สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

เครื่องวัดมุมกับพารามิเตอร์ต่าง ๆ

ปพ.
ว๖๔๔๓
๒๕๓๖

เลขหมู่.....
เลขทะเบียน.....
วัน,เดือน,ปี.....

6125584๔

นายนาถ ดนตรี

นายสุวรรณ ลิขิตวัฒนานุรักษ์

โครงงานนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิทยาศาสตร์บัณฑิต

ภาควิชาฟิสิกส์ประยุกต์

คณะวิทยาศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2536

GEONIOMETER

Mr.Nart Dontree

Mr.Suwan likitwattananurak

A Special Project Submitted in Partial Fulfillment of the

Requirement for the Degree of Bachelor of Science

Department of Applied Physics

Faculty of Science

King Mongkut's Institute of Technology Ladkrabang

1993

หัวข้อโครงงานพิเศษ	เครื่องวัดมุมกับพารามิเตอร์ต่าง ๆ
โดย	นายนาถ คนตรี
	นายสุวรรณ ลิขิตวัฒนานุรักษ์
ภาควิชา	ฟิสิกส์ประยุกต์
อาจารย์ที่ปรึกษา	ผศ.ดร.อารีย์ วิเชียรฉาย

ภาควิชาฟิสิกส์ประยุกต์ คณะวิทยาศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง อนุมัติให้นับโครงงานพิเศษฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิทยาศาสตร์บัณฑิต

---------- หัวหน้าภาค

(ผศ.ดร.ปรีชา เทียนสมประสงค์)

คณะกรรมการโครงงานพิเศษ

---------- กรรมการ

(ผศ.ดร.อารีย์ วิเชียรฉาย)

---------- กรรมการ

(ผศ.ดร.ปรีชา เทียนสมประสงค์)

---------- กรรมการ

(ผศ.ศิริศักดิ์ เดชะทวีกุล)

ลิขสิทธิ์ของภาควิชาฟิสิกส์ประยุกต์ คณะวิทยาศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

หัวข้อโครงงานพิเศษ	เครื่องวัดมุมกับพารามิเตอร์ต่าง ๆ
นักศึกษา	นายนาถ ดนตรี
	นายสุวรรณ ลิขิตวัฒนานุรักษ์
อาจารย์ที่ปรึกษา	ผศ.ดร.อารีย์ วิเชียรฉาย
ภาควิชา	ฟิสิกส์ประยุกต์
ปีการศึกษา	2536

บทคัดย่อ

โครงงานพิเศษนี้เป็นการเก็บข้อมูลขององศาโดยใช้จาน Encoder ผ่านตัว Sensor ของแสงโดยใช้ Photo Diode และ Photo Transistor เป็นตัวตรวจวัด รหัสของ องศาต่าง ๆ จากแผ่น Encoder ซึ่งแผ่น Encoder ได้นำรหัสเกรย์มาประยุกต์ใช้ ซึ่ง รหัสเกรย์ใช้มากในระบบการตรวจรับสัญญาณด้วยแสง รหัสแบบนี้เป็นแบบ Non Weighted ในระหว่างกลุ่มรหัส ที่เรียงลำดับกันไปจะมีการเปลี่ยนแปลงของรหัสครั้งละ 1 บิต เท่านั้น ทำให้โอกาสความผิดพลาดในการรับรหัสน้อยมาก นำค่าขององศามาใช้กับพารามิเตอร์ ต่าง ๆ ในโครงงานนี้ได้นำมาใช้กับค่าของความเข้มแสง ณ องศาต่าง ๆ จาก 0-180 องศา โดยได้ทำการศึกษา และออกแบบเครื่อง Goniometer วงจรขับสเตปมอเตอร์ ซึ่งถูกควบคุมด้วยไมโครคอมพิวเตอร์โดยผ่านการ์ดควบคุม ที่ใช้ IC เบอร์ 8255 เป็น ตัวควบคุมการหมุนของ Stepping Motor วงจรแปลงสัญญาณอนาล็อกไปเป็นดิจิตอล และ ทำการเก็บข้อมูลที่ได้จากตัว Solar Cell ณ องศาต่าง ๆ ที่ละ 1.8 องศา ผ่านวงจร แปลงสัญญาณอนาล็อกเป็นดิจิตอล ทำการเก็บข้อมูลที่ได้เข้าไมโครคอมพิวเตอร์เพื่อทำการ สร้างกราฟระหว่าง ความเข้มแสง (V) กับ ตำแหน่งองศา โดยใช้ภาษาปาสคาลทำการ ประมวลผลและสร้างกราฟตลอดจนควบคุมการหมุนของ Stepping Motor

Special Project title	Geoniometer
Name	Mr.Nart Dontree
	Mr.Suwan likitwattananurak
Special Project Advisor	Dr.Aree Wichaeanchai
Department	Applied Physics
Acedemic Year	1993

Abstract

In this project, we read in the angles by using an encoder plate which turns around and passes between the light source (photo diode) and a photo detector (photo transistor). The encoding plate which tells the angle by codes, which lets the light from the light source to pass into the light detector or not. The encoder plate has been encoded by a method called "Gray Code", which is oftenly used. This kind of code is a "non weighted code", which the bits in the byte change and one bit a time , this makes the errors received very small. At each angle (from 0-180), the interest of the light is measured. This system is called a "Goniometer", which is driven by a stepping motor and controled through IC 8255 from the Microcomputer. The interes at each angle is measured in an analog to digital converter, the data is stored in the computer, then the computer creates graph between the intensity of light (V) and the angle. Pascal language is used to analyze the data, create the graph and control the stopping motor.

กิตติกรรมประกาศ

โครงงานพิเศษนี้สามารถเสร็จสมบูรณ์ได้ ด้วยความช่วยเหลือจากบุคคลต่าง ๆ ดังนี้

ผศ.ดร.อารีย์ วิเชียรฉาย	ที่คอยให้คำปรึกษาด้านทฤษฎีและรายละเอียดต่าง ๆ อีกทั้งคอยให้กำลังในในการทำงาน
อ.วิชิต ศิริโชติ	ที่คอยให้คำปรึกษาในด้านวงจรต่าง ๆ อีกทั้งคอยให้กำลังใจในการทำงาน
อ.ธวัชชัย	ที่คอยให้ความช่วยเหลือและความสะดวกในด้านอุปกรณ์เครื่องกล
คุณ ทรงวิทย์ เจริญรวยวัฒนา	ที่ให้ความช่วยเหลือทางด้านซอร์ฟแวร์
คุณ สมศักดิ์ ปัณฑุวงศ์	ที่ให้ความช่วยเหลือในด้านซอร์ฟแวร์
คุณ ภูเบศร์ อุดมทรัพย์	ที่ให้ความช่วยเหลือทางด้านวงจรอิเล็กทรอนิกส์
คุณ สุภัทชดา ผาสุข	ที่ให้ความช่วยเหลือเรื่องรายงานโครงงาน
คุณ นรสีห์ สิงหลกะ	ที่ให้ความช่วยเหลือในด้านภาษาต่างประเทศ

เพื่อน ๆ และ น้อง ๆ ทุกคนที่ให้ความช่วยเหลือและเอื้อเฟื้อในด้านต่าง ๆ และคอยให้กำลังใจ จึงขอขอบคุณกลุ่มบุคคลเหล่านี้เป็นอย่างสูง

สารบัญ

สารบัญตาราง

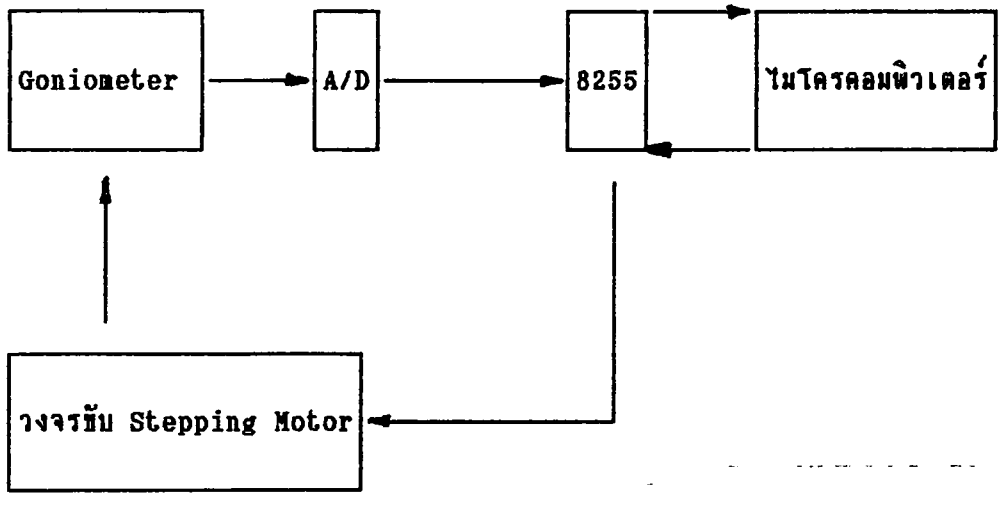
สารบัญรูป

สารบัญรูป

บทที่ 1

บทนำ

โครงงานพิเศษเรื่อง Goniometer นี้ เป็นการทำเครื่องมือที่ใช้วัดหาค่ามุมที่ใช้กับพารามิเตอร์ต่าง ๆ ซึ่งในโครงงานนี้ใช้หาค่ามุมของพารามิเตอร์ความเข้มแสงรอบแหล่งกำเนิด เป็นมุม 180 องศา โดยที่ Goniometer จะมีส่วนประกอบสำคัญ ๆ ตามรูปที่ 1.1



รูปที่ 1.1 แสดงแผนภาพการทำงานของโครงงานพิเศษ

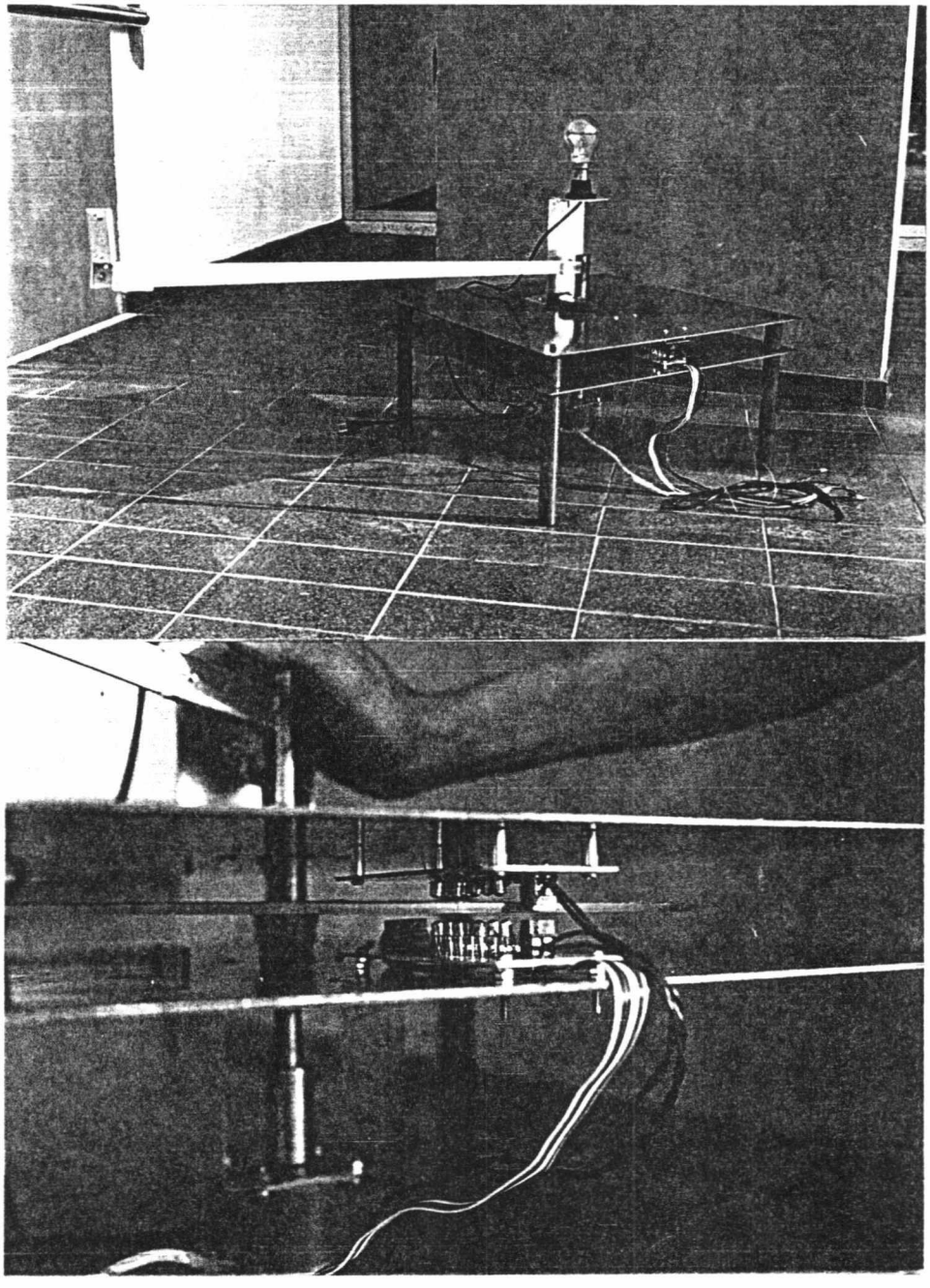
จากรูปจะเห็นว่าในโครงงานพิเศษประกอบไปด้วยส่วนสำคัญ ๆ ดังนี้

1. Goniometer
2. วงจรแปลงสัญญาณอนาลอกเป็นสัญญาณดิจิตอล
3. การ์ด 8255 PPI
4. วงจรขับ Stepping Motor

อธิบายส่วนประกอบต่าง ๆ ของสังเขปดังนี้

1. Goniometer

เป็นเครื่องกลที่มี แหล่งกำเนิดแสงติดอยู่ทางด้านบน ส่วนข้างใต้นั้นจะติดอุปกรณ์ที่ใช้ถอดรหัสตำแหน่งและ Stepping Motor และจะมีแขนที่ติดตัวตรวจวัด ซึ่งแขนนี้สามารถหมุนกวาดได้ 180 องศา



รูปที่ 1.2 Goniometer

2.วงจรแปลงสัญญาณอนาลอกเป็นสัญญาณดิจิตอล

เป็นวงจรที่แปลงสัญญาณอนาลอกที่ได้รับจาก Goniometerไปเป็นสัญญาณดิจิตอล เพื่อนำไปเชื่อมต่อเข้ากับไมโครคอมพิวเตอร์ ในโครงงานนี้ จะใช้ IC เบอร์ 7409 ทำการแปลงสัญญาณอนาลอกเป็นสัญญาณดิจิตอล ซึ่งมีการทำงานเป็นแบบสโลบคู่

3.การ์ด 8255 PPI

เป็นส่วนที่ใช้ในการเชื่อมต่อวงจรขับ Stepping Motor กับไมโครคอมพิวเตอร์ และใช้ในการควบคุมทิศทางการหมุนของ Stepping motor เนื่องจาก Stepping Motor จะทำงานโดยมีการกระตุ้นแบบเป็นเฟส จึงใช้พอร์ท A ของ 8255 ในการควบคุมทิศทางการหมุนของมอเตอร์ และใช้ติดต่อกับวงจรแปลงสัญญาณอนาลอกเป็นสัญญาณดิจิตอล เพื่อรับข้อมูลที่ได้ทำการแปลงเป็นสัญญาณดิจิตอล

4.วงจรขับ Stepping Motor

เป็นวงจรขับกำลังให้ Stepping Motor เพื่อให้ Stepping Motor สามารถหมุนได้

วัตถุประสงค์ของโครงงานพิเศษนี้

1.เพื่อศึกษาลักษณะการกระจายของความเข้มแสงจากแหล่งกำเนิด

2.เพื่อนำไปประยุกต์ใช้หาค่าพารามิเตอร์อื่น ๆ เช่น ความดังของเสียง การแหร่ของรังสีโดยการเปลี่ยนตัวตรวจวัด

ขั้นตอนการทำโครงงานพิเศษ

1. ทำการออกแบบและสร้างเครื่อง Goniometer
2. ศึกษาการทำงาน และออกแบบวงจรขับ Stepping Motor เพื่อใช้ในการหมุน Stepping Motor ที่เครื่อง Goniometer
3. ศึกษาและทำการออกแบบ การ์ด 8255 ซึ่งใช้ในการติดต่อเพื่อควบคุมทิศทางการหมุนของ Stepping Motor และส่งผลข้อมูลที่ทำการวัดได้ไปยังเครื่องไมโครคอมพิวเตอร์
4. ทำการศึกษา และออกแบบวงจรแปลงอนาลอกเป็นสัญญาณดิจิตอลเนื่องจากข้อมูลที่ได้จาก Goniometer เป็นสัญญาณอนาลอก ซึ่งสัญญาณที่ส่งไปยังเครื่องไมโครคอมพิวเตอร์นั้นต้องเป็นสัญญาณดิจิตอล เพื่อแสดงผล จึงต้องทำการแปลงสัญญาณ
5. ทำการเขียนโปรแกรมเพื่อควบคุมวงจรขับ Stepping Motor และเพื่อแสดงผลของข้อมูลที่ได้รับมาจาก Goniometer

บทที่ 2

การเข้ารหัส

การเข้ารหัสของข่าวสารคือการสร้างตัวเลขหรือตัวอักษร โดยการใช้สัญลักษณ์ใด ๆ เพื่อความปลอดภัยของข่าวสารในการป้องกันไม่ให้ผู้อื่นรับรู้ข่าวสารนั้น สำหรับในเรื่องของคอมพิวเตอร์คือการเปลี่ยนตัวอักษรหรือตัวเลขฐานสิบให้เป็นกลุ่มของเลขฐานสอง โดยเป็นรหัสทางระดับของแรงดัน กลุ่มของเลขฐานสองดังกล่าวจะใช้ติดต่อกับเครื่องคอมพิวเตอร์ ซึ่งมีอยู่หลายรูปแบบด้วยกัน แล้วแต่หน้าที่การทำงานและความเหมาะสมต่าง ๆ ต่อไปจะได้กล่าวถึงรหัสในรูปแบบต่าง ๆ ที่นิยมใช้กันในเครื่องพิวเตอร์

2.1 รหัส BCD-8421

รหัส BCD-8421 (Binary Code Decimal) เป็นรหัสที่ใช้กันบ่อย ซึ่งปกติจะประกอบด้วยเลขฐานจำนวน 4 ตัว เรียกว่า 4 บิต เริ่มตั้งแต่บิต 0 จนถึงบิต 3 แต่ละบิตมีลักษณะไม่เหมือนกัน ดังแสดงในตารางที่ 2.1

เลขฐานสิบ	BCD-8421
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

ตารางที่ 2.1 BCD-8421

2.2 รหัสเลขฐานแปดและรหัสเลขฐานสิบหก

รหัสนี้ถูกพัฒนามาจากรหัส BCD-8421 เพราะว่าในกรณีที่นำเอาเลขมาบวกกัน เช่น 7+5 = 12 หรือ 0111 + 0101 = 1100 นี้ไม่มีในรหัส BCD-8421 จึงได้ใช้รหัสนี้แทน คือพิจารณาทีละ 3 บิตจะเป็นรหัสเลขฐานแปด และถ้าแทนทีละ 4 บิตจะเป็นรหัสเลขฐานสิบหก

รหัสเลขฐานแปด

เช่น 101 001 010 010 011 010

5 1 2 2 3 2 = $(512232)_8$

รหัสฐานสิบหก

เช่น 1010 1001 0100 1001 1010

A 9 4 9 A = $(A949A)_{16}$

2.3 การใช้พาริตี้ในรหัส

ในการส่งข่าวสารทางสัญญาณดิจิตอล (Digital) นั้น เราสามารถตรวจสอบความถูกต้องของข่าวสารที่ส่งออกไปได้ ซึ่งวิธีที่นิยมใช้กันก็คือการใช้พาริตี้บิต (Parity Bit)

พาราตี้บิตคือบิต (เลข 0 หรือ 1) ที่เติมเข้าไปใน Code Worc ใด ๆ ก็ตาม แล้วทำให้ Code Worc นั้น ๆ มีจำนวนของเลข 1 เป็นจำนวนคู่ (even) หรือคี่ (odd) ก็ได้ตามต้องการ การใช้พาริตี้ในรหัสจิงแบ่งออกเป็น 2 แบบ คือ

1.เลขจำนวนคู่ (even parity) เช่น 01111

2.เลขจำนวนคี่ (odd parity) เช่น 01011

เลขฐานสิบ	รหัส BCD-8421	BCD with oodparity	BCD with even parity
0	0000	0000 1 หรือ 00001	0000 0 หรือ 00000
1	0001	0001 0 หรือ 00010	0001 1 หรือ 00011
2	0010	0010 0 หรือ 00100	0010 1 หรือ 00101
3	0011	0011 1 หรือ 00111	0011 0 หรือ 00110
4	0100	0100 0 หรือ 01000	0100 1 หรือ 01001
5	0101	0101 1 หรือ 01011	0101 0 หรือ 01010
6	0110	0110 1 หรือ 01101	0110 0 หรือ 01100
7	0111	0111 0 หรือ 01110	0111 0 หรือ 01111
8	1000	1000 0 หรือ 10000	1000 1 หรือ 10001
9	1001	1001 1 หรือ 10011	1001 0 หรือ 10010

ตารางที่ 2.2 รหัส BCD-8421

2.4 รหัสเกิน 3

รหัสเกิน 3 ดัดแปลงมาจากรหัส BCD-8421 เมื่อเปรียบเทียบรหัสเกิน 3 กับรหัส BCD-8421 ตามตาราง จะเห็นได้ว่ารหัสเกิน 3 จะมีค่ามากกว่ารหัส BCD-8421 อยู่ 3

เลขฐานสิบ	รหัส BCD–8421	รหัสเกิน 3
0	0000	0011
1	0001	0100
2	0010	0101
3	0011	0110
4	0100	0111
5	0101	1000
6	0110	1001
7	0111	1010
8	1000	1011
9	1001	1100

ตารางที่ 2.3 การเปรียบเทียบระหว่างรหัส BCD-8421 กับรหัสเกิน 3

2.5 รหัสเกรย์

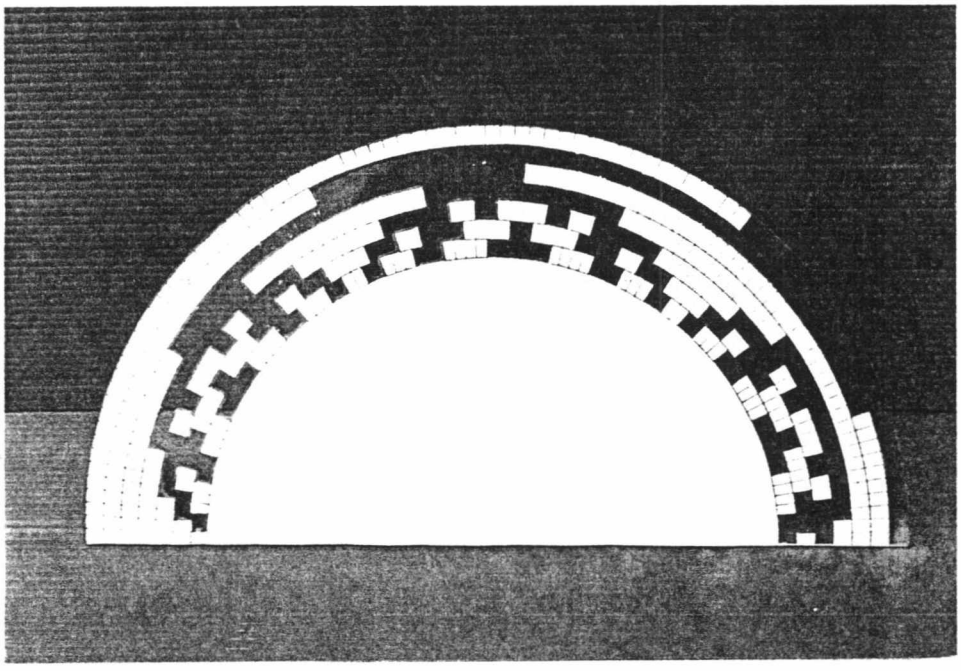
รหัสเกรย์ใช้กันมากในระบบการตรวจจับสัญญาณด้วยแสง หรือระบบทำรหัสด้วยแกนหมุนทางกล เพื่อบอกตำแหน่งของเพลาหมุน รหัสแบบนี้เป็นแบบ non weighted ในระหว่างกลุ่มรหัส (code group) ที่เรียงลำดับกันไปจะมีการเปลี่ยนแปลงของรหัสครั้งละ 1 บิตเท่านั้นทำให้โอกาสความผิดพลาดในการรับรหัสเป็นไปได้น้อยมาก

การเปลี่ยนรหัสเลขฐานสองให้เป็นรหัสเกรย์ สามารถทำได้โดยนำบิตที่ 2 ดังลงมาเป็นคำตอบต่อจากนั้นนำเอาบิตที่ 2 และบิตที่ 1 มาเปรียบเทียบกัน ถ้าบิตทั้งสองไม่เหมือนกันผลลัพธ์ที่ได้จะเป็น "1" เสมอ แต่ถ้าเปรียบเทียบกันแล้วเหมือนกัน ผลลัพธ์ที่ได้จะเป็น "0" จากนั้นทำเช่นนี้ไปเรื่อย ๆ จนถึงบิตสุดท้าย

เลขฐานสิบ	เลขฐานสอง	รหัสเกรย์
0	0000	0000
1	0001	0001
2	0010	0011
3	0011	0010
4	0100	0110
5	0101	0111
6	0110	0101
7	0111	0100
8	1000	1100
9	1001	1101
10	1010	1111
11	1011	1110

ตารางที่ 2.4 รหัสเกรย์

ในโครงงานพิเศษนี้ได้นำรหัสเกรย์มาประยุกต์ใช้ในการตรวจจับสัญญาณด้วยแสง โดยใช้ Photo Diode MLED 930 และ Photo Transistor MRD 370 เพื่อบอกตำแหน่งจากจานหมุน Encoder ทำการเก็บค่า แล้วทำการแปลงเป็นค่าขององศาตามที่กำหนด



รูป 2.1 จานหมุน Encoder

บทที่ 3

ทฤษฎีและหลักการทำงานของ Stepping Motor

Stepping Motor เป็น DC Motor ที่มีการทำงานโดยเคลื่อนที่ไปทีละสเตป อย่างแม่นยำ การควบคุมการทำงาน Stepping Motor จะทำโดยการกระตุ้นทีละเฟส เรียงกันไป ซึ่งการกระตุ้นครั้งหนึ่งจะทำให้มอเตอร์ เคลื่อนที่ไปหนึ่งสเตป ถ้าเปรียบเทียบ Stepping Motor กับมอเตอร์ ทั่ว ๆ ไปแล้ว Stepping Motor จะมีข้อได้เปรียบที่สำคัญกว่ามอเตอร์ ทั่วไป 3 ประการคือ

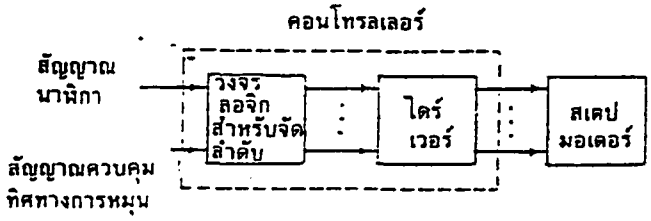
- ไม่ต้องมี Feedback ในการควบคุมตำแหน่ง และความเร็ว
- ไม่มีความผิดพลาดของตำแหน่ง
- Stepping Motor เหมาะกับอุปกรณ์ Digital สมัยใหม่

ด้วยเหตุผลนี้ Stepping Motor จึงเหมาะที่จะใช้กับการควบคุมด้วยไมโครโปรเซสเซอร์และคอมพิวเตอร์

3.1 นิยามของ Stepping Motor

- จะมีเพลาหมุนเป็นสเตป
- ป้อนอินพุทที่มีความถี่ค่าหนึ่ง
- จะหมุนไปทีละสเตป ในแต่ละพัลส์
- ขนาดของสเตป ขึ้นอยู่กับการออกแบบ Stepping Motor
- จะสามารถควบคุมการเคลื่อนที่ด้วยความเร็ว

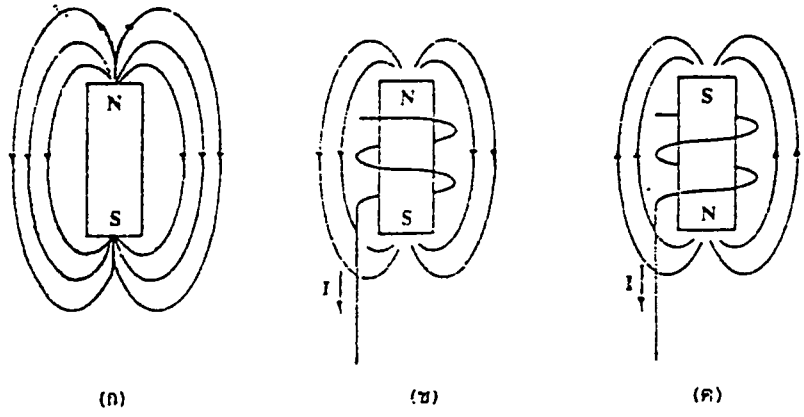
การทำงานของ Stepping Motor ขึ้นอยู่กับพัลซ์ที่ป้อนให้กับขดลวดแต่ละเฟสของ Motor ในลำดับที่ถูกต้อง ด้วยวงจรจัดลำดับลอจิก (Sequencer Logic) ด้วยกระแสที่พอเพียง จากวงจรขับ (Drive) การควบคุมการหมุนของ Stepping Motor สามารถควบคุมได้โดยวงจรจัดลำดับลอจิก ดังรูป 3.1



จากรูป 3.1 Stepping Motor จะทำงานเมื่อมีการป้อนสัญญาณนาฬิกา (Clock pulse) และอินพุทสำหรับควบคุมทิศทาง

3.2 หลักการทำงานของ Stepping Motor

ในรูปที่ 3.2 แสดงหลักการพื้นฐานของเส้นของเส้นแรงแม่เหล็ก รูป ก แสดงสนามแม่เหล็กที่เกิดจากแม่เหล็กถาวร รูป b แสดงสนามแม่เหล็กไฟฟ้าที่เกิดจากกระแส I ในรูป ค ขั้วแม่เหล็กกลับทิศทางเมื่อขดลวดถูกพันกลับทิศทาง และทิศทางการไหลของกระแสไม่เปลี่ยนแปลง

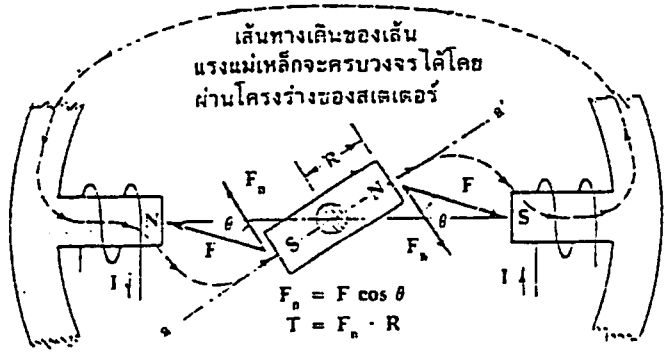


รูปที่ 3.2 แสดงถึงสนามแม่เหล็กที่เกิดขึ้นในลักษณะต่าง ๆ

ก) สนามแม่เหล็กที่เกิดขึ้นจากแม่เหล็กถาวร

ข) สนามแม่เหล็กไฟฟ้าที่เกิดจากกระแส I

ค) ขั้วแม่เหล็กกลับทิศทาง เมื่อขดลวดถูกพันกลับทิศ การไหลของกระแสไม่เปลี่ยนแปลง



รูปที่ 3.3 แสดงแรงดิงดูดทำให้เกิดแรงบิดที่หมุนอเมเจอร์ให้ไปอยู่ในตำแหน่งสมดุลย์

ในรูป 3.3 แท่งแม่เหล็กถาวรติดอยู่บนเพลา และหมุนได้อิสระเหมือนอเมเจอร์มีขั้วแม่เหล็กไฟฟ้า 2 ขั้ว ซึ่งเป็นส่วนหนึ่งของโครงโลหะที่เป็นสเตเตอร์ (Stator) ตำแหน่งแกนของขั้วแม่เหล็กคือ a-a' ซึ่งต่างไปจากแกนขั้วแม่เหล็กไฟฟ้าเล็กน้อยเป็นมุม •

แรงแม่เหล็กที่เกิดจากการดึงดูดของขั้วแม่เหล็กที่ต่างกันทำให้เกิดส่วนของแรง

$F_n = F\cos\bullet$ (แรงทั้งหมดนี้ตั้งฉากกับแกน a-a')

แรงบิด หรือทอร์คผลรวม

$T = F_nR$ (ทำให้อเมเจอร์หมุนไปในทิศทางตามเข็มนาฬิกาจนกว่าแกนของอเมเจอร์ a-a' จะอยู่ในแนวเดียวกับแกนของขั้วสเตเตอร์)

ถ้าหากมีคู่ขั้วแม่เหล็กไฟฟ้าหลาย ๆ คู่ ขั้วรอบ ๆ สเตเตอร์ และถ้าหากขั้วเหล่านี้ถูกกระตุ้นด้วยกระแสพัลซ์ในรูปแบบที่เรียงลำดับกันไป อเมเจอร์ก็จะหมุนในรูปลักษณะของสเตป ที่เป็นไปตามการหมุนของสนามแม่เหล็กที่เกิดจากการสวิทช์ที่เรียงลำดับของขดลวดขั้วแม่เหล็กไฟฟ้าบนสเตเตอร์

3.3 การแบ่งชนิดของ Stepping Motor

แบ่งชนิดตามลักษณะโครงสร้างได้ 3 ชนิด คือ

3.3.1 Variable reluctance Stepping Motor

3.3.2 Permanent Magnet Stepping Motor

3.3.3 Hybrid Stepping Motor

3.3.1 Variable reluctance Stepping Motor

มอเตอร์ชนิดนี้มีโรเตอร์เป็นซี่ฟัน และเป็นเหล็กอ่อน ตัวสเตเตอร์ถูกพันด้วยขดลวดตามปกติ การหมุนเกิดขึ้นได้โดยเราให้กระแสไฟฟ้าต่อขดลวดที่พันบนสเตเตอร์ทำให้เกิดอำนาจแม่เหล็กไปดึงดูดให้โรเตอร์หมุนได้ตามตำแหน่งของ Stator Pole ที่ต้องการ Rotor Inertia ของมอเตอร์ชนิดนี้มีค่าต่ำและมีการตอบสนองที่เร็ว ถ้าขดลวดไม่ได้ถูก energize แล้ว Static torqe ของมอเตอร์ชนิดนี้จะมีค่าเป็นศูนย์

- VR Stepping Motor แบบมีสแต็คเดียว

โครงสร้างของ VR Stepping Motor แสดงไว้ดังรูปที่ 3.4 VR Stepping Motor ที่มีสแต็คเดียวจะมีโรเตอร์เดียว และ VR Stepping Motor แบบหลายสแต็คจะหมายถึงมีหลายโรเตอร์ ซึ่งโรเตอร์และสเตเตอร์ทำจากสารแม่เหล็กส่วน Stepping Motor ในรูปที่ 3.4 มี 3 เฟส แต่ละเฟสใช้ขดลวดพันบน 4 ขั้วหรือชี่ฟันของสเตเตอร์

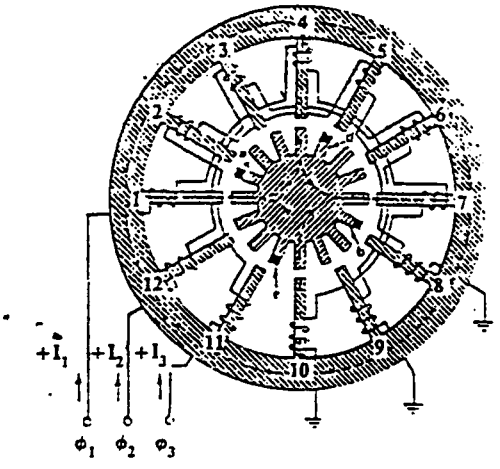
ตัวอย่าง เฟสที่ 1 พันอยู่ที่ 1,4,7,9 ของสเตเตอร์ ดังนั้นสเตเตอร์จะมี 12 ชี่ฟันและในที่นี้กำหนดให้โรเตอร์มี 16 ชี่ฟัน ขั้วของสเตเตอร์ที่อยู่ตรงกันข้ามจะพันด้วยขดลวดในลักษณะที่ต่างกันเพื่อให้มีความสมดุลย์ระหว่างเส้นแรงแม่เหล็กเข้า และออกจากโรเตอร์

โดยที่ เฟส (1) - 1,4,7,10

เฟส (2) - 12,3,6,9

เฟส (3) - 2,5,8,11

(ดังรูปที่ 3.4)



รูปที่ 3.4 VR Stepping Motor แบบสแต็คเดียว

สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

ซึ่งมีรายละเอียดดังนี้ $N_R = 16, N_s = 12, X = 4$ โพล

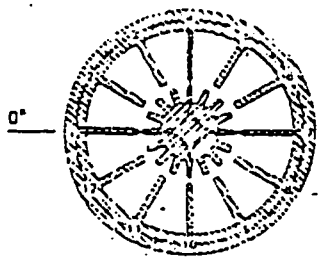
$\theta_s = 7.5$ องศา , $R_s = 48$ สเตป/รอบ

สมมติว่ากระแส I_1 ป้อนให้กับเฟสที่ 1 ดังแสดงดังรูปที่ 3.4 และโรเตอร์ทั้ง 4 ซี่ฟันจะอยู่ในแนวซี่ฟันที่ 1,4,7,10 ของสเตเตอร์ เส้นแรงแม่เหล็กจะเข้าสู่โรเตอร์จากซี่ฟันที่ 4 และ 10 และออกจากโรเตอร์ไปยังซี่ฟันของสเตเตอร์ที่ 1 และ 7 ซึ่งเป็นทางเดินของเส้นแรงแม่เหล็กที่ครบวงจรโดยผ่านโครงร่างของสเตเตอร์ เราจะสังเกตได้ว่าปลายของซี่ฟันโรเตอร์ ซึ่งอยู่ในแนวเดียวกับซี่ฟันที่ 4 ของสเตเตอร์จะเป็นเส้นทางผ่านเข้าไปยังโรเตอร์ของเส้นแรงแม่เหล็กอย่างต่อเนื่องผ่านช่องว่าง (Gap) ระหว่างซี่ฟันทั้งสองที่อยู่ในแนวเดียวกัน ส่วนซี่ฟันของสเตเตอร์และโรเตอร์ที่เหลืออีก 3 คู่ก็เกิดลักษณะของแม่เหล็กในทำนองเดียวกัน

ในสภาวะต่อไปเราจะให้โรเตอร์หมุนไป 1 สเตปในทิศทางตามเข็มนาฬิกาเราจะต้องจ่ายพลังงานให้กับเฟส 3 ที่มีขดลวดพันอยู่บนซี่ฟันที่ 2 ,5 ,8 ,11 ของสเตเตอร์ด้วยกระแส I_3 หลังจากหยุดกระแส I_1 แล้ว ในตอนนี้เส้นแรงแม่เหล็กจะหาทางเดินที่ต่างไปจากเดิม เพื่อทำให้วงจรแม่เหล็กครบวงจร (เหมือนกับกระแสไฟฟ้าจะหาเส้นทางไหลในส่วนที่ความต้านทานต่ำสุด) ในทำนองเดียวกันเส้นแรงแม่เหล็กในวงจรแม่เหล็กก็จะหาทางเดินที่มีรีลัคแตนซ์ต่ำสุด (ช่องว่างอากาศระหว่างซี่ฟัน จะทำให้เกิดค่ารีลัคแตนซ์ต่อเส้นแรงแม่เหล็กช่องว่างกว้างมากค่ารีลัคแตนซ์ก็จะมีค่ามาก) ด้วยเหตุผลดังกล่าวเส้นแรงแม่เหล็กจะออกจากขั้วที่ 2 และ 8 ของสเตเตอร์ซึ่งถูกทำให้เป็นขั้วเหนือ และเส้นแรงแม่เหล็กนี้ก็จะกระโดดผ่าช่องว่างไปยังซี่ฟันของโรเตอร์ที่ใกล้ที่สุด ซี่ฟัน a และ b ของโรเตอร์เป็นโรเตอร์ที่อยู่ใกล้ที่สุดและถูกเหนี่ยวนำให้เป็นขั้วได้ เส้นแรงแม่เหล็กจะออกจากซี่ฟัน d และ e ของโรเตอร์ผ่านช่องว่างอากาศเข้าสู่ซี่ฟันที่ 5 และ 11 ของสเตเตอร์ ดังนั้นส่วนที่เหลือของวงจรแม่เหล็กจะสมบูรณ์โดยผ่านโครงร่างของสเตเตอร์ ในระหว่างเวลานั้นแรงแม่เหล็กหรือแรงดึงดูดจะเกิดขึ้นระหว่างซี่ฟันที่ 2 ของสเตเตอร์ (ถูกเหนี่ยวนำเป็นขั้วเหนือ) และซี่ฟัน a ของโรเตอร์ (ถูกเหนี่ยวนำเป็นขั้วใต้) แรงดึงดูดจะเกิดขึ้นระหว่างคู่ขั้ว (11,e),(8,b),(5,d) ดังได้อธิบายในรูปที่ 3.3 ผลที่เกิดขึ้นนี้จะทำให้เกิดทอร์คกระทำต่อโรเตอร์หมุนไปจนกระทั่งซี่ฟัน a,d,b,e ของโรเตอร์อยู่

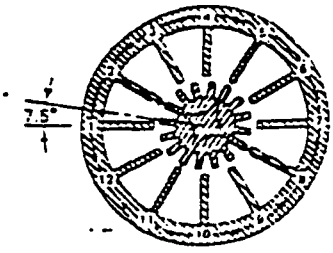
ในแนวเดียวกับชี่ฟัน 2,5,8และ 11 ของสเตเตอร์ตามลำดับ ขณะเวลาดังกล่าวช่องว่างระหว่างชี่ฟันตามลำดับจะมีค่าน้อยที่สุด ผลลัพธ์ของค่ารีลัคแตนซ์จะมีค่าต่ำที่สุดและเส้นแรงแม่เหล็กจะมีค่าสูงสุดผ่านวงจรแม่เหล็ก ที่ตำแหน่งนี้เป็นตำแหน่งที่สมดุลย์ของการขับเฟส 3 ในขบวนการที่กล่าวมาแล้วโรเตอร์จะเคลื่อนในทิศทางตามเข็มนาฬิกา หนึ่งสเตปเป็นมุม 7.5 องศา

ลำดับการทำงานที่สมดุลย์แสดงได้ดังรูปที่ 3.5 เมื่อตำแหน่งเริ่มต้นของชี่ฟันของโรเตอร์จะเป็นสีดำ เพื่อให้เราทำความเข้าในได้ชัดเจน ถึงการหมุนของโรเตอร์ในทิศทางตามเข็มนาฬิกา เมื่อเฟสถูกขับในลักษณะเรียงลำดับ 1-3-2-1 ชี่ฟันของโรเตอร์ที่เป็นสีดำจะเคลื่อนที่ไป 3 สเตปคิดเป็นมุมได้เท่ากับ 22.5 องศา เราจะขับเฟสในลักษณะเรียงลำดับเดิมซ้ำใหม่อีกเมื่อต้องการให้โรเตอร์หมุนต่อเนื่องในทิศทางตามเข็มนาฬิกา แต่ถ้าเราต้องการให้โรเตอร์หมุนในทิศทางทวนเข็มนาฬิกา เราต้องกลับการเรียงลำดับเฟสเป็น 1-2-3-1

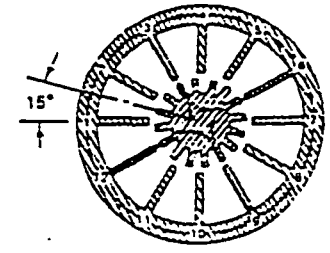
การเรียงลำดับเฟส	ตำแหน่งของโรเตอร์และเส้นแรงแม่เหล็ก
ตำแหน่งโรเตอร์เริ่มต้น: - เฟล ϕ_1 ได้รับพลังงาน - ชี่ฟันของโรเตอร์จะอยู่ในแนวชี่ฟัน ที่ 1,4,7,10 ของสเตเตอร์	

รูปที่ 3.5

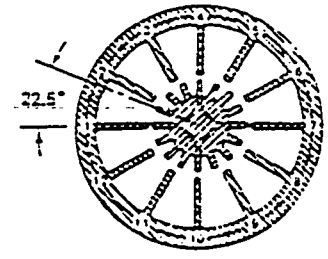
สเตปที่ 1: เฟส ϕ_3 ได้รับพลังงาน
- ซี่ฟันของโรเตอร์จะอยู่ในแนวซี่ฟันที่2,5,8,11 ของสเตเตอร์
- โรเตอร์จะเคลื่อนที่ไปในทิศทาง CW เป็นมุม 7.5 องศา (1/3 ช่องห่างระหว่างซี่ฟันของโรเตอร์)



สเตปที่ 2: เฟส ϕ_2 ได้รับพลังงาน
- ซี่ฟันของโรเตอร์จะอยู่ในแนวซี่ฟันที่ 3,6,9,12 ของสเตเตอร์
- โรเตอร์จะเคลื่อนที่ไปในทิศทาง CW รวมเป็นมุม 7.5 องศา



สเตปที่ 3: เฟส ϕ_1 ได้รับพลังงาน
- ซี่ฟันของโรเตอร์จะอยู่ในแนวซี่ฟันที่ 1,4,7,10 ของสเตเตอร์
- โรเตอร์จะเคลื่อนที่ไปในทิศทาง CW รวมเป็นมุม 22.5 องศา (เคลื่อนไปได้ 1 ช่องห่างระหว่างซี่ฟันของโรเตอร์)



รูปที่ 3.5 แสดงลำดับการสวิทช์ 3 สเตปของ VR Stepping Motor แบบสแต็คเดียวและแสดงถึงตำแหน่งของโรเตอร์และทางของเส้นแรงแม่เหล็ก เมื่อโรเตอร์เคลื่อนที่ไปในแต่ละสเตป

สัญญลักษณ์ต่าง ๆ ของ VR Stepping Motor

N_r = จำนวนซี่ฟันของโรเตอร์

N_s = จำนวนซี่ฟันของสเตเตอร์

N_p = จำนวนเฟส

P_r = ความห่างระหว่างปลายซี่ฟันของโรเตอร์ (องศา)

P_s = ความห่างระหว่างปลายซี่ฟันของสเตเตอร์ (องศา)

θ_s = มุมสเตป (องศา)

R_s = อัตราสเตปหรือความเร็วในการสเตป (สเตป/รอบ)

$X = N_s/N_p$ = จำนวนซี่ฟันของสเตเตอร์ต่อเฟส

พารามิเตอร์ต่าง ๆ ของ Stepping Motor

1.ความห่างระหว่างปลายซี่ฟันของโรเตอร์และสเตเตอร์ (tooth pitch)

$$P_r = 360/N_r \quad \text{และ} \quad P_s = 360/N_s$$

2.มุมสเตป (step angle)

ในรูป 2.5 โรเตอร์จะเคลื่อนที่ในขนาดมุม P_r ได้เท่ากับ N_p สเตป ดังนั้นเราจะหามุมสเตปได้

$$\theta_s = P_r/N_p = N_r/N_p \quad \text{องศา/สเตป}$$

มุมสเตปจะเท่ากับความแตกต่างระหว่าง P และ P ดังนั้นเราหามุมสเตปได้เป็น

$$\theta_s = \left|P_r - P_s\right| \quad \text{องศา/สเตป}$$

3.อัตราการสเตป (Stepping Rate)

ความเร็วในการสเตปต่อรอบ (360 องศา) หาได้เป็น

$$R_s = 360/\theta = N_r N_p \quad \text{สเตป/รอบ}$$

4.ความเร็วของ Stepping Motor (Speed of stepping motor)

เมื่อเราป้อนอินพุทพัลซ์ที่มีความถี่ (f) สเตปต่อพัลซ์ให้กับ Stepping Motor มอเตอร์ จะสเตปไปด้วยความเร็ว (สเตป/พัลซ์) x f (พัลซ์/วินาที)

$1/R_s$ (รอบ/สเตป) x f(พัลซ์/พัลซ์)(สเตป/พัลซ์) x 60(วินาที/นาที)

ความเร็วของมอเตอร์ (w) = $60f/R_s = 60f/N_pN_r = \theta_s f/6$(rpm)

5.จำนวนของโพลสเตเตอร์ต่อเฟส (number of stator poles per phase)

จำนวนของโพลสเตเตอร์ต่อเฟส (x) = N_s/N_p

หรือ

$$X = R_s/N_p(N_p+1) = N_r/(N_p+1)$$

จำนวนของโพลสเตเตอร์ต่อเฟส (x) จะสัมพันธ์กับอัตราการสเตปหรือจำนวนซี่ฟันของโรเตอร์ Stepping Motor ในรูปที่ 3.4 เราสามารถสรุปเลือกพารามิเตอร์บางตัวของ Stepping Motor ได้ดังตารางที่ 3.1

N_p	R_s	N_r	X	N_s
3	48	16	4	12
			8	24
4	48	12	4	16
4	64	16	7	7

ตารางที่ 3.1 แสดงการเลือกพารามิเตอร์ของ Stepping Motor

ตัวอย่าง การหาพารามิเตอร์ของ Stepping Motor

ขั้นแรกเรากำหนดความต้องการของมุมสเตป = 9 องศา

มุมสเตปจะเป็นตัวจำกัดอัตราการสเตป = 360/9 = 40 สเตป/รอบ

ในเงื่อนไขเหล่านี้เราอาจต้องใช้สเตปที่มี 4 หรือ 5 เฟส ที่มีสเตเตอร์ 2 โพล/เฟส

ถ้า $N_p = 4$

$R_s = R_s/N = 40/4 = 10$

$N_s = N_p * S = 4*2 = 8$

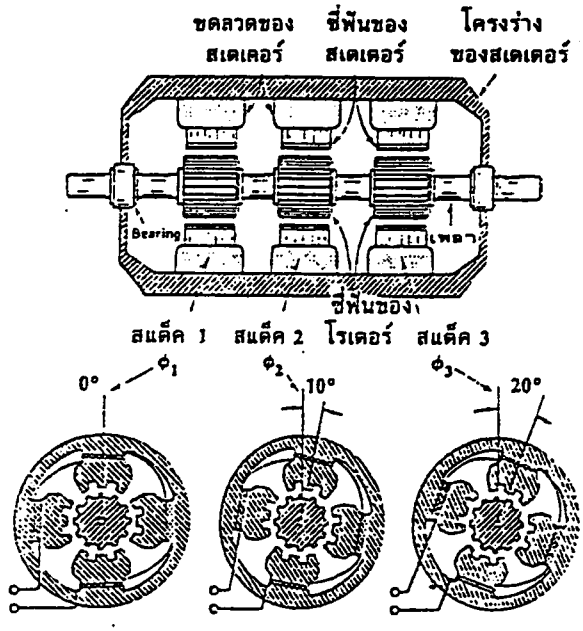
ถ้า $N_p = 5$

$N_R = 40/5 = 8$

$N_s = 5*2 = 10$

- VR Stepping Motor แบบหลายสแต็ค

สแต็คในที่นี้หมายรวมไปถึงโรเตอร์ที่เป็นชี่ฟัน และโครงข่ายของสเตเตอร์อยู่รอบนอก VR Stepping Motor แบบ 3 สแต็ค (หมายถึง 3 เฟส) มีโครงสร้างดังรูปที่ 3.6 ซึ่งถูกออกแบบให้สเตเตอร์ของแต่ละสแต็คประกอบด้วย 4 โพลและแต่ละโพลจะมีชี่ฟัน 3 ชี่ซึ่งต่างจาก VR แบบสแต็คเดียว (แต่ละโพลจะมีชี่เดียว) ข้อสังเกตในแต่ละสแต็ค จำนวนชี่ฟันของโรเตอร์และสเตเตอร์จะมีจำนวนเท่ากัน ซึ่งต่างกับ VR แบบสแต็คเดียวคือ จำนวนชี่ฟันของโรเตอร์และสเตเตอร์จะเท่ากันไม่ได้ ถ้าหากมีจำนวนชี่ฟันเท่ากันมันจะไม่ทำงาน



รูปที่ 3.6 แสดงโครงสร้างของ VR Stepping Motor ที่มี 3 เฟส

จากรูป 3.6 โรเตอร์ และสเตเตอร์ของแต่ละเฟส (สแต็ค) จะมี 12 ขี้ฟันและมุมสเตป (θ_s) = 10 องศา แต่ละเฟสของสเตเตอร์ที่เรียงลำดับต่อเนื่องกันจะถูกจัดตำแหน่งให้ต่างกัน = 1/3 ของช่องห่างระหว่างขี้ฟันของโรเตอร์ (10 องศา)

- การทำงานของ VR Stepping Motor ที่มี 3 สแต็ค

ไดอะแกรมส่วนล่างของรูปที่ 3.6 แสดงถึงโครงสร้างของโรเตอร์และสเตเตอร์ของ VR Stepping Motor ที่มี 3 สแต็ค โดยแต่ละสแต็คจะมี $N_r = N_s$ แต่ละสแต็คจะมีตำแหน่งของสเตเตอร์แตกต่างจากตำแหน่งถัดไป = 10 องศา ส่วนขี้ฟันของโรเตอร์ทั้ง 3 อัน จะประกอบอยู่บนแกนเดียวกันและได้รับการปรับแต่งให้อยู่แนวเดียวกันอย่างสมบูรณ์

ตามปกติเราจะหาค่ามุมสเตป (INDEX ANGLE) ได้จากสมการ

$$\theta_s = P_R/N_D = 360/N_R N_D$$

ในที่นี้เราจะหา θ_1 (INDEX ANGLE) ได้จากสมการเดียวกันคือ

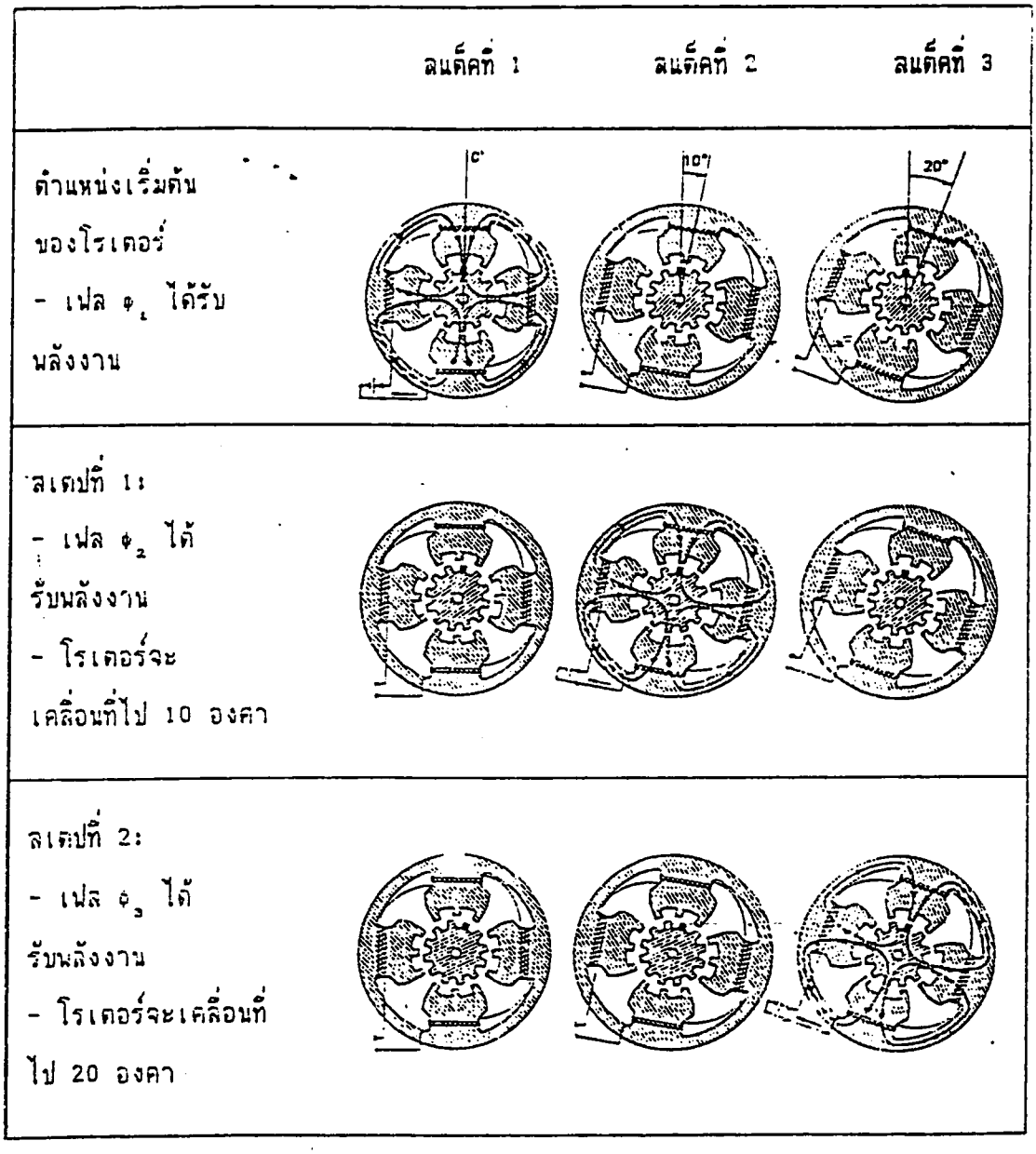
$$\theta_1 = P_R/N_D = \theta_s$$

ในกรณี $N_R = N_s = 12$ ดังนั้นเราหา $P_D = 360/12 = 30$ องศา และค่า $\theta_1 = 30/3 = 10$ องศา

Stepping Motor แบบ 3 สแต็ค ถึงแม้ว่าโรเตอร์ทั้ง 3 อันจะติดอยู่บนเพลาอันเดียวกันสแต็คทั้ง 3 จะมีวงจรแม่เหล็กที่แยกกันดังนี้

ถ้าเฟสที่ 1 ถูกขับด้วยกระแสเป็นเฟสเริ่มต้นให้ซี่ฟันของโรเตอร์ สเตเตอร์อยู่ในแนวเดียวกัน ส่วนซี่ฟันของโรเตอร์และสเตเตอร์ในสแต็คที่ 2 ในขณะนั้นจะมีตำแหน่งต่างกัน 10 องศา และซี่ฟันของโรเตอร์และสเตเตอร์ในสแต็คที่ 3 จะมีตำแหน่งต่างกัน 20 องศา ต่อจากนั้นเราหยุดจ่ายกระแส (กระแสขดลวดสเตเตอร์) ในสแต็คที่ 1 และป้อนกระแสให้กับสแต็คที่ 2 โรเตอร์จะหมุนไปอีก 10 องศา ซึ่งจะทำให้ซี่ฟันของโรเตอร์และสเตเตอร์ในสแต็คที่ 2 อยู่ในแนวเดียวกัน ในขณะนั้นซี่ฟันของโรเตอร์และสเตเตอร์ในสแต็คที่ 3 จะมีตำแหน่งต่างกัน 10 องศา ต่อจากนั้นเราจะหยุดจ่ายกระแสในสแต็คที่ 2 และป้อนกระแสในสแต็คที่ 3 โรเตอร์จะหมุนไปอีก 10 องศา ซึ่งจะทำให้ซี่ฟันของโรเตอร์และสเตเตอร์ในสแต็คที่ 3 อยู่ในแนวเดียวกัน ส่วนซี่ฟันของโรเตอร์ในสแต็คที่ 1 จะมีตำแหน่งต่างกัน 10 องศา

ลำดับการสวิทช์กระแสให้แต่ละสแต็คแสดงได้ดังรูปที่ 3.7 ซึ่งแสดงให้เห็นว่าเพลาของสเตเตอร์มอเตอร์จะเคลื่อนที่ไปเท่ากับ 1 ช่องของระยะห่างระหว่างซี่ฟันของโรเตอร์ (30 องศา) ภายใน 3 สเตป



รูปที่ 3.7 แสดงลำดับการสเตปของ VSRM แบบ 3 เฟส

รูปที่ 3.7 แสดงลำดับการสเตปของ VSRM แบบ 3 เฟส $N_r = N_s = 12$, $P_r = 30$ องศา และ $\theta_s = 10$ องศา ซี่ฟันของโรเตอร์สีดำจะเคลื่อนที่ไปในทิศทาง CW 10 องศา ในแต่ละสเตปรวมทั้งหมด 30 องศา เมื่อสเตปไปครบ 3 สเตปสำหรับการหมุนในทิศทาง CW ลำดับการขับเฟส 1-2-3-1 และเมื่อต้องการให้หมุนในทิศทาง CCW ลำดับการขับเฟสก็ต้องกลับเป็น 1-3-2-1

