เครื่องต่อเชื่อมโครงข่ายคอมพิวเตอร์

COMPUTER NETWORK INTERFACES

โดย

นายจตุพร เจตนานุวัฒนา

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิศวกรรมศาสตรบัณฑิต

สาขาอิเลคทรอนิคส์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2535

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

032534

ปริญญานิพนธ์ปีการศึกษา 2535

ภาควิชา อิเลคทรอนิคส์

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง เครื่องต่อเชื่อมโครงข่ายคอมพิวเตอร์

ผู้จัดทำ

นายจตุพร เจตนานุวัฒนา 321047

อาจารย์ที่ปรึกษา

(อิทธิชัย อรุณศรีแสงไชย)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

032534

เครื่องต่อเชื่อมโครงข่ายคอมพิวเตอร์

นายจตุพร เจตนานุวัฒนา 321047

อาจารย์ที่ปรึกษา

อาจารย์อิทธิชัย อรุณศรีแสงไชย

ปีการศึกษา 2535

บทคัดย่อ

โครงงานนี้จัดทำขึ้นสำหรับการสื่อสารข้อมูลระหว่างคอมพิวเตอร์ โดยได้ออกแบบให้เป็นลักษณะโลคัลแอเรียเนตเวอร์ก (Local Area Network) ใช้โครงสร้างแบบบัส อ้างอิงถึงโปรโตคอลแบบ CSMA/CD(Carrier Sense Multiple Access with Collission Detection) ซึ่งเป็นไปตามมาตรฐานของ IEEE 802.3 หรือที่เรียกกันว่า Ethernet หัวใจหลักของโครงงานนี้คือการสื่อสารแบบอนุกรม โดยใช้ชิป 8250 ซึ่งใช้เป็นที่แพร่หลายของพอร์ตอนุกรมในคอมพิวเตอร์ตระกูลไอบีเอ็ม ส่วนสำคัญของโครงงานนี้อาจแบ่งได้เป็น 2 ส่วน ได้แก่ฮาร์ดแวร์และซอฟแวร์ โดยส่วน ฮวร์ดแวร์จะสร้างเป็นการ์ดอินเตอร์เฟส ส่วนซอฟแวร์ก็จะครอบคลุมไปถึงโปรโตคอลและการประยุกต์ใช้งานเช่น Electronics Mails เป็นต้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

COMPUTER NETWORK INTERFACES

CHATUPORN CHETHANANUWATHANA 321047

ADVISER

ITTHICHAI ARUNSRISANGCHAI

ACADEMIC YEAR 1992

ABSTRACT

This research project emphasizes on data communications between personal computers in Local Area Networks with the configuration of bus topology and CSMA/CD (Carrier Sense Multiple Access with Collission Detection) protocol. CSMA/CD protocol is knowed in common name as " Ethernet" or IEEE802.3 standard.The 8250 chip is used for serial data com munications which is also used in general at the serial port of IBM PCs family. This project is devided into 2 main parts which are the part of hardware in the form of interface card and the part of software in the form of application protocol such as electronics mail etc.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ

		หน้า
บทที่ 1	บทนำ	1
บทที่ 2	ทฤษฎี	2
	การสื่อสารข้อมูล	2
	โครงข่ายท้องถิ่น	6
	โปรโตคอล CSMA/CD	8
	การส่งข้อมูลแบบบาลานซ์ RS-422-A,RS-485	12
บทที่ 3	หลักการออกแบบ	15
	แสดง block diagram interface card	17
	แสดงวงจรส่วนรับส่งข้อมูลและส่วนสร้างสัญญาณอ้างอิง	18
	ภาพแสดงชิ้นงาน	19
บทที่ 4	ผลการทดลอง	20
บทที่ 5	สรุป	22
ภาคผนวก	รวมโปรแกรม	
กิตติกรรมประกาศ		
หนังสืออ้างอิง		

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 1

บทนำ

ในชีวิตประจำวันการสื่อสารนับวันจะมีบทบาทมากขึ้น เริ่มตั้งแต่อดีตที่มีการคิดค้นรหัสมอร์ส ได้มีการพัฒนาเป็นโทรเลข จากโทรเลขก็พัฒนามาถึงยุควิทยุกระจายเสียง โทรพิมพ์ โทรศัพท์ จนกระทั่งมาถึงยุกสมัยของคอมพิวเตอร์ ความจำเป็นในการสื่อสารข้อมูลระหว่างคอมพิวเตอร์ก็มีมากขึ้น การพัฒนาการสื่อสารข้อมูลทำให้การใช้งานของคอมพิวเตอร์มีประโยชน์มหาศาล กิจการหลายอย่างได้อาศัยความสะดวกสบายของการสื่อสารข้อมูล เช่น งานด้านฐานข้อมูล สายการบิน การธนาคาร เป็นต้น ในช่วงไม่กี่ปีที่ผ่านมาเทคโนโลยีไมโครคอมพิวเตอร์ได้ก้าวหน้าขึ้นมากมีขีดความสามารถทางฮาร์ดแวร์สูงขึ้น ได้มีการพัฒนาการสื่อสารข้อมูลโดยนำไมโครคอมพิวเตอร์หลายๆเครื่องมาต่อรวมกันเป็นระบบที่เรียกว่า มัลติโปรเซสเซอร์สำหรับระบบมัลติโปรเซสเซอร์ที่เป็นที่นิยมกันในปัจจุบันก็คือระบบ LAN ซึ่งเป็นระบบที่เน้นในเรื่องความเร็วความแน่นอนในการส่งข้อมูล ในอาณาบริเวณที่ไม่ห่างไกลกันมากนัก และระบบ LAN ก็มีรูปแบบให้เลือกใช้มากมายหลายชนิดทั้งในด้านขนาดราคา และความสามารถ ให้ความเหมาะสมกับงานแต่ละอย่างได้ดี

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

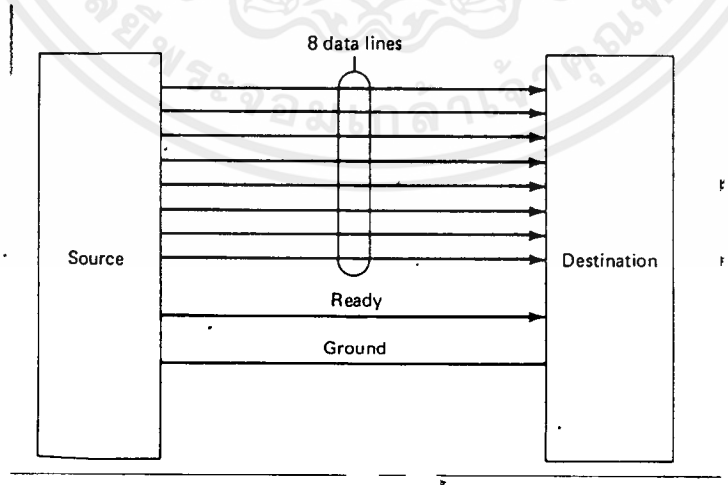
บทที่ 2

ทฤษฎี

การสื่อสารข้อมูล

การสื่อสารในระบบไมโครคอมพิวเตอร์ เป็นการส่งข้อมูลติดต่อกันระหว่างตัวคอมพิวเตอร์และอุปกรณ์รอบข้างได้แก่ คีย์บอร์ด จอแสดงผล และอื่นๆ โดยสามารถแยกออกได้เป็น 2 ชนิด คือการส่งข้อมูลแบบขนาน และการส่งข้อมูลแบบอนุกรม

1. การส่งข้อมูลแบบขนาน ทำโดยการส่งข้อมูลออกมาทีละ 1ไบต์ คือ 8 บิต จากอุปกรณ์ส่งไปยังอุปกรณ์รับ ตัวกลางระหว่าง 2 เครื่อง จะต้องมีช่องทางให้ข้อมูลเดินทางอย่างน้อย 8 ช่อง โดยมากจะเป็นสายขนานให้กระแสไฟฟ้าวิ่ง เนื่องจากมีสัญญาณสูญเสียไปกับความต้านทานของสาย ระยะทางระหว่าง 2 เครื่อง จึงสั้นไม่ควรเกิน 100 ฟุต และยังมีปัญหาของระดับกราวน์ทางไฟฟ้าที่จุดรับผิดไปจากจุดส่ง ทำให้เกิดการผิดพลาดของข้อมูล ซึ่งจะมีการแก้โดยมีสายควบคุมเพิ่มไปด้วย เช่น สายควบคุมการโต้ตอบ (Ready)

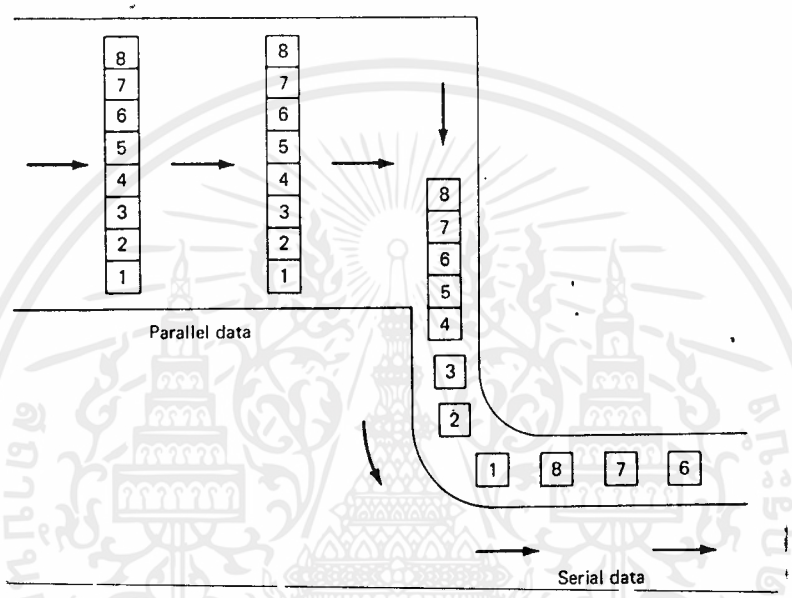


รูปที่ 1 การส่งข้อมูลแบบขนาน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2. การส่งข้อมูลแบบอนุกรม ข้อมูลจะถูกส่งออกมาทีละบิต ระหว่างจุดส่งและจุดรับ โดยการส่งข้อมูลแบบนี้จะช้ากว่าแบบขนาน แต่ที่ดีกว่าคือมีช่องทางการสื่อสารเพียงช่องเดียว และในกรณีการส่งระยะทางไกลๆ จะสามารถใช้ระบบการสื่อสารทางโทรศัพท์มาช่วยสนับสนุนได้

การสื่อสารข้อมูลแบบขนานจะถูกเปลี่ยนให้เป็นแบบอนุกรม และส่งผ่านตัวกลางจนถึงตัวรับ ตัวรับจะเปลี่ยนข้อมูลแบบอนุกรมไปเป็นแบบขนานอีกที



