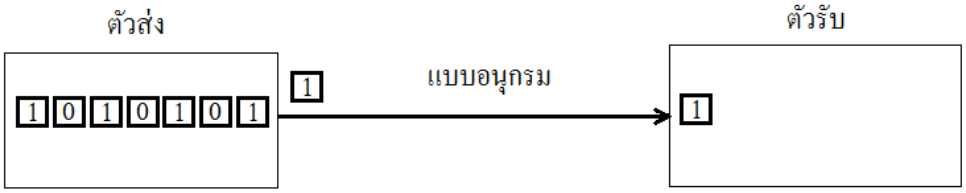
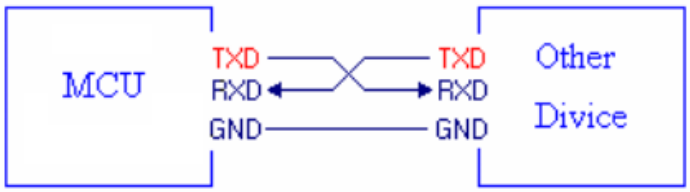
# การสื่อสารทางพอรตอนุกรม

การสื่อสารข้อมูลทางพอรตอนุกรม คือการสงข้อมูลครั้งละ เวลา การสงและรับข้อมูลจะกระทำได ครั้งละ 1 บิต ซึ่งแตกตางจากการสื่อสารทางพอรตขนานที่ สามารถสงข้อมูลครั้งละหลายๆ บิต พรอมๆ กันได จึงทำให้การสงขอมูลดวยพอรตอนุกรมจะชา กวาการสงข้อมูลดวยพอรตขนาน แตข้อดีของการสื่อสารขอมูลทางพอรตอนุกรมคือ สามารถสง ขอมูลไดในระยะทางที่ไกลกวา และใชจำนวนสายสัญญาณที่น้อยกวาการสื่อสารขอมูลแบบขนาน



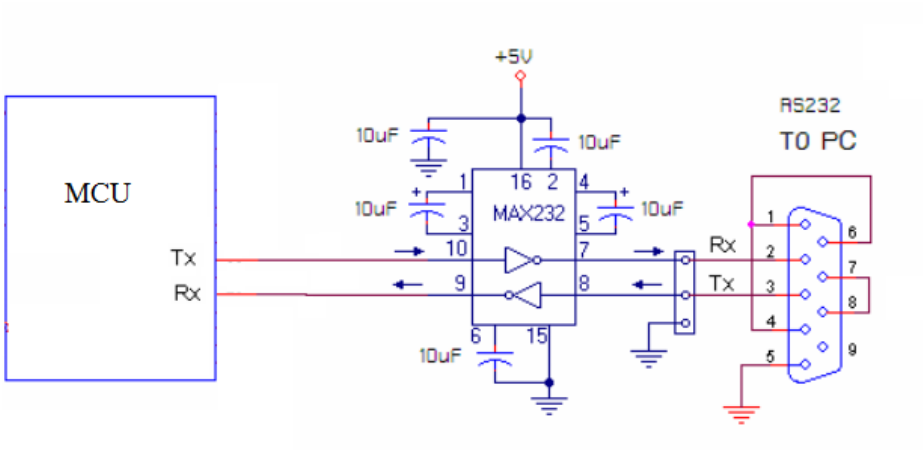

**รูปที่ 1** ลักษณะการสื่อสารข้อมูลแบบอนุกรม

การสื่อสารระหวางไมโครคอนโทรลเลอรกับอุปกรณอื่นๆ ผานทางพอรตสื่อสารอนุกรมที่มีระดับแรงดันเดียวกัน คือ +5 โวลท และ 0 โวลท สามารถตอถึงกันโดยตรง ขาสง สัญญาณ (TX) ของไมโครคอนโทรลเลอร จะตองตอกับขารับสัญญาณ (RX) ของอุปกรณอื่น และขารับสัญญาณ (RX) ของไมโครคอนโทรลเลอรจะต่องตอกับขาสงสัญญาณ (TX) ของอุปกรณอื่น ดังวงจรในรูปที่ 2




