ใบความรู้ที่ 3

การควบคุมการทำงานของเอาท์พุตดิจิตอล

ไมโครคอนโทรลเลอร์ Arduino หรือ อาดูอิโน่ เป็นบอร์ดไมโครคอนโทรลเลอร์ตระกูล AVR ดังที่กล่าวมาแล้วโดยมีขาในการทำงานทั้งหมด 14 ขาคือ Pin 0-13 โดยแต่ละขามีการทำงานที่แตกต่างกันออกไปเช่น ขาสัญญาณที่ติดต่อพอร์ตอนุกรม (Serial Port) คือ Pin 0, 1 เป็นขา Rx, Tx ส่วนขาสัญญาณที่ใช้ติดต่อสัญญาณอินพุตและเอาท์พุต คือ Pin 2 – Pin 13 และบางขาสัญญาณยังทำหน้าที่นอกเหนือจากขาสัญญาณอินพุตและเอาท์พุต เช่น Pin 3, 5, 6, 9, 10,11 ซึ่งสามารถทำหน้าที่เป็นขาสัญญาณพัลส์วิดท์มอดูเลชั่น (Pulse Width Modulation) หรือที่เรียกทั่วไปคือ PWM และยังมีขาที่รับสัญญาณอะนาลอกอีก 6 ขา คือ Pin A0, A1, A2, A3, A4, A5, A6 อีกด้วย

3.1 วงจรดิจิตอลในระบบสมองกลฝังตัว

วงจรดิจิตอล คือวงจรที่ทำงานอยู่บนพื้นฐานของสถานะเปิด และสถานะปิด ให้คำนึงถึงการเปิด-ปิดหลอดไฟ หากเปิดสวิตซ์ให้หลอดไฟติด จะหมายถึงมีการจ่ายกระแสไฟฟ้าให้หลอดไฟ เรียกสถานะนี้ว่าสถานะลอจิก 1 หรือสถานะ HIGH และเมื่อปิดสวิตซ์ให้หลอดไฟดับ จะหมายถึงไม่มีกระแสไฟฟ้าไหลไปที่หลอดไฟ เรียกสถานะนี้ว่าลอจิก 0 หรือ สถานะ LOW

ในไมโครคอนโทรลเลอร์จะใช้แรงดันที่เทียบขา GPIO กับกราวนด์(GND) ในการตัดสินสถานะ โดยหากมีแรงดันที่ขา GPIO มากกว่า 2.8 โวลต์ แต่ไม่เกินแรงดัน VCC หรือ 5 โวลต์จะตัดสินให้เป็นสถานะลอจิก 1 หรือสถานะ HIGH แต่หากแรงดันน้อยกว่า 2.8 โวลต์ แต่ไม่น้อยกว่า 0 โวลต์ จะถือว่าเป็นสถานะลอจิก 0 หรือ LOW

3.2 ฟังก์ชั่นการควบคุมเอาท์พุตแบบดิจิตอล

3.2.1 ฟังก์ชั่นกำหนดโหมดการทำงานให้กับขาพอร์ต สามารถกำหนดได้ทั้งขาดิจิตอลโดยใส่เพียงตัวเลขของขา (0, 1, 2,...13) และขาแอนาลอกที่ต้องการให้ทำงานในโหมดดิจิตอลได้จะต้องใส่ A นำหน้า ซึ่งใช้ได้เฉพาะ A0, A1,...A5 ส่วนขา A6 และ A7 (ที่มีในบอร์ด Arduino รุ่น Mini และ Nano) ไม่สามารถใช้งานในโหมดดิจิตอลได้ รูปแบบของฟังก์ชั่นเป็นดังนี้

```
pinMode(pin, mode);
```

pin : หมายเลขขาที่ต้องการเซตโหมด

mode : INPUT, OUTPUT, INPUT_PULLUP

ตัวอย่างเช่น

pinMode(13, OUTPUT); หมายถึง กำหนดให้ขา D13 ทำงานเป็นเอาต์พุตพอร์ต

pinMode(12, INPUT); หมายถึง กำหนดให้ขา D12 ทำงานเป็นอินพุตพอร์ต

pinMode(11, INPUT_PULLUP); หมายถึง กำหนดให้ขา D11 ทำงานเป็นอินพุตพอร์ตที่ใช้ตัวต้านทานพูลอัพภายในชิพ

