ระบบเก็บและวิเคราะห์ข้อมูลความเร็วสูง

High speed sampling and analysis system

สุรเมธ สัจจอิสริยาวุฒิ

Mr. Suramet Sajjaisariyavut

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต

สาขาวิศวกรรมไฟฟ้า

บัณฑิตวิทยาลัย

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

พ.ศ. 2537

ISBN 974-621-170-6

HIGH SPEED SAMPLING AND ANALYSIS SYSTEM

SURAMET SAJJAISARIYAVUT

A THESIS SUBMITTED IN PARTIAL FULFILLMENT

OF THE REQUIREMENT FOR THE DEGREE

MASTER OF ENGINEERING IN ELECTRICAL ENGINEERING

GRADUATE SCHOOL

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

1994

ISBN 974-621-170-6

หัวข้อวิทยานิพนธ์	ระบบเก็บและวิเคราะห์ข้อมูลความเร็วสูง
นักศึกษา	นาย สุรเมธ สัจจอิสริยาวุฒิ
อาจารย์ผู้ควบคุมวิทยานิพนธ์	ผศ. พลผดุง ผดุงกุล
ระดับการศึกษา	วิศวกรรมศาสตรมหาบัณฑิต สาขาวิศวกรรมไฟฟ้า
ปีการศึกษา	2537

บทคัดย่อ

ปัจจุบันขั้นตอนการเก็บข้อมูลจากระบบในธรรมชาติต้องอาศัยอุปกรณ์ตัวแปลงสัญญาณซึ่งทำหน้าที่แปลงสัญญาณที่ต่อเนื่อง (analog) เป็นสัญญาณแบบไม่ต่อเนื่อง (digital) เพื่อนำข้อมูลเหล่านี้เข้าสู่ขบวนการวิเคราะห์ข้อมูลด้วยคอมพิวเตอร์ แต่ระบบเก็บและวิเคราะห์ข้อมูลที่ออกแบบขึ้นโดยทั่วไปในประเทศในปัจจุบันเป็นระบบที่มีความเร็วค่อนข้างต่ำ (น้อยกว่า 1 เมกกะเฮิร์ท) และถ้าต้องการใช้งานกับสัญญาณที่มีความเร็ว 5 เมกกะเฮิร์ทขึ้นไป เช่น สัญญาณภาพ ต้องอาศัยเครื่องมือและโปรแกรมวิเคราะห์จากต่างประเทศซึ่งมีราคาสูง วิทยานิพนธ์นี้เป็นการออกแบบสร้างเครื่องต้นแบบเพื่อเป็นพื้นฐานของการพัฒนาระบบเก็บและวิเคราะห์ข้อมูลความเร็วสูงขึ้นใช้เองภายในประเทศ ระบบทางฮาร์ดแวร์ประกอบด้วยส่วนแปลงข้อมูล จากสัญญาณอนาลอกขนาด 0-2.55 โวลต์เป็นสัญญาณดิจิตอล ขนาด 8 บิทด้วยอัตราเร็วในการสุ่มข้อมูล 20 ล้านครั้งต่อวินาที จากช่องสัญญาณอานาลอกที่กำหนดซึ่งมีทั้งสิ้น 8 ช่องสัญญาณ เก็บไว้ในหน่วยความจำความเร็วต่ำ (ความเร็วในการเข้าถึงข้อมูล 70 ns) ที่ได้รับการออกแบบให้ใช้กับการเก็บข้อมูลที่ความเร็วสูงขึ้นด้วยวิธีการ bank switching ขนาด 256 กิโลไบท์ (128 Kbyte x 2) ซึ่งสามารถขยายขนาดความจุหน่วยความจำได้สูงถึง 16 เมกกะไบท์ เมื่อข้อมูลได้รับการวิเคราะห์และ/หรือประมวลผลแล้ว ผู้ใช้สามารถแปลง ข้อมูลที่ได้ให้กลับไปอยู่ในรูปของสัญญาณอนาลอกเพื่อนำไปใช้งานต่อไป นอกจากนั้นระบบที่ออกแบบขึ้นนี้ยังรองรับการทำงานให้ลักษณะที่ต้องการเก็บและแสดงผลที่ได้ทันทีอีกด้วย โดยที่ประสิทธิภาพด้านความเร็วในการแสดงผลนั้นขึ้นอยู่กับประสิทธิภาพของระบบคอมพิวเตอร์ที่นำมาเชื่อมต่อ ในส่วนซอฟท์แวร์นั้นเพื่อให้สะดวกต่อการใช้งานและเรียนรู้จึงได้ทำการพัฒนาขึ้นบนระบบปฏิบัติการไมโครซอฟท์วินโดว์ส

Thesis Title	High speed sampling and analysis system
Student	Suramet Sajjaisariyavut
Thesis Advisor	Asst. Prof. Polphadung Phadungkul
Level of Study	Master of Engineering in Electrical Engineering
Department	Electronics
	King Mongkut's Institue of Technology Ladkrabang
Academic Year	1994

Abstract

Analog to Digital and Digital to Analog converter (ADC & DAC) are only one tool for join the real world quantities (Analog) with computer data (Digital) for analysis. This system is designed to be low cost but still remain the advantage of high speed system. The system's hardware is consist of converter for convert analog input signal (0-2.55 Volt) from one selected of eight input channel to be 8 bits digital signal and keep in 256 Kbytes (128Kb x 2), expanded to 16 Mbytes, low speed memory which is designed to operate in higher speed by using "bank switching" technique. For the real time data sampling works, this system is designed to support too and the system's real time performance is belong to the connected computer system. Morever, because of this system software is developed under Microsoft Windows operating system, so the user is easy to learn and used.

กิตติกรรมประกาศ

ขอกราบขอบพระคุณ

ท่านอาจารย์ ผศ. พลผดุง ผดุงกุล อาจารย์ที่ปรึกษา ผู้ให้การสนับสนุนทั้งความรู้ คำปรึกษา ความคิด กำลังใจ ชี้แนะแนวทาง ให้กับผู้เขียนมาโดยตลอดเวลาที่ศึกษา ค้นคว้า และการทำวิทยานิพนธ์ฉบับนี้จนสำเร็จการศึกษา

ขอกราบขอบพระคุณ คุณพ่อ คุณแม่ ที่กรุณาให้การสนับสนุนในการศึกษามาโดยตลอด

ขอขอบคุณ

บริษัท อิเลคทรอนิคส์ซอร์ส จำกัด ที่กรุณาเป็นสื่อกลางในการติดต่อขอตัวอย่างอุปกรณ์เพื่อใช้ในการทดลอง

บริษัท Harris Semiconductor USA. ที่กรุณาเอื้อเฟื้อสนับสนุนอุปกรณ์อนาลอกสวิทช์ความเร็วสูง

คุณ ชมธร ชัยจรูญพร ที่ให้ความสนับสนุนช่วยเหลือด้านอุปกรณ์ คำแนะนำ และหนังสือค้นคว้า

คุณ ศิระ ริกุลสุรกาน ที่ช่วยเหลือด้านงานเอกสาร และกำลังใจ

คุณ เจริญศักดิ์ มันตาดิลก ที่ช่วยเหลือด้านงานเอกสาร

คุณ ธนภัทร ฉายากุล ที่ช่วยเหลือด้านงานเอกสาร

และ พี่ๆ น้องๆ และเพื่อนทุกท่านที่ให้ความช่วยเหลือและกำลังใจมาโดยตลอดเวลาในการศึกษา โดยเฉพาะอย่างยิ่ง ในช่วงการทำวิทยานิพธ์ฉบับนี้

สารบัญ

ภาคผนวก

สารบัญรูป

บทที่ 1
บทนำ

1.1 กล่าวนำ

เมื่อทำการพิจารณาปริมาณที่ปรากฏในธรรมชาติจะพบว่าปริมาณเกือบทั้งหมดเป็นปริมาณที่มีการเปลี่ยนแปลงใน ลักษณะที่ต่อเนื่องกันเช่น อุณหภูมิของอากาศ ปริมาณของน้ำในถังน้ำที่อยู่ใต้ก๊อกน้ำที่เปิดอยู่ ปริมาณไฟฟ้าสถิตย์ภายในแห่งอำพันเมื่อถูกกับผ้าสักหลาดเป็นต้นคือ สัญญาณอนาลอก (Analog) และ สัญญาณดิจิตอล (Digital) ต่อมาเมื่อการคิดค้นคอมพิวเตอร์จึงเกิดคอมพิวเตอร์ขึ้นสองชนิดคือ อนาลอกคอมพิวเตอร์ และ ดิจิตอลคอมพิวเตอร์ ด้วยความง่ายและสะดวกในการออกแบบสร้างจึงทำให้ดิจิตอลคอมพิวเตอร์ได้รับการพัฒนาไปอย่างรวดเร็วจนในปัจจุบันกว่า 99 เปอร์เซ็นต์ของคอมพิวเตอร์ในโลกเป็นดิจิตอลคอมพิวเตอร์ทั้งสิ้นด้วยคุณสมบัติการทำงานภายในของดิจิตอลคอมพิวเตอร์ ซึ่งทำงานในลักษณะการเปิดและปิดของสัญญาณซึ่งเป็นแบบไม่ต่อเนื่อง ทำให้ไม่สามารถเชื่อมระบบในธรรมชาติซึ่งเป็นลักษณะอนาลอกเข้ากับคอมพิวเตอร์ในปัจจุบันได้โดยตรง ดังนั้นจึงต้องอาศัยอุปกรณ์ตัวกลางกลุ่มหนึ่งในการเชื่อมต่อเรียกอุปกรณ์นี้ว่าตัวแปลงสัญญาน(converter) ซึ่งมีอยู่สองชนิดคือ ตัวแปลงสัญญานจากอนาลอกเป็นดิจิตอล เรียกว่า analog to digital converter และ ตัวแปลงสัญญานจากดิจิตอลเป็น อนาลอก เรียกว่า digital to analog converter เมื่อได้ ข้อมูลแล้วจึงทำการเก็บและนำไปวิเคราะห์ต่อไป

ในปัจจุบันมีการตื่นตัวกันมาในงานการวิเคราะห์สัญญานต่างๆ เช่น การจดจำเสียงพูด การจดจำภาพ การแยกแยะวัตถุ การจดจำลายนิ้วมือ เป็นต้น ซึ่งงานเหล่านี้ต้องทำการแปลงข้อมูลที่ได้ทั้งหมดให้อยู่ในรูปของข้อมูลแบบดิจิตอลทั้งสิ้น บางครั้งจำเป็นจะต้องใช้อุปกรณ์ที่มีความเร็วในการเก็บข้อมูลสูง จึงจำเป็นต้องจัดหาซื้อระบบฮาร์ดแวร์และซอฟท์แวร์ในการวิเคราะห์มาจากต่างประเทศ ทำให้มีหน่วยงานไม่กี่แห่งสามารถทำงานวิจัยในการวิเคราะห์ข้อมูลที่ต้องอาศัยเครื่องมือที่มีความสามารถสูง แนวทางหลักของงานวิจัยนี้จึงเป็นการออกแบบต้นแบบระบบเก็บและวิเคราะห์ข้อมูลที่มีความเร็วสูง แต่จะต้องใช้อุปกรณ์ที่สามารถจัดหาได้ภายในประเทศมากที่สุดและราคาของระบบโดยรวมแล้วจะต้องไม่สูงมากนักเป็นหลักเพื่อให้เกิดความกระชับของเนื้อหาดังนั้นสำหรับในวิทยานิพนธ์นี้จะใช้คำว่า "ระบบเก็บข้อมูล" แทนระบบเก็บและวิเคราะห์ข้อมูลความเร็วสูง และ "หน่วยความจำ" แทน หน่วยความจำเก็บข้อมูลของระบบเก็บข้อมูล

1.2 วัตถุประสงค์

ในวิทยานิพนธ์นี้เป็นการนำเสนอการสร้างและออกแบบต้นแบบระบบเก็บข้อมูลความเร็วสูงขึ้นใช้เองภายในประเทศ และ เพื่อเป็นการเพิ่มทักษะและพื้นฐานความรู้ความเข้าใจในการออกแบบระบบดิจิตอลที่ความเร็วประมาณ 20 เมกกะเฮิร์ทและระบบอนาลอก ซึ่งทำงานที่ความเร็วประมาณ 5 เมกกะเฮิร์ท เนื่องจากระบบซอฟท์แวร์ที่ใช้กับระบบเก็บและวิเคราะห์ข้อมูลความเร็วสูงได้ทำการพัฒนาขึ้นบนระบบปฏิบัติการไมโครซอฟท์วินโดว์สซึ่งเป็นระบบปฏิบัติการแบบกราฟฟิกทำให้ผู้ใช้สามารถใช้งานระบบได้ง่ายขึ้นและเป็นการสร้างทักษะและความชำนาญในการออกแบบ

โปรแกรมภายในระบบปฏิบัติการไมโครซอฟท์วินโดว์ส ซึ่งเป็นพื้นฐานของระบบปฏิบัติการ Windows NT ซึ่งคาดว่าจะเป็นที่นิยมใช้มากในอนาคต

1.3 สิ่งที่ได้รับการพัฒนาขึ้นสำหรับวิทยานิพนธ์

เนื่องจากระบบทำการสุ่มข้อมูลด้วยความเร็ว 20 เมกกะเฮิร์ท ทำให้หน่วยความจำที่มีในปัจจุบันซึ่งมีความเร็วในการเข้าถึงข้อมูล (Access time) 70 นาโนวินาที ทำงานไม่ทัน จึงได้ทำการพัฒนาเทคนิคการสลับชุดหน่วยความจำให้สามารถใช้หน่วยความจำความเร็วต่ำกับความเร็วในการอ่านเขียนหน่วยความจำที่ความเร็วสูงกว่า 2 เท่า ของความเร็วในอ่านเขียนหน่วยความจำความเร็วต่ำนั้น โดยเพิ่มอุปกรณ์ไม่มากนักและเมื่อต้องการเพิ่มความเร็วในการอ่านเขียนให้มากขึ้นก็สามารถเพิ่มความเร็วขึ้นไปได้อีก โดยการเปลี่ยนอุปกรณ์ให้มีคุณสมบัติทำงานที่ความเร็วสูงขึ้นเท่านั้น โดยไม่ต้องจัดผังเวลาสัญญานของระบบใหม่

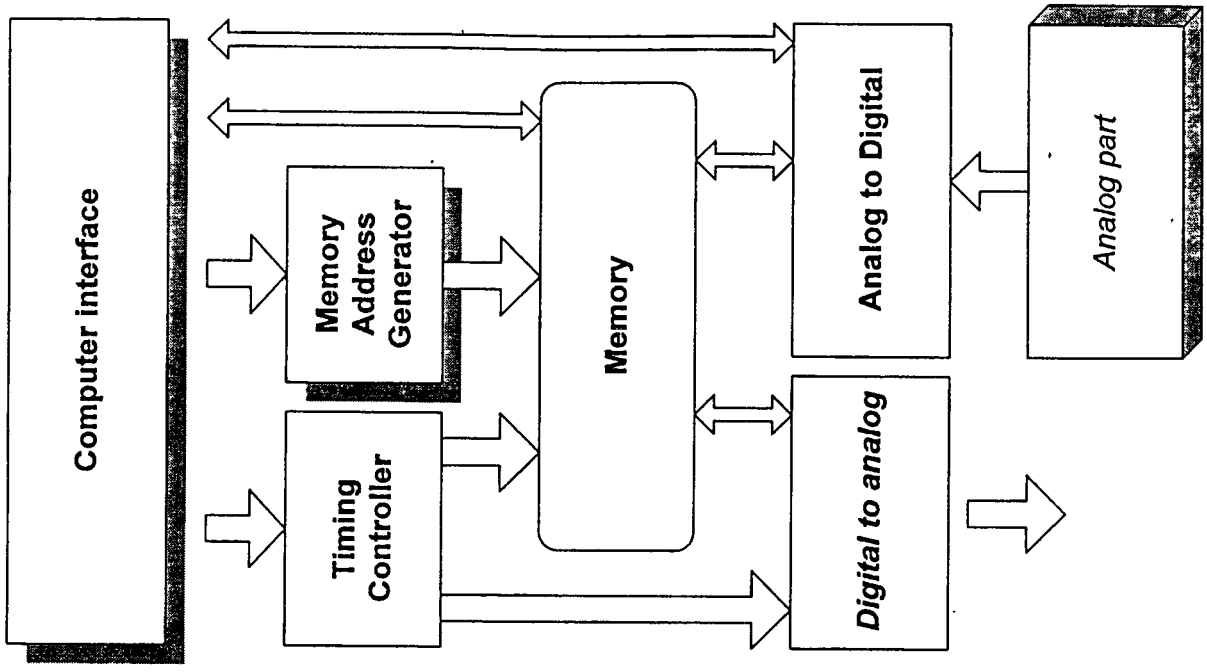
เทคนิคอีกอันหนึ่งที่ทำการพัฒนาขึ้น คือการสร้างสัญญาณที่มีขนาดแคบมาก เพื่อป้องกันปัญหาอันเนื่องมาจากการเขียนหน่วยความจำผิดพลาด ในขณะที่มีการเปลี่ยนแปลงสถานะ (Transition) ของสัญญาณตำแหน่งหน่วยความจำ

1.4 ขอบเขตประสิทธิภาพของระบบที่ออกแบบและพัฒนาขึ้น

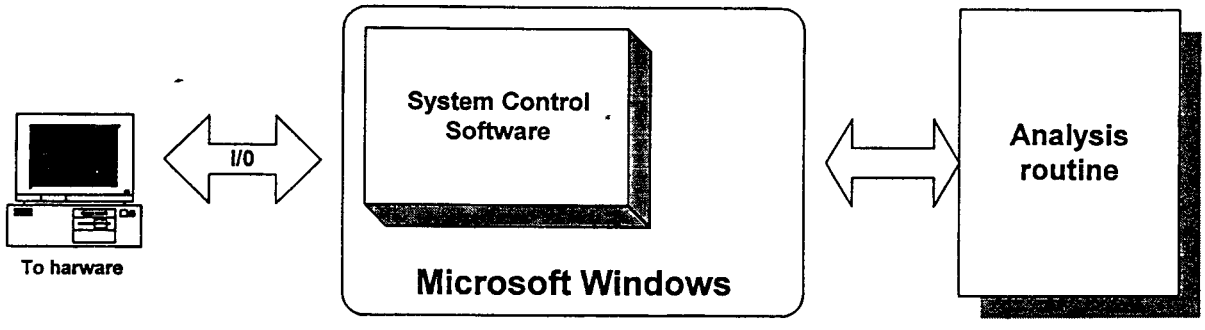
- มีช่องสัญญานเข้าได้ 8 ช่อง และเลือกได้โดยกำหนดจากซอฟท์แวร์
- รับสัญญานอนาลอกในรูปของความต่างศักย์โดยตรง ตั้งแต่ 0.00 - 2.55 โวลต์
- เลือกอัตราขยายของสัญญานที่เข้าได้ 3 ระดับ คือ 1, 10, 100 เท่า โดยทางซอฟท์แวร์
- มีหน่วยความจำเก็บข้อมูลเริ่มต้นไม่น้อยกว่า 256 กิโลไบท์ ขยายได้ถึง 16 เมกกะไบท์
- มีอัตราเร็วในการสุ่มข้อมูล 20 ล้านข้อมูลต่อวินาที
- มีอัตราเร็วในการเก็บข้อมูลลงหน่วยความจำได้ถึง 24 ค่า (ตั้งแต่ 2 Hz ถึง 20 MHz)
- สามารถกำหนดช่วงหน่วยความจำใน การเก็บ และ/หรือ วิเคราะห์ข้อมูล ได้
- สามารถแสดงผลที่ได้จากการเก็บข้อมูลในรูปกราฟที่ต้องการ

1.5 โครงสร้างและแผนภาพของระบบ

ระบบเก็บและวิเคราะห์ข้อมูลความเร็วสูงนี้แบ่งออกเป็น2ส่วนหลักคือ ส่วนฮาร์ดแวร์ และ ส่วนซอฟท์แวร์ ซึ่งจะกล่าวถึงโดยวิธีการและเทคนิคในการออกแบบโดยละเอียดในบทที่ 3 และบทที่ 4 ลักษณะของโครงสร้างของระบบทั้งสองส่วนจะเป็นดังรูป 1.1 และ 1.2



รูป 1.1 โครงสร้างของระบบฮาร์ดแวร์



รูป 1.2 โครงสร้างของระบบซอฟท์แวร์

1.6 รายละเอียดของวิทยานิพนธ์

ในวิทยานิพนธ์นี้ได้แบ่งออกเป็น 6 บท และภาคผนวก 2 ภาค โดยมีรายละเอียดของแต่ละบทโดยสังเขปดังต่อไปนี้

บทที่ 1 เป็นการกล่าวนำถึงวัตถุประสงค์และรายละเอียดของวิทยานิพนธ์

บทที่ 2 เป็นการกล่าวถึงการใช้งานอุปกรณ์ที่สำคัญบางชิ้น ซึ่งยังไม่มีการใช้เป็นที่แพร่หลาย

บทที่ 3 กล่าวถึงโครงสร้าง แนวทาง และเทคนิคในการออกแบบระบบ ส่วนฮาร์ดแวร์

บทที่ 4 กล่าวถึงโครงสร้าง การออกแบบ และวิธีการใช้งานระบบส่วนซอฟท์แวร์

บทที่ 5 เป็นผลการทดสอบระบบที่สร้างขึ้น

บทที่ 6 เป็นบทสรุป วิจารณ์ และแนวทางในการพัฒนาประสิทธิภาพของระบบ

ในส่วนท้ายของวิทยานิพนธ์นี้เป็นภาคผนวกซึ่งแต่ละภาคจะมีรายละเอียดดังนี้

ภาคผนวก ก วงจรทั้งหมด และวงจรแบ่งเป็นส่วนๆ ของส่วนฮาร์ดแวร์

ภาคผนวก ข แสดงรูปแผ่นวงจรพิมพ์ทั้งหมดที่ใช้ในการพัฒนาระบบ

ภาคผนวก ค โปรแกรมควบคุมระบบที่พัฒนาขึ้นบนไมโครซอฟท์วินโดวส์

ภาคผนวก ง โปรแกรมที่ใช้ทดลองระบบบน DOS.

บทที่ 2

คุณสมบัติ วิธีใช้งานและการออกแบบสำหรับอุปกรณ์ความเร็วสูงบางตัวที่ใช้ในวิทยานิพนธ์นี้

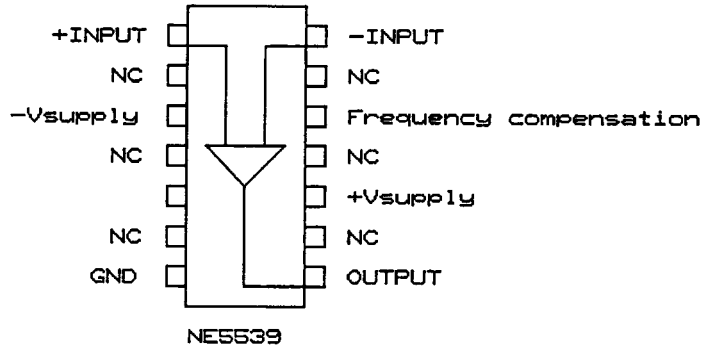
2.1 NE5539 Ultra high speed oprational amplifier

2.1.1 คุณสมบัติและหน้าที่ของขาสัญญาณ

NE5539 เป็นออปแอมป์ทำงานที่ความเร็วสูงได้รับการออกแบบเพื่อใช้ในวงจรขยายสัญญาณภาพ วงจรขยายสัญญาณ ย่านความถี่วิทยุ และวงจรที่ต้องการความเร็วในการเปลี่ยนแปลงสัญญาณต่อเวลาสูง ในการใช้งานไม่จำเป็นต้องมีการชดเชยจากภายนอกเมื่อกำหนดให้ออปแอมป์มีอัตราขยายมากกว่า 7 เท่า แต่ก็ยังคงทำงานได้ในกรณีที่อัตราขยายน้อยกว่า 7 เท่าโดยการเพิ่ม อุปกรณ์ทำการชดเชยจากภายนอก เนื่องจากทางอินพุทของ NE5539 เป็นวงจร emitter follower ทำให้อุปกรณ์ตัวนี้มีค่าอินพุทอิมพีแดนซ์แบบดิฟเฟอร์เรนท์เชี่ยลที่ค่อนข้างสูง คุณสมบัติโดย ทั่วไปของ NE5539 จะเป็นดังนี้

- ผลคูณของอัตราขยายกับความถี่ (Gain Bandwidth Product) เป็น 1.2 GHz ที่อัตราขยาย 17 dB
- Slew rate 600 โวลต์ต่อไมโครวินาที
- Full Power response 48 เมกกะเฮิทซ์
- อัตราขยายความต่างศักย์ที่ open loop เป็น 52 dB

ลักษณะตำแหน่งของขาสัญญาณเป็นดังรูป 2.1



รูป 2.1 ตำแหน่งขาสัญญาณของ NE5539

ขา 1 (+INPUT)

เป็นขาสัญญาณอินพุทด้านบวก

ขา 3 (-Vsupply)

เป็นขาความต่างศักย์ไฟเลี้ยงด้านลบ ปกติมีค่าไม่เกิน -12 โวลต์

ขา 7 (GND)

เป็นขากราวด์ของความต่างศักย์ไฟเลี้ยง

ขา 8 (OUTPUT)

เป็นเอาท์พุทของออปแอมป์

ขา 10 (+Vsupply)

เป็นขาความต่างศักย์ไฟเลี้ยงด้านบวก ปกติมีค่าไม่เกิน 12 โวลต์

ขา 12 (Frequency compensation)

เป็นขาสัญญาณที่ใช้ในการชดเชย เมื่ออุปกรณ์ทำงานที่อัตราขยายน้อยกว่า 7 เท่า

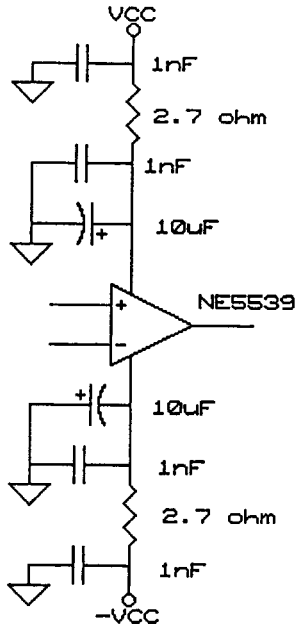
ขา 14 (-INPUT)

เป็นขาสัญญาณอินพุทด้านลบ

2.1.2 การใช้งาน NE5539

โครงสร้างภายในของ NE5539 ประกอบด้วยวงจรขยายความถี่สูงสองส่วน ส่วนแรกด้านอินพุทประกอบด้วยทรานซิสเตอร์ต่อเป็นแบบดาลิงตัน สองชุด ประกอบกันเป็นวงจรขยายแบบดิฟเฟอร์เรนเชียลที่มีอัตราขยาย 50 เท่า ในส่วนสองเป็นวงจรขยายชั้นเดียวที่มีอินพุทเฟสหนึ่งมาจากด้านหนึ่งของวงจรขยายแบบดิฟเฟอร์เรนเชียล ส่วนเฟสที่เหลือจะถูกรวมเข้ากับเอาท์พุทของส่วนนี้ ในส่วนของเอาท์พุทประกอบด้วยทรานซิสเตอร์ต่อเป็นวงจรขยายแบบดาลิงตัน ในการใช้งาน NE5539 นั้นเนื่องจากอุปกรณ์นี้ได้รับการออกแบบให้สามารถทำงานกับความถี่สูง ดังนั้นในการออกแบบวงจรและใช้งานต้องทำบนแผ่นวงจรพิมพ์แบบสองหน้าและในบริเวณพื้นที่ของแผ่นวงจรพิมพ์ที่ไม่มีการใช้งานจะต้องทำเป็นกราวด์เพลน การเชื่อมต่อระหว่างอุปกรณ์ภายในวงจรที่ออกแบบขึ้นต้องระมัดระวังเรื่องระยะของสายในการเชื่อมต่อระหว่างอุปกรณ์ภายในวงจร ถ้าจำเป็นต้องใช้ซอกเก็ตจะต้องใช้ซอกเก็ตแบบขากลมซึ่งมีค่าความจุแฝงที่รอยต่อระหว่างขาอุปกรณ์กับซอกเก็ตต่ำ นอกจากนั้นในส่วนของวงจรไฟเลี้ยงต้องต่อผ่านตัวต้านทานหรือตัวต้านทานที่มีขดลวดต่อคร่อมอยู่ค่า 2.7 โอหม์ โดยมีตัวเก็บประจุค่า 10 ไมโครฟารัดและ 1 นาโนฟารัดต่ออยู่ดังรูป 2.2 เพื่อลดสัญญาณรบกวนที่มากับระบบไฟเลี้ยง

ในการใช้งานทางด้านความถี่วิทยุควรใช้สายนำสัญญาณที่มีค่าอิมพีแดนซ์ต่ำเช่น 50 โอมห์ สำหรับงานด้านโทรคมนาคมทั่วไปและ 75 โอหม์ในกรณีที่ใช้กับงานระบบภาพ โดยปกติแล้ว NE5539 มีเอาท์พุทอิมพีแดนซ์ต่ำ แต่สำหรับการใช้งานบางประเภทที่ต้องมีอิมพีแดนซ์ของระบบเท่ากัน วิธีการง่ายที่สุดสำหรับ NE5539 คือ การเพิ่มตัวต้านทานที่มีค่าเท่ากับอิมพีแดนซ์ที่ต้องการต่ออนุกรมกับเอาท์พุทของ NE5539



รูป 2.2 การต่อวงจรเพิ่มเพื่อลดสัญญาณรบกวนที่มากับระบบไฟเลี้ยง

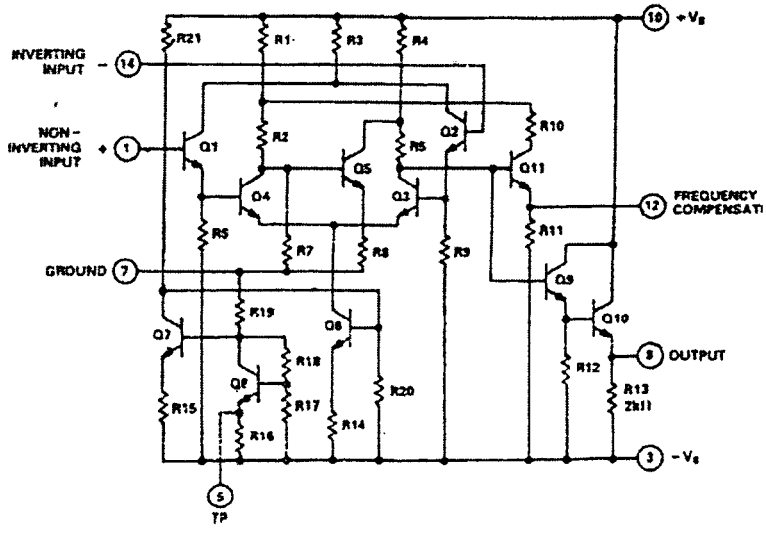


Figure 12. AD5539 Circuit

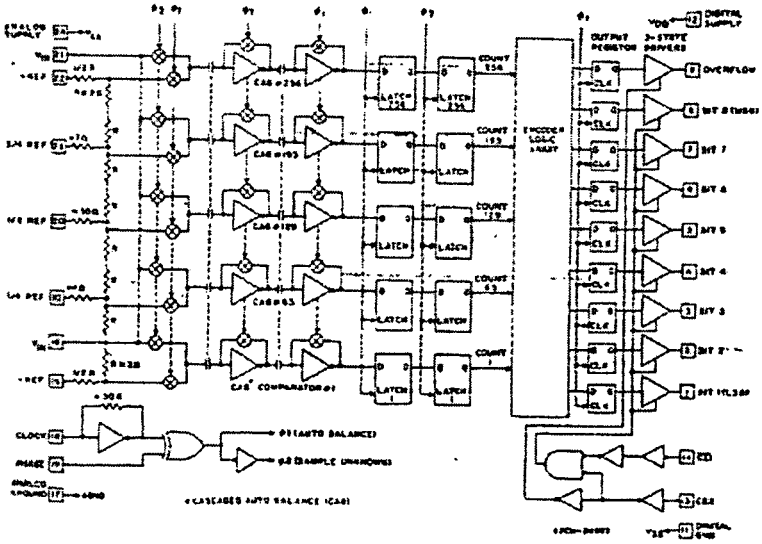
รูป 2.3 วงจรภายในของ NE5539

2.2 CA3318 High speed analog to digital convertor

2.2.1 คุณสมบัติและหน้าที่ของขาสัญญาณ

CA 3318 เป็นอุปกรณ์วงจรรวมที่ทำหน้าที่แปลงสัญญาณอนาลอกเป็นดิจิตอลแบบแฟลชขนาด 8 บิท ขนาด 24 ขา โครงสร้างภายในเป็นสวิชท์อิเลคทรอนิคส์ทำหน้าที่ผ่านสัญญาณที่ต้องการสุ่มเข้าสู่ตัวเปรียบเทียบจำนวน 256 ชุด ซึ่งทำหน้าที่เปรียบเทียบสัญญาณอินพุทที่เป็นอนาลอกกับแรงดันอ้างอิงของตัวเปรียบเทียบทั้ง 256 ชุด ซึ่งกำหนดจากภายนอก ผลการเปรียบเทียบที่ได้จะถูกส่งเข้าฟลิบฟลอบทำหน้าที่ค้างผลและเลื่อนผลที่ได้เมื่อได้รับสัญญาณนาฬิกาเข้าสู่วงจรเข้ารหัส เพื่อทำการเปลี่ยนผลให้อยู่ในรูปของเลขฐานสอง ขนาด 8 บิท ในกรณีที่ผลในการถอดรหัสมีค่าเกิน 255 จะให้เอาท์พุทออกทางบิทที่ 9 ค่าเลขฐานสองทั้ง 9 บิทนี้จะถูกค้างเอาไว้ด้วยฟลิบฟลอบอีกชุดหนึ่ง ซึ่งจะมีเอาท์พุทแบบสามสถานะ โดยจังหวะการทำงานภายในของ CA3318 นี้จะมีอยู่สองแบบซึ่งจะให้ผลการแปลงในครึ่งและหนึ่งสัญญาณนาฬิกา คุณสมบัติโดยทั่วไปของ CA3318 คือ

- เป็นตัวแปลงสัญญาณจากอนาลอกเป็นดิจิตอลแบบแฟลช ขนาด 8 บิท
- อัตราการแปลงสัญญาณ 20 ล้านครั้งต่อวินาที เมื่อให้ไฟเลี้ยง 5 โวลต์
- มีการแยกระบบกราวด์ของอนาลอกกับดิจิตอลอย่างเด็ดขาด



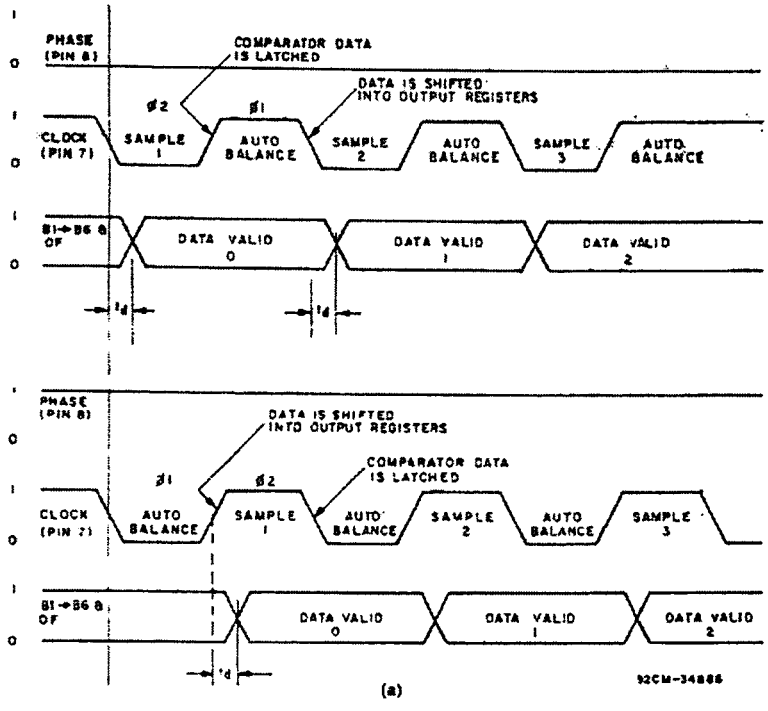
รูป 2.4 โครงสร้างภายใน CA3318

2.2.2 การกำหนดจังหวะการทำงานของ CA3318

ในการแปลงสัญญาณของ CA 3318 นั้นมีอยู่สองแบบคือแบบแปลงในครึ่งสัญญาณนาฬิกาและแปลงในหนึ่งสัญญาณนาฬิกา โดยการกำหนดลอจิกให้กับขา 19 (phase) เมื่อกำหนดให้ขาเฟสเป็นลอจิกต่ำ เอาท์พุท $\phi 1$ จะมี

ค่าเป็นตรงข้ามกับลอจิกของสัญญาณนาฬิกาที่เข้า ในขณะที่ $\phi 2$ จะให้ลอจิกตามสัญญาณนาฬิกา ผลของการเปรียบเทียบจากตัวเปรียบเทียบ จะค้างไว้ที่ขอบขาขึ้นของสัญญาณ $\phi 1$ และเมื่อ $\phi 1$ เป็นขอบขาลง จะเป็นการเลื่อนผลการถอดรหัสที่ได้ให้กับฟลิบฟลอบที่เอาท์พุท ซึ่งจะให้เอาท์พุทได้ภายในครึ่งสัญญาณนาฬิกา

เมื่อกำหนดให้ขาเฟสมีลอจิกเป็นสูง จะทำให้เอาท์พุท $\phi 1$ มีค่าลอจิกตามสัญญาณนาฬิกาจากภายนอก ในขณะที่ $\phi 2$ มีค่าลอจิกตรงข้ามกับสัญญาณนาฬิกาจากภายนอก เมื่อ $\phi 2$ เป็นขอบลงข้อมูลจากตัวเปรียบเทียบก็จะถูกค้างไว้จนกว่าจะ $\phi 2$ เป็นขอบขาขึ้นอีกครั้งในสัญญาณนาฬิกาลูกถัดมา จึงทำการเลื่อนข้อมูลให้กับฟลิบฟลอบที่เอาท์พุท ซึ่งจะเห็นได้ว่าการแปลงจะใช้เวลาหนึ่งสัญญาณนาฬิกา ดังนั้นในการใช้งานนั้นปกติแล้วมักจะกำหนดให้ CA 3318 ทำงานด้วยวิธีการแรก ซึ่งให้เอาท์พุทออกมาเร็วกว่าวิธีการสอง



รูป 2.5 ผังเวลาการทำงานของ CA3318 เมื่อกำหนดลอจิกที่ขาเฟสเป็นลอจิกต่ำและลอจิกสูง

ขาสัญญาณของ CA3318

ขา 1-8 (D0-D7)

ขาสัญญาณเอาท์พุทแบบดิจิตอลของการแปลงสัญญาณอนาลอก

ขา 9 (OVERFLOW)

เป็นขาสัญญาณที่บอกว่าขณะนี้สัญญาณอนาลอกที่แปลงเข้ามามีค่ามากกว่าความต่างศักย์อ้างอิง

ขา 21,16 (Vin)

เป็นขารับสัญญาณอนาลอกที่ต้องการแปลงเป็นดิจิตอล

ขา 22 (Vref), 23 (3/4 Vref), 20 (1/2 Vref) , 10 (1/4 Vref)

เป็นขาสัญญาณที่รับความต่างศักย์อ้างอิง

ขา 15 (-Vref)

เป็นขาสัญญาณรับสัญญาณด้านลบของสัญญาณอนาลอกที่ต้องการแปลงเป็นดิจิตอล

ขา 18 (CLOCK)

เป็นขาสัญญาณรับสัญญาณนาฬิกาที่ใช้กำหนดจังหวะการแปลงข้อมูลจากอนาลอกเป็นดิจิตอล

ขา 24 (AVcc)

เป็นขาความต่างศักย์ไฟเลี้ยงส่วนอนาลอกภายใน

ขา 17 (AGND)

เป็นขากราวด์ของความต่างศักย์ไฟเลี้ยงส่วนอนาลอกภายใน

ขา 12 (DVcc)

เป็นขาความต่างศักย์ไฟเลี้ยงส่วนดิจิตอลภายใน

ขา 11 (DGND)

เป็นขากราวด์ของความต่างศักย์ไฟเลี้ยงส่วนดิจิตอลภายใน

ขา 14 (-CE)

เป็นขาสัญญาณกำหนดให้ผลการแปลงข้อมูลจากอนาลอกเป็นดิจิตอลออกทางขาสัญญาน D0-D7

ขา 13 (CE2)

เป็นขาสัญญาณทำหน้าที่เหมือนกับ -CE แต่ต่างกันเฉพาะลอจิกควบคุมเป็นลอจิกสูงและทำหน้าที่ควบคุมเอาท์พุทของขาสัญญาณ OVERFLOW ด้วย

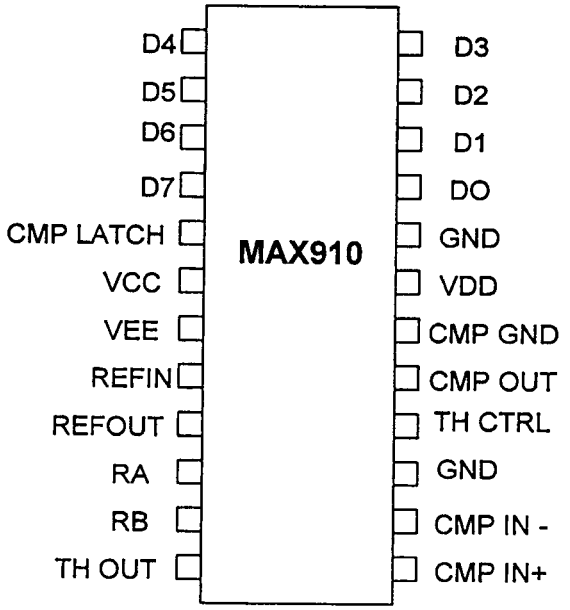
2.3 MAX910 High speed programable threshold voltage comparators

2.3.1 คุณสมบัติและหน้าที่ของขาสัญญาณ MAX 910

MAX910 เป็นอุปกรณ์เปรียบเทียบความต่างศักย์ความเร็วสูงตัวแรกที่มีวงจรแปลงสัญญาณจากดิจิตอลเป็นอนาลอกแบบ 8 บิทอยู่ภายในพร้อมกับมีวงจรกำเนิดระดับแรงดันอ้างอิงภายใน เพื่อให้สามารถกำหนดค่าความต่างศักย์อ้างอิงสำหรับการเปรียบเทียบค่า โดยขาสัญญาณทางด้านดิจิตอลเป็นระดับ TTL ทำให้สามารถเชื่อมต่อเข้ากับระบบไมโครโปรเซสเซอร์ได้โดยตรง วงจรเปรียบเทียบภายในทำงานได้เร็วมากสามารถให้เอาท์พุทของการเปรียบเทียบได้ภายในเวลาเพียง 8 นาโนวินาที ทำให้เหมาะสมกับการใช้งานด้านเครื่องมือสำหรับการทดสอบสัญญาณต่างๆ เนื่องจากมีขนาดเล็กวงจรที่จำเป็นทั้งหมดบรรจุไว้ภายในอุปกรณ์เพียงตัวเดียวทำให้ลดค่าของความจุแฝงอันเนื่องมาจากการออกแบบวงจรปกติ

ระดับความละเอียดของค่าความต่างศักย์ที่ใช้ในการเปรียบเทียบของตัวแปรสัญญาณจากดิจิตอลเป็นอนาลอก

กำหนดได้ 2 ค่าคือ 10 มิลลิโวลต์และ 20 มิลลิโวลต์ (หรือค่าความต่างศักย์เอาท์พุทสูงสุดของแต่ละแบบ คือ 2.56 และ 5.12 โวลต์) เมื่อใช้วงจรกำหนดภายใน วงจรภายในส่วนดิจิตอลและอนาลอกของ MAX 910 จะแยกแหล่งจ่ายไฟฟ้าออกจากกัน ซึ่งทั้งสองส่วนสามารถจ่ายระดับแรงดันเป็น +5,-5 หรือ +5,-5.2 โวลต์ก็ได้ ตำแหน่งและหน้าที่ของขาสัญญาณต่างๆจะเป็นดังรูป 2.6



รูป 2.6 ตำแหน่งและหน้าที่ของขาสัญญาณของ MAX 910

ขา 1-4,21-24 (D0-D7)

เป็นขาสัญญาณในระดับ TTL ที่รับข้อมูลที่ต้องการแปลงเป็นอนาลอก

ขา 5 (CMP LATCH)

เป็นขาสัญญาณที่เป็นสัญญาณอินพุทที่ใช้กำหนดให้ค้างผลลัพย์ของการเปรียบเทียบค่าโดยวงจรเปรียบเทียบภายในให้ค้างเอาไว้

ขา 6 (VCC)

เป็นขารับแรงดันไฟฟ้าที่จ่ายเลี้ยงให้กับระบบอนาลอกภายใน ในการใช้งานโดยทั่วไปจะต่อกับไฟเลี้ยงระบบอนาลอกทั้งหมด ปกติต่อกับระดับแรงดัน 5 โวลต์

ขา 7 (VEE)

เป็นขารับแรงดันไฟฟ้าด้านลบที่จ่ายเลี้ยงให้กับระบบอนาลอกภายใน โดยปกติจะต่อกับระดับแรงดัน 5 โวลต์

หรือ -5.2 โวลต์

ขา 8 (REFIN)

เป็นขาที่รับระดับแรงดันอ้างอิงในการให้เอาท์พุท อาจต่อกับขา REFOUT เพื่อใช้ระดับแรงดันอ้างอิงภายในหรือจะต่อกับระดับแรงดันอ้างอิงภายนอกก็ได้

ขา 9 (REFOUT)

เป็นขาที่ให้ระดับแรงดันอ้างอิง +2.56 โวลต์ ในกรณีถ้าต้องการให้แรงดันที่เอาท์พุทของวงจรแปลงข้อมูลจากอนาลอกเป็นดิจิตอลเป็น +2.56 โวลต์ให้ต่อขานี้เข้ากับขา REFIN

ขา 10 (RA)

ตัวต้านทานค่า 320 โอหม์ ต่อเข้ากับขาสัญญาณ TH OUT เพื่อกำหนดให้เอาท์พุทมีค่าสูงสุดที่ +2.55 โวลต์ และมีระดับความละเอียดเป็น 10 มิลลิโวลต์

ขา 11 (RB)

ตัวต้านทานค่า 640 โอหม์ ต่อเข้ากับขาสัญญาณ TH OUT เพื่อกำหนดให้เอาท์พุทมีค่าสูงสุดที่ +5.10 โวลต์ และมีระดับความละเอียดเป็น 20 มิลลิโวลต์

ขา 12 (TH OUT)

เป็นขาสัญญาณเอาท์พุทที่ได้จากการแปลงสัญญาณดิจิตอลเป็นอนาลอก ต้องต่อเข้ากับขาสัญญาณ RA หรือ RB เพื่อกำหนดเอาท์พุทสูงสุดและระดับความละเอียด

ขา 13 (CMP IN+)

เป็นขาสัญญาณด้านบวกของสัญญาณที่ต้องการนำมาเปรียบเทียบ

ขา 14 (CMP IN-)

เป็นขาสัญญาณด้านลบของสัญญาณที่ต้องการนำมาเปรียบเทียบ

ขา 15,20 (GND)

เป็นขากราวด์ของระดับแรงดันไฟเลี้ยงส่วนอนาลอก ซึ่งแยกจากขากราวด์ของระบบดิจิตอล

ขา 16 (TH CTRL)

เป็นขาสัญญาณที่ใช้ในการปรับอินพุทที่เข้าสู่ส่วนแปลงข้อมูลจากดิจิตอลเป็นอนาลอก ปกติจะต่อผ่านค่าความต้านทานแบบปรับค่าละเอียดขนาด 10 กิโลโอหม์ โดยต่อระหว่างตัวต้านทาน 500 กิโลโอหม์จำนวนสองตัว ซึ่งต่ออยู่กับแรงดันไฟเลี้ยงส่วนอนาลอกด้านบวกและด้านลบ เพื่อใช้ในการปรับค่าของระดับความต่างศักย์ที่ระดับต่ำ

ขา 17 (CMP OUT)

ขาสัญญาณเอาท์พุทของผลการเปรียบเทียบค่าของส่วนเปรียบเทียบค่าความต่างศักย์ให้ระดับแรงดันเป็นแบบ TTL

ขา 18 (CMP GND)

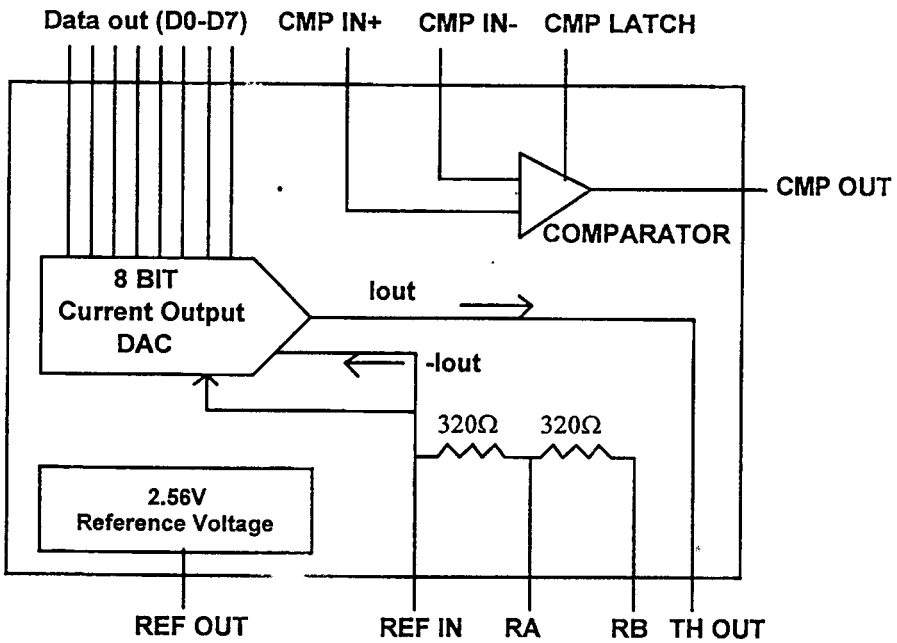
เป็นขาระดับแรงดันอ้างอิงของส่วนเปรียบเทียบค่าความต่างศักย์

ขา 19 (VDD)

ระดับแรงดันไฟฟ้าสำหรับเลี้ยงส่วนดิจิตอล ปกติต่อเข้ากับแรงดันไฟฟ้า 5 โวลต์ของส่วนดิจิตอลของระบบที่ใช้งาน

2.3.2 การใช้งาน MAX910

ในการใช้งาน MAX 910 สิ่งที่ต้องคำนึงถึงคือ ส่วนแปลงข้อมูลจากดิจิตอลเป็นอนาลอกของอุปกรณ์นี้ให้ กระแสชิงค์ จากเอาท์พุทแบบคอมพรีเมนทารีที่อินพุทดิจิตอลมีค่าสูงสุดได้เพียง 8 มิลลิแอมป์ ซึ่งในการใช้งานจะต้องนำไปแปลงเป็นระดับความต่างศักย์ 2.56 หรือ 5.12 โวลต์ โดยการต่อขาเอาท์พุท TH OUT เข้ากับขา สัญญาณ RA หรือ RB ค่าดิจิตอลที่อินพุทจะเป็นค่าที่ใช้ในการแบ่งกระแสเอาท์พุทระหว่าง Iout และ -Iout ของเอาท์พุท ส่วนแปลงสัญญาณจากดิจิตอลเป็นอนาลอก เมื่อค่าดิจิตอลที่อินพุทมีค่าเป็น 0 ทั้งหมด TH OUT จะชิงค์กระแสค่าสูงสุดจาก Iout และ -Iout จะไม่มีการชิงค์กระแสและกระแสจะกลับกันเมื่อค่าดิจิตอลที่อินพุทมีค่าเป็น 1 ทั้งหมด ดังนั้นเมื่อค่าดิจิตอลที่อินพุทมีค่าอื่นก็จะเป็นการเฉลี่ยของกระแส Iout และ -Iout ลักษณะของโครงสร้างภายในเป็นดังรูป 2.7



รูป 2.7 โครงสร้างภายในของ MAX 910

ในการใช้งาน MAX 910 จะต้องต่อขาสัญญาณ TH OUT เข้ากับขาสัญญาณ RA หรือ RB เพื่อกำหนดค่า

ความต่างศักย์สูงสุดโดยค่าตัวต้านทานภายในที่ขาสัญญาณ RA และ RB จะทำหน้าที่สร้างค่าความต่างศักย์จากกระแสที่ได้จากเอาท์พุท TH OUT ค่าของความต่างศักย์ที่ได้จะเป็นไปดังตาราง 2.1 เมื่อต่อ TH OUT เข้ากับจุด RA หรือ RB โดยใช้แรงดันอ้างอิงภายใน (REF OUT) หรือใช้แรงดันอ้างอิงจากภายนอก (VRefExt)

แรงดันอ้างอิง	ความต้านทาน	Vth(max)	Vth(min)	ความละเอียด
REFOUT	RA (320)	2.56	0.01	10 mV
REFOUT	RB (640)	+2.56	-2.54	20 mV
VRefExt	RA (320)	VRefExt	(VRef_ext)/256	(VRef_ext)/256
VRefExt	RB (640)	VRefExt	-(VRef_ext)x(254/256)	2x(VRef_ext)/256

ตาราง 2.1 ค่าความต่างศักย์ที่ได้เมื่อระดับแรงดันอ้างอิงและตัวต้านทานที่ TH OUT ต่างกัน

ในกรณีที่ต้องการใช้งานที่มีความแม่นยำสูงขึ้นจำต้องต่อระดับแรงดันอ้างอิงจากภายนอก จะต้องต่อเข้ากับ REFIN ซึ่งค่าของความต่างศักย์เอาท์พุท โดยขึ้นอยู่กับการเลือกตัวต้านทานจากขาสัญญาณ RA หรือ RB ค่าความต่างศักย์อ้างอิง และค่าของความต่างศักย์ที่ขาสัญญาณ TH CTRL
ซึ่งคำนวนได้จากสมการ

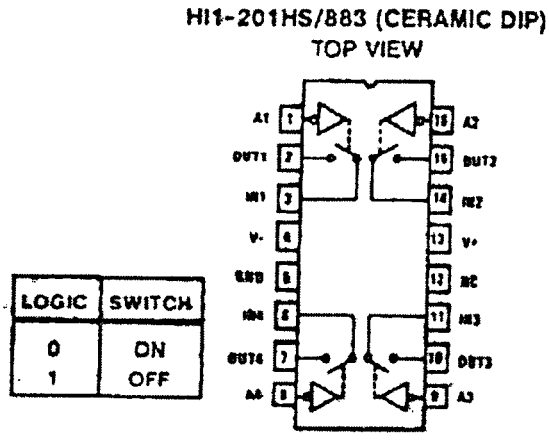
$$VTH = VRefIn-[(255/256)x(RSPAN/320) x (VRefIn-VTH CTRL)]$$

เมื่อ R SPAN มีค่าเป็น 320 โอหม์เมื่อต่อเข้ากับ RA หรือ 640 โอหม์เมื่อต่อเข้ากับ RB

2.4 HI-201HS High Speed Quard SPST CMOS Analog Switch

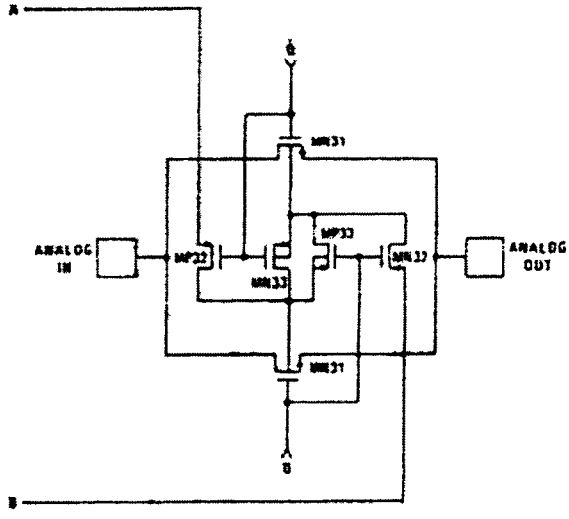
2.4.1 คุณสมบัติบางประการ การใช้งาน และขาสัญญาณ

HI-201HS เป็นอนาลอกสวิทช์แบบ CMOS ที่สามารถทำการเปิดปิดด้วยความเร็วสูงโดยมีค่าความต้านทานเมื่อสวิทช์ปิดต่ำ ภายในบรรจุสวิทช์แบบ SPST แยกเป็นอิสระต่อกันจำนวน 4 ชุด โดยมีความเร็วในการสวิทช์เพียง 50 นาโนวินาทีและมีค่าความต้านทานเมื่อสวิทช์ปิด 50 โอห์ม สามารถใช้งานกับสัญญาณอนาลอกช่วง +-15 โวลต์ ทนกระแสสูงสุดอย่างต่อเนื่องไม่เกิน 25 มิลลิแอมป์ สัญญาณที่ใช้ควบคุมสามารถใช้ได้ทั้งระดับ CMOS และระดับ TTL ตำแหน่งของขาสัญญาณเป็นดังรูป 2.8



รูป 2.8 ตำแหน่งขาสัญญาณของ HI-201HS

โครงสร้างภายในของ HI-201HS ประกอบด้วยวงจร Level shifter driver ทำหน้าที่รับสัญญาณควบคุม ซึ่งเป็นระดับ CMOS หรือ TTL มาทำการควบคุมการผ่านของสัญญาณอนาลอกของ switch cell ซึ่งมีโครงสร้างประกอบขึ้นจาก MOS จำนวน 6 ตัว ดังรูป 2.9



รูป 2.9 โครงสร้างของ switch cell ภายใน HI-201HS

ขาสัญญาณ

ขา 3,6,11,14 (In1, In4, In3, In2)