**รูปที่ 2** การเชื่อมต่อสื่อสารข้อมูลแบบอนุกรม

หากไมโครคอนโทรลเลอรตองการติดตอกับอุปกรณอื่นที่มีระดับแรงดันแตกตางกัน เชนการสื่อสารกับคอมพิวเตอรทางพอรตอนุกรม ระดับแรงดันพอรตอนุกรมของคอมพิวเตอรนั้น จะมีระดับแรงดัน -15 ไมโครคอนโทรลเลอรกับพอรตอนุกรม RS232 ของคอมพิวเตอรไดโดยตรง จะต่องใชไอซีเปลี่ยนระดับสัญญาณเชน เบอร MAX232 เปนตัวเปลี่ยนระดับสัญญาณ + - 15 โวลท เปนระดับสัญญาณ TTL ดังวงจรในรูปที่ 3




**รูปที่ 3** การสื่อสารข้อมูลแบบอนุกรมกับพอร์ตอนุกรมของคอมพิวเตอร์

## การสื่อสารขอมูลแบบอะซิงโครนัส

การสื่อสารข่อมูลอนุกรมของไมโครคอนโทรลเลอรมีลักษณะเปนแบบอะซิงโครนัส (Asynchronous) คือการรับสงข่อมูล โดยที่ไมใชสัญญาณนาฬิกา (Clock) เปนจังหวะในการสง ข่อมูล แตจะกำหนดความเร็วของตัวสงและตัวรับข่อมูล (Baud rate) ใหมีอัตราที่เทากัน ซึ่งเปน จำนวนบิตตอวินาที เชน 300, 1,200, 2,400, 4,800 , 9,600 ,14,400 ,19,200, 38,400 ,56,000 เปนตน โดยรูปแบบขอมูลแบบอะซิงโครนัส ประกอบดวย

1. บิตเริ่มตน (Start bit) มีขนาด 1 บิต

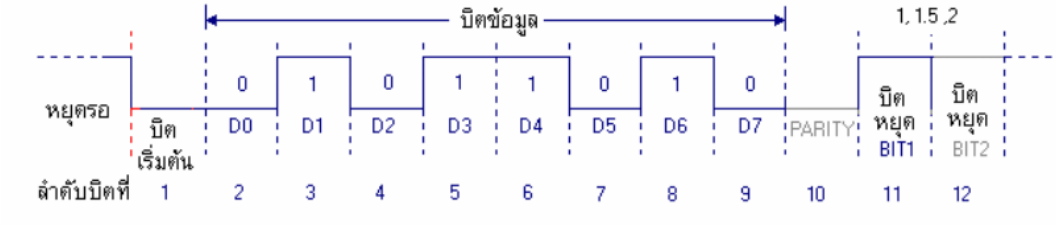
2. บิตขอมูล (Data) มีขนาด 5,6,7 หรือ 8 บิต

3. บิตตรวจสอบพาริตี้ (Parity bit) มีขนาด 1 บิตหรือไมมี

4. บิตหยุด (Stop bit) มีขนาด 1, 1.5, 2 บิต

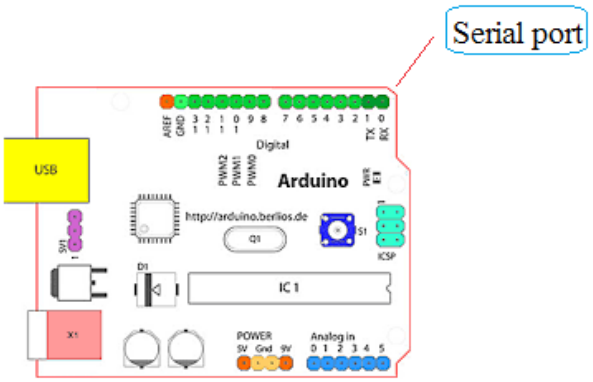
การสงข่อมูลของระบบสื่อสารอนุกรมแบบอะซิงโครนัสมีลำดับการสงข่อมูลคือ เมื่อไมมีการสงข่อมูล ขา Tx จะมีสถานะเปนโลจิก "1" หรือ สถานะหยุดรอ (Waiting stage) เมื่อเริ่มตน สงขอมูลจะทำใหขา Tx เปลี่ยนเปนลอจิก "0" จำนวน 1 บิต เรียกวาบิตเริ่มตน (Start bit) จากนั้นก็ จะเริ่มตนสงขอมูล โดยสงบิตต่ำไปกอน (LSB) จนครบขอมูลจำนวน 8 บิต แลวตามดวยพาริตี้บิต ซึ่งอาจจะมีหรือไมมีก็ได ขึ้นอยูกับการกำหนดคา ของทั้งตัวสงและตัวรับสัญญาณ และตามดวยการ สงบิตหยุด (Stop bit) ซึ่งเปนลอจิก "1" ซึ่งอาจ

