01076001
วิศวกรรมคอมพิวเตอร์เบื้องต้น
Introduction to Computer Engineering

Arduino #5

Timer Interrupt

Accelerometer, Serial Communication

Timer Interrupt

- ปกติใน ไมโครคอนโทรลเลอร์ มักจะมีการสร้าง Hardware สำหรับทำงานกับเรื่องของเวลา เช่น ใช้นับเวลาในคำสั่ง millis() หรือ delay() จะเรียก Hardware นี้ว่า Timer
- ใน ATmega328 ซึ่งเป็นไมโครโพรเซสเซอร์ของ Arduino จะมี Timer อยู่ 3 ตัว คือ Timer0, Timer1 และ Timer 2 โดยมีขนาด 8,16 และ 8 บิตตามลำดับ
- Timer ที่เราจะใช้ คือ Timer1 ซึ่งมี 16 บิต ดังนั้นจะนับได้ 65535 โดย Input ของ Timer คือ สัญญาณ Clock ของระบบ (16 MHz) แต่เนื่องจากมีความถี่มากเกินไป ระบบจึงกำหนดให้มีตัวหาร (Prescaler) โดยเลือกได้ 5 ค่า คือ 1, 8, 64, 256, 1024 ซึ่งในระบบของเราจะเลือกตัวหาร 256
- จากความถี่ 16 MHz เมื่อนำไปหาร 256 จะได้ 16,000,000 / 256 = 62500 แปลว่า ใน 1 วินาทีจะมีสัญญาณมาที่ Timer = 62500 ครั้ง

Timer Interrupt

- เมื่อได้รับ 1 clock ตัว Timer จะเพิ่มค่า 1 เมื่อค่าเพิ่มขึ้นเป็น 65535 และได้รับ clock ต่อไป Timer จะรีเซ็ทกลับเป็นค่า 0 และเกิด Interrupt ซึ่งเราสามารถนำเอา Interrupt ไปใช้งานได้
- นอกจากจะกำหนดตัวหารแล้ว Timer ยังอนุญาตให้เรากำหนดค่าเริ่มต้นสำหรับการนับได้อีกด้วย
- เนื่องจากเราต้องการให้ Interrupt ทุกๆ 1 วินาที เพื่อใช้เป็นตัวนับวินาที เราก็จะต้องกำหนดค่าเริ่มต้น โดยเอา 65535-62500+1 = 3036 ซึ่งจะใช้เป็นค่าเริ่มต้น เมื่อมี clock เข้ามาครบ 62500 ก็จะเป็นเวลา 1 วินาทีพอดี

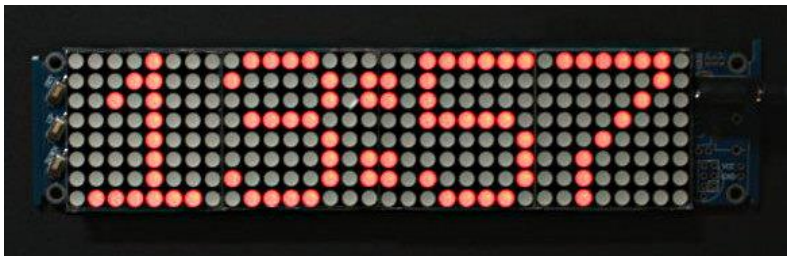
```
#define ledPin 13
int timer1_counter;
int t1=0;
void setup()
{
   Serial.begin(9600);
   pinMode(ledPin, OUTPUT);

   // initialize timer1
   noInterrupts();           // disable all interrupts
   TCCR1A = 0;
   TCCR1B = 0;

   timer1_counter = 3036;   // preload timer 65536-16MHz/256/1Hz

   TCNT1 = timer1_counter;   // preload timer
   TCCR1B |= (1 << CS12);    // 256 prescaler
   TIMSK1 |= (1 << TOIE1);   // enable timer overflow interrupt
   interrupts();             // enable all interrupts
}

ISR(TIMER1_OVF_vect)        // interrupt service routine
{
   TCNT1 = timer1_counter;   // preload timer
   digitalWrite(ledPin, digitalRead(ledPin) ^ 1);
   Serial.println(t1++);
}

void loop()
{
  // your program here...
}
```

Activity

- ให้โหลดโปรแกรมข้างต้นแล้วทดลองทำงาน พร้อมทั้งทำความเข้าใจกับโปรแกรม

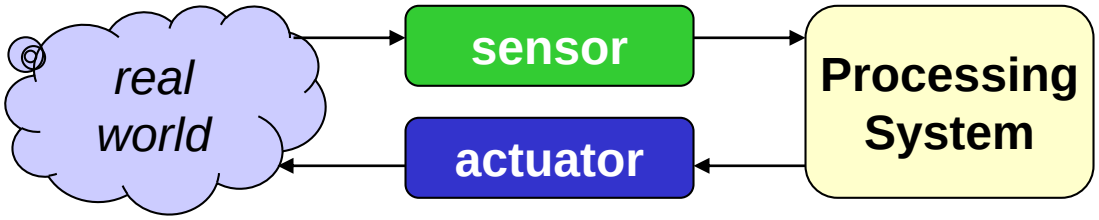
Project #1 : mini clock

- ให้ใช้แผง LED Dot Matrix 32x8 ทำเป็นนาฬิกาดิจิตอล สำหรับความสามารถให้นักศึกษากำหนดเอง โดยต้องใช้อุปกรณ์ประกอบ 1 ตัว เช่น ADXL335 หรือ LDR
- กำหนดส่ง 30 ตุลาคม 2562 (เป็นงานกลุ่ม)
  - รายงาน ประกอบด้วย 1) แนวคิดการออกแบบ (Conceptual Design) 2) การใช้งานโดยย่อ 3) โปรแกรมและการอธิบายโปรแกรมโดยย่อ (อธิบายในระดับฟังก์ชัน)
  - คะแนน 5 คะแนน เกณฑ์การให้คะแนน 1) ฟังก์ชันการใช้งาน 2) ลูกเล่น 3) ความละเอียดครบถ้วนของรายงาน 4) Code โดยใช้หลักผลงานดีที่สุดได้เต็ม ที่เหลือลดลงตามส่วน



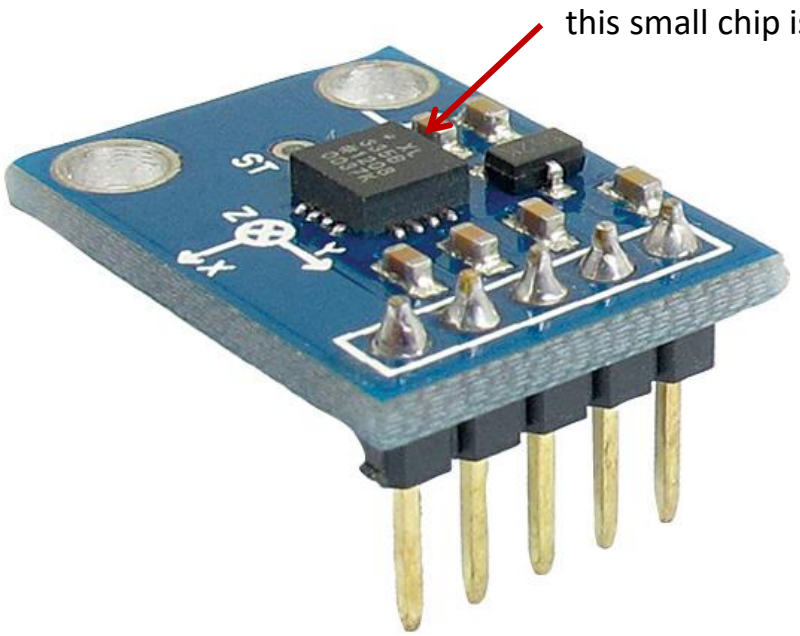
Sensor & Actuator

- Sensor (e.g., thermometer)
  - acquires a physical parameter from the “real world” and converts it into a signal suitable for processing
- Actuator (e.g., heater)
  - a device that generates a signal or stimulus



