เครื่องป้องกันการดักฟังทางโทรศัพท์
โดยการเข้ารหัสแบบ มาตรฐาน

SCRAMBLER BY ALGORITHYM

DATA ENCRYPTION STANDARD

โดย

นาย สมพงศ์ ชวดจีง

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิศวกรรมศาสตร์บัณฑิต

สาขาวิศวกรรมศาสตร์ ภาควิชาโทรคมนาคม

สถาบันเทคโนโลยีพระเกล้าเจ้าคุณทหาร ฉลาดกระบัง

ปีการศึกษา ๒๕๓๖

เครื่องป้องกันการดักฟัง ทางโทรศัพท์โดยการเข้ารหัสแบบมาตรฐาน
SCRAMBLER BY ALGORITHYM DATA ENCRYPTION STANDARD

นายสมพงศ์ ฮวดจึง
Mr SOMPONG HOUDJUNG
อาจารย์ สมยศ จุณณะปิยะ อาจารย์ที่ ปรึกษา
Mr SOMYOT CHUNNAPIYA ADVISOR

บทคัดย่อ

โครงงานนี้เป็นเครื่องป้องกันการดักฟังทางโทรศัพท์ โดยนำการเข้ารหัสแบบ มาตรฐาน และใช้ไมโครคอนโทรลเล่อร์ ตระกูล MCS - 51 .ในการควบคุมระบบทั้งหมด การเข้ารหัส เป็นการเข้ารหัสแบบดิจิตอล ขอบเขตของโครงงานเป็นลักษณะต่อเชื่อมกันแบบ จุดต่อจุดโดยส่งผ่านการเชื่อมต่อ แบบ RS 232 ส่งข้อมูลแบบ อนุกรม อะซิงโครนัส ฟูลดูเพลกซ์ ด้วยความเร็ว 9600 บิท/ 1วินาที

Abstract

This project is Scrambler Telephone that encode by algorithym Data Encryption Standard and to used Microcontroller for control all system . Encode is endcode digital . This is Project interface type point to point ,RS 232 ,send data by asyncronouse serie type full duplex 9600 bit/s

สารบาญ

บทที่ 1
บทนำ

ในปัจจุบันการติดต่อสื่อสารเป็นสิ่งที่จำเป็นมาก ทั้งในทางราชการ พลเรือน ทางทหาร และวงการธุรกิจ เพื่อเพิ่มความสะดวกรวดเร็ว ประหยัดค่าใช้จ่ายและใช้เวลาระบบการสื่อสารด้วยกันหลายทาง เช่นทางคู่สายโทรศัพท์ ทางวิทยุโทรทัศน์ ซึ่งเป็นข้อมูล วิชาการหรือความบันเทิงต่าง ๆ ที่ไม่เป็นความลับใด ๆ แต่ในการติดต่อสื่อสารระหว่างหน่วยงานที่สำคัญ บางครั้งต้องคำนึงถึงความปลอดภัย ในการแพร่ข่าวสารผ่านวิทยุมือถือ หรือคู่สายโทรศัพท์ ซึ่งถูกผู้ลอบดักฟังได้ง่ายจากผู้หวังประโยชน์ ยังผลให้เกิดความเสียหายจากการแพร่ของข่าวนั้น ในวิทยานิพนธ์ฉบับนี้จึงได้ออกแบบและพัฒนาเครื่องมือที่ใช้ป้องกันการดักฟังทางช่องสื่อสารต่าง ๆ โดยทางผู้ส่งจะมีเครื่องที่ใช้ในการแปลงสัญญาน หรือข้อมูล ข่าวสาร ก่อนส่งออกทางช่องสื่อสาร และทางผู้รับก็จะมีเครื่องมือที่ใช้แปลงสัญญานกลับเพื่อให้สามารถเข้าใจข่าวสารต่าง ๆ ที่ทางผู้ส่งส่งมาให้ สำหรับบุคคลที่ลักลอบดักฟังจะไม่สามารถเข้าใจข่าวสารต่างๆ เป็นการรักษาความลับในการส่งข่าวสารได้อย่างมีประสิทธิภาพในระดับหนึ่ง

บทที่2

ประเภทของการเข้ารหัสเสียงพูด

ในการติดต่อสื่อสาร ในสื่อต่าง ๆ ที่มีอยู่นั้น มีจุดประสงค์เพียงเพื่อที่จะสื่อข่าวสารให้ถูกต้องรวดเร็ว ลดค่าใช้จ่ายในการเดินทาง แต่ในการติดต่อสื่อสารบางครั้งก็จำเป็นต้องรักษาข่าวเพื่อกันการรั่วไหล ปัญหาหลักอันหนึ่งที่ผู้ออกแบบเครื่องมือป้องกันต้องคำนึงถึง คือ ความก้าวหน้าและทันสมัยในการเอาเทคนิคใหม่มาใช้ในระบบสื่อสารหลายเทคนิคจำเป็นที่ต้องมีขีดจำกัดและกฎข้อบังคับสำหรับการสื่อสารประเภทนั้น ตัวอย่างเช่น การกำหนดย่านความถี่ (bandwidth) ของระบบโทรศัพท์และวิทยุ และอีกสิ่งหนึ่งที่ต้องคำนึงถึง คือ คุณภาพของเสียงที่ใช้ในระบบสื่อสารประเภทนั้น ว่ามีความชัดเจนเพียงใด ความชัดเจนของเสียงจะไม่เพิ่มขึ้น ถ้าระบบการเชื่อมโยงการส่งผ่าน (transmission link) ไม่ดีพอ การย้ำข่าวสารเพื่อความถูกต้องเนื่องจากเหตุผลดังกล่าว เป็นการไม่ปลอดภัยสำหรับข่าวสารนั้น จึงควรที่จะพิจารณาถึงความเหมาะสมกับการนำไปใช้งานสำหรับแต่ละชนิดของเครื่องป้องกัน และชนิดของระบบเชื่อมโยงการส่งผ่านที่ใช้อยู่

การเข้ารหัสเสียงพูด (Encrypting speech) แบ่งตามเทคนิคได้ 2 วิธี คือ ทางด้านดิจิตอล (Digital) และอนาลอค (Andlogue)

<u>การเข้ารหัสเสียงพูดด้วยวิธีการทางอานาลอค</u>(Analogue encrypting speechg)

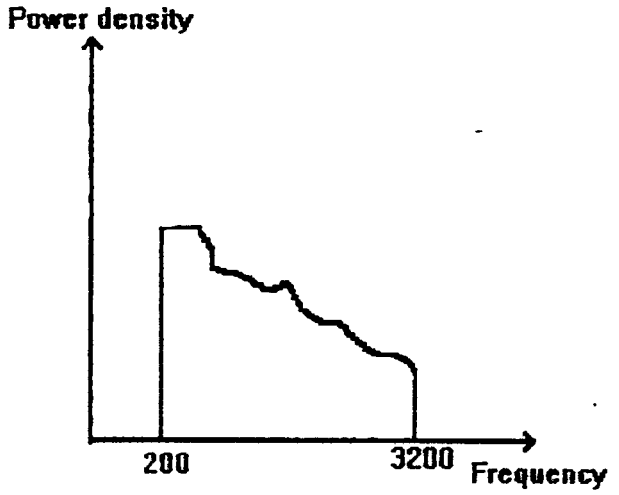
มีหลายแบบด้วยกันเช่น

1. Speech Invertion
2. Band - shift Innversion
3 .Bandscrambler or bandsplitter
4. Time Element Scrambler

ในแต่ละแบบสามารถอธิบายโดยสังเขบได้ดังนี้

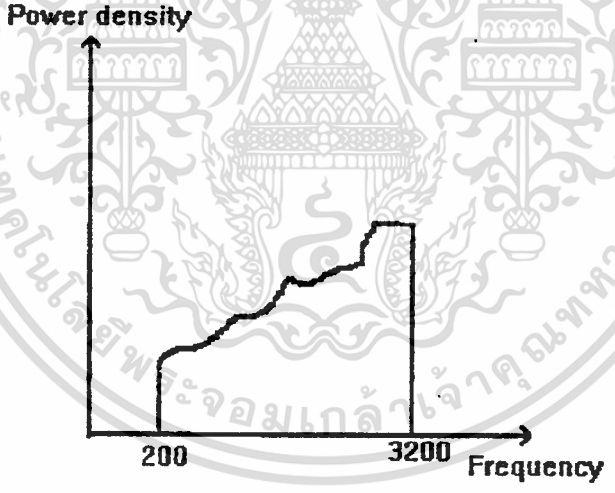
<u>1. Speech Invertion</u>

Speech Invertion เป็นการซแครมบลิ่ง (scrambling) ความถี่แบบหนึ่งเป็นที่รู้จักกันดี สมมุติว่าเรามีสัญญานเสียงที่มี แบนวิดจำกัดอยู่ในช่วง 200-3200 Hz ดังในรูปที่ 2.1



รูปที่ 2.1 A speech signal band - limite to 200-3200 Hz

ความคิดพื้นฐานของวิธีการดังกล่าวเพื่อที่จะเปลี่ยนจากความถี่สูงให้เป็นความถี่ต่ำ และจากความถี่ต่ำให้เป็นความถี่สูง ความสัมพันธ์นี้ค่อนข้างที่จะเข้าใจง่าย ผลของการเปลี่ยนแปลงดังกล่าวแสดงไว้ดังรูปที่ 2. 2ระบบดังกล่าวเป็นระบบที่ไม่ซับซ้อนมากนักการซแครมบลิ่งด้วยวิธีการดังกล่าวจะไม่ปลอดภัย เพราะสามารถที่จะ ดีซแครมบลิ่ง (descrambling) ได้โดย รีอินเวิร์ท (reinvert) แบบลองผิดลองถูก (traial and error) ก็สามารถที่จะได้สัญญานเดิมกลับมา

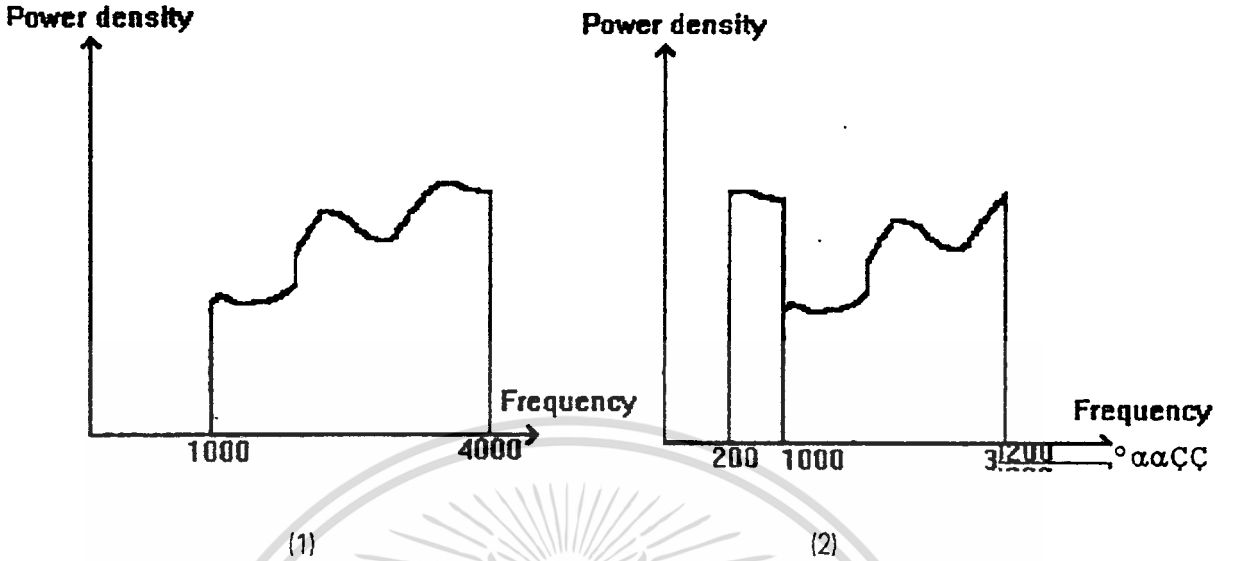


รูปที่ 2.2 Power density spectrum of inverted speech signal

2. Band - shift Invertion

เป็นการปรับปรุง Speech Inversion โดยที่สัญญานที่อยู่ในช่วง 200 - 3200 Hz จะถูกอินเวิร์ท (invert) และเลื่อน (shift) แบนวิด (1000 - 4000) สเปคตรัม (spectrum) ของสัญญานดังกล่าวแสดงในรูปที่ 2.3 (1) สัญญานนี้อยู่นอกแบนแตกต่างจากสัญญานตัวแรก แต่เราสามารถที่จัดการเอาส่วนที่เกิน 3200 Hz ย้ายมาอยู่ทางด้านความถี่ต่ำ (สังเกตุดูถึงแม้ว่าสัญญาน

ในรูป2.3จะมีขอบเขต (range) ของความถี่ที่แตกต่างกัน แต่ก็มีแบนวิดเท่ากับสัญญานตัวแรก) หลักการของ Band - shift Inverting ได้แสดงไวในรูปที่ 2.3 (2)



รูปที่ 2.3 หลักการของ Band - shift inversion

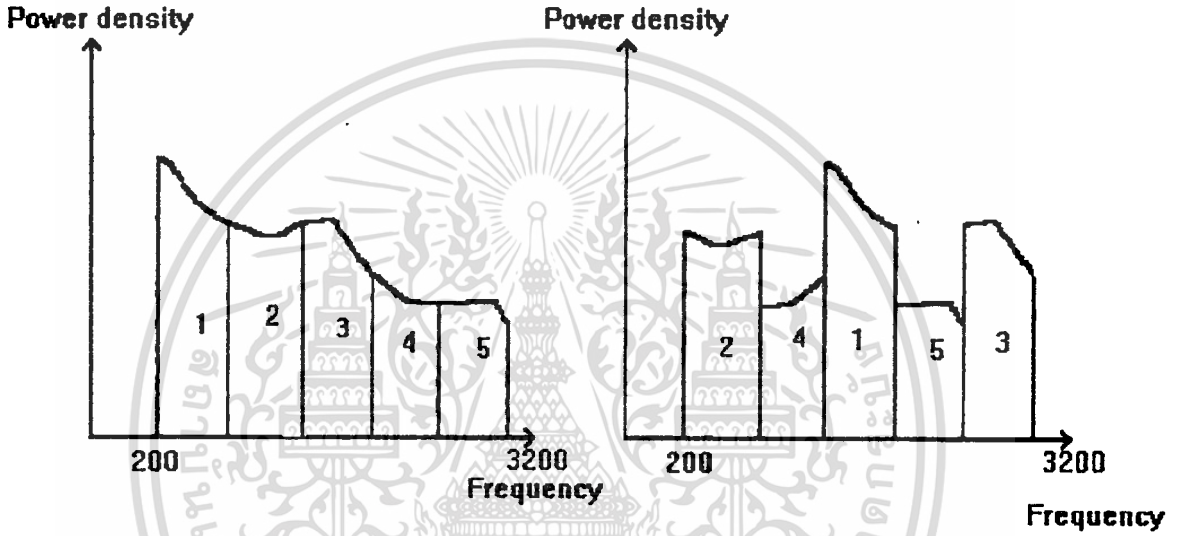
ตัวอย่างของ Band - shift inversion อันหนึ่งมีการอินเวิร์ทสัญญานด้วยความถี่พาหะ (carrier frequncy) ที่แตกต่างกันโดยมีการเลื่อน (shift) แบนแบบคงที่ทำให้คอมบิเนชั่นที่เป็นไปได้ยาก(possible combination) มีจำนวนจำกัด วิธีที่จะเพิ่มคอมบิเนชั่น อาจจะใช้ pseudo random gennerater เป็นตัวเลือกการเลื่อน (shift) ที่แตกต่างกัน และแต่ละแบบให้มีช่วงเวลา (time interval) ประมาณ 10 ถึง 20 ms แล้วจัดลำดับให้สลับเปลี่ยนกันไปเป็นลักษณะไซคลิค (cyclic) หลักการอันนี้เรียกว่า Cyclical band - shift Inversion ทำให้คอมบิเนชั่นที่เป็นไปได้มีมากขึ้น

ข้อเสียของ Band - shift inversion ที่เด่นชัดมีอยู่ 2 ข้อ ข้อแรกเนื่องจากคอมบิเนชั่น (descram- bling) ทำได้ง่ายโดยวิธีการลองผิดลองถูก (trail and error) อันที่สองความสามารถในการเข้าใจจากรายละเอียดที่ยังเหลืออยู่ (residual intengibility) หมายถึงความสามารถความชำนาญและความคุ้นเคยที่จะเข้าใจเสียงที่ถูกซแครมบลิ่งแล้ว แต่คงมีบางส่วนที่การซแครมบลิ่งเปลี่ยนแปลงสัญญานไปไม่มากนัก ทำให้สามารถที่จะเดาความหมายจากบางส่วนที่ยังเหลืออยู่ได้ ค่อนข้างสูง และจะยิ่งสูงขึ้นเมื่อข่าวสารได้ถูกรีอินเวิร์ท (reinvert) ด้วยแล้ว

3 Bandscrambler

ในกรณีของ bandscrambler หรือ bandspliter แบนวิดของเสียงพูด(speech bandwidth) จะแบ่งออกเท่าๆ กันเป็นหลายๆ ส่วนเรียกว่าแบนย่อย (sub - band) แต่ละแบนย่อย จะถูกซแครบลิ่งโดยการ สลับลำดับ (permutation) แบนย่อยเสียใหม่ ในบางระบบอาจจะมีการอินเวิร์ทในแต่แบนย่อยด้วย รูปที่ 2.4 แสดง ตัวอย่างของ bandscrambler แบบง่ายโดยแบ่งออก

เป็น 5 แบนย่อย จากรูปแบนย่อยที่ 1,2 และ 5 จะถูกอินเวิร์ท และถูกสลับตำแหน่ง ตัวอย่างดังกล่าว มีการจัดลำดับที่เป็นไปได้ (possible reorderings) เท่ากับ 5! และมีคอมบิเนชั่น สำหรับการ อินเวิร์ททั้ง 5 ตำแหน่งเท่ากับ 2^5 นั่นหมายถึงว่าสามารถที่จะมีคอมบิเนชั่นได้ถึง $5!x2^5$ เท่ากับ 3840 แบบ แต่ไม่ใช่ทั้งหมดที่สามารถนำไปใช้ได้ มีบางส่วน หลัง จากสัญญาณถูกซแครมบลิ่งแล้ว มีการเปลี่ยนแปลงสมบูรณ์ (คือไม่สามารถที่จะเข้าใจรายละเอียดได้) ส่วนที่เลือซึ่งเป็นส่วนใหญ่ไม่สามารถนำมาใช้ซแครมบลิ่งได้เพราะยังพอที่จะเข้าใจความได้แบบคลุมเคลือ



รูปที่ 2.4 Band scrambling technique

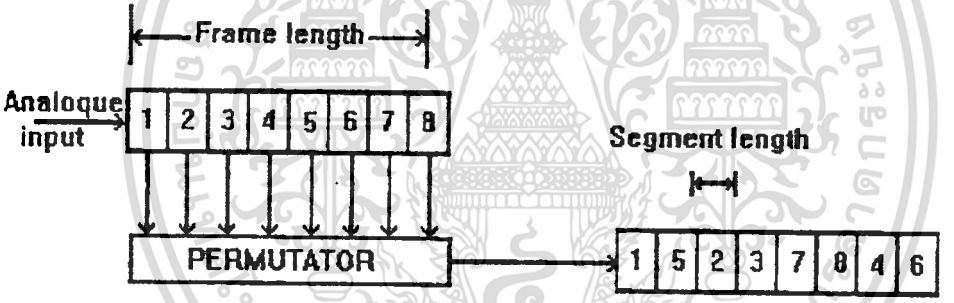
สำหรับการจัดลำดับใหม่ (reordering) เพียงอย่างเดียว โดยไม่ได้อินเวิร์ทมีเพียง 10 % เท่านั้นที่ซแครมบลิ่งแล้วใช้ได้ มีการหาเหตุผลว่าทำไมถึงเป็นเช่นนั้น ได้มีการทดลองเอาแบนย่อย(subband)บางอันออก แล้วจัดลำดับใหม่ พบว่าก็ยังมีบางส่วนที่พอเข้าใจได้ จากการวิเคราะห์ 40 เปอร์เซนต์ของpower spectrum energy ของเสียงอยู่ที่สองแบนย่อยแรก (200-1700 Hz) นั่นหมายความว่ามีเพียงสองแบนย่อยแรก ก็สามารถพอที่จะเข้าใจความหมายได้ เพราะฉะนั้น ไม่เป็นการยากเลยสำหรับผู้ที่เชี่ยวชาญที่หาตำแหน่งของสองแบนด์นั้น มาจัดเรียงใหม่ทำให้มีโอกาสเข้าใจข่าวสารนั้นได้ วิธีแก้ก็คือ ต้องแบ่งสองแบนด์ย่อยแรกให้มากขึ้น แล้วใช้pseudo random generator เป็นตัวกำหนดการจัดลำดับที่แตกต่างกันทุก ๆ 100-200 msโดยเรียกใช้จากการแอดเดรซซิ่ง (addressing) ที่ตำแหน่งใดตำแหน่งหนึ่ง

ตัวอย่างที่ได้แสดง ไว้อธิบาย การทำงานได้ดังนี้ สำหรับ 5 แบนด์ย่อยสามารถจัดลำดับได้ 32 แบบ คอมบิเนชั่นของ การอินเวิร์ท แบนด์ย่อยเท่ากับ 2^5 แบบและ 1024 แบบ สำหรับการจัดเรียง(rearragement) 10 บิท แอดเดรซของหน่วยความจำ จะเห็นว่าคอมบิเนชั่น

เป็นไปได้ (possible combination) มีมากขึ้นถ้าเราพิจารณาถึงการเพิ่มจำนวนแบนย่อยของ Bandscrambler ดูเหมือนจะเป็นการ เพิ่มจำนวนการจัดลำดับ และความปลอดภัยของข่าวสาร แต่ถ้ามากเกินไป จะทำให้ยุ่งยากใน ทางปฏิบัติ เพราะการเพิ่มจำนวนการแบ่งแบนด์ย่อย จะต้องใช้ฟิลเตอร์ และส่วนประกอบอื่น ๆ เพิ่ม ตาม ทำให้สัญญาณรบกวนในระบบมีมากขึ้น ดูเหมือนว่าการปรับปรุงแก้ไข (modification) การขแครมบลิ่งมากเกินไป ไม่ได้ทำให้คุณภาพของเสียงดีขึ้น

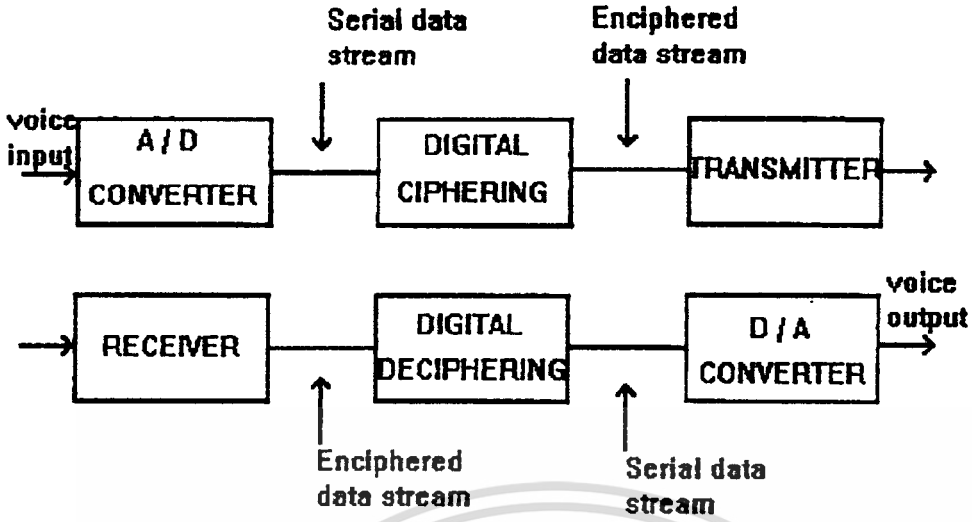
4. Time Element Scramblers (T.E.S)

การทำงานของ Time Element Scramblers (T.E.S) อาศัยหลักการพื้นฐานโดยขั้น แรก แบ่งสัญญาณอนาลอคเป็นคาบเวลา (time period) เท่า ๆ กัน โดยแต่ละส่วนเรียกว่าเฟรม(frame) แล้วแต่ละเฟรมจะถูกแบ่งย่อยออกเป็นคาบเวลาเล็ก ๆ เรียกว่า เซกเมนต์ (segment) และในทุก ๆ เฟรม ของอินพุทจะขแครมบลิ่งเซ็กเมนท์เหล่านี้ ด้วยวิธีการสลับลำดับ(permutation) ดังแสดงใน รูป 2.5 ซึ่งในที่นี้แต่ละเฟรมจะถูกแบ่งออกเป็น 8 เซกเมนท์



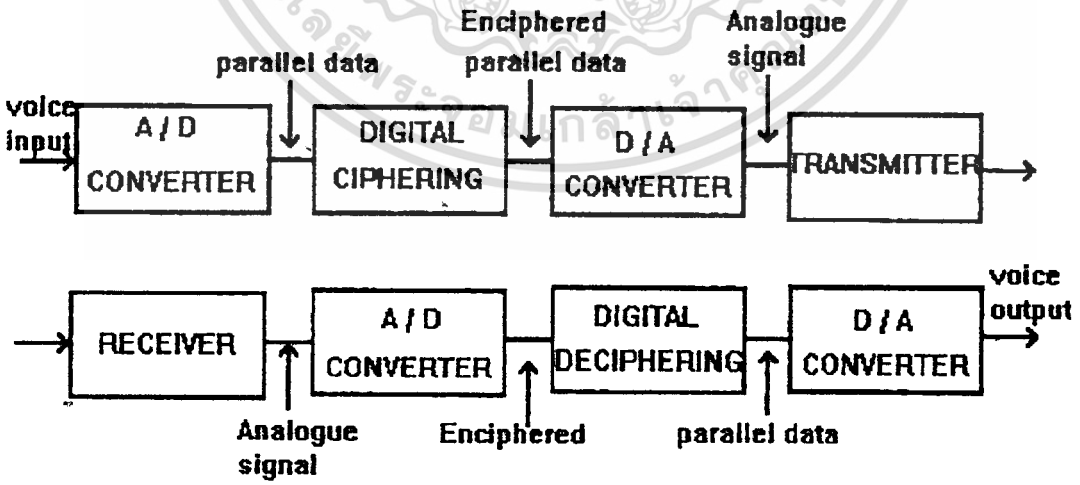
รูปที่ 2.5 Time Element Scrambler

การเข้ารหัสเสียงพูดด้วยวิธีทางดิจิตอล (Digtal encrypting speech)



รูปที่ 2.6 Digital cipher system

จากรูปที่ 2.6 เสียงที่ถูกเปลี่ยน (convert) เป็นสัญญาณดิจิตอลจะอยู่ในรูปของ Serial data stream ซึ่งอาจจะเป็น 64 K bit/S , 32 K bit/S , 16 K bit/S , 9.6 K bit/S , 4.8 K bit/S หรือ 2.4 K bit/S แต่ถ้าอัตราความเร็วของบิท (bit rate) ขนาด 9.6 K bit/S และที่สูงกว่า จะเป็นการเพิ่มแบนวิดของสัญญาณทำให้เพิ่มความยุ่งยากในการนำไปใช้งานโดยจะต้องพิจารณาถึงระบบเชื่อมโยงการส่งผ่าน (transmission link) ที่สามารถตอบสนองต่อสัญญาณที่ใช้ได้สำหรับอัตราความเร็วของบิทที่ต่ำกว่า 9.6 K bit/S สามารถที่จะนำมาใช้ได้ก็จริง แต่ทั้งนี้ ประสิทธิภาพสัมพันธ์กับการลดรูปของการจำรูปแบบของเสียง (Reduction in voice recognition) ขบวนการที่ยุ่งยากและซับซ้อน ต้องใช้อุปกรณ์จำนวนมาก และมีราคาแพง

