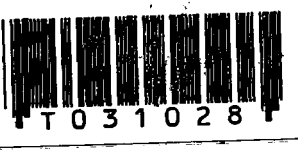
การกระจายเทอร์มินัลผ่านคอมพิวเตอร์ส่วนบุคคล

UNIX Terminals Server using PC

นายอภิรมย์ แซ่เล้า
MR.APIROM SAELAO



วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต
สาขาวิชาวิทยาการคอมพิวเตอร์และเทคโนโลยีสารสนเทศ
บัณฑิตวิทยาลัย
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
พ.ศ. 2541

ISBN 974-622-192-2

UNIX Terminals Server using PC

APIROM SAELAO

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE
MASTER OF SCIENCE IN COMPUTER SCIENCE AND
INFORMATION TECHNOLOGY
SCHOOL OF GRADUATE STUDIES
KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG
1998

ISBN 974-622-192-2

หัวข้อวิทยานิพนธ์	การกระจายเทอร์มินัลผ่านคอมพิวเตอร์ส่วนบุคคล
นักศึกษา	นายอภิรมย์ แซ่เล้า
อาจารย์ผู้ควบคุมวิทยานิพนธ์	อาจารย์ กวิน สนธิเพิ่มพูน
ระดับการศึกษา	วิทยาศาสตรมหาบัณฑิต สาขาวิชาวิทยาการคอมพิวเตอร์และเทคโนโลยีสารสนเทศ
ภาควิชา	คณิตศาสตร์และวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
พ.ศ.	2541

บทคัดย่อ

การต่อเทอร์มินัลในระบบเน็ตเวิร์คมี 3 วิธีใหญ่ๆ ได้แก่ การต่อผ่านพอร์ตอนุกรมบนโฮส, การต่อเป็นโหนดๆหนึ่งในเน็ตเวิร์ค และการต่อผ่านเทอร์มินัลเซิฟเวอร์ โดย 2 วิธีแรกมีข้อจำกัดอยู่ที่จำนวนพอร์ตอนุกรมบนโฮสและจำนวนโหนดในเน็ตเวิร์ค ส่วนการต่อผ่านเทอร์มินัลเซิฟเวอร์นั้นสามารถแก้ปัญหาข้างต้นได้ แต่มีราคาค่อนข้างสูง

วิทยานิพนธ์นี้ จะอธิบายการพัฒนาโปรแกรมบริการเทอร์มินัลระบบยูนิกซ์บนเครื่องคอมพิวเตอร์ส่วนบุคคล โดยการนำคอมพิวเตอร์ส่วนบุคคลทั่วไปมาใช้เป็นเทอร์มินัลเซิฟเวอร์ หมายถึง คอมพิวเตอร์ตัวนั้น จะสามารถให้บริการคอมพิวเตอร์ตัวอื่นๆที่มาต่อพอร์ตอนุกรมให้เป็นจอเทอร์มินัลในระบบเครือข่าย เครือข่ายที่ใช้ในงานวิจัยเป็นเครือข่ายแบบอีเทอร์เน็ต อาศัยรูปแบบแพ็คเก็ตของโปรโตคอล TCP/ IP โดยใช้หมายเลขพอร์ต TCP ในการจำแนกการติดต่อกันระหว่างโฮสกับพอร์ตอนุกรมของเทอร์มินัลเซิฟเวอร์

โปรแกรมบริการเทอร์มินัลระบบยูนิกซ์บนเครื่องคอมพิวเตอร์ส่วนบุคคลนี้ พัฒนาภายใต้ระบบปฏิบัติการ DOS และใช้แพ็คเก็ตไดร์ฟเวอร์ในการติดต่อกับเน็ตเวิร์คอินเตอร์เฟซการ์ด

Thesis Title	UNIX Terminals Server using PC
Student	Mr. Apirom Saelao
Thesis Advisor	Mr. Kawin Sonthipermpoon
Level of Study	Master of Science in Computer Science and Information Technology
Department	Mathematic and Computer Science King Mongkut's Institute of Technology Ladkrabang
Year	1998

Abstract

The terminal connection in a network system has 3 general methods , terminal connectted by using local serial ports on the host , terminal connectted via a node in the network and terminal connectted by using serial port on a terminal server. The first and the second are limit by amount of serial ports on the host and node in the network, this problem can be solved by using the terminal server but the cost of this method is high.

This thesis describes a development of UNIX terminal server program on a personal computer , the use of personal computer (PC) as terminal server. Specifically , any PC connected to a PC acting as terminal server through serial port connection can emulate a UNIX terminal. The network protocol used through this research is ethernet using TCP/IP protocol encapsulated format and used TCP port number to separate the connection between host and each serial port on terminal server.

This program developed in DOS environment and used packet driver programming interface with the network interface card.

กิตติกรรมประกาศ

การจัดทำวิทยานิพนธ์ในครั้งนี้สำเร็จลุล่วงไปได้ด้วยดี เพราะได้รับความเมตตากรุณาจากท่านอาจารย์กวิน สนธิเพิ่มพูน ซึ่งได้ให้คำปรึกษาและแนะนำผู้วิจัยตลอดมา ผู้วิจัยรู้สึกซาบซึ้งในความอนุเคราะห์จากท่าน และกราบขอบพระคุณเป็นอย่างสูง

ผู้วิจัยขอกราบขอบพระคุณ เจ้าหน้าที่คณะเทคโนโลยีสารสนเทศ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังทุกท่าน ที่ให้ความช่วยเหลือในด้านต่างๆ เกี่ยวกับเอกสาร ขั้นตอนการดำเนินงานต่างๆ

ขอขอบพระคุณบริษัท โอเพ่น คอมพิวติ้ง เทคโนโลยีส์ จำกัด ที่ได้ให้ความอนุเคราะห์ผู้วิจัยในการยืมเครื่องคอมพิวเตอร์เพื่อใช้ในงานวิจัยครั้งนี้ คุณค่าและประโยชน์อันพึงมีจากวิทยานิพนธ์ฉบับนี้ผู้วิจัย ขอมอบแด่ผู้มีพระคุณทุกท่าน

อภิรมย์ แซ่เล้า

สารบัญ

สารบัญ (ต่อ)

สารบัญตาราง

สารบัญภาพ

หน้า

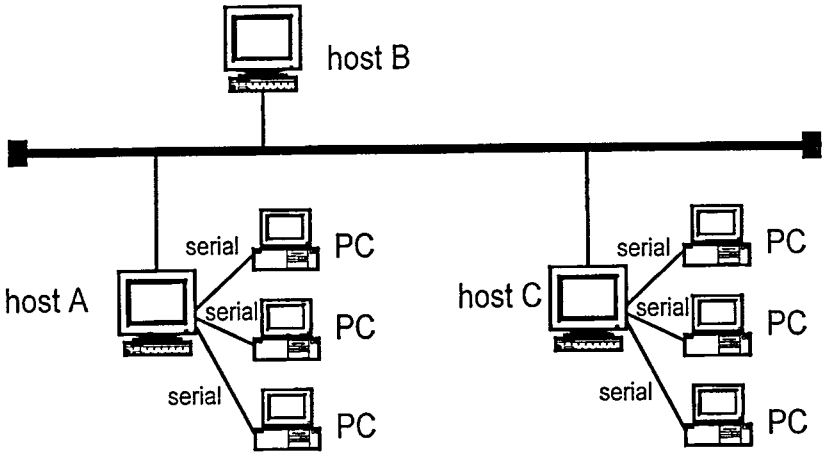
สารบัญภาพ (ต่อ)

หน้า

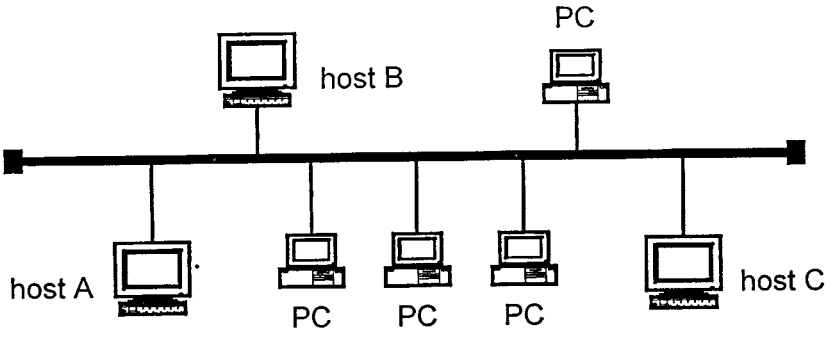
บทที่ 1
บทนำ

ความเป็นมาและความสำคัญของปัญหา

ในเครือข่ายของระบบยูนิกซ์ ซึ่งทั่วไปจะใช้โปรโตคอล TCP/IP วิธีการเชื่อมต่อเทอร์มินัลโดยใช้เครื่องคอมพิวเตอร์ส่วนบุคคล (Personal Computer) มี 3 วิธีใหญ่ๆ คือ ทางพอร์ตอนุกรมของโฮสคอมพิวเตอร์ (ดังรูปที่ 1.) ,ทางเครือข่าย (Network) ของระบบโดยผ่านทางอินเตอร์เฟซการ์ดที่คอมพิวเตอร์ส่วนบุคคล (ดังรูปที่ 2.) และทางเทอร์มินัลเซิฟเวอร์ (ดังรูปที่ 3.)

