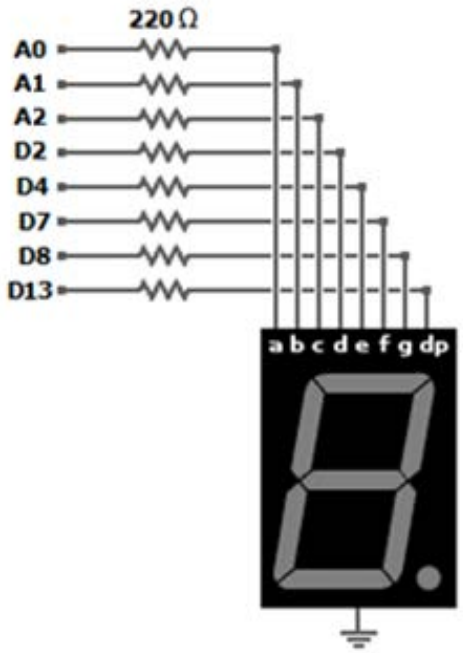
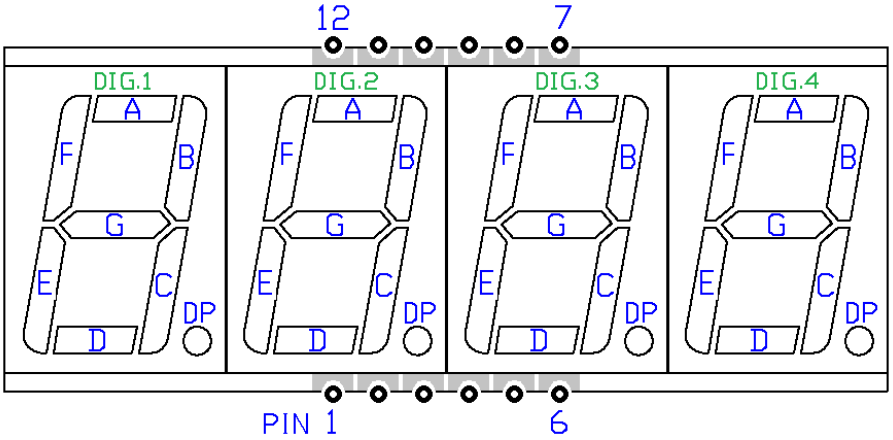
7-Segment การต่อวงจรเพื่อสั่งงานให้ไมโครคอนโทรลเลอร์แสดงผลของข้อมูลต่างๆ เป็นตัวเลขออก LED ที่เป็น 7-Segment แบบ 4 หลัก โดยใช้ LED ที่เป็นไดโอดเปล่งแสง จะต้องต่อขา Cathode คือขา Digit1 เข้ากับขั้วไฟลบหรือลงกราวด์ และจ่ายไฟเข้าขั้ว Anode ในแต่ละขาเป็นไฟบวกคือสัญญาณ HIGH เพื่อให้ LED สว่าง และสัญญาณ LOW เพื่อให้ LED ดับ โดยไฟที่จะสั่งให้ค่าในแต่ละบิท จำนวน 8 บิทไปแสดงผลเป็นค่าตัวเลขต่างๆนั้นได้จากการต่อกับขั้วของบอร์ด Arduino ตามวงจรด้านล่าง ซึ่งจะมีรีซิสเตอร์ขนาด 220 โอห์ม มาใช้ในการควบคุมกระแสที่ไหลผ่านในแต่ละ Segment



ตัวอย่างของ 7-Segment เบอร์ 5641 จะมีตำแหน่งในแต่ละ Segment และขาต่างๆ ที่จะนำมาใช้ในการเชื่อมต่อแสดงได้ดังรูป รายละเอียดต่างๆเพิ่มเติมให้เปิดดูได้จากเอกสาร 7SEGMENT5641_DataSheet