กำหนดบิตหยุด จำนวน 1, 1.5 หรือ 2 บิตเปนการ สิ้นสุดการสงขอมูล ทั้งนี้ตองกำหนดใหตรงกันทั้งฝายสงและฝายรับ



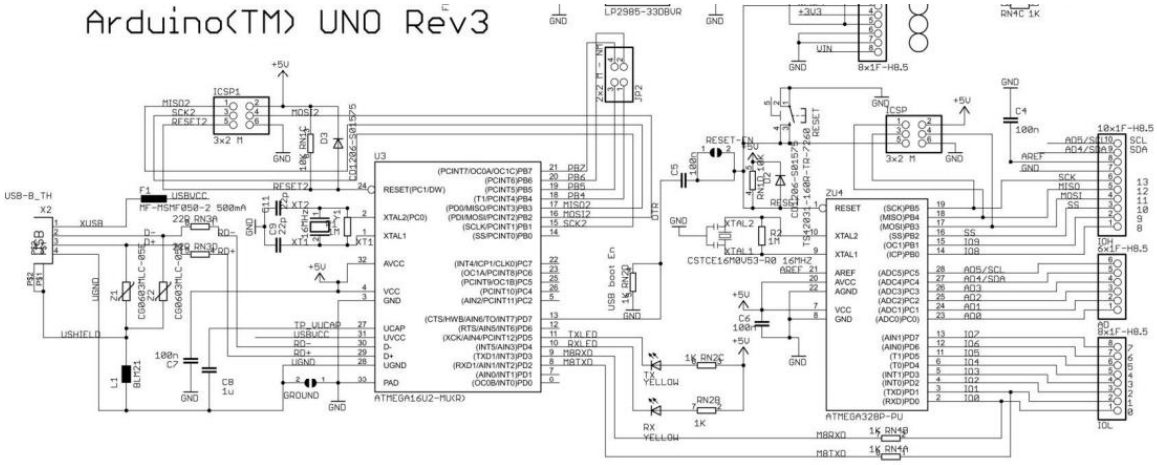

**รูปที่ 4** การสื่อสารข้อมูลแบบอนุกรม

สำหรับบอร์ด Arduino R3 พอร์ตสื่อสารอนุกรม คือขา D0 ทำหน้าที่ Rx รับข้อมูลจากอุปกรณ์อื่น และ D1 ทำหน้าที่ Tx ส่งข้อมูลให้กับอุปกรณ์อื่น ในรูปแบบการสื่อสารอนุกรม




**รูปที่ 5** พอร์ตอนุกรมของบอร์ด Arduino

ส่วนการเชื่อมต่อระหว่าง MCU กับพอร์ตคอมพิวเตอร์ผ่านทางพอร์ต USB บนบอร์ด Arduino จะมีไอซีอีกตัวทำหน้าที่รับส่งข้อมูลผ่านต่อไปยังคอมพิวเตอร์ โดยมีลักษณะวงจรดังรูป




**รูปที่ 6** วงจรของบอร์ด Arduino

**รูปที่ 7** การเชื่อมต่อกับพอร์ต USB

ดังนั้นจึงสามารถใช้คอมพิวเตอร์สื่อสารแบบอนุกรมกับ MCU บนบอร์ด Arduino R3 ได้อย่างสะดวก เนื่องจากใช้พอร์ตเดียวกันกับการ Up Load โปรแกรมลงในบอร์ด สามารถใช้คำสั่งจากไมโครคอนโทรลเลอร์เพื่อแสดงผล หรือค่าข้อมูลจากภายใน MCU บนจอคอมพิวเตอร์ ผู้พัฒนาโปรแรมเห็นการเปลี่ยนค่าข้อมูลที่เกิดขึ้นในตัวแปรของ MCU ทำให้สามารถแก้ปัญหา ด้านการเขียนโปรแกรมได้สะดวกยิ่งขึ้น

