บทนำ

ไมโครคอนโทรลเลอร์ (Microcontroller) หมายถึง ไมโครโปรเซสเซอร์ตัวหนึ่งที่มี หน่วยความจำชั่วคราว (RAM : Random Access Memory) หน่วยความจำถาวร (ROM : Read Only Memory) หน่วยรับ/ส่งข้อมูลจากภายนอก (Input/Output Port) อยู่ภายในตัวมันเพียงตัวเดียว (Single Chip) ซึ่งไมโครโปรเซสเซอร์ต้องอาศัยหน่วยความจำและหน่วยอินพุต/เอาต์พุตภายนอก ฉะนั้นไมโครคอนโทรลเลอร์จึงมีข้อดีกว่าไมโครโปรเซสเซอร์มาก ในเรื่องของความประหยัดและความสะดวกในการใช้งาน ปัจจุบันจึงนิยมใช้ไมโครคอนโทรลเลอร์ มากกว่าโดยเฉพาะในงานควบคุมทางด้านอุตสาหกรรม และเครื่องใช้ไฟฟ้าในชีวิตประจำวัน เช่น การควบคุมอุณหภูมิ การควบคุมหุ่นยนต์ เครื่องซักผ้าแบบโปรแกรมได้ การควบคุมการปิด-เปิดไฟฟ้าในอาคารการควบคุมไฟวิ่ง เป็นต้น

ในปัจจุบันผู้ผลิตได้พัฒนาให้ไมโครคอนโทรลเลอร์สามารถทำงานได้เร็วขึ้นโดยเพิ่มความสามารถในการรองรับคริสตอลความถี่ที่สูงขึ้น รวมไปถึงการปรับปรุงการทำงานภายในให้ไมโครคอนโทรลเลอร์ใช้จำนวนสัญญาณนาฬิกาในการสร้างแมชชีนไซเคิลน้อยลง โดยในบางรุ่น 1 แมชชีนไซเคิลใช้สัญญาณนาฬิกาเพียงแค่ 1 ลูกเท่านั้น

ซึ่งการทดลองสำหรับวิชา 305381 ไมโครโพรเซสเซอร์และภาษาแอสเซมบลี (Microprocessor and Assembly Language) จะใช้ไมโครคอนโทรลเลอร์ตระกูลMCS51 ในการทดลอง ใช้ภาษา Assembly ในการโปรแกรมควบคุมไมโครคอนโลเลอร์ และใช้งานโปรแกรม Flash Magic สำหรับดาวน์โหลดโปรแกรมลงบอร์ดไมโครคอนโทรลเลอร์ การทดลองนี้เป็นการทดลอง เกี่ยวกับการใช้งานเบื้องต้น สำหรับใช้ไมโครคอนโทรลเลอร์ตระกูล MCS51 89V51RD2 เป็นการทดลองการใช้พอร์ท P0-P3 เป็นอินพุต - เอาท์พุตพอร์ท โดยการใช้ภาษาแอสเซมบลีในการเขียนโปรแกรมคำสั่งในการทดลองต่างๆ และเรียนรู้การดาวน์โหลดโปรแกรมที่ผู้ทดลองเขียนขึ้นลงในตัว ไมโครคอนโทรลเลอร์ MCS51 89V51RD2

วัตถุประสงค์ของการทดลอง

1. เพื่อเรียนรู้การทำงานของ Microcontroller MCS-51
2. เพื่อเรียนรู้วิธีการใช้งานโปรแกรม Flash Magic สำหรับดาวน์โหลดโปรแกรมลงบอร์ดไมโครคอนโทรลเลอร์
3. เพื่อศึกษาการเขียนโปรแกรม rotate
4. เพื่อศึกษาการเขียนโปรแกรม Delay
5. เพื่อศึกษาการเขียนโปรแกรม คำสั่ง JB,
6. เพื่อควยคุมการทำงานของมอเตอร์
7.

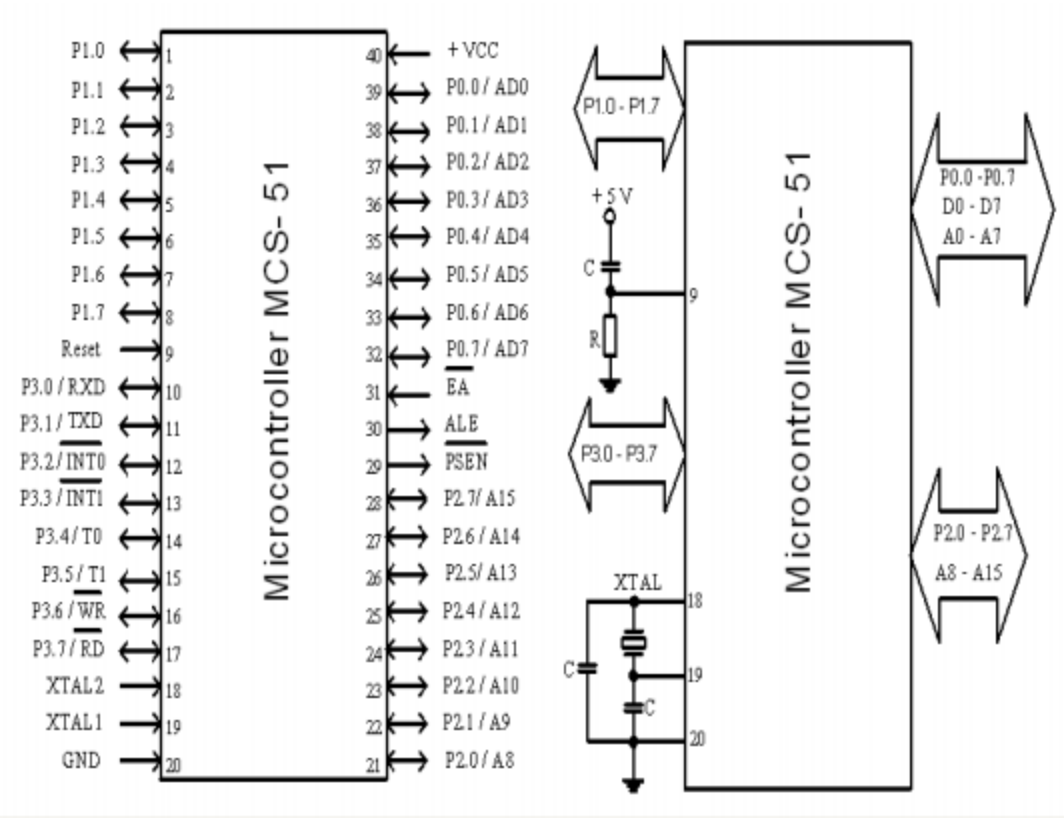
เนื้อหาและทฤษฎี

<u>ไมโครคอนโทรลเลอร์ MCS-51</u>

มีไทเมอร์/เคาน์เตอร์ขนาด 16 บิต 2 ตัว คือไทเมอร์ 0 และไทเมอร์ 1 ส่วนในไมโครคอนโทรลเลอร์อนุกรม 8052 ซึ่งออกมาทีหลังจะมีไทเมอร์/เคาน์เตอร์ 3 ตัว นั้นคือมีไทเมอร์ 2 เพิ่มขึ้นมา ไทเมอร์/เคาน์เตอร์แต่ละตัวสามารถเลือกใช้งานเป็นไทเมอร์ หรือเคาน์เตอร์ก็ได้ และทำงานได้อย่างเป็นอิสระต่อกัน ในการทำงานเป็นไทเมอร์นั้นจะใช้หลักการเพิ่มค่ารีจิสเตอร์ไทเมอร์ ทุก ๆ Machine Cycle ซึ่งมีช่วงเท่ากับ 12 คาบสัญญาณนาฬิกาที่ถูกสร้างขึ้นจากคริสตอลที่ต่อใช้งานให้กับไมโครคอนโทรลเลอร์นั้นเอง สำหรับบทความนี้จะอธิบายรายละเอียดและตัวอย่างการใช้งานไทเมอร์ 0 และ 1 เท่านั้นนะครับ สำหรับไทเมอร์ 2 สามารถอ่านรายละเอียดได้ที่บทความการใช้พอร์ตอนุกรมของไมโครคอนโทรลเลอร์ MCS-51

