เครื่องปรับทิศทางสายอากาศ
ด้วยสเตปปิ้งมอเตอร์

ANTENNA POSITIONER
WITH STEPPING MOTOR

โดย
นาย ปรีชา สุริย์แสง
นาย สิทธิเดช ปิยะชาติ

ปริญญานิพนธ์นี้ เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาอุตสาหกรรมศาสตร์บัณฑิต
สาขาวิชาเทคโนโลยีอิเล็กทรอนิกส์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2536

หัวข้อปริญญานิพนธ์	เครื่องปรับทิศทางสายอากาศด้วยสเตปปิ้งมอเตอร์
จัดทำโดย	นายปรีชา สุริย์แสง 34131219
	นายสิทธิเดช ปิยะชาติ 34131236
สาขาวิชา	เทคโนโลยีอิเล็กทรอนิกส์อุตสาหกรรม
อาจารย์ที่ปรึกษา	อาจารย์ ประดิษฐ์ วัชรพิบูลย์
ปีการศึกษา	2536

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง อนุมัติให้นับปริญญานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร ปริญญาอุตสาหกรรมศาสตร์บัณฑิต

---------- คณบดี คณะวิศวกรรมศาสตร์

(รศ.ดร.สมเกียรติ ศุภเดช)

---------- อาจารย์ที่ปรึกษา

(อาจารย์ประดิษฐ์ วัชรพิบูลย์)

คณะกรรมการสอบปริญญานิพนธ์

---------- ประธานกรรมการ

( )

---------- กรรมการ

( )

---------- กรรมการ

( )

---------- กรรมการ

( )

---------- กรรมการ

( )

เครื่องปรับทิศทางสายอากาศด้วยสเตปปิ้งมอเตอร์

ผู้ร่วมงาน

นายปรีชา สุริย์แสง 34131219

นายสิทธิเดช ปิยะชาติ 34131236

อาจารย์ที่ปรึกษา

อาจารย์ ประดิษฐ์ วัชรพิบูลย์

ปีการศึกษา 2536

บทคัดย่อ

สเตปปิ้งมอเตอร์ เป็นมอเตอร์ชนิดหนึ่งที่วงการอุตสาหกรรม นิยมนำมาใช้ควบคุมตำแหน่งและทิศทาง เพราะมีความแม่นยำทางตำแหน่งสูง สำหรับการควบคุมส่วนใหญ่จะใช้ Micro processer เนื่องจากปัจจุบันเทคโนโลยีทางด้าน Microprocesser ได้พัฒนาสูงขึ้นเรื่อยๆ จนทำให้ความสามารถในการทำงานสูง การใช้งานไม่ยุ่งยากซับซ้อน มีอุปกรณ์สนับสนุนมากมายจึงทำให้การพัฒนาเป็นไปอย่างมีประสิทธิภาพ อีกทั้งด้านราคาก็ไม่แพง

โครงการนี้ก็เป็นอีกโครงการหนึ่งที่นำ Microprocesser (เบอร์ Z-80) มาควบคุมการทำงานของสเตปปิ้งมอเตอร์ โดยมีจุดประสงค์ที่จะนำสเตปปิ้งมอเตอร์ มาประยุกต์ใช้กับการควบคุมการปรับทิศทางของสายอากาศ

Antenna Positioner with Stepping Motor

Mr. Preecha Surisang 34131219

Mr. Sittidech Piyachat 34131236

Advisor

Mr. Pradit Vachrapibool

1993

ABSTRACT

Stepping motor is the motor to be used in industial. This type motor used for control position and direction. Because, it have high accuracy of the position. The most for controlled by the Computer System to use Microprocesser. Because, today the technology for Microprocesser has developed until the certain stage nowaday with high capacity. The operation is not too complicate, many equipment to support which the development has efficency and the price of cheap.

This project is one of project to use Microprocesser # Z-80 control the operation of Stepping motor.

The purpose of this project to take the Stepping motor to apply for control the direction of antenna.

กิตติกรรมประกาศ

การจัดทำปริญญานิพนธ์ฉบับนี้ ได้รับความกรุณาเอื้อเฟื้อจากบุคคลต่างๆ ในการให้คำแนะนำ ให้ข้อมูลและอำนวยความสะดวก ดังมีรายนามต่อไปนี้

1. อาจารย์ประดิษฐ์	วัชรพิบูลย์	อาจารย์ประจำคณะวิศวกรรมศาสตร์ และอาจารย์ที่ปรึกษาในการจัดทำปริญญานิพนธ์
2. คุณศักดา	สุขใย	วิศวกรฝ่าย Test Engineer บริษัท NS Electronics Bangkok (1993) จำกัด ช่วยให้คำแนะนำในการทำโครงงาน
3. คุณพงเทพ	ชุติวิทูรชัย	วิศวกรฝ่าย Automation Support บริษัท ซีเกทเทคโนโลยี (ประเทศไทย) จำกัด ช่วยให้คำแนะนำในการทำโครงงาน

สารบัญ

หน้า

บทที่ 1
บทนำ

เครื่องปรับทิศทางสายอากาศ ด้วยสเตปปิ้งมอเตอร์ (Antenna Positioner With Stepping Motor) เป็นอุปกรณ์ที่ใช้ปรับ,หมุนสายอากาศตั้งแต่ 0-360 องศา ทางแนวราบ ซึ่งเหมาะสำหรับการหมุนสายอากาศเพื่อทดสอบ แพททอร์น (Radiation Pattern) ซึ่งทำให้สะดวกกว่าการปรับด้วยมือมาก จากการศึกษาการปรับสายอากาศที่ทดสอบด้วยมือนั้น พบว่าเกิดค่าผิดพลาดเนื่องจากตัวบุคคลที่ปรับ จะอยู่ใกล้สายอากาศ จึงทำให้คลื่นมีการสะท้อนเนื่องจากตัวบุคคลมีผลให้ Radiation Pattern ที่วัดไม่ได้ค่าที่แท้จริง โครงงานนี้จึงเกิดขึ้นเพื่อแก้ปัญหานี้ โดยการใช้สเตปปิ้งมอเตอร์มาควบคุมการหมุนของสายอากาศแทนคน

โครงงานนี้ ใช้ Z-80 CPA (Z-80 Control Pack) เป็นตัวควบคุมและสั่งการให้ระบบโดยจะเลือกพอร์ทเบอร์ 8255 ทำงานร่วมกับ Z-80 CPU เนื่องจากมีจุดประสงค์ให้วงจรมีขนาดเล็กกะทัดรัด ใช้อุประกรณ์น้อยชิ้นและมีราคาประหยัด และที่สำคัญมีขายตามบ้านเรามากที่สุด การเขียนซอฟท์แวร์ก็ทำได้ง่าย

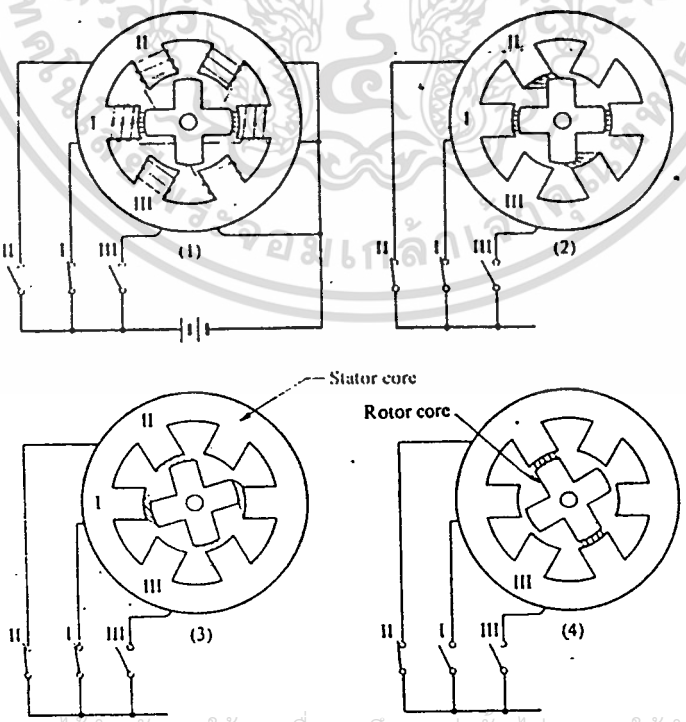
โครงงานนี้ใช้ สเตปปิ้งมอเตอร์ เป็นตัวขับ ซึ่งต่ออยู่กับ Ratio Gear เพื่อเพิ่มทอร์ค (Togue)ให้สูงพอที่จะใช้ในการขับตัวสายอากาศได้และเพิ่มความละเอียดความถูกต้อง(Accuracy) ในการปรับของสายอากาศอีกด้วย

บทที่ 2
ทฤษฎีที่เกี่ยวข้อง

2.1 ทฤษฎีสเตปปิ้งมอเตอร์

สเตปปิ้งมอเตอร์และลักษณะโดยทั่วไป

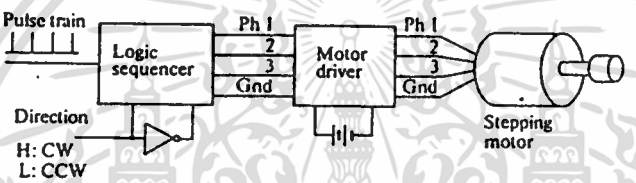
การทำงานของสเตปปิ้งมอเตอร์ ดูจากรูป 2.1 เป็นสเตปปิ้งมอเตอร์แบบวาริเอเบิลรีลัคแตนซ์ (Variable Reluctance) แกนเหล็กสเตเตอร์ (Stator Core) มีขี้ฟัน(Teeth) 6 ขี้ ขณะที่โรเตอร์ (Rotor) มีฟัน 4 ขี้ทั้งโรเตอร์และสเตเตอร์เป็นเหล็กอ่อน ขดลวด(Coil) 3 ชุด ถูกต่ออยู่ดังรูป แต่ละชุดขดลวดมี 2 ขดลวดต่ออนุกรมกัน เรียกแต่ละชุดว่าเฟส (Phase) และผลจากการต่อแบบนี้เรียกว่า มอเตอร์ 3 เฟส (3 Phase Motor) กระแสถูกจ่ายไปยังขดลวด แต่ละขดโดยผ่านสวิทช์I,I และIII ในสภาวะ(1) ชุดลวดของเฟส 1ได้รับกระแสโดยผ่านสวิทช์I หรือเฟส1 ถูกกระตุ้น เส้นแรงแม่เหล็กที่เกิดขึ้นในช่องอากาศ (Air Gap) เกิดขึ้นเนื่องจากการกระตุ้น ซึ่งแสดงด้วยลูกศร ในสภาวะนี้สเตเตอร์ 2 ขั้วของเฟส 1 จะอยู่ในแนวเดียวกับ 2 ขี้ที่อยู่ตรงข้ามกันของโรเตอร์ นี่เป็นสภาวะสมดุลย์ ซึ่งอยู่ในเทอมของไดนามิค(Dynamics) เมื่อสวิทช์II ปิดเพื่อกระตุ้นเฟส 2 กับเฟส 1เส้นแรงแม่เหล็กจะถูกสร้างขึ้นที่ขั้วของสเตเตอร์ของ



รูปที่ 2.1 วงจรกระตุ้นสเตปปิ้งมอเตอร์ 3 Phase

เฟส 2 ในลักษณะซึ่งแสดงในสภาวะ (2) ทอร์ค (Torgue) ทิศทางทวนเข็มนาฬิกาจะถูกสร้างขึ้นจากความเครียด (Tension) ในฟลักซ์แม่เหล็กเอียงไปยังแกนมอเตอร์ที่อยู่ใกล้ หลังจากนั้นโรเตอร์จะมาอยู่ในสภาวะ (3)

ดังนั้นโรเตอร์จะหมุนไปด้วยมุมการเปลี่ยนแปลงคงที่ ซึ่งเรียกว่ามุมสเตป(Step angle) ในที่นี้คือ 15 องศา ขณะที่สวิทช์จะมีการเปลี่ยนแปลงอีกครั้งหนึ่งคือ สวิทช์I จะถูกเปิด เพื่อลดพลังงานในเฟส 1โดยโรเตอร์จะหมุนไป 15 องศา มาอยู่ในสภาวะ(4) ตำแหน่งมุมของโรเตอร์จะถูกควบคุมโดยการเปิด-ปิดสวิทช์ ถ้าสวิทช์ถูกปิด-เปิดเป็นลำดับ โรเตอร์จะหมุนไปในลักษณะที่เป็นสเตป ความเร็วเฉลี่ยจะสามารถควบคุมได้ด้วยการเปิด-ปิดสวิทช์ดังแผนภาพรูปที่ 2.1



รูปที่ 2.2 ไดอะแกรมของระบบขับสเตปปิ้งมอเตอร์

จากที่กล่าวพอที่จะกล่าวถึงคุณสมบัติของสเตปปิ้งมอเตอร์ได้ว่า

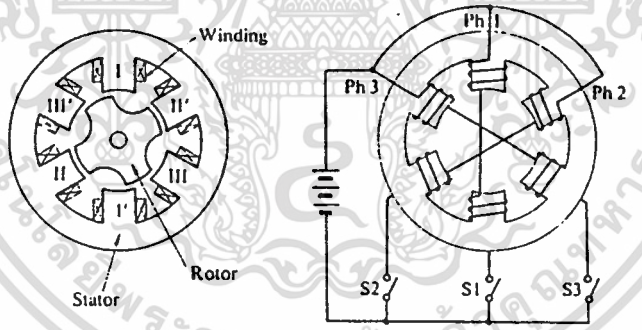
1. การหมุนของมอเตอร์จะเป็นสเตป (เป็นขั้นๆ) สเตปละกี่องศาขึ้นอยู่กับชนิดของมอเตอร์
2. ความเร็วในการหมุนขึ้นอยู่กับสัญญาณพลัซ์ที่ให้เข้ามาทางอินพุทของมอเตอร์(ความถี่)
3. ความผิดพลาดในการหมุน 1 สเตปมีค่าน้อยมาก แต่ต้องจำกัดอยู่ในความเร็วที่พอดีด้วย
4. คุณสมบัติของการตอบสนองสัญญาณในขณะที่มอเตอร์เริ่มทำงานและหยุดทำงานดีมาก
5. เนื่องจากไม่มีแปลงถ่าน (Commutator) เหมือนมอเตอร์ในระบบดีซี ดังนั้นจึงมีความแน่นอนในการทำงานสูง
6. แหล่งจ่ายแรงดันไฟที่ใช้ในการขับมอเตอร์มีค่าไม่มาก
7. ไม่ทำให้เกิดเสียงรบกวนและการสั่นได้ง่าย
8. ทำงานแบบ โอเพนลูป (Open loop)

สเตปปิ้งมอเตอร์ที่ใช้กันส่วนมากมี 3 ชนิด

1. วาริเอเบิลริลัคแทนซ์สเตปปิ้งมอเตอร์ (Variable Stapping Motor)
2. สเตปปิ้งมอเตอร์แบบแม่เหล็กถาวร (Permanent magnetic Stepping Motor)
3. สเตปปิ้งมอเตอร์แบบไฮบริด (Hybrid Stepping Motor)

2.1.1 วาริเอเบิลรีลัคแทนซ์สเตปปิ้งมอเตอร์

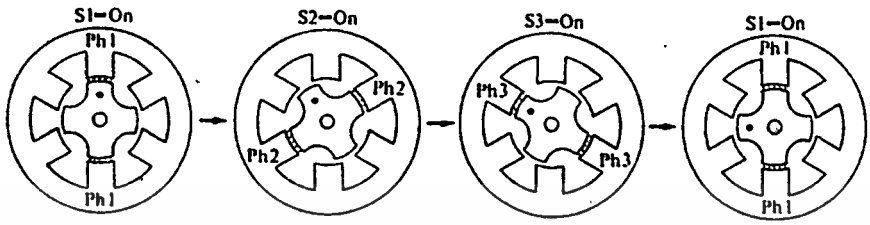
มอเตอร์แบบนี้โรเตอร์ทำด้วยเหล็กอ่อน ซึ่งค่าซึมซาบแม่เหล็ก (Permeability) สูง และสามารถให้ฟลักซ์แม่เหล็กผ่านได้มาก โดยโรเตอร์จะติดอยู่กับแกนมอเตอร์ และสเตเตอร์ติดอยู่กับโครงของตัวมอเตอร์ จากรูป 2.3 เป็นภาพตัดขวางของสเตปปิ้งมอเตอร์แบบนี้ ซึ่งเป็นมอเตอร์ 3 เฟส มีขั้วฟัน 6 ขั้ว ฟันของสเตเตอร์จะอยู่ตรงข้ามจะต่อกันเป็นอนุกรม หรือขนานก็ได้ (ในที่นี้ต่อแบบอนุกรม)

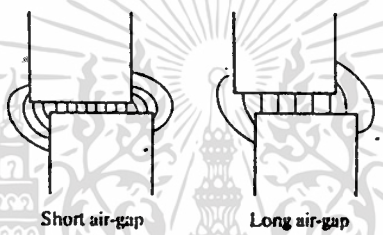


รูปที่ 2.3 โครงสร้างวาริเอเบิลรีลัคแทนซ์มอเตอร์

เราจะเห็นได้ว่าฟันของสเตเตอร์ 2 ขั้ว ที่มีเฟสเดียวกัน จะมีขั้วแม่เหล็กตรงข้ามกันและกันดังแสดงดังรูป สมมุติว่าฟัน I,II และIII มีขั้วเป็นขั้วเหนือ ฟัน I,II และIII จะเป็นขั้วใต้เมื่อถูกกระตุ้น

กระแสแต่ละเฟสจะถูกกระตุ้น ฟลักซ์แม่เหล็กก็จะเกิดดังรูป 2.4 ฟันของโรเตอร์ก็จะมีตำแหน่งในแนวเดียวกันกับฟันของสเตเตอร์ ซึ่งจะมีผลให้แมกเนติกรีลัคแทนซ์(Megnetic Reluctance) น้อยที่สุด สภาวะนี้คือตำแหน่งสมดุลย์





รูปที่ 2.4 ฟลักซ์แม่เหล็กที่เกิดขึ้น

<u>โครงสร้างเบื้องต้น ของมอเตอร์แบบนี้ จะมีลักษณะดังนี้</u>

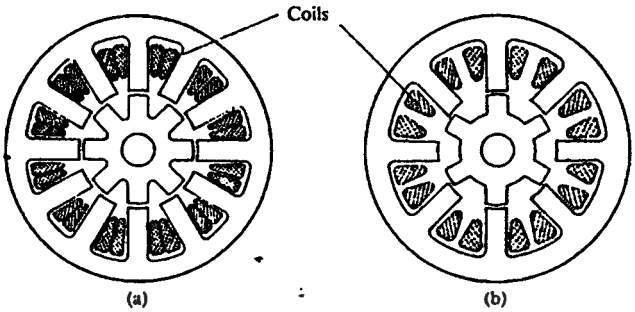
1. ช่องว่างอากาศควรจะเล็กที่สุด เท่าที่จะเป็นไปได้ ช่องว่างอากาศระหว่างฟันของโรเตอร์กับฟันของสเตเตอร์ควรมีค่าความห่างกันน้อยมาก เพื่อที่จะได้ทอร์คสูง และตำแหน่งที่แน่นอนขึ้น

2. สำหรับมุมสเตปเล็กๆ จากรูป 2.5a แสดง 3 เฟสมอเตอร์ที่สเตเตอร์มีฟัน 12 ชี้ และฟันมีโรเตอร์ 8 ชี้ รูป 2.5b เป็นรูป 4 เฟส มอเตอร์ที่สเตเตอร์มีฟัน 8 ชี้และโรเตอร์มีฟัน 6 ชี้ ซึ่งทั้งสองรูปนี้มีมุมสเตปเท่ากับ 15 องศา

ความสัมพันธ์ระหว่างมุมสเตป θs, จำนวนเฟส m, โดยจำนวนฟันของโรเตอร์ Nr และจำนวนสเตปใน 1 รอบ S จะหาได้โดย

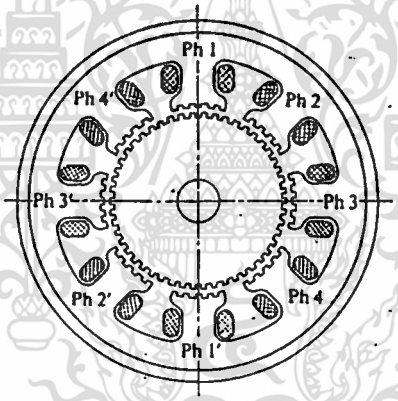
$$S = 360/\theta s = m(Nr)$$

นั่นคือ เราสามารถลดมุมสเตปลงได้โดยเพิ่มจำนวนฟันบนโรเตอร์ ดังแสดงในรูป 2.6



รูปที่ 2.5 โรเตอร์และสเตเตอร์

รูป 2.6 หน้าตัดของ 4 เฟส มอเตอร์มีฟันโรเตอร์ 50 ชี่ มุมสเตป 1.8 องศา

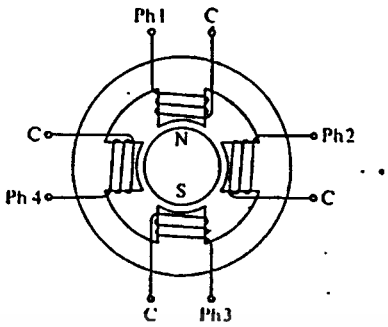


รูปที่ 2.6 หน้าตัดของสเตปปิ้งมอเตอร์ 4 เฟส

2.1.2 สเตปปิ้งมอเตอร์แบบแม่เหล็กถาวร

สเตปปิ้งมอเตอร์แบบนี้ใช้แม่เหล็กแบบถาวร รูปที่ 2.7 เป็นตัวอย่างของสเตปปิ้งมอเตอร์แบบถาวรแบบ 4 เฟส โรเตอร์เป็นทรงกระบอก สเตเตอร์มีฟัน 4 ชี่ โดยที่แต่ละชี่มีขดลวดพันรอบ ถ้าจำนวนชี่บนสเตเตอร์และขั้วแม่เหล็กบนโรเตอร์เพิ่มขึ้นเป็น 2เท่า มุมแต่ละสเตปลดลงจากเดิมครึ่งหนึ่ง ดังนั้นเพื่อที่จะลดมุมสเตปลงไปอีก ในสเตปปิ้งมอเตอร์แบบแม่เหล็กถาวรจะต้องเพิ่มจำนวนแม่เหล็กและชี่ฟัน อย่างไรก็ตาม ขั้วแม่เหล็กที่จะเพิ่มขึ้นได้นั้นมีจำนวนจำกัด

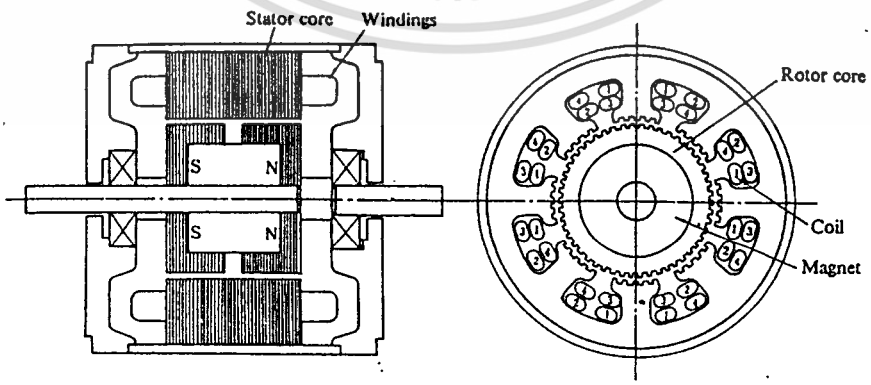
ลักษณะทั่วไปของมอเตอร์แบบนี้ก็คือ โรเตอร์จะถูกยึดอยู่กับที่ แม้ว่าไม่มีการกระตุ้นเฟส ลักษณะเช่นนี้เรียกว่า ดีเทนท์ แมคคานิคชั่ม (DETENT MECANISM)



รูปที่ 2.7 โครงสร้างสเตปปิ้งมอเตอร์ 4 เฟส

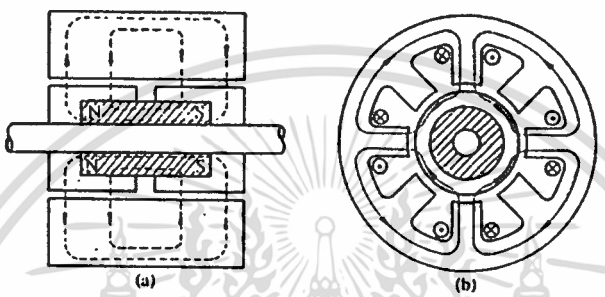
2.1.3 สเตปปิ้งมอเตอร์แบบไฮบริด

เป็นสเตปปิ้งมอเตอร์แบบหนึ่ง ที่มีโรเตอร์เป็นแบบแม่เหล็กถาวรการใช้ชื่อไฮบริดได้มาจากการรวมหลักสำคัญของมอเตอร์ แบบแม่เหล็กถาวรและแบบวาริเอเบิลรีลัคแทนซ์ โครงสร้างแกนของสเตเตอร์จะคล้ายกับแบบ วาริเอรีลัคแทนซ์ แต่การพันและการต่อขดลวดจะต่างจากแบบวาริเอเบิลรีลัคแทนซ์มอเตอร์ ซึ่งมีเพียง 1ขดจาก 2ขด ของ 1เฟส ที่ถูกพันบนขั้วเดียว ในขณะที่ 4เฟส ไฮบริดมอเตอร์ ชุดขดลวด 2เฟส ที่แตกต่างกันจะถูกพันบนขั้วเดียวกันดังรูป 2.8 เพราะฉะนั้น ขั้วหนึ่งจะไม่มีเพียงเฟสเดียว ขดลวด 2 ขด จะถูกพันเป็นขั้วเดียวกันแบบ ไบฟิลา (Bifilar Winding) ซึ่งจะทำให้ขั้วแม่เหล็กต่างกันขณะมีการกระตุ้น



รูปที่ 2.8 โครงสร้างของสเตปปิ้งมอเตอร์แบบไฮบริด

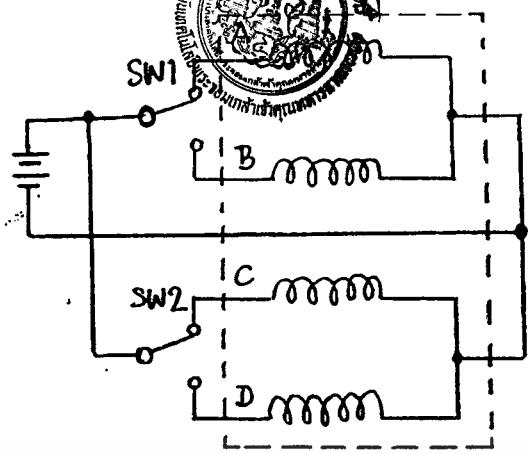
ลักษณะที่สำคัญอีกอย่างหนึ่ง ของไฮบริดมอเตอร์คือ โรเตอร์นั้น จะเป็นแม่เหล็กรูปร่างทรงกระบอกอยู่ในแกนเหล็กของโรเตอร์ มันถูกทำให้เป็นแม่เหล็กตามยาวเพื่อสร้างสนามขั้วเดียว ดังรูปที่ 2.9 แต่ละขั้วของแม่เหล็กจะถูกล้อมรอบด้วยฟันเหล็กอ่อน ซึ่งฟันของโรเตอร์กับสเตเตอร์อยู่เหลื่อมกันอยู่ครึ่งช่วงฟัน



รูปที่ 2.9 การวางแม่เหล็กตามยาวเพื่อสร้างสนามขั้วเดียวกัน

การทำงานพื้นฐาน

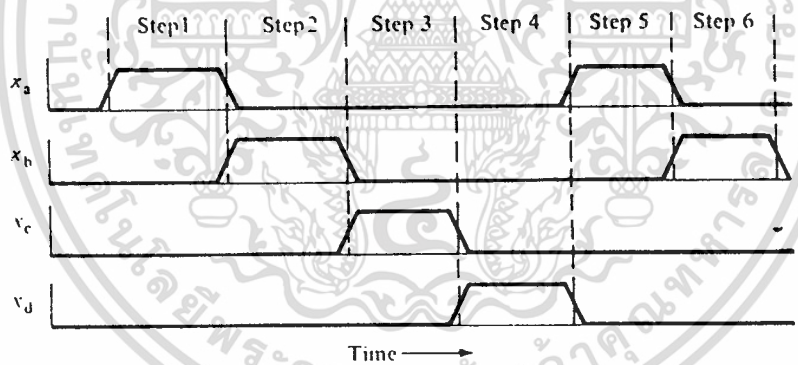
สเตปปิ้งมอเตอร์เป็นมอเตอร์ที่ถูกใช้ทำงานโดยสัญญาณอินพุทที่เป็นพัลซ์ ทุกๆ สัญญาณพัลซ์ที่ให้เข้ามาจะให้การเปลี่ยนแปลงสถานะสนามแม่เหล็กไฟฟ้าที่เกิดขึ้น และให้การหมุนของมอเตอร์เป็นมุมที่คงที่ ซึ่งจะแตกต่างกับการหมุนของมอเตอร์แบบธรรมดาในระบบควบคุมที่ใช้สัญญาณดิจิตอลนี้ เช่น ในการส่งข้อมูลการควบคุมข้อมูลโดยที่ เป็นปริมาณค่าของสัญญาณดิจิตอลทั้งหมด จึงทำให้การควบคุมการทำงานของสเตปปิ้งมอเตอร์โดยตรงได้เป็นอย่างดี การทำงานพื้นฐานของสเตปปิ้งมอเตอร์แสดงในรูปที่ 2.10 ซึ่งแสดงบล็อกของวงจรขับสเตปปิ้งมอเตอร์ ทำให้กระแสไฟ ดีซี เรียงลำดับเข้าไปตามเฟสต่างๆ ของมอเตอร์หมุนไปตามองศาที่กำหนดไว้การที่จะทำให้กระแสไฟ ดีซี เรียงลำดับเข้าไปที่มอเตอร์ได้ ทำได้ด้วยการใช้งานของสวิทช์ และเพื่อให้มอเตอร์หมุนจำเป็นต้องมีวงจรขับสเตปปิ้งมอเตอร์ที่จะควบคุมการไหลของกระแสไฟ คือสวิทช์ 1 และสวิทช์ 2 ถ้าให้การทำงานของสวิทช์ตามตารางการทำงาน ที่แสดงไว้ในรูปที่ 2.11 ถ้าให้การทำงานโดยใช้รีเรย์(Relay) หรือมือโยก สเตปมอเตอร์ก็จะหมุนเหมือนกันแต่หมุนช้า เนื่องจากรีเลย์หรือมือโยกให้การเปลี่ยนแปลงของสวิทช์ช้า ซึ่งจะทำได้เร็วเท่ากับความถี่สูงสุดของสัญญาณพัลซ์ที่มอเตอร์ทำงานสามารถทำงานได้ (เป็นจำนวนพัลซ์ต่อวินาที)



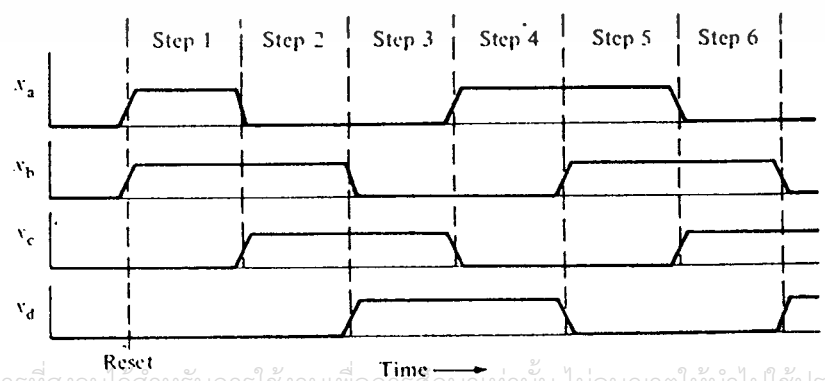
รูปที่ 2.10 การทำงานพื้นฐานของสเตปปิ้งมอเตอร์

