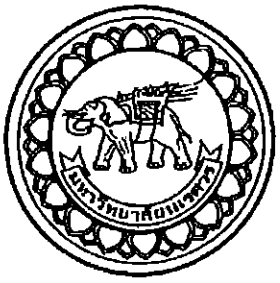


ระบบ เพื่อการแสดงผลและควบคุม การถ่วงทุ่นเครื่องกำเนิดไฟฟ้าของรถยนต์ โดยใช้โปรแกรมแมทแลบ

Interfacing to Roter Balancing Machine by Matlab

นายสังวรณ์ สีสุทัศน์ รหัส 47364138

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต
สาขาวิชาวิศวกรรมไฟฟ้า ภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร
ปีการศึกษา 2550

ใบรับรองโครงงานวิศวกรรม

หัวข้อโครงงาน	ระบบ เพื่อการแสดงผลและควบคุม การถ่วงทุ่นเครื่องกำเนิดไฟฟ้าของรถยนต์โดยใช้โปรแกรมแมทแลบ
ผู้ดำเนินโครงงาน	นายสังวรณ์ สีสุทัศน์ รหัส 47364138
อาจารย์ที่ปรึกษา	ดร.อัครพันธ์ วงศ์กังแห
สาขาวิชา	วิศวกรรมไฟฟ้า
ภาควิชา	วิศวกรรมไฟฟ้าและคอมพิวเตอร์
ปีการศึกษา	2550

.....

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร อนุมัติให้โครงงานฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมไฟฟ้า

คณะกรรมการสอบโครงงานวิศวกรรม

.....ประธานกรรมการ
(ดร.อัครพันธ์ วงศ์กังแห)

.....กรรมการ
(ดร.ชัยรัตน์ พินทอง)

.....กรรมการ
(อาจารย์ ปิยดนัย ภาชนะพรรณ์)

หัวข้อโครงงาน	ระบบ เพื่อการแสดงผลและควบคุม การถ่วงทุ่นเครื่องกำเนิดไฟฟ้าของรถยนต์โดยใช้โปรแกรมแมทแลบ
ผู้ดำเนินโครงงาน	นายสังวรณ์ สีสุทัศน์ รหัส 47364138
อาจารย์ที่ปรึกษา	ดร.อัครพันธ์ วงศ์กังแห
สาขาวิชา	วิศวกรรมไฟฟ้า
ภาควิชา	วิศวกรรมไฟฟ้าและคอมพิวเตอร์
ปีการศึกษา	2550

บทคัดย่อ

โครงงานนี้จะเป็นการใช้คอมพิวเตอร์ติดต่อกับไมโครคอนโทรลเลอร์ ในการออกแบบจะใช้การเขียนโปรแกรม MATLAB เพื่อควบคุมไมโครคอนโทรลเลอร์ในการติดต่อผ่านพอร์ท COM หรือ RS-232 เพื่อที่จะแลกเปลี่ยนข้อมูลกันระหว่างไมโครคอนโทรลเลอร์กับโปรแกรม MATLAB โดยจะใช้โปรแกรม MATLAB แสดงผลข้อมูลที่ไมโครคอนโทรลเลอร์ได้ประมวลผลจากเซ็นเซอร์ส่งมาผ่านทาง RS-232 ออกทางจอภาพ

ผลที่ได้จากโครงงานนี้คือ โปรแกรมแสดงผลข้อมูลที่ได้รับมาจากไมโครคอนโทรลเลอร์ และโปรแกรมส่งสัญญาณควบคุมไมโครคอนโทรลเลอร์ เพื่อให้ไมโครคอนโทรลเลอร์นั้นไปควบคุมเครื่องถ่วงสมดุลให้ทำงานตามที่ผู้ใช้งานต้องการ

Project Title	Interfacing to Roter Balancing Machine by Matlab
Name	Mr. Sungvorn Seesutus ID. 47364138
Project Advisor	Dr. Akaraphunt Vongkunghae
Major	Electrical Engineering
Department	Electrical and Computer Engineering
Academic Year	2007

ABSTRACT

This Project use computer interfacing to control the programming of microcontroller control use the MATLAB 7.0 program to interface by port com or RS.232 for interchange the information between MATLAB program and Microcontroller. This for control balance machine and process data from sensor by using microcontroller send the information to the MATLAB 7.0 program for show data to monitor.

The result of this project is the program data show to receiver form microcontroller and send control signal to microcontroller

กิตติกรรมประกาศ

การทำโครงงานฉบับนี้ สำเร็จลุล่วงได้ด้วยดีเพราะได้รับความกรุณาจาก ดร.อัครพันธ์ วงศ์กังแห อาจารย์ที่ปรึกษา และคณะกรรมการทุกท่าน ที่ได้กรุณาให้คำแนะนำปรึกษา ตลอดจนผู้ที่ได้ช่วยตรวจสอบแก้ไข และให้ความรู้แก่ผู้ดำเนินโครงงานจนโครงงานนี้เสร็จสมบูรณ์ และผู้ดำเนินโครงงาน ขอกราบขอบพระคุณเป็นอย่างสูงไว้ ณ ที่นี้

ขอกราบขอบพระคุณ คณาจารย์ผู้สอน ประจำภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์ทุกท่านที่ให้ความรู้ แนะแนวทางการทำงานอย่างเป็นระบบ ซึ่งช่วยในการแก้ปัญหาที่เกิดขึ้นได้เป็นอย่างดี

ผลการทำโครงงานฉบับนี้ คงเป็นประโยชน์กับผู้ที่สนใจศึกษา หากมีความผิดพลาดประการใด ผู้ศึกษาขออภัยมา ณ ที่นี้

ผู้จัดทำ

นายสังวรณ์ สีสุทัศน์

สารบัญ

หน้า

สารบัญ(ต่อ)

หน้า

สารบัญรูป

รูปที่ หน้า

สารบัญรูป(ต่อ)

รูปที่ หน้า

สารบัญตาราง

ตารางที่ หน้า

บทที่ 1

บทนำ

ในบทนี้จะเป็นการแนะนำส่วนประกอบการทำงานอย่างคร่าวๆของโครงงานนี้ ซึ่งประกอบไปด้วยที่มาและความสำคัญของโครงงาน วัตถุประสงค์ในการทำโครงงาน ขอบเขตของโครงงานที่ทำ ระยะเวลาในการดำเนินงานของแต่ละขั้นตอน ผลที่คาดว่าจะได้รับจากโครงงาน และการใช้งบประมาณในการดำเนินงาน ซึ่งจะอธิบายได้ดังนี้

1.1 ที่มาและความสำคัญของโครงงาน

มีปัญหาหลากหลายปัญหาที่ผู้ประกอบการส่วนใหญ่พบในการผลิตไดชาร์จ ซึ่งเราจะนำเสนอปัญหาหนึ่งที่มีความสำคัญต่อผลิตภัณฑ์ที่ได้นั้นก็คือ ปัญหาเรื่องของทุ่นไดชาร์จที่บางครั้งในกระบวนการผลิตอาจจะไม่สมบูรณ์แบบ ทำให้เวลานำไปประกอบเป็นไดชาร์จอาจเกิดผลจากแรงเหวี่ยงของตัวทุ่นไดชาร์จที่ไม่สมดุล เกิดการเหวี่ยงหนีศูนย์มากเกินกว่าที่อุปกรณ์ประกับทุ่นจะทนได้ ซึ่งอุปกรณ์ประกับหล่อลื่นจะเป็นลูกปืนหรือแบบหมอนรอง Bush ทำให้ไดชาร์จเกิดความเสียหายทำงานผิดพลาด และส่งผลต่ออายุการใช้งานของไดชาร์จสั้นลง

ดั้นนั้นผู้ประกอบการส่วนใหญ่ จึงพยายามคิดหาวิธีการในการที่จะแก้ปัญหาในเรื่องนี้ แต่เดิมใช้แรงดึงดูดของโลกในการถ่วงสมดุล โดยใช้ทุ่นไดชาร์จหมุนอยู่บนแกนลูกกลิ้งดังรูป

รูปที่ 1.1 เครื่องถ่วงสมดุลของทุ่นไดชาร์จแบบดั่งเดิม

แล้วใช้สายตาในการดู ประกอบกับความรู้สึกเพื่อตัดสินใจในการเจาะถ่วงสมดุลทุ่นไดชาร์จ โดยดูว่าจุดไหนหมุนแล้วเวลาหยุดมีการหมุนกลับไปมา จุดที่ไม่สมดุลจะมีแรงโน้มถ่วงมากกว่าจุดอื่นๆ ทำให้จุดนั้นหมุนกลับไปอยู่ด้านล่าง เราก็จะทำการเจาะจุดนั้นเพื่อที่จะถ่วงทุ่นไดชาร์จให้ได้สมดุลมากที่สุดเท่าที่จะทำได้ แต่ก็ไม่สามารถเจาะถ่วงทุ่งไดชาร์จได้ดีเท่าที่ควร

เราจึงคิดหาวิธีการในการทดสอบและเจาะถ่วงทุ่นไดชาร์จขึ้น โดยใช้หลักการทำงานง่ายๆ คือ ใช้แรงเหวี่ยงหนีศูนย์กลางของวัตถุที่กำลังหมุนด้วยความเร็ว ถ้าทุ่นไดชาร์จมีความสมดุลแรงเหวี่ยงหนีศูนย์กลางจะต้องมีค่าน้อยมาก ซึ่งจะส่งผลต่อตัวของทุ่นไดชาร์จทำให้โมเมนตัมรอบตัวมีผลรวมของแรงเหวี่ยงมีค่าเป็นศูนย์ด้วย

โครงงานนี้เป็นการใช้คอมพิวเตอร์ติดต่อสื่อสารผ่านสายสัญญาณ RS-232 เพื่อทำการรับส่งข้อมูลและประมวลผลข้อมูลที่ได้รับจากไมโครคอนโทรลเลอร์ที่ควบคุมเครื่องถ่วงสมดุล ซึ่งจะแสดงผลของข้อมูลในรูปแบบที่ผู้ใช้งานสามารถนำไปตัดสินใจได้ทันที โดยใช้โปรแกรม MATLAB เพื่อทำการประมวลผลและทำการแสดงผลลัพธ์ให้กับผู้ใช้งาน

1.2 วัตถุประสงค์

1. รู้จักการเขียนโปรแกรมคอมพิวเตอร์ประยุกต์ใช้งานติดต่อสื่อสารกับอุปกรณ์ภายนอก เพื่อรับ-ส่งข้อมูล และทำการควบคุมอุปกรณ์ภายนอก
2. เรียนรู้การเขียนอัลกอลิทึมของโปรแกรมคอมพิวเตอร์ เพื่อการประยุกต์ใช้งาน
3. นำความรู้ที่ได้มาจากการเรียน มาประยุกต์ใช้ในการทำงาน
4. เรียนรู้การโปรแกรม MATLAB 7.0 ในการนำมาประยุกต์ใช้งานกับการรับ-ส่งข้อมูล ผ่านสายสัญญาณ RS-232
5. เพื่อใช้คอมพิวเตอร์ช่วยให้ทำงานได้สะดวกสบายมากขึ้น
6. เรียนรู้และทำความเข้าใจกับการทำงานของไมโครคอนโทรลเลอร์และไมโครคอมพิวเตอร์
7. ศึกษากลไกการทำงานของเครื่องเจาะถ่วงทุ่นไดชาร์จ
8. ฝึกการใช้ปัญญาในการแก้ไขปัญหาที่พบในการเขียนโปรแกรม

1.3 ขอบข่ายโครงงาน

1. เขียนโปรแกรม MATLAB ในการติดต่อกับไมโครคอนโทรลเลอร์เพื่อทำการรับ-ส่งข้อมูลผ่านทางพอร์ทอนุกรม (Serial port) ผ่านสายสัญญาณ RS-232
2. นำข้อมูลที่ได้แสดงผลเป็นกราฟออกทางจอภาพ เพื่อให้ผู้ใช้งานสามารถนำข้อมูลไปใช้งานได้สะดวกสบายยิ่งขึ้น
3. สามารถส่งข้อมูลออกมาทางจอภาพบอกตำแหน่งที่จะทำการเจาะ และค่าน้ำหนักของแต่ละตำแหน่ง
4. ทดสอบโปรแกรมโดยการเจาะถ่วงสมดุลของตัวทุ่นไดชาร์จ เพื่อตรวจสอบชุด Sensor ว่ามีความถูกต้องตามที่ได้โปรแกรมไว้
5. จัดทำคู่มือการใช้งานเครื่องเจาะถ่วงทุ่นไดชาร์จ เพื่อให้ผู้ใช้งานสามารถใช้งานได้ง่ายขึ้น

1.4 ขั้นตอนการดำเนินงาน

ตารางที่ 1.1 แสดงขั้นตอนการดำเนินงานของโครงงานของปี 2550

กิจกรรม	มิ.ย.	ก.ค.	ส.ค.	ก.ย.	ต.ค.	พ.ย.	ธ.ค.	ม.ค.
1. ทำการรวบรวมข้อมูล ที่เกี่ยวข้องกับการทำโครงงาน	←	→						
2. ศึกษาทฤษฎีที่ใช้ในการทำโครงงาน		←	→					
3. ศึกษาวงจรต่างๆ ของไมโครคอนโทรลเลอร์			←	→				
4. ศึกษา Code ภาษาซี โปรแกรมไมโครคอนโทรลเลอร์			←	→				
5. ศึกษาโปรแกรม MATLAB				←	→			

กิจกรรม	มิ.ย.	ก.ค.	ส.ค.	ก.ย.	ต.ค.	พ.ย.	ธ.ค.	ม.ค.
6. ทำการเขียนโปรแกรม MATLAB ในการรับ-ส่งข้อมูลผ่านสายสัญญาณ RS-232				←	→			
7. เขียนโปรแกรม MATLAB แสดงกราฟ					← →			
8. ทดสอบเจาะถ่วงตัวทุ่นไดชาร์จ						← →		
9. สรุปผลการทดลอง							← →	
10. จัดทำคู่มือการใช้งานโปรแกรมทดสอบ								← →

1.5 ผลที่คาดว่าจะได้รับ

1. สามารถเขียนโปรแกรมภาษาซีใช้งานติดต่อสื่อสารกับอุปกรณ์ภายนอก **เพื่อรับ-ส่งข้อมูล** และทำการควบคุมอุปกรณ์ภายนอก
2. สามารถเขียนอัลกอลิทึมของโปรแกรมคอมพิวเตอร์ ได้อย่างคล่องแคล่ว
3. สามารถเขียนโปรแกรม MATLAB 7.0 ในการนำมาประยุกต์ใช้งานกับ**การรับ-ส่งข้อมูล** ผ่านสายสัญญาณ RS-232
4. มีความเข้าใจในการทำงานของไมโครคอนโทรลเลอร์และไมโครคอมพิวเตอร์
5. มีความเข้าใจกลไกการทำงานของเครื่องเจาะถ่วงสมดุลทุ่นไดชาร์จ
6. เพื่อใช้คอมพิวเตอร์ช่วยให้ทำงานได้สะดวกสบายมากขึ้น
7. **สามารถนำไปใช้ทดสอบการถ่วงสมดุลทุ่นไดชาร์จได้จริง**

1.6 งบประมาณที่ใช้

1. ค่าวัสดุอุปกรณ์อิเล็กทรอนิกส์	500	บาท
2. บอร์ดไมโครคอนโทรลเลอร์	1,000	บาท
3. ชุดดาวน์โหลดโปรแกรมลงบอร์ด	800	บาท
4. ค่าเอกสารและอุปกรณ์อื่นๆ	500	บาท
รวมเป็นเงินทั้งสิ้น	2,800	บาท

บทที่ 2

หลักการและทฤษฎี

ก่อนที่เราจะเริ่มทำโครงงาน เราต้องมีความรู้และความเข้าใจในสิ่งที่เราจะทำอย่างแม่นยำก่อน เพื่อให้การทำงานเกิดความผิดพลาดน้อยที่สุดและใช้ระยะเวลาน้อยลง ซึ่งจะทำให้เรามีเวลาเหลือพอที่จะทำการพัฒนาโปรแกรม แก้ไขข้อผิดพลาดที่เกิดขึ้นได้ และปรับปรุงประสิทธิภาพของโปรแกรมให้ทำงานได้รวดเร็วขึ้น ดังนั้นการศึกษาหาความรู้ในเรื่องที่เกี่ยวข้องในการทำงานจึงเป็นสิ่งที่สำคัญมากก่อนการลงมือทำงานจริง

2.1 แนวความคิดในการทำโครงงาน

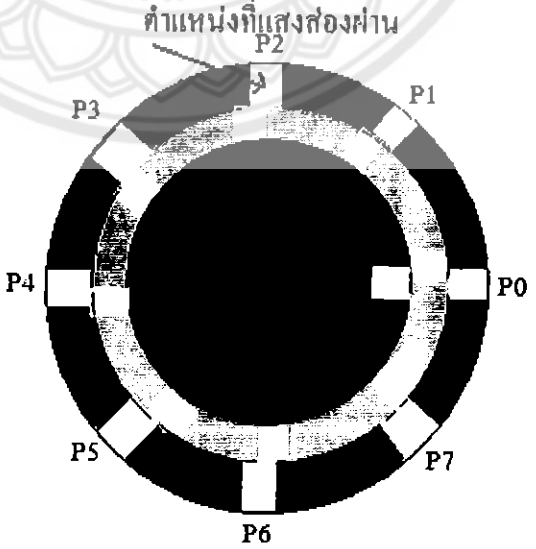
รูปที่ 2.1 ชุดแกนหมุนตัวทุ่นไดชาร์จ

เราจะใช้มอเตอร์ติดแกนหมุนตัวของทุ่นไดชาร์จ เพื่อที่จะสร้างแรงเหวี่ยงที่ตัวทุ่นไดชาร์จ แล้วทำการวัดตำแหน่งและแรงเหวี่ยงในเวลาเดียวกันขณะที่ตัวทุ่นไดชาร์จกำลังหมุนอยู่ โดยใช้ไมโครคอนโทรเลอร์ในการควบคุมการประมวลผลค่าที่อ่านได้จากเซ็นเซอร์ทั้งสองจุด โดยจะส่งข้อมูลที่ประมวลผลได้ผ่านทางสายสัญญาณ RS-232 ไปยังคอมพิวเตอร์ เพื่อที่จะให้คอมพิวเตอร์แสดงผลข้อมูลที่ได้รับออกมาเป็นกราฟให้แก่ผู้ใช้งาน ซึ่งกราฟที่ได้จะแสดงแรงเหวี่ยงที่จุดต่างๆ

พร้อมกับช่วงของตำแหน่งที่ผู้ใช้งานจะต้องเจาะออกเพื่อถ่วงสมดุลตัวของทุ่นไดชาร์จให้เกิดความสมดุล

รูป 2.2 เซนเซอร์โหลดเซลล์วัดค่าน้ำหนักและเซนเซอร์วัดตำแหน่ง

ส่วนของการวัดแรงเหวี่ยงเราจะใช้เซนเซอร์ที่เรียกว่า โหลดเซลล์ ในการวัดค่าแรงเหวี่ยงที่ได้จากการหมุนในแต่ละตำแหน่งดังรูปที่ 2.2

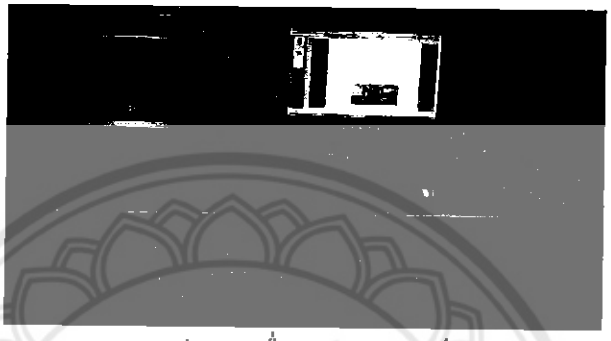


รูป 2.3 จานหมุนที่ใช้ในการวัดตำแหน่ง

ส่วนของการวัดตำแหน่งเราจะใช้เซนเซอร์อินฟาร์เรดในการตรวจจับตำแหน่ง โดยเราจะใช้การตรวจจับจากจานหมุนที่ติดอยู่กับแกนหมุน ซึ่งจะประกอบไปด้วยตำแหน่งที่เราทำการเจาะให้แสงส่องผ่านจำนวน 8 ตำแหน่ง ตามรูปที่ 2.3 และตำแหน่งข้างในสุดแสดงถึงจุดเริ่มต้นของตำแหน่ง P0 แล้ววนไปจนถึง P7 ดังรูป 2.3