โครงสร้างพื้นฐานที่สำคัญของไมโครคอนโทรลเลอร์ตระกูล 8051 ซึ่งมีรายละเอียดดังนี้

1. เป็นไมโครคอนโทรลเลอร์ที่มีหน่วยประมวลผลกลางแบบ 8 บิต
2. มีคำสั่งคำนวณทางคณิตศาสตร์ และตรรกศาสตร์ (Boolean processor)
3. มีแอดเดรสบัสขนาด 16 บิตทำให้สามารถอ้างตำแหน่งหน่วยความจำโปรแกรม และหน่วยความจำข้อมูลได้ 64 กิโลไบต์
4. มีหน่วยความจำ (RAM) ภายในขนาด 128 ไบต์ (8051/8031) หรือ 256 ไบต์ (8052/8032)
5. มีพอร์ตอนุกรมทำงานแบบดูเพล็กซ์เต็ม (Full Duplex) 1 พอร์ต
6. มีพอร์ตอินพุต/เอาต์พุตแบบขนานจำนวน 32 บิต
7. มีไทเมอร์ 2 ตัว (8051/8031) หรือ 3 ตัว (8052/8032)
8. มีวงจรควบคุมการเกิดอินเตอร์รัปต์ 5 ประเภท (8051/8031) หรือ 6 ประเภท (8052/8032)
9. มีวงจรออสซิลเลเตอร์ภายในตัว



<u>ไมโครคอนโทรลเลอร์ MCS-51</u>

ประกอบด้วยพอร์ตที่ใช้เป็นอินพุต/เอาต์พุตเพื่อติดต่อกับอุปกรณ์รอบนอกได้ 4 พอร์ต คือP0, P1, P2 และ P3 มีรายละเอียดแต่ละพอร์ตดังนี้

-P0 (P0.0-P0.7) เป็นอินพุตหรือเอาต์พุตพอร์ตถ้ามีการขยายหน่วยความจำภายนอกหรืออินพุต/ เอาต์พุตพอร์ตภายนอกจะใช้เป็น Data Bus (D0-D7) และ Address Bus (A0-A7)

-P1 (P1.0-P1.7) เป็นอินพุตหรือเอาต์พุตพอร์ต

-P2 (P2.0-P2.7) เป็นอินพุตหรือเอาต์พุตพอร์ต ถ้ามีการขยายหน่วยความจำภายนอกจะใช้เป็น Address Bus (A8-A15)

-P3 (P3.0-P3.7) เป็นอินพุตหรือเอาต์พุตพอร์ต ถ้าไม่ใช้เป็นอินพุต /เอาต์พุตพอร์ตก็สามารถทำหน้าที่ตามชื่อหลังได้ดังนี้

-P3.0/RxD (Receive Data) ใช้เป็นขาอินพุตสำหรับรับข้อมูลจากการสื่อสารแบบอนุกรม

-P3.1/TxD (Transmit Data) ใช้เป็นขาเอาต์พุตสำหรับส่งข้อมูลจากการสื่อสารแบบอนุกรม

-P3.2/ INT0 (Interrupt 0) รับสัญญาณขัดจังหวะจากภายนอก No. 0

-P3.3/ INT1 (Interrupt 1) รับสัญญาณขัดจังหวะจากภายนอก No. 1

-P3.4/T0 (Timer/Counter0) สามารถโปรแกรมได้ว่าจะให้เป็น Timer หรือ Counter ถ้าใช้สัญญาณ Clock จากภายนอกเข้ามาจะเป็น Counter ถ้าใช้สัญญาณ Clock จากภายในจะเป็น Timer

-P3.5/T1 (Timer/Counter1) ทำหน้าที่ทำนองเดียวกับ P3.4/T0
-P3.6/ WR (Write) ส่งสัญญาณควบคุมการเขียนข้อมูลจาก MCS-51ไปยังภายนอก
-P3.7/ RD (Read) ส่งสัญญาณควบคุมการอ่านข้อมูลจากภายนอกเข้ามายัง MCS-51 -Reset เป็นขาอินพุต

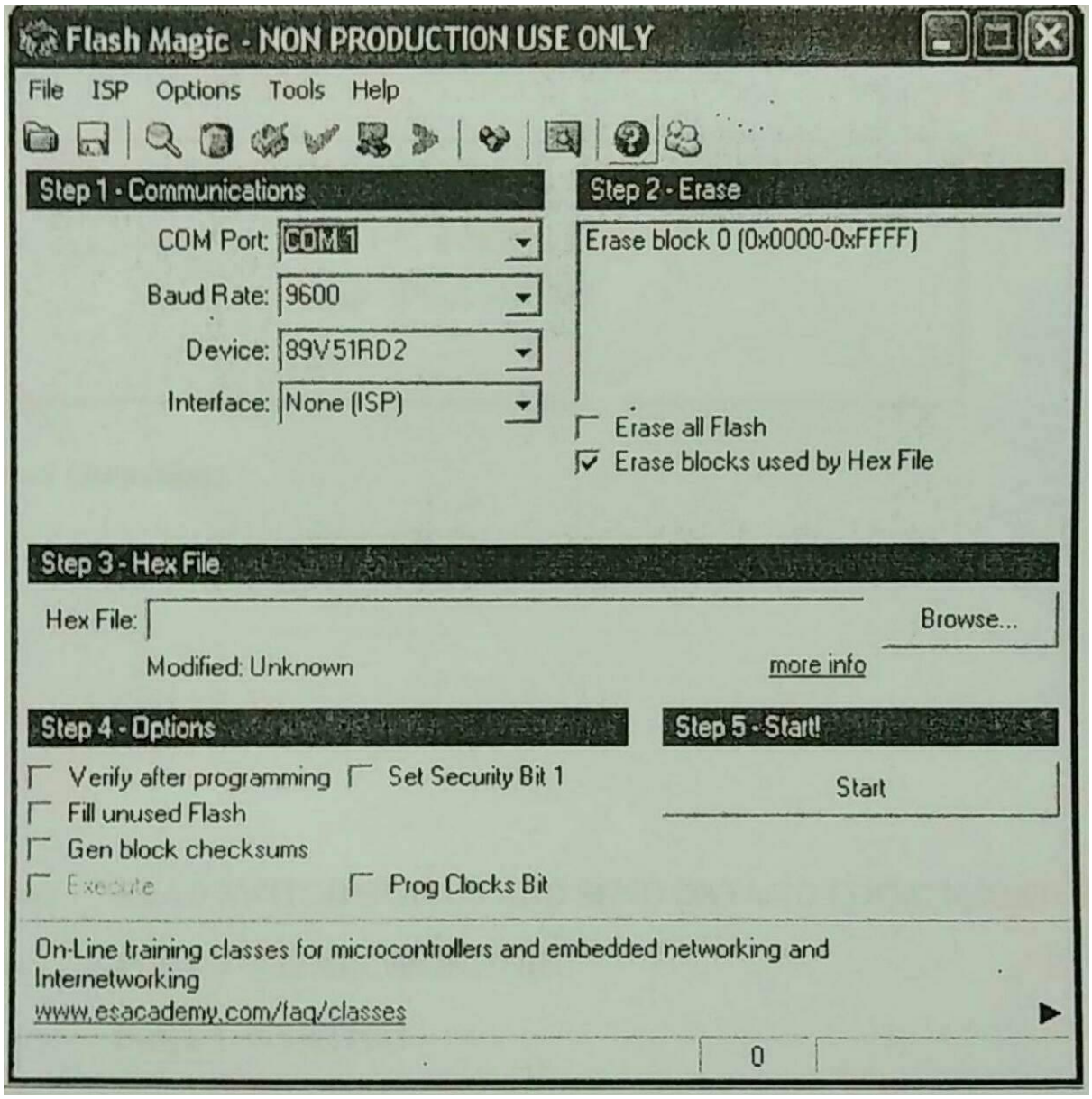
<u>การใช้งานโปรแกรม Flash Magic สำหรับดาวน์โหลดโปรแกรมลงบอร์ดไมโครคอนโทรลเลอร์</u>

