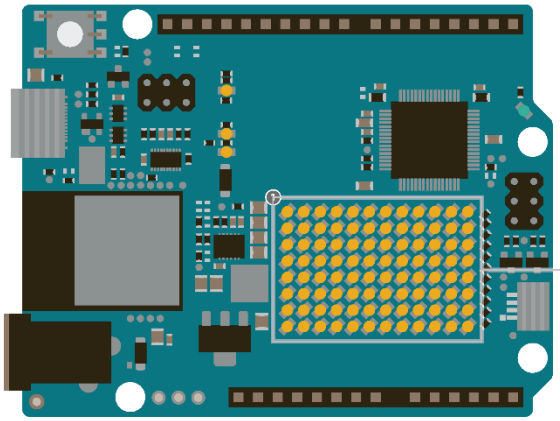
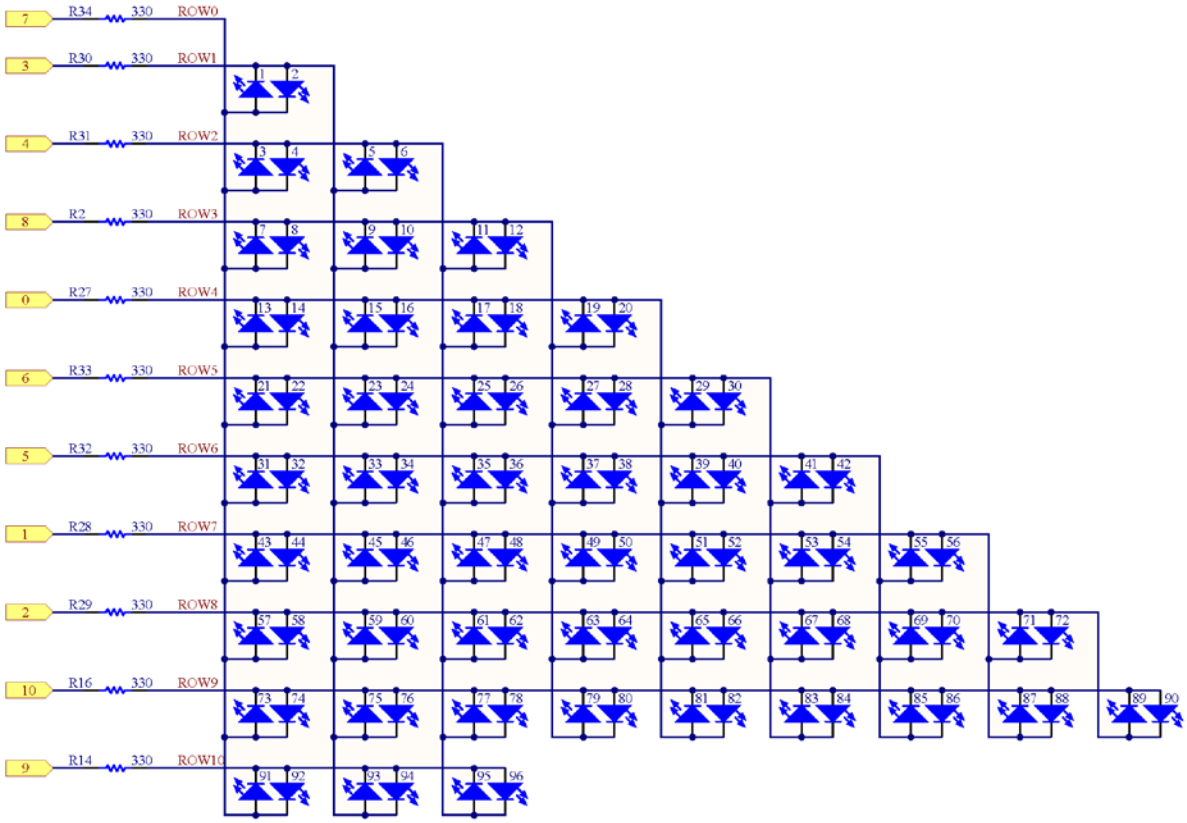
LED Matrix บอร์ด UNO R4 WiFi จะมีเมทริกซ์ LED สีแดงขนาด 12x8 มีการเชื่อมต่อ GPIO แบบมัลติเพล็กซ์เพื่อควบคุม LEDs ใช้เทคนิคที่เรียกว่า Tri-state multiplexing (Charlieplexing) โดยการใช้คุณสมบัติของ pin ในไมโครคอนโทรลเลอร์ที่มีอยู่สามสถานะ (Tri-State) คือ Low , High และ Hi Impedance ถ้ามีจำนวน pin เท่ากับ n จะสามารถใช้ควบคุม LED จำนวนสูงสุดที่จะขับให้ทำงานได้เท่ากับ $n^2 - n$ ตัวอย่างการต่อวงจร LED Matrix 12x8 ตามในรูป



