

รหัสวิชา 06016271

Microprocessors

อ. ปานวิทย์ ชุภะนุตติ

Email : panwit@it.kmitl.ac.th

คำอธิบายรายวิชา ตามหน้าหลักสูตรของคณะ

- สถาปัตยกรรมของไมโครโปรเซสเซอร์ ไมโครคอนโทรลเลอร์
- ชุดคำสั่ง
- ชนิดของหน่วยความจำไฟฟ้าและแผนผังวงจร
- การเชื่อมต่อคุปกรณ์อินพุต เอาต์พุต
- การเชื่อมต่อแบบอนุกรมและแบบขนาน
- การเชื่อมต่อแบบซิงโครนัสและอะซิงโครนัส
- การประยุกต์ สำหรับคุปกรณ์อินพุต เอาต์พุตในลักษณะต่างๆ ตลอดจน การเชื่อมต่อระหว่างสัญญาดิจิตอลกับสัญญาณอนาล็อก

ວັດຖຸປະສົງຄໍ

1. ເຮືອນຮູ້ແລະເຂົ້າໃຈໂຄຮສຮ້າງພື້ນສູານແລະສັບປັບຍກຽມກາຍໃນຂອງໄມໂຄຣປິໂປຣເຊີເຊົ່ວແລະໄມໂຄຣຄອນໂທຣລເລອ່ວ
2. ເຮືອນຮູ້ແລະເຂົ້າໃຈກາຈັດສຽບທີ່ມີຄວາມຈຳຂອງໄມໂຄຣຄອນໂທຣລເລອ່ວ
3. ເຮືອນຮູ້ແລະເຂົ້າໃຈວິທີການທຳງານແລະການໃໝ່ງານຮົຈີສເຕອຣຂອງໄມໂຄຣຄອນໂທຣລເລອ່ວ
4. ເຮືອນຮູ້ແລະເຂົ້າໃຈກາທຳງານຂອງຊຸດຄຳສັ່ງຂອງໄມໂຄຣຄອນໂທຣລເລອ່ວ
5. ເຮືອນຮູ້ແລະເຂົ້າໃຈວິທີກາເຂືອນຜັງງານແລະໂປຣແກຣມຄວບຄຸມໄມໂຄຣຄອນໂທຣລເລອ່ວ
6. ເຮືອນຮູ້ແລະເຂົ້າໃຈວິທີກາໃໝ່ງານພອ່ວຍຕົນພຸດແລະເຂາດພຸດ
7. ເຮືອນຮູ້ແລະເຂົ້າໃຈວິທີກາເຫື່ອມຕ່ອກບໍ່ທີ່ມີຄວາມຈຳກາຍນອກ
8. ເຮືອນຮູ້ແລະເຂົ້າໃຈວິທີກາໃໝ່ງານວ່າງຈານນັບ/ຈັບເວລາ
9. ເຮືອນຮູ້ແລະເຂົ້າໃຈວິທີກາຮັບສິ່ງຂໍ້ມູນແບບອນຸກຮມ
10. ເຮືອນຮູ້ແລະເຂົ້າໃຈວິທີກາອີນເຕອຣວັພົດແລະການໃໝ່ງານ
11. ເຮືອນຮູ້ແລະເຂົ້າໃຈວິທີກາໃໝ່ໄມໂຄຣຄອນໂທຣລເລອ່ວຄວບຄຸມອຸປກຣົມຕ່າງໆ
12. ແກີດທັກະນຸ່າໃນກາເຂືອນໂປຣແກຣມຄວບຄຸມຮະບບາງານຕ່າງໆ

ກາງວັດແລະກາປະເມີນຜລ

ວິທີກາປະເມີນຜລຮາຍວິຊາ

ສອບກາລາງການເຮືອນ	25 %
ສອບປາຍການເຮືອນ	25 %
Mini project	25 %
Laboratory	25 %
ຄະແນນຮວມ	100 %

ເວລາເຮືອນໄມ່ນ້ອຍກວ່າ **80 %**

Mini project

- วัตถุประสงค์
 - รวมแรงรวมใจ ช่วยเหลือสังคม
- ตัวอย่าง
 - นาฬิกา Synchronize
 - ระบบปิดเปิดไฟ อัตโนมัติ
 - ระบบรถน้ำด้วยไม้อัตโนมัติ
 - ระบบให้อาหารสัตว์ อัตโนมัติ
 - Remote ควบคุมการปิดเปิดอุปกรณ์ไฟฟ้า



1. Introduction

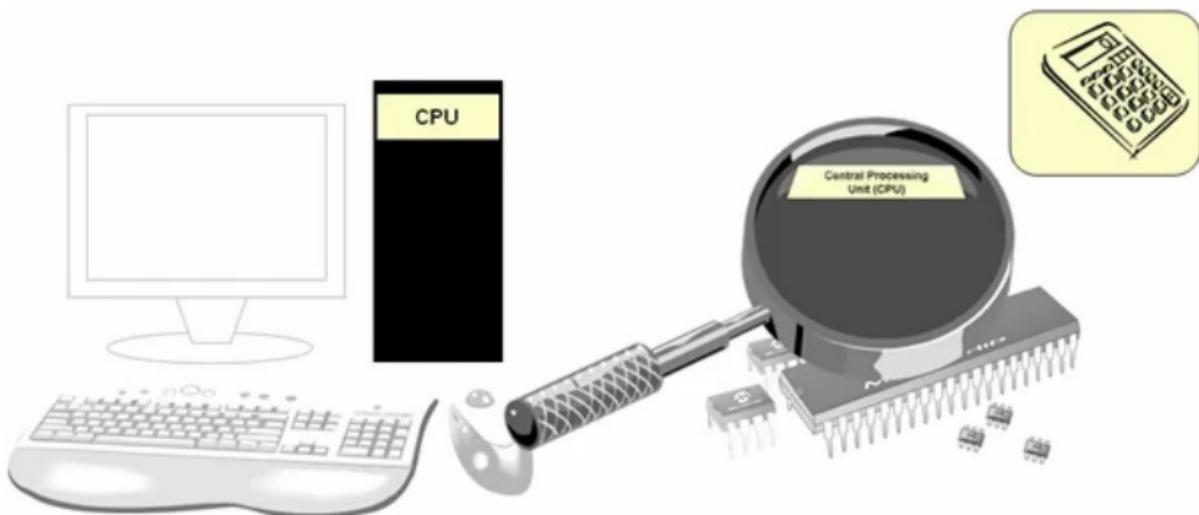


1. What is a Microcontroller?



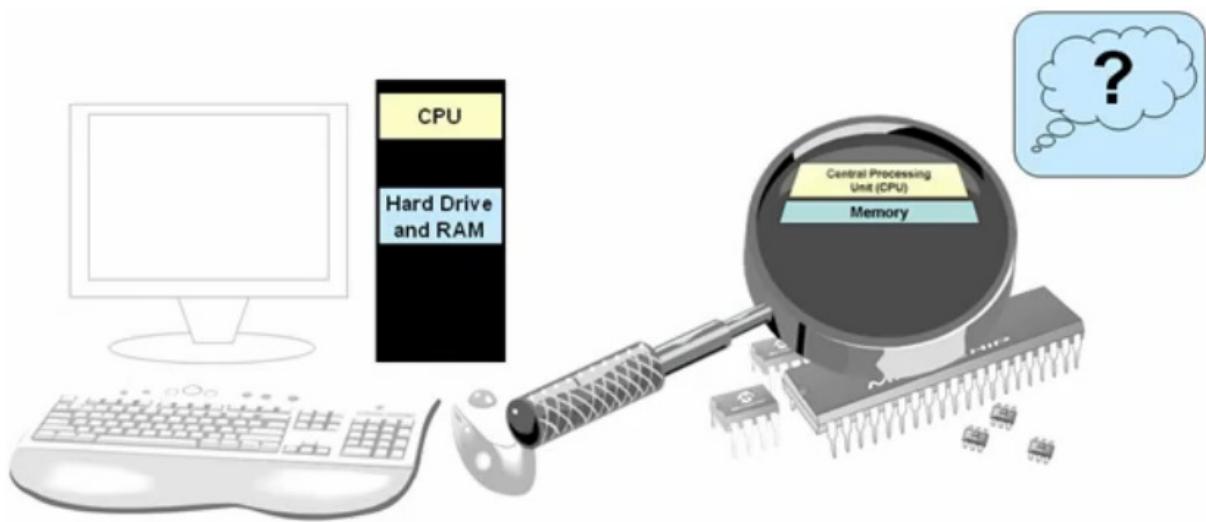
- A microcontroller (**μC**, **uC** or **MCU**) is a single integrated circuit that contains the four major parts of every computer system
- A device optimized for control applications

What is in a Microcontroller?



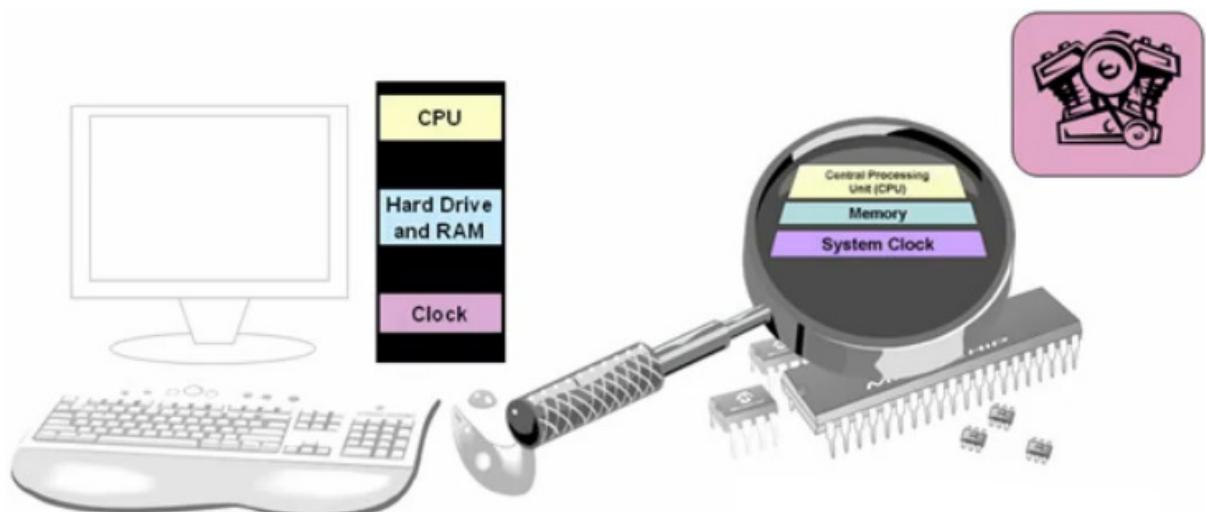
- All computers have a CPU (central processing unit) that executes programs

What is in a Microcontroller?



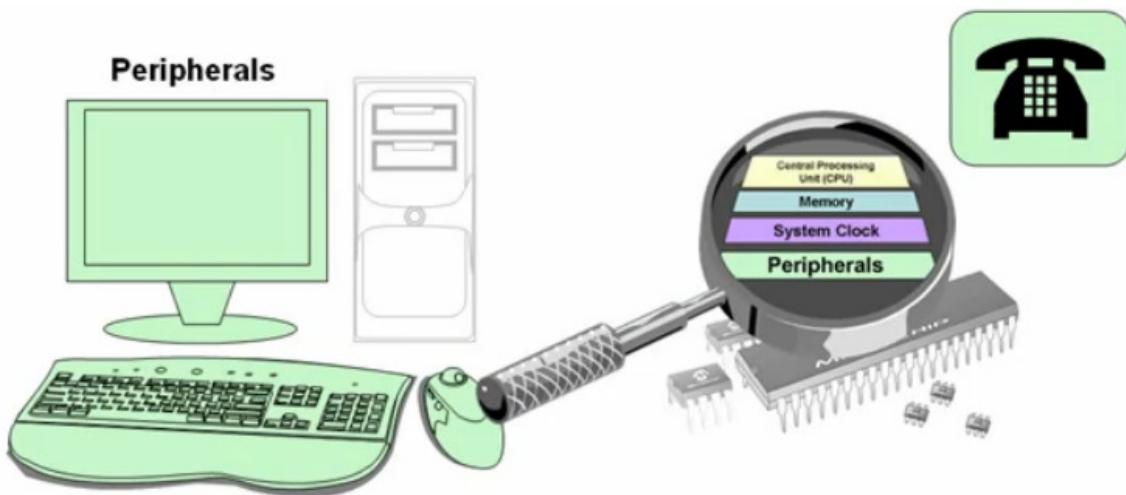
- A computer has some type of memory where it can store variables and instructions or steps
- Memory may be RAM, ROM, (E)EPROM, Flash

What is in a Microcontroller?



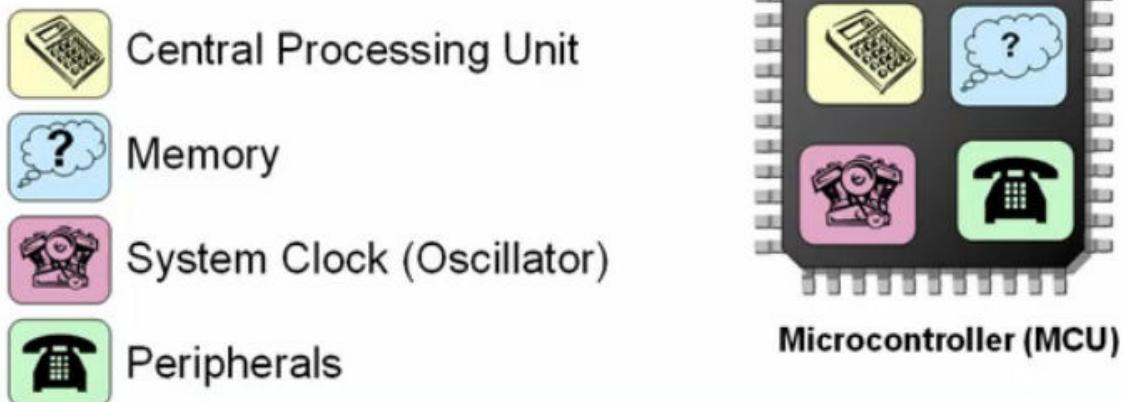
- Computers have a clock or oscillator that determine the speed of program execution

What is in a Microcontroller?