ตามปกติเพลาของมอเตอร์จะเคลื่อนที่ไป 1 ช่องของระยะห่างระหว่างซี่ฟันของโรเตอร์ (ROTOR TOOTH PITCH) ด้วยการสเตปไป N_p คือจำนวนสแต็คที่ใช้(หรือเท่ากับจำนวนเฟส)

ลำดับการสวิทช์ที่แสดงในรูปที่ 3.7 เราสามารถนำเข้ามาเขียนเป็นตารางได้ดังรูปที่ 3.8 วงจรการสวิทช์ประกอบด้วย VSRM แบบ 3 เฟส (สัญญาณของ Stepping Motor การขับเฟสได้ด้วยสวิทช์แหล่งกำเนิดดีซี

Step	S_1	S_2	S_3
1	X		
2		X	
3			X
1	X		

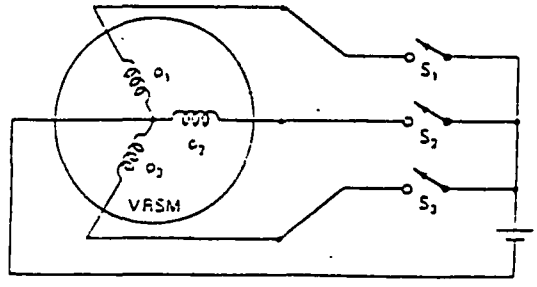
ก

Step	S_1	S_2	S_3
1	X	X	
2		X	X
3	X		X
1	X	X	

ข

Step	S_1	S_2	S_3
1	X	X	
2		X	
3		X	X
4			X
5	X		X
6	X		
1	X	X	

ค



ง

รูปที่ 3.8 แสดงถึง VRSM แบบ 3 เฟส

(ก) ตารางแสดงลำดับการขับเฟสเดียวในทิศทางตามเข็มนาฬิกา

(ข) ตารางแสดงลำดับการขับแบบ 2 เฟสในทิศทางตามเข็มนาฬิกา

(ค) การขับแบบครึ่งสเตปในทิศทางตามเข็มนาฬิกา

(ง) วงจรการสวิทช์เมื่อต้องการให้หมุนในทิศทางทวนเข็มนาฬิกา จะต้องกลับลำดับการขับในตาราง (a),(b),(c)

จากตาราง (ก) ถ้าเราขับเฟสที่ 1 และ 2 เรียงลำดับมอเตอร์จนหมุนไป 1 สเตป ตาราง (ข) ถ้าเราขับเฟสที่ 1 และ 2 พร้อมกัน เพลาของมอเตอร์จะหมุนไป 3/2 สเตป ต่อจากนั้นเราขับเฟสที่ 2 และ 3 พร้อมกันอีก ก็จะทำให้มอเตอร์หมุนไปครบเต็ม 1 สเตป ดังนั้นการขับ แบบ 2 เฟส เราเรียงลำดับการขับได้ดังนี้ 1-2,2-3,3-1,1-2 กระทำซ้ำเดิมไปเรื่อย ๆ

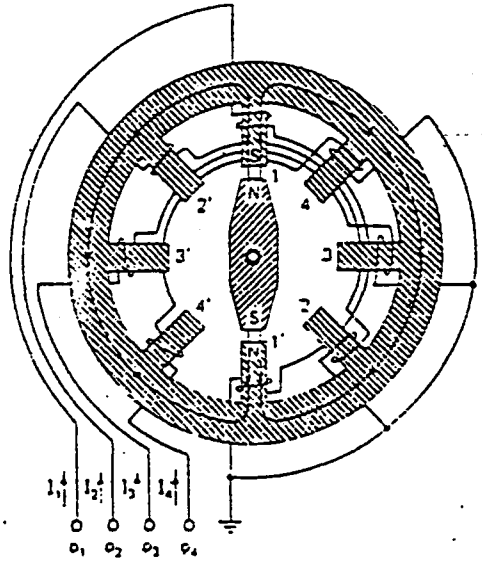
อย่างไรก็ตามการขับแบบ 2 เฟสหรือว่า 1 เฟสจะให้การหมุนเป็นสเตปเท่ากันที่ต่างกันก็คือการขับแบบ 2 เฟสจะให้การหมุนของโรเตอร์นำหน้าการขับเฟสเดียวด้วยขนาด 1/2 สเตป นอกจากนี้การขับแบบ 2 เฟสจะต้องการกระแสเป็น 2 เท่าของแบบเฟสเดียว

ตาราง (ค) แสดงการขับแบบ 2 เฟส สลับกับแบบเฟสเดียวซึ่งจะทำให้โรเตอร์หมุนไปครึ่งสเตปเท่านั้น การขับแบบนี้ จะทำให้จำนวนสเตปต่อรอบเพิ่มขึ้นเป็น 2 เท่าจากเดิม

3.3.2 <u>Permanent Magnet Stepping Motor</u>

มอเตอร์ชนิดนี้มีโรเตอร์เป็นแม่เหล็กถาวร ซึ่งมีอำนาจแม่เหล็กตามแนวรัศมีดึงดูดกับ Stator Pole มอเตอร์ชนิดนี้จะมี holding torqe เกิดขึ้นแม้ไม่ได้ถูก energize ที่ขดลวดบนสเตเตอร์ ดังนั้นในการออกแบบนี้สเตเตอร์จะต้องมี 8 ขั้ว

โครงสร้างของมอเตอร์ชนิดนี้แสดงไว้ดังรูปที่ 3.9



รูปที่ 3.9 โครงสร้างของสเตเตอร์แบบแม่เหล็กถาวรมี 4 เฟส และแต่ละเฟสพันด้วยขดลวดบน 2 ขั้วของสเตเตอร์มุมสเตป = 45 องศา

โรเตอร์ทำจากแม่เหล็กถาวรและอยู่ในแนวของขั้วสเตเตอร์ 1 และ 1' มันหยุดตำแหน่งนี้ได้ด้วยกระแส I_1 ที่ไหลอยู่ในเฟส 1

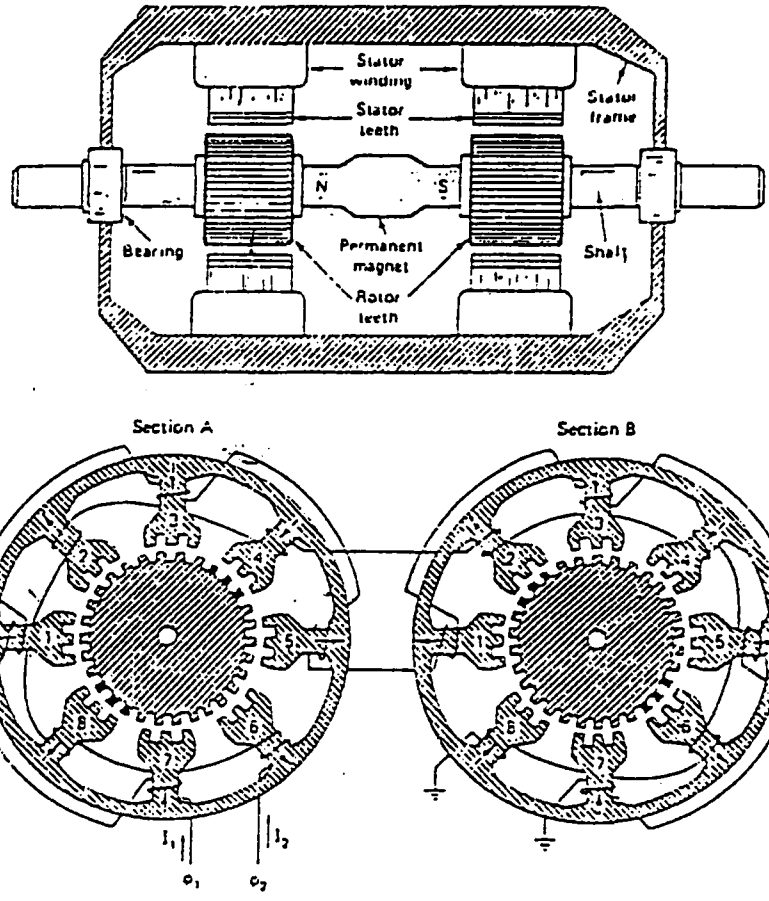
ขดลวดของเฟส ϕ_1, ϕ_4, ϕ_3 และ ϕ_2 (1-4-3-2 ตามลำดับ) จะได้รับพลังงานด้วยกระแสพัลส์ที่สอดคล้องกัน I_1, I_4, I_3 และ I_2 (กระแสแต่ละเฟสในทิศทางที่แสดงไดอะแกรม)แต่ละสเตป โรเตอร์จะหยุดหมุนไปตามทิศทางเข็มนาฬิกา 45 องศา (360/8)

เมื่อขั้วเหนือของโรเตอร์ (แม่เหล็กถาวร) หมุนไปถึงขั้วของสเตเตอร์หมายเลข 2 ลำดับการขับขดลวดเฟสของ Stepping Motor คือ 1-4-3-2 จะต้องกระทำเหมือนเดิม (เพื่อให้มอเตอร์ไปตามเข็มนาฬิกา 180 องศา) ยกเว้นเราต้องการให้หมุนกลับทิศทางใน 180 องศาที่เหลือด้วยการป้อนกระแสกลับทิศทาง เพื่อให้เกิดการเหนี่ยวนำเป็นขั้วใต้ที่ขั้วสเตเตอร์ 1,4,3,2 ตามลำดับ (ทิศทางของกระแสแสดงในรูปที่ 3.9)

3.3.3 Hybrid Stepping Motor

มอเตอร์ชนิดนี้เป็นแบบรวมระหว่าง PM (Permanent Magnet) กับ VR (Variable Reluctance) คือชุดโรเตอร์จะเป็น PM ส่วนชุดสเตเตอร์จะเป็น VR

ไฮบริด Stepping Motor (HSM) แสดงดังรูปที่ 3.10 แสดงถึงโครงสร้างของ HSM ประกอบด้วย 2 ตอนกับแถบแม่เหล็กอยู่ระหว่าง 2 ตอน แต่ละตอนประกอบด้วยซี่ฟันของโรเตอร์และโพลของสเตเตอร์ที่มีซี่ฟันเช่นกันและพันด้วยขดลวด รายละเอียดโครงสร้างของสเตเตอร์และโรเตอร์แต่ละตอนแสดงในไดอะแกรมรูปที่ 3.10



รูปที่ 3.10 โครงสร้างของไฮบริดสเตเตอร์ N_R = 30, N_s = 24 ซี่ฟันของสเตเตอร์ทั้ง 2 ตอนจะอยู่ในแนวเดียวกับส่วนซี่ฟันของโรเตอร์ทั้ง 2 ตัว จะมีตำแหน่งต่างกัน $(1/2)P_R$ = 6 องศา , θ = 3 องศา

ลักษณะโครงสร้างของไฮบริด Stepping Motor

- จำนวนซี่ฟันของโรเตอร์และสเตเตอร์ไม่เท่ากัน
- ตอน A และตอน B มีโครงสร้างเหมือนกัน
- ซี่ฟันของสเตเตอร์ทั้ง 2 ตอน จะอยู่ในแนวเดียวกันอย่างถูกต้อง
- ส่วนซี่ฟันทั้ง 2 ของโรเตอร์ทั้ง 2 ตอน จะมีตำแหน่งที่แตกต่างกัน $(1/2)P_R$ (ในรูปที่ 2.10 กำหนดให้ P_R = 360/30 = 12 องศา ดังนั้นตำแหน่งซี่ฟันของโรเตอร์ทั้ง 2 ตอนจะแตกต่างกัน 6 องศา)
- สเตเตอร์ของแต่ละตอนมี 8 โพล แบ่งออกเป็น 2 สเตเตอร์เฟส
- เฟสที่ 1 จะพันขดลวดบนสเตเตอร์โพลหมายเลข 1,3,5,7 ของทั้งในตอน A และตอน B
- แกนแม่เหล็กถาวรจะเหนี่ยวนำโรเตอร์ในตอน A ให้เป็นแม่เหล็กขั้วเหนือและโรเตอร์ในตอน B ให้เป็นแม่เหล็กขั้วใต้ ความซับซ้อนจะเพิ่มมากขึ้นเนื่องจากการแบ่งส่วนของขดลวดเฟสใน 2 ตอน ทำให้ได้วงจรแม่เหล็กที่ซับซ้อนและได้ทางเดินของเส้นแรงแม่เหล็กที่ แตกต่างกันเป็นวงกลม ทิศทางเดินของสนามของสนามแม่เหล็กของสเตเตอร์โพลจะขึ้นอยู่กับทิศทางการไหลของกระแสเฟสดังแสดงด้วยลูกศรในรูปที่ 3.10

การทำงานของไฮบริด Stepping Motor

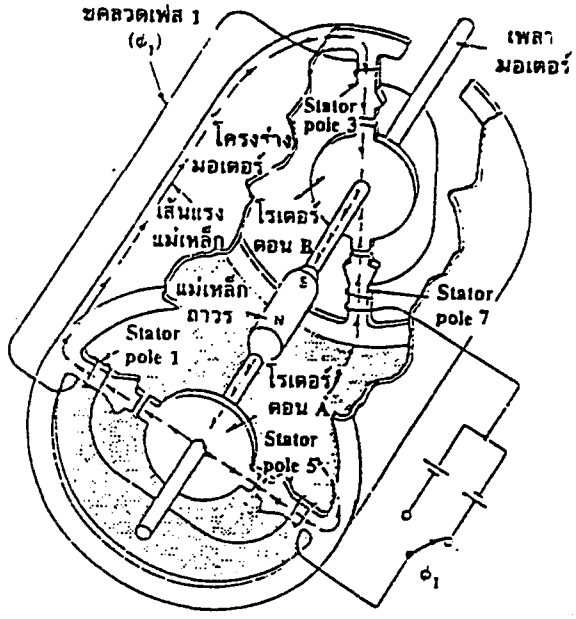
ขณะที่เฟสที่ 1 (ϕ_1) ได้รับพลังงานโดยการป้อนกระแส I_1 ในทิศทางดังแสดงด้วยลูกศรซี่ฟันของโรเตอร์ในตอน A จะอยู่ในแนวเดียวกับซี่ฟันของสเตเตอร์ของโพลที่ 1 และโพลที่ 5 ส่วนของตอน B จะอยู่ในแนวเดียวกับซี่ฟันของโพลที่ 3 และโพลที่ 7 ดังแสดงในรูปที่ 3.11

เพื่อให้เพลาของมอเตอร์หมุนไป 1 สเตปในทิศทางตามเข็มนาฬิกา เราจะต้องหยุดป้อนกระแส I_1 และป้อนกระแส I_2 ให้กับเฟสที่ 2 (ϕ_2)

ในรูปที่ 3.10 ซี่ฟันของโรเตอร์ที่เป็นสีดำใช้สำหรับอ้างอิง ซี่ฟันสีดำจะอยู่ใกล้แนวซี่ฟันของสเตเตอร์ของโพลที่ 4 และโพลที่ 8 ในตอน A และโพลที่ 6 ในตอน B มากที่สุด(ซี่ฟันของโรเตอร์ที่เป็นสีดำอยู่ห่างซี่ฟันของสเตเตอร์ = 1 สเตปพอดี)

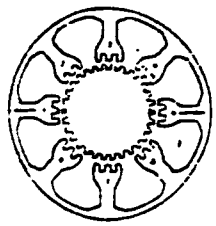
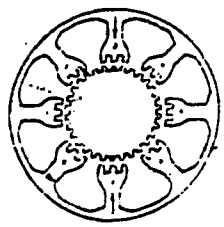
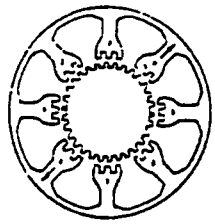
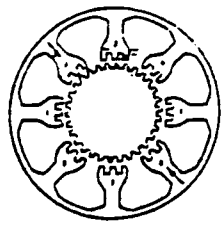
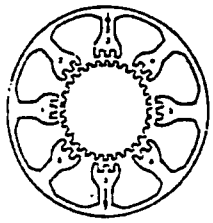
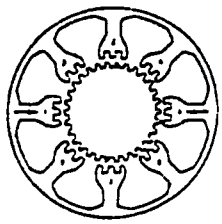
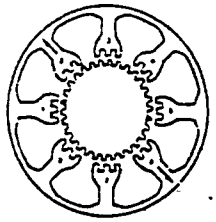
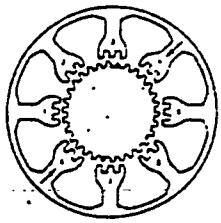
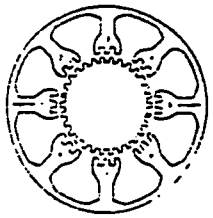
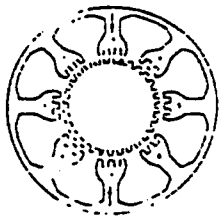
เราจะต้องป้อนกระแส I_2 ในทิศทางที่ถูกต้องคือจะต้องให้โพลที่ 4 โพลที่ 8

และโพลที่ 6 ถูกเหนี่ยวนำเป็นแม่เหล็กในทิศทางที่ถูกต้อง(เกิดวงจรแม่เหล็กที่สมบูรณ์)ด้วย I_2 ในกรณีนี้ I_2 จะต้องเป็นลบ



รูปที่ 3.11 วงจรแม่เหล็กของ HSM แสดงถึงเส้นทางเดินทางของเส้นแรงแม่เหล็ก เมื่อเฟสที่ 1 ได้รับพลังงานและเส้นแรงแม่เหล็กเกิดขึ้นในคอน A จะผ่านโพลที่ 1 และโพลที่ 5 เข้าไปยังโรเตอร์ของคอน B ผ่านโพลที่ 3 เข้าไปยังโรเตอร์ของคอน B ผ่านโพลที่ 3 และ โพลที่ 7 เข้าสู่ขั้วใต้ (s) ของแม่เหล็กถาวร

ในรูปที่ 3.12 แสดงถึงลำดับการสวิทช์ (การให้กระแสโพล) ให้มอเตอร์หมุนไปในทิศทางตามเข็มนาฬิกา 4 สเตป ซึ่งแสดงถึงตำแหน่งของโรเตอร์และทิศทางการเป็นแม่เหล็กของสเตเตอร์โพลในแต่ละตอนด้วยการกำหนดทิศทางการไหลของกระแสเฟส สำหรับการหมุนในทิศทางตามเข็มนาฬิกา (ดังแสดงในรูป) เราจะต้องกำหนดลำดับของกระแสเฟสดังนี้ $1^+, 2^-, 2^-$ และ 1^+ ตามลำดับ ถ้าเราต้องการหมุนในทิศทางทวนเข็มนาฬิกา ลำดับเหล่านี้ก็กลับไปเป็น $1^+, 2^+, 1^-, 2^-$ และ 1^+

Step	σ_1 I_1	σ_2 I_2	Flux out sec. A pole nos.	Flux in sec B pole nos	Section A	Section B
1	−		1, 5	3, 7		
2		−	4, 8	2, 6		
3	−		3, 7	1, 5		
4		+	2, 6	4, 8		
1	+		1, 5	3, 7		

รูปที่ 3.12 ลำดับ 4 สเตปของ HSM แบบ 2 เฟส ในแต่ละสเตปแสดงถึงตำแหน่งของโรเตอร์และทิศทางของเส้นแรงแม่เหล็ก $N_p = 30, N_r = 24, \theta_s = 3$ องศา ชี้นของโรเตอร์ที่เป็นสีดำจะหมุนในทิศทางตามเข็มนาฬิกา ไป 3 องศา ในแต่ละสเตปได้เป็น 12 องศา เมื่อครบตามจำนวนลำดับ (หนึ่งช่องระหว่างชี้ของโรเตอร์) สำหรับการหมุนในทิศทางตามเข็มนาฬิกาจะต้องจัดลำดับการนับเป็น 1^+ , 2^- , 1^- , 2^+ , 1^+

เพลาของมอเตอร์หมุนไปได้ 1 ช่องห่างระหว่างซี่ฟันภายใน 4 สเตป ดังนั้นมุมสเตปจะต้องเท่ากับ $(1/4)P_r$ หรือมีค่า เท่ากับ $\left| P_s - P_r \right|$ ดังนั้น

$$\theta_s = P_r/4 = 360/4N_r = 90/N_r$$

$$\theta_s = \left| P_s - P_r \right|$$

ในรูปที่ 2.12 $N_r = 30$ และ $N_s = 24$

ดังนั้น $\theta_s = 90/30 = 3$ องศา

$= 360/24 - 360/30 = 3$ องศา

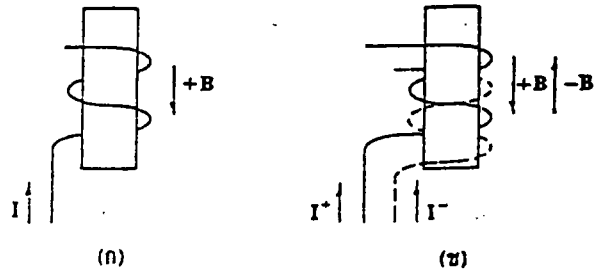
ไฮสบริด Stepping Motor (HSM) จะทำงานด้วยกระแสเฟสที่มีการโพลได้ 2 ทิศทางดังนั้นเราจำเป็นจะต้องใช้ Power Supply 2 ตัว (Bipolar Drive)

การแก้ปัญหาด้วยการขับไฮบริด Stepping Motor ให้ทำงานด้วย Power Supply เพียงตัวเดียว (Unipolar Drive) ได้โดยดัดแปลงโครงสร้างการพันขดลวดเฟสของสเตเตอร์

การพันขดลวดเฟสของสเตเตอร์แบบ Bifilar (การพันแบบ 2 แถวสลับกัน) สามารถขับได้ด้วย Unipolar Drive

ขดลวดแบบ Unifilar เราต้องกลับทิศทางของกระแสเพื่อกลับทิศทางของเส้นแรงแม่เหล็ก B

ขดลวดแบบ Bifilar ถ้าเราต้องกลับทิศทางของเส้นแรงแม่เหล็กเป็น -B เราสามารถทำได้โดยป้อนกระแสขนาดเดิมจาก Power Supply ตัวเดิมเข้าที่ขดลวดที่เป็นเส้นปะ จะทำให้ทางการเหนี่ยวนำแม่เหล็กและทิศทางของเส้นแรงแม่เหล็ก (-B) กลับทิศทางได้

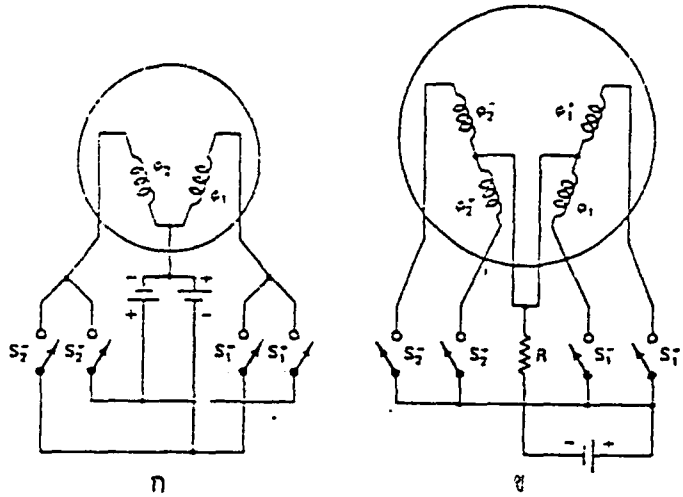


รูปที่ 3.13 การพันขดลวดเฟสของสเตเตอร์

ก) แบบ unifilar ข) แบบ bifilar

ถ้าหาก HSM ในรูปที่ 3.10 มีขดลวดเฟสของสเตเตอร์เป็นแบบ Bifilar ขดลวดเฟส ϕ_1 เดิมจะถูกแบ่งตัวออกเป็น 2 ขดลวดเฟส ϕ_1^+ และเฟส ϕ_1^-

ในตอนนี้ก็จะทำให้เราได้ขดลวดเฟสถึง 4 เฟส และแต่ละเฟสสามารถขับได้ด้วยกระแสที่ไหลในทิศทางเดียว ส่วนเครื่องหมาย + และ - ใช้สำหรับแสดงถึงทิศทางการเกิดสนามแม่เหล็กของสเตเตอร์โพล



ก

ข

Step	S_1^-	S_1^-	S_2^-	S_2^-
1	X			
2				X
3		X		
4			X	
1	X			

ก

Step	S_1^-	S_1^-	S_2^-	S_2^-
1	X			X
2		X		X
3		X	X	
4	X		X	
1	X			X

ง

Step	S_1^-	S_1^-	S_2^-	S_2^-
1	X			X
2				X
3		X		X
4		X		
5		X	X	
6			X	
7	X		X	
8	X			
1	X			X

จ

ในรูปที่ 3.14 (ก) ,(ข) แสดงวงจรการสวิทช์ 2 วงจร สำหรับ HSM แบบ 2 เฟสและแบบ 4 เฟส รูป (ค),(ง) และ (จ) แสดงตารางลำดับการขับแบบทีละเฟสและขับทีละ 2 เฟส และการขับแบบครึ่งเฟสตามลำดับ

3.4 การกระตุ้น (Stepping Motor Excitation)

แบบที่นิยมใช้มีอยู่ 3 แบบ คือ

3.4.1 กระตุ้นเฟสเดียว (one phase exitation)

3.4.2 กระตุ้นสองเฟส (two phase exitation)

3.4.3 กระตุ้นครึ่งสเตป (half step exitation)

สเตป / เฟส	1	2	3	4	1
ϕ_1	0	0	0	1	0
ϕ_2	1	0	0	0	1
ϕ_3	0	1	0	0	0
ϕ_4	0	0	0	0	0
ทวนเข็มฯ	→				→
ตามเข็ม	←				←

ก

สเตป / เฟส	1	2	3	4	1
ϕ_1	1	0	0	1	1
ϕ_2	1	1	0	0	1
ϕ_3	0	1	1	0	0
ϕ_4	0	0	1	1	0
ทวนเข็มฯ					
ตามเข็มฯ					

ข

สเตป / เฟส	1	2	3	4	5	6	7	8	1
ϕ_1	1	1	0	0	0	0	0	1	1
ϕ_2	0	1	1	1	0	0	0	0	0
ϕ_3	0	0	0	1	1	1	0	0	0
ϕ_4	0	0	0	0	0	1	1	1	0
ทวนเข็มฯ	→								→
ตามเข็มฯ	←								←

ค

รูปที่ 3.15 แสดงแผนภูมิเวลาของการกระตุ้นแบบต่างๆ

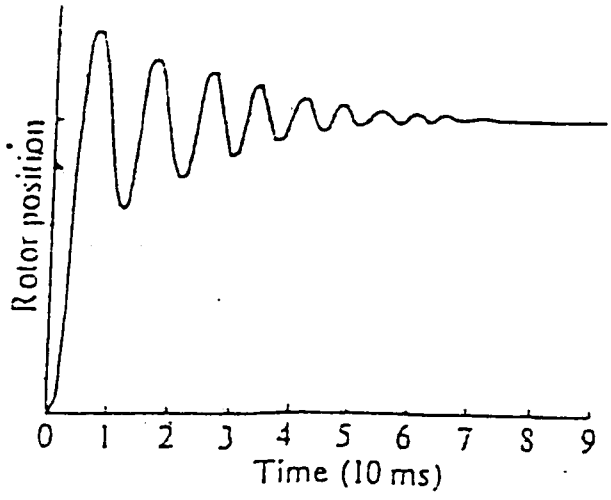
ก) การกระตุ้นเฟสเดียว

ข) การกระตุ้นสองเฟส

ค) การกระตุ้นครึ่งสเตป

3.4.1 การกระตุ้นเฟสเดียว

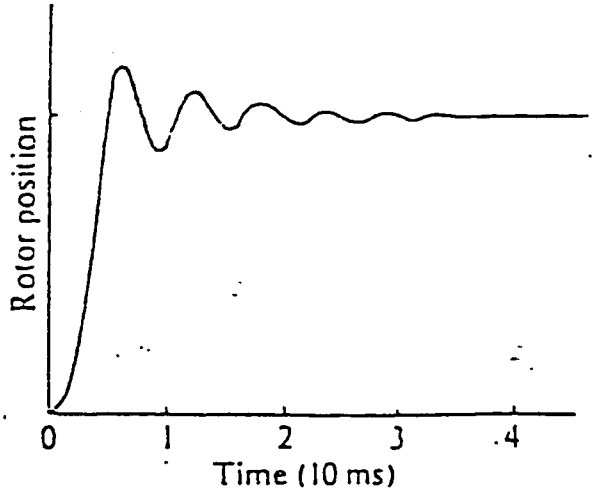
แบบนี้จะมีแรงบิดน้อยที่ทั้งในขณะเริ่มเคลื่อนที่และเคลื่อนที่อยู่ มี over shoot สูงเข้าสู่ตำแหน่งแต่ละสเตปช้าแต่เข้าสู่สภาวะ steady state แล้วจะได้ไม่มีการ oscillate



รูปที่ 3.16 กราฟแสดงผลตอบสนองของ stepping motor ต่อการกระตุ้นเฟสเดียว

3.4.2 การกระตุ้นสองเฟส

การกระตุ้นแบบนี้มีแรงบิดขณะเริ่มต้นสูง มี over shoot ต่ำและมี oscillate เล็กน้อยในสภาวะ steady state



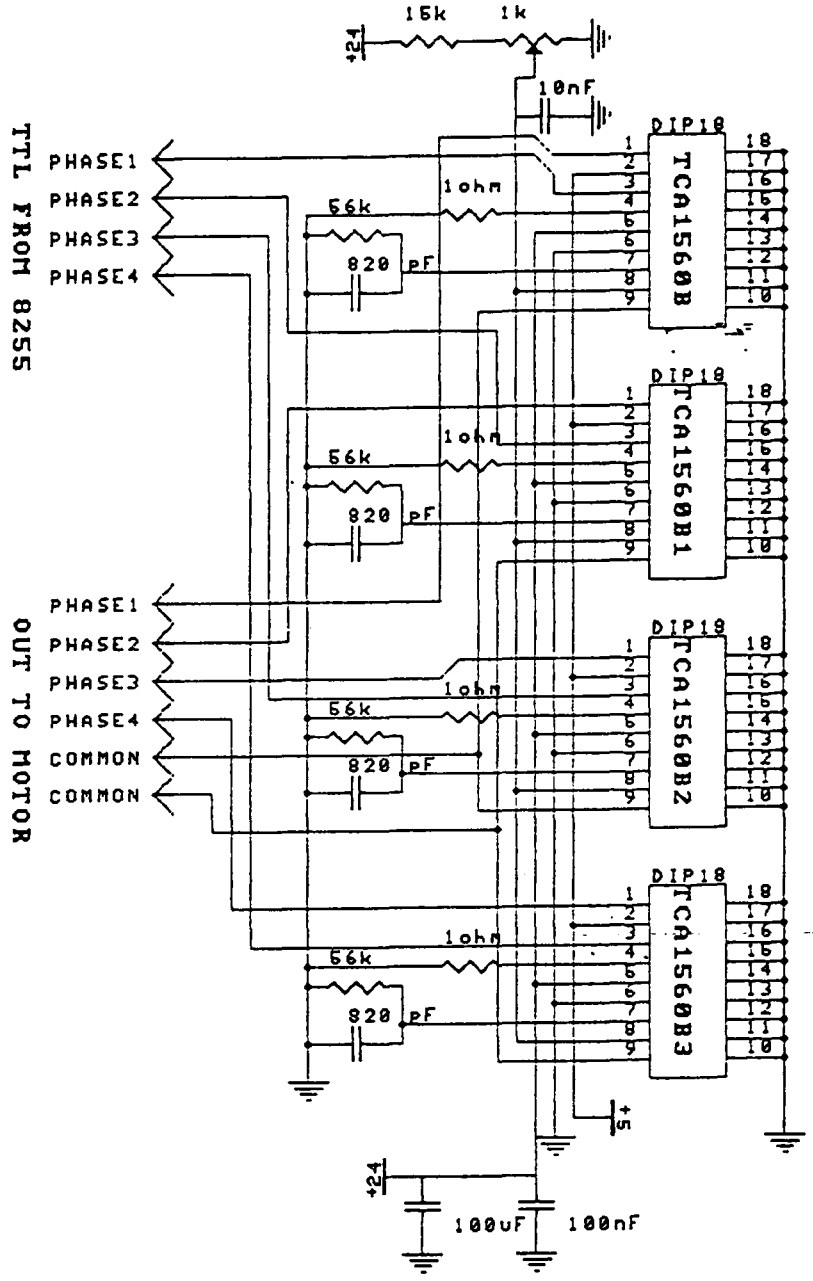
รูปที่ 3.17 กราฟแสดงผลตอบสนองของ stepping motor ต่อการกระตุ้นสองเฟส

3.4.3 การกระตุ้นครึ่งสเตป

สำหรับ Stepping Motor ที่ใช้การกระตุ้นแบบครึ่งสเตปจะทำให้มุมองศาในแต่ละ สเตปจะมีค่าลดลงจากค่าที่กำหนดไว้ครึ่งหนึ่งจากค่ามุมองศาที่กำหนดไว้ที่ Data ของ Stepping Motor ตัวนั้น การกระตุ้นแบบนี้จะรวมข้อดีของทั้งสองแบบไว้โดยขณะเริ่มต้นจะกระตุ้นแบบสองเฟส ทำให้ได้แรงบิดสูงที่สุดและเมื่อเข้าสู่สภาวะ steady state จะกระตุ้นเพียงเฟสเดียว ซึ่งจะทำให้ไม่เกิดการ oscillate แต่ข้อเสียของการกระตุ้นแบบนี้จะมีลักษณะเช่นเดียวกับการกระตุ้นแบบเฟสเดียว กล่าวคือในขณะ steady state ค่าแรงบิดจะมีค่าน้อย

3.5 วงจรขับมอเตอร์

จากที่ได้กล่าวมาแล้วการที่จะทำการขับ Stepping Motor นั้น จะต้องใช้การกระตุ้นโดยพัลส์หรือจากวงจรจัดลำดับลอจิก (Sequencer Logic) เมื่อได้เอาท์พุทจากวงจรจัดลำดับออกมาก็นำมาทำการต่อกับวงจรขับกำลัง (Power Drive) โดยเอาท์พุทที่ออกมาจะทำการเปิดปิดวงจรขับกำลังเพื่อไปขับ Stepping Motor ซึ่งวงจรขับกำลังนั้นจะเรียกว่าวงจรขับมอเตอร์



รูปที่ 3.18 วงจรขับมอเตอร์

รายละเอียดเกี่ยวกับวงจร

OUTPUT

เอาท์พุท Q_1,Q_2 (ขา 19) ถูกป้อนโดยภาคเอาท์พุท PUSH PULL FREE WHEEL ไดโอด 2 ตัวจะถูกต่อกับกราวด์และแหล่งจ่ายไฟตามลำดับ ซึ่งมันจะป้องกัน IC จากความต่างศักย์ FEED BACK โดยโหลดเหนี่ยวนำ

ENABLE

เอาท์พุท Q_1,Q_2 นั้นจะปิดลงเมื่อความต่างศักย์ V_{13} น้อยกว่าหรือเท่ากับ 0.8 โวลท์ ถูกป้อนเข้ามา 3 กระแสแหล่งจ่ายไฟตรงจะลดลงอย่างมาก ลงมามีค่าประมาณ 1 mA และกรณีเดียวกันนี้ก็อาจจะเกิดขึ้นซ้ำกันถ้าหากว่า ขา 3 นั้นปิด

Sink Transistor จะเปิดเมื่อ V_{13} มากกว่าหรือเท่ากับ 2 โวลท์

PHASE

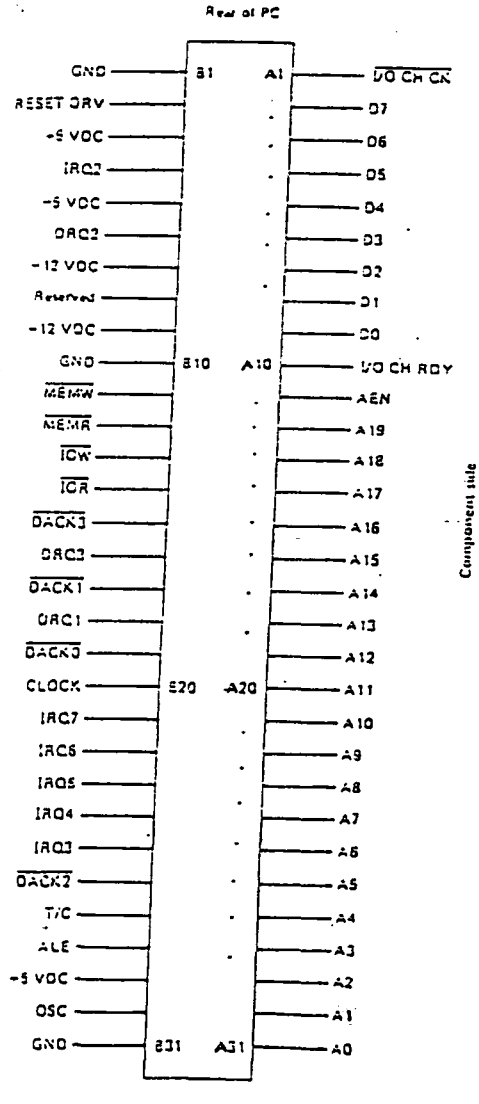
ความต่างศักย์ที่ขา 2 จะเป็นตัวกำหนดตำแหน่งเฟสของกระแสเอาท์พุท เอาท์พุท a_1 นั้นจะประพฤติตัวเป็น Sink สำหรับกรณี V_{12} น้อยกว่าหรือเท่ากับ 0.8 โวลท์ และประพฤติตัวเป็นแหล่งกำเนิดเมื่อ V_{12} มากกว่าหรือเท่ากับ 2 โวลท์

คล้าย ๆ กับเอาท์พุท Q ประพฤติตัวเป็น Sink เมื่อ V_{12} มากกว่าหรือเท่ากับ 2 โวลท์และแหล่งกำเนิดเมื่อ V_{12} น้อยกว่าหรือเท่ากับ 0.8 โวลท์ Sink Transistor เป็นต่อ Current-chopped

บทที่ 4

การเชื่อมต่อ IBM/PC

ภายใน IBM/PC ได้มีการออกแบบให้สามารถที่จะเพิ่มเติมวงจรอินเทอร์เฟสเข้าไปในภายหลังได้ โดยผ่านทางสล็อตที่อยู่บนเมนบอร์ด (Main Board) สำหรับสล็อตบนเมนบอร์ดนี้จะมีจำนวน 5 สล็อต (สำหรับใน IBM PC/XT จะมี 8 สล็อตซึ่งจะกล่าวถึงภายหลัง) ซึ่งแต่ละสล็อตมีจำนวนขาทั้งสิ้น 62 ขา แบ่งออกเป็น 2 ข้าง ข้างละ 31 ขา ส่วนการเรียกตำแหน่งขาของสล็อตเหล่านี้จะขึ้นอยู่กับว่าขานั้นอยู่ข้างใด (ซ้ายหรือขวา)



รูปที่ 4.1 ระบบบัสของ IBM/PC

ของสล็อต โดยขาที่อยู่ทางด้านซ้ายของสล็อตจะเรียกโดยใช้อักษร "B" นำหน้าเลขตำแหน่งของขา เช่น ขา B16 ก็คือขาทางด้านซ้ายของสล็อตขาที่ 16 (นับจากทางด้านท้ายของเครื่อง) ส่วนขาที่อยู่ทางด้านขวาของสล็อตจะเรียกโดยใช้อักษร "A" นำหน้าเลขตำแหน่งของขา เช่น ขา A24 ก็คือขาทางด้านขวาของสล็อตขาที่ 24 (นับจากทางด้านท้ายของเครื่อง)

แต่ละขาของสล็อตเหล่านี้จะเชื่อมต่อกับเส้นสัญญาณต่าง ๆ บนเมนบอร์ด ทำให้การสร้างวงจรอินเทอร์เฟสกับ IBM/PC สามารถทำได้โดยสะดวก ซึ่งเส้นสัญญาณที่เชื่อมต่อกับขาของสล็อตเหล่านี้จะประกอบไปด้วย เส้นสัญญาณของบัสแอดเดรส (Address Bus) บัสข้อมูล (Data Bus) บัสควบคุมสำหรับการเขียน/อ่าน ข้อมูลจากหน่วยความจำ หรือพอร์ท I/O เส้นสัญญาณสำหรับการขออินเทอร์รัพท์ของวงจรอินเทอร์เฟส เส้นสัญญาณสำหรับการขอ DMA สัญญาณฐานเวลา (Timing Signal) ต่าง ๆ ที่ใช้ในระบบ ,เส้นสัญญาณแสดงการรีเฟรชหน่วยความจำ และสัญญาณสำหรับการตรวจสอบความผิดพลาด (I/O CHCK)

นอกจากเส้นสัญญาณเหล่านี้แล้ว สล็อตบนเมนบอร์ดยังเชื่อมต่อกับแหล่งจ่ายไฟต่าง ๆ ที่ใช้ในระบบอีกด้วย คือ +5 Vdc, -5 Vdc, +12 Vdc และ -12 Vdc

4.1 รายละเอียดเกี่ยวกับสัญญาณต่าง ๆ

OSC (Oscillator; ขา B30) :

ขานี้เป็นขาเอาท์พุทที่เชื่อมต่อกับสัญญาณคล็อก ที่มีค่าความถี่สูงสุดบนเมนบอร์ดคือ 14.31818 Mhz ซึ่งมีคาบเวลาประมาณ 70 nS และมี Duty Cycle (ช่วงเวลาใน 1 คาบ ที่สัญญาณคล็อกมีลอจิกเป็น "1" หารด้วยคาบเวลาทั้งหมด) ประมาณ 50% สัญญาณคล็อกอื่น ๆ ของระบบ เช่น คล็อกที่ป้อนให้กับ 8088 หรือ ชิพชัพพอร์ทต่าง ๆ นั้นจะถูกสร้างขึ้นโดยการหารสัญญาณคล็อกนี้ อย่างไรก็ตามสิ่งหนึ่งที่จะต้องคำนึงถึงในการใช้งานสัญญาณ OSC ก็คือ สัญญาณนี้จะไม่ Synchronize กับสัญญาณอื่น ๆ บนบัสของระบบ ดังนั้นจึงไม่ควรที่จะนำสัญญาณจากขา OSC นี้ไปใช้เป็นสัญญาณคล็อกสำหรับวงจรภายนอกอื่นที่ทำงานร่วมกับระบบ

CLK (Clock; ขา B20) :

ขาสัญญาณนี้เป็นเอาท์พุท ซึ่งต่อกับสัญญาณคล็อกที่ถูกสร้างขึ้นโดยการหารสัญญาณ OSC ด้วย 3 ทำให้ได้ความถี่ประมาณ 4.77 MHz (14.31818 MHz/3) หรือ มีช่วงเวลาใน 1 คาบ (ช่วงเวลาของคล็อก 1 ลูก) เท่ากับ 210 nS (1/4.77 MHz) สำหรับค่า Duty Cycle ของสัญญาณนี้จะมีค่าประมาณ 1/3 คือ ใน 1 คาบจะมีช่วงเวลาที่เป็นลอจิก "1" เท่ากับ 1/3 ของคาบเวลาทั้งหมด หรือประมาณ 70 nS และช่วงเวลาลอจิก "0" เท่ากับ 2/3 ของคาบเวลาทั้งหมด หรือประมาณ 140 nS สัญญาณนี้เป็นสัญญาณที่ถูกใช้เป็นคล็อกของระบบ

RESET DRV (ขา B2) :

ขาสัญญาณนี้เป็นเอาท์พุท ซึ่งจะแอคทีฟ ลอจิก "1") ในช่วงที่เราเริ่มจ่ายไฟให้กับระบบ และจะยังคงแอคทีฟไปจนกว่าระบบต่าง ๆ ภายใน IBM/PC จะพร้อมที่จะทำงาน ได้จากนั้นสัญญาณนี้ก็จะเปลี่ยนกลับเป็นลอจิก "0" นอกจากนี้ในระหว่างการทำงานของ IBM/PC ถ้าระดับแรงดันของแหล่งจ่ายไฟตกลง สัญญาณนี้ก็จะถูกทำให้แอคทีฟเช่นกัน โดยทั่วไปแล้วสัญญาณนี้จะถูกนำไปใช้ในการรีเซ็ทวงจรอินเทอร์เฟสหรืออุปกรณ์ I/O ต่าง ๆ ในช่วงที่เริ่มจ่ายไฟให้กับระบบ ซึ่งจะเป็นการทำให้วงจรหรืออุปกรณ์เหล่านั้นถูกปรับให้อยู่ในสภาวะที่แน่นอน ก่อนที่จะเริ่มต้นการทำงานในระบบ (สภาวะนี้เป็นสภาวะที่เราทราบ และต้องการให้วงจรทำงานในขณะที่ระบบถูกรีเซ็ท)

A0 - A19 (Address Bus; ขา A13 - A12) :

ขาสัญญาณทั้ง 20 ขานี้เป็นเอาท์พุท ซึ่งใช้สำหรับกำหนดแอดเดรสของหน่วยความจำหรืออุปกรณ์ I/O ที่ 8088 ต้องการติดต่อด้วย โดยสัญญาณ A0 จะมีนัยสำคัญต่ำสุด (Least Significant Bit) และ A19 จะมีนัยสำคัญสูงสุด (Most Significant Bit) สำหรับค่าแอดเดรสบนบัสแอดเดรส A0 - A19 นี้ จะถูกกำหนดโดย 8088 ในระหว่างขบวนการอ่าน/เขียนข้อมูลลงในหน่วยความจำหรืออุปกรณ์ I/O แต่ในช่วงของขบวนการ DMA นั้น DMA-Controller จะเป็นผู้กำหนดค่าแอดเดรสบนบัสแอดเดรสเอง (ในระหว่างนี้ 8088 จะถูกตัดออกจากระบบ)

จะเห็นได้ว่าจำนวนเส้นแอดเดรสนี้จะมีอยู่ 20 เส้น ซึ่งสามารถที่จะอ้างแอดเดรสของหน่วยความจำได้ถึง 1 Mbyte แต่อย่างไรก็ตามจะมีแอดเดรสบางแอดเดรสที่ถูกใช้งานโดย IBM/PC อยู่ก่อนแล้ว คือ แอดเดรสของหน่วยความจำ RAM บนเมนบอร์ดที่ถูกใช้โดยระบบจำนวน 64 Kbyte (สำหรับ IBM PC/XT จะเป็นจำนวน 256 Kbyte) และแอดเดรสสำหรับหน่วยความจำ ROM อีก 48 Kbyte ซึ่งถูกจัดในช่วงของแอดเดรสบนสุดใน 1 Kbyte คือ 0FC00H จนถึง 0FFFFH (สำหรับ IBM PC/XT จะเป็น 64 Kbyte)

สำหรับการอ้างแอดเดรสของพอร์ท I/O นั้น จะใช้เส้นแอดเดรสเพียง 16 เส้น คือ A0 - A15 ซึ่งจะทำให้อ้างแอดเดรสของพอร์ทได้ 64 พอร์ท โดยผ่านทางชุดคำสั่ง IN และ OUT ส่วนเส้นแอดเดรสที่เหลือคือ A16 - A19 นั้นจะไม่ถูกใช้งาน อย่างไรก็ตามภายในและค่าแอดเดรสที่ใช้งานจะต้องอยู่ในช่วง 0200H จนถึง 03FFH เท่านั้น

D0 - D7 (Data Bus; ขา A9 - A2) :

ขาสัญญาณนี้จะเป็นแบบ Bi-Directional ซึ่งต่อกับบัสข้อมูลของระบบ เพื่อทำหน้าที่ในการส่งผ่านข้อมูลระหว่างพอร์ท I/O กับ IBM/PC โดยบิท D0 จะมีนัยสำคัญต่ำสุด และบิท D7 จะมีนัยสำคัญสูงสุด

สำหรับในบัสไซเคิลของการเขียนข้อมูลที่สร้างขึ้นโดย 8088 นั้น ข้อมูลจะถูกส่งออกมาบนข้อมูล ก่อนที่สัญญาณ IOW (ในกรณีที่ต้องการส่งข้อมูลให้กับพอร์ท) หรือ MEMW (ในกรณีที่ต้องการส่งข้อมูลให้กับหน่วยความจำ) จะเปลี่ยนจากลอจิก "0" เป็นลอจิก "1" (ขอบขาขึ้น) ซึ่งโดยทั่วไปขอบขาขึ้นของสัญญาณ IOW หรือ MEMW นี้ จะถูกใช้เพื่อสั่งให้พอร์ท I/O หรือหน่วยความจำที่มีแอดเดรสตรงกับค่าแอดเดรสบนบัสแอดเดรสนั้นรับข้อมูลไปเก็บไว้

สำหรับในบัสไซเคิลของการอ่านข้อมูลที่สร้างขึ้นโดย 8088 นั้น พอร์ท I/O หรือหน่วยความจำที่ถูกอ้างถึงจะต้องส่งข้อมูลออกมาบนบัสข้อมูล ก่อนที่สัญญาณ IOR (ในกรณีที่ต้องการอ่านข้อมูลจากพอร์ท) หรือ MEMR (ในกรณีที่ต้องการอ่านข้อมูลจากหน่วยความจำจะเปลี่ยนจากลอจิก "0" เป็นลอจิก "1" (ขอบขาขึ้น)

ALE (Address Latch Enable; ขา B28) :

ขาสัญญาณนี้เป็นสัญญาณเอาท์พุทที่ 8288 Bus Controller สร้างขึ้นเพื่อใช้สำหรับแสดงการเริ่มต้นของบัสไซเคิล และแสดงให้อุปกรณ์ภายนอกทราบว่าแอดเดรสที่ 8088 ต้องการจะติดต่อด้วยนั้นถูกส่งออกมาบนบัสแอดเดรสแล้ว โดยที่สัญญาณ ALE นี้จะเปลี่ยนจากลอจิก "1" เป็น "0" เมื่อค่าแอดเดรสที่ถูกต้องถูกส่งออกมาบนบัสข้อมูลเรียบร้อยแล้ว ดังนั้นขอบขาลงสัญญาณ ALE นี้จะถูกใช้ในการแลทช์ค่าแอดเดรสจากบัสแอดเดรส/ข้อมูล (Address/Data Bus; AD0 - AD7) ของ 8088 ทำให้สามารถแยกค่าแอดเดรส (A0 - A19) และข้อมูล (D0 - D7) ออกจากกันได้ อย่างไรก็ตามสัญญาณ ALE จะแอคทีฟเฉพาะในบัสไซเคิลที่สร้างขึ้นโดย 8088 เท่านั้น โดยจะไม่แอคทีฟในระหว่างขบวนการ DMA

I/O CHCK (I/O Channel Check; ขา A1) :

ขาสัญญาณนี้เป็นอินพุทที่ใช้ในการแสดงความผิดพลาดเกี่ยวกับพาริตี้ ที่เกิดขึ้นในการทำงานของวงจรอินเทอร์เฟสหรืออุปกรณ์ I/O เมื่อขาสัญญาณนี้ได้รับลอจิก "0" จะทำให้ 8088 ถูกอินเทอร์รัพท์แบบ Non-Maskable (NMI) อย่างไรก็ตามเราสามารถที่จะกำหนดให้วงจรภายใน IBM/PC ทำการขออินเทอร์รัพท์ (เมื่อได้รับสัญญาณ I/O CHCK) หรือไม่ก็ได้โดยการกำหนดลอจิกของบิทข้อมูลของพอร์ทที่ควบคุมการขออินเทอร์รัพท์แบบ NMI คือบิท D7 ของพอร์ท 00A0H ในกรณีที่บิท D7 ของพอร์ท 00A0H ถูกเซ็ทเป็น "1" ก็จะทำให้วงจรภายนอกของอินเทอร์รัพท์แบบ NMI ได้ (Enable) แต่ถ้าบิท D7 ของพอร์ท 00A0H ถูกเซ็ทเป็น "0" ก็จะเป็นการดิสเอเบิล (Disable) การขออินเทอร์รัพท์แบบ NMI ดังนี้

Enable : ใช้คำสั่ง OUT ส่งข้อมูล 80H ไปยังพอร์ท 00A0H

Disable : ใช้คำสั่ง OUT ส่งข้อมูล 00H ไปยังพอร์ท 00A0H

และเนื่องจากยังมีอุปกรณ์อื่นที่สามารถขออินเทอร์รัพท์แบบ NMI ได้อีก ดังนั้นซอฟท์แวร์ที่ใช้งานจะต้องสามารถตรวจสอบว่าการขออินเทอร์รัพท์นั้นเกิดขึ้นจากแหล่งใดได้ด้วย

I/O CHRDY (I/O Channel Ready ; ขา A10) :

ขาสัญญาณนี้เป็นอินพุทที่ใช้เพิ่มช่วงเวลาในบัสไซเคิลในกรณีที่อุปกรณ์ I/O หรือหน่วยความจำที่เกี่ยวข้องกับขบวนการในบัสไซเคิลที่เกิดขึ้นนั้น ไม่สามารถทำงานทันตามช่วงเวลาปกติของบัสไซเคิลนั้น ๆ ได้ (ช่วงเวลาของบัสไซเคิลที่เกี่ยวกับหน่วยความจำใช้ช่วงเวลาเท่ากับช่วงเวลาของคล็อก 5 ลูกหรือ 1.05 sec.)