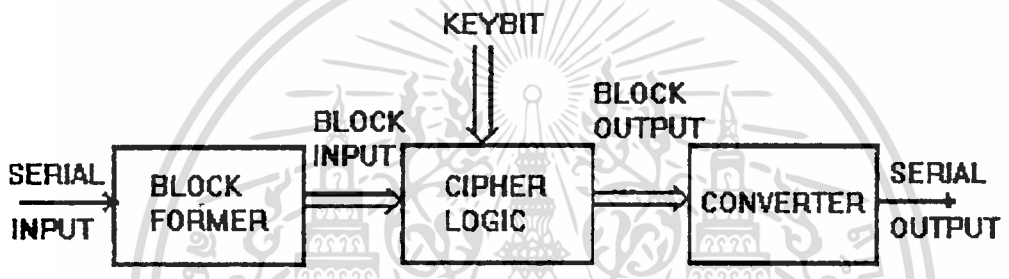


รูปที่ 2.7 Modify digital cipher system

จากรูปที่ 2.7เป็นการดัดแปลงโดยเสียงที่ถูกเปลี่ยนเป็นสัญญาณดิจิตอลแล้วผ่านเข้ารหัสทางดิจิตอล (Digital ciphering) จากนั้นเปลี่ยนสัญญาณดิจิตอลกลับมาเป็นอานาลอค โดยผ่าน D/A และจึงส่งผ่านให้กับตัวส่ง (transmitter) ส่วนทางด้านรับก็เพิ่ม A/D เข้าไป จะเห็นแบนด์วิดของสัญญาณไม่ถูกเปลี่ยนแปลงไป

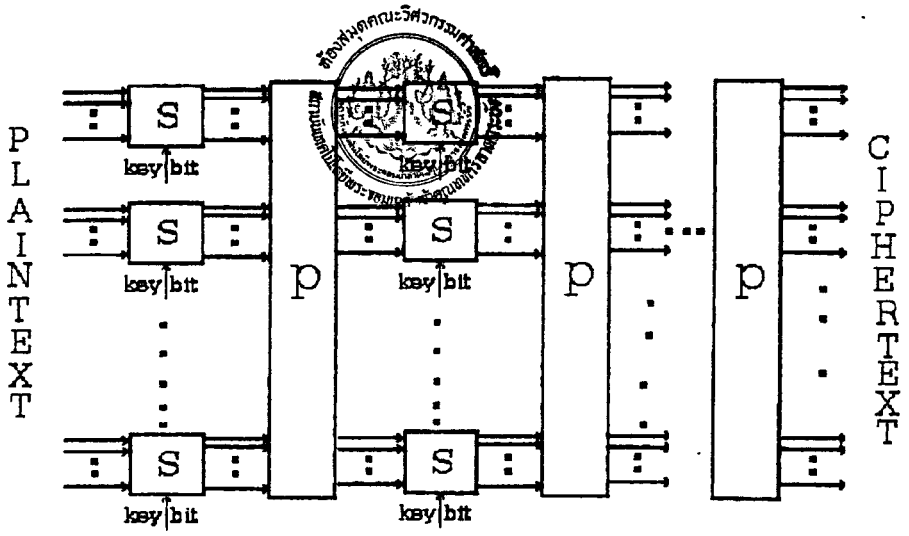
1.บล็อคไซเฟอร์ (block cipher)

บล็อคไซเฟอร์เป็นวิธีการเข้ารหัสแบบดิจิตอลอย่างง่าย จึงมีโอกาสที่จะถูกแกะออกมาได้ด้วยการวิเคราะห์ความถี่ของบล็อค ดังนั้น เพื่อเพิ่มความปลอดภัย ต้องใช้บล็อคที่มีขนาดใหญ่ ลักษณะทั่วไปของการเข้ารหัสและถอดรหัสวิธีนี้ แสดงให้เห็นในรูปนี้ 2.8

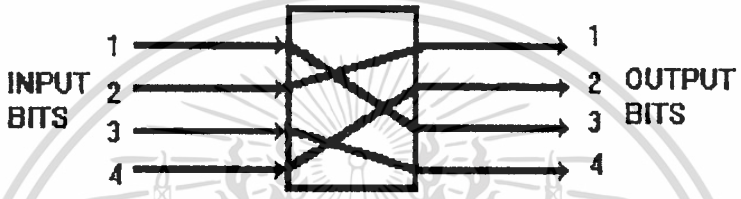


รูปที่ 2.8 รูปแบบทั่วไปของการเข้าและถอดรหัสแบบบล็อค

การกำหนดให้บล็อคมีขนาดใหญ่นั้น จำนวนบิทของคีย์ที่ใช้ต้องมีจำนวนมากขึ้นด้วย ถ้าขนาดของบล็อคมีค่าเท่ากับ n บิท จะมีบล็อครูปแบบต่าง ๆ กัน 2^n บล็อค เมื่อผ่านการเข้ารหัสแล้วจะได้บล็อคที่มีรูปแบบต่าง ๆ กัน 2^n บล็อค เมื่อผ่านการ $[\log(2^n!)]+1$ เพื่อลดจำนวนของคีย์ลง วิธีการหนึ่ง คือ จะต้องใช้รูปแบบการเข้ารหัสแน่นอน ดังตัวอย่างในรูปที่ 2.9บล็อคของข้อความที่นำมาเข้ารหัสจะถูกแปลงเปลี่ยนไปอย่างต่อเนื่อง สลับกันระหว่างกลุ่มของ S boxes และ P boxes ภายใน S boxes จะประกอบไปด้วยดิจิตอลลอจิค ซึ่งอาจจะให้ค่าเอาท์พุทเป็นฟังก์ชั่นบูลีนของอินพุทกับคีย์ ส่วน P boxes เป็นการจัดเรียง หรือสับเปลี่ยนที่ของบิทอินพุท ซึ่งได้แสดงไว้ในรูปที่ 2.10



รูปที่ 2.9 ระบบการเข้ารหัสแบบบล็อคที่มีหลายสเตจ



รูปที่ 2.10 การสลับจำนวน 4 บิท ของ P BOX

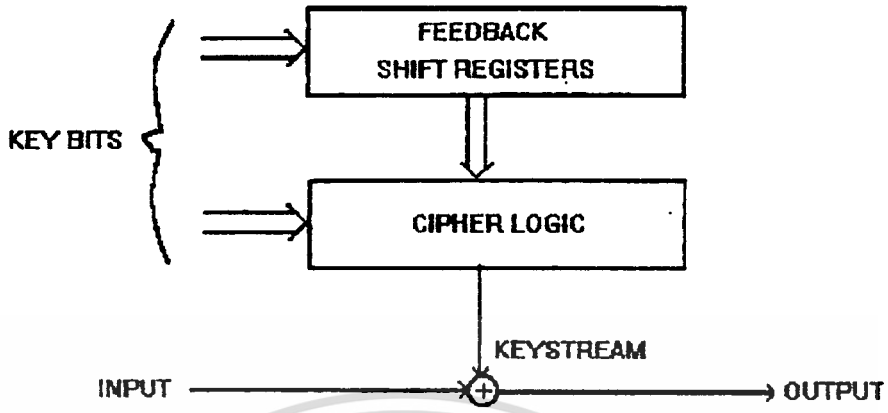
ถ้าภายใน P boxes ไม่มีการใช้ค่าคีย์ จำนวนบิทของคีย์ที่ต้องการใช้สำหรับการเข้ารหัสมีค่าเท่ากับจำนวนของ S boxes ทั้งหมด การถอดรหัสทำได้โดยการนำข้อมูลในลำดับที่กลับผ่านเข้าไปในบล็อคต่างๆ เหล่านี้ ซึ่งจะมีการเรียงย้อนกลับกันจากการเข้ารหัส ในทางปฏิบัติแล้วจำนวนของบล็อคต่าง ๆ ที่ใช้ในระบบนี้จะถูกจำกัดด้วยค่าเวลาด้วยค่าเวลาที่ใช้ในการเข้าหรือถอดรหัสในแต่ละบล็อค

2. ซิงค์โครนัส ไซเฟอร์ (synchronous cipher)

การเข้ารหัสต่อเนื่อง (stream cipher) มีอยู่ 2 ประเภท คือ ประเภทที่ค่าของคีย์สตรีม (key steam) ขึ้นอยู่กับค่าของข้อมูลที่นำมาเข้ารหัส และประเภทที่ค่าของคีย์สตรีมเป็นฟังก์ชันของข้อมูลที่นำมาเข้ารหัส ประเภทแรกนั้นเรียกว่า ซิงค์โครนสไซเฟอร์ (synchronous cipher) เพราะว่ามันต้องการซิงค์กันระหว่างค่าของคีย์กับข้อมูลอินพุท เพื่อทำให้ถอดรหัสเป็นไปอย่างถูกต้อง

ค่าของคีย์สตรีมถูกสร้างขึ้นมาอย่างสะเปะสะปะ มีความยาวเท่ากับข้อมูลที่นำมาเข้ารหัส และเพราะว่าค่าของคีย์ที่ผ่านไปแล้ว จะไม่มีการนำมาใช้ซ้ำอีก การเข้ารหัสวิธีนี้จึงเรียกว่า one-time tape หรือ one-time pad การเปลี่ยนค่าคีย์ให้ถี่ขึ้น หรือยาวขึ้นจะไม่มีผลในการนำไปใช้จริง ดังนั้น จะใช้จำนวนบิทของคีย์และฟีดแบคชิฟท์รีจิสเตอร์ในจำนวนที่พอดี เพื่อสร้างค่าคีย์สตรีม รูป

แบบในการเข้า และถอดรหัส วิธีนี้ ได้แสดงไว้ในรูปที่ 2.11 ซึ่งเครื่องหมายพวกแสดงถึงการทำมอโดลู-ทู (modulo-two)



รูปที่ 2.11 รูปแบบทั่วไปของการเข้าและถอดรหัสแบบซิงค์โครนัส

ลิเนียร์ฟีดแบคชิฟท์รีจิสเตอร์ จะสร้างคีย์สตรีมค่าสะเปะสะปะที่มีความยาวต่อเนื่องกันไป แต่อย่างไรก็ตาม โครงสร้างการทำงานแบบลิเนียร์นี้ทำให้ง่ายต่อการที่จะวิเคราะห์หาข้อมูลที่จริงออกมาได้ และยังมีข้อจำกัดของจำนวนชิฟท์รีจิสเตอร์ในการนำไปใช้งาน ดังนั้น จึงเปลี่ยนการทำงานในบางส่วนให้มีการทำงานแบบนอน-ลิเนียร์ เช่น ในส่วนของเอาท์พุทหรือส่วนฟีดแบค เพื่อ ลดจำนวนของชิฟท์รีจิสเตอร์ให้น้อยลง รูปที่ 2.12 และรูปที่ 2.13 แสดงให้เห็นถึงการสร้างคีย์

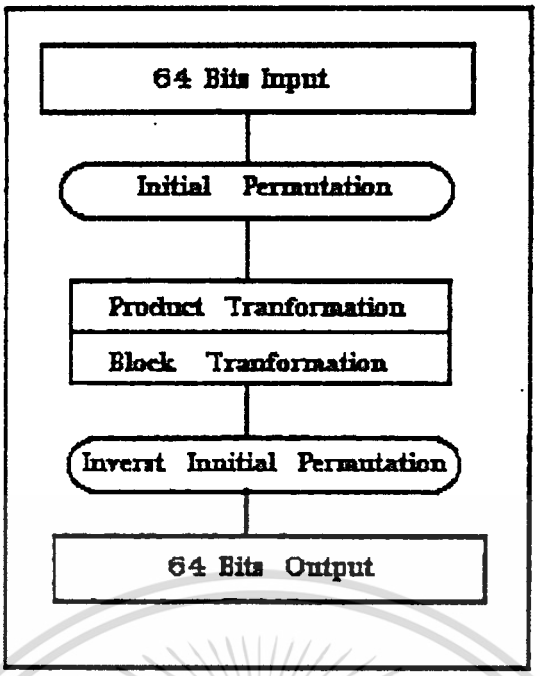
บทที่ 3 ทฤษฎีและหลักการทำงาน

ในบทนี้จะกล่าวถึงทฤษฎีต่าง ๆ ที่ได้นำมาประยุกต์ใช้ในโครงงานนี้ ได้แก่ ANALOG TO DIGITAL CONVERSION ซึ่งใช้ในการแปลงสัญญาณเสียงเป็น DIGITAL เมื่อนำมาเข้ารหัสแบบ DIGITAL , DIGITAL TO ANALOG CONVERTION การแปลงกลับในส่วนของสัญญาณที่ถอดรหัสแล้วที่เครื่องรับเป็นสัญญาณเสียงที่สามารถฟังได้ การเข้ารหัสแบบมาตราฐาน (DATA ENCRYPTION STANDARD) เดินการเข้ารหัสที่มีความซับซ้อนของบิทมาก ใช้ในทางคอมพิวเตอร์อย่างแพร่หลาย รวมถึงการใช้ค่าคีย์ 64 บิท (ในโครงงานนี้ดัดแปลงเหลือ 48 บิท) ทำให้การแกะหาข้อมูลเดิมเป็นไปได้ยากมาก ในส่วนของการเข้ารหัสจะเป็นฮาร์ดแวร์ล้วน เพราะการเข้ารหัสทางซอฟท์แวร์ ใช้เวลาในการเข้ารหัสมาก ทำให้เสียเวลาในการสุ่มสัญญาณ (sampling) สูง ส่วนของฮาร์ดแวร์ส่วนใหญ่จะเป็นอุปกรณ์ทางลอจิก เช่น การ latch data เก็บไว้ การ exclusive or ของ DATA และทฤษฎีของ 8031 ซึ่งเป็นไมโครคอนโทรเลอร์ที่ ควบคุมการทำงานของระบบทั้งหมด ให้ประสานงานสอดคล้องด้วยดี

3.1 การเข้ารหัส แบบมาตรฐาน DES (Data Encryption Standsrd)

3.1.1 หลักการเบื้องต้น

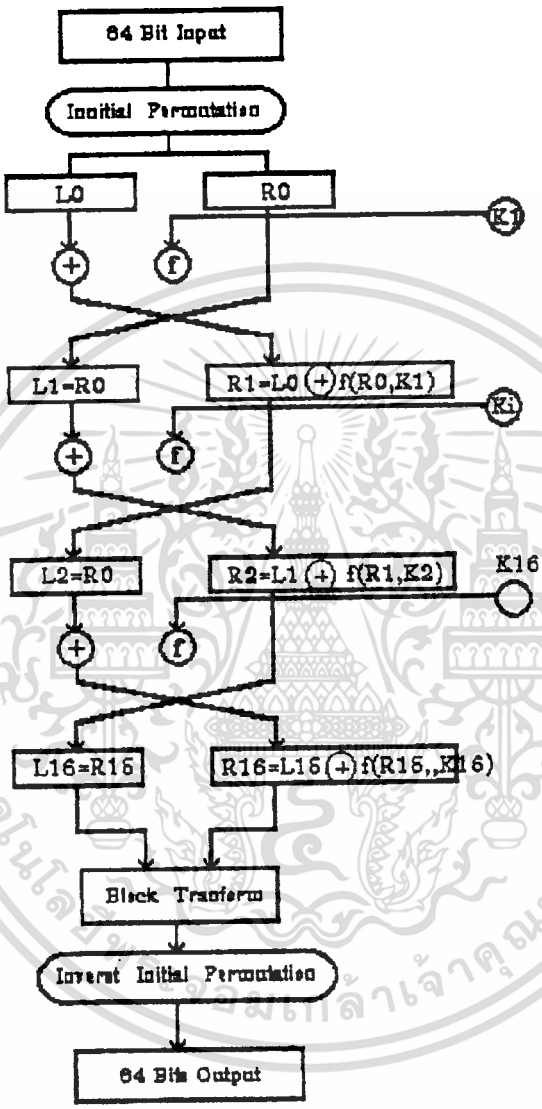
รูปที่ 3.1 เป็นแผนภูมิแสดงการทำงานของการเข้ารหัสแบบดีอีเอสแบบพื้นฐานข้อความที่ต้องการเข้ารหัส (plaintext) ทีละ 64 บิท จะถูกเพอร์มิวเทด (permutate) เสียก่อนให้บิทสลับที่กันแล้วจึงทำ product transformation ซึ่งเป็นส่วนที่ยุ่งยากที่สุด ส่วน block transformation เป็นเพียงการสลับที่ 32 บิท ชีกซ้ายชีกขวาแล้ว ขั้นตอนสุดท้าย คือ การหา inverse initial transfortion กลับกับการเพอร์มิวเทดขั้นแรก ก็จะได้ข้อความที่เข้ารหัส ออกมาตามต้องการ



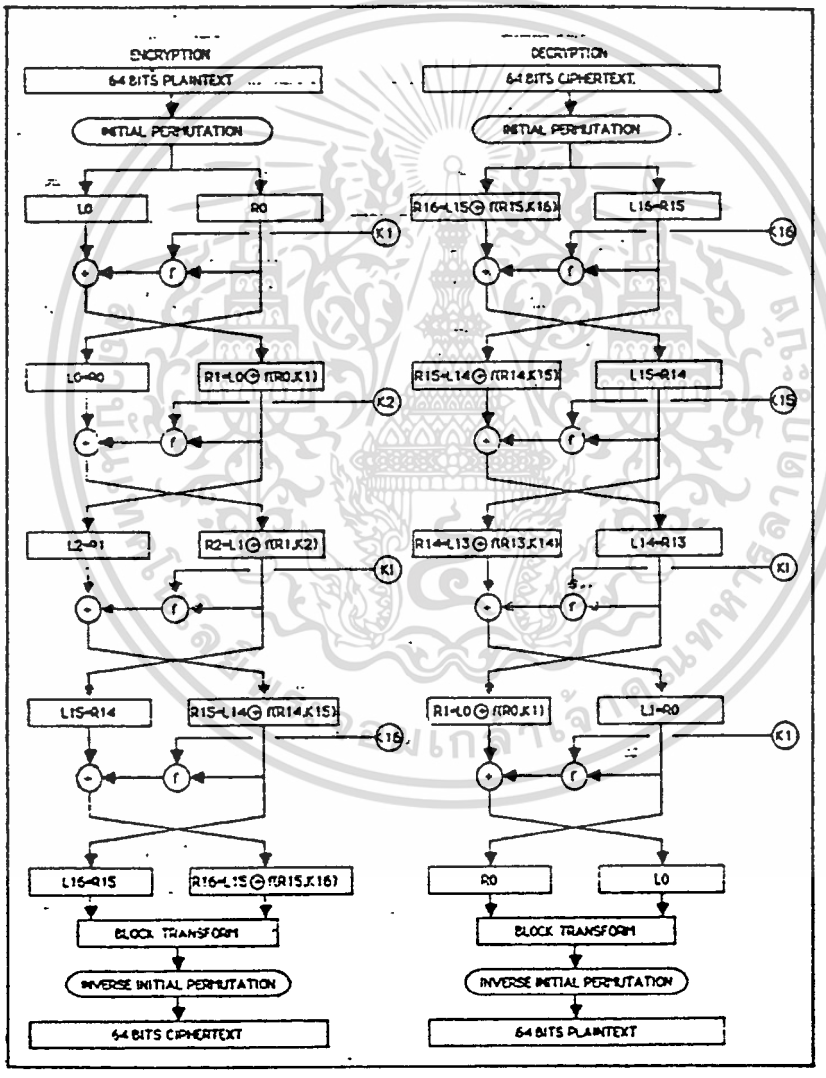
รูปที่ 3.1 โฟล์ชาร์ทแสดงการเข้ารหัสดีอีเอส

แผนผังการทำงานที่ละเอียดขึ้นในรูปที่ 3.2 แสดงให้เห็นว่าหลังจากการเพอร์มิวเทคขั้นแรกแล้ว ข้อมูล 64 บิท จะถูกแบ่งครึ่งออกเป็นซีกซ้าย (L) และซีกขวา (R) ดังที่เขียนไว้ว่า L0 และ R0 ก็หมายถึง L กับ R ในรอบที่ 0 ก่อนจะเริ่มตั้งต้น จากนั้น L0 กับ R0 ก็จะถูกคลุก เคล้าผสมกับคีย์ (K1 ถึง K16) ด้วยฟังก์ชัน f และสลับที่ซ้ายขวา รวมทั้งสิ้น 16 รอบ ด้วยกัน ก็จะได้ได้เป็น L1 ถึง L16 และ R1 ถึง R16 เป็นอันเสร็จ product transfortion ส่วน block transfortion นั้นก็คือ การสลับที่ L16 กับ R16 เท่านั้นเอง

รูปที่ 3.2 โฟล์ชาร์ทแสดงรายละเอียดอัลกอริธึมของรหัสดีอีเอส



f ในรูปที่ 3.6 เป็นฟังก์ชันที่เราจะมาดูรายละเอียดกันในหัวข้อที่ 3 ส่วน K1 นั้น K16 เป็นคีย์ย่อย ๆ ยาว 48 บิท ที่ได้มาจากคีย์ใหญ่ ซึ่งยาว 56 บิท แต่จะได้มาอย่างไรนั้น เราจะดูรายละเอียดกันในหัวข้อที่ 4 สำหรับเครื่องหมายบวกในวงกลมก็คือ การบวกแบบมอดูโลสอง (bitwise exclusive OR) ในกรณีนี้ พูดง่ายๆ ก็คือ การเอ็กคลูซีฟออร์ทีละบิทนั่นเอง คุณสมบัติอย่างหนึ่งของระบบดีอีเอส คือ เป็นระบบที่ทั้งเข้า และถอดรหัสได้ ดังเช่น ในรูปที่ 3.7 จะเห็นได้ว่า การถอดรหัสนั้นทำได้ง่ายมาก เพียงแต่ป้อนข้อความที่เข้ารหัสแล้ว (ciphertext) เข้าทางอินพุทแล้วเรียงสลับทิศของคีย์ย่อยเสียใหม่ ก็จะได้เอาท์พุทเป็นข้อความที่ถอดรหัสออกมา แล้ว



รูปที่ 3.3 โฟล์ชาร์ทของการเข้าและถอดรหัสดีอีเอส

3.1.2 เพอร์มิวเทชันขั้นแรกและขั้นสุดท้าย

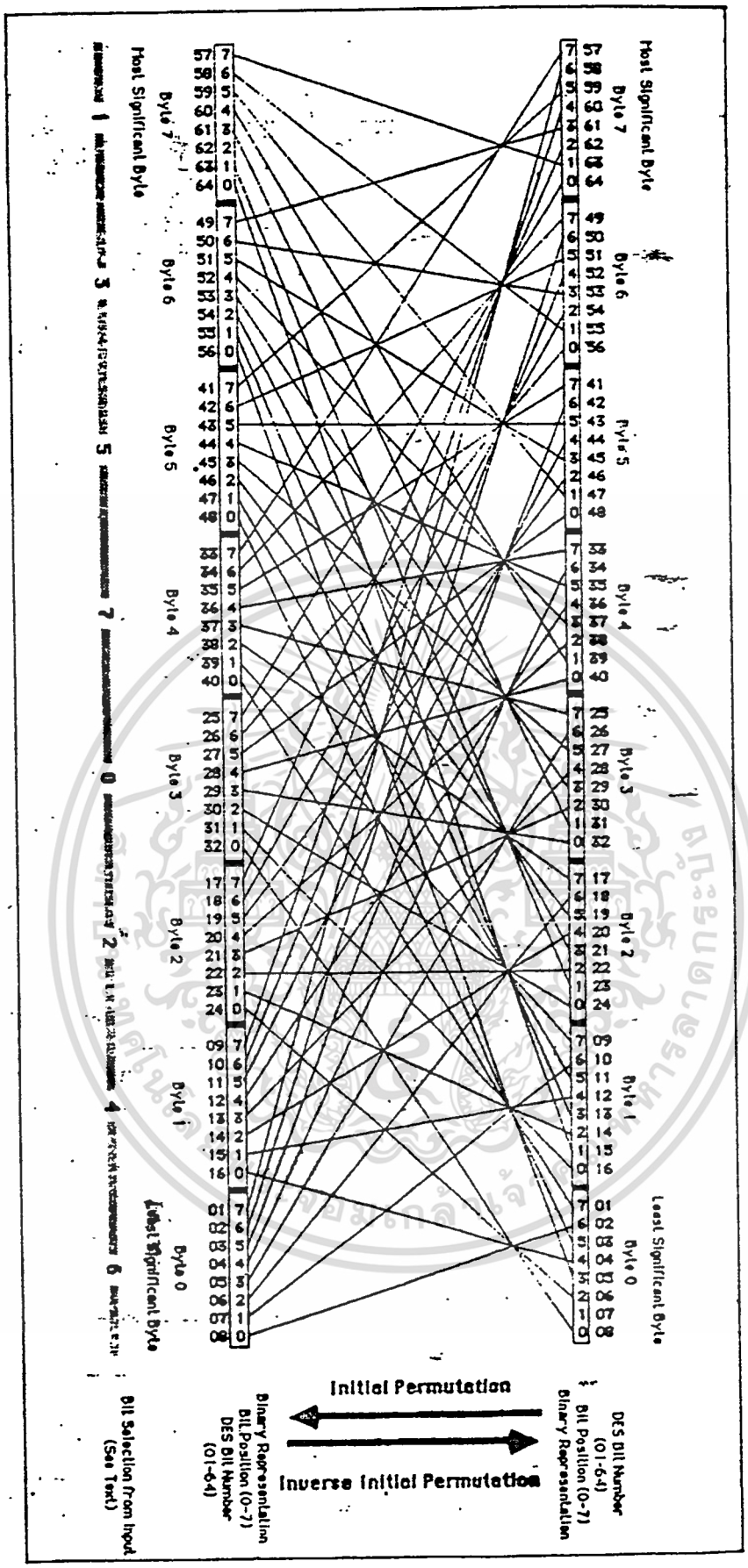
ในรูปที่ 3.4ก และ 3.4ข แสดงตารางเพอร์มิวเทชันขั้นต้น และขั้นสุดท้าย ซึ่งกลับทิศกัน (inverse) กับขั้นต้น ถ้าเรานำหมายเลขบิทของอินพุทกับเอาท์พุทไปเปลี่ยนเป็นหมายเลขบิทแบบไบนารีธรรมดาแล้วลากเส้นเชื่อมโยงจากอินพุทไปยังเอาท์พุท ก็จะได้ผลตามรูปที่ 3.4ค ซึ่งจะเห็นได้โดยง่ายว่า แต่ละไบท์ของเอาท์พุทจะเป็นบิทในตำแหน่งต่าง ๆ ที่มาจากอินพุทตามลำคือ คือ 1-3-5-7-0-2-4-6 ซึ่งง่ายต่อการจดจำและทำความเข้าใจมากกว่าตารางในรูปที่ 3.4ก ส่วน ตาราง inverse initial permutation ในรูปที่ 3.8ข ก็จะมีลักษณะกลับกันกับตารางใน รูปที่ 3.4ก หมายความว่า เพลน เท็กซ์ผ่านการเพอร์มิวเทชันครั้งหนึ่งแล้วนำผลที่ได้ไปผ่าน การเพอร์มิวเทคกลับ (inverse permutation) ก็จะได้เพลนเท็กซ์ของเดิมกลับมาให้อยู่ใน รูปของหมายเลขบิทแบบไบนารีธรรมดา

IP							
58	50	42	34	26	18	10	02
60	52	44	36	28	20	12	04
62	54	46	38	30	22	14	06
64	56	48	40	32	24	16	08
57	49	41	33	25	17	09	01
59	51	49	35	27	19	11	09
61	53	45	37	29	21	13	05
63	55	47	39	31	23	15	07