2.2 อุปกรณ์ที่ใช้ในการทำโครงงาน

1. คอมพิวเตอร์ 1 เครื่อง



รูป 2.4 เครื่องคอมพิวเตอร์

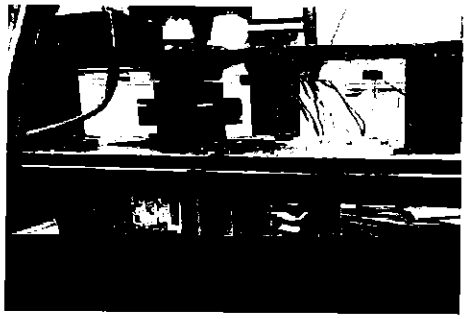
2. ชุดไมโครคอนโทรเลอร์ 1 ชุด

รูป 2.5 ชุดไมโครคอนโทรเลอร์ 1 ชุด

3. เครื่องสว่านเจาะถ่วงทุ่นไดชาร์จ 1 เครื่อง

รูป 2.6 เครื่องสว่านเจาะถ่วงทุ่นไดชาร์จ 1 เครื่อง

4. ชุดมอเตอร์ติดแกนหมุนตัวทุ่นไดชาร์จ 1 ชุด



รูป 2.7 ชุดมอเตอร์ติดแกนหมุนตัวทุ่นไดชาร์จ 1 ชุด

5. เซนเซอร์ประกอบด้วย โหลดเซลล์วัดค่าน้ำหนักและเซนเซอร์วัดตำแหน่ง

รูป 2.8 เซนเซอร์โหลดเซลล์วัดค่าน้ำหนักและเซนเซอร์วัดตำแหน่ง

2.3 ทฤษฎี

2.3.1 ภาษาสั่งงานคอมพิวเตอร์

คือชุดคำสั่งที่เขียนขึ้นตามรูปแบบและโครงสร้างของภาษา เพื่อสั่งงานคอมพิวเตอร์ทำงานตามชุดคำสั่งหรือโปรแกรมซึ่งถูกเขียนขึ้นโดยโปรแกรมเมอร์(Programmer) ภาษาสั่งงานคอมพิวเตอร์จำแนกได้ 3 ระดับ [จากหนังสือคู่มือการใช้งาน MATLAB ฉบับสมบูรณ์]

1. ภาษาระดับต่ำ (Low Level Language)

เป็นภาษาที่คอมพิวเตอร์สามารถเข้าใจคำสั่งได้ง่ายแต่มนุษย์เข้าใจได้ยาก ใช้เวลาในการศึกษาเขียนโปรแกรมนานและต้องเข้าใจหลักการทำงานของฮาร์ดแวร์ ภาษาระดับต่ำสามารถติดต่อกับทางฮาร์ดแวร์ได้ดีทำให้ทำงานได้รวดเร็ว ซึ่งภาษาระดับต่ำมีอยู่ 2 ภาษาคือ

A. ภาษาเครื่อง(Machine Language) เป็นชุดคำสั่งที่อยู่ในรูปแบบเลขฐานสองติดต่อกับฮาร์ดแวร์ได้โดยตรง คอมพิวเตอร์สามารถเข้าใจคำสั่งได้ทันทีโดยไม่ต้องใช้ตัวแปลภาษาทำให้คอมพิวเตอร์สามารถทำงานได้รวดเร็ว แต่มนุษย์เข้าใจยากและใช้เวลาในการเขียนโปรแกรมมาก

B . ภาษาแอสแซมบลี (Assembly Language) เป็นภาษาที่อยู่ในรูปแบบชุดคำสั่งสั้นๆ มนุษย์เข้าใจได้ง่ายกว่าภาษาเครื่อง แต่ก็ยังเข้าใจได้ยากกว่าภาษาระดับกลางและภาษาระดับสูง ภาษาแอส

แซมบลีสามารถทำงานได้รวดเร็ว การติดต่อทางฮาร์ดแวร์ทำได้ดี และการสั่งงานให้คอมพิวเตอร์ทำงานต้องมีการแปลความหมายให้เป็นภาษาเครื่องก่อนโดยใช้ตัวแปลภาษาที่เรียกว่าแอสแซมเบลอร์(Assembler)

2. ภาษาระดับกลาง (Medium Level Language)

เป็นภาษาที่มีลักษณะผสมกันระหว่างภาษาระดับสูงกับภาษาระดับต่ำ คือมีลักษณะของคำสั่งคล้ายกับประโยคทางภาษาอังกฤษ และยังมีบางคำสั่งไปคล้ายกับภาษาระดับต่ำ ซึ่งสามารถทำงานได้รวดเร็วและใช้เวลาในการศึกษาเขียนโปรแกรมน้อยกว่าภาษาระดับต่ำ การสั่งงานให้คอมพิวเตอร์ทำงานต้องมีการแปลความหมายให้เป็นภาษาเครื่องก่อนโดยใช้ตัวแปลภาษาที่เรียกว่าคอมไพเลอร์(Compiler) ตัวอย่างของภาษาระดับกลางอย่างเช่น ภาษาซี เป็นต้น

3. ภาษาระดับสูง (High Level Language)

เป็นภาษาที่สามารถศึกษาและเข้าใจได้ง่าย มีลักษณะของคำสั่งคล้ายกับประโยคทางภาษาอังกฤษซึ่งง่ายต่อการทำความเข้าใจและใช้เวลาในการเขียนโปรแกรมน้อย แต่การสั่งงานให้คอมพิวเตอร์ทำงานช้ากว่าและสั่งงานได้เพียงบางส่วนของคอมพิวเตอร์เท่านั้น การสั่งงานให้คอมพิวเตอร์ทำงานต้องมีการแปลความหมายให้เป็นภาษาเครื่องก่อนโดยใช้ตัวแปลภาษาที่เรียกโดยทั่วไปว่า อินเทอร์พลีเตอร์(Interpreter) หรือคอมไพเลอร์ (Compiler)

2.3.2 MATLAB คืออะไร

MATLAB เป็นภาษาคอมพิวเตอร์ขั้นสูง (High Level Language) ใช้สำหรับการคำนวณทางคณิตศาสตร์ที่ประกอบไปด้วยการคำนวณเชิงตัวเลข กราฟิกที่ซับซ้อน และการจำลองแบบเพื่อให้มองเห็นภาพพจน์ได้ง่ายและชัดเจน ชื่อของ MATLAB ย่อมาจาก matrix laboratory เดิมโปรแกรม MATLAB ได้เขียนขึ้นเพื่อใช้คำนวณทาง matrix หรือเป็น matrix software ที่พัฒนาจากโปรเจ็คที่ชื่อ LINKPACK และ EISPACK

MATLAB ได้พัฒนามาจากปัญหาที่ส่งมาจากหลายๆผู้ใช้งาน จึงทำให้ MATLAB มีฟังก์ชันต่างๆให้เลือกใช้อยู่มากมาย ในบางมหาวิทยาลัยได้ใช้โปรแกรม MATLAB เป็นหลักสูตรพื้นฐานในการศึกษาทางด้านคณิตศาสตร์ วิศวกรรม และวิทยาศาสตร์แขนงต่างๆ ตลอดจนในด้านอุตสาหกรรมได้ใช้โปรแกรม MATLAB เป็นเครื่องมือสำหรับใช้ในงานวิจัย พัฒนาและวิเคราะห์

โปรแกรม MATLAB จะมีกล่องเครื่องมือที่ใช้ในการหาคำตอบเรียกว่า Toolbox โดยโปรแกรม MATLAB จะมี Toolbox ในแต่ละสาขา เช่น การประมวลผลสัญญาณ(signal processing toolbox) การประมวลผลภาพ (image processing toolbox) ระบบควบคุม (control system toolbox) โครงข่ายประสาท (neural networks toolbox) สถิติ (statistics toolbox) ฟัซซี่ลอจิก (fuzzy logic toolbox) เวฟเลท (wavelet toolbox) การติดต่อสื่อสาร (communication

toolbox) และสาขาอื่นๆมากมาย ภายใน toolbox แต่ละสาขาจะมีฟังก์ชันต่างๆ ที่เกี่ยวข่องกับการแก้ปัญหาในสาขานั้นๆ ให้เลือกประยุกต์ใช้งานเป็นจำนวนมาก

2.3.3 โครงสร้าง MATLAB

โครงสร้างของโปรแกรม MATLAB ประกอบด้วย 5 ส่วนใหญ่ๆ คือ

1. ภาษาโปรแกรม MATLAB (The MATLAB language)

MATLAB เป็นโปรแกรมภาษาชั้นสูงที่ใช้ควบคุม flow statements ฟังก์ชัน โครงสร้างข้อมูลอินพุท/เอาท์พุท และลักษณะโปรแกรมใน MATLAB นั้นเป็นทั้งแบบ Structure Programming และ Object-Oriented Programming ทำให้การเขียนโปรแกรมมีความยืดหยุ่นสูงเมื่อเทียบกับการเขียนโปรแกรมด้วยภาษาอื่นๆ เช่น C , Fortran , Java , Basic เป็นต้น

2. สถาปัตยกรรมในการทำงานของ MATLAB (The MATLAB working environment)

MATLAB จะมีกลุ่มของเครื่องมือที่เป็นประโยชน์สำหรับการทำงานของผู้ใช้โปรแกรมหรือโปรแกรมเมอร์ ประโยชน์ที่กล่าวนี้ก็คือการจัดการตัวแปลใน Workspace การนำข้อมูลหรือการผ่านค่าตัวแปรเข้า/ออกและกลุ่มของเครื่องมือต่างๆนี้ ก็จะใช้สำหรับพัฒนา จัดการ ตรวจสอบความผิดพลาดของโปรแกรม (debugging) ที่ได้เขียนขึ้น

3. ฟังก์ชันในการคำนวณทางคณิตศาสตร์ (The MATLAB mathematical function library)

MATLAB จะมีไลบรารีทั่วไปที่ใช้ในการคำนวณอย่างกว้างขวาง เช่น sine , cosine , และพีชคณิตเชิงซ้อน โดยสามารถนำไปประยุกต์ใช้เป็นฟังก์ชันหรือไลบรารีเพิ่มเติมขึ้นจากไลบรารีที่ใช้กันอยู่โดยทั่วไป เช่น ฟังก์ชันในการหา eigenvalues และ eigenvectors การแยกตัวประกอบและส่วนประกอบของเมตริกซ์ด้วยวิธีต่างๆ การวิเคราะห์ข้อมูล การหาความน่าจะเป็น และการแก้ปัญหาของระบบสมการเชิงเส้นที่เป็นพื้นฐานของวิชาสาขาต่างๆ เป็นต้น ทำให้โปรแกรม MATLAB มีฟังก์ชันสำหรับการใช้งานเป็นจำนวนมากและครอบคลุมในรายละเอียดของการคำนวณในสาขาวิชาต่างๆ ได้มากขึ้น

4. Handle Graphics

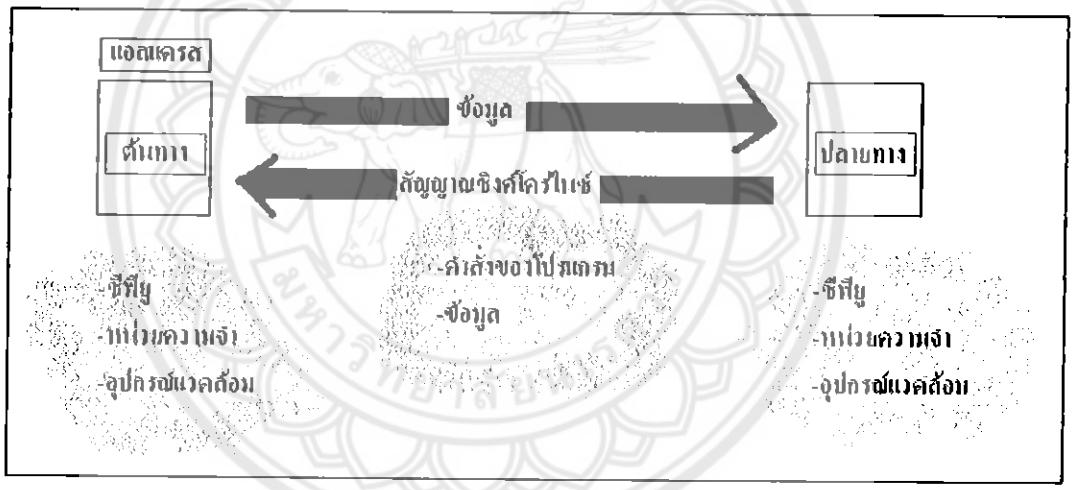
ระบบกราฟิกของ MATLAB จะประกอบไปด้วยคำสั่งชั้นสูงสำหรับการพล็อตกราฟโดยมีพื้นฐานอยู่บนแนวความคิดที่ว่าทุกๆ สิ่งบนหน้าต่างรูปภาพของโปรแกรม MATLAB จะเป็นวัตถุ (Object) ซึ่งมีเอกลักษณ์เฉพาะตัว Handle Graphics ประกอบด้วยคำสั่งชั้นสูงให้คุณได้เลือกใช้ในการสร้าง Graphics User Interface บนพื้นฐานการประยุกต์ใช้งานของคุณ นอกจากนี้โปรแกรม MATLAB ยังมีฟังก์ชันที่ใช้สำหรับการแสดงภาพสองมิติ ภาพสามมิติและการสร้างภาพเคลื่อนไหว

5. The MATLAB Application Program Interface (API)

API จะใช้เพื่อสนับสนุนการติดต่อจากอุปกรณ์ภายนอกโดยใช่โปรแกรมที่เป็น mex ไฟล์ซึ่งเป็นไฟล์ที่เขียนขึ้นโดยใช้ mex ฟังก์ชันใน MATLAB ซึ่งจะเรียกใช้รูทีนจากโปรแกรมภาษา C และ Fortran หรืออาจกล่าวได้ว่า API เป็นไลบรารีที่เขียนด้วยโปรแกรมภาษา C และ Fortran ที่มีการเชื่อมต่อกับโปรแกรม MATLAB ด้วยไฟล์ที่เป็น mex ฟังก์ชันอีกทั้ง MATLAB API นี้ยังมีความสามารถสำหรับการเรียก routine จาก MATLAB (dynamic linking) ก็ได้

2.3.4 การอินเตอร์เฟสกับคอมพิวเตอร์หรือไมโครโปรเซสเซอร์

คือ การทำงานติดต่อร่วมกันระหว่างซีพียูกับอุปกรณ์อื่นๆกับการถ่ายโอนข้อมูลระหว่างอุปกรณ์ต่างๆ นอกเหนือจะต้องทำงานติดต่อกับ แรม รอม แล้วยังต้องมีการติดต่อกับอุปกรณ์ภายนอกที่มีการส่งข้อมูล อินพุท เอาท์พุท อีกทางหนึ่ง ซึ่งเป็นการเพิ่มประสิทธิภาพในระบบต่างๆของอุปกรณ์อิเล็กทรอนิกส์ จะทำงานต่อเนื่องเป็นลูกโซ่ ดังเช่น การส่งรับข้อมูลจากซีพียูไปยังส่วนอื่นๆ เป็นต้น



รูปที่ 2.9 การทำงานติดต่อกันระหว่างซีพียูกับอุปกรณ์อื่นๆ

การที่จะโยกย้ายข้อมูลทุกตัวได้จะต้องมีแหล่งที่ส่งข้อมูล และแหล่งที่รับข้อมูลสำหรับกระบวนการเหล่านี้ จะมีส่วนที่สำคัญของข้อมูลที่ประกอบด้วย Address และ Data จะส่งไปยังจุดไหน ตัวอย่างเช่น ส่งไปยังหน่วยความจำหรืออุปกรณ์ อินพุท เอาท์พุท และจะส่งเมื่อไร การทำงานเหล่านี้โดยทั่วๆไป จะต้องมีสัญญาณในการตรวจสอบอุปกรณ์ว่าพร้อมที่จะส่ง/รับข้อมูลหรือยัง ก่อนเสมอ เนื่องจากจุดที่ส่งและรับ ข้อมูล จะต้องมีสัญญาณตรวจสอบความพร้อมเสมอเพื่อที่จะให้ข้อมูลที่ใช้งานนั้นๆเป็นระเบียบ ตัวอย่างเช่น ส่งข้อมูลจากซีพียูไปยังอุปกรณ์รอบข้าง เป็นต้น ซึ่งจุดรับส่งคู่หนึ่งๆ อาจจะเป็นระหว่างซีพียูด้วยกัน หรือซีพียูหน่วยความจำ หรือซีพียูกับอุปกรณ์รอบข้าง หรือระหว่างอุปกรณ์รอบข้างด้วยกัน หรือระหว่างหน่วยความจำกับอุปกรณ์รอบ

ข้าง ก็ได้ สำหรับข้อมูลที่โยกย้ายไปมานั้นจะอยู่ในลักษณะของเลขฐานสอง ตัวอย่างเช่น 01101100 ซึ่งเลขแต่ละตัวจะแทนด้วย 1 บิต อาจเป็น 8 บิต หรือ 16 บิต ก็ขึ้นอยู่กับของระบบนั้นๆ ถ้าหากเป็นการต่อจากพอร์ทพีซีไม่ว่าจะเป็น พอร์ทอนุกรม หรือพอร์ทขนาน ในสัญญาณที่ส่งมาจะมีระบบแรงดันไฟฟ้า จะเห็นได้ว่าระดับสัญญาณแรงดันไฟฟ้าที่ยกมาให้ดูนี้ เราสามารถที่จะควบคุมและนำมาใช้กับอุปกรณ์รอบข้างหรืออุปกรณ์ภายนอกได้ ดังจะยกตัวอย่าง เช่น Parallel(Printer port) ระดับแรงดันไฟฟ้าประมาณ 5 โวลต์ สามารถนำมาใช้ในการขับรีเลย์ ทรานซิสเตอร์ หลอดไฟ 5 โวลต์ หรือแอลอีดีให้ทำงานได้ โดยการเขียนโปรแกรมคอมพิวเตอร์ไปควบคุมที่พอร์ทขนาน เป็นต้น

ดังนั้นการที่จะนำพีซีมาประยุกต์ใช้งานให้เกิดประสิทธิผลกับชีวิตประจำวันนั้นเป็นไปได้หลายวิธี อีกทั้งฮาร์ดแวร์ต่างๆของพีซีที่มีอยู่กับเครื่องสามารถใช้ให้เกิดประโยชน์ได้ในหลายๆ ด้าน เพราะฉะนั้นเราจะกล่าวถึงเฉพาะสิ่งที่เราควรต้องรู้ต่อไปในการใช้พีซีติดต่ออุปกรณ์ ก็คือการเชื่อมต่ออุปกรณ์กับพอร์ทต่อพ่วงของพีซีบางชนิดเท่านั้น และระบบการติดต่อสื่อสารของพีซี ดังมีรายละเอียดต่อไปนี้

2.3.5 การสื่อสารข้อมูลแบบอนุกรม

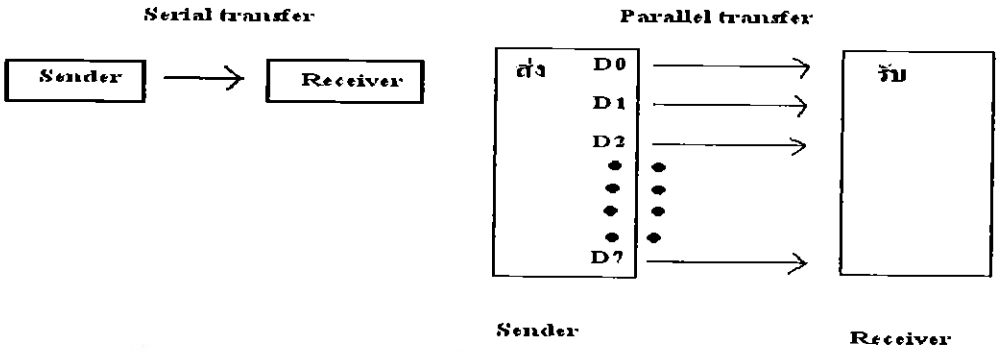
ระบบคอมพิวเตอร์สามารถส่งข้อมูลได้สองรูปแบบคือแบบขนาน (parallel) ซึ่งจะส่งข้อมูลทุกบิตออกไปพร้อมกัน และแบบอนุกรม (serial) ซึ่งจะส่งข้อมูลออกไปครั้งละบิต การส่งข้อมูลแบบขนานนี้จะต้องใช้สายในการส่งข้อมูลจำนวนมากซึ่งไม่เหมาะกับการส่งระยะไกล โดยมากแล้วจะใช้ในการส่งระยะใกล้ เช่น ระหว่างไมโครโพรเซสเซอร์กับฮาร์ดดิสก์ เครื่องพิมพ์ เป็นต้น ส่วนการส่งข้อมูลระยะไกลๆ ควรใช้การส่งข้อมูลแบบอนุกรมโดยจะส่งข้อมูลออกไปครั้งละบิต การส่งข้อมูลแบบอนุกรมนั้นจะส่งได้ช้ากว่าการส่งแบบขนานแต่ค่าใช้จ่ายจะต่ำกว่า