ADXL 335

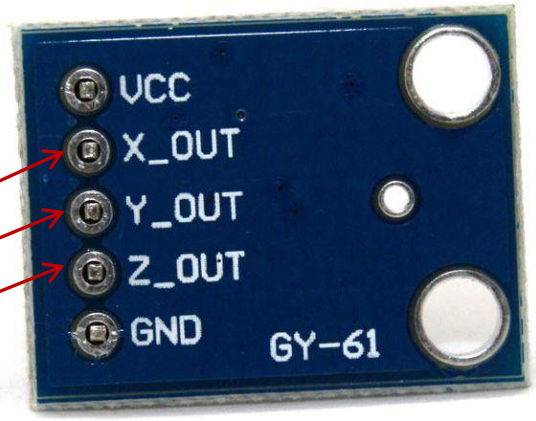
- ใช้ตัววัดความเร่ง เบอร์ ADXL335 ซึ่งเป็นอุปกรณ์ประเภท MEMS
- มีรูสำหรับยึดกับอุปกรณ์ที่ต้องการ
- สามารถวัดความเร่งได้ 3 แกน



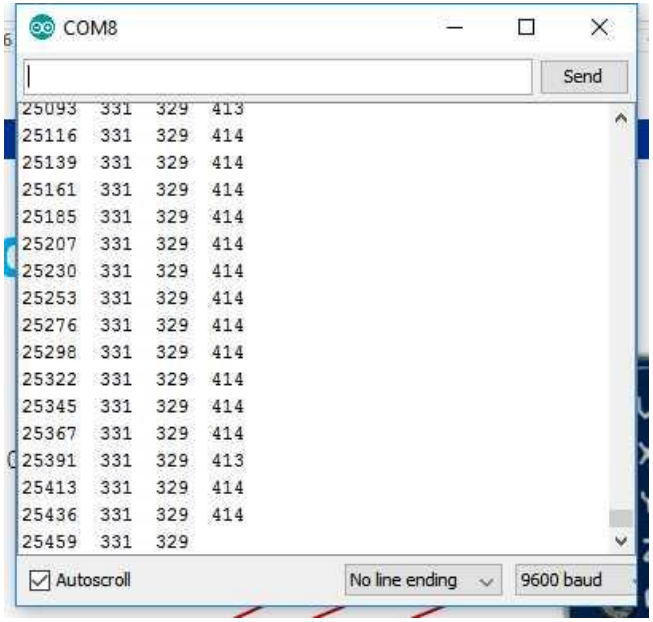
- COM - ground
- VSS - power (we will provide 5V)
- X - acceleration in x-direction
- Y - acceleration in y-direction
- Z - acceleration in z-direction

การใช้งาน Accelerometer

```
void setup()
{
  Serial.begin(9600);
}

void loop()
{
  int xaccel = analogRead(A0);
  int yaccel = analogRead(A1);
  int zaccel = analogRead(A2);
  unsigned long timevar = millis();

  Serial.print(timevar);
  Serial.print("  ");
  Serial.print(xaccel);
  Serial.print("  ");
  Serial.print(yaccel);
  Serial.print("  ");
  Serial.println(zaccel);
}
```



associates a time with each set of accelerations

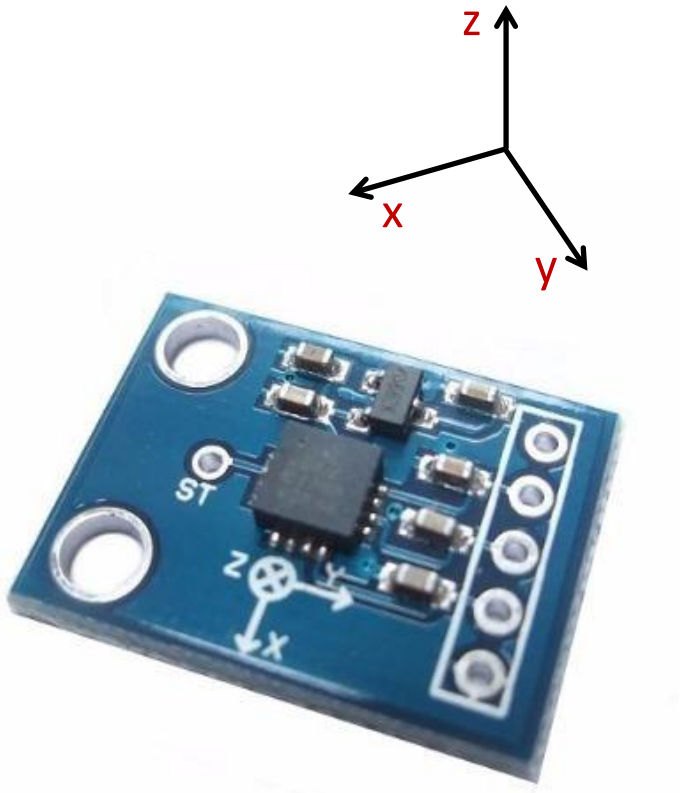


Activity ทดสอบ ADXL335

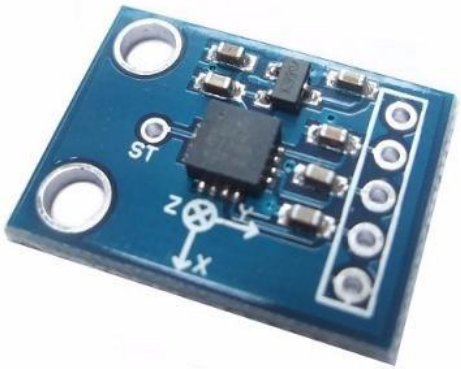
- ให้ใช้โปรแกรมจากหน้าที่แล้ว จากนั้นทดลองหมุนตามแกนต่างๆ ว่า ข้อมูลที่ส่งกลับมาใน Serial Monitor มีการเปลี่ยนแปลงหรือไม่