เมื่ออุปกรณ์ I/O หรือหน่วยความจำต้องการที่จะเพิ่มช่วงเวลาในบัสไซเคิลให้นานขึ้นอีกนั้น จะสามารถทำได้โดยการป้อนลอจิก "0" ให้กับขา I/O CHRDY ในช่วงเวลาที่ I/O หรือ หน่วยความจำที่ถูกกำหนดนั้น ได้รับสัญญาณจากการดีโค้ดแอดเดรส และสัญญาณ MEMR, MEMW, IOR หรือ IOW แอคทีฟ สำหรับรายละเอียดเกี่ยวกับการเพิ่มช่วงเวลาในบัสไซเคิลนั้นจะกล่าวอีกครั้งในเรื่อง "การสร้าง Wait State"

IRQ2-IRQ7 (Interrupt Request 2 Through 7; ขา B4 และ B25-B21):

ขาสัญญาณทั้ง 5 นี้เป็นขาอินพุทที่ใช้สำหรับการอินเทอร์รัพท์จาก 8088 โดยสัญญาณเหล่านี้จะต่อเข้ากับ 8259A บนเมนบอร์ดโดยตรง โปรแกรมในส่วน BIOS ของ IBM/PC จะทำการโปรแกรม 8259A ให้ IRQ2 มีลำดับความสำคัญสูงสุด (Highest Priority) และ IRQ ขาใดขาหนึ่งถูกเปลี่ยนจากลอจิก "0" เป็นลอจิก "1" (ขอบขาขึ้น 8259A ก็จะทำการส่งสัญญาณ INT ให้กับ 8088 เพื่อทำการขออินเทอร์รัพท์

สิ่งสำคัญในการขออินเทอร์รัพท์โดยผ่านทาง IRQ2 - IRQ7 นี้ ก็คือ อุปกรณ์ที่ทำการขออินเทอร์รัพท์โดยผ่านทาง IRQ ขาใดก็จะต้องรักษาระดับสัญญาณที่ขา IRQ นั้นให้แอคทีฟ (ลอจิก "1") อยู่จนกว่าจะได้รับสัญญาณ INTA (Interrupt Acknowledge) จาก 8088 เสียก่อน ถ้าไม่เช่นนั้นการขออินเทอร์รัพท์จะถูกยกเลิก และอินเทอร์รัพท์ Level 7 (IRQ7) ก็จะถูกสร้างขึ้นโดยอัตโนมัติ ไม่ว่าการขออินเทอร์รัพท์ที่ถูกยกเลิกนั้นจะเป็นการขออินเทอร์รัพท์ใน Level หรือขาใด

แต่อย่างไรก็ตามสัญญาณ INTA นี้จะไม่ถูกต่อออกมาที่ขาของสล็อตด้วย ดังนั้นโปรแกรมที่ทำการตอบสนองต่อการขออินเทอร์รัพท์ (Interrupt Service Routine) จะต้องทำการรีเซ็ทสัญญาณ IRQ เอง โดยใช้คำสั่ง OUT ไปยังพอร์ท I/O ที่เกี่ยวข้อง (ดูรายละเอียดเกี่ยวกับการอินเทอร์รัพท์ในเรื่อง "การจัดอินเทอร์รัพท์ของระบบ"

IOR (I/O Read; ขา B14) :

ขาสัญญาณนี้เป็นเอาท์พุทแอคทีฟลอจิก "0" สร้างโดย 8288 Bus Controller เพื่อใช้ในการแสดงว่าบัสไซเคิลที่เกิดขึ้นนี้ เป็นบัสไซเคิลของการอ่านข้อมูลจากพอร์ท I/O เพื่อให้พอร์ท I/O ที่แอดเดรสตรงกับแอดเดรสบนบัสแอดเดรส ส่งข้อมูลออกมาบนบัสข้อมูล โดยข้อมูลจะต้องถูกส่งออกมาบนบัสข้อมูลก่อนขอบขาขึ้นของสัญญาณ IOR ประมาณ 30 nS เพื่อให้มั่นใจได้ว่า 8088 สามารถรับข้อมูลได้ถูกต้อง สำหรับในขบวนการ DMA 8237A-5 DMA ต้องการจะนำข้อมูลไปเก็บ การที่พอร์ทใดจะส่งข้อมูลออกมาบนบัสข้อมูลนั้น จะอาศัยสัญญาณ DACK จาก DMA Controller เป็นตัวกำหนด เช่นกรณีที่สัญญาณ DACK1 แอคทีฟ ก็จะแสดงว่าพอร์ท I/O ที่จะต้องส่งข้อมูลออกมาบนบัสข้อมูลก็คือพอร์ท I/O ที่ขอ DMA ผ่านทางแชนแนลที่ 1 (DTQ1) เป็นต้น

IOW (I/O Write; ขา B13) :

ขาสัญญาณนี้เป็นเอาท์พุทแอคทีฟลอจิก "0" สร้างโดย 8288 Bus Controller เพื่อใช้แสดงว่าบัสไซเคิลที่เกิดขึ้นนี้เป็นบัสไซเคิลของการเขียนข้อมูลลงบนพอร์ท I/O เพื่อให้พอร์ท I/O ที่มีแอดเดรสตรงกับแอดเดรสบนบัสแอดเดรสนั้น รับข้อมูลที่อยู่บัสข้อมูลไปเก็บไว้ อย่างไรก็ตามเนื่องจากในช่วงเวลาที่สัญญาณ IOW นี้แอคทีฟ (ลอจิก "0") นั้น ข้อมูลบนบัสข้อมูลอาจจะยังไม่สมบูรณ์ ดังนั้นในการออกแบบจึงควรใช้ขอบขาขึ้นของสัญญาณ IOW แทนขอบขาลงในการทำให้พอร์ท I/O ที่เกี่ยวข้องรับข้อมูลไปเก็บไว้ เพื่อให้ข้อมูลบนบัสสมบูรณ์เสียก่อน สำหรับในขบวนการ DMA-Controller จะทำการสร้างสัญญาณ IOW เอง โดยที่ค่าแอดเดรสที่อยู่บนบัสแอดเดรสจะเป็นค่าแอดเดรสของหน่วยความจำที่พอร์ท I/O ที่ขอ DMA ต้องการจะอ่านข้อมูล

MEMW (Memory Write; ขา B11) :

ขานี้เป็นเอาท์พุทแอคทีฟที่ลอจิก "0" ซึ่ง 8288 Bus Controller สร้างขึ้นในระหว่างบัสไซเคิลในการเขียนข้อมูลลงในหน่วยความจำของ 8088 สัญญาณ MEMW นี้จะถูกส่งออกมาเพื่อให้หน่วยความจำที่แอดเดรสตรงกับค่าแอดเดรสบนบัสแอดเดรสนั้น ทำการรับข้อมูลที่อยู่บนบัสข้อมูลไปเก็บไว้ โดยทั่วไปหน่วยความจำจะรับข้อมูลในช่วงขอบขาขึ้นสัญญาณ MEMW

สำหรับในระหว่างขบวนการ DMA นั้น 8237A-5 DMA-Controller จะทำการควบคุมบัสต่าง ๆ ของระบบแทน 8088 และสัญญาณ MEMW จะถูกใช้ในบัสไซเคิลของการเขียนข้อมูลลงในหน่วยความจำ (ข้อมูลถูกส่งจากหน่วยความจำไปให้กับอุปกรณ์ I/O)

DRQ1-DRQ3 (DMA ReQuest 1-3 ; ขา B18,B6 และ B16) :

ขาสัญญาณทั้งสามเป็นสัญญาณอินพุทแอคทีฟที่ลอจิก "1" ซึ่งอุปกรณ์ภายนอกสามารถใช้ในการขอ DMA จากระบบ โดยการป้อนระดับสัญญาณลอจิก "1" ให้กับขา DRQ ขาใดขาหนึ่ง (ขา DRQ ทั้งสามนี้จะต่อเข้ากับ DRQ1-DRQ3 ของ 8237A-5)

เมื่อ 8237A-5 ได้รับสัญญาณนี้แล้วก็จะตรวจสอบว่ามีการขอ DMA ในแชนแนลที่มีลำดับความสำคัญ (Priority) สูงกว่าหรือไม่ ถ้าไม่มีก็จะทำการขอ DMA จาก 8088 และตอบรับการขอ DMA จากอุปกรณ์ภายนอก (สัญญาณ DACK ของแชนแนลที่ขอ DMA จะแอคทีฟ) แต่ถ้ามี 8237A-5 ก็จะทำการขอ DMA ให้กับแชนแนลที่มีลำดับความสำคัญสูงกว่าก่อนแล้วจึงทำการขอ DMA ให้กับแชนแนลที่มีลำดับความสำคัญต่ำกว่า ภายใน ROM BIOS ของ IBM/PC จะโปรแกรม 8237A-5 ให้ DRQ1 มีลำดับความสำคัญสูงสุด และ DRQ3 มีลำดับความสำคัญต่ำสุด ดังนั้นถ้ามีการขอ DMA ของอุปกรณ์ภายนอกผ่านทางแชนแนลที่ (DRQ1) และแชนแนลที่ 2 (DRQ2) 8237A-5 ก็จะทำการขอ DMA ให้กับแชนแนลที่ 1 ก่อน จากนั้นเมื่อเสร็จจากขบวนการ DMA ให้กับแชนแนลที่ 2

อย่างไรก็ตาม 8237A-5 ยังมีแชนแนลสำหรับการขอ DMA อยู่อีก 1 แชลแนล คือแชลแนล 0 (DRQ0) ซึ่งในความเป็นจริงแล้วแชลแนลนี้จะมีลำดับความสำคัญที่สูงกว่าแชนแนลที่ 1 แต่จะไม่ถูกต่อออกมายังขาสล็อต เนื่องจาก IBM/PC จะใช้แชนแนลที่ 0 นี้ในการรีเฟรชหน่วยความจำที่เป็น Dynamic Ram

ในการขอ DMA นั้นสัญญาณ DRQ นี้ จะต้องแอคทีฟอยู่ในช่วงระยะเวลาหนึ่งเท่านั้น ถ้าสัญญาณนี้แอคทีฟอยู่นานเกินไป จะทำให้เกิดขบวนการ DMA ขึ้นมากกว่า 1 ขบวนการได้ สำหรับวงจรที่ขอ DMA โดยทั่วไปแล้วจะใช้สัญญาณตอบรับการขอ DMA หรือสัญญาณ DACK ของแชนแนลที่ขอ DMA นั้น ในการรีเซ็ทสัญญาณ DRQ เช่น อุปกรณ์ภายนอกที่ขอ DMA ผ่านทางแชนแนลที่ 1 (DRQ1) ก็จะคอยตรวจสอบการตอบรับการขอ DMA จากสัญญาณ DACK ของแชนแนลที่ 1 (DACK1) เมื่อได้รับสัญญาณจาก DACK1 แล้ว ก็จะรีเซ็ทสัญญาณ DRQ1 (เปลี่ยนจากลอจิก "1" เป็น "0")

DACK - DACK3 (DMA Acknowledge 0-3; ขา B19,B17,B26 และB15) :

สัญญาณทั้ง 4 นี้เป็นเอาท์พุทแอคทีฟที่ลอจิก "0" ซึ่ง 8237A-5 สร้างขึ้นเพื่อแสดงให้วงจรภายนอกที่ขอ DMA ทราบว่าการขอ DMA นั้นได้รับการตอบสนองแล้ว และ 8237A-5 จะเข้าสู่ขบวนการ DMA เพื่อให้การส่งผ่านข้อมูลระหว่างอุปกรณ์ I/O ที่ขอ DMA กับหน่วยความจำเกิดขึ้นได้โดยตรง (คือไม่ต้องผ่าน 8088) โดยสัญญาณ DACK นี้จะแอคทีฟในแชนแนลใดก็ขึ้นอยู่กับว่าขบวนการ DMA ที่เกิดขึ้นนั้น เป็นการตอบสนองต่อการขอ DMA ในแชนแนลใดเช่น ถ้าขบวนการ DMA ที่เกิดขึ้นนั้นเป็นการตอบสนองต่อการขอ DMA ในแชนแนลที่ 2 (DRQ2) สัญญาณ DACK2 ก็จะแอคทีฟ เป็นต้น

ดังที่ได้กล่าวแล้วว่าสัญญาณ DRQ0 นั้นจะไม่ถูกต่อออกมายังขาของสล็อต ดังนั้นวงจรอินเทอร์เฟสจึงไม่สามารถจะขอ DMA ผ่านทางแชนแนล 0 ได้ สัญญาณ DACK0 จะถูกต่อออกมายังสล็อตด้วย (ขา B19) ทั้งนี้ก็เพื่อที่จะแสดงให้วงจรอินเทอร์เฟสต่าง ๆ ทราบว่าขบวนการ DMA ที่เกิดขึ้นในช่วงเวลาที่ DACK0 แอคทีฟนั้น เป็นขบวนการที่ใช้สำหรับการรีเฟรชหน่วยความจำที่เป็น Dynamic RAM ที่วงจรอินเทอร์เฟสที่ใช้หน่วยความจำประเภทนี้สามารถนำไปใช้ในการรีเฟรช Dynamic RAM ที่อยู่ในวงจรได้

โดยที่การรีเฟรชหน่วยความจำนั้นจะเกิดขึ้นในทุก ๆ 15.12 sec หรือ ทุก ๆ 72 คล็อก ดังนั้นสัญญาณ DACK0 นี้ก็จะแอคทีฟในทุก ๆ 15.12 sec ด้วย

AEN (Address Enable ;ขา A11):

สัญญาณนี้เป็นเอาท์พุทที่ใช้ในการแสดงว่า บัสไซเคิลที่เกิดขึ้นในช่วงเวลาที่สัญญาณ AEN แอคทีฟ (ลอจิก "1")นั้น เป็นบัสไซเคิลของขบวนการ DMA

สำหรับเมนบอร์ดของ IBM/PC นั้น จะใช้สัญญาณนี้ในการดิสเอเบิล (Disable) 8288 Bus Controller และจะใช้ดิสเอเบิลพอร์ท I/O ต่าง ๆ ที่ไม่เกี่ยวข้องกับขบวนการ DMA ที่เกิดขึ้น จำเป็นต้องทำเช่นนี้ก็เพราะในระหว่างขบวนการ DMA นั้น 8237A-5 จะส่งแอดเดรสของหน่วยความจำออกมาบนบัสแอดเดรส และจะทำให้สัญญาณ IOR หรือ IOW แอคทีฟ ดังนั้นถ้าไม่ทำการดิสเอเบิลพอร์ท I/O ที่เกี่ยวข้องไว้ ก็อาจจะทำให้พอร์ท I/O ที่แอดเดรสตรงกับค่าแอดเดรสบนบัสแอดเดรส (ซึ่งเป็นแอดเดรสของหน่วยความจำ) นั้น ทำการอ่านหรือส่งข้อมูลออกมาบนบัสข้อมูลทำให้เกิดความผิดพลาดขึ้นได้

T/C (Terminal Count;ขา B27):

สัญญาณนี้ถูกสร้างขึ้นมาจากการนำเอาสัญญาณเอาท์พุทที่ขา EOP ของ 8237A-5 มากลับลอจิก (โดยใช้เกท Inverter) ทำให้สัญญาณ T/C นี้แอคทีฟ

สำหรับสัญญาณนี้จะแอคทีฟเมื่อจำนวนไบท์ในการส่งผ่านข้อมูลของขบวนการ DMA ในแชนแนลใดแชนแนลหนึ่ง ครบตามจำนวนที่กำหนดไว้ โดยทั่วไปแล้วสัญญาณที่จะถูกใช้ในการสิ้นสุดขบวนการ DMA ที่ทำการส่งผ่านข้อมูลเป็นบล็อก เนื่องจากสัญญาณนี้จะแอคทีฟโดยไม่แสดงว่าเป็นสัญญาณของแชนแนลใด เพื่อให้สามารถทราบได้ว่า สัญญาณ T/C นี้ผ่านเกท Interter แล้วนำไป OR กับสัญญาณ DACK เพื่อให้สามารถทราบได้ว่า สัญญาณ T/C ที่เกิดขึ้นนั้นเป็นสัญญาณของแชนแนลใด สำหรับในแชนแนลที่ 0 นั้น สัญญาณ T/C จะแอคทีฟในช่วงเวลาที่คงที่คือ ทุก ๆ 990.804 mS ซึ่งก็คือช่วงเวลาที่ใช้ในการรีเฟรชหน่วยความจำขนาด 64 Kbyte นั่นเอง

4.2 บัสของแหล่งจ่ายไฟของระบบ

+5 Vdc (ขา B3 และ B29):

ขาทั้งสองนี้ต่อกับแหล่งจ่ายไป DC +5 V ของระบบ โดยจะมีค่าความเที่ยงตรง (Recgulated) 5% คืออยู่ในช่วง +4.75 ถึง +5.25 Vdc

+12 Vdc (ขา B9):

ขานี้จะต่อกับแหล่งจ่ายไฟ DC +12 V ของระบบ โดยจะมีค่าความเที่ยงตรง (Recgulated) 5% คืออยู่ในช่วง +11.4 ถึง +12.6 Vdc

-5 Vdc (ขา B5):

ขานี้จะต่อกับแหล่งจ่ายไฟ DC -5 V ของระบบ โดยจะมีค่าความเที่ยงตรง (Recgulated) 10% คืออยู่ในช่วง -5.5 ถึง -4.5 Vdc

-12 Vdc (ขา B7):

ขานี้จะต่อกับแหล่งจ่ายไฟ DC -12 V ของระบบ โดยจะมีค่าความเที่ยงตรง (Recgulated) 10% คืออยู่ในช่วง -13.2 ถึง -10.8 Vdc

GND (ขา B1,B10 และ B31) :

ขาทั้งสามนี้จะต่อเข้ากับกราวด์ (Ground) ของระบบ

4.3 การจัดสล็อตบนสล็อตของ IBM PC/XT

สำหรับใน IBM PC/XT นั้นจะมีสล็อตสำหรับเชื่อมต่อกับวงจรภายนอกได้มากขึ้น คือใน IBM PC/XT จะทำการเพิ่มจำนวนสล็อตบนเมนบอร์ดขึ้นเป็น 8 สล็อต จากเดิมที่มีอยู่เพียง 5 สล็อตบน IBM/PC โดยการจัดสัญญาณต่าง ๆ ในทั้ง 8 สล็อตยังคงเหมือนกับใน IBM/PC เพียงแต่สัญญาณต่าง ๆ ที่จะถูกส่งออกมายังของสล็อตที่ 8 นั้น จะถูกต่อผ่านวงจรบัฟกระแส (Buffer) ก่อน และในสล็อตที่ 8 นี้ขา B8 จะถูกใช้งานด้วย โดย

จะถูกใช้เป็นขา CARD SLCTD (หรือ Card selected) ซึ่งขาสัญญาณนี้จะเป็นสัญญาณอินพุทจากวงจรภายนอกที่เสียบอยู่บนสล็อตที่ 8 เพื่อให้วงจรบนเมนบอร์ดทราบว่าคาร์ดที่อยู่บนสล็อตนี้ถูกเลือกใช้งานอยู่ซึ่งจะทำให้ Drive บนเมนบอร์ดทำการอ่านหรือส่งข้อมูลไปยังสล็อตที่ 8

4.4 การแบ่ง Address ของ IBM/PC

ใน IBM/PC มีการแบ่งสรรการจัด Addess ต่าง ๆ ไว้แน่นอน ซึ่งได้แสดงไว้ดังรูป 4.2 ส่วนที่เราจะนำมาใช้ในการ Decode address นั้น จะอยู่ในส่วนของ Prototype Card ซึ่งมี Adress อยู่ในช่วง 300H-31FH ทั้งหมด 32 พอร์ท Adress

Hex range	Usage	
000-00F	DMA chip 8237A-5	Assigned to system board components
020-021	Interrupt 8259A	
040-043	Timer 8253-5	
060-063	PPI 8255A-5	
080-083	DMA page registers	
0Ax	NMI mask register	
0Cx	Reserved	
0Ex	Reserved	
100-1FF	Not usable	
200-20F	Game control	Assigned to feature card ports
210-217	Expansion unit	
220-24F	Reserved	
278-27F	Reserved	
2F0-2F7	Reserved	
2F8-2FF	Asynchronous communications (2)	
300-31F	Prototype card	
320-32F	Fixed disk	
378-37F	Printer	
380-38C	SDLC communications	
380-389	Binary synchronous communications (2)	
3A0-3A9	Binary synchronous communications (1)	
3B0-3BF	IBM monochrome display/printer	
3C0-3CF	Reserved	
3D0-3DF	Color/graphics	
3E0-3F7	Reserved	
3F0-3F7	Diskette	
3F8-3FF	Asynchronous communications (1)	

รูปที่ 4.2 การแบ่ง Address ของ IBM/PC

4.5 บัสไซเคิลของระบบ

สิ่งสำคัญในการเชื่อมต่อกับไมโครคอมพิวเตอร์นั้น ต้องพิจารณาถึงช่วงเวลาของสัญญาณนาฬิกาของระบบใน IBM/PC จะใช้สัญญาณนาฬิกาป้อนให้กับ 8088 จำนวน 4 ลูก ความถี่ประมาณ 4.77 MHz หรือคาบเวลาสัญญาณนาฬิกา 1 ลูกประมาณ 210 nS ดังนั้น ใน 1 บัสไซเคิล ใช้เวลา 4 x 210 nS หรือ 840 nS อย่างไรก็ตาม กรณีที่การติดต่อกับอุปกรณ์ภายนอกเช่น I/O พอร์ท นั้นมีความเร็วในการทำงานต่ำ ดังนั้นใน IBM/PC จึงเพิ่มช่วงเวลาในบัสไซเคิลจากสัญญาณนาฬิกา 4 ลูก (840 nS) ใน 1 บัสไซเคิลเป็น 5 ลูก (1.05 S) ใน 1 บัสไซเคิล สำหรับสัญญาณนาฬิกาที่เพิ่มมานั้นจะเรียกว่า Tw และภาวะที่ 8088 หยุดรอ เพื่อให้อุปกรณ์ภายนอกรับหรือส่งข้อมูลได้ทันนั้นคือ Wait State

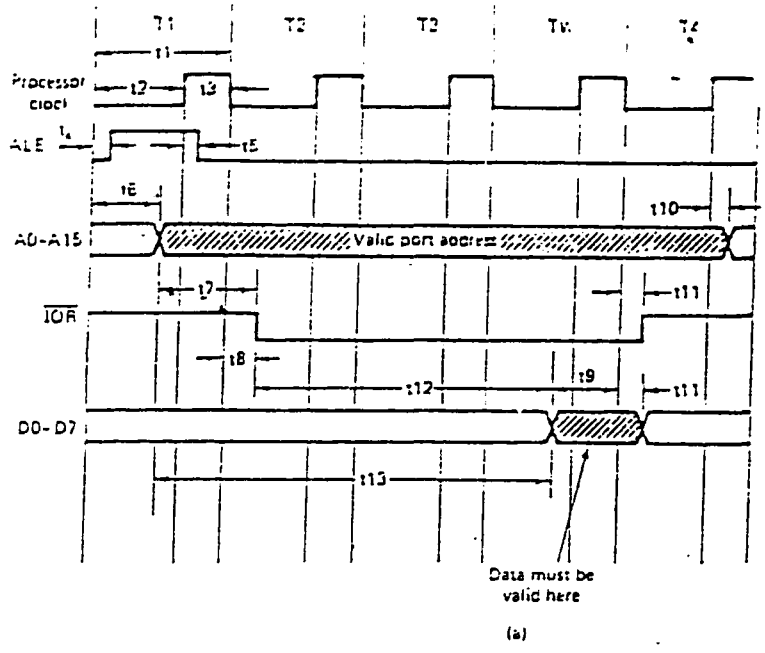
4.6 บัสไซเคิลในการอ่านและเขียนข้อมูลจากพอร์ท

เมื่อ 8088 ทำการ เอ็กซีคิวท์ (Execute) คำสั่ง IN ซึ่งเป็นชุดคำสั่งในการอ่านข้อมูลจากพอร์ท รูปที่ 4.3 ระหว่างช่วงเวลา T1 สัญญาณ ALE จะแอคทีฟ (ลอจิก 1) สัญญาณนี้จะถูกใช้เพื่อแสดงให้อุปกรณ์ที่ทำงานร่วมกับ 8088 รู้ว่าข้อมูลที่อยู่บนบัสแอดเดรสในช่วงขอบขาลงของสัญญาณ ALE นั้น เป็นแอดเดรสของพอร์ทที่ต้องการจะติดต่อด้วย (ในที่นี้ คือต้องการอ่านข้อมูล) ช่วงเวลา T2 สัญญาณ IOR จะแอคทีฟ (ลอจิก 0) เป็นการแสดงว่าบัสไซเคิลนี้เป็นบัสไซเคิลในการอ่านข้อมูลจากพอร์ท และเป็นการทำให้พอร์ทที่มีแอดเดรสตรงกับค่าแอดเดรสที่อยู่บนบัสแอดเดรสนั้นส่งข้อมูลออกมาบนบัสข้อมูล จากนั้นในช่วงเวลาเริ่มต้น T4 8088 ก็จะทำการอ่านข้อมูลนั้นเข้ามา จากนั้นสัญญาณ IOR จะปรับให้เป็นลอจิก 1 และ จะสิ้นสุดการทำงานในบัสไซเคิลเมื่อสิ้นสุดช่วงเวลาของ T4

จะเห็นว่าการเพิ่มบัสไซเคิลนี้อีก 1 ลูกคือช่าง Tw ซึ่งอยู่ในช่วงต่อระหว่าง T3 และ T4 เพื่อให้พอร์ท I/O ซึ่งมีความเร็วในการทำงานต่ำ สามารถที่จะส่งข้อมูลออกมาบนบัสข้อมูลได้ทัน

ในกรณีที่ 8088 ทำการเอ็กซีคิวท์ (Execute) ชุดคำสั่ง OUT รูป 4.3 ระหว่างช่วงเวลา T1 สัญญาณ ALE จะแอคทีฟ (ลอจิก 1) สัญญาณนี้จะถูกใช้เพื่อแสดงให้อุปกรณ์ที่ทำงานร่วมกับ 8088 รู้ว่าข้อมูลที่อยู่บนบัสแอดเดรสในช่วงขอบขาลงของสัญญาณ ALE นั้น เป็นแอดเดรสของพอร์ทที่ต้องการจะติดต่อด้วย (ในที่นี้ คือต้องการอ่านข้อมูล)

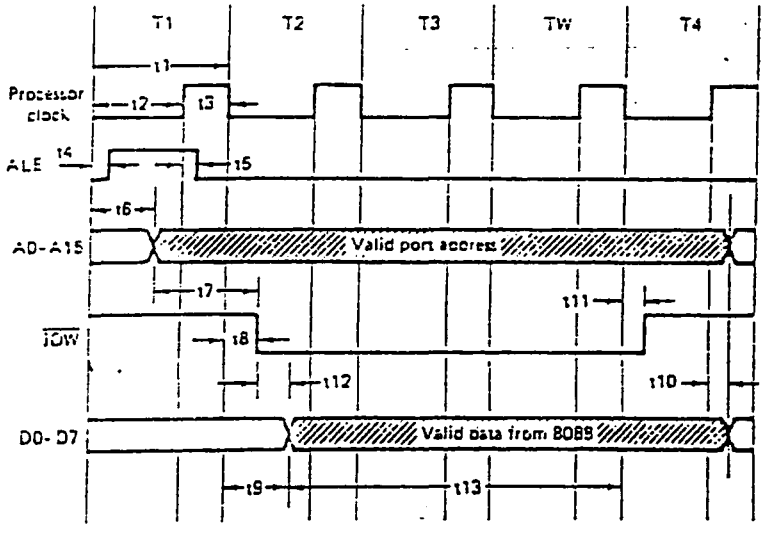
ช่วงเวลา T2 สัญญาณ IOW จะแอคทีฟ (ลอจิก 0) เป็นการแสดงว่าบัสไซเคิลนี้เป็นบัสไซเคิลในการอ่านข้อมูลจากพอร์ทและเป็นการทำให้พอร์ทที่มีแอดเดรสตรงกับค่าแอดเดรสที่อยู่บนบัสแอดเดรสนั้น ส่งข้อมูลออกมาบนบัสข้อมูล จากนั้นในช่วงเวลาเริ่มต้น T4 ก็จะทำการอ่านข้อมูลนั้นเข้ามา และสัญญาณ IOW จะปรับให้เป็นลอจิก 1 และสิ้นสุดการทำงานบัสไซเคิลเมื่อสิ้นสุดช่วงเวลาของ T4



Symbol	Max.	Min.
t1	–	209.5
t2	–	124.5
t3	–	71.5
t4	15	–
t5	15	–
t6	128	16
t7	–	91.5
t8	35	10
t9	–	42
t10	35	10
t11	–	10
t12	–	551.5
t13	–	668

*All times are in nanoseconds

(a)



Symbol	Max.	Min.
t1	–	209.5
t2	–	124.5
t3	–	71.8
t4	15	–
t5	15	–
t6	128	16
t7	–	91.5
t8	35	10
t9	122	14
t10	–	10
t11	35	10
t12	112	–
t13	–	506.5

*All times are in nanoseconds

(b)

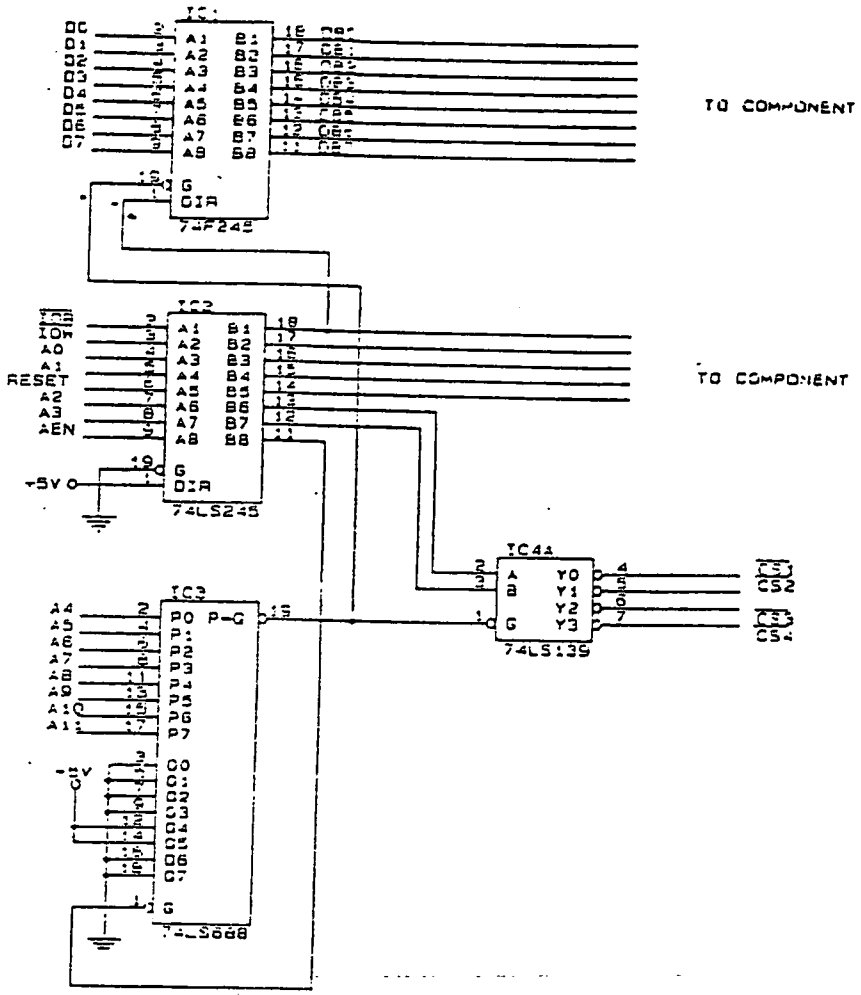
รูปที่ 4.3 แสดงบัสไซเคิลของการอ่านและเขียนข้อมูลจากพอร์ท

4.7 การ Decode

ในโครงงานพิเศษนี้ทำการ Decode Address ที่ 300H-30FH นำมาใช้ในการเชื่อมต่อกับ 8255 PPI และวงจรแปลงอนาลอกเป็นดิจิตอล ซึ่ง Address ดังกล่าวนั้นอยู่ในส่วนของ Prototype Card ซึ่งได้วงจร Decode ดังแสดงดังรูป 4.4

จากวงจรไอซี 74LS688 ซึ่งเป็นตัวเปรียบเทียบ ทำการเปรียบเทียบ ตำแหน่งที่ต้องการถอดรหัสว่าเป็นตำแหน่งที่เราตั้งไว้หรือไม่ ถ้าตรงตามที่ตั้งไว้จะให้ เอาท์พุทออกที่ขา 19 เป็น LOW ถ้าไม่ตรงจะเป็น HIGH สัญญาณที่ออกจากขา 19 นี้ ถ้าเป็น LOW จะทำให้ไอซี 74F245 และ 74LS139 สามารถทำงานได้ โดยไอซี 74F245 จะยอมให้ข้อมูลผ่านเข้าและออกได้ ไอซี 74LS139 เป็นตัวทำการเลือกชิพโดยผ่าน A2 และ A3

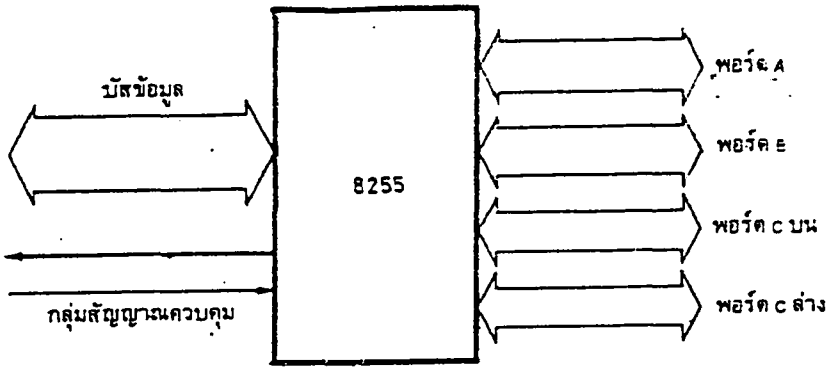
จากวงจร Decode ถ้าเราต้องการที่จะต่อวงจรอื่น ๆ เช่น 8255 หรือ A/D โดยนำมาเชื่อมต่อกับไมโครคอมพิวเตอร์ ก็นำมาเชื่อมต่อได้เลย โดยมีเอาท์พุทของไอซี 74LS139 เป็นตัวเลือกชิพ ซึ่งจะกล่าวถึงการเชื่อมต่อกับ 8255 PPI และ A/D ในบทต่อ ๆ ไป



รูป 4.4 แสดงวงจร Decode

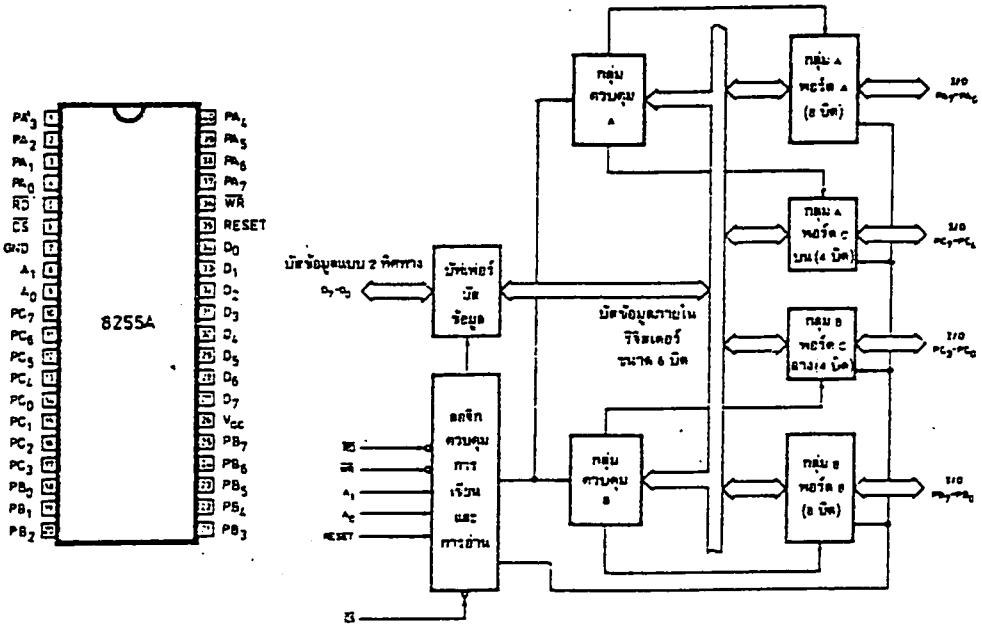
4.8 8255 พอร์ทข้อมูลแบบขนาน

8255 เป็นไอซีที่มี 40 ขา ได้รับการออกแบบมาให้มีสัญญาณเพื่อเชื่อมต่อกับ 8088 8255เป็นไอซีที่ต่อเป็นพอร์ทให้ไมโครโปรเซสเซอร์ได้ 3 พอร์ท โดยมีโครงสร้างพื้นฐานแสดงดังรูป 4.5



รูปที่ 4.5 แผนผังโครงสร้างของไอซี 8255

การเรียกพอร์ทของ 8255 จะเรียกพอร์ทต่าง ๆ ว่า พอร์ท A พอร์ท B และพอร์ท C โดยพอร์ท C แยกเป็น 2 ส่วนคือ พอร์ท C ล่างหรือตั้งแต่ PC0-PC3 มีจำนวน 4 บิต และพอร์ท C บน หรือตั้งแต่ PC4-PC7 ที่พิเศษคือ พอร์ททุกพอร์ทเป็นได้ทั้งพอร์ทอินพุทและพอร์ทเอาท์พุท



รูปที่ 4.6 แผนผังวงจรภายในและการจัดขาของไอซี 8255

รูปที่ 4.6 เป็นแผนผังภายในของไอซีและการจัดวางขาของไอซี 8255 การทำงานของวงจรจะใช้สัญญาณจากไมโครโปรเซสเซอร์มาควบคุมการทำงาน โดยไมโครโปรเซสเซอร์จะส่งคำสั่งมาโปรแกรมการทำงานหรือกำหนดรูปแบบของพอร์ทให้เป็นอินพุทหรือเอาท์พุทได้

4.8.1 ขาต่าง ๆ ของ 8255

D0-D7 เป็นขาที่ข้อมูลอินพุทเอาต์พุทจะต้องผ่านเข้าออกจากส่วนนี้ D0-D7 จึงต่อเข้ากับระบบบัสของไมโครโปรเซสเซอร์ เพื่อให้ไมโครโปรเซสเซอร์สามารถอ่านหรือเขียนข้อมูลจากพอร์ทผ่านทางบัสนี้

CS (สัญญาณเลือกชิป) ขานี้เป็นขาอินพุตที่จะรับสัญญาณจากภายนอกเพื่อเลือกชิป 8255 โดยเมื่อขานี้เป็น "0" จะทำให้ 8255 ต่อเข้ากับระบบบัสของไมโครโปรเซสเซอร์ เพื่อให้ไมโครโปรเซสเซอร์เขียนหรืออ่านข้อมูลจากพอร์ทได้

RD (สัญญาณการอ่าน) เป็นสัญญาณอินพุตที่ต้องส่งมาจากซีพียูอ่านข้อมูลจากบัสในขณะที่เป็นพอร์ทอินพุต

WR เป็นสัญญาณการเขียน จะแอคตีฟเมื่อสัญญาณ WR และสัญญาณ CS เป็น "0" สัญญาณนี้จะมาจากซีพียูเมื่อต้องการเขียนข้อมูลลงบนพอร์ทที่กำหนด

A0 - A1 (สัญญาณแอดเดรส) ลอจิกของสัญญาณทั้งสองจะถอดรหัสออกเป็น 4 รหัส เพื่อกำหนดรีจิสเตอร์ภายในที่เชื่อมต่อกับพอร์ทอินพุตของ 8255

RESET (สัญญาณรีเซต) เป็นสัญญาณที่ส่งจากภายนอกเข้ามาทำการรีเซต 8255 เพื่อเคลียร์สถานะต่าง ๆ ของ 8255 ได้รับการรีเซต ก็จะกลับเข้าสู่โหมดอินพุตหรือทุกพอร์ทที่เป็นพอร์ทอินพุต

PB0 - PB7 เป็นสายสัญญาณพอร์ท B ของ 8255 ถูกเลือกโดยสัญญาณแอดเดรส A0 - A1

PC0 - PC7 เป็นสายสัญญาณพอร์ท C ของ 8255 การกำหนดพอร์ทจะได้รับการกำหนดโดยสัญญาณแอดเดรส A0 - A1 พอร์ท C นี้แบ่งเป็น 2 กลุ่มคือ กลุ่ม PC0 -PC3 และกลุ่ม PC4 - PC7

4.8.2 รีจิสเตอร์ภายในของ 8255

เมื่อต่อ 8255 เข้ากับไมโครโปรเซสเซอร์ได้แล้ว สิ่งที่ผู้ใช้จะต้องทำก็คือ การโปรแกรมให้ 8255 ทำงานตามที่ต้องการ จากการที่ 8255 มีพอร์ทที่ไมโครโปรเซสเซอร์มองเห็น 4 พอร์ท แต่ละพอร์ทเสมือนเป็นรีจิสเตอร์ที่สามารถเขียนและอ่านได้ รีจิสเตอร์แต่ละตัวนี้จึงถูกกำหนดด้วยแอดเดรสตามที่ตั้งไว้ เช่น ในกรณีที่เป็นแอดเดรส 10H,11H,

12H และ 13H รีจิสเตอร์แต่ละตัวจะได้รับการกำหนดควบคู่กับสัญญาณ RD และ WR เพื่อแสดงความหมาย ตัวอย่างเช่น พอร์ท 10H เป็นพอร์ท A ซึ่งเมื่อเขียนที่พอร์ทนี้ จะเป็นการส่งข้อมูลเอาท์พุท และถ้าอ่านพอร์ทนี้ก็จะเป็นการอินพุตข้อมูลจากพอร์ท ดังนั้นสัญญาณของขาควบคุมที่ประกอบกันจะแสดงความหมายดังตารางที่ 4.1

RD	WR	A1	A0	ความหมาย
1	0	0	0	เขียนพอร์ต A ซึ่งเป็นข้อมูล
0	1	0	0	อ่านพอร์ต A ซึ่งเป็นข้อมูล
1	0	0	1	เขียนพอร์ต B ซึ่งเป็นข้อมูล
0	1	0	1	อ่านพอร์ต B ซึ่งเป็นข้อมูล
1	0	1	0	เขียนพอร์ต C ซึ่งเป็นข้อมูล
0	1	1	0	อ่านพอร์ต C ซึ่งเป็นข้อมูล
1	0	1	1	เขียนข้อมูล ซึ่งเป็นรหัสควบคุม
0	1	1	1	อ่านเข้ามา ซึ่งไม่มีความหมายใด

ตารางที่ 4.1 สัญญาณควบคุมการกระทำของ 8255

การใช้งาน 8255 จะต้องส่งรหัสควบคุม (Control code) เข้าไปยังพอร์ทข้อมูลควบคุมการทำงานของ 8255 โดยใช้สัญญาณควบคุมพอร์ทหมายเลข 13H การควบคุมการทำงานของ 8255 มีหลายโหมด แต่ละโหมดจะแตกต่างกันออกไป การโปรแกรมให้ 8255 ทำงานจะทำได้ 3 โหมดคือ โหมด 0 โหมด 1 และ โหมด 2

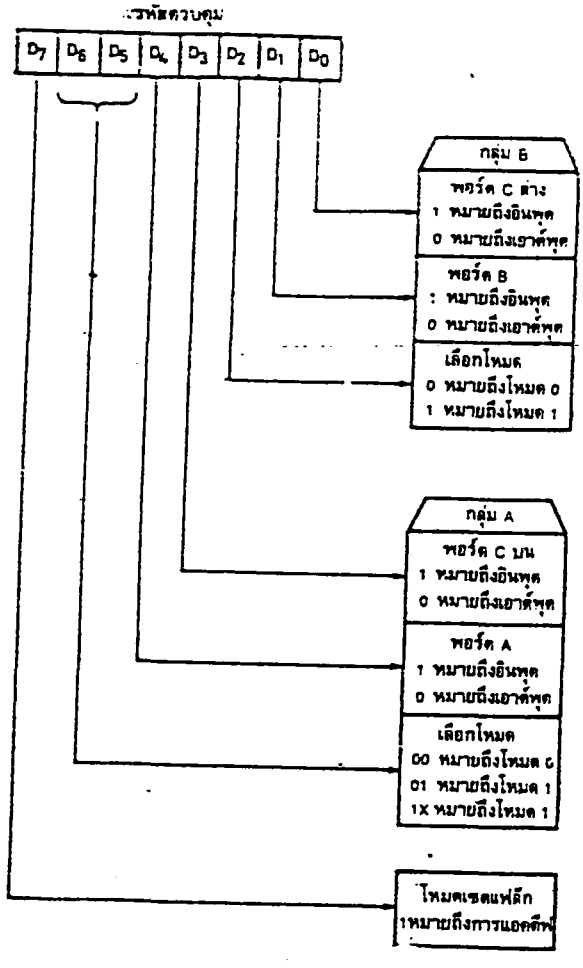
โหมด 0 หรือ อินพุตเอาต์พุตแบบพื้นฐาน

การกำหนดในการทำงานจะต้องส่งข้อมูลคำสั่งเข้าไปโปรแกรมในพอร์ทควบคุม ของ 8255 แต่ละบิตของข้อมูลที่ส่งไปจะมีความหมายในตัวเอง ลักษณะความหมายของแต่ละบิต ในรหัสควบคุมแสดงได้ดังรูปที่ 4.7

การโปรแกรม 8255 คือ การให้ค่ารหัสบิตต่าง ๆ เข้าไปในรหัสควบคุมแล้วส่งไปยังรีจิสเตอร์ของพอร์ทควบคุม ความหมายของบิตต่าง ๆ มีดังนี้

บิต D7 เป็นบิตที่แสดงรหัสคำสั่งควบคุม ถ้าบิตนี้เป็น "1" หมายถึงรหัสควบคุมนี้จะมีผลต่อการเปลี่ยนแปลงการเซตโหมดต่าง ๆ ของ 8255

บิต D6 และ D5 เป็นการเลือกโหมดของพอร์ท A ซึ่งมี 3 โหมดคือ โหมด 0 โหมด 1 และ โหมด 2 ดังแสดงในรูปที่ 4.7



รูป 4.7 ความหมายของบิตต่าง ๆ ในรหัสควบคุม

บิต D4 ถ้ามีค่าเป็น "0" หมายถึง การกำหนดพอร์ท A เป็นเอาท์พุตถ้าที่ค่าเป็น "1" จะหมายถึงการกำหนดให้พอร์ท A เป็นอินพุต

บิต D3 เป็นบิตที่บอกถึงการเซตของพอร์ท C บน ถ้าเป็น "0" จะทำให้พอร์ท C บนเป็นเอาท์พุท

บิต D2 เป็นบิตที่บอกถึงการเซตโหมดของพอร์ท B ถ้าเป็น "0" หมายถึงการเลือกพอร์ท B เป็นโหมด 0 และถ้าเป็น "1" หมายถึงการเลือกโหมด 1

บิต D1 เป็นการกำหนดอินพุตเอาท์พุทของพอร์ท B ถ้าเป็น "0" หมายถึงเอาต์พุต ถ้าเป็น "1" หมายถึงอินพุต

บิต D0 เป็นการกำหนดอินพุตเอาท์พุทของพอร์ท C ล่าง ถ้าเป็น "0" หมายถึงเอาท์พุท ถ้าเป็น "1" หมายถึงอินพุต

ไอซี 8255 นี้ซึ่งจะเป็นไอซีที่ประกอบด้วยพอร์ทใช้งาน 3 พอร์ท และอีก 1 พอร์ทควบคุม ก่อนที่จะใช้งาน 8255 เราจะต้องส่งข้อมูลไปให้ยังพอร์ทควบคุมก่อนว่าจะให้พอร์ททั้ง 3 ของ 8255 ที่เหลือนั้นทำหน้าที่อะไร เป็นอินพุทพอร์ท หรือ เอาต์พุทพอร์ท เราต้องเป็นผู้กำหนดรหัสควบคุมพอร์ท

การโปรแกรม 8255 จะเริ่มจากการเซตค่าที่ต้องการแล้วเอาต์พุทไปยังพอร์ทควบคุม เช่น ถ้าต้องการโปรแกรมให้ทั้งพอร์ท A,B,และ C เป็นพอร์ทเอาท์พุทหมด เราจะเลือก 8255 ให้อยู่ในโหมด 0 โดยมีรหัสควบคุมเป็น 10000000 หรือ 80H แล้วทำการส่งเอาท์พุตไปยังพอร์ทควบคุม ก่อนที่จะส่งข้อมูลว่าให้พอร์ทไหนเป็นอะไรต้องทำการโดยการส่งรหัสของพอร์ทออกไปก่อนโดยผ่าน A0 และ A1 รหัสเลือกพอร์ทเป็นดังนี้

A1	A0	
0	0	พอร์ท A
0	1	พอร์ท B
1	0	พอร์ท C
1	1	คอนโทรลพอร์ท

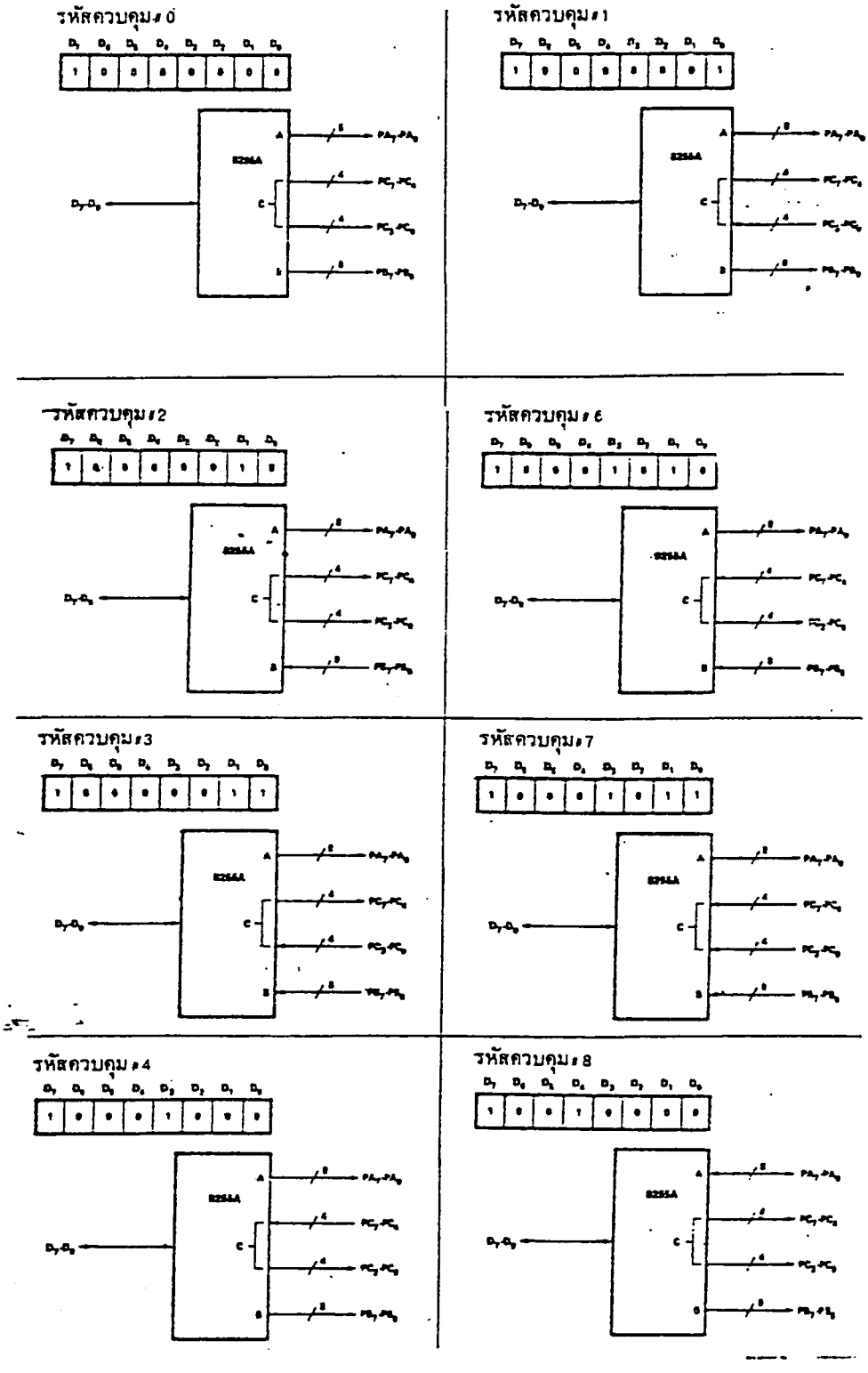
ตารางที่ 4.2 แสดงรหัสในการเลือกพอร์ท

การใช้งาน 8255 นั้นต้องการทำการส่งรหัสควบคุมพอร์ทผ่านพอร์ทคอนโทรลก่อน เพื่อให้ 8255 รู้ว่าพอร์ทไหนทำหน้าที่อะไร จากนั้นจึงค่อยส่งคำสั่งอินพุท หรือ เอาท์พุท พอร์ทออกไปที่พอร์ท A,B, และ C ตามที่กำหนดไว้

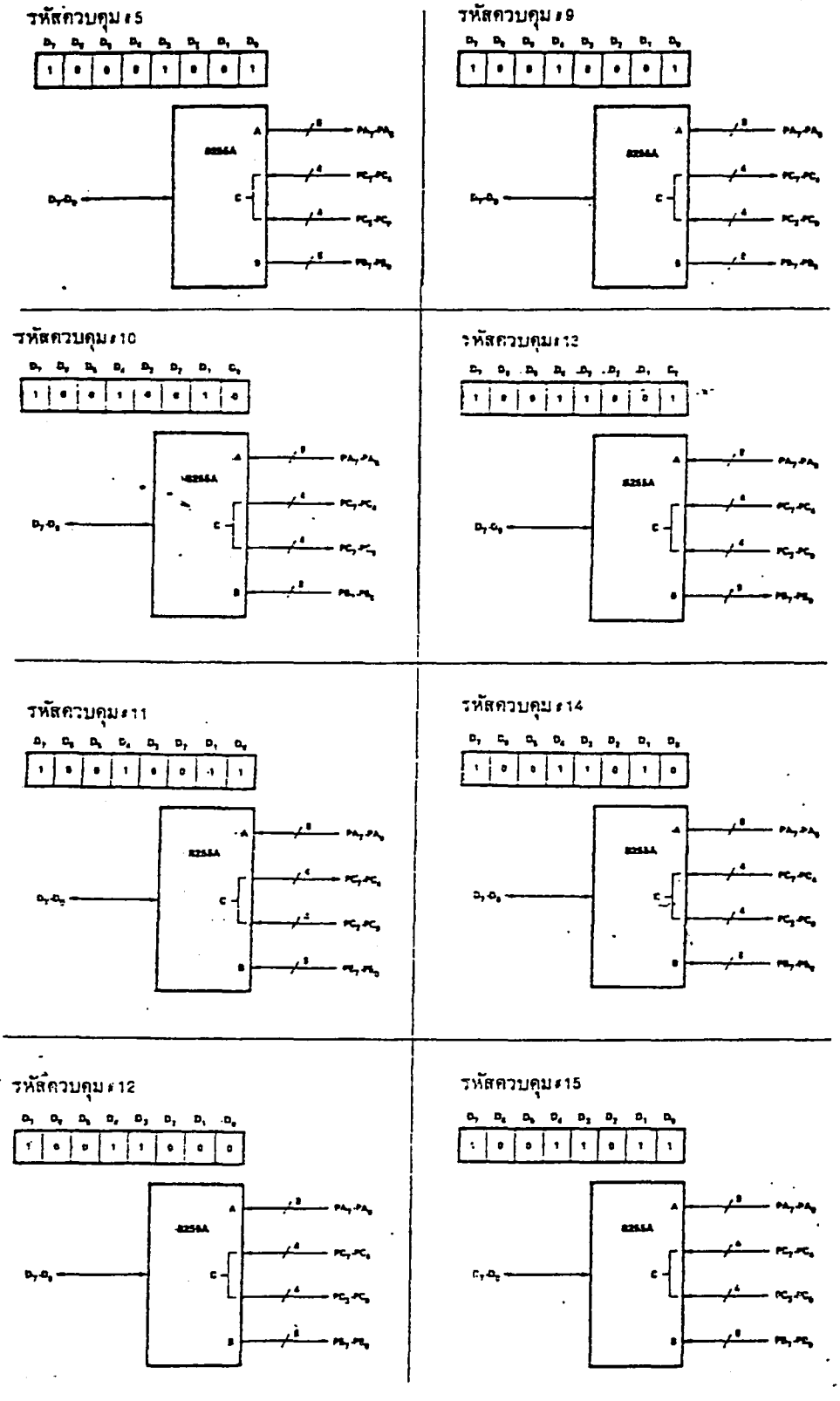
เนื่องจากที่พอร์ทที่รับส่งข้อมูล 3 พอร์ทคือ พอร์ท A พอร์ท B และพอร์ท C ซึ่ง พอร์ท C จะแยกออกเป็น 2 ส่วนคือ พอร์ท C ล่าง และพอร์ท C บน เราสามารถ โปรแกรมให้ทั้ง 4 พอร์ทนี้ เป็นอินพุตหรือเอาท์พุทก็ได้เช่น ถ้าให้รหัสควบคุมเป็น 82H จะ ทำให้พอร์ท B เป็นอินพุต พอร์ท A และพอร์ท C เป็นเอาท์พุต

การทำงานในโหมด 0

โหมด 0 เป็นโหมดที่กำหนดให้พอร์ททุกพอร์ทบนตัว 8255 เป็นพอร์ทอินพุท เอาท์ พุทแบบพื้นฐาน รูปแบบความเป็นไปได้จึงมีทั้งสิ้น 16 รูปแบบตามลักษณะของพอร์ท A พอร์ท B พอร์ท C บน และพอร์ท C ล่าง ลักษณะของรหัสควบคุมแต่ละแบบ เป็นดังรูปที่ 4.8



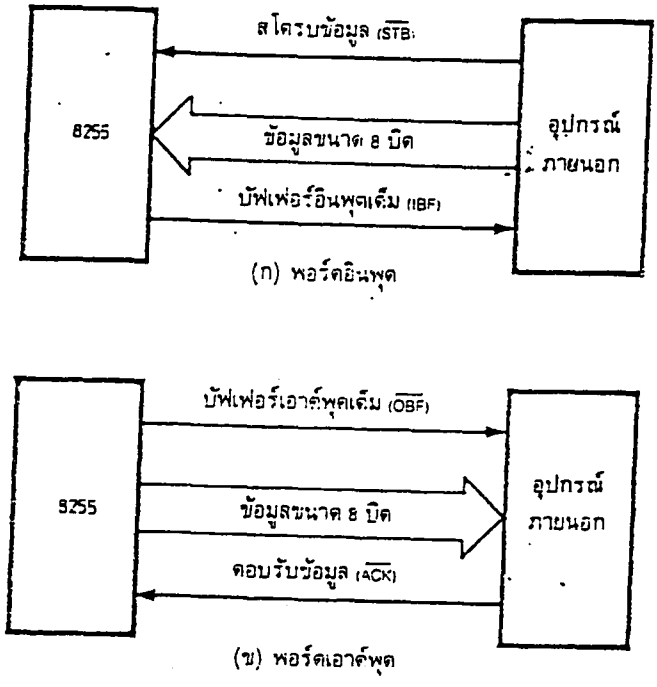
รูปที่ 4.8 ลักษณะของรหัสควบคุมแบบต่าง ๆ ในโหมด 0



รูปที่ 4.8 (ต่อ) ลักษณะของรหัสควบคุมแบบต่าง ๆ ในโหมด 0

การทำงานของ 8255 ในโหมด 1

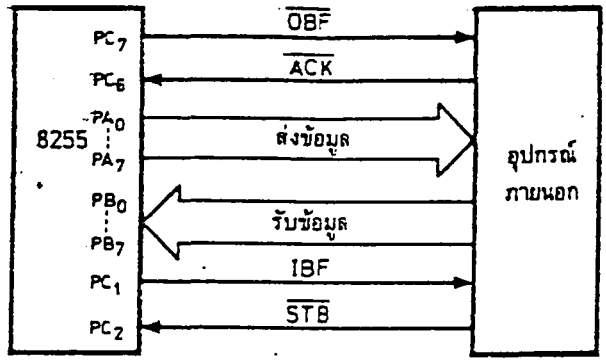
การทำงานของ 8255 ในโหมด 1 เป็นโหมดที่ทำให้อินพุตเอาท์พุทมีการตรวจสอบสัญญาณ (Handshaking) โดยใช้อินพุตเอาท์พุทของพอร์ท A และพอร์ท B เป็นหลัก และใช้พอร์ท C บนเป็นตัวตรวจสอบสัญญาณ (Handshake) ของพอร์ท A ส่วนพอร์ท C ล่างเป็นตัวตรวจสอบสัญญาณของพอร์ท B การจัดสัญญาณต่าง ๆ เหล่านี้ แสดงได้ดังรูปที่ 4.9



รูปที่ 4.9 โครงสร้างตัวตรวจสอบสัญญาณของพอร์ทอินพุทและพอร์ทเอาต์พุท

แนวความคิดของการใช้พอร์ทอินพุตเอาท์พุทโดยมีตัวตรวจสอบสัญญาณก็เพื่อให้มีการซิงโครไนซ์ระหว่างอุปกรณ์ภายนอกที่ทำงานได้กับการทำงานของคอมพิวเตอร์ที่ทำงานได้เร็วเช่นเครื่องพิมพ์ทำงานได้ช้า เมื่อคอมพิวเตอร์ส่งตัวอักษรตัวแรกมาพิมพ์ เครื่องพิมพ์ที่รับตัวอักษรและกำลังจะพิมพ์ คอมพิวเตอร์ก็ส่งตัวอักษรตัวที่ 2 ตัวที่ 3 ตามมา ทำให้การประมวลผลของอุปกรณ์เครื่องพิมพ์ทำงานไม่ทัน ซึ่งอาจทำให้ข้อมูลสูญหาย ดังนั้นเครื่องพิมพ์จึงต้องส่งสัญญาณบอกคอมพิวเตอร์ว่ายังไม่พร้อมที่จะรับ ลักษณะของการรับส่งข้อมูล

อินพุต เอาท์พุทแบบมีตัวตรวจสอบสัญญาณ จะใช้ PA_0-PA_7 เป็นเอาท์พุท และ PB_0-PB_7 เป็นอินพุตโดยมีพอร์ท C เป็นตัวตรวจสอบสัญญาณ ดังแผนผังในรูปที่ 4.10



รูปที่ 4.10 วงจรการต่อ 8255 ในโหมด 1

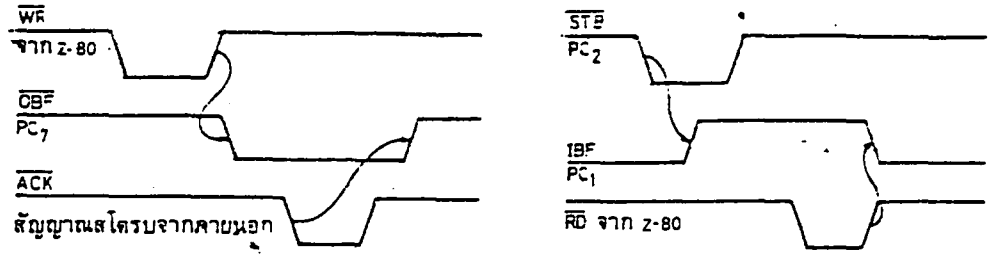
เมื่อโปรแกรม 8255 เป็นโหมด 1 แล้ว ตัว 8255 จะให้พอร์ท C เป็นสัญญาณควบคุม โดยแต่ละบิตของพอร์ท C เป็นไปตามที่กำหนดไว้ ดังตารางที่ 4.3

ขา	กรณีอินพุท	กรณีเอาท์พุท
PC0	$INTR_B$	$INTR_B$
PC1	IBF_B	OBF_B
PC2	STB_B	ACK_B
PC3	$INTR_A$	$INTR_A$
PC4	STB_A	I/O
PC5	IBF_A	I/O
PC6	I/O	ACK_A
PC7	I/O	OBF_A

ตารางที่ 4.3 หน้าที่ของสัญญาณต่าง ๆ ของพอร์ท C ในการทำงานเป็นตัวตรวจสอบสัญญาณเมื่อ 8255 ทำงานในโหมด1

โดยปกติ 8255 จะให้สัญญาณอินเตอร์รัพต์ไปบอกซีพียูด้วย สัญญาณอินเตอร์รัพต์ของ 8255 จะเกิดขึ้นที่ PC_0 และ PC_3 โดยที่เมื่อบัฟเฟอร์พร้อมแล้วและต้องการให้มีซีพียูส่งอินพุต หรือเอาท์พุทมาที่บัฟเฟอร์ สัญญาณอินเตอร์รัพต์ก็จะเกิดขึ้น สังเกตว่า สัญญาณอินเตอร์รัพต์เป็นสัญญาณแอคทีฟ "1" ซึ่งตรงกับของ 8080

โครงสร้างการตรวจสอบสัญญาณของ 8255 แสดงด้วยสัญญาณทางไฟฟ้าได้ดังรูปที่ 4.11



(ก) เมื่อเป็นพอร์ตเอาต์พุต (ข) เมื่อเป็นพอร์ตอินพุต

รูปที่ 4.11 แผนผังเวลาการรับและส่งข้อมูลโดยใช้ตัวตรวจสอบสัญญาณ

สังเกตว่า การทำงานของ 8255 จะเกี่ยวข้องกับสัญญาณ RD และ WR ซึ่งจะทำให้สัญญาณควบคุมเปลี่ยนแปลงไป การตรวจสอบสัญญาณซึ่งกันและกันนี้ เป็นวิธีการรับส่งที่มีประสิทธิภาพ เช่น ในกรณีอินพุต เมื่ออุปกรณ์ภายนอกต้องการส่งข้อมูลให้ซีพียู ก็จะส่งข้อมูลแบบขนานเข้ามาพร้อมทั้งสโตรบ(STB) บอก 8255 ซึ่ง 8255 จะนำข้อมูลนั้นไปเก็บไว้ในรีจิสเตอร์ภายในก่อนแล้วส่งสัญญาณตอบบอกว่า บัฟเฟอร์ยังเต็มแล้ว (IBF) ครั้นเมื่อซีพียูอ่านข้อมูลจากรีจิสเตอร์ไปแล้ว ส่วนของสัญญาณบัฟเฟอร์ยังเต็มแล้ว (IBF) ก็จะบอกว่าบัฟเฟอร์ว่างอุปกรณ์ภายนอกก็จะส่งข้อมูลมาให้อีก

ทำนองเดียวกัน สำหรับพอร์ทเอาท์พุท เมื่อซีพียูส่งข้อมูลออกทางพอร์ทเอาท์พุทให้กับ 8255 ตัว 8255 ก็จะรับไว้ในรีจิสเตอร์ภายใน พร้อมทั้งส่งสัญญาณออกไปบอกอุปกรณ์ภายนอกว่าข้อมูลในบัฟเฟอร์พร้อม (OBF) อุปกรณ์ภายนอกเมื่อทราบ และพร้อมที่จะอ่านก็จะส่งสัญญาณตอบรับ (ACK) พร้อมกับอ่านข้อมูลไป โดยสัญญาณ ACK จะมีความหมายว่า ทำการอ่านข้อมูลไปแล้ว ตัว 8255 ก็จะทำการนำข้อมูลใหม่ส่งมาให้อีก

