the Blink folder

2.4 Parts

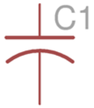
LED



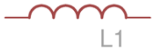
Resistor



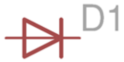
Ceramic capacitor



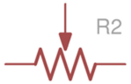
Inductor

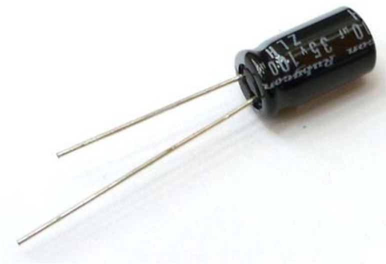


Diode

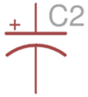


Potentiometer

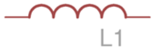




Aluminum capacitor

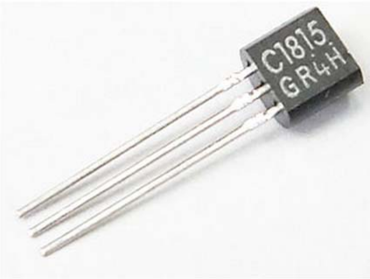


Coil

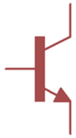


Courtesy of Akizukidenshi

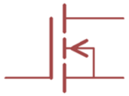
2.4 Parts (continued)

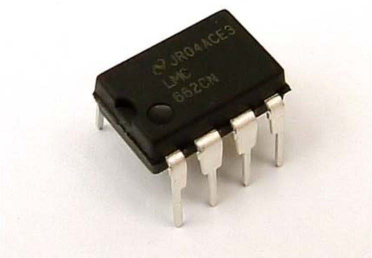


Transistor
NPN

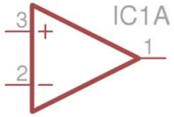


P channel
MOS-FET

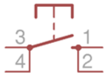


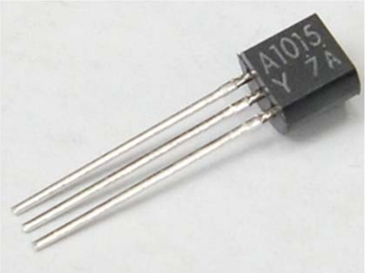


Operational
Amplifier

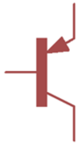


Push Switch

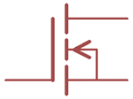




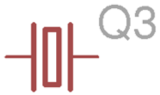
Transistor
PNP

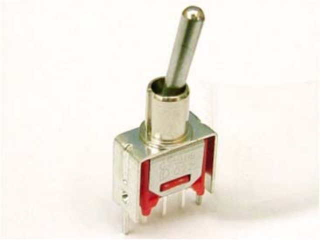


N Channe
MOS-FET

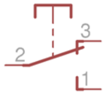


Crystal





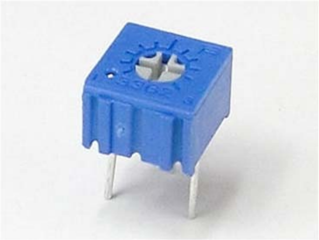
Switch



Courtesy of Akizukidenshi

2.4.1 Label of Parts

Resistor



Potentiometer

	1st	2nd	3rd Band
Black	0	0	x 1
Brown	1	1	x 10
Red	2	2	x 100
Orange	3	3	x 1000 (k)
Yellow	4	4	x 10000
Green	5	5	x 100000
Blue	6	6	x 1000000 (M)
Violet	7	7	…
Gray	8	8	
White	9	9	
Silver/Gold			10%/5% (Tolerance)

Example) Resistor

Brown Black Red

1 0 x 100 = 1000 Ω = 1 kΩ

1st 2nd 3rd

Example) Potentiometer

1 0 3 = 10 x 10^3 = 10kΩ

1st 2nd 3rd

Ceramic capacitor

105 = 1.0 uF

104 = 0.1 uF

103 = 10000 pF

102 = 1000 pF

101 = 100 pF

10 = 10 pF

Example) Capacitor

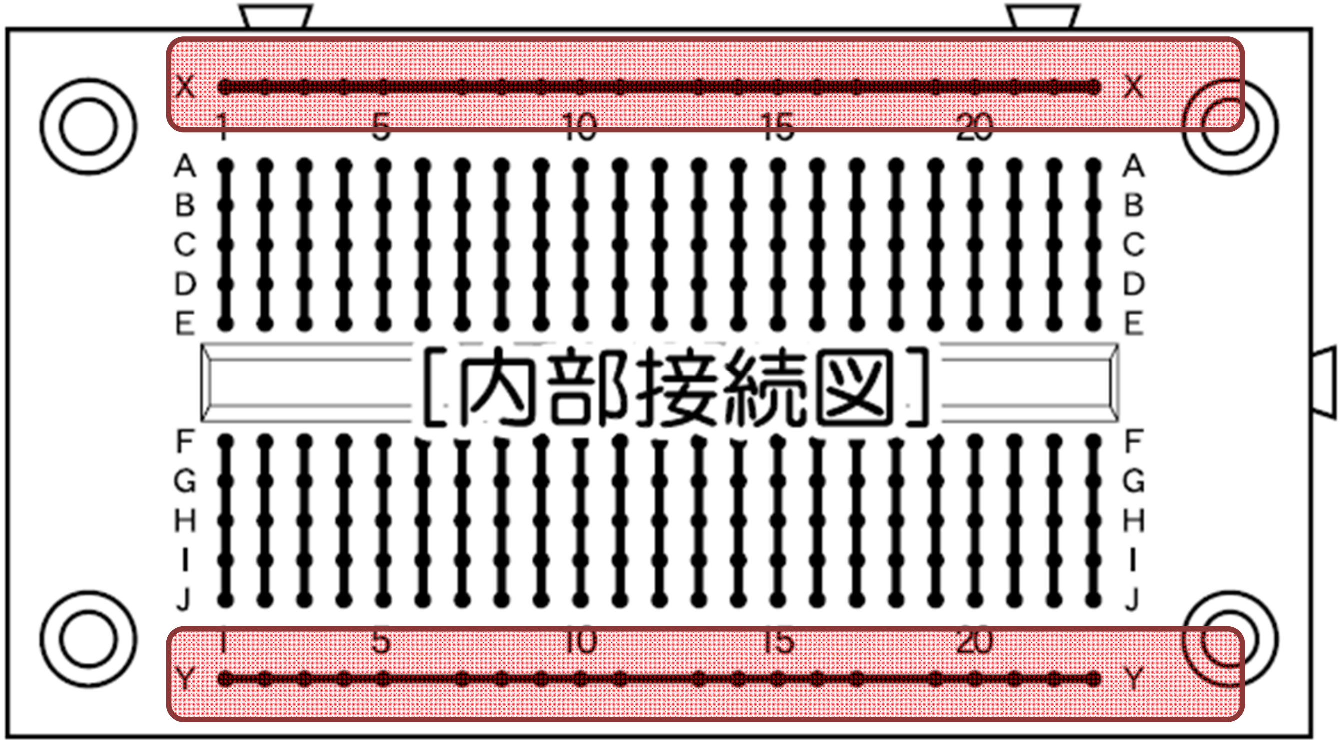
2 2 4

2 2 x 10^4 = 220000 pF = 2.2 μF

Courtesy of Akizukidenshi

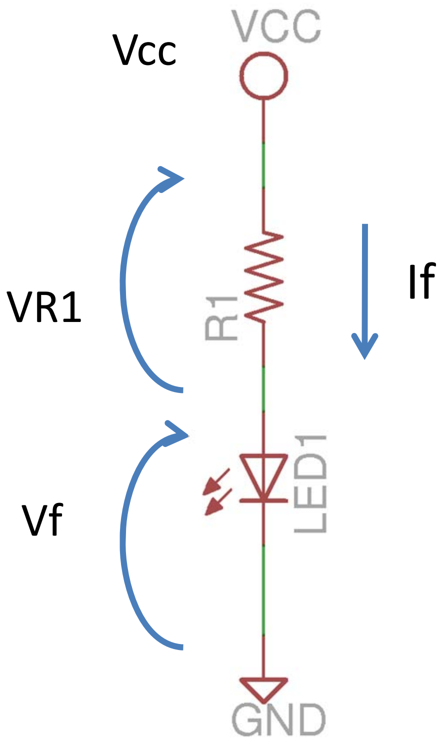
2. 5 Using the Breadboard

For Power Supply (Vcc or GND)



2.5.1 Schematic

Ex) LED circuit



Schematic

$$\begin{cases} Vcc = VR1 + Vf \\ VR1 = R1 \times If \end{cases}$$

Vcc = 5V, If = 10 mA, Vf=2.1V

R1 = ?

Ohm's Law

V = R x I

V : Voltage [V]

R : Resistor [Ω]

I : Current [A]

Typical Vf voltage:

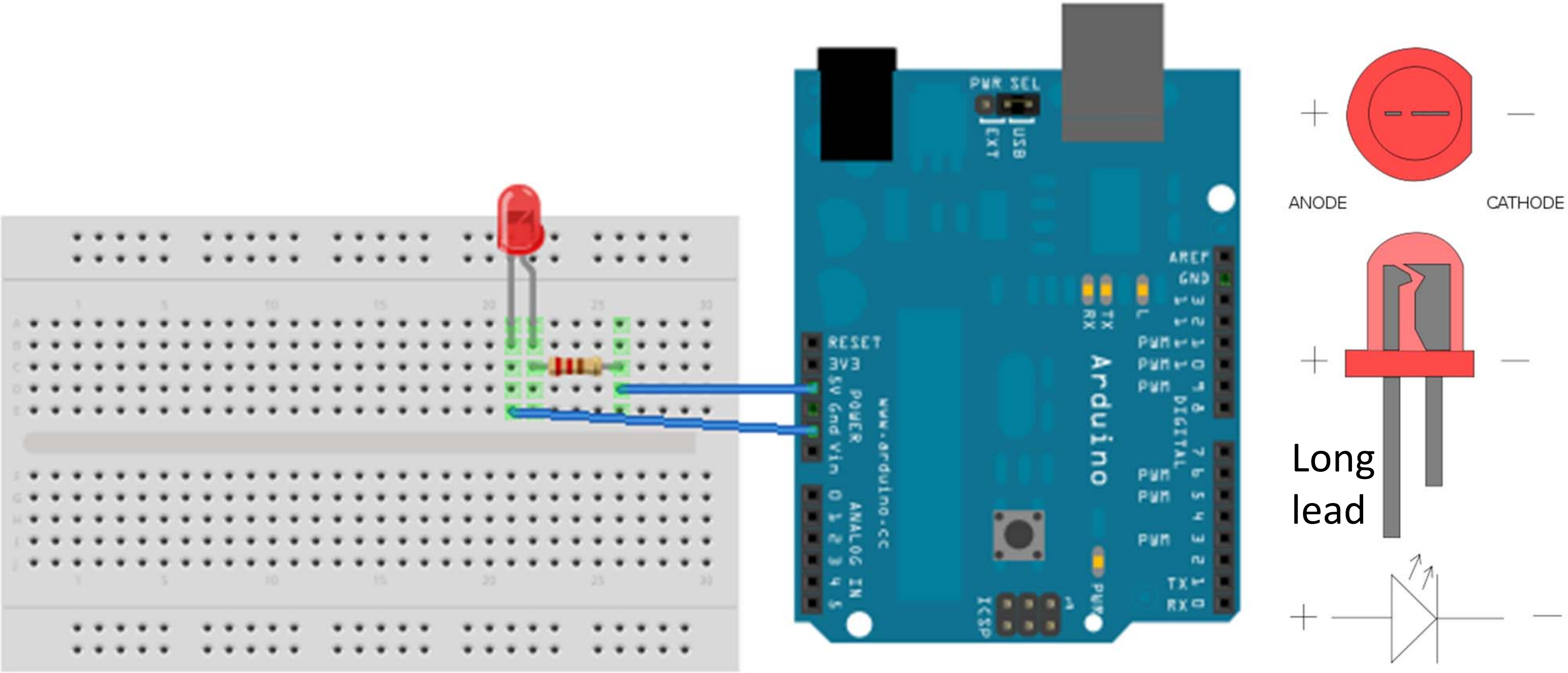
Red, Green, Yellow LED

1.7V – 2.1V

Blue, White LED

3V – 3.4V

2.5.2 LED power supply from Arduino



Programming the Microcontroller 1

Day 2

Estimate: 2 hours

2012/3/22(Wed) 10:00—12:00

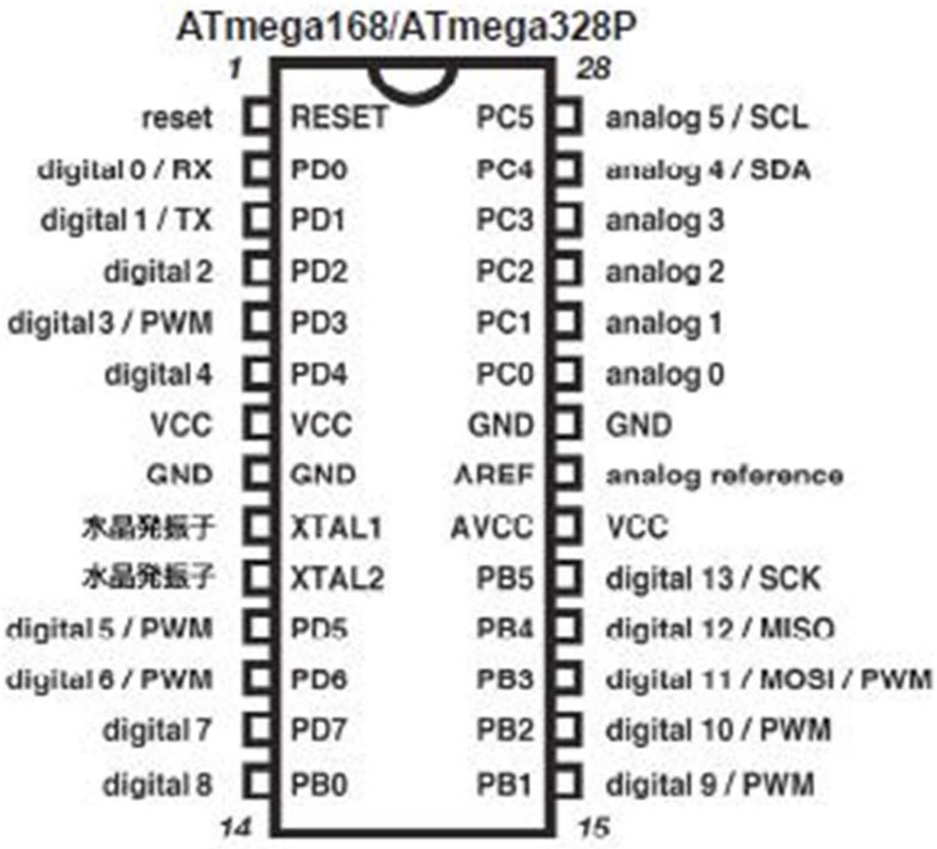
Agenda

1. How it works
   - Specification, Arduino, Pin Assignment, Work Flow

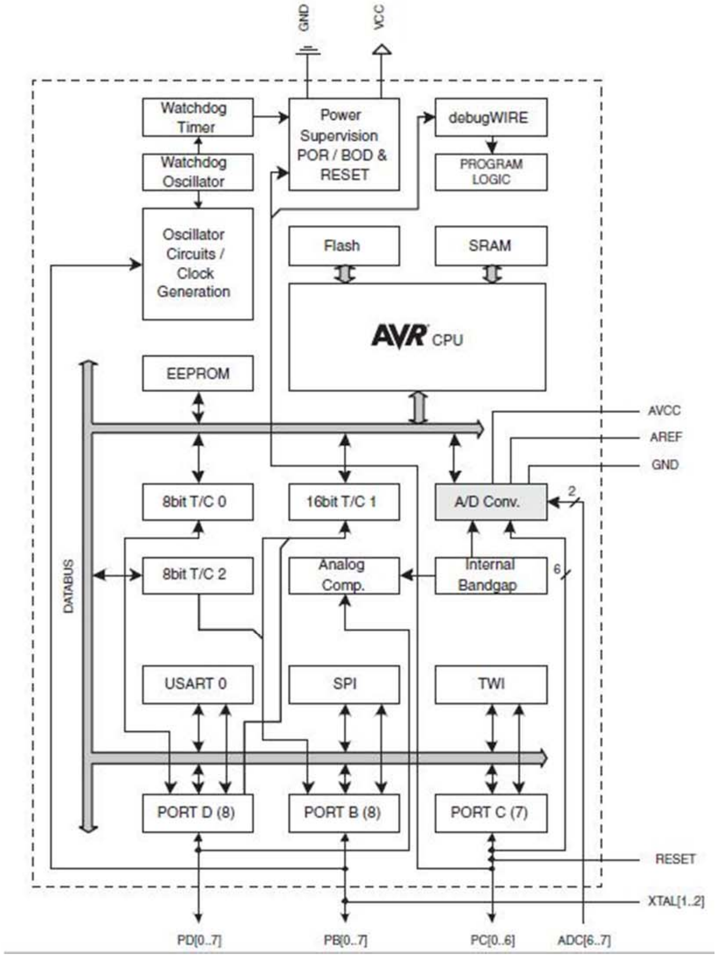
2. Embedded System Programming
   - Polling, Polling and Interrupt, Interrupt
   - Peripherals
   - Data Types

3. I/O Port

1 How it works



Pin assignment

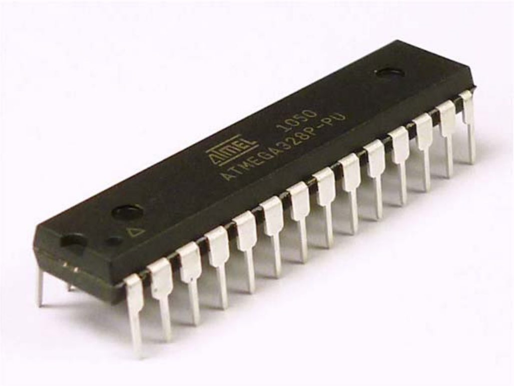


Block diagram

Courtesy of Atmel Corp.

1.1 Specification

Atmel, AVR Atmega 328P



ROM: 32kB

RAM: 2kB

EEPROM: 1kB

Frequency: 16 MHz

Power supply: 5V (or 3.3V)

28 pin PDIP package

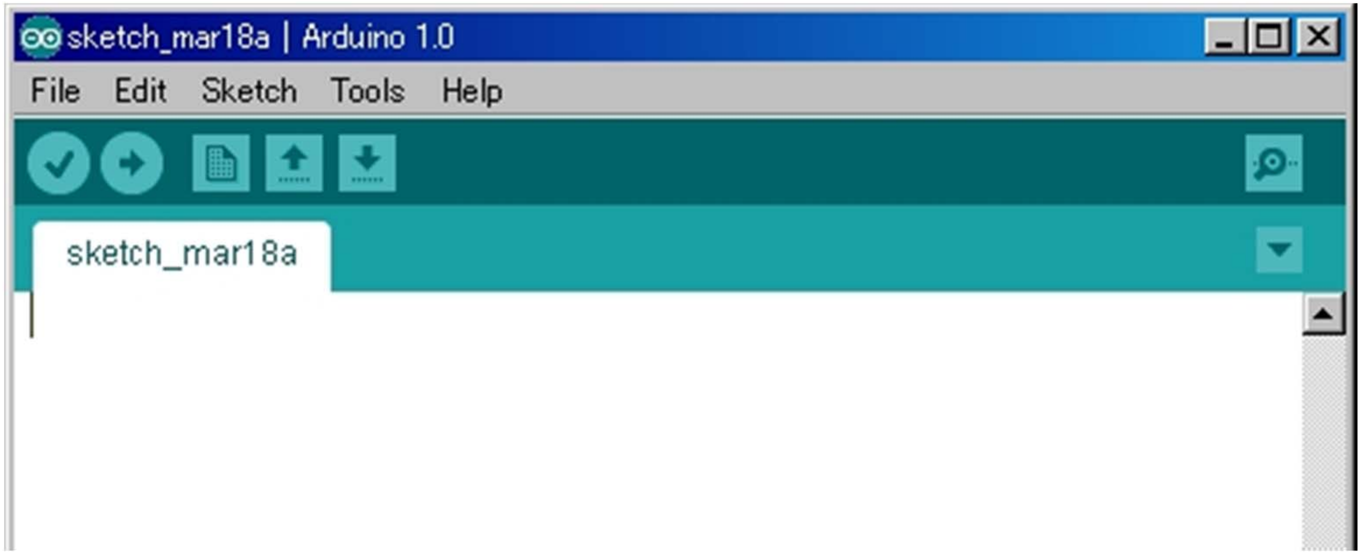
Function:

Digital I/O x9, 8-Bit Timer x2, 16-Bit Timer x1,
PWM Channel x6, 10-Bit ADC x5,
Serial UART x1, SPI x1

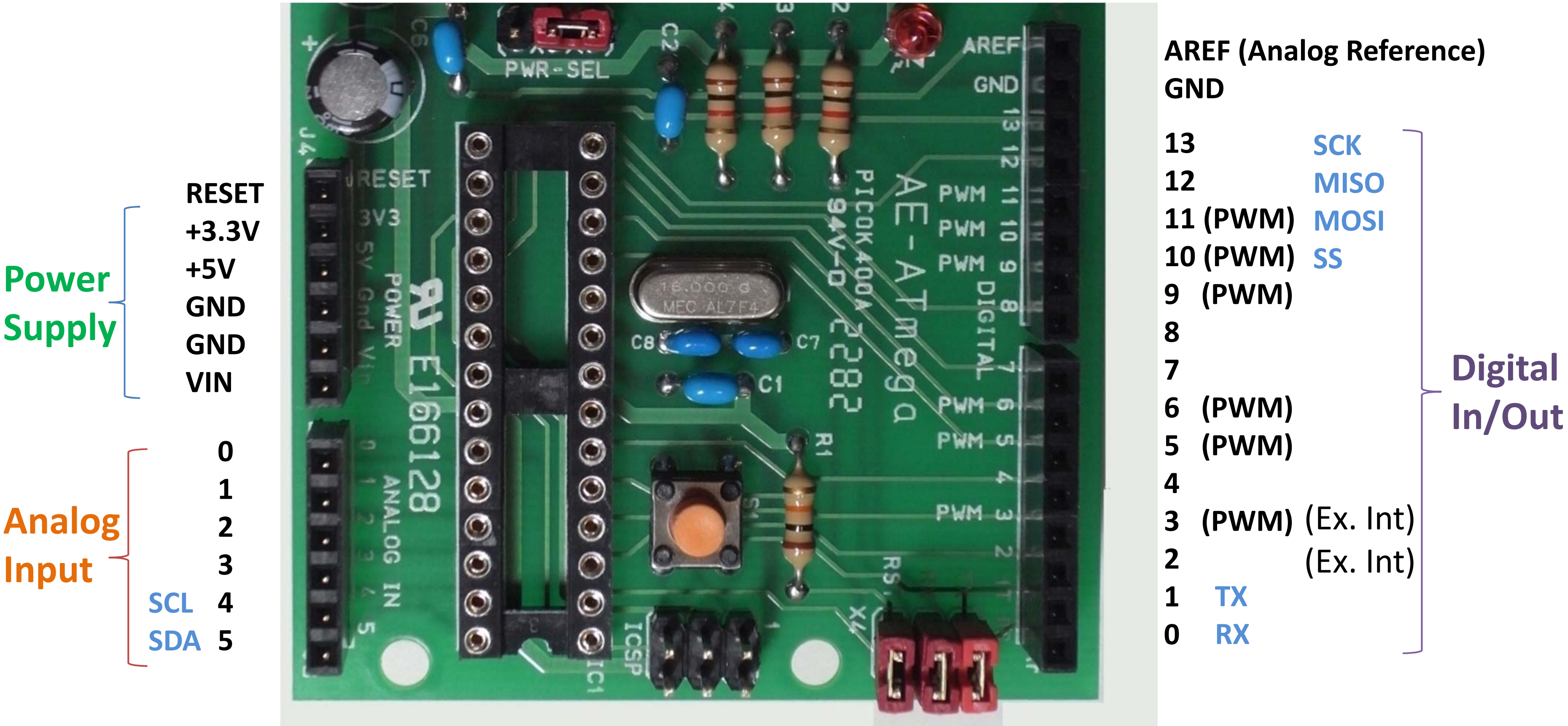
1.2 Arduino

Arduino development environment

- has started developing in Italy
- easy to use for beginners, no need software or electronics experience
- C / C++ language
- IDE (Integrated Development Environment)
- APIs