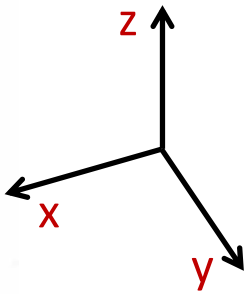
ADXL335 Calibration

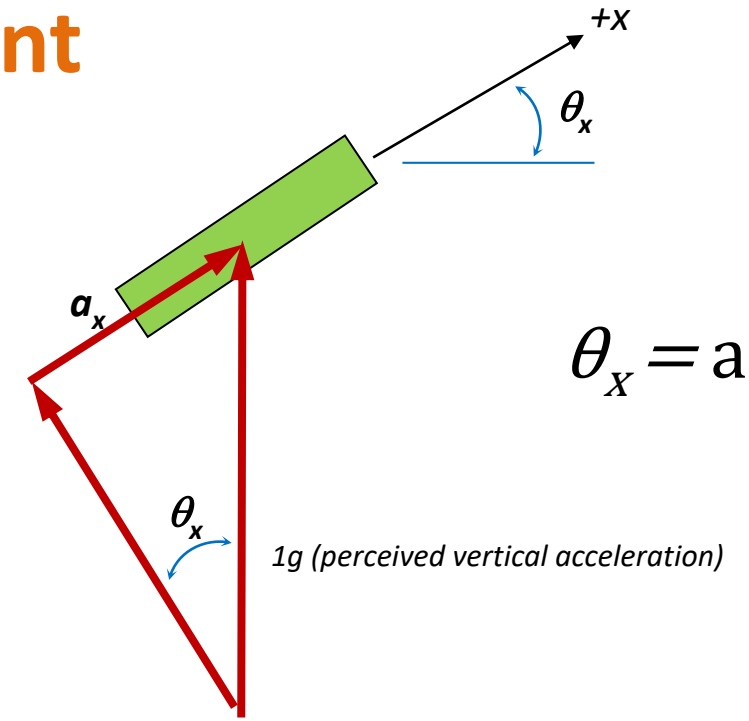
- เนื่องจากภายในเป็นกลไก ซึ่งการอ่านค่าแต่ละตัวอาจจะไม่เท่ากัน ดังนั้นจึงต้อง Calibrate ก่อน เพื่อหาช่วงข้อมูล มิฉะนั้นเมื่อนำไปใช้จะอ่านข้อมูลผิด
- ค่า X เมื่อหันขึ้นด้านบน = 266
- ค่า X เมื่อหันด้านล่าง = 399
- ค่าความแตกต่าง = 399-266 = 133
- ค่าความเร่งเปลี่ยน = 2g
- ดังนั้นค่าความเร่งต่อ g = 133/2 = 66
- ค่าเมื่อวางในแนวราบ = 399-66 = 333



Angle Measurement







$$\theta_X = \text{asin}(a_x/g)$$

- สมมติว่า Arduino อ่านค่าได้ 333 เมื่อวาง accelerator ในแนวราบ และเปลี่ยนแปลง 66 ต่อ 1 g
- คำถาม : Arduino จะให้ output เท่าไร เมื่อวางบอร์ดเอียง 45 องศา?
  - ax = sin(45).g = 0.707g
  - = 333 + 0.707g * 66/g = 379

ADXL335 Buffering

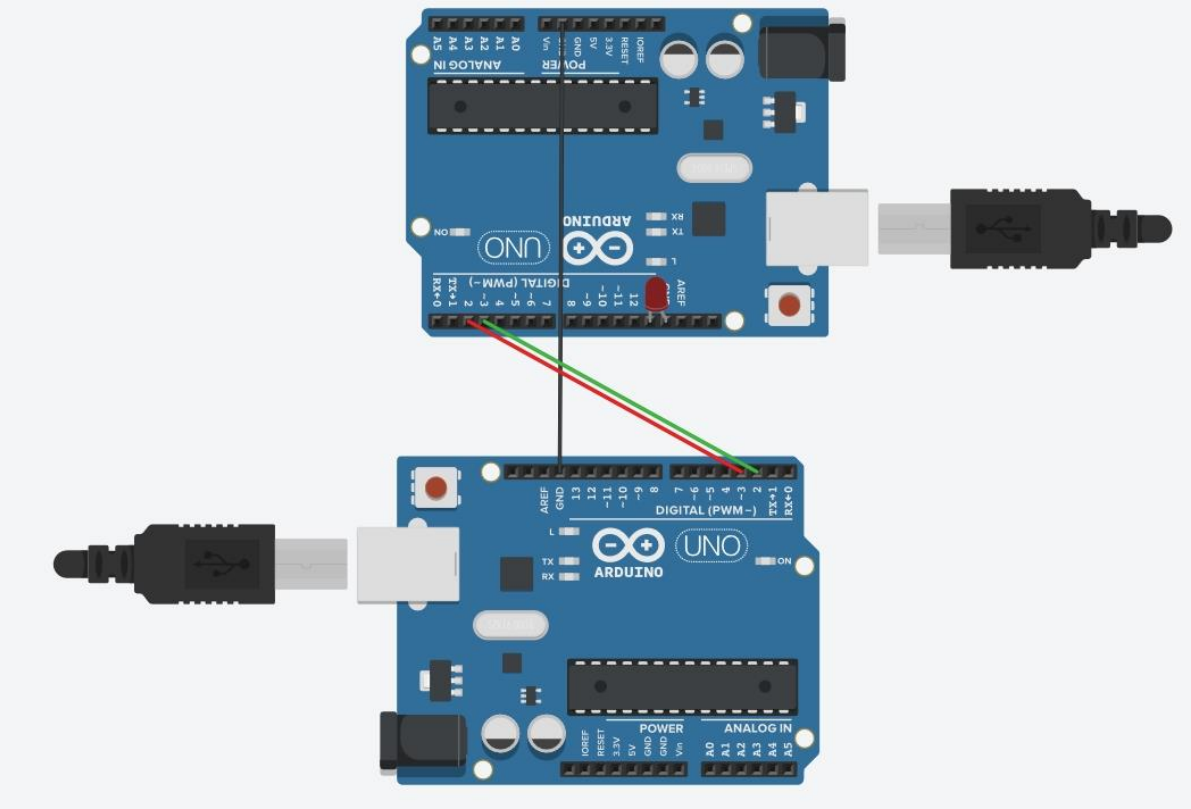
- ในการรับข้อมูลแต่ละครั้ง อาจมีการกระโดดของค่า (เรียกว่า jitter) ดังนั้นควรมีการเฉลี่ยค่า เพื่อให้ข้อมูลมีความนิ่งมากขึ้น

```
const unsigned int X_AXIS_PIN = 2;
const unsigned int Y_AXIS_PIN = 1;
const unsigned int Z_AXIS_PIN = 0;
const unsigned int NUM_AXES = 3;
const unsigned int PINS[NUM_AXES] = {
 X_AXIS_PIN, Y_AXIS_PIN, Z_AXIS_PIN
};
const unsigned int BUFFER_SIZE = 16;
const unsigned int BAUD_RATE = 9600;
int buffer[NUM_AXES][BUFFER_SIZE];
int buffer_pos[NUM_AXES] = { 0 };

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

int get_axis(const int axis) {
  delay(1);
  buffer[axis][buffer_pos[axis]] = analogRead(PINS[axis]);
  buffer_pos[axis] = (buffer_pos[axis] + 1) % BUFFER_SIZE;
  long sum = 0;
  for (unsigned int i = 0; i < BUFFER_SIZE; i++)
    sum += buffer[axis][i];
  return round(sum / BUFFER_SIZE);
}
```

```
int get_x() { return get_axis(0); }
int get_y() { return get_axis(1); }
int get_z() { return get_axis(2); }
void loop() {
  Serial.print(get_x());
  Serial.print(" ");
  Serial.print(get_y());
  Serial.print(" ");
  Serial.println(get_z());
}
```