ในการที่โปรแกรมโหมด 1 เราใช้รหัสควบคุมเป็น 101(I/O) 01(I/O)0 ในส่วน I/O หมายถึง ถ้าเป็นอินพุตก็คือ "1" ถ้าเป็นเอาท์พุท "0" โดย I/O ตัวแรกเป็นของพอร์ท A ตัวที่ 2 เป็นของพอร์ท B เช่น ถ้าต้องการให้พอร์ท A เป็นเอาต์พุท และพอร์ท B เป็นอินพุท เราจะใช้รหัสควบคุมเป็น 10100110 หรือ A6H

จากการพิจารณาการทำงานของซีพียูจะเห็นว่า ทำอย่างไรจึงจะเขียนหรืออ่านพอร์ทได้ถูกต้อง วิธีที่ง่ายวิธีหนึ่งคือ ซีพียูจะคอยตรวจสอบสัญญาณของ 8255 เช่น กรณีเอาท์พุท ซีพียูจะคอยอ่านพอร์ท C แล้วตรวจสอบบิต 7(OBF) หลังจากส่งข้อมูลไปแล้ว ถ้าบิต 7 ยังเป็น "0" แสดงว่ายังไม่ได้รับการสโตรบ แต่ถ้าเป็น "1" แล้ว แสดงว่าอุปกรณ์ภายนอกรับข้อมูลไปแล้วสำหรับกรณีอินพุตก็คอยตรวจสอบจากสัญญาณ IBF ได้เช่นกันว่า มีข้อมูลใหม่เข้ามาหรือยัง คือตรวจสอบบิต PC_1 ของพอร์ท C

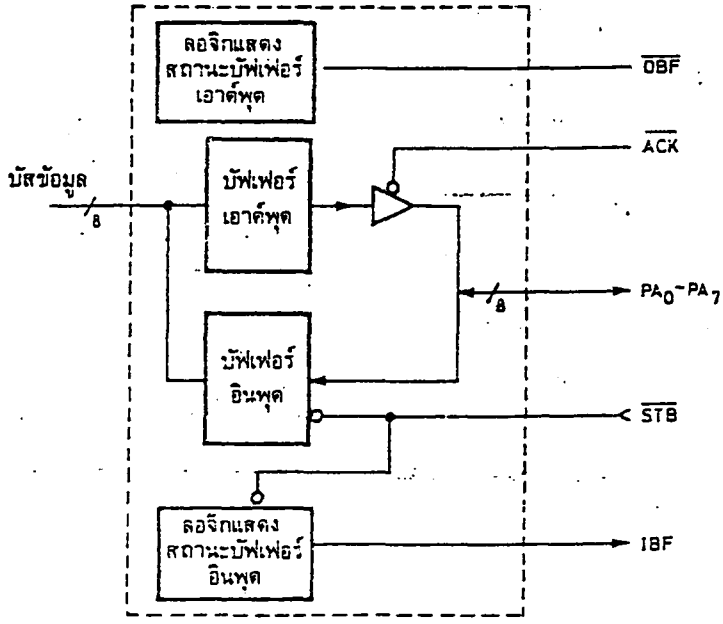
การทำงานของ 8255 ในโหมด 2

8255 ยังโหมดการทำงานอีกโหมดหนึ่งคือ โหมด 2 ซึ่งทำได้เฉพาะพอร์ท A ในโหมดนี้ 8255 ใช้พอร์ท A ทำหน้าที่เป็นพอร์ทแบบ 2 ทิศทางคือ สามารถเป็นได้ทั้งพอร์ทอินพุทและเอาท์พุต โดยโครงสร้างของพอร์ท A ทั้งอินพุตเอาต์พุตมีตัวตรวจสอบสัญญาณทั้งคู่ ส่วนพอร์ท C จะทำหน้าที่เป็นสัญญาณตรวจสอบ โดยมีสัญญาณแต่ละขาดังตารางที่ 4.4

พอร์ท C	ความหมาย
PC0	I/O
PC1	I/O
PC2	I/O
PC3	INTR
PC4	STB
PC5	IBF
PC6	ACK
PC7	OBF

ตารางที่ 4.4 หน้าที่ของพอร์ท C ในโหมด 2

โครงสร้างของพอร์ท A ที่ทำงานแบบ 2 ทิศทาง แสดงได้ดังรูปที่ 4.12

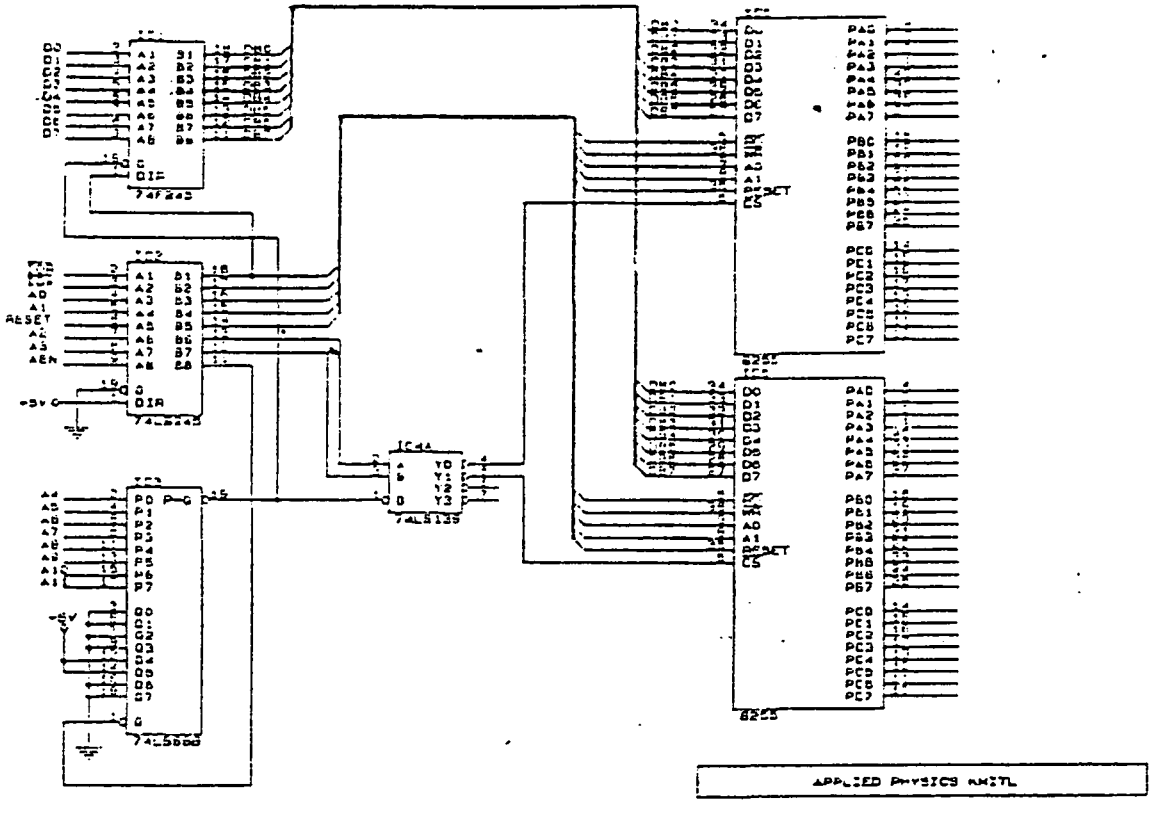


รูปที่ 4.12 โครงสร้างของพอร์ท A ที่ทำงานแบบพอร์ท 2 ทิศทาง

สังเกตว่า เมื่อโปรแกรมพอร์ท A เป็นโหมด 2 แล้วพอร์ท B จะต้องโปรแกรมเป็น โหมด 0 หรือไม่ 1 ก็ได้ ซึ่งก็ทำงานแบบแยกอิสระอีก ในการใช้งานพอร์ทแบบ 2 ทิศทางนี้ใช้ได้กับงานบางประเภท เช่น ใช้ในการรับส่งข้อมูลของพอร์ทมาตรฐานบางประเภท เช่น IEEE 488 หรือไม่เชื่อมโยงระหว่างคอมพิวเตอร์กับคอมพิวเตอร์ในการรับส่งข้อมูลสลับกันไปและกลับ

การต่อ 8255 เข้ากับวงจร Decode

จากวงจร Decode ที่เคยกล่าวมาแล้วในบทก่อน ๆ เรานำมาทำการต่อกับ 8255 ได้ดังรูปที่ 4.13



รูปที่ 4.13 แสดงการต่อ 8255 กับวงจร Decode

ซึ่งจะใช้การเรียกพอร์ท แต่ละพอร์ทของ 8255 ดังนี้

		A11.....A0
พอร์ท A	300H	0011 0000 0000
พอร์ท B	301H	0011 0000 0001
พอร์ท C	302H	0011 0000 0010
คอนโทรลพอร์ท	303H	0011 0000 0011

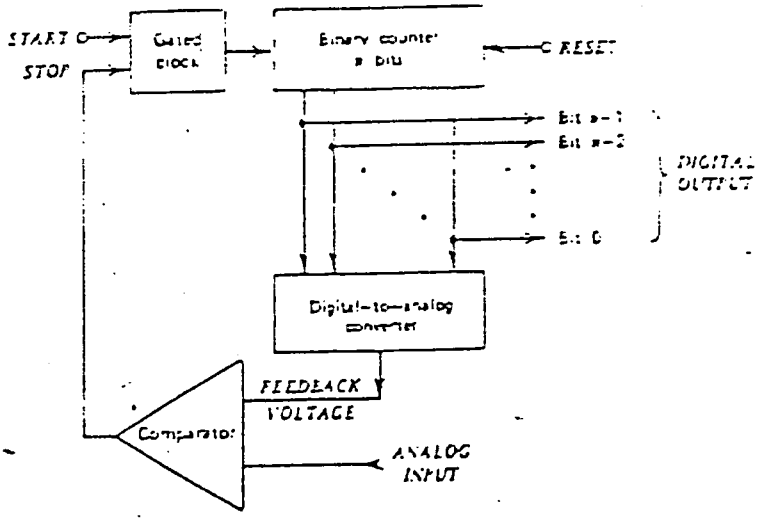
จะเห็นว่าการเรียกพอร์ทแต่ละพอร์ทนั้นจะต้องควบคุมโดยผ่าน A0 และ A1 ดังได้กล่าวมาแล้วในตอนต้น

4.9 การแปลงสัญญาณอนาลอกเป็นสัญญาณดิจิตอล

ในการวัดสัญญาณต่าง ๆ นั้นโดยทั่วไปถ้าเป็นในวงจรใหญ่ที่ใช้ไมโครคอมพิวเตอร์ควบคุมการทำงาน และประมวลผลค่าสัญญาณอนาลอกนั้น จะต้องมีการแปลงสัญญาณอนาลอกให้เป็นสัญญาณดิจิตอล เพื่อสามารถป้อนให้กับไมโครคอมพิวเตอร์ทำการประมวลผลต่อไปได้ ส่วนใหญ่แล้วการแปลงสัญญาณอนาลอกเป็นสัญญาณดิจิตอลมีได้หลายวิธี แต่วิธีที่นิยมใช้ส่วนใหญ่ได้แก่

1. การแปลงสัญญาณอนาลอกเป็นดิจิตอลโดยวิธีธรรมดา หรือ วิธีแลมป์ (Basic ADC or Ramp ADC)
2. การแปลงสัญญาณอนาลอกเป็นดิจิตอลโดยวิธีต่อเนื่อง (Continous Convert)
3. การแปลงสัญญาณอนาลอกเป็นดิจิตอลโดยวิธีประมาณค่าทีละบิต (Sucessive Approximation Convertor)
4. การแปลงสัญญาณอนาลอกเป็นดิจิตอลโดยวิธีแฟลช (Flash ADC)
5. การแปลงสัญญาณอนาลอกเป็นดิจิตอลโดยวิธีสโลปคู่ (Dual - Slope Intergrator ADC)

4.9.1 การแปลงสัญญาณอนาลอกเป็นดิจิตอลโดยวิธีธรรมดา หรือ วิธีแลมป์

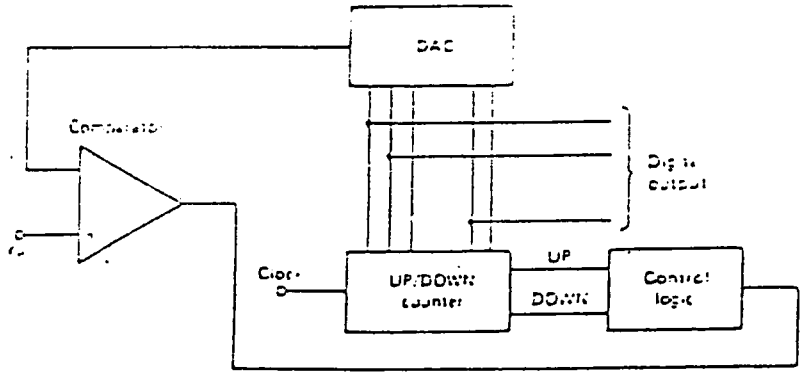


รูปที่ 4.14 แสดงวงจรการแปลงสัญญาณอนาลอกเป็นดิจิตอลโดยวิธีธรรมดา หรือ วิธีแลมป์

จากรูปที่ 4.14 จะเห็นได้ว่า วงจรนี้ประกอบด้วย วงจรเปรียบเทียบแรงดัน (Voltage Comparator) , วงจรแปลงสัญญาณดิจิตอลเป็นอนาลอก (Digital to Analog Convertor หรือ DAC) วงจรนับเลขฐานสอง (Binary Counter) , แนนด์เกท และวงจรกำเนิดพัลส์ (Clock Pulse หรือ CP) ซึ่งมีขั้นตอนในการทำงานดังนี้

เริ่มต้นวงจรนับเลขฐานสองจะถูกตั้งให้มีค่าเป็น 0 เมื่อสัญญาณจากกวงจรกำเนิดพัลส์ ผ่านแนนด์เกทเข้ามา จะทำให้วงจรนับเลขฐานสองเริ่มทำการนับ สัญญาณเอาท์พุทที่ได้จากวงจรนับเลขฐานสองจะนำไปเข้าวงจรแปลงสัญญาณจากดิจิตอลเป็นอนาลอก เพื่อทำการแปลงให้เป็นสัญญาณอนาลอก แล้วนำสัญญาณอนาลอกที่ได้มาเปรียบเทียบกับสัญญาณอินพุทที่เข้ามาโดยใช้วงจรเปรียบเทียบแรงดัน เมื่อสัญญาณอนาลอกที่ได้ จากวงจรแปลงสัญญาณดิจิตอลเป็นอนาลอกมีค่าเท่ากับ หรือมากกว่า สัญญาณอินพุท ที่เข้ามา วงจรนับเลขฐานสองจะหยุดทำงาน ดังนั้นสัญญาณดิจิตอลเอาท์พุท ที่ได้จากวงจรนับเลขฐานสอง จะมีค่าเท่ากับสัญญาณอินพุทที่เข้ามา

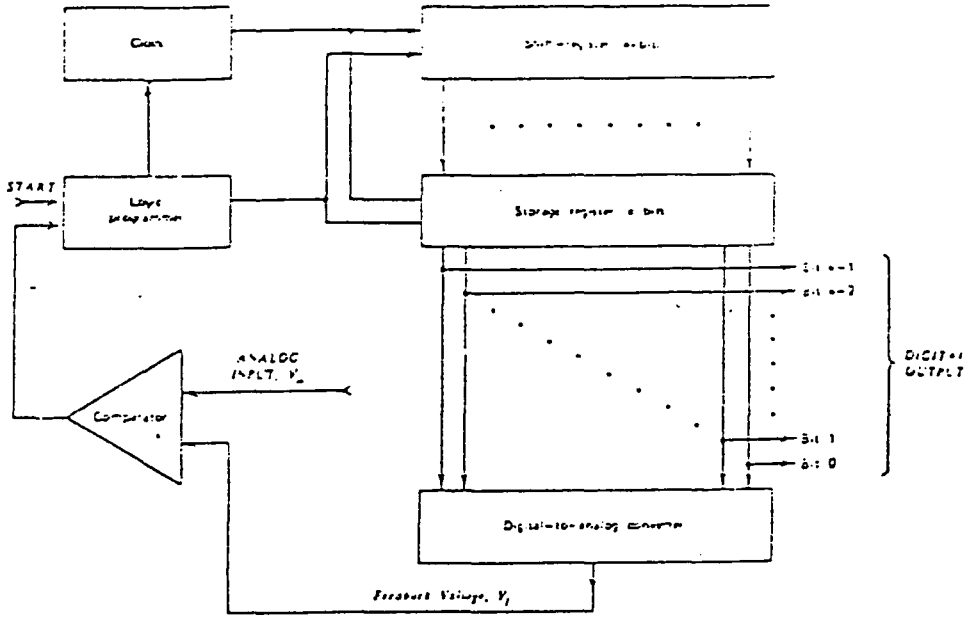
4.3.2 การแปลงสัญญาณอนาลอกเป็นดิจิตอลโดยวิธีต่อเนื่อง



รูปที่ 4.15 แสดงวงจรการแปลงสัญญาณอนาลอกเป็นดิจิตอลโดยวิธีต่อเนื่อง

วงจรนี้มีหลักการทำงานคล้ายกับ การแปลงสัญญาณอนาลอกเป็นดิจิตอล โดยวิธีธรรมดา หรือวิธีแลมป์ เพียงแต่ต่างกันที่สัญญาณเอาท์พุทของวงจรเปลี่ยบเทียบแรงดันจะไปป้อนเข้าที่วงจรควบคุมลอจิก (Control Logic) เพื่อทำการเปลี่ยนแปลงค่าของวงจรนับเลขฐานสองขึ้น-ลง (Up - Down Binary Counter) ถ้าสัญญาณเอาท์พุทของวงจรแปลงสัญญาณดิจิตอลเป็นอนาลอก น้อยกว่าสัญญาณ e_{in} จะทำให้วงจรควบคุมลอจิก ไปควบคุมวงจรนับเลขฐานสองขึ้น-ลง ทำให้สัญญาณเอาท์พุทของวงจรนี้ มีค่าเพิ่มขึ้นจนกว่าจะได้สัญญาณเอาท์พุทจากวงจรนับเลขฐานสองขึ้น-ลง เท่ากับสัญญาณอนาลอก e_{in} วงจรควบคุมลอจิก ก็จะทำการเก็บค่านั้นไว้จนกว่าจะมีการเปลี่ยนแปลงของสัญญาณอนาลอก e_{in}

4.3.3 <u>การแปลงสัญญาณอนาลอกเป็นสัญญาณดิจิตอลโดยวิธีประมาณทีละบิต</u>

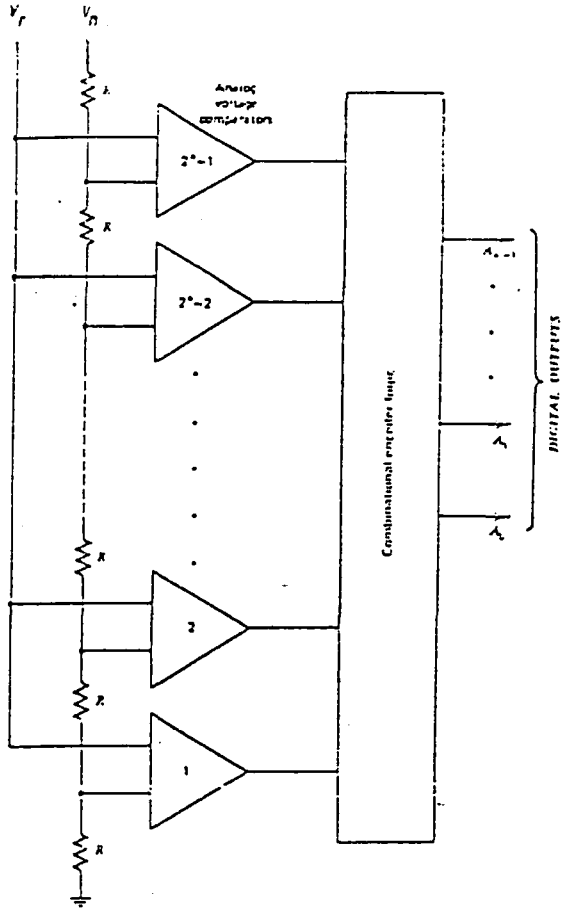


รูปที่ 4.16 แสดงวงจรการแปลงสัญญาณอนาลอกเป็นดิจิตอลโดยวิธีประมาณทีละบิต

เป็นวงจรแปลงสัญญาณอนาลอกเป็นดิจิตอล ที่มีความละเอียด และรวดเร็ว โดยมีหลักการดังนี้คือ

เริ่มต้นชีฟต์รีจีสเตอร์ จะถูกเคลียร์ให้เป็น 0 และ 1 จะถูกใส่เข้าไปยังตำแหน่งบิตที่มีค่านัยสำคัญสูงสุด (Most Significant Bit หรือ MSB) ในชีฟต์รีจีสเตอร์ สัญญาณเอาท์พุทที่ได้จากชีฟต์รีจีสเตอร์ จะผ่านไปยังวงจรแปลงสัญญาณดิจิตอลเป็นอนาลอก และนำสัญญาณอนาลอกที่ได้ มาเข้าวงจรเปรียบเทียบแรงดัน เพื่อเปรียบเทียบแรงดัน เพื่อเปรียบเทียบกับสัญญาณอินพุทที่เข้ามา ถ้าสัญญาณอินพุทมีค่ามากกว่า สัญญาณอนาลอก บิตที่มีค่านัยสำคัญสูงสุดจะคงค่าเดิม 1 ไว้แต่ถ้าสัญญาณอินพุทมีค่าน้อยกว่า สัญญาณอนาลอก บิตที่มีค่านัยสำคัญสูงสุดจะเปลี่ยนเป็น 0 แทน หลังจากนั้น บิตที่มีค่านัยสำคัญต่ำสุด (Least Significant Bit หรือ LSB) ดังนั้นสัญญาณดิจิตอลเอาท์พุทที่ได้จากชีฟต์รีจีสเตอร์ จะมีค่าเท่ากับสัญญาณอินพุทที่เข้ามา

4.3.4 <u>การแปลงสัญญาณอนาลอกเป็นดิจิตอลโดยวิธีแฟลช</u>



รูปที่ 4.17 แสดงวงจรการแปลงสัญญาณอนาลอกเป็นดิจิตอลโดยวิธีแฟลช

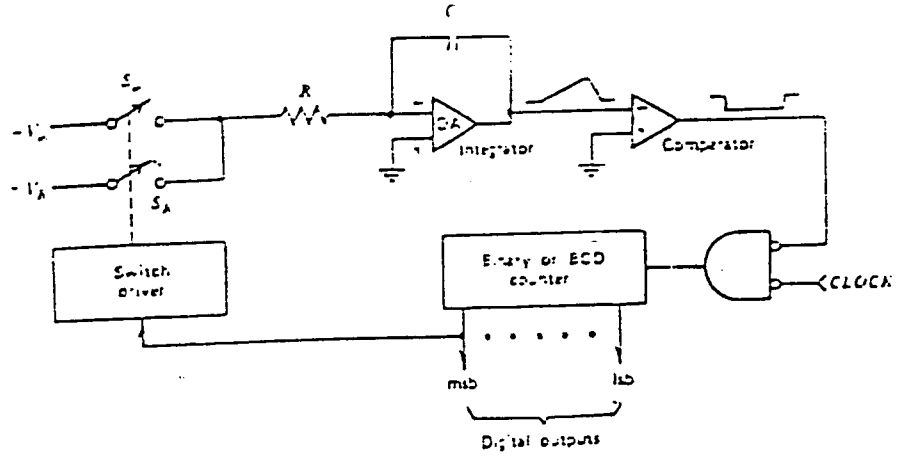
การแปลงสัญญาณอนาลอกเป็นสัญญาณดิจิตอลโดยวิธีแฟลช มีความเร็วสูงมาก และเป็นที่นิยมกันมากในปัจจุบัน แต่ราคาของ I.C. ยังสูงอยู่ จากรูปเห็นได้ว่าวงจรประเภทนี้ประกอบด้วย วงจรเปรียบเทียบแรงดัน ซึ่งมีจำนวน 2^{n-1} และวงจรถอดรหัสรวม (Combination Encoder Logic) ซึ่งใช้หลักการทำงานดังนี้คือ

1.) สัญญาณอนาลอก e_{in} ที่ต้องการแปลงเป็นสัญญาณดิจิตอล จะถูกนำมาเข้าวงจรเปรียบเทียบแรงดันทั้งหมด

2.) อินพุทอีกขาหนึ่งวงจรเปรียบเทียบแรงดันจะต่อเข้ากับ V_{ref} ที่ทราบค่าแน่นอนโดยใช้ตัวต้านทานเป็นตัวแบ่งกระแสให้มีความต่างศักย์ต่าง ๆ กัน โดยให้ลดลงมาเรื่อย ๆ แล้วนำ V_{ref} ที่แบ่งแรงดันแล้วนำมาเข้าวงจรเปรียบเทียบแรงดันทุกตัวดังนั้นก็จะได้ V_{ref} มีค่าลดลงเรื่อย ๆ ในวงจรเปรียบเทียบแรงดันทั้งหมด

3.) สัญญาณเอาท์พุทที่ได้จากวงจรเปรียบเทียบแรงดัน จะถูกนำมาเข้าวงจรถอดรหัสรวม โดยนำสัญญาณเอาท์พุทที่ได้จากวงจรเปรียบเทียบแรงดัน ที่สัญญาณอนาลอก e_{in} มีค่าเท่ากับ V_{ref} มาถอดรหัสเป็นสัญญาณดิจิตอล n บิต ซึ่งมีค่าเท่ากับสัญญาณอนาลอก e_{in} ที่เข้ามา

4.3.5 การแปลงสัญญาณอนาลอกเป็นดิจิตอลโดยวิธีสโลปคู่



รูปที่ 4.18 แสดงวงจรการแปลงสัญญาณอนาลอกเป็นดิจิตอลโดยวิธีสโลปคู่

เป็นวงจรที่ไม่จำเป็นต้องใช้ วงจรแปลงสัญญาณดิจิตอลให้เป็นอนาลอกเข้ามาช่วยดังรูปที่ 4.18 ซึ่งมีหลักการทำงานดังนี้

1.) เมื่อสัญญาณเอาท์พุทของวงจรนับเลขฐานสองถูกเซ็ทให้เป็น 0 สวิตช์ SI จะปิดวงจรโดยวงจรสวิตช์ไดรฟ์ ทำหน้าที่ควบคุมการปิด-เปิดของสวิตช์ สัญญาณเอาท์พุทของวงจรอินทริเกรเตอร์จะเพิ่มขึ้นเรื่อย ๆ (Positive - Going Ramp) เข้าหา e_{in} ที่มีค่าเป็นลบ

2.) เมื่อสัญญาณเอาท์พุทของวงจรอินทริเกรเตอร์ ผ่านเข้าในวงจรเปรียบเทียบแรงดัน จะได้สัญญาณเอาท์พุทที่ออกจากวงจรมีค่าเป็นลบ ($-e_c$)

3.) สัญญาณเอาท์พุทของวงจรเปรียบเทียบแรงดัน จะถูกป้อนเข้าอินเวอร์เตอร์ (Invertor) เพื่อทำให้สัญญาณเอาท์พุทที่ออกมามีค่าเป็นบวก คือเป็นระดับสูง

4.) สัญญาณเอาท์พุทของวงจรเปรียบเทียบแรงดันจะนำไปเปิดแนนด์เกท ซึ่งอีกขาหนึ่งแนนด์เกทนี้ต่อเข้ากับเอาท์พุทที่ได้จากแนนด์เกทจะนำไปควบคุมวงจรนับเลขฐานสองให้ทำงาน

5.) เมื่อวงจรนับเลขฐานสองทำงาน จนกระทั่งสัญญาณเอาท์พุทของทุกบิตมีค่าเป็น 1 ยกเว้นบิตที่มีเลขนัยสำคัญสูงสุด สัญญาณพัลส์ลูกต่อไปจะทำการเปลี่ยนค่าบิตทุกบิต โดยจะใส่ 1 ที่บิตที่มีเลขนัยสำคัญสูงสุด และใส่ 0 ในบิตที่เหลือ

6.) ซึ่งช่วงเวลาที่ใช้ในการเปลี่ยนแปลงเอาท์พุท ของวงจรนับเลขฐานสอง ตั้งแต่ 000...000 ไปจนถึง 011..111 มีค่าคงที่คือ T1

7.) ในระหว่างช่วงเวลา T1 สัญญาณเอาท์พุทของวงจรอินทริเกรเตอร์จะมีค่าเป็น $e_{in}T1/RC$ และในพัลส์ต่อมาจะทำให้ สัญญาณเอาท์พุทของวงจรนับเลขฐานสองจะเพิ่มขึ้นจาก 011..111 เป็น 100...000

8.)เมื่อบิตที่มีเลขนัยสำคัญสูงสุดมีค่าเป็น 1 จะทำให้วงจรสวิตช์ไดร์ฟ เปิดสวิตช์ S1 และปิดสวิตช์ S2 จึงทำให้เกิดการป้อนสัญญาณ V_- เข้าที่ขาอินพุทของวงจรอินทิเกรเตอร์

9.) สัญญาณเอาท์พุทจากวงจรอินทริเกรเตอร์ จะเปลี่ยนลดลงเรื่อย ๆ (Negative - Going Ramp) ถ้าสัญญาณอินพุทเป็นบวก เมื่อสัญญาณเอาท์พุทของวงจรอินทิเกรเตอร์ต่ำกว่า 0 V สัญญาณเอาท์พุทของวงจรเปรียบเทียบแรงดันจะมีค่าเป็นบวก

ทำให้สัญญาณเอาท์พุทจากอินเวอร์เตอร์ เปลี่ยนเป็นระดับต่ำ จึงเป็นการปิดแนนด์เกท ทำให้วงจรกำเนิดพัลส์ และวงจรนับเลขฐานสอง หยุดทำงาน ซึ่งวงจรนับเลขฐานสองก็จะให้ได้สัญญาณดิจิตอลเอาท์พุทออกมา

10.) ถ้า V_r ที่ให้เข้ามาทำให้เกิดช่วงเวลา T2 สัญญาณเอาท์พุทของวงจรเปรียบเทียบแรงดัน สามารถบอกได้ด้วยช่วงระยะเวลา T2 ตั้งแต่เริ่มระยะเวลา T2 ซึ่งสัญญาณเอาท์พุทจากวงจรนับเลขฐานสองมีค่าเป็น 100...000

11.) ในช่วงเวลา T2 การเปลี่ยนแปลงสัญญาณเอาท์พุทของวงจรอินทิเกรเตอร์มีค่าเป็น $V_r(T2/RC)$ ซึ่งสามารถคำนวณหา e_{in} ได้จาก

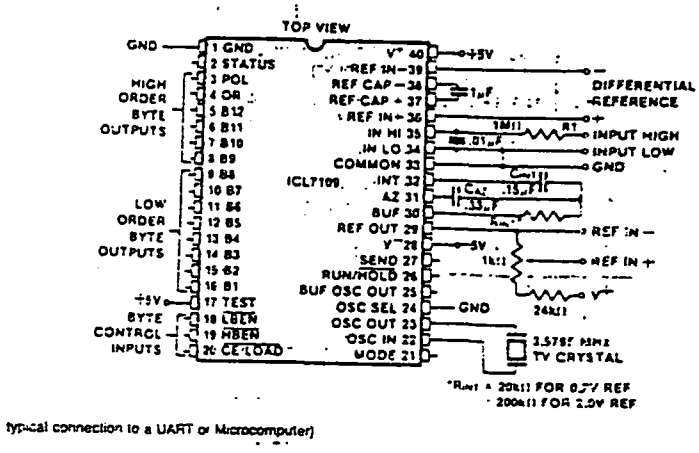
$$e_{in}T1/RC = V_rT2/RC$$

$$e_{in} = V_rT2/T1$$

เนื่องจาก Vr และ T1 เป็นค่าคงที่ ดังนั้น e_{in} มีค่าเปลี่ยนแปลงตาม T2

12.) เนื่องจาก T2 สามารถบอกค่าสัญญาณเอาท์พุทของวงจรนับเลขฐานสอง ซึ่งก็สามารถบอกค่าสัญญาณอนาลอก e_{in} ด้วย

ในส่วนของ ADC ที่ใช้ในโครงงานพิเศษนี้ใช้ ADC เบอร์ ICL7109 ซึ่งเป็น ADC ชนิด Dual integrating ขนาด 12 bit โดย ICL7109 เป็น IC CMOS ซึ่งกินกำลังไฟต่ำ และสามารถติดต่อกับไมโครโปรเซสเซอร์ได้ง่าย โดยข้อมูลเอาท์พุท (12 bit,bit polarity และ bit overange) สามารถควบคุมได้โดยขาอินพุท 2 ขา และ chip select input สำหรับการติดต่อแบบง่าย ๆ และ ยังมีการ hand shake แบบ UART อีกแบบหนึ่ง รูปที่ 4.19 แสดงการต่อวงจรทดสอบ ของ ICL7109



typical connection to a UART or Microcomputer)

รูปที่ 4.19 แสดงการต่อวงจรทดสอบของ ICL7109

หน้าที่การทำงานของขาต่าง ๆ ของ ICL7109 แสดงดังตาราง 4.5

Pin	Symbol	Description		
1	GND	Digital Ground, 0V. Ground return for all digital logic.		
2	STATUS	Output High during integrate and deintegrate until data is latched. Output Low when analog section is in Auto-Zero configuration		
3	POL	Polarity — HI for Positive input.		
4	OR	Overrange — HI if Overranged.		
5	B12	Bit 12	(Most Significant Bit)	All three state output data bits
6	B11	Bit 11		
7	B10	Bit 10		
8	B9	Bit 9		
9	B8	Bit 8		
10	B7	Bit 7	HI = true	
11	B6	Bit 6		
12	B5	Bit 5		
13	B4	Bit 4		
14	B3	Bit 3		
15	B2	Bit 2		
16	B1	Bit 1	(Least Significant Bit)	
17	TEST	Input High — Normal Operation. Input Low — Forces all bit outputs high. Note: This input is used for test purposes only. Tie high if not used.		
18	$\overline{LBEN}$	Low Byte Enable — With Mode (Pin 21) low, and $\overline{CE/LOAD}$ (Pin 20) low, taking this pin low activates low order byte outputs B1 — B8. — With Mode (Pin 21) high, this pin serves as a low byte flag output used in handshake mode. See Figures 8, 9, 10.		
19	$\overline{HBEN}$	High Byte Enable — With Mode (Pin 21) low, and $\overline{CE/LOAD}$ (Pin 20) low, taking this pin low activates high order byte outputs B9 — B12, POL, OR. — With Mode (Pin 21) high, this pin serves as a high byte flag output used in handshake mode. See Figures 8, 9, 10.		
20	$\overline{CE/LOAD}$	Chip Enable Load — With Mode (Pin 21) low, $\overline{CE/LOAD}$ serves as a master output enable. When high, B1 — B12, POL, OR outputs are disabled. — With Mode (Pin 21) high, this pin serves as a load strobe used in handshake mode. See Figures 8, 9, 10.		

Pin	Symbol	Description
21	MODE	Input Low — Direct output mode where $\overline{CE/LOAD}$ (Pin 20), $\overline{HBEN}$ (Pin 19) and $\overline{LBEN}$ (Pin 18) act as inputs d... enabling byte outputs. Input Pulsed High — Causes immediate entry into handshake mode and output of data as in Figure 10. Input High — Enables $\overline{CE/LOAD}$ (Pin 20), $\overline{HBEN}$ (Pin 19), and $\overline{LBEN}$ (Pin 18) as outputs, handshake mode will be entered and data output as in Figures 8 and 9 at ... conversion completion
22	OSC IN	Oscillator Input
23	OSC OUT	Oscillator Output
24	OSC SEL	Oscillator Select — Input high configures OSC IN, OSC OUT, BUF OSC OUT as RC oscillator — clock will be same phase and duty cycle as BUF OSC OUT. — Input low configures OSC IN, OSC OUT for crystal oscillator — clock frequency will be 1/58 of frequency at BUF OSC OUT.
25	BUF OSC OUT	Buffered Oscillator Output
26	RUN/$\overline{HOLD}$	Input High — Conversions continuously performed every 8192 clock pulses. Input Low — Conversion in progress completed, converter will step in Auto-Zero 7 counts before integrate.
27	SEND	Input — Used in handshake mode to indicate ability of an external device to accept data. Connect to + 5V if not used.
28	V−	Analog Negative Supply — Nominally − 5V with respect to GND (Pin 1).
29	REF OUT	Reference Voltage Output — Nominally 2.8V down from V+ (Pin 40).
30	BUFFER	Buffer Amplifier Output
31	AUTO-ZERO	Auto-Zero Node — Inside foil of C_{AZ}
32	INTEGRATOR	Integrator Output — Outside foil of C_{INT}
33	COMMON	Analog Common — System is Auto-Zeroed to COMMON
34	INPUT LO	Differential Input Low Side
35	INPUT HI	Differential Input High Side
36	REF IN +	Differential Reference Input Positive
37	REF CAP +	Reference Capacitor Positive
38	REF CAP −	Reference Capacitor Negative
39	REF IN −	Differential Reference Input Negative
40	V+	Positive Supply Voltage — Nominally + 5V with respect to GND (Pin 1).

ตารางที่ 4.5 แสดงหน้าที่การทำงานของขาต่าง ๆ ของ ICL7109

ในส่วนอนาลอก

ICL7109 จะทำงานเมื่อป้อนไฟ +5 V และ -5 V โดยจะสามารถแปลงค่าได้สูงสุด 4 V full scale ทั้งทางด้านบวกและลบ ในการเลือกค่าอุปกรณ์เพิ่มเติมที่ต้องต่อกับ ICL7109 มีดังนี้

- INTEGRATING REGISTOR สำหรับ 4.096 volt full scale ใช้ค่า 200 K สำหรับ 409.6 mV full scale ใช้ค่า 20 K แต่สำหรับค่า full scale อื่น ๆ หาค่า R_{INT} ได้จาก

$$R_{INT} = \frac{\text{full scale voltage}}{20\ A}$$

- INTEGRATING CAPACITOR สำหรับ ICL7109 ที่จ่ายไฟ +5 V และ analog common ต่อลง GND สำหรับการแปลงข้อมูลด้วยอัตรา 7.5 ครั้งต่อวินาที (ความถี่นาฬิกา 61.72 KHz) จะใช้ C_{INT} และ C_{AZ} เท่ากับ 0.15 F และ 0.33 ตามลำดับสำหรับความถี่นาฬิกาอื่น ๆ สามารถหาค่า C_{INT} ได้จาก

$$C_{INT} = \frac{(2048 * \text{clock period})(20\ A)}{\text{integrator output voltage swing}}\ F$$

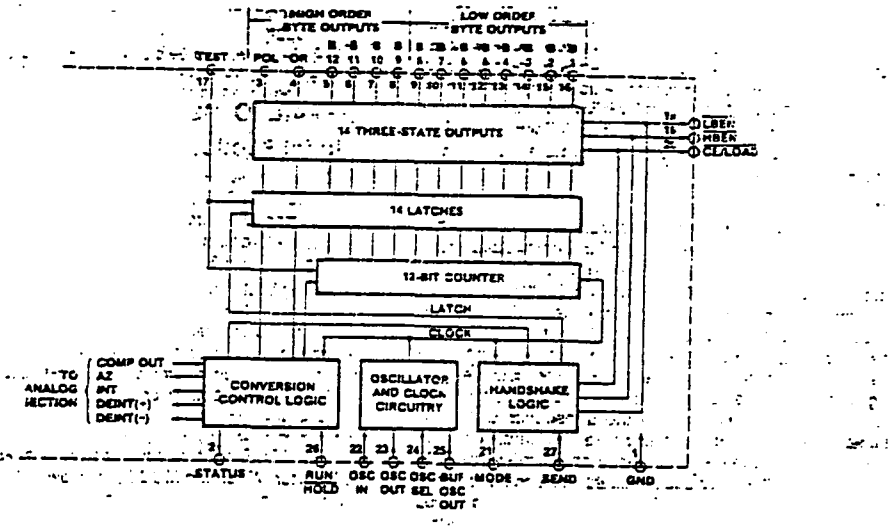
- AUTO-ZERO สามารถหาค่าได้จาก

$$C_{AZ} = 2C_{INT}$$

- REFERENCE CAPACITOR ให้ใช้ค่าตัวเก็บประจุเท่ากับ 1 F

- REFERENCE VOLTAGE ในการกำหนดค่า reference Voltage นั้นจะขึ้นอยู่กับค่า input voltage full scale โดย $V_{in} = 2V_{REF}$ เช่นถ้าใช้ 4.096 V full scale จะให้ $V_{REF} = 2.048$ V
- REFERENCE SOURCE REFERENCE VOLTAGE นั้นจะต้องมีค่าเสถียร เพื่อจะให้การแปลงสัญญาณเป็นไปอย่างถูกต้องที่สุด ซึ่ง ICL7109 มีขา reference output (ขา 29) เพื่อสร้าง reference voltage ขึ้นโดยขานี้จะรับกระแสเข้าได้สูงสุดประมาณ 20 mA โดยปกติขานี้จะให้ voltage เท่ากับ 2.8 V ซึ่งจะต้องต่อขา REF OUT เข้ากับขา REF- และ REF+ ต่อกับขากลางของ ตัวต้านทานปรับค่าได้ โดยวงจรสำหรับ reference 204.8 ใช้ตามวงจรทดสอบ สำหรับ reference 2.048 V นำตัวต้านทานคงที่ออกและใช้ ตัวต้านทานปรับค่าได้

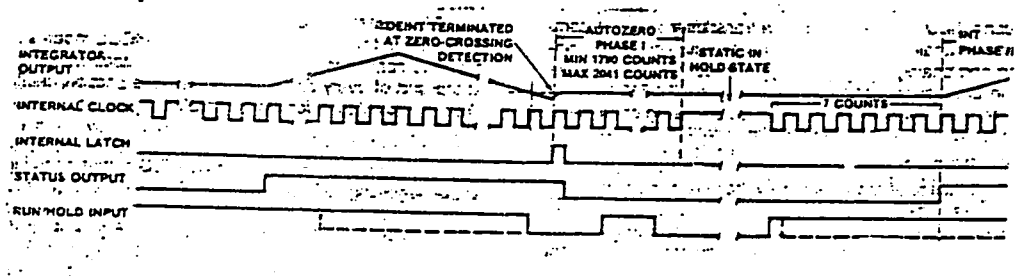
ในส่วนดิจิตอล



รูปที่ 4.20 แสดงส่วน Digital section

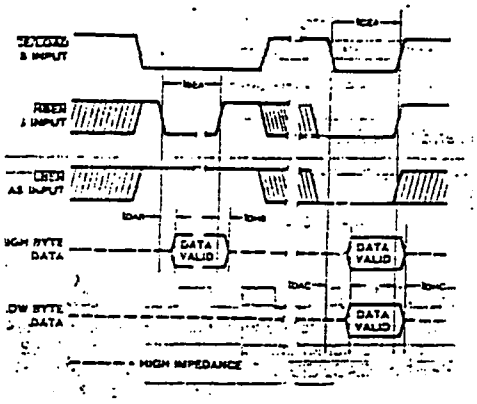
ในส่วนนี้จะมีส่วน ภาคกำเนิดสัญญาณนาฬิกา (click oscillator) , สัญญาณข้อมูลต่าง ๆ และสัญญาณควบคุมต่าง ๆ ด้วย ดังนี้

- MODE INPUT ใช้เลือกโหมดในการแปลง เมื่อขา mode เป็น low หรือ open ตัวแปลงสัญญาณ จะเป็นโหมด direct เมื่อให้เป็น pulse high ตัวแปลงสัญญาณจะเป็นโหมด UART handshake และข้อมูลเอาท์พุทเป็นแบบ 2 ไบท์ จากนั้นจะกลับไปเป็นแบบโหมด direct เมื่อให้โหมดเป็น high ตัวแปลงสัญญาณจะให้ข้อมูลเอาท์พุทในแบบโหมด handshake ในทุก ๆ วงรอบของการแปลงเสร็จ
- STATUS OUTPUT ระหว่างวงรอบการแปลงสัญญาณ status จะเป็น high ตอนเริ่มแรกและจะตกเป็น low หลังจากข้อมูลใหม่ที่ได้จากการแปลงสัญญาณ
- RUN/HOLD INPUT เมื่ออินพุทเป็น high หรือ open วงจรจะทำการแปลงข้อมูลอย่างต่อเนื่องถ้า RUN/HOLD เป็น low ที่เวลาใดระหว่างการดีอินทิเกรด วงจรจะไปทำงานแบบ auto-zero แทนคือจะไม่แปลงข้อมูลใหม่ ตามรูปที่ 4.21



รูปที่ 4.21 แสดงการทำงานของ RUN/HOLD

- DIRECT MODE เมื่อ mode เป็น low จะเป็นการติดต่อกับไมโครโปรเซส เซอร์แบบโดยตรง ดังแสดงดังรูป 4.22 และตารางที่ 4.6

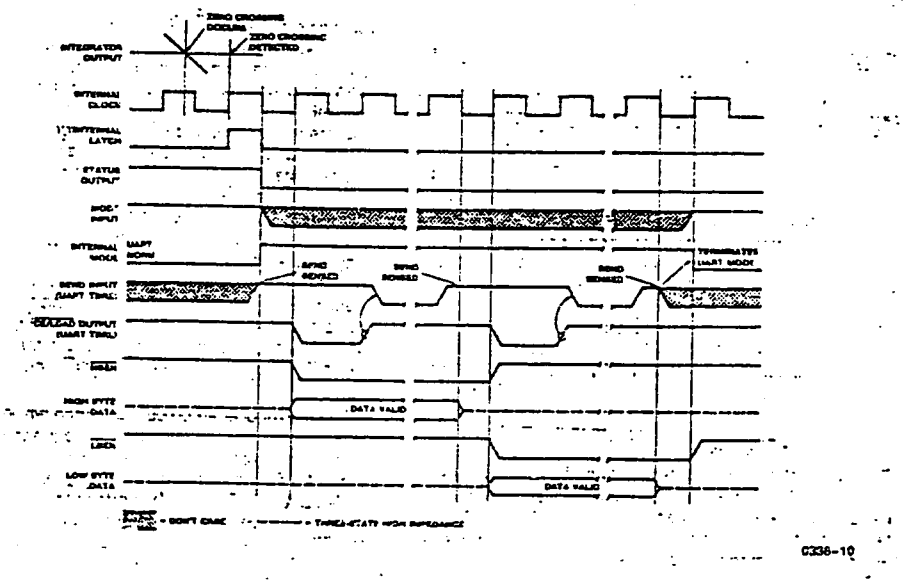


รูปที่ 4.22 แสดง TIMING DIAGRAM ของการติดต่อโดยตรง

SYMBOL	DESCRIPTION	MIN	TYP	MAX	UNIT
t_{BEA}	Byte Enable Width	350	220		ns
t_{OAB}	Data Access Time from Byte Enable		210	350	ns
t_{OHB}	Data Hold Time from Byte Enable		150	300	ns
t_{CEA}	Chip Enable Width	400	260		ns
t_{DAC}	Data Access Time from Chip Enable		260	400	ns
t_{OHC}	Data Hold Time from Chip Enable		240	400	ns

ตารางที่ 4.6 แสดงเวลาของขนาดสัญญาณต่าง ๆ ของโหมดโดยตรง

- HANDSHAKE MODE เป็นการติดต่อกับไมโครโปรเซสเซอร์ที่จะต้องมีการทำ handshake ซึ่งมีอยู่หลายแบบด้วยกันดังแสดงดังรูป 4.23



รูปที่ 4.23 แสดง TIMING DIAGRAM ของการติดต่อแบบ HANDSHAKE

- OSCILLATOR ICL7109 สามารถที่จะทำวงจร oscillator ที่มีอยู่ภายในโดยต่ออุปกรณ์ภายนอกโดยเลือกได้ 2 แบบคือ ใช้ RC หรือใช้ crystal ซึ่งเลือกใช้ oscillator select เป็นตัวเลือกโดยเมื่อเป็น high หรือ open จะเป็นแบบ RC โดยต่อวงจรดังรูปที่ 4.24

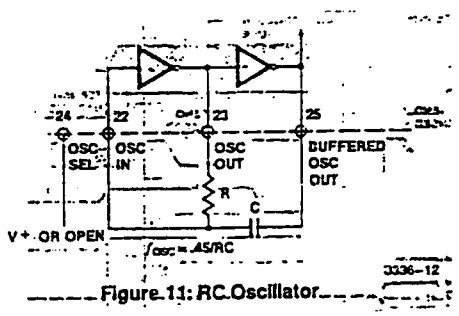


Figure 11: RC Oscillator

รูปที่ 4.24 แสดงการต่อวงจร OSCILLATOR แบบใช้ RC

โดยสามารถหาค่า C ได้จาก

$$f = 0.45/(RC)$$

$$f = 8192/\text{เวลาของการแปลง 1 ครั้ง}$$

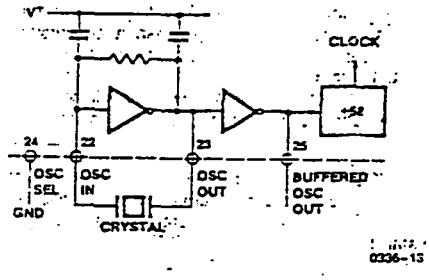
ให้ใช้ $R = 100\ K$

เมื่อ oscillator select เป็น low จะใช้ crystal ตั้งแต่ 1-5 MHz ซึ่งความถี่ที่ crystal จะถูกหารด้วย 58 ซึ่งถ้าใช้การแปลงในอัตรา 7.5 ครั้งต่อวินาทีจะใช้ crystal ความถี่ 3.579 MHz ซึ่งสามารถหาเวลาในการ integration ได้จาก

$$T_{INT} = (2048\ \text{clock periods})*(T_{clock})$$

โดย $T_{clock} = 58/3.579\ MHz$

โดยต่อวงจร oscillator ที่ใช้ crystal ตามรูป 4.25

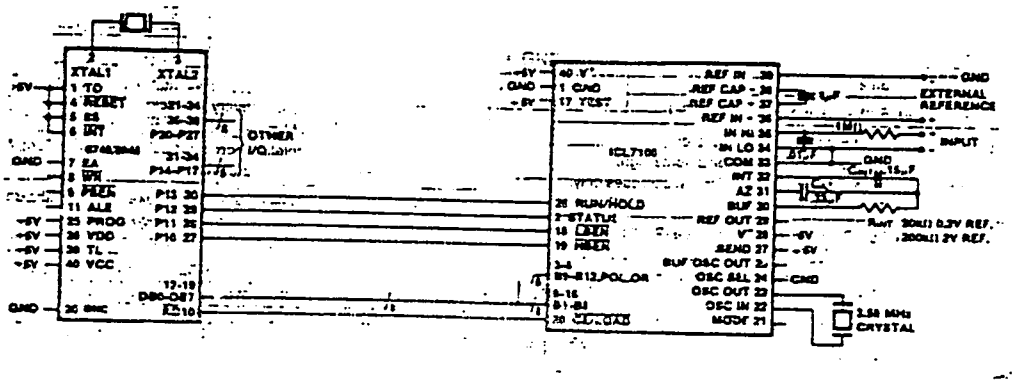


รูปที่ 4.25 แสดงการต่อวงจร OSCILLATOR ที่ใช้ CRYSTAL

- TEST INPUT เมื่อ test input มีระดับแรงดันเป็นครึ่งหนึ่งของ V^+ เมื่อเทียบ GND ทำให้ counter output latch จะทำงานซึ่งสามารถทดสอบได้ทุกเวลา เมื่อ test ต่อ GND จะทำให้ counter output เป็น high state และ internal clock จะไม่ทำงาน และเมื่อลินพุทกลับไปเป็นครึ่งหนึ่งของ v^+ เทียบ GND และให้ clock เข้าไป 1 ลูก counter output จะกลับเป็น low ทำให้ตรวจสอบได้ง่าย

- การต่อวงจรแปลงสัญญาณอะนาลอกเป็นสัญญาณดิจิตอลที่ใช้เบอร์ ICL7109

โดยต่อวงจรตามรูปที่ 4.26

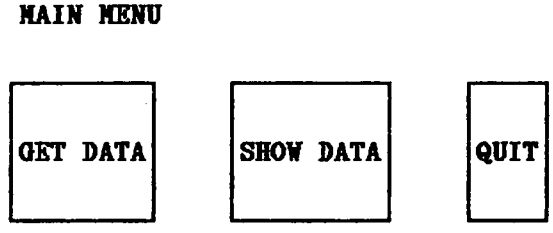


รูปที่ 4.26 แสดงการต่อ ICL7109

บทที่ 5

การใช้งานโปรแกรมในส่วนของการรับข้อมูลและแสดงผล

เมื่อทำการเรียกโปรแกรมในส่วนของการรับข้อมูลและแสดงผล จอภาพจะปรากฏภาพดังรูปที่ 5.1



รูปที่ 5.1 ภาพในส่วนการรับข้อมูลและแสดงผล

ทำการเลือกแถบสว่างเพื่อทำงานโดยใช้ปุ่มลูกศรซ้ายและขวาในการเลือกเพื่อเลือกว่าจะทำงานในส่วนใดแล้วกดปุ่ม ENTER

ถ้าเลือก GET DATA

จะขึ้นแถบสว่างให้ใส่ชื่อFile โดยชื่อ File จะไม่เกิน 8 ตัวอักษรโดยเริ่มจากการกด SPACE BAR ก่อนแล้วจึงใส่ชื่อ File ที่ต้องการเก็บข้อมูลลงไป เมื่อทำเสร็จแล้วกด ENTER แล้วทำการเลื่อนแถบสว่างโดยใช้ปุ่ม ลูกศรขึ้นและลง ในการเลื่อนขึ้นลงเพื่อทำการควบคุมแกนของตัว Detector ให้หมุนไปทางซ้ายหรือขวา ถ้าเลือกตัวเลข 1 แกนของตัว Detector จะหมุนไปทางขวา แต่ถ้าเลือกตัวเลข 2 จะหมุนไปทางซ้าย แล้วจึงกดปุ่ม ENTER คอมพิวเตอร์จะเริ่มทำการควบคุม Stepping Motor และทำการรับข้อมูล ของตำแหน่งและค่าของความต่างศักย์ เปลี่ยนแปลงตามมุมต่าง ๆ และแสดงขึ้นหน้าจอ เมื่อเก็บข้อมูลหมดแล้ว หน้าจอจะขึ้น

"SUCCESS PRESS ANY KEY TO CONTINUE"

เมื่อกดปุ่มใด ๆ แล้วก็จะกลับมาที่ MAIN MENU

ถ้าเลือก SHOW DATA

จะขึ้นแถบสว่างให้ใส่ชื่อ Path ที่มีชื่อ File ที่เราเก็บข้อมูล อยู่แล้วกด ENTER แล้วทำการเลื่อนแถบสว่างโดยใช้ปุ่ม ลูกศรขึ้นและลง ในการเลื่อนขึ้นและลงเพื่อใส่ชื่อ File ที่มีข้อมูล ที่เราเก็บไว้อยู่และกด ENTER ก็จะแสดงกราฟระหว่าง ตำแหน่ง กับความต่างศักย์ เมื่อต้องการออกจาก MENU SHOWDATA ให้กดปุ่มใด ๆ แล้วกลับมาที่ MAIN MENU

ถ้าเลือก QUIT

จะแสดงหน้าจอให้เลือก Y หรือ N ในการออกจากโปรแกรม ถ้าเลือก Y จะออกจากโปรแกรม แต่ถ้าเลือก N ถ้าจะอยู่ที่ MAIN MENU ในการเลือก MENU QUIT จะเหมือนกับการกดปุ่ม ESC

บทที่ 6

ผลการวิจัย

6.1 ขั้นตอนการวิจัย

1.ทำการตั้งเครื่อง Goniometer และคอมพิวเตอร์

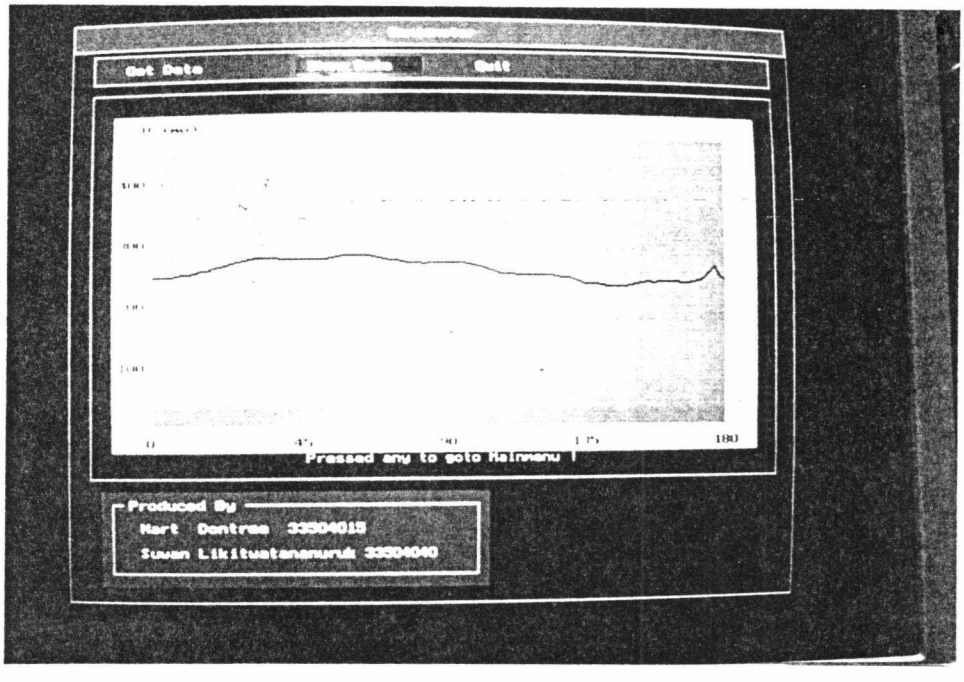
2.นำแหล่งกำเนิดที่ต้องการวัดความเข้มแสง ณ องศาต่าง ๆ มาติดตั้ง

3.เริ่มทำการหมุนแกนทีละสเตป สเตปละ 1 องศา ไปจนถึง 180 องศา

4.ทำการเก็บข้อมูลเข้าคอมพิวเตอร์และทำการสร้างกราฟ

6.2 ผลการวิจัย

จากการวิจัยเราจะได้ค่าองศากับค่าความเข้มแสงเมื่อนำมาเขียนกราฟ โดยที่แกน X เป็นค่าตำแหน่งองศาต่าง ๆ และแกน Y เป็นค่าความเข้มแสงจะได้กราฟ ดังรูปที่ 6.1



รูปที่ 6.1 ผลการวิจัย

บทที่ 7

สรุปผลการศึกษาและข้อเสนอแนะ

7.1 สรุปผลการศึกษา

จากผลการทดลองพบว่า ผลที่ได้เป็นที่น่าพอใจในระดับหนึ่งคือสามารถได้ค่าองศาและค่าความเข้มแสง แต่ค่าองศาที่ได้นั้นยังไม่ค่อยแน่นอนเนื่องมาจากอุปกรณ์ทางเครื่องกล เช่น แขน ที่ยาวเกินไปทำให้เกิดการแกว่ง จานถอดรหัสที่ไม่ได้สมมาตรทำให้เกิดแรงเหวี่ยงขณะที่มันหมุน

ซึ่งสิ่งเหล่านี้เป็นผลทำให้เกิดการแกว่งของจานถอดรหัส ทำให้การถอดรหัสค่าองศามีการเคลื่อนพอสมควร

จากเครื่อง Goniometer นี้เราสามารถนำไปประยุกต์ใช้ในการวัดค่าองศากับพารามิเตอร์ อื่น ๆ อีกเช่น ความดังของเสียง การแผ่รังสี ที่ตำแหน่งของมุมต่าง ๆ กัน

7.2 ข้อเสนอแนะ

จากการสั่นของจานถอดรหัส ทำให้การถอดรหัสตำแหน่งนั้นคาดเคลื่อนดังนั้นเราควรที่จะลดการสั่นของมัน ซึ่งพอสรุปได้เป็นข้อ ๆ ได้ดังนี้

1. ลดความยาวของแขนที่ติดตัวเซลแสงอาทิตย์
2. ทำจานถอดรหัสให้สมมาตรเพื่อลดแรงเหวี่ยง
3. นำอุปกรณ์ที่มีความฝืดเล็กน้อยเพื่อมาเป็นตัวยึดจานถอดรหัสให้แกว่งน้อยลง