ในไมโครโพรเซสเซอร์ 8051 จะมีพอร์ตอนุกรมอยู่ภายในซึ่งจะรับส่งข้อมูลได้สองทิศทางในพอร์ตเดียวกัน สำหรับในหัวข้อนี้จะกล่าวถึงการสื่อสารข้อมูลของ 8051 โดยหัวข้อ 2.3.4.1 จะกล่าวถึงการสื่อสารทั่วไป ในหัวข้อ 2.3.4.6 จะกล่าวถึงการสื่อสารแบบ RS-232 โดยจะนำไอซี MAX232 มาร่วมใช้งานด้วย

1. พื้นฐานการสื่อสารอนุกรม

เมื่อไมโครโพรเซสเซอร์ต้องการติดต่อกับอุปกรณ์ภายนอก ตัวมันจะส่งข้อมูลออกมามีขนาดเป็นไบต์ หรือ 8 บิต แต่เนื่องจากไมโครโพรเซสเซอร์มีบัสข้อมูลขนาด 8 บิต การโอนถ่ายข้อมูลต่างๆ จะทำแบบขนานถ้าต้องการส่งข้อมูลออกไปแบบอนุกรมจะต้องเปลี่ยนข้อมูลแบบขนานนี้ให้เป็นข้อมูลอนุกรมเสียก่อนแล้วจึงส่งออกไป ส่วนการรับข้อมูลแบบอนุกรมนี้จะรับข้อมูลเข้ามาครั้งละบิตและเปลี่ยนข้อมูลให้เป็นข้อมูลแบบขนานเพื่อส่งให้ไมโครโพรเซสเซอร์ประมวลผลต่อไป

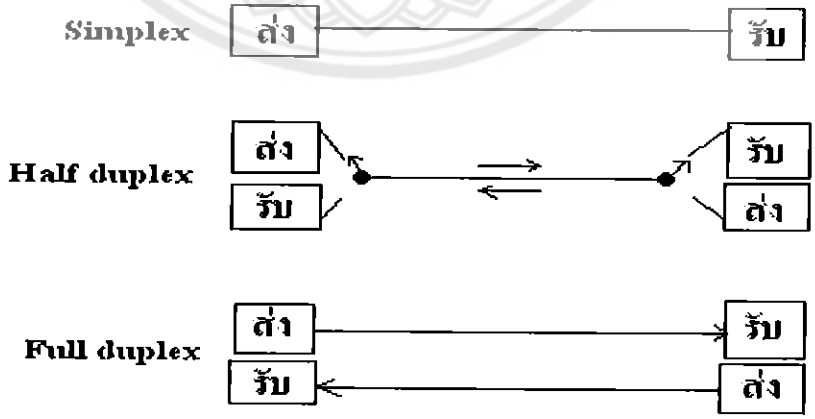
ในระบบคอมพิวเตอร์ตัวที่เปลี่ยนข้อมูลอนุกรมเป็นขนานและเปลี่ยนข้อมูลขนานเป็นอนุกรมจะใช้อุปกรณ์ที่มีชื่อว่า UART (Universal asynchronous receiver transmitter)



รูปที่ 2.10 ลักษณะการส่งข้อมูลแบบอนุกรมใช้สายสัญญาณในการส่ง 1 เส้น(ไม่รวมกราวด์) และแบบขนาน 8 บิตใช้สายสัญญาณในการส่ง 8 เส้น(ไม่รวมกราวด์)

2. รูปแบบการสื่อสารข้อมูล

การสื่อสารข้อมูลระหว่างตัวรับและตัวส่งนั้นมีหลายวิธี ถ้าหากตัวส่งทำหน้าที่ส่งอย่างเดียวและตัวรับทำหน้าที่รับอย่างเดียวจะเรียกว่าการสื่อสารแบบซิมเพล็กซ์ (simplex) เช่น การส่งข้อมูลระหว่างคอมพิวเตอร์กับเครื่องพิมพ์ แต่ถ้าหากต้วรับและตัวส่งสามารถรับและส่งข้อมูลได้แต่ทำในเวลาต่างกันเรียกว่าการสื่อสารแบบ ฮาล์ฟดูเพล็กซ์ (half duplex) เช่น การสื่อสารแบบเครื่องอินเตอร์คอมที่มีสายส่งสัญญาณได้ทางเดียว แต่ถ้าหากตัวรับและตัวส่งสามารถรับส่งข้อมูลได้สองทิศทางในเวลาเดียวกันจะเรียกว่าเป็นการสื่อสารแบบฟูลล์ดูเพล็กซ์ (full duplex) การสื่อสารแต่ละแบบแสดงได้ดังรูปที่ 2.11



รูปที่ 2.11 การรับส่งข้อมูลแบบ simplex, half duplex, full duplex

สำหรับวิธีในการส่งข้อมูลแบบอนุกรมมีสองวิธีคือ การส่งแบบเข้าจังหวะเวลา (synchronous) และการส่งแบบไม่เข้าจังหวะเวลา (asynchronous) โดยการส่งแบบเข้าจังหวะเวลาจะต้องมีการส่งสัญญาณนาฬิการ่วมไปด้วยเพื่อควบคุมการรับส่งข้อมูล แต่การรับส่งแบบไม่เข้าจังหวะเวลาจะไม่ต้องมี แต่จะใช้การกำหนดอัตราเร็วในการรับส่งข้อมูลให้มีค่าเท่ากันที่เรียกว่าอัตราบอด หรือบอดเรต (baud rate) ซึ่งมีหน่วยเป็นบิตต่อวินาที (bit per second : bps)

3. การส่งข้อมูลแบบซิงโครนัส

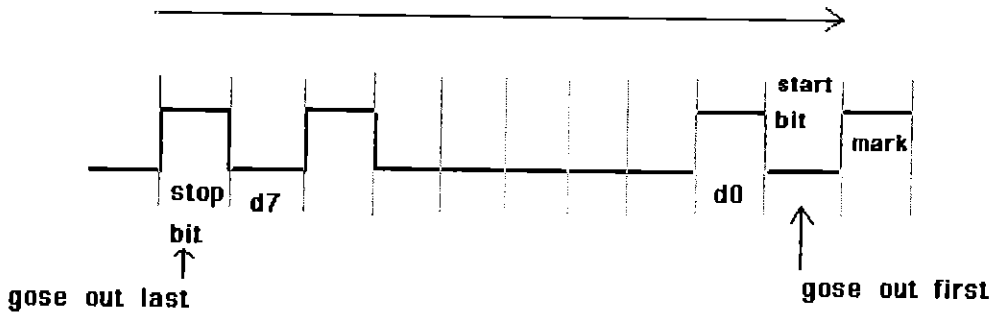
เมื่ออุปกรณ์สองตัวจะสื่อสารข้อมูลซึ่งกันและกันจะต้องกำหนดรูปแบบกฎเกณฑ์การสื่อสารซึ่งกันและกันเพื่อจะให้เข้าใจกันได้ รูปแบบการรับส่งข้อมูลนี้เรียกว่าโพรโตคอล (protocol) ซึ่งจะเป็นตัวบอกลักษณะของข้อมูล การเริ่มข้อมูล การจบข้อมูล เป็นต้น

การรับส่งข้อมูลแบบซิงโครนัสจะประกอบด้วย 4 ส่วนดังนี้

1. บิตเริ่มต้น (start bit)
2. ข้อมูลอนุกรม
3. บิตตรวจสอบความถูกต้อง
4. บิตสุดท้าย (stop bit)

4. บิตเริ่มต้นและบิตสุดท้าย

การส่งข้อมูลแบบอะซิงโครนัสนั้นจะส่งข้อมูลออกไปเป็นชุดเรียกว่า เฟรม (framing) ภายในเฟรมจะประกอบด้วยข้อมูลหรือรหัส ASCII ก็ได้ ในแต่ละเฟรมจะเริ่มต้นด้วยบิตเริ่มต้น (start bit) ที่จะบอกว่าสิ่งที่ตามมาคือข้อมูล และจะจบข้อมูลด้วยบิตสุดท้าย (stop bit) ที่จะบอกว่าข้อมูลในเฟรมนั้นๆ สิ้นสุดลงแล้วในการส่งข้อมูลแบบนี้ถ้าหากยังไม่มีการส่งข้อมูลระดับลอจิกที่สายส่งจะเป็นลอจิก "1" เรียกว่าสภาวะรอ (waiting stage) ถ้าหากมีข้อมูลส่ง สัญญาณลอจิก "0" ในช่วงเวลาหนึ่ง บิตข้อมูลบิตนี้เรียกว่าบิตเริ่มต้น จากนั้นจะตามด้วยข้อมูล ถ้าหากจะส่งรหัส ASCII ของตัว "A" จะเป็นดังรูปที่ 2.12 โดยส่งค่าข้อมูล 8 บิต (01000001) ออกไปจากนั้นจะจบด้วยบิตปิดท้ายหรือบิตหยุดที่มีภาวะเป็นลอจิก "1" โดยบิตหยุดนี้อาจเป็นลอจิก "1" ในช่วงเวลา 1 บิต, 1.5 บิต หรือ 2 บิตก็ได้ขึ้นกับการกำหนดลักษณะการส่งข้อมูล



รูปที่ 2.12 เฟรมรหัส ASCII ของตัว "A"

สำหรับข้อมูลที่รับส่งกันนั้นอาจมีขนาด 7 บิต หรือ 8 บิตก็ได้ ขึ้นอยู่กับการออกแบบระบบ และข้อมูลสามารถทำการตรวจสอบความถูกต้องได้ระดับหนึ่งเรียกว่า การตรวจสอบบิตพาริตี (parity bit) ซึ่งจะเป็นบิตข้อมูลที่จะเพิ่มเข้าไปก่อนบิตปิดท้าย โดยการตรวจสอบบิตพาริตีนี้สามารถกำหนดเป็นพาริตีคี่ (odd) หรือพาริตีคู่ (even) ก็ได้ การกำหนดบิตพาริตีคี่หมายความว่าบิตข้อมูลทั้ง 7 หรือ 8 บิต ถ้าหากรวมกับบิตพาริตีแล้วจะต้องเป็นลิจิก "1" คี่บิต ถ้าหากเป็นพาริตีคู่จะต้องรวมแล้วได้ลิจิก "1" คู่บิต ตัวอย่างเช่น ถ้าหากส่งรหัส ASCII ของตัว "A" ออกไป ซึ่งข้อมูลจะเป็น 0100 0001 ถ้ากำหนดเป็นพาริตีคู่ บิตพาริตีต้องมีลอจิกเป็น "0" แต่ถ้ากำหนดเป็นพาริตีคี่บิตพาริตีต้องเป็นลอจิก "1" การสร้างบิตพาริตีนี้จะถูกสร้างจากภาคส่งของ UART ซึ่งเราสามารถโปรแกรมได้ว่าการส่งนั้นจะเป็นแบบพาริตีคี่หรือคู่ หรือไม่มีพาริตี้

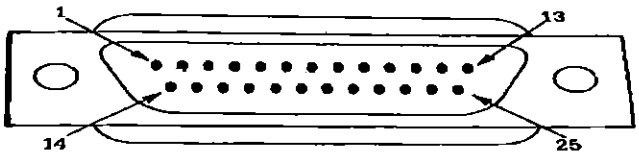
5. อัตราการรับ-ส่งข้อมูล

ความเร็วในการรับส่งข้อมูลแบบอนุกรมจะบอกเป็นจำนวนบิตต่อวินาที (bits per second : bps) ที่เรียกว่าบอดเรต (baud rates) พอร์ตอนุกรมของ MCS-51 สามารถติดต่อกับเครื่องคอมพิวเตอร์ส่วนบุคคลได้แต่ต้องกำหนดอัตราความเร็วให้เท่ากัน ในการสื่อสารทางพอร์ตอนุกรม RS-232 ของคอมพิวเตอร์ PC ได้กำหนดอัตราเร็วไว้หลายค่าตั้งแต่ 100 ถึง 9600 bps สำหรับคอมพิวเตอร์ที่ใช้ไมโครโพรเซสเซอร์เพนเทียมสามารถส่งข้อมูลได้ความเร็วสูงถึง 56 กิโลบิตต่อนาที

6. มาตรฐาน RS-232 [จากหนังสือภาษาแอสเซมบลีสำหรับ MCS-51]

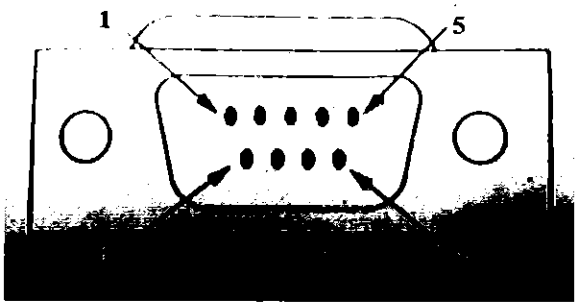
การสื่อสารแบบอนุกรมกับคอมพิวเตอร์ PC มักจะใช้รูปแบบมาตรฐาน RS 232 ซึ่งกำหนดโดย Electronics Industries Association หรือ EIA ในปี 1960 และได้มีการแก้ไขมาตรฐานในปี 1963, 1965 และ 1969 เรียกว่า RS-232A, และ RS-232C ตามลำดับ ระดับแรงดันของลอจิกที่ใช้ในการสื่อสาร RS 232 นั้นลอจิก "1" จะแทนด้วยแรงดัน -3 ถึง -25 โวลต์ ส่วนลิจิก "0" จะแทนด้วยแรงดัน +3 ถึง +25 โวลต์ แรงดันในช่วง +3 จะไม่ถูกกำหนดให้ใช้งานซึ่งจะเห็นว่าแรงดันดังกล่าวไม่สามารถใช้กับไมโครคอนโทรลเลอร์ตระกูล MCS-51 ได้ โดยทั่วไปแล้วถ้าหากต้องการให้ไมโครคอนโทรลเลอร์ติดต่อกับคอมพิวเตอร์ PC ตามมาตรฐาน RS-232 จะต้องออกแบบวงจรอิเล็กทรอนิกส์เพิ่มเติม แต่ในปัจจุบันจะใช้ไอซี MAX232 ทำหน้าที่เปลี่ยนระดับแรงดันทางลอจิกให้อยู่ในมาตรฐาน RS-232 แต่ถ้าหากจะให้ไมโครคอนโทรลเลอร์สองตัวสื่อสารกันจะไม่ใช้มาตรฐานนี้ก็ได้

ในคอมพิวเตอร์ PC จะมีขั้วต่อ RS-232 หรือที่เรียกว่า คอนเนกเตอร์ (connector) อยู่สองแบบคือ ขั้วต่อแบบ DB-25 และขั้วต่อแบบ DP-9 ขั้วต่อแบบ DB-25 และขาต่างๆ แสดงได้ดังรูปที่ 2.13 ส่วนขั้วต่อแบบ DB-9 และขาต่างๆ แสดงได้ดังรูปที่ 2.14



รูปที่ 2.13 ขั้วต่อแบบ DB-25 และหน้าที่ของขาต่างๆ[จากหนังสือภาษาแอสเซมบลีสำหรับ MCS-51]

Pin	Description
1	Protective ground
2	Transmitted dat (TxD)
3	Received data (RxD)
4	Request to send (RTS)
5	Clear to send (CTS)
6	Data set ready (DSR)
7	Signal ground (GND)
8	Data carrier detect (DCD)
9/10	Reserve for data testing
11	Unassigne d
12	Secondary data carrier detect
13	Secondary clear to send
14	Secondary transmitted data
15	Transmit signal element timing
16	Secondary received data
17	Receive signal element timing
18	Unassigned
19	Secondary request to send
20	Data terminal ready (DTR)
21	Signal quality detector
22	Ring indicator
23	Data signal rate select
24	Transmit signal element timing
25	Unassigned



Pin	Description
1	Data carrier detect (DCD)
2	Received data (RxD)
3	Transmitted data (TxD)
4	Data terminal ready (DTR)
5	Signal ground (GND)
6	Data set ready (DSR)
7	Request to send (RTS)
8	Clear tosend (CTS)
9	Ring indicator (RI)

รูปที่ 2.14 ขั้วต่อแบบ DB-9 และหน้าที่ของขาต่างๆ

ในระบบ RS-232 มีขาต่างๆ ที่สำคัญดังนี้

1. DTR (data terminal ready) เป็นสายสำหรับการทำแฮนด์เชคจาก DTE **ไปยัง** DCE
2. DSE (data set ready) เป็นสายสำหรับการทำแฮนด์เชคจาก DCE ไปยัง DTE
3. RTS (reguest to send) เมื่ออุปกรณ์ DTE ต้องการส่งข้อมูลมันจะส่งสัญญาณที่ขานี้เป็นลอจิก "Low" ออกไป
4. CTS (clear to send) เป็นสายสัญญาณสำหรับการทำแฮนด์เชคจาก DCE **ไปยัง** DTE
5. DCD (carrier detect, หรือ data carrier detect)
6. RI (ring indicator) เป็นสายที่ใช้โดยโมเด็มเพื่อบอกว่าตัวมันเองได้รับสัญญาณเรียกเข้ามา

ในคอมพิวเตอร์ส่วนบุคคล IBM ที่ใช้ไมโครโพรเซสเซอร์ตระกูล X86 **จะมีพอร์ตสื่อสารอนุกรมอยู่สองพอร์ตเรียกว่าพอร์ต** COM1 และ COM2 โดยใช้คอนเนกเตอร์แบบ DB-25 และ DB-9 **เราสามารถนำพอร์ตทั้งสองนี้มาเชื่อมต่อกับพอร์ตอนุกรมของ 8051 ได้ ดังตัวอย่างที่จะพบในโปรแกรมต่อไป**

บทที่ 3

วิธีดำเนินงาน

ในบทนี้จะกล่าวถึงวิธีการดำเนินงานของโครงงานนี้ ซึ่งสามารถสรุปเป็นขั้นตอนคร่าวๆได้ดังนี้

- ทำการศึกษารายละเอียด หลักการ ความรู้ ทฤษฎีต่างๆที่เกี่ยวข้องกับงาน
- ศึกษาวงจรอิเล็กทรอนิกส์ และการทำงานของวงจรไมโครคอนโทรลเลอร์
- การออกแบบทางด้านการเขียนโปรแกรม
- ขั้นตอนในการทดลองและทดสอบโปรแกรม

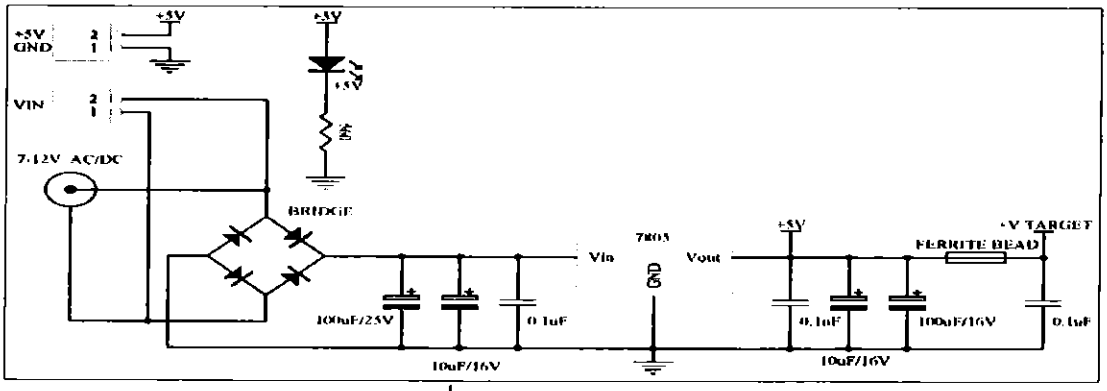
3.1 ทำการเก็บรวบรวมข้อมูลต่างๆ

การศึกษาค้นคว้า และการเก็บรวบรวมข้อมูลที่เกี่ยวข้องกับการเขียนโปรแกรมคอมพิวเตอร์เพื่อใช้ในการควบคุมติดต่อกับอุปกรณ์ฮาร์ดแวร์และอุปกรณ์ทางไฟฟ้าอิเล็กทรอนิกส์ที่ต่ออยู่ภายนอกเครื่องคอมพิวเตอร์ เช่น ไมโครคอนโทรลเลอร์ที่เราจะนำมาใช้ในการประมวลผลหรืออ่านค่าจากเซนเซอร์ต่างๆ รวมไปถึงการเชื่อมต่อให้คอมพิวเตอร์สามารถสื่อสารกับไมโครคอนโทรลเลอร์ได้ โดยทำการรวบรวมข้อมูลจากเอกสารคู่มือที่ประกอบมากับบอร์ดไมโครคอนโทรลเลอร์ อินเทอร์เนท และหนังสือต่างๆจากหลายๆแหล่งประกอบกัน อีกทั้งยังสอบถามข้อมูลบางส่วนจากผู้มีประสบการณ์ และอาจารย์ที่ปรึกษาในการทำโครงงานนี้