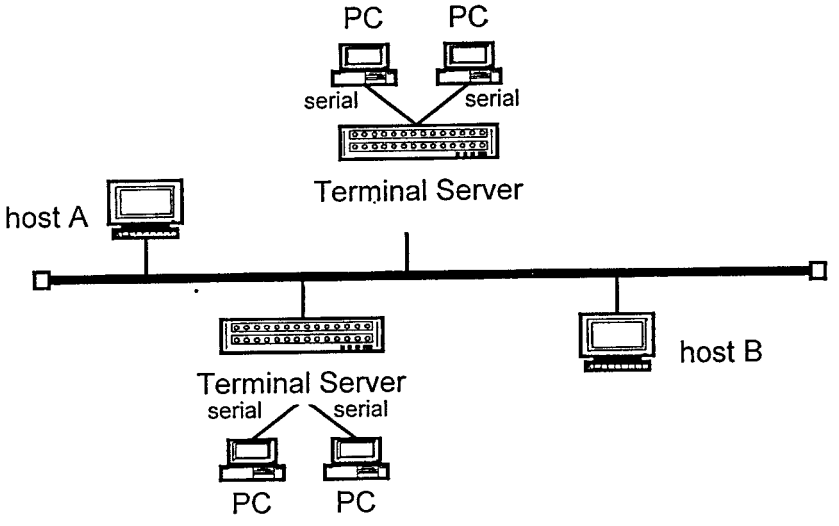


รูปที่ 1
แสดงการเชื่อมต่อเทอร์มินัลผ่านทางพอร์ตอนุกรมของโฮส



รูปที่ 2

แสดงการเชื่อมต่อเทอร์มินัลผ่านทางเครือข่าย



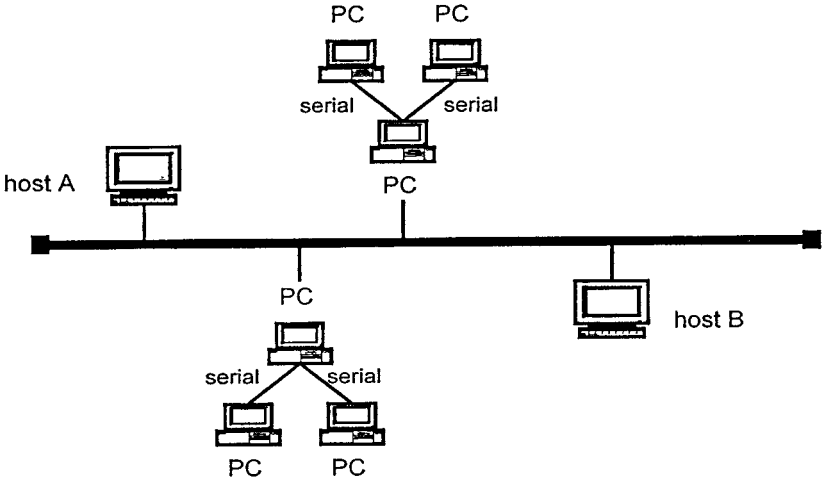
รูปที่ 3

แสดงการเชื่อมต่อเทอร์มินัลผ่านทางเทอร์มินัลเซิฟเวอร์

ความมุ่งหมายและวัตถุประสงค์ของการศึกษา

การเพิ่มจำนวนเทอร์มินัลในเครือข่ายจะถูกจำกัดโดยจำนวนโหนดในเน็ตเวิร์คและจำนวนพอร์ตอนุกรมของโฮสนั้นๆ

ส่วนการเพิ่มจำนวนเทอร์มินัลโดยใช้เทอร์มินัลเซิฟเวอร์มีราคาค่อนข้างสูง ดังนั้นการทำให้เครื่องคอมพิวเตอร์ที่มีอินเตอร์เฟซการ์ดอยู่แล้ว สามารถที่จะให้บริการคอมพิวเตอร์ส่วนบุคคลตัวอื่นๆให้เป็นเทอร์มินัลโดยผ่านพอร์ตอนุกรม (ดังรูป 4.) นั้น จึงเป็นอีกวิธีหนึ่งในการเพิ่มเทอร์มินัลในเครือข่ายได้



รูปที่ 4

แสดงการเชื่อมต่อเทอร์มินัลผ่านทางคอมพิวเตอร์ส่วนบุคคลเป็นเทอร์มินัลเซิร์ฟเวอร์

ทฤษฎีที่นำมาใช้ในการวิจัย

1. การสื่อสารข้อมูลแบบอนุกรม

2. ระบบเครือข่ายแบบท้องถิ่น (Local Area Network) ซึ่งจะใช้โทโปโลยีแบบบัส และโปรโตคอล TCP/IP

วิธีการดำเนินงาน

1. ศึกษาระบบและความเป็นไปได้ในการวิจัย
2. ศึกษาและทดลองเขียนโปรแกรมติดต่อกับอินเตอร์เฟซการ์ดโดยผ่านแพ็คเก็ตไดร์ฟเวอร์
3. ศึกษา telnet โปรโตคอล
4. เขียนโปรแกรมติดต่อระหว่าง PC กับโฮส โดยใช้ telnet
5. ศึกษาวิธีการติดต่อกันระหว่าง PC โดยผ่านพอร์ตอนุกรม
6. เขียนโปรแกรมติดต่อระหว่าง PC โดยผ่านพอร์ตอนุกรม
7. ผนวกโปรแกรม telnet เข้ากับโปรแกรมสื่อสารพอร์ตอนุกรม
8. ทดสอบการทำงานและแก้ไขจุดบกพร่อง

ขอบเขตการวิจัย

1. การวิจัยนี้จำกัดขอบเขตของพอร์ตอนุกรมบนเครื่องคอมพิวเตอร์ส่วนบุคคลเพียง 2 พอร์ต คือ พอร์ตอนุกรม 1 (COM 1:) และพอร์ตอนุกรม 2 (COM 2:)
2. การวิจัยนี้จำกัดขอบเขตของเครือข่ายอยู่ภายในเน็ตเวิร์คเดียวกัน ไม่สามารถติดต่อออกนอกเน็ตเวิร์คได้
3. การวิจัยนี้ไม่มีกระบวนการตรวจสอบความผิดพลาดในการรับส่ง กล่าวคือ ไม่มีโปรโตคอล ICMP (Internet Control Message Protocol)

ประโยชน์ที่คาดว่าจะได้รับ

1. สามารถนำเทอร์มินัลรุ่นเก่ามาต่อเข้ากับระบบเครือข่ายได้ เนื่องจากชนิดของเทอร์มินัลขึ้นอยู่กับว่าโฮสนั้นสามารถสนับสนุนการทำงานของเทอร์มินัลชนิดนั้นได้หรือไม่

2. ลดจำนวนอินเตอร์เน็ตแอดเดรสลง แต่สามารถต่อเทอร์มินัลได้จำนวนเท่าเดิม

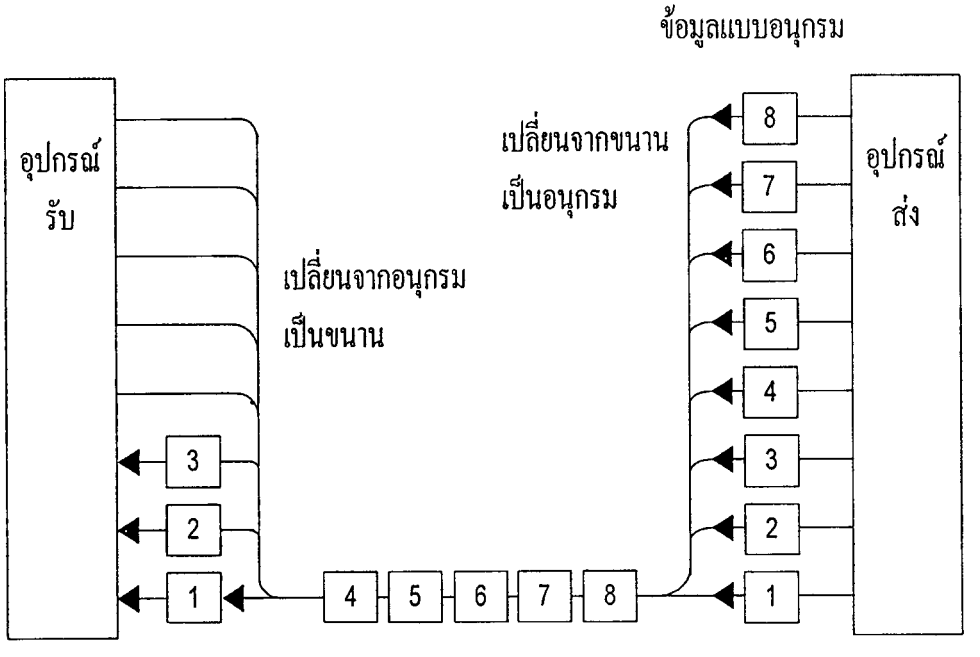
3. สามารถนำไมโครคอมพิวเตอร์รุ่นเก่า เช่น ไมโครคอมพิวเตอร์รุ่นที่ใช้ไมโครโปรเซสเซอร์ 8086, 80286, 80386SX, 80386 เป็นต้น มาใช้เป็นเทอร์มินัลเซิฟเวอร์ ซึ่งจะมีราคาต่ำกว่าเทอร์มินัลเซิฟเวอร์ทั่วไป

4. เป็นต้นแบบในการพัฒนาโปรแกรมเป็นโมเด็มเซิฟเวอร์และพรินเตอร์เซิฟเวอร์

บทที่ 2
ทฤษฎีพื้นฐาน

<u>การถ่ายโอนข้อมูลแบบอนุกรม</u>

การส่งข้อมูลแบบอนุกรม ข้อมูลจากจุดส่งจะถูกเปลี่ยนให้เป็นอนุกรมก่อนแล้วค่อยทะยอยส่งออกทีละบิตไปยังจุดรับ ณ ที่จุดรับจะต้องมีกลไกในการเปลี่ยนข้อมูลที่ส่งมาทีละบิต ให้เป็นสัญญาณแบบขนานซึ่งลงตัวพอดี คือ บิต 1 ลงที่บัสข้อมูลเส้นที่ 1 ดังรูปที่ 5.