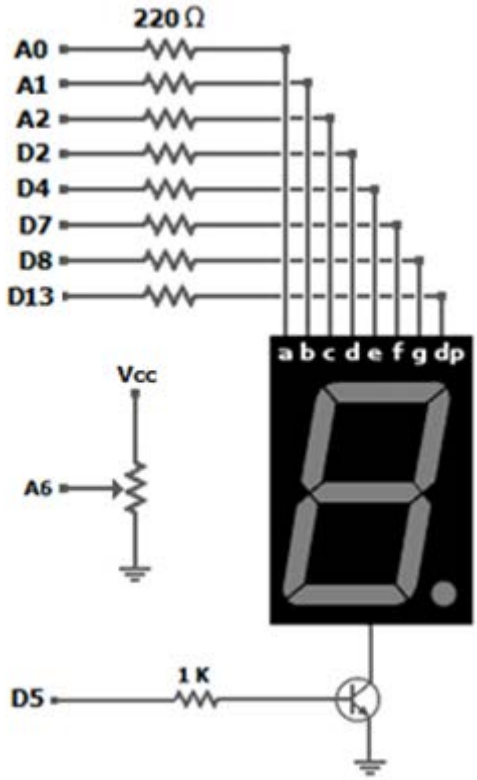
1. ให้เชื่อมต่อสายวงจร 7-Segment กับบอร์ด Arduino ทำการเขียนโปรแกรมที่ทำหน้าที่สั่งงานให้ LED บน 7-Segment แสดงผลเป็นตัวเลขตั้งแต่ 1 ถึง 3 จากนั้นให้คอมไพล์แล้ว Upload โปรแกรมที่ได้ลงบนบอร์ด

```
// 4 digit 7 segment display
int segmentA = A0;
int segmentB = A1;
int segmentC = A2;
int segmentD = 2;
int segmentE = 4;
int segmentF = 7;
int segmentG = 8;
int segmentDP = 13;

void setup()
{
  pinMode(segmentA, OUTPUT);
  pinMode(segmentB, OUTPUT);
  pinMode(segmentC, OUTPUT);
  pinMode(segmentD, OUTPUT);
  pinMode(segmentE, OUTPUT);
  pinMode(segmentF, OUTPUT);
  pinMode(segmentG, OUTPUT);
  pinMode(segmentDP, OUTPUT);
}

void loop()
{
    displayNumber();
}

void displayNumber()
{
  for(int digit = 1 ; digit <= 3 ; digit++)
  {
    displaySegment(digit);                          // แสดงผลบน 7-Segment ขนาด 1 หลัก
    delay(500);                                     // หน่วงเวลา 0.5 วินาที
  }
}

void displaySegment(int numberToDisplay)
{
  switch (numberToDisplay)
    {
    case 1:                                         // แสดงผลเลข 1
      digitalWrite(segmentA, LOW);
      digitalWrite(segmentB, HIGH);
      digitalWrite(segmentC, HIGH);
      digitalWrite(segmentD, LOW);
      digitalWrite(segmentE, LOW);
      digitalWrite(segmentF, LOW);
      digitalWrite(segmentG, LOW);
      break;

    case 2:                                         // แสดงผลเลข 2
      digitalWrite(segmentA, HIGH);
      digitalWrite(segmentB, HIGH);
      digitalWrite(segmentC, LOW);
      digitalWrite(segmentD, HIGH);
      digitalWrite(segmentE, HIGH);
      digitalWrite(segmentF, LOW);
      digitalWrite(segmentG, HIGH);
      break;

    case 3:                                         // แสดงผลเลข 3
      digitalWrite(segmentA, HIGH);
      digitalWrite(segmentB, HIGH);
      digitalWrite(segmentC, HIGH);
      digitalWrite(segmentD, HIGH);
      digitalWrite(segmentE, LOW);
      digitalWrite(segmentF, LOW);
      digitalWrite(segmentG, HIGH);
      break;
    }
}
```

2. ให้แก้ไขโปรแกรมเพื่อให้ 7 Segment แสดงผลเป็นการนับตัวเลขตั้งแต่ 0 ถึง 9 เรียงลำดับกันไป โดยใช้คำสั่ง case