3.2.2 ฟังก์ชั่นส่งค่าลอจิกดิจิตอลไปยังขาพอร์ต โดยค่า HIGH เป็นการส่งลอจิก 1 และค่า LOW เป็นการส่งลอจิก 0 ออกไปยังขาพอร์ต ฟังก์ชั่นนี้จะทำงานได้ต้องมีการใช้ฟังก์ชั่น pinMode ก่อน

```
digitalWrite(pin, value);
```

pin: หมายเลขขาที่ต้องการเขียนลอจิกออกพอร์ต

value: HIGH or LOW

ตัวอย่างเช่น

digitalWrite(13, HIGH); หมายถึง กำหนดให้ส่งลอจิก 1 ไปที่ขา D13

digitalWrite(13, LOW); หมายถึง กำหนดให้ส่งลอจิก 0 ไปที่ขา D13

3.3 ฟังก์ชั่นหน่วงเวลาหรือฟังก์ชั่นหยุดค้าง

3.3.1 ฟังก์ชั่นหน่วงเวลา หน่วยเป็นมิลลิวินาที

การใช้งานสามารถกำหนดตัวเลขของเวลาที่ต้องการหยุดค้างโดยตัวเลขที่ใส่เป็นตัวเลขของเวลาหน่วยเป็นมิลลิวินาที ตัวเลขของเวลาที่ใส่ได้สูงสุดคือ 4,294,967,295 ซึ่งเป็นขนาดของตัวแปร unsigned long

```
delay(ms);
```

ms: จำนวนตัวเลขที่หยุดค้างของเวลาหน่วยมิลลิวินาที (unsigned long)

ตัวอย่างเช่น

delay(500); หมายถึง หยุดค้าง (หน่วงเวลา) ไว้เป็นเวลา 500 มิลลิวินาที (1/2 วินาที)

delay(1000); หมายถึง หยุดค้าง (หน่วงเวลา) ไว้เป็นเวลา 1000 มิลลิวินาที (1 วินาที)

3.3.2 ฟังก์ชั่นหน่วงเวลา หน่วยเป็นไมโครวินาที

การใช้งานสามารถกำหนดตัวเลขของเวลาที่ต้องการหยุดค้างโดยตัวเลขที่ใส่เป็นตัวเลขของเวลาหน่วยเป็นไมโครวินาที ตัวเลขของเวลาที่ใส่ได้สูงสุดคือ 65,535 ซึ่งเป็นขนาดของตัวแปร unsigned int

```
delayMicroseconds(us);
```

us: จำนวนตัวเลขที่หยุดค้างของเวลาหน่วยไมโครวินาที (unsigned int)

ตัวอย่างเช่น

delay(500); หมายถึง หยุดค้าง (หน่วงเวลา) ไว้เป็นเวลา 500 ไมโครวินาที

delay(1000); หมายถึง หยุดค้าง (หน่วงเวลา) ไว้เป็นเวลา 1000 ไมโครวินาที

ตัวอย่างที่ 3.1 การเขียนโค้ดโปรแกรมควบคุม LED สว่างและดับ(ไฟกระพริบ)

วัตถุประสงค์

1. รู้วิธีการเขียนโปรแกรมที่มีฟังก์ชั่น void setup()
2. รู้วิธีการเขียนโปรแกรมที่มีฟังก์ชั่น void loop()
3. รู้วิธีการส่งค่าลอจิก ออกทางพอร์ตเอาท์พุตที่ต้องการ
4. เข้าใจการใช้ฟังก์ชั่นหน่วงเวลา

1) วัสดุอุปกรณ์

- ไมโครคอนโทรลเลอร์ Arduino UNO R3
- บอร์ดทดลอง
- หลอดไฟ LED
- สายไฟ ผู้-ผู้ 3 เส้น
- สายเชื่อมต่อ USB