ในการเขียนโปรแกรมภาษาแอสแซมบลีเพื่อควบคุมการทำงานของไมโครคอนโทรลเลอร์นั้นจะต้องมีการคอมไฟล์โปรแกรมดังกล่าวให้เป็น Hex File แล้วจึงทำการดาวน์โหลดไฟล์นั้นลงบอร์ดไมโครคอนโทรลเลอร์ โดยในการทดลองนี้จะเลือกใช้งานโปรแกรม Flash Magic สำหรับดาวน์โหลดโปรแกรม โดยนิสิตจำเป็นต้องเลือก Option ให้สอดคล้องกับบอร์ดทดลองที่ใช้งานดังนี้

1. เลือกเมนู Options >> Advanced Options ... จะปรากฏหน้าต่าง Advanced Options
2. คลิกที่ TAB ชื่อ Hardware Config แล้วทำการยกเลิกเครื่องหมายถูกที่ปรากฏในช่อง Use DTR to control RST

เมื่อปรับเลือกค่า Option ให้เหมาะสมต่อการใช้งานแล้ว โปรแกรม Flash Magic จะพร้อมสำหรับการโหลดโปรแกรมลงชิปต่อไป โดยมีขั้นตอนดังนี้

- **STEP 1:** เลือกเบอร์ไมโครคอนทรลเลอร์เป็น 89V51RD2 จากนั้นให้กำหนดพอร์ตสื่อสารข้อมูล COM-x ให้ตรงกับพอร์ตสื่อสารของคอมพิวเตอร์ที่ใช้งานอยู่ จากนั้นให้กำหนดค่าความเร็วที่ใช้สื่อสารข้อมูล (ในที่นี้กำหนดให้เป็น 9600 bps ดังแสดงในรูปที่ 1)
- **STEP 2:** เลือกรูปแบบของการลบข้อมูลภายในหน่อยความจำโปรแกรม (Flash Memory) ก่อนที่จะดำเนินการโปรแกรมข้อมูลใหม่ลงไป โดยกำหนดเป็น Erase block used Hex File ซึ่งเป็นการลบข้อมูลเฉพาะ block ที่ต้องการสำหรับการเขียนโปรแกรมข้อมูลใหม่เท่านั้น(โดยจะส่งผลให้การทำงานเร็วกว่าการลบข้อมูลทั้งหมดด้วยคำสั่ง Erase all Flash)
- **STEP 3:** คลิกปุ่ม browse ... เพื่อเปิดหน้าต่าง Select Hex File และเลือกไฟล์ที่ต้องการโปรแกรมลงสู่ไมโครคอนโทรลเลอร์



รูปที่ 1: หน้าต่างของโปรแกรม Flash Magic

- **STEP 4:** เลือก Options การทำงานเพิ่มเติมตามต้องการ
- **STEP 5:** กดปุ่ม Start เพื่อเริ่มขั้นตอนการโปรแกรมลงชิปไมโครคอนโทรลเลอร์

เมื่อปรากฏหน้าต่าง Reset Device ขึ้นมาแล้ว ให้กดปุ่ม RESET บนบอร์ดไมโครคอนโทรลเลอร์ ซึ่งจะเป็นการเริ่มต้นการดาวน์โหลดโปรแกรมลงสู่ชิปทันที โดยจะสามารถสังเกตขั้นตอนการทำงานได้จาก Status bar ที่ขอบด้านล่างของโปรแกรมและเมื่อขั้นตอนการโปรแกรมเสร็จสมบูรณ์ (Finished) ก็ให้กดปุ่ม RESET บนบอร์ดไมโครคอนโทรลเลอร์อีกครั้ง ไมโครคอนโทรลเลอร์จะเริ่มต้นทำงานตามโปรแกรมที่ได้ดาวน์โหลดลงไปใหม่ทันที

การทดลองในรายวิชานี้มีวัตถุประสงค์เพื่อเรียนรู้วิธีการเขียนโปรแกรมภาษาแอสแซมบลีสำหรับใช้ควบคุมการทำงานของไมโครคอนโทรเลอร์ตระกูล MCS-51 โดยเนื้อหาในการทดลองจะประกอบด้วยการทดลองจำนวน 5 การทดลอง

มอเตอร์ คือ เครื่องกลไฟฟ้าที่ทำหน้าที่เปลี่ยนพลังงานไฟฟ้าเป็นพลังงานกล โดยสร้างมอเตอร์จะเหมือนกับเครื่องกำเนิดไฟฟ้ากระแสตรงทุกอย่าง จะมีข้อแตกต่างออกไปบ้างก็เพียงเล็กน้อยเท่านั้น ทั้งนี้เพราะว่าสภาพที่นำมาใช้งานแตกต่างกัน ตัวอย่างเช่น เครื่องกำเนิดไฟฟ้าทั่วไปจะเป็นชนิดเปิด (open type) กล่าวคือ ขดลวดอาร์เมเจอร์ และขดลวดสนามแม่เหล็กจะพันเป็นแบบเปิดทั้งนี้ก็เพื่อป้องกันไม่ให้เกิดความเสียหายขึ้นกับลวดอย่างไรก็ตามเครื่องกลไฟฟ้ากระแสตรงเครื่องเดียว สามารถใช้ทำเป็นเครื่องกำเนิดไฟฟ้าหรือเป็นมอเตอร์ไฟฟ้าได้