เป็นขาสัญญาณอนาลอกอินพุท

ขา 2,7,10,15 (Out1, Out4, Out3, Out2)

เป็นขาสัญญาณอนาลอกเอาท์พุท

ขา 1,6,11,14 (A1, A4, A3, A2)

เป็นขาสัญญาณควบคุมการเปิด/ปิดของสวิทช์แต่ละชุด เมื่อต้องการให้ผ่านสัญญาณให้ต่อเข้ากับลอจิกต่ำ และให้ลอจิกสูงเมื่อต้องการให้เปิดวงจร สัญญาณควบคุมที่ขานี้อาจเป็นสัญญาณระดับ TTL หรือ CMOS ก็ได้

ขา 4 (V-)

เป็นขาแรงดันไฟเลี้ยงด้านลบ ต้องมีค่าไม่เกิน -18 โวลต์

ขา 13 (V+)

เป็นขาแรงดันไฟเลี้ยงด้านบวก ต้องมีค่าไม่เกิน +18 โวลต์

ขา 5 (GND)

เป็นขากราวด์ของไฟเลี้ยง

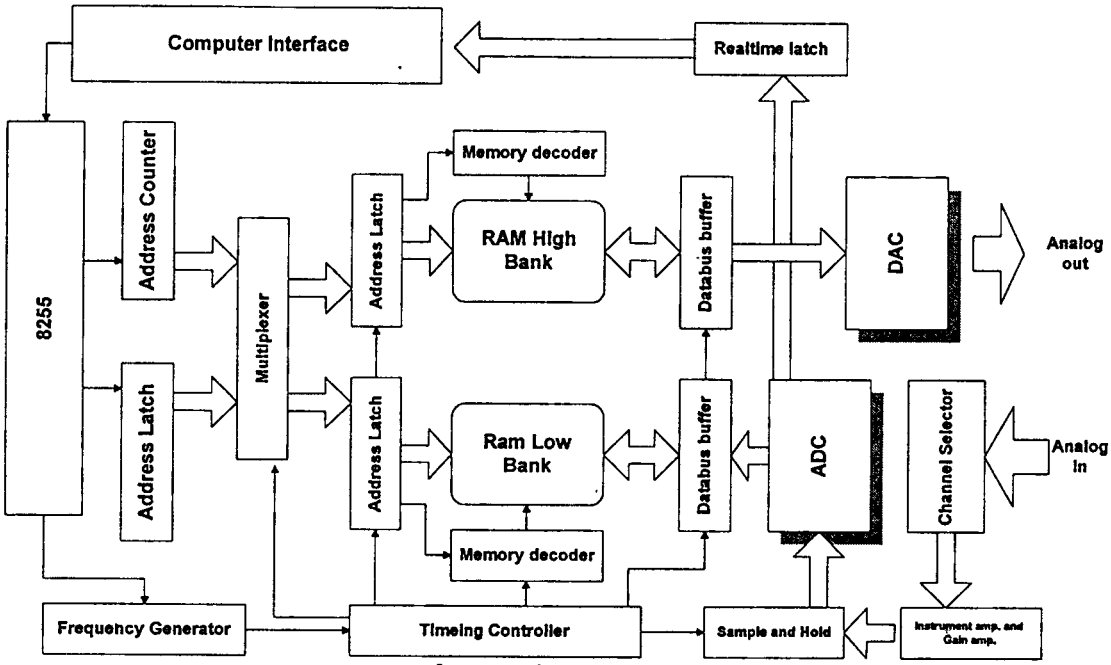
บทที่ 3
การออกแบบระบบส่วนฮาร์ดแวร์

3.1 โครงสร้างและแผนภาพของระบบ ส่วนฮาร์ดแวร์

ในส่วนของฮาร์ดแวร์จะแบ่งออกได้เป็น 2 ส่วนคือ ส่วนอนาลอก ส่วนดิจิตอล ในส่วนอนาลอกจะประกอบด้วยวงจรเลือกช่องสัญญาณเข้าทำการเลือกช่องสัญญาณเข้า 8 ช่องสัญญาณ โดยใช้อนาลอกสวิทช์เป็นอุปกรณ์ส่งผ่านสัญญาณที่ต้องการเข้าวงจรขยายสัญญาณแบบอินสทรูเมนต์ ที่มีอัตราขยายเป็นหนึ่งเพื่อให้ระบบมีอินพุทอิมพิแดนซ์สูงและเปลี่ยนระดับแรงดันอ้างอิงของสัญญาณเข้า เป็นระดับแรงดันอ้างอิงของระบบ จากนั้นเข้าวงจรขยายสัญญาณแบบกำหนดการขยายได้ เพื่อใช้ในกรณีที่สัญญาณเข้ามานั้นต่ำแล้วผ่านเข้าวงจรค้างสัญญาณเพื่อคงค้างระดับสัญญาณเข้าไว้ จนกว่าวงจรแปลงสัญญาณจากอนาลอกเป็นดิจิตอลทำการแปลงข้อมูลเรียบร้อย และในส่วนของวงจรแปลงสัญญาณจากดิจิตอลเป็นอนาลอก ทำหน้าที่ส่งข้อมูลดิจิตอลที่ทำการแปลงแล้วให้กลับไปอยู่ในรูปของสัญญาณอนาลอกเพื่อใช้งานต่อไป

ส่วนดิจิตอลการทำงานจะเริ่มต้นจากวงจรสร้างสัญญาณนาฬิกาเพื่อเป็นฐานเวลาในการทำงานของระบบ ทั้งหมด ซึ่งผู้ใช้สามารถเลือกค่าความเร็วของสัญญาณนาฬิกาได้ 24 ค่า โดยมีช่วงตั้งแต่ 0 - 20 เมกกะเฮิร์ท สัญญาณนาฬิกาที่ความเร็วสูงสุดของระบบจะถูกส่งเข้าจ่ายเพื่อกำหนดจังหวะการสุ่มข้อมูลของวงจรแปลงสัญญาณจากดิจิตอลเป็นอนาลอก เพื่อแก้ปัญหาของการจัดผังเวลาของส่วนจัดการหน่วยความจำที่อัตราการแปลงข้อมูลต่างกัน จึงได้ออกแบบให้วงจรส่วนแปลงสัญญาณจากอนาลอกเป็นดิจิตอลทำการสุ่ม ข้อมูลด้วยความเร็วสูงสุดของระบบ แล้วใช้วิธีการเปลี่ยนแปลงความเร็วของสัญญาณนาฬิกาที่ใช้ควบคุมจังหวะการเขียนหน่วยความจำ เพื่อกำหนดอัตราการบันทึกข้อมูลลงหน่วยความจำ โดยตำแหน่งที่นับจะถูกส่งต่อให้รับวงจรเลือกชุดตำแหน่งหน่วยความจำ ซึ่งทำหน้าที่เลือกกลุ่มตำแหน่งหน่วยความจำที่จะติดต่อเก็บข้อมูลว่าจะมาจากระบบคอมพิวเตอร์ หรือมาจากวงจรกำเนิดตำแหน่งหน่วยความจำของระบบเก็บข้อมูล ค่าของตำแหน่งหน่วยความจำที่เลือกแล้วนี้จะถูกส่งผ่านให้กับวงจรค้างตำแหน่งหน่วยความจำ เมื่อต้องการอ่านค่าจากหน่วยความจำสามารถ ทำการอ่านได้โดยตรงจากพอร์ทข้อมูล

แผนภาพของโครงสร้างระบบฮาร์ดแวร์จะเป็นดังรูป 3.1



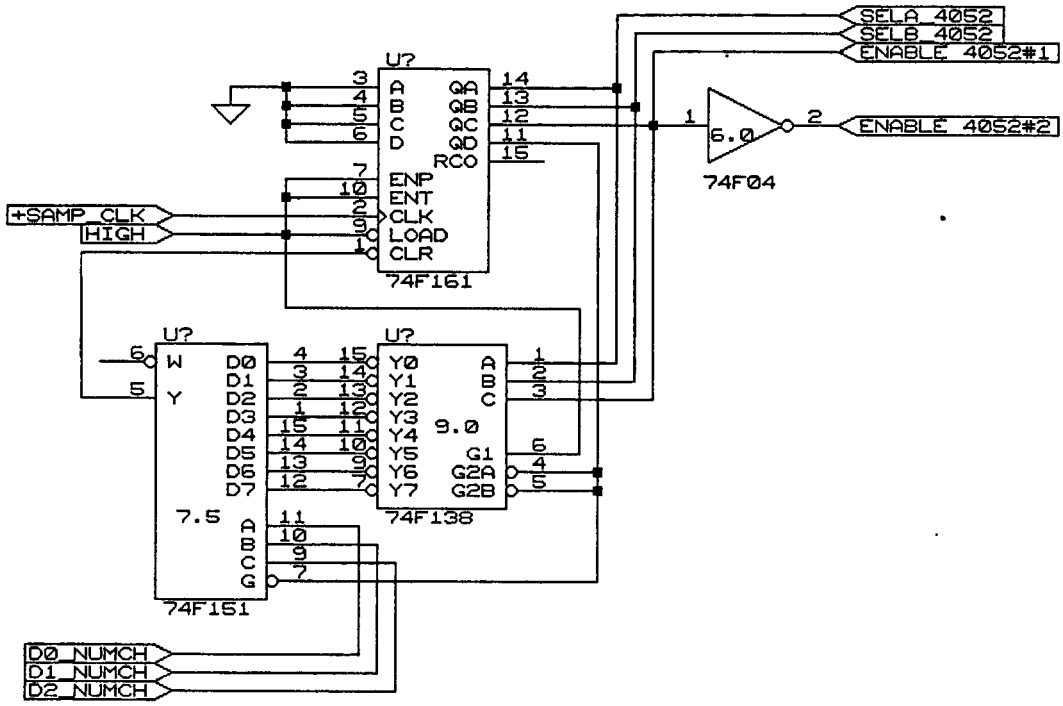
รูป 3.1 แผนภาพโครงสร้างโดยละเอียดของระบบฮาร์ดแวร์

3.2 การออกแบบระบบส่วนอนาลอก

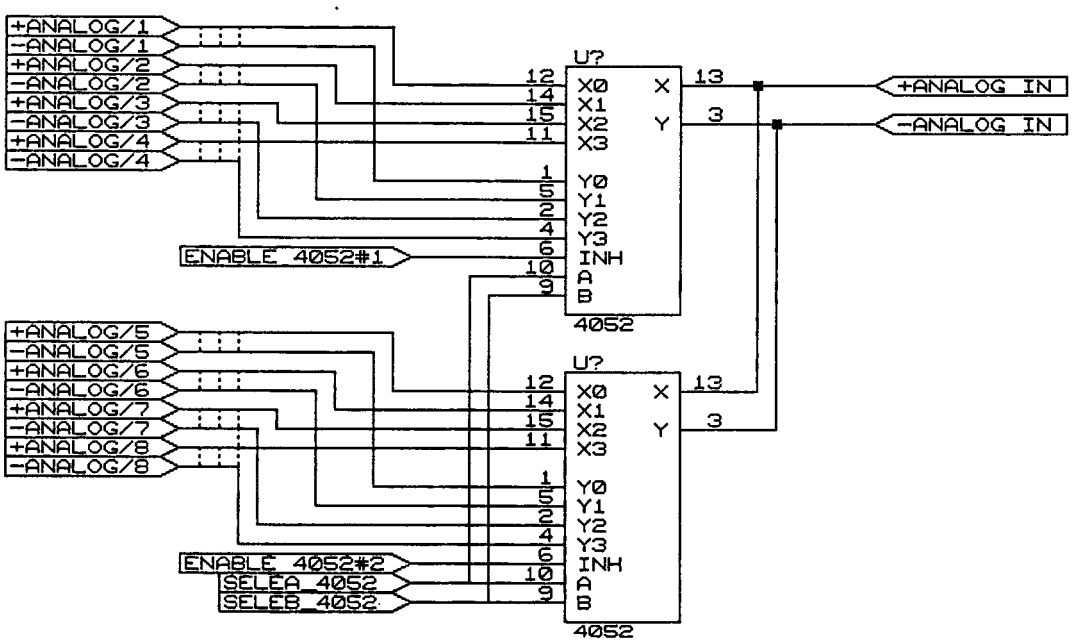
3.2.1 วงจรเลือกช่องสัญญาณเข้า

จากสัญญาณเข้ามีค่าความถี่ได้ตั้งแต่ DC ถึง 5 เมกกะเฮิร์ท เพื่อให้เลือกรับสัญญาณได้ 8 ช่องจึงเลือกใช้ analog switch 4052 differential 4 channel ซึ่งมีค่าของ bandwidth 40 MHz (sine wave) จำนวน 2 ตัวทำหน้าที่เลือกสัญญาณที่เข้าจากทั้งหมด 8 ช่องสัญญาณ โดยจัดให้ 74F161 ทำหน้าที่เป็นวงจรนับแบบเดินหน้าเพื่อกำหนดช่องสัญญาณที่ต้องการอ่านค่าโดยกำหนดให้เริ่มต้นเลือกช่องสัญญาณจากช่อง 0 เสมอและจำนวนช่องสัญญาณที่จะวนรับสัญญาณเข้าต้องกำหนดเข้าที่ขาสัญญาณ D0_NUMCH, D1_NUMCH, และ D2_NUMCH ซึ่งต่อกับ 74F151 ทำการเลือกให้สัญญาณที่ได้รับการ decode จาก 74F138 ผ่านออกไป เมื่อวงจรนับทำการนับ 74F138 จะให้เอาท์พุทของ Y0-Y7 ตามค่าที่อินพุทของ 74F138 ซึ่งจะให้เอาท์พุทเป็นลอจิกต่ำ เมื่อถึงค่าที่กำหนดให้กับ 74F151 ค่าของเอาท์พุทของ 74F138 ซึ่งให้ลอจิกต่ำนั้นจะผ่านออกไปทำการรีเซตวงจรนับให้เริ่มนับค่า 0 ใหม่ ลักษณะวงจรและผังเวลาจะเป็นดังรูป 3.2 และ รูป 3.4 จากการพิจารณาวงจรพบว่าในกรณีที่ทำงานที่ความเร็วสูงจะต้องให้มีการรีเซตค่าใหม่ก่อนที่จะเกิดการนับครั้งต่อไป เนื่องจากความเร็วสูงสุดในการเก็บข้อมูลคือ 20 เมกกะเฮิร์ท ดังนั้นวงจรนับจะต้องถูกรีเซตภายในเวลา 50 นาโนวินาที ดังนั้นจึงเลือกใช้ 74F161, 74F138 และ 74F151 จากคู่มือ IC [1] จะได้ค่าเวลาในการกลับมารีเซตวงจรนับเท่ากับ 9.0 นาโนวินาที

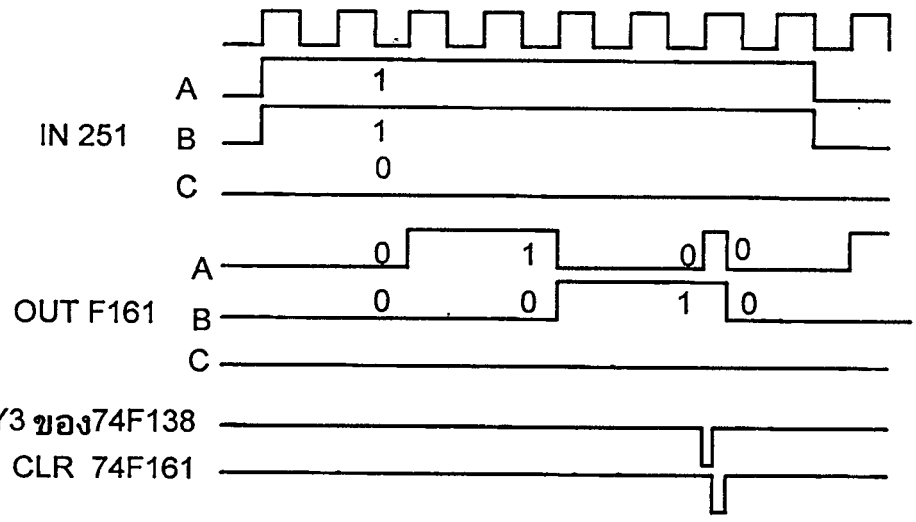
(delay 74F138) + 7.5 นาโนวินาที (delay 74F151) = 16.5 นาโนวินาที



รูป 3.2 วงจรควบคุมการเลือกช่องสัญญาณเข้า



รูป 3.3 วงจรเลือกช่องสัญญาณเข้า



รูป 3.4 ผังเวลาของวงจรนับสำหรับเลือกช่องสัญญาณเข้า

3.2.2 วงจรขยายแบบอินสทรูเมนต์

เพื่อแยกระบบเก็บข้อมูลความเร็วสูงออกจากระบบภายนอกและปรับระดับสัญญาณภายนอกให้มีระดับสัญญาณอ้างอิงของระบบ จึงจัดวงจรทางอินพุทเป็นวงจรขยายแบบอินสทรูเมนต์ จากความเร็วของสัญญาณเข้าอนาลอกมีค่าไม่น้อยกว่า 5 เมกกะเฮิร์ท อัตราขยายสัญญาณสูงสุด 100 เท่า และขนาดความต่างศักย์ของสัญญาณเข้าเป็น 0-2.55 โวลต์ ดังนั้นออปแอมป์ที่นำมาใช้งานต้องพิจารณาคุณสมบัติดังต่อไปนี้

- Gain bandwidth product (GBW)

ในวงจรนี้จะต้องมี GBW เป็นมากกว่า 5MHz x 100 = 500 MHz = 0.5GHz ซึ่งสูงมากเกินกว่าออปแอมป์ทั่วไป จึงจัดแบ่งเป็นวงจรขยาย 2 วงจร โดยให้แต่ละวงจรมีอัตราขยายเป็น 10 เท่า จึงทำให้ค่า GBW ของแต่ละวงจรเป็น 50 เมกกะเฮิร์ท

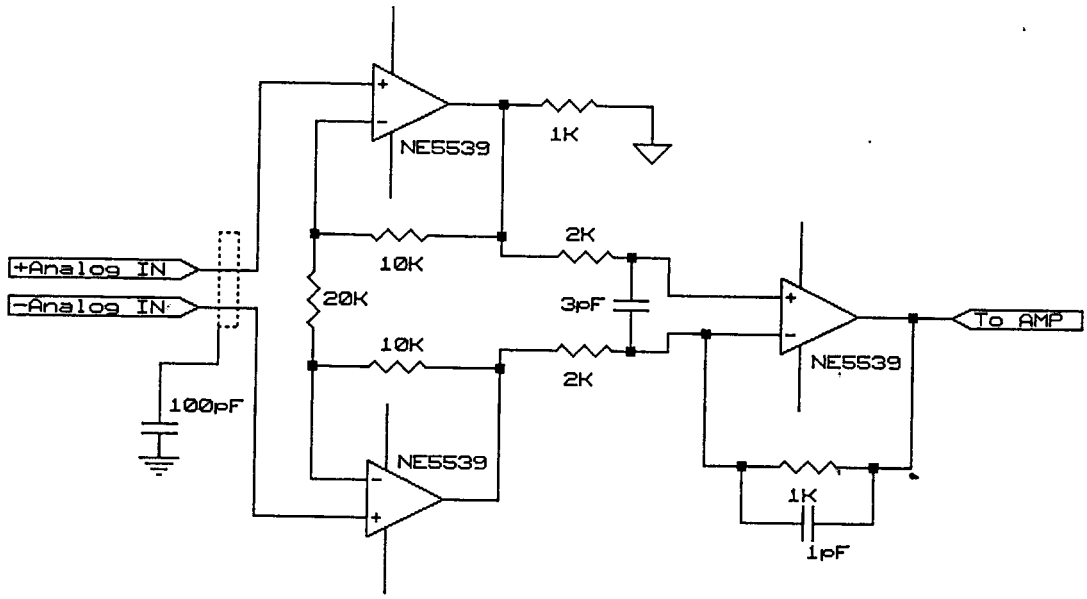
- Slew rate

ด้วยขนาดสัญญาณอินพุท 0-2.55 โวลต์ ความถี่มากกว่า 5 เมกกะเฮิร์ท แสดงว่า สัญญาณอินพุท ที่ได้รับจะมีอัตราการเปลี่ยนแปลงจาก 0 โวลต์เป็น 2.55 ได้ภายใน 0.2 ไมโครวินาที (1/5MHz) ดังนั้น slew rate ของออปแอมป์ที่ใช้ควรมีค่า ไม่ต่ำกว่า 12.75 โวลต์ต่อไมโครวินาที (2.55V/0.2uS)

- Input offset voltage

เนื่องจากสัญญาณอินพุทที่เข้ามีค่าได้สูงสุด 2.55 โวลต์ และถูกแปลงเป็นข้อมูลแบบดิจิตอลขนาด 8 บิท ดังนั้นสัญญาณอินพุทจึงถูกแบ่งเป็นระดับ ระดับละ 10 มิลลิโวลต์ ดังนั้น input offset voltage ของออปแอมป์ที่ใช้จะต้องน้อยกว่า 10 มิลลิโวลต์

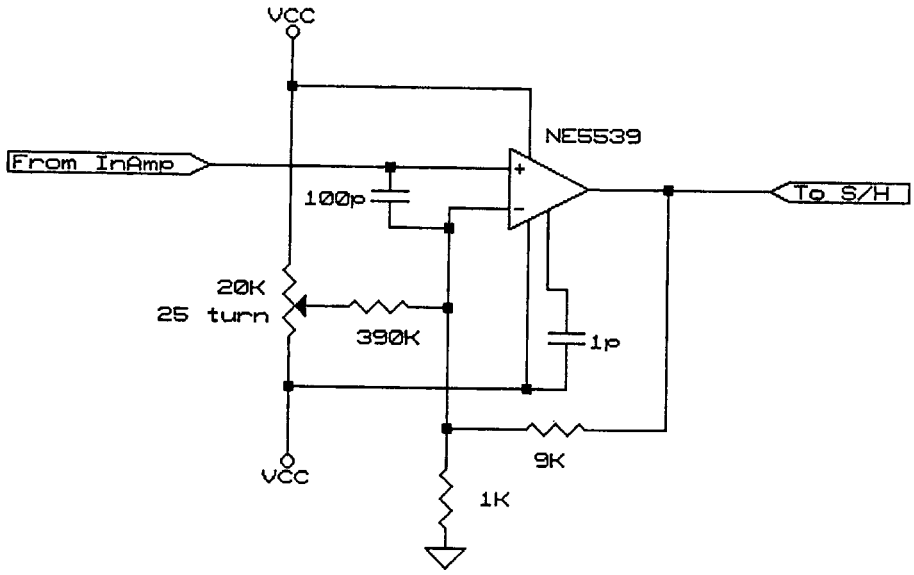
จากข้อมูลที่ได้จึงเลือกใช้ออปแอมป์ NE 5539 ซึ่งมีค่า GBW = 1200 MHz ที่อัตราขยาย 17 dB, GBW = 350 MHz , slew rate = 600 โวลต์ / ไมโครวินาที และ มีค่า input offset voltage น้อยกว่า 5 มิลลิโวลต์ จำนวน 3 ตัวจัดเป็นวงจรขยายแบบอินสทรูเมนต์ ที่มีอัตราขยายเป็น 1 ดังรูป 3.5



รูป 3.5 วงจรขยายแบบอินสทรูเมนต์

3.2.3 วงจรขยายสัญญาณแบบกำหนดการขยายได้

ใช้เพื่อขยายสัญญาณอินพุทที่มีค่าต่ำให้เหมาะสมสำหรับการสุ่มข้อมูล โดยใช้ออปแอมป์ NE 5539 ต่อเป็นวงจรขยายแบบ non - inverting ที่มีอัตราขยายเป็น 1 โดยใช้ trimpot ขนาด 20 กิโลโอห์ม และตัวต้านทาน 390 กิโลโอห์ม ทำหน้าที่ปรับ offset ของวงจร ลักษณะวงจรเป็นดังรูป 3.6



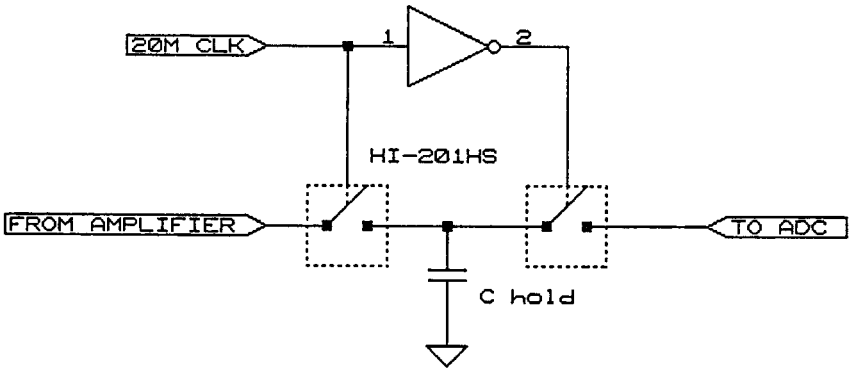
รูป 3.6 วงจรขยายสัญญาณแบบกำหนดการขยายได้

3.2.4 วงจรเก็บและค้างสัญญาณ

เป็นวงจรที่ใช้เพื่อคงค่าความต่างศักย์ของสัญญาณอินพุทให้คงที่ไว้ ในขณะที่วงจรส่วนแปลงข้อมูลจากอนาลอกเป็นดิจิตอลกำลังทำการสุ่มข้อมูลอยู่ ลักษณะของวงจรเป็นดังรูป 3.7 จะเห็นได้ว่าวงจรที่ใช้มีขนาดไม่ซับซ้อน เนื่องจากได้ออกแบบให้วงจรแปลงข้อมูลจากอนาลอกเป็นดิจิตอลทำการสุ่มข้อมูลด้วยอัตราเร็วสูงสุดของระบบคือ 20 ล้านครั้งต่อวินาที ซึ่งมีความเร็วมากพอที่ไม่จำเป็นต้องมีวงจรซดเซย ผลอันเนื่องมาจากรั่วไหลของประจุจากตัวเก็บประจุ C hold ในสภาวะค้างสัญญาณ และด้วยอัตราการสุ่มที่แน่นอนค่าเดียวคือ 20 เมกกะเฮิทซ์ ทำให้สามารถใช้ค่า C เพียงค่าเดียวได้ เวลาที่ใช้ในการประจุให้กับตัวเก็บประจุ C hold สามารถคำนวณได้จากสมการ

เวลาที่ใช้ในการประจุ = ค่าความต้านทานของอนาลอกสวิทซ์ X ค่าของตัวเก็บประจุ

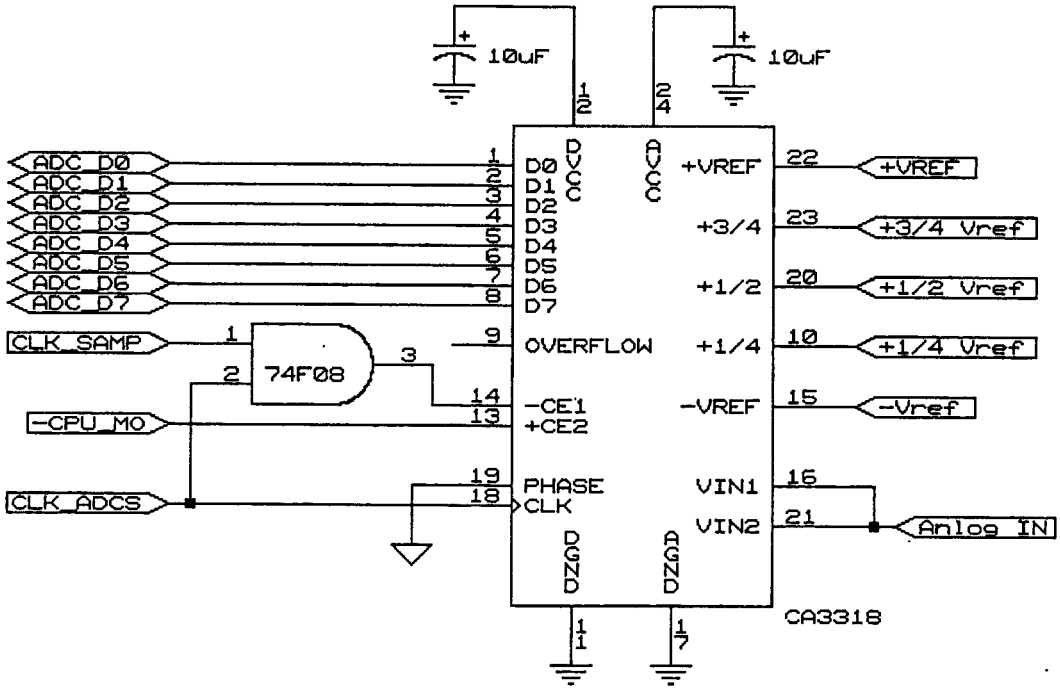
ซึ่งในวิทยานิพนธ์นี้ ได้เลือกใช้ตัวเก็บประจุแบบ เซรามิค เนื่องจากทำงานได้ดีกับความถี่สูง มีค่า 68 พิโคฟารัด และค่าความต้านทานของ อนาลอกสวิทซ์ เมื่อต่อวงจรมีค่าเป็น 35 โอหม์ เมื่อจ่ายแรงดันไฟเลี้ยงเป็น $\pm$15 โวลต์ ดังนั้นเวลาที่ใช้ในประจุตัวประจุ C hold จะมีค่าเป็น 2.38 นาโนวินาที ซึ่งมีค่าน้อย เมื่อเทียบกับการสวิทซ์ในขณะทำการเก็บและค้างสัญญาณซึ่งมีการสวิทซ์ทุก 50 นาโนวินาที สัญญาณที่เอาท์พุทของวงจรเก็บและค้างสัญญาณนี้จะถูกส่งต่อเข้าวงจรแปลงข้อมูลจากอนาลอกเป็นดิจิตอล



รูป 3.7 วงจรเก็บและค้างสัญญาณ

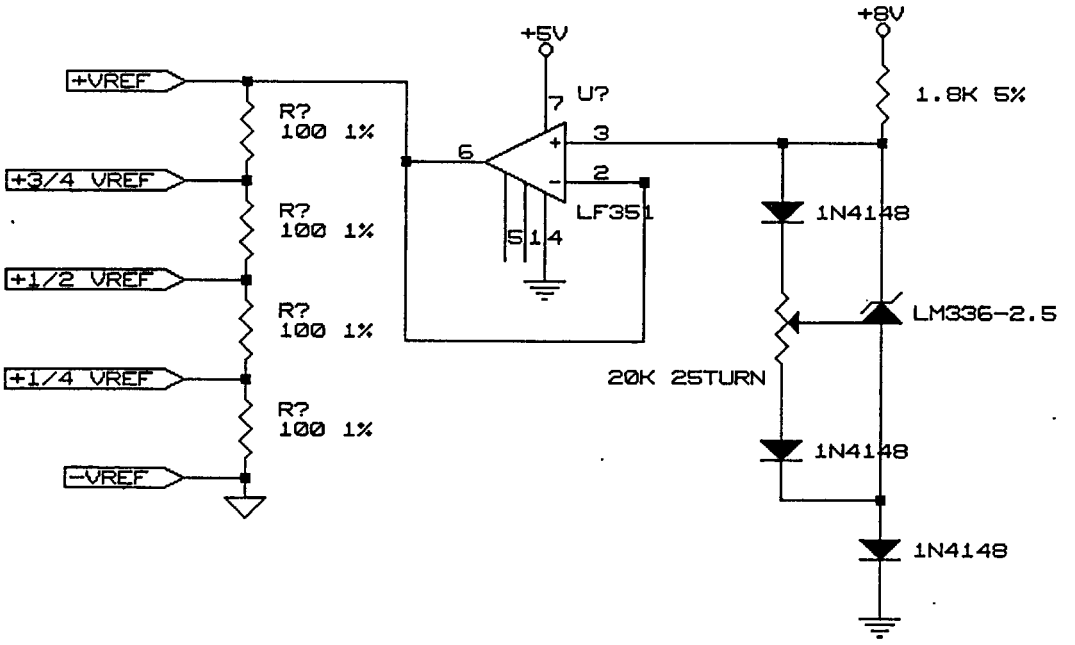
3.2.5 วงจรแปลงข้อมูลจากอนาลอกเป็นดิจิตอล

ในการแปลงข้อมูลจากอนาลอกเป็นดิจิตอลได้ทำการพิจารณาวิธีเข้ารหัสข้อมูลแบบต่าง ๆ พบว่า วิธีการเข้ารหัสแบบ PCM นั้นมีความเหมาะสมมากที่สุดในการใช้งาน เนื่องจากต้องการให้ระบบทำการสุ่มข้อมูลด้วยความเร็วสูงถึง 20 เมกกะเฮิร์ท ดังนั้นจึงกำหนดให้ระบบทำงานด้วยความเร็วสูงสุด 20 เมกกะเฮิร์ทและเก็บข้อมูล 8 บิท จึงเลือกใช้ RCA CA3318 ซึ่งเป็นไอซีแปลงข้อมูลจากอนาลอกเป็นดิจิตอลแบบแฟลชที่สามารถแปลงข้อมูลด้วยความเร็วสูงถึง 20 เมกกะเฮิร์ท โดยมีเอาท์พุทแบบ tri-state เพื่อให้ CA3318 ทำการสุ่มข้อมูลด้วยความสามารถสูงสุด จึงกำหนดให้ CA3318 ทำงานอยู่ในโหมด 1 โดยการต่อขา phase ลงกราวด์ ซึ่ง CA3318 จะทำการแปลงข้อมูลเสร็จภายในครึ่งสัญญาณนาฬิกา เพื่อให้สัญญาณการสุ่มข้อมูล (20เมกกะเฮิร์ท) สัมพันธ์กับสัญญาณเก็บข้อมูล (PRG.CLK) จึงต่อผ่าน 74F08 (AND gate) ลักษณะของวงจรเป็นดังรูป 3.8



รูป 3.8 วงจรแปลงข้อมูลจากอนาลอกเป็นดิจิตอล

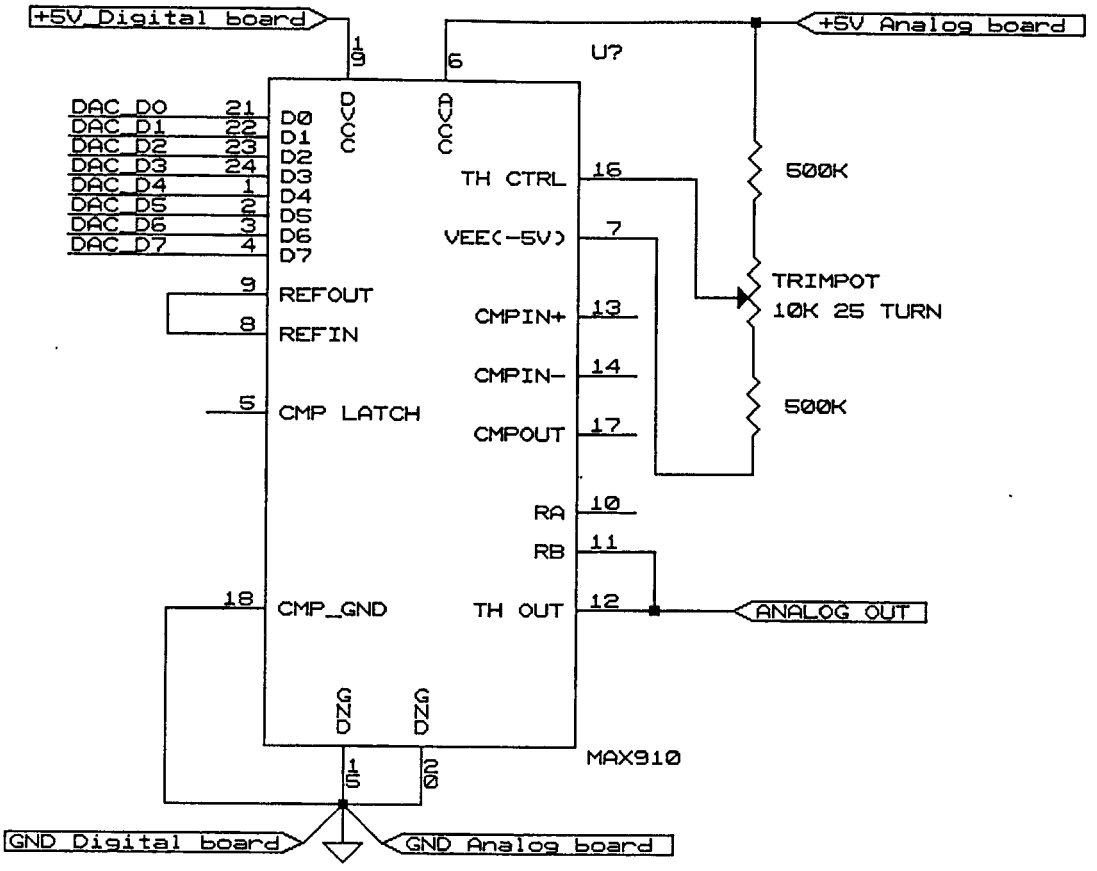
วงจรระดับแรงดันอ้างอิงของ CA3318 ประกอบด้วย LM336-2.5 จัดเป็นวงจรระดับแรงดันอ้างอิงที่ปรับค่าได้และใช้ไดโอด 1N4148 ยกระดับแรงดันอ้างอิงสูงสุดให้สูงขึ้นอีก 0.6 โวลต์เป็น 3.1 โวลต์ ค่าของระดับแรงดันอ้างอิงปรับให้มีค่าเท่ากับ 2.56 โวลต์ด้วย trim pot 20 กิโลโอห์ม 25 รอบ แล้วต่อเข้ากับ LF 351 ทำหน้าที่เป็นวงจรบัฟเฟอร์เพื่อป้องกันการดึงกระแสโดย CA3318 จากนั้นแรงดันอ้างอิงที่ได้จะถูกต่อผ่านวงจรแบ่งระดับแรงดัน ซึ่งประกอบขึ้นจากตัวต้านทาน 100 โอหม์ 1% เพื่อแบ่งแรงดันมีค่าเป็น 0.25, 0.5, 0.75 และ 1 เท่าของแรงดันอ้างอิง จ่ายให้กับ CA3318 ซึ่งเป็นตัวแปลงข้อมูลจากอนาลอกเป็นดิจิตอลแบบแฟลช ดังรูป 3.9



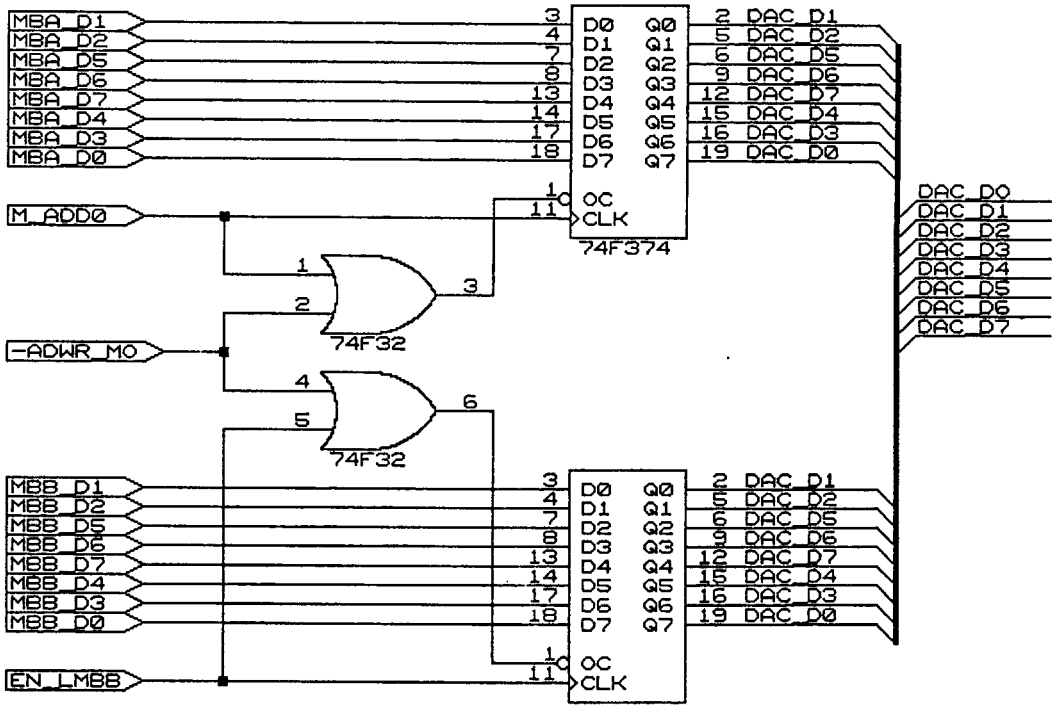
รูป 3.9 วงจรระดับแรงดันอ้างอิงของ CA3318

3.2.6 วงจรแปลงข้อมูลจากดิจิตอลเป็นอนาลอก

ใช้เพื่อนำข้อมูลที่ได้รับการวิเคราะห์และ/หรือประมวลผลแล้วส่งออกให้เป็นรูปสัญญาณอนาลอกเพื่อใช้งานต่อไป ดังรูปที่ 3.10 วงจรส่วนนี้ได้ใช้ MAX 910 ซึ่งเป็น programmable threshold comparator ที่มีวงจรระดับแรงดันอ้างอิงไว้ภายในแล้ว มาทำการดัดแปลงนำเอาเฉพาะส่วนวงจร DAC ความเร็ว 50 นาโนวินาที ซึ่งทำหน้าที่กำหนดค่า threshold voltage มาใช้ ค่าของความต่างศักย์เอาท์พุทสูงสุดกำหนดโดยการเลือกต่อค่าความต้านทานภายใน RA หรือ RB เข้ากับเอาท์พุทเพื่อให้สัมพันธ์กับสัญญาณที่เข้าจึงได้กำหนดให้ต่อกับ RB ซึ่งจะได้ค่าความต่างศักย์เอาท์พุทในช่วง 0.00-2.55 โวลต์ จากหน่วยความจำเก็บข้อมูลมี 2 ชุดจึงต่อเข้ากับ 74F374 สองตัวทำการ latch ข้อมูลที่ได้จากหน่วยความจำเก็บข้อมูลสลับกันควบคุมโดยสัญญาณ M_ADD0 และใช้สัญญาณ ADWR_MO เพื่อกำหนดให้ทำการ latch ข้อมูลเฉพาะเมื่อระบบอยู่ในสภาวะอ่านข้อมูลโดยระบบเก็บข้อมูลเท่านั้น ดังรูป 3.10 (ข)



รูปที่ 3.10(ก) วงจรส่วนกำหนดข้อมูลดิจิตอลให้กับวงจรแปลงข้อมูลจากดิจิตอลเป็นอนาลอก

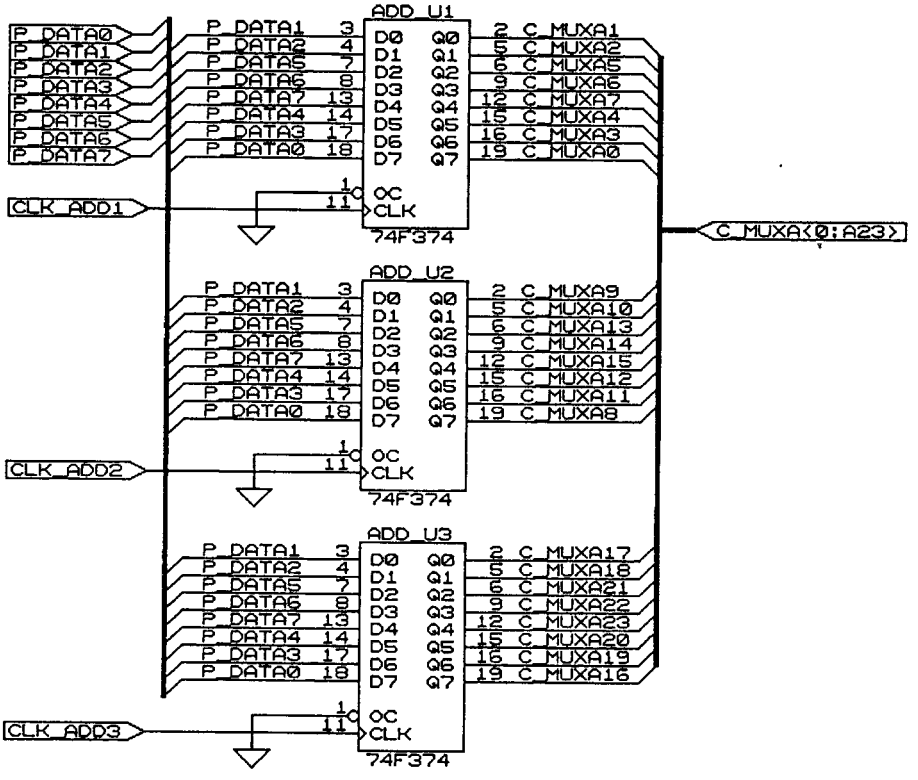


รูปที่ 3.10(ข) วงจรส่วนกำหนดข้อมูลดิจิตอลให้กับวงจรแปลงข้อมูลจากดิจิตอลเป็นอนาลอก

3.3 การออกแบบระบบส่วนดิจิตอล

เนื่องจากระบบเก็บและวิเคราะห์ข้อมูลความเร็วสูง ทำการสุ่มข้อมูลด้วยความเร็ว 20 เมกกะเฮิตซ์ ดังนั้น ข้อมูลดิจิตอลที่ได้จากวงจรแปลงสัญญาณอนาลอกเป็นดิจิตอล จะเกิดการเปลี่ยนแปลงทุก 50 นาโนวินาที จากการพิจารณาอุปกรณ์ Programmable timer/counter โดยทั่วไปที่มีอยู่ในประเทศ พบว่าไม่สามารถทำงานที่ความเร็วที่ต้องการได้ จึงทำการออกแบบวงจรส่วนดิจิตอลทั้งหมดขึ้นเองโดยใช้อุปกรณ์ดิจิตอลพื้นฐาน

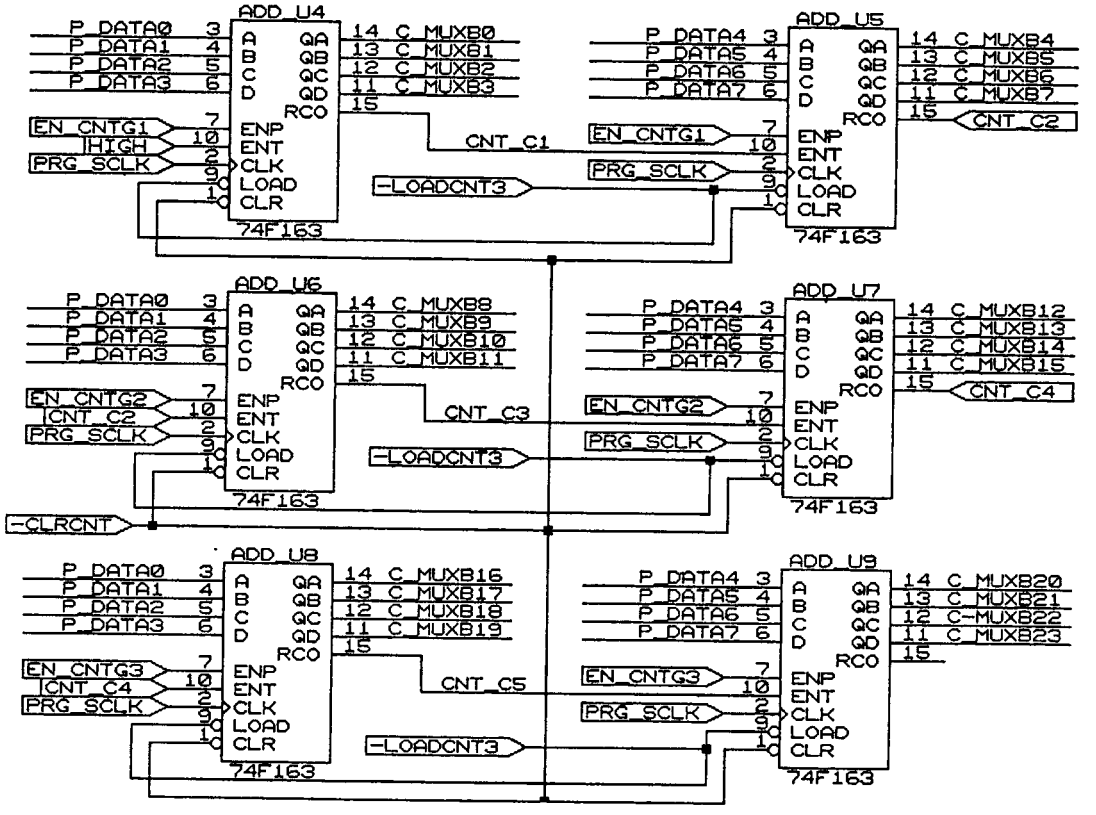
3.3.1 วงจรกำเนิดสัญญาณตำแหน่งหน่วยความจำ



รูป 3.11 วงจรกำเนิดสัญญาณตำแหน่งหน่วยความจำจากระบบคอมพิวเตอร์

วงจรส่วนนี้จะแบ่งออกเป็น 2 ส่วน คือ ส่วนกำเนิดสัญญาณตำแหน่งหน่วยความจำเพื่อใช้ในสภาวะเก็บข้อมูลและส่วนกำเนิดสัญญาณตำแหน่งหน่วยความจำเมื่อระบบคอมพิวเตอร์ต้องการติดต่อหน่วยความจำ ในส่วนที่ทำหน้าที่ให้กำเนิดสัญญาณตำแหน่งหน่วยความจำเพื่อใช้ในสภาวะเก็บข้อมูล กลุ่มสัญญาณตำแหน่งหน่วยความจำนี้มีการเปลี่ยนแปลงค่าแบบเดินหน้าโดยสัมพันธ์กับความเร็วในการเก็บข้อมูล เมื่อระบบทำการเก็บข้อมูล ที่ความเร็ว 20 เมกกะเฮิตซ์ แสดงว่าตำแหน่งหน่วยความจำต้องเปลี่ยนแปลงภายในเวลาน้อยกว่า 50 นาโนวินาที ดังนั้นเอาท์พุทของวงจรนับทุกบิทจะต้องมีการเปลี่ยนแปลงเสร็จสิ้นภายใน 50 นาโนวินาทีเช่นกัน จึงใช้ 74163 ซึ่งเป็นวงจรนับแบบไบนารีขนาด 4 บิทที่กำหนดจุดเริ่มนับได้ เพื่อให้สามารถกำหนดจุดเริ่มต้นของการนับได้และสามารถรีเซทเอาท์พุทกลายเป็นศูนย์ทุกบิทโดยสัมพันธ์กับสัญญาณนาฬิกาที่ต่อให้ ในการเลือกตระกูลของอุปกรณ์ที่นำมาใช้งานพบว่า 74LS163 มีค่า propragation delay time (max) เป็น 27 นาโนวินาที ซึ่งใกล้เคียงเวลาที่ใช้ในการสุ่มข้อมูลแต่ละครั้ง จึงพิจารณาเลือกอุปกรณ์ TTL ตระกูล F ที่มีค่า propragation delay time เป็น 11.5 นาโนวินาทีแทน ดังนั้นวงจรจึงประกอบด้วย 74F163 ซึ่งเป็นวงจรนับแบบไบนารีขนาด 4 บิท จำนวน 6 ตัวต่อแบบเรียงกันเป็นวงจรนับขนาด 24 บิท เพื่อให้อ้างหน่วยความจำได้สูงสุด 16 เมกกะไบท์ โดยที่สัญญาณ reset, load

กันเป็นวงจรนับขนาด 24 บิท เพื่อให้อ้างหน่วยความจำได้สูงสุด 16 เมกกะไบท์ โดยที่สัญญาณ reset, load และสัญญาณนาฬิกาของวงจรนับทุกตัวถูกต่อเข้าด้วยกัน โดยแยกตามกลุ่มเพื่อควบคุมการ reset, load และการนับให้พร้อมกันทั้ง 6 ตัว ดังรูป 3.12



รูป 3.12 วงจรนับกำเนิดสัญญาณตำแหน่งหน่วยความจำ

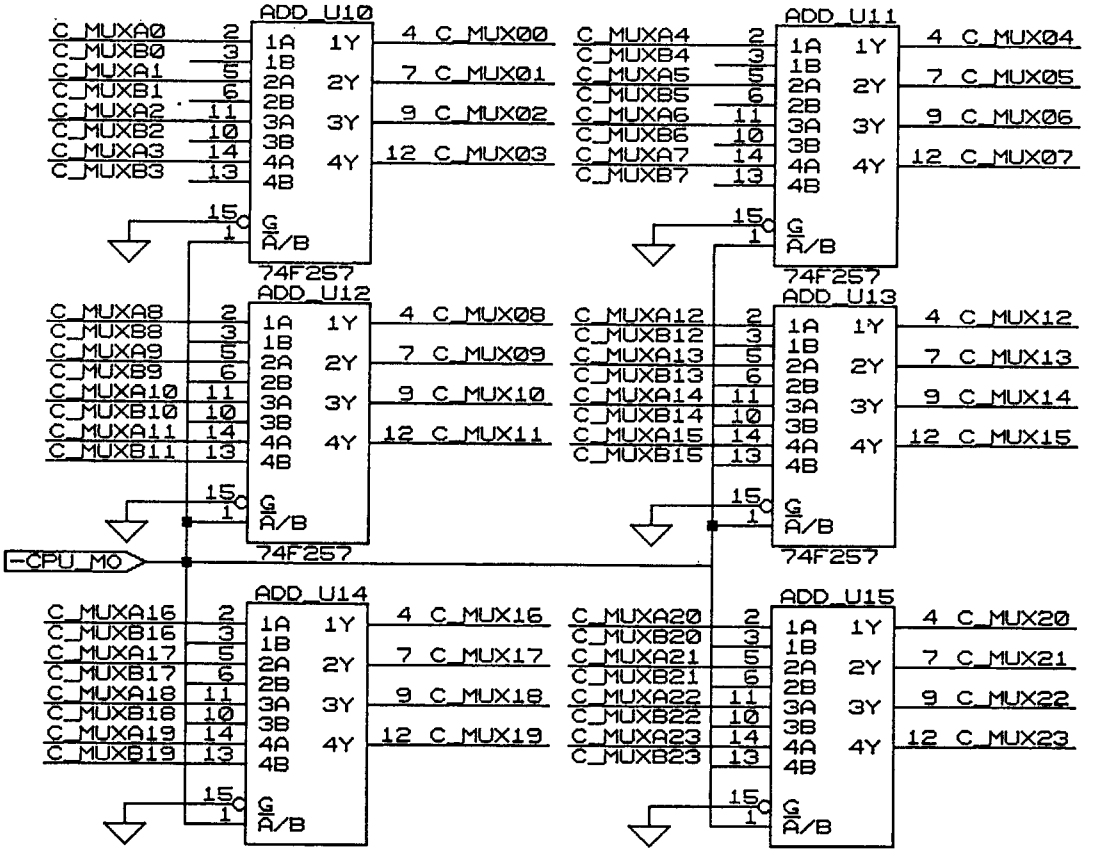
ในการติดต่อหน่วยความจำจากระบบคอมพิวเตอร์ที่นำมาเชื่อมต่อ เนื่องจากข้อกำหนดในการออกแบบที่ต้องการให้สามารถนำไปใช้งานกับระบบคอมพิวเตอร์ที่แตกต่างกันได้ และข้อกำหนดในการติดต่อฮาร์ดแวร์กับระบบปฏิบัติการไมโครซอฟท์วินโดว์ส ซึ่งกำหนดไว้ว่าโปรแกรมใดที่พัฒนาขึ้นภายใต้ระบบปฏิบัติการไมโครซอฟท์วินโดว์สต้องไม่มีการติดต่อกับฮาร์ดแวร์ใดโดยการติดต่อหน่วยความจำโดยตรง จึงใช้วิธีกำหนดค่าตำแหน่งโดยผ่านพอร์ท B ของ 8255 โดยแบ่งตำแหน่งหน่วยความจำ ขนาด 24 บิท ออกเป็น 3 ไบท์ และโปรแกรมค่าให้กับ 74F373 สามตัว ซึ่งเป็นค้างตำแหน่งหน่วยความจำทั้ง 24 บิท โดยโปรแกรมตำแหน่งผ่านพอร์ท C กลุ่มสัญญาณตำแหน่งหน่วยความจำทั้งสองกลุ่มนี้ จะนำไปต่อกับวงจรเลือกกลุ่มสัญญาณเพื่อเลือกสัญญาณที่จะติดต่อหน่วยความจำต่อไป ดังรูป 3.11

นอกจากกำหนดจุดเริ่มต้นการนับของวงจรนับแล้ว ในการใช้งานบางครั้งต้องการเก็บข้อมูลเพียงกลุ่มหนึ่งไม่ได้ต้องการเก็บจนหมดหน่วยความจำ จึงได้ออกแบบวงจรเพื่อหยุดการนับให้ตรงตำแหน่งที่ต้องการ โดยใช้ 74F521 ต่อเป็นวงจรเปรียบเทียบตำแหน่งตำแหน่งหน่วยความจำที่ให้หยุดนับ โดยค่าที่เปรียบเทียบนั้นได้มาจากการโปรแกรมตำแหน่งหน่วยความจำที่ต้องการหยุดให้กับ 74F374 ที่ทำหน้าที่เป็น วงจรกำเนิดสัญญาณตำแหน่งหน่วยความจำจากระบบคอมพิวเตอร์ เป็นการประหยัดอุปกรณ์เนื่องจากในสภาวะเก็บข้อมูลระบบเก็บและวิเคราะห์

ข้อมูลความเร็วสูงไม่มีการใช้งานในหน้าที่การติดต่อกับหน่วยความจำ เมื่อตำแหน่งหน่วยความจำทั้งสองตรงกันจะทำให้เอาท์พุทที่ 74F521 ทั้งสามเปลี่ยนเป็นลอจิก "0" และเอาท์พุททั้งสามนี้ก็นำไปต่อกับ nor gate เพื่อให้เอาท์พุทสุดท้ายเป็น 1 เป็นการกำหนด or gate ที่ทำหน้าที่ควบคุมการเปิดปิดการให้ผ่านของสัญญาณนาฬิกาเพื่อการเก็บข้อมูลนั้นปิด ทำให้วงจรนับหยุดการทำงานในที่สุด ซึ่งจะได้กล่าวถึงโดยละเอียดในหัวข้อ 3.3.4 ต่อไป