LED Matrix 12x8

1	2	3	4	5	6	7	8	9	10	11	12
13	14	15	16	17	18	19	20	21	22	23	24
25	26	27	28	29	30	31	32	33	34	35	36
37	38	39	40	41	42	43	44	45	46	47	48
49	50	51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70	71	72
73	74	75	76	77	78	79	80	81	82	83	84
85	86	87	88	89	90	91	92	93	94	95	96



1. ให้เชื่อมต่อสาย USB ของบอร์ดกับคอมพิวเตอร์ เปิดโปรแกรม Arduino จากนั้นทำการเขียนโปรแกรมที่ทำหน้าที่สั่งงานให้ LED Matrix 12x8 ที่อยู่บนบอร์ดไมโครคอนโทรลเลอร์แสดงรูปภาพตามค่าใน 2D array มีขนาดเป็นไบต์ที่มีจำนวนทั้งหมด 96 ไบต์ต่อ 1 fame ของภาพ โดยใช้ไลบรารีของ Arduino LED Matrix แล้วทำการ Upload โปรแกรมที่ได้ลงบนบอร์ด Arduino และให้ทดลองการทำงาน

```
#include "Arduino_LED_Matrix.h"          // Include the LED_Matrix Library
ArduinoLEDMatrix matrix;                 // Create an instance of the ArduinoLEDMatrix class

uint8_t frame[8][12] = {
  { 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0 },
  { 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0 },
  { 1, 1, 0, 1, 1, 0, 0, 1, 1, 0, 1, 1 },
  { 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1 },
  { 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1 },
  { 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1 },
  { 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0 },
  { 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0 }
};                                       // Pre-defined 2D array

void setup() {
  matrix.begin();                        // Initialize LED matrix
}

void smile(){
  frame[4][3] = 1;
  frame[4][8] = 1;
}

void face(){
  frame[4][3] = 0;
  frame[4][8] = 0;
}

void loop(){
face();
matrix.renderBitmap(frame, 8, 12);       // Display pattern on the LED matrix
delay(1000);

smile();
matrix.renderBitmap(frame, 8, 12);
delay(1000);
}
```

2. ให้แก้ไขโปรแกรมในข้อที่ 1 เพื่อให้ LED Matrix 12x8 แสดงผลเป็นรูปภาพอื่นตามแบบร่างที่กำหนดขึ้นเอง โดยใช้คำสั่งจากค่าใน 2D array ที่เขียนขึ้นใหม่ด้านล่าง

```
uint8_t frame[8][12] = {
  { .... , .... , .... , .... , .... , .... , .... , .... , .... , .... , .... , .... },
  { .... , .... , .... , .... , .... , .... , .... , .... , .... , .... , .... , .... },
  { .... , .... , .... , .... , .... , .... , .... , .... , .... , .... , .... , .... },
  { .... , .... , .... , .... , .... , .... , .... , .... , .... , .... , .... , .... },
  { .... , .... , .... , .... , .... , .... , .... , .... , .... , .... , .... , .... },
  { .... , .... , .... , .... , .... , .... , .... , .... , .... , .... , .... , .... },
  { .... , .... , .... , .... , .... , .... , .... , .... , .... , .... , .... , .... },
  { .... , .... , .... , .... , .... , .... , .... , .... , .... , .... , .... , .... }};
```