หลักการทำงานของมอเตอร์

เมื่อมีกระแสไหลในขดลวดตัวนำที่พันอยู่บนแกนอาร์เมเจอร์ จะเกิดสันแรงแม่เหล็กรอบ ๆ ตัวนำ และทำปฏิกิริยากับเส้นแรงแม่เหล็กที่เกิดจากขั้วแม่เหล็กของมอเตอร์ ทำให้เกิดแรงผลักขึ้นบนตัวนำทำให้อาร์เมเจอร์หมุนไปได้ ขดลวดที่มีกระแสไฟฟ้าไหลและวางอยู่บนแกนของอาร์เมเจอร์ โดยวางห่างจากจุดศูนย์กลางเป็นระยะ r กำหนดให้กระแสไฟฟ้าไหลเข้าขดลวดที่ปลาย A และไหลออกที่ปลาย B จากคุณสมบัติของเส้นแรงแม่เหล็กจะไม่ตัดผ่านซึ่งกันและกัน ดังนั้นปริมาณของเส้นแรงแม่เหล็กจะมีจำนวนมากที่ด้านบนของปลาย A จึงทำให้เกิดแรง F1 กดตัวนำ A ลงด้านล่างและขณะเดียวกันที่ปลาย B นั้น เส้นแรงแม่เหล็กจะมีปริมาณมากที่ด้านหน้าทำให้เกิดแรง F2 ดันให้ตัวนำ B เคลื่อนที่ด้านบนของแรง F1 และ F2 นี้เองทำให้อาร์เมเจอร์ของมอเตอร์เกิดการเคลื่อนที่ไปได้ ดังนั้นการทำงานของมอเตอร์จึงขึ้นอยู่กับหลักการที่ว่า เมื่อเอาตัวนำที่มีกระแสไฟฟ้าไหลผ่านไปวางในสนามแม่เหล็ก มันจึงพยายามทำให้ตัวนำเคลื่อนที่ไปในทิศทางที่ตั้งฉากกับสนามแม่เหล็ก

คุณสมบัติของมอเตอร์

คุณสมบัติของมอเตอร์ไฟฟ้าสามารถแบ่งออกได้เป็น 2 ลักษณะ คือ คุณสมบัติทั่วไปและคุณสมบัติทางเทคนิค ดังนี้

คุณสมบัติทั่วไป

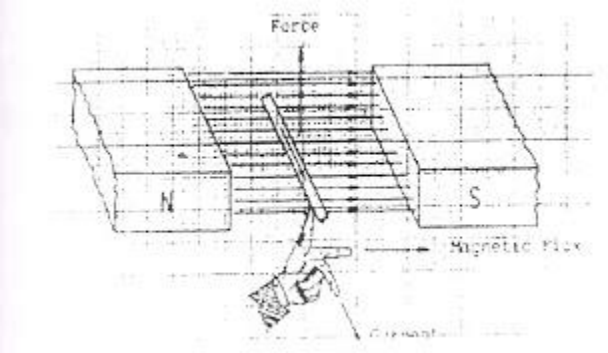
เป็นคุณสมบัติประจำตัวของมอเตอร์ ไฟฟ้าแต่ละประเภทที่ควรจะทราบอย่างกว้าง ๆ โดยมิได้เจาะลึกเข้าไปในเนื้อหาเชิงวิชาการแต่อย่างใด ได้แก่ ลักษณะโครงสร้าง ลักษณะงาน ลักษณะของวงจรเช่นคุณสมบัติ ของมอเตอร์อนุกรม คือ ลักษณะโครงสร้าง ประกอบด้วยลวดหนามแม่เหล็กที่มีความต้านทานต่ำมาก (พันด้วยลวดทองแดงเส้นใหญ่น้อยรอบแกนขั้วแม่เหล็ก) ต่อเป็นอนุกรมกับอาร์เมเจอร์และต่อโดยตรงกับแรงดันเมน ลักษณะวงจร A1 – A2 เป็นอาร์เมเจอร์ต่อเป็นอนุกรมกับขดลวดสนามแม่เหล็กชุดอนุกรม D1 – D2 และต่อโดยตรงกับสายเมน L+, L- และลักษณะสนามแม่เหล็กทำให้ความเร็วสูงเมื่อโหลดลง จึงเป็นมอเตอร์ที่หมุนไม่คงที่ความเร็วเปลี่ยนแปลงไปตามโหลดจะเหมาะสมอย่างยิ่งที่จะใช้เป็นมอเตอร์สตาร์ทเครื่องพ่นน้ำ

คุณสมบัติทางเทคนิค

เป็นคุณสมบัติประจำเครื่องกลไฟฟ้าแต่ละประเภทเช่นเดียวกัน ที่ให้รายละเอียดซึ่งเจาะลึกเข้าไปในเชิงวิชาการ สามารถทดสอบและวัดด้วยเครื่องวัดได้ด้วยวิธีทดลองในห้องปฏิบัติการทดลอง ส่วนใหญ่จะแสดงด้วยกราฟเพื่อแสดงให้เห็นความสัมพันธ์ระหว่างค่าหนึ่งกับอีกค่าหนึ่ง เช่น สมรรถในการกำเนิดแรงเคลื่อนไฟฟ้าของเครื่องกำเนิดไฟฟ้าแสดงด้วย “กราฟแม่เหล็กอิ่มตัว (Saturation หรือ Magnetization curve)” สมรรถนะในการจ่ายโหลดของเครื่องกำเนิดไฟฟ้าแสดงด้วย External Characteristic ส่วนคุณสมบัติทางเทคนิคของมอเตอร์จะแสดงด้วย Performance Curve ซึ่งได้แก่ สมรรถนะในการหมุนขับโหลด (Speed load Curves หรือ Speed/load Characteristic) แสดงให้เห็นความสัมพันธ์ระหว่างความเร็วรอบกับกระแสมอเตอร์ (n = ความเร็วรอบให้อยู่บนแกน Y หรือ Ordinate และ Ia = กระแสอาร์เมเจอร์ให้อยู่บนแกน X หรือ abscissae) หรืออาจให้แสดงความสัมพันธ์ระหว่างความเร็วรอบ
(n เ เป็น ordinate หรือ แกน Y) กับทอร์ค หรือกำลังที่หมุนขับงาน (T= ทอร์ด, P=กำลังวัตต์หรือกิโลวัตต์ ให้อยู่บนแกน x หรือ abscissae) จุดประสงค์เพื่อต้องการแสดงให้เห็นถึงความเปลี่ยนแปลงของความเร็วรอบของมอเตอร์ที่หมุนขับโหลดว่าจะมีการเปลี่ยนแปลงไปอย่างไรเมื่อโหลดเปลี่ยนแปลงไป

กฎมือซ้ายสำหรับมอเตอร์

เนื่องจากมีความสัมพันธ์อย่างแน่นอนเกิดขึ้นระหว่าทิศทางของสนามแม่เหล็ก ทิศทางของกระแสไฟฟ้าในตัวนำและทิศทางที่ตัวนำเคลื่อนที่ซึ่งมีความสัมพันธ์ของปริมาณเหล่านี้ให้ตั้งเป็นกฎมอเตอร์ขึ้น ซึ่งกฎนี้ได้นำไปใช้แบบเดียวกันกับกฎมือขวาของเครื่องกำเนิดไฟฟ้าเป็นแต่เพียงใช้มือซ้ายแทนเท่านั้น กฎนี้ ได้แสดงให้เห็นดังรูปที่ 1 และได้กล่าวไว้ดังนี้คือ กลางหัวแม่มือ นิ้วชี้และนิ้วกลาง ให้ตั้งฉากซึ่งกันและกัน โดยใช้นิ้วชี้ ชี้ไปตามทิศทางของสนามแม่เหล็ก (Magnotic flux = B) นิ้วกลางชี้ไปตามทิศทางการไหลของกระแสไฟฟ้า (Current = I) แล้วหัวแม่มือจะบอกทิศทางของการเคลื่อนที่ของตัวนำ (Force = F)