รูปที่ 5
แสดงการส่งข้อมูลแบบอนุกรม

การที่จะทำให้การแปลงสัญญาณจากอนุกรมทีละบิตให้ลงพอดีนั้นจำเป็นจะต้องมีกลไกที่เหมาะสม เพื่อป้องกันการผิดพลาดในการรับ กลไกที่ว่านี้แบ่งเป็น 2 แบบ คือ

1. การสื่อสารแบบซิงโครนัส
2. การสื่อสารแบบอะซิงโครนัส

ในที่นี้จะกล่าวถึงเฉพาะการสื่อสารแบบอะซิงโครนัส ซึ่งใช้ในงานวิจัยนี้

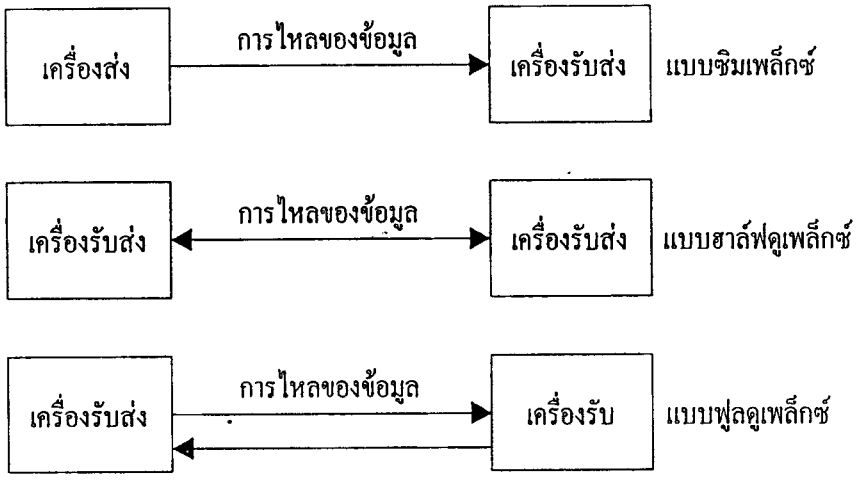
<u>รูปแบบของการติดต่อสื่อสารแบบอนุกรม</u>

การติดต่อแบบอนุกรมแบ่งตามรูปลักษณะได้ 3 แบบ ตามรูปที่ 6

1. แบบซิมเพล็กซ์ (simplex) ข้อมูลส่งได้ในทางเดียวเท่านั้น บางครั้งก็เรียกว่าการสั่งทิศทางเดียว (Unidirectional transmission)

2. แบบฮาล์ฟดูเพล็กซ์ (half duplex) ข้อมูลสามารถส่งได้ทั้งสองสถานี แต่จะต้องผลัดกันส่งและผลัดกันรับ จะส่งและรับพร้อมกันไม่ได้

3. แบบฟูลดูเพล็กซ์ (full duplex) ทั้งสองสถานีสามารถรับและส่งได้ในเวลาเดียวกัน



รูปที่ 6

แสดงรูปแบบของการติดต่อสื่อสารข้อมูลแบบอนุกรม

ความเร็วในการถ่ายโอนข้อมูลแบบอนุกรม

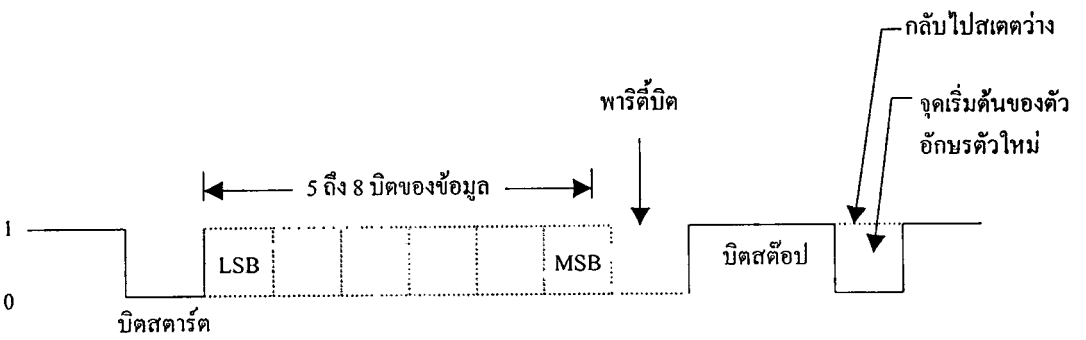
ความเร็วของการถ่ายโอนข้อมูลแบบอนุกรม หน่วยวัดเป็นบิตต่อวินาที (bps) หน่วยที่บรรยายถึงการเปลี่ยนแปลงของสัญญาณใน 1 วินาที เรียกว่าบอดเรต (baud rate) หรืออัตราบอด การเปลี่ยนแปลงของสัญญาณ 1 ครั้ง อาจจะแสดงถึงการส่งข้อมูลแบบอนุกรมมากกว่า 1 บิต ถ้าเขียนในรูปของสมการทางคณิตศาสตร์เราก็จะได้

อัตราบิต = n * อัตราบอด

n = จำนวนบิตใน 1 บอด

การสื่อสารแบบอะซิงโครนัส

การส่งแบบอะซิงโครนัสนี้ พัฒนามาจากการส่งโทรพิมพ์ในสมัยก่อน ลักษณะของสัญญาณแสดงไว้ในรูปที่ 7 เพื่อเพิ่มกลไกในการรับส่งอย่างถูกต้อง สัญญาณอะซิงโครนัสจะประกอบด้วยบิตเริ่มต้นหรือบิตสตาร์ต (start bit) และบิตสิ้นสุดหรือบิตสต๊อป (stop bit)



รูปที่ 7
แสดงรูปแบบการสื่อสารแบบอะซิงโครนัส

ขณะที่สถานะของการส่งเป็นแบบว่าง (Idle) คือ ยังไม่มีสัญญาณส่งออกมาจะมีสัญญาณหรือมีแรงดัน (หรือกระแส) ตลอดเวลา เพื่อความแน่ใจว่าฝ่ายรับยังติดต่ออยู่กับฝ่ายส่ง เมื่อเริ่มจะส่งข้อมูล สัญญาณของอะซิงโครนัสจะเป็น 0 หนึ่งช่วงสัญญาณนาฬิกา บิตนี้เรียกว่าสตาร์ตบิต ตามหลังของสตาร์ตบิตก็จะเป็นข้อมูลสำหรับ 1 ตัวอักษร ซึ่งอาจจะมีขนาดตั้งแต่ 5 บิต จนถึง 8 บิต โดยบิตที่มีค่าน้อยที่สุด (LSB) จะถูกส่งออกมาก่อนไล่ไปจนถึงบิตที่มีค่ามากที่สุด (MSB) ตามหลังข้อมูลก็จะเป็นพาริตี้บิต ซึ่งอาจจะใช้หรือไม่ใช้ก็ได้ พาริตี้บิตทำหน้าที่เป็นตัวตรวจสอบความถูกต้องของสัญญาณที่ได้รับ พาริตี้บิตอาจจะเป็นแบบ คู่ (Even) หรือแบบ คี่ (Odd) หมายความว่าถ้าหากเป็นพาริตี้คู่ จำนวนบิตที่เป็น 1 ในช่วงบิตข้อมูลกับพาริตี้รวม แล้วจะต้องเป็นจำนวนคู่ ผู้ส่งจะต้องทำหน้าที่ตรวจสอบข้อมูลแล้วใส่พาริตี้บิตเอง ฝ่ายรับเมื่อรับแล้วก็จะต้องตรวจสอบดูว่าเป็นจริงดังสถานการณ์ที่ตั้งเอาไว้หรือไม่ หากผิดพลาดก็หมายความว่าสัญญาณที่รับนั้นผิดพลาดไปจากสถานีส่งส่งออกมา

หลังจากบิตพาริตี้แล้วก็จะต้องมีสต๊อปบิตซึ่งเป็น 1 ความกว้างของสต๊อปบิตอาจจะเป็น 1, 1.5 หรือ 2 แล้วแต่ผู้รับและผู้ส่งจะตกลงใช้กันเอง การเริ่มใช้พอร์ตอนุกรมจึงจำเป็นจะต้องตั้งค่าต่างๆสำหรับเป็นการส่งแบบอนุกรมอันได้แก่

1. ความเร็วในการส่ง
2. ความยาวรหัส 1 อักขระ
3. บิตตรวจสอบ
4. จำนวนสต๊อปบิต

<u>พอร์ต RS 232C</u>

โดยปกติไมโครคอมพิวเตอร์จะมีพอร์ตที่เป็นแบบอนุกรม เรียกว่าพอร์ต RS 232C อยู่ในตัวเองอยู่แล้ว พอร์ต RS 232C นี้ทำหน้าที่รับและส่งข้อมูลในแบบอนุกรม เรียกว่า Universal Asynchronous Adapter เหตุที่มีชื่อเรียกว่า RS 232C เนื่องจากสมาคมผู้ผลิตอุปกรณ์อิเล็กทรอนิกส์ของอเมริกาหรือ EIA ได้กำหนดมาตรฐานของอุปกรณ์การสื่อสารแบบอนุกรมเอาไว้ภายใต้ชื่อว่า RS 232C มาตรฐานของการส่งข้อมูลแบบอนุกรมมีหลายมาตรฐาน แต่ที่นิยมกันมากที่สุดสำหรับไมโครคอมพิวเตอร์ก็คือ RS 232C

หน้าที่สำคัญของการสื่อสารแบบอะซิงโครนัสก็คือ

รับสัญญาณ

1. เปลี่ยนสัญญาณเข้ามาแบบอนุกรมให้เป็นแบบขนาน
2. ตรวจสอบความผิดพลาดของสัญญาณที่รับ
3. ตัดสต๊อปบิตและพาริตี้บิตออก
4. ส่งสัญญาณให้ซีพียูรู้ว่ารับสัญญาณไว้แล้ว
5. เพิ่มสัญญาณควบคุมโมเด็มที่ต่อเชื่อม