3.2 วงจรอิเล็กทรอนิกส์และการเชื่อมต่อเข้ากับไมโครคอนโทรลเลอร์

3.2.1 วงจรแหล่งจ่ายไฟ

วงจรแหล่งจ่ายไฟสามารถใช้งานได้กับไฟ AC และ DC ขนาด 7-12V ได้ทันที โดยวงจรภาคแหล่งจ่ายไฟในส่วนที่เป็นวงจร Regulate นั้นจะมีส่วนที่เป็น +5V เพื่อจ่ายให้กับวงจรต่างๆ



รูปที่ 3.1 วงจรแหล่งจ่ายไฟ

3.2.2 วงจรในส่วนของ ET-AVR STAMP ATMEGA64

เป็นบอร์ดในตระกูล AVR ของบริษัท ATMEL ออกแบบโครงสร้างเป็นบอร์ดขนาดเล็ก ง่ายต่อการนำไปประยุกต์ต่อใช้งาน หรือใช้ต่อเข้ากับ PROJECT BOARD ใช้ในการต่อวงจรทดลองต่างๆ ก็ได้

• ใช้ MCU ตระกูล AVR เบอร์ ATMEGA64-16AI แบบ 64 PIN TQFP

• หน่วยความจำแบบ FLASH ขนาด 64KBYTE, RAM 4KBYTE และ EEPROM 2KBYTE • ใช้ X' TAL 16MHz

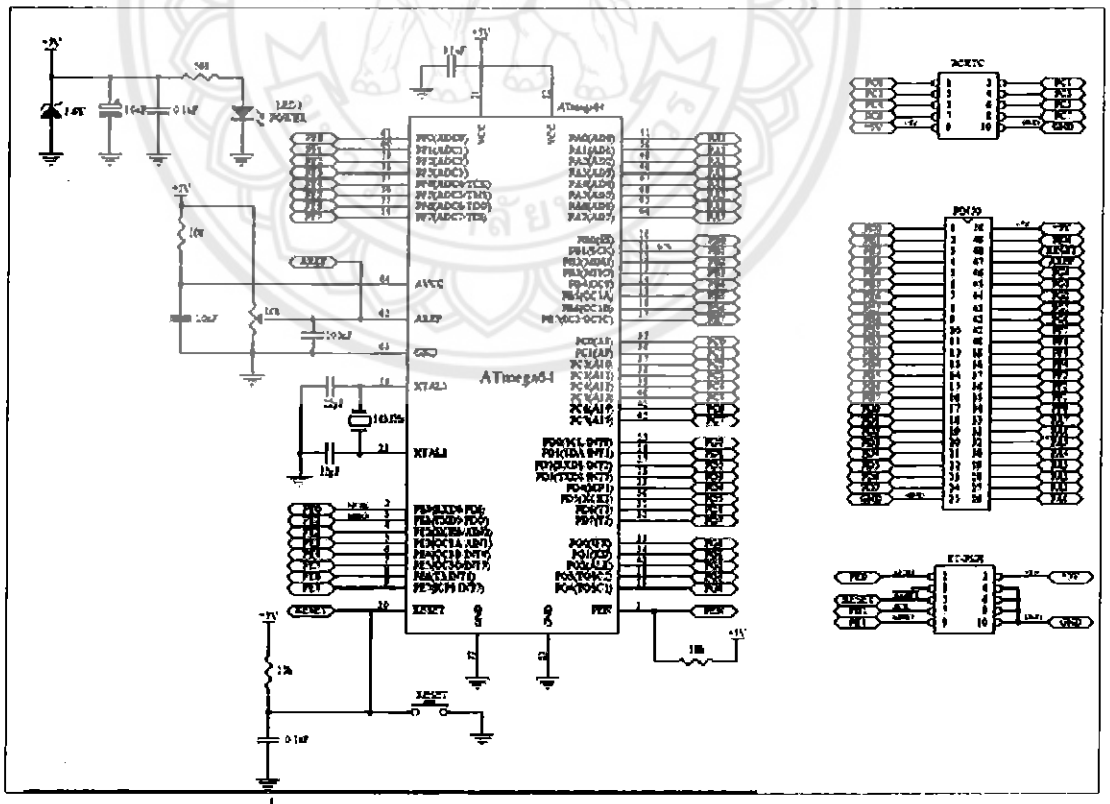
• รองรับการโปรแกรมเข้าตัว MCU ได้ 2 แบบ

1. แบบ SPI โดยใช้ชุด ET-AVR ISP ในแบบประหยัด ใช้ขั้ว PRINTER PORT

2. แบบ JTAG โดยใช้ชุด ET-AVR JTAG (RS232) V1.0 (คู่กับบอร์ด ET-AVR START KIT) สามารถดาวน์โหลดโปรแกรม และทำการดีบักแบบเรียลไทม์

• 53 I/O PIN สามารถต่อกับ I/O ระดับ 5V ได้

• 10 BIT A TO D จำนวน 8 CH, USART จำนวน 2 CH, SPI จำนวน 1 CH, I2C จำนวน 1 CH, TIMER/COUNTER 8 BIT จำนวน 2 CH, TIMER/COUNTER 16 BIT จำนวน 2 CH, 8 BIT PWM จำนวน 2 CH, WATCHDOG TIMER, REAL TIME COUNTER



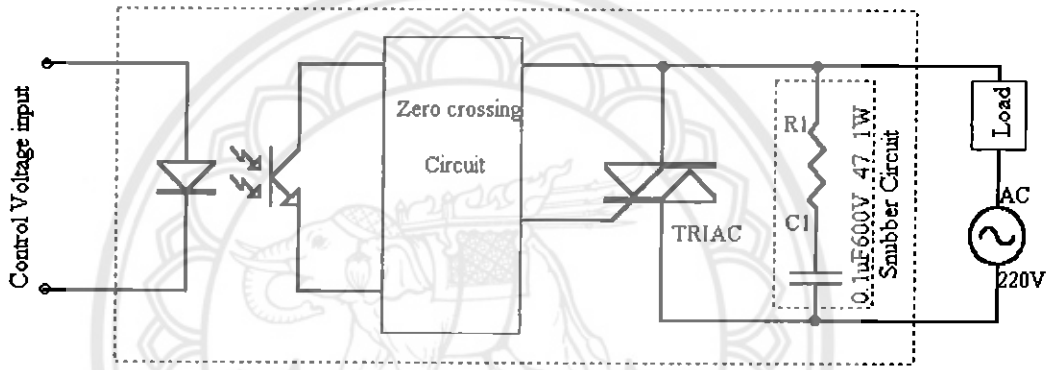
รูปที่ 3.2 วงจรในส่วนของ ET-AVR STAMP ATMEGA64

3.2.3 วงจรสวิตซ์ควบคุมและหลอด LED แสดงผล

- ET-BUSIO-SSRAC

บอร์ดทดลอง ET-BUSIO-SSRAC นี้ทำ หน้าที่เป็นโซลิดสเตตรีเลย์ (Solid state Relay : SSR) ซึ่งทำ หน้าที่เหมือนรีเลย์กลไกแบบธรรมดาที่ประกอบไปด้วยขดลวด และ หน้าสัมผัส เพียงแต่โซลิดสเตรีเลย์จะมีโครงสร้างภายในเป็นสารกึ่งตัวนำ ที่ใช้สำ หรับการตัดต่อวงจร แทนการตัดต่อวงจรด้วยหน้าสัมผัสเหมือนในรีเลย์แบบธรรมดา

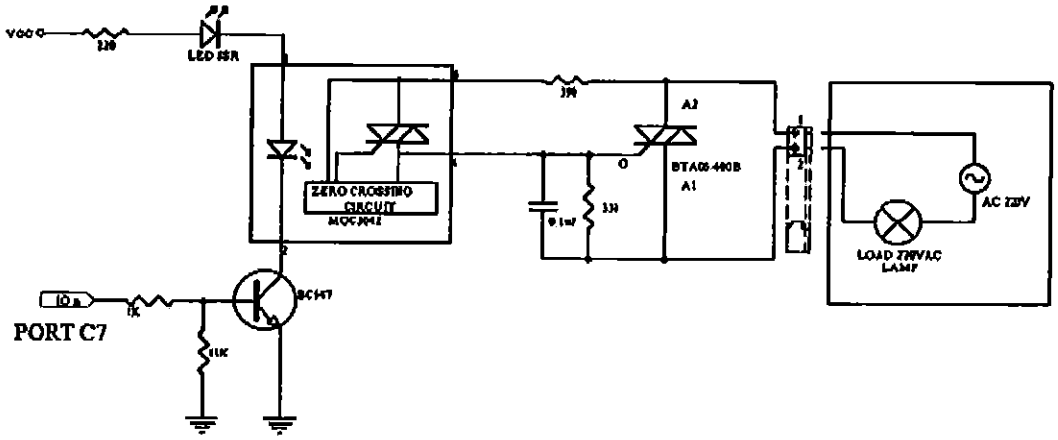
การทำงาน : โครงสร้างการทำงานโดยทั่วไปของ SSR แสดงในรูปที่ 1 ซึ่งจะเห็นว่าขั้วทางอินพุตจะทำ หน้าที่รับสัญญาณกระตุ้นเพื่อควบคุมการเปิดและปิดของวงจรทางเอาท์พุตโดยอินพุตและเอาท์พุต จะแยกกันทางไฟฟ้าซึ่งโดยทั่วไปจะควบคุมเอาท์พุตด้วยการเชื่อมโยงทางแสง



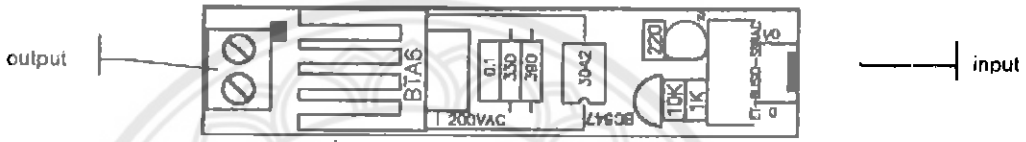
รูปที่ 3.3 โครงสร้างการทำงานโดยทั่วไปของ SSR

เมื่อผู้อ่านเข้าใจการทำงานของวงจร SSR ในรูปที่ 3.3 แล้วให้ผู้อ่านดูรูปวงจรจริงที่ส่วนท้ายของหัวข้อนี้โดยการทำงานจะใช้ไอซี MOC3042 ซึ่งเป็นไอซี Optoisolators โดยภายในจะรวมเอาส่วนของการเชื่อมโยงทางแสงและ Zero Crossing Circuit ไว้ด้วยกัน ซึ่งหน้าที่ของ Zero Crossing Circuit คือ จะตรวจสอบแรงดันที่จุดศูนย์เพื่อใช้ในการกำหนดจุดที่จะป้อนกระแสเกทให้กับ Triac ในส่วนของ R = 330 และ C = 0.1uF ที่ต่อกับขาเกทของ Triac นั้นทำหน้าที่แบ่งกระแสที่เข้าขาเกทไม่ให้สูงเกินไปและป้องกันสัญญาณรบกวนจากภายนอกตามลำดับ

การใช้งาน : วงจรนี้ใช้กับไฟ AC เท่านั้น เมื่อผู้อ่านป้อนลอจิก '1' ทางอินพุตจะทำให้ทรานซิสเตอร์ BC547 นำกระแส , LED จะสว่างซึ่งหมายความว่าโหลดจะครบวงจรในส่วนของภาคเอาท์พุตนั้น Triac สามารถทนกระแสสูงสุดที่ 6 A. แรงดัน 400 V. และในกรณีที่ผู้อ่านต่อโหลดที่เป็นตัวเหนี่ยวนำเช่น มอเตอร์ ผู้อ่านควรจะต่อวงจร Snubber เพิ่มเติมเข้าไปด้วยซึ่งก็คือ R1= 47 ohm/ 1 W และ C1 = 0.1uF/600V ดังแสดงในรูปที่ 3.3



รูปที่ 3.4 วงจรของ ET-BUSIO-SSRAC



รูปที่ 3.5 ตำแหน่งอุปกรณ์บน PCB

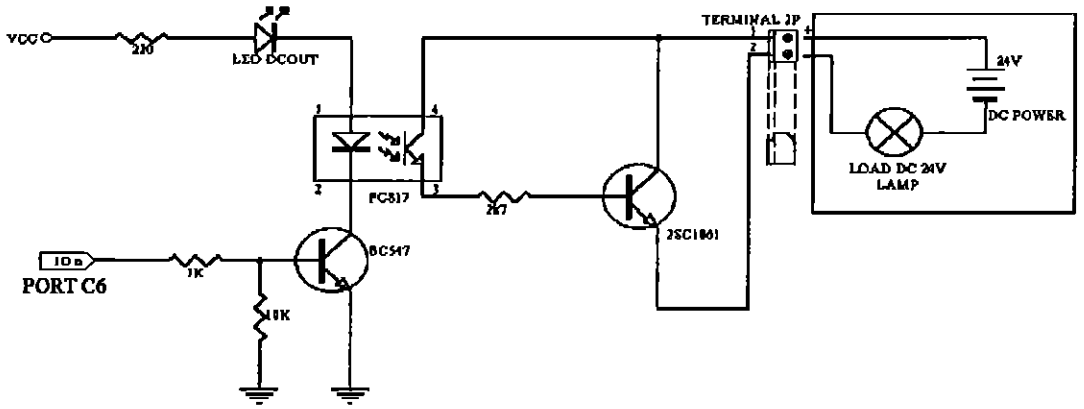
- ET-BUSIO-DCOUT

บอร์ดทดลอง ET-BUSIO-DCOUT เป็นบอร์ดที่ใช้ทดลองสำหรับตัดต่อแรงดันไฟ DC ทางเอาท์พุต โดยใช้ Power Transistor เบอร์ 2SC1061 ทำหน้าที่เป็นสวิตช์ตัดต่อวงจร ซึ่งเป็น Power Transistor ชนิด NPN ที่มีคุณสมบัติเด่นคือ แรงดันตกคร่อมที่ขา C-E ในขณะที่ Saturation มีค่าต่ำ Vce(sat) = 1.0 V.(Max) ที่กระแส I_c=2.0Aและ I_b=0.2A

การทำงาน: เมื่อมีสัญญาณลอจิก "1" เข้ามาทางอินพุตแรงดัน 5 โวลต์นี้จะถูกแบ่งแรงดันด้วยตัวต้านทาน 1K และ 10Kโดยแรงดันที่ถูกแบ่งนี้จะไปกระตุ้นทางขา B ของทรานซิสเตอร์BC547 ส่งผลให้ LED ใน photocouplerนั้นทำงานจึงทำให้ขา 3 และ ขา 4 ของ photocoupler นำกระแสไปกระตุ้นให้ power Transistor เบอร์ 2SC1061 นั้นนำกระแสตามไปด้วยส่งผลให้โหลดทางเอาท์พุตนั้นครบวงจร

การใช้งาน: สามารถต่อโหลดที่แรงดันไฟตรงทางเอาท์พุตของบอร์ด ET-BUSIO-DCOUT ได้สูงสุด 50 V. ที่กระแสไหลต่อเนื่อง เท่ากับ 3.0 Aผ่านขา Collector,พลังสูญเสียเท่ากับ 25 W ที่อุณหภูมิ 25 องศาเซลเซียส เมื่ออ่านลอจิก "1" ทางอินพุตจะทำให้ LED สว่างและโหลดจะครบวงจร

รูปที่ 3.6 วงจรของ- ET-BUSIO-DCOUT



รูปที่ 3.7 วงจรของ ET-BUSIO-DCOUT

- ET-BUSIO-SW

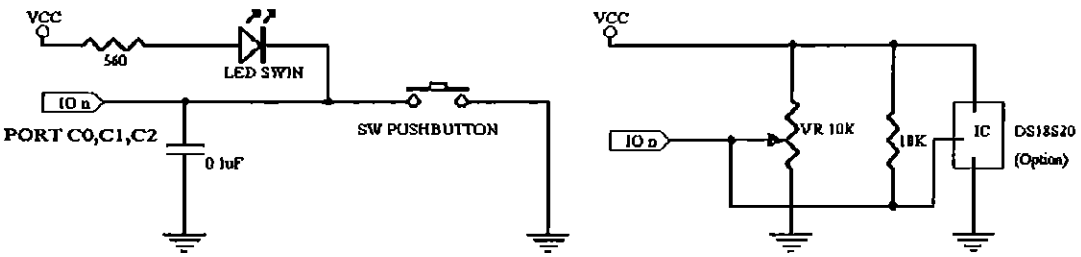
บอร์ดทดลอง ET-BUSIO-SW เป็นบอร์ดทดลองที่ใช้สำหรับค่าสถานะแรงดันจากการกดสวิทช์ SW IN ซึ่งกดเมื่อ LED จะสว่างและสถานะทางขา IO จะมีลอจิก"O"นอกจากนั้นในบอร์ดยังสามารถจะเพิ่มไอซี DS18S20 ซึ่งเป็นไอซี 1-Wire Digital Thermometer โดยสามารถให้อุณหภูมิด้วยการติดต่อในระบบบัสแบบ 1-Wire Busกับไมโครคอนโทรลเลอร์ (ซึ่งเป็นOption)

การทำงาน: กรณีใช้ SW IN เมื่อกดสวิทช์จะทำให้เกิดลอจิก"O"ออกไปทางขา IO ดังนั้นในสภาวะปกติสวิทช์จะไม่ถูกกดดังนั้นที่ขา IO จะมีลอจิก "1"

- กรณีใช้ VR 10K :การทำงานจะเหมือนเป็นการแบ่งค่าแรงดันให้กับขา IO

- กรณีใช้ไอซี DS18S20 : ไอซี DS18S20 มีขนาด 3 ขา โดยขา 1 จะเป็น GND, ขา 2 จะเป็นขาสัญญาณ (DQ) และขา 3 เป็น V_{DD} ซึ่งการใช้งานจะต้องมี R-pullup ประมาณ 10K ต่อระหว่างขา V_{DD} กับขา DQ และขณะใช้งานไม่ควรต่อ VR 10K