หลักการของระบบตัวเลขจะถูกนำมาใช้ในการทำงานของสัญญาณดิจิตอล โดยแทนสภาวะการทำงานของสัญญาณ HIGH และ LOW ด้วยค่าคงที่เป็นเลขฐานสอง (Binary Number) ได้คือ 1 และ 0 โดยที่แต่ละสายข้อมูล 1 เส้น จะประกอบจากหน่วยข้อมูลที่เรียกว่า บิต (Bit) กลุ่มของตัวเลขฐานสองขนาด 8 บิตจะเรียกว่าไบต์ (Byte) ใน 1 ไบต์จะมีจำนวนข้อมูลทั้งหมดเท่ากับ $2^8 = 256$ ค่า โดยไบต์ที่มีค่าเป็น 0 จะแสดงเป็น 00000000 และไบต์ที่ไม่ใช่ศูนย์จะเป็นการผสมกันของเลข 1 และ 0 เช่น 01001011 บิตที่อยู่ทางซ้ายสุดของชุดข้อมูลไบนารีเรียกว่า บิตที่มีนัยสำคัญมากที่สุด (Most Significant Bit) เรียกย่อว่า MSB และบิตขวาสุดจะเรียกว่า บิตที่มีนัยสำคัญน้อยที่สุด (Least Significant Bit) ย่อว่า LSB เพื่อให้เห็นค่าของข้อมูลแต่ละบิตในรูปเลขฐานสิบ จะเขียนให้อยู่ในรูปของเลขชี้กำลัง (Exponent) โดยให้ตำแหน่งบิตเป็นค่าของตัวเลขที่แสดงค่ายกกำลัง และมีฐานเป็นเลขสอง เริ่มจากบิตที่มีค่าน้อยที่สุดทางขวา (LSB) ฐานที่มีค่าเป็น 2 ยกกำลัง 0 จะมีค่าผลลัพธ์เท่ากับ 1 บิตต่อมาเรียงตามลำดับตำแหน่งบิตถัดไปทางด้านซ้ายที่ละ 1 บิต เลขชี้กำลังที่ได้แต่ละค่าจะเพิ่มขึ้นที่ละ 1 ดังนั้นหลักต่อมาจะได้ฐาน 2 ยกกำลัง 1 มีค่าเท่ากับ 2 เมื่อข้อมูลไบนารีมีขนาดเท่ากับ 1 ไบต์ บิตที่มีนัยสำคัญมากที่สุด (MSB) จะมีค่าเลขชี้กำลังเป็น 7 ถ้าให้ตำแหน่งบิตของข้อมูลเขียนย่อเป็นตัวอักษร D โดยเริ่มจาก D0, D1, D2 , D3, ฯลฯ ดังนั้นผลลัพธ์ของข้อมูลในแต่ละบิตในรูปเลขฐานสิบ (Decimal) จะได้ดังนี้

Bit	D7	D6	D5	D4	D3	D2	D1	D0
Base^{exponent}	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Decimal	128	64	32	16	8	4	2	1

การแสดงตัวเลขไบนารีเมื่อมีค่าของข้อมูลจำนวนมาก จะทำให้มีจำนวนหลักของข้อมูลที่มาก ซึ่งยุ่งยากต่อการเขียน เพื่อให้ง่ายเข้าโดยทั่วไปในการแสดงผลของข้อมูลจึงนิยมเขียนให้อยู่ในรูปของเลขฐานสิบหก (Hexadecimal) หรือย่อว่า HEX ดังนั้นเราจะได้จำนวนข้อมูลทั้งหมดเท่ากับ 16 ค่า คือ จาก 0 ถึง 15 การใช้เลขฐานสิบหกสามารถทำให้เขียนได้ง่ายขึ้น โดยแทน 0 ถึง 9 เหมือนเลขฐานสิบ และเฉพาะค่าเลข 10 ถึง 15 ซึ่งเป็นเลข 2 หลัก จะเปลี่ยนให้เป็น 1 หลัก โดยแทนด้วยตัวอักษร A ถึง F ตารางต่อไปนี้แสดงค่าของเลขฐานสิบหก (Hexadecimal) เทียบกับเลขฐานสอง (Binary) และเลขฐานสิบ (Decimal)