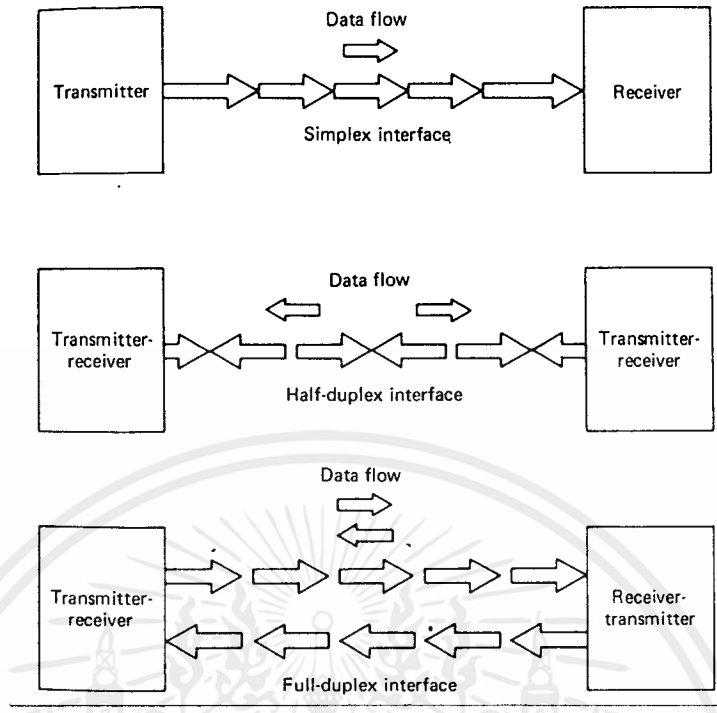
รูปที่ 2 การส่งข้อมูลแบบอนุกรม

<u>รูปแบบการสื่อสารข้อมูล</u>

การติดต่อแบบอนุกรมสามารถแบ่งได้ 3 แบบคือ

1. แบบซิมเพล็กซ์ (Simplex) ข้อมูลส่งได้ในทิศทางเดียวเท่านั้น
2. แบบฮาล์ฟดูเพล็กซ์ (Half Duplex) ข้อมูลสามารถส่งได้ทั้งสองสถานี แต่จะต้องผลัดกันรับส่ง จะส่งและรับพร้อมกันไม่ได้
3. แบบฟูลดูเพล็กซ์ (Full Duplex) ทั้งสองสถานีสามารถรับส่งได้ในเวลาเดียวกัน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



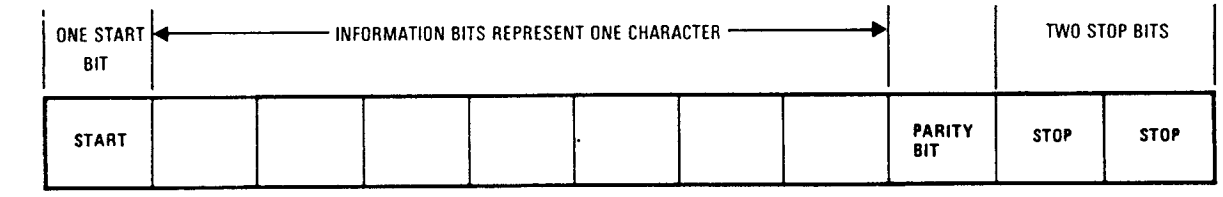
รูปที่ 3 รูปแบบการสื่อสารข้อมูล

ความเร็วของการถ่ายโอนข้อมูลแบบอนุกรม มีหน่วยเป็นบิตต่อวินาที (bps) และหน่วยที่แสดงถึงการเปลี่ยนแปลงข้อมูลของสัญญาณใน 1 วินาที เรียกว่า อัตราบอด (buad rate) สำหรับในโครงงานนี้อัตราบิตเท่ากับอัตราบอด

การสื่อสารแบบอนุกรมยังสามารถแบ่งออกได้เป็น การสื่อสารแบบซิงโครนัส และอะซิงโครนัส

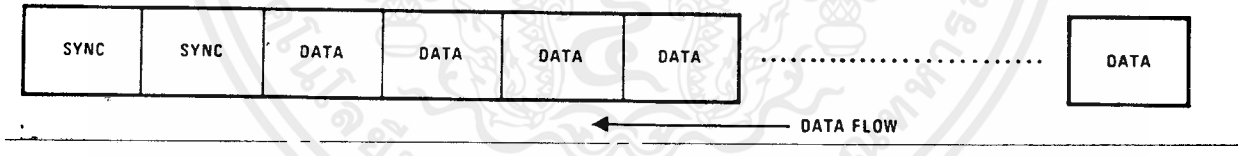
1. การสื่อสารแบบอะซิงโครนัส มีลักษณะแสดงดังรูปที่ 4 เป็นการส่งข้อมูลทีละ 1 ตัวอักษร สัญญาณจะประกอบด้วยบิตเริ่มต้นหรือบิตสตาร์ต บิตสิ้นสุดหรือบิตสต๊อป และอาจมีการแทรกบิตพาริตี้ ขณะที่สถานะการส่งเป็นแบบว่างคือ ยังไม่มีสัญญาณส่งออกมาจะมีสัญญาณหรือแรงดันตลอดเวลา เพื่อความแน่ใจว่าฝ่ายรับยังติดต่ออยู่กับฝ่ายส่ง เมื่อเริ่มจะส่งข้อมูล สัญญาณจะมีค่าเป็น 0 หนึ่งช่วงสัญญาณนาฬิกา เรียกว่าสตาร์ตบิต ตามหลังของสตาร์ตบิตก็จะเป็นข้อมูลสำหรับ 1 ตัวอักษร อาจมีขนาดตั้งแต่ 5 บิต ถึง 8 บิต โดยการเข้ารหัสก็จะใช้รหัส ASCII ตามหลังข้อมูลก็จะเป็นพาริตี้บิต ซึ่งใช้ตรวจสอบความถูกต้องอาจจะมีหรือไม่มีก็ได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4 การสื่อสารแบบอะซิงโครนัส

2. การสื่อสารแบบซิงโครนัส เป็นการส่งข้อมูลของกลุ่มตัวอักษร คือ ข้อมูลมีความต่อเนื่องในการส่งไม่มีสตาร์ตบิต สต๊อปบิต หรือพาริตี้บิตเหมือนในแบบอะซิงโครนัส โปรโตคอลที่ใช้เป็นในระดับอักขระ แต่ละตัวมีขอบเขตแน่นอน ไม่มีขอบเขตจำกัดของจำนวนข้อมูลที่จะส่ง การซิงโครนัสจะกระทำที่ตอนเริ่มต้นของชุดข้อมูล ส่วนหน้าของข้อมูลอาจมีการบรรยายข้อมูลที่ส่งมา และมีส่วนตรวจสอบข้อมูลในตอนท้ายของชุดข้อมูล โปรโตรคอลที่ใช้เป็นมาตรฐานเช่น Bisyn, SDLC



รูปที่ 5 การสื่อสารแบบซิงโครนัส

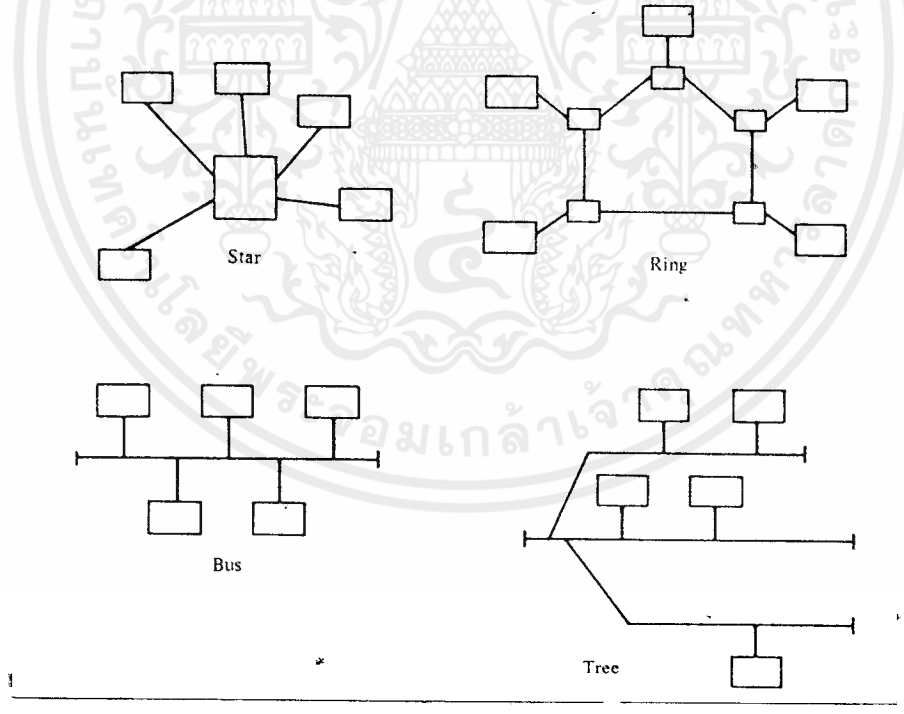
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โครงข่ายท้องถิ่น (Local Area Network)

ระบบ LAN คือ การนำเอาระบบไมโครคอมพิวเตอร์หลายเครื่อง มาต่อรวมกันเป็นเนตเวอร์ก ซึ่งมีจุดมุ่งหมายคือ ต้องการแลกเปลี่ยนข้อมูลข่าวสาร และต้องการแบ่งการใช้ในอุปกรณ์ที่มีราคาแพงร่วมกัน เช่น เครื่องพิมพ์ ,ฮาดดิสค์

คำว่า โทโปโลยี่ เป็นคำศัพท์ที่ใช้ในเรื่องระบบเนตเวอร์ก ซึ่งหมายถึงรูปร่างลักษณะของเนตเวอร์ก ในที่นี้จะใช้คำว่า โครงสร้าง แทน โครงสร้างที่นิยมใช้กันก็เช่น แบบดาว(Star) แบบบัส(Bus) แบบต้นไม้(Tree) และแบบวงแหวน(Ring)

โครงสร้างแบบต่างๆ ก็มีจุดดีจุดด้อยแตกต่างกันไปแล้วแต่ว่าจะเลือกใช้งานในลักษณะใดที่เหมาะสม



รูปที่ 6 แสดงโครงสร้างแบบต่างๆของโครงข่ายท้องถิ่น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สำหรับในโครงงานนี้ได้เลือกโครงสร้างแบบ บัส ซึ่งเป็นแบบที่มีโครงสร้างไม่ยุ่งยาก ไม่จำเป็นต้องใช้อุปกรณ์สวิตชิ่งเหมือนกับแบบดาว หรือรีพีตเตอร์เหมือนกับแบบวงแหวน ทุกๆสถานีจะเชื่อมต่อเข้าหาบัสโดยผ่านอุปกรณ์อินเตอร์เฟสที่เป็นฮาดแวร์ และโครงสร้างแบบบัสจัดได้ว่าเป็นระบบที่มีความยืดหยุ่นสูง ทุกสถานีมีความสำคัญในระดับเท่าเทียมกัน ไม่ขึ้นกับตัวใดตัวหนึ่ง สามารถเพิ่มหรือขยายระบบได้ง่าย

ตัวกลางที่ใช้ในระบบ LAN ก็มีหลายชนิด เช่น สายคู่บิตเกลียว (Twisted pair) สายโคแอกเชียล (Coaxial cable) เส้นใยแสง (Optical fiber) โดยตัวกลางแต่ละชนิดก็มีจุดดีจุดด้อยแตกต่างกันออกไป เช่น ในด้านของราคา, ความต้านทานต่อสัญญาณรบกวน, ขอบเขตและคุณสมบัติในการรับส่งสัญญาณ ซึ่งการที่จะเลือกตัวกลางชนิดใดก็แล้วแต่ความเหมาะสมของงาน

การรับส่งสัญญาณในตัวกลางที่กล่าวมา มีวิธีการทางเทคนิคอยู่ 2 แบบ คือ เบสแบนด์ และบอร์ดแบนด์ ความหมายของเบสแบนด์คือ การส่งสัญญาณข้อมูลเดิมโดยแท้จริงไม่มีการมอดูเลต ซึ่งเมื่อนำมาใช้กับระบบ LAN ก็คือการส่งสัญญาณเป็นดิจิตอลนั่นเอง สำหรับบอร์ดแบนด์นั้นหมายถึงการสื่อสารในช่วงความถี่ที่ใช้แบนด์วิดท์เกินกว่า 4 กิโลเฮิรตซ์ และในระบบ LAN ก็จะเป็นในลักษณะการส่งสัญญาณที่ใช้การมอดูเลตและการมัลติเพล็กซ์