1.3 Pin Assignment for Arduino



Power Supply

RESET
+3.3V
+5V
GND
GND
VIN

Analog Input

0
1
2
3
SCL 4
SDA 5

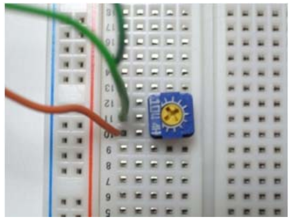
AREF (Analog Reference)
GND

Digital In/Out

13 SCK
12 MISO
11 (PWM) MOSI
10 (PWM) SS
9 (PWM)
8
7
6 (PWM)
5 (PWM)
4
3 (PWM) (Ex. Int)
2 (Ex. Int)
1 TX
0 RX

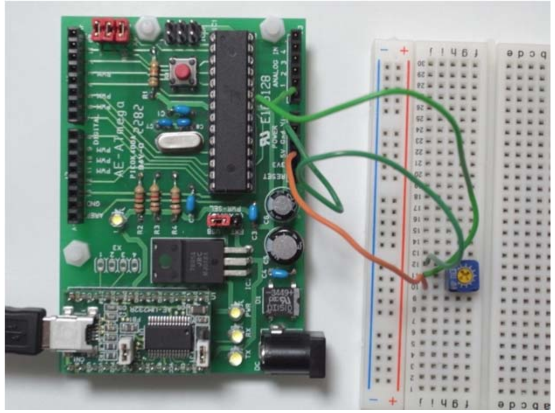
1.4 Work Flow

1. Make a circuit on the bread board



Note: DO NOT SHORT +5V and GND

2. Connect a circuit to the development board by wire

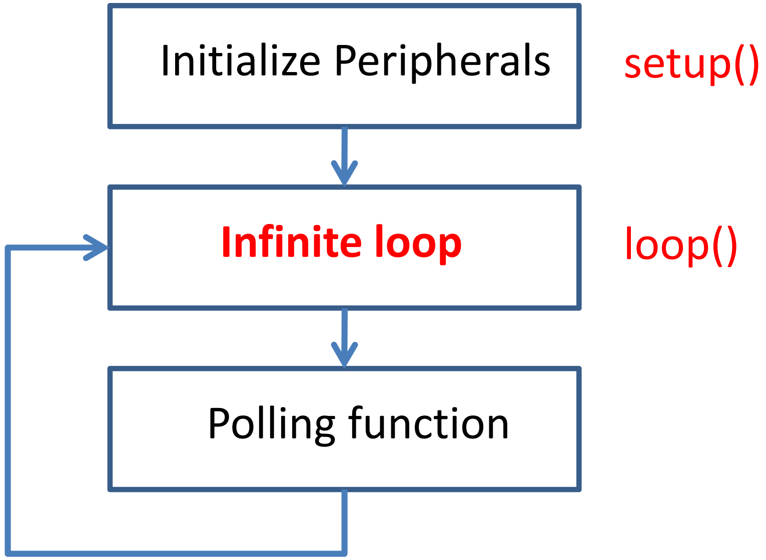


Note: DO NOT SUPPLY POWER

3. Programming on the PC
4. Connect a USB cable to the PC (Power Supply)
5 Upload a program to the microcontroller

2 Embedded System Programming

Basic structure



Polling

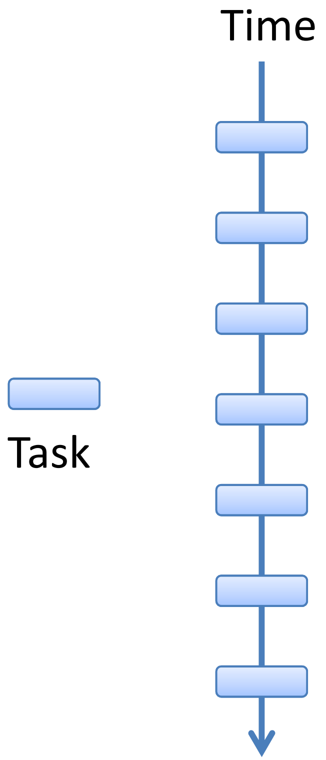
Interrupt function

Interrupt requested from internal or external hardware

MCU has to do an interrupt function
Immediately even if a polling function
Is not finished.

Interrupt

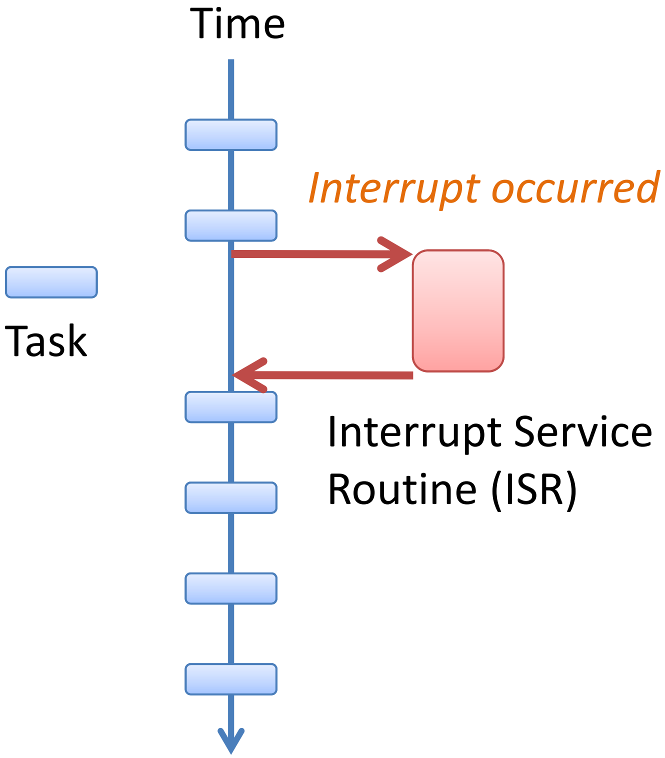
2.1 Polling



Polling
without Interrupt

```
int main()
{
   init();        //  initializes a hardware

   setup();     //  setup your sketch's function

   while (1) {  //  do the task forever
     loop();     //  Task, polling function
   }
    // no terminate, no return
}
```

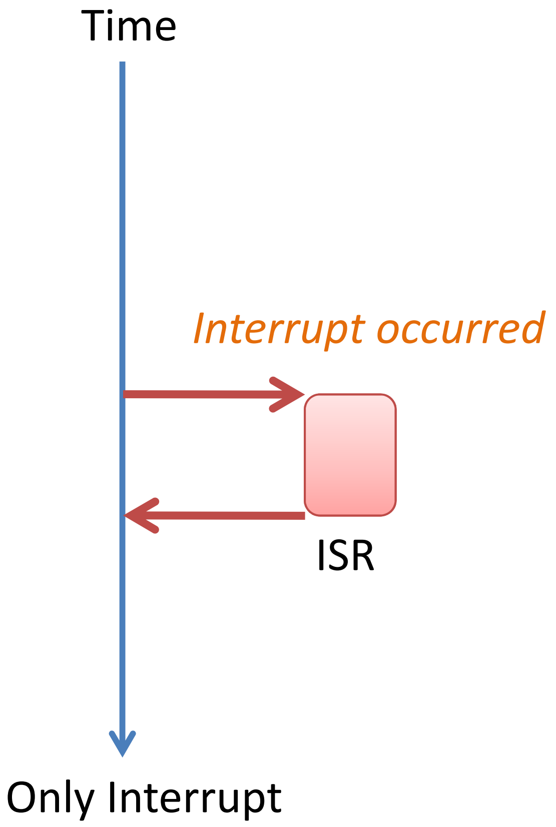
2.2 Polling and Interrupt



Polling with Interrupt

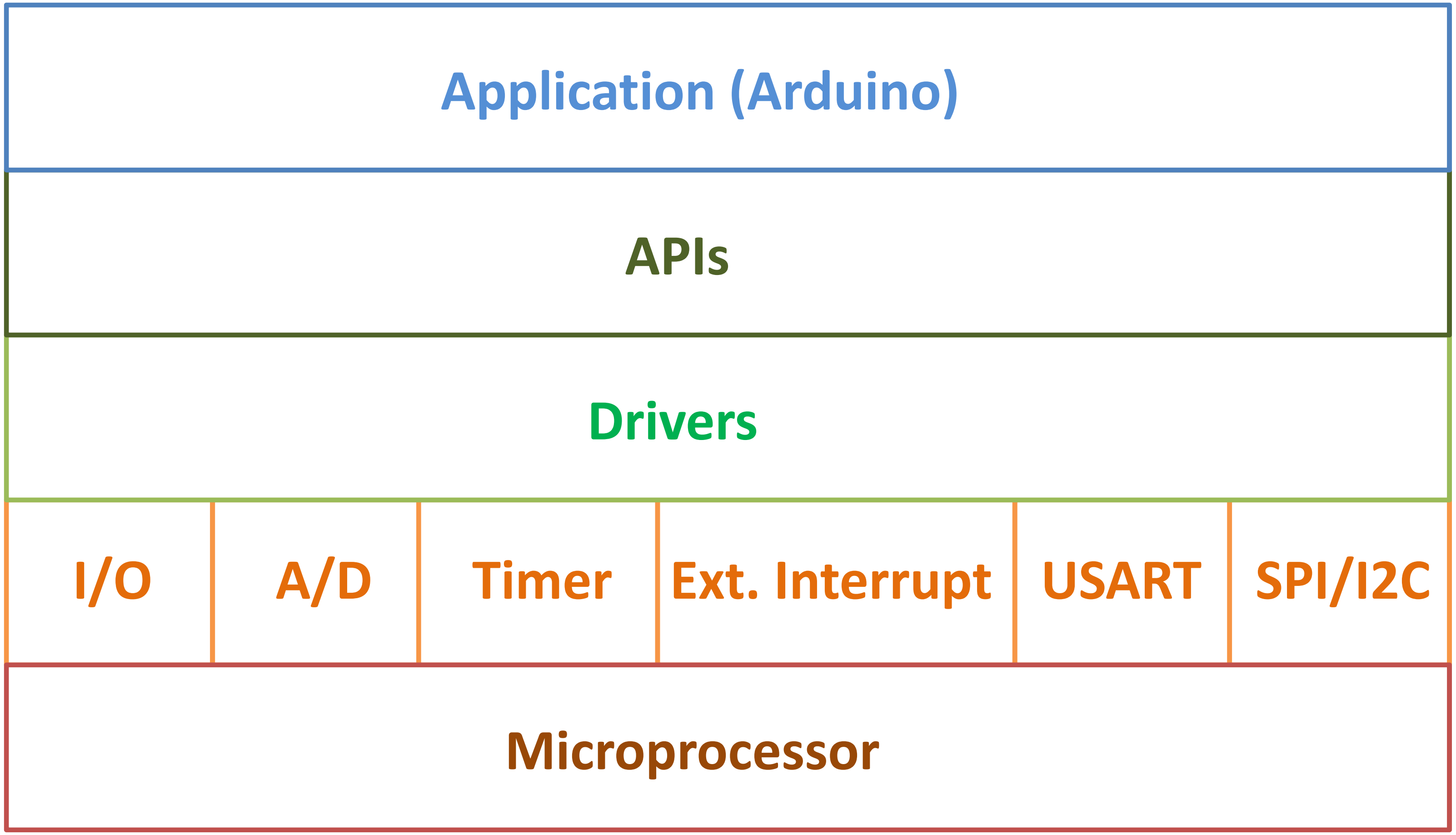
```
int main()
{
    init();         //  initializes a hardware

    setup();      //  setup your sketch’s function

    while (1) {  //  do the task forever
      loop();     //  Task, polling function
    }
     // no terminate, no return
}

void isr() {
    // interrupt service routine
}
```

2.3 Interrupt



```
int main()
{
   init();        //  initializes a hardware

   setup();     //  setup an interrupt routine

   while (1);  // sleep, no task
    // no terminate, no return
}

void isr() {  // awake by an interrupt request
   // interrupt service routine
}
```

2.4 Peripherals



Application (Arduino)

APIs

Drivers

I/O

A/D

Timer

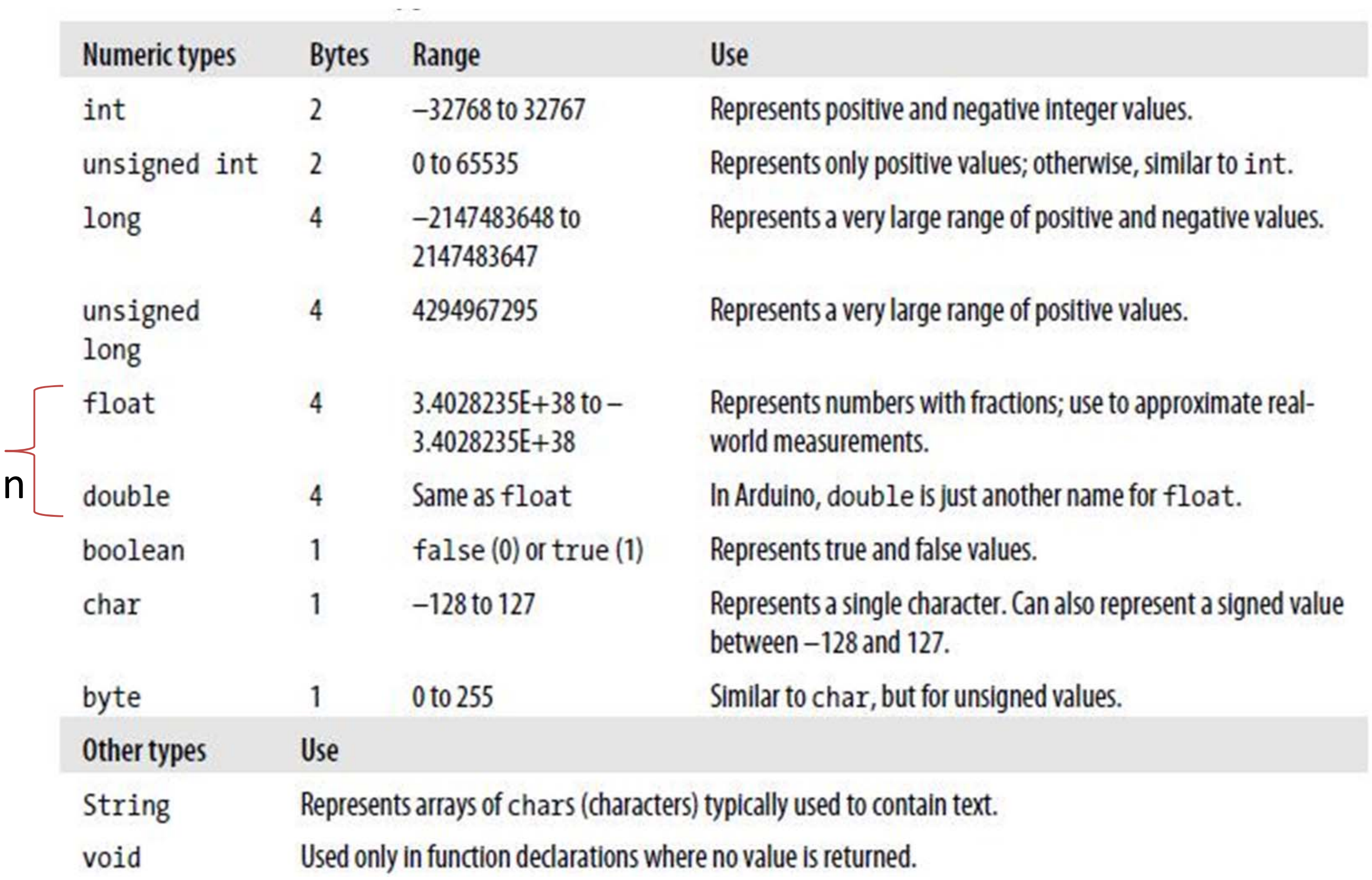
Ext. Interrupt

USART

SPI/I2C

Microprocessor

2.5 Data tyeps



Numeric types	Bytes	Range	Use
int	2	−32768 to 32767	Represents positive and negative integer values.
unsigned int	2	0 to 65535	Represents only positive values; otherwise, similar to int.
long	4	−2147483648 to 2147483647	Represents a very large range of positive and negative values.
unsigned long	4	4294967295	Represents a very large range of positive values.
float	4	3.4028235E+38 to −3.4028235E+38	Represents numbers with fractions; use to approximate real-world measurements.
double	4	Same as float	In Arduino, double is just another name for float.
boolean	1	false (0) or true (1)	Represents true and false values.
char	1	−128 to 127	Represents a single character. Can also represent a signed value between −128 and 127.
byte	1	0 to 255	Similar to char, but for unsigned values.

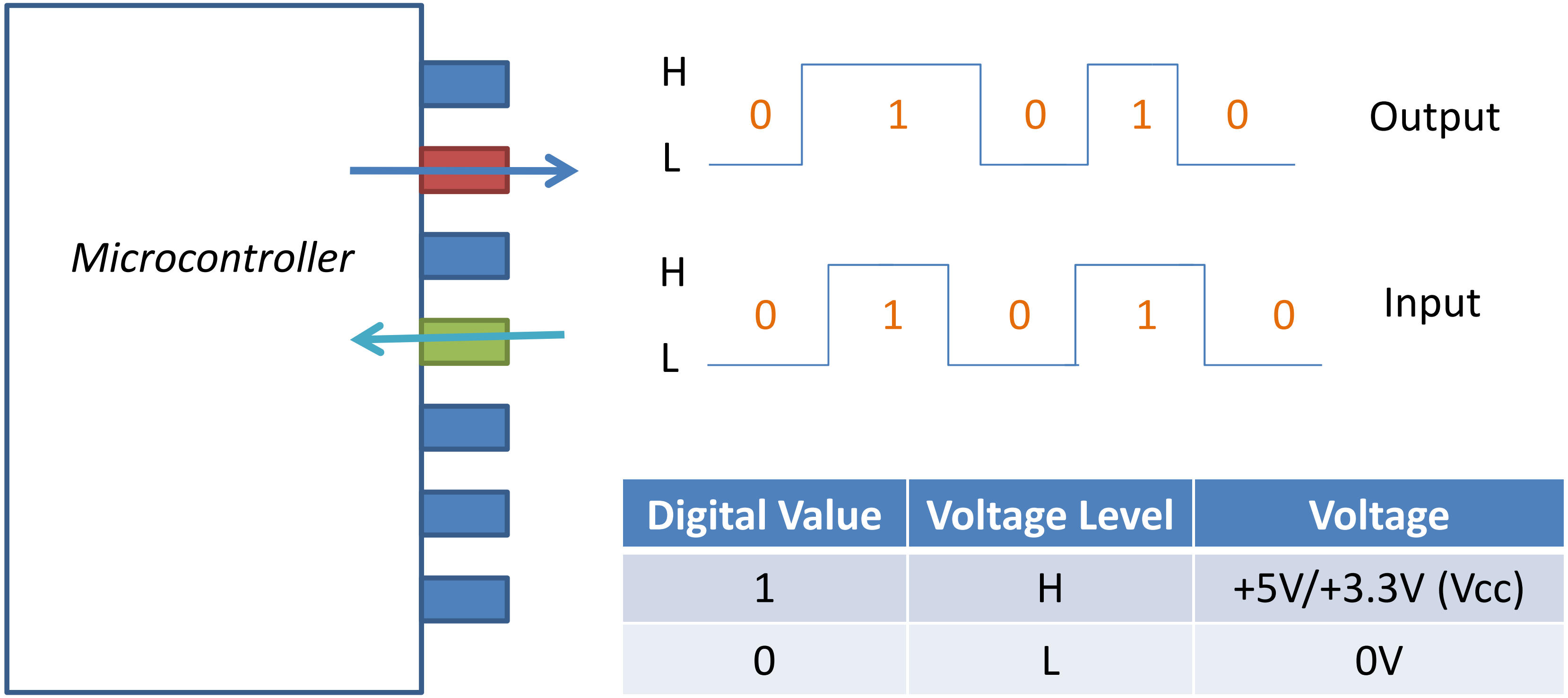
Other types	Use
String	Represents arrays of chars (characters) typically used to contain text.
void	Used only in function declarations where no value is returned.