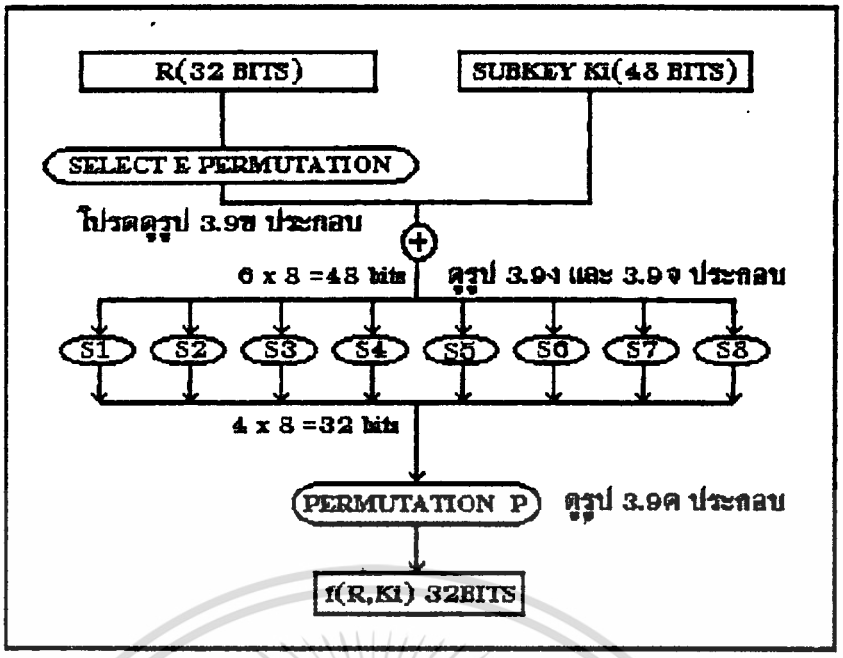
IP^{-1}							
40	08	48	16	56	24	64	32
39	07	47	15	55	23	63	31
38	06	46	14	54	22	62	30
37	05	45	13	53	21	61	29
36	04	44	12	52	20	60	28
35	03	43	11	51	19	59	27
34	02	42	10	50	18	58	26
33	01	41	09	49	17	57	25

รูปที่ 3.4ก ตาราง Initial Permutationอ่านจากซ้ายไปขวาและบนลงล่าง เช่น บิท1 ของ permuted data มีค่าเท่ากับ บิท 58 ของข้อมูลที่ป้อนเข้าไป

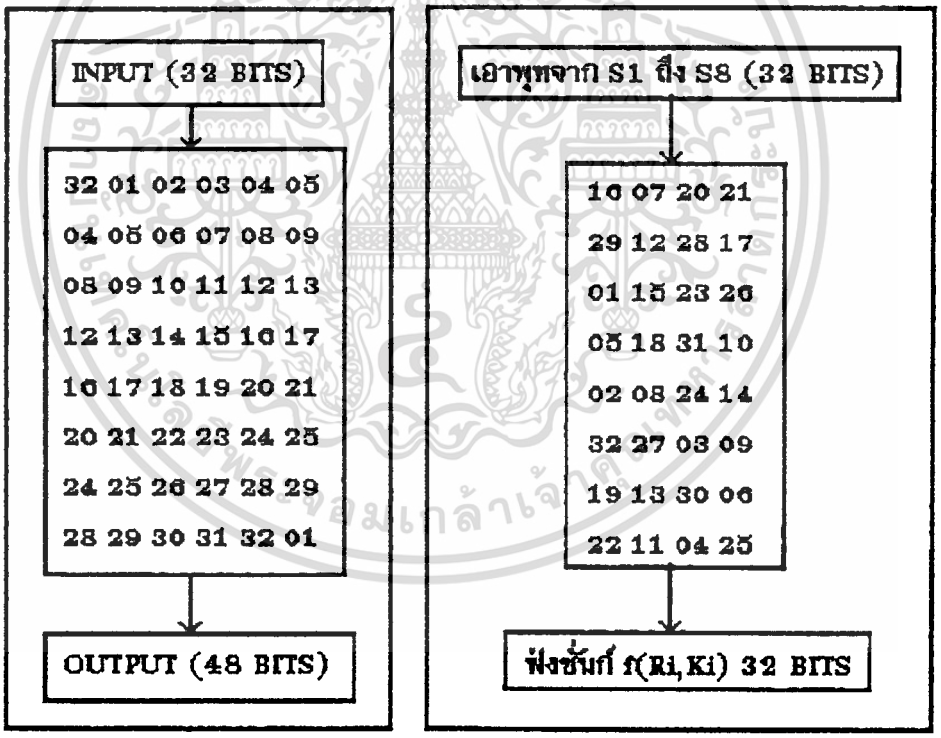
รูปที่ 3.4ข ตาราง Inverse Initial Permutation อ่านเช่นเดียวกับตารางในรูปที่ 3.4 ก จะเห็นได้ว่าบิทหมายเลข 58 ของเอาท์พุทมีค่าเท่ากับบิท 1 ของ อินพุท คือ กลับกันกับตารางในรูปที่ 3.4ก



รูปที่ 3.4 ความสัมพันธ์ระหว่างอินพุทกับเอาท์พุทของตาราง Initial Permutation และ Inverse Initial Permutatio



รูปที่ 3.5ก โฟล์ชาร์ทแสดงรายละเอียดฟังก์ชัน f(Ri-1,Ki)



รูปที่ 3.5ข ตาราง Select E Permutation Table ยืดข้อมูล 32 บิทของ R ในกลายเป็น 48 บิท

รูปที่ 3.5ค ตาราง Permutation P เปลี่ยนข้อมูลที่ได้จาก S1 ถึง S8 ให้ เป็น f(Ri-1,Ki)

S1

14	04	13	01	02	15	11	08	03	10	06	12	05	09	00	07
00	15	07	04	14	02	13	01	10	06	12	11	09	05	03	08
04	01	14	08	13	06	02	11	15	12	09	07	03	10	05	00
15	12	08	02	04	09	01	07	05	11	03	14	10	00	06	13

S2

15	01	08	14	06	11	03	04	09	07	02	13	12	00	05	10
03	13	04	07	15	02	08	14	12	00	01	10	06	09	11	05
00	14	07	11	10	04	13	01	05	08	12	06	09	03	02	15
13	08	10	01	03	15	04	02	11	06	07	12	00	05	14	09

S3

10	00	09	14	06	03	15	05	01	13	12	07	11	04	02	08
13	07	00	09	03	04	06	10	02	08	05	14	12	11	15	01
13	06	04	09	08	15	03	00	11	01	02	12	05	10	14	07
01	10	13	00	06	09	08	07	04	15	14	03	11	05	02	12

S4

07	13	14	03	00	06	09	10	01	02	08	05	11	12	04	15
13	08	11	05	06	15	00	03	04	07	02	12	01	10	14	09
10	06	09	00	12	11	07	13	15	01	03	14	05	02	08	04
03	15	00	06	10	01	13	08	09	04	05	11	12	07	12	14

S5

02	12	04	01	07	10	11	06	08	05	03	15	13	00	14	09
14	11	02	12	04	07	13	01	05	00	15	10	03	09	08	06
04	02	01	11	10	13	07	08	15	09	12	05	06	03	00	14
11	08	12	07	01	14	02	13	06	15	00	09	10	04	05	03

S6

12	01	10	15	09	02	06	08	00	13	03	04	14	07	05	11
10	15	04	02	07	12	09	05	06	01	13	14	00	11	03	08
09	14	15	05	02	08	12	03	07	00	04	10	01	13	11	06
04	03	02	12	09	05	15	10	11	14	01	07	06	00	08	13

S7

04	11	02	14	15	00	08	13	03	12	09	07	05	10	06	01
13	00	11	07	04	09	01	10	14	03	05	12	02	15	08	06
01	04	11	13	12	03	07	14	10	15	06	08	00	05	09	02
06	11	13	08	01	04	10	07	09	05	00	15	14	02	03	12

S8

13	02	08	04	06	15	11	01	10	09	03	14	05	00	12	07
01	15	13	08	10	03	07	04	12	05	06	11	00	14	09	02
07	11	04	01	09	12	14	02	00	06	10	13	15	03	05	08
02	01	14	07	04	10	08	13	15	12	09	00	03	05	06	11

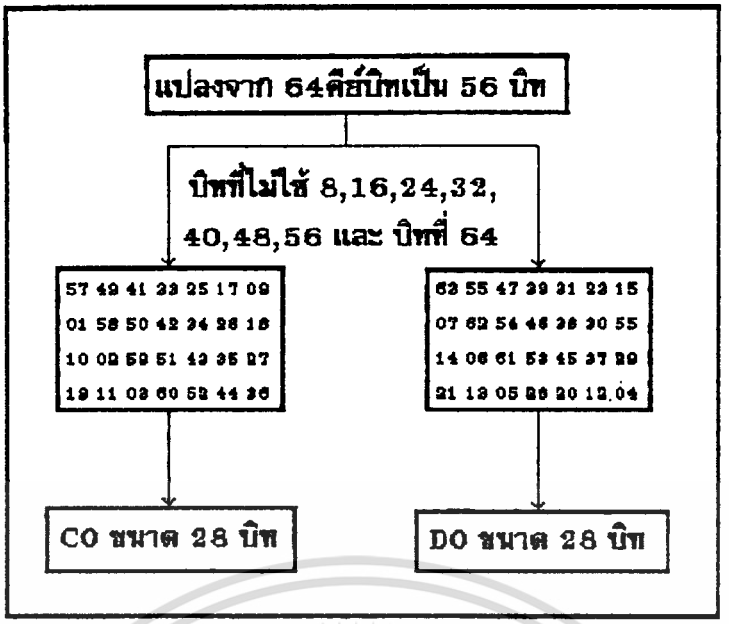
รูปที่ 3.5ง ตารางฟังก์ชัน S1 ถึง S8 โปรดดูวิธีใช้ในรูปที่ 3.5จ

3.1.3 การคำนวณฟังก์ชัน f(Ri-1,Ki)

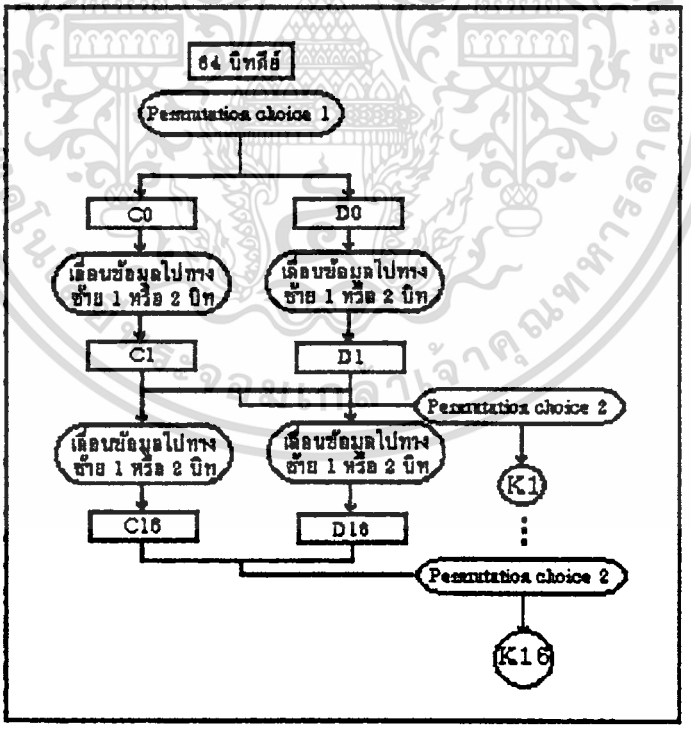
ฟังก์ชัน f(Ri-1,Ki) มีรายละเอียดดังแสดงไว้ในรูปที่ 3.5 ก ขั้นแรก Ri-1 จะถูกเพอร์มิวเทตเสียก่อนด้วยตาราง Select E ที่แสดงไว้ในรูปที่ 3.9ข ซึ่งจะเปลี่ยนข้อมูล 32 บิทของ Ri-1 ให้เป็น 48 บิท เพื่อที่จะนำมาบวกแบบมอดูโลสองกับคีย์ย่อย Ki ที่ยาว 48 บิท เหมือนกัน จากนั้นผลบวก 48 บิทจะถูกยุบให้เหลือ 32 บิท ด้วยฟังก์ชัน S ดังจะอธิบายต่อไป ขั้นสุดท้าย คือ การเพอร์มิวเทตด้วยตาราง Permutation P ดังแสดงไว้ในรูปที่ 3.9ค ได้ผลลัพธ์ออกมาเป็น f(Ri-1,Ki)

รูปที่ 3.5จ การใช้ตาราง S1 ถึง S8 ยกตัวอย่างว่า เรามีข้อมูลป้อนเข้าเท่ากับ 010001 เรา ก็ ดึงหลักต้นกับหลักท้ายมาเป็นเลขบอกแถว และสี่หลักกลางมาเป็นเลขบอกคอลัมน์ อ่านจากตาราง (สมมติว่า S1) ได้ผลลัพธ์ออกมาเป็น 10 ฐานสิบ ก็เท่ากับ 1010 นั่นเอง อนึ่งจะเห็นได้ว่า เป็นการเปลี่ยนแปลงข้อมูล 6 บิทให้เหลือ 4 บิทด้วย

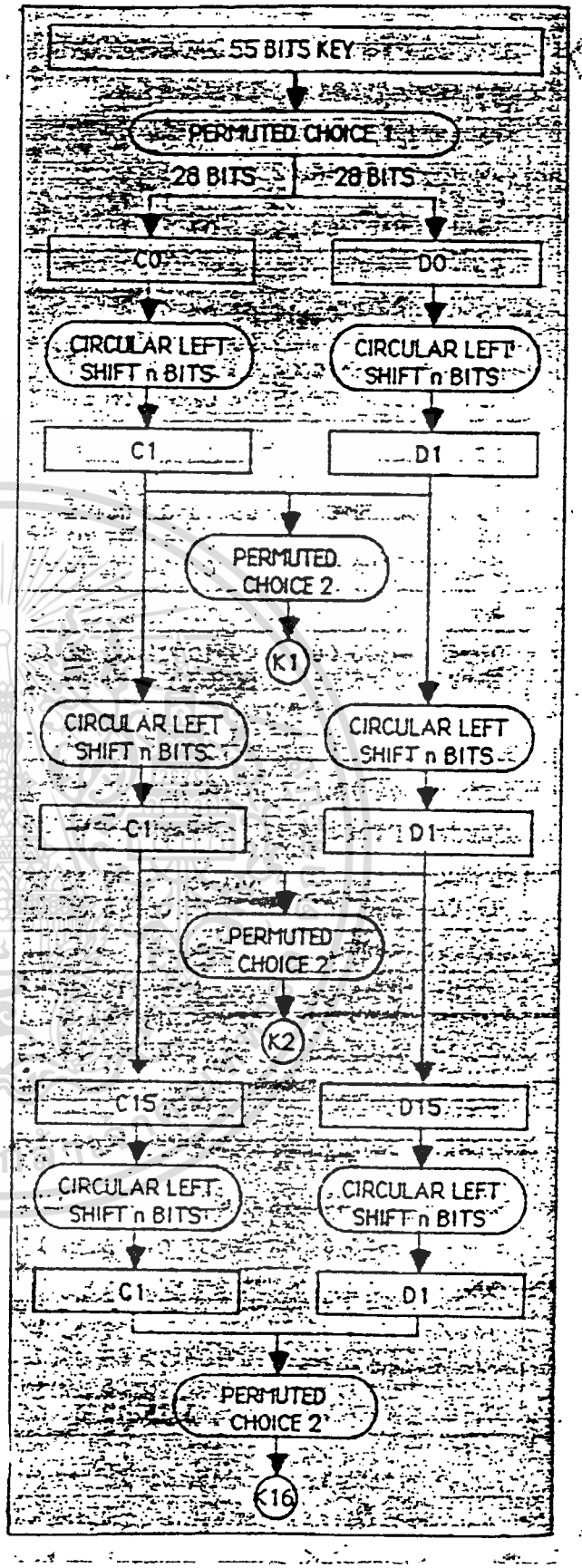
การยุบข้อมูล 48 บิทให้เหลือ 32 บิท นั้นทำได้โดยใช้ตารางแฟ็ดตารางทำงานให้ขนานกับแต่ละตารางมีอินพุท 6 บิท และเอาท์พุท 4 บิท เรียกว่า Selection function S1 ถึง S8 ดังแสดงไว้ในรูปที่ 3.5ง แต่ละตารางมีวิธีอ่านเหมือนกันดังแสดงไว้ในรูปที่ 3.5จ ส่วนตาราง Permution P ขั้นสุดท้ายนั้น ก็แสดงไว้ในรูปที่ 3.9จ ซึ่งไม่มีอะไรพิเศษมาก เพียงแต่เปลี่ยนข้อมูล 32 บิทที่ได้จาก S1 ถึง S8 ให้เป็นผลลัพธ์ f(Ri-1,Ki) ซึ่งมีความยาว 32 บิทเช่นกัน



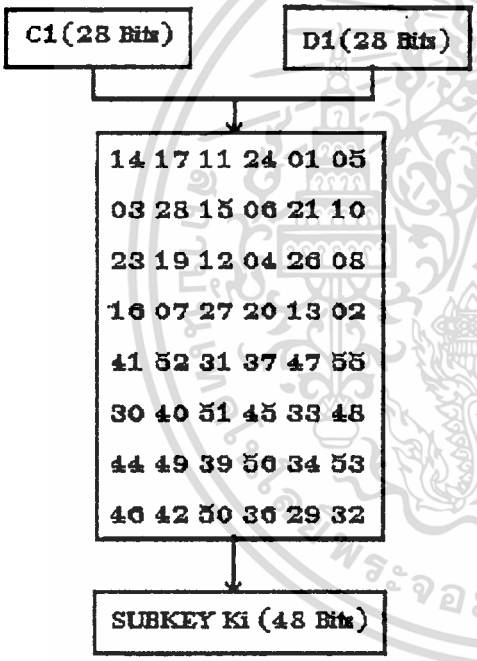
รูปที่ 3.6ข ตาราง Permuted choice 1
ซึ่งแยกคีย์ออกเป็น C0 กับ D0
แต่ละส่วนยาว 28 บิท



รูปที่ 3.6ค ตารางแสดงจำนวนครั้ง
ของการหมุนข้อมูลไบนารีไปทาง



รูปที่ 3.6ก โฟล์ชาร์ทแสดงการคีย์ย่อย ๆ จากคีย์ใหญ่ยาว 56 บิท ที่ผู้ใช้ป้อนเข้าไป



รูปที่ 3.6ง การสร้าง Subkey Ki จาก Ci และ Di ด้วยตาราง Permuted choice 2

3.1.4 การสร้างคีย์ย่อยจากคีย์ใหญ่

ในขณะใช้งาน ผู้ใช้จะต้องป้อนคีย์ยาว 56 บิท เข้าไปเป็น K ที่เครื่องจะใช้สำหรับสร้างคีย์ย่อย ๆ (K) ขึ้นมาแสดงในรูปที่ 3.6ก แรกทีเดียวคีย์จะถูกเพอร์มิวเทตด้วยตาราง

Permuted choice 1 ซึ่งแสดงในรูปที่ 3.10ข ออกมาเป็นข้อมูลสองชุด แต่ละชุดยาว 28 บิท เรียกว่า CO และ DO จากนั้น CO และ DO จะถูกหมุนไปทางซ้ายไม่หนึ่งก็สองบิท ขึ้นอยู่กับครั้งของขั้นตอนการทำ product transformation ซึ่งแสดงไว้ในตารางของรูปที่ 3.6ค ขั้นสุดท้ายคือการเพอร์มิวเทตด้วยตาราง Permuted choice 2 ซึ่งช่วยยุบข้อมูล 28 บิทสองชุด เข้าเป็นคีย์ย่อย ยาว 48 บิท ดังแสดงไว้ในรูปที่ 3.6ง

3.2 คุณสมบัติ MCS-51

รายละเอียดต่าง ๆ ของ 8031

8031 เป็น CPU ที่อยู่ในตระกูล MCS-51 ของอินเทลที่ออกแบบมาให้ใช้งานได้สะดวกมาก โดยต่ออุปกรณ์ประกอบเพียงเล็กน้อยก็สามารถทำงานได้ รวมทั้งยังมีวงจรนับ/ตั้งเวลาและพอร์ตรับส่งข้อมูลแบบอนุกรมอยู่ภายในตัว CPU ด้วย แต่ในเทอมนี้ศึกษาเฉพาะความสามารถของ CPU ที่จะสามารถรับส่งแบบ full duplex ได้ ในส่วนของฮาร์ดแวร์และซอฟท์แวร์นี้ จะดำเนินการในเทอมที่สอง ซึ่งส่วนนี้เป็นส่วนสำคัญในการควบคุมการทำงานทั้งหมดของวงจร และใช้เทคนิค D.E..S (data encryption standard) ได้ใช้ไมโครโปรเซสเซอร์ 8031 เขียนโปรแกรมออกมารายละเอียดคร่าว ๆ ของ 8031 มีดังนี้

คุณสมบัติของ 8031

- สามารถอ้างแอดเดรสเป็นบิทได้ (bit addressable)
- มีแรมภายในขนาด 128 ไบท์
- มีวงจรตั้งเวลา/วงจรนับ ขนาด 16 บิท 2 ตัว
- กำหนดเป็น UART (Universal Synchronous Asynchronous Receiver Transmitter) ที่รับส่งข้อมูลอนุกรมได้สองทิศทาง
- สายอินพุท และเอาท์พุทมีจำนวน 32 เส้น
- อินเตอร์รัพท์ แบ่งออกเป็น 2 ระดับ จาก 5 แหล่ง
- มีสัญญาณนาฬิกาภายใน
- มีแอดเดรสของหน่วยความจำสำหรับเก็บโปรแกรม (Program memory) จำนวนทั้งหมด 64 กิโลไบท์
- มีแอดเดรสของหน่วยความจำสำหรับเก็บข้อมูล (Data memory) จำนวนทั้งหมด 64 กิโลไบท์
- มีรีจิสเตอร์ทั้งหมด 32 แบ่งเป็น 4 แบงค์ (blank) แบงค์ละ 8 ตัว

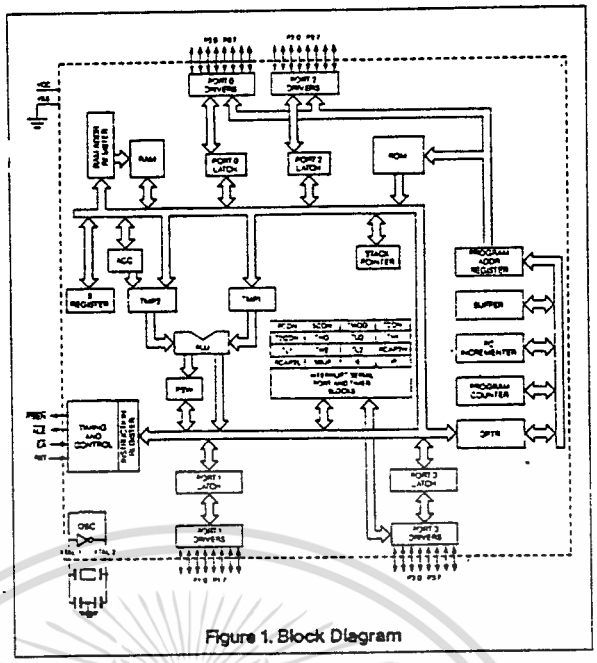
ขาต่าง ๆ ของ 8031

- **Vcc**

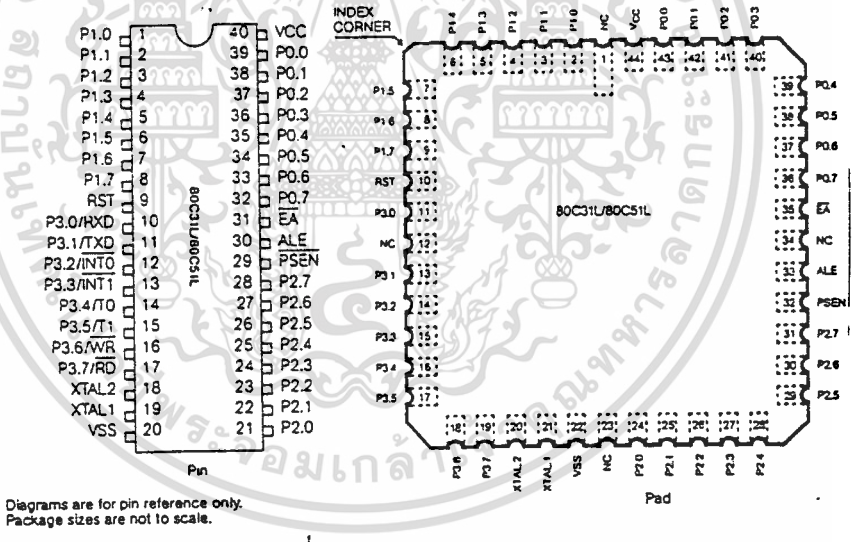
เป็นขาไฟเลี้ยงบวก

- **Vss**

เป็นขาไฟเลี้ยงลบ



รูปที่ 3.7 โครงสร้างของ 8031



รูปที่ 3.8 รูปแสดงการจัดขาอุปกรณ์

- Port 0

พอร์ต 0 เป็นพอร์ต 1/0 ขนาด 8 บิท 2 ทิศทาง สามารถต่อเข้ากับ TTL แบบ LS ได้ 8 ตัว พอร์ต 0 นี้จะถูกใช้มัลติเพล็กซ์ระหว่างค่าแอดเดรสไบท์ต่ำกับบัสข้อมูล เมื่อมีการแอดแซสกันกับหน่วย ความจำภายนอกทั้งโปรแกรมเมมโมรี และดาต้าเมมโมรี

- Port 1

พอร์ต 1 เป็นพอร์ต I/O ขนาด 8 บิท ซึ่งมีการพลูอัพ (Pull up) ภายในเอาท์พุทบัฟเฟอร์ของพอร์ต 1 สามารถต่อกับ TTL ได้ 4 ตัว เมื่อมีการส่งค่า "1" มายังพอร์ต 1 ขา ของพอร์ต 1 นี้นี้จะมีค่าสูงเนื่องจากมีการพูลอัพภายใน และในสภาวะนี้สามารถใช้เป็นอินพุทได้

- Port 2

พอร์ต 2 นี้เป็นเป็นพอร์ต I/O ขนาด 8 บิท 2 ทิศทางเหมือนกับพอร์ต 1 ทุกประการ แต่เมื่อมีการต่อกับโปรแกรมเมมโมรี่ภายนอก พอร์ต 2 นี้จะให้ค่าแอดเดรสสูงออกมาเพื่อใช้ในการเฟทช์รหัส (code) จากโปรแกรมเมมโมรี่นี้ ถ้ามีการแอคเซสกับดาต้าเมมโมรีโดยใช้คำสั่งที่มีการอ้างแอดเดรสขนาด 16 บิท (MOVX @DPTR) ค่าของพอร์ต 2 นี้ก็จะเป็นค่าของแอดเดรสไบท์สูงเหมือนกัน ยกเว้นเมื่อมีการแอคเซสกับดาต้าเมมโมรีด้วยคำสั่งที่มีค่าแอดเดรสขนาด 8 บิท (MOVX @Ri) พอร์ต 2 นี้จะเป็นค่าของ P2 ซึ่งอยู่ในพื้นที่ของ SFR

- Port 3

นอกจากพอร์ต 3 จะเป็นพอร์ต I/O ขนาด 8 บิท 2 ทิศทางเหมือนกับพอร์ต 1 แล้ว ใน แต่ละขาของพอร์ต 3 ยังมีหน้าที่พิเศษอีกดังต่อไปนี้

P3.0 RxD รับข้อมูลของพอร์ตอนุกรม

P3.1 TxD ส่งข้อมูลของพอร์ตอนุกรม

P3.2 INT0 รับสัญญาณอินเตอร์รัพท์ภายนอกหมายเลข 0

P3.3 INT1 รับสัญญาณอินเตอร์รัพท์ภายนอกหมายเลข 1

P3.4 T0 อินพุทของไทม์เมอร์/เคาน์เตอร์ 0

P3.5 T1 อินพุทของไทม์เมอร์/เคาน์เตอร์ 1

P3.6 WR สร้างสัญญาณในการเขียนดาต้าเมมโมรี

P3.7 RD สร้างสัญญาณในการอ่านดาต้าเมมโมรี่

- RST (Reset)

เป็นขารีเซ็ท ซึ่งการรีเซ็ทจะต้องให้ขานี้มีสภาวะ "1" เป็นเวลาอย่างน้อยมากกว่า 2 แมชีนไซเคิล (machine cycle) ในขณะที่วงจรกำเนิดสัญญาณนาฬิกาทำงานอยู่

- ALE (Address latch enable)