Activity

- ให้นักศึกษาต่อ Arduino 2 บอร์ดเข้าด้วยกันตามรูป



Activity

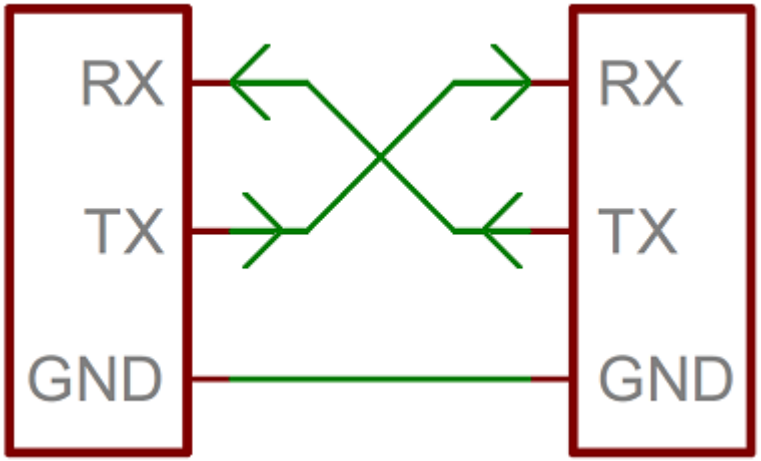
- ขา 2 เป็น Input ขา 3 เป็น Output โดยต่อครอสกันระหว่างขา 2 และ 3 ของทั้งสองบอร์ด
- ให้บอร์ดด้านบนรับข้อมูลจากบอร์ดด้านล่าง โดยรับเป็น 0 หรือ 1
- บอร์ดด้านล่างรับข้อมูลจากสวิตซ์ โดยถ้ากดสวิตซ์ให้ไฟที่ด้านบนติด

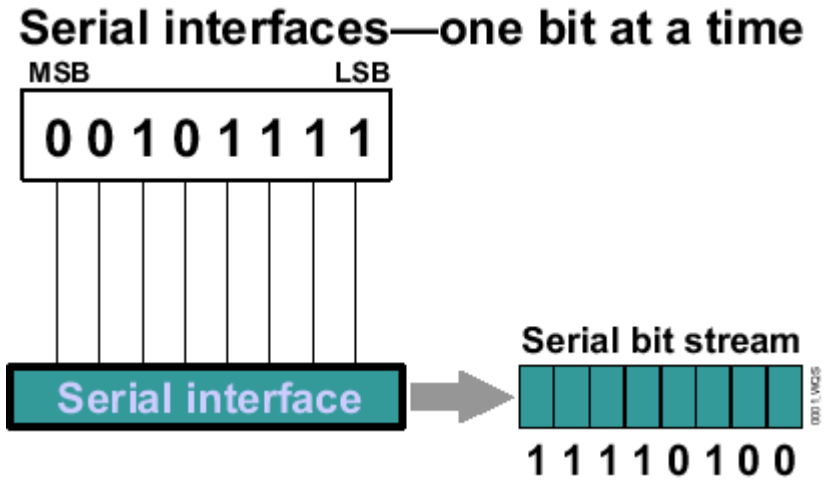
Disscussion

- ถ้าบอร์ดด้านบนต้องการส่งคำว่า Hello มาแสดงใน Serial Monitor ของบอร์ดด้านล่าง จะทำได้หรือไม่ ผ่านวงจรข้างต้น
- มีปัญหา หรือ อุปสรรคอะไร หรือไม่
- หากมี จะมีแนวทางการแก้ปัญหาอย่างไร

Serial Communication

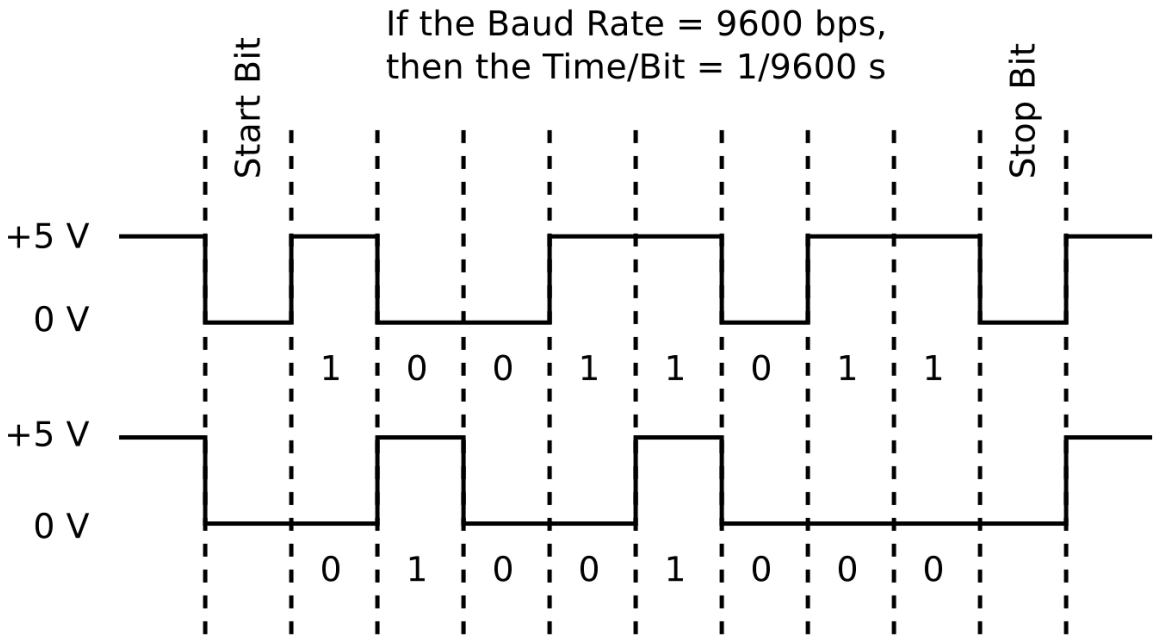
- เป็นการส่งข้อมูลครั้งละ 1 บิต โดย หากส่งด้านเดียวเรียก Simplex หากส่งได้ 2 ด้าน เรียก Duplex (หากรับส่งพร้อมกันเรียก Full Duplex หากรับส่งทีละข้าง เรียก Half Duplex)
- กรณีที่จะส่งข้อมูลหลายบิต จะต้องค่อยๆ เลื่อนมาทีละบิตเพื่อส่ง