วงจรขับมอเตอร์มีหลายชนิด ขึ้นอยู่กับชนิดสเตปปิ้งมอเตอร์ที่มีอยู่ 3 ระบบคือ

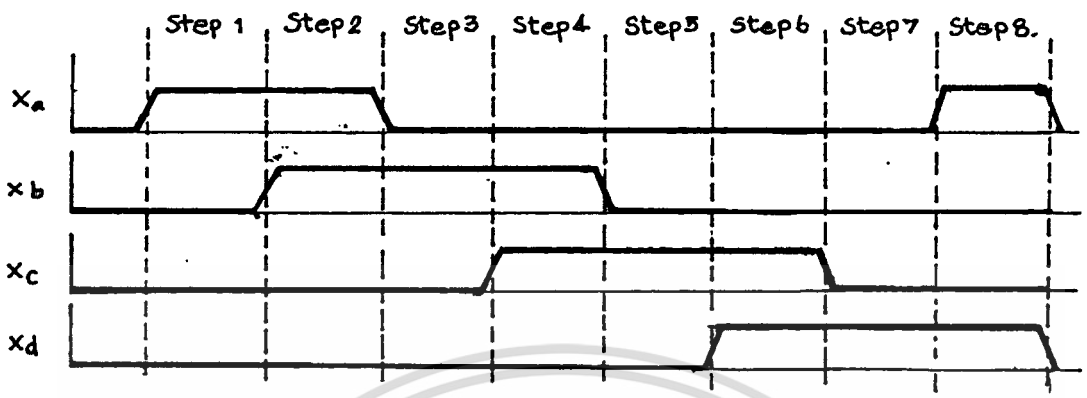
1. ระบบการกระตุ้นสนามแม่เหล็กเฟสเดียว (Single Phase Excitation)
2. ระบบการกระตุ้นสนามแม่เหล็ก 2 เฟส (2-Phase Excitation)
3. ระบบการกระตุ้นสนามแม่เหล็ก 1-2 เฟส (Half-Step Excitation)



Single Phase Excitation



2-Phase Excitation

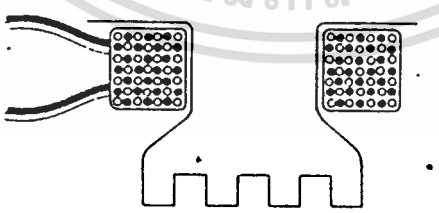


Haft-Step Excitation

รูปที่ 2.11 ชนิดของการขับสเตปปิ้งมอเตอร์

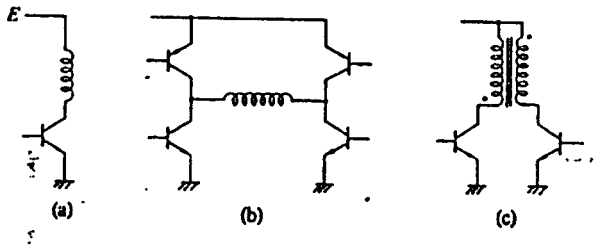
การพันลวดแบบโมโนฟิลาและไบฟิลา (Monofilar and Bifilar Winding)

แบบโมโนฟิลา มีการพันลวดเส้นเดียว ส่วนแบบไบฟิลา เส้นลวดทั้ง 2 เส้นนี้จะถูกพันเหมือนกับเส้นเดียวกันขั้วดังรูปที่ 2.12 และขดลวดทั้งสองเส้นนี้จะแยกกันที่ปลายเป็นลักษณะ 2 เส้นแยกกัน ถ้าขดลวดเป็นของเฟส 1 อีกขดหนึ่งเป็นของเฟส 3 และทำนองเดียวกัน ถ้าขดหนึ่งเป็นของเฟส 2 หนึ่งของเฟส 4 (กรณีมอเตอร์ 4 เฟส)



รูปที่ 2.12 การพันแบบไบฟิลา

จุดประสงค์ของการพันแบบไบฟิลาก็คือ เพื่อที่จะให้พลังงานกับขั้วแม่เหล็กสเตเตอร์โดยการสลับขั้วแม่เหล็ก การกระตุ้นแต่ละเฟสอาจเป็นแบบใดแบบหนึ่งใน 3 แบบในรูป



รูปที่ 2.13 การพันแบบโมโนฟิลา

ในวงจรรูปที่ 2.13a เป็นโมโนฟิลา ขั้วแม่เหล็กจะถูกกระตุ้นเป็นขั้วเหนือขั้วใต้เสมอ ซึ่งแสดงว่าไม่สามารถลับขั้วแม่เหล็กได้ การกระตุ้นแบบนี้เป็นการกระตุ้นแบบขั้วเดียว(Unipolar Exited)

ในวงจรรูปที่ 2.13b ทิศทางของกระแสในขดสามารถสลับได้เนื่องจากเป็นวงจรบริดจ์ (Bridge Exited)

อย่างไรก็ตามจะต้องใช้ทรานซิสเตอร์ถึง 4 ตัวต่อขดลวด 1 ขด แบบนี้จะทำให้ทอร์คที่ความเร็วต่ำมาก กว่าแบบไบฟิลา แต่ก็มีโอกาสที่ทรานซิสเตอร์จะพังได้เนื่องจากจัดเวลาผิดผลาด

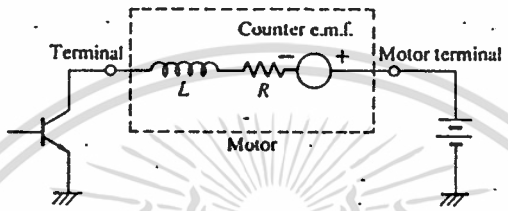
ในวงจรรูปที่ 2.13C เกี่ยวกับคู่สายไบฟิลา และทรานซิสเตอร์ 2ตัว โดยทำให้สเตเตอร์ถูกกระตุ้นเป็นขั้วแบบไบฟิลา ก็จะเกิดสนามแม่เหล็กคัปปลิ้ง(Coupling) เมื่อขั้วใดขั้วหนึ่งถูกกระตุ้น ถ้าแทนการพันแบบไบฟิลา ด้วยเส้นลวด 2เส้น ที่แยกจากกันความแตกต่างของอินดัคแทนซ์ (Inductance) จะปรากฏระหว่าง ขด 2ขด ทำให้ตำแหน่งผิดไป

โดยทั่วไปประสิทธิภาพของมอเตอร์แบบแม่เหล็กถาวรกับแบบไฮบริด ที่ใช้แบบฟิลานี้จะได้ประสิทธิภาพดีกว่าแบบ โมโนฟิลา

2.1.4 วงจรขับมอเตอร์ (MOTOR DRIVE CIRCUIT)

จุดประสงค์ของวงจรขับคือ เพื่อที่จะให้ได้โวลท์เตจ และกระแสที่ถูกต้องไปยังมอเตอร์ในช่วงเวลาที่สั้น และลักษณะที่มีประสิทธิภาพ ทิศทางของกระแสในขดลวดของมอเตอร์แบบแม่เหล็กถาวรและแบบไฮบริดมีความสำคัญ เพราะจะต้องใช้ในทิศทางที่เหมาะสมในเวลาต่างๆ สำหรับการขับมอเตอร์แบบวาริเอเบิลรีลัคแทนซ์ จะไม่ขึ้นกับทิศทางของกระแสที่ไหลในขดลวด เนื่องจากไม่มีแม่เหล็กอยู่ในตัวมันเลย

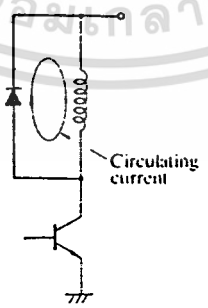
วงจรขับมอเตอร์จะรับสัญญาณควบคุมมาจากวงจรจัดลำดับลอจิก ในการสร้างวงจรขับมอเตอร์จะมีปัญหาเกี่ยวกับ อินดัคแทนซ์และรีซิสแทนซ์ (Resistance) อนุกรมกันในวงจรสมมูลย์ของมอเตอร์ ดังรูปที่ 2.14 อีกทั้งในการหมุนยังมีปัญหาเพิ่มขึ้นเกี่ยวกับแรงดันไฟกลับ (BACK ELECTROMOTIVE FORCE : BACK EMF.) นอกจากนี้ยังต้องคำนึงถึงแหล่งจ่ายไฟ ดีซี และการใช้กับการป้องกัน Power ทรานซิสเตอร์ด้วย



รูปที่ 2.14 วงจรสมมูลย์ของสเตปปิ้งมอเตอร์

จากรูปที่ 2.14 เมื่อทรานซิสเตอร์หยุดทำงานจะเกิดสไปค์โวลท์เตจ(Spike Voltage) เนื่องจากขดลวด (ตัวเหนี่ยวนำ) ซึ่งอาจทำให้ทรานซิสเตอร์พังได้ จึงต้องมีวงจรป้องกันและลดลักษณะแบบนี้อย่างรวดเร็ว โดยวิธีต่อไปนี้

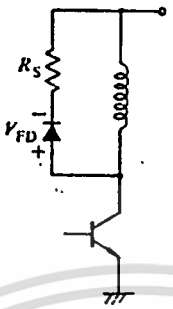
1. ลดโดยใช้ไดโอด (DIODE SUPPRESSOR) ใช้ไดโอดต่อคร่อมขนานกับขดลวดดังรูปที่ 2.15 กระแสจะไหลอยู่ในวงจรขดลวดกับไดโอด หลังจากทรานซิสเตอร์หยุดทำงาน ซึ่งกระแสนี้จะลดลงไปตามเวลา แต่ก็ใช้เวลานานมากจึงยังเกิดทอร์คยิดอยู่



รูปที่ 2.15 วงจร Diode Suppressor

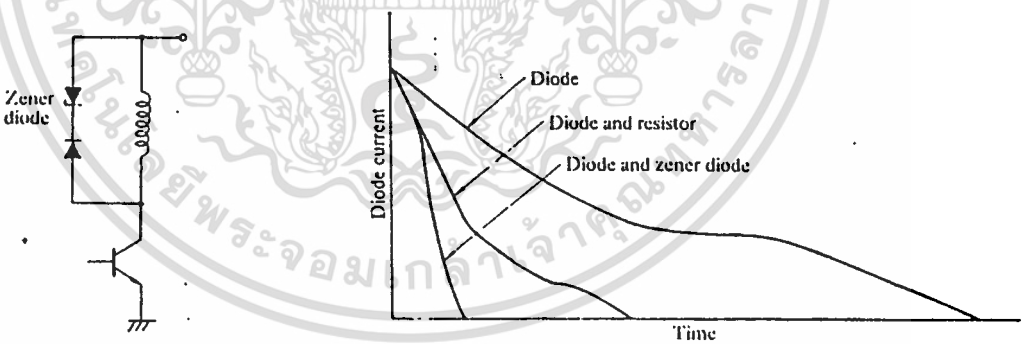
2. โดยใช้ไดโอดและตัวต้านทาน (DIODE AND RESISTOR SUPPRESSOR) โดยการต่อความต้านทานอนุกรมกับไดโอดแล้วต่อขนานคร่อมขดลวด ดังรูปที่ 2.16 วิธีนี้จะช่วยลดเวลา

ในการที่จะทำให้กระแสหยุดไหลวน ยิ่งใช้ตัวต้านทานค่ามาก กระแสก็จะหยุดไหลเร็วขึ้น แต่ทรานซิสเตอร์ต้องทนโวลท์เตจคร่อม คอลเลคเตอร์-อีมิเตอร์ เพิ่มขึ้นด้วย



รูปที่ 2.16 วงจร Diode/Resistor Suppressor

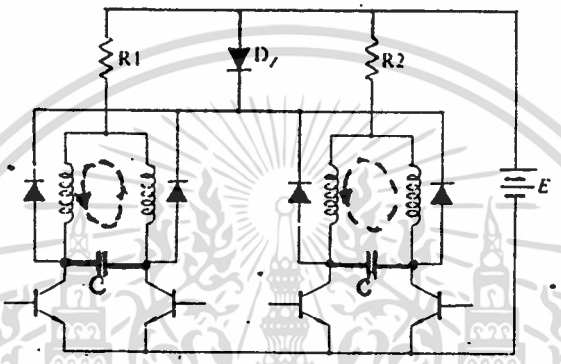
3. โดยใช้ซีเนอร์ไดโอด (ZENER DIODE SUPPRESSOR) โดยการต่อซีเนอร์ไดโอดดังรูปที่ 2.17 การต่อแบบนี้เมื่อเปรียบเทียบกับ 2 แบบแรก แล้วจะมีประสิทธิภาพมากกว่า ซึ่งสามารถลดกระแสที่เกิดขึ้นได้เร็วกว่า



รูปที่ 2.17 วงจร Zener Diode Suppressor

4. โดยใช้คอนเดนเซอร์ (CONDENSER SUPPRESSOR) วิธีนี้มักจะใช้กับสเตปปิ้งมอเตอร์แบบไบฟิลา การต่อวงจรดังรูป 2.18 ของมอเตอร์ 4เฟส ตัวคอนเดนเซอร์ที่ต่ออยู่ระหว่างเฟส 1กับเฟส 3 และเฟส 2กับเฟส 4 จะทำหน้าที่ดังนี้ ในตอนที่ทรานซิสเตอร์ T1 หยุดทำงาน ในการทำงานแบบกระตุ้นเฟสเดียว T2 หรือ T4 เริ่มทำงาน แต่ T3 ยังไม่เริ่มทำงาน เนื่องจากเฟส 1 เฟส 3 ต่ออยู่ในลักษณะ ไบฟิลา กระแสที่เกิดขึ้นหลังจากที่ T1 หยุดทำงานจะไหลวนไป

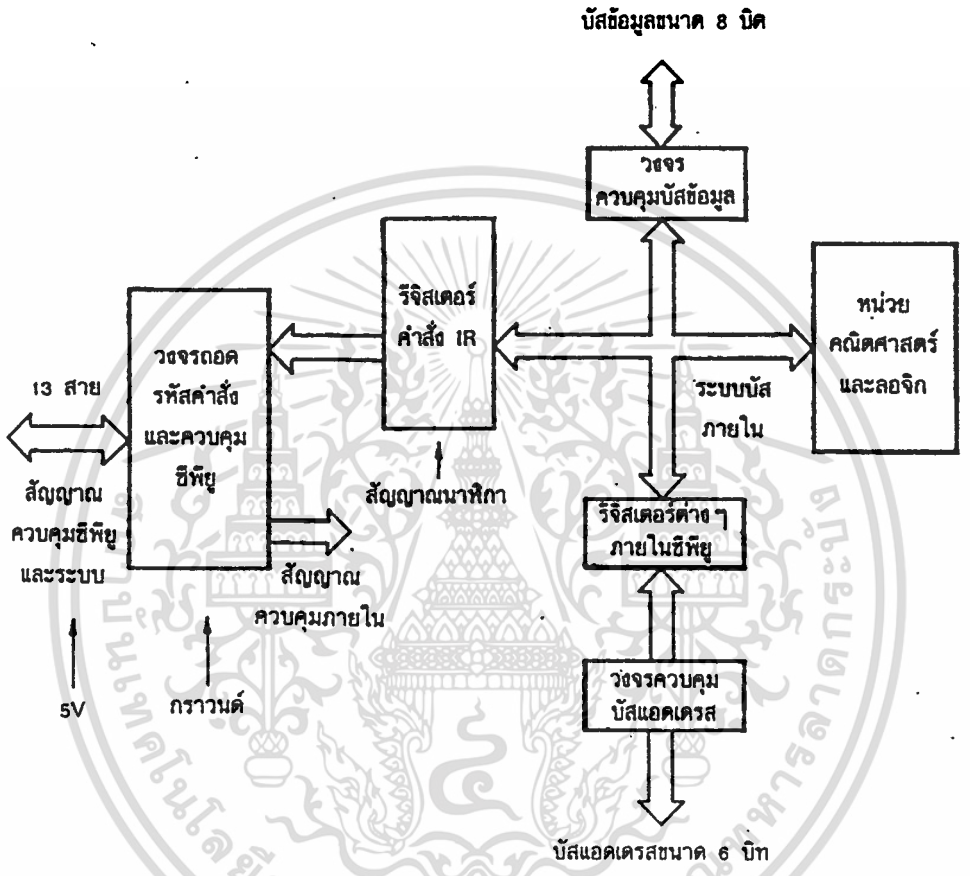
ตามเส้นไข่ปลาดังแสดงในรูป ถ้า T3หรือ T4 เริ่มทำงานแต่ T3 ยังไม่เริ่มทำงาน ประจุที่ชาร์จ (Charge) เก็บไว้ในคอนเดนเซอร์จะช่วยเพิ่มทอร์คให้ โดยการปล่อยกระแสไหลผ่านเฟส 1 ทั้งนี้การต่อคอนเดนเซอร์สามารถใช้กับการกระตุ้นแบบ 2เฟสได้ด้วย สำหรับค่า R1,R2 ใส่ไว้เพื่อปรับกระแสให้เหมาะสม นอกจากนั้นคอนเดนเซอร์ยังช่วยลดการสั่นของมอเตอร์ โดยเปลี่ยนพลังงานกลเป็นพลังงานความร้อนแทน



รูปที่ 2.18 วงจร Condenser Suppressor

2.2 ลักษณะโครงสร้างของไมโครโปรเซสเซอร์ เบอร์ Z-80

Z-80 เป็นไมโครโปรเซสเซอร์ ที่พัฒนามาจากไมโครโปรเซอร์เดิมคือ เบอร์ 8080 ฉะนั้นลักษณะของโครงสร้าง, คำสั่งการใช้งาน จึงคล้ายกันหากแต่ว่า Z-80 ได้มีความสามารถมากขึ้นหลายอย่าง ทั้งในด้านของคำสั่งและการอินเตอร์รันท์เป็นต้น



รูปที่ 2.19 ลักษณะโครงสร้างภายในของ ซีพียู Z-80

ดังที่ได้กล่าวมาแล้วในข้างต้นแล้วว่าตัว ซีพียู จะมีส่วนประกอบใหญ่ๆ คือ หน่วยควบคุมและหน่วยคำนวณอยู่ภายในรูปที่ 2.19 เป็นโครงสร้างของ ซีพียู Z-80 จะเห็นว่าจะส่วนของรีจีสเตอร์ที่ทำหน้าที่เป็นหน่วยความจำชั่วคราวย่อยๆ เพิ่มขึ้นมาเพื่อให้หน่วยความจำได้ใช้งาน การเชื่อมต่อระหว่างบล็อกภายใน ตัวซีพียู เป็นขนาด 8 บิท ที่คิดเวลาของการเคลื่อนย้ายข้อมูลภายในเป็นศูนย์ (คือไม่เสียเวลาเลย)

การทำงานอย่างคร่าวๆ ของตัวซีพียู ที่แสดงดังรูปที่ 2.19 เป็นดังนี้

ส่วนควบคุมซีพียู จะสั่งให้ส่วนวงจรควบคุมบัสแอดเดรสให้สัญญาณการอ้างแอดเดรสออกไปสู่หน่วยความจำภายนอก เพื่อสั่งให้ที่เก็บในหน่วยความจำนั้นเข้ามาสู่วงจรควบคุมบัสข้อมูลแล้ว

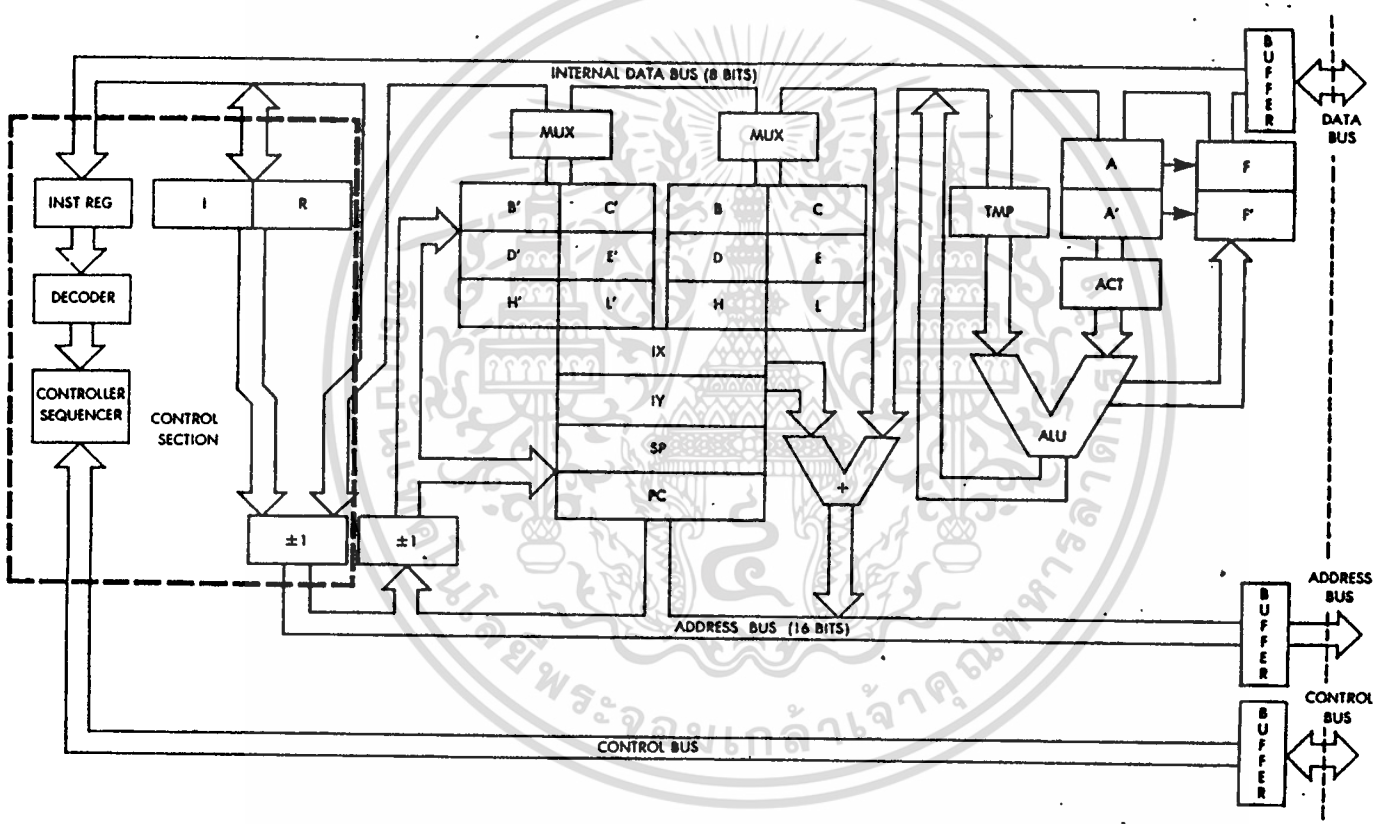
เข้าสู่รีจีสเตอร์คำสั่ง IR รอส่งต่อไปถอดระหัสคำสั่งที่วงจรถอดระหัสคำสั่งและควบคุมซีพียู หลังจากถอดระหัสแล้ว ก็จะมีสัญญาณควบคุมออกมาจากส่วนควบคุม เพื่อทำตามคำสั่งที่ถอดรหัสได้นั้น เช่น หากเป็นการคำนวณ ส่วนของหน่วยคำนวณก็จะทำการคำนวณโดยใช้ความจำชั่วคราว คือรีจีสเตอร์ต่างๆ ช่วยคำนวณ แล้วจึงส่งผลที่ได้ออกไปทางบัสข้อมูลต่อไป (จะให้ผลการทำงานเป็นแบบไหนก็ขึ้นอยู่กับคำสั่งที่ถอดระหัสได้)

2.2.1 โครงสร้างที่ละเอียดขึ้นและทำงานภายใน ในคำสั่งบางอย่าง

ที่ได้กล่าวมาข้างต้นเป็น ลักษณะบล็อกของโครงสร้างภายใน Z-80 ทั่วไป แต่ในส่วนประกอบภายในที่ละเอียดขึ้นจะแสดงดังรูปที่ 2.20 ซึ่งจะเห็นเส้นทางบัสข้อมูลและส่วนประกอบของรีจีสเตอร์ต่างๆ อย่างชัดเจนยิ่งขึ้น ซึ่งจะได้กล่าวถึงความสัมพันธ์ของส่วนประกอบภายในนี้อีกครั้งหนึ่ง

ในรูปที่ 2.20 ALU จะมีลักษณะเป็นตัว "V" และมีอินพุทที่ ACT และ TEMP ซึ่งทำหน้าที่เป็นที่พักข้อมูลหรือ Temporary register ดังได้กล่าวมาแล้วส่วน Accumulator และflag คือรีจีสเตอร์ในรูป A',F'ในรูปด้านขวามือบนตามลำดับ และมีรีจีสเตอร์สำรองให้อีก คือ A',F' ซึ่งมีขนาดเท่ากับ A,F แต่ถูกใช้งานเพียงเก็บข้อมูลชั่วคราวให้แก่ A,F เท่านั้น กรณี A,F จำเป็นต้องถูกนำไปใช้งานอย่างอื่นก่อน ส่วนตรงกลางภาพเป็นตารางรีจีสเตอร์ที่มีขนาดทั้ง 8 บิท และ 16 บิท ซึ่งจะเห็นได้ว่า Z-80 ได้ให้จำนวนรีจีสเตอร์มากมายให้ใช้ได้อย่างเพียงพอ โดยขนาด 8 บิทนั้น นอกจากแอคคิวมูเลอร์และแฟลกแล้ว ยังมีรีจีสเตอร์ B,C,D,E,H,L และชุดสำรองเพื่อเก็บข้อมูลชุดจริงชั่วคราวอีกคือ B',C',D',E',H',L'(ในส่วนของซอฟแวร์จะมีคำสั่งที่ใช้เคลื่อนย้ายข้อมูลระหว่างชุดจริงและชุดสำรอง) ในรูปไดอะแกรมจะเห็นได้ว่ารีจีสเตอร์ขนาด 8 บิทนี้ได้ถูกต่อเข้ากับสายสัญญาณแอดเดรสบัสร่วมกับรีจีสเตอร์ขนาด 16 บิท และรีจีสเตอร์ขนาด 8 บิทนี้จึงมีความสามารถนำคู่ของมันในแต่ละคู่เช่น BC,DE มาต่อให้ข้อมูลรวมกันได้ 16 บิท เพื่อใช้ในการอ้างแอดเดรสได้ด้วย (เอ้าท์พุทที่สัญญาณแอดเดรสบัส) ฉะนั้นรีจีสเตอร์คู่ 8 บิทเหล่านี้จึงมีความสามารถเพิ่มขึ้นอีกนอกจากการคำนวณเท่านั้น

เนื่องจากว่าภายในมีการเชื่อมต่อบัสเพียงกลุ่มเดียวดังนั้น ในการย้ายข้อมูลเข้าออกกับกลุ่มของตารางรีจีสเตอร์ที่มีทั้ง 8 บิทและ 16 บิทนี้ จึงต้องมีส่วนของ "MUX" เพื่อมัลติเพล็กสัญญาณให้เข้าสู่รีจีสเตอร์ให้ถูกต้องตำแหน่งตัว (ไบท์สูงหรือต่ำในรีจีสเตอร์ขนาด 16 บิท (IX,IY, SP,PC)



รูปที่ 2.20 แสดงโครงสร้างภายในของ Z-80

IX,IY REGISTER เรียกว่า "INDEX REGISTER" ใช้ในการเก็บค่าของแอดเดรสหน่วยความจำหลักที่ใดที่หนึ่งเพื่อเก็บไว้อ้างอิงถึงตำแหน่งของหน่วยความจำอื่นๆ ได้ ในรูปจะเห็นว่า ได้มีเอ้าท์พุทที่นำไปบวกเพิ่มกับข้อมูลในบัสข้อมูลได้ นั่นหมายความว่าค่าของ IX,IY ที่ส่งออกไปบนแอดเดรสบัสสามารถที่จะเปลี่ยนแปลงได้ แล้วแต่ผู้ใช้จะกำหนดข้อมูลที่นำมาบวก (แต่ในค่าของ IX,IY ไม่เปลี่ยนแปลง) จึงสะดวกในการอ้างแอดเดรสหลายพื้นที่เมื่อใช้ IX,IY เป็นแอดเดรสหลัก (Index) ค่าที่คำนวณเข้านี้เรียกว่า "Displacement"

เครื่องหมาย + ,-1 ที่ย้อนจากสายแอดเดรสบัสเข้าสู่ตัวรีจีสเตอร์ต่างๆ นั้น เป็นการบอกว่าค่าของรีจีสต่างๆ นี้ เมื่อนำมาใช้ในการกำหนดแอดเดรสแล้วสามารถป้อนกลับมาเพื่อหรือลดค่าของมันได้อีก อันนี้จะมีประโยชน์มากในการทำคำสั่งพวกวนลูป (loop) เช่น การย้ายข้อมูลเป็นกลุ่มเป็นต้น

I register เรียกว่า "Interrupt page address register" ซึ่งใช้ในแอดเดรสไบท์สูงของการอินเทอร์รัพท์ โดยการใช้งานนำไปรวมกับค่าของแอดเดรสไบท์ต่ำที่ส่งมาจากอุปกรณ์อินเตอร์รัพท์ให้สังเกตุในรูปว่า จะมีสัญญาณข้อมูลจากบัสมารวม (บล็อก +/-) ก่อนออกสู่แอดเดรสบัส (ใช้ในโหมด 2 ของอินเทอร์รัพท์ Z-80)

R register คือ Memory-refresh register ใช้ในเก็บค่าของแอดเดรสที่จะถูกรีเฟรสของหน่วยความจำแบบไดนามิก (Dynamic RAM) ซึ่ง R register จะทำให้ค่าของแอดเดรสออกมาที่เพิ่มค่าโดยอัตโนมัติทุกครั้งที่อยู่ในช่วงแมคชีนไซเกิ้ลของการเฟทช์คำสั่ง

INST register เป็นรีจีสเตอร์ที่ใช้เก็บข้อมูลคำสั่งที่ถูกเฟทช์เข้ามาในช่วงของแมทชีนเฟทช์คำสั่ง เพื่อรอคำสั่งต่อไปให้กับส่วนวงจรถอดระหัสคำสั่งต่อไป

DECODE เป็นวงจรที่ใช้ในการถอดระหัสคำสั่งที่เก็บใน INST register จากนั้นส่งสัญญาณให้ส่วนวงจร Control Sequencer ทำสัญญาณออกมาควบคุมอุปกรณ์ต่างๆ เพื่อให้ทำงานตามคำสั่งนั้นๆ

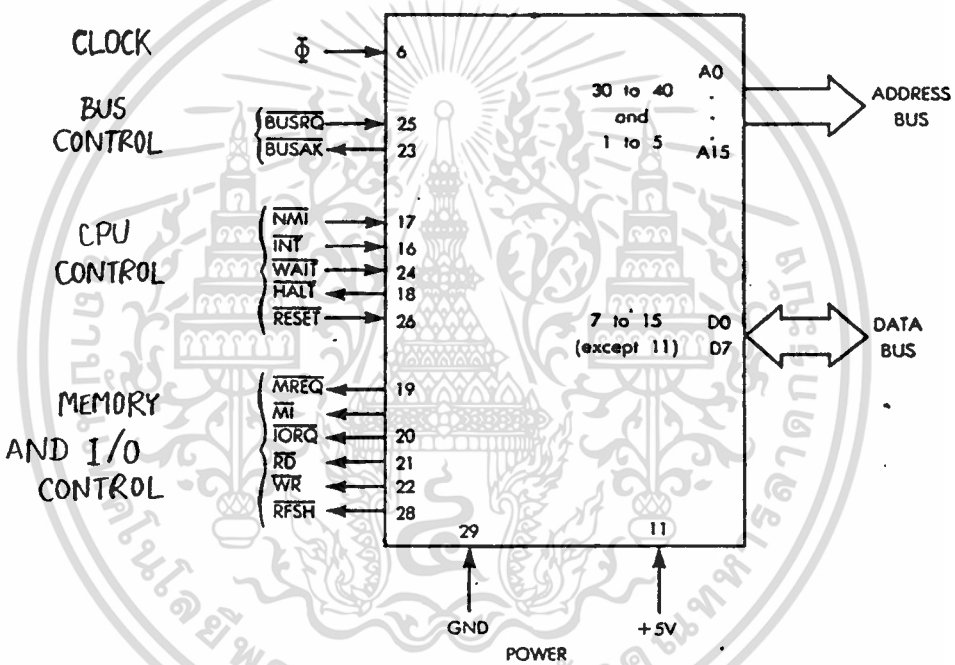
Program Counter (PC) เป็นรีจีสเตอร์ 16 บิท ที่ใช้ในการชี้ค่าแอดเดรสหน่วยความจำเพื่อทำงานตามลำดับโปรแกรม โดย PC จะเพิ่มค่าตัวเองอัตโนมัติ ดังที่กล่าวมาแล้ว