Not available or Compiler emulation to take a time to calculate

3 I/O port

I/O Port = **Digital** Input / Output port

has Direction, Input or Output. handles 0 or 1, Voltage level: High or Low level



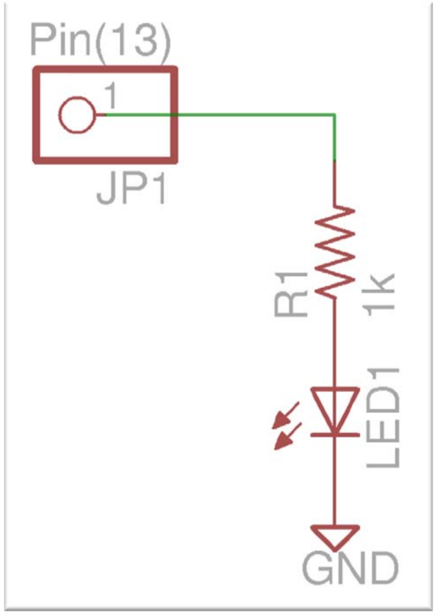
Digital Value	Voltage Level	Voltage
1	H	+5V/+3.3V (Vcc)
0	L	0V

LAB1: I/O Port (Output)

Blink: Blink the LED, Pin(13) is connected to the LED on the board

```
void setup() {    // Initialize peripherals
  // initialize the digital pin as an output.
  // Pin 13 has an LED connected on the board
  pinMode(13, OUTPUT);
}

void loop() {  // Infinite 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
}
```

Polling function

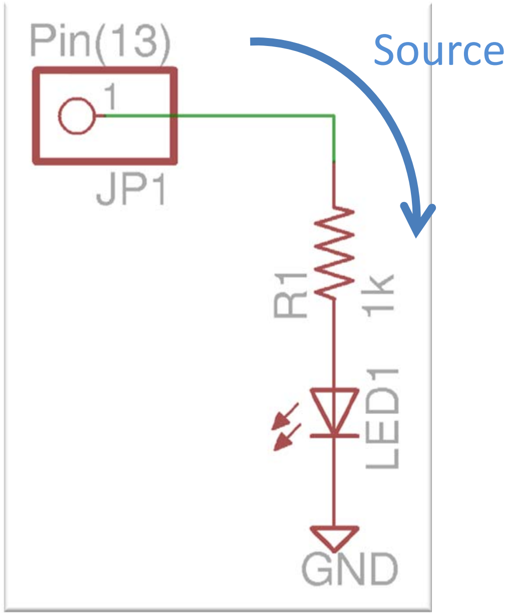


On board LED

LAB2: I/O Port (Output)

BlinkPin13: Declare the pin assignment

```
const int ledPin = 13;

void setup() {    // Initialize peripherals
  pinMode(ledPin, OUTPUT);
}

void loop() {  // Infinite 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
}
```



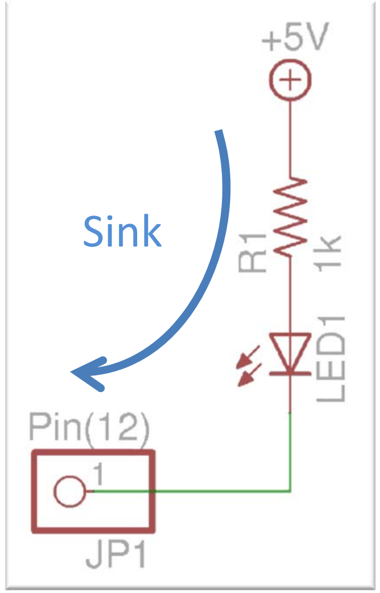
On board LED

LAB3: I/O Port (Output)

BlinkPin12: Make an LED circuit on the board
Connect the circuit to the Pin(12)

```
const int ledPin = 12;

void setup() {    // Initialize peripherals
  pinMode(ledPin, OUTPUT);
}

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

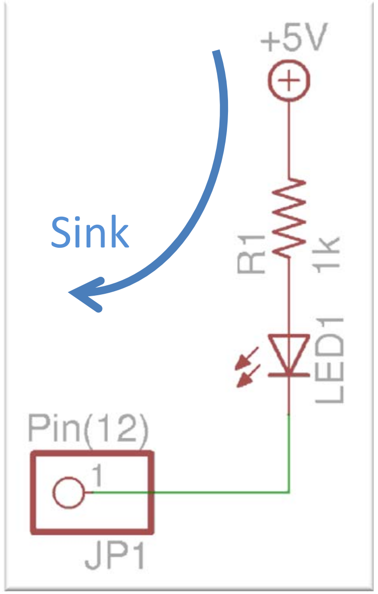
Polling function



LAB4: I/O Port (Output)

BlinkFunc: Use a function

```
const int ledPin = 12;

void setup() {    // Initialize peripherals
  pinMode(ledPin, OUTPUT);
}
void loop() {  // Infinite loop
    blink(1000);
}
void blink(int ms) {
  digitalWrite(ledPin, LOW);   // set the LED on
  delay(ms);              // wait for a second
  digitalWrite(ledPin, HIGH);    // set the LED off
  delay(ms);              // wait for a second
}
```

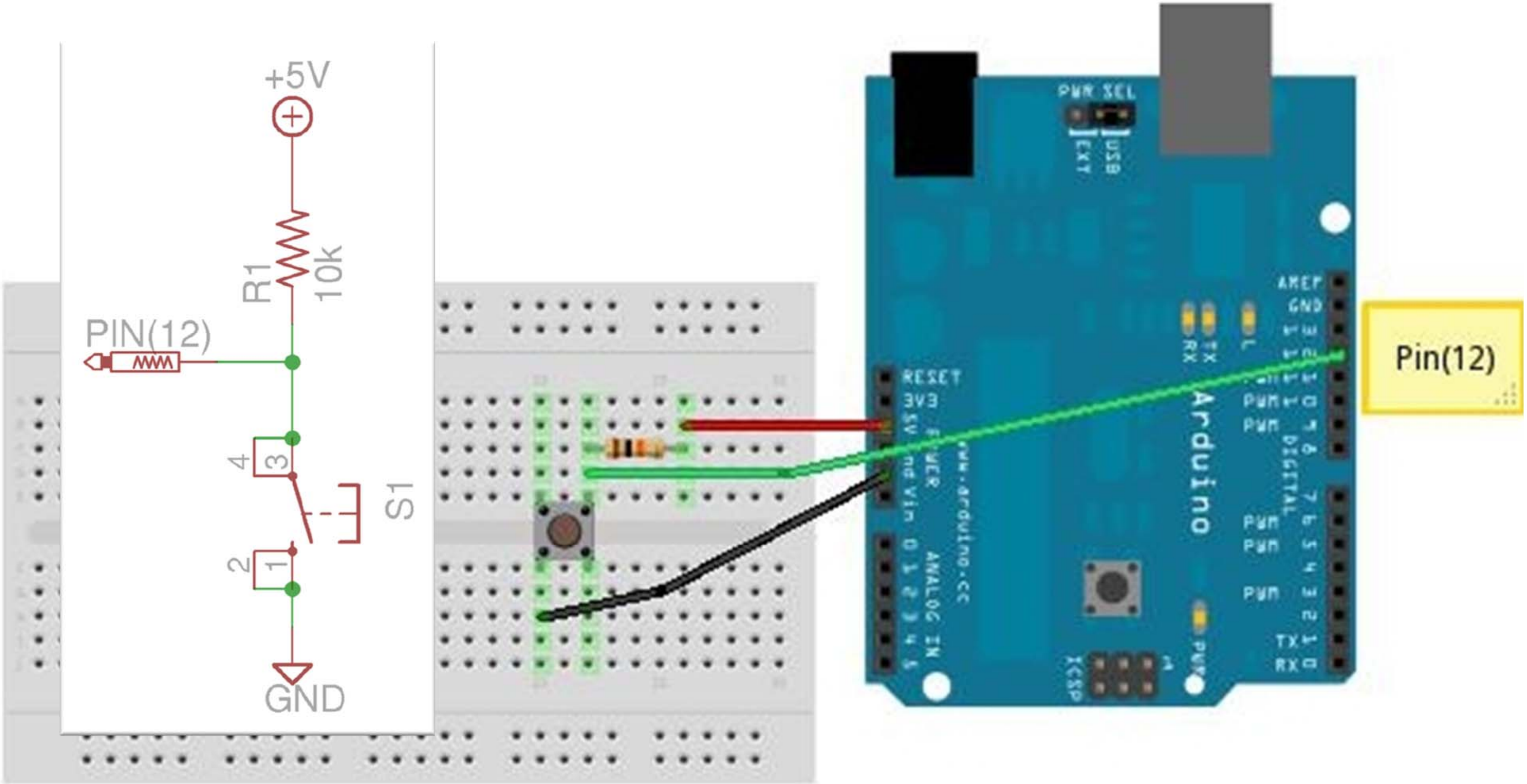


Using a function and a declaration is to keep portability, easy to modify, and well understand the code

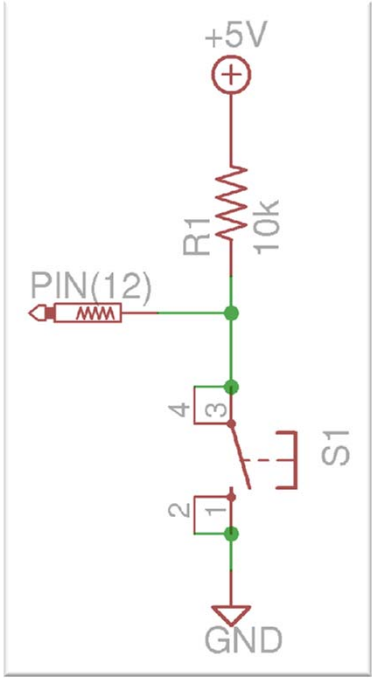
Mar 3, 2012, The University of Tokushima,
Akinori Tsuji

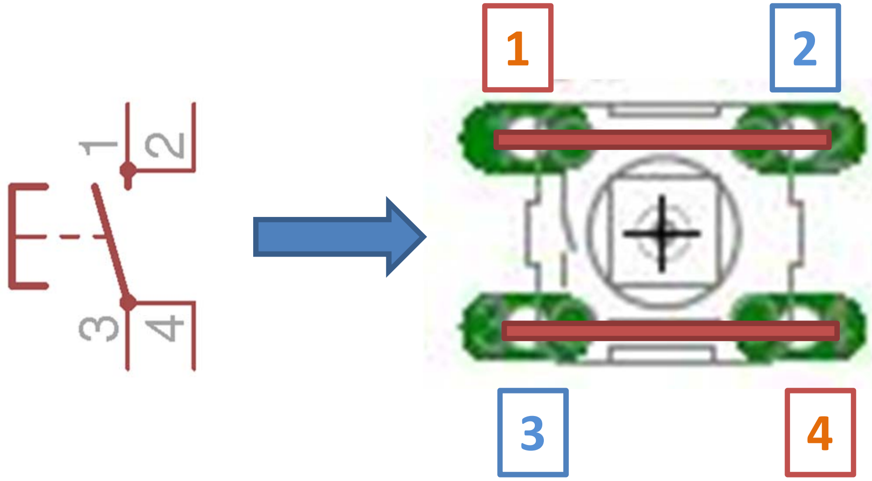
LAB5: I/O Port (Input)

Example) Push switch and LED



Push Switch





Use Diagonal Pins

Schematic

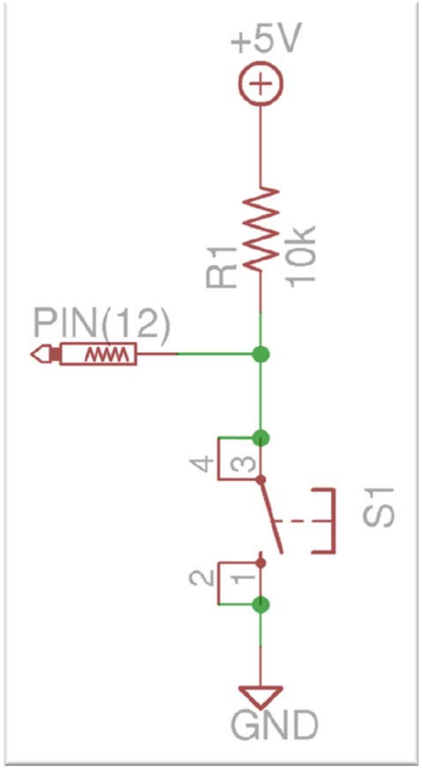
Parts

Switch state	Voltage Level	Digital Value
ON	LOW	0
OFF	HIGH	1

Sample Code

PushSwitch: Push switch and LED

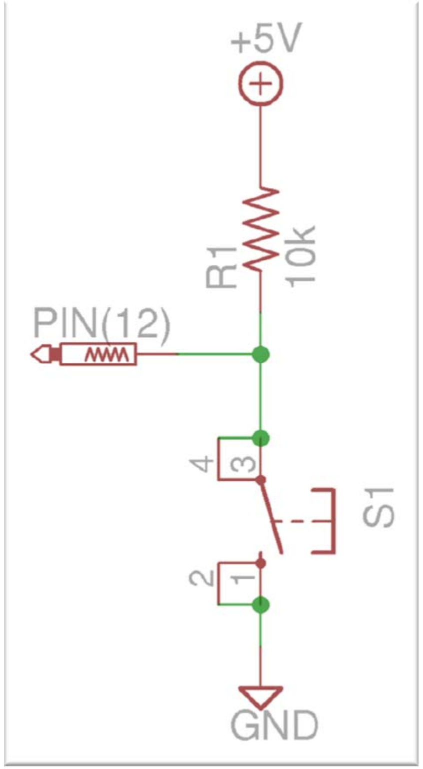
```
const int switchPin = 12;
const int ledPin       = 13;

void setup() {
  pinMode(switchPin, INPUT);  // digital pin as an input
  pinMode(ledPin, OUTPUT); // digital pin as an output
}

void loop() {
  if (digitalRead(switchPin) == LOW) {
      digitalWrite(ledPin, HIGH);   // set the LED on
  } else {
      digitalWrite(ledPin, LOW);    // set the LED off
  }
}
```



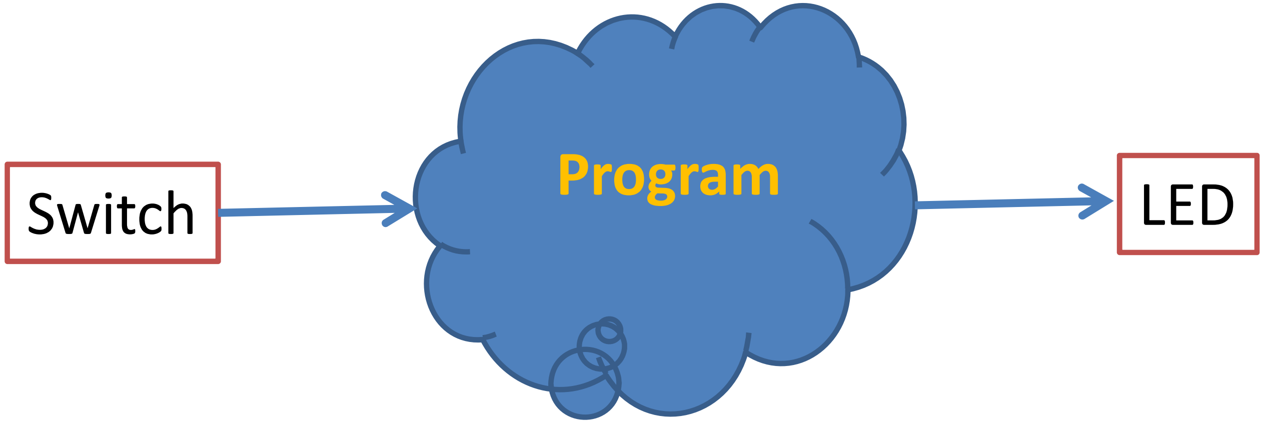
LAB6: I/O Port (Input and Output)

BlinkSwitch: While a switch is pushed, blinking faster

```
const int switchPin = 12;
const int ledPin      = 13;

void setup() {
  pinMode(switchPin, INPUT);  // digital pin as an input
  pinMode(ledPin, OUTPUT); // digital pin as an output
}

