หน่วยที่ 4 ฟังก์ชั่นพื้นฐานของ ARDUINO และการควบคุมหลอดไฟ LED

สาระสำคัญ

โปรแกรม Arduino IDE ได้จัดเตรียมฟังก์ชั่นพื้นฐาน เช่นฟังก์ชั่นเกี่ยวกับขาพอร์ตอินพุตเอาต์พุตดิจิตอล, อินพุตเอาต์พุตแอนะล็อกเป็นต้น ดังนั้นในการเขียนโปรแกรมจึงเรียกใช้ฟังก์ชั่นเหล่านี้ได้ทันทีโดยไม่ต้องใช้คำสั่ง #include เพื่อผนวกไฟล์เพิ่มเติมแต่อย่างใด นอกจากฟังก์ชั่นพื้นฐานเหล่านี้แล้ว นักพัฒนาท่านอื่นๆ ที่ร่วมในโครงการ Arduino นี้ก็ได้เพิ่มไลบรารีอื่นๆ เช่นไลบรารีควบคุมมอเตอร์, การติดต่อกับอุปกรณ์บัส I2C ฯลฯ ในการเรียกใช้งานต้องเพิ่มบรรทัด #include เพื่อผนวกไฟล์ที่เหมาะสมก่อน จึงจะเรียกใช้ฟังก์ชั่นได้

เนื้อหาสาระการเรียนรู้

4.1 ฟังก์ชั่นอินพุต เอาต์พุตดิจิตอล (Digital I/0)
4.2 ฟังก์ชั่นเกี่ยวกับการสื่อสารผ่านพอร์ตอนุกรม
4.3 ฟังก์ชั่นอินพุตเอ้าต์พุตแอนะล็อก
4.4 ฟังก์ชั่นเกี่ยวกับเวลา
4.5 ฟังก์ชั่นเกี่ยวกับอินเตอร์รัปต์ภายนอก
4.6 ฟังก์ชั่นทางคณิตศาสตร์
4.7 ฟังก์ชั่นเกี่ยวกับเลขสุ่ม

จุดประสงค์การเรียนรู้

จุดประสงค์ทั่วไป

1. เพื่อให้มีความรู้ความเข้าใจเกี่ยวกับลักษณะฟังก์ชั่นและกระบวนการทำงานของ Arduino
2. เพื่อให้สามารถนำความรู้ไปประยุกต์ใช้ในการเขียนโปรแกรมกำหนดการทำงานพื้นฐานของ Arduino
3. เพื่อให้ตระหนักถึงความสำคัญของลักษณะฟังก์ชั่นและกระบวนการทำงานของ Arduino

จุดประสงค์เชิงพฤติกรรม

1. ใช้ฟังก์ชั่นอินพุต เอาต์พุตดิจิตอล (Digital I/0) ได้
2. ใช้ฟังก์ชั่นเกี่ยวกับการสื่อสารผ่านพอร์ตอนุกรมได้
3. ใช้ฟังก์ชั่นอินพุตเอ้าต์พุตแอนะล็อกได้
4. ใช้ฟังก์ชั่นเกี่ยวกับเวลาได้
5. ใช้ฟังก์ชั่นเกี่ยวกับอินเตอร์รัปต์ภายนอกได้
6. ใช้ฟังก์ชั่นทางคณิตศาสตร์ได้
7. ใช้ฟังก์ชั่นเกี่ยวกับเลขสุ่มได้

แบบทดสอบก่อนเรียน หน่วยที่ 4
เรื่อง ฟังก์ชั่นพื้นฐานของ Arduino

เรื่อง ฟังก์ชั่นพื้นฐานของ Arduino **ใช้เวลา** 20 **นาที**

วิชา ไมโครคอนโทรลเลอร์เบื้องต้น **รหัสวิชา** (2127-2107)

ระดับชั้น ประกาศนียบัตรวิชาชีพ (ปวช.) **สาขาวิชา** เมคคาทรอนิกส์

****

คำชี้แจง 1. แบบทดสอบมีทั้งหมด 10 ข้อ (10 คะแนน)

2. ให้ผู้เรียนเลือกคำตอบที่ถูกที่สุดแล้วกาเครื่องหมายกากบาท (✕) ลงในกระดาษคำตอบ

1. โปรแกรม Arduino IDE สามารถเรียกใช้ฟังก์ชั่นพื้นฐานได้ทันทีโดยไม่ต้อง
   - ก. ใช้คำสั่ง #include เพื่อผนวกไฟล์เพิ่ม
   - ข. ใช้คำสั่ง #plus เพื่อผนวกไฟล์เพิ่ม
   - ค. ใช้คำสั่ง #add เพื่อผนวกไฟล์เพิ่ม
   - ง. ใช้คำสั่ง #defile เพื่อผนวกไฟล์เพิ่ม
2. ฟังก์ชั่น digitalWrite (pin, value) ทำงานอย่างไร
   - ก. ใช้เลือกโหมดการทำงานเป็น OUTPUT
   - ข. ใช้เลือกโหมดการทำงานเป็น INPUT
   - ค. ใช้กำหนดขาพอร์ต INPUT ให้มีสถานะเป็นลอจิกสูงหรือลอจิกต่ำ
   - ง. ใช้กำหนดขาพอร์ต OUTPUT ให้มีสถานะเป็นลอจิกสูงหรือลอจิกต่ำ
3. ฟังก์ชั่นเกี่ยวกับการสื่อสารผ่านพอร์ตอนุกรมคือข้อใด
   - ก. Serial .begin (int datarate)
   - ข. digitalWrite
   - ค. pinMode
   - ง. digitalRead
4. ฟังก์ชั่น int Serial .available ( ) ใช้สำหรับ
   - ก. ใช้แจ้งว่าได้รับข้อมูลตัวอักษร และพร้อมสำหรับการอ่านไปใช้งาน
   - ข. ใช้แจ้งว่าได้รับสัญญาณแล้ว และพร้อมสำหรับการอ่านไปใช้งาน
   - ค. ส่งค่ากลับจากฟังก์ชั่น และพร้อมสำหรับการอ่านไปใช้งาน
   - ง. เป็นการเลือกอัตราบอดเท่ากับ 9600 บิตต่อวินาที

5. ฟังก์ชั่น int Serial .read ( ) ใช้สำหรับ
   ก. อ่านค่าข้อมูลที่ได้รับจากพอร์ตอนุกรม
   ข. อ่านค่าข้อมูลที่ได้รับจากพอร์ตขนาน
   ค. ใช้เขียนแจ้งว่ากำลังลบคำสั่ง
   ง. ใช้เขียนแจ้งว่ากำลังอ่านคำสั่ง
6. . ถ้าต้องการกำหนดขาอินพุตแอนะล็อกต้องกำหนดด้วยฟังก์ชั่น
   ก. AnalogRead
   ข. AnalogWrite
   ค. ModePin
   ง. digitalRead
7. ฟังก์ชั่น unsigned long millis() คืนค่าเป็นค่าเวลาในหน่วยมิลลิวินาที ได้ประมาณกี่ชั่วโมง
   ก. ประมาณ 9 ชั่วโมง
   ข. ประมาณ 8 ชั่วโมง
   ค. ประมาณ 7 ชั่วโมง
   ง. ประมาณ 6 ชั่วโมง
8. ฟังก์ชั่น detachInterrupt(interrupt) ใช้สำหรับ
   ก. ยกเลิกการอินเตอร์รัปต์ภายนอก
   ข. เริ่มการอินเตอร์รัปต์ภายนอก
   ค. เลือกประเภทสัญญาณที่ใช้กระตุ้นให้เกิดการอินเตอร์รัปต์
   ง. ฟังก์ชั่นกระโดดไปทำงานเมื่อเกิดอินเตอร์รัปต์
9. ฟังก์ชั่น constrain(x, a, b) ทำงานอย่างไร
   ก. หาค่าสัมบูรณ์ของตัวเลข
   ข. หาค่าตัวเลขที่มากที่สุดของตัวเลขสองตัว
   ค. หาค่าตัวเลขที่น้อยที่สุดของตัวเลขสองตัว
   ง. ปัดค่าตัวเลขที่น้อยกว่าหรือมากกว่าให้อยู่ในช่วงที่กำหนด
10. randomSeed(seed) คือฟังก์ชั่นอะไร
   ก. ใช้กำหนดตัวแปรสำหรับสร้างตัวเลขแบบสุ่ม
   ข. ใช้หาค่าสัมบูรณ์ของตัวเลขแบบสุ่ม
   ค. ใช้หาค่าที่น้อยที่สุดของตัวเลขสองตัวแบบสุ่ม
   ง. ใช้สร้างตัวเลขเสมือนแบบสุ่ม

หน่วยที่ 4
ฟังก์ชั่นพื้นฐานของ Arduino และการควบคุมหลอดไฟ LED

โปรแกรม Arduino IDE ได้จัดเตรียมฟังก์ชั่นพื้นฐาน เช่นฟังก์ชั่นเกี่ยวกับขาพอร์ตอินพุตเอาต์พุตดิจิตอล, อินพุตเอาต์พุตแอนะล็อกเป็นต้น ดังนั้นในการเขียนโปรแกรมจึงเรียกใช้ฟังก์ชั่นเหล่านี้ได้ทันทีโดยไม่ต้องใช้คำสั่ง #include เพื่อผนวกไฟล์เพิ่มเติมแต่อย่างใด

นอกจากฟังก์ชั่นพื้นฐานเหล่านี้แล้ว นักพัฒนาท่านอื่นๆ ที่ร่วมในโครงการ Arduino นี้ก็ได้เพิ่มไลบรารีอื่นๆ เช่น ไลบรารีควบคุมมอเตอร์, การติดต่อกับอุปกรณ์บัส I2C ฯลฯ ในการเรียกใช้งานต้องเพิ่มบรรทัด #include เพื่อผนวกไฟล์ที่เหมาะสมก่อน จึงจะเรียกใช้ฟังก์ชั่นได้

ในบทนี้จะอธิบายถึงการเรียกใช้ฟังก์ชั่นและตัวอย่างโปรแกรมสำหรับทำการทดลอง โดยใช้บอร์ด Arduino Uno สำหรับวิธีการทดลองสามารถดูได้จากหน่วยที่ 2

4.1 ฟังก์ชั่นอินพุต เอาต์พุตดิจิตอล (Digital I/0)

คำอธิบายและการเรียกใช้ฟังก์ชั่น

4.1.1 pinMode (pin,mode)

ใช้กำหนดขาพอร์ตใดๆให้เป็นพอร์ตดิจิตอล

พารามิเตอร์

pin – ใช้กำหนดขาพอร์ตใดๆ

mode – โหมดการทำงานเป็น INPUT หรือ OUTPUT (ค่าเป็น int)

ตัวอย่างที่ 4.1

```
int ledPin = 13;                    / / LED connected to Di pin 13
 void setup ( )
{
pinMode ( ledPin, OUTPUT) ;         / / sets as output
{
void  loop ( )
{
digitalWrite(ledpin,  HIGH);        / / LED on
delay (1000);                       / / waits for a second
digitalWrite (ledPin, LOW);         / / LED off
delay (1000); }
```

4.1.2 digitalWrite (pin, value)

สั่งงานให้ขาพอร์ที่ระบุไว้มีค่าสถานะเป็นลอจิกสูง (HIGH หรือ 1) หรือลอจิกต่ำ (LOW หรือ 0)

พารามิเตอร์

pin – ขาพอร์ของโมดูล

value – มีค่า HIGH หรือ LOW

ตัวอย่างที่ 4.2

```
int  ledpin = 13;                    / / LED connected to Di pin 13
void setup ()
{
pinMode (ledPin, OUTPUT)             / / sets as output
}
void loop ()
{
digitalWrite(ledPin, HIGH);          / / LED on
delay (300);                         / / waits for 0.3 second
digitalWrite(ledPin, LOW);           / / LED off
delay (300);                         / / waits for 0.3 second
}
```

กำหนดให้ขา 13 เป็น HIGH (มีลอจิกเป็น 1) หน่วงเวลา 1 วินาที แล้วจึงสั่งให้ขา 13 กลับเป็น LOW มีลอจิกเป็น 0 อีกครั้ง

4.1.3 digitalRead (pin)

อ่านค่าสถานะของขาที่ระบุไว้ว่ามีค่าเป็น HIGH หรือ LOW

พารามิเตอร์

pin – ขาพอร์ตที่ต้องการอ่านค่า ซึ่งต้องเป็นขาพอร์ตดิจิตอล ทำให้มีค่าได้จาก 0 ถึง 13 หรือเป็นตัวแปรที่มีค่าอยู่ในช่วง 0 ถึง 13 ก็ได้

ค่าที่ส่งกลับ

เป็น HIGH หรือ LOW

ตัวอย่างที่ 4.3

```
Int ledPin = 13;                / / LED connected to Di pin 13
int inPin = 7;                  / / pushbutton connected to digital pin 7
int val = 0;                    / / variable to store the read value
```

```
void setup ()
        {
        pinMode (ledPin, OUTPUT);          / / sets Di 31 as output
        pinMode (inPin, INPUT);            / / sets Di 7 as input
        {
Void loop ()
        {
        val = digitalRead(inPin)           / / read input pin
        digitalWrite(ledPin, val);         / / seta LED to the button's value
        {
```

กำหนดให้ขา 7 เป็นอินพุต สถานะของ LED ที่ขา 13 จะเปลี่ยนแปลงตามสถานะของอินพุตขา 7

4.1.4 การกำหนดโหมดของขาพอร์ต

ก่อนใช้งานต้องกำหนดโหมดการทำงานของขาพอร์ตดิจิตอล ให้เป็นอินพุตหรือเอาต์พุตกำหนดจากฟังก์ชั่น pinMode() มีรูปแบบดังนี้

pinmode(pin,mode);

เมื่อ pin คือ หมายเลขขาที่ต้องการ

Mode คือ โหมดการทำงาน (INPUT หรือ OUTPUT)

หลังจากที่กำหนดให้เป็นเอาต์พุตแล้วเมื่อต้องการเขียนค่าไปยังขานั้นๆ ให้เรียกใช้ฟังก์ชั่น digitalWrite() โดยมีรูปแบบดังนี้

digitalWrite(pin,value);

เมื่อ pin คือหมายเลขขาที่ต้องการ

value สถานะลอจิกที่ต้องการ (HIGH หรื อ LOW)

4.1.5 โปรแกรมสั่งให้ LED กะพริบ

ในการทดลองเกี่ยวกับไมโครคอนโทรลเลอร์ เรื่องแรกก็คือการสั่ งให้พอร์ตทำงานเป็นเอาต์พุต และสั่งให้มีค่าเป็น HIGH หรือ LOW ได้ตามที่ต้ องการ โดยจะต่อกับ LED และสั่งให้ LED ติดดับต่อเนื่องกันตลอดเวลาเรียกว่าไฟกะพริบ ในการทดลองขับ LED อย่างง่ายได้ยกตัวอย่างไฟล์ Blink ซึ่งมีตัวอย่างในโปรแกรมArduino IDE อยู่แล้ว

ในตัวอย่างนี้จะนำโปรแกรม Blink.ino มาประยุกต์สั่งเอาต์พุตควบคุม LED สองดวงให้ติดดับสลับกัน เริ่มต้นด้วยการต่อวงจรส่วนของ LED จะต่อวงจรให้ LED ทำงานที่ลอจิก 1 คือเมื่อสั่งให้ขาเป็น HIGH จะทำให้ LED ติด เมื่อสั่งให้ขาเป็น LOW หลอดจะดับ ในการต่อ LED สำหรับบอร์ดสามารถใช้แผงวงจรต่อตรงกับจุดต่อของพอร์ตได้ทันที เมื่อต่ออุปกรณ์แล้วให้เขียนโปรแกรมตามโปรแกรมที่ 4.1 ทดลองคอมไพล์และอัปโหลดลงบอร์ด ศึกษาผลการทำงานวงจรทดลองดจิติอลเอาต์พุตสั่งให LED 2 ดวงกะพริบสลับกัน

โปรแกรมที่ 4.1

ไฟล์ TwoLED_Blink.ino โปรแกรมภาษา C ของ Arduino ควบคุมให้บอร์ดขับ LED 2 ดวงกะพริบสลับกัน

```
/* Basic code for turn on LED1 and turn off LED2 for 1 second,
* then off LED1 and on LED2 for one second, and so on...
* File : TwoLED_Blink.ino
*/
        #define LED1_PIN 11          // LED1 connected to digital pin 11
        #define LED2_PIN 13          // LED2 connected to digital pin 13
void setup() { // Run once at startup
        pinMode(LED1_PIN, OUTPUT); // Call function pinMode to set Di 11 as OUTPUT
        pinMode(LED2_PIN, OUTPUT); // Call function pinMode to set Di 13 as OUTPUT
        }
void loop() { // run over and over again
        digitalWrite(LED1_PIN, HIGH); // Turn on LED1
        digitalWrite(LED2_PIN, LOW); // Turn off LED2
        delay(1000); // wait 1 second (1000 milisecond)
        digitalWrite(LED1_PIN, LOW); // Turn off LED1
        digitalWrite(LED2_PIN, HIGH); // Turn on LED2
```