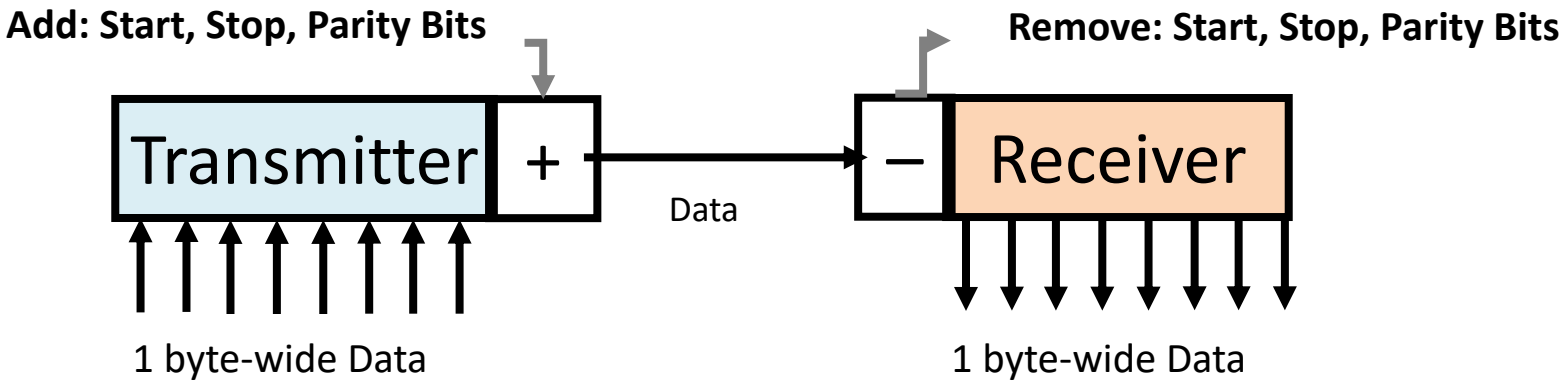
Serial Communication

- การส่งนิยมส่งเป็นชุด ชุดละ 1 ไบต์ (8 บิต)
- ในการส่งจะต้องมีการ Synchronize ระหว่าง 2 ข้างเสียก่อน เพื่อให้รู้ว่าจุดเริ่มของการส่งอยู่ที่ไหน (มักเรียกว่า start bit) (1->0) จากนั้นเป็นข้อมูล 8 บิต และตามด้วย stop bit
- คาบเวลาของแต่ละบิตจะต้องเท่ากันทั้งสองด้าน เรียกว่า baud rate



Serial Communication

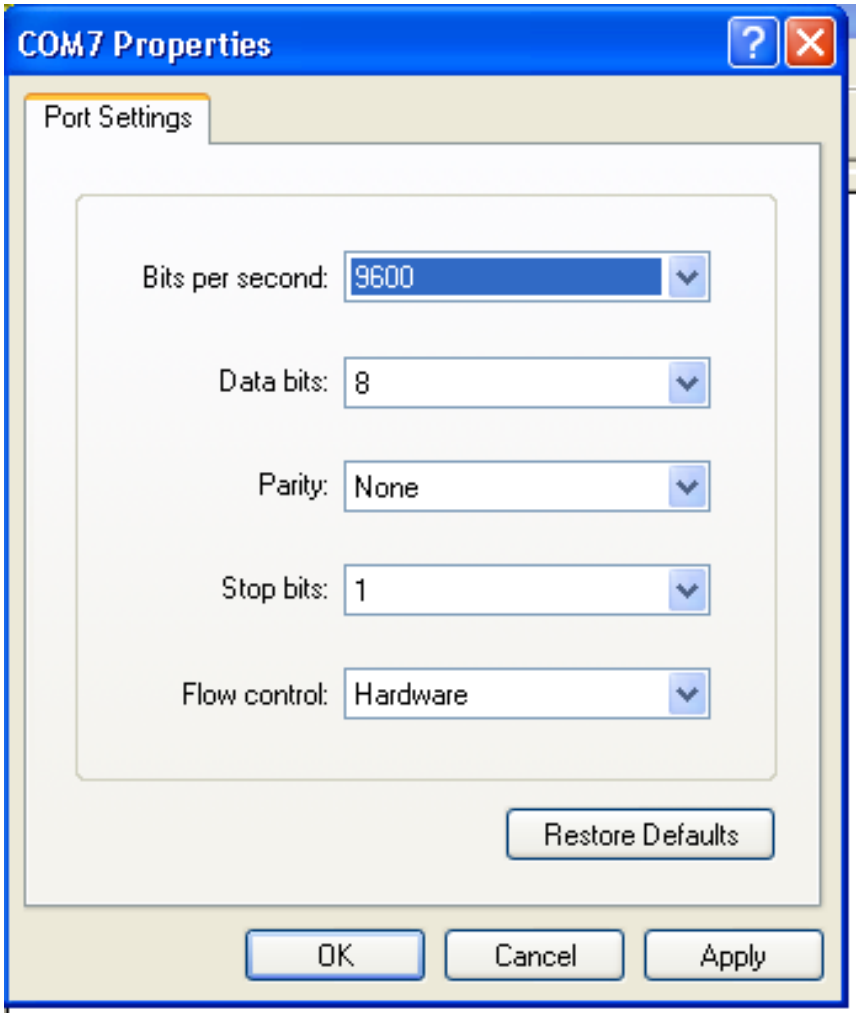
- ฝั่งรับและฝั่งส่ง จะมีเนื้อที่หน่วยความจำที่ใช้เพื่อการเลื่อนบิตรับและส่ง โดยจะเรียกหน่วยความจำส่วนนี้ว่า Transmit Buffer และ Receive Buffer (ขนาด Buffer อาจแตกต่างกันไปตามประเภทของ Hardware)
- Receive Buffer จะมีอีก 1 หน้าที่ คือ บอกว่า มีข้อมูลมาถึงหรือยัง
- สำหรับบอร์ด Arduino จะมี IC ที่ทำหน้าที่แปลง Serial <-> USB