ภาคผนวก ก

โปรแกรมเก็บข้อมูลและแสดงผล

```
Program Gonio;
Uses Crt,Dos,Graph;
Type Menu = Array[1..3] Of String;
     String50 = String[50];
     Ar_Init = Array[1..4]  Of Byte;
     Ar_Data = Array[1..4] Of Byte;
     DatType = Array [1..100] Of Record
     Posit : Word;
     Value : Word;
     End;
Const  MaxMenu = 3;
       Item : Menu = ('Get Data',
                      'Show Data',
                      'Quit');

       PA  = $0300;  PB  = $0301;
       PC  = $0302;  PCC = $0303;
       Data_Out : Ar_Data = ($10,$20,$40,$80);

Var d,dir,i,j,k,n,o,p,q,r,x,y,z,OldX,OldI,ForG,BacG : Integer;
    Color,Size : Word;
    Mono,VgaMono,TisMode,Check   : Boolean;
    Index,Ind,A,B : Byte; Pt : Pointer;
    Ch : Char; Msg  : String50;
    CurX,CurY,NoStr,Code : Integer;
    Sentence,Data1 : String[8]; Num : String[2];
```

```
    Data : Word;
    Fo : File Of DatType;
    Indexs : Dattype;
    Fi : File Of Word;

            {********** Title ************}

Procedure Opengraph;
Var GrDriver,GrMode : Integer;
Begin
  Mono := False; VgaMono := False;
  Color := 3;
  DetectGraph(GrDriver,GrMode);
   Case GrDriver Of
     5 : VgaMono := True;
     7 : Begin Mono := True; Color := 1; End;
     9 : GrMode := 2;
   End;
  InitGraph(GrDriver,GrMode,'');
End;

Procedure OutTxt(x,y : Integer;Size : Byte;Color : Word; Msg : String);
Begin
  SetColor(Color); SetTextStyle(0,0,Size);
  OutTextXY(x,y,Msg);
End;
```

```
Procedure Frame0(x1,y1,x2,y2 : Integer; BackGr : Word);
Begin
  SetFillStyle(1,BackGr);  Bar(x1,y1,x2,y2);
  SetColor(15); Rectangle(x1,y1,x2,y2);
End;

Procedure Frame1(x1,y1,x2,y2 : Integer; BackGr : Word);
Begin
  SetFillStyle(1,BackGr); Bar(x1,y1,x2-10,y2-10);
  SetColor(0); Rectangle(x1+5,y1+5,x2-15,y2-15);
  SetFillStyle(1,0);
  Bar(x2-10,y1+10,x2,y2); Bar(x1+10,y2-10,x2,y2);
  SetColor(15); Rectangle(x1,y1,x2-10,y2-10);
End;

Procedure SubTit(x1,y1,x2,y2,Space : Integer; Msg : String);
Begin
  SetColor(15); Rectangle(x1,y1,x2,y2);
  SetFillStyle(1,9); Bar(x1+10,y1-10,x1+Space,y1+5);
  OutTxt(x1+15,y1-3,1,14,Msg); SetColor(15);
End;

Procedure Key;
Begin
  SubTit(30,320,380,370,40,'Key');
  OutTxt(40,330,1,14,'[Esc]          [Enter] ');
```

```
  OutTxt(40,330,1,15,'      - Quit          - Continue');
  OutTxt(40,350,1,14,'[Left] & [Right] ');
  OutTxt(40,350,1,15,'                  Arrow - Select other menu');
End;

Procedure Title;
Begin
  Frame0(0,0,GetMaxX,GetMaxY,1); Frame0(0,0,GetMaxX,25,13);
  OutTxt(270,10,1,15,'Goniometer');
  SetFillStyle(1,9); Bar(20,30,620,55);
  SetFillStyle(1,9);
  Bar(30,390,360,465); Rectangle(20,30,620,55);
  Bar(20,65,620,380);  Rectangle(20,65,620,380);
  SubTit(40,400,350,455,110,'Produced By');
  OutTxt(65,415,1,15,'Nart  Dontree  33504015');
  OutTxt(65,435,1,15,'Suwan Likitwatananuruk 33504040');
End;

Procedure Describe1;
Begin
  SetFillStyle(1,9); Bar(30,75,610,305);
  SubTit(30,80,610,305,85,'Get Data');
End;

Procedure Describe2;
Begin
```

```
  SetFillStyle(1,9); Bar(30,75,610,305);
  SubTit(30,80,610,305,90,'Show Data');
End;

Procedure Describe3;
Begin
  SetFillStyle(1,9); Bar(30,75,610,305);
  SubTit(30,80,610,305,50,'Quit');
  OutTextXY(45,100,'If you want to return to Dos , Pressed [Enter] to continue !');
  OutTextXY(45,120,'And confirm this command by select ');
  OutTextXY(45,140,'  - (Y)es - To goto Dos.');
  OutTextXY(45,160,'  - (N)o  - To return to mainmenu.');
End;

            {********* End of Title ********}

Procedure CtlText(x,y : Integer; Forg,Bacg : Byte; Msg : String);
Var PosX : Integer;
Begin
  SetFillStyle(1,Bacg); SetColor(Forg);
  OutTextXY(x,y,Msg);
End;

Function PopUp(x,y,Forg,Bacg : Integer; Item : Menu): Byte;

Procedure ChangeMenu;
```

```
Begin
  SetFillStyle(1,Bacg); Bar(x-10,y-5,x+100,y+10);
  SetFillStyle(1,9);    Bar(OldX-10,y-5,OldX+100,y+10);
  CtlText(OldX,y,Forg,Bacg,Item[OldI]);
  CtlText(x,y,ForG,BacG,Item[Index]);
  SetFillStyle(1,Bacg);
End;


Begin
  Repeat
   Ch := Readkey;
     If Ch = #0 Then
      Begin
        Ch := Readkey;
        Case Ch Of
          #75 : Begin
                  OldI := Index; Index := Index - 1;
                  If Index = 0 Then Index := 3;
                  x := 50 + ((Index - 1) * 150); OldX := x + 150;
                  If x = 350 Then OldX := 50;
                  ChangeMenu;
                End;
          #77 : Begin
                  x := 50 + (Index * 150); OldX := x - 150;
                  OldI := Index; Index := Index + 1;
                  If OldX = 350 Then x := 50;
```

```
                If Index = 4 Then Index := 1; ChangeMenu;
              End
          Else If (Index <> OldI) Then  z := 1;
        End;
      End Else z := 1;
    If (Index <> OldI) And (z = 0) Then
      Case Index Of
        1 : Describe1;  2 : Describe2;
        Else Describe3;
      End;
      z := 0;
  Until (Ch = #13) Or (Ch = #27);
    If (Index = 1) And (Ch = #13) Then PopUp := 1 Else
    If (Index = 2) And (Ch = #13) Then PopUp := 2 Else
    If (Ch = #27) Or ((Index = 3) And (Ch = #13)) Then PopUp := 0;
End;

Procedure SaveScreen(x1,y1,x2,y2 : Integer);
Begin
  Size := ImageSize(x1,y1,x2,y2);
  GetMem(Pt,Size);
  GetImage(x1,y1,x2,y2,Pt^);
End;

Procedure RestoreScreen(x1,y1 : Integer);
Begin
```

```
  PutImage(x1,y1,Pt^,0);
End;

Procedure CursorOn;
Begin
  SetFillStyle(1,15); Bar(CurX,CurY+5,CurX+5,CurY+4);
End;

Procedure CursorOff;
Begin
  SetFillStyle(1,BacG); Bar(CurX,CurY+5,CurX+5,CurY+4);
End;

Procedure InputName;
Begin
  CursorOn; SetColor(15); NoStr := 0; Sentence := '';
  SetFillStyle(1,BacG);
  Repeat
   Ch := Readkey;
   If (Ch = #8) And (NoStr <> 0) Then
    Begin
      CursorOff; CurX := CurX - TextWidth(Ch);
      Bar(CurX,CurY+5,CurX+7,CurY-3); CursorOn;
      Delete(Sentence,NoStr,1);  NoStr := NoStr - 1;
    End;
    If (NoStr < 8) And (Ch <> #8) And (Ch <> #13) And
```

```
        (Ch <> #0) And (Ch <> #27) Then
        Begin
         CursorOff; OutTextXY(CurX,CurY-2,Ch);
         CurX := CurX + TextWidth(Ch); CursorOn;
         Sentence := Sentence + Ch;  NoStr := NoStr + 1;
        End;
  Until Ch = #13;
  CursorOff; p := 1;
End;

Procedure InputNum;
Begin
  CursorOn; SetColor(15); NoStr := 0; Num := '';
  SetFillStyle(1,BacG);
  Repeat
   Ch := Readkey;
   If (Ch = #8) And (NoStr <> 0) Then
    Begin
     CursorOff; CurX := CurX - TextWidth(Ch);
     Bar(CurX,CurY+5,CurX+7,CurY-3);
     CursorOn; Delete(Num,NoStr,1);
     NoStr := NoStr - 1;
    End;
   If (NoStr < 3) And (Ch in ['0'..'9']) Then
    Begin
     CursorOff; OutTextXY(CurX,CurY-2,Ch);
```

```
      CurX := CurX + TextWidth(Ch);
      CursorOn; Num := Num + Ch;
      NoStr := NoStr + 1;
     End;
  Until Ch = #13;
  CursorOff;
  CurX := x; CurY := y; p := 1;
End;

        {************** Get Data **************}

Procedure Get_Data;
Var x,y,OldY,Index,OldIndex,a,b,d : Integer;
    Filename : String[12];
    RotNum : Integer;
    Data2  : String[20];
    positStr,ValStr : String;

Procedure WriteData;
Begin
  If (p <> 0) Then
   Begin
     SetColor(15);
     If Data1 <> '' Then
       Begin
         OutTextXY(140,110,Filename);
       End;
     If Num <> '' Then
```

```
        Begin
          OutTextXY(140,140,Data2);
        End;
    End;
End;

Procedure GetData;
Var a,b,d,Fase : Integer;
    c : Byte;

Procedure ChangeData;
Var f1,f2 : byte;
    Mode,Base,Degree : Integer;
Begin
     f1 := c Shr 4;
     f2 := c And $0f;
     If f1 = 00 Then
     Begin
     Mode := 0;
     Base := 0;
     End
     Else If f1 = 1 Then
     Begin
     Mode := 1;
     Base := 16;
     End
     Else If f1 = 3 Then
     Begin
```

```
Mode := 0;
Base := 32;
End
Else If f1 = 2 Then
Begin
Mode := 1;
Base := 48;
End
Else If f1 = 6 Then
Begin
Mode := 0;
Base := 64;
End
Else If f1 = 4 Then
Begin
Mode := 1;
Base := 80;
End
Else If f1 = 5 Then
Begin
Mode := 0;
Base := 96;
End
Else If f1 = 7 Then
Begin
mode := 1;
```

```
      Base :=  112;
      End;
      If f2 = 0  Then Degree := 1  Else
      If f2 = 1  Then Degree := 2  Else
      If f2 = 3  Then Degree := 3  Else
      If f2 = 2  Then Degree := 4  Else
      If f2 = 6  Then Degree := 5  Else
      If f2 = 4  Then Degree := 6  Else
      If f2 = 5  Then Degree := 7  Else
      If f2 = 7  Then Degree := 8  Else
      If f2 = 15 Then Degree := 9  Else
      If f2 = 14 Then Degree := 10 Else
      If f2 = 10 Then Degree := 11 Else
      If f2 = 8  Then Degree := 12 Else
      If f2 = 12 Then Degree := 13 Else
      If f2 = 13 Then Degree := 14 Else
      If f2 = 9  Then Degree := 15 Else
      If f2 = 11 Then Degree := 16 ;
      fase := 0;
      If Mode = 1 Then
      Fase := Base + (17 - Degree) Else
      Fase := Base + Degree;

End;

Begin
```

```
  Setfillstyle(1,9);
  Port[$305] := $00;
  a := Port[$304];
  b := Port[$306];
  c := Port[pb];
  ChangeData;
  d := a + (b*16);
  Data :=  i;                          { Fase; }
  Write(fi,Data);
  Str(Data,positStr);
  Data := d;
  Write(Fi,Data);
  Str(Data,ValStr);
  Rectangle(200,260,450,300);
  Bar(205,262,445,295);
  OutTextXY(220,276,' position = '+positstr+'  Data = '+ValStr);
  Port[pa] := Data_Out[j];
  Port[$305] := $01;
  Delay(500);
End;

Begin
  SetFillStyle(1,9); Bar(35,85,605,300); Bar(35,325,375,365);
  SubTit(40,200,380,245,65,'Status');
  OutTxt(45,110,1,14,'Filename :'); OutTxt(45,140,1,14,'Rotate   :');
  OutTxt(55,170,1,14,'( 1 - Right Rotation , 2 - Left Rotation )');
```

```
OutTxt(40,330,1,14,'[Esc]          [Spacebar] ');
OutTxt(40,330,1,15,'       - Quit              - Continue');
OutTxt(40,350,1,14,'[Up] & [Down] ');
OutTxt(40,350,1,15,'               Arrow - Select choice');
x := 135; y := 105; CurX := 140; CurY := 110;
SetFillStyle(1,BacG); Bar(x,y,250,y+13);
Index := 1; z := 0; Data1 := ''; Data2 := '';
Repeat
  Ch := Readkey;
  If Ch = #0 Then
    Begin
     Ch := Readkey;
     Case Ch Of
      #72 : Begin
              Index := Index - 1; OldIndex := Index;
              OldY := y; y := y - 30;
              If Index = 0 Then
               Begin
                 Index := 2; y := 135; OldY := 105;
               End;
              SetFillStyle(1,BacG); Bar(x,y,250,y+13);
              SetFillStyle(1,9); Bar(x,OldY,250,OldY+13);
              WriteData;
            End;
      #80 : Begin
              Index := Index + 1; OldIndex := Index;
```

```
            OldY := y; y := y + 30;
            If Index = 3 Then
             Begin
               Index := 1; y := 105; OldY := 135;
             End;
            SetFillStyle(1,BacG); Bar(x,y,250,y+13);
            SetFillStyle(1,9); Bar(x,OldY,250,OldY+13);
            WriteData;
          End;
    End;
  End;
If Ch = #32 Then
  Case Index Of
   1 : Begin
         CurX := 140; CurY := 110;
         SetFillStyle(1,BacG); Bar(135,105,250,118);
         InputName; Data1 := Sentence;
         Filename := Sentence + '.goi';
       End;
   2 : Begin
         Repeat
           CurX := 140; CurY := 140;
           SetFillStyle(1,BacG); Bar(135,135,250,148);
           InputNum; Val(Num,RotNum,Code);
           Case RotNum Of
            1 : Data2 := 'Right Rotate';
```

```
            2 : Data2 := 'Left  Rotate';
           End;
         Until (RotNum = 1) Or (RotNum = 2);
       End;
  End;
If (Data1 <> '') And (Data2 <> '') Then
 Begin
   OutTxt(60,220,1,14,'Getting data Now ! Please , Wait.');
   Port[$307] := $99;
   Port[pcc]  := $82;
   Port[$305] := $01;
   Assign (fi,Filename);
   Rewrite (fi);
   Case RotNum Of
    1 : Begin
          i := 0;
          For i:= 1 To 99 Do
           Begin
             j := j + 1;
             If  j > 4 Then j := 1;
             GetData;
           End;
          Setfillstyle(1,9);
          Port[$305] := $00;
          a := Port[$304];
          b := Port[$306];
```

```
        d := a + (b*16);
        Data :=100;
        write(fi,Data);
        Str(Data,PositStr);
        Data := d;
        Write(Fi,Data);
        Str(Data,ValSTR);
        Rectangle(200,260,450,300);
        bar(205,262,445,295);
        Outtextxy(220,276,' Position = '+positstr+'  Data= '+ValStr);
        Port[$305] := $01;
      End;
  2 : Begin
       i := 0;
       For i:= 1 To 99 Do
        Begin
          j := j - 1;
          If j < 1 Then j := 4;
          GetData;
        End;
     Setfillstyle(1,9);
     Port[$305] := $00;
     a := Port[$304];
     b := Port[$306];
     d := a + (b*16);
     Data :=100;
```

```
            write(fi,Data);
            Str(Data,PositStr);
            Data := d;
            Write(Fi,Data);
            Str(Data,ValSTR);
            Rectangle(200,260,450,300);
            bar(205,262,445,295);
            Outtextxy(220,276,' Position = '+positstr+'  Data= '+ValStr);
            Port[$305] := $01;
            End;
        End;
        Port[pa] := $00;
        Close(Fi);
        SetFillStyle(1,9); Bar(45,205,375,240);
        OutTxt(60,220,1,14,'Success ! Pressed any key to continue.');
        Ch := Readkey; Ch := #27;
      End;
   Until Ch = #27;
   If Ch = #27 Then
    Begin
      Describe1; Bar(35,325,375,365); Key;
    End;
End;

        {********** End Of Scan ***********}
```

```
        {************ Show Data ***********}


Procedure Show_Data;
Var Path,PathFile : String;
    x,y,OldY,Index,OldIndex : Integer;
    Data : String[8]; DirInfo : SearchRec;
    Filename : String[12];


Procedure ShowData;
Var  i,y,x : Integer;
Begin
     Assign(Fo,Filename);
     {$I-}
     Reset(Fo);
     {$I+}
     If IOResult <> 0 Then
      Begin
        Halt(0);
        Writeln('Error opening file : ',Filename);
      End;
     Read(Fo,Indexs);
     Close(fo);
     SetFillStyle(1,1); Bar(20,65,620,380);
     SetColor(15);
     Rectangle(20,65,620,380);
     Setfillstyle(1,15); bar(40,85,600,358);
```

```
      Setfillstyle(1,7);
      bar(70,105,570,338); Setcolor(0);
      Outtextxy(65,93,'V (mv)'); Outtextxy(45,138,'200');
      Outtextxy(45,238,'100');   Outtextxy(45,288,' 50');
      Outtextxy(45,188,'150');
      Outtextxy(560,348,'180  '); Outtextxy(315,348,'90');
      Outtextxy(190,348,'45'); Outtextxy(435,348,'135');
      Outtextxy(70,348,'0');
      MoveTo(70+(5*Indexs[1].Posit),338-Round(Indexs[1].Value));
      SetColor(0);
      i := 1;
      For i := 1 To 100 Do
      Begin
       x := 70 + (5*Indexs[i].Posit);
       y := 338 - Round(Indexs[i].Value/11);
       LineTo(x,y);
       End;
       OutTxt(200,360,1,14,'Pressed any to goto Mainmenu !');
       Ch := Readkey; Ch := #27;
       SetFillStyle(1,9); Bar(20,65,620,380);
       Rectangle(20,65,620,380);
End;

Procedure WriteData1;
Begin
  SetColor(15);
```

```
  If Path <> '' Then
   Begin
     OutTextXY(140,110,Path);
   End;
  If (p <> 0) Then
   If Filename  <> '' Then OutTextXY(140,140,Filename);
End;

Begin
  SetFillStyle(1,9); Bar(35,85,605,300); Bar(35,325,375,365);
  SubTit(40,180,300,225,65,'Status');
  OutTxt(40,330,1,14,'[Esc]          [Spacebar] ');
  OutTxt(40,330,1,15,'      - Quit              - Continue');
  OutTxt(40,350,1,14,'[Up] & [Down] ');
  OutTxt(40,350,1,15,'               Arrow - Select choice');
  OutTxt(45,110,1,14,'Path     :');
  OutTextXY(45,140,'Filename :');
  x := 135; y := 105; CurX := 140; CurY := 110;
  SetFillStyle(1,BacG); Bar(x,y,250,y+13);
  Index := 1; z := 0; Filename := '';
  GetDir(0,Path); OutTxt(140,110,1,15,Path);
  Repeat
    Ch := Readkey;
    If Ch = #0 Then
     Begin
       Ch := Readkey;
```

```
  Case Ch Of
   #72 : Begin
           Index := Index - 1; OldIndex := Index;
           OldY := y; y := y - 30;
           If Index = 0 Then
            Begin
              Index := 2; y := 135; OldY := 105;
            End;
           SetFillStyle(1,BacG); Bar(x,y,250,y+13);
           SetFillStyle(1,9); Bar(x,OldY,250,OldY+13);
           WriteData1;
         End;
   #80 : Begin
           Index := Index + 1; OldIndex := Index;
           OldY := y; y := y + 30;
           If Index = 3 Then
            Begin
              Index := 1; y := 105; OldY := 135;
            End;
           SetFillStyle(1,BacG); Bar(x,y,250,y+13);
           SetFillStyle(1,9); Bar(x,OldY,250,OldY+13);
           WriteData1;
         End;
  End;
End;
If Ch = #32 Then
```

```
Case Index Of
  1 : Begin
       SetFillStyle(1,Bacg); Bar(135,105,250,118);
       CurX := 140; CurY := 110; SetColor(15);
       CursorOn; NoStr := 0; Path := '';
       Repeat
        Ch := Readkey;
        If (Ch = #8) And (NoStr <> 0) Then
         Begin
           CursorOff; CurX := CurX - TextWidth(Ch);
           Bar(CurX,CurY+5,CurX+7,CurY-3);
           CursorOn; Delete(Path,NoStr,1);
           NoStr := NoStr - 1;
         End;
       If (NoStr < 15) And (Ch <> #8) And (Ch <> #13) And
        (Ch <> #0) And (Ch <> #27) Then
        Begin
           CursorOff; OutTextXY(CurX,CurY-2,Ch);
           CurX := CurX + TextWidth(Ch);
           CursorOn; Path := Path + Ch;
           NoStr := NoStr + 1;
        End;
      Until Ch = #13;
      CursorOff; CurX := x; CurY := y; p := 1;
     End;
  2 : Begin
```

```
            SetColor(15);
            SetFillStyle(1,Bacg); Bar(135,135,250,148);
            CurX := 140; CurY := 140; InputName; Data := Sentence;
            Filename := Data + '.goi';
          End;
   End;  {End of Index}
  If (Path <> '') And (Filename <> '') And
     (Ch = #13) Then
  Begin
   If (Path = 'A:\') Or (Path = 'B:\') Or (Path = 'a:\') Or
      (Path = 'b:\') Then Delete(Path,3,1);
  PathFile := Path + '\' + Filename;
  FindFirst(PathFile,Archive,DirInfo);
  Case DosError Of
    0 : Begin
          OutTxt(60,200,1,14,'Loading !  Please , Wait !');
          ShowData;
        End;
    3 : Begin
          OutTxt(60,200,1,14,'Path Not Found ! Try again.');
          Ch := Readkey;
          SetFillStyle(1,9); Bar(45,185,275,220);
        End;
  2,18 : Begin
          OutTxt(60,200,1,14,'File Not Found ! Try again.');
          Ch := Readkey;
```

```
                SetFillStyle(1,9); Bar(45,185,275,220);
              End
          Else Begin
                OutTxt(60,200,1,14,'Error ! Error ! Try again.');
                Ch := Readkey;
                SetFillStyle(1,9); Bar(45,185,275,220);
              End;
        End;
      End;
  Until Ch = #27;
  If Ch = #27 Then
   Begin
     Describe2; Bar(35,325,375,365); Key;
   End;
End;

        {***********End Of Show Data **********}

          {************** Quit *************}

Procedure Quit;
Begin
  SaveScreen(240,200,440,280);
  Frame1(240,200,440,280,4);
  OutTxt(278,220,1,15,' EXIT PROGRAM ');
  OutTxt(260,240,1,15,'Are You Sure ? (Y/N)');
```

```
  Repeat
   Ch := Readkey;
   If UpCase(Ch) = 'Y' Then
    Begin
      Check := True; Exit;
    End;
   If UpCase(Ch) = 'N' Then
    Begin
      RestoreScreen(240,200);
      FreeMem(Pt,ImageSize(240,200,440,280));
      i := 5;
    End
  Until (UpCase(Ch) = 'Y') Or (UpCase(Ch) = 'N');
End;

          {************* End of Quit ***************}

            {***************MAIN********************}

Begin
 ClrScr; Opengraph;
 Title;
 Check := False; Index := 1; Describe1; Key;
 x := 50; y := 40; z := 0; ForG := 15; BacG := 2;
 SetFillStyle(1,Bacg); Bar(x-10,y-5,x+100,y+10);
 For i := 1 To MaxMenu Do
```

```
    Begin
      CtlText(x,y,Forg,Bacg,Item[i]); x := x + 150;
    End;
    Repeat
      A := PopUp(x,y,ForG,BacG,Item);
      Case A Of
        0 : Quit;
        1 : Get_Data;
        2 : Show_Data;
        3 : Quit;
      End;
   Until Check = True; Delay(500);
 Closegraph; Delay(500);
 Writeln('CopyRight (c) Applied Physics KMIT'#39'L 1993-1994. ');
 Writeln('Thanks !! See You Again. ');
End.
```

ภาคผนวก ข

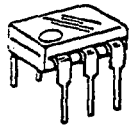
รายการข้อมูลอุปกรณ์ทางอิเลคทรอนิกส์

MOTOROLA
SEMICONDUCTOR
TECHNICAL DATA

6-Pin DIP Optoisolators
Transistor Output

4N35
4N36
4N37

6-PIN DIP OPTOISOLATORS TRANSISTOR OUTPUT

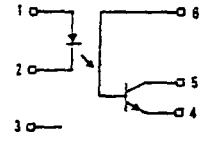


CASE 730A-02
PLASTIC

These devices consist of a gallium arsenide infrared emitting diode optically coupled to a monolithic silicon phototransistor detector.

- Convenient Plastic Dual-In-Line Package
- High Current Transfer Ratio — 100% Minimum at Spec Conditions
- Guaranteed Switching Speeds
- High Input-Output Isolation Guaranteed — 7500 Volts Peak
- UL Recognized, File Number E54915
- VDE approved per standard 0883/6.80 (Certificate number 41853), with additional approval to DIN IEC380/VDE0806, IEC435/VDE0805, IEC65/VDE0860, VDE0110b, covering all other standards with equal or less stringent requirements, including IEC204/VDE0113, VDE0160, VDE0832, VDE0833, etc.
- Meets or Exceeds All JEDEC Registered Specifications
- Special lead form available (add suffix "T" to part number) which satisfies VDE0883/6.80 requirement for 8 mm minimum creepage distance between input and output solder pads.
- Various lead form options available. Consult "Optoisolator Lead Form Options" data sheet for details.

SCHEMATIC



1. LED ANODE
2. LED CATHODE
3. N.C.
4. EMITTER
5. COLLECTOR
6. BASE

MAXIMUM RATINGS (T_A = 25°C unless otherwise noted)

Rating	Symbol	Value	Unit
INPUT LED			
Reverse Voltage	V_R	6	Volts
Forward Current — Continuous	I_F	60	mA
LED Power Dissipation @ T_A = 25°C with Negligible Power in Output Detector	P_D	120	mW
Derate above 25°C		1.41	mW/°C
OUTPUT TRANSISTOR			
Collector-Emitter Voltage	V_{CEO}	30	Volts
Emitter-Base Voltage	V_{EBO}	7	Volts
Collector-Base Voltage	V_{CBO}	70	Volts
Collector Current — Continuous	I_C	150	mA
Detector Power Dissipation @ T_A = 25°C with Negligible Power in Input LED	P_D	150	mW
Derate above 25°C		1.76	mW/°C
TOTAL DEVICE			
Isolation Source Voltage (1) (Peak ac Voltage, 60 Hz, 1 sec Duration)	V_{ISO}	7500	Vac
Total Device Power Dissipation @ T_A = 25°C	P_D	250	mW
Derate above 25°C		2.94	mW/°C
Ambient Operating Temperature Range	T_A	−55 to +100	°C
Storage Temperature Range	T_{stg}	−55 to +150	°C
Soldering Temperature (10 seconds, 1/16" from case)	—	260	°C

(1) Isolation surge voltage is an internal device dielectric breakdown rating. For this test, Pins 1 and 2 are common, and Pins 4, 5 and 6 are common.

6

ELECTRICAL CHARACTERISTICS (T_A = 25°C unless otherwise noted)

Characteristic		Symbol	Min	Typ	Max	Unit
INPUT LED						
Forward Voltage (I_F = 10 mA)	T_A = 25°C T_A = −55°C T_A = 100°C	V_F	0.8 0.9 0.7	1.15 1.3 1.05	1.5 1.7 1.4	V
Reverse Leakage Current (V_R = 6 V)		I_R	—	—	10	μA
Capacitance (V = 0 V, f = 1 MHz)		C_J	—	18	—	pF
OUTPUT TRANSISTOR						
Collector-Emitter Dark Current (V_{CE} = 10 V, T_A = 25°C) (V_{CE} = 30 V, T_A = 100°C)		I_{CEO}	— —	1 —	50 500	nA μA
Collector-Base Dark Current (V_{CB} = 10 V)	T_A = 25°C T_A = 100°C	I_{CBO}	— —	0.2 100	20 —	nA
Collector-Emitter Breakdown Voltage (I_C = 1 mA)		$V_{(BR)CEO}$	30	45	—	V
Collector-Base Breakdown Voltage (I_C = 100 μA)		$V_{(BR)CBO}$	70	100	—	V
Emitter-Base Breakdown Voltage (I_E = 100 μA)		$V_{(BR)EBO}$	7	7.8	—	V
DC Current Gain (I_C = 2 mA, V_{CE} = 5 V)		h_{FE}	—	400	—	—
Collector-Emitter Capacitance (f = 1 MHz, V_{CE} = 0)		C_{CE}	—	7	—	pF
Collector-Base Capacitance (f = 1 MHz, V_{CB} = 0)		C_{CB}	—	19	—	pF
Emitter-Base Capacitance (f = 1 MHz, V_{EB} = 0)		C_{EB}	—	9	—	pF
COUPLED						
Output Collector Current (I_F = 10 mA, V_{CE} = 10 V)	T_A = 25°C T_A = −55°C T_A = 100°C	I_C	10 4 4	30 — —	— — —	mA
Collector-Emitter Saturation Voltage (I_C = 0.5 mA, I_F = 10 mA)		$V_{CE(sat)}$	—	0.14	0.3	V
Turn-On Time	(I_C = 2 mA, V_{CC} = 10 V, R_L = 100 Ω, Figure 11)	t_{on}	—	7.5	10	μs
Turn-Off Time		t_{off}	—	5.7	10	
Rise Time		t_r	—	3.2	—	
Fall Time		t_f	—	4.7	—	
Isolation Voltage (f = 60 Hz, t = 1 sec)		V_{ISO}	7500	—	—	Vac(pk)
Isolation Current (V_{I-O} = 3550 Vpk) (V_{I-O} = 2500 Vpk) (V_{I-O} = 1500 Vpk)	4N35 4N36 4N37	I_{ISO}	— — —	— — 8	100 100 100	μA
Isolation Resistance (V = 500 V)		R_{ISO}	10^{11}	—	—	Ω
Isolation Capacitance (V = 0 V, f = 1 MHz)		C_{ISO}	—	0.2	2	pF

TYPICAL CHARACTERISTICS

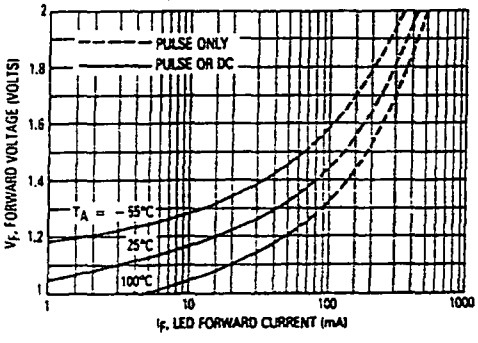


Figure 1. LED Forward Voltage versus Forward Current

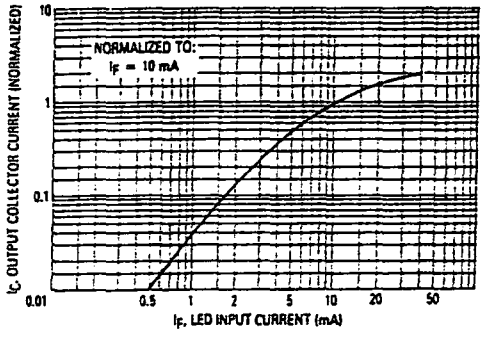


Figure 2. Output Current versus Input Current

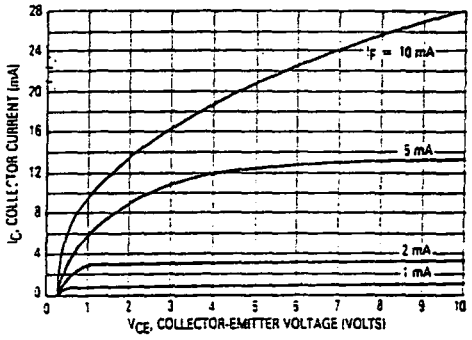


Figure 3. Collector Current versus Collector-Emitter Voltage

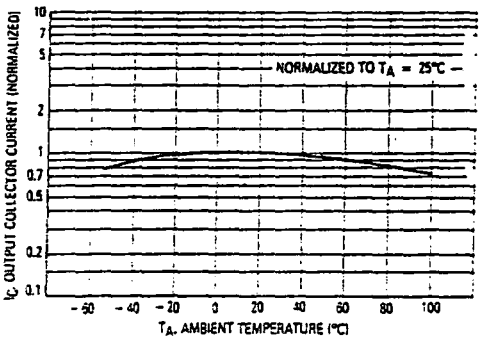


Figure 4. Output Current versus Ambient Temperature

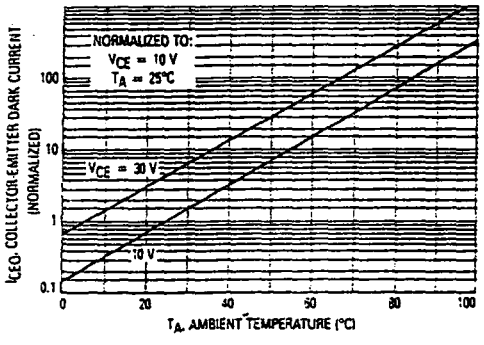


Figure 5. Dark Current versus Ambient Temperature

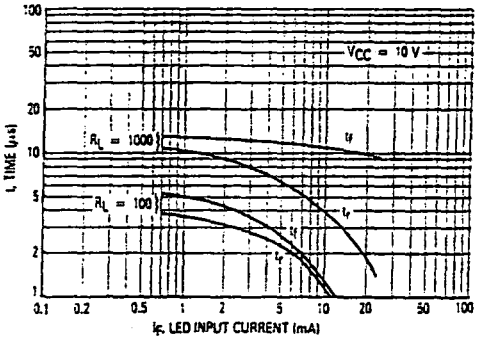


Figure 6. Rise and Fall Times

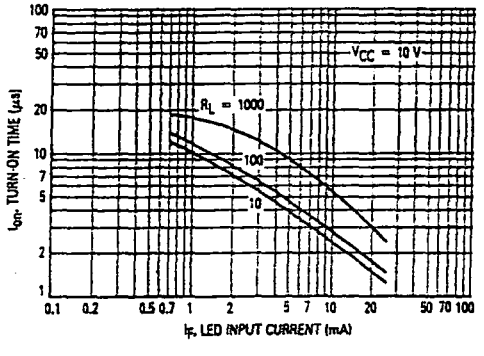


Figure 7. Turn-On Switching Times

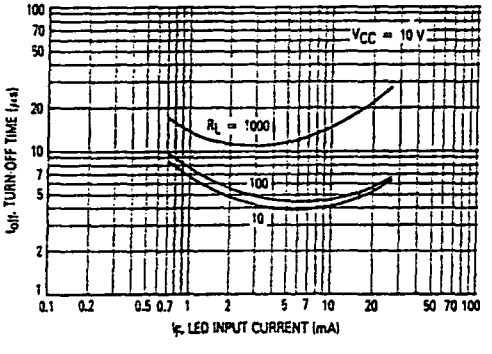


Figure 8. Turn-Off Switching Times

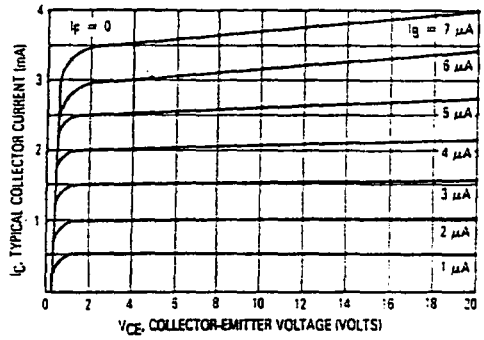


Figure 9. DC Current Gain (Detector Only)

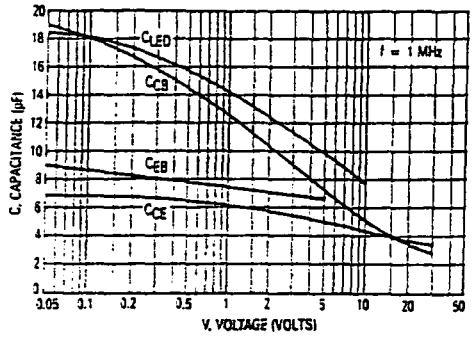


Figure 10. Capacitances versus Voltage

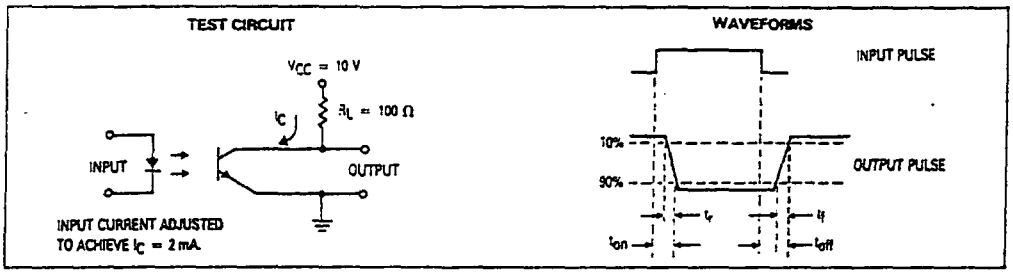
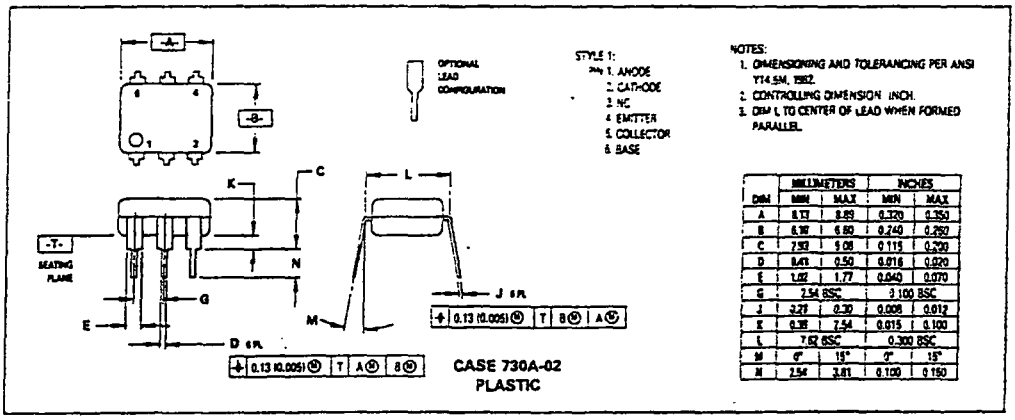


Figure 11. Switching Times

OUTLINE DIMENSIONS



DIM	MILLIMETERS MIN	MILLIMETERS MAX	INCHES MIN	INCHES MAX
A	8.13	8.89	0.320	0.350
B	6.10	6.60	0.240	0.260
C	2.92	5.08	0.115	0.200
D	0.41	0.50	0.016	0.020
E	1.02	1.77	0.040	0.070
G	2.54 BSC		0.100 BSC	
J	0.23	0.30	0.009	0.012
K	0.38	7.54	0.015	0.100
L	7.62 BSC		0.300 BSC	
M	0°	15°	0°	15°
N	2.54	3.81	0.100	0.150

6

HARRIS SEMICONDUCTOR

ICL7109
12-Bit μP-Compatible A/D Converter

GENERAL DESCRIPTION

The ICL7109 is a high performance, CMOS, low power integrating A/D converter designed to easily interface with microprocessors.

The output data (12 bits, polarity and overrange) may be directly accessed under control of two byte enable inputs and a chip select input for a simple parallel bus interface. A UART handshake mode is provided to allow the ICL7109 to work with industry-standard UARTs in providing serial data transmission, ideal for remote data logging applications. The RUN/HOLD input and STATUS output allow monitoring and control of conversion timing.

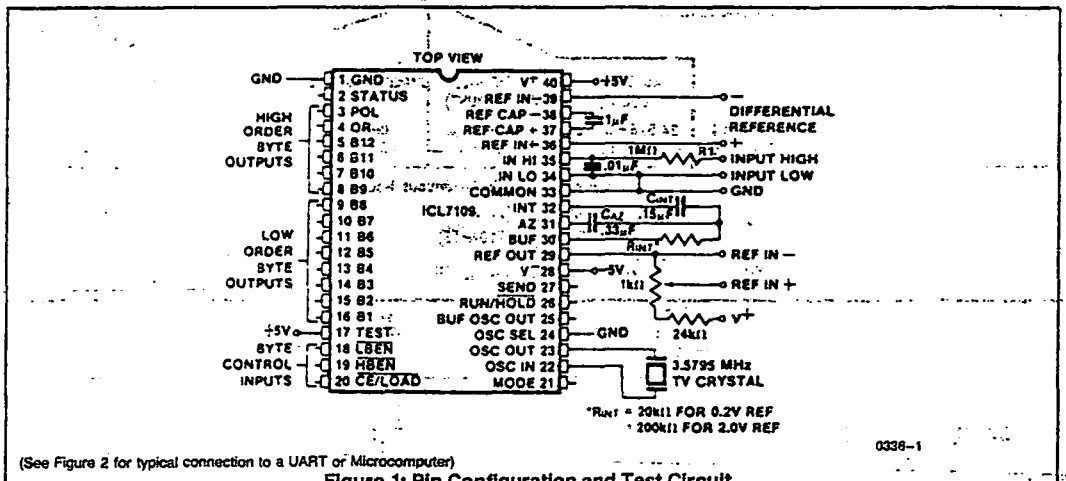
The ICL7109 provides the user with the high accuracy, low noise, low drift, versatility and economy of the dual-slope integrating A/D converter. Features like true differential input and reference, drift of less than 1μV/°C, maximum input bias current of 10pA, and typical power consumption of 20mW make the ICL7109 an attractive per-channel alternative to analog multiplexing for many data acquisition applications.

FEATURES

- **12 Bit Binary (Plus Polarity and Overrange) Dual Slope Integrating Analog-to-Digital Converter**
- **Byte-Organized TTL-Compatible Three-State Outputs and UART Handshake Mode for Simple Parallel or Serial Interfacing to Microprocessor Systems**
- **RUN/HOLD Input and STATUS Output Can Be Used to Monitor and Control Conversion Timing**
- **True Differential Input and Differential Reference**
- **Low Noise — Typically 15μV p-p**
- **1pA Typical Input Current**
- **Operates At Up to 30 Conversions Per Second**
- **On-Chip Oscillator Operates With Inexpensive 3.58MHz TV Crystal Giving 7.5 Conversions Per Second for 60Hz Rejection May Also Be Used With An RC Network Oscillator for Other Clock Frequencies**

ORDERING INFORMATION

Part Number	Temp. Range	Package
ICL7109MDL	−55°C to +125°C	40-Pin Ceramic DIP
ICL7109IDL	−25°C to +85°C	40-Pin Ceramic DIP
ICL7109IJL	−25°C to +85°C	40-Pin CERDIP
ICL7109CPL	0°C to 70°C	40-Pin Plastic DIP



(See Figure 2 for typical connection to a UART or Microcomputer)

Figure 1. Pin Configuration and Test Circuit

4.19

HARRIS SEMICONDUCTOR'S SOLE AND EXCLUSIVE WARRANTY OBLIGATION WITH RESPECT TO THIS PRODUCT SHALL BE THAT STATED IN THE WARRANTY ARTICLE OF THE CONDITION OF SALE. THE WARRANTY SHALL BE EXCLUSIVE AND SHALL BE IN LIEU OF ALL OTHER WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, INCLUDING THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR USE.

NOTE: All typical values have been characterized but are not tested.

301655-003

ABSOLUTE MAXIMUM RATINGS

Positive Supply Voltage (GND to V+) ..... +6.2V
Negative Supply Voltage (GND to V−) ..... −9V
Analog Input Voltage (Lo or Hi) (Note 1) ..... V+ to V−
Reference Input Voltage (Lo or Hi) (Note 1) ..... V+ to V−
Digital Input Voltage ..... V+ + 0.3V
(Pins 2-27) (Note 2) ..... GND − 0.3V

Power Dissipation (Note 3)
Ceramic Package ..... 1W @ +85°C
Plastic Package ..... 500mW @ +70°C
Operating Temperature
Ceramic Package (MDL) ..... −55°C to +125°C
Ceramic Package (IDL) ..... −25°C to +85°C
Plastic Package (CPL) ..... 0°C to +70°C
Storage Temperature ..... −65°C to +150°C
Lead Temperature (Soldering, 10sec) ..... +300°C

NOTE: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. These are stress ratings only and functional operation of the device at these or any other conditions above those indicated in the operational sections of the specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

ELECTRICAL CHARACTERISTICS

(V+ = +5V, V− = −5V, GND = 0V, T_A = 25°C, f_{CLK} = 3.58 MHz, unless otherwise indicated.) Test circuit as shown on first page of this data sheet.

ANALOG SECTION

Symbol	Parameter	Test Conditions	Min	Typ	Max	Unit
	Zero Input Reading	V_{IN} = 0.0000V V_{REF} = 204.8 mV	−0000	±0000	+0000	Counts
	Ratiometric Error(4)	V_{IN} = V_{REF} = 204.8 mV	−3		0	Counts
	Non-Linearity (Max deviation from best straight line fit)	Full Scale = 409.6mV to 2.048V Over full operating temperature range. (Note 4), (Note 6)	−1	±.2	+1	Counts
	Roll-over Error (difference in reading for equal pos. and neg. inputs near full scale)	Full Scale = 409.6mV to 2.048V (Note 5), (Note 6)	−1	±.2	+1	Counts
CMRR	Common Mode Rejection Ratio	V_{CM} ±1V, V_{IN} = 0V Full Scale = 409.6mV		50		µV/V
VCMR	Input Common Mode Range	Input Hi, Input Lo, Common (Note 4)	V− +1.5		V+ −1.0	V
e_n	Noise (p-p value not exceeded 95% of time)	V_{IN} = 0V Full Scale = 409.6mV		15		µV
I_{ILK}	Leakage current at Input	V_{IN} = 0 All devices at 25°C ICL7109CPL, 0°C ≤ T_A ≤ +70°C (Note 4) ICL7109IDL, −25°C ≤ T_A ≤ +85°C (Note 4) ICL7109MDL, −55°C ≤ T_A ≤ +125°C		1 20 100 2	10 100 250 5	pA pA pA nA
	Zero Reading Drift	V_{IN} = 0V R_1 = 0Ω (Note 4)		0.2	1	µV/°C
	Scale Factor Temperature Coefficient	V_{IN} = 408.9mV = >7770 reading Ext. Ref. 0 ppm/°C (Note 4)		1	5	ppm/°C
I+	Supply Current V+ to GND	V_{IN} = 0, Crystal Osc 3.58MHz test circuit		700	1500	µA
I_{SUPP}	Supply Current V+ to V−	Pins 2-21, 25, 26, 27, 29; open		700	1500	µA
V_{REF}	Ref Out Voltage	Referred to V+, 25kΩ between V+ and REF OUT	−2.4	−2.8	−3.2	V
	Ref Out Temp. Coefficient	25kΩ between V+ and REF OUT		80		ppm/°C

NOTE: All typical values have been characterized but are not tested.

ELECTRICAL CHARACTERISTICS ($V^+ = +5V$, $V^- = -5V$, GND = 0V, $T_A = 25°C$, unless otherwise indicated.) Test circuit as shown on first page of this data sheet. (Continued)

DIGITAL SECTION

Symbol	Parameter		Test Conditions	Min	Typ	Max	Unit
V_{OH}	Output High Voltage		$I_{OUT} = 100\mu A$ Pins 2-16, 18, 19, 20	3.5	4.3		V
V_{OL}	Output Low Voltage		$I_{OUT} = 1.6mA$		0.2	0.4	V
	Output Leakage Current		Pins 3-16 high impedance		±.01	±1	µA
	Control I/O Pullup Current		Pins 18, 19, 20 $V_{OUT} = V^+ - 3V$, MODE input at GND		5		µA
	Control I/O Loading		HBEN Pin 19 LBEN Pin 18 (Note 4)			50	pF
V_{IH}	Input High Voltage		Pins 18-21, 26, 27 referred to GND	3.0			V
V_{IL}	Input Low Voltage		Pins 18-21, 26, 27 referred to GND			1	V
	Input Pull-up Current		Pins 26, 27 $V_{OUT} = V^+ - 3V$		5		µA
	Input Pull-up Current		Pins 17, 24 $V_{OUT} = V^+ - 3V$		25		µA
	Input Pull-down Current		Pin 21 $V_{OUT} = GND + 3V$		5		µA
$-O_{OH}$	Oscillator Output Current	High	$V_{OUT} = 2.5V$		1		mA
O_{OL}		Low	$V_{OUT} = 2.5V$		1.5		mA
BO_{OH}	Buffered Oscillator Output Current	High	$V_{OUT} = 2.5V$		2		mA
$-BO_{OL}$		Low	$V_{OUT} = 2.5V$		5		mA
t_W	MODE Input Pulse Width		(Note 4)	50			ns

NOTES: 1. Input voltages may exceed the supply voltages provided the input current is limited to ±100µA.
2. Due to the SCR structure inherent in the process used to fabricate these devices, connecting any digital inputs or outputs to voltages greater than V+ or less than GND may cause destructive device latchup. For this reason it is recommended that no inputs from sources other than the same power supply be applied to the ICL7109 before its power supply is established, and that in multiple supply systems the supply to the ICL7109 be activated first.
3. This limit refers to that of the package and will not be obtained during normal operation.
4. This parameter is not production tested, but is guaranteed by design.
5. Roll-over error for $T_A = -55°C$ to $+125°C$ is ±3 counts maximum.
6. A full scale voltage of 2.048V is used because a full scale voltage of 4.096V exceeds the devices Common Mode Voltage Range.
7. For Cerdip package the Ratiometric error can be ±4 (Min.).

NOTE: All typical values have been characterized but are not tested.

TABLE 1: Pin Assignment and Function Description

Pin	Symbol	Description
1	GND	Digital Ground, 0V. Ground return for all digital logic.
2	STATUS	Output High during integrate and deintegrate until data is latched. Output Low when analog section is in Auto-Zero configuration.
3	POL	Polarity — HI for Positive input.
4	OR	Overrange — HI if Overranged.
5	B12	Bit 12 (Most Significant Bit)
6	B11	Bit 11
7	B10	Bit 10
8	B9	Bit 9
9	B8	Bit 8
10	B7	Bit 7
11	B6	Bit 6
12	B5	Bit 5
13	B4	Bit 4
14	B3	Bit 3
15	B2	Bit 2
16	B1	Bit 1 (Least Significant Bit)

(Pins 5–16: HI = true. All three state output data bits.)

Pin	Symbol	Description
17	TEST	Input High — Normal Operation. Input Low — Forces all bit outputs high. Note: This input is used for test purposes only. Tie high if not used.
18	$\overline{LBEN}$	Low Byte Enable — With Mode (Pin 21) low, and $\overline{CE/LOAD}$ (Pin 20) low, taking this pin low activates low order byte outputs B1 — B8. — With Mode (Pin 21) high, this pin serves as a low byte flag output used in handshake mode. See Figures 8, 9, 10.
19	$\overline{HBEN}$	High Byte Enable — With Mode (Pin 21) low, and $\overline{CE/LOAD}$ (Pin 20) low, taking this pin low activates high order byte outputs B9 — B12, POL, OR. — With Mode (Pin 21) high, this pin serves as a high byte flag output used in handshake mode. See Figures 8, 9, 10.
20	$\overline{CE/LOAD}$	Chip Enable/Load — With Mode (Pin 21) low, $\overline{CE/LOAD}$ serves as a master output enable. When high, B1 — B12, POL, OR outputs are disabled. — With Mode (Pin 21) high, this pin serves as a load strobe used in handshake mode. See Figures 8, 9, 10.

Pin	Symbol	Description
21	MODE	Input Low — Direct output mode where $\overline{CE/LOAD}$ (Pin 20), $\overline{HBEN}$ (Pin 19) and $\overline{LBEN}$ (Pin 18) act as inputs directly controlling byte outputs. Input Pulsed High — Causes immediate entry into handshake mode and output of data as in Figure 10. Input High — Enables $\overline{CE/LOAD}$ (Pin 20), $\overline{HBEN}$ (Pin 19), and $\overline{LBEN}$ (Pin 18) as outputs, handshake mode will be entered and data output as in Figures 8 and 9 at conversion completion.
22	OSC IN	Oscillator Input
23	OSC OUT	Oscillator Output
24	OSC SEL	Oscillator Select — Input high configures OSC IN, OSC OUT, BUF OSC OUT as RC oscillator — clock will be same phase and duty cycle as BUF OSC OUT. — Input low configures OSC IN, OSC OUT for crystal oscillator — clock frequency will be 1/58 of frequency at BUF OSC OUT.
25	BUF OSC OUT	Buffered Oscillator Output
26	RUN/HOLD	Input High — Conversions continuously performed every 8192 clock pulses. Input Low — Conversion in progress completed, converter will stop in Auto-Zero 7 counts before integrate.
27	SEND	Input — Used in handshake mode to indicate ability of an external device to accept data. Connect to +5V if not used.
28	V^-	Analog Negative Supply — Nominally −5V with respect to GND (Pin 1).
29	REF OUT	Reference Voltage Output — Nominally 2.8V down from V^+ (Pin 40).
30	BUFFER	Buffer Amplifier Output
31	AUTO-ZERO	Auto-Zero Node — Inside foil of C_{AZ}
32	INTEGRATOR	Integrator Output — Outside foil of C_{INT}
33	COMMON	Analog Common — System is Auto-Zeroed to COMMON
34	INPUT LO	Differential Input Low Side
35	INPUT HI	Differential Input High Side
36	REF IN +	Differential Reference Input Positive
37	REF CAP +	Reference Capacitor Positive
38	REF CAP −	Reference Capacitor Negative
39	REF IN −	Differential Reference Input Negative
40	V^+	Positive Supply Voltage — Nominally +5V with respect to GND (Pin 1).

Note: All digital levels are positive true.

NOTE: All typical values have been characterized but are not tested.

ICL7109

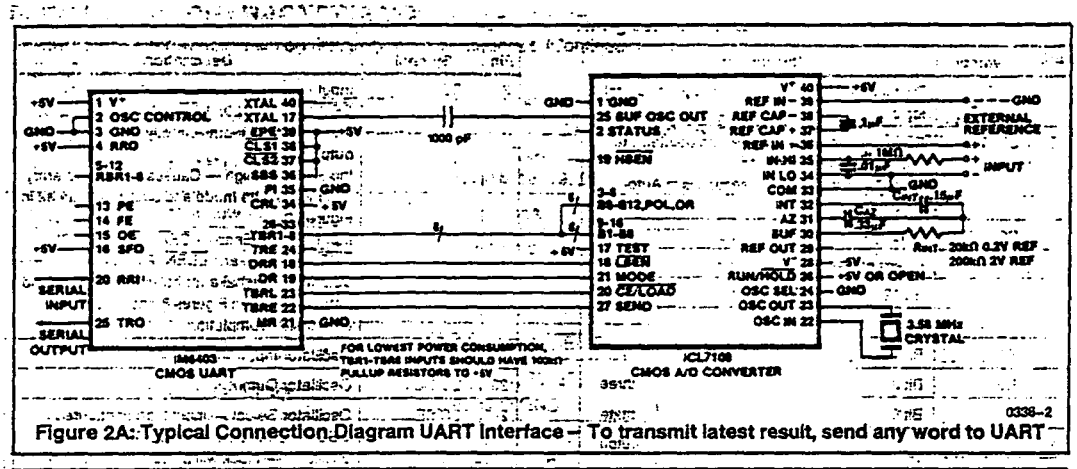


Figure 2A: Typical Connection Diagram UART Interface – To transmit latest result, send any word to UART

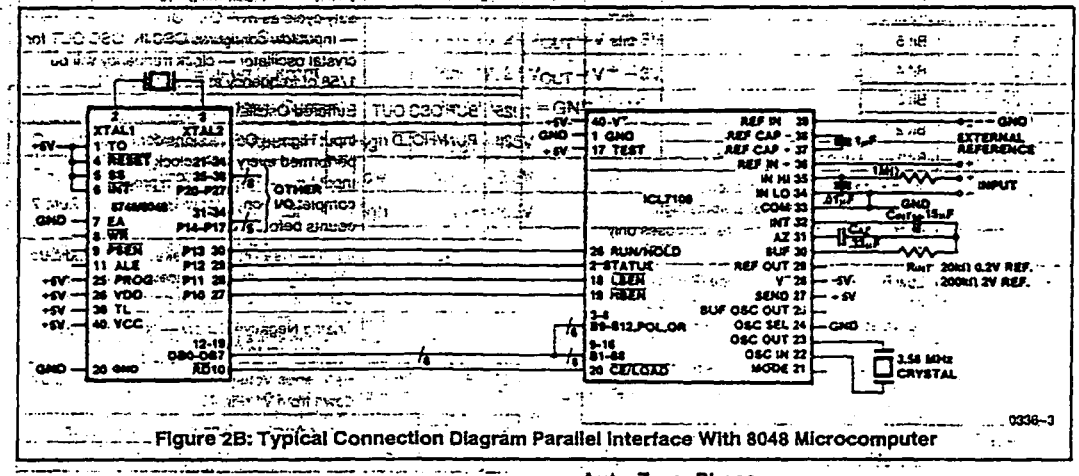


Figure 2B: Typical Connection Diagram Parallel Interface With 8048 Microcomputer

DETAILED DESCRIPTION

Analog Section

Figure 3 shows the equivalent circuit of the Analog Section of the ICL7109. When the RUN/HOLD input is left open or connected to V^+, the circuit will perform conversions at a rate determined by the clock frequency (8192 clock periods per cycle). Each measurement cycle is divided into three phases as shown in Figure 4. They are (1) Auto-Zero (AZ), (2) Signal Integrate (INT) and (3) Deintegrate (DE).

Auto-Zero Phase

During auto-zero three things happen. First, input high and low are disconnected from their pins and internally shorted to analog COMMON. Second, the reference capacitor is charged to the reference voltage. Third, a feedback loop is closed around the system to charge the auto-zero capacitor C_{AZ} to compensate for offset voltages in the buffer amplifier, integrator, and comparator. Since the comparator is included in the loop, the AZ accuracy is limited only by the noise of the system. In any case, the offset referred to the input is less than 10μV.

NOTE: All typical values have been characterized but are not tested.