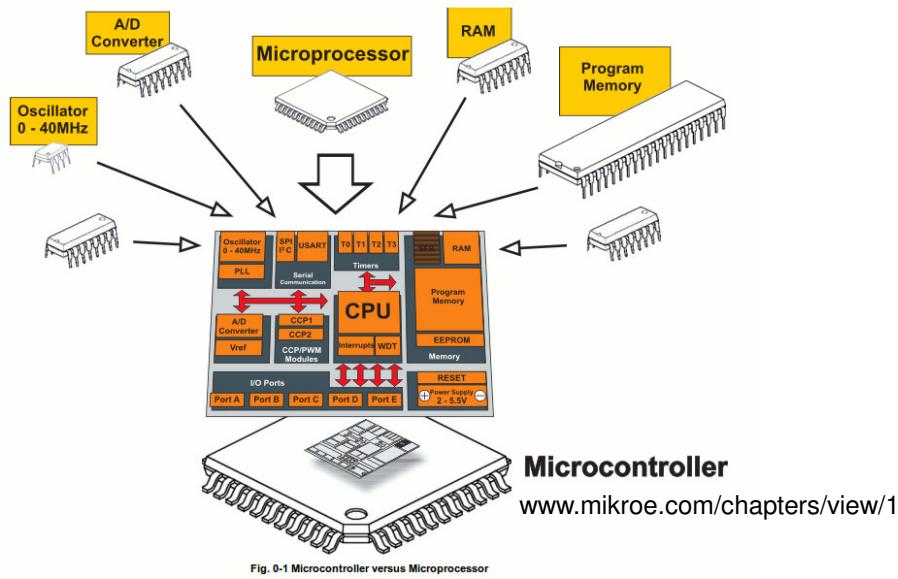
- Computers have various input and output (I/O) capabilities to support various peripherals.

What is in a Microcontroller?



- Microcontroller range from 8 – 40+ pin packages
- Extra pins provide analog and digital I/O options
- 8, 16, 32 & 64 bit instructions & data controllers

What is a Microcontroller



- A small computer on a single chip
 - containing a processor, memory, and input/output
- Typically "**embedded**" inside some device that they control
- A microcontroller is often small and low cost

Where To Find Microcontrollers?



- Microcontroller are found everywhere!
- uC's are considered embedded systems

2. สถาปัตยกรรมของไมโครคอนโทรลเลอร์

- **MCS-51 (8-bit)** ออกแบบโดย intel
- **MCS-96 (16-bit)** ออกแบบโดย intel
- **PIC** ออกแบบโดย Microchip Technology
- **AVR** ออกแบบโดย Atmel
- **ARM** ออกแบบโดย ARM Holdings
- **68HC11** ออกแบบโดย Motorola
- **Rabbit 2000** ออกแบบโดย Rabbit Semiconductor



15

- ในอดีต ไมโครคอนโทรลเลอร์ที่นิยมใช้งาน คือตระกูล **MCS-51** และ ในปัจจุบันความต้องการ ความสามารถในการประมวลผลมีเพิ่มขึ้น ทำให้มีสถาปัตยกรรมอื่นๆ ที่มีประสิทธิภาพสูงกว่าเริ่มได้รับความนิยมในการใช้งานขึ้นมาแทน
- ในวิชานี้จะสอนให้นักศึกษาได้เรียนรู้สถาปัตยกรรม **AVR** เนื่องจากมีประสิทธิภาพสูง และมีราคาถูกกว่าสถาปัตยกรรมอื่นๆ ในราคาก็ใกล้เคียงกัน นอกจากราคาที่ยังมีข้อดีคือ เป็น **Open Source** ทำให้มีข้อมูลต่างๆ และ **Library** ต่างๆ เป็นจำนวนมาก และสามารถพัฒนาระบบที่ต้องใช้ภาษาระดับต่ำ คือภาษา **Assembly** และ ภาษาระดับสูง เช่น ภาษา **C** เป็นต้น

16

3. สถาปัตยกรรมของ AVR

- แบ่งออกเป็น 2 ตระกูลคือ 8-bit AVR และ 32-bit AVR
- ในวิชานี้จะกล่าวถึงสถาปัตยกรรมของ AVR ขนาด 8 บิตเท่านั้น
- สถาปัตยกรรม AVR ออกแบบโดย ATME^L เมื่อปี 1996 เป็นชิปปุ่มแบบ RISC (Reduced Instruction Set Computer)
- มีสถาปัตยกรรมการต่อหน่วยความจำแบบ Harvard ซึ่งแยกหน่วยความจำโปรแกรมและหน่วยความจำข้อมูล ออกจากกันโดยเด็ดขาด
- ใช้หน่วยความจำแบบ Flash สำหรับเป็นหน่วยความจำโปรแกรม
- ใช้หน่วยความจำแบบ SRAM สำหรับหน่วยความจำข้อมูล
- นอกจากนี้ยังมีหน่วยความจำแบบ EEPROM ซึ่งสามารถเก็บข้อมูลเอาไว้ได้โดยไม่จำเป็นต้องมีไฟเลี้ยง

17

4. ประเภทการใช้งาน AVR ขนาด 8 บิต

- tinyAVR** เป็นชิปปุ่มในรุ่นเล็ก ซึ่งต้องการความลึกกะทัดรัดของวงจรโดยเหมาะสมกับระบบควบคุมขนาดเล็กๆ ที่ต้องการหน่วยความจำและวงจรสนับสนุนไม่มากนัก ชิปปุ่มในรุ่นนี้จะมีราคาถูกกว่ากลุ่มอื่น



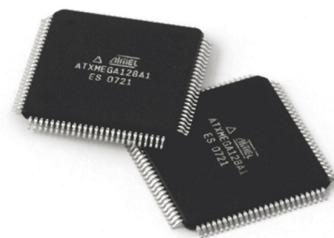
- megaAVR** จะมีชื่ออีกอย่างว่า ATmega โดยมีวงจรสนับสนุนภายในเพิ่มเติมตลอดจนเพิ่มขนาดของหน่วยความจำให้ใช้งานมากกว่าตระกูล Tiny เหมาะสมกับงานควบคุมทั่วไป



(C) MWTech, 2008

18

- **XMEGA** เพิ่มความละเอียดของวงจร A/D จากปกติมีความละเอียด 10 บิตในรุ่นเล็กกว่าเป็น 12 บิต และวงจร DMA controller ซึ่งช่วยลดภาระของซีพียูในการควบคุมการรับส่งข้อมูลระหว่าง อุปกรณ์ I/O กับหน่วยความจำ



- **FPSLIC (AVR core with FPGA)** สำหรับงานที่ต้องการควบคุมที่ต้องการความยืดหยุ่นในขั้นตอนการอุปกรณ์แบบและพัฒนา โดยผู้ออกแบบสามารถออกแบบวงจรในระดับฮาร์ดแวร์เพิ่มเติมด้วยภาษาบรรยายฮาร์ดแวร์ (HDL: Hardware Description Language) เช่น ภาษา VHDL หรือภาษา Verilog และให้วงจรที่ออกแบบทำงานร่วมกับซีพียู AVR core

19

- **Application Specific AVR** เป็นซีพียูที่ออกแบบมาโดยเพิ่มวงจรควบคุมเฉพาะด้านเข้าไปซึ่งไม่พบในซีพียูกลุ่มนี้ เช่นวงจร USB controller หรือวงจร CAN bus เป็นต้น

AVR มีให้เลือกใช้งานหลายเบอร์ แต่ละเบอร์จะมีขนาด ราคา ความสามารถ และขนาด หน่วยความจำตลอดจนถึงวงจรสนับสนุนภายในที่แตกต่างกันออกไป ในวิชา Microprocessor นี้ จะเลือกใช้ Platform Arduino ซึ่งใช้ซีพียูรุ่น **ATmega328**

5. ATmega328

- หน่วยความจำโปรแกรมแบบ FLASH ขนาด 32 Kbyte
- หน่วยความจำข้อมูลแบบ SRAM ขนาด 2 Kbyte
- หน่วยความจำข้อมูลแบบ EEPROM ขนาด 1 Kbyte
- สนับสนุนการเชื่อมต่อแบบ I²C bus
- พอร์ตอินพุตเอาต์พุตจำนวน 22 ports
- วงจรสื่อสารอนุกรม
- วงจรนับ/จับเวลาขนาด 8 บิต จำนวน 2 ตัว และขนาด 16 บิตจำนวน 1 ตัว
- สนับสนุนช่องสัญญาณสำหรับสร้าง PWM จำนวน 6 ช่องสัญญาณ
- วงจรแปลงอนาลอกเป็นดิจิตอลขนาด 10 บิตในตัว จำนวน 8 ช่อง
- ทำงานได้ตั้งแต่ย่านแรงดัน 1.8-5.5 Volts
- ความถี่ใช้งานสูงสุด 20 MHz

21

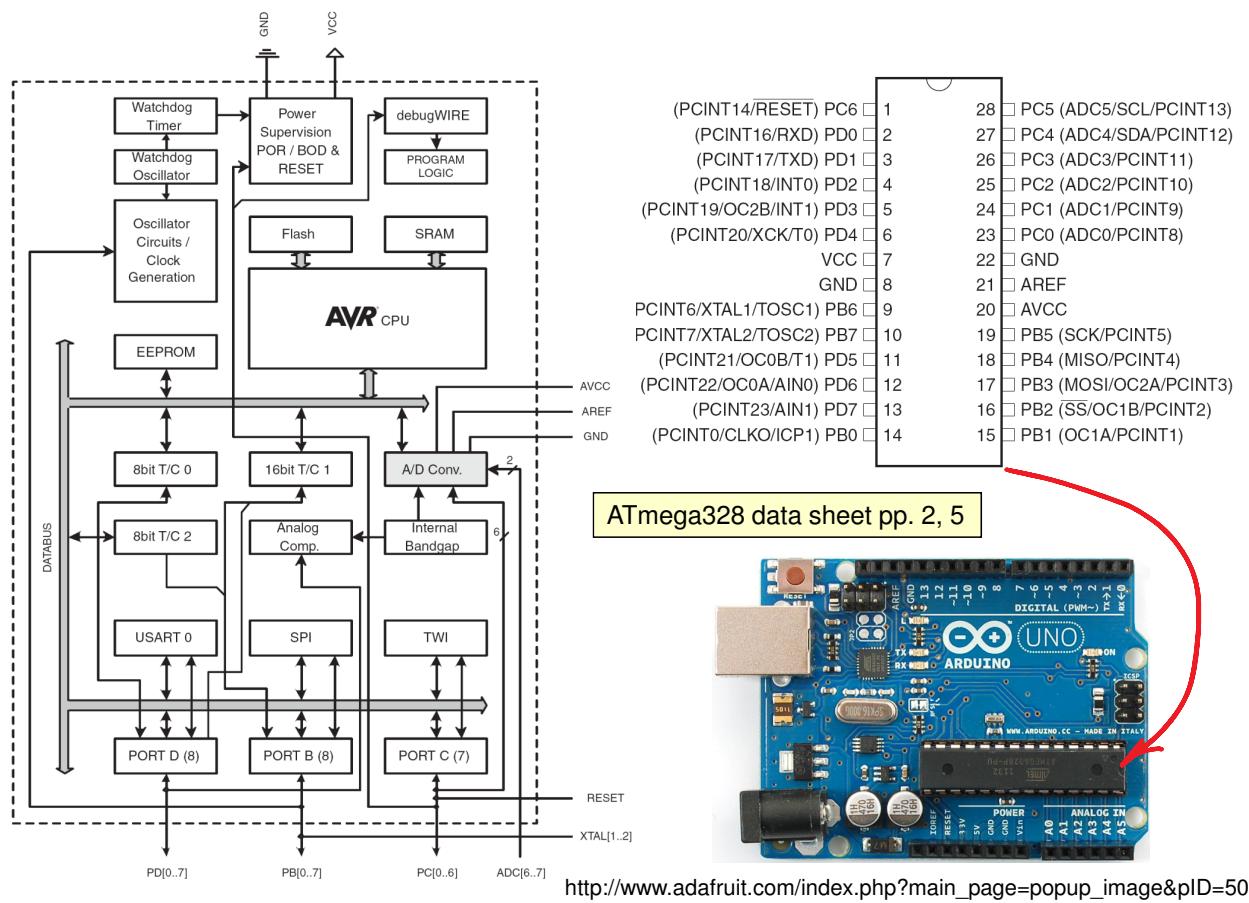
รายละเอียด ขา ต่างๆ ของ ATmega168

ATMega328P and Arduino Uno Pin Mapping

Arduino function		Arduino function
reset	(PCINT14/RESET) PC6	1
digital pin 0 (RX)	(PCINT16/RXD) PD0	2
digital pin 1 (TX)	(PCINT17/TXD) PD1	3
digital pin 2	(PCINT18/INT0) PD2	4
digital pin 3 (PWM)	(PCINT19/OC2B/INT1) PD3	5
digital pin 4	(PCINT20/XCK/T0) PD4	6
VCC	VCC	7
GND	GND	8
crystal	(PCINT6/XTAL1/TOSC1) PB6	9
crystal	(PCINT7/XTAL2/TOSC2) PB7	10
digital pin 5 (PWM)	(PCINT21/OC0B/T1) PD5	11
digital pin 6 (PWM)	(PCINT22/OC0A/AIN0) PD6	12
digital pin 7	(PCINT23/AIN1) PD7	13
digital pin 8	(PCINT0/CLK0/ICP1) PB0	14
		28
		27
		26
		25
		24
		23
		22
		21
		20
		19
		18
		17
		16
		15
		PC5 (ADC5/SCL/PCINT13)
		PC4 (ADC4/SDA/PCINT12)
		PC3 (ADC3/PCINT11)
		PC2 (ADC2/PCINT10)
		PC1 (ADC1/PCINT9)
		PC0 (ADC0/PCINT8)
		GND
		AREF
		AVCC
		PB5 (SCK/PCINT5)
		PB4 (MISO/PCINT4)
		PB3 (MOSI/OC2A/PCINT3)
		PB2 (SS/OC1B/PCINT2)
		PB1 (OC1A/PCINT1)
		analog input 5
		analog input 4
		analog input 3
		analog input 2
		analog input 1
		analog input 0
		GND
		analog reference
		VCC
		digital pin 13
		digital pin 12
		digital pin 11(PWM)
		digital pin 10 (PWM)
		digital pin 9 (PWM)

Digital Pins 11,12 & 13 are used by the ICSP header for MOSI, MISO, SCK connections (Atmega168 pins 17,18 & 19). Avoid low-impedance loads on these pins when using the ICSP header.