แรงที่เกิดขึ้นในตัวนำ

การกระทำของแรงที่เกิดขึ้นเป็นตัวนำที่มีกระแสไฟฟ้าไหลผ่านในขณะที่มันวางอยู่ในสนามแม่เหล็กจะเป็นปฏิภาคโดยตรงกับความหนาแน่นของเส้นแรงแม่เหล็ก ความยาวของตัวนำและค่ากระแสไฟฟ้าที่ไหลผ่านตัวนำแรงที่เกิดขึ้นบนตัวนำสามารถหาได้จากสมการ

F = BIL

เมื่อ F = แรงที่เกิดขึ้นบนตัวนำหนึ่งตัว หน่วย นิวตัน

B = ความหนาแน่นสนามแม่เหล็ก หน่วย Wb/m2

I = กระแสที่ไหลในตัวนำ หน่วย แอมแปร์ (A)

L = ความยาวของตัวนำ หน่วย เมตร (m)

แรงเคลื่อนไฟฟ้าต่อต้าน เกิดขึ้นเนื่องจากเมื่อขดลวดตัวนำหมุนอยู่ในสนามแม่เหล็ก มันจะติดกับเส้นแรงแม่เหล็ก แรงเคลื่อนไฟฟ้าที่เหนี่ยวขึ้นในขดลวด แรงเคลื่อนไฟฟ้าเหนี่ยวนำที่เกิดขึ้นจะมีทิศทางขัดขวางกับแรงเคลื่อนที่ไฟฟ้าที่จ่ายให้มอเตอร์ จึงเรียกว่า “แรงเคลื่อนไฟฟ้าต่อต้าน” (Back e.m.f) ซึ่งมันจะเกิดขึ้นในขดลวดอาร์เมเจอร์เสมอ ดังนั้นแรงเคลื่อนไฟฟ้าที่มีผลต่อการใช้งานจริง ๆ ในอาร์เมเจอร์จึงมีค่าเท่ากับแรงเคลื่อนไฟฟ้าที่จ่ายให้ลบด้วยแรงเคลื่อนไฟฟ้าต้านกลับจึงเขียนสมการได้ดังนี้

Vt = Ia + Eb

หรือ IaRa = Vt - Eb

เมื่อ Eb = แรงเคลื่อนไฟฟ้าที่ต้านกลับ หน่วยโวลท์ (V)

Vt = แรงเคลื่อนไฟฟ้าที่จ่ายให้กับมอเตอร์ หน่วยโวลท์ (V)

Ia = กระแสที่ไหลในอาร์เมเจอร์ หน่วยแอมแปร์ (A)

Ra = ความต้านทานของขดลวดในอาร์เมเจอร์ หน่วยโฮม ()

สมการแรงเคลื่อนไฟฟ้าของมอเตอร์

จากวงจรสามารถเขียนเป็นสมการได้คือ

Vt = Eb + IaRa

เมื่อ Vt = แรงเคลื่อนไฟฟ้าที่จ่ายให้กับมอเตอร์ หน่วยโวลท์ (V)

Eb = แรงเคลื่อนไฟฟ้าที่ต้านกลับ หน่วยโวลท์ (V)

IaRa = แรงเคลื่อนไฟฟ้าตกคร่อมในอาร์เมเจอร์ หน่วยโวลท์ (V)

กำลังที่เกิดขึ้นในมอเตอร์

จากสมการแรงเคลื่อนไฟฟ้าของมอเตอร์

Vt = Eb + IaRa

นำเอาค่า Ia คูณตลอดเพื่อหา Power จะได้คือ

Vt Ia = Ia Eb+ Ia2Ra

จะได้ Vt Ia = กำลังงานจ่ายให้กับมอเตอร์ หน่วยวัตต์ (W)

Eb Ia = กำลังงานที่เกิดขึ้นจากอาร์เมเจอร์ หน่วยวัตต์ (W)

Ia2Ra = กำลังงานการสูญเสียที่เกิดขึ้นที่อาร์เมเจอร์ หน่วยวัตต์ (W)

MICROCONTROLLER MCS-51 – LAB 5

Summary: STEPPING MONITOR CONTROL PROGRAM

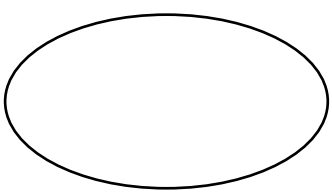
LAB 5-1

Description: CONTROL THE STEPPING MONITOR BY FULL STEP METHOD

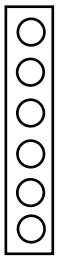
Hardware: PORT 2 -> STEPPING MOTOR

NOTE: 1. ให้นิสิตเชื่อมต่อขาขาสัญญาณของมอเตอร์ดังรูป

2. การทดลองนี้อาจมีปัญหาปริมาณกระแสไม่เพียงพอ นิสิตอาจแก้ปัญหาโดยการเชื่อมต่อสัญญาณที่แต่ละเฟสของสเต็ปมอเตอร์เข้ากับ LED บนบอร์ด ANALAB (เพื่อช่วยในการขับกระแสและเพื่อใช้สังเกตลำดับการทำงาน)



Stepping Monitor



Gnd	————	To Gnd on ANALAB and MCS-51
IND	————	To P2.3
INC	————	To P2.2
INB	————	To P2.1
INA	————	To P2.0
Vcc	————	To Vcc on ANALAB

รูป: ขาสัญญาณของ Stepping Monitor ที่ใช้ในการทดลอง

ASM Code:

```
                ORG     0000H
                STEP_MOTOR  EQU     P2
                MOV     DPTR, #STEP_DRIVE

START:          MOV     R0, #04
                MOV     R1, #00H

STEP_LOOP:      MOV     A, R1
                MOVC    A, @A+DPTR
                MOV     STEP_MOTOR, A
                INC     R1
                CALL    DELAY
                DINZ    R0, STEP_LOOP
                SJMP    START

STEP_DRIVE:     DB      01H, 02H, 04H, 08H

DELAY:          MOV     R5, #1
DELAT0:         MOV     R6, #255
DELAY1:         MOV     R7, #255
DELAY2:         DJNZ    R7, #DELAY2
                DJNZ    R6, #DELAY1
                DJNZ    R5, #DELAY0
                RET
                END
```