2) การเขียนโปรแกรมควบคุมไฟกระพริบ สามารถเขียนลำดับการทำงานโดยใช้รหัสลำลอง และผังงานได้ดังนี้

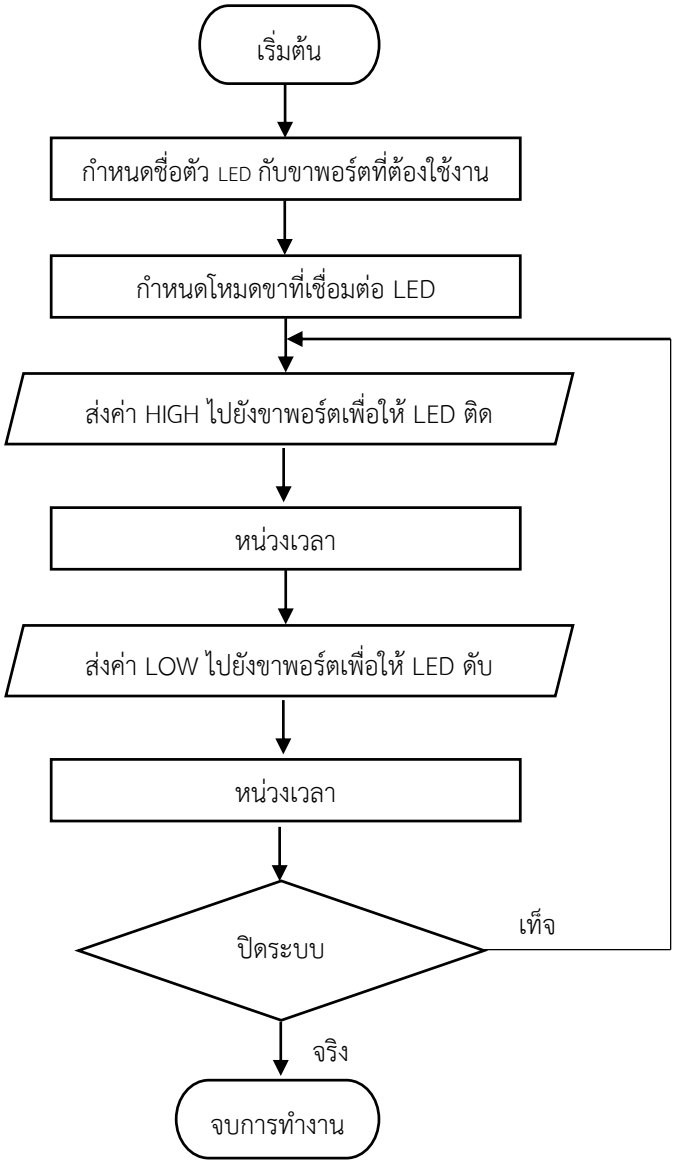
2.1 การเขียนรหัสลำลอง

เริ่มต้น

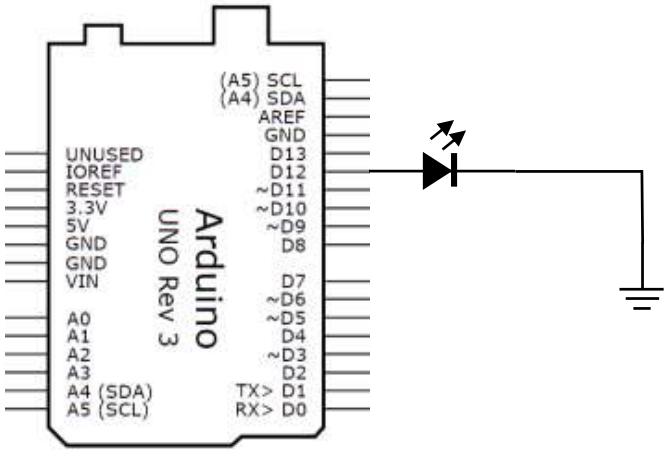
1. กำหนดชื่อตัว LED กับขาพอร์ตที่ต้องใช้งาน
2. กำหนดโหมดขาที่เชื่อมต่อ LED
3. ส่งค่า HIGH ไปยังขาพอร์ตเพื่อให้ LED ติด
4. หน่วงเวลา
5. ส่งค่า LOW ไปยังขาพอร์ตเพื่อให้ LED ดับ
6. หน่วงเวลา
7. วนกลับไปทำลำดับที่ 3 ซ้ำ