ในโปรแกรมตัวอย่างที่ 4.1 บรรทัด #define LED1_PIN 11 คือการกำหนดค่าคงที่ให้ข้อความ LED1_PIN มีค่าเท่ากับ 11 หลังจากบรรทัดนี้ ในโปรแกรมเมื่อพบข้อความ LED1_PIN ให้นำค่า 11 ไปแทนที่แล้วจึงคอมไพล์โปรแกรม

เมื่อโปรแกรมทำงานได้แล้ว ทดลองแก้ไขโปรแกรมเพื่อเปลี่ยนตำแหน่งขาเอาต์พุตที่ต่อกับ LED โดยเปลี่ยนค่าตัวเลขของบรรทัด #define LED1_PIN 11 เป็นค่าอื่นๆ ระหว่าง 1 ถึง 13 บรรทัด delay(1000); เป็นการเรียกใช้ฟังก์ชั่น delay() โดยส่งค่า 1000 ให้กับฟังก์ชั่นเพื่อให้เกิดการหน่วงเวลา 1000 มิลลิวินาที (ms) หรือ 1 วินาที ถ้าต้องการปรับให LED กะพริบเร็วขึ้นหรือช้าลง สามารถเปลี่ยนค่าในวงเล็บเป็นค่าอื่นๆ ได้โดยค่ายิ่งมาก LED ยิ่งกะพริบช้าลง

4.1.6 โปรแกรมไฟวิ่ง LED 4 ดวง

ในตัวอย่างนี้จะนำบอร์ด Arduino มาต่อควบคุม LED จำนวน 4 ตัว โดยสั่งให้ LED ติดตามลำดับเริ่มจากLED1 ไปยัง LED4 แล้ววนกลับมาเริ่มที่ LED1 ต่อเนื่องตลอดเวลา

โปรแกรมที่ 4.2

```
/*
* Code for turn on and off LED1, LED2, LED3, LED4, and so on..
* File : FourLED_Moving.ino
*/
```

ไฟล์ FourLED_Moving.ino โปรแกรมภาษา C/C++ ของ Arduino เพื่อควบคุมไฟวิ่ง LED 4 ดวง

```
#define LED1_PIN 10 // LED1 connected to digital pin 10
#define LED2_PIN 11 // LED2 connected to digital pin 11
#define LED3_PIN 12 // LED3 connected to digital pin 12
#define LED4_PIN 13 // LED4 connected to digital pin 13
void setup() // Run once at startup
{
pinMode(LED1_PIN, OUTPUT); // Set Digital pin 14 as OUTPUT
pinMode(LED2_PIN, OUTPUT); // Set Digital pin 15 as OUTPUT
pinMode(LED3_PIN, OUTPUT); // Set Digital pin 16 as OUTPUT
pinMode(LED4_PIN, OUTPUT); // Set Digital pin 17 as OUTPUT
}
void loop() // run over and over again
{
digitalWrite(LED1_PIN, HIGH); // Turn on LED1
delay(200); // wait for a 0.2 second. (200 ms)
digitalWrite(LED1_PIN, LOW); // Turn off LED1
delay(200); // wait for a 0.2 second. (200 ms)
digitalWrite(LED2_PIN, HIGH); // Turn on LED2
delay(200); // wait for a 0.2 second. (200 ms)
digitalWrite(LED2_PIN, LOW); // Turn off LED2
delay(200); // wait for a 0.2 second. (200 ms)
digitalWrite(LED3_PIN, HIGH); // Turn on LED3
delay(200); // wait for a 0.2 second. (200 ms)
digitalWrite(LED3_PIN, LOW); // Turn off LED3
delay(200); // wait for a 0.2 second. (200 ms)
digitalWrite(LED4_PIN, HIGH); // Turn on LED1
```

```
delay(200); // wait for a 0.2 second. (200 ms)
digitalWrite(LED4_PIN, LOW); // Turn off LED1
delay(200); // wait for a 0.2 second. (200 ms)
}
```

การทำงานของโปรแกรมนี้เริ่มด้วยการสั่งให้ LED1 ติด 0.2 วินาทีดับ 0.2 วินาที แล้วสั่งให้ LED2 ติด 0.2 วินาที ดับ 0.2 วินาที ตามด้วย LED3 และ LED4 ติดและดับเป็นลำดับวนต่อเนื่องตลอดเวลา จากโปรแกรมสั่งให้ LED ติดตามลำดับดังโปรแกรมที่ 4.2 นำมาเขียนใหม่โดยใช้ตัวแปรอะเรย์จะได้เป็น

โปรแกรมที่ 4.3

ผลการทำงานเหมือนกันแต่โปรแกรมที่ 4.3 จะกระชับและเมื่อคอมไพล์ แล้วได้ไฟล์ภาษาเครื่องที่มีขนาดเล็กกว่า

```
/*
* Code for turn on and off LED1, LED2, LED3, LED4, and so on..
* Same as MovingLED1 but used array.
* File : MovingLED_Array.ino
*/
#define DELAY_TIME 200
int led_pin[ ]={10,11,12,13};
int count;
void setup() // Run once at startup
{
for(count=0; count<4; count++)
pinMode(led_pin[count], OUTPUT);
// Call function pinMode to set Digital pin 14,15,16,17 as OUTPUT
}
void loop() // run over and over again
{
for(count=0; count<4; count++)
{
digitalWrite(led_pin[count], HIGH); // Turn on LED
delay(DELAY_TIME); // wait for a 0.2 second. (200 ms)
```

```
digitalWrite(led_pin[count], LOW); // Turn off LED
delay(DELAY_TIME);
}
}
```

โปรแกรมที่ 4.3 ไฟล์ MovingLED_Array.ino โปรแกรมภาษา C/C++ ของ Arduino เพื่อควบคุมไฟวิ่ง LED แบบใช้ตัวแปรอะเรย์

4.1.7 การทดลองอินพุตดิจิตอลของ Arduino Uno

คุณสมบัติของขาพอร์ตอินพุต

ขาพอร์ตของ Arduino Uno จะถูกกำหนดเป็นอินพุตตั้งแต่เริ่มต้น จึงไม่จำเป็นต้องใช้ฟังก์ชั่น pinMode () ในการกำหนดให้เป็นอินพุต ขาพอร์ตที่ถูกกำหนดเป็นอินพุตจะมีสถานะเป็นอิมพีแดนซ์สูง ทำให้มีความต้องการกระแสไฟฟ้าจากอุปกรณ์ที่ต้องการอ่านค่าอินพุตน้อยมาก ทำให้ไม่สามารถรับหรือจ่ายกระแสให้กับวงจรภายนอก ทำให้เขาที่เป็นอินพุตนี้ไปใช้งานบางประเภท เช่นสร้างตัวตรวจจับการสัมผัสที่อาศัยการวัดค่าความจุไฟฟ้า

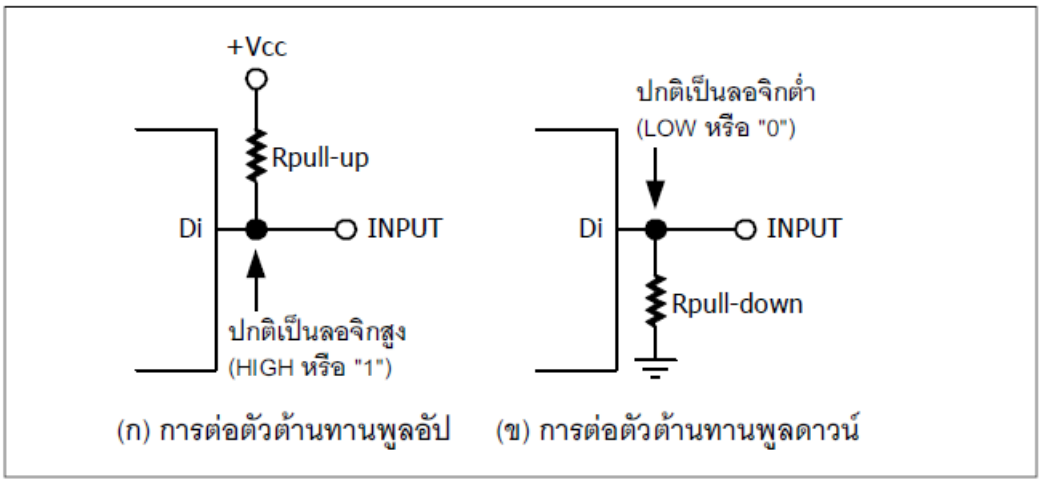
สำหรับขาอินพุต เมื่อไม่มีอินพุตป้อนจะต้องกำหนดค่าแรงดันให้แน่นอน ทำได้โดยต่อตัวต้านทานพลูอัป (Pull-up Resistor) โดยต่อขาของตัวต้านทานขาหนึ่งไปยังไฟเลี้ยง หรือต่อพลูดาวน์ (Pull-down) ซึ่งต่อขาหนึ่งของตัวต้านทานจากขาพอร์ตลงกราวด์ ค่าตัวต้านทานที่ใช้ทั่วไปคือ 10kΩ ดังรูปที่ 4.1

Arduino Uno มีขาพอร์ตดิจิตอลที่กำหนดให้เป็นอินพุตหรือเอาต์พุตจำนวน 13 ขา ถ้าต้องการกำหนดเป็นอินพุตต้องกำหนดด้วยฟังก์ชั่น pinMode และอ่านค่าอินพุตได้จากฟังก์ชั่น digitalRead ซึ่งมีรูปแบบดังนี้

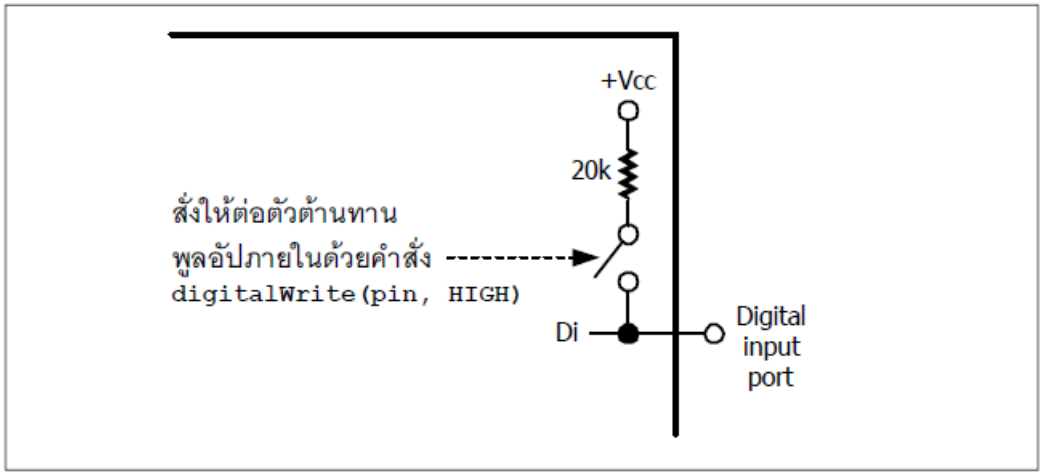
digitalRead (pin);

เมื่อ pin คือหมายเลขขาที่ต้องการอ่านค่าสถานะ

เมื่อฟังก์ชั่นทำงานค่าเป็น LOW (ค่าเป็น “0”) หรือ HIGH (ค่าเป็น “1”)



รูปที่ 4.1 แสดงการต่อตัวต้านทานเพื่อกำหนดสภาวะของขาพอร์ตอินพุตในขณะที่ยังไม่มีอินพุตส่งเข้ามา



รูปที่ 4.2 แสดงการต่อตัวท้านทานพลูอัปภายในที่ขาพอร์ตอินพุตดิจิตอลซึ่งควบคุมได้ด้วยกระบวนการทางซอฟต์แวร์

ภายในขาพอร์ตของไมโครคอนโทรลเลอร์ ATmega ซึ่งเป็นไมโครคอนโทรลเลอร์หลักการ Arduino Uno จะมีการต่อตัวต้านทานพลูอัปค่า 20kΩ เตรียมไว้ให้ ซึ่งสามารถสั่งต่อใช้งานผ่านทางซอฟต์แวร์ ดังในรูปที่ 4.2 สำหรับตัวอย่างโปรแกรมเพื่อใช้งานที่มีดังนี้

ตัวอย่างที่ 4.4

```
pinMode (pin, INPUT);          // set pin to input
digitalWrite (pin, HIGH);      // turn on pullup resistors
```

4.1.8 การรับค่าสวิตซ์อย่างง่าย

ในการทดลองนี้จะทดลองอ่านค่าสถานะของสวิตช์แบบกดติดปล่อยดับ เพื่อควบคุมหลอด LED เมื่อกดสวิตช์ S1 ทำให้ LED ติดสว่าง เมื่อปล่อยสวิตช์ LED1 จะดับและเขียนเป็นโปรแกรมได้ดังนี้

โปรแกรมที่ 4.4

ในการใช้งานขาอินพุตดิจิตอลต้องต่อตัวต้านทานพลูอัป (ต่อตัวต้านทานจากไฟเลี้ยง +5V มายังขาอินพุต) เพื่อกำหนดสถานะที่แน่นอนให้กับขาอินพุตในภาวะที่ไม่มีการกดสวิตช์ ดังวงจรในรูปที่ 4.5 โดยต่อขา 7 ผ่านตัวต้านทานค่า 10kΩ ไปยังไฟเลี้ยง +5V เมื่อไม่ได้กดสวิตช์ SW1 ที่ขา 7 จะมีสถานะเป็นลอจิกสูง HIGH หรือ “1” เมื่อกดสวิตช์จะทำให้ขา 7 ต่อลงกราวด์ อ่านค่าสถานะเป็นลอจิกต่ำ LOW หรือ “0”

```
/*
*  Read input from push button for control status of LED.
*  Modify from button (http://www.arduino.cc/en/Tutorial/button)
* File : Button_LED. Ino
```

```
        */
        #define LED_PIN 11                  // choose the pin for the LED
        #define IN_PIN 7                    // choose the input pin (for a pushbutton)
        int val = 0;
void setup ()
        {
        pinMode (LED_PIN, OUTPUT);          // declare LED as output
        pinMode (IN_PIN, INPUT);            // declare pushbutton as input
        }
void loop ()
        {
        val = digitalRead (IN_PIN);         // read input value
 if (val == LOW)                            // check the input as LOW (button pushed)
        {
        digitalWrite (LED_PIN, HIGH);       // turn LED ON
        }
else
        {
        digitalWrite (LED_PIN, LOW);        // turn LED OFF
        }
```

โปรแกรมที่ 4.4 ไฟล์ button_LED.ino โปรแกรมภาษา C ของ Arduino สำหรับอ่านค่าอินพุตจากสวิตช์แบบปุ่มกดเพื่อควบคุม LED

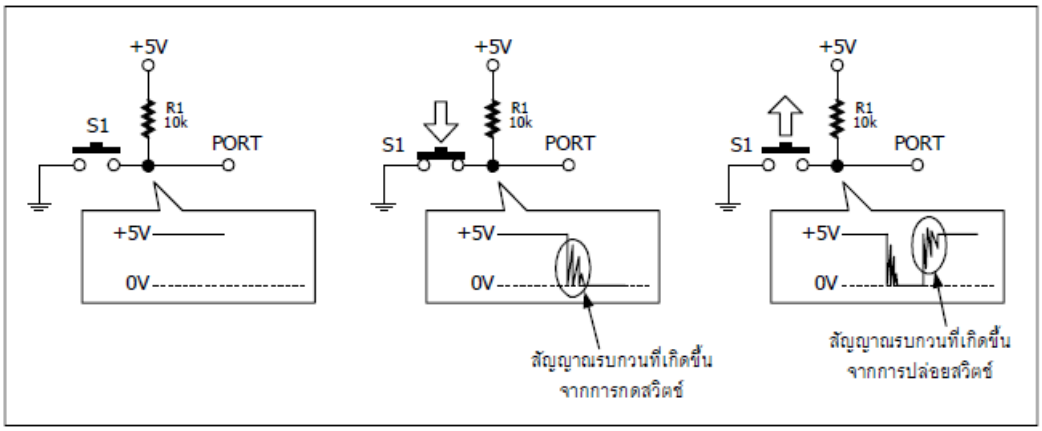
การทำงานของ LED1 จะตรงข้ามกับสถานะของสวิตช์ คือเมื่อไม่กดสวิตช์ จะอ่านสถานะของขา 7 ได้ลอจิกสูง จึงต้องสั่งให้ขา 11 เป็นลอจิกต่ำหรือ “0” เพื่อทำให้ LED1 ดับ เมื่อกดสวิตช์ อ่านค่าสถานะของขา 7 ได้ลอจิก “0” ต้องสั่งให้ขา 11 เป็น “1” เพื่อขับ LED1 ติดสว่าง