3.3.2 วงจรเลือกกลุ่มตำแหน่งหน่วยความจำและค้างตำแหน่งหน่วยความจำ

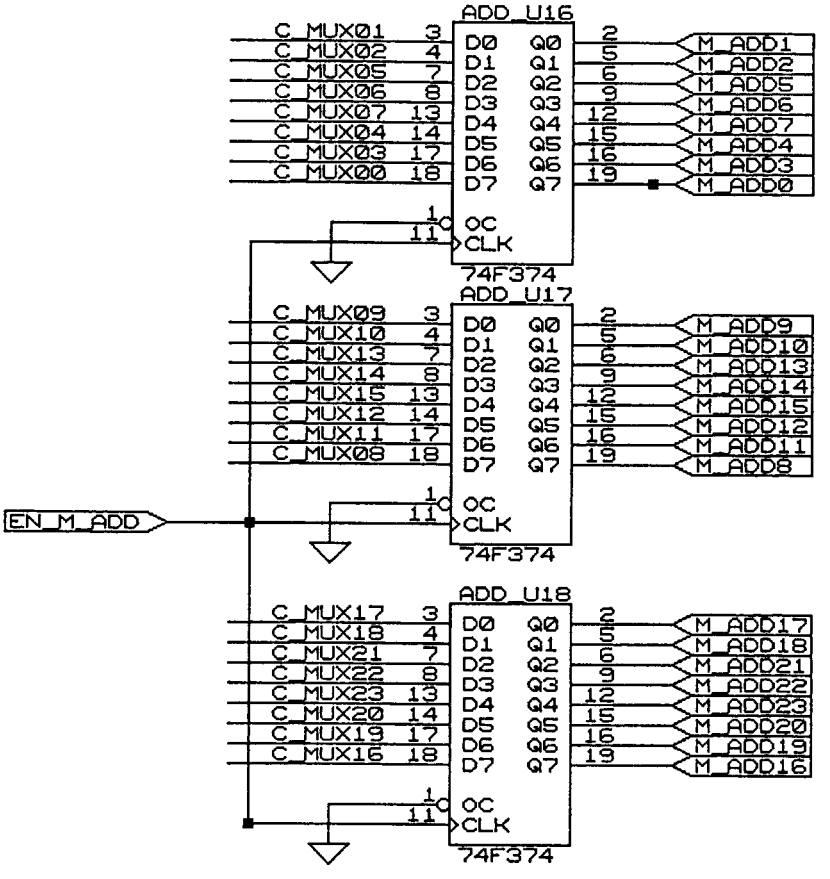
วงจรเลือกกลุ่มตำแหน่งหน่วยความจำในรูปที่ 3.13 (ก) จะทำหน้าที่เลือกกลุ่มสัญญาณตำแหน่งหน่วยความจำที่จะติดต่อกับหน่วยความจำ ระหว่างตำแหน่งหน่วยความจำจากวงจรนับ กับตำแหน่งหน่วยความจำที่มาจากระบบคอมพิวเตอร์ที่เชื่อมต่ออยู่ โดยใช้ Multiplexer แบบ 2 เป็น 1 74F257 จำนวน 6 ตัวต่อขนานกัน เป็นวงจรเลือกตำแหน่งหน่วยความจำขนาด 24 บิท โดยสัญญาณควบคุมจะใช้สัญญาณ -CPU_MO ซึ่งผู้ใช้สามารถกำหนดได้ว่าขณะนี้ต้องการให้หน่วยความจำติดต่อกับระบบใด



รูป 3.13 (ก) วงจรเลือกกลุ่มตำแหน่งหน่วยความจำ

เนื่องจากเวลาของการอ่านเขียนหน่วยความจำในระหว่างการเก็บข้อมูล มีเวลาที่สั้นกว่าเวลาที่ใช้ในการอ่านเขียนหน่วยความจำ จึงออกแบบให้ใช้เทคนิคการติดต่อหน่วยความจำแบบ bank switching การทำงานจะเริ่มต้นเมื่อวงจรนับหน่วยความจำตำแหน่งคู่ สัญญาณตำแหน่งหน่วยความจำจะผ่านเข้าวงจรค้างสัญญาณ ดังรูป 3.13 (ข) ซึ่งประกอบขึ้นจาก 74F374 และได้รับสัญญาณให้ค้างข้อมูลจากวงจรควบคุมการอ่านเขียนมีความเร็วสูง ซึ่งจะกล่าวถึงในหัวข้อ 3.3.4 ทำให้ตำแหน่งหน่วยความจำที่ส่งให้หน่วยความจำกลุ่มคู่คงค้างไว้จนกว่ามีการติดต่อตำแหน่งคู่อีกครั้ง ในขณะที่ตำแหน่งหน่วยความจำจากวงจรนับเปลี่ยนเป็นตำแหน่งคี่ สัญญาณตำแหน่งหน่วยความจำที่ตำแหน่งคู่ก็ยังคงค้างอยู่ เนื่องจากไม่มีสัญญาณควบคุมไปเปลี่ยนสภาวะของวงจรค้างสัญญาณ

แหน่งคู่ก็ยังคงค้างอยู่ เนื่องจากไม่มีสัญญาณควบคุมไปเปลี่ยนสภาวะของวงจรค้างสัญญาณ

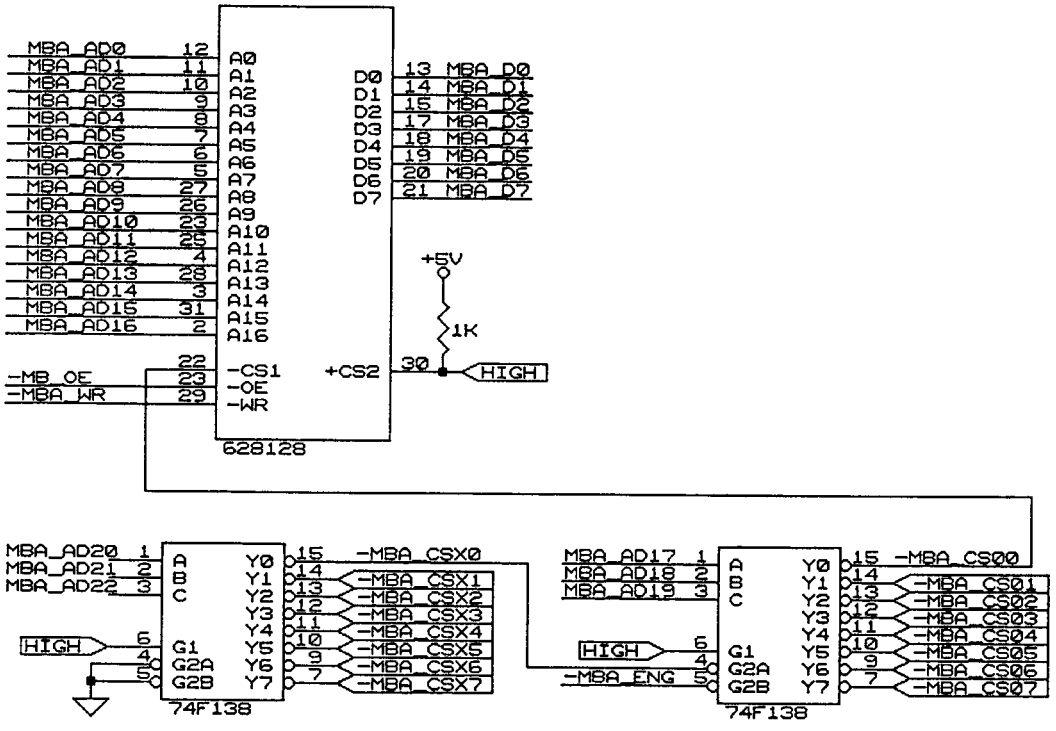


รูป 3.13(ข) วงจรค้างตำแหน่งหน่วยความจำ

3.3.3 วงจรหน่วยความจำ

เนื่องจากเวลาในการเก็บข้อมูลแต่ละครั้งประมาณ 50 นาโนวินาที จากการพิจารณาพบว่าหน่วยความจำแรมแบบไดนามิกไม่เหมาะสมในการใช้งานถึงแม้ว่าจะมีราคาต่อขนาดความจุถูกกว่าก็ตาม ปัจจุบันไดนามิกแรมมีความเร็วมากที่สุดที่หาได้ในประเทศคือ 70 นาโนวินาที แต่เนื่องจากไดนามิกแรมต้องมีขั้นตอนในการรีเฟรชข้อมูล ซึ่งต้องใช้เวลาเท่ากับการอ่านเขียนหนึ่งครั้งแม้ว่าในไดนามิกแรมรุ่นใหม่ จะมีการรีเฟรชแบบ hidden ได้ก็ตามแต่ก็ยังต้องเสียเวลาไปอีกประมาณครึ่งหนึ่งของเวลาในการอ่านเขียนปกติ แต่ในการเก็บข้อมูลจะต้องเก็บข้อมูลให้ทันข้อมูลใหม่เสมอ จึงเลือกใช้หน่วยความจำแบบสแตติก 628128 ขนาด 128 กิโลไบท์ ที่มีเวลาของการเข้าถึงข้อมูลเป็น 70 นาโนวินาที จำนวน 2 ตัว เป็นหน่วยความจำหลักของระบบ จะเห็นได้ว่าหน่วยความจำมีความเร็วใกล้เคียงกับเวลาในการอ่านเขียนของระบบมาก ถ้าทำงานที่ความเร็วสูงขึ้นอีกเพียงเล็กน้อยหน่วยความจำก็จะทำงานไม่ได้ทันที เพื่อแก้ข้อจำกัดนี้จึงได้ออกแบบส่วนของหน่วยความจำทำงานแบบ bank switching โดยให้ค้างสัญญาณต่างๆ ที่จำเป็นในการอ่านเขียนไว้จนกว่าจะครบเวลาทำงานของหน่วยความจำแล้วจึงเปลี่ยนค่า ในขณะที่ค้างสัญญาณของชุดแรกอยู่นั้น ระบบจะย้ายไปติดต่อกับกลุ่มหน่วยความจำอีกชุดหนึ่ง ทำให้เพิ่มเวลาในการอ่านเขียนหน่วยความจำได้ครึ่งหนึ่งของเวลาในการอ่านเขียนเดิม กลุ่มสัญญาณตำแหน่งหน่วยความจำจากวงจรเลือกกลุ่มตำแหน่งหน่วยความจำจะต่อเข้ากับ 74F374 ทำหน้าที่เป็นบัฟเฟอร์ 2 ชุด โดยที่ชุดหนึ่งจะทำงานเมื่อตำแหน่ง

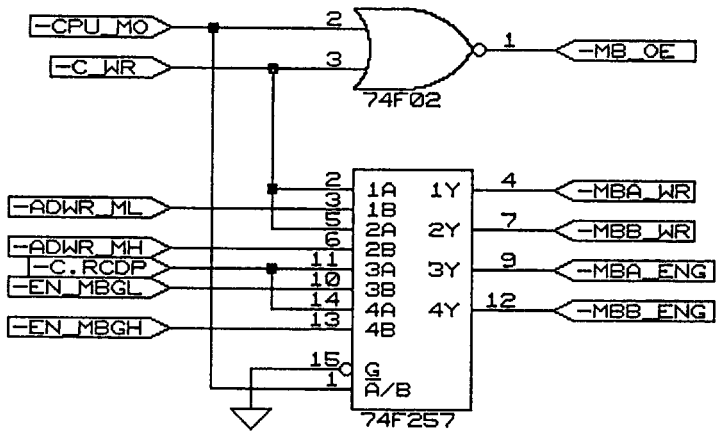
กลุ่มตำแหน่งหน่วยความจำจะต่อเข้ากับ 74F374 ทำหน้าที่เป็นบัฟเฟอร์ 2 ชุด โดยที่ชุดหนึ่งจะทำงานเมื่อตำแหน่งหน่วยความจำติดต่อหน่วยความจำเป็นตำแหน่งหน่วยความจำคี่ และอีกชุดติดต่อเมื่อเป็นตำแหน่งหน่วยความจำคู่ โดยลักษณะการจัดวงจรหน่วยความจำของทั้งสองชุดนี้จะเหมือนกันทั้งหมดแตกต่างเฉพาะจังหวะของสัญญาณอ่านเขียนหน่วยความจำเท่านั้น ลักษณะการจัดวงจรของแต่ละชุดประกอบด้วยตำแหน่งหน่วยความจำ MB#_A0-MB#_A17 ถูกต่อเข้ากับ 628128 ทุกตัว MB#_A18-MB#_A22 ถูกจัดให้ต่อเข้ากับ 74F138 เพื่อแบ่งช่วงหน่วยความจำ 16 MB ออกเป็น 8 ส่วน และผ่านเข้า 74F138 เพื่อทำการแบ่งย่อยออกเป็นอีก 8 ส่วน ๆ ละ 128 KByte แต่ในระบบเริ่มต้นใช้เพียงชุดละ 128 KByte ซึ่งเพิ่มเติมได้ภายหลัง ลักษณะของวงจรเป็นดังรูป 3.14



รูป 3.14 วงจรหน่วยความจำและส่วน decode

3.3.4 วงจรควบคุมจังหวะการอ่านเขียนหน่วยความจำที่ความเร็วสูง

เนื่องจากระบบใช้เทคนิคในการสลับหน่วยความจำในการเก็บข้อมูลและความเร็วในการอ่านเขียนหน่วยความจำอยู่ในระดับนาโนวินาที จึงออกแบบวงจรควบคุมจังหวะการอ่านเขียนหน่วยความจำขึ้นเองโดยใช้วงจรดิจิตอล เนื่องจากสัญญาณที่ติดต่อกับหน่วยความจำเก็บข้อมูลจะมีมาจากสองแหล่งคือจากระบบคอมพิวเตอร์และวงจรเก็บข้อมูล สำหรับสัญญาณตำแหน่งหน่วยความจำนั้นมีวงจรเลือกสัญญาณตำแหน่งหน่วยความจำทั้งสองกลุ่มแล้วสัญญาณ -CS ของหน่วยความจำได้มาจากการ decode ตำแหน่งหน่วยความจำที่มาจากวงจรเลือกสัญญาณตำแหน่งหน่วยความจำสิ่งที่ต้องพิจารณาต่อไปคือสัญญาณที่จะต่อเข้ากับขาสัญญาณ OE,WR ของหน่วยความจำเก็บข้อมูล ซึ่งสัญญาณที่นำมาต่อนี้จะแยกกันเด็ดขาดไม่มีความสัมพันธ์กันจึงใช้ 74F251 ทำหน้าที่เลือกกลุ่มของ

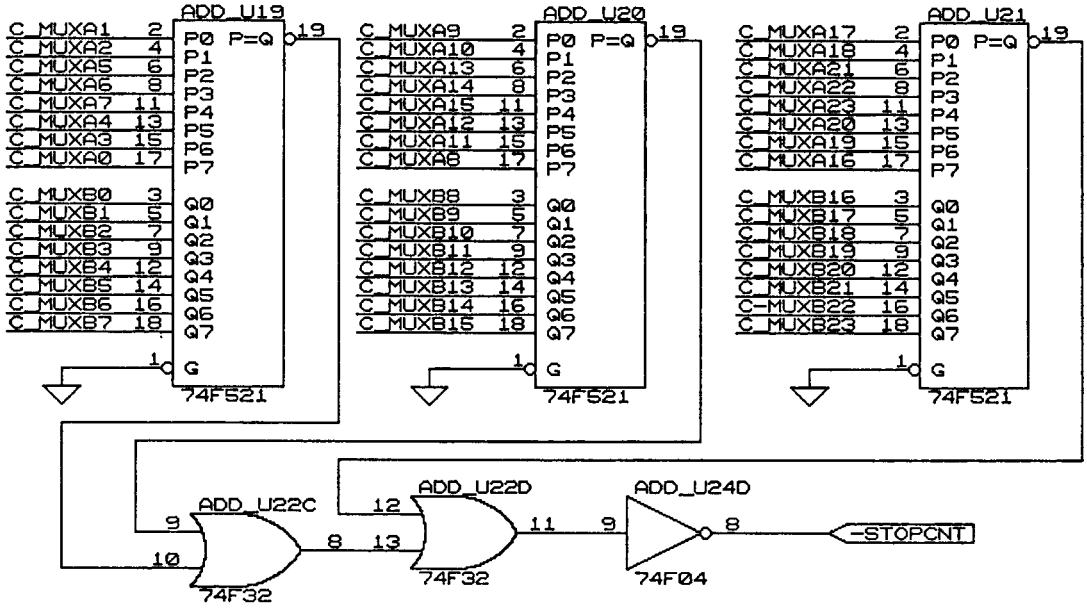


รูป 3.15 วงจรส่วนเลือกกลุ่มสัญญาณที่จะติดต่อกับหน่วยความจำเก็บข้อมูล

จากวงจรกำหนดให้สัญญาณกลุ่ม A เป็นกลุ่มสัญญาณที่มาจากระบบคอมพิวเตอร์ และกลุ่ม B เป็นกลุ่มสัญญาณที่มาจากระบบเก็บข้อมูล สัญญาณที่เป็นสัญญาณ WR ให้กับหน่วยความจำเก็บข้อมูลได้เลือกให้ใช้สัญญาณ C_WR ซึ่งเป็นสัญญาณเขียนที่มาจากระบบคอมพิวเตอร์โดยตรงและต่อให้กับหน่วยความจำเก็บข้อมูลทั้งสองชุดเพราะระบบคอมพิวเตอร์จะต้องมองหน่วยความจำทั้งสองเป็นกลุ่มเดียวโดยมีตำแหน่งหน่วยความจำสลับคู่คี่ ในสภาวะที่ระบบเก็บข้อมูลทำการติดต่อกับหน่วยความจำ (ขาสัญญาณ -CPU_MO เป็นลอจิกต่ำ) วงจรนี้จะทำหน้าที่สลับผ่านสัญญาณ ADWR_ML (Address WRite Memory Low) และ ADWR_MH (Address WRite Memory High) ไปเป็นสัญญาณเขียนหน่วยความจำและผ่านสัญญาณ EN_MBGL (ENable Memory Bank Gate Low) กับ EN_MBGH (ENable Memory Bank Gate High) ไปเป็นสัญญาณกำหนดการกำเนิดพัลส์สำหรับขาสัญญาณ -CS ของหน่วยความจำ เมื่ออยู่ในช่วงระบบคอมพิวเตอร์ติดต่อกับหน่วยความจำ(ขาสัญญาณ -CPU_MO เป็นลอจิกสูง) วงจรนี้จะสลับให้สัญญาณ -C_WR (Card WRite) ไปเป็นสัญญาณเขียนของหน่วยความจำสำหรับขาสัญญาณ -OE ของหน่วยความจำนั้น จะใช้ nor gate ทำหน้าที่กลับลอจิกของสัญญาณ -C_WR เมื่อระบบคอมพิวเตอร์ติดต่อกับหน่วยความจำมาต่อ เพื่อป้องกันไม่ให้เกิดสภาวะอ่านข้อมูลจากหน่วยความจำในขณะที่ระบบกำลังเกิดการเปลี่ยนแปลงของสัญญาณที่ใช้เขียนหน่วยความจำและเป็นการกำหนดให้หน่วยความจำอยู่ในสภาวะการทำงานแบบ -OE เป็นลอจิกต่ำ ซึ่งจะป้องกันปัญหาที่ข้อมูลถูกส่งออกมาจากหน่วยความจำโดยไม่ต้องการได้ สำหรับการสร้างสัญญาณ ADWR_ML และADWR_MH ได้กล่าวถึงโดยละเอียดในส่วนหน่วยความจำเก็บข้อมูลแล้ว

จังหวะการเขียนหน่วยความจำ จะเริ่มต้นจากสัญญาณนาฬิกาที่ใช้กำหนดอัตราเก็บข้อมูลจ่ายให้กับ or gate โดยมีสัญญาณ -OSC_ON ซึ่งมาจากพอร์ท A บิท 2 ของ 8255 ทำหน้าที่ควบคุมการปิด/เปิดแล้วผ่านให้กับ or gate อีกตัวหนึ่งซึ่งทำการปิด/เปิดโดยสัญญาณจากวงจรเปรียบเทียบตำแหน่งหน่วยความจำเพื่อให้หยุดนับ

เมื่อถึงตำแหน่งหน่วยความจำที่ต้องการโดยการตัดสัญญาณนาฬิกาที่จ่ายให้กับวงจรนับ วงจรส่วนเปรียบเทียบค่าจะเป็นดังรูป 3.16

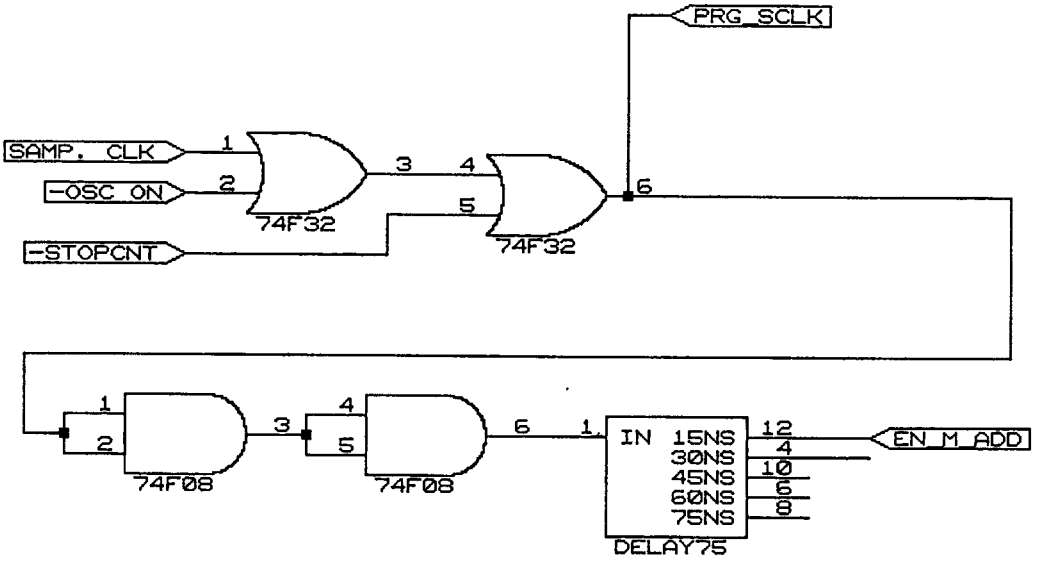


รูป 3.16 วงจรเปรียบเทียบค่าตำแหน่งหน่วยความจำ

เนื่องจากระบบเก็บข้อมูลทำงานด้วยความเร็วค่อนข้างสูง จึงใช้วงจรเปรียบเทียบค่าแบบดิจิตอลพื้นฐานทำการเปรียบเทียบค่าที่ได้จากวงจรนับตำแหน่งหน่วยความจำ และค่าที่ได้จากวงจรกำเนิดสัญญาณตำแหน่งหน่วยความจำ เมื่อทุกบิทมีค่าเท่ากัน วงจรส่วนนี้จะให้เอาท์พุทเป็นลอจิกสูง จึงนำมากลับลอจิก เพื่อจ่ายให้กับ or gate ทำหน้าที่หยุดการผ่านสัญญาณนาฬิกา สิ่งที่ต้องพิจารณาสำหรับวงจรส่วนนี้คือ วงจรจะต้องทำการเปรียบเทียบแล้วได้ผลลัพธ์ในการเปรียบเทียบก่อนที่สัญญาณนาฬิกาเก็บข้อมูลลูกใหม่จะมา จากพิจารณาพบว่า 74F521 กับอินเวอร์เตอร์เกต 74F04 จะมีค่าหน่วงของอุปกรณ์ไม่เกิน 7 และ 6 นาโนวินาทีตามลำดับ ดังนั้น ในส่วนวงจรเปรียบเทียบค่านี้จะให้เอาท์พุท ในเวลาไม่เกิน 11+7+7+6 = 31 นาโนวินาที ซึ่งเมื่อทำการทดลองกับความเร็วการเก็บข้อมูลระดับ 20 เมกกะเฮิตซ์ ซึ่งมีคาบเวลาเป็น 50 นาโนวินาที วงจรนี้ก็ยังคงทำงานได้อย่างไม่ผิดพลาด

หลังจากสัญญาณนาฬิกาเก็บข้อมูลผ่าน or gate ซึ่งควบคุมโดยวงจรหยุดนับเวลา สัญญาณนี้จะนำไปจ่ายให้กับวงจรนับตำแหน่งหน่วยความจำต่อไป สัญญาณส่วนหนึ่งจะถูกส่งเข้าและถูกหน่วงโดย and gate สองตัว ซึ่งจะหน่วงไปไม่น้อยกว่า 14 นาโนวินาที แล้วผ่านเข้า delay line หน่วงไปอีก 15 นาโนวินาที รวมเป็น 29 นาโนวินาที เพื่อให้มั่นใจว่าค่าตำแหน่งหน่วยความจำซึ่งกำเนิดโดยวงจรนับได้ผ่านวงจรเลือกกลุ่มตำแหน่งหน่วยความจำตรงที่อินพุทของ 74F374 แล้วแน่นอน ซึ่งเวลาที่ใช้ตั้งแต่วงจรนับได้รับสัญญาณนาฬิกาให้กำเนิดค่าตำแหน่งบนหน่วยความจำและผ่านวงจรเลือกกลุ่มตำแหน่งหน่วยความจำนั้น มีค่าไม่เกิน 11 และ 6 นาโนวินาทีตามลำดับ ซึ่งรวมแล้วไม่เกิน 17 นาโนวินาที ดังนั้นสัญญาณที่หน่วงไปประมาณ 29 นาโนวินาที นี้จะทำหน้าที่ทริกให้ 74F374 ผ่านค่าตำแหน่งหน่วยความจำให้กับ 74F374 สองชุดซึ่งทำหน้าที่ค้างค่าตำแหน่งหน่วยความจำในระหว่างการสลับการติดต่อกับหน่วยความจำเก็บข้อมูลทั้งสองชุด ดังรูป 3.17

งการสลับการติดต่อกับหน่วยความจำเก็บข้อมูลทั้งสองชุด ดังรูป 3.17

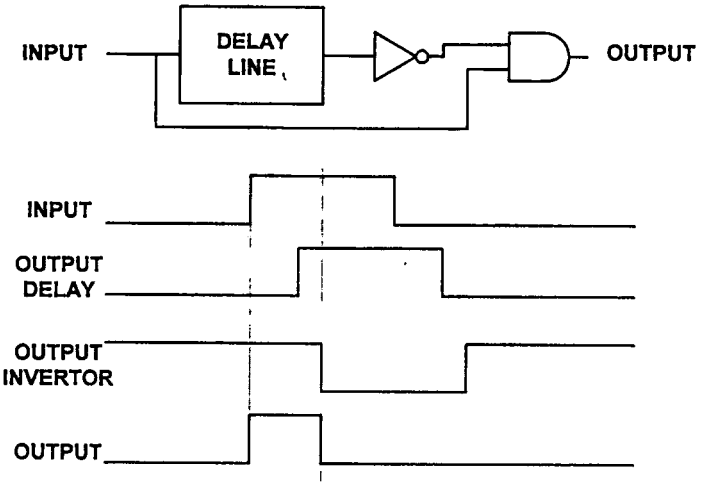


รูปที่ 3.17 วงจรส่วนหน่วงสัญญาณเพื่อรอการเปลี่ยนตำแหน่งหน่วยความจำ

สัญญาณส่วนหนึ่งจะผ่านเข้าวงจรกำเนิดพัลส์ขนาดเล็กมากดังรูป 3.18 ซึ่งทำหน้าที่สร้างสัญญาณ พัลส์ขนาดเล็กมากสำหรับการอ่านเขียนหน่วยความจำ สาเหตุที่ต้องมีวงจรนี้เนื่องจากในระบบเก็บข้อมูลนี้ได้ออกแบบให้ทำงานแบบมีการค้างสัญญาณที่จำเป็นสำหรับการอ่านเขียนหน่วยความจำไว้ ซึ่งประกอบด้วยตำแหน่งหน่วยความจำ ข้อมูล(กรณีเขียนเข้าหน่วยความจำ) และสัญญาณควบคุมต่างๆเช่นสัญญาณ chip select สัญญาณ write เป็นต้น ปัญหาที่จะเกิดขึ้นคือเมื่อระบบเก็บข้อมูลมีการเปลี่ยนแปลงค่าตำแหน่งหน่วยความจำเป็นตำแหน่งใหม่ ในขณะที่กำลังเปลี่ยนนั้นถ้าสัญญาณ chip select และสัญญาณ write ยังคงค้างสภาวะทำงานอยู่อาจจะทำให้เกิดการเขียนข้อมูลลงในตำแหน่งหน่วยความจำที่ไม่ต้องการได้ นอกจากนั้นถ้าสัญญาณ write เปลี่ยนจากลอจิกทำงานเป็นลอจิกไม่ทำงานก็จะทำให้หน่วยความจำอยู่ในสภาวะอ่านข้อมูล ซึ่งทำให้เกิดการชนกันของข้อมูลที่ค้างเอาไว้สำหรับเขียนลงหน่วยความจำตำแหน่งใหม่ ดังนั้นที่จุดนี้การทำงานที่ดีที่สุดคือการเปลี่ยนให้สัญญาณ chip select ไม่ทำงานชั่วขณะที่กำลังเปลี่ยนแปลงตำแหน่งหน่วยความจำ จากการพิจารณาผังเวลาในการเขียนหน่วยความจำของระบบจะพบว่ามีเวลาในการอ่านเขียนหน่วยความจำเพียง 50 นาโนวินาที ที่อัตราการเก็บข้อมูลเป็น 20 เมกกะเฮิร์ท ดังนั้นการเปลี่ยนของสัญญาณสภาวะของสัญญาณ chip select จากสภาวะทำงานเป็นไม่ทำงานชั่วขณะจะต้องน้อยที่สุดเท่าที่เป็นไปได้เพื่อให้มีเวลามากพอในการอ่านเขียนหน่วยความจำ แต่สิ่งที่ต้องคำนึงถึงคือจะต้องให้ขนาดของสัญญาณพัลส์มีค่ากว้างกว่าเวลาที่วงจรค้างตำแหน่งหน่วยความจำใช้ในการเปลี่ยนเอาท์พุท จากการพิจารณาคู่มือพบว่า 74F374 จะให้เอาท์พุทนับตั้งแต่ได้สัญญาณอินพุทภายในเวลาไม่เกิน 10 นาโนวินาที เพื่อป้องกันปัญหาจากอุปกรณ์คุณภาพต่ำกว่าที่ระบุในหนังสือคู่มือ จึงเลือกสร้างพัลส์ขนาดประมาณ 15-20 นาโนวินาที ดังนั้นถ้าระบบเก็บข้อมูลด้วยความเร็ว 20 เมกกะเฮิร์ท คือมีคาบเวลาในการเก็บข้อมูลเป็น 50 นาโนวินาที และใช้เทคนิค bank switching ซึ่งเป็นการสลับชุดหน่วยความจำที่ติดต่อด้วย จะทำให้เพิ่มเวลาในการอ่านเขียนหน่วยความจำเป็นสองเท่าคือ 100 นาโนวินาที เมื่อหักลบจากเวลาที่ใช้ในการเปลี่ยนตำแหน่งหน่วยความจำแล้วจะได้ว่ามีเวลาในการติดต่อหน่วยความจำเป็น 80 นาโนวินาที ซึ่งมากพอสำหรับหน่วยความจำที่มีขายอยู่ทั่วไป

ในการติดต่อหน่วยความจำเป็น 80 นาโนวินาที ซึ่งมากพอสำหรับหน่วยความจำที่มีขายอยู่ทั่วไป

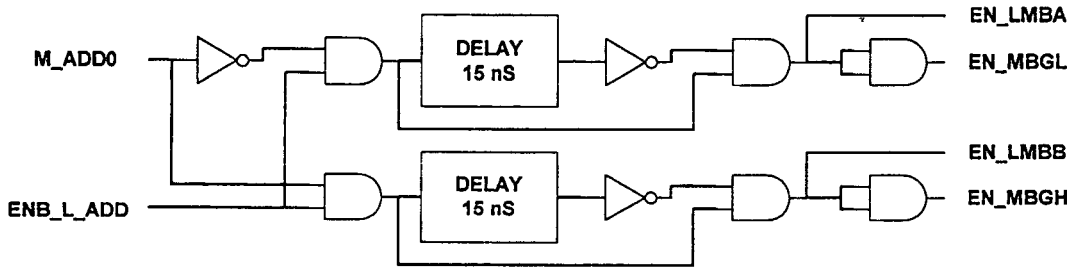
เนื่องจากสัญญาณพัลส์ที่สร้างขึ้นนี้มีระดับเป็นนาโนวินาทีซึ่งมีขนาดแคบมาก จึงนำอุปกรณ์โมโนสเตเบิลทั่วไปมาใช้ไม่ได้จึงทำการออกแบบขึ้นเองดังรูป 3.18



รูป 3.18 วงจรกำเนิดสัญญาณพัลส์ขนาดเล็กมากและผังเวลา

จากรูป 3.18 เมื่อสัญญาณที่เข้ามามีการเปลี่ยนแปลงจากลอจิกต่ำเป็นลอจิกสูงสัญญาณส่วนหนึ่งจะไปรอที่อินพุทของ and gate อีกส่วนหนึ่งจะถูกหน่วงไปด้วย delay line ไป 15 นาโนวินาทีแล้วถูกกลับลอจิกเข้าอินพุทของ and gate อีกข้างหนึ่ง ในช่วงเวลาที่หน่วงไปนี้เอาท์พุทของ and gate จะเปลี่ยนไปเป็นลอจิกสูงตามอินพุท เมื่อเอาท์พุทของอินเวอร์เตอร์เปลี่ยนตามอินพุทแล้วจะบังคับให้เอาท์พุทของ and gate กลายเป็นลอจิกต่ำเช่นเดิมทำให้เกิดเป็นพัลส์แคบๆ โดยที่ความกว้างของพัสส์นี้มีค่าเท่ากับเวลาที่ถูกหน่วงไปโดย delay line และเวลาที่อินเวอร์เตอร์ให้เอาท์พุทออกมาซึ่งในระบบได้กำหนดให้หน่วงเวลาไป 15 นาโนวินาทีและใช้อินเวอร์เตอร์ 74F04 ซึ่งให้เอาท์พุทภายในเวลาประมาณ 4-6 นาโนวินาที ดังนั้นขนาดของพัลส์จะมีค่าประมาณ 20 นาโนวินาทีตามต้องการ

สัญญาณนาฬิกาที่ทำหน้าที่ทริกให้ 74F374 ทำการค้างข้อมูลตำแหน่งหน่วยความจำส่วนหนึ่งจะถูกส่งผ่านเข้าวงจรกำเนิดสัญญาณเพื่อควบคุมการเปลี่ยนของสัญญาณ chip select ดังรูป 3.19

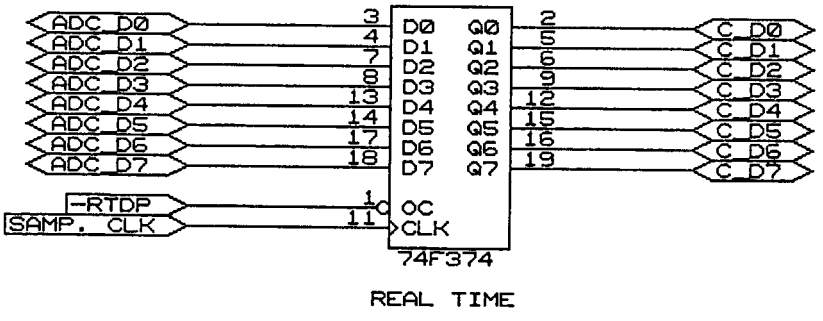


รูป 3.19 วงจรควบคุมการเปลี่ยนสัญญาณ chip select ของหน่วยความจำ

สัญญาณ M_ADD0 เป็นสัญญาณตำแหน่งหน่วยความจำบิท 0 ในที่นี้นำมาใช้เพื่อกำหนดว่าเป็นการติดต่อกับหน่วยความจำชุดคู่หรือชุดคี่ เมื่อสัญญาณเก็บข้อมูล EN_L_ADD (Enable Latch ADDress) เกิดขึ้น สัญญาณ M_ADD0 จะทำหน้าที่เลือกชุดหน่วยความจำที่จะผ่านสัญญาณ EN_L_ADD นี้ให้ ซึ่งจะถูกใช้ในการกำเนิดสัญญาณพัลส์ขนาดแคบมากให้กับวงจรเลือกหน่วยความจำ เพื่อให้มีการเปลี่ยนแปลงสัญญาณ chip select ของหน่วยความจำ จากเดิมเป็นลอจิกต่ำ(สภาวะทำงาน)เป็นลอจิกสูง(สภาวะไม่ทำงาน) ชั่วขณะที่สัญญาณ EN_L_ADD ส่วนหนึ่งทำหน้าที่ทริกให้เกิดการเปลี่ยนตำแหน่งหน่วยความจำเป็นตำแหน่งใหม่

3.3.5 วงจรส่วนเก็บข้อมูลตามเวลาจริง

วงจรส่วนนี้ประกอบด้วย 74F374 ทำหน้าที่เป็นวงจรค้างข้อมูลทุกครั้งที่วงจรส่วนแปลงข้อมูลอนาลอกเป็นดิจิตอลแปลงข้อมูลเรียบร้อย เมื่อใดที่ระบบคอมพิวเตอร์ต้องการเก็บข้อมูลตามเวลาจริง ก็สามารถอ่านค่าได้ทันที โดยการอ่านจากพอร์ทตำแหน่ง 04 จะเห็นได้ว่าประสิทธิภาพในการเก็บข้อมูลตามเวลาจริงนี้ขึ้นอยู่กับความเร็วในการอ่านข้อมูลไปจากพอร์ทเท่านั้นคือขึ้นกับเครื่องคอมพิวเตอร์ที่นำมาต่อนั่นเอง จากการทดสอบโดยใช้เครื่องคอมพิวเตอร์คล้ายไอบีเอ็ม รุ่น 80386SX ทำงานที่สัญญาณนาฬิกา 25 MHz และ รุ่น 80286 ทำงานที่สัญญาณนาฬิกา 12 MHz พบว่าอัตราเร็วสูงสุดที่อ่านข้อมูลได้คือ ประมาณ 3 และ 2 MHz ตามลำดับ ลักษณะวงจรเป็นดังรูป 3.20

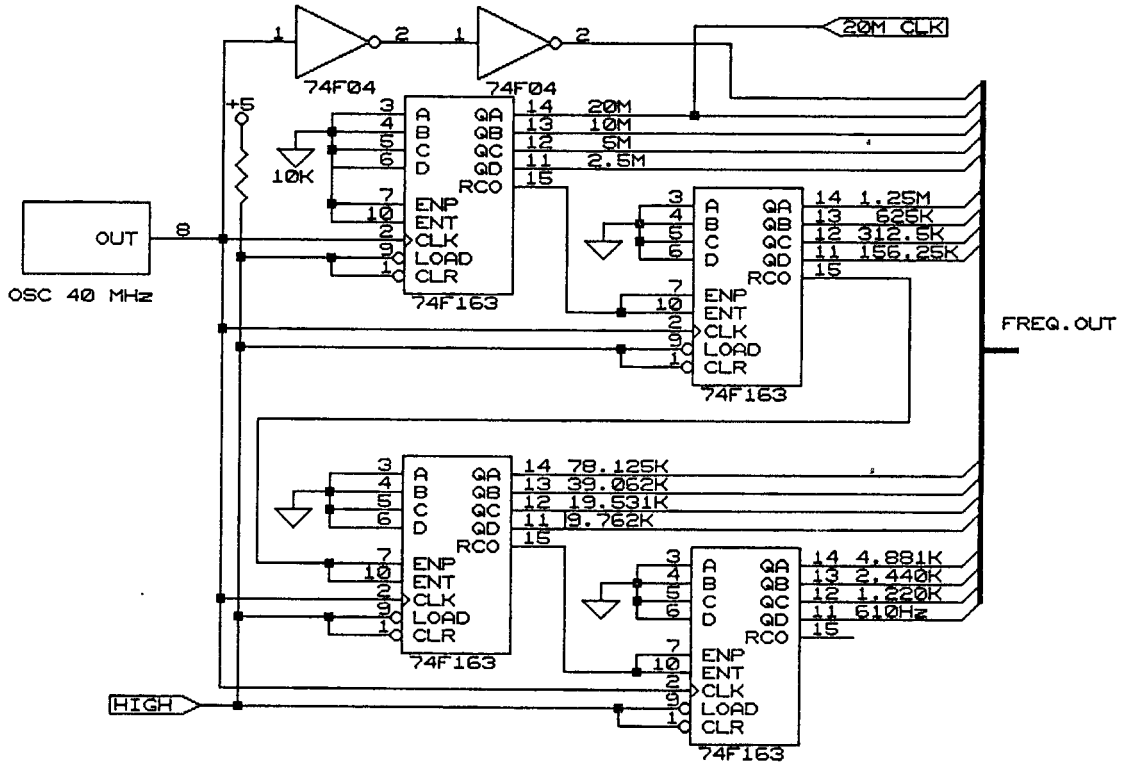


รูป 3.20 วงจรเก็บข้อมูลตามเวลาจริง

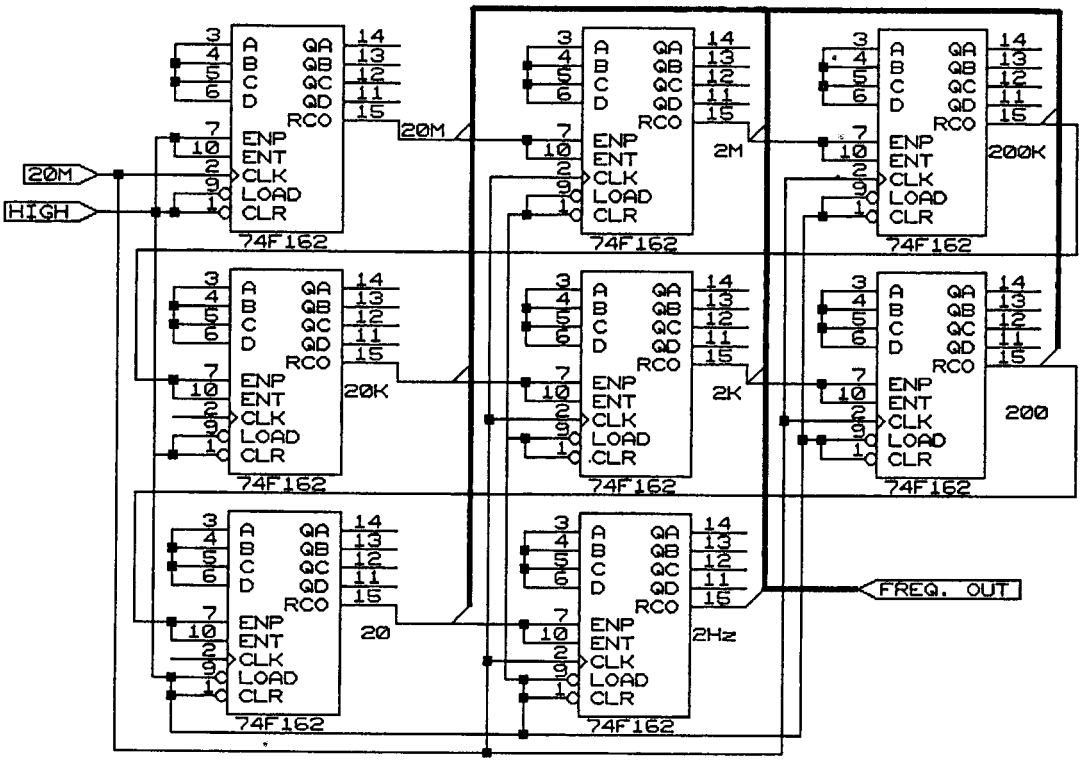
3.3.6 วงจรกำเนิดสัญญาณนาฬิกา

เพื่อให้สามารถรองรับกับการใช้งานได้กว้างขึ้นจึงได้ออกแบบให้ระบบมีสัญญาณนาฬิกาพื้นฐานเพื่อการเก็บข้อมูลทั้งหมด 24 ค่า โดยที่ผู้ใช้สามารถเลือกค่าที่ต้องการได้ตามความเหมาะสม ประกอบด้วยค่าความถี่ 20 MHz, 10 MHz, 5MHz,.....,610Hz และ 20MHz, 2MHz, 200KHz,.....,2Hz วงจรประกอบด้วย oscillator กำเนิดสัญญาณนาฬิกา 40 MHz ผ่านเข้า 74F04 เพื่อขยายกระแสขับ ส่วนหนึ่งผ่านเข้าวงจรนับแบบ binary ประกอบด้วย 74F163 จำนวน 4 ตัวได้เอาท์พุททั้งหมด 16 บิท ซึ่งแต่ละบิทจะเป็นค่าความถี่ที่ถูกหารด้วย 2 ตามลำดับ ดังรูป 3.21 สัญญาณอีกส่วนหนึ่งถูกป้อนให้กับ 74F162 ซึ่งเป็นวงจรนับ 10 โดยที่ค่าที่ได้จากเอาท์พุท

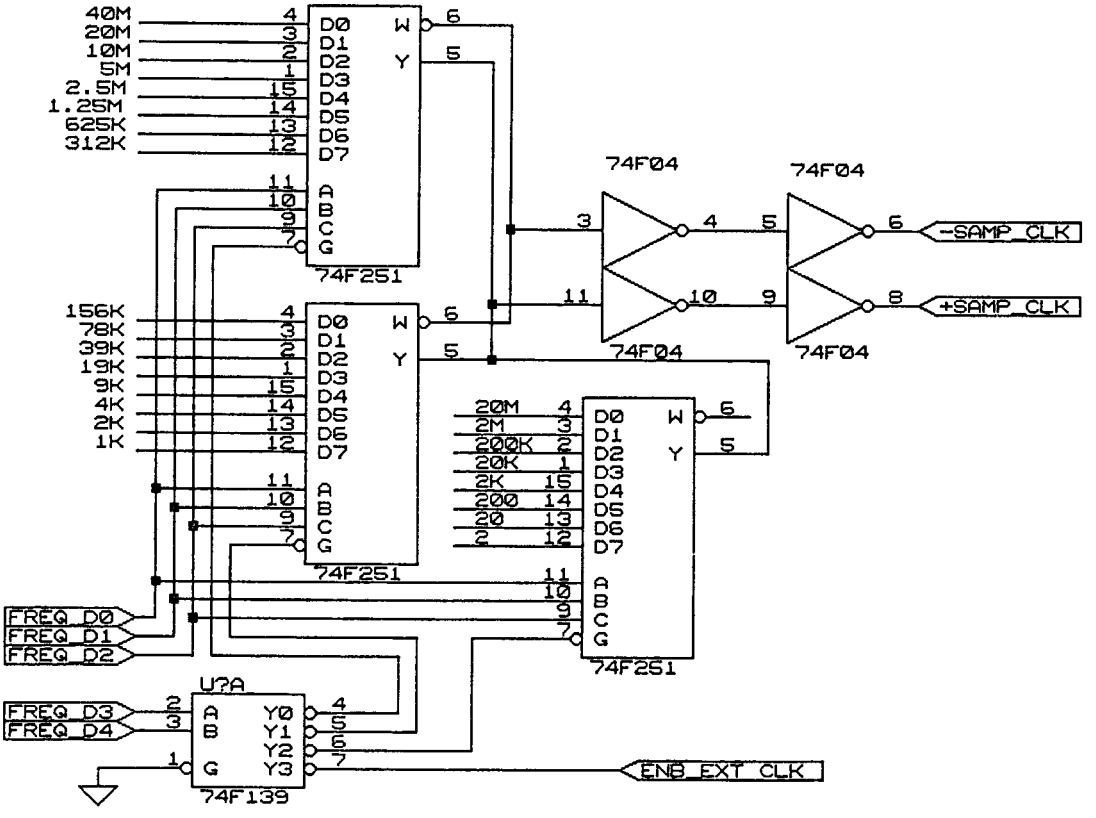
ท QA ของทุกตัวจะเป็นหาร 10 ทั้งหมด จะได้สัญญาณนาฬิกา 20 MHz, 2MHz, 200K,....,2Hz ดังรูป 3.22 สัญญาณทั้ง 24 ค่านี้จะถูกต่อเข้ากับ 74F251 ทำหน้าที่เลือกให้เฉพาะค่าความถี่ต้องการผ่านออกไปจ่ายให้กับวงจรนับกำหนดค่าตำแหน่งหน่วยความจำ โดยกำหนดรหัสค่าความถี่ผ่าน 8255 พอร์ท B แล้วสั่งทริกให้ค้างค่าโดยใช้พอร์ท A บิท 6 ซึ่งบิทที่ 3 และ 4 ของค่ารหัสความถี่นี้จะถูก decode โดย 74F139 เพื่อเลือกกลุ่ม และใช้บิท 2-0 ทำหน้าที่เลือกความถี่ที่ต้องการในกลุ่มนั้น ดังรูป 3.23



รูป 3.21 วงจรกำเนิดสัญญาณนาฬิกาส่วนหารแบบไบนารี่



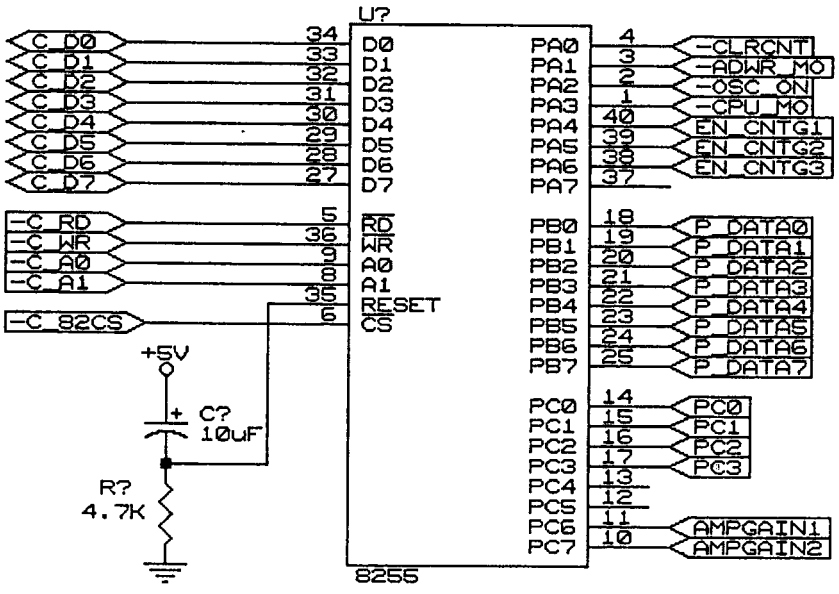
รูป 3.22 วงจรกำเนิดสัญญาณนาฬิกาส่วนหารสิบ



รูป 3.23 วงจรเลือกความถี่

3.3.7 วงจรกำหนดสภาวะการทำงาน

เป็นวงจรที่ใช้ควบคุมการทำงานตามสภาวะต่างๆ ของระบบเก็บข้อมูล โดยใช้ 8255 ต่อเข้ากับสัญญาณ C_D0 ถึง C_D7 ซึ่งเป็น data bus โดยมีสัญญาณควบคุมเป็น -C_RD , -C_WR , -C_A0 , : -C_A1 และ -C_8255 ลักษณะของวงจรเป็นดังรูป 3.24

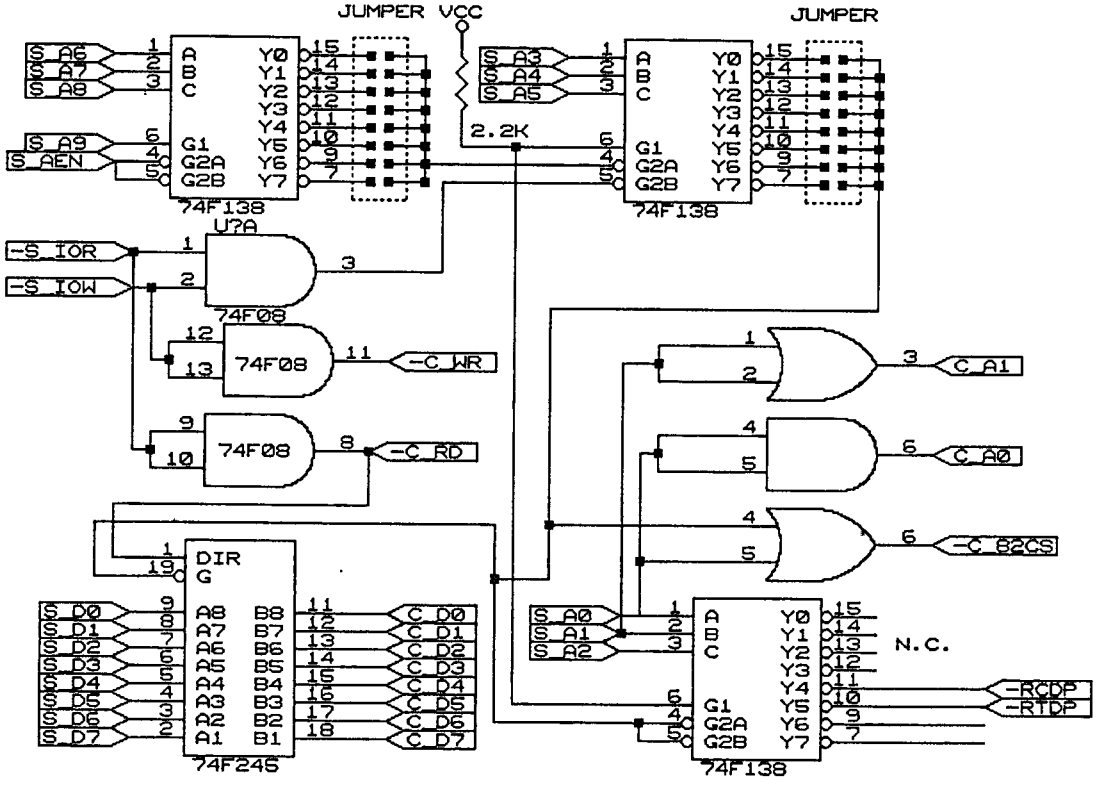


รูป 3.24 วงจรกำหนดสภาวะการทำงาน

3.4 ส่วนเชื่อมต่อกับระบบคอมพิวเตอร์

เนื่องจากต้องการให้ระบบเก็บข้อมูลที่สร้างขึ้น สามารถนำไปใช้งานกับระบบคอมพิวเตอร์ อื่นๆนอกเหนือจากระบบไมโครคอมพิวเตอร์ไอบีเอ็มหรือคล้ายกัน และจากข้อกำหนดในการออกแบบระบบซอฟท์แวร์ที่ต้องการพัฒนาบนระบบปฏิบัติการไมโครซอฟท์วินโดว์ส ซึ่งกำหนดให้โปรแกรมที่พัฒนาภายใต้ระบบปฏิบัติการไมโครซอฟท์วินโดว์สทำการติดต่อกับฮาร์ดแวร์ได้เฉพาะผ่านทางพอร์ท I/O เท่านั้น จึงกำหนดให้ระบบคอมพิวเตอร์ติดต่อกับระบบเก็บและวิเคราะห์ข้อมูลความเร็วสูงโดยผ่านพอร์ท I/O และเพื่อเป็นการลดจำนวนพอร์ทที่ต้องใช้ในการติดต่อกับระบบ จึงใช้วิธีการ multiplex โดยกำหนดให้พอร์ท B ของ 8255 เป็นพอร์ทข้อมูลและพอร์ท C เป็นพอร์ทสั่งโปรแกรมระบบฮาร์ดแวร์ โดยนำเอาสัญญาณตำแหน่งหน่วยความจำ 10 บิทล่าง (A0-A9) ผ่าน 74F138 จำนวนสามตัว โดย 74F138 ตัวแรกทำหน้าที่กำหนดช่วงของพอร์ตจากทั้งหมด 512 พอร์ท (200H-3FFH) แบ่งเป็น 8 ช่วง ช่วงละ 64 พอร์ท โดยนำเอาสัญญาณ A9 มาต่อที่ขา Enable 2 เพื่อกำหนดให้ 74F138 ตัวแรกทำงานเฉพาะในช่วงพอร์ท 200H ขึ้นไปเท่านั้น เนื่องจากต้องการติดต่อเฉพาะผ่านพอร์ท I/O เท่านั้นจึงนำสัญญาณ -IOR และ -IOW จากสล๊อทมาผ่าน and gate เพื่อให้มีสัญญาณเป็น ลอจิกต่ำ ทุกครั้งที่มีการส่งสัญญาณติดต่อกับ I/O ไม่ว่าจะเป็นการอ่านหรือเขียนจากไอบีเอ็มพีซีก็ตาม และต่อสัญญาณตำแหน่งหน่วยความจำ A6-A8 เข้ากับขา A B C ตามลำดับ ดังนั้นพอร์ทที่เลือกได้จาก 74F138 ตัวแรกจะเป็น 200H-23FH, 240H-27FH,..,3C0H-3FFH ในทำนองเดียวกัน 74F138 ตัวสองทำหน้าที่แบ่ง 64 พอร์ทที่เลือกแล้วมาแบ่งเป็น 8 ช่วงๆละ 8 พอร์ท โดยใช้สัญญาณตำแหน่งหน่วยความจำ A5-A3 ทำให้ได้พอร์ทย่อยออกเป็น 00H-07H,08H-0FH,...,38H-3FH

เลือกแล้วมาแบ่งเป็น 8 ช่วงๆละ 8 พอร์ท โดยใช้สัญญาณตำแหน่งหน่วยความจำ A5-A3 ทำให้ได้พอร์ทย่อยออกเป็น 00H-07H,08H-0FH,...,38H-3FH และ 74F138 ตัวสุดท้ายทำหน้าแบ่งให้เลือกเป็นแต่ละตำแหน่งหน่วยความจำ เนื่องจาก 8255 มีการใช้งานพอร์ทรวม 4 พอร์ท ดังนั้นจึงนำเอาสัญญาณตำแหน่งหน่วยความจำ A2 ซึ่งกำหนดช่วงของตำแหน่งหน่วยความจำได้ระดับกลุ่มละ 4 บิทมาผ่าน or gate ร่วมกับสัญญาณเลือกกลุ่มตำแหน่งหน่วยความจำที่ได้จาก 74F138 ตัวที่สอง แล้วจึงนำไปใช้เป็นสัญญาณ CS ให้กับ 8255 ต่อไป (รายละเอียดของพอร์ทต่างๆที่ใช้ในการติดต่อจะกล่าวถึงโดยละเอียดในบทที่ 4 ต่อไป) และใช้ 74F245 เป็นบัฟเฟอร์ในการติดต่อกับ data bus ของระบบคอมพิวเตอร์ โดยนำเอาสัญญาณ -IOR และ -IOW จากสล๊อท วงจรเป็นดังรูป 3.25



