เครื่องทดสอบสัญญาณแรนดัมแสดงผลบนคอมพิวเตอร์

TRULY RANDOM TEST BED

โดย

นายชัยวัฒน์ พันธุวัฒน์

นายชาญวิทย์ นาคไพรัชช์

ปริญญานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

ภาควิชาวิศวกรรมสารสนเทศ

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2547

TRULY RANDOM TEST BED

BY

MR. CHAIWAT PHANTUWAT

MR. CHANVIT NAKPHAIRAT

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF

THE REQUIREMENT FOR THE DEGREE OF

BACHELOR IN DEPARTMENT OF INFORMATION ENGINEERING

FACULTY OF ENGINEERING

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

2004

หัวข้อปริญญานิพนธ์	เครื่องทดสอบสัญญาณแรมดัมแสดงผลบนคอมพิวเตอร์
นักศึกษา	นายชัยวัฒน์ พันธุวัฒน์ รหัสนักศึกษา 44010108
	นายชาญวิทย์ นาคไพรัชช์ รหัสนักศึกษา 44010113
อาจารย์ที่ปรึกษา	อาจารย์กฤดากร กล่อมการ
ระดับการศึกษา	ปริญญาตรี วิศวกรรมศาสตรบัณฑิต
ภาควิชา	วิศวกรรมสารสนเทศ
ปีการศึกษา	2547

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง อนุมัติให้ปริญญานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิศวกรรมสารสนเทศบัณฑิต

กฤดากร

.....

(อาจารย์กฤดากร กล่อมการ)

อาจารย์ที่ปรึกษา

หัวข้อปริญญานิพนธ์	เครื่องทดสอบสัญญาณแรนดัมแสดงผลบนคอมพิวเตอร์	
นักศึกษา	นายชัยวัฒน์ พันธุวัฒน์	รหัสนักศึกษา 44010108
	นายชาญวิทย์ นาคไพรัชช์	รหัสนักศึกษา 44010113
อาจารย์ที่ปรึกษา	อาจารย์กฤดากร กล่อมการ	
ระดับการศึกษา	ปริญญาตรี วิศวกรรมศาสตรบัณฑิต	
ภาควิชา	วิศวกรรมสารสนเทศ	
ปีการศึกษา	2547	

บทคัดย่อ

ในปัจจุบันนี้การสร้างสัญญาณแรนดัมขึ้นมา เช่น วงจรกำเนิดค่าสัญญาณแรนดัมเพื่อใช้ในการวิเคราะห์ยังทำได้ค่อนข้างลำบาก เนื่องจากการแสดงผลบน Oscilloscope หรือ โปรแกรมอย่าง MATLAB ก็ยังใช้งานได้ยากอยู่ เนื่องจากสัญญาณแรนดัมที่จะนำมาใช้งานได้จริงนั้น ต้องผ่านการวิเคราะห์ทางสถิติโดยดำเนินการตามมาตรฐาน The Federal Information Processing Standard (FIPS 140-2)

โครงงานนี้จะทำการสร้างเครื่องทดสอบสัญญาณแรนดัมโดยสามารถนำวงจรกำเนิดค่าแรนดัมมาเชื่อมต่อกับวงจรเครื่องทดสอบสัญญาณแรมดอมซึ่งจะทำหน้าที่เก็บข้อมูลในรูปแบบไบนารี่ จากนั้นจะนำข้อมูลที่ได้เก็บไว้ มาวิเคราะห์ ทดสอบค่าทางสถิติได้ว่าถูกต้องตามมาตรฐานและยอมรับได้หรือไม่และแสดงผลบนจอคอมพิวเตอร์

Thesis Title	Truly Random Test Bed		
Student	Mr. Chaiwat	Phantuwat	ID. 44010108
	Mr. Chanvit	Nakphairat	ID. 44010113
Advisor	Mr. Kitdakorn	Klomkarn	
Graduate Level	Bachelor Degree of Information Engineering		
Department	Information Engineering		
Academic Year	2004		

Abstract

Now , making Truly Random Signal (ie. Generated Random Signal Circuit) for analysis is hard because the way to display result on Oscilloscope or Program such as MATLAB is hard for end users . Truly Random signal must pass standard analysis method (The Federal Information Processing Standard : FIPS 140-1)

This project is to make Truly Random Test Bed . It can connect and correct data (binary bit) from Generated Random Signal Circuit .Data will be send to Computer for standard test and display result on computer .

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้ ทางผู้จัดทำ ได้ทำจนประสบความสำเร็จขึ้นมาได้ เพราะได้รับความอนุเคราะห์ทางด้านต่างๆ ทั้งคำปรึกษาทางด้านวิชาการ และคำแนะนำในการปฏิบัติงานรวมถึงการช่วยเหลือทางด้านอุปกรณ์และเครื่องมือต่างๆจาก อาจารย์กฤดากร กล่อมการ ทางผู้จัดทำขอขอบพระคุณท่านอาจารย์มา ณ ที่นี้ด้วย

ทางผู้จัดทำขอขอบพระคุณ คุณพ่อ คุณแม่ ที่คอยให้ความรักความห่วงใยตลอดจนให้การสนับสนุนทางด้านการศึกษาด้วยดีมาตลอด รวมถึง คุณเจตน์ ออสวัสดิ์ ที่ให้ความช่วยเหลือ และให้คำแนะนำทางด้านวิชาการ ตลอดจนให้ยืมอุปกรณ์ต่างๆที่ใช้ในการทำโครงงาน

คุณประโยชน์ต่างๆที่เกิดขึ้นจากปริญญานิพนธ์ฉบับนี้ ทางผู้จัดทำขอมอบแด่ผู้มีพระคุณทุกๆท่าน

นายชัยวัฒน์ พันธุวัฒน์
นายชาญวิทย์ นาคไพรัชช์
ผู้จัดทำ

สารบัญ

สารบัญรูปภาพ

สารบัญตาราง

บทที่ 1
บทนำ

1.1 แนวคิดและที่มาของปัญหา

ปัจจุบันเครื่องจักรต่าง ๆ เช่น เครื่องเล่นเกมส์เสี่ยงโชค , เครื่องออกสลากรางวัล การกำเนิดพาสเวิร์ด (Password) , การกำเนิดกุญแจลับ ต้องการความเป็น แรนดัม แบบแท้จริง เพื่อไม่ให้ผู้อื่นสามารถคาดเดาผลที่จะออกมาได้ ดังนั้นความต้องการวงจรกำเนิดค่า แรนดัม จึงเป็นสิ่งจำเป็น

แต่เนื่องจากปัจจุบันยังไม่มีอุปกรณ์ที่ใช้ทดสอบมาตรฐานของเครื่องกำเนิดสัญญาณแรนดัม ซึ่งเป็นอุปกรณ์สำคัญที่ใช้ในการวิเคราะห์วิจัยด้าน Cryptographic โครงงานนี้จึงมีส่วนสำคัญในการพัฒนาการวิเคราะห์วิจัยด้าน Cryptographic ภายในประเทศ

1.2 จุดประสงค์ของโครงงาน

1. เพื่อสร้างอุปกรณ์ทดสอบวงจรกำเนิดสัญญาณ แรนดัม โดยการต่อเข้ากับทดสอบสัญญาณแรนดัม เพื่อนำสัญญาณ แรนดัม เข้าสู่คอมพิวเตอร์เพื่อนำมาวิเคราะห์ต่อไป

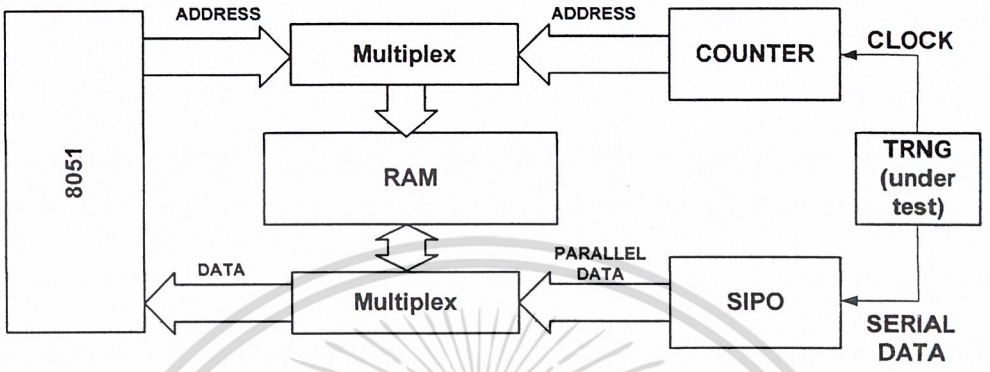
2. สร้างโปรแกรมที่สามารถวิเคราะห์สัญญาณ แรนดัมที่อยู่บนอุปกรณ์ทดสอบสัญญาณแรนดัมว่าผ่านตามมาตรฐานที่กำหนดได้อัตโนมัติ รวมถึงแสดงผลลัพธ์ในรูปแบบกราฟฟิกยูสเซอร์อินเตอร์เฟส (GUI) เพื่อให้วิเคราะห์ได้ง่ายและสะดวกขึ้น

3. เพิ่มทางเลือกในการวิเคราะห์ นอกจาก เครื่องออสซิโลสโคป (Oscilloscope) และโปรแกรม MATLAB ซึ่งใช้งานในการวิเคราะห์ในส่วนนี้ได้ยากกว่า

1.3 ขอบเขตของโครงงาน

ขอบเขตของโครงงานนี้จะสร้าง อุปกรณ์ชุดทดสอบคุณสมบัติทางสถิติ ของวงจรกำเนิดค่าแรนดัม โดยสามารถทำการ วัดวงจรกำเนิดสัญญาณ แรนดัม ได้อย่างสะดวก ซึ่งวงจรหลักจะใช้ไมโครคอนโทลเลอร์ 8051 เป็นตัวควบคุมการทำงานและเก็บข้อมูลบิตที่เกิดจากวงจรกำเนิดสัญญาณ และนำข้อมูลที่ได้ไปทดสอบทางสถิติบนคอมพิวเตอร์ โดยค่าทางสถิติที่ทดสอบจะดำเนินตามมาตรฐาน The Federal Information Processing Standard (FIPS 140-1) และแสดงบนจอคอมพิวเตอร์ ในรูปแบบกราฟฟิกยูสเซอร์อินเตอร์เฟส ที่สร้างด้วยโปรแกรม Visual Basic

1.4 สถาปัตยกรรมหลักของระบบ



รูปที่ 1.1 โครงสร้างของอุปกรณ์ Truly Random Test bed

1.5 ผลที่คาดว่าจะได้รับ

1. สามารถสร้างอุปกรณ์สำคัญที่ใช้ในการพัฒนาและวิจัยทางด้าน Cryptographic
2. สามารถตรวจวงจรกำเนิดสัญญาณแรนดัม ว่าได้มาตรฐานและยอมรับได้หรือไม่
3. การวัดค่าทางสถิติสามารถทำได้สะดวกและใช้งานง่ายโดยผ่านกราฟฟิกยูสเซอร์อินเตอร์เฟส

1.6 อุปกรณ์ที่ใช้การพัฒนาโครงงาน

1.6.1 ฮาร์ดแวร์

1. คอมพิวเตอร์ 1 เครื่อง
2. ไมโครคอนโทลเลอร์ 8051
3. วงจรกำเนิดสัญญาณ แรนดัม
4. ET-AFP V1.0 (Atmel Flash Programmer)

1.6.2 ซอร์ฟแวร์

1. โปรแกรม Visual Basic
2. โปรแกรม Keil
3. โปรแกรม Emulator 8051
4. โปรแกรม Protel 99 SE
5. โปรแกรม ET-AFP V3.4 (Atmel Flash Program)

บทที่ 2
ทฤษฎีที่เกี่ยวข้อง

2.1 ค่าแรนดัม (Random number)

ค่าแรมดัม หมายถึง เลขสุ่มที่มีกระจายอย่างอิสระและไม่สามารถระบุตำแหน่งที่แน่นอนได้ ซึ่งจะมีลักษณะไม่มีรอบการเกิดซ้ำขึ้นมาใหม่ (Unpattern) ทำให้ไม่สามารถคาดเดาผลที่จะออกมาได้(Unpredictable) ค่าที่เป็นแรมดัมที่แท้จริงนั้นเมื่อนำไปบีบอัดข้อมูล(Compression)จะได้อัตราการบีบอัดที่ต่ำมาก

ค่าแรนดัม สามารถนำไปใช้ได้ในหลายๆ เรื่องตามแต่จุดประสงค์ ทั้งงานวิจัยด้านการเข้ารหัส (Encryption) การจำลองแบบ (Simulation) รวมทั้งเกมส์การเสี่ยงโชคต่างๆ ถึงแม้ว่าการสร้างจากการเขียนทางภาษาคอมพิวเตอร์ (Generator Program) จะสามารถสร้างค่าแรนดัมขึ้นมาได้ แต่ก็ไม่ได้ค่าที่เป็นแรนดัมแท้จริง (Truly Random) เพราะผลที่ออกมาสามารถที่จะคาดเดาได้ การสร้างค่าแรนดัมจากการกำเนิดค่าแรนดัม (Random Number Generator) โดยใช้ฮาร์ดแวร์ จึงเป็นสิ่งจำเป็น

การกำเนิดค่าแรนดัม (Random Number Generator)

การกำเนิดค่าแรนดัมจะสร้างค่าเลขซีเควนซ์ (Sequence Number) ที่มีการกระจายที่เป็น Uniform โดยที่เลขชุดนี้ต้องมีความเป็นอิสระที่แท้จริง เราสามารถแบ่งแยกชนิดได้จากอัลกอรึทึมที่ใช้ระหว่าง

1.อัลกอรึทึมแบบดีเทอมินิสติก (Deterministic algorithms) สามารถสร้างได้จากทั้งฮาร์ดแวร์กับแบบซอฟแวร์ ซึ่งจะเป็นการสร้างค่าแรมดัมที่มีผลลัพธ์เป็นเชิงเส้น (Linear) เราจะเรียกว่าเครื่องกำเนิดสัญญาณแบบกึ่งแรมดัม (Pseudo-Random Number Generators)

2.อัลกอรึทึมแบบนอลดีเทอมินิสติก (Non-deterministic algorithms) สามารถสร้างได้จากฮาร์ดแวร์เท่านั้น ซึ่งจะเป็นการสร้างค่าแรมดัมที่มีผลลัพธ์ไม่เป็นเชิงเส้น (Nonlinear) เราจะเรียกว่าเครื่องกำเนิดสัญญาณแบบแรมดัม (Random Number Generators)

ซึ่งในการใช้แบบเข้ารหัสลับจะไม่ใช้อัลกอรึทึมที่เป็นแบบเชิงเส้นเพราะสามารถคาดเดาผลได้และเนื่องจากแต่ละแอพพลิเคชั่นมีความต้องการค่าแรนดัมที่ใช้การกำเนิดค่าแรนดัมในรูปแบบที่แตกต่างกัน ดังนั้นการจะวิเคราะห์ถึงการกำเนิดค่าแรนดัมว่ามี

คุณสมบัติหรือเหมาะกับแอพพลิเคชั่นประเภทไหนจึงมีความจำเป็น โดยสามารถตรวจสอบได้จากการวัดค่าทางสถิติ (Statistical Testing)

แม้ว่าจะมีซอฟแวร์ที่สามารถสร้างค่าแรนดัมออกมาได้แต่ก็ไม่ได้ความเป็นอิสระระหว่างซีเควนซ์ เพราะ ยังใช้อัลกอริทึมในการคำนวณค่าถัดไปจากค่าก่อนหน้านี้อยู่ ยกตัวอย่างเช่น การใช้คอมพิวเตอร์ในงานจำลองสถานการณ์จริง(simulation) หรือเพียงแต่ใช้เครื่องคิดเลขแบบพกติดตัวเล่นการพนันคงจะทราบกันดีว่าซอฟแวร์ในคอมพิวเตอร์นั้นสามารถสร้างเลขสุ่ม (random number) ได้ อย่างไรก็ตามเลขสุ่มที่เกิดขึ้นมานี้ ถึงจะดูเหมือนว่าเกิดขึ้นมาโดยไม่มีแบบแผนนั้น ก็เป็นเพียงเลขสุ่มแบบกึ่งแรมดัม(psuedo-random number)เท่านั้น ซึ่งต่างจากเลขสุ่มแบบแท้จริงที่เกิดจากการทอดลูกเต๋าเพราะเลขสุ่มของคอมพิวเตอร์เกิดขึ้นจากโปรแกรม ง่าย ๆ เช่น

$$X(n+1) = cX(n) \bmod m \tag{2.1}$$

โดยที่ $X(n)$ คือเลขสุ่มครั้งที่ n ส่วน c และ m เป็นเลขจำนวนเต็ม และ mod หมายถึงการหารเลขจำนวนเต็มแล้วเอาเฉพาะเศษ เช่น 5 mod 3 จะได้ 2 (5 หาร 3 เหลือเศษ 2)

การเกิดจากโปรแกรมง่าย ๆ นี้จึงมีความหมายว่า การเกิดของเลขสุ่มนี้แฝงไปด้วยความเป็นรูปแบบการวนซ้ำขึ้นมาใหม่(Pattern) โดยสามารถอธิบายได้ด้วยโปรแกรมนั้นนั่นเอง เลขสุ่มที่เกิดจากคอมพิวเตอร์นี้มีลักษณะเด่นอีกอย่างหนึ่งคือจะเกิดขึ้นอย่างแตกต่างกันมากหากเริ่มต้นด้วยตัวตั้งต้น $X(0)$ หรือที่เรียกกันว่าค่าเริ่มต้น (Seed) คนละตัวกัน นั่นก็คือ ระบบสร้างเลขสุ่มแบบกึ่งแรมดัมนี้เป็นระบบที่ไวต่อสภาวะตั้งต้น (Initial condition) นั้นเอง