Stack pointer (SP) เป็นรีจีสเตอร์ 16 บิท ที่ใช้ในการชี้ค่าแอดเดรสของพื้นที่หน่วยความจำที่กำหนดเป็นพื้นที่ของ Stack ค่าของแอดเดรสสามารถเพิ่มหรือลดได้โดยอัตโนมัติ

จากรูปที่ 2.20 จะเห็นว่า เอ้าท์พุทจากโครงสร้างภายในสู่ภายนอกเป็นชุดของสัญญาณที่มีด้วยกัน 3ชุดคือ ชุดของสัญญาณควบคุม ชุดของสัญญาณแอดเดรสหน่วยความจำชุดของสัญญาณข้อมูล

2.2.2 การจัดขาและความหมาย

ตัวชิฟ ซีพียู เป็นลักษณะตัวแบนสีดำที่มีขนาด 40ขา ซึ่งแยกออกเป็นส่วนต่างๆ คือ บัสข้อมูลขนาด 8 บิท ซึ่งเป็นบัสชนิด 2 ทิศทาง คือ สามารถวิ่งเข้าออกจากตัว ซีพียูได้ และบัสแอดเดรสที่มีขนาด 16 บิท ทำให้สามารถอ้างแอดเดรสได้ถึง 64Kbyte และบัสแอดเดรสนี้ยังมีส่วนใช้ในการอ้างอิงถึง แอดเดรสของพอร์ทอินพุท เอ้าท์พุทด้วย และอีกกลุ่มคือ กลุ่มของสัญญาณควบคุมการทำงาน ซึ่งจะมีทั้งหมด 13 สาย ที่เหลือเป็นขาสัญญาณนาฬิกาและแรงดันไฟเลี้ยง



รูปที่ 2.21 แสดงการจัดขาของ Z-80

รายละเอียดและหน้าที่ที่สำคัญของสายสัญญาณต่างๆ มีดังนี้

A0-A15 เป็นกลุ่มของสายบัสขนาด 16 เส้น มีลักษณะเป็นลอจิกสามสถานะ และจะมีสัญญาณที่ใช้แยกว่าเวลาใดจะเป็นการอ้างแอดเดรสของบัสหรือแอดเดรสของอินพุท เอ้าท์พุท ทั้งนี้เพราะการใช้สัญญาณที่ร่วมกัน

D0-D7 เป็นกลุ่มสายสัญญาณขนาด 8 เส้น ลักษณะของสายสัญญาณเป็นแบบสามสถานะเช่นกัน ข้อมูลสามารถที่จะวิ่งได้มองทิศทางตามแต่ลักษณะการอ่านหรือเขียนระหว่างตัวซีพียู และอุปกรณ์ภายนอก

$\overline{M1}$ ลักษณะเป็นสัญญาณเอ้าท์พุท โดยส่งสัญญาณออกมาบอกว่า ตอนนี้ทำงานอยู่ในสภาวะเฟทช์ (fresh) (สภาวะการเอาข้อมูลคำสั่งจากหน่วยความจำเข้าสู่ตัวซีพียู) โดยแอคทีฟที่ลอจิก "0"

$\overline{MREQ}$ เป็นสายสัญญาณเอ้าท์พุทที่บอกว่า ขณะนี้สัญญาณที่บัสแอดเดรสมีค่าของแอดเดรสที่ต้องการติดต่อกับหน่วยความจำ โดยการแอคทีฟที่ ลอจิก "0"

$\overline{IORQ}$ เป็นสายสัญญาณเอ้าท์พุทที่บอกว่า สัญญาณในแอดเดรสที่ขา A0-A7 มีค่าของแอดเดรสพอร์ทอินพุท เอ้าท์พุท อยู่ ฉะนั้น จึงเป็นสัญญาณที่บ่งบอกถึงการที่ ซีพียูต้องการติดต่อกับหน่วยอินพุท เอ้าท์พุทพอร์ท

$\overline{RD}$ เป็นสัญญาณเอ้าท์พุทที่บอกว่าขณะที่ตัวซีพียู ต้องการอ่านข้อมูลจากหน่วยความจำหรืออุปกรณ์อินพุท เอ้าท์พุทพอร์ท

$\overline{WR}$ เป็นสัญญาณเอ้าท์พุทที่บอกว่าขณะที่ตัวซีพียู ต้องการอ่านข้อมูลลงสู่หน่วยความจำหรืออุปกรณ์อินพุท เอ้าท์พุท

$\overline{RFSH}$ เป็นสายสัญญาณเพื่อบอกว่า ขณะนี้ บัสแอดเดรสมีค่าของแอดเดรสสำหรับการรีเฟรชหน่วยความจำชนิดไดนามิก

$\overline{HALT}$ เป็นสายสัญญาณเอ้าท์พุทที่ แอคทีฟเมื่อ ซีพียูกระทำคำสั่ง HALT โดยจะให้แอคทีฟ ลอจิก "0"

$\overline{WAIT}$ เป็นสัญญาณที่บอกให้ทราบว่า ขณะนี้หน่วยความจำหรืออุปกรณ์อินพุท เอ้าท์พุทยังไม่พร้อมที่จะรับหรือส่งผ่านข้อมูล นั่นคือ เมื่อสัญญาณนี้เข้าไปสู่ตัวซีพียูก็จะหยุดรอจนกว่าสัญญาณ WAIT เลิกไป

$\overline{INT}$ เป็นขารับสัญญาณจากอุปกรณ์อินพุทหรือเอ้าท์พุท ที่ทำการอินเตอร์รัพท์ตัวซีพียู การอินเตอร์รัพท์มีอยู่ด้วยกันหลายโหมดสำหรับสัญญาณที่เข้าสู่ขานี้ จะเรียกว่า มาสเคเบิลอินเตอร์รัพท์ (Maskable Interrupt)

$\overline{RESET}$ เป็นขาสัญญาณที่รับข้อมูลจากอุปกรณ์ภายนอกที่ต้องการรีเซตตัว ซีพียู หรือต้องการให้โปรแกรมเคาร์เตอร์มีค่าเป็น "0"

$\overline{NMI}$ เป็นขารับสัญญาณอินเตอร์รัพท์อีกแบบหนึ่งที่เรียกว่า นอนมาสเคเบิลเตอร์รัพท์เป็นสัญญาณอินเตอร์รัพท์ที่มีความสำคัญสูงสุดที่ ซีพียูต้องรับเสมอ

$\overline{BUSRQ}$ เป็นขาสัญญาณอินพุทที่บอกให้ซีพียูรู้ว่าขณะนี้ อุปกรณ์ภายนอกต้องการใช้ระบบบัส ซึ่งตัวซีพียู ต้องตัดตัวเองออกจากระบบบัสโดยการทำให้ตัวซีพียู อยู่ในสถานะ อิมพีแดนท์สูงและ

ส่งสัญญาณที่ขอใช้บัสทราบว่าขณะนี้ ตัวซีพียูได้ตัดตัวเองออกแล้วพร้อมที่จะให้ใช้งานบัสได้แล้ว

$\overline{BUSAK}$ เป็นสัญญาณที่ส่งออกไปจากตัวซีพียูไม่ได้ใช้ระบบบัสแล้ว

Φ เป็นขารับสัญญาณนาฬิกาเพื่อเป็นฐานเวลาหลักในการทำงานของระบบ

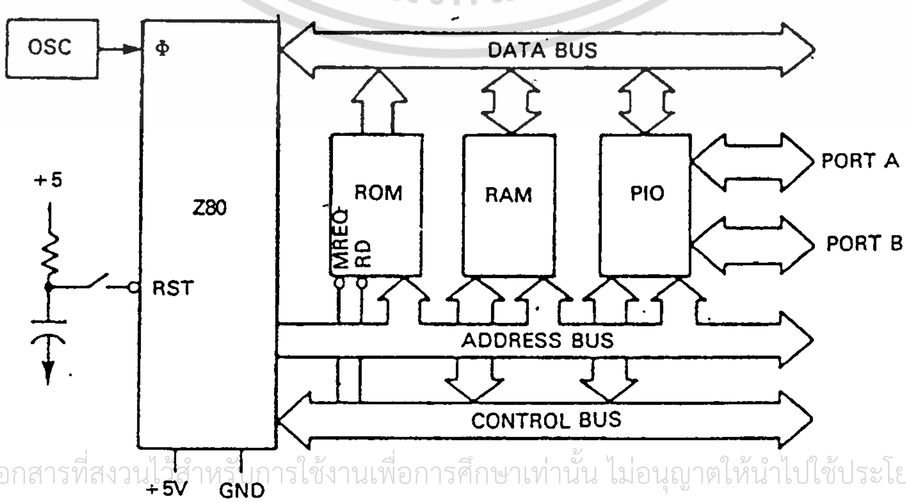
+5V เป็นแรงดันไฟเลี้ยงตัวซีพียู

2.2.3 ระบบของไมโครโปรเซสเซอร์ ของ Z-80

ถึงแม้ว่าตัวของ Z-80 จะมีความสามารถสูงในการประมวลผลต่างๆ แต่โดยลำพังของตัว Z-80 เพียงตัวเดียวนั้น ไม่สามารถที่จะทำงานเป็นระบบไมโครโปรเซสเซอร์ที่สมบูรณ์ได้มันจะต้องมีอุปกรณ์ประกอบเป็นระบบขึ้นมาเพิ่มอีก ระบบจะต้องประกอบด้วยชิ้นส่วนพื้นฐานคือ หน่วยความจำซึ่งจะต้องมี ROM,RAM และอุปกรณ์อินพุท เอ้าท์พุทต่างๆ ดังนี้ระบบฮาร์ดแวร์ของไมโครคอมพิวเตอร์ Z-80 จะประกอบด้วยดังนี้

1. แหล่งจ่ายไฟตรงขนาด 5 โวลท์
2. วงจรกำเนิดสัญญาณนาฬิกาฐานเวลาของระบบ (ซึ่งปกติมักจะมีความถี่ที่จะได้จากคริสตอล)
3. อุปกรณ์หน่วยความจำ RAM (Random Access Memory), ROM (Read Only Memory)
4. วงจรอินพุท เอ้าท์พุท
5. ตัวของ Z-80

ระบบทั้งหมดสามารถแสดงเป็นแผนผังได้ดังรูปที่ 2.22



รูปที่ 2.22 แสดงถึงบล็อกหรือผังส่วนประกอบของระบบ Z-80

จากรูป จะเห็นว่า ตัว CPU จะใช้แหล่งจ่ายไฟเพียง 5 โวลท์ ซึ่งสามารถใช้แหล่งจ่ายไฟร่วมกับระบบหน่วยความจำ หรือ อุปกรณ์อื่นๆ ที่เป็นแบบ TTL ได้ และในการต่อเชื่อมระบบหน่วยความจำนั้นก็สามารถเอาแอดเดรสบัส ของ Z-80 ต่อเข้ากับแอดเดรสบัสของหน่วยความจำได้โดยตรง และบัสข้อมูลก็สามารถต่อร่วมกับขาข้อมูลของหน่วยความจำได้เช่นกัน ส่วนสัญญาณที่ใช้ควบคุมการติดต่อ กับหน่วยจำก็คือ MREQ ซึ่งเป็นสัญญาณเพื่อบอกว่า CPU ต้องการติดต่อกับหน่วยความจำ และสัญญาณที่บอกการอ่านหรือเขียนสู่หน่วยความจำนั้น ก็เป็น RD,WR ตามลำดับ ซึ่งในการเชื่อมต่อทางฮาร์ดแวร์ เราก็จะมาพิจารณาถึงสัญญาณเหล่านี้ เพื่อนำมาใช้อย่างถูกต้อง

ก่อนที่จะได้กล่าวถึงการเชื่อมต่อทางฮาร์ดแวร์ ต่อระบบหน่วยความจำและอุปกรณ์อินพุทเอ้าท์พุทต่างๆ แล้วนั้นจะกล่าวถึงระบบของไมโครโปรเซสเซอร์ เล็กน้อยคือ ระบบไมโครโปรเซสเซอร์นั้น ใช่ว่าเมื่อเรานำเอา Z-80 มาต่อร่วมกับหน่วยความจำ อุปกรณ์อินพุท เอ้าท์พุทต่างๆ แล้วนั้น ระบบจะทำงานได้ทันที ระบบยังจะทำงานไม่ได้ ทั้งนี้เนื่องจากว่าไม่มีโปรแกรมควบคุม การทำงานให้แก่มัน

ดังที่เราได้ทราบมาแล้วว่า ไมโครโปรเซสเซอร์ นั้น จะทำงานโดยการที่มันจะส่งค่าของ PC ไปอ้างแอดเดรสของหน่วยความจำเพื่อดึงเอาคำสั่งมาปฏิบัติ และจะปฏิบัติโดยจะอ่านคำสั่งมาปฏิบัติอย่างต่อเนื่องเรื่อยๆ เมื่อเป็นเช่นนี้หากเรายังไม่มีโปรแกรมที่จะทำให้ CPU อ่านมาปฏิบัติ มันก็ไม่สามารถที่จะทำอะไรได้ เราจึงต้องทำโปรแกรมอันหนึ่งใส่ไว้ในเนื้อที่หน่วยความจำพื้นที่หนึ่งที่ ตัว CPU จะเริ่มต้นทำงานหรือให้แหล่งไฟแก่วงจร เราได้ทราบมาแล้วว่า เมื่อเริ่มทำงานหรือหลังภาวะการรีเซตของ CPU นั้น ค่าของ PC ของ CPU จะเท่ากับ 0000H ดังนั้นเรานั้น เราควรจะเอาโปรแกรมควบคุมการทำงานนั้นไปไว้ที่ แอดเดรส 0000H และควรเป็นพื้นที่หน่วยความจำที่ไม่มีการสูญหาย เมื่อมีการปิดแหล่งจ่ายไฟแก่ระบบไมโครโปรเซสเซอร์ จึงควรเก็บโปรแกรมนี้ไว้ในหน่วยความจำชนิด ROM

โปรแกรมควบคุมที่กล่าวถึงนี้ จะเรียกอีกอย่างหนึ่งว่า **Monitor Program** จะทำหน้าที่ในการ รับอินพุทจากคีย์บอร์ด แสดงผลข้อมูลและ มีฟังก์ชั่นต่างๆ เช่น การแสดงผลค่าแอดเดรสหน่วยความจำและค่าข้อมูล ซึ่งจะเป็นโปรแกรมย่อย อยู่ในหน่วยความจำที่บรรจุโปรแกรมควบคุมนั้น ในระดับ Single board จะเป็นเพียง ROM ตัวที่เดียวเท่านั้น (เช่นในกรณี ET Board) หากเป็นระดับไมโครคอมพิวเตอร์แบบแยกส่วน เช่น IBM PC นั้น ก็เรียกอีกอย่างว่า ROM BIOS ซึ่งจะมี ROM อยู่หลายตัวที่เดียว ความสามารถของโปรแกรมควบคุมนี้จะมากน้อยหรือ มีฟังก์ชั่นที่อำนวยความสะดวกเพียงใดก็ขึ้นอยู่กับผู้เขียนซอฟท์แวร์นั้น ว่ามีอุปกรณ์มากน้อยเพียงใด เพราะผู้

เขียนซอฟท์แวร์เพื่อจะควบคุมระบบจะเขียนตามใจชอบไม่ได้ ต้องคำนึงถึงระบบฮาร์ดแวร์ที่เขียนควบคุมนั้นด้วย อันนี้ก็เป็นสาเหตุหนึ่งว่า ทำให้เราจะเอา ROM ที่บรรจุมอนิเตอร์ของโปรแกรม Single board ไปใส่ให้กับ IBM PC แล้วทำงานไม่ได้หรือเอา ROM BIOS (ซึ่งก็คือมอนิเตอร์โปรแกรม) ของ IBM PC ซึ่งเป็นไมโครคอมพิวเตอร์ระดับ 16 บิท ไปใส่ให้กับเครื่อง NEC PC9801 ซึ่งก็เป็นไมโครคอมพิวเตอร์ระดับ 16 บิท เช่นกัน และใช้ CPU เบอร์เดียวกันแต่ก็ไม่สามารถทำงานได้ ทั้งนี้เป็นเพราะการเชื่อมต่อทางฮาร์ดแวร์ของแต่ละเครื่องของแต่ละเครื่องไม่เหมือนกันนั่นเอง

2.3 8255 พอร์ทแบบขนานที่โปรแกรมได้

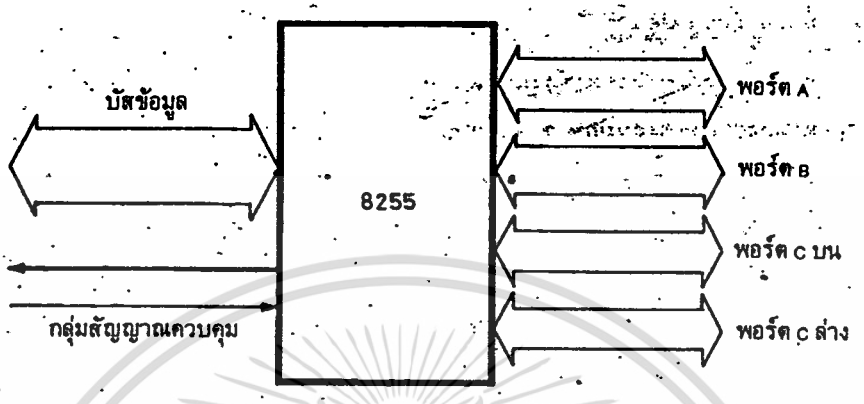
ในการนำเอาไมโครโปรเซสเซอร์ไปใช้งานนั้น จำเป็นต้องให้ไมโครโปรเซสเซอร์สามารถติดต่อกับโลกภายนอกได้ที่รู้จักกันดีคือ ให้มันสามารถส่งสัญญาณมาควบคุมอุปกรณ์ต่างๆ ได้เช่น สเตปปิ้งมอเตอร์, ควบคุมอุปกรณ์ไฟฟ้า ต่างๆ ส่วนที่ทำให้ไมโครโปรเซสเซอร์สามารถติดต่อกับโลกภายนอกได้ที่รู้จักกันก็คือ พอร์ท (PORT) ซึ่งมีอยู่หลายลักษณะด้วยกันเช่น เป็นแบบไอซีไตรสเตท (Tri-state) เบอร์ 74LS244 หรือพวกแลทช์ (Latch) เช่น เบอร์ 74LS374 เหล่านี้สามารถนำไปต่อใช้งานได้กับ CPU ได้โดยง่ายที่สุด โดยตัว CPU จะเป็นตัวควบคุมการอ่านการเขียน พอร์ท หากเป็นการอ่านข้อมูลจากพอร์ทก็มักจะใช้ ไอซี แบบไตรสเตทเป็นพอร์ทอินพุท โดยตัว CPU จะส่งสัญญาณไปเปิดเกทของไตรสเตทนี้ ให้ข้อมูลเข้ามาสู่สายข้อมูล (DATA BUS) และเข้าสู่ไมโครโปรเซสเซอร์หรือ CPU ต่อไป แต่สำหรับมาแลทช์ไว้ที่ตัวมัน (CPU รับสัญญาณมาทริก เพื่อให้อุปกรณ์ภายนอกนั้นรับสัญญาณจากตัวแลทช์ตัวนี้ไปอีกทีหนึ่ง ที่ทำเช่นนี้ ก็เพราะตัวไมโครโปรเซสเซอร์หรือ CPU นี้ทำงานเร็วมาก ซึ่งในช่วงของการส่งข้อมูลออกจากพอร์ทจะใช้เวลาไม่กี่ไมโครเซค (μS) ซึ่งอาจทำให้อุปกรณ์ภายนอกรับไม่ทัน

มีบริษัทต่างๆ เล็งเห็นความสำคัญของการติดต่อระหว่าง CPU กับอุปกรณ์ภายนอกนี้จึงได้ทำไอซีสำเร็จรูปขึ้นหลายเบอร์และที่ขอกล่าวในที่นี้ก็คือ ไอซีเบอร์ 8255ซึ่งเป็นของบริษัท อินเทล ซึ่งได้ออกแบบมาใช้กับ CPU เบอร์ 8080 แต่เราสามารถนำมาประยุกต์ใช้กับ เบอร์อื่นๆ ได้โดยไม่ยาก

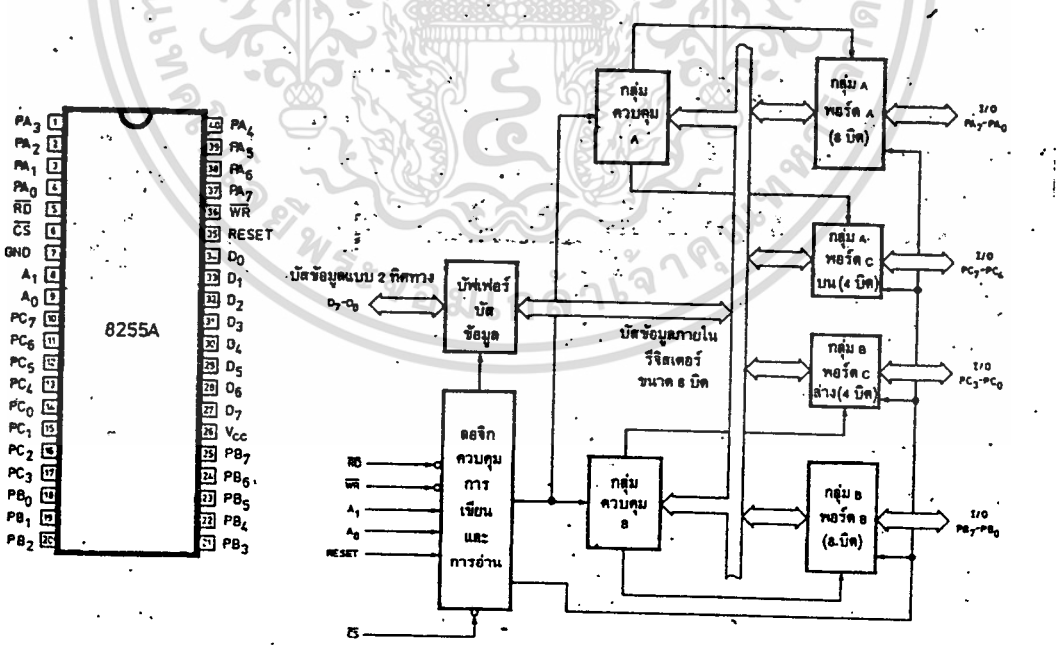
สาเหตุที่ 8255 เป็นที่นิยมมากก็เพราะว่า มันสามารถถูก โปรแกรมให้ทำงานในลักษณะต่างๆ ไม่ว่าจะเป็น อินพุท,เอ้าท์พุท,หรือแม้แต่แบบ แฮนด์เช็คกิ้ง (Handshaking)ได้ ทั้งยังราคาถูกอีกหาก

2.3.1 ลักษณะทั่วไปของ 8255

เป็นไอซีขนาด 40ขา ตัวแบนโดยแยกเป็นลักษณะของบล็อกง่ายๆ ดังรูปที่ 2.23 คือมีพอร์ทใช้งานได้ถึง 3 พอร์ท (เป็นขนาด 8 บิท) พอร์ท A,พอร์ท B,และพอร์ท C โดยพอร์ท C นี้สามารถแยกได้เป็น 2 ส่วนคือ พอร์ท C บน ตั้งแต่ PC4-PC7 จำนวน 4 บิทและพอร์ท C ล่าง ตั้งแต่ PC0-PC3 โดยพอร์ททุกพอร์ท (A,B,C) สามารถโปรแกรมได้ให้เป็น อินพุท เอ้าท์พุท ซึ่งจะได้กล่าวถึงรายละเอียดต่อไป



รูปที่ 2.23 แสดงบล็อกเส้นทางของพอร์ท



รูปที่ 2.24 แผนผังวงจรภายในและการจัดขา ไอซี

ในรูปที่ 2.24 จะเห็นโครงสร้างภายในที่แสดงถึงกลุ่มควบคุมที่มีอยู่ 3 กลุ่ม คือ

- กลุ่มควบคุม A จะควบคุม พอร์ท A และพอร์ท C บน
- กลุ่มควบคุมชุด B จะควบคุม พอร์ท B และพอร์ท C ล่าง
- กลุ่มควบคุมลอจิกการเขียนและการอ่าน

การทำงานของ 8255 จะใช้สัญญาณควบคุมจากตัวไมโครโปรเซสเซอร์มาควบคุมโดยมีคำสั่ง (Control Word)มาที่กลุ่มควบคุมชุด A,B แล้ว กลุ่มควบคุมชุดนี้ก็จะสั่งต่อไปที่พอร์ทเพื่อให้เป็นไปตามข้อกำหนดของคำสั่งนั้นๆ เช่นให้พอร์ท A เป็นอินพุต พอร์ท B เป็นเอ้าท์พุท เหล่านี้เป็นต้น ส่วนกรณีเมื่อมีการอ่านเขียนพอร์ทจาก CPU นั้น กลุ่มควบคุมลอจิกการเขียนอ่าน จะเป็นตัวส่งสัญญาณไปบอกแก่ กลุ่มควบคุมชุด ในแต่ละชุดอีกทีทั้งนี้แล้วแต่ว่า CPU จะมีการอ่านเขียนพอร์ทของ กลุ่มควบคุมชุดใด

ความหมายของขาต่างๆ ของ ไอซี 8255 มีดังนี้

D0-D7 เป็นขาข้อมูลของ 8255 ที่ใช้ติดต่อกับตัวไมโครโปรเซสเซอร์ซึ่งข้อมูลที่จะเข้าออกสู่พอร์ทต่างๆ ของ 8255 จะต้องผ่านขาข้อมูลนี้

$\overline{CS}$ เป็นขาอินพุทที่รับสัญญาณลอจิก "0" จากภายนอกเพื่อแสดงว่าต้องการเลือกใช้ไอซีเบอร์นี้ หากได้รับลอจิก "1" ก็จะทำให้ไอซีตัวนี้ไม่ทำงานคือ ไม่รับรู้สัญญาณใดๆ ทั้งสิ้น

$\overline{RD}$ เป็นขาอินพุทที่รับสัญญาณจากตัวไมโครโปรเซสเซอร์โดยหากมีลอจิกเป็น "0" จะแสดงว่า CPU ต้องการที่จะอ่านข้อมูลจากตัว 8255

$\overline{WR}$ เป็นอินพุทที่รับสัญญาณจากตัวไมโครโปรเซสเซอร์โดยหากมีลอจิกเป็น "0"ก็จะแสดงว่า CPU ต้องการที่จะเขียนข้อมูลจากตัว 8255

A0-A1 เป็นอินพุทที่รับแอดเดรสจากตัวไมโครโปรเซสเซอร์ ที่ถอดระหัสตำแหน่งของ 8255 เรียบร้อยแล้วโดยจะมีตำแหน่งใช้งาน 4 ตำแหน่ง เพื่ออ่าน เขียนรีจีสเตอร์ (พอร์ท) ของ 8255 ที่มีอยู่ด้วยกัน 4 ตัว

RESET เป็นอินพุท ที่รับสัญญาณจากภายนอกเข้ามาทำการรีเซตตัว 8255 โดยหากได้รับลอจิก "1" จะทำให้พอร์ททุกพอร์ทเป็นอินพุทพอร์ทหมดทั้งนี้ เพื่อไม่ต้องการให้มีสัญญาณออกไปกวนต่อระบบภายนอกเมื่อ 8255 ได้รับสัญญาณรีเซต

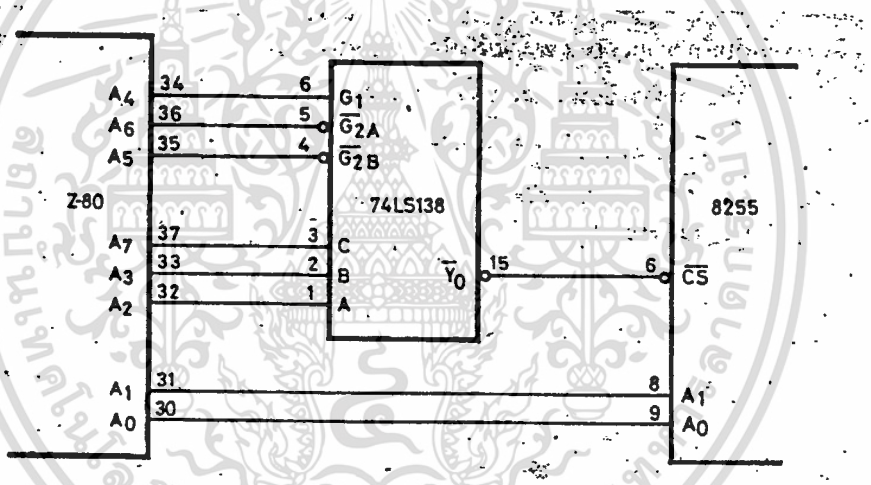
PA0-PA7 เป็นขาสัญญาณพอร์ท A ที่ใช้ติดต่อกับโลกภายนอก

PB0-PB7 เป็นขาสัญญาณพอร์ท B ที่ใช้ติดต่อกับโลกภายนอก

PC0-PC7เป็นขาสัญญาณพอร์ท C ที่ใช้ติดต่อกับโลกภายนอก ซึ่งพอร์ทนี้จะแบ่งเป็น 2กลุ่ม คือ PC0-PC3 และ PC4-PC7 ซึ่งสามารถโปรแกรมแยกกันได้อีก

2.3.2 การต่อใช้งาน 8255

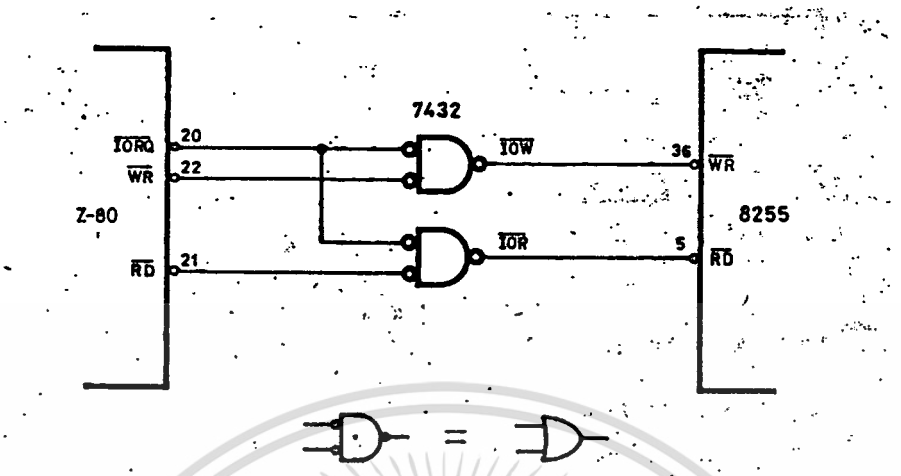
หากดูที่ขาของ 8255 ที่รูปที่ 2.24 แล้วจะสังเกตุเห็นได้ว่าเราสามารถต่อขา 8255 บางส่วนได้โดยตรงกับขา ไมโครเซสเซอร์เลย (Z-80) เช่นขา D0-D7, A0-A1 เป็นต้น หากแต่บางขาเราจำเป็นต้องมีการดัดแปลงสัญญาณที่ได้จาก CPU (ซึ่งกรณีนี้ เราใช้เบอร์ Z-80)เสียก่อน โดยหากเราถอดระหัสแอดเดรสของ 8255 ให้เป็นพอร์ทที่แอดเดรส 10H,11H,12H,13H เราจะสามารถทำ ดีโค็ดเดอร์โดยง่ายดังรูปที่ 2.25