รูป 3.25 วงจรส่วนเชื่อมต่อกับระบบคอมพิวเตอร์แบบ IBM PC

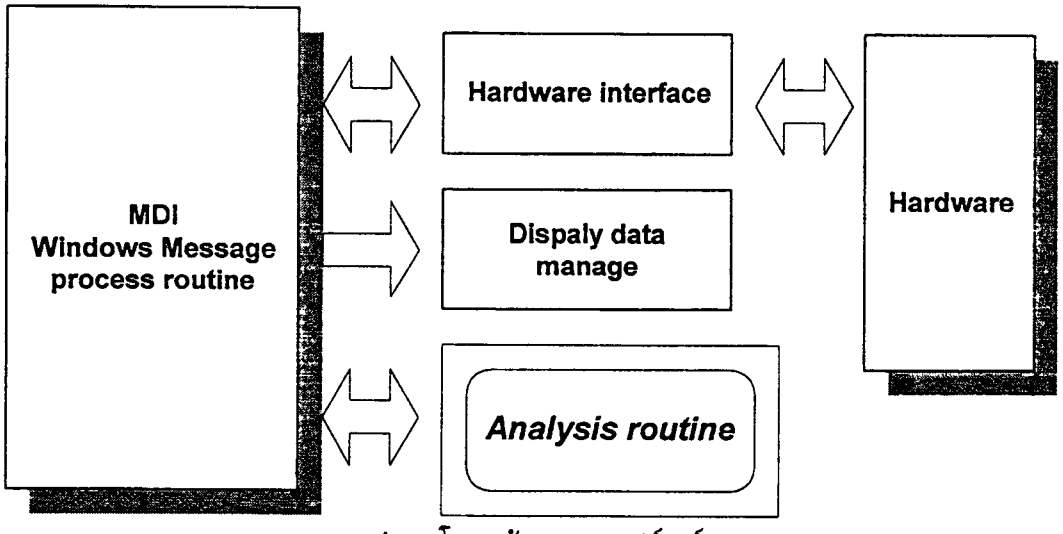
บทที่ 4
การออกแบบระบบส่วนซอฟท์แวร์

ในการพัฒนาระบบเก็บข้อมูลนี้โดยหลักแล้วต้องการเน้นการพัฒนาทางด้านฮาร์ดแวร์ ดังนั้นในส่วนของซอฟท์แวร์จะเน้นในเฉพาะส่วนของการเชื่อมโยงข้อมูลกับระบบฮาร์ดแวร์ การนำเสนอผล ส่วนทางด้านการวิเคราะห์ข้อมูลนั้น ผู้ใช้หรือผู้พัฒนาระบบต่อไปสามารถเพิ่มเติมได้ตามความต้องการเนื่องจากได้ออกแบบระบบรองรับเอาไว้แล้ว

4.1 โครงสร้างของระบบซอฟท์แวร์

เนื่องจากระบบเก็บข้อมูลได้รับการออกแบบให้เชื่อมโยงกับระบบคอมพิวเตอร์ โดยให้เครื่องคอมพิวเตอร์ทำหน้าที่วิเคราะห์ข้อมูล(ในวิทยานิพนธ์นี้ได้ทำการทดสอบโดยเชื่อมต่อกับระบบคอมพิวเตอร์แบบ IBM PC) โดยผ่านระบบพอร์ท Input/Output (I/O) แทนการใช้วิธีการอ้างหน่วยความจำของระบบเก็บข้อมูลเป็นหน่วยความจำของระบบคอมพิวเตอร์ที่นำมาเชื่อมต่อโดยตรงด้วยเหตุผลที่ต้องการให้ระบบฮาร์ดแวร์สามารถนำไปประยุกต์ใช้งานกับคอมพิวเตอร์ได้หลายแบบโดยไม่จำกัดเฉพาะแบบใดแบบหนึ่ง โดยการปรับปรุงนั้นขึ้นอยู่กับการออกแบบส่วนเชื่อมต่อกับระบบคอมพิวเตอร์ใหม่เท่านั้น สำหรับการออกแบบซอฟท์แวร์เพื่อการทดสอบระบบเก็บข้อมูลนั้นได้ทำการออกแบบบนระบบปฏิบัติการไมโครซอฟท์วินโดว์ส ซึ่งมีข้อจำกัดว่าโปรแกรมใดที่ทำการพัฒนาบนระบบปฏิบัติการนี้จะต้องไม่มีการอ้างอิงถึงตำแหน่งหน่วยความจำโดยตรงต้องเรียกหรือขอใช้หน่วยความจำโดยผ่านระบบปฏิบัติการเท่านั้นจึงเป็นเหตุผลเสริมในการออกแบบให้การติดต่อหน่วยความจำเก็บข้อมูล เป็นแบบ I/O โครงสร้างของระบบซอฟท์แวร์เป็นดังรูป 4.1

ในการทำงานของส่วนซอฟท์แวร์จะใช้วิธีการจัดการวินโดว์สแบบ MDI (Multi Document Interface) โดยที่ระบบจะทำการสร้างวินโดว์สของโปรแกรมแล้วทำการสร้างวินโดวส์เพื่อรองรับการจัดการวินโดว์สย่อย โดยเรียกวินโดว์นี้ว่า Client window หรือวินโดว์หลัก และเรียกวินโดว์ย่อยเป็น Child windows หรือบางครั้งเรียกว่า วินโดว์เอกสาร สำหรับในระบบที่ออกแบบขึ้นนี้จะใช้ Child window ทำหน้าที่เป็นส่วนแสดงภาพ



รูป 4.1 โครงสร้างระบบซอฟท์แวร์

4.2 องค์ประกอบของ MDI

ระบบ MDI เป็นวิธีจัดการโปรแกรมประยุกต์บน Microsoft Windows วิธีจัดการนี้จะอธิบายถึงโครงสร้างของวินโดว์สและการติดต่อกับผู้ใช้ที่อนุญาติให้ผู้ใช้ใช้งานเอกสารหลายๆเอกสารพร้อมกันในโปรแกรมประยุกต์เดียวกัน ระบบ MDI นี้ ถูกนำมาใช้ครั้งแรกในโปรแกรม Microsoft Excel บน Windows รวมทั้ง Program Manager และ File Manager ของ Windows 3 ก็เป็นระบบ MDI ด้วยเหมือนกัน ถึงแม้ว่าวิธีจัดการ MDI นี้จะมีมาตั้งแต่ Windows 2 แต่สมัยนั้นยังเขียนได้ยากอยู่ ต่อมาใน Windows 3 มีเครื่องมือใหม่ช่วยในการพัฒนาให้ง่ายขึ้นคือ window class 1 class, function 4 functions, data structure 2 ชุด และ message 11 messages

วินโดว์สหลักของโปรแกรมประยุกต์ ที่มีระบบนี้ปกติแล้วจะประกอบด้วย title bar, menu, sizing border, system menu icon และ minimize/maximize icons พื้นที่ของผู้ใช้ ปกติแล้วจะเรียกว่า workspace และไม่ใช้ในการแสดงผลโดยตรงใน workspace จะมีวินโดว์สย่อยลงไปที่ใช้แสดงเอกสาร หรือไม่มีเลยก็ได้

วินโดว์สย่อยก็จะดูเหมือนกับหน้าต่างปกติของโปรแกรมประยุกต์ คือมี title bar, sizing order, system menu icon, minimize/maximize icons และอาจจมี scroll bar ด้วย แต่ไม่มี menu

ในเวลาหนึ่งจะมีเพียงเอกสารเดียวที่พร้อมทำงานอยู่ เอกสารอื่นจะถูกแสดงอยู่ข้างหลังแต่จะไม่หลุดออกไปนอกวินโดว์สของ โปรแกรมประยุกต์

- วินโดว์สเอกสารแบบ MDI จะสามารถลดขนาดลงเหลือแต่ icon ได้ และ icon นั้นก็จะแสดงอยู่ที่ด้านล่างของ workspace

- วินโดว์สเอกสารแบบ MDI จะสามารถเพิ่มขนาดได้เต็มที่ โดย title bar ของวินโดว์สนี้จะหายไป และชื่อแฟ้มจะไปปรากฏอยู่ที่ title bar ของ โปรแกรมประยุกต์ ต่อมา system menu icon ของวินโดว์สก็จะไปอยู่บน menu ของ โปรแกรมประยุกต์ ตำแหน่งแรกสุด และ icon ที่ใช้เรียกขนาดของวินโดว์สคืนมาก็จะไปอยู่ตำแหน่งขวามือสุดของ menu

- มีคำสั่งจากคีย์บอร์ดเพื่อปิดวินโดว์สเอกสารเหมือนการปิดวินโดว์สหลักคือในขณะที่ปกติจะใช้ Alt-F4 เพื่อปิดวินโดว์สของ โปรแกรมประยุกต์ ก็จะใช้ Ctrl-F4 ในการปิดวินโดว์สของเอกสาร และสามารถใช้ Ctrl-F6 เพื่อสลับไปทำงานที่วินโดว์สเอกสารอื่นที่เปิดอยู่ได้ Alt-Spacebar เพื่อเปิด system menu ของวินโดว์สหลัก และ Alt-เครื่องหมายลบ เพื่อเปิด system menu ของวินโดว์สเอกสาร

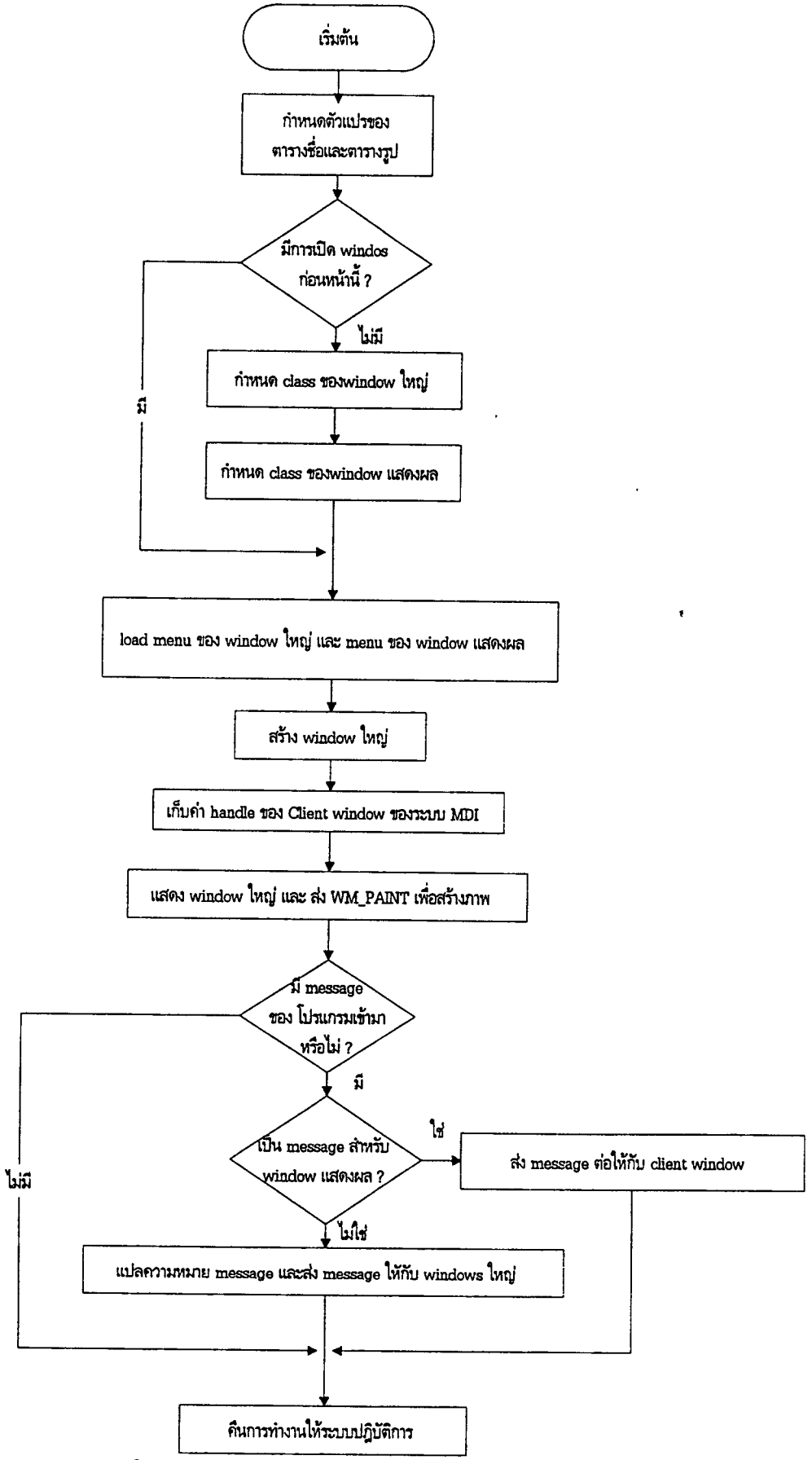
- ในการใช้ปุ่มลูกศรเลื่อนหัวข้อในเมนู ตัวเลือกจะเลื่อนจาก system menu ไปยังตัวเมนูหัวข้อแรกใน menu bar ในระบบ MDI ตัวเลือกจะเลื่อนจาก system menu ของ โปรแกรมประยุกต์ ไปยัง system menu ของ วินโดว์สที่ใช้งานอยู่ แล้วเลื่อนต่อไปยัง ตัวเมนูหัวข้อแรกใน menu bar

- ถ้า โปรแกรมประยุกต์ สามารถใช้งานวินโดว์สย่อยได้หลายแบบ เช่น วินโดว์ส worksheet และวินโดว์ส chart ใน Microsoft Excel ตัวเมนูจะต้องสามารถตอบรับการใช้งานวินโดว์สแบบนั้น คือโปรแกรมต้องเปลี่ยนไปเมื่อวินโดว์สอีกแบบถูกใช้งาน ถ้าไม่มีวินโดว์สใดใช้งานอยู่จะมีเพียงบางเมนูที่ใช้งานได้เท่านั้น

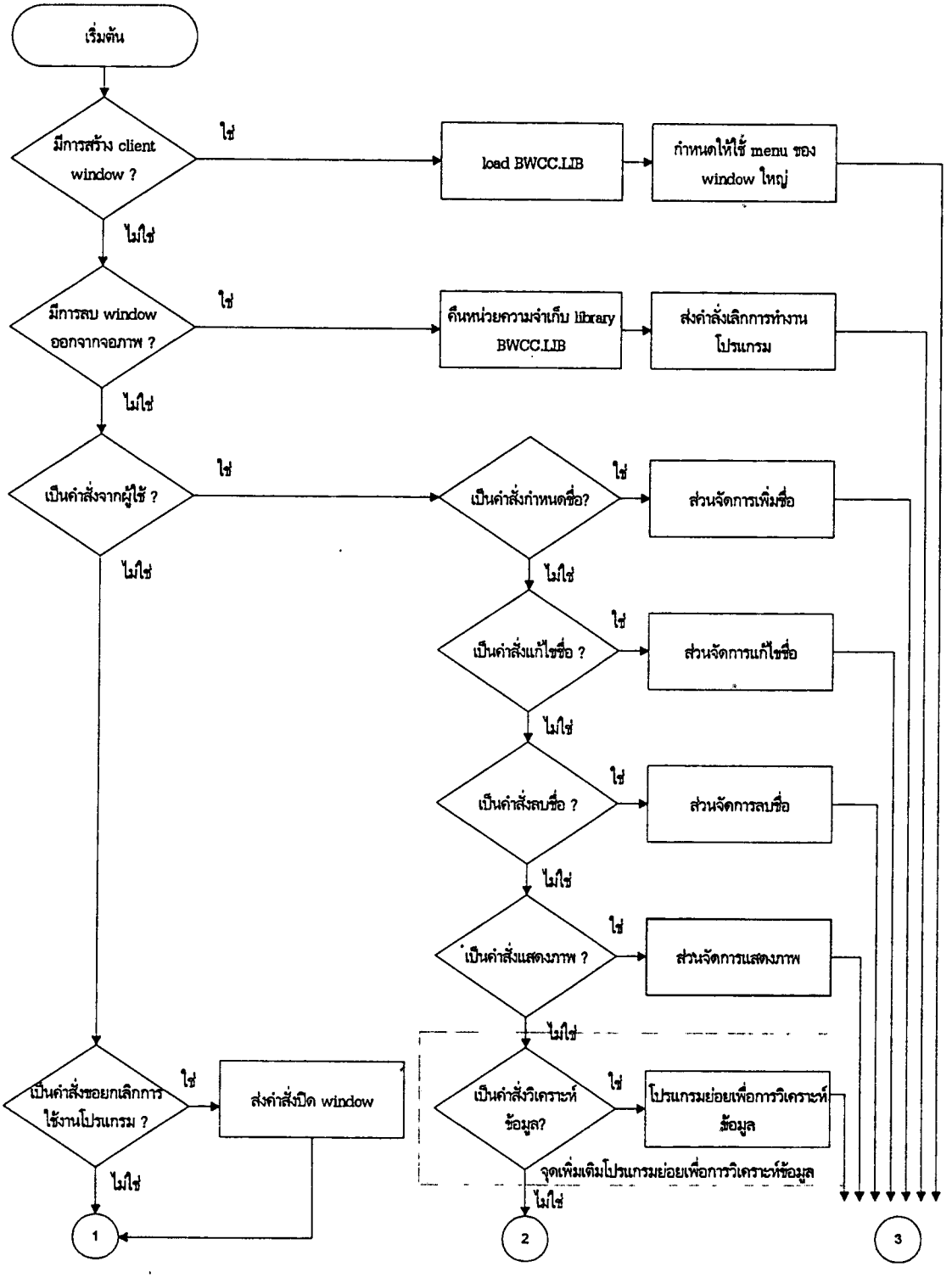
- ในเมนูของ โปรแกรมประยุกต์ จะมีหัวข้อ Window อยู่ โดยตกลงว่าจะอยู่เป็นหัวข้อสุดท้าย ก่อนหัวข้อ Help ในหัวข้อ Window จะมีตัวเลือกที่สั่งให้จัดเรียงตำแหน่งของวินโดว์สใน workspace คือสามารถ "cascaded" จากมุมซ้ายมือบน หรือ "tiled" ให้เห็นทุกวินโดว์สพร้อมกันได้ และจะมีรายชื่อของเอกสารที่เปิดใช้งานอยู่ เพื่อที่จะได้เลือกเอกสารใดเอกสารหนึ่งขึ้นมาใช้งาน

สิ่งเหล่านี้เป็นระบบ MDI บน Windows 3 บางหัวข้อเป็นสิ่งจำเป็น แต่ไม่จำเป็นต้องยึดตามอย่างตายตัว

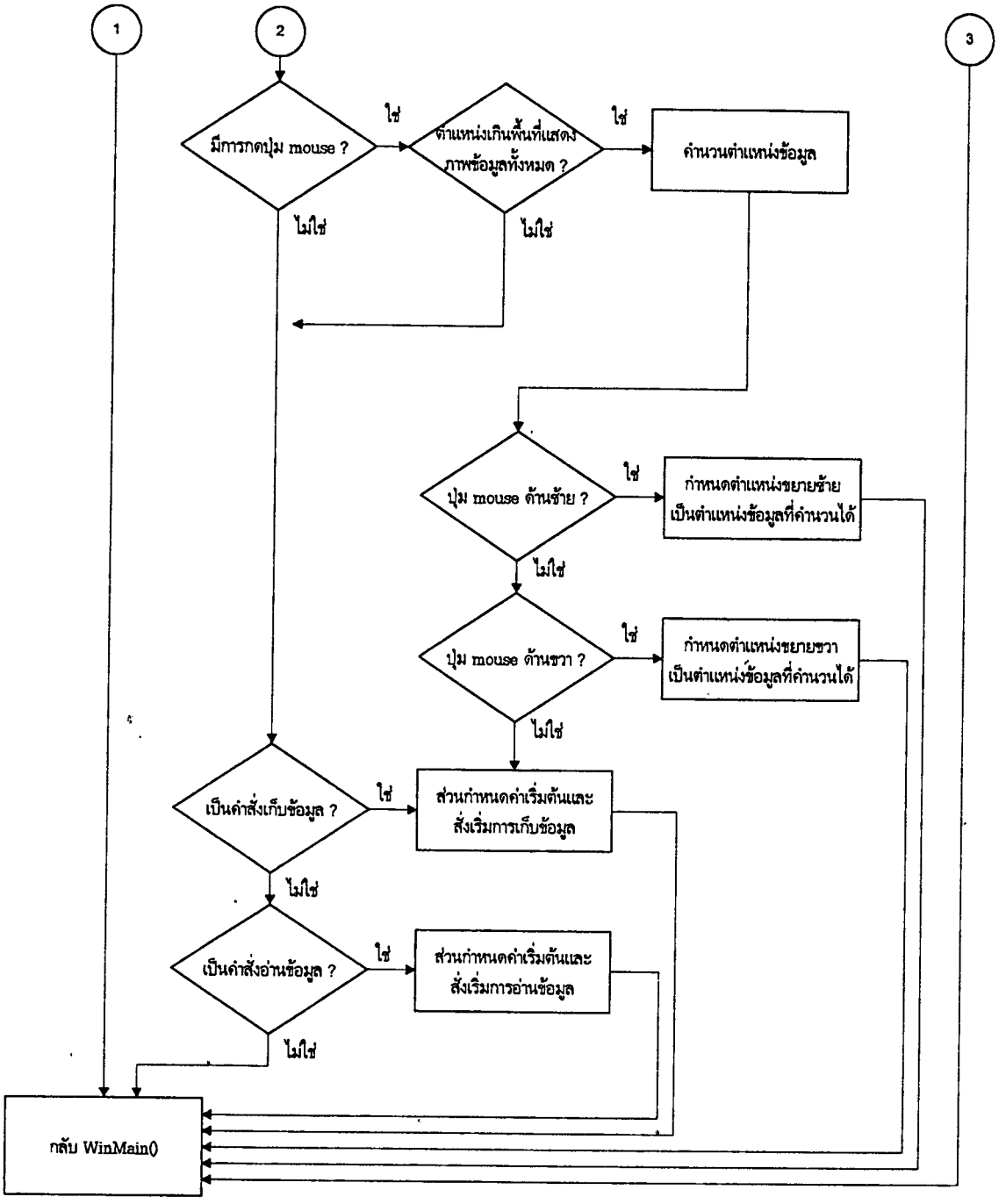
สำหรับระบบซอฟแวร์ที่ออกแบบขึ้นนี้ จะมีขั้นตอนในการทำงานเป็นดังรูป 4.2, 4.3, 4.4 และ 4.5 สำหรับโปรแกรมที่สมบูรณ์จะอยู่ในภาคผนวก ค



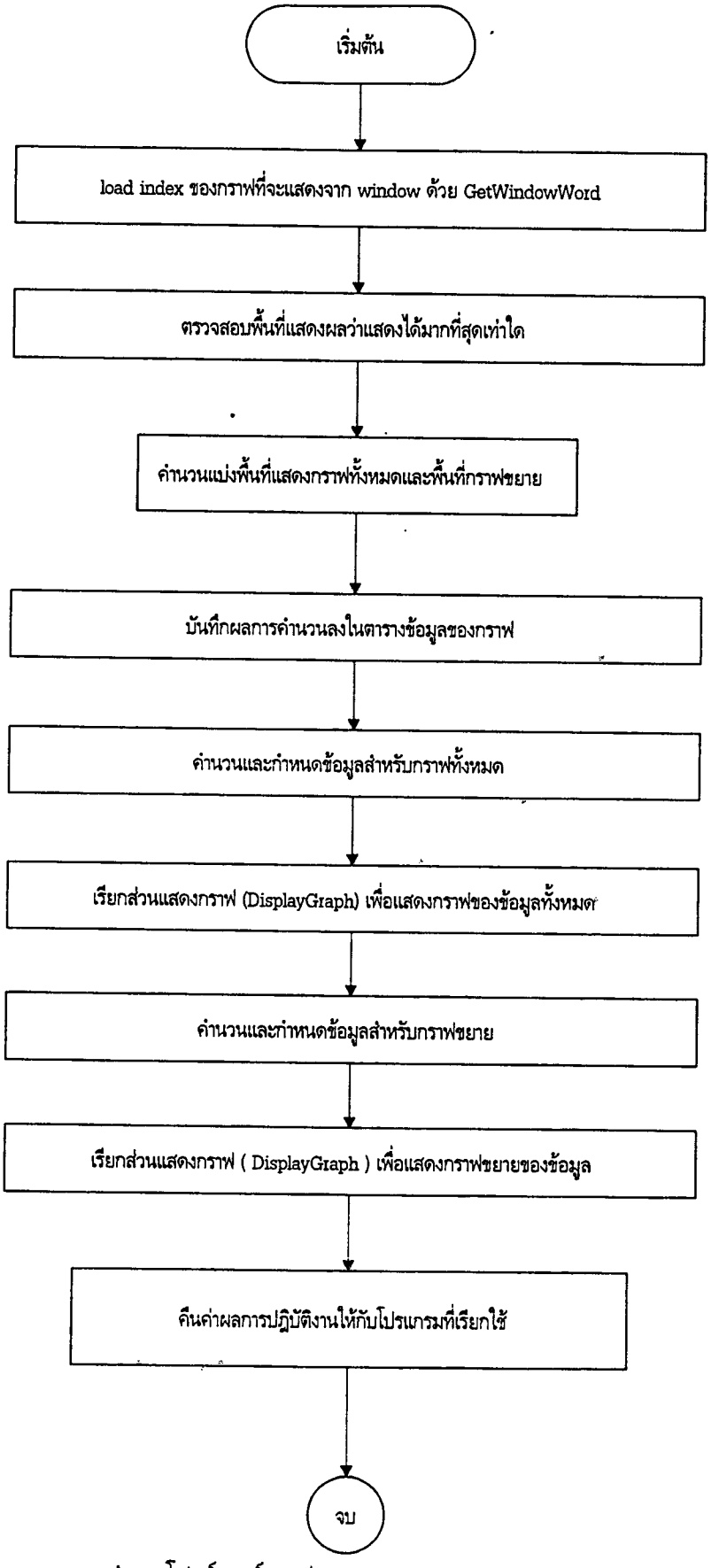
รูป 4.2 โฟลว์ชาร์ตของส่วนรับ message จากระบบปฏิบัติการไมโครซอฟท์วินโดว์ส WinMain



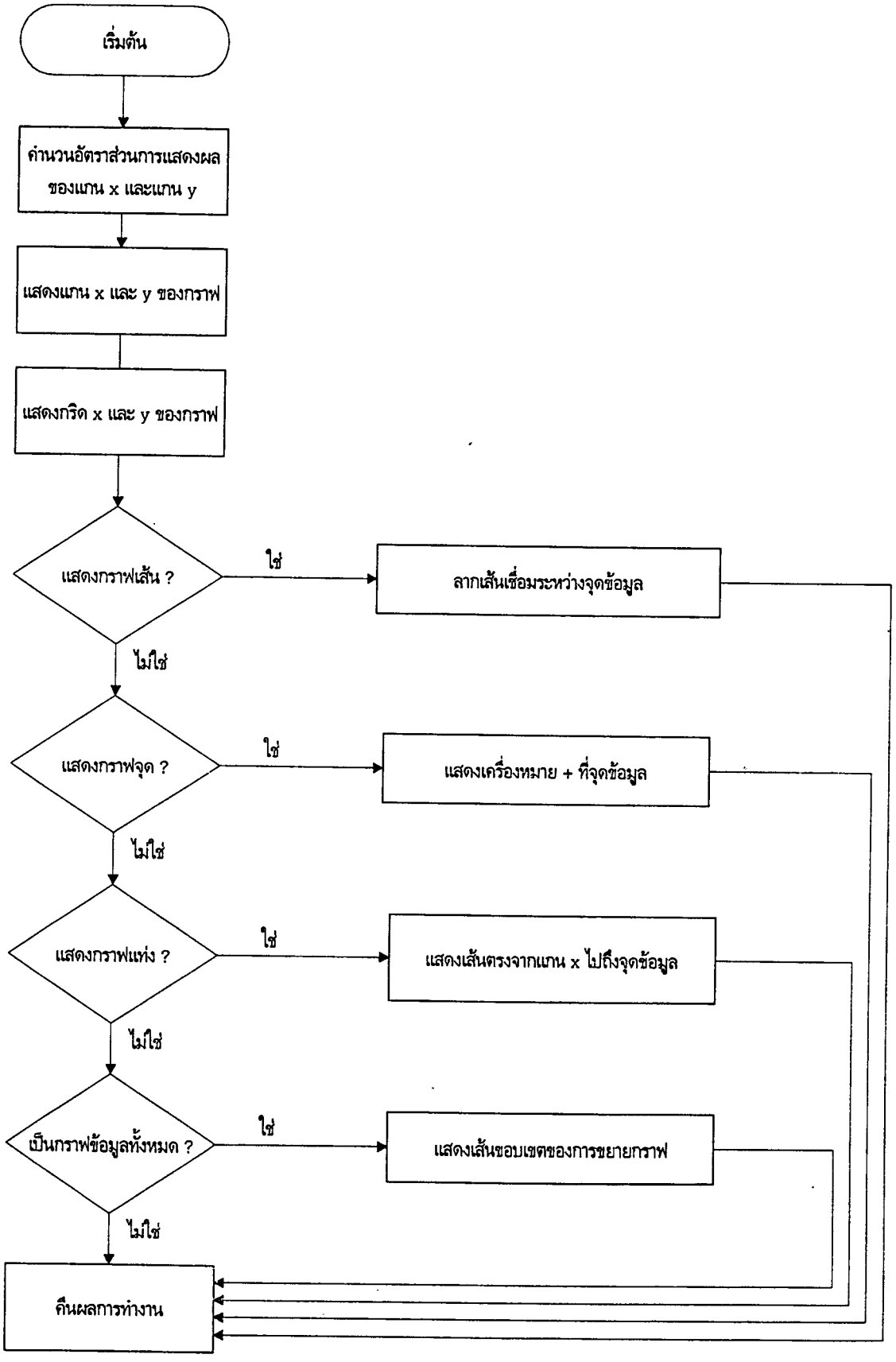
รูป 4.3 โฟลว์ชาร์ตของส่วนจัดการ message ของ Client window



รูป 4.3 (ต่อ) โฟลว์ชาร์ตของส่วนจัดการ massage ของ Client window (ต่อ)



รูป 4.4 โฟลว์ชาร์ตของส่วนแสดงกราฟ (ShowGraph)



รูป 4.5 โฟลว์ชาร์ตของส่วนจัดการกราฟ (Display Graph)

4.3 ตำแหน่งพอร์ท I/O และวิธีการสั่งงานระบบเก็บข้อมูล

ระบบเก็บข้อมูลจะใช้งานพอร์ททั้งหมด 6 พอร์ท โดยกำหนดให้เรียงลำดับกันดังนี้

พอร์ท	หน้าที่
00-03	เป็นพอร์ทสำหรับการสั่งงานและควบคุม 8255 เพื่อกำหนดหน้าที่ของส่วนต่าง ระบบเก็บข้อมูล
04	เป็นพอร์ทอ่านข้อมูลจากหน่วยความจำเก็บข้อมูล (Memory data port)
05	เป็นพอร์ทอ่านข้อมูลแบบเวลาจริง (Real time data)

4.3.1 ตำแหน่งรายละเอียดของพอร์ท

พอร์ท 00

เป็นพอร์ท A ของ 8255 ซึ่งแต่ละบิททำหน้าที่ดังนี้

บิท	ชื่อสัญญาน	หน้าที่การทำงาน
0	-CLRCNT (CLeaR CouNT)	เป็นบิททที่กำหนดให้เอาท์พุทของวงจรนับ ศูนย์(รีเซต)
1	-ADWR_MO (ADc WRite MOde)	เป็นบิทที่กำหนดให้ระบบเก็บข้อมูลทำงานในสภาว ข้อมูลเข้าหน่วยความจำ
2	-OSC ON	เป็นบิทที่กำหนดการเปิด/ปิดสัญญานนาฬิกาเก็บข้อ
3	-CPU_MO	เป็นบิทกำหนดว่าหน่วยความจำเก็บข้อมูลจะทำกา ต่อกับระบบคอมพิวเตอร์(ที่ลอจิกต่ำ)หรือติดต่ ระบบเก็บข้อมูล(ที่ลอจิกสูง)
4-5	No Connect	สำรองสำหรับการขยายระบบ
6	L_SEL_FR (Latch_SELect FRequency	เป็นบิทที่ใช้ในการทริกให้วงจร latch ทำการค้างค่า ความถี่ที่ต้องการให้กับวงจรสร้างสัญญานนาฬิกา
7	L_NUM_CH (Latch Number Channel	เป็นบิทที่ใช้ในการทริกให้วงจร latch ทำการค้างจำ ของช่องสัญญานที่ต้องการให้กับวงจรเลือกช่องสัญ น

พอร์ท 01

เป็นพอร์ท B ของ 8255 ใช้ในการกำหนดข้อมูลเพื่อการกำหนดค่าต่างๆให้กับระบบเก็บข้อมูลเช่น ค่าตำแหน่งหน่วยความจำเริ่มต้นสำหรับการเก็บข้อมูล ค่าจำนวนช่องสัญญาน ค่ารหัสของความถี่ที่ต้องการใช้งาน ค่าตำแหน่งหน่วยความจำที่หยุดเก็บข้อมูล เป็นต้น

พอร์ท 02

เป็นพอร์ท C ของ 8255 ใช้ในการกำหนดค่าตำแหน่งหน่วยความความจำเริ่มต้นและค่าตำแหน่ง หน่วยความจำที่ระบบคอมพิวเตอร์ต้องการติดต่อด้วย โดยการกำหนดให้ค่าดังต่อไปนี้

ค่า	ผลการทำงาน
0	สั่งให้ระบบเก็บข้อมูลอยู่ในสภาวะไม่รับข้อมูล เพื่อป้องกันการเปลี่ยนข้อมูลที่ทำการโปรแกร
1	กำหนดไบท์ต่ำ (บิท 0-7) ของตำแหน่งหน่วยความจำที่ระบบคอมพิวเตอร์ต้องการติดต่อด้ว
2	กำหนดไบท์กลาง (บิท 8-15) ของตำแหน่งหน่วยความจำที่ระบบคอมพิวเตอร์ต้องการติ ด้วย
3	กำหนดไบท์สูง (บิท 16-23) ของตำแหน่งหน่วยความจำที่ระบบคอมพิวเตอร์ต้องการติ ด้วย
4	กำหนดไบท์ต่ำ (บิท 0-7) ของตำแหน่งเริ่มต้นของวงจรนับกำเนิดตำแหน่งความจำ
5	กำหนดไบท์กลาง (บิท 8-15) ของตำแหน่งเริ่มต้นของวงจรนับกำเนิดตำแหน่งความจำ
6	กำหนดไบท์สูง (บิท16-23) ของตำแหน่งเริ่มต้นของวงจรนับกำเนิดตำแหน่งความจำ
7	สำรองสำหรับการขยายระบบ

4.3.2 วิธีการสั่งงานระบบ

ในที่นี้จะสมมุติว่าระบบได้ถูกกำหนดพอร์ท I/O เริ่มต้นที่ตำแหน่ง 300H จนถึง 307H

-การกำหนดให้เอาท์พุทวงจรนับเป็นศูนย์ทั้งหมด (รีเซต)

```
OUT 300H,A
```

-การกำหนดค่าเริ่มต้นของการอ่าน/เขียนข้อมูลหน่วยความจำ

ในการกำหนดค่าเริ่มต้นจะต้องแบ่งค่าตำแหน่งหน่วยความทั้ง 24 บิทออกเป็น 3 ไบท์ คือ สูง กลาง และต่ำ และกำหนดตำแหน่งที่จะโปรแกรมเป็น 3 , 2 และ 1 ตามลำดับ

```
OUT 301H,ไบท์สูง
OUT 302H, 3
OUT 301H,ไบท์กลาง
OUT 302H, 2
OUT 301H,ไบท์ต่ำ
OUT 302H, 1
OUT 302H, 00
```

-การกำหนดจุดหยุดนับของการอ่าน/เขียนข้อมูลหน่วยความจำ

ใช้วิธีการเดียวกับการการอ่าน/เขียนข้อมูลหน่วยความจำ

-การกำหนดให้ระบบเริ่มทำการเก็บข้อมูลเข้าหน่วยความจำ

ในการกำหนดให้ระบบเริ่มทำการเก็บข้อมูลจะต้องสั่งให้ผ่านสัญญานนาฬิกาเก็บข้อมูลและสัญญานเขียนให้กับระบบ ดังนั้นคือ บิท 2 และบิท 1 ของพอร์ท A 8255 ต้องเป็น ลอจิกต่ำ

OUT 300H, 09H

-การกำหนดให้ระบบทำการอ่านข้อมูลจากหน่วยความจำ

ในการกำหนดให้ระบบเริ่มทำการเก็บข้อมูลจะต้องสั่งให้ผ่านสัญญานนาฬิกาเก็บข้อมูลและสัญญานอ่านให้กับระบบ ดังนั้นคือ บิท 2 ของพอร์ท A 8255 ต้องเป็น ลอจิกต่ำ

OUT 300H,. 0BH

-การกำหนดความถี่ของการเก็บข้อมูล

OUT 301H, xx xx = ค่ารหัสของความถี่ที่ต้องการ

OUT 300H, 4FH

-การกำหนดจำนวนช่องสัญญาน

OUT 301H,xx xx = จำนวนช่องสัญญานเข้าที่ต้องการ

OUT 300H, 8FH

-การอ่าน/เขียนค่าจากหน่วยความจำเก็บข้อมูลจากระบบคอมพิวเตอร์

OUT 301H,ไบท์สูง

OUT 302H, 3

OUT 301H,ไบท์กลาง

OUT 302H, 2

OUT 301H,ไบท์ต่ำ

OUT 302H, 1

OUT 302H, 00

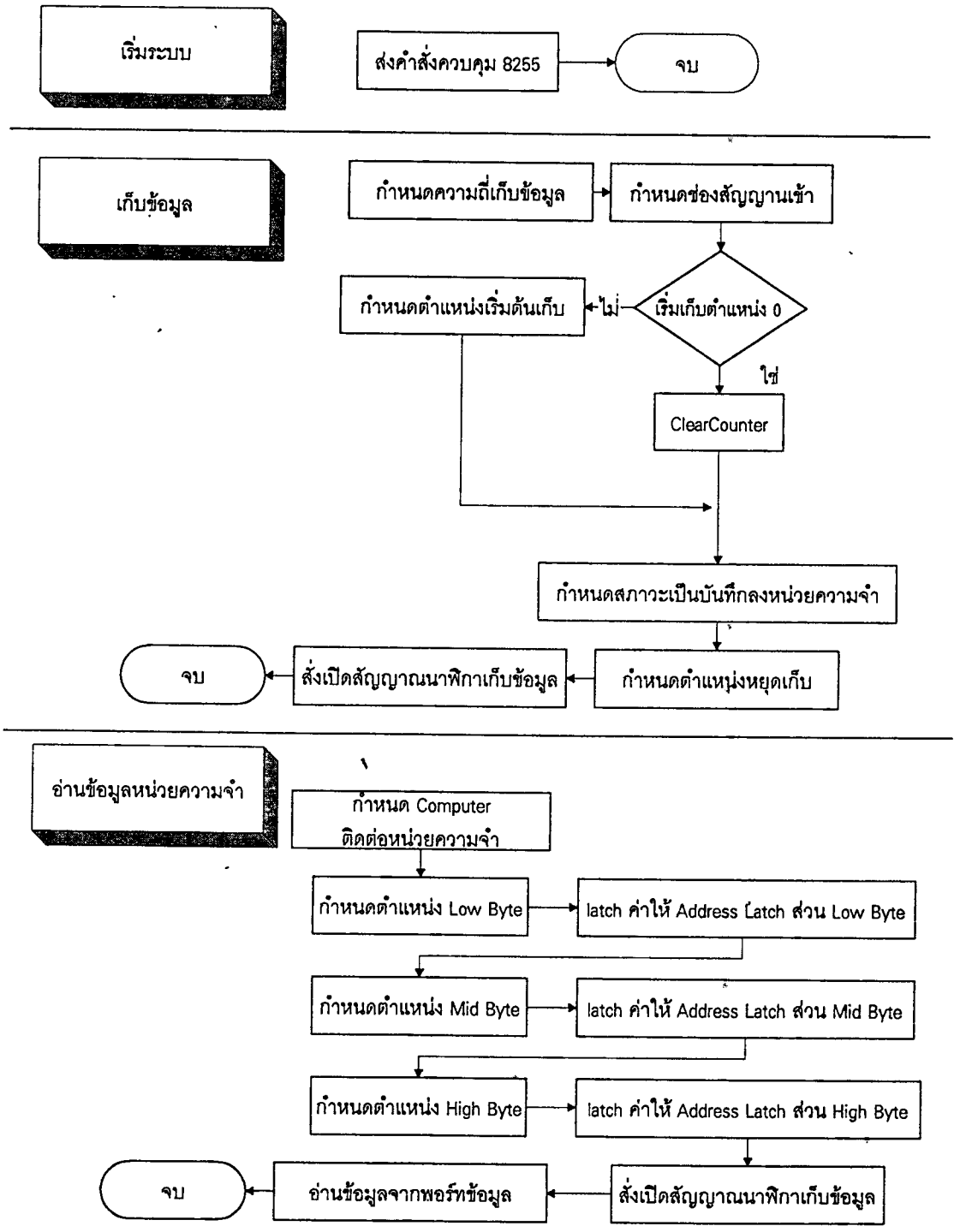
IN xx,304H หรือ OUT 304H,xx

-การอ่านค่าแบบเวลาจริง

IN 0x0305

-การรีเซทระบบ

OUT 303,80H



รูป 4.6 วิธีการควบคุมการทำงานของระบบ

4.4 โปรแกรม FFT และ IFFT ที่นำมาใช้

โปรแกรม FFT และ IFFT ที่นำมาใช้ทำเป็นโปรแกรมที่พัฒนาเขียนขึ้นโดย PAUL M.EMBREE และ BRUCE KIMBLE ซึ่งนำมาจากเอกสารอ้างอิงนี้ [13] นำมาดัดแปลงเพื่อให้ใช้กับระบบซอฟท์แวร์ที่ออกแบบขึ้นเป็นโปรแกรม FFT แบบ Radix two FFT ซึ่งจะทำงานได้เฉพาะเมื่อมีจำนวนข้อมูลให้เป็นค่า ยกกำลังของ 2 เช่น $2^3 = 8$, $2^{10} = 1024$ ดังนั้นในการใช้งานที่มีจำนวนข้อมูลน้อยกว่าค่ายกกำลังของ 2 จะต้องแทนค่าจำนวนที่เหลือด้วย 0 ทำให้บางกรณีที่โปรแกรมนี้ไม่สามารถทำงานได้ โดยเฉพาะในกรณีที่จำนวนข้อมูลไม่อยู่ในรูปของ ยกกำลังของ 2

4.4.1 วิธีทำงานของโปรแกรม

จะขออธิบายการทำงานของโปรแกรมในลักษณะของต้นฉบับ ซึ่งเขียนด้วยโครงสร้างของภาษา C บนระบบปฏิบัติการ DOS โปรแกรม FFT ที่นำมาใช้นี้เป็น Cooley-Turkey FFT จะใช้วิธีการ Decimation in time butterfly (DIT) ซึ่งใช้วิธีการแทนค่าลงข้อมูลเดิมที่เข้าทำให้ประหยัดการใช้หน่วยความจำได้มาก วิธีการทำ butterfly แบบแทนที่จะเป็นดังนี้

(1) ชี้ไปที่ตำแหน่งจุดบนของข้อมูลเข้า

(2) หาผลต่างของตำแหน่งจุดบนข้อมูลเข้าและจุดล่างข้อมูลเข้า

(3) คำนวณหาค่า W แล้วเก็บไว้

เมื่อเป็นภาษา C จะใช้ตัวแปรดังนี้

x : ตัวชี้จุดเริ่มต้นของอาเรย์ของตัวแปร COMPLEX ซึ่งใช้เก็บข้อมูล input และ output

i : ตัวชี้ตำแหน่งบนของ butterfly ใน อาเรย์ x

le : ผลต่างระหว่างค่าจุดบนและจุดล่าง

wptr : ตัวชี้ค่า W ปัจจุบัน

n : จำนวนจุดข้อมูลใน FFT

m : ค่ายกกำลังของ 2 ของจำนวนจุดข้อมูล

กำหนดให้ตัวแปร 2 ตัวเป็นและตัวแปรชั่วคราวแบบ COMPLEX คือ temp และ tm แล้วจะได้โปรแกรมเป็น

```
xi = x + i;

xip = xi + le;
```

```
temp.real = xi -> real + xip -> real;

temp.imag = xi -> imag + xip -> imag;

tm.real = xi -> real - xip -> real;

tm.imag = xi -> imag - xip -> imag;

xip -> real = tm.real * wptr -> real - tm.imag * wptr -> imag;

xip -> imag = tm.real * wptr.imag + tm.imag * wptr -> real;

*xi = temp;
```

เมื่อคำสั่งสุดท้ายจะทำการโอนค่าในตัวแปรชั่วคราวไปไว้ที่ตำแหน่งที่ถูกชี้โดย xi (ค่า output ขาบน จากการทำ butterfly) ปัจจุบันการคำนวณแบบ butterflyได้รับการพัฒนาต่อไป ดัง DIF FFT ในรูป 5.1b การ butterfly จะถูกจัดในรูป $\log_2 n$ ใน stack เรียกว่า passes แต่ละ passes butterflies จะถูกพิจารณาเป็นกลุ่มที่ใช้ค่า W ไม่เหมือนกัน ในแต่ละ pass ค่า butterfly แรก ในแต่ละกลุ่มจะมีค่า W เดียวกัน ค่าต่อมาจะเป็นคนละค่า และต่อไปเราจึงได้ประโยชน์จากโครงสร้างนี้โดยการทำ butterfly ที่มีค่า W เดียวกันในวงรอบเดียวกัน ขั้นตอนต่อไปเป็นการทำ butterflies เพื่อให้ได้ผลดังนี้

```
for  (i = j ; i < n ; i = i + 2*le)  {

    xi = x + i;

    xip = xi + le;

    temp.real = xi -> real + xip -> real;

    temp.imag = xi -> imag + xip -> imag;

    tm.real = xi -> real - xip -> real;

    tm.imag = xi -> imag - xip -> imag;

    xip -> real = tm.real * wptr -> real - tm.imag * wptr -> imag;

   xip -> imag = tm.real * wptr -> imag+ tm.imag * wptr -> real;
```

```
        *xi = temp;

    }
```

ในวงรอบนี้ ตัวชี้ไปที่ขาบน เริ่มที่ค่า j และ เพิ่มค่าระยะทางระหว่างจุดบนและจุดล่างของ butterfly ทีละสองเท่า เราสามารถตรวจสอบได้จากรูปการคำนวณ butterfly ที่ใช้ค่า W เดียวกัน ค่าต่อไปในกระบวนการก็คือ pass แต่ละ pass ถูกจัดเตรียมโดยวงรอบต่อไปนี้

```
    for  (j =0 ; j < le ; j++)  {

        for (i = j ; i < n ; i = i + 2*1e)  {

            xi = x + i;

            xip = xi + le;

            temp.real = xi -> real + xip -> real;

            temp.imag = xi -> imag + xip -> imag;

            tm.real = xi -> real - xip -> real;

            tm.imag = xi -> imag - xip -> imag;

            xip -> real = tm.real * wptr -> real - tm.imag * wptr -> imag;

            xip -> imag = tm.real * wptr -> imag + tm.imag * wptr -> real;

            *xi = temp;

        }

        wptr = wptr + windex;

    }
```

วงรอบนี้จะเพิ่มค่าของตัวชี้ขาบน ,j, ไปที่ขาบนของกลุ่มแรกใน pass วงรอบในจะกระทำการกับกลุ่มอื่นๆโดยใช้ j เป็นจุดเริ่มต้น ฟังก์ชันอื่นในวงรอบจะทำการเพิ่มค่าตัวชี้ไปที่ค่าอาเรย์ W ต่อไป (wptr) โดย windex ค่า windex ก็จะเปลี่ยนไปในแต่ละ pass

ขั้นตอนสุดท้ายในการจัดเตรียม การทำ FFT คือทำกับ m passes ทั้งหมด วงรอบนอกของการทำ FFT เป็นดังนี้

ในที่นี้ W คือตัวชี้ไปที่จุดเริ่มต้นของอาเรย์ของค่า twiddle factor W ก่อนการทำแต่ละ pass ระยะทางระหว่างจุดบนและจุดล่างของ butterfly ,le, เป็นครึ่งหนึ่ง และ wptr เริ่มที่ค่า twiddle factor แรก ที่แต่ละการสิ้นสุดของ pass ค่า windex จะเป็นสองเท่า

ส่วนของโปรแกรมข้างบนนี้เป็นส่วนหลังของโปรแกรม FFT ฟังก์ชัน FFT ที่สมบูรณ์,fft, แสดงไว้ใน ภาคผนวก ไม่เหมือนกับส่วนของโปรแกรมที่กล่าวกันมาก่อนแล้ว ที่ฟังก์ชัน fft จะใช้ประโยชน์จากค่า W{0} ที่เท่ากับ 1.0 นั้น หมายความว่าในวงรอบ for ของตัวแปร j ที่จุดเริ่มต้น (j=0) เมื่อ wptr ไม่ถูก windex เพิ่มค่า,ไม่ต้องการการทวีคูณ, ต้องการเพียงการลดหรือเพิ่ม ฟังก์ชัน fft จะใช้ประโยชน์จากการสร้างวงรอบที่ไม่สัมพันธ์กันเมื่อ j=0

ส่วนของโปรแกรมข้างบนนี้จะประมาณว่าอาเรย์ W ถูกให้ค่าไว้แล้ว และส่วนของ fft จะทำการให้ค่าอาเรย์ W โดยการเรียกตัวเองซ้ำ ด้วยประสิทธิภาพของการคำนวณวิธีนี้ทำให้ไม่ต้องเรียกใช้ฟังก์ชัน cos และ sin เลย และคุณลักษณะอื่นของโปรแกรมนี้คือ โปรแกรมจะให้ค่า W อย่าง static และถ้าการเรียกใช้ fft ครั้งต่อไปใช้ค่า FFT ที่ยาวเท่าเดิม (หรือว่าถ้าตัวแปร m มีค่าเท่ากับ mstore) ค่า W ก็จะนำมาใช้โดยไม่มีการคำนวณซ้ำ

ในส่วนสุดท้าย fft คือการกลับบิท ค่า input ที่ป้อนให้ขบวนการ DIF ปกติแล้วจะถูกเรียงจากข้างบนลงมาข้างล่างของหน้า ที่ output ลำดับขององค์ประกอบทางความถี่จะถูกเข้ารหัสไว้ ใน FFT แบบ radix 2 จะทำการเข้ารหัสเสมอด้วยวิธีการกลับบิท นั่นหมายความว่าถ้าตัวชี้ตำแหน่ง output ถูกเก็บเป็นแบบเลขฐานสองและมีจำนวนบิทพอที่จะแสดงค่าตัวชี้สูงสุด (m bits) ลำดับของ output ของ FFT สามารถอ่านโดยกลับลำดับการอ่านเช่น 00101011 (ตำแหน่ง 75 จากจุด FFT ทั้งหมด 256 จุด) ก็จะกลายเป็น 11010110 หรือ 212

ฟังก์ชัน fft จะทำการกลับบิทด้วยกระบวนการที่ดัดแปลงจากกระบวนการ Cooley-Turkey ส่วนของโปรแกรมต่อไปนี้ก็จะทำการกลับบิทเหมือนกัน แต่ใช้วิธีใช้ประโยชน์จากความสามารถทางบูลลีนของภาษา C

ถึงแม้ว่าวิธีที่สองนี้จะช้ากว่าวิธี Cooley-Turkey แต่แสดงให้เห็นการกลับบิทได้อย่างชัดเจน บิทจากตัวชี้ i จะนำไปเพิ่มให้กับตัวชี้ j ในลำดับที่กลับกัน เพื่อจะได้ให้ค่าตัวชี้ที่กลับบิทสุดท้าย

สำหรับ IFFT นั้นโปรแกรมเหมือนกับ FFT ยกเว้นจะมีการใช้ Complex conjugate ของสัมประสิทธิ์ และมีการทำการลดทอนผลลงด้วยอัตรา 1/N ในตอนจบของโปรแกรม

4.4.2 วิธีการแก้ไขโปรแกรม FFT ให้ใช้งานกับไมโครซอฟต์วินโดวส์

เดิมโปรแกรม FFT และ IFFT ดังกล่าวในหัวข้อ 4.3.1 นั้นทำการเขียนขึ้นบนระบบปฏิบัติการ ดอส ซึ่งมีการอ้างและขอใช้หน่วยความจำแตกต่างไปจากการทำงานบนไมโครซอฟท์วินโดวส์ จึงต้องทำการแก้ไขโปรแกรม จุดหลักในการแก้ไขคือการขอใช้หน่วยความจำ โปรแกรมเดิมนั้นใช้ฟังก์ชั่นของ C คือ calloc( ) ทำการจองหน่วยความจำขนาดเท่ากับ จำนวนจุดข้อมูลเข้า x ขนาดของตัวแปรแบบ Complex จึงต้องปรับมาใช้ฟังก์ชั่นของวินโดว์ส

แทน คือ GobalAlloc ( ) ซึ่งเป็นการขอใช้หน่วยความจำหลักของระบบ สาเหตุที่ใช้การขอหน่วยความจำของระบบ เนื่องจากในแต่ละวินโดว์ที่สร้างขึ้น จะสามารถขอหน่วยความจำแบบ local ได้จำกัด และได้ตัดส่วนการคำนวณค่า ของ w ซึ่งเดิมค้างเอาไว้ออก เนื่องจากในการใช้งานหส่วยความจำบนวินโดว์ส. ไม่ควรมีการ lock ใช้หน่วยความจำ เป็นเวลานาน และใช้วิธีการคำนวณใหม่ทุกครั้งแทน เนื่องจากทำการขอใช้หน่วยความจำแบบ Gobal ซึ่งสามารถ อ้างหน่วยความจำได้สูงสุดเท่ากับหน่วยความจำของระบบในขณะนั้น ผลที่ได้จากการขอจองแล้ว lock หน่วยความ จำ คือค่า pointer ของตัวแปรแบบ far ดังนั้นในโปรแกรมทั้งหมดจึงต้องเปลี่ยนตัวแปร pointer ที่อ้างอิงถึงเป็น ตัวแปรแบบ far แทนตัวแปรแบบ near เดิม ก็จะทำให้โปรแกรมสามารถนำมาใช้กับระบบปฏิบัติการไมโครซอฟท์ วินโดว์สได้ แต่ยังมีสิ่งที่ต้องคำนึงถึงคือในไมโครซอฟท์วินโดว์สจะไม่อนุญาตให้มีการใช้การติดต่ออินพุทเอาท์พุทกับ อุปกรณ์ต่างๆ เช่น คีย์บอร์ด เมาส์ ด้วยตัวโปรแกรมที่เราเขียนขึ้นเอง จะต้องขอใช้จากวินโดว์สเท่านั้น จุดสำคัญ อีกจุดหนึ่งคือ เมื่อขอใช้หน่วยความจำและ lock ได้แล้วจะต้องทำงานให้เสร็จเร็วที่สุด และคืนหน่วยความจำให้กับ วินโดว์สเร็วที่สุดเท่าที่จะเป็นไปได้ เนื่องจากวินโดว์สเป็นระบบปฏิบัติการแบบ Multitasking จึงต้องมีการใช้หน่วย ความจำร่วมกับโปรแกรมอื่นๆด้วย ซึ่งถ้าทำการ lock หน่วยความจำไว้นานเกินไป อาจทำให้เกิดปัญหาขึ้นกับข้อมูล ของหน่วยความจำส่วนนั้นๆได้ .