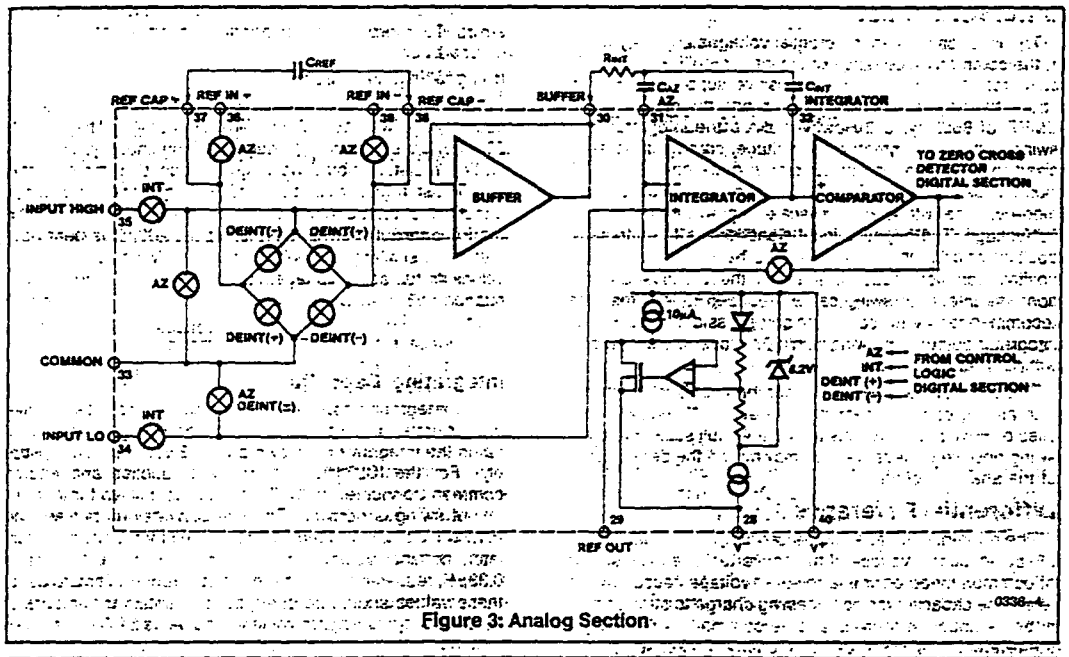


Figure 3: Analog Section

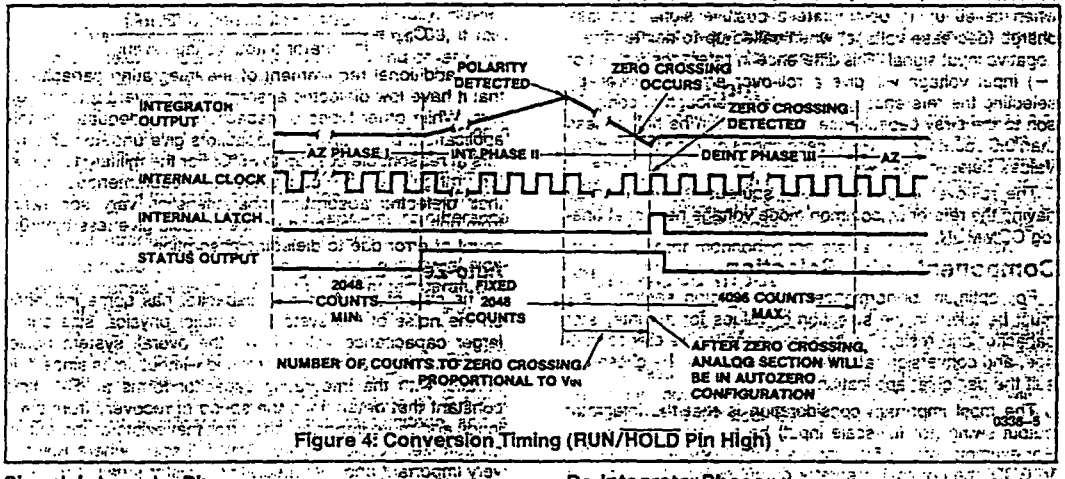


Figure 4: Conversion Timing (RUN/HOLD Pin High)

Signal Integrate Phase

During signal integrate, the auto-zero loop is opened, the internal short is removed and the internal high and low inputs are connected to the external pins. The converter then integrates the differential voltage between IN HI and IN LO for a fixed time of 2048 clock periods. Note that this differential voltage must be within the common mode range of the inputs. At the end of this phase, the polarity of the integrated signal is determined.

De-Integrate Phase

The final phase is de-integrate, or reference integrate. Input low is internally connected to analog COMMON and input high is connected across the previously charged (during auto-zero) reference capacitor. Circuitry within the chip ensures that the capacitor will be connected with the correct polarity to cause the integrator output to return to zero crossing (established in Auto Zero) with a fixed slope. Thus the time for the output to return to zero (represented by the number of clock periods counted) is proportional to the input signal.

NOTE: All typical values have been characterized but are not tested.

ICL7109

Differential Input

The input can accept differential voltages anywhere within the common mode range of the input amplifier, or specifically from 1.0 volts below the positive supply, to 1.5 volts above the negative supply. In this range the system has a CMRR of 86dB typical. However, since the integrator also swings with the common mode voltage, care must be exercised to assure the integrator output does not saturate. A worst case condition would be a large positive common mode voltage with a near full-scale negative differential input voltage. The negative input signal drives the integrator positive when most of its swing has been used up by the positive common mode voltage. For these critical applications the integrator swing can be reduced to less than the recommended 4V full scale with some loss of accuracy. The integrator output can swing within 0.3 volts of either supply without loss of linearity.

The ICL7109 has, however, been optimized for operation with analog common near digital ground. With power supplies of +5V and −5V, this allows a 4V full scale integrator swing positive or negative thus maximizing the performance of the analog section.

Differential Reference

The reference voltage can be generated anywhere within the power supply voltage of the converter. The main source of common mode error is a roll-over voltage caused by the reference capacitor losing or gaining charge to stray capacity on its nodes. If there is a large common mode voltage, the reference capacitor can gain charge (increase voltage) when called up to deintegrate a positive signal but lose charge (decrease voltage) when called up to deintegrate a negative input signal. This difference in reference for (+) or (−) input voltage will give a roll-over error. However, by selecting the reference capacitor large enough in comparison to the stray capacitance, this error can be held to less than 0.5 count for the worst case condition (see Component Values Selection below).

The roll-over error from these sources is minimized by having the reference common mode voltage near or at analog COMMON.

Component Value Selection

For optimum performance of the analog section, care must be taken in the selection of values for the integrator capacitor and resistor, auto-zero capacitor, reference voltage, and conversion rate. These values must be chosen to suit the particular application.

The most important consideration is that the integrator output swing (for full-scale input) be as large as possible. For example, with ±5V supplies and COMMON connected to GND, the nominal integrator output swing at full scale is ±4V. Since the integrator output can go to 0.3V from either supply without significantly affecting linearity, a 4V integrator output swing allows 0.7V for variations in output swing due to component value and oscillator tolerances. With ±5V supplies and a common mode range of ±1V required, the component values should be selected to provide ±3V integrator output swing. Noise and rollover errors will be slightly worse than in the ±4V case. For larger common mode voltage ranges, the integrator output swing must be reduced further. This will increase both noise and rollover errors. To improve the performance, supplies of ±6V may be used.

Integrating Resistor

Both the buffer amplifier and the integrator have a class A output stage with 100μA of quiescent current. They supply 20μA of drive current with negligible non-linearity. The integrating resistor should be large enough to remain in this very linear region over the input voltage range, but small enough that undue leakage requirements are not placed on the PC board. For 4.096 volt full scale, 200kΩ is near optimum and similarly a 20kΩ for a 409.6mV scale. For other values of full scale voltage, R_{INT} should be chosen by the relation

$$R_{INT} = \frac{\text{full scale voltage}}{20\mu A}$$

Integrating Capacitor

The integrating capacitor C_{INT} should be selected to give the maximum integrator output voltage swing without saturating the integrator (approximately 0.3 volt from either supply). For the ICL7109 with ±5 volt supplies and analog common connected to GND, a ±3.5 to ±4 volt integrator output swing is nominal. For 7-½ conversions per second (61.72kHz clock frequency) as provided by the crystal oscillator, nominal values for C_{INT} and C_{AZ} are 0.15μF and 0.33μF, respectively. If different clock frequencies are used, these values should be changed to maintain the integrator output voltage swing. In general, the value of C_{INT} is given by

$$C_{INT} = \frac{(2048 \times \text{clock period})(20\mu A)}{\text{integrator output voltage swing}}\ \mu F$$

An additional requirement of the integrating capacitor is that it have low dielectric absorption to prevent roll-over errors. While other types of capacitors are adequate for this application, polypropylene capacitors give undetectable errors at reasonable cost up to 85°C. For the military temperature range, Teflon® capacitors are recommended. While their dielectric absorption characteristics vary somewhat from unit to unit, selected devices should give less than 0.5 count of error due to dielectric absorption.

Auto-Zero Capacitor

The size of the auto-zero capacitor has some influence on the noise of the system: a smaller physical size and a larger capacitance value lower the overall system noise. However, C_{AZ} cannot be increased without limits since it, in parallel with the integrating capacitor forms an R-C time constant that determines the speed of recovery from overloads and more important the error that exists at the end of an auto-zero cycle. For 409.6mV full scale where noise is very important and the integrating resistor small, a value of C_{AZ} twice C_{INT} is optimum. Similarly for 4.096V full scale where recovery is more important than noise, a value of C_{AZ} equal to half of C_{INT} is recommended.

For optimal rejection of stray pickup, the outer foil of C_{AZ} should be connected to the R-C summing junction and the inner foil to pin 31. Similarly the outer foil of C_{INT} should be connected to pin 32 and the inner foil to the R-C summing junction. Teflon®, or equivalent, capacitors are recommended above 85°C for their low leakage characteristics.

NOTE: All typical values have been characterized but are not tested.

Reference Capacitor

A 1μF capacitor gives good results in most applications. However, where a large reference common mode voltage exists (i.e. the reference low is not at analog common) and a 409.6mV scale is used, a larger value is required to prevent roll-over error. Generally 10μF will hold the roll-over error to 0.5 count in this instance. Again, Teflon®, or equivalent capacitors should be used for temperatures above 85°C for their low leakage characteristics.

Reference Voltage

The analog input required to generate a full scale output of 4096 counts is $V_{IN} = 2V_{REF}$. Thus for a normalized scale, a reference of 2.048V should be used for a 4.096V full scale, and 204.8mV should be used for a 0.4096V full scale. However, in many applications where the A/D is sensing the output of a transducer, there will exist a scale factor other than unity between the absolute output voltage to be measured and a desired digital output. For instance, in a weighing system, the designer might like to have a full scale reading when the voltage from the transducer is 0.682V. Instead of dividing the input down to 409.6mV, the input voltage should be measured directly and a reference voltage of 0.341V should be used. Suitable values for integrating resistor and capacitor are 33kΩ and 0.15μF. This avoids a divider on the input. Another advantage of this system occurs when a zero reading is desired for non-zero input. Temperature and weight measurements with an offset or tare are examples. The offset may be introduced by connecting the voltage output of the transducer between common and analog high, and the offset voltage between common and analog low, observing polarities carefully. However, in processor-based systems using the ICL7109, it may be more efficient to perform this type of scaling or tare subtraction digitally using software.

Reference Sources

The stability of the reference voltage is a major factor in the overall absolute accuracy of the converter. The resolution of the ICL7109 at 12 bits is one part in 4096, or 244ppm. Thus if the reference has a temperature coefficient of 80ppm/°C (onboard reference) a temperature difference of 3°C will introduce a one-bit absolute error.

For this reason, it is recommended that an external high-quality reference be used where the ambient temperature is not controlled or where high-accuracy absolute measurements are being made.

The ICL7109 provides a REFerence OUTput (pin 29) which may be used with a resistive divider to generate a suitable reference voltage. This output will sink up to about 20mA without significant variation in output voltage, and is provided with a pullup bias device which sources about 10μA. The output voltage is nominally 2.8V below V+, and has a temperature coefficient of ±80ppm/°C typ. When using the onboard reference, REF OUT (Pin 29) should be connected to REF+ (pin 39), and REF− should be connected to the wiper of a precision potentiometer between REF OUT and V+. The circuit for a 204.8mV reference is shown in the test circuit. For a 2.048mV reference, the fixed resistor should be removed, and a 25kΩ precision potentiometer between REF OUT and V+ should be used.

Note that if pins 29 and 39 are tied together and pins 39 and 40 accidentally shorted (e.g., during testing), the reference supply will sink enough current to destroy the device. This can be avoided by placing a 1kΩ resistor in series with pin 39.

DETAILED DESCRIPTION

Digital Section

The digital section includes the clock oscillator and scaling circuit, a 12-bit binary counter with output latches and TTL-compatible three-state output drivers, polarity, overrange and control logic, and UART handshake logic, as shown in Figure 5.

Throughout this description, logic levels will be referred to as "low" or "high". The actual logic levels are defined in the Electrical Characteristics Table. For minimum power consumption, all inputs should swing from GND (low) to V+ (high). Inputs driven from TTL gates should have 3-5kΩ pullup resistors added for maximum noise immunity.

MODE Input

The MODE input is used to control the output mode of the converter. When the MODE pin is low or left open (this input is provided with a pulldown resistor to ensure a low level when the pin is left open), the converter is in its "Direct" output mode, where the output data is directly accessible under the control of the chip and byte enable inputs. When the MODE input is pulsed high, the converter enters the UART handshake mode and outputs the data in two bytes, then returns to "direct" mode. When the MODE input is left high, the converter will output data in the handshake mode at the end of every conversion cycle. (See section entitled "Handshake Mode" for further details).

STATUS Output

During a conversion cycle, the STATUS output goes high at the beginning of Signal Integrate (Phase II), and goes low one-half clock period after new data from the conversion has been stored in the output latches. See Figure 4 for details of this timing. This signal may be used as a "data valid" flag (data never changes while STATUS is low) to drive interrupts, or for monitoring the status of the converter.

RUN/HOLD Input

When the RUN/HOLD input is high, or left open, the circuit will continuously perform conversion cycles, updating the output latches after zero crossing during the Deintegrate (Phase III) portion of the conversion cycle (See Figure 4). In this mode of operation, the conversion cycle will be performed in 8192 clock periods, regardless of the resulting value.

If RUN/HOLD goes low at any time during Deintegrate (Phase III) after the zero crossing has occurred, the circuit will immediately terminate Deintegrate and jump to Auto-Zero. This feature can be used to eliminate the time spent in Deintegrate after the zero-crossing. If RUN/HOLD stays or goes low, the converter will ensure minimum Auto-Zero time, and then wait in Auto-Zero until the RUN/HOLD input goes high. The converter will begin the Integrate (Phase II) portion of the next conversion (and the STATUS output will go high) seven clock periods after the high level is detected at RUN/HOLD. See Figure 6 for details.

3

NOTE: All typical values have been characterized but are not tested.

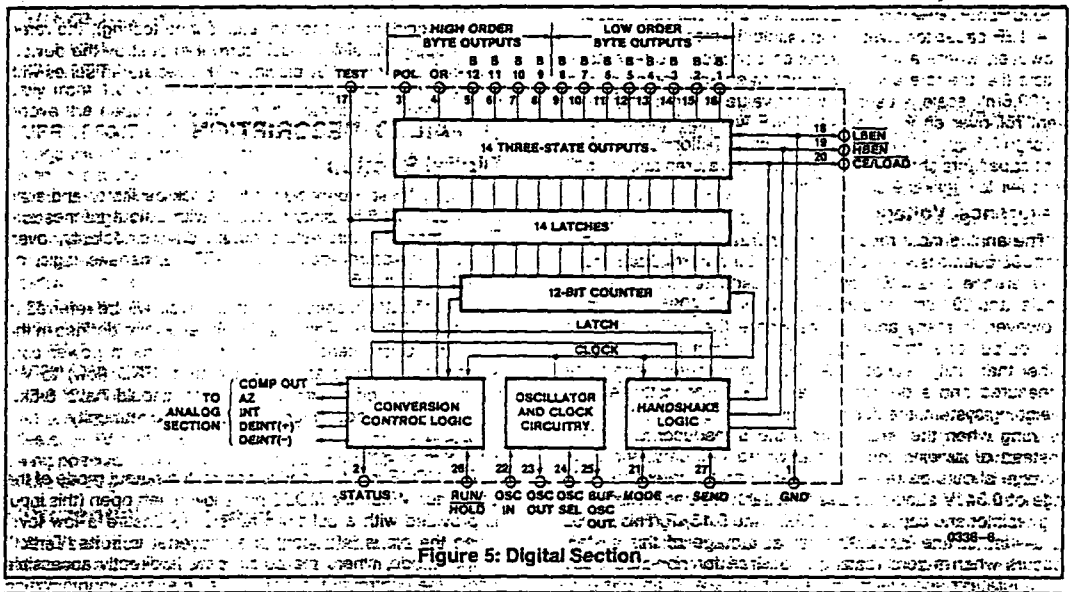


Figure 5: Digital Section

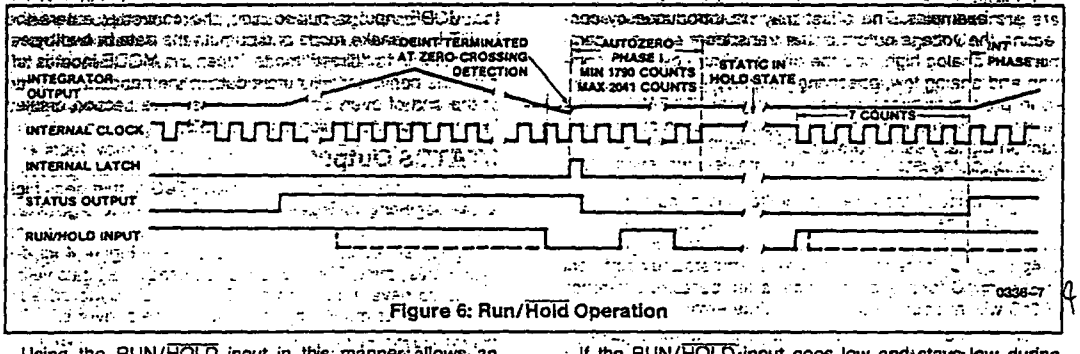


Figure 6: Run/Hold Operation

Using the RUN/HOLD input in this manner allows an easy "convert on demand" interface to be used. The converter may be held at idle in auto-zero with RUN/HOLD low. When RUN/HOLD goes high the conversion is started, and when the STATUS output goes low the new data is valid (or transferred to the UART — see Handshake Mode). RUN/HOLD may now be taken low which terminates deintegrate and ensures a minimum Auto-Zero time before the next conversion.

Alternately, RUN/HOLD can be used to minimize conversion time by ensuring that it goes low during Deintegrate, after zero crossing, and goes high after the hold point is reached. The required activity on the RUN/HOLD input can be provided by connecting it to the Buffered Oscillator Output. In this mode the conversion time is dependent on the input value measured. Also refer to Harris Application Bulletin, A032 for a discussion of the effects this will have on Auto-Zero performance.

If the RUN/HOLD input goes low and stays low during Auto-Zero (Phase I), the converter will simply stop at the end of Auto-Zero and wait for RUN/HOLD to go high. As above, Integrate (Phase II) begins seven clock periods after the high level is detected.

Direct Mode

When the MODE pin is left at a low level, the data outputs (bits 1 through 8 low order byte, bits 9 through 12, polarity and over-range high order byte) are accessible under control of the byte and chip enable terminals as inputs. These three inputs are all active low, and are provided with pullup resistors to ensure an inactive high level when left open. When the chip enable input is low, taking a byte enable input low will allow the outputs of that byte to become active (three-stated on). This allows a variety of parallel data accessing techniques to be used, as shown in the section entitled "Interfacing." The timing requirements for these outputs are shown in Figure 7 and Table 2.

NOTE: All typical values have been characterized but are not tested.

Table 2 — Direct Mode Timing Requirements
(See Note 4 of Electrical Characteristics)

SYMBOL	DESCRIPTION	MIN	TYP	MAX	UNIT
t_{BEA}	Byte Enable Width	350	220		ns
t_{DAB}	Data Access Time from Byte Enable		210	350	ns
t_{DHB}	Data Hold Time from Byte Enable	150	300		ns
t_{CEA}	Chip Enable Width	400	260		ns
t_{DAC}	Data Access Time from Chip Enable		260	400	ns
t_{DHC}	Data Hold Time from Chip Enable	240	400		ns

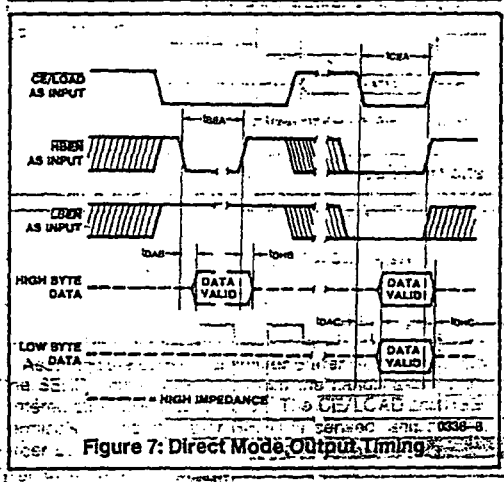


Figure 7: Direct Mode Output Timing

It should be noted that these control inputs are asynchronous with respect to the converter clock — the data may be accessed at any time. Thus it is possible to access the latches while they are being updated, which could lead to erroneous data. Synchronizing the access of the latches with the conversion cycle by monitoring the STATUS output will prevent this. Data is never updated while STATUS is low.

Handshake Mode

The handshake output mode is provided as an alternative means of interfacing the ICL7109 to digital systems, where the A/D converter becomes active in controlling the flow of data instead of passively responding to chip and byte enable inputs. This mode is specifically designed to allow a direct interface between the ICL7109 and industry-standard UARTs (such as the Harris IM6402/3) with no external logic required. When triggered into the handshake mode, the ICL7109 provides all the control and flag signals necessary to sequentially transfer two bytes of data into the UART and initiate their transmission in serial form. This greatly eases the task and reduces the cost of designing remote data acquisition stations using serial data transmission.

Entry into the handshake mode is controlled by the MODE pin. When the MODE terminal is held high, the ICL7109 will enter the handshake mode after new data has been stored in the output latches at the end of a conversion (See Figures 8 and 9). The MODE terminal may also be used to trigger entry into the handshake mode on demand. At any time during the conversion cycle, the low to high transition of a short pulse at the MODE input will cause immediate entry into the handshake mode. If this pulse occurs while new data is being stored, the entry into handshake mode is delayed until the data is stable. While the converter is in the handshake mode, the MODE input is ignored, and although conversions will still be performed, data updating will be inhibited (See Figure 10) until the converter completes the output cycle and clears the handshake mode.

When the converter enters the handshake mode, or when the MODE input is high, the chip and byte enable terminals become TTL-compatible outputs which provide the control signals for the output cycle (See Figures 8, 9, and 10).

In handshake mode, the SEND input is used by the converter as an indication of the ability of the receiving device (such as a UART) to accept data.

Figure 8 shows the sequence of the output cycle with SEND held high. The handshake mode (internal MODE high) is entered after the data latch pulse, and since MODE remains high the $\overline{CE/LOAD}$, $\overline{LBEN}$ and $\overline{HBEN}$ terminals are active as outputs. The high level at the SEND input is sensed on the same high to low internal clock edge that terminates the data latch pulse. On the next low to high internal clock edge the $\overline{CE/LOAD}$ and the $\overline{HBEN}$ outputs assume a low level, and the high-order byte (bits 9 through 12, POL, and OR) outputs are enabled. The $\overline{CE/LOAD}$ output remains low for one full internal clock period only; the data outputs remain active for 1-½ internal clock periods, and the high byte enable remains low for two clock periods. Thus the $\overline{CE/LOAD}$ output low-level-or-low-to-high edge may be used as a synchronizing signal to ensure valid data, and the byte enable as an output may be used as a byte identification flag. With SEND remaining high the converter completes the output cycle using $\overline{CE/LOAD}$ and $\overline{LBEN}$ while the low order byte outputs (bits 1 through 8) are activated. The handshake mode is terminated when both bytes are sent.

Figure 9 shows an output sequence where the SEND input is used to delay portions of the sequence, or handshake to ensure correct data transfer. This timing diagram shows the relationships that occur using an industry-standard IM6402/3 CMOS UART to interface to serial data channels. In this interface, the SEND input to the ICL7109 is driven by the TBRE (Transmitter Buffer Register Empty) output of the UART, and the $\overline{CE/LOAD}$ terminal of the ICL7109 drives the TBRL (Transmitter Buffer Register Load) input to the UART. The data outputs are paralleled into the eight Transmitter Buffer Register inputs.

3

NOTE: All typical values have been characterized but are not tested.

ICL7109

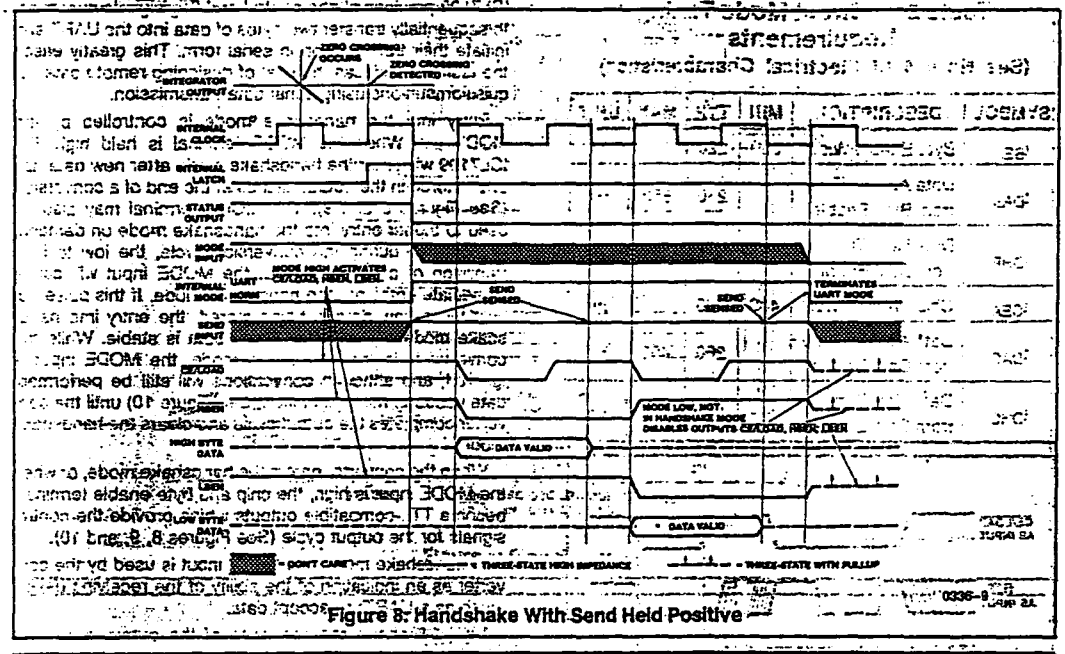


Figure 8. Handshake With Send Held Positive

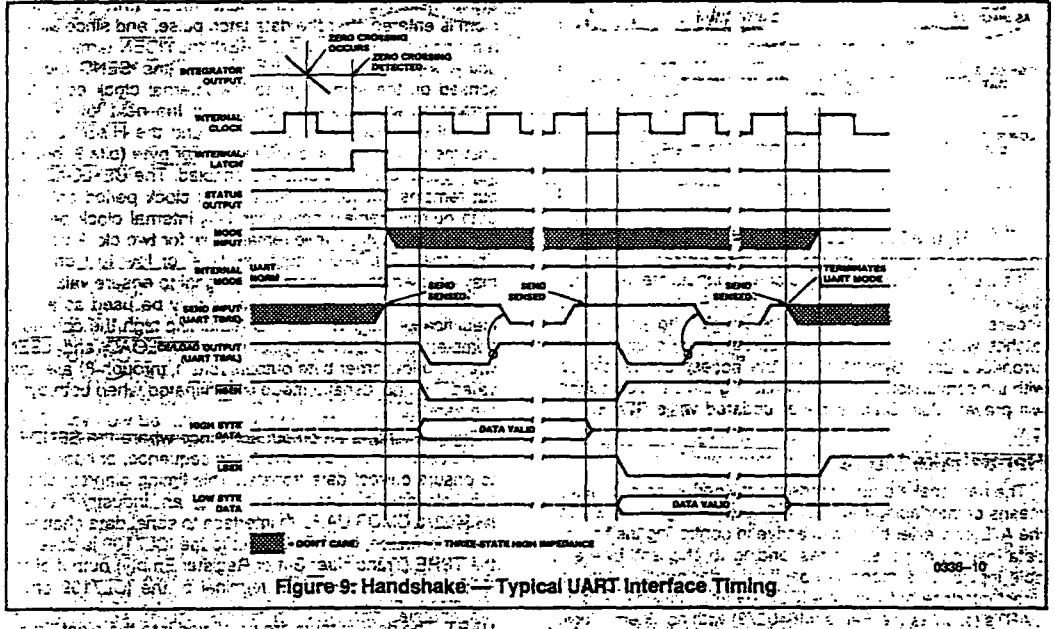


Figure 9. Handshake — Typical UART Interface Timing

NOTE: All typical values have been characterized but are not tested.

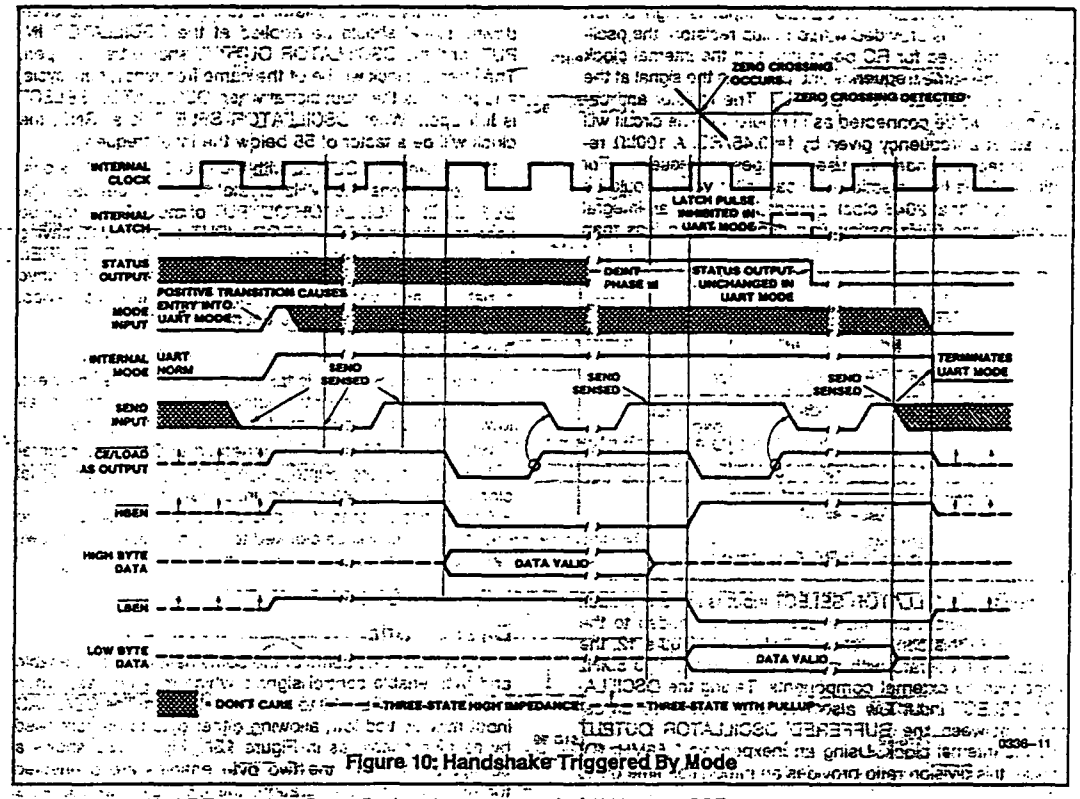


Figure 10. Handshake Triggered By Mode

Assuming the UART Transmitter Buffer Register is empty, the SEND input will be high when the handshake mode is entered after new data is stored. The CE/LOAD and HBEN terminals will go low after SEND is sensed, and the high order byte outputs become active. When CE/LOAD goes high at the end of one clock period, the high order byte data is clocked into the UART Transmitter Buffer Register. The UART TBRE output will now go low, which halts the output cycle with the HBEN output low, and the high order byte outputs active. When the UART has transferred the data to the Transmitter Register and cleared the Transmitter Buffer Register, the TBRE returns high. On the next ICL7109 internal clock high to low edge, the high order byte outputs are disabled, and one-half internal clock later, the HBEN output returns high. At the same time, the CE/LOAD and LBEN outputs go low, and the low order byte outputs become active. Similarly, when the CE/LOAD returns high at the end of one clock period, the low order data is clocked into the UART Transmitter Buffer Register, and TBRE again goes low. When TBRE returns to a high it will be sensed on the next ICL7109 internal clock high to low edge, disabling the data outputs. One-half internal clock later, the handshake mode will be cleared, and the CE/LOAD, HBEN, and LBEN terminals return high and stay active (as long as MODE stays high).

With the MODE input remaining high as in these examples, the converter will output the results of every conversion except those completed during a handshake operation. By triggering the converter into handshake mode with a low to high edge on the MODE input, handshake output sequences may be performed on demand. Figure 10 shows a handshake output sequence triggered by such an edge. In addition, the SEND input is shown as being low when the converter enters handshake mode. In this case, the whole output sequence is controlled by the SEND input, and the sequence for the first (high order) byte is similar to the sequence for the second byte. This diagram also shows the output sequence taking longer than a conversion cycle. Note that the converter still makes conversions, with the STATUS output and RUN/HOLD input functioning normally. The only difference is that new data will not be latched when in handshake mode, and is therefore lost.

Oscillator

The ICL7109 is provided with a versatile three terminal oscillator to generate the internal clock. The oscillator may be overdriven, or may be operated with an RC network or crystal. The OSCILLATOR SELECT input changes the internal configuration of the oscillator to optimize it for RC or crystal operation.

NOTE: All typical values have been characterized but are not tested.

3

When the OSCILLATOR SELECT input is high or left open (the input is provided with a pullup resistor), the oscillator is configured for RC operation, and the internal clock will be of the same frequency and phase as the signal at the BUFFERED OSCILLATOR OUTPUT. The resistor and capacitor should be connected as in Figure 11. The circuit will oscillate at a frequency given by $f = 0.45/RC$. A 100kΩ resistor is recommended for useful ranges of frequency. For optimum 60Hz line-rejection, the capacitor value should be chosen such that 2048 clock periods is close to an integral multiple of the 60Hz period (but should not be less than 50pF).

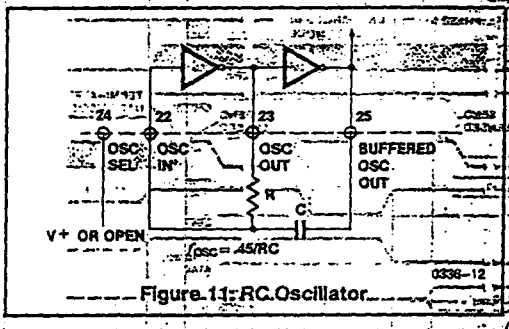


Figure 11: RC Oscillator

When the OSCILLATOR SELECT input is low a feedback device and output and input capacitors are added to the oscillator. In this configuration, as shown in Figure 12, the oscillator will operate with most crystals in the 1 to 5MHz range with no external components. Taking the OSCILLATOR SELECT input low also inserts a fixed ÷58 divider circuit between the BUFFERED OSCILLATOR OUTPUT and the internal clock. Using an inexpensive 3.58MHz TV crystal, this division ratio provides an integration time given by:

$$T_{INT.} = (2048 \text{ clock periods}) \times (T_{CLOCK}) = 33.18 \text{ ms}$$

where $T_{CLOCK} = \frac{58}{3.58 \text{ MHz}}$

This time is very close to two 60Hz periods or 33.33ms. The error is less than one percent, which will give better than 40dB 60Hz rejection. The converter will operate reliably at conversion rates of up to 30 per second, which corresponds to a clock frequency of 245.8kHz.

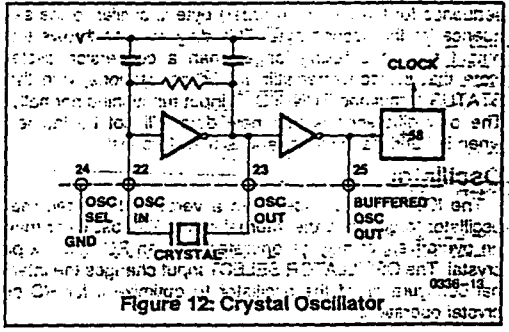


Figure 12: Crystal Oscillator

If at any time the oscillator is to be overdriven, the overdriving signal should be applied at the OSCILLATOR INPUT, and the OSCILLATOR OUTPUT should be left open. The internal clock will be of the same frequency, duty cycle, and phase as the input signal when OSCILLATOR SELECT is left open. When OSCILLATOR SELECT is at GND, the clock will be a factor of 58 below the input frequency.

When using the ICL7109 with the IM6403 UART, it is possible to use one 3.58MHz crystal for both devices. The BUFFERED OSCILLATOR OUTPUT of the ICL7109 may be used to drive the OSCILLATOR INPUT of the UART, saving the need for a second crystal. However, the BUFFERED OSCILLATOR OUTPUT does not have a great deal of drive capability, and when driving more than one slave device, external buffering should be used.

Test Input

When the TEST input is taken to a level halfway between V+ and GND, the counter output latches are enabled, allowing the counter contents to be examined anytime.

When the TEST input is connected to GND, the counter outputs are all forced into the high state, and the internal clock is disabled. When the input returns to the ½ (V+ – GND) voltage (or to V+) and one clock is applied, all the counter outputs will be clocked to the low state. This allows easy testing of the counter and its outputs.

INTERFACING

Direct Mode

Figure 13 shows some of the combinations of chip enable and byte enable control signals which may be used when interfacing the ICL7109 to parallel data lines. The CE/LOAD input may be tied low, allowing either byte to be controlled by its own enable as in Figure 13A. Figure 13B shows a configuration where the two byte enables are connected together. In this configuration, the CE/LOAD serves as a chip enable, and the HBEN and LBEN may be connected to GND or serve as a second chip enable. The 14 data outputs will all be enabled simultaneously. Figure 13C shows the HBEN and LBEN as flag inputs, and CE/LOAD as a master enable, which could be the READ strobe available from most microprocessors.

Figure 14 shows an approach to interfacing several ICL7109s to a bus, ganging the HBEN and LBEN signals to several converters together, and using the CE/LOAD inputs (perhaps decoded from an address) to select the desired converter.

1.25

NOTE: All typical values have been characterized but are not tested.

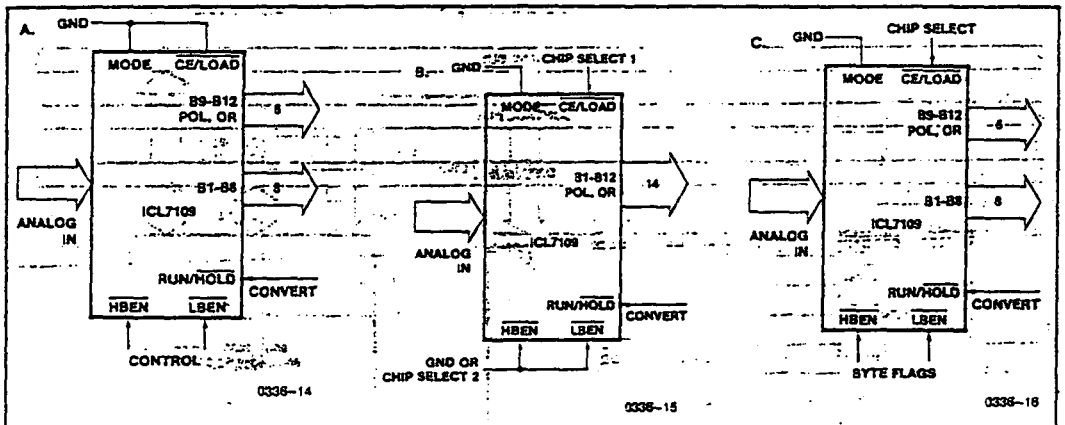


Figure 13: Direct Mode Chip and Byte Enable Combinations

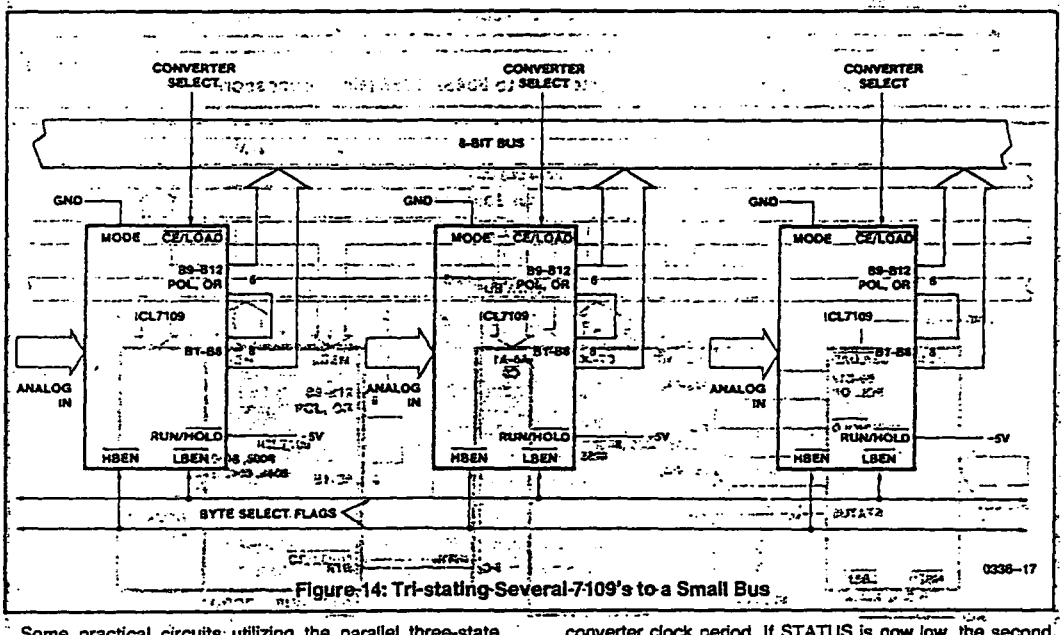


Figure 14: Tri-stating Several 7109's to a Small Bus

Some practical circuits utilizing the parallel three-state output capabilities of the ICL7109 are shown in Figures 15 through 20. Figure 15 shows a straightforward application to the Intel 8048/80/85 microprocessors via an 8255PPI, where the ICL7109 data outputs are active at all times. The I/O ports of an 8155 may be used in the same way. This interface can be used in a read-anytime mode, although a read performed while the data latches are being updated will lead to scrambled data. This will occur very rarely, in the proportion of setup skew times to conversion time. One way to overcome this is to read the STATUS output as well, and if it is high, read the data again after a delay of more than ½ converter clock period. If STATUS is now low, the second reading is correct, and if it is still high, the first reading is correct. Alternatively, this timing problem is completely avoided by using a read-after-update sequence, as shown in Figure 16. Here the high-to low transition of the STATUS output drives an interrupt to the microprocessor causing it to access the data latches. This application also shows the RUN/HOLD input being used to initiate conversions under software control.

A similar interface to Motorola MC6800, or Rockwell R650X systems is shown in Figure 17. The high to low transition of the STATUS output generates an interrupt via the

NOTE: All typical values have been characterized but are not tested.

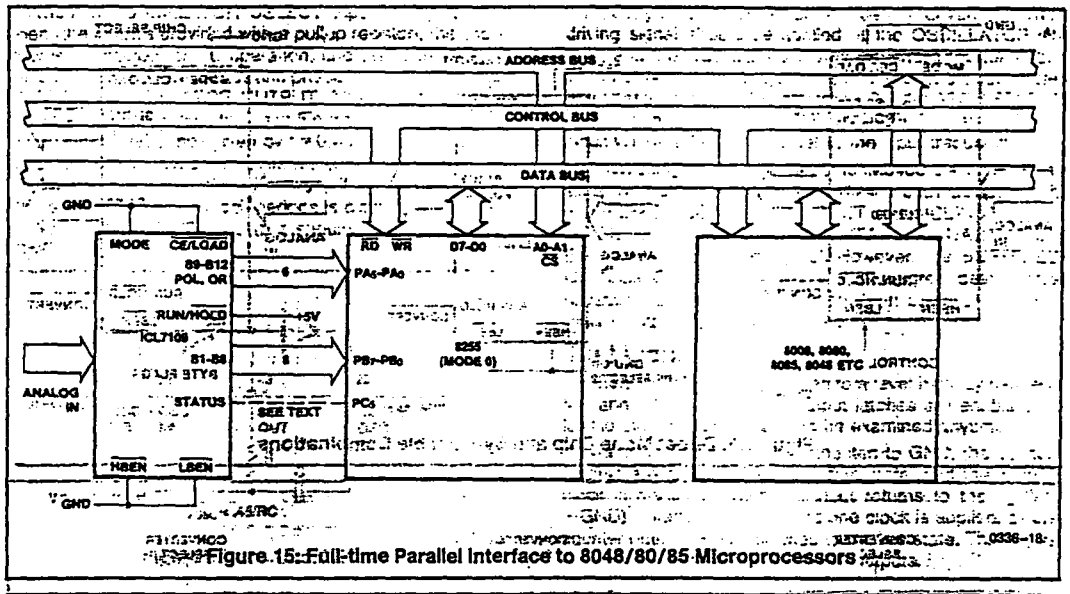


Figure 15: Full-time Parallel Interface to 8048/80/85 Microprocessors

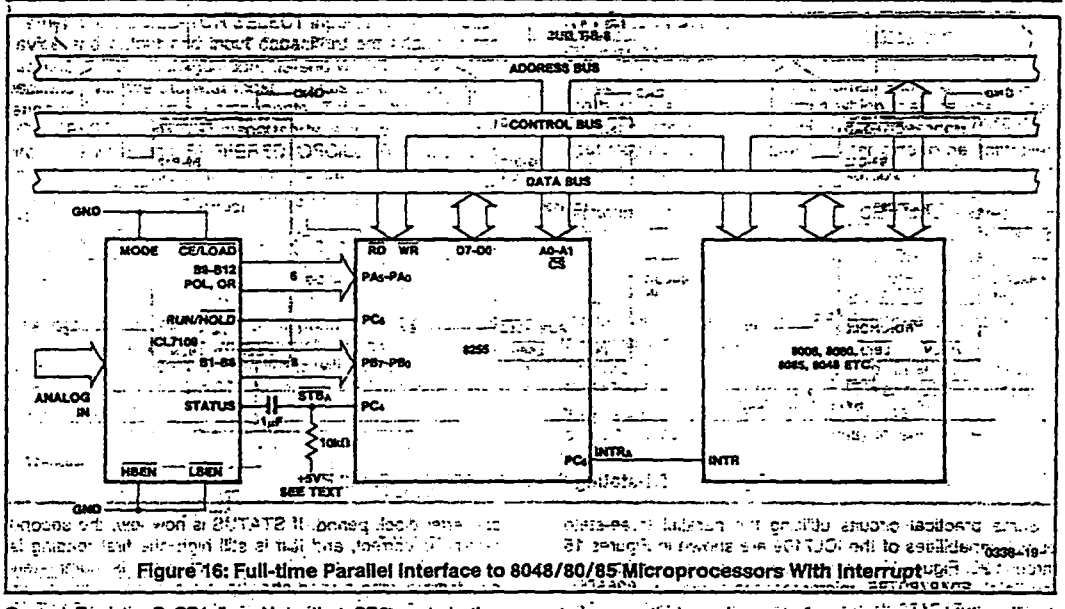


Figure 16: Full-time Parallel Interface to 8048/80/85 Microprocessors With Interrupt

Control Register B CB1 line. Note that CB2 controls the RUN/HOLD pin through Control Register B, allowing software-controlled initiation of conversions in this system as well.

The three-state output capability of the ICL7109 allows direct interfacing to most microprocessor busses. Examples of this are shown in Figures 18 and 19. It is necessary to carefully consider the system timing in this type of interface, to be sure that requirements for setup and hold times, and minimum pulse widths are met. Note also the drive limitations on long buses. Generally this type of interface is only favored if the memory peripheral address density is low so that simple address decoding can be used. Interrupt handling can also require many additional components, and using an interface device will usually simplify the system in this case.

NOTE: All typical values have been characterized but are not tested.

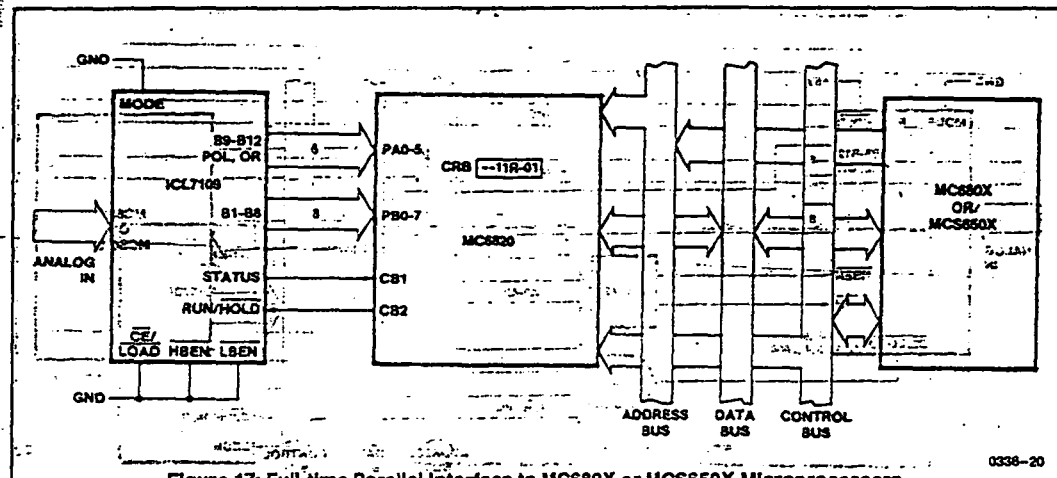


Figure 17: Full-time Parallel Interface to MC680X or MCS650X Microprocessors

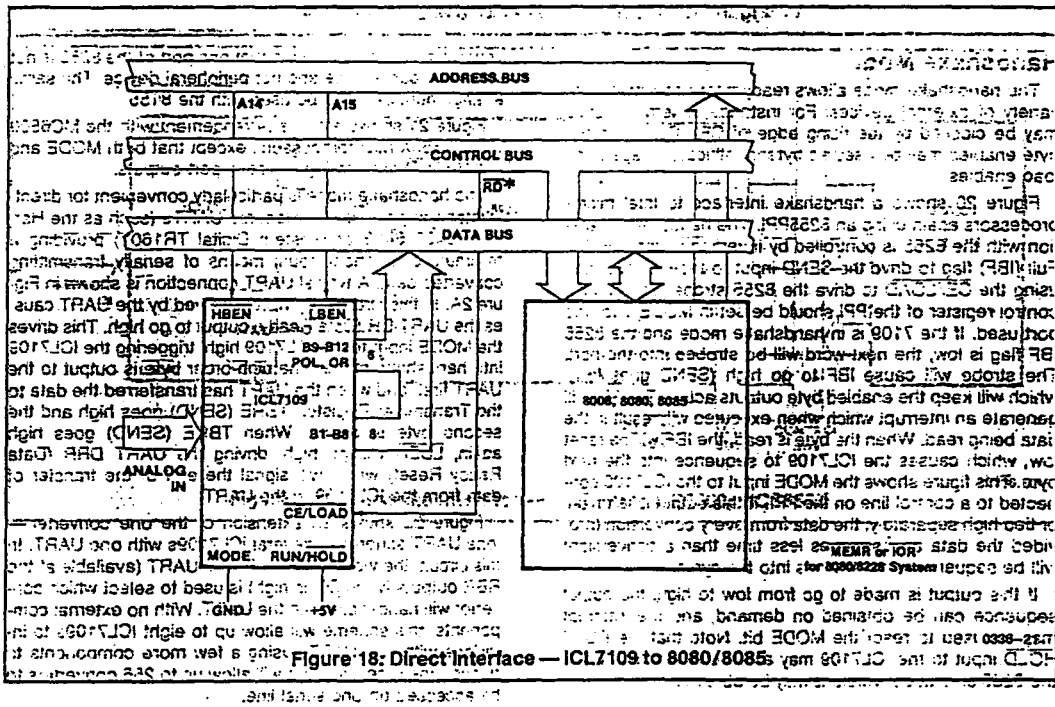


Figure 18: Direct Interface — ICL7109 to 8080/8085

3

NOTE: All typical values have been characterized but are not tested.

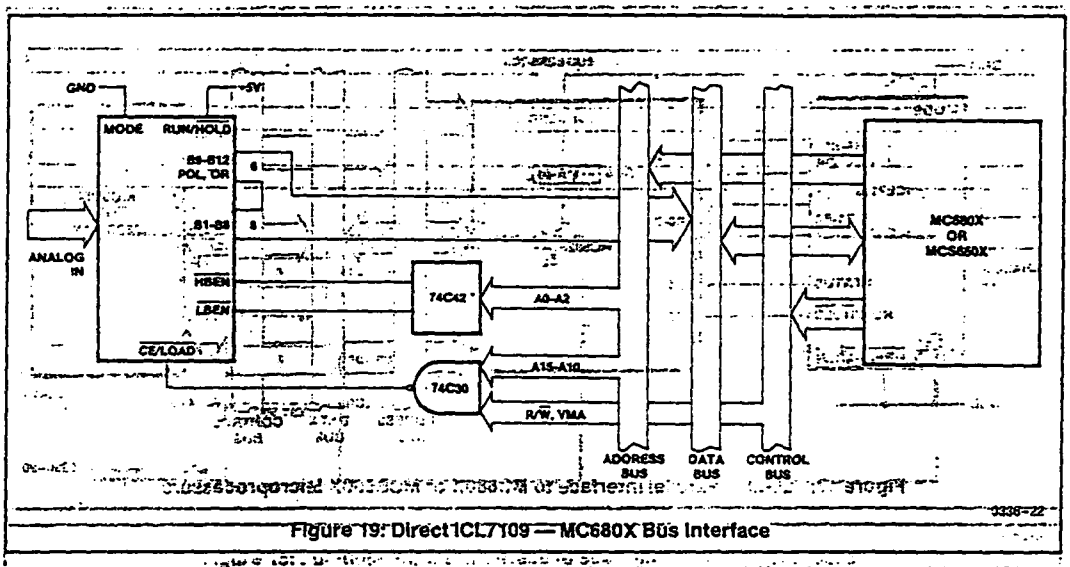


Figure 19. Direct ICL7109 — MC680X Bus Interface

Handshake Mode

The handshake mode allows ready interface with a wide variety of external devices. For instance, external latches may be clocked by the rising edge of CE/LOAD, and the byte enables may be used as byte identification flags or as load enables.

Figure 20 shows a handshake interface to Intel microprocessors again using an 8255PPI. The handshake operation with the 8255 is controlled by inverting its Input Buffer Full (IBF) flag to drive the SEND input to the ICL7109, and using the CE/LOAD to drive the 8255 strobe. The internal control register of the PPI should be set in MODE 1 for the port used. If the 7109 is in handshake mode and the 8255 IBF flag is low, the next word will be strobed into the port. The strobe will cause IBF to go high (SEND goes low), which will keep the enabled byte outputs active. The PPI will generate an interrupt which when executed will result in the data being read. When the byte is read, the IBF will be reset low, which causes the ICL7109 to sequence into the next byte. This figure shows the MODE input to the ICL7109 connected to a control line on the PPI. If this output is left high, or tied high separately, the data from every conversion (provided the data access takes less time than a conversion) will be sequenced in two bytes into the system.

If this output is made to go from low to high, the output sequence can be obtained on demand, and the interrupt may be used to reset the MODE bit. Note that the RUN/HOLD input to the ICL7109 may also be driven by a bit of the 8255 so that conversions may be obtained on command under software control. Note that one port of the 8255 is not excused, and can service another peripheral device. The same arrangement can also be used with the 8155.

Figure 21 shows a similar arrangement with the MC6800 or MCS650X microprocessors, except that both MODE and RUN/HOLD are tied high to save port outputs.

The handshake mode is particularly convenient for directly interfacing to industry standard UARTs (such as the Harris IM6402/6403 or Western Digital TR1602) providing a minimum component count means of serially transmitting converted data. A typical UART connection is shown in Figure 2A. In this circuit, any word received by the UART causes the UART DR (Data Ready) output to go high. This drives the MODE input to the ICL7109 high, triggering the ICL7109 into handshake mode. The high order byte is output to the UART first, and when the UART has transferred the data to the Transmitter Register, TBRE (SEND) goes high and the second byte is output. When TBRE (SEND) goes high again, LBEN will go high, driving the UART DRR (Data Ready Reset) which will signal the end of the transfer of data from the ICL7109 to the UART.

Figure 22 shows an extension of the one converter — one UART scheme to several ICL7109s with one UART. In this circuit, the word received by the UART (available at the RBR outputs when DR is high) is used to select which converter will handshake with the UART. With no external components, this scheme will allow up to eight ICL7109s to interface with one UART. Using a few more components to decode the received word will allow up to 256 converters to be accessed on one serial line.

NOTE: All typical values have been characterized but are not tested.

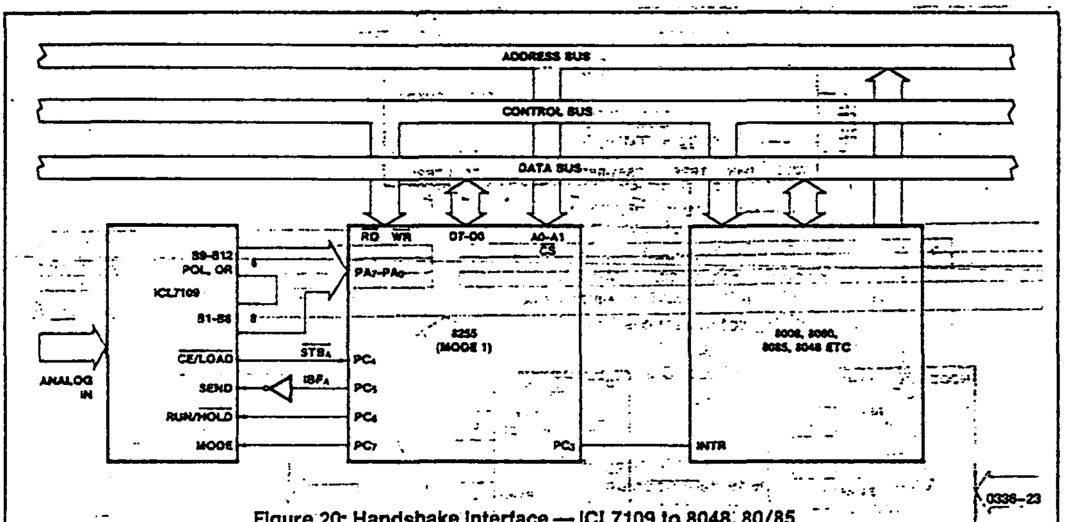


Figure 20: Handshake Interface — ICL7109 to 8048, 80/85

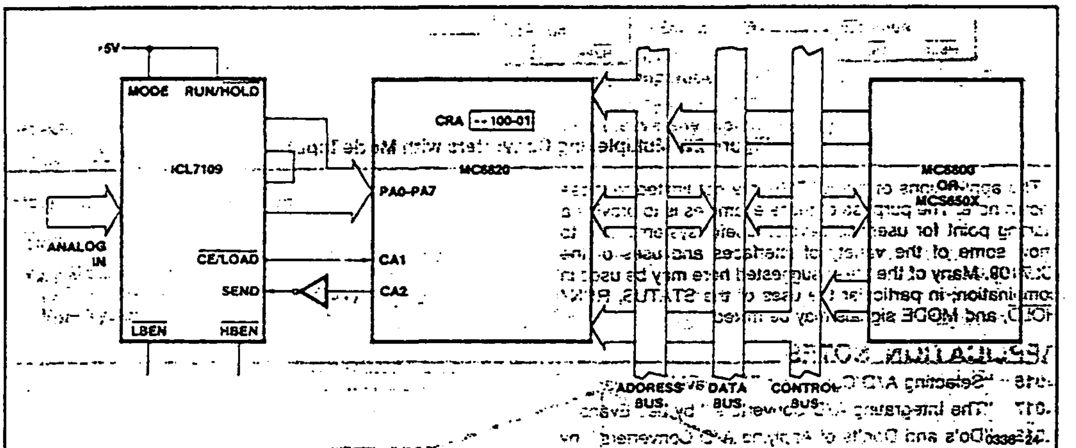


Figure 21: Handshake Interface — ICL7109 to MC6800, MCS650X

NOTE: All typical values have been characterized but are not tested.

ICL7109

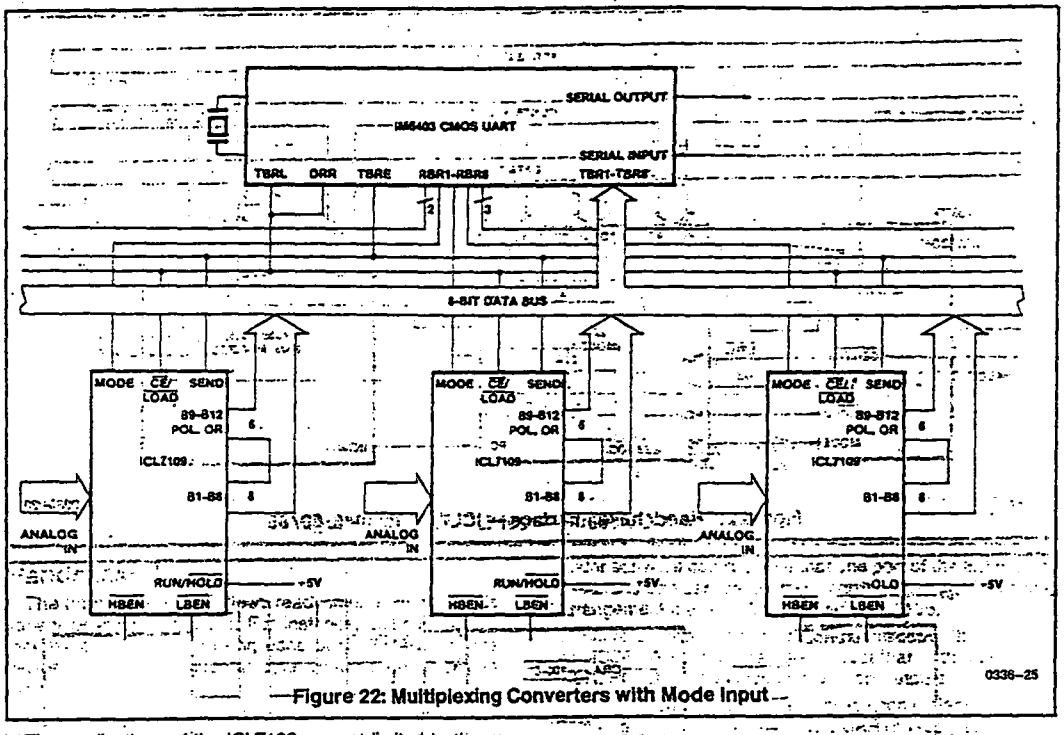


Figure 22: Multiplexing Converters with Mode Input

The applications of the ICL7109 are not limited to those shown here. The purpose of these examples is to provide a starting point for users to develop useful systems, and to show some of the variety of interfaces and uses of the ICL7109. Many of the ideas suggested here may be used in combination; in particular the uses of the STATUS, RUN/HOLD, and MODE signals may be mixed.