เป็นขาเอาท์พุทที่จะสร้างพัลซ์สำหรับแลทช์ค่าของแอดเดรสไบท์ต่ำ ในเวลาแอดเซสกับหน่วยความจำภายนอก ในภาวะปกติ ALE จะสร้างสัญญาณพัลซ์ออกมาทุก ๆ 1/6 ของความถี่นาฬิกา ซึ่ง เราอาจจะนำไปใช้เป็นสัญญาณนาฬิกาภายนอก หรือเป็นฐานเวลาของวงจรตั้งเวลาภายนอกได้ แต่อย่างไรก็ตาม เมื่อ CPU มีการกระทำคำสั่งเกี่ยวกับเมมโมรีภายนอก พัลซ์ของ ALE จะหาย ไป 1 พัลซ์เสมอ

- PSEN (Program strobe enable)

PSEN จะสร้างสัญญาณที่ใช้สำหรับอ่านค่าจากโปรแกรมเมมโมรีภายนอก เมื่อมีการเอ็กซิคิว (execute)คำสั่งจากโปรแกรมเมมโมรีภายนอก จะมีสัญญาณ PSEN เกิดขึ้น 2 ครั้งใน 1 แมชชีน ไซเคิลยกเว้นเมื่อมีการกระทำคำสั่งเกี่ยวกับข้อมูลที่อยู่ในดาต้าเมมโมรีภายนอกจะไม่มีสัญญาณ PSEN ออกมา

- EA (External access enable)

ขา EA นี้เมื่อต่อกับ Vcc CPU จะทำการเฟทช์คำสั่งจากโปรแกรมเมมโมรีที่อยู่ภายใน และเมื่อต่อเข้ากับ Vss CPU จะทำการเฟทช์คำสั่งจากโปรแกรมเมมโมรีที่อยู่ภายนอก มีแอดเดรสเริ่มต้นที่ 0000H จนถึง FFFFH ซึ่ง 8031 จะต้องต่อขานี้เข้ากับ Vss เท่านั้น

- XTAL1

เป็นอินพุทของวงจรกำเนิดสัญญาณนาฬิกาภายใน 8031

- XTAL2

เป็นเอาท์พุทของวงจรกำเนิดสัญญาณนาฬิกาภายใน 8031

ไทม์เมอร์/เคาน์เตอร์ (TIMER/COUNTER)

8031 มีรีจิสเตอร์ไทม์เมอร์/เคาน์เตอร์ ขนาด 16 บิท อยู่ 2 ตัว เราสามารถกำหนดให้ มันทำงานเป็นไทม์เมอร์ หรือเคาน์เตอร์ได้อย่างใดอย่างหนึ่ง

เมื่อทำหน้าที่เป็นไทม์เมอร์ รีจิสเตอร์ตัวนี้จะเพิ่มค่าของตัวมันเองในทุกแมชีนไซเคิล เหตุนี้ทำให้ดูเหมือนว่ามันคือ เคาน์เตอร์ที่มีแมชีนไซเคิลเป็นอินพุท มื่อใน 1 แมชีนไซเคิลประกอบด้วยสัญญาณนาฬิกาจำนวน 12 ลูก ดังนั้นอัตราในการนับจะมีค่าเท่ากับ 1/12 ของความถี่ออสซิลเลเตอร์

เมื่อทำหน้าที่เป็นเคาน์เตอร์ รีจิสเตอร์นี้จะมีการเพิ่มค่าก็ต่อเมื่อขาอินพุท T0 หรือ T1 มี การเปลี่ยนแปลงสภาวะจาก "1" เป็น "0" ซึ่งการตรวจสอบสภาวะนี้จะใช้เวลาในช่วง S5P1 ของทุกแมชีนไซเคิล ถ้าในแมชีนไซเคิลใดขาอินพุท (T0 หรือ T1) มีค่าเป็น "1" และในแมชีน ไซเคิลถัดมาขาอินพุทมีค่าเป็น "0" เคาน์เตอร์จะเพิ่มค่าในรีจิสเตอร์ขึ้นอีก 1 จะเห็นได้ว่าต้อง ใช้ถึง 2 แมชีนไซเคิลในการดีเทคสภาวะ "1" เป็น "0" ดังนั้นอัตราการนับสูงสุดคือ 1/24 ของความถี่ออสซิลเลเตอร์

การเลือกให้รีจิสเตอร์นี้มีการทำงานเป็นไทม์เมอร์หรือเคาน์เตอร์ ก็โดยการกำหนดค่าบิท C/T ใน SFR TMOD และบิท M1 และ M0 ใน TMOD จะเป็นการเลือกโหมดการทำงานของไทม์ เมอร์/เคาน์เตอร์ ซึ่งมีทั้งหมด 4 โหมดด้วยกัน คือ

โหมด 0 13 บิท ไทม์เมอร์/เคาน์เตอร์

โหมด 1 16 บิท ไทม์เมอร์/เคาน์เตอร์

โหมด 2 8 บิท ออโต้รีโหลด (auto reload)

โหมด 3 ไทม์เมอร์ 0 จะเป็น 8 บิท ไทม์เมอร์ 2 ตัว

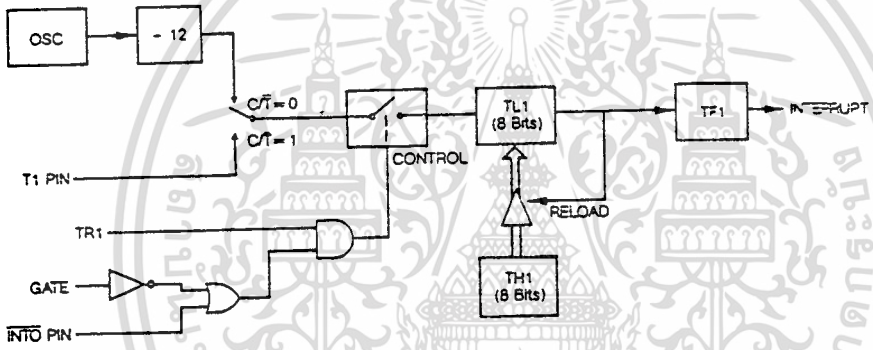
เคาน์เตอร์ 0 จะเป็น 8 บิท เคาน์เตอร์ 1 ตัว

ไทม์เมอร์/เคาน์เตอร์ 1 จะไม่ทำงาน

ในโครงงานนี้ จะใช้ในโหมด 2 เพราะฉะนั้น จะกล่าวคร่าวของการทำงานโหมด 2 คือ

โหมด 2

ในโหมด 2 ริจิสเตอร์จะเป็นแบบ 8 บิท โดยที่ TL1 จะสามารถโหลดข้อมูลจาก TH1 ได้ ใหม่ (AUTO-RELOAD) เมื่อเกิดโอเวอร์โฟลวจาก TL1 (ดูรูปที่) โดยที่ค่าใน ธTH1 จะไม่ ถูกเปลี่ยน การทำงานอื่น ๆ จะเหมือนกับโหมด 0



รูปที่3.9 Timer mode 2

ในรูปที่ เป็นไดอะแกรมของวงจร Timer 1 ใน 8051 ที่ทำงานโหมด 2 Timer 0 และ Timer1 มีการทำงานในโหมด 2 เหมือนกัน โดยจะสามารถกำหนดให้ทำหน้าที่เป็น Timerหรือ Counter ได้ โดยบิท C/T และควบคุมการนับได้โดยข้อมูลในบิท TR1 และ GATE ในรีจิสเตอร์ TMOD กับสัญญาณที่ขา INTx เมื่อเริ่มการทำงานข้อมูลในรีจิสเตอร์ TH1 จะถูกโหลด (Load) ไปยังรีจิสเตอร์ TL1 ทำให้รีจิสเตอร์ TH1 และ TL1 มีค่าเหมือนกัน เมื่อเกิดการนับจำนวนไซเคิล ของสัญญาณที่ออกจากสวิตช์ Control จะทำให้ค่าจากการนับในรีจิสเตอร์ TL1เพิ่ม ขึ้นไปเรื่อย ๆ ทีละ 1 จนถึง 0FFH ในการนับครั้งต่อไปจะทำให้บิท TF1 ในรีจิสเตอร์ TCON ไม่เป็น 1 และข้อมูลในรีจิสเตอร์ TH1 จะถูกโหลดไปยังรีจิสเตอร์ TL1 เพื่อเป็นค่าเริ่มต้นใน การนับต่อไป

ในโหมด 2 นี้ จะมีรีจิสเตอร์ควบคุม Timer อยู่คือ TMOD Timer/Counter mode register ดังรายละเอียด ดังนี้

TMOD Timer /Counter mode register

ตำแหน่งหน่วยความจำภายในเท่ากับ 89H

TMOD เป็นรีจิสเตอร์ขนาด 8 บิท ที่มีหน้าที่ควบคุมการทำงานของ Timer 0 และ Timer 1 แต่ละบิทในรีจิสเตอร์นี้ มีความหมายเฉพาะดังรูปที่

GATE	C/T	M1	M0	GATE	C/T	M1	M0
TIMER 1				TIMER 0			

GATE When TRx (in TCON) is set and GATE = 1, TIMER/COUNTERx will run only while INTx pin is high (hardware control). When GATE = 0, TIMER/COUNTERx will run only while TRx = 1 (software control).

C/T Timer or Counter selector. Cleared for Timer operation (input from internal system clock). Set for Counter operation (input from Tx input pin).

M1 Mode selector bit. (NOTE 1)

M0 Mode selector bit. (NOTE 1)

NOTE 1 :

M1	M0	Operating Mode
0	0	0 13-bit Timer
0	1	1 16-bit Timer/Counter
1	0	2 8-bit Auto-Reload Timer/Counter
1	1	3 (Timer 0) TL0 is an 8-bit Timer/Counter controlled by the standard Timer 0 control bits. TH0 is an 8-bit Timer and is controlled by Timer 1 control bits.
1	1	3 (Timer 1) Timer/Counter 1 stopped.

รูปที่ 10 TMOD Timer/Counter Mode Register

ในรูปที่ M0 เป็นชื่อของบิท 0 และ GATE ทางซ้ายสุดเป็นชื่อของบิท 7 รีจิสเตอร์นี้แบ่งข้อมูลออกเป็น 2 ชุด ชุดละ 4 บิท คือ บิท 0-3 ใช้สำหรับควบคุมการทำงานของ Timer 0และบิท 4-7 ใช้ควบคุมการทำงานของ Timer 1 หน้าที่ในการควบคุม Timer ของแต่ละบิทที่มีชื่อเดียวกันจะเหมือนกัน

GATE เป็นบิทที่ใช้ควบคุมให้ Timer ทำงานหรือไม่ ถ้าบิทนี้ของ Timer x ถูกตั้งเป็น 1จะทำให้ Timer ทำงานก็ต่อเมื่อที่ขา INTx มีสภาวะลอจิกเป็น 1 และบิท TRx ในรีจิสเตอร์ TCON เป็น 1 ด้วย

C/T บิทนี้ใช้สำหรับเลือกการทำงานของ Timer ว่าจะใช้เป็น Timer หรือ Counter ถ้าบิทนี้เป็น 1 ก็หมายความว่า เลือกทำงานเป็น Counter ซึ่งจะนับจำนวนไซเคิลของสัญญาณที่ เข้ามาทางขา Tx

M1,M0 เป็น 2 บิทที่ใช้ร่วมกันเพื่อเลือกโหมดการทำงานของ Timer การทำงานโหมด 0,1 และ 2 ของ Timer 0 จะเหมือนกับ Timer 1 แต่ในโหมด 3 การทำงานของทั้งสองจะ ต่างกัน ค่าใน M1 และ M0 จะเลือกโหมดการทำงานดังนี้

M1 M0 การทำงาน

0 - 0 โหมด 0 รีจิสเตอร์ THx และ TLx ทำตัวเป็นตัวนับ 13 บิท ค่าจากการนับ 8 บิทบนมาจาก 8 บิทของ THx และอีก 5 บิทล่าง มาจากค่า 5 บิทของรีจิสเตอร์ TLx โดยที่ 3 บิทบนของ TLx จะไม่ต้องสนใจเลย

0 1 โหมด 1 รีจิสเตอร์ THx และ TLx ทำตัวเป็นตัวนับ 16 บิท ค่าจากการนับ 8 บิทบนอยู่ในรีจิสเตอร์ THx และค่าจากการนับ 8 บิทล่างอยู่ที่ TLx

1 0 โหมด 2 ในการนับของรีจิสเตอร์ TLx ขนาด 8 บิทเมื่อนับถึงค่าสูงสุดคือFFH เมื่อทำการนับต่อไปจะเกิดการ Overflow แล้วก็จะ Reload เอาข้อมูลจาก THx เข้าไปยัง TLx เพื่อเป็นค่าเริ่มต้นในการนับครั้งต่อไป

1 1 โหมด 3 การทำงานของ Timer 0 และ Timer 1 จะต่างกันดังที่จะกล่าวต่อไป

พอร์ตอนุกรม (SERIAL INTERFACE)

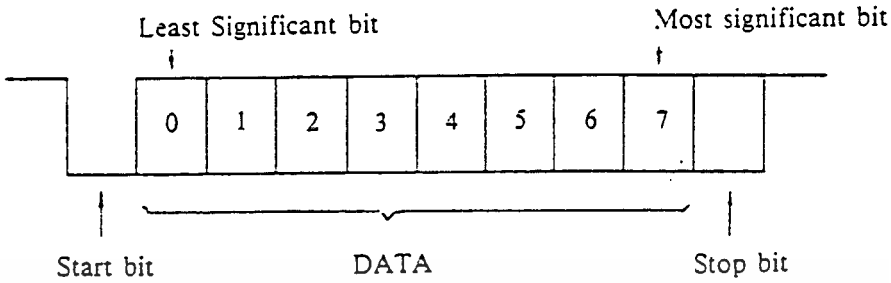
พอร์ตอนุกรมนี้เป็นแบบฟลูดูเพล็กซ์ (full duplex) คือ สามารถส่ง และรับข้อมูลได้ในเวลาเดียวกัน ในการรับจะมีบัฟเฟอร์ไว้สำหรับเก็บข้อมูล เมื่อมีการรับข้อมูลเข้ามาทุกครั้งมันจะถูกเก็บไว้ในบัฟเฟอร์ตัวนี้ ถ้าหากไม่มีการอ่านข้อมูลไบท์แรกที่รับได้มาเก็บไว้ จนกระทั่งมีข้อมูลไบท์ที่สองเข้ามา ค่าในบัฟเฟอร์จะเป็นค่าของข้อมูลไบท์ที่สอง ทำให้ข้อมูลไบท์แรกสูญหายไป ดังนั้น ทุกครั้งที่มีการรับข้อมูลเข้ามาจะต้องมีการอ่านค่าจากบัฟเฟอร์มาเก็บไว้เพื่อไม่ให้ข้อมูลเกิดการสูญหาย

บัฟเฟอร์ในการส่ง และรับข้อมูลนั้น เป็นตัวเดียวกันก็คือ SBUF การเขียนค่าลงใน SBUF เป็นการโหลดค่าให้กับรีจิสเตอร์ตัวส่งและข้อมูลในรีจิสเตอร์ตัวนี้จะถูกส่งผ่านออกทางขา TxD และการอ่านค่าจาก SBUF เป็นการกระทำกับรีจิสเตอร์ตัวรับซึ่งบรรจุข้อมูลที่รับได้ทางขา R xD ไว้ รีจิสเตอร์ทั้งตัวรับและตัวส่งไม่ใช่ตัวเดียวกัน

พอร์ตอนุกรมนี้ มีการทำงานทั้งหมด 4 โหมดด้วยกัน คือ

โหมด 0 ข้อมูลอนุกรมจะส่งและรับผ่านขา RxD ในขณะที่ขา TxD ให้เอาท์พุทเป็นสัญญาณนาฬิกาเพื่อใช้ในการเลื่อนข้อมูลที่มีขนาด 8 บิท ซึ่งจะถูกส่งและรับโดยมีบิทนัยสำคัญต่ำสุดเป็นบิทแรก ส่วนบอดเรท (buad rate) จะมีค่าคงที่เท่ากับ 1/12 ของความถี่ออสซิลเลเตอร์

โหมด 1 10 : บิทเริ่มต้น ("0") ,ข้อมูล 8 บิท (บิท LSB เป็นบิทแรก) และบิทหยุด ("1") จะถูกส่ง (ผ่านขา TxD) หรือถูกรับ (ผ่านขา RxD) ในการรับบิทหยุดจะเก็บอยู่ใน RB8 ซึ่งเป็นบิทหนึ่งของ SFR ที่ชื่อ SCON และสามารถเรียกบอดเรทได้



รูปที่ 11 ชุดข้อมูลอนุกรมในโหมด 1

โหมด 2 11 บิท : บิทเริ่มต้น ("0") ,ข้อมูล 8 บิท ,ข้อมูล 9 บิท ที่สามารถโปรแกรมได้ และบิทหยุด ("1") จะถูกส่ง (ผ่านขา TxD) หรือถูกรับ (ผ่านขา RxD) ใน การส่งข้อ มูล 9 บิท (TB8 ใน SCON) สามารถกำหนดให้มีค่าเป็น "0" หรือ "1" ได้ หรืออาจจะใช้ค่าของพาริตี้บิท (P ใน PSW) เป็นตัวกำหนดก็ได้ ในการรับข้อมูล 9 บิท จะเก็บไว้ที่บิท RB8 ของ SCON และจะไม่สนใจบิทหยุด ค่าบอดเรทสามารถกำ หนดได้เป็น 2 ค่า คือ 1/32 หรือ 1/64 ของความถี่ออสซิลเลเตอร์

รูปที่ 12 ชุดข้อมูลอนุกรมในโหมด 2

โหมด 3 มีการทำงานเหมือนกับโหมด 2 ทุกประการ ยกเว้นค่าบอดเรท ซึ่งในโหมดนี้สามารถที่จะเลือกค่าบอดเรทได้

ทั้ง 4 โหมดนี้ การส่งข้อมูลจะเริ่มต้นเมื่อมีคำสั่งใด ๆ ที่มี SBUF เป็นรีจิสเตอร์ปลายทาง ส่วนการรับ ในโหมด 0 จะเริ่มได้ด้วยเงื่อนไขที่ให้ RI=0 และ REN=1 ในโหมดอื่น จะเริ่มขึ้นเมื่อมีบิทเริ่มต้นเข้ามาในขณะที่ REN=1

ในการรับส่งจะมีรีจิสเตอร์ที่ควบคุมการรับส่ง คือ Register SCON (Serial Port Control Register) มีรายละเอียดดังนี้

SCON (Serial Port Control Register) ตำแหน่งหน่วยความจำภายในเท่ากับ 98H

รีจิสเตอรี SCON มีขนาด 8 บิท ใช้สำหรับควบคุมการส่งและรับข้อมูลผ่านทาง Serial Port แต่ละบิทของข้อมูลในรีจิสเตอร์นี้มีความหมายเฉพาะดังรูปที่ 5.5

SCON : SERIAL PORT CONTROL REGISTER. BIT ADDRESSABLE.

SM0	SM1	SM2	REN	TB8	RB8	TI	RI

SM0	SCON.7	Serial Port mode specifier.(NOTE 1).
SM1	SCON.6	Serial Port mode specifier. (NOTE 1).
SM2	SCON.5	Enables the multiprocessor communication feature in mode 2 & 3. In mode 2 or 3, if SM2 is set to 1 then RI will not be activated if the received 9th data bit (RB8) is 0. In mode 1, if SM2 = 1 then RI will not be activated if a valid stop bit was not received. In mode 0, SM2 should be 0. (See Table 9).
REN	SCON.4	Set/Cleared by software to Enable/Disable reception.
TB8	SCON.3	The 9th bit that will be transmitted in modes 2 & 3. Set/Cleared by software.
RB8	SCON.2	In modes 2 & 3, is the 9th data bit that was received. In mode 1, if SM2 = 0, RB8 is the stop bit that was received. In mode 0, RB8 is not used.
TI	SCON.1	Transmit interrupt flag. Set by hardware at the end of the 8th bit time in mode 0, or at the beginning of the stop bit in the other modes. Must be cleared by software.
RI	SCON.0	Receive interrupt flag. Set by hardware at the end of the 8th bit time in mode 0, or halfway through the stop bit time in the other modes (except see SM2). Must be cleared by software.

NOTE 1 :

SM0	SM1	Mode	Description	Baud Rate
0	0	0	SHIFT REGISTER	Fosc./12
0	1	1	8-Bit UART	Variable
1	0	2	9-Bit UART	Fosc./64 OR Fosc./32
1	1	3	9-Bit UART	Variable

SERIAL PORT SET-UP : Table 9

MODE	SCON	SM2 VARIATION
0	10H	Single Processor Environment (SM2 = 0)
1	50H	
2	90H	
3	D0H	
0	NA	Multiprocessor Environment (SM2 = 1)
1	70H	
2	B0H	
3	F0H	

รูป 13 Serial Port Control Register (SCON)

ในรูปที่ บิท RI จะเป็นชื่อของบิท 0 และ SMO จะเป็นบิท 7 ของรีจิสเตอร์ SCON ซึ่ง ความหมายหรือการทำงานของแต่ละบิทมีดังนี้

RI Receive Interrupt Flag

บิทนี้จะถูกกำหนดโดยฮาร์ดแวร์ให้มีค่าเป็น 0 หรือ 1 โดยที่ในการรับข้อมูลโหมด 0 นั้น

บิท RB8 จะมีค่าเป็น 1 เมื่อมีข้อมูลเข้ามาครบทั้ง 8 บิท ส่วนในโหมดอื่น บิท RB8 จะเป็น 1 ก็ต่อเมื่อข้อมูลเข้ามาถึงเวลาครึ่งหนึ่งของ Stop Bit (ยกเว้นบางกรณีให้ดูที่เรื่องบิท SM2 ของรีจิสเตอร์ SCON) บิทนี้จะสามารถ Clear ให้มีค่าเป็น 0 ได้ โดยใช้คำสั่ง CLR bit โดยค่าตำแหน่งของบิทมีค่าเท่ากับ 98H บิทนี้มีประโยชน์ให้รู้ว่า ข้อมูลได้เข้ามาอยู่ใน SBUF ครบทั้งชุดแล้วพร้อมที่ซีพียู จะอ่านไปเก็บในหน่วยความจำต่อไป หรืออาจกำหนดค่าในรีจิสเตอร์ IE และ IP เพื่อเมื่อมีข้อมูลเข้ามาทางพอร์ทอนุกรมแล้ว จะทำให้เกิดการขัดจังหวะ การทำงานของโปรแกรมหลัก (Main Program) แล้วกระโดดไปทำงานในโปรแกรมตอบสนองการขัดจังหวะ (Interrupt Service Routine)

TI Transmit Interrupt Flag

ค่าในบิท TI จะถูกกำหนดให้เป็น 1 หรือ 0 ด้วยฮาร์ดแวร์ โดยในการส่งข้อมูลแบบอนุกรมโหมด 0 จะเป็น 1 เพื่อจะบอกว่าการส่งข้อมูลในรีจิสเตอร์ SBUF ออกไปทางพอร์ทอนุกรมครบทั้ง 8 บิท แต่ถ้าเป็นการส่งข้อมูลแบบอนุกรมในโหมดอื่น จะทำให้ข้อมูลในบิท TI เป็น 1 เมื่อเริ่มการส่ง Stop Bit ข้อมูลบิทนี้จะสามารถ Clear เป็น 0 ได้ด้วยคำสั่ง CLR bit โดยที่ค่าแหน่งของบิทนี้เท่ากับ 99H บิทนี้ยังมีประโยชน์เพื่อบอกว่าการส่งข้อมูลจาก SBUF ออก ไปทางพอร์ทอนุกรมนั้นสิ้นสุดแล้ว พร้อมที่จะให้โปรแกรมเขียนข้อมูลลงไปยัง SBUF สำหรับการส่งออกต่อไปได้ นอกจากนี้การกำหนดค่าในรีจิสเตอร์ IE และ IP ยังสามารถที่จะกำหนดให้เกิดการขัดจังหวะการทำงานของโปรแกรมได้เมื่อบิทนี้ถูกฮาร์ดแวร์ทำให้มีค่าเป็น 1

RB8

เมื่อมีการกำหนดให้รับข้อมูลในโหมด 2 และ 3 จะใช้บิทนี้สำหรับเก็บข้อมูลบิทที่ 9 ที่เข้ามาทางพอร์ทอนุกรม ส่วนในโหมด 1 นั้น บิทนี้จะเก็บ Stop Bit ซึ่งมีค่าเป็น 1 นั่นเอง ใน โหมด 0 บิทนี้จะไม่ถูกใช้งาน ค่าตำแหน่งของบิทนี้คือ 9AH

TB8

ในการส่งข้อมูลแบบอนุกรมโหมด 2 และ 3 จะใช้บิทนี้เก็บข้อมูลบิทที่ 9 ส่วนโหมดอื่น จะไม่ใช้งานบิทนี้ การกำหนดค่าในบิทนี้ สามารถทำได้โดยใช้คำสั่ง SETB bit หรือ CLR bit ค่า ตำแหน่งของบิทนี้ คือ 9BH

REN Receive Enable

เป็นบิทที่ใช้กำหนดให้ทำการรับข้อมูลเข้ามาจากพอร์ทอนุกรม (Serial Port) หรือไม่ถ้า บิท

นี้เป็น 1 ก็จะรับข้อมูลเข้ามา แต่ถ้าเป็น 0 ก็จะมีรับข้อมูลที่ขา RXD เข้ามา การให้บิทนี้เป็น 1 หรือ 0 ทำได้โดยใช้คำสั่ง SETB หรือ CLR bit ค่าตำแหน่งของบิทนี้ คือ 9CH

SM2

เป็นบิทสำหรับควบคุมการทำงานของฮาร์ดแวร์ที่จะทำให้บิท RI เป็น 1 หรือไม่ ในกรณีที่บิท SM2 เป็น 0 ค่าในบิท RI ก็จะเป็นไปตามที่ได้อธิบายมาแล้วในเรื่องบิท RI แต่ถ้าบิท SM1 = 1

โหมด 2 และ 3 ซึ่งปกติแล้วบิท RI จะเป็น 1 เมื่อข้อมูลบิทที่ 9 เข้ามา แต่เมื่อ SM2 เป็น 1 แล้ว RI จะเป็น 1 ก็ต่อเมื่อข้อมูลบิทที่ 9 ที่เข้ามามีค่าเป็น 0 จะไม่ทำให้บิท RI มีค่าเป็น 1 (คือบิท RI จะเป็น 0)

ในโหมด 1 บิท RI มีค่าเป็น 1 เมื่อข้อมูล Stop Bit เข้ามายังพอร์ทอนุกรมถูกต้องแต่ ถ้า Stop Bit ไม่เข้ามายังพอร์ทอนุกรมอันอาจเกิดจากปัญหาในการส่งข้อมูลแล้วบิท RI จะมีค่าเป็น 0

ในโหมด 0 บิทนี้จะมีค่าเป็น 0 เสมอ

SM0,SM1

เป็น 2 บิทที่ใช้งานร่วมกัน เพื่อกำหนดโหมดของการรับ-ส่งข้อมูลของพอร์ทอนุกรม ค่าใน 2 บิทนี้จะกำหนดโหมดได้ดังนี้

SM0	-	SM1	-	MODE	-	Description
0	-	0	-	0	-	Shift register
0	-	1	-	1	-	8 -bit UART
1	-	0	-	2	-	9 -bit UART
1	-	1	-	3	-	9 -bit UART

การรับ-ส่งข้อมูลทางพอร์ทอนุกรม

ในการรับ-ส่งข้อมูลแบบอนุกรมผ่านทางพอร์ทอนุกรมนั้น จะต้องมีการกำหนดโหมดทางการทำงานในรีจิสเตอร์ SCON และในบางโหมดของการทำงานจะสามารถกำหนดอัตราการส่งข้อมูลได้ โดยการโปรแกรมใน Timer ข้อมูลที่จะส่งออกหรือรับเข้าทางพอร์ทอนุกรมจะอยู่ที่

รีจิสเตอร์ SBUF การทำงานของโครงงานนี้ใช้การรับส่งข้อมูลในโหมดที่ 1 เพราะเป็นโหมดที่ใช้งานง่าย และสะดวก โดยมีรายละเอียดดังนี้