ATmega328 Internal Architecture



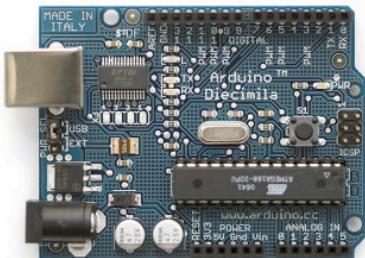
6. What is Arduino ? [1]

- Arduino เป็นโครงการพัฒนาไมโครคอนโทรลเลอร์ของตระกูล AVR แบบ Open Source
- Arduino เป็นชื่อเรียกของ **platform** ซึ่งประกอบไปด้วย
 - Development Board ที่ใช้ microcontroller ตระกูล AVR
 - IDE เพื่อใช้ในการพัฒนาโปรแกรม โดย IDE พัฒนาโดยภาษา Java สามารถให้สามารถทำงานได้ทุก OS เช่น Linux, MAC OS, Windows และ IDE มี library มาตรฐาน ในการอ้างอิงกับบอร์ด Arduino จำนวนมาก ทำให้สะดวกในการพัฒนาโปรแกรม
- โดย microcontroller ตระกูล AVR ที่นำมาใช้นั้น จะต้องมีการติดตั้ง **Firmware** ไว้แล้ว โดยหน้าที่หลักของ Firmware คือ การรับโปรแกรมที่ Compile แล้ว จาก IDE มาเขียนไว้ที่ตัวมันเอง เพื่อทำงานตามโปรแกรมที่เขียน

What is Arduino ? [2]

The word “Arduino” can mean 3 things

A physical piece of hardware



A programming environment



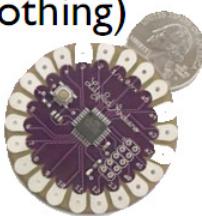
A community & philosophy



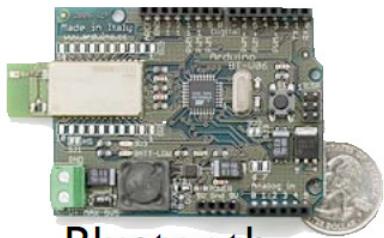
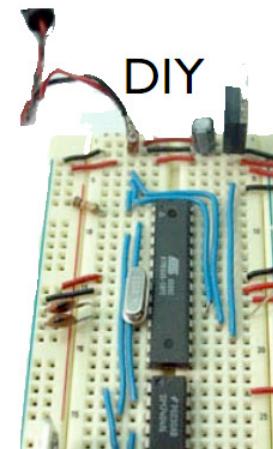
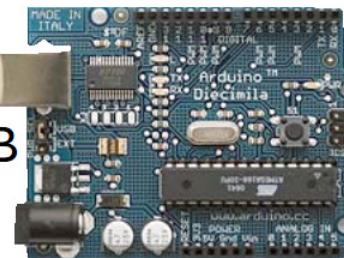
25

6.1 Arduino Hardware Variety

LilyPad
(for clothing)

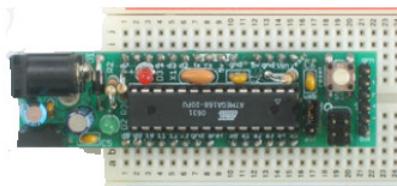


USB



Bluetooth

Boarduino Kit



“Stamp”-sized



many different variations to suite your needs

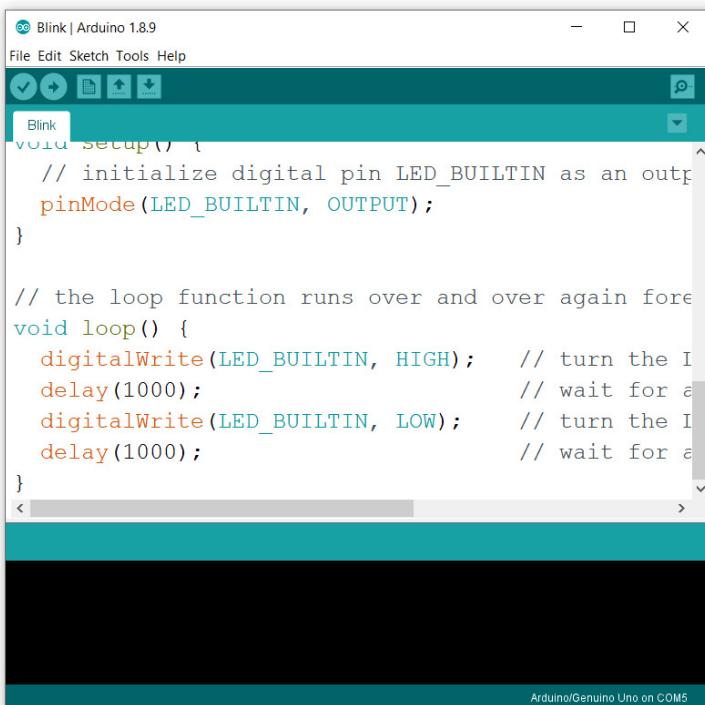
Openness has its advantages, many different varieties.

Anyone can build an Arduino work-alike in any form-factor they want.

Product images from Sparkfun.com and Adafruit.com

26

6.2. Arduino IDE



- Like a text editor
- View/write/edit sketches
- But then you program them into hardware

27

6.3 ข้อดี-ข้อเสีย Arduino

ข้อดี

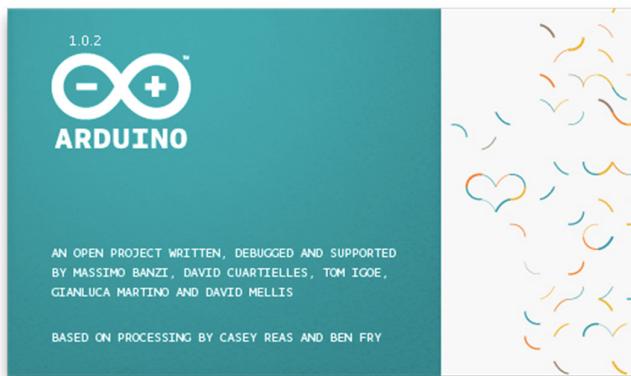
- พัฒนาโปรแกรมด้วยภาษา C++ จึงสามารถใช้คุณสมบัติ OOP ได้
- ไม่ต้องใช้อุปกรณ์ ISP ในการโหลดโปรแกรมไปยัง microcontroller
- โครงสร้างการเขียนเข้าใจง่าย ไม่ซับซ้อน
- ต้นทุนมีราคาถูก
- เป็นโครงการ opensource จึงทำให้มีผู้ให้ความสนใจมาก และมี Library ให้ใช้งานเป็นจำนวนมาก

ข้อเสีย

- เนื่องจากพัฒนาโปรแกรมด้วยภาษา C++ จึงใช้ทรัพยากรบากพอสมควร

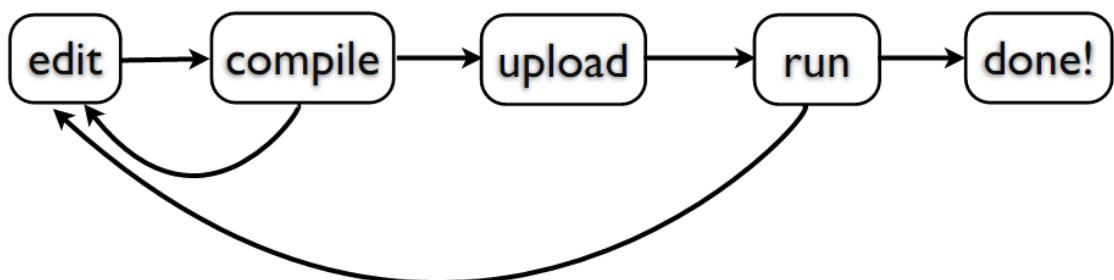
28

2. Arduino Software and Instruction set



1. ขั้นตอนการพัฒนาโปรแกรมด้วย Arduino (Development Circle)

- Make as many changes as you want
- Not like most web programming: edit → run
- Edit → compile → upload → run

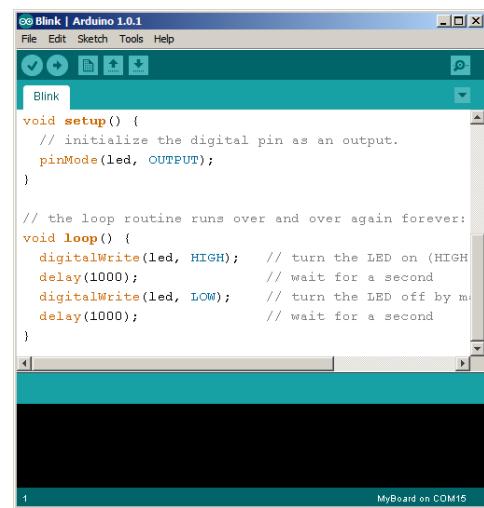


2. Arduino IDE

The Arduino programming platform was designed in JAVA to help newcomers become familiar with programming. The language used to write code is C/C++ .

Each Arduino program is called a **SKETCH** and has two required functions, called ROUTINES.

void setup (){}, void loop (){}



3. Arduino Program (Sketch) Structure

Declare variables at top

- Initialize
 - **setup()** – run ONCE at beginning, set pins
- Running
 - **loop()** – run repeatedly, after **setup()**

All of the code within the curly braces will be run again, and again, until the power is removed.

Example of a bare minimum program:

```
void setup()
{
}

void loop()
{
}
```

4. Programming - Syntax

Similar to C, the formatting requirement is the same.

//
/* */
{ }
;

- Single line comment
- Multiline comment
- used to define a block of code that starts and ends.
- used to define the end of a line of code.

5. Programming Concepts: Variables

```
ProtosnapProMiniExample2 $  
  
// Comments go here  
// Written by: Joesephine Jones  
// Date: April 12, 2013  
  
int sensorValue;  
int ledPin;  
  
void setup()  
{  
    // put your setup code here, to run once:  
    int setupVariable;  
  
}  
  
void loop()  
{  
    // put your main code here, to run repeatedly:  
    int loopScopeVariable  
}
```

Variable Scope

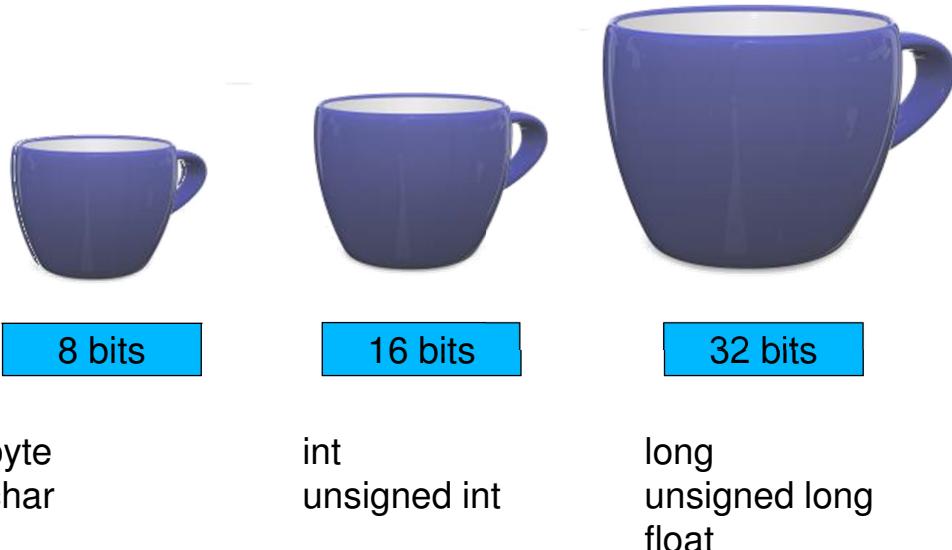
• Global

•---

• Function-level

Variable Types

- **Variable Types:**



Variables

TYPE	NAME	VALUE	
int	number	→ 1	Stored only Integer
int	sum	→ 500500	Stored only Integer
double	radius	→ 5.5	Stored only floating-point number
double	area	→ 95.0334	Stored only floating-point number
String	greeting	→ Hello	Stored only texts
String	statusMsg	→ Game Over	Stored only texts

A *variable* has a **name**, stores a **value** of the declared **type**.

Format of variables

All variables have to be **declared** before they are used.

Declaring a variable means **defining its type**, and optionally, setting an initial value (initializing the variable).

For example,

```
int inputVariable = 0;
```

declares that variable `inputVariable` is of type `int`, and that its initial value is zero.

37

Format of variables

1. **char** – a data type that takes up **1 byte** of memory that stores a character value. Character literals are written in single quotes, like this: 'A'.
2. **byte** – a byte stores an **8-bit** unsigned number, from **0 to 255**.
3. **int** – integers are your primary data type for number storage, and store a **2 byte** value. This yields a range of **-32,768 to 32,767**.
4. **unsigned int** – unsigned integers are the same as integers in that they store a **2 byte** value. Instead of storing negative numbers however they only store positive values, yielding a useful range of **0 to 65,535**.

38

Format of variables

5. **long** – long variables are extended size variables for number storage, and store 32 bits (**4 bytes**), from **-2,147,483,648** to **2,147,483,647**.
6. **unsigned long** – unsigned long variables are extended size variables for number storage, and store 32 bits (**4 bytes**). Unlike standard longs unsigned longs won't store negative numbers, making their range from **0 to 4,294,967,295**

39

Format of variables

7. **Float** – datatype for floating-point numbers, a number that has a decimal point. They are often used to approximate analog and continuous values because they have greater resolution than integers. Floating-point numbers can be as large as **3.4028235E+38** and as low as **-3.4028235E+38**. They are stored as 32 bits (**4 bytes**) of information.
8. **double** – double precision floating point number. Occupies **4 bytes**. The double implementation on the Arduino is currently exactly the same as the float, with no gain in precision.

40

Format of variables

Example of variables:

```
char myChar = 'A';  
char myChar = 65; // both are equivalent  
  
byte b = B10010; // "B" is the binary  
                  formatter (B10010 = 18  
                  decimal)  
  
int ledPin = 13;  
  
unsigned int ledPin = 13;
```

41

6. Arduino data types

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

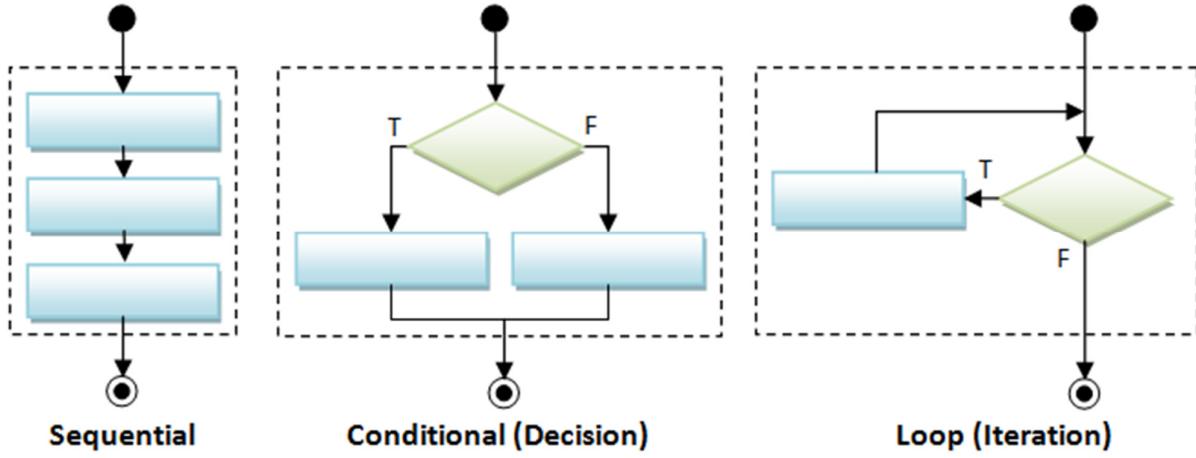
7. Arithmetic Operators

Math	Description
=	Assignment : makes something equal to something
%	Modulo : this gives the remainder when one number is divided by another
+	Addition
-	Subtraction
*	Multiplication
/	Division

8. Comparison Operators

Comparison	Description
() == ()	is equal?
() != ()	is not equal?
() > ()	greater than
() >= ()	greater than or equal
() < ()	less than
() <= ()	less than or equal