4.5 โปรแกรมทางสถิติที่นำมาใช้

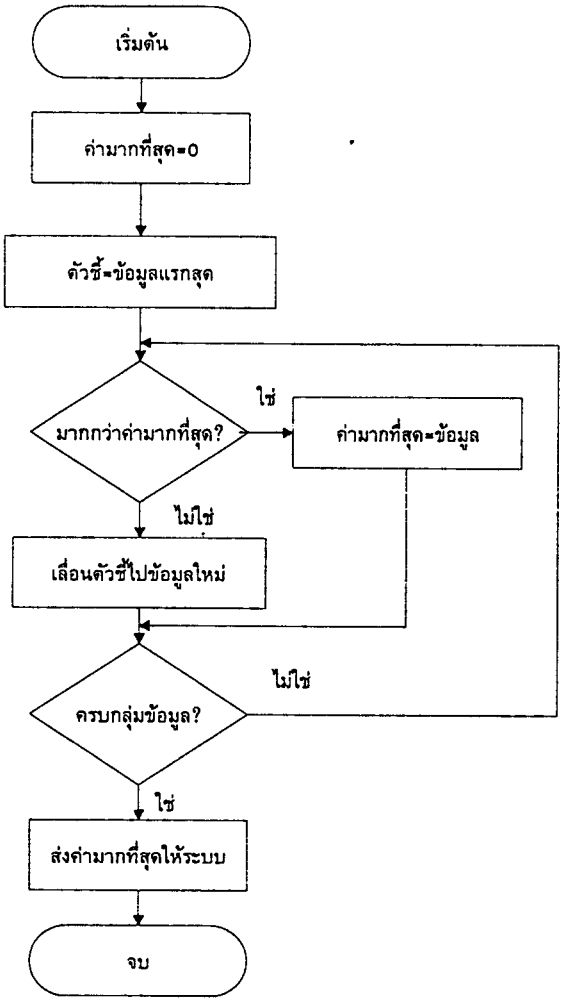
โปรแกรมทางสถิติที่ได้ทำการทดลองเขียนขึ้นใช้กับระบบในที่นี้ คือ โปรแกรมคำนวณหาค่ามากที่สุด, โปรแกรมคำนวณหาค่าน้อยที่สุด, โปรแกรมหาค่า S.D. และโปรแกรมหาค่ากลางตามปกติโดยทั่วไป ซึ่งโปรแกรม เหล่านี้ได้บรรจุไว้ในโปรแกรมของระบบที่พัฒนาขึ้นดังในภาคผนวก ค. แล้ว ดังนั้นจึงขอยกตัวอย่างลักษณะการ ทำงานเพื่อการหาค่ามากที่สุดและน้อยที่สุดเท่านั้น ซึ่งเป็นดังโฟลว์ชาร์ต รูป 4.7

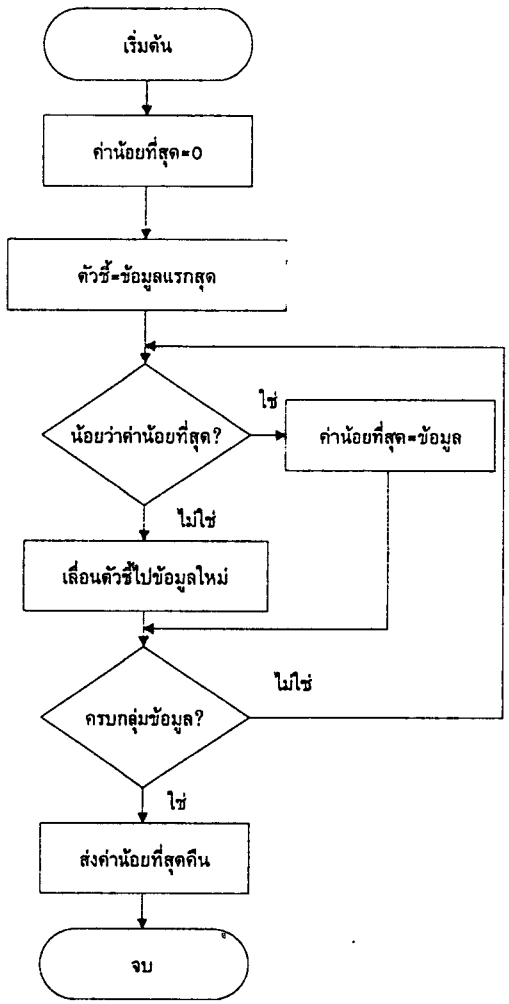
4.6 วิธีสร้างกราฟในการแสดงผล

ในการสร้างกราฟเพื่อการแสดงผลนั้น หลักการที่นำมาใช้คือเมื่อระบบซอฟท์แวร์ได้รับคำสั่งให้สร้างกราฟ โปรแกรมส่วนแสดงกราฟก็จะทำการคำนวณและแบ่งวินโดว์ส์ที่กำลังแสดงผลออกเป็น 3 ส่วน โดยที่ 2 ส่วนบน ทำหน้าที่แสดงกราฟที่ขยาย และส่วนล่างแสดงกราฟรวมของข้อมูลทั้งหมด เมื่อมีการกดเมาส์ให้กับจุดใดจุดหนึ่ง บนพื้นที่แสดงภาพรวม ซึ่งขั้นตอนโฟลว์ชาร์ตของวิธีการจัดการกราฟ ดังรูป 4.4 และ 4.5

4.7 วิธีเพิ่มโปรแกรมย่อยเพื่อการวิเคราะห์

- เพิ่มคำสั่ง #include <"ชื่อไฟล์ของโปรแกรมที่เพิ่ม">
- เพิ่มรายการของ Menu ในหัวข้อ Analysis
- เพิ่ม ID ของ Menu ที่เพิ่มขึ้นใหม่
- แทรกคำสั่ง case <ID ของ Menu> ใน case ของ WM_COMMAND
- แทรกการเรียกใช้โปรแกรมวิเคราะห์บรรทัดถัดจาก case <ID ของ Menu>
- กรณีที่มีต้องมีการเก็บผลการวิเคราะห์แล้ว ให้ตรวจสอบหาตำแหน่งว่างในตารางของ MainVar ว่ามี ตำแหน่งใดว่างบ้างแล้วนำค่าไปเก็บไว้พร้อมตั้งชื่อ
- การแสดงภาพผู้ใช้ระบบจะต้องเป็นผู้สั่งด้วยการเลือก menu Name -> Display





รูป 4.7 โฟลว์ชาร์ตการทำงานของการหาค่ามากที่สุดและน้อยที่สุด

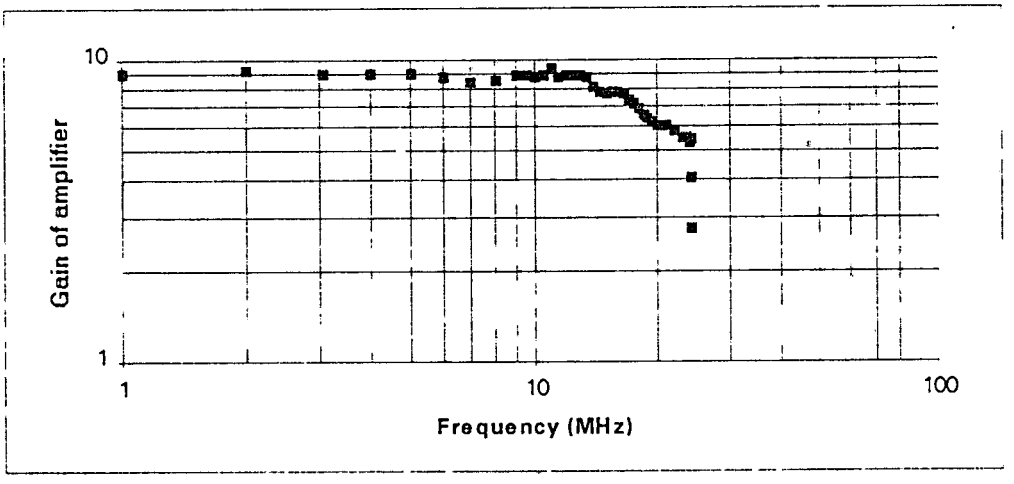
บทที่ 5

ผลการทดสอบระบบ

ในการทดสอบการทำงานได้แบ่งทำการทดสอบเป็นส่วนอนาลอกและส่วนดิจิตอล

5.1 ส่วนอนาลอก

5.1.1 ผลการทดสอบอัตราขยายของวงจรขยายที่ออกแบบขึ้น



รูป 5.1 กราฟแสดงอัตราขยายของวงจรต่อความถี่

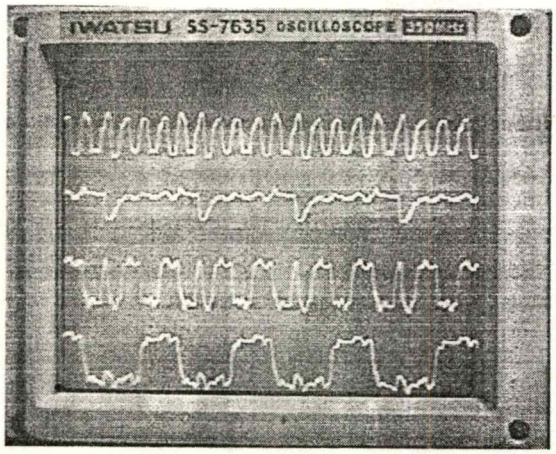
5.2 ส่วนดิจิตอล

เนื่องจากเครื่องมือในการเก็บภาพเช่น logic analyzer มีอยู่มีความเร็วไม่มากพอในการจับสัญญาณของระบบเก็บข้อมูลดังนั้นในวิทยานิพนธ์นี้ได้ใช้การเก็บภาพจากสโคปด้วยกล้องถ่ายภาพโทรทัศน์ลงในเครื่องคอมพิวเตอร์แทน ซึ่งสามารถแสดงสัญญาณได้พร้อมกันเพียง 4 สัญญาณเท่านั้น และสัญญาณบางกลุ่มเช่นสัญญาณที่เกิดขึ้นในขณะระบบกำลังทำการเก็บข้อมูลนั้นไม่สามารถบันทึกได้เนื่องจากค่าตำแหน่งและสัญญาณมีการเปลี่ยนแปลงไปตลอดเวลาจึงไม่ได้ทำการบันทึกไว้ในวิทยานิพนธ์

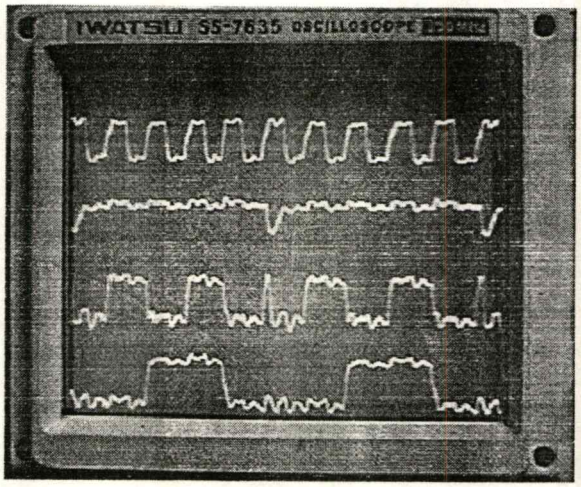
5.2.1 ผลการทดสอบวงจรเลือกช่องสัญญาณ

สำหรับการทดสอบในส่วนนี้ได้จับสัญญาณ 4 สัญญาณพร้อมกัน โดยที่สัญญาณแต่ละเส้นบนจอภาพ เป็นสัญญาณดังต่อไปนี้

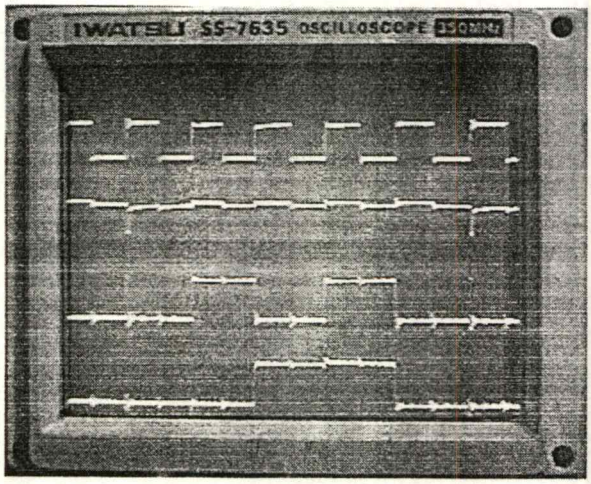
- สัญญาณเส้นบนสุดเป็นสัญญาณอัตราเก็บข้อมูล
- สัญญาณเส้นสองเป็นสัญญาณรีเซทที่กลับมาเคลียร์วงจร
- สัญญาณเส้นสามเป็นสัญญาณที่เอาท์พุทวงจรนับ บิทต่ำสุด (บิท 0)
- สัญญาณเส้นสี่เป็นสัญญาณที่เอาท์พุทวงจรนับ บิท 1



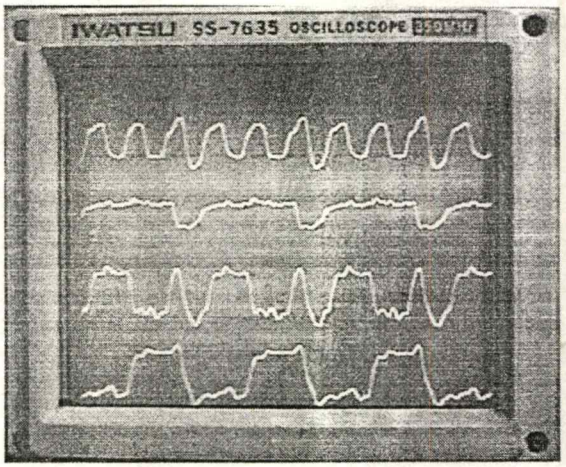
รูป 5.2 สัญญาณของวงจรเลือกช่องสัญญาณที่อัตราเก็บข้อมูล 20 เมกะเฮิทซ์ ที่จำนวนช่องสัญญาณเข้า 5 ช่อง



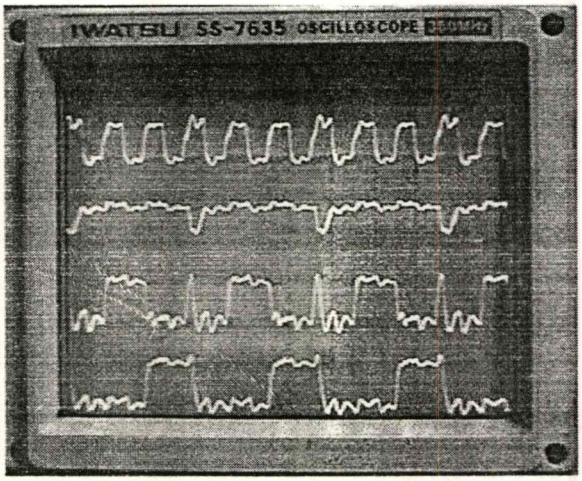
รูป 5.3 สัญญาณของวงจรเลือกช่องสัญญาณที่อัตราเก็บข้อมูล 10 เมกะเฮิทซ์ ที่จำนวนช่องสัญญาณเข้า 5 ช่อง



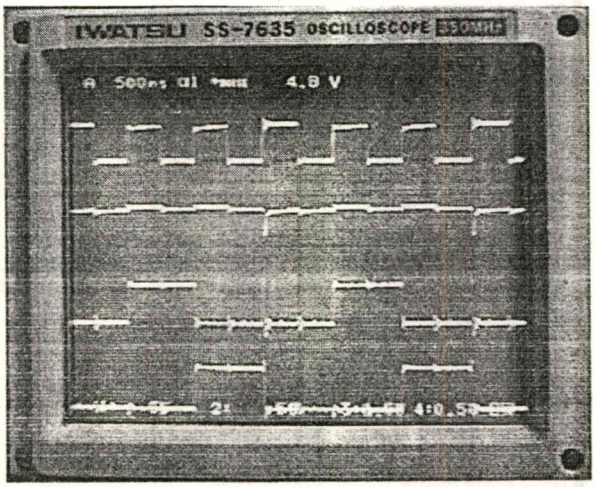
รูป 5.4 สัญญาณของวงจรเลือกช่องสัญญาณที่อัตราเก็บข้อมูล 1 เมกะเฮิทซ์ ที่จำนวนช่องสัญญาณเข้า 5 ช่อง



รูป 5.5 สัญญาณของวงจรเลือกช่องสัญญาณที่อัตราเก็บข้อมูล 20 เมกะเฮิทซ์ ที่จำนวนช่องสัญญาณเข้า 3 ช่อง



รูป 5.6 สัญญาณของวงจรเลือกช่องสัญญาณที่อัตราเก็บข้อมูล 10 เมกะเฮิทซ์ ที่จำนวนช่องสัญญาณเข้า 3 ช่อง

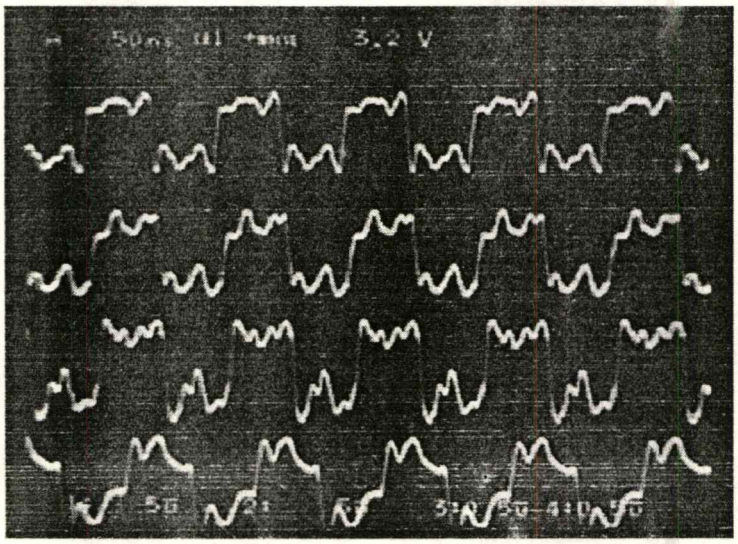


รูป 5.7 สัญญาณของวงจรเลือกช่องสัญญาณที่อัตราเก็บข้อมูล 1 เมกะเฮิทซ์ ที่จำนวนช่องสัญญาณเข้า 3 ช่อง

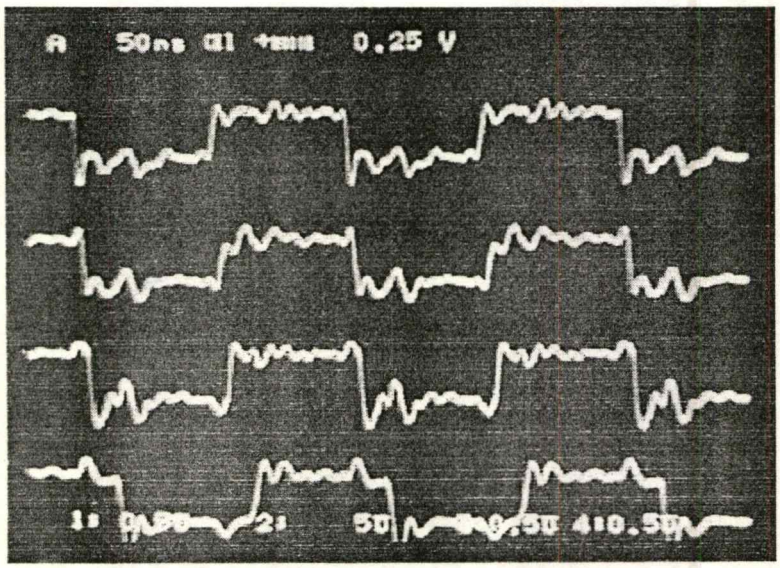
5.2.2 ผลการทดสอบวงจรควบคุมจังหวะการอ่านเขียนหน่วยความจำที่ความเร็วสูง

สำหรับการทดสอบในส่วนนี้ได้จับสัญญาณ 4 สัญญาณพร้อมกัน โดยที่สัญญาณแต่ละเส้นบนจอภาพเป็นสัญญาณดังต่อไปนี้

- สัญญาณเส้นบนสุดเป็นสัญญาณอัตราเก็บข้อมูลที่ผ่านวงจรหารสองแล้ว
- สัญญาณเส้นสองเป็นสัญญาณที่ผ่าน and gate เพื่อเพิ่มเวลาหน่วง
- สัญญาณเส้นสามเป็นสัญญาณก่อนเข้า delay line เพื่อหน่วงไป 15 นาโนวินาที
- สัญญาณเส้นสี่เป็นสัญญาณที่หน่วงไปแล้วและนำไปทริก latch ให้ผ่านค่าตำแหน่งหน่วยความจำ



รูป 5.8 ลักษณะสัญญาณวงจรควบคุมจังหวะการอ่านเขียนหน่วยความจำ ที่อัตราเก็บข้อมูล 10 เมกกะเฮิทซ์

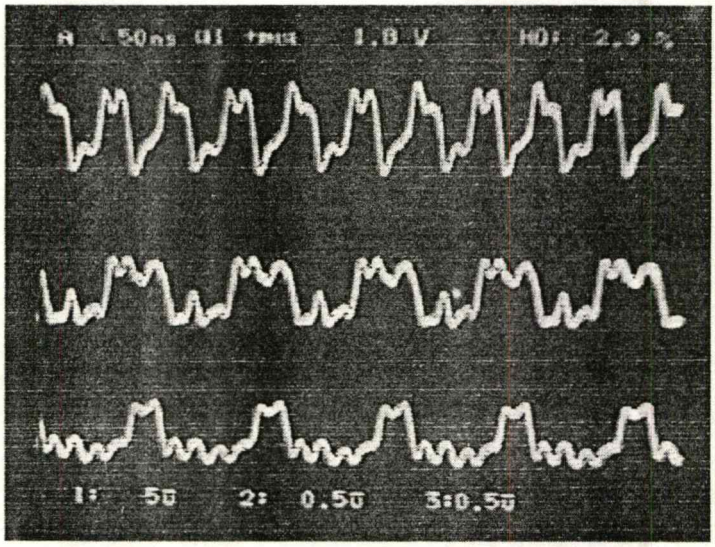


รูป 5.9 ลักษณะสัญญาณวงจรควบคุมจังหวะการอ่านเขียนหน่วยความจำ ที่อัตราเก็บข้อมูล 5 เมกกะเฮิทซ์

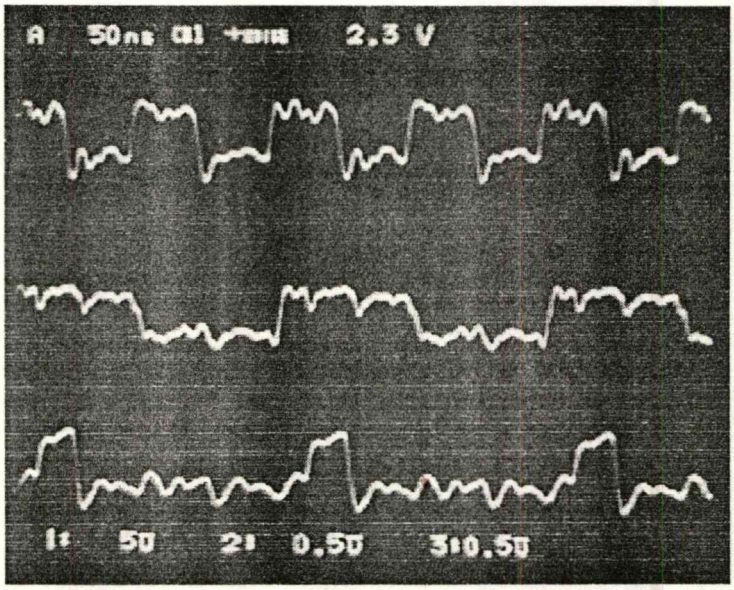
5.2.3 ผลการทดสอบวงจรสร้างสัญญาณพัลส์แคบ

สำหรับการทดสอบในส่วนนี้ได้จับสัญญาณ 3 สัญญาณพร้อมกัน โดยที่สัญญาณแต่ละเส้นบนจอภาพเป็นสัญญาณดังต่อไปนี้

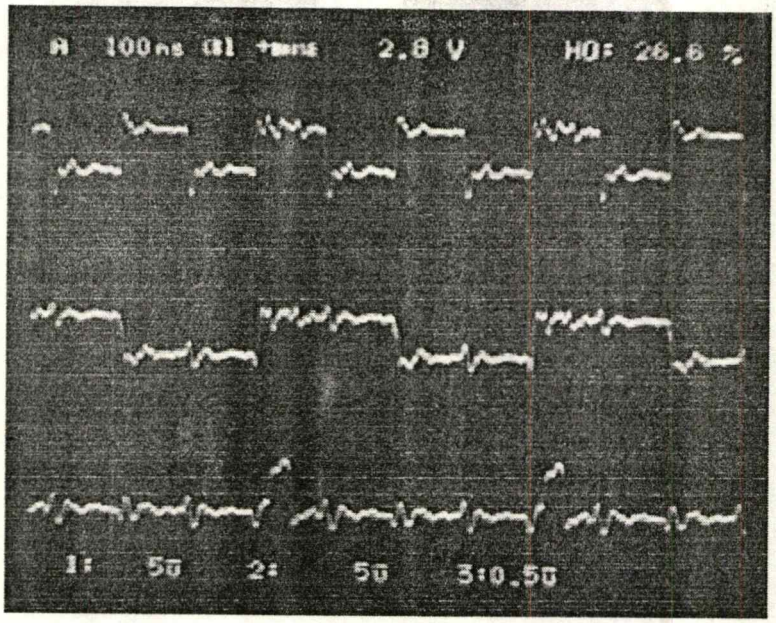
- สัญญาณเส้นบนสุดเป็นสัญญาณอัตราเก็บข้อมูล
- สัญญาณเส้นสองเป็นสัญญาณที่ทริก latch ให้ค้างค่าตำแหน่งหน่วยความจำ
- สัญญาณเส้นสามเป็นสัญญาณพัลส์ขนาดแคบมากที่ใช้ในการควบคุมการกำหนดสัญญาณ chip select ของหน่วยความจำ



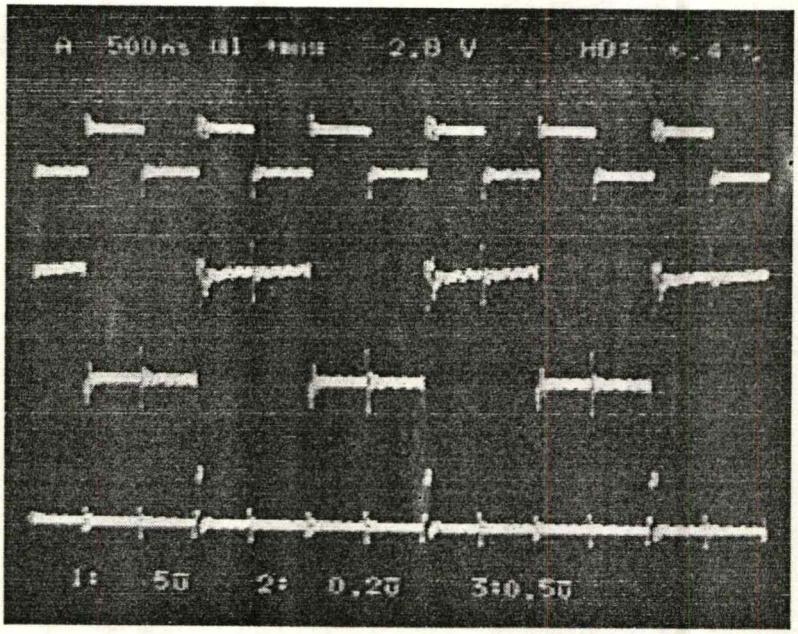
รูป 5.10 สัญญาณของวงจรสร้างพัลส์แคบที่อัตราเก็บข้อมูล 20 เมกกะเฮิทซ์



รูป 5.11 สัญญาณของวงจรสร้างพัลส์แคบที่อัตราเก็บข้อมูล 10 เมกกะเฮิทซ์



รูป 5.12 สัญญาณของวงจรสร้างพัลส์แคบที่อัตราเก็บข้อมูล 5 เมกกะเฮิทซ์



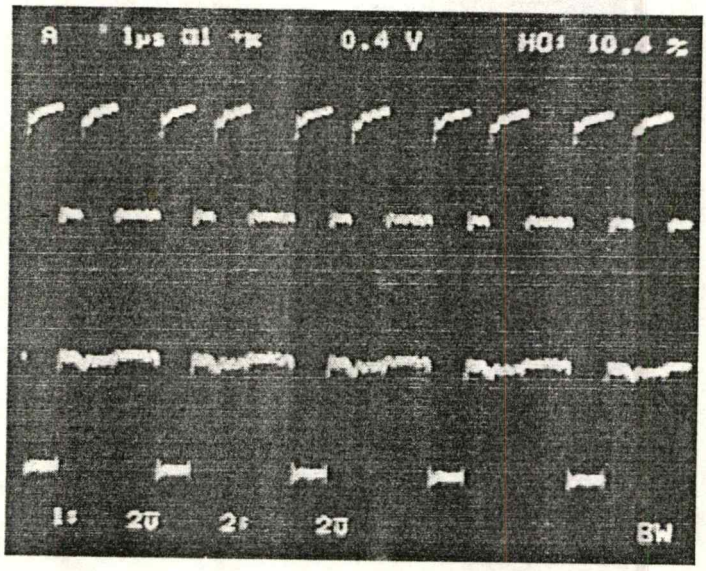
รูป 5.13 สัญญาณของวงจรสร้างพัลส์แคบที่อัตราเก็บข้อมูล 1 เมกกะเฮิทซ์

5.2.4 ผลการทดสอบวงจรหน่วยความจำเมื่อติดต่อกับระบบคอมพิวเตอร์

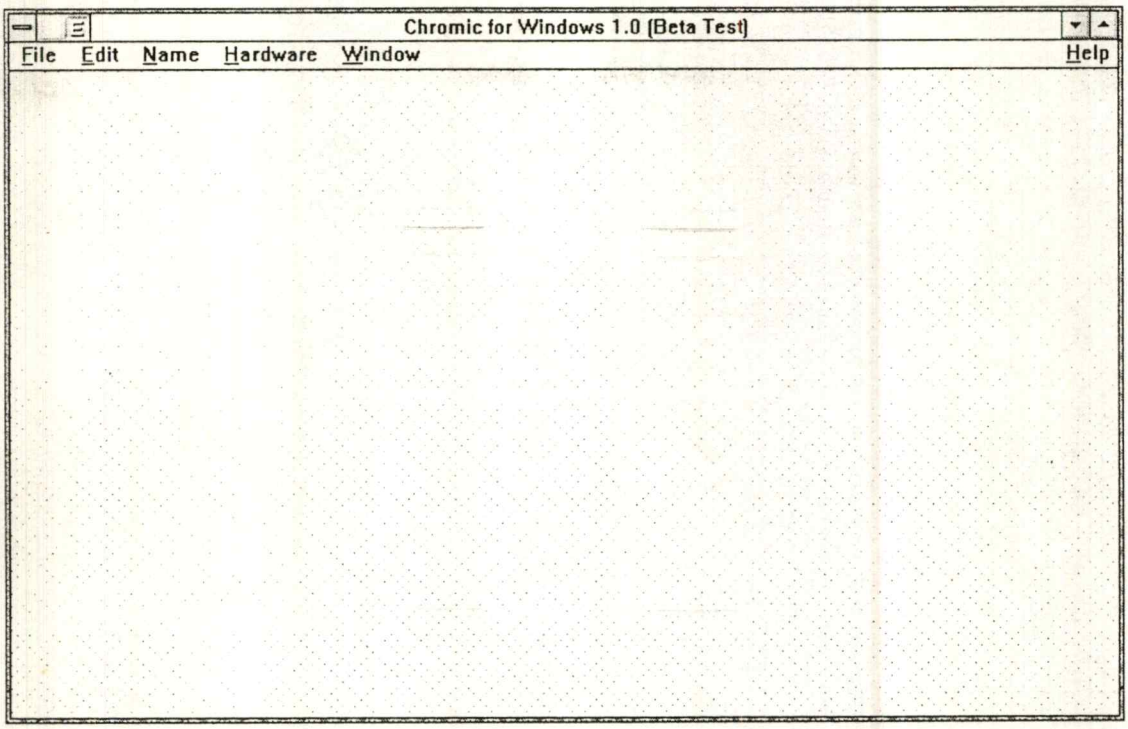
สำหรับการทดสอบในส่วนนี้ได้จับสัญญาณ 2 สัญญาณพร้อมกัน โดยที่สัญญาณแต่ละเส้นบนจอภาพเป็นสัญญาณดังต่อไปนี้

- สัญญาณเส้นบนสุดเป็นสัญญาณที่ขา -OE ของหน่วยความจำ
- สัญญาณเส้นสองเป็นสัญญาณที่ขาสัญญาณ chip selct ของหน่วยความจำ

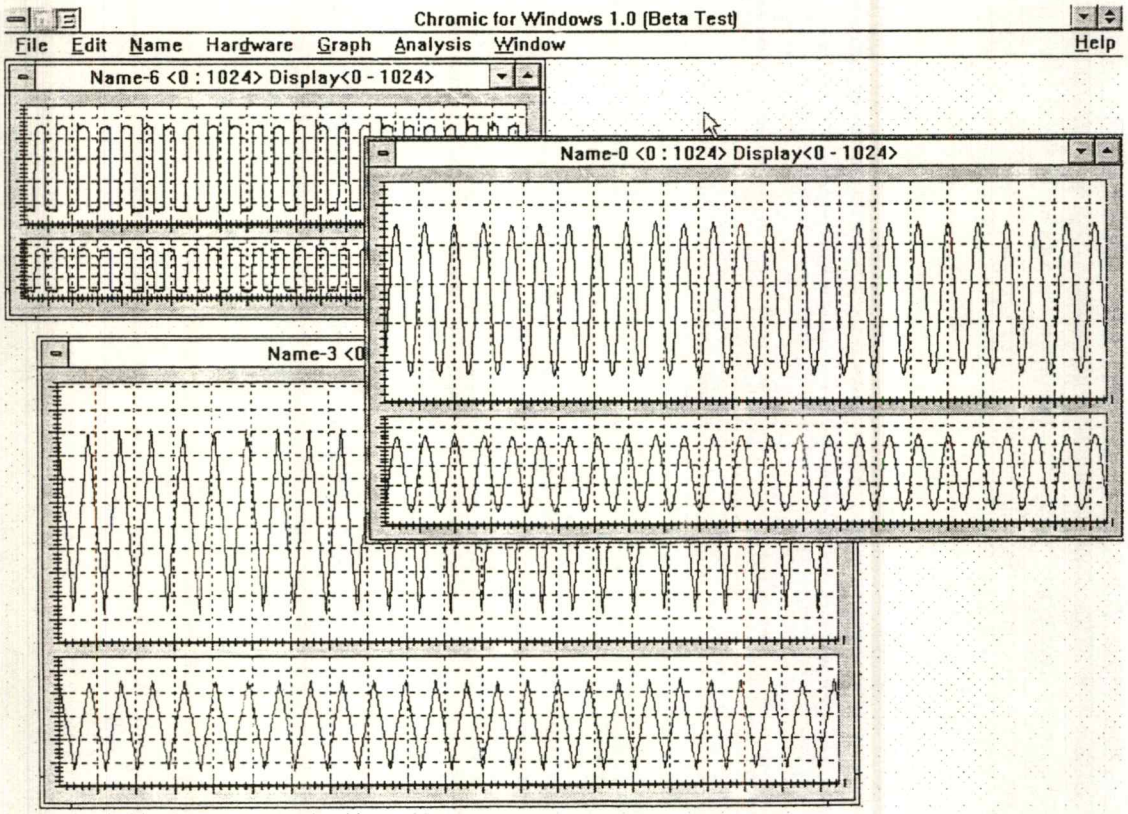
จากรูป 5.14 เป็นสัญญาณในขณะที่ระบบคอมพิวเตอร์กำลังเขียนข้อมูลเข้าสู่หน่วยความจำของระบบเก็บข้อมูล จะเห็นได้ว่าสัญญาณที่ขา -OE ถูกสั่งเปลี่ยนลอจิกไปเป็นลอจิกสูง ในขณะที่เกิดการเลือกหน่วยความจำที่จะเขียนข้อมูล



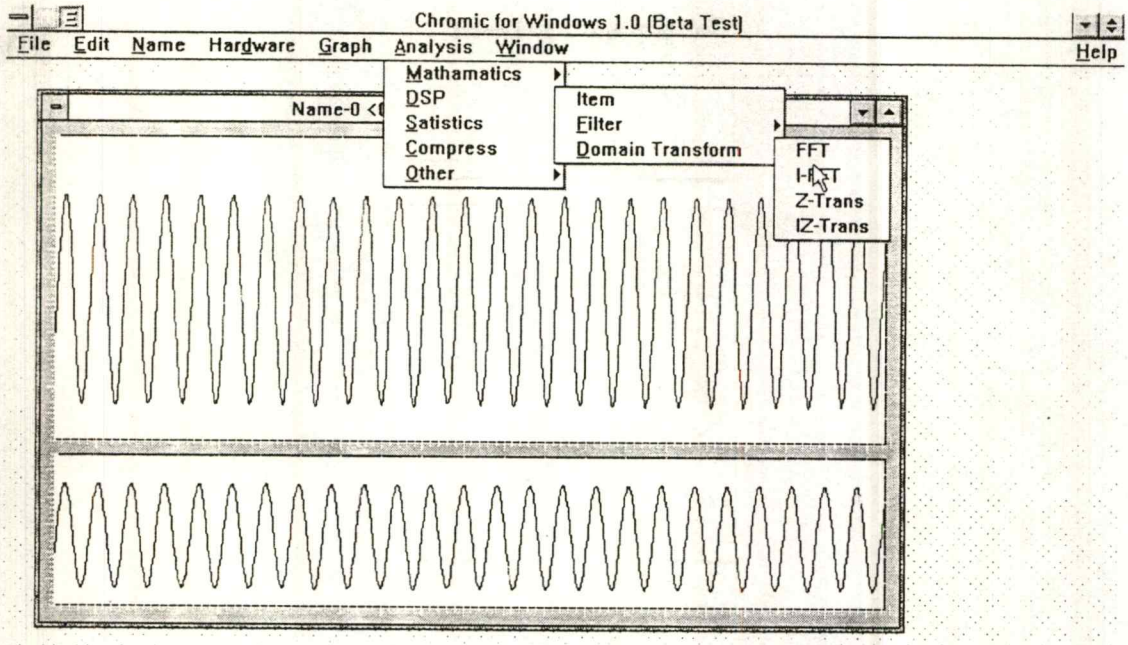
รูป 5.14 สัญญาณที่ขาสัญญาณบางขาของหน่วยความจำในขณะที่กำลังติดต่อกับระบบคอมพิวเตอร์



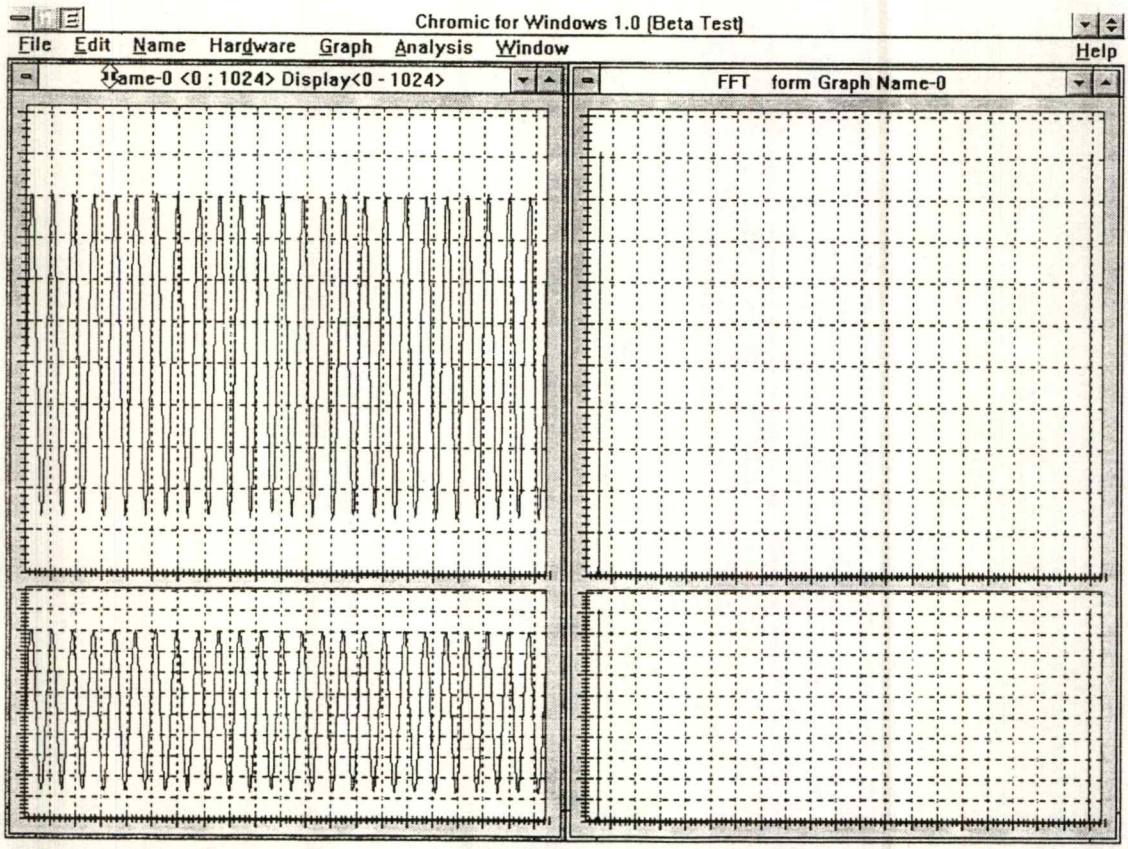
รูป 5.15 แสดงจอภาพคอมพิวเคอร์เมื่อระบบซอฟท์แวร์ทำงานในช่วงเริ่มต้น



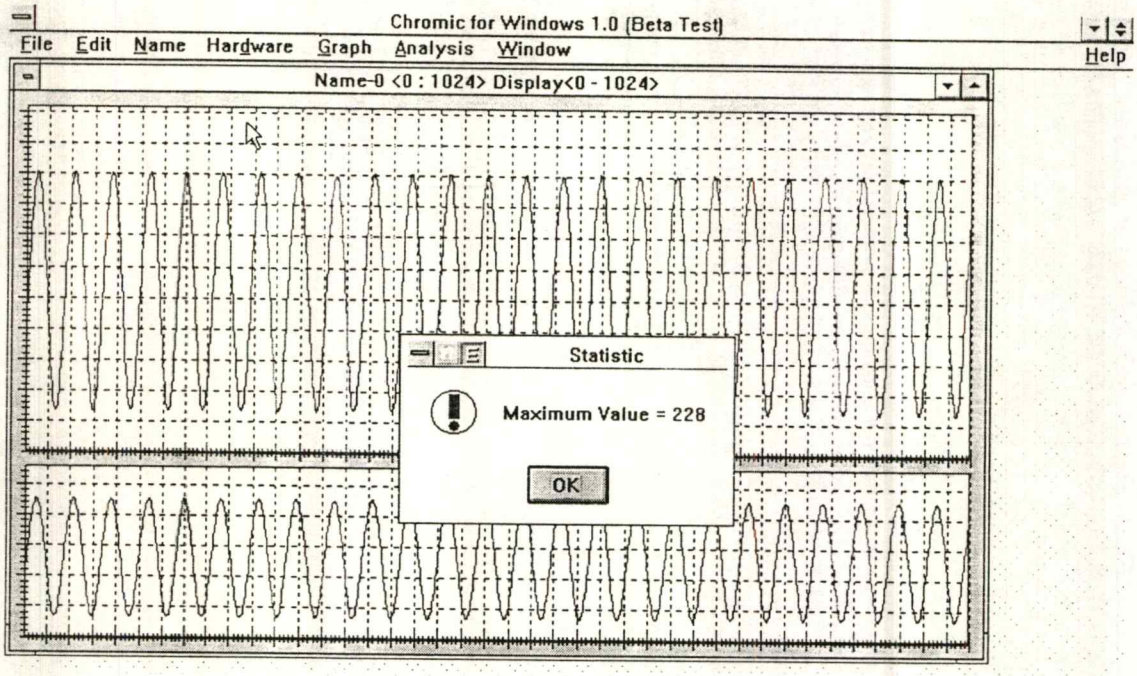
รูป 5.16 แสดงจอภาพคอมพิวเตอร์เมื่อระบบซอฟท์แวร์แสดงภาพของสัญญานอนาลอกเข้า 500 กิโลเฮิร์ท ลักษณะ sine, triangleและ square ที่อัตราเก็บข้อมูล 20 เมกกะเฮิร์ท



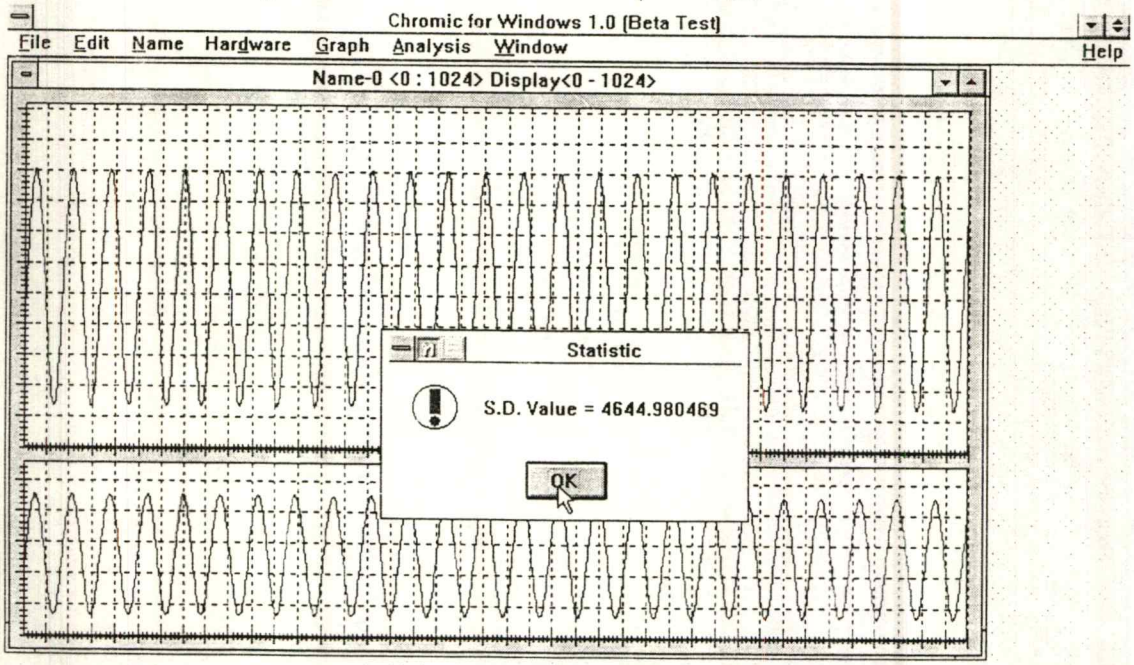
รูป 5.17 แสดงจอภาพคอมพิวเตอร์เมื่อใช้ระบบซอฟท์แวร์ ในขณะที่กำลังเลือกการทำ FFT



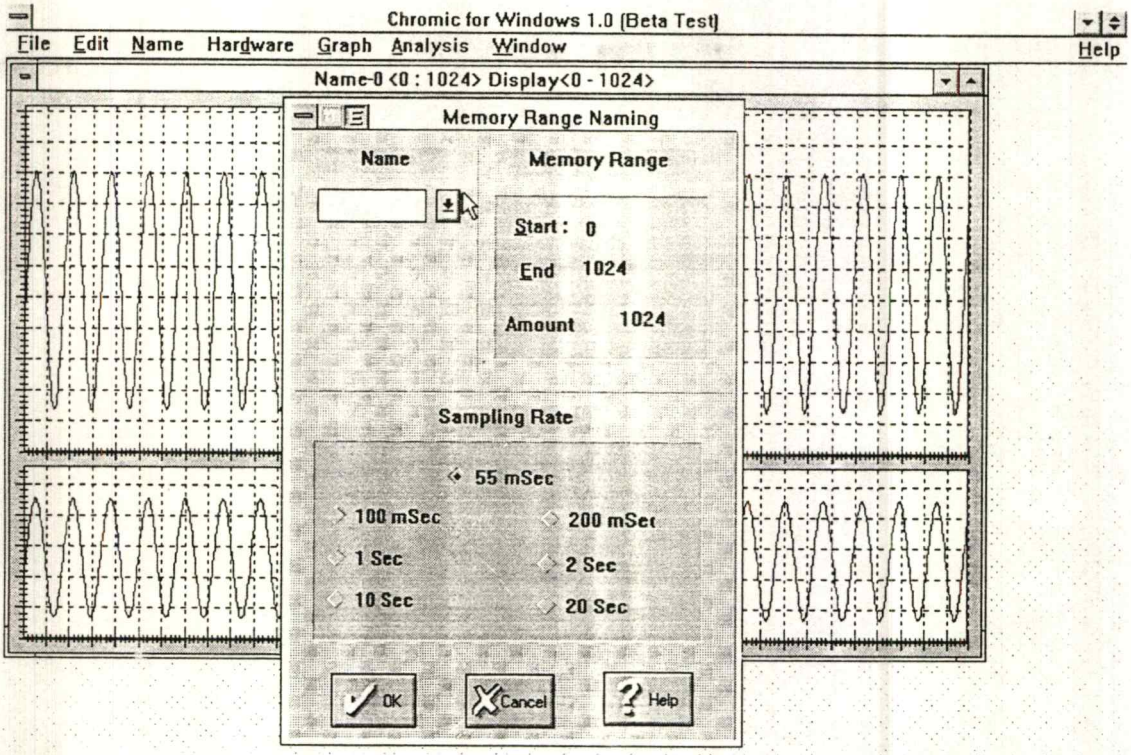
รูป 5.18 แสดงภาพสัญญาณเข้าแบบ sine และผลการทำ FFT



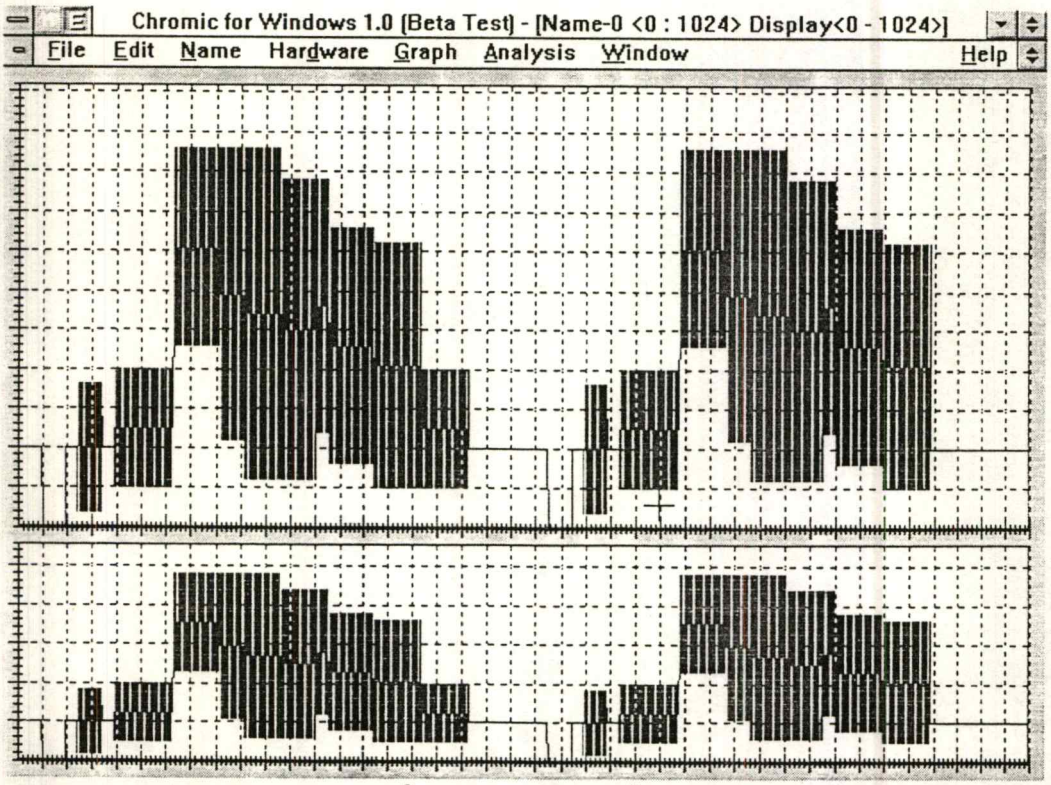
รูป 5.19 แสดงภาพการหาค่ามากที่สุดของสัญญาณเข้า



รูป 5.20 แสดงภาพการหาค่า SD ของสัญญาณเข้า



รูป 5.21 แสดงการกำหนดชื่อเพื่อเก็บข้อมูลแบบ real time



รูป 5.22 แสดงผลของการใช้งานระบบเก็บข้อมูลในการเก็บสัญญาณโทรทัศน์

เอกสารอ้างอิง

[1] "Fast Advanced Schottky TTL Logic Databook", National Semiconductor, 1988

[2] "Samsung Mos Memory Databook", Samsung Semiconductor, 1989

[3] "Signatics Analog Data Manual", Signetics Corporation, 1983

[4] "Linear Products Databook", Analog Devices, Inc., 1988

[5] "High Speed Design Seminar", Analog Devices Inc,1989

[6] "RCA Data conversion book", RCA, 1989

[7] "Borland C++ 2.0 Library reference", Borland International,. Inc, 1991

[8] Charles Petzold, "Programming Windows", Microsoft Press, 1990

[9] Conger, Jame L., "Windows API Bible", The Waite Group Press, 1992

[10] Conger, Jame L., "Windows Programming Primer Plus", The Waite Group Press, 1992

[11] Douglas F. Elliott, "Handbook of Digital Signal Processing", Academic press Inc., 1987

[12] Lee Adams, "High-Performance C Graphics Programming for Windows", Windcrest Books,1992

[13] Paul M. Embree and Bruce Kimble, "C Language Algorithms for Digital Signal Processing", Prentice-Hall International, Inc., 1991

[14] Peter Norton & Paul L. Yao, "Windows 3.0 Power Programming Technique", Bantum Books Inc., 1990

[15] Sergio Franco, "Design with Operational Amplifiers and Analog Integrated Circuits", McGraw-Hill book Co., 1988

บทที่ 6
สรุป วิจารณ์ และ แนวทางการแก้ไขปรับปรุงประสิทธิภาพ

สำหรับผลของการศึกษาตามวิทยานิพนธ์นี้ได้พบว่าสามารถออกแบบระบบเก็บและวิเคราะห์ข้อมูลความเร็วสูงขึ้นใช้เองในประเทศได้ ถึงแม้ว่าจะมีความยุ่งยากบ้างในการจัดหาอุปกรณ์ แต่สิ่งได้รับคือพื้นฐานความรู้และความเข้าใจในการออกแบบระบบทั้งอนาลอกและดิจิตอลที่ความเร็วสูง ซึ่งในปัจจุบันมีการศึกษาระบบแบบนี้ไม่มากนักและเป็นพื้นฐานของการพัฒนาระบบที่ความเร็วสูงขึ้นในอนาคต จากการออกแบบสร้างเครื่องต้นแบบ พบว่าสามารถทำการเก็บข้อมูลเข้าหน่วยความจำแบบดิจิตอลได้เร็วถึง 20 เมกกะเฮิร์ท

สิ่งที่พบในการทดลองระบบคือเมื่อนำระบบดิจิตอลและระบบอนาลอกมาต่อเชื่อมกันเป็นระบบที่สมบูรณ์เกิดมีสัญญาณรบกวนขึ้นกับระบบทั้งหมด ทำให้ข้อมูลที่ได้จากการสุ่มข้อมูลผิดพลาด นอกจากนั้นยังพบว่าสัญญาณรบกวนจากภายนอก เช่น ในขณะที่กระแสไฟฟ้าบ้านตกหรือกระพริบ จะทำให้วงจรส่วนดิจิตอลเกิดสภาวะไม่รับคำสั่งในการทำงานได้ และสิ่งที่พบอีกอันหนึ่งคือ ระบบซอฟท์แวร์มีการทำงานผิดพลาด โดยเฉพาะในส่วนของโปรแกรมวิเคราะห์ข้อมูล IFFT ให้ผลการวิเคราะห์ผิดจากความเป็นจริง และการอ้างอิงหน่วยความจำผิดพลาด ทำให้มีการแสดงภาพผิดรูป

ปัญหาที่เกิดขึ้นในการทำวิทยานิพนธ์นี้แบ่งได้สองส่วนคือปัญหาที่เกิดขึ้นระหว่างการออกแบบระบบและปัญหาที่เกิดขึ้นเมื่อทำการสร้าง ในการออกแบบปัญหาที่พบคือจะต้องพิจารณาข้อมูลของอุปกรณ์เพื่อทำการเลือกใช้ซึ่งหาได้ค่อนข้างยาก และการเลือกอุปกรณ์มาใช้งาน ซึ่งจำเป็นต้องคำนึงถึงการจัดหาให้ได้ภายในประเทศเป็นหลัก ในระหว่างการออกแบบระบบนั้นสิ่งที่ต้องระมัดระวังมากคือ ผังเวลาในการทำงานของระบบเนื่องจากระบบทำงานที่ความเร็วค่อนข้างสูง การที่ออกแบบผังเวลาของการทำงานของระบบผิดพลาดเพียงเล็กน้อย จะกระทบกระเทือนถึงวงจรส่วนอื่นซึ่งมีความสัมพันธ์กันหมด ทำให้เสียเวลาไปค่อนข้างมากในการออกแบบระบบให้ลงตัว ในการสร้างระบบตามที่ออกแบบขึ้นปัญหาที่พบมากที่สุดคือคุณสมบัติของอุปกรณ์แตกต่างจากมาตรฐานไปมาก จนบางครั้งทำให้การออกแบบซึ่งออกแบบให้เผื่ออุปกรณ์ไม่ได้มาตรฐานยังไม่สามารถทำงานได้ แม้ว่าจะทำการจัดหามาใหม่ก็ยังคงไม่ได้ตามมาตรฐาน นอกจากนั้นอุปกรณ์ที่ใช้ในกลุ่มของการทำงานที่ความเร็วสูงนั้นหาซื้อได้ยากและจะมีไม่กี่แห่ง ส่วนที่มีก็มีเบอร์ให้เลือกใช้ไม่มากนัก ทำให้บางครั้งต้องทำการออกแบบใหม่และทำให้การออกแบบลายวงจรซับซ้อนมากขึ้น เป็นผลทำให้เกิดข้อผิดพลาดได้ง่าย

ปัญหาที่พบในการออกแบบระบบอนาลอกคือมีสัญญาณรบกวนเข้าสู่ระบบได้ง่ายมากเมื่อระบบที่สร้างขึ้นทำงานกับความถี่สูง การเปลี่ยนแปลงอุปกรณ์เพียงเล็กน้อยก็ทำให้วงจรได้ผลที่ไม่ต้องการออกมา บางครั้งทำให้เกิดการออสซิเลต อุปกรณ์ออปแอมป์ที่นำมาใช้ในครั้งแรกทำงานได้ดีที่ระดับสัญญาณขนาดเล็กๆเท่านั้น เมื่อมีสภาวะที่สัญญาณเอาท์พุทมากกว่า 0.5 โวลต์ก็จะเกิดการตัดยอดของสัญญาณ ทำให้เสียเวลาในการหาคำตอบไปส่วนหนึ่ง เนื่องจากคาดว่าความผิดพลาดเกิดขึ้นจากการออกแบบ นอกจากนี้ระบบจ่ายไฟเลี้ยงของวงจรที่ทดลองก็เป็นส่วนหนึ่งที่มีปัญหามาก เนื่องจากสัญญาณจากระบบไฟเลี้ยงในการทดลองบางครั้งเข้าไปกวนทำให้ระบบให้เอาท์พุทที่ผิดพลาดหรือบางครั้งทำให้ระบบทำงานไม่ได้

เนื่องจากระบบมีปัญหาเรื่องสัญญาณรบกวนค่อนข้างมาก ดังนั้นจึงควรปรับปรุงเรื่องการป้องกันสัญญาณรบกวนเข้าสู่ระบบโดยแยกพิจารณาเป็นส่วนๆดังนี้คือ ระบบไฟเลี้ยง,ระบบกราวด์ และการเดินลายวงจร

ระบบไฟเลี้ยง ควรแยกไฟเลี้ยงของวงจรอนาลอกและวงจรดิจิตอลออกจากกันอย่างเด็ดขาดโดยใช้ Ferrite bead ขนาดเล็ก (โดยปกติจะมีค่าความต้านทานประมาณ 10 ถึง 50 โอห์มที่ 10 เมกกะเฮิร์ท) ทำการแยก (isolate) แรงดันไฟเลี้ยงของระบบ ไฟเลี้ยงที่จ่ายให้กับไอซีแต่ละตัวควร decouple ด้วยตัวเก็บประจุแบบเซรามิคที่มีค่าเหนี่ยวนำต่ำ ค่าประมาณ 0.01 - 0.1 ไมโครฟารัดและตัวเก็บประจุนี้ควรจะอยู่ใกล้กับอุปกรณ์มากที่สุดเท่าที่จะเป็นไปได้ เพื่อให้ได้ผลดีที่สุดควรเลือกใช้ตัวเก็บประจุแบบไร้ขา (Surface mounted chip capacitor) เพื่อไม่ให้มีค่าความเหนี่ยวนำแฝงที่เกิดขึ้นจากลวดขาของตัวเก็บประจุ นอกเหนือจากการบายพาสด้วยตัวเก็บประจุแบบเซรามิคแล้ว เพาเวอร์ซับพลายของแต่ละส่วนควรใช้ตัวเก็บประจุแบบแทนทาลัมที่มีคุณภาพสูง ค่าประมาณ 3 ถึง 20 ไมโครฟารัด บายพาสลงสู่กราวด์เพลนเพื่อกำจัดสัญญาณรบกวนความถี่ต่ำออกไป ข้อควรระวังข้อหนึ่งคือค่าเหนี่ยวนำของ Ferrite bead อาจจะทำให้เกิดวงจรเรโซแนนท์กับตัวเก็บประจุบายพาสได้.