รูปที่ 2.25 แสดงการดีโค็ดแอดเดรสพอร์ทให้ 8255

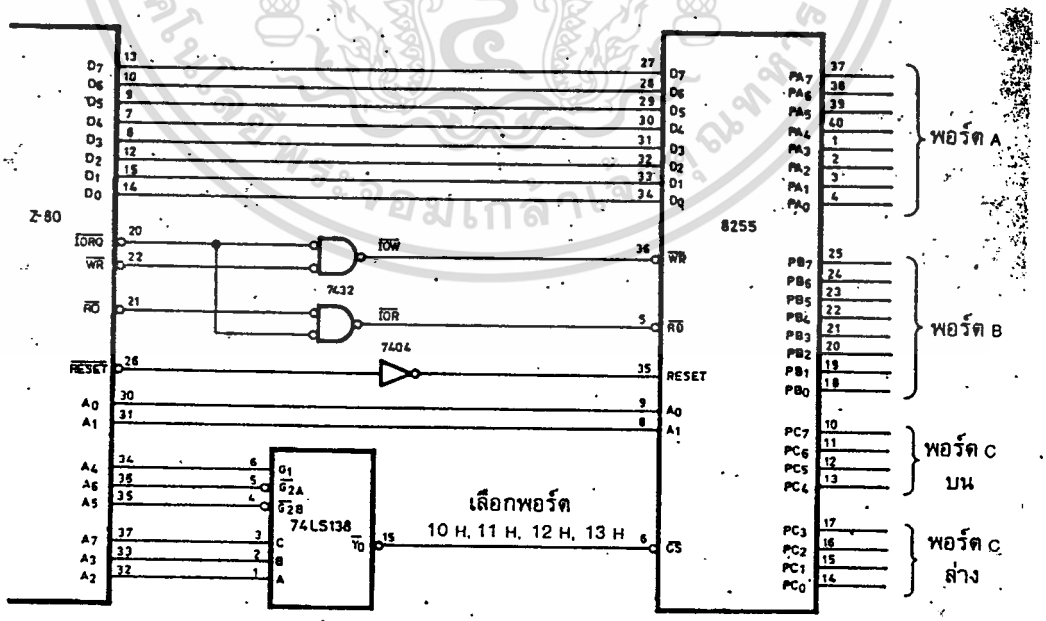
สังเกตุได้ว่า CS จะแอคทีฟทุกครั้งหาก Z-80 อ้างพอร์ทที่แอดเดรส 0001000XX โดยค่าของ XXคือ A0,A1 ที่เราจะต่อตรงเข้ากับ 8255 เพื่อทำการเลือกรีจีสเตอร์ควบคุมและพอร์ททั้งสามของ 8255

สัญญาณการควบคุมอีกส่วนที่สำคัญก็คือ สัญญาณการอ่านเขียนพอร์ทของ 8255 โดยหาก WR ได้รับแอคทีฟ "0" ก็จะเป็นการเขียนข้อมูลจาก CPU เข้าสู่ตัว 8255 หรือออกสู่พอร์ทที่ต้องการ และเช่นเดียวหาก RD ได้รับลอจิก "0" ก็จะเป็นการอ่านข้อมูลพอร์ทจากตัว 8255 เข้าสู่ CPU เข้ากับตัว Z-80 แล้ว เราจะต้องทำเป็นสัญญาณการอ่านเขียนพอร์ทของ Z-80 ให้ถูกต้องเสียก่อนดังรูป 2.26



รูปที่ 2.26 การเชื่อมต่อสัญญาณ อ่านเขียนพอร์ทระหว่าง Z-80,8255

และสุดท้ายคือสัญญาณ RESET แอคทีฟ "0" ฉะนั้นเราจะต้องมาผ่านอินเวอร์เตอร์ก่อน ก็จะได้ลักษณะการต่อร่วมกับ CPU ดังนี้



รูปที่ 2.27 แสดงการต่อ 8255 ร่วมกับ Z-80

2.3.3 การโปรแกรม 8255

เราได้ทราบมาแล้วในรูปที่ 2.24 ว่าโครงสร้างภายในของ 8255 มีการควบคุมชุดอยู่ 3 กลุ่ม ซึ่งทั้งสามกลุ่มนี้จะทำงานร่วมกันดังกล่าวมา และเราสามารถจะควบคุมการทำงานของพอร์ทจาก CPU ได้โดยสั่งงานมาที่ กลุ่มควบคุมดังกล่าว แต่ตัว CPU จะมองเห็น 8255 เป็น 4 พอร์ทด้วยกัน โดยแต่ละพอร์ทเสมือนเป็นรีจีสเตอร์ที่ CPU สามารถจะทำการอ่าน/การเขียนได้ แต่ละพอร์ทจะอยู่กันคนละแอดเดรส ดังที่ทำการดีโค้ดให้ 8255 ที่แอดเดรส 10H,11H,12H,13H (ตามสัญญาณ A0-A1) และเราจะสามารถได้ตำแหน่งของพอร์ท 8255 แต่ละตัวดังรูป

10H ----------> พอร์ท A

11H ----------> พอร์ท B

12H ----------> พอร์ท C

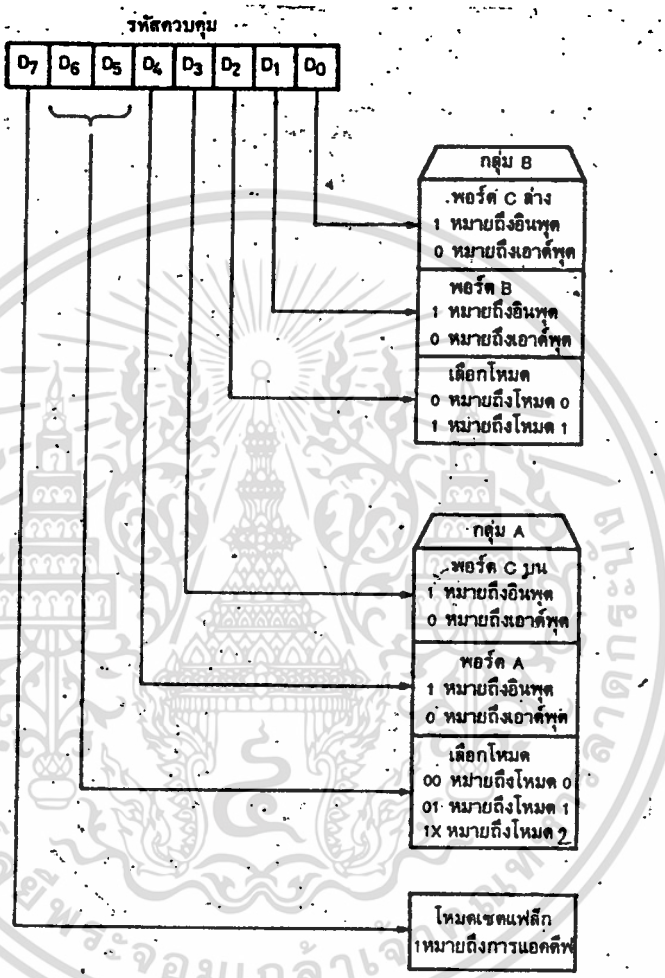
13H ----------> พอร์ท Control

ซึ่งหากมีการอ่านเขียนไปยังพอร์ทดังกล่าว ก็จะใช้ร่วมกับสัญญาณ RD, WR โดย WR หมายถึงเอ้าท์พุทข้อมูลและ RD แอคทีฟหมายถึง อินพุทข้อมูล ดังนั้นเราจะได้ลอจิกที่ขาของ 8255 ในลักษณะต่างๆ ดังนี้

RD	WR	A1	A0	ความหมาย
1	0	0	0	เขียนพอร์ท A ซึ่งเป็นข้อมูล
0	1	0	0	อ่านพอร์ท A ซึ่งเป็นข้อมูล
1	0	0	1	เขียนพอร์ท B ซึ่งเป็นข้อมูล
0	1	0	1	อ่านพอร์ท B ซึ่งเป็นข้อมูล
1	0	1	0	เขียนพอร์ท C ซึ่งเป็นข้อมูล
0	1	1	0	อ่านพอร์ท C ซึ่งเป็นข้อมูล
1	0	1	1	เขียนข้อมูล ซึ่งเป็นระหัสควบคุม
0	0	1	1	อ่านเข้ามา ซึ่งไม่มีความหมายใด

ตารางที่ 2.1 แสดงลอจิกเมื่อทำการติดต่อกับ 8255

การใช้งาน เราจะต้องส่งระหัสควบคุม (Control code) หรือเรียกอีกอย่างหนึ่งว่า รีจีสเตอร์ควบคุม ซึ่งจะเป็นข้อมูลขนาด 1ไบท์ ส่งไปที่แอดเดรส 13H (กรณีนี้เราถอดระหัสไว้ที่ 13H) โดยหมายความของแต่ละบิทที่เราส่งไปโปรแกรมการทำงานเป็นดังนี้



รูปที่ 2.28 ความหมายของแต่และบิทในระหัสควบคุม

บิท D7 เป็นบิทที่แสดงว่าในไบท์นี้เป็นระหัสควบคุม ถ้าเป็น "1" โดยแต่ละบิทจะมีผล มีการเปลี่ยนแปลงโหมดต่างๆ ของ 8255หากเป็น "0" การเซทพอร์ทของC

บิท D6,D5เป็นการเลือกโหมดของพอร์ท A ซึ่งจะมีอยู่ ด้วยกัน 3 โหมดคือ 0,1,2

บิท D4 เป็นการกำหนดให้พอร์ท A เป็นอินพุทหรือเอ้าท์พุท โดยหากเป็น "1" ก็จะเป็นอินพุท หากเป็น "0" ก็จะเป็นเอ้าท์พุ้ท

บิท D3 เป็นการกำหนดให้พอร์ท C บน ให้เป็นอินพุทหรือเอ้าท์พุท โดยหากเป็น "1"

ก็เป็นอินพุท หากเป็น "0" ก็จะเป็นเอ้าท์พุท

บิท D2 เป็นการกำหนดโหมดการทำงานของพอร์ท Bโดยหากเป็น "0"หมายถึงเลือกให้พอร์ท B ทำงานในโหมด 0 หากเป็น "1"เป็นการเลือกให้พอร์ท"1"เป็นการเลือกให้พอร์ท B ทำงานในโหมด 1

บิท D1 เป็นการกำหนดให้พอร์ท B ให้เป็นอินพุทหรือเอ้าท์พุท โดยหากเป็น "1" ก็จะเป็นอินพุท หากเป็น "0" ก็แสดงว่าให้เป็นเอ้าท์พุท

บิท D0 เป็นการกำหนดให้พอร์ท C ล่าง ให้เป็นอินพุท โดยหากเป็น "1" ก็จะเป็นอินพุท หากเป็น "0" ก็จะแสดงว่าให้เป็นเอ้าท์พุท

การโปรแกรมก็จะเริ่มจากการส่งค่าระหัสควบคุม 1ไบท์ดังกล่าวนี้ไปสู่พอร์ทควบคุม หลังจากนั้นหากต้องการเรียกไปถึงพอร์ทใดก็พร้อมสามารถอ้างได้ตามแอดเดรสทันที เช่นต้องการโปรแกรมให้พอร์ท A,B,C ทั้งหมดให้เป็นเอ้าท์พุทพอร์ท เราก็จะได้ระหัสควบคุมเป็น 10000000 หรือ 80H เราก็จะสั่งได้เป็น

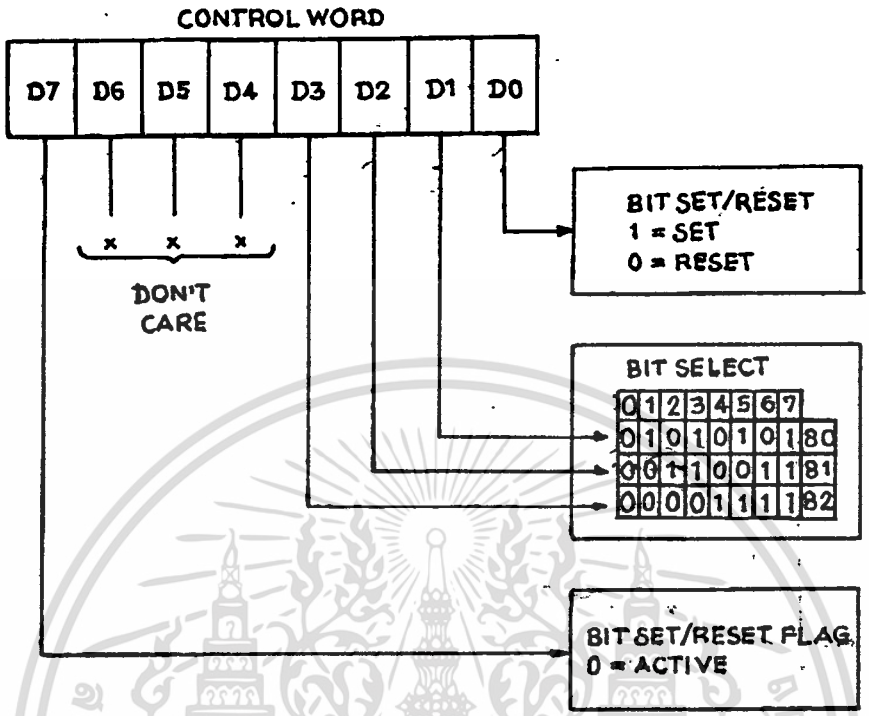
```
LD   A,80H    ; กำหนดระหัสควบคุม
OUT  (13H),A  ; เป็นการส่งระหัสควบคุมสู่รีจีสเตอร์ควบคุม 8255
```

จากนี้ทั้งสามพอร์ท ก็จะเป็นเอ้าท์พุทพอร์ท ให้เราตามต้องการเมื่อเราจะส่งค่าออกไปก็สามารถกระทำได้โดยง่าย เช่น ต้องการส่งค่า 88H ออกไปที่พอร์ท B,C เราจะทำดังนี้

```
LD   A,88H   ;ค่าของข้อมูลที่ต้องการส่ง
OUT  (11H),A ;ส่งออกไปพอร์ท B
OUT  (12H),A ;ส่งออกไปพอร์ท C
```

<u>กรณีพิเศษของระหัสควบคุม</u>

ปกติระหัสที่เราต้องส่งออกไปถึงพอร์ทควบคุมหรือรีจีสเตอร์ควบคุมนี้ จะต้องเป็นการเซทโหมด,พอร์ทอินพุท, เอ้าท์พุท และระหัสนั้น บิท 7 จะต้องเป็น "1" เสมอ ที่นี้หากบิท 7 นี้เป็น "0" บ้างจะเกิดอะไรขึ้นถ้าหากบิต 7 เป็น "0" และถูกส่งไปที่ แอดเดรสของพอร์ทควบคุมแล้ว 8255 จะถือว่าเป็นคำสั่งของการ เซทบิตของพอร์ท C ทันที โดยจะมี ฟอร์แมทดังรูปที่ 2.29 ดังนี้



รูปที่ 2.29 แสดงถึงระหัสควบคุมที่ใช้ในการเซท/รีเซท บิท

ตัวอย่างเช่น

```
LD   A,80H          ;ระหัสควบคุมที่ใช้ทั้งสามพอร์ทเป็น OUT PUT
OUT  (13H),A        ;ส่งสู่รีจีสเตอร์ควบคุม
LD   A,00010001B    ;ระหัสควบคุมใช้เซทบิทที่ 4 ของพอร์ท C "1"
OUT  (13H),A        ;บิท 4 พอร์ท C เป็น "1"
DEC  A              ;เปลี่ยนเป็นรีเซท
OUT  (13H),A        ;กำหนดให้บิท 4 พอร์ท C เป็น "0"
```

จะเห็นว่าสามารถนำไปประยุกต์ใช้งานได้ เช่น เป็นตัวสร้าง พัลซ์ และกำหนดใช้งาน เปิด_ปิด อุปกรณ์ด้วย พอร์ท C ที่มีคำสั่งไม่ยุ่งยากและเป็นอิสระเป็นต้น

2.3.4 การทำงานในโหมด 0

จัดว่าเป็นโหมดพื้นฐานที่ นิยมใช้กันมากที่สุด เนื่องด้วยความง่ายตรงไปตรงมา คือทั้งสามพอร์ทเราสามารถจะโปรแกรมได้ชุดใดชุดหนึ่ง เป็นอินพุทหรือเป็นเอ้าท์พุทได้อีก ฉะนั้นโดย

สรุปแล้วก็จะมีพอร์ทที่จะโปรแกรมให้เป็นอินพุทหรือเอ้าท์พุทได้เสมือน 4 พอร์ท คือพอร์ท A,พอร์ท B,พอร์ท C บนและพอร์ท C ล่าง (แต่โปรแกรมแยกเฉพาะบิทในแต่ละพอร์ทไม่ได้)

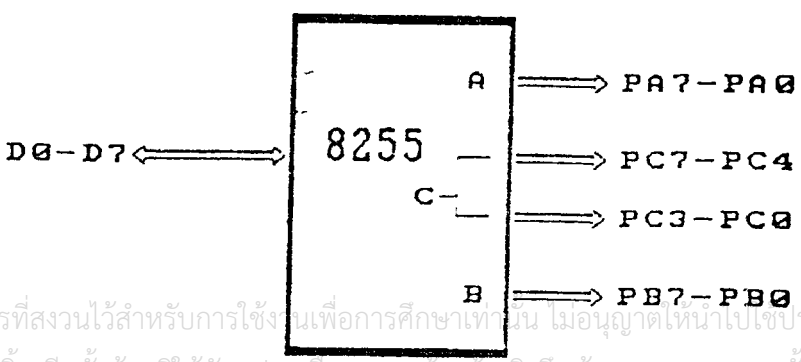
1	0	0	X	X	0	X	X
BIT	7	6	5	4	3	2	1	0

ตารางที่ 2.2 แสดงระหัสคำสั่งของโหมด 0

ซึ่งหากเราดูที่ระหัสคำสั่งแล้วจะเห็นว่ามีอยู่ 4 บิท ที่ถูกกำหนดตายตัวคือ บิท 7,บิท 5, บิท 6 และบิท 2 ส่วนที่เหลืออีก 4 บิทก็คือกำหนดว่าให้พอร์ทใดเป็นพอร์ทอินพุท/เอ้าท์พุท นั่นเอง ซึ่งหากเราให้พอร์ทใดเป็นอินพุท เราก็ใส่ลอจิก "1"ที่บิทนั้น หรือหากเราต้องการให้พอร์ทใดเป็นเอ้าท์พุทก็ให้ใส่ "0"ที่บิทนั้น จะเห็นว่ามีความเป็นไปได้ในการกำหนดลักษณะของพอร์ทใน โหมดนี้อยู่ 16 อย่างด้วยกัน

ตัวอย่างเช่น 1.ต้องการให้ทุกพอร์ทเป็นเอ้าท์พุทหมด เราจะได้ระหัสคำสั่งเป็น

1	0	0	0	0	0	0	0



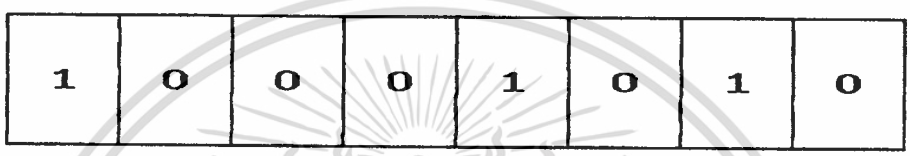
2. ต้องการให้ พอร์ท A --------> เอ้าท์พุท

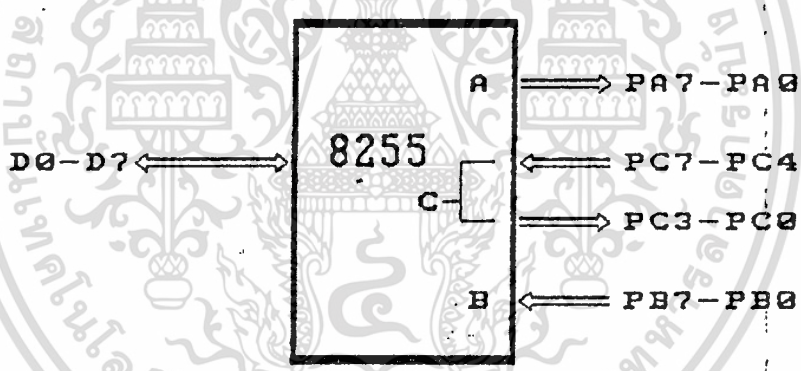
พอร์ท B --------> อินพุท

พอร์ท C บน --------> อินพุท

พอร์ท C ล่าง--------> เอ้าท์พุท

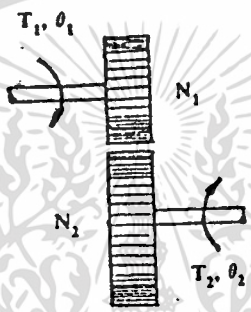
เราจะได้ระหัสคำสั่งเป็น





2.4 ระบบเกียร์

ระบบเกียร์ คานงัดหรือสายพานในลูกรอก เป็นอุปกรณ์เครื่องกลซึ่งสามารถส่งพลังงานจากส่วนหนึ่งของระบบไปยังส่วนอื่นๆ ได้ในรูปของแรงงาน แรงบิด ความเร็วและการเคลื่อนที่ นอกจากนั้นอุปกรณ์เหล่านี้ยังเป็นเสมือนอุปกรณ์สำหรับประสาน (Matching) ที่สามารถใช้ส่งผ่านกำลังให้ได้ค่าสูงสุด รูปที่ 2.30 แสดงถึงการคัพปลิ้งเกียร์ 2 ตัวเข้าด้วยกัน แรงเฉื่อยและแรงเฉื่อยและแรงเสียดทานของเกียร์ไม่นำมาคิดในเมื่อนิจารณาถึงเกียร์ในอุดมคติ



รูปที่ 2.30 ระบบการคัพปลิ้งของเกียร์

ความสัมพันธ์ระหว่างแรงบิด T1 และ T2 การเคลื่อนที่เชิงมุม 01 และ 02 และจำนวนชี่ฟัน N1 และ N2 ของระบบเกียร์สามารถหาได้จากหลักเกณฑ์ต่อไปนี้

(1) จำนวนชี่ฟันของเกียร์จะเป็นสัดส่วนกับรัศมี r1 และ r2 ของเกียร์ นั่นคือ

$$r_1 N_2 = r_2 N_1 \qquad \text{----------(1)}$$

(2) ระยะทางการที่เคลื่อนที่ไปของเกียร์แต่ละตัวจะมีเท่ากัน ดังนั้น

$$\theta_1 r_1 = \theta_2 r_2 \qquad \text{----------(2)}$$

(3) แรงงานที่ได้จากเกียร์ตัวหนึ่งจะเท่ากับแรงงานที่ได้จากเกียร์อีกตัวหนึ่ง เนื่องจากสมมติให้ว่าไม่มีการสูญเสียแรงงาน ดังนั้น

$$T_1 \theta_1 = T_2 \theta_2 \qquad \text{----------(3)}$$

ถ้าความเร็วเชิงมุมของเกียร์ทั้งหมดคือ W_1 และ W_2 สมการที่ (1),(2)และ(3)เขียนใหม่ได้เป็น

$$(T_1/T_2) = (\theta_2/\theta_1) = (N_1/N_2) = (W_2/W_1) = (r_1/r2)\text{----------(4)}$$

ในทางปฏิบัติ ตัวเกียร์จริงๆ จะต้องมีแรงเฉื่อยและแรงเสียดทาน เกิดขึ้นเนื่องจากการคัพปลิ้งระหว่างซี่ฟันของเกียร์ซึ่งไม่สามารถตัดทิ้งได้ ระบบสมมูลย์ของเกียร์ให้มีวิสคอฟริคชั่นคูว์ลอมป์ฟริคชั่นและแรงเฉื่อย พิจารณาเป็นส่วนประกอบของระบบ ตัวแปรและพารามิดตอร์ต่อไปนี้อธิบายถึงระบบเกียร์

T = แรงบิดที่ป้อนให้ระบบเกียร์

θ_1 และ θ_2 = ระยะทางการเคลื่อนเป็นแบบเชิงมุม

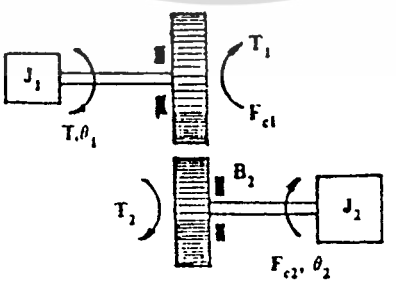
T_1 และ T_2 = แรงบิดที่ส่งผ่านมายังเกียร์

J_1 และ J_2 = แรงเฉื่อยของเกียร์

N_1 และ N_2 = จำนวนซี่ฟัน

F_{c1} และ F_{c2} = ตัวสัมประสิทธิ์ของคูว์ลอมป์ฟริคชั่น

B_1 และ B_2 = ตัวสัมประสทธิ์ของวิสคอสฟริคชั่น



รูปที่ 2.31 ระบบคัพปลิ้งเกียร์ที่มีแรงเฉื่อย

สมการแรงบิดของเกียร์ทั้งสองเขียนได้เป็น

$$T_2(t) = J_2 \{d^2\theta_2(t)/dt^2\} + B_2\{d\theta_2(t)/dt\}+F_{c2} \{\theta_2/\theta_2\} \quad -----(5)$$

สมการแรงบิดทางซ้ายของเกียร์ดังแรงได้เป็น

$$T_1(t) = J_1 \{d^2\theta_1(t)/dt^2\} + B_1\{d\theta_1(t)/dt\}+F_{c1}\{\theta_1/\theta_1\}+T_1(t)---(6)$$

โดยใช้สมการที่ (4) สมการที่ (5) จะแปลงใหม่ได้เป็น

$$T_1(t) = (N_1/N_2)T_2(t) = (N_1/N_2)^2 J_2\{d^2\theta_1(t)/dt^2\}+$$

$$(N_1/N_2)^2B_2\{d\theta_1(t)/dt\}+(N_1/N_2) F_{c2}\{\theta_2/\theta_2\} \quad -----(7)$$

สมการที่(7)แสดงให้เห็นว่าเป็นไปได้ที่จะเป็นไปได้ที่จะสะท้อนแรงเฉื่อย แรงเสียดทาน แรงบิด ความเร็วและการเคลื่อนที่จากข้างหนึ่งของระบบเกียร์ไปยังอีกข้างหนึ่งของระบบเกียร์

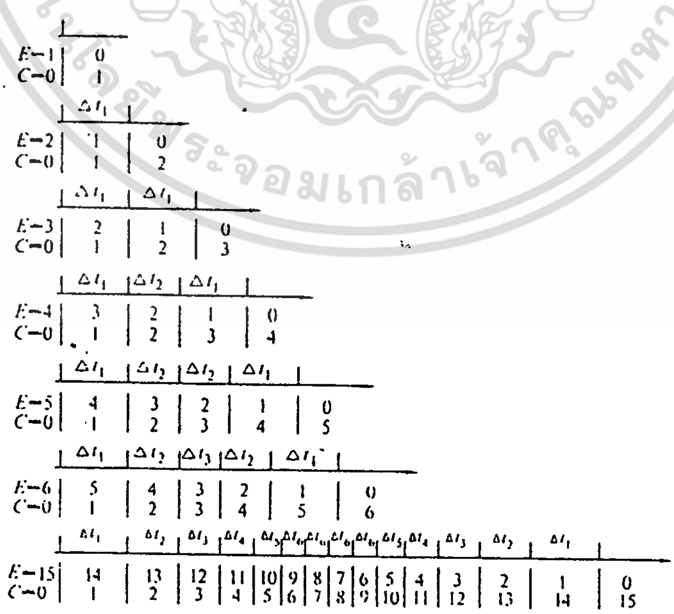
บทที่ 3
เทคนิคการสร้าง

เพื่อการเริ่มต้นของการหมุนในการทำงานของสเตปปิ้งมอเตอร์ เป็นไปอย่างมีประสิทธิภาพจึงนำเอาเทคนิค อัตราเร่ง/อัตราลด มาใช้ในโครงการนี้ จึงทำให้แน่ใจว่าจะไม่มีการสูญเสียสเตปของการทำงานเกิดขึ้น ในระบบการขับมอเตอร์แบบ Open-loop เมื่อเริ่มต้นอัตราการสร้างสเตปขึ้นจะต้องต่ำ ซึ่งจะทำให้การหลุดปราศจากตำแหน่งที่ผิดพลาด

ไมโครโปรเซสเซอร์ จะเป็นตัวกำหนดระยะเวลาพัลซ์แทนวงจร Logic Sequencing โดยการเขียนโปรแกรมควบคุมพอร์ทเอ้าท์พุท (8255) เพื่อให้ความเร็ว, จำนวนพัลซ์และทิศทางเป็นไปตามโปรแกรม

3.1 แนวความคิดพื้นฐานของการใช้ Z-80 Microprocesser

CPU หรือ Central Processing Unit ซึ่งเป็นหน่วยประมวลผลกลาง มีรีจีสเตอร์เป็นหน่วยความจำชั่วคราวสามารถใช้งานได้เอนกประสงค์ ในระบบโปรแกรมที่จะสร้างขึ้นนี้จะใช้ Register ในการทำงานต่างๆ ตามลำดับดังนี้

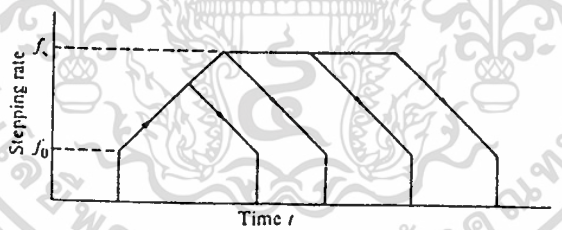


รูปที่ 3.1 อัตราการเพิ่ม/ลด ของพัลซ์และการเปลี่ยนข้อมูลในรีจีสเตอร์ E,C

Register B รีสเตอร์นี้จะใช้ในการเก็บสถานะการหมุนปัจจุบัน

Register C และ E รีสเตอร์ C จะใช้ในการนับสเตปที่ทำเสร็จแล้ว และ รีสเตอร์ E จะทำการเก็บจำนวนสเตป ซึ่งเหลืออยู่ในการเคลื่อนที่ เพื่อให้มีการเคลื่อนที่มีจำนวนเท่ากับ P สเตป จำนวน P จะถูกโหลดไว้ที่ รีสเตอร์ E และเคลียรีสเตอร์ C ให้เป็นศูนย์ หลังจากส่งสัญญาณสเตปออกสู่เอ้าท์พุท C จะเพิ่มค่า และ E จะลดลงที่ละหนึ่ง ตัวอย่างความต่อเนื่องระยะระหว่างพัลซ์และการเปลี่ยนค่าของรีสเตอร์ C และ E ดังแสดงในรูปที่ 3.1

เป็นแบบทั่วๆ ไปของการสเตปคำสั่งความเร็วและเวลาชั่วขณะจะแสดงให้ดูในรูปที่ 3.2 มอเตอร์จะเริ่มต้นที่ อัตราการสเตป Fo โดยเพิ่มขึ้น (เร่ง) อย่างเป็นเชิงเส้นจนถึงค่าสูงสุด (Slewing) ที่ Fs และทำงานที่อัตรานี้คงที่ ก่อนที่มอเตอร์จะมีการหยุด มอเตอร์จะมีอัตราการสเตปลดลงจนหยุดในที่สุด การที่หยุดหลังจากความเร็วแบบนี้ จะทำให้หยุดมอเตอร์โดยไม่เกิดการทำงานเกินสเตปที่สั่ง



รูปที่ 3.2 อัตราส่วนความเร็วกับเวลาชั่วขณะ

ในโปรแกรมตัวอย่างนี้ มอเตอร์มีอัตราเร่งด้วย 6 พัลซ์ และอัตราลดลง 6 พัลซ์โดยใช้การทำงานซึ่งมีระยะพัลซ์เหมือนกันกับอัตราเร่งในทิศทางกลับกัน ในตัวอย่างที่ 3.1นั้นถ้าการเคลื่อนที่มีค่าน้อยกว่า 13 สเตป การทำงานจะไม่มีค่าความเร็วสูงสุด มอเตอร์จะมีอัตราเร่ง, อัตราสูงสุดและอัตราลดลง เมื่อการเคลื่อนที่มีค่าเท่ากับ 13 สเตป หรือมากกว่า