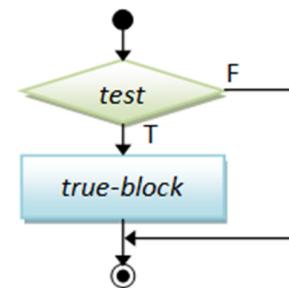
9. Flow Control



Condition Flow Control

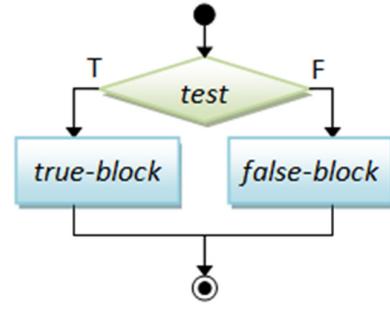
1. if- then

```
if(expression)
{
    True Block;
}
```



2. if- then-else

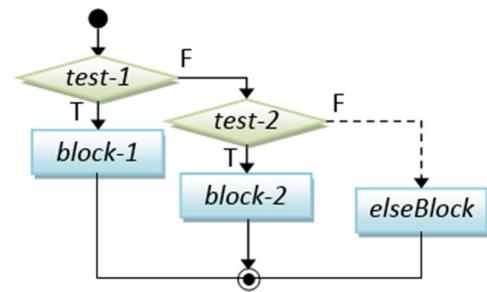
```
if(expression)
{
    True Block;
} else
{
    False Block;
}
```



10. Condition Flow Control

3. nested-if

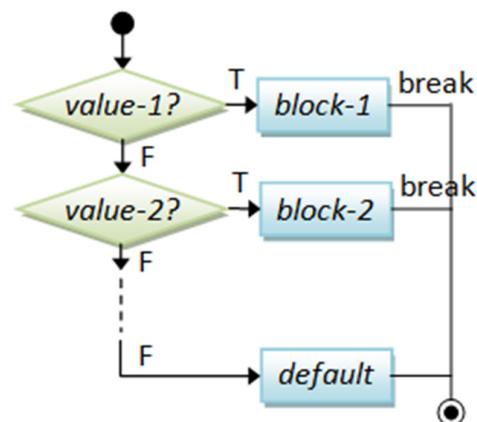
```
if(expression 1)
{
    True Block-1;
} else if(expression 2)
{
    True Block-2;
} else if(expression 3)
{
    ...
} else
{
    else Block
}
```



Condition Flow Control

4. switch-case

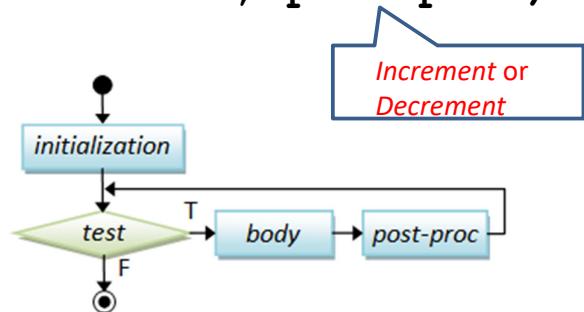
```
switch(selector) {
    case value-1:
        Block-1;
        break;
    case value-2:
        Block-2;
        break;
    case value-3:
        Block-3;
        break;
    .....
    default
        Default Block
        break;
}
```



11. Loop Flow Control

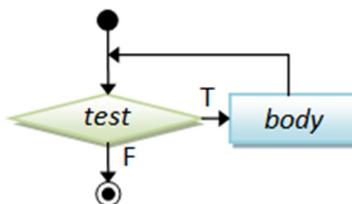
1. for

```
for (initial; test-condition; post-proc)
{
    body;
}
```



2. while

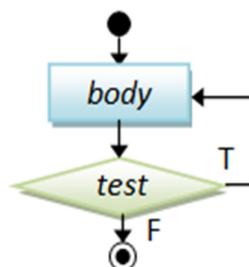
```
while(expression)
{
    body;
}
```



Loop Flow Control

3. do..while

```
do
{
    Body;
}
while (expression);
```



12. Arduino Function

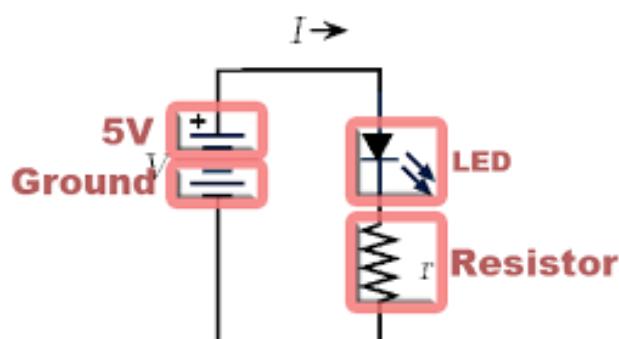
Language is standard C (but made easy)

core function

Digital I/O	Advanced I/O	Math	Random Numbers
<ul style="list-style-type: none">pinMode()digitalWrite()digitalRead()	<ul style="list-style-type: none">tone()noTone()shiftOut()shiftIn()pulseIn()	<ul style="list-style-type: none">min()max()abs()constrain()map()pow()sqrt()	<ul style="list-style-type: none">randomSeed()random()
Analog I/O	Time	Trigonometry	
<ul style="list-style-type: none">analogReference()analogRead()analogWrite() - PWM	<ul style="list-style-type: none">millis()micros()delay()delayMicroseconds()	<ul style="list-style-type: none">sin()cos()tan()	
Communication			
<ul style="list-style-type: none">Serial			

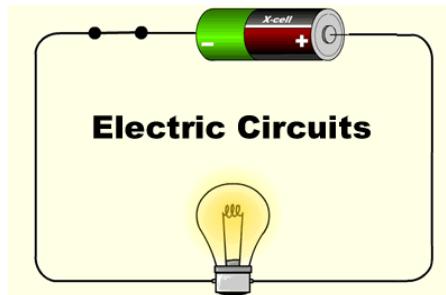
51

3. Basic Circuit Diagram



What is a Electric Circuit ?

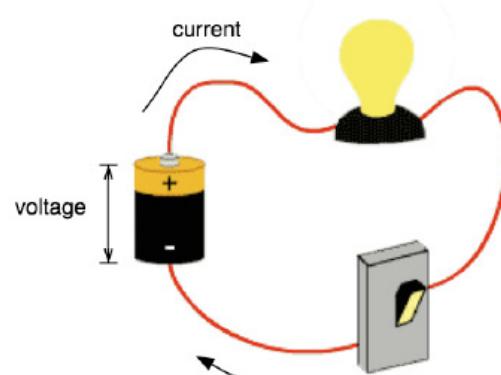
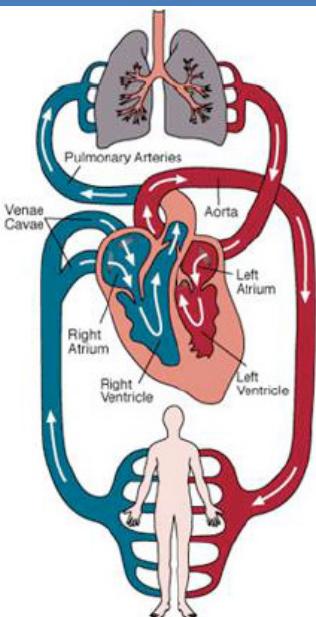
Electric Circuit หรือ วงจรไฟฟ้า คือ การต่อ กันของ อุปกรณ์ไฟฟ้า



อุปกรณ์ไฟฟ้า มี 2 ประเภท

- Active คือ ตัวจ่ายกำลังงาน เช่น แบตเตอรี่
- Passive คือ ตัวรับกำลังงาน เช่น หลอดไฟ ตัวต้านทาน..

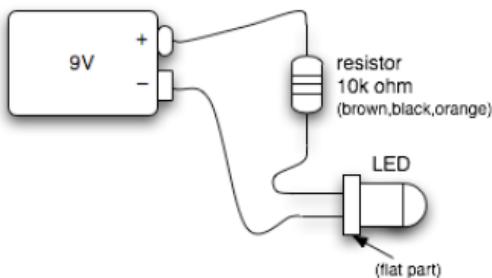
Making Circuits



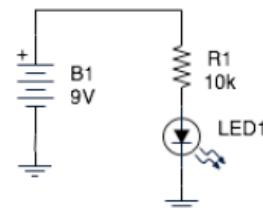
heart pumps, blood flows

voltage pushes, current flows

LED flashlight



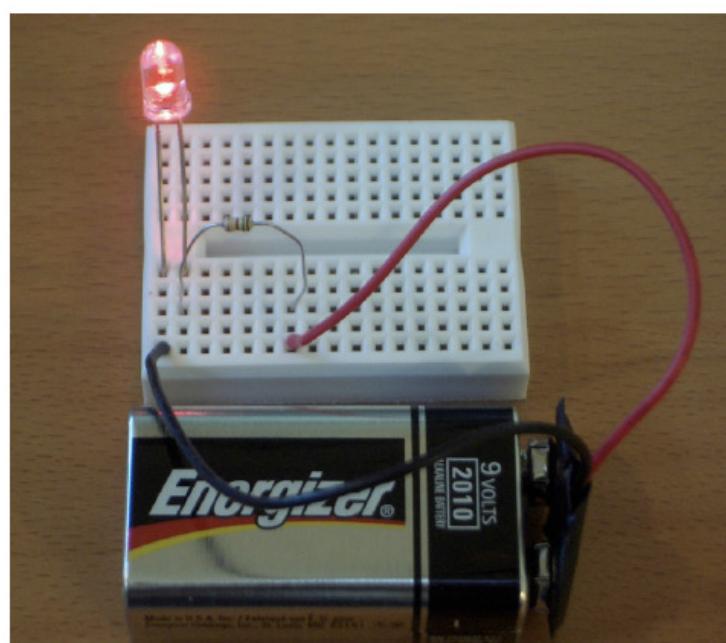
wiring diagram



schematic

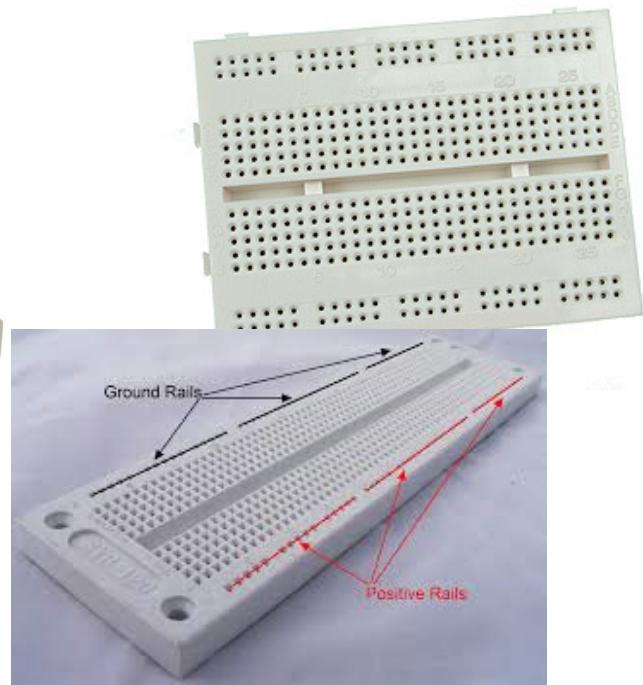
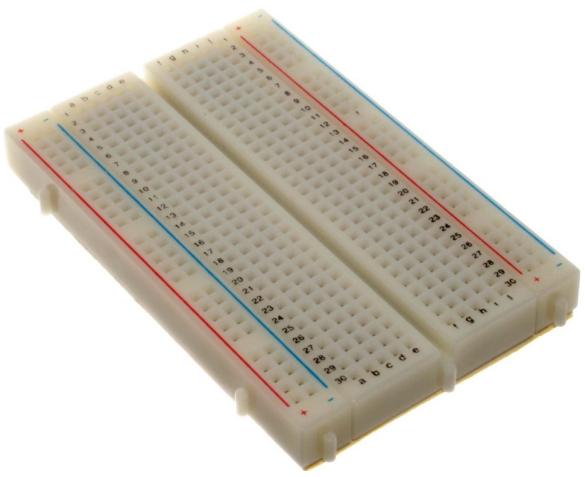
55