APPLICATION NOTES

A016 "Selecting A/D Converters," by David Fullagar

A017 "The Integrating A/D Converters," by Lee Evans.

A018 "Do's and Don'ts of Applying A/D Converters," by Peter Bradshaw and Skip Osgood.

A030 "The ICL7104—A Binary Output A/D Converter for Microprocessors," by Peter Bradshaw.

A032 "Understanding the Auto-Zero and Common Mode Performance of the ICL7106 Family," by Peter Bradshaw

R005 "Interfacing Data Converters & Microprocessors," by Peter Bradshaw et al, Electronics, Dec. 9, 1976.

NOTE: All typical values have been characterized but are not tested.

ICL7109
INTEGRATING A/D CONVERTER
EQUATIONS

Oscillator Frequency

$f_{OSC} = 0.45/RC$

$C_{OSC} > 50$ pF; $R_{OSC} > 50$ kΩ

f_{OSC} typ. = 60 kHz

or

f_{OSC} = 3.58 MHz Crystal

Oscillator Period

$t_{OSC} = RC/0.45$

$t_{OSC} = 1/3.58$ MHz (Crystal)

Integration Clock Frequency

$f_{CLOCK} = f_{OSC}$ (RC Mode)

$f_{CLOCK} = f_{OSC}/58$ (Crystal)

$t_{CLOCK} = 1/f_{CLOCK}$

Integration Period

$t_{INT} = 2048 \times t_{CLOCK}$

60/50 Hz Rejection Criterion

$t_{INT}/t_{60\ Hz}$ or $t_{INT}/t_{50\ Hz}$ = Integer

Optimum Integration Current

I_{INT} = 20.0 μA

Full Scale Analog Input Voltage

V_{INFS} Typically = 200 mV or 2.0V

Integrate Resistor

$$R_{INT} = \frac{V_{INFS}}{I_{INT}}$$

Integrate Capacitor

$$C_{INT} = \frac{(t_{INT})(I_{INT})}{V_{INT}}$$

Integrator Output Voltage Swing

$$\Delta V_{INT} = \frac{(t_{INT})(I_{INT})}{C_{INT}}$$

V_{INT} Maximum Swing:

$(V^- + 0.5V) < V_{INT} < (V^+ - 0.5V)$

ΔV_{INT} Typically = 2.0V

Display Count

$$COUNT = 2048 \times \frac{V_{IN}}{V_{REF}}$$

Conversion Cycle

$t_{CYC} = t_{CLOCK} \times 8192$

(In Free-Run Mode, Run/HOLD = 1)

when f_{CLOCK} = 60 kHz, t_{CYC} = 133 ms

Common Mode Input Voltage

$(V^- + 1.0V) < V_{IN} < (V^+ - 0.5V)$

Auto Zero Capacitor

0.01 μF ≤ C_{AZ} < 1.0 μF

Reference Capacitor

0.1 μF ≤ C_{REF} ≤ 1.0 μF

V_{REF}

Biased between V^+ and V^-

$V_{REF} = V^+ - 2.8V$

Regulation lost when V^+ to V^- ≤ 6.4V

If V_{REF} is not used, float output pin.

Power Supply: Dual ±5.0V

V^+ = +5.0 to GND

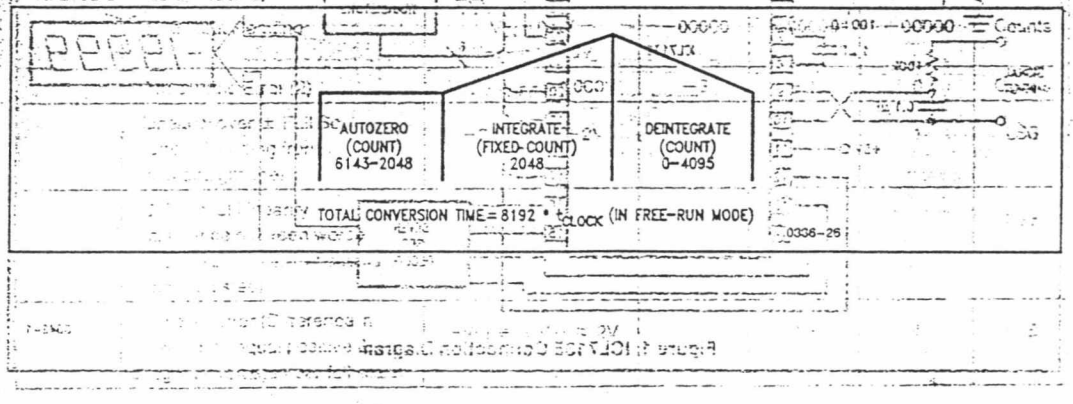
V^- = -5.0 to GND

Output Type:

Binary Amplitude with Polarity and Overrange Bits

Tips: Always tie TEST pin HIGH.

Don't leave any inputs floating.



NOTE: All typical values have been characterized but are not tested.

ELECTRICAL CHARACTERISTICS ($V^+ = +5V$, $V^- = -5V$, GND = 0V, $T_A = 25°C$, unless otherwise indicated.) Test circuit as shown on first page of this data sheet. (Continued)

DIGITAL SECTION

Symbol	Parameter		Test Conditions	Min	Typ	Max	Unit
V_{OH}	Output High Voltage		$I_{OUT} = 100\mu A$ Pins 2-16, 18, 19, 20	3.5	4.3		V
V_{OL}	Output Low Voltage		$I_{OUT} = 1.6mA$		0.2	0.4	V
	Output Leakage Current		Pins 3-16 high impedance		±.01	±1	μA
	Control I/O Pullup Current		Pins 18, 19, 20 $V_{OUT} = V^+ - 3V$ MODE input at GND		5		μA
	Control I/O Loading		$\overline{HBEN}$ Pin 19 $\overline{LSEN}$ Pin 18 (Note 4)			50	pF
V_{IH}	Input High Voltage		Pins 18-21, 26, 27 referred to GND	3.0			V
V_{IL}	Input Low Voltage		Pins 18-21, 26, 27 referred to GND			1	V
	Input Pull-up Current		Pins 26, 27 $V_{OUT} = V^+ - 3V$		5		μA
	Input Pull-up Current		Pins 17, 24 $V_{OUT} = V^+ - 3V$		25		μA
	Input Pull-down Current		Pin 21 $V_{OUT} = GND + 3V$		5		μA
O_{OH}	Oscillator Output Current	High	$V_{OUT} = 2.5V$		1		mA
O_{OL}		Low	$V_{OUT} = 2.5V$		1.5		mA
BO_{OH}	Buffered Oscillator Output Current	High	$V_{OUT} = 2.5V$		2		mA
BO_{OL}		Low	$V_{OUT} = 2.5V$		5		mA
t_W	MODE Input Pulse Width		(Note 4)	50			ns

NOTES: 1. Input voltages may exceed the supply voltages provided the input current is limited to ±100μA.
2. Due to the SCR structure inherent in the process used to fabricate these devices, connecting any digital inputs or outputs to voltages greater than V+ or less than GND may cause destructive device latchup. For this reason it is recommended that no inputs from sources other than the same power supply be applied to the ICL7109 before its power supply is established, and that in multiple supply systems the supply to the ICL7109 be activated first.
3. This limit refers to that of the package and will not be obtained during normal operation.
4. This parameter is not production tested, but is guaranteed by design.
5. Roll-over error for $T_A = -55°C$ to $+125°C$ is ±3 counts maximum.
6. A full scale voltage of 2.048V is used because a full scale voltage of 4.096V exceeds the devices Common Mode Voltage Range.
7. For Cerdip package the Ratiometric error can be −4 (Min.).

NOTE: All typical values have been characterized but are not tested.

TABLE 1: Pin Assignment and Function Description

Pin	Symbol	Description		
1	GND	Digital Ground, 0V. Ground return for all digital logic.		
2	STATUS	Output High during integrate and deintegrate until data is latched. Output Low when analog section is in Auto-Zero configuration.		
3	POL	Polarity — HI for Positive input.		All three state output data bits
4	OR	Overrange — HI if Overranged.		
5	B12	Bit 12	(Most Significant Bit)	
6	B11	Bit 11		
7	B10	Bit 10		
8	B9	Bit 9		
9	B8	Bit 8		
10	B7	Bit 7	HI = true	
11	B6	Bit 6		
12	B5	Bit 5		
13	B4	Bit 4		
14	B3	Bit 3		
15	B2	Bit 2		
16	B1	Bit 1	(Least Significant Bit)	
17	TEST	Input High — Normal Operation. Input Low — Forces all bit outputs high. Note: This input is used for test purposes only. Tie high if not used.		
18	$\overline{LBEN}$	Low Byte Enable — With Mode (Pin 21) low, and $\overline{CE/LOAD}$ (Pin 20) low, taking this pin low activates low order byte outputs B1 — B8. — With Mode (Pin 21) high, this pin serves as a low byte flag output used in handshake mode. See Figures 8, 9, 10.		
19	$\overline{HBEN}$	High Byte Enable — With Mode (Pin 21) low, and $\overline{CE/LOAD}$ (Pin 20) low, taking this pin low activates high order byte outputs B9 — B12, POL, OR. — With Mode (Pin 21) high, this pin serves as a high byte flag output used in handshake mode. See Figures 8, 9, 10.		
20	$\overline{CE/LOAD}$	Chip Enable Load — With Mode (Pin 21) low, $\overline{CE/LOAD}$ serves as a master output enable. When high, B1 — B12, POL, OR outputs are disabled. — With Mode (Pin 21) high, this pin serves as a load strobe used in handshake mode. See Figures 8, 9, 10.		

Pin	Symbol	Description
21	MODE	Input Low — Direct output mode where $\overline{CE/LOAD}$ (Pin 20), $\overline{HBEN}$ (Pin 19) and $\overline{LBEN}$ (Pin 18) act as inputs directly controlling byte outputs. Input Pulsed High — Causes immediate entry into handshake mode and output of data as in Figure 10. Input High — Enables $\overline{CE/LOAD}$ (Pin 20), $\overline{HBEN}$ (Pin 19), and $\overline{LBEN}$ (Pin 18) as outputs, handshake mode will be entered and data output as in Figures 8 and 9 at conversion completion.
22	OSC IN	Oscillator Input
23	OSC OUT	Oscillator Output
24	OSC SEL	Oscillator Select — Input high configures OSC IN, OSC OUT, BUF OSC OUT as RC oscillator — clock will be same phase and duty cycle as BUF OSC OUT. — Input low configures OSC IN, OSC OUT for crystal oscillator — clock frequency will be 1/58 of frequency at BUF OSC OUT.
25	BUF OSC OUT	Buffered Oscillator Output
26	RUN/$\overline{HOLD}$	Input High — Conversions continuously performed every 8192 clock pulses. Input Low — Conversion in progress completed, converter will stop in Auto-Zero 7 counts before integrate.
27	SEND	Input — Used in handshake mode to indicate ability of an external device to accept data. Connect to + 5V if not used.
28	V−	Analog Negative Supply — Nominally − 5V with respect to GND (Pin 1).
29	REF OUT	Reference Voltage Output — Nominally 2.8V down from V+ (Pin 40).
30	BUFFER	Buffer Amplifier Output
31	AUTO-ZERO	Auto-Zero Node — Inside foil of C_{AZ}
32	INTEGRATOR	Integrator Output — Outside foil of C_{INT}
33	COMMON	Analog Common — System is Auto-Zeroed to COMMON
34	INPUT LO	Differential Input Low Side
35	INPUT HI	Differential Input High Side
36	REF IN +	Differential Reference Input Positive
37	REF CAP +	Reference Capacitor Positive
38	REF CAP −	Reference Capacitor Negative
39	REF IN −	Differential Reference Input Negative
40	V+	Positive Supply Voltage — Nominally + 5V with respect to GND (Pin 1).

Note: All digital levels are positive true.

NOTE: All typical values have been characterized but are not tested.

ICL7109

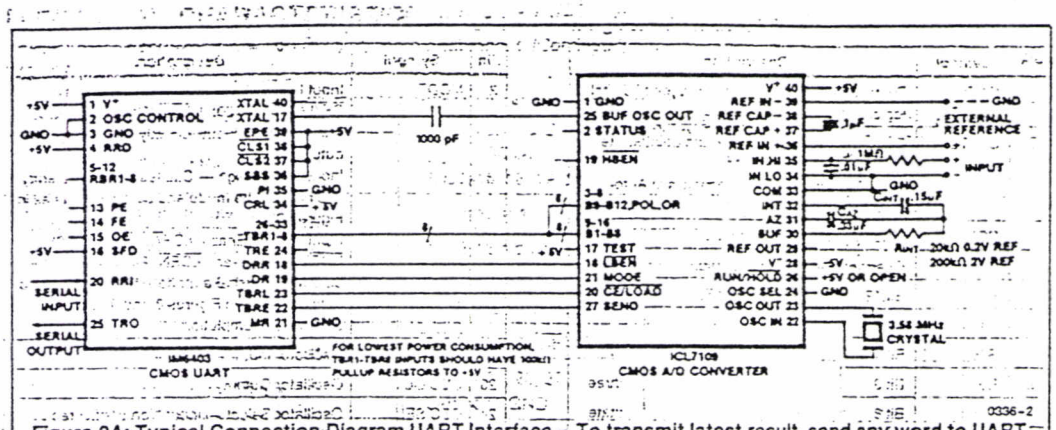


Figure 2A: Typical Connection Diagram UART Interface – To transmit latest result, send any word to UART

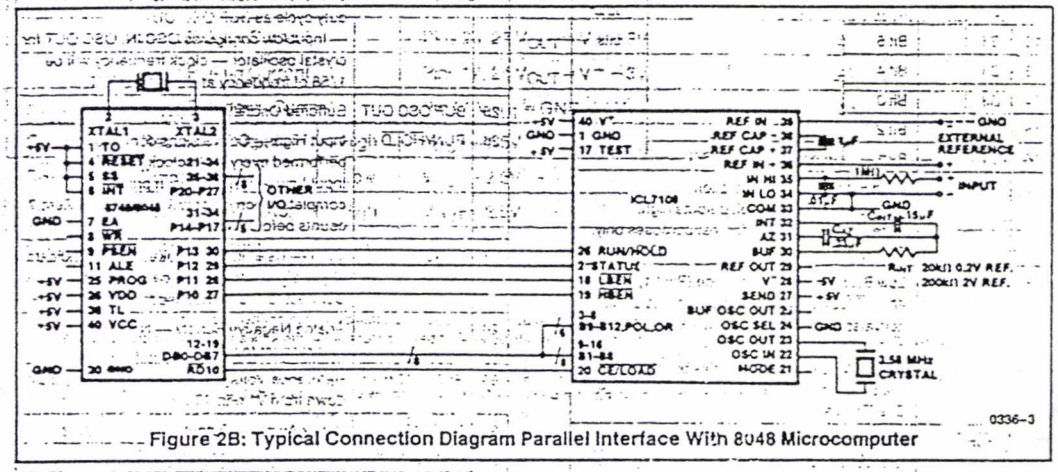


Figure 2B: Typical Connection Diagram Parallel Interface With 8048 Microcomputer

DETAILED DESCRIPTION

Analog Section

Figure 3 shows the equivalent circuit of the Analog Section of the ICL7109. When the RUN/HOLD input is left open or connected to V+, the circuit will perform conversions at a rate determined by the clock frequency (8192 clock periods per cycle). Each measurement cycle is divided into three phases as shown in Figure 4. They are (1) Auto-Zero (AZ), (2) Signal Integrate (INT) and (3) Deintegrate (DE).

Auto-Zero Phase

During auto-zero three things happen. First, input high and low are disconnected from their pins and internally shorted to analog COMMON. Second, the reference capacitor is charged to the reference voltage. Third, a feedback loop is closed around the system to charge the auto-zero capacitor C_{AZ} to compensate for offset voltages in the buffer amplifier, integrator, and comparator. Since the comparator is included in the loop, the AZ accuracy is limited only by the noise of the system. In any case, the offset referred to the input is less than 10µV.

NOTE: All typical values have been characterized but are not tested.

ICL7109

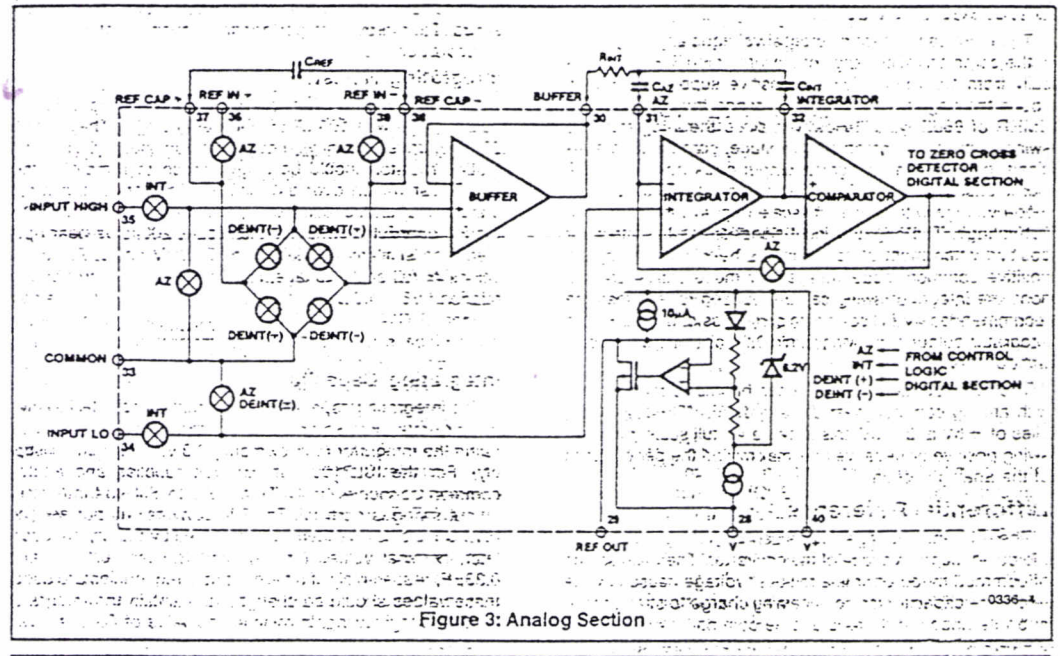


Figure 3: Analog Section

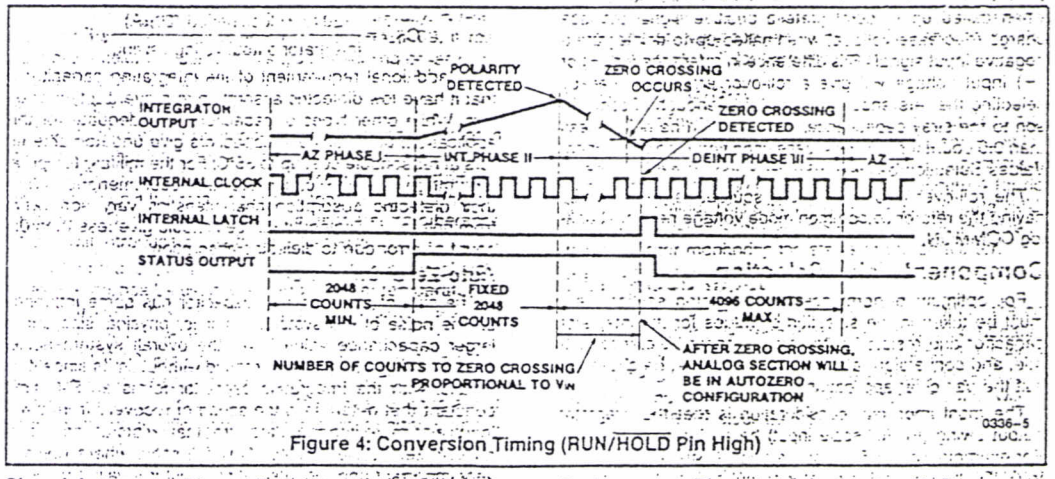


Figure 4: Conversion Timing (RUN/HOLD Pin High)

Signal Integrate Phase

During signal integrate, the auto-zero loop is opened, the internal short is removed and the internal high and low inputs are connected to the external pins. The converter then integrates the differential voltage between IN HI and IN LO for a fixed time of 2048 clock periods. Note that this differential voltage must be within the common mode range of the inputs. At the end of this phase, the polarity of the integrated signal is determined.

De-Integrate Phase

The final phase is de-integrate, or reference integrate. Input low is internally connected to analog COMMON and input high is connected across the previously charged (during auto-zero) reference capacitor. Circuitry within the chip ensures that the capacitor will be connected with the correct polarity to cause the integrator output to return to zero crossing (established in Auto Zero) with a fixed slope. Thus the time for the output to return to zero (represented by the number of clock periods counted) is proportional to the input signal.

NOTE: All typical values have been characterized but are not tested.

Differential Input

The input can accept differential voltages anywhere within the common mode range of the input amplifier, or specifically from 1.0 volts below the positive supply, to 1.5 volts above the negative supply. In this range the system has a CMRR of 86dB typical. However, since the integrator also swings with the common mode voltage, care must be exercised to assure the integrator output does not saturate. A worst case condition would be a large positive common mode voltage with a near full-scale negative differential input voltage. The negative input signal drives the integrator positive when most of its swing has been used up by the positive common mode voltage. For these critical applications the integrator swing can be reduced to less than the recommended 4V full scale with some loss of accuracy. The integrator output can swing within 0.3 volts of either supply without loss of linearity.

The ICL7109 has, however, been optimized for operation with analog common near digital ground. With power supplies of +5V and −5V, this allows a 4V full scale integrator swing positive or negative thus maximizing the performance of the analog section.

Differential Reference

The reference voltage can be generated anywhere within the power supply voltage of the converter. The main source of common mode error is a roll-over voltage caused by the reference capacitor losing or gaining charge to stray capacity on its nodes. If there is a large common mode voltage, the reference capacitor can gain charge (increase voltage) when called up to deintegrate a positive signal but lose charge (decrease voltage) when called up to deintegrate a negative input signal. This difference in reference for (+) or (−) input voltage will give a roll-over error. However, by selecting the reference capacitor large enough in comparison to the stray capacitance, this error can be held to less than 0.5 count for the worst case condition (see Component Values Selection below).

The roll-over error from these sources is minimized by having the reference common mode voltage near or at analog COMMON.

Component Value Selection

For optimum performance of the analog section, care must be taken in the selection of values for the integrator capacitor and resistor, auto-zero capacitor, reference voltage, and conversion rate. These values must be chosen to suit the particular application.

The most important consideration is that the integrator output swing (for full-scale input) be as large as possible. For example, with ±5V supplies and COMMON connected to GND, the nominal integrator output swing at full scale is ±4V. Since the integrator output can go to 0.3V from either supply without significantly affecting linearity, a 4V integrator output swing allows 0.7V for variations in output swing due to component value and oscillator tolerances. With ±5V supplies and a common mode range of ±1V required, the component values should be selected to provide ±3V integrator output swing. Noise and rollover errors will be slightly worse than in the ±4V case. For larger common mode voltage ranges, the integrator output swing must be reduced further. This will increase both noise and rollover errors. To improve the performance, supplies of ±6V may be used.

Integrating Resistor

Both the buffer amplifier and the integrator have a class A output stage with 100μA of quiescent current. They supply 20μA of drive current with negligible non-linearity. The integrating resistor should be large enough to remain in this very linear region over the input voltage range, but small enough that undue leakage requirements are not placed on the PC board. For 4.096 volt full scale, 200kΩ is near optimum and similarly a 20kΩ for a 409.6mV scale. For other values of full scale voltage, R_{INT} should be chosen by the relation

$$R_{INT} = \frac{\text{full scale voltage}}{20\mu A}$$

Integrating Capacitor

The integrating capacitor C_{INT} should be selected to give the maximum integrator output voltage swing without saturating the integrator (approximately 0.3 volt from either supply). For the ICL7109 with ±5 volt supplies and analog common connected to GND, a ±3.5 to ±4 volt integrator output swing is nominal. For 7-½ conversions per second (61.72kHz clock frequency) as provided by the crystal oscillator, nominal values for C_{INT} and C_{AZ} are 0.15μF and 0.33μF, respectively. If different clock frequencies are used, these values should be changed to maintain the integrator output voltage swing. In general, the value of C_{INT} is given by

$$C_{INT} = \frac{(2048 \times \text{clock period})(20\mu A)}{\text{integrator output voltage swing}}\ \mu F$$

An additional requirement of the integrating capacitor is that it have low dielectric absorption to prevent roll-over errors. While other types of capacitors are adequate for this application, polypropylene capacitors give undetectable errors at reasonable cost up to 85°C. For the military temperature range, Teflon® capacitors are recommended. While their dielectric absorption characteristics vary somewhat from unit to unit, selected devices should give less than 0.5 count of error due to dielectric absorption.

Auto-Zero Capacitor

The size of the auto-zero capacitor has some influence on the noise of the system: a smaller physical size and a larger capacitance value lower the overall system noise. However, C_{AZ} cannot be increased without limits since it, in parallel with the integrating capacitor forms an R-C time constant that determines the speed of recovery from overloads and more important the error that exists at the end of an auto-zero cycle. For 409.6mV full scale where noise is very important and the integrating resistor small, a value of C_{AZ} twice C_{INT} is optimum. Similarly for 4.096V full scale where recovery is more important than noise, a value of C_{AZ} equal to half of C_{INT} is recommended.

For optimal rejection of stray pickup, the outer foil of C_{AZ} should be connected to the R-C summing junction and the inner foil to pin 31. Similarly the outer foil of C_{INT} should be connected to pin 32 and the inner foil to the R-C summing junction. Teflon®, or equivalent, capacitors are recommended above 85°C for their low leakage characteristics.

NOTE: All typical values have been characterized but are not tested.

Reference Capacitor

A 1µF capacitor gives good results in most applications. However, where a large reference common mode voltage exists (i.e. the reference low is not at analog common) and a 409.6mV scale is used, a larger value is required to prevent roll-over error. Generally 10µF will hold the roll-over error to 0.5 count in this instance. Again, Teflon®, or equivalent capacitors should be used for temperatures above 85°C for their low leakage characteristics.

Reference Voltage

The analog input required to generate a full scale output of 4096 counts is $V_{IN} = 2V_{REF}$. Thus for a normalized scale, a reference of 2.048V should be used for a 4.096V full scale, and 204.8mV should be used for a 0.4096V full scale. However, in many applications where the A/D is sensing the output of a transducer, there will exist a scale factor other than unity between the absolute output voltage to be measured and a desired digital output. For instance, in a weighing system, the designer might like to have a full scale reading when the voltage from the transducer is 0.682V. Instead of dividing the input down to 409.6mV, the input voltage should be measured directly and a reference voltage of 0.341V should be used. Suitable values for integrating resistor and capacitor are 33kΩ and 0.15µF. This avoids a divider on the input. Another advantage of this system occurs when a zero reading is desired for non-zero input. Temperature and weight measurements with an offset or tare are examples. The offset may be introduced by connecting the voltage output of the transducer between common and analog high, and the offset voltage between common and analog low, observing polarities carefully. However, in processor-based systems using the ICL7109, it may be more efficient to perform this type of scaling or tare subtraction digitally using software.

Reference Sources

The stability of the reference voltage is a major factor in the overall absolute accuracy of the converter. The resolution of the ICL7109 at 12 bits is one part in 4096, or 244ppm. Thus if the reference has a temperature coefficient of 80ppm/°C (onboard reference) a temperature difference of 3°C will introduce a one-bit absolute error.

For this reason, it is recommended that an external high-quality reference be used where the ambient temperature is not controlled or where high-accuracy absolute measurements are being made.

The ICL7109 provides a REFerence OUTput (pin 29) which may be used with a resistive divider to generate a suitable reference voltage. This output will sink up to about 20mA without significant variation in output voltage, and is provided with a pullup bias device which sources about 10µA. The output voltage is nominally 2.8V below V+, and has a temperature coefficient of ±80ppm/°C typ. When using the onboard reference, REF OUT (Pin 29) should be connected to REF− (pin 39), and REF+ should be connected to the wiper of a precision potentiometer between REF OUT and V+. The circuit for a 204.8mV reference is shown in the test circuit. For a 2.048mV reference, the fixed resistor should be removed, and a 25kΩ precision potentiometer between REF OUT and V+ should be used.

Note that if pins 29 and 39 are tied together and pins 39 and 40 accidentally shorted (e.g., during testing), the reference supply will sink enough current to destroy the device. This can be avoided by placing a 1kΩ resistor in series with pin 39.

DETAILED DESCRIPTION

Digital Section

The digital section includes the clock oscillator and scaling circuit, a 12-bit binary counter with output latches and TTL-compatible three-state output drivers, polarity, overrange and control logic, and UART handshake logic, as shown in Figure 5.

Throughout this description, logic levels will be referred to as "low" or "high". The actual logic levels are defined in the Electrical Characteristics Table. For minimum power consumption, all inputs should swing from GND (low) to V+ (high). Inputs driven from TTL gates should have 3-5kΩ pullup resistors added for maximum noise immunity.

MODE Input

The MODE input is used to control the output mode of the converter. When the MODE pin is low or left open (this input is provided with a pulldown resistor to ensure a low level when the pin is left open), the converter is in its "Direct" output mode, where the output data is directly accessible under the control of the chip and byte enable inputs. When the MODE input is pulsed high, the converter enters the UART handshake mode and outputs the data in two bytes, then returns to "direct" mode. When the MODE input is left high, the converter will output data in the handshake mode at the end of every conversion cycle. (See section entitled "Handshake Mode" for further details).

STATUS Output

During a conversion cycle, the STATUS output goes high at the beginning of Signal Integrate (Phase II), and goes low one-half clock period after new data from the conversion has been stored in the output latches. See Figure 4 for details of this timing. This signal may be used as a "data valid" flag (data never changes while STATUS is low) to drive interrupts, or for monitoring the status of the converter.

RUN/HOLD Input

When the RUN/$\overline{\text{HOLD}}$ input is high, or left open, the circuit will continuously perform conversion cycles, updating the output latches after zero crossing during the Deintegrate (Phase III) portion of the conversion cycle (See Figure 4). In this mode of operation, the conversion cycle will be performed in 8192 clock periods, regardless of the resulting value.

If RUN/$\overline{\text{HOLD}}$ goes low at any time during Deintegrate (Phase III) after the zero crossing has occurred, the circuit will immediately terminate Deintegrate and jump to Auto-Zero. This feature can be used to eliminate the time spent in Deintegrate after the zero-crossing. If RUN/$\overline{\text{HOLD}}$ stays or goes low, the converter will ensure minimum Auto-Zero time, and then wait in Auto-Zero until the RUN/$\overline{\text{HOLD}}$ input goes high. The converter will begin the Integrate (Phase II) portion of the next conversion (and the STATUS output will go high) seven clock periods after the high level is detected at RUN/$\overline{\text{HOLD}}$. See Figure 6 for details.

NOTE: All typical values have been characterized but are not tested.

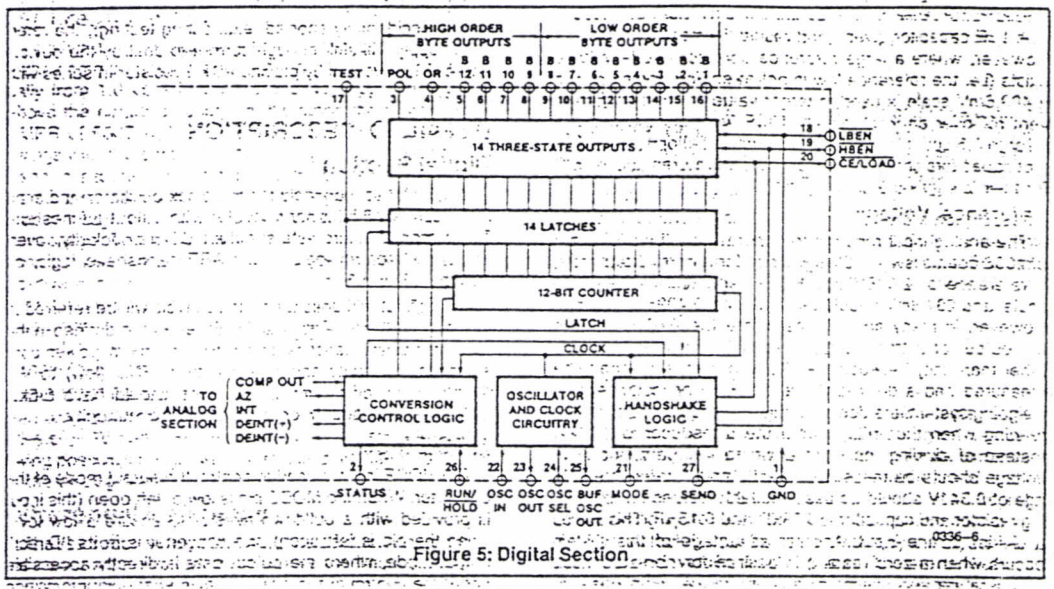


Figure 5: Digital Section

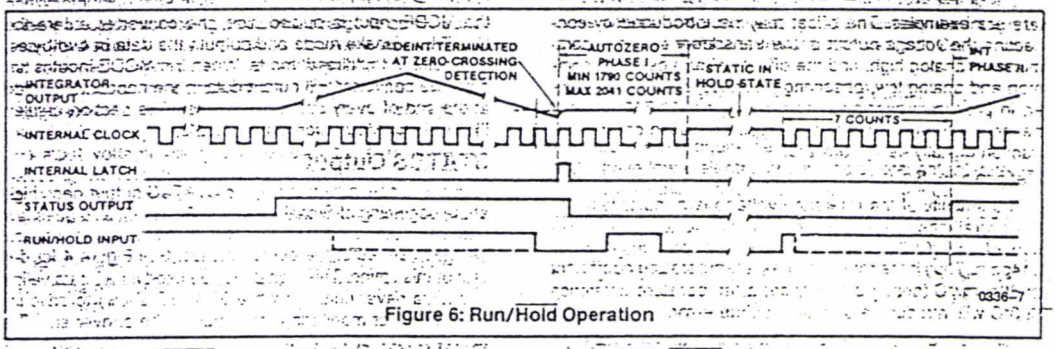


Figure 6: Run/Hold Operation

Using the RUN/HOLD input in this manner allows an easy "convert on demand" interface to be used. The converter may be held at idle in auto-zero with RUN/HOLD low. When RUN/HOLD goes high the conversion is started, and when the STATUS output goes low the new data is valid (or transferred to the UART — see Handshake Mode). RUN/HOLD may now be taken low which terminates deintegrate and ensures a minimum Auto-Zero time before the next conversion.

Alternately, RUN/HOLD can be used to minimize conversion time by ensuring that it goes low during Deintegrate, after zero crossing, and goes high after the hold point is reached. The required activity on the RUN/HOLD input can be provided by connecting it to the Buffered Oscillator Output. In this mode the conversion time is dependent on the input value measured. Also refer to Harris Application Bulletin, A032 for a discussion of the effects this will have on Auto-Zero performance.

If the RUN/HOLD input goes low and stays low during Auto-Zero (Phase I), the converter will simply stop at the end of Auto-Zero and wait for RUN/HOLD to go high. As above, Integrate (Phase II) begins seven clock periods after the high level is detected.

Direct Mode

When the MODE pin is left at a low level, the data outputs (bits 1 through 8 low order byte, bits 9 through 12, polarity and over-range high order byte) are accessible under control of the byte and chip enable terminals as inputs. These three inputs are all active low, and are provided with pullup resistors to ensure an inactive high level when left open. When the chip enable input is low, taking a byte enable input low will allow the outputs of that byte to become active (three-stated on). This allows a variety of parallel data accessing techniques to be used, as shown in the section entitled "Interfacing." The timing requirements for these outputs are shown in Figure 7 and Table 2.

NOTE: All typical values have been characterized but are not tested.

Table 2 — Direct Mode Timing Requirements
(See Note 4 of Electrical Characteristics)

SYMBOL	DESCRIPTION	MIN	TYP	MAX	UNIT
t_{BEA}	Byte Enable Width	350	220		ns
t_{DAB}	Data Access Time from Byte Enable		210	350	ns
t_{DHB}	Data Hold Time from Byte Enable	150	300		ns
t_{CEA}	Chip Enable Width	400	260		ns
t_{DAC}	Data Access Time from Chip Enable		260	400	ns
t_{DHC}	Data Hold Time from Chip Enable		240	400	ns

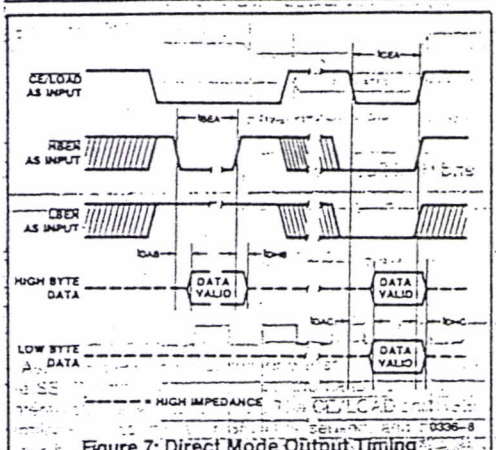


Figure 7: Direct Mode Output Timing

It should be noted that these control inputs are asynchronous with respect to the converter clock — the data may be accessed at any time. Thus it is possible to access the latches while they are being updated, which could lead to erroneous data. Synchronizing the access of the latches with the conversion cycle by monitoring the STATUS output will prevent this. Data is never updated while STATUS is low.

Handshake Mode

The handshake output mode is provided as an alternative means of interfacing the ICL7109 to digital systems, where the A/D converter becomes active in controlling the flow of data instead of passively responding to chip and byte enable inputs. This mode is specifically designed to allow a direct interface between the ICL7109 and industry-standard UARTs (such as the Harris IM6402/3) with no external logic required. When triggered into the handshake mode, the ICL7109 provides all the control and flag signals necessary to sequentially transfer two bytes of data into the UART and initiate their transmission in serial form. This greatly eases the task and reduces the cost of designing remote data acquisition stations using serial data transmission.

Entry into the handshake mode is controlled by the MODE pin. When the MODE terminal is held high, the ICL7109 will enter the handshake mode after new data has been stored in the output latches at the end of a conversion (See Figures 8 and 9). The MODE terminal may also be used to trigger entry into the handshake mode on demand. At any time during the conversion cycle, the low to high transition of a short pulse at the MODE input will cause immediate entry into the handshake mode. If this pulse occurs while new data is being stored, the entry into handshake mode is delayed until the data is stable. While the converter is in the handshake mode, the MODE input is ignored and although conversions will still be performed, data updating will be inhibited (See Figure 10) until the converter completes the output cycle and clears the handshake mode.

When the converter enters the handshake mode, or when the MODE input is high, the chip and byte enable terminals become TTL-compatible outputs which provide the control signals for the output cycle (See Figures 8, 9, and 10).

In handshake mode, the SEND input is used by the converter as an indication of the ability of the receiving device (such as a UART) to accept data.

Figure 8 shows the sequence of the output cycle with SEND held high. The handshake mode (Internal MODE high) is entered after the data latch pulse, and since MODE remains high the CE/LOAD, LBEN and HBEN terminals are active as outputs. The high level at the SEND input is sensed on the same high to low internal clock edge that terminates the data latch pulse. On the next low to high internal clock edge the CE/LOAD and the HBEN outputs assume a low level, and the high-order byte (bits 9 through 12, POL, and OR) outputs are enabled. The CE/LOAD output remains low for one full internal clock period only, the data outputs remain active for 1-½ internal clock periods, and the high byte enable remains low for two clock periods. Thus the CE/LOAD output low level or low to high edge may be used as a synchronizing signal to ensure valid data, and the byte enable as an output may be used as a byte identification flag. With SEND remaining high the converter completes the output cycle using CE/LOAD and LBEN while the low order byte outputs (bits 1 through 8) are activated. The handshake mode is terminated when both bytes are sent.

Figure 9 shows an output sequence where the SEND input is used to delay portions of the sequence, or handshake to ensure correct data transfer. This timing diagram shows the relationships that occur using an industry-standard IM6402/3 CMOS UART to interface to serial data channels. In this interface, the SEND input to the ICL7109 is driven by the TBRE (Transmitter Buffer Register Empty) output of the UART, and the CE/LOAD terminal of the ICL7109 drives the TBRL (Transmitter Buffer Register Load) input to the UART. The data outputs are paralleled into the eight Transmitter Buffer Register inputs.

3

NOTE: All typical values have been characterized but are not tested.

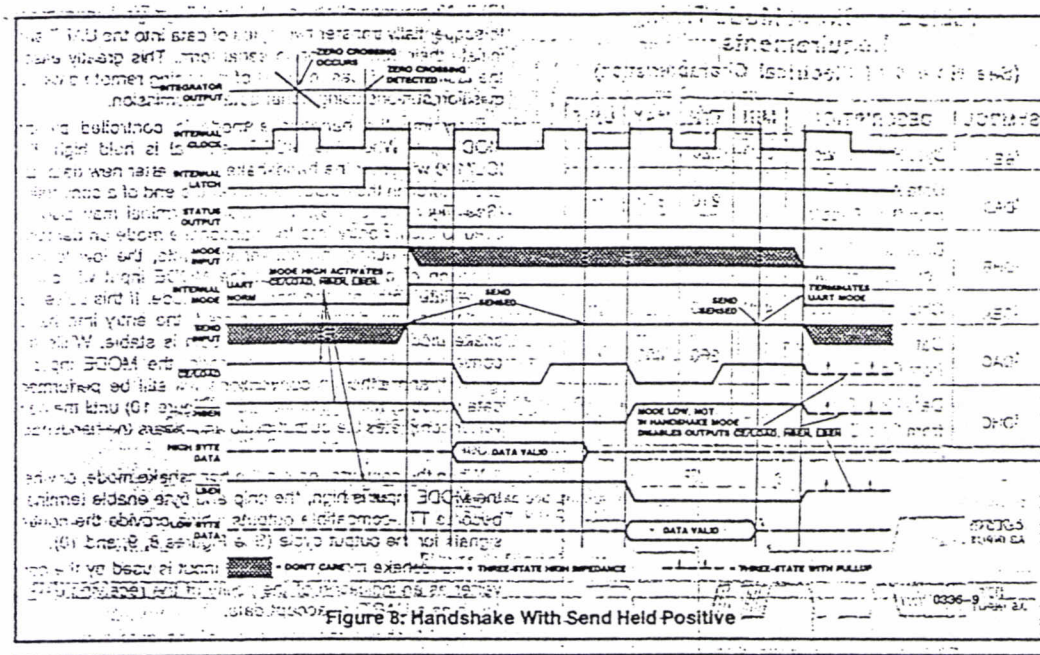


Figure 8: Handshake With Send Held Positive

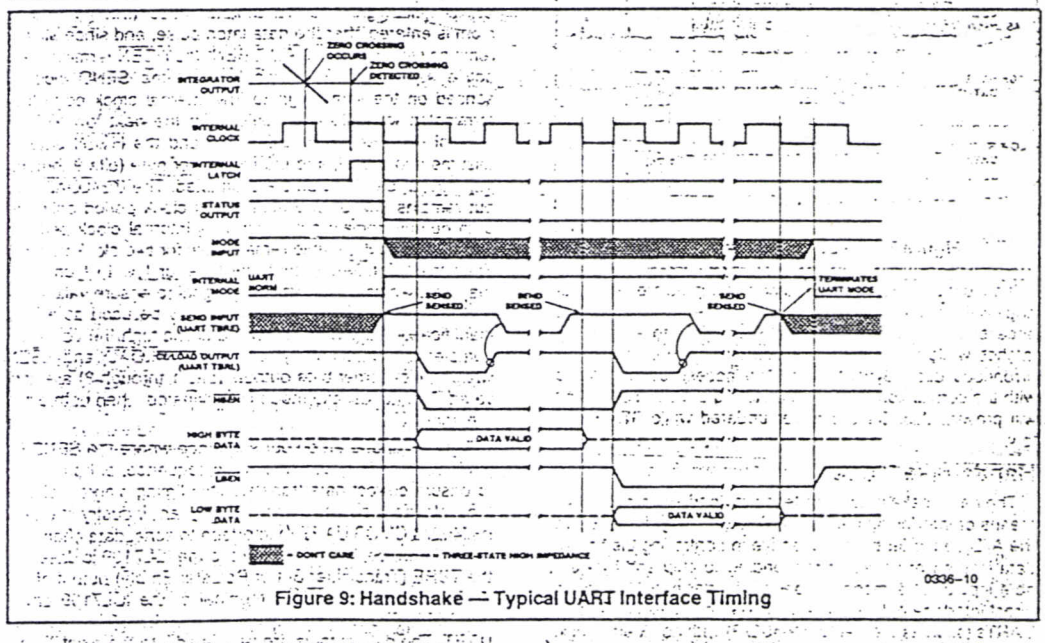


Figure 9: Handshake — Typical UART Interface Timing

NOTE: All typical values have been characterized but are not tested.

ICL7109

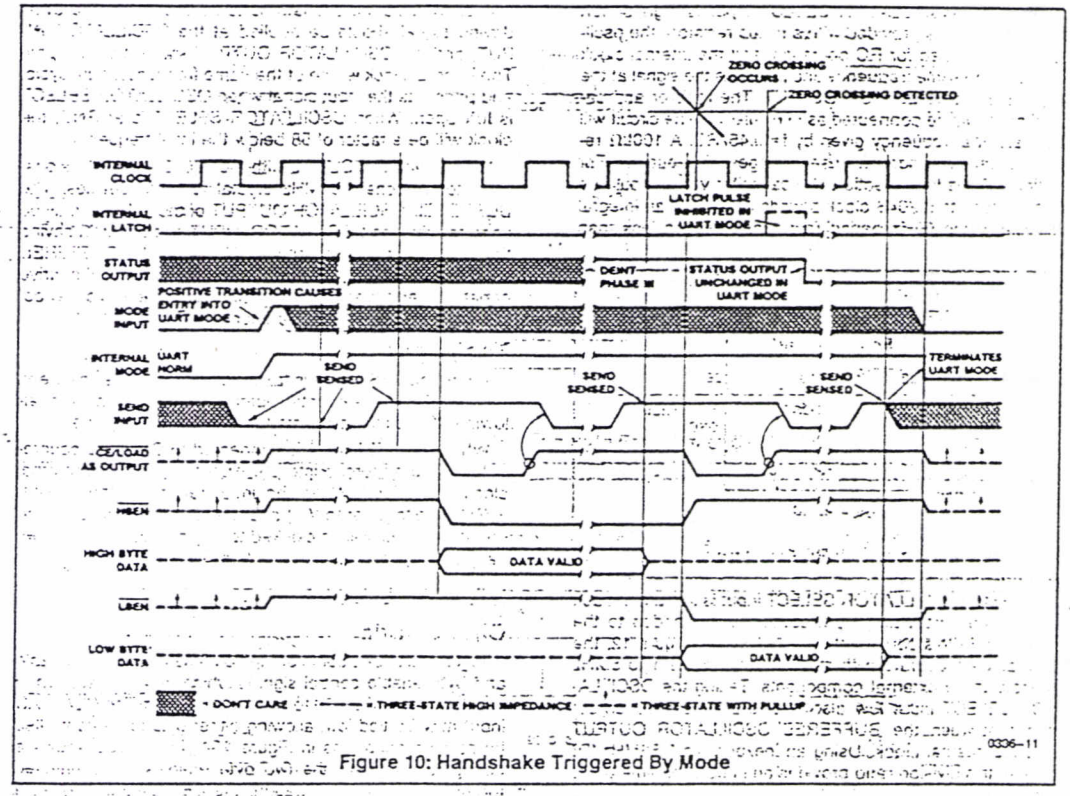


Figure 10: Handshake Triggered By Mode

Assuming the UART Transmitter Buffer Register is empty, the SEND input will be high when the handshake mode is entered after new data is stored. The $\overline{CE/LOAD}$ and $\overline{HBEN}$ terminals will go low after SEND is sensed, and the high order byte outputs become active. When $\overline{CE/LOAD}$ goes high at the end of one clock period, the high order byte data is clocked into the UART Transmitter Buffer Register. The UART TBRE output will now go low, which halts the output cycle with the $\overline{HBEN}$ output low, and the high order byte outputs active. When the UART has transferred the data to the Transmitter Register and cleared the Transmitter Buffer Register, the TBRE returns high. On the next ICL7109 internal clock high to low edge, the high order byte outputs are disabled, and one-half internal clock later, the $\overline{HBEN}$ output returns high. At the same time, the $\overline{CE/LOAD}$ and $\overline{LBEN}$ outputs go low, and the low order byte outputs become active. Similarly, when the $\overline{CE/LOAD}$ returns high at the end of one clock period, the low order data is clocked into the UART Transmitter Buffer Register, and TBRE again goes low. When TBRE returns to a high it will be sensed on the next ICL7109 internal clock high to low edge, disabling the data outputs. One-half internal clock later, the handshake mode will be cleared, and the $\overline{CE/LOAD}$, $\overline{HBEN}$, and $\overline{LBEN}$ terminals return high and stay active (as long as MODE stays high).

With the MODE input remaining high as in these examples, the converter will output the results of every conversion except those completed during a handshake operation. By triggering the converter into handshake mode with a low to high edge on the MODE input, handshake output sequences may be performed on demand. Figure 10 shows a handshake output sequence triggered by such an edge. In addition, the SEND input is shown as being low when the converter enters handshake mode. In this case, the whole output sequence is controlled by the SEND input, and the sequence for the first (high order) byte is similar to the sequence for the second byte. This diagram also shows the output sequence taking longer than a conversion cycle. Note that the converter still makes conversions, with the STATUS output and RUN/$\overline{HOLD}$ input functioning normally. The only difference is that new data will not be latched when in handshake mode, and is therefore lost.

Oscillator

The ICL7109 is provided with a versatile three terminal oscillator to generate the internal clock. The oscillator may be overdriven, or may be operated with an RC network or crystal. The OSCILLATOR SELECT input changes the internal configuration of the oscillator to optimize it for RC or crystal operation.

NOTE: All typical values have been characterized but are not tested.

When the OSCILLATOR SELECT input is high or left open (the input is provided with a pullup resistor), the oscillator is configured for RC operation, and the internal clock will be of the same frequency and phase as the signal at the BUFFERED OSCILLATOR OUTPUT. The resistor and capacitor should be connected as in Figure 11. The circuit will oscillate at a frequency given by $f = 0.45/RC$. A 100kΩ resistor is recommended for useful ranges of frequency. For optimum 60Hz line rejection, the capacitor value should be chosen such that 2048 clock periods is close to an integral multiple of the 60Hz period (but should not be less than 50pF).

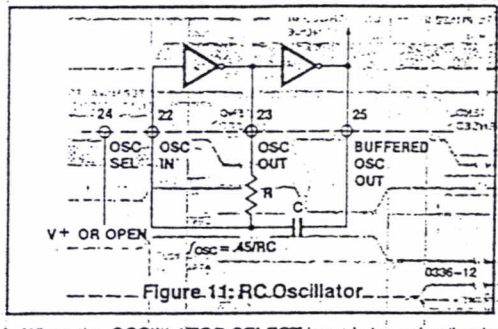


Figure 11: RC Oscillator

When the OSCILLATOR SELECT input is low a feedback device and output and input capacitors are added to the oscillator. In this configuration, as shown in Figure 12, the oscillator will operate with most crystals in the 1 to 5MHz range with no external components. Taking the OSCILLATOR SELECT input low also inserts a fixed ÷58 divider circuit between the BUFFERED OSCILLATOR OUTPUT and the internal clock. Using an inexpensive 3.58MHz TV crystal, this division ratio provides an integration time given by:

$$T_{INT} = (2048 \text{ clock periods}) \times (T_{CLOCK}) = 33.18 \text{ ms}$$

$$\text{where } T_{CLOCK} = \frac{58}{3.58 \text{ MHz}}$$

This time is very close to two 60Hz periods or 33.33ms. The error is less than one percent, which will give better than 40dB 60Hz rejection. The converter will operate reliably at conversion rates of up to 30 per second, which corresponds to a clock frequency of 245.8kHz.

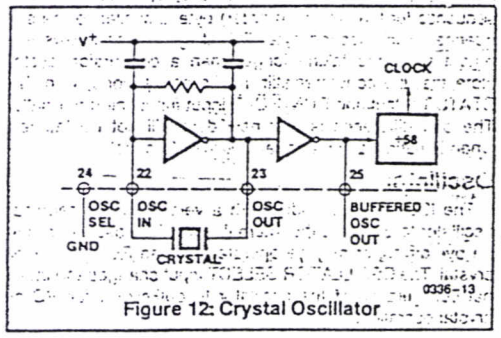


Figure 12: Crystal Oscillator

If at any time the oscillator is to be overdriven, the overdriving signal should be applied at the OSCILLATOR INPUT, and the OSCILLATOR OUTPUT should be left open. The internal clock will be of the same frequency, duty cycle, and phase as the input signal when OSCILLATOR SELECT is left open. When OSCILLATOR SELECT is at GND, the clock will be a factor of 58 below the input frequency.

When using the ICL7109 with the IM6403 UART, it is possible to use one 3.58MHz crystal for both devices. The BUFFERED OSCILLATOR OUTPUT of the ICL7109 may be used to drive the OSCILLATOR INPUT of the UART, saving the need for a second crystal. However, the BUFFERED OSCILLATOR OUTPUT does not have a great deal of drive capability, and when driving more than one slave device, external buffering should be used.

Test Input

When the TEST input is taken to a level halfway between V+ and GND, the counter output latches are enabled, allowing the counter contents to be examined anytime.

When the TEST input is connected to GND, the counter outputs are all forced into the high state, and the internal clock is disabled. When the input returns to the ½ (V+ – GND) voltage (or to V+) and one clock is applied, all the counter outputs will be clocked to the low state. This allows easy testing of the counter and its outputs.

INTERFACING

Direct Mode

Figure 13 shows some of the combinations of chip enable and byte enable control signals which may be used when interfacing the ICL7109 to parallel data lines. The CE/LOAD input may be tied low, allowing either byte to be controlled by its own enable as in Figure 13A. Figure 13B shows a configuration where the two byte enables are connected together. In this configuration, the CE/LOAD serves as a chip enable, and the HBEN and LBEN may be connected to GND or serve as a second chip enable. The 14 data outputs will all be enabled simultaneously. Figure 13C shows the HBEN and LBEN as flag inputs, and CE/LOAD as a master enable, which could be the READ strobe available from most microprocessors.

Figure 14 shows an approach to interfacing several ICL7109s to a bus, ganging the HBEN and LBEN signals to several converters together, and using the CE/LOAD inputs (perhaps decoded from an address) to select the desired converter.

NOTE: All typical values have been characterized but are not tested.

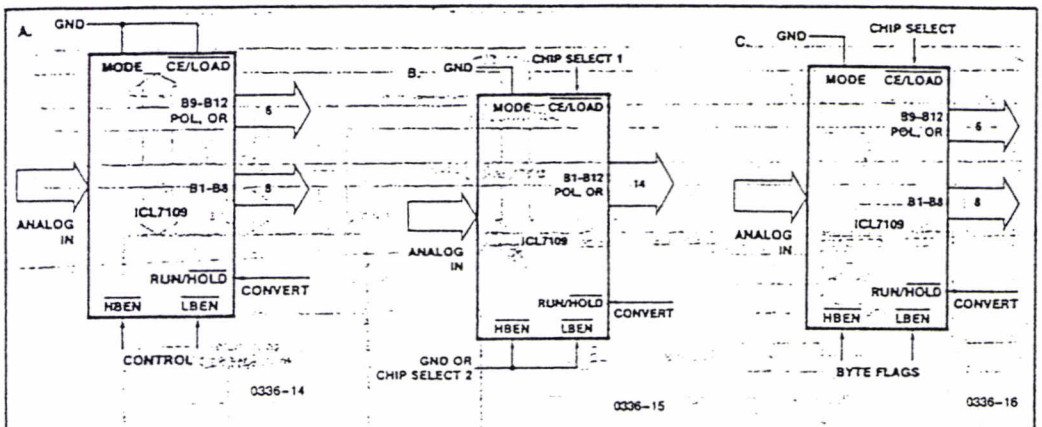


Figure 13: Direct Mode Chip and Byte Enable Combinations

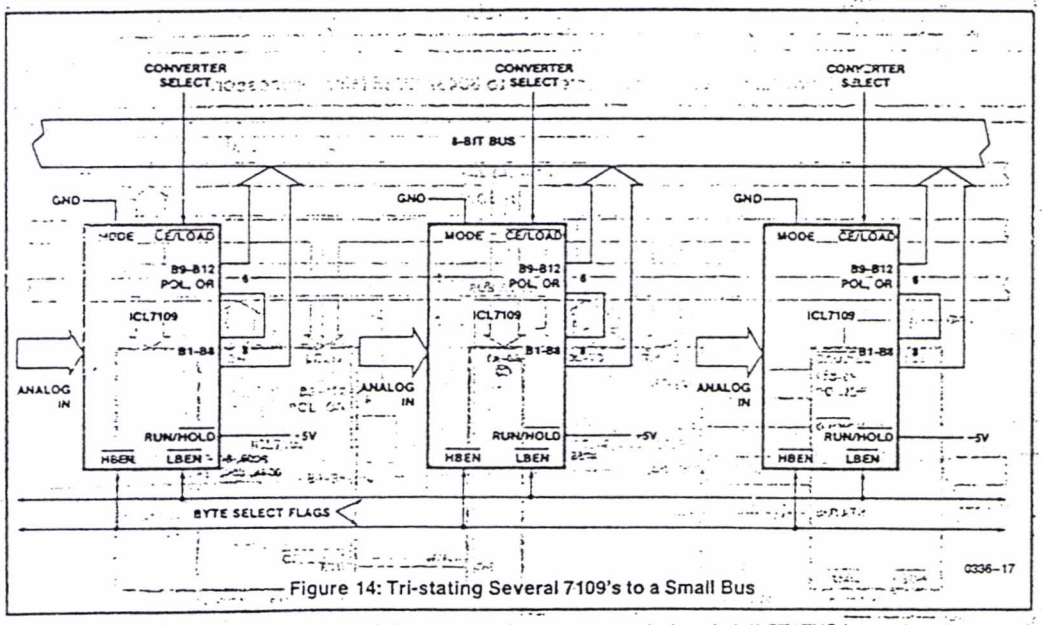


Figure 14: Tri-stating Several 7109's to a Small Bus

Some practical circuits utilizing the parallel three-state output capabilities of the ICL7109 are shown in Figures 15 through 20. Figure 15 shows a straightforward application to the Intel 8048/80/85 microprocessors via an 8255PPI, where the ICL7109 data outputs are active at all times. The I/O ports of an 8155 may be used in the same way. This interface can be used in a read-anytime mode, although a read performed while the data latches are being updated will lead to scrambled data. This will occur very rarely, in the proportion of setup-skew times to conversion time. One way to overcome this is to read the STATUS output as well, and if it is high, read the data again after a delay of more than ½ converter clock period. If STATUS is now low, the second reading is correct, and if it is still high, the first reading is correct. Alternatively, this timing problem is completely avoided by using a read-after-update sequence, as shown in Figure 16. Here the high to low transition of the STATUS output drives an interrupt to the microprocessor causing it to access the data latches. This application also shows the RUN/HOLD input being used to initiate conversions under software control.

A similar interface to Motorola MC6800 or Rockwell R650X systems is shown in Figure 17. The high to low transition of the STATUS output generates an interrupt via the

NOTE: All typical values have been characterized but are not tested.