รีสเตอร์ C และ E จะนำไปคำนวณหาตำแหน่งที่เก็บข้อมูลของช่วงเวลาพัลซ์ที่เกิดขึ้น

รีสเตอร์ D จะใช้เก็บระหัสของทิศทางของการหมุนตามนี้

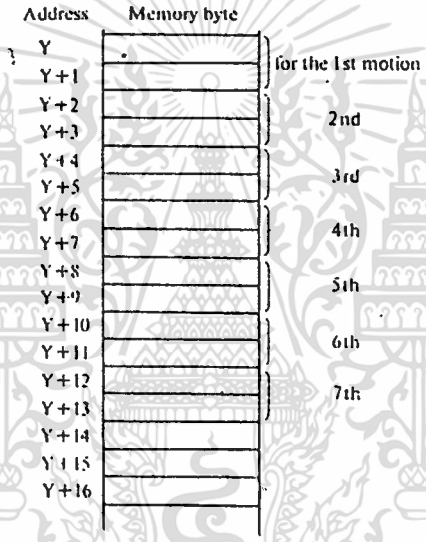
D = 0 หรือ 00000000 สำหรับ CW (ตามเข็มนาฬิกา)

D = 1 หรือ 00000001 สำหรับ CCW (ทวนเข็มนาฬิกา)

ถ้า D มีค่าเท่ากับ 2 ขบวนการทำงานจะสิ้นสุดลงของข้อมูลสำหรับการเคลื่อนที่ ดังนั้น

D = 2 หรือ 00000010 สำหรับ END ของโปรแกรม

ทิศทางและจำนวนสเตปที่หมุนในการเคลื่อนที่ จะถูกเก็บไว้บนพื้นที่หน่วยความจำที่เริ่มต้นจากตำแหน่ง Y ซึ่งแสดงในรูป 3.3 ก่อนการเคลื่อนที่ครั้งแรก ข้อมูลจากตำแหน่ง Y จะถูกย้ายเข้าไปในรีสเตอร์ D และใน Y+1 และ Y+3 ก็จะย้ายเข้าไปในรีจีสเตอร์ D และ E ตามลำดับ



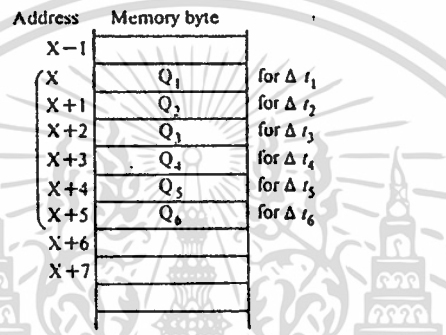
รูปที่ 3.3 พื้นที่หน่วยความจำสำหรับทิศทางและจำนวนสเตป

รีจีสเตอร์คู่ HL ในโปรแกรมนี้ รีจีสเตอร์ HL จะใช้เก็บตำแหน่งของหน่วยความจำของข้อมูลระยะพัลซ์ รูปที่ 3.4 แสดงให้เห็นถึงพื้นที่ในหน่วยความจำเก็บข้อมูลของระยะพัลซ์ซึ่งประกอบด้วย 6 ตำแหน่งจาก X ถึง X+5 ระยะพัลซ์ t_m คำนวณหาได้จากข้อมูล Q_m ในแต่ละไบท์ในหน่วยความจำ โดยใช้สูตร

$$t_m = aQ_m + b$$

เมื่อ a และ b เป็นค่าคงที่ ซึ่งกำหนดโดยซอฟท์แวร์และความถี่สัญญาณนาฬิกาใน Micro processer

ตำแหน่งหน่วยความจำ ที่ใช้ในการเร่งจะเริ่มต้นตั้งแต่ X ขึ้นไปและ ที่ใช้ในการลดจะเริ่มต้นแต่ X+5 ลงมา ระยะพัลซ์ของความเร็วสูงสุด (Slewing-Pulse) จะสร้างจากข้อมูล Q6 ซึ่งเก็บข้อมูลอยู่ในตำแหน่ง X+5

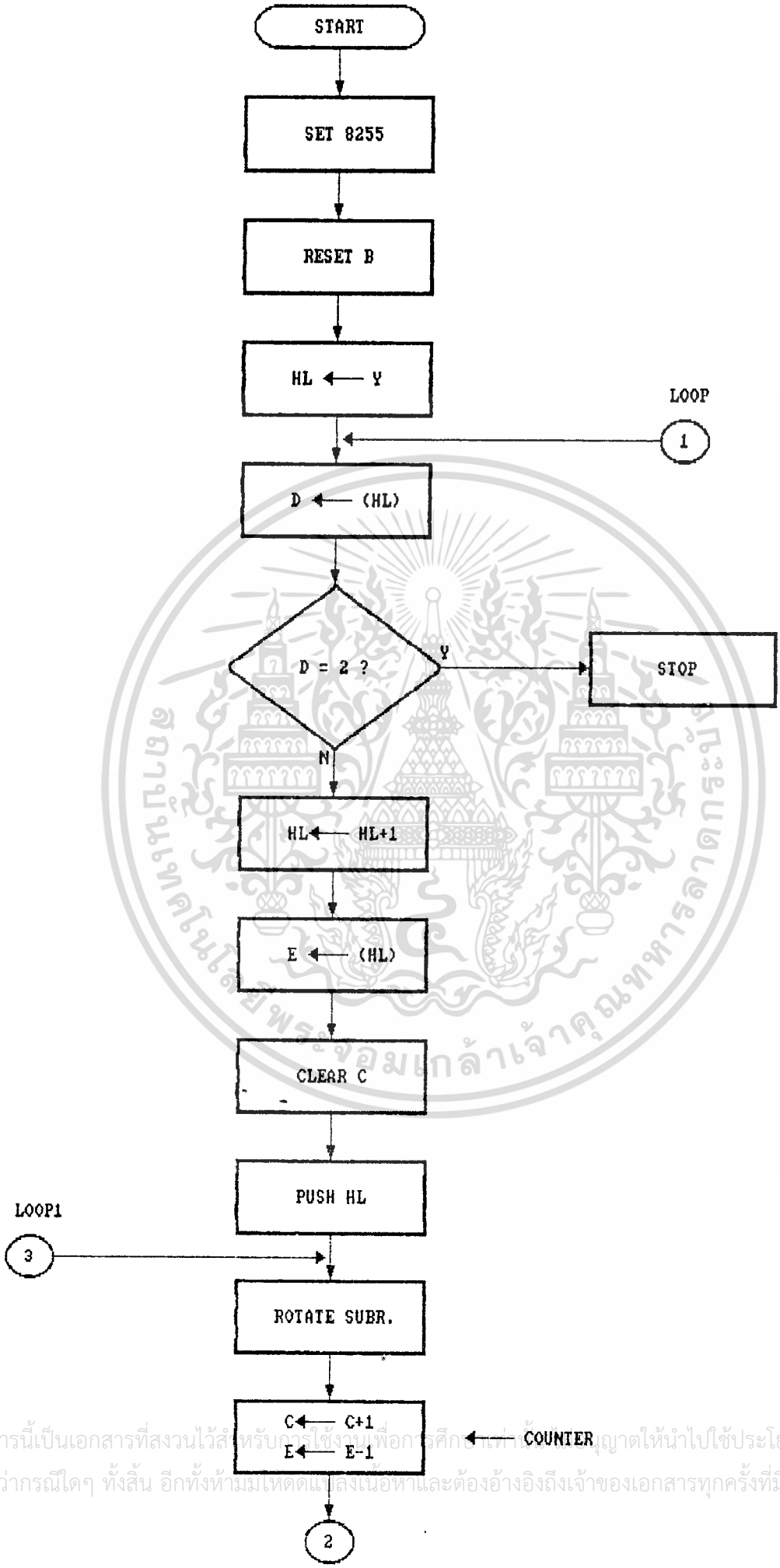


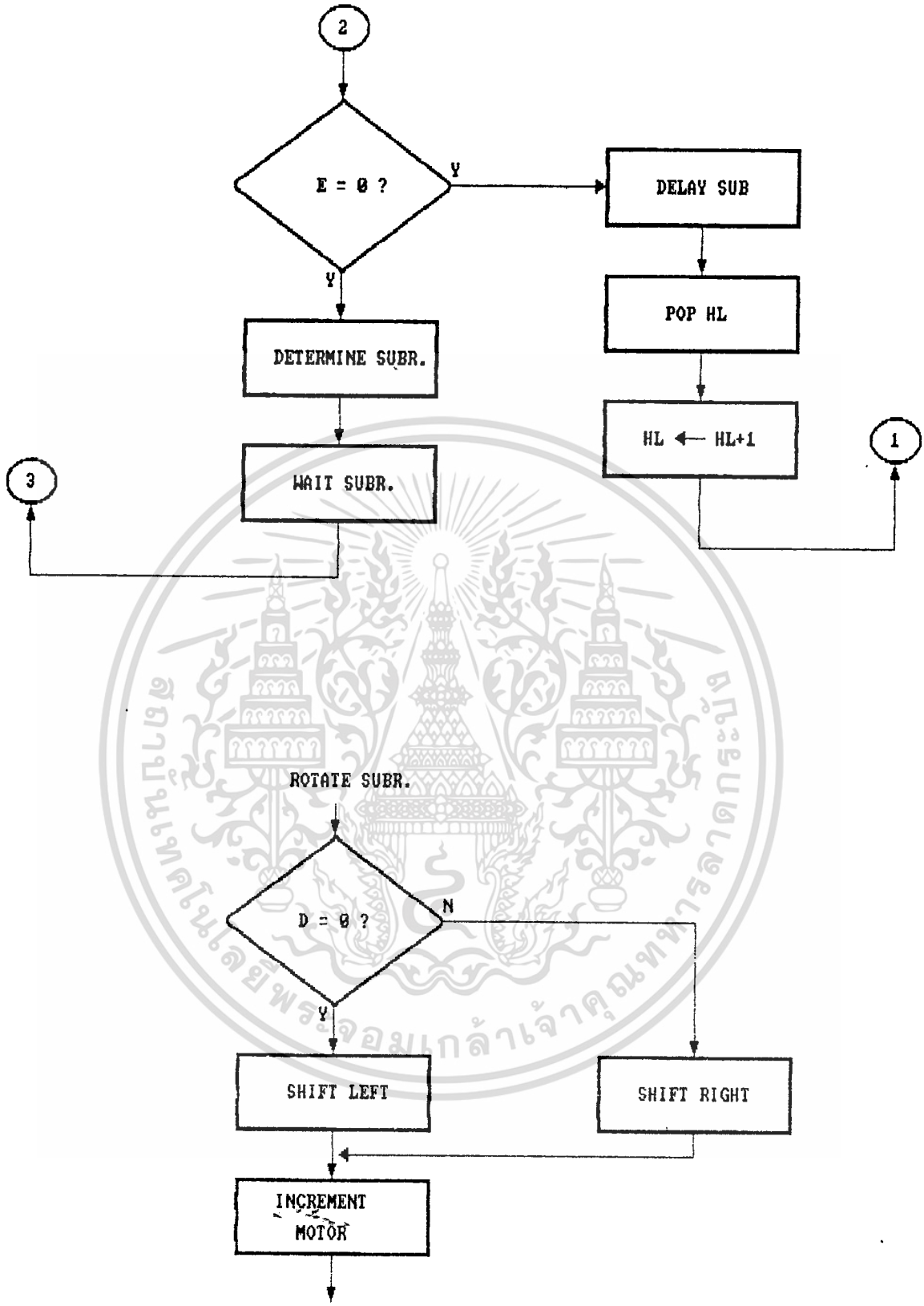
รูปที่ 3.4 พื้นที่หน่วยความจำสำหรับข้อมูลระยะพัลซ์

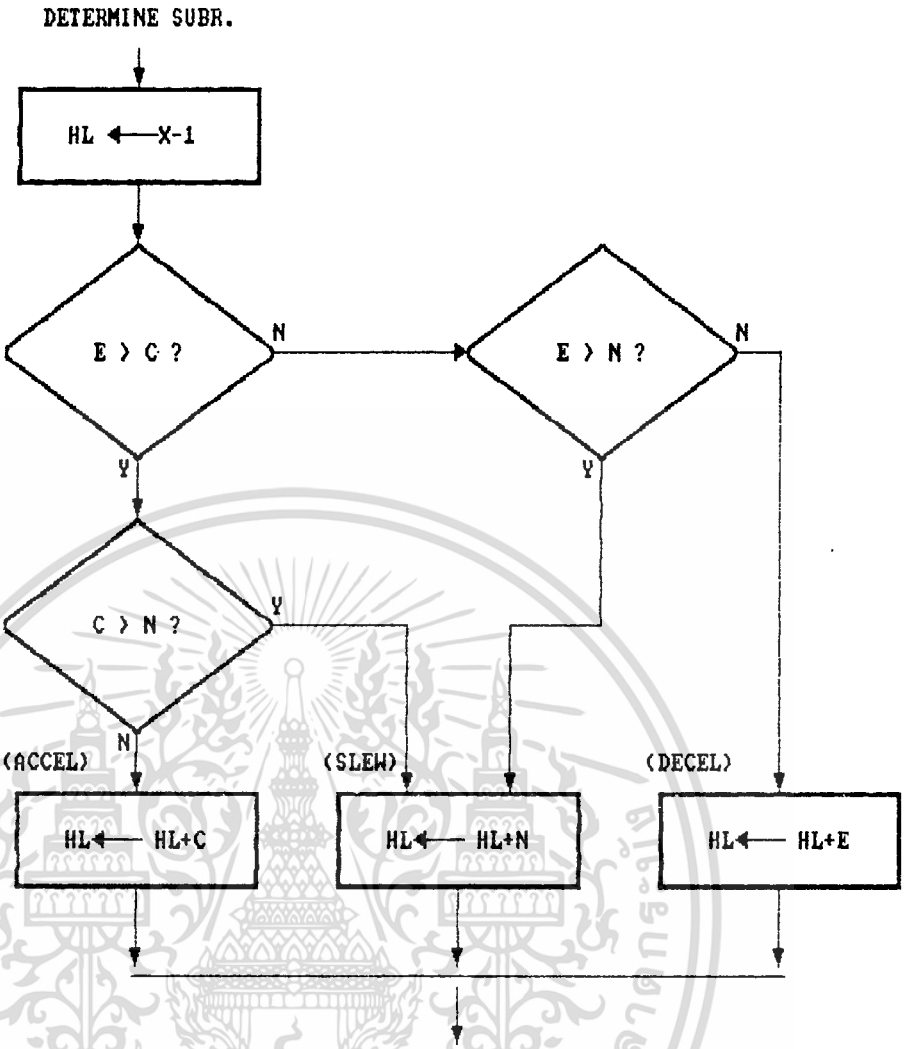
รีจีสเตอร์คู่ HL ก็ใช้ในการทำรายการตำแหน่งของข้อมูลบนทิศทางและจำนวนของสเตปในการหมุนอีกด้วย

3.2 Flow-Chart and Program

โฟล์ชาร์ทของโปรแกรมแสดงไว้ในรูปที่ 3.5







2500 A.D. Z80 Macro Assembler - Version 4.02a

Input Filename : testdri.asm
Output Filename : testdri.obj

```
1                                ;*********************************
2                                ; TEST DRIVE FOR STEPPING MOTOR *
3                                ;        ET-BOARD V 2.0          *
4                                ;       FILE TESTDRI.ASM         *
5                                ;*********************************
6                                ;LABEL      CODE OPERAND        COMMENT
7                                ;
8   8000                                    ORG  8000H
9                                ;
10          0000                 DRIVER:    EQU  00H            ;PORT A
11          0003                 CONT_PORT: EQU  03H
12          0006                 N:         EQU  6
13                               ;
14  8000    3E 81                START:     LD   A,81H          ;CONTROL PORT OUT
15  8002    D3 03                           OUT  (CONT_PORT),A;SET 8255 (OUTPUT PORT)
16  8004    3E 11                RESET:     LD   A,11H          ;00110011B -> A
17  8006    D3 00                           OUT  (DRIVER),A     ;OUT STEP
18  8008    47                              LD   B,A            ;A -> B
19  8009    21 75 80                        LD   HL,Y
20  800C    56                   LOOP:      LD   D,(HL)         ;LOAD ROTATE
21  800D    7A                              LD   A,D
22  800E    FE 02                           CP   2
23  8010    28 5C                           JR   Z,STOP         ;IF D=2 JMP TO STOP
24  8012    23                              INC  HL
25  8013    5E                              LD   E,(HL)         ;LOAD STEP COUNT
26  8014    0E 00                           LD   C,0
27  8016    E5                              PUSH HL             ;SAVE ADDRESS DATA
28  8017    7A                   ROTATE:    LD   A,D
29  8018    FE 00                           CP   0
30  801A    28 05                           JR   Z,LEFT         ;IF D=0 JMP TO LEFT
31  801C    78                   RIGHT:     LD   A,B            ;ROTATE RIGHT
32  801D    0F                              RRCA
33  801E    C3 26 80                        JP   STEP           ;JMP TO STEP
34  8021    78                   LEFT:      LD   A,B            ;ROTATE LEFT
35  8022    07                              RLCA
36  8023    C3 26 80                        JP   STEP
37  8026    D3 00                STEP:      OUT  (DRIVER),A     ;OUT STEP
38  8028    47                              LD   B,A
39  8029    0C                   COUNT:     INC  C              ;INCEASE C
40  802A    1D                              DEC  E              ;DECEASE E
41  802B    28 30                           JR   Z,DELAY        ;IF E=0 JMP TO DELAY
42  802D    21 6E 80             DETERM:    LD   HL,X-1
43  8030    79                              LD   A,C
44  8031    93                              SUB  E
45  8032    38 0D                           JR   C,CSUBN        ;IF E>C JMP TO CSUBN
46  8034    7B                   ESUBN:     LD   A,E
47  8035    FE 06                           CP   N              ;COMPARE E WITH N
```

```
48  8037  30 15                    JR   NC,SLEW      ;IF E>N JMP TO SLEW
49  8039  D5              DECEL:    PUSH DE
50  803A  16 00                     LD   D,0
51  803C  19                        ADD  HL,DE
52  803D  D1                        POP  DE
53  803E  C3 56 80                  JP   WAIT        ;JMP TO WAIT
54  8041  79              CSUBN:    LD   A,C
55  8042  FE 06                     CP   N           ;COMPARE C WITH N
56  8044  30 08                     JR   NC,SLEW     ;IF C>N JMP TO SLEW
57  8046  C5              ACCEL:    PUSH BC
58  8047  06 00                     LD   B,0
59  8049  09                        ADD  HL,BC
60  804A  C1                        POP  BC
61  804B  C3 56 80                  JP   WAIT        ;JMP TO WAIT
62  804E  C5              SLEW:     PUSH BC
63  804F  06 00                     LD   B,0
64  8051  0E 06                     LD   C,N
65  8053  09                        ADD  HL,BC
66  8054  C1                        POP  BC
67  8055  00                        NOP
68  8056  7E              WAIT:     LD   A,(HL)
69  8057  3D              LOOP1:    DEC  A
70  8058  20 FD                     JR   NZ,LOOP1    ;IF A NONZERO JMP LOOP1
71  805A  C3 17 80                  JP   ROTATE      ;JMP TO ROTATE
72  805D  3E FF           DELAY:    LD   A,0FFH
73  805F  C5                        PUSH BC
74  8060  06 FF                     LD   B,0FFH
75  8062  05              DELAY1:   DEC  B
76  8063  20 FD                     JR   NZ,DELAY1   ;IF B NONZERO JMPDELAY1
77  8065  3D              DELAY2:   DEC  A
78  8066  20 FD                     JR   NZ,DELAY2   ;IF A NONZERO JMPDELAY2
79  8068  C1                        POP  BC
80  8069  E1                        POP  HL
81  806A  23                        INC  HL
82  806B  C3 0C 80                  JP   LOOP        ;JMP TO LOOP
83  806E  76              STOP:     HALT
84  806F  FC              X:        DEFB 252         ;X TABLE
85  8070  B7                        DEFB 183
86  8071  95                        DEFB 149
87  8072  81                        DEFB 129
88  8073  73                        DEFB 115
89  8074  68                        DEFB 104
90  8075  01              Y:        DEFB 1           ;Y TABLE
91  8076  32                        DEFB 50
92  8077  00                        DEFB 0
93  8078  32                        DEFB 50
94  8079  02                        DEFB 2
95  807A                            END
```

Defined	Symbol Name		Value	References		
57	ACCEL		8046			
Pre	CODE		8000	8		
11	CONT_PORT	=	0003	15		
39	COUNT		8029			
54	CSUBN		8041	45		
Pre	DATA		0000			
49	DECEL		8039			
72	DELAY		805D	41		
75	DELAY1		8062	76		
77	DELAY2		8065	78		
42	DETERM		802D			
10	DRIVER	=	0000	17	37	
46	ESUBN		8034			
34	LEFT		8021	30		
20	LOOP		800C	82		
69	LOOP1		8057	70		
12	N	=	0006	47	55	64
16	RESET		8004			
31	RIGHT		801C			
28	ROTATE		8017	71		
62	SLEW		804E	48	56	
14	START		8000			
37	STEP		8026	33	36	
83	STOP		806E	23		
68	WAIT		8056	53	61	
84	X		806F	42		
90	Y		8075	19		

Lines Assembled : 95 Assembly Errors : 0

บทที่ 4
การทดลอง

วัตถุประสงค์

1. ศึกษาการขับสัญญาณการกระตุ้นจากไมโครโปรเซสเซอร์ในแบบต่างๆ
2. เปรียบเทียบการกระตุ้นสเตปปิ้งมอเตอร์ของแต่ละแบบ
3. นำผลการทดลองที่ได้ไปประยุกต์ใช้สำหรับการควบคุมตำแหน่ง

สเตปปิ้งมอเตอร์

สเตปปิ้งมอเตอร์เป็นมอเตอร์ทำงาน โดยเปลี่ยนพลังงานไฟฟ้าแบบ สวิทชิ่ง ที่ป้อนให้กับขดลวดสเตเตอร์ให้เป็นการเคลื่อนที่เชิงมุมบนแกนโรเตอร์ ในลักษณะการหมุนทีละสเตปตามสัญญาณอินพุท ของเฟสต่างๆ การเคลื่อนที่และหยุดในแต่ละสเตป จะมีความแม่นยำทางตำแหน่งสูงมาก แต่มีข้อจำกัดอยู่เหมือนกันคือ ขนาดของแรงบิดน้อยจะทำให้ไม่สามารถใช้งานหนักมากได้ แบบของ สเตปปิ้งมอเตอร์ที่ใช้งานมากพบบ่อยๆ ก็คือ แบบไฮบริด โรเตอร์ทำด้วยแม่เหล็กถาวร และเป็นชนิดขดลวดสเตเตอร์ 4 เฟส ไฟฟ้าที่จ่ายให้กับขดลวดสเตเตอร์ในแต่ละเฟสจะมีผลทำให้เกิดสนามแม่เหล็กในทิศทางนั้นๆ เกิดแรงผลักกับแม่เหล็กถาวร ที่ตัวโรเตอร์ไปในทิศทางหนึ่ง การสั่งให้โรเตอร์หมุนในทิศทางใดก็สามารถทำได้โดยการจ่ายกระแสไฟฟ้าแก่ขดสเตเตอร์ในเฟสต่างๆ อย่างถูกต้องต่อไป

จากสเตปปิ้งมอเตอร์มีขนาด 4เฟสนี้เราจะบังคับให้เกิดสเตปการหมุนได้เป็น 3 ลักษณะขึ้นอยู่กับ การป้อนพัลซ์ ดังนี้

1.One-excitation หรือ half drive เป็นการจ่ายกระแสให้กับสเตปปิ้งมอเตอร์ครั้งละหนึ่งเฟสคือ $\phi1, \phi2, \phi3, \phi4$ เรียงลำดับ หมุนเวียนกันไปแบบนี้ แรงบิดจะน้อยดังตารางที่ 4.1

------> No.	ϕ_1	ϕ_2	ϕ_3	ϕ_4
1	1	0	0	0
2	0	1	0	0
3	0	0	1	0
4	0	0	0	1

ตารางที่ 4.1 one-excitation (กำลังครึ่งเดียว)

2. Two-excitation หรือ full step เป็นการจ่ายกระแสให้กับสเตเตอร์ครั้งละ 2 เฟสพร้อมกัน ได้แก่ $\phi1\phi2, \phi2\phi3, \phi3\phi4, \phi4\phi1$ หมุนเวียนกันไปแบบนี้แรงบิดที่ได้จะมากกว่าแบบแรก

------> No.	ϕ_1	ϕ_2	ϕ_3	ϕ_4
1	1	1	0	0
2	0	1	1	0
3	0	0	1	1
4	1	0	0	1

ตารางที่ 4.2 two-excitation

------> No.	ϕ_1	ϕ_2	ϕ_3	ϕ_4
1	1	0	0	0
2	1	1	0	0
3	0	1	0	0
4	0	1	1	0
5	0	0	1	0
6	0	0	1	1
7	0	0	0	1
8	1	0	0	1

ตารางที่ 4.3 one-two excitation

3.One-two excitation หรือ half step เป็นการจ่ายกระแสให้กับขดลวดสเตเตอร์ 1เฟส สลับกันไป แบบนี้ ทำให้จำนวนของสเตปเพิ่มขึ้นเป็น 2 เท่า ของ 2 แบบแรก แต่แรงบิทจะน้อยกว่าแบบที่ 2 เล็กน้อย

ในการควบคุมเตปการทำงานของสเตปปิ้งมอเตอร์ทั้งสามรูปแบบ ที่ทราบมานั้นเราสามารถทำได้เป็น สองลักษณะคือ ใช้วงจรซีเควันลอจิกและการใช้ไมโครโปรเซสเซอร์ควบคุมจะใช้ไมโครโปรเซสเซอร์จำลองสัญญาณการกระตุ้นทั้งสามแบบที่กล่าวมานั้น ซึ่งสามารถทำได้โดยสะ

ดวกกว่าและเป็นที่นิยมมากกว่า เพราะไมโครโปรเซสเซอร์ได้ถูกลงมากในปัจจุบันและ ผู้ใช้สามารถพลิกแพลงการใช้ได้ตามต้องการ

ก่อนอื่น ต้องการให้เข้าใจส่วนการขับสัญญาณกระตุ้น ที่ได้จากไมโครโปรเซสเซอร์ก่อน เพราะสัญญาณการกระตุ้นที่ได้จาก 8255 นั้น เราทราบว่าเป็นระดับ TTL ซึ่งมีค่าโดยประมาณ 4 โวลท์เท่านั้น ซึ่งหากนักศึกษาต้องการใช้กับ สเตปปิ้งมอเตอร์ที่มีแรงดันสูงขึ้นกระแสมากขึ้นก็ต้องมีการสร้างส่วนขับเพิ่มเข้ามาซึ่งแบบง่ายมักจะใช้เป็นแบบ ดาร์ลิงตัน

<u>โปรแกรมควบคุมสเตปปิ้งในการกระตุ้นแบบต่างๆ</u>

จากวงจรในรูปที่ 5.3 มีแอดเดรสของพอร์ทดังนี้

พอร์ท A อยู่ที่ 00H และพอร์ท B อยู่ที่ 01H, พอร์ท C อยู่ที่ 02H และพอร์ทควบคุมอยู่ที่ 03H โดยจะใช้เพียง 2 พอร์ทในโปรแกรมคือ พอร์ท A ใช้ชื่อว่า PA และพอร์ทควบคุมใช้ชื่อว่า PCC

<u>โปรแกรมแรก</u> เป็นการกระตุ้นการทำงานโดยใช้แบบ One-excitation ค่าที่ส่งไปคือข้อมูลที่อยู่ในหน่วยความทรงจำ data แล้วใช้วิธีการส่งออกควบคุมแต่ละครั้ง

<u>โปรแกรมที่สอง</u> เป็นการกระตุ้นการทำงานโดยใช้แบบ one-excitation

<u>โปรแกรมที่สาม</u> เป็นการกระตุ้นแบบ one-two excitation ซึ่งข้อมูลในการส่งควบคุมจะถูกเก็บไว้ที่ตัวแปร Data แล้วใช้หลักการเช่นเดียวกับโปรแกรมแรก คือเลื่อนบิทแล้วจึงส่งค่าออกไป เพียงแต่จะมีการสลับค่าข้อมูลก่อนที่จะทำการส่งเท่านั้น

โปรแกรมแสดง จะสังเกตุได้ว่าทุกครั้งที่มีการส่งค่าข้อมูลออกไปควบคุมสเตปปิ้งมอเตอร์นั้นจะต้องมีดีเลย์เวลาก่อนเสมอ ทั้งนี้เนื่องจากว่าการทำงานของไมโครโปรเซสเซอร์นั้นทำงานได้เร็วมาก มอเตอร์อาจเกิดการ ออสซิลเลทได้ทั้งนี้ขึ้นอยู่กับ ข้อจำกัดของสเตปปิ้งเตอร์ในแต่ละตัวไป

การใช้ไมโครโปรเซสเซอร์ควบคุม สามารถพลิกแพลงได้หลายรูปแบบ เช่น กำหนดเดินหน้า ถอยหลัง บังคับจำนวนสเตปในการหมุนแต่ละครั้ง ยิ่งถ้าเราใช้สเตปปิ้งมอเตอร์ 2 ตัว เพื่อควบคุมในลักษณะของแกน x,แกน Y แล้ว เราก็จะสามารถจดจำตำแหน่งต่างๆ นำไปประยุกต์ใช้งานเป็นเครื่องเขียนภาพ (Plotter) และแขนหุ่นยนต์ในอุตสาหกรรมเป็นต้น

2500 A.D. Z80 Macro Assembler - Version 4.02a

Input Filename : step.asm
Output Filename : step.obj

```
1                                       ;************************
2                                       ; TEST STEPPING MOTOR  *
3                                       ;  FOR ONE-EXCITATION  *
4                                       ;************************
5                               ;
6   8000                                ORG     8000h
7           0000                PA:     EQU     00H
8           0003                PCC:    EQU     03H
9                               ;
10  8000    3E 80               START:  LD      A,80H       ;CONTROL PORT OUTPUT
11  8002    D3 03                       OUT     (PCC),A
12  8004    3E 11                       LD      A,00010001B ;ONE-PHASE DATA
13  8006    D3 00               LOOP:   OUT     (PA),A      ;OUT STEP
14  8008    0F                          RRCA
15  8009    CD 0F 80                    CALL    DELAY
16  800C    20 F8                       JR      NZ,LOOP
17  800E    76                          HALT
18                              ;
19  800F    F5                  DELAY:  PUSH    AF
20  8010    21 00 20                    LD      HL,2000H
21  8013    2B                  DEL1:   DEC     HL
22  8014    7C                          LD      A,H
23  8015    B5                          OR      L
24  8016    20 FB                       JR      NZ,DEL1
25  8018    F1                          POP     AF
26  8019    C9                          RET
27  801A                                END
```

2500 A.D. Z80 Macro Assembler - Version 4.02a

Input Filename : step1.asm
Output Filename : step1.obj

```
 1                                      ;************************
 2                                      ; TEST STEPPING MOTOR  *
 3                                      ;  FOR TWO-EXCITATION  *
 4                                      ;************************
 5                              ;
 6  8000                                ORG     8000H
 7          0000                PA:     EQU     00H
 8          0003                PCC:    EQU     03H
 9                              ;
10  8000    3E 80               START:  LD      A,80H       ;CONTROL PORT OUTPUT
11  8002    D3 03                       OUT     (PCC),A
12  8004    3E 33                       LD      A,00110011B ;TWO-PHASE DATA
13  8006    D3 00               LOOP:   OUT     (PA),A      ;OUT STEP
14  8008    07                          RLCA
15  8009    CD 0E 80                    CALL    DELAY
16  800C    18 F8                       JR      LOOP
17                              ;
18  800E    F5                  DELAY:  PUSH    AF
19  800F    21 00 20                    LD      HL,2000H    ;STEP DELAY
20  8012    2B                  DEL1:   DEC     HL
21  8013    7C                          LD      A,H
22  8014    B5                          OR      L
23  8015    20 FB                       JR      NZ,DEL1
24  8017    F1                          POP     AF
25  8018    C9                          RET
26  8019                                END
```