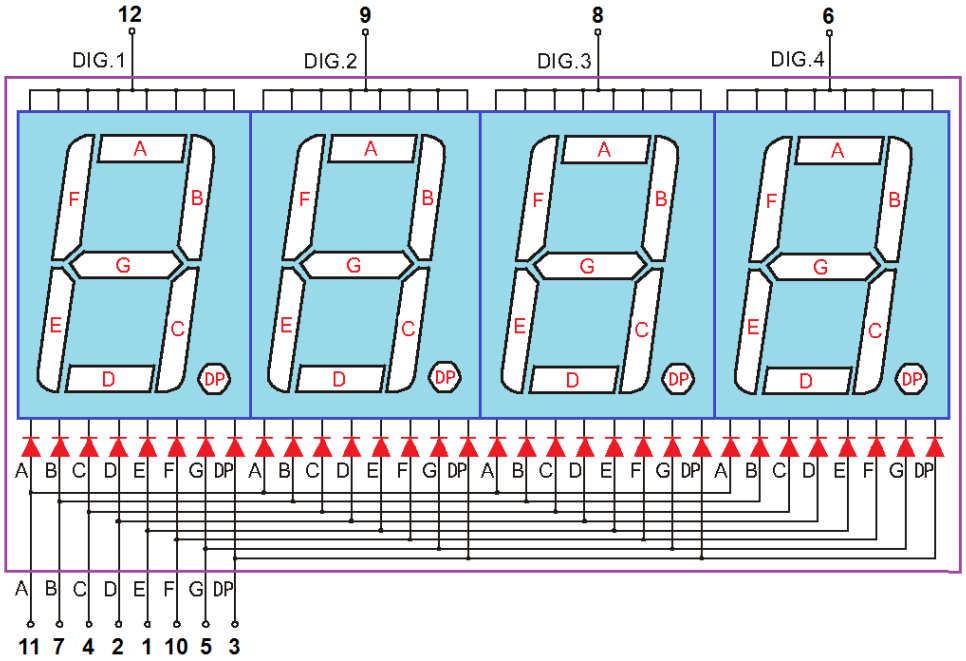
3. จากข้อ 2 ให้เพิ่ม Switch ที่ต่อระหว่างขา D12 และขากราวด์ เสร็จแล้วให้เขียนโปรแกรมกำหนดขา D12 เป็น Input ที่มีการต่อ Internal Resistor แบบ pull-up โดยใช้คำสั่ง pinMode(12,INPUT_PULLUP) และให้ทำการตรวจสอบสถานะของขา D12 โดยใช้คำสั่ง if (!digitalRead(12)) การทำงานกำหนดเงื่อนไขไว้ว่า เมื่อมีการกด Switch ให้ 7 Segment แสดงผลเป็นการนับตัวเลขถอยหลังตั้งแต่ 9 ถึง 0 เรียงลำดับกันไปครั้งละ 1 หลัก แต่ถ้าไม่ใช้คือไม่ได้กดสวิทช์ให้แสดงผลตามข้อ 2 เหมือนเดิม

4. ให้ต่อวงจรโดยเพิ่มตัว Transistor เข้าที่ขา Common Cathode ของ 7-Segment และมีตัวความต้านทานขนาด 1 K ต่อเข้าที่ขา B ของ Transistor ทำหน้าที่ป้อนไฟจากขา D5 เพื่อสั่งให้ Transistor เปิด-ปิดการทำงาน เสมือนทำหน้าที่เป็นสวิทช์เปิด-ปิดการแสดงผลของ 7-Segment ตามวงจรในรูป