LED flashlight

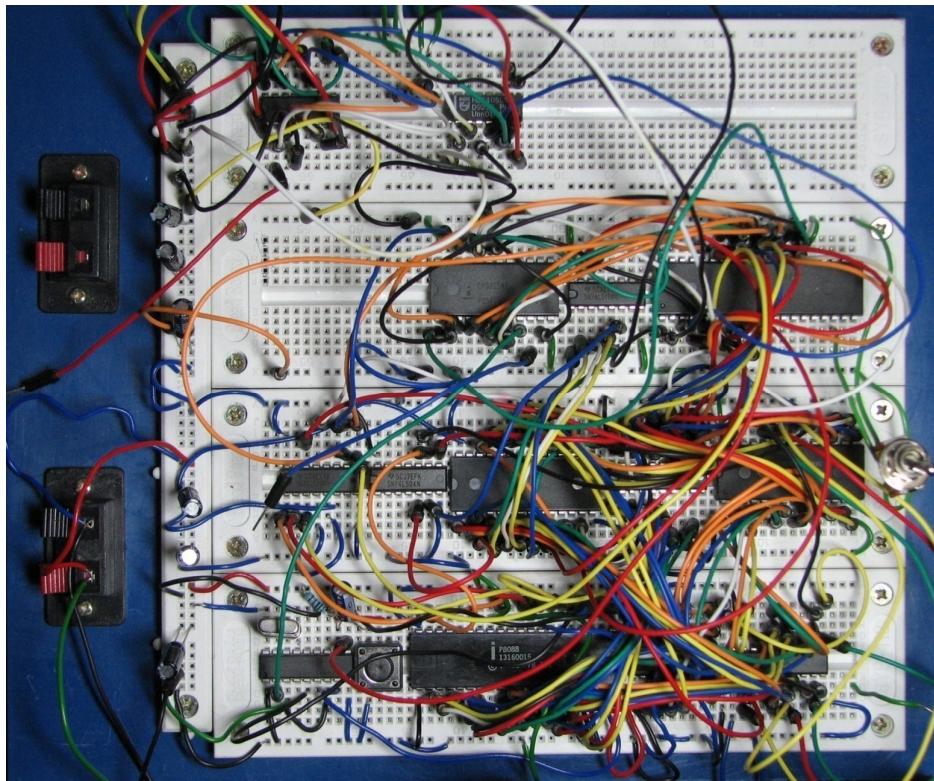


56

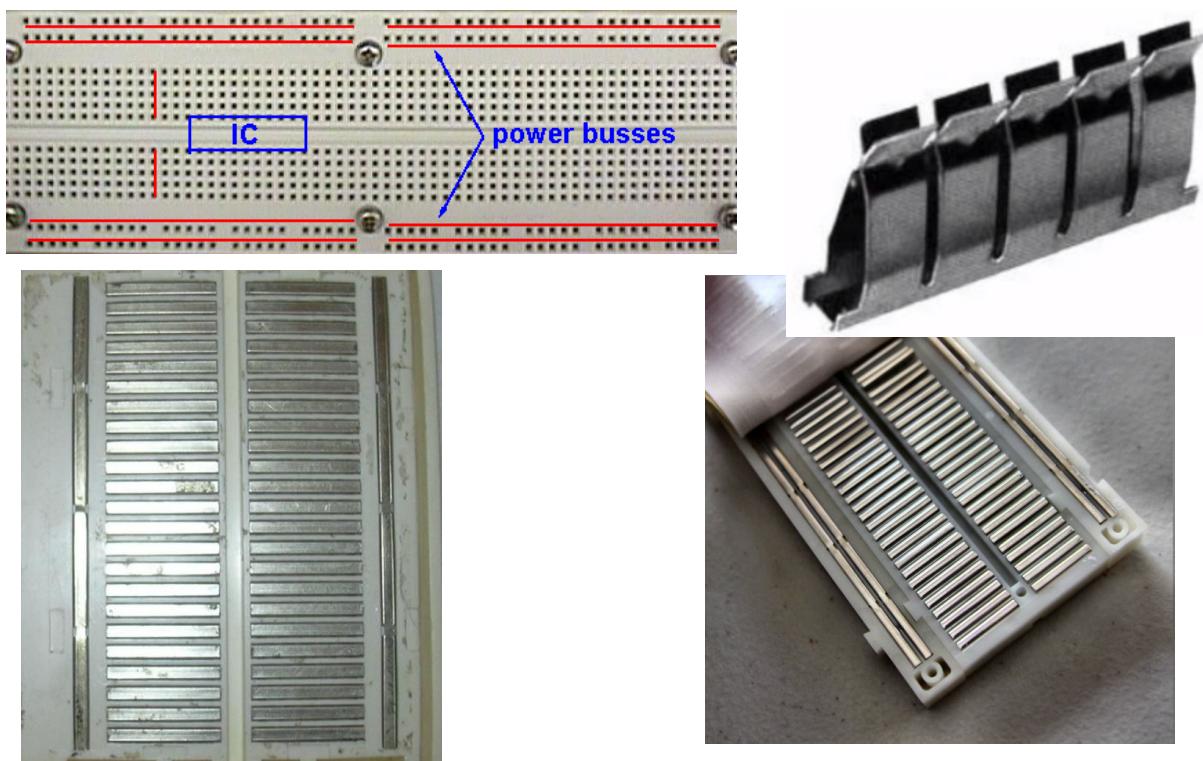
What is a Breadboard?



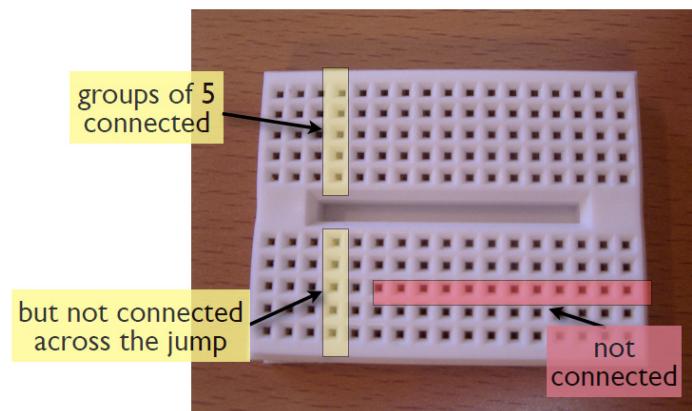
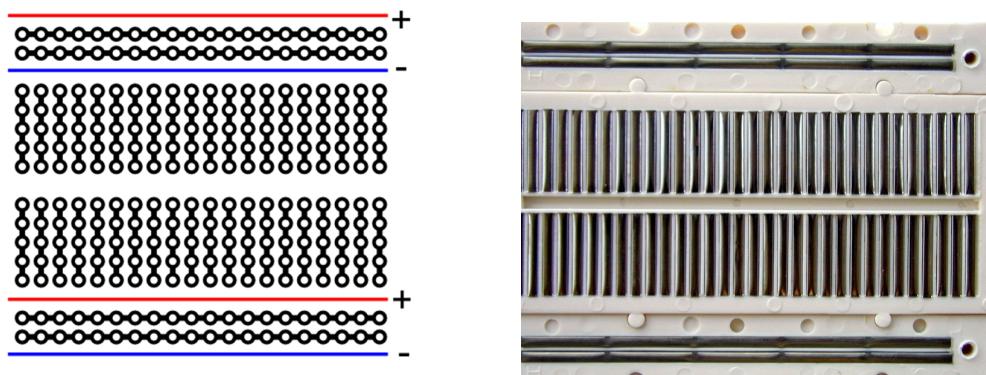
What circuit are we going to breadboard today?



How does a breadboard work?

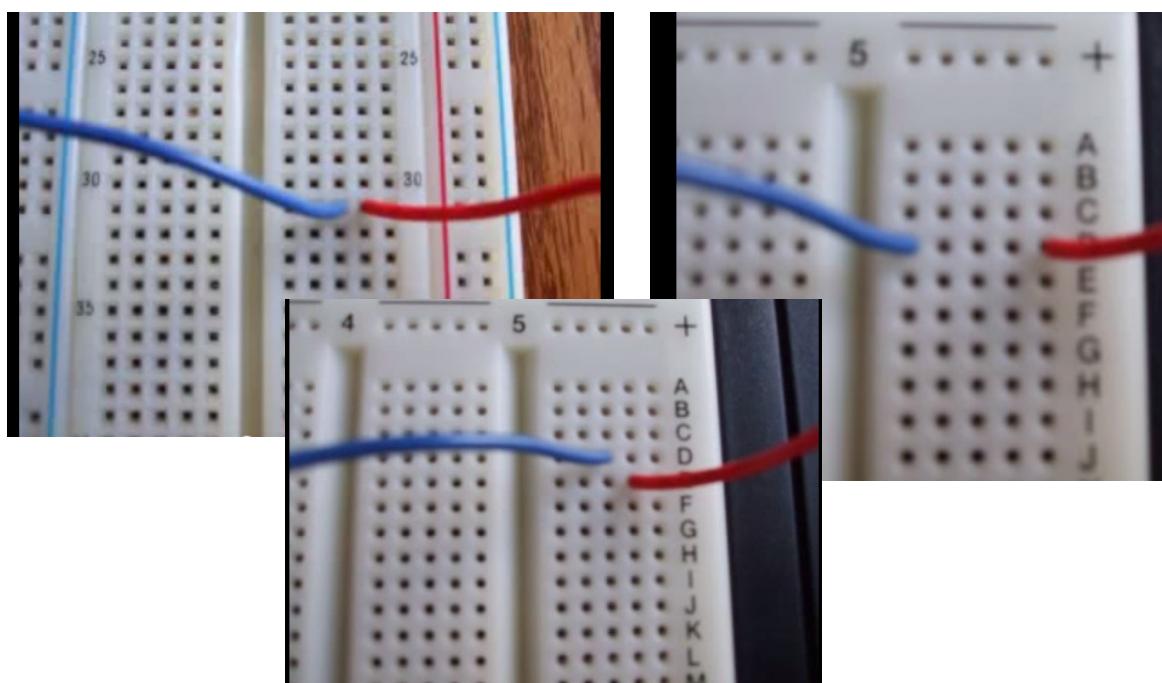


How does a breadboard work?

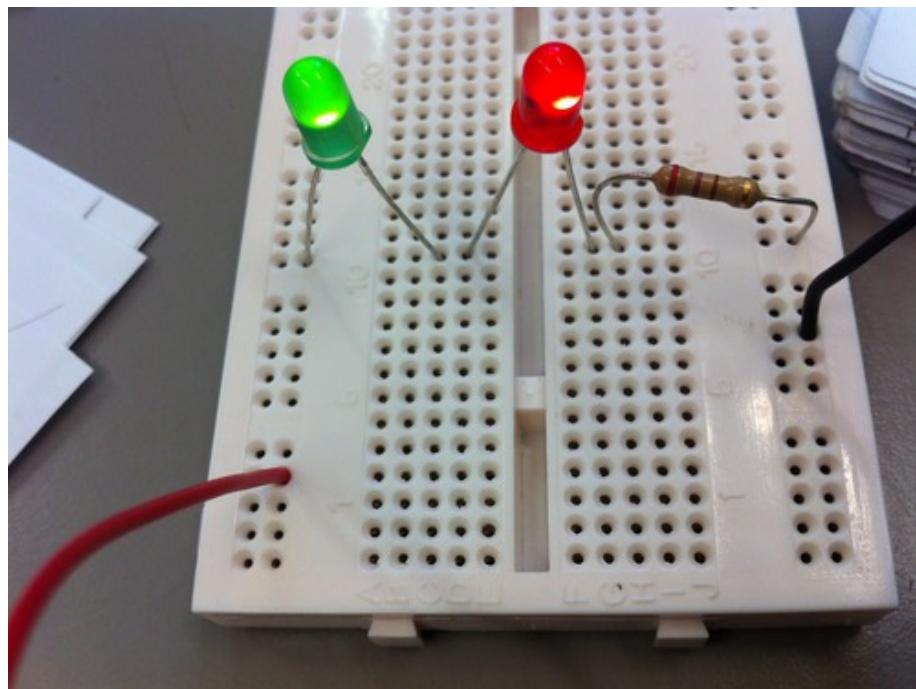


61

Are these wires connected?

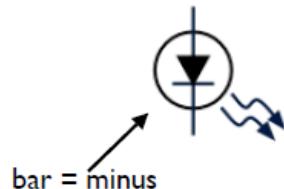
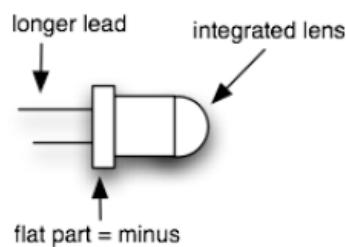


How to use a breadboard



LEDs

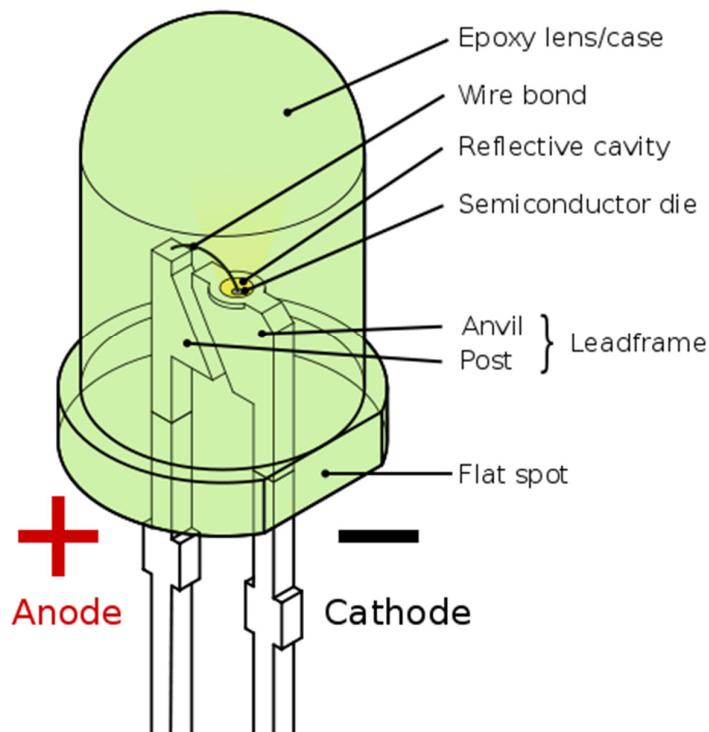
- LED = Light-Emitting Diode
 - electricity only flows one way in a diode
- Needs a “current limiting” resistor, or burns out



physical characteristics

schematic symbol

What are LEDs?



การอ่านค่า ตัวต้านทาน

0	1	2	3	4	5	6	7	8	9
Black	Brown	Red	Orange	Yellow	Green	Blue	Purple	Grey	White
0	1	2	3	4	5	6	7	8	9

Colour Codes

4 Band Resistors

5 Band Resistors

These colour codes and their uses are explained in the text

Resistor Color Code Examples:

4 Band Resistors: 27K EXAMPLE

Brown ±1%	Red ±2%	Gold ±5%	Silver ±10%*
0	X1	1	2
1	X10	2	2
2	X100	2	2
3	X1000	3	3
4	X10000	4	4
5	X100000	5	5
6	X1000000	6	6
7	+10 Gold	7	7
8	+100 Silver	8	8
9		9	9

5 Band Resistors: 15K EXAMPLE

Brown ±1%	Red ±2%	Gold ±5% *	Silver ±10% *
0	0	X1	1
1	1	X10	2
2	2	X100	2
3	3	X1000	3
4	4	X10000	4
5	5	+10 Gold	5
6	6	+100 Silver	6
7	7		7
8	8		8
9	9		9

Laboratory

การทดลองที่ 1 Getting Start with Arduino

1. จัดเตรียม Hardware และ Software เพื่อการพัฒนาโปรแกรมโดยแบ่งเป็น 3 ขั้นตอนดังนี้
 - 1.1 ติดตั้งโปรแกรม Arduino IDE
 - 1.2 ต่อบอร์ด Arduino Uno เข้ากับ Computer ด้วยสาย Mini USB
2. ทดสอบ บอร์ด Arduino Uno และโปรแกรม ว่า สามารถใช้งานได้จริง

67

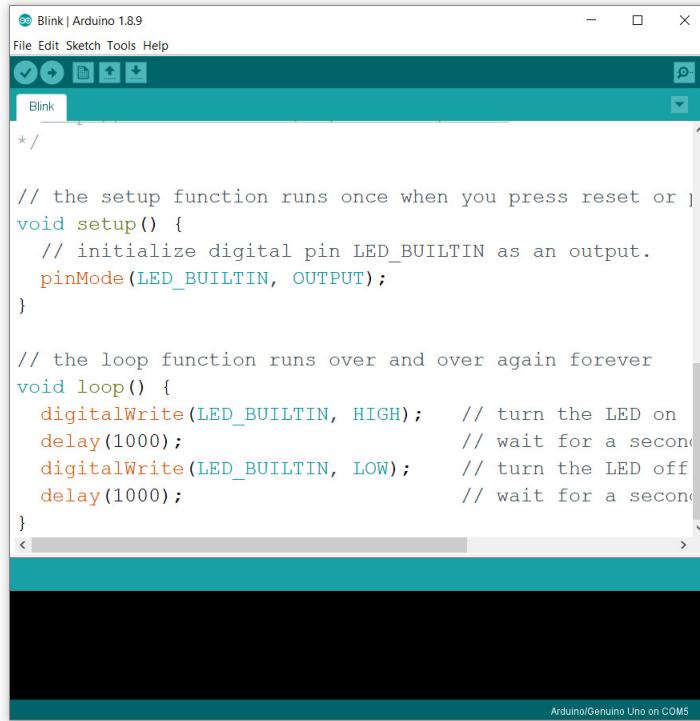
1.1 Installing Arduino IDE

- **Download the Arduino environment**
<http://arduino.cc/en/Main/Software>
โดยเลือก Version ล่าสุด ในที่นี่คือ Arduino 1.8.9
และเลือกตามระบบปฏิบัติการ ที่เราใช้ (Mac, Windows,...)
- **Install the Software**
Windows
 - Unzip โปรแกรม ไปที่ folder ที่ต้องการติดตั้ง**MAC OS**
 - Copy the Arduino application into the Applications folder