2500 A.D. Z80 Macro Assembler - Version 4.02a

Input Filename : step2.asm
Output Filename : step2.obj

```
                                        ;**************************
                                        ;  TEST STEPPING MOTOR   *
                                        ; FOR ONE-TWO EXCITATION *
                                        ;**************************
                                ;
8000                            ORG     8000H
        0000                    PA:     EQU     00H
        0003                    PCC:    EQU     03H
                                ;
8000    3E 80                   START:  LD      A,80H       ;CONTROL PORT OUTPUT
8002    D3 03                           OUT     (PCC),A
8004    01 33 11                        LD      BC,1133H    ;B=ONE-PHASE,C=TWO-PHASE
8007    78                              LD      A,B
8008    D3 00                   LOOP:   OUT     (PA),A      ;OUT STEP
800A    CD 13 80                        CALL    DELAY
800D    07                              RLCA
800E    47                              LD      B,A         ;ALTERNATE PHASE
800F    79                              LD      A,C
8010    48                              LD      C,B
8011    18 F5                           JR      LOOP
                                ;
8013    F5                      DELAY:  PUSH    AF
8014    21 00 20                        LD      HL,2000H    ;STEP DELAY
8017    2B                      DEL1:   DEC     HL
8018    7C                              LD      A,H
8019    B5                              OR      L
801A    20 FB                           JR      NZ,DEL1
801C    F1                              POP     AF
801D    C9                              RET
801E                                    END
```

สรุปผลการทดลอง

1. จากการทดลองการขับสเตปปิ้งมอเตอร์ โดยการใช้การกระตุ้นแบบ 1 เฟส และ 2 เฟส (one-excitation and two-excitation) พบว่าที่ความเร็วต่ำ จะทำให้เกิดการสั่นสะเทือนและเกิดเสียงดังได้ แต่จะให้ความหนาแน่นของสเตปได้ดี

2. จากการทดลองการขับสเตปปิ้งมอเตอร์ โดยการใช้การกระตุ้นแบบ 1-2 เฟส (one-two excitation) สามารถที่จะแก้ไขปัญหาการสั่นสะเทือนได้

ข้อเสียของการกระตุ้นแบบนี้ จะทำให้หนาแน่นของการสเตปได้ไม่ดี แต่จะให้ความละเอียดของการสเตปได้มากกว่าการกระตุ้นแบบ 1 เฟส และ 2 เฟส ถึง 2 เท่า

3. จากการกระตุ้นทั้งสามแบบ ที่ทำการทดลอง พบว่า ที่ความเร็วเพิ่มขึ้น จะทำให้แรงบิด (ทอร์ค) น้อยลง

บทที่ 5
ระบบปรับทิศทางสายอากาศด้วยสเตปปิ้งมอเตอร์

การทำงานของ ANT. POSITIONER WITH STEPPING MOTOR

การทำงานของระบบนั้นเป็นการปรับทิศทางหรือตำแหน่งของสายอากาศ โดยใช้ไมโครโปรเซสเซอร์ของ ZILOG เบอร์ Z-80 CPU เป็นหัวใจในการทำงานทั้งหมด ทำหน้าที่สั่งงานให้สเตปปิ้งมอเตอร์ทำงานตามขั้นตอนของโปรแกรม โดยจะรับการสั่งงานจากคีย์บอร์ดและส่วนของการแสดงผลจะใช้ 7-Segment จำนวน 4 Digit แสดงตำแหน่งเป็นจำนวนองศา การควบคุมการทำงานของสเตปปิ้งมอเตอร์ใช้การควบคุมแบบเปิด (Open loop) ซึ่งการควบคุมแบบนี้จะอาศัยความแน่นอน และความเที่ยงตรงของการหมุนเป็นสเตป โดยการหมุนแต่ละครั้งจะหมุนไปเป็นมุม 1.8 องศา(1.8/STEP) ฉะนั้นการสั่งงานให้สเตปปิ้งมอเตอร์ เราสามารถนับจำนวนองศาที่จำนวนพัลซ์ที่เราป้อนได้อย่างแม่นยำ โดยไม่ต้องการการป้อนกลับ ทั้งนี้ทั้งนั้นจะต้องคำนึงถึงโหลดด้วย ในการเพิ่มการรับโหลดให้ได้มากนั้น จะใช้อัตราการทดของเกียร์

ระบบของเครื่องปรับทิศทางสายอากาศด้วยสเตปปิ้งมอเตอร์ แสดงในรูปที่ 5.1

5.1 CONTROLLER BOARD

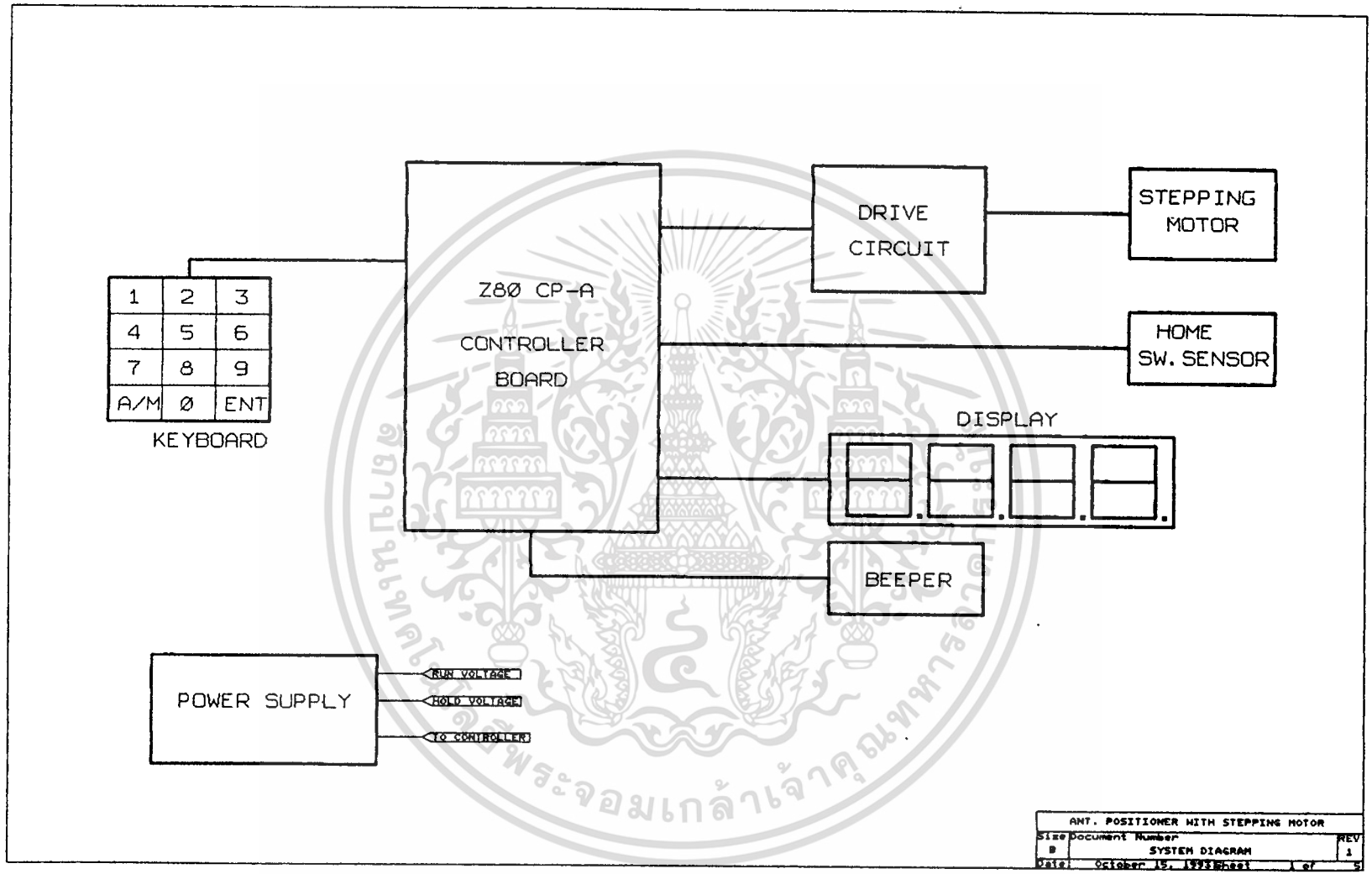
CONTROLLER BOARD ใช้ไมโครโปรเซสเซอร์ เบอร์ Z-80 CPU ทำหน้าที่สั่งงานและประมวลผล โดยใช้หน่วยความจำ ROM เบอร์ 2732 เก็บโปรแกรมมอเตอร์ ซึ่งเขียนด้วยภาษาแอสแซมบลี้ และมีหน่วยความจำ RAM เบอร์ 6116 เก็บค่าสแตคและบัฟเฟอร์ข้อมูลต่างๆ ส่วนไอซี 8255 จะเป็น พอร์ทอินพุท/เอ้าท์พุท ของ CONTROLLER BOARD นี้

นอกจากนี้ ยังมีวงจร Regulator 5V เพื่อจ่ายแรงดันคงที่ กับ CONTROLLER BOARD และวงจรควบคุมต่างๆ ที่ต่อร่วมอีกด้วย

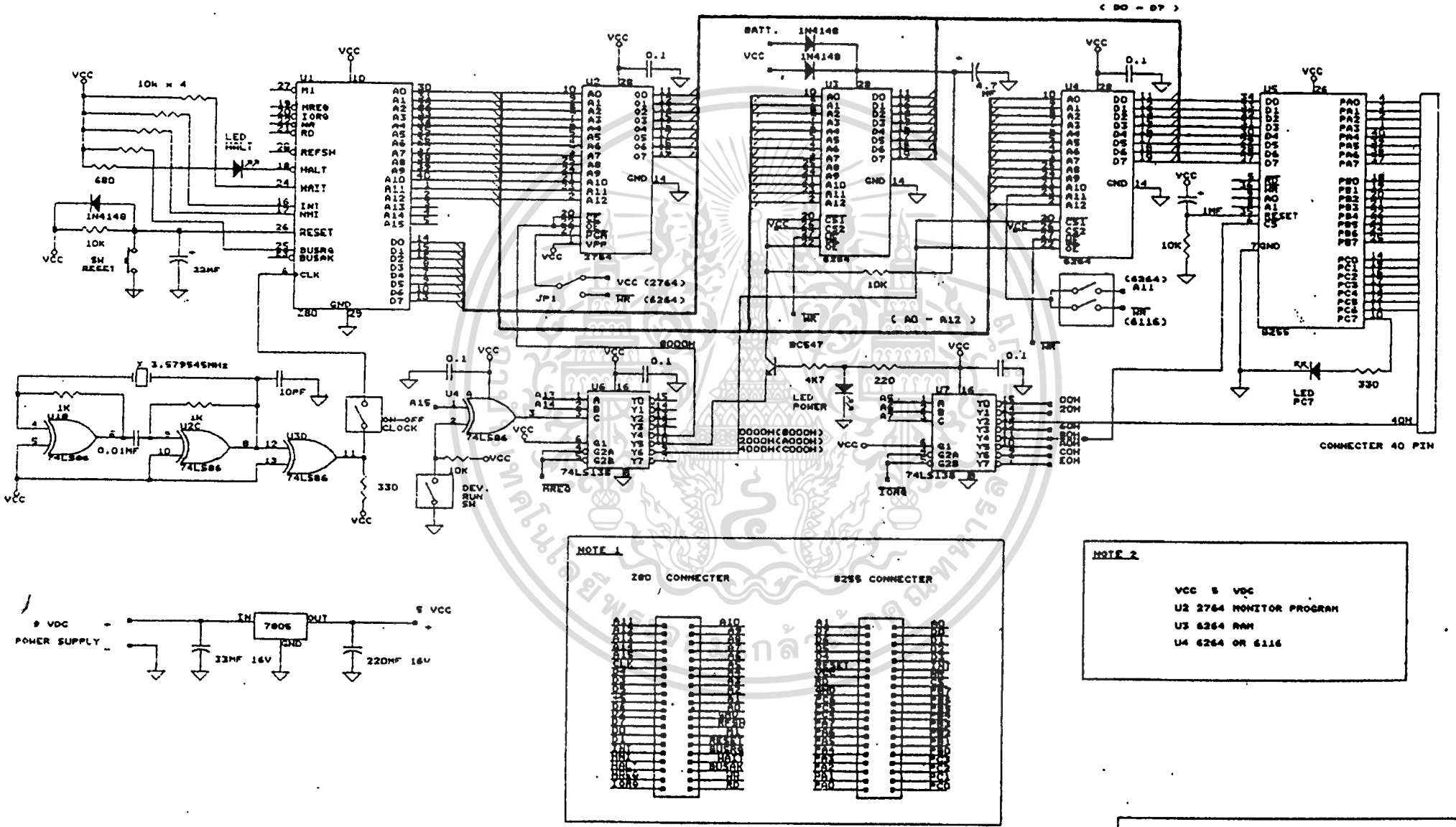
วงจรของ CONTROLLER BOARD แสดงในรูปที่ 5.2

5.2 วงจรขับสเตปปิ้งมอเตอร์

วงจรขับสเตปปิ้งมอเตอร์ ซึ่งแสดงอยู่ในรูปที่ 5.3 ใช้อุปกรณ์คัปปลิ้ง คือ ออฟโตไอโซเลเตอร์ (ออฟโตคัปเปลอร์) เบอร์ 4N33 ซึ่งมีตัวรับเป็นโฟโตดาร์ลิงตัน เป็นวงจรอินเตอร์เฟสเพื่อแยกส่วนหรือ ไอโซเลชั่น (ISOLATION) ระหว่างวงจรควบคุมกับวงจรขับ เพื่อป้องกันการ

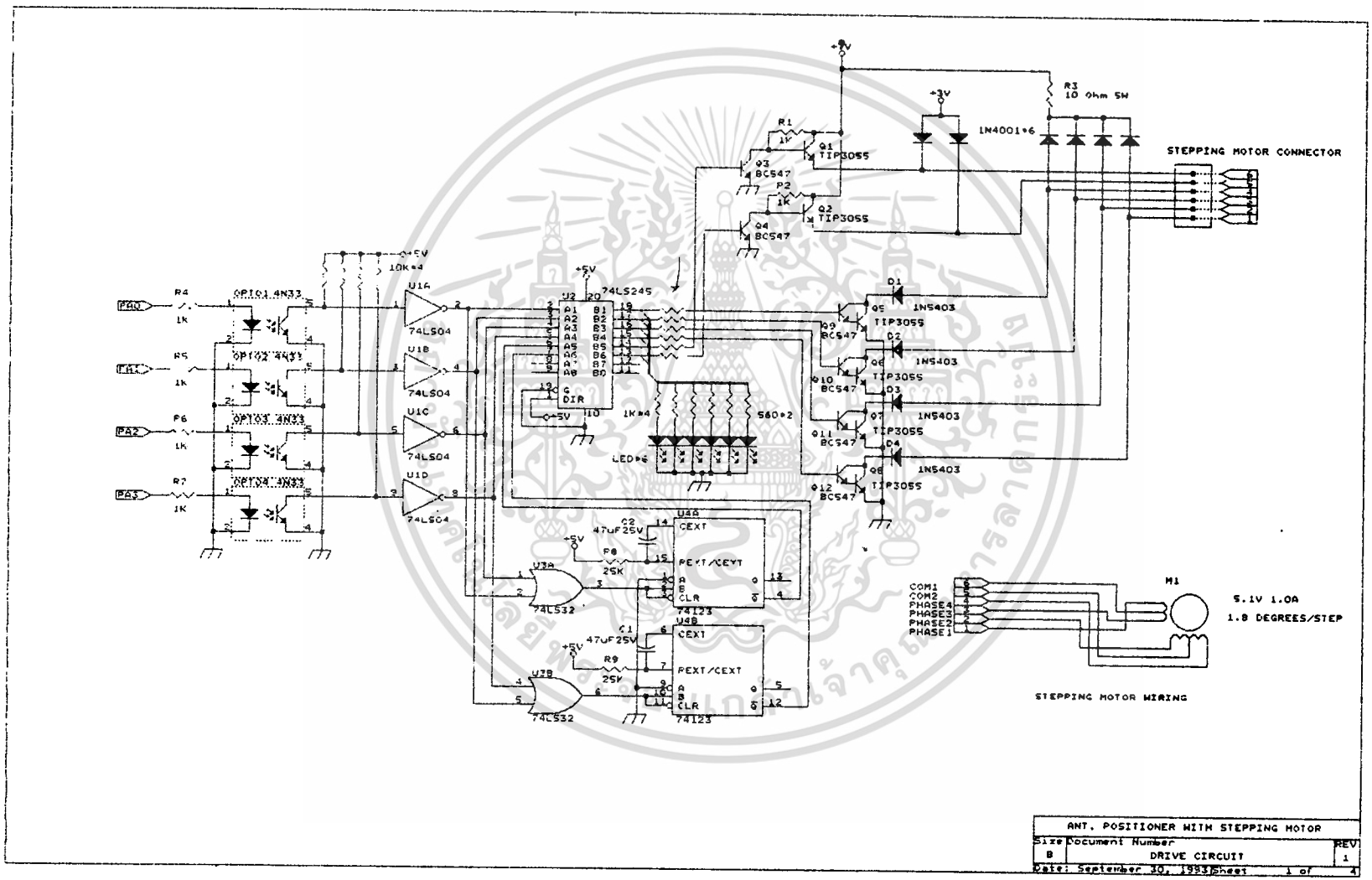


รูปที่ 5.1 ระบบของเครื่องปรับทิศทางสายอากาศด้วยสเตปปิ้งมอเตอร์



รูปที่ 5.2 แสดงวงจรของ Controller Board

ETT Co.,LTD		
Title: Z80 CPA		
Size: A	Document Number	REV: 1
Date: January 24,1990	Sheet 1 of 1	



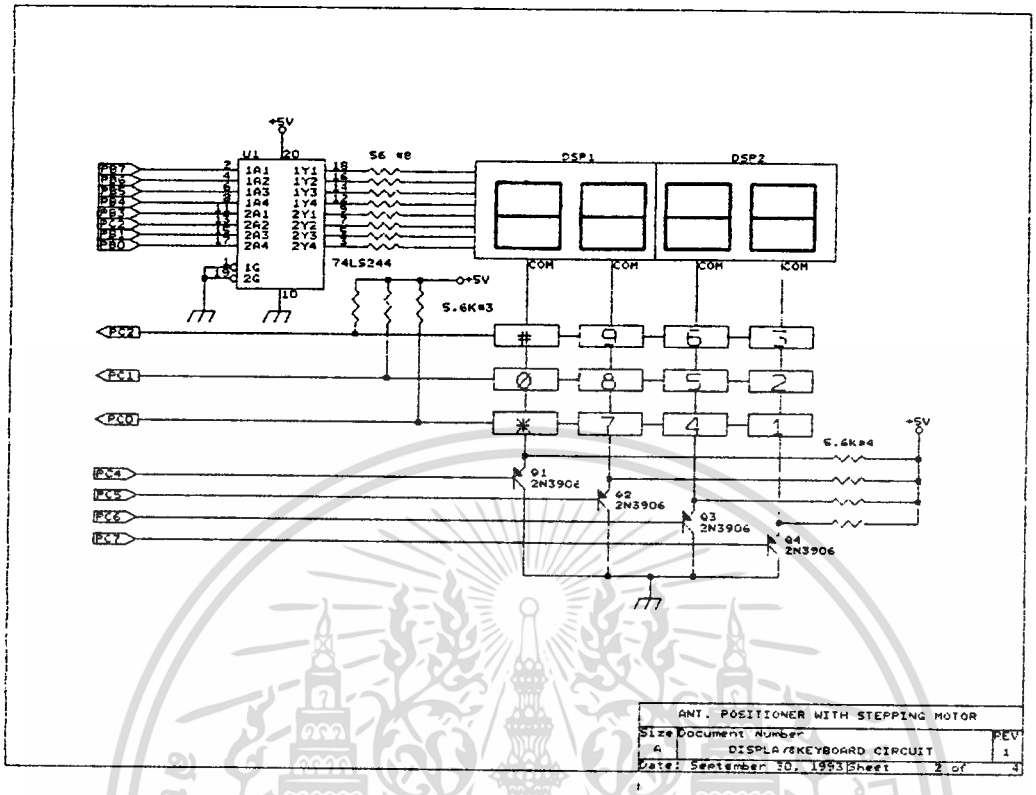
รูปที่ 5.3 แสดงวงจรขับสเตปปิ้งมอเตอร์

ดึงกระแสจากวงจรควบคุม ซึ่งจะทำให้วงจรควบคุมอาจเกิดความเสียหายขึ้นได้ R 10K 4 ตัวที่ต่อกับขา 5 ของออฟโตคัปเปลอร์ จะหน้าที่ทำให้แรงดันที่อินพุทของ IC เบอร์ 74LS04 (NOT GATE) มีค่าเป็น "1" ในกรณีที่ไม่มีพัลซ์ป้อนให้ออฟโตคัปเปลอร์ IC เบอร์ 74LS04 จะเปลี่ยนสถานะจาก "1" เป็น "0" แต่ถ้ามีพัลซ์เข้ามาก็จะให้แรงดันที่อินพุท NOT GATE เป็น "0" และเอ้าท์พุทจะเป็น "1"

IC 74LS245เป็น IC ที่ทำหน้าที่เป็นบัฟเฟอร์และสามารถให้กระแสเอ้าท์พุทได้สูงสุดถึง 200 mA ซึ่งเพียงพอที่จะขับทรานซิสเตอร์ที่ต่อดาร์ลิงตันอยู่ ในแต่ละเฟส ทรานซิสเตอร์ BC 547 ซึ่งต่อแบบดาร์ลิงตันกับเบอร์ TIP3055 เพื่อจะขยายกระแสให้สูงมากพอที่จะขับมอเตอร์ในแต่ละเฟสได้ เพราะกระแสที่ใช้ขับในแต่ละเฟสนั้นใช้ถึง 1.0 แอมป์ที่แรงดัน 5.1 V เนื่องจากทรานซิสเตอร์เบอร์ TIP3055 ต้องขับกระแสสูงเช่นนี้ ย่อมเกิดความร้อนขึ้นมาก จึงจำเป็นต้องใช้Heat Sink เป็นตัวช่วยระบายความร้อนที่เกิดขึ้น

ในการควบคุมสเตปปิ้งมอเตอร์ให้มีประสิทธิภาพสูงสุดนั้น จะต้องมีการ HOLD กระแสต่ำๆ ไว้หนึ่งเฟส เพื่อจะล็อคการหมุนของสเตปปิ้งมอเตอร์ในขณะที่ไม่มีการสั่งงานให้สเตปปิ้งมอเตอร์หมุนของหมุน ซึ่งมีผลทำให้การหยุดของสเตปปิ้งมอเตอร์เป็นไปอย่างมีประสิทธิภาพและแม่นยำมากจึงใช้วงจร โมโนสเตเบิลมัลติไวเบรเตอร์ มาช่วย โดยใช้ OR GATE เบอร์ 74LS32 และ ไอซีเบอร์ 74123 ไอซีเบอร์ 74123 เป็นไอซีโมโนสเตเบิลแบบรับการกระตุ้นซ้ำ โดยจะมี 2 ตัวอยู่ในตัวเดียว ตัวแรกจะใช้ขับเฟส 1 และเฟส 3 ส่วนตัวที่ 2 จะใช้ขับเฟส 2 และเฟส 4 ก็จะทำงาน ถ้ามีการสั่งงานให้สเตปปิ้งมอเตอร์ ไอซี 74123 ก็จะหน้าที่กำหนดให้ ทรานซิสเตอร์ TIP3055 (Q_1,Q_2) ทำงานสลับกันตามพัลซ์ที่ป้อนให้ในแต่ละเฟส ซึ่งจะจ่ายแรงดันสูงให้กับสเตปปิ้งมอเตอร์ หรือจ่ายกระแสสูงขณะทำงาน (RUN CURRENT) ถ้ามีการหยุดป้อนพัลซ์เมื่อไร ทรานซิสเตอร์ Q_1,Q_2 ก็จะหยุดจ่ายกระแสตามค่าเวลาของวงจรโมโนสเตเบิล จากนั้นแรงดันต่ำ (+3V) ก็จะป้อนเข้าที่ขดลวดของสเตปปิ้งมอเตอร์ซึ่งมีพัลซ์บวกล็อคอยู่หนึ่งเฟส กระแสที่ใช้ล็อคนี้เรียกว่า HOLD CURRENT ซึ่งมีค่ากับ 20 % ของ RUN CURRENT

เพื่อการป้องกันความเสียหายที่จะเกิดขึ้นกับทรานซิสเตอร์เพาเวอร์ เนื่องจากปัญหาของแรงดันไฟกลับ (BACK EMF) ที่เกิดขึ้นเนื่องจากการหยุดการป้อนไฟให้ขดลวดสเตปปิ้งมอเตอร์จึงใช้ไดโอด 1N 4001 4 ตัวกับรีซิสเตอร์ 10 Ohm 5W เป็นตัวลดกระแสที่เกิดจาก"BACK EMF" ได้ ไดโอด D_1-D_4 (1N5403) ป้องกันสไปด์โวลท์เตจเนื่องจากขดลวด ที่เกิดขึ้นเมื่อทรานซิสเตอร์หยุดทำงาน



รูปที่ 5.4 วงจรแสดงผลและคีย์บอร์ด

5.3 วงจรแสดงผลและคีย์บอร์ด

ส่วนแสดงผลใช้ Sevent Segment หรือ LED 7 ส่วนจำนวน 4ชุด ใช้ไอซีเบอร์ 74LS244 เป็นตัวขับส่วนแสดงผลจาก PORT B การแสดงผลจะใช้การ MULTIPLEX จะสแกนส่วนแสดงผลที่ละชุดโดยใช้ PORT C (UPPER) เป็นเอ้าท์พุทสำหรับขับทรานซิสเตอร์ 2N 3906 การสแกนจะแสดงผลชุดละ 2 mS

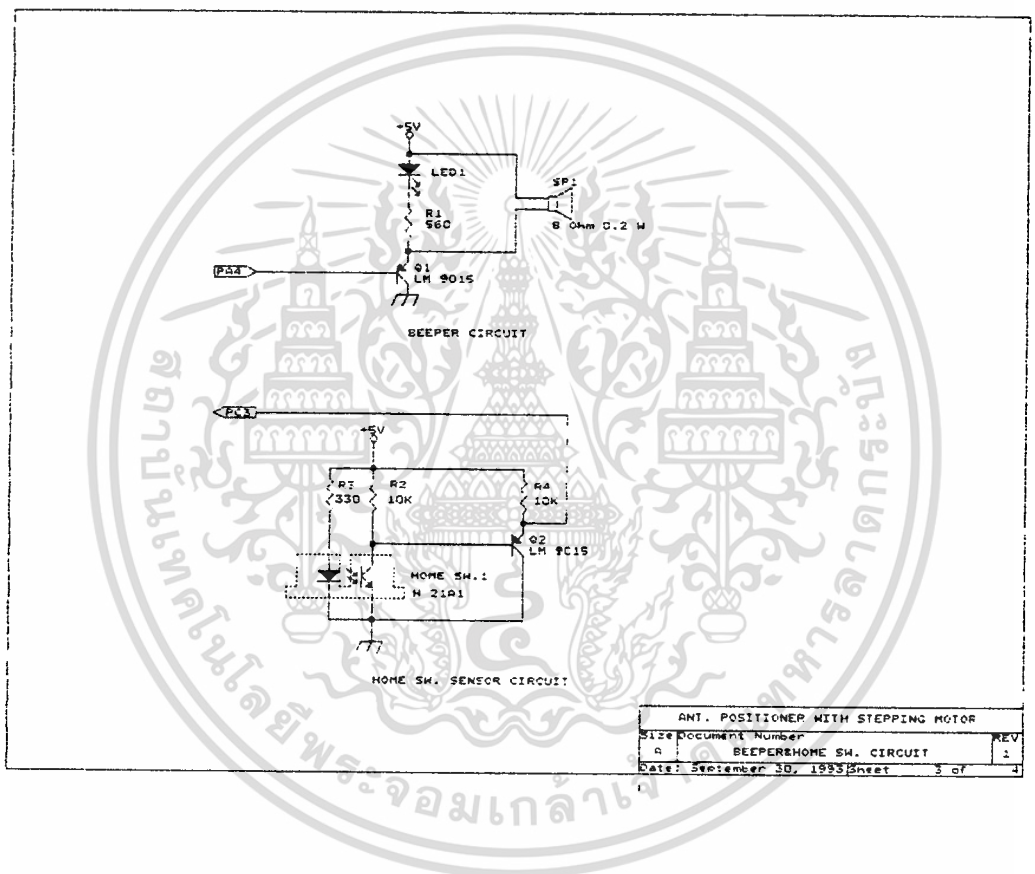
คีย์บอร์ดจะถูกสแกนพร้อมๆ กับการสแกนส่วนแสดงผล เมื่อคีย์บอร์ดใดถูกกด ก็จะถูกนำเข้าไปเปลี่ยนเป็นรหัสของคีย์บอร์ดนั้นๆ โดยใช้ PORT C (LOWER) ดังแสดงในรูปที่ 5.4

5.4 วงจร Beeper และ Home SW. Sensor

วงจร Beeper จะใช้ Port A4 เป็นเอ้าท์พุทถ้า เป็น "LOW" จะทำให้ทรานซิสเตอร์นำกระแส เกิดเสียง BEEP ขึ้นพร้อมกับ LED จะสว่างด้วยถ้าเอ้าท์พุท PA4 เป็น"HIGH" ทรานซิสเตอร์หยุดนำกระแส เกิดเสียง Beep ออกลำโพง LED ก็จะดับ

ส่วนของ Home SW. Sensor จะใช้ออฟโตคัปเปลอร์ เบอร์ H21A1 ต่อร่วมกับทราน

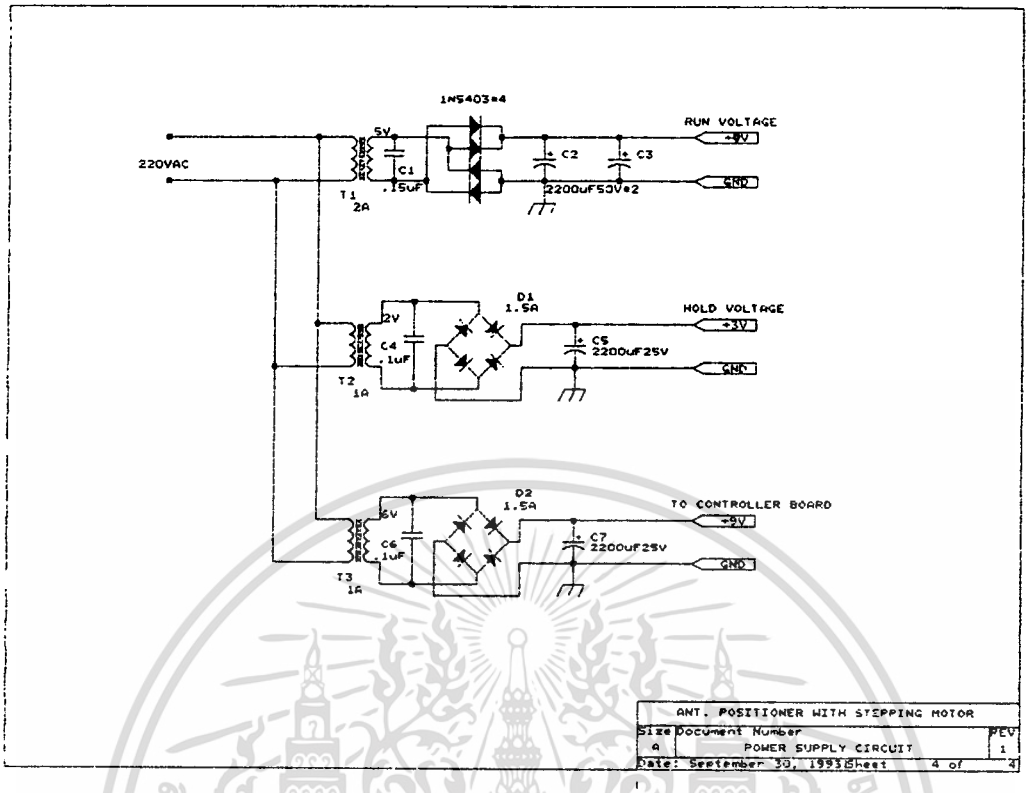
ชิสเตอร์ เบอร์ LM 9015 เพื่อจะตรวจสอบหรือหาตำแหน่งเริ่มต้น (0) ณ ตำแหน่งจะไม่มีอะไรมาบังแสงจะทำทรานชิสเตอร์ ในตัวออฟโตคัปเปลอร์นำกระแสทำให้ขา 4 มีสัญญาณ "LOW"ที่ขา E ส่งไปยัง PORT C3 ซึ่ง PORT C3 นี้จะเป็นอินพุทพอร์ทสำหรับตรวจตำแหน่ง HOME. วงจร Beeper และ Home SW. Sensor แสดงในรูปที่ 5.5