ส่งสัญญาณ

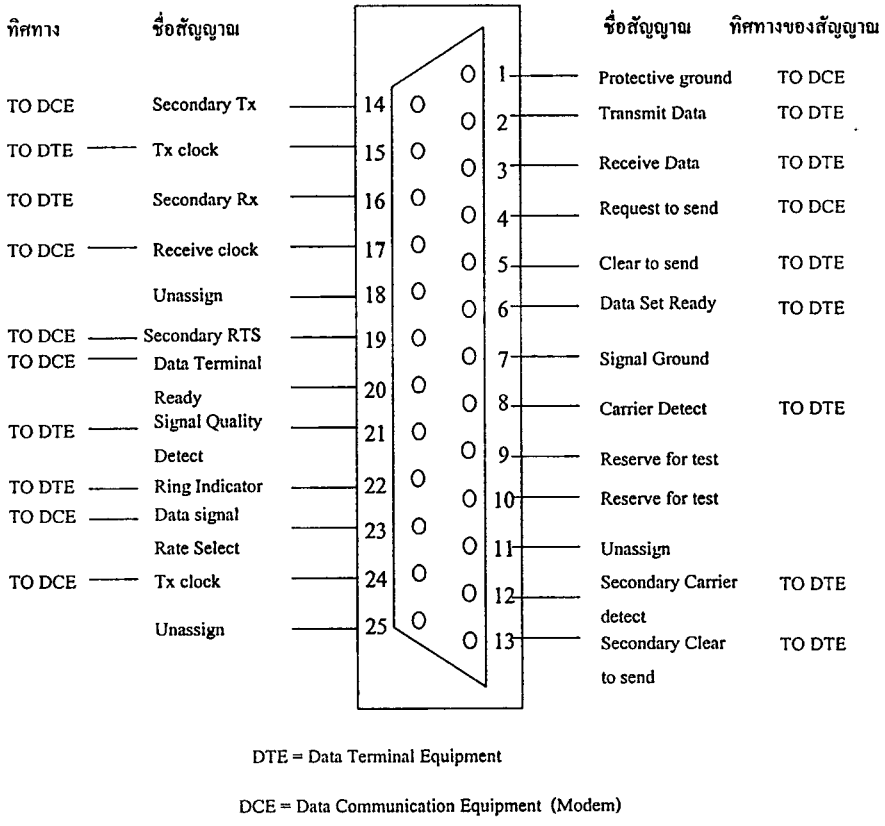
1. เปลี่ยนสัญญาณแบบขนานจากซีพียูค่อยทะยอยส่งออกเป็นแบบอนุกรม
2. เพิ่มบิตสำหรับตรวจสอบความผิดพลาดของสัญญาณที่ส่ง
3. เพิ่มสต๊อปบิตและพาริตี้บิต
4. ส่งสัญญาณให้ซีพียูรู้ว่าส่งสัญญาณไปแล้ว
5. เพิ่มสัญญาณควบคุมโมเด็มที่ต่อเชื่อม

มาตรฐาน RS 232C

มาตรฐาน RS 232C ได้จัดพิมพ์ขึ้นเมื่อปีค.ศ. 1969 โดยสมาคมผู้ผลิตอุปกรณ์อิเล็กทรอนิกส์แห่งสหรัฐอเมริกา RS ย่อมาจาก Recommened Standard ส่วน 232 เป็นหมายเลขบ่งบอกของมาตรฐานตัวนี้ C เป็นรุ่นของมาตรฐานตัวนี้ จุดประสงค์ของมาตรฐานนี้ก็เพื่อบรรยายคุณลักษณะของการเชื่อมต่ออุปกรณ์รับส่งข้อมูลปลายทาง (Data Terminal Equipment DTE) กับอุปกรณ์สื่อสารข้อมูล (Data Communication Equipment DCE) สำหรับผู้ใช้ไมโครคอมพิวเตอร์ DTE หมายถึงไมโครคอมพิวเตอร์และ DCE หมายถึงโมเด็ม อุปกรณ์อื่นๆ เช่น เครื่องพิมพ์ที่รับสัญญาณแบบอนุกรมอาจจะเป็นได้ทั้ง DTE และ DCE ขึ้นอยู่กับผู้ผลิต

การกำหนดจุดขั้วต่อของ RS 232C

มาตรฐานของ RS 232C กำหนดข้อต่อแบบ DB-25 แต่ละขาของขั้วต่อกำหนดไว้ดังในรูปที่ 8



รูปที่ 8
การกำหนดของขั้วต่อ RS 232C

Transmit Data (TD ขาที่ 2)

เป็นสัญญาณที่ส่งออกจาก DTE (หรือตัวไมโครคอมพิวเตอร์) ไปยังโมเด็มหรือต่อเข้าโดยตรงกับไมโครคอมพิวเตอร์ตัวอื่น หรือเครื่องพิมพ์ เมื่อไม่มีสัญญาณส่งออกสถานภาพของลอจิกที่ขานี้จะมีค่าเท่ากับ "1" หรือเทียบเท่ากับสต๊อปบิต

Receive Data (RD ขาที่ 3)

เป็นทางของสัญญาณเข้าไปยัง DTE หรือไมโครคอมพิวเตอร์เมื่อไม่มีสัญญาณรับเข้ามา ขานี้จะมีสถานภาพทางลอจิก เป็น "1"

Request To Send (RTS ขาที่ 4)

ใช้สำหรับส่งสัญญาณไปยังโมเด็มหรือเครื่องพิมพ์เป็นการเรียกร้องที่จะส่งสัญญาณทาง ขา 2 สัญญาณนี้ใช้คู่กับ CTS หรือ Clear to send อุปกรณ์รับหากได้รับสัญญาณ RTS จะตรวจสอบตัวเองว่าพร้อมจะรับสัญญาณได้หรือยัง หากพร้อมที่จะรับก็ส่งสัญญาณออกไปที่สาย CTS

Clear To Send (CTS ขาที่ 5)

ดังอธิบายไว้ใน RTS เมื่อสัญญาณนี้อยู่ในสถานะออฟ (negative voltage หรือลอจิก "1") หมายความว่า อุปกรณ์รับกำลังบอกว่าพร้อมที่จะรับข้อมูลแล้ว

Data Set Ready (DSR ขาที่ 6)

เมื่อสัญญาณสายนี้อยู่ในสถานะออน (หรือลอจิก 0) เป็นการบอกไมโครคอมพิวเตอร์หรือฝ่ายส่งว่า โมเด็มต่อเข้ากับสายโทรศัพท์เรียบร้อยแล้วและพร้อมที่จะส่งได้แล้ว โมเด็มที่มีการหมุนหมายเลขอัตโนมัติจะส่งสัญญาณสายนี้ไปบอกให้คอมพิวเตอร์รู้ว่าต่อโทรศัพท์ได้สำเร็จแล้ว

Signal Ground (SG ขาที่ 7)

SG ทำหน้าที่เป็นระดับแรงดันอ้างอิงสำหรับทุกๆสายของสัญญาณ จะมีแรงดันเป็น"0" เมื่อเทียบกับสัญญาณตัวอื่น

Carrier Detect (CD ขาที่ 8)

โมเด็มจะส่งสัญญาณที่อยู่ในสถานะออน (ลอจิก "0") ไปบอกไมโครคอมพิวเตอร์ เมื่อได้รับสัญญาณจากโมเด็มของอีกฝ่ายหนึ่ง

Data Terminal Ready (DTR ขาที่ 20)

คอมพิวเตอร์เปิดสัญญาณสายนี้ให้ออน (ลอจิก "0") เมื่อพร้อมที่จะติดต่อกับโมเด็ม โมเด็มส่วนมากจะไม่รายงานสถานภาพของตัวเอง (CD , DSR , CTS) ให้คอมพิวเตอร์รู้ หากคอมพิวเตอร์ไม่เปิดสัญญาณ DTR

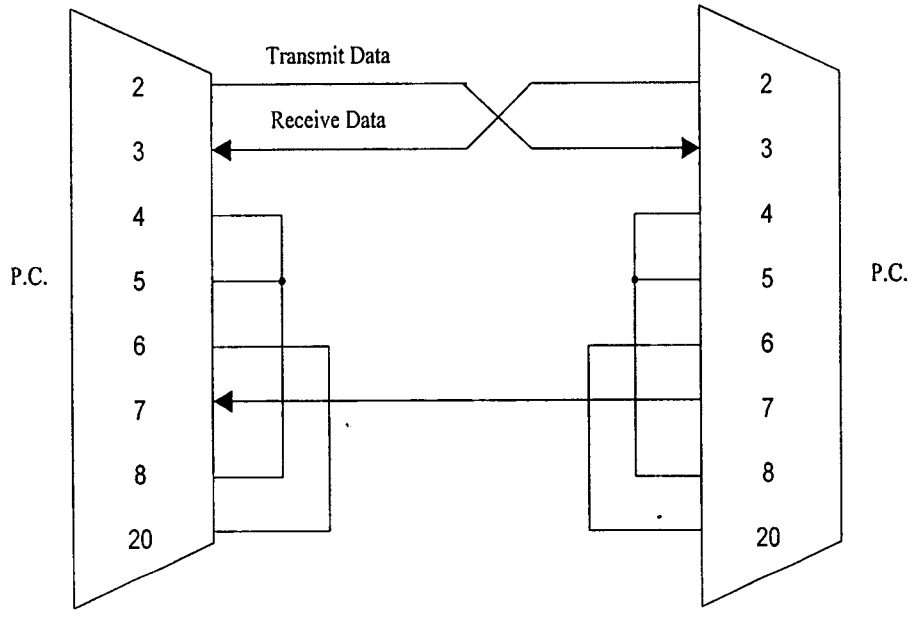
Ring Indicator (RI ขาที่ 22)

สัญญาณนี้จะออนเมื่อมีสัญญาณกระดิ่งมา

<u>การเชื่อมต่อคอมพิวเตอร์กับคอมพิวเตอร์โดยตรง</u>

การต่อเครื่องคอมพิวเตอร์เข้าด้วยกันโดยใช้ RS 232C แล้วถ่ายโอนข้อมูลจากเครื่องหนึ่งไปอีกเครื่องหนึ่ง วิธีการต่อแบบนี้เรียกว่า Null Modem คือ ไม่ใช้โมเด็มนั่นเอง (ในกรณีที่ระยะทางไม่เกิน 50 ฟุต)

วิธีการต่อ RS 232C เข้าระหว่างเครื่องคอมพิวเตอร์โดยตรงมีอยู่หลายวิธีตามแต่ขบวนการที่จะใช้ ถ้าไม่ต้องการมีการตรวจสอบสัญญาณกันก็ต่อ RD เข้า TD ของอีกเครื่องหนึ่ง สายกราวนด์ต่อถึงกันดังรูปที่ 9 ก็สามารถใช้งานถ่ายโอนข้อมูลได้แล้ว