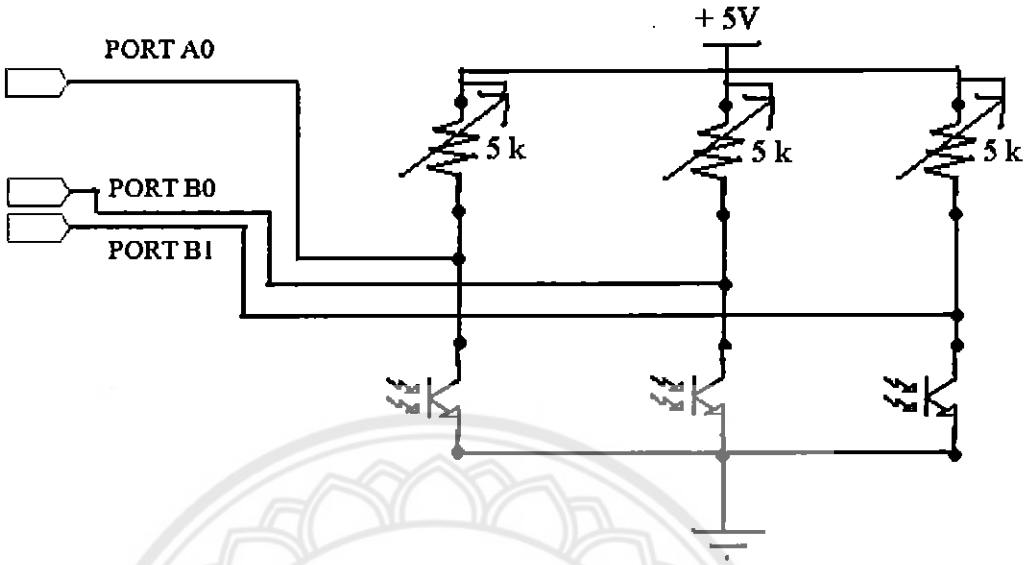
รูปที่ 3.8 ตำแหน่งอุปกรณ์บน PCB



รูปที่ 3.9 วงจรของ ET-BUSIO-SW

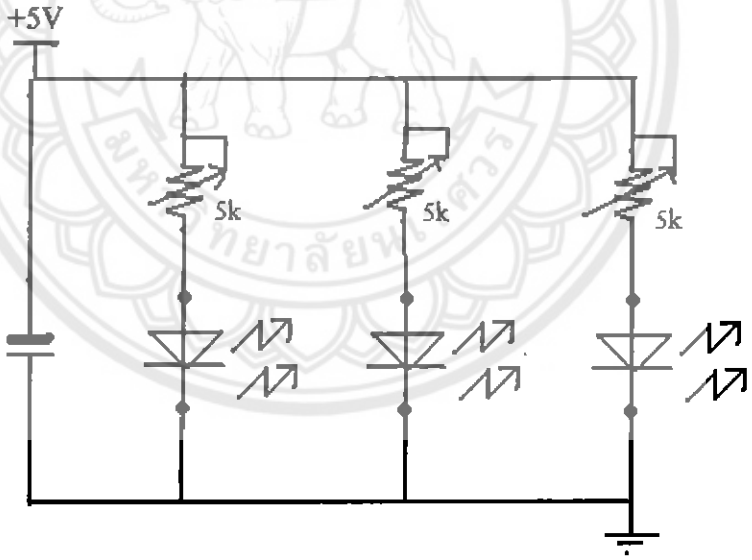
3.2.4 วงจร Sensor วัดตำแหน่ง

Sensor วัดแสงอินฟาร์เรด



รูปที่ 3.10 วงจร Sensor วัดตำแหน่ง

แหล่งกำเนิดแสง Sensor



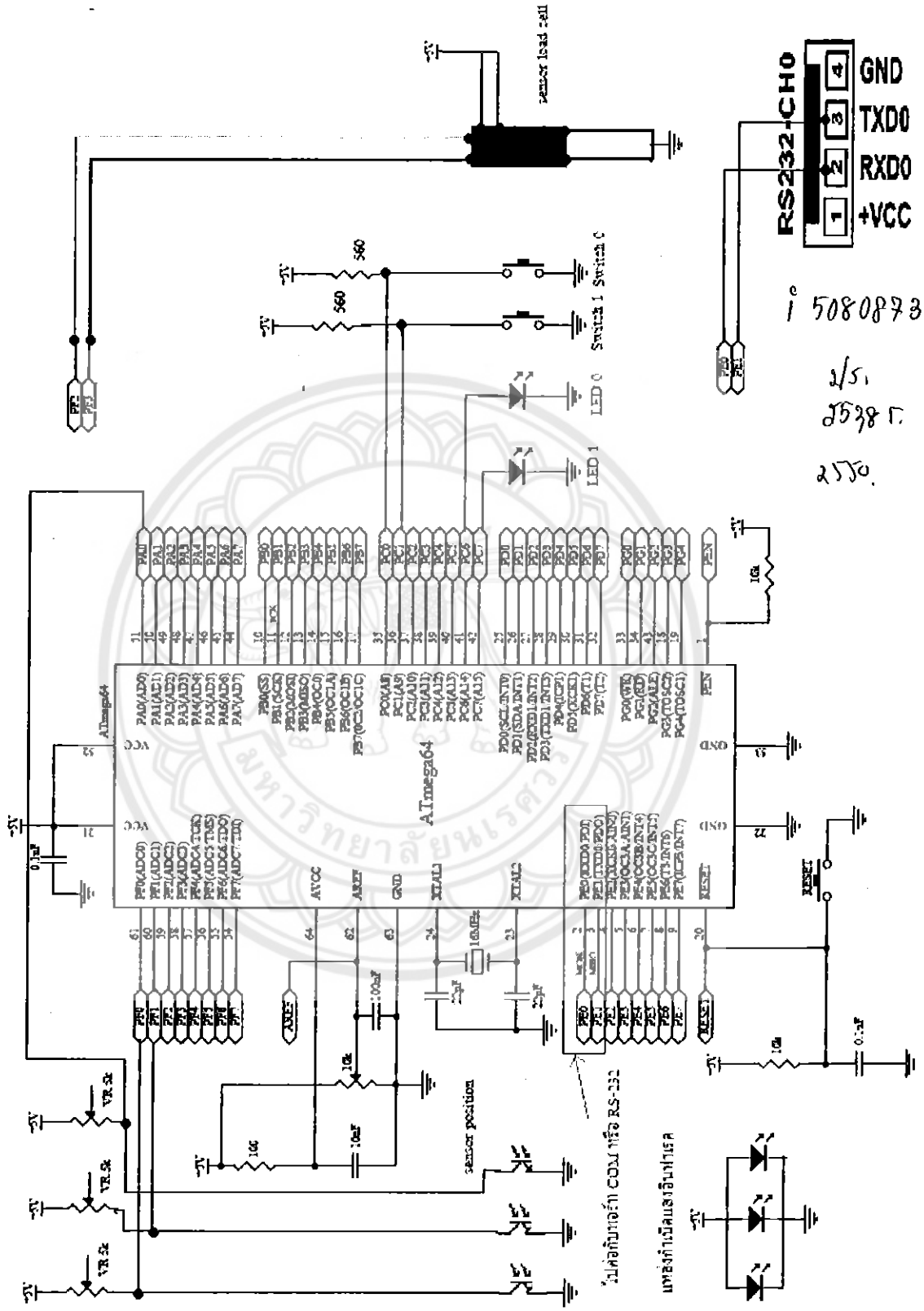
รูปที่ 3.11 วงจรแหล่งกำเนิดแสง sensor

3.2.6 Sensor โหลดเซลล์วัดค่าแรงเหวี่ยง

รูปที่ 3.12 Sensor โหลดเซลล์วัดค่าแรงเหวี่ยง

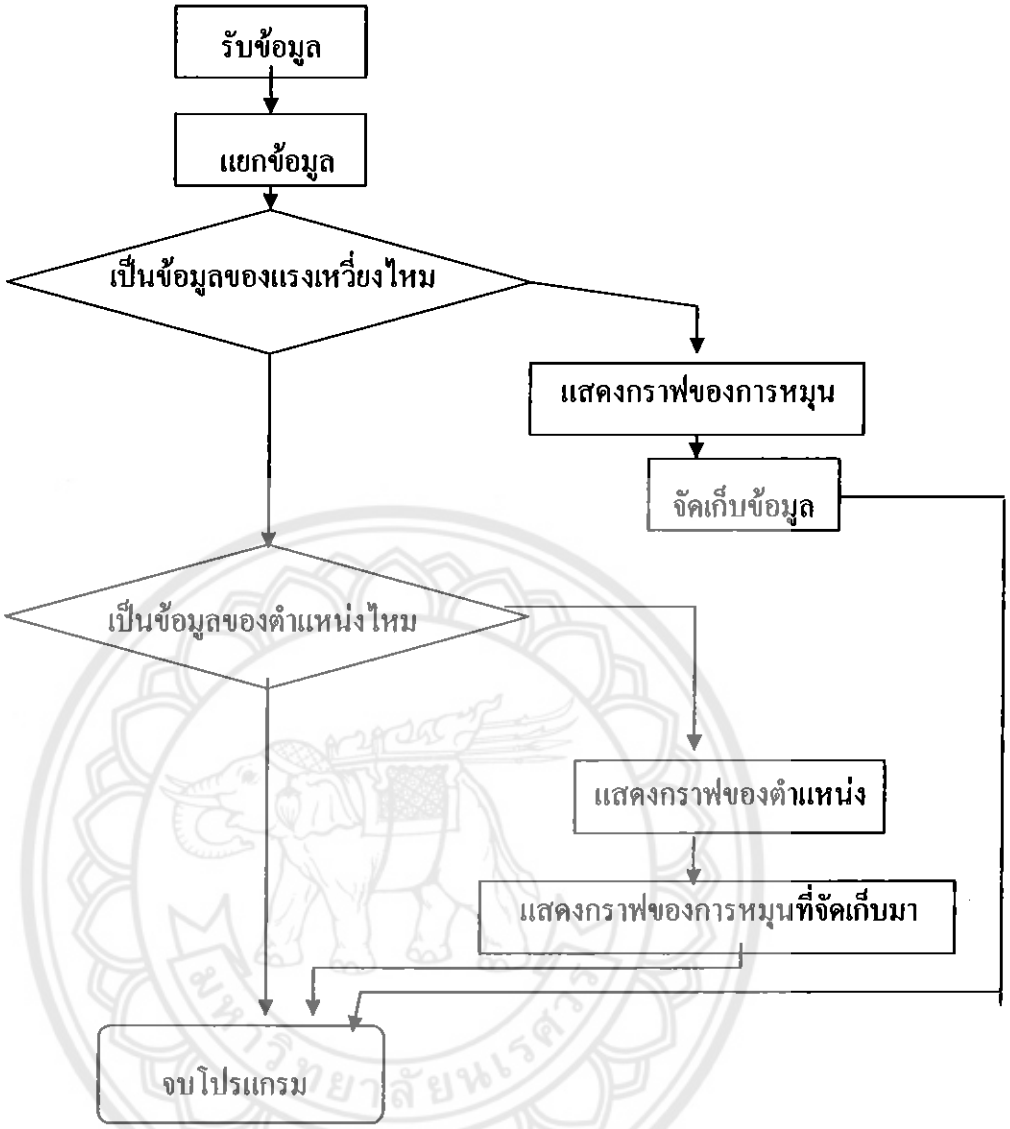
- สายสีฟ้าและสีเขียว : ต่อเข้ากับไฟเลี้ยง 5V
- สายสีดำและสายสีน้ำตาล : ต่อเข้ากับไฟลบ 0V
- สายสีแดง : ต่อเข้ากับพอร์ท F3
- สายสีขาว : ต่อเข้ากับพอร์ท F2

3.2.7 วงจรรวม



รูปที่ 3.13 วงจรรวมของชุดไมโครคอนโทรลเลอร์

3.3 การออกแบบทางด้านการเขียนโปรแกรม



รูปที่ 3.14 โฟลวชาร์ตการออกแบบทางด้านการเขียนโปรแกรม

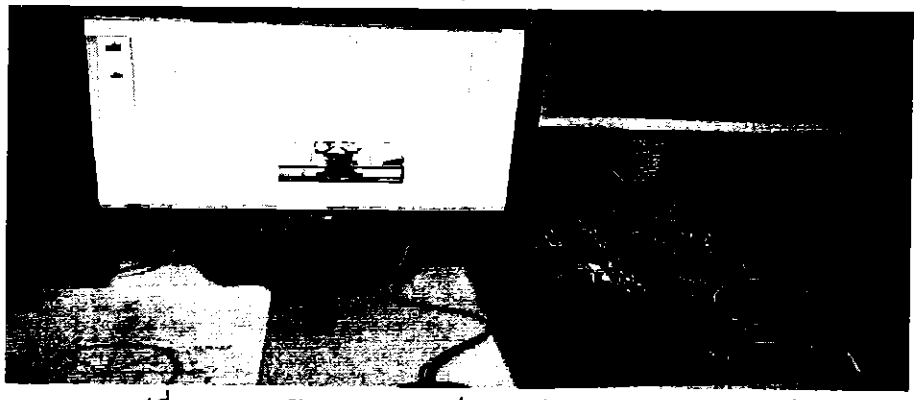
เราจะทำการกำหนดให้แมทแลบทำงานในโหมดของการเฝ้าตรวจจับหรือเป็นอินเทอร์รัฟของการสื่อสารข้อมูลแบบอนุกรม เมื่อมีการส่งข้อมูลมายังแมทแลบ แมทแลบจะกระโดดไปทำงานตามฟังก์ชันที่เรากำหนดไว้ก็ต่อเมื่อมีการตรวจพบ CR\LF("\r\n") โดยที่เราจะทำการกำหนดในการสื่อสารแบบอนุกรม(s1)

ซึ่งเราสามารถดูการกำหนดค่าต่างๆได้โดยใช้คำสั่ง get(s1) (ภาคผนวก ก.) ซึ่งตัวแปร S1 ของเราจะเก็บข้อมูลเกี่ยวกับการกำหนดค่าการสื่อสารแบบ RS-232 เราจะต้องทำการกำหนดดังนี้

BytesAvailableFcn = @bafblm00	(s1.BytesAvailableFcn=@bafblm00)
BytesAvailableFcnMode = terminator	(s1.BytesAvailableFcnMode ='terminator')
Terminator = CR/LF	(s1.Terminator = 'CR/LF')

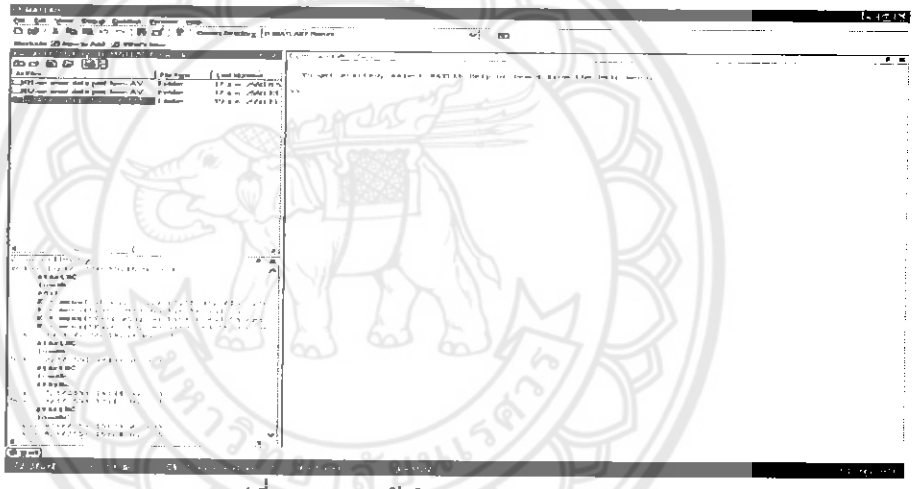
3.4 ขั้นตอนในการทดลองและทดสอบโปรแกรม

1. ทำการเปิดเครื่องคอมพิวเตอร์และเปิดชุดประมวลผลไมโครคอนโทรเลอร์



รูปที่ 3.15 การเปิดคอมพิวเตอร์และชุดไมโครคอนโทรลเลอร์

2. เข้าโปรแกรม MATLAB7.0 แล้วเลือกโฟลเดอร์ที่เก็บโปรแกรมของเราไว้



รูปที่ 3.16 การเข้าโปรแกรม MATLAB 7.0

3. จากนั้นพิมพ์คำสั่งที่ Command Windows >> startMC แล้วกด Enter หน้าจอ Command Windows จะปรากฏดังรูป

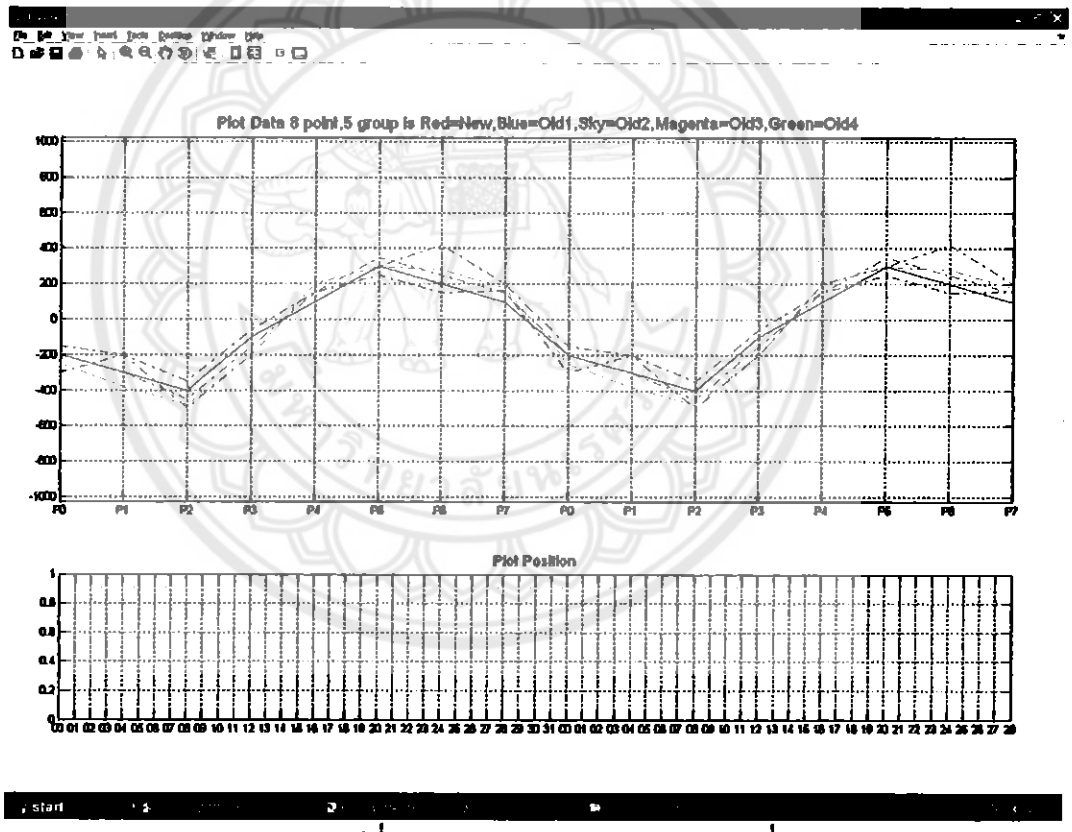
รูปที่ 3.17 การพิมพ์คำสั่ง startMC

4. จากนั้นพิมพ์คำสั่งที่ Command Windows >> loadMC แล้วกด Enter หน้าจอ Command Windows จะปรากฏดังรูป

```
Command Window
Connect to AVR
>> loadMC
sent to AVR
>>
```

รูปที่ 3.18 การพิมพ์คำสั่ง loadMC

5. ทำการหมุนตัวของทุ่นไดชาร์จโดยการ Start ตัวของมอเตอร์ที่จะทำการหมุนจะได้รูปกราฟดังรูป

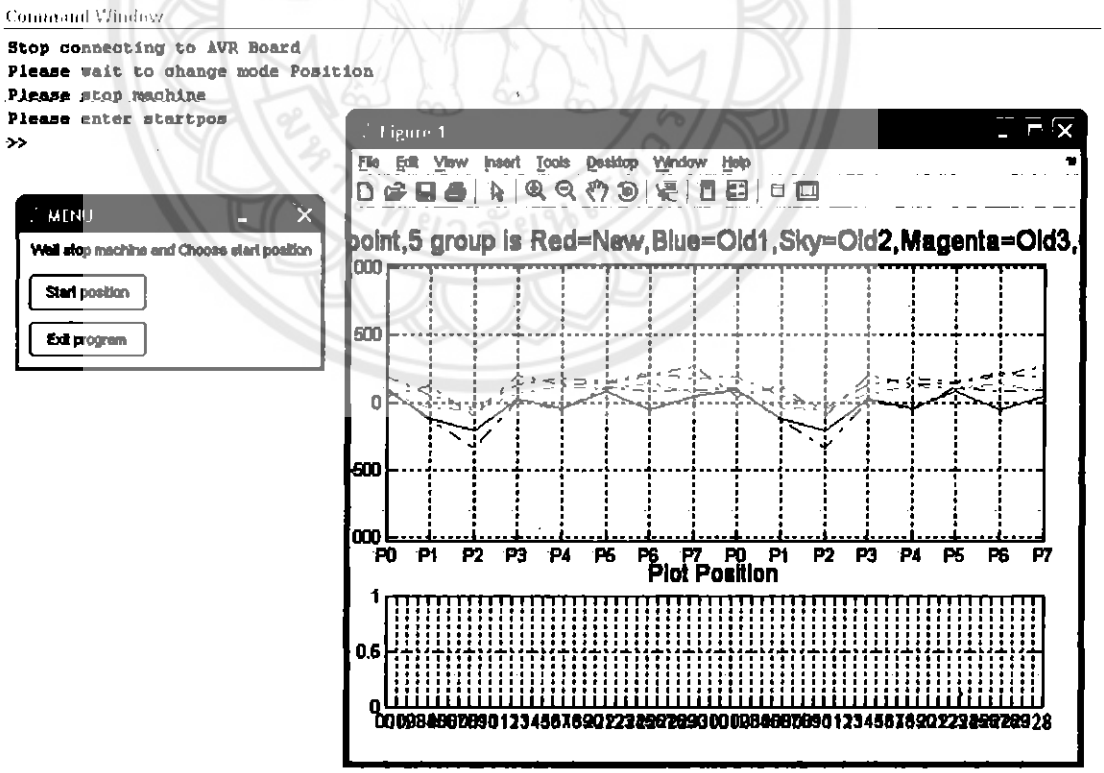


รูปที่ 3.19 กราฟของการทดสอบแรงเหวี่ยง

6. เมื่อได้รูปกราฟแสดงตำแหน่งที่ต้องการเจาะแล้วก็จะทำการเปลี่ยนโหมดไปสู่โหมดแสดงตำแหน่ง โดยการกดที่ปุ่ม switch 1 รอจนกว่าหลอดไฟสัญญาณ switch 1 ติด จึงจะปล่อยสวิตช์