เมื่อโปรแกรมทำงานได้ผลตามที่ต้องการแล้วให้ทดลองตัวต้านทานพลูอัปค่า 10kΩ ออก เพื่อให้ขา 7 ลอย สังเกตกรณีนี้ เมื่อยังไม่มีการกดสวิตช์ LED1 อาจติดกะพริบด้วยความเร็วสูง เห็น LED สว่างเรื่อยๆ เนื่องจากสถานะของขา Di7 เป็น “0” และ “1” สลับกันไม่แน่นอน

4.1.9 โปรแกรมแก้ปัญหาสัญญาณรบกวนในการกดสวิตช์โดยใช้ซอฟต์แวร์

ในหัวข้อนี้จะทดลองเขียนโปรแกรมรับค่าของสวิตช์ ซึ่งเป็นแบบกดติดปล่อยดับ ให้มีการทำงานเป็นแบบกดติดกดดับ คือเมื่อเริ่มต้นโปรมแกรม LED ดับอยู่ เมื่อกดสวิตช์ LED จะติดสว่างเมื่อกดสวิตช์อีกครั้ง LED จะดับสลับกันไปมาตลอดเวลา

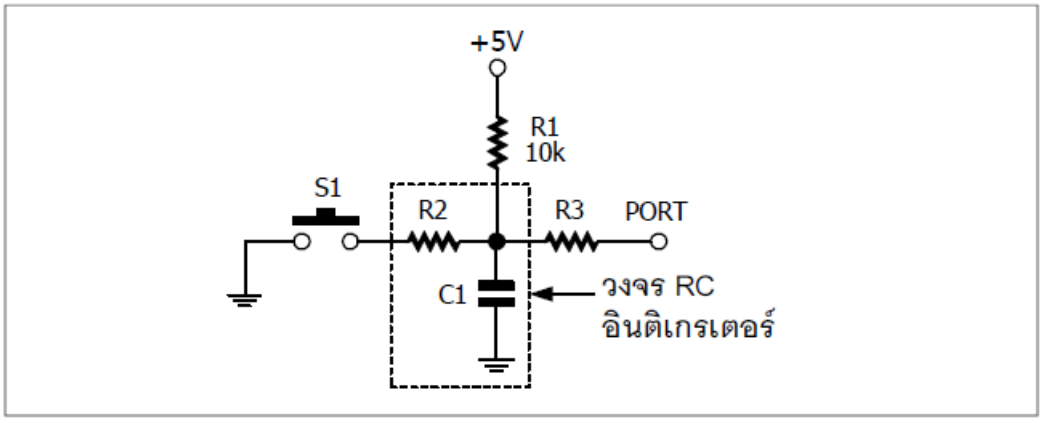
โดยทั่วไปแล้ว สวิตช์ที่ใช้จะเป็นสวิตช์ทางกลที่ประกอบด้วยหน้าสัมผัสโลหะ ในการกดสวิตช์ให้ต่อวงจรพบว่าหน้าสัมผัสของสวิตช์จะไม่สัมผัสกันสนิททันที โดยมีช่วงเวลาที่เริ่มสัมผัส และหลุดเป็นช่วงเวลาสั้นๆ ก่อนที่หน้าสัมผัสของสวิตช์จะต่ออย่างสมบรูณ์ และเมื่อวัดสัญญาณที่ได้จากสวิตช์พบว่าระดับสัญญาณมีการสั่น



รูปที่ 4.3 แสดงการเกิดสัญญาณรบกวนเมื่อมีการกดและปล่อยสวิตช์ในวงจรดิจิตอล

การสั่นที่เรียกว่าเบาซ์ (Bounce) อยู่ชั่วขณะดังแสดงในรูปที่ 4.3 โดยระยะเวลาที่สัญญาณเกิดการเบาวซ์นี้มีระยะเวลาตั้งแต่ไม่กี่มิลลิวินาที (ms) ไปจนถึงหลายสิบมิลลิวินาที ขึ้นกับประเภทของสวิตช์ที่ใช้

การแก้ปัญหาที่ระดับสัญญาณเกิดการสั่นนี้เรียกว่าการดีเบาซ์ (Debounce) หลักการแก้ไขสัญญาณรบกวนแบบบนคือ หน่วงเวลาการเกิดขึ้นของสัญญาณพัลส์เล็กน้อย เพื่อให้วงจรไม่สนใจสัญญาณที่เกิดขึ้นในช่วงเริ่มต้นกดสวิตช์ ซึ่งทำได้หลายวิธี เช่นวิธีการแรกทำได้โดยใช้อุปกรณ์อิเล็กทรอนิกส์พื้นฐาน อย่างตัวต้านทานและตัวเก็บประจุ โดยต่อกันในลักษณะวงจร RC อินติเกรเตอร์ดังในรูปที่ 4.4 ด้วยวิธีการนี้จะช่วยลดผลของสัญญาณรบกวนที่เกิดขึ้นจากการกดสวิตช์ได้ในระดับหนึ่ง โดยประสิทธิภาพของวงจรจะขึ้นกับการเลือกค่าของตัวต้านทานและตัวเก็บประจุ หากเลือกค่าของตัวเก็บประจุน้อยเกินไป อาจไม่สามารถลดสัญญาณรบกวนได้ แต่ถ้าเลือกค่ามากเกินไปจะทำให้ความไวในการตรวจจับการกดสวิตช์ลดลง นั่นคืออาจต้องกดสวิตช์มากกว่า 1 ครั้งเพื่อให้ได้สัญญาณที่ต้องการ



รูปที่ 4.4 การต่อวงจร RC อินติเกรเตอร์ เพื่อแก้ไขปัญหาสัญญาณรบกวนจากการกดสวิตช์

หรือถ้าแก้โดยใช้ไมโครคอนโทรลเลอร์ จะสามารถแก้ไขได้ด้วยกระบวนการทางซอฟต์แวร์ โดยเขียนโปรแกรมให้ตรวจจับการเปลี่ยนแปลงระดับลอจิกของสวิตช์ ถ้าพบว่ามีการเริ่มกดสวิตช์ ก็คือระดับลอจิกเปลี่ยนจาก “1” เป็น “0” โปรแกรมจะหน่วงเวลารอจนเลยช่วงเวลาเกิดเบาซ์ของสวิตช์ เมื่อเลยเวลาแล้วให้ทำการอ่านค่าสถานะของสวิตช์อีกครั้งถ้ายังคงเป็น “0” แสดงว่ากดสวิตช์สมบูรณ์แล้ว

ระยะเวลาที่ต้องหน่วงเวลาหาได้จากการใช้ออสซิลโลสโคปวัดระดับสัญญาณ หรือใช้การทดลองป้อนค่าหน่วงเวลาแล้วปรับค่าจนกระทั่งได้ค่าที่แก้ปัญหาการเบาวซ์ของสวิตช์ได้อย่างสมบูรณ์ ในโปรแกรมทดลองที่ 4.5 ได้ทดลองโดยใช้ค่าของการหน่วงเวลาเท่ากับ 10 มิลลิวินาที ซึ่งสามารถเปลี่ยนได้ที่บรรทัด

#define Debounce 10

โปรแกรมที่ 4.5

ไฟล์ DebounceSW.ino โปรแกรมภาษา C ของ Arduino อ่านค่าสวิตช์เพื่อควบคุม LED แบบมีการแก้สัญญาณรบกวนจากการกดสวิตช์หรือดีเบาซ์

```
/*
* Read input from push button for control status of LED.
* Change operation of push button to togle switch.
* File : DebouncedSW.ino */
#define IN_PIN 7 // the number of the input pin
#define OUT_PIN 11 // the number of the output pin
#define Debounce 10 // debounce time = 10 ms
int state = HIGH; // the current state of the output pin
int reading; // the current reading from the input pin
int previous = HIGH; // the previous reading from the input pin
```

```
void setup()
        {
        pinMode(IN_PIN, INPUT);
        pinMode(OUT_PIN, OUTPUT);
        digitalWrite(OUT_PIN, state);
        }
void loop()
        {
        reading = digitalRead(IN_PIN);
        // if we just pressed the button (i.e. the input went from HIGH to LOW.
if (reading == LOW && previous == HIGH)
        {
        delay(Debounce); // wait for decounce
if(digitalRead(IN_PIN) == LOW) // if the input remain LOW
        state = !state; // invert the state of output LED
        digitalWrite(OUT_PIN, state);
        } previous = reading;
```

4.2 ฟังก์ชั่นเกี่ยวกับการสื่อสารผ่านพอร์ตอนุกรม

ใช้สำหรับสื่อสารข้อมูลระหว่างฮาร์ดแวร์ Arduino กับคอมพิวเตอร์ หรืออุปกรณ์อื่นๆ โดยจะแบ่งพอร์ตสำหรับเชื่อมต่อออกเป็น 2 ส่วนคือ ส่วนแรกติดต่อพอร์ตอนุกรมเสมือน (Virtual Com Port) จากการทำงานของส่วนเชื่อมต่อพอร์ต USB ฟังก์ชั่นที่ใช้คือ Serial

อีกส่วนหนึ่งคือ ขาพอร์ตสื่อสารข้อมูลอนุกรมโดยใช้ขา0 (RxD) และ 1 (TxD) ฟังก์ชั่นของ Arduino ที่ใช้คือ Serial1 ดังนั้นเมื่อเลือกใช้งานเป็นขาพอร์ตสื่อสารข้อมูลอนุกรมแล้วจะไม่สามารถใช้ขาพอร์ต 0 และ 1 เป็นพอร์ตดิจิตอลได้

4.2.1 Serial .begin (int datarate)

กำหนดค่าอัตราบอดของการรับส่งข้อมูลอนุกรมในหน่วยบิตต่อวินาที (bits per second : bps) ใช้ค่าต่อไปนี้ 300, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 38400, 57600 หรือ 115200

พารามิเตอร์

Int datarate ในหน่วยบิตต่อวินาที (baud หรือ bps)

ตัวอย่างที่ 4.5

```
void setup ( )
        {
        Serial .begin (9600) ;     / / opens serial port, baudrate 9600 bps
        }
```

เป็นการเลือกอัตราบอดเท่ากับ 9600 บิตต่อวินาที

4.2.2 Serial .available ( )

ใช้แจ้งว่าได้รับข้อมูลตัวอักษร (characters) แล้ว และพร้อมสำหรับการอ่านไปใช้งาน

ค่าที่ส่งกลับจากฟังก์ชั่น

จำนวนไบต์ที่พร้อมสำหรับการอ่านค่า โดยเก็บข้อมูลในบัฟเฟอร์ตัวรับ ถ้าไม่มีข้อมูลจะมีค่าเป็น 0 ถ้ามีข้อมูลฟังก์ชั่นจะคืนค่าที่มากว่า 0 โดยบัฟเฟอร์สามารถเก็บข้อมูลได้สูงสุด 128 ไบต์

ตัวอย่างที่ 4.6

```
        int incomingByte = 0 ; / / for incoming serial data
void setup ( )
        {
        Serial . begin (9600) ;  / / opens serial port, baud rate 9600 bps
        delay (5000) ;
        }
void loop ( )
        {
if (Serial .available ( ) > 0)       / /  send data only when you receive data :
        {
        incomingByte = Serial .read ( ) ; / / read the incoming byte :
        Serial .print (“I received : “) ; / / say what you got :
        Serial .println (incomingByte, DEC) ;
        }
        }
```

ในตัวอย่างนี้ ใช้อัตราบอด 9,600 บิตต่อวินาที เมื่อรันโปรแกรมจะต้องเปิดหน้าต่าง Serial Monitor เพื่อป้อนข้อมูลมายังบอร์ด Arduino Uno ด้วย ถ้ามีข้อมูลเข้ามาจะเก็บไว้ในตัวแปร incomingByte แล้วนำไปแสดงที่หน้าต่าง Serial Monitor โดยต่อท้ายข้อความ I received : ค่าที่แสดงจะเป็นค่าข้อมูลในรูปของเลขฐานสิบ

ยกตัวอย่างหากป้อนเลข 2 เข้ามาจะแสดงข้อความ I received : 50 เนื่องจากรหัสแอสกี้ของ 2 คือ 32 ฐานสิบหก เท่ากับ 50 ฐานสิบ

4.2.3 Serial .read ( )

ใช้อ่านค่าข้อมูลที่ได้รับจากพอร์ตอนุกรม

ค่าที่ส่งกลับจากฟังก์ชั่น

เป็นเลข int ที่เป็นไบต์แรกของข้อมูลที่ได้รับ (หรือเป็น -1 ถ้าไม่มีข้อมูล)

ตัวอย่างที่ 4.7

```
        int incomingByte = 0 ;          / / for incoming serial data
void setup ( )
        {
        Serial .begin (9600) ;          / / opens serial port, baud rate 9600 bps
        delay (5000) ;
        }
void loop ( )
        {
if (Serial .available ( ) > 0 )          / / send data only when you receive data ;
        {
        incomingByte = Serial .read ( ) ; / / read the incoming byte :
        Serial .print ( “I received : ”) :    / / say what you got :
        Serial .println (incomingByte, DEC ) ;
        }
        }
```

4.2.4 Serial .flush ( )

ใช้ล้างบัฟเฟอร์ตัวรับข้อมูลพอร์ตอนุกรมให้ว่าง

4.2.5 Serial .print (data)

ใช้ส่งข้อมูลออกทางพอร์ตอนุกรรม

พารามิเตอร์

Data – เป็นข้อมูลเลขจำนวนเต็มได้แก่ char, int หรือเลขทศนิยมที่ตัดเศษออกเป็นจำนวนเต็ม

รูปแบบฟังก์ชั่น

คำสั่งนี้สามารถเขียนได้หลายรูปแบบ

Serial .print (b) เป็นการเขียนคำสั่งแบบไม่ได้ระบุรูปแบบ จะพิมพ์ค่าตัวแปร b เป็นเลขฐานสิบ โดยพิมพ์ตัวอักษรรหัส ASCII

ตัวอย่างที่ 4.8

```
int b = 79 ;
Serial .print (b) ;
Serial .print (b, DEC)
```

เป็นคำสั่งพิมพ์ค่าตัวแปร b เป็นตัวเลขฐานสิบ โดยพิมพ์ตัวอักษรตามรหัส ASCII ดังตัวอย่าง

```
int b = 79 ;
Serial .print (b) ;
Serial .print (b, HEX)
```

เป็นคำสั่งพิมพ์ค่าแปร b เป็นตัวเลขฐานสิบหก โดยพิมพ์ตัวอักษรตามรหัส ASCII ดังตัวอย่าง

```
int b = 79 ;
Serial .print (b, HEX)
Serial .print (b, OCT) ;
```

เป็นคำสั่งพิมพ์ค่าตัวแปร b เป็นตัวเลขฐานแปด โดยพิมพ์ตัวอักษรตามรหัส ASCII ดังตัวอย่าง

```
int b = 79 ;
serial .print (b, OCT) ;
Serial .print (b, BIN)
```

เป็นคำสั่งพิมพ์ค่าตัวแปร b เป็นตัวเลขฐานสอง โดยพิมพ์ตัวอักษรตามรหัส ASCII ดังตัวอย่าง

```
int b = 79 ;
Serial .print (b, BIN) ;
Serial .print (b, BYTE)
```

เป็นคำสั่งพิมพ์ค่าตัวแปร b ขนาด 1 ไบต์ ดังตัวอย่าง

```
int b = 79 ;
Serial .print (b, BYTE) ;
Serial .print (str)
```

เป็นคำสั่งพิมพ์ค่าข้อความในวงเล็บ หรือข้อความที่เก็บในตัวแปร str ดังตัวอย่าง

```
Serial .print (“Hello World”) ;
```

พารามิเตอร์

b – ไบต์ข้อมูลที่ต้องการพิมพ์ออกทางพอร์ตอนุกรรม

str – ตัวแปรสตริงที่เก็บข้อความสำหรับส่งออกพอร์ตอนุกรม

ตัวอย่างที่ 4.9

```
/* Analog input reads an analog input on analog in 0 , prints the value out.
created by Tom Igoe */
int analogValue = 0 ;                    // variable to hold the analog value
void setup ( )
{
Serial .begin (9600) ;                   // open the serial port at 9600 bps :
delay (5000) ;
}
void loop ( )
{
  analogValue = analogRead (0) ;         // read the analog  input on pin 0 :
  Serial .print (analogValue) ;          // print it out in many formats :
 // print as an ASCII – encoded decimal
  Serial .print (“\ t”) ;                // print a tad character
  Serial .print (analogValue, DEC) ;     // print as an ASCII – encoded decimal
  Serial .print (“\ t”) ;                // print a tab character
  Serial .print (analogValue, HEX) ;     // print as an ASCII – encoded hexadecimal
  Serial .print (“\ t”) ;                // print a tab character
  Serial .print (analogValue, OCT) ;     // print as an ASCII – encoded octal
  Serial .print (“\ t”) ;                // print a tab character
  Serial .print (analogValue, BIN) ;     // print as an ASCLL – encoded binary
  Serial .print (“\ t”) ;                // print a tab character
  Serial .write (analogValue/4) ;
  // print as a raw byte value (divide the value by 4 because
  // analogRead ( ) returns numbers from 0 to 1023,
  // but a byte can only hold values up to 255)
```

```
Serial .print (“\ t”) ;          // print a tab character
Serial.println();                // print a linefeed character
Delay(10);                       // delay 10 milliseconds before the next reading
}
```