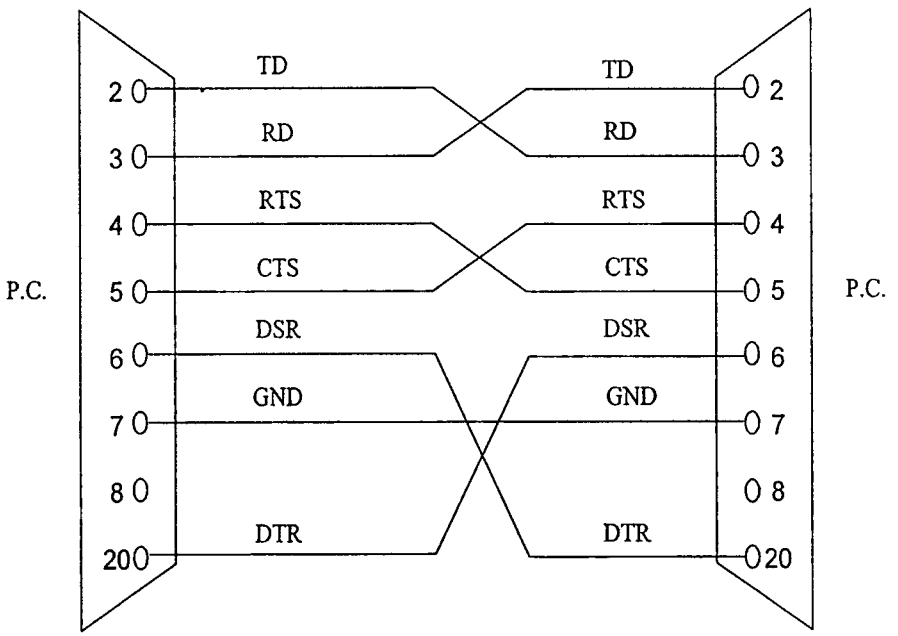
รูปที่ 9

แสดงการถ่าย RS 232C ระหว่างไมโครคอมพิวเตอร์อย่างง่ายๆ

ปกติซอฟแวร์ที่ให้บริการเกี่ยวกับพอร์ต RS 232 จะส่งสัญญาณ RTS หรือ Request to send ออกมาที่ขา 4 ก่อน เมื่อ CTS หรือ Clear to send ที่ขา 5 เป็นลอจิก "1" (หรือไฟลบ) จึงจะเริ่มทำการส่งข้อมูลที่โอเปอเรเตอร์บอกให้ส่งออกไปที่ขา 2 ในกรณีที่เป็นการต่อแบบง่ายๆ ในรูปที่ 9 จึงถือว่าเป็นการหลอกคอมพิวเตอร์โดยเอาขา 4 RTS ต่อเข้ากับขา 5 หรือ CTS เพื่อให้คอมพิวเตอร์ส่งข้อมูลได้ทันทีโดยไม่ต้องการความเรียบร้อยของฝ่ายรับ สำหรับขา 6 Data Set Ready ต่อเข้ากับขาที่ 20 Data Terminal Ready ก็ทำนองเดียวกัน โดยปกติคอมพิวเตอร์จะถามอุปกรณ์ที่มาต่อพ่วงกับ RS 232 (C) ว่าพร้อมที่จะส่งแล้วยัง โดยส่งสัญญาณถามที่ขา 20 และรอคำตอบที่ขา 6

ในการต่อแบบนี้ฝ่ายรับจะต้องรอรับอยู่ก่อนแล้ว ก่อนที่ฝ่ายส่งจะเป็นผู้ส่งไม่เช่นนั้นข้อมูลที่ส่งออกมาจะสูญหาย เพราะฝ่ายส่งไม่ได้ตรวจสอบความเรียบร้อยของฝ่ายรับก่อน

การต่อสายให้มีการตรวจสอบสัญญาณโต้ตอบ (Hand Shake) จะแสดงในรูปที่ 10



รูปที่ 10

แสดงการต่อไมโครคอมพิวเตอร์ผ่าน RS 232C แบบมี Hand Shake

ระบบเครือข่ายแบบท้องถิ่น (Local Area Network)

ไมโครคอมพิวเตอร์ได้รับการออกแบบมาให้เหมาะสมกับผู้ใช้ โดยได้กำหนดรูปแบบการใช้ไมโครคอมพิวเตอร์แบบผู้ใช้คนเดียว และทำงานทีละงาน (single user single tasking) การทำงานของไมโครคอมพิวเตอร์ในยุคแรกนี้ใช้ไมโครโปรเซสเซอร์ที่มีขีดความสามารถทางด้านความเร็วในการทำงานจำกัด ดังนั้นหากทำเป็นระบบแบ่งเวลาการทำงาน (time sharing) ก็จะยิ่งทำให้ระบบมีการทำงานช้าลงไปอีกมาก การทำงานของไมโครคอมพิวเตอร์อาจจะไม่พึงต้องการของผู้ใช้ก็ได้ อย่างไรก็ดี การพัฒนาในเชิงระบบจัดการให้ไมโครคอมพิวเตอร์ทำงานได้ดีขึ้นก็ต้องอาศัยเวลาในการพัฒนา

การแบ่งอุปกรณ์บางอย่างที่มีราคาแพงมาใช้ร่วมกันจึงมีความต้องการมากขึ้น โดยเฉพาะระบบคอมพิวเตอร์ขนาดเล็ก เพราะในสำนักงานอาจมีการใช้ไมโครคอมพิวเตอร์กันหลายเครื่อง แต่ก็ไม่สามารถจะให้ทุกเครื่องมีเครื่องพิมพ์คุณภาพสูงดังนั้นจึงอาจต้องใช้วิธีการในการใช้ร่วมกันเพื่อให้เกิดประโยชน์คุ้มค่ามากยิ่งขึ้น

ไมโครคอมพิวเตอร์จึงได้รับการพัฒนาเพื่อให้ใช้งานได้ดีขึ้น โดยมีแนวทางในการพัฒนา เพื่อให้ใช้งานได้หลายคนพร้อมกัน (multiuser) รูปแบบของการพัฒนาเป็นไปในแนวทางสองแนวทาง คือ ระบบโปรแกรมจัดระบบงานที่จะให้ใช้งานร่วมกันหลายๆคน และระบบเน็ตเวิร์ค

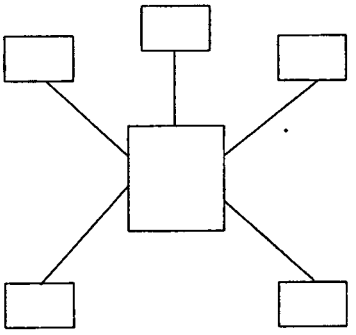
การนำเอาระบบไมโครคอมพิวเตอร์หลายๆเครื่องมาต่อรวมกันเป็นเน็ตเวิร์ค เพื่อให้ได้ใช้งานคล้ายระบบ multi-user โดยแต่ละคนจะมีไมโครโปรเซสเซอร์ของตัวเอง ที่จะทำงานโดยอิสระได้ ระบบเน็ตเวิร์คที่ต่อเป็นระบบนี้จะครอบคลุมในพื้นที่ขนาดไม่กว้างมากนัก เรียกว่าโลคัลเน็ตเวิร์ค (local network)

โครงข่ายของระบบเน็ตเวิร์ค

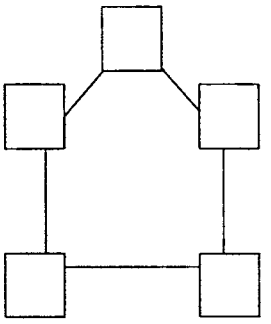
โทโปโลยี (topology) หมายถึงรูปร่างของเน็ตเวิร์ค โดยพิจารณาจากการลากเส้นมาต่อรวมกันเป็นกิ่งก้านหรือรูปแบบของเน็ตเวิร์ค คอมพิวเตอร์เน็ตเวิร์คหมายถึง รูปโครงเชื่อมต่อ โดยให้คอมพิวเตอร์เสมือนเป็นสถานีสายส่งสัญญาณ คือเส้นต่อของสถานี โครงข่ายของ LAN จึงเป็นรูปแบบที่นำเอาคอมพิวเตอร์หรืออุปกรณ์อื่นๆ มาประกอบโดยเชื่อมโยงกันและสายที่เชื่อมโยง

นี้เราจะเรียกว่า ลิงค์ (link) โทโปโลยีของเน็ตเวิร์คที่ใช้กัน ได้แก่ โทโปโลยีในรูปแบบดาว (star) บัส (bus) ต้นไม้ (tree) และแบบริง (ring) ดังรูปที่ 11.

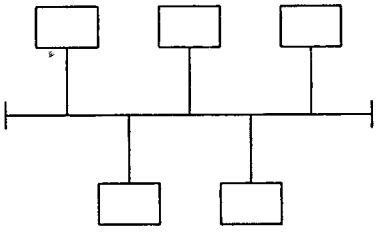
ในที่นี้จะกล่าวถึงเฉพาะโทโปโลยีแบบบัสซึ่งใช้ในวิทยานิพนธ์นี้



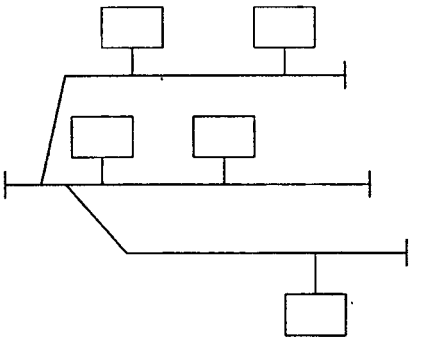
(ก) แบบดาว



(ข) แบบวงแหวน



(ค) แบบบัส



(ง) แบบต้นไม้

รูปที่ 11

แสดงโทโปโลยีของ LAN

โทโปโลยีแบบบัส (Bus topology)

โทโปโลยีแบบบัสนั้นเป็นแบบที่มีโครงสร้างไม่ยุ่งยาก ทุกๆสถานีจะเชื่อมต่อเข้าหาบัสโดยผ่านทางอุปกรณ์อินเตอร์เฟสที่เป็นฮาร์ดแวร์ การจัดส่งข้อมูลลงไปในบัสนี้จึงสามารถทำให้ข้อมูลไปถึงได้ทุกสถานีเพราะอยู่บนบัสร่วมกัน

สถาปัตยกรรมของเน็ตเวิร์ค (Network Architectures)

สถาปัตยกรรม OSI (Open Systems Interconnection) ประกอบโดยชั้นของโปรโตคอล 7 ชั้นด้วยกัน แต่ละชั้นจะมีหน้าที่แตกต่างกันไปโดยชั้นหนึ่งๆจะเกี่ยวข้องกับอีก 3 ชั้นคือ

- ชั้นที่สูงกว่า จะต้องการการบริการจากชั้นนี้
- ชั้นที่ต่ำกว่า จะคอยให้การบริการกับชั้นนี้
- ชั้นที่เท่าเทียมกัน ชั้นนี้จะติดต่อส่งข้อมูลกับชั้นที่เท่ากันของคอมพิวเตอร์เครื่องอื่น