ระบบกราวด์ ระบบที่มีการแยกกราวด์ออกเป็นกราวด์ของระบบอนาลอกและกราวด์ของระบบดิจิตอล ซึ่งกราวด์ของระบบทั้งสองนี้จะเชื่อมต่อกันที่แหล่งจ่ายไฟของระบบทั้งหมด จะทำให้เกิดปัญหากับตัวแปลงสัญญาณอนาลอกเป็นดิจิตอลแบบแฟลช เนื่องจากตัวแปลงสัญญาณอนาลอกเป็นดิจิตอลแบบแฟลชนี้มีการแยกขากราวด์ภายในของไอซีเป็นขาอนาลอกกราวด์และขาดิจิตอลกราวด์ภายในตัวไอซี เพื่อให้ตัวแปลงสัญญาณอนาลอกเป็นสัญญาณดิจิตอลมีประสิทธิภาพในการทำงานสูงสุด ดังนั้นขากราวด์ทั้งสองจะต้องเชื่อมต่อกับกราวด์เพลนของแต่ละระบบที่จุดที่ใกล้ตัวไอซีให้มากที่สุด แต่ในความเป็นจริงนั้นไม่สามารถออกแบบให้เป็นเช่นนั้นได้ ทางแก้ปัญหาอันหนึ่งคือออกแบบให้ไอซีนี้อยู่บนส่วนของวงจรอนาลอกที่มีกราวด์เพลนต่อไปถึงกราวด์ของระบบอนาลอกที่แหล่งจ่ายไฟของระบบ อีกวิธีหนึ่งคือ ยังคงแยก DC กราวด์เพลนของอนาลอกและดิจิตอลกราวด์เพลนไว้ แต่ต่อถึงกันทาง AC โดยใช้ตัวเก็บประจุแบบเซรามิคที่มีคุณภาพสูง (มีค่าประมาณ 0.01 ถึง 0.1 ไมโครฟารัด). การต่อขาแต่ละขาของไอซีลงสู่กราวด์เพลนควรเดินลายทองแดงสั้นที่สุด เพื่อลดค่าความเหนี่ยวนำแฝงให้น้อยที่สุด ถ้าเป็นไปได้ควรทำเป็นแบบเพลตทรูโฮลลงสู่กราวด์เพลนจะเป็นการดีที่สุด แต่ถ้าไม่สามารถทำได้ให้ลงกราวด์โดยเจาะตาไก่ลงบนแผ่นวงจรพิมพ์ กราวด์เพลนควรจะมีครอบคลุมพื้นที่ของแผ่นวงจรพิมพ์ให้มากที่สุดเท่าที่จะเป็นไปได้ (โดยปกติควรจะมากกว่า 75%) ถ้าใช้แผ่นวงจรพิมพ์แบบสองหน้า ด้านที่วางอุปกรณ์ควรเป็นกราวด์เพลนทั้งหมด และอีกด้านหนึ่งจะเป็นลายวงจรที่ใช้เชื่อมต่ออุปกรณ์ กราวด์เพลนควรจะแผ่เข้าไปใต้ตัวอุปกรณ์ให้มากที่สุดเท่าที่จะมากได้ ถ้าหากวงจรมีอุปกรณ์หนาแน่นมาก ทำให้มีพื้นที่สำหรับกราวน์เพลนน้อยลง ดังนั้นจึงควรเลือกใช้แผ่นวงจรพิมพ์แบบหลายชั้น เพื่อที่จะสามารถเลือกใช้ชั้นใดชั้นหนึ่งเป็นกราวด์เพลนได้

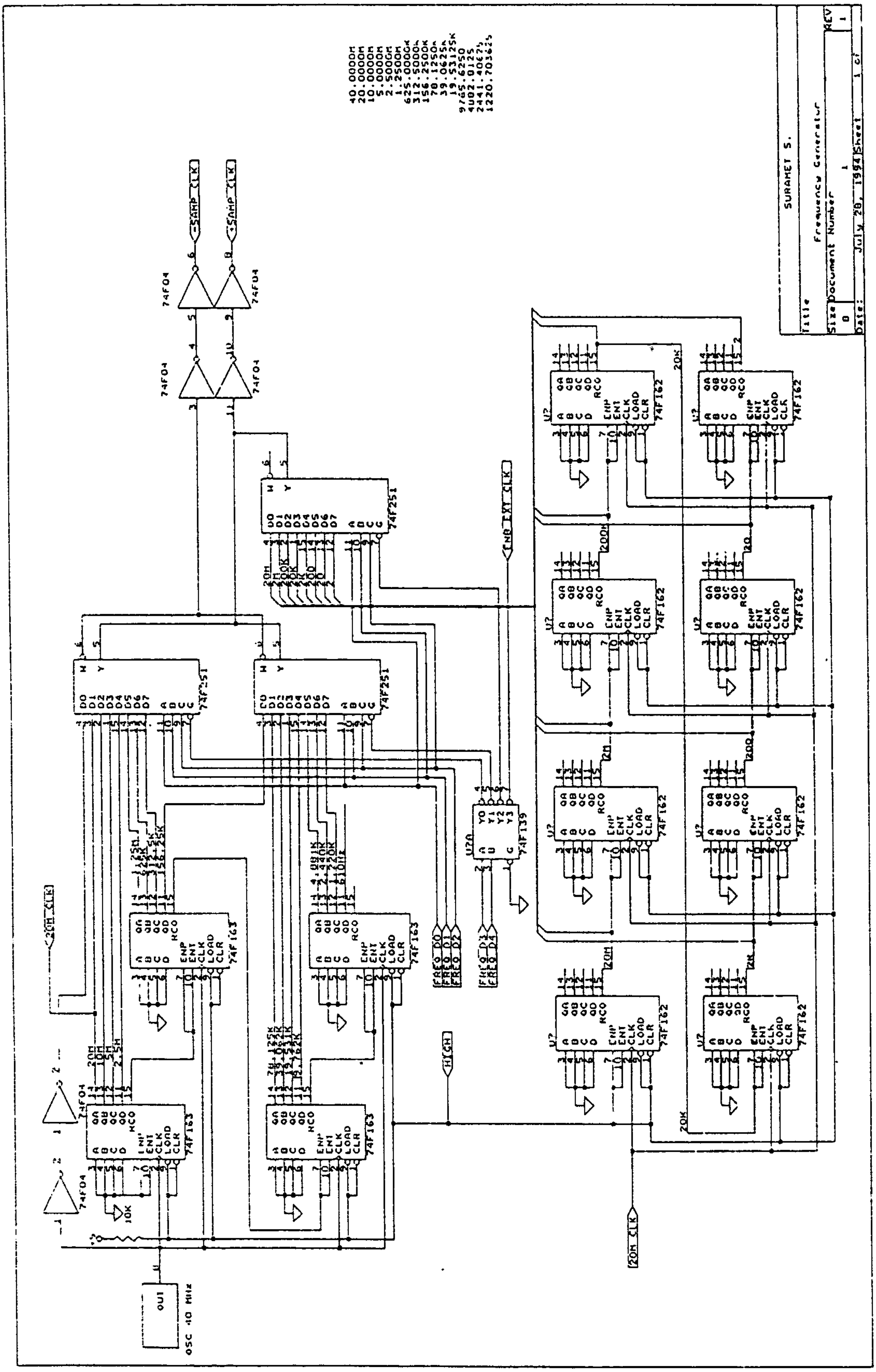
การเดินลายวงจร สำหรับระบบความเร็วสูง การเหนี่ยวนำระหว่างลายวงจรทำให้เกิดสัญญาณรบกวนและปัญหาอื่นๆอีกมาก จึงจำเป็นที่จะต้องจัดวางอุปกรณ์บนแผ่นวงจรพิมพ์เพื่อให้ได้การเดินลายทองแดงสั้นที่สุดสำหรับวงจรดิจิตอล ทำให้ไม่สร้างสัญญาณรบกวนกับระบบอนาลอก และต้องระวังไม่ให้มีการข้ามกันระหว่างลายวงจรของส่วนดิจิตอลและส่วนอนาลอก อีกทั้งจะต้องเลือกใช้ชนิดของแผ่นวงจรพิมพ์ให้เหมาะสม เนื่องจากชนิดของซับสเตรทของแผ่นวงจรพิมพ์แต่ละชนิดมีคุณสมบัติแตกต่างกัน ซึ่งมีผลต่อระบบทั้งหมด

สำหรับการพัฒนาระบบเก็บข้อมูลนั้นควรปรับปรุงในส่วนอุปกรณ์ที่ใช้ให้มีการเดินลายวงจรน้อยลงเช่น เปลี่ยนจาก 74F374 เป็น 74F574 ซึ่งแยกกลุ่มของสัญญาณเข้าออกเป็นซ้ายขวา ทำให้การเดินลายวงจรทำได้ง่าย และลดความซับซ้อนลงเป็นต้น สำหรับในส่วนของซอฟท์แวร์เนื่องจากในวิทยานิพนธ์นี้ได้เน้นในทางการศึกษาและออกแบบสร้างหลักด้านฮาร์ดแวร์ จึงได้พัฒนาซอฟท์แวร์เน้นเฉพาะในส่วนที่เป็นส่วนเชื่อมต่อกับระบบฮาร์ดแวร์ แต่ได้ออกแบบซอฟท์แวร์ส่วนอื่นไว้เป็นเค้าโครงสำหรับผู้ที่ต้องการพัฒนาต่อไป ดังนั้นในการพัฒนาควรเพิ่มสำหรับโปรแกรมย่อยเพื่อการวิเคราะห์ข้อมูลทางด้านต่างๆเช่น การประมวลสัญญาณเชิงเลข การคำนวณทางสถิติ การคำนวณทางคณิตศาสตร์ การแสดงภาพเป็นกราฟฟิกสามมิติ ซึ่งสิ่งเหล่านี้จะช่วยในระบบเก็บข้อมูลทำงานได้สมบูรณ์และตรงกับการใช้งานมากขึ้น สิ่งที่ควรปรับปรุงในระบบส่วนซอฟท์แวร์ คือการปรับปรุงการใช้หน่วยความจำของระบบให้ดีขึ้น โดยการปรับวิธีการเรียกโปรแกรมสำหรับทำงานต่างๆ จากเดิมที่รวมไว้ในตัวโปรแกรมหลักเพียงโปรแกรมเดียว เป็นการแยกไว้เป็นโมดูลย่อยไว้ในหน่วยความจำสำรอง เช่น ฮาร์ดดิสก์ เมื่อต้องการใช้โปรแกรมส่วนนั้นๆ ก็ทำการอ่านเข้าสู่หน่วยความจำเพื่อให้ทำงาน แล้วยกเลิกส่วนที่อยู่ในหน่วยความจำทิ้งไป เพื่อให้โปรแกรมวิเคราะห์อื่นๆเข้ามาต่อไป ซึ่งวิธีการแบบนี้บนระบบปฏิบัติการ DOS เรียกว่าการใช้ Overlay ในขณะที่บนระบบปฏิบัติการไมโครซอฟท์วินโดวส์จะเรียกว่า DLL (Dynamic Link Library) ซึ่งช่วยในการจัดการหน่วยความจำระบบซอฟท์แวร์ที่ออกแบบขึ้นทำได้สะดวกและโอกาสผิดพลาดลดลงไป

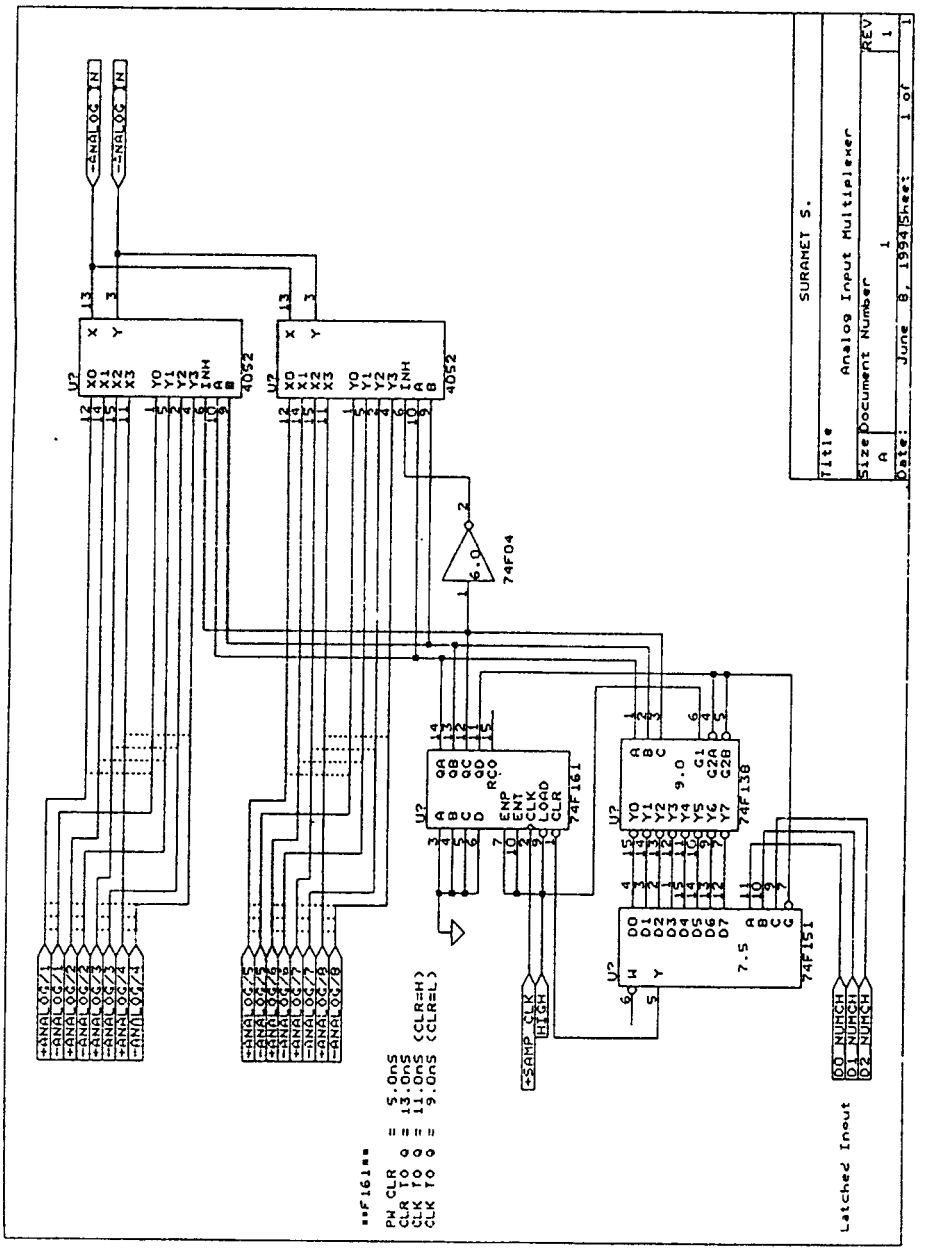
ถึงแม้ว่าวิทยานิพนธ์นี้จะยังมีข้อบกพร่องอยู่หลายจุด แต่ก็สามารถเป็นต้นแบบและแนวทางสำหรับการสร้างระบบเก็บและวิเคราะห์ข้อมูลที่มีความเร็วสูงต่อไปในอนาคต

ภาคผนวก ก

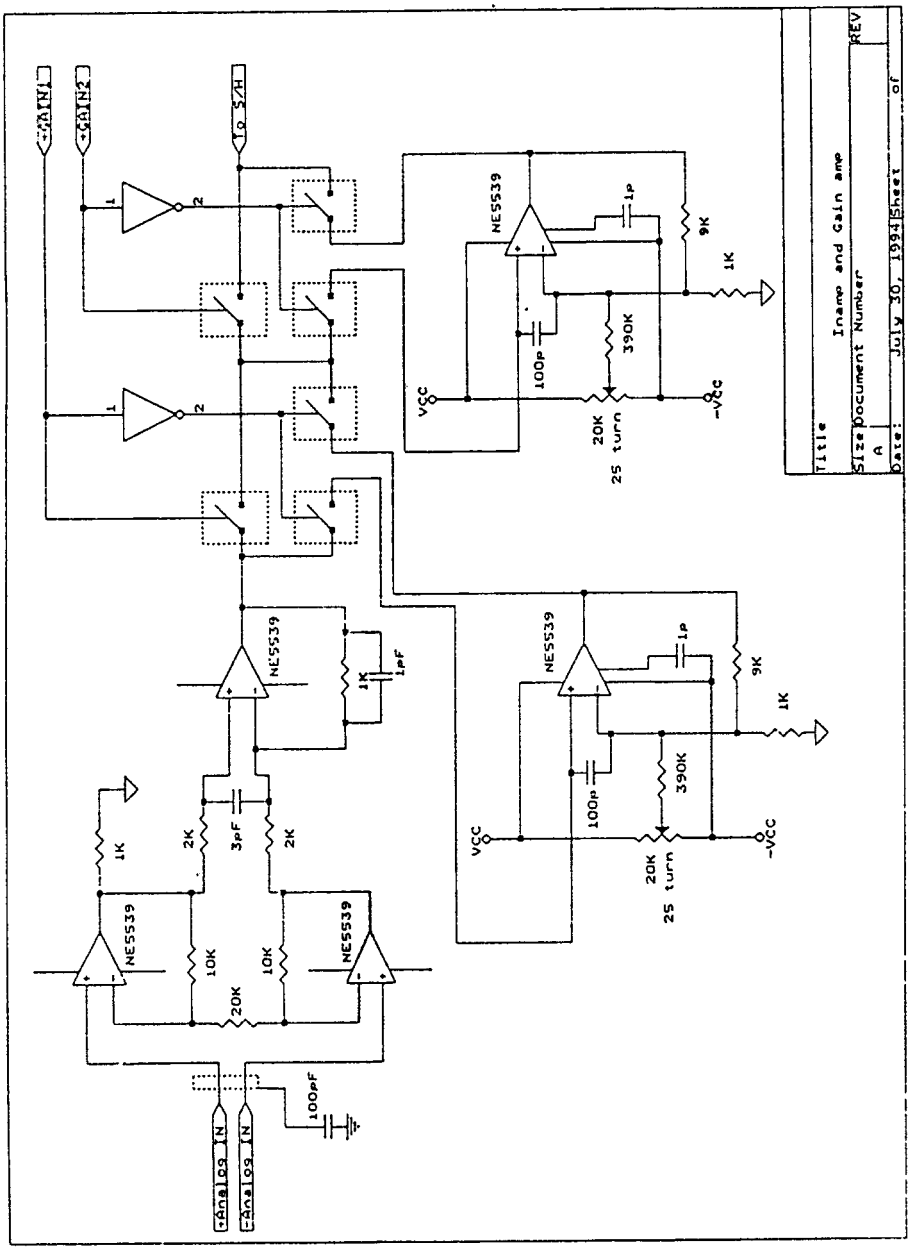
วงจรทั้งหมดที่ใช้ในระบบ



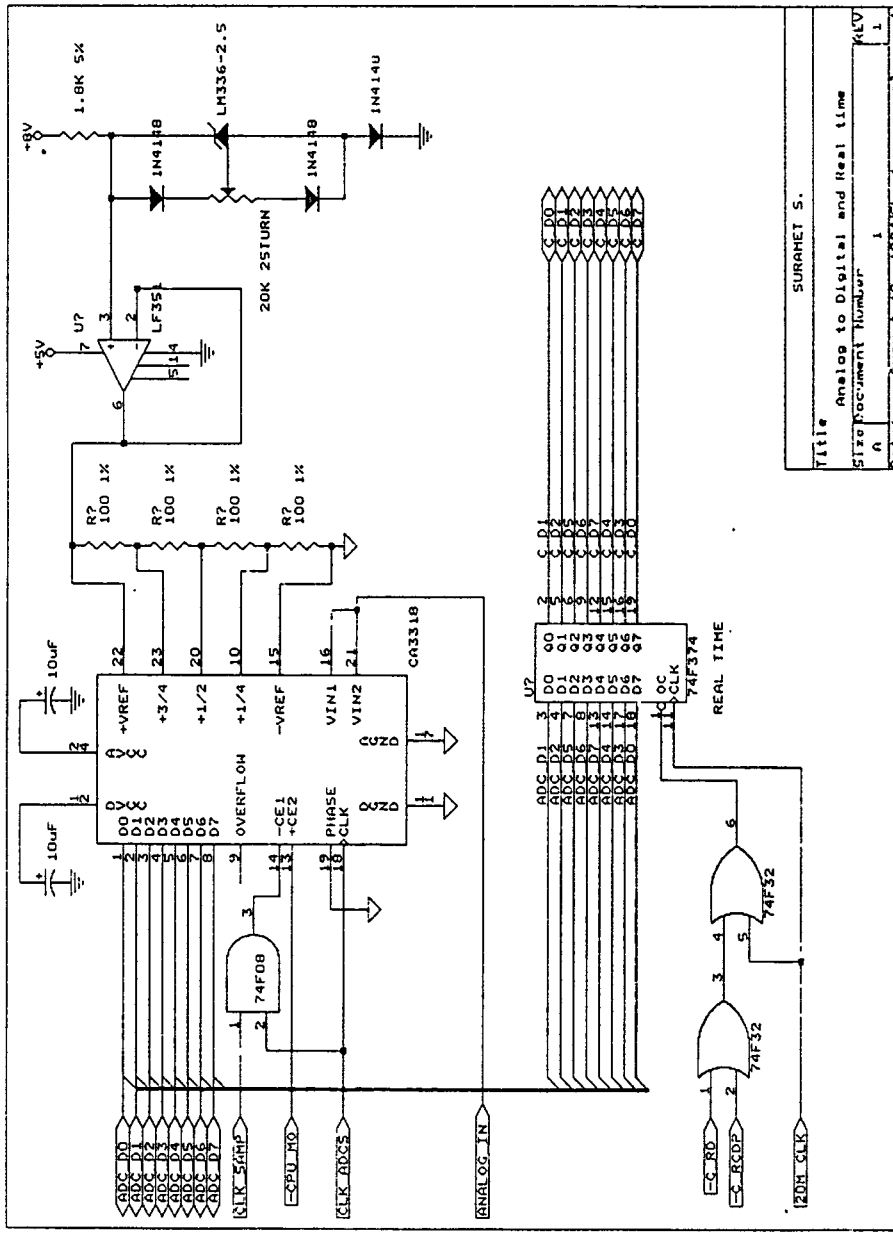
รูป ก.1 วงจรกำเนิดสัญญาณนาฬิกา



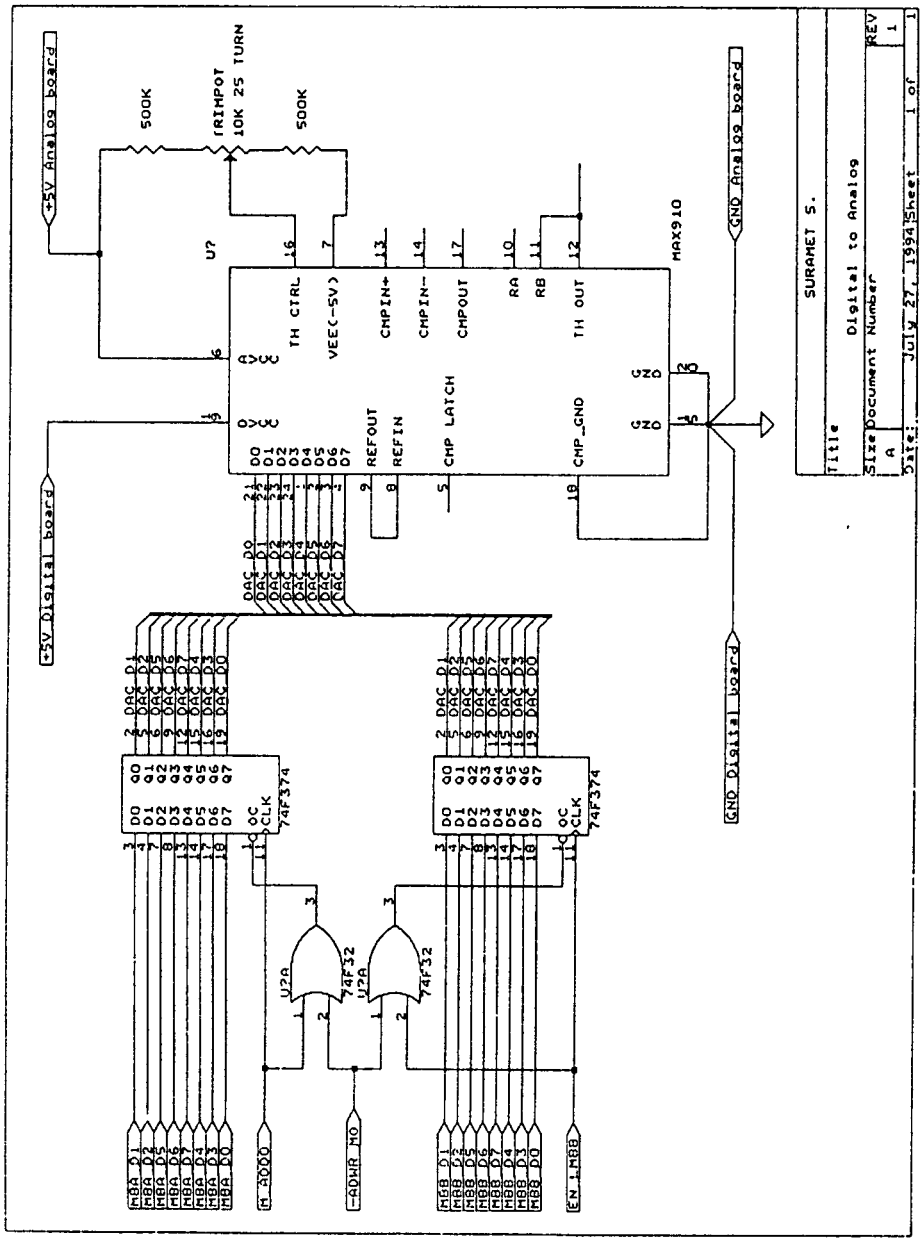
รูป ก.2 วงจรช่องสัญญาณเข้าและส่วนควบคุม



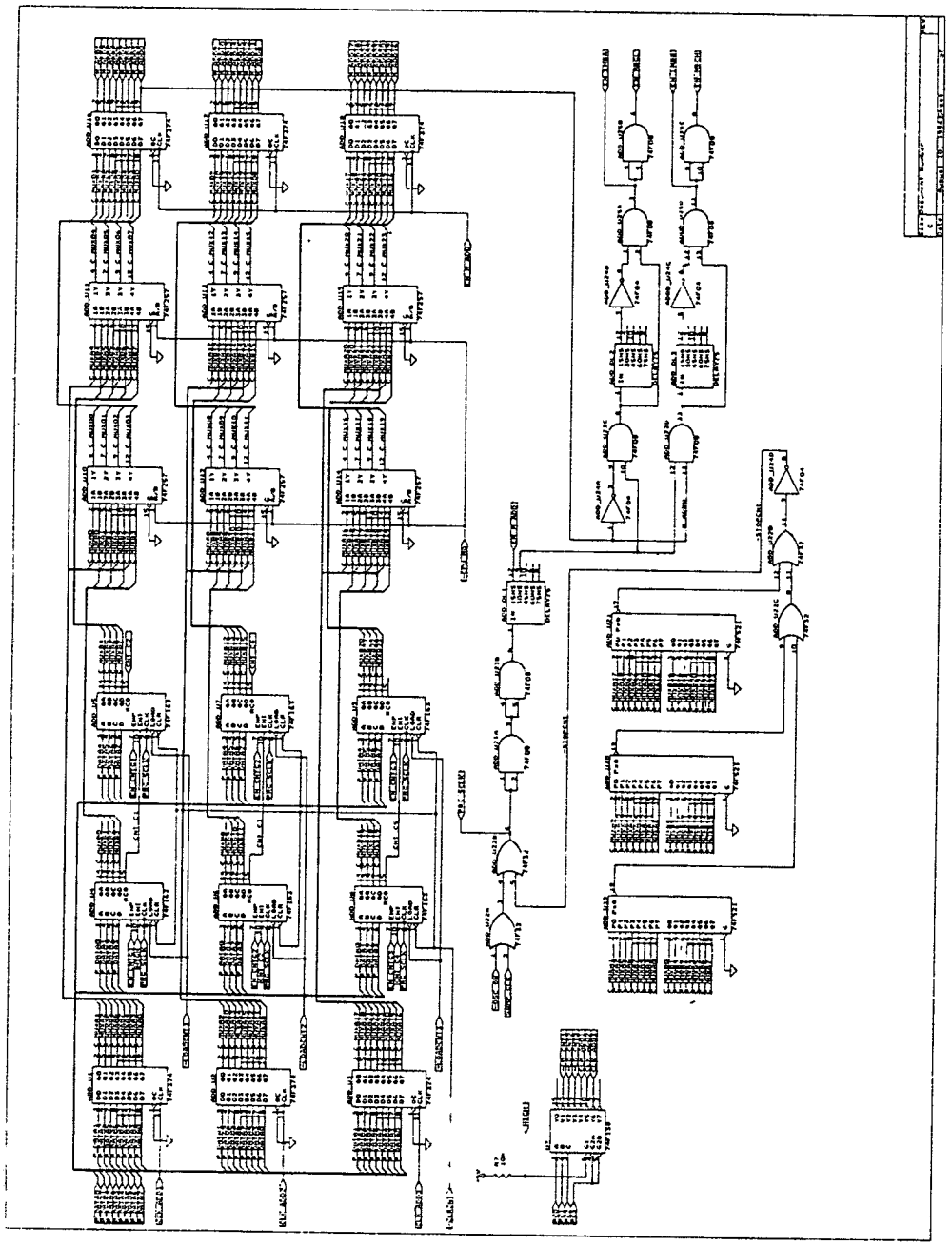
รูป ก.3 วงจรขยายแบบอินสทรูเมนต์และวงจรขยายกำหนดการขยายได้



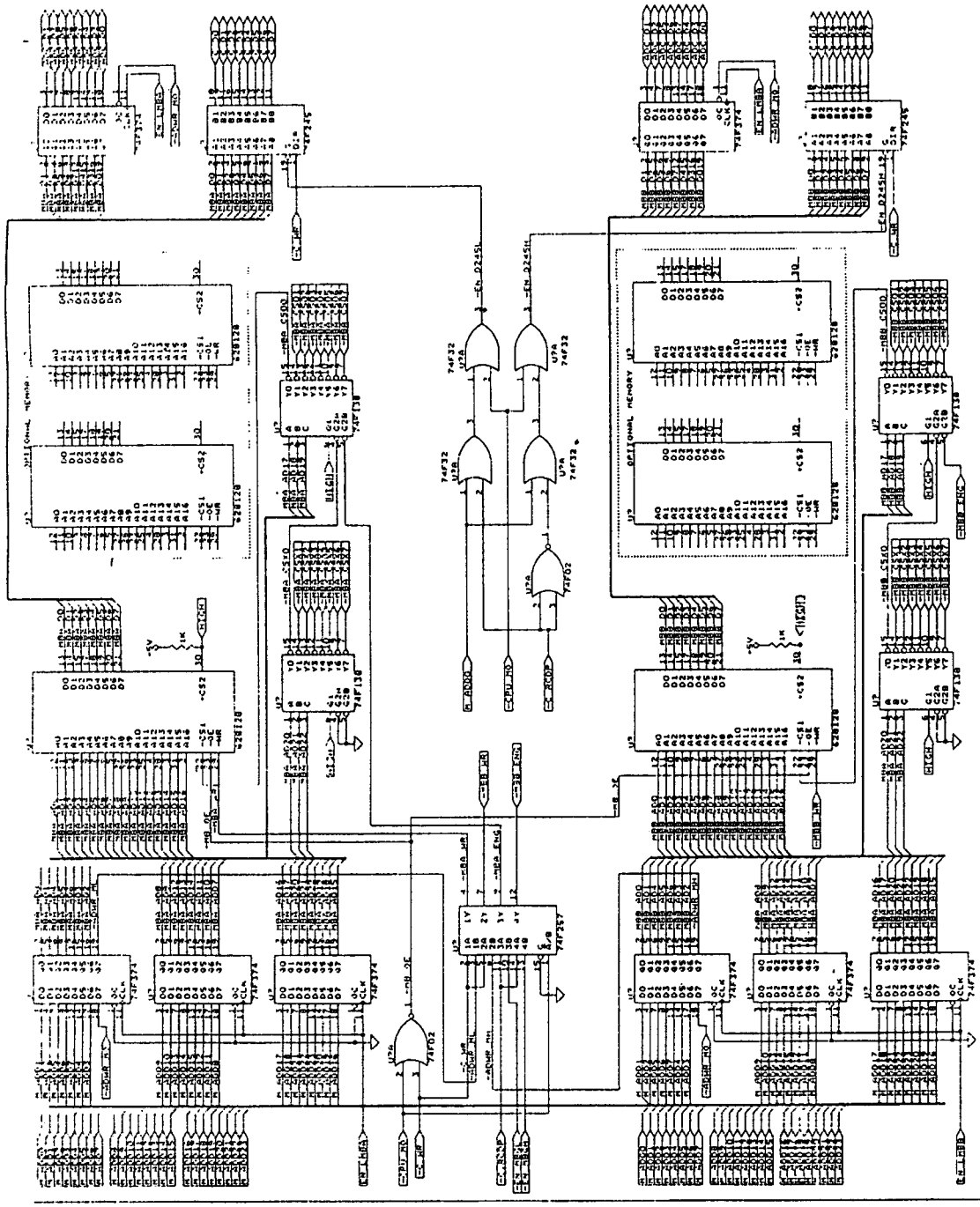
รูป ก.4 วงจรแปลงข้อมูลอนาลอกเป็นดิจิตอล



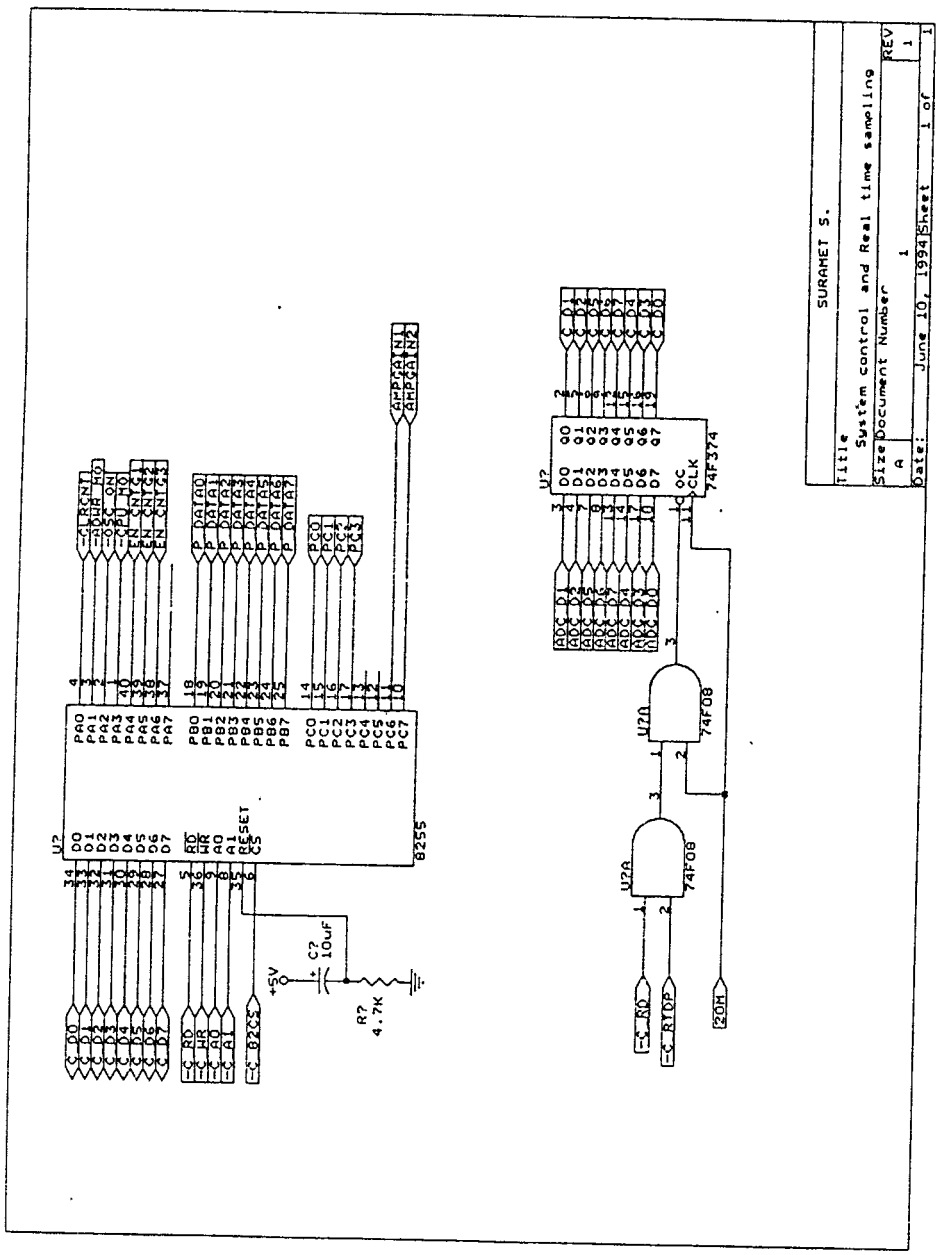
รูป ก.5 วงจรแปลงข้อมูลดิจิตอลเป็นอนาลอก



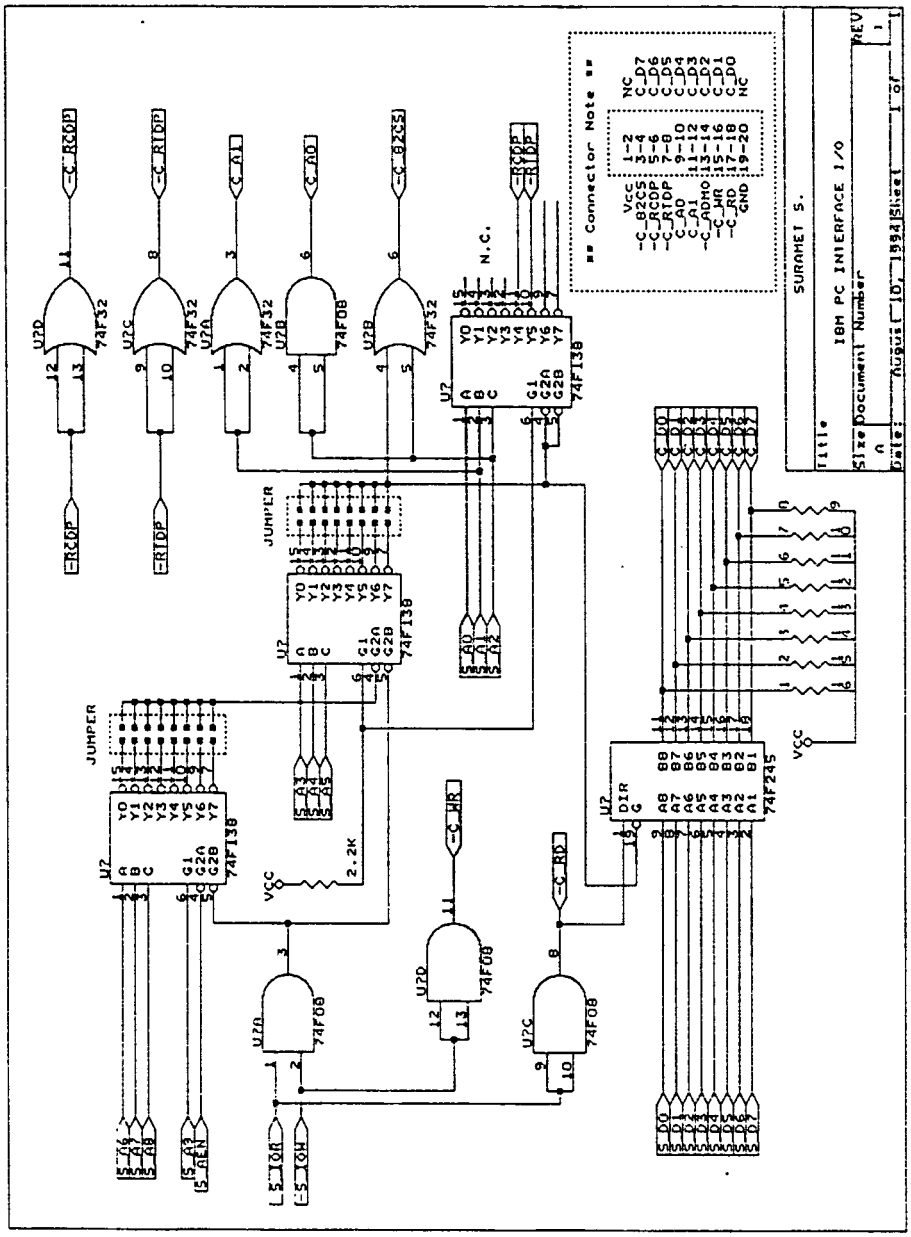
รูป ก.6 วงจรกำเนิดและคงค้างตำแหน่งหน่วยความจำและส่วนควบคุมการอ่านเขียนที่ความเร็วสูง



รูป ก.7 วงจรหน่วยความจำพร้อมส่วนบัฟเฟอร์และส่วน decode



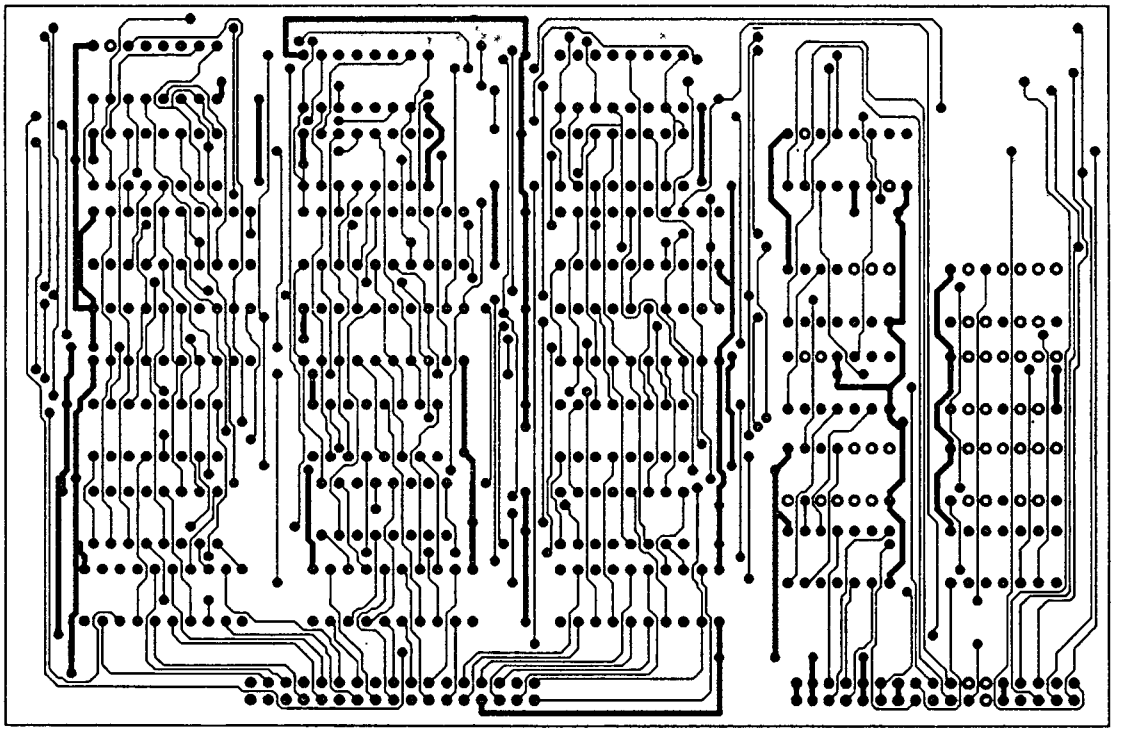
รูป ก.8 วงจรควบคุมการสั่งการระบบ



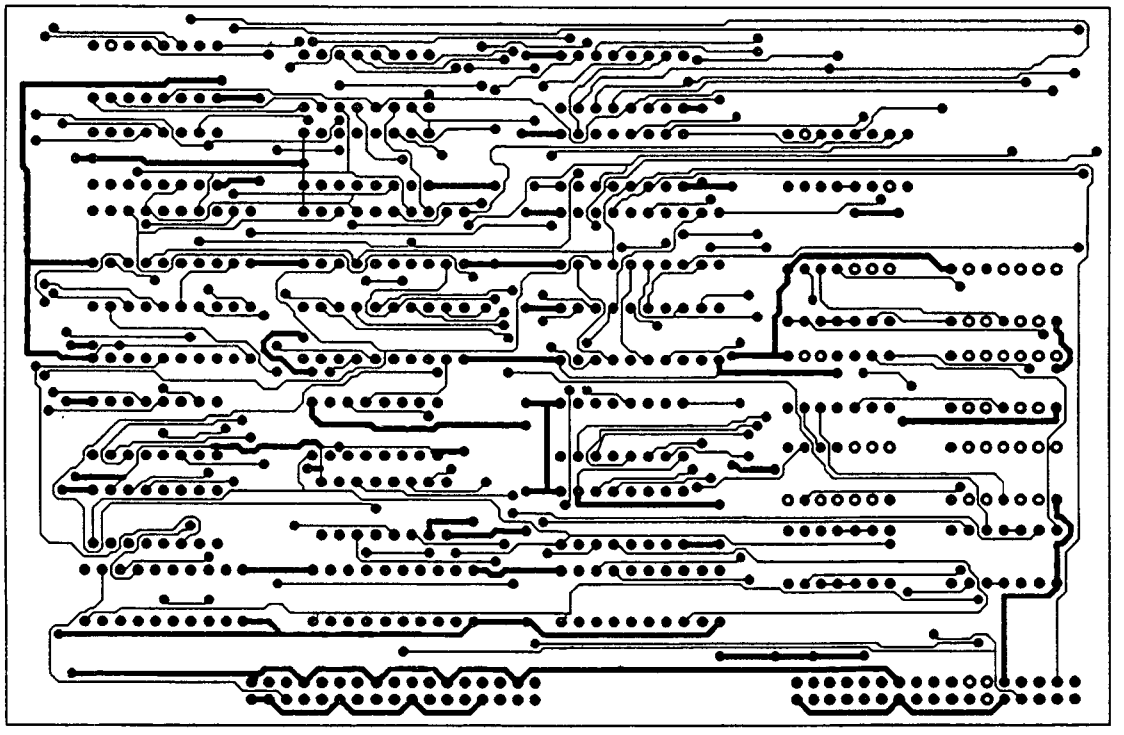
รูป ก.9 วงจรเชื่อมต่อกับระบบคอมพิวเตอร์

ภาคผนวก ข

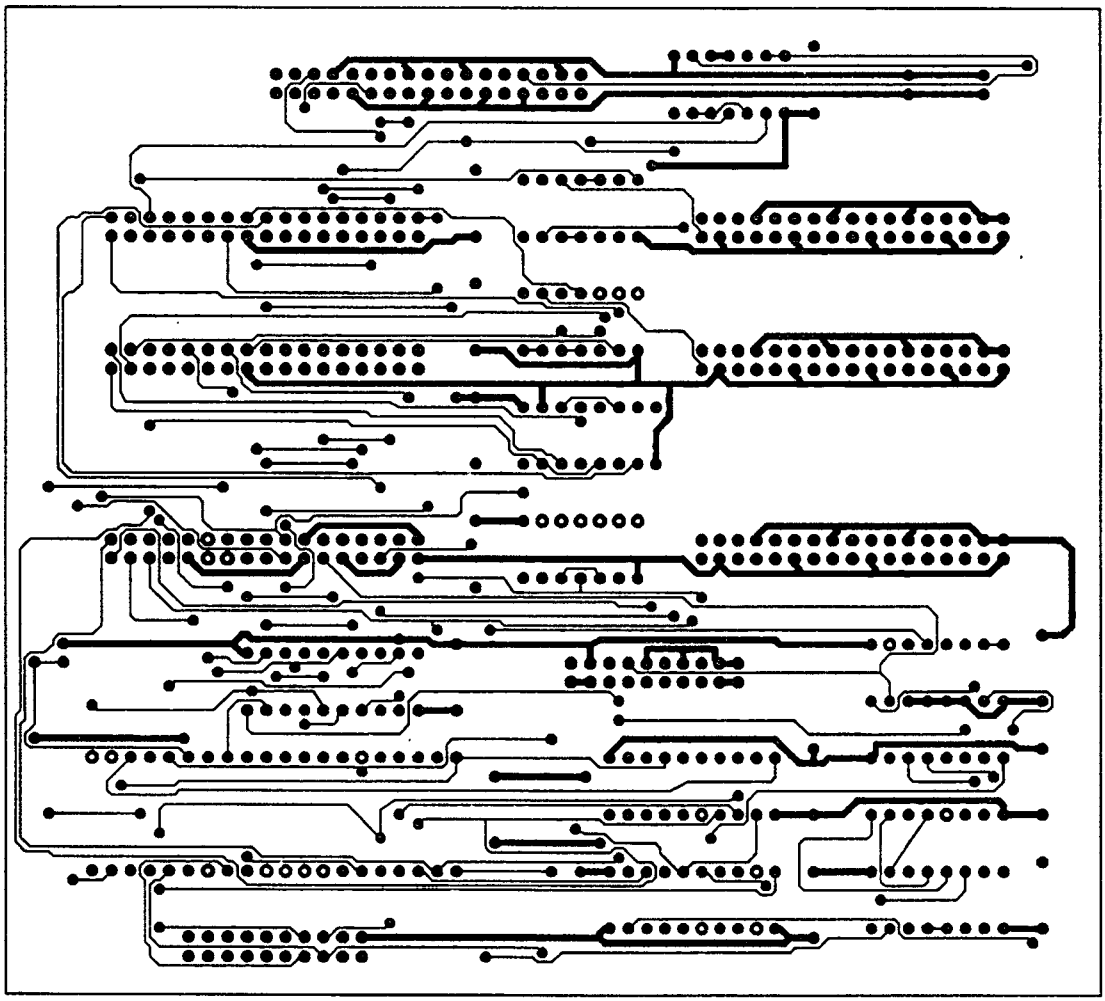
ลายวงจรพิมพ์ที่ใช้ในระบบ



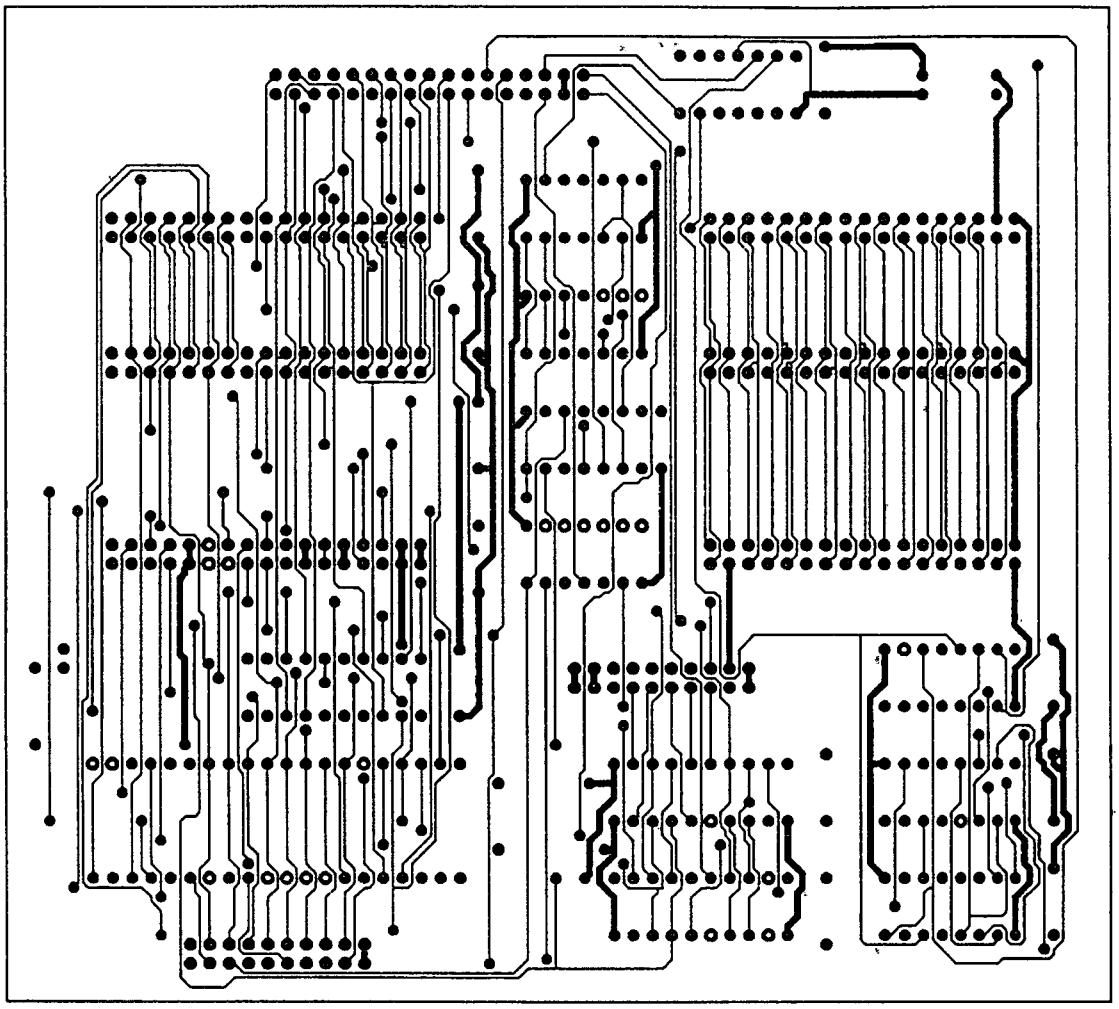
รูป ข.1 ลายวงจรพิมพ์ส่วนกำเนิดสัญญาณตำแหน่งหน่วยความจำและวงจรควบคุม ด้านอุปกรณ์
(เท่าของจริง)



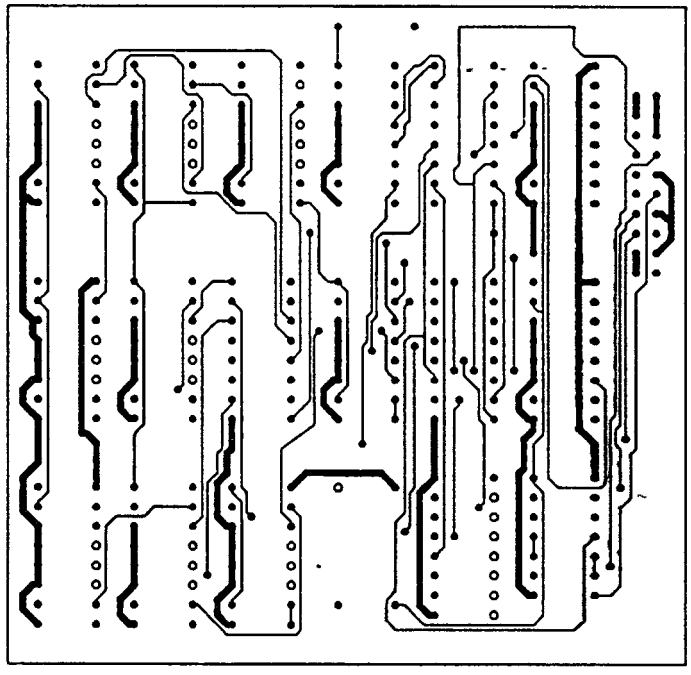
รูป ข.2 ลายวงจรพิมพ์ส่วนกำเนิดสัญญาณตำแหน่งหน่วยความจำและวงจรควบคุม ด้านบัดกรี
(เท่าของจริง)



