การประยุกต์ใช้เทคนิคการออกแบบระบบทนต่อความผิดพร่องเพื่อพัฒนาวงจรสวิตช์ดิจิทัลสำหรับระบบชุมสายโทรศัพท์

An Application of Fault Tolerant System Design Techniques for Telephone Exchange's Digital Switching Network Development

มนตรี กาญจนะเดชะ

Montri Karnjanadecha

วิทยานิพนธ์วิศวกรรมศาสตรมหาบัณฑิต สาขาวิชาวิศวกรรมไฟฟ้า

มหาวิทยาลัยสงขลานครินทร์

Master of Engineering Thesis in Electrical Engineering

Prince of Songkla University

2538

เลขหมู่ TK9686.D5 ม132 2538 ก.2

Bib Key 90110

ชื่อวิทยานิพนธ์	การประยุกต์ใช้เทคนิคการออกแบบระบบทนต่อความผิดพร่องเพื่อพัฒนาวงจรสวิตช์ดิจิทัลสำหรับระบบชุมสายโทรศัพท์
ผู้เขียน	นายมนตรี กาญจนะเดชะ
สาขาวิชา	วิศวกรรมไฟฟ้า

คณะกรรมการที่ปรึกษา

_____ ประธานกรรมการ
(อาจารย์วีระพันธุ์ มุสิกสาร)

_____ กรรมการ
(ดร.ชูศักดิ์ ลิ่มสกุล)

คณะกรรมการสอบ

_____ ประธานกรรมการ
(อาจารย์วีระพันธุ์ มุสิกสาร)

_____ กรรมการ
(ดร.ชูศักดิ์ ลิ่มสกุล)

_____ กรรมการ
(ดร.เกริกชัย ทองหนู)

_____ กรรมการ
(รองศาสตราจารย์บุญเหลือ พงศ์ดารา)

บัณฑิตวิทยาลัย มหาวิทยาลัยสงขลานครินทร์ อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษา ตามหลักสูตรวิศวกรรมศาสตรมหาบัณฑิต สาขาวิชาวิศวกรรมไฟฟ้า

_____
(ดร.ไพรัตน์ สงวนไทร)
คณบดีบัณฑิตวิทยาลัย

ชื่อวิทยานิพนธ์	การประยุกต์ใช้เทคนิคการออกแบบระบบทนต่อความผิดพร่องเพื่อพัฒนาวงจรสวิตช์ดิจิทัลสำหรับระบบชุมสายโทรศัพท์
ผู้เขียน	นายมนตรี กาญจนะเดชะ
สาขาวิชา	วิศวกรรมไฟฟ้า
ปีการศึกษา	2538

บทคัดย่อ

วิทยานิพนธ์นี้อธิบายการประยุกต์ใช้เทคนิคการออกแบบระบบทนต่อความผิดพร่อง เพื่อพัฒนาวงจรสวิตช์ดิจิทัลสำหรับใช้งานในระบบชุมสายโทรศัพท์สาธารณะ โดยได้เลือกใช้เทคนิคผสมระหว่างเทคนิค Duplication with Comparison กับเทคนิค Standby Sparing (DCSS) ในการออกแบบ วงจรสวิตช์นี้จะถูกนำไปใช้ในระบบชุมสายโทรศัพท์ดิจิทัล TDSS-1R ซึ่งมีการวิจัยและพัฒนาที่ภาควิชาวิศวกรรมคอมพิวเตอร์ วงจรสวิตช์ที่ได้มีลักษณะเป็นมอดูล ในแต่ละมอดูลประกอบด้วยวงจรสับเปลี่ยนช่องสัญญาณเชิงเวลา (Time Switch) ขนาด 1,024 X 1,024 ช่อง ที่ทำงานเหมือนกันจำนวน 2 วงจร และมีวงจรตรวจหาความผิดพลาดซึ่งมีการทำงานเป็นวงจรเปรียบเทียบสัญญาณจากเอาต์พุตของวงจรสวิตช์ทั้งสอง และส่งสัญญาณให้ไมโครโพรเซสเซอร์ที่ควบคุมอยู่รับรู้ โดยผ่านทางวงจรเชื่อมประสาน เมื่อสัญญาณที่เอาต์พุตทั้งสองของวงจรสวิตช์แตกต่างกัน ในการใช้งานจะต้องมีมอดูลของวงจรสวิตช์อย่างน้อย 2 มอดูลจึงจะมีความสามารถทนต่อความผิดพร่องได้ โดยมอดูลหนึ่งจะทำหน้าที่หลัก และอีกมอดูลหนึ่งทำหน้าที่เป็นมอดูลสำรอง ซึ่งสามารถทำงานแทนมอดูลหลักได้ทันทีที่มอดูลหลักทำงานผิดพลาด โดยมีการควบคุมการทำงานของมอดูลด้วยซอฟต์แวร์ มอดูลต่างๆ จะเชื่อมต่อกับไมโครโพรเซสเซอร์โดยทางวงจรแผงหลัง มอดูลวงจรสวิตช์ต้นแบบที่ได้สร้างขึ้น มีคุณสมบัติใกล้เคียงกับข้อกำหนดที่ระบุไว้ในขั้นตอนการออกแบบ จากการทดสอบวงจรสามารถทำงานเป็นวงจรสวิตช์สัญญาณได้อย่างถูกต้อง และสามารถควบคุมให้มอดูลสำรองทำงานแทนมอดูลหลักได้เมื่อมีข้อผิดพลาดในมอดูลหลัก โดยใช้เวลาในการกู้ระบบคืน 38 ไมโครวินาที ซึ่งเป็นระยะเวลาที่สั้นมากพอที่จะไม่สามารถรับรู้สัญญาณรบกวนที่เกิดขึ้น

Thesis Title	An Application of Fault Tolerant System Design Techniques for Telephone Exchange's Digital Switching Network Development
Author	Mr.Montri Karnjanadecha
Major Program	Electrical Engineering
Academic Year	1995

Abstract

The thesis describes an application of fault tolerant system design techniques to develop a public telephone exchange's building block for a digital switching network. The combination of Duplication with Comparison and Standby Sparing (DCSS) techniques were chosen for the switch design. The building block was designed in modules, each module composes of 2 blocks of identical 1,024 X 1,024 channels timeslot interchange circuits. Its intended application is mainly for TDSS-1R digital telephone exchange developed at the Computer Engineering Department. An error detection circuit consisting of a comparator for comparing all parallel output bits of the 2 timeslot interchange circuits. Error signal resulting from the differences between 2 outputs will be sent to the controlling microprocessor via interfacing circuits. In order to be fault tolerant, 2 modules are required, the first one works as a main module and the second one works as a spare module which will take over the main module's operation when the main module fails. All actions were controlled by software running on the controlling microprocessor. A backplane was used to interface modules to the microprocessor. A prototype circuit developed has almost all the characteristics stated in the design specifications. From the results of extensive testing, it can perform switching functions correctly, the main module and the spare module can work together well and behave as a fault tolerant subsystem with recovery time of 38 microsecond. The time was short enough that noise generated caused by module switching could not be heard by human ears.

กิตติกรรมประกาศ

ผู้วิจัยขอแสดงความขอบคุณต่อ อ.วีระพันธุ์ มุสิกสาร ประธานกรรมการที่ปรึกษา, ดร.ชูศักดิ์ ลิ่มสกุล อาจารย์ที่ปรึกษาร่วม, ดร.เกริกชัย ทองหนู และรองศาสตราจารย์บุญเหลือ พงศ์ดารา ที่กรุณาให้ข้อมูลและคำแนะนำตลอดจนความช่วยเหลืออย่างดียิ่งในการทำวิทยานิพนธ์ และขอขอบคุณคณาจารย์และเจ้าหน้าที่ทุกท่านของภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยสงขลานครินทร์ ที่ได้เอื้อเพื่อสถานที่ และให้หยิบยืมอุปกรณ์และเครื่องมือเครื่องใช้ต่างๆ ในการทำวิจัย

สุดท้ายนี้ผู้วิจัยขอกราบขอบพระคุณบิดา มารดา และขอขอบคุณพี่ น้อง และภรรยาที่ได้ให้ความรัก ความอบอุ่น ความช่วยเหลือ และเป็นกำลังใจที่ดีตลอดมา

มนตรี กาญจนะเดชะ

สารบาญ

รายการตาราง

รายการภาพประกอบ

รายการภาพประกอบ (ต่อ)

รายการภาพประกอบ (ต่อ)

ตัวย่อและสัญลักษณ์

CM	=	Connection Memory
DCSS	=	Duplication with Comparison and Standby Sparing
DM	=	Data Memory
IC	=	Integrated Circuit
NMR	=	N-Modular Redundancy
PAC	=	Parallel Access Circuit
PAL	=	Programmable Array Logic
PCM	=	Pulse Code Modulation
RAM	=	Random Access Memory
SLMA	=	Subscriber Line Module Analog
SMM	=	Switch Matrix Module
TDM	=	Time Division Multiplex
TMR	=	Triple Modular Redundancy

อภิธานศัพท์

active	แอ็กทีฟ
address	เลขที่อยู่
address bus	บัสเลขที่อยู่
analog	แอนะล็อก
application	งานประยุกต์
availability	ความพร้อม
bit	บิต
buffer	บัฟเฟอร์, กันชน
bus	บัส
channel	ช่อง
clock	นาฬิกา
comparator	วงจรเปรียบเทียบ
compile	คอมไพล์
component	ชิ้นส่วน, ส่วนประกอบ
connector	ขั้วต่อ
control	ควบคุม
control signal	สัญญาณควบคุม
control word	คำควบคุม
counter	ตัวนับ
data	ข้อมูล
data bus	บัสข้อมูล
design	ออกแบบ, การออกแบบ
detector	วงจรตรวจจับ
digital	ดิจิทัล
disable	ปิดทาง
electronic	อิเล็กทรอนิกส์
enable	เปิดทาง
error	ความผิดพลาด, ข้อผิดพลาด
error containment	การจำกัดขอบเขตความผิดพลาด

อภิธานศัพท์ (ต่อ)

error detection	การตรวจหาความผิดพลาด
error location	การบอกตำแหน่งความผิดพลาด
error recovery	การกู้ความผิดพลาด
failure	ความล้มเหลว
fault	ความผิดพร่อง
fault tolerant	ทนต่อความผิดพร่อง
flowchart	ผังงาน
frame	กรอบ, เฟรม
frame rate	อัตราเร็วเฟรม
hardware	ฮาร์ดแวร์
IC	ไอซี
information	สนเทศ, สารสนเทศ, สารนิเทศ
initialization	การกำหนดค่าเริ่มต้น
initialize	กำหนดค่าเริ่มต้น
input	อินพุต
kilobyte	กิโลไบต์
latch	แลตช์
logic	ลอจิก, ตรรกะ
majority voting	(การลง) คะแนนเสียงส่วนใหญ่
memory	หน่วยความจำ
message	ข้อความ
microcomputer	ไมโครคอมพิวเตอร์
microprocessor	ไมโครโพรเซสเซอร์
microsecond	ไมโครวินาที
millisecond	มิลลิวินาที
module	มอดูล
multiplex	มัลติเพล็กซ์
output	เอาต์พุต
PAL	พีเอแอล

parallel	ขนาน
passive	พาสซีฟ
performability	ความสามารถในการทำงาน
permanent fault	ข้อผิดพร่องถาวร
RAM (Random Access Memory)	แรม (หน่วยความจำเข้าถึงโดยสุ่ม)
recovery	การกู้
redundancy	การซ้ำซ้อน
reliability	ความเชื่อถือได้
reset	ตั้งใหม่
safety	ความปลอดภัย
serial	อนุกรม
shielding	การชิลด์
shift register	เรจิสเตอร์แบบเลื่อน
slot	สล็อต
software	ซอฟต์แวร์
source code	รหัสต้นฉบับ
stream	กระแส (ข้อมูล)
switch	สวิตช์
switching	การสวิตช์
synchronization	การทำให้เข้าจังหวะ
synchronize	ทำให้เข้าจังหวะ
technique	เทคนิค
testability	การทดสอบได้
timer	ตัวจับเวลา
timeslot	ช่องเวลา
timeslot interchange circuit	วงจรสับเปลี่ยนช่องเวลา
transient fault	ข้อผิดพร่องชั่วครู่
voter	(วงจร) ลงคะแนนเสียง

บทที่ 1

บทนำ

ความสำคัญและที่มาของหัวข้อการวิจัย

การสร้างระบบดิจิทัลในอดีตนั้นมีข้อจำกัดอยู่มาก อันเนื่องมาจากเทคโนโลยีที่ใช้ในการออกแบบและการสร้าง การที่จะสร้างระบบที่มีความเชื่อถือได้ (reliability) ในการทำงานสูงนั้นจำเป็นจะต้องใช้เทคนิคในการออกแบบวงจรที่ดี และจะต้องนำไปสร้างด้วยอุปกรณ์ที่ทันสมัย ซึ่งด้วยข้อจำกัดทางเทคโนโลยีในด้านต่างๆ ทำให้ไม่สามารถหาอุปกรณ์ที่มีคุณสมบัติตามต้องการได้ จากเหตุผลดังกล่าว ระบบที่มีความเชื่อถือได้สูงจึงต้องใช้กระบวนการออกแบบที่พิถีพิถัน และเลือกใช้แต่อุปกรณ์ที่มีคุณภาพสูง อันจะมีผลทำให้การสร้างระบบต้องใช้งบประมาณสูงมาก จึงพบว่ามีการนำระบบที่มีความเชื่อถือได้สูงไปใช้ในงานที่สำคัญมากๆ เท่านั้น

ปัจจุบันเทคโนโลยีต่างๆ ได้เจริญรุดหน้าไปมาก โดยเฉพาะเทคโนโลยีทางอิเล็กทรอนิกส์ เป็นผลทำให้อุปกรณ์อิเล็กทรอนิกส์ในปัจจุบันจึงมีราคาถูกลงในขณะที่มีความเชื่อถือได้สูงขึ้น กอปรกับความก้าวหน้าทางเทคโนโลยีคอมพิวเตอร์ ซึ่งมีการนำมาประยุกต์ใช้ในหลายแขนงวิชารวมทั้งการออกแบบระบบดิจิทัล ทำให้นักออกแบบสามารถออกแบบระบบได้อย่างมีประสิทธิภาพ

ในยุคข้อมูลข่าวสารเช่นในปัจจุบัน ข้อมูลข่าวสารต่างๆ ของมนุษย์ไม่ว่าจะอยู่ในรูปแบบใด นับว่ามีความสำคัญอย่างยิ่ง การนำเอาเครื่องมือเครื่องใช้ทางอิเล็กทรอนิกส์มาใช้ในการเก็บบันทึก วิเคราะห์ หรือส่งผ่านข้อมูล จึงจำเป็นจะต้องใช้เครื่องมือที่มีความเชื่อถือได้สูง เพื่อป้องกันการผิดพลาดหรือสูญหายของข้อมูลอันจะนำมาซึ่งผลเสียต่างๆ แก่มนุษย์

ด้วยปัจจัยหลายอย่างทำให้ระบบหรือเครื่องมือเครื่องจักรต่างๆ ที่มนุษย์ประดิษฐ์ขึ้น ไม่ว่าจะถูกออกแบบด้วยเทคนิคที่ดีเลิศเพียงใด หรือใช้วัสดุในการสร้างที่มีคุณภาพสูงเพียงใด ก็ย่อมมีโอกาสที่จะเกิดข้อผิดพลาดในการทำงานขึ้นได้ เมื่อความผิดพลาดเป็นสิ่งที่หลีกเลี่ยงไม่ได้ นักออกแบบจึงหาวิธีการที่จะออกแบบระบบให้ยังคงสามารถทำงานได้แม้ว่าจะมีความผิดพลาดเกิดขึ้นบางส่วนในระบบ ระบบที่สามารถทำงานได้เมื่อมีข้อผิดพลาดเกิดขึ้นเรียกว่าระบบทนต่อความผิดพร่องได้ (fault tolerant system)

เทคนิคการออกแบบระบบทนต่อความผิดพร่องได้มีการคิดค้นมากว่า 40 ปีแล้ว และมีการคิดค้นเทคนิคใหม่ๆ และมีการนำมาใช้ในการออกแบบระบบต่างๆ มากขึ้นในปัจจุบัน (Toy, W.N. 1978.) (Johnson, B.W. 1989.) (Avizienis, A. 1978.) (Sewiorek, D.P. 1991.) ในขณะที่มีการวิจัยและพัฒนาระบบทนต่อความผิดพร่องได้กันมากในต่างประเทศ แต่การวิจัยแขนงนี้ยังไม่ค่อยมีมากนักในประเทศไทย

ผู้วิจัยจึงมีความมุ่งหวังที่จะพัฒนาการวิจัยในสาขานี้ขึ้น ซึ่งในงานวิจัยนี้จะไม่เน้นที่การสร้างเทคนิคใหม่ แต่จะเป็นลักษณะของการติดตามเทคโนโลยี โดยการศึกษาเทคนิคต่างๆ ที่มีอยู่แล้วเลือกวิธีที่เหมาะสมมาใช้

การพิสูจน์ทฤษฎีหรือแนวความคิดใดๆ เมื่อไม่สามารถแสดงด้วยสมการคณิตศาสตร์ หรือการสร้างแบบจำลองได้ ก็อาจทำได้โดยการสร้างเครื่องต้นแบบ (prototype) ขึ้นเพื่อทดสอบการทำงาน จากนั้นจึงจะสรุปผลของทฤษฎีหรือแนวความคิดนั้นออกมา ในการวิจัยนี้จะเป็นการนำเอาเทคนิคการออกแบบระบบทนต่อความผิดพร่องได้มาสร้างวงจรสวิตช์แบบดิจิทัลสำหรับระบบชุมสายโทรศัพท์สาธารณะ เหตุผลที่เลือกการพิสูจน์ทฤษฎีและแนวความคิดด้วยวงจรนี้คือ ระบบชุมสายโทรศัพท์เป็นระบบอิเล็กทรอนิกส์ระบบหนึ่งที่ต้องการความเชื่อถือได้ในการทำงานสูง จะต้องสามารถให้บริการผู้ใช้ได้ตลอดเวลา เพื่อไม่ให้การติดต่อสื่อสารต้องหยุดชะงัก ประกอบกับมีงานวิจัยเกี่ยวกับการพัฒนาระบบชุมสายโทรศัพท์สาธารณะ เพื่อใช้งานเป็นระบบชุมสายในชนบท (วีระพันธุ์ มุสิกสาร และคณะ. 2535.) ที่มีการดำเนินการอยู่ที่ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยสงขลานครินทร์ จึงนับเป็นโอกาสดีที่จะนำเทคนิคการออกแบบระบบทนต่อความผิดพร่องได้มาประยุกต์ใช้ อันจะเป็นผลดีในการติดตามเทคโนโลยีเพื่อการพัฒนาประเทศ

การตรวจเอกสาร

จากการศึกษาบทความวิจัยต่างๆ พบว่าการออกแบบระบบทนต่อความผิดพร่องได้นั้นจะทำการออกแบบทั้งระบบให้มีความทนต่อความผิดพร่องได้ แต่เนื่องจากระบบชุมสายโทรศัพท์เป็นระบบที่มีขนาดใหญ่ การออกแบบระบบทั้งหมดให้มีความสามารถดังกล่าว ไม่สามารถกระทำได้โดยผู้วิจัยเพียงคนเดียวภายใต้งบประมาณ และเวลาวิจัยที่จำกัด งานวิจัยนี้จึงเน้นที่การพัฒนาส่วนย่อยส่วนหนึ่งของระบบเท่านั้น ทั้งนี้เพื่อให้การพิสูจน์ทฤษฎีและแนวความคิดสำเร็จได้ในเงื่อนไขที่จำกัด

วัตถุประสงค์

การวิจัยนี้มีวัตถุประสงค์เพื่อประยุกต์ใช้การออกแบบระบบทนต่อความผิดพร่องได้สำหรับพัฒนาวงจรต้นแบบของวงจรสวิตช์ดิจิทัลสำหรับใช้งานในระบบชุมสายโทรศัพท์ โดยใช้วงจรรวมที่ทันสมัย สำหรับเป็นพื้นฐานในการพัฒนาระบบดิจิทัลอื่นๆ ที่มีความเชื่อถือได้สูงและทนต่อความผิดพร่องได้ต่อไป

บทที่ 2

ระบบทนต่อความผิดพร่อง

ความสำคัญของระบบทนต่อความผิดพร่อง

ในการสร้างระบบดิจิทัลเพื่อประยุกต์ใช้ในงานต่างๆ ของมนุษย์นั้น สิ่งหนึ่งที่ไม่สามารถหลีกเลี่ยงได้ก็คือข้อผิดพร่อง (fault) หรือข้อผิดพลาด (error) ต่างๆ แม้ว่าเมื่อเริ่มแรกที่นำระบบไปใช้งานนั้น ระบบยังทำงานได้ถูกต้องสมบูรณ์ทุกประการ เมื่อเวลาผ่านไป สิ่งแวดล้อมเปลี่ยนไป และอายุการใช้งานของระบบเพิ่มขึ้นเรื่อยๆ ความบกพร่องของส่วนประกอบต่างๆ ของระบบย่อมมีโอกาสเกิดขึ้นได้ ความบกพร่องต่างๆ ไม่ว่าจะเป็นทางฮาร์ดแวร์หรือทางซอฟต์แวร์ก็ตาม อาจเกิดจากการใช้งานไม่ถูกต้องหรือเกิดจากการออกแบบที่ไม่ถูกต้อง โดยในขั้นตอนการทดสอบการทำงานนั้นไม่สามารถตรวจพบข้อผิดพลาดเหล่านั้นได้ จนกว่าจะมีการนำระบบไปใช้งานจริง

เมื่อความบกพร่องหรือความผิดพลาดเป็นสิ่งที่ไม่สามารถหลีกเลี่ยงได้ ผู้ออกแบบจึงได้คิดค้นเทคนิคต่างๆ ที่ใช้ในการออกแบบ เพื่อให้ระบบมีข้อผิดพร่องน้อยที่สุด วิธีการนี้เรียกว่าการหลีกเลี่ยงข้อผิดพลาด (error avoidance) ในระบบที่มีความซับซ้อนมากๆ การหลีกเลี่ยงข้อผิดพลาดได้สามารถทำได้อย่างสมบูรณ์ นักออกแบบจึงใช้เทคนิคการซ่อนข้อผิดพลาด (error hiding) เพื่อให้ข้อผิดพลาดที่เกิดขึ้นไม่กระทบกระเทือนการทำงานของระบบ เมื่อใช้วิธีนี้ระบบจะทำงานต่อไปเสมือนว่าไม่มีข้อผิดพลาดใดๆ เกิดขึ้น

เมื่อการซ่อนข้อผิดพลาดไม่สามารถกระทำได้สมบูรณ์ ระบบก็จะทำงานไม่ถูกต้อง วิธีการสุดท้ายของนักออกแบบคือการหาเทคนิค ที่จะออกแบบระบบอย่างไรจึงจะให้ระบบสามารถทำงานได้ถึงแม้ว่าจะมีข้อผิดพร่องหรือข้อผิดพลาดเกิดขึ้นในระบบ ระบบที่มีความสามารถดังกล่าวเรียกว่าระบบทนต่อความผิดพร่อง (fault tolerant system)

งานประยุกต์หลักของระบบทนต่อความผิดพร่อง คืองานที่มีความสำคัญต่อความปลอดภัยของชีวิตมนุษย์ และงานที่จะต้องสูญเสียทรัพย์สินจำนวนมากเมื่อระบบทำงานผิดพลาด ตัวอย่างเช่น ระบบคอมพิวเตอร์ในยานอวกาศ ระบบควบคุมเครื่องบิน ระบบรายงานสภาพผู้ป่วยในโรงพยาบาล ระบบควบคุมปฏิกรณ์ปรมาณู ระบบงานธนาคาร และระบบชุมสายโทรศัพท์สาธารณะ เป็นต้น

ผลของการใช้เทคนิคการออกแบบระบบทนต่อความผิดพร่อง

เมื่อนำเทคนิคการออกแบบระบบทนต่อความผิดพร่องมาใช้จะสามารถเพิ่มคุณสมบัติต่างๆ ของระบบดังนี้

1. Reliability R(t) หรือความเชื่อถือได้เป็นฟังก์ชันของเวลา คือความน่าจะเป็นที่ระบบจะทำงานได้อย่างถูกต้อง ตลอดช่วงเวลาหนึ่ง ความเชื่อถือได้ถูกนำไปใช้มากในการบ่งบอกคุณสมบัติของระบบที่ไม่สามารถยอมรับความผิดพลาดในช่วงเวลาสั้นๆ และไม่สามารถซ่อมแซมได้ ช่วงเวลาที่กำหนดค่าความเชื่อถือได้ในแต่ละระบบจะมีค่าไม่เท่ากัน เช่นในยานอวกาศ ช่วงเวลาอาจนานเป็น 10 ปี แต่ในเครื่องบินอาจนานเพียงไม่กี่ชั่วโมงเท่านั้น แต่ค่าความเชื่อถือได้สามารถมีค่าสูงถึง 0.9999999 หรือสูงกว่า

2. Availability A(t) เป็นฟังก์ชันของเวลา คือความน่าจะเป็นที่ระบบสามารถทำงานได้อย่างถูกต้อง และสามารถจัดหาได้ เพื่อเรียกใช้งานในช่วงเวลาสั้นๆ ตัวอย่างระบบที่ต้องการ Availability สูง คือระบบคำนวณแบบแบ่งเวลา (time-shared computing system) ระบบซื้อขาย (transaction) เป็นต้น ในระบบดังกล่าว ผู้ใช้ระบบต้องการเรียกใช้งานระบบเพียงชั่วครู่เท่านั้น ระบบจะต้องสามารถทำงานได้อย่างถูกต้องในช่วงเวลานั้น ส่วนในช่วงที่ไม่มีการเรียกใช้งานระบบอาจทำงานไม่ถูกต้องก็ได้ แต่ระบบจะต้องมีความสามารถที่จะซ่อมแซมตัวเองให้ทันเพื่อรอเรียกใช้งานต่อไป

3. Safety S(t) เป็นความน่าจะเป็นที่ระบบจะสามารถทำงานได้อย่างถูกต้อง และหยุดการทำงานโดยไม่มีผลกระทบต่อส่วนอื่นๆ ของระบบ หรือให้ความปลอดภัยกับผู้ใช้งาน

4. Performability P(L,t) เป็นฟังก์ชันของเวลา นิยามโดยความน่าจะเป็นที่ระบบจะทำงานในระดับที่เท่ากับหรือสูงกว่าระดับ L ในช่วงเวลาขณะใดขณะหนึ่ง ถ้ากล่าวถึง Performability ในระบบคอมพิวเตอร์ที่มีหลายโปรเซสเซอร์แล้ว อาจกล่าวได้ว่า ระดับของการทำงานของระบบคือจำนวนโปรเซสเซอร์ที่สามารถทำงานได้ขณะนั้น

5. Maintainability M(t) เป็นการวัดความง่ายของการซ่อมแซม เมื่อระบบเสียหาย หรือ กล่าวได้อีกอย่างหนึ่งว่า Maintainability เป็นความน่าจะเป็นที่ระบบที่ล้มเหลวจะสามารถกู้คืนให้สามารถกลับมาทำงานได้ในช่วงเวลาที่กำหนด

6. Testability การทดสอบคือการวัดคุณภาพและการวัดความมีอยู่ของค่าบางอย่างในระบบ เช่น ถ้าคอมพิวเตอร์เครื่องหนึ่งถูกกำหนดว่าจะต้องทำงานได้ หนึ่งล้านคำสั่งในหนึ่งวินาที เราจะต้องออกแบบวิธีที่จะทดสอบให้ได้ว่าคอมพิวเตอร์นั้นทำงานได้ตามนั้นจริงๆ

7. Dependability เป็นคุณภาพของการบริการที่ระบบมีให้ การหาค่า Reliability Availability Safety Maintainability Performability และ Testability เป็นการวัดเพื่อใช้สำหรับหาค่า Dependability ของระบบ

งานประยุกต์ของระบบทนต่อความผิดพร่อง

ในปัจจุบันได้มีการสร้างระบบที่มีความทนต่อความผิดพร่องมากขึ้นเนื่องจากผู้ออกแบบมีความเข้าใจในระบบลักษณะนี้มากขึ้น เพราะว่าได้มีการวิจัยและพัฒนาในสาขานี้กันมาก รวมทั้งมีการพัฒนาการออกแบบวงจรรวมขนาดใหญ่มากกันมากขึ้น ซึ่งได้มีการนำเอาเทคนิคการออกแบบระบบที่มีความทนต่อความผิดพร่องมาใช้ ทำให้มีเทคนิคต่างๆ เพิ่มขึ้นมากมาย งานประยุกต์ของระบบประเภทนี้สามารถแบ่งออกได้เป็น 4 ประเภทคือ

1. ระบบที่ต้องถูกใช้งานเป็นเวลานาน เช่น ยานอวกาศ ดาวเทียม เป็นต้น ระบบประเภทนี้ส่วนมากแล้วจะถูกสร้างให้มี Reliability ไม่ต่ำกว่า 0.95 ตลอดช่วงเวลาทำงาน 10 ปี

2. ระบบที่มีการคำนวณแบบวิกฤต ส่วนมากจะนำไปใช้เพื่อความปลอดภัยของมนุษย์ และสิ่งแวดล้อม เช่น ในเครื่องบินพาณิชย์ ในกระสวยอวกาศ เป็นต้น โดยส่วนมากระบบชนิดนี้จะต้องมี Reliability 0.9999999

3. ระบบที่สามารถเลื่อนเวลาการบำรุงรักษาได้ ใช้มากในระบบที่มีค่าบำรุงรักษาแพงมาก ไม่สะดวกหรือยากแก่การบำรุงรักษา เช่นระบบประมวลผลระยะไกล และงานประยุกต์ทางอวกาศบางชนิดเป็นต้น

4. ระบบที่ต้องมีความพร้อมแก่การให้บริการผู้ใช้สูง เช่นระบบธนาคาร และระบบชุมสายโทรศัพท์สาธารณะ เป็นต้น

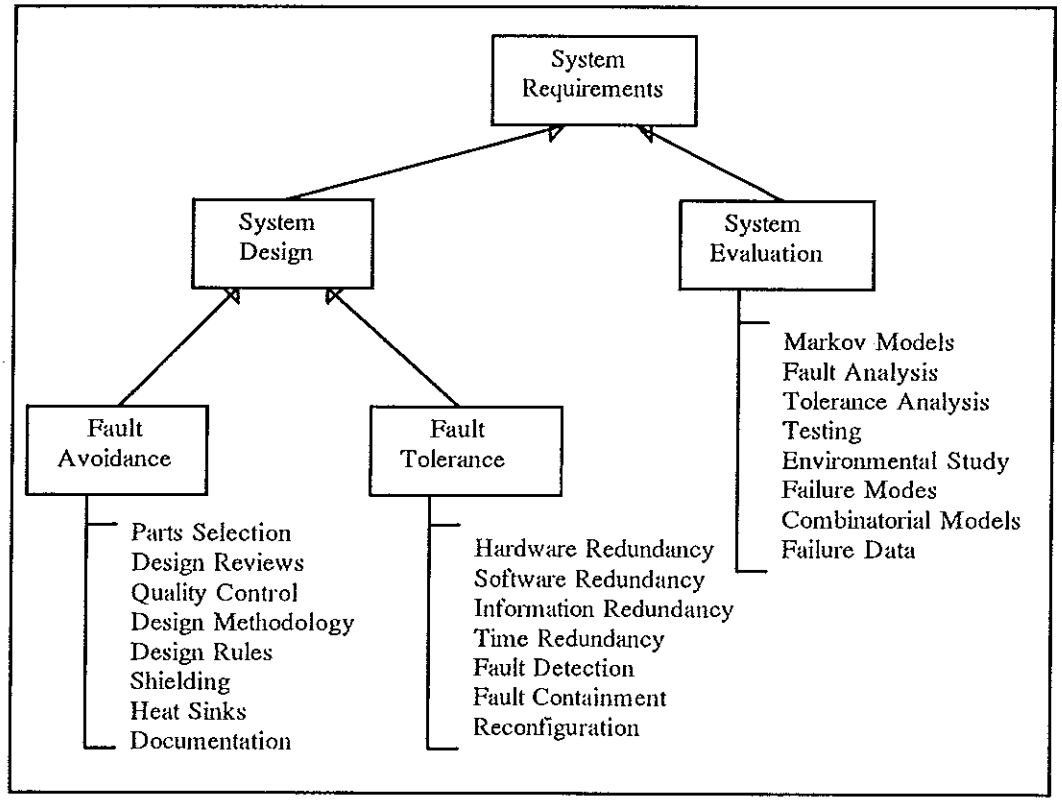
กระบวนการออกแบบ

ในการสร้างระบบนั้นอาจมีทางเลือกอยู่หลายทางในการสร้างระบบให้มีความสามารถในการทำงานตามต้องการ แต่ทั้งนี้ทั้งนั้นเราจะต้องพิจารณาสิ่งต่างๆ เหล่านี้ด้วย คือ ราคา ประสิทธิภาพ ความง่ายแก่การทดสอบ ความยากง่ายในการซ่อม ฯลฯ ดังนั้นการเลือกวิธีใดวิธีหนึ่งเพื่อนำมาสร้างระบบนั้นจะต้องคำนึงถึงปัจจัยต่างๆ ดังกล่าว

เพื่อให้ได้ระบบที่ต้องการเราจะเริ่มต้นจากการกำหนดสิ่งที่ต้องการเสียก่อน ซึ่งเราจะต้องกำหนดงบประมาณ และอัตราสิ้นเปลืองกำลังไฟฟ้า Reliability Availability ฯลฯ

ภาพประกอบ 1 แสดงกระบวนการออกแบบระบบทนต่อความผิดพร่อง การที่จะได้มาซึ่งระบบที่มีคุณสมบัติตามต้องการนั้นได้มาจาก 2 ขั้นตอน คือขั้นตอนการออกแบบระบบ (system design) และการประเมินระบบ (system evaluation)

การออกแบบระบบจะมีการใช้วิธีการหลีกเลี่ยงข้อผิดพลาด (fault avoidance) และ วิธีการทนทานต่อความผิดพลาดได้ (fault tolerance) เทคนิคการหลีกเลี่ยงข้อผิดพลาดสามารถทำได้โดยการการใช้อุปกรณ์ที่มีคุณภาพสูง การใช้กฎการออกแบบที่ดี และมีการทบทวนการออกแบบตลอดเวลา และใช้เครื่องมือช่วยการออกแบบที่มีประสิทธิภาพ ตัวอย่างเทคนิคที่ใช้ในการออกแบบระบบทนต่อความผิดพร่องคือการซ้ำซ้อนทางฮาร์ดแวร์ และการลงคะแนนเสียงส่วนใหญ่



ภาพประกอบ 1 แสดงกระบวนการออกแบบ

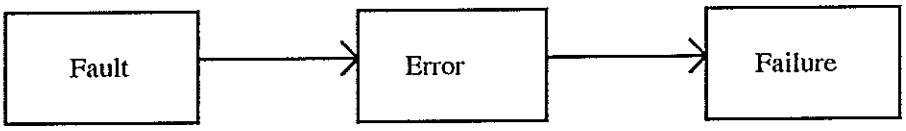
นิยามเบื้องต้น

เพื่อให้มีความเข้าใจถูกต้องตรงกัน จึงต้องมีการนิยามความหมายของคำต่างๆ ที่ใช้ในการออกแบบระบบทนต่อความผิดพร่องได้ดังนี้

1. ข้อผิดพร่อง (fault) คือสิ่งผิดปกติทางกายภาพ หรือความไม่สมบูรณ์ในบางส่วนของฮาร์ดแวร์ หรือซอฟต์แวร์ เช่นการลัดวงจรของตัวนำบนแผ่นวงจรพิมพ์ หรือสิ่งผิดปกติของโปรแกรมเป็นต้น

2. ความผิดพลาด (error) คือผลที่เกิดจากข้อผิดพร่อง ตัวอย่างเช่น ที่จุดหนึ่งของวงจรเกิดการลัดวงจรจนมีลอจิก 1 ตลอดเวลา (stuck at logic 1) เมื่อมีเงื่อนไขบางอย่างที่จุดนั้นจะต้องมีลอจิก 0 ดังนั้นจะเกิดความผิดพลาดขึ้นที่จุดนั้น

3. ความล้มเหลว (failure) เมื่อมีความผิดพลาดเกิดขึ้น ย่อมส่งผลทำให้ระบบไม่สามารถทำงานได้อย่างถูกต้อง จึงทำให้ระบบล้มเหลว หรือไม่สามารถทำงานได้



ภาพประกอบ 2 แสดงความสัมพันธ์ของ ข้อผิดพร่อง ความผิดพลาด และความล้มเหลว

ข้อผิดพร่องในระบบสามารถเกิดได้จากสิ่งต่างๆ มากมาย เช่น จากอุปกรณ์อิเล็กทรอนิกส์ เกิดจากอุปกรณ์ภายนอก หรือเกิดจากขั้นตอนการออกแบบ เราอาจแบ่งสาเหตุของข้อผิดพร่องให้ชัดเจนยิ่งขึ้นดังนี้

1. ความผิดพลาดจากการกำหนดคุณสมบัติ (specification mistakes) เกิดจากการการใช้อัลกอริทึมไม่ถูกต้อง สถาปัตยกรรมไม่ถูกต้อง หรือการกำหนดคุณลักษณะทางฮาร์ดแวร์และซอฟต์แวร์ไม่ถูกต้อง ตัวอย่างเช่นในขั้นตอนการออกแบบวงจรดิจิทัล ผู้ออกแบบอาจจะออกแบบตารางจังหวะเวลาของวงจรไม่ถูกต้องทำให้เมื่อมีการสร้างวงจรขึ้นมาจริงวงจรไม่สามารถทำงานได้

2. ความผิดพลาดจากการนำไปสร้าง (implementation mistakes) เป็นความผิดพลาดที่เกิดจากการนำเอาส่วนที่ได้ออกแบบไว้แล้วไปสร้างได้ไม่ถูกต้อง เช่นมีความผิดพลาดของลายวงจรบนแผ่นวงจรพิมพ์ เป็นต้น