ชั้นทั้ง 7 ของ OSI มีดังนี้

1. Application Layer เป็นชั้นบนสุด และคอยให้บริการแก่โปรแกรมประยุกต์ของผู้ใช้งาน

2. Presentation Layer จะแปลงข้อมูลเพื่อให้อยู่ในรูปแบบที่คอมพิวเตอร์แต่ละยี่ห้อ เข้าใจได้ มีการเข้ารหัสข้อมูล (Data Encryption) และการลดจำนวนข้อมูล(Data Compression)

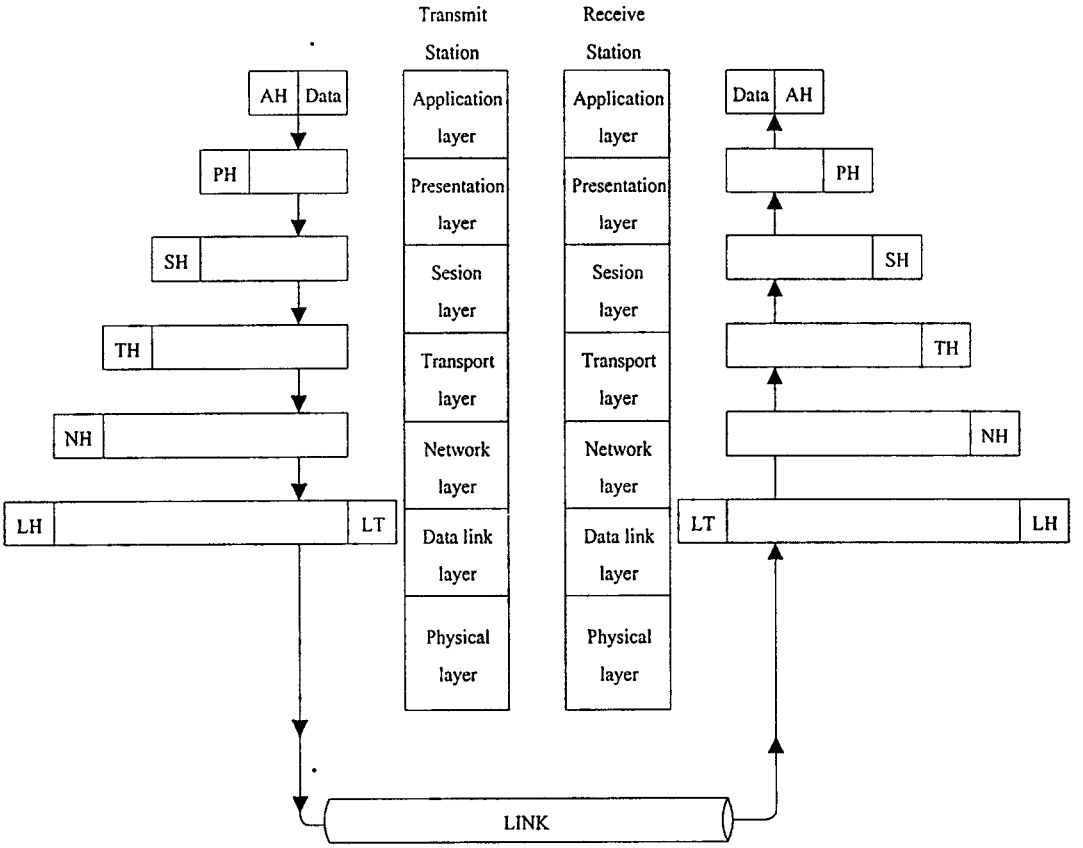
3. Session Layer เป็นชั้นเริ่มต้นและดูแลประสานงานการสื่อสารระหว่างเน็ตเวิร์คทั้ง 2 ระบบ

4. Transport Layer ชั้นนี้จะจัดการในด้านการส่งข้อมูล ระหว่างเครื่องคอมพิวเตอร์รวมทั้งการจัดลำดับส่วนต่างๆของข้อความ อัตราเร็วในการส่งและการตัดส่วนของข้อมูลที่ซ้ำกันทิ้งไป

5. Network Layer จัดการการตั้งแอดเดรส และการกำหนดเส้นทางของกลุ่มข้อมูลต่างๆให้ไปถึงปลายทาง

6. Datalink Layer จะจัดการในเรื่องการรับส่งและแปลงข้อมูลจากระดับฟิสิคัลที่เป็นบิตให้อยู่ในรูปแบบที่มีความหมาย และตรวจสอบความผิดพลาดระหว่างข่ายงานที่ติดต่อกันอยู่

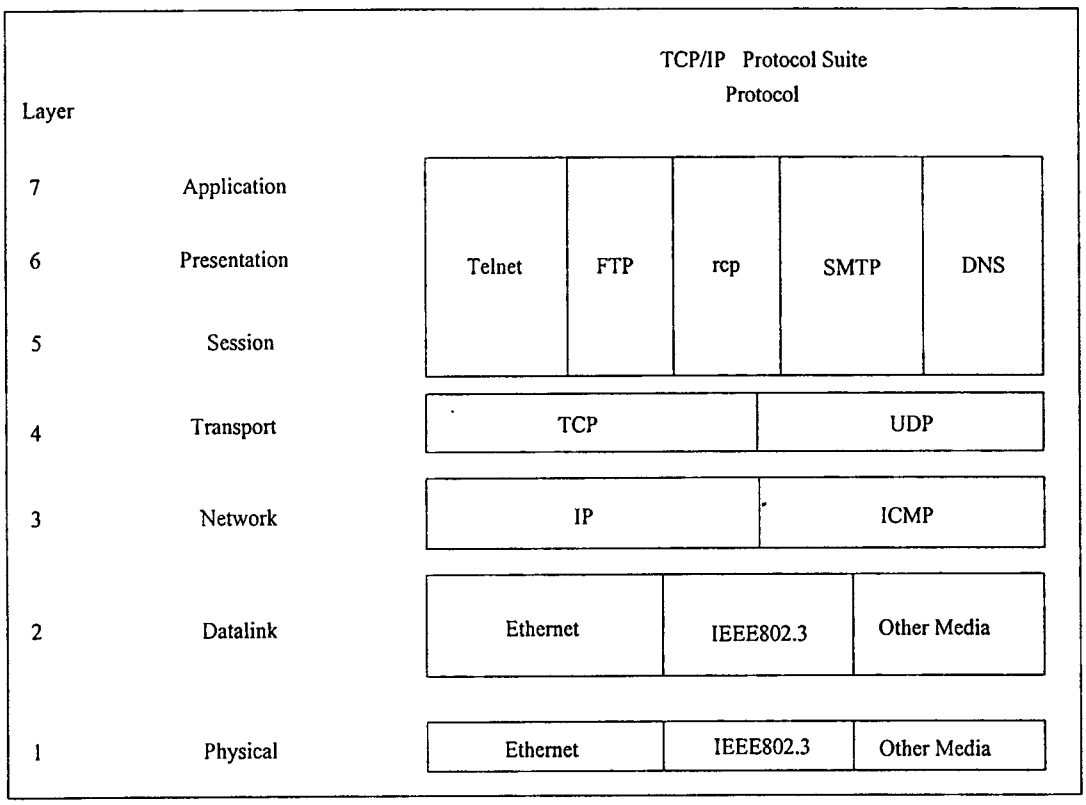
7. Physical Layer เป็นคุณสมบัติทางฮาร์ดแวร์ของสายส่ง และอุปกรณ์ในการส่ง ซึ่งการส่งข้อมูลระหว่างเครื่องจะมองในลักษณะรูปแบบที่เป็นบิต



รูปที่ 12
แสดงชั้นทั้ง 7 ของ OSI

โปรแกรมประยุกต์ที่เขียนสำหรับ TCP/IP จะถูกกำหนดโดยชั้น (layer) ของโปรโตคอลที่เรียกกันโดยทั่วไปว่า โปรโตคอลสแตก (Protocol stack) โปรแกรมประยุกต์ที่คอยให้บริการ

บนชั้นที่สูงกว่าจะส่งผ่านข้อมูลลงมาให้กับชั้นของโปรโตคอลที่ต่ำกว่า จนกระทั่งถึงขั้นที่ต่ำสุด เมื่อข้อมูลถูกส่งออกไป แล้วเครื่องคอมพิวเตอร์ของผู้รับรับข้อมูลจากโปรโตคอลชั้นล่างก็จะค่อยๆ ส่งผ่านข้อมูลชั้นไปยังชั้นของโปรโตคอลที่สูงกว่าโดยโปรแกรมประยุกต์ชั้นสูงกว่านั่นเอง ดังรูปที่ 13.[8]



รูปที่ 13

แสดงชั้นทั้ง 7 ของ OSI เปรียบเทียบกับ TCP/IP โปรโตคอล

ตารางที่ 1

ตาราง TCP/IP โปรโตคอล

Protocol	Service	OSI Layer
Internet Protocol (IP)	ให้บริการรับส่งแพ็คเก็ตระหว่างโหนด	Layer 3
Internet Control Message Protocol (ICMP)	ควบคุมความผิดพลาดระหว่างการรับส่งและควบคุมข้อความ (messages)ระหว่างโฮสและเกตเวย์	Layer 3
Transmission Control Protocol (TCP)	จัดการสตรีม(stream)สำหรับรับส่งเซอร์วิสระหว่าง client	Layer 4
User Datagram Protocol (UDP)	จัดการแพ็คเก็ตสำหรับรับส่งเซอร์วิสระหว่าง client	Layer 4
File Transfer Protocol (FTP)	ระดับของโปรแกรมประยุกต์สำหรับรับส่งไฟล์	Layer 5-7
Telnet	ให้บริการเทอร์มินัลอีมูเลต (Terminal Emulation)	Layer 5-7
Domain Name	แปลงชื่อโดเมน(Domain Names) เป็นอินเตอร์เน็ตแอดเดรส	Layer 5-7