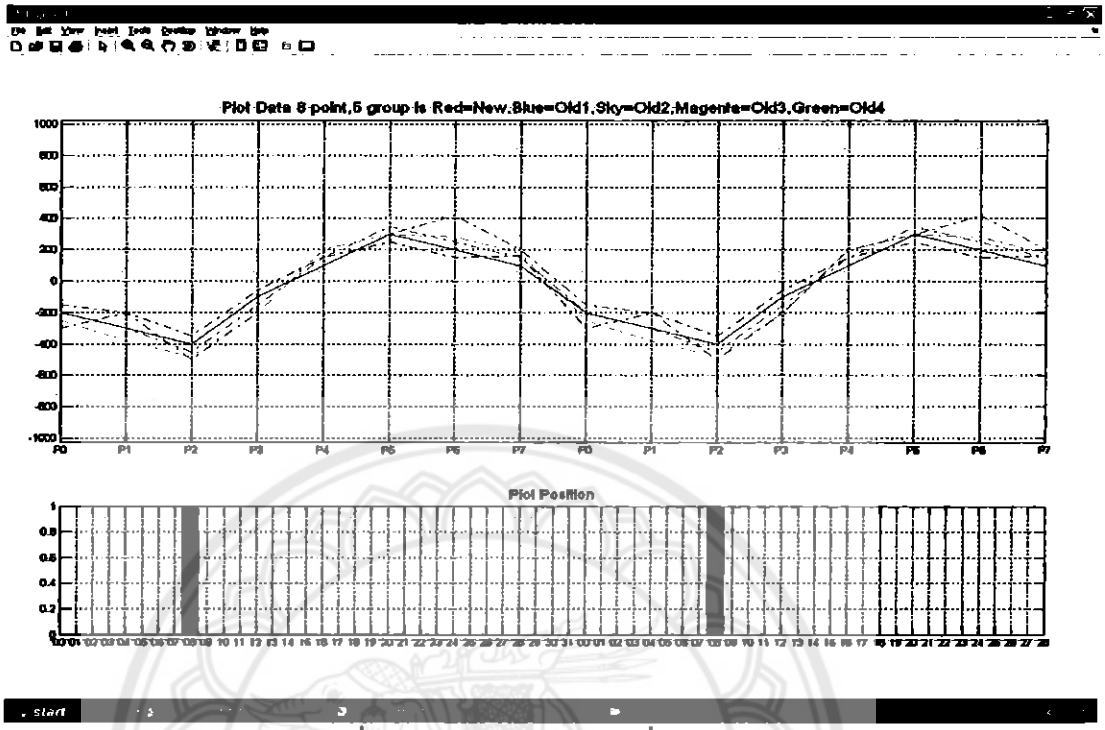
รูปที่ 3.20 การกดปุ่ม Switch 1

7.ทำการหยุดหมุนตัวทุ่นไดชาร์จโดยการ Stop มอเตอร์ เมื่อตัวทุ่นไดหยุดหมุนสังเกตุที่หน้าจอคอมพิวเตอร์จะปรากฏ ปุ่มเมนูเลือกการทำงานขึ้นมา 2 ปุ่มดังรูป จากนั้นทำการเลือกเมนู Start position เพื่อทำงานโหมดแสดงตำแหน่งต่อไป



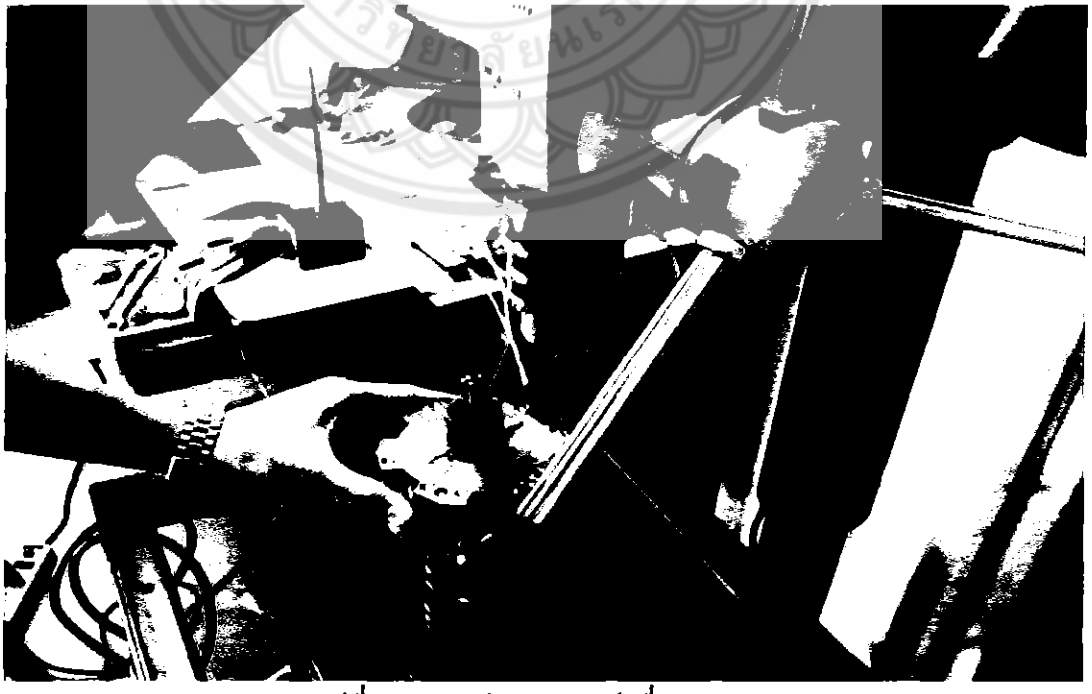
รูปที่ 3.21 การเปลี่ยนโหมด

8.ใช้มือในการหมุนตัวทุ่นไดชาร์จ แล้วดูกราฟเพื่อหาตำแหน่งที่ต้องการจะเจาะดังรูป



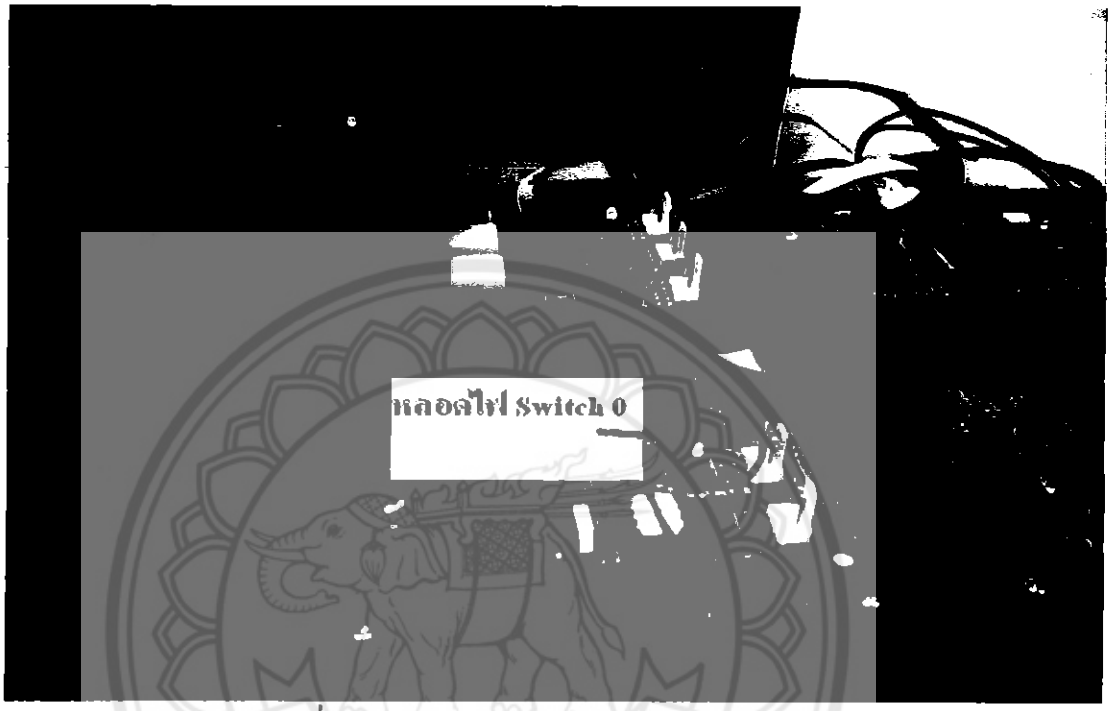
รูปที่ 3.22 กราฟของตำแหน่งที่จะทำการเจาะออก

9.เมื่อได้ตำแหน่งตามที่เราต้องการแล้ว เราจะทำการเจาะทุ่นไดชาร์จโดยการเปิดเครื่องเจาะสว่านเจาะลงไปตรงที่ตำแหน่งที่เราต้องการเจาะ ดังรูป



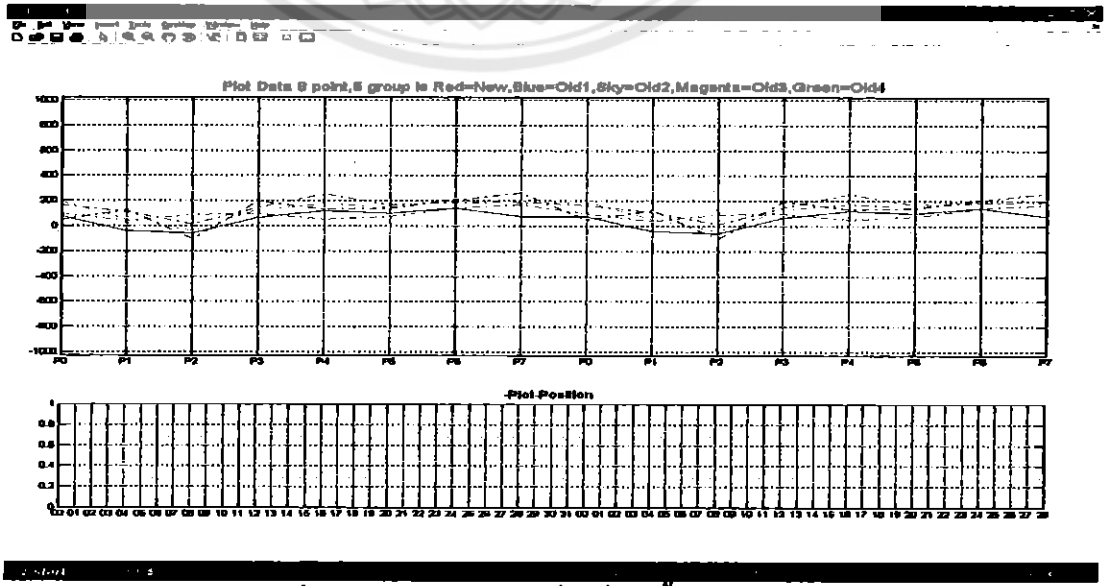
รูปที่ 3.23 กราฟของตำแหน่งที่จะทำการเจาะออก

10. เมื่อเราทำการเจาะตำแหน่งที่ไม่สมดุลออกเรียบร้อยแล้ว เราก็จะต้องมาทำทดสอบผลของการเจาะ ว่าที่เราทำการเจาะออกไปนั้นมีผลต่อการถ่วงสมดุลของตัวทุ่นไดชาร์จได้ขนาดไหน เราก็จะต้องทำการเปลี่ยนโหมดการทำงานจากโหมดแสดงตำแหน่ง ไปเป็นโหมดของการทดสอบการหมุน โดยกดที่ Switch 0 จะเห็นหลอด Switch 0 ติดดังรูป



รูปที่ 3.24 การกด Switch 0 แล้วหลอด LED switch 0 ติด

11. เมื่อเราทำการเปลี่ยนโหมดการทำงานไปเป็นโหมดทดสอบการหมุนเรียบร้อยแล้ว ก็ทำการเริ่ม Start มอเตอร์ที่ทำการหมุนตัวทุ่นไดชาร์จใหม่ จะได้กราฟที่แสดงขึ้นใหม่ดังรูป



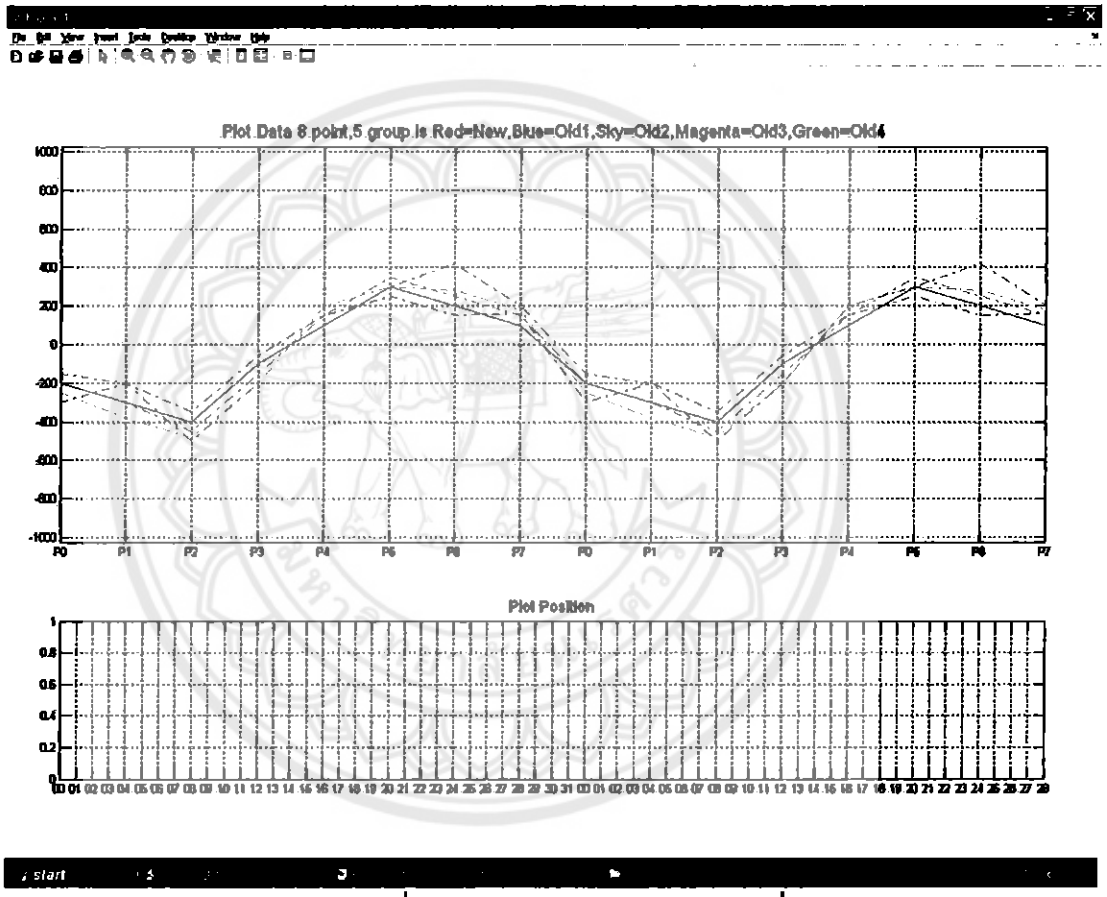
รูปที่ 3.25 กราฟของแรงเหวี่ยงที่เกิดขึ้นหลังถูกเจาะออกแล้ว

บทที่ 4

ผลการทดลองและการวิเคราะห์

4.1 ผลการทดลองโปรแกรมก่อนทำการเจาะถ่วงสมดุล

เมื่อเริ่มทำการทดลอง โดยการเริ่มทำการทดสอบการหมุนตัวทุ่นไดชาร์จไปเรื่อยๆ จนกระทั่งกราฟเฉลี่ยของการหมุนเริ่มคงที่มีรูปร่างใกล้เคียงกัน ดังรูป จึงจะสามารถเปลี่ยนโหมดของการหมุนทดสอบไปเป็นโหมดของการแสดงตำแหน่งการเจาะได้



รูปที่ 4.1 กราฟของการทดสอบแรงเหวี่ยง

จากผลการทดสอบการหมุนจะเห็นได้ว่า กราฟของการวัดน้ำหนักในแต่ละตำแหน่งยังขาดสมดุลอยู่มาก สังเกตได้จากกราฟการทดสอบการหมุนมีการแกว่งออกจากสมดุล ก็คือค่า 0 ค่อนข้างมากอยู่ ซึ่งเป็นผลมาจากแรงเหวี่ยงของตัวทุ่นไดชาร์จในแต่ล่ะตำแหน่งมีค่าไม่เป็น 0 นั้นก็หมายความว่า ตัวไดชาร์จมีบางตำแหน่งที่ยังขาดสมดุลอยู่ต้องทำการเจาะออกเพื่อถ่วงน้ำหนักในแต่ล่ะตำแหน่งให้มีค่าเข้าใกล้ 0 มากที่สุด โดยการถ่วงจะต้องคำนึงถึงตำแหน่งที่อยู่ตรงข้ามกันด้วย ดังนี้

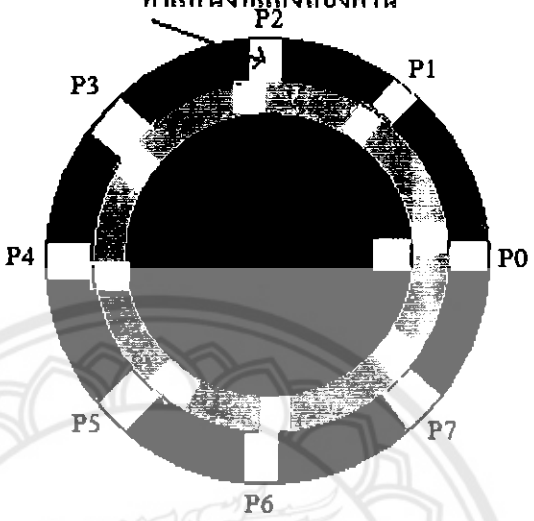
P0<---------->P4

P1<---------->P5

P2---------->P6

P3<---------->P7

ตำแหน่งที่แสงส่องผ่าน

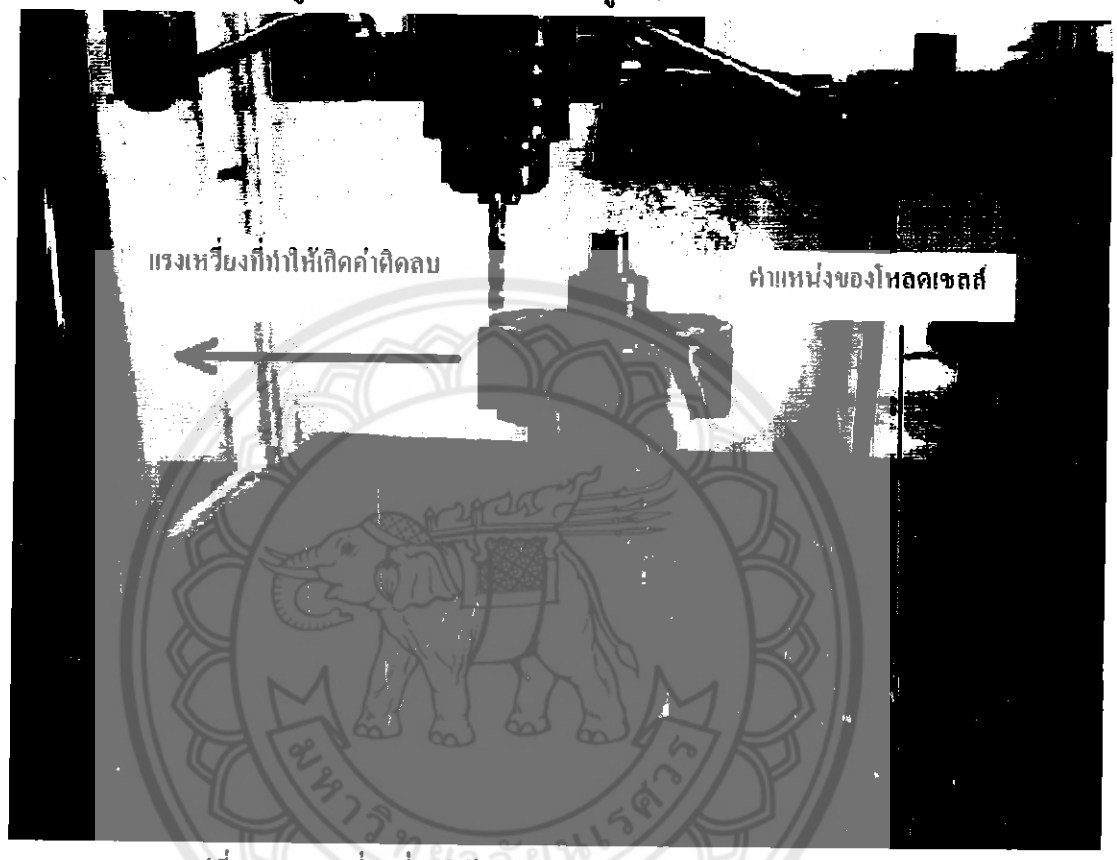


รูป 4.2 จานหมุนที่ใช้ในการวัดตำแหน่ง

ทั้ง 4 จุดจะสมดุลกันก็คือ เวลาถ่วงแล้วจุดทั้ง 4 จะมีค่าเท่ากันเป็นคู่ๆ ไปตามรูปที่ 4.2 เพราะแต่ละคู่เป็นจุดที่อยู่ตรงข้ามกัน เปรียบเสมือนกับการช่างน้ำหนักในตราชั่งสมัยโบราณ คือจุดสองจุดที่อยู่ตรงข้ามกันจะสมดุลกันเมื่อจุดทั้งสองจุดมีน้ำหนักเท่ากัน จึงจะทำให้ไม่เกิดแรงกดไปด้านใดด้านหนึ่ง