จบการทำงาน

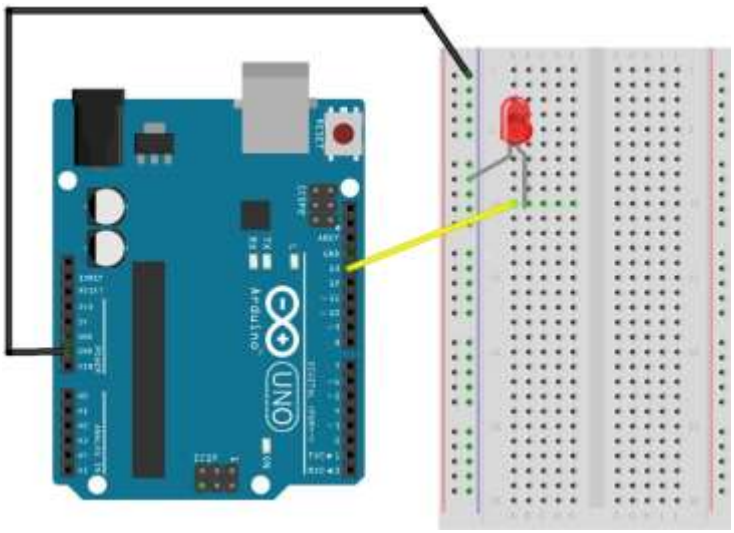
2.2 การเขียนผังงาน



3) วงจรสำหรับการทดลองดังรูป



4). ภาพตัวอย่างการต่ออุปกรณ์ที่ใช้ในการทดลอง



5) การเขียนโค้ดคำสั่งโปรแกรมที่ใช้ในการทดลอง

	คำสั่งโปรแกรม	คำอธิบายโปรแกรม
1	#define LED 13	// กำหนดชื่อตัว LED กับขาพอร์ตที่ต้องใช้งาน
2	void setup() {	
3	pinMode(LED, OUTPUT);	// กำหนดโหมดขาที่เชื่อมต่อ LED
4	}	
5	void loop() {	
6	digitalWrite(LED, HIGH);	//ส่งค่า HIGH ไปยังขาพอร์ตเพื่อให้ LED ติด
7	delay(1000);	//หน่วงเวลา 1 วินาที
8	digitalWrite(LED, LOW);	//ส่งค่า LOW ไปยังขาพอร์ตเพื่อให้ LED ดับ
9	delay(1000);	//หน่วงเวลา 1 วินาที
10	}	

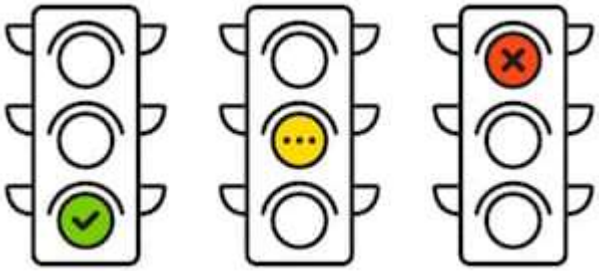
6. ทำการอัพโหลดโปแกรมเข้าสู่ไมโครคอนโทรลเลอร์

7. สังเกตการณ์ติดดับของหลอดไฟ LED

จากตัวอย่างที่ 3.1 การทดลองดังกล่าวสามารถนำมาใช้เป็นการทดสอบการทำงานของบอร์ด Arduino ว่า ' ทำงานได้สมบูรณ์หรือไม่ สามารถทำการอัพโหลดจากโปรแกรม Arduino IDE ได้หรือไม่ ซึ่งเป็นการเขียนโค้ดโปรแกรมให้ LED ที่อยู่บนบอร์ดติดดับสลับกันในเวลาที่กำหนดซึ่ง LED ดังกล่าวต่ออยู่กับขาดิจิตอลขา D13

คำถามชวนคิด?