void loop() {
  static int  ms = 1000;
  if (digitalRead(switchPin) == LOW) {
      blink(ms);   // set the LED on
      ms -= 50;
      if (ms == 0) ms = 1000;
  }
}
```



I/O Port



Switch

Program

LED

Programming the Microcontroller 2

Day 3

Estimate: 2 hours

2012/3/29(Thu) 10:00—12:00

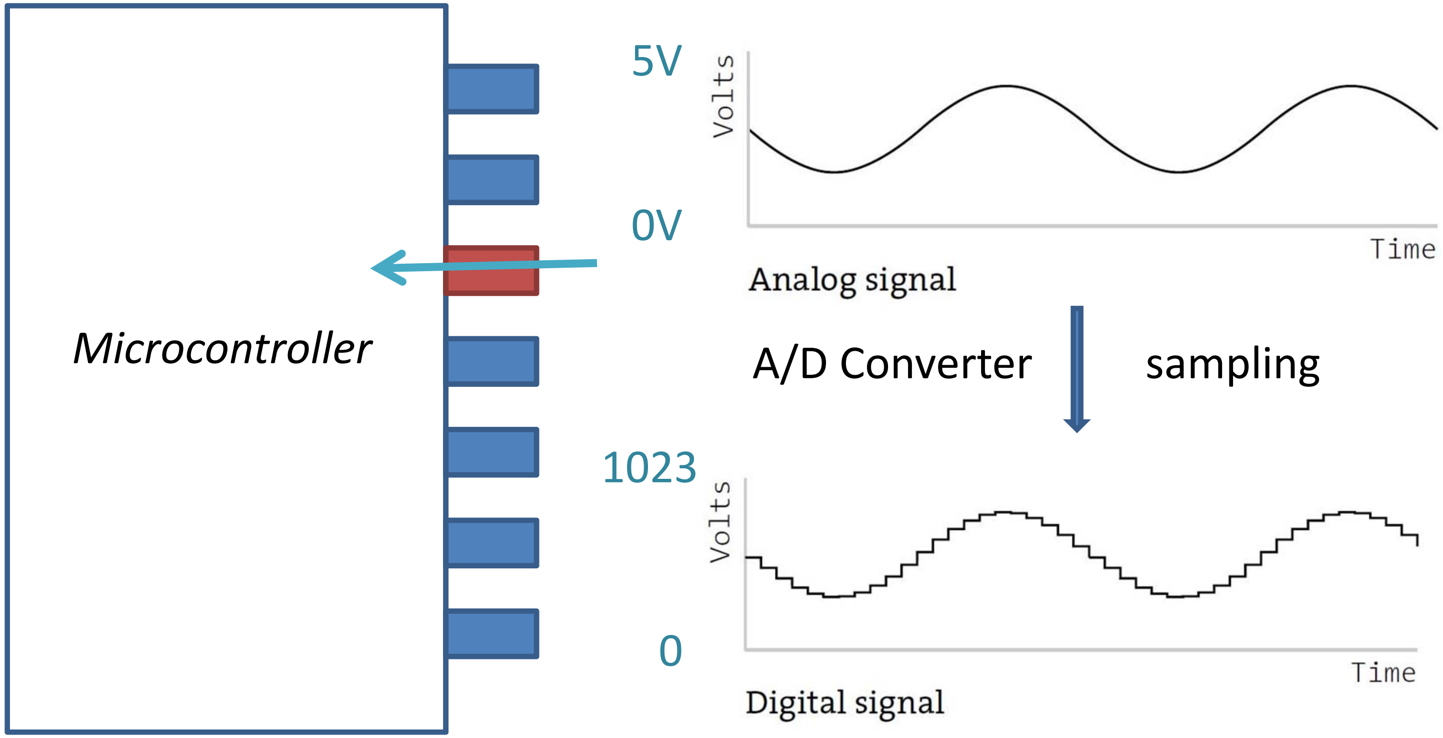
Agenda

1. A/D Converter
2. Timer Interrupt
3. External Interrupt
4. Serial Communication

1 A/D converter

10-Bit Analog to Digital Converter

Analog signal, 0 to 5 V, converts digital value 0 to 1023(2^10-1)



Microcontroller

5V

0V

Analog signal

A/D Converter

sampling

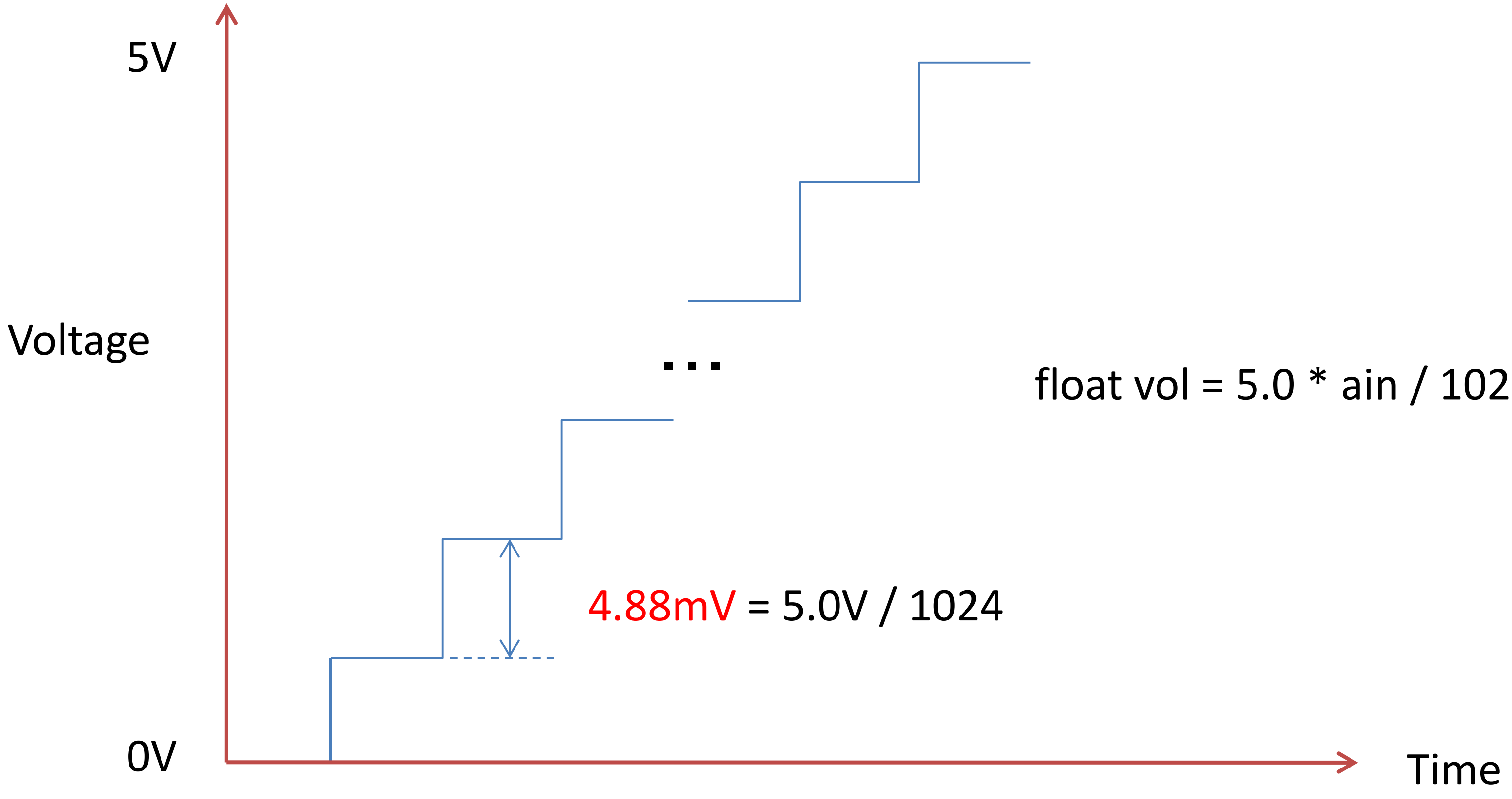
1023

0

Digital signal

1.1 Resolution

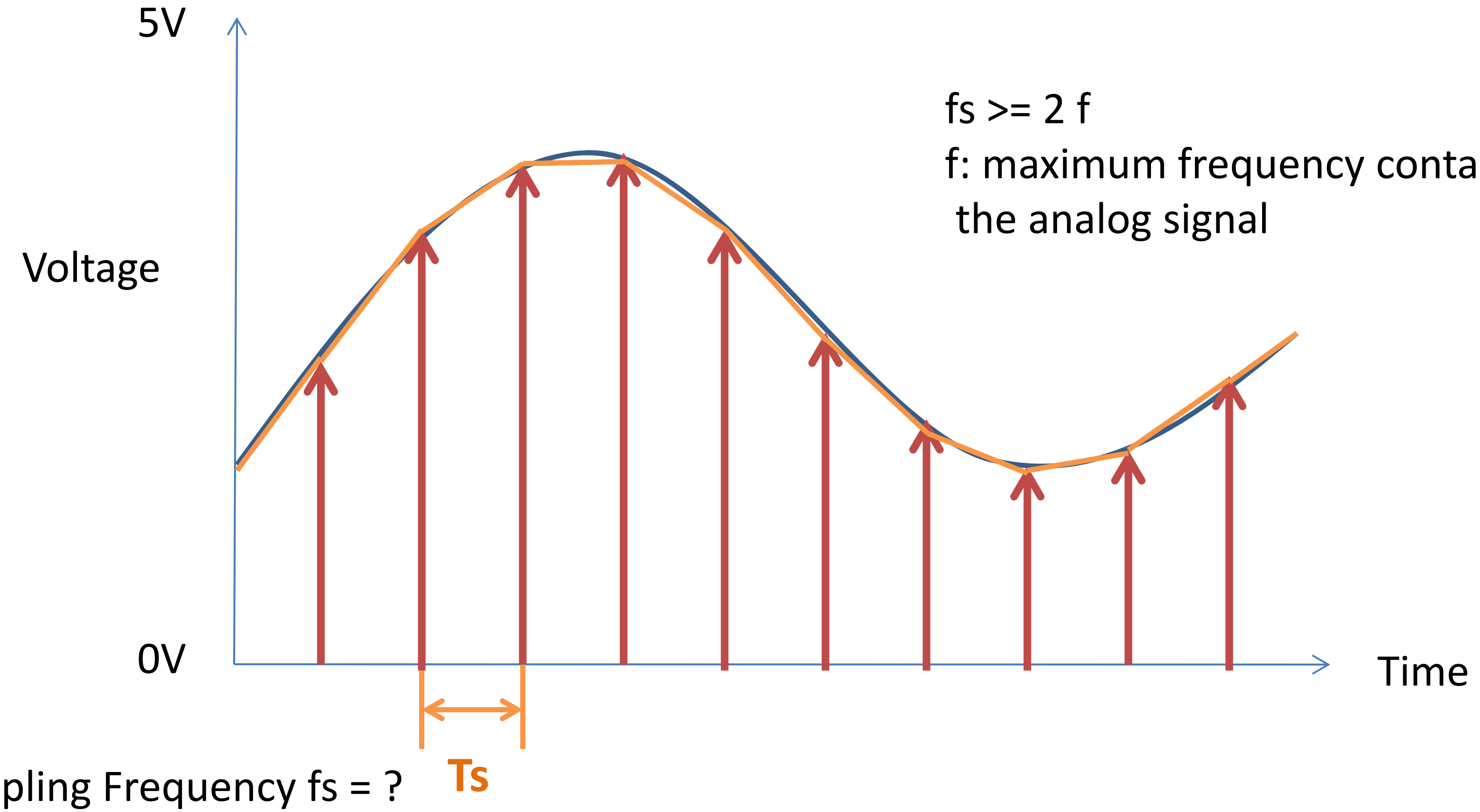
Analog signal, 0 to 5 V, converts digital value 0 to 1023(2^10-1)



float vol = 5.0 * ain / 1024;

1.2 Sampling Frequency

A/D converter: sampled an analog signal followed the sampling frequency



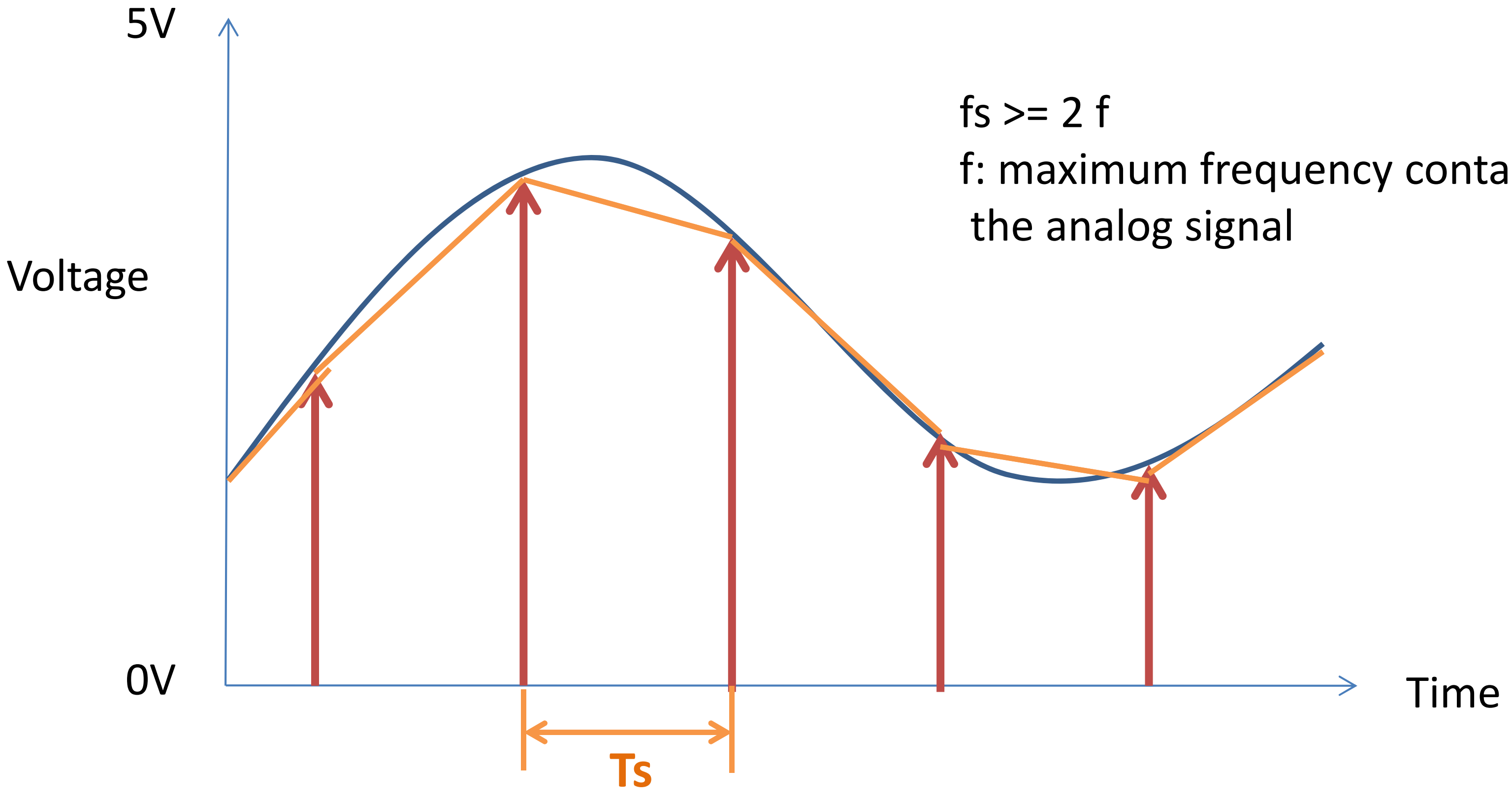
fs >= 2 f

f: maximum frequency containing in the analog signal

Sampling Frequency fs = ?

1.2 Sampling Frequency

A/D converter: sampled an analog signal followed the sampling frequency

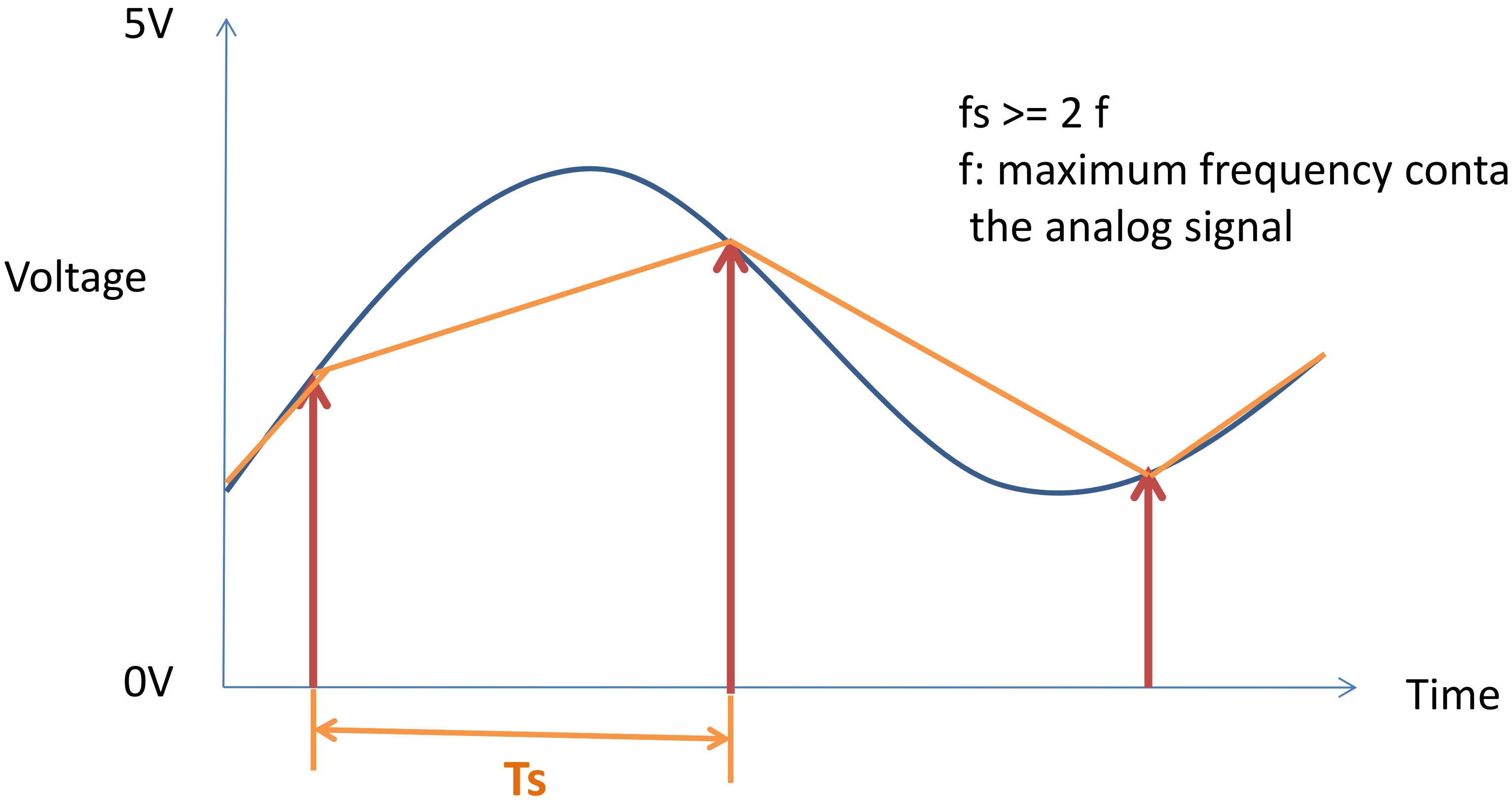


fs >= 2 f

f: maximum frequency containing in the analog signal

1.2 Sampling Frequency

A/D converter: sampled an analog signal followed the sampling frequency

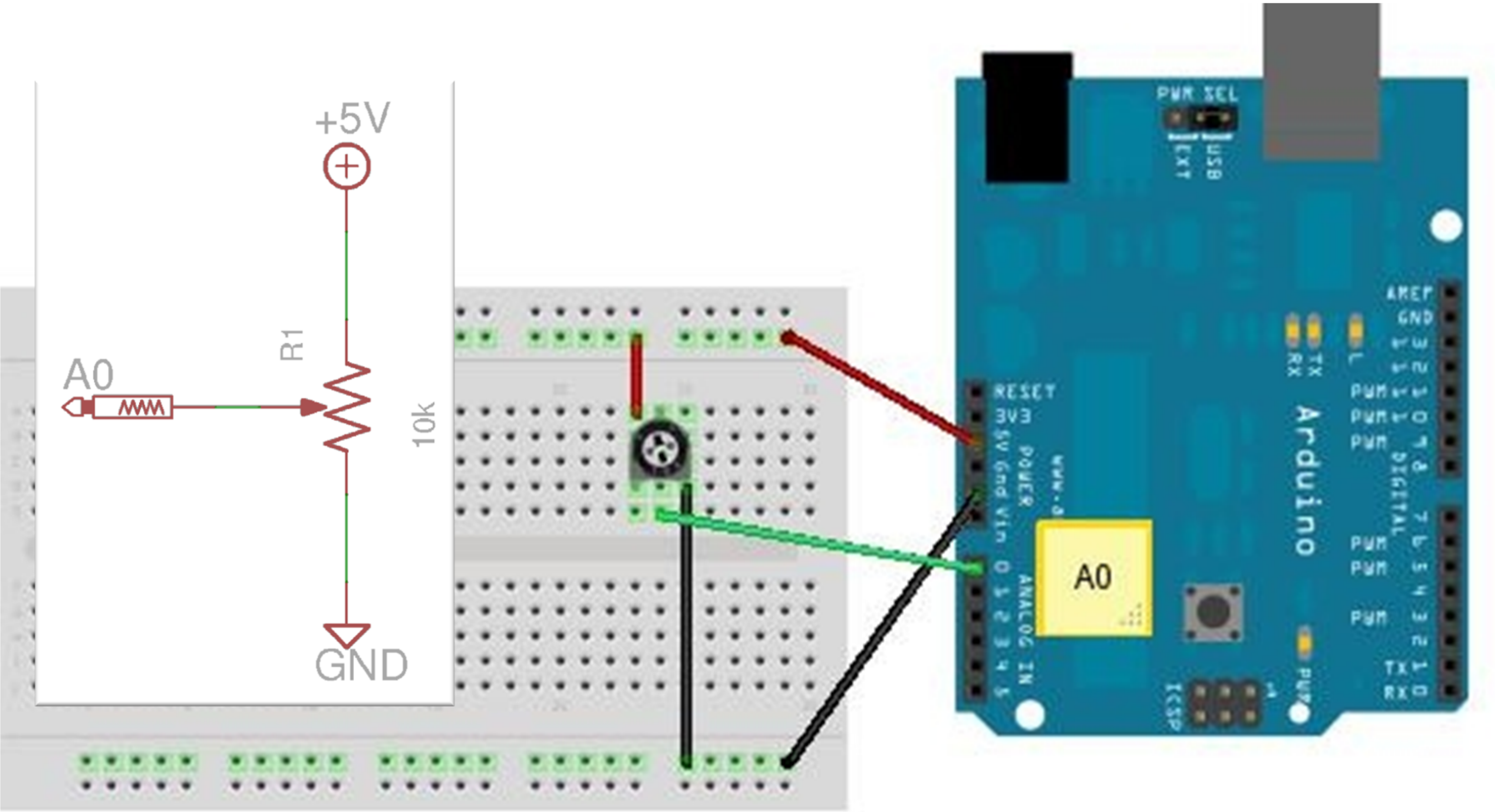


fs >= 2 f

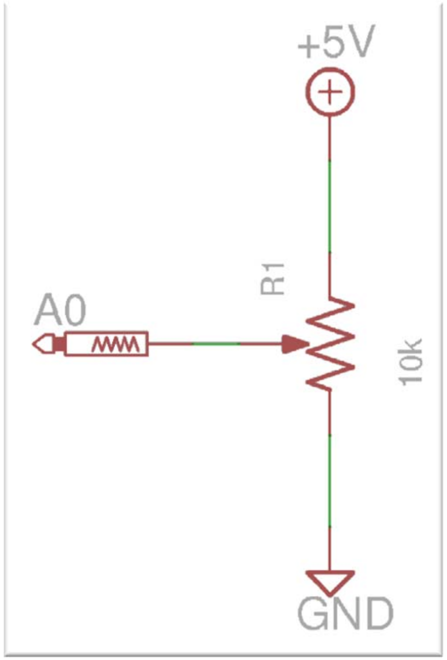
f: maximum frequency containing in the analog signal

LAB1　A/D converter

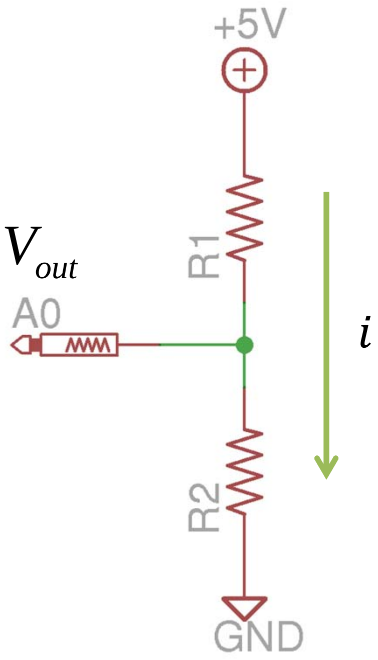
Example) Potentiometer



Potentiometer



Circuit diagram



$$\begin{cases} V_{cc} = (R_1 + R_2)i \\ V_{out} = R_2 i \end{cases}$$

$$V_{out} = \frac{R_2}{R_1 + R_2} V_{cc}$$

LAB1: LED and Potentiometer

SensorIn: Turn on/off the LED depends on the voltage of a potentiometer

```
const int potPin = A0;    // select the analog input pin for the potentiometer
const int ledPin  = 13;     // select the pin for the LED

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

void loop() {
  int potValue = 0;  // variable to store the value coming from the sensor

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

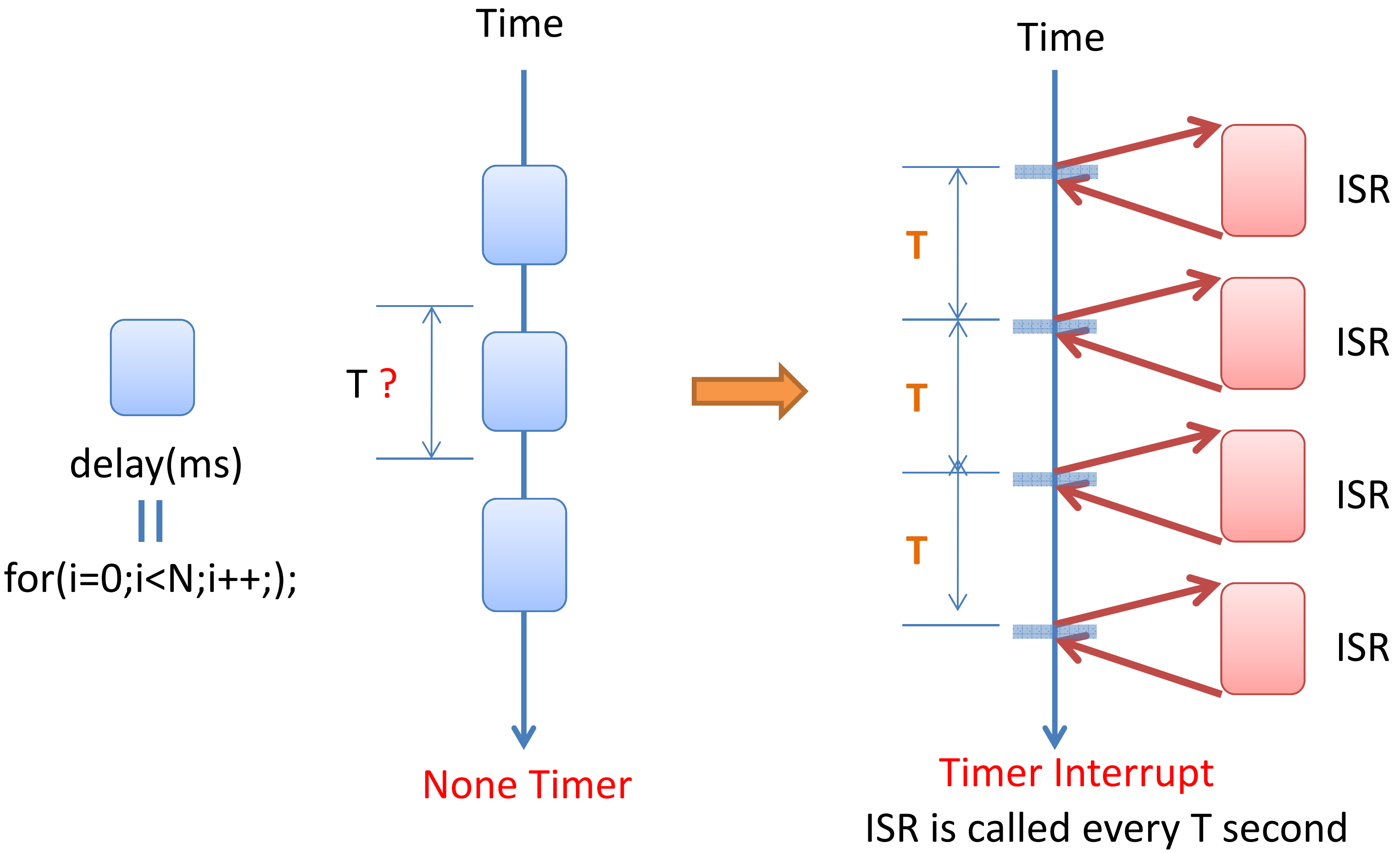
LAB2: LED and Potentiometer

float vol = 5.0 * ain / 1024;

SensorIn: Turn on the LED more than 2.5V

```
const int potPin = A0;    // select the analog input pin for the potentiometer
const int ledPin  = 13;     // select the pin for the LED

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

void loop() {
  int potValue = 0;  // variable to store the value coming from the sensor

  potValue = analogRead(potPin);    // read the value from the sensor:
  if (potValue >= 512) {
    digitalWrite(ledPin, HIGH);   // turn the ledPin on
  } else {
    digitalWrite(ledPin, LOW);  // turn the ledPin off:
  }
}
```

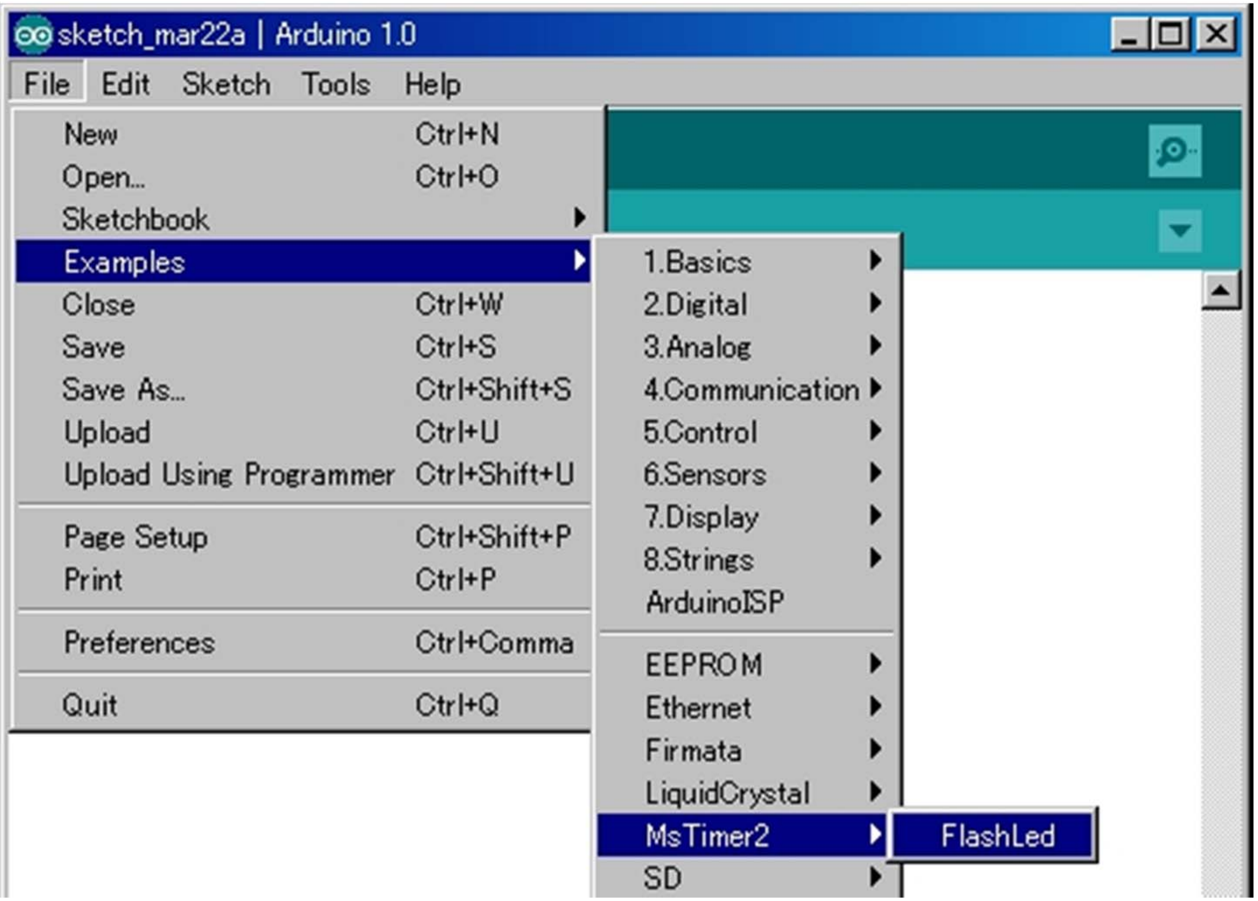
2 Timer Interrupt



ISR is called every T second

MSTimer2 Library

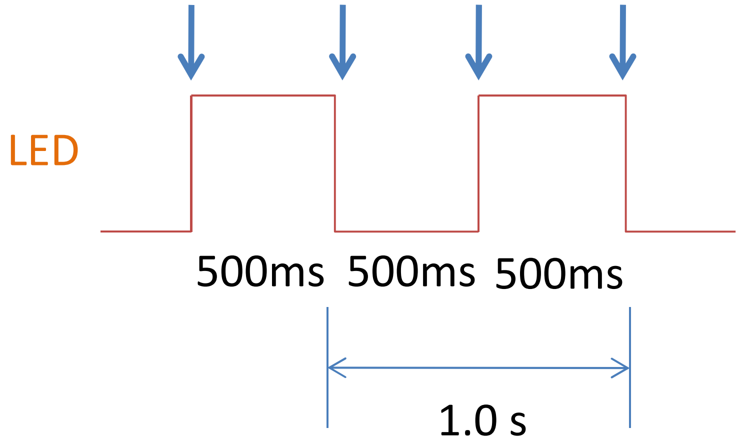
1. Extract the archive "MsTimer2.zip".
2. Move MsTimer2 folder to C:¥arduino-1.0¥libraries
3. Run Arduino
4. File -> Examlpes -> MsTimer2 -> FlashLed