Binary	Decimal	Hexadecimal
0000	0	0
0001	1	1
0010	2	2
0011	3	3
0100	4	4
0101	5	5
0110	6	6
0111	7	7
1000	8	8
1001	9	9
1010	10	A
1011	11	B
1100	12	C
1101	13	D
1110	14	E
1111	15	F

3. จากข้อที่ 1 เป็นการสร้างอาร์เรย์สองมิติเก็บค่าเป็นไบต์ ซึ่งวิธีนี้จะใช้หน่วยความจำมากกว่าที่จำเป็น เพราะว่า LED แต่ละตัวต้องการเพียงบิตเดียวในการจัดเก็บสถานะ ดังนั้นจึงให้ทำการเปลี่ยนขนาดของจุดภาพจาก 1 ไบต์ ไปเป็น 1 บิตต่อจุดภาพ ในการจัดเก็บภาพจึงเปลี่ยนมาใช้อาร์เรย์ของจำนวนเต็ม 32 บิต ทำให้วิธีนี้มีประสิทธิภาพในการใช้หน่วยความจำมากขึ้น จากตัวอย่างอาร์เรย์ข้อ 1 ด้านล่างให้แก้ไขรวมจำนวน 4 ค่าจากเลขฐาน 2 เปลี่ยนให้เป็นเลขฐาน 16 จำนวน 1 ค่า

{ 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0 } แปลงเป็นเลขฐาน 16 เท่ากับ ….. , ….. , …..
{ 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0 } แปลงเป็นเลขฐาน 16 เท่ากับ ….. , ….. , …..
{ 1, 1, 0, 1, 1, 0, 0, 1, 1, 0, 1, 1 } แปลงเป็นเลขฐาน 16 เท่ากับ ….. , ….. ; …..
{ 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1 } แปลงเป็นเลขฐาน 16 เท่ากับ ….. , ….. , …..
{ 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1 } แปลงเป็นเลขฐาน 16 เท่ากับ ….. , ….. , …..
{ 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1 } แปลงเป็นเลขฐาน 16 เท่ากับ ….. ; ….. , …..
{ 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0 } แปลงเป็นเลขฐาน 16 เท่ากับ ….. , ….. , …..
{ 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0 } แปลงเป็นเลขฐาน 16 เท่ากับ ….. , ….. , …..

เมื่อเขียนเป็นเลขฐาน 16 ขนาด 32 บิตจะได้ ……..……...….. , ……..……..…….. , ……...……...……..

4. จากค่าในข้อที่ 3 เมื่อนำไปสร้างอาร์เรย์ใหม่ที่เป็นเลขจำนวนเต็ม 32 บิตแทนในข้อที่ 1 โดยที่ LED Matrix จะมีจุดภาพ 12 x 8 = 96 ดังนั้นอาร์เรย์จะมีค่าที่เก็บทั้งหมด 96/32 = 3 ค่า โดยโปรแกรมที่แก้ไขแล้วเพื่อแสดงผลใน LED Matrix 12x8 เป็นดังนี้

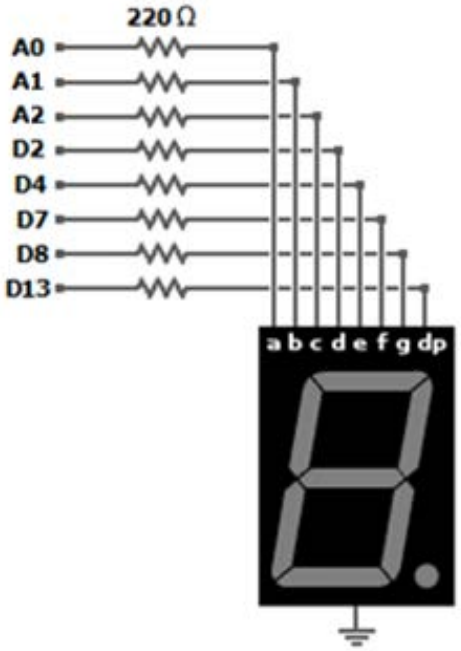
```
#include "Arduino_LED_Matrix.h"
ArduinoLEDMatrix matrix;

void setup() {
  matrix.begin();
}

const uint32_t smile[] = {
    0x3fc606d9,
    0xb801909c,
    0xf36063fc
};
const uint32_t happy[] = {
    0x19819,
    0x80000001,
    0x81f8000
};

void loop(){
  matrix.loadFrame(smile);
  delay(500);

  matrix.loadFrame(happy);
  delay(500);
}
```