การทำงานโหมด 1

การส่งข้อมูล

จากรูปที่ 5.18 บิท SMOD จะเป็นตัวเลือกว่าสัญญาณ Timer 1 Overflow ที่ส่งไปยัง วงจรวงจรหาร 46 จะถูกหาร 2 ก่อนหรือไม่ ถ้า SMOD เป็น 1 สัญญาณ Timer 1 Overflow จะ ไม่ถูกหาร แต่ถ้า SMOD เป็น 0 สัญญาณ Timer Overflow จะถูกหาร 2 ก่อนที่จะเข้าวงจร หาร 16 การส่งข้อมูลจะเริ่มจากการที่มีคำสั่งเขียนข้อมูลไปยังรีจิสเตอร์ SBUF จะมีสัญญาณ Write to SBUF เกิดขึ้นเพื่อรับข้อมูลจาก Internal Bus ด้านบนไปเก็บยังรีจิสเตอร์ SBUF และทำให้เอาท์พุทของ D FLIP FLOP ทางซ้ายของ SBUF มีค่าเป็น 1 และเป็นบิทที่ 9 ของ การส่งข้อมูล สัญญาณ Write to SBUF ยังส่งไปยัง TX control ด้วย ขณะนี้ข้อมูลในวงจร หาร 16 มีค่าเป็นอะไรไม่ทราบจึงจะรอจนกว่าข้อมูลในวงจรหาร 16 นับเพิ่มขึ้นจนถึงค่าสูงสุด แล้ววนกลับเป็น 0 คือเกิดการวนกลับ ทำให้เริ่มการส่งข้อมูลที่เวลา S1P1 ของไซเคิลเครื่องถัด ไป (การส่งข้อมูลอกจะสัมพันธ์กับการเกิด Overflow ในวงจรหาร 16) สัญญาณ SEND จาก TX Control เปลี่ยนสภาวะลอจิกเป็น 0 แล้วเริ่มส่งข้อมูลที่เป็น Start bit (0) ออกไป เมื่อส่ง Start Bit ออกไปแล้ว วงจร Tx Control ก็จะทำให้สัญญาณ DATA เป็น 1 เพื่อ เลื่อนข้อมูลใน SBUF ออกไป เริ่มจากบิท 0 จนถึงบิทที่ 7 การส่งข้อมูลนี้จะเกิดขึ้น เมื่อสัญญาณ Tx Clock เปลี่ยนสถานะจาก 0 เป็น 1 ดังในรูปที่ 5.18 ขณะที่ข้อมูลถูกเลื่อนออกไปนั้น จะมี 0 ถูกเลื่อนเข้ามาทางซ้ายของรีจิสเตอร์ SBUF เมื่อข้อมูลเลื่อนออกไปทั้ง 8 บิทแล้วบิทที่ 9 ซึ่ง เป็น 1 และตอนต้นอยู่ทางซ้ายสุด จะถูกเลื่อนมาอยู่ในตำแหน่งสุดท้ายทางขวาของรีจิสเตอร์ SBUF และทางซ้ายของหลักนี้จะมี 0 อยู่ทั้ง 8 บิทใน SBUF ทำให้ Zero Detector รู้ว่าเป็น ข้อมูลบิทสุดท้ายแล้วที่ส่งออก โดยจะมีสัญญาณมาบอกกับวงจร Tx Control ด้วย เมื่อ Tx Control ส่งสัญญาณ Shift ออกไปเป็นการส่งข้อมูลบิทสุดท้าย (บิท 7) ออกไป ก็จะรออีก 1 Tx Clock (Bit Clock) ก็จะทำให้ขา TXD ส่งข้อมูล Stop Bit(1) ออกมา สัญญาณ DATA ซึ่งมีสภาวะลอจิกเป็น 1 มาตั้งแต่เริ่มส่งข้อมูลบิท 0 ก็จะกลับเป็น 0 และบิท TI จะเป็น 1 เพื่อบอกการสิ้นสุดการส่งข้อมูลทั้งหมด จะสิ้นสุดการส่งข้อมูลทั้งหมดเมื่อสัญญาณ TX Clock ไซเคิลที่ 10 นับตั้งแต่สัญญาณ SEND เปลี่ยนสภาวะเป็นลอจิกเป็น 0

การรับข้อมูล

การรับข้อมูลจะขึ้นกับอัตราการเกิด Overflow ใน Timer 1 แล้วหาร 2 หรือไม่ขึ้น กับค่าของบิท SMOD สัญญาณนี้จะไปเข้าวงจรหาร 16 และเป็นตัวกำหนดอัตราการรับข้อมูล จะเริ่มจากวงจร 1-TO-0 Transition Detector พบว่าสัญญาณที่ขา RXD เปลี่ยนจาก1 เป็น 0 ซึ่งหมายถึงมีข้อมูล Start bit เข้ามา การตรวจสอบนี้ จะกระทำด้วยอัตรเดียวกับสัญญาณที่ เข้าวงจรหาร 16 เมื่อพบการเปลี่ยนสถานะลอจิกที่ขา RXD ก็จะเริ่มการรับข้อมูล ขณะนี้จะรีเซ็ทวงจรหาร 16 ให้มีค่าเป็น 0 เพื่อสร้างสัญญาณ RX Clock ให้เข้าจังหวะ (Synchronous) กับข้อมูลที่เข้ามาโดยสัญญาณ RX Clock จะเป็น 1 เมื่อการนับของวงจรหาร 16 มีค่าเป็น 15 ขณะที่วงจรหาร 16 นับถึง 7,8 และ 9 จะมีการตรวจสอบข้อมูลที่เข้ามาทางขา RXD เพื่อเป็นการตรวจว่าข้อมูลนั้นเป็นอะไร ถ้าอย่างน้อยข้อมูล 2 ใน 3 เป็นค่าเท่าใด ก็จะถือว่าข้อมูลที่เข้ามาเป็นค่านั้น ถ้าในการตรวจสอบ Start Bit แล้วพบว่าผิดพลาด คือ ไม่เป็น 0 ก็จะรีเซ็ทการทำงาน เพื่อไปตวจสอบการเปลี่ยนสถานะจาก 1 เป็น 0 ของข้อมูลที่ขา RXD ใหม่ แต่ถ้าพบ Start bit ก็จะเก็บข้อมูลทั้งหมดที่เข้ามาโดยเลื่อนข้อมูลเข้าไปยัง Input Shift Register ที่มีสัญญาณควบคุมการเลื่อนข้อมูล (Shift) ส่งมาจาก RX control ในตอนเริ่มต้น การรับข้อมูลจะมีการเขียนข้อมูล 1FFH ไปเก็บใน Input Shift Register ขณะที่ข้อมูลถูกเลื่อนเข้า ไปทางขวาของ Input Shift Register ก็จะมี 1 ถูกเลื่อนออกไปทางซ้ายทุกครั้งที่มีข้อมูล เข้ามา เมื่อ Start bit ที่รับเข้ามาถูกเลื่อนไปถึงซ้ายสุดของ Input Shift Register ก็ จะมีสัญญาณไปบอก RX Control Block หลังจากข้อมูลบิทสุดท้ายเข้ามาแล้วจะโหลด (Load)เอาข้อมูล 8 บิทไปเก็บในรีจิสเตอร์ SBUF พร้อมทั้ง Set ค่า RI และ RB8 ของรีจิสเตอร์ SCON แต่การโหลดข้อมูลไปเก็บนี้จะเกิดขึ้นได้ ก็ต่อเมื่อ

1. RI = 0 และ
2. SM2 = 0 หรือถ้า SM2 = 1 จะต้องได้รับ stop bit เป็น 1

ถ้าไม่มีสภาวะใดสภาวะหนึ่งดังกล่าวแล้ว ข้อมูลที่รับเข้ามาก็จะถูกทิ้งไป คือ ไม่โหลดไปเก็บไว้รีจิสเตอร์ SBUF ถ้ามีสภาวะดังกล่าวถูกต้อง Stop Bit จะถูกนำไปเก็บในรีจิสเตอร์ SBUF และบิท RI จะเป็น 1

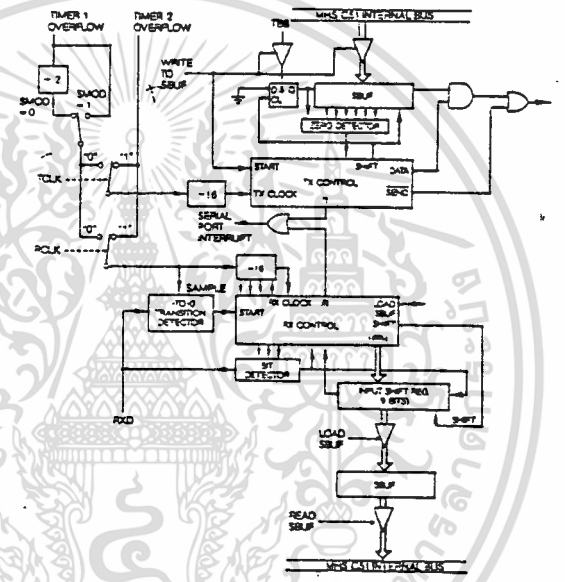
แต่ไม่ว่าทั้ง 2 กรณีจะเกิดหรือไม่ ก็จะกลับไปสู่การตรวจสภาวะเปลี่ยนจาก 1 เป็น 0 ที่ ขา RXD เพื่อรับข้อมูลต่อไป

ในการรับข้อมูลแบบอนุกรมโหมด 1 นี้ อัตราการส่งข้อมูลแต่ละบิท (Baund Rate) นี้จะ ต้อง Disable ไม่ให้เกิดการขัดจังหวะเนื่องมาจากการ Overflow Timer 1 อาจใช้ใน โหมดของ Timer หรือ Counter ก็ได้ ซึ่งเมื่อการนับในรีจิสเตอร์ตัวนับมีค่าสูงสุด แล้วกลับมา เป็น 0 ก็จะเกิด Overflow เช่นเดียวกัน แต่โดยปกติแล้ว จะใช้ Timer 1 นี้ ในโหมดของ Timer ที่มี

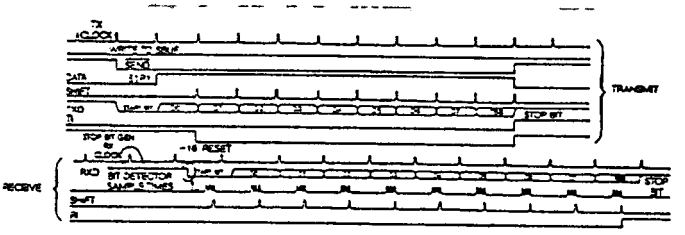
การทำงานแบบ Auto Reload 2 เพื่อว่าเมื่อค่าในการนับโดยรีจิสเตอร์ TL1 ถึงค่า สูงสุดก็จะโหลดค่าในรีจิสเตอร์ TH1 มาไว้ใน TL1 สำหรับเป็นค่าเริ่มต้นการนับต่อไป ซึ่ง Baud rate จะมีค่า

Baud rate $=2^{smod}$x(Timer 1 Overflowrate)/32

โดยที่ SMOD เป็นบิทหนึ่งในรีจิสเตอร์ PCON



รูป 14 Serial Port mode 1

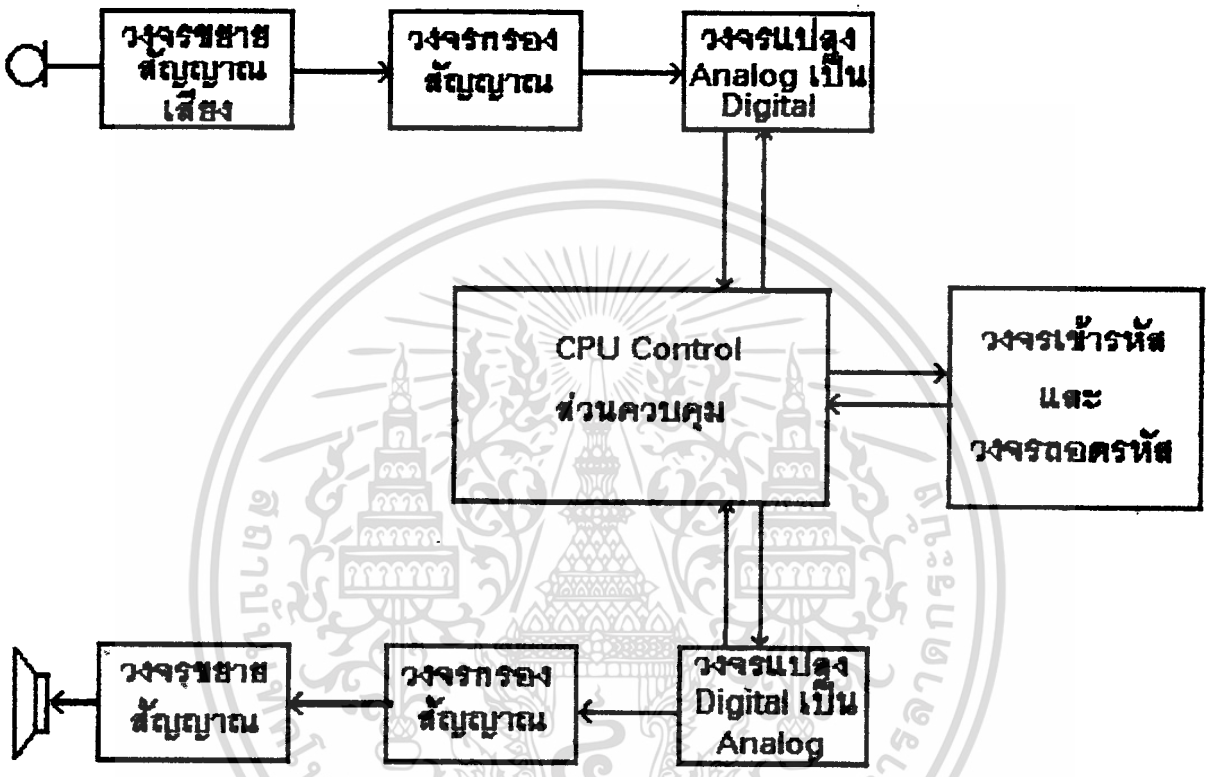


ตารางที่ 1 TIMER 1 GENERATED COMMONLY USED BAUD RATES

BAUD RATE		fOSC	SMOD	TIMER 1		
				C/T	MODE	RELOAD VALUE
MODE 0 MAX	: 1MHZ	12MHz	X	X	X	X
MODE 2 MAX	: 375K	12MHz	1	X	X	X
MODE 1,3	: 62.5K	12MHz	1	0	2	FFH
	19.2K	11.059MHz	1	0	2	FDH
	9.6K	11.059MHz	0	0	2	FDH
	4.8K	11.059MHz	0	0	2	FAH
	2.4K	11.059MHz	0	0	2	F4H
	1.2K	11.059MHz	0	0	2	E8H
	137.5	11.059MHz	0	0	2	1DH
	110	6MHz	0	0	2	72H
	110	12MHz	0	0	1	FEEBH

บทที่ 4
โครงสร้างและส่วนประกอบโครงงาน

โครงงานเครื่องป้องกันการดักฟัง นี้แสดงตาม บล็อกไดอะแกรมข้างล่างนี้



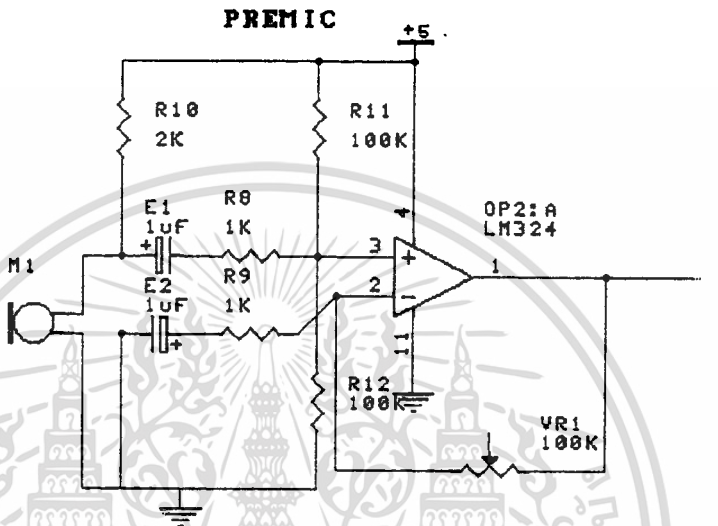
รูปที่ 4.1 แสดงบล็อคไดอะแกรม ของ โครงงานเครื่องป้องกันการดักฟัง

จากบล็อคไดอะแกรมจะเห็นว่ามีส่วนสำคัญอยู่ทั้งหมด คือ

1. PREMIC
2. AMPLIER
3. LOW PASS FILTER
4. ANALOG TO DIGITAL CONVERTER
5. DIGITAL TO ANALOG CONVERTER
6. SCRAMBLE & DESCRAMBLE
7. CPU CONTROL

1. PREMIC

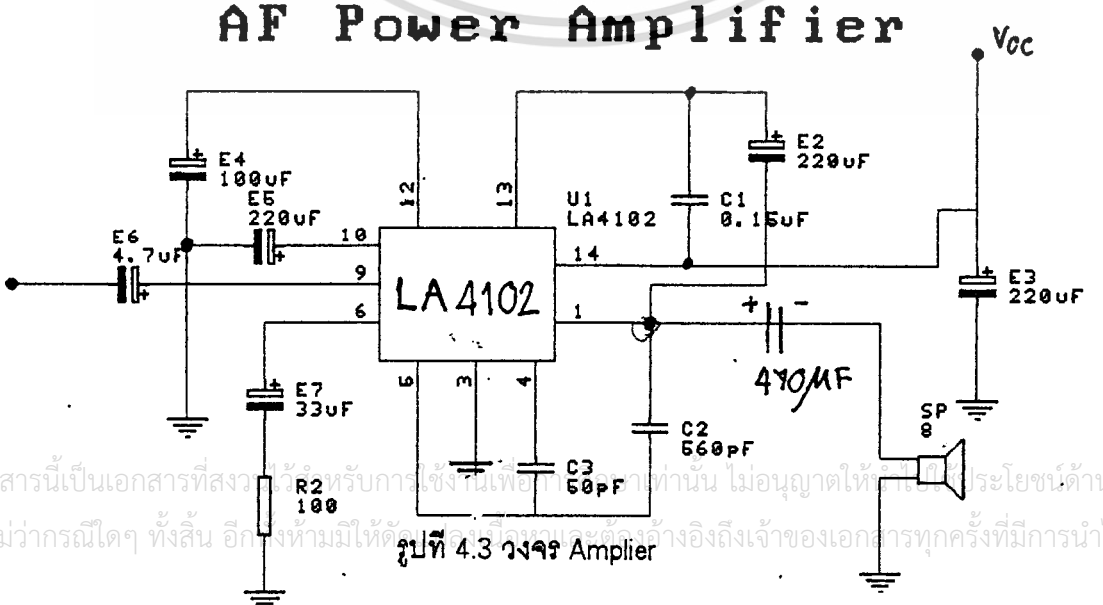
วงจร PREMIC เป็นวงจรขยายระดับสัญญาณเสียงให้มีความแรงมากพอในการที่จะทำให้แปลงเป็นดิจิตอลได้โดยปรับให้สัญญาณสวิงอยู่ในช่วง -2.5 v ถึง 2.5v ดูรูปภาพวงจรเพื่อที่จะนำอินพุท ไปผ่าน Low Pass Filter



รูปที่ 4.2 แสดงวงจร PREMIC

2. AMPLIER

เป็นวงจรที่ทำการขยายสัญญาณ ที่ผ่านมาจาก Low Pass Filter ทำหน้าที่ขยายสัญญานเพื่อที่จะป้อนเข้าสู่ มอนิเตอร์ (ลำโพง)



รูปที่ 4.3 วงจร Amplier

3. LOW PASS FILTER

วงจรกรองสัญญาณเป็นส่วนหนึ่งของเครื่องเข้าระหัสเสียงพูด ซึ่งนับได้ว่ามีความสำคัญใน การขจัดปัญหาสิ่งรบกว่นที่สอดแทรกมากับสัญญาณหรือสัญญาณรบกวนที่เกิดขึ้นจากตัวอุปกรณ์เอง การออกแบบวงจรกรองสัญญาณ จะต้องคำนึงถึงขีดความสามารถในการทอนสัญญาณส่วนที่ไม่ ต้องการให้มีผลต่อการรับฟังของเครื่องเข้าระหัสเสียงพูดชนิดน้อยที่สุด

3.1 การออกแบบวงจรกรองสัญญาณ

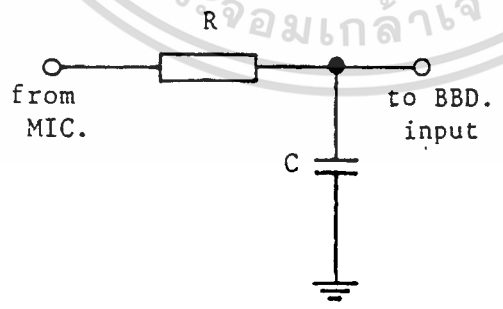
การนำวงจรกรองสัญญาณมาใช้ในเครื่องดังกล่าวมีจุดที่นำมาใช้สำคัญ ๆ อยู่ 2 แห่ง คือ

1.การนำมาใช้ภาคอินพุท ของ ANALOG to DIGITAL CONVERTER

2.การนำใช้ที่ภาคเอ้าพุทของ DIGITAL TO ANALOG CONVERTER

การออกแบบวงจรกรองสัญญาณกระทำได้เฉพาะสัญญาณรบกวนที่อยู่นอกแบนวิดท์ของเสียงเท่านั้น (200-3200 Hz) ในส่วนนี้ประกอบด้วยวงจรกรองสัญญาณความถี่ต่ำผ่าน ที่ไม่ต้องการอัตราการลดทอน ของสัญญาณมากนัก จึงใช้วงจรกรองสัญญาณความถี่ต่ำแบบ RC-เน็ทเวิร์คอันดับหนึ่ง (first- order-RC network) ดังรูป โดยความถี่คัทออฟ (cutoff-freauency) สำหรับวงจรกรอง สัญญาณแบบนี้ กำหนดได้จาก

$$f_c = 1/\omega RC$$

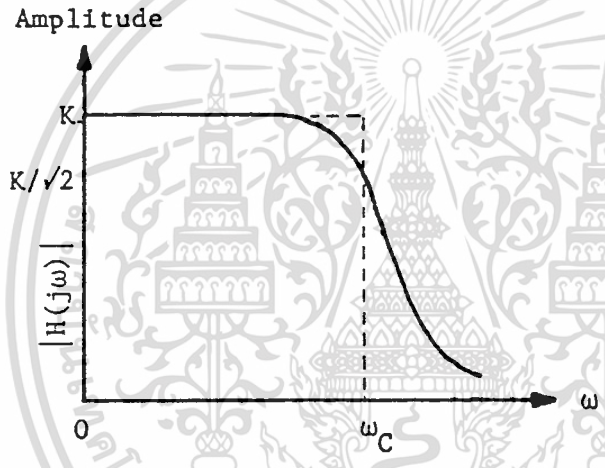


รูปที่ 4.4 RC-lowpass filter

3.2 การออกแบบวงจรกรองสัญญาณในภาคเอาพุท

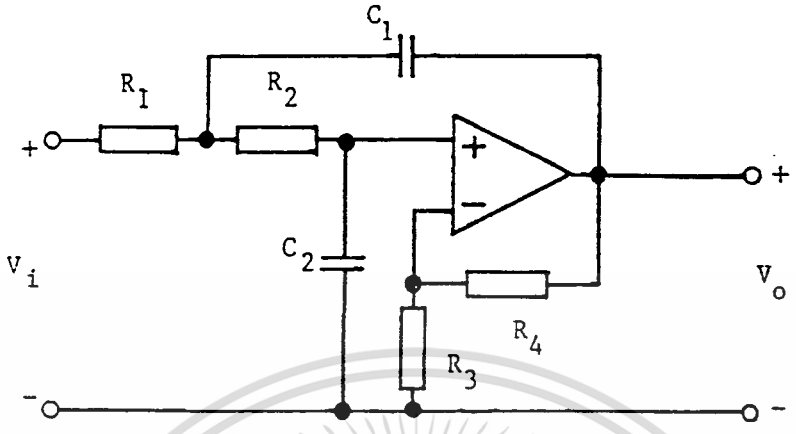
ในส่วนนี้ประกอบด้วยวงจรกรองสัญญาณความถี่ต่ำผ่านที่ความถี่คัทออฟเท่ากับ 3000 Hz. และวงจรกรองสัญญาณความถี่สูงผ่านที่ความถี่คัทออฟเท่ากับ 200 Hz ความจริงแล้วในส่วนนี้ควรจะใช้วงจรกรองสัญญาณแบบย่านความถี่ผ่านแต่ทั้งนี้เนื่องจากต้องการให้อัตราการ ลดทอนของสัญญาณสูงโดยใช้วงจรกรองสัญญาณแบบแอคทีฟอันดับสองที่ใช้ออปแอม และจะกล่าวรายละเอียดดังต่อไปนี้

การออกแบบวงจรกรองชนิดความถี่ต่ำผ่าน



รูปที่ 4.3 การตอบสนองของวงจรกรองชนิดความถี่ต่ำผ่าน

การเลือกออกแบบวงจรกรองชนิดความถี่ต่ำผ่านที่ใช้งานเราใช้แบบ แอคทีปฟิลเตอร์ที่ใช้ออปแอมซึ่งง่ายต่อการออกแบบมากตามรูปเป็นวงจรกรองชนิดความถี่ต่ำผ่านแบบบัตเตอร์เวิทอันดับที่ 2 ซึ่งมีคุณสมบัติตามที่ต้องการ โดยมีขนาดของสัญญาณออกแบบราบเรียบ



รูปที่ 4.5 วงจร Second order low pass Butterworth filter

ลักษณะของทรานส์เฟอร์พันก์ชั่นของวงจรกรองชนิดความถี่ต่ำผ่านที่มีอันดับ 2 จะอยู่ในลักษณะ

$$V_o(s) / V_i(s) = K/(s+as+b)$$

โดยที่ K = ค่าคงที่

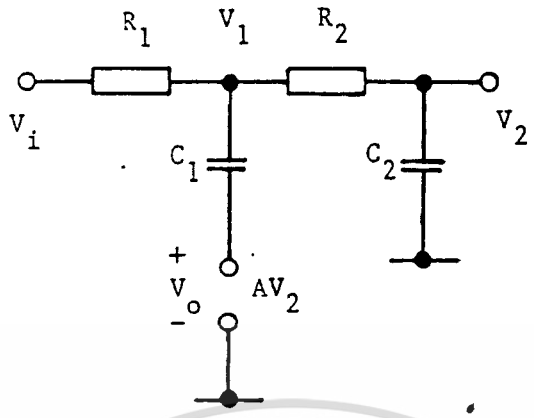
a,b = ค่าคงที่ในการเลือกออกแบบของวงจรกรองความถี่

ถ้าให้ $H(s) = V_o(s) / V_i(s)$

จากสมการ (5.3) การตรวจสอบขนาดของสัญญาณเขียนได้เป็น

$$H(j\omega) = K' \; 1+(w\omega/\omega_c)^{2n}$$

จากรูป 5.3 สามารถเขียนวงจรทัดเทียมได้ตามรูปที่ 5.4



รูปที่ 4.6 แสดงวงจรทัดเทียมของวงจรกรองชนิดความถี่ต่ำผ่านอันดับ 2

จากรูป 4.6และสมการ สามารถกำหนดค่าต่าง ๆ ได้ดังนี้

$$K = A/R_1 R_2 C_1 C_2$$

$$a = (1-G)/R_2 C_2 + 1/R_1 C_1 + 1/R_2 C_2)$$

$$b = 1/R_1 R_2 C_1 C_2$$

โดยที่ $A = (R_3 + R_4)/R_3$

G = อัตราขยายวงจร

ในการออกแบบวงจรกรองชนิดความถี่ต่ำผ่านที่ใช้งาน ต้องการอัตราการลดทอนสัญญาณสูง จึงได้ออกแบบเป็นวงจรกรองความถี่ต่ำผ่านอันดับ 6 โดยใช้วงจรกรองที่มีอันดับ 2 สามวงจรมาต่ออนุกรมกัน การเลือกใช้ค่า R และ C (ค่าต่าง ๆ ในวงจรนั้นทำการออกแบบโดยใช้ตาราง กราฟที่มีอยู่แล้ว (design by inspection of graph) ในภาคผนวก