## ฟังก์ชั่นการสื่อสารอนุกรม

1. ฟังก์ชั่นกำหนดความเร็ว

**Serial.begin(speed);**

เป็นฟังก์ชั่นกำหนดอัตราความเร็วของการสื่อสารที่เรียกกันว่า อัตราบอด (Baud rate) มีหน่วยเป็นบิตต่อวินาที (bps : bit per second) ซึ่งต้องกำหนดให้เท่ากันระหว่างตัวส่งและตัวรับ โดยมีอัตราความเร็วให้เลือกใช้ตั้งแต่ 300,600,1200,2400,4800,9600,14400,19200,28800,38400,57600, และ 115200 bps ซึ่งยิ่งความเร็วยิ่งสูง ความผิดพลาดในการรับส่งข้อมูลก็จะสูงเช่นเดียวกัน ดังนั้นผู้ใช้จึงต้องเลือกใช้โดยพิจารณาจากปริมาณข้อมูลในการรับส่ง ระยะทางของสาย และชนิดของสายตัวนำ เป็นต้น ตัวอย่างการใช้ฟังกชั่นกำหนดความเร็วเช่น

```
Serial.begin(9600);
```

หมายถึงการกำหนดอัตราความเร็วการสื่อสารพอร์ตอนุกรมด้วยความเร็ว 9600 บิตต่อวินาที (bps) ซึ่งอุปกรณ์ที่เชื่อมต่อด้วยก็ต้องกำหนดความเร็ว 9600 bps เช่นเดียวกัน

**2. ฟังก์ชั่นตรวจสอบข้อมูล**

**Serial.available();**

เป็นฟังก์ชั่นตรวจสอบว่ามีข้อมูลส่งเข้ามาที่ขา Rx หรือไม่ โดยปกติหากไม่มีข้อมูลเข้า เอาท์พุตของฟังก์ชั่นนี้ จะเป็น 0 แต่หากมีข้อมูลเข้ามาจะมีค่า มากกว่า 0 ดังนั้นในการนำไปใช้ตรวจสอบจะใช้ร่วมกับคำสั่ง if เช่น

```
 if(Serial.available() > 0)  {
      อ่านข้อมูลที่เข้ามา
}
```

เมื่อเอาท์ของฟังก์ชั่น Serial.available() มากกว่า 0 จะทำการอ่านข้อมูลที่เข้ามา แต่หาก เป็น 0 ก็จะข้ามการอ่านข้อมูล

**3. ฟังก์ชั่นอ่านข้อมูล**

**Serial.read();**

เป็นฟังก์ชั่นอ่านค่าข้อมูลที่เข้ามาทางขา Rx ซึ่งสามารถนำไปใช้อ่านข้อมูลหลังจากการตรวจสอบของฟังก์ชั่นก่อนนี้ โดยการใช้งานจะต้องมีตัวแปรมาทำหน้ารับค่าข้อมูลที่อ่านมาได้เสมอ เช่น

```
x = Serial.read();
```

หมายถึง อ่านค่าข้อมูลจากพอร์ตอนุกรม Rx มาเก็บในตัวแปร x

ซึ่งหากนำมาใช้ร่วมกับฟังก์ชั่นตรวจสอบ ก็จะได้เป็นการเขียนโปรแกรมดังนี้

```
if(Serial.available() > 0) {
    x = Serial.read();
}
```

**4. ฟังก์ชั่นลบข้อมูลในหน่วยความจำบัฟเฟอร์**

**Serail.flush();**

เป็นการลบข้อมูล ของหน่วยความจำที่ทำหน้าที่เป็นบัฟเฟอร์รับข้อมูลจากขา Rx

**5. ฟังก์ชั่นส่งข้อมูล**

**Serial.print(val);**

**Serail.print(val,format);**

เป็นฟังก์ชั่นการส่งข้อมูลออกทางขา Tx ไปยังอุปรณ์เชื่อมต่ำภายนอก ซึ่งอาจจะเป็น คอมพิวเตอร์ SMS โมดูล หรืออุปกรณ์อื่นๆที่รับข้อมูลแบบอนุกรม โดยข้อมูลเป็นค่าที่อยู่ในวงเล็บ (val) จะถูกส่งออกไปในรูปแบบตัวอักษร Character หากตัวรับเป็นคอมพิวเตอร์หากส่งออกไปด้วยคำสั่ง Serail.plrint(val); จะแสดงข้อมูลบนจอภาพดังนี้

| | |
|---|---|
| Serial.print(89); | 89 |
| Serial.print(15.56475); | 15.56 |
| Serial.print('S'); | S |

| | |
|---|---|
| Serial.print(“HELLO”); | HELLO |
| Serial.print(59,BIN); | 01011001 |
| Serial.print(59,DEC); | 59 |
| Serial.print(59,HEX); | 3B |
| Serial.print(15.56475,2); | 15.56 |
| Serial.print(15.56475,3); | 15.565 |