5. ให้แก้ไขโปรแกรมในข้อที่ 4 เพื่อให้ LED Matrix 12x8 แสดงผลเป็นรูปภาพอื่น

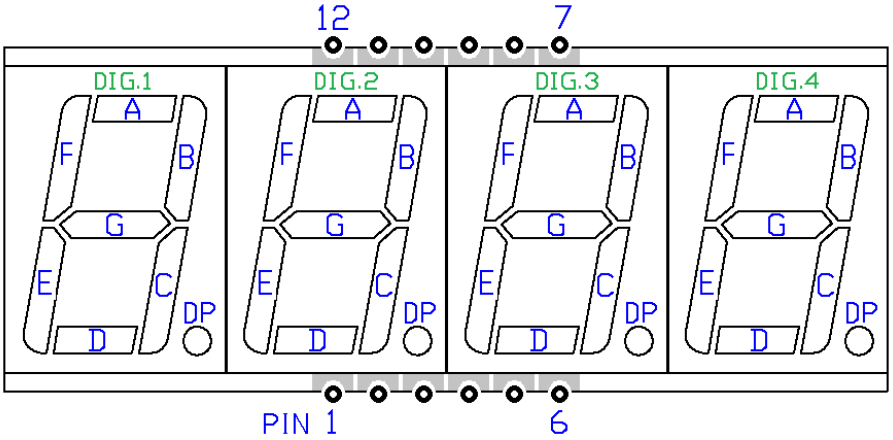
6. ให้เขียนโปรแกรมเพื่อให้ LED Matrix 12x8 แสดงผลกราฟิกชื่อนักศึกษาเป็นภาษาไทยเคลื่อนไหวจากด้านขวาไปซ้าย โดยมีรูปแบบการเขียนโปรแกรมตามตัวอย่างด้านล่าง

```
#include "Arduino_LED_Matrix.h"          // Include the LED_Matrix library
ArduinoLEDMatrix matrix;                 // Create an instance of the ArduinoLEDMatrix class

const uint32_t frames[][4] = {
        { 0x0, 0x10010010, 0x1000000, 100 },
        { 0x100, 0x20030020, 0x2000000, 100 },
        { 0x300, 0x40070040, 0x4000000, 100 },
        { 0x600, 0x900f0090, 0x9000000, 100 },
        { 0xc01, 0x201e0120, 0x12000000, 100 },
        { 0x1902, 0x503d0250, 0x25000000, 100 },
        { 0x3304, 0xa07b04a0, 0x4a000000, 100 },
        { 0x6709, 0x40f70940, 0x94000000, 100 },
        { 0xce12, 0x91ee1291, 0x29000000, 100 },
        { 0x19c25, 0x23bc2522, 0x52000000, 100 },
        { 0x3394a, 0x57b94a54, 0xa5000000, 100 },
        { 0x67394, 0xaf7294a9, 0x4b000000, 100 },
        { 0xce729, 0x4ee42942, 0x97000000, 100 },
        { 0x9ce52, 0x9dc95295, 0x2e000000, 100 },
        { 0x39ca5, 0x2b92a52a, 0x5c000000, 100 },
        { 0x7394a, 0x57254a54, 0xb8000000, 100 },
        { 0xe7294, 0xae4a94a9, 0x71000000, 100 },
        { 0xce429, 0x4c942942, 0xe3000000, 100 },
        { 0x9c952, 0x99295295, 0xc6000000, 100 },
        { 0x392a5, 0x2252a52b, 0x8c000000, 100 },
        { 0x7254a, 0x44a44a47, 0x19000000, 100 },
        { 0xe4b94, 0x9949949e, 0x33000000, 100 },
        { 0xc9729, 0x2292292c, 0x67000000, 100 },
        { 0x92e52, 0x45245248, 0xce000000, 100 },
        { 0x25da4, 0x9a49a491, 0x9d000000, 100 },
        { 0x4ba49, 0x34924923, 0x3a000000, 100 },
        { 0x97492, 0x69259246, 0x74000000, 100 },
        { 0x2e924, 0xd24b249c, 0xe9000000, 100 },
        { 0x5d249, 0xa4964929, 0xd2000000, 100 },
        { 0xba493, 0x592d9253, 0xa4000000, 100 },
        { 0x74926, 0xa25a24a7, 0x49000000, 100 },
        { 0xe934d, 0x44b4494e, 0x93000000, 100 },
        { 0xd269a, 0x9969929d, 0x26000000, 100 },
        { 0xa4c35, 0x22d2252a, 0x4c000000, 100 },
        { 0x4986a, 0x45a44a44, 0x98000000, 100 },
        { 0x930d4, 0x8b489489, 0x30000000, 100 },
        { 0x260a9, 0x6902902, 0x60000000, 100 },
        { 0x4c052, 0xd205204, 0xc0000000, 100 },
        { 0x980a4, 0xa40a409, 0x80000000, 100 },
        { 0x30048, 0x4804803, 0x0, 100 },
        { 0x60090, 0x9009006, 0x0, 100 },
        { 0xc0020, 0x200200c, 0x0, 100 },
        { 0x80040, 0x4004008, 0x0, 100 },
        { 0x80, 0x8008000, 0x0, 100 },
        { 0x0, 0x0, 0x0, 100 }
};

void setup() {
        matrix.loadSequence(frames);
        matrix.begin();
        matrix.play(true);
}

void loop() {
}
```