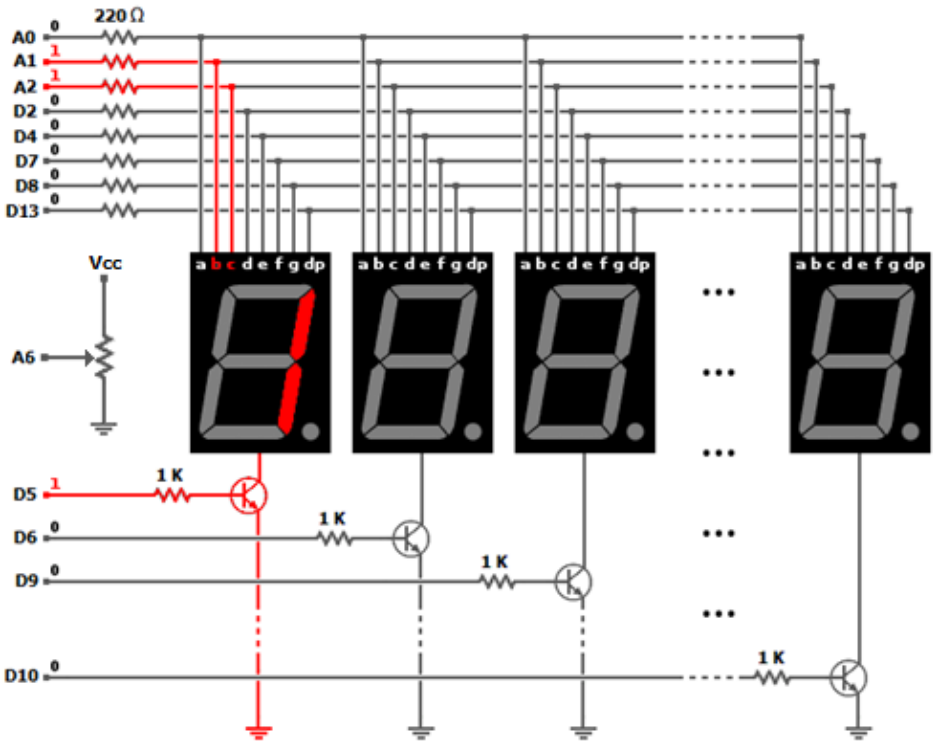
5. ให้ต่อวงจรโดยเพิ่มตัวความต้านทานปรับค่าได้ (VR) ทำหน้าที่แบ่งแรงดันไฟจาก Vcc ไปป้อนเข้าขาอนาล็อกอินพุท A6 ของบอร์ด Arduino Nano และให้แก้ไขโปรแกรมเพิ่มการอ่านค่าแรงดันไฟฟ้าที่ขา A6 ด้วยคำสั่ง analogRead ซึ่งค่าที่อ่านเข้ามาจะผ่านวงจร ADC ที่ทำหน้าที่แปลงแรงดันไฟฟ้าจากอนาล็อกไปเป็นสัญญาณดิจิตอลขนาด 10 บิท ในช่วงระหว่าง 0 ถึง 1023 โดยกำหนดว่าถ้าค่าเป็น 0 ให้ 7-Segment สว่างน้อยที่สุดและความสว่างจะเพิ่มขึ้นจนสูงสุดที่ค่า 1023 วิธีการปรับค่าความสว่างหรือหรี่หลอดไฟ ของ 7-Segment จะทำโดยการส่งสัญญาณที่เป็น PWM ไปที่ขา D5 โดยขา PWM นี้จะป็นขนาด 8 บิท มีค่าระหว่าง 0 ถึง 255 เมื่อขา A6 มีอินพุทเข้ามาตามการปรับค่า VR จะต้องให้ขา D5 ส่งเอาท์พุทออกไปด้วยคำสั่ง analogWrite ควบคุมปรับค่าดิวตี้ไซเคิลตามการควบคุมคาบเวลาของสัญญาณที่เป็นลอจิก 1 เทียบกับคาบเวลาที่เป็นลอจิก 0 ซึ่งค่าที่ได้จะเป็นเปอร์เซ็นตามค่าของอินพุทที่เข้ามาจากขา A6

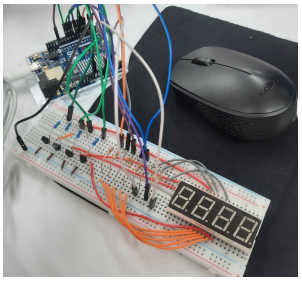
7-Segment ที่ใช้ในการทดลองจะมี 4 หลัก โดยปกติแล้วจะใช้ขา Segment ร่วมกัน เพื่อประหยัดขาที่จะต่อออกมาภายนอก ถ้าวงจรเป็นแบบ Common Cathode จะต้องมีขา Common Cathode ของแต่ละ Digit แยกออกจากกันเป็น 4 ขา ตามจำนวน Digit ที่ใช้ที่มีอยู่ 4 หลัก เมื่อต้องการแสดงตัวเลขบน Digit ไหน ให้ส่งสัญญาณ HIGH และ LOW ไปที่ Segment Pin และต่อจุดร่วมที่เป็น Common Cathode ของ Digit นั้นลงกราวด์ ตัวอย่างของ 7-Segment เบอร์ FYLQ-5641A วงจรรวมภายในที่เป็นแบบ Common Cathode จะมีลักษณะดังนี้