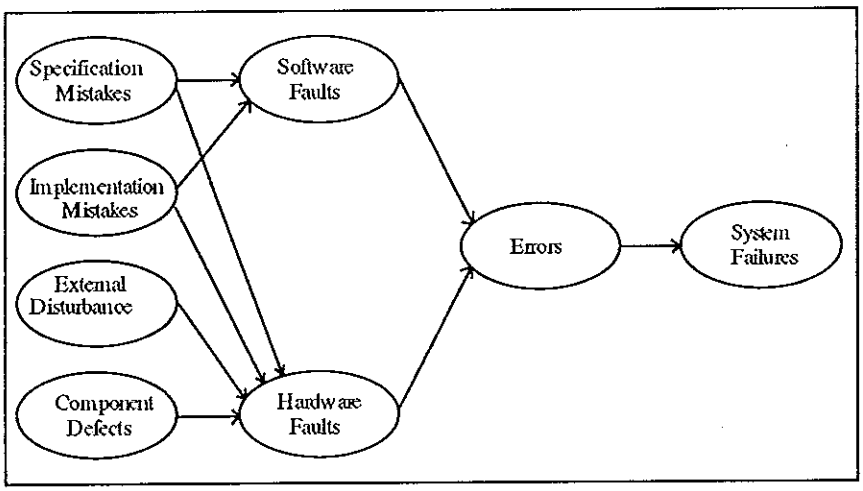
3. ความผิดปกติของชิ้นส่วน (component defects) เกิดจากความผิดพลาดในขั้นตอนการผลิต ทำให้ได้ชิ้นส่วนที่ทำงานไม่ถูกต้อง หรือเกิดจากความเสียหายของชิ้นส่วนด้วยสาเหตุอื่น

4. การถูกรบกวนจากภายนอก (external disturbance) เช่นการถูกรบกวนจากคลื่นวิทยุ จากสนามแม่เหล็ก หรือเกิดจากความผิดพลาดของผู้ใช้ เป็นต้น

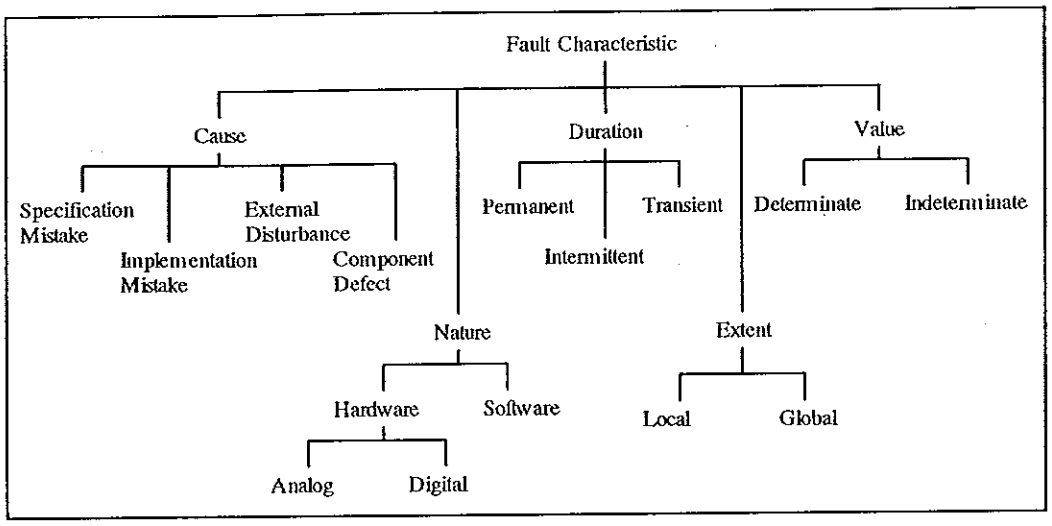
จากภาพประกอบ 3 สาเหตุต่างๆ ที่ทำให้การทำงานบกพร่องมีอยู่ด้วยกัน 4 สาเหตุดังที่ได้กล่าวมาแล้ว ซึ่งสามารถแบ่งออกเป็น 2 กลุ่มหลักๆ คือข้อผิดพร่องทางฮาร์ดแวร์ (hardware faults) และข้อผิดพร่องทางซอฟต์แวร์ (software faults) จากข้อผิดพร่องทั้งสองก็เป็นเหตุให้เกิดความผิดพลาดขึ้น เมื่อมีความผิดพลาดเกิดขึ้นก็มีผลทำให้ระบบทำงานล้มเหลว (system failures)

ข้อผิดพร่องที่เกิดขึ้นในระบบมีอยู่ด้วยกันหลายแบบ แต่ละแบบมีคุณสมบัติที่แตกต่างกันดังภาพประกอบ 4

ถ้าแบ่งคุณสมบัติของข้อผิดพร่องตามสาเหตุที่เกิดจะแบ่งออกได้เป็น 4 สาเหตุดังที่ได้กล่าวมาแล้ว แต่ถ้าแบ่งตามธรรมชาติของข้อผิดพร่องก็จะแบ่งได้เป็น 2 อย่างคือทางฮาร์ดแวร์ และทางซอฟต์แวร์ ถ้าแบ่งตามระยะเวลาที่เกิดข้อผิดพร่องจะแบ่งออกเป็น 3 ประเภทคือข้อผิดพร่องชนิดถาวร ข้อผิดพร่องชนิดชั่วคราว และข้อผิดพร่องที่เกิดขึ้นเป็นพักๆ ตัวอย่างของข้อผิดพร่องชนิดถาวร เช่น เกิดการลัดวงจรขึ้นภายในตัวไอซี ซึ่งจะเกิดขึ้นและคงค้างอยู่ตลอดเวลา ตัวอย่างของข้อผิดพร่องชนิดชั่วคราว เช่น สัญญาณรบกวนจากฟ้าผ่า ซึ่งจะทำให้วงจรทำงานผิดพลาดเพียงชั่วขณะ ส่วนตัวอย่างของข้อผิดพร่องที่เกิดขึ้นเป็นพักๆ เช่นการที่มีการเชื่อมต่อสายไฟระหว่างสองวงจรอย่างหลวมๆ ทำให้การทำงานของวงจรผิดๆ ถูกๆ เป็นพักๆ



ภาพประกอบ 3 สาเหตุและผลของข้อผิดพร่องในระบบ



ภาพประกอบ 4 การแบ่งคุณสมบัติของข้อผิดพร่อง

การจัดการกับข้อผิดพร่อง

มีเทคนิคพื้นฐานอยู่ 3 ประการที่ใช้สำหรับปรับปรุงการทำงานของระบบให้ทนทานต่อความผิดพร่องได้คือ

1. การหลีกเลี่ยงข้อผิดพร่อง (fault avoidance) เป็นการป้องกันข้อผิดพร่องที่อาจเกิดขึ้นได้ อาจโดยการทบทวนการออกแบบโดยละเอียด คัดเลือกอุปกรณ์เป็นพิเศษ การทดสอบอุปกรณ์ทุกชิ้น การควบคุมคุณภาพอย่างดี หรือแม้แต่การชิลด์ (shielding) วงจรอย่างดี เป็นต้น

2. การซ่อนข้อผิดพร่อง (fault hiding) เป็นการซ่อนข้อผิดพร่องที่เกิดขึ้นเพื่อที่จะได้ไม่ทำให้ระบบล้มเหลว เช่นการแก้ไขข้อมูลภายในหน่วยความจำก่อนที่จะมีการเรียกข้อมูลนั้นไปใช้งาน ซึ่งระบบจะไม่รู้ว่ามีข้อผิดพร่องเกิดขึ้น ระบบจึงสามารถทำงานได้ตามปกติ อีกวิธีหนึ่งของการซ่อนข้อผิดพร่องคือการใช้การลงคะแนนเสียงส่วนใหญ่ (majority voting) เช่นเมื่อมีวงจร 3 วงจรที่เหมือนกัน ถ้าทำงานชนิดเดียวกันแล้วมีวงจรหนึ่งให้ผลลัพธ์ที่แตกต่างจากผลลัพธ์จาก 2 วงจรที่เหลือแสดงว่า ผลลัพธ์ที่ได้จากวงจรทั้ง 2 ที่เหมือนกันนั้นเป็นผลที่ถูกต้อง

3. การทนต่อความผิดพร่อง (fault tolerant) เป็นความสามารถของระบบที่สามารถทำงานต่อไปได้อย่างถูกต้องถึงแม้ว่าจะมีข้อผิดพรองเกิดขึ้น เทคนิคการทำให้ระบบมีความทนต่อความผิดพร่องได้นั้นมีอยู่หลาย

เทคนิคด้วยกัน การซ่อนข้อผิดพร่องก็เป็นเทคนิคหนึ่ง อีกเทคนิคหนึ่งคือการหาตำแหน่งของข้อผิดพร่องในระบบ และทำการจัดระบบใหม่ (reconfigure) เพื่อกำจัดส่วนประกอบที่เสียหายออกไป ก่อนที่จะทำการจัดระบบใหม่ได้จะต้องมีขั้นตอนการทำงานดังนี้

3.1 การตรวจหาข้อผิดพร่อง (fault detection) เป็นกระบวนการที่จะตรวจรู้ได้ว่ามีข้อผิดพร่องเกิดขึ้นแล้ว

3.2 การหาตำแหน่งข้อผิดพร่อง (fault location) เป็นกระบวนการตรวจสอบว่าข้อผิดพร่องเกิดขึ้นที่ส่วนใดหรือจุดใดของระบบ

3.3 การจำกัดขอบเขตข้อผิดพร่อง (fault containment) เป็นกระบวนการแยกข้อผิดพร่องออกจากระบบ เพื่อที่จะไม่ทำให้เกิดข้อผิดพลาดขยายกว้างออกไป

3.4 การกู้ข้อผิดพร่อง (fault recovery) เป็นกระบวนการที่ทำให้ระบบสามารถทำงานต่อไปได้ในกรณีที่มีข้อผิดพร่องเกิดขึ้น

การออกแบบเพื่อให้ได้มาซึ่งความทนต่อความผิดพร่อง

การที่จะออกแบบระบบใดๆ ให้มีความสามารถในการทำงานต่อไปได้เมื่อมีข้อผิดพร่องเกิดขึ้นนั้นมีเทคนิคที่ได้มีการค้นคว้าศึกษามาแล้วหลายเทคนิคด้วยกัน แต่ละเทคนิคก็จะมีข้อดีข้อเสียที่แตกต่างกันออกไป บางเทคนิคซับซ้อน บางเทคนิคมีความเชื่อถือได้ต่ำ ฯลฯ เทคนิคการออกแบบที่เป็นพื้นฐานที่สุดและสำคัญที่สุดคือการใช้แนวความคิดของการซ้ำซ้อน (redundancy) (Johnson, B.W. 1989.)

การซ้ำซ้อนคือการเพิ่มข้อมูล เวลา หรือทรัพยากร เป็นพิเศษนอกเหนือจากที่ต้องการตามปกติให้กับระบบ เพื่อใช้ตรวจหาข้อผิดพลาด หรือทำงานแทนส่วนที่เสีย การซ้ำซ้อนมีได้หลายรูปแบบคือ

1. การซ้ำซ้อนทางฮาร์ดแวร์ (hardware redundancy) คือการใส่ฮาร์ดแวร์เพิ่มลงไปมากกว่าความต้องการตามปกติเพื่อวัตถุประสงค์ในการตรวจหาข้อผิดพร่อง และทำงานแทนส่วนที่ทำงานบกพร่อง

2. การซ้ำซ้อนทางซอฟต์แวร์ (software redundancy) คือการใส่ซอฟต์แวร์เพิ่มลงไปมากกว่าความต้องการตามปกติ เพื่อให้ทำงานตามฟังก์ชันที่ต้องการ และเพื่อวัตถุประสงค์ในการตรวจหาข้อผิดพร่อง และทำงานแทนส่วนที่ทำงานไม่ถูกต้อง

3. การซ้ำซ้อนทางข้อมูล (information redundancy) คือการเพิ่มข้อมูลหรือรายละเอียดลงไปมากกว่าปกติเพื่อช่วยในการตรวจหาความผิดพลาด และเป็นการสำรองข้อมูลเพื่อใช้ในการกู้ระบบ

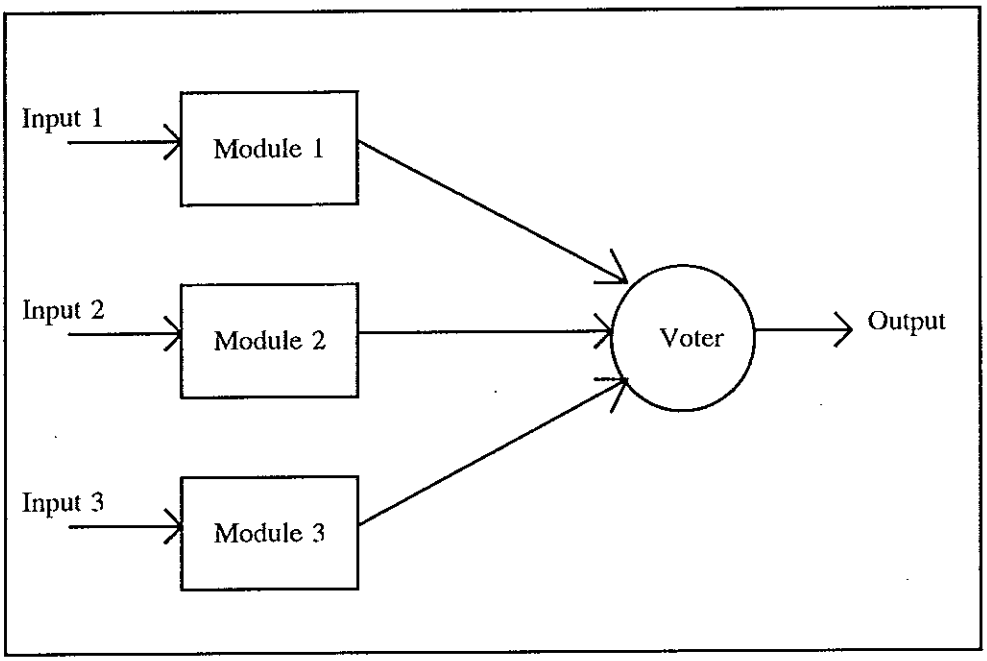
4. การซ้ำซ้อนทางเวลา (time redundancy) วิธีการนี้จะไม่ทำให้ฮาร์ดแวร์ของระบบมีขนาดใหญ่ แต่จะทำให้ระบบทำงานช้าลง หลักการทำงานของวิธีนี้คือการใช้เวลาในการทำฟังก์ชันใดๆ มากกว่าปกติ เช่นมีการทำฟังก์ชันมากกว่า 1 ครั้งทั้งนี้เนื่องจากข้อผิดพลาดที่เกิดขึ้นอาจจะเป็นข้อผิดพลาดแบบชั่วคราว เมื่อมีการประวิงเวลาเล็กน้อยแล้วทำการคำนวณซ้ำอีก ก็สามารถที่จะตรวจหา และหลีกเลี่ยงความผิดพลาดนั้นได้

การซ้ำซ้อนทางฮาร์ดแวร์

เป็นวิธีการที่นิยมกันมากในระบบดิจิทัลปัจจุบัน เนื่องจากว่าอุปกรณ์สารกึ่งตัวนำมีขนาดเล็กลงและราคาถูกลง การซ้ำซ้อนทางฮาร์ดแวร์มีอยู่ด้วยกัน 3 รูปแบบคือ

1 การซ้ำซ้อนทางฮาร์ดแวร์แบบพาสซีฟ (passive hardware redundancy) เป็นการใช้กลไกการลงคะแนนเสียงส่วนใหญ่เป็นหลัก ไม่มีการตรวจหาข้อ บกพร่อง และไม่มีการจัดระบบใหม่ โดยจะใช้วิธีการซ่อนข้อผิดพร่องไว้ และไม่มีการกระทำใดๆ ที่จะกำจัดส่วนที่บกพร่องออกไป ซึ่งมีหลายเทคนิคดังนี้

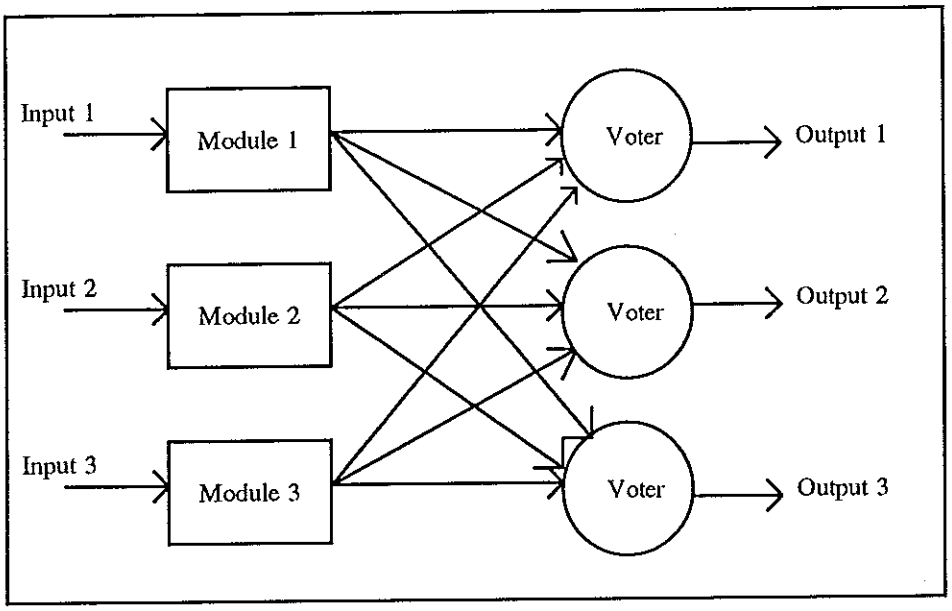
1.1 การซ้ำซ้อน 3 มอดูล (triple modular redundancy :TMR) เป็นรูปแบบที่ง่ายที่สุดของการซ้ำซ้อนทางฮาร์ดแวร์ ภาพประกอบ 5 แสดงหลักการทำงานของระบบดังกล่าว



ภาพประกอบ 5 หลักการทำงานของ TMR

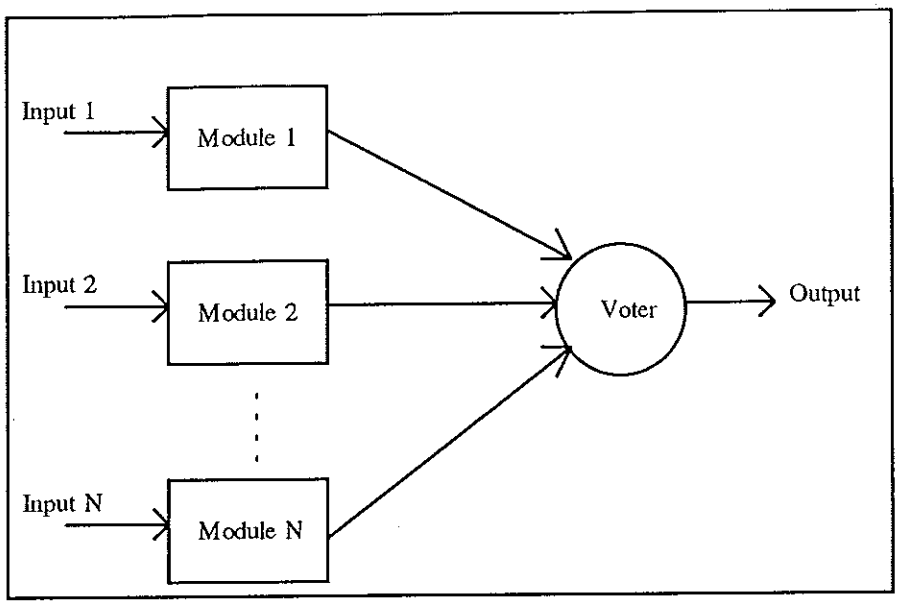
จากภาพประกอบ 5 เมื่อมีมอดูลใดมอดูลหนึ่ง ใน 3 มอดูล ทำงานผิดพลาด แต่อีก 2 มอดูลที่เหลือสามารถทำงานได้อย่างถูกต้อง จากการลงคะแนนเสียงส่วนใหญ่เอาต์พุตที่ได้จะยังคงถูกต้องอยู่ ในกรณีที่มีข้อผิดพร่องเกิดขึ้นมากกว่าหนึ่งมอดูล วิธีนี้จะไม่สามารถซ่อนข้อผิดพร่องไว้ได้

จากภาพประกอบ 5 ตัวลงคะแนนเสียง (voter) อาจทำงานผิดพลาดได้ เราสามารถซ่อนข้อผิดพร่องที่เกิดจากตัวลงคะแนนเสียงได้ โดยการใช้ตัวลงคะแนนเสียง 3 ตัว ซึ่งจะยอมให้ตัวลงคะแนนเสียงเสียได้เพียงตัวเดียว ดูการทำงานในภาพประกอบ 6



ภาพประกอบ 6 การซ้ำซ้อน 3 มอดูลที่มีตัวลงคะแนนเสียง 3 ตัว

1.2 การซ้ำซ้อนแบบ N มอดูล (N-modular redundancy:NMR) ในการซ้ำซ้อนแบบ 3 มอดูลนั้นจะยอมให้เกิดข้อผิดพร่องได้เพียงมอดูลเดียวเท่านั้น เมื่อต้องการให้ระบบสามารถยอมให้เกิดข้อผิดพร่องได้มากกว่า 1 มอดูลจะต้องเพิ่มจำนวนมอดูลเข้าไป เช่น ในการซ้ำซ้อนแบบ 5 มอดูล จะยอมให้มีข้อผิดพร่องเกิดขึ้นได้ไม่เกิน 2 มอดูลพร้อมกัน ซึ่งระบบยังคงสามารถทำงานได้อย่างถูกต้อง

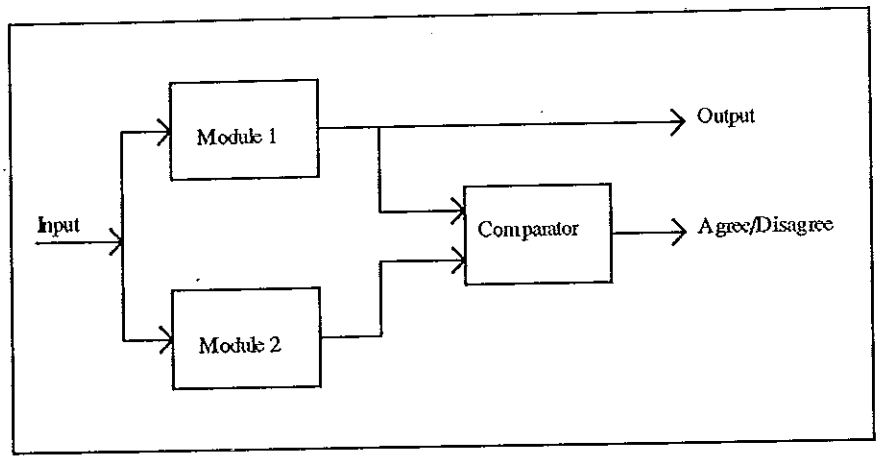


ภาพประกอบ 7 การซ้ำซ้อนแบบ N มอดูล

2. การซ้ำซ้อนทางฮาร์ดแวร์แบบแอ็กทีฟ (active hardware redundancy) เทคนิคนี้จะทำให้ระบบทนทานต่อความผิดพลาดได้โดยการตรวจหาข้อผิดพร่อง การหาตำแหน่งข้อผิดพร่อง และการกู้ระบบคืน โดยการกำจัดส่วนที่ทำงานผิดพลาดออกไปจากระบบ เทคนิคนี้มีข้อได้เปรียบกว่าการซ้ำซ้อนแบบพาสซีฟ คือมีส่วนที่ตรวจหาข้อผิดพร่อง ทำให้ทราบได้ว่าระบบมีข้อผิดพร่องหรือไม่ การออกแบบระบบโดยใช้เทคนิคการซ้ำซ้อนทางฮาร์ดแวร์แบบแอ็กทีฟมีอยู่หลายวิธีคือ

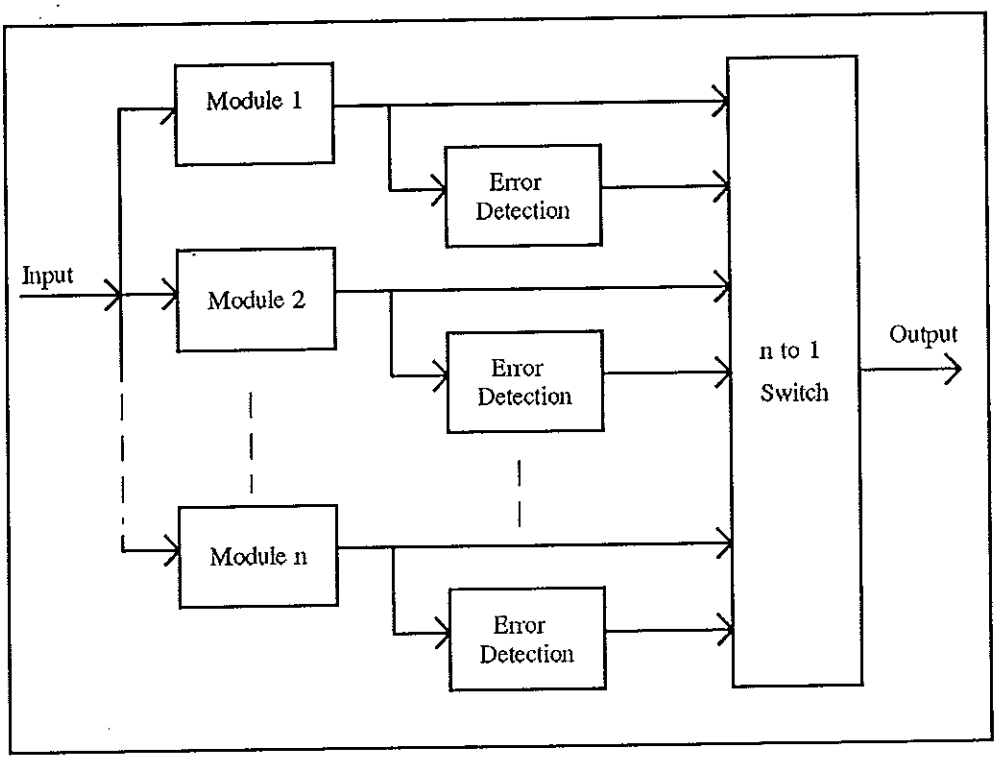
2.1. เทคนิค Duplication with Comparison เป็นการสร้างมอดูลสองมอดูลที่เหมือนกันทุกประการและให้ทำงานขนานกันไป แล้วมีการเปรียบเทียบเอาต์พุตที่ได้จากมอดูลทั้งสอง เมื่อเอาต์พุตที่ได้ต่างกันก็จะมีสัญญาณบอกว่ามีความผิดพลาดเกิดขึ้น อย่างไรก็ตามวิธีไม่สามารถบอกได้ว่ามอดูลที่เสียคือมอดูลใด วิธีนี้จึงนิยมใช้เป็นส่วนประกอบหนึ่งของการซ้ำซ้อนแบบแอ็กทีฟ

ปัญหาจากเทคนิคนี้มีอยู่ 2 ประการ ประการแรก ถ้ามอดูลทั้งสองได้รับอินพุตที่ผิด ก็จะไม่แสดงข้อผิดพลาดออกมา เพราะผลลัพธ์ที่ได้จะเป็นผลลัพธ์ที่ผิดเหมือนกัน ประการที่สอง วงจรเปรียบเทียบไม่สามารถทำงานได้อย่างถูกต้องในทุกกรณี ทั้งนี้ขึ้นอยู่กับงานที่นำไปประยุกต์ใช้ เช่น การอ่านค่าอินพุตที่เป็นสัญญาณอนาล็อกแล้วแปลงเป็นสัญญาณดิจิทัลของสองมอดูล อาจให้ผลที่ไม่เท่ากันอย่างพอดี ซึ่งในความเป็นจริงแล้วไม่ได้มีมอดูลใดที่บกพร่อง



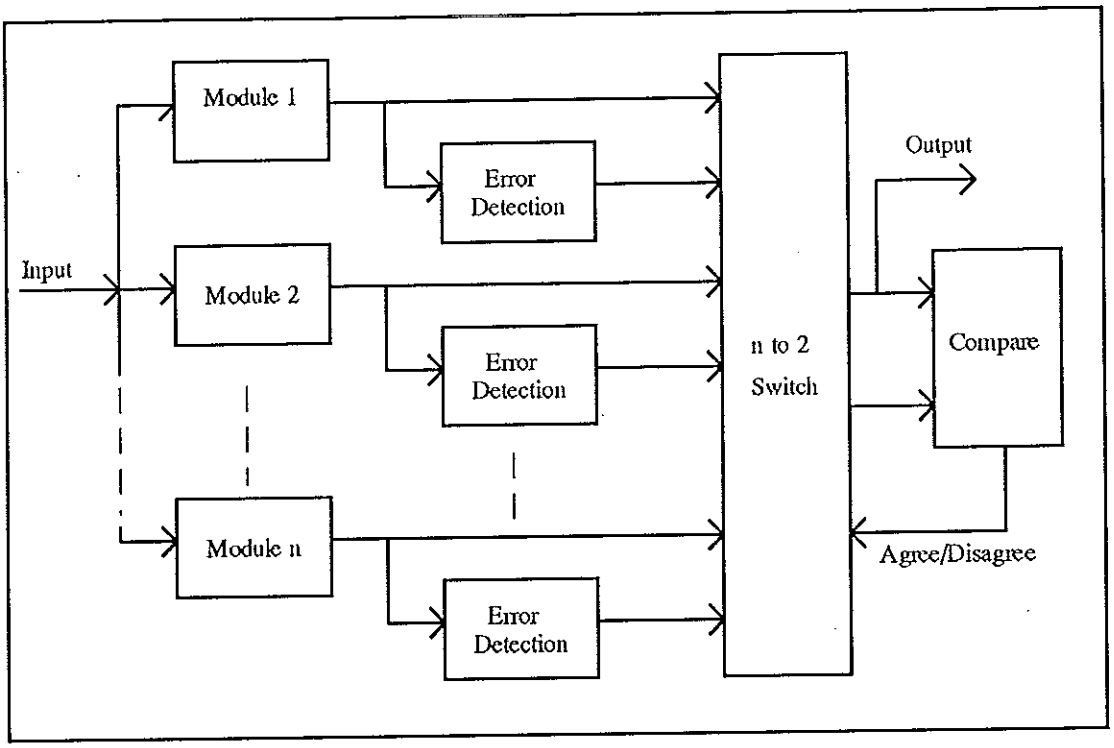
ภาพประกอบ 8 เทคนิค Duplication with Comparison

2.2. เทคนิค Standby Sparing แสดงการทำงานในภาพประกอบ 9 โดยหลักการจะมีมอดูลหนึ่งทำงานในขณะที่มอดูลที่เหลือจะทำหน้าที่เป็นมอดูลสำรอง ในทุกมอดูลจะมีส่วนสำหรับตรวจหาข้อผิดพร่องอยู่ เมื่อมอดูลที่ทำงานอยู่เกิดบกพร่องขึ้น มอดูลนั้นก็จะถูกตัดออกไป แล้วมอดูลที่สำรองอยู่จะทำงานแทน โดยมีวงจรสวิตช์เป็นตัวเลือกมอดูล



ภาพประกอบ 9 เทคนิค Standby Sparing

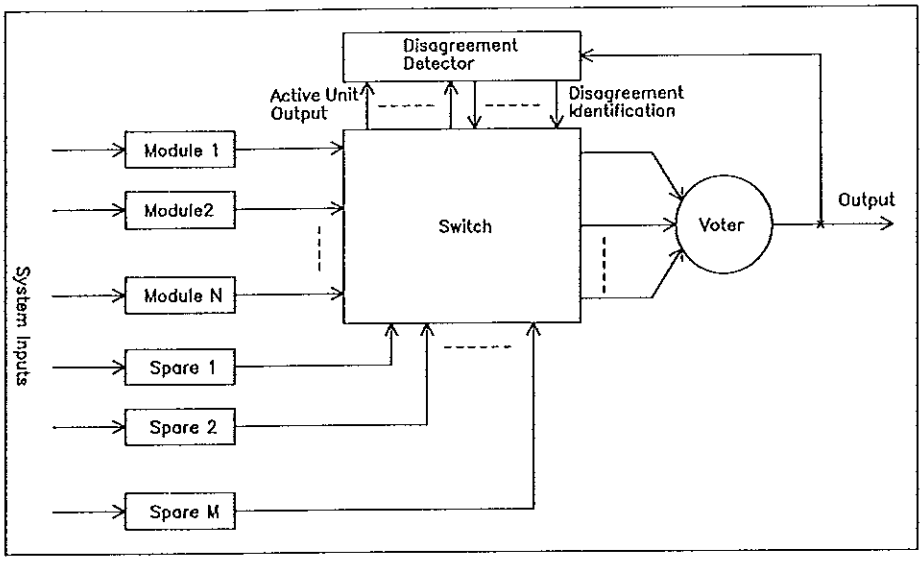
2.3. เทคนิค Pair and a Spare เป็นการผสมระหว่าง 2 วิธีแรก แสดงได้ดังภาพประกอบ 10 จากภาพวงจรสวิตช์จะเลือก 2 มอดูลจากที่มีทั้งหมด n มอดูลให้ทำงานและนำเอาต์พุตของมอดูล ทั้งสองมาเปรียบเทียบกัน ถ้าเอาต์พุตของมอดูลทั้งสองต่างกัน ก็แสดงว่ามีมอดูลที่บกพร่อง วงจรสวิตช์ก็จะทำหน้าที่เลือกมอดูลสำรองที่มีอยู่และให้เอาต์พุตที่ตรงกับมอดูลหลักมากทำงานแทนมอดูลที่เสีย



ภาพประกอบ 10 เทคนิค Pair and a Spare

3 การซ้ำซ้อนแบบผสม (hybrid hardware redundancy) เป็นการรวมเอาความสามารถในการซ่อนข้อผิดพร่องที่มีในการซ้ำซ้อนแบบพาสซีฟ กับความสามารถในการตรวจหาความผิดพลาดได้ของการซ้ำซ้อนแบบแอกทีฟ ระบบที่ได้จึงมีความเชื่อถือได้สูงมาก อย่างไรก็ตามเทคนิคนี้มีความซับซ้อนมาก และมีค่าใช้จ่ายในการพัฒนาสูง จึงเหมาะสำหรับสร้างระบบที่ต้องการความเชื่อถือได้สูงมากเท่านั้น

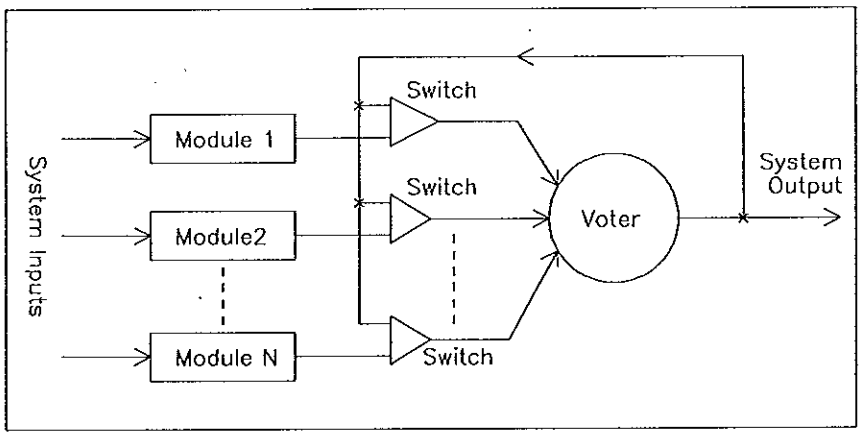
3.1 เทคนิค NMR with Spares เป็นวิธีที่นำเอาเทคนิค NMR มาเพิ่มความสามารถที่จะตรวจหาความผิดพลาดได้ โดยการนำเอาต์พุตที่ได้จากตัวลงคะแนนเสียงกลับมาเปรียบเทียบกับเอาต์พุตของมอดูล ต่างๆ เพื่อหาว่ามอดูลใดที่ให้ผลลัพธ์แตกต่างจากมอดูลอื่น และมีวงจรสวิตช์ที่ทำหน้าที่ตัดมอดูลที่เสียออกไปและต่อมอดูลสำรองเข้าทำงานแทน ภาพประกอบ 11 แสดงแผนภาพการทำงานของเทคนิคนี้



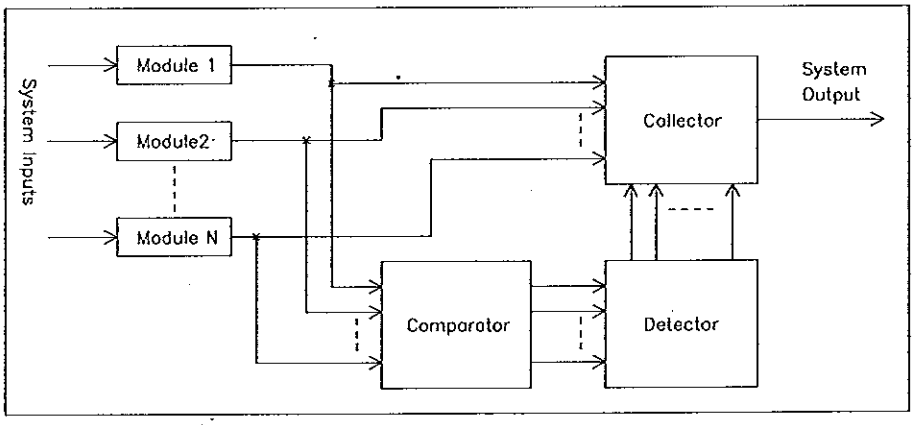
ภาพประกอบ 11 เทคนิค NMR with Spares

3.2 เทคนิค Self-Purging Redundancy เทคนิคนี้มีการทำงานคล้ายกับเทคนิค NMR with Spares แต่ต่างกันที่มอดูลทั้งหมดจะถูกต่อกับวงจรลงคะแนนเสียงตลอดเวลา ยกเว้นมอดูลที่บกพร่องจะถูกตัดออกจากอินพุตของวงจรลงคะแนนเสียง โดยใช้วงจรสวิตช์ที่เอาต์พุตของแต่ละมอดูล วงจรลงคะแนนเสียงที่ใช้จะต้องเป็น Threshold Gate (Johnson, B.W. 1989.) ภาพประกอบ 12 แสดงหลักการทำงานของเทคนิคนี้

3.3 เทคนิค Sift-Out Modular Redundancy มีการใช้มอดูลจำนวน n มอดูลเช่นเดียวกับวิธีอื่นๆ แต่มีการเพิ่มวงจรพิเศษเข้าไปคือวงจร Comparators วงจร Detectors และวงจร Collectors เข้าไปดังภาพประกอบ 13 วงจร Comparators จะทำหน้าที่เปรียบเทียบสัญญาณจากเอาต์พุตของทุกๆ มอดูลที่มีอยู่ ตัวอย่าง ถ้ามี 3 มอดูลจะต้องมีวงจรเปรียบเทียบ 3 วงจร หรือถ้ามี 5 มอดูลก็จะต้องมีวงจรเปรียบเทียบ 10 วงจร เป็นต้น สัญญาณจากวงจรเปรียบเทียบจะส่งไปให้วงจร Detector ตีความว่ามีมอดูลใดบ้างที่เสีย แล้วส่งผลลัพธ์ไปให้วงจร Collectors เพื่อเลือกเอาวงจรที่ทำงานถูกต้องมาทำงาน



ภาพประกอบ 12 เทคนิค Self-Purging Redundancy



ภาพประกอบ 13 เทคนิค Sift-Out Modular Redundancy

การซ้ำซ้อนทางข้อมูล (information redundancy)

เป็นการเพิ่มข้อมูลข่าวสารหรือรายละเอียดมากกว่าที่ต้องการตามปกติแก่ฮาร์ดแวร์หรือซอฟต์แวร์ เพื่อประโยชน์ในการตรวจหาข้อผิดพร่อง การสำรองข้อมูล และการกู้คืนระบบ ตัวอย่างเทคนิคการซ้ำซ้อนทางข้อมูลที่นิยมใช้กันทั่วไปมีดังนี้

1. Parity Codes เป็นการเพิ่มบิตของข้อมูลขึ้นเพื่อใช้เก็บคุณสมบัติของข้อมูล เช่นมี Parity Bit เพื่อบ่งบอกว่าข้อมูลที่เก็บอยู่มีจำนวนบิตที่มีลอจิก 1 อยู่เป็นจำนวนคู่หรือจำนวนคี่ เมื่อมีการเข้าถึงข้อมูลก็จะมีการตรวจสอบว่าค่า Parity Bit มีความสัมพันธ์กับข้อมูลถูกต้องหรือไม่

2. Duplication Codes เป็นการเก็บข้อมูลเดียวกันที่มากกว่า 1 รูปแบบ เช่นเก็บข้อมูลให้มีการเรียงบิตที่แตกต่างกัน เป็นต้น วิธีนี้มีข้อเสียคือต้องให้จำนวนบิตมากเป็น 2 เท่าหรือมากกว่า

3. Checksums เป็นการตรวจสอบความผิดพลาดของกลุ่มข้อมูลอย่างง่ายโดยการตรวจสอบค่าผลรวมของข้อมูลทั้งหมดซึ่งทั้งฝ่ายรับและฝ่ายส่งต้องมีค่าเท่ากัน วิธีนี้มีข้อเสียคือไม่สามารถตรวจสอบความผิดพลาดได้ทุกกรณี

4. Cyclic Codes เป็นวิธีการนำข้อมูลมาสร้างรหัสบางอย่างที่เป็นเฉพาะของกลุ่มข้อมูลนั้นๆ

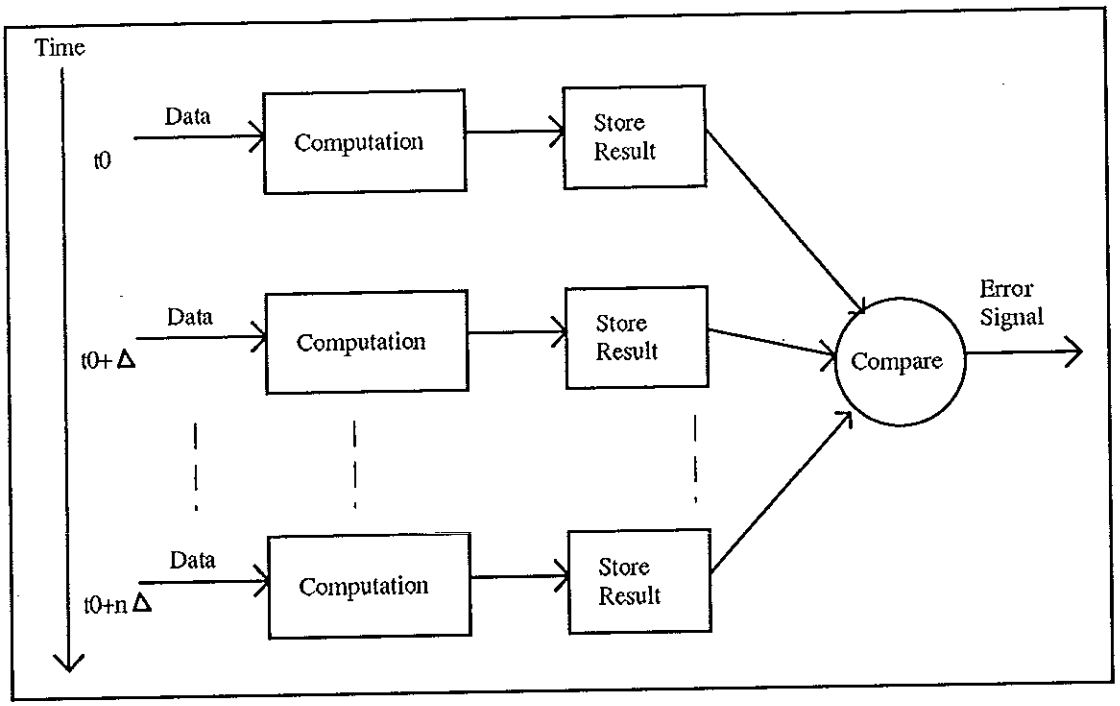
นอกจากนี้ยังมีเทคนิคต่างๆ อีกเช่น Arithmetic Codes, Berger Codes, Horizontal and Vertical Parity, Hamming Error-Correcting Codes เป็นต้น

การซ้ำซ้อนทางเวลา (time redundancy)

เป็นการซ้ำซ้อนที่ไม่ต้องเสียค่าใช้จ่ายในการสร้างฮาร์ดแวร์เพิ่ม แต่ระบบจะทำงานได้ช้าลงเพราะมีการใช้เวลาเพิ่มขึ้นในการกระทำฟังก์ชันใดๆ ซ้ำ ประโยชน์ของการซ้ำซ้อนทางเวลาคือ

1. การตรวจหาข้อผิดพร่องชั่วครู่ (transient fault detection) เมื่อมีข้อผิดพร่องเกิดขึ้นกับระบบ อาจเป็นไปได้ว่าข้อผิดพร่องนั้นเป็นข้อผิดพร่องชั่วครู่ ซึ่งอาจเกิดจากถูกสัญญาณรบกวนจากภายนอก ข้อผิดพร่องจะไม่ปรากฏอีกเมื่อเวลาผ่านไป ดังนั้นเมื่อตรวจได้ว่ามีข้อผิดพร่องเกิดขึ้นแล้วสันนิษฐานว่าเป็นข้อผิดพร่องชั่วครู่ สามารถพิสูจน์ได้โดยการคำนวณค่าอีกครั้งหรือประมวลผลซ้ำอีกโดยเว้นช่วงเวลาสักระยะ ซึ่งเมื่อมี

การคำนวณซ้ำหลายๆ ครั้งแล้วนำผลที่ได้มาทำการลงคะแนนเสียงส่วนใหญ่ เราจะได้ผลลัพธ์ที่ถูกต้อง ถ้าข้อผิดพร่องที่เกิดขึ้นเป็นข้อผิดพร่องชั่วครู่ ภาพประกอบ 14 แสดงหลักการของการซ้ำซ้อนทางเวลาเพื่อตรวจหาข้อผิดพร่องชั่วครู่



ภาพประกอบ 14 การตรวจหาข้อผิดพร่องชั่วครู่โดยการซ้ำซ้อนทางเวลา

2. การตรวจหาข้อผิดพร่องถาวร (permanent fault detection) โดยใช้การซ้ำซ้อนทางเวลาเราสามารถที่จะตรวจหาได้ข้อผิดพร่องแบบถาวรได้ โดยไม่เพิ่มความซับซ้อนกับระบบมากนัก วิธีการคือในการคำนวณครั้งแรก จะทำการคำนวณตามปกติ แต่เมื่อเวลาผ่านไปช่วงหนึ่ง จะมีการคำนวณซ้ำอีกครั้ง โดยในครั้งหลังนี้ข้อมูลอินพุตจะถูกเข้ารหัสไว้ ซึ่งเป็นรหัสที่จะช่วยในการตรวจหาข้อผิดพร่องได้ เมื่อคำนวณได้ผลลัพธ์มาแล้วจะทำการถอดรหัสออก แล้วนำผลที่ได้จากการคำนวณทั้งสองครั้งมาเปรียบเทียบกันว่าถูกต้องหรือไม่

การซ้ำซ้อนทางซอฟต์แวร์ (software redundancy)

เป็นการเพิ่มซอฟต์แวร์เป็นพิเศษสำหรับตรวจหาข้อผิดพร่อง และแก้ไข สำหรับตัวซอฟต์แวร์เองและวงจรฮาร์ดแวร์

ฝ่ายหอสมุด
คุณหญิงหลง อรรถกระวีสุนทร

บทที่ 3

การออกแบบฮาร์ดแวร์ของวงจรสวิตช์

พื้นฐานของวงจรสวิตช์แบบดิจิทัล

ในการสวิตช์ข้อมูลแบบดิจิทัลผ่านวงจรสวิตช์นั้น โดยทั่วไปแล้วถ้าข้อมูลที่ต้องการถูกสวิตช์มีจำนวนหลายช่อง ข้อมูลแต่ละช่องจะต้องถูกมัลติเพล็กซ์ (multiplex) ให้อยู่ในรูป Time Division Multiplex (TDM) ก่อนเพื่อลดจำนวนสายส่งสัญญาณ ซึ่งข้อมูลที่ถูกมัลติเพล็กซ์แล้วนั้นอาจอยู่ในรูปแบบอนุกรม (serial) หรือขนาน (parallel) ก็ได้ การทำ TDM นั้นข้อมูลแต่ละช่องจะมีช่องเวลา (timeslot) เป็นของตัวเอง ดังนั้นวงจรสวิตช์ก็คือวงจรที่ทำหน้าที่นำสัญญาณจากช่องเวลาหนึ่งไปใส่ไว้ในอีกช่องเวลาหนึ่ง การแลกเปลี่ยนข้อมูลจึงเกิดขึ้นได้ หรืออาจกล่าวได้ว่าวงจรสวิตช์ก็คือวงจรสับเปลี่ยนช่องเวลา (timeslot interchange circuit) นั่นเอง

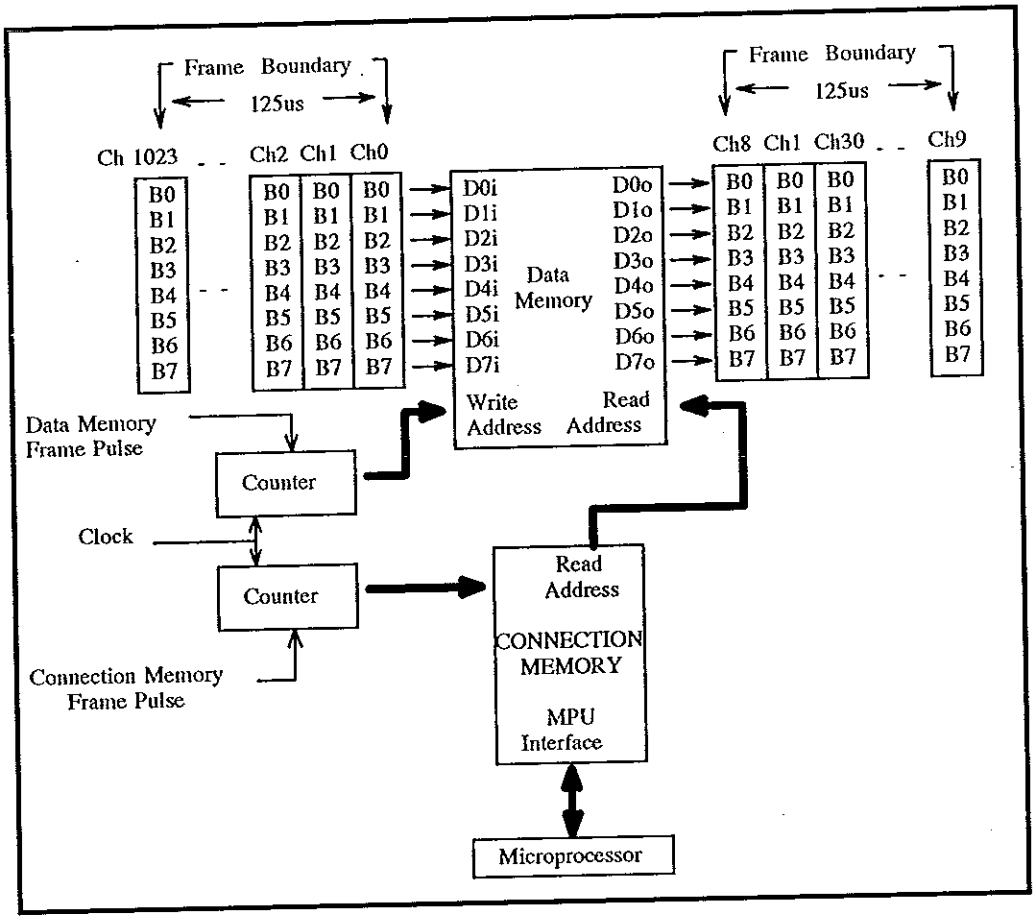
ในระบบชุมสายโทรศัพท์ที่ใช้วงจรสวิตช์แบบดิจิทัลนั้น สัญญาณเสียงซึ่งอยู่ในรูปแอนะล็อกจะถูกแปลงให้อยู่ในรูปแบบดิจิทัลโดยวิธีการทำ Pulse Code Modulation (PCM) โดยมีอัตราเร็วของข้อมูล (data rate) ของสัญญาณแต่ละช่องเท่ากับ 8 กิโลไบต์ต่อวินาทีในแบบขนาน หรือ 64 กิโลบิตต่อวินาทีในแบบอนุกรม

โดยทั่วไปจะมีการรวมช่องสัญญาณหลายๆ ช่องเป็นกลุ่มซึ่งเรียกว่าเฟรม (frame) โดยในแต่ละเฟรมอาจมีจำนวนช่องสัญญาณเท่ากับ 32 หรือ 64 หรืออื่นๆ ทั้งนี้ขึ้นอยู่กับผู้ออกแบบสถาปัตยกรรมของระบบ อย่างไรก็ตามอัตราเร็วเฟรม (frame rate) จะต้องมีค่าคงที่ตามมาตรฐานคือ 8000 เฟรมต่อวินาที ดังนั้นถ้ามีจำนวนช่องต่อหนึ่งเฟรมมากอัตราเร็วข้อมูลก็จะสูงขึ้น และความกว้างของช่องเวลาก็จะแคบลง แผนภาพการทำงานของวงจรสับเปลี่ยนช่องเวลาแสดงไว้ในภาพประกอบ 15

จากภาพประกอบ 15 Data Memory คือหน่วยความจำที่เก็บข้อมูลทั้งหมดที่เข้าทางอินพุต โดยอินพุตแต่ละช่องจะมีตำแหน่งหน่วยความจำเป็นของตัวเอง เช่นช่องที่ 1 จะถูกเก็บลงในตำแหน่งหน่วยความจำที่ 1 ช่องที่ 2 ถูกเก็บลงในหน่วยความจำตำแหน่งที่ 2 เป็นต้น โดยจะมีวงจรนับ (Counter) สำหรับนับตำแหน่งหน่วยความจำเขียน (Write Address) ที่สอดคล้องกับสัญญาณอินพุตที่เข้ามา ข้อมูลใน Data Memory จะถูกอ่านออกมาจากตำแหน่งหน่วยความจำอ่าน (Read Address) ที่ถูกกำหนดโดย Connection Memory

Connection Memory เป็นหน่วยความจำที่เก็บข้อมูลที่จะบอกว่าจะนำข้อมูลใน Data Memory จากตำแหน่งใดไปใส่ในช่องเวลาใดของเอาต์พุต โดยสัญญาณที่ส่งออกจาก Connection Memory ไปยัง Data Memory นั้นจะเป็นตำแหน่งของหน่วยความจำใน Data Memory ที่ต้องการส่ง

ออกในขณะนั้นๆ ซึ่งตำแหน่งหน่วยความจำที่ส่งไปนั้น จะมีจังหวะที่สอดคล้องกับตำแหน่งช่องเวลาที่เอาต์พุตโดยมีวงจรนับเป็นตัวกำหนดและมีการทำให้เข้าจังหวะ (synchronization) กันด้วยสัญญาณ Data Memory Frame Pulse และ Connection Memory Frame Pulse



ภาพประกอบ 15 หลักการทำงานของวงจรสับเปลี่ยนช่องเวลา

ส่วน MPU Interface ของ Connection Memory จะทำหน้าที่รับส่งข้อมูลกับไมโครโพรเซสเซอร์ โดยไมโครโพรเซสเซอร์จะเป็นผู้เขียนตำแหน่งที่ต้องการอ่านข้อมูลจาก Data Memory ลงในหน่วยความจำของ Connection Memory ในเลขที่อยู่ (address) ที่เหมาะสม เพราะเลขที่อยู่แต่ละเลขที่อยู่ใน Connection Memory จะสอดคล้องกับช่องเวลาที่เอาต์พุต

จากการทำงานของวงจรสับเปลี่ยนช่องเวลาจะเห็นว่าในช่วงเวลา 1 ช่องเวลาจะมีการเข้าถึง (access) Data Memory 2 ครั้งเริ่มจากจังหวะการเขียนลงในตำแหน่งหน่วยความจำที่ถูกกำหนดโดย Counter และจังหวะการอ่านที่ถูกกำหนดตำแหน่งหน่วยความจำโดย Connection Memory ดังนั้นจำนวนช่องของสัญญาณอินพุตที่วงจรสามารถทำการสับเปลี่ยนได้จะขึ้นอยู่กับความเร็วในการเข้าถึงหน่วยความจำ

(memory access time) ของ Data Memory ถ้า Data Memory มีระยะเวลาในการเข้าถึงสั้นก็จะสามารถสวิตช์สัญญาณได้หลายช่องในเวลาเดียวกัน ตัวอย่างเช่น Data Memory ที่มีระยะเวลาในการเข้าถึง 60 นาโนวินาที จะสามารถสวิตช์สัญญาณได้สูงสุด 1,024 ช่องพร้อมกัน

จากภาพประกอบ 15 สัญญาณอินพุตและเอาต์พุตเป็นแบบขนานขนาด 8 บิต (B0 ถึง B7) แต่โดยทั่วไปแล้วสัญญาณอินพุตแบบ PCM ของสัญญาณเสียงจะถูกส่งมาในภาพแบบอนุกรม เช่นการส่งสัญญาณโดยใช้มาตรฐาน ST-BUS (Mitel Corporation. 1991.) เป็นต้น โดยใน ST-BUS 1 กระแส (Stream) จะมีช่องสัญญาณ 32 ช่อง แต่ละช่องจะมีขนาด 8 บิตอนุกรม จะเห็นว่าการส่งสัญญาณแบบอนุกรมมีข้อดีคือสามารถส่งสัญญาณได้หลายช่องโดยใช้สายส่งสัญญาณเส้นเดียว อย่างไรก็ตามการส่งข้อมูลแบบอนุกรมจะต้องใช้อัตราเร็วที่สูงกว่าเมื่อเทียบกับการส่งข้อมูลแบบขนาน ในระบบชุมสายโทรศัพท์ซึ่งมีความสามารถในการสวิตช์สัญญาณได้หลายๆ พันช่องพร้อมกัน ถ้าไม่ใช้การมัลติเพล็กซ์สัญญาณ และการแปลงสัญญาณจากขนานเป็นอนุกรมเข้าช่วยแล้ว จำนวนสายส่งสัญญาณจะมีมากมายมหาศาล ซึ่งจะก่อให้เกิดปัญหาในการสร้างได้ แต่เนื่องจากมีข้อจำกัดในการผลิตหน่วยความจำที่ใช้สร้าง Data Memory ซึ่งไม่สามารถสร้างหน่วยความจำให้มีความเร็วในการเข้าถึงสูงมากๆ ได้ ทางออกก็คือให้มีการเขียนอ่านข้อมูลได้หลายๆ บิตพร้อมกัน แทนที่จะให้มีการเขียนอ่านครั้งละ 1 บิตซึ่งเสียเวลามาก ดังนั้นในวงจรสวิตช์ส่วนมากจะต้องมีวงจรที่ใช้ในการแปลงสัญญาณจากอนุกรมเป็นแบบขนานก่อนที่จะส่งมาให้วงจรสับเปลี่ยนช่องเวลาและจะต้องมีวงจรแปลงสัญญาณจากแบบขนานเป็นแบบอนุกรมเพื่อแปลงสัญญาณให้เหมาะสมก่อนที่จะส่งออกไปยังวงจรอื่นๆ

การออกแบบวงจรสับเปลี่ยนช่องเวลาโดยใช้ไอซี MT9080

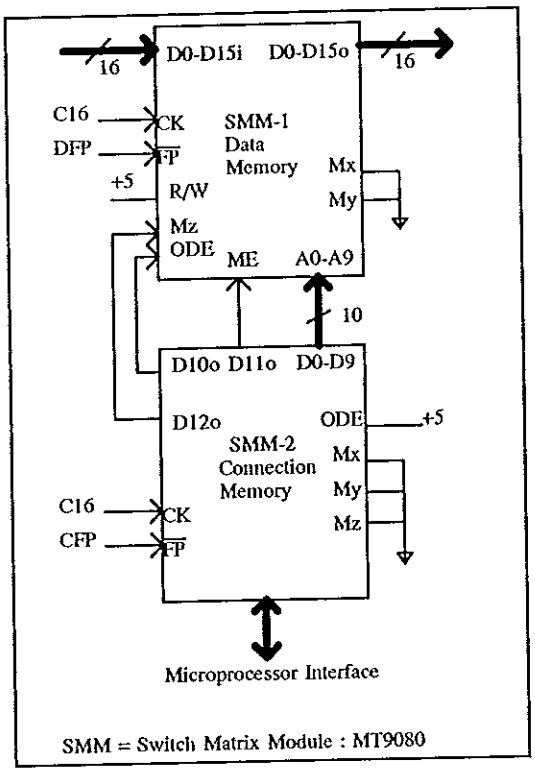
1. การทำงานของวงจร

จากคู่มือวงจรรวมของบริษัทไมเทลคอร์โปเรชั่น (Mitel Corporation, 1991.) สามารถนำวงจรรวมเบอร์ MT9080 2 ตัวมาสร้างเป็นวงจรสับเปลี่ยนช่องเวลาได้ โดยตัวแรกจะถูกควบคุมให้ทำงานเป็น Data Memory และอีกตัวหนึ่งทำหน้าที่เป็น Connection Memory แผนภาพของวงจรแสดงดังภาพประกอบ 16

จากแผนภาพข้อมูลที่จะถูกสวิตช์จะถูกส่งมาที่ SMM-1 ซึ่งทำหน้าที่เป็น Data memory โดยผ่านทางบัสข้อมูล (data bus) ขนาด 16 บิต และข้อมูลนี้จะถูกเก็บลงในตำแหน่งหน่วยความจำที่เรียงกันไปซึ่งการกำหนดตำแหน่งหน่วยความจำนี้จะมีวงจรนับขนาด 11 บิตภายในทำงานอยู่

ข้อมูลจะถูกอ่านออกจาก Data Memory จากตำแหน่งหน่วยความจำที่ถูกกำหนดให้โดย Connection Memory SMM-2 ซึ่งส่งออกมาเป็นอนุกรมโดยวงจรนับภายในขนาด 11 บิต โดยตำแหน่งหน่วยความจำนี้ได้มีการกำหนดไว้แล้วจากไมโครโพรเซสเซอร์

เนื่องจากเป็นวงจรขนาด 1024 ช่อง ดังนั้นในการอ้างเลขที่อยู่หน่วยความจำ 1024 ตำแหน่งจึงใช้สัญญาณเลขที่อยู่เพียง 10 เส้น โดยจะใช้ D0 ถึง D9 จาก Connection Memory ต่อเข้ากับ A0 ถึง A9 ของ Data Memory สัญญาณข้อมูลที่เหลือจาก Connection Memory สามารถนำมาใช้ควบคุมการทำงานอื่นๆ ได้ เช่นในตัวอย่างวงจรที่ได้แสดงไว้จะนำเอา D10 มาควบคุมสัญญาณเปิดข้อความ (message enable : ME) ส่วน D11 ถึง D15 นั้นไม่ได้นำมาใช้งาน



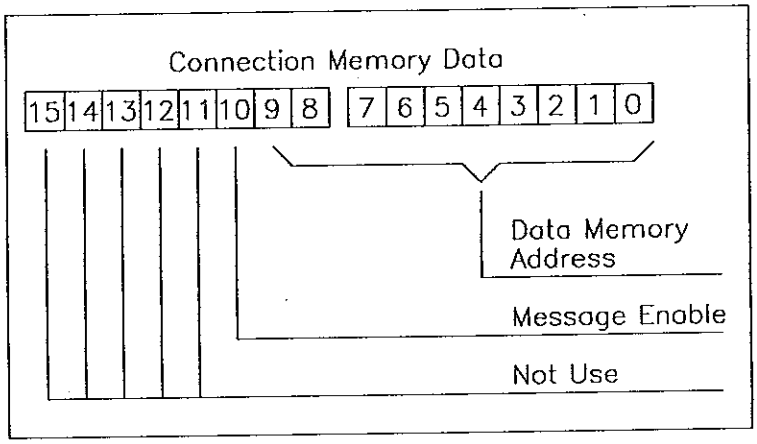
ภาพประกอบ 16 การสร้างวงจรสับเปลี่ยนช่องเวลาขนาด 1,024 ช่องจาก MT9080

ในการต่อลักษณะนี้ ทำให้การอ้างเลขที่อยู่ (addressing) และการควบคุมการทำงานสามารถจัดเป็นบิตต่างๆ ของ Connection Memory เพื่อสร้างเป็นคำควบคุม (control word) ขนาด 16 บิตได้ดังภาพประกอบ 17

จากภาพประกอบ 16 สัญญาณนาฬิกา C16 ที่ป้อนให้กับ SMM-1 และ SMM-2 นั้นมีความถี่เท่ากันคือ 16.384 เมกกะเฮิร์ตซ์ อัตราเร็วข้อมูลแบบขนานทางด้านอินพุตและเอาต์พุตสำหรับวงจรนี้มีค่าเท่ากับ 8.192 เมกกะบิตต่อวินาที ความกว้างของช่องเวลามีค่าเท่ากับสองเท่าของคาบเวลาของสัญญาณนาฬิกา C16

2. วงจรสับเปลี่ยนช่องเวลาที่มีสัญญาณอินพุตและเอาต์พุตเป็นแบบอนุกรม

ดังที่ได้กล่าวมาแล้วว่าสัญญาณอินพุตของวงจรสวิตช์นั้นจะอยู่ในรูปอนุกรมเพื่อลดจำนวนสายส่งสัญญาณ ดังนั้นวงจรสับเปลี่ยนช่องเวลาจึงต้องมีความสามารถที่จะรับข้อมูลอินพุตแบบอนุกรมได้ด้วย ในการออกแบบวงจรเพื่อให้สามารถรับสัญญาณอินพุตแบบอนุกรมได้นี้สามารทำได้โดยการเพิ่มวงจรเรจิสเตอร์แบบเลื่อน (shift register) วงจรหนึ่งเพื่อแปลงสัญญาณจากอนุกรมเป็นขนาน และอีกวงจรหนึ่งสำหรับแปลงสัญญาณจากขนานเป็นอนุกรม ไอซีที่ทำหน้าที่ทั้งสองนี้ได้คือเบอร์ MT9085 (Mitel Corporation. 1991.) ซึ่งสามารถทำงานทั้งสองอย่างได้โดยการควบคุมที่เหมาะสม สัญญาณอนุกรมที่วงจรรวมสามารถทำงานได้สูงสุดคือ 32 กระแส โดยแต่ละกระแสมีจำนวนสัญญาณเท่ากับ 32 ช่อง และสัญญาณขนานที่รับหรือแปลงได้จะมีขนาด 8 บิต



ภาพประกอบ 17 รูปแบบของคำควบคุม

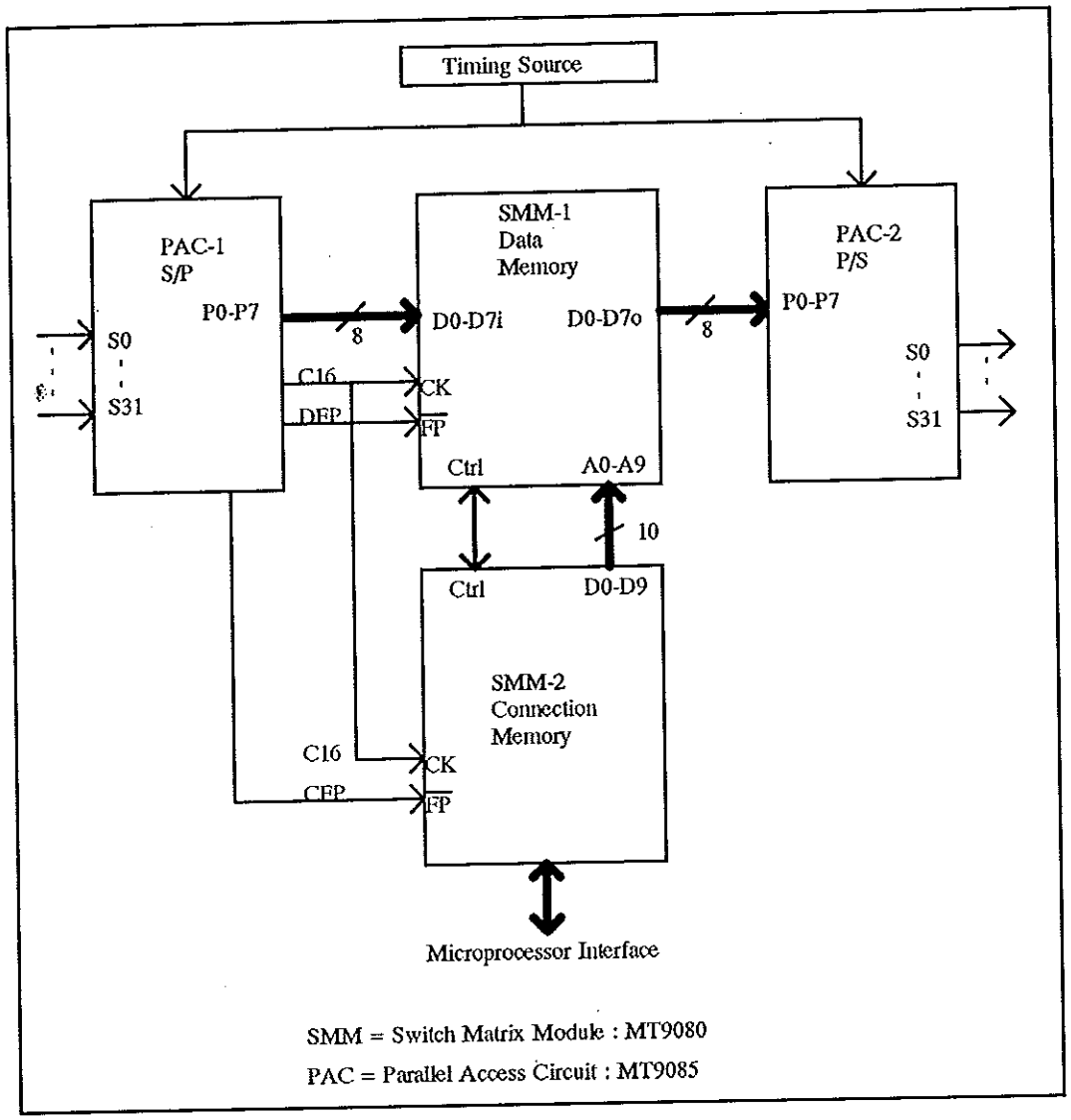
วงจรสวิตช์ที่มีสัญญาณอินพุตและเอาต์พุตแบบอนุกรมแสดงไว้ในภาพประกอบ 18 ซึ่งจากภาพจะเห็นว่าวงจรทั้งหมดจะใช้สัญญาณนาฬิกาจากแหล่งเดียวกันและการเข้าจังหวะ (synchronize) ถูกควบคุมโดยสัญญาณ DFP (Data-Memory Frame Pulse) ซึ่งเป็นสัญญาณที่ใช้ในการตั้งใหม่ (reset) วงจรนับภายใน Data Memory และสัญญาณ CFP (Connection-Memory Frame Pulse) ซึ่งเป็นสัญญาณที่ใช้ในการตั้งใหม่วงจรนับภายใน Connection Memory โดยสัญญาณทั้งสองนี้ถูกส่งมาจาก PAC-1

3. การโปรแกรมการทำงานของวงจรสวิตช์

การควบคุมการทำงานของวงจรสวิตช์จะกระทำโดยใช้ซอฟต์แวร์บนไมโครโพรเซสเซอร์ ซึ่งการสวิตช์ข้อมูลอินพุตกับเอาต์พุตนั้นสามารถทำได้โดยการเขียนตำแหน่งเลขที่อยู่ของหน่วยความจำของข้อมูลอินพุตลงไปใน Connection Memory ในตำแหน่งที่ตรงกันกับตำแหน่งช่องเวลาของเอาต์พุต

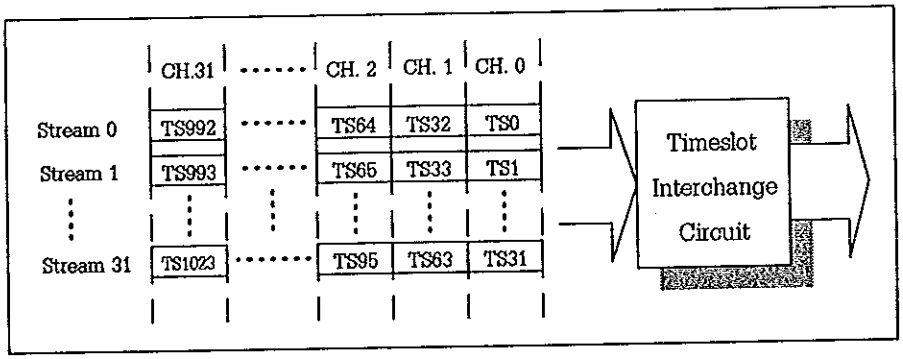
ตัวอย่างเช่นต้องการสวิตช์สัญญาณอินพุตจากช่องที่ 7 ไปยังเอาต์พุตช่องที่ 9 ก็สามารถทำได้โดยเขียนค่า 7 ลงในเลขที่อยู่ที่ 9 ของ Connection Memory

ดังที่ได้ยกตัวอย่างวงจรมาแล้ว จะเห็นว่าเราสามารถที่จะควบคุมการทำงานของ Data Memory ได้ในเวลาเดียวกันกับการเขียนข้อมูลการสวิตช์ลงใน Connection Memory ดังนั้นจากตัวอย่างข้างต้นค่าที่เขียนลงใน Connection Memory อาจไม่ใช่ค่า 9 ซึ่งในรายละเอียดจะต้องพิจารณาจากภาพประกอบ 17



ภาพประกอบ 18 วงจรสับเปลี่ยนช่องเวลาแบบอนุกรม

เนื่องจากเลขที่อยู่ที่ข้อมูลในช่องเวลาต่างๆ จะถูกนำไปเก็บลงใน Data Memory นั้นมีค่าเดียวกันกับอันดับของช่องเวลาตัวอย่างเช่น ข้อมูลในช่องเวลาที่ 100 จะเก็บลงในเลขที่อยู่ที่ 100 ของ Data Memory หรือข้อมูลในช่องเวลาที่ 999 จะเก็บลงในเลขที่อยู่ที่ 999 เป็นต้น ในขณะที่สัญญาณอินพุตที่เข้ามานั้นมีอยู่ทั้งสิ้น 32 กระแส โดยแต่ละกระแสมีจำนวนช่องสัญญาณเท่ากับ 32 ช่อง การคำนวณหาความสัมพันธ์ระหว่างเลขที่อยู่ของ Data Memory กับช่องสัญญาณใดๆ ในกระแสต่างๆ สามารถทำได้โดยการพิจารณารูปแบบการวางตัวของช่องเวลาในกระแสต่างๆ ดังภาพประกอบ 19



ภาพประกอบ 19 รูปแบบการวางตัวของช่องเวลาในกระแสต่างๆ

จากภาพประกอบ 19 สามารถหาตำแหน่งเลขที่อยู่ของ Data Memory ได้จากสมการต่อไปนี้

Address = (Channel X 32) + Stream ..... (1)

โดย Address คือเลขที่อยู่ของ Data Memory ที่ข้อมูลจะถูกนำไปเก็บ Channel คือหมายเลขช่องของสัญญาณ และ Stream คือหมายเลขกระแสของสัญญาณ

ตัวอย่าง ต้องการหาตำแหน่งเลขที่อยู่ที่เก็บข้อมูลจากช่องที่ 31 ของกระแสที่ 1

แทนค่าลงในสมการที่ 1 จะได้

Address = (1 X 32) + 1

= 993

การที่ทราบว่าสัญญาณอินพุตที่เข้ามาทางช่องสัญญาณใดๆ ของกระแสต่างๆ มีประโยชน์สำหรับการโปรแกรมค่าใน Connection Memory เมื่อทราบค่าเลขที่อยู่ของ Data Memory ที่เก็บข้อมูลอินพุตที่

ต้องการแล้วการโปแกรมค่าใน Connection Memory ก็เพียงแต่ทำการเขียนค่าเลขที่อยู่ดังกล่าว ลงไปในตำแหน่งเลขที่อยู่ของ Connection Memory ที่มีค่าตรงกับหมายเลขช่องเวลาที่ต้องการให้สัญญาณอินพุตนั้นๆ ถูกส่งออกไป โดยหมายเลขช่องเวลาทางด้านเอาต์พุตที่ต้องการนั้นสามารถใช้วิธีการคำนวณจากสมการที่ 1 ได้

ตัวอย่าง ต้องการควบคุมวงจรสวิตช์ให้ทำการสวิตช์สัญญาณจากอินพุตช่องที่ 5 ของกระแสที่ 7 ไปยังเอาต์พุตช่องที่ 9 ของ กระแส ที่ 19

วิธีทำ

หาหมายเลขช่องเวลาหรือ เลขที่อยู่ของ Data Memory ของสัญญาณอินพุตได้จากสมการที่ 1 ซึ่งจะได้

DM_Address = (5 X 32) + 7 = 167

และหาตำแหน่งเลขที่อยู่ของ Connection Memory ที่ตรงกับหมายเลขช่องเวลาทางเอาต์พุตโดยใช้สมการที่ 1 เช่นกันซึ่งจะได้

CM_Address = (9 X 32) + 19 = 307

ดังนั้นการสวิตช์สัญญาณจากอินพุตไปเอาต์พุตทำได้โดยการเขียนค่า DM_Address คือค่า 167 ลงในตำแหน่งเลขที่อยู่ที่ CM_Address คือค่า 307 ลงใน Connection Memory

แนวความคิดในการออกแบบ

จากเทคนิคการออกแบบระบบทนต่อความผิดพร่องดังที่ได้กล่าวมาแล้วนั้นพบว่า การออกแบบระบบใดๆ ก็ตามที่ผู้ออกแบบต้องการออกแบบให้ระบบนั้นมีความทนต่อความผิดพลาดได้ ระบบจะมีความซับซ้อนมากขึ้น ความซับซ้อนในที่นี้หมายถึง ระบบมีขนาดใหญ่ขึ้นเนื่องจากมีการเพิ่มวงจรพิเศษหรือซอฟต์แวร์พิเศษเข้าไป เพื่อทำงานแทนในกรณีที่ส่วนของระบบเดิมไม่สามารถทำงานต่อไปได้ หรือเพื่อตรวจสอบว่าส่วนใดของระบบที่ทำงานผิดพลาด ทั้งนี้ระบบจะใหญ่ขึ้นมากหรือน้อยขึ้นอยู่กับเทคนิคที่ผู้ออกแบบใช้ และระดับความเชื่อถือได้ที่ผู้ออกแบบต้องการ เมื่อระบบมีความซับซ้อนมากขึ้นทำให้การนำไปสร้างยากขึ้นเพราะผู้สร้างจะต้องนำเทคนิคต่างๆ มาใช้เพื่อการลดสัญญาณรบกวน ลดความสิ้นเปลืองพลังงานไฟฟ้า ลดความร้อนในการทำงานที่จะเกิดขึ้น ลดขนาด ลดน้ำหนัก ลดต้นทุนการผลิต ฯลฯ นอกจากนี้การพัฒนาระบบยังต้องใช้เวลามากขึ้น และเสียค่าใช้จ่ายสูง แต่สิ่งที่ได้มาคือความเชื่อถือได้ในการทำงานของระบบ

ซึ่งในงานประยุกต์ที่มีความสำคัญมากๆ บางชนิดก็มีความจำเป็นต้องใช้ และมีความคุ้มค่าที่จะใช้ระบบทนต่อความผิดพร่อง

จากหลักการทำงานของวงจรสวิตช์แบบดิจิทัลที่ใช้ในระบบชุมสายโทรศัพท์ ซึ่งวงจรสวิตช์นี้จะทำหน้าที่เป็นตัวสลับข้อมูลที่เข้ามาทางอินพุตช่องหนึ่งให้ออกไปยังเอาต์พุตอีกช่องหนึ่ง ทำให้การแลกเปลี่ยนข้อมูลข่าวสารเกิดขึ้นได้ ซึ่งข้อมูลข่าวสารที่แลกเปลี่ยนนี้อาจเป็นข้อมูลเสียงของการสนทนาของคน 2 คน หรือเป็นข้อมูลที่ส่งผ่านหรือแลกเปลี่ยนกันภายในระบบชุมสายโทรศัพท์ก็ได้ โดยข้อมูลทั้งหมดที่ผ่านวงจรสวิตช์จะต้องอยู่ในรูปแบบที่เหมาะสม คืออยู่ในรูปของสัญญาณดิจิทัล และอยู่ในช่วงเวลาที่เหมาะสมด้วย เพราะวงจรสวิตช์สามารถสลับสัญญาณได้หลายช่องพร้อมกัน ดังนั้นข้อมูลที่เข้าออกแต่ละช่องจึงต้องอยู่ในจังหวะเวลาที่ถูกต้อง

การออกแบบวงจรสวิตช์ขนาดใหญ่ ที่มีความสามารถสลับสัญญาณได้ตั้งแต่หลายร้อย จนถึงหลายพันช่องได้พร้อมกัน ในอดีตนั้นจะใช้วิธีการแบ่งสวิตช์ออกเป็น 3 ตอน คือ Time Switch - Space Switch - Time Switch (ธวัชชัย เลื่อนฉวี. 2533) (Neufang K. 1981.) ซึ่งเป็นการใช้วงจรสวิตช์ขนาดเล็กหลายตัวมาต่อกันเพื่อให้สามารถสวิตช์สัญญาณจำนวนหลายๆ ช่องได้ ทั้งนี้เนื่องมาจากมีข้อจำกัดทางเทคโนโลยีบางอย่าง และผู้ออกแบบต้องการลดต้นทุนการผลิต ข้อเสียของวงจรสวิตช์ในอดีตคือ มีค่าหน่วงเวลาของสัญญาณเมื่อผ่านวงจรสวิตช์สูง เพราะมีการแยกวงจรออกเป็น 3 ตอน แต่ในปัจจุบันนี้เทคโนโลยีทางอิเล็กทรอนิกส์ได้เจริญไปมากการสร้างวงจรรวมที่มีความซับซ้อนมาก และมีความเร็วในการทำงานสูง สามารถทำได้ไม่แพงนัก และมีความน่าเชื่อถือสูง อายุการใช้งานยาวนาน ทำให้การสร้างวงจรสวิตช์ขนาดกลางไม่จำเป็นต้องแบ่งวงจรออกเป็นหลายตอนเช่นในอดีต

ในงานวิจัยนี้จะใช้ไอซีที่ผลิตโดยบริษัทไมเทลคอร์โปเรชัน ซึ่งเป็นไอซีที่ถูกออกแบบมาเพื่อ สร้างวงจรสวิตช์โดยเฉพาะ มีไอซีที่ใช้อยู่ 2 เบอร์คือ MT9080 และ MT9085 ซึ่งแต่ละตัวสามารถถูกควบคุมให้ทำงานได้หลายอย่าง โดยที่เบอร์ MT9080 จะทำหน้าที่เป็น Connection Memory (CM) และเป็น Data Memory (DM) ส่วนเบอร์ MT9085 ทำหน้าที่เป็นตัวแปลงสัญญาณแบบอนุกรมเป็นสัญญาณแบบขนาน (serial to parallel conversion) และ แปลงสัญญาณขนานเป็นอนุกรม (parallel to serial conversion) (Mitel Corporation. 1991.)

คุณสมบัติของวงจรสวิตช์

ได้กำหนดคุณสมบัติของวงจรสวิตช์ที่ต้องการออกแบบไว้ดังต่อไปนี้

1. สามารถสวิตช์สัญญาณได้อย่างน้อย 1,024 ช่อง
2. สามารถตรวจหาข้อผิดพลาดที่เกิดขึ้นในวงจรได้ และสามารถแจ้งข้อผิดพลาดให้ผู้ใช้รับรู้ได้
3. สามารถทำงานต่อไปได้เมื่อมีข้อผิดพร่องเกิดขึ้นกับบางส่วนของวงจร

4. สามารถควบคุมการทำด้วยวงจรไมโครโพรเซสเซอร์โดยทางซอฟต์แวร์
5. สามารถทำการซ่อมแซมวงจรได้โดยไม่กระทบกระเทือนการทำงานของระบบ
6. สามารถกู้ระบบคืนได้ภายในระยะเวลาไม่เกิน 0.1 วินาที
7. สัญญาณรบกวนที่เกิดขึ้นระหว่างการกู้ระบบจะต้องมีค่าไม่มากจนสามารถรับรู้ได้ด้วยการฟัง
8. วงจรอาจแบ่งออกเป็นหลายมอดูลได้ โดยแต่ละมอดูลจะต้องมีขนาดไม่เกินมาตรฐานดับเบิ้ลยูโรการ์ด และจะต้องมีอัตราสิ้นเปลืองกำลังไฟฟ้าเฉลี่ยไม่เกิน 5 วัตต์ต่อมอดูล
9. สามารถขยายจำนวนช่องสัญญาณที่สวิตช์ได้

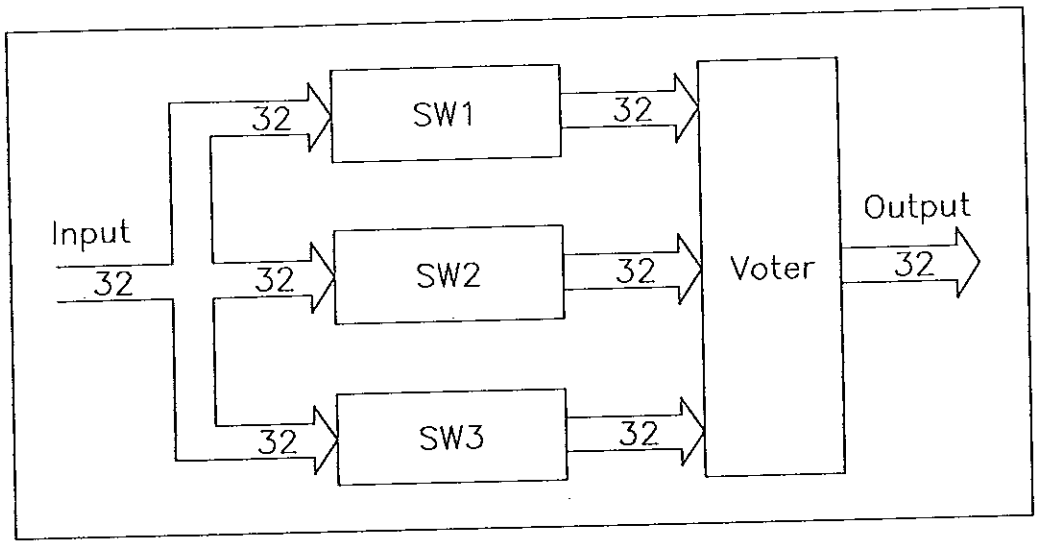
เทคนิคที่ใช้ในการออกแบบวงจรสวิตช์

ในงานวิจัยนี้จะทำการออกแบบวงจรสวิตช์แบบดิจิทัลสำหรับระบบชุมสายโทรศัพท์ที่มีความสามารถในการสวิตช์สัญญาณได้ 1,024 ช่องพร้อมกัน โดยสัญญาณอินพุตที่เข้ามาจะเป็นสัญญาณแบบอนุกรม โดยใน 1 กระแสของสัญญาณจะมีจำนวนช่องเท่ากับ 32 ช่อง โดยรูปแบบของสัญญาณจะใช้ตามมาตรฐานของ ST-BUS เมื่อต้องการสร้างวงจรสวิตช์ขนาด 1,024 X 1,024 ช่องจึงต้องมีจำนวนกระแสทั้งหมดเท่ากับ 32 กระแส (32X32=1024)

จากการศึกษาเกี่ยวกับการออกแบบระบบทนต่อความผิดพร่องจากเอกสารและงานวิจัยต่างๆ (Johnson, B.W. 1989.) (Lala, P.K. 1985.) (Avizienis, A. 1978.) พบว่ามีเทคนิคการออกแบบอยู่ 2 เทคนิคคือ TMR และ Duplication with Comparison ผสมกับ Standby Sparing (DCSS) ส่วนเทคนิคอื่นๆ ส่วนมากไม่เหมาะกับลักษณะงานเช่นวงจรสวิตช์ เพราะจากการพิจารณาลักษณะของวงจรสวิตช์พบว่าวงจรสวิตช์มีจำนวนสัญญาณอินพุตและเอาต์พุตมากถึง 64 เส้น และมีการเชื่อมต่อภายในที่ซับซ้อนพอสมควร รวมทั้งลักษณะสัญญาณเป็นลักษณะสุ่ม มีผลทำให้เทคนิคบางอย่างไม่เหมาะสมที่จะนำมาใช้เพราะจะมีผลทำให้วงจรมีขนาดใหญ่และสิ้นเปลือง และมีปัญหาในการสร้างใช้งานจริง ลักษณะของวงจรเมื่อใช้เทคนิคการออกแบบที่เลือกแล้วทั้งสองแสดงไว้ในภาพประกอบ 20 และ 21

จากภาพประกอบ 20 SW1 SW2 และ SW3 เป็นวงจรสวิตช์ที่เหมือนกันทุกประการ มีจำนวนอินพุตและเอาต์พุตอย่างละ 32 เส้น Voter เป็นวงจรที่ทำหน้าที่ผ่านข้อมูลที่ถูกส่งมาจากเอาต์พุตของวงจรสวิตช์ทั้ง 3 โดยจะส่งสัญญาณที่เหมือนกันอย่างน้อย 2 สัญญาณจากวงจรสวิตช์ออกทางเอาต์พุต ดังนั้นถ้ามีวงจรสวิตช์วงจรใดวงจรหนึ่งทำงานบกพร่อง หมายถึงสัญญาณที่เอาต์พุตแตกต่างจากเอาต์พุตของวงจรอื่นวงจร ลงคะแนนเสียงจะทำหน้าที่กั้นสัญญาณจากวงจรสวิตช์ไม่ให้ออกไปทางเอาต์พุต แต่จะเลือกเอาสัญญาณที่ถูกต้องส่งออกไปแทน ในกรณีที่มีวงจรสวิตช์มากกว่า 1 วงจรที่บกพร่อง ซึ่งอาจจะมีผลทำให้สัญญาณจากเอาต์พุตทั้ง 3 แตกต่างกันทั้งหมด ในกรณีนี้วงจรลงคะแนนเสียงจะไม่สามารถทำงานได้อย่างถูกต้อง จึงเป็นข้อจำกัดของเทคนิค TMR ที่ไม่สามารถทำงานได้ถูกต้องเมื่อมีวงจรมากกว่า 1 วงจรทำงาน

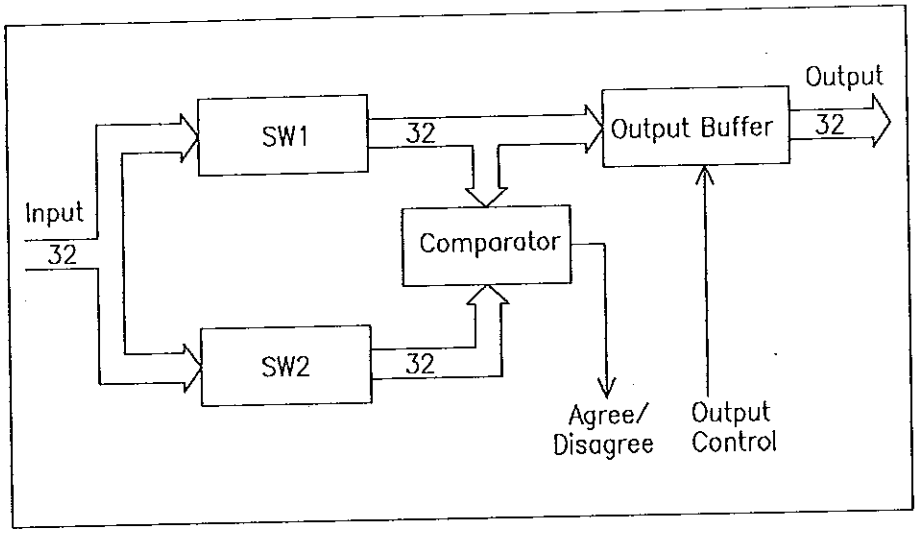
ผิดพลาด และจากภาพจะเห็นว่าไม่มีส่วนที่ใช้ในการตรวจหาข้อผิดพร่อง ทำให้ไม่สามารถทราบได้ว่าขณะทำงานนั้นวงจรส่วนใดทำงานผิดปกติบ้าง



ภาพประกอบ 20 การออกแบบมอดูลวงจรสวิตช์โดยใช้เทคนิค TMR

เมื่อพิจารณาวงจรลงคะแนนเสียงจะเห็นว่ามีจำนวนอินพุตถึง 96 อินพุตและมีจำนวนเอาต์พุต 32 เอาต์พุต ซึ่งนับว่าเป็นจำนวนที่สูงมาก จากวงจรทั่วไปในการสร้างวงจรลงคะแนนเสียงชนิดสมวาร (synchronous voter) (Johnson, B.W. 1989.) พบว่าจะต้องใช้ฟลิปฟลอป 8 ตัว และเกตอีก 4 ตัว ต่อสัญญาณอินพุต 3 สัญญาณและเอาต์พุต 1 สัญญาณ ดังนั้นกรณีนี้จึงต้องใช้ฟลิปฟลอป ถึง 256 ตัวและเกตอีก 128 ตัวในการสร้าง ผนวกกับวงจรสับเปลี่ยนช่องเวลาอีก 3 วงจร ซึ่งแต่ละวงจรก็มีขนาดค่อนข้างใหญ่ ทำให้การออกแบบและสร้างไม่สามารถทำได้อย่างมีประสิทธิภาพ เมื่อต้องการจำกัดขนาดมอดูลและอัตราสิ้นเปลืองกำลังไฟ

จากภาพประกอบ 21 ซึ่งแสดงการออกแบบวงจรสวิตช์โดยใช้วิธี DCSS จะเห็นว่ามีวงจรสวิตช์ที่เหมือนกันอยู่สองวงจรคือ SW1 และ SW2 เอาต์พุตของวงจรสวิตช์ทั้งสองจะถูกส่งไปยังวงจรเปรียบเทียบสัญญาณขนาด 32 บิต ซึ่งวงจรนี้จะทำงานในลักษณะเปรียบเทียบบิตต่อบิตแบบเข้าจังหวะ ซึ่งภายในวงจรเปรียบเทียบนี้นอกจากจะมีจำนวนอินพุตน้อยกว่าวงจรลงคะแนนเสียงแล้ว การทำงานยังซับซ้อนน้อยกว่าด้วย สัญญาณที่ออกจากวงจรเปรียบเทียบจะเป็นผลลัพธ์ของการเปรียบเทียบของสัญญาณทุกๆ บิตที่เอาต์พุต ซึ่งสัญญาณนี้จะถูกส่งไปบอกให้วงจรไมโครโพรเซสเซอร์ที่ควบคุมการทำงานของวงจรสวิตช์อยู่รับทราบว่าสวิตช์ทำงานถูกต้องหรือไม่



ภาพประกอบ 21 การออกแบบมอดูลวงจรสวิตช์โดยใช้เทคนิค DCSS

สัญญาณ Output Control ใช้สำหรับควบคุมวงจรเอาต์พุตบัฟเฟอร์ให้ขับสัญญาณออกหรือไม่ เมื่อวงจรเปรียบเทียบตรวจสอบได้ว่าสัญญาณจากวงจรสวิตช์ทั้งสองไม่ตรงกันซึ่งหมายถึงมีวงจรสวิตช์วงจรใดวงจรหนึ่งที่ทำงานผิดพลาด วงจรเปรียบเทียบก็จะส่งสัญญาณไปแจ้งให้วงจรไมโครโพรเซสเซอร์รับรู้ จากนั้นวงจรไมโครโพรเซสเซอร์จะทำการตัดมอดูลที่บกพร่องออกจากระบบ โดยการควบคุมที่สัญญาณ Output Control จากนั้นก็จะควบคุมให้มอดูลที่สำรองที่เสียบอยู่ในระบบ และยังสามารถทำงานได้อย่างถูกต้องมารับหน้าที่ทำงานแทน

จากการศึกษาพบว่าเทคนิคทั้ง 2 ที่กล่าวมามีข้อดีและข้อเสียที่แตกต่างกันซึ่งพอที่จะสรุปได้ดังตาราง 1

จากตารางพบว่าถ้าไม่พิจารณาถึงความสิ้นเปลืองและขนาดของ TMR แล้ว วิธีนี้จะมีข้อดีคือสามารถทนทานต่อความผิดพลาดได้โดยใช้เพียง 1 มอดูล ส่วนอีกวิธีหนึ่งจะต้องมีถึง 2 มอดูลจึงจะทนต่อความผิดพร่องได้ แต่ถ้าพิจารณาลงไปว่า TMR ไม่สามารถตรวจสอบได้ว่ามอดูลใดบ้างหรือวงจรสวิตช์ภายในมอดูลวงจรใดบ้างที่ทำงานผิดพลาด ดังนั้นถ้ามีวงจรสวิตช์ทำงานผิดพลาดมากกว่า 1 วงจรวิธีนี้ก็จะให้ผลลัพธ์ที่ไม่ถูกต้อง และถ้าสามารถออกแบบให้วิธี TMR สามารถตรวจสอบว่ามอดูลใดบ้างที่บกพร่อง ก็จำเป็นจะต้องมีมอดูลที่เหมือนกันอย่างน้อย 2 มอดูลในระบบ เพื่อทำงานแทนในกรณีที่อีกมอดูลหนึ่งทำงานผิดพลาด ซึ่งเมื่อใช้ TMR 2 มอดูลทำให้ระบบมีขนาดใหญ่ขึ้นมากเมื่อเทียบกับเทคนิค DCSS

ตาราง 1 เปรียบเทียบข้อดีข้อเสียระหว่างการออกแบบโดยเทคนิค TMR และ DCSS

TMR	DCSS
ออกแบบได้ง่าย (ยกเว้นวงจรลงคะแนนเสียง)	ออกแบบได้ยากกว่า TMR
นำไปสร้างได้ยากเนื่องจากวงจรมีขนาดใหญ่มาก เพราะต้องมีวงจรสวิตช์ที่เหมือนกัน 3 วงจร และต้องมีวงจรลงคะแนนเสียงที่มี 96 อินพุต และ 32 เอาต์พุต	นำไปสร้างได้ง่ายกว่าเพราะใน 1 มอดูลมีวงจรสวิตช์เพียง 2 วงจร และวงจรเปรียบเทียบมีขนาด 64 อินพุต และ 32 เอาต์พุต
อัตราการสิ้นเปลืองกำลังไฟฟ้าต่อมอดูลสูงกว่า	อัตราการสิ้นเปลืองกำลังไฟฟ้าต่อมอดูลต่ำกว่า
ในหนึ่งมอดูลสามารถทนทานต่อความผิดพลาดได้เมื่อวงจรสวิตช์ไม่เกิน 1 วงจรทำงานบกพร่อง	ไม่สามารถทำงานได้อย่างถูกต้องถ้ามีการใช้งานเพียงมอดูลเดียวและมีวงจรสวิตช์ 1 วงจรที่ไม่สามารถทำงานได้ จึงต้องมี 2 มอดูลเป็นอย่างน้อย
ใช้วิธีซ่อนข้อผิดพลาดจึงไม่สามารถตรวจสอบได้ว่ามอดูลใดหรือวงจรสวิตช์ใดทำงานผิดพลาด	ใช้วิธีตรวจหาข้อผิดพลาดจึงสามารถตรวจสอบได้ว่ามีมอดูลใดบ้างที่ทำงานผิดพลาด แล้วสวิตช์ไปใช้งานโมดูลที่ทำงานได้ถูกต้อง

ในงานวิจัยนี้จะเลือกใช้เทคนิค DCSS ในการออกแบบวงจรสวิตช์ขนาด 1,024 X 1,024 ช่อง เพราะมีความเหมาะสมที่สุด หลักการทำงานของเทคนิคนี้จะคล้ายกับเทคนิค Pair and a Spare มากเพียงแต่ว่าในงานวิจัยนี้ได้กำหนดให้การทำงานของวงจรสวิตช์ถูกควบคุมโดยซอฟต์แวร์ ดังนั้นความยุ่งยากในการสร้างวงจรสวิตช์ (N to 2 Switch) จะลดลงไปได้มาก ทำให้มอดูลมีความซับซ้อนทางฮาร์ดแวร์น้อยลง แต่ต้องมีการเพิ่มการทำงานของซอฟต์แวร์บางส่วนเข้าไปเพื่อทำงานแทน

การออกแบบมอดูลวงจรสวิตช์

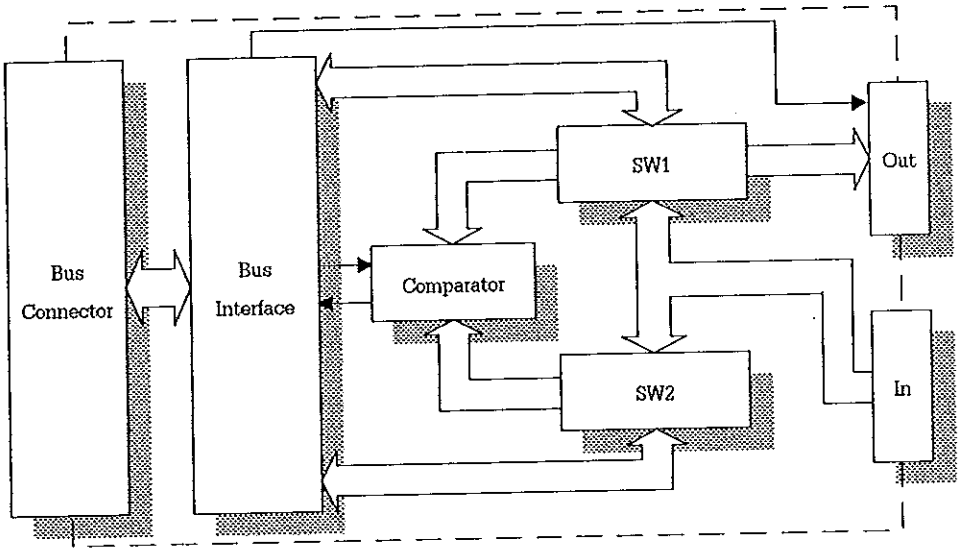
ดังที่ได้กล่าวมาแล้วว่าวิธี DCSS เป็นวิธีที่เหมาะสมที่สุด ในการสร้างวงจรสวิตช์ในงานวิจัยนี้ ผู้วิจัยจึงได้ออกแบบมอดูลวงจรสวิตช์ขึ้นโดยใช้เทคนิคดังกล่าว ซึ่งภาพประกอบ 22 ได้แสดงมอดูลของวงจรสวิตช์ 1 มอดูลที่ประกอบด้วยวงจรสับเปลี่ยนช่องเวลาแบบอนุกรม 2 วงจร วงจรเปรียบเทียบ 1 วงจร และวงจรบัฟเฟอร์ทางเอาต์พุตอีก 1 วงจร และเพื่อให้วงจรสามารถทำงานได้จริง จำเป็นจะต้องมีวงจรส่วนอินเตอร์เฟสสำหรับไมโครโพรเซสเซอร์ และวงจรบัฟเฟอร์ทางอินพุตอีกด้วย

แต่ละมอดูลของวงจรสวิตช์ที่ออกแบบ จะมีความสามารถในการตรวจสอบความผิดพลาดได้เท่านั้น ถ้ามีข้อผิดพลาดเกิดขึ้นที่จุดใดจุดหนึ่งในมอดูล มอดูลนั้นก็ไม่สามารถทำงานได้อย่างถูกต้อง เพื่อให้

บรรลุวัตถุประสงค์ในการออกแบบวงจรสวิตช์ที่ทนต่อความผิดพร่อง ในระบบจึงจำเป็นจะต้องมีมอดูลวงจรสวิตช์อย่างน้อย 2 มอดูลทำงานอยู่ โดยมีมอดูลหนึ่งที่หน้าที่เป็นมอดูลหลัก และมอดูลที่เหลือทำหน้าที่เป็นมอดูลสำรอง

วงจรเชื่อมต่อบัส (bus interface circuit)

เพื่อให้การสร้างและการทดสอบการทำงานของวงจรต้นแบบได้สะดวกและรวดเร็ว ระบบบัสที่ใช้ในงานวิจัยนี้จึงไม่ได้ใช้ระบบบัสที่ต่อจากไมโครโพรเซสเซอร์โดยตรง แต่เป็นการจำลองบัสขึ้นมาโดยใช้การ์ด 8255 (ทีมงานอีทีที. 2535.) ของบริษัท อีทีที จำกัด การ์ดนี้จะมีไอซี 8255 ทั้งหมด 3 ตัวโดยสามารถต่ออินพุตเอาต์พุตพอร์ตขนาด 8 บิตได้ทั้งสิ้น 9 พอร์ต ซึ่งมากเพียงพอที่จะนำไปสร้างระบบบัสจำลองสำหรับใช้งานได้ โดยการ์ดนี้เป็นการ์ดที่ต้องเสียบใช้งานบนเครื่องคอมพิวเตอร์ส่วนบุคคล ดังนั้นจึงสามารถควบคุมการทำงานของการ์ดด้วยซอฟต์แวร์จากเครื่องคอมพิวเตอร์ส่วนบุคคลได้ โดยการจัดสรรพอร์ตต่างๆ ของไอซี 8255 เป็นดังตาราง 2 และภาพถ่ายของการ์ด 8255 ที่ใช้งานได้แสดงไว้ในภาคผนวก ก.



ภาพประกอบ 22 แผนภาพวงจรสมบูรณ์ของมอดูลวงจรสวิตช์

ตาราง 2 การจัดสรรพอร์ตของไอซี 8255 บนการ์ด 8255

พอร์ต	อินพุต/เอาต์พุต	หน้าที่
P1A และ P1B	เอาต์พุต	บัสเลขที่อยู่ A0 - A15
P1C	เอาต์พุต	Address Modifier -AM0 - -AM7
P2A และ P2B	อินพุตและเอาต์พุต	บัสข้อมูล D0 - D15
P3A	อินพุต	ตรวจสอบสถานะการทำงานของมอดูล Fail0 - Fail7
P3B	อินพุต	ตรวจสอบว่ามีมอดูลเสียบอยู่หรือไม่ Loop0-Loop7
P3C	เอาต์พุต	ควบคุมการเขียนอ่าน R/-W

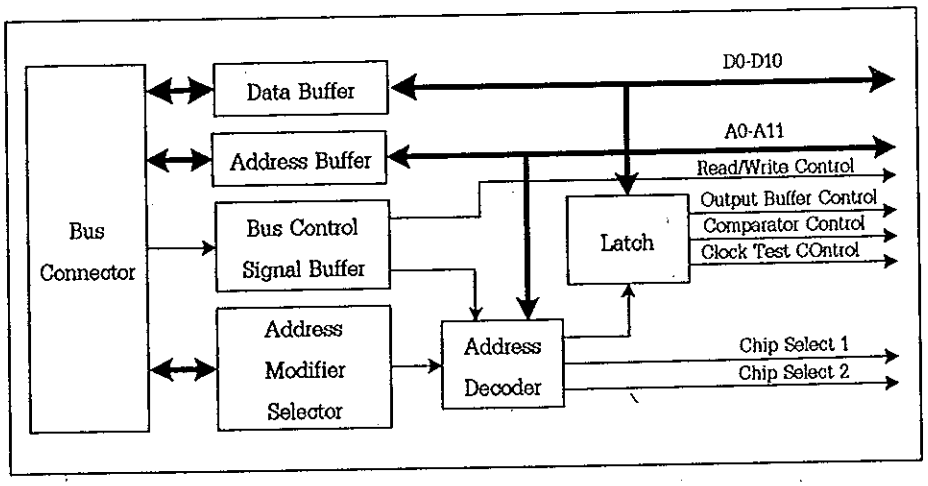
การออกแบบสัญญาณต่างๆ ของระบบบัสมีแนวความคิดบางส่วนมาจากระบบบัส VME (นรินทร์ ว่องพงศาวิวัฒน์. 2533.) ซึ่งเป็นบัสที่ใช้กันอย่างแพร่หลายในระบบคอมพิวเตอร์เพื่องานอุตสาหกรรม และเนื่องจากต้องการออกแบบมอดูลของวงจรสวิตช์ที่มีการถอดเข้าออกได้ง่ายในกรณีที่ต้องการถอดมอดูลที่เสียออกแล้วเสียบมอดูลที่ดีเข้าไปแทน ดังนั้นการเชื่อมต่อที่เหมะสมควรจะเป็นการเชื่อมต่อโดยใช้ขั้วต่อ (connector) ในงานวิจัยนี้จึงใช้การเชื่อมต่อมอดูลของวงจรสวิตช์กับบัสโดยใช้ขั้วต่อแบบ DIN ขนาด 64 ขา การจัดขาสัญญาณต่างๆ ของบัสที่ได้ออกแบบไว้แสดงไว้ในภาคผนวก ก.

ในการออกแบบระบบบัสนั้นผู้วิจัยได้ออกแบบให้สามารถเสียบมอดูลได้สูงสุด 8 มอดูล จึงจะต้องมีการสร้างแผงหลัง (backplane) สำหรับเสียบมอดูลได้ทั้ง 8 มอดูล โดยที่แผงหลังจะเชื่อมต่อกับการ์ด 8255 ที่เสียบอยู่ในเครื่องคอมพิวเตอร์ส่วนบุคคล ในการสร้างแผงหลังเพื่อการทดสอบมอดูลวงจรสวิตช์ในงานวิจัยนี้นั้นจะสร้างแผ่นวงจรพิมพ์สำหรับแผงหลังขนาด 4 มอดูลเท่านั้น แต่อย่างไรก็ตาม โดยการออกแบบระบบบัสและอุปกรณ์ที่ใช้แล้วสามารถที่จะขยายเป็น 8 มอดูลโดยไม่ยาก วงจรและภาพถ่ายของวงจรแผงหลังได้แสดงไว้ในภาคผนวก ก.

สัญญาณต่างๆ บนบัสจะถูกควบควบคุมให้เป็นไปตามขั้นตอนที่ถูกต้องโดยซอฟต์แวร์บนเครื่องคอมพิวเตอร์ส่วนบุคคล และก่อนการรับหรือส่งสัญญาณใดๆ จากบัสจะต้องมีการโปรแกรมการทำงานของไอซี 8255 ให้ถูกต้องเสียก่อน การทำงานของซอฟต์แวร์ที่ใช้ควบคุมระบบบัสนี้จะได้กล่าวในส่วนของการออกแบบซอฟต์แวร์

วงจรเชื่อมต่อบัสในภาพประกอบ 23 มีหน้าที่หลักในการบัฟเฟอร์สัญญาณเลขที่อยู่ สัญญาณข้อมูล และสัญญาณควบคุมการทำงานของบัส และถอดรหัสสัญญาณเลือกชิพ (chip select) สำหรับใช้ในการเปิดทาง (enable) การทำงานของวงจร Connection Memory และวงจรแลตช์ (latch) ซึ่งใช้ค้างสัญญาณควบคุมต่างๆ คือสัญญาณควบคุมการทำงานของวงจรบัฟเฟอร์เอาต์พุต (Output Buffer

Control) วงจรเปรียบเทียบ (Comparator Control) และวงจรตรวจสอบสัญญาณนาฬิกา (Clock Test Control)



ภาพประกอบ 23 แผนภาพการทำงานของวงจรเชื่อมต่อบัส

วงจร Address Modifier Selector คือสวิตช์สำหรับเลือกว่าจะให้มอดูลทำงานที่เลขที่อยู่ช่วงใด การที่จะนำมอดูลไปเสียบในสล็อตใดๆ ของแผงหลังจะต้องมีการกำหนดค่าในวงจรให้ถูกต้องด้วย

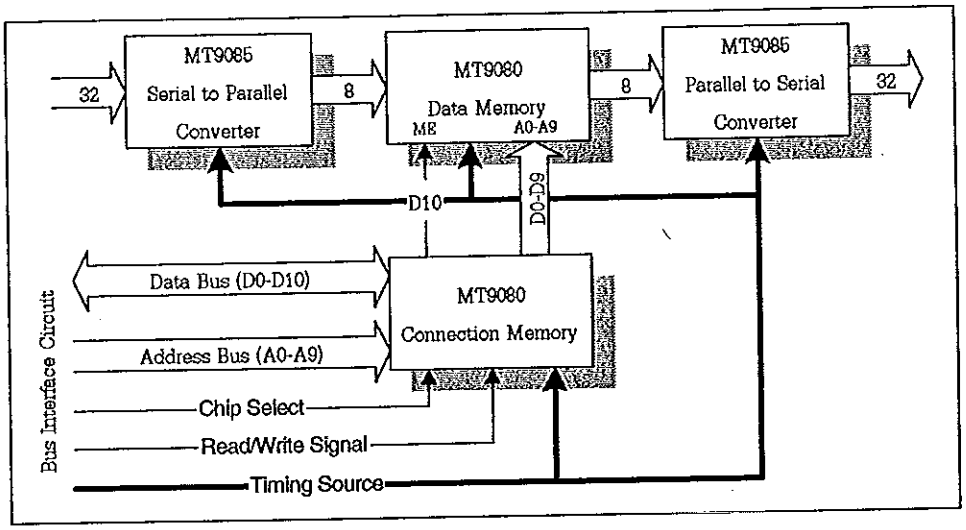
การออกแบบวงจรสับเปลี่ยนช่องเวลา

ภาพประกอบ 24 แสดงแผนภาพการทำงานของวงจรสับเปลี่ยนช่องเวลาขนาด 1,024 X 1,024 ช่องแบบอนุกรมที่ได้มีการออกแบบขึ้น โดยสัญญาณอินพุตแบบอนุกรมทั้ง 32 กระแสที่เข้ามาจะถูกแปลงให้อยู่ในรูปสัญญาณแบบขนาน โดยใช้ไอซีเบอร์ MT9085 ข้อมูลแบบขนานที่มีความกว้าง 8 บิตถูกส่งไปให้ Data Memory เพื่อนำข้อมูลที่เข้ามาไปเก็บในเลขที่อยู่ที่เหมาะสม ไอซีที่ทำหน้าที่เป็น Data Memory คือ MT9080

ไอซีเบอร์ MT9080 ที่ทำหน้าที่เป็น Connection Memory ซึ่งถูกควบคุมโดยไมโครโพรเซสเซอร์ ผ่านทางบัสข้อมูล (data bus) D0-D10 บัสเลขที่อยู่ (address bus) A0-A9 สัญญาณควบคุมการเขียนอ่าน (Read/Write Signal) และสัญญาณเลือกชิพ (chip select)

แหล่งฐานเวลา (Timing Source) ใช้สำหรับให้จังหวะในการทำงานให้สอดคล้องกันไปโดยประกอบด้วยสัญญาณคือ -C4 -F0 และ C16 ซึ่งเป็นสัญญาณนาฬิกาความถี่ 4.096 เมกกะเฮิร์ตซ์ สัญญาณ Frame 0 และสัญญาณนาฬิกาความถี่ 16.384 เมกกะเฮิร์ตซ์ตามลำดับ โดยสัญญาณ Frame 0 มี

ลักษณะเป็นพัลส์ (pulse) ที่บอกจุดเริ่มต้นของเฟรมเพื่อให้วงจรนับเลขที่อยู่ภายใน Data Memory และ Connection Memory ทำงานเป็นไปตามจังหวะที่ถูกต้อง



ภาพประกอบ 24 วงจรสับเปลี่ยนช่องเวลาที่ออกแบบขึ้น

วงจรเปรียบเทียบ (comparator circuit)

จุดประสงค์ของการใช้วงจรเปรียบเทียบก็เพื่อที่จะตรวจจับข้อผิดพลาดในวงจรสวิตช์ โดยการทำงานของวงจรเปรียบเทียบนี้จะนำสัญญาณเอาต์พุตทุกบิตของวงจรสวิตช์ที่เหมือนกัน 2 วงจรมาเปรียบเทียบกัน ถ้าผลของการเปรียบเทียบบิตไม่ถูกต้องแสดงว่ามีวงจรสวิตช์วงจรใดวงจรหนึ่งหรือทั้งสองวงจรทำงานผิดพลาด ผลลัพธ์ที่ได้จากการเปรียบเทียบจะถูกส่งไปให้ไมโครโพรเซสเซอร์

เนื่องจากการเปรียบเทียบจะต้องกระทำหลังจากการสวิตช์ข้อมูลเสร็จสิ้นแล้ว ดังนั้นจำนวนอินพุตของวงจรเปรียบเทียบสัญญาณจะมีขนาดอินพุตคือ 64 เส้น และมีเอาต์พุตแสดงผลของการเปรียบเทียบเพียงเส้นเดียว จากภาพประกอบ 25 เป็นวงจรเปรียบเทียบสัญญาณ 32 บิตกับสัญญาณ 32 บิตที่ได้ออกแบบขึ้น โดยใช้ไอซีเปรียบเทียบสัญญาณ ขนาด 8 บิตกับ 8 บิต จำนวน 4 ตัว เอาต์พุตที่ได้จากการเปรียบเทียบบิตทั้ง 8 จะให้ลอจิกต่ำเมื่อค่าที่อินพุตเท่ากัน สัญญาณ E1 ถึง E4 ที่ได้จากการเปรียบเทียบจะส่งไปยัง Programmable Array Logic (PAL) ซึ่งเป็นไอซีที่โปรแกรมฟังก์ชันการทำงานได้ ในที่นี้จะโปรแกรมให้ไอซีนี้ทำหน้าที่เป็นวงจรเปรียบเทียบแบบเข้าจังหวะ (synchronous comparator) โดยมีสัญญาณนาฬิกา Clock เป็นตัวให้จังหวะ นอกจากทำหน้าที่เป็นวงจรเปรียบเทียบแล้ว ไอซีนี้ยังถูกโปรแกรมให้ทำหน้าที่เป็นวงจรตรวจสอบสัญญาณนาฬิกาของระบบได้อีกด้วย

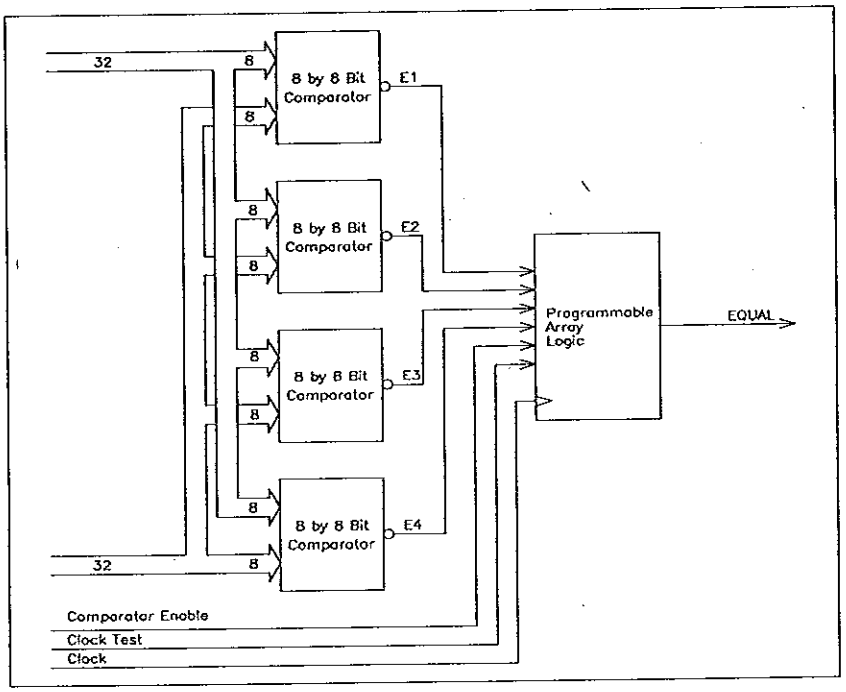
การควบคุมให้วงจรทำงานเป็นวงจรเปรียบเทียบสัญญาณ หรือให้เป็นวงจรตรวจสอบสัญญาณนาฬิกานั้นสามารถกระทำได้โดยการควบคุมระดับลอจิกของสัญญาณ Comparator Enable

ในการทำงานเป็นวงจรเปรียบเทียบ สัญญาณเอาต์พุตคือ EQUAL จะมีความสัมพันธ์กับสัญญาณจาก E1 ถึง E4 และสัญญาณนาฬิกาเท่านั้น ส่วนในการทำงานเป็นวงจรตรวจสอบสัญญาณนาฬิกาว่ามีอยู่หรือไม่นั้น เอาต์พุต EQUAL จะมีค่าตรงกับระดับลอจิกของสัญญาณ Clock Test ซึ่งมีสัญญาณนาฬิกาเป็นผู้ให้จังหวะการทำงาน ในกรณีที่ไม่มีสัญญาณนาฬิกาสัญญาณ EQUAL จะไม่เปลี่ยนแปลงไม่ว่ากรณีใดๆ

ในงานวิจัยนี้ได้ออกแบบฟังก์ชันการทำงานของ PAL ไว้ดังนี้

EQUAL = (!CMPEN . CLKTEST) +
(CMPEN . CLKTEST . EQUAL . !E1 . !E2 . !E3 . !E4)

โดย CMPEN คือ สัญญาณ Comparator Enable, CLKTEST คือสัญญาณ Clock Test



ภาพประกอบ 25 วงจรเปรียบเทียบ

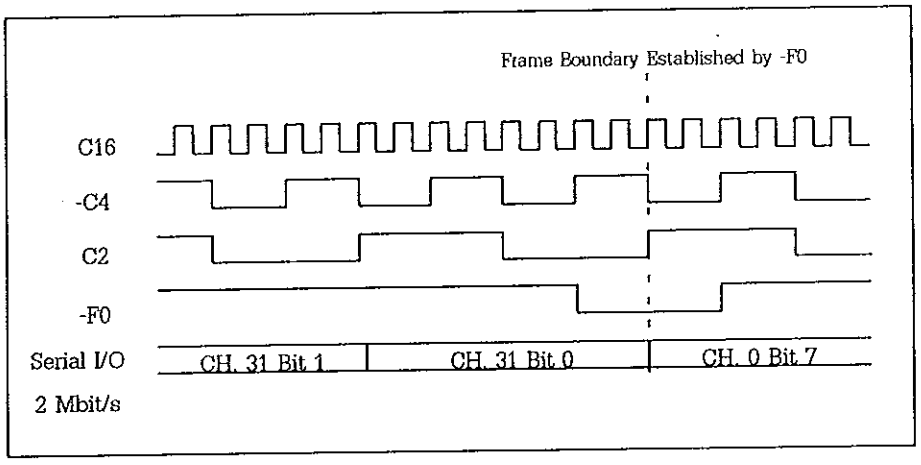
จากสมการข้างต้น PAL จะทำงานเป็นวงจรเปรียบเทียบเมื่อสัญญาณ CMPEN และสัญญาณ CLKTEST มีลอจิกสูง สัญญาณเอาต์พุต EQUAL จะขึ้นอยู่กับสัญญาณเอาต์พุตในอดีต และสัญญาณจาก

วงจรเปรียบเทียบขนาด 8 บิต ทั้ง 4 วงจร (สัญญาณ E1 ถึง E4) ไอซีจะให้เอาต์พุตเป็นลอจิกสูงเมื่อสัญญาณเอาต์พุตในอดีตเป็นลอจิกสูง และสัญญาณ E1 ถึง E4 มีลอจิกต่ำทั้งหมด ในกรณีนี้แสดงว่าสัญญาณอินพุตมีค่าเท่ากัน ถ้าสัญญาณ E1 ถึง E4 สัญญาณใดสัญญาณหนึ่งหรือหลายสัญญาณมีค่าลอจิกสูง เอาต์พุต EQUAL จะได้ลอจิกต่ำ ซึ่งแสดงว่ามีความต่างกันของสัญญาณอินพุต เมื่อเอาต์พุต EQUAL มีลอจิกต่ำแล้ว ในจังหวะการทำงานต่อไปเอาต์พุตจะยังคงเป็นลอจิกต่ำอยู่ตลอดไป ซึ่งสัญญาณนี้จะส่งไปแจ้งให้ไมโครโพรเซสเซอร์รับรู้

ในโหมดการทำงานที่เป็นวงจรเปรียบเทียบนั้น ถ้าเอาต์พุตมีลอจิกต่ำ วงจรจะหยุดการทำงานเป็นวงจรเปรียบเทียบทันที โดยจะคงค่าเอาต์พุตนั้นไว้ การที่จะควบคุมให้วงจรกลับมาทำหน้าที่อีกครั้งหนึ่งทำได้โดยการควบคุมให้สัญญาณ CMPEN มีลอจกต่ำ เมื่อมีสัญญาณนาฬิกาเข้ามาสัญญาณเอาต์พุตจะเปลี่ยนไปตามสัญญาณ Clock Test ทันที ถ้าต้องการให้เอาต์พุตมีลอจิกสูงก็ต้องควบคุมให้ Clock Test มีลอจิกสูงด้วย

การสร้าง Fuse Map ของ PAL นี้ใช้โปรแกรม CUPL (ทีมงานอีทีที. 2535.) ช่วยในการออกแบบ Document ที่ได้จากการคอมไพล์ (compile) โปรแกรมได้แสดงไว้ในภาคผนวก ค.

จากการพิจารณาการทำงานของไอซี MT9085 (Mitel Corporation. 1991.) พบว่าข้อมูลแบบอนุกรมที่เอาต์พุตของวงจรสวิตช์แต่ละบิตที่ส่งออกมานั้นจะถูกส่งออกมาโดยมีความสัมพันธ์กับสัญญาณนาฬิกาสัญญาณนาฬิกาที่มีความถี่ 2.048 เมกกะเฮิร์ตซ์ (C2) ซึ่งเป็นสัญญาณที่ได้จากการหารสัญญาณ C16 ดูภาพประกอบ 26 เนื่องจากสัญญาณแต่ละบิตจะถูกส่งออกที่ขอบขาขึ้นของ C2 ดังนั้นจุดกึ่งกลางของบิตข้อมูลจะอยู่ที่ขอบขาลงของ C2 ดังนั้นสัญญาณ Clock ที่ใช้ในวงจรเปรียบเทียบก็คือส่วนกลับของสัญญาณ C2 นั่นเอง



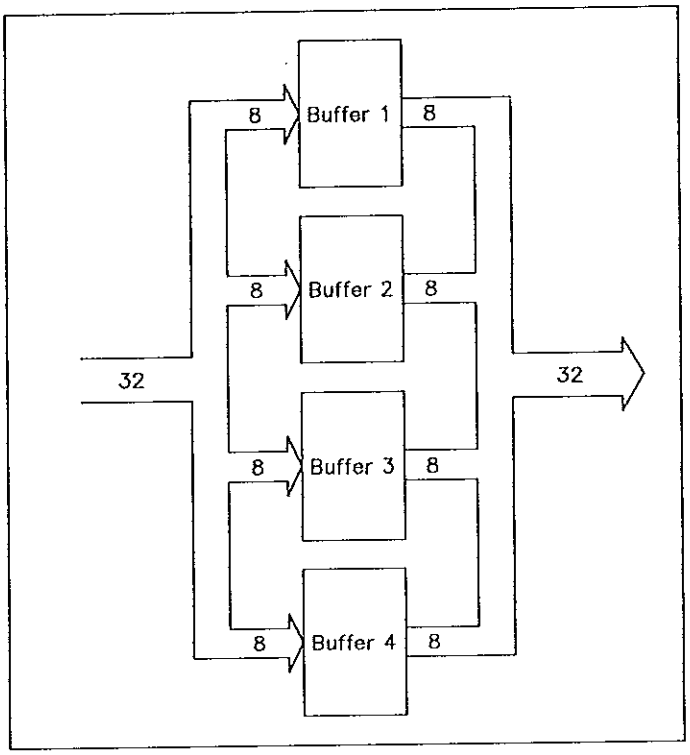
ภาพประกอบ 26 ความสัมพันธ์ระหว่าง C2 กับข้อบิตมูลเอาต์พุต

วงจรบัฟเฟอร์อินพุต (input buffer circuit)

วงจรนี้ทำหน้าที่เป็นกันชนสัญญาณทางด้านอินพุต ซึ่งมีสัญญาณ 32 เส้น และสัญญาณหลังจากผ่านวงจรนี้แล้วจะแยกออกเป็น 2 ทางสำหรับวงจรสับเปลี่ยนช่องเวลาวงจรที่ 1 และวงจรที่ 2 ภาพประกอบ 27 แสดงการทำงานของวงจร

วงจรบัฟเฟอร์เอาต์พุต (output buffer circuit)

การทำงานของวงจรบัฟเฟอร์เอาต์พุตที่ออกแบบขึ้นนอกจากจะมีความสามารถในการขยายสัญญาณจากวงจรสวิตช์แล้วยังจะต้องสามารถให้เอาต์พุตเป็นแบบอิมพีแดนซ์สูงได้ ทั้งนี้เพื่อจะให้สัญญาณเอาต์พุตจากวงจรสวิตช์ไม่มีผลต่อวงจรอื่น เพราะในขณะใดขณะหนึ่งจะมีเอาต์พุตเพียงเอาต์พุตเดียวเท่านั้นที่ทำงาน ส่วนเอาต์พุตอื่นๆ ที่เป็นของวงจรสวิตช์ที่ทำหน้าที่สำรองการทำงาน จะต้องอยู่ในสภาวะอิมพีแดนซ์สูง การทำงานของวงจรบัฟเฟอร์เอาต์พุตนี้ จะถูกควบคุมโดยวงจรไมโครโพรเซสเซอร์ แผนภาพวงจรที่ออกแบบและสร้างขึ้นจริงในงานวิจัยนี้แสดงในภาพประกอบ 28



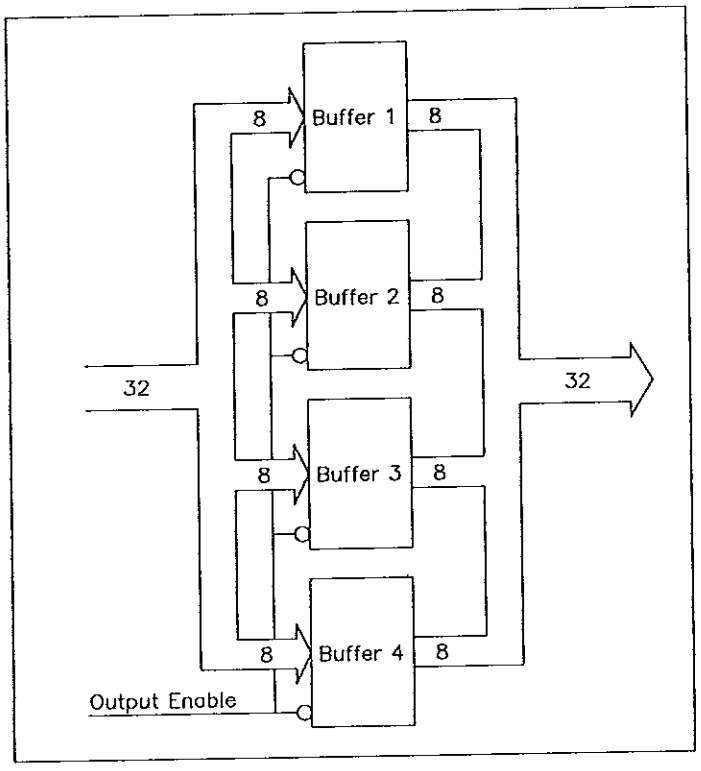
ภาพประกอบ 27 วงจรบัฟเฟอร์อินพุต

มอดูลกำเนิดสัญญาณนาฬิกา (clock generator module)

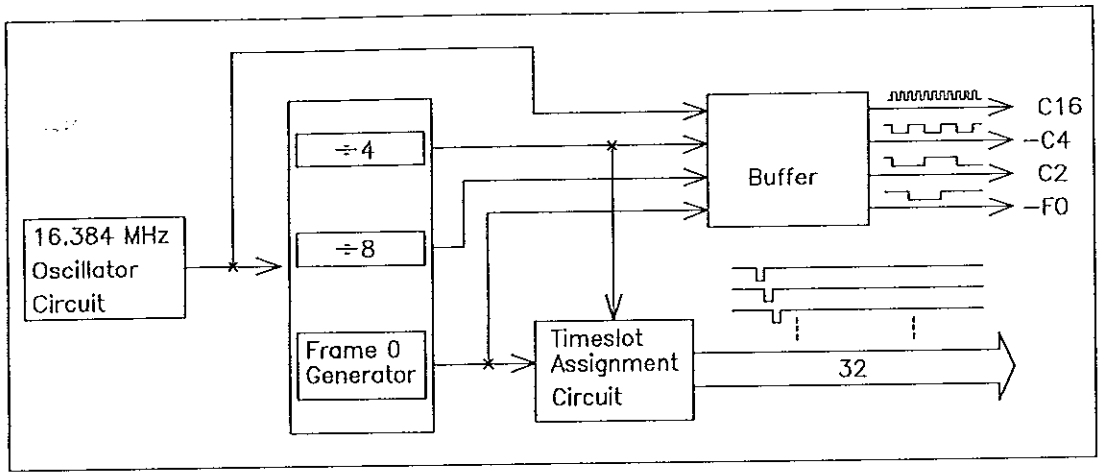
เพื่อให้วงจรสวิตช์ และวงจร SLMA (Subscriber Line Module Analog) (วีระพันธุ์ มุสิกสาร และคณะ. 2535.) สามารถทำงานได้จึงต้องมีการสร้างวงจรกำเนิดสัญญาณนาฬิกาขึ้น โดยจะต้องออกแบบให้สัญญาณต่างมีคาบเวลาและจังหวะที่ถูกต้องตามที่ผู้ผลิตไอซีได้กำหนดไว้ เพื่อให้ระบบทำงานได้อย่างถูกต้องและสามารถทดสอบการทำงานได้

แผนภาพแสดงการทำงานของวงจรกำเนิดสัญญาณนาฬิกาแสดงไว้ดังภาพประกอบ 29 ซึ่งมีวงจรสร้างสัญญาณนาฬิกาความถี่ 16.384 เมกกะเฮิร์ตซ์สำหรับเป็นฐานเวลาในการสร้างสัญญาณนาฬิกาอื่นๆ

สัญญาณจากวงจรจัดสรรช่องเวลา (Timeslot Assignment Circuit) จะถูกใช้ในวงจร SLMA เพื่อควบคุมให้วงจร CODEC (Coder Decoder Circuit) แต่ละตัวป้อนข้อมูลลงในช่องเวลาในจังหวะเวลาที่เหมาะสม



ภาพประกอบ 28 วงจรบัฟเฟอร์เอาต์พุต



ภาพประกอบ 29 ภาพแสดงการทำงานของวงจรสร้างสัญญาณนาฬิกา

บทที่ 4

การออกแบบซอฟต์แวร์

การออกแบบซอฟต์แวร์ที่ใช้สำหรับวงจรสวิตช์ในงานวิจัยนี้แบ่งออกเป็น 3 ส่วนคือซอฟต์แวร์สำหรับควบคุมการทำงานพื้นฐานของวงจรสวิตช์ ซอฟต์แวร์ที่เพิ่มขึ้นเป็นพิเศษเพื่อให้วงจรสามารทนทานต่อความผิดพร่องได้ และซอฟต์แวร์เพื่อการทดสอบการทำงานของระบบ โดยการออกแบบซอฟต์แวร์จะมีการแบ่งออกเป็นมอดูลย่อยๆ ตามฟังก์ชันการทำงานเพื่อให้การนำซอฟต์แวร์นี้ไปรวมกับซอฟต์แวร์อื่นในระบบเป็นไปได้ง่าย

ซอฟต์แวร์ฟังก์ชั่นพื้นฐานของวงจรสวิตช์

หน้าที่หลักของวงจรสวิตช์ก็คือการสวิตช์ข้อมูลอินพุตจากช่องหนึ่งไปยังเอาต์พุตอีกช่องหนึ่งตามแต่ระบบโพรเซสเซอร์กลาง (central processor) จะสั่งงาน นอกเหนือจากการสร้างโปรแกรมควบคุมวงจรสวิตช์ให้มีความสามารถในการสวิตช์ได้ตามต้องการแล้ว โปรแกรมควบคุมการทำงานขั้นพื้นฐานของวงจรสวิตช์ยังจะต้องมีหน้าที่ต่อไปนี้

1. กำหนดค่าเริ่มต้น (initialize) ในการทำงานของวงจรสวิตช์ ค่าเริ่มต้นในการทำงานที่สำคัญของวงจรสวิตช์คือค่าของ Connection Memory ซึ่งเป็นค่าที่จะบอกว่าเอาต์พุตช่องใดรับสัญญาณจากอินพุตช่องใด ซึ่งโดยทั่วไปเมื่อเริ่มใช้งานวงจรสวิตช์ครั้งแรกนั้นยังไม่มีการสวิตช์ค่าใดๆ เกิดขึ้น ดังนั้นค่าที่เหมาะสมที่จะกำหนดให้คือให้ค่าในตำแหน่งเลขที่อยู่ใดๆ ของ Connection Memory มีค่าเท่ากับค่าเลขที่อยู่นั้นๆ ซึ่งหมายถึงอินพุตกับเอาต์พุตของช่องเดียวกันจะถูกต่อเข้าด้วยกัน ในการออกแบบซอฟต์แวร์นี้จะมีการจองเนื้อที่หน่วยความจำส่วนหนึ่งของระบบไมโครโพรเซสเซอร์ที่ใช้ควบคุมการทำงานของวงจรสวิตช์ เพื่อเก็บข้อมูลที่จะต้องเขียนลง Connection Memory ซึ่งจะมีลักษณะเป็นการสำเนาข้อมูลไว้ในกรณีที่ข้อมูลใน Connection Memory ของวงจรสวิตช์สูญหาย โดยใช้กระบวนการทางซอฟต์แวร์ก็สามารถที่จะคัดลอกค่าที่ถูกต้องให้กับ Connection Memory ได้

2. ทดสอบการทำงานของวงจรสวิตช์ โดยจะมีการเรียกใช้เมื่อเริ่มใช้งานวงจรสวิตช์ครั้งแรก และเรียกใช้เมื่อวงจรสวิตช์ทำงานผิดพลาดแล้วต้องการตรวจสอบว่าเกิดจากส่วนใด นอกจากนี้ยังสามารถถูกเรียกใช้ในกรณีที่โพรเซสเซอร์กลางต้องการทราบสถานะการทำงานของวงจรสวิตช์ได้อีกด้วย การทดสอบการทำงานของวงจรสวิตช์มีขั้นตอนดังนี้

2.1 ทดสอบว่าค่าใน Connection Memory ทุกตำแหน่งมีค่าตรงกับค่าในหน่วยความจำหลักของไมโครโพรเซสเซอร์ที่สำเนาไว้หรือไม่

2.2 ทดสอบแต่ละเลขที่อยู่ของ Connection Memory ว่าสามารถเขียนและอ่านค่าต่างๆ ได้อย่างถูกต้องหรือไม่

2.3 ทดสอบว่ามีสัญญาณนาฬิกาในระบบหรือไม่ โดยการควบคุมที่สัญญาณอินพุตของวงจรเปรียบเทียบ

3. สวิตช์สัญญาณ โดยพารามิเตอร์ที่รับเข้ามาจะเป็นหมายเลขช่อง 2 หมายเลขที่ต้องการจะคุยหรือต้องการจะส่งข้อมูลถึงกัน หน้าที่การทำงานของฟังก์ชันนี้คือนำค่าหมายเลขช่องที่ 1 ไปใส่ไว้ใน Connection Memory ในเลขที่อยู่ที่ตรงกับค่าหมายเลขช่องที่ 2 และนำค่าหมายเลขช่องที่ 2 ไปใส่ไว้ใน Connection Memory ในเลขที่อยู่ที่ตรงกับค่าหมายเลขช่องที่ 1 นอกจากนี้ยังจะมีการปรับเปลี่ยนค่าในหน่วยความจำหลักของไมโครโพรเซสเซอร์ที่เป็นตัวสำเนาของ Connection Memory อีกด้วย ในการสวิตช์สัญญาณนี้จะทำการให้ค่าแก่ Connection Memory ทุกตัวที่มีอยู่

4. แสดงค่าใน Connection Memory ตัวที่กำหนดออกทางจอภาพเพื่อตรวจสอบดูความถูกต้อง

นอกจากหน้าที่ทั้ง 4 ข้อที่กล่าวมาแล้ว ซอฟต์แวร์จะต้องมีฟังก์ชั่นสำหรับควบคุมการทำงานของการ์ด 8255 ให้สามารถรับส่งข้อมูลจากระบบบัสได้ตามที่ต้องการ จากรหัสต้นฉบับของโปรแกรมที่ใช้ภาษาซีเขียนขึ้นในภาคผนวก ข. ฟังก์ชันที่ใช้ในการควบคุมการทำงานพื้นฐานของวงจรสวิตช์มีดังนี้

1. void Init8255 (void) เป็นฟังก์ชันสำหรับให้ค่าเริ่มต้นการทำงานของการ์ด 8255
2. int ReadMem (int addr) เป็นฟังก์ชันสำหรับอ่านค่าจากตำแหน่ง addr
3. void WriteMem (int addr, int data) เป็นฟังก์ชันสำหรับเขียนค่า data ลงในตำแหน่ง addr
4. int WriteVerify (int addr, int data) เป็นฟังก์ชันสำหรับเขียนค่า data ลงในตำแหน่ง addr พร้อมตรวจสอบความถูกต้องในการทำงาน
5. int InitSWN (int SWNno) เป็นฟังก์ชันสำหรับให้ค่าเริ่มต้นในการทำงานของวงจรสวิตช์ในมอดูลที่ SWNno
6. int TestClock (int SWNno) เป็นฟังก์ชันสำหรับทดสอบสัญญาณนาฬิกาของมอดูลที่ SWNno
7. int TestCM (int SWNno, int cm) เป็นฟังก์ชันสำหรับทดสอบ Connection Memory ตัวที่ cm ของมอดูลที่ SWNno
8. int Talk (int ch1, int ch2) เป็นฟังก์ชันสำหรับสวิตช์สัญญาณจาก ch1 ไป ch2 และ ch2 ไป ch1

9. void ShowCMMem (int SWNno, int from, int to) เป็นฟังก์ชันสำหรับแสดงค่าใน Connection Memory ของมอดูลที่ SWNno จากเลขที่อยู่ที่ from ถึงเลขที่อยู่ที่ to

10. void EnableCT (int SWNno) เป็นฟังก์ชันสำหรับเปิดทางสัญญาณ Clock Test ของมอดูลที่ SWNno

11. void DisableCT (int SWNno) เป็นฟังก์ชันสำหรับปิดทางสัญญาณ Clock Test ของมอดูลที่ SWNno

ซอฟต์แวร์สนับสนุนการทำงานที่ทนทานต่อความผิดพร่อง

ส่วนของซอฟต์แวร์ส่วนนี้เป็นส่วนที่เพิ่มเติมจากซอฟต์แวร์ควบคุมการทำงานพื้นฐานของวงจรสวิตช์ โดยมีวัตถุประสงค์เพื่อให้วงจรสวิตช์ในระบบที่มีซ้ำซ้อนกันอยู่สามารถทำงานได้ตามวัตถุประสงค์ที่ตั้งไว้ ฟังก์ชันการทำงานที่เพิ่มเติมจากฟังก์ชันพื้นฐานคือ

1. ฟังก์ชันสำหรับตรวจสอบว่ามีมอดูลใดบ้างที่เสียบอยู่ในแผงหลัง โดยการออกแบบฮาร์ดแวร์ของมอดูลวงจรสวิตช์ให้มีการเชื่อมต่อขา 47 และขา 49 ของขั้วต่อบัสของมอดูลวงจรสวิตช์เข้าด้วยกัน จึงสามารถตรวจสอบได้ว่ามีมอดูลเสียบอยู่หรือไม่โดยการอ่านระดับลอจิกที่อินพุต Loop0-Loop7 ของบัส

2. การตรวจสอบสถานะของความถูกต้องในการทำงานของมอดูลวงจรสวิตช์ โดยการอ่านค่าผลลัพธ์การเปรียบเทียบสัญญาณเอาต์พุตของวงจรสวิตช์ที่เหมือนกันทั้ง 2 วงจรในมอดูลวงจรสวิตช์ โดยใช้วงจรเปรียบเทียบที่ได้ออกแบบขึ้น

3. ตรวจสอบการทำงานของวงจรเปรียบเทียบเองว่าทำงานได้ถูกต้องหรือไม่ เพราะถ้าวงจรเปรียบเทียบทำงานไม่ถูกต้องแล้ว ผลของการเปรียบเทียบย่อมไม่ถูกต้องด้วย การทดสอบการทำงานของวงจรเปรียบเทียบทำได้โดยควบคุมสัญญาณอินพุตทั้ง 64 เส้นของวงจรโดยไมโครโพรเซสเซอร์ ซึ่งในการทำงานปกติแล้วอินพุตของวงจรเปรียบเทียบจะขึ้นอยู่กับอินพุตที่เข้ามาของวงจรสวิตช์และค่าต่างๆ ใน Connection Memory ดังนั้นในภาวะปกติเราไม่สามารถกำหนดสัญญาณอินพุตของวงจรเปรียบเทียบจากไมโครโพรเซสเซอร์ได้ แต่ด้วยการออกแบบไอซีเบอร์ MT9080 และจากการออกแบบฮาร์ดแวร์ของมอดูลวงจรสวิตช์ที่ได้ออกแบบเผื่อไว้แล้วนั้น จึงสามารถควบคุมสัญญาณที่อินพุตของวงจรเปรียบเทียบจากไมโครโพรเซสเซอร์ได้โดยการควบคุมที่สัญญาณ ME จากวงจรสับเปลี่ยนช่องเวลาที่ได้ออกแบบ ในภาคผนวก ก. เมื่อสัญญาณ ME เป็นลอจิก 1 สัญญาณที่จะถูกส่งออกทางเอาต์พุตของ Data Memory จะไม่ใช่ข้อมูลที่เก็บอยู่ใน Data Memory แต่จะเป็นค่าตำแหน่งเลขที่อยู่ที่ถูกส่งมาจาก Connection Memory การควบคุม ME ให้มีลอจิก

สูงนั้นทำได้โดยการเขียนค่าที่เหมาะสมลงไปใน Connection Memory จากวงจรสวิตช์จะเห็นว่า สัญญาณ ME จะถูกต่ออยู่กับ D10o ของ Connection Memory ซึ่งค่าของ D10o จะขึ้นอยู่กับ D10i ที่ส่งมาจากไมโครโพรเซสเซอร์ ดังนั้นถ้าสามารถควบคุมลอจิกที่ D10i ได้ก็จะสามารถควบคุมสัญญาณ ME ได้ เช่น ต้องการให้ค่าเอาต์พุตที่ช่องที่ 2 ของกระแสที่ 1 มีค่า 55H จากการโปรแกรมวงจรสวิตช์ที่ได้กล่าวมาแล้วในบทที่ 3 พบว่าจะต้องเขียนค่า 455H (400H + 55H) ลงในเลขที่อยู่ 65 ((2 X 32) + 1) = 65) ของ Connection Memory โดยวิธีที่ได้กล่าวมาจะสามารถควบคุมอินพุตของวงจรเปรียบเทียบให้เป็นไปตามต้องการได้ และสามารถทดสอบการทำงานของวงจรเปรียบเทียบได้

4. การควบคุมบัฟเฟอร์เอาต์พุตให้ขับสัญญาณหรือไม่ขับสัญญาณ เพื่อให้สามารถต่อหลายมอดูลร่วมกันได้ โดยมอดูลที่ทำหน้าที่สำรองจะไม่ขับเอาต์พุต แต่มอดูลที่ทำหน้าที่หลักจะทำหน้าที่ขับเอาต์พุต

5. การควบคุมให้วงจรเปรียบเทียบทำงานหรือไม่ทำงาน ในบางกรณีเช่นเมื่อต้องการสวิตช์สัญญาณซึ่งจะต้องมีการเปลี่ยนแปลงค่าใน Connection Memory และจากการที่ในมอดูลหนึ่งถูกออกแบบให้มีวงจรสวิตช์ 2 วงจรดังนั้นการเปลี่ยนค่าในวงจรสวิตช์ทั้งสองไม่สามารถทำได้พร้อมกัน อันจะมีผลทำให้ผลลัพธ์ที่ได้จากวงจรสวิตช์ทั้งสองมีค่าแตกต่างกันชั่วขณะ ซึ่งนานเพียงพอที่วงจรเปรียบเทียบจะตรวจจับความแตกต่างที่เกิดขึ้นได้ ดังนั้นเพื่อหยุดการทำงานของวงจรเปรียบเทียบในช่วงเวลาที่ต้องการปรับค่าใน Connection Memory จึงต้องมีการยกเลิกการเปรียบเทียบชั่วคราว

6. การตรวจสอบผลลัพธ์ที่ได้จากวงจรเปรียบเทียบ เพื่อตรวจดูว่ามีมอดูลใดบ้างในสวิตช์ที่ทำงานไม่ถูกต้องโดยการอ่านผลลัพธ์ที่ได้จากวงจรเปรียบเทียบของแต่ละมอดูล

7. ฟังก์ชั่นสำหรับการเพิ่มมอดูลวงจรสวิตช์เข้าไปในระบบ โดยมอดูลที่ถูกเสียบเพิ่มเข้าไปจะต้องถูกกำหนดค่าเริ่มต้นต่างๆ ให้เหมือนกับมอดูลที่ทำงานอยู่แล้วทุกประการ เว้นแต่จะไม่มีการสั่งให้ขับสัญญาณเอาต์พุตในกรณีที่มีมอดูลอื่นเสียบอยู่ก่อนแล้ว

8. ฟังก์ชันสำหรับการถอดมอดูลสวิตช์ออกจากระบบ ซึ่งจะถูกเรียกใช้ในกรณีที่ต้องการนำมอดูลที่เสียบอยู่ออกมาซ่อมแซมหรือเพื่อเปลี่ยนมอดูลที่ใหม่กว่าเข้าไปแทนในขั้นตอนของการบำรุงรักษา

ฟังก์ชันภาษาซีที่ออกแบบขึ้นสำหรับส่วนนี้มีดังนี้

1. void EnableOutput (int SWNno) เป็นฟังก์ชันสำหรับเปิดทางสัญญาณเอาต์พุตของมอดูลที่ SWNno

2. void DisableOutput (int SWNno) เป็นฟังก์ชันสำหรับปิดทางสัญญาณเอาต์พุตของมอดูลที่ SWNno
3. void EnableCMP (int SWNno) เป็นฟังก์ชันสำหรับเปิดทางสัญญาณ Comparator Test ของมอดูลที่ SWNno
4. void DisableCMP (int SWNno) เป็นฟังก์ชันสำหรับปิดทางสัญญาณ Comparator Test ของมอดูลที่ SWNno
5. int TestCMP (int SWNno) เป็นฟังก์ชันสำหรับทดสอบการทำงานของวงจรเปรียบเทียบ ของมอดูลที่ SWNno
6. int CheckSWNStatus (int SWNno) เป็นฟังก์ชันสำหรับตรวจสอบสถานะการทำงานของมอดูลที่ SWNno
7. int CheckSWNPresent (int SWNno) เป็นฟังก์ชันสำหรับตรวจสอบดูว่ามีมอดูลที่ SWNno อยู่ในระบบหรือไม่
8. int AddSWN (int SWNno) เป็นฟังก์ชันสำหรับเรียกใช้เมื่อต้องการเพิ่มมอดูลที่ SWNno เข้าในระบบ
9. int RemoveSWN (int SWNno) เป็นฟังก์ชันสำหรับเรียกใช้เมื่อต้องการถอดมอดูลที่ SWNno ออกจากระบบ

ซอฟต์แวร์เพื่อการทดสอบระบบ

ในการออกแบบซอฟต์แวร์ส่วนนี้จะเน้นที่การทดสอบการทำงานของระบบ โดยมีเป้าประสงค์ที่จะทดสอบว่ามอดูลวงจรสวิตช์ที่ได้ออกแบบและสร้างขึ้นนั้นสามารถตรวจจับความผิดพลาดที่เกิดขึ้นได้หรือไม่ และสามารถที่ทนต่อความผิดพลาดที่เกิดขึ้นได้หรือไม่

กระบวนการทางซอฟต์แวร์ในส่วนนี้จะสอดคล้องกับทฤษฎีของการออกแบบระบบที่ทนทานต่อความผิดพลาดได้คือ

1. ตรวจจับข้อผิดพลาด (error detection) โดยการใช้วงจรเปรียบเทียบทางฮาร์ดแวร์ ผลลัพธ์จะถูกรับรู้ได้โดยกระบวนการทางซอฟต์แวร์ การตรวจจับข้อผิดพลาดของฮาร์ดแวร์ที่ได้ออกแบบขึ้นมีความสามารถทำได้ 3 อย่างคือ ตรวจสอบความผิดพลาดของวงจรสับเปลี่ยนช่องเวลาทั้งสองในมอดูลว่าทำงานตรงกันหรือไม่ โดยใช้วงจรเปรียบเทียบที่ออกแบบขึ้นเฉพาะ ตรวจสอบว่าวงจรเปรียบเทียบทำงานได้ถูกต้องหรือไม่ และสามารถตรวจสอบว่าสัญญาณนาฬิกาขาดหายไปหรือไม่

2. การบอกตำแหน่งของจุดที่เกิดข้อผิดพลาด (error location) เนื่องจากมีการออกแบบวงจรฮาร์ดแวร์สำหรับตรวจจับความผิดพลาดเพียง 3 อย่างดังที่ได้กล่าวในข้อที่ 1 ดังนั้นการบ่งบอกตำแหน่งที่เกิดความผิดพลาดจึงสามารถทำได้เพียง 3 ตำแหน่งเท่านั้น

3. การจำกัดขอบเขตของความผิดพลาด (error containment) ในงานวิจัยนี้จะออกแบบซอฟต์แวร์ให้ควบคุมให้มอดูลหยุดรับสัญญาณทันทีหลังจากที่รับทราบว่ามีข้อผิดพลาดเกิดขึ้นที่มอดูลนั้น โดยจะสั่งให้มอดูลสำรองที่มีอยู่ทำงานแทน

4. การกู้ระบบ (error recovery) เมื่อมีข้อผิดพลาดเกิดขึ้นซอฟต์แวร์ส่วนนี้จะทำหน้าที่ที่จะกู้ระบบให้คืนกลับมาทำงานได้ถูกต้องเหมือนเดิม ซึ่งในส่วนของซอฟต์แวร์ที่ได้ออกแบบไว้นี้จะหาว่ามีมอดูลสำรองอยู่ในระบบหรือไม่ ถ้ามีก็จะสั่งงานให้มอดูลสำรองทำงานเป็นมอดูลหลักที่เสียแทน ถ้าไม่มีมอดูลสำรองในระบบซอฟต์แวร์ก็จะทำการตั้งค่าเริ่มต้นในการทำงานให้มอดูลที่เสียเพื่อทดสอบการทำงานใหม่ ถ้ามอดูลยังไม่สามารถทำงานได้ถูกต้องอีกหลังจากการพยายามหลายครั้งแล้วก็แสดงว่ามอดูล บกพร่องจริง ก็จะมีการส่งสัญญาณไปบอกให้ไมโครโพรเซสเซอร์กลางรับทราบเพื่อที่จะดำเนินการต่อไป ในกรณีที่มีมอดูลสำรองอยู่ในระบบ หลังจากที่มอบหน้าที่การทำงานให้มอดูลสำรองทำงานแล้ว จะมีการตั้งค่าเริ่มต้นในการทำงานของมอดูลที่เสียใหม่เพื่อทดสอบการทำงาน เนื่องจากบางครั้งข้อผิดพลาดที่เกิดขึ้นเป็นข้อผิดพลาดแบบชั่วครู่ เมื่อเวลาผ่านไปสักระยะหนึ่งข้อผิดพลาดดังกล่าวก็จะไม่เกิดขึ้นอีก การหน่วงเวลาสักครู่หนึ่งแล้วให้มอดูลกลับมาทำงานใหม่จะสามารถทำงานได้ ซึ่งในกรณีนี้จะไม่ถือว่ามอดูลเสียหาย เว้นแต่ว่ามีการพยายามหลายครั้งแล้วยังไม่สำเร็จ ซึ่งก็หมายความว่ามอดูลเสียใช้การไม่ได้อีกต่อไป ต้องมีการถอดมอดูลออกจากระบบเพื่อทำการซ่อมแซม

ผังงานแสดงการทำงานของซอฟต์แวร์ และรหัสต้นฉบับของซอฟต์แวร์ที่พัฒนาด้วยภาษาซี ทั้งหมดได้แสดงไว้ในภาคผนวก ข.

บทที่ 5

ผลการวิจัย

ข้อกำหนดในการทดสอบระบบ

เนื่องจากข้อจำกัดในงบประมาณของการวิจัย ในการสร้างระบบต้นแบบเพื่อทำการทดสอบสามารถทำได้เพียง 2 มอดูลเท่านั้น ซึ่งเพียงพอสำหรับการทดสอบแนวความคิดได้ในระดับหนึ่ง โดยแต่ละมอดูลประกอบด้วยวงจรสับเปลี่ยนช่องเวลาขนาด 1,024 X 1,024 ช่อง วงจรเปรียบเทียบสัญญาณ วงจรเชื่อมต่อบัส และวงจรบัฟเฟอร์อินพุตเอาต์พุต

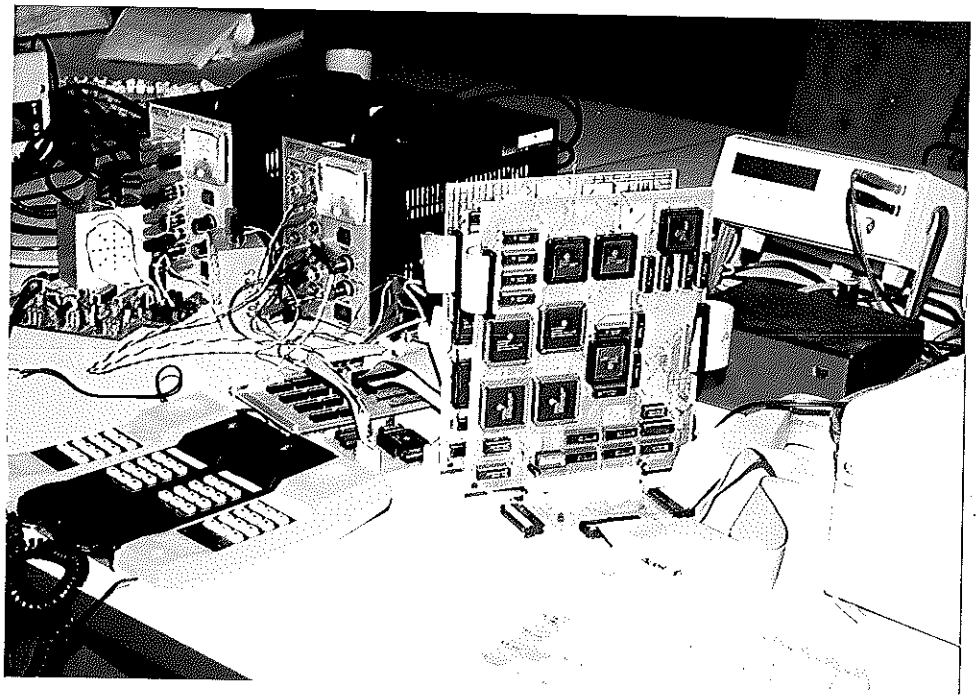
แผงหลังที่ใช้ได้ออกแบบขึ้นให้มีความสามารถที่จะเสียบมอดูลได้สูงสุดเพียง 4 มอดูลเท่านั้น ซึ่งความเป็นจริงแล้วจากการออกแบบสามารถทำได้ถึง 8 มอดูล แต่ในการทดสอบนี้มีเพียง 2 มอดูลเท่านั้น จึงไม่จำเป็นที่จะต้องสร้างให้มีความสามารถถึง 8 มอดูล

สัญญาณอินพุตที่จะป้อนเข้าสู่วงจรสวิตช์เพื่อทำการทดสอบการทำงานนั้นได้มาจากวงจร SLMA ซึ่งเป็นเครื่องต้นแบบที่ได้จากโครงงานนักศึกษาของภาควิชาวิศวกรรมคอมพิวเตอร์ (อะหมัด มามุ. 2537.) โดยเครื่องต้นแบบของวงจร SLMA ที่มีอยู่สามารถต่อเครื่องโทรศัพท์ได้เพียง 4 ตัวเพราะมีวงจร CODEC เพียง 4 วงจรเท่านั้น ภาพถ่ายแสดงวงจร SLMA ที่ใช้แสดงไว้ในภาคผนวก ก.

สัญญาณนาฬิกาที่ควบคุมการทำงานทั้งหมดของระบบจะนำมาจากวงจรสร้างสัญญาณนาฬิกาที่ได้กล่าวมาแล้วในเรื่องการออกแบบฮาร์ดแวร์ โดยสัญญาณนาฬิกาที่ใช้ในมอดูลวงจรสวิตช์ และที่ใช้ในวงจร SLMA นั้นนำมาจากแหล่งเดียวกันคือวงจรกำเนิดสัญญาณนาฬิกาเพื่อให้มีการทำงานเข้าจังหวะกัน

เครื่องคอมพิวเตอร์ส่วนบุคคลที่ใช้ในการทดสอบวงจรสวิตช์ในงานวิจัยนี้เป็นเครื่องคอมพิวเตอร์ส่วนบุคคลที่ใช้ซีพียูเบอร์ 80386 ทำงานที่ความถี่สัญญาณนาฬิกา 40 เมกกะเฮิร์ตซ์ ระบบปฏิบัติการที่ใช้คือไมโครซอฟต์ดอสเวอร์ชัน 6.20 และมีการ์ด 8255 เสียบอยู่เพื่อทำหน้าที่เชื่อมต่อกับระบบบัสที่มอดูล วงจรสวิตช์เสียบอยู่ ภาพประกอบ 30 แสดงภาพถ่ายของระบบที่ใช้ทดสอบทั้งหมด

ภาพประกอบ 31 แสดงการแสดงผลทางจอภาพของซอฟต์แวร์ที่ใช้ทดสอบการทำงานของวงจรสวิตช์ จากรูปจะเห็นว่ามีคำสั่งที่ใช้สั่งงานวงจรสวิตช์อยู่ 9 คำสั่ง ซึ่งเป็นการสั่งงานฟังก์ชันต่างๆ ดังที่ได้กล่าวในบทที่ 4 ส่วนบนของจอภาพจะแสดงสถานะการทำงานของมอดูลต่างๆ เช่นแสดงว่ามีมอดูลเสียบอยู่ที่แผงหลังช่องใดบ้าง มอดูลใดทำงานถูกต้องหรือไม่ถูกต้อง มอดูลใดที่กำลังชับเอาต์พุตอยู่ และแสดงจำนวนครั้งของการเกิดข้อผิดพร่องในมอดูลต่างๆ



ภาพประกอบ 30 ภาพถ่ายของระบบที่ทดสอบ

SWN No.	Present	Status	Output	Comparator	Clock Test	Failure Count
1	YES	WORK	ON	ON	NO	0
2	NO	FAIL	OFF	OFF	NO	0
3	YES	WORK	OFF	ON	NO	0
4	NO	FAIL	OFF	OFF	NO	0

1) Talk
2) Insert SWN
3) Remove SWN
4) Test Connection Memory
5) Test Comparator
6) Test Clock
7) Show Main Connection Memory
8) Show Connection Memory
9) Swap Active SWN
0) Quit

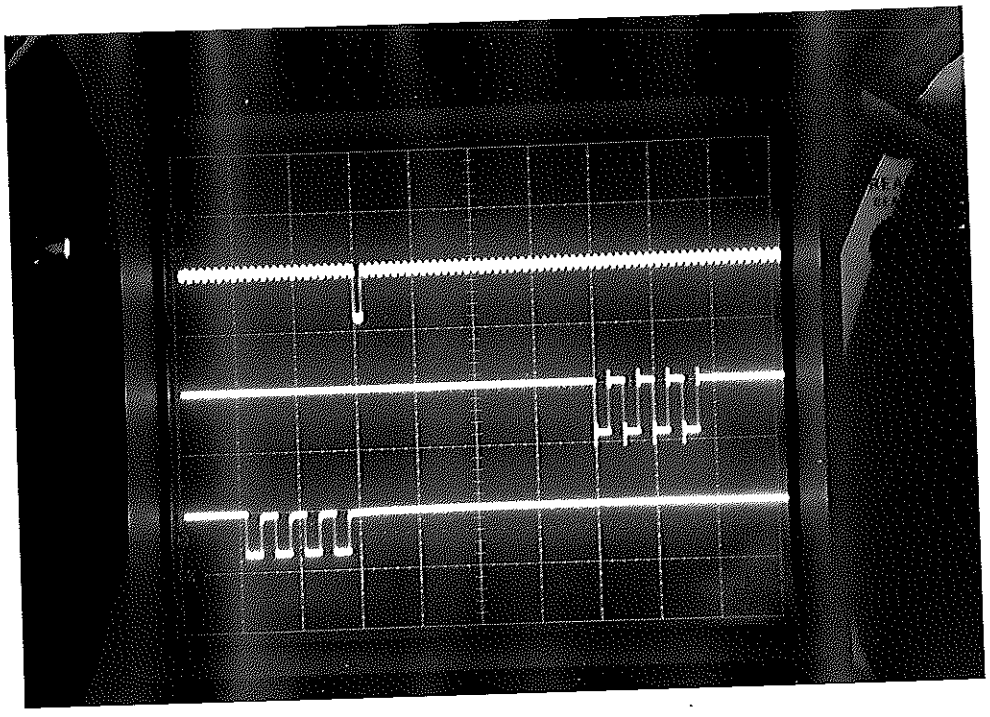
Command :•

ภาพประกอบ 31 การแสดงผลทางจอภาพของซอฟต์แวร์ที่ใช้ทดสอบ

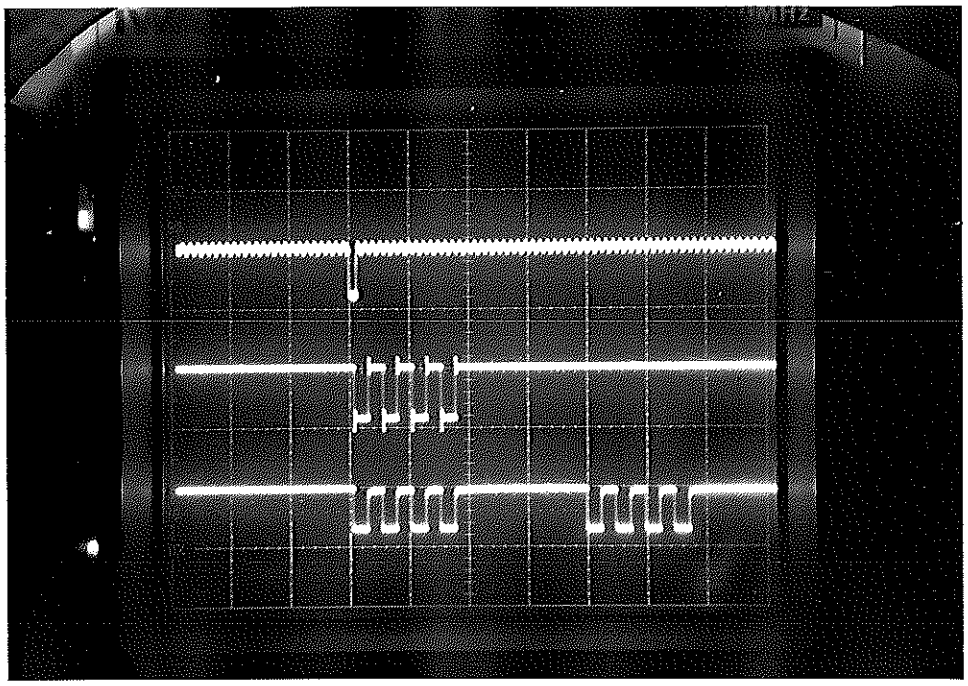
วิธีการทดสอบและผล

แนวความคิดหลักในการทดสอบระบบ คือการทดสอบว่าระบบสามารถทำงานตามหน้าที่พื้นฐานที่วงจรสวิตช์ควรจะมีได้ถูกต้องสมบูรณ์หรือไม่ และทดสอบว่าเมื่อมีข้อผิดพร่องเกิดขึ้นแล้วระบบยังจะสามารถทำงานได้อย่างถูกต้องหรือไม่ โดยมีการทดสอบดังนี้

1. ทดสอบการทำงานเป็นวงจรสวิตช์ โดยการสร้างสัญญาณเทดสอบขึ้นด้วยการควบคุมสัญญาณ ME ให้เป็นลอจิกสูงแล้วส่งสัญญาณต่างๆ เพื่อการทดสอบออกไป จากเอาต์พุตของวงจรสวิตช์ก็นำสัญญาณนั้นส่งกลับเข้ามาที่กระแสอินพุตที่ต้องการทดสอบ แล้วจึงควบคุมการทำงานของวงจรสวิตช์ให้สวิตช์สัญญาณที่อินพุตให้ออกไปยังเอาต์พุตในหมายเลขช่อง และหมายเลขกระแสที่ต้องการ มีการสังเกตุผลจากออสซิลโลสโคป ได้ผลดังภาพต่อไปนี้



ภาพประกอบ 32 แสดงสัญญาณจากการสวิตช์จากอินพุตช่องที่ 2 กระแสที่ 0
ไปยังเอาต์พุตช่องที่ 31 ของ กระแสที่ 1



ภาพประกอบ 33 แสดงสัญญาณจากการสวิตช์จากอินพุตช่องที่ 0 กระแสที่ 0
ไปยังเอาต์พุตช่องที่ 0 และ 2 ของ กระแสที่ 31

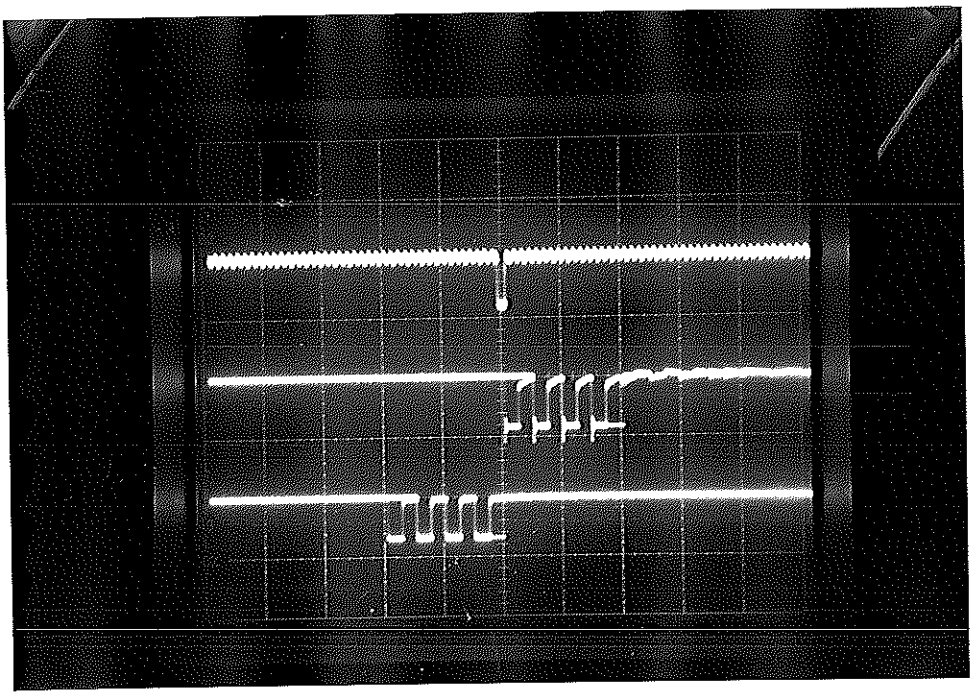
จากภาพประกอบ 32 และ 33 สัญญาณเส้นบนสุดเป็นสัญญาณ F0 เส้นกลางเป็นสัญญาณอินพุต และเส้นล่างเป็นสัญญาณเอาต์พุต

2. ทดสอบการสวิตช์สัญญาณเสียงที่อยู่ในรูป PCM จากวงจร SLMA โดยให้โทรศัพท์ทั้ง 4 เครื่องต่อกับวงจรสวิตช์ดังตาราง 3

ตาราง 3 การต่อเครื่องโทรศัพท์กับวงจรสวิตช์

เครื่องที่	Stream ที่	ช่องที่	ตำแหน่งแอดเดรสของ Connection Memory
1	0	0	0
2	1	1	33
3	2	2	66
4	3	3	99

ผลการทดสอบทุกเครื่องสามารถถูกกำหนดให้พูดคุยกันได้



ภาพประกอบ 34 แสดงการสวิตช์สัญญาณ PCM ของเสียงจากโทรศัพท์เครื่องที่ 1 ไปยังเอาต์พุตช่องที่ 31 ของ กระแสที่ 9

3. ทดสอบการทำงานของวงจรเปรียบเทียบ โดยขณะที่มอดูลกำลังทำงานอยู่ได้มีการต่อสัญญาณที่อินพุตของวงจรเปรียบเทียบกับแรงดัน 0 โวลต์เป็นเวลาสั้นๆ และทดลองเปลี่ยนอินพุตไปเรื่อยๆ ปรากฏว่าวงจรเปรียบเทียบสามารถแสดงข้อผิดพลาดออกมาได้ในทุกครั้ง

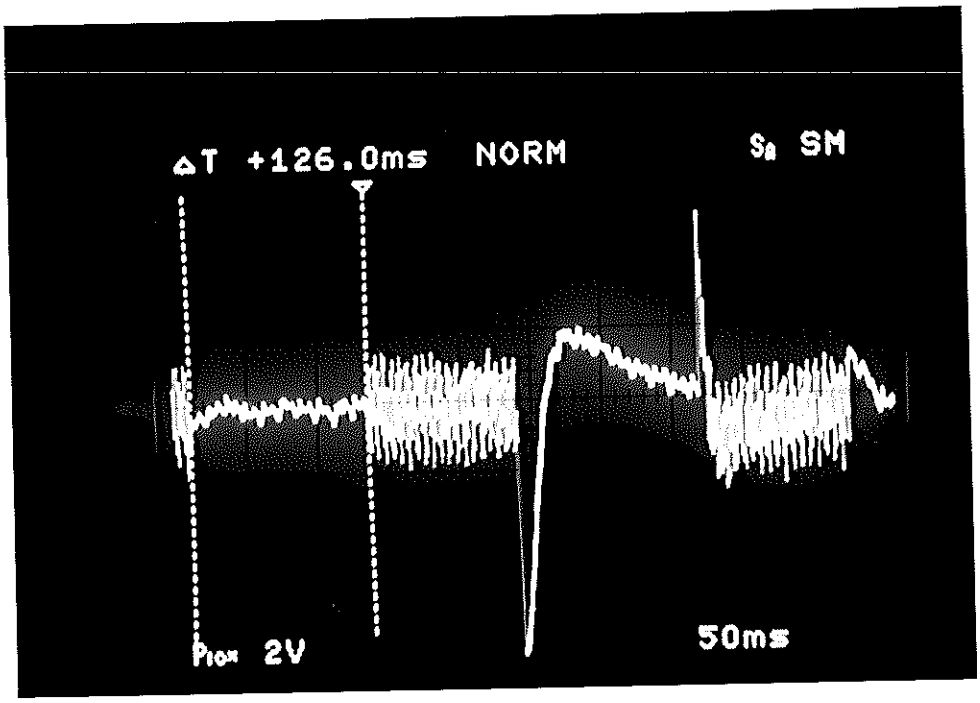
4. ทดสอบการทำงานของซอฟต์แวร์ซึ่งทำหน้าที่ควบคุมมอดูลที่เสียไม่ให้ทำงาน และสั่งงานให้มอดูลสำรองทำงานแทน โดยการจำลองการเกิดข้อผิดพร่องเช่นเดียวกันที่ทำในข้อ 3 แล้วดูว่ามอดูลที่สำรองอยู่จะถูกเรียกขึ้นใช้งานหรือไม่ ผลการทดลองคือมอดูลสำรองสามารถทำงานแทนได้โดยสัญญาณเสียงจากหูฟังโทรศัพท์ไม่ขาดหายไปจนรู้สึกได้ และเมื่อจับสัญญาณรบกวนที่เกิดขึ้นจากออสซิลโลสโคป ไม่สามารถจับสัญญาณรบกวนได้

5. ทดสอบการถอดออกและเสียบมอดูลเข้าโดยใช้ซอฟต์แวร์ที่สร้างขึ้น ผลการทดลองคือกรณีที่ถอดมอดูลหนึ่งออก มอดูลที่เหลือสามารถทำงานแทนต่อไปได้ และเมื่อเสียบมอดูลเข้าไป ซอฟต์แวร์ก็สามารถที่จะ

กำหนดค่าเริ่มต้นในการทำงานให้และสามารถทำงานต่อไปได้โดยไม่รู้สึกว่าข้อมูลเสียงจากหูฟังโทรศัพท์ขาดหายไป

6. การทดสอบจำนวนครั้งของการทำงานผิดพร่องของมอดูลในช่วงเวลาหนึ่ง โดยให้เครื่องทำงานติดต่อกันเป็นเวลา 24 ชั่วโมงไม่ปรากฏว่ามีความผิดพร่องเกิดขึ้น

7. การทดสอบสัญญาณรบกวนในขณะที่ทำการตั้งค่าเริ่มต้นของมอดูลเมื่อมอดูลทำงานบกพร่อง ในขณะที่มีการใช้งานเพียงมอดูลเดียวในระบบ สามารถวัดสัญญาณรบกวนที่เกิดขึ้นได้ดังภาพประกอบ 35 จากภาพสัญญาณความถี่สูงที่มีขนาดประมาณ 0.2 โวลต์จากยอดถึงยอด คือสัญญาณ DTMF ที่ส่งผ่านคู่สายโทรศัพท์ในภาวะการทำงานปกติ ส่วนสัญญาณยอดแหลมๆ เป็นสัญญาณรบกวนที่เกิดขึ้นในขณะที่ทำการตั้งค่าเริ่มต้นในการทำงานของมอดูล



ภาพประกอบ 35 สัญญาณรบกวนที่เกิดขึ้นขณะตั้งค่าเริ่มต้นของมอดูล

8. ทดลองสร้างข้อผิดพร่องชนิดถาวรให้กับมอดูล ผลการทดลองเมื่อทดลองลัดวงจรบางส่วนในมอดูลเป็นเวลานานๆ ซอฟต์แวร์จะตอบสนองโดยการพยายามกู้ระบบคืน แต่ไม่สามารถกู้ได้ จึงแสดงข้อความออกมาว่ามอดูลดังกล่าวไม่สามารถทำงานได้

จากการทดลองและการวัดค่าคุณสมบัติในการทำงานของมอดูลวงจรสวิตช์ได้ดังนี้

1. สามารถสวิตช์สัญญาณได้ 1,024 ช่องพร้อมกัน
2. สามารถตรวจจับข้อผิดพร่องได้และแสดงผลทางฮาร์ดแวร์และซอฟต์แวร์
3. สามารถทำงานต่อไปได้เมื่อมีข้อบกพร่องเกิดขึ้นที่บางส่วนของมอดูล
4. ควบคุมการทำงานด้วยซอฟต์แวร์
5. สามารถเสียบมอดูลเข้าหรือถอดมอดูลออกจากระบบขณะระบบกำลังทำงานอยู่โดยไม่ทำให้การทำงานของระบบผิดพลาด และไม่ทำให้ระบบเสียหาย
6. ในกรณีที่ข้อผิดพร่องที่เกิดขึ้นเป็นแบบชั่วคราว จะใช้เวลาในการกู้ระบบคืน 38 ไมโครวินาที เมื่อมีมอดูลสำรอง และ 0.102 วินาทีเมื่อไม่มีมอดูลสำรอง
7. เมื่อมีมอดูลสำรอง ไม่สามารถวัดสัญญาณรบกวนได้ขณะทำการกู้ระบบ ในกรณีที่ไม่มีมอดูลสำรองสัญญาณจะขาดหายไปไม่เกิน 126 มิลลิวินาที (ดูภาพประกอบ 35)
8. ขนาดมอดูลคือ 7.2 X 8.8 นิ้ว และกินกระแส 1.1 แอมแปร์ ที่ 5 โวลต์
9. สามารถขยายจำนวนช่องสัญญาณได้โดยการนำเอาหลายๆ มอดูลมาต่อกัน ซึ่งจากการออกแบบสามารถขยายได้สูงสุด 2,048 X 2,048 ช่องแบบทนทานต่อความผิดพร่องได้

สรุปผลและข้อเสนอแนะ

จากคุณสมบัติของวงจรสวิตช์ที่ต้องการออกแบบที่กำหนดไว้ และจากผลการวิจัยพบว่า ได้วงจรสวิตช์ที่มีคุณสมบัติใกล้เคียงกับที่ต้องการมาก จะแตกต่างเพียงเล็กน้อยในส่วนของอัตราสิ้นเปลืองกำลังไฟ และเวลาที่ใช้ในการกู้ระบบคืน ซึ่งมีค่าเกินกว่าที่ต้องการน้อยมาก และสามารถออกแบบให้มีคุณสมบัติที่ดีกว่าที่กำหนดไว้ได้

วงจรสวิตช์ที่ถูกสร้างโดยใช้วิธี DCSS สามารถที่จะมีความทนทานต่อความผิดพร่องได้อยู่ระดับหนึ่ง ซึ่งขึ้นอยู่กับจำนวนมอดูลสำรองที่มีอยู่ ถ้าจำนวนมอดูลสำรองมีอยู่มากระดับของความทนทานต่อความผิดพร่องได้ก็จะสูงขึ้น และถึงแม้ว่าระบบจะมีเพียงมอดูลเดียวระบบก็ยังมีความสามารถในการตรวจสอบความถูกต้องในการทำงานของตัวเองได้ กอรปกับกระบวนการทางซอฟต์แวร์ ซึ่งสามารถที่จะกู้ระบบกลับคืนได้ในกรณีที่ความผิดพร่องที่เกิดขึ้นเป็นความผิดพร่องแบบชั่วคราว ดังนั้นอาจกล่าวได้ว่ามอดูลของวงจรสวิตช์ที่ได้ออกแบบขึ้นนี้มีระดับความทนทานต่อความผิดพร่องอยู่ระดับต่ำๆ ระดับหนึ่งเมื่อมีการใช้งานเพียงมอดูลเดียว

ความสามารถในการทนทานต่อความผิดพร่องของระบบ จะมีความสัมพันธ์อย่างยิ่งกับความสามารถในการตรวจจับความผิดพร่องที่เกิดขึ้น ระบบจะมีระดับความสามารถในการทนทานต่อความผิด

พร่องสูง ถ้าเปอร์เซนต์ของการที่จะตรวจพบความผิดพร่องในระบบมีอยู่สูง แต่การให้ได้มาของเป้าหมายดังกล่าวจะมีผลทำให้ระบบมีขนาดใหญ่และซับซ้อนขึ้น รวมถึงค่าใช้จ่ายต่างๆ ที่จะสูงขึ้นด้วย ในงานวิจัยนี้ได้เน้นที่การออกแบบระบบที่ไม่ซับซ้อนจนเกินไป

ในการออกแบบมอดูลวงจรสวิตช์ในงานวิจัยนี้นั้นยังมีจุดบกพร่องอยู่หลายประการเช่น

1. วงจรเปรียบเทียบไม่สามารถทำงานได้อย่างถูกต้อง เมื่อสัญญาณอินพุตที่เข้ามาในมอดูลมีความผิดพลาด เพราะในกรณีที่วงจรสับเปลี่ยน ช่องเวลา ทำงานได้ถูกต้อง และข้อมูลที่เข้าผิดพลาดผลลัพธ์จากการสวิตช์ที่ได้ก็จะผิดพลาดเหมือนกัน ดังนั้นวงจรเปรียบเทียบจึงไม่สามารถทำงานได้อย่างถูกต้อง

2. ไม่มีคุณสมบัติในการบอกตำแหน่งที่เกิดข้อบกพร่องได้ละเอียดเพียงพอ

3. ไม่มีการออกแบบเพื่อให้มีการทดสอบ (design for testablility) ได้มากเท่าที่ควร

4. ไม่ได้เชื่อมต่อกับระบบบัสของวงจรไมโครโปรเซสเซอร์ที่ใช้ควบคุมจริง

ข้อเสนอแนะในการพัฒนาระบบ

เนื่องจากมีข้อจำกัดในหลายๆ ด้านในการทำงานวิจัยครั้งนี้ จึงมีผลทำให้ผลการวิจัยอาจจะไม่ดีเท่าที่ควรหรือไม่สามารถนำไปประยุกต์ใช้งานได้อย่างจริงจัง ผู้วิจัยจึงขอเสนอแนะสำหรับผู้ที่ต้องการพัฒนาวงจรนี้ต่อไปดังนี้

1. ควรมีการออกแบบสถาปัตยกรรมของระบบทั้งหมดโดยรวมก่อนนั่นคือต้องออกแบบระบบทั้งหมดให้มีความทนทานต่อความผิดพร่องได้ เนื่องจากในการวิจัยครั้งนี้ได้มุ่งเน้นการออกแบบวงจรสวิตช์ที่มีความสามารถทนทานต่อความผิดพร่องได้เพียงส่วนเดียว ทำให้ระดับความน่าเชื่อถือไม่สูงเท่าที่ควร เช่นควรจะออกแบบให้สัญญาณต่างๆ มีอย่างน้อย 2 สัญญาณที่เหมือนกันโดยเฉพาะสัญญาณที่มีความสำคัญมากๆ เช่นสัญญาณนาฬิกาของระบบ สัญญาณอินพุตและเอาต์พุตจากวงจร SLMA ซึ่งถ้าวงจร SLMA ให้สัญญาณออกมา 2 ชุดที่เหมือนกันแล้วส่งมาให้วงจรสับเปลี่ยน ช่องเวลา ที่มีอยู่ 2 วงจรใน 1 มอดูลแล้ว ระดับความสามารถในการตรวจสอบความผิดพร่องของวงจรสวิตช์และของระบบก็จะสูงขึ้น

2. ควรมีการใส่ความสามารถในการตรวจสอบความถูกต้องในการทำงานของวงจรให้มากกว่านี้ โดยเฉพาะส่วนที่ติดต่อกับไมโครโปรเซสเซอร์ เพราะถ้าส่วนนี้ทำงานผิดพลาดการควบคุมการทำงานของสวิตช์ก็จะไม่ถูกต้อง

3. ควรใส่วงจรป้องกันมากขึ้น เช่นวงจรป้องกันแรงดันเกิน เป็นต้น

4. ควรออกแบบแผ่นวงจรพิมพ์ให้มีขนาดตามมาตรฐาน ควรออกแบบให้สัญญาณอินพุตและเอาต์พุตทั้ง 64 เส้นเชื่อมต่อกับมอดูลโดยใช้สล็อตแบบเสียบเช่นดียวกันกับการเชื่อมต่อระบบบัส โดยอาจใช้ขนาดแผ่นวงจรพิมพ์และขนาดขั้วต่อสัญญาณตามมาตรฐานของดับเบิ้ลยูโรการ์ด (นรินทร์ ว่องพงศาวิวัฒน์. 2533.) ก็ได้

บรรณานุกรม

คณะกรรมการบัญญัติศัพท์คอมพิวเตอร์ ราชบัณฑิตยสถาน. 2533. ศัพท์บัญญัติคอมพิวเตอร์ (ฉบับร่าง). กรุงเทพฯ : .เอ. อาร์. อินฟอร์เมชัน แอนด์ พับลิเคชัน.

ธวัชชัย เลื่อนฉวี. 2533. เทคโนโลยีโทรศัพท์. กรุงเทพฯ : บรรเทิงการพิมพ์, กรุงเทพฯ.

นรินทร์ ว่องพงศาวิวัฒน์. 2533. "VME บัสระบบบัสมาตรฐานอุตสาหกรรม", เซมิคอนดัคเตอร์อิเล็คทรอนิคส์. กรุงเทพฯ : บริษัท ซีเอ็ดยูเคชัน จำกัด

วีระพันธุ์ มุสิกสาร และคณะ. 2535. การพัฒนาชุมสายโทรศัพท์ระบบดิจิตอล TDSS-1R เพื่อใช้เป็นชุมสายชนบท. การประชุมวิชาการครั้งที่ 4 ศูนย์อิเล็กทรอนิกส์และคอมพิวเตอร์แห่งชาติ สำนักงานพัฒนาวิทยาศาสตร์และเทคโนโลยีแห่งชาติ กระทรวงวิทยาศาสตร์ เทคโน โลยีและสิ่งแวดล้อม.

สงขลานครินทร์, มหาวิทยาลัย. บัณฑิตวิทยาลัย. 2536. คู่มือการเขียนวิทยานิพนธ์.

อะหมัด มามุ. 2537. โมดูลอินเตอร์เฟสสำหรับผู้ใช้คู่สายอนาลอก (8 คู่สาย). โครงงานนักศึกษาภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยสงขลานครินทร์

ทีมงานอีทีที. 2535. My First PAL Design. กรุงเทพฯ : บริษัท อีทีที จำกัด.

ทีมงานอีทีที. 2535. ETT-PC8255. กรุงเทพฯ : บริษัท อีทีที จำกัด.

Avizienis, A. 1978. "Fault-Tolerance: The Survival Attribute of Digital Systems", Proceeding of the IEEE., Vol. 66, No. 10, October 1978.

Johnson, B.W. 1989. Design and Analysis of Fault Tolerant Digital Systems, U.S.A. : Addison-Wesley Publishing Company, Inc.

Lala, P.K. 1985. Fault Tolerant & Fault Testable Hardware Design. USA : Prentice-Hall International, Inc., London.

บรรณานุกรม (ต่อ)

Mitel Corporation. 1991. Microelectronics DIGITAL Communications Handbook, Canada : MITEL Corporation.

Neufang, K. 1981. "The EWSD Digital Switching Network". Germany : telcom report special issue "EWSD Digital Switching System".

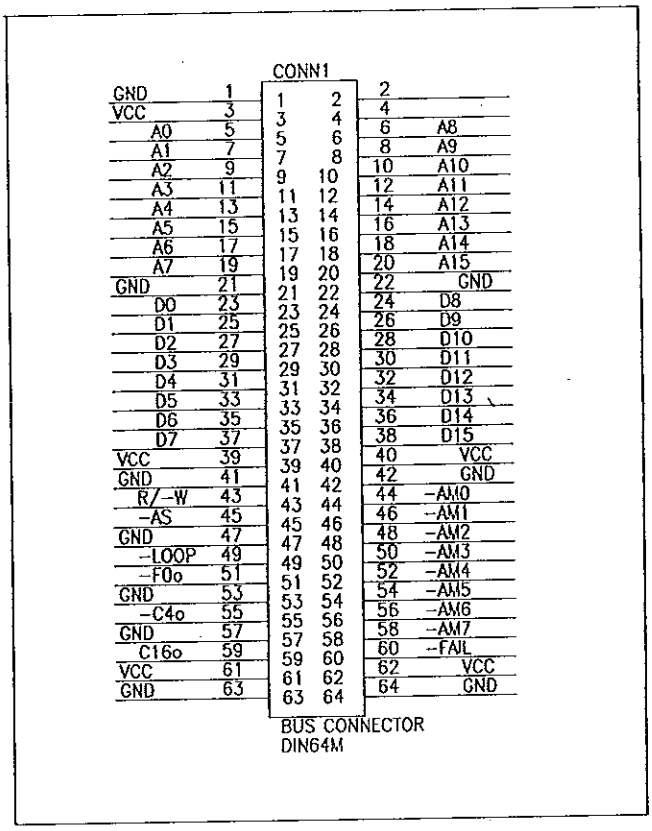
Siewiorek, D.P. 1991, "Architecture of Fault-Tolerant Computers: An Historical Perspective ", Proceedings of the IEEE,, Vol. 79, No. 12, December 1991.

Suckfull, H. 1979. "Architecture of a New Line of Digital Public Telephone Exchange"., telcom report., Vol. 2 No. 4.

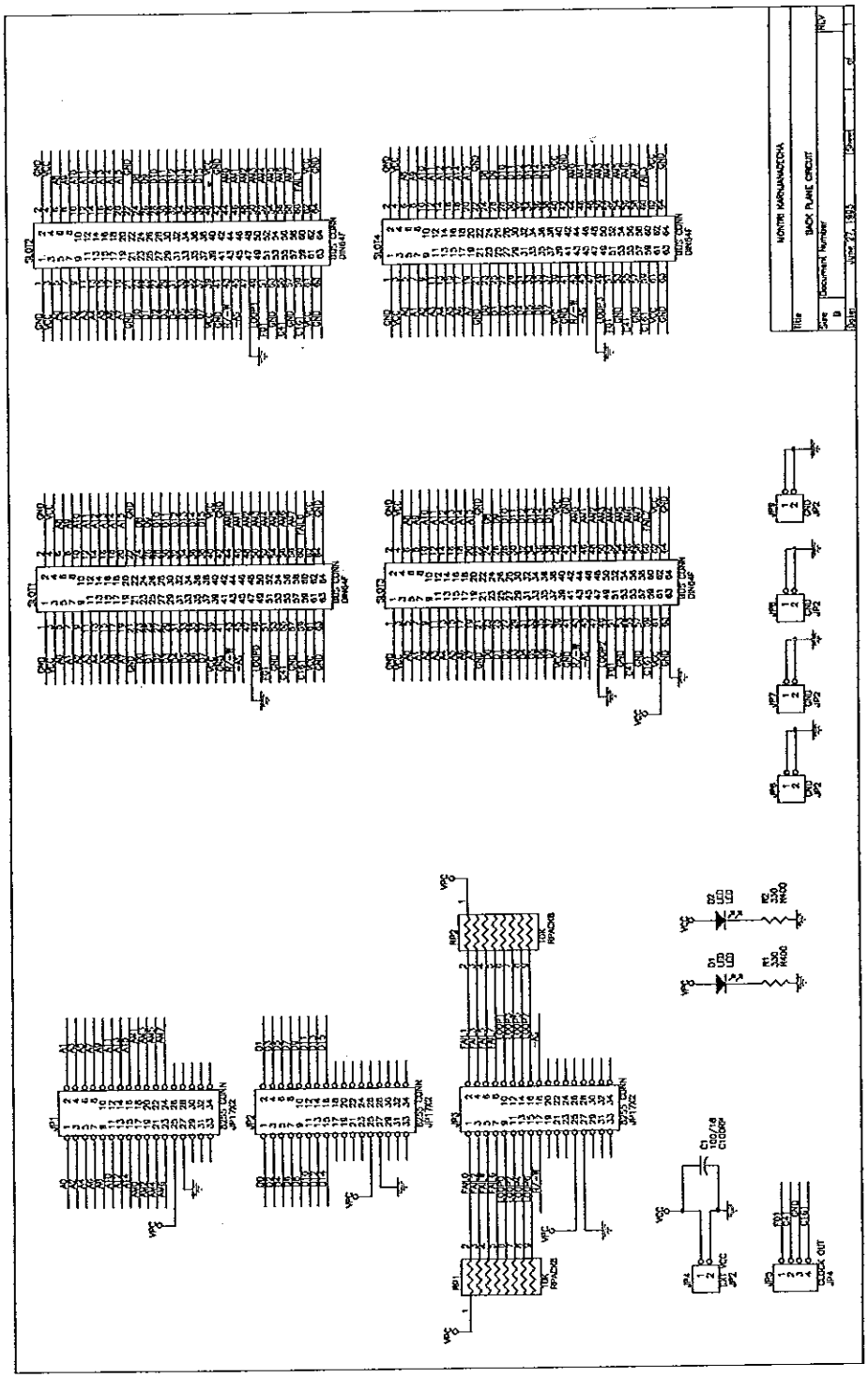
Toy, W.N. 1978. "Fault-Tolerance Design of Local ESS Processors", Proceeding of the IEEE., Vol. 66, No. 10, October 1978.

ภาคผนวก ก.

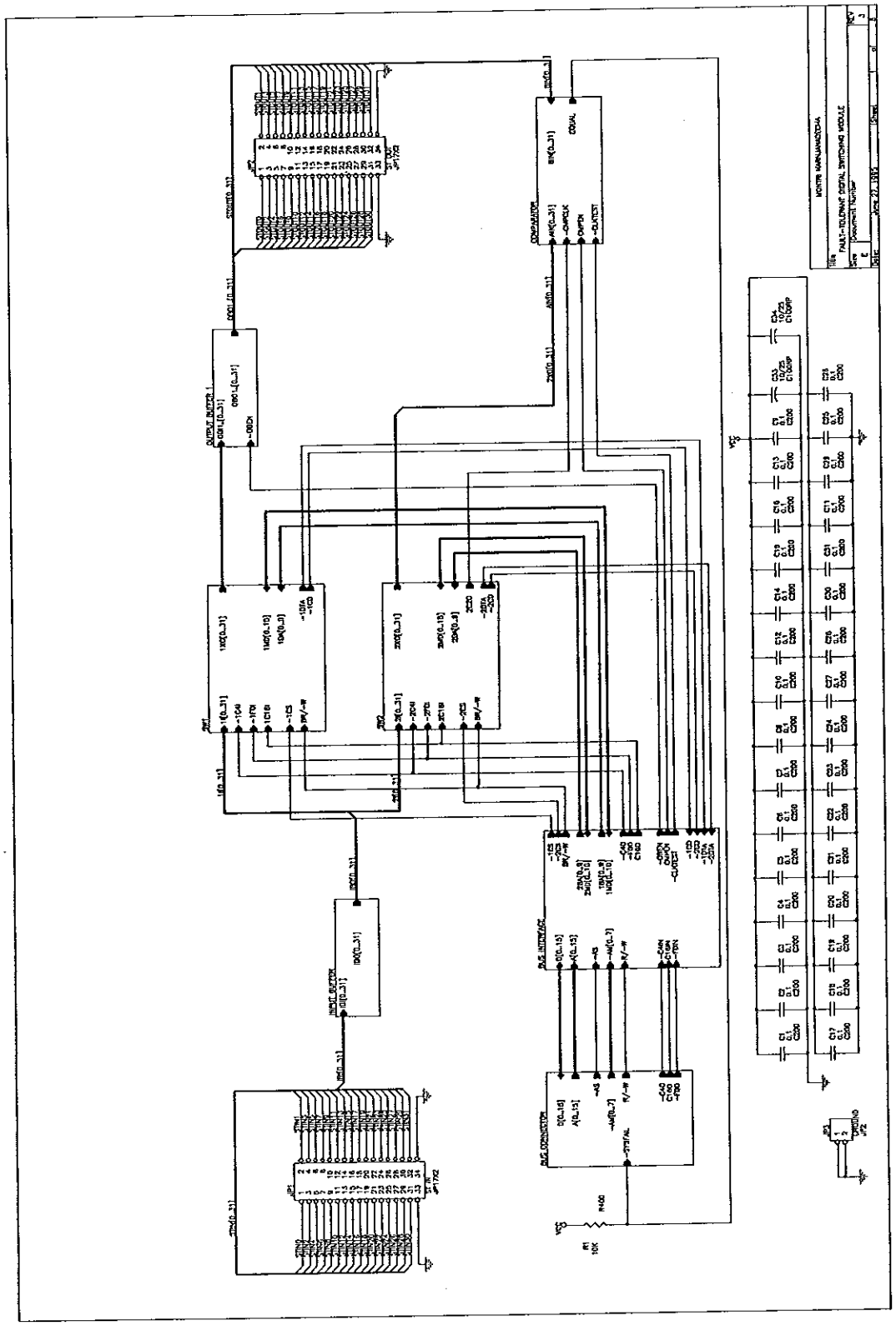
วงจรและภาพถ่าย



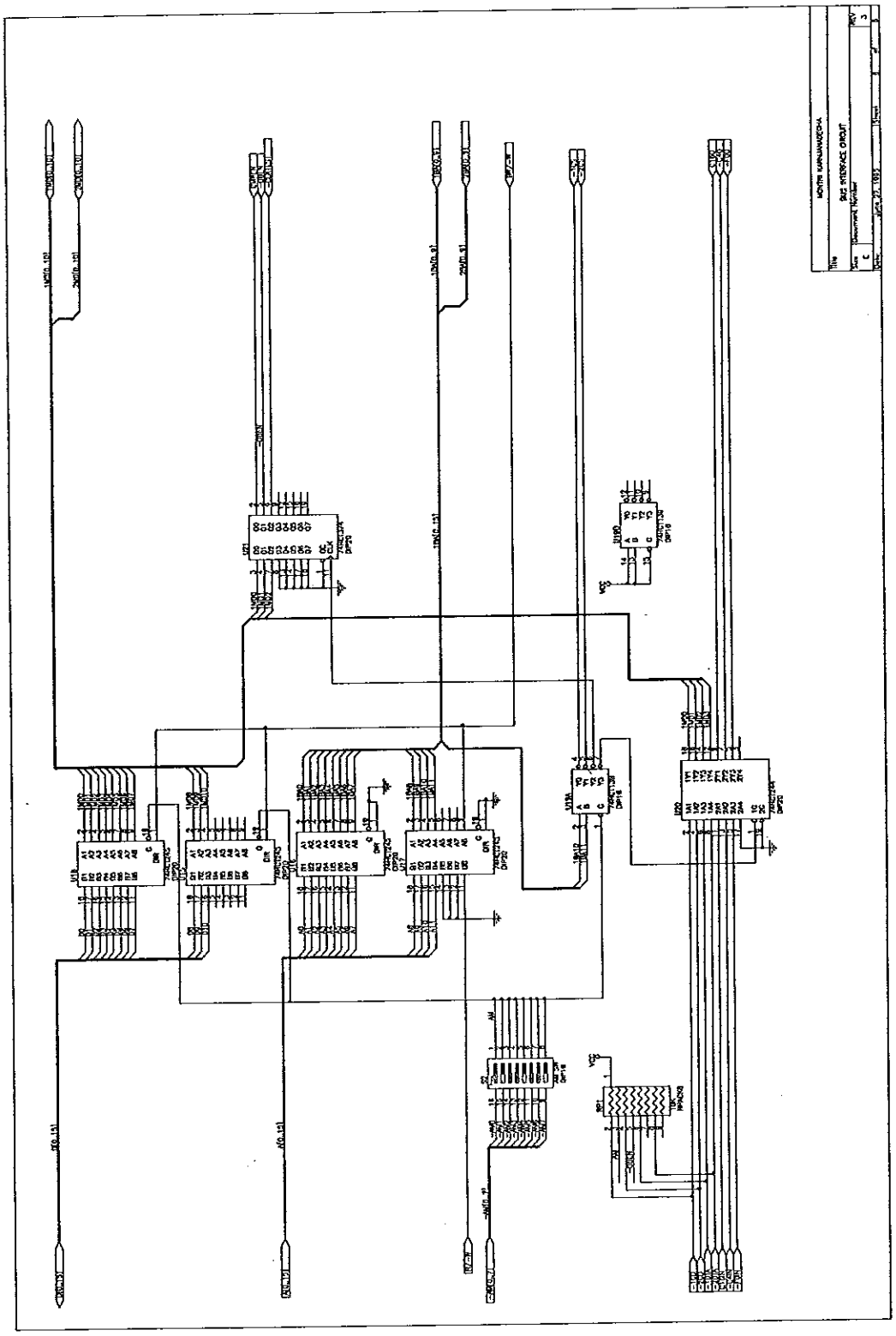
ภาพประกอบ 36 การจัดสัญญาณต่างๆ บนบัส



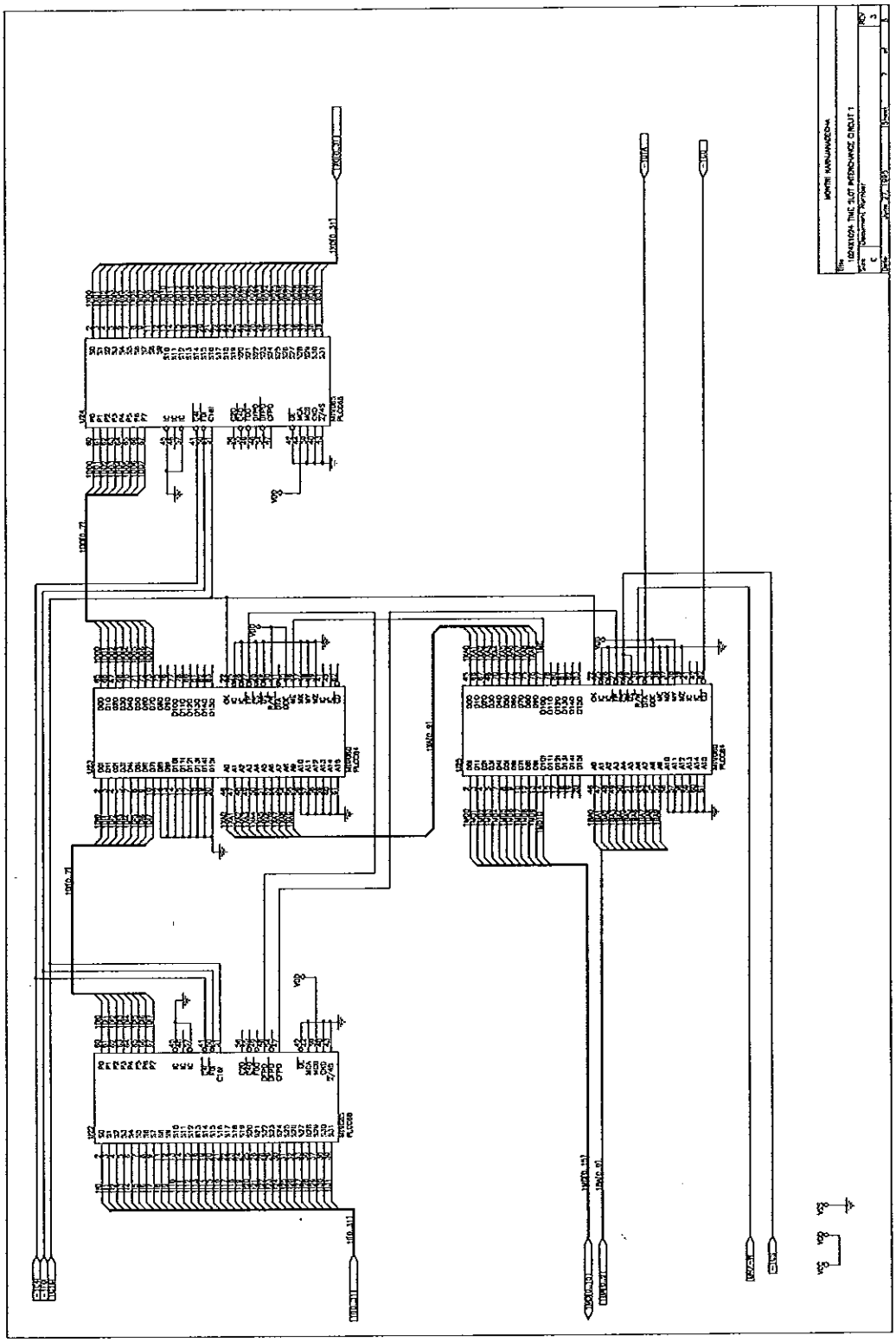
ภาพประกอบ 37 วงจรแผงหลัง



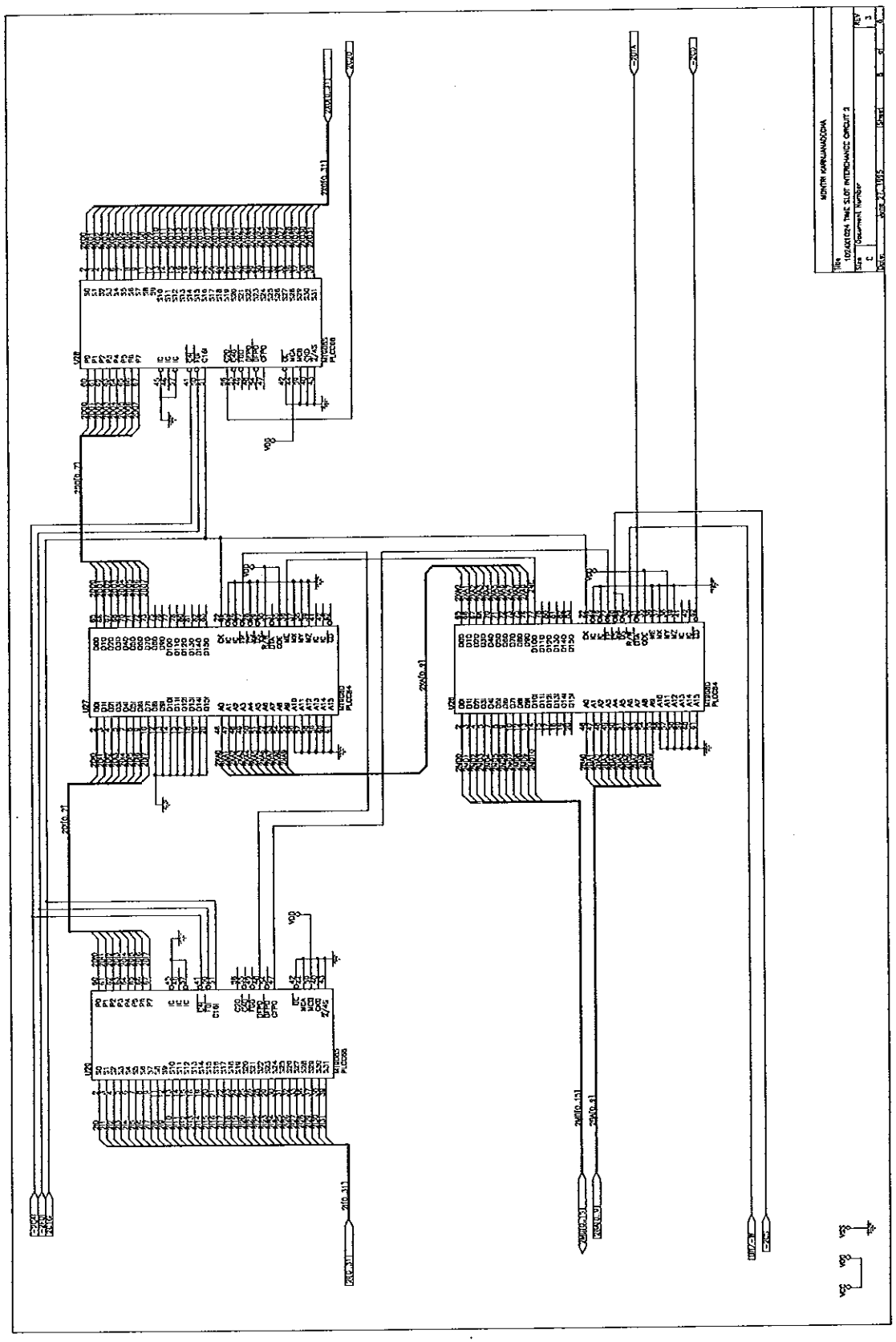
ภาพประกอบ 38 วงจรมอดูลวงจรสวิตช์



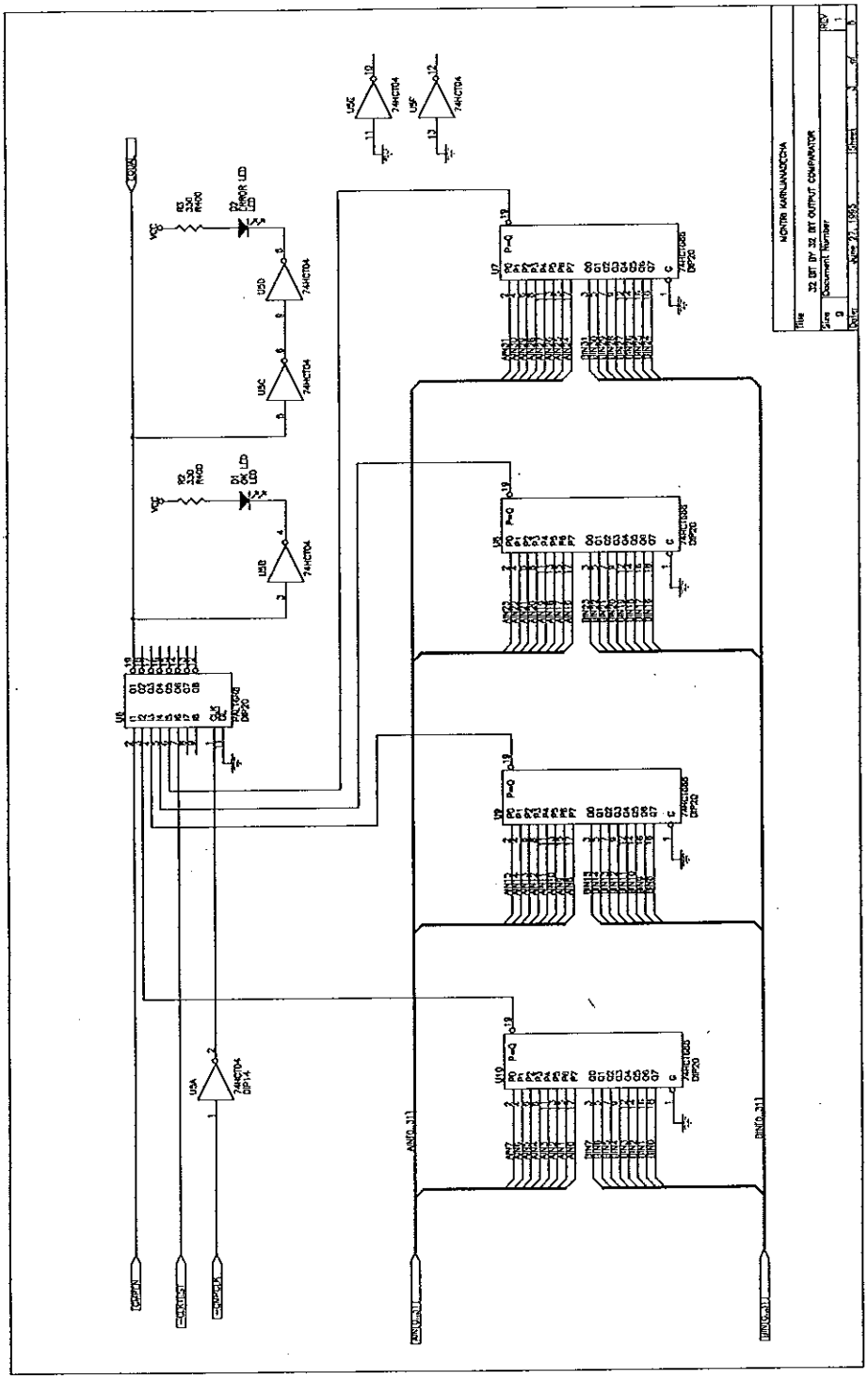
ภาพประกอบ 39 วงจรเชื่อมต่อบัส



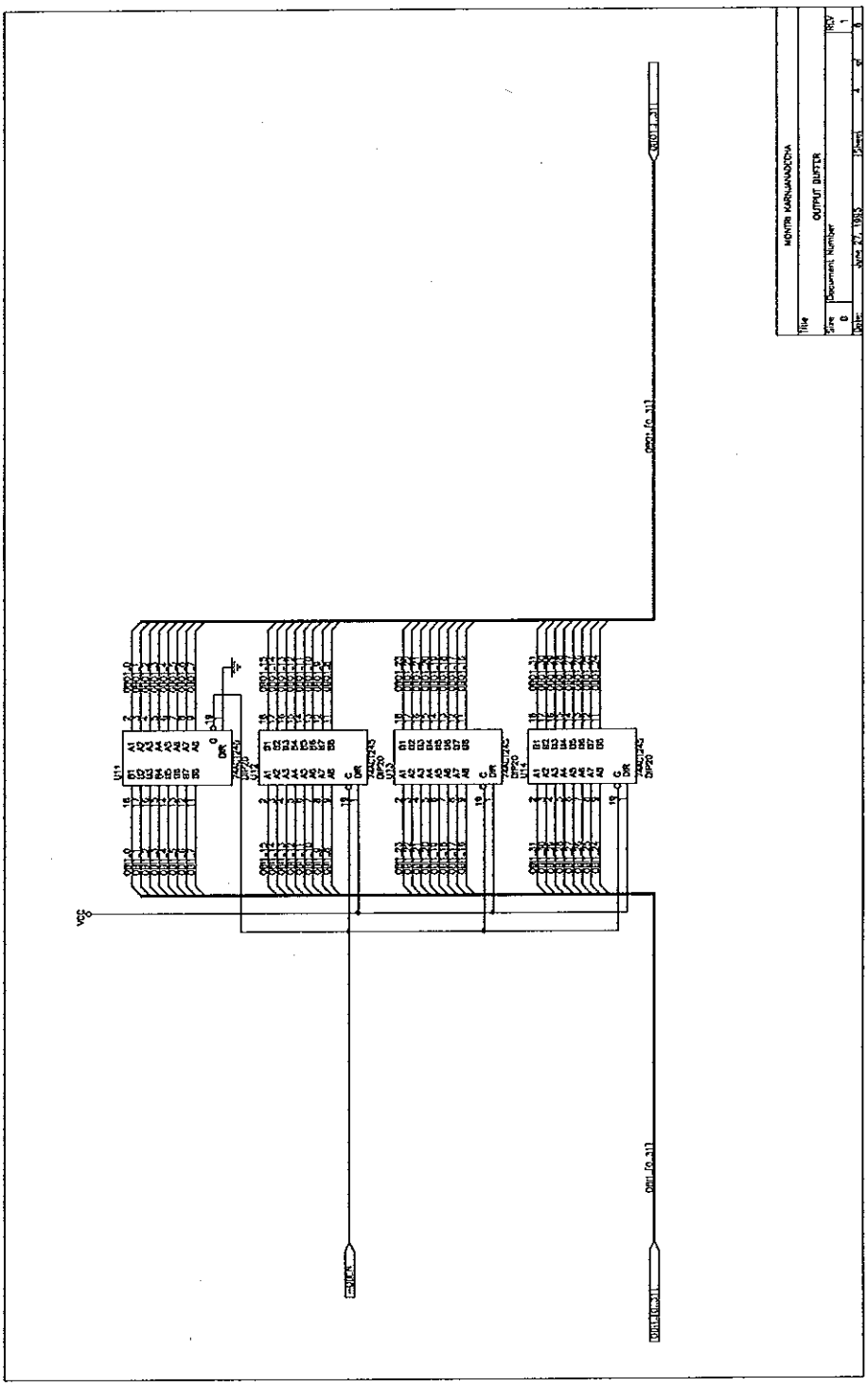
ภาพประกอบ 40 วงจรสับเปลี่ยนช่องเวลาวงจรที่ 1



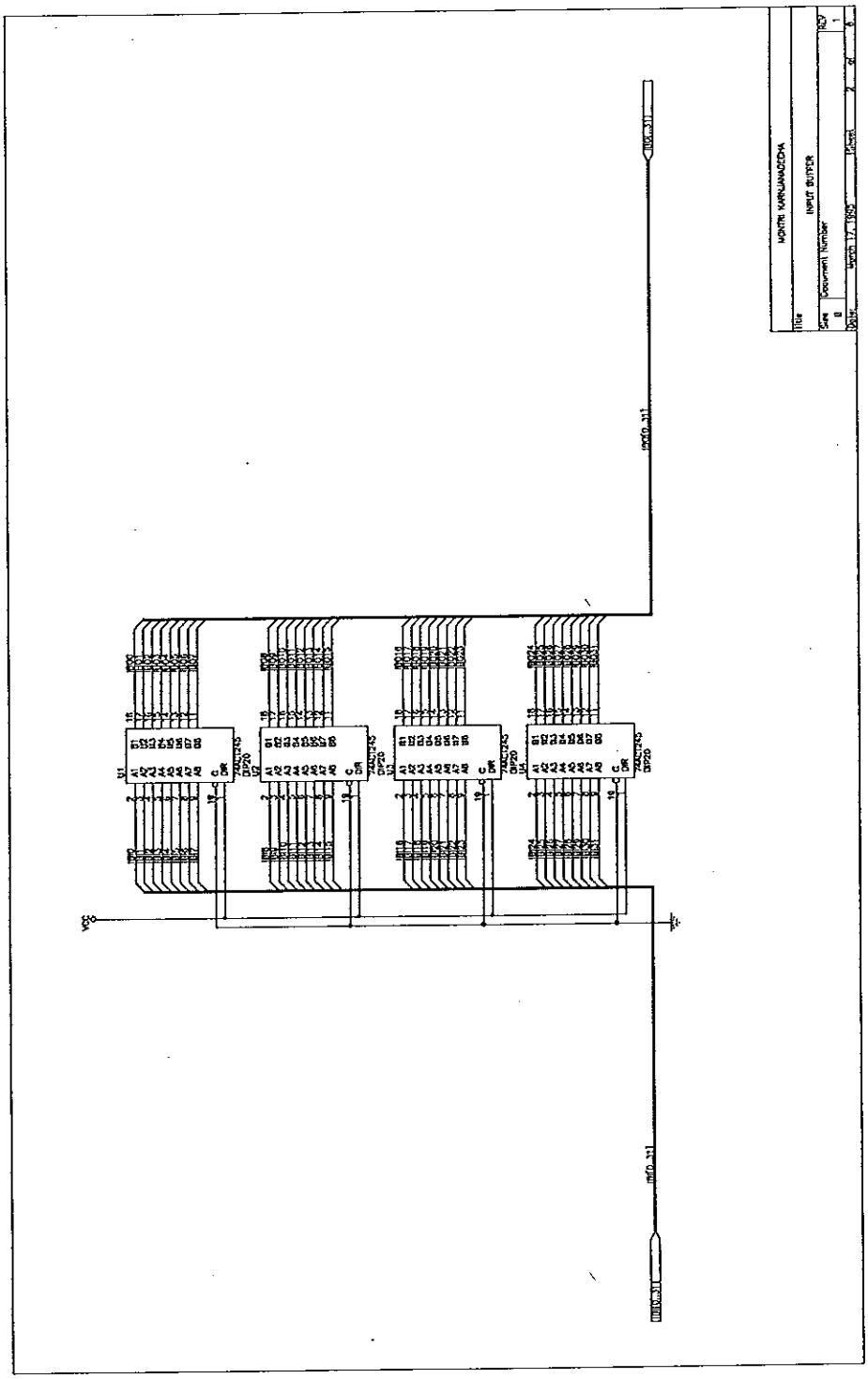
ภาพประกอบ 41 วงจรสับเปลี่ยนช่องเวลาวงจรที่ 2



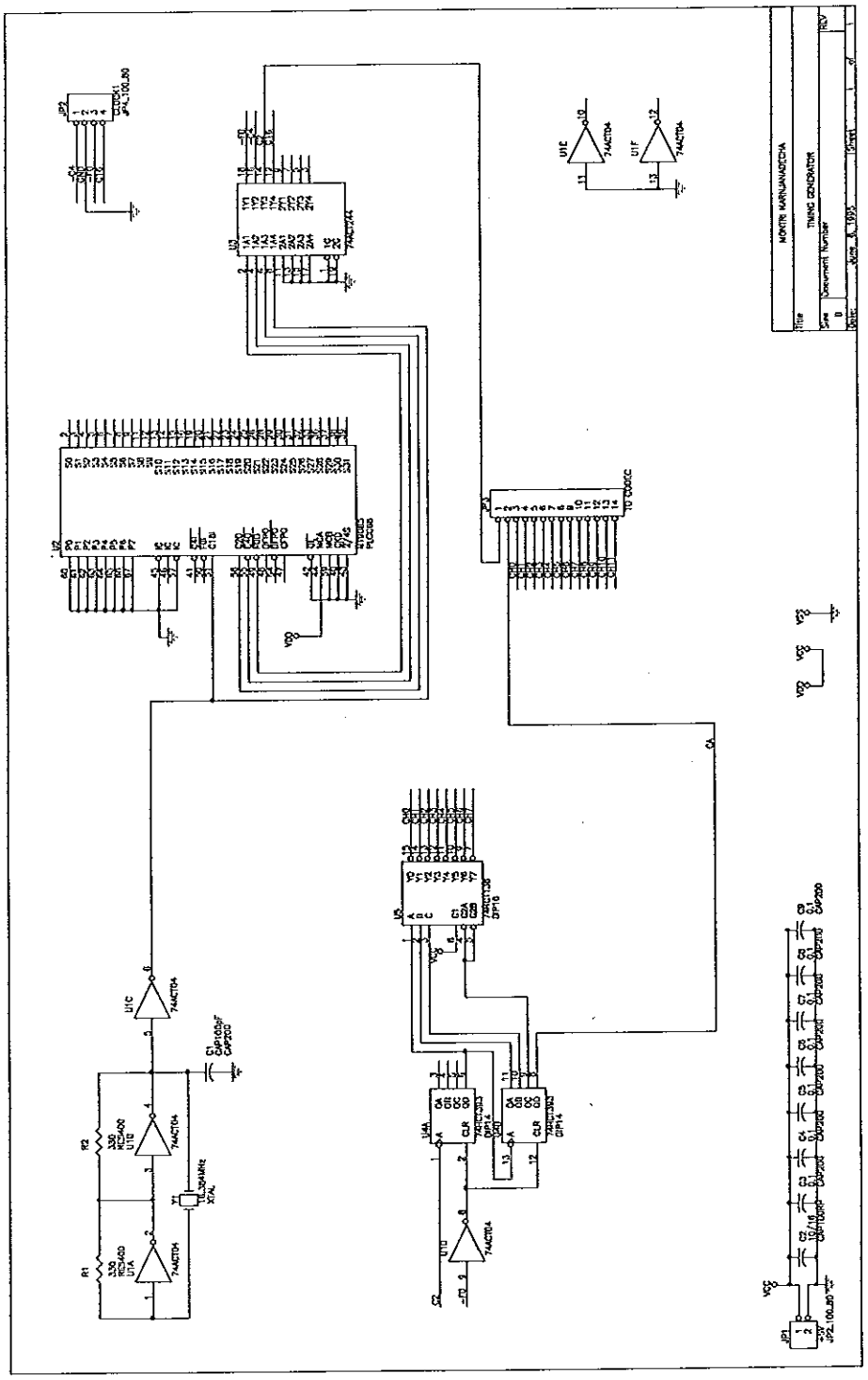
ภาพประกอบ 42 วงจรเปรียบเทียบ



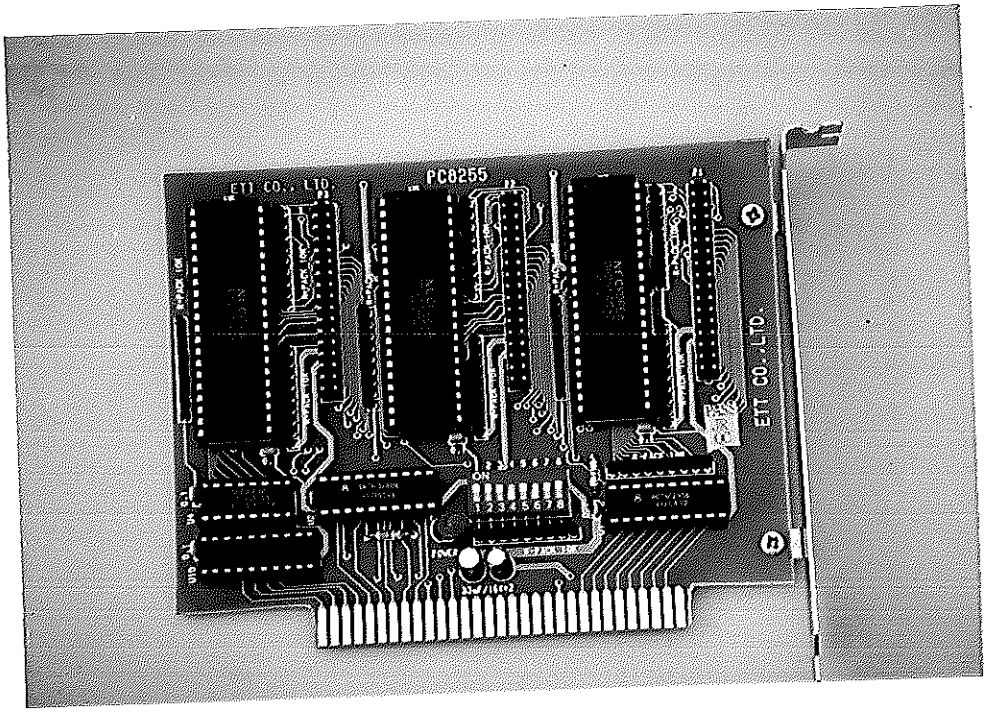
ภาพประกอบ 43 วงจรบัฟเฟอร์เอาต์พุต



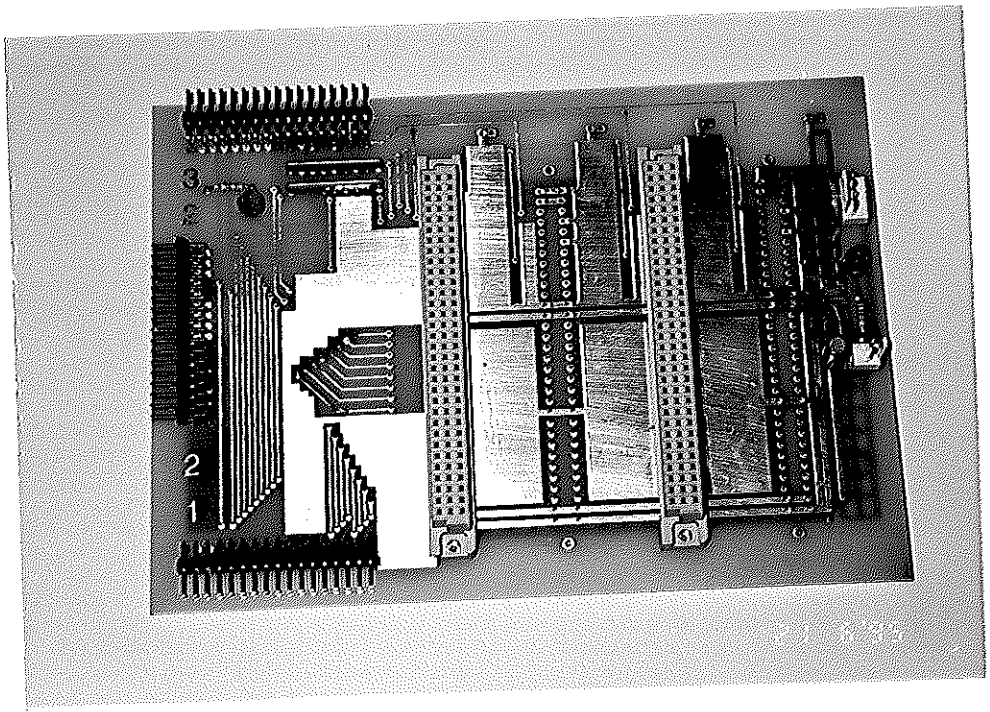
ภาพประกอบ 44 วงจรบัฟเฟอร์อินพุต



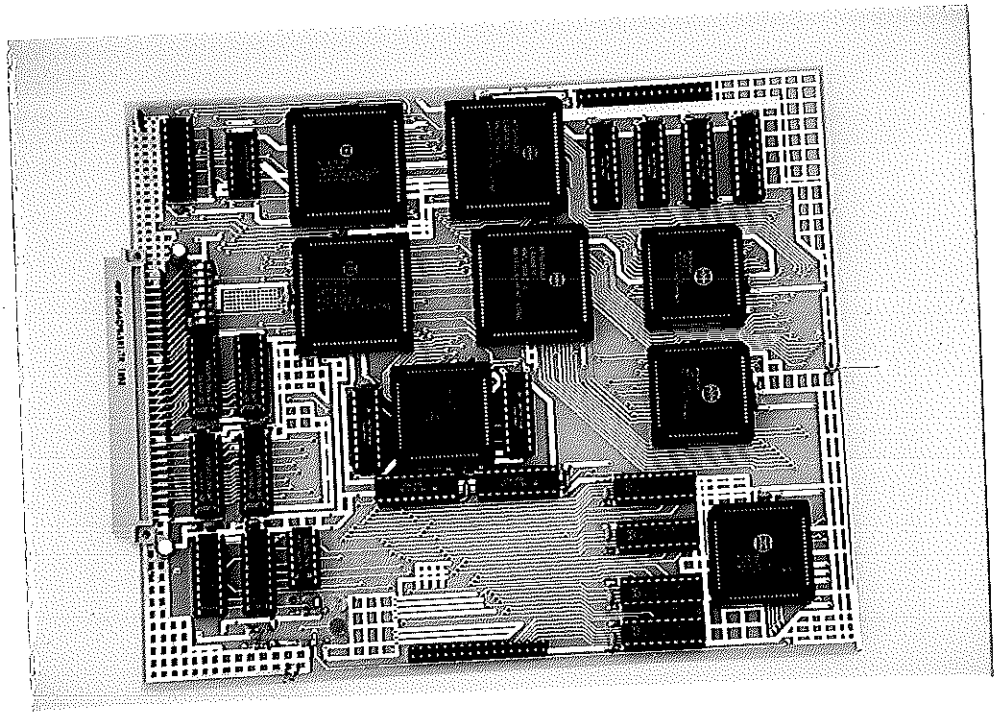
ภาพประกอบ 45 วงจรกำเนิดสัญญาณนาฬิกา



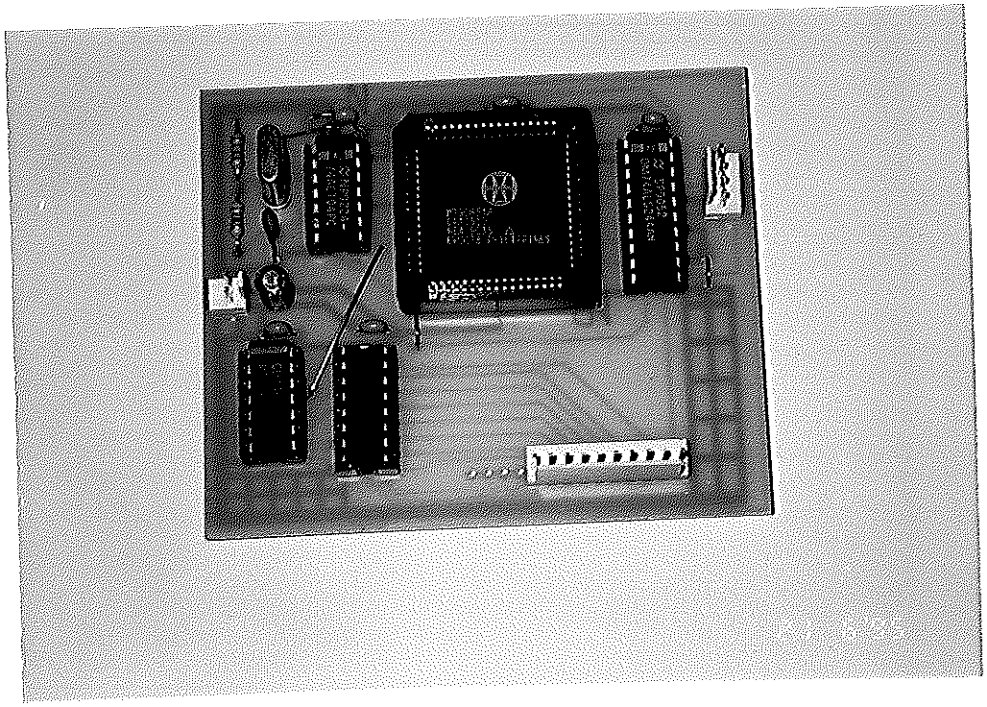
ภาพประกอบ 46 ภาพถ่ายของการ์ด 8255



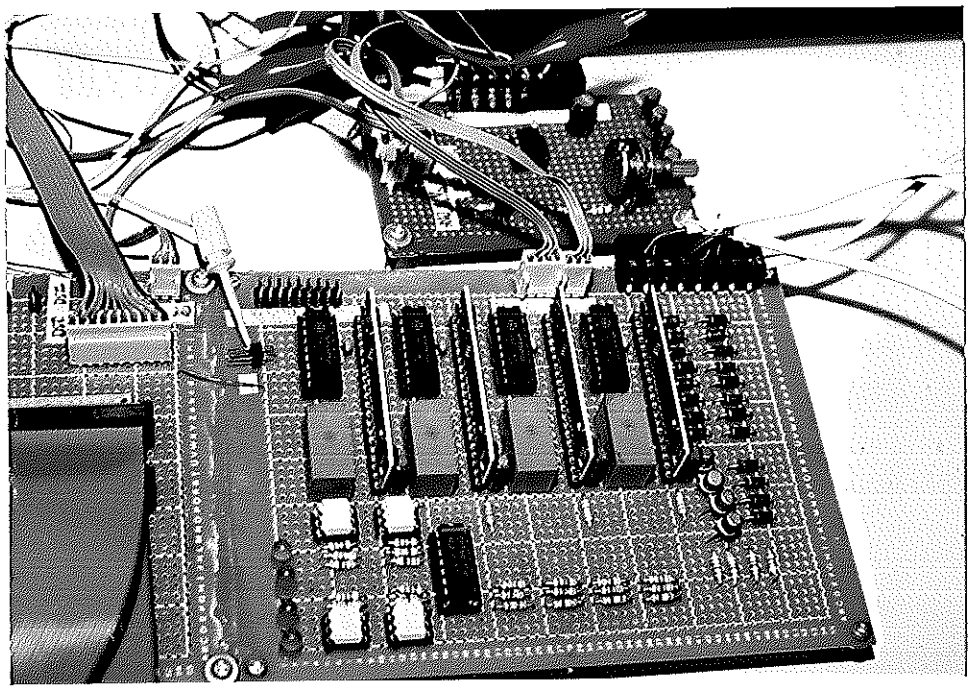
ภาพประกอบ 47 ภาพถ่ายของแผ่นวงจรพิมพ์แผงหลัง



ภาพประกอบ 48 ภาพถ่ายวงจรมอดูลวงจรสวิตซ์



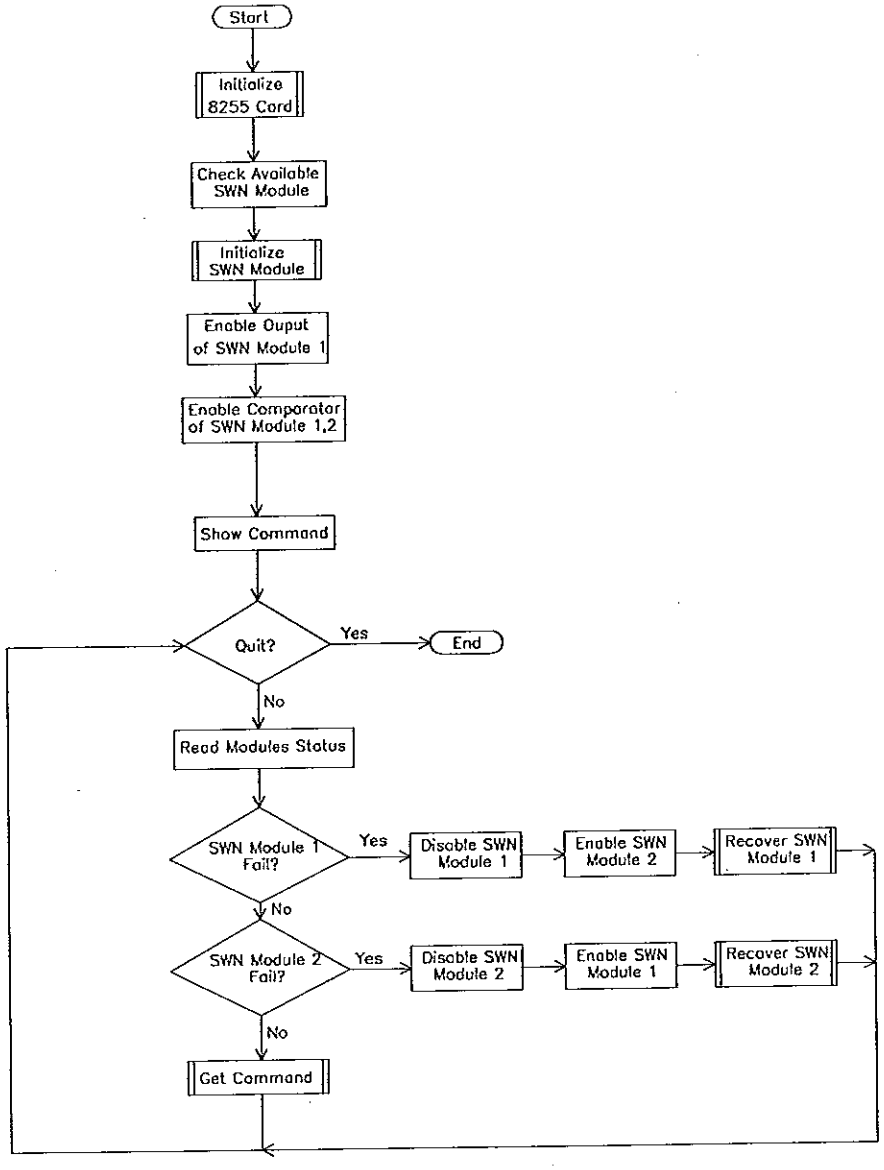
ภาพประกอบ 49 ภาพถ่ายมอดูลกำเนิดสัญญาณนาฬิกา



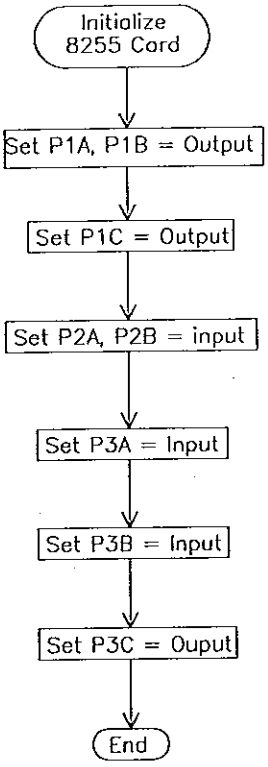
ภาพประกอบ 50 ภาพถ่ายมอดูล SLMA

ภาคผนวก ช.

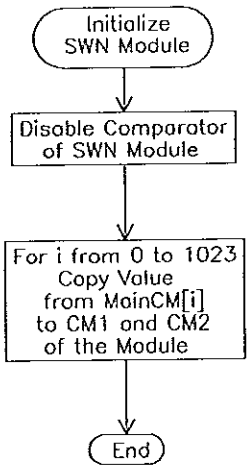
ผังงานของซอฟต์แวร์และรหัสต้นฉบับ



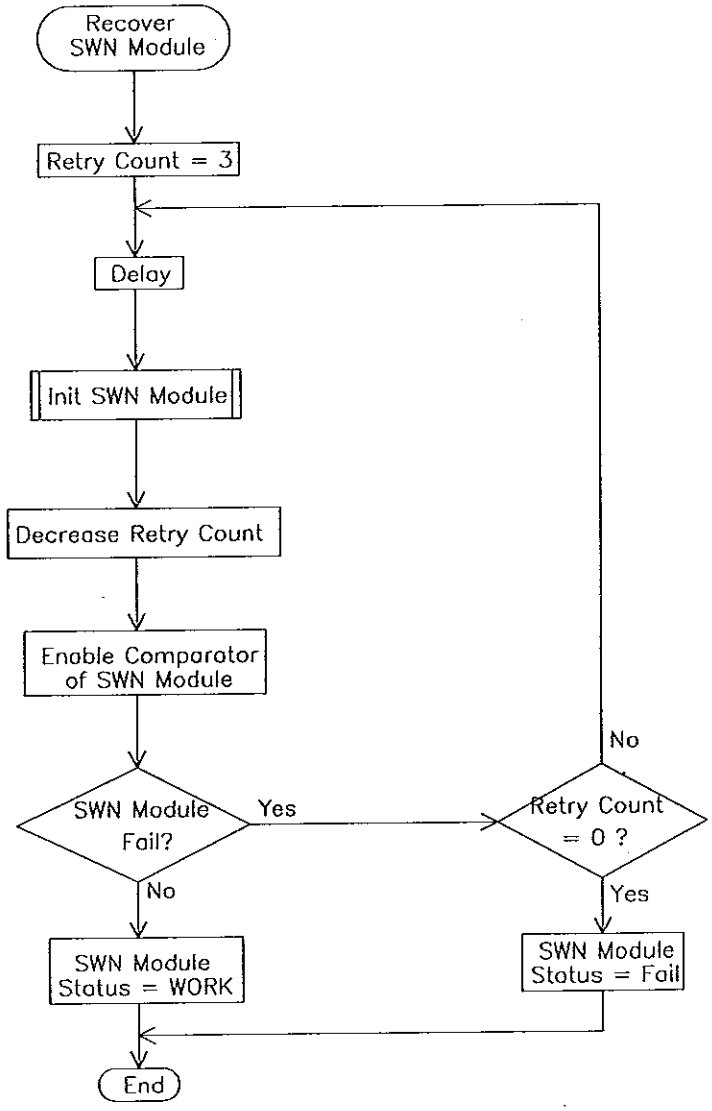
ภาพประกอบ 51 ผังงานของโปรแกรมหลัก



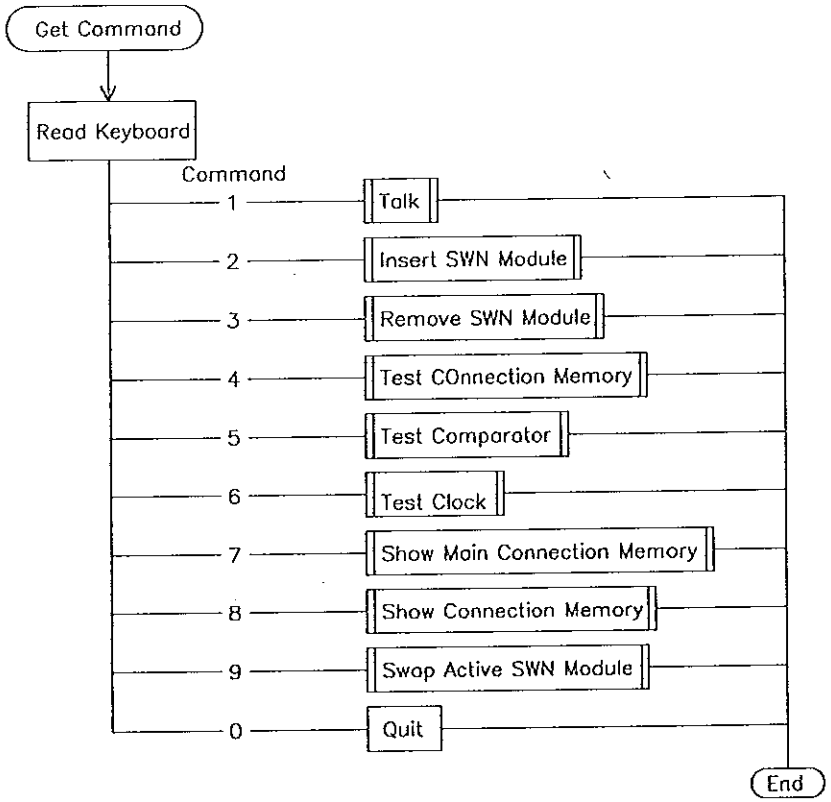
ภาพประกอบ 52 ผังงานของฟังก์ชัน Init 8255



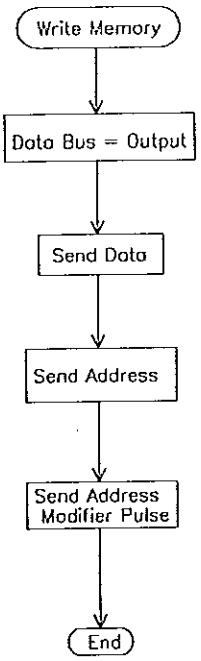
ภาพประกอบ 53 ผังงานของฟังก์ชัน InitSWN



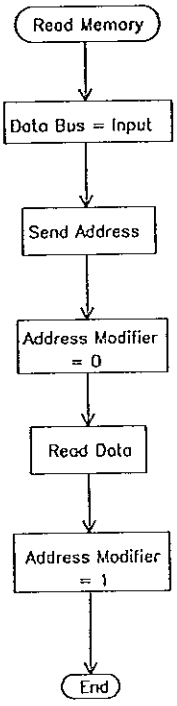
ภาพประกอบ 54 ผังงานของฟังก์ชันการกู้ระบบคืน



ภาพประกอบ 55 ผังงานของฟังก์ชัน Get Command



ภาพประกอบ 56 ผังงานของฟังก์ชัน Write Memory



ภาพประกอบ 57 ผังงานของฟังก์ชัน Read Memory

รหัสต้นฉบับของซอฟต์แวร์

```
#include <stdio.h>
#include <dos.h>
#include <conio.h>
#include <ctype.h>

#define TRUE        0
#define FALSE               -1
#define SWNWORK  TRUE
#define SWNFAIL     FALSE
#define SWNPLUGGED          TRUE
#define SWNUNPLUGGED        FALSE
#define NORMALRETRY         20
#define QUICKRETRY          3

#define PADDR       0x280
#define PAM         0x282
#define PACTRL      0x283
#define PDATA       0x284
#define PDTA        0x286
#define PDCTRL      0x287
#define PFAIL       0x288
#define PLOOP       0x289
#define PRW         0x28A
#define PFCTRL      0x28B

int SWNlist[4]={0,0,0,0};
int
SWNstatus[4]={SWNFAIL,SWNFAIL,
SWNFAIL,SWNFAIL};
int SWNcontrol[4]={6,6,6,6};
int mainCM[1024];
int SWNfailing,retry;

void Beep(void) {
        printf("%c",7);
}

void Init8255(void) {

        outportb(PACTRL,0x80);
        outport(PADDR,0xFFFF);
        outportb(PAM,255);
        outportb(PDCTRL,0x9B);
        outportb(PFCTRL,0x92);
        outportb(PRW,255);
}

void DataIn(void) {
        outportb(PDCTRL,0x9B);
}

void DataOut(void) {
        outportb(PDCTRL,0x89);
}

void WriteMem(int addr, int data) {
int am;
        DataOut();
        outport(PDATA,data&0x7FF);
        outport(PADDR,addr);
        outportb(PRW,254);
        am=0xFF ^ (1 << (addr>>12));
        outportb(PAM,am);
        outportb(PAM,255);
}
int ReadMem(int addr) {
int am;
int din;
        DataIn();
        outport(PADDR,addr);
        outportb(PRW,255);
        am=0xFF ^ (1 << (addr>>12));
        outportb(PAM,am);
        din=inport(PDATA);
        outportb(PAM,255);
        return(din&0x7FF);
}

int WriteVerify(int addr,int data) {

int din;

        WriteMem(addr,data);
        din=ReadMem(addr);
        if (data!=din) {
         return(din+0x8000);
        } else return(0);
}

int TestCM (int SWNno, int cm) {

int i,offs;

        offs=(SWNno-1)*0x1000+(cm-1)*0x400;
        printf("\n");
        for (i=offs;i<offs+0x400; i++) {
         gotoxy(1,wherey());
         printf("%02XH",i);
         if (WriteVerify(i,0)!=0)
          { printf(" Error at %0X (%0X,%0X)%
c\n",i,0,ReadMem(i),7); }
         if (WriteVerify(i,0x7ff)!=0x7ff)
          { printf(" Error at %0X (%0X,%0X)%
c\n",i,0x3FF,ReadMem(i),7); }
         if (WriteVerify(i,0x555)!=0x555)
          { printf(" Error at %0X (%0X,%0X)%
c\n",i,0x155,ReadMem(i),7); }
         if (WriteVerify(i,0x2aa)!=0x2aa)
          { printf(" Error at %0X (%0X,%0X)%
c\n",i,0x2AA,ReadMem(i),7); }
        }
        printf("\n");
        return(0);
}

void FillCM (int SWNno, int cm, int data) {

int i,offs;

        offs=(SWNno-1)*0x1000+(cm-1)*0x400;
        for (i=offs;i<offs+0x400; i++) {
         if (WriteVerify(i,data)!=data) { printf(" Error at
%0X%c\n",i,7); }
        }
}

void ShowCMMem(int SWNno,int from, int to) {

int i,offs,line;

        offs=(SWNno-1)*0x1000;
        printf("\n");
        line=0;
        for (i=from; i<=to; i++) {
```

```
            printf("%5d = %4d   ",i+offs,ReadMem(i+offs));
            if(++line==24*5) {
              line=0;
              if(getch()==27) { return; }
             }
            }
            printf("\nPress any key to continue . . "); getch();

}

void EnableOutput(int SWNno) {
int offs,d;

            offs=(SWNno-1)*0x1000 + 0x800;
            d=SWNcontrol[SWNno-1] & 5;
            WriteMem(offs,d);
            SWNcontrol[SWNno-1]=d;
}

void DisableOutput(int SWNno) {
int offs,d;

            offs=(SWNno-1)*0x1000 + 0x800;
            d=SWNcontrol[SWNno-1] | 2;
            WriteMem(offs,d);
            SWNcontrol[SWNno-1]=d;
}

void EnableCMP(int SWNno) {
int offs,d;

            offs=(SWNno-1)*0x1000 + 0x800;
            d=SWNcontrol[SWNno-1] | 1;
            WriteMem(offs,d);
            SWNcontrol[SWNno-1]=d;
}

void DisableCMP(int SWNno) {
int offs,d;

            offs=(SWNno-1)*0x1000 + 0x800;
            d=SWNcontrol[SWNno-1] & 6;
            WriteMem(offs,d);
            SWNcontrol[SWNno-1]=d;
}


int Switch(int SWNno, int cm, int x, int y) {
int offs,result;

   offs=(SWNno-1)*0x1000+(cm-1)*0x400;
   result=WriteVerify(offs+x,y);
   if (result!=0) { printf("CM error! (Slot %d, CM %d)%
c\n",SWNno,cm,7); }
   return (result);
}

int Cross(int SWNno, int cm, int x, int y) {

   Switch(SWNno,cm,x,y);
   Switch(SWNno,cm,y,x);
   return (0);
}

int Talk(int x, int y) {
int SWNno,i;

   mainCM[x]=y;  mainCM[y]=x;
   i=0;
   while (SWNlist[i]!=0) {
    SWNno=SWNlist[i];
    DisableCMP(SWNno);
   Cross(SWNno,1,x,y);
   Cross(SWNno,2,x,y);
   EnableCMP(SWNno);
   i++;
  }
  return(0);
}

int InitSWN (int SWNno) {
int i;

            DisableCMP(SWNno);
            for (i=0; i<1024; i++) {
             Switch(SWNno,1,i,mainCM[i]);
             Switch(SWNno,2,i,mainCM[i]);
            }
            delay(20);
//          EnableCMP(SWNno);
            return (0);
}

int CheckSWNStatus(int SWNno) {
int mask;

   mask = 1 << (SWNno-1);
   if ((inportb(PFAIL) & mask)==0) {
    return (SWNFAIL);
   } else return (SWNWORK);
}

int CheckSWNPresent(int SWNno) {
int mask;

   mask = 1 << (SWNno-1);
   if ((inportb(PLOOP) & mask)==0) {
    return (SWNPLUGGED);
   } else return (SWNUNPLUGGED);
}

int AddSWN(int SWNno) {

int i;

   if ((SWNno<1) || (SWNno>4)) { printf("No SWNno!%
c\n",7); return (-1); }
   if ((SWNlist[0]==SWNno) || (SWNlist[1]==SWNno) ||
          (SWNlist[2]==SWNno) || (SWNlist[3]==SWNno))
{
    printf("SWN has already installed!\n%c",7);
   } else {
          while (CheckSWNPresent(SWNno)==
SWNUNPLUGGED) {
           if (kbhit()) { getch(); return(0); }
          }
          delay(500);
          DisableOutput(SWNno);
          i=0;
          while (SWNlist[i]!=0) { i++; }
          SWNlist[i]=SWNno;
          InitSWN(SWNno);
          if (i==0) { EnableOutput(SWNno); } else
DisableOutput(SWNno);
          EnableCMP(SWNno);
    }
   return (0);
}


int RemoveSWN(int SWNno) {

int i,j;
```

```
  if ((SWNno<1) || (SWNno>4)) { printf("No SWNno!%
c\n",7); return (-1); }
  if ((SWNlist[0]!=SWNno) && (SWNlist[1]!=SWNno) &&
        (SWNlist[2]!=SWNno) && (SWNlist[3]!=
SWNno)) {
    printf("No SWN installed at SWNno %d!\n%c",SWNno,7);
  } else {
          i=0;
          while (SWNlist[i]!=SWNno) { i++; }
          j=0;
          while (SWNlist[j]!=0) { j++; }
          switch(j) {
           case 1: SWNlist[0]=0;
                   break;
           case 2: DisableOutput(SWNlist[0]);
                   EnableOutput(SWNlist[1]);
                   SWNlist[0]=SWNlist[1];
                   SWNlist[1]=0;
                   break;
           case 3: DisableOutput(SWNlist[0]);
                   EnableOutput(SWNlist[2]);
                   SWNlist[0]=SWNlist[2];
                   SWNlist[2]=0;
                   break;
           case 4: DisableOutput(SWNlist[0]);
                   EnableOutput(SWNlist[3]);
                   SWNlist[0]=SWNlist[3];
                   SWNlist[3]=0;
                   break;
          }
  }
  while (CheckSWNPresent(SWNno)==SWNPLUGGED) { ;
}
  delay(500);
  EnableCMP(SWNlist[0]);
  return (0);
}

void ShowFirstScreen(void) {

          clrscr();
          printf(" SWN No.  Present  Status  Output
Comparator  Clock Test\n");
          printf(" ------  -------  ------  ------  ----------  ----------
-\n");
          gotoxy(1,10);
          printf(" 1) Talk\n");
          printf(" 2) Insert SWN\n");
          printf(" 3) Remove SWN\n");
          printf(" 4) Show CM data\n");
          printf(" 5) Show Main CM data\n");
          printf(" 9) Quit\n");
          gotoxy(1,17); printf(" Command : ");
}



int GetCommand(void) {
char ch;
char s[50];
int x,y,done;

          done=FALSE;
          ch=toupper(getch()); printf("%c",ch);
          switch(ch) {
           case '0':
           case '1': window(1,19,80,25);
                     printf("Channel VS Channel : ");
                     gets(s);
                     if (s[0]!=0) {
                      sscanf(s,"%d %d",&x,&y);
                      Talk(x,y);
                     }
                     clrscr();
                     window(1,1,80,25);
                     break;
           case '2': window(1,19,80,25);
                     printf("SWN No. to be added : ");
                     gets(s);
                     if (s[0]!=0) {
                      sscanf(s,"%d",&x);
                      AddSWN(x);
                     }
                     clrscr();
                     window(1,1,80,25);
                     break;
           case '3': window(1,19,80,25);
                     printf("SWN No. to be removed : ");
                     gets(s);
                     if (s[0]!=0) {
                      sscanf(s,"%d",&x);
                      RemoveSWN(x);
                     }
                     clrscr();
                     window(1,1,80,25);
                     break;
           case '4': window(1,19,80,25);
                     printf("SWN No. : ");
                     gets(s);
                     if (s[0]!=0) {
                      sscanf(s,"%d",&x);
                      ShowCMMem(x,0,2047);
                     }
                     clrscr();
                     window(1,1,80,25);
                     ShowFirstScreen();
                     break;
           case '5': y=0;
                     printf("\n");
                     for (x=0;x<1024;x++) {
                      printf("%4d=%4d ",x,mainCM[x]);
                      if (++y>=24*8) {if (getch()==27)
{break;} y=0;}
                     }
                     ShowFirstScreen();
                     break;
           case '9':
           case 27 : done=TRUE;
                     break;
           default :
                     break;
          }
          return(done);
}

void UpdateScreen(void) {
int i;
          gotoxy(1,3);
          for(i=1;i<=4;i++) {
           printf("   %d",i);
           gotoxy(14,i+2);
           if ( (SWNlist[0]==i) || (SWNlist[1]==i) ||
              (SWNlist[2]==i) || (SWNlist[3]==i) ) {
            printf("YES");
           } else printf("NO ");
           gotoxy(24,i+2);
           if (SWNstatus[i-1]==SWNWORK) {
            printf("WORK");
           } else printf("FAIL");
           gotoxy(34,i+2);
           if ((SWNcontrol[i-1]&2)==0) {
            printf("ON ");
           } else printf("OFF");
```

```
            gotoxy(45,i+2);
            if ((SWNcontrol[i-1]&1)==1) {
              printf("ON \n");
            } else printf("OFF\n");
            gotoxy(58,i+2);
            if ((SWNcontrol[i-1]&4)==4) {
              printf("NO \n");
            } else printf("YES\n");
           }
           gotoxy(12,17); clreol();
}


void main (void) {

int i,j,tmp;
int done,OneSWN;

           directvideo=1;
           Init8255();
           ShowFirstScreen();
           for (i=0; i<1024; i++) {
            mainCM[i]=i;
           }
           j=0;
           for (i=1; i<=4; i++) {
            if (CheckSWNPresent(i)==SWNPLUGGED) {
              SWNlist[j++]=i;
              DisableOutput(i);
              DisableCMP(i);
             }
           }
           for (i=0;i<j;i++) {
            InitSWN(SWNlist[i]);
            SWNstatus[SWNlist[i]-1]=CheckSWNStatus
 (SWNlist[i]);
           }
           EnableOutput(SWNlist[0]);
           EnableCMP(SWNlist[0]);
           EnableCMP(SWNlist[1]);
           done=FALSE;
           while (done==FALSE) {
            i=0;
            if (SWNstatus[SWNlist[1]-1]==SWNWORK) {
 OneSWN=TRUE; } else OneSWN=FALSE;
            if (OneSWN==TRUE) {
             if (CheckSWNStatus(SWNlist[0])==SWNFAIL)
 {
               SWNfailing=SWNlist[0];
               retry=QUICKRETRY;
           } else { SWNfailing=0; retry=0; }
          } else {
           // > 2 SWN
            if (CheckSWNStatus(SWNlist[0])==
SWNFAIL) {
                 if (SWNstatus[SWNlist[1]-1]==
SWNWORK) {
                  DisableOutput(SWNlist[0]);
                  EnableOutput(SWNlist[1]);
                  tmp=SWNlist[0];
                  SWNlist[0]=SWNlist[1];
                  SWNlist[1]=tmp;
                  SWNfailing=SWNlist[1];
                  retry=NORMALRETRY;
                 } else {
                   SWNfailing=SWNlist[0];
                   retry=QUICKRETRY;
                  }
             }
             if ((SWNfailing==0) && (CheckSWNStatus
 (SWNlist[1])==SWNFAIL)) {
                  retry=NORMALRETRY;
                  SWNfailing=SWNlist[1];
             }
            }
           if ((SWNfailing!=0) && (retry!=0) &&
 (SWNstatus[SWNfailing-1]!=SWNFAIL)) {
            DisableCMP(SWNfailing);
            Beep();
            InitSWN(SWNfailing);
            EnableCMP(SWNfailing);
            retry--;
            if (CheckSWNStatus(SWNfailing)==
 SWNWORK) {
                  retry=0;
                  SWNfailing=0;
             }
             if (retry==0) {
                  SWNstatus[SWNfailing-1]=SWNFAIL;
             }
           }
           if (kbhit()) {
             done=GetCommand();
           }
           UpdateScreen();
          }
 }
```

ภาคผนวก ค.

การออกแบบ PAL

เอกสารที่ได้จากการคอมไพล์โปรแกรม CUPL มีดังนี้

```
*************************************************************
                        COMPARATOR
*************************************************************

CUPL        4.0a Serial# MD-STS-1000
Device      p16r8  Library DLIB-h-249-10
Created     Mon Jun 12 11:14:41 1995
Name        COMPARATOR
Partno      U6
Revision    01
Date        06/21/95
Designer    MONTRI
Company     . ELECTRICAL ENGINEERING
Assembly    None
Location    U6

============================================================
                    Expanded Product Terms
============================================================

Q0.d  =>
   !CT
   # EN & !Q0
   # EN & I1
   # EN & I2
   # EN & I3
   # EN & I4

============================================================
                      Symbol Table
============================================================

Pin Variable                        Pterms  Max   Min
Pol  Name          Ext   Pin   Type  Used  Pterms Level
---  ----          ---   ---   ----  ----- ------ -----

     CLOCK           1     V     -     -     -
     CT              7     V     -     -     -
     EN              2     V     -     -     -
     I1              3     V     -     -     -
     I2              4     V     -     -     -
     I3              5     V     -     -     -
     I4              6     V     -     -     -
  !  OE              11    V     -     -     -
     Q0              19    V     -     -     -
     Q0         d    19    X     6     8     1


LEGEND   F : field      D : default variable      M : extended node
         N : node       I : intermediate variable T : function
         V : variable   X : extended variable     U : undefined
```

```
============================================================
                        Fuse Plot
============================================================

Pin #19
 00000 ----------------x-------->
 00032 x--x------------------------>
 00064 x---x------------------------>
 00096 x-----x------------------------>
 00128 x-------x------------------------>
 00160 x---------x------------------------>
 00192 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 00224 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
Pin #18
 00256 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 00288 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 00320 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 00352 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 00384 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 00416 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 00448 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 00480 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
Pin #17
 00512 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 00544 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 00576 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 00608 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 00640 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 00672 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 00704 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 00736 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
Pin #16
 00768 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 00800 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 00832 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 00864 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 00896 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 00928 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 00960 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 00992 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
Pin #15
 01024 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 01056 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 01088 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 01120 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 01152 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 01184 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 01216 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 01248 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
```

Pin #14

01280 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
01312 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
01344 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
01376 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
01408 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
01440 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
01472 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
01504 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

Pin #13

01536 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
01568 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
01600 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
01632 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
01664 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
01696 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
01728 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
01760 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

Pin #12

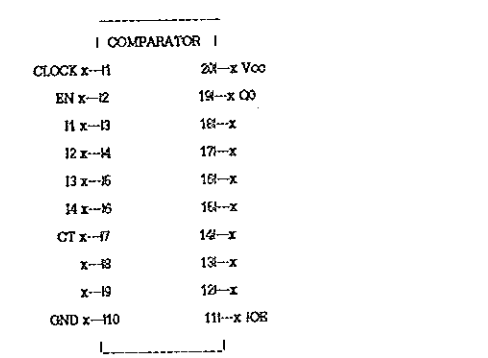
01792 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
01824 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
01856 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
01888 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
01920 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
01952 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
01984 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
02016 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

LEGEND X : fuse not blown
- : fuse blown

==========

Chip Diagram

==========



ประวัติผู้เขียน

ชื่อ นายมนตรี กาญจนะเดชะ

วัน เดือน ปีเกิด 28 เมษายน 2511

วุฒิการศึกษา

วุฒิ	ชื่อสถาบัน	ปีที่สำเร็จการศึกษา
วิศวกรรมศาสตรบัณฑิต (วิศวกรรมไฟฟ้า)	มหาวิทยาลัยสงขลานครินทร์	2533

ตำแหน่งและสถานที่ทำงาน

ตำแหน่ง อาจารย์

สถานที่ทำงาน ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์
มหาวิทยาลัยสงขลานครินทร์