การออกแบบวงจรกรองทั้งชนิดความถี่ต่ำและความถี่สูงผ่านในทางปฏิบัติ

การออกแบบในทางทฤษฎี จะเห็นว่าไม่มีปัญหา สามารถคำนวณค่าออกมาได้ แต่เมื่อนำมาต่อเป็นวงจรแล้ว จะมีปัญหาเกิดขึ้น ซึ่งพอจะสรุปได้ ดังนี้

1. เกิดการโหลดกับสัญญาณขาเข้า เนื่องจากอินพุทอิมพิแดนซ์ของวงจร โดยเฉพาะ การเลือกค่า R_1 , C_2 ในรูปที่ 4.5 และ R_1 , C ในรูปที่ 5.6 จะเป็นตัวกำหนดอิมพิแดนซ์ของวงจร

2. เกิดการออสซิลเลทขึ้น เมื่อพิจารณาจากรูป 4.5 แล้ว จะเห็นได้ ว่าวงจรที่ใช้จะมีส่วนที่เป็นสัญญาณป้อนกลับแบบเพิ่มกำลัง ซึ่งเป็นตัวกำหนดความถี่คัทออฟ และ ส่วนที่เป็นสัญญาณป้อนกลับแบบลดกำลังจะเป็นตัวกำหนดอัตราการขยายวงจร ดังนั้นในการออก แบบจะต้องคำนึงถึงการป้อนกลับของสัญญาณ (feed back) ทั้งสองให้มีค่าใกล้เคียงกัน ถ้าส่วนใดส่วนหนึ่งมีการป้อนกลับมากเกินไป จะทำให้เกิดการออสซิลเลทขึ้นในวงจรได้

3. เกิดสิ่งรบกวนจากภายนอกเข้ามา จากรูปที่ 3.5 และ R_3 และ R_4 ทำหน้าที่เป็นตัวกำหนดอัตราการขยายของวงจร โดยมี R_4 เป็นตัวป้อนกลับ ค่า R_4 ที่เลือกใช้ไม่ควรมีค่าเกิน 1 เมกะโอห์ม เพราะจะทำให้มีสิ่งรบกวนจากภายนอกเข้ามาได้

4. เกิดความผิดพลาดจากภาคทฤษฎีมาก เช่น อัตราการขยายของวงจรความถี่คัทออฟ ซึ่งสาเหตุนี้เกิดขึ้นเมื่อเลือกออปแอมมาใช้ไม่เหมาะสมกับสิ่งที่ต้องการ ซึ่งจะต้องคำนึงถึงคุณสมบัติดังต่อไปนี้

- มีอินพุทอิมพิแดนซ์สูงพอ มิฉะนั้นจะทำให้การออกแบบผิดพลาด เนื่องจากทางทฤษฎีเรากำหนดว่าอินพุทอิมพิแดนซ์ของออปแอมเท่ากับ
- ผลคูณระหว่างค่าขยายและแบนวิด
- มีผลตอบสนองต่อความถี่ที่ดีมาก

จากคุณสมบัติข้างบน ในการเลือกออปแอมมาใช้จะทำให้การออกแบบได้ตามวัตถุประสงค์ คือ

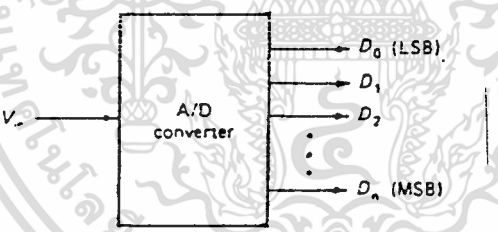
- ค่าอินพุทอิมพิแดนซ์ของวงจร
- อัตราการขยายของวงจร
- อัตราการลดทอนสัญญาณ
- จำนวนอันดับของวงจรกรอง

ในการออกแบบวงจรกรองสัญญาณขึ้นใช้งานนั้น จากการทดลองมีความคลาดเคลื่อนของความถี่คัทออฟเล็กน้อย ซึ่งปัญหาเกิดจากการเลือกค่า R และ C ไม่ได้ถูกต้องกับความต้องการ แต่ความผิดพลาดที่เกิดขึ้นนี้ มีผลน้อยมากกับการออกแบบเครื่องซแครมเบอร์ เนื่องจากคัทออฟที่คลาดเคลื่อนเพียงเล็กน้อยไม่มีผลต่อการรับฟังมากนัก สิ่งที่ควรพิจารณา ที่จะช่วยให้เครื่อง ดังกล่าวทำงานได้ดี คือ การเพิ่มอัตราการลดทอนสัญญาณให้สูงขึ้น แต่ทั้งนี้การแก้ปัญหาเนื่องจากสัญญาณรบกวนด้วยการกรองสัญญาณจะกระทำได้ก็เฉพาะนอกแบนวิดเสียง (speed bandwitdth) เท่านั้น

อนึ่งการซแครมบลิ่งสัญญาณไม่ว่าจะเป็นวิธีใดก็ตาม ถ้ายิ่งเพิ่มประสิทธิ์ภาพของการซแครมบลิ่งมากขึ้นเท่าใด อัตราของสัญญาณต่อสัญญาณที่ลดลงเท่านั้น

4. ANALOG TO DIGITAL CONVERTER

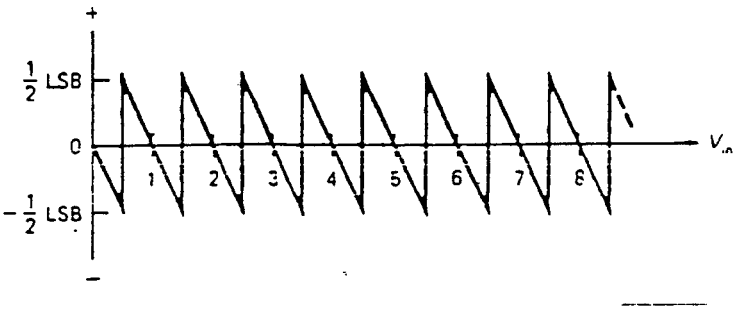
ปกติแล้ว การทำงานของซีพียู หรือไมโครโปรเซสเซอร์ มักจะมีการติดต่อกับอุปกรณ์ทางอนาลอก จึงจำเป็นที่จะต้องรับสัญญาณ หรือข้อมูลเข้ามา เพื่อที่จะประมวลผลต่อไป เช่นกันกับโครงงานนี้ADC จะนำมาใช้ในการแปลงสัญญาณเสียงที่เข้ามาให้เป็นสัญญาณดิจิตอล เพื่อที่จะนำไป ส่งเข้าวงจร SCRAMBLE ในการเข้ารหัส



รูปที่4.7 สัญลักษณ์ของ A/D CONVERTER

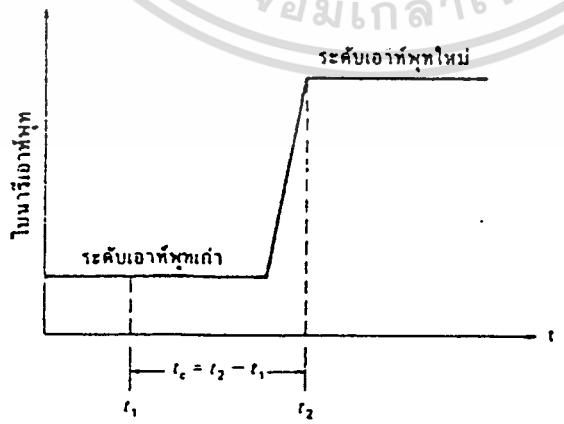
4.1 หลักการทำงานของ A/D

กุญแจสำคัญที่เราใช้ในการพิจารณาคุณสมบัติของการเปลี่ยนแปลง สัญญาณอนาลอกเป็นดิจิตอลนั้นสามารถสังเกตได้จากรูปที่ ในกรณีของการแปลงสัญญาณจากดิจิตอลเป็นอนาลอกนั้นตัวที่กำหนดความถูกต้องก็คือจำนวนบิตในกรณีองการแปลงสัญญาณจากอนาลอกเป็นดิจิตอลก็เช่น เดียวกัน จากรูป จะเห็นว่า จำนวนขั้นบันได (stair - step) ทั้งหมดมี 16 ขั้น เอาท์พุทของ A/D คอนเวอร์เตอร์นั้น จะถูกประมาณว่าเป็นสัญญาณอินพุทแบบดิจิตอล กราฟแสดงข้อผิดพลาดใน เอาท์พุทที่จุดต่าง ๆ ดูได้จากรูปที่ 4.8



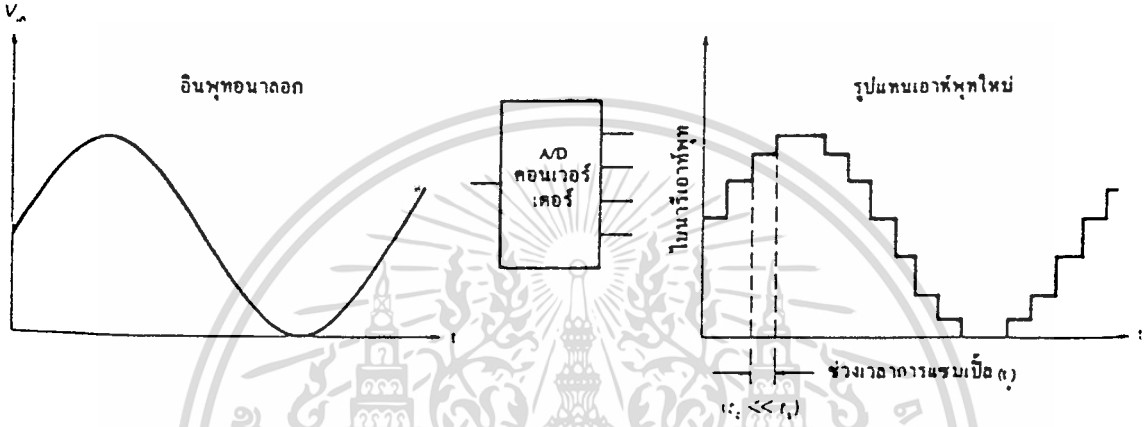
รูปที่4.8 ข้อผิดพลาดของ A/D CONVERTER

ในกรณีของตัวแปลงที่มีเรสโซลูชั่นสูง ๆ (อินพุทบิทมาก) ความกว้างสูงสุดของข้อผิดพลาดจะลดลงตามบททฤษฎีแล้ว ถ้าเป็น A/D คอนเวอร์เตอร์ในอุดมคติจะต้องมีเอาท์พุทมากจนนับไม่ได้ (จนถึงα) ดังนั้นจะมีค่าเรสโซลูชั่นจนถึง α จะทำให้กราฟรูปที่ แกนตั้งเป็น 0 แต่ไม่มีทางเกิดขึ้นได้ในทางปฏิบัติ ข้อผิดพลาดที่เกิดจากมี เรสโซลูชั่น ที่จำกัด เราเรียกว่า ข้อผิดพลาดควอนไตร์ (quantizing error) ข้อผิดพลาดชนิดนี้ ไม่สามารถจำกัดได้เอาท์พุทของ A/D คอนเวอร์เตอร์ ก็คือ ระดับอินพุทซึ่งจะคงที่ในเวลาหนึ่ง สิ่งนี้ชี้ให้เห็น ว่า A/D คอนเวอร์เตอร์ทำงานโดยการแซมปลิ้งปริมาณของสัญญาณอนาล็อกและต้องแน่ใจสัญญาณจะคงที่ช่วงเวลานั้น เราจึงต้องมีวงจรสำหรับค้างค่า (hold) ของสัญญาณ จึงทำให้เรานิยมใช้วงจรแซมเปิ้ล และโฮลด์ (sample and hold) เพื่อแน่ใจว่าปริมาณ ของอินพุทจะไปเปลี่ยน ขณะที่กำลังทำการแปลง สัญญาณเวลาในการแปลงสัญญาณ และอัตราการแซมเปิ้ล (sample) เป็นปัจจัยในการพิจารณาอย่างมาก เวลาในการแปลงสัญญาณ (conversion time) t_c คือเวลาที่ช้าไประหว่างที่อินพุทเข้ามาจนถึงการ แสดงค่าของไบนารีเอาท์พุทในกรณีที่เอาท์พุทจะเริ่มต้น เปลี่ยนจาก 0 ไปถึงค่าที่มากที่สุด ในรูปที่ 4.9 เป็นตัวอย่างของเวลาหน่วง (time delay)



รูปที่ 4.9 แสดงการตอบสนองของเวลาแปลงสัญญาณ (conversion time) ของ A/D converter

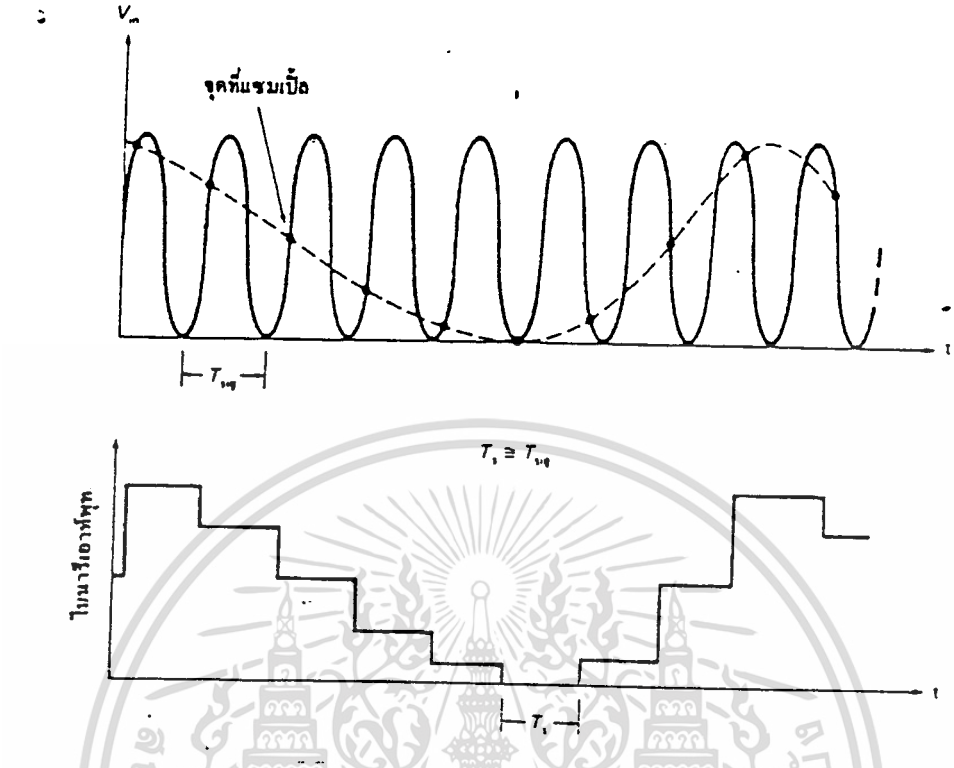
อินพุทที่เข้าในคอนเวอร์เตอร์จะอยู่ ณ เวลา t1 และสัญญาณตอบสอนง ณ t2 ผลต่างของบวกทั้งสองคือเวลาแปลงสัญญาณ t เวลาแปลงสัญญาณเป็นอัตราที่มากที่สุดซึ่งสัญญาณถูกแซมปลิ้งเวลาแปลงสัญญาณเวลาแปลงสัญญาณเป็นอัตราที่มากที่สุดซึ่งสัญญาณถูกแซมเปิ้ล ช่วงเวลาของการแซมเปิ้ลเรียกว่าเวลาแซมเปิ้ล(Sampletime) อัตราแซมเปิ้ลใช้ช่วงในการบอกเวลาแซมเปิ้ล เพื่อให้ทราบถึงผลการแซมเปิ้ลบนสัญญาณอินพุทอนาลอกไปเป็นปริมาณ ของดิจิตอล พิจารณาจากรูปกราฟไซน์ ดังรูป 4.10



รูปที่4.10 ขบวนการแปลงสัญญาณดิจิตอลด้วยการ A/D converter

ถ้าเราให้เวลาแปลงสัญญาณ (t) น้อยมาก ๆ จนตัดทิ้งได้เวลาในการแซมเปิ้ล 1/10 ของสัญญาณอินพุทจะได้กราฟเป็นรูปลักษณะแบบรูปถ้าเราเพิ่มอัตราแซมเปิ้ลและเพิ่มเรสโซลูชั่น (จำนวนเอาท์พุท)ให้มากขึ้น ก็จะได้ว่าเอาท์พุทที่ใกล้เคียงกับสัญญาณอนาลอกจากอินพุทมากขึ้น

ปัญหาอีกอย่างหนึ่ง คือ ถ้าอินพุทเปลี่ยนแปลงระดับอย่างรวดเร็วเมื่อเทียบกับอัตราแซมเปิ้ล A/D คอนเวอร์เตอร์ไม่สามารถเปลี่ยนแปลงสัญญาณได้ถูกต้องและจะเกิดการเพี้ยนของสัญญาณ ปัญหาเช่นนี้สามารถแสดงให้เห็นจากระบบเวลาแซมเปิ้ล (time sample system) เช่น A/D คอนเวอร์เตอร์อัตราความถี่ของการแซมเปิ้ลต้องอย่างน้อย 2 ครั้ง ต่อหนึ่งลูก คลื่นของสัญญาณอินพุท การกำหนดความถี่ในการแซมเปิ้ลแบบนี้ ก็คือ ทฤษฎีในควิสต์แซมเปิ้ล (myguist sampling theorem) รูป แสดงผลของการไม่ทำตามกฎของในควิสต์



รูปที่ 4.11การเพี้ยนเกิดขึ้นเมื่ออัตราแซมเปิ้ลต่ำเกินไปเมื่อเทียบกับคาบเวลาของสัญญาณอินพุท

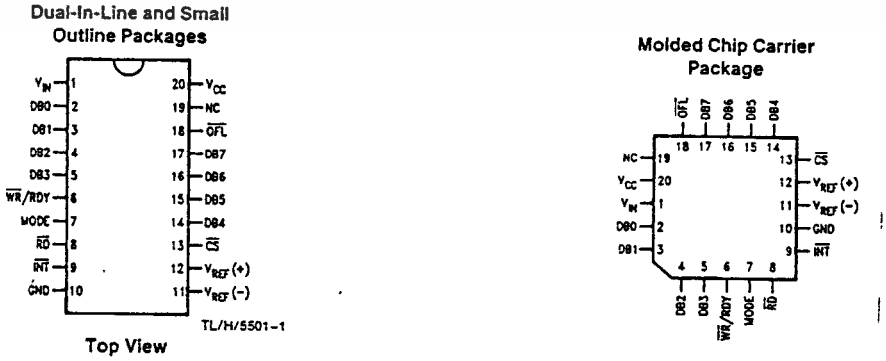
เราจะได้เอาท์พุทของ A/D คอนเวอร์เตอร์เป็นรูปเพี้ยน (alias) ในการใช้งานที่มีการเปลี่ยนแปลงสัญญาณอินพุทเร็ว ๆ นี้ เราควรที่จะใช้การแซมเปิ้ลแบบความเร็วสูง เพื่อจะให้ได้ความถูกต้องมากในปัจจุบัน การควบคุมระบบ (เช่น การนำวิถี) จะใช้การแซมเปิ้ลที่มีอัตราสูง อัตราการแซมเปิ้ลที่มากสุดถูกจำกัด โดยความเร็วในการแปลงสัญญาณของ A/D คอนเวอร์เตอร์ เช่น ถ้าแซมเปิ้ลทุก ๆ 10 ns ก็จะไม่ดี เมื่อความเร็วในการแปลงสัญญาณเป็น 500 ns จะทำให้เอาท์พุทออกมาแบบนำไปใช้งานไม่ได้

อีกจุดหนึ่งที่สำคัญ ก็คือ ข้อผิดพลาดที่เกิดขึ้นของ A/D คอนเวอร์เตอร์จะเกิดเป็น ข้อผิดพลาดทางออฟเซต (offset) เกน (gain) และความไม่เป็นเส้นตรง (nonliniarity) ซึ่งมีผลต่อความเที่ยงตรงของการแปลงสัญญาณข้อผิดพลาดพวกนี้ จากวงจรที่นำมาสร้างเป็น A/D คอนเวอร์เตอร์การแปลงสัญญาณอนาลอกเป็นสัญญาณดิจิตอลมีหลายวิธีด้วยกันในโครงงานนี้ใช้วิธีการแปลงแบบขนาน หรือ แบบ FLASH

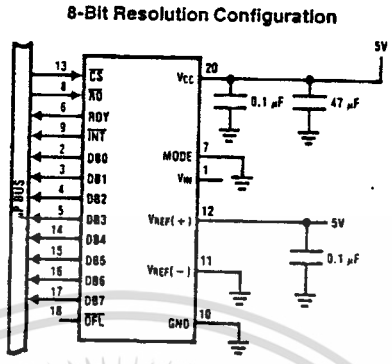
A/D converter แบบขนาน ถ้าเราคำนึงด้านความเร็วในการแปลงสัญญาณเป็นอันดับแรก เราควรจะใช้วิธีการ A/D converter แบบขนาน คอนเวอร์เตอร์แบบนี้ ไม่ใช่การเรียงลำดับของเอาท์พุทพร้อม ๆ กันเข้าไปในกลุ่มของคอมพาราเตอร์ที่มีต่อแบบขนาน ซึ่งแต่ละตัวก็จะทำหน้าที่ของมันในรูปที่ เป็น A/D converter แบบขนาน 3 บิท แรงดันอ้างอิงจะถูกต่อกับตัวต้านทานแบบอนุกรม ซึ่งจะเกิดแรงดันแบ่ง

ข้อเสียประการหนึ่ง คือ เอาพุทที่ได้ไม่เป็นเลขฐานสอง ต้องมีวงจรเพิ่มเติมไปทำการเข้ารหัสข้อดีของวงจรเอทูดีแบบขนานนี้ คือ ความเร็วสูงมาก บางครั้งจึงเรียกวงจรเอทูดี แบบนี้ว่า แบบ "แฟลช" (Flash type A/D converter) วงจรเอทูดีชนิดนี้ ใช้เวลาใน การแปลงได้เร็วในระดับนาโนวินาทีทีเดียว

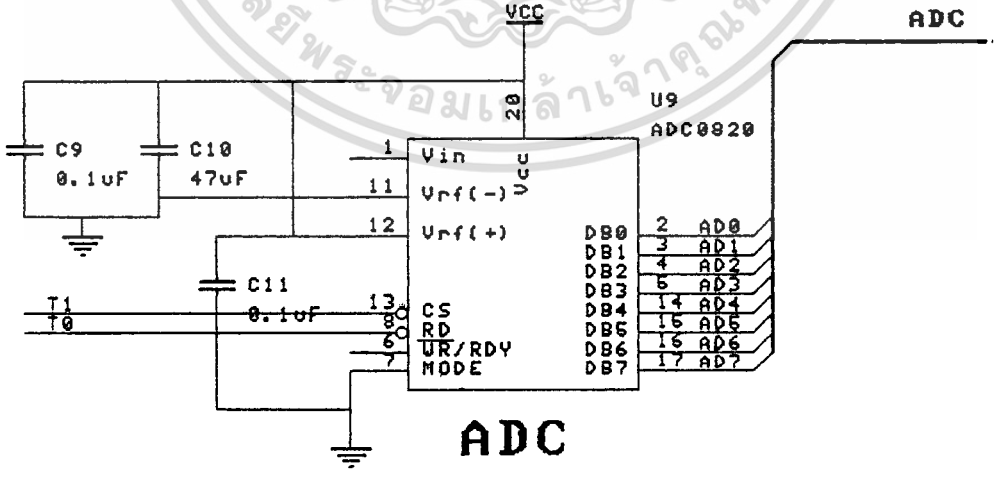
ในการใช้งานเอทูดีคอนเวอร์เตอร์โดยปกติ เราจะใช้ A/D มากกว่า การต่อเป็นวงจรในโครงงานนี้ ใช้ IC A/D เบอร์ ADC0820 ของบริษัทเนชั่นแนลเซมิคอนดักเตอร์ ดัง รูปที่ 4.13 เป็นไอซีที่มีการแปลงเร็วมาก ประมาณ 2.5 ns ในโหมด RD และ 1.5 ns ในโหมด WR-RD โดยใช้เทคนิคการแปลงแบบ half-flash ซึ่งเทคนิคอันนี้ประกอบไปด้วยตัวเปรียบเทียบทั้งหมด 32 ตัว 2 ชุด คือ ชุดละ 4 บิท 2 ชุด คือชุดนัยสำคัญสูง (MSB) และชุดนัยสำคัญต่ำ (MSL) สัญญาณอินพุทที่จะนำเข้ามาแปลงใน ADC0820 จำเป็นที่จะต้องมีวงจรแซมปลิ้งอนุกรมภายนอก หากว่าสัญญาณที่เข้ามามีค่าสัญญาณน้อยกว่า 100 mV/s และ ADC0820 สามารถที่จะอินเทอร์เฟสกับไมโครโปรเซสเซอร์ได้โดยง่าย ซึ่ง ADC0820 ถูกออกแบบให้ตัวชีพียูมองเหมือนเป็นตำแหน่งของเมมโมรี่ หรือมองเป็น I/O PORT โดยปราศจากความจำเป็นที่จะ ต้องทำการอินเทอร์เฟสภายนอก ดังแสดงในรูปที่ 4.14



รูปที่4.13 แสดงการจัดขาของไอซี ADC0820



รูปที่ 4.14 แสดงลักษณะการอินเทอร์เฟสกับไมโครโปรเซสเซอร์



รูปที่ 4.15 เป็นการประยุกต์เพื่อใช้กับไมโครโปรเซสเซอร์ 8031

ลักษณะของขาของ ADC0820

ขาที่ 1 Vin = เป็นขาของสัญญาณอินพุทที่เป็นสัญญาณอนาลอก โดยแรงดันที่ขานี้จะต้องมีค่าอยู่ระหว่าง GND และ Vcc

ขาที่ 2 DB 0 = Tristate data o/p บิทที่ 1

ขาที่ 3 DB 1 = Tristate data o/p บิทที่ 2

ขาที่ 4 DB 2 = Tristate data o/p บิทที่ 3

ขาที่ 5 DB 3 = Tristate data o/p บิทที่ 4

ขาที่ 6 WR/RDY มีสองโหมด คือ WR-RD และ RD

- ในโหมด WR เมื่อขา CS (ขาที่ 13) มีค่าเป็น low การแปลงก็จะเริ่มขึ้นที่ขอบขาลงของ WR โดยใช้เวลาประมาณ 800 ns นับจากขอบขาขึ้นของ WR ผลของการแปลงจะถูกอ่านเข้าไปยังที่ o/p latch ซึ่งสัญญาณ RD จะไม่สามารถปรากฏได้ในช่วงนี้ สังเกตรูปที่ 4.16 (a) และ 4.16 (b)

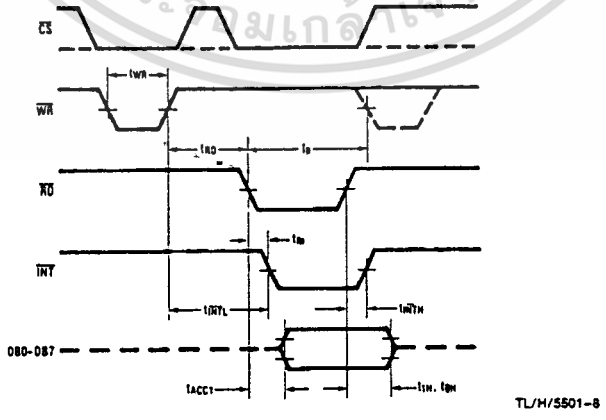
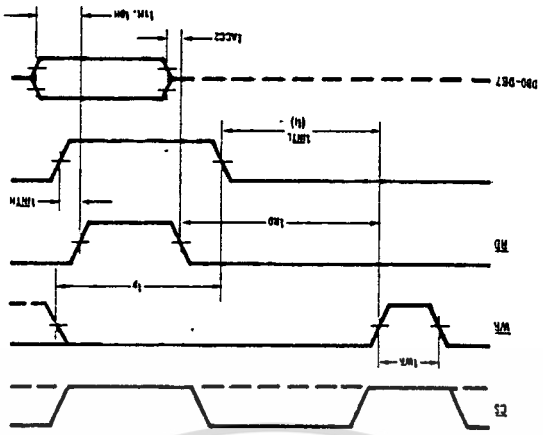