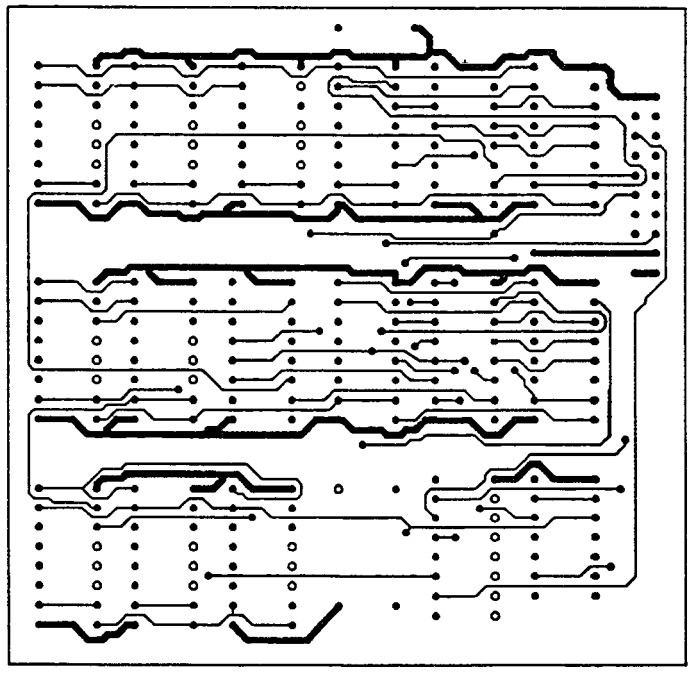
รูป ข.3 ลายวงจรพิมพ์ส่วนแผ่นวงจรหลัก ด้านอุปกรณ์ (เท่าของจริง)



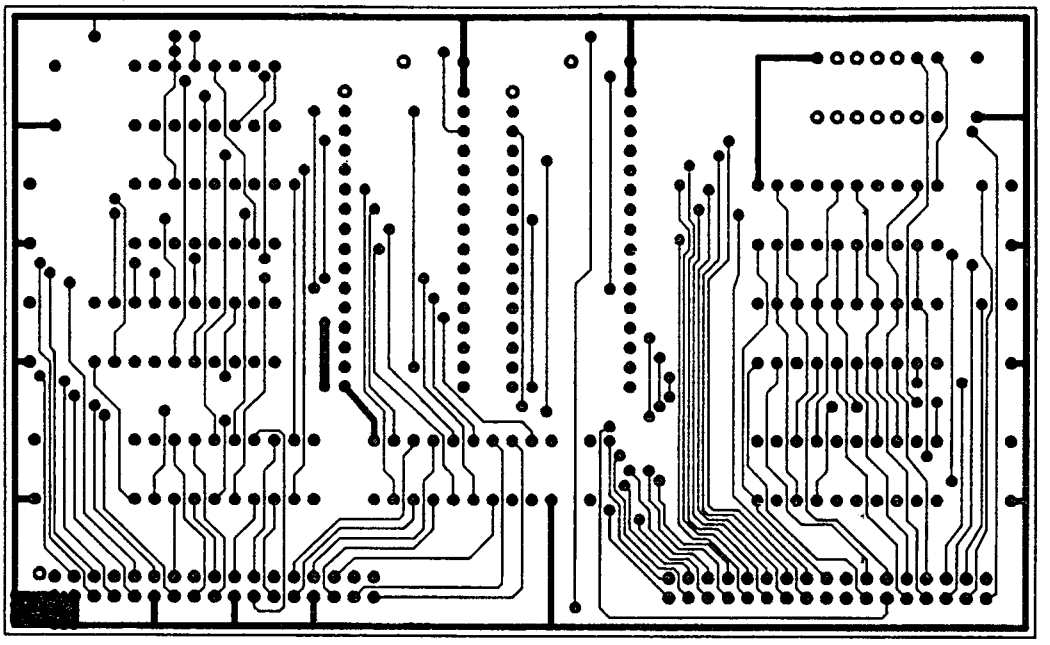
รูป ข.4 ลายวงจรพิมพ์ส่วนแผ่นวงจรหลัก ด้านบัดกรี (เท่าของจริง)



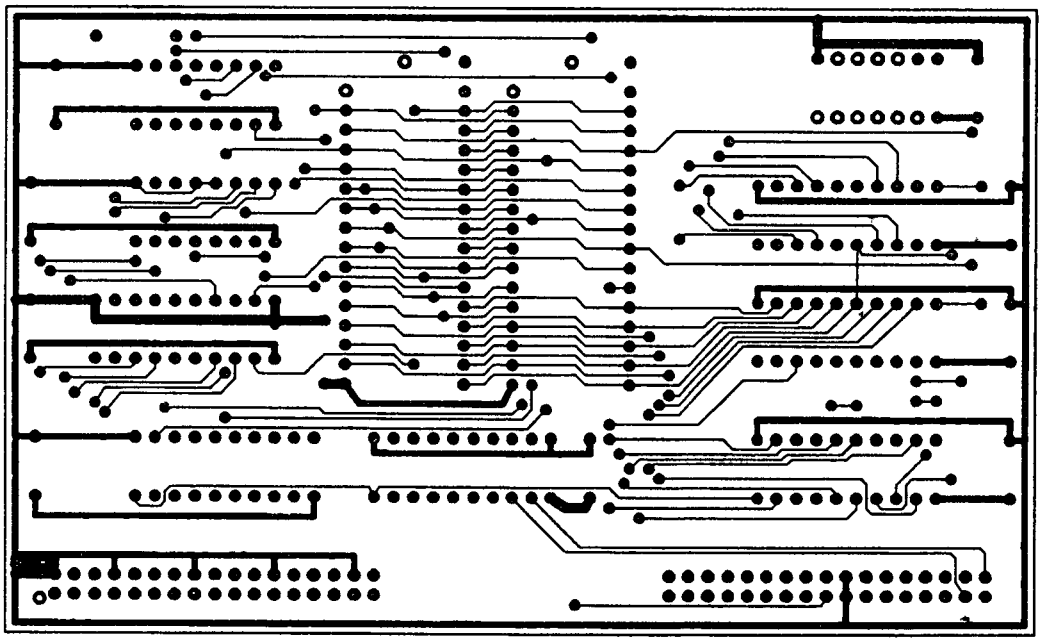
รูป ข.5 ลายวงจรพิมพ์ส่วนกำเนิดความถี่ ด้านอุปกรณ์ (เท่าของจริง)



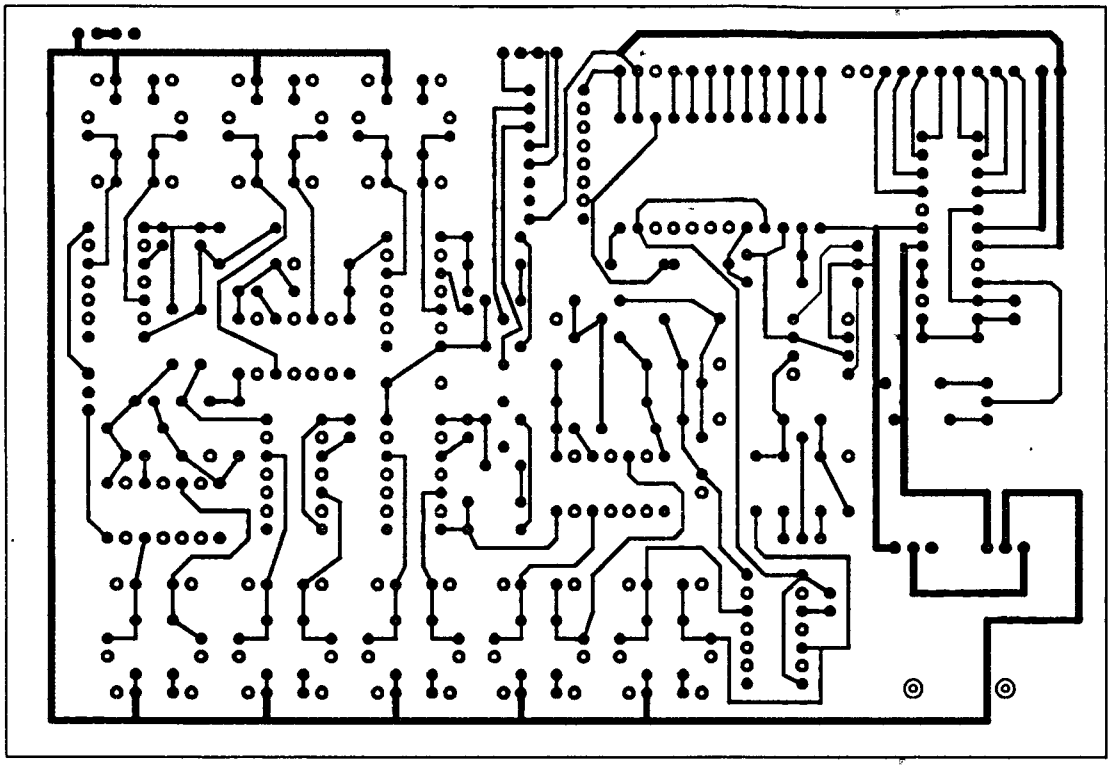
รูป ข.6 ลายวงจรพิมพ์ส่วนกำเนิดความถี่ ด้านบัดกรี (เท่าของจริง)



รูป ข.7 ลายวงจรพิมพ์ส่วนหน่วยความจำ ด้านอุปกรณ์ (เท่าของจริง)



รูป ข.8 ลายวงจรพิมพ์ส่วนกำเนิดความถี่ ด้านบัดกรี (เท่าของจริง)



รูป ข.9 ลายวงจรพิมพ์ส่วนอนาลอก ด้านบัดกรี (เท่าของจริง)

ภาคผนวก ค

โปรแกรมของระบบที่พัฒนาบนไมโครซอฟท์วินโดว์ส

```
#include <windows.h>
#include "hardware.h"
#include <bwcc.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>
#include <drivinit.h>
#include <print.h>

#include "chromicw.h"
#include "c_defvar.h"

/*-----------------PROTOTYPE DEFINE-------------*/
extern BOOL FAR PASCAL DlgProc_help_about (HWND,WORD,WORD,LONG);
extern BOOL FAR PASCAL DlgProc_file_open  (HWND,WORD,WORD,LONG);
extern BOOL FAR PASCAL DlgProc_name_new   (HWND,WORD,WORD,LONG);
extern BOOL FAR PASCAL DlgProc_name_edit  (HWND,WORD,WORD,LONG);
extern BOOL FAR PASCAL DlgProc_name_real  (HWND,WORD,WORD,LONG);
extern BOOL FAR PASCAL DlgProc_name_select(HWND,WORD,WORD,LONG);
extern BOOL FAR PASCAL CloseEnumProc      (HWND,LONG);
BYTE ReadIO(LONG Adddress);
void ClearCounter(void);
void SetCounter(LONG Address);
void StopCounter(LONG Address);
void StartSampling(void);
void EndSampling(void);
void SetFreq(int freq);
void ResetCard(void);
void StartPlayback(void);
void StopPlayback(void);
BYTE ReadRealtime(void);
void SetChannel(int num);

long GraphRefresh(HWND,WORD,LONG);
long DisplayGraph(GRAPHPARA);

/* COMPLEX STRUCTURE*/

typedef struct {
   float real, imag;
} COMPLEX;

/* function prototypes for dft and inverse dft functions */

extern void fft(COMPLEX far *,int);
extern void ifft(COMPLEX far*,int);

/*-----------------WINDOWS STATIC DATA-------------*/
HANDLE hInst;
HWND   hD_real_set,hD_help_about;
HMENU  hMenuMainWin, hMenuGraphWin, hMenuMain, hMenuGraph;
int    State,nTimer,num;
HWND   hWndGraph;
HWND   hWndClient;
MDICREATESTRUCT          MDIStructVar;

/*----------------- CHROMICS GLOBAL VARIABLE----------*/
extern long  RtMemStart=0,RtMemEnd=0;
extern char  RtChannel=1;

/*----------------- Temporary Variable for Test----------*/
extern char *Name[10]={ "Name-0","Name-1","Name-2","Name-3","Name-4",..
                        "Name-5","Name-6","Name-7","Name-8","Name-9"};
NAMESTRUCT     MainVar[255];
GRAPHSTRUCT    Mg[255];
//HANDLE                 Data[255];
unsigned int far *pData,*pData2;
COMPLEX          far          *fData;

/*------ Dialog Procedure include -- .CDP = C of DlgProc --------*/
#include "c_name_n.cdp"
#include "c_name_e.cdp"
#include "c_name_s.cdp"
#include "c_real_s.cdp" .
//#include "c_file_o.cdp"
```

```
/*------ Window Procedure include -- .CWP = C of Window Proc ----------*/
#include "c_misc.cwp"
//#include "c_graph.cwp"
#include "c_fft.c"

/*------------------ Define Proc ----------------*/
long FAR PASCAL ChromicWndProc( HWND hWnd,WORD msg,WORD wParam,LONG lParam);
long FAR PASCAL ChromicGraphProc( HWND, WORD, WORD, LONG);

int  FAR PASCAL sInitSysMemo();
int  FAR PASCAL sDrawOneGraph(HWND);
int  FAR PASCAL sPrinterSetupProc( HWND,WORD,WORD,LONG);

long sGraphDisplay(GRAPHPARA);
/*------------------ Main Program ----------------*/
int PASCAL WinMain (HANDLE hInstance, HANDLE hPrevInstance,
                    LPSTR  lpszCmdLine, int cmdShow)
{
  HWND    hwnd;
  MSG     msg;
  WNDCLASS wndclass;
  HANDLE   hMainAcc,hWndClient;
  int a;

  MainVar[0].Use=1;
  strcpy(MainVar[0].Name,Name[0]);
  MainVar[0].Start = 0;
  MainVar[0].End = 1024;
  MainVar[0].Ch = 1;
  MainVar[0].Freq = 0;


  if (!hPrevInstance)
   { wndclass.lpszClassName = "ChromicWin";
         wndclass.hInstance     = hInstance;
         wndclass.lpfnWndProc   = (WNDPROC)ChromicWndProc;
         wndclass.hCursor       = LoadCursor (NULL, IDC_ARROW);
         wndclass.hIcon         = LoadIcon (hInstance, "Chromic_i_sys");
         wndclass.lpszMenuName  = NULL;                          // MainMenu
         wndclass.hbrBackground = COLOR_APPWORKSPACE+1;
         wndclass.style         = CS_HREDRAW | CS_VREDRAW;
         wndclass.cbClsExtra    = 0;
         wndclass.cbWndExtra    = 0;
         RegisterClass(&wndclass);

         wndclass.lpszClassName = "ChromicGraph";
         wndclass.hInstance     = hInstance;
         wndclass.lpfnWndProc   = (WNDPROC)ChromicGraphProc;
         wndclass.hCursor       = LoadCursor (NULL, IDC_CROSS);
         wndclass.hIcon         = NULL;
         wndclass.lpszMenuName  = NULL;                          //GraphMenu;
         wndclass.hbrBackground = GetStockObject(LTGRAY_BRUSH);
         wndclass.style         = CS_HREDRAW | CS_VREDRAW;
         wndclass.cbClsExtra    = 0;
         wndclass.cbWndExtra    = sizeof(WORD);
         RegisterClass(&wndclass);
   }

  hInst = hInstance;

  hMenuMain  = LoadMenu(hInst,(LPSTR)MenuMain);
  hMenuGraph = LoadMenu(hInst,(LPSTR)MenuGraph);
  /*------ Get handle of submenu in Specify Menu --------
        like handle of file in hMenuMain              */
  hMenuMainWin  = GetSubMenu( hMenuMain , 0);
  hMenuGraphWin = GetSubMenu( hMenuGraph, 0);



  hwnd = CreateWindow("ChromicWin","Chromic for Windows 1.0 (Beta Test)",
                      WS_OVERLAPPEDWINDOW | WS_CLIPCHILDREN,
                      CW_USEDEFAULT,CW_USEDEFAULT,CW_USEDEFAULT,CW_USEDEFAULT,
                      NULL,hMenuMain,hInstance,NULL );
```

```
  hWndClient = GetWindow (hwnd,GW_CHILD);
  ShowWindow (hwnd, cmdShow);
  UpdateWindow(hwnd);

/*  hMainAcc = LoadAccelerators (hInstance, "KeyTable"); */

  while (GetMessage(&msg,NULL,0,0))
   { if (!TranslateMDISysAccel (hWndClient,&msg))
    { TranslateMessage(&msg);
          DispatchMessage (&msg);
    }
   }
  return msg.wParam;
}


long FAR PASCAL ChromicWndProc ( HWND hWnd, WORD msg, WORD wParam, LONG lParam)
{
 FARPROC lpPD_inst_tmp, lpfnEnum;
 static HANDLE hBwccLib;
 COMPLEX far     *pTemp;
 static HWND   hWndTemp;
 int                i,a,j,k;
 static int offset;
 RECT               rect;
 char               cBuf[128],Title[128];
 BYTE               Value;
 LONG               num1;
 float              sd;

 CLIENTCREATESTRUCT    ClientStructVar;

 switch (msg)
 {
  case WM_CREATE:
   if( (hBwccLib = LoadLibrary("BWCC.LIB")) < 32 )
        {
         MessageBox(hWnd,"Can't Open BWCC.DLL"," ERROR",MB_ICONHAND);
    DestroyWindow(hWnd);
        }
   ClientStructVar.hWindowMenu  = hMenuMainWin;
   ClientStructVar.idFirstChild = M_child_first;

   hWndClient = CreateWindow("MDICLIENT",NULL,WS_CHILD|WS_CLIPCHILDREN|WS_VISIBLE,
                             0,0,0,0, hWnd, 1,hInst, (LPSTR)&ClientStructVar);
   offset = 0;
   return 0;

  case WM_DESTROY:
   FreeLibrary(hBwccLib);
   PostQuitMessage(0);
   return 0;

  case WM_TIMER:
   switch(State)
   {
        case 3:
         KillTimer(hWnd,nTimer);
         if(Mg[num].Mem = GlobalAlloc(GMEM_FIXED,1024*sizeof(*pData)))
    {
          if(pData = GlobalLock(Mg[num].Mem))
          {
/*                  outportb(0x301,1);
                    outportb(0x300,0x4f);
                    outportb(0x300,0x0f);
                    outportb(0x300,0x0a);
 */         outportb(0x300,3);
                    for(i=0;i<1024;i++)
         {
                         *pData = (UINT)(ReadIO((LONG)(MainVar[num].Start+i)));
                         pData++;
                    }
                    outportb(0x300,0x0f);
         GlobalUnlock(Mg[num].Mem);
          }
```

```
            else
                    MessageBox(hWnd,"Cannot Lock Memory !!","Message",MB_ICONEXCLAMATION|MB_OK);
          }
          else
            MessageBox(hWnd,"Cannot Allocates Memory !!","Message",MB_ICONEXCLAMATION|MB_OK);
          MessageBox(hWnd,"Sampling Complete !!","Message",MB_ICONEXCLAMATION|MB_OK);
          State = 0;
          break;
        case 5:
          if(pData = GlobalLock(Mg[num].Mem))
          {
                  if(offset >= 1024)
        {
                    KillTimer(hWnd,nTimer);
                    InvalidateRect(hWndGraph,NULL,TRUE);
                    MessageBox(hWnd,"Realtime Complete !!","Message",MB_ICONEXCLAMATION|MB_OK);
                    State = 0;
                    offset = 0;
                  }
                  else
        {
                    for(i=0;i<offset;i++)
                      pData++;
                    *pData = (UINT)(ReadRealtime());
                    offset++;
                  }
                  GlobalUnlock(Mg[num].Mem);
                  GetClientRect(hWndGraph,&rect);
                  rect.left = rect.left+(offset*0.95);
                  rect.right = rect.left + 10;
        if(offset%10 == 0)
                    InvalidateRect(hWndGraph,&rect,TRUE);
          }
          else
            MessageBox(hWnd,"Cannot Lock Memory !!","Message",MB_ICONEXCLAMATION|MB_OK);
          break;
    }
    return 0;

   case WM_COMMAND:
    switch(wParam)
    {
          /*-------- Service File Menu ---------*/
 /*       case M_file_new:
          hWndTemp = LOWORD(SendMessage(hWndClient,WM_MDIGETACTIVE,0.0L));
          sDrawOneGraph(hWndTemp);
          return 0;
        case M_file_open:
          lpPD_inst_tmp= MakeProcInstance(DlgProc_file_open,hInst);
          DialogBox (hInst,MAKEINTRESOURCE(D_file),hWnd,lpPD_inst_tmp);
          FreeProcInstance(lpPD_inst_tmp);
          return 0;
        case M_file_save:
          return 0;
        case M_file_save_as:
          return 0;
        case M_file_prn:
          return 0;
        case M_file_prn_setup:
          sPrinterSetupProc( hWnd, msg, wParam, lParam);
          return 0;*/
        case M_file_exit:
          SendMessage(hWnd,WM_CLOSE,0,0L);
          return 0;


/*        case M_name_disp:
          //Call DialogBox to get name to display.  return value=name handle
      // Keep handle
          MDIStructVar.szClass  = "ChromicGraph";
          MDIStructVar.szTitle  = "Graph of Data";
          MDIStructVar.hOwner   = hInst;
          MDIStructVar.x     = CW_USEDEFAULT;
          MDIStructVar.y     = CW_USEDEFAULT;
          MDIStructVar.cx    = CW_USEDEFAULT;
```

```
        MDIStructVar.cy    = CW_USEDEFAULT;
        MDIStructVar.style    = 0;
        MDIStructVar.lParam   = NULL;
        hWndGraph = SendMessage(hWndClient, WM_MDICREATE, 0,
                                (LONG)(LPMDICREATESTRUCT)&MDIStructVar);
    //Set name handle to WindowWord para for ref. to data
        SetWindowWord(hWndGraph,0,1);
        return 0;
*/
      case M_name_new:
        lpPD_inst_tmp= MakeProcInstance(DlgProc_name_new,hInst);
        DialogBox (hInst,MAKEINTRESOURCE(D_name_new),hWnd,lpPD_inst_tmp);
        FreeProcInstance(lpPD_inst_tmp);
        return 0;

      case M_name_edit:
        lpPD_inst_tmp= MakeProcInstance(DlgProc_name_edit,hInst);
        DialogBox (hInst,"D_name_edit",hWnd,lpPD_inst_tmp);
        FreeProcInstance(lpPD_inst_tmp);
        return 0;

      case M_name_release:
    State = 1;
        lpPD_inst_tmp= MakeProcInstance(DlgProc_name_select,hInst);
        DialogBox (hInst,"D_name_select",hWnd,lpPD_inst_tmp);
        FreeProcInstance(lpPD_inst_tmp);
        return 0;

      case M_name_display:
        State = 2;
        lpPD_inst_tmp= MakeProcInstance(DlgProc_name_select,hInst);
        DialogBox (hInst,"D_name_select",hWnd,lpPD_inst_tmp);
        FreeProcInstance(lpPD_inst_tmp);
        return 0;

      case M_adc_sampling:
        State = 3;
        lpPD_inst_tmp= MakeProcInstance(DlgProc_name_select,hInst);
        DialogBox (hInst,"D_name_select",hWnd,lpPD_inst_tmp);
        FreeProcInstance(lpPD_inst_tmp);
        return 0;

      case M_dac_playback:
        State = 4;
        lpPD_inst_tmp= MakeProcInstance(DlgProc_name_select,hInst);
        DialogBox (hInst,"D_name_select",hWnd,lpPD_inst_tmp);
        FreeProcInstance(lpPD_inst_tmp);
        return 0;

      /*------- Service Real time Menu---------*/
      case M_realtime:
        State = 5;
        lpPD_inst_tmp= MakeProcInstance(DlgProc_name_real,hInst);
        DialogBox (hInst,"D_realtime",hWnd,lpPD_inst_tmp);
        FreeProcInstance(lpPD_inst_tmp);
        return 0;

      case M_max:
        hWndGraph = LOWORD(SendMessage(hWndClient,WM_MDIGETACTIVE,0,0L));
        GetWindowText(hWndGraph,cBuf,7);
        for(a=0;a<255;a++)
        {
                if(strcmp(MainVar[a].Name,cBuf) == 0)
                {
                  j= a;
                  a = 260;
                }
        }
    Value = 0;
    if(pData = GlobalLock(Mg[j].Mem))
        {
                for(a=0;a<1024;a++)
       {
                if(*pData > Value)
                    Value = *pData;
```

```
                pData++;
                }
        GlobalUnlock(Mg[j].Mem);
         }
         wsprintf(cBuf,"Maximum Value = %d",Value);
         MessageBox(hWnd,cBuf,"Statistic",MB_ICONEXCLAMATION|MB_OK);
    return 0;

        case M_min:
         hWndGraph = LOWORD(SendMessage(hWndClient,WM_MDIGETACTIVE,0,0L));
         GetWindowText(hWndGraph,cBuf,7);
         for(a=0;a<255;a++)
         {
                if(strcmp(MainVar[a].Name,cBuf) == 0)
                {
                  j= a;
                  a = 260;
                }
         }
    Value = 255;
    if(pData = GlobalLock(Mg[j].Mem))
         {
                for(a=0;a<1024;a++)
       {
                if(*pData < Value)
                    Value = *pData;
       pData++;
                }
        GlobalUnlock(Mg[j].Mem);
         }
         wsprintf(cBuf,"Minimum Value = %d",Value);
         MessageBox(hWnd,cBuf,"Statistic",MB_ICONEXCLAMATION|MB_OK);
         return 0;

        case M_mean:
         hWndGraph = LOWORD(SendMessage(hWndClient,WM_MDIGETACTIVE,0,0L));
         GetWindowText(hWndGraph,cBuf,7);
         for(a=0;a<255;a++)
         {
                if(strcmp(MainVar[a].Name,cBuf) == 0)
                {
                  j= a;
                  a = 260;
                }
         }
    num1 = 0;
    if(pData = GlobalLock(Mg[j].Mem))
         {
                for(a=0;a<1024;a++)
       {
                    num1 = *pData + num1;
          pData++;
                }
                Value = num1/1024;
        GlobalUnlock(Mg[j].Mem);
         }
         wsprintf(cBuf,"Mean Value = %d",Value);
         MessageBox(hWnd,cBuf,"Statistic",MB_ICONEXCLAMATION|MB_OK);
         return 0;

        case M_sd:
         hWndGraph = LOWORD(SendMessage(hWndClient,WM_MDIGETACTIVE,0,0L));
         GetWindowText(hWndGraph,cBuf,7);
         for(a=0;a<255;a++)
         {
                if(strcmp(MainVar[a].Name,cBuf) == 0)
                {
                  j= a;
                  a = 260;
                }
         }
    num = 0;
    if(pData = GlobalLock(Mg[j].Mem))
         {
                for(a=0;a<1024;a++)
```

```
{
                num = *(pData+a) + num;
            }
            Value = num/1024;
sd = 0.0;
            for(a=0;a<1024;a++)
            {
   sd = sd + (*(pData+a)-Value) * (*(pData+a)-Value);
            }
sd = sqrt(sd);
GlobalUnlock(Mg[j].Mem);
 }
 sprintf(cBuf,"S.D. Value = %f",sd);
 MessageBox(hWnd,cBuf,"Statistic",MB_ICONEXCLAMATION|MB_OK);
 return 0;

case 101:
 for (a=0;a<=255;a++)
 {
        if (MainVar[a].Use == 0)
        {
 i = a;
 a =260;
}
 }
 MainVar[i].Use = 1;
 hWndGraph = LOWORD(SendMessage(hWndClient,WM_MDIGETACTIVE,0,0L));
 GetWindowText(hWndGraph,cBuf,7);
 for(a=0;a<255;a++)
 {
        if(strcmp(MainVar[a].Name,cBuf) == 0)
        {
          j= a;
          a = 260;
        }
 }

 if(Mg[i].Mem = GlobalAlloc(GMEM_FIXED,1024*sizeof(*fData)))
{
  if((pData = GlobalLock(Mg[j].Mem)) && (fData = GlobalLock(Mg[i].Mem)))
  {
        pTemp = fData;
        for(a=0;a<1024;a++)
        {
          fData->real = *pData;
          fData->imag = 0.0;
          fData++;
          pData++;
        }
        GlobalUnlock(Mg[j].Mem);
        ifft(pTemp,10);                                  //----------FFT-----
        GlobalUnlock(Mg[i].Mem);
  }
  else
        MessageBox(hWnd,"Cannot Lock Memory !!","Message",MB_ICONEXCLAMATION|MB_OK);
  for (a=0;a<=255;a++)
  {
if (MainVar[a].Use == 0)
        { k = a; a =260; }
  }

  MainVar[k].Use = 1;
 if(Mg[k].Mem = GlobalAlloc(GMEM_FIXED,1024*sizeof(*pData)))
 {
  if((pData = GlobalLock(Mg[k].Mem)) && (fData = GlobalLock(Mg[i].Mem)))
{
        for(a=0;a<1024;a++)
        {
/*              if(pTemp->real > 127.0)
                  *pData = 255;
                else if(pTemp->real < 0)
                  *pData = 0;
                else
*/              *pData = (UINT)(pTemp->real);
                pData++;
```

```
                    pTemp++;
                }
                GlobalUnlock(Mg[k].Mem);
                GlobalUnlock(Mg[i].Mem);
                GlobalFree(Mg[i].Mem);
                if(Mg[i].Mem = GlobalAlloc(GMEM_FIXED,1024*sizeof(*pData)))
                {
            if((pData = GlobalLock(Mg[k].Mem)) && (pData2 = GlobalLock(Mg[i].Mem)))
                  {
                   for(a=0;a<1024;a++)
                   {
                          *pData2 = *pData;
                          pData2++;
          pData++;
                    }
                  GlobalUnlock(Mg[i].Mem);
       GlobalUnlock(Mg[k].Mem);
                  }
                }
       }
}
    }
    else
      MessageBox(hWnd,"Cannot Allocates Memory !!","Message",MB_ICONEXCLAMATION|MB_OK);
     // display
            MainVar[k].Use = 1;
            strcpy(MainVar[k].Name,"FFT   ");

            strcpy(Title,MainVar[k].Name);
            wsprintf(cBuf," form Graph ");
            strcat(Title,cBuf);
   strcat(Title,MainVar[j].Name);
        MDIStructVar.szClass  = "ChromicGraph";
        MDIStructVar.szTitle  = Title;
        MDIStructVar.hOwner   = hInst;
        MDIStructVar.x              = CW_USEDEFAULT;
        MDIStructVar.y              = CW_USEDEFAULT;
        MDIStructVar.cx             = CW_USEDEFAULT;
        MDIStructVar.cy             = CW_USEDEFAULT;
        MDIStructVar.style    = 0;
            MDIStructVar.lParam   = NULL;
        hWndGraph = SendMessage(hWndClient, WM_MDICREATE, 0,
                        (LONG)(LPMDICREATESTRUCT)&MDIStructVar);
//Set name handle to WindowWord para for ref. to data
            SetWindowWord(hWndGraph,0,1);

    MessageBox(hWnd,"fft Complete !!","Message",MB_ICONEXCLAMATION|MB_OK);
    return 0;

   case 105:
    for (a=0;a<=255;a++)
    {
            if (MainVar[a].Use == 0)
            {
    i = a;
    a =260;
  }
    }
    MainVar[i].Use = 1;
    hWndGraph = LOWORD(SendMessage(hWndClient,WM_MDIGETACTIVE,0,0L));
    GetWindowText(hWndGraph,cBuf,7);
    for(a=0;a<255;a++)
    {
            if(strcmp(MainVar[a].Name,cBuf) == 0)
            {
              j= a;
              a = 260;
            }
    }

    if(Mg[i].Mem = GlobalAlloc(GMEM_FIXED,1024*sizeof(*fData)))
{
      if((pData = GlobalLock(Mg[j].Mem)) && (fData = GlobalLock(Mg[i].Mem)))
      {
            pTemp = fData;
```

```
                for(a=0;a<1024;a++)
                {
                  fData->real = *pData;
                  fData->imag = 0.0;
                  fData++;
                  pData++;
                }
                GlobalUnlock(Mg[j].Mem);
                fft(pTemp,10);                    //------IFFT--------
                GlobalUnlock(Mg[i].Mem);
        }
        else
                MessageBox(hWnd,"Cannot Lock Memory !!","Message",MB_ICONEXCLAMATION|MB_OK);
        for (a=0;a<=255;a++)
        {
    if (MainVar[a].Use == 0)
              { k = a; a =260; }
        }
        MainVar[k].Use = 1;
      if(Mg[k].Mem = GlobalAlloc(GMEM_FIXED,1024*sizeof(*pData)))
      {
        if((pData = GlobalLock(Mg[k].Mem)) && (fData = GlobalLock(Mg[i].Mem)))
   {
                for(a=0;a<1024;a++)
                {
/*                      if(pTemp->real > 127.0)
                        *pData = 255;
                      else if(pTemp->real < 0)
                        *pData = 0;
                      else
    */                  *pData = (UINT)(pTemp->real);
                      pData++;
                      pTemp++;
                }
                GlobalUnlock(Mg[k].Mem);
                GlobalUnlock(Mg[i].Mem);
                GlobalFree(Mg[i].Mem);
               if(Mg[i].Mem = GlobalAlloc(GMEM_FIXED,1024*sizeof(*pData)))
               {
           if((pData = GlobalLock(Mg[k].Mem)) && (pData2 = GlobalLock(Mg[i].Mem)))
              {
                for(a=0;a<1024;a++)
                {
                      *pData2 = *pData;
                      pData2++;
          pData++;
                }
                GlobalUnlock(Mg[i].Mem);
      GlobalUnlock(Mg[k].Mem);
              }
            }
       }
  }
      }
      else
        MessageBox(hWnd,"Cannot Allocates Memory !!","Message",MB_ICONEXCLAMATION|MB_OK);
       // display
              MainVar[k].Use = 1;
              strcpy(MainVar[k].Name,"IFFT ");

              strcpy(Title,MainVar[k].Name);
              wsprintf(cBuf," form Graph ");
              strcat(Title,cBuf);
    strcat(Title,MainVar[j].Name);
          MDIStructVar.szClass  = "ChromicGraph";
          MDIStructVar.szTitle  = Title;
          MDIStructVar.hOwner   = hInst;
          MDIStructVar.x              = CW_USEDEFAULT;
          MDIStructVar.y              = CW_USEDEFAULT;
          MDIStructVar.cx             = CW_USEDEFAULT;
          MDIStructVar.cy             = CW_USEDEFAULT;
          MDIStructVar.style    = 0;
              MDIStructVar.lParam   = NULL;
          hWndGraph = SendMessage(hWndClient, WM_MDICREATE, 0,
                          (LONG)(LPMDICREATESTRUCT)&MDIStructVar);
```

```
//Set name handle to WindowWord para for ref. to data
                SetWindowWord(hWndGraph,0,1);

        MessageBox(hWnd,"ifft Complete !!","Message",MB_ICONEXCLAMATION|MB_OK);
        return 0;

/*      case 105:
        for (a=0;a<=255;a++)
        {
      if (MainVar[a].Use == 0)
                {
        i = a;
        a =260;
      }
        }
        hWndGraph = LOWORD(SendMessage(hWndClient,WM_MDIGETACTIVE,0,0L));
        GetWindowText(hWndGraph,cBuf,7);
        for(a=0;a<255;a++)
        {
                if(strcmp(MainVar[a].Name,cBuf) == 0)
                {
                  j= a;
                  a = 260;
                }
        }

        if(Mg[i].Mem = GlobalAlloc(GMEM_FIXED,1024*sizeof(*fData)))
{
          if((pData = GlobalLock(Mg[j].Mem)) && (fData = GlobalLock(Mg[i].Mem)))
 {
                  for(a=0;a<1024;a++)
                  {
                    fData->real = *pData;
          fData->imag = 0.0;
                    fData++;
          pData++;
                  }
                  GlobalUnlock(Mg[j].Mem);
                  ifft(fData,10);
                  GlobalUnlock(Mg[i].Mem);
        }
        for (a=0;a<=255;a++)
        {
      if (MainVar[a].Use == 0)
                {
        k = a;
        a =260;
      }
        }
        MainVar[k].Use = 1;
        if((pData = GlobalLock(Mg[k].Mem)) && (fData = GlobalLock(Mg[i].Mem)))
 {
                  for(a=0;a<1024;a++)
                  {
                    pData = (UINT)(fData->real*100)+128,
                    pData++;
                    fData++;
                  }
                  GlobalUnlock(Mg[k].Mem);
                  GlobalUnlock(Mg[i].Mem);
 }
        }
        else
          MessageBox(hWnd,"Cannot Allocates Memory !!","Message",MB_ICONEXCLAMATION|MB_OK);
         // display
        MainVar[k].Use = 1;
        strcpy(MainVar[k].Name,"IFFT ");
                strcpy(Title,MainVar[k].Name);
            wsprintf(cBuf," form Graph ");
                strcat(Title,cBuf);
      strcat(Title,MainVar[j].Name);
            MDIStructVar.szClass  = "ChromicGraph";
            MDIStructVar.szTitle  = Title;
            MDIStructVar.hOwner   = hInst;
            MDIStructVar.x              = CW_USEDEFAULT;
```

```
                MDIStructVar.y          = CW_USEDEFAULT;
                MDIStructVar.cx         = CW_USEDEFAULT;
                MDIStructVar.cy         = CW_USEDEFAULT;
                MDIStructVar.style   = 0;
                MDIStructVar.lParam   = NULL;
                hWndGraph = SendMessage(hWndClient, WM_MDICREATE, 0,
                                (LONG)(LPMDICREATESTRUCT)&MDIStructVar);
      //Set name handle to WindowWord para for ref. to data
                        SetWindowWord(hWndGraph,0,1);

           MessageBox(hWnd,"ifft Complete !!","Message",MB_ICONEXCLAMATION|MB_OK);
           return 0;
*/
          /*-------- Service Windows Menu -------*/
          case M_wind_titl:
       SendMessage(hWndClient,WM_MDITILE,0,0L);
           return 0;
          case M_wind_casc:
           SendMessage(hWndClient,WM_MDICASCADE,0,0L);
           return 0;
          case M_wind_arra:
           SendMessage(hWndClient,WM_MDIICONARRANGE,0,0L);
           return 0;
          case M_wind_clea_all:
           lpfnEnum = MakeProcInstance(CloseEnumProc,hInst);
           EnumChildWindows(hWndClient, lpfnEnum,0L);
           FreeProcInstance(lpfnEnum);
       return 0;

      /*-------- Service Help Menu----------*/
           case M_help_about:
            lpPD_inst_tmp= MakeProcInstance(DlgProc_help_about,hInst);
            DialogBox (hInst,MAKEINTRESOURCE(D_about),hWnd,lpPD_inst_tmp);
            FreeProcInstance(lpPD_inst_tmp);
            return 0;

      default:
             hWndGraph = LOWORD(SendMessage(hWndClient,WM_MDIGETACTIVE,0,0L));
             if (IsWindow(hWndGraph))
              SendMessage(hWndGraph,WM_COMMAND,wParam,lParam);
        break;

     } //----End of WM_COMMAND switch wParam----
     break;
    case WM_CLOSE:
     /*---Post claer all command to the message queue but now not process ----*/
     SendMessage(hWnd,WM_COMMAND,M_wind_clea_all,0L);
     /*---Check for opening window. if ture return to Main sytem to process clear all message ---*/
     if(NULL!=GetWindow(hWndClient,GW_CHILD))
           return 0;
     break;
  }
 return(DefFrameProc(hWnd,hWndClient,msg,wParam,lParam));
}


long FAR PASCAL ChromicGraphProc( HWND CGP_h, WORD CGP_msg, WORD CGP_wpara, LONG CGP_lpara)
 {
  HDC             hDC;
  RECT            UpdateArea;
  PAINTSTRUCT     ps;
  static HWND  hWndClient,hwndFrame;

  static unsigned char MouseLeft=0, MouseRight=0;

  switch(CGP_msg)
   {
    case WM_CREATE:
     hWndClient = GetParent(CGP_h);
     hwndFrame  = GetParent(hWndClient);
     return 0;

    case WM_MDIACTIVATE:
     if (CGP_wpara == TRUE)
```

```
            SendMessage (hWndClient, WM_MDISETMENU, 0, MAKELONG(hMenuGraph, hMenuGraphWin) );
        else
            SendMessage (hWndClient, WM_MDISETMENU, 0, MAKELONG(hMenuMain, hMenuMainWin) );
        DrawMenuBar (hwndFrame);
//      SendMessage(CGP_h,WM_PAINT,0,0L);
        return 0;

    case WM_PAINT:
        if (!IsIconic(CGP_h))
            GraphRefresh(CGP_h,CGP_wpara,CGP_lpara);
        //*------ Invalid rect are updated now, stop send WM_PAINT -----
//      GetUpdateRect(CGP_h,&UpdateArea,0);  // 0=not erase bkg
//      ValidateRect(CGP_h,&UpdateArea);
        return 0;

    case WM_QUERYENDSESSION:
    case WM_CLOSE:
//      if (IDOK != BWCCMessageBox(CGP_h,"Are you sure to Quit?","Quit
Graph",MB_ICONQUESTION|MB_OKCANCEL))
//          return 0;
        break;

    case WM_LBUTTONDOWN:
        MouseLeft=1;
        return 0;
    case WM_RBUTTONDOWN:
        MouseRight=1;
        return 0;
    case WM_LBUTTONUP:
        if (MouseLeft==1)
            { MouseLeft=0;  // clear to prepare for next mouse click
              SendMessage(CGP_h,WM_PAINT,0,CGP_lpara);   //  lpara <0 mouse left
            }
        return 0;
    case WM_RBUTTONUP:
        if (MouseRight==1)
            { MouseRight=0;                 // clear to prepare for next mouse click
              SendMessage(CGP_h,WM_PAINT,0,-CGP_lpara);   // lpara > 0  mouse left
            }
        return 0;

    case WM_DESTROY:
        return 0;

    case WM_COMMAND:
        switch(CGP_wpara)
        {
            case M_graph_line:
                SetWindowWord(CGP_h,0,1);
                InvalidateRect(CGP_h,NULL,TRUE);
                return 0;
            case M_graph_point:
                SetWindowWord(CGP_h,0,2);
                InvalidateRect(CGP_h,NULL,TRUE);
                return 0;
            case M_graph_bar:
                SetWindowWord(CGP_h,0,3);
                InvalidateRect(CGP_h,NULL,TRUE);
                return 0;
        } // end of WM_COMMAND switch CGP_wpara
        return 0;
    }
    return (DefMDIChildProc(CGP_h, CGP_msg, CGP_wpara, CGP_lpara) );
}

/*-------------------------------------------------------------*/
long GraphRefresh(HWND hWnd, WORD wPara, LONG lPara)
{
    int             InDataNum=1024;
    HDC     hDC;
    HPEN            hPen,hPenOld;
    HBRUSH          hBrush,hBrushOld;
    RECT            GClient,GWork,GAll,GZoom, GAllText, GZoomText;
    PAINTSTRUCT     ps;
    GRAPHPARA       gp;
```

```
char            cBuf[10];
long            TempLong=0, a=0;
int             FrameThick=10, TempInt=0,i,j;
int             MouseX=0, MouseY=0, MouseButton=0;

static unsigned long  CursorX1, CursorX2;  // curcor location in data unit
if (lPara != 0)
 { MouseButton = (lPara > 0) ? 1 : 2 ;          // negative value is right button
   lPara = (lPara<0) ? -lPara : lPara;
   MouseY=HIWORD(lPara);   MouseX=LOWORD(lPara); //-- mouse x,y in pixel
 }
hDC = BeginPaint(hWnd,&ps);
GetClientRect(hWnd,&GClient);
//*---- check for minium frame --------*/

GWork.top   = GClient.top   +FrameThick;
GWork.bottom = GClient.bottom-FrameThick;
GWork.left   = GClient.left  +FrameThick;
GWork.right  = GClient.right -FrameThick;

a = (GWork.bottom-GWork.top-FrameThick)/3;

GZoom.top   = GWork.top;
GZoom.bottom = GWork.top+(a<<1);
GZoom.left   = GWork.left;
GZoom.right  = GWork.right;

GAll.top    = GZoom.bottom+FrameThick;
GAll.bottom  = GAll.top+a;
GAll.left    = GWork.left;
GAll.right   = GWork.right;

GetWindowText(hWnd,cBuf,7);
for(j=0;j<255;j++)
{
  if(strcmp(MainVar[j].Name,cBuf) == 0)
  {
        i = j;
        j = 260;
  }
}
//-- set para for DisplayGraph routine to show All graph window ----
//-- and call to display all data graph
gp.hWnd          = hWnd;
gp.hDC           = hDC;
gp.top           = GAll.top;
gp.bottom        = GAll.bottom;
gp.left    = GAll.left;
gp.right         = GAll.right;
gp.GraphType  = GetWindowWord(hWnd,0);
gp.BkgColor   = RGB(0,200,0);   gp.GraphColor=RGB(0,0,255);
gp.GridColor  = RGB(255,0,0);   gp.AxiesColor=gp.GridColor;
gp.DataNum    = lnDataNum;
gp.hData         = Mg[i].Mem;
gp.MinX = 0;          gp.MaxX=0;          gp.GridX=0;
gp.MinY = 0;       gp.MinY=0;             gp.GridY=0;
gp.CursorX1   = CursorX1;   gp.CursorX2=CursorX2;
gp.MouseButton = MouseButton;      // Pass mouse botton click
gp.CursorStyle = PS_DASHDOTDOT;
gp.CursorColor = RGB(255,255,255);
//--- check  location off mouse click are in all graph area ? -----
if (MouseButton>0)
{ if ( (MouseX>=GAll.left) && (MouseX<=GAll.right) && (MouseY>=GAll.top) && (MouseY<=GAll.bottom) )
   { gp.MouseX=MouseX;   gp.MouseY=MouseY; }
  else
   { gp.MouseX=0;   gp.MouseY=0; }
}
TempLong=DisplayGraph(gp);
//update new cursor location
if (TempLong != 0)
 {  CursorX1 = HIWORD(TempLong);  CursorX2=LOWORD(TempLong); }

//-- set para for DisplayGraph routine to show Zoom graph window ----
//-- and call to display zoom graph
```

```
  if (CursorX2<CursorX1)    // if Cursor2 < Cursor1 use automode
   { gp.MaxX=0; gp.MinX=0; }
  else
   { gp.MaxX=CursorX2; gp.MinX=CursorX1; }
 //  gp.MaxX=60; gp.MinX=30;
  gp.top              = GZoom.top;
  gp.bottom           = GZoom.bottom;
  gp.left    = GZoom.left;
  gp.right            = GZoom.right;
  gp.GraphType  = GetWindowWord(hWnd,0);
  gp.BkgColor   = RGB(0,200,0);   gp.GraphColor=RGB(0,0,255);
  gp.GridColor  = RGB(255,0,0);   gp.AxiesColor=gp.GridColor;
  gp.DataNum    = lnDataNum;
  gp.hData            = Mg[i].Mem;
  gp.GridX      = 0;  // gp.Min and gp.Max are set
  gp.MinY = 0;  gp.MinY=0;      gp.GridY=0;
  gp.CursorX1   = 0;  gp.CursorX2=0;  gp.CursorStyle=0;
  gp.MouseX     = 0;  gp.MouseY=0;
  gp.MouseButton= 0;
  DisplayGraph(gp);

  /*---- Clear unused Objects -------*/
  EndPaint(hWnd,&ps);
return 0;
}


/*-------------------------------------------------------------

 Use :  set handle, set graph size and type, bkgcolor,graphcolor,gridcolor,axiescolor
          set number of data, pass start add of plot data
          set scale and grid  of x,y or 0=auto

*/
long DisplayGraph(GRAPHPARA Gp)
{
 HBRUSH          hBkgBrush,hBkgBrushOld;
 HPEN        hGraphPen,hGraphPenOld;
 HPEN            hGridPen,hGridPenOld;
 HPEN            hAxiesPen,hAxiesPenOld;
 HDC             hDC;

 POINT   PlotVar;

 char            ScaleMarkSize=3,ScaleMarkStepX=5,ScaleMarkStepY=5;
 unsigned int far *lnData;
 int             lnDataMax=0,lnDataMin=0,lnDataRange=0;
 float     lnStepMarkX =1.0, lnStepMarkY =1.0;
 float           lnPlotRatioX=0.0, lnPlotRatioY=0.0;
 int             dxWork=0,dyWork=0;
 long       a=0, TempMinX=0, TempMaxX=0, TempLong=0, d=0, CursorX=0;
 char            b=0, c=0 ;

 /*---- Draw Work Area -------*/
 hDC = Gp.hDC;
 hBkgBrush = CreateSolidBrush(Gp.BkgColor);              //--set work bkg color
 hBkgBrushOld = SelectObject(hDC,hBkgBrush);
 SetBkColor(hDC,Gp.BkgColor);                                    //--set bk of chr & line
 Rectangle(hDC,Gp.left,Gp.top,Gp.right,Gp.bottom);
 SelectObject(hDC,hBkgBrushOld);
 DeleteObject(hBkgBrush);

 /*---- Transform to plot coordinate ------*/
 dxWork = Gp.right  - Gp.left;
 dyWork = Gp.bottom - Gp.top;

 // Scan for range of indata
 lnDataMax = 255;
 lnDataMin = 0;

/*  Gp.MinX = (Gp.MinX>Gp.MaxX) ? 0 : Gp.MinX;

 for ( a=1; a<=Gp.DataNum; a++,lnData++)
 { lnDataMax = ( *lnData%255 > lnDataMax) ?  (*lnData%255) : lnDataMax ;
   lnDataMin = ( *lnData%255 < lnDataMin) ?  (*lnData%255) : lnDataMin ;
```

```
  }*/
  lnDataRange = 1.1*(lnDataMax-lnDataMin);

 /*---------- Set Scale and Grid axies to selected mode ----------*/
  lnPlotRatioX = (Gp.MinX == 0) ? (float)Gp.DataNum  : (float)(Gp.MaxX-Gp.MinX) ;
  lnPlotRatioY = (Gp.MinY == 0) ? (float)lnDataRange : (float)(Gp.MaxY-Gp.MinY) ;

  lnPlotRatioX =  ((float)dxWork / lnPlotRatioX );
  lnPlotRatioY =  ((float)dyWork / lnPlotRatioY );
//-------------------------------------------------------//
  if (lnPlotRatioX < 3.0)
   { lnStepMarkX = (lnPlotRatioX<3.0) ? 5.0  : 1.0 ;
     lnStepMarkX = (lnPlotRatioX<0.6) ? 10.0 : lnStepMarkX ; }
  if (lnPlotRatioY < 3.0)
   { lnStepMarkY = (lnPlotRatioY<3.0) ? 5.0  : 1.0 ;
     lnStepMarkY = (lnPlotRatioY<0.6) ? 10.0 : lnStepMarkY ; }


  ScaleMarkStepX = (Gp.GridX==0) ? 5 : Gp.GridX;    //--- MarkStep = time of unit
  ScaleMarkStepY = (Gp.GridY==0) ? 5 : Gp.GridY;
/*---- Set Graph Axies ---must be pass (0,0) both x,y -------*/
  hAxiesPen = CreatePen(PS_SOLID,1,Gp.AxiesColor);      //--set axies color
  hGridPen  = CreatePen(PS_DOT  ,1,Gp.GridColor );
  hAxiesPenOld = SelectObject(hDC,hAxiesPen);
  MoveTo (hDC,Gp.left ,Gp.top);
  LineTo (hDC,Gp.left ,Gp.bottom);
  LineTo (hDC,Gp.right,Gp.bottom);
  /*---- Set Scale mark on Axies and Grids -------*/
  if(Gp.MinX==0)
   { TempMinX=1;      TempMaxX=Gp.DataNum;}
   // lnData=Gp.lnData;         }
  else
   { TempMinX=Gp.MinX; TempMaxX=Gp.MaxX;}
   //    lnData=Gp.lnData+Gp.MinX-1; }

// SelectObject(hDC,hAxiesPen);
  TempLong = TempMinX;
  for (d=1 ; a<Gp.right;d++,TempLong++)
  { c = ScaleMarkSize;
    a = (int)( (float)Gp.left + ( lnPlotRatioX*lnStepMarkX*(float)d ) );
    if ( (TempLong % ScaleMarkStepX)==0)
     { c=c<<1;
       hGridPenOld = SelectObject(hDC,hGridPen);
            MoveTo(hDC,a,Gp.top); LineTo(hDC,a,Gp.bottom);
       SelectObject(hDC,hGridPenOld);
     }
    MoveTo (hDC,a,Gp.bottom-c);
    LineTo (hDC,a,Gp.bottom+c);
  }
  b=0; a=Gp.bottom;
  for (d=1 ;a>=Gp.top ;d++)
  { c = ScaleMarkSize;
    if (b==ScaleMarkStepY)
     { c=c<<1;   b=0;
            hGridPenOld = SelectObject(hDC,hGridPen);
            MoveTo(hDC,Gp.left ,a); LineTo(hDC,Gp.right,a);
            SelectObject(hDC,hGridPenOld);
     }
    MoveTo(hDC,Gp.left-c,a);
    LineTo(hDC,Gp.left+c,a);
    b++;
    a = (int)((float)Gp.bottom - ( lnPlotRatioY*lnStepMarkY* (float)d) );
  }
  SelectObject(hDC,hAxiesPenOld);
  DeleteObject(hGridPen);
  DeleteObject(hAxiesPen);

  /*---- set Pen for draw graph of data ---------*/
  hGraphPen = CreatePen(PS_SOLID,1,Gp.GraphColor);
  hGraphPenOld = SelectObject(hDC,hGraphPen);

  /*---- Plot ----*/
  // PlotStepX = range X / 75% of total pixel in x axies
// lnData = Gp.lnData+Gp.MinX;
  switch (Gp.GraphType)
```

```
  { case 1:             //GT_LINE:
    InData = GlobalLock(Gp.hData);
    PlotVar.x =  (int)( (float)Gp.left );
    PlotVar.y =  (int)( (float)Gp.bottom - ((float)(*InData%255)*InPlotRatioY) );
    MoveTo (hDC,PlotVar.x,PlotVar.y);
    InData++;
    for (a=1;a<=(TempMaxX-TempMinX);a++,InData++)
    { PlotVar.x = (int)( (float)Gp.left   + ( (float)a          *InPlotRatioX) );
          PlotVar.y = (int)( (float)Gp.bottom - ( (float)(*InData%255)*InPlotRatioY) );
          LineTo (hDC,PlotVar.x,PlotVar.y);
    }
    GlobalUnlock(Gp.hData);
    break;
   case 2:              //GT_POINT:
    InData = GlobalLock(Gp.hData);
    for (a=1;a<=(TempMaxX-TempMinX);a++,InData++)
    { PlotVar.x =  (int)( (float)Gp.left   + ((float)a*InPlotRatioX) );
          PlotVar.y =  (int)( (float)Gp.bottom - ((float)(*InData%255)*InPlotRatioY  ));
          if (PlotVar.y<Gp.top)
      { PlotVar.y =  Gp.top;
            MoveTo (hDC,PlotVar.x-2,PlotVar.y-2);
            LineTo (hDC,PlotVar.x+2,PlotVar.y+2);
            MoveTo (hDC,PlotVar.x+2,PlotVar.y-2);
            LineTo (hDC,PlotVar.x-2,PlotVar.y+2);
      }
          MoveTo (hDC,PlotVar.x-2,PlotVar.y);
          LineTo (hDC,PlotVar.x+2,PlotVar.y);
          MoveTo (hDC,PlotVar.x,PlotVar.y-2);
          LineTo (hDC,PlotVar.x,PlotVar.y+2);
    }
    GlobalUnlock(Gp.hData);
    break;
   case 3:              //GT_BAR:
    InData = GlobalLock(Gp.hData);
    for (a=1;a<=(TempMaxX-TempMinX);a++,InData++)
    { PlotVar.x = (int)( (float)Gp.left   + ((float)a          *InPlotRatioX) );
          PlotVar.y = (int)( (float)Gp.bottom - ((float)(*InData%255) *InPlotRatioY) );
          PlotVar.y = (PlotVar.y<Gp.top) ? Gp.top : PlotVar.y;
          MoveTo (hDC,PlotVar.x,PlotVar.y);
          LineTo (hDC,PlotVar.x,Gp.bottom);
    }
    GlobalUnlock(Gp.hData);
    break;
   case 4:              //GT_PIE:
    break;
 }                      //------End of switch GraphType------
 SelectObject(hDC,hGraphPenOld);
 DeleteObject(hGraphPen);

 /*---------- Display Cursor if Gp.CursorX1 >0 ----------*/
 if (Gp.MouseButton > 0 )  //--- WM_PRINT from press mouse button
 {
  //-- transform mouse location to data unit -------
  CursorX = (long)((float)(Gp.MouseX-Gp.left)/InPlotRatioX);
  if (Gp.MouseButton==1)        // left button
     Gp.CursorX1 = CursorX;
  if (Gp.MouseButton==2)        //  right button
     Gp.CursorX2 = CursorX;
  hGridPen  = CreatePen(Gp.CursorStyle,1,Gp.CursorColor);
  hGridPenOld = SelectObject(hDC,hGridPen);
  PlotVar.x = (int)( (float)Gp.left + ( (float)Gp.CursorX1*InPlotRatioX) );
  MoveTo (hDC,PlotVar.x,Gp.top);
  LineTo (hDC,PlotVar.x,Gp.bottom);
  PlotVar.x = (int)( (float)Gp.left + ( (float)Gp.CursorX2*InPlotRatioX) );
  MoveTo (hDC,PlotVar.x,Gp.top);
  LineTo (hDC,PlotVar.x,Gp.bottom);
  SelectObject(hDC,hGridPenOld);
  DeleteObject(hGridPen);
  a =  (Gp.CursorX1<<16)+Gp.CursorX2 ;
 }
 else
  a = 0;
//--- return new cursor location HIWORD=X1   LOWORD=X2---//
return (a);
}
```

```
/*--------- c_real -------*/
extern intState;
extern intnTimer;
extern    int         num;
extern    HANDLE hInst;
extern HWND   hWndGraph;
extern HWND   hWndClient;
extern MDICREATESTRUCT MDIStructVar;
extern GRAPHSTRUCT       Mg[255];

BOOL FAR PASCAL DlgProc_name_real (HWND Dlg_h, WORD Dlg_msg , WORD Dlg_wpara, LONG Dlg_lpara)
{
 int  maxch=0,a=0,b=0,i;
 int  TempIntVar=0;
 float TempFloatVar=0.0;
 char *TempCharVar;
 char TempCharVar1[12];
 BOOL TempBOOL;
 char     Title[128],cBuf[128];
 static int rt;
// static DWORD num;
 HWND  hWnd;

 static NAMESTRUCT  TempName;

 long  TempLong=0,TempLong1=0;

 switch (Dlg_msg)
 {
  case WM_DESTROY:
   return TRUE;

  case WM_INITDIALOG:
   num = 0;
   SendDlgItemMessage (Dlg_h,DCI_name_name,CB_RESETCONTENT,0,0L);
   for (TempIntVar=0;TempIntVar<=255;TempIntVar++)
   { if (MainVar[TempIntVar].Use==1)
          SendDlgItemMessage (Dlg_h,DCI_name_name,CB_ADDSTRING,0,(long)MainVar[TempIntVar].Name);
   }
   SendMessage(Dlg_h,CB_SETCURSEL,num,0L);
   ltoa(MainVar[0].Start,TempCharVar,10);
   SetDlgItemText(Dlg_h,DCI_name_memstart,TempCharVar);
   ltoa(MainVar[0].End,TempCharVar,10);
   SetDlgItemText(Dlg_h,DCI_name_memend,TempCharVar);
   if (MainVar[0].End <= MainVar[0].Start)
     SetDlgItemText(Dlg_h,DCI_name_memnum,"--â°éäÁèä´é--");
   else
   {
     TempLong1 =MainVar[0].End - MainVar[0].Start;
     SetDlgItemText(Dlg_h,DCI_name_memnum,itoa(TempLong1,TempCharVar1,10));
   }
   CheckRadioButton(Dlg_h,DCI_real_55,DCI_real_20000,DCI_real_55);
   rt = 55;
  break;

  case WM_COMMAND:
   switch(Dlg_wpara)
   {
          case DCI_real_55:
          case DCI_real_100:
          case DCI_real_200:
          case DCI_real_1000:
          case DCI_real_2000:
          case DCI_real_10000:
     case DCI_real_20000:
            CheckRadioButton(Dlg_h,DCI_real_55,DCI_real_20000,Dlg_wpara);
       rt = Dlg_wpara;
       break;

          case IDOK:
            //RealTime
             num = (DWORD)SendDlgItemMessage(Dlg_h,DCI_name_name,CB_GETCURSEL,0,0L);
             if(MainVar[num].Start == 0)
                  ClearCounter();
```

```
                else
                        SetCounter(MainVar[num].Start);
                StopCounter(MainVar[num].End);
                SetChannel(0);
                SetFreq(MainVar[num].Freq);
                hWnd = GetParent(Dlg_h);
                if(!(Mg[num].Mem = GlobalAlloc(GMEM_FIXED | GMEM_ZEROINIT,1024*sizeof(*pData))))
                  MessageBox(hWnd,"Cannot Allocates Memory !!","Message",MB_ICONEXCLAMATION|MB_OK);

                if(!(nTimer = SetTimer(hWnd,1,------*/
                        MessageBox(hWnd,"No Timers !","Message",MB_OK);
                else
        {
                  // display
                        strcpy(Title,MainVar[num].Name);
                  wsprintf(cBuf," <%d : %d> Display<%d -
%d>",(int)MainVar[num].Start,(int)MainVar[num].End,(int)MainVar[num].Start,(int)MainVar[num].Start+1024);
          strcat(Title,cBuf);
                  MDIStructVar.szClass  = "ChromicGraph";
                  MDIStructVar.szTitle  = Title;
                  MDIStructVar.hOwner   = hInst;
                  MDIStructVar.x              = CW_USEDEFAULT;
                  MDIStructVar.y              = CW_USEDEFAULT;
                  MDIStructVar.cx             = CW_USEDEFAULT;
                  MDIStructVar.cy             = CW_USEDEFAULT;
                  MDIStructVar.style    = 0;
                  MDIStructVar.lParam   = NULL;
                  hWndGraph = SendMessage(hWndClient, WM_MDICREATE, 0,
                                  (LONG)(LPMDICREATESTRUCT)&MDIStructVar);
      //Set name handle to WindowWord para for ref. to data
                        SetWindowWord(hWndGraph,0,1);
                        ShowWindow(hWndGraph,SW_SHOWMAXIMIZED);
                        Mg[num].Use = 1;
//                      Mg[num].DataType = 0;
                        Mg[num].ZoomX1=0;
                        Mg[num].ZoomX2=0;
                  }
                  EndDialog (Dlg_h,TRUE);
          return TRUE;

          case IDCANCEL:
            EndDialog (Dlg_h,TRUE);
            return TRUE;

          case IDHELP:
            MessageBox(Dlg_h," Not SUPPORT now! ","ERROR",MB_ICONEXCLAMATION|MB_OK);
            break;
          case DCI_name_name:
            num = SendDlgItemMessage(Dlg_h,DCI_name_name,CB_GETCURSEL,0,0L);
            if(MainVar[num].Use == 1)
            {
              itoa(MainVar[num].Start,TempCharVar,10);
              SetDlgItemText(Dlg_h,DCI_name_memstart,TempCharVar);
              itoa(MainVar[num].End,TempCharVar,10);
              SetDlgItemText(Dlg_h,DCI_name_memend,TempCharVar);
              if (MainVar[num].End <= MainVar[num].Start)
        SetDlgItemText(Dlg_h,DCI_name_memnum,"--â*éäÁèä´é--");
              else
              {
        TempLong1 =MainVar[0].End - MainVar[0].Start;
        SetDlgItemText(Dlg_h,DCI_name_memnum,itoa(TempLong1,TempCharVar1,10));
              }
            }
    }
    default: /*------ switch Dlg_msg -------*/
      return FALSE;
  }
  return TRUE;
 }
*------c_name_s------*
extern intState;
extern intnTimer;
extern    int         num;
extern    HANDLE hInst;
extern HWND   hWndGraph;
```

```
extern HWND  hWndClient;
extern MDICREATESTRUCT MDIStructVar;

BOOL FAR PASCAL DlgProc_name_select (HWND Dlg_h, WORD Dlg_msg , WORD Dlg_wpara, LONG Dlg_lpara)
{
 int  maxch=0,a=0,b=0,i;
 int  TempIntVar=0;
 float TempFloatVar=0.0;
 char *TempCharVar;
 char TempCharVar1[12];
 BOOL TempBOOL;
 char     Title[128],cBuf[128];
// static DWORD num;
 HWND  hWnd;

 static NAMESTRUCT  TempName;

 long TempLong=0,TempLong1=0;

 switch (Dlg_msg)
 {
  case WM_DESTROY:
   return TRUE;

  case WM_INITDIALOG:
   num = 0;
   SendDlgItemMessage (Dlg_h,DCI_name_name,CB_RESETCONTENT,0,0L);
   for (TempIntVar=0;TempIntVar<=255;TempIntVar++)
   { if (MainVar[TempIntVar].Use==1)
         SendDlgItemMessage (Dlg_h,DCI_name_name,CB_ADDSTRING,0,(long)MainVar[TempIntVar].Name);
   }
   SendMessage(Dlg_h,CB_SETCURSEL,num,0L);
   itoa(MainVar[0].Start,TempCharVar,10);
   SetDlgItemText(Dlg_h,DCI_name_memstart,TempCharVar);
   itoa(MainVar[0].End,TempCharVar,10);
   SetDlgItemText(Dlg_h,DCI_name_memend,TempCharVar);
   if (MainVar[0].End <= MainVar[0].Start)
     SetDlgItemText(Dlg_h,DCI_name_memnum,"--ã*éäÅèä´é--");
   else
   {
     TempLong1 =MainVar[0].End - MainVar[0].Start;
     SetDlgItemText(Dlg_h,DCI_name_memnum,itoa(TempLong1,TempCharVar1,10));
   }
   CheckRadioButton(Dlg_h,DCI_name_20M,DCI_name_13,DCI_name_20M+MainVar[0].Freq);
   CheckRadioButton(Dlg_h,DCI_name_maxch1,DCI_name_maxch8,DCI_name_maxch1+MainVar[0].Ch);
   switch(MainVar[0].Ch)
   {
    case 1:wsprintf(TempCharVar,"20 MHz");break;
    case 2:wsprintf(TempCharVar,"10 MHz");break;
    case 3:wsprintf(TempCharVar,"6.67 MHz");break;
    case 4:wsprintf(TempCharVar,"5 MHz");break;
    case 5:wsprintf(TempCharVar,"4 MHz");break;
    case 6:wsprintf(TempCharVar,"3.34 MHz");break;
    case 7:wsprintf(TempCharVar,"2.85 MHz");break;
    case 8:wsprintf(TempCharVar,"2.5 MHz");break;
   }
    /*---- place the point at location whicn in TempIntVar---*/
   SetDlgItemText(Dlg_h,DCI_name_maxfrq,TempCharVar);
   SendDlgItemMessage (Dlg_h,DCI_name_20M,WM_ENABLE,TRUE,0L);
   SendDlgItemMessage (Dlg_h,DCI_name_10M,WM_ENABLE,TRUE,0L);
   SendDlgItemMessage (Dlg_h,DCI_name_10M3,WM_ENABLE,TRUE,0L);
   SendDlgItemMessage (Dlg_h,DCI_name_5M,WM_ENABLE,TRUE,0L);
   if (TempName.Ch >= 5)
     SendDlgItemMessage (Dlg_h,DCI_name_5M,WM_ENABLE,FALSE,0L);
   if (TempName.Ch >= 3)
   {
    SendDlgItemMessage (Dlg_h,DCI_name_10M3,WM_ENABLE,FALSE,0L);
    SendDlgItemMessage (Dlg_h,DCI_name_10M,WM_ENABLE,FALSE,0L);
   }
   if (TempName.Ch >= 2)
         SendDlgItemMessage (Dlg_h,DCI_name_20M,WM_ENABLE,FALSE,0L);

   TempName.Ch  = MainVar[0].Ch;
   TempName.Freq= MainVar[0].Freq;
   break;
```

```
case WM_COMMAND:
 if ( (Dlg_wpara>=DCI_name_20M) && (Dlg_wpara<=DCI_name_13))
 {
        TempName.Freq = (char)(Dlg_wpara-8561);   // 8561 must change when reassign dialog id
        CheckRadioButton(Dlg_h,DCI_name_20M,DCI_name_13,Dlg_wpara);
  return TRUE;
 }
 else if ( (Dlg_wpara>=DCI_name_maxch1) && (Dlg_wpara<=DCI_name_maxch8))
 {
        TempName.Ch = (char)(Dlg_wpara-8260);
        CheckRadioButton(Dlg_h,DCI_name_maxch1,DCI_name_maxch8,Dlg_wpara);
        switch(TempName.Ch)
        {
          case 1:wsprintf(TempCharVar,"20 MHz");break;
          case 2:wsprintf(TempCharVar,"10 MHz");break;
          case 3:wsprintf(TempCharVar,"6.67 MHz");break;
          case 4:wsprintf(TempCharVar,"5 MHz");break;
          case 5:wsprintf(TempCharVar,"4 MHz");break;
          case 6:wsprintf(TempCharVar,"3.34 MHz");break;
          case 7:wsprintf(TempCharVar,"2.85 MHz");break;
          case 8:wsprintf(TempCharVar,"2.5 MHz");break;
        }
          /*---- place the point at location which in TempIntVar----*/
        SetDlgItemText(Dlg_h,DCI_name_maxfrq,TempCharVar);
        SendDlgItemMessage (Dlg_h,DCI_name_20M,WM_ENABLE,TRUE,0L);
        SendDlgItemMessage (Dlg_h,DCI_name_10M,WM_ENABLE,TRUE,0L);
        SendDlgItemMessage (Dlg_h,DCI_name_10M3,WM_ENABLE,TRUE,0L);
        SendDlgItemMessage (Dlg_h,DCI_name_5M,WM_ENABLE,TRUE,0L);
        if (TempName.Ch >= 5)
          SendDlgItemMessage (Dlg_h,DCI_name_5M,WM_ENABLE,FALSE,0L);
        if (TempName.Ch >= 3)
  {
          SendDlgItemMessage (Dlg_h,DCI_name_10M3,WM_ENABLE,FALSE,0L);
          SendDlgItemMessage (Dlg_h,DCI_name_10M,WM_ENABLE,FALSE,0L);
  }
        if (TempName.Ch >= 2)
          SendDlgItemMessage (Dlg_h,DCI_name_20M,WM_ENABLE,FALSE,0L);
  return TRUE;
 }

 switch (Dlg_wpara)
 {
        case IDOK:
          switch (State)
          {
      case 1:
            num = (DWORD)SendDlgItemMessage(Dlg_h,DCI_name_name,CB_GETCURSEL,0,0L);
            MainVar[num].Use  = 0;  // save setup value of name
            BWCCMessageBox(Dlg_h,"********* Name are Release now *****","Set Release Name",MB_OK);
            EndDialog (Dlg_h,TRUE);
            return TRUE;
           case 2:
           // display
            strcpy(Title,MainVar[num].Name);
            wsprintf(cBuf," <%d : %d> Display<%d -
%d>",(int)MainVar[num].Start,(int)MainVar[num].End,(int)MainVar[num].Start,(int)MainVar[num].Start+1024);
       strcat(Title,cBuf);
            MDIStructVar.szClass  = "ChromicGraph";
            MDIStructVar.szTitle  = Title;
            MDIStructVar.hOwner   = hInst;
            MDIStructVar.x                = CW_USEDEFAULT;
            MDIStructVar.y                = CW_USEDEFAULT;
            MDIStructVar.cx               = CW_USEDEFAULT;
            MDIStructVar.cy               = CW_USEDEFAULT;
            MDIStructVar.style    = 0;
            MDIStructVar.lParam   = NULL;
            hWndGraph = SendMessage(hWndClient, WM_MDICREATE, 0,
                                    (LONG)(LPMDICREATESTRUCT)&MDIStructVar);
     //Set name handle to WindowWord para for ref. to data
            SetWindowWord(hWndGraph,0,1);
            EndDialog (Dlg_h,TRUE);
            break;
           case 3:
           // ADC
```

```
            num = (DWORD)SendDlgItemMessage(Dlg_h,DCI_name_name,CB_GETCURSEL,0,0L);
/*          if(MainVar[num].Start == 0)
                ClearCounter();
            else
                SetCounter(MainVar[num].Start);
            StopCounter(MainVar[num].End);
            SetChannel(MainVar[num].Ch);
*///         ClearCounter();
            SetFreq(MainVar[num].Freq);
            StartSampling();
//      EndSamping();
        hWnd = GetParent(Dlg_h);
            if(!(nTimer = SetTimer(hWnd,1,1000,NULL)))
         MessageBox(hWnd,"No Timers !","Message",MB_OK);
            EndDialog (Dlg_h,TRUE);
            break;
           case 4:
           //DAC
            num = (DWORD)SendDlgItemMessage(Dlg_h,DCI_name_name,CB_GETCURSEL,0,0L);
            if(MainVar[num].Start == 0)
                ClearCounter();
            else
                SetCounter(MainVar[num].Start);
            StopCounter(MainVar[num].End);
            SetChannel(MainVar[num].Ch);
            SetFreq(MainVar[num].Freq);
            StartPlayback();
        StopPlayback();
            EndDialog (Dlg_h,TRUE);
            break;
          }
          return TRUE;

         case IDCANCEL:
          EndDialog (Dlg_h,TRUE);
          return TRUE;

         case IDHELP:
          MessageBox(Dlg_h," Not SUPPORT now! ","ERROR",MB_ICONEXCLAMATION|MB_OK);
          break;
         case DCI_name_name:
          num = SendDlgItemMessage(Dlg_h,DCI_name_name,CB_GETCURSEL,0,0L);
          if(MainVar[num].Use == 1)
          {
           itoa(MainVar[num].Start,TempCharVar,10);
           SetDlgItemText(Dlg_h,DCI_name_memstart,TempCharVar);
           itoa(MainVar[num].End,TempCharVar,10);
           SetDlgItemText(Dlg_h,DCI_name_memend,TempCharVar);
           if (MainVar[num].End <= MainVar[num].Start)
        SetDlgItemText(Dlg_h,DCI_name_memnum,"--ã°éäÁèä´é--");
            else
            {
        TempLong1 =MainVar[0].End - MainVar[0].Start;
        SetDlgItemText(Dlg_h,DCI_name_memnum,itoa(TempLong1,TempCharVar1,10));
            }
            CheckRadioButton(Dlg_h,DCI_name_20M,DCI_name_13,DCI_name_20M+MainVar[num].Freq);

CheckRadioButton(Dlg_h,DCI_name_maxch1,DCI_name_maxch8,DCI_name_maxch1+MainVar[num].Ch);
            switch(MainVar[0].Ch)
       {
        case 1:wsprintf(TempCharVar,"20 MHz");break;
        case 2:wsprintf(TempCharVar,"10 MHz");break;
        case 3:wsprintf(TempCharVar,"6.67 MHz");break;
        case 4:wsprintf(TempCharVar,"5 MHz");break;
        case 5:wsprintf(TempCharVar,"4 MHz");break;
        case 6:wsprintf(TempCharVar,"3.34 MHz");break;
        case 7:wsprintf(TempCharVar,"2.85 MHz");break;
        case 8:wsprintf(TempCharVar,"2.5 MHz");break;
            }
            /*---- place the point at location whicn in TempIntVar---*/
       SetDlgItemText(Dlg_h,DCI_name_maxfrq,TempCharVar);
       SendDlgItemMessage (Dlg_h,DCI_name_20M,WM_ENABLE,TRUE,0L);
       SendDlgItemMessage (Dlg_h,DCI_name_10M,WM_ENABLE,TRUE,0L);
       SendDlgItemMessage (Dlg_h,DCI_name_10M3,WM_ENABLE,TRUE,0L);
       SendDlgItemMessage (Dlg_h,DCI_name_5M,WM_ENABLE,TRUE,0L);
```

```
        if (TempName.Ch >= 5)
          SendDlgItemMessage (Dlg_h,DCI_name_5M,WM_ENABLE,FALSE,0L);
        if (TempName.Ch >= 3)
        {
          SendDlgItemMessage (Dlg_h,DCI_name_10M3,WM_ENABLE,FALSE,0L);
          SendDlgItemMessage (Dlg_h,DCI_name_10M,WM_ENABLE,FALSE,0L);
        }
        if (TempName.Ch >= 2)
              SendDlgItemMessage (Dlg_h,DCI_name_20M,WM_ENABLE,FALSE,0L);
            }
            break;
          case DCI_name_memend:
            GetDlgItemText(Dlg_h,DCI_name_memstart,TempCharVar,10);
            TempName.Start = atol(TempCharVar);
            GetDlgItemText(Dlg_h,DCI_name_memend,TempCharVar,10);
            TempName.End = atol(TempCharVar);
            if (TempName.End <= TempName.Start)
              SetDlgItemText(Dlg_h,DCI_name_memnum,"--ã"éäÁèä´é--");
            else
      {
              TempLong1 =TempName.End - TempName.Start;
              SetDlgItemText(Dlg_h,DCI_name_memnum,itoa(TempLong1,TempCharVar1,10));
      }
            break;
          default:
            break;
    }
   default: /*------ switch Dlg_msg -------*/
     return FALSE;
 }
 return TRUE;
}
```

```
/*---c_sele_n---*/
BOOL FAR PASCAL DlgProc_name_new (HWND Dlg_h, WORD Dlg_msg , WORD Dlg_wpara, LONG Dlg_lpara)
{
 int  maxch=0,a=0,b=0;
 int  TempIntVar=0;
 float TempFloatVar=0.0;
 char *TempCharVar;
 char TempCharVar1[12];
 BOOL TempBOOL;

 static NAMESTRUCT TempName;

 long TempLong=0,TempLong1=0;

 switch (Dlg_msg)
 {
  case WM_DESTROY:
    return TRUE;

  case WM_INITDIALOG:
```

```
    for (a=0;a<=255;a++)
    {
     if (MainVar[a].Use == 0)
          {
            SetDlgItemText(Dlg_h,DCI_name_name,Name[a]);
       a =260;
     }
    }
    SetDlgItemText(Dlg_h,DCI_name_mode,"New Name");
//    SetDlgItemText(Dlg_h,DCI_name_name,"Test");
    SetDlgItemText(Dlg_h,DCI_name_memnum,"--ã³éäÁèä´é--");

    CheckRadioButton(Dlg_h,DCI_name_20M,DCI_name_13,DCI_name_20M);
    CheckRadioButton(Dlg_h,DCI_name_maxch1,DCI_name_maxch8,DCI_name_maxch1);
    SetDlgItemText(Dlg_h,DCI_name_maxfrq,"20 MHz");
    TempName.Ch  =0;
    TempName.Freq=0;
    break;

   case WM_COMMAND:
    if ( (Dlg_wpara>=DCI_name_20M) && (Dlg_wpara<=DCI_name_13))
    {
          TempName.Freq = (char)(Dlg_wpara-8561);    // 8561 must change when reassign dialog id
          CheckRadioButton(Dlg_h,DCI_name_20M,DCI_name_13,Dlg_wpara);
     return TRUE;
    }
    else if ( (Dlg_wpara>=DCI_name_maxch1) && (Dlg_wpara<=DCI_name_maxch8))
    {
          TempName.Ch = (char)(Dlg_wpara-8260);
          CheckRadioButton(Dlg_h,DCI_name_maxch1,DCI_name_maxch8,Dlg_wpara);
          switch(TempName.Ch)
          {
           case 1:wsprintf(TempCharVar,"20 MHz");break;
           case 2:wsprintf(TempCharVar,"10 MHz");break;
           case 3:wsprintf(TempCharVar,"6.67 MHz");break;
           case 4:wsprintf(TempCharVar,"5 MHz");break;
           case 5:wsprintf(TempCharVar,"4 MHz");break;
           case 6:wsprintf(TempCharVar,"3.34 MHz");break;
           case 7:wsprintf(TempCharVar,"2.85 MHz");break;
           case 8:wsprintf(TempCharVar,"2.5 MHz");break;
          }
           /*— place the point at location whicn in TempIntVar---*/
          SetDlgItemText(Dlg_h,DCI_name_maxfrq,TempCharVar);
          SendDlgItemMessage (Dlg_h,DCI_name_20M,WM_ENABLE,TRUE,0L);
          SendDlgItemMessage (Dlg_h,DCI_name_10M,WM_ENABLE,TRUE,0L);
          SendDlgItemMessage (Dlg_h,DCI_name_10M3,WM_ENABLE,TRUE,0L);
          SendDlgItemMessage (Dlg_h,DCI_name_5M,WM_ENABLE,TRUE,0L);
          if (TempName.Ch >= 5)
           SendDlgItemMessage (Dlg_h,DCI_name_5M,WM_ENABLE,FALSE,0L);
          if (TempName.Ch >= 3)
    {
           SendDlgItemMessage (Dlg_h,DCI_name_10M3,WM_ENABLE,FALSE,0L);
           SendDlgItemMessage (Dlg_h,DCI_name_10M,WM_ENABLE.FALSE,0L);
    }
          if (TempName.Ch >= 2)
           SendDlgItemMessage (Dlg_h,DCI_name_20M,WM_ENABLE,FALSE,0L);
     return TRUE;
    }

    switch (Dlg_wpara)
    {
          case IDOK:
           /*— Find name in listbox, if not found then record in name table -----*/
           GetDlgItemText(Dlg_h,DCI_name_name,(LPSTR)TempCharVar1,7);
           for (a=0;a<=255;a++)
           { if (MainVar[a].Use == 0)
       { for (b=0;b<=7;b++)
             { MainVar[a].Name[b] = *(TempCharVar1+b); }
             MainVar[a].Use  = 1;  // save setup value of name
             MainVar[a].Start = TempName.Start;
             MainVar[a].End = TempName.End;
             MainVar[a].Ch = TempName.Ch;
             MainVar[a].Freq = TempName.Freq;
             BWCCMessageBox(Dlg_h,"********** Name are record now *****","ERROR Set New Name",MB_OK);
             EndDialog (Dlg_h,TRUE);
```

```
                return TRUE;
        }
            }
            if (a==256)
       {
             BWCCMessageBox(Dlg_h,"********** Name Buffer Full! ******\n *** Can't record. Command cancel
***","ERROR Set New Name",MB_OK);
             EndDialog (Dlg_h,TRUE);
            }
           return TRUE;

           case IDCANCEL:
            EndDialog (Dlg_h,TRUE);
            return TRUE;

           case IDHELP:
            MessageBox(Dlg_h," Not SUPPORT now! ","ERROR",MB_ICONEXCLAMATION|MB_OK);
            break;

           case DCI_name_memend:
            GetDlgItemText(Dlg_h,DCI_name_memstart,TempCharVar,10);
            TempName.Start = atol(TempCharVar);
            GetDlgItemText(Dlg_h,DCI_name_memend,TempCharVar,10);
            TempName.End = atol(TempCharVar);
            if ((TempName.End < (TempName.Start+1024)) || (TempName.End > 65535))
             SetDlgItemText(Dlg_h,DCI_name_memnum,"--ã*éäÁèä´é--");
            else
      {
             TempLong1 =TempName.End - TempName.Start;
             SetDlgItemText(Dlg_h,DCI_name_memnum,itoa(TempLong1,TempCharVar1,10));
      }
           break;
    default:
           break;
    }
   default: /*------ switch Dlg_msg -------*/
    return FALSE;
 }
 return TRUE;
}
```

```
/*----c_name_e---*/
BOOL FAR PASCAL DlgProc_name_edit (HWND Dlg_h, WORD Dlg_msg , WORD Dlg_wpara, LONG Dlg_lpara)
{
 int  maxch=0,a=0,b=0;
 int  TempIntVar=0;
 float TempFloatVar=0.0;
 char *TempCharVar;
 char  TempCharVar1[12];
 BOOL  TempBOOL;
 static DWORD num;

 static NAMESTRUCT  TempName;

 long  TempLong=0,TempLong1=0;

 switch (Dlg_msg)
 {
  case WM_DESTROY:
   return TRUE;

  case WM_INITDIALOG:
   num = 0;
   SendDlgItemMessage (Dlg_h,DCI_name_name,CB_RESETCONTENT,0,0L);
   for (TempIntVar=0;TempIntVar<=255;TempIntVar++)
   { if (MainVar[TempIntVar].Use==1)
          SendDlgItemMessage (Dlg_h,DCI_name_name,CB_ADDSTRING,0,(long)MainVar[TempIntVar].Name);
   }
```

```
SendMessage(Dlg_h,CB_SETCURSEL,num,0L);
Itoa(MainVar[0].Start,TempCharVar,10);
SetDlgItemText(Dlg_h,DCI_name_memstart,TempCharVar);
Itoa(MainVar[0].End,TempCharVar,10);
SetDlgItemText(Dlg_h,DCI_name_memend,TempCharVar);
if (MainVar[0].End <= MainVar[0].Start)
  SetDlgItemText(Dlg_h,DCI_name_memnum,"--ã*éäÁèä´é--");
else
{
  TempLong1 =MainVar[0].End - MainVar[0].Start;
  SetDlgItemText(Dlg_h,DCI_name_memnum,itoa(TempLong1,TempCharVar1,10));
}
CheckRadioButton(Dlg_h,DCI_name_20M,DCI_name_13,DCI_name_20M+MainVar[0].Freq);
CheckRadioButton(Dlg_h,DCI_name_maxch1,DCI_name_maxch8,DCI_name_maxch1+MainVar[0].Ch);
switch(MainVar[0].Ch)
{
 case 1:wsprintf(TempCharVar,"20 MHz");break;
 case 2:wsprintf(TempCharVar,"10 MHz");break;
 case 3:wsprintf(TempCharVar,"6.67 MHz");break;
 case 4:wsprintf(TempCharVar,"5 MHz");break;
 case 5:wsprintf(TempCharVar,"4 MHz");break;
 case 6:wsprintf(TempCharVar,"3.34 MHz");break;
 case 7:wsprintf(TempCharVar,"2.85 MHz");break;
 case 8:wsprintf(TempCharVar,"2.5 MHz");break;
}
 /*---- place the point at location whicn in TempIntVar---*/
SetDlgItemText(Dlg_h,DCI_name_maxfrq,TempCharVar);
SendDlgItemMessage (Dlg_h,DCI_name_20M,WM_ENABLE,TRUE,0L);
SendDlgItemMessage (Dlg_h,DCI_name_10M,WM_ENABLE,TRUE,0L);
SendDlgItemMessage (Dlg_h,DCI_name_10M3,WM_ENABLE,TRUE,0L);
SendDlgItemMessage (Dlg_h,DCI_name_5M,WM_ENABLE,TRUE,0L);
if (TempName.Ch >= 5)
  SendDlgItemMessage (Dlg_h,DCI_name_5M,WM_ENABLE,FALSE,0L);
if (TempName.Ch >= 3)
{
 SendDlgItemMessage (Dlg_h,DCI_name_10M3,WM_ENABLE,FALSE,0L);
 SendDlgItemMessage (Dlg_h,DCI_name_10M,WM_ENABLE,FALSE,0L);
}
if (TempName.Ch >= 2)
      SendDlgItemMessage (Dlg_h,DCI_name_20M,WM_ENABLE,FALSE,0L);

TempName.Ch  = MainVar[0].Ch;
TempName.Freq= MainVar[0].Freq;
break;

case WM_COMMAND:
 if ( (Dlg_wpara>=DCI_name_20M) && (Dlg_wpara<=DCI_name_13))
 {
      TempName.Freq = (char)(Dlg_wpara-8561);   // 8561 must change when reassign dialog id
      CheckRadioButton(Dlg_h,DCI_name_20M,DCI_name_13,Dlg_wpara);
  return TRUE;
 }
 else if ( (Dlg_wpara>=DCI_name_maxch1) && (Dlg_wpara<=DCI_name_maxch8))
 {
      TempName.Ch = (char)(Dlg_wpara-8260);
      CheckRadioButton(Dlg_h,DCI_name_maxch1,DCI_name_maxch8,Dlg_wpara);
      switch(TempName.Ch)
      {
       case 1:wsprintf(TempCharVar,"20 MHz");break;
       case 2:wsprintf(TempCharVar,"10 MHz");break;
       case 3:wsprintf(TempCharVar,"6.67 MHz");break;
       case 4:wsprintf(TempCharVar,"5 MHz");break;
       case 5:wsprintf(TempCharVar,"4 MHz");break;
       case 6:wsprintf(TempCharVar,"3.34 MHz");break;
       case 7:wsprintf(TempCharVar,"2.85 MHz");break;
       case 8:wsprintf(TempCharVar,"2.5 MHz");break;
      }
       /*---- place the point at location whicn in TempIntVar---*/
      SetDlgItemText(Dlg_h,DCI_name_maxfrq,TempCharVar);
      SendDlgItemMessage (Dlg_h,DCI_name_20M,WM_ENABLE,TRUE,0L);
      SendDlgItemMessage (Dlg_h,DCI_name_10M,WM_ENABLE,TRUE,0L);
      SendDlgItemMessage (Dlg_h,DCI_name_10M3,WM_ENABLE,TRUE,0L);
      SendDlgItemMessage (Dlg_h,DCI_name_5M,WM_ENABLE,TRUE,0L);
      if (TempName.Ch >= 5)
       SendDlgItemMessage (Dlg_h,DCI_name_5M,WM_ENABLE,FALSE,0L);
```

```
        if (TempName.Ch >= 3)
    {
          SendDlgItemMessage (Dlg_h,DCI_name_10M3,WM_ENABLE,FALSE,0L);
          SendDlgItemMessage (Dlg_h,DCI_name_10M,WM_ENABLE,FALSE,0L);
    }
        if (TempName.Ch >= 2)
          SendDlgItemMessage (Dlg_h,DCI_name_20M,WM_ENABLE,FALSE,0L);
     return TRUE;
    }

    switch (Dlg_wpara)
    {
     case IDOK:
                    num = (DWORD)SendDlgItemMessage(Dlg_h,DCI_name_name,CB_GETCURSEL,0,0L);

//SendDlgItemMessage(Dlg_h,DCI_name_name,CB_GETLBTEXT,num,(DWORD)(LPSTR)TempCharVar);
                    //strcpy(MainVar[num].Name,TempCharVar);
                    GetDlgItemText(Dlg_h,DCI_name_memstart,TempCharVar,10);
                    TempName.Start = atol(TempCharVar);
               GetDlgItemText(Dlg_h,DCI_name_memend,TempCharVar,10);
               TempName.End = atol(TempCharVar);

                    //MainVar[num].Use  = 1;  // save setup value of name
                    MainVar[num].Start = TempName.Start;
                    MainVar[num].End = TempName.End;
                    MainVar[num].Ch = TempName.Ch;
                    MainVar[num].Freq = TempName.Freq;
               BWCCMessageBox(Dlg_h,"********* Name are record now ******","Set Edit Name",MB_OK);
                    EndDialog (Dlg_h,TRUE);
             return TRUE;

          case IDCANCEL:
           EndDialog (Dlg_h,TRUE);
           return TRUE;

          case IDHELP:
           MessageBox(Dlg_h," Not SUPPORT now! ","ERROR",MB_ICONEXCLAMATION|MB_OK);
           break;
          case DCI_name_name:
           num = SendDlgItemMessage(Dlg_h,DCI_name_name,CB_GETCURSEL,0,0L);
           if(MainVar[num].Use == 1)
           {
            itoa(MainVar[num].Start,TempCharVar,10);
            SetDlgItemText(Dlg_h,DCI_name_memstart,TempCharVar);
            itoa(MainVar[num].End,TempCharVar,10);
            SetDlgItemText(Dlg_h,DCI_name_memend,TempCharVar);
            if (MainVar[num].End <= MainVar[num].Start)
        SetDlgItemText(Dlg_h,DCI_name_memnum,"--ã³éäÁèä´é--");
            else
            {
        TempLong1 =MainVar[0].End - MainVar[0].Start;
        SetDlgItemText(Dlg_h,DCI_name_memnum,itoa(TempLong1,TempCharVar1,10));
            }
            CheckRadioButton(Dlg_h,DCI_name_20M,DCI_name_13,DCI_name_20M+MainVar[num].Freq);

CheckRadioButton(Dlg_h,DCI_name_maxch1,DCI_name_maxch8,DCI_name_maxch1+MainVar[num].Ch);
            switch(MainVar[0].Ch)
       {
        case 1:wsprintf(TempCharVar,"20 MHz");break;
        case 2:wsprintf(TempCharVar,"10 MHz");break;
        case 3:wsprintf(TempCharVar,"6.67 MHz");break;
        case 4:wsprintf(TempCharVar,"5 MHz");break;
        case 5:wsprintf(TempCharVar,"4 MHz");break;
        case 6:wsprintf(TempCharVar,"3.34 MHz");break;
        case 7:wsprintf(TempCharVar,"2.85 MHz");break;
        case 8:wsprintf(TempCharVar,"2.5 MHz");break;
            }
            /*--- place the point at location whicn in TempIntVar---*/
       SetDlgItemText(Dlg_h,DCI_name_maxfrq,TempCharVar);
       SendDlgItemMessage (Dlg_h,DCI_name_20M,WM_ENABLE,TRUE,0L);
       SendDlgItemMessage (Dlg_h,DCI_name_10M,WM_ENABLE,TRUE,0L);
       SendDlgItemMessage (Dlg_h,DCI_name_10M3,WM_ENABLE,TRUE,0L);
       SendDlgItemMessage (Dlg_h,DCI_name_5M,WM_ENABLE,TRUE,0L);
       if (TempName.Ch >= 5)
        SendDlgItemMessage (Dlg_h,DCI_name_5M,WM_ENABLE,FALSE,0L);
```

```
        if (TempName.Ch >= 3)
        {
          SendDlgItemMessage (Dlg_h,DCI_name_10M3,WM_ENABLE,FALSE,0L);
          SendDlgItemMessage (Dlg_h,DCI_name_10M,WM_ENABLE,FALSE,0L);
        }
        if (TempName.Ch >= 2)
              SendDlgItemMessage (Dlg_h,DCI_name_20M,WM_ENABLE,FALSE,0L);
           }
           break;
         case DCI_name_memend:
           GetDlgItemText(Dlg_h,DCI_name_memstart,TempCharVar,10);
           TempName.Start = atol(TempCharVar);
           GetDlgItemText(Dlg_h,DCI_name_memend,TempCharVar,10);
           TempName.End = atol(TempCharVar);
           if ((TempName.End < (TempName.Start+1024)) || (TempName.End > 65535))
            SetDlgItemText(Dlg_h,DCI_name_memnum,"--ã*éäÁèä´é--");
           else
      {
            TempLong1 =TempName.End - TempName.Start;
            SetDlgItemText(Dlg_h,DCI_name_memnum,itoa(TempLong1,TempCharVar1,10));
      }
           break;
         default:
          break;
    }
   default: /*------ switch Dlg_msg -------*/
    return FALSE;
 }
 return TRUE;
}
```

```
*----c_io----*
#define IOPORT              0x0200
#define PORTA               IOPORT
#define PORTB        IOPORT+1
#define PORTC               IOPORT+2
#define CTRL8255  IOPORT+3
#define MEMPORT             IOPORT+4
#define REALTIME IOPORT+5

SetIO()
{
 outportb (CTRL8255,80);    // 80h = set all port to out
 outportb (PORTB,0x00);
 outportb (PORTA,0x0f);
 outportb (PORTC,0x00);
}

//------ return = data one byte -------//
ReadIO(long Address)
{
 unsigned char Temp1=Temp2=Temp3=0;

 Temp1=(char)Address & 0x000000ff;
 Temp2=(char)Address & 0x0000ff00; Temp2=Temp2>>8 ;
 Temp3=(char)Address & 0x00ff0000; Temp3=Temp3>>16;
 outportb (PORTB,Temp1);   outportb (PORTC,4);
 outportb (PORTB,Temp1);   outportb (PORTC,5);
 outportb (PORTB,Temp1);   outportb (PORTC,6);
 outportb (PORTC,0);
 return( inportb(MEMPORT));
}

WriteIO(long Address,char Data)
{
 unsigned char Temp1=Temp2=Temp3=0;
```

```
 Temp1=(char)Address & 0x000000ff;
 Temp2=(char)Address & 0x0000ff00; Temp2=Temp2>>8 ;
 Temp3=(char)Address & 0x00ff0000; Temp3=Temp3>>16;
 outportb (PORTB,Temp1);   outportb (PORTC,4);
 outportb (PORTB,Temp1);   outportb (PORTC,5);
 outportb (PORTB,Temp1);   outportb (PORTC,6);
 outportb (PORTC,0);
 return( outportb(MEMPORT,Data));
}

SetOSC(unsigned char mode)  //-- 0 =ON  1=OFF
{
 unsigned char Temp1=0;

 Temp1=inportb(PORTA);
 Temp1= (mode==0) ? Temp1&0xfb: Temp1|0x04;   // 0xfb = 1111 1011
 outportb(PORTA,Temp1);
}

SetADC(unsigned char mode)  //-- 0 =Write  1=Read
{
 unsigned char Temp1=0;

 Temp1=inportb(PORTA);
 Temp1= (mode==0) ? Temp1&0xfd: Temp1|0x02;   // 0xfb = 1111 1011
 outportb(PORTA,Temp1);
}

SetConnectRAM(unsigned char mode)  //-- 0 =CPU  1=ADC
{
 unsigned char Temp1=0;

 Temp1=inportb(PORTA);
 Temp1= (mode==0) ? Temp1&0xfe: Temp1|0x01;   // 0xfb = 1111 1011
 outportb(PORTA,Temp1);
}

ClrCount()
{
 unsigned char Temp1=0;

 Temp1=inportb(PORTA);
 outportb(PORTA,Temp1&0xfe);
 outportb(PORTA,Temp1);

}

SetFreq(unsigned char Freq)
{
 unsigned char Temp1=0;

 outportb(PORTB,Freq);
 Temp1=inportb(PORTA);
 outportb(PORTA,Temp1|0x80);
 outportb(PORTA,Temp1);

}

SetChannel(unsigned char Channel)
{
 unsigned char Temp1=0;

 outportb(PORTB,Channel);
 Temp1=inportb(PORTA);
 outportb(PORTA,Temp1|0x40);
 outportb(PORTA,Temp1);

}
```