ตัวอย่างนี้แสดงการพิมพ์ข้อมูลจากฟังก์ชั่น Serial. print() และ Serial. write() ในรูปแบบต่างๆ แสดงผ่านทางหน้าต่าง Serial Monitor

เทคนิคสำหรับการเขียนโปรแกรม Serial. print()

จะตัดเศษเลขทศนิยมเหลือเป็นเลขจำนวนเต็ม ทางแก้ไขทางหนึ่งคือ คูณเลขทศนิยมด้วย 10, 100, 1000 ฯลฯ ขึ้นอยู่กับจำนวนหลักของเลขทศนิยม เพื่อแปลงเลขทศนิยมเป็นจำนวนเต็มก่อน แล้วจึงส่งออกพอร์ตอนุกรม จากนั้นที่ฝั่งภาครับให้ทำการหารค่าที่รับได้เพื่อแปลงกลับเป็นเลขทศนิยม

4.2.6 Serial.println(data)

เป็นฟังก์ชั่นพิมพ์ (หรือส่ง) ข้อมูลออกทางพอร์ตอนุกรมตามด้วยรหัส carriage return (รหัส ASCII หมายเลข 13 หรือ \r) และ linefeed (รหัส ASCII หมายเลข 10 หรือ \n) เพื่อให้เกิดการเลื่อนบรรดและขึ้นบรรทัดใหม่ หลังจากพิมพ์ข้อความมีรูปแบบเหมือนคำสั่ง Serial.print()

รูปแบบฟังก์ชั่น

Serial.println(b)

เป็นคำสั่งพิมพ์ข้อมูลแบบไม่ได้ระบุรูปแบบจะพิมพ์ ค่าตัวแปรเป็นเลขฐานสิบ ตามด้วยรหัสอักษร carriage return และ linefeed

Serial.println(b, DEC) เป็นคำสั่งพิมพ์ค่าตัวแปร b เป็นตัวเลขฐานสิบ ตามด้วยรหัสอักษร carriage return และ linefeed

Serial.println(b, HEX) เป็นคำสั่งพิมพ์ค่าตัวแปร b เป็นตัวเลขฐานสิบหก ตามด้วยรหัสอักษร carriage return และ linefeed

Serial.println(b, OCT) เป็นคำสั่งพิมพ์ค่าตัวแปร b เป็นตัวเลขฐานแปด ตามด้วยรหัสอักษร carriage return และ linefeed

Serial.println(b, BIN) เป็นคำสั่งพิมพ์ค่าตัวแปร b เป็นตัวเลขฐานสอง ตามด้วยรหัสอักษร carriage return และ linefeed

Serial.println(str) พิมพ์ค่าในวงเล็บหรือข้อความที่ เก็บในตัวแปร str ตามด้วยรหัสอักษร carriage return และ linefeed

Serial.println() เป็นคำสั่งพิมพ์รหัส carriage return และ linefeed

พารามิเตอร์

b - ไบต์ข้อมูลที่ต้องการพิมพ์ ออกทางพอร์ตอนุกรม

str - ตัวแปรสตริงที่เก็บข้อความสำหรับส่งออกพอร์ตอนุกรม

ตัวอย่างที่ 4.10

```
int analogValue = 0; // variable to hold the analog value
void setup()
{
Serial. begin(9600); // open the serial port at 9600 bps:
delay(5000);
}
void loop()
{
analogValue = analogRead(0); // read the analog input on pin 0:
Serial. println(analogValue); // print it out in many formats:
Serial. println(analogValue, DEC); // print as an ASCII-encoded decimal
Serial. println(analogValue, HEX); // print as an hexadecimal
Serial. println(analogValue, OCT); // print as an ASCII-encoded octal
Serial. println(analogValue, BIN); // print as an ASCII-encoded binary
delay(10);}
```

4.2.7 การทดลองใช้งาน UART เพื่อติดต่อกับคอมพิวเตอร์

บอร์ด Arduino ติดต่อกับคอมพิวเตอร์เพื่อสื่อสารข้อมูลอนุกรมผ่านพอร์ต USB โดยใช้พอร์ตอนุกรมเสมือนหรือ Virtual COM Port ที่เกิดขึ้นจากการทำงานของส่วนเชื่อมต่อพอร์ต USB ของไมโครคอนโทรลเลอร์ ATmega32U4 และไดรเวอร์ โดยปกติแล้วจะเชื่อมต่อพอร์ตอนุกรมเสมือนผ่านทางพอร์ต USB เพื่อติดต่อกับคอมพิวเตอร์ในการอัปโหลดโปรแกรมเป็นหลัก แต่นำมาใช้รับส่งข้อมูลจากโปรแกรมของผู้ใช้งานกับคอมพิวเตอร์ได้

4.2.8 ฟังก์ชั่นที่เกี่ยวข้องกับการรับส่งข้อมูลผ่านพอร์ตอนุกรม

ArduinoIDE มีฟังก์ชั่นเกี่ยวกับการรับส่งข้อมูลผ่านพอร์ตอนุกรมมาพร้อมใช้งาน ดังนี้

- Serial. begin(speed) ใช้กำหนดอัตราเร็วของการถ่ายทอดข้อมูลหรืออัตราบอดหรือบอดเรต
- Serial. available() ใช้ตรวจสอบว่ามีข้อมูลด้านรับหรือไม่โดยคืนค่าเป็น int ตามจำนวนไบต์ข้อมูลที่รับเข้า
- Serial. read(data) ใช้อ่านค่าข้อมูลจากพอร์ตอนุกรม
- Serial. write(data) ใช้เขียนข้อมูลไบต์ไปยังพอร์ตอนุกรม

- Serial. flush() ใช้ล้างบัฟเฟอร์ทั้งด้านรับและส่ง
- Serial. print(data) ใช้ส่งข้อมูลออกพอร์ตอนุกรม
- Serial. println(data) ใช้ส่งข้อมูลออกพอร์ตอนุกรมพร้อมกับขึ้นบรรทัดใหม่

4.2.9 โปรแกรมส่งข้อมูลออกพอร์ตอนุกรม

เริ่มต้นด้วยการใช้ฟังก์ชั่น Serial. begin() เพื่อสั่งเปิดพอร์ตอนุกรมและกำหนดอัตราถ่ายทอดข้อมูลที่ใช้ในการสื่อสารข้อมูลหรืออัตราบอด มีรูปแบบการเขียนโปรแกรมดังนี้

Serial. begin(speed);

เมื่อ speed คืออัตราบอดมีค่า 300, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 38400,57600 หรือ 115200 บิตต่อวินาที ปกติที่ใช้คือ 9600

หลังจากเปิดพอร์ต เมื่อต้องการส่งข้อมูลให้ใช้ฟังก์ชั่น Serial. print() หรื อ Serial. println() ฟังก์ชั่นทั้งสองตัวทำงานให้ผลคล้ายกัน ต่างกันเมื่อ Serial. println() ส่งข้อมลูแล้วจะขึ้นบรรทัดใหม่ให้อัตโนมัติ

ฟังก์ชั่น Serial. print() และ Serial. println() มีรูปแบบดังนี้

Serial. print(b,FORMAT); กับ Serial. println(b,FORMAT);

โดยที่ b คือค่าตัวแปรประเภทเลขจำนวนเต็มที่ต้องการส่งออกทางพอร์ตอนุกรม ถ้าไม่ระบุรูปแบบจะพิมพ์ออกเป็นรหัส ASCII ของค่าตัวแปร

FORMAT เป็นรูปแบบของการพิมพ์มี DEC (เลขฐานสิบ), HEX (เลขฐานสิบหก), OCT (เลขฐานแปด) และ BIN (เลขฐานสอง)

โปรแกรมที่ 4.6

ไฟล์ Serial01.ino โปรแกรมทดสอบการส่งข้อมลูออกพอร์ตอนุกรมเสมือนผ่านทางพอร์ต USB

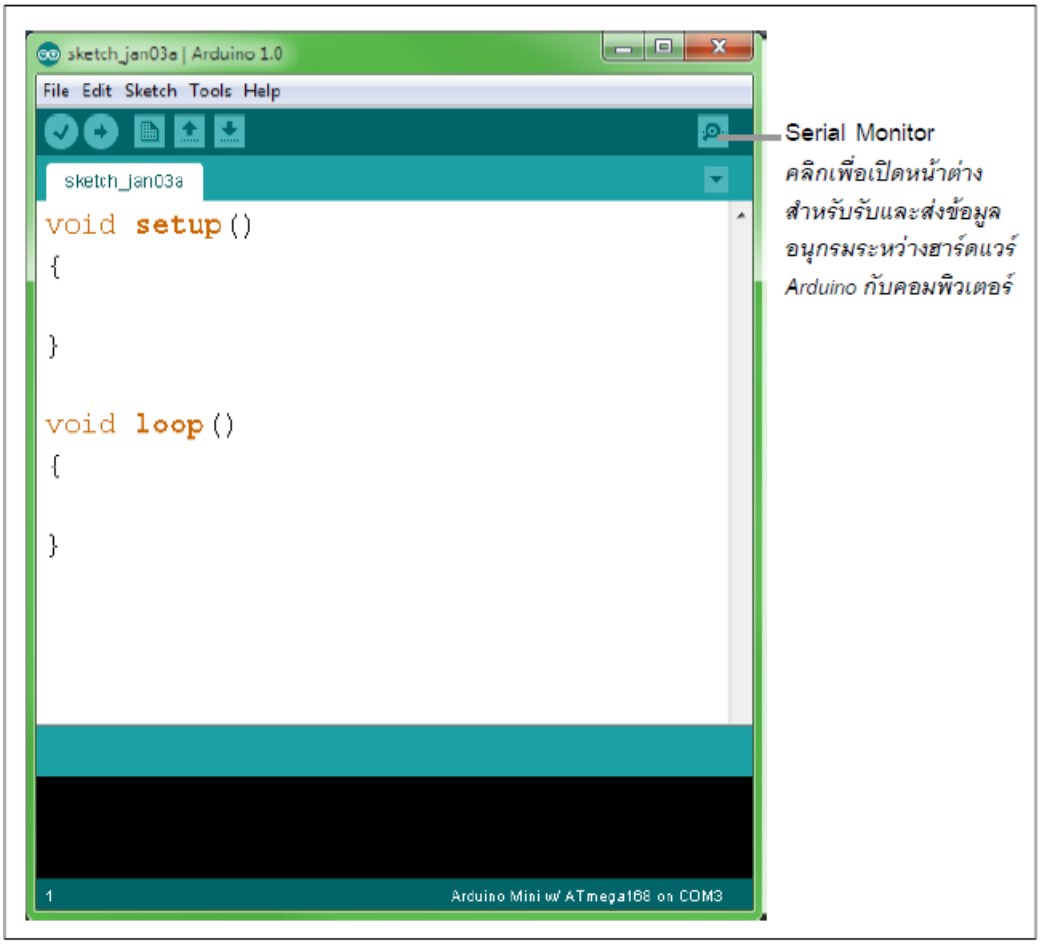
```
/* Code for send out data to serial port
* File : Serial01.pde
*/
void setup()
{
Serial. begin(9600); // Set serial port to 9600 bit per second
}
void loop()
{
int data = 2345; // print string to serial port
Serial. println("Welcome to Arduino Programming"); // print title
```

```
Serial. print(data, DEC); // print as an ASCII-encoded decimal
Serial. print("\t"); // print a tab character
Serial. print(data, HEX); // print as an ASCII-encoded hexadecimal
Serial. print("\t"); // print a tab character
Serial. print(data, OCT); // print as an ASCII-encoded octal
Serial. print("\t"); // print a tab character
Serial. print(data, BIN); // print as an ASCII-encoded binary
Serial. print("\t"); // print a tab character
Serial. write(data); // print as a raw byte value
Serial. println(); // print a line feed character
delay(1000); // Wait 1 second
}
```

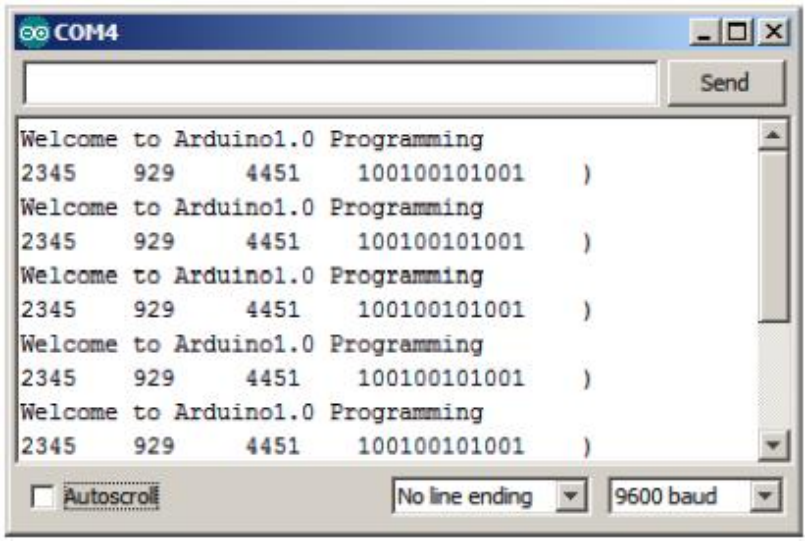
โปรแกรมที่ 4.6 เป็นโปรแกรมทดลองส่งข้อมูลออกพอร์ตอนกุรม โดยส่งข้อความและค่าของตัวแปรในการทดลองส่งค่าตัวแปร จะมีการกำหนดรูปแบบหรือ FORMAT ของคำสั่ง Serial. print() ไว้หลายๆแบบในการทดสอบการทำงานโปรแกรม Arduino IDE จะมีหน้าต่าง Serial Monitor เพื่ออำนวยความสะดวกในการรับและส่งข้อมูลผ่านพอร์ตอนุกรม ซึ่งจะใช้พื้นที่ร่วมกับพื้นที่แสดงข้อมูล (Text area) ซึ่งอยู่ทางด้านล่างของหน้าต่างโปรแกรม Arduino

การเปิดหน้าต่างนี้ทำได้โดยคลิกที่ปุ่ม Serial Monitor ดังแสดงในรูปที่ 4.7 ในรูปที่ 4.8 แสดงผลการทำงานของโปรแกรมที่ 4.6 หลังจากที่อัปโหลดโปรแกรมไปยังบอร์ด Arduino ในโปรแกรมกำหนดค่าตัวแปร data=2345 ซึ่งก็คือ

2,345 แปลงเป็นเลขฐาน 16 คือ 929 เป็นเลขฐาน 8 คือ 4451 เป็นเลขฐานสองคือ 100100101001 กรณีที่สั่งพิมพ์โดยใช้คำสั่ง Serial. write(data); จะส่งค่าเป็นเลขฐานสอง 8 บิตล่าง 00101001 ซึ่งก็คือ 41 ฐานสิบ เทียบเป็นรหัส ASCII คืออักขระ ที่หน้าต่างของ Serial Monitor จึงแสดงเป็นเครื่องหมาย) หรือวงเล็บปิดนั่นเอง



รูปที่ 4.7 แสดงการเลือกเปิดหน้าต่าง Serial Monitor



รูปที่ 4.8 ผลการทำงานของโปรแกรมที่ 4.6 เมื่อดูค่าผ่านทาง Serial Monitor

4.2.10 โปรแกรมรับค่าจากพอร์ตอนุกรมเพื่อกำหนดความเร็วในการกระพริบของ LED

ในการรับค่าจากพอร์ตอนุกรมจะใช้ฟังก์ชั่น 2 ตัวคือ Serial. avaliable() และ Serial. read() โดยเริ่มจากใช้ฟังก์ชั่น Serial. avaliable() เพื่อตรวจสอบว่ามีข้อมูลหรือไม่ ฟังก์ชั่นจะคืนค่าเป็นเลขจำนวนเต็ม แสดงจำนวนข้อมูลในบัฟเฟอร์ตัวรับของพอร์ตอนุกรม ถ้าอ่านค่าได้เท่ากับ 0 แสดงว่าไม่มีข้อมูล