FIGURE 3a. WR-RD Mode (Pin 7 is High and $t_{RD} < t_i$)

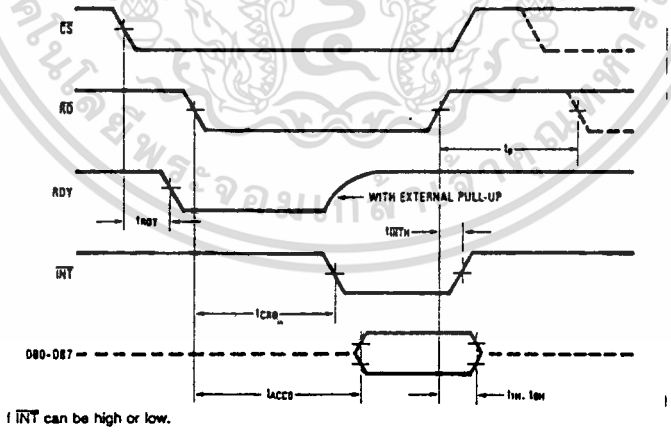
รูปที่ 4.16 (a) WR-RD MODE (ขา 7 มีค่าลอจิก high และ t RD < t i)



รูปที่ 4.16 (b) WR-RD MODE (ขา 7 มีค่าลอจิก high และ t RD > t i)

- ในโหมด RD นี้เป็นแบบ open drain output

(ไม่มีอุปกรณ์ pull-up ภายใน) RDY จะมีค่า low หลังจากขอบขาลงของการแปลง ได้ถูกอ่านเอาไปเป็น o/p latch แล้ว ดูรูปที่ 4.17



รูปที่ 4.17 RD MODE (ขา 7 มี logic = low)

ขาที่ 7 MODE เป็นขาเลือกโหมด input ภายใน

RD MODE เมื่อขา 7 นี้ มีค่า logic = low

WR-RD MODE เมื่อขา 7 นี้ มีค่า logic = high

ขาที่ 8 RD

- ที่ MODE WR-RD

เมื่อ CS low o/p (DB 0 -DB 7) จะ Active เมื่อ RD = low (ดูรูปที่ 5.32) RD สามารถจะเพิ่มความเร็วในการแปลงได้โดยการอ่านลำดับ data ไปแสดง ถ้าทำเช่นนี้ ผลของ data ที่ส่งไปยัง o/p latch จะถูก latch ไว้ หลังจากขอบขาลงของ RD ดูรูป 4.16 (a) และ 4.16 (b)

- ที่ RD-MODE

เมื่อ CS = low การแปลงจะ start ด้วยการทำให้ขา RD = low ด้วย RD จะ Enable ให้ tristate data output มีการแปลงอย่างสมบูรณ์ และ INT จะเท่ากับ low เมื่อการแปลงเสร็จสมบูรณ์

ขาที่ 9 INT

ที่ WR-RD โหมด

- INT = low เพื่อบอกว่าการแปลงเสร็จแล้ว และผลของข้อมูลก็จะไปอยู่ที่ o/p latch ขา INT จะมีค่าเป็น low โดยใช้เวลาประมาณ 800 ns หลังจากขอบขาลง WR

ที่ RD โหมด

- INT จะ low เป็นตัวชี้ว่า การแปลงได้เสร็จสิ้นแล้ว และให้ data มาไว้ที่ o/p latch

ขาที่ 10 GND

ขาที่ 11 V REF (-) มีค่าระหว่าง GND และ V REF (+)

ขาที่ 12 V REF (+) มีค่าระหว่า V REF (-) และ Vcc

ขาที่ 13 CS CS จะแสดงค่า low เมื่อที่จะเริ่มทำการแปลง

ขาที่ 14 DB 4 Tristate data o/p บิทที่ 4

ขาที่ 15 DB 5 Tristate data o/p บิทที่ 5

ขาที่ 16 DB 6 Tristate data o/p บิทที่ 6

ขาที่ 17 DB 7 Tristate data o/p บิทที่ 7

ขาที่ 18 OFL ขาที่เกิดจากการ overflow ของข้อมูล เมื่ออินพุทมีค่ามากกว่า V REF (+)

ขาที่ 19 NC ไม่มีการเชื่อมต่อ

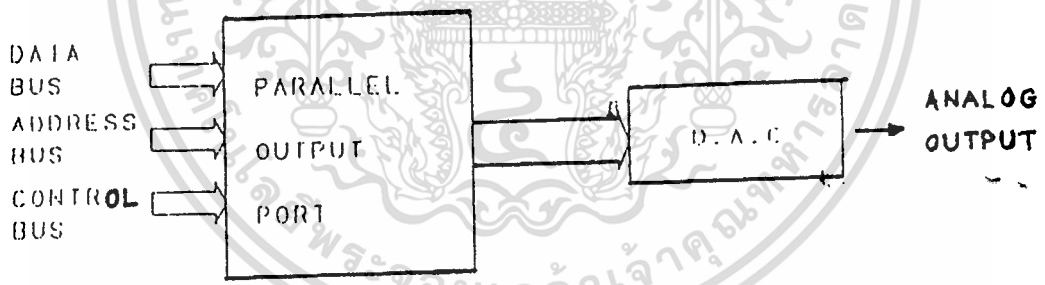
ขาที่ 20 Vcc ไฟเลี้ยง

5 วงจร D/A (DIGITAL-TO-ANALOG-CONVERTER)

D/Aคอนเวอร์เตอร์หรือเรียกย่อๆว่าตัว DACเป็นตัวแปลงรหัสเลขฐานสองจากคอมพิวเตอร์หรือจากวงจรดิจิตอลใดๆ ให้กลายเป็นระดับแรงดันอนาล็อกที่มีความสัมพันธ์กับระบบเลขฐานสอง

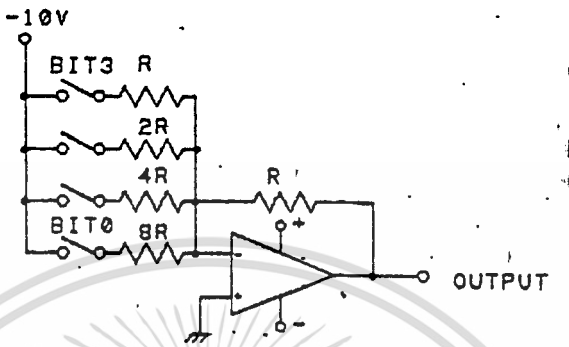
สัญญาณในระบบไฟฟ้ามีสัญญาณพื้นฐานอยู่ 2 แบบ คือ สัญญาณดิจิตอลและสัญญาณอนาล็อก สัญญาณอนาล็อก หมายถึง สัญญาณที่มีค่าการเปลี่ยนแปลงต่อเนื่อง แต่สัญญาณดิจิตอลเป็นสัญญาณที่มีระดับการเปลี่ยนแปลงของสัญญาณ 2 ระดับเท่านั้น ซึ่งสัญญาณสองระดับนีเรากำหนดให้เป็น 0 และ 1 ในลักษณะของสัญญาณไบนารี่ ซึ่งมีการใช้งานในระบบของ ไมโครโปรเซสเซอร์ ดังนั้น หากเราต้องการนำไมโครโปรเซสเซอร์ไปต่อกัน อุปกรณ์ที่ใช้สัญญาณอนาล็อก เราจำเป็นต้องมีการเปลี่ยนสัยญาณจากดิจิตอลเป็นสัญญาณอนาล็อก (Digital to Analog Conversion DAC) และการเปลี่ยนสัญญาณจากอนาล็อกเป็นสัญญาณดิจิตอล (Analog to Digital Converion ADC) เพื่อให้ไมโครโปรเซสเซอร์สามารถต่อกับอุปกรณ์ที่รับ และส่งข้อมูลแบบอนาล็อกได้

การส่งข้อมูลของสัญญาณดิจิตอลจากไมโครโปรเซสเซอร์ไปให้กับอุปกรณ์ทำหน้าที่เปลี่ยนสัญญาณดิจิตอลนั้น เราใช้การส่งข้อมูลออกไปที่พอร์ตเอาท์พุทแบบขนานดังแสดงในรูป 4.18



รูป 4.18 ไดอะแกรมของการต่อซีพียูกับ DAC

หลักการเบื้องต้นของการเปลี่ยนแปลงสัญญาณดิจิตอลเป็นสัญญาณอนาล็อกนั้นเราอาศัยหลักการง่าย ๆ คือ ระดับของแรงดันอนาล็อกที่เกิดขึ้น จะต้องมีค่าเท่ากับตำแหน่งของเลขฐานสองที่คิดตามความสำคัญของบิท เช่น บิท 0 จะมีค่าของแรงดันเท่ากับ 2



รูป 4.19 แสดงวงจร DAC เบื้องต้น

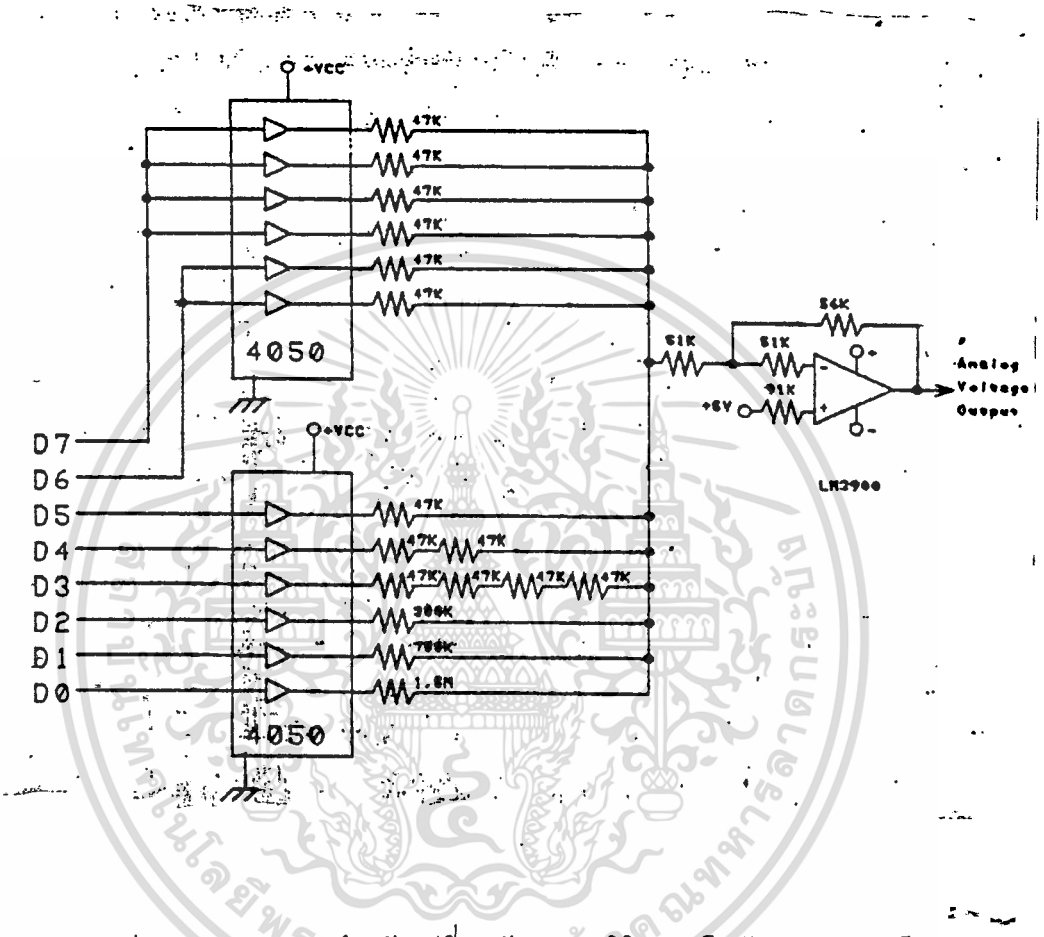
จากรูป 2 วงจรประกอบด้วย สวิตช์ 4 ตัว และความต้านทาน 4 ตัว ต่อเป็นอินพุทของวงจรรวมสัญญาณโดยมีความต้านทาน 1 ตัวต่อเป็นวงจรป้อนกลับ ค่าของความต้านทานที่ต่ออยู่กับตำแหน่งของบิทต่าง ๆ จะมีค่าเป็น 1,2,3,4,8 ตามลำดับ ซึ่งจะทำให้เกณฑ์การขยายมีค่าเป็น -1/8 ,-1/4 ,-1/2 และ -1 ตามลำดับ เราสามารถทดสอบการทำงานของวงจรได้โดยการทำให้สวิตช์ทุกตัวเปิดหมด จะได้เอาท์พุทออกมามีค่าเท่ากับ 0 โวลต์ หากเราเปิดสวิตช์ 0 ก็จะได้แรงดันแรงดันเอาท์พุทออกมาเท่ากับ 1.25 โวลต์ เมื่อเราปิดสวิตช์ 1 ก็ทำให้แรงดันออกมาเท่ากับ 2.5 โวลต์ หากเราปิดสวิตช์ 1 และ 0 พร้อมกันก็จะทำให้ได้แรงดันออกมาเท่ากับ 1.25

สำหรับวงจรการใช้งานสำหรับการเปลี่ยนสัญญาณดิจิตอลขนาด 8 บิทเป็นสัญญาณอนาล็อกแสดงได้ดังรูป 3 โดยมีความต้านทานค่าต่าง ๆ ต่อกับพอร์ตเอาท์พุทผ่านบัฟเฟอร์โดยค่าความต้านทานที่บิท 7 มีค่าเท่ากับ 11.75 กิโลโอห์ม ซึ่งได้จากการต่อค่าความต้านทาน 47 กิโลโอห์มขนานกัน 4 ตัว ที่บิท 6 มีความต้านทาน 47 กิโลโอห์มขนานกัน 2 ตัวซึ่งจะเป็นความแตกต่างของแต่ละบิท จากวงจรจะได้แรงดันเอาท์พุทมีค่าดังนี้

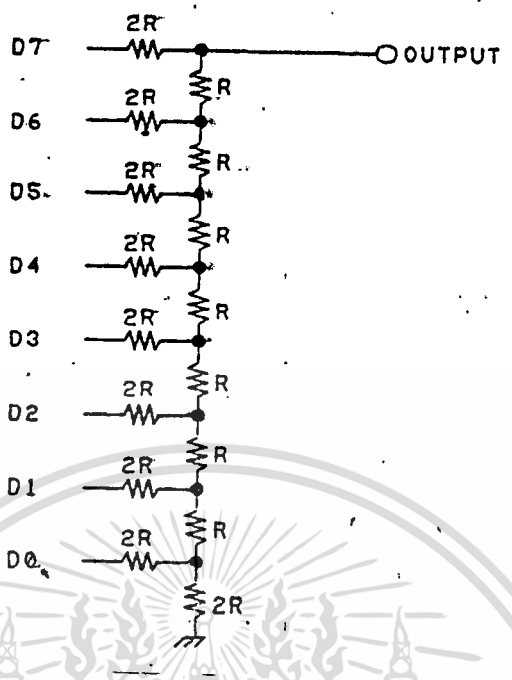
$$V_{out} = 5(n/255) \text{ โวลต์}$$

n เป็นค่าของเลขฐานสิบ ที่ได้จากเลขฐานสอง 8 บิท มีค่าตั้งแต่ 0-255 ดังนั้นค่าของข้อมูล 10001011 (139 ฐานสิบ) เมื่อส่งไปให้เอาท์พุทเท่ากับ 2.7 โวลต์ ในลักษณะนี้ แรงดันเอาท์พุทจะ ถูกแบ่งออกเป็น 255 ระดับ

วงจรสำหรับเปลี่ยนสัญญาณดิจิตอลเป็นสัญญาณอนาล็อกอีกแบบหนึ่ง คือ ใช้วงจร LADDER ซึ่งใช้ค่าความต้านทานเพียง 2 ตัว โดยค่าความต้านทานสองค่านี้จะมีค่าแตกต่างกันเท่าตัว การนำมาต่อกัน จะต่อในลักษณะของ R-2R LADDER ดังแสดงอยู่ในรูป 4 ซึ่งการไหลของกระแสจากข้อมูลที่บิทต่าง ๆ มาที่เอาท์พุทจะมีความค่าตามความสำคัญของบิท

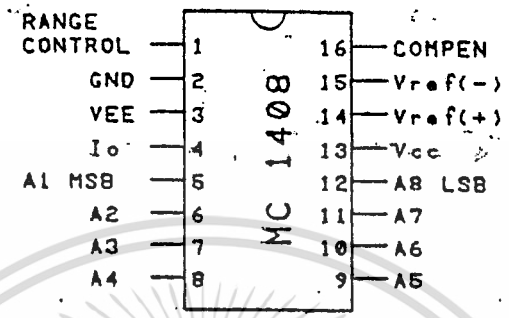


รูป 4.20 แสดงวงจรสำหรับเปลี่ยนสัญญาณดิจิตอลเป็นสัญญาณอนาล็อก



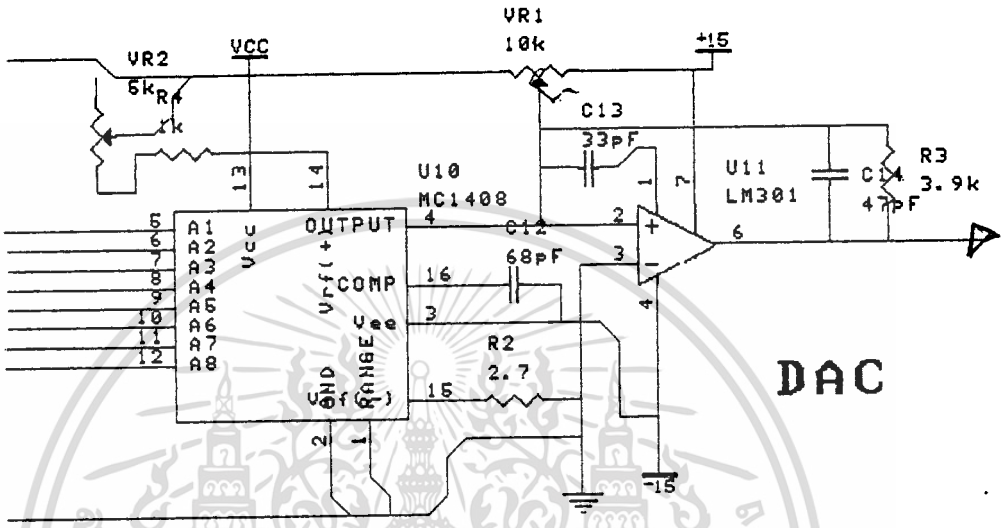
รูป 4.21 วงจร DAC แบบ R-2R LADDER

การใช้งานจริงนั้น วงจรสำหรับเปลี่ยนสัญญาณดิจิตอลเป็นสัญญาณอนาล็อก จะถูกรวมอยู่ในอุปกรณ์ตัวเดียว เช่น เบอร์ MC 1408 ซึ่งเป็นผลิตภัณฑ์ของโมโตโรล่า ใช้สำหรับเปลี่ยนสัญญาณดิจิตอลขนาด 8 บิท เป็นสัญญาณอนาล็อก มีสัญญาณต่าง ๆ แสดงอยู่ในรูป 4.22 และวงจรการใช้งาน แสดงอยู่ในรูปที่ 4.23 ซึ่งเราสามารถปรับค่า Refferent Voltage ให้เปลี่ยนแปลงไปได้ MC 1408 จะให้ เอ้าพุทอยู่ในช่วงประมาณ 0 - 2 mA โดยจะมี Opamp เบอร์ LM 301 ทำหน้าที่เปลี่ยนแปลงกระแสเป็นแรงดัน (Current to voltage converter) เพื่อให้ได้แรงดันเอาพุทอยู่ในช่วง 0 - 5 โวลท์



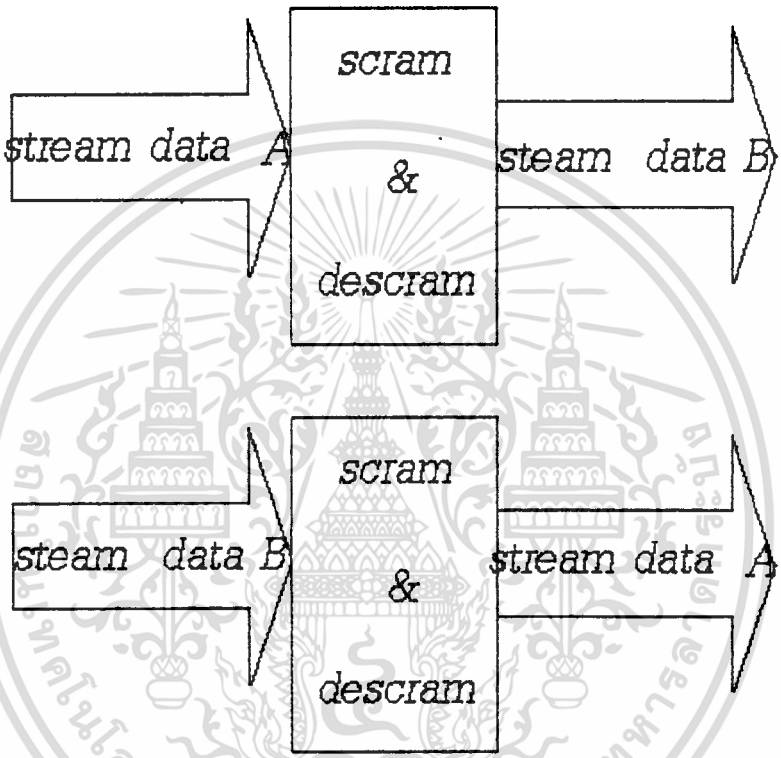
รูปที่ 4.22 แสดงลักษณะการวางขา ของ MC 1408

รูปที่ 4.23แสดงวงจรการใช้งาน ของ MC 1408



6.SCRAMBLE & DESCRAMBLE

วงจรการเข้ารหัสและถอดรหัสเป็นวงจรตัวเดียวกัน หมายความว่า เมื่อทำการเข้ารหัสก็ข้อมูล 64 บิท ก็ได้ ข้อมูลที่ผ่านการเข้ารหัสแล้ว 64 บิท เมื่อทำข้อมูล ก็เข้ารหัสมาผ่านอีกที ก็จะเป็นข้อมูลตัวเดิม โดยเงื่อนไขที่ว่า ค่าคีย์บิทจะต้องเหมือนกัน

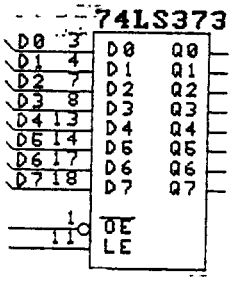


รูปที่ 4.23 BLOCK DIAGRAM ของ การส่งผ่านข้อมูล

โดยวงจรในบางส่วนของ SCRAMBLE & DESCRAMBLE จะถูกควบคุม โดยซีพียู 8031 ภายในวงจร SCRAMBLE & DESCRAMBLE แบ่งได้เป็น 4 ส่วนใหญ่ ๆ คือ

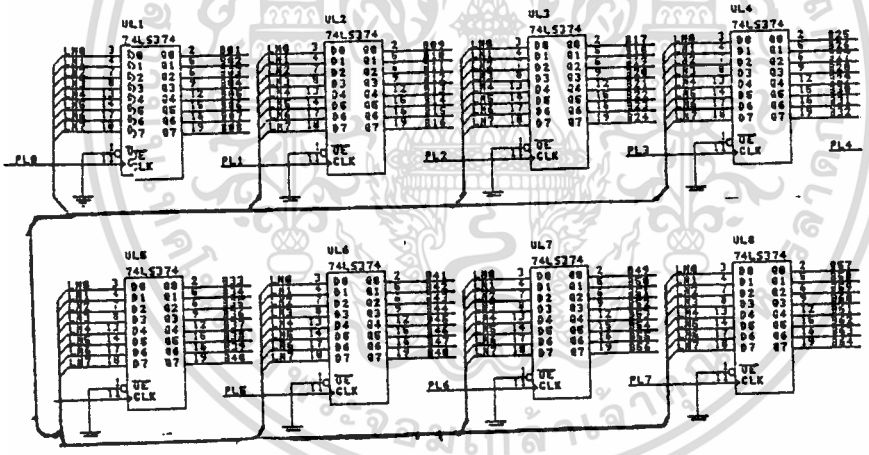
1. <u>ชุด latch ข้อมูลจาก CPU</u>

จากรูปจะเห็นว่า จะรับข้อมูลมากจากซีพียู โดยผ่านพอร์ท B ของ 8255 ข้อมูลนี้จะมาจากการแปลง A/D แล้วมาเก็บไว้ในค่าแรมภายในที่แอดเดรส 20H-27H และแต่ละตัวที่ขา 11 (clock) จะทำการเก็บค่าข้อมูลที่ขอบขาขึ้น ดังแสดงในตาราง ตรงขานี้เราจะใช้พอร์ท P1 ของ 8031 ควบคุมโดยจ่ายคล็อก 1 ลูก สภาวะปกติของแต่ละขาของพอร์ท P1 คือ high เรา จะบังคับให้ขาใดขาหนึ่งเป็น low แล้ว high เราก็จะได้คล็อก 1 ลูก ส่งไป ที่ 742S374 เพื่อ เก็บค่าที่เราส่งออกขา output enable จะถูกต้องลงกราวน์ เพื่อให้เอาท์พุทออกตลอดเวลา



รูปที่ 4.24 ภาพการแสดง ขา ของ 74 LS 374

ในวงจรจะใช้ latch 742S374 ทั้งหมด 8ตัว เพราะค่าที่ต้องการส่งมีทั้งหมด 8 ไบท์ (64 บิท) โดย latch แต่ละตัวจะรับได้ 1 ไบท์

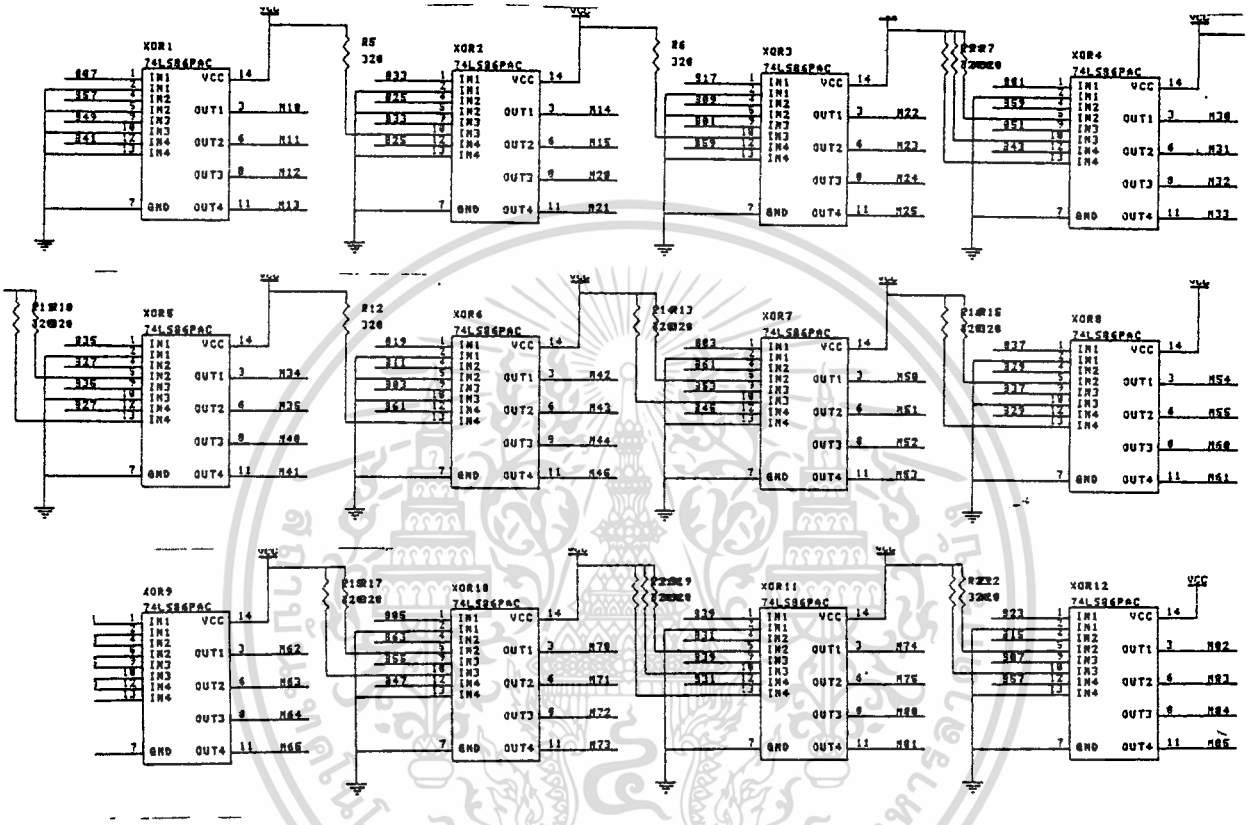


รูปที่ 4.25 แสดงการเชื่อมต่อครบทั้ง 8 ตัว

2. ชุด exclusive-or กับ ค่า key bit

ค่า keyเป็นค่าที่เราสามารถที่จะตั้งได้ตามความสะดวกของผู้ใช้ แต่ค่า key bit ของเครื่องส่ง และเครื่องรับปลายทางจะต้องมีค่า key bit ที่เหมือนกัน หากต่างกันข้อมูลที่ส่งไป