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



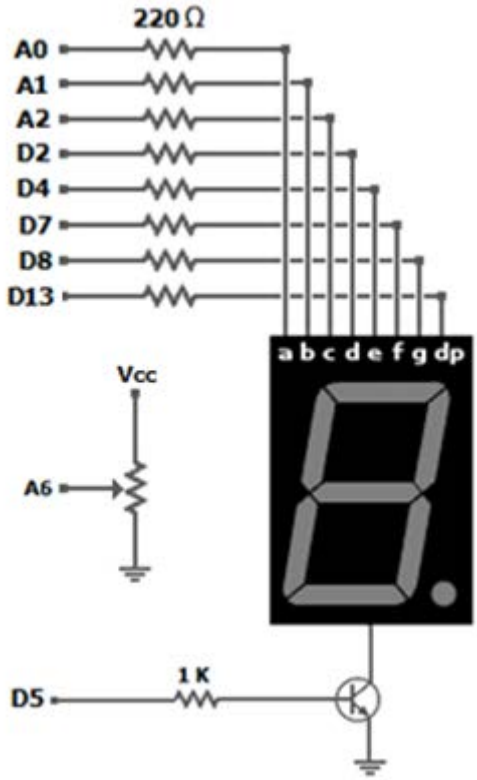
1. ให้เชื่อมต่อสายวงจร 7-Segment กับบอร์ด Arduino ทำการเขียนโปรแกรมที่ทำหน้าที่สั่งงานให้ LED บน 7-Segment แสดงผลเป็นตัวเลขตั้งแต่ 1 ถึง 3 จากนั้นให้คอมไพล์แล้ว Upload โปรแกรมที่ได้ลงบนบอร์ด

```
// 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

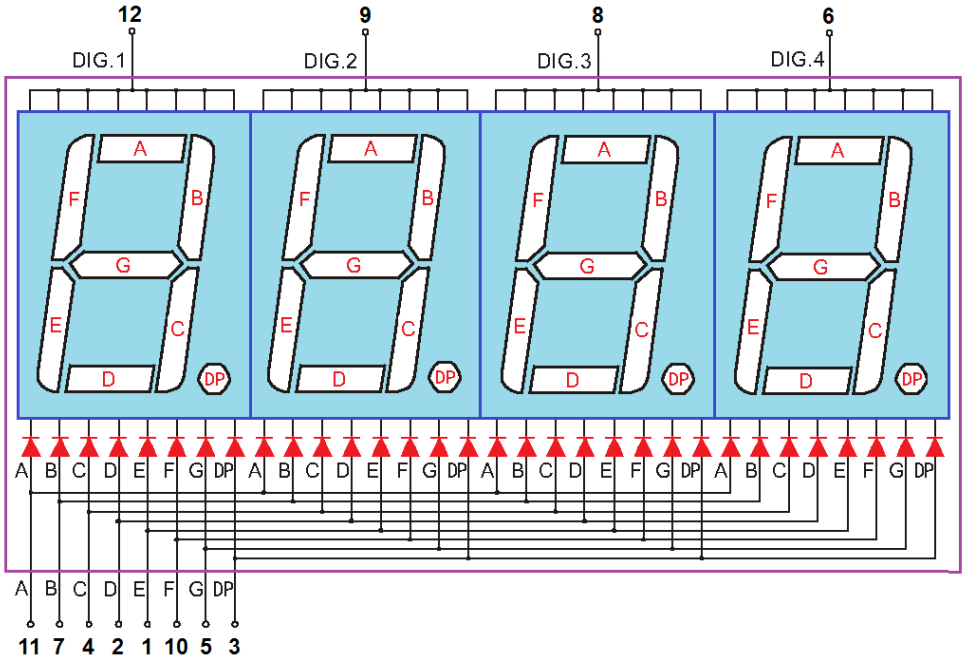
3. จากข้อ 2 ให้เพิ่ม Switch ที่ต่อระหว่างขา D12 และขากราวด์ เสร็จแล้วให้เขียนโปรแกรมกำหนดขา D12 เป็น Input ที่มีการต่อ Internal Resistor แบบ pull-up โดยใช้คำสั่ง pinMode(12,INPUT_PULLUP) และให้ทำการตรวจสอบสถานะของขา D12 โดยใช้คำสั่ง if (!digitalRead(12)) การทำงานกำหนดเงื่อนไขไว้ว่า เมื่อมีการกด Switch 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