เมื่อทดสอบพบว่าฟังก์ชั่น Serial. avaliable() คืนค่าไม่เท่ากับ 0 ถัดมาให้ใช้ฟังก์ชั่น Serial. read() เพื่ออ่านค่าจากบัฟเฟอร์ ตัวรับฟังก์ชั่นคืนค่าเป็นเลขจำนวนเต็มที่เป็นไบต์แรกของข้อมูล (หรือเป็น -1 ถ้าไม่มีข้อมูล)

ตัวอย่างการรับค่าจากพอร์ตอนุกรม เพื่อนำค่าที่รับได้ไปควบคุมอัตราการกระพริบของ LED มีโปรแกรมดังแสดงในโปรแกรมที่ 4.7 มีส่วนของโปรแกรมที่ควรทราบอยู่แห่งหนึ่งคือ หากผู้พัฒนาโปรแกรมต้องการให้มีการแสดงข้อความบนหน้าต่าง Serial monitor ของ Arduino1.0 ในทุกครั้งที่เริ่มต้นทำงานใหม่ จะต้องหน่วงเวลารอให้วงจร USB ภายใน Aธmega16U2 ของ Arduino เตรียมความพร้อมในการทำงาน หรืออีนัมเมอเรชั่นให้เสร็จสมบูรณ์เสียก่อนด้วยการแทรกคำสั่ง delay(5000); ก่อนใช้ คำสั่ง Serial. print();

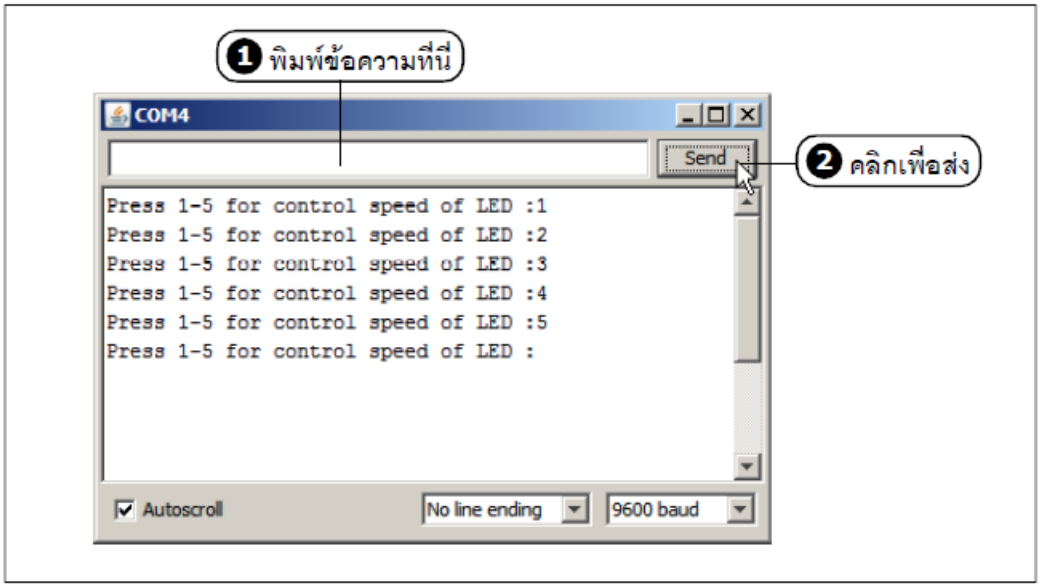
โปรแกรมที่ 4.7

ไฟล์ Serial02.ino โปรแกรมภาษา C ของ Arduino สำหรับทดสอบการรับค่าจากพอร์ตอนุกรมเพื่อกำหนดความเร็วในการกะพริบของ LED

```
/*
* Code for blinking LED that is received data from serial port
* to control rate of blinking. Receive keyboard button 1-5 only.
* File : Serial02.pde
*/
#define LED1_PIN 11 // LED pin as pin 11
int incomingByte = 0; // for incoming serial data.
int delayTime = 100; // Initial value of delay time.
void setup()
{
pinMode(LED1_PIN, OUTPUT); // set pin 11 as OUTPUT
Serial.begin(9600); // opens serial port at 9600 bps
delay(5000); // Delay for USB enumeration
Serial.print("Press 1-5 for control speed of LED :");
}
```

```
void loop()
        {
if (Serial.available() > 0)
        {
        incomingByte = Serial.read(); // read the incoming byte:
        // ASCII '1'=49, '2'=50, '3'=51, '4'=52 and '5'=53
        // calculate new delay time
If (incomingByte >= 49 && incomingByte <=53)
        { // echo to user
        Serial.write(incomingByte);
        Serial.println();
        delayTime=(incomingByte-48)*100;
        }
        Serial.print("Press 1-5 for control speed of LED :");
        }
        digitalWrite(LED1_PIN, HIGH); // Turn on LED1
        delay(delayTime); // wait for delayTime
        digitalWrite(LED1_PIN, LOW); // Turn off LED1
        delay(delayTime); // wait for delayTime
```

ผลการทำงานของโปรแกรมที่ 4.7 แสดงได้ดังรูปที่ 4.8 โปรแกรมนี้จะพิมพ์ข้อความ Press 1-5 for control speed of LED ออกทางพอร์ตอนุกรมแล้วรอให้ผู้ทดลองกดปุ่ม 1 ถึง 5 ที่แป้นคีย์บอร์ด โดยกด 1 คือให้ LED ติด 0.1 วินาที และดับ 0.1 วินาที และถ้ากด 5 จะทำให้ LED ติด 0.5 วินาที และดับ 0.5 วินาที ถ้าไม่ใช่ปุ่ม 1 ถึง 5 โปรแกรมจะไม่ตอบสนอง เมื่อกดแล้วโปรแกรมจะส่งรหัส ASCII ของปุ่มนี้กลับคืนเพื่อแจ้งผู้ทดลอง แล้วนำค่าที่ได้ไปคำนวณค่าหน่วงเวลาการติดดับของ LED



รูปที่ 4.8 ผลการทำงานของโปรแกรมที่ 4.7

ที่หน้าต่าง Serial Monitor ของ Arduino เมื่อต้องการส่งข้อความให้พิมพ์ข้อความที่ต้องการในช่องว่างด้านบน เมื่อพิมพ์เสร็จให้คลิกปุ่ม Send ค่ารหัส ASCII ของตัวอักษร 1,2,3,4 และ 5 คือ 49,50,51,52 และ 53 ตามลำดับ ในการคำนวณค่าหน่วงเวลาของฟังก์ชั่น delay() ทำได้โดยนำค่ารหัส ASCII ของตัวอักษรลบด้วย 48 แล้วนำค่าที่ได้คูณด้วย 100 ซึ่งมีสูตรดังนี้

delayTime=(incomingByte-48)*100; การตรวจจับอินพุตจากการกดปุ่ม 1 ถึง 5 ของผู้ใช้งานทำได้โดยใช้คำสั่ง if โดยกำหนดเงื่อนไขดังนี้

if(incomingByte >= 49 && incomingByte <=53) เงื่อนไขของคำสั่งนี้จะเป็นจริงเมื่อค่าของข้อมูลที่รับได้มากกว่าหรือเท่ากับ 49 และต้องน้อยกว่าหรือเท่ากับ 53 จึงจะคำนวณค่าหน่วงเวลาใหม่ ถ้าเป็นปุ่มอื่นเงื่อนไขของคำสั่ง if เป็นเท็จ จะกำหนดให้ข้ามส่วนคำนวณค่าหน่วงเวลาไป

4.3 ฟังก์ชั่นอินพุตเอ้าต์พุตแอนะล็อก

คำอธิบายและการเรียกใช้ฟังก์ชั่น

4.3.1 analogRead(pin)

อ่านค่าจากขาพอร์ตที่กำหนดให้เป็นอินพุตแอนะล็อก Arduino มีวงจรแปลงสัญญาณแอนะล็อกเป็นดิจิตอลความละเอียด 10 บิต ทำให้แปลงค่าแรงดันอินพุต 0 ถึง +5V ให้เป็นข้อมูลตัวเลขจำนวนเต็มระหว่าง 0 ถึง 1,023

พารามิเตอร์

pin - หมายเลขของขาอินพุตแอนะล็อก มีค่า 0 ถึง 11 หรือเป็นตัวแปรที่ใช้แทนค่า 0 ถึง 11

ค่าที่ส่งกลับ

เลขจำนวนเต็มจาก 0 ถึง 1023

หมายเหตุ

สำหรับขาที่เป็นอินพุตแอนะล็อกไม่จำเป็นต้องประกาศแจ้งว่าเป็นอินพุตหรือเอาต์พุต

ตัวอย่างที่ 4.11

```
int ledPin = 13; // LED connected to digital pin 13
int analogPin = 3; // potentiometer connected to analog pin 3
int val = 0; // variable to store the read value
int threshold = 512; // threshold
void setup()
{
pinMode(ledPin, OUTPUT); // sets the digital pin 13 as output
}
void loop()
{
val = analogRead(analogPin); // read the input pin
if (val >= threshold)
{
digitalWrite(ledPin, HIGH); // LED on
}
else
{
digitalWrite(ledPin, LOW); // LED off
}
}
```

ตัวอย่างนี้จะสั่งให้ขา 13 เป็น HIGH เมื่ออ่านค่าจากขา analogPin แล้วมีค่ามากกว่าหรือเท่ากับค่าเงื่อนไขที่กำหนดไว้ (ในตัวอย่างค่าเงื่อนไขหรือ threshold = 255) ทำให้ LED ที่ต่ออยู่ติดสว่าง แต่ถ้ามีค่าน้อยกว่าขา 13 จะเป็น LOW ทำให้ LED ดับ

4.3.2 analogWrite(pin, value)

ใช้ในการเขียนค่าแอนะล็อกไปยังขาพอร์ตที่กำหนดไว้เพื่อสร้างสัญญาณ PWM

พารามิเตอร์

pin - หมายเลขขาพอร์ตของ MCU

value - เป็นค่าดิวตี้ไซเกิลมีค่าระหว่าง 0 ถึ ง 255

เมื่อค่าเป็น 0 แรงดันของขาพอร์ตที่กำหนดจะเป็น 0V เมื่อมีค่าเป็น 255 แรงดันที่ขาพอร์ตจะเป็น +5V สำหรับค่าระหว่าง 0 ถึง 255 จะทำให้ขาพอร์ตที่กำหนดไว้มีค่าแรงดันเปลี่ยนแปลงในย่าน 0 ถึ ง 5V ค่าที่ส่งกลับจากฟังก์ชั่นเลขจำนวนเต็มจาก 0 ถึง 255

หมายเหตุ

ขาพอร์ตที่ใช้สร้างสัญญาณ PWM ด้วยฟังก์ชั่น analogWrite () ซึ่งก็คือขา 3, 5, 9, 10 และ 11 จะพิเศษจากขาที่เป็นพอร์ตดิจิตอลปกติ คือไม่ต้องกำหนดค่าเพื่อเลือกเป็น INPUT หรือOUTPUT ค่าความถี่ของสัญญาณ PWM มีค่าประมาณ 490Hz คำสั่ง analogWrite ทำงานกับขา 3, 5, 9, 10, 11 เท่านั้น สำหรับขาอื่นๆ จะต้องเขียนค่าดิจิตอลกำหนดเป็น 0 หรือ 5V

ผู้ใช้งานสามารถนำสัญญาณที่ได้จากคำสั่งนี้ไปใช้ในการปรับความสว่างของ LED หรือต่อขยายกระแสเพื่อต่อปรับความเร็วของมอเตอร์ได้ หลังจากเรียกใช้คำสั่งนี้ที่ขาพอร์ตที่กำหนดจะมีสัญญาณ PWM ส่งออกมาอย่างต่อเนื่องจนกว่าจะเรียกใช้ analogWrite (หรือเรียกคำสั่ง digitalRead หรือ digitalWrite ที่ขาเดียวกัน)

ตัวอย่างที่ 4.12

```
int ledPin = 11; // LED connected to digital pin 11
int analogPin = 0; // potentiometer connected to analog pin 0
int val = 0; // variable to store the read value
void setup()
{
pinMode(ledPin, OUTPUT); // sets the pin as output
}
void loop()
{
val = analogRead(analogPin); // read the input pin
analogWrite(ledPin, val / 4); // analogRead values go from 0 to 1023,
// analogWrite values from 0 to 255
}
```

ตัวอย่างนี้จะควบคุมความสว่างของ LED ที่ต่อกับขา 11 ให้เป็นไปตามค่าที่อ่านได้จากตัวต้านทานปรับค่าได้ที่ต่อกับขา A0

4.3.3 การทดลองอินพุตแอนะล็อกของ Arduino

ภายในไมโครคอนโทรลเลอร์ Atmega328 บนบอร์ด Arduino มีวงจรแปลงสั ญญาณแอนะล็อกเป็นดิจิตอล ความละเอียด 10 บิตจำนวน 6 ช่อง ซึ่งได้กำหนดขาต่อเป็น A0 ถึง A6 วงจรแปลงสัญญาณแอนะล็อกเป็นดิจิตอลมีความละเอียด 10 บิตทำหน้าที่แปลงแรงดันอินพุต 0 ถึง 5V เป็นค่าเลขจำนวนเต็ม 0 ถึง 1023

ฟังก์ชั่นสำหรับอ่านค่าจากอินพุตแอนะล็อกคือ analogRead() โดยค่าในวงเล็บคือหมายเลขของช่องสัญญาณที่ต้องการอ่านค่า (0 ถึง 6) เมื่อฟังก์ชั่นทำงานเสร็จจะคืนค่าเป็นเลขจำนวนเต็ม (int) จาก 0 ถึง 1,023 ถ้าต้องการคำนวณเป็นค่าแรงดันที่มีหน่วยเป็นโวลต์ (Volt : V) กระทำได้จากความสัมพันธ์ดังนี้

volt = ค่าที่อ่านได้ x 5 /1023

4.4 ฟังก์ชั่นเกี่ยวกับเวลา

4.4.1 unsigned long millis()

คืนค่าเป็นค่าเวลาในหน่วยมิลลิวินาที นับตั้งแต่เริ่มรันโปรแกรมปัจจุบัน

ค่าที่ส่งกลับจากฟังก์ชั่น

ค่าเวลาในหน่วยเป็นมิลลิวินาที ตั้งแต่เริ่มรันโปรแกรมปัจจุบันคืนค่าเป็น unsigned long ค่าตัวเลขจะเกิดการโอเวอร์โฟล์ว (ค่าเกินแล้วกลับเป็นศูนย์) เมื่อเวลาผ่านไปประมาณ 9 ชั่วโมง

ตัวอย่างที่ 4.12

```
long time;
void setup()
{
Serial.begin(9600);
}
void loop()
{
Serial.print("Time: ");
time = millis();
Serial.println(time); //prints time since program started
delay(1000); }
 // wait a second so as not to send massive amounts of data
```

4.4.2 delay(ms)

เป็นฟังก์ชั่นชะลอการทำงานหรือหน่วงเวลาของโปรแกรมตามเวลาที่กำหนดในหน่วยมิลลิวินาที

พารามิเตอร์

ms - ระยะเวลาที่ต้องการหน่วงเวลาหน่วยเป็นมิลลิวินาที (1000 ms เท่ากับ 1 วินาที)

ตัวอย่างที่ 4.13

```
        int ledPin = 13;                // LED connected to pin 13
void setup()
        {
        pinMode(ledPin, OUTPUT);    // sets as output
        }
void loop()
        {
        digitalWrite(ledPin, HIGH);     // LED on
        delay(1000);                    // waits for a second
        digitalWrite(ledPin, LOW);      // LED off
        delay(1000);                    // waits for a second
        }
```

จากตัวอย่างนี้กำหนดให้ pin หมายเลข 13 เป็นเอาต์พุต สั่งให้เป็น HIGH (LED ติด) หยุดรอ 1000 มิลลิวินาที (1 วินาที) แล้วสั่งเป็น LOW (LED ดับ) แล้วหยุดรอ 1000 มิลลิวินาที

4.4.3 delayMicroseconds(us)

เป็นฟังก์ชั่นชะลอการทำงานหรือหน่วงเวลาของโปรแกรมตามเวลาที่กำหนด ในหน่วยไมโครวินาที

พารามิเตอร์

us - ค่าหน่วงเวลาในหน่วยไมโครวินาที

(1000 ไมโครวินาที = 1 มิลลิวินาที และหนึ่งล้านไมโครวินาที = 1 วินาที)

ตัวอย่างที่ 4.13

```
        int outPin = 11; // digital pin 11
void setup()
        {
        pinMode(outPin, OUTPUT); // sets as output
        }
```

```
void loop()
        {
        digitalWrite(outPin, HIGH); // sets the pin on
        delayMicroseconds(50); // pauses for 50 microseconds
        digitalWrite(outPin, LOW); // sets the pin off
        delayMicroseconds(50); // pauses for 50 microseconds
        }
```