รูปที่ 5.5 แสดงวงจร Beeper และ Home SW. Sensor

5.5 วงจรแหล่งจ่ายไฟ

แหล่งจ่ายไฟ (POWER SUPPLY) จะมี 3 ชุด ชุดแรงดันไฟสูง +9V (RUN VOLTAGE) จ่ายสเตปปิ้งมอเตอร์ขณะทำงาน ชุดที่สองเป็นแรงดันไฟทำงาน ชุดที่ต่ำ +3V (HOLD VOLTAGE) เป็นแรงดันที่ใช้สำหรับจ่ายให้ขดลวดของสเตปปิ้งมอเตอร์ในขณะหยุดหมุนหรืออยู่นิ่ง ส่วนชุดที่สาม เป็นแรงดัน +9v จ่ายให้กับชุดของวงจรควบคุมระบบทั้งหมด



รูปที่ 5.5 วงจรแหล่งจ่ายไฟ

5.6 การใช้งานเครื่องปรับทิศทางสายอากาศด้วยสเตปปิ้งมอเตอร์

หลังจากเปิด SW.เพื่อจ่ายไฟเข้าเครื่องจะแสดงผล"Ant.p" แล้วทำการค้นหาตำแหน่งเริ่มต้น (HOME POSITION) ในขณะที่ทำการค้นหาก็จะแสดงผล "init"ด้วย ถ้าพบตำแหน่งเริ่มต้นแล้วสเตปปิ้งมอเตอร์จะหมุนกลับอีก 10 Stepแล้วค้นหาด้วยความเร็วต่ำอีกครั้ง เพื่อความแม่นยำของตำแหน่งเริ่มต้น แต่ถ้าการค้นหาผิดพลาดเช่น HOME SW.เสียเครื่องจะฟ้อง "HErr" ออกทางส่วนแสดงผล

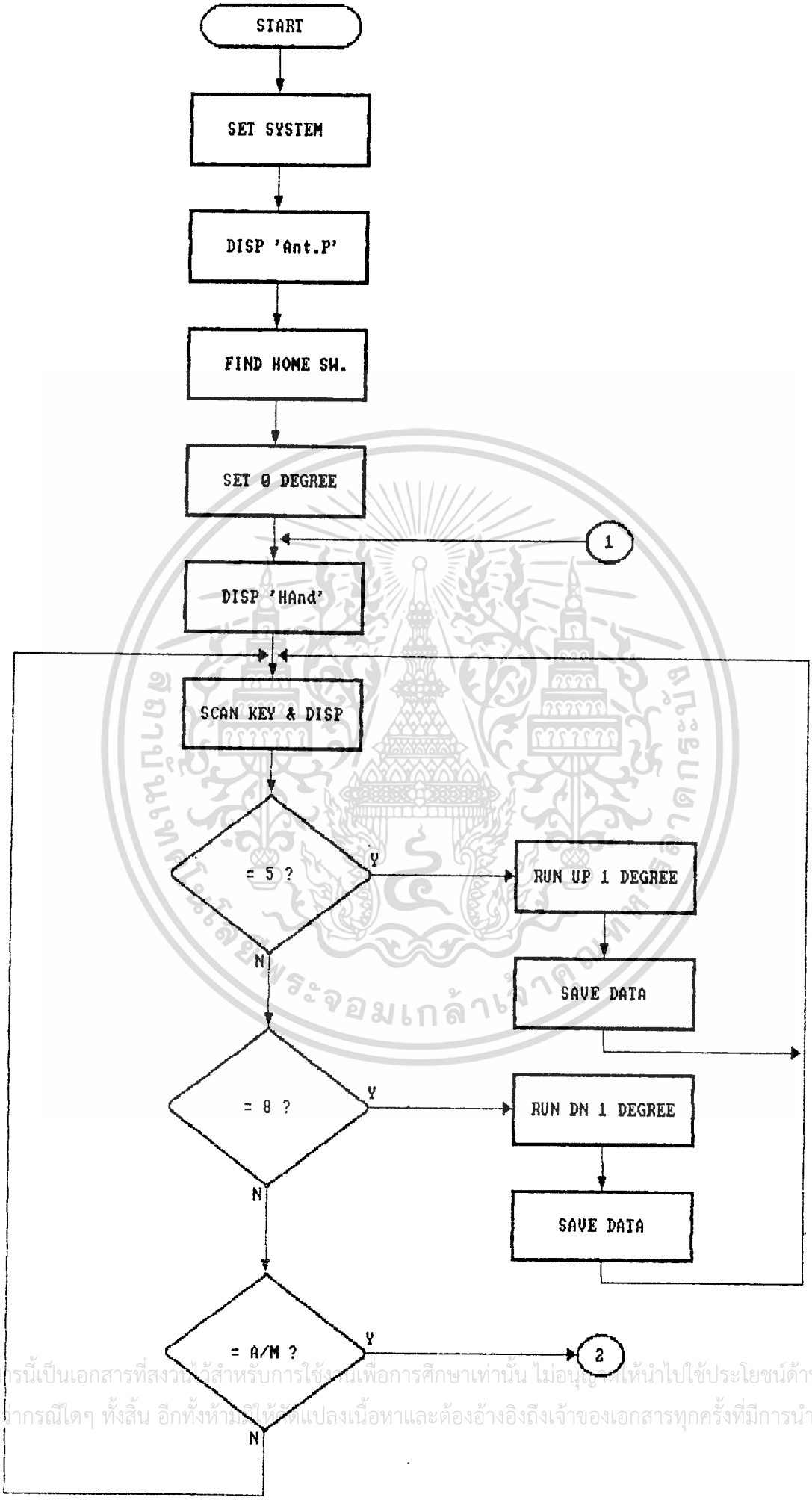
หลังจากพบตำแหน่งเริ่มต้นที่แน่นอนแล้ว เครื่องจะเข้าโหมด MANUAL ให้โดยอัตโนมัติ โดยจะแสดงผล "HAnd" ออกมาแล้วแสดงผลตำแหน่งเริ่มต้นเป็นศูนย์องศา โหมด MANUAL จะใช้คีย์เพียง 3 คีย์ คือ คีย์ 5 ใช้สำหรับสั่งให้หมุนไปทางบวกหรือตำแหน่งองศาที่เพิ่มขึ้น การเพิ่มขึ้นจะเพิ่ม 1 องศาต่อการกดคีย์ 1 ครั้ง แต่ถ้ากดคีย์นี้ค้างไว้ก็จะเพิ่มขึ้นอย่างต่อเนื่องด้วยอัตราเร่ง แล้วก็หมุนด้วยความเร็วสูงคงที่ค่าหนึ่งไปจนกว่าจะปล่อยคีย์ ส่วนคีย์ที่ใช้อีกคีย์หนึ่งคือ คีย์ 8 ใช้สำหรับสั่งให้หมุนไปทางลบหรือตำแหน่งองศาที่ลดลง การลดลงก็จะลดลง 1 องศาต่อการกดคีย์ 1 ครั้ง แต่ถ้ากดคีย์นี้ค้างไว้ก็จะลดลงอย่างต่อเนื่องด้วยอัตราเร่ง แล้วก็หมุนด้วย

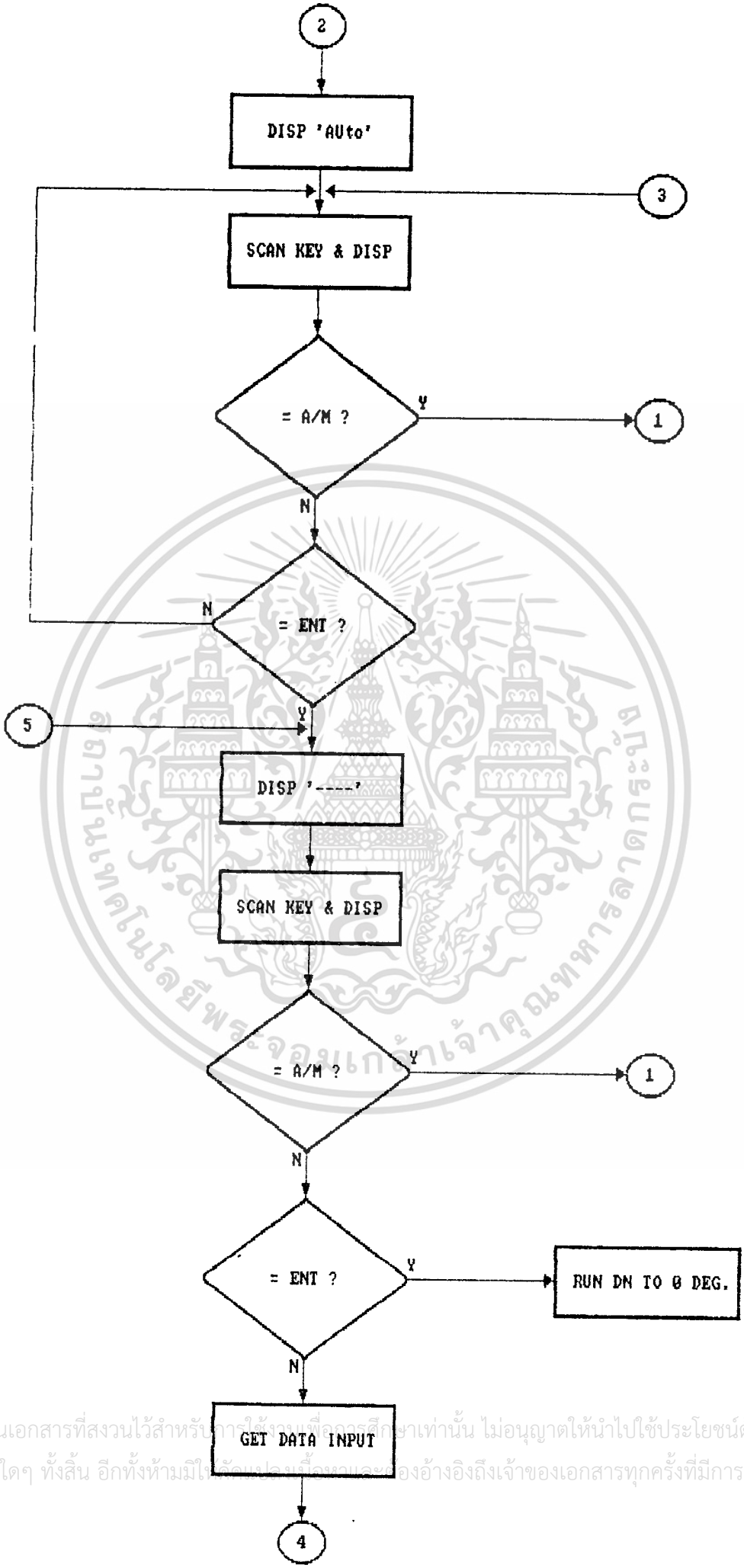
ความเร็วสูงคงที่ค่าหนึ่งไปจนกว่าจะปล่อยคีย์ ในโหมด MANUAL นี้จะตรวจสอบตำแหน่งสูงสุดและตำแหน่งต่ำสุดด้วย ถ้าถึงตำแหน่ง 360 แล้วแต่จะมีการกดคีย์ (5)อีก เครื่องจะฟ้องว่า"-Err" ส่วนการตรวจสอบตำแหน่งต่ำสุด ถ้าถึง 0 องศาแล้ว แต่ยังมีการกดคีย์ (8)อีก เครื่องก็จะฟ้อง "-Err" เช่นกัน ส่วนคีย์สุดท้ายที่ใช้ในโหมดนี้คือคีย์ A/M (Auto/Manual) ถ้ากดคีย์นี้ในขณะอยู่ในโหมด MANUAL เครื่องก็จะเปลี่ยนโหมดการทำงานเป็นแบบ AUTO ทันทีในทางกลับกันถ้าอยู่ในโหมด AUTO ถ้ากดคีย์นี้เครื่องจะเข้าโหมด MANUAL ทันที

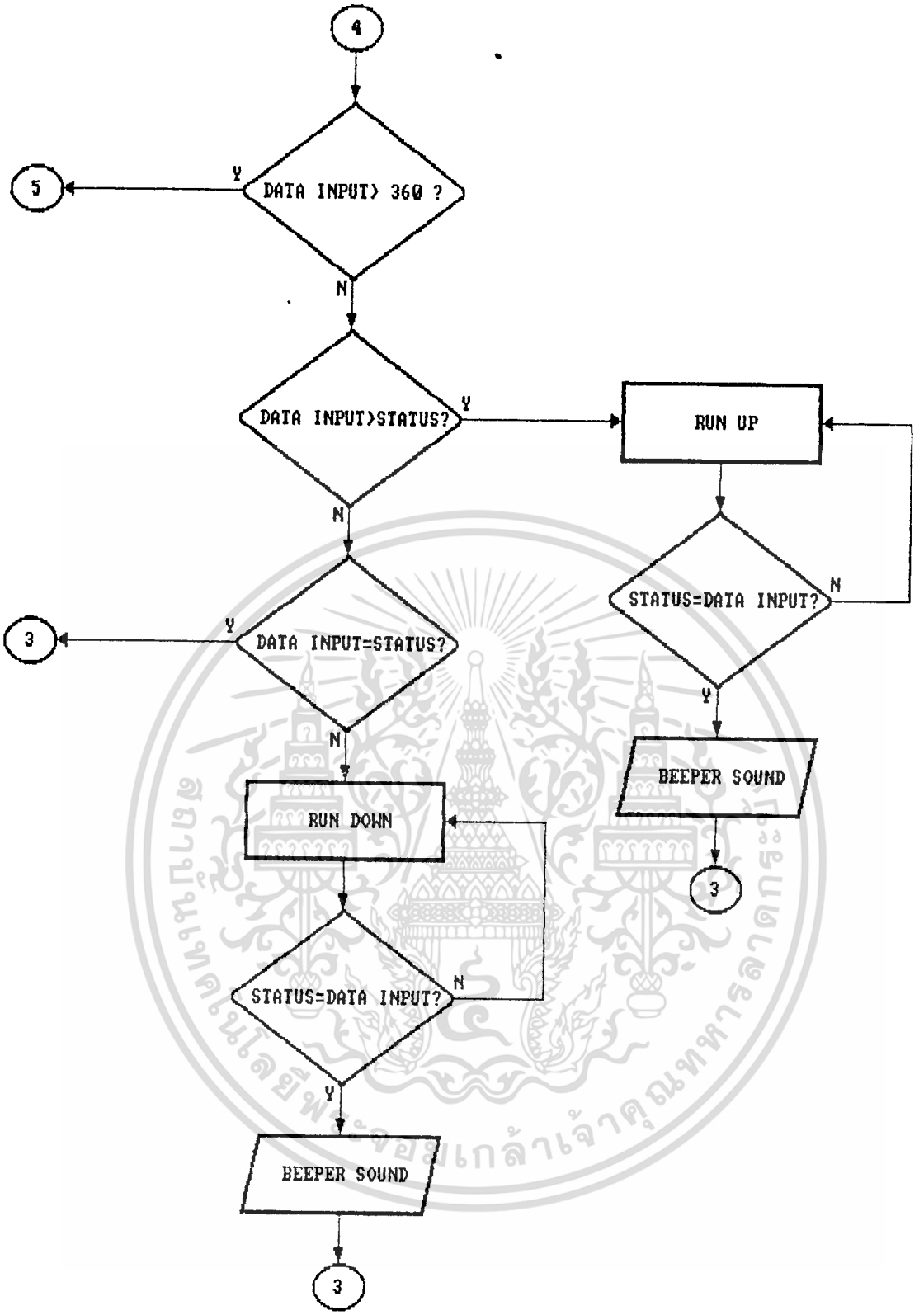
ส่วนโหมด AUTO ถ้ากดคีย์ A/M เครื่องจะแสดงผล AUTO ให้เพื่อรู้ว่าขณะนี้เครื่องเข้าโหมด AUTO ให้แล้วในโหมดนี้จะทำการให้สเตปปิ้งมอเตอร์ หมุนไปที่ตำแหน่งองศาที่เราต้องการ ไม่ว่าจะอยู่ที่ตำแหน่งไหนก็แล้วแต่ โดยการกดคีย์ ENT แล้ว เครื่องจะแสดงผล"----"ให้เราป้อนตัวเลข ตัวเลขที่ป้อนนี้จะป้อนค่าได้ตั้งแต่ 0-360 แต่ถ้าเกินค่าที่กำหนดนี้ เครื่องจะฟ้อง "-Err" ออกมาแต่ถ้ากด 4 ครั้งเครื่องจะแสดงผล "----" เพื่อให้ป้อนค่าใหม่

หลังจากป้อนค่าที่ถูกต้องแล้ว (อยู่ในค่าที่กำหนด) ก็กดคีย์ ENT อีกครั้ง เครื่องจะทำการหมุนไปที่ตำแหน่งองศาที่เราต้องการ แล้วจะมีเสียง Beep เตือนให้เรารู้ว่าถึงตำแหน่งที่เราต้องการแล้ว เมื่อต้องการตำแหน่งใหม่ก็กดคีย์ ENT แล้วป้อนตำแหน่งที่ต้องการ แล้วกดคีย์ ENT เพื่อสั่งทำงาน ถ้าต้องการกลับไปสู่โหมด MANUAL ก็กดคีย์ A/M 1 ครั้ง เครื่องจะแสดงผล "HAnd" เพื่อบอกให้รู้ว่ากำลังเข้าสู่โหมดการใช้แบบ MANUAL