ข้อกำหนดเพื่อให้การส่งข้อมูลถูกต้อง

- Baud Rate : (Bit per second)
- Data bits
- Parity bit
- Stop bit

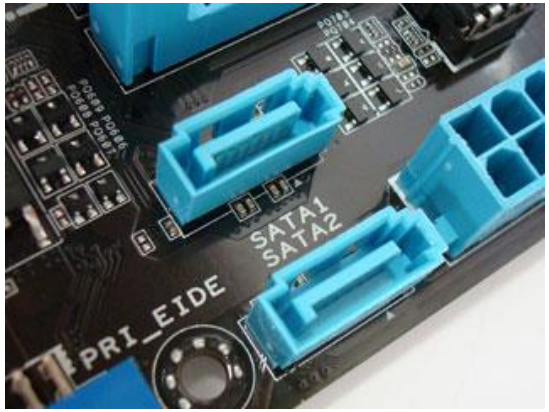
- 9600 – 8 – N - 1



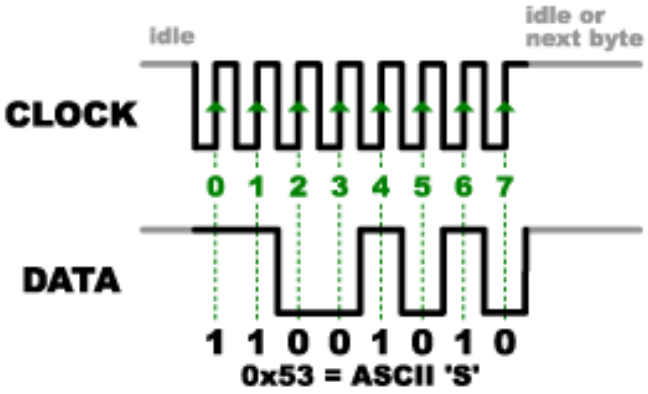
Serial Port

USB cable and port

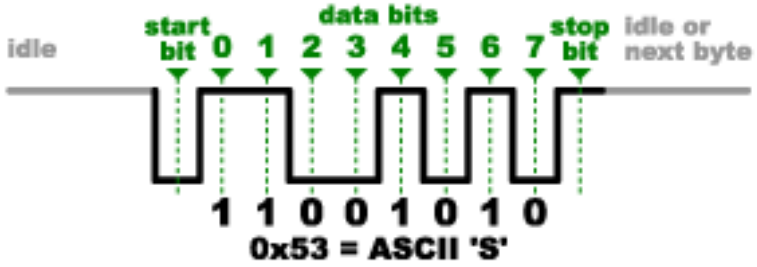
ComputerHope.com



Serial Communication



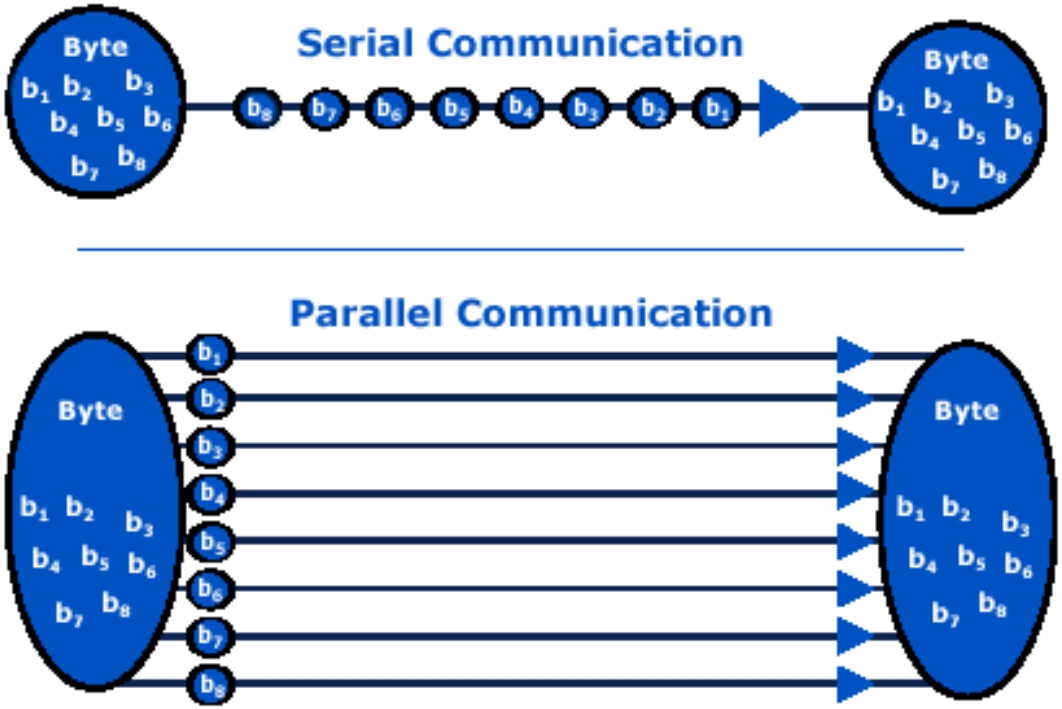
Synchronous



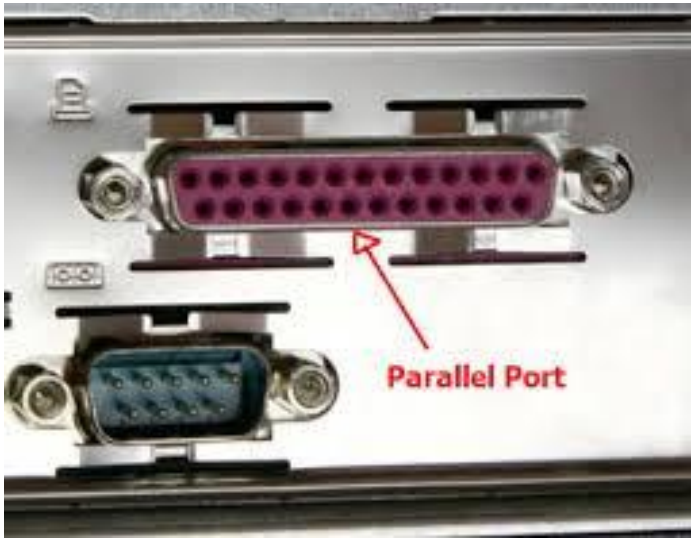
Asynchronous

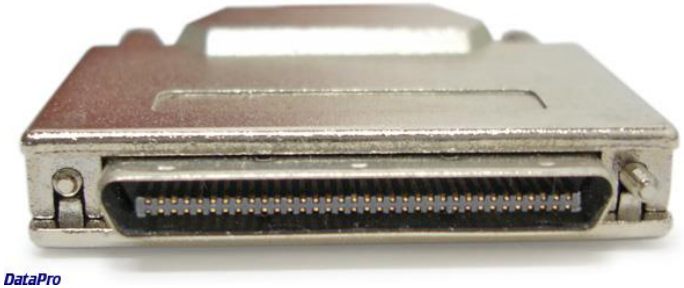
Parallel Communication

- การสื่อสารอีกรูปแบบหนึ่งที่มีการใช้กันมากในยุคก่อน คือ การสื่อสารแบบขนาน

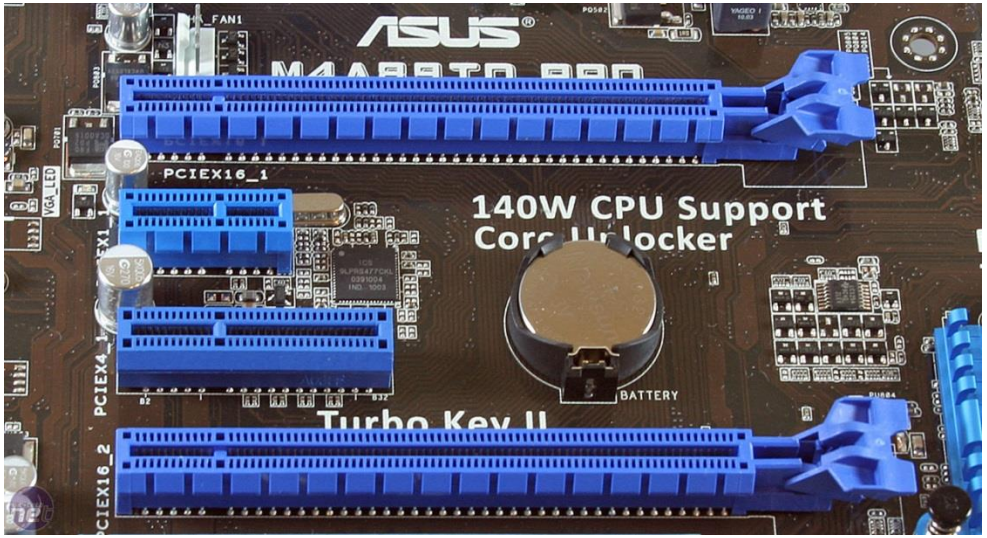


Parallel Port



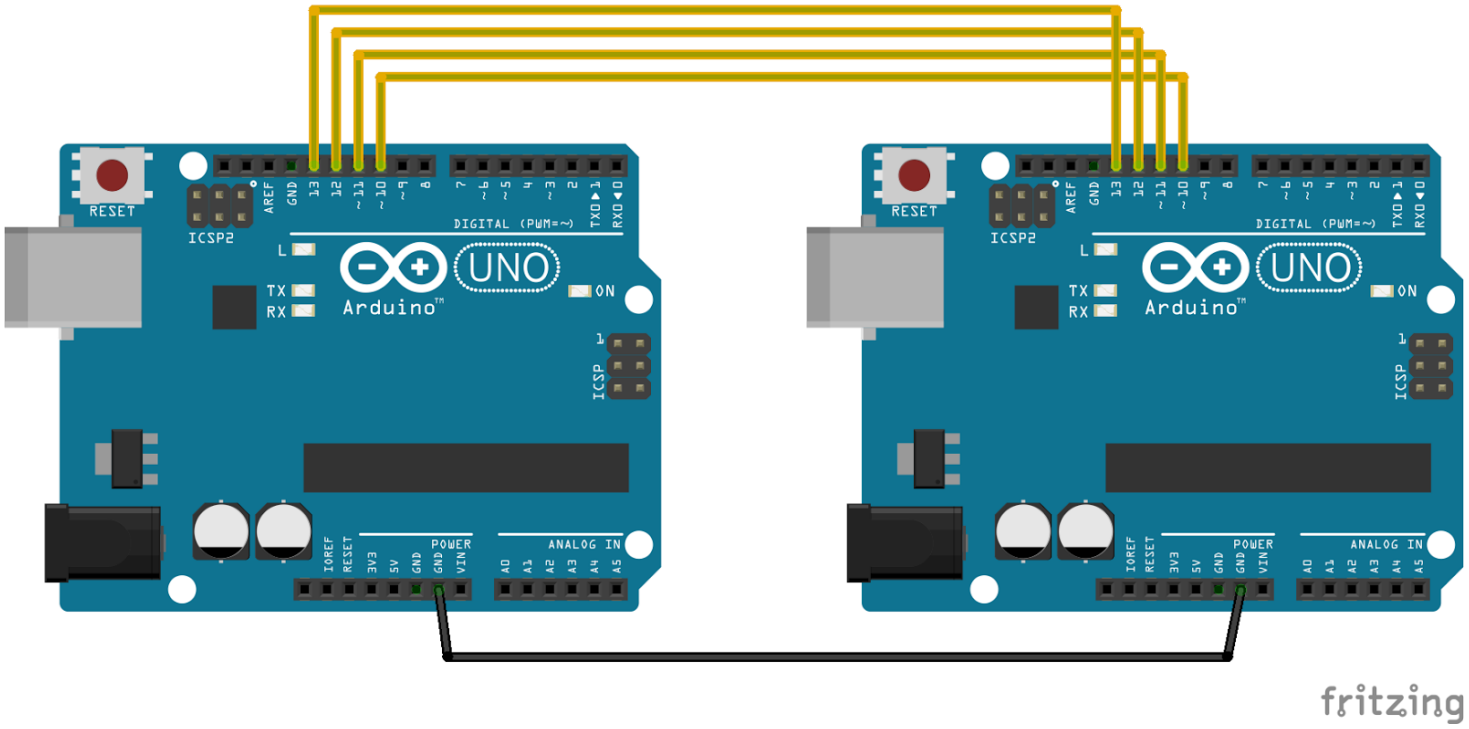


DataPro



Disscussion

- จากวงจรที่ผ่านมา หากต้องการส่งข้อมูลแบบขนานสามารถทำได้หรือไม่
- คิดว่าจะมีปัญหา อุปสรรคใดหรือไม่ และจะเลือกแบบไหนใช้งาน เพราะเหตุใด



Discussion