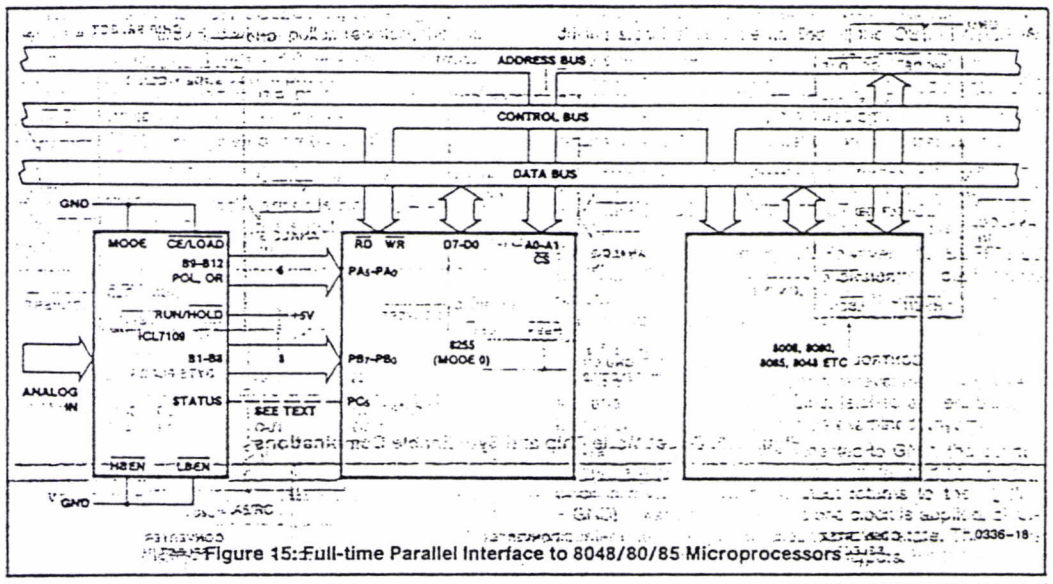


Figure 15: Full-time Parallel Interface to 8048/80/85 Microprocessors

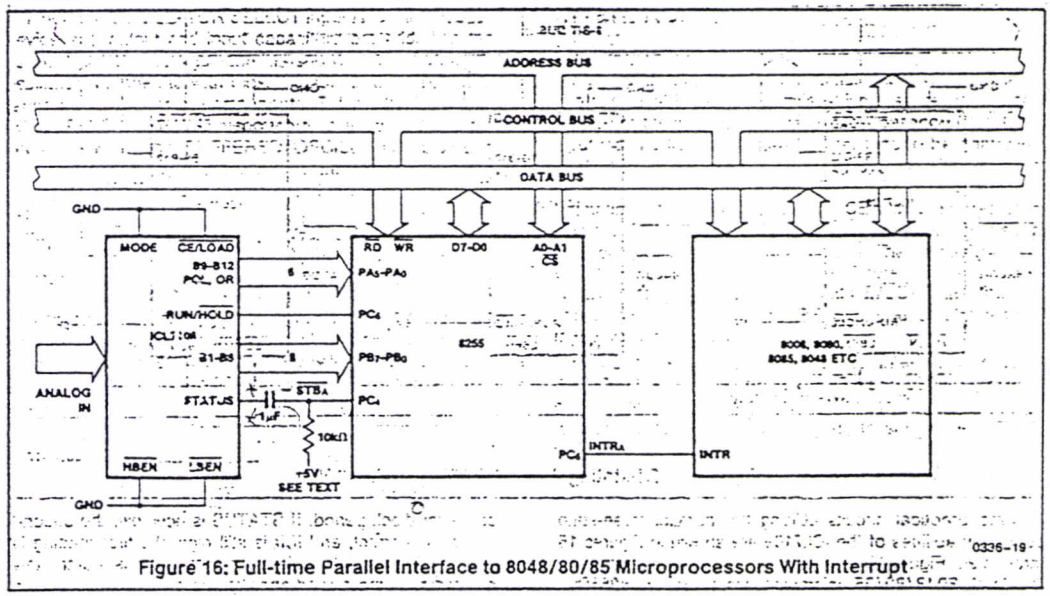


Figure 16: Full-time Parallel Interface to 8048/80/85 Microprocessors With Interrupt

Control Register B CB1 line. Note that CB2 controls the RUN/HOLD pin through Control Register B, allowing software-controlled initiation of conversions in this system as well.

The three-state output capability of the ICL7109 allows direct interfacing to most microprocessor busses. Examples of this are shown in Figures 18 and 19. It is necessary to carefully consider the system timing in this type of interface, to be sure that requirements for setup and hold times, and minimum pulse widths are met. Note also the drive limitations on long buses. Generally this type of interface is only favored if the memory peripheral address density is low so that simple address decoding can be used. Interrupt handling can also require many additional components, and using an interface device will usually simplify the system in this case.

NOTE: All typical values have been characterized but are not tested.

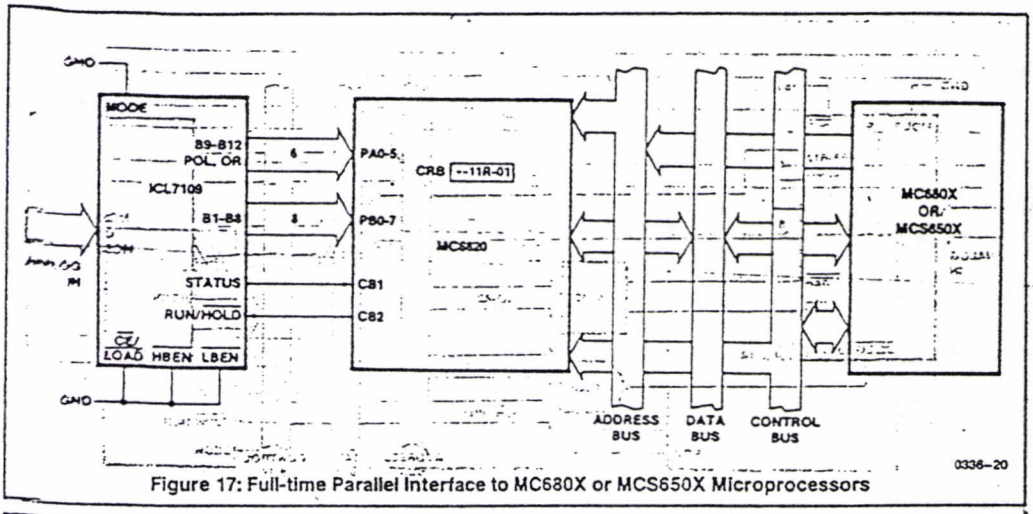


Figure 17: Full-time Parallel Interface to MC680X or MCS650X Microprocessors

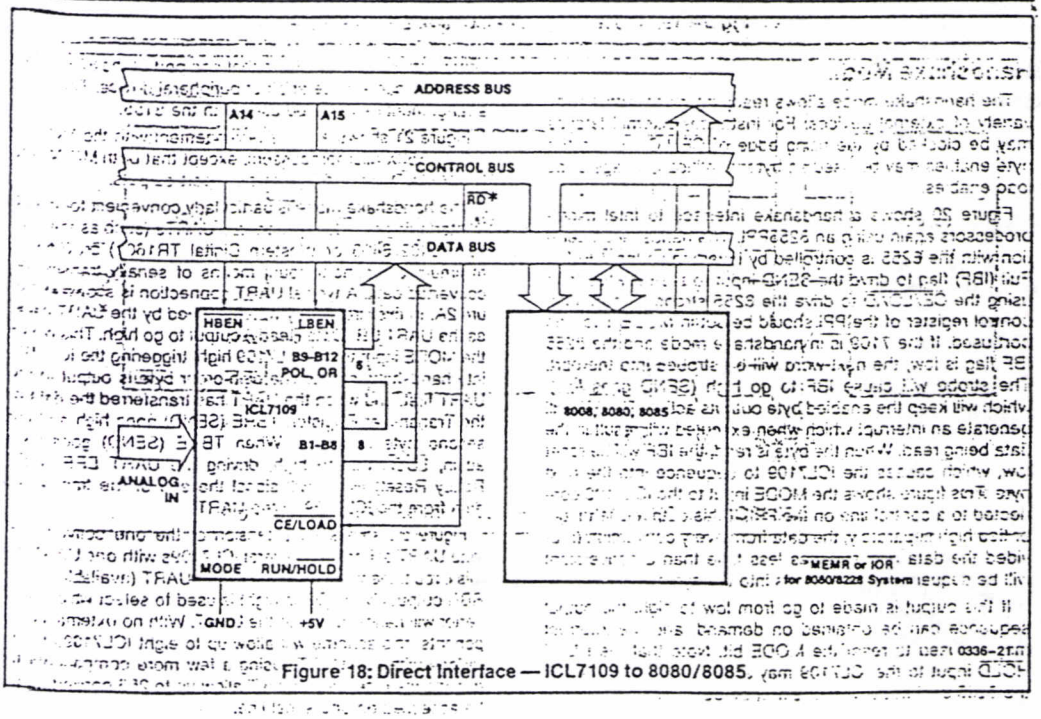


Figure 18: Direct Interface — ICL7109 to 8080/8085

3

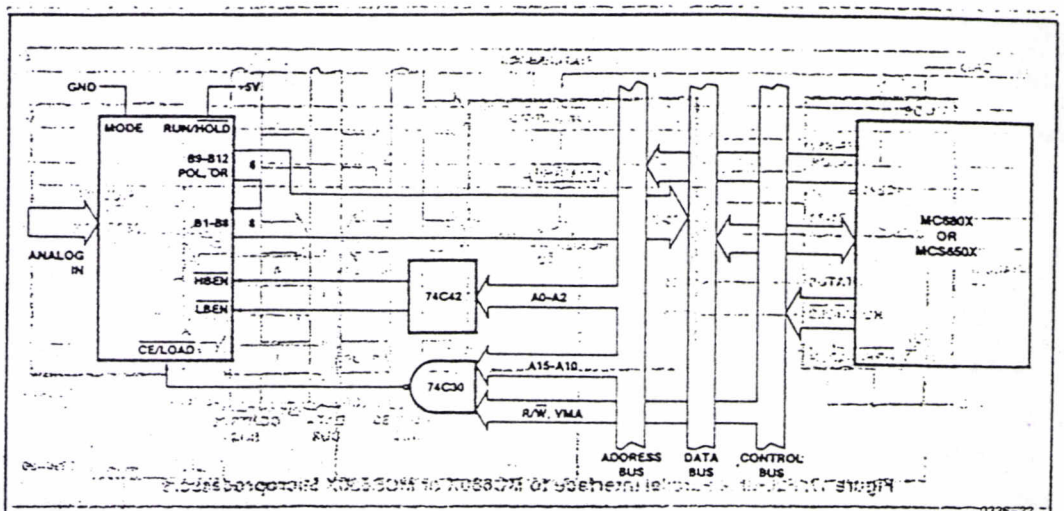


Figure 19: Direct ICL7109 — MC680X Bus Interface

Handshake Mode

The handshake mode allows ready interface with a wide variety of external devices. For instance, external latches may be clocked by the rising edge of CE/LOAD, and the byte enables may be used as byte identification flags or as load enables.

Figure 20 shows a handshake interface to Intel microprocessors again using an 8255PPI. The handshake operation with the 8255 is controlled by inverting its Input Buffer Full (IBF) flag to drive the SEND input to the ICL7109, and using the CE/LOAD to drive the 8255 strobe. The internal control register of the PPI should be set in MODE 1 for the port used. If the 7109 is in handshake mode and the 8255 IBF flag is low, the next word will be strobed into the port. The strobe will cause IBF to go high (SEND goes low), which will keep the enabled byte outputs active. The PPI will generate an interrupt which when executed will result in the data being read. When the byte is read, the IBF will be reset low, which causes the ICL7109 to sequence into the next byte. This figure shows the MODE input to the ICL7109 connected to a control line on the PPI. If this output is left high, or tied high separately, the data from every conversion (provided the data access takes less time than a conversion) will be sequenced in two bytes into the system.

If this output is made to go from low to high, the output sequence can be obtained on demand, and the interrupt may be used to reset the MODE bit. Note that the RUN/HOLD input to the ICL7109 may also be driven by a bit of the 8255 so that conversions may be obtained on command under software control. Note that one port of the 8255 is not used, and can service another peripheral device. The same arrangement can also be used with the 8155.

Figure 21 shows a similar arrangement with the MC6800 or MC6850X microprocessors, except that both MODE and RUN/HOLD are tied high to save port outputs.

The handshake mode is particularly convenient for directly interfacing to industry standard UARTs (such as the Harris IM6402/6403 or Western Digital TR1602) providing a minimum component count means of serially transmitting converted data. A typical UART connection is shown in Figure 2A. In this circuit, any word received by the UART causes the UART DR (Data Ready) output to go high. This drives the MODE input to the ICL7109 high, triggering the ICL7109 into handshake mode. The high order byte is output to the UART first, and when the UART has transferred the data to the Transmitter Register, TBRE (SEND) goes high and the second byte is output. When TBRE (SEND) goes high again, LBEN will go high, driving the UART DRR (Data Ready Reset) which will signal the end of the transfer of data from the ICL7109 to the UART.

Figure 22 shows an extension of the one converter — one UART scheme to several ICL7109s with one UART. In this circuit, the word received by the UART (available at the RBR outputs when DR is high) is used to select which converter will handshake with the UART. With no external components, this scheme will allow up to eight ICL7109s to interface with one UART. Using a few more components to decode the received word will allow up to 256 converters to be accessed on one serial line.

NOTE: All typical values have been characterized but are not tested.

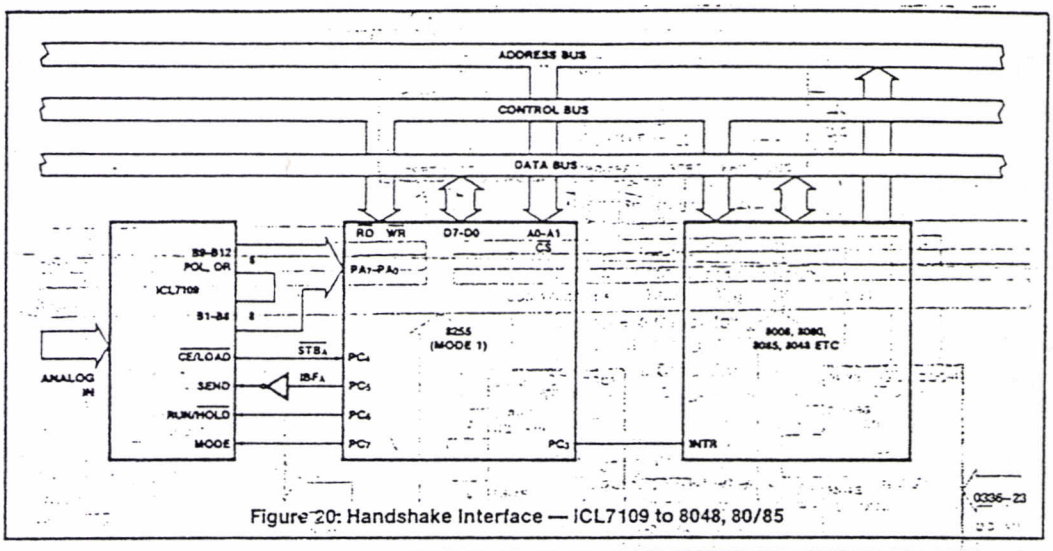


Figure 20: Handshake Interface — ICL7109 to 8048, 80/85

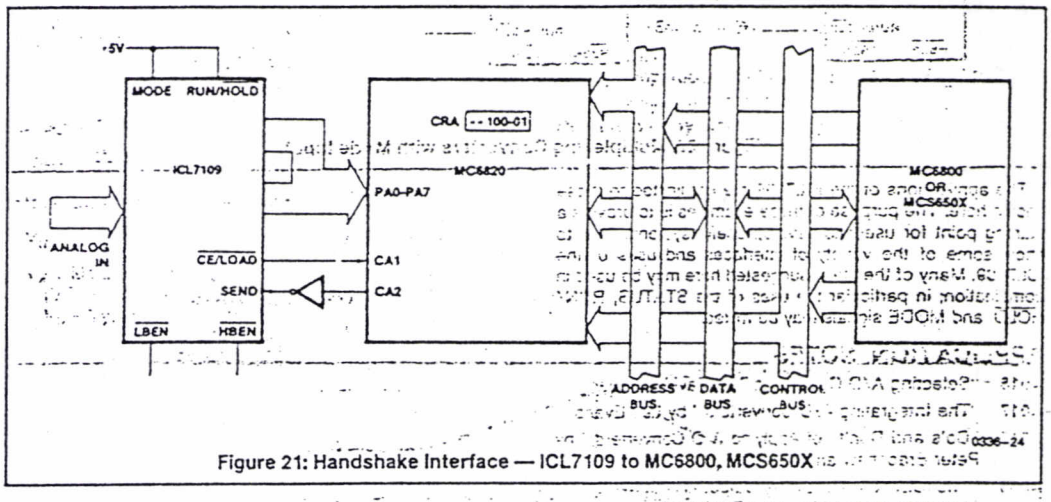


Figure 21: Handshake Interface — ICL7109 to MC6800, MCS650X

3

NOTE: All typical values have been characterized but are not tested.

ICL7109

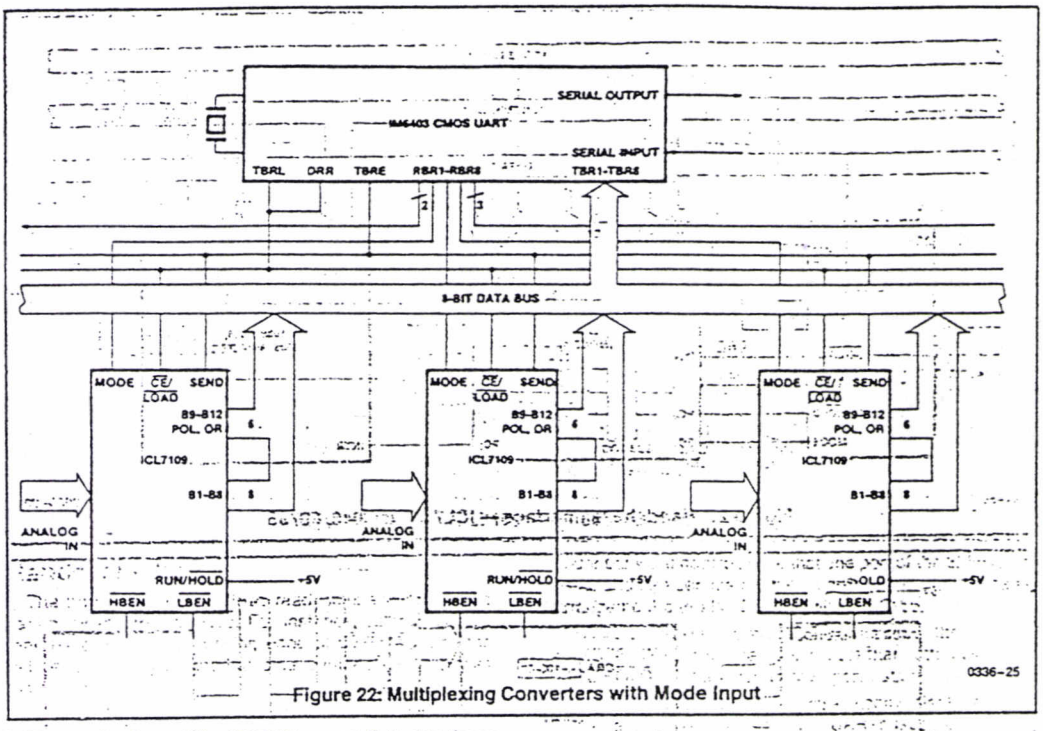


Figure 22: Multiplexing Converters with Mode Input

The applications of the ICL7109 are not limited to those shown here. The purpose of these examples is to provide a starting point for users to develop useful systems, and to show some of the variety of interfaces and uses of the ICL7109. Many of the ideas suggested here may be used in combination; in particular the uses of the STATUS, RUN/$\overline{\text{HOLD}}$, and MODE signals may be mixed.

APPLICATION NOTES

A016 "Selecting A/D Converters," by David Fullagar

A017 "The Integrating A/D Converters," by Lee Evans.

A018 "Do's and Don'ts of Applying A/D Converters," by Peter Bradshaw and Skip Osgood.

A030 "The ICL7104 — A Binary Output A/D Converter for Microprocessors," by Peter Bradshaw

A032 "Understanding the Auto-Zero and Common Mode Performance of the ICL7106 Family," by Peter Bradshaw

R005 "Interfacing Data Converters & Microprocessors," by Peter Bradshaw et al, Electronics, Dec. 9, 1976.

NOTE: All typical values have been characterized but are not tested.

ICL7109
INTEGRATING A/D CONVERTER

EQUATIONS

Oscillator Frequency
$f_{OSC} = 0.45/RC$;
$C_{OSC} > 50$ pF; $R_{OSC} \geq 50\ k\Omega$
f_{OSC} typ. = 60 kHz
or
f_{OSC} = 3.58 MHz Crystal

Oscillator Period
$t_{OSC} = RC/0.45$
$t_{OSC} = 1/3.58$ MHz (Crystal)

Integration Clock Frequency
$f_{CLOCK} = f_{OSC}$ (RC Mode)
$f_{CLOCK} = f_{OSC}/58$ (Crystal)
$t_{CLOCK} = 1/f_{CLOCK}$

Integration Period
$t_{INT} = 2048 \times t_{CLOCK}$

60/50 Hz Rejection Criterion
$t_{INT}/t_{50\ Hz}$ or $t_{INT}/t_{50\ Hz}$ = Integer

Optimum Integration Current
I_{INT} = 20.0 μA

Full Scale Analog Input Voltage
V_{INFS} Typically = 200 mV or 2.0V

Integrate Resistor

$$R_{INT} = \frac{V_{INFS}}{I_{INT}}$$

Integrate Capacitor

$$C_{INT} = \frac{(t_{INT})\ (I_{INT})}{V_{INT}}$$

Integrator Output Voltage Swing

$$\Delta V_{INT} = \frac{(t_{INT})\ (I_{INT})}{C_{INT}}$$

V_{INT} Maximum Swing:
$(V^- + 0.5V) < -V_{INT} < (V^+ - 0.5V)$
ΔV_{INT} Typically = 2.0V

Display Count

$$COUNT = 2048 \times \frac{V_{IN}}{V_{REF}}$$

Conversion Cycle
$t_{CYC} = t_{CLOCK} \times 8192$
(In Free-Run Mode, Run/HOLD = 1)
when f_{CLOCK} = 60 kHz, t_{CYC} = 133 ms

Common Mode Input Voltage
$(V^- + 1.0V) < V_{IN} < (V^+ - 0.5V)$

Auto Zero Capacitor
0.01 μF ≤ C_{AZ} ≤ 1.0 μF

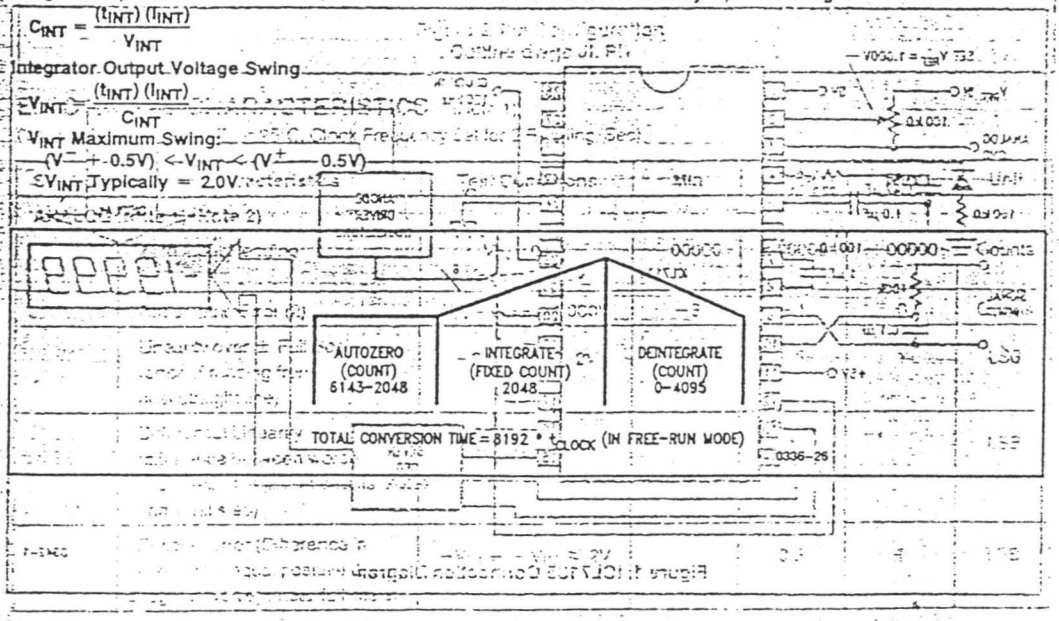
Reference Capacitor
0.1 μF ≤ C_{REF} ≤ 1.0 μF

V_{REF} Biased between V^+ and V^-
$V_{REF} = V^+ - 2.8V$
Regulation lost when V^+ to V_{REF} < 6.4V
If V_{REF} is not used, float output pin.

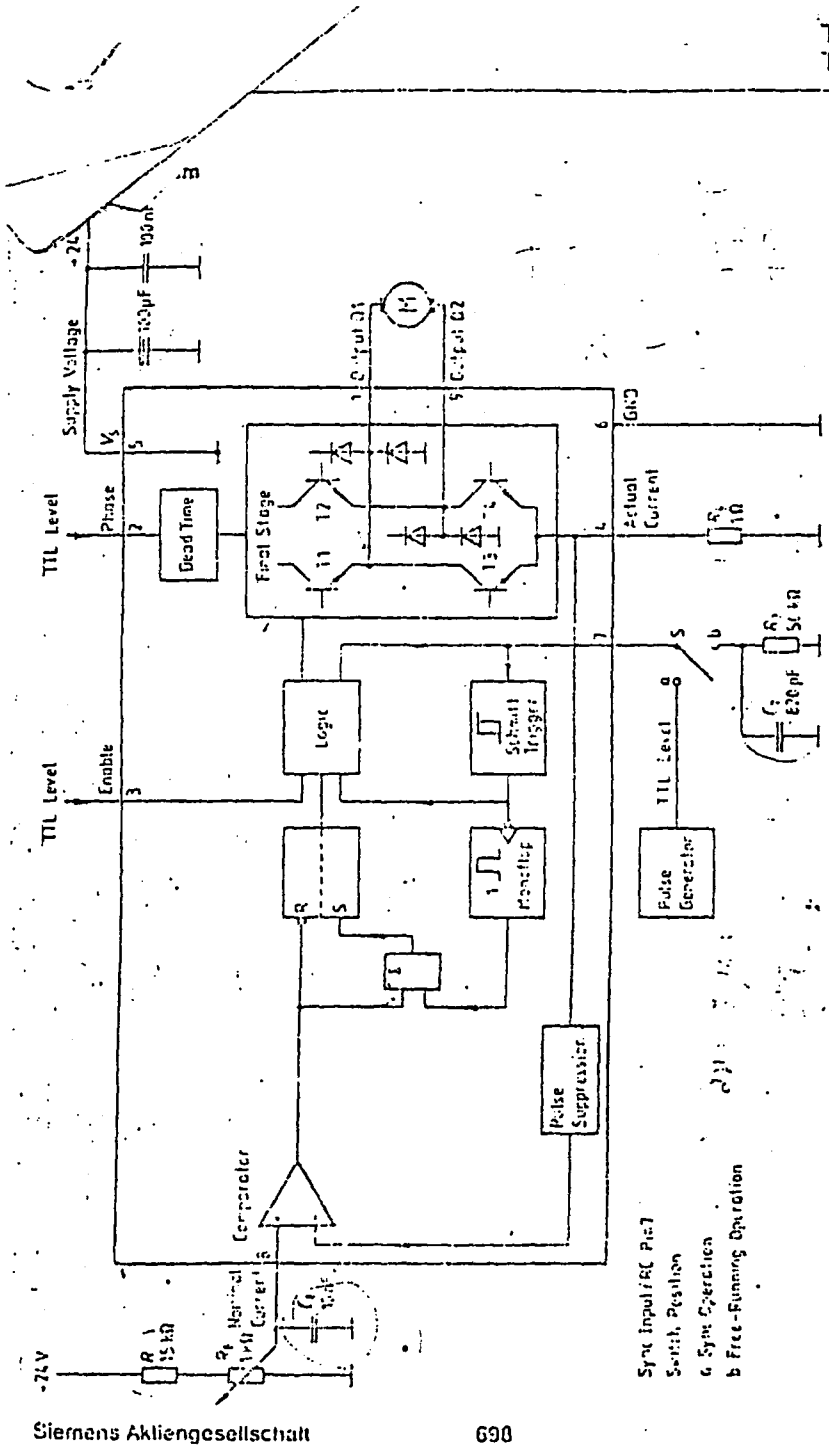
Power Supply: Dual ±5.0V
V^+ = +5.0 to GND
V^- = −5.0 to GND

Output Type:
Binary Amplitude with Polarity and Overrange Bits

Tips: Always tie TEST pin HIGH.
Don't leave any inputs floating.



NOTE: All typical values have been characterized but are not tested.


Supply Voltage
TIL Level
Phase
Dead Time
Final Stage
Output Q1
Output Q2
GND
Actual Current
Logic
Schmitt Trigger
Monoflop
Pulse Suppressor
Pulse Generator
Comparator
Enable
Sync Input/AC Port
Switch Position
a Sync Operation
b Free-Running Operation

Circuit Description

Outputs

Outputs Q1, Q2 (pins 1, 9) are fed by push-pull output stages. The two integrated free-wheel diodes, referred to ground or supply voltage respectively, protect the IC against flyback voltages from an inductive load.

Enable

Outputs Q1 and Q2 are turned off when voltage $V_{13} \leq 0.8$ V is applied to pin 3. The supply current then decreases maximally to 1 mA. The same occurs if pin 3 is open. The sink transistors are turned on when $V_{13} \geq 2$ V.

Phase

The voltage at pin 2 determines the phase position of the output current. Output Q1 acts as sink for $V_{12} \leq 0.8$ V and as source for $V_{12} \geq 2$ V.

Similarly output Q2 acts as

sink when $V_{12} \geq 2$ V and
source when $V_{12} \leq 0.8$ V

The sink transistors are current-chopped. An internal circuit avoids undesired cross-over currents at phase change.

Nominal Current Input

The peak current in the motor winding is determined by the voltage at pin 8. A comparator compares this with the voltage drop at the actual current sensor at pin 4. If the nominal current is exceeded, the output sink transistors are turned off by a logic circuit.

Sync Input/RC

Outputs are turned on by a signal at pin 7. Two operation modes are possible: Synchronizing by a fed-in TTL signal or free running with the external RC combination.

Free-Running Operation

When the supply voltage is applied, capacitor C_7 at pin 7 charges to a limiting voltage, typically 2.4 V. With increasing current in the motor winding, the voltage rises at the actual current sensor R_4 (pin 4). After exceeding the predetermined value at the nominal current input (pin 8) the comparator, in conjunction with pulse suppression, resets an RS flipflop. The logic turns off sink transistors T3 and T4. C_7 ceases charging and the parallel resistance R_7 then discharges C_7. The sink transistors remain turned off until the lower threshold voltage of the Schmitt trigger is reached. This off period is thus controlled by the time constant $\tau = R_7 \times C_7$. After the lower trigger threshold has been passed, the monoflop is triggered by the falling edge of the Schmitt trigger output and, provided the voltage at the actual current sensor (pin 4) is lower than the nominal value at pin 8, the RS flipflop is reset. The logic circuit then turns on the sink transistors T3 or T4 and recharges capacitor C_7. If the voltage at pin 4 rises above the comparator value at pin 8, the sink transistors T3 and T4 are turned off again. Turn-on cannot be repeated until capacitor C_7 has discharged to the lower trigger threshold, the discharge time being a function of R_7 and C_7.

...nous Operation

A TTL level sync signal is fed to pin 7, the negative edge sets the RS flipflop, via the Schmitt trigger/monoflop combination, provided that the voltage at pin 4 is below the nominal value at pin 8. As in the free-running operation mode, the relevant output transistors become conducting. Similarly they are cut off by resetting the RS flipflop once the voltage at pin 4 is higher than the nominal value at pin 8.

Pulse Suppression

In all cases the pulse suppression circuit eliminates positive pulses, typically of 0.5 µs duration, at pin 4. These can result from cross-over currents in chopper operation through the integrated free-wheel diodes. As a result, the voltage at pin 4 rises well above the nominal value, and without pulse suppression this would lead to dynamic current limiting. The duration of these basically unavoidable cross-over currents is of the same order of magnitude as the reverse-recovery time of the free-wheel diodes.

Temperature Safeguard

If the temperature of the IC rises to approx. 150 °C, the final stages are turned off. At approx. 130 °C they are turned on again.

Logic Table

Enable		L	L	H	H
Phase		L	H	L	H
Output	Q1	/	/	L	H
Output	Q2	/	/	H	L
Transistor	T1	X	X	X	·
Transistor	T2	X	X	·	X
Transistor	T3	X	X	··	X
Transistor	T4	X	X	X	··

at:
$V_t > 10$ mV
$R_t > 0\ \Omega$

L = Low voltage level, input open
H = High voltage level
X = Transistor turned off
· = Transistor conducting
·· = Transistor conducting with current limiting turned on
/ = Output high-impedance

Maximum Ratings
$T_C = -25\,°C$ to $+85\,°C$

Description	Symbol	min	max	Unit
Supply voltage, pin 5	V_S	−0.3	45	V
Supply current, pin 5	I_S	0	2.5	A
Peak current in output transistors, pin 1, 9	I_Q	−2.5	2.5	A

Diode currents

Diode to $+V_S$	I_{FH}		2.5	A
Diode to ground	I_{FL}		2.5	A
Input voltage, pins 2, 3, 7, 8	V_I	−0.3	6	V
Output current, pin 4	I_4	−2.5		A
Voltage, pin 4	V_4	−0.3	6	V
Ground current, pin 6	I_6		2.5	A
Junction temperature	T_j		150	°C
Storage temperature	T_{stg}	−40	125	°C
Thermal resistance system – ambient	$R_{th\,SA}$		70	K/W
system – case	$R_{th\,SC}$		8	K/W

Operating Range

Supply voltage, pin 5	V_S	8	40	V
Package temperature	T_C	−25	85	°C
Input voltage, pins 2, 3, 7	V_I		5	V
Output current	I_Q	−2	2	A

Characteristics

$V_S = 24$ V; $T_C = 25$ °C

Description	Symbol	Test conditions	min	typ	max	Unit
Supply current, pin 5	I_5	$V_{I3} = V_{IH}$		18	30	mA
Supply current, pin 5	I_5	$V_{I3} = V_{IL}$		0.5	1	mA
Output, pins 1, 9						
Output voltage: source	V_{OH}	$\|I_Q\| = 1$ A		1.7	1.9	V
Output voltage: source	V_{OH}	$\|I_Q\| = 1.5$ A		1.9	2.1	V
Output voltage: sink	V_{OL}	$\|I_Q\| = 1$ A		1.2	1.4	V
Output voltage: sink	V_{OL}	$\|I_Q\| = 1.5$ A		1.5	1.7	V
Reverse current	$\|I_{QS}\|$				300	µA
Phase dead time	t_T	Figure 1	0.1	0.3	1.0	µs
Forward voltage of diodes to $+V_S$	V_{FH}	$I_{FH} = 1$ A		1.0	1.2	V
	V_{FH}	$I_{FH} = 1.5$ A		1.1	1.3	V
Forward voltage of diodes to ground	V_{FL}	$I_{FL} = 1$ A		1.1	1.3	V
	V_{FL}	$I_{FL} = 1.5$ A		1.3	1.5	V
Inputs: enable, pin 3 and phase, pin 2						
H input voltage	V_{IH}		2			V
L input voltage	V_{IL}				0.8	V
H input current	I_{IH}	$V_{IH} = 5$ V		50	100	µA
L input current	$-I_{IL}$	$V_{IL} = 0$ V			100	µA
Rise and fall time	t_r, t_f				2	µs
Nominal current, pin 8						
Control range	V_8		0		2	V
Input current	$-I_{I8}$	$V_8 = 0$ V			5	µA
Input offset voltage	$V_{I(8-4)}$	Figure 5		0		mV
Actual current, pin 4						
Control range	V_4	Figure 5	0		2	V
Turn-off delay	t_d	Figure 3		2	3	µs
Sync input/RC, pin 7						
Sync frequency	f	Duty cycle, D	1		100	kHz
Duty cycle	D	$f = 40$ kHz	0.1		0.9	
Rise and fall time	t_r, t_f				2	µs
Output current, pin 7	$-I_{Q7}$		1.2	1.6	2.0	mA
Trigger threshold, pin 7	V_{T7}	Figure 2		0.6	0.8	V
Charging limit C_T	V_{C7}		2.2	2.4		V
Off period	t_{OFF}	Figure 4		0.4		µs
Dynamic input resistance pin 7	R_{I7}	$V_7 = 1.5$ V		1		kΩ

Maximum Ratings

T_c = 25 °C bis +85 °C

Description	Symbol	min	max	Unit
Supply voltage, pin 5	V_S	−0.3	45	V
Supply current, pin 5	I_S	0	1.25	A
Peak current in output transistors, pins 1, 9	I_Q	−1.25	1.25	A

Diode currents, pins 1,9

Diode against +V_S	I_{Fm}		1.25	A
Diode against ground	I_{Fl}		1.25	A
Input voltage, pins 2, 3, 7, 8	V_I	−0.3	6	V
Output current, pin 4	I_4	−1.25		A
Voltage, pin 4	V_4	−0.3	5	V
Ground current, pin 6	I_6		1.25	A
Junction temperature	T_j		150	°C
Storage temperature	T_{stg}	−40	125	°C
Thermal resistance system – ambient	$R_{th\,SA}$		70	K/W
system – case (measured at pin 14)	$R_{th\,SC}$		15	K/W

Operating Range

Supply voltage, pin 5	V_S	8	40	V
Package temperature measured at pin 14	T_C	−25	85	°C
Input voltage, pins 2, 3, 7	V_I		5	V
Output current, pins 1, 9	I_Q	−1	1	A

...racteristics
...s = 24 V; T_C = 25 °C

Description	Symbol	Test conditions	min	typ	max	Unit
Supply current, pin 5	I_C	$V_{i3} = V_{iH}$		18	30	mA
Supply current, pin 5	I_S	$V_{i3} = V_{iL}$		0.5	1	mA
Output, pins 1, 9						
Output voltage: source	V_{QH}	$\|I_Q\| = 0.5$ A		1.6	1.8	V
Output voltage: source	V_{QH}	$\|I_Q\| = 0.75$ A		1.65	1.90	V
Output voltage: sink	V_{QL}	$\|I_Q\| = 0.5$ A		1.0	1.2	V
Output voltage: sink	V_{QL}	$\|I_Q\| = 0.75$ A		1.1	1.4	V
Reverse current	$\|I_{QS}\|$				300	µA
Phase dead time	t_t	Figure 1	0.1	0.3	1.0	µs
Forward voltage of diodes to $+V_S$	V_{FH}	$I_{FH} = 0.5$ A		0.9	1.1	V
	V_{FH}	$I_{FH} = 0.75$ A		0.95	1.15	V
Forward voltage of diodes to ground	V_{FL}	$I_{FL} = 0.5$ A		0.95	1.15	V
	V_{FL}	$I_{FL} = 0.75$ A		1.0	1.2	V
Inputs: enable, pin 3 and phase, pin 2						
H input voltage	V_{iH}		2			V
L input voltage	V_{iL}				0.8	V
H input current	I_{iH}	$V_{iH} = 5$ V		50	100	µA
L input current	$-I_{iL}$	$V_{iL} = 0$ V			100	µA
Rise and fall time	t_r, t_f				2	µs
Nominal current, pin 8						
Control range	V_{i8}		0		2	V
Input current	$-I_{i8}$	$V_{i8} = 0$ V			5	µA
Input offset voltage	$V_{i(8-4)}$	Figure 5		0		mV
Actual current, pin 4						
Regulating range	V_{i4}	Figure 5	0		2	V
Turn-off delay	t_D	Figure 3		2	3	µs
Sync input/RC, pin 7						
Sync frequency	f	Duty cycle: 0.5	1		100	kHz
Duty cycle	D	f = 10 kHz	0.1		0.9	
Rise and fall time	t_r, t_f				2	µs
Output current, pin 7	$-I_{Q7}$		1.2	1.6	2.0	mA
Trigger threshold, pin 7	V_{L7}	Figure 2		0.6	0.8	V
Charging limit C_7	V_{C7}		2.2	2.4		V
Off period	t_{OFF}	Figure 4		64		µs
Dynamic input resistance pin 7	R_{i7}	$V_7 = 1.5$ V		1		kΩ

Internal Wiring of Pins

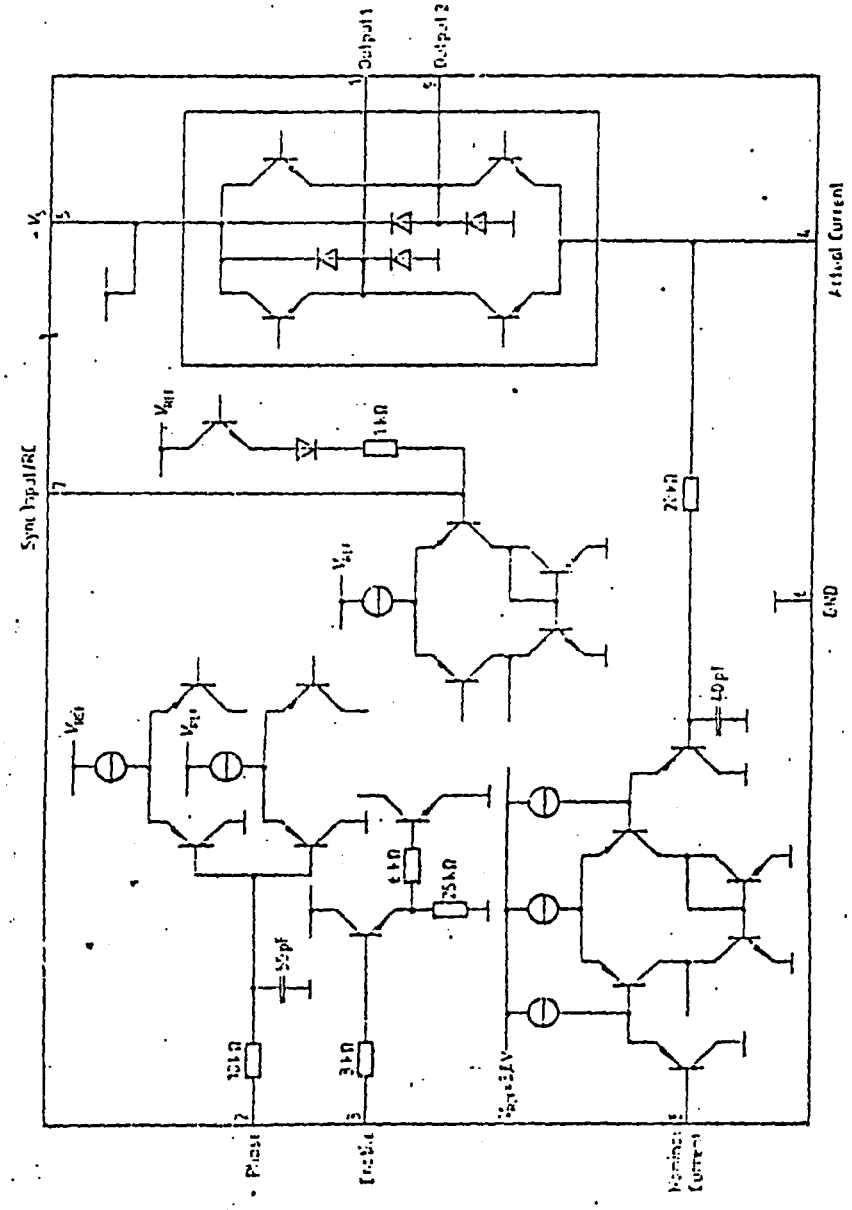


Figure 1
Phase Dead Time

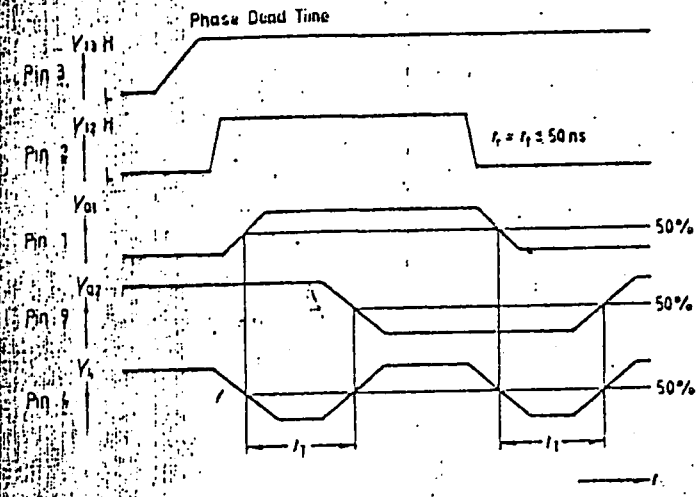


Figure 2
Trigger Threshold

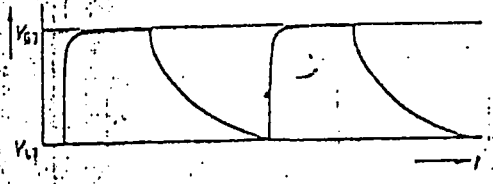


Figure 3
Turn-OFF Delay

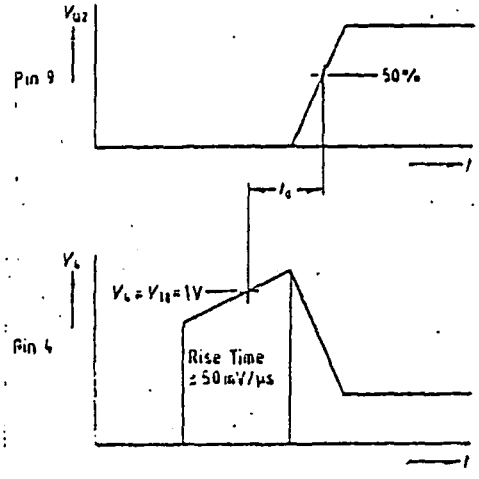


Figure 4
OFF Period versus Capacitance

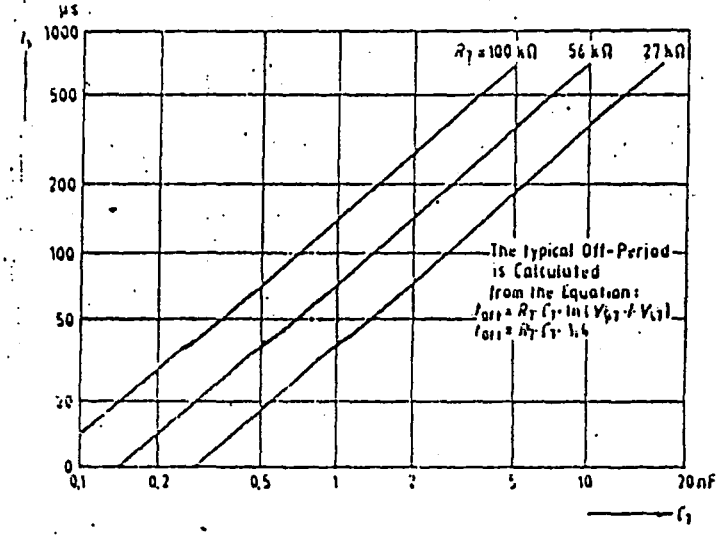
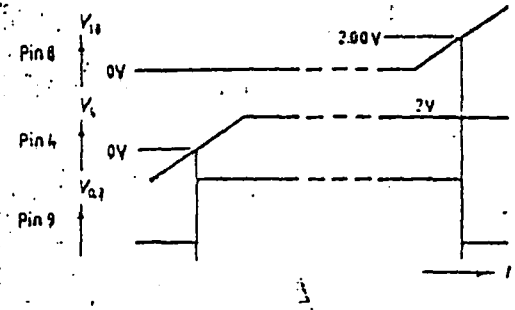
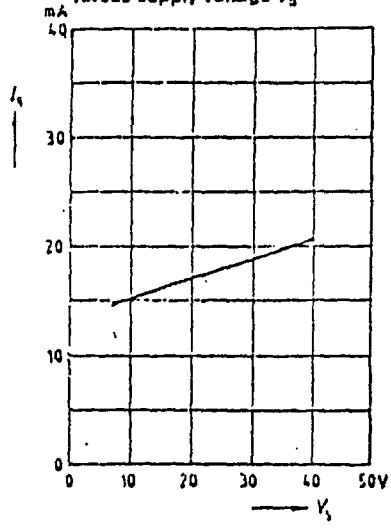


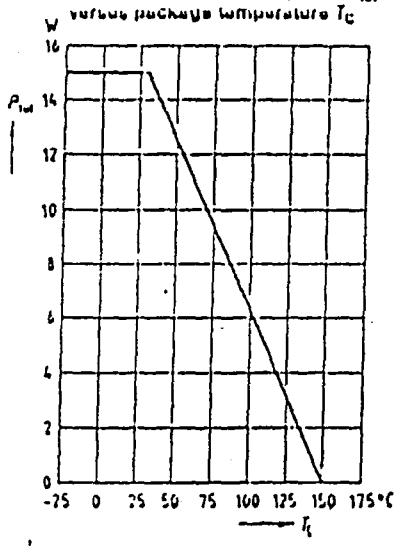
Figure 5
Control Range, Input Offset Voltage



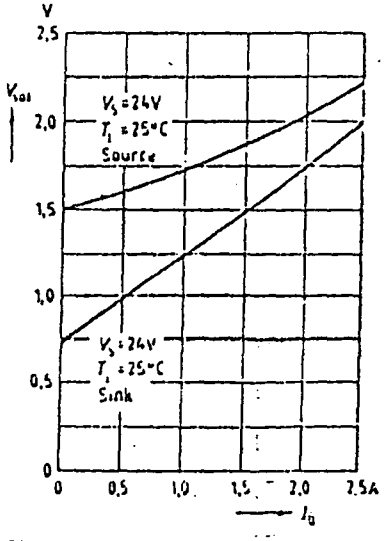
Quiescent current I_q
versus supply voltage V_s



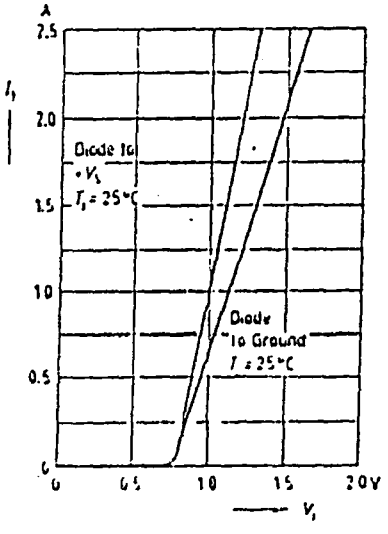
Permissible power dissipation P_{tot}
versus package temperature T_C



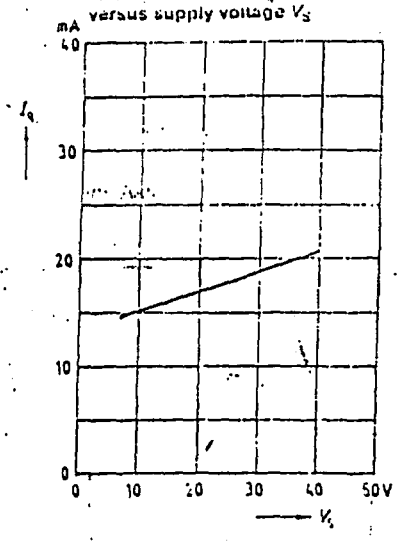
Output saturation voltages V_{sat}
versus output current I_Q



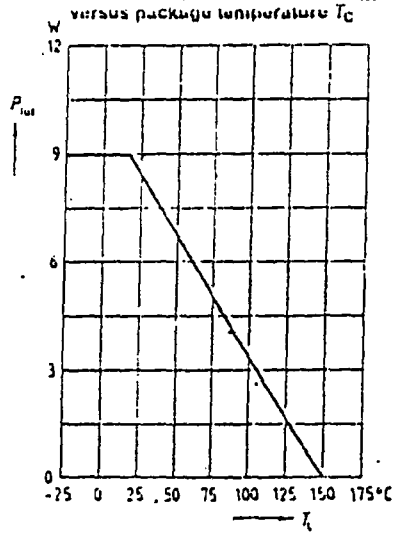
Forward current I_F of free-wheel
diodes versus forward voltages V_F



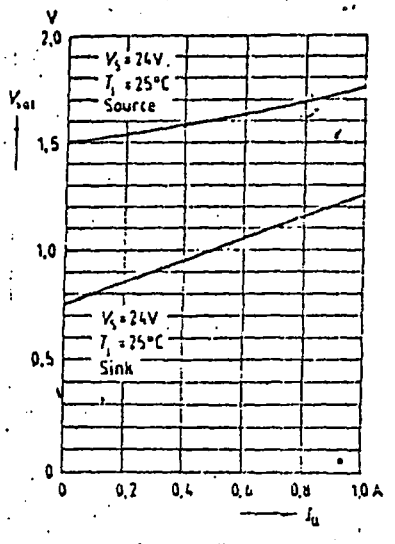
Quiescent current I_q versus supply voltage V_S



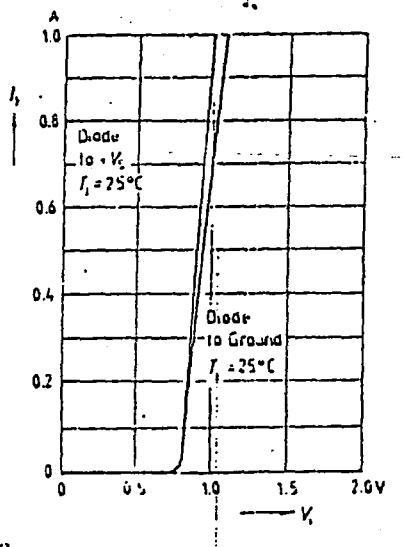
Permissible power dissipation P_{tot} versus package temperature T_C



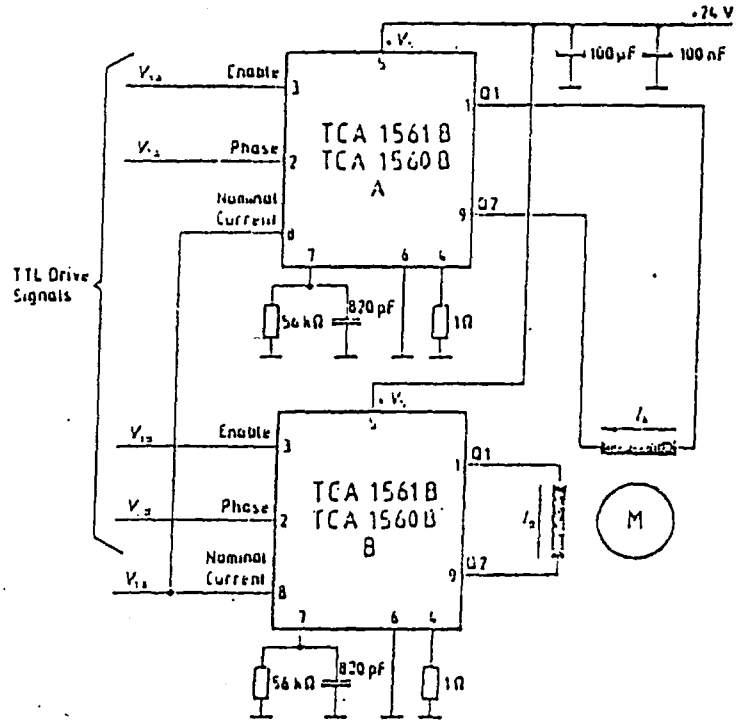
Output saturation voltages V_{sat} versus output current I_Q



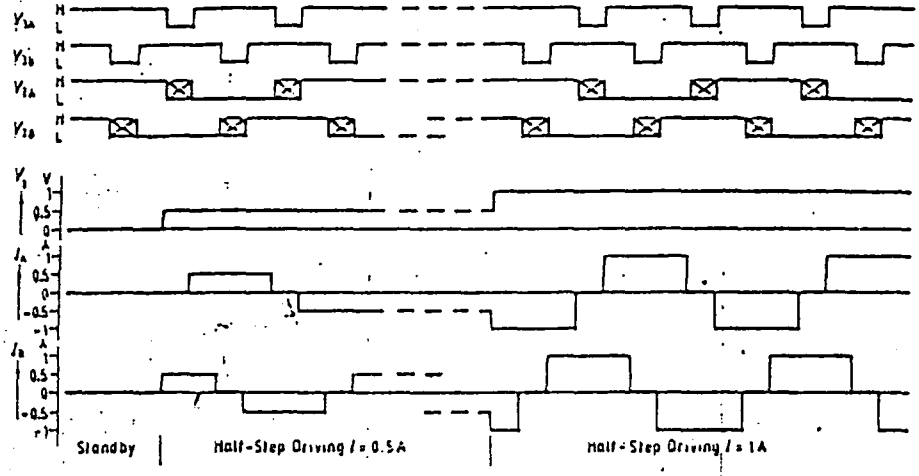
Forward current I_F of free-wheel diodes versus forward voltages V_F



Application Circuit



Pulse Diagram for Application Circuit



Calculation of Power Dissipation

The total power dissipation P_{tot} comprises

Saturation losses P_{sat}	(transistor saturation voltage and diode forward voltages)
Quiescent current losses P_q	(quiescent current multiplied by supply voltage)
Switching losses P_s	(turn-on/turn-off operation)

The following equations give the power dissipation for chopper operation without phase reversal. This can be regarded as "worst case", as, in addition to the switching losses, full-load current flows for the entire time.

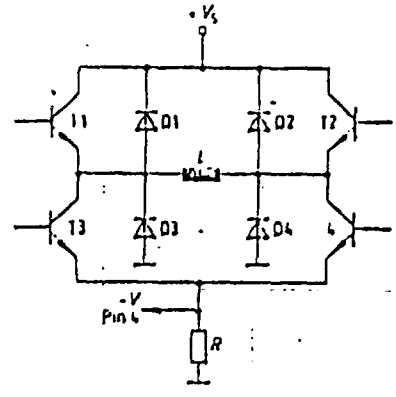
$$P_{tot} = P_{sat} + P_q + P_s$$

with $$P_{sat} \simeq I_n \left| V_{satu} \cdot D + V_{Fo} (1 - D) + V_{satu} \right|$$

$$P_q = I_q \cdot V_S$$

$$P_s \simeq \frac{V_S}{T} \left\{ \frac{i_o \cdot t_{o\,on}}{2} + \frac{(i_o + i_r)\, t_{on}}{4} + \frac{I_n}{2} \left(t_{o\,off} + t_{on} \right) \right\} f_w$$

I_n = Rated current (mean value)
I_q = Quiescent current
i_o = Reverse current during turn-on delay time
i_r = Peak reverse current
t_o = Conducting time of chop transistor
t_{on} = Turn-on time
t_{off} = Turn-off time
$t_{o\,on}$ = Turn-on delay time
$t_{o\,off}$ = Turn-off delay time
T = Cycle duration
D = Duty cycle t_o/T
V_{satu} = Saturation voltage of sink transistor (T3, 4)
V_{sato} = Saturation voltage of source transistor (T1, 2)
V_{Fo} = Forward voltage of free-wheel diode (D1, 2)
V_S = Supply voltage



เอกสารอ้างอิง

Takashi Kenjo, Stepping Motors and their Microprocessor Controls, 1990

Ramakant GayaKwad, Leonard Sokoloff, Analog and Digital Control System, Prentice-Hall International Inc,1988

Daniel H. Sheingold, Analog-Digital Conversion Handbook, Printic-Hall Englewood Cliffs, 1986.

Mustafa A. Mustafa, Digital and Analogue Interfacing for Microcomputer, Oxford Blackwell Scientific Publications, 1991.

โยธิน เปรมปราณีรัชต์, ระบบเซอร์โว และอิเลคทรอนิคส์คอนโทรลมอเตอร์, ตำราชุดวิศวกรรมศาสตร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง กรุงเทพฯ,2535.

ทินกร ดุ๊ก, การอินเตอร์เฟส IBM PC, พิสิกส์เซ็นเตอร์, 2532.

ยืน ภู่วรวรรณ, ทฤษฎีและการใช้งานไมโครโปรเซสเซอร์ Z 80, ซีเอ็ดยูเคชั่น, 2535.

บุญเลิศ เอี่ยมทัศนา ,เรียนภาษาปาสคาลด้วยเทอร์โบปาสคาล 4.0 -5.0, ซีเอ็ดยูเคชั่น , 2532.