จะไม่สามารถถอดรหัสได้ ค่า key bit ในโครงงานนี้ได้ทำให้เลือกได้ โดยใช้ dip switch จำนวน 6 ตัว (คือ ค่า ket bit ทั้งหมด 48 บิท) นำมา exclusive-or กับค่าเอาท์พุทที่ ออกมาจาก latch ทั้ง 8 นั้น ค่าเอาท์พุทที่ออกจาก latch จะต้องผ่านการสลับบิทที่กล่าวมา แล้วในเรื่องการเข้ารหัส คือ initial permutation ดังแสดงในรูป (การเชื่อมต่อขาให้สังเกต comment ที่ขาและขา) และแบ่งเป็น 32 บิทซ้าย และขวา ทางด้านขวาจะนำมาผ่านการ expand bit ให้เปลี่ยนจาก 32 บิท เป็น 48 บิท ที่จะได้มีบิทเท่ากับค่า key bit ในการ exclusive-or



รูปที่ 4.26 รูปแสดงการเชื่อมต่อแต่บิทของ exclusive-or

3. ชุดการยุบบิทจาก 48 บิทเป็น 32 บิท

การยุบบิทจาก 48 บิทเป็น 32 บิท นำอินพุทมาจากเอาท์พุทของจุด exclusive-or กับค่า k ซึ่งในเอาท์พุททั้งหมด 48 บิท แบ่งแยกเป็น 8 ชุด ชุดละ 6 บิท มาเป็น address input ของ memory (EPROM) เบอร์ 2732 ซึ่งเป็นตัวเก็บข้อมูล

โดยให้ บิทแรก ที่มาเป็น A_6

บิทที่ 2 ที่มาเป็น A_4

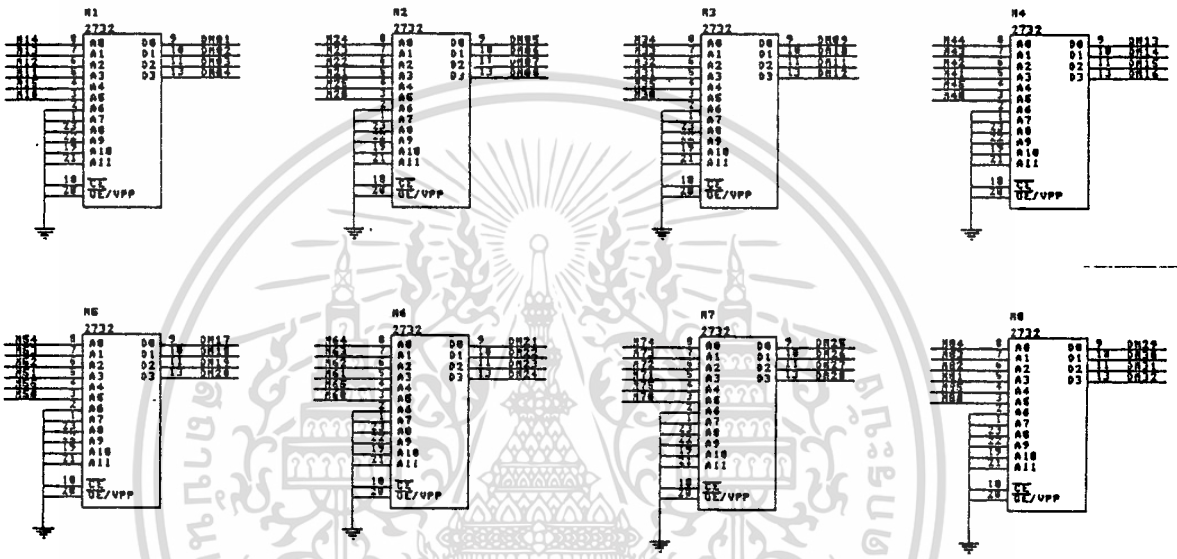
บิทที่ 3 ที่มาเป็น A_3

บิทที่ 4 ที่มาเป็น A_2

บิทที่ 5 ที่มาเป็น A_1

บิทที่ 6 ที่มาเป็น A_5

ถ้าเปรียบเทียบกับอัลกอริทึมของ DES ก็คือเปรียบ 2732 แต่ละตัวเป็น S_1 ถึง S_8 ดังรูปอัลกอริทึมที่แสดงการเชื่อม PIN แต่ละ PIN ดูได้ดังรูปที่



รูปที่ 4.27 แสดงการเชื่อมขาต่างๆ กับอุปกรณ์อื่น

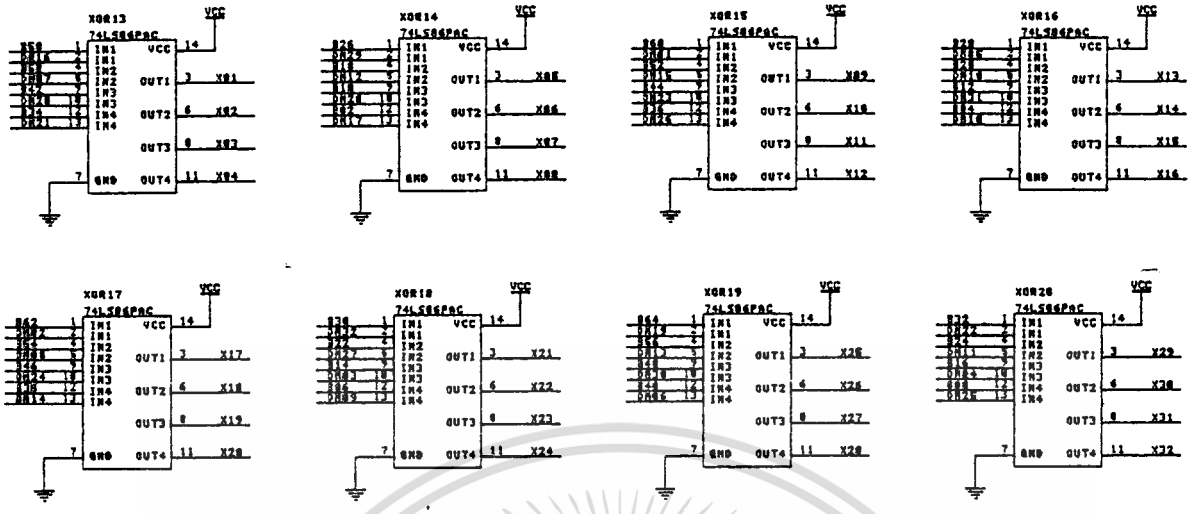
ขา CE และ OE จะลงกราวด์ตลอด MEMORY 2732 จะส่งค่าเอาท์พุทออกมาตลอด เมื่อมีแอดเดรสเข้ามา เอาท์พุทที่เราต้องการมีค่าต้องการเพียง 4 บิทเท่านั้น รวมทั้งหมด ก็จะได้ 32 บิท เรียงลำดับตั้งแต่ 1-32 ตามลำดับ

ค่าข้อมูลของแต่ละ MEMORY แสดงที่ภาคผนวก

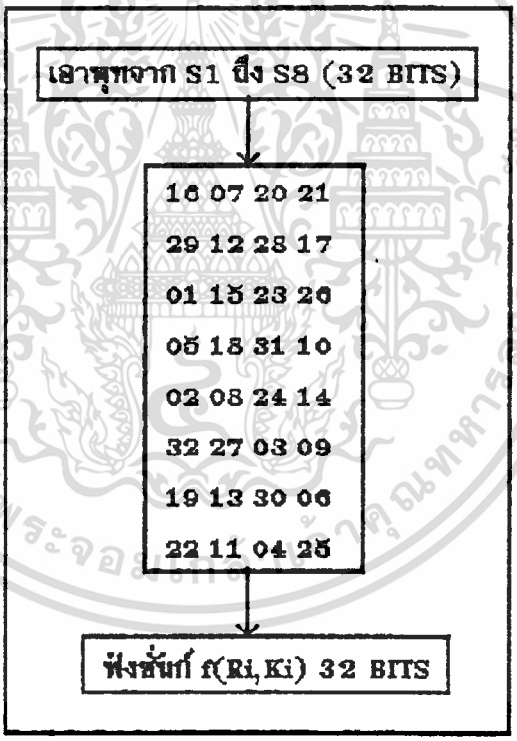
4. ชุด exclusive-or กับ bit output

บิทเอาท์พุทที่ออกมาจาก latch จะแบ่งออกเป็น 2 ส่วน ส่วนละ 32 บิท หลังจากการ ผ่าน initial permutation ส่วนแรกจะทำการ exclusive-or กับค่าคีย์บิท และอีกส่วน หนึ่งจะทำการ

exclusive-or กับข้อมูลที่ออกมาจาก EPROM 2732 ในจุดยุบบิทจาก 48 บิท เป็น 32 บิท โดย
มูลที่ออกมาจาก EPROM จะต้องเป็นการสลับบิทที่เรียกว่า PERMUTATION P



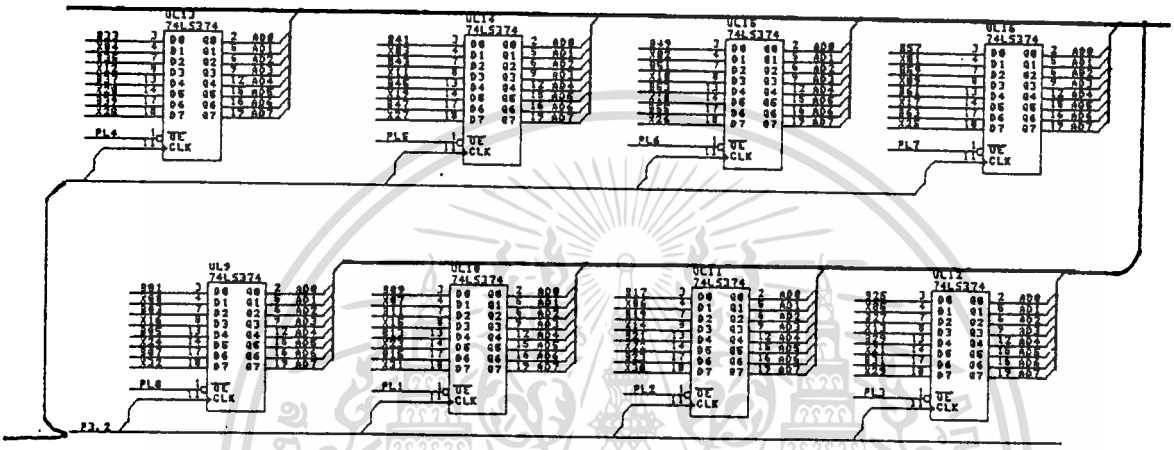
รูปที่4.28 แสดงการเชื่อมต่อขาต่าง ๆ กับรูปอื่น ๆ



ตารางที่ PERMUTATION P

5. ชุด LATCH OUTPUT

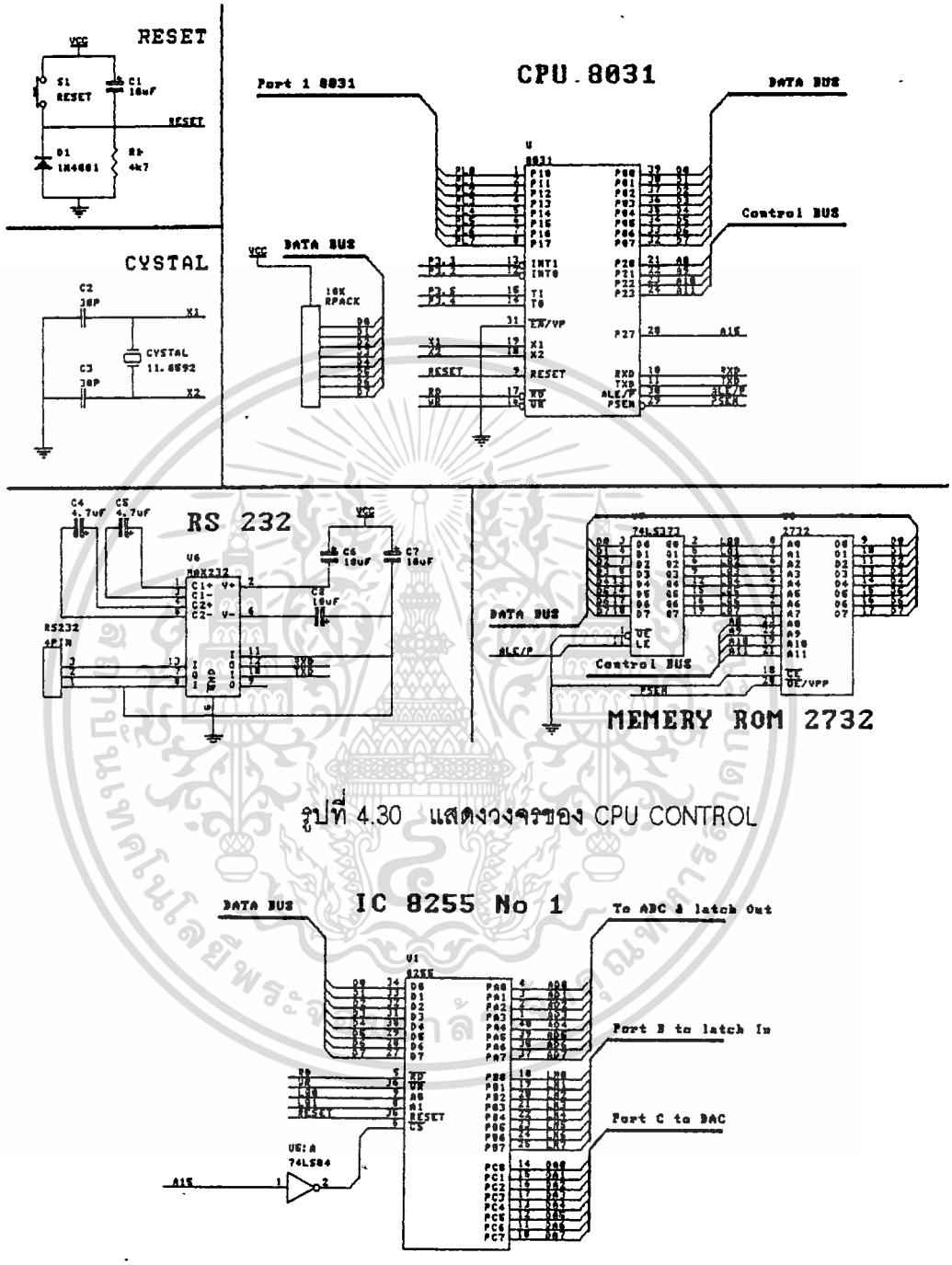
นำข้อมูลที่เป็นเอาท์พุทจาก exclusive-or ในวงจรที่ 4 มา latch เก็บค่าไว้แล้วนำ มารวมกับค่าที่ก่อนจะทำการ exclusive-or การคีย์บิทนำค่า 2 ชุด (ชุดละ 32 บิท) มาทำ การสลับบล็อกซ้ายขวา (TRANSFER FUNCTION) แล้วนำมาผ่านการสลับแบบย้อนกลับ เรียกว่า INVERSE PERMUTATION จากนั้นก็นำค่ามาเก็บไว้ใน latch output เพื่อที่จะรอส่งเข้าไปยัง ซีพียูทาง port C ของ 8255 โดยการเลือก control ขา 1 output enable ของแต่ละตัว โดยขาที่ 1 ของแต่ละ latch จะต่อกับขา P 3.2 ของ 8031 และขาข้อมูลของ latch จะต่อ กับ port A ของ 8255 ดังแสดงในรูป



รูปที่ 4.29 แสดงการเชื่อมต่อของ Latch Output

7. CPU CONTROL

ใช้ซีพียู 8031 ในการควบคุมวงจรต่าง ๆ ให้ทำงานกันอย่างมีความสัมพันธ์กัน ในส่วนของซีพียูไม่มีการใช้แรมภายนอก มีการเชื่อมเพียง EPOM ที่บรรจุโปรแกรมการทำงานไว้ และ 8255 ซึ่งทำหน้าที่ขยาย portโดย port A ทำหน้าที่เป็น port input รับค่าจาก ADC และ latch output จากวงจร SCRAMBLE port B เป็น port output ที่ส่งค่าให้ latch in ทั้ง 8 ตัว ส่วน port C จะทำหน้าที่เป็น output เช่นกัน โดยจะทำการส่งค่าที่ส่งผ่านการเข้า รหัสแล้ว ออกไปทาง DAC เพื่อที่จะแปลงเป็นสัญญาณอนาลอกต่อไป ดังแสดงในรูป



รูปที่ 4.30 แสดงวงจรของ CPU CONTROL

รูปที่ 4.31 การ เชื่อมต่อ 8255

การอ้างแอดเดรสของ 8255 จะอยู่ที่ 8000H-8003H

โดย port A = 8000H

port B = 8001H

port C = 8002H

control port = 8003H

โดยทำการต่อขาA15 ของ 8031 ผ่าน inverter ไปยังขา CS เมื่อ A15 = 1 จะ หมายว่า เลือก 8255 เป็น port อะไร จะใช้ค่า A0 และ A1 เป็นตัวบอก โดย A0 และ A1 ที่ผ่าน latch 74LS373 แล้ว

และวงจรการรับส่งข้อมูลผ่านทาง serial port ใช้ IC MAX 232 เป็นตัวช่วยในการรับ และส่ง ดังรูป โดยนำมาต่อร่วมกับขา TX และ RX ของ CPU 8031

โปรแกรมควบคุมการทำงานของระบบ

การควบคุมการทำงานของระบบ ใช้โปรแกรมของ MCS-51 ควบคุมการทำงานของระบบทั้งหมด

การเข้ารหัสทางโปรแกรมนั้น สามารถจะทำได้ แต่ว่าใช้เวลาในการเข้ารหัสมากเกินไป ทำให้ไม่มีเวลาพอในการสุ่มสัญญาณ จึงได้มีการคิดฮาร์ดแวร์ขึ้นมา

การควบคุมการทำงานของระบบประกอบไปด้วย

1. **การรับส่งข้อมูลทางพอร์ทอนุกรม** ซึ่งได้กล่าวทฤษฎีไปแล้วในบทต้น ๆ

การรับข้อมูล โปรแกรมจะทำการตรวจสอบบิท Ri ใน register SCON ว่ามีค่าเป็น 1 เมื่อไร แสดงว่ารับข้อมูลมาครบแล้วทั้ง 8 ไบท์ ซอฟท์แวร์ก็จะทำการเคลียร์บิทให้มีค่าความเป็น 0 เป็นการอีนาเบิ้ลในการรับต่อไป

การส่งข้อมูล จะใช้ลักษณะของการตรวจเช็คบิท Ti ว่าเป็น 1 เมื่อไร คือ มีการส่งข้อมูลออกไปครบทั้ง 8 บิท และเมื่อส่งค่าแล้วจะต้องทำการ CLR บิทนี้ด้วย เพื่อการอีนาเบิ้ลการส่ง

2. **ควบคุมวงจร** ANALOG TO DIGITAL

จะใช้ซอฟท์แวร์เป็นตัวกำหนดความถี่ในการแซมปลิ้ง การ enable chip และการเก็บค่าไว้ในแรมภายใน

3. **การควบคุมวงจร** DIGITAL TO ANALOG CONVERTER

ใช้โปรแกรมควบคุมในการเลือกจังหวะการทำงาน และส่งข้อมูลออกทีละ 8 ไบท์

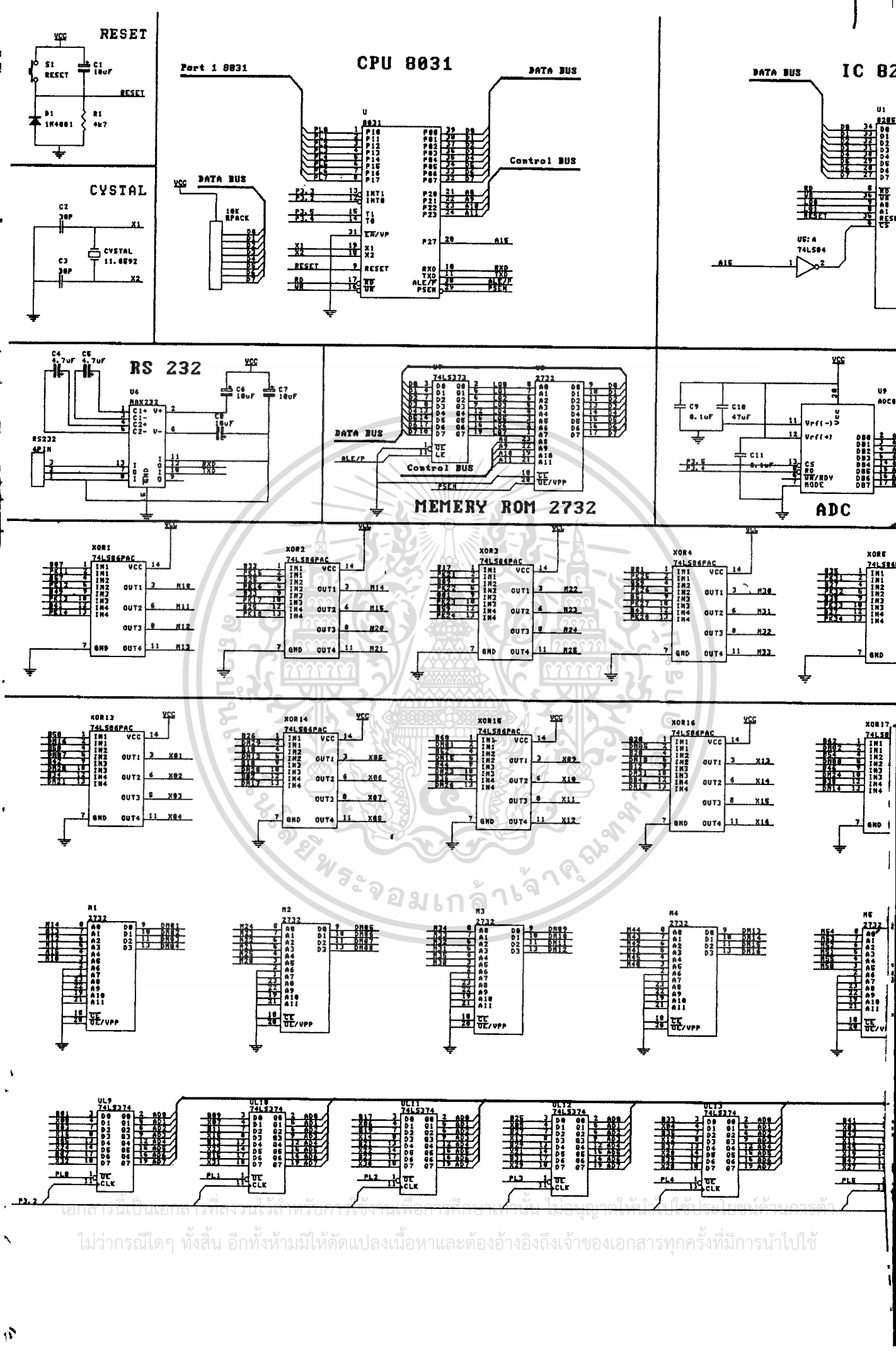
4. **ควบคุมวงจร** SCRAMBLE & DESCRAMBLE

จะเริ่มต้นด้วยการส่งข้อมูลออกทาง port B ของ 8255 โดยการควบคุมขา 11 ของ latch ทั้ง 8 ตัว ในวงจรโดยขา P1.0 - P1.7 จากนั้นจะเก็บค่าหลังจากข้อมูลออกมาทาง เอาท์พุทแล้ว โดยเก็บเข้าทางพอร์ท C ของ 8255 โดยการควบคุมขาเอาท์พุทอีนาเบิ้ลของ latch ทางข้อมูลขาออก

บทที่ 5
บทวิจารณ์และสรุป

โครงงานนี้ เป็นการนำเอาไมโครโปรเซสเซอร์ 8031 มาใช้ในการควบคุมส่วนต่าง ๆ ของวงจร การเข้ารหัสนั้นใช้วิธีแบบ DES ซึ่งได้ทำการออกแบบวงจรถอดรหัส และวงจรเข้ารหัส ซึ่งเป็นเป็นวงจรอันเดียวกัน โดยมีฮาร์ดแวร์ดังที่กล่าวไปแล้วข้างต้น การนำเสนอขอบเขตของผลงานเป็นลักษณะของการเชื่อมต่อแบบโทรศัพท์จำลอง จุดต่อจุด โดยมีการรับส่งในระยะทางใกล้ ๆ โดยใช้มาตราฐานแบบ RS232 ซึ่งในการพัฒนาต่อไป จะต้องพัฒนาเพื่อให้ทำการรับส่งได้ในสายโทรศัพท์จริง ๆ อาจจะใช้ลักษณะการแปลงข้อมูลจากดิจิตอลเป็นอนาลอกที่เครื่องส่ง และจากอนาลอกเป็นดิจิตอลที่เครื่องรับในลักษณะของโมเด็ม

วงจรการถอดรหัส และการเข้ารหัส สามารถถอดรหัส และเข้ารหัสได้อย่างถูกต้อง โดยผ่านการเข้ารหัสทีละ 8 ไบท์ ซึ่งตามมาตราฐานของ DES ก็คือ 64 บิทนั่นเอง แต่ค่า key bit โครงการนี้ถูกกำหนดให้ใช้แค่ 48 บิท โดยลดการทำงานทั้งหมด 16 รอบ เหลือเพียง 1 รอบ ซึ่งแน่นอนว่า มีความปลอดภัยในข้อมูลสูง โดยลดขั้นตอนที่เรียกว่า การสร้าง key bit 64 บิท ทั้ง 16 รอบ ทั้งนี้ key bit ในโครงงานนี้ถูกกำหนดให้เป็นข้อมูลที่สามารถกำหนดให้เปลี่ยนแปลงได้ด้วยตนเอง



To ABC & latch Out

Port B to latch In

Port C to DAC

ABC

DAC

DAC

บรรณานุกรม

1. Don J. Torrieri, "Principle of Secure Communication Systems" ARTECH HOUSE .Inc,pp367-411,1985

2. Don Lancaster , "Active - Filter Cookbook "' ,A Diision of Macmillan , Inc ., pp 367-411, 1975

3. Intel., "คู่มือไอซีไมโครโปรเซสเซอร์ MCS-51"

4. Willian Sinnema, "Digital, Analog, and Delta Communication Section edition", Prentice - Hall International. Inc., pp 301-305 ,1986

5. จิติพงษ์ ศรีสว่าง และ สุดฤทัย นันตะโรหิต, "การเข้ารหัสของสัญญาณเสียงอย่างมีประสิทธิภาพ", สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง, 2534

6.ประกอบ แซ่ซี , "เครื่องป้องกันการดักฟัง" , สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง, 2533

7. ดร.เลอสรร ธนสุกาญจน์, "เรียนรหัสด้วยคอมพิวเตอร์ ตอน รหัสมาตราฐาน", วารสารคอมพิวเตอร์

8. สมพงษ์ เกียรติคุณรัตน์, " Delta modulation " , สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง, 2528