ระหว่าง Parallel กับ Serial อะไรส่งข้อมูลได้เร็วกว่ากัน

Serial Function

Serial.available()

- คืนค่าจำนวนไบต์ (ตัวอักษร) ที่อยู่ใน Receive Buffer ที่พร้อมจะให้อ่านออกมาได้ โดย Receive Buffer จะมีขนาด 64 ไบต์
- มักจะใช้คู่กับฟังก์ชัน Serial.read() ตามตัวอย่าง โดยฟังก์ชัน read จะส่งค่าไบต์แรกที่สามารถอ่านออกมาได้ (-1 ถ้าไม่มีข้อมูลให้อ่าน)

```
while (Serial.available()) // recheck serial is available
{
  char inChar = (char)Serial.read(); // get the new byte:
}
```

Serial Function

Example:

```
void setup()
{
  Serial.begin(9600);
}

void loop()
{
        while (Serial.available()) // recheck serial is available
        {
                char inChar = (char)Serial.read();          // get the new byte:
                Serial.print(inChar);
        }
 }
```

Activity

- เขียนโปรแกรมรับข้อมูลจาก Serial Monitor โดยรับเป็นตัวเลข 1-9
- สั่งให้ไฟที่ขา 13 กระพริบตามจำนวนตัวเลขที่รับมา