จากโมเดลของการสื่อสารข้อมูลที่กำหนดโดย ISO (International Organization For Standardization) ได้แบ่งระดับของการสื่อสารข้อมูลเป็น 7 ระดับ โดยระดับต่างๆ จะควบคุมการทำงานให้เป็นไปตามมาตรฐานได้แก่

1. Physical layer
2. Data Link layer
3. Network layer
4. Transport layer
5. Session layer

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

6. Presentation layer

7. Application layer

โดย 4 ระดับล่างคือ (1-4)เป็นเรื่องเกี่ยวกับเนตเวอร์ก และอีก 3 ระดับต่อไป (5-7) เป็นเรื่องเกี่ยวกับซอฟแวร์และการประยุกต์ใช้งาน

ในที่นี้จะกล่าวถึงระดับที่เกี่ยวข้องกับโครงงานที่จะทำคือ

1.Physical layer เป็นระดับที่เกี่ยวข้องกับการรับส่งข้อมูลเป็นบิต ระดับแรงดันไฟฟ้า อุปกรณ์ทางกลและที่เกี่ยวข้องกับการเชื่อมต่อ

2.Data Link layer เป็นระดับที่ทำการจัดรูปแบบข้อมูลเป็นแพกเก็จ (Packet) และมีการตรวจสอบข้อผิดพลาดเช่น CRC

โปรโตคอล CSMA/CD

โครงงานนี้ได้จัดทำขึ้นโดยอ้างอิงถึงมาตรฐาน IEEE 802.3 ซึ่งเป็นบัสเนตเวอร์กใช้โปรโตรคอล CSMA/CD และเป็นที่รู้จักกันดีในชื่อว่า Ethernet

CSMA เป็นโปรโตรคอลที่ได้พัฒนามาจาก ALOHA โดย ALOHA เป็นวิธีการส่งสัญญาณวิทยุเป็นแพกเก็ต มีสถานีภาคพื้นดินหลายสถานีรวมกันเป็นเนตเวอร์กและโครงร่างก็คล้ายกับระบบบัสที่มีตัวกลางเป็นอากาศ

CSMA (Carrier Sense Multiple Access) มีการทำงานในลักษณะฟังก่อนพูด การส่งข้อมูลของสถานีใดจะเริ่มด้วยการตรวจสอบตัวกลางนำสัญญาณก่อนที่จะส่งข้อมูล แบ่งได้เป็น 3 ชนิดคือ

1) nonpersistent CSMA มีขั้นตอนคือ

1.1 สถานีที่จะส่งข้อมูลจะตรวจสอบตัวกลาง หากตัวกลางว่างจะส่งข้อมูล

1.2 ถ้าตัวกลางนำสัญญาณไม่ว่าง จะรอด้วยช่วงเวลาหนึ่ง ซึ่งมีค่าเป็นเลขสุ่มและให้กลับไปทำตามข้อ 1.1 (การใช้ตัวเลขสุ่มเพื่อลดการชนกันของข้อมูล)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2) 1-persistent CSMA มีขั้นตอนคือ

2.1 สถานีที่จะส่งข้อมูลจะตรวจสอบตัวกลาง หากตัวกลางว่างจะส่งข้อมูล

2.2 ถ้าหากตัวกลางไม่ว่างก็จะทำการตรวจสอบจนกระทั่งตัวกลางว่าง แล้วจะส่งข้อมูลทันที

2.3 ถ้าหากเกิดการชนกันของข้อมูล (collision)จะรอด้วยคาบเวลาช่วงหนึ่งที่ได้จากการสุ่ม และทำซ้ำข้อ 2.1

3) p-persistent CSMA มีขั้นตอนคือ

3.1 สถานีที่จะส่งข้อมูลจะตรวจสอบตัวกลาง หากตัวกลางว่างจะมีโอกาสส่งข้อมูลหรือไม่ส่งข้อมูลโดยใช้ค่าความน่าจะเป็น มาคำนวณเทียบกับค่าที่ได้จากการสุ่มหากสถานีไม่ส่งข้อมูลจะรอด้วยค่าหนึ่งช่วงเวลา

3.2 หากตัวกลางไม่ว่างจะตรวจสอบต่อไปจนกระทั่งว่าง แล้วทำซ้ำข้อ 2.1

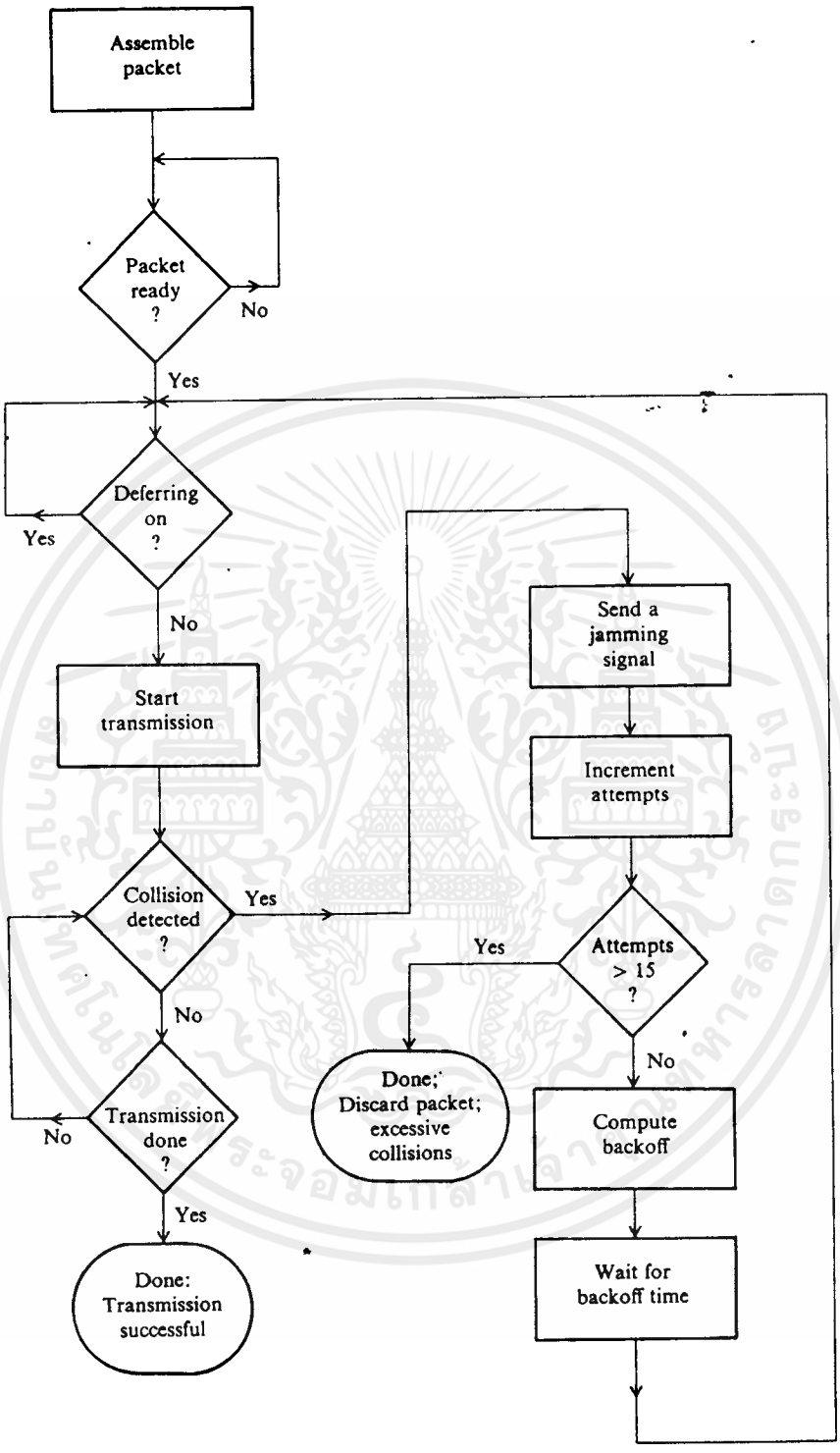
3.3 เมื่อรอหนึ่งช่วงเวลาในข้อ 3.1 เสร็จแล้วจะทำซ้ำข้อ 3.1

CSMA/CD (Carrier Sense Multiple Access with Collision detection)

ถึงแม้ว่าระบบ CSMA จะมีประสิทธิภาพดี แต่ก็ยังมีการชนกันของข้อมูลเกิดขึ้น ข้อมูลที่ 2 สถานีส่งออกมาในเวลาเดียวกันจะเป็นข้อมูลเสีย และใช้เวลาในการที่จะพบการชนกันของข้อมูลอย่างน้อยที่สุด เท่ากับเวลาเดินทางของข้อมูลระหว่าง 2 สถานีที่ทำการส่ง จึงได้มีการเพิ่มระบบความปลอดภัย ซึ่งถ้าหากว่าสถานีใดตรวจพบการชนกันของข้อมูลก็จะมีการส่งสัญญาณ Jamming เพื่อบอกให้ทั้งระบบรับทราบ หลังจากที่สถานีใดๆ รับทราบการ Jamming ก็จะรอด้วยคาบเวลาที่ได้จากการสุ่มแล้วจึงทำการส่งข้อความเดิมอีกครั้ง

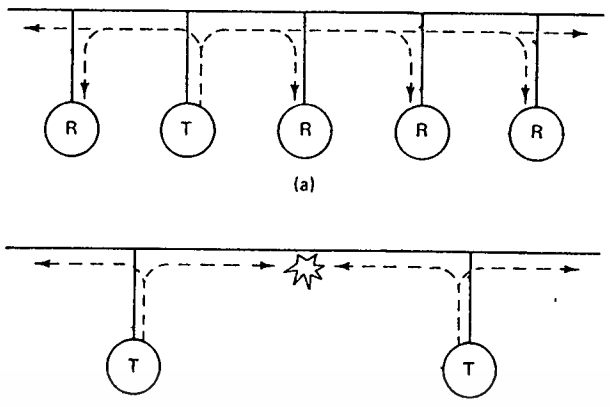
โดยทั่วไปจะตรวจสอบการชนกันของข้อมูลจากระดับไฟตรงของสัญญาณในตัวกลาง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้
032534



รูปที่ 7 แสดง flow diagram CSMA/CD

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

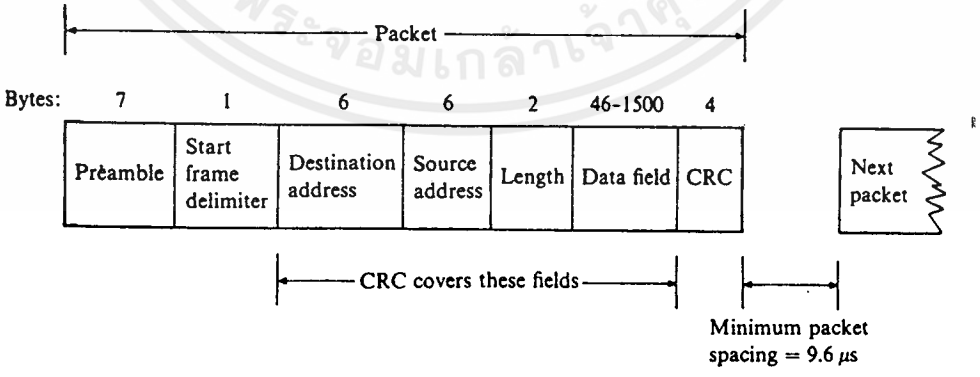


Bus broadcast and collision:
(a) Broadcast transmission, (b) Collision.