6. **ฟังก์ชั่นส่งข้อมูลบรรทัดใหม่**

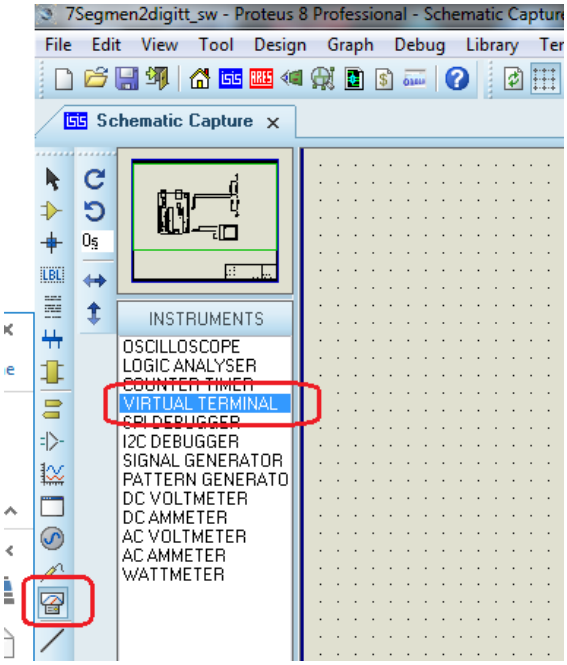
**Serial.println(val);**

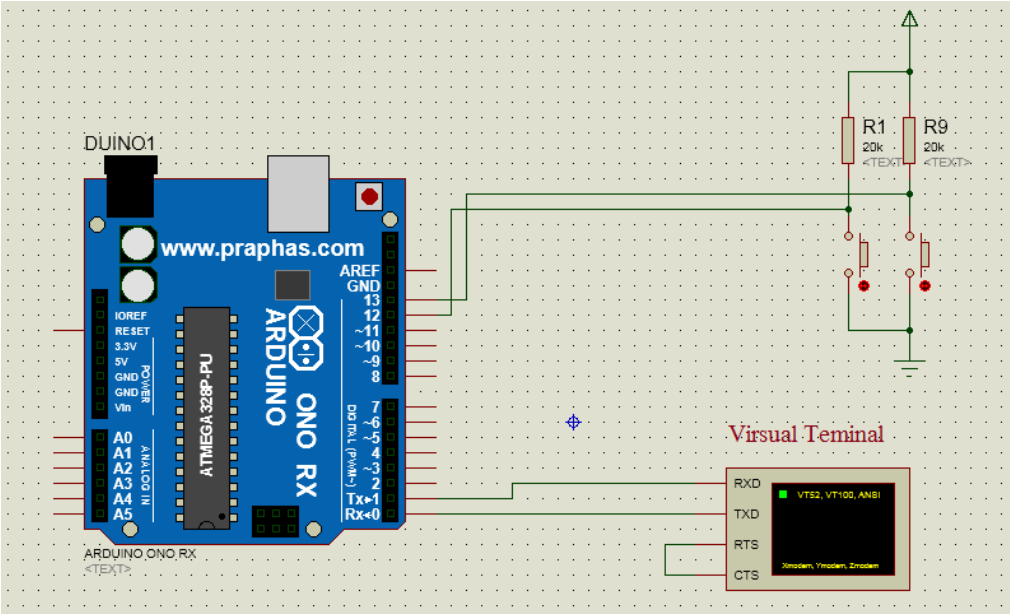
**Serial.println(val,format);**

เป็นฟังก์ชั่นที่ทำงานเหมือนกับฟังก์ชั่น Serial.print(val); และ Serial.print(val,format); เพียงแต่ฟังก์ชั่นนี้หลังจากแสดงข้อมูลเสร็จแล้ว จะขึ้นบรรทัดใหม่