ภาคผนวก







```
          2500 A.D. Z80 Macro Assembler  -  Version 4.02a
          ------------------------------------------------

                Input  Filename : ANTPOS1.ASM
                Output Filename : ANTPOS1.obj


 1                              ;
 2                              ;*****************************************************************************
 3                              ;*                  ANTENNA POSITIONER WITH STEPPING MOTOR                   *
 4                              ;*                              VERSION 1.0                                  *
 5                              ;*                                                                           *
 6                              ;*          By :    Preecha    Surisang     (34131219)                       *
 7                              ;*                  Sittidech  Piyachat     (34131236)                       *
 8                              ;*                     ET 3L   KMIT-L                                        *
 9                              ;*****************************************************************************
10                              ;
11  0000                                ORG     0000H
12          0100                BEGIN   EQU     100H
13          0003                P8255   EQU     03H         ;8255 Control port
14          0002                DIGIT   EQU     02H         ;8255 port C (PC7-PC4)
15          0001                SEG7    EQU     01H         ;8255 port B (PC7-PC0)
16          0002                KEYIN   EQU     02H         ;8255 port C (PC3-PC0)
17          0000                STOUT   EQU     00H         ;8255 port A (PA3-PA0)
18          0088                FBEEP   EQU     88H         ;Beep frequancy
19          005E                TBEEP   EQU     5EH         ;Time duration of beep
20          0005                N       EQU      5          ;Accel count
21          4700                STK     EQU   4700H        ;System stack address
22                              ;
23                              ;
24                              ;
25                              ;
26                              ;
27  0000    C3 00 01                    JP      BEGIN
28  0100                        BEGIN   ORG     BEGIN
29  0100    06 00                       LD      B,0
30  0102    10 FE                       DJNZ    $           ;Power-up delay
31  0104    3E 81                       LD      A,10000001B ;Initial 8255 to mode 0
32  0106    D3 03                       OUT     (P8255),A   ;port A output,port B output
33                                                          ;port C(lower) input,port C
34                                                          ;(upper) output
35  0108    3E F0                       LD      A,0F0H
36  010A    D3 02                       OUT     (DIGIT),A   ;All out port are cleared to 0
37  010C    21 04 47                    LD      HL,STEPBF   ;Port A is cleared
38  010F    36 10                       LD      (HL),10H
39  0111    7E                          LD      A,(HL)
40  0112    D3 00                       OUT     (STOUT),A
41  0114    31 00 47                    LD      SP,SYSSTK   ;initial system stack
42                              ;
43  0117    21 00 40                    LD      HL,4000H    ;lowest RAM address is tested
44  011A    01 00 08                    LD      BC,800H     ;total RAM address
45  011D    CD 92 04    RAMTEST:        CALL    RAMCHK
46  0120    28 01                       JR      Z,NEXT
47  0122    76                          HALT                ;If error
```

```
 48  0123  ED A1              NEXT:       CPI
 49  0125  EA 1D 01                      JP      PE,RAMTEST
 50                          ;
 51  0128  DD 21 57 06       INI:        LD      IX,BLANK    ;initial pattern
 52  012C  0E 05                         LD      C,5         ;pattern count
 53  012E  06 20             INI1:       LD      B,20H       ;display 0.16 sec
 54  0130  CD E6 04          INI2:       CALL    SCAN2
 55  0133  10 FB                         DJNZ    INI2
 56  0135  DD 2B                         DEC     IX          ;next pattern
 57  0137  0D                            DEC     C
 58  0138  20 F4                         JR      NZ,INI1
 59  013A  DD 21 53 06                   LD      IX,ANTP
 60  013E  06 EE                         LD      B,0EEH
 61  0140  CD E6 04          INI3:       CALL    SCAN2
 62  0143  10 FB                         DJNZ    INI3
 63  0145  CD 73 04                      CALL    BEEP
 64                          ;Find home sw position
 65  0148  16 11             HOME:       LD      D,11H
 66  014A  0E 00                         LD      C,0
 67  014C  21 DA 02                      LD      HL,730
 68  014F  CD B8 05          HLOOP:      CALL    STDRIVE
 69  0152  07                            RLCA
 70  0153  57                            LD      D,A
 71  0154  0C                            INC     C
 72  0155  2B                            DEC     HL
 73  0156  7C                            LD      A,H
 74  0157  B5                            OR      L
 75  0158  28 48                         JR      Z,HOMEER
 76  015A  E5                            PUSH    HL
 77  015B  21 87 06                      LD      HL,ACCEL-1
 78  015E  79                            LD      A,C
 79  015F  D6 05                         SUB     N
 80  0161  30 14                         JR      NC,HSLEW    ;If C>N jump to HSLEW
 81  0163  06 00                         LD      B,0
 82  0165  09                            ADD     HL,BC
 83  0166  7E                            LD      A,(HL)
 84  0167  DD 21 71 06       HLOOP1:     LD      IX,INIT
 85  016B  CD B1 05                      CALL    DELAY
 86  016E  CD C3 05                      CALL    HCHK
 87  0171  28 0A                         JR      Z,FINDH1
 88  0173  E1                            POP     HL
 89  0174  C3 4F 01                      JP      HLOOP
 90  0177  0E 05             HSLEW:      LD      C,N
 91  0179  09                            ADD     HL,BC
 92  017A  7E                            LD      A,(HL)
 93  017B  18 EA                         JR      HLOOP1
 94  017D  06 0B             FINDH1:     LD      B,11
 95  017F  CD B8 05          FINDLP1:    CALL    STDRIVE
 96  0182  0F                            RRCA
 97  0183  57                            LD      D,A
 98  0184  3E 12                         LD      A,18
 99  0186  DD 21 71 06                   LD      IX,INIT
100  018A  CD B1 05                      CALL    DELAY
101  018D  10 F0                         DJNZ    FINDLP1
102  018F  06 0F             FINDH2:     LD      B,15
103  0191  CD B8 05          FINDLP2:    CALL    STDRIVE     ;Find home sw sensor
104  0194  07                            RLCA
```

```
105  0195  57                          LD      D,A
106  0196  3E 19                       LD      A,25
107  0198  DD 21 71 06                 LD      IX,INIT
108  019C  CD B1 05                    CALL    DELAY
109  019F  05                          DEC     B
110  01A0  20 0D                       JR      NZ,FINDLP3
111  01A2  CD 73 04          HOMEER:   CALL    BEEP
112  01A5  3E FF                       LD      A,0FFH
113  01A7  DD 21 84 06                 LD      IX,HERROR
114  01AB  CD B1 05                    CALL    DELAY
115  01AE  76                          HALT
116  01AF  CD C3 05          FINDLP3:  CALL    HCHK
117  01B2  20 DD                       JR      NZ,FINDLP2
118  01B4  CD 73 04                    CALL    BEEP
119  01B7  21 05 47          ZEROSET:  LD      HL,DTSAVE
120  01BA  36 00                       LD      (HL),0
121  01BC  23                          INC     HL
122  01BD  36 00                       LD      (HL),0
123                          ;Manual Mode routine
124  01BF  3E C8             MANUAL:   LD      A,200       ;Display 'HAnd'
125  01C1  DD 21 75 06                 LD      IX,HAND
126  01C5  CD B1 05                    CALL    DELAY
127  01C8  31 00 47          MSTART:   LD      SP,SYSSTK   ;Initial system stack
128  01CB  CD A2 05                    CALL    STATUSDP    ;Display degrees status
129  01CE  CD 9A 04          MLOOP:    CALL    SCAN
130  01D1  FE 05                       CP      5
131  01D3  CA E3 01                    JP      Z,UP        ;If key5 is pressed jump to UP
132  01D6  FE 08                       CP      8
133  01D8  CA 9C 02                    JP      Z,DN        ;If key8 is pressed jump to DN
134  01DB  FE 0A                       CP      10
135  01DD  CA 4A 03                    JP      Z,AUTOM     ;If key10 is pressed jump to AUTO Mode
136  01E0  C3 CE 01                    JP      MLOOP
137  01E3  AF                UP:       XOR     A
138  01E4  ED 5B 05 47                 LD      DE,(DTSAVE)
139  01E8  21 68 01                    LD      HL,360
140  01EB  ED 52                       SBC     HL,DE
141  01ED  20 06                       JR      NZ,UP1
142  01EF  CD C8 05                    CALL    ERRORM
143  01F2  C3 C8 01                    JP      MSTART
144  01F5  06 03             UP1:      LD      B,3
145  01F7  DD 21 87 06                 LD      IX,ACCEL-1
146  01FB  21 04 47                    LD      HL,STEPBF
147  01FE  56                          LD      D,(HL)
148  01FF  CD B8 05          UPLOOP:   CALL    STDRIVE
149  0202  0F                          RRCA
150  0203  57                          LD      D,A
151  0204  DD 23                       INC     IX
152  0206  DD 7E 00                    LD      A,(IX)
153  0209  DD E5                       PUSH    IX
154  020B  DD 21 00 47                 LD      IX,DISPBF
155  020F  CD B1 05                    CALL    DELAY
156  0212  DD E1                       POP     IX
157  0214  10 E9                       DJNZ    UPLOOP
158  0216  2A 05 47                    LD      HL,(DTSAVE)
159  0219  23                          INC     HL
160  021A  22 05 47                    LD      (DTSAVE),HL
161  021D  CD A2 05          UPREP1:   CALL    STATUSDP
```

```
162  0220  CD AF 04              CALL    SCAN1
163  0223  FE 04                 CP      4
164  0225  20 A1                 JR      NZ,MSTART
165  0227  ED 5B 05 47           LD      DE,(DTSAVE)
166  022B  AF                    XOR     A
167  022C  21 68 01              LD      HL,360
168  022F  ED 52                 SBC     HL,DE
169  0231  20 06                 JR      NZ,UP2
170  0233  CD C8 05              CALL    ERRORM
171  0236  C3 C8 01              JP      MSTART
172  0239  06 02        UP2:     LD      B,2
173  023B  DD 21 89 06           LD      IX,ACCEL1-1
174  023F  3A 04 47              LD      A,(STEPBF)
175  0242  0F                    RRCA
176  0243  57                    LD      D,A
177  0244  CD B8 05     UPACCEL: CALL    STDRIVE
178  0247  0F                    RRCA
179  0248  57                    LD      D,A
180  0249  DD 23                 INC     IX
181  024B  DD 7E 00              LD      A,(IX)
182  024E  DD E5                 PUSH    IX
183  0250  DD 21 00 47           LD      IX,DISPBF
184  0254  CD B1 05              CALL    DELAY
185  0257  DD E1                 POP     IX
186  0259  10 E9                 DJNZ    UPACCEL
187  025B  2A 05 47              LD      HL,(DTSAVE)
188  025E  23                    INC     HL
189  025F  22 05 47              LD      (DTSAVE),HL
190  0262  CD A2 05     UPREP2:  CALL    STATUSDP
191  0265  CD AF 04              CALL    SCAN1
192  0268  FE 04                 CP      4
193  026A  C2 C8 01              JP      NZ,MSTART
194  026D  AF                    XOR     A
195  026E  ED 5B 05 47           LD      DE,(DTSAVE)
196  0272  21 68 01              LD      HL,360
197  0275  ED 52                 SBC     HL,DE
198  0277  20 06                 JR      NZ,UP3
199  0279  CD C8 05              CALL    ERRORM
200  027C  C3 C8 01              JP      MSTART
201  027F  06 02        UP3:     LD      B,2
202  0281  3A 04 47              LD      A,(STEPBF)
203  0284  0F                    RRCA
204  0285  57                    LD      D,A
205  0286  CD B8 05     UPSLEW:  CALL    STDRIVE
206  0289  0F                    RRCA
207  028A  57                    LD      D,A
208  028B  3E 0B                 LD      A,11
209  028D  CD B1 05              CALL    DELAY
210  0290  10 F4                 DJNZ    UPSLEW
211  0292  2A 05 47              LD      HL,(DTSAVE)
212  0295  23                    INC     HL
213  0296  22 05 47              LD      (DTSAVE),HL
214  0299  C3 62 02              JP      UPREP2
215  029C  ED 5B 05 47  DN:      LD      DE,(DTSAVE)
216  02A0  7A                    LD      A,D
217  02A1  B3                    OR      E
218  02A2  20 06                 JR      NZ,DN1
```

```
219  02A4  CD C8 05              CALL    ERRORM
220  02A7  C3 C8 01              JP      MSTART
221  02AA  06 03          DN1:   LD      B,3
222  02AC  DD 21 87 06           LD      IX,ACCEL-1
223  02B0  21 04 47              LD      HL,STEPBF
224  02B3  56                    LD      D,(HL)
225  02B4  CD B8 05       DNLOOP: CALL   STDRIVE
226  02B7  07                    RLCA
227  02B8  57                    LD      D,A
228  02B9  DD 23                 INC     IX
229  02BB  DD 7E 00              LD      A,(IX)
230  02BE  DD E5                 PUSH    IX
231  02C0  DD 21 00 47           LD      IX,DISPBF
232  02C4  CD B1 05              CALL    DELAY
233  02C7  DD E1                 POP     IX
234  02C9  10 E9                 DJNZ    DNLOOP
235  02CB  2A 05 47              LD      HL,(DTSAVE)
236  02CE  2B                    DEC     HL
237  02CF  22 05 47              LD      (DTSAVE),HL
238  02D2  CD A2 05       DNREP1: CALL   STATUSDP
239  02D5  CD AF 04              CALL    SCAN1
240  02D8  FE 07                 CP      7
241  02DA  C2 C8 01              JP      NZ,MSTART
242  02DD  ED 5B 05 47           LD      DE,(DTSAVE)
243  02E1  7A                    LD      A,D
244  02E2  B3                    OR      E
245  02E3  20 06                 JR      NZ,DN2
246  02E5  CD C8 05              CALL    ERRORM
247  02E8  C3 C8 01              JP      MSTART
248  02EB  06 02          DN2:   LD      B,2
249  02ED  DD 21 89 06           LD      IX,ACCEL1-1
250  02F1  3A 04 47              LD      A,(STEPBF)
251  02F4  07                    RLCA
252  02F5  57                    LD      D,A
253  02F6  CD B8 05       DNACCEL: CALL  STDRIVE
254  02F9  07                    RLCA
255  02FA  57                    LD      D,A
256  02FB  DD 23                 INC     IX
257  02FD  DD 7E 00              LD      A,(IX)
258  0300  DD E5                 PUSH    IX
259  0302  DD 21 00 47           LD      IX,DISPBF
260  0306  CD B1 05              CALL    DELAY
261  0309  DD E1                 POP     IX
262  030B  10 E9                 DJNZ    DNACCEL
263  030D  2A 05 47              LD      HL,(DTSAVE)
264  0310  2B                    DEC     HL
265  0311  22 05 47              LD      (DTSAVE),HL
266  0314  CD A2 05       DNREP2: CALL   STATUSDP
267  0317  CD AF 04              CALL    SCAN1
268  031A  FE 07                 CP      7
269  031C  C2 C8 01              JP      NZ,MSTART
270  031F  ED 5B 05 47           LD      DE,(DTSAVE)
271  0323  7A                    LD      A,D
272  0324  B3                    OR      E
273  0325  20 06                 JR      NZ,DN3
274  0327  CD C8 05              CALL    ERRORM
275  032A  C3 C8 01              JP      MSTART
```

```
276  032D  06 02          DN3:      LD      B,2
277  032F  3A 04 47                 LD      A,(STEPBF)
278  0332  07                       RLCA
279  0333  57                       LD      D,A
280  0334  CD B8 05        DNSLEW:   CALL    STDRIVE
281  0337  07                       RLCA
282  0338  57                       LD      D,A
283  0339  3E 0B                    LD      A,11
284  033B  CD B1 05                 CALL    DELAY
285  033E  10 F4                    DJNZ    DNSLEW
286  0340  2A 05 47                 LD      HL,(DTSAVE)
287  0343  2B                       DEC     HL
288  0344  22 05 47                 LD      (DTSAVE),HL
289  0347  C3 14 03                 JP      DNREP2
290                                 ;
291                                 ;Auto mode routine
292  034A  3E C8          AUTOM:    LD      A,200
293  034C  DD 21 79 06              LD      IX,AUTO
294  0350  CD B1 05                 CALL    DELAY
295  0353  31 00 47       ASTART:   LD      SP,SYSSTK   ;Initial system stack
296  0356  CD A2 05                 CALL    STATUSDP
297  0359  CD 9A 04       ALOOP:    CALL    SCAN
298  035C  FE 0A                    CP      10
299  035E  CA BF 01                 JP      Z,MANUAL
300  0361  FE 0B                    CP      11
301  0363  28 03                    JR      Z,CLRBF
302  0365  C3 53 03                 JP      ASTART
303  0368  06 04          CLRBF:    LD      B,4
304  036A  DD 21 09 47              LD      IX,INPUTBF
305  036E  DD 36 00 00     CLRLP:    LD      (IX),0
306  0372  DD 23                    INC     IX
307  0374  10 F8                    DJNZ    CLRLP
308  0376  DD 21 80 06              LD      IX,GETDT
309  037A  CD 9A 04                 CALL    SCAN
310  037D  FE 0A                    CP      10
311  037F  CA BF 01                 JP      Z,MANUAL
312  0382  FE 0B                    CP      11
313  0384  28 57                    JR      Z,ARUN
314  0386  0E 00                    LD      C,0
315  0388  06 02                    LD      B,2
316  038A  21 09 47                 LD      HL,INPUTBF
317  038D  ED 6F           SAVEIP:   RLD
318  038F  E5              ADP:      PUSH    HL
319  0390  C5                       PUSH    BC
320  0391  CD 62 05                 CALL    DTOH
321  0394  AF                       XOR     A
322  0395  21 68 01                 LD      HL,360
323  0398  ED 5B 0B 47              LD      DE,(TEMP1)
324  039C  ED 52                    SBC     HL,DE
325  039E  38 34                    JR      C,GETER
326  03A0  ED 5B 09 47              LD      DE,(INPUTBF)
327  03A4  CD 0C 05                 CALL    DATADP
328  03A7  3E 40                    LD      A,40H
329  03A9  32 03 47                 LD      (DISPBF+3),A
330  03AC  DD 21 00 47              LD      IX,DISPBF
331  03B0  CD 9A 04                 CALL    SCAN
332  03B3  C1                       POP     BC
```

```
333  03B4  E1                        POP     HL
334  03B5  FE 0A                     CP      10
335  03B7  CA BF 01                  JP      Z,MANUAL
336  03BA  FE 0B                     CP      11
337  03BC  28 1F                     JR      Z,ARUN
338  03BE  0C                        INC     C
339  03BF  F5                        PUSH    AF
340  03C0  79                        LD      A,C
341  03C1  FE 03                     CP      3
342  03C3  20 05                     JR      NZ,GETNEXT
343  03C5  31 00 47                  LD      SP,SYSSTK   ;Initial system stack
344  03C8  18 9E                     JR      CLRBF
345  03CA  F1              GETNEXT:  POP     AF
346  03CB  10 C0                     DJNZ    SAVEIP
347  03CD  ED 6F                     RLD
348  03CF  23                        INC     HL
349  03D0  77                        LD      (HL),A
350  03D1  C3 8F 03                  JP      ADP
351  03D4  CD C8 05        GETER:    CALL    ERRORM
352  03D7  31 00 47                  LD      SP,SYSSTK   ;Initial system stack
353  03DA  C3 68 03                  JP      CLRBF
354  03DD  ED 5B 0B 47     ARUN:     LD      DE,(TEMP1)
355  03E1  2A 05 47                  LD      HL,(DTSAVE)
356  03E4  7C                        LD      A,H
357  03E5  B5                        OR      L
358  03E6  20 09                     JR      NZ,MAXCHK
359  03E8  AF                        XOR     A
360  03E9  21 68 01                  LD      HL,360
361  03EC  ED 52                     SBC     HL,DE
362  03EE  CA 53 03                  JP      Z,ASTART
363  03F1  AF              MAXCHK:   XOR     A
364  03F2  21 68 01                  LD      HL,360
365  03F5  ED 5B 05 47               LD      DE,(DTSAVE)
366  03F9  ED 52                     SBC     HL,DE
367  03FB  20 08                     JR      NZ,EQCHK
368  03FD  2A 0B 47                  LD      HL,(TEMP1)
369  0400  7C                        LD      A,H
370  0401  B5                        OR      L
371  0402  CA 53 03                  JP      Z,ASTART
372  0405  AF              EQCHK:    XOR     A
373  0406  2A 05 47                  LD      HL,(DTSAVE)
374  0409  ED 5B 0B 47               LD      DE,(TEMP1)
375  040D  ED 52                     SBC     HL,DE
376  040F  CA 53 03                  JP      Z,ASTART
377  0412  38 2A                     JR      C,AUTOUP
378  0414  29                        ADD     HL,HL
379  0415  01 00 00                  LD      BC,0
380  0418  1E 02                     LD      E,2
381  041A  3A 04 47                  LD      A,(STEPBF)
382  041D  07                        RLCA
383  041E  57              ASTDN:    LD      D,A
384  041F  CD B8 05                  CALL    STDRIVE
385  0422  07                        RLCA
386  0423  F5                        PUSH    AF
387  0424  DD 21 87 06               LD      IX,ACCEL-1
388  0428  03                        INC     BC
389  0429  2B                        DEC     HL
```

```
390  042A  1D                        DEC     E
391  042B  CC D5 05                  CALL    Z,DECDT
392  042E  7C                        LD      A,H
393  042F  B5                        OR      L
394  0430  20 06                     JR      NZ,DDETERM
395  0432  CD 73 04                  CALL    BEEP
396  0435  C3 53 03                  JP      ASTART
397  0438  CD FB 05          DDETERM: CALL   DETERM
398  043B  F1                        POP     AF
399  043C  18 E0                     JR      ASTDN
400  043E  ED 5B 0B 47       AUTOUP:  LD     DE,(TEMP1)
401  0442  2A 05 47                  LD      HL,(DTSAVE)
402  0445  EB                        EX      DE,HL
403  0446  AF                        XOR     A
404  0447  ED 52                     SBC     HL,DE
405  0449  29                        ADD     HL,HL
406  044A  01 00 00                  LD      BC,0
407  044D  1E 02                     LD      E,2
408  044F  3A 04 47                  LD      A,(STEPBF)
409  0452  0F                        RRCA
410  0453  57                ASTUP:  LD      D,A
411  0454  CD B8 05                  CALL    STDRIVE
412  0457  0F                        RRCA
413  0458  F5                        PUSH    AF
414  0459  DD 21 87 06               LD      IX,ACCEL-1
415  045D  03                        INC     BC
416  045E  2B                        DEC     HL
417  045F  1D                        DEC     E
418  0460  CC E8 05                  CALL    Z,INCDT
419  0463  7C                        LD      A,H
420  0464  B5                        OR      L
421  0465  20 06                     JR      NZ,UDETERM
422  0467  CD 73 04                  CALL    BEEP
423  046A  C3 53 03                  JP      ASTART
424  046D  CD FB 05          UDETERM: CALL   DETERM
425  0470  F1                        POP     AF
426  0471  18 E0                     JR      ASTUP
427                        ;
428                        ;
429                        ;
430                        ;
431                        ;
432                        ;*****************************************************************************
433                        ;*                                                                           *
434                        ;*                          UTILITY  SUBROUTINE                              *
435                        ;*                                                                           *
436                        ;*****************************************************************************
437                        ;
438                        ;Generate beep sound
439  0473  21 5E 00          BEEP:   LD      HL,TBEEP
440  0476  29                        ADD     HL,HL
441  0477  11 01 00                  LD      DE,1
442  047A  3E 00                     LD      A,0
443  047C  F5                SQWAVE: PUSH    AF
444  047D  3A 04 47                  LD      A,(STEPBF)
445  0480  4F                        LD      C,A
446  0481  CB A1                     RES     4,C
```

```
447  0483  F1                        POP     AF
448  0484  B1                        OR      C
449  0485  D3 00                     OUT     (STOUT),A
450  0487  06 88                     LD      B,FBEEP
451  0489  10 FE                     DJNZ    $
452  048B  EE 10                     XOR     10H
453  048D  ED 52                     SBC     HL,DE
454  048F  20 EB                     JR      NZ,SQWAVE
455  0491  C9                        RET
456                          ;
457                          ;Check if a memory adress is in RAM
458  0492  7E                RAMCHK: LD      A,(HL)
459  0493  2F                        CPL                     ;invert in A regiter
460  0494  77                        LD      (HL),A
461  0495  7E                        LD      A,(HL)
462  0496  2F                        CPL
463  0497  77                        LD      (HL),A
464  0498  BE                        CP      (HL)
465  0499  C9                        RET
466                          ;
467                          ;Scan keyboard and display loop until a key is pressed
468  049A  06 08             SCAN:   LD      B,8
469  049C  CD AF 04          SCANX:  CALL    SCAN1
470  049F  30 F9                     JR      NC,SCAN
471  04A1  10 F9                     DJNZ    SCANX
472  04A3  CD AF 04          SCLOOP: CALL    SCAN1
473  04A6  38 FB                     JR      C,SCLOOP
474  04A8  21 5B 06          KEYMAP: LD      HL,KEYTAB
475  04AB  85                        ADD     A,L
476  04AC  6F                        LD      L,A
477  04AD  7E                        LD      A,(HL)
478  04AE  C9                        RET
479                          ;
480                          ;Scan keyboard and display one cycle
481  04AF  37                SCAN1:  SCF
482  04B0  08                        EX      AF,AF'
483  04B1  D9                        EXX
484  04B2  0E 00                     LD      C,0
485  04B4  1E 7F                     LD      E,01111111B
486  04B6  26 04                     LD      H,4
487  04B8  7B                KCOL:   LD      A,E
488  04B9  D3 02                     OUT     (DIGIT),A
489  04BB  DD 7E 00                  LD      A,(IX)
490  04BE  D3 01                     OUT     (SEG7),A
491  04C0  06 C8                     LD      B,0C8H          ;Column delay time
492  04C2  10 FE                     DJNZ    $
493  04C4  06 C8                     LD      B,0C8H          ;Column delay time
494  04C6  10 FE                     DJNZ    $
495  04C8  06 03                     LD      B,3
496  04CA  DB 02                     IN      A,(KEYIN)
497  04CC  57                        LD      D,A
498  04CD  CB 1A             KROW:   RR      D
499  04CF  38 02                     JR      C,NOKEY
500  04D1  79                        LD      A,C
501  04D2  08                        EX      AF,AF'
502  04D3  0C                NOKEY:  INC     C
503  04D4  10 F7                     DJNZ    KROW
```

```
504  04D6  DD 23                    INC     IX
505  04D8  7B                       LD      A,E
506  04D9  0F                       RRCA
507  04DA  5F                       LD      E,A
508  04DB  25                       DEC     H
509  04DC  20 DA                    JR      NZ,KCOL
510  04DE  11 FC FF                 LD      DE,-4
511  04E1  DD 19                    ADD     IX,DE
512  04E3  D9                       EXX
513  04E4  08                       EX      AF,AF'
514  04E5  C9                       RET
515                         ;
516                         ;Scan display one cycle.
517  04E6  08               SCAN2:  EX      AF,AF'
518  04E7  D9                       EXX
519  04E8  1E 7F                    LD      E,01111111B
520  04EA  26 04                    LD      H,4
521  04EC  7B               DLOOP:  LD      A,E
522  04ED  D3 02                    OUT     (DIGIT),A
523  04EF  DD 7E 00                 LD      A,(IX)
524  04F2  D3 01                    OUT     (SEG7),A
525  04F4  06 90                    LD      B,90H
526  04F6  10 FE                    DJNZ    $
527  04F8  06 90                    LD      B,90H
528  04FA  10 FE                    DJNZ    $
529  04FC  DD 23                    INC     IX
530  04FE  7B                       LD      A,E
531  04FF  0F                       RRCA
532  0500  5F                       LD      E,A
533  0501  25                       DEC     H
534  0502  20 E8                    JR      NZ,DLOOP
535  0504  11 FC FF                 LD      DE,-4
536  0507  DD 19                    ADD     IX,DE
537  0509  D9                       EXX
538  050A  08                       EX      AF,AF'
539  050B  C9                       RET
540                         ;
541                         ;Convert the 2 byte data stored in DE to 7-segment
542  050C  21 00 47         DATADP: LD      HL,DISPBF
543  050F  7B                       LD      A,E
544  0510  CD 1B 05                 CALL    HEX7SG
545  0513  7A                       LD      A,D
546  0514  CD 1B 05                 CALL    HEX7SG
547  0517  2B                       DEC     HL
548  0518  36 00                    LD      (HL),0
549  051A  C9                       RET
550                         ;
551                         ;Convert binary data to 7-segment display format.
552  051B  F5               HEX7SG: PUSH    AF
553  051C  CD 2C 05                 CALL    HEX7
554  051F  77                       LD      (HL),A
555  0520  23                       INC     HL
556  0521  F1                       POP     AF
557  0522  0F                       RRCA
558  0523  0F                       RRCA
559  0524  0F                       RRCA
560  0525  0F                       RRCA
```

```
561  0526  CD 2C 05                    CALL    HEX7
562  0529  77                          LD      (HL),A
563  052A  23                          INC     HL
564  052B  C9                          RET
565                              ;
566                              ;Convert binary data to 7-aegment display format.
567  052C  E5                    HEX7: PUSH    HL
568  052D  21 67 06                    LD      HL,SEGTAB
569  0530  E6 0F                       AND     0FH
570  0532  85                          ADD     A,L
571  0533  6F                          LD      L,A
572  0534  7E                          LD      A,(HL)
573  0535  E1                          POP     HL
574  0536  C9                          RET
575                              ;Convert hex code 2 byte in DTSAVE memory to dec code stored in STATUS memory
576  0537  AF                    HTOD: XOR     A
577  0538  DD 21 07 47                 LD      IX,STATUS
578  053C  2A 05 47                    LD      HL,(DTSAVE)
579  053F  11 64 00                    LD      DE,100
580  0542  06 FF                       LD      B,0FFH      ; B =(-1)
581  0544  04                    D100LP: INC     B
582  0545  ED 52                       SBC     HL,DE       ;Sub DTSAVE with 100
583  0547  30 FB                       JR      NC,D100LP
584  0549  19                          ADD     HL,DE
585  054A  DD 70 01                    LD      (IX+1),B    ;Save D100 in STATUS
586  054D  7D                          LD      A,L
587  054E  06 FF                       LD      B,0FFH      ; B =(-1)
588  0550  04                    D10LP: INC     B
589  0551  D6 0A                       SUB     10
590  0553  30 FB                       JR      NC,D10LP
591  0555  C6 0A                       ADD     A,10
592  0557  4F                          LD      C,A
593  0558  78                          LD      A,B
594  0559  07                          RLCA
595  055A  07                          RLCA
596  055B  07                          RLCA
597  055C  07                          RLCA
598  055D  B1                          OR      C
599  055E  DD 77 00                    LD      (IX),A
600  0561  C9                          RET
601                              ;Convert BCD in INPUTBF to binary in TEMP1
602  0562  E5                    DTOH: PUSH    HL
603  0563  26 00                       LD      H,0
604  0565  3A 0A 47                    LD      A,(INPUTBF+1)
605  0568  6F                          LD      L,A
606  0569  29                          ADD     HL,HL
607  056A  29                          ADD     HL,HL
608  056B  22 0B 47                    LD      (TEMP1),HL
609  056E  29                          ADD     HL,HL
610  056F  29                          ADD     HL,HL
611  0570  29                          ADD     HL,HL
612  0571  22 0D 47                    LD      (TEMP2),HL
613  0574  29                          ADD     HL,HL
614  0575  ED 5B 0B 47                 LD      DE,(TEMP1)
615  0579  19                          ADD     HL,DE
616  057A  ED 5B 0D 47                 LD      DE,(TEMP2)
617  057E  19                          ADD     HL,DE
```

```
618  057F  22 0D 47                LD      (TEMP2),HL
619  0582  21 09 47                LD      HL,INPUTBF
620  0585  7E                      LD      A,(HL)
621  0586  E6 F0                   AND     0F0H
622  0588  0F                      RRCA
623  0589  0F                      RRCA
624  058A  0F                      RRCA
625  058B  0F                      RRCA
626  058C  87                      ADD     A,A
627  058D  47                      LD      B,A
628  058E  87                      ADD     A,A
629  058F  87                      ADD     A,A
630  0590  80                      ADD     A,B
631  0591  47                      LD      B,A
632  0592  7E                      LD      A,(HL)
633  0593  E6 0F                   AND     0FH
634  0595  80                      ADD     A,B
635  0596  4F                      LD      C,A
636  0597  06 00                   LD      B,0
637  0599  2A 0D 47                LD      HL,(TEMP2)
638  059C  09                      ADD     HL,BC
639  059D  22 0B 47                LD      (TEMP1),HL
640  05A0  E1                      POP     HL
641  05A1  C9                      RET
642                        ;Degrees status are dispaied.
643  05A2  CD 37 05       STATUSDP: CALL    HTOD
644  05A5  ED 5B 07 47             LD      DE,(STATUS)
645  05A9  CD 0C 05                CALL    DATADP
646  05AC  DD 21 00 47             LD      IX,DISPBF
647  05B0  C9                      RET
648                        ;Delay time used by HL register
649  05B1  3D             DELAY:   DEC     A
650  05B2  CD E6 04                CALL    SCAN2
651  05B5  20 FA                   JR      NZ,DELAY
652  05B7  C9                      RET
653                        ;Out step to driver
654  05B8  7A             STDRIVE: LD      A,D
655  05B9  32 04 47                LD      (STEPBF),A
656  05BC  F5                      PUSH    AF
657  05BD  CB E7                   SET     4,A
658  05BF  D3 00                   OUT     (STOUT),A
659  05C1  F1                      POP     AF
660  05C2  C9                      RET
661                        ;Check home sw input port
662  05C3  DB 02          HCHK:    IN      A,(KEYIN)
663  05C5  E6 08                   AND     8
664  05C7  C9                      RET
665                        ;Error massage display
666  05C8  CD 73 04       ERRORM:  CALL    BEEP
667  05CB  3E 96                   LD      A,150
668  05CD  DD 21 7D 06             LD      IX,ERROR
669  05D1  CD B1 05                CALL    DELAY
670  05D4  C9                      RET
671                        ;Decrement data in DTSAVE memory.
672  05D5  D9             DECDT:   EXX
673  05D6  DD E5                   PUSH    IX
674  05D8  2A 05 47                LD      HL,(DTSAVE)
```

```
675  05DB  2B                        DEC     HL
676  05DC  22 05 47                  LD      (DTSAVE),HL
677  05DF  CD A2 05                  CALL    STATUSDP
678  05E2  DD E1                     POP     IX
679  05E4  D9                        EXX
680  05E5  1E 02                     LD      E,2
681  05E7  C9                        RET
682                        ;Increment data in DTSAVE memory.
683  05E8  D9                INCDT:  EXX
684  05E9  DD E5                     PUSH    IX
685  05EB  2A 05 47                  LD      HL,(DTSAVE)
686  05EE  23                        INC     HL
687  05EF  22 05 47                  LD      (DTSAVE),HL
688  05F2  CD A2 05                  CALL    STATUSDP
689  05F5  DD E1                     POP     IX
690  05F7  D9                        EXX
691  05F8  1E 02                     LD      E,2
692  05FA  C9                        RET
693                        ;Determine outstep delay time for ACCEL,SLEW,DECEL.
694  05FB  D5                DETERM: PUSH    DE
695  05FC  E5                        PUSH    HL
696  05FD  AF                        XOR     A
697  05FE  ED 42                     SBC     HL,BC
698  0600  38 1B                     JR      C,HLSUBN
699  0602  AF                        XOR     A
700  0603  21 05 00                  LD      HL,N
701  0606  ED 42                     SBC     HL,BC
702  0608  38 33                     JR      C,ASLEW
703  060A  DD E5             AACEL:  PUSH    IX
704  060C  DD 09                     ADD     IX,BC
705  060E  DD 7E 00                  LD      A,(IX)
706  0611  DD 21 00 47               LD      IX,DISPBF
707  0615  CD B1 05                  CALL    DELAY
708  0618  DD E1                     POP     IX
709  061A  E1                        POP     HL
710  061B  D1                        POP     DE
711  061C  C9                        RET
712  061D  E1                HLSUBN: POP     HL
713  061E  E5                        PUSH    HL
714  061F  AF                        XOR     A
715  0620  11 05 00                  LD      DE,N
716  0623  ED 52                     SBC     HL,DE
717  0625  30 16                     JR      NC,ASLEW
718  0627  E1                ADECEL: POP     HL
719  0628  E5                        PUSH    HL
720  0629  EB                        EX      DE,HL
721  062A  DD E5                     PUSH    IX
722  062C  DD 19                     ADD     IX,DE
723  062E  DD 7E 00                  LD      A,(IX)
724  0631  DD 21 00 47               LD      IX,DISPBF
725  0635  CD B1 05                  CALL    DELAY
726  0638  DD E1                     POP     IX
727  063A  E1                        POP     HL
728  063B  D1                        POP     DE
729  063C  C9                        RET
730  063D  11 05 00          ASLEW:  LD      DE,N
731  0640  DD E5                     PUSH    IX
```

```
0642  DD 19                     ADD     IX,DE
0644  DD 7E 00                  LD      A,(IX)
0647  DD 21 00 47               LD      IX,DISPBF
064B  CD B1 05                  CALL    DELAY
064E  DD E1                     POP     IX
0650  E1                        POP     HL
0651  D1                        POP     DE
0652  C9                        RET
                        ;
                        ;
                        ;
                        ;*************************************************************************
                        ;*                                                                       *
                        ;*                      Define ROM data and table.                       *
                        ;*                                                                       *
                        ;*************************************************************************
                        ;
                        ;
0653  73                ANTP:   DEFB    73H         ;'P'
0654  F8                        DEFB    0F8H        ;'t'
0655  54                        DEFB    54H         ;'n'
0656  77                        DEFB    77H         ;'A'
0657  00                BLANK:  DEFB    0
0658  00                        DEFB    0
0659  00                        DEFB    0
065A  00                        DEFB    0
                        ;
                        ;Key-position-code to key-internal-code conversion table.
065B  01                KEYTAB: DEFB    1           ;C0,R0
065C  02                        DEFB    2           ;C0,R1
065D  03                        DEFB    3           ;C0,R2
065E  04                        DEFB    4           ;C1,R0
065F  05                        DEFB    5           ;C1,R1
0660  06                        DEFB    6           ;C1,R2
0661  07                        DEFB    7           ;C2,R0
0662  08                        DEFB    8           ;C2,R1
0663  09                        DEFB    9           ;C2,R2
0664  0A                        DEFB    10          ;C3,R0
0665  00                        DEFB    0           ;C3,R1
0666  0B                        DEFB    11          ;C3,R2
                        ;
0667  3F                SEGTAB: DEFB    3FH         ;0
0668  06                        DEFB    06H         ;1
0669  5B                        DEFB    5BH         ;2
066A  4F                        DEFB    4FH         ;3
066B  66                        DEFB    66H         ;4
066C  6D                        DEFB    6DH         ;5
066D  7D                        DEFB    7DH         ;6
066E  07                        DEFB    07H         ;7
066F  7F                        DEFB    7FH         ;8
0670  67                        DEFB    67H         ;9
0671  F8                INIT:   DEFB    0F8H        ;'t'
0672  04                        DEFB    04          ;'i'
0673  54                        DEFB    54H         ;'n'
0674  04                        DEFB    04          ;'i'
0675  5E                HAND:   DEFB    5EH         ;'d'
0676  54                        DEFB    54H         ;'n'
```

```
789  0677  77                        DEFB    77H         ;'A'
790  0678  76                        DEFB    76H         ;'H'
791  0679  5C              AUTO:     DEFB    5CH         ;'o'
792  067A  78                        DEFB    78H         ;'t'
793  067B  1C                        DEFB    1CH         ;'U'
794  067C  77                        DEFB    77H         ;'A'
795  067D  50              ERROR:    DEFB    50H         ;'r'
796  067E  50                        DEFB    50H         ;'r'
797  067F  79                        DEFB    79H         ;'E'
798  0680  40              GETDT:    DEFB    40H         ;'-'
799  0681  40                        DEFB    40H         ;'-'
800  0682  40                        DEFB    40H         ;'-'
801  0683  40                        DEFB    40H         ;'-'
802  0684  50              HERROR:   DEFB    50H         ;'r'
803  0685  50                        DEFB    50H         ;'r'
804  0686  79                        DEFB    79H         ;'E'
805  0687  76                        DEFB    76H         ;'H'
806  0688  0F              ACCEL:    DEFB     15
807  0689  0E                        DEFB     14
808  068A  0D              ACCEL1:   DEFB     13
809  068B  0C                        DEFB     12
810  068C  0B                        DEFB     11
811                         ;
812                         ;
813                         ;
814                         ;RAM system stack area.
815  4700                           ORG     STK
816  4700                   SYSSTK:
817  4700                   DISPBF   DEFS    4
818  4704                   STEPBF   DEFS    1
819  4705                   DTSAVE   DEFS    2
820  4707                   STATUS   DEFS    2
821  4709                   INPUTBF  DEFS    2
822  470B                   TEMP1    DEFS    2
823  470D                   TEMP2    DEFS    2
824  470F                   END
825
```

Defined	Symbol Name		Value	References							
703	AACEL		060A								
806	ACCEL		0688	77	145	222	387	414			
808	ACCEL1		068A	173	249						
718	ADECEL		0627								
318	ADP		038F	350							
297	ALOOP		0359								
750	ANTP		0653	59							
354	ARUN		03DD	313	337						
730	ASLEW		063D	702	717						
295	ASTART		0353	302	362	371	376	396	423		
383	ASTDN		041E	399							
410	ASTUP		0453	426							
791	AUTO		0679	293							
292	AUTOM		034A	135							
400	AUTOUP		043E	377							
439	BEEP		0473	63	111	118	395	422	666		
12	BEGIN	=	0100	27	28						
754	BLANK		0657	51							
303	CLRBF		0368	301	344	353					
305	CLRLP		036E	307							
Pre	CODE		4700	11	28	815					
581	D100LP		0544	583							
588	D10LP		0550	590							
Pre	DATA		0000								
542	DATADP		050C	327	645						
397	DDETERM		0438	394							
672	DECDT		05D5	391							
649	DELAY		05B1	85	100	108	114	126	155	184	209
				294	651	669	707	725	735		
694	DETERM		05FB	397	424						
14	DIGIT	=	0002	36	488	522					
817	DISPBF		4700	154	183	231	259	329	330	542	646
521	DLOOP		04EC	534							
215	DN		029C	133							
221	DN1		02AA	218							
248	DN2		02EB	245							
276	DN3		032D	273							
253	DNACCEL		02F6	262							
225	DNLOOP		02B4	234							
238	DNREP1		02D2								
266	DNREP2		0314	289							
280	DNSLEW		0334	285							
602	DTOH		0562	320							
819	DTSAVE		4705	119	138	158	160	165	187	189	195
				235	237	242	263	265	270	286	288
				401	578	674	676	685	687		
824	END		470F								
372	EQCHK		0405	367							
795	ERROR		067D	668							
666	ERRORM		05C8	142	170	199	219	246	274	351	
18	FBEEP	=	0088	450							
94	FINDH1		017D	87							
102	FINDH2		018F								
95	FINDLP1		017F	101							
103	FINDLP2		0191	117							
116	FINDLP3		01AF	110							
798	GETDT		0680	308							

Defined	Symbol Name		Value	References							
351	GETER		03D4	325							
345	GETNEXT		03CA	342							
787	HAND		0675	125							
662	HCHK		05C3	86	116						
802	HERROR		0684	113							
567	HEX7		052C	553	561						
552	HEX7SG		051B	544	546						
68	HLOOP		014F	89							
84	HLOOP1		0167	93							
712	HLSUBN		061D	698							
65	HOME		0148								
111	HOMEER		01A2	75							
90	HSLEW		0177	80							
576	HTOD		0537	643							
683	INCDT		05E8	418							
51	INI		0128								
53	INI1		012E	58							
54	INI2		0130	55							
61	INI3		0140	62							
783	INIT		0671	84	99	107					
821	INPUTBF		4709	304	316	326	604	619			
487	KCOL		0488	509							
16	KEYIN	=	0002	496	662						
474	KEYMAP		04A8								
760	KEYTAB		065B	474							
498	KROW		04CD	503							
124	MANUAL		01BF	299	311	335					
363	MAXCHK		03F1	358							
129	MLOOP		01CE	136							
127	MSTART		01C8	143	164	171	193	200	220	241	247
20	N	=	0005	79	90	700	715	730			
48	NEXT		0123	46							
502	NOKEY		04D3	499							
13	P8255	=	0003	32							
458	RAMCHK		0492	45							
45	RAMTEST		011D	49							
317	SAVEIP		038D	346							
468	SCAN		049A	129	297	309	331	470			
481	SCAN1		04AF	162	191	239	267	469	472		
517	SCAN2		04E6	54	61	650					
469	SCANX		049C	471							
472	SCLOOP		04A3	473							
15	SEG7	=	0001	490	524						
773	SEGTAB		0667	568							
443	SQWAVE		047C	454							
820	STATUS		4707	577	644						
643	STATUSDP		05A2	128	161	190	238	266	296	677	688
654	STDRIVE		05B8	68	95	103	148	177	205	225	253
818	STEPBF		4704	37	146	174	202	223	250	277	381
21	STK	=	4700	815							
17	STOUT	=	0000	40	449	658					
816	SYSSTK		4700	41	127	295	343	352			
19	TBEEP	=	005E	439							
822	TEMP1		470B	323	354	368	374	400	608	614	639
823	TEMP2		470D	612	616	618	637				
424	UDETERM		046D	421							
137	UP		01E3	131							

Defined	Symbol Name	Value	References
144	UP1	01F5	141
172	UP2	0239	169
201	UP3	027F	198
177	UPACCEL	0244	186
148	UPLOOP	01FF	157
161	UPREP1	021D	
190	UPREP2	0262	214
205	UPSLEW	0286	210
119	ZEROSET	01B7	

Lines Assembled : 825 Assembly Errors : 0

หนังสืออ้างอิง

1. Takashi Kenjo, 1985, Stepping Motor and Their Microprocessor Controls , Clarendon press., Oxford

2. Lance A. Leventhal, 1988, Z80 Assembly Language Programming, Osborne/McGraw-Hill, Berkeley, California

3. วิบูลย์ ชื่นแขก, 2532, ไมโครโปรเซสเซอร์พื้นฐาน, สถาบันเทคโนโลยีพระจอมเกล้าพระนครเหนือ

4. อรรถสิทธ์ หล่าสกุล, ไมโครโปรเซสเซอร์พื้นฐาน, ตำราชุดวิศวกรรมศาสตร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

5. บริษัท อีทีที จำกัด, ET-SMCC STEPPING MOTOR CONTROL