รูปที่ 8 การส่งข้อมูลแบบ CSMA/CD

รูปแบบของแพกเก็จตาม CSMA/CD ก็เป็นลักษณะเฟรมข้อมูลที่ประกอบด้วย

- ปรีแอมเบิ้ล (Preamble) ใช้สำหรับให้ตัวรับซิงโครไนซ์การรับส่ง
- DA (Destination Address) บอกแอดเดรสของสถานีปลายทาง ซึ่งอาจเป็นแบบเจาะจง หรือเป็นกลุ่ม
- SA (Source Address) บอกแอดเดรสตัวส่งสัญญาณ
- ข้อมูล (Data Field) เป็นข้อมูลที่ใช้ในการสื่อสาร
- FCS (Frame Check Seguence) เป็นรหัสตรวจสอบข้อมูล



รูปที่ 9 แสดงแพกเก็จของ CSMA/CD

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เนื่องจากในโครงงานนี้ ใช้มาตรฐาน RS-485 เป็นระบบของการส่งสัญญาณใน network bus จึงขอกล่าวถึงรายละเอียดดังต่อไปนี้

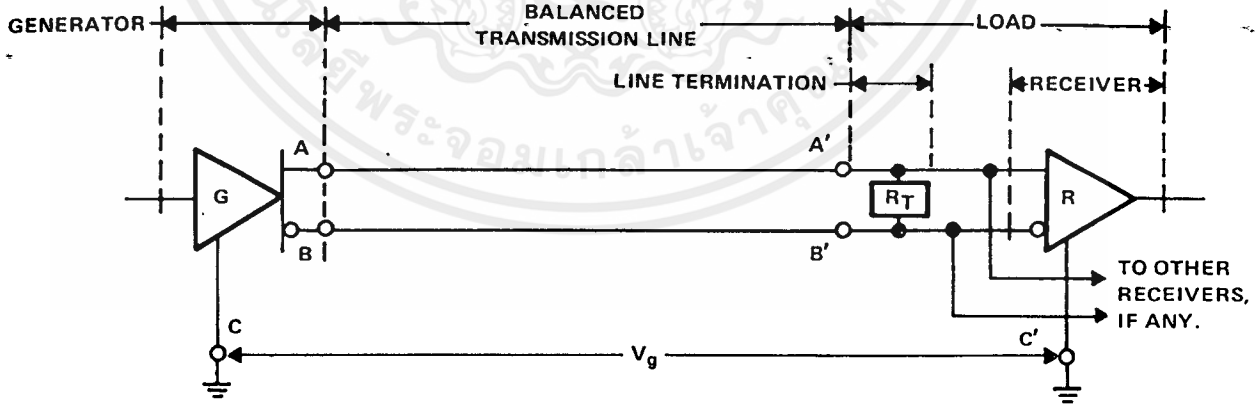
การส่งข้อมูลแบบบาลานซ์ RS-422-A, RS-485

การส่งข้อมูลแบบบาลานซ์(balance) เป็นการรับส่งข้อมูลตามมาตรฐานของ EIA มีคุณสมบัติเหนือกว่าส่งข้อมูลแบบซิงเกิ้ลเอ็นด์(single ended) เช่น RS-232-C มีความสามารถในการรับส่งสัญญาณได้ในอัตราสูง 100 Kbps -10 Mbps สามารถจำแนกข้อดีได้ดังนี้

1. สามารถรับส่งได้ในระยะทางไกลถึง 4000 ft
2. ลดผลกระทบจากสัญญาณรบกวนได้สูงกว่าแบบซิงเกิ้ลเอ็นด์

ระบบของการส่งสัญญาณตามมาตรฐาน RS-422-A นี้จะประกอบไปด้วย

1. ตัวขับสัญญาณ(line driver)
2. สายส่ง(transmission line) โดยทั่วไปคือ twisted pair
3. โหลดซึ่งจะประกอบด้วยตัวรับสัญญาณ(receiver) และตัวต้านทานปลายสาย(terminate resister)



LEGEND: R_T = Terminating resistance, V_g = Ground voltage difference
A, B = Generator-Line Interface, A', B' = Load-Line interface
C = Generator ground C' = Load ground.

รูปที่ 10 แสดงการส่งสัญญาณตามมาตรฐาน RS-422-A

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

PARAMETER		RS-232-C	RS-423-A	RS-422-A	RS-485
Mode of operation		Single-ended	Single-ended	Differential	Differential
Number of drivers and receivers allowed		1 Driver 1 Receiver	1 Driver 10 Receivers	1 Driver 10 Receivers	32 Drivers 32 Receivers
Maximum cable length (ft)		50	4000	4000	4000
Maximum data rate bits per second		20k	100k	10M	10M
Maximum common-mode voltage		± 25 V	± 6 V	6 V -0.25 V	12 V -7 V
Driver output		± 5 V min ± 15 V max	± 3.6 V min ± 6.0 V max	± 2 V min	± 1.5 V min
Driver load		3 kΩ to 7 kΩ	450 Ω min	100 Ω min	60 Ω min
Driver slew rate		30 V/μs max	Externally controlled	NA	NA
Driver output short circuit current limit		500 mA to V_{CC} or GRD	150 mA to GRD	150 mA to GRD	150 mA to GRD 250 mA to 8 V or 12 V
Driver output resistance (High Z state)	Power on	NA	NA	NA	120 kΩ
	Power off	300 Ω	60 kΩ	60 kΩ	120 kΩ
Receiver input resistance Ω		3 kΩ to 7 kΩ	4 kΩ	4 kΩ	12 kΩ
Receiver sensitivity		± 3 V	± 200 mV	± 200 mV	± 200 mV

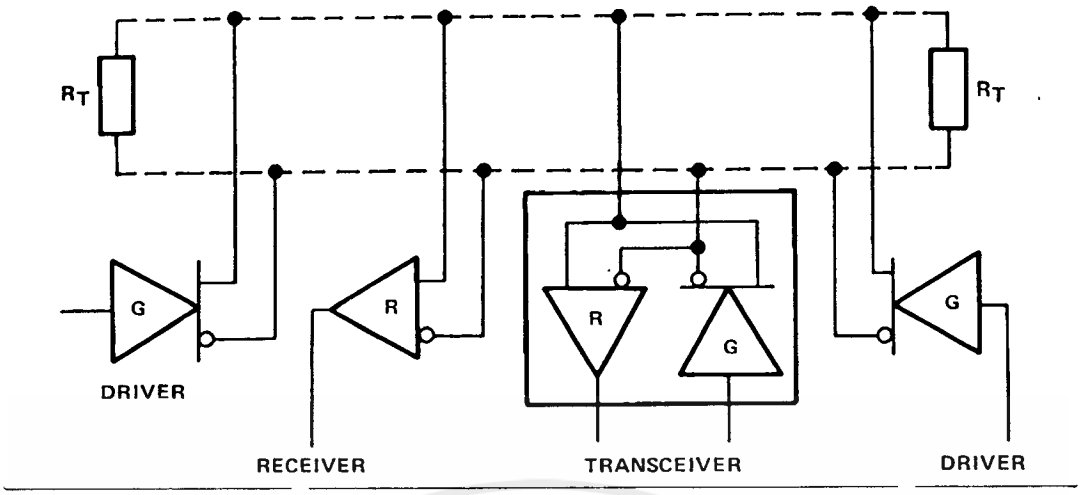
รูปที่ 11 ตารางแสดงคุณสมบัติการส่งสัญญาณตามมารตฐานของ EIA

ข้อแตกต่างระหว่างมาตรฐาน RS-422-A และ RS-485

RS-485 เป็นระบบที่พัฒนามาจาก RS-422-A โดยโครงข่ายของ RS-422-A จะประกอบไปด้วย ตัวขับสัญญาณ 1 ตัว ตัวรับสัญญาณ 10 ตัว ซึ่งเป็นลักษณะโครงข่ายแบบ point to multipoints ส่วนกรณีของ RS-485 จะประกอบด้วยตัวขับสัญญาณ 32 ตัว ตัวรับสัญญาณ 32 ตัวเป็นโครงข่ายแบบ multipoints to multipoints

สำหรับตัวต้านทานปลายสายโดยทั่วไปจะมีค่าเท่ากับค่าอิมพีแดนซ์ของสายส่งมีค่าประมาณ 120 ohm และจะมีเพียง 2 ตัวที่ปลายสายทั้ง 2 ด้าน ดังแสดงในรูปที่ 12

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 12 แสดงการประยุกต์ใช้งานตามมารตฐาน RS-485

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 3

หลักการออกแบบ

การออกแบบการ์ดอินเตอร์เฟสสำหรับการสื่อสารแบบ อะซิงโครนัส สามารถแบ่งได้เป็นโครงสร้างหลัก 3 ส่วน คือ

1. ส่วนรับส่งข้อมูลใช้ชิป 8250
2. ส่วนสร้างสัญญาณอ้างอิงใช้ชิป 8254
3. ส่วนตรวจสอบการชนกันของข้อมูล

ส่วนรับส่งข้อมูลเป็นส่วนที่ทำหน้าที่ติดต่อกันในเนตเวอร์ก ส่วนสร้างสัญญาณอ้างอิงจะเป็นส่วนสนับสนุนทางด้านเวลา สำหรับส่วนตรวจสอบการชนของข้อมูลจะทำหน้าที่ตรวจสอบความผิดพลาดที่อาจจะเกิดขึ้นได้ในระบบ

ได้เลือกพอร์ตที่แอดเดรส 300-310โดยใช้ 74LS30 8-input Positive Nand Gate เป็นตัวเลือกช่วงระหว่าง 300-31F (แอดเดรส 300-31F เป็นแอดเดรสของ Prototype Card ของเครื่อง IBM- PCและเครื่องในตระกูล IBM ที่ compatible) ใช้ 74LS138 เป็นตัวเลือกพอร์ตของแต่ละส่วน ส่วนรับส่งข้อมูลใช้พอร์ต 300-307 ส่วนสร้างสัญญาณอ้างอิงใช้พอร์ต 308-30F ส่วนตรวจสอบการชนของข้อมูลใช้พอร์ต 310

ในส่วนของวงจรรับส่งข้อมูลจะมี 74LS245 เป็นบัฟเฟอร์ของดาต้าบัส [Do-D7] 74LS244 เป็นบัฟเฟอร์ของแอดเดรสบัสและสัญญาณควบคุมที่สำคัญได้แก่ Reset,IRQ,IOW,IOR บิตเรตกำหนดไว้ที่ 19.2 Kbps ใช้ Single Packet Crystal Oscillator เป็นตัวกำเนิด Clock ความถี่อ้างอิงที่ 1.8432 MHzให้กับ 8250,8254 ใช้ DS75176 RS422/RS485 Transceivers เป็นตัวรับส่งสัญญาณกับบัสของเนตเวอร์ก