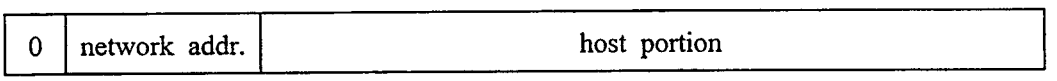
ฟิสิคัลแอดเดรสและอินเตอร์เน็ตแอดเดรส

ในระดับดาต้าลิงค์ (datalink) โหนดบนเน็ตเวิร์คติดต่อกันโดยใช้ฟิสิคัลเน็ตเวิร์คโหนดบนเน็ตเวิร์คทุกๆโหนดจะมีแอดเดรสเป็นของตัวเองซึ่งจะไม่ซ้ำกัน สำหรับเน็ตเวิร์คบนอีเทอร์เน็ตจะใช้ตัวเลข 6 ไบต์ ในการกำหนดฟิสิคัลเน็ตเวิร์ค เช่น 08-00-14-57-69-69

อินเตอร์เน็ตโปรโตคอลแอดเดรส (IP address) ใช้สำหรับกำหนดแอดเดรสของโหนดในระดับลอจิคัล (logical) ซึ่งประกอบด้วยตัวเลขขนาด 4 ไบต์ เช่น 129.47.6.17 อินเตอร์เน็ตแอดเดรสถูกแบ่งออกเป็น 2 ส่วน ส่วนเน็ตเวิร์คกับส่วนโฮส (Host) และแบ่งออกเป็นคลาสดังนี้

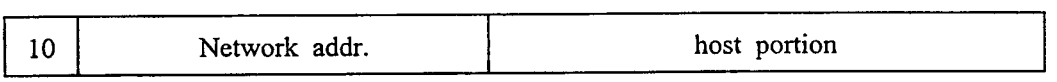
Class A

อินเตอร์เน็ตแอดเดรสคลาส A แบ่ง 1 ไบต์มาใช้สำหรับเน็ตเวิร์คแอดเดรส และ 3 ไบต์สำหรับโฮส คลาสนี้สามารถมีเน็ตเวิร์คได้ทั้งหมด 127 เน็ตเวิร์ค โดยแต่ละเน็ตเวิร์คสามารถมีโหนดได้ 16,777,214 โหนด



Class B

คลาส B แบ่ง 2 ไบต์เป็นเน็ตเวิร์คแอดเดรส และ 2 ไบต์สำหรับโฮส โดยกำหนด 2 บิตแรกของไบต์แรกมีค่า 10 จึงทำให้เน็ตเวิร์คคลาสนี้มีจำนวนเน็ตเวิร์คได้ 16,384 เน็ตเวิร์ค โดยแต่ละเน็ตเวิร์คมีโหนดได้ 65,534 โหนด



Class C

คลาส C แบ่ง 3 ไบต์เป็นเน็ตเวิร์คแอดเดรส และ 1 ไบต์สำหรับโฮส โดยกำหนด 3 บิตแรกของไบต์แรกมีค่า 110 จึงทำให้คลาสนี้มีจำนวนเน็ตเวิร์ค 2,097,152 เน็ตเวร์คโดยแต่ละเน็ตเวิร์คมีจำนวนโหนดได้ 254 โหนด

110	network addr.	host portion

แอดเดรสยกเว้น

มีการกำหนดแอดเดรสที่ใช้สำหรับเป็นแอดเดรสพิเศษ 2 แอดเดรส ได้แก่

- เน็ตเวิร์คแอดเดรส จะเป็นแอดเดรสในส่วนของโฮสที่มีค่าเป็น 0 สำหรับทุกๆเน็ตเวิร์คในคลาสนั้นๆ เช่น แอดเดรส 129.47.0.0

-บรอดคลาสแอดเดรส จะเป็นแอดเดรสโดยบิตของโฮสมีค่าเป็น 1 หมด เช่น 129.47.255.255

<u>อีเทอร์เน็ต (Ethernet)</u>

อีเทอร์เน็ตเป็นโปรโตคอลซึ่งใช้สำหรับเชื่อมต่อระหว่างเครื่องในระดับ link level การรับส่งข้อมูลจะใช้ลักษณะเป็นเฟรม (Frame) ขนาดของเฟรมจะไม่คงที่เป็น variable length อยู่ระหว่าง 64-1518 ไบต์ ดังรูปที่ 14 แสดงถึงรูปแบบเฟรมซึ่งประกอบด้วย ฟิสิคัลแอดเดรสของผู้ส่งและผู้รับ

ในแต่ละเฟรมของอีเทอร์เน็ต นอกจากจะประกอบด้วยฟิสิคัลแอดเดรสของผู้ส่งและผู้รับแล้ว แต่ละเฟรมยังประกอบด้วย ฟิลด์ preamble , ฟิลด์ type , ฟิลด์ข้อมูล และ ฟิลด์ CRC (Cyclic Redundancy Check)

ฟิลด์ Preamble ประกอบด้วย 64 bit ของ bit 0 และ 1 เพื่อช่วยในการซิงโครไนซ์เน็ตเวิร์ค ก่อนส่ง ฟิลด์ CRC ขนาด 32 bit ใช้สำหรับเช็คข้อผิดพลาดในระหว่างส่ง โดยผู้ส่งเป็นผู้คำนวณค่าแล้วเก็บในฟิลด์นี้ เมื่อผู้รับได้รับก็จะเช็คฟิลด์นี้อีกครั้ง

ฟิลด์ type ประกอบด้วยตัวเลขมีขนาด 16 bit ใช้สำหรับกำหนดชนิดของข้อมูลที่อยู่ในเฟรม ฟิลด์นี้มีความสำคัญมาก เนื่องจากผู้รับจะใช้ฟิลด์นี้ในการกำหนดว่า เฟรมที่รับเข้ามาเป็นโปรโตคอลชนิดใด เพื่อที่จะกำหนดซอฟแวร์โมดูลในการประมวลผลเฟรมนั้น

Preamble	Destination Address	Source Address	Frame Type	Frame Data	CRC
64 bits	48 bits	48 bits	16 bits	368-12000 bits	32 bits

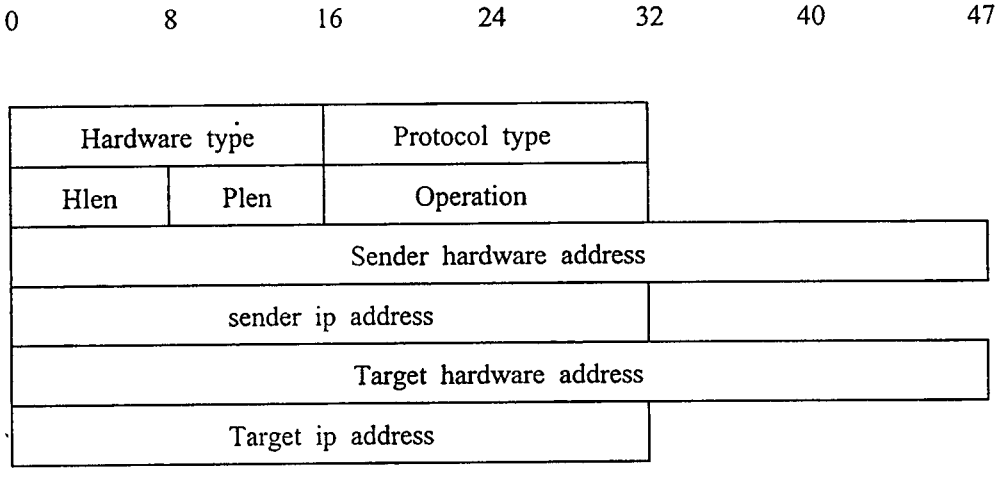
รูปที่ 14
แสดงรูปแบบของอีเทอร์เน็ตเฟรม

ARP โปรโตคอล

ARP (Address Resolution Protocol) โปรโตคอลเป็นโปรโตคอลที่ใช้สำหรับสร้างการติดต่อระหว่างอีเทอร์เน็ตแอดเดรสกับอินเตอร์เน็ตแอดเดรส การส่งแพ็คเก็ตผ่านทางเน็ตเวิร์คผู้ส่งจะต้องรู้อีเทอร์เน็ตแอดเดรสของผู้รับ แต่ถ้าหากผู้ส่งรู้เฉพาะอินเตอร์เน็ตแอดเดรสของผู้รับเท่านั้น ผู้ส่งจะต้องส่ง ARP แพ็คเก็ต โดยบรอดคลาส ARP Request ไปที่เน็ตเวิร์คด้วย IP แอดเดรสของผู้รับ ทุกโหนดในเน็ตเวิร์ครับ ARP Request แพ็คเก็ต แต่จะมีเพียงโหนดเดียวเท่านั้นที่ส่ง ARP Response กลับไปที่ผู้ส่ง

รูปที่ 15

แสดงรูปแบบของ ARP โปรโตคอล



รูปที่ 16

แสดงรูปแบบของ ARP Message

hardware type : ประกอบด้วย 2 ไบต์ ใช้กำหนดลักษณะการเชื่อมต่อ ถ้าเป็นอีเทอร์เน็ตจะมีค่าเท่ากับ 1

protocol type - ประกอบด้วย 2 ไบต์ ใช้กำหนดชนิดของโปรโตคอล

IP = 0800 ฐาน 16

ARP= 0806 ฐาน 16

hlen - มีขนาด 1 ไบต์ เก็บค่าความยาวของฮาร์ดแวร์แอดเดรส

plen - มีขนาด 1 ไบต์ เก็บค่าความยาวของโปรโตคอล

operation - มีขนาด 2 ไบต์

1 = ARP Request

2 = ARP Response

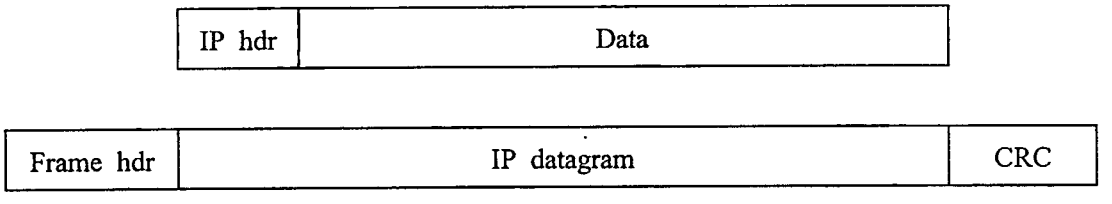
3 = RARP Request

4 = RARP Response

sender hardware address	- มีขนาด 6 ไบต์ เก็บฮาร์ดแวร์แอดเดรสของผู้ส่ง
sender IP address	- มีขนาด 4 ไบต์ เก็บ IP แอดเดรสของผู้ส่ง
target hardware address	- มีขนาด 6 ไบต์ เก็บฮาร์ดแวร์แอดเดรสของผู้รับปลายทาง
target IP address	- มีขนาด 4 ไบต์ เก็บ IP แอดเดรสของผู้รับปลายทาง