LAB3: Timer Interrupt

TimerInt: Turn on and off the LED every 500 ms

```
#include <MsTimer2.h>
const int ledPin = 13;

// ISR: Interrupt service routine
void flash() {
  static boolean output = HIGH;
  digitalWrite(ledPin, output);
  output = !output;  // toggle the LED
}
void setup() {
  pinMode(ledPin, OUTPUT);

  MsTimer2::set(500, flash); // 500ms period, attach an interrupt service routine
  MsTimer2::start();         // Timer starts
}
void loop() {
  // Nothing to do
}
```



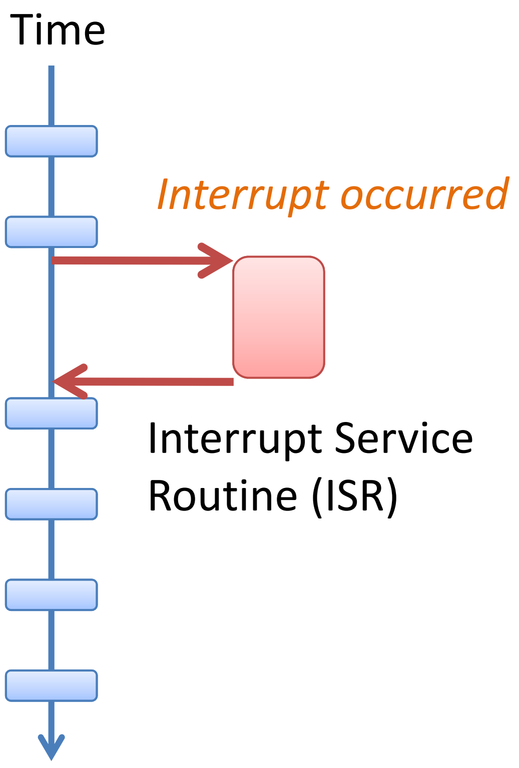
3 External Interrupt

```
attachInterrupt(
  interrupt,
  function,
  [LOW,CHANGE,RISING,FALLING]
);
```

interrupt
- 0 for the pin(1)
- 1 for the pin(2)

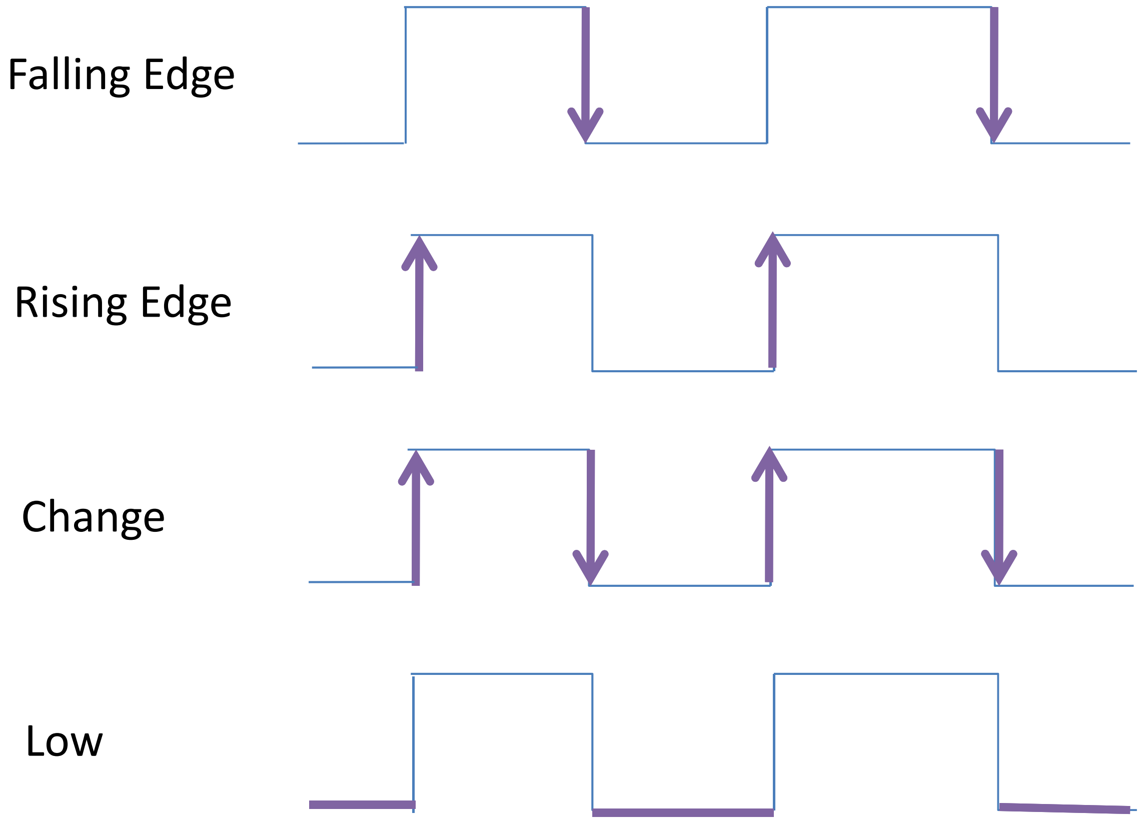
function:
- the name of the Interrupt service routine

conditions:
- LOW, CHANGE, RISING, FALLING



Polling with Interrupt

3.1 External Interrupt



Falling Edge

Rising Edge

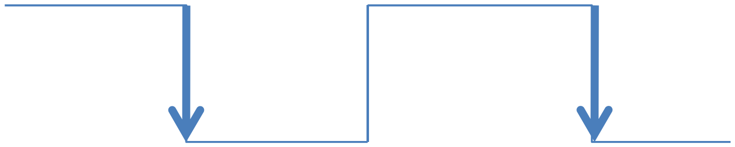
Change

Low

LAB4: External Interrupt

ExternalInt: Toggle the LED when the switch has pushed

```
const int ledPin = 13;
const int intPin0 = 2; // interrupt 0, pin(2)
volatile int state = LOW;
void setup()
{
  pinMode(ledPin, OUTPUT);
  digitalWrite(intPin0, HIGH);  // set HIGH state
  pinMode(intPin0, INPUT);
  attachInterrupt(0, isrSwitch, FALLING); // switch pin on interrupt 0 (pin 2)
}
void loop() {
  digitalWrite(ledPin, state);
}
void isrSwitch() {
  state = !state;  // NO delay function in the interrupt function
}
```



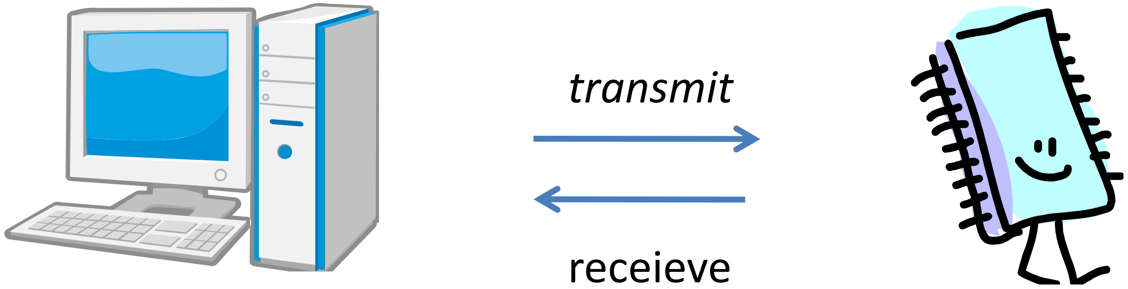
Falling Edge

4 Serial Communication

Receive/Transmit some characters between the PC and the microcontroller

UART = Universal Asynchronous Receiver Transmitter

How much speed?



transmit

receieve

Build a program

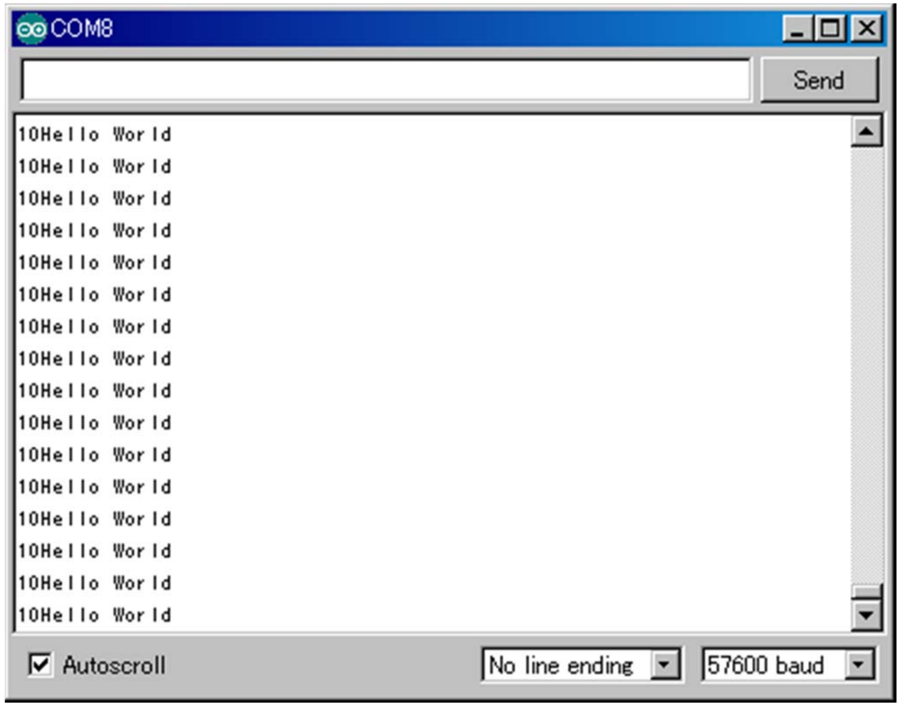
Microcontroller

LAB5: Send data to PC

SendChar: Send characters by the print function

```
void setup() {
  Serial.begin(57600);
}

void loop()
{
   int  n = 10;
   Serial.print(n);
   Serial.println(“Hello World”);
}
```

Tools -> Serial Monitor (Set 57600 baud)



```
10Hello World
10Hello World
10Hello World
10Hello World
10Hello World
10Hello World
10Hello World
10Hello World
10Hello World
10Hello World
10Hello World
10Hello World
10Hello World
10Hello World
10Hello World
10Hello World
```

☑ Autoscroll   No line ending   57600 baud

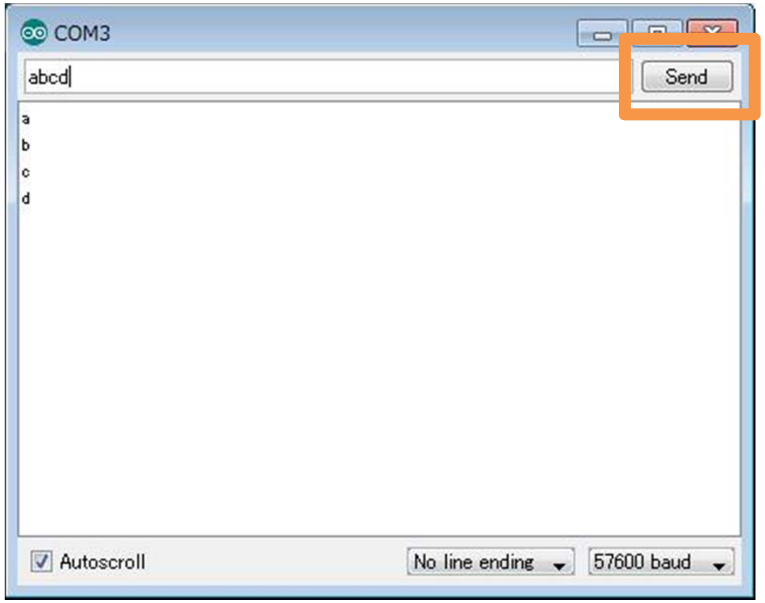
LAB6: Receive data from PC

RcvChar: Blink the LED when MCU received a character

```
const int ledPin = 13;

void setup() {
  Serial.begin(57600);
  pinMode(ledPin, OUTPUT);
}

void loop()
{
  // Check if at least one character is available
  if (Serial.available()) {
      char ch = Serial.read(); // read one character
      blink(500);  // blink LED
      Serial.println(ch); // loop back the character
  }
}
```

Tools -> Serial Monitor

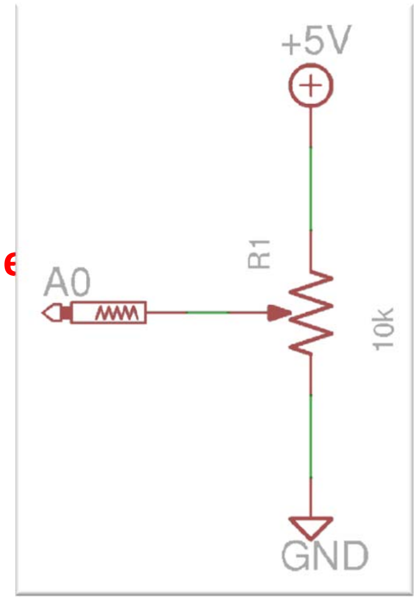
Enter characters and click on the “Send” Button



abcd

```
a
b
c
d
```

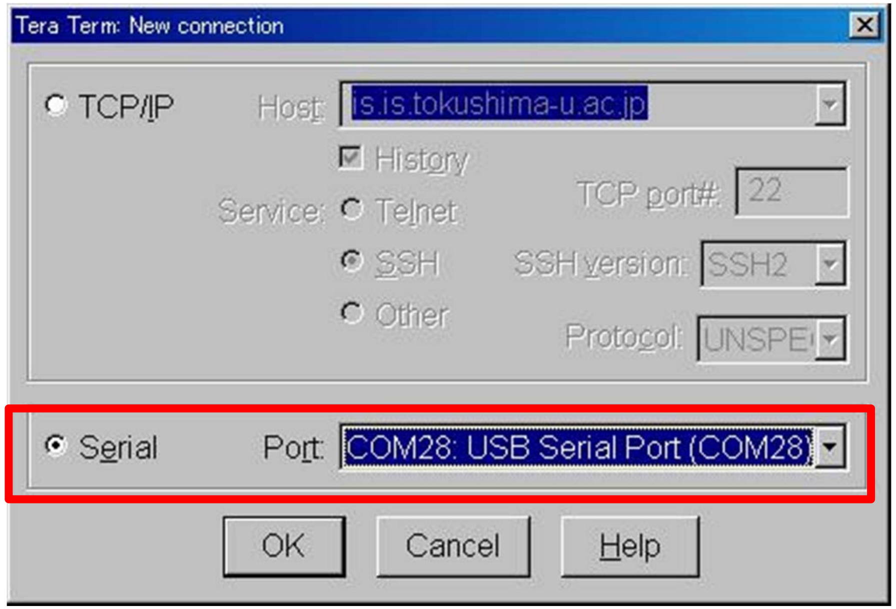
Data Acquisition



DataAcq: Read the output voltage of a potentiometer

```
const int sensorPin = A0;    // select the input pin for the potentiome

void setup() {
  Serial.begin(57600);
}

void loop() {
  int sensorValue ;  // variable to store the value coming from the sensor

  sensorValue = analogRead(sensorPin);    // read the value from the sensor:
  float volt = sensorValue * 5.0 / 1024;  // converts to the voltage
  Serial.println(volt);
}
```

1. Run TeraTerm
2. Choose Port COMxx: USB Serial Port and Click OK



○ TCP/IP Host: is.is.tokushima-u.ac.jp

☑ History

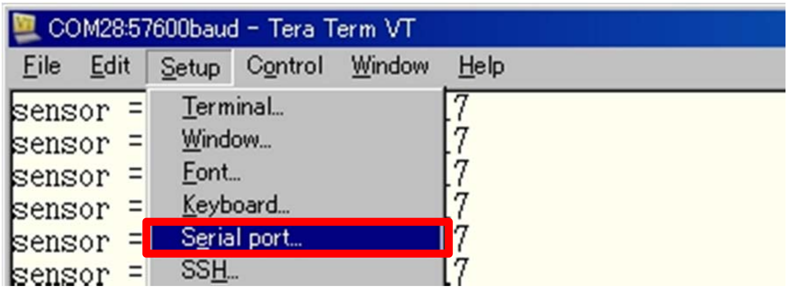
Service: ○ Telnet TCP port#: 22

● SSH SSH version: SSH2

○ Other Protocol: UNSPE

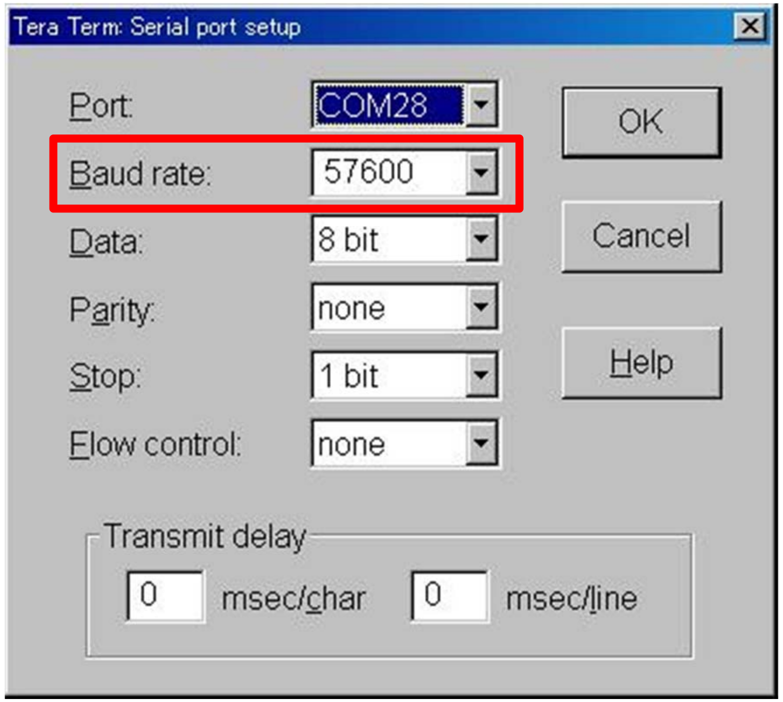
● Serial Port: COM28: USB Serial Port (COM28)

3. Setup -> Serial Port



```
sensor =
sensor =
sensor =
sensor =
sensor =
sensor =
```

4. Setup the Port as the following setting



Port: COM28

Baud rate: 57600

Data: 8 bit

Parity: none

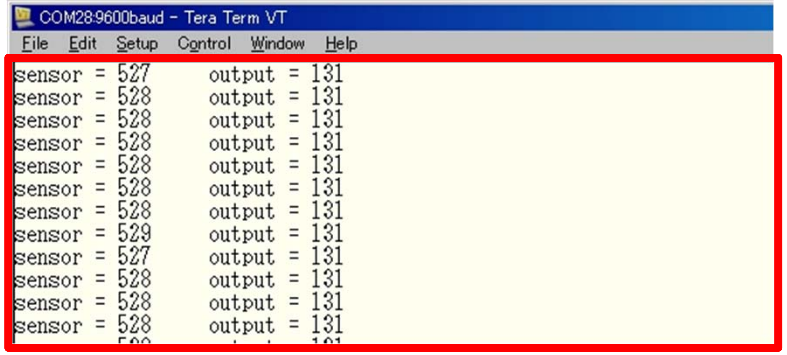
Stop: 1 bit

Flow control: none

Transmit delay

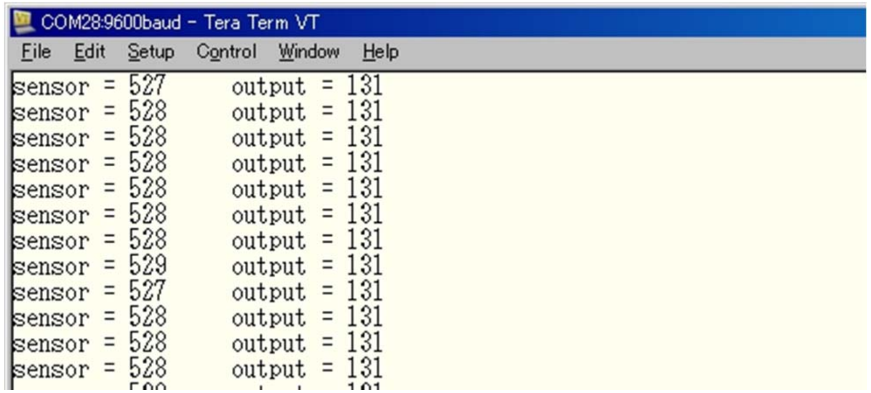
0 msec/char 0 msec/line

5. Data is coming from the microcontroller



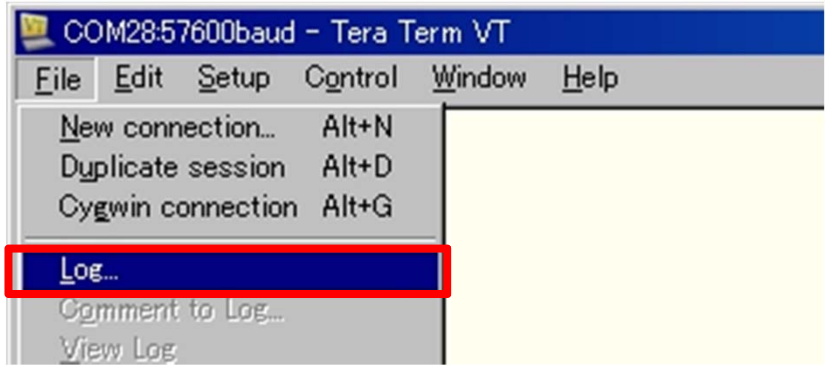
```
sensor = 527     output = 131
sensor = 528     output = 131
sensor = 528     output = 131
sensor = 528     output = 131
sensor = 528     output = 131
sensor = 528     output = 131
sensor = 528     output = 131
sensor = 529     output = 131
sensor = 527     output = 131
sensor = 528     output = 131
sensor = 528     output = 131
sensor = 528     output = 131
```