6. ให้ต่อวงจรในส่วนของขา Common Cathode ในแต่ละ Digit เพิ่มตั้งแต่ Digit1 ถึง Digit4 โดยใช้ Transistor และตัวต้านทานขนาด 1 K เพื่อจะทำหน้าที่เป็นการ Scan ค่าของหลักที่ต้องการแสดง โดยใช้วงจรตามในรูป

หลัก 4
7-segment 5





การเขียนโปรแกรมจะต้องให้ Scan การแสดงผลตัวเลขทีละ Digit เรียงไปจนครบ 4 Digit โดยจะต้องส่งข้อมูลของ 7-Segment มา Latch ไว้ใน 7-Segment ของแต่ละ Digit ก่อน แล้วจึงทำการ Scan เพื่อให้ Digit นั้นทำงาน การทำงานของโปรแกรมจะต้องเร็วจนดูเหมือนว่า LED สว่างทั้ง 4 Digit พร้อมๆกัน เป็นการ Multiplex LED แบบหลาย Digit การทำงานของวงจรในลักษณะนี้ จะมีกระแสไหลผ่านในแต่ละ Digit ของขา Common Cathode เป็นจำนวนมากเกินกว่าที่ขาของ Arduino จะรับได้ จึงต้องใช้วิธีสั่งเปิด-ปิดในแต่ละ Digit ผ่านทางทรานซิสเตอร์เพื่อทำหน้าที่เป็นสวิทช์เปิด-ปิดแทน สำหรับการเขียนโปรแกรม Multiplex LED แบบ 4 หลัก ต้องสั่งให้ควบคุมทั้ง Segment และ Digit ให้ทำงานตรงกัน โดยการวนลูปให้แสดงทีละ Digit สลับไปเรื่อยๆ ซึ่งจะต้องลูปให้เร็วจนสายตามนุษย์มองไม่เห็นการกระพริบหรือดูไม่ทัน การ Multiplex ในแต่ละ Digit จะใช้ช่วงเวลา 1 ใน 4 ของเวลาทั้งหมด ทำให้ความสว่างของ LED ในแต่ละ Digit จะลดลงเหลือเพียง 1/4 ของค่าความสว่างปกติ โดยทั่วไป LED จะยอมให้ Burst กระแสมากกว่าพิกัดในช่วงเวลาอันสั้น โดยดูได้จาก Parameter ค่า Peak Forward Current ของ Datasheet ดังนั้นหากต้องการให้ความสว่างมากขึ้นต้องคำนวณหาค่าความต้านทานที่ใช้ในการควบคุมกระแสให้เหมาะสมในแต่ละ Segment

7. ให้คำนวณหากระแสที่ใช้ใน LED แต่ละหลอดใน 7 Segment แบบ 4 Digit เมื่อกำหนดให้ตัวความต้านทานเท่ากับ 220 โอห์ม และให้คำนวณหาค่าขนาดของความต้านทานค่าต่ำสุดที่จะใช้เพื่อต้องการให้ได้ความสว่างสูงสุด โดยค่าต่างๆที่นำมาใช้คำนวณให้อ้างอิงกับค่าใน Datasheet

① I เมื่อ R = 220 Ω

② ความสว่างสูงสุด R_{min} = ?

① $V = IR$

$V_s - V_{LED} = IR$

$5 - 1.8 = I(220)$

$3.2 = 220I$

$I = 0.015$ A

② จาก DataSheet $V_{LED} = 1.8$ V, $I = 30$ mA ; $V = IR$

$V_s - V_{LED} = IR_{min}$

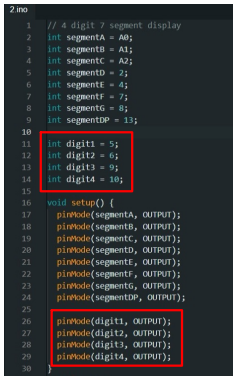
$5 - 1.8 = 0.03 R_{min}$

$3.2 = 0.03 R_{min}$

$R_{min} = 106.67\ \Omega$

8. ให้เพิ่มโปรแกรมเพื่อให้ 7 Segment แสดงผลเป็นแบบ 4 Digit โดยกำหนดตัวแปรเพิ่มจาก Hardware ที่ต่อไว้ดังนี้

```
int digit1 = 5;
int digit2 = 6;
int digit3 = 9;
int digit4 = 10;
```