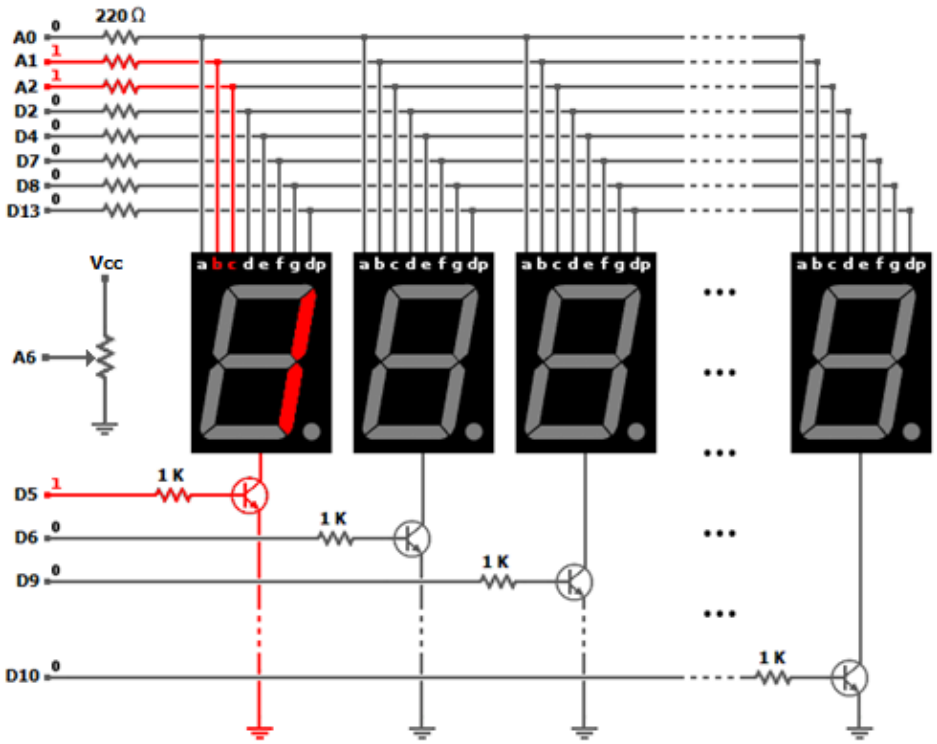


5. ให้ต่อวงจรโดยเพิ่มตัวความต้านทานปรับค่าได้ (VR) ทำหน้าที่แบ่งแรงดันไฟจาก Vcc ไปป้อนเข้าขาอนาล็อกอินพุท A6 ของบอร์ด Arduino Nano และให้แก้ไขโปรแกรมเพิ่มการอ่านค่าแรงดันไฟฟ้าที่ขา A6 ด้วยคำสั่ง analogRead ซึ่งค่าที่อ่านเข้ามาจะผ่านวงจร ADC ที่ทำหน้าที่แปลงแรงดันไฟฟ้าจากอนาล็อกไปเป็นสัญญาณดิจิตอลขนาด 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



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

   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....
   .....

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

    ……

    ……

    ……

    ……..

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
}
```