1. Get data from sensors

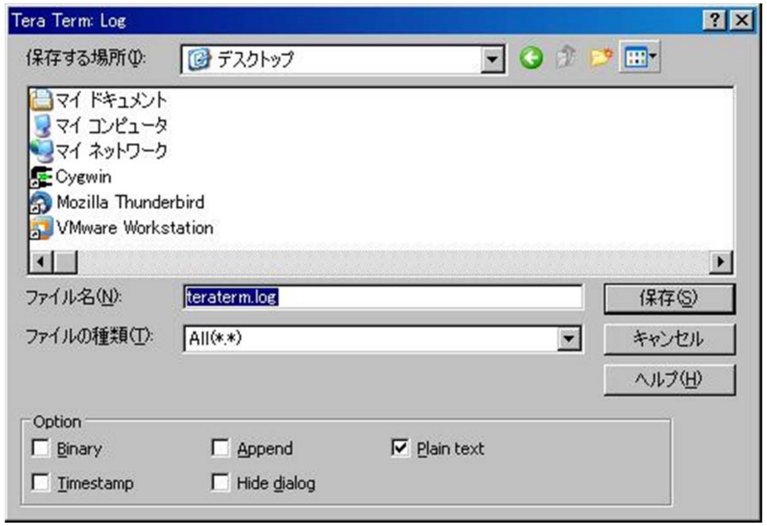


```
sensor = 527      output = 131
sensor = 528      output = 131
sensor = 528      output = 131
sensor = 528      output = 131
sensor = 528      output = 131
sensor = 528      output = 131
sensor = 528      output = 131
sensor = 529      output = 131
sensor = 527      output = 131
sensor = 528      output = 131
sensor = 528      output = 131
sensor = 528      output = 131
```

2. File -> log



3. Save as CSV file <filename>.csv and Click on the Save



保存する場所(I): デスクトップ

ファイル名(N): teraterm.log

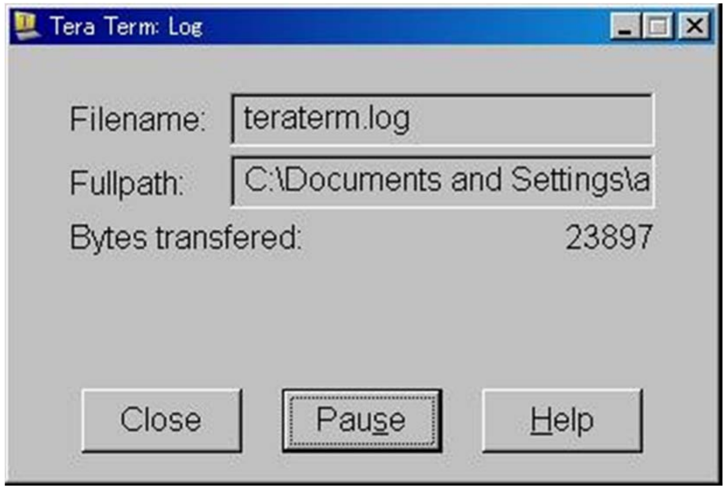
ファイルの種類(T): All(*.*)

Option

□ Binary □ Append ☑ Plain text

□ Timestamp □ Hide dialog

4. Data logging is started



Filename: teraterm.log

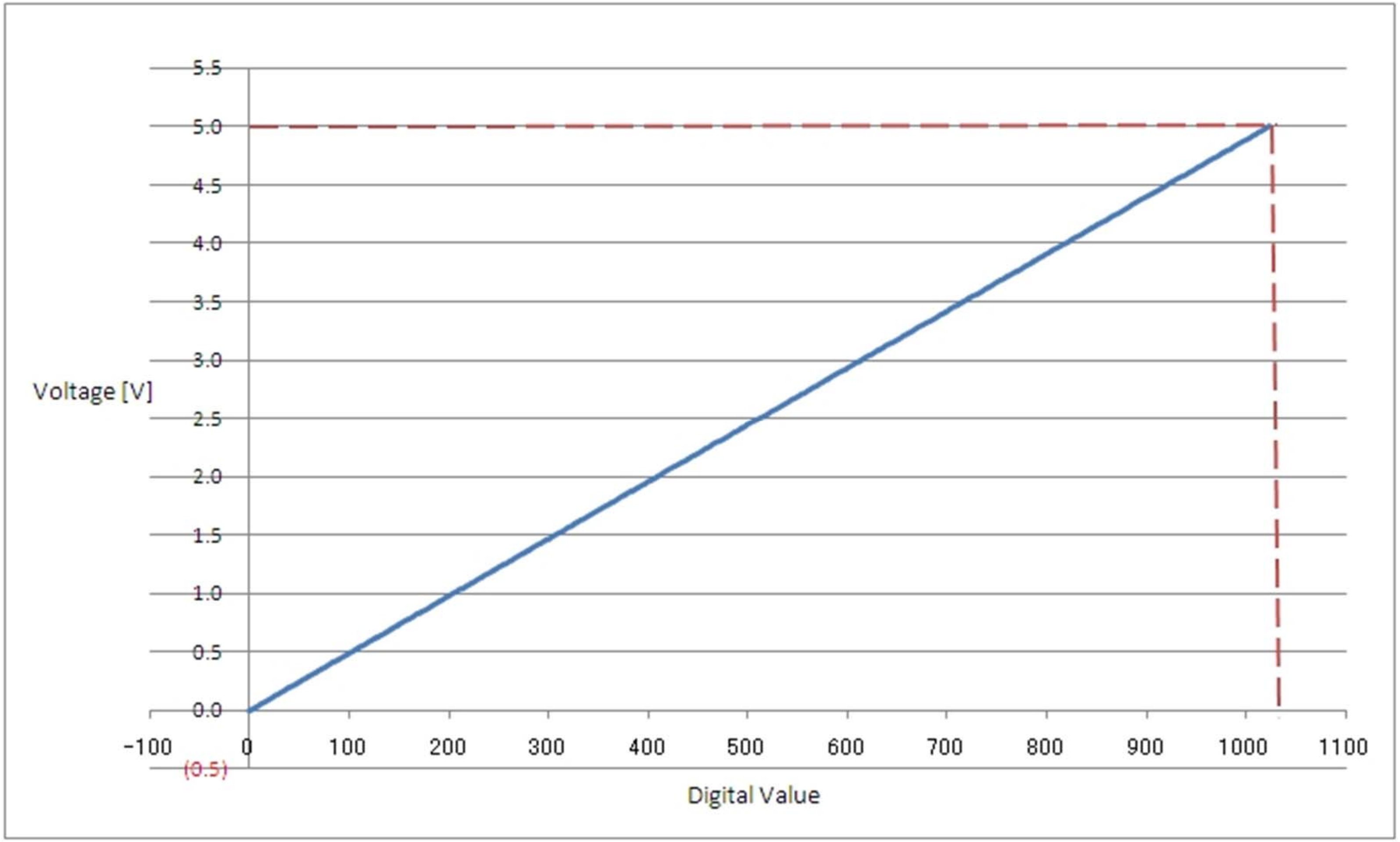
Fullpath: C:\Documents and Settings\a

Bytes transfered: 23897

5. Stop logging by click on the Pause

6. Open a log file by Excel and plot the graph

Example) Characteristics of the potentiometer



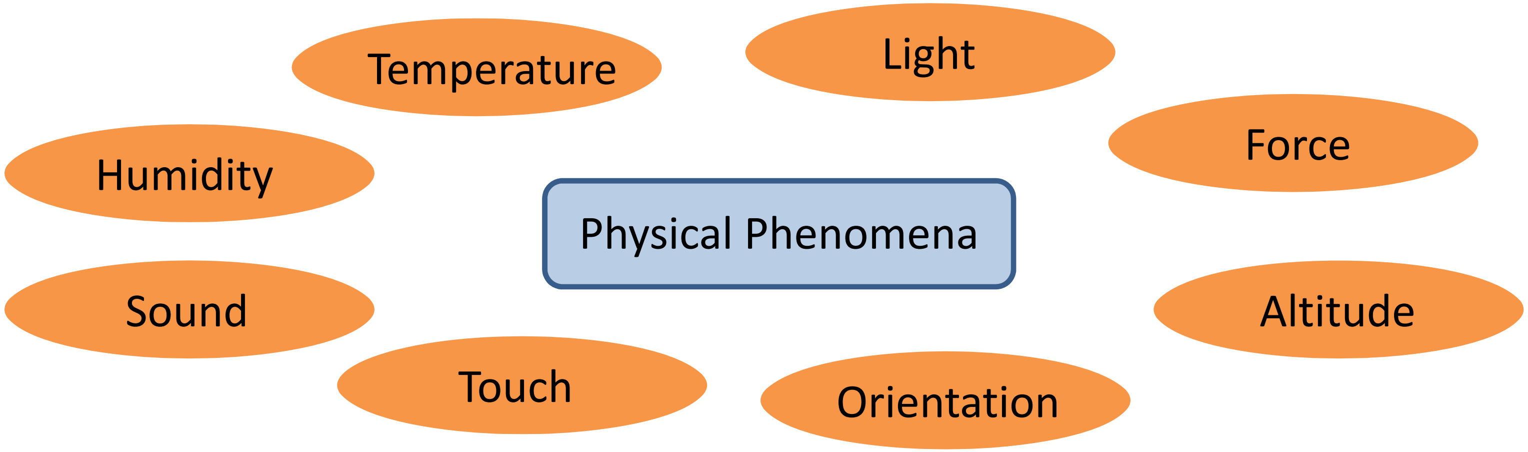
Sensors and Actuators

Day 4

Estimate: 2 hours

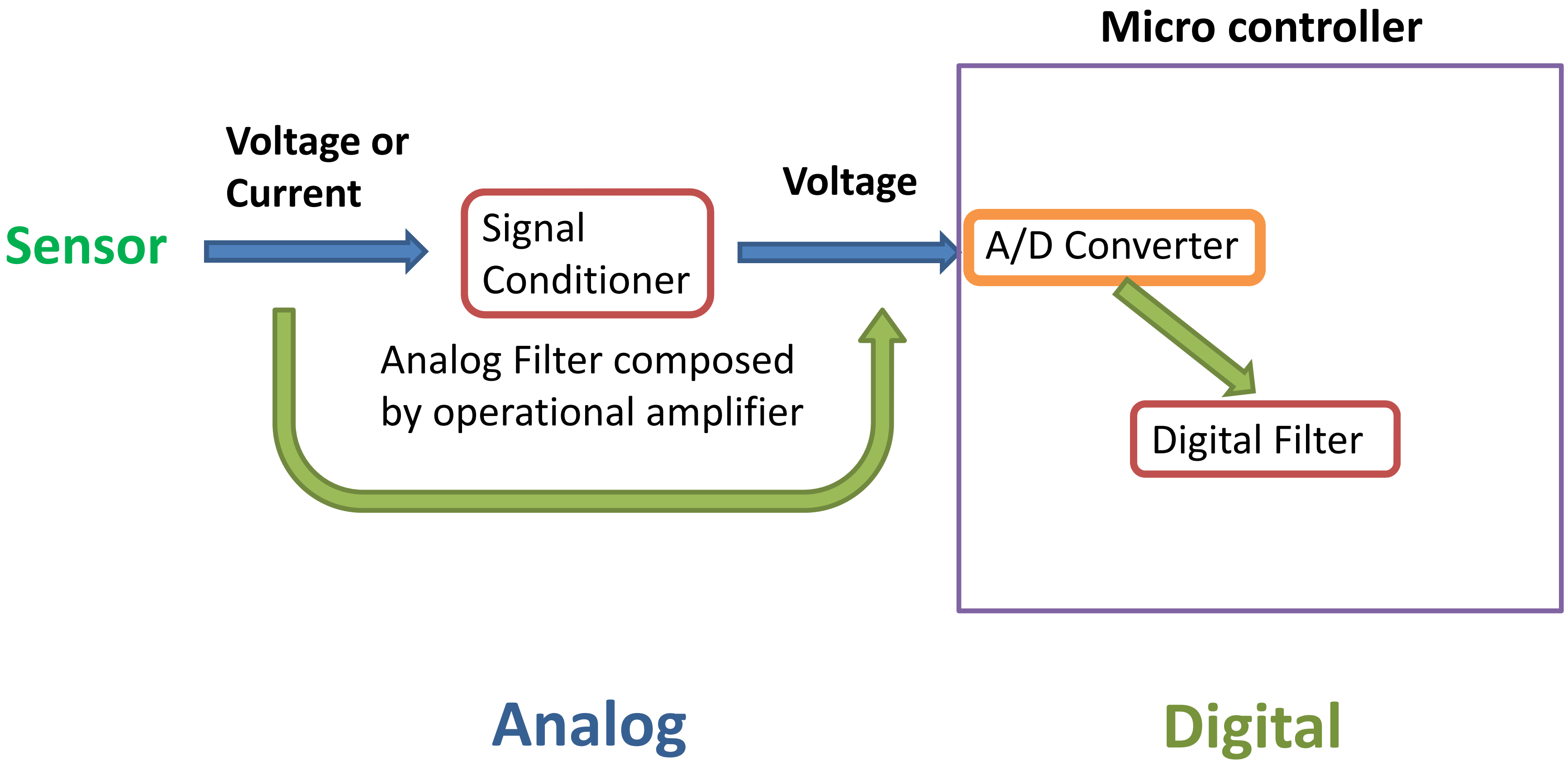
2012/4/2(Mon) 10:00—12:00

1 Sensors and Actuators



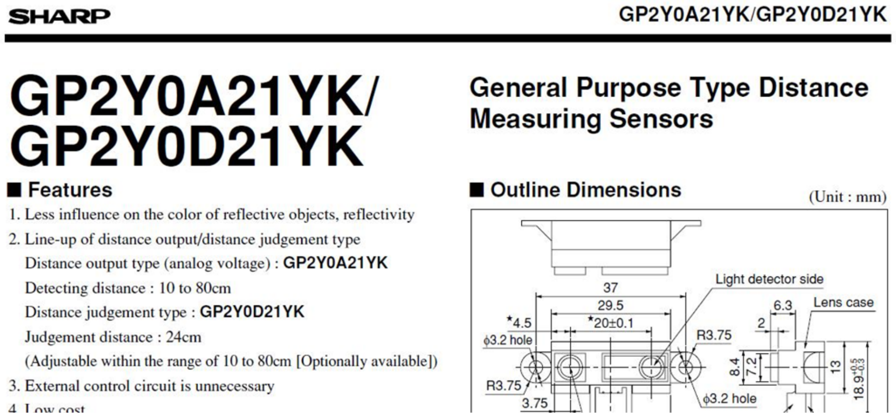
Measured by Sensors

Filter（Analog or Digital)



Datasheet

Device name
Features
Applications
Packages
Pin Assignment
Absolute Maximum Ratings
Characteristics
Block Diagram
Timing Chart
Samle Circuit
Notes



SHARP GP2Y0A21YK/GP2Y0D21YK

GP2Y0A21YK/ GP2Y0D21YK

General Purpose Type Distance Measuring Sensors

■ **Features**

1. Less influence on the color of reflective objects, reflectivity
2. Line-up of distance output/distance judgement type
   Distance output type (analog voltage) : **GP2Y0A21YK**
   Detecting distance : 10 to 80cm
   Distance judgement type : **GP2Y0D21YK**
   Judgement distance : 24cm
   (Adjustable within the range of 10 to 80cm [Optionally available])
3. External control circuit is unnecessary
4. Low cost

■ **Outline Dimensions** (Unit : mm)

2 Temperature Sensor

National Semiconductor LM60BIZ

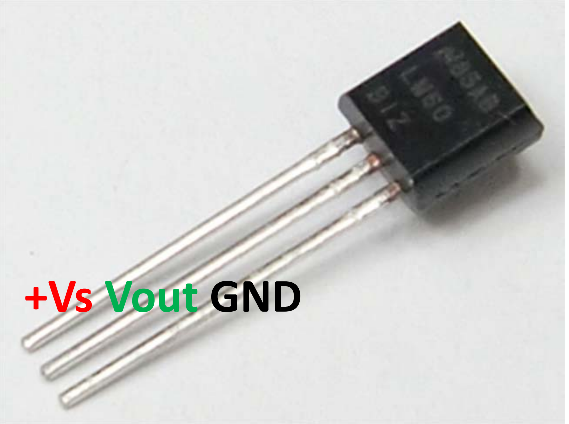
Output: Analog

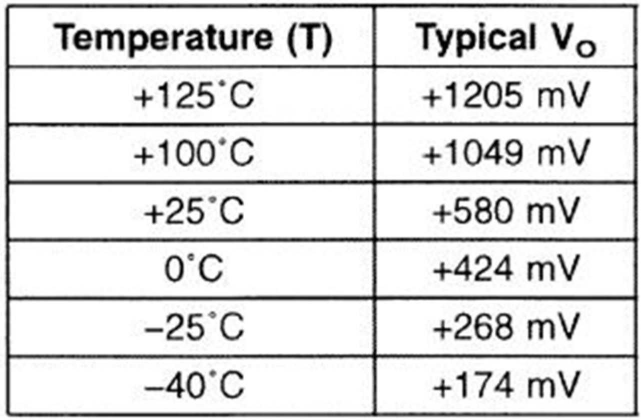
Operating Voltage: DC 2.7V – 10V

Measurement Temperature: -25 deg. -- +125 deg.

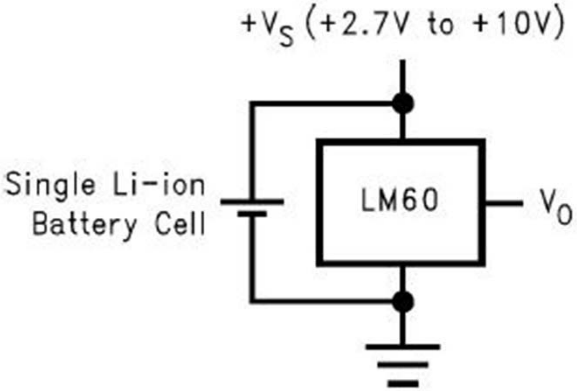
6.25 mV / deg.

Torrelance: ±2 deg. (@ 25 deg.)





Temperature (T)	Typical V_O
+125°C	+1205 mV
+100°C	+1049 mV
+25°C	+580 mV
0°C	+424 mV
−25°C	+268 mV
−40°C	+174 mV



$$V_O = (+6.25\ \text{mV}/℃ \times T℃) + 424\ \text{mV}$$

2.1 Measure the temperature

Voltage:

Vo = +6.25mV * T + 424 mV

6.25 mV * T = Vo – 424 mV

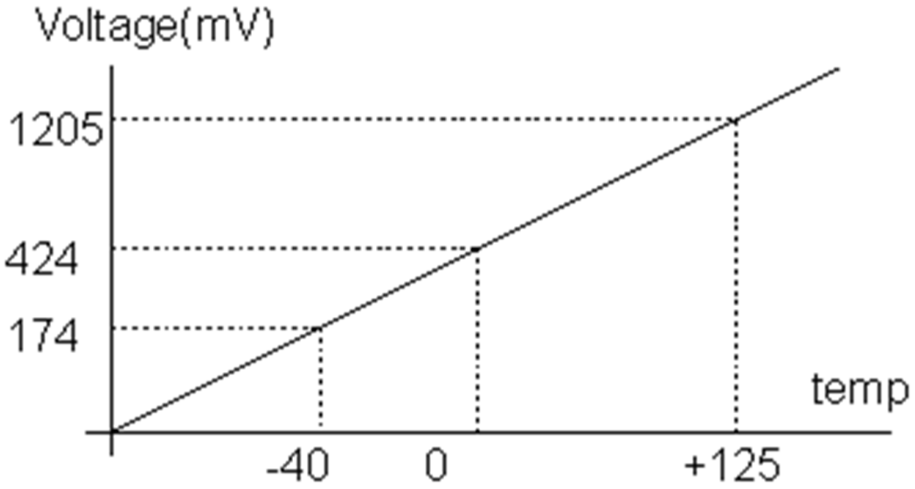
Temperature:

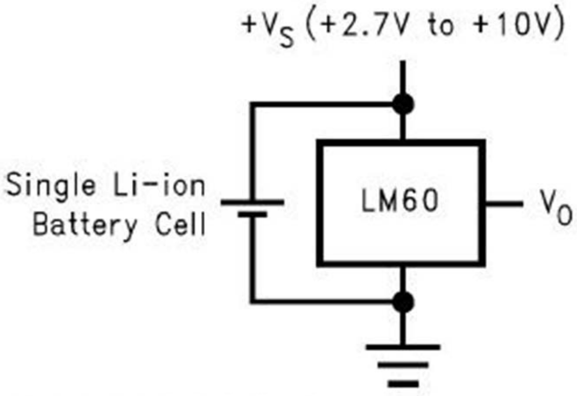
T = (Vo – 424 mV) / 6.25 mV

Analog Value:

Vo = ain * 5.0 / 1024 (V)

= ain * 5000.0 / 1024 (mV)





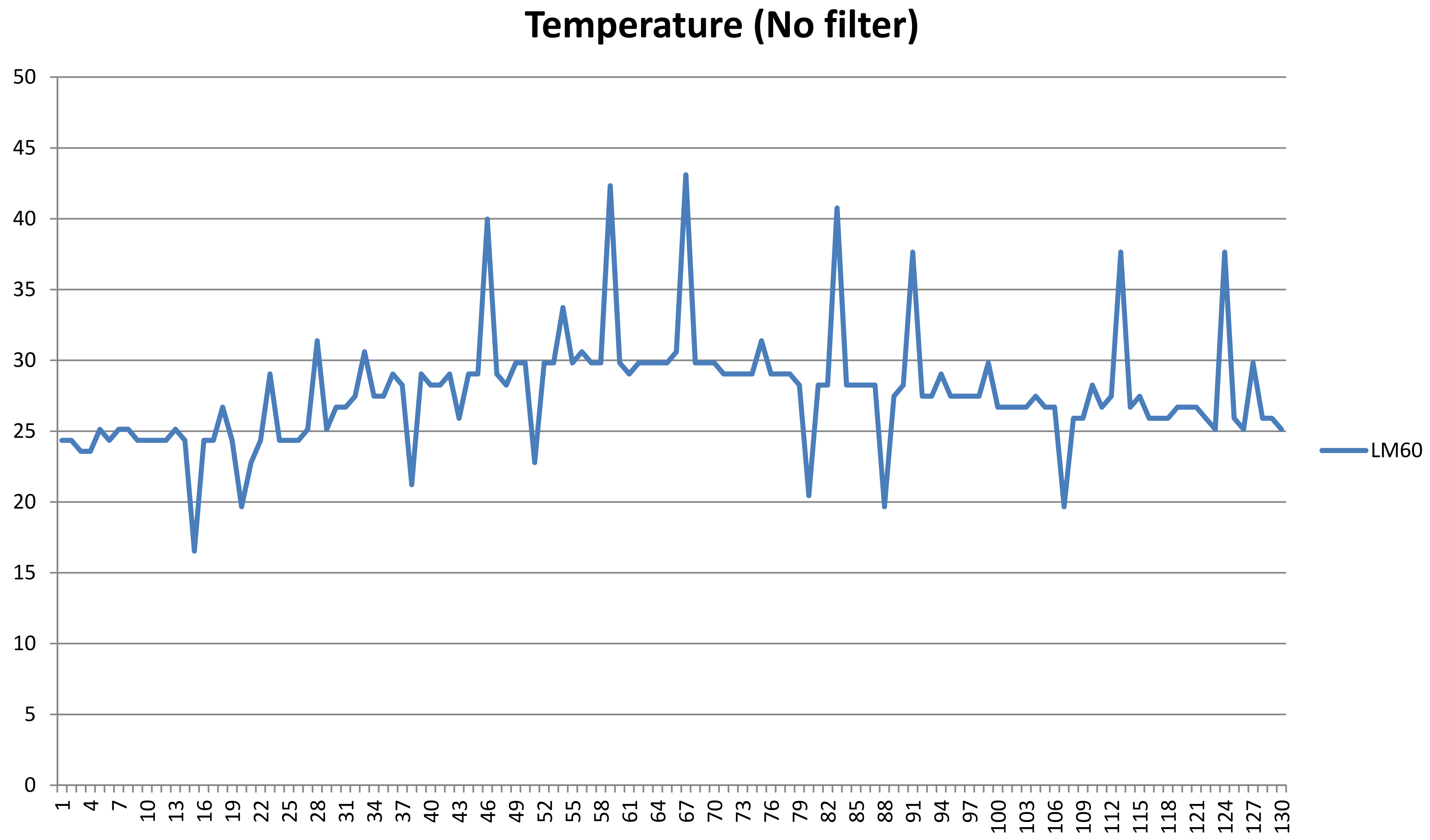
$V_O = (+6.25\ \text{mV}/℃ \times T℃) + 424\ \text{mV}$