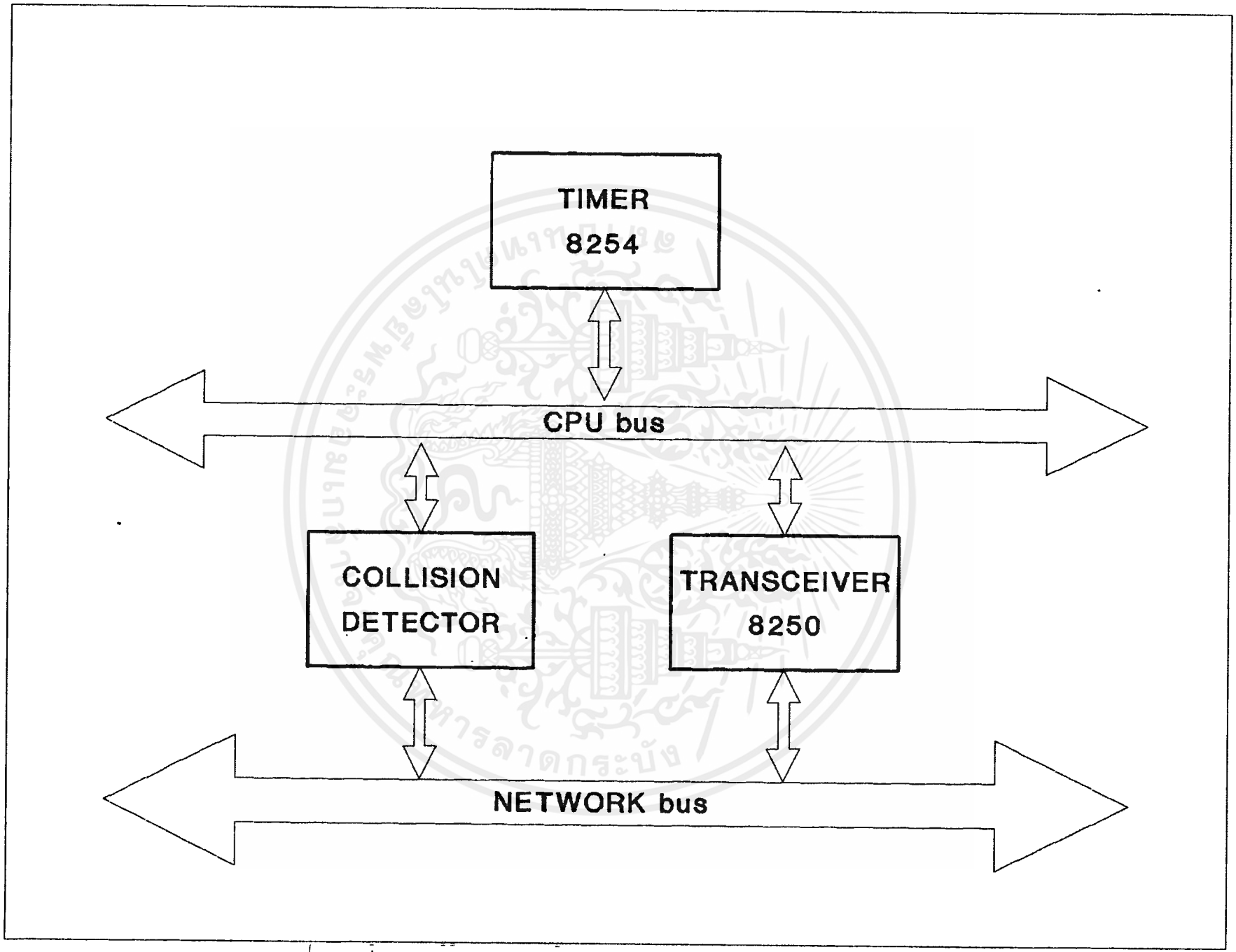
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การโปรแกรมการทำงานของชิป 8250 ระหว่างพอร์ต 300-307 จะไม่กล่าวในที่นี้โดยจะหาอ่านได้จากคู่มือ และในโครงงานได้โปรแกรมให้ทำงานที่บิตเรต 19.2 Kbps ข้อมูลขนาด 8 บิต สต๊อปบิต 1 บิต และไม่มีพาริตี้บิต

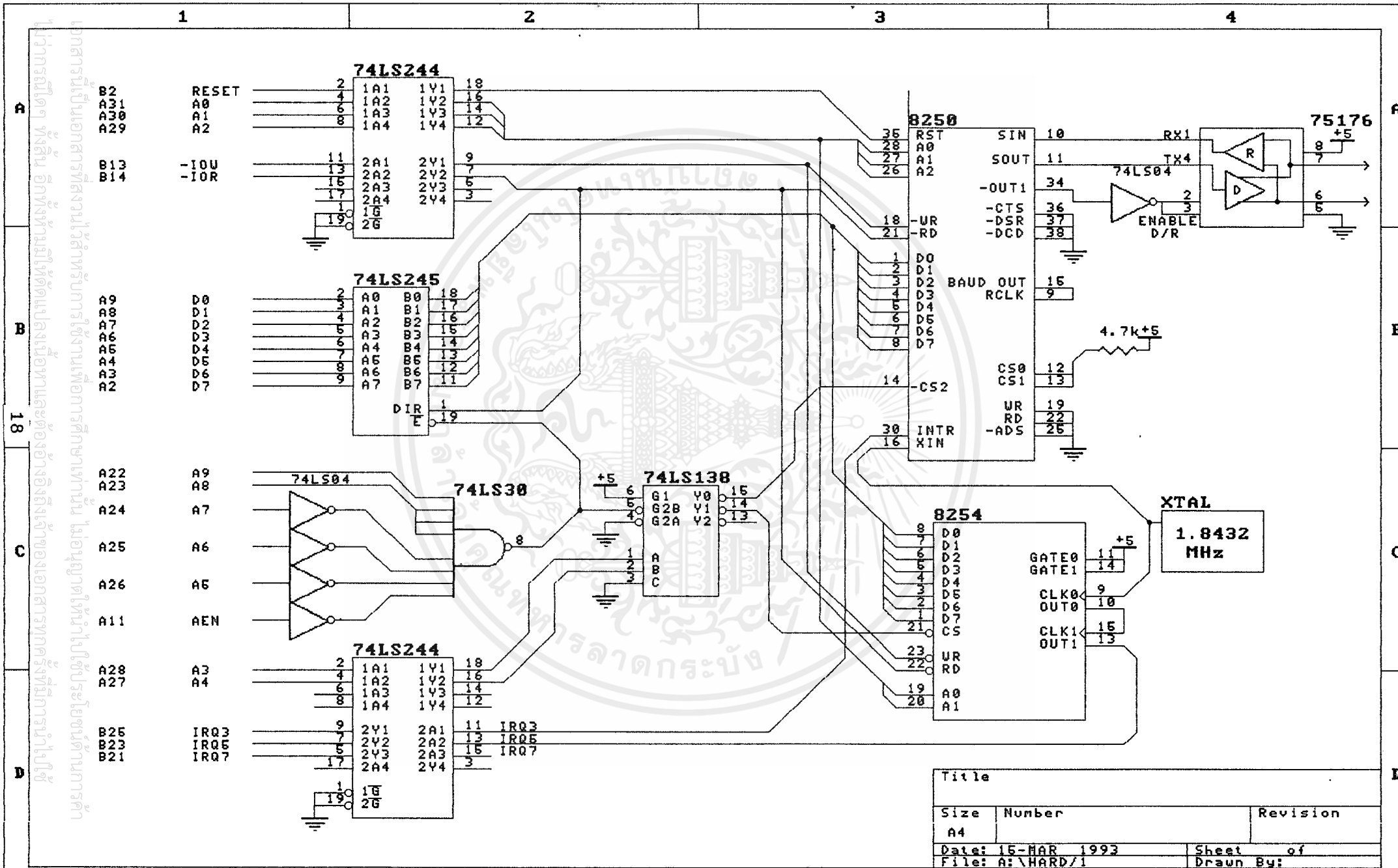
ส่วนของวงจรสร้างสัญญาณอ้างอิง ไทม์เมอร์ชิป 8254 จะรับ clock โดยมีบัฟเฟอร์ 74LS14 ภายใน 8254 จะมีเคาน์เตอร์อยู่ 3 ชุด ชุด 0,1,2 จะโปรแกรมเคาน์เตอร์ 0 ในโมด rate generator สร้างสัญญาณอ้างอิง 100 KHz เอาพุตของเคาน์เตอร์ 0 จะใช้เป็น clock ของเคาน์เตอร์ 1 เคาน์เตอร์ 1 จะถูกโปรแกรมในโมด interrupt on terminal count เป็นเคาน์เตอร์แบบนับลงที่ควบคุมการโหลดข้อมูล และจะส่ง intertupt ให้ CPU ถ้าหากใช้เวลาในการรับส่งข้อมูลผิดปกติ

ส่วนตรวจสอบการชนของข้อมูล สร้างโดยใช้หลักการของวงจรคอมพาราเตอร์ เป็นตัวเปรียบเทียบระดับของแรงดันในตัวกลาง ถ้าหากมีการชนกันของข้อมูลเกิดขึ้น ผลรวมของระดับแรงดันในตัวกลางจะมีค่าเพิ่มขึ้น และจะถูกตรวจพบ

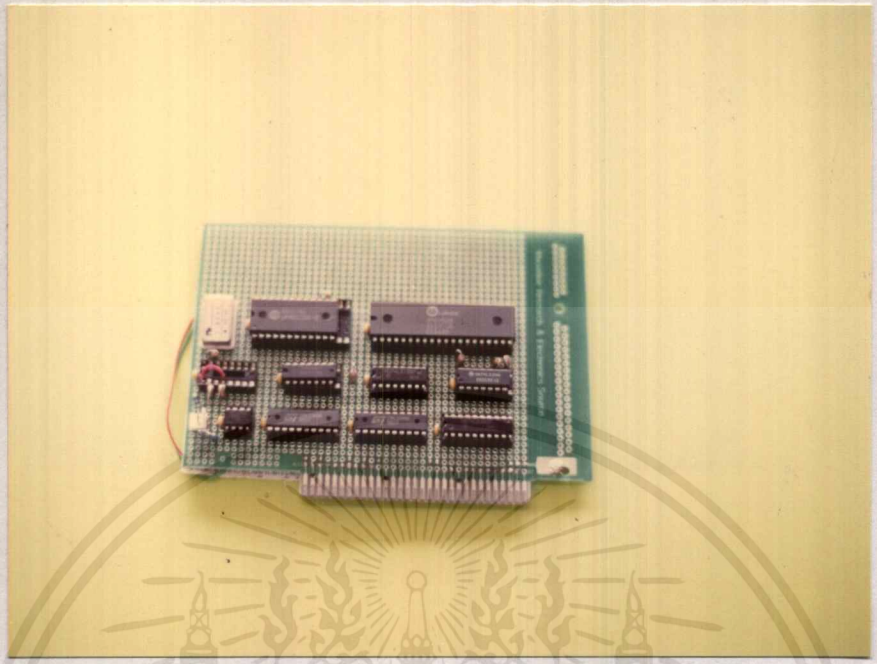
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 13 แสดง block diagram interface card



รูปที่ 14 แสดงวงจรส่วนรับส่งข้อมูล และส่วนสร้างสัญญาณอ้างอิง



รูปที่ 15 แสดงชิ้นงาน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 4

ผลการทดลอง

การ์ดอินเฟสเตอร์ที่สร้างขึ้นมา เป็นการ์ดรับส่งข้อมูลระหว่างคอมพิวเตอร์แบบฮาล์ฟดูเพล็กซ์ สามารถส่งได้ในอัตราบิตสูงถึง 56 Kbps

ในการทดลองนี้ทำการทดลองส่งข้อมูลระหว่างเครื่องคอมพิวเตอร์รุ่น 80386 กับ 8088 ใช้อัตราบิตแค่เพียง 19.2 Kbps อันเป็นผลเนื่องมาจากขีดจำกัดของตัวคอมพิวเตอร์ ซึ่งถ้าหากว่าใช้อัตราบิตมากกว่านี้ การรับส่งข้อมูลไม่สามารถกระทำได้ 8088 จะทำงานไม่ทัน และส่งผ่านสายโทรศัพท์ได้ในระยะทาง 100 m ได้ถูกต้องสมบูรณ์