จากตัวอย่างนี้กำหนดให้ขา 11 ทำงานเป็นเอาต์พุต เพื่อส่งสัญญาณพัลส์ที่มีคาบเวลา 100 ไมโครวินาที ต่อเนื่องตลอดเวลา

4.5 ฟังก์ชั่นเกี่ยวกับอินเตอร์รัปต์ภายนอก

4.5.1 attachInterrupt(interrupt, function, mode)

ใช้ระบุว่าเมื่อขาอินพุตที่รับสัญญาณอินเตอร์รัปต์จากภายนอก มีการเปลี่ยนแปลงเกิดขึ้นจะกำหนดให้ซีพียูกระโดดไปยังฟังก์ชั่นใด โดยมีขาอินพุตรับสัญญาณอินเตอร์รัปต์จากภายนอก 2 ขาคือ D2 และ D3 ซึ่งตรง

หมายเหตุ

ฟังก์ชั่นที่ทำงานเมื่อการอินเตอร์รัปต์ จะไม่สามารถเรียกใช้ฟังก์ชั่น milis() และ delay() ได้เมื่อเกิดการตอบสนองอินเตอร์รัปต์แล้ว ดังนั้นข้อมูลที่เข้ามาทางขา serial data อาจสูญหายได้

พารามิเตอร์

Interrupt - หมายเลขของช่องอินพุตอินเตอร์รัปต์ (เป็น int)

function – ฟังก์ชั่นที่จะกระโดดไปทำงานเมื่อเกิดอินเตอร์รัปต์ ฟังก์ชั่นนี้ต้องไม่รับค่าพารามิเตอร์และไม่มีการคืนค่า

mode - เลือกประเภทสัญญาณที่ใช้กระตุ้นให้เกิดการอินเตอร์รัปต์

LOW	เกิดอินเตอร์รัปต์เมื่อขาสัญญาณเป็นลอจิก “0”
CHANGE	เกิดอินเตอร์รัปต์เมื่อมีการเปลี่ยนแปลงลอจิก
RISING	เกิดอินเตอร์รัปต์เมื่อมีการเปลี่ยนลอจิก “0” เป็น “1”
FALLING	เกิดอินเตอร์รัปต์เมื่อเปลี่ยนลอจิก “1” เป็น “0”

ตัวอย่างที่ 4.14

```
int pin = 13;
volatile int state = LOW;
```

```
void setup()
        {
        pinMode(pin, OUTPUT);
        attachInterrupt(3, blink, CHANGE);
        }
void loop()
        {
        digitalWrite(pin, state);
        }
void blink()
        {
        state = !state;
        }
```

ตัวอย่างนี้เลือกอินพุตอินเตอร์รัปต์ช่อง 0 กำหนดให้กระโดดไปทำงานที่ฟังก์ชั่น blink เพื่อเปลี่ยนสถานะลอจิกที่ขา 13 เมื่อเกิดการอินเตอร์รัปต์จากการเปลี่ยนแปลงลอจิกที่ขา 3

4.5.2 detachInterrupt(interrupt)

ยกเลิกการอินเตอร์รัปต์

พารามิเตอร์

Interrupt - หมายเลขของช่องอินพุตอินเตอร์รัปต์ที่ต้องการยกเลิก (ค่าเป็น 0 หรือ 1)

4.6 ฟังก์ชั่นทางคณิตศาสตร์

4.6.1 min(x, y)

หาค่าตัวเลขที่น้อยที่สุดของตัวเลขสองตัว

พารามิเตอร์

x - ตัวเลขตัวแรกเป็นข้อมูลประเภทใดก็ได้

y - ตัวเลขตัวที่สองเป็นข้อมูลประเภทใดก็ได้

ค่าที่ส่งกลับจากฟังก์ชั่น

ค่าที่น้อยที่สุดของตัวเลขสองตัวที่ให้

ตัวอย่างที่ 4.15

```
sensVal = min(sensVal, 100);
```

```
// assigns sensVal to the smaller of sensVal or 100.
// ensuring that it never gets above 100.
```

ตัวอย่างนี้จะได้ค่าของ sensVal ที่ไม่เกิน 100 กลับจากฟังก์ชั่น

4.6.2 max(x, y)

หาค่าตัวเลขที่มากที่สุดของตัวเลขสองตัว

พารามิเตอร์

x - ตัวเลขตัวแรกเป็นข้อมูลประเภทใดก็ได้

y - ตัวเลขตัวที่สองเป็นข้อมูลประเภทใดก็ได้

ค่าที่ส่งกลับจากฟังก์ชั่น

ค่าที่มากที่สุดของตัวเลขสองตัวที่ให้

ตัวอย่างที่ 4.16

```
sensVal = max(senVal, 20);
// assigns sensVal to the bigger of sensVal
// or 20 (effectively ensuring that it is at least 20)
```

จากตัวอย่างนี้ค่าของ sensVal จะมีค่าอย่างน้อย 20

4.6.3 abs(x)

หาค่าสัมบูรณ์ (absolute) ของตัวเลขเป็นการทำให้ค่าของตัวแปรเป็นค่าจำนวนเต็มบวก

พารามิเตอร์

x - ตัวเลขค่าที่ส่งกลับจากฟังก์ชั่น x มีค่ามากกว่าหรือเท่ากับศูนย์ (x มีค่าเป็นบวกหรือศูนย์)

-x - เมื่อ x มีค่าน้อยกว่าศูนย์ (x มีค่าติดลบ)

4.6.4 constrain(x, a, b)

ปัดค่าตัวเลขที่น้อยกว่าหรือมากกว่าให้อยู่ในช่วงที่กำหนด

พารามิเตอร์

x - ตัวเลขที่ต้องการปัดค่าให้อยู่ในช่วงที่กำหนดสามารถเป็นข้อมูลชนิดใดก็ได้

a - ค่าต่ำสุดของช่วงที่กำหนด

b - ค่าสูงสุดของช่วงที่กำหนด

ค่าที่ส่งกลับจากฟังก์ชั่น

x เมื่อ x มีค่าอยู่ระหว่าง a และ b

a เมื่อ x มีค่าน้อยกว่า a

b เมื่อ x มี ค่ามากกว่า b

ตัวอย่างที่ 4.17

```
sensVal = constrain(sensVal, 10, 150);
// limits range of sensor values to between 10 and 150
```

จากตัวอย่างนี้ค่าของ sensVal จะอยู่ ในช่วง 10 ถึ ง 150

4.7 ฟังก์ชั่นเกี่ยวกับเลขสุ่ม

4.7.1 randomSeed(seed)

ใช้กำหนดตัวแปรสำหรับสร้างตัวเลขสุ่มโดยสามารถใช้ตัวแปรได้หลากหลายรูปแบบ โดยทั่วไปจะใช้ค่าเวลาปัจจุบัน (จากฟังก์ชั่น milis()) แต่สามารถใช้ค่าอย่างอื่นได้ เช่นค่าที่ได้เมื่อผู้ใช้กดสวิช์หรือค่าสัญญาณรบกวนที่อ่านได้จากขาอินพุตแอนะล็อก

พารามิเตอร์

seed เป็นค่าตัวเลขแบบ long int

ตัวอย่างที่ 4.18

```
long randNumber;
void setup()
{
Serial.begin(19200);
}
void loop()
{
randomSeed(analogRead(0));
randNumber = random(300);
Serial.println(randNumber);
}
```

ในตัวอย่างนี้กำหนดให้เกิดการสุ่มตัวเลขขึ้นเมื่ออ่านค่าจากอินพุตแอนะล็อกช่อง 0 (A0) ย่านของตัวเลขสุ่มคือ 0 ถึง 300 เมื่อทำการสุ่มตัวเลขแล้วให้แสดงค่านั้นที่หน้าต่าง Serial Monitor

4.7.2 long random(max), long random (min, max)

ใช้สร้างตัวเลขสุ่มเสมือน (pseudo-random numbers) เพื่อนำไปใช้ในโปรแกรมก่อนใช้ฟังก์ชั้นนี้จะต้องเรียกใช้ฟังก์ชั่น randomSeed() ก่อน

พารามิเตอร์

min กำหนดค่าตัวเลขสุ่มไม่น้อยกว่าค่านี้ (เป็นออปชั่นเพิ่มเติม)

max กำหนดค่าสูงสุดของตัวเลขสุ่ม

ค่าที่ส่งกลับจากฟังก์ชั่น

คืนค่าเป็นตัวเลขสุ่มในช่วงที่กำหนด (เป็นตัวแปร long int)

ตัวอย่างที่ 4.19

```
long randNumber;
void setup()
{
Serial.begin(19200);
}
void loop()
{
randomSeed(analogRead(2)); // return a random number from 50 - 300
randNumber = random(50,300);
Serial.println(randNumber);
}
```

ในตัวอย่างนี้กำหนดให้สุ่มตัวเลข เมื่ออ่านค่าจากอินพุตแอนะล็อกช่อง 2 (A2) ย่านของตัวเลขสุ่มคือ 50 ถึง 300 เมื่อทำการสุ่มตัวเลขแล้ว ให้แสดงค่านั้นที่หน้าต่าง Serial Monitor

สรุปเนื้อหาสาระสำคัญ

Arduino เป็นภาษาอิตาลี โดยเป็นชื่อโครงการพัฒนาไมโครคอนโทรลเลอร์ตระกูล AVR ในรูปแบบ Open Source คือวิธีการในการออกแบบ พัฒนา และแจกจ่ายสำหรับต้นฉบับของสินค้าหรือความรู้ โดยเฉพาะซอฟต์แวร์ โดยโอเพนซอร์ซถูกพิจารณาว่าเป็นทั้งรูปแบบหนึ่งในการออกแบบ และแผนการในการดำเนินการ โดยโอเพนซอร์ซเปิดโอกาสให้บุคคลอื่นนำเอาระบบนั้นไปพัฒนาได้ต่อไป พัฒนามาจากโครงการ Open Source เดิมของ AVR ที่ชื่อ Wiring โดยโครงการ Wiring ใช้ไมโครคอนโทรลเลอร์ AVR เบอร์ ATmega128 ซึ่งมีข้อจำกัดหลายด้าน เช่น เป็นชิปที่มีตัวถังแบบ SMD ทำให้นำมาใช้งานยากเพราะตัวไมโครคอนโทรลเลอร์มีขนาดเล็กเกินไป ทำให้ไม่สะดวกในการต่อใช้งานจริง มีขาอินพุทและเอ้าท์พุทจำนวนมากเกินไป ตัวบอร์ดมีขนาดใหญ่เกินไป ไม่เหมาะสมสำหรับผู้ที่เริ่มต้นเรียนรู้ด้านไมโครคอนโทรลเลอร์ ด้วยเหตุผลข้างต้นจึงทำให้ไม่ได้รับความนิยม ระยะต่อมาทีมงาน Arduino จึงได้นำโครงการ Wiring มาพัฒนาใหม่โดยใช้ไมโครคอนโทรลเลอร์ AVR ขนาดเล็ก คือ Mega8 และ Mega168 ทำให้ได้รับความนิยมจนถึงปัจจุบันนี้

เรื่อง ฟังก์ชั่นพื้นฐานของ Arduino และการควบคุมหลอดไฟ LED ใช้เวลา 20 นาที

****

คำชี้แจง แบบฝึกหัดมีทั้งหมด 2 ตอน ประกอบด้วยตอนที่ 1 และตอนที่ 2 (20 คะแนน)

2. แบบฝึกหัดตอนที่ 1 เป็นคำถามแบบถูก-ผิด มีทั้งหมด 20 ข้อ (10 คะแนน)

3. แบบฝึกหัดตอนที่ 2 เป็นคำถามแบบปรนัย มีทั้งหมด 10 ข้อ (10 คะแนน)

คำชี้แจง ให้ผู้เรียนกาเครื่องหมายถูก ✓ ในข้อที่คิดว่าถูก และกาเครื่องหมายผิด ✗ ในข้อที่คิดว่าผิด

_____ 1. Serial.begin(speed) ใช้กำหนดอัตราเร็วของการถ่ายทอดข้อมูลหรืออัตราบิตหรือบิตเรต

_____ 2. Serial.available() ใช้ตรวจสอบว่ามีข้อมูลด้านรับหรือไม่โดยคืนค่าเป็น int ตามข้อมูลที่รับเข้า

_____ 3. Serial.read(data) ใช้อ่านค่าข้อมูลจากพอร์ตขนาน

_____ 4. Serial.write(data) ใช้เขียนข้อมูลไบต์ไปยังพอร์ตอนุกรม

_____ 5. Serial.flush() ใช้ล้างบัฟเฟอร์ทั้งด้านรับและส่ง

_____ 6. Serial.print(data) ใช้ส่งข้อมูลออกพอร์ตขนาน

_____ 7. Serial.println(data) ใช้ส่งข้อมูลออกพอร์ตอนุกรมพร้อมกับขึ้นบรรทัดใหม่

_____ 8. min(x, y) หาค่าตัวเลขที่น้อยที่สุดของตัวเลขสองตัว

_____ 9. max(x, y)หาค่าตัวเลขที่มากที่สุดของตัวเลขสองตัว

_____10. Serial.println(b, DEC) เป็นคำสั่งพิมพ์ค่าตัวแปร b เป็นตัวเลขฐานสิบ ตามด้วยรหัสอักษร carriage return และ linefeed

<u>คำชี้แจง</u> ให้ผู้เรียนเลือกคำตอบที่ถูกที่สุดแล้วกาเครื่องหมายกากบาท (✕) ให้ครบทุกข้อ

1. ฟังก์ชั่น pinMode (pin,mode) ทำงานอย่างไร
   - ก. pin – ใช้กำหนดขาพอร์ตใดๆ / mode – โหมดการทำงานเป็น INPUT หรือ OUTPUT
   - ข. pin – ใช้กำหนดขาพอร์ตใดๆ / mode – โหมดการทำงานเป็น INPUT
   - ค. pin – ใช้กำหนดขาพอร์ตใดๆ / mode – โหมดการทำงานเป็น OUTPUT
   - ง. pin – ใช้กำหนดขาเป็น INPUT / mode – โหมดการทำงานเป็น INPUT หรือ OUTPUT
2. ฟังก์ชั่น int digitalRead (pin) ทำงานอย่างไร
   - ก. อ่านค่าสถานะของขาที่ระบุไว้ว่ามีค่าเป็น HIGH หรือ LOW
   - ข. อ่านค่าสถานะของขาที่ระบุไว้ว่ามีค่าเป็น ON หรือ OFF
   - ค. อ่านค่าสถานะของขาที่ระบุไว้ว่ามีค่าเป็น INPUT หรือ OUTPUT
   - ง. อ่านค่าสถานะของขาที่ระบุไว้ว่ามีค่าเป็นจริง หรือ เท็จ
3. ถ้าต้องการปรับให LED กะพริบเร็วขึ้นหรือช้าลง ทำอย่างไร
   - ก. เปลี่ยนค่าในฟังก์ชั่น delay() เป็นค่าอื่นๆ โดยค่ายิ่งมาก LED ยิ่ง<u>กะพริบเร็ว</u>
   - ข. เปลี่ยนค่าในฟังก์ชั่น delay() เป็นค่าอื่นๆ โดยค่ายิ่งมาก LED ยิ่ง<u>กะพริบช้า</u>
   - ค. เปลี่ยนค่าในฟังก์ชั่น delay() เป็นค่าอื่นๆ โดยค่ายิ่งมาก LED ยิ่ง<u>กะพริบคงที่</u>
   - ง. เปลี่ยนค่าในฟังก์ชั่น delay() เป็นค่าอื่นๆ โดยค่ายิ่งมาก LED ยิ่ง<u>กะพริบไม่คงที่</u>
4. ฟังก์ชั่น Serial.print() และ Serial.println() มีรูปแบบอย่างไร
   - ก. Serial.print(b,FORMAT);
   - ข. Serial.println(b,FORMAT);
   - ค. initialization; condition;
   - ง. Serial.print(b,FORMAT); กับ Serial.println(b,FORMAT);
5. อ่านค่าอินพุตได้จากฟังก์ชั่น
   - ก. Readdigital
   - ข. digitalRead
   - ค. Modepin
   - ง. pinMode