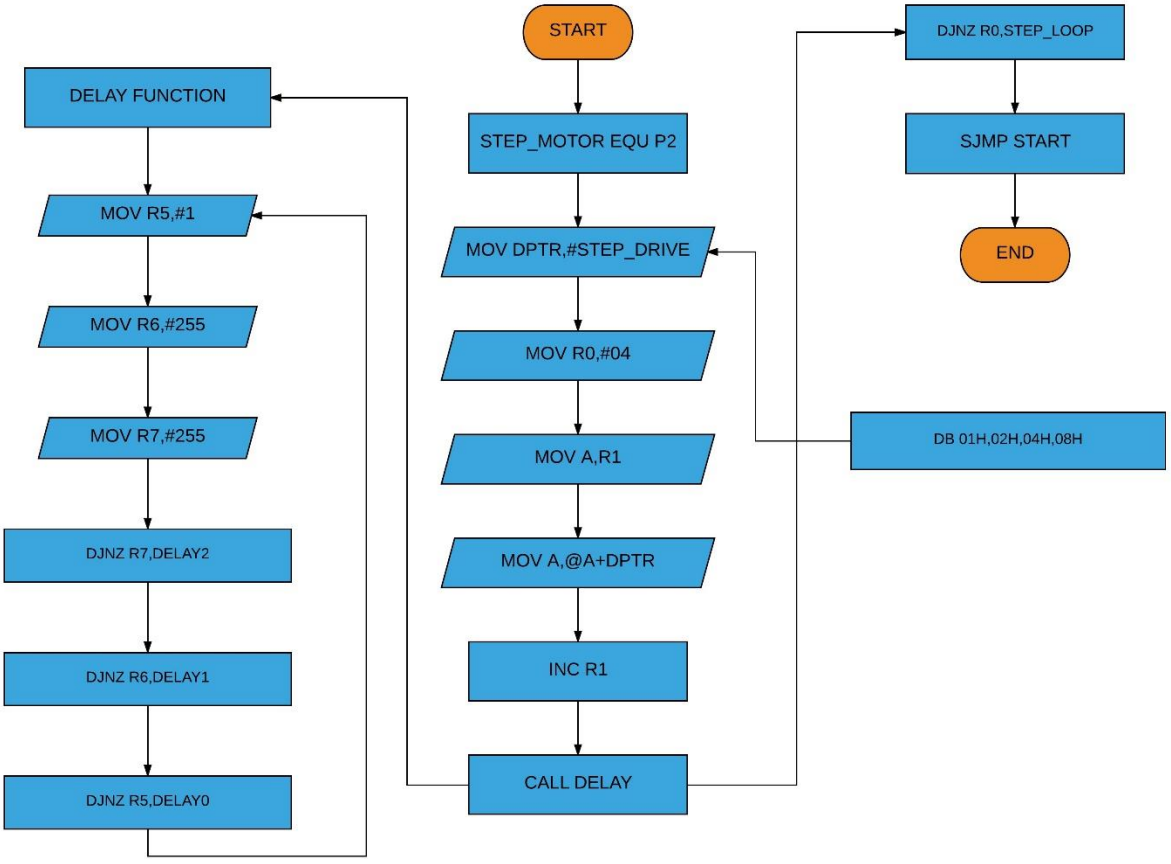
วิธีการทดลอง

1. เขียนโค้ดที่โจทย์กำหนดให้ จากนั้นทำการแปลงไฟล์ให้ได้ไฟล์นามสกุล .hex

2. ทำการรัน ASM Code จากโปรแกรม Flash Magic เพื่อดาวน์โหลดโค้ดลงบอร์ดไมโครคอนโทรลเลอร์ผ่านสายสื่อข้อมูล

3. สังเกตผลการทดลองและบันทึกผลการทดลอง

Flow Chart :



ORG 0000H	เริ่มต้นการทำงานที่ตำแหน่ง 0000H
STEP_MOTOR EQU P2	ประกาศตัวแปร STEP_MOTOR สำหรับเรียกแทนพอร์ต 2
MOV DPTR, #STEP_DRIVE	นำ DPTR ชี้ไปที่ STEP_DRIVE
START: MOV R0, #04	เก็บค่า 04 ไว้ใน รีจิสเตอร์ R0
MOV R1, #00H	กำหนดให้ รีจิสเตอร์ R1 = 0
STEP_LOOP: MOV A, R1	ย้ายค่าใน R1 ไปเป็นข้อมูลใน A

MOVC A, @A+DPTR	นำค่าในตาราง DATA มาไว้ที่รีจิสเตอร์ A
MOV STEP_MOTOR, A	ย้ายค่าในรีจิสเตอร์ A ไปเป็นข้อมูลใน STEP_MOTOR
INC R1	เพิ่มค่า R1 ไป 1 ค่า
CALL DELAY	เรียกใช้โปรแกรมย่อยหน่วงเวลา
DJNZ R0, STEP_LOOP	ลดค่า R0 ลง 1 ค่า ถ้าไม่เท่ากับ 0 ให้กระโดดไปที่ตำแหน่ง STEP_LOOP
SJMP START	กระโดดไปทำงานที่ตำแหน่ง START
STEP_DRIVE: DB 01H, 02H, 04H, 08H	
DELAY: MOV R5, #1	โปรแกรมย่อยเพื่อหน่วงเวลา เก็บค่า 1 ไว้ในรีจิสเตอร์ R5
DELAY0: MOV R6, #255	เก็บค่า 255 ไว้ใน รีจิสเตอร์ R6
DELAY1: MOV R7, #255	เก็บค่า 255 ไว้ใน รีจิสเตอร์ R7
DELAY2: DJNZ R7, #DELAY2	ลดค่า R7 ลง 1 ค่า ถ้าไม่เท่ากับ 0 ให้กระโดดไปที่ตำแหน่ง DELAY2
DJNZ R6, #DELAY1	ลดค่า R6 ลง 1 ค่า ถ้าไม่เท่ากับ 0 ให้กระโดดไปที่ตำแหน่ง DELAY1
DJNZ R5, #DELAY0	ลดค่า R5 ลง 1 ค่า ถ้าไม่เท่ากับ 0 ให้กระโดดไปที่ตำแหน่ง DELAY0
RET	สิ้นสุดโปรแกรมหน่วงเวลา
END	ตำแหน่งสิ้นสุดโปรแกรม

Results and discussion:

มอเตอร์หมุนทวนเข็มนาฬิกา

LAB 5-2

Description: CONTROL SPEED BY BIT SWITCH

Hardware: PORT 1-> SWITCH

PORT 2 -> STEPPING MOTOR

```
                ORG     0000H
                STEP_MOTOR  EQU     P2
                SWITCH      EQU     P1
                SW0         EQU     P1.0

                MOV     SWITCH, #0FFH
                MOV     DPTR, STEP_DRIVE
CHK:            JB      SW0, HI_SPEED
REGULAR_SPEED:  MOV     R0, #04
                MOV     R1, #00H
STEP_LOOP0:     MOV     A, R1
                MOVC    A, @A+DPTR
                MOV     STEP_MOTOR, A
                INC     R1
                CALL    DELAY_R
                DINZ    R0, STEP_LOOP0
                SJMP    CHK

HI_SPEED:       MOV     R0, #04
                MOV     R1, #00H
STEP_LOOP1:     MOV     A, R1
                MOVC    A, @A+DPTR
                MOV     STEP_MOTOR, A
                INC     R1
                CALL    DELAY_H
                DINZ    R0, STEP_LOOP1
                SJMP    CHK

STEP_DRIVE:     DB      01H, 02H, 04H, 08H

DELAY_R:        MOV     R5, #1
DELAT_R0:       MOV     R6, #255
DELAT_R1:       MOV     R7, #255
DELAT_R2:       DJNZ    R7, #DELAY_R2
                DJNZ    R6, #DELAY_R1
                DJNZ    R5, #DELAY_R0
                RET
DELAY_H:        MOV     R6, #55
DELAT_H0:       MOV     R7, #255
DELAT_H1:       DJNZ    R7, DELAT_H1
                DJNZ    R6, DELAT_H0
                RET
                END
```