เนื่องจากว่าการ์ดอินเตอร์เฟสที่สร้างขึ้นมา จะต้องมีโปรแกรมควบคุมการทำงาน ดังนั้นผลการทดลองจะกล่าวควบคู่ไปกับโปรแกรม

ไฟล์ ACE เป็นไฟล์ที่รวมฟังชันควบคุมการทำงาน ของส่วนรับส่งข้อมูล(8250)

ไฟล์ 54 เป็นไฟล์เป็นที่รวมฟังชันควบคุมการทำงาน ของส่วนสร้างสัญญาณอ้างอิง(8254)

ไฟล์ 59 เป็นไฟล์ที่รวมฟังชันควบคุมการทำงาน ของตัวควบคุมการอินเตอร์รัพท์(8259)

โปรแกรม 1T และ 1R ใช้ในการทดสอบการ์ดอินเตอร์เฟส โดยเป็นการรับ-ส่งในลักษณะซิมเพล็กซ์ ส่งข้อมูลทีละ 1 ไบต์ สามารถรับส่งได้ถูกต้อง

โปรแกรม TRX เป็นโปรแกรมรับส่งข้อมูลแบบฮาล์ฟดูเพล็กซ์ ส่งข้อมูลเป็นแพกเก็ต สามารถที่จะรับส่งข้อมูลแบบ real time ได้

โปรแกรม TRX2 เป็นโปรแกรมที่พัฒนาขึ้นมาจาก TRX โดยมีส่วนที่เพิ่มขึ้นมา คือ มีการระบุแอดเดรสได้ว่าส่งจากเครื่องใดไปยังเครื่องใด ต้องการส่งแบบกระจายข่าว(broadcast)คือทุกเครื่องรับข้อมูลได้ หรือต้องการส่งให้เครื่องใดเครื่องหนึ่งเท่านั้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โปรแกรม TRX3 เป็นโปรแกรมที่พัฒนาขึ้นมาจาก TRX2 โดยมีส่วนที่เพิ่มขึ้นมาคือ เป็นการรับส่งข้อมูลแบบ CSMA คือจะมีการตรวจสถานะของ network bus ก่อนที่จะส่ง

โปรแกรม TRX4 เป็นโปรแกรมที่พัฒนาขึ้นมาจาก TRX3 โดยมีส่วนที่เพิ่มขึ้นมาคือ สามารถที่จะส่งไฟล์ได้(file transfer) โดยส่งไฟล์ข้อมูล(text file) ได้ถูกต้องสมบูรณ์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 5

สรุป

การ์ดรับส่งข้อมูลที่สร้างขึ้นมา สร้างเสร็จไปเพียง 2ส่วน ได้แก่ ส่วนรับส่งข้อมูล และส่วนสร้างสัญญาณอ้างอิง สามารถทำการรับส่งข้อมูลได้ผลเป็นที่น่าพอใจ และเมื่อใช้กับโปรแกรมที่พัฒนาขึ้นมา ก็สามารถทำการส่งไฟล์ข้อมูลได้

แนวทางการพัฒนาต่อไป ได้แก่การสร้างฮาร์ดแวร์เพิ่มซึ่งได้แก่ส่วนตรวจสอบการชนกันของข้อมูล และการพัฒนาโปรแกรมต่อจนถึงระดับ server

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
 * lancon.h - lan protocol constants definition file */
/*******************************************************/

/* special characters */
#define     PRE             0X00
#define     ETX             0X03
#define     CR              0X0D
#define     LF              0X0A
#define     SPACE           0X20
#define     ESC             0X1B
#define     RX              0X12
#define     TX              0X14
#define     FT              0X01
#define     EOT1            0X11
#define     ACK             0X06
```

```
/* lanvar.h - lan protocol variables definition file */
/*******************************************************/

char rbuf[2005],
     tbuf[2005];

char sa, da ;

char nbyte, tbyte,

int i,ii,wait;

int byi, byo, byw,
    byt, byr, bys,
    ch, ac;

int state;

int rbf,tbf;

char *tborg, *rborg,
     *tbnxt, *rbnxt,
     *tbend, *rbend;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
/* ace.h the header file for ACE related functions */
/*************************************************/

#define     ACEBASE     0X300

#define     RBR         ACEBASE
#define     THR         ACEBASE

#define     IER         ACEBASE + 1
#define     IIR         ACEBASE + 2

#define     DLL         ACEBASE
#define     DLM         ACEBASE + 1

#define     LCR         ACEBASE + 3
#define     MCR         ACEBASE + 4
#define     LSR         ACEBASE + 5

#define     MODE        0X03
#define     DIVISOR     0X83

#define     RRDY        0X01
#define     TRDY        0X20

#define     ENBRI       0X01
#define     ENBTI       0X02
#define     DISI        0X00

#define     BD24L       06
#define     BD24M       00

#define     ERROR       0X0A
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
/* ace.c - 8250 service file */
/*****************************/

#include "ace.h"

init_ace()
{
outportb(LCR, DIVISOR);
outportb(DLL, BD24L);
outportb(DLM, BD24M);
outportb(LCR, MODE);
outportb(IER, DISI);
}

set_ace_irqs(cmd)
char cmd;
{
outportb(IER, cmd);
}

getc_ace()
{
int data;
while((inportb(LSR) & RRDY) == 0)
      ;
data = inportb(RBR);
return(data);
}

putc_ace(data)
char data;
{
outportb( MCR, 0X04);
while((inportb(LSR) & TRDY) == 0)
      ;
outportb(THR, data);
delay();
outportb( MCR, 0X00);
}

error_ace()
{
int code;
if((inportb(LSR) & ERROR) == 0) code = 0;
else code = 1;
return (code);
}


delay()
{
int a;
for( a = 500; a > 0; --a)
   ;
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
 /* 54.h - the header file for 8254 related functions */
/*****************************************************/

#define      BASE          0X308

#define      CNTCTRL       BASE + 3
#define      COUNT0        BASE
#define      COUNT1        BASE + 1

#define      CNT0MODE      0X34
#define      CNT1MODE      0X70

#define      CNT0LO        18
#define      CNT0HI        00
```

```
/* 54.c - the 8254 service file */
/********************************/

#include "54.h"

start_cnt0()
{
outportb(CNTCTRL, CNT0MODE);
outportb(COUNT0, CNT0LO);
outportb(COUNT0, CNT0HI);
}

start_cnt1(low,high)
int low,high;
{
outportb(CNTCTRL, CNT1MODE);
outportb(COUNT1, low);
outportb(COUNT1, high);
}

stop_cnt1()
{
outportb(CNTCTRL, CNT1MODE);
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
/* 59.h - header file for 8259 interrupt priority controller */
/*************************************************************/

#define     BASE59      0X20
#define     IMR         BASE59 + 1

#define     IRQ3        0X0B
#define     IRQ5        0X0D
#define     IRQ7        0X0F

#define     ENB3        0XF7
#define     ENB5        0XDF
#define     ENB7        0X7F

#define     DIS3        0X08
#define     DIS5        0X20
#define     DIS7        0X80

#define     EOI         0X20
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
/* 59.C - 8259 interrupt controller service file */
/***********************************************/

#include "59.h"

enable_int3()
{
char mask;
mask = inportb(IMR);
mask &= ENB3;
outportb(IMR, mask);
}

disable_int3()
{
char mask;
mask = inportb(IMR);
mask |= DIS3;
outportb(IMR, mask);
}

enable_int5()
{
char mask;
mask = inportb(IMR);
mask &= ENB5;
outportb(IMR, mask);
}

disable_int5()
{
char mask;
mask = inportb(IMR);
mask |= DIS5;
outportb(IMR, mask);
}

enable_int7()
{
char mask;
mask = inportb(IMR);
mask &= ENB7;
outportb(IMR, mask);
}

disable_int7()
{
char mask;
mask = inportb(IMR);
mask |= DIS7;
outportb(IMR, mask);
}

clr_ints()
{
outportb(BASE59, EOI);
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
/* lab1t.c - program for the lab 1 transmitter node */
/*************************************************/

#include      "LANCON.H"
#include      "LANVAR.H"
#include      "ACE.C"

main()
{
init_ace();

nbyte = 0;
while (nbyte != ESC)
     {
     if (kbhit())
        {

        nbyte = getch();
        putch(nbyte);

        putc_ace(nbyte);

        if (nbyte == CR)
           {
           putch(LF);

           putc_ace(LF);
           }
        }
     }
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
/* lab1r.c - program for the lab1 receiver node */
/*********************************************/

#include "LANCON.H"
#include "LANVAR.H"
#include "ACE.C"
#include "59.C"

void setup_irq3();
void restore_irq3();

void interrupt (*OLDIRQ3)();

main()
{
init_ace();
outportb( MCR, 0X00);

set_ace_irqs(ENBRI);
setup_irq3();
clr_ints();
enable_int3();

nbyte = 0;
while (nbyte != ESC)
  ;
disable_int3();
restore_irq3();
}

void interrupt nisr()
{
nbyte = getc_ace();
putch(nbyte);
clr_ints();
}

void setup_irq3()
{
OLDIRQ3 = (void interrupt (*)() )getvect(IRQ3);
setvect(IRQ3, nisr);
}

void restore_irq3()
{
setvect(IRQ3, OLDIRQ3);
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
/*TRX  new generation program for transeiver */
/*********************************************/

#include "LANCON.H"
#include "LANVAR.H"
#include "ACE.C"
#include "59.C"
#include <stdio.h>

void setup_irq3r();

void interrupt risr();

void restore_irq3();

void interrupt (*OLDIRQ3)();

main()
{
int done = 1;

clrscr();
clear_flags();
initialize();

while (done != 0)
  {
     if(ch == 1)
       rmode();
     else if(ch == 2)
       tmode();
     else if(ch == 3)
       done = 0;
  }
}

clear_flags()
{
set_ace_irqs(DISI);
outportb( MCR, 0X00);
bys = byr = 0;
byi = byw = 0;
byo = ac = 0;
ch = 1;
}

initialize()
{
rborg = rbuf;
rbnxt = rborg;

tborg = tbuf;
tbnxt = tborg+1;
*tborg = PRE;

init_ace();
}

restore()
{
disable_int3();
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
restore_irq3();
}

tmode()
{
int out;
  clrscr();
  printf("\n\n\n*************************************\n\007");
  printf("            Transmit Mode\n");

  bys = 1;

  out = input();
   while( bys != 0)
    transmit();
  if(out == 3)
    ch = 3;
  else  ch = 1;
}

input()
{
int c = 0;
  printf("            Input Message\n");
  printf("    Press Ctrl_Q to Send Message\n");
  printf("       Exit Program Press ESC\n");
  printf("*************************************\n\n");
tbyte = 0;

while((tbyte != EOT1) & (tbyte != ESC) & (tbyte != RX))
   {
   if(kbhit())
    {
    tbyte = getch();
    putch( tbyte);
    *tbnxt++ = tbyte;
    if ( tbyte == CR)
       {
       tbyte = LF;
       putch( tbyte);
       *tbnxt++ = tbyte;
       }