ส่วนการสร้างค่าแรมดัมจากฮาร์ดแวร์ หรือ การใช้อัลกอริทึมแบบนอลเทอมินิสติกจะเป็นเลขสุ่มอย่างแท้จริงเนื่องจากเป็นผลของปรากฏการณ์ทางกายภาพของอุปกรณ์ฮาร์ดแวร์เองทำให้ไม่สามารถคาดเดาผลของค่าที่จะเกิดขึ้นต่อไปได้ ยกตัวอย่างเช่น ช่วงเวลาในการอ่อนกำลังลงของวัตถุกัมมันตรังสี หรือสัญญาณรบกวน(Noise) ที่เกิดจากความร้อนที่ตัวอุปกรณ์ อิเล็กทรอนิกส์ประเภทสารกึ่งตัวนำ หรือ วงจรกำเนิดสัญญาณเคออส (Chaotic circuit) เป็นต้น

ทุกการกำเนิดค่าแรนดัมจะมีคุณสมบัติที่แตกต่างกัน ไม่มีการกำเนิดค่าแรนดัมไหนที่จะมีคุณสมบัติเหมาะกับทุกแอพพลิเคชั่น ตัวอย่างเช่น การกำเนิดค่าแรนดัม ที่ใช้ได้

ดีกับการจำลองแบบเคออสติก (Stochastic simulation) แต่จะไม่เหมาะกับการใช้งานแบบการเข้ารหัสลับ(Cryptographic applications) เพราะค่าที่ออกมาสามารถคาดเดาได้ แต่อีกด้านหนึ่งการกำเนิดค่าแรนดัม ที่ใช้กับการใช้งานแบบเข้ารหัสลับ(Cryptographic applications) จะทำงานช้าจึงไม่เหมาะกับการใช้งานจำลองแบบมอลติแคร์โร(Monte Carlo simulations)ที่ต้องการความเร็วสูง เป็นต้น

การกำเนิดค่าแรนดัมถูกใช้งานอย่างแพร่หลาย เช่น เกมส์การเสี่ยงโชค การวิเคราะห์ทางวิทยาศาสตร์(มีการวิจัยที่เรียกว่า Monte Carlo simulations) การวิจัยทางการทหาร รวมถึงใช้ในซอฟแวร์บางตัว นอกจากนี้ยังเป็นส่วนสำคัญในการพัฒนาด้านการวิจัยทางด้านการเข้ารหัสลับ (Cryptographic Applications) ยกตัวอย่างเช่น

- Random session keys
- RSA prime factors
- Random numbers for DSS
- Zero-knowledge-proofs
- Challenge-response-protocols
- IV vectors

เป็นต้น

การทดสอบค่าแรนดัม (Random Number Test)

ในการทดสอบค่าจากเครื่องกำเนิดสัญญาณแรนดัมมีวิธีที่ใช้อยู่กัน 2 วิธี วิธีหนึ่งคือ Structural test และอีกวิธีคือการใช้รูปแบบฟังชั่นเชิงสถิติ (Statistical Test or Empirical Test) ซึ่งวิเคราะห์ผลจากเลขบิตซีเควนซ์ของเครื่องกำเนิดค่าแรนดัม โดยงานวิจัยนี้จะพูดถึงการวิเคราะห์เชิงสถิติเป็นหลัก ซึ่งวิธีนี้มีประโยชน์กับเครื่องกำเนิดค่าแรนดัมที่มีความเป็นอิสระ ดังนั้นจึงสามารถวิเคราะห์ได้ทั้งด้านฮาร์ดแวร์และซอฟแวร์

โดยลักษณะของค่าแรมดัมที่แท้จริงนั้นจะต้องไม่มีรูปแบบการวนซ้ำ (Unpattern) ซึ่งจะเกิดขึ้นก็ต่อเมื่อค่าเอ็นโทรปี(Entropy) ของซีเควนนั้นมีค่าสูงที่สุดหรือเท่ากับ 1

จากสมการเอ็นโทรปี

$$H(x) = \sum P(x_i) \log_2 (\frac{1}{P(x_i)}) \tag{2.2}$$

$$I(x) = \log_2 (\frac{1}{P(x)}) \quad bit / symbol \tag{2.3}$$

โดย $H(X)$:ค่าเอ็นโทรปี

$P(X)$:ความน่าจะเป็นของการเกิดบิต 0 หรือ 1

$I(X)$:ปริมาณข้อมูลที่รับเข้ามาในแต่ละสัญลักษณ์(Symbol)

สามารถสรุปได้ว่าความน่าจะเป็นของการเกิด 0 และ 1 จะต้องเท่ากันคือ $P(0) = P(1) = 1/2$ จึงจะทำให้ได้ค่าเอ็นโทรปีที่สูงสุด

ส่วนการวิเคราะห์เชิงสถิติจะเปรียบเทียบลักษณะของชุดเลขซีเควนซ์ที่แน่นอนกับเลขซีเควนซ์ทดสอบที่นำมาจากเครื่องกำเนิดค่าแรนดัม ถ้าเลขซีเควนซ์ที่นำมาทดสอบนั้นไม่ผ่านมาตรฐานจะสรุปว่าไม่ได้ค่าแรนดัมที่แท้จริงออกมา แต่ถ้าผ่านการทดสอบนี้ก็จะถือว่าเครื่องกำเนิดค่าแรนดัมชุดนั้นเป็นไปตามมาตรฐาน อย่างไรก็ตามวิธีนี้ก็เป็นวิธีทางสถิติผลที่ออกจะเป็นผลของความน่าจะเป็นเท่านั้นไม่ใช่ว่าจะถูกต้องแน่นอนเสมอไป

2.2 วิธีทดสอบค่าแรนดัมที่เป็นมาตรฐาน

ในเอกสารของมาตรฐานของ FIPS PUB 140-1 ได้อธิบายถึงการทดสอบการกำเนิดค่าแรนดัมเชิงสถิติ (Statistical random number generator tests) ว่าเป็นการทดสอบด้วยตัวเองเป็นส่วนสำคัญที่สามารถวัดความน่าเชื่อถือว่าค่าแรนดัมที่สร้างขึ้นมาสามารถยอมรับได้หรือไม่และเป็นส่วนประกอบที่สำคัญในการสร้างความปลอดภัยในระดับสูง โดยจะวัดจากค่าบิตสตรีม (Bit steam) จำนวน 20,000 บิต ที่ได้จากเครื่องกำเนิดสัญญาณแรนดัม ซึ่งมีวิธีการต่างๆ ดังนี้

2.2.1 การทดสอบความถี่ (Frequency Type Test)

วิธีนี้มีหลักการคือ นับเลข “1” จากข้อมูลอินพุต 20,000 บิตโดย ความน่าจะเป็นของการเกิดเลข “0” และ “1” จะเป็นอิสระต่อกันโดยที่ จำนวน “1” ที่นับได้จะอยู่ในช่วง $9,654 < X < 10,346$

2.2.2 การทดสอบแบบโปเกอร์ (Poker Test)

วิธีนี้มีหลักการคือ จะมีการแบ่งข้อมูลอินพุต 20,000 บิตเป็น 5000 ชุด โดยที่แต่ละชุดจะต้องไม่มีการโอเวอร์แลปกัน (Non-overlapping) และ แต่ละชุดจะมีช่วงความยาว (Length) 4 บิต โดยค่า X จะต้องอยู่ในช่วง $1.03 < X < 57.4$

$$X = \frac{16}{5000}\left(\sum_{i=0}^{15^m} n_i^2\right) - 5000 \qquad (2.4)$$

2.2.3 การทดสอบแบบรัน (Runs Type Test)

มีหลักการคือ นับจำนวนครั้งที่เกิดการซ้ำติดต่อของทั้ง "1" และ "0" จากข้อมูลทั้งหมด 20000 บิต แล้วเทียบจำนวนครั้งที่เกิดการซ้ำติดต่อกันโดยใช้ตารางที่ 2.1

Length of run (i)	Required interval
1	2267 – 2733
2	1079 – 1421
3	502 – 748
4	223 – 402
5	90 – 223
6+	90 – 223

ตารางที่ 2.1 มาตรฐาน FIP 140-1 สำหรับ Runs type test

ข้อมูลอินพุตจำนวน 20,000 บิต จะต้องมีจำนวนค่าที่เกิดซ้ำติดต่อกัน อยู่ในช่วงที่กำหนดไว้ในตาราง 2.1 ซึ่งเป็นมาตรฐาน FIP 140-1 สำหรับ Runs type test ในกรณีที่จำนวนค่าที่เกิดซ้ำติดต่อกันมากกว่า 6 จะให้ถือว่ามีค่าความยาวเป็น 6

2.2.4 การทดสอบแบบล็องเกสรัน (Longest Runs Test)

วิธีนี้มีหลักการคือ จะตรวจสอบว่าข้อมูลอินพุต จำนวน 20,000 บิต มีการเกิด "1" หรือ "0" ซ้ำติดต่อกันเกิน 26 บิตหรือไม่ ถ้าเกินจะถือว่าเป็นข้อมูลแรนดัมที่ไม่ได้มาตรฐาน

2.3.การทดสอบโดยการบีบอัดข้อมูลแบบ LZ78

การทดสอบความเป็นแรมดัมอีกวิธีหนึ่งคือการบีบอัดข้อมูล กล่าวคือถ้ามีความเป็นแรมดัมมากอัตราการบีบอัดข้อมูลจะต่ำ ซึ่งวิธีบีบอัดข้อมูลแบบ LZ78 ก็เป็นวิธีที่ง่ายต่อการทดสอบอีกวิธีหนึ่ง

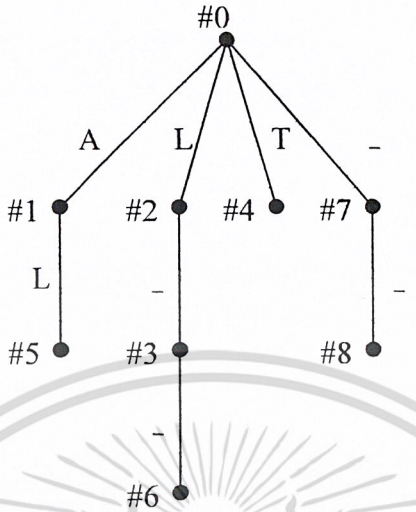
การบีบอัดข้อมูลแบบ LZ78 เป็นวิธีการบีบอัดข้อมูลโดยการสร้างพจนานุกรมที่บรรจุสตริงที่เคยเกิดขึ้นโดยไม่จำกัด โดยถ้าพบว่าสตริงที่ต้องการส่งเคยเกิดขึ้นแล้วจะแทนสตริงนี้ด้วยรหัสที่ชี้ตำแหน่งสตริงที่มีอยู่ในพจนานุกรม สำหรับขั้นตอนการเข้ารหัส LZ78 สามารถอธิบายได้ดังนี้

เอาท์พุตของการเข้ารหัสตัวอักษรทุกตัวจะอยู่ในรูปแบบ (#n,C) โดย #n คือค่าชี้ตำแหน่งของสตริงที่อยู่ในพจนานุกรม และ C คืออักษรที่ทำให้สตริงที่เกิดใหม่ไม่อยู่ในพจนานุกรมของลำดับการเข้ารหัส LZ78

1. กำหนด prefix string :<PS> หรืออาเรย์จำนวนหนึ่งสำหรับเก็บสตริงที่ยังไม่เคยปรากฏ โดยเริ่มต้นไม่มีค่าใดบรรจุอยู่
2. อ่านอักษรตัวแรกและให้เอาท์พุต <#,อักษรตัวแรก> ตำแหน่ง #0 หมายถึงอักษรหรือสตริงที่ตามหลังตัวชี้นี้ยังไม่เคยถูกสร้างเก็บไว้ในพจนานุกรม
3. อ่านอักษรตัวต่อไป (แทนด้วย C) ต่ออักษรตัวที่อ่านใหม่เข้ากับตัวอักษรที่อยู่ใน prefix string หรือ <PS>*C และตรวจสอบว่ามีในพจนานุกรมหรือไม่
   - 3.1 ถ้ามีให้ค่า <PS> = <PS>*C และวนกลับไปทำข้อ 3 ใหม่
   - 3.2 ถ้าไม่มี
     - 3.2.1 ให้เก็บ <PS>*C ไว้ในพจนานุกรม พร้อมกับให้รหัสระบุตำแหน่ง
     - 3.2.2 ให้เอาท์พุต <#n,C> โดย #n เป็นตำแหน่งของ <PS>ที่อยู่ในพจนานุกรม
4. ลบค่าใน <PS> และวนกลับมาทำข้อ 2 จนกระทั่งหมดตัวอักษร

การแทนรหัส LZ78 ด้วยกิ่งไม้

รหัสแบบLZ78 สามารถแทนด้วยต้นไม้หลายกิ่ง(multi-tree) โดยการสร้างกิ่งทุกครั้งหลังจากที่ส่งรหัส(#n,C) ออกไป ค่า #n จะแทนโนดที่ให้กำเนิดกิ่งมีอักษร (C) เป็นค่าของกิ่งและที่ปลายกิ่งจะแสดงรหัสที่ใช้เป็นตัวชี้ของอักษรหรือสตริงที่กำเนิดขึ้น การอ่านสตริงของหมายเลข #n ใดๆ จะอ่านจากรากจนถึงปลายกิ่งที่มีหมายเลขรหัสนั้นๆ จากตัวอย่างต้นไม้ถูกสร้างขึ้น โดยเริ่มต้นที่ราก มีหมายเลขโนดคือ #0 หมายถึงว่ากิ่งที่เกิดจากโนดรากนี้ยังไม่มีอยู่ในพจนานุกรม ลำดับแรกเมื่อส่ง<#0,A> แล้วกิ่ง A จะถูกสร้างจากราก #0 และมีปลายกิ่งเป็นโนด#1 และเมื่อส่ง <#0,L> กิ่ง L จะถูกสร้างขึ้นและมีหมายเลขโนด #2 และเมื่อรหัสส่งตัวที่ 3 คือ <#2,_> เนื่องจาก L มีอยู่ในพจนานุกรมแล้ว โดยมีหมายเลขคือ #2 ดังนั้นกิ่งของสัญญาณ _ จึงกำเนิดจากโนด #2 และมีหมายเลขปลายกิ่งคือ #3



รูปที่ 2.1 ต้นไม้ของรหัส LZ78

การถอดรหัส LZ78

เมื่อภาครับจะรับรหัสอินพุตที่อยู่ในรูปแบบ <#n,C> โดย C หมายถึงอักษรที่ทำให้สตริงไม่สามารถเข้าคู่กับสตริงในพจนานุกรม การถอดรหัสแทน #n ด้วยสตริงที่อยู่ในพจนานุกรม จากนั้นต่อสตริงด้วยอักษร C และสร้างสตริงใหม่นี้พร้อมทั้งหมายเลขตัวชี้ไว้ในพจนานุกรม

ลำดับ	อินพุต	พจนานุกรม	รหัส #n	เอาต์พุต
1.	<#0,A>	A	#1	A
2.	<#0,L>	L	#2	L
3.	<#2,L>	L_	#3	L_
4.	<#0,T>	T	#4	T
5.	<#1,L>	AL	#5	AL
6.	<#3,_>	L_ _	#6	L_ _
7.	<#0,_>	_	#7	_
8.	<#7,_>	_ _	#8	_ _

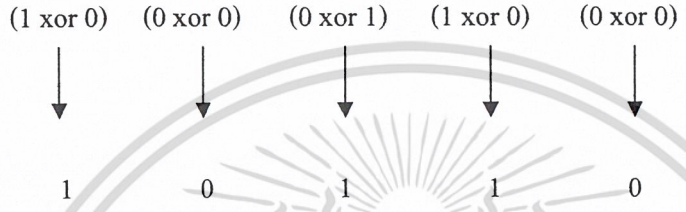
ตารางที่ 2.2 การถอดรหัสจากรูป 2.1

2.4 ไบนารีเดอริเวทีฟ (Binary Derivatives)

ไบนารี่เดอริเวทีฟเป็นวิธีการทดสอบค่าแรนดัมอย่างแท้จริง ในกรณีที่การบีบอัดข้อมูลไม่สามารถวัดผลที่แน่ชัดได้ สำหรับขั้นตอนการทดสอบแบบไบนารี่เดอริเวทีฟ สามารถอธิบายได้ดังนี้

1. ทำการเอ็กซ์ครูซีฟออร์ (Exclusive Or) ระหว่างค่าแรกกับตัวถัดไป

   1.1 ตัวอย่างเช่นเลขซีเควนซ์ 100100 ทำการเดอริเวทีฟ โดย



   จะได้ค่าผลของการเดอริเวทีฟ ครั้งที่1 คือ 10110

   1.2 ทำการเดอริเวทีฟ ครั้งต่อไปจากผลที่ได้จาก 1.1 สำหรับจำนวนครั้งในการเดอริเวทีฟขึ้นอยู่กับว่าต้องการผลที่ละเอียดมากแค่ไหน

2. หาค่าความน่าจะเป็นของ 1 ของการเดอริเวทีฟในแต่ละครั้ง โดย ค่าความน่าจะเป็น(P) ของ 1 = จำนวนบิต 1 ทั้งหมด หารด้วย จำนวนบิตทั้งหมด

3. หาค่า r จาก

$$r = P(Max) - P(Min) \qquad (2.2)$$

Attribute	Patterned Function	"Random" Function
p(0)	variable	close to 0.5
$\sum_{i=0}^{n} \frac{p(i)}{(n+1)}$	Low	close to 0.5
p(max)-p(min)	high	low