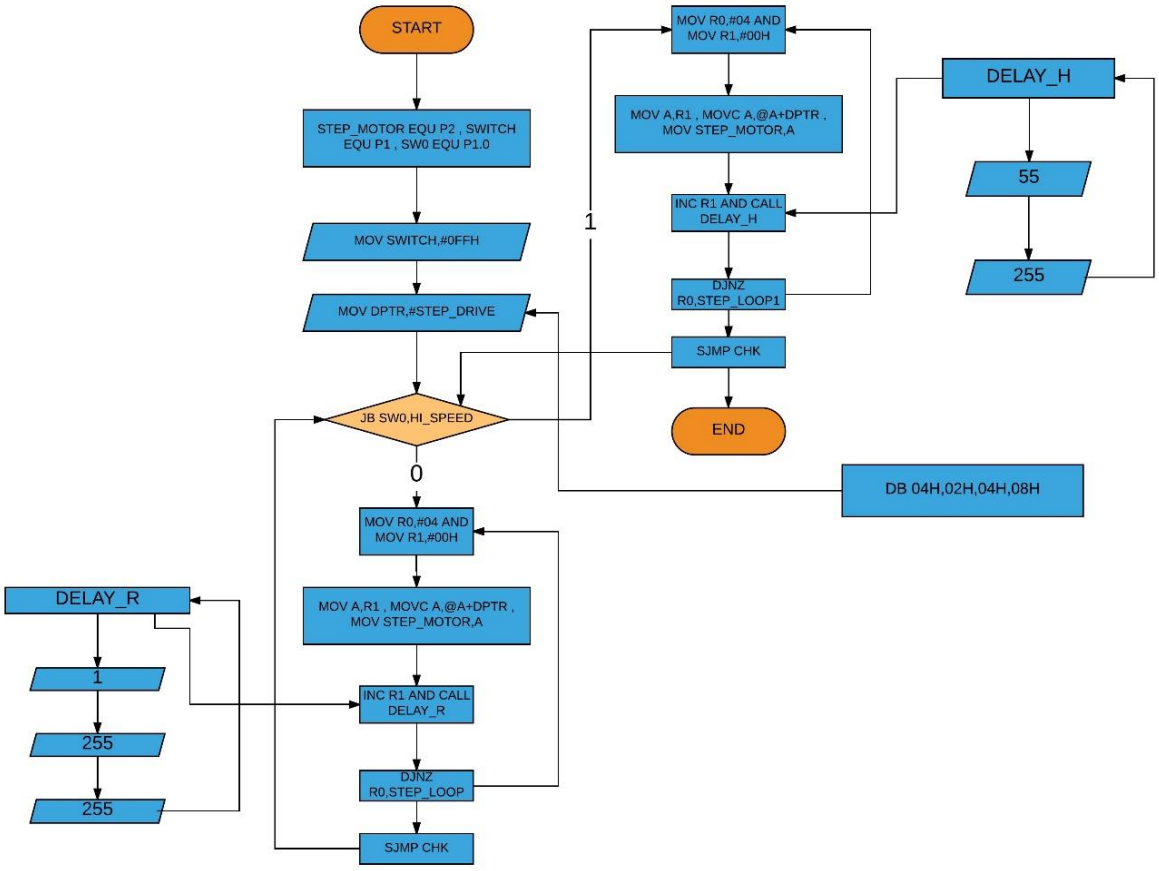
วิธีการทดลอง

1. เขียนโค้ดที่โจทย์กำหนดให้ จากนั้นทำการแปลงไฟล์ให้ได้ไฟล์นามสกุล .hex

2. ทำการรัน ASM Code จากโปรแกรม Flash Magic เพื่อดาวน์โหลดโค้ดลงบอร์ดไมโครคอนโทรลเลอร์ผ่านสายสื่อข้อมูล

3. สังเกตผลการทดลองและบันทึกผลการทดลอง

Flow Chart :



ORG 0000H	เริ่มต้นการทำงานที่ตำแหน่ง 0000H
STEP_MOTOR EQU P2	ประกาศตัวแปร STEP_MOTOR สำหรับเรียกแทนพอร์ต 2
SWITCH EQU P1	ประกาศตัวแปร SWITCH สำหรับเรียกแทนพอร์ต 1
SW0 EQU P1.0	ประกาศตัวแปร SW0 สำหรับเรียกแทนพอร์ต 1 บิต 0
MOV SWITCH, #0FFH	ให้ SWITCH = 0FFH
MOV DPTR, #STEP_DRIVE	นำ DPTR ชี้ไปที่ STEP_DRIVE
CHK: JB SW0, HI_SPEED	ถ้า SW0 ถูกกด ให้ไปที่ตำแหน่ง HI_SPEED
REGULAR_SPEED: MOV R0, #04	เก็บค่า 04 ไว้ใน รีจิสเตอร์ R0
MOV R1, #00H	กำหนดให้ รีจิสเตอร์ R1 = 0
STEP_LOOP0: MOV A, R1	ย้ายค่าใน R1 ไปเป็นข้อมูลใน A
MOVC A, @A+DPTR	นำค่าในตาราง DATA มาไว้ที่รีจิสเตอร์ A
MOV STEP_MOTOR, A	ย้ายค่าในรีจิสเตอร์ A ไปเป็นข้อมูลใน STEP_MOTOR
INC R1	เพิ่มค่า R1 ไป 1 ค่า
CALL DELAY_R	เรียกใช้โปรแกรมย่อยหน่วงเวลา R
DJNZ R0, STEP_LOOP0	ลดค่า R0 ลง 1 ค่า ถ้าไม่เท่ากับ 0 ให้กระโดดไปที่ตำแหน่ง STEP_LOOP0
SJMP CHK	กระโดดไปทำงานที่ตำแหน่ง CHK
STEP_LOOP1: MOV A, R1	ย้ายค่าใน R1 ไปเป็นข้อมูลใน A
MOVC A, @A+DPTR	นำค่าในตาราง DATA มาไว้ที่รีจิสเตอร์ A
MOV STEP_MOTOR, A	ย้ายค่าในรีจิสเตอร์ A ไปเป็นข้อมูลใน STEP_MOTOR
INC R1	เพิ่มค่า R1 ไป 1 ค่า
CALL DELAY_H	เรียกใช้โปรแกรมย่อยหน่วงเวลา H
DJNZ R0, STEP_LOOP1	ลดค่า R0 ลง 1 ค่า ถ้าไม่เท่ากับ 0 ให้กระโดดไปที่ตำแหน่ง STEP_LOOP1
SJMP CHK	กระโดดไปทำงานที่ตำแหน่ง CHK
DELAY_R: MOV R5, #1	โปรแกรมย่อยเพื่อหน่วงเวลา เก็บค่า 1 ไว้ใน รีจิสเตอร์ R5
DELAY_R0: MOV R6, #255	เก็บค่า 255 ไว้ใน รีจิสเตอร์ R6
DELAY_R1: MOV R7, #255	เก็บค่า 255 ไว้ใน รีจิสเตอร์ R7
DELAY_R2: DJNZ R7, #DELAY_R2	ลดค่า R7 ลง 1 ค่า ถ้าไม่เท่ากับ 0 ให้กระโดดไปที่ตำแหน่ง DELAY_R2
DJNZ R6, #DELAY_R1	ลดค่า R6 ลง 1 ค่า ถ้าไม่เท่ากับ 0 ให้กระโดดไปที่ตำแหน่ง DELAY_R1

DJNZ R5, #DELAY_R0	ลดค่า R5 ลง 1 ค่า ถ้าไม่เท่ากับ 0 ให้กระโดดไปที่ตำแหน่ง DELAY_R0
RET	สิ้นสุดโปรแกรมหน่วงเวลา
DELAY_H: MOV R6, #55	เก็บค่า 55 ไว้ใน รีจิสเตอร์ R6
DELAY_H0: MOV R7, #255	เก็บค่า 255 ไว้ใน รีจิสเตอร์ R7
DELAY_H1: DJNZ R7, #DELAY_H1	ลดค่า R7 ลง 1 ค่า ถ้าไม่เท่ากับ 0 ให้กระโดดไปที่ตำแหน่ง DELAY_H1
DJNZ R6, #DELAY_H0	ลดค่า R6 ลง 1 ค่า ถ้าไม่เท่ากับ 0 ให้กระโดดไปที่ตำแหน่ง DELAY_H0
RET	สิ้นสุดโปรแกรมหน่วงเวลา
END	ตำแหน่งสิ้นสุดโปรแกรม

Results and discussion:

มอเตอร์หมุนทวนเข็มนาฬิกา เมื่อ SW0 = 1 มอเตอร์จะทวนเข็มนาฬิกาแต่ความเร็วจะเพิ่มขึ้น และเมื่อ SW0 = 0 มอเตอร์จะทวนเข็มนาฬิกาความเร็วจะกลับเป็นปกติ

EXERCISE

1. ให้นิสิตดัดแปลงโปรแกรมที่ LAB5-1 เพื่อกลับทิศทางการหมุนและเปลี่ยนแปลงความเร็วในการหมุนของสเต็ปมอเตอร์(ให้เร็วขึ้นหรือช้าลง)

```
                ORG     0000H
                STEP_MOTOR  EQU     P2
                MOV     DPTR, #STEP_DRIVE

START:          MOV     R0, #04
                MOV     R1, #00H

STEP_LOOP:      MOV     A, R1
                MOVC    A, @A+DPTR
                MOV     STEP_MOTOR, A
                INC     R1
                CALL    DELAY
                DINZ    R0, STEP_LOOP
                SJMP    START

STEP_DRIVE:     DB      01H, 02H, 04H, 08H

DELAY:          MOV     R5, #1
DELAT0:         MOV     R6, #55
DELAT1:         MOV     R7, #255
DELAT2:         DJNZ    R7, #DELAY2
                DJNZ    R6, #DELAY1
                DJNZ    R5, #DELAY0
                RET
                END
```

2. จงดัดแปลงโปรแกรม LAB5-2 โดยเพิ่มการใช้งาน SW0 – SW1 ให้มีกรทำงานดังนี้

- หาก SW1 = 0 และ SW0 = 0 ให้มอเตอร์หมุนทวนเข็มนาฬิกาที่ความเร็วปกติ
- หาก SW1 = 0 และ SW0 = 1 ให้มอเตอร์หมุนทวนเข็มนาฬิกาที่ความเร็วปสูง
- หาก SW1 = 1 และ SW0 = 0 ให้มอเตอร์หมุนตามเข็มนาฬิกาที่ความเร็วปกติ
- หาก SW1 = 1 และ SW0 = 1 ให้มอเตอร์หมุนทวนเข็มนาฬิกาที่ความเร็วสูง

```
                ORG 0000H
                STEP_MOTOR  EQU P2
                SWITCH      EQU P1
                SW0         EQU P1.0
                SW1         EQU P1.1

                MOV    SWITCH, #0FFH
CHK:            JB SW0, CHK_2
                JB SW1, CW_NORMAL

                MOV    DPTR, #STEP_DRIVE2
CCW_NORMAL:     MOV R0, #04
                MOV    R1, #00H
STEP_LOOP0:     MOV    A, R1
                MOVC   A, @A+DPTR
                MOV    STEP_MOTOR, A
                INC R1
                LCALL  DELAY_R
                DJNZ   R0, STEP_LOOP0
                SJMP   CHK

CHK_2:          JB     SW1, CW_HIGH
                MOV    DPTR, #STEP_DRIVE2
CCW_HIGH:       MOV    R0, #04
                MOV    R1, #00H
STEP_LOOP1:     MOV    A, R1
                MOVC   A, @A+DPTR
                MOV    STEP_MOTOR, A
                INC    R1
                LCALL  SPEED_H
                DJNZ   R0, STEP_LOOP1
                SJMP   CHK
```

```
CW_HIGH:      MOV   DPTR, #STEP_DRIVE
              MOV    R0, #04
              MOV   R1, #00H
STEP_LOOP3:   MOV   A, R1
              MOVC  A, @A+DPTR
              MOV   STEP_MOTOR, A
              INC   R1
              LCALL SPEED_H
              DJNZ  R0,      STEP_LOOP3
              SJMP  CHK

CW_NORMAL:    MOV   DPTR, #STEP_DRIVE
              MOV   R0, #04
              MOV   R1, #00H
STEP_LOOP2:   MOV   A, R1
              MOVC   A, @A+DPTR
              MOV   STEP_MOTOR, A
              INC   R1
              LCALL DELAY_R
              DJNZ  R0, STEP_LOOP2
              SJMP  CHK

STEP_DRIVE2:  DB 01H,08H,04H,02H
STEP_DRIVE:   DB 01H,02H,04H,08H

DELAY_R:      MOV   R5, #1
DELAY_R0:     MOV   R6, #255
DELAY_R1:     MOV   R7, #255
DELAY_R2:     DJNZ  R7, DELAY_R2
              DJNZ  R6, DELAY_R1
              DJNZ  R5, DELAY_R0
              RET
```

```
SPEED_H:    MOV   R6, #55
SPEED_H0:   MOV   R7, #255
SPEED_H1:   DJNZ  R7, SPEED_H1
            DJNZ  R6, SPEED_H0
            RET
            END
```

วิเคราะห์ผลการทดลอง

จากการทดลอง เราสามารถเปลี่ยนทิศทางการหมุนของมอเตอร์ ที่ตารางโดยสลับค่าของตัวเลข และเพิ่มความเร็วที่โปรแกรมย่อยหน่วงเวลา

จาก LAB5-2 เมื่อยังไม่กดสวิตช์มอเตอร์มีความเร็วปกติ แต่เมื่อกดสวิตช์ SW0 = 1 จะทำให้มอเตอร์หมุนเร็วขึ้น และเมื่อ SW0 = 0 มอเตอร์ก็จะมีความเร็วปกติเท่ากับตอนที่ยังไม่กดสวิตช์

เนื่องจากกระแสของไมโครคอนโทลเลอร์ ไม่พอสำหรับการหมุนมอเตอร์ได้ จึงต่อเข้ากับบอร์ด ANALAB เพื่อให้มีกำลังขับมอเตอร์ที่เพียงพอ

สรุปผลการทดลอง

สเต็ปมอเตอร์ สามารถทำให้เราควบคุมความเร็ว จำนวนรอบ และทิศทางของการหมุนของมอเตอร์ได้

เอกสารอ้างอิง

พนัส นัถฤทธิ์. (2560). **ไมโครคอนโทรลเลอร์และการประยุกต์ใช้งานควบคุมหุ่นยนต์**. พิมพ์ครั้งที่ 1. พิษณุโลก: รัตนสุวรรณการพิมพ์ 3

http://www.mind-tek.net/8051.php

http://www.ett.co.th/article/Robot/et_robot_rd2/rd2_001.html

http://www.olearning.siam.edu/2011-11-28-08-10-01/593-100-101-

http://www.thaiall.com/assembly/asmcommand.htm