    }
   }

if(tbyte == ESC)
  {
  bys = 0;
  c = 3;
  }
else if(tbyte == RX)
  {
  bys = 0;
  c = 1;
  }

tbend = tbnxt;
tbnxt = tborg;
return(c);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
}

transmit()
{
if ( tbnxt == tbend)
   {
   bys = 0;
   tbnxt = tborg + 1;
   }
else
   putc_ace(*tbnxt++);
}

rmode()
{
int rm = 0;

setup_irq3r();
clr_ints();
enable_int3();

  printf("\n\n\n**********************************\n");
  printf("           Receive Mode\n");
  printf("   To Transmit Mode Press Ctrl-T\n");
  printf("      Exit Program Press ESC\n");
  printf("**********************************\n\n");

  byo = 0;
  byw = 1;
  set_ace_irqs(ENBRI);

  while (byw != 0)
    {
    if(kbhit())
    byw = 0;
    }
 output();

 if(ac == 0)
   {
    rm = getch();
    if(rm == TX)
     ch = 2;
    else if(rm == ESC)
     ch = 3;
   }
 else
   {
   ch = 1;
   ac = 0;
   }
restore();
}

output()
{
 if(byo == 1)
   {
   printf("Receive Data \n\n");
   while ( rbnxt < rbend)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
      putch( *rbnxt++);
    rbnxt = rborg;
    printf("\n\n");
    }
 byo = 0;
}

void setup_irq3r()
{
OLDIRQ3 = (void interrupt (*)() )getvect(IRQ3);
setvect(IRQ3, risr);
}

void interrupt risr()
{
receive();
clr_ints();
}

receive()
{
nbyte = getc_ace();
if((byr == 0) && (nbyte == PRE))
  byr = 1;
else if(byr == 1)
       {
       if(nbyte == EOT1)
         {
         ac = 1;
         byo = 1;
         byw = 0;
         byr = 0;
         rbend = rbnxt;
         rbnxt = rborg;
         set_ace_irqs(DISI);
         }
       else
         *rbnxt++ = nbyte;
       }
}

void restore_irq3()
{
setvect(IRQ3, OLDIRQ3);
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
/*TRX2  address - framed program for transeiver */
/***********************************************/

#include "LANCON.H"
#include "LANVAR.H"
#include "ACE.C"
#include "59.C"
#include <stdio.h>

void setup_irq3r();

void interrupt risr();

void restore_irq3();

void interrupt (*OLDIRQ3)();

main()
{
int done = 1;
sa = '3';

clrscr();
clear_flags();
initialize();

while (done != 0)
  {
     if(ch == 1)
       rmode();
     else if(ch == 2)
       tmode();
     else if(ch == 3)
       done = 0;
  }
}

clear_flags()
{
set_ace_irqs(DISI);
outportb( MCR, 0X00);
bys = byr = 0;
byi = byw = 0;
byo = ac = 0;
state = i = ii = 0;
ch = 1;
}

initialize()
{
rborg = rbuf;
rbnxt = rborg;

tborg = tbuf;
tbnxt = tborg + 3;
tborg[0] = PRE;
tborg[2] = sa;

init_ace();
}
```

เอกสารนี้สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
restore()
{
disable_int3();
restore_irq3();
}

tmode()
{
int out;
  clrscr();
  printf("\n\n\n**************************************\n\a");
  printf("            Transmit Mode\n");
  printf("       Exit Program Press ESC\n");
  printf("**************************************\n");

  bys = 1;

  out = input();
   while( bys != 0)
    transmit();
  if(out == 3)
    ch = 3;
  else  ch = 1;
}

input()
{
int c = 0;
input_da();

  printf("\nInput Message\n");
  printf("Press Ctrl_Q to Send Message\n\n");
tbyte = 0;

while((tbyte != EOT1) & (tbyte != ESC) & (tbyte != RX))
   {
   if(kbhit())
    {
    tbyte = getch();
    putch( tbyte);
    *tbnxt++ = tbyte;
    if ( tbyte == CR)
       {
       tbyte = LF;
       putch( tbyte);
       *tbnxt++ = tbyte;
       }

    }
   }

if(tbyte == ESC)
  {
  bys = 0;
  c = 3;
  }
else if(tbyte == RX)
  {
  bys = 0;
  c = 1;
  }
```

เอกสารนี้สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
tbend = tbnxt;
tbnxt = tborg;

return(c);
}

input_da()
{
char dd;
printf("This Terminal is 386\n");
printf("Input Destination Address: ");
dd = getch();
da = toupper(dd);
putch(da);
while((dd = getch()) != CR)
  putch(toupper(dd));
putch(LF);
putch(CR);
tborg[1] = da;
}

transmit()
{
if ( tbnxt == tbend)
   {
   bys = 0;
   tbnxt = tborg + 3;
   }
else
   putc_ace(*tbnxt++);
}

rmode()
{
int rm = 0;

setup_irq3r();
clr_ints();
enable_int3();

  printf("\n\n\n************************************\n");
  printf("           Receive Mode\n");
  printf("   To Transmit Mode Press Ctrl-T\n");
  printf("      Exit Program Press ESC\n");
  printf("************************************\n\n");

  byo = 0;
  byw = 1;
  set_ace_irqs(ENBRI);

  while (byw != 0)
    {
    if(kbhit())
    byw = 0;
    }
  chk_add();
  find();
  output();
```

เอกสารนี้มีลิขสิทธิ์การที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
 if(ac == 0)
   {
    rm = getch();
    if(rm == TX)
     ch = 2;
    else if(rm == ESC)
     ch = 3;
   }
 else
   {
   ch = 1;
   ac = 0;
   }
restore();
}

output()
{
 if(byo == 1)
   {
   while ( rbnxt < rbend)
     putch( *rbnxt++);
   rbnxt = rborg;
   printf("\n\n");
   }
 byo = 0;
 i = 0;
}

void setup_irq3r()
{
OLDIRQ3 = (void interrupt (*)() )getvect(IRQ3);
setvect(IRQ3, risr);
}

void interrupt risr()
{
receive();
clr_ints();
}

receive()
{
nbyte = getc_ace();
if((byr == 0) && (nbyte == PRE))
  byr = 1;
else if(byr == 1)
 {
 if(nbyte == EOT1)
   {
   ii = 1;
   ac = 1;
   byw = 0;
   state = 0;
   rbend = rbnxt;
   set_ace_irqs(DISI);
   }
 else
   *rbnxt++ = nbyte;
 }
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
chk_add()
{
if(ii == 1)
  {
  while(rborg[i] == PRE)
    i=i+1;
  if((rborg[i] == sa) || (rborg[i] == 'A'))
    {
    state = 1;
    byo = 1;

    rbnxt = rborg +i+2;
    }
  else rbnxt = rborg;
  }
ii = 0;
}


find()
{
if(state == 1)
  {
  if((rborg[i] == 'A') && (rborg[i+1] == '8'))
    printf("Receive Broadcast Data from 8088\n\n\a");
  else if(rborg[i+1] == '8')
    printf("Receive Data from 8088\n\n\a");
  }
state = 0;
}

void restore_irq3()
{
setvect(IRQ3, OLDIRQ3);
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
/*TRX3  CSMA program for transeiver */
/***********************************/

#include "LANCON.H"
#include "LANVAR.H"
#include "ACE.C"
#include "59.C"
#include "54.C"
#include <stdio.h>

void setup_irq3r(),setup_irq5();

void interrupt risr(),interrupt timer();

void restore_irq3(),restore_irq5();

void interrupt (*OLDIRQ3)(),interrupt (*OLDIRQ5)();

main()
{
int done = 1;
sa = '3';

clrscr();
clear_flags();
initialize();

while (done != 0)
  {
     if(ch == 1)
       rmode();
     else if(ch == 2)
       tmode();
     else if(ch == 3)
       done = 0;
  }
}

clear_flags()
{
set_ace_irqs(DISI);
outportb( MCR, 0X00);
bys = byr = 0;
byi = byw = 0;
byo = ac = 0;
state = i = ii = 0;
wait = 0;
ch = 1;
}

initialize()
{
rborg = rbuf;
rbnxt = rborg;

tborg = tbuf;
tbnxt = tborg + 3;
tborg[0] = PRE;
tborg[2] = sa;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
init_ace();
}

restore()
{
disable_int3();
restore_irq3();
}

tmode()
{
int out;
  clrscr();
  printf("\n\n\n*************************************\n\a");
  printf("            Transmit Mode\n");
  printf("       Exit Program Press ESC\n");
  printf("*************************************\n");

  bys = 1;
  out = input();

setup_irq5();
clr_ints();
enable_int5();

start_cnt0();
start_cnt1(11,00);

  wait = 1;
while(wait == 1)
  {
  if(!getc_ace())
    ;
  else
    wait = 0;
  }
disable_int5();
restore_irq5();

  if(out == 3)
    ch = 3;
  else  ch = 1;
}

input()
{
int c = 0;
input_da();

  printf("\nInput Message\n");
  printf("Press Ctrl_Q to Send Message\n\n");
tbyte = 0;

while((tbyte != EOT1) & (tbyte != ESC) & (tbyte != RX))
   {
   if(kbhit())
    {
    tbyte = getch();
    putch( tbyte);
    *tbnxt++ = tbyte;
    if ( tbyte == CR)
       {
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        tbyte = LF;
        putch( tbyte);
        *tbnxt++ = tbyte;
        }

     }
    }

if(tbyte == ESC)
  {
  bys = 0;
  c = 3;
  }
else if(tbyte == RX)
  {
  bys = 0;
  c = 1;
  }

tbend = tbnxt;
tbnxt = tborg;

return(c);
}

input_da()
{
char dd;
printf("This Terminal is 386\n");
printf("Input Destination Address: ");
dd = getch();
da = toupper(dd);
putch(da);
while((dd = getch()) != CR)
  putch(toupper(dd));
putch(LF);
putch(CR);
tborg[1] = da;
}

transmit()
{
if ( tbnxt == tbend)
   {
   bys = 0;
   tbnxt = tborg + 3;
   }
else
   putc_ace(*tbnxt++);
}

void setup_irq5()
{
OLDIRQ3 = (void interrupt (*)() )getvect(IRQ5);
setvect(IRQ5, timer);
}

void interrupt timer()
{
if(wait == 1)
  {
  while( bys != 0)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น ทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    transmit();
  clr_ints();
  wait = 0;
  }

}

void restore_irq5()
{
setvect(IRQ5, OLDIRQ5);
}

rmode()
{
int rm = 0;

setup_irq3r();
clr_ints();
enable_int3();

  printf("\n\n\n************************************\n");
  printf("          Receive Mode\n");
  printf("   To Transmit Mode Press Ctrl-T\n");
  printf("      Exit Program Press ESC\n");
  printf("************************************\n\n");

  byo = 0;
  byw = 1;
  set_ace_irqs(ENBRI);

  while (byw != 0)
    {
    if(kbhit())
    byw = 0;
    }
  chk_add();
  find();

  output();

 if(ac == 0)
   {
    rm = getch();
    if(rm == TX)
     ch = 2;
    else if(rm == ESC)
     ch = 3;
   }
 else
   {
   ch = 1;
   ac = 0;
   }
restore();
}

output()
{
 if(byo == 1)
   {
   while ( rbnxt < rbend)
     putch( *rbnxt++);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    rbnxt = rborg;
    printf("\n\n");
    }
 byo = 0;
 i = 0;
}

void setup_irq3r()
{
OLDIRQ3 = (void interrupt (*)() )getvect(IRQ3);
setvect(IRQ3, risr);
}

void interrupt risr()
{
receive();
clr_ints();
}

receive()
{
nbyte = getc_ace();
if((byr == 0) && (nbyte == PRE))
  byr = 1;
else if(byr == 1)
 {
 if(nbyte == EOT1)
   {
   ii = 1;
   ac = 1;
   byw = 0;
   state = 0;
   rbend = rbnxt;
   set_ace_irqs(DISI);
   }
 else
   *rbnxt++ = nbyte;
 }
}

chk_add()
{
if(ii == 1)
  {
  while(rborg[i] == 0x00)
    i=i+1;
  if((rborg[i] == sa) || (rborg[i] == 'A'))
    {
    state = 1;
    byo = 1;

    rbnxt = rborg +i+2;
    }
  else rbnxt = rborg;
  }
ii = 0;
}

find()
{
if(state == 1)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  {
  if((rborg[i] == 'A') && (rborg[i+1] == '8'))
    printf("Receive Broadcast Data from 8088\n\n\a");
  else if(rborg[i+1] == '8')
    printf("Receive Data from 8088\n\n\a");
  }
state = 0;
}

void restore_irq3()
{
setvect(IRQ3, OLDIRQ3);
}
```


เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
/*TRX4  File Transfered program for transeiver */
/*********************************************/

#include "LANCON.H"
#include "LANVAR.H"
#include "ACE.C"
#include "59.C"
#include <stdio.h>
#include <sys\stat.h>

void setup_irq3r(),setup_irq3r2(),restore_irq3();

void interrupt risr(),interrupt risr2();

void interrupt (*OLDIRQ3)();

main()
{
int done = 1;
sa = '3';

clrscr();
clear_flags();
initialize();

while (done != 0)
  {
     if(ch == 1)
       rmode();
     else if(ch == 2)
       tmode();
     else if(ch == 3)
       done = 0;
     else if(ch == 4)
       file();
  }
}

clear_flags()
{
set_ace_irqs(DISI);
outportb( MCR, 0X00);
bys = byr = 0;
byi = byw = 0;
byo = ac = 0;
state = i = 0;
ii = 0;
ch = 1;
}

initialize()
{
rborg = rbuf;
rbnxt = rborg;

tborg = tbuf;
tbnxt = tborg + 3;
tborg[0] = PRE;
tborg[2] = sa;

init_ace();
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
}

restore()
{
disable_int3();
restore_irq3();
}

tmode()
{
int out;
  clrscr();
  printf("\n\n\n**************************************\n\a");
  printf("            Transmit Mode\n");
  printf("       Exit Program Press ESC\n");
  printf("**************************************\n");

  bys = 1;

  out = input();
   while( bys != 0)
    transmit();
  if(out == 3)
    ch = 3;
  else if(out == 4)
    ch = 4;
  else  ch = 1;
}

input()
{
int c = 0;
input_da();

  printf("\nInput Message\n");
  printf("Press Ctrl_Q to Send Message\n\n");
tbyte = 0;

while((tbyte != EOT1) & (tbyte != ESC) & (tbyte != RX)
       & (tbyte != FT))
   {
   if(kbhit())
    {
    tbyte = getch();
    putch( tbyte);
    *tbnxt++ = tbyte;
    if ( tbyte == CR)
       {
       tbyte = LF;
       putch( tbyte);
       *tbnxt++ = tbyte;
       }

    }
   }

if(tbyte == ESC)
  {
  bys = 0;
  c = 3;
  }
else if(tbyte == RX)
  {
  bys = 0;
  c = 1;
  }
```

เอกสารนี้จัดทำขึ้นเพื่อสงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
else if(tbyte == FT)
  {
  bys = 0;
  c = 4;
  }

tbend = tbnxt;
tbnxt = tborg;

return(c);
}

input_da()
{
char dd;
printf("\nThis Terminal is 386\n");
printf("Input Destination Address: ");
dd = getch();
da = toupper(dd);
putch(da);
while((dd = getch()) != CR)
  putch(toupper(dd));
putch(LF);
putch(CR);
tborg[1] = da;
}

transmit()
{
if ( tbnxt == tbend)
   {
   bys = 0;
   tbnxt = tborg + 3;
   }
else
   putc_ace(*tbnxt++);
}

rmode()
{
int rm = 0;

setup_irq3r();
clr_ints();
enable_int3();

  printf("\n\n\n************************************\n");
  printf("           Receive Mode\n");
  printf("   To Transmit Mode Press Ctrl-T\n");
  printf("      Exit Program Press ESC\n");
  printf("************************************\n\n");

  byo = 0;
  byw = 1;
  set_ace_irqs(ENBRI);

  while (byw != 0)
    {
    if(kbhit())
    byw = 0;
    }
  chk_add();
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  find();

  output();

 if(ac == 0)
 . {
    rm = getch();
    if(rm == TX)
     ch = 2;
    else if(rm == ESC)
     ch = 3;
    else if(rm == FT)
     ch = 4;
   }
 else
   {
   ch = 1;
   ac = 0;
   }
restore();
}

output()
{
 if(byo == 1)
   {
   while ( rbnxt < rbend)
     putch( *rbnxt++);
   rbnxt = rborg;
   printf("\n\n");
   }
 byo = 0;
 i = 0;
}

void setup_irq3r()
{
OLDIRQ3 = (void interrupt (*)() )getvect(IRQ3);
setvect(IRQ3, risr);
}

void interrupt risr()
{
receive();
clr_ints();
}

receive()
{
nbyte = getc_ace();
if((byr == 0) && (nbyte == PRE))
  byr = 1;
else if(byr == 1)
 {
 if(nbyte == EOT1)
   {
   ii = 1;
   ac = 1;
   byw = 0;
   state = 0;
   rbend = rbnxt;
   set_ace_irqs(DISI);
   }
 else
   *rbnxt++ = nbyte;
```

เอกสารนี้จัดทำขึ้นเพื่อการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
 }
}

chk_add()
{
if(ii == 1)
  {
  while(rborg[i] == 0x00)
    i=i+1;
  if((rborg[i] == sa) || (rborg[i] == 'A'))
    {
    state = 1;
    byo = 1;

    rbnxt = rborg +i+2;
    }
  else rbnxt = rborg;
  }
ii = 0;
}

find()
{
if(state == 1)
  {
  if((rborg[i] == 'A') && (rborg[i+1] == '8'))
    printf("Receive Broadcast Data from 8088\n\n\a");
  else if(rborg[i+1] == '8')
    printf("Receive Data from 8088\n\n\a");
  }
state = 0;
}

void restore_irq3()
{
setvect(IRQ3, OLDIRQ3);
}


file(name)
char *name;
{
char c;

clrscr();
printf("File Transfer Mode\a\a\n\n");
printf("Send File Press 1 ,or Receive File Press 2\n");
printf("Select :    ");
c = select();


if(c == '1')
  s_file(name);
else if(c == '2')
  r_file();

ch = 1;
}

select()
{
char c;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
char d = 0;
while(d == 0)
  {
  c = getch();

  if(c == '1')
    {
    d = 1;
    putch(c);
    }
  else if(c == '2')
    {
    d = 1;
    putch(c);
    }
  else printf("\nSelect again :    ");
  }
return(c);
}

s_file(fname)
char *fname;
{
FILE *fp;

struct stat ss;
long c;
long bb = 0;

input_da();

printf("\nEnter File Name:    ");
gets(fname);

if((fp = fopen(fname, "rb")) == 0 )
  {
  printf("\nCannot Open Input File\n");
  exit(1);
  }
s_file_name(fname);

wait_a();

stat(fname,&ss);
c = ss.st_size;

while(bb < c)
  {
  int a = 0;
  ftell(fp);
  while((a++ < 2000) && (bb < c))
    {
    bb++;
    *tbnxt++ = getc(fp);
    }
    *tbnxt++ = ETX;

    tbend = tbnxt;
    tbnxt = tborg;
    bys = 1;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    while(bys != 0)
      transmit();

    wait_a();

  }

fclose(fp);
}

s_file_name(f)
char *f;
{
printf("\nSending File Name...");
while(*f)
  *tbnxt++ = *f++;
  *tbnxt++ = ETX;

tbend = tbnxt;
tbnxt = tborg;

bys = 1;
while(bys != 0)
  transmit();
}

wait_a()
{
printf("wait..");
get_data();

i = 0;
while(rborg[i] == 0x00)
  i=i+1;
if(rborg[i] == sa)
  {
  if(rborg[i+2] == ACK)
     printf("OK   ");
  else
     {
     printf("Error\n");
     exit(1);
     }
  }
else
  {
  printf("Error\n");
  exit(1);
  }

}

r_file()
{
FILE *fp;

char fn[20];
int n;
int c = 0;
  g_file_name();
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  while((fn[c++] = *rbnxt++) != 0x00);

  printf("%s",fn);

if((fp = fopen(fn, "wb")) == 0 )
  {
  printf("Cannot Open Output File\n");
  exit(1);
  }

  ack();

do{
  get_data();
  get_data2();

  ftell(fp);
  n =(int)(rbend - rbnxt);
  printf("%d",n);
  fwrite(rbnxt,1,n,fp);

  ack();
  }while((int)(rbend - rbnxt) >= 2000);

fclose(fp);
}

g_file_name()
{
printf("\nReceiving File Name...");

get_data();
get_data2();

}


ack()
{
printf("\nack\n");

tborg[1] = rborg[i+1];
*tbnxt++ = ACK;
*tbnxt++ = ETX;

tbend = tbnxt;
tbnxt = tborg;

bys = 1;
while(bys != 0)
  transmit();
rbnxt = rborg;
}


get_data()
{
setup_irq3r2();
clr_ints();
enable_int3();

  byw = 1;
```

เอกสารนี้สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  set_ace_irqs(ENBRI);

  while (byw != 0)
    {
    if(kbhit())
    exit(1);
    }

restore();
}

get_data2()
{
i = 0;
while(rborg[i] == 0x00)
  i++;
if(rborg[i] == sa)
  rbnxt = rborg +i+2;
else exit(1);
}


delay2(t)
int *t;
{
for(; t > 0; t--)
  ;
}

void setup_irq3r2()
{
OLDIRQ3 = (void interrupt (*)() )getvect(IRQ3);
setvect(IRQ3, risr2);
}

void interrupt risr2()
{
receive2();
clr_ints();
}

receive2()
{
nbyte = getc_ace();
if((byr == 0) && (nbyte == PRE))
  byr = 1;
else if(byr == 1)
 {
 if(nbyte == ETX)
   {
   byw = 0;
   rbend = rbnxt;
   set_ace_irqs(DISI);
   }
 else
   *rbnxt++ = nbyte;
 }
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กิตติกรรมประกาศ

วิทยานิพนธ์นี้สำเร็จลุล่วงได้ด้วยดี โดยได้รับคำแนะนำและช่วยเหลือจาก อาจารย์อิทธิชัย อรุณศรีแสงชัย และพี่ๆปริญญาโท กระผมขอขอบคุณทุกท่านเป็นอย่างสูง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หนังสืออ้างอิง

1.LESZEK REISS, INTRODUCTION TO LOCAL AREA NETWORKS WITH MICROCOMPUTER EXPERIMENTS, PRENTICE HALL, 1987

2.GERD E.KEISER, LOCAL AREA NETWORKS, MCGRAW HILL, 1989

3.น.ต. ด.ร.ไพศาล สงวนหมู่, ร.ศ. ยืน ภู่วรวรรณ, การสื่อสารข้อมูลและไมโครคอมพิวเตอร์เนตเวอกค์, สำนักพิมพ์ซีเอ็ด, 2531

4.JOE CAMPBELL, C PROGRAMMER'S GUIDE TO SERIAL COMMUNICATION, HOWARD W.SAMS & COMPANY, 1987

5.TEXAS INSTRUMENTS ENGINEERING STAFF, LINEAR AND INTERFACE CIRCUITS APPLICATIONS, SECOND EDITION, MCGRAW HILL, 1988

6.ศิวัฒน์ ศิวะบวร, พรชัย จักรช่ารงค์, จิรศักดิ์ ชัยวิริยะกุล, การประยุกต์ใช้งานภาษาซี, สำนักพิมพ์ซีเอ็ด, 2535

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้