13. ถ้าต้องการทำเป็นโวลต์มิเตอร์ที่วัดค่าได้ตั้งแต่ 0 ถึง 10.00 V โดยมีความถูกต้องในแต่ละ step เท่ากับ 0.01 V จะต้องแก้ไขในส่วนใดบ้าง ให้แสดงวิธีคำนวณหาค่าพร้อมทั้งยกตัวอย่าง

    ……

    ……

    ……

    ……

    ……

    ……

    ……

14. จากข้อ 12 ให้แก้ไขโปรแกรมสร้างเป็นโวลต์มิเตอร์ที่วัดค่าได้ตั้งแต่ 0 ถึง 10.00 V โดยเขียนโปรแกรมอ่านค่าที่ได้จากการวัด ซึ่งเป็นสัญญาณดิจิตอลขนาด 10 บิท แล้วแปลงค่าที่ได้ไปเป็นระดับแรงดันไฟฟ้าออกมาแสดงผลบน 7 Segment แบบ 4 Digit ให้มีความถูกต้องในแต่ละ step เท่ากับ 0.01 V

15. จากการทดลองเรื่องการวัดอุณหภูมิ โดยใช้ตัวเทอร์มิสเตอร์มาใช้เป็นเซ็นเซอร์ ที่แสดงผลออกมาทาง Serial Monitor ให้แก้ไขโปรแกรมเพิ่มการแสดงผลให้ไปแสดงผลที่ 7-segment ขนาด 4 หลัก โดยให้แสดงผลการวัดอุณหภูมิเป็นทศนิยม 2 ตำแหน่ง

16. จากข้อ14 ให้แก้ไขวงจรและโปรแกรม เพื่อให้ใช้เป็นเครื่องวัดค่าตัวความต้านทาน หรือโอห์มมิเตอร์ ที่มีช่วงการวัดได้ 200 KΩ โดยให้แสดงผลเป็น 7-segment ขนาด 4 หลัก

17. จากข้อ14 ให้แก้ไขวงจรและโปรแกรม เพื่อให้เป็นเครื่องวัดค่า Diode หรือ LED โดยให้แสดงค่า Forward Voltage ของอุปกรณ์ที่วัดได้เป็นทศนิยม 1 ตำแหน่ง

18. จากข้อ17 ให้นำเอาเครื่องค่า LED ไปใช้ในการวัด LED แบบต่างๆและบันทึกผลที่ได้ลงในตาราง

LED	**Wavelength (nm)**	**ค่าที่วัดได้ Forward Voltage (V)**
Infrared		
สีแดง		
สีเขียว		
สีน้ำเงิน		
Ultra Violet		

ตารางสร้างตัวอักษร