4.2 การเจาะถ่วงสมดุล

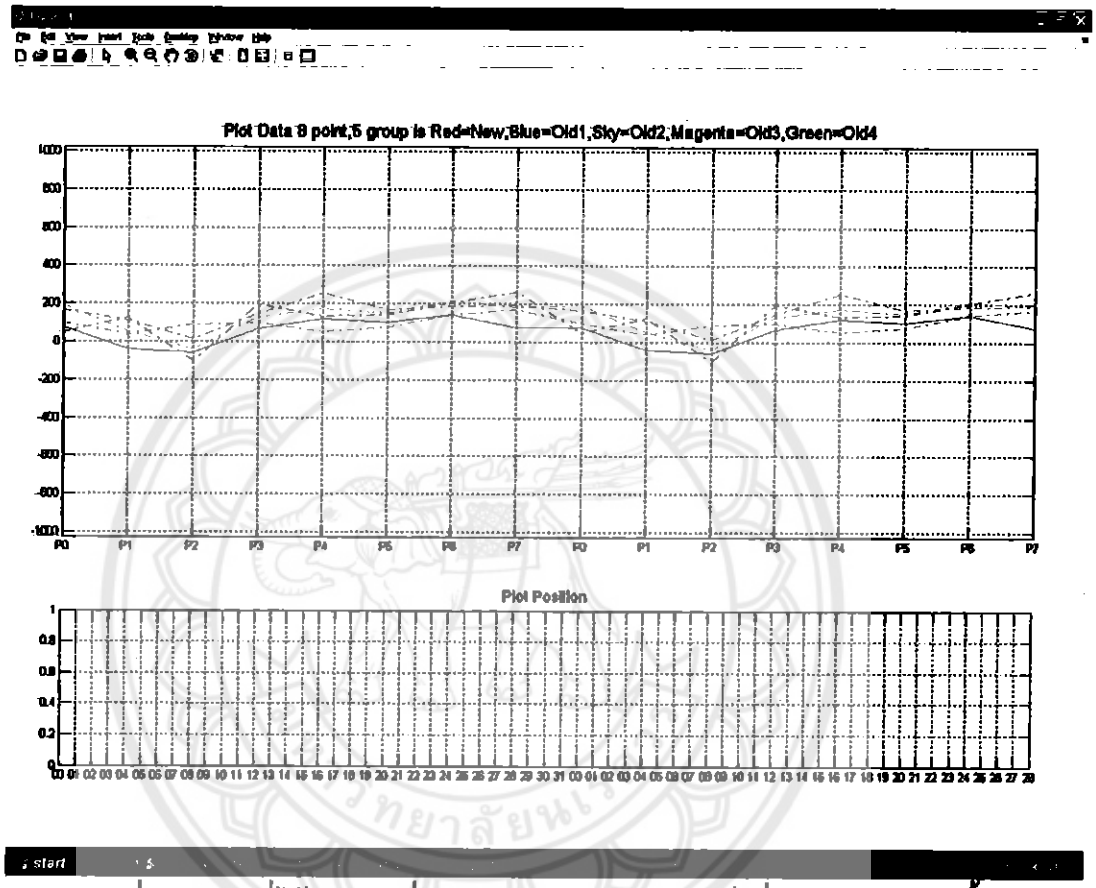
หลังจากที่เราได้รูปกราฟแสดงแรงเหวี่ยงที่ตำแหน่งต่างๆแล้ว เราก็จะเริ่มทำการเจาะตามรูปกราฟที่ได้โดยจะทำการเจาะในตำแหน่งที่เราพบว่ามีค่าติดลบ เพราะว่าตำแหน่งที่มีค่าติดลบจะเกิดแรงเหวี่ยงในด้านที่อยู่ตรงข้ามกับโหลดเซลล์ ดังรูป 4.3



รูปที่ 4.3 แรงเหวี่ยงที่ทำให้เกิดค่าติดลบและตำแหน่งของโหลดเซลล์

4.3 ผลการทดลองโปรแกรมหลังทำการเจาะถ่วงสมดุล

หลังจากที่ทำการเจาะถ่วงทุ่นไดชาร์จเรียบร้อยแล้ว เราก็จะต้องมาทำการทดสอบการหมุนใหม่โดยการเปลี่ยนจากโหมดแสดงตำแหน่งมาเป็นโหมดของการทดสอบการหมุนเหมือนเดิม โดยทดสอบการหมุนจนได้กราฟที่ค่อนข้างคงที่ ดังรูป 4.4



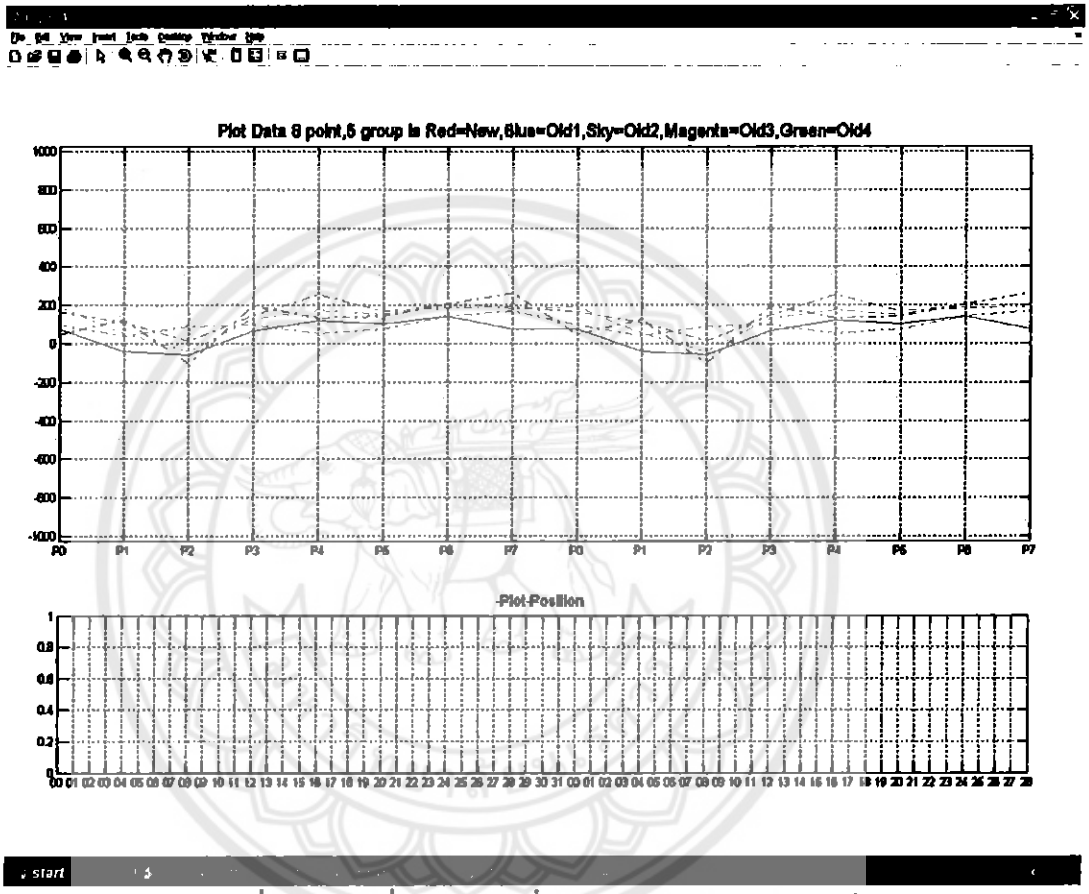
รูปที่ 4.4 กราฟที่ได้หลังจากที่เราทำการเจาะถ่วงทุ่นไดชาร์จเริ่มเข้าสู่สมดุลมากขึ้น

จากผลการทดสอบการหมุนตัวทุ่นไดชาร์จอีกครั้ง เราจะพบว่ากราฟที่ได้หลังจากที่เราทำการเจาะถ่วงทุ่นไดชาร์จเริ่มเข้าสู่สมดุลมากขึ้นกว่าเดิมมาก โดยสังเกตได้จากแต่ละจุดมีค่าเข้าใกล้ค่า 0 มากขึ้น นั้นก็หมายความว่า ตัวทุ่นไดชาร์จของเราเข้าใกล้สมดุลเข้าไปทุกที จากผลการทดลองจะเห็นได้ว่าตัวของโปรแกรมและเครื่องทดสอบของเราสามารถใช้งานได้ แถมยังช่วยทำให้วิธีการเจาะถ่วงทุ่นไดชาร์จของเราง่ายกว่าแบบเดิมมากขึ้นด้วย

จากการวิเคราะห์ผลการทดลอง เราจะเห็นได้ว่าโปรแกรมของเราสามารถใช้งานได้ค่อนข้างดีเพราะไม่มีการเกิดข้อผิดพลาดหรือ Error เกิดขึ้นในระหว่างที่ทำการทดสอบเมื่อเราทำตามขั้นตอนของโปรแกรมที่ได้เขียนไว้

บทที่ 5
บทสรุป

5.1 สรุปผลโครงงาน



รูปที่ 5.1 กราฟที่ได้หลังจากที่เราทำการเจาะถ่วงทุ่นไดชาร์จ

จากผลการทดสอบการหมุนตัวทุ่นไดชาร์จ เราจะพบว่ากราฟที่ได้หลังจากที่เราทำการเจาะถ่วงทุ่นไดชาร์จเริ่มเข้าสู่สมดุลมากขึ้นกว่าเดิมมาก โดยสังเกตได้จากแต่ละจุดมีค่าเข้าใกล้ค่า 0 มากขึ้น นั้นก็หมายความว่า ตัวทุ่นไดชาร์จของเราเข้าใกล้สมดุล

จากการวิเคราะห์ผลการทดลอง เราจะเห็นได้ว่าโปรแกรมของเราสามารถใช้งานได้ค่อนข้างดีเพราะไม่มีการเกิดข้อผิดพลาดหรือ Error เกิดขึ้นในระหว่างที่ทำการทดสอบเมื่อเราทำตามขั้นตอนของโปรแกรมที่ได้เขียนไว้

5.2 ปัญหาและวิธีการแก้ปัญหา

5.2.1 ปัญหา: ปัญหาเรื่องของเซนเซอร์โหลดเซลล์เกิดอ่านค่าน้ำหนักผิดพลาดเมื่อเราไปสัมผัสกับตัวของโหลดเซลล์ และสายไฟที่เชื่อมต่อกับไมโครคอนโทรลเลอร์ทำให้ข้อมูลที่ไมโครคอนโทรลเลอร์อ่านค่าและทำการส่งข้อมูลมาผิดพลาด

แนวทางการแก้ปัญหา: เวลาทดสอบต้องระมัดระวังอย่าไปสัมผัสกับโหลดเซลล์และสายไฟที่เชื่อมต่อกับไมโครคอนโทรลเลอร์ในระหว่างทำการทดสอบการหมุนเพื่ออ่านค่าน้ำหนักของแรงเหวี่ยง

5.2.2 ปัญหา: เวลาที่เราทำการเปลี่ยนโหมดของการทดสอบการหมุนเพื่ออ่านค่าน้ำหนักไปเป็นโหมดแสดงตำแหน่งของการเจาะโปรแกรม MATLAB เกิดการผิดพลาดรับข้อมูลไม่ทันในจังหวะที่ทำการเปลี่ยนโหมด

แนวทางการแก้ปัญหา: แก้ไขที่โปรแกรมไมโครคอนโทรลเลอร์ให้ทำการรอการกดสวิตซ์ที่ต่ออยู่กับพอร์ทของไมโครคอนโทรลเลอร์ก่อนทำการเปลี่ยนโหมดจากโหมดของการทดสอบการหมุนเพื่ออ่านค่าน้ำหนักไปเป็นโหมดแสดงตำแหน่งของการเจาะแล้วไมโครคอนโทรลเลอร์จะต้องรอรับข้อมูลจากโปรแกรม MATLAB ซึ่งเราจะต้องทำการหยุดการหมุนของมอเตอร์ก่อนแล้วจึงจะส่งข้อมูลเพื่อสื่อสารอ่านค่าตำแหน่งของการเจาะ

5.2.3 ปัญหา: การเก็บข้อมูลไว้เป็น Mex file หลายๆไฟล์แล้วทำการโหลดข้อมูลมาทำการแสดงผล ทำให้โปรแกรม MATLAB ทำงานช้าลง

แนวทางการแก้ปัญหา: เราจึงทำการแก้ปัญหาโดยการนำข้อมูลที่เราต้องการจะเก็บไว้ ไปเก็บไว้ในไฟล์ที่เดียวกันหมดคือ Obj.UserData ซึ่งอยู่ในไฟล์ S1

5.3 ข้อเสนอแนะ

ระยะเวลาในการศึกษาและทำการปรับปรุงเครื่องถ่วงสมดุลไดชาร์จมีน้อย ประกอบกับการพัฒนาเครื่องถ่วงสมดุลไดชาร์จทำได้ยากต้องใช้ความรู้ทางเครื่องจักรกลด้วยซึ่งเราไม่มีความถนัดมากนัก และเครื่องถ่วงสมดุลไดชาร์จยังไม่ค่อยสมบูรณ์มากนักเนื่องจากตำแหน่งที่เจาะไม่รู้ว่าจะทำการเจาะลึกเท่าไร ถ้ามีเวลามากกว่านี้อาจพัฒนาจนสามารถใช้งานในอุตสาหกรรมได้

เอกสารอ้างอิง

[1] ธีรวัฒน์ ประกอบผล. ภาษาแอสเซมบลีสำหรับ MCS-51. กรุงเทพมหานคร : ส.ส.ท. สมาคมส่งเสริมเทคโนโลยี (ไทย-ญี่ปุ่น). 2547.

[2] มนัส สังวรศิลป์. คู่มือโปรแกรม MATLAB ฉบับสมบูรณ์. กรุงเทพมหานคร : อินโฟเพรส. 2543.

ภาคผนวก ก

Source Code ของโปรแกรม MATLAB

1. Source Code ของฟังก์ชันที่ใช้งานประกอบด้วย

1.1 Source Code ฟังก์ชัน bafblm00

```
function bafblm00(obj, event)
w = obj.UserData.w;
state = obj.UserData.state;
y = obj.UserData.y;
out = fscanf(obj);
ntout = out(1:end-2);
x1=1:1:61;
y1=zeros(1,61);
r=0;
t=0:1:15;
if state == 0
  switch lower(ntout)
    case {'info'}          % % มีข้อมูลเข้า
      disp(ntout);


    case 'Testing'         %% เริ่มทดสอบ
      fprintf('Start testing please waint\n');
      buf=0;


    case 'npos'            %% จำนวนตำแหน่ง
      fprintf('Number of pos=');
      state = 1;


    case 'nloop'           %% จำนวนรอบ
      fprintf('Number of loop=');
```

```
        state = 2;

    case 'sdata'            %% มีการส่งข้อมูล
        disp('receiver data from AVR board');
        state = 3;
        h0=axes('position',[0.05 0.4 0.9 0.5]);
        showdata5(t,obj);
        set(h0,'XTick',[0:1:15]);
        set(h0,'XLim',[0 15]);
        set(h0,'XTickLabel',['P0';'P1';'P2';'P3';'P4';'P5';'P6';'P7';'P0';'P1';'P2';'P3';'P4';'P5';'P6';'P7'])
        set(h0,'YLim',[-1024 1024]);
        t0=title('Plot Data 8 point,5 group is
Red=New,Blue=Old1,Sky=Old2,Magenta=Old3,Green=Old4');
        set(t0,'FontSize',15);
        showdata4(t,obj);
        showdata3(t,obj);
        showdata2(t,obj);

    case 'posting'          %% มีการส่งค่าตำแหน่ง
        clc
        fprintf('Stop connecting to AVR Board\n');
        fprintf('Please wait to change mode Position\n');
        fprintf('Please stop machine\n');
        fprintf('Please enter startpos\n');
        startpos();

    case 'anticlockwise'        %% หมุนทวนเข็ม
        fprintf('Anticlockwise\n');
        state=4;

    case 'forward'              %% หมุนตามเข็ม
        fprintf('Forward clockwise\n');
```

```
        state=5;

    case 'startposition'          %% ผ่านจุดเริ่มต้น
        fprintf('start position point 0000\n');
        state=6;

    case 'errorposition'          %% ตรวจสอบพบ Error
        clc
        fprintf('Found Error position\n');
        errorpoint();

    case 'stoppos'                %% หยุดการส่งข้อมูลเพื่อเปลี่ยนโหมด
        fprintf('Change Mode receiver data form AVR\n');
        receiverAVR();

    otherwise
        disp('otherwise1');
        disp(ntout);

  end
  ntold = ntout;
  obj.UserData.state = state;
elseif state ~= 0
  switch state
    case 1
      disp(out);

    case 2
      disp(out);

    case 3                        %% ทำการแยกข้อมูล จัดเก็บ และพลอตกราฟ
      [m,n]=size(out);
```

```
rdata=out(3:n);
disp(rdata);
x=n-9;
b=1;
a=1;
r=1;
for i=1:1:x
  if rdata(1,i)=='k'
    y(1,b)=str2num(buf);
    buf=0;
    b=b+1;
    a=i+1;
  end;
  if rdata(1,i)~='k'
    buf=rdata(a:i);
  end;
end;
obj.UserData.y = y;
showdata1(t,obj);
h1=axes('position',[0.05 0.1 0.9 0.2]);
p1=bar(x1,y1);grid on
set(h1,'XTick',[1:1:61]);
set(h1,'XLim',[1 61]);
set(h1,'YLim',[0 1]);
t1=title('Plot Position');
set(t1,'FontSize',13);

set(h1,'XTickLabel',['00';'01';'02';'03';'04';'05';'06';'07';'08';'09';'10';'11';'12';'13';'14';'15';'16';'17';'18'
;'19';'20';'21';'22';'23';'24';'25';'26';'27';'28';'29';'30';'31';'00';'01';'02';'03';'04';'05';'06';'07';'08';'09';'10'
;'11';'12';'13';'14';'15';'16';'17';'18';'19';'20';'21';'22';'23';'24';'25';'26';'27';'28']);
```

```
case 4                    %% โหมดของการแสดงตำแหน่ง
   point=str2num(out(8:9));
   showposition(state,point,y1,x1);
   fprintf('Position point=%d\n',point');
   if point==0
      showdata1_5(t,obj);
   end;

case 5                    %% โหมดของการแสดงตำแหน่ง
   point=str2num(out(8:9));
   showposition(state,point,y1,x1);
   fprintf('Position point=%d\n',point');
   if point==0
      showdata1_5(t,obj);
   end;

case 6                    %% แสดงการผ่านจุดเริ่มต้น
   point=str2num(out(8:9));
   showposition(state,point,y1,x1);
   fprintf('Start position 0000.\n');
   showdata1_5(t,obj);

otherwise
   disp('otherwise2');
   disp(out);

end
obj.UserData.state = 0;
end
```

```
%%%%%%%%%%%%%% OLD DATA %%%%%%%%%%%%%%%%%%%%%%%
if r==1
   copyw3(w,obj);
   copyw2(w,obj);
   copyw1(w,obj);
   copyw(y,obj);
   receiverAVR();
end;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

1.2 Source Code ฟังก์ชัน showdata1