ตาราง 2.3 วิธีทดสอบแบบ ไบนารีเดอริเวทีฟ

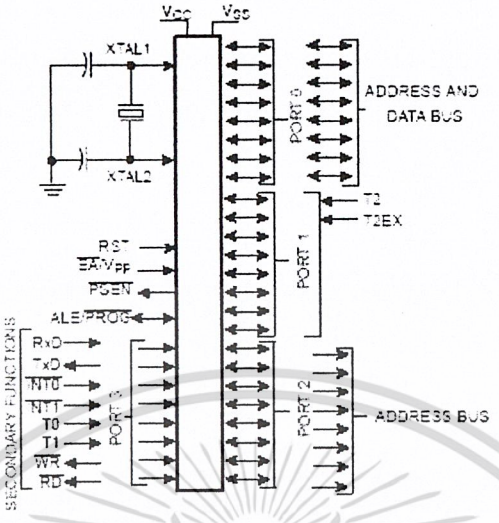
2.5 ไมโครคอนโทลเลอร์ (Microcontroller)

ไมโครคอนโทรลเลอร์ 8051 ที่จะกล่าวต่อไปนี้นั้นจะอ้างอิงกับบริษัท Atmel เป็นหลัก ข้อดีของไมโครคอนโทรลเลอร์ชนิดนี้นั้น ก็มีหลายประการด้วยกัน เช่น

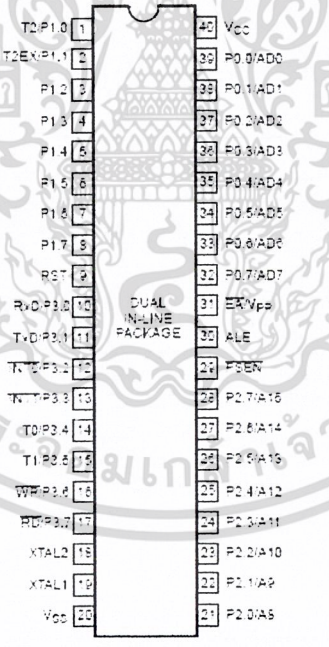
1. หน่วยความจำภายในเป็นแบบแฟลช (Flash memory)สามารถที่จะลบและเขียนใหม่ได้เป็นพันครั้ง
2. มีส่วนป้องกันการคัดลอกหรืออ่านข้อมูลของหน่วยความจำภายใน
3. ในบางเบอร์ของไมโครคอนโทรลเลอร์ก็สามารถทำการโปรแกรมได้ โดยที่ไม่ต้องถอดตัวไมโครคอนโทรลเลอร์ ออกมาจากตัวบอร์ดเพื่อทำการโปรแกรมใหม่ ซึ่งการโปรแกรมแบบนี้เรียกว่าการโปรแกรมแบบไอเอสพี (ISP : In System Programming)
4. ต้นทุนและระยะเวลาในการพัฒนาลดลง อีกทั้งยังสามารถเลือกใช้งานได้หลากหลาย เพราะทางบริษัทได้ผลิต ตัวไมโครคอนโทรลเลอร์ออกมาให้เลือกใช้งานหลายเบอร์

คุณสมบัติของไมโครคอนโทรลเลอร์ตระกูล 8051

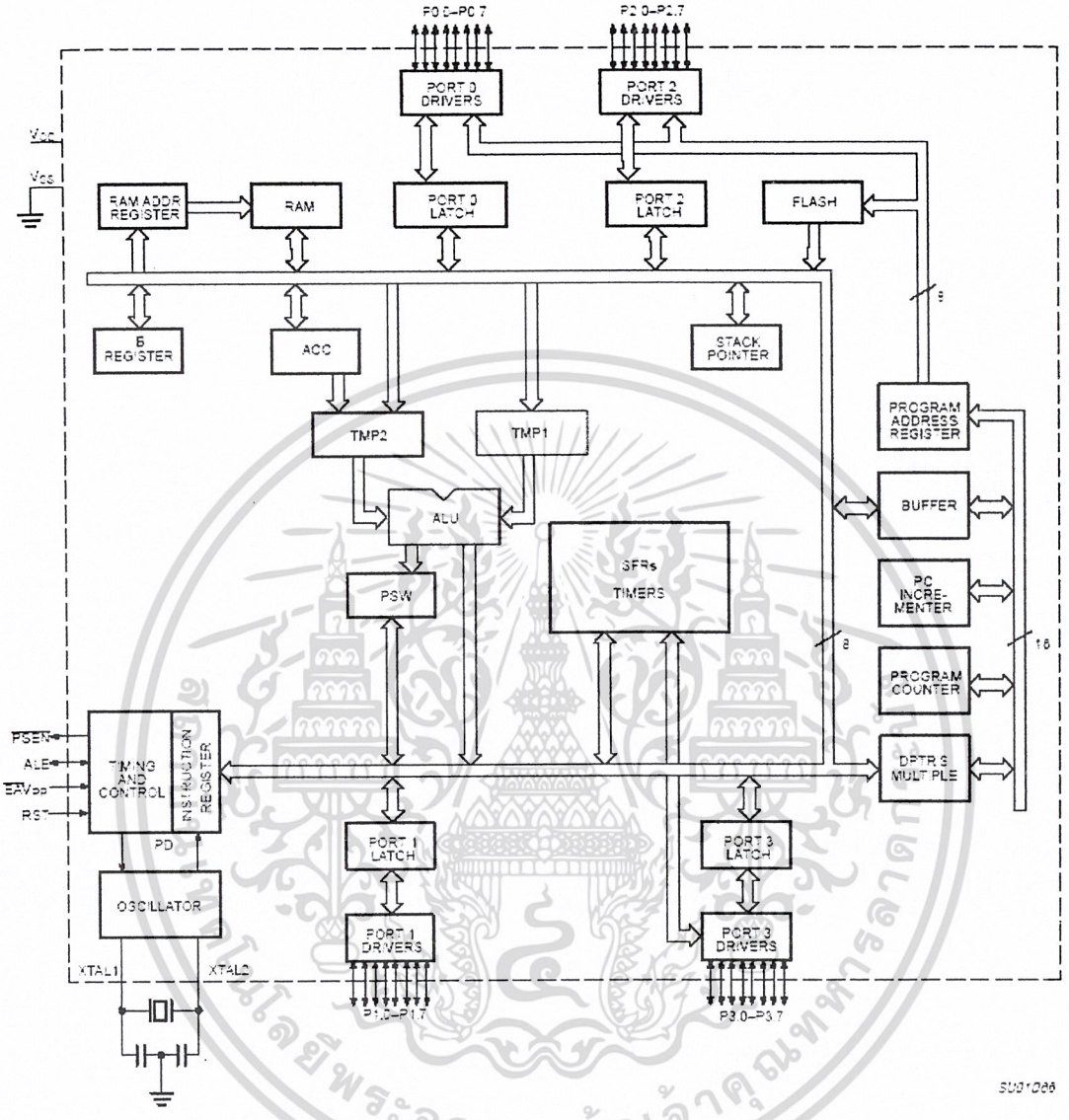
1. เป็นไมโครคอนโทรลเลอร์ขนาด 8 บิต
2. หน่วยความจำโปรแกรมเป็นแบบแฟลช สามารถลบและเขียนใหม่ได้เป็นพันครั้ง
3. มีหน่วยความจำภายในเป็นแบบแรม และเป็นแบบ อีอีพรอมในบางเบอร์ ซึ่งไม่ต้องกลัวว่าข้อมูลจะหาย เมื่อไม่มีไฟมาเลี้ยงวงจร
4. พอร์ตใช้งานเป็นแบบสองทิศทาง สามารถใช้งานได้ทั้งเป็นแบบอินพุตและเอาต์พุต
5. สามารถขยายหน่วยความจำโปรแกรมภายนอกได้สูงสุด 64 กิโลไบต์
6. มีไทเมอร์/เคาเตอร์ขนาด 16 บิต อย่างน้อย 2 ตัว
7. มีพอร์ตสื่อสารอนุกรมแบบฟูลดูเพล็กซ์ และแบบเอสพีไอ (SPI)
8. รองรับแหล่งกำเนิดอินเตอร์รัปต์ได้ 6 ประเภท
9. มีวอตช์ด็อกไทเมอร์ในตัว ใน AT89Sxx Series
10. มีวงจรกำเนิดสัญญาณนาฬิกาอยู่ภายใน



รูปที่ 2.2 โครงสร้างทางสถาปัตยกรรมของไมโครคอนโทรลเลอร์ตระกูล 8051



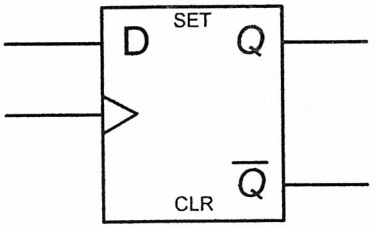
รูปที่ 2.3 การจัดขามาตรฐานของไมโครคอนโทรลเลอร์ตระกูล 8051



รูปที่ 2.4 รายละเอียดโครงสร้างหลักของไมโครคอนโทรลเลอร์ตระกูล 8051

2.6 วงจรดีฟลิปฟลอป (D flip-flop)

ฟลิปฟลอปมี เอาต์พุตคงที่อยู่ 2 สภาวะ สำหรับ เอาต์พุต ทั้ง สองจะตั้งเงื่อนไขว่า เอาต์พุตหนึ่งจะเป็นคอมพลีเม้นท์ (Complement) ของอีก เอาต์พุตหนึ่งจนกว่าจะมี อินพุตพัลส์ มากระตุ้นถึงจะทำให้มีการเปลี่ยนแปลง สภาวะไป ดี ฟลิปฟลอป จะมี 1 อินพุต คือ อินพุตดี และมี 2 เอาต์พุต คือ คิว (Q) และ คิวคอมพลีเม้นท์ ($\overline{Q}$)



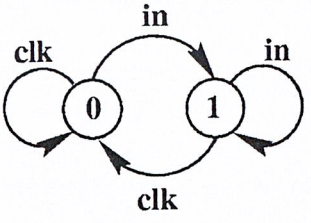
รูปที่ 2.5 สัญลักษณ์ของ ดี ฟลิปฟลอป

การทำงานของดีฟลิปฟลอป

กล่าวคือในสภาวะที่สัญญาณนาฬิกา (CK) เป็น ลอจิก (Logic) 0 ค่าของ ดี (D)จะเป็น ลอจิก 1หรือ ลอจิก 0 ก็ตาม เอาต์พุต ของ ดี ฟลิปฟลอป จะไม่มีการเปลี่ยนสภาวะ คือจะคงสภาวะตัวเดิม แต่ถ้า สัญญาณนาฬิกา เปลี่ยนจาก ลอจิก 0 เป็น ลอจิก 1 เอาต์พุต ของ ดี ฟลิปฟลอป จะเปลี่ยนสภาวะตาม ตารางความจริง (Truth table) ของ ดี ฟลิปฟลอป คือ เมื่อ อินพุต ดี เป็น ลอจิก 0 เอาต์พุต จะมีค่าเป็น ลอจิก 0 ถ้า อินพุต ดี เป็น ลอจิก 1 เอาต์พุต จะมีค่าเป็น ลอจิก 1

Clock	D	Q_{n+1}
H	0	0
H	1	1
0	x	Q_n

ตารางที่ 2.4 ตารางความจริง (Truth table) ของ ดี ฟลิปฟลอป

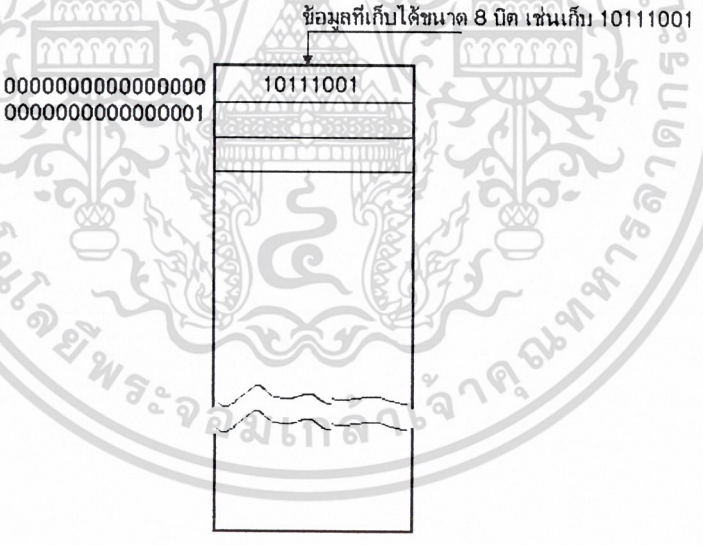


รูปที่ 2.6 ไดอะแกรมแสดงสถานะการทำงาน (State Diagram) ของ ดี ฟลิปฟลอป

2.7 หน่วยความจำ (Memory)

เป็นแผงวงจรซึ่งประกอบด้วยกลุ่มของวงจรไฟฟ้าขนาดเล็กจำนวนมาก ทำงานได้โดยต้องมีกระแสไฟฟ้าไหลวนเลี้ยงวงจรอยู่ตลอดเวลา ทำหน้าที่เก็บข้อมูล ในขณะที่คอมพิวเตอร์กำลังทำงานอยู่ แผงวงจรไฟฟ้านี้ทำด้วยสารกึ่งตัวนำที่สามารถรับกระแสไฟฟ้าได้ รวดเร็วมาก แต่ไม่สามารถเก็บข้อมูลหรือคำสั่งได้เลยเมื่อปิดเครื่องหรือไม่มีกระแสไฟฟ้าเข้าสู่วงจร แผงวงจรเล็กๆ นี้เรียกว่า ชิป (chip) ซึ่งมีขนาดเล็กมาก

โครงสร้างการจัดเก็บข้อมูลจัดเก็บสถานะซึ่งแทนด้วยเลขไบนารี โดยมีการกำหนดตำแหน่งที่เก็บที่เรียกว่า แอดเดรส โดยทั่วไปจัดโครงสร้างของหน่วยความจำให้มีความกว้างขนาด 8 บิต และตำแหน่งแอดเดรสบอกขนาดของแรม (RAM) ทั้งหมด เช่น ถ้าแรม มีขนาด 64 กิโลไบต์ (64 k) ก็หมายถึงขนาดของแรมมีความกว้างขนาด 8 บิต หรือ 1 ไบต์ และมีตำแหน่งที่เก็บได้เท่ากับ 65536 ตำแหน่ง (2 ยกกำลัง 16) โดยมีแอดเดรสกำหนดตำแหน่งทั้งหมด 16 บิต

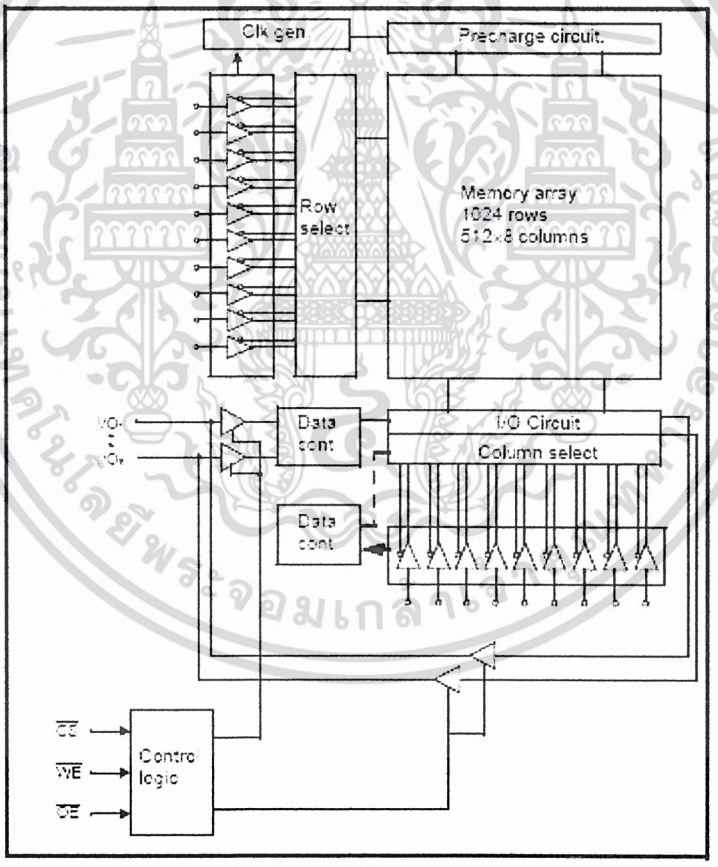


รูปที่ 2.7 โครงสร้างของหน่วยความจำ

แรม (RAM : Random Access Memory) คือเนื้อที่ที่ใช้ในการเก็บข้อมูลและคำสั่งในขณะที่เครื่องทำงานอยู่ มีสภาวะเป็น โวเลต์ไทล์ (volatile) คือไม่สามารถเก็บข้อมูลหรือชุดคำสั่งใดเมื่อไม่มีกระแสไฟฟ้าเข้าสู่วงจร แบ่งออกเป็น 2 ประเภทคือ ดีแรม (DRAM)และ เอสแรม (SRAM)

ดีแรม (DRAM : Dynamic Random Access Memory) จะทำการเก็บข้อมูลในตัวเก็บประจุ (Capacitor) ซึ่งจำเป็นจะต้องมีการรีเฟรช (refresh) เพื่อ เก็บข้อมูลให้คงอยู่ โดยการรีเฟรชนี้ ทำให้เกิดการหน่วงเวลาขึ้นในการเข้าถึงข้อมูล และก็เนื่อง จากดีแรมต้องรีเฟรชตัวเองอยู่ตลอดเวลานี้เอง จึงเป็นเหตุให้ได้ชื่อว่า ไดนามิค (Dynamic RAM)