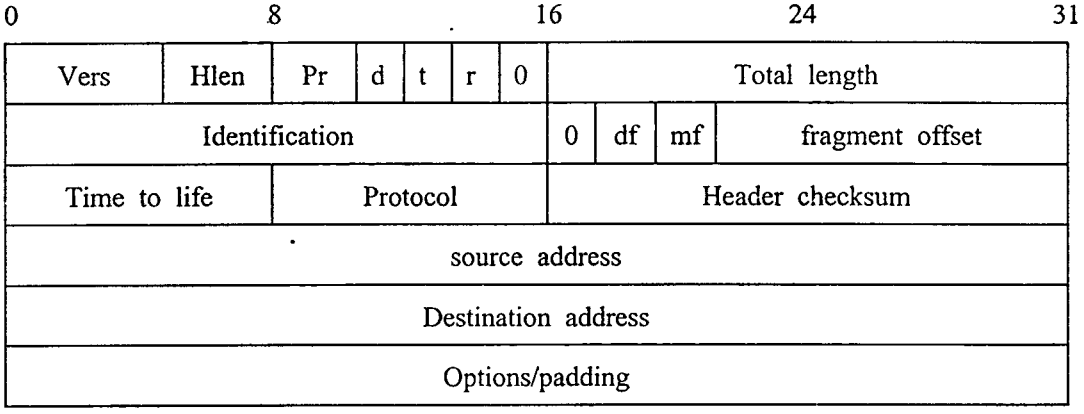
IP โปรโตคอล

ลักษณะการบรรจุดาต้าแกรมของ IP แสดงดังรูปที่ 17 และ IP Header แสดงดังรูปที่ 18.



รูปที่ 17

รูปแบบของ IP โปรโตคอล



รูปที่ 18

รูปแบบของ IP header

vers - มีขนาด 4 บิต เป็นเวอร์ชั่นของอินเตอร์เน็ตโปรโตคอลซึ่งเท่ากับ 4

hlen - มีขนาด 4 บิต เก็บขนาดของ Datagram header

type of service - มีขนาด 1 ไบต์ ประกอบด้วย

. pr - 3 บิต (precedence)

d - 1 บิต (delay)

t - 1 บิต (throughput)

r - 1 บิต (reliability)

reserved - 2 บิต มีค่าเป็น 0

total length - มีขนาด 2 ไบต์ เก็บความยาวของดาต้าแกรมรวมทั้ง header และ ข้อมูล

identification - มีขนาด 2 ไบต์ ใช้กำหนดลำดับของเฟรมในกรณีที่มีหลาย fragment

2 ไบต์ ต่อไปใช้สำหรับการ fragmentation

บิตแรก - reserve

df - 0 = may fragment

1 = don't fragment

mf - 0 = fragment สุดท้าย

1 = more fragment

13 บิต ที่เหลือเป็นออฟเซ็ตของ fragment

time to live - มีขนาด 1 ไบต์ กำหนดจำนวนเวลาสูงสุดที่จะให้ datagram คงอยู่ในระบบ

protocol - มีขนาด 1 ไบต์ กำหนดชนิดของข้อมูลใน datagram

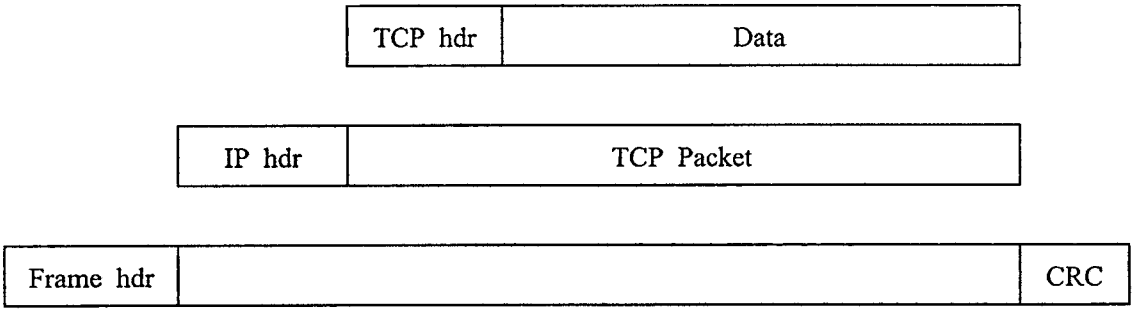
1 = ICMP

6 = TCP

17 = UDP

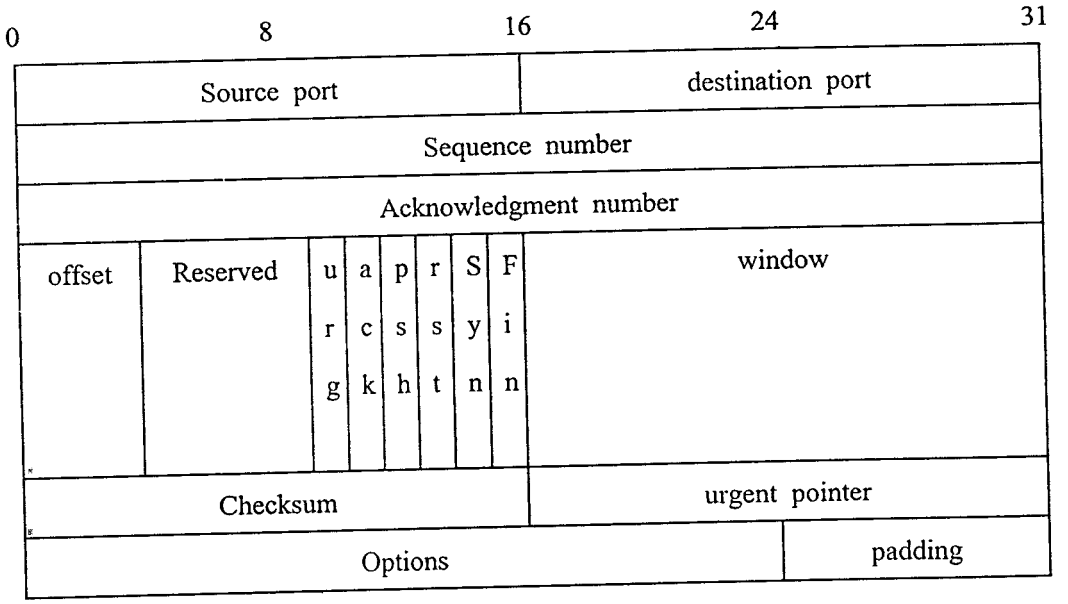
TCP โปรโตคอล

ลักษณะการบรรจุแพ็คเก็ตของ TCP (Transmission Control Protocol) แสดงดังรูปที่ 19 และ IP Header แสดงดังรูปที่ 20.



รูปที่ 19

แสดงรูปแบบของ TCP โปรโตคอล



รูปที่ 20

แสดงรูปแบบของ TCP header

source port - มีขนาด 2 ไบต์ ใช้กำหนดหมายเลขพอร์ตของสถานีต้นทาง

destination port - มีขนาด 2 ไบต์ ใช้กำหนดหมายเลขพอร์ตของสถานีปลายทาง

sequence number - มีขนาด 4 ไบต์ เก็บตำแหน่งของข้อมูลเริ่มต้นในแพ็คเก็ตของผู้ส่ง

acknowledgment number - มีขนาด 4 ไบต์ เก็บค่าของ sequence number อันต่อไปที่คาดว่าจะได้รับ

offset - มีขนาด 4 บิต เก็บความยาวของ header ในเทอมของ WORD (32 บิต)

reserved - มีขนาด 6 บิต มีค่าเป็น 0

control - 6 บิต

urg - Urgent Pointer

ack - Acknowledgment

psh - Push function

rst - Reset

syn - Synchronize Sequence number

fin - หมดข้อมูลจากผู้ส่ง

window - มีขนาด 2 ไบต์ ใช้กำหนดขนาดของข้อมูลที่ผู้รับสามารถรับได้

checksum - มีขนาด 2 ไบต์ ใช้ตรวจสอบความถูกต้องของข้อมูล

urgent pointer - มีขนาด 2 ไบต์ ใช้กำหนดตำแหน่งถัดไปของข้อมูลหลังจากบิต urgent ถูกเซ็ต

options - มีขนาดเท่าใดขึ้นอยู่กับชนิดของ TCP เซ็กเมนต์ มีขนาดตั้งแต่ 1-4 ไบต์

padding - มีขนาดเท่าใดก็ได้ขึ้นอยู่กับฟิลด์ options เพื่อที่จะเติมให้เต็ม 4 ไบต์

Telnet โปรโตคอล

Telnet โปรโตคอลเป็นโปรโตคอลบน TCP/IP ใช้สำหรับสร้าง TCP connection ระหว่างเครื่อง client ไปหา server เมื่อมีการส่ง key ใดๆจาก client เทอร์มินัลก็จะส่งไปที่ server และถ้ามี output ใดๆจาก server ก็จะอาศัย connection นี้ส่งไปที่ client เช่นกัน

เนื่องจาก Telnet โปรโตคอลเป็นการสร้าง connection ระหว่างเครื่องหนึ่งไปยังอีกเครื่องหนึ่ง ซึ่งเครื่องคอมพิวเตอร์แต่ละเครื่องจะเข้าใจความหมายของตัวอักษรหรือ key ใดๆ ต่างกันขึ้นอยู่กับคอมพิวเตอร์เครื่องนั้นและระบบปฏิบัติการ ดังนั้นเพื่อให้เป็นมาตรฐานระหว่างกัน จึงมีการกำหนดรูปแบบของการส่งผ่านอักษรที่จะใช้ในการควบคุมฝ่ายผู้รับ ยกตัวอย่าง เช่น Key CONTROL – C เป็น Key interrupt ใช้สำหรับการสั่งให้โปรแกรมนั้นหยุดการทำงาน รูปแบบของรหัสควบคุมแสดงในตารางที่ 2.