6. สัญญาณรบกวนในการกดสวิตช์เรียกว่า
   ก. bounce
   ข. debounce
   ค. boun
   ง. deboun
7. หลักการแก้ไขสัญญาณรบกวนทำได้หลายแบบ เช่นต่อกันในลักษณะวงจร
   ก. วงจร RL อินติเกรเตอร์
   ข. วงจร RLC อินติเกรเตอร์
   ค. วงจร LC อินติเกรเตอร์
   ง. วงจร RC อินติเกรเตอร์
8. อัตราบอดของการรับส่งข้อมูลอนุกรมมีหน่วยเป็น
   ก. บิตต่อวินาที
   ข. บิตต่อนาที
   ค. บิตต่อครึ่งชั่วโมง
   ง. บิตต่อชั่วโมง
9. ฟังก์ชั่น delayMicroseconds(us) คือ
   ก. ฟังก์ชั่นชะลอการทำงานในหน่วยไมโครวินาที
   ข. ฟังก์ชั่นชะลอการทำงานในหน่วยวินาที
   ค. ฟังก์ชั่นชะลอการทำงานในหน่วยนาที
   ง. ฟังก์ชั่นชะลอการทำงานในหน่วยนาโนวินาที
10. constrain(x, a, b) คือฟังก์ชั่นอะไร
   ก. ค่าต่ำสุดของช่วงที่กำหนด
   ข. ค่าสูงสุดของช่วงที่กำหนด
   ค. ปัดค่าตัวเลขที่น้อยกว่าหรือมากกว่าให้อยู่ในช่วงที่กำหนด
   ง. ตัวแปรจำนวนเต็ม 64 บิต แบบไม่คิดเครื่องหมาย

ปฏิบัติการทดลองหน่วยที่ 4
เรื่อง ฟังก์ชั่นพื้นฐานของ Arduino และการควบคุมหลอดไฟ LED

****

<u>คำชี้แจง</u> ให้ผู้เรียนทุกคนทำการทดลองตามปฏิบัติการทดลองหน่วยที่ 4 เรื่อง ฟังก์ชั่นพื้นฐานของ Arduino และการควบคุมหลอดไฟ LED โดยใช้เวลา 180 นาที (20 คะแนน)

จุดประสงค์เชิงพฤติกรรม

1. สามารถใช้ฟังก์ชั่นพื้นฐานของ Arduino ได้ถูกต้อง
2. สามารถควบคุมหลอดไฟ LED ได้ถูกต้อง
3. สามารถแก้ปัญหาในการทำงานของบอร์ด Arduino Uno R3 ได้
4. สามารถต่อใช้งานและอัพโหลดโปรแกรมของบอร์ด Arduino Uno R3 ได้

อุปกรณ์การทดลอง

1.	เครื่องคอมพิวเตอร์และโปรแกรม Arduino IDE 1.6.9	1	ชุด
2.	USB Cable Arduino Uno R3	1	เส้น
3.	Arduino Uno R3 Board	1	บอร์ด
6.	Hook-up Wires	10	เส้น
7.	Breadboard	1	แผง
8.	LED	6	ตัว
9.	Pushbutton Switch	2	ตัว
10.	10K Ohm Resistor	2	ตัว
11.	220 Ohm Resistor	6	ตัว

ข้อควรระวัง

1. ควรระวังไม่วางบอร์ด Arduino Uno R3 หรือชีลต่างๆ บนโต๊ะโลหะหรือที่วางที่เป็นโลหะเพราะอาจเกิดการลัดวงจรของภาคจ่ายไฟได้

2. ไม่ควรต่อสายต่อวงจรในบอร์ด Arduino Uno R3 ทิ้งไว้ ควรถอดสายต่อวงจรออกให้หมด เพราะผลการทดลองอาจเกิดการผิดพลาดไม่เป็นไปตามทฤษฎีได้

3. ไม่ควรถอดสายสายโหลด USB เข้าออกตลอดเวลา เพราะอาจทำให้ Arduino Uno R3 เสียหายได้

การทดลองที่ 4.1 เอาต์พุตดิจิตอลของ Arduino Uno โปรแกรมสั่งให้ LED กะพริบ 5 ดวง

ฟังก์ชั่น pinmode(pin,mode);

เมื่อ pin คือ หมายเลขขาที่ต้องการ

Mode คือ โหมดการทำงาน (INPUT หรือ OUTPUT)

หลังจากที่กำหนดให้เป็นเอาต์พุตแล้วเมื่อต้องการเขียนค่าไปยังขานั้นๆ ให้เรียกใช้ฟังก์ชั่น digitalWrite() โดยมีรูปแบบดังนี้

digitalWrite(pin,value);

เมื่อ pin คือหมายเลขขาที่ต้องการ

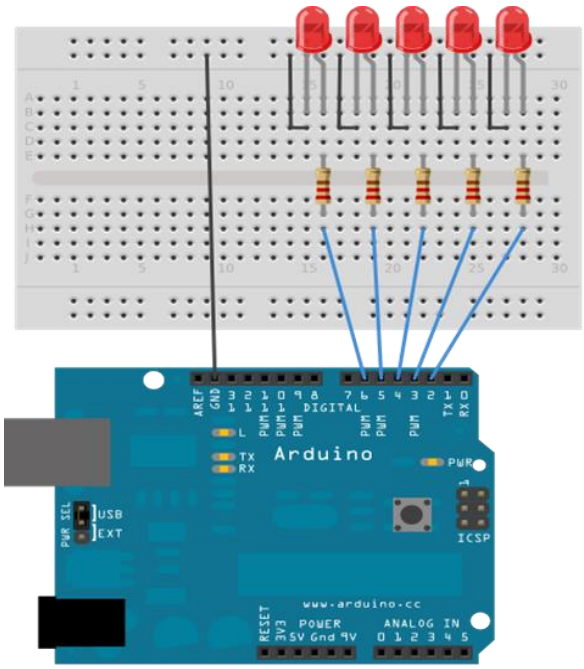
value สถานะลอจิกที่ต้องการ (HIGH หรื อ LOW)

ในตัวอย่างนี้จะนำบอร์ด Arduino มาต่อควบคุม LED จำนวน 5 ตัว โดยสั่งให้ LED ติดตามลำดับเริ่มจาก LED1 ไปยัง LED5 แล้ววนกลับมาเริ่มที่ LED1 ต่อเนื่องตลอดเวลา

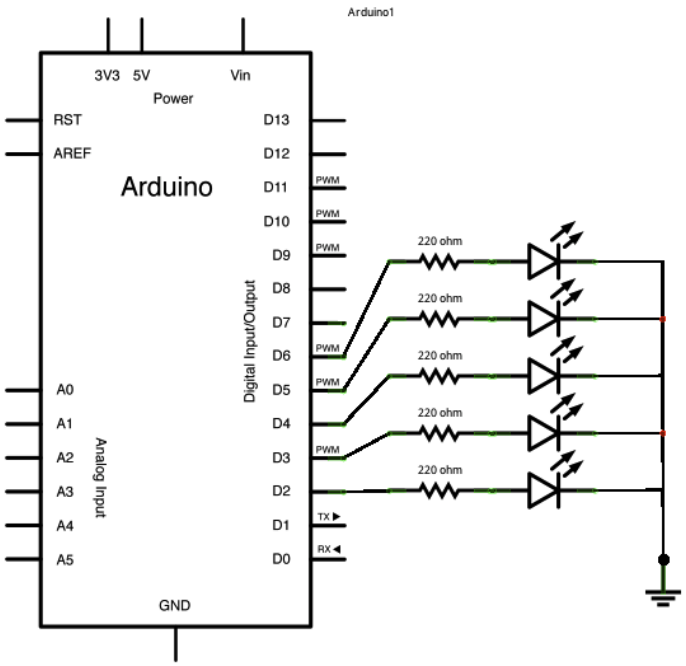
Hardware Required

1. Arduino Uno Board
2. 220 ohm resistor 5 PCS
3. hook-up wires
4. breadboard
5. LED 6 PCS

Circuit



Schematic



Code

```
            #define DELAY_TIME 200
            Int led_pin[ ]={2,3,4,5,6};
            int count;
void setup() // Run once at startup
            {
for(count=0; count<5; count++)
            pinMode(led_pin[count], OUTPUT);
            // Call function pinMode to set Digital pin 2,3,4,5,6 as OUTPUT
            }
void loop() // run over and over again
            {
for(count=0; count<5; count++)
            {
            digitalWrite(led_pin[count], HIGH); // Turn on LED
            delay(DELAY_TIME); // wait for a 0.2 second. (200 ms)
            digitalWrite(led_pin[count], LOW); // Turn off LED
            delay(DELAY_TIME);}}
```

ผลการทดลอง

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

การทดลองที่ 4.2 อินพุตดิจิตอลของ Arduino Uno

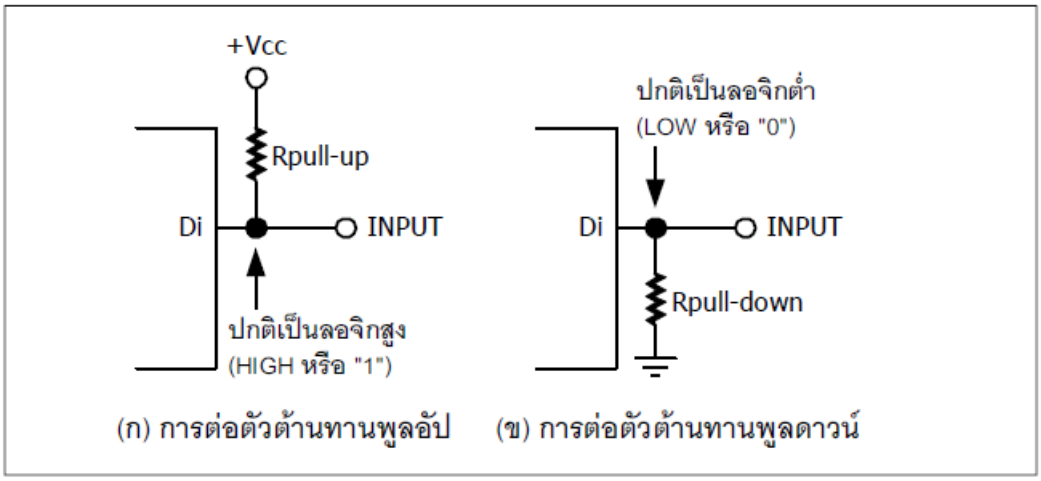
Arduino Uno มีขาพอร์ตดิจิตอลที่กำหนดให้เป็นอินพุตหรือเอาต์พุตจำนวน 13 ขา ถ้าต้องการกำหนดเป็นอินพุตต้องกำหนดด้วยฟังก์ชั่น pinMode และอ่านค่าอินพุตได้จากฟังก์ชั่น digitalRead ซึ่งมีรูปแบบดังนี้

digitalRead (pin);

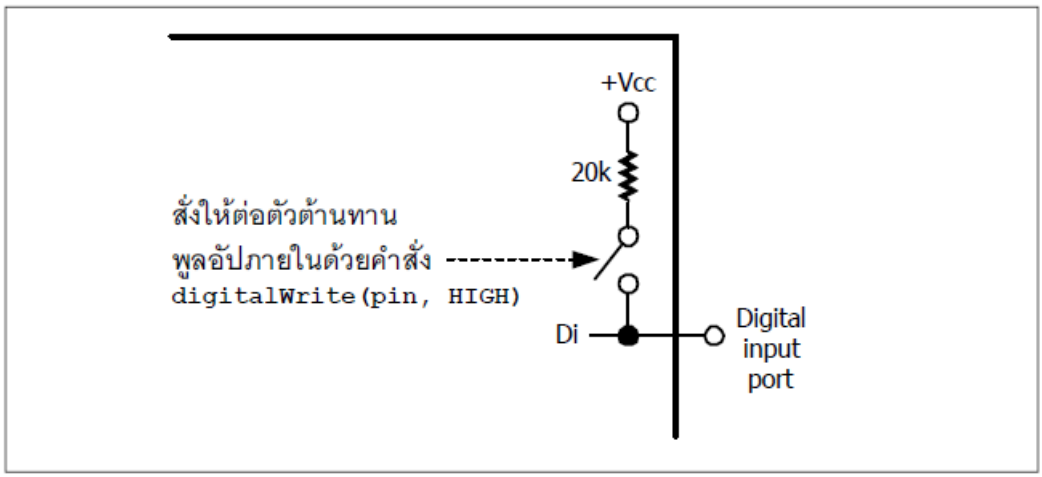
เมื่อ pin คือหมายเลขขาที่ต้องการอ่านค่าสถานะ

เมื่อฟังก์ชั่นทำงาน คือค่าเป็นสถานะของขาที่ต้องการอ่านค่า โดยคือค่าเป็น LOW (ค่าเป็น “0”) หรือ HIGH (ค่าเป็น “1”)

ในการใช้งานขาอินพุตดิจิตอลต้องต่อตัวต้านทานพลูอัป (ต่อตัวต้านทานจากไฟเลี้ยง +5V มายังขาอินพุต) เพื่อกำหนดสถานะที่แน่นอนให้กับขาอินพุตในภาวะที่ไม่มีการกดสวิตช์ ดังวงจรในรูปที่ 4.5 โดยต่อขา 7 ผ่านตัวต้านทานค่า 10kΩ ไปยังไฟเลี้ยง +5V เมื่อไม่ได้กดสวิตช์ SW1 ที่ขา 7 จะมีสถานะเป็นลอจิกสูงหรือ HIGH หรือ “1” เมื่อกดสวิตช์ จะทำให้ขา 7 ต่อลงกราวด์ อ่านค่าสถานะเป็นลอจิกต่ำหรือ LOW หรือ “0”



(ก) การต่อตัวต้านทานพูลอัป (ข) การต่อตัวต้านทานพูลดาวน์

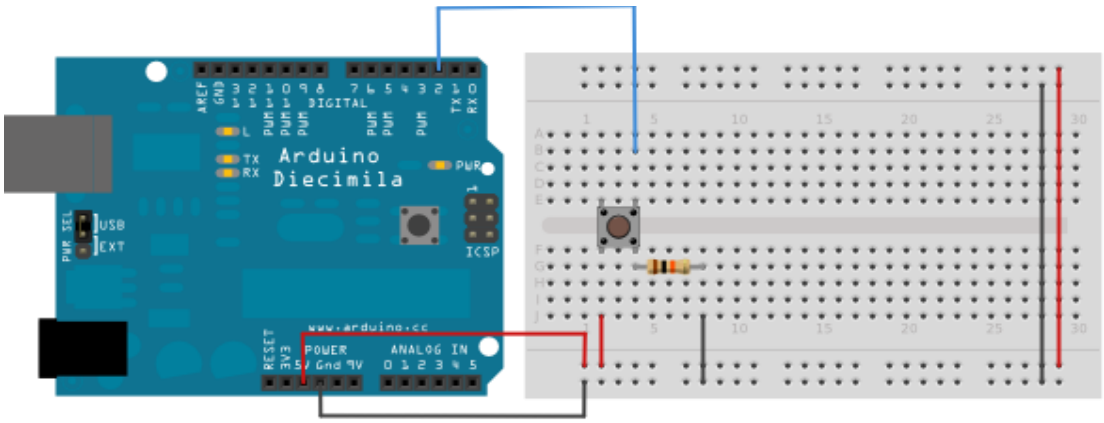


ภายในขาพอร์ตของไมโครคอนโทรลเลอร์ Arduino Uno จะมีการต่อตัวต้านทานพลูอัปค่า 20kΩ เตรียมไว้ให้

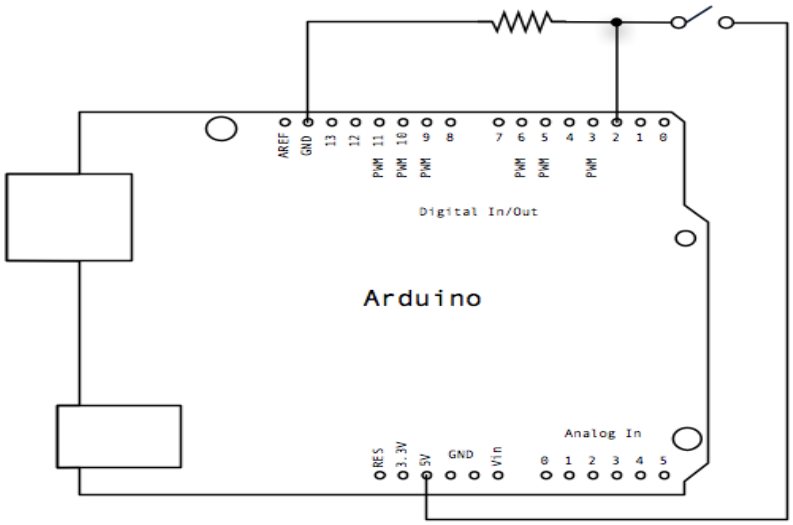
Hardware Required

1. Arduino uno Board
2. button or Switch
3. 10K ohm resistor
4. hook-up wires
5. breadboard

Circuit



Schematic



Code

```
/*
* Read input from push button for control status of LED.
* Modify from button (http://www.arduino.cc/en/Tutorial/button)
* File : Button_LED. Ino
*/
#define LED_PIN 11                // choose the pin for the LED
#define IN_PIN 7                  // choose the input pin (for a pushbutton)
int val = 0;
void setup ()
{
```

```
        pinMode (LED_PIN, OUTPUT);              // declare LED as output
        pinMode (IN_PIN, INPUT);            // declare pushbutton as input
        }
void loop ()
        {
        val = digitalRead (IN_PIN);         // read input value
 if (val == LOW)                            // check the input as LOW (button pushed)
                {
                digitalWrite (LED_PIN, HIGH);               // turn LED ON
                }
 else
                {
                digitalWrite (LED_PIN, LOW);        // turn LED OFF
                }
```