**การจำลองในโปรแกรม Proteus**

การจำลองการทำงานด้วยโปรแกรม Proteus สามารถใช้เครื่องมือ Virtual Terminal แทนการแสดงผลบนจอคอมพิวเตอร์ได้ การแสดงข้อความใดๆใน Virtual Terminal คือข้อความที่แสดงผลบนจอคอมพิวเตอร์เมื่อต่อวงจรจริง



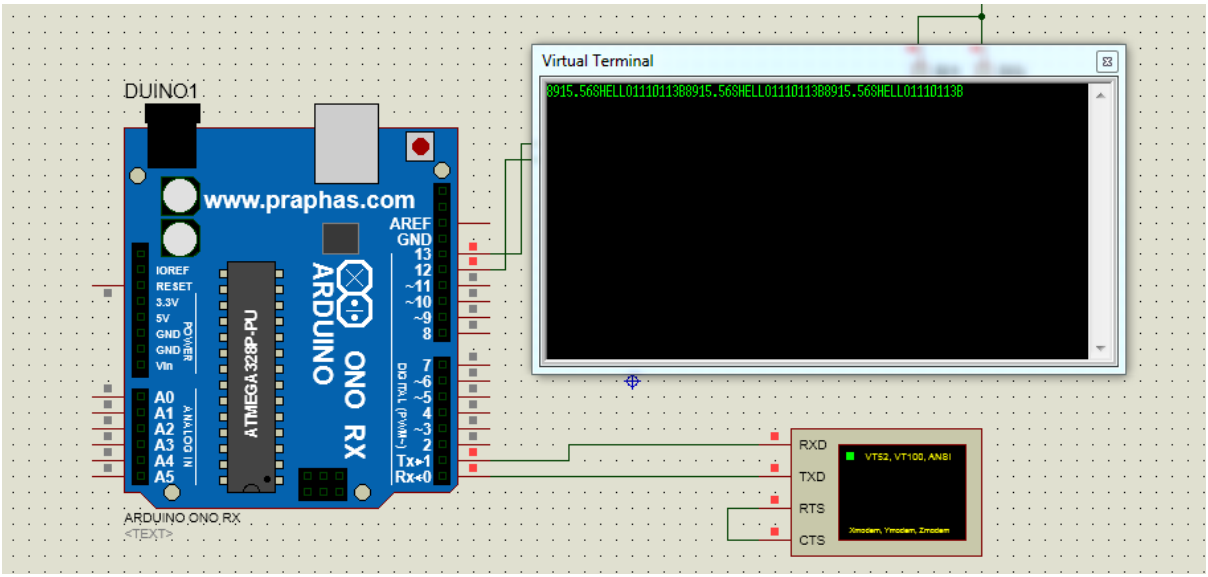




ตัวอย่างโปรแกรมบน IDE Arduino

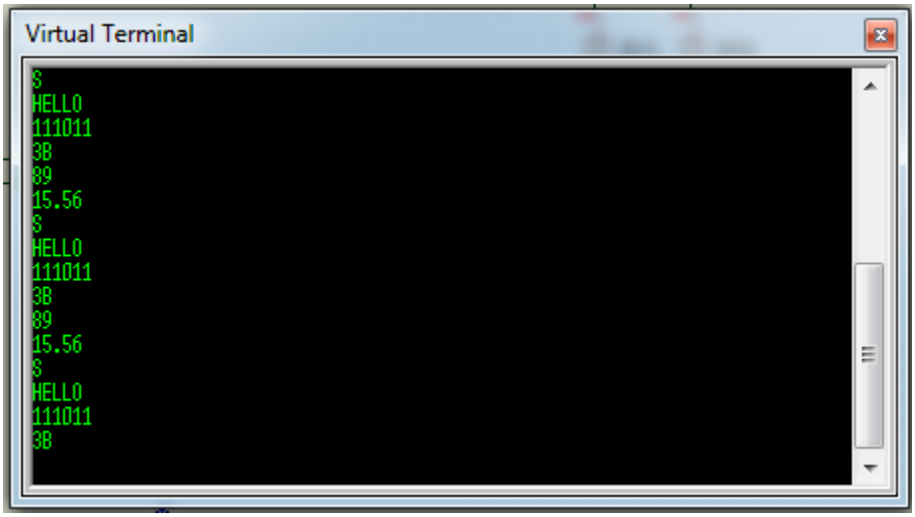
```
void setup() {

 Serial.begin(9600);

}
```

```
void loop() {
  Serial.print(89);
  Serial.print(15.56475);
  Serial.print('S');
  Serial.print("HELLO");
  Serial.print(59,BIN);
 Serial.print(59,HEX);
 delay(2000);
}
```