การกำหนดให้ตำแหน่งขาที่ต่อเป็นเอาท์พุทลงใน void setup() เพิ่มเป็นดังนี้

```
pinMode(digit1, OUTPUT);
pinMode(digit2, OUTPUT);
pinMode(digit3, OUTPUT);
pinMode(digit4, OUTPUT);
```

9. ให้แก้ไขโปรแกรมเพื่อให้ 7 Segment แสดงผลเป็นเลข 1234 ในแต่ละ Digit ตามลำดับ โดยในส่วนของ displayNumber() ให้เพิ่มโปรแกรมการเปิดการทำงานของ Digit ต่างๆดังนี้

```
//Turn on a digit
```

```
switch(digit)
 {
        case 1:
          digitalWrite(digit1, HIGH);
          break;
        case 2:
          digitalWrite(digit2, HIGH);
          break;
        case 3:
          digitalWrite(digit3, HIGH);
          break;
        case 4:
          digitalWrite(digit4, HIGH);
          break;
 }
```

และเมื่อแสดงผลเรียบร้อยแล้วให้สั่งปิดการทำงานของ Digit ทั้งหมดดังนี้

```
//Turn off all digits
digitalWrite(digit1, LOW);
digitalWrite(digit2, LOW);
digitalWrite(digit3, LOW);
digitalWrite(digit4, LOW);
```

10. ถ้าจะแก้ไขโปรแกรมเพื่อให้ 7 Segment แบบ 4 Digit แสดงผลเป็นเลข 4321 พร้อมกันทั้ง 4 Digit โดยไม่มีการกระพริบต้องแก้ไขที่ส่วนไหนบ้าง

1) แก้ไข case ใน switch(digit) จาก 1,2,3,4 เป็น 4,3,2,1
2) เปลี่ยน delay จาก 500 เป็น 5

11. ให้เขียนโปรแกรมเพื่อให้ 7 Segment แบบ 4 Digit แสดงผลเป็นรหัสนักศึกษาทั้ง 8 หลัก โดยเลื่อนข้อมูลจากขวาไปซ้าย

12. ให้ต่อวงจรเพื่อให้ทำหน้าที่เป็นโวลต์มิเตอร์ โดยใช้ตัวความต้านทานที่ปรับค่าได้ High Precision Trimmer Potentiometer Variable Resistor เบอร์ 3296W-102 ขนาด 1 K ohm ซึ่งสามารถปรับค่าความต้านทานได้โดยการหมุนปรับค่า จากขากลางของ VR ต่อเข้ากับขา A0 ของ Arduino และขาด้านข้างอีกสองขาให้ต่อกับกราวด์ และขั้วของอุปกรณ์ที่ต้องการวัดค่าแรงดันไฟฟ้า เขียนโปรแกรมใช้คำสั่ง analogRead อ่านค่าจากขา A0 ซึ่งเป็นสัญญาณดิจิตอลขนาด 10 บิท แล้วแปลงค่าที่ได้เป็นระดับแรงดันไฟฟ้าออกมาแสดงค่าบนจอภาพ โดยใช้ serial monitor ซึ่งเป็นการสื่อสารแบบอนุกรมส่งค่าจากไมโครคอนโทรลเลอร์มายังคอมพิวเตอร์ ซึ่งมีประโยชน์ใช้ในการตรวจสอบการทำงานของไมโครคอนโทรลเลอร์ได้

```
void setup()
{
  Serial.begin(115200);                    // initialize serial communication at 115200 bits per second
}

void loop()
{
  int sensorValue = analogRead(A0);              // read the input on analog pin 0
  float voltage = sensorValue * (5.0 / 1023.0);  // Convert the analog reading (0 - 1023  to  0 - 5V)

  Serial.println(voltage);                       // print out the value
}
```