Sample Code

TempMeas: Read the output voltage of the temperature sensor

```
const int analogInPin = A0;  // Analog input pin that the temperature sensor is connected
int sensorValue = 0;     // value read from the temperature sensor

void setup() {
  // initialize serial communications at 57600 bps:
  Serial.begin(57600);
}

void loop() {
  sensorValue = analogRead(analogInPin);
  float Vo = sensorValue * 5000.0 / 1024;  // converts to Voltage (mV)
  float T = (Vo - 424) / 6.25; // converts the temperature (Centigrade Degree)

  Serial.println(T); // print the results to the serial monitor:

  delay(10);  // wait 10 miliseconds for the AD converter to settle after the last reading
}
```

Measurement Result



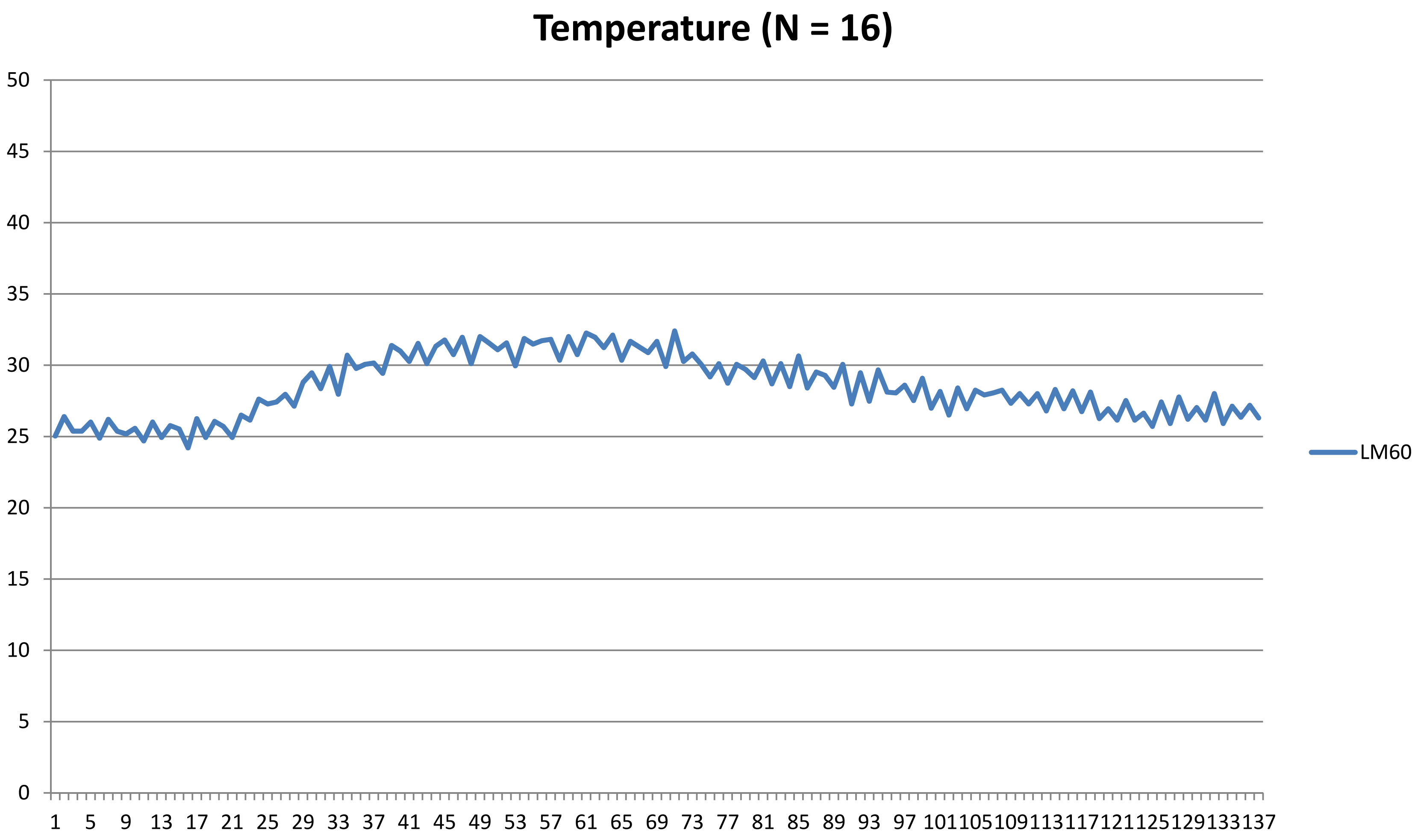
Sample Code (with filter)

TempMeasWithFilter: Read the output voltage of the temperature sensor

```
const int analogInPin = A0;  // Analog input pin that the temperature sensor is connected
const int N = 16;  // Sample number for average filter
int sensorValue = 0;        // value read from the temperature
void setup() {
  Serial.begin(57600);
}
void loop() {
  float T;
  T = 0.0;
  for (int i=0; i<N; i++) {
    sensorValue = analogRead(analogInPin);
    float Vo = sensorValue * 5000.0 / 1024;
    Vo = (Vo - 424) / 6.25;
    T += Vo;
    delay(10);
  }
  Serial.println(T / N);
  delay(500);
}
```

Average filter

Measurement Result (With Filter)



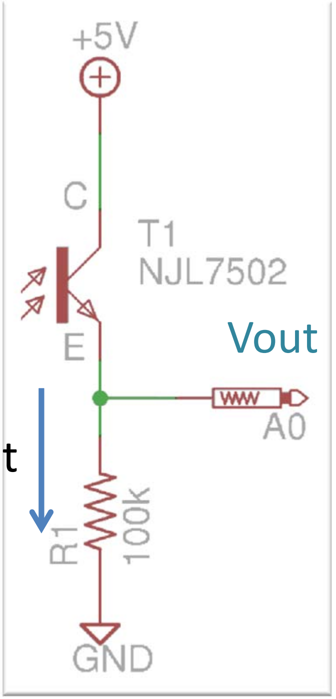
3 Light Sensor

Photo Transistor

JRC NJL7502

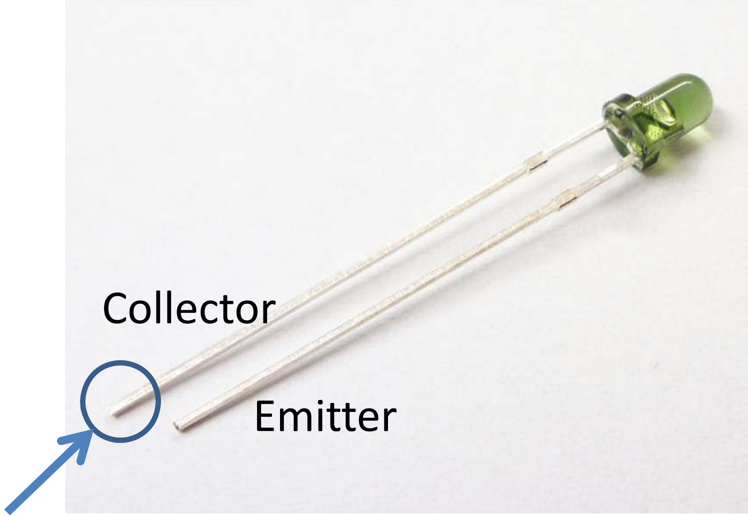
Peak Sensitivity 560 nm

Optical Current 33 uA



Vout

Iopt



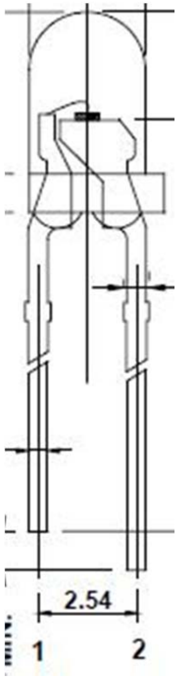
Collector

Emitter

Long lead

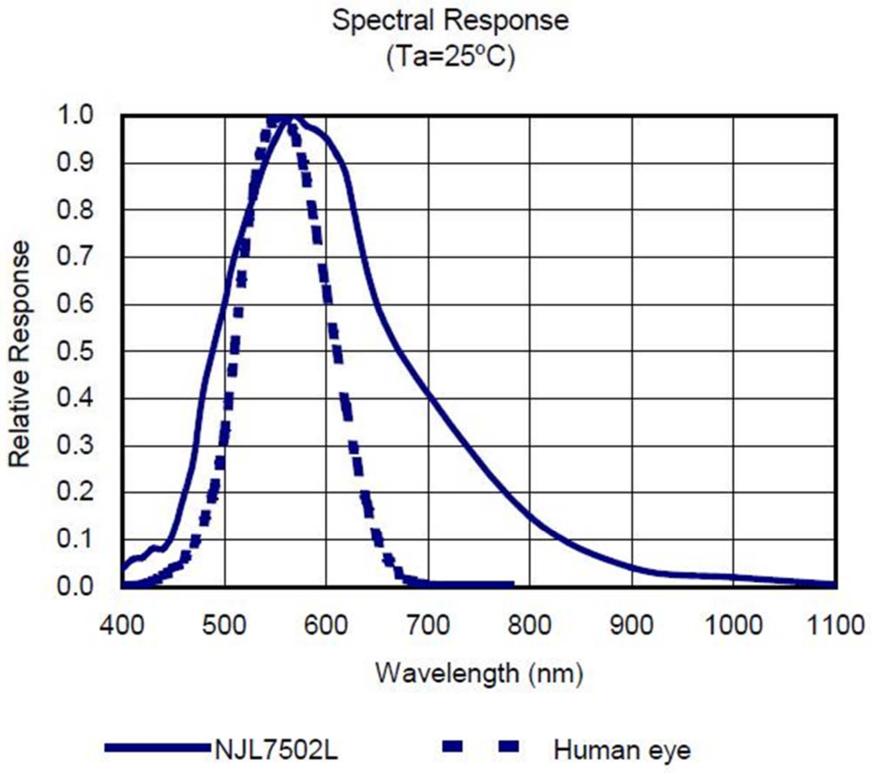
Vout = 100k * Iopt

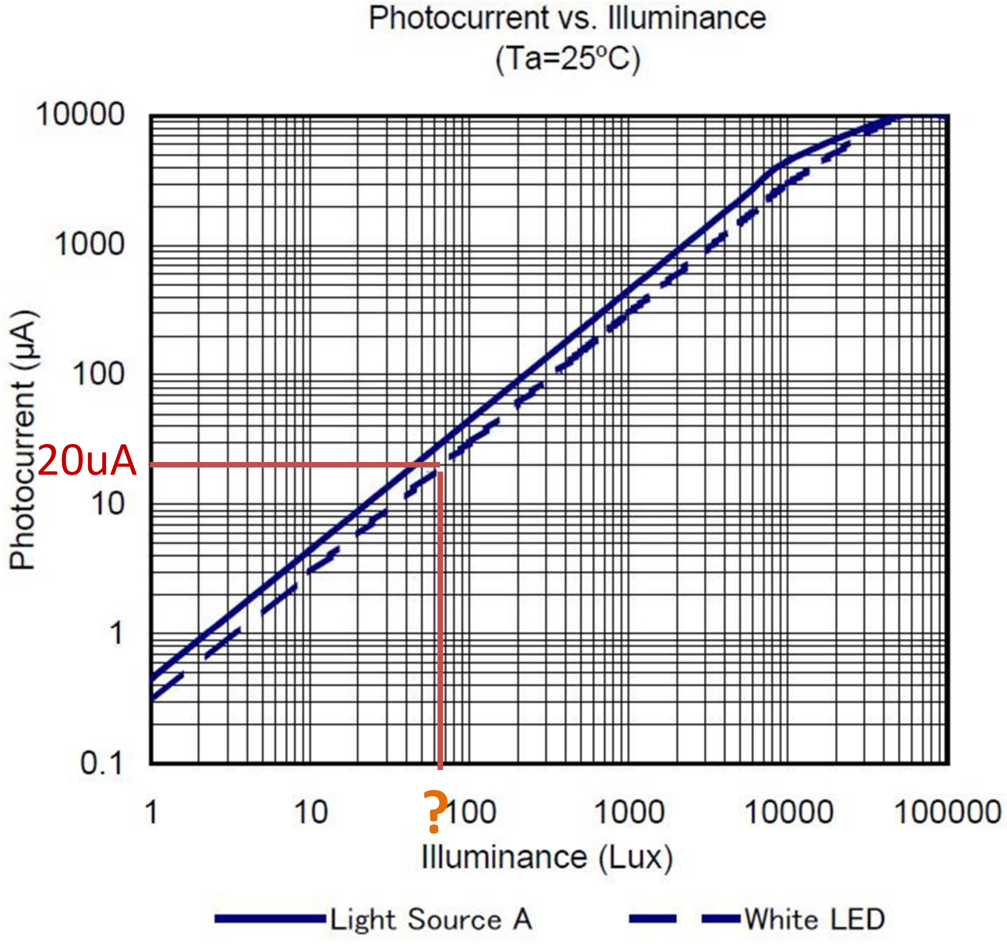
Iopt = Vout * 10 [μV]



Package

3.1 Illuminance





Sample Code

PhotoTrans: Read the output voltage of the temperature sensor

```
int sensorPin = A0;    // select the input pin for the potentiometer

void setup() {
  Serial.begin(57600);
}

void loop() {
  int sensorValue ;  // variable to store the value coming from the sensor

  sensorValue = analogRead(sensorPin);    // read the value from the sensor:
  float volt = sensorValue * 5.0 / 1024;  // converts to the voltage
  Serial.println(volt);
  delay(50);
}
```

Example) in the room

Voltage output is Vout =1 [V], then photo current is Iopt=10[uA]. From the photo current & illuminance graph, get the 60 [lx].

4 Photo Interrupter

Refrective Object Sensor

Letex Technology Corp. LBR127-HDD

Cut-off Visible Wavelenght: λ = 840 nm

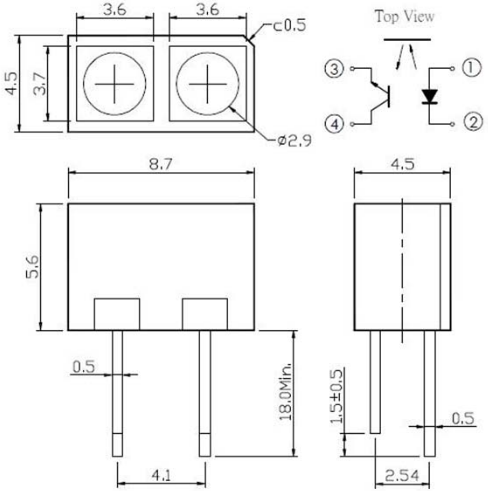
Input:

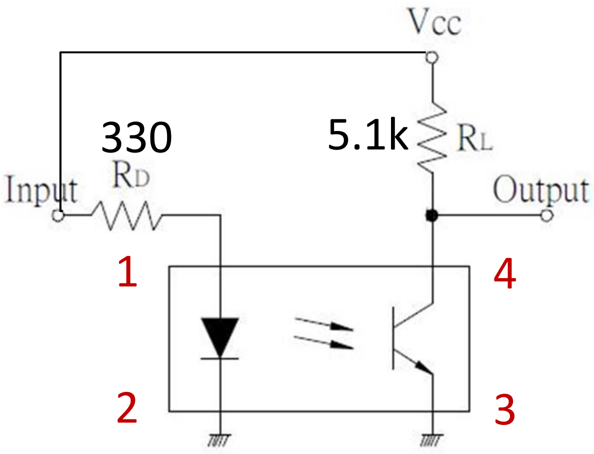
Forward Voltage: Vf = 1.2 – 1.5V

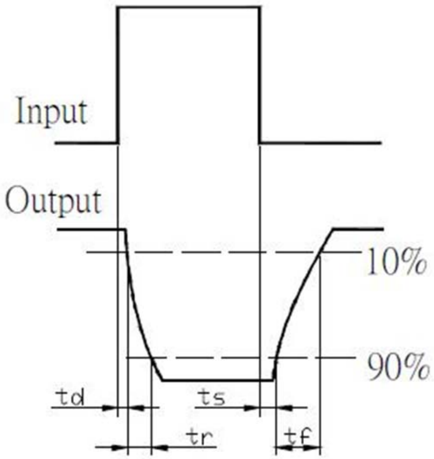
View Angle: 35 Deg. (2θ ½) (If = 20mA)

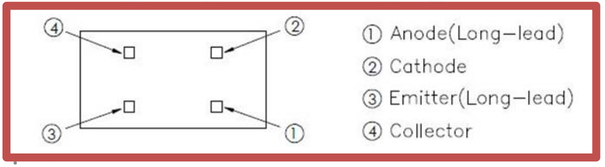
Output:

Voltage (C-E sat.): Vce = 0.4 V (Ic=2mA, Ib=0.1mA)









5 PSD Sensor

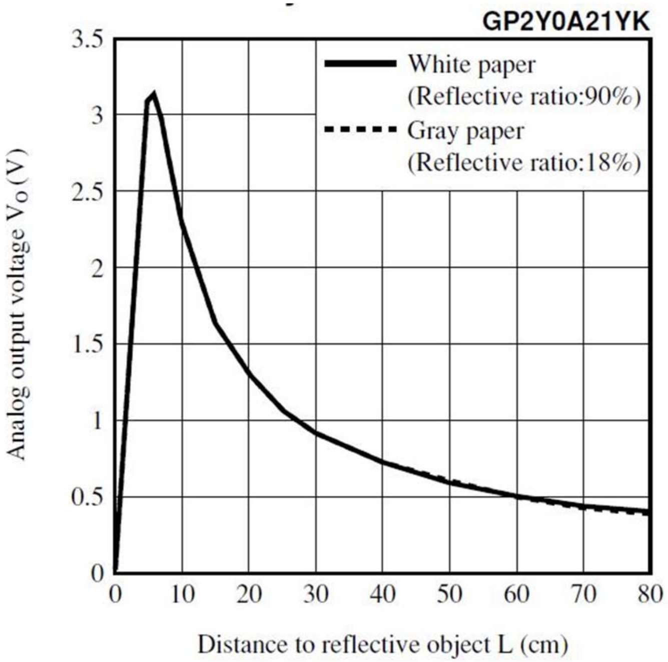
PSD (Position Sensitive Devices) Sensor

Sharp Measuring Sensor GP2Y0A21YK

Output: Analog

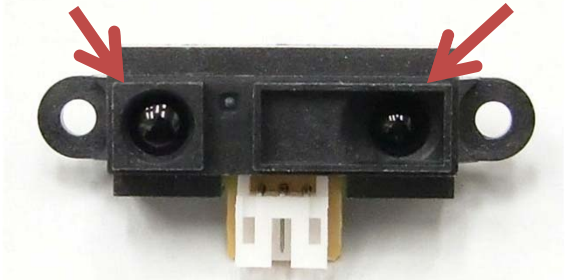
Detecting Distance: 10 cm – 80 cm

Operating Voltage: DC 4.5 V – 5.5 V

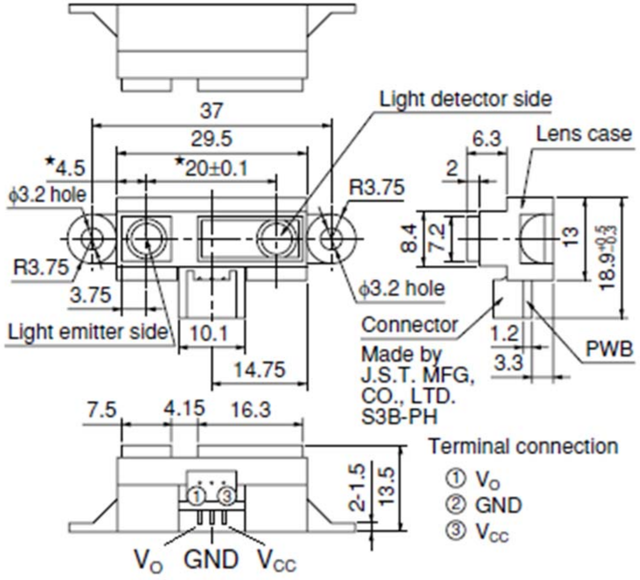


Light Emitter

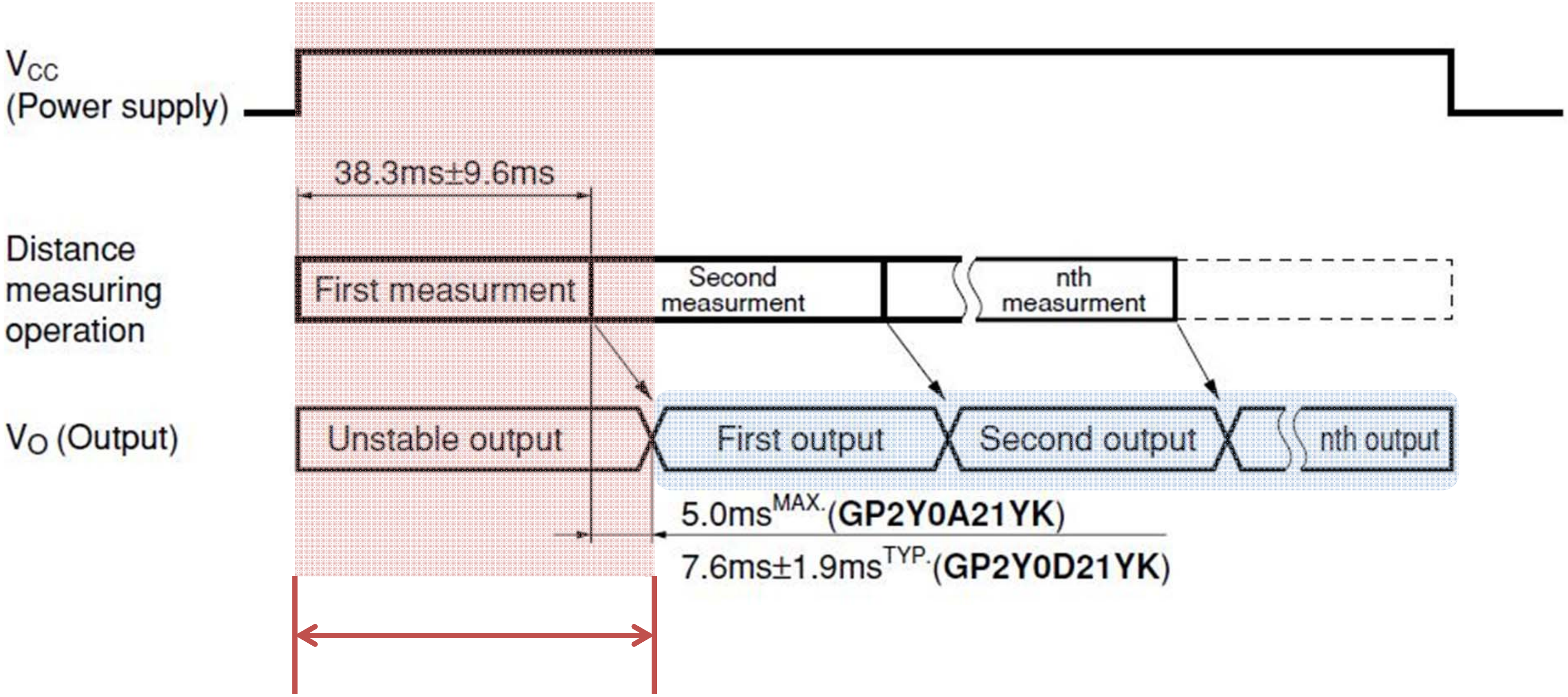
Light Detector



Vo GND Vcc(+5V)