เมื่อทำการคอมไพล์แล้วรันบนโปรแกรมจำลอง Proteus



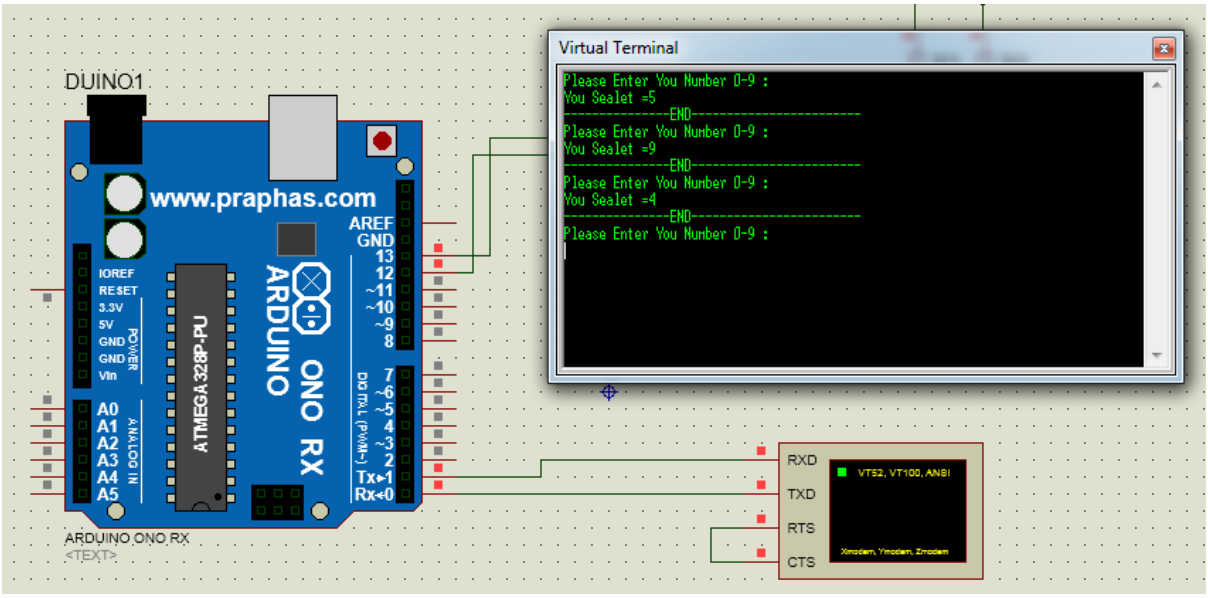

ทดลองเปลี่ยนฟังก์ชั่น Serial.print(val); เป็น Serial.println(val); การแสดงผลบนจอภาพ เมื่อแสดงข้อความของแต่ละคำสั่งเสร็จสิ้น ก็จะขึ้นบรรทัดใหม่เสมอ ดังแสดงในรูป




**การตรวจสอบและอ่านข้อมูลจาก Rx**

```
void setup() {
 Serial.begin(9600);
 Serial.println("Please Enter You Number 0-9 :");
}
void loop() {
  char data;
 if(Serial.available()>0)       // ตรวจสอบข้อมูลเข้าหรือไม่
  {
    data = Serial.read();      // อ่านข้อมูลเก็บไว้ในตัวแปร data
    Serial.print("You Select =");       // แสดงข้อความบนจอภาพ
    Serial.println(data);       // แสดงค่าข้อมูลของตัวแปร data
    Serial.println("----------END----------");
```

```
    Serial.println("Please Enter You Number 0-9 :");

  }

}
```



# แบบฝึกหัด

จงเขียนโปรแกรม ตรวจสอบการ SW1 และ SW2 แบบ Toggle SW โดยให้ LED1 , LED2 ติดสว่าง เมื่อถูกกด แล้วหาก กด SW1 และ SW2 อีกครั้ง ให้ LED แต่ละดวงดับ พร้อมแสดง สถานะการติด หรือดับ บนหน้าจอคอมพิวเตอร์