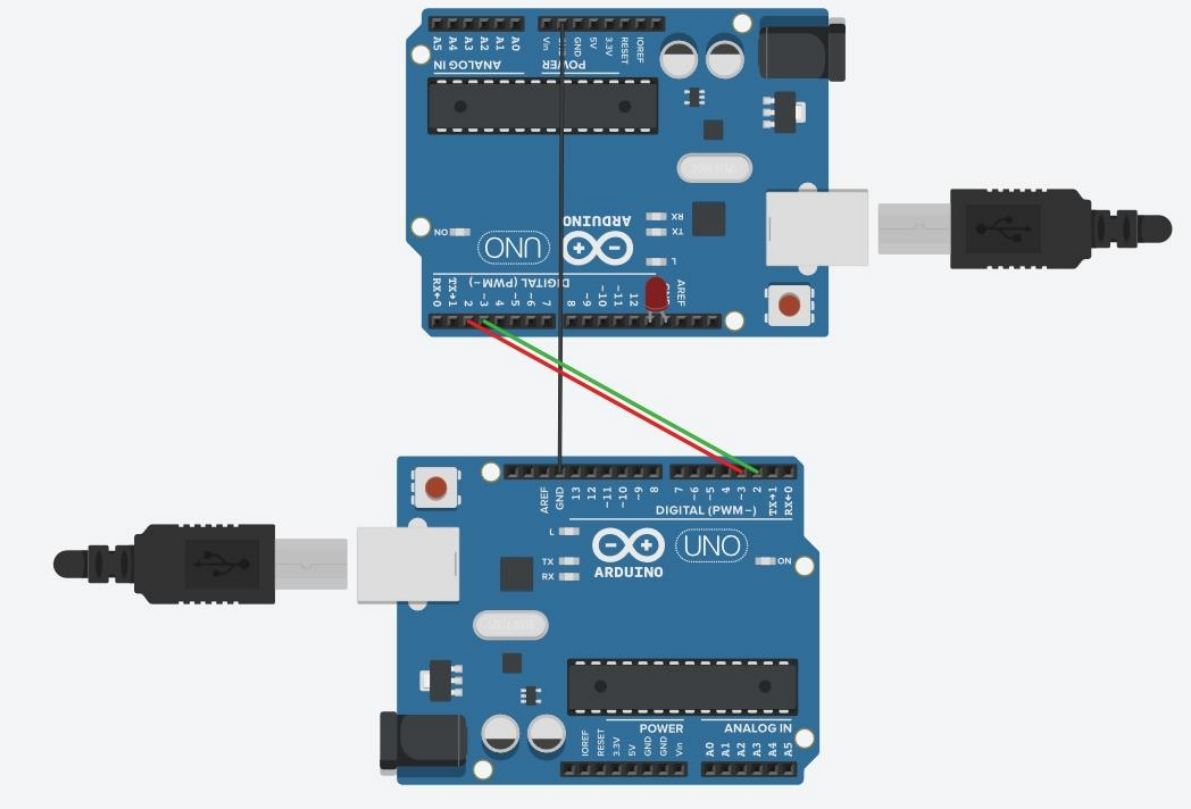
Serial Event

- นอกเหนือจากการวนลูปเพื่อตรวจสอบข้อมูลที่ส่งเข้ามาแล้ว ยังใช้กลไกการ Interrupt เพื่อตรวจสอบได้เช่นเดียวกัน

```
void setup()
{
    Serial.begin(9600); // initialize serial:
    pinMode(13,OUTPUT);
}
void loop()
{
    Serial.println("Wait for command");
    delay(500);
}
void serialEvent()
{
    while(Serial.available()) {
        char inChar=(char)Serial.read(); // get the new byte:
        if(inChar=='1') { // check received 'enter' (0x0D)
            digitalWrite(13,HIGH);
        }
        else if(inChar=='0') {
            digitalWrite(13,LOW);
        }
     }
}
```

Activity

- แก้ไขโปรแกรมที่ส่งข้อมูลระหว่างบอร์ดจาก Polling เป็น Software Event



Software Serial

- ในบอร์ด Arduino จะมี Serial (ที่เป็น Hardware) อยู่เพียงชุดเดียว ทำให้ต้องเลือกว่า
  - ต่อกับ Computer เพื่อ Upload หรือใช้ Serial Monitor
  - ต่อกับอุปกรณ์อื่นๆ
- จะใช้กันพร้อมกัน 2 ฟังก์ชันไม่ได้
- จึงมีการพัฒนาซอฟต์แวร์ที่สามารถทำให้ขา GPIO ใดๆ ใช้เป็น Serial ได้

Software Serial

```
#include <SoftwareSerial.h>
String inputString = ""; // a string to hold incoming data
SoftwareSerial mySerial(10,11); // SoftwareSerial(rxPin, txPin)

void setup()
{
    // Open serial communications and wait for port to open:
    Serial.begin(4800);
    Serial.println("Hello World");
    // set the data rate for the SoftwareSerialport
    mySerial.begin(4800); // recommentlow speed
    mySerial.println("Software Serial->Hello, world?");
}
void loop() // run over and over
{
    if(mySerial.available())
    {
        Serial.print((char)mySerial.read());
    }
}
```

Software Serial

```
void serialEvent()
{
     while(Serial.available()) // recheck serial is available
     {
          char inChar=(char)Serial.read(); // get the new byte:
          inputString+=inChar; // add it to the inputString:
          if (inChar=='\r') // check received 'enter' (0x0D)
          {
                mySerial.print("TX from Software serial -> ");
                mySerial.println(inputString);
                inputString="";
          }
     }
}
```

Software Serial Interrupt

- เป็นการปรับปรุงให้การรับข้อมูลใช้วิธี Interrupt

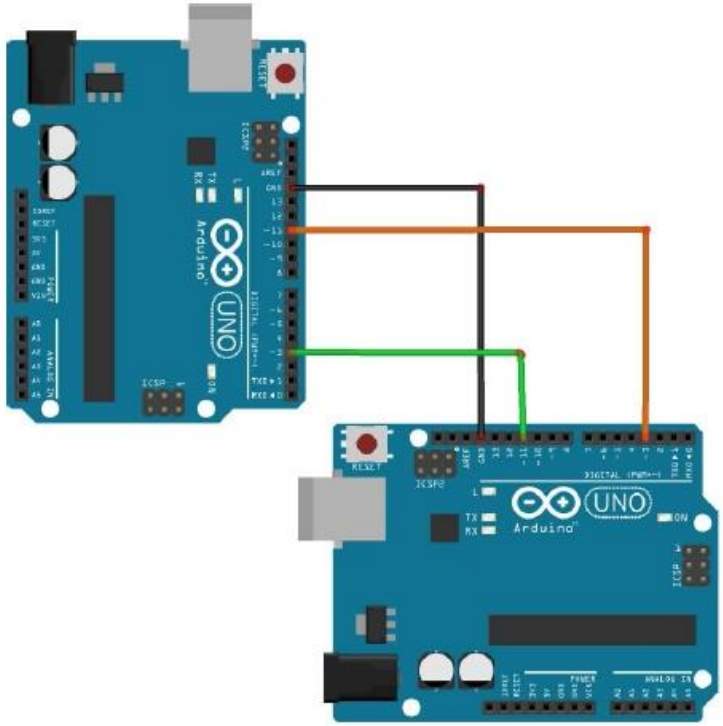
```
#include <SoftwareSerial.h>
String inputString = ""; // a string to hold incoming data
SoftwareSerial mySerial(3,11); // SoftwareSerial(rxPin, txPin)

void setup()
{
    // Open serial communications and wait for port to open:
    Serial.begin(4800);
    Serial.println("Hello World");
    // set the data rate for the SoftwareSerialport
    mySerial.begin(4800); // recommentlow speed
    mySerial.println("Software Serial->Hello, world?");

    // attachInterrupt(interrupt, ISR, mode) interrupt-> 1(pin3)
    attachInterrupt(1,SoftwareSerialEvent,FALLING);
}
void loop() // run over and over
{

}
```

Software Serial Interrupt

```
void serialEvent()
{
     while(Serial.available()) // recheck serial is available
     {
          char inChar=(char)Serial.read(); // get the new byte:
          inputString+=inChar; // add it to the inputString:
          if (inChar=='\r') // check received 'enter' (0x0D)
          {
                mySerial.print("TX from Software serial -> ");
                mySerial.println(inputString);
                inputString="";
          }
     }
}


void SoftwareSerialEvent()
{
      if(mySerial.available()) // test this condition by connecting pin rxsoftware with pin'0'(Rx)
      {
          Serial.print((char)mySerial.read());
      }
}
```

Activity

- ให้นำบอร์ด Arduino 2 บอร์ดต่อกันตามรูป ให้เขียนโปรแกรม Serial Chat โดยป้อนข้อมูลคุยกันระหว่างคอมพิวเตอร์ 2 เครื่อง (ใช้ขา D11,D3) ผ่าน Serial Monitor



Assignment #5 : Level Meter

- สร้างเครื่องวัดความเอียง โดยใช้ 2 บอร์ด บอร์ดหนึ่งให้วัดความเอียงโดยใช้ Accelerometer อีกบอร์ดให้เป็นส่วนแสดงผลโดยใช้ LED 8x32 โดยเมื่อนำไปวางเป็นพื้นผิวแล้ว สามารถบอกความเอียงเป็นองศาได้
- ต้องมีการ Calibrate ก่อนใช้งาน ในการทดสอบจะใช้ App ในโทรศัพท์มือถือเป็นตัวเทียบ
- ต้องมีการปรับค่าให้นิ่ง ไปเปลี่ยนไปมา กรณีที่ไม่ได้ขยับอุปกรณ์
- การตรวจเทียบกับ App ในโทรศัพท์ โดยค่าต้องมีความนิ่ง
- คะแนน : 3 คะแนน การส่ง 1) VDO แสดงการทำงาน 2) รายงานอธิบายแนวคิดและโครงสร้างของโปรแกรม 3) Code

For your attention