68

1.2 Test the Arduino board

- เปิดโปรแกรมตัวอย่าง File > Examples > 01.Basics > Blink



The screenshot shows the Arduino IDE interface with the 'Blink' sketch open. The code is as follows:

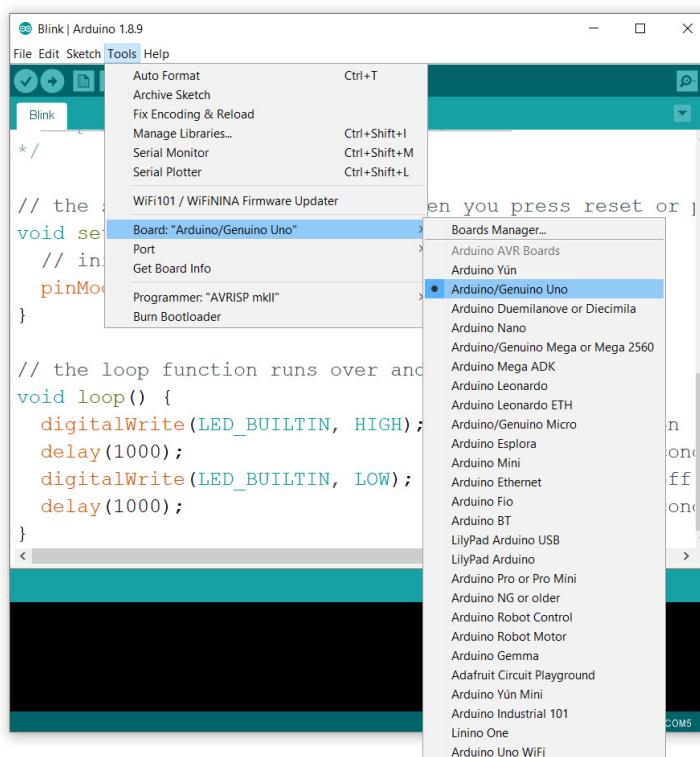
```
// the setup function runs once when you press reset or ]
void setup() {
    // initialize digital pin LED_BUILTIN as an output.
    pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
    digitalWrite(LED_BUILTIN, HIGH);      // turn the LED on
    delay(1000);                         // wait for a second
    digitalWrite(LED_BUILTIN, LOW);        // turn the LED off
    delay(1000);                         // wait for a second
}
```

The status bar at the bottom indicates "Arduino/Genuino Uno on COM5".

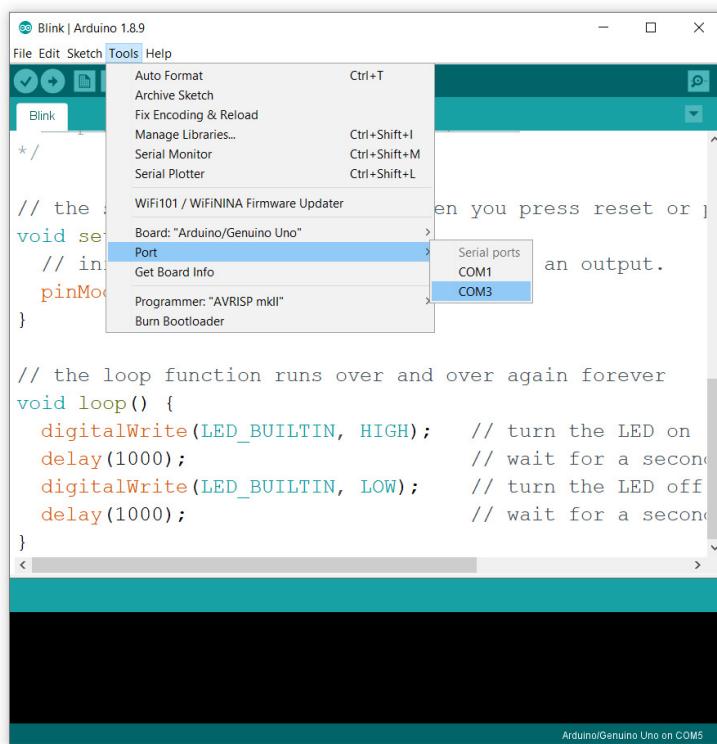
69

- Select your board : เลือกเป็น Arduino/Genuino Uno



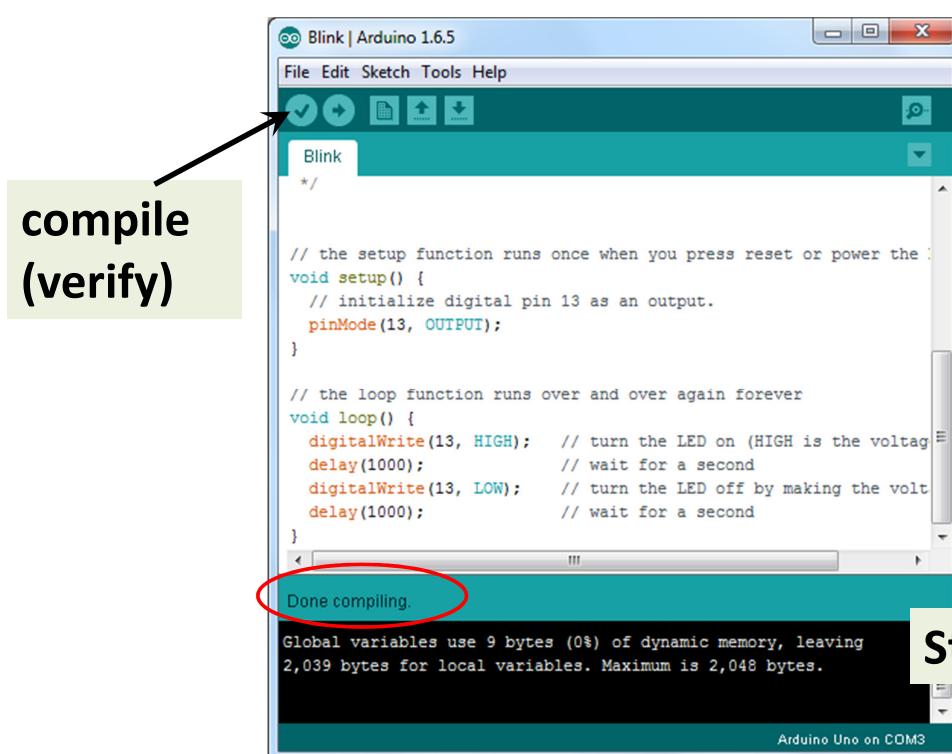
70

- **Select your Serial port :** เลือกเป็นพอร์ตที่ติดตั้งไว้ ปกติจะเป็น **COM3**



71

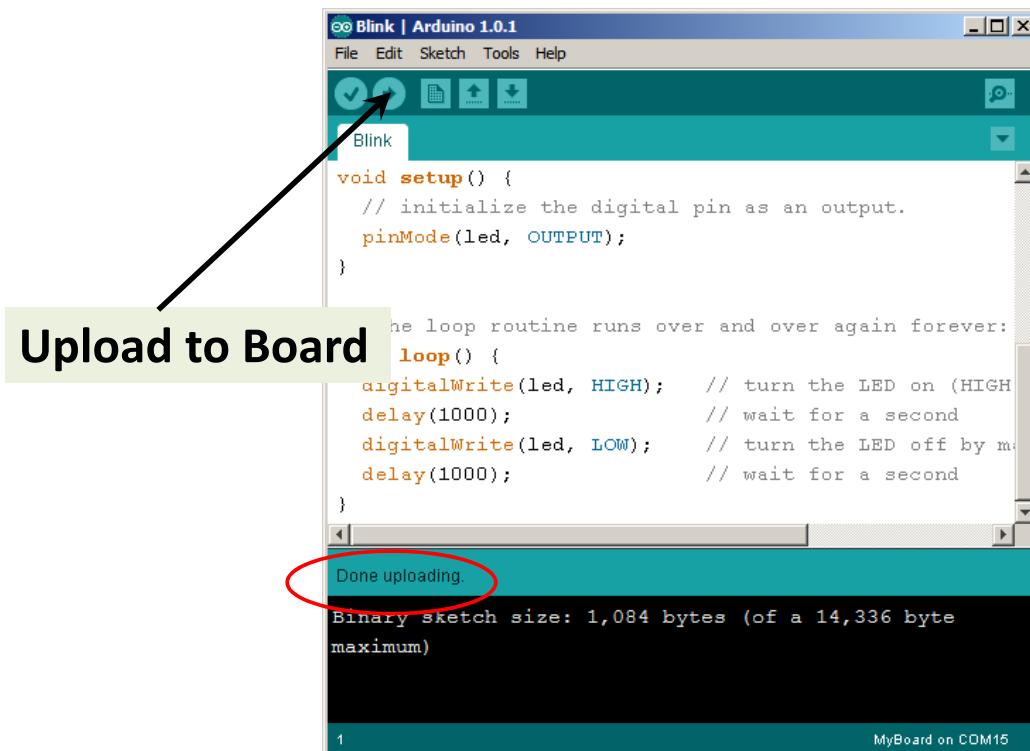
- **Compile your program** โดยการกดปุ่ม **verify**
และ รอจนกว่า **status** เป็น **Done Compiling**



Status Area

72

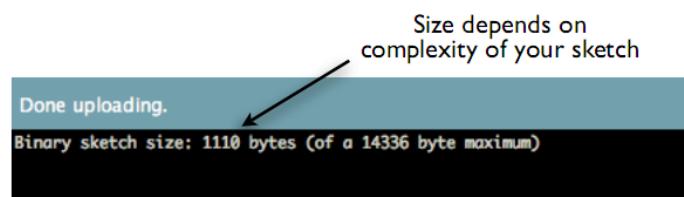
- Upload the program



73

1.3 Status Message

Uploading worked

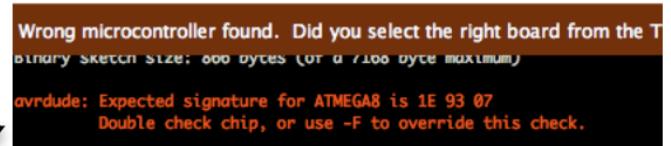


Wrong serial port selected



Wrong board selected

nerdy cryptic error messages



74

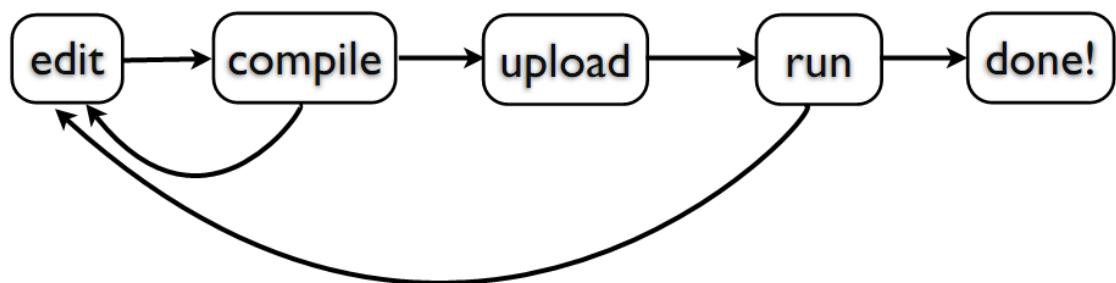
4. การส่งข้อมูลออกจากพอร์ต



arduino cube

1. ขั้นตอนการพัฒนาโปรแกรมด้วย Arduino (Development Circle)

- Make as many changes as you want
- Not like most web programming: edit → run
- Edit → compile → upload → run



2. Some useful function

- **pinMode()** – set a pin as input or output
- **digitalWrite()** – set a digital pin high/low
- **digitalRead()** – read a digital pin's state
- **analogRead()** – read an analog pin
- **analogWrite()** – write an “analog” value
- **delay()** – wait an amount of time
- **millis()** – get the current time

77

3. การเขียนโปรแกรมบน Arduino

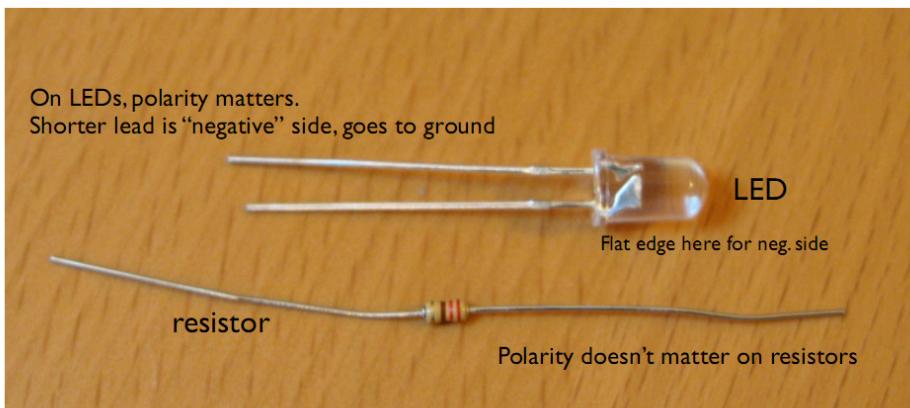
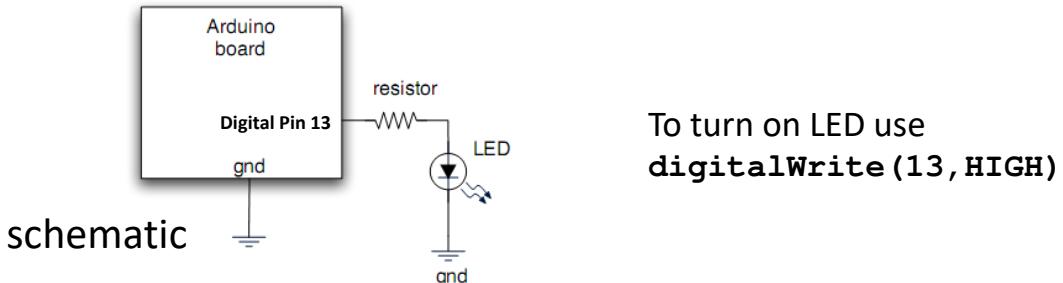
การเขียนโปรแกรมบน Arduino จะคล้ายกับการเขียนโปรแกรม C/C++
ไฟล์โปรแกรม ที่ทำการเขียนจะเรียกว่า **sketch files** มีส่วนประกอบสาม
ส่วน คือ

- **Declare variables at top**
- **Initialize**
 - **setup()** – run once at beginning, set pins
- **Running**
 - **loop()** – run repeatedly, after **setup()**

78

4. การส่งค่าออกพอร์ต แบบดิจิตอล

การทดลองที่ 1. LED Blink : Hello world of Microcontroller



รายละเอียดข้า อญ่าหน้า 19

79

โปรแกรม LED Blink แบบที่ 1

```
lab01
void setup() // Setup function
{
    pinMode(13,OUTPUT); // Digital Pin13 = Output pin
}

void loop() // Main function
{
    digitalWrite(13,HIGH); // Digital Pin13 = High
    delay(250); // wait 250 ms
    digitalWrite(13,LOW); // Digital Pin13 = Low
    delay(250); // wait 250 ms
}

Done compiling.
```

80

โปรแกรม LED Blink แบบที่ 2

```
lab02
int ledPIN = 13;           // ledPIN = DigitalPin 13
void setup()               // Setup function
{
    pinMode(ledPIN,OUTPUT); // ledPIN = Output pin
}

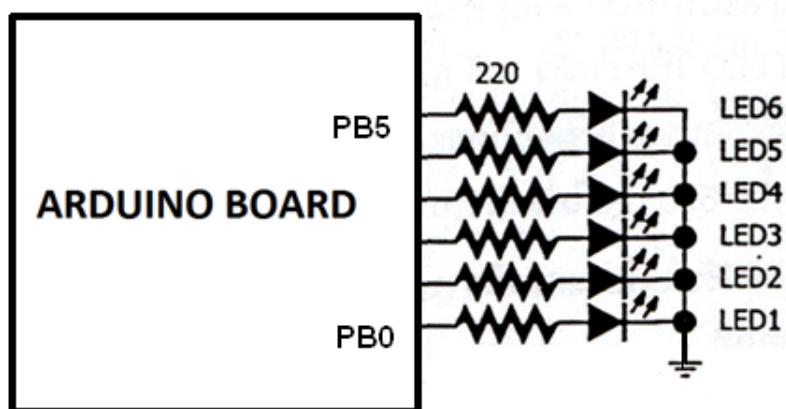
void loop()                // Main function
{
    digitalWrite(ledPIN,HIGH); // ledPIN = High
    delay(250);             // wait 250 ms
    digitalWrite(ledPIN,LOW); // ledPIN = Low
    delay(250);             // wait 250 ms
}

Done compiling.
```

81

การทดลองที่ 2 โปรแกรมไฟวิ่ง 6 ดวง

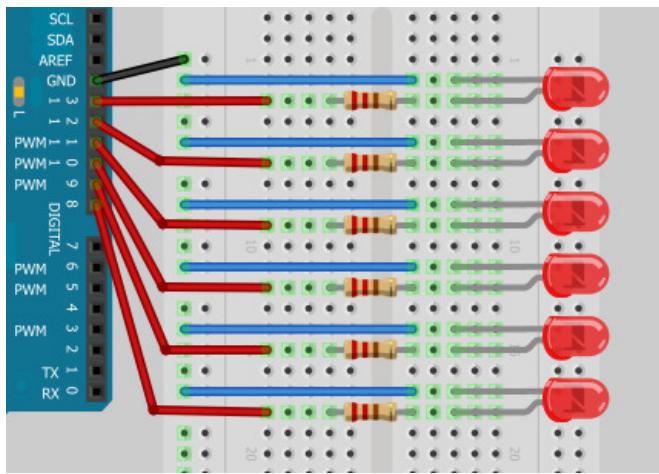
ต่อ LED เข้ากับ DigitalPin8 – DigitalPin13 (Port B0- B5)



รูป วงจรที่ใช้ในการทดลอง

82

โปรแกรมไฟวิ่ง



รูป การต่อใน Protoboard

```

lab03
void setup()
{
    pinMode(8,OUTPUT);
    pinMode(9,OUTPUT);
    pinMode(10,OUTPUT);
    pinMode(11,OUTPUT);
    pinMode(12,OUTPUT);
    pinMode(13,OUTPUT);
}
void loop()
{
    PORTB=0b000000001;
    delay(1000);
    PORTB=0b000000010;
    delay(1000);
    PORTB=0b000000100;
    delay(1000);
    PORTB=0b000001000;
    delay(1000);
    PORTB=0b000010000;
    delay(1000);
    PORTB=0b001000000;
    delay(1000);
}

```

83

Done compiling.

แบบฝึกหัดที่ 1.1 โปรแกรมควบคุม 7-segment

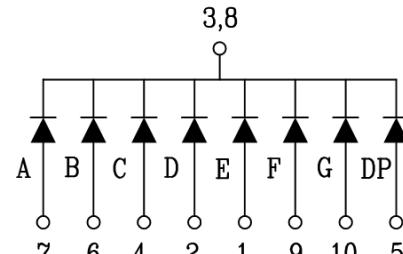
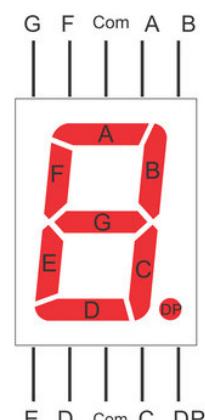
- ดัดแปลงวงจรไฟวิ่งในการทดลองที่ 2 โดยแทน LED ด้วย 7-segment

ATMega328P and Arduino Uno Pin Mapping

Arduino function		Arduino function
reset	(PCINT14/RESET) PC6	PC5 (ADC5/SCL/PCINT13)
digital pin 0 (RX)	(PCINT16/RXD) PD0	PC4 (ADC4/SDA/PCINT12)
digital pin 1 (TX)	(PCINT17/TXD) PD1	PC3 (ADC3/PCINT11)
digital pin 2	(PCINT18/INT0) PD2	PC2 (ADC2/PCINT10)
digital pin 3 (PWM)	(PCINT19/OC2B/INT1) PD3	PC1 (ADC1/PCINT9)
digital pin 4	(PCINT20/XCK/T0) PD4	PC0 (ADC0/PCINT8)
VCC	VCC	GND
GND	GND	AREF
crystal	(PCINT6/XTAL1/TOSC1) PB6	AVCC
crystal	(PCINT7/XTAL2/TOSC2) PB7	PB5 (SCK/PCINT5)
digital pin 5 (PWM)	(PCINT21/OC0B/T1) PD5	PB4 (MISO/PCINT4)
digital pin 6 (PWM)	(PCINT22/OC0A/AIN0) PD6	PB3 (MOSI/OC2A/PCINT3)
digital pin 7	(PCINT23/AIN1) PD7	PB2 (SS/OC1B/PCINT2)
digital pin 8	(PCINT0/CLKO/ICP1) PB0	PB1 (OC1A/PCINT1)

Digital Pins 11,12 & 13 are used by the ICSP header for MOSI.
MISO, SCK connections (Atmega16 pins 17,18 & 19). Avoid low-impedance loads on these pins when using the ICSP header.

- จงเขียนโปรแกรมแสดง 0-9 โดยเว้นช่วงละ 1 วินาที

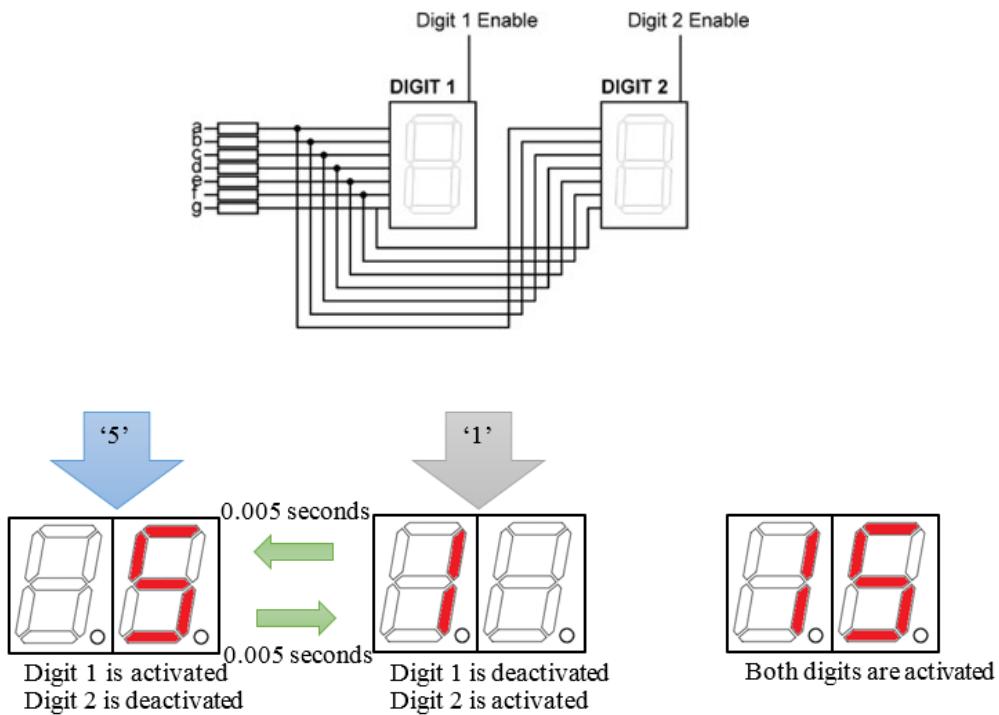


การจัดขาของ 7-segment

84

แบบฝึกหัดที่ 1.2 โปรแกรมควบคุม 7-segment

จงเขียนโปรแกรมแสดงเลข 2 หลัก เช่น 11 12 13 ... ช่วงละ 1 วินาที



85

แบบฝึกหัดที่ 2 โปรแกรมควบคุม Color LED

จงเขียนโปรแกรมแสดงสี 7 สี โดยเว้นช่วงสีละ 1 วินาที
กำหนดสีเอง ตามใจชอบ

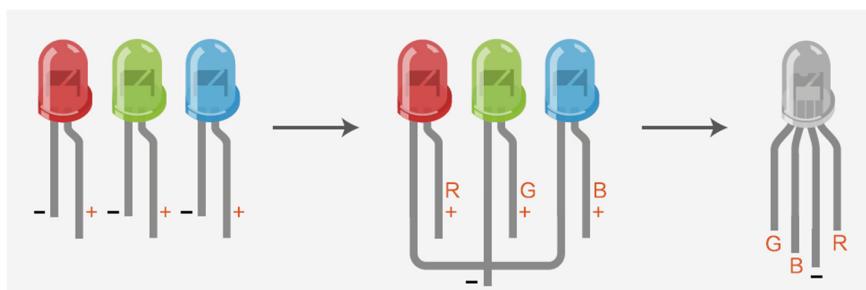


Fig. 5-4 Common Cathode RGB Diagram

Common Cathode

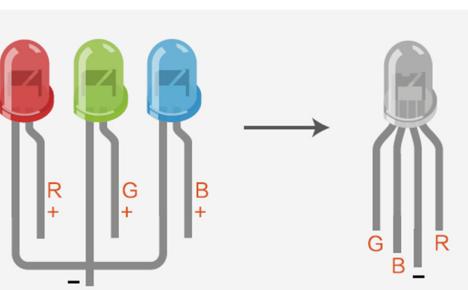
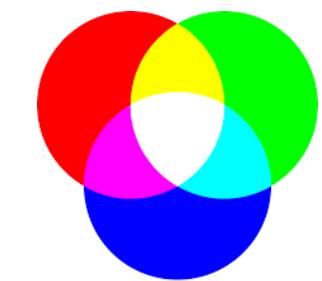


Fig. 5-5 Common Anode RGB Diagram

Common Anode



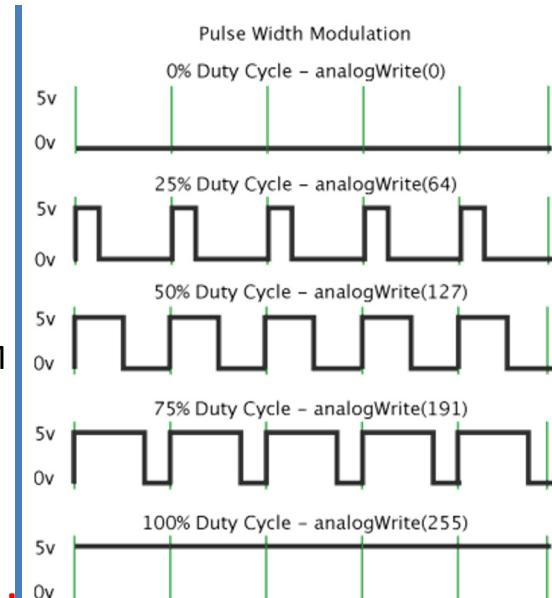
86

5. การส่งค่าออกพอร์ตแบบอนาล็อก

analogWrite()

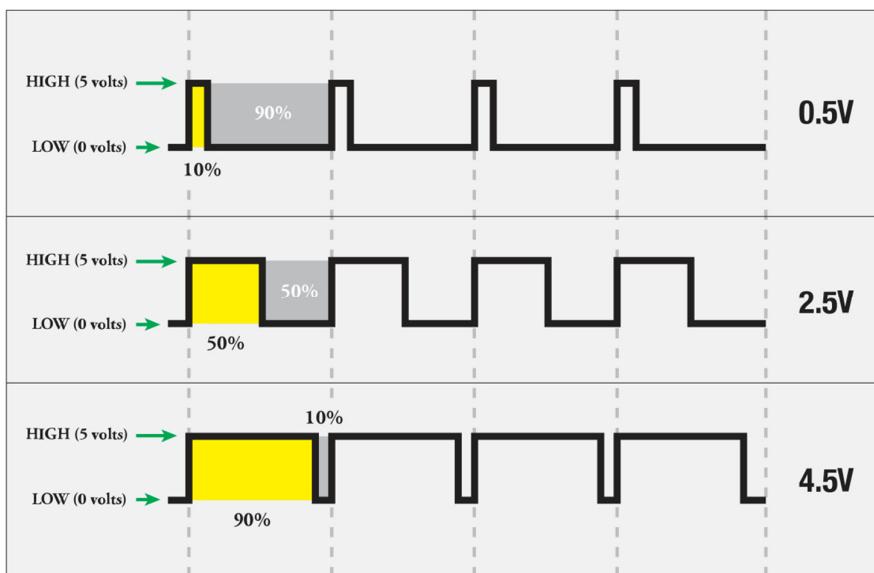
• Description

- Writes an analog value (PWM wave) to a pin. Can be used to light a LED at varying brightnesses or drive a motor at various speeds. After a call to `analogWrite()`, the pin will generate a steady square wave of the specified duty. The frequency of the PWM signal is approximately 490 Hz.
- On most Arduino boards (those with the ATmega168 or ATmega328), this function works on digital pins 3, 5, 6, 9, 10, and 11.
- You do not need to call `pinMode()` to set the pin as an output before calling `analogWrite()`.



87

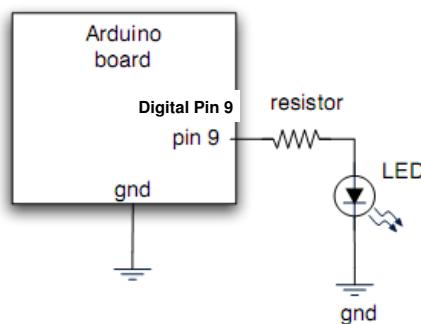
5. การส่งค่าออกพอร์ตแบบอนาล็อก



$$V_{DC\ output} = (\text{duty cycle}) \times V_{cc}$$

88

การทดลองที่ 3 LED Fading



- Same circuit as Blink circuit but pin 9 instead of pin 13
- The PWM pins work with the “`analogWrite(pin, value)`” command where
 - pin**: the pin to write to.
 - value**: the duty cycle: between 0 (always off) and 255 (always on).
- To turn LED to half-bright, use `analogWrite(9, 128)`

โปรแกรม LED Fading : Off to Full bright

```
lab04
int ledPin = 9;
void setup()
{
}
void loop()
{
    for(int fadeValue = 0 ; fadeValue <= 255; fadeValue +=5)
    {
        analogWrite(ledPin, fadeValue);
        delay(30);
    }
}
```

Done compiling.