นักเรียนคิดว่า ระบบไฟจราจรมีลำดับการทำงานเป็นอย่างไร ?



ตัวอย่างที่ 3.2 เรื่องการเขียนโค้ดโปรแกรมควบคุมไฟจราจร

วัตถุประสงค์

1. รู้วิธีการส่งค่าลอจิก ออกทางพอร์ตเอาท์พุตที่ต้องการ
2. เข้าใจการใช้ฟังก์ชั่นหน่วงเวลา
3. ประยุกต์การใช้ฟังก์ชั่นควบคุมเอาท์พุตแบบดิจิตอล

1) วัสดุอุปกรณ์

1. ไมโครคอนโทรลเลอร์ Arduino UNO R3
2. บอร์ดทดลอง
3. หลอดไฟ LED สีแดง 1 ดวง
4. หลอดไฟ LED สีส้ม 1 ดวง
5. หลอดไฟ LED สีเขียว 1 ดวง
6. สายไฟ ผู้-ผู้ 4 เส้น
7. สายเชื่อมต่อ USB

2) การเขียนโปรแกรมโปรแกรมควบคุมไฟจราจรสามารถเขียนลำดับการทำงานโดยใช้รหัสลำลอง และผังงานได้ดังนี้

2.1 รหัสลำลองไฟจราจร

เริ่มต้น

1. กำหนดชื่อตัวแปรสัมพันธ์กับขาพอร์ตที่ต้องใช้งาน
2. กำหนดโหมดขาที่เชื่อมต่อ
3. ส่งค่า HIGH ไปยังขาพอร์ตเพื่อให้ LED สีแดง ติด
4. หน่วงเวลา
5. ส่งค่า LOW ไปยังขาพอร์ตเพื่อให้ LED สีแดง ดับ
   ส่งค่า HIGH ไปยังขาพอร์ตเพื่อให้ LED สีเขียว ติด
6. หน่วงเวลา
7. ส่งค่า LOW ไปยังขาพอร์ตเพื่อให้ LED สีแดง ดับ
   ส่งค่า LOW ไปยังขาพอร์ตเพื่อให้ LED สีเขียว ดับ
   ส่งค่า HIGH ไปยังขาพอร์ตเพื่อให้ LED สีส้ม ติด
8. หน่วงเวลา
9. วนกลับไปทำลำดับที่ 3 ซ้ำ