long FAR PASCAL ChromicGraphProc(HWND CGP_h, WORD CGP_msg, WORD CGP_wpara, LONG GCP_lpara)

```
{
  HDC             hDC;
  PAINTSTRUCT     ps;
  static HWND  hWndClient,hwndFrame;

  switch(CGP_msg)
  {
   case WM_CREATE:
    hWndClient = GetParent(CGP_h);
    hwndFrame  = GetParent(hWndClient);
    return 0;

   case WM_MDIACTIVATE:
    if (CGP_wpara == TRUE)
        SendMessage (hWndClient, WM_MDISETMENU, 0, MAKELONG(hMenuGraph, hMenuGraphWin) );
    else
        SendMessage (hWndClient, WM_MDISETMENU, 0, MAKELONG(hMenuMain, hMenuMainWin) );
    DrawMenuBar (hwndFrame);
    SendMessage(CGP_h,WM_PAINT,0,0L);
    return 0;

   case WM_PAINT:
    if (!IsIconic(CGP_h))
        GraphRefresh(CGP_h);
    return 0;

   case WM_QUERYENDSESSION:
   case WM_CLOSE:
    if (IDOK != BWCCMessageBox(CGP_h,"Are you sure to Quit?","Quit
Graph",MB_ICONQUESTION|MB_OKCANCEL))
        return 0;
    break;

   case WM_DESTROY:
    return 0;

   case WM_COMMAND:
    switch(CGP_wpara)
    {
        case M_graph_line:
         SetWindowWord(CGP_h,0,1);
      SendMessage(CGP_h,WM_PAINT,0,0L);
         break;
        case M_graph_point:
         SetWindowWord(CGP_h,0,2);
      SendMessage(CGP_h,WM_PAINT,0,0L);
         break;
        case M_graph_bar:
         SetWindowWord(CGP_h,0,3);
      SendMessage(CGP_h,WM_PAINT,0,0L);
         break;
    } // end of WM_COMMAND switch CGP_wpara
    return 0;
  }
  return DefMDIChildProc(CGP_h, CGP_msg, CGP_wpara, GCP_lpara);
}

/*-------------------------------------------------------------*/
long GraphRefresh(HWND hWnd)
{
  int             InDataNum=20;
  char            InData[30]={1,22,34,84,33,75,252,8,19,12,190,78,37,6,75,14,36,26,1,0};

  HDC   hDC;
  RECT            GClient,GWork,GAll,GZoom, UpdateArea;
  GRAPHPARA       gp;

  long            a=0;
  int             FrameThick=10;

  hDC = GetDC(hWnd);
  GetClientRect(hWnd,&GClient);
  GetUpdateRect(hWnd,&UpdateArea,0);  // 0=not erase bkg
  //*--- check for minium frame ------*/
```

```
  GWork.top   = GClient.top   +FrameThick;
  GWork.bottom = GClient.bottom-FrameThick;
  GWork.left   = GClient.left  +FrameThick;
  GWork.right  = GClient.right -FrameThick;

  a = (GWork.bottom-GWork.top-FrameThick)/3;

  GZoom.top   = GWork.top;
  GZoom.bottom = GWork.top+(a<<1);
  GZoom.left   = GWork.left;
  GZoom.right  = GWork.right;

  GAll.top    = GZoom.bottom+FrameThick;
  GAll.bottom  = GAll.top+a;
  GAll.left    = GWork.left;
  GAll.right   = GWork.right - ((GWork.right-GWork.left)/5);

  //-- set para for DisplayGraph routine to show All graph window ----
  //-- and call to display all data graph
  gp.hWnd             = hWnd;
  gp.hDC              = hDC;
  gp.top              = GAll.top;
  gp.bottom           = GAll.bottom;
  gp.left    = GAll.left;
  gp.right            = GAll.right;
  gp.GraphType  = GetWindowWord(hWnd,0);
  gp.BkgColor   = RGB(0,200,0);   gp.GraphColor=RGB(0,0,255);
  gp.GridColor  = RGB(255,0,0);   gp.AxiesColor=gp.GridColor;
  gp.DataNum    = InDataNum;
  gp.InData           = InData;
  gp.MinX = 0;          gp.MaxX=0;        gp.GridX=0;
  gp.MinY = 0;     gp.MinY=0; gp.GridY=0;
  DisplayGraph(gp);

  //-- set para for DisplayGraph routine to show Zoom graph window ----
  //-- and call to display zoom graph
  gp.top              = GZoom.top;
  gp.bottom           = GZoom.bottom;
  gp.left    = GZoom.left;
  gp.right            = GZoom.right;
  gp.GraphType  = GetWindowWord(hWnd,0);
  gp.BkgColor   = RGB(0,200,0);   gp.GraphColor=RGB(0,0,255);
  gp.GridColor  = RGB(255,0,0);   gp.AxiesColor=gp.GridColor;
  gp.DataNum    = InDataNum;
  gp.InData           = InData;
  gp.MinX = 3;          gp.MaxX=15;       gp.GridX=0;
  gp.MinY = 0;     gp.MinY=0; gp.GridY=0;
  DisplayGraph(gp);

  /*------ Tell Windows that area are updated now, stop send WM_PAINT ------*/
  ValidateRect(hWnd,&UpdateArea);

  /*------ Clear unused Objects ---------*/
  ReleaseDC(hWnd,hDC);
 return 0;
 }


 /*-------------------------------------------------------------------

  Use :  set handle, set graph size and type, bkgcolor,graphcolor,gridcolor,axiescolor
         set number of data, pass start add of plot data
         set scale and grid  of x,y or 0=auto

 */
 long DisplayGraph(GRAPHPARA Gp)
 {
  HBRUSH          hBkgBrush;
  HPEN       hGraphPen,hGridPen,hAxiesPen;
  HDC             hDC;

  POINT   PlotVar;

  char            ScaleMarkSize=3,ScaleMarkStepX=5,ScaleMarkStepY=5;
  unsigned char *InData;
```

```
int             InDataMax=0,InDataMin=0,InDataRange=0;
float    InStepMarkX =1.0, InStepMarkY =1.0;
float           InPlotRatioX=0.0, InPlotRatioY=0.0;
int             dxWork=0,dyWork=0;
unsigned long a=0, TempMinX=0, TempMaxX=0, TempLong=0;
char            b=0, c=0, d=0;

InData = Gp.InData;
/*----- Draw Work Area -------*/
hDC = Gp.hDC;
hBkgBrush = CreateSolidBrush(Gp.BkgColor);          //--set work bkg color
SelectObject(hDC,hBkgBrush);
SetBkColor(hDC,hBkgBrush);                                  //--set bk of chr & line
Rectangle(hDC,Gp.left,Gp.top,Gp.right,Gp.bottom);
DeleteObject(hBkgBrush);

/*----- Transform to plot coordinate ------*/
dxWork = Gp.right  - Gp.left;
dyWork = Gp.bottom - Gp.top;

// Scan for range of indata
InDataMax = 0;
InDataMin = 0;

Gp.MinX = (Gp.MinX>Gp.MaxX) ? 0 : Gp.MinX;

for ( a=1; a<=Gp.DataNum; a++,InData++)
 { InDataMax = ( *InData > InDataMax) ?  *InData : InDataMax ;
   InDataMin = ( *InData < InDataMin) ?  *InData : InDataMin ;
 }
InDataRange = 1.1*(InDataMax-InDataMin);

/*------------ Set Scale and Grid axies to selected mode ----------*/
InPlotRatioX = (Gp.MinX == 0) ? (float)Gp.DataNum  : (float)(Gp.MaxX-Gp.MinX) ;
InPlotRatioY = (Gp.MinY == 0) ? (float)InDataRange : (float)(Gp.MaxY-Gp.MinY) ;

InPlotRatioX =  ((float)dxWork / InPlotRatioX  );
InPlotRatioY =  ((float)dyWork / InPlotRatioY  );
//-------------------------------------------------//
if (InPlotRatioX < 3.0)
 { InStepMarkX = (InPlotRatioX<3.0) ? 5.0  : 1.0 ;
   InStepMarkX = (InPlotRatioX<0.6) ? 10.0 : InStepMarkX ; }
if (InPlotRatioY < 3.0)
 { InStepMarkY = (InPlotRatioY<3.0) ? 5.0  : 1.0 ;
   InStepMarkY = (InPlotRatioY<0.6) ? 10.0 : InStepMarkY ; }


ScaleMarkStepX = (Gp.GridX==0) ? 5 : Gp.GridX;    //--- MarkStep = time of unit
ScaleMarkStepY = (Gp.GridY==0) ? 5 : Gp.GridY;

/*----- Set Graph Axies ---must be pass (0,0) both x,y ------*/
hAxiesPen = CreatePen(PS_SOLID,1,Gp.AxiesColor);      //--set axies color
hGridPen  = CreatePen(PS_DOT  ,1,Gp.GridColor );
SelectObject(hDC,hAxiesPen);
MoveTo (hDC,Gp.left ,Gp.top);
LineTo (hDC,Gp.left ,Gp.bottom);
LineTo (hDC,Gp.right,Gp.bottom);
/*----- Set Scale mark on Axies and Grids -------*/
if(Gp.MinX==0)
 { TempMinX=1;      TempMaxX=Gp.DataNum;  InData=Gp.InData;          }
else
 { TempMinX=Gp.MinX; TempMaxX=Gp.MaxX;    InData=Gp.InData+Gp.MinX-1; }

SelectObject(hDC,hAxiesPen);
TempLong = TempMinX; a=Gp.left;
for (d=1 ; a<=Gp.right;d++)
 { c = ScaleMarkSize;
   if ( (TempLong % ScaleMarkStepX)==0)
    { c=c<<1;
      SelectObject(hDC,hGridPen);
           MoveTo(hDC,a,Gp.top); LineTo(hDC,a,Gp.bottom);
      SelectObject(hDC,hAxiesPen);
    }
   MoveTo (hDC,a,Gp.bottom-c);
   LineTo (hDC,a,Gp.bottom+c);
```

```
    TempLong++;
    a = (int)( (float)Gp.left + ( InPlotRatioX*InStepMarkX*(float)d ) );
   }
   b=0; a=Gp.bottom;
   for (d=1 ;a>=Gp.top ;d++)
   { c = ScaleMarkSize;
    if (b==ScaleMarkStepY)
     { c=c<<1;   b=0;
          SelectObject(hDC,hGridPen);
          MoveTo(hDC,Gp.left ,a); LineTo(hDC,Gp.right,a);
          SelectObject(hDC,hAxiesPen);
     }
    MoveTo(hDC,Gp.left-c,a);
    LineTo(hDC,Gp.left+c,a);
    b++;
    a = (int)((float)Gp.bottom - ( InPlotRatioY*InStepMarkY* (float)d) );
   }
   DeleteObject(hGridPen);
   DeleteObject(hAxiesPen);

   /*---- set Pen for draw graph of data ---------*/
   hGraphPen = CreatePen(PS_SOLID,1,Gp.GraphColor);
   SelectObject(hDC,hGraphPen);

   /*---- Plot ----*/
   switch (Gp.GraphType)
   { case 1:          //GT_LINE:
     PlotVar.x =  (int)( (float)Gp.left );
     PlotVar.y =  (int)( (float)Gp.bottom - ((float)(*InData)*InPlotRatioY) );
     MoveTo (hDC,PlotVar.x,PlotVar.y);
     InData++;
     for (a=1;a<=(TempMaxX-TempMinX);a++,InData++)
     { PlotVar.x = (int)( (float)Gp.left  + ( (float)a       *InPlotRatioX) );
          PlotVar.y = (int)( (float)Gp.bottom - ( (float)(*InData)  *InPlotRatioY) );
          LineTo (hDC,PlotVar.x,PlotVar.y);
     }
     break;
    case 2:          //GT_POINT:
     for (a=1;a<=(TempMaxX-TempMinX);a++,InData++)
     { PlotVar.x =  (int)( (float)Gp.left  + ((float)a*InPlotRatioX) );
          PlotVar.y =  (int)( (float)Gp.bottom - ((float)(*InData)*InPlotRatioY ));
          if (PlotVar.y<Gp.top)
       { PlotVar.y =  Gp.top;
          MoveTo (hDC,PlotVar.x-2,PlotVar.y-2);
          LineTo (hDC,PlotVar.x+2,PlotVar.y+2);
          MoveTo (hDC,PlotVar.x+2,PlotVar.y-2);
          LineTo (hDC,PlotVar.x-2,PlotVar.y+2);
       }
          MoveTo (hDC,PlotVar.x-2,PlotVar.y);
          LineTo (hDC,PlotVar.x+2,PlotVar.y);
          MoveTo (hDC,PlotVar.x,PlotVar.y-2);
          LineTo (hDC,PlotVar.x,PlotVar.y+2);
     }
     break;
    case 3:          //GT_BAR:
     for (a=1;a<=(TempMaxX-TempMinX);a++,InData++)
     { PlotVar.x = (int)( (float)Gp.left  + ((float)a       *InPlotRatioX) );
          PlotVar.y = (int)( (float)Gp.bottom - ((float)(*InData)  *InPlotRatioY) );
          PlotVar.y = (PlotVar.y<Gp.top) ? Gp.top : PlotVar.y;
          MoveTo (hDC,PlotVar.x,PlotVar.y);
          LineTo (hDC,PlotVar.x,Gp.bottom);
     }
     break;
    case 4:          //GT_PIE:
     break;
   }                  //-----End of switch GraphType-----
  DeleteObject(hGraphPen);
 return 0;
 }
```

ภาคผนวก ง

โปรแกรมที่ใช้ทดสอบระบบบนดอส

```
Plot.BAS                    REM Routine to test Sampling and Show at highspeed
OUT &H303, &H80
OUT &H300, 0: OUT &H300, 255
OUT &H301, 0: OUT &H301, 255
OUT &H302, 0: OUT &H302, 255

5 INPUT "Enter the frequency code ", a
IF a = 0 THEN 5
OUT &H301, a
OUT &H300, &H4F
OUT &H300, &HF
REM  RESET COUNTER
OUT &H300, &HA
OUT &H300, 3


INPUT "Press any key to sampling Sampling.....", a$
IF a$ <> "n" THEN
 OUT &H300, 9
 FOR a = 1 TO 30000: NEXT
 OUT &H300, &HF
 OUT &H300, 3
 INPUT "Sampling ready press any key to show data...", a$
END IF

SCREEN 12
COLOR 3
LINE (0, 0)-(0, 450)
LINE (0, 450)-(639, 450)
LINE (639, 450)-(639, 0)
LINE (639, 0)-(0, 0)

COLOR 7
  OUT &H301, 0: OUT &H302, 6: OUT &H302, 0
h = 0
10 FOR a! = 0 TO 255
REM OUT &H301, 0: OUT &H302, 6: OUT &H302, 0
 OUT &H301, a!: OUT &H302, 5: OUT &H302, 0
FOR b! = 0 TO 255
REM  OUT &H301, 0: OUT &H302, 6: OUT &H302, 0
  OUT &H301, b!: OUT &H302, 4: OUT &H302, 0
REM  OUT (&H304), b
 c! = INP(&H304)
 LOCATE 25, 10: PRINT "A="; a!, "B="; b!, c!
REM LINE (x, 350)-(x, 350 - c!)
 LINE (x, 350 - c!)-(x, 350 - c!)
 IF x = 635 THEN x = 0
 REM LOCATE 25, 70: INPUT a$
 x = x + 1
 COLOR 0
 LINE (x, 350)-(x, 1)
 COLOR 7
NEXT b!
NEXT a!
```

```
Real.BAS                          REM Routine to Show data Realtime
OUT &H303, &H80
OUT &H300, &HF

SCREEN 12
COLOR 3
LINE (0, 0)-(0, 450)
LINE (0, 450)-(639, 450)
LINE (639, 450)-(639, 0)
LINE (639, 0)-(0, 0)

COLOR 7
x = 0
10 c = INP(&H305)
 LOCATE 24, 20: PRINT x, c
 IF c < 230 THEN
  LINE (x, 350 - c)-(x, 350 - c)
 END IF
 IF x = 635 THEN x = 0
 x = x + 1
 COLOR 0
 LINE (x, 350)-(x, 1)
  COLOR 7
GOTO 10

Show.BAS                          REM Routine to Show data in Memory
5 INPUT "Enter the frequency code ", a
IF a = 0 THEN 5
OUT &H301, a
OUT &H300, &H4F
OUT &H300, &HF
REM  RESET COUNTER
OUT &H300, &HA
OUT &H300, 3


 INPUT "Sampling ready press any key to show data...", a$

SCREEN 12
COLOR 3
LINE (0, 0)-(0, 450)
LINE (0, 450)-(639, 450)
LINE (639, 450)-(639, 0)
LINE (639, 0)-(0, 0)

COLOR 7
  OUT &H301, 0: OUT &H302, 6: OUT &H302, 0
h = 0
10 FOR a! = 0 TO 255
REM OUT &H301, 0: OUT &H302, 6: OUT &H302, 0
 OUT &H301, a!: OUT &H302, 5: OUT &H302, 0
FOR b! = 0 TO 255
REM  OUT &H301, 0: OUT &H302, 6: OUT &H302, 0
  OUT &H301, b!: OUT &H302, 4: OUT &H302, 0
REM  OUT (&H304), b
 c! = INP(&H304)
 LOCATE 25, 10: PRINT "A="; a!, "B="; b!, c!
REM LINE (x, 350)-(x, 350 - c!)
 LINE (x, 350 - c!)-(x, 350 - c!)
 IF x = 635 THEN x = 0
 REM LOCATE 25, 70: INPUT a$
 x = x + 1
 COLOR 0
 LINE (x, 350)-(x, 1)
 COLOR 7
NEXT b!
NEXT a!
```

```
Real.C                          /* Routine to Show data in Realtime quicker than BASIC */
#include <graphics.h>
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>

main()
{
 int gdriver = DETECT, gmode,errorcode;
 int xmax, ymax,x,y;
 unsigned char data;
 outportb (0x303,0x80);
 outportb (0x300,0xf);

 initgraph(&gdriver, &gmode, "");
 errorcode = graphresult();
 setcolor(getmaxcolor());
 xmax = getmaxx();
 ymax = getmaxy();
 line(0,0,0,ymax);
 line(0,ymax,xmax,ymax);
 x=1; y=1;
 do
 {
 data = inportb(0x305);
 data = (data>220) ? 220 : data;
 putpixel(x,350-data,y);
 x++;
 y++;
 y = (y>15) ? 1: y;
 x = (x>xmax-1) ? 2 : x;
 setcolor(0);
 line(x,0,x,ymax-1);
 } while (!kbhit());
 getch();
 closegraph();
 return 0;
}
```