Things to Try With “Fading”

- Make it go really fast or really slow
- Fading from half- to full-bright
- Try other PWM pins
- Multiple fading LEDs, at different rates

91

แบบฝึกหัดที่ 3 จงเขียนโปรแกรม **LED Fading**

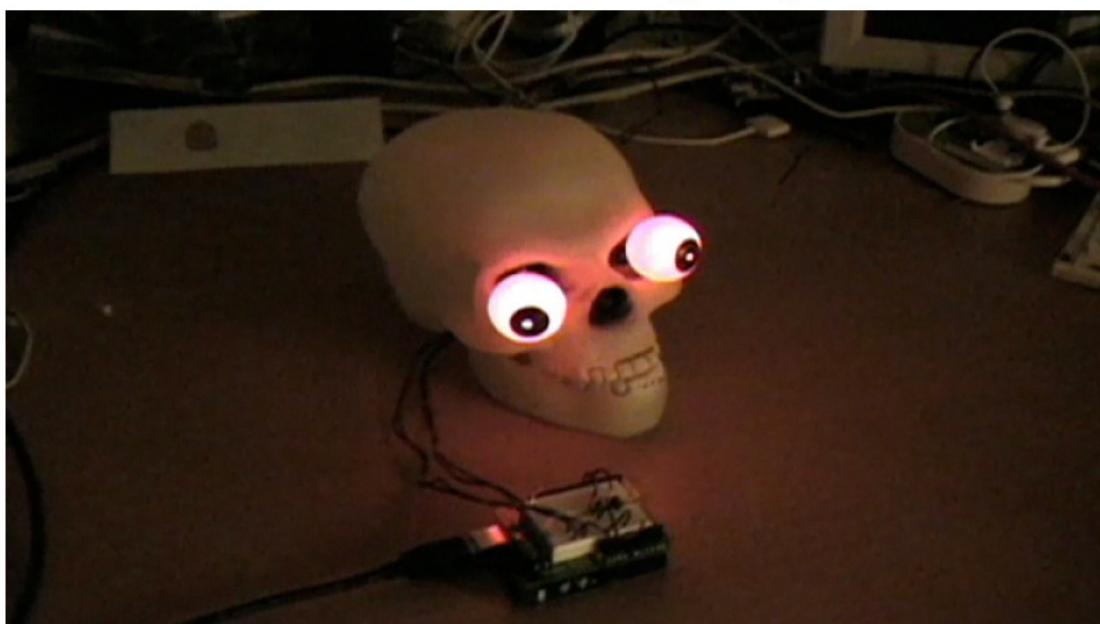
- เลือกใช้ PWM Pin จำนวน 4 Pin
- โดยแต่ละ Pin มีการ Fading ดังนี้
 - LED1 Fading จาก Off ไปยัง Full-Bright
 - LED2 Fading จาก Off ไปยัง Full-Bright เร็วเป็นสองเท่าของ LED1
 - LED3 Fading จาก Full-Bright ไปยัง Off
 - LED4 Fading จาก Half-Bright ไปยัง Full-Bright

92



93

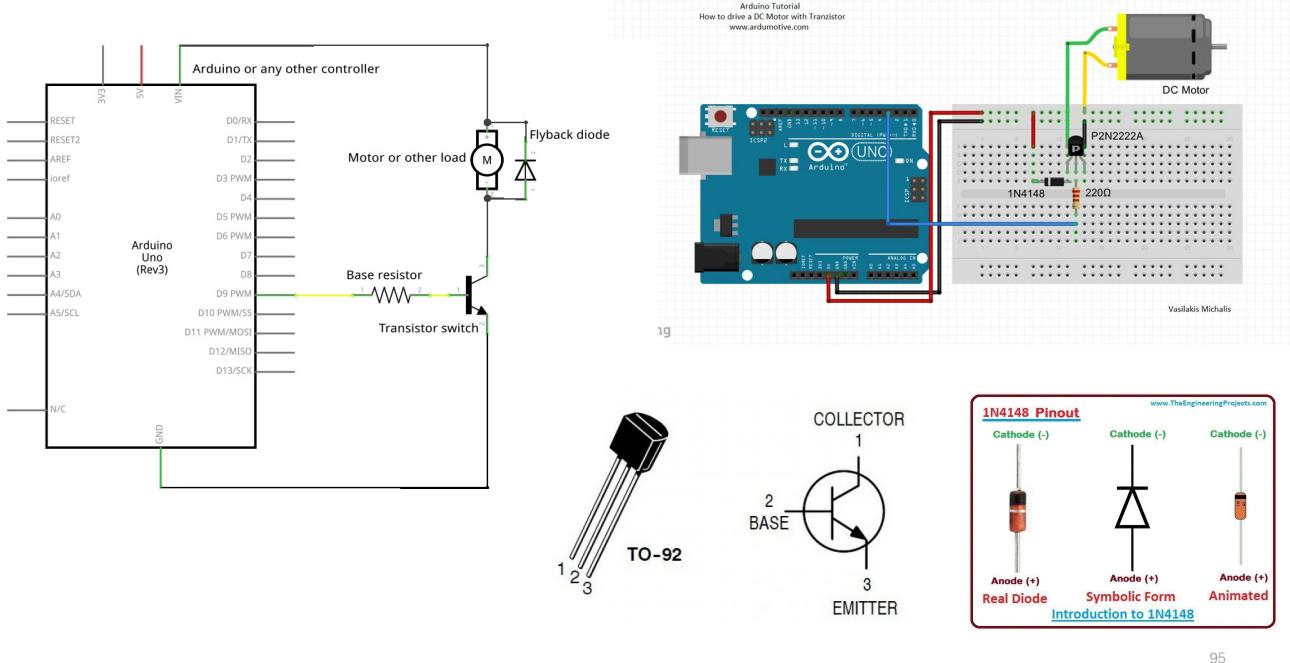
Evil Glowing Eyes



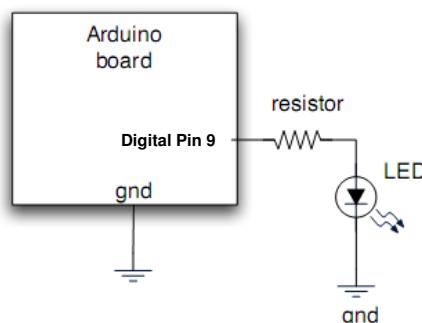
94

ตัวอย่าง Application

- Speed Control motor



สรุป : ข้อสังเกต



To turn ON:
To turn OFF:
To set brightness:

`digitalWrite(9,HIGH)`
`digitalWrite(9,LOW)`
`analogWrite(9,val)`

6. การขับเสียงออกลำโพง

tone()

- **Description**

Generates a square wave of the specified frequency (and 50% duty cycle) on a pin. A duration can be specified, otherwise the wave continues until a call to [noTone\(\)](#). The pin can be connected to a piezo buzzer or other speaker to play tones.

- **Syntax**

tone(pin, frequency)

tone(pin, frequency, duration)

- **Parameters**

pin: the pin on which to generate the tone

frequency: the frequency of the tone in hertz - *unsigned int*

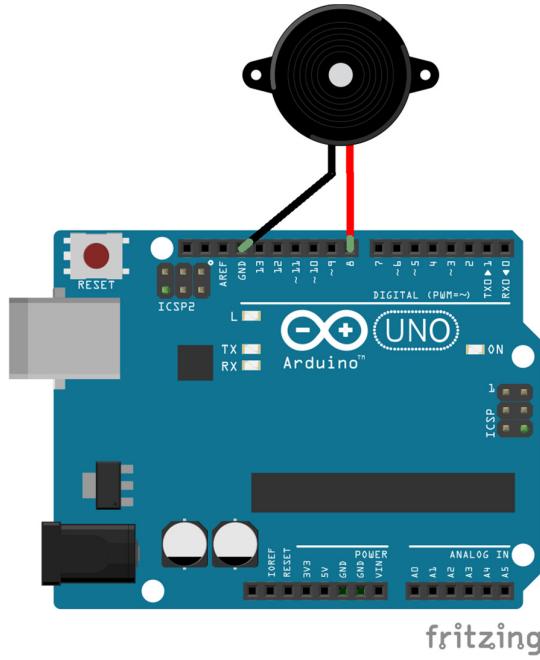
duration: the duration of the tone in milliseconds (optional) - *unsigned long*⁹⁷



Constant Name	Frequency (Hz)						
NOTE_B2	123	NOTE_FS4	370	NOTE_CS6	1109	NOTE_GS7	3322
NOTE_C3	131	NOTE_G4	392	NOTE_D6	1175	NOTE_A7	3520
NOTE_CS3	139	NOTE_GS4	415	NOTE_DS6	1245	NOTE_AS7	3729
NOTE_D3	147	NOTE_A4	440	NOTE_E6	1319	NOTE_B7	3951
NOTE_DS3	156	NOTE_AS4	466	NOTE_F6	1397	NOTE_C8	4186
NOTE_E3	165	NOTE_B4	494	NOTE_FS6	1480	NOTE_CS8	4435
NOTE_F3	175	NOTE_C5	523	NOTE_G6	1568	NOTE_D8	4699
NOTE_FS3	185	NOTE_CS5	554	NOTE_GS6	1661	NOTE_DS8	4978
NOTE_G3	196	NOTE_DS5	587	NOTE_A6	1760		
NOTE_GS3	208	NOTE_E5	622	NOTE_AS6	1865		
NOTE_A3	220	NOTE_F5	659	NOTE_B6	1976		
NOTE_AS3	233	NOTE_FS5	698	NOTE_C7	2093		
NOTE_B3	247	NOTE_G5	740	NOTE_CS7	2217		
NOTE_C4	262	NOTE_GS5	784	NOTE_D7	2349		
NOTE_CS4	277	NOTE_A5	831	NOTE_DS7	2489		
NOTE_D4	294	NOTE_AS5	880	NOTE_E7	2637		
NOTE_DS4	311	NOTE_B5	932	NOTE_F7	2794		
NOTE_E4	330	NOTE_C6	988	NOTE_FS7	2960		
NOTE_F4	349	NOTE_G6	1047	NOTE_G7	3136		

การทดลองที่ 4 Tone melody

- ต่อ ลำโพงเปียโซ อนุกรมกับตัวต้านทาน ที่ขา digital 8 ของบอร์ด ดังรูป



99

โปรแกรม tone melody

```
toneMelody § pitches.h
#include "pitches.h"
int melody[] = {NOTE_C4, NOTE_G3, NOTE_G3, NOTE_A3, NOTE_G3, 0, NOTE_B3, NOTE_C4};
int noteDurations[] = {4, 8, 8, 4, 4, 4, 4, 4 };
void setup() {
    // iterate over the notes of the melody:
    for (int thisNote = 0; thisNote < 8; thisNote++) {
        int noteDuration = 1000/noteDurations[thisNote];
        tone(8, melody[thisNote],noteDuration);
        int pauseBetweenNotes = noteDuration * 1.30;
        delay(pauseBetweenNotes);
        noTone(8);
    }
}
void loop() { // no need to repeat the melody.
}
```

Done compiling.

Note!! : note durations: 4 = quarter note, 8 = eighth note

100

แบบฝึกหัดที่ 4 จงเขียนโปรแกรม Melody

- จงเขียนโปรแกรม
Melody เพลง
Jingle Bells

โดยมีโน๊ต แสดงดังรูป



Jingle Bells

Primer Level

J. Pierpont

Arr: Gilbert DeBenedetti

With good cheer

Jin - gle, bells, Jin - gle, bells, Jin - gle all the way,

Oh, what fun it is to ride a one horse o - pen sleigh.

Note !! Basic of Music

ในการอ่านโน๊ต เล่นเพลง จะใช้ค่า 3 อย่าง

- โน๊ตที่เล่น
- อัตราสัดส่วนของโน๊ต แต่ละตัว ว่า เล่นนาน หรือ ช้า เท่าไหร่
- ความเร็วในการเล่น โน๊ต (tempo)

1. โน๊ต บนบรรทัด 5 เส้น

C⁴ D⁴ E⁴ F⁴ G⁴ A⁴ B⁴ C⁵ D⁵ E⁵ F⁵ G⁵
E² F² G² A² B² C³ D³ E³ F³ G³ A³ B³

Note Frequency

C	262
D	294
E	330
F	349
G	392
A	440
B	495



2. Tempo หมายถึง ความเร็วของบทเพลงต่าง ๆ ที่อัตราความช้าเร็วต่างกันออกไป ทั้งนี้ขึ้นอยู่กับผู้ประพันธ์เพลงผู้กำหนดว่าจะให้มีความช้า – เร็ว เท่าไร อาจมีจังหวะเร็ว ปานกลาง หรือช้าก็ได้

largo	(very slow, broad) 40-56	ช้ามาก
grave	(very slow, solemn)	ช้ามาก ๆ
adagio	(slow) 58-70	ช้า ๆ ไม่รีบร้อน
andante	(moderately slow) 72-90	ช้า, ก้าวสบาย ๆ
moderrato	(moderate) 93-100	ความเร็วปานกลาง
allegretto	(moderately fast) 102-120	ค่อนข้างเร็ว
allegro	(fast) 125-134	เร็ว
vivace	(lively) 136-172	เร็วชื่นแบบมีชีวิตชีวา
presto	(very fast) 174-216	เร็วมากทันทีทันใด
prestissimo	(as fast as possible) 218-	เร็วที่สุด

tempo 60 คือ มี **60** จังหวะ ในหนึ่งนาที

103

Note !!

3. เลขเศษส่วน 4:4 (Time Signature) ที่อยู่บนบรรทัดหัวเส้น คือเครื่องหมายกำหนดจังหวะเลขด้วยบัน หมายถึงจำนวนของตัวโน้ตในหนึ่งห้อง สำหรับเพลงนี้ต้องมีโน้ต 4 จังหวะ : 1 ห้อง เลขตัวล่าง หมายถึงลักษณะของตัวโน้ตที่ต้องการให้ยืดเป็นเกณฑ์หนึ่งจังหวะ สำหรับเพลงนี้ ตัวล่างเป็น 4 หมายความว่า ตัวคำ 1 ตัวนับเป็น 1 จังหวะ

4. อัตราของตัวโน้ต (Note Values) อัตรายาวและสั้นของตัวโน้ตลักษณะต่าง ๆ มีดังนี้

ITEM	NOTE	REST	VALUE (number of beats)
Whole note/rest	○	---	4
Half note/rest	♩	—	2
Quarter note/rest	♪	♪	1
Eighth note/rest	♪	♪	1/2
Sixteenth note/rest	♪	♪	1/4

©EnchantedLearning.com

104

The END