ผลการทดลอง

……

……

……

……

การทดลองที่ 4.3 การส่งข้อมูลออกพอร์ตอนุกรม

เริ่มต้นด้วยการใช้ฟังก์ชั่น Serial.begin() เพื่อสั่งเปิดพอร์ตอนุกรมและกำหนดอัตราถ่ายทอดข้อมูลที่ใช้ในการสื่อสารข้อมูลหรืออัตราบอด มีรูปแบบการเขียนโปรแกรม ดังนี้

Serial.begin(speed);

เมื่อ speed คืออัตราบอดมีค่า 300, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 38400,57600 หรือ 115200 บิตต่อวินาที ปกติที่ใช้คือ 9600

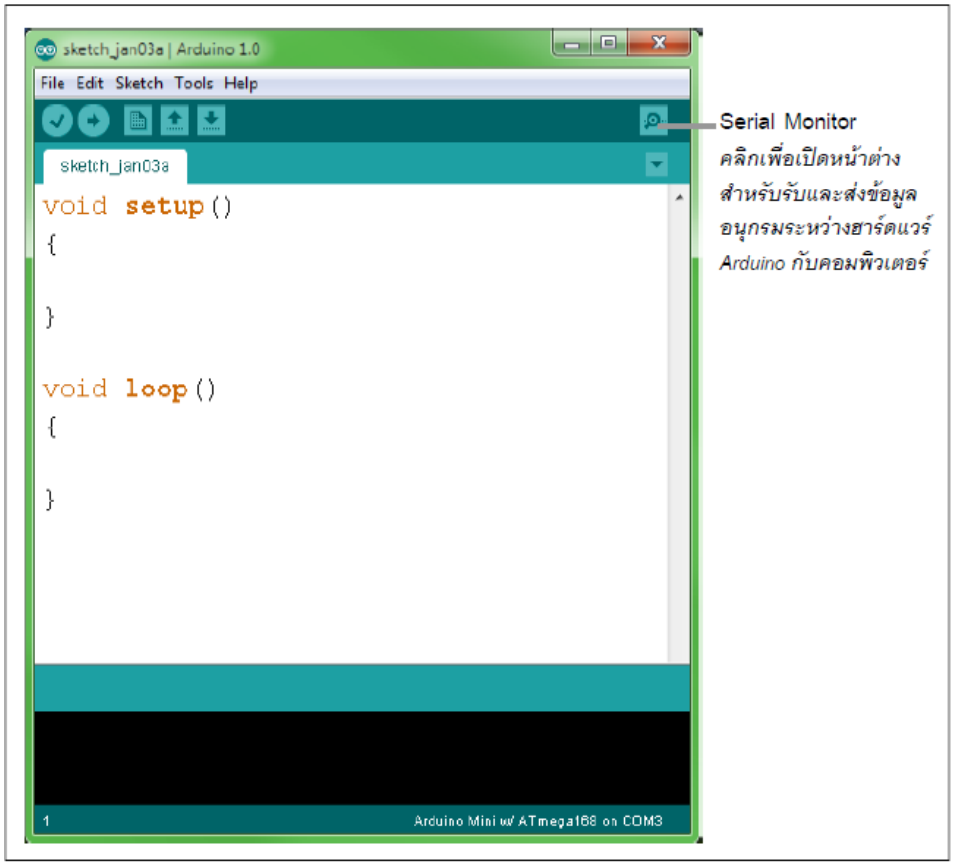
หลังจากเปิดพอร์ต เมื่อต้องการส่งข้อมูลให้ใช้ฟังก์ชั่น Serial.print() หรื อ Serial.println() ฟังก์ชั่นทั้งสองตัวทำงานให้ผลคล้ายกัน ต่างกันเมื่อฟังก์ Serial.println() ส่งข้อมลูแล้วจะขึ้นบรรทัดใหม่ให้อัตโนมัติ

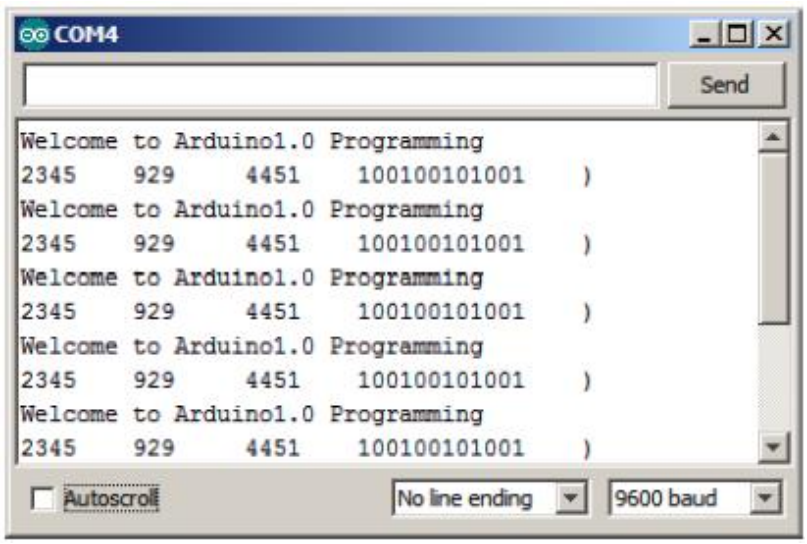
ฟังก์ชั่น Serial.print() และ Serial.println() มีรูปแบบดังนี้

Serial.print(b,FORMAT); กับ Serial.println(b,FORMAT);

โดยที่ b คือค่าตัวแปรประเภทเลขจำนวนเต็มที่ต้องการส่งออกทางพอร์ตอนุกรม ถ้าไม่ระบุรูปแบบจะพิมพ์ออกเป็นรหัส ASCII ของค่าตัวแปร

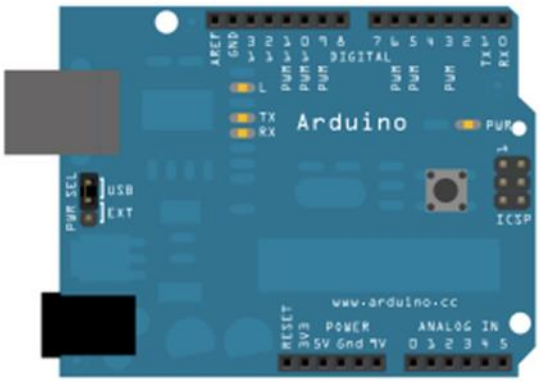
FORMAT เป็นรูปแบบของการพิมพ์มี DEC (เลขฐานสิบ), HEX (เลขฐานสิบหก), OCT (เลขฐานแปด) และ BIN (เลขฐานสอง)





Hardware Required

1. Arduino uno Board



Code

ไฟล์ Serial01.ino โปรแกรมทดสอบการส่งข้อมลูออกพอร์ตอนุกรมเสมือนผ่านทางพอร์ต USB

```
/* Code for send out data to serial port
* File : Serial01.pde
*/
void setup()
{
Serial.begin(9600); // Set serial port to 9600 bit per second
}
void loop()
{
int data = 2345; // print string to serial port
Serial.println("Welcome to Arduino Programming"); // print title
Serial.print(data, DEC); // print as an ASCII-encoded decimal
Serial.print("\t"); // print a tab character
Serial.print(data, HEX); // print as an ASCII-encoded hexadecimal
Serial.print("\t"); // print a tab character
Serial.print(data, OCT); // print as an ASCII-encoded octal
Serial.print("\t"); // print a tab character
Serial.print(data, BIN); // print as an ASCII-encoded binary
```

```
Serial.print("\t"); // print a tab character
Serial.write(data); // print as a raw byte value
Serial.println(); // print a line feed character
delay(1000); // Wait 1 second
}
```

ผลการทดลอง

.....

.....

.....

.....

.....

.....

การทดลองที่ 4.4 ฟังก์เกี่ยวกับเวลา delayMicroseconds(us)

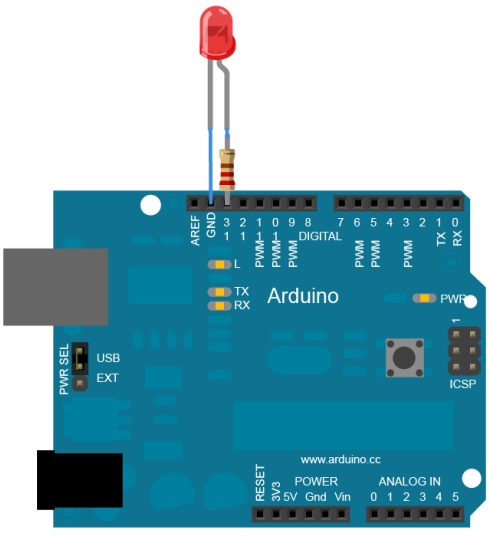
เป็นฟังก์ชั่นชะลอการทำงานหรือหน่วงเวลาของโปรแกรมตามเวลาที่กำหนด ในหน่วยไมโครวินาที

พารามิเตอร์ us - ค่าหน่วงเวลาในหน่วยไมโครวินาที (1000 ไมโครวินาที = 1 มิลลิวินาที และหนึ่งล้านไมโครวินาที = 1 วิ นาที)

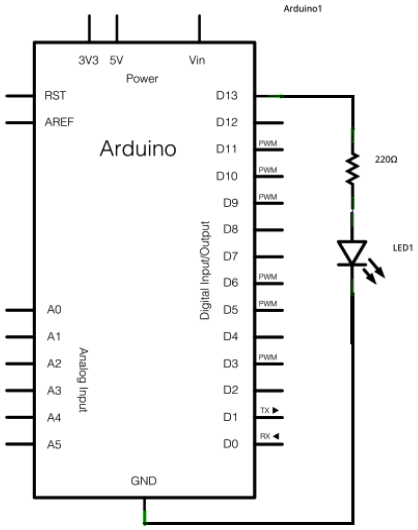
Hardware Required

- Arduino uno Board
- 220 ohm resistor
- LED
- hook-up wires

Circuit



Schematic



Code

```
        int outPin = 11; // digital pin 11
void setup()
        {
        pinMode(outPin, OUTPUT); // sets as output
        }
void loop()
        {
        digitalWrite(outPin, HIGH); // sets the pin on
        delayMicroseconds(50); // pauses for 50 microseconds
        digitalWrite(outPin, LOW); // sets the pin off
```

```
    delayMicroseconds(50); // pauses for 50 microseconds
    }
```

ผลการทดลอง

.....

.....

.....

.....

.....

.....

สรุปผลการทดลอง

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

ปัญหาอุปสรรคหรือข้อเสนอแนะ

.....

.....

ตารางการประเมินผลคะแนนภาคปฏิบัติ

หัวข้อการพิจารณาภาคปฏิบัติ	ระดับคะแนน
การทดลองที่ 4.1 เอาต์พุตดิจิตอลของ Arduino Uno โปรแกรมสั่งให้ LED กะพริบ 5 ดวง	5 คะแนน
การทดลองที่ 4.2 อินพุตดิจิตอลของ Arduino Uno	5 คะแนน
การทดลองที่ 4.3 การส่งข้อมูลออกพอร์ตอนุกรม	5 คะแนน
การทดลองที่ 4.4 ฟังก์เกี่ยวกับเวลา delayMicroseconds(us)	5 คะแนน
รวมคะแนนภาคปฏิบัติ	.....คะแนน

แบบทดสอบหลังเรียน หน่วยที่ 4
เรื่อง ฟังก์ชั่นพื้นฐานของ Arduino

เรื่อง ฟังก์ชั่นพื้นฐานของ Arduino **ใช้เวลา** 20 **นาที**

วิชา ไมโครคอนโทรลเลอร์เบื้องต้น **รหัสวิชา** (2127-2107)

ระดับชั้น ประกาศนียบัตรวิชาชีพ (ปวช.) **สาขาวิชา** เมคคาทรอนิกส์

****

คำชี้แจง 1. แบบทดสอบมีทั้งหมด 10 ข้อ (10 คะแนน)

2. ให้ผู้เรียนเลือกคำตอบที่ถูกที่สุดแล้วกาเครื่องหมายกากบาท (✕) ลงในกระดาษคำตอบ

1. โปรแกรม Arduino IDE สามารถเรียกใช้ฟังก์ชั่นพื้นฐานได้ทันทีโดยไม่ต้อง
   - ก. ใช้คำสั่ง #plus เพื่อผนวกไฟล์เพิ่ม
   - ข. ใช้คำสั่ง #include เพื่อผนวกไฟล์เพิ่ม
   - ค. ใช้คำสั่ง #defile เพื่อผนวกไฟล์เพิ่ม
   - ง. ใช้คำสั่ง #add เพื่อผนวกไฟล์เพิ่ม
2. ฟังก์ชั่น digitalWrite (pin, value) ทำงานอย่างไร
   - ก. ใช้เลือกโหมดการทำงานเป็น INPUT
   - ข. ใช้เลือกโหมดการทำงานเป็น OUTPUT
   - ค. ใช้กำหนดขาพอร์ต OUTPUT ให้มีสถานะเป็นลอจิกสูงหรือลอจิกต่ำ
   - ง. ใช้กำหนดขาพอร์ต INPUT ให้มีสถานะเป็นลอจิกสูงหรือลอจิกต่ำ
3. ฟังก์ชั่นเกี่ยวกับการสื่อสารผ่านพอร์ตอนุกรมคือข้อใด
   - ก. digitalWrite
   - ข. Serial .begin (int datarate)
   - ค. digitalRead
   - ง. pinMode
4. ฟังก์ชั่น int Serial .available ( ) ใช้สำหรับ
   - ก. ใช้แจ้งว่าได้รับสัญญาณแล้ว และพร้อมสำหรับการอ่านไปใช้งาน
   - ข. ใช้แจ้งว่าได้รับข้อมูลตัวอักษร และพร้อมสำหรับการอ่านไปใช้งาน
   - ค. เป็นการเลือกอัตราบอดเท่ากับ 9600 บิตต่อวินาที
   - ง. ส่งค่ากลับจากฟังก์ชั่น และพร้อมสำหรับการอ่านไปใช้งาน

5. ฟังก์ชั่น int Serial .read ( ) ใช้สำหรับ
   ก. อ่านค่าข้อมูลที่ได้รับจากพอร์ตขนาน
   ข. อ่านค่าข้อมูลที่ได้รับจากพอร์ตอนุกรม
   ค. ใช้เขียนแจ้งว่ากำลังอ่านคำสั่ง
   ง. ใช้เขียนแจ้งว่ากำลังลบคำสั่ง
6. . ถ้าต้องการกำหนดขาอินพุตแอนะล็อกต้องกำหนดด้วยฟังก์ชั่น
   ก. AnalogWrite
   ข. AnalogRead
   ค. digitalRead
   ง. ModePin
7. ฟังก์ชั่น unsigned long millis() คืนค่าเป็นค่าเวลาในหน่วยมิลลิวินาที ได้ประมาณกี่ชั่วโมง
   ก. ประมาณ 6 ชั่วโมง
   ข. ประมาณ 7 ชั่วโมง
   ค. ประมาณ 8 ชั่วโมง
   ง. ประมาณ 9 ชั่วโมง
8. ฟังก์ชั่น detachInterrupt(interrupt) ใช้สำหรับ
   ก. เริ่มการอินเตอร์รัปต์ภายนอก
   ข. ยกเลิกการอินเตอร์รัปต์ภายนอก
   ค. ฟังก์ชั่นกระโดดไปทำงานเมื่อเกิดอินเตอร์รัปต์
   ง. เลือกประเภทสัญญาณที่ใช้กระตุ้นให้เกิดการอินเตอร์รัปต์
9. ฟังก์ชั่น constrain(x, a, b) ทำงานอย่างไร
   ก. หาค่าตัวเลขที่มากที่สุดของตัวเลขสองตัว
   ข. หาค่าสัมบูรณ์ของตัวเลข
   ค. ปัดค่าตัวเลขที่น้อยกว่าหรือมากกว่าให้อยู่ในช่วงที่กำหนด
   ง. หาค่าตัวเลขที่น้อยที่สุดของตัวเลขสองตัว
10. randomSeed(seed) คือฟังก์ชั่นอะไร
   ก. ใช้หาค่าสัมบูรณ์ของตัวเลขแบบสุ่ม
   ข. ใช้กำหนดตัวแปรสำหรับสร้างตัวเลขแบบสุ่ม
   ค. ใช้สร้างตัวเลขเสมือนแบบสุ่ม
   ง. ใช้หาค่าที่น้อยที่สุดของตัวเลขสองตัวแบบสุ่ม