จบการทำงาน

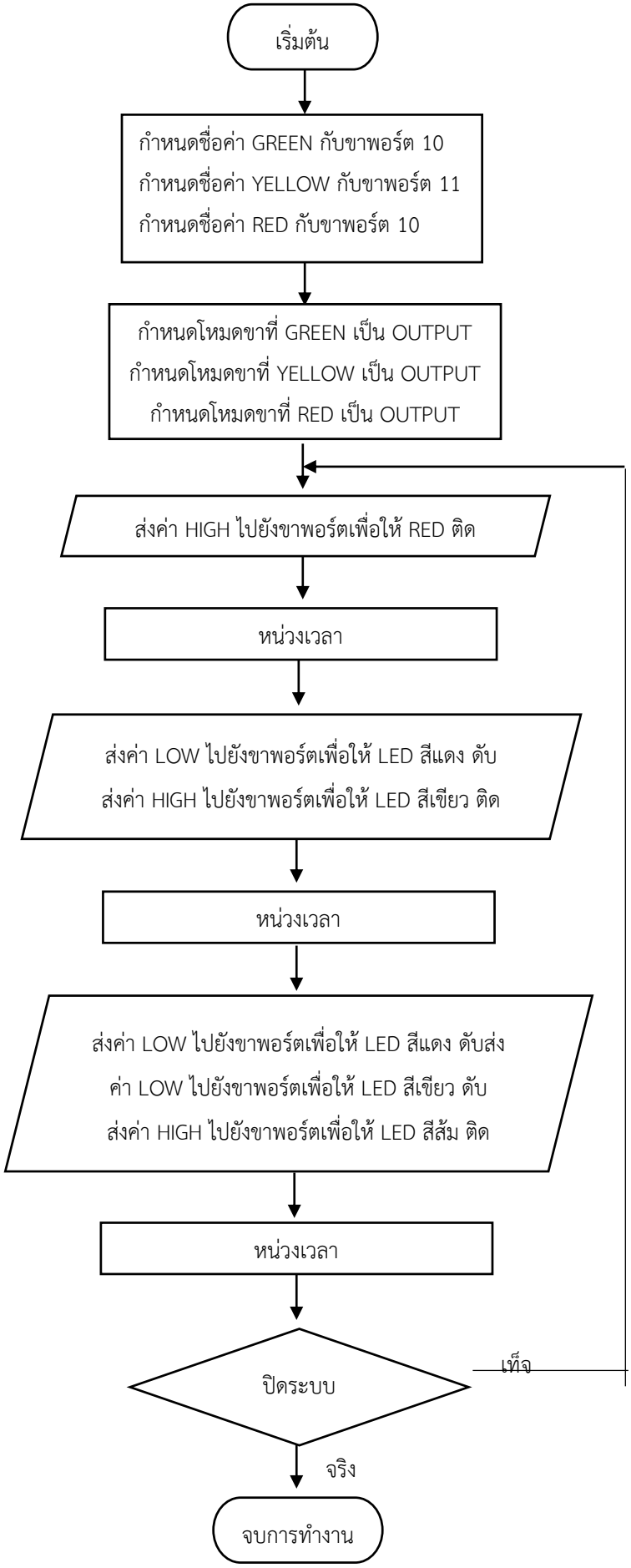
ภาพแสดงลำดับการทำงานของไฟจราจร

จังหวะที่ 1

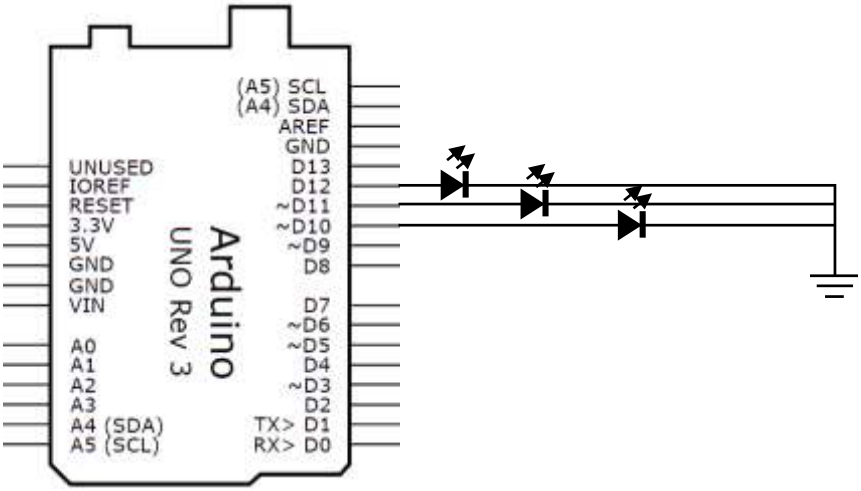
จังหวะที่ 2

จังหวะที่ 3

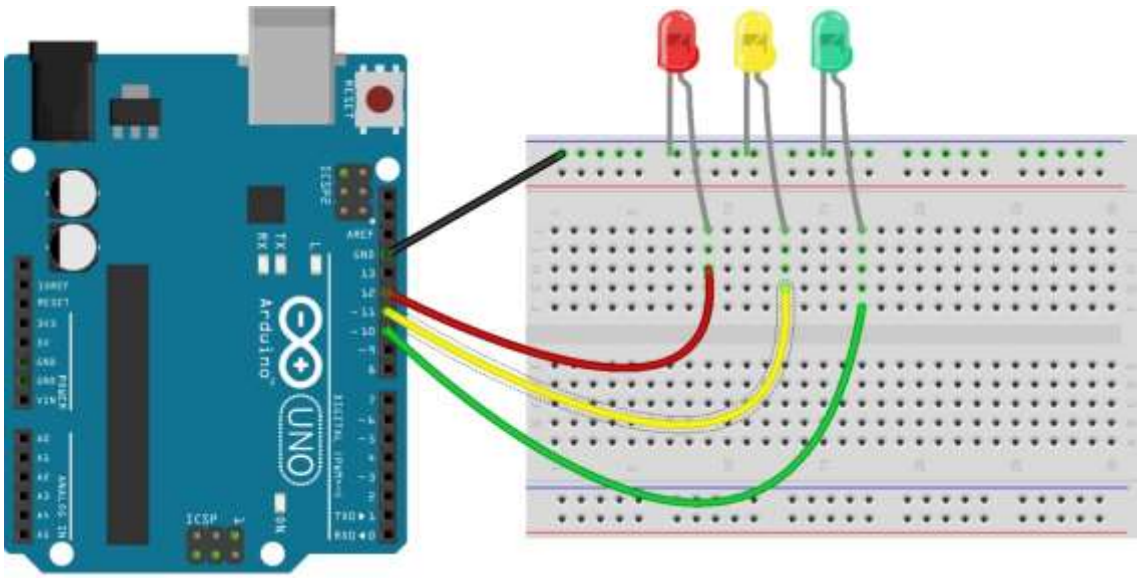
2.2 ผังงานไฟจราจร



3) วงจรสำหรับการทดลองดังรูป



4) ภาพตัวอย่างการต่ออุปกรณ์ที่ใช้ในการทดลอง



5) การเขียนโค้ดคำสั่งโปรแกรมที่ใช้ในการทดลอง

	คำสั่งโปรแกรม	คำอธิบายโปรแกรม
1	#define GREEN 10	// กำหนดชื่อ GREEN กับขาพอร์ต 10 ไฟเขียว
2	#define YELLOW 11	// กำหนดชื่อ YELLOW กับขาพอร์ต 11 ไฟเหลือง
3	#define RED 12	// กำหนดชื่อ RED กับขาพอร์ต 12 ไฟแดง
4	void setup() {	
5	pinMode(GREEN, OUTPUT);	//กำหนดให้พอร์ต GREEN เป็นพอร์ตเอาท์พุท
6	pinMode(YELLOW, OUTPUT);	//กำหนดให้พอร์ต YELLOW เป็นพอร์ตเอาท์พุท