5.1 Timing Chart

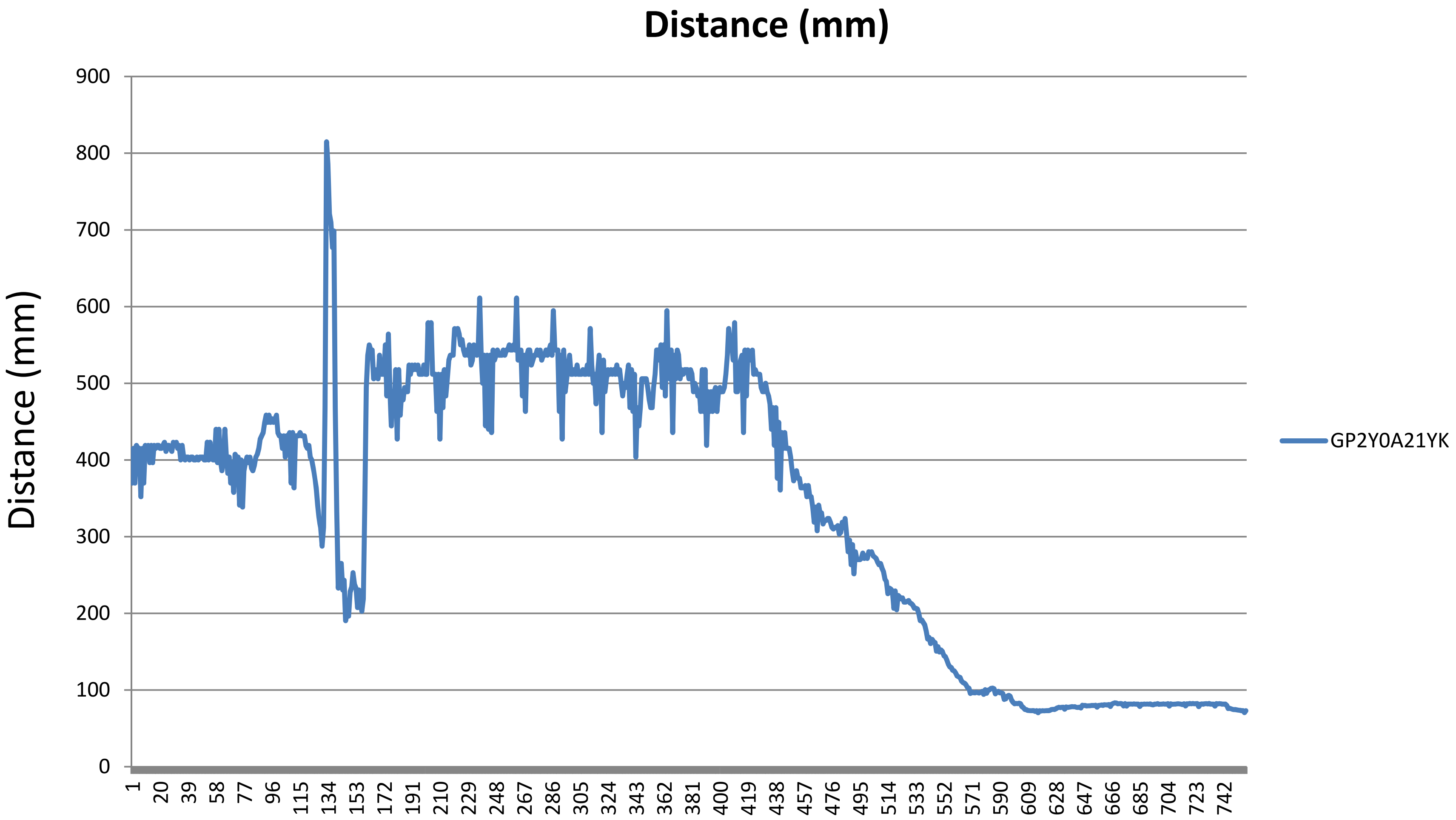


To get a correct output you must wait at least 38.3 ms + 9.6 ms + 5.0 ms = 52.9ms after power up

Sample Code

```
const int analogInPin = A0;  // Analog input pin that the potentiometer is attached to

int sensorValue = 0;        // value read from the pot

void setup() {
  // initialize serial communications at 57600 bps:
  Serial.begin(57600);
  delay(52);
}

void loop() {
  sensorValue = analogRead(analogInPin);

  float dist = 220000 / (sensorValue * 5.0 - 200); // converts to distance (mm)

  Serial.println(dist);
  delay(10);
}
```

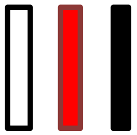
Measurement Result



6 Servo Motor

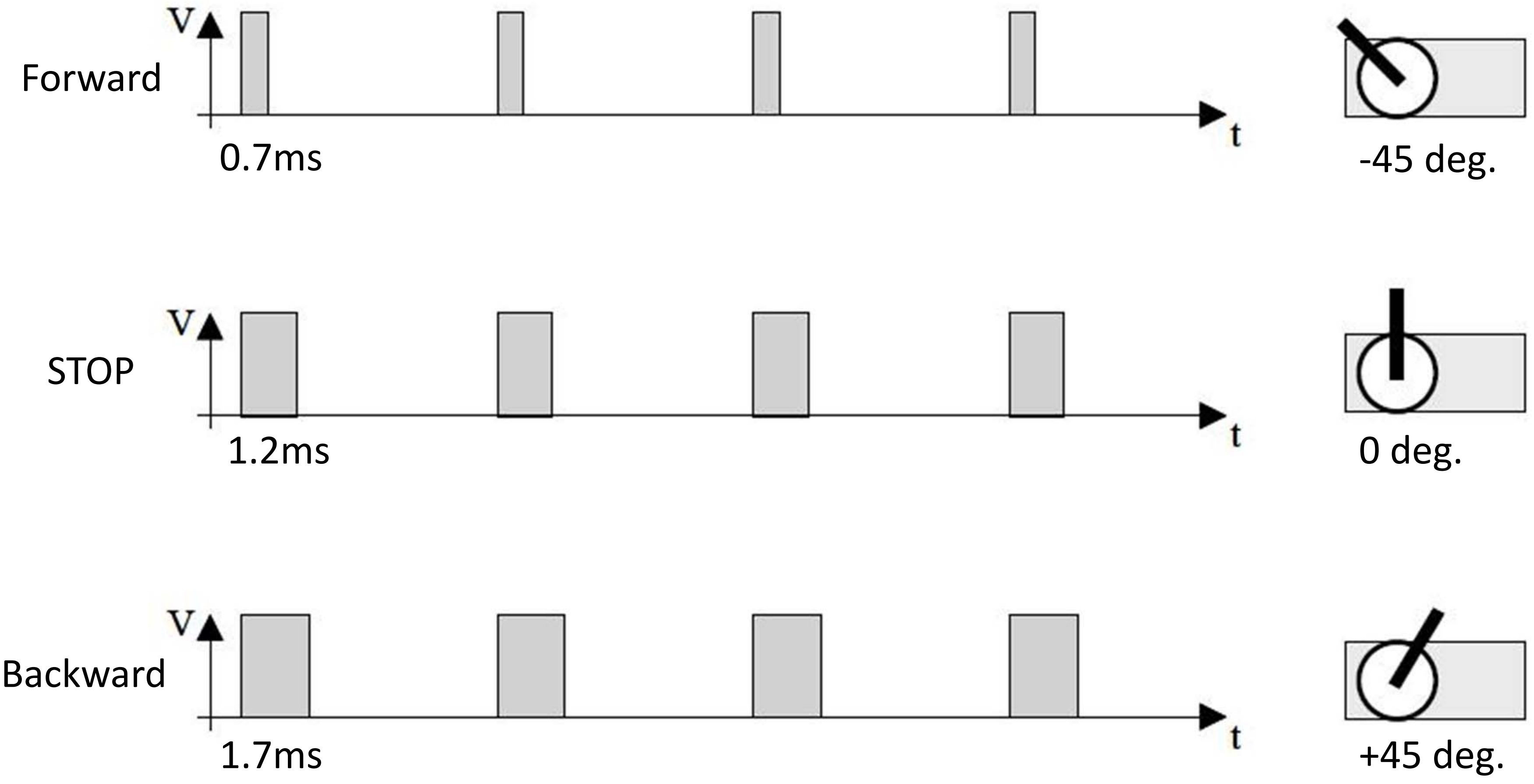
Servo Motor: GWS PICO/STD/F
Weight: 5.4 g
Torque: 0.7 kg
Speed: 0.12 sec / 60 degrees
Size: 22.8 x 9.5 x 16.5 mm

Pin Assignment



Vin +5V GND
Pin9

6.1 Servo Motor



Forward

0.7ms

-45 deg.

STOP

1.2ms

0 deg.

Backward

1.7ms

+45 deg.

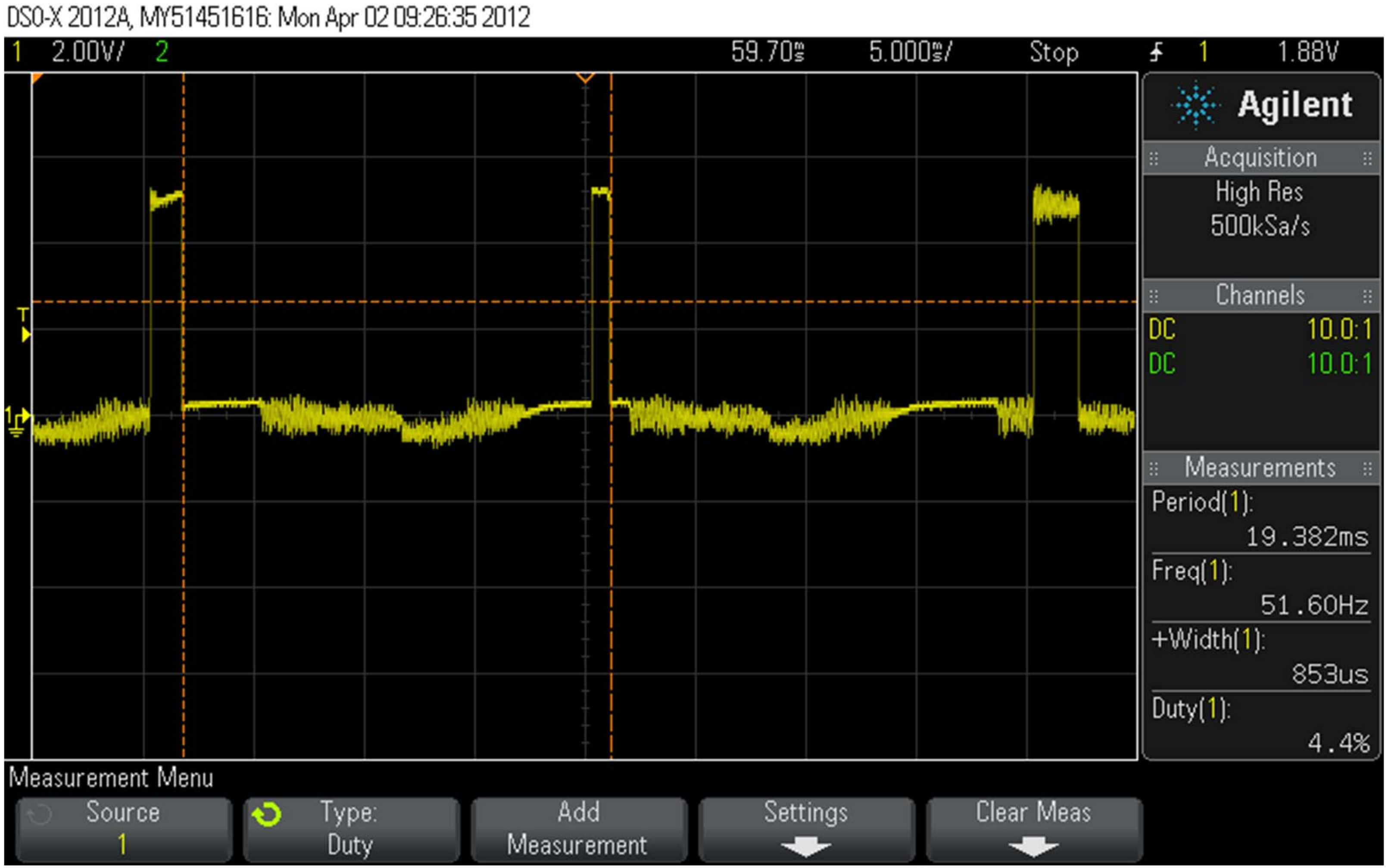
f = 50 Hz (T = 20ms)

Courtesy of Embedded Robotics

Sample Code

```
#include <Servo.h>
Servo myservo;  // create servo object to control a servo
int val;    // variable to set the servo (0--179)
void setup()  {
  myservo.attach(9);  // attaches the servo on pin 9 to the servo object
}
void loop()  {
  val = 90; // center  90 degree
  myservo.write(val);                 // sets the servo position
  delay(15);                        // waits for the servo to get there
  delay(1000);
  val = 30;  // left  -60 degree
  myservo.write(val);                 // sets the servo position according to the scaled value
  delay(15);                        // waits for the servo to get there
  delay(1000);
  val = 150; // right  +60 degree
  myservo.write(val);                 // sets the servo position according to the scaled value
  delay(15);                        // waits for the servo to get there
  delay(1000);
}
```

Measurement Result



DSO-X 2012A, MY51451616: Mon Apr 02 09:26:35 2012

```
1   2.00V/   2          59.70ms    5.000ms/    Stop    1    1.88V
```

Acquisition
High Res
500kSa/s

Channels

DC	10.0:1
DC	10.0:1

Measurements

```
Period(1):
        19.382ms
Freq(1):
         51.60Hz
+Width(1):
           853us
Duty(1):
            4.4%
```

Measurement Menu

7 LCD

LCD SC162B

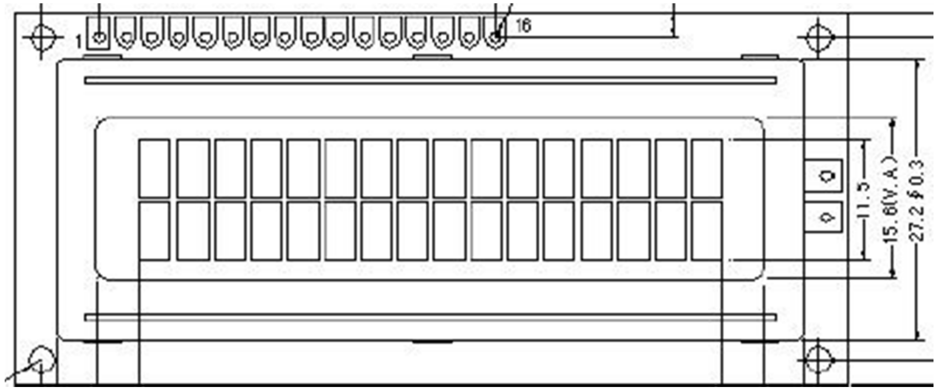
Number of Characters: 16 Character x 2 lines

Drive Method: 1/5 bias, 1 / 16 duty

Operating Voltage: 4.5V – 5.5V

Back Light: LED

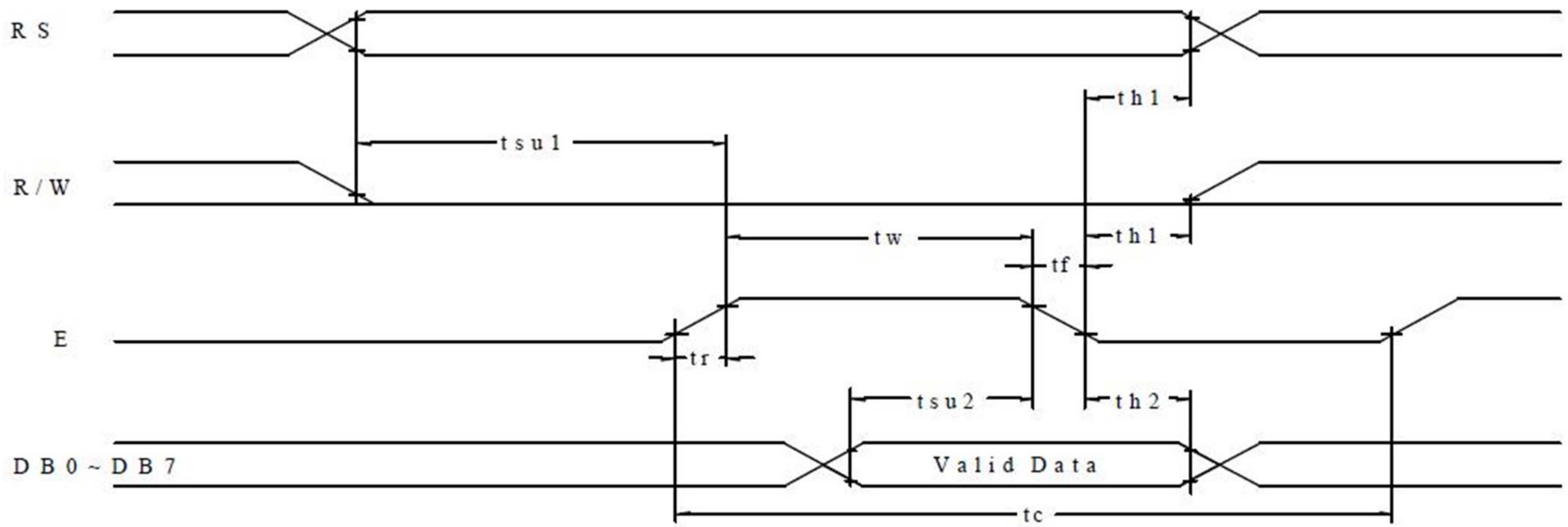
Pin 1 2 3 4 5 6 7 8 9 10 11 12



LiquidCrystal(rs, enable, d4, d5, d6, d7)

Pin assignment	Arduino Pin
1 Vss (GND)	
2 Vdd (+5V)	
4 RS	D12
5 R/W(GND)	
6 E	D11
11 DB4	D5
12 DB5	D4
13 DB6	D3
14 DB7	D2

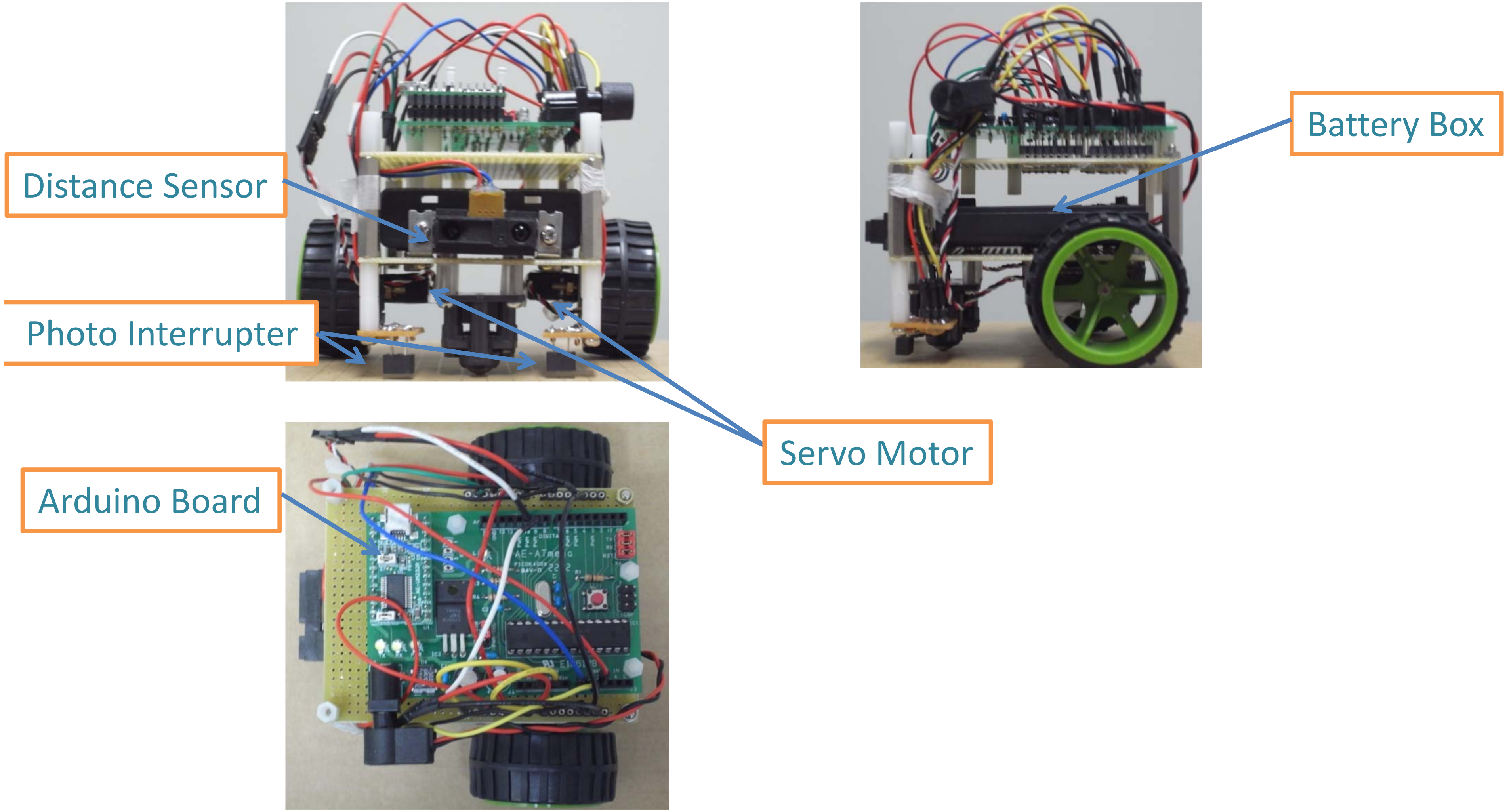
7.1 Timing Chart



Sample Code

```
#include <LiquidCrystal.h>

// initialize the library with the numbers of the interface pins
LiquidCrystal lcd(12, 11, 5, 4, 3, 2); // RS, E, D4, D5, D6, D7

void setup() {
  // set up the LCD's number of columns and rows:
  lcd.begin(16, 2);
  // Print a message to the LCD.
  lcd.print("hello, world!");
}

void loop() {
  // set the cursor to column 0, line 1
  // (note: line 1 is the second row, since counting begins with 0):
  lcd.setCursor(0, 1);
  // print the number of seconds since reset:
  lcd.print(millis()/1000);
}
```

8. Small Robot



Distance Sensor

Photo Interrupter

Arduino Board

Servo Motor

Battery Box