เอสแรม (SRAM : Static Random Access Memory) จะต่างจากดีแรม ตรงที่ว่าดีแรมจะต้องทำการรีเฟรชข้อมูลอยู่ตลอดเวลา แต่ในขณะที่เอสแรมจะเก็บข้อมูลนั้นๆ ไว้ และจะไม่ทำการรีเฟรชโดยอัตโนมัติ ซึ่งมันจะทำการรีเฟรชก็ต่อเมื่อ สั่งให้มันรีเฟรชเท่านั้น ซึ่งข้อดีของเอสแรม ก็คือความเร็ว ซึ่งเร็วกว่าดีแรมปกติมาก แต่ก็ด้วยราคาที่สูงกว่ามาก จึงเป็นข้อด้อยของเอสแรม เช่นกัน



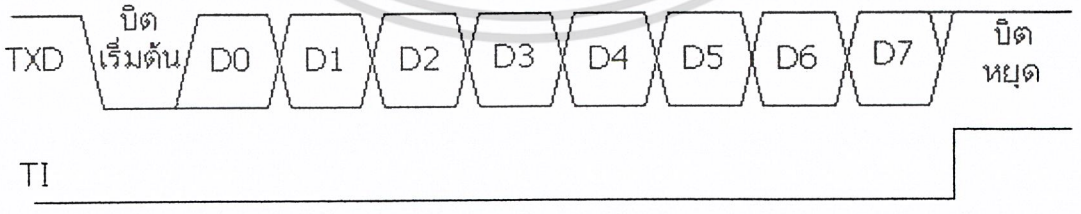
รูปที่ 2.8 ไดอะแกรมการทำงานของเอสแรมขนาด 512Kx8 บิต

Pin Name	Function
$\overline{WE}$	Write Enable Input
$\overline{CS}$	Chip Select Input
$\overline{OE}$	Output Enable Input
A0~A18	Address Inputs
I/O1~I/O8	Data Inputs/Outputs
Vcc	Power
Vss	Ground

ตารางที่ 2.5 สัญญาลักษณ์ต่างๆ ของไอซีเอสแรม

2.8 การติดต่อทางพอร์ตอนุกรมของไมโครคอนโทรลเลอร์ โหมด 1

ลักษณะของการติดต่อทางพอร์ตอนุกรมของไมโครคอนโทรลเลอร์ โหมด 1 จะเป็นการรับและส่งข้อมูลขนาด 10 บิต สามารถใช้ในการติดต่อสื่อสารอนุกรมกับมาตรฐานของ อาร์เอส-232ซี (RS-232C) ของไมโครคอมพิวเตอร์ได้ ซึ่งข้อมูลอนุกรม 10 บิต จะเข้ามาทางขา อาร์เอ็กซ์ดี (RXD) และ ส่งข้อมูลออกแบบอนุกรมทางขา ทีเอ็กซ์ดี (TXD) โดยจะประกอบด้วย 1 บิตแรกเป็นบิตเริ่มต้น (Start bit ค่า 0) 8 บิตต่อมาจะเป็นบิตของข้อมูล (การรับ/ส่งจะเริ่มจากบิตต่ำก่อน และ บิตหยุดอีก 1 บิต (Stop bit ค่า 1) ส่วนทางด้านรับข้อมูลจะนำค่าบิตหยุด (Stop bit) ที่รับเข้ามาได้นำไปเก็บไว้ในบิตอาร์บี 8 (RB8) ที่อยู่ในรีจิสเตอร์เอสซีโอเอนท์ (SCON) และความเร็วของการส่งข้อมูลในโหมด 1 จะขึ้นอยู่กับบิตเอสเอ็มโอดี (SMOD) ที่อยู่ในรีจิสเตอร์พีซีโอเอนท์ (PCON) และอัตราโอเวอร์โฟลว์ของไทเมอร์ 1 ซึ่งอัตราการรับส่งข้อมูลในโหมดนี้สามารถกำหนดได้ตามต้องการ



รูปที่ 2.9 สัญญาณการส่งข้อมูลในโหมด 1

ตัวอย่างโปรแกรมส่งข้อมูลผ่านพอร์ตอนุกรมโดยใช้ไมโครคอนโทรลเลอร์ กำหนดอัตราบอดที่ 9600 บิตต่อวินาที

```
;*****************************************************************************
;*** Transmit Data Serial (Mode1) 9600 Baud to Microcomputer ***
;*****************************************************************************
DIP_0   BIT     P3.7    ;เป็นไดเร็กทีพที่ใช้กำหนดค่าดิฟสวิตซ์ DIP_0 แทน P3.7
DIP_1   BIT     P3.5    ;เป็นไดเร็กทีพที่ใช้กำหนดค่าดิฟสวิตซ์ DIP_1 แทน P3.5
DIP_2   BIT     P3.4    ;เป็นไดเร็กทีพที่ใช้กำหนดค่าดิฟสวิตซ์ DIP_2 แทน P3.4
DIP_3   BIT     P3.3    ;เป็นไดเร็กทีพที่ใช้กำหนดค่าดิฟสวิตซ์ DIP_3 แทน P3.3
SW_LOAD BIT P3.2        ;เป็นไดเร็กทีพที่ใช้กำหนดค่า SW_LOAD แทน P3.2
CR EQU 0DH              ;Carriage return
LF EQU 0AH              ;Line feed
;*****************************************************
;*** กำหนดค่าเริ่มต้นของรีจิสเตอร์ ****
;*****************************************************
ORG 0000H               ;เริ่มต้นที่แอดเดรส 0000H
MOV P1,#0FFH            ;กำหนดให้พอร์ต P1 เป็นสถานะสูง (LED ต่อแบบแอโนดร่วม)
MOV P3,#0FFH            ;กำหนดให้พอร์ต P3 เป็นสถานะสูง(อินพุตพอร์ต)
MOV 20H,#00H            ;กำหนดค่าคงที่ 00H ให้กับแอดเดรส 20H
MOV PCON,#00000000B ;กำหนดอัตราการรับส่งข้อมูล
MOV SCON,#01000000B ;กำหนดโหมดการรับส่งพอร์ตอนุกรมในโหมด1
MOV TMOD,#20H           ;กำหนดใช้ไทม์เมอร์ 1 โหมด 2 Auto reload
MOV TL1,#0FDH           ;กำหนด Baud ret 9600 BPS
MOV TH1,#0FDH           ;กำหนด Baud ret 9600 BPS
SETB TR1                ;เริ่มทำงานของโหมดตั้งเวลาไทม์เมอร์1
;*****************************************************
;*** ส่งข้อมูลที่อยู่ในในตาราง Look up table ****
;*****************************************************
MOV DPTR,#SHOW          ;กำหนดค่าของฐานแอดเดรสให้กับรีจิสเตอร์ DPTR
MOV R0,#00H             ;กำหนดให้รีจิสเตอร์ R0 = 00H
```

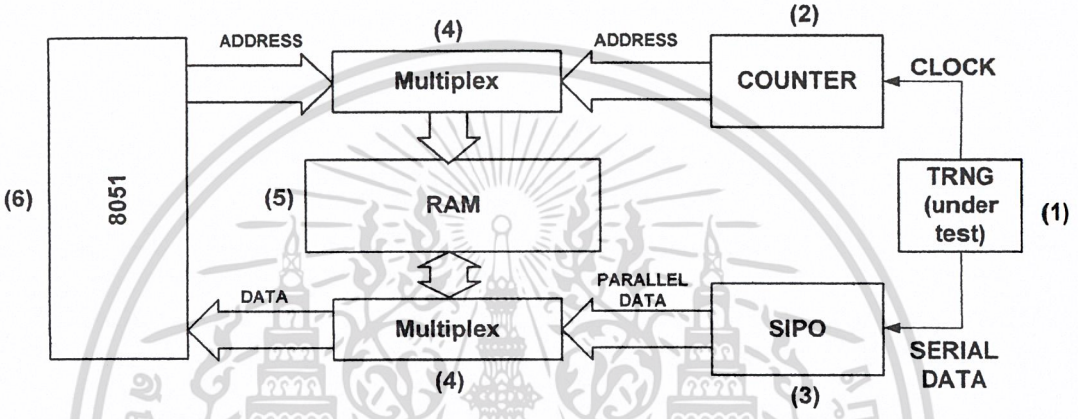
```
SEND_SHOW: MOV A,R0 ;นำค่าข้อมูลในรีจิสเตอร์ R0 เก็บไว้ที่รีจิสเตอร์ A
MOVC A,@A+DPTR    ;ทำการเปิดตารางข้อมูล แล้วนำข้อมูลที่ได้มาเก็บไว้ที่รีจิสเตอร์ A
MOV SBUF,A        ;นำข้อมูลในรีจิสเตอร์ A ไปเก็บไว้ที่รีจิสเตอร์ SBUF
JNB TI,$          ;ถ้าบิต TI ไม่ถูกเซตจะวนส่งข้อมูลจนกว่าจะครบ หลังจากนั้นจะเซตที่บิต
CLR TI            ;เคลียรบิต TI เพื่อเตรียมส่งข้อมูลในไบต์ต่อไป
INC R0            ;เพิ่มค่าข้อมูลในรีจิสเตอร์ R0
JNZ SEND_SHOW     ;ถ้าข้อมูลไม่เท่ากับ 00H ให้กระโดดกลับไปส่งข้อมูลใหม่ที่เลเบล LOOP
;*********************************************************
;*** ตรวจสอบการส่งข้อมูลที่กำหนดโดยดิฟสวิตซ์ ****
;*********************************************************
KEY_CHK: JB SW_LOAD,$ ;ถ้าสวิตซ์ SW_LOAD เป็น "1" ให้ทำคำสั่งที่บรรทัดเดิม
ACALL DELAY       ;จากคำสั่งที่ผ่านมาถ้ามีการกดสวิตซ์จะเรียกโปรแกรมย่อยหน่วงเวลา
JB SW_LOAD,$      ;ทำการตรวจสอบสถานะของสวิตซ์อีกครั้งหนึ่ง
MOV C,DIP_0       ;DIP SWITCH 0 นำค่าสถานะของ P3.7 มาเก็บไว้ที่บิตทด
MOV 20H.0,C       ;นำข้อมูลในบิตทดมาเก็บไว้ที่แอดเดรส 20H บิต 0 หรือ ตำแหน่ง 00H
MOV C,DIP_1       ;DIP SWITCH 1 นำค่าสถานะของ P3.5 มาเก็บไว้ที่บิตทด
MOV 20H.1,C       ;นำข้อมูลในบิตทดมาเก็บไว้ที่แอดเดรส 20H บิต 1 หรือ ตำแหน่ง 01H
MOV C,DIP_2       ;DIP SWITCH 2 นำค่าสถานะของ P3.4 มาเก็บไว้ที่บิตทด
MOV 20H.2,C       ;นำข้อมูลในบิตทดมาเก็บไว้ที่แอดเดรส 20H บิต 2 หรือ ตำแหน่ง 02H
MOV C,DIP_3       ;DIP SWITCH 3 นำค่าสถานะของ P3.3 มาเก็บไว้ที่บิตทด
MOV 20H.3,C       ;นำข้อมูลในบิตทดมาเก็บไว้ที่แอดเดรส 20H บิต 3 หรือ ตำแหน่ง 03H
MOV A,20H         ;นำข้อมูลตำแหน่งแอดเดรส 20H ไปเก็บไว้ที่รีจิสเตอร์ A
CPL A             ;กลับค่าข้อมูลในรีจิสเตอร์ A (LED ต่อแบบแอโนดร่วม)
MOV P1,A          ;นำค่าข้อมูลในรีจิสเตอร์ A ไปแสดงผลที่พอร์ต P1
;*********************************************************
;*** ส่งข้อมูลที่กำหนดโดยดิฟสวิตซ์ ****
;*********************************************************
CPL A             ;กลับค่าข้อมูลในรีจิสเตอร์ A ทุกบิต
CJNE A,#00H,KEY1  ;เปรียบเทียบข้อมูลระหว่างค่าที่กำหนดกับค่าดิฟสวิตซ์ ถ้าไม่เท่ากัน
                  ;กระโดดไปที่ KEY1
```

```
MOV A,#30H          ;จากคำสั่งที่ผ่านมาถ้าเท่ากัน จะให้ค่าในรีจิสเตอร์ A = 30H (ASCII = 0)
SJMP SEND           ;กระโดดไปทำการส่งข้อมูลที่เลเบล SEND
KEY1: CJNE A,#01H,KEY2 ;เปรียบเทียบข้อมูลระหว่างค่าที่กำหนดกับค่าดิฟสวิตซ์ ถ้าไม่เท่ากัน
                    ;กระโดดไปที่ KEY2
MOV A,#31H          ;จากคำสั่งที่ผ่านมาถ้าเท่ากัน จะให้ค่าในรีจิสเตอร์ A = 31H (ASCII = 1)
SJMP SEND           ;กระโดดไปทำการส่งข้อมูลที่เลเบล SEND
KEY2: CJNE A,#02H,KEY3 ;เปรียบเทียบข้อมูลระหว่างค่าที่กำหนดกับค่าดิฟสวิตซ์ ถ้าไม่เท่ากัน
                    ;กระโดดไปที่ KEY3
MOV A,#32H          ;จากคำสั่งที่ผ่านมาถ้าเท่ากัน จะให้ค่าในรีจิสเตอร์ A = 32H (ASCII = 2)
SJMP SEND           ;กระโดดไปทำการส่งข้อมูลที่เลเบล SEND
KEY3: CJNE A,#03H,KEY4 ;เปรียบเทียบข้อมูลระหว่างค่าที่กำหนดกับค่าดิฟสวิตซ์ ถ้าไม่เท่ากัน
                    ;กระโดดไปที่ KEY4
MOV A,#33H          ;จากคำสั่งที่ผ่านมาถ้าเท่ากัน จะให้ค่าในรีจิสเตอร์ A = 33H (ASCII = 3)
SJMP SEND           ;กระโดดไปทำการส่งข้อมูลที่เลเบล SEND
KEY4: CJNE A,#04H,KEY5 ;เปรียบเทียบข้อมูลระหว่างค่าที่กำหนดกับค่าดิฟสวิตซ์ ถ้าไม่เท่ากัน
                    ;กระโดดไปที่ KEY5
MOV A,#34H          ;จากคำสั่งที่ผ่านมาถ้าเท่ากัน จะให้ค่าในรีจิสเตอร์ A = 34H (ASCII = 4)
SJMP SEND           ;กระโดดไปทำการส่งข้อมูลที่เลเบล SEND
KEY5: CJNE A,#05H,KEY6 ;เปรียบเทียบข้อมูลระหว่างค่าที่กำหนดกับค่าดิฟสวิตซ์ ถ้าไม่เท่ากัน
                    ;กระโดดไปที่ KEY6
MOV A,#35H          ;จากคำสั่งที่ผ่านมาถ้าเท่ากัน จะให้ค่าในรีจิสเตอร์ A = 35H (ASCII = 5)
SJMP SEND           ;กระโดดไปทำการส่งข้อมูลที่เลเบล SEND
KEY6: CJNE A,#06H,KEY7 ;เปรียบเทียบข้อมูลระหว่างค่าที่กำหนดกับค่าดิฟสวิตซ์ ถ้าไม่เท่ากัน
                    ;กระโดดไปที่ KEY7
MOV A,#36H          ;จากคำสั่งที่ผ่านมาถ้าเท่ากัน จะให้ค่าในรีจิสเตอร์ A = 36H (ASCII = 6)
SJMP SEND           ;กระโดดไปทำการส่งข้อมูลที่เลเบล SEND
KEY7: CJNE A,#07H,KEY8 ;เปรียบเทียบข้อมูลระหว่างค่าที่กำหนดกับค่าดิฟสวิตซ์ ถ้าไม่เท่ากัน
                    ;กระโดดไปที่ KEY8
MOV A,#37H          ;จากคำสั่งที่ผ่านมาถ้าเท่ากัน จะให้ค่าในรีจิสเตอร์ A = 37H (ASCII = 7)
SJMP SEND           ;กระโดดไปทำการส่งข้อมูลที่เลเบล SEND
```

```
KEY8: CJNE A,#08H,KEY9 ;เปรียบเทียบข้อมูลระหว่างค่าที่กำหนดกับค่าดิฟสวิตซ์ ถ้าไม่เท่ากัน
                       ;กระโดดไปที่ KEY9
MOV A,#38H             ;จากคำสั่งที่ผ่านมาถ้าเท่ากัน จะให้ค่าในรีจิสเตอร์ A = 38H (ASCII = 8)
SJMP SEND              ;กระโดดไปทำการส่งข้อมูลที่เลเบล SEND
KEY9: CJNE A,#09H,SEND ;เปรียบเทียบข้อมูลระหว่างค่าที่กำ หนดกับค่าดิฟสวิตซ์ ถ้าไม่เท่ากัน
                       ;กระโดดไปที่ SEND
MOV A,#39H             ;จากคำสั่งที่ผ่านมาถ้าเท่ากัน จะให้ค่าในรีจิสเตอร์ A = 39H (ASCII = 9)
SJMP SEND              ;กระโดดไปทำการส่งข้อมูลที่เลเบล SEND
SEND: MOV SBUF,A       ;นำค่าที่ได้จากรีจิสเตอร์ A (ดิฟสวิตซ์ เป็น ASCII)เก็บไว้ที่รีจิสเตอร์
                       ;SBUF
JNB TI,$               ;ถ้าบิต TI ไม่ถูกเซตจะวนส่งข้อมูลจนกว่าจะครบ หลังจากนั้นจะเซตที่บิต
                       ;TI
CLR TI                 ;เคลียรบิต TI เพื่อเตรียมส่งข้อมูลในไบต์ต่อไป
JNB SW_LOAD,$          ;ถ้าไม่ปล่อยมือจากคีย์สวิตซ์ให้ทำที่บรรทัดเดิม
SJMP KEY_CHK           ;กระโดดวนกลับไปตรวจสอบค่าคีย์ใหม่ที่เลเบล KEY_CHK
;***************************************
;******** DELAY DEBOUNCE ********
DELAY: MOV R1,#90H ;ส่วนของโปรแกรมย่อยหน่วงเวลาเพื่อทดสอบว่าเป็นการกดสวิตช์จริง
                   ;หรือไม่
DELAY_1: MOV R2,#0FFH
DJNZ R2,$
DJNZ R1,DELAY_1
RET
;*****************************************************************************
;******** ข้อมูลที่จะส่งให้กับไมโครคอมพิวเตอร์ ********
;*****************************************************************************
SHOW: DB CR,LF," MCS-51 LAB :: HOBBY ELECTRONICS ",CR
DB CR,LF," Serial Port Program ",CR,LF
DB 00H
END
```

บทที่ 3
การออกแบบโครงงาน

ภาพรวมของโครงงานแสดงได้ดังต่อไปนี้



รูปที่ 3.1 โครงสร้างของอุปกรณ์ทดสอบสัญญาณแรนดอม

จากรูปที่ 3.1 แสดงโครงสร้างอุปกรณ์ทดสอบสัญญาณแรนดอม อธิบายได้ดังนี้

1.วงจรกำเนิดค่าแรนดัม (Random Number Generator circuit) จะส่งข้อมูลอนุกรม(Serial Data)และสัญญาณนาฬิกา (Clock)

2.วงจรนับ (Counter) จะรับสัญญาณนาฬิกา (Clock)จากวงจรกำเนิดค่าแรนดัมเพื่อใช้ในการ กำหนดค่าแอดเดรส

3.วงจรเอสไอพีโอ (SIPO) จะรับข้อมูลอนุกรม(Serial Data) จากวงจรกำเนิดค่าแรนดัมเพื่อใช้ในการเขียนข้อมูลลงแรมในแต่ละแอดเดรสที่ถูกกำหนดโดยวงจรนับ

4.วงจรมัลติเพล็กซ์(Multiplex)จะใช้ในการสลับโหมดการทำงาน โดยเมื่อแรมถูกเขียนข้อมูลจนเต็มทุกแอดเดรสแล้วจะมีการสลับการทำงานเพื่อใช้เป็นโหมดการอ่านข้อมูลจากแรมแทน

5.แรม(RAM) จะใช้ในการเก็บข้อมูลที่ส่งมาจากวงจรกำเนิดค่าแรมดัม โดยจะเก็บเอาไว้ในแต่ละแอดเดรสของแรม

6. ไมโครคอนโทลเลอร์ 8051 จะทำการอ่านข้อมูลจากแรมโดยตรง แล้วส่งข้อมูลที่อ่านได้ไปยัง พอร์ตอนุกรมเข้าคอมพิวเตอร์เพื่อใช้ในการทดสอบค่าแรนดัม (Random Number Test) โดยโปรแกรมทดสอบค่าแรมดัมต่อไป

3.1 หน่วยความจำ 2 ทาง (Two port RAM)

เป็นการใช้งาน แรม ร่วมกันโดยจะมีการสลับการทำงาน 2 โหมด ระหว่าง

1. การเขียนข้อมูลลง แรม (Write Data) โดยผ่านการทำงานของวงจรนับ (counter) และ เอสไอพีโอ (SIPO)
2. การอ่านข้อมูลจาก แรม (Read Data) โดยใช้ ไมโครคอนโทลเลอร์ 8051 ควบคุมผ่าน แรม โดยตรงแล้วส่งข้อมูลที่อ่านได้ไปยังพอร์ตอนุกรมเข้าคอมพิวเตอร์เพื่อใช้ในการประมวลผล

3.2 โหมดการทำงานของวงจรเครื่องทดสอบสัญญาณแรนดัม

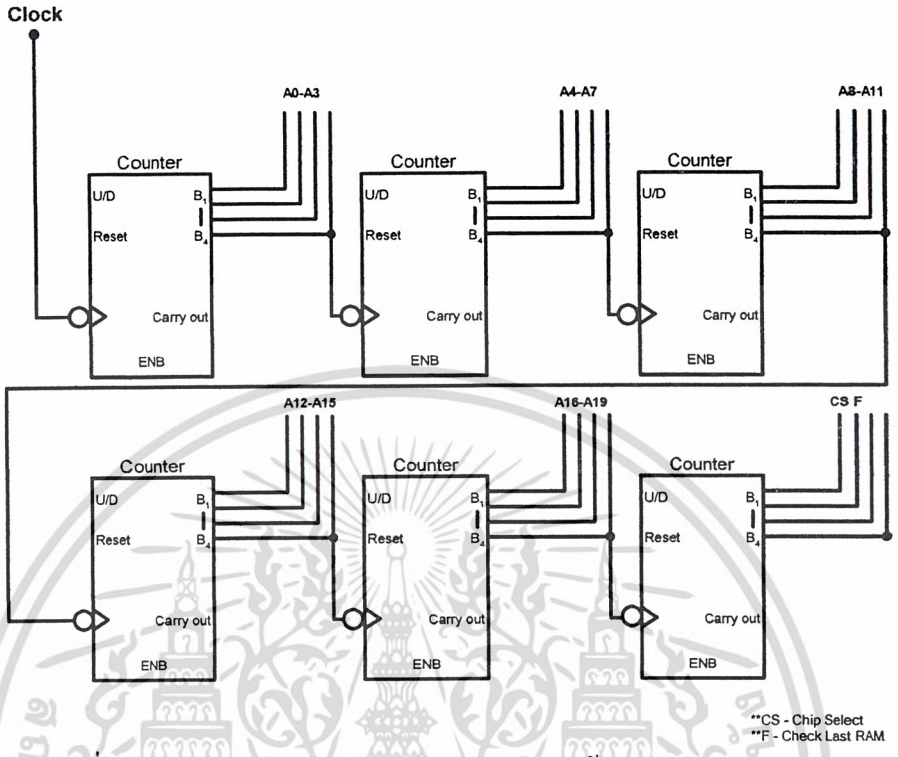
โหมดหลักในการทำงานของวงจรเครื่องทดสอบสัญญาณแรนดัมมีอยู่ 2 โหมดด้วยกันคือ โหมดการเขียนข้อมูลลงหน่วยความจำ (Write data) และ โหมดการอ่านข้อมูลจากหน่วยความจำ (Read data)

3.2.1 การเขียนข้อมูลลงหน่วยความจำ (Write data)

มีขั้นตอนหลัก 3 ขั้นตอนคือ

1. การใช้วงจรนับกำหนดค่าแอดเดรส (Address) ของแรม

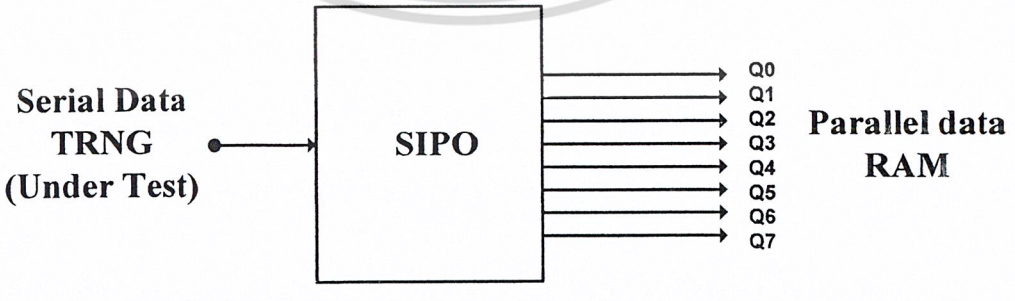
ในโหมดนี้การกำหนดค่าแอดเดรสของแรมทำได้โดย จากรูปที่ 3.2 วงจรนับจะสามารถกำหนดแอดเดรสของแรม 19 ขา (A0-A18) ได้โดยใช้ วงจรนับ(Counter) ทั้งหมด 6 ตัวซึ่ง วงจรนับ แต่ละตัวสร้างมาจาก วงจรนับ 4 บิต (4-Bit Binary counter)



รูปที่ 3.2 การกำหนดตำแหน่งแอดเดรสของแรมด้วยวงจรนับ

2. การใช้วงจร SIPO ในการรับข้อมูล (Data) จากวงจรกำเนิดค่าแรนดัม

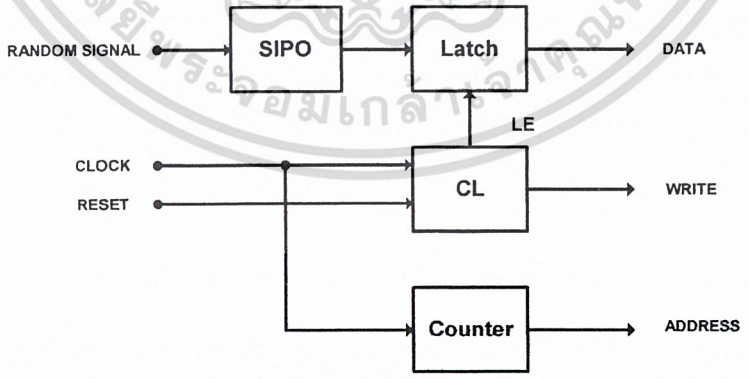
ในโหมดนี้การรับข้อมูลจากวงจรกำเนิดค่าแรนดัมที่ส่งข้อมูลแบบอนุกรมเข้ามา เราจะใช้วงจรเอสไอพีโอ (SIPO : Serial-in Parallel-out) ในการรับข้อมูลเป็นอนุกรมและส่งออกแบบขนานเป็น 8 บิต เพื่อส่งต่อไปให้ขาข้อมูล 8 บิต ของแรม



รูปที่ 3.3 ลักษณะอินพุตและเอาต์พุตของ SIPO 8 บิต

3. การใช้วงจรควบคุม (Control) ในการระบุค่าแอดเดรสและการรับข้อมูล (Data) ของแรม

ในโหมดนี้เราจะใช้วงจรควบคุมการแลตซ์ค่า (Combine logic : CL) หาช่วงจังหวะในการสร้าง พัลส์ (Pulse) ที่ขา $\overline{LE}$ และ $\overline{WR}$ จากไทมิ่งไดอะแกรม (Timing Diagram) แสดงการเขียนข้อมูล(Data)ลงแรม เพื่อใช้ในการเขียนข้อมูลลงแรม โดยมีหลักการคือ ข้อมูลที่เข้ามาเป็นแบบอนุกรม (Serial In) และ ข้อมูลจะออกมาแบบขนาน (Parallel Out) เนื่องจาก แต่ละ แอดเดรสของแรม ใช้ ข้อมูล 8 บิต เพราะ ฉะนั้นจึงต้อง แลตซ์ (Latch) ค่าข้อมูล เมื่อครบ 8 บิต ค้างเอาไว้ โดยการสร้าง พัลส์ ที่ขา $\overline{LE}$ เมื่อจบ พัลส์ที่ 8 ของข้อมูล หลังจากสร้าง พัลส์ที่ขา $\overline{LE}$ แล้วจะสร้าง พัลส์ที่ขา $\overline{WR}$ ตามมา และต้องสร้างก่อนที่จะเกิดการ แลตซ์ข้อมูล 8 บิต ชุดถัดไป เพื่อทำการเขียนข้อมูลลง แรม ลักษณะการทำงานคือ เมื่อ แอดเดรสเริ่มต้นเข้ามา (00H) ข้อมูล ที่เข้ามาจะถูก ควบคุมโดย CL เพื่อตรวจสอบว่า ข้อมูลครบ 8 บิต หรือยัง ถ้าครบแล้วจะไปกระตุ้นที่ขา $\overline{LE}$ ให้มีการ แลตซ์ข้อมูลค้างเอาไว้ และจะกระตุ้นที่ขา $\overline{WR}$ เพื่อให้เขียนข้อมูลลงไปที่ แอดเดรส นั้น แล้วจึงเริ่มกระบวนการใหม่โดยเพิ่มค่า แอดเดรส เป็น (01H) จากวงจรนับ(Counter) และทำการ แลตซ์ข้อมูล 8 บิต ชุดถัดไป ก่อนที่จะเขียนข้อมูลลงที่ แอดเดรส ใหม่นี้ โดยจะทำเช่นนี้ไปเรื่อยๆ จนกว่าจะครบ แอดเดรส ของ แรม (7FFFFH)



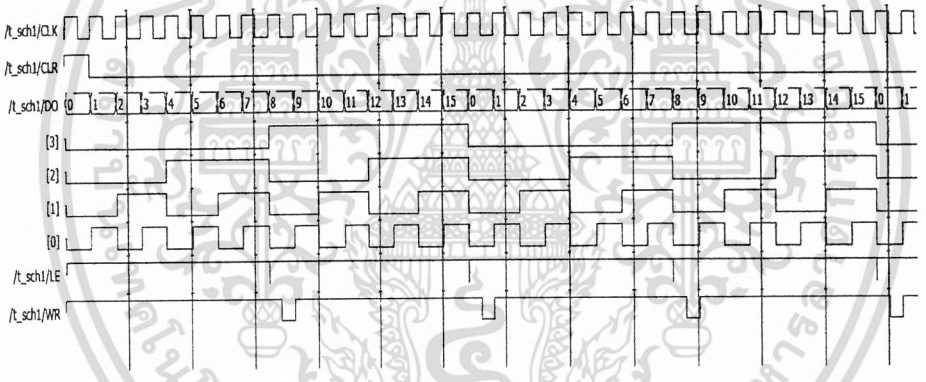
รูปที่ 3.4 โครงสร้างของส่วนควบคุม

3.3.2 การอ่านข้อมูลจากหน่วยความจำ (Read data)

เนื่องจาก ไมโครคอนโทรลเลอร์ 8051 สามารถต่อกับหน่วยความจำข้อมูลภายนอก (External data memory) ได้สูงสุด 64 กิโลไบต์ มี แอดเดรส ในช่วง 0000H – FFFFH โดยการใช้คำสั่ง movx ในการติดต่อกับหน่วยความจำข้อมูลภายนอก แล้วจากนั้นจึงนำข้อมูลจากหน่วยความจำภายนอกส่งเข้าเพื่อประมวลผลในเครื่องคอมพิวเตอร์โดยผ่านพอร์ต อนุกรม (Serial port)

แต่เนื่องจากอุปกรณ์ทดสอบสัญญาณแรนดอมต้องการหน่วยความจำข้อมูลภายนอกมากถึง 1 เมกกะไบต์ ซึ่งแบ่งออกเป็น แรม ขนาด 512 กิโลไบต์ จำนวน 2 ตัว ซึ่งเรียกว่าเทคนิคการทำเพจจิ่ง (Paging) และการเลือกแรม เข้ามาช่วย

3.3 ขั้นตอนการสร้างวงจรควบคุมการแลตซ์ค่าข้อมูลจากไทม์มิ่งไดอะแกรม

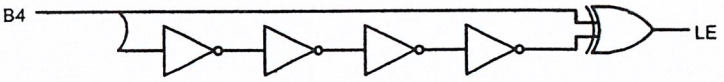


รูปที่ 3.5 ไทมิ่งไดอะแกรมแสดงการเขียนข้อมูลลงแรม

3.3.1 การแลตซ์ค่าข้อมูล 8 บิต โดยใช้ขา $\overline{LE}$ ของแรม

จากไทมิ่งไดอะแกรมรูปที่ 3.5 แสดงการเขียนข้อมูล(Data)ลงแรมสามารถวิเคราะห์จากพัลส์ข้อมูล ซึ่งตรงกับ D0 ของไทมิ่งไดอะแกรมและเอาต์พุตขาที่ 4 (B4) ของวงจรนับ ซึ่งตรงกับ [3] ของไทมิ่งไดอะแกรมซึ่งจะเปลี่ยนลอจิกเมื่อครบ 8 บิต ของ ข้อมูล

สามารถสังเคราะห์วงจรได้โดยสร้างวงจรวิเคราะห์จากขาของ B4 ของวงจรนับ ได้ดังนี้



รูปที่ 3.6 วงจรสร้างพัลล์ในการแลตซ์ค่าข้อมูล

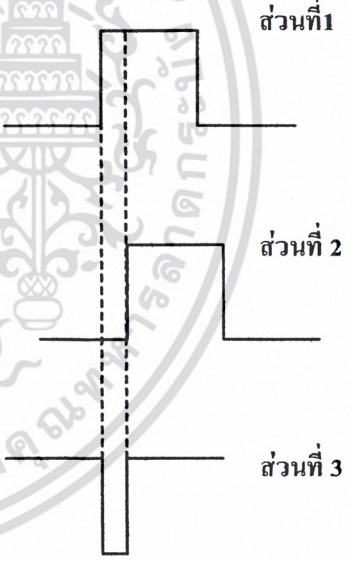
จากวงจรรูปที่ 3.6 ผลจากวิเคราะห์ขอบขาของ B4 สามารถสร้างพัลล์ที่เกิดจากทุกๆการเปลี่ยนแปลงค่าลอจิกที่ขอบขาของ B4 เพื่อให้สามารถแลตซ์ค่า 8 บิตก่อนหน้านี้ได้

วิธีการทำงานของวงจรวิเคราะห์ขอบขา B4 ของวงจรนับ

ส่วนที่ 1
พัลล์ปกติของ B4 ที่วิ่งเข้ามา

ส่วนที่2
พัลล์ปกติของ B4 ที่เข้าน๊อตเกท 4 ตัว
ซึ่งทุกตัวจะมีดีเลย์ อยู่ทำให้เอาต์พุต
ออกมาเกิดการหน่วงเวลา

ส่วนที่ 3
พัลล์ปกติของ B4 และพัลล์ปกติของ B4
ที่เข้าน๊อตเกท (Not gate)4 ตัว ผ่านวงจร เอ็กคลูซีฟออร์
(XOR) จะได้พัลล์แลตซ์ที่สามารถแลตซ์ค่าที 8 บิตข้อมูลได้



รูปที่ 3.7 วิธีการแลตซ์ค่าของวงจรสร้างพัลล์

3.3.2 การเขียนข้อมูล 8 บิตลงแรมในแต่ละแอดเดรสโดยใช้ขา $\overline{WR}$ ของแรม

จากไทมิ่งไดอะแกรม (Timing Diagram) แสดงการเขียนข้อมูล(Data)ลงแรม สามารถวิเคราะห์จาก ขาสัญญาณนาฬิกา และ ขาเอาต์พุต ที่ 1(B1) , 2(B2) และ 3(B3) ซึ่งตรงกับ [0] [1] และ [2] ของไทมิ่งไดอะแกรมซึ่งมีความสัมพันธ์กัน

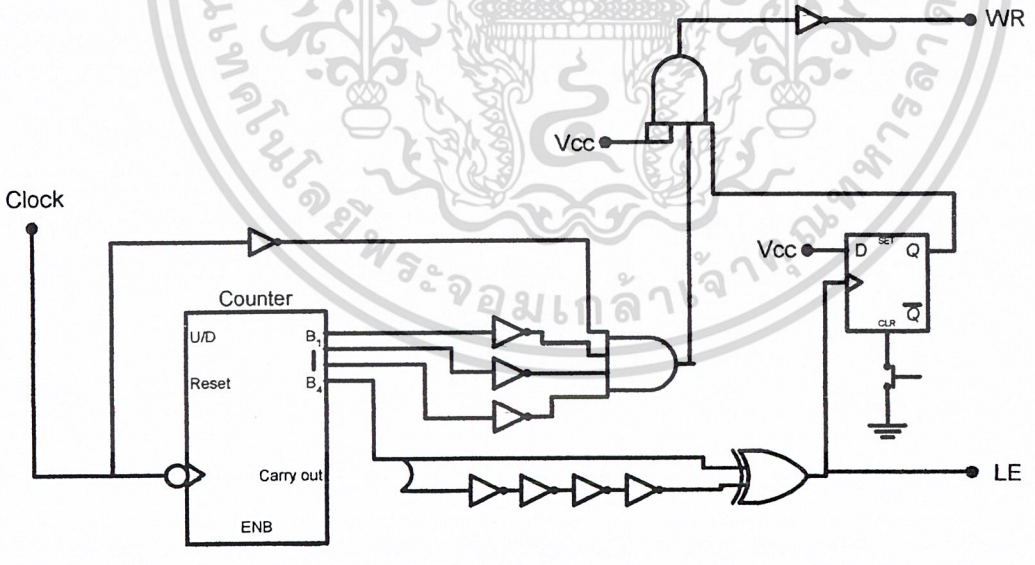
สามารถสังเคราะห์วงจรได้ดังนี้ จากพัลส์ของขา $\overline{WR}$ ที่เป็น 0 จะได้ความสัมพันธ์ คือ

- Clock เป็น 0 ได้ลอจิกเป็น $\overline{CK}$
- D0 เป็น 0 ได้ลอจิกเป็น $\overline{D0}$
- D1 เป็น 0 ได้ลอจิกเป็น $\overline{D1}$
- D2 เป็น 0 ได้ลอจิกเป็น $\overline{D2}$

สามารถสร้างสมการลอจิกได้คือ

$$\overline{WR} = \overline{CK} \bullet \overline{D0} \bullet \overline{D1} \bullet \overline{D2}$$

จากสมการลอจิกสามารถสร้างวงจรได้ดังรูปที่ 3.8



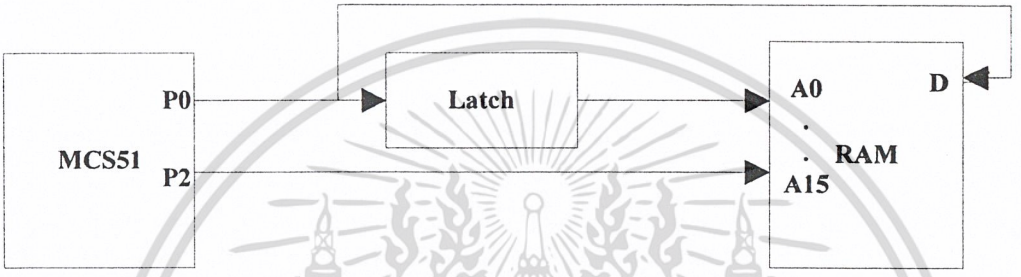
LE :สัญญาณควบคุมการแลตซ์ค่าข้อมูล 8 บิต โดยใช้ขา $\overline{LE}$ ของแรม

WR:สัญญาณการเขียนข้อมูล 8 บิตลงแรมในแต่ละแอดเดรสโดยใช้ขา $\overline{WR}$ ของแรม

รูปที่ 3.8 วงจรควบคุมการแลตซ์ค่า

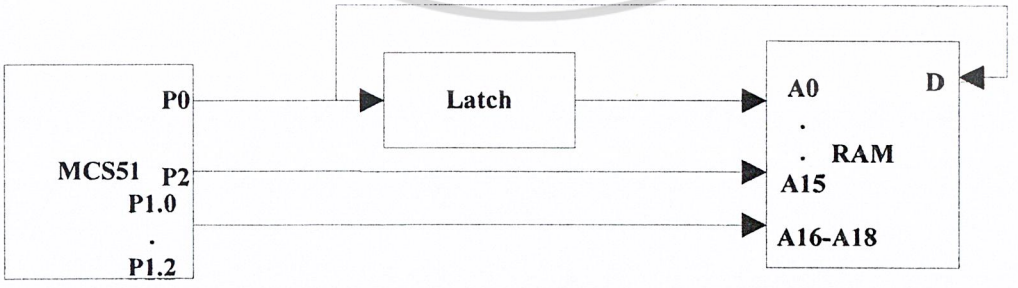
3.3 วิธีการทำ Paging

3.3.1 นำไมโครคอนโทรลเลอร์ 8051 มาทำการต่อขา P0.0 - P0.7 , P2.0 – P2.7 เข้ากับแรมที่ ขาแอดเดรส A0 – A7 โดยพอร์ต P0 ของไมโครคอนโทรลเลอร์ 8051 ต้องต่อผ่านวงจรแลตซ์ ขั้นตอนนี้จะเหมือนกับการติดต่อกับหน่วยความจำข้อมูลภายนอกขนาด 64 กิโลไบต์

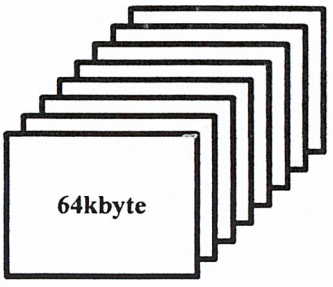


รูปที่ 3.9 การเชื่อมต่อไมโครคอนโทรลเลอร์ 8051 เข้ากับหน่วยความจำภายนอก 64 กิโลไบต์

3.3.2 ทำเทคนิคการเพจจิ่ง โดยนำไมโครคอนโทรลเลอร์ 8051 มาทำการต่อขา P1.0 – P1.2 เข้ากับแรม ที่ขาแอดเดรส A16 – A18 โดยการต่อวงจรในลักษณะนี้จะเสมือนกับเป็นการแบ่งหน่วยความจำเป็น 8 หน้า โดยแต่ละหน้าจะมีขนาด 64 กิโลไบต์ เนื่องจากไมโครคอนโทรลเลอร์สามารถเชื่อมต่อกับหน่วยความจำได้ครั้งละ 64 กิโลไบต์รวมแล้วจะได้ 512 กิโลไบต์

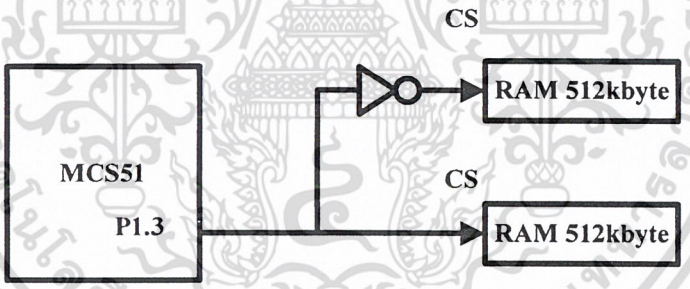


รูปที่ 3.10 การเชื่อมต่อไมโครคอนโทรลเลอร์ 8051 เข้ากับหน่วยความจำภายนอก 512 กิโลไบต์



รูปที่ 3.11 การเปรียบเทียบการเชื่อมต่อไมโครคอนโทรลเลอร์ 8051 เข้ากับหน่วยความจำภายนอก 512 กิโลไบต์

3.3.3 ขั้นตอนสุดท้ายก็จะเป็นการเลือกการใช้งานแรม ว่าจะเลือกอ่านจากตัวไหนก่อน โดยนำไมโครคอนโทรลเลอร์ 8051 มาต่อขา P1.3 เข้ากับแรม ที่ ขา CS ทั้ง 2 ตัวโดยแรม ตัวใดตัวหนึ่งจะต้องต่อ น๊อตเกท (Not gate) ไว้ที่ขา CS ด้วยเพื่อใช้ในการเลือกว่าจะอ่านข้อมูลจากแรม ตัวใด



รูปที่ 3.12 การใช้ไมโครคอนโทรลเลอร์ 8051 ในการเลือกใช้งานความหน่วยความจำภายนอก

3.4 การสลับโหมดการทำงานด้วยวงจรมัลติเพล็กซ์ (Multiplex)

หลักการ คือ เมื่อวงจรเลือก (Selector) มีข้อมูลเข้ามาพร้อมกัน 2 ชุด วงจรมัลติเพล็กซ์จะต้องสามารถเลือกได้ว่าจะใช้ข้อมูลชุดไหน โดยใช้สัญญาณเลือก (Select)เพื่อใช้ในการตัดสินใจ

หลังจากทำการเขียนข้อมูลลงแรมจนเสร็จแล้ว ต้องมีการทำมัลติเพล็กซ์ เพื่อใช้ในการสลับการทำงานของเครื่องทดสอบสัญญาณแรนดัม จากการเขียนข้อมูลลงแรม มาเป็นการอ่านข้อมูลจากแรม โดยใช้ ไมโครคอนโทรลเลอร์ 8051 เป็นตัวตัดสินใจ และ ส่งสัญญาณเลือก ไปยังวงจรมัลติเพล็กซ์เพื่อใช้ในการสลับการทำงาน

3.5 การส่งข้อมูลผ่านพอร์ตอนุกรม

เครื่องทดสอบสัญญาณ แรนดัม แสดงผลบนคอมพิวเตอร์ จะใช้พอร์ตอนุกรม RS-232 เพื่อใช้ในการส่งข้อมูล โดยใช้โหมดการทำงานของพอร์ตอนุกรมในไมโครคอนโทรลเลอร์ 8051 โหมดที่ 1 เนื่องจากมีกระบวนการที่ไม่ซับซ้อนและสามารถทำการรับส่งข้อมูลกับคอมพิวเตอร์ได้อย่างมีประสิทธิภาพ

โดยอาศัยการเชื่อมต่อผ่านไอซีพิเศษที่ทำหน้าที่ในการแปลงระดับสัญญาณ ซึ่งจะทำการแปลงข้อมูลส่งของไมโครคอนโทรลเลอร์ 8051 จากระดับทีทีแอล (TTL)ไปเป็นระดับของพอร์ตอนุกรมและทำการแปลงข้อมูลจากคอมพิวเตอร์จากระดับของพอร์ตอนุกรมเป็นระดับทีทีแอล เพื่อให้สามารถถ่ายทอดไปยังไมโครคอนโทรลเลอร์ 8051 ได้อย่างสมบูรณ์

3.6 สรุปขั้นตอนการเขียนและอ่านข้อมูล

3.6.1 ขั้นตอนการเขียนข้อมูลลงแรม

ขั้นตอนการเขียนข้อมูลลงแรมจะเริ่มจากการรับข้อมูลอินพุตแบบไบนารีพร้อมกับสัญญาณนาฬิกาจากวงจรกำเนิดสัญญาณแรนดัม โดยมีขั้นตอนดังนี้

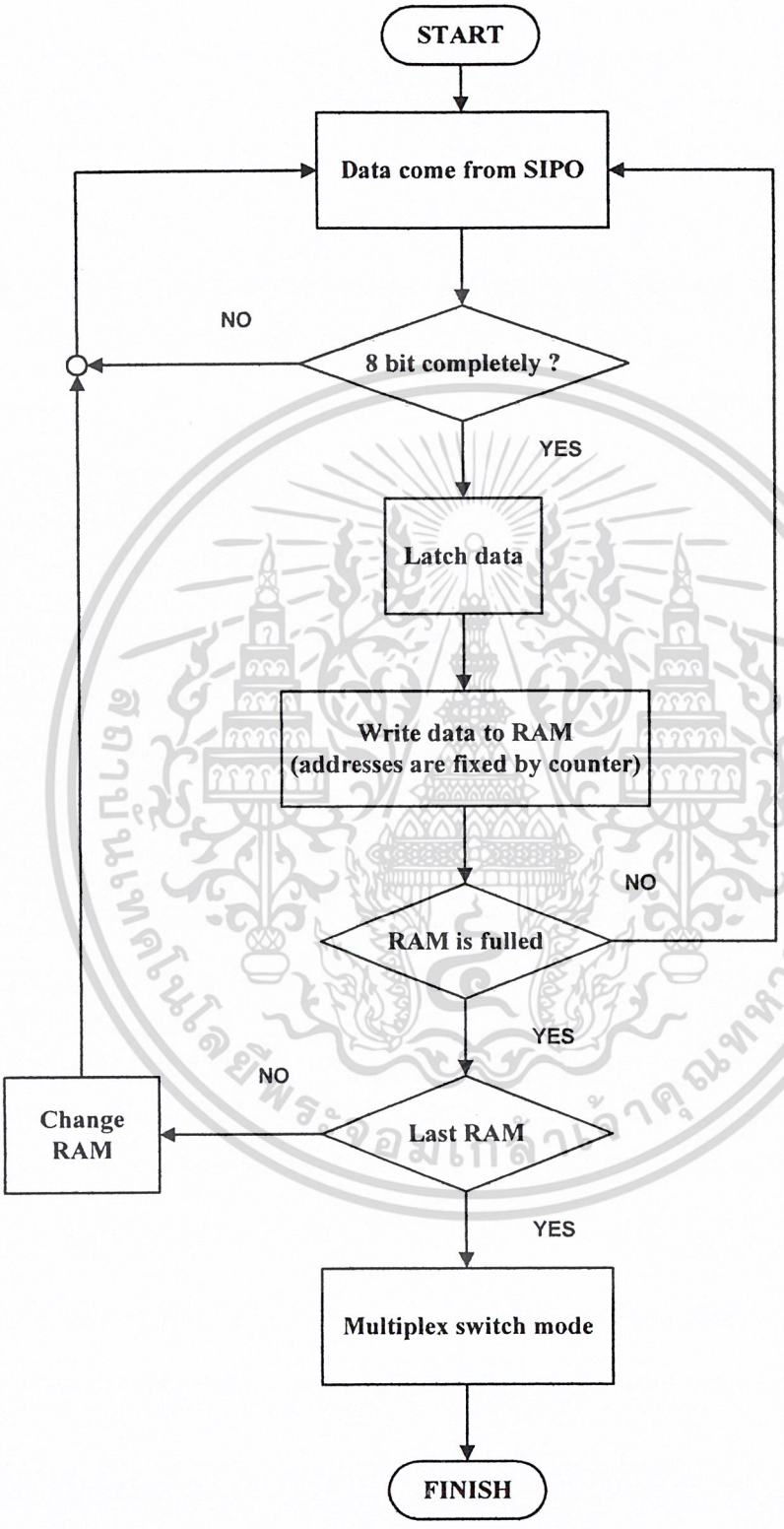
1. วงจรเอสไอพีโอ จะรับข้อมูลอินพุตเข้ามาจนครบ 8 บิต
2. แลตช์ค่าข้อมูลค้างไว้ ก่อนที่จะทำการเขียนข้อมูลลงแรม
3. เขียนข้อมูลลงแรมในตำแหน่งแอดเดรสที่ถูกกำหนดโดยวงจรนับ และ ทำการเพิ่มตำแหน่งแอดเดรสขึ้นเรื่อย ๆ
4. ซึ่งขณะที่เขียนจะมีการตรวจสอบว่าแรมที่เขียนนั้นความจุเต็มหรือไม่โดยใช้ไมโครคอนโทรลเลอร์ 8051 ในการตรวจสอบถ้าเต็มจะใช้เปลี่ยนไปเขียนแรมอีกตัวหนึ่ง
5. เมื่อเขียนข้อมูลเสร็จครบทั้ง 1 เมกกะไบต์ แล้ว จะใช้ไมโครคอนโทรลเลอร์ 8051 ทำการสลับโหมดการทำงานโดยใช้วงจรมัลติเพล็กซ์

3.6.2 ขั้นตอนการอ่านข้อมูลจากแรม

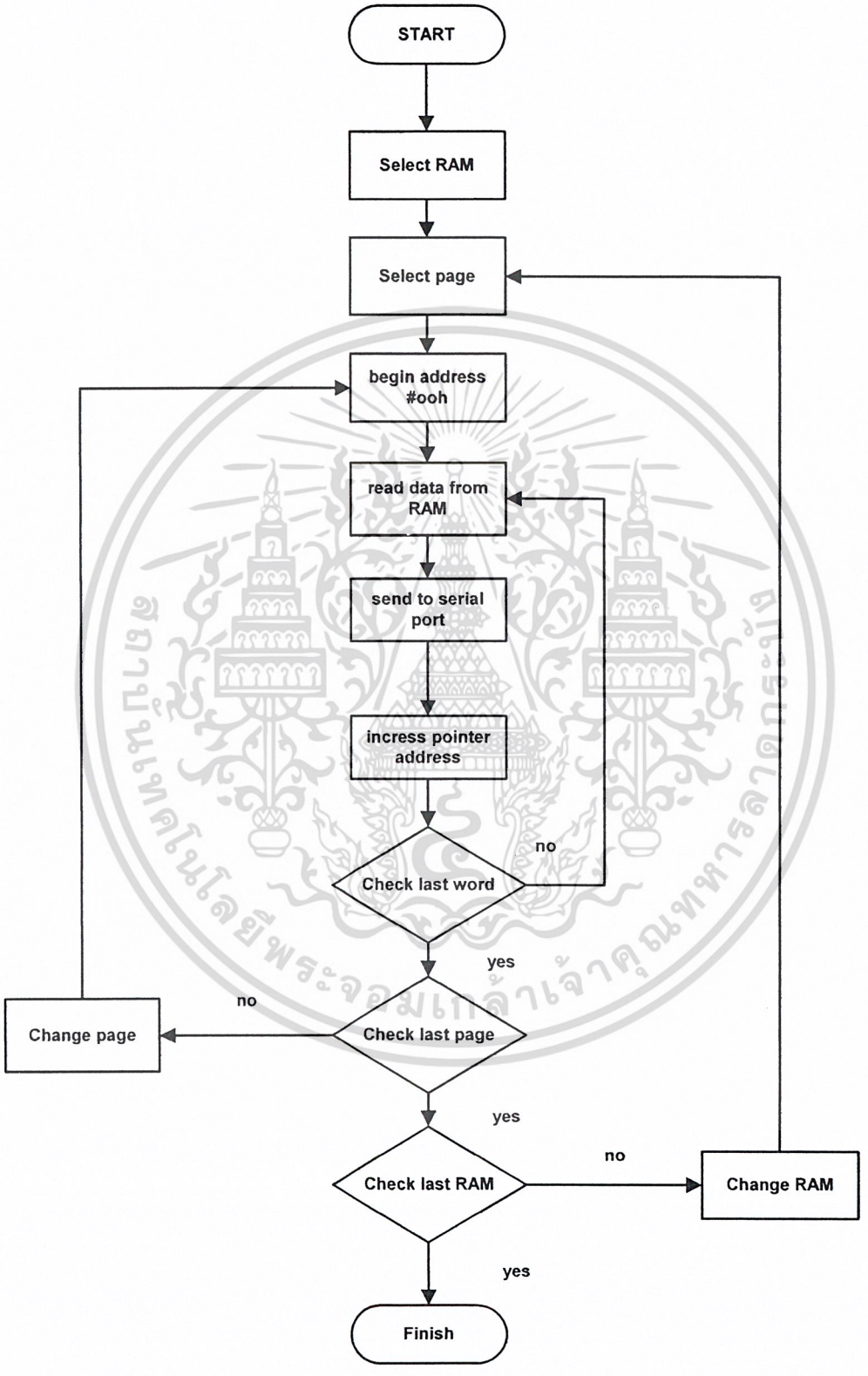
หลังจากใช้ไมโครคอนโทรลเลอร์ 8051 ทำการสลับโหมดการทำงานโดยใช้วงจรมัลติเพล็กซ์เสร็จแล้วก็จะเป็นขั้นตอนการอ่านข้อมูลจากแรมโดยมีขั้นตอนดังนี้

1. เริ่มด้วยการเลือกแรมที่จะอ่านเป็นตัวแรก ซึ่งจะกำหนดให้เริ่มอ่านแรมตัวที่ 0 เป็นตัวแรก
2. กำหนดหน้าของแรม (Page) ที่จะทำการอ่านข้อมูลโดยตั้งค่าให้เริ่มอ่านจะหน้าที่ 0 เป็นหน้าแรก

3. ทำการกำหนดแอดเดรส ที่จะทำการอ่านข้อมูล โดยตั้งค่าให้เริ่มอ่านที่แอดเดรส #00h
4. จากนั้นจะทำการส่งข้อมูลเข้ารีจิสเตอร์บัฟเฟอร์ของพอร์ตอนุกรม (SBUF : Serial data buffer register) เพื่อส่งข้อมูลผ่านพอร์ตอนุกรม
5. ทำการเพิ่มตำแหน่งแอดเดรสแล้วอ่านข้อมูลจนจบหน้านั้น
6. เปลี่ยนหน้าของแรมไปเรื่อย ๆ จนกว่าจะอ่านข้อมูลหมดทั้งแรม
7. ไมโครคอนโทรลเลอร์ 8051 จะทำการเปลี่ยนไปอ่านข้อมูลจากแรมอีก 1 ตัว และทำขั้นตอนเดิมจนจบการทำงาน



รูปที่ 3.13 โฟลว์ชาร์ตแสดงขั้นตอนการเขียนข้อมูลลงแรม



รูปที่ 3.14 โฟลว์ชาร์ตแสดงการอ่านข้อมูลจากแรม

บทที่ 4
การทดลอง

4.1 การทดลองเขียนข้อมูลลงแรมและตรวจสอบข้อผิดพลาดในการเขียนข้อมูล

จุดประสงค์

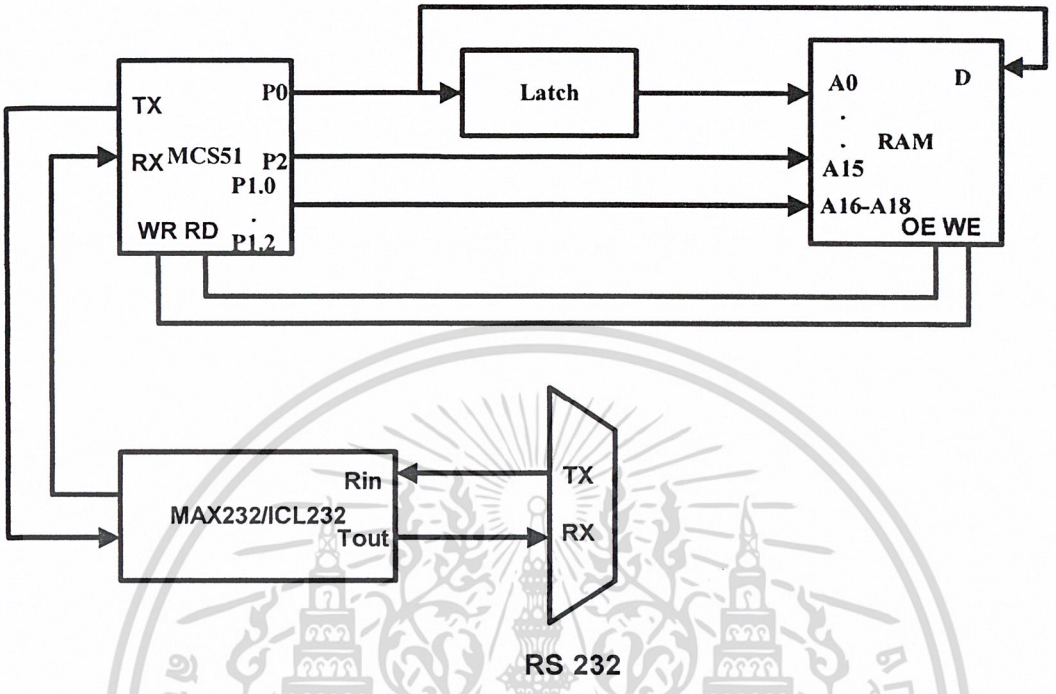
1. เพื่อทดลองการทำเพจจิ่ง
2. ตรวจสอบว่าแรมไม่มีแอดเดรสเสีย
3. ทดลองส่งข้อมูลผ่านพอร์ตอนุกรมโดยแสดงผลทางโปรแกรมไฮเปอร์เทอร์มิ-นอล (Hyper Teminal)

อุปกรณ์

1. วงจรไมโครคอนโทรลเลอร์ 8051 ที่เชื่อมต่อกับแรมขนาดความจุ 512 กิโลไบต์
2. วงจรส่งข้อมูลผ่านพอร์ตอนุกรม
3. เครื่องคอมพิวเตอร์ 1 เครื่อง

ขั้นตอนการทดลอง

1. เริ่มโดยการนำไมโครคอนโทรลเลอร์ 8051 มาทำการต่อขาพอร์ต 0 และพอร์ต เข้ากับแรมที่ขาแอดเดรส A0-A15 โดยพอร์ต 0 ต้องต่อผ่านวงจรแลตซ์
2. โดยนำไมโครคอนโทรลเลอร์ 8051 มาทำการต่อขา P1.0 – P1.2 เข้ากับแรม ที่ขาแอดเดรส A16 – A18
3. ทำการเขียนโปรแกรมทดสอบแรม โดยให้เขียนค่า “A” เข้าไปทุกแอดเดรสของแรม จากนั้นจึงทำการอ่านข้อมูลจากแอดเดรสนั้นว่าเป็น “A” หรือไม่ โดยจะแสดงผลทางหน้าจากคอมพิวเตอร์ผ่านโปรแกรมไฮเปอร์เทอร์มินอล
4. ถ้าค่าที่อ่านจากแอดเดรสนั้นไม่ใช่ค่า “A” โปรแกรมจะทำการแสดงแอดเดรสที่เกิดข้อผิดพลาด
5. เมื่อทำการตรวจสอบครบ 1 หน้า ของแรม ไมโครคอนโทรลเลอร์ 8051 จะสั่งให้มีการเปลี่ยนหน้า
6. ทำกระบวนการนี้จนครบทั้ง 8 หน้า (512 กิโลไบต์) ถ้าไม่เกิดข้อผิดพลาดในการเขียนและอ่านค่า “A” จากแรม ไมโครคอนโทรลเลอร์ 8051 จะทำการแจ้งจบกระบวนการทดลอง



รูปที่ 4.1 วงจรทดลองเขียนข้อมูลลงแรมและแสดงผลทางโปรแกรมไฮเปอร์เทอร์มินอล

ผลการทดลอง

จากการทดลอง แรมที่ใช้สามารถเขียนข้อมูลและอ่านข้อมูลได้ทุกแอดเดรส และไมโครคอนโทรลเลอร์สามารถส่งข้อมูลผ่านพอร์ตอนุกรมได้อย่างถูกต้อง

4.2 การทดลองโปรแกรมทดสอบค่าแรนดัมจากวงจรกำเนิดสัญญาณกึ่งแรนดัม โดยใช้มาตรฐาน FIPS PUB 140-1

จุดประสงค์

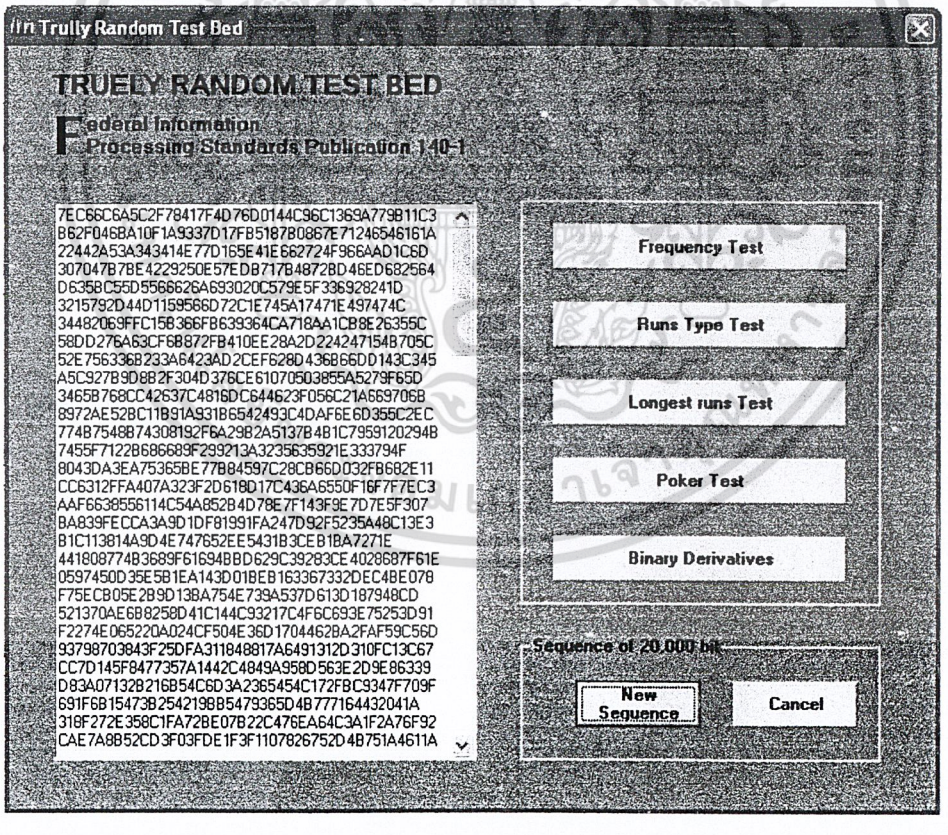
เพื่อทดสอบการทำงานของโปรแกรมทดสอบค่าแรนดัม

อุปกรณ์

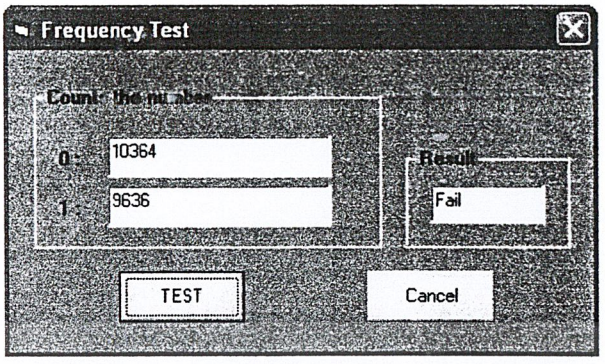
1. โปรแกรมทดสอบค่าแรนดัมโดยใช้มาตรฐาน FIPS PUB 140-1
2. วงจรกำเนิดสัญญาณกึ่งแรนดัม (Pseudo-Random Number Generators)

ขั้นตอนการทดลอง

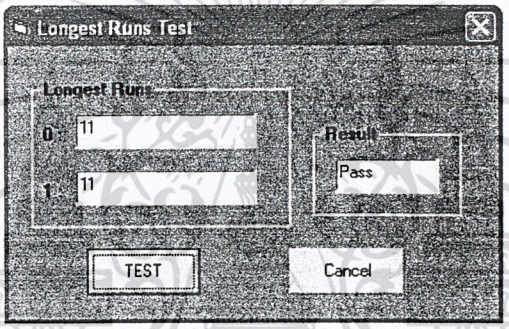
1. นำวงจรกำเนิดสัญญาณกึ่งแรนดัมมาเชื่อมต่อกับโปรแกรมไฮเปอร์เทอร์มินอล
2. นำข้อมูลที่ส่งมาจากวงจรกำเนิดสัญญาณมาสร้างไฟล์เอกสาร
3. ใช้โปรแกรมทดสอบค่าแรนดัมตรวจสอบค่าแรนดัมที่ได้จากวงจร



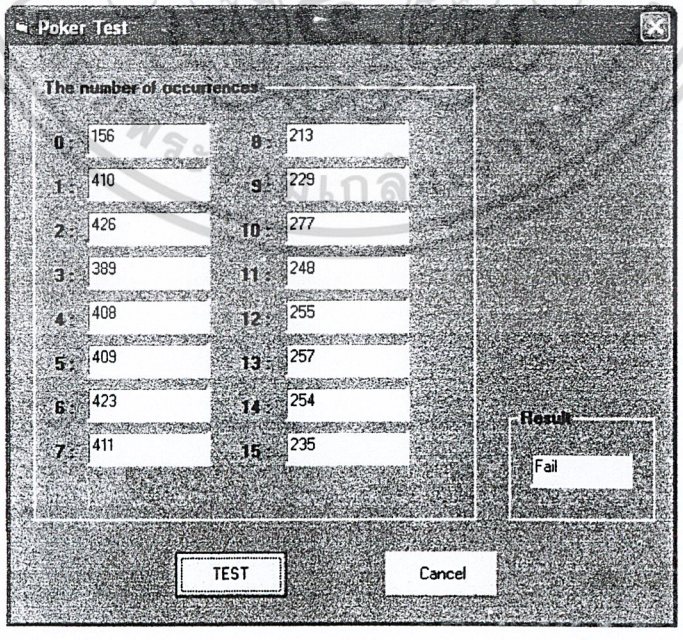
รูปที่ 4.2 การใช้โปรแกรมทดสอบค่าแรนดัมจากวงจรกำเนิดสัญญาณกึ่งแรนดัม



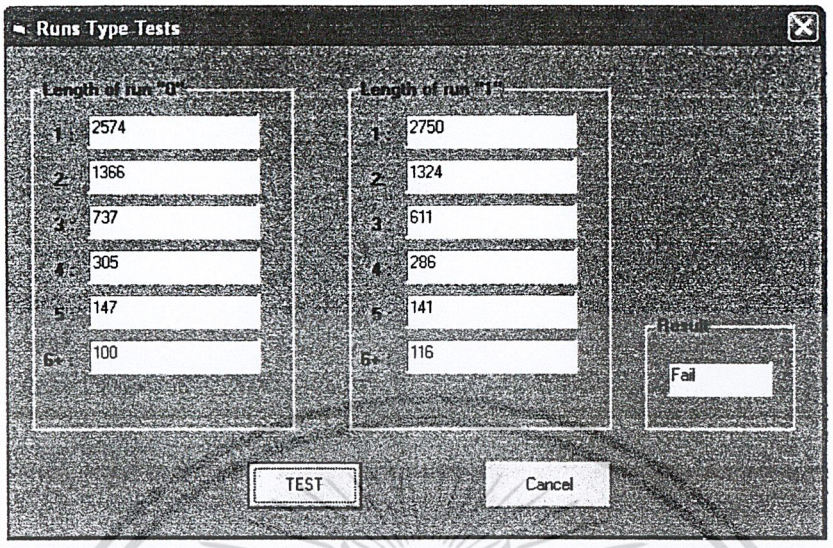
รูปที่ 4.3 ทดสอบค่าแรนดัมจากวงจรกำเนิดสัญญาณกึ่งแรนดัมด้วยวิธีการทดสอบความถี่



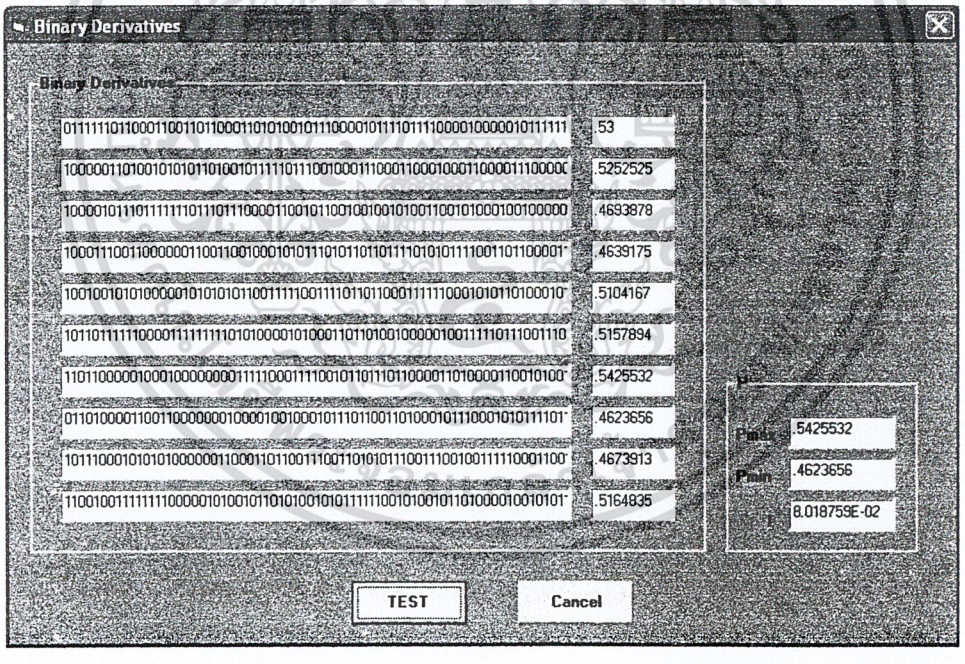
รูปที่ 4.4 ทดสอบค่าแรนดัมจากวงจรกำเนิดสัญญาณกึ่งแรนดัมด้วยวิธีล็องเกสรัน



รูปที่ 4.5 ทดสอบค่าแรนดัมจากวงจรกำเนิดสัญญาณกึ่งแรนดัมด้วยวิธีโปเกอร์



รูปที่ 4.6 ทดสอบค่าแรนดัมจากวงจรกำเนิดสัญญาณกึ่งแรนดัมด้วยวิธีรันไทด์



รูปที่ 4.7 ทดสอบค่าแรนดัมจากวงจรกำเนิดสัญญาณกึ่งแรนดัมด้วยวิธีไบนารีเดอริเวทีฟ

ผลการทดลอง

จากการทดลองพบว่าค่าแรนดัมที่สร้างมาจากวงจรกำเนิดสัญญาณกึ่งแรนดัม (Pseudo-Random Number Generators) จะไม่ผ่านการทดสอบตามมาตรฐาน FIPS 140-1 เพราะ ค่าแรนดัมที่ได้จากวงจรไม่เป็นแรนดัมที่แท้จริง

4.3 การทดลองโปรแกรมทดสอบค่าแรนดัมจากวงจรกำเนิดสัญญาณแรนดัมแบบดับเบิ้ลสโคล (Double scroll)โดยใช้มาตรฐาน FIPS PUB 140-1

จุดประสงค์

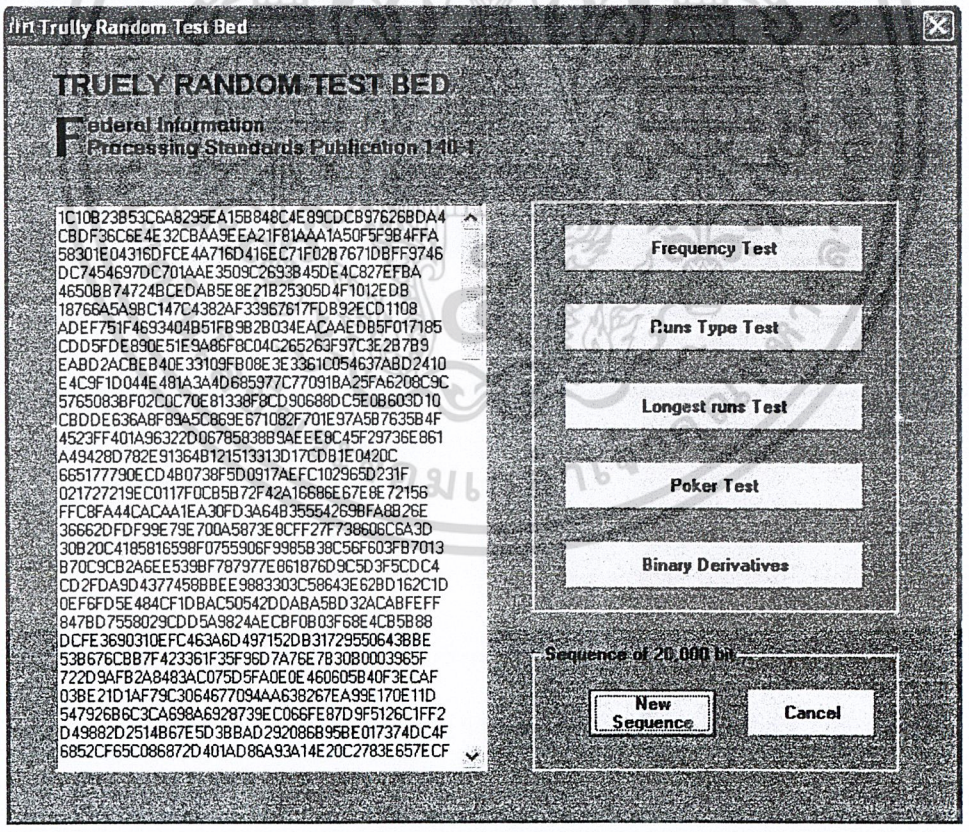
เพื่อทดสอบการทำงานของโปรแกรมทดสอบค่าแรนดัม

อุปกรณ์

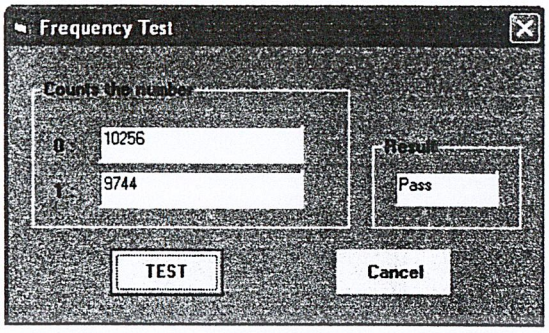
1. โปรแกรมทดสอบค่าแรนดัมโดยใช้มาตรฐาน FIPS PUB 140-1
2. วงจรกำเนิดสัญญาแรนดัมแบบดับเบิ้ลสโคล

ขั้นตอนการทดลอง

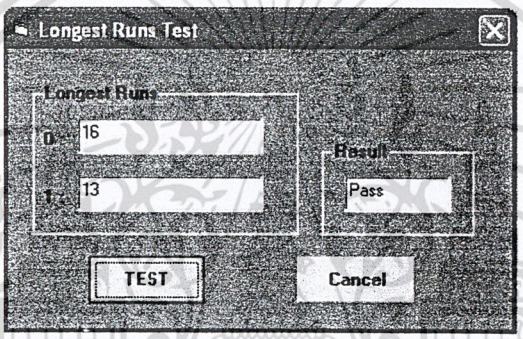
1. นำข้อมูลที่เกิดมาจากวงจรกำเนิดสัญญาณมาสร้างไฟล์เอกสาร
3. ใช้โปรแกรมทดสอบค่าแรนดัมตรวจสอบค่าแรนดัมที่ได้จากวงจร



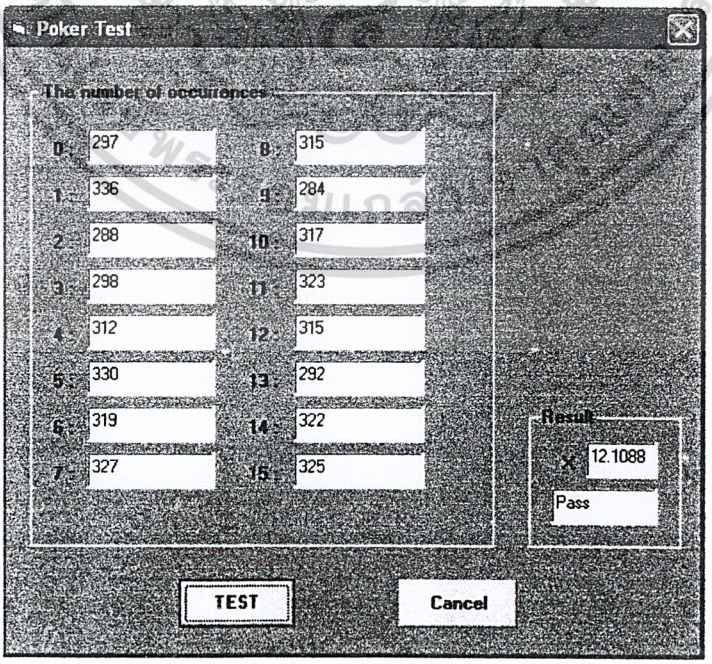
รูปที่ 4.8 การใช้โปรแกรมทดสอบค่าแรนดัมจากวงจรกำเนิดสัญญาณแรนดัม



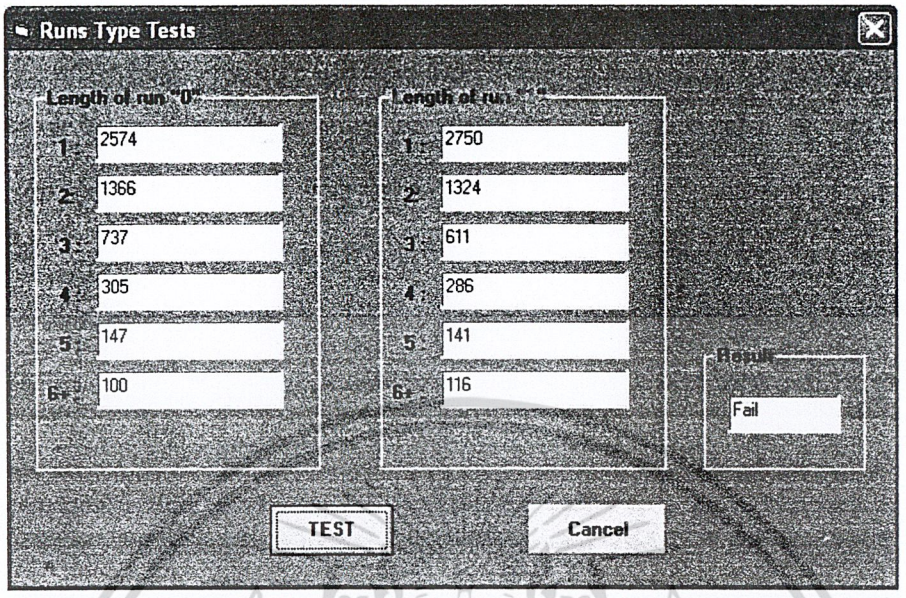
รูปที่ 4.9 ทดสอบค่าแรนดัมจากวงจรกำเนิดสัญญาณแรนดัมด้วยวิธีการทดสอบความถี่



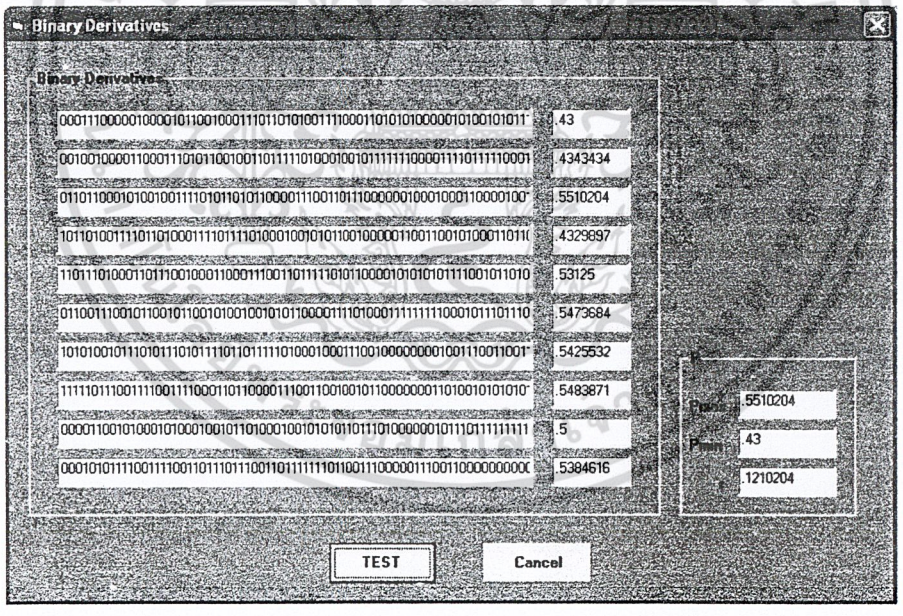
รูปที่ 4.10 ทดสอบค่าแรนดัมจากวงจรกำเนิดสัญญาณแรนดัมด้วยวิธีล็องเกสรัน



รูปที่ 4.11 ทดสอบค่าแรนดัมจากวงจรกำเนิดสัญญาณแรนดัมด้วยวิธีโปเกอร์



รูปที่ 4.12 ทดสอบค่าแรนดัมจากวงจรกำเนิดสัญญาณแรนดัมด้วยวิธีรันไทด์



รูปที่ 4.13 ทดสอบค่าแรนดัมจากวงจรกำเนิดสัญญาณแรนดัมด้วยวิธีไบนารีเดอริเวทีฟ

ผลการทดลอง

จากการทดลองพบว่าค่าแรนดัมที่สร้างมาจากวงจรกำเนิดสัญญาณแรนดัม แบบดับเบิ้ลส-โคล ผ่านการทดสอบตามมาตรฐาน FIPS 140-1 เพราะ ค่าแรนดัมที่ได้จากวงจรเป็นแรนดัมที่แท้จริง

บทที่ 5
สรุปผลการทดลอง

5.1 สรุปผลการทดลอง

ในโครงงานนี้เป็นการนำค่าแรนดัมที่เกิดจากวงจรกำเนิดค่าแรนดัมมาทำการวิเคราะห์ทางสถิติ โดยใช้โปรแกรมทดสอบค่าแรนดัม ซึ่งใช้มาตรฐาน FIPS 140-1 เป็นมาตรฐานหลักในการทดสอบ

5.2 ปัญหาที่เกิดขึ้นในการทดลอง

การสร้างวงจรทดสอบสัญญาณแรนดัม ออกแบบได้ยากเนื่องจากการรับค่าจากเครื่องกำเนิดสัญญาณแรนดัมมาเก็บในหน่วยความจำข้อมูลของวงจรทดสอบสัญญาณแรนดัม ถ้ามีสัญญาณรบกวนเพียงเล็กน้อย ข้อมูลที่รับเข้ามาอาจจะผิดเพี้ยนได้ ซึ่งมีผลต่อการทดสอบ ความเป็นแรนดัมอย่างแท้จริง ของโปรแกรมทดสอบค่าแรนดัม ทำให้การวิเคราะห์วงจรกำเนิดสัญญาณแรนดัมผิดพลาดได้ ซึ่งปัญหาเหล่านี้ ต้องออกแบบวงจรทดสอบสัญญาณแรนดัมให้มีสัญญาณรบกวนน้อยที่สุด โดยใช้เครื่องมือทดสอบและอุปกรณ์ที่มีมาตรฐานสูง จึงจะได้วงจรทดสอบสัญญาณแรนดัมที่มีประสิทธิภาพสูง

5.3 แนวทางในการพัฒนา

ในการพัฒนาเครื่องกำเนิดสัญญาณแรนดัม จำเป็นต้องมีเครื่องทดสอบสัญญาณแรนดัม โปรแกรมทดสอบสัญญาณแรนดัม เพื่อง่ายต่อการวิเคราะห์ผลให้ได้ตามมาตรฐาน FIPS 140-1 ซึ่งเป็นพื้นฐานสำคัญของการวิจัยด้านวิทยาการด้านเข้ารหัสลับ ซึ่งจะส่งผลให้งานวิจัยด้านนี้พัฒนาได้อย่างรวดเร็ว

บรรณานุกรม

1. National institute of standards and technology, **Federal Information Processing Standards Publication 140-1**, U.S. DEPARTMENT OF COMMERCE, 11 January 1994

2. John M. Carroll, Lynda E. Robbins , **Using Binary Derivatives to Test an Enhancement of DES,** Cryptologia, Vol.12, pp.193--208, 1988

3. Andrew Weigl, Walter Anheier, **Hardware Comparison of Seven Random Number Generator Tests for Smart Cards**, Institute for Electromagnetic Theory and Microelectronics

4. กฤดากร กล่อมการ, **การสื่อสารข้อมูล(Data Communications)**, ภาควิชาวิศวกรรม-สารสนเทศ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

5. วรพจน์ กรแก้ววัฒนกุล, ชัยวัฒน์ ลิ้มพรจิตรวิไล, **เรียนรู้และปฏิบัติการไมโครคอนโทรล-เลอร์ MCS-51 แบบแฟลช**, บริษัท อินโนเวตีฟ เอ็กเพอริเมนต์ จำกัด