7	pinMode(RED, OUTPUT);	//กำหนดให้พอร์ต RED เป็นพอร์ตเอาท์พุท
8	}	
9	void loop() {	
10	digitalWrite(RED, HIGH);	//ส่งค่า HIGH ไปยังขาพอร์ตเพื่อให้ไฟสีแดง ติด
11	delay(1000);	//หน่วงเวลา 1 วินาที
12	digitalWrite(RED, LOW);	//ส่งค่า LOW ไปยังขาพอร์ตเพื่อให้ไฟสีแดง ดับ
13	digitalWrite(GREEN, HIGH);	//ส่งค่า HIGH ไปยังขาพอร์ตเพื่อให้ไฟสีเขียว ติด
14	delay(1000);	//หน่วงเวลา 1 วินาที
15	digitalWrite(RED, LOW);	//ส่งค่า LOW ไปยังขาพอร์ตเพื่อให้ไฟสีแดง ดับ
16	digitalWrite(YELLOW, HIGH);	//ส่งค่า LOW ไปยังขาพอร์ตเพื่อให้ไฟสีเหลือง ติด
17	digitalWrite(GREEN, LOW);	//ส่งค่า HIGH ไปยังขาพอร์ตเพื่อให้ไฟสีเขียว ดับ
18	delay(1000);	//หน่วงเวลา 1 วินาที
19	}	

6. ทำการอัพโหลดโปแกรมเข้าสู่ไมโครคอนโทรลเลอร์

7. สังเกตการณ์ติดดับของหลอดไฟ LED ทั้ง 3 ดวง