```
function showdata1(t,obj)
y = obj.UserData.y;
%%%%%%%%%%%%%%%%%%%%%%%% Converter %%%%%%%%%%%%%%%%%%%%%%%
        [m,n]=size(t);
        for i=1:1:n
           if i<=8
              z(1,i)=y(1,i);
           end;
           if i>8
              z(1,i)=y(1,i-8);
           end;
        end;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%show new data%%%%%%%%%%%%%%%%%%
        plot(t,z,'r-');grid on
        state=0;
        hold off
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%
```

1.3 Source Code ฟังก์ชัน showdata2

```
function showdata2(t,obj)
w = obj.UserData.w;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%
        [m,n]=size(t);
        for i=1:1:n
          if i<=8
             z(1,i)=w(1,i);
          end;
          if i>8
             z(1,i)=w(1,i-8);
          end;
        end;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%
%%%%%%%%%%%%%%%%%% Plot data old %%%%%%%%%%%%%%
        plot(t,z,'b-.');grid on
        hold on
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

1.4 Source Code ฟังก์ชัน showdata3

```
function showdata2(t,obj)
w = obj.UserData.w;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        [m,n]=size(t);
        for i=1:1:n
          if i<=8
```

```
            z(1,i)=w(1,i);
          end;
          if i>8
            z(1,i)=w(1,i-8);
          end;
        end;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%% Plot data old %%%%%%%%%%%%%%
        plot(t,z,'b-.');grid on
        hold on
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

1.5 Source Code ฟังก์ชัน showdata4

```
function showdata4(t,obj)
w2 = obj.UserData.w2;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%
        [m,n]=size(t);
        for i=1:1:n
          if i<=8
            z(1,i)=w2(1,i);
          end;
          if i>8
            z(1,i)=w2(1,i-8);
          end;
        end;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%
%%%%%%%%%%%%%%%%%% Plot data old %%%%%%%%%%%%%%
```

```
        plot(t,z,'m-.');grid on
        hold on
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

1.6 Source Code ฟังก์ชัน showdata5

```
function showdata5(t,obj)
w3 = obj.UserData.w3;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%
        [m,n]=size(t);
        for i=1:1:n
          if i<=8
              z(1,i)=w3(1,i);
          end;
          if i>8
              z(1,i)=w3(1,i-8);
          end;
        end;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%
%%%%%%%%%%%%%%%%%%%%% Plot data old %%%%%%%%%%%%%%%
        plot(t,z,'g-.');grid on
        hold on
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

1.7 Source Code ฟังก์ชัน showdata1_5

```
function showdata1_5(t,obj)
h0=axes('position',[0.05 0.4 0.9 0.5]);
showdata5(t,obj);
set(h0,'XTick',[0:1:15]);
```

```
set(h0,'XLim',[0 15]);
set(h0,'XTickLabel',['P0';'P1';'P2';'P3';'P4';'P5';'P6';'P7';'P0';'P1';'P2';'P3';'P4';'P5';'P6';'P7'])
set(h0,'YLim',[-1024 1024]);
t0=title('Plot Data 8 point,5 group,Red=New,Blue=Old1,Sky=Old2,Magenta=Old3,Green=Old4');
set(t0,'FontSize',15);
showdata4(t,obj);
showdata3(t,obj);
showdata2(t,obj);
showdata1(t,obj);
```

Source Code ฟังก์ชัน copyw

```
function copyw(y,obj)
w=y;
obj.UserData.w=w;
```

Source Code ฟังก์ชัน copyw1

```
function copyw1(w,obj)
w=obj.UserData.w;
w1=w;
obj.UserData.w1=w1;
```

1.10 Souour Code ฟังก์ชัน copyw2

```
function copyw2(w,obj)
w1=obj.UserData.w1;
w2=w1;
obj.UserData.w2=w2;
```

1.11 Source Code ฟังก์ชัน copyw3

```
function copyw3(w,obj)
w2=obj.UserData.w2;
w3=w2;
obj.UserData.w3=w3;
```

1.12 Source Code ฟังก์ชัน errorpoint

```
function errorpoint()
fprintf('Modify error\n');
load s1
fprintf(s1,'a\n');
```

1.13 Source Code ฟังก์ชัน receiverAVR

```
function receiverAVR()
load s1
fprintf(s1,'a\n');
```

1.14 Source Code ฟังก์ชัน showposition

```
function showposition(state,point,y1,x1)
if point~=0
   y1(1,1+point)=1;
   a1=33+point;
   if a1<=61
      y1(1,a1)=1;
   end;
end;
if point==0
   y1(1,1)=1;
   y1(1,33)=1;
end;
h1=axes('position',[0.05 0.1 0.9 0.2]);
p1=bar(x1,y1);grid on
set(p1,'BarWidth',0.2);
set(h1,'XTick',[1:1:61]);
set(h1,'XLim',[1 61]);
set(h1,'YLim',[0 1]);
t1=title('Plot Position');
```

```
set(t1,'FontSize',13);
set(h1,'XTickLabel',['00';'01';'02';'03';'04';'05';'06';'07';'08';'09';'10';'11';'12';'13';'14';'15';'16';'17';'18'
;'19';'20';'21';'22';'23';'24';'25';'26';'27';'28';'29';'30';'31';'00';'01';'02';'03';'04';'05';'06';'07';'08';'09';'10'
;'11';'12';'13';'14';'15';'16';'17';'18';'19';'20';'21';'22';'23';'24';'25';'26';'27';'28']);
```

1.15 Source Code ฟังก์ชัน startpos

```
function startpos()
K = menu('Wait stop machine and Choose start position','Start position','Exit program') ;
if K==1
   load s1
   fprintf(s1,'a\n');
end;
if K==2
   clc
   fclose(s1)
   fprintf('Disconnect\n');
   exit
end;
```

2. Source Code คำสั่งที่สร้างขึ้นเป็น Mex ไฟล์

2.1 Source Code คำสั่ง startMC

```
clc
close all
clear all
load s1
fopen(s1)
fprintf('Connect to AVR\n');
```

2.2 Source Code คำสั่ง loadMC

```
fprintf(s1,'a\n');
fprintf('sent to AVR\n');
```

2.3 Source Code คำสั่ง clearMC

```
fclose(s1)
clear s1
load s1
fopen(s1)
clear s1
load s1
s1
```

2.4 Source Code คำสั่ง stopMC

```
clc
fclose(s1)
fprintf('Disconnect\n');
```

3. ค่าที่กำหนดในรีจิสเตอร์ S1

```
ByteOrder = littleEndian
BytesAvailable = 0
BytesAvailableFcn = [1x1 function_handle]
BytesAvailableFcnCount = 48
BytesAvailableFcnMode = terminator
BytesToOutput = 0
ErrorFcn =
InputBufferSize = 512
Name = Serial-COM1
ObjectVisibility = on
OutputBufferSize = 512
OutputEmptyFcn =
RecordDetail = compact
RecordMode = overwrite
RecordName = record.txt
RecordStatus = off
Status = closed
Tag =
```

Timeout = 10

TimerFcn =

TimerPeriod = 1

TransferStatus = idle

Type = serial

UserData = [1x1 struct]

ValuesReceived = 0

ValuesSent = 0

SERIAL specific properties:

BaudRate = 9600

BreakInterruptFcn =

DataBits = 8

DataTerminalReady = on

FlowControl = none

Parity = none

PinStatus = [1x1 struct]

PinStatusFcn =

Port = COM1

ReadAsyncMode = continuous

RequestToSend = on

StopBits = 1

Terminator = CR/LF

ภาคผนวก ข

Source Code ของโปรแกรมไมโครคอนโทรลเลอร์

```
/*****************************************************
This program was produced by the
CodeWizardAVR V1.24.6 Professional
Automatic Program Generator
© Copyright 1998-2005 Pavel Haiduc, HP InfoTech s.r.l.
http://www.hpinfotech.com
e-mail:office@hpinfotech.com

Project : BLM00
Version : V1.0
Date    : 6/30/2007
Author  : F4CG
Company : F4CG
Comments:


Chip type           : ATmega64
Program type        : Application
Clock frequency     : 16.000000 MHz
Memory model        : Small
External SRAM size  : 0
Data Stack size     : 1024
*****************************************************/

#include <mega64.h>
#include <stdio.h>
#include <string.h>
#include <delay.h>
//#include <math.h>
```

```
#include <c:\cvavr\bin\blmachine\blmmachine00.h>
#include <c:\cvavr\bin\blmachine\supportfunc.c>

void main(void)
{
unsigned int n = 0;
// Declare your local variables here
#include <c:\cvavr\bin\blmachine\init_main.c>
#define cst 0b0000
#define cf1 0b0001
#define cf2 0b0111
#define cf3 0b1110
#define cf4 0b1000
#define ca1 0b0010
#define ca2 0b1011
#define ca3 0b1101
#define ca4 0b0100
#define cer 0b1111
/* disable interrupts */
#asm("cli")
// Assume that the pos is not at zero position.
// This lead to find the initial zero pos on machine has been installed first

while (1)
{
        unsigned int i,c0, c1, c2;
        zr = 1;
        //printf("Disable Interrupt All\r\n");
        do {
          nchar = scanf("%s",str0);
          printf("%s\r\n",str0);
          //printf("%d  %d \r\n",strlen(str0),nchar);
```

```
    }
    while (strcmpf(str0,wstop) == 0);
      c0 = PINC.0;
      c1 = PINC.1;
      c2 = PINC.2;

      if (c0 == 0)
      {
          PORTC.6 = 0b01;//testing mode
          PORTC.7 = 0b00;
      };
      if (c1 == 0)
      {
          PORTC.6 = 0b00;//position mode
          PORTC.7 = 0b01;
      };
if (PORTC.7 == 0)
   {
     for (i = 0;i< 32; i++){
       x[i] = 0;
       }

      n = 0;
      pos = 0;
      state = 0b0000;

      while  ((zr != 0))
          {
          a0 = PINA.0;
          d1d0 = PIND & 0b00000011;
          d1 = d1d0 >> 1;
          d0 = d1d0 & 0b00000001;
```

```
zr = d0|d1|a0;

if (zr == 0)
{
    printf("info\r\n");
    printf("Testing\r\n");
};
};

while ((state != cer) & (n < maxpv))
    {
    unsigned char oldstate;
    bit nd1d0;
    a0 = PINA.0;
    d1d0 = PIND & 0b00000011;
    nd1d0 = !d1d0;
    oldstate = state;

    switch (state)
    {
    case cst :
            switch (d1d0) {
            case 0b00 :
            state = cst;
            break;
            case 0b01 : state = cf1;
            subpos();
             break;
            case 0b10 : state = ca1;
            addpos();
            break;
            default   : state = cer;};
```

```
    break;

case ca1 :
        switch (d1d0) {
        case 0b00 : state = cf4;
        subpos(); break;
        case 0b01 : state = cer;
                break;
        case 0b10 : state = ca1;
                break;
        case 0b11 : state = ca2;
        addpos();
        if (pos == 2) {
            n = n + 1;}
        break;
        default   : state = cer;};
    break;

case ca2 :
        switch (d1d0) {
        case 0b00 : state = cer;
                break;
        case 0b01: state = ca3;
        addpos();       break;
        case 0b10: state = cf3;
        subpos();       break;
        case 0b11: state = ca2;
                break;
        default: state = cer;};
    break;

case ca3 :
```

```
                switch (d1d0) {
                case 0b00 :
                      start_adc();
                      state = ca4;
                      addpos();
                      x[pos] = x[pos] + get_adc();
                      break;
                case 0b01 : state = ca3;
                            break;
                case 0b10 : state = cer;
                            break;
                case 0b11 : state = cf2;
                 subpos();       break;
                default   : state = cer;};
          break;

     case ca4 :
                switch (d1d0) {
                case 0b00 : state = ca4;
                            break;
                case 0b01 : state = cf1;
                 subpos();       break;
                case 0b10 : state = ca1;
                addpos();       break;
                case 0b11 : state = cer;
                            break;
                default: state = cer;};
          break;

     case cf1 :
                switch (d1d0) {
                case 0b00 : state = ca4;
```

```
                addpos();       break;
                case 0b01 : state = cf1;
                        break;
                case 0b10 : state = cer;
                        break;
                case 0b11 : state = cf2;
                 subpos();
                 if (pos == 2) {
                      n = n - 1;}
                 break;
                default: state = cer;};
        break;

case cf2 :
                switch (d1d0) {
                case 0b00 : state = cer;
                 if (pos == 2) {
                      n = n - 1;}
                        break;

                case 0b01 : state = ca3;
                addpos();       break;
                case 0b10 : state = cf3;
                 subpos();      break;
                case 0b11 : state = cf2;
                        break;
                default   : state = cer;};
        break;


case cf3 :
                switch (d1d0) {
                case 0b00 :
```

```
                start_adc();
                state = cf4;
                subpos();
                x[pos] = x[pos] + get_adc();
                break;
            case 0b01 : state = cer;
                    break;
            case 0b10 : state = cf3;
                    break;
            case 0b11 : state = ca2;
            addpos();       break;
            default: state = cer;};
        break;

    case cf4 :
            switch (d1d0) {
            case 0b00 : state = cf4;
                    break;
            case 0b01 : state = cf1;
            subpos();     break;
            case 0b10 : state = ca1;
            addpos();       break;
            case 0b11 : state = cer;
                    break;
            default: state = cer;};
        break;
    default :     state = cer;

    };

};
```

```
if (state == cer) {
printf("info\r\n");
printf("-- State Error -- \r\n"); }

printf("nloop\r\n %3d\r\n",n);
printf("npos\r\n %3d\r\n",numofpos);

n = 0; // reuse variable again
printf("sdata\r\n");//start data

for (i = 0;i< numofpos; i= i + 4)
{
      printf("%6dk",x[i]);
      x[i] = 0;
}

;printf("edata\r\n");//end data
n = 0;
}
else if (PORTC.7 == 1)
{
     for (i = 0;i< 32; i++){
       x[i] = 0;
       }

     n = 0;
     pos = 0;
     state = 0b0000;

     while  ((zr != 0))
          {
          a0 = PINA.0;
```

```
d1d0 = PIND & 0b00000011;
d1 = d1d0 >> 1;
d0 = d1d0 & 0b00000001;
zr = d0|d1|a0;

if (zr == 0)
{
    printf("info\r\n");
    printf("posting\r\n");
    nchar = scanf("%s",str0);
    printf("%s\r\n",str0);
};
};

while ((state != cer) & (PORTC.7 == 1))
{
unsigned char oldstate;
bit nd1d0;
a0 = PINA.0;
d1d0 = PIND & 0b00000011;
nd1d0 = !d1d0;
oldstate = state;

switch (state)
{
case cst :
        switch (d1d0) {
        case 0b00 :
        state = cst;
        break;
        case 0b01 : state = cf1;
        subpos();
```

```
            break;
            case 0b10 : state = ca1;
            addpos();
            break;
            default   : state = cer;};
    break;

case ca1 :
            switch (d1d0) {
            case 0b00 : state = cf4;
            subpos(); break;
            case 0b01 : state = cer;
                    break;
            case 0b10 : state = ca1;
                    break;
            case 0b11 : state = ca2;
            addpos();
            if (pos == 2) {
                n = n + 1;}
            break;
            default   : state = cer;};
    break;

case ca2 :
            switch (d1d0) {
            case 0b00 : state = cer;
                    break;
            case 0b01: state = ca3;
            addpos();       break;
            case 0b10: state = cf3;
            subpos();       break;
            case 0b11: state = ca2;
```

```
                    break;
            default: state = cer;};
        break;

case ca3 :
            switch (d1d0) {
            case 0b00 :
                start_adc();
                state = ca4;
                addpos();
                x[pos] = x[pos] + get_adc();
                break;
            case 0b01 : state = ca3;
                    break;
            case 0b10 : state = cer;
                    break;
            case 0b11 : state = cf2;
             subpos();      break;
            default   : state = cer;};
        break;

case ca4 :
            switch (d1d0) {
            case 0b00 : state = ca4;
                    break;
            case 0b01 : state = cf1;
             subpos();      break;
            case 0b10 : state = ca1;
            addpos();       break;
            case 0b11 : state = cer;
                    break;
            default: state = cer;};
```

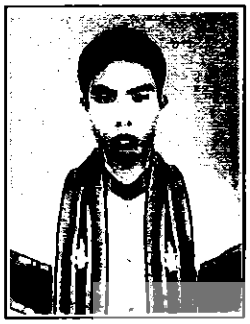
```
        break;

case cf1 :
            switch (d1d0) {
            case 0b00 : state = ca4;
            addpos();       break;
            case 0b01 : state = cf1;
                        break;
            case 0b10 : state = cer;
                        break;
            case 0b11 : state = cf2;
            subpos();
            if (pos == 2) {
                n = n - 1;}
            break;
            default: state = cer;};
      break;

case cf2 :
            switch (d1d0) {
            case 0b00 : state = cer;
            if (pos == 2) {
                n = n - 1;}
                        break;

            case 0b01 : state = ca3;
            addpos();       break;
            case 0b10 : state = cf3;
            subpos();      break;
            case 0b11 : state = cf2;
                        break;
            default   : state = cer;};
```

```
        break;

case cf3 :
            switch (d1d0) {
            case 0b00 :
                start_adc();
                state = cf4;
                subpos();
                x[pos] = x[pos] + get_adc();
                break;
            case 0b01 : state = cer;
                        break;
            case 0b10 : state = cf3;
                        break;
            case 0b11 : state = ca2;
            addpos();       break;
            default: state = cer;};
        break;

case cf4 :
            switch (d1d0) {
            case 0b00 : state = cf4;
                        break;
            case 0b01 : state = cf1;
            subpos();     break;
            case 0b10 : state = ca1;
            addpos();     break;
            case 0b11 : state = cer;
                        break;
            default: state = cer;};
        break;
default :     state = cer;
```

```
};
if (oldstate != state)
    switch (state) {
    case cst : printf("startposition\r\n 0000 %d\r\n",pos);
                break;
    case ca1 : printf("anticlockwise\r\n1 0010 %d\r\n",pos);
                break;
    case ca2 : printf("anticlockwise\r\n2 1011 %d\r\n",pos);
                break;
    case ca3 : printf("anticlockwise\r\n3 1101 %d\r\n",pos);
                break;
    case ca4 : printf("anticlockwise\r\n4 0100 %d\r\n",pos);
                break;
    case cf1 : printf("forward\r\n1 0001 %d\r\n",pos);
                break;
    case cf2 : printf("forward\r\n2 0111 %d\r\n",pos);
                break;
    case cf3 : printf("forward\r\n3 1110 %d\r\n",pos);
                break;
    case cf4 : printf("forward\r\n4 1000 %d\r\n",pos);
                break;
    default  : printf("errorposition\r\n 1111 %d\r\n",pos);
    };

     c0 = PINC.0;
     c1 = PINC.1;
     c2 = PINC.2;

     if (c0 == 0)
     {
          PORTC.6 = 0b01;//testing mode
```

```
                PORTC.7 = 0b00;
                printf("stoppos\r\n");

            };
            if (c1 == 0)
            {
                PORTC.6 = 0b00;//position mode
                PORTC.7 = 0b01;
            };

        };

        if (state == cer)
        {
            printf("info\r\n");
            printf("-- State Error -- \r\n");
        }
        n = 0;

    }
};
}
```

ประวัติผู้เขียนโครงงาน



ชื่อ นายสังวรณ์ สีสุทัศน์

ภูมิลำเนา 135 หมู่ที่ 8 ต.แม่สุก อ.แม่ใจ จ.พะเยา 56130

ประวัติการศึกษา

- จบประถมศึกษาจากโรงเรียนบ้านแม่จว้า
- จบมัธยมศึกษาจากโรงเรียนแม่ใจวิทยาคม
- ปัจจุบันกำลังศึกษาอยู่ระดับปริญญาตรีชั้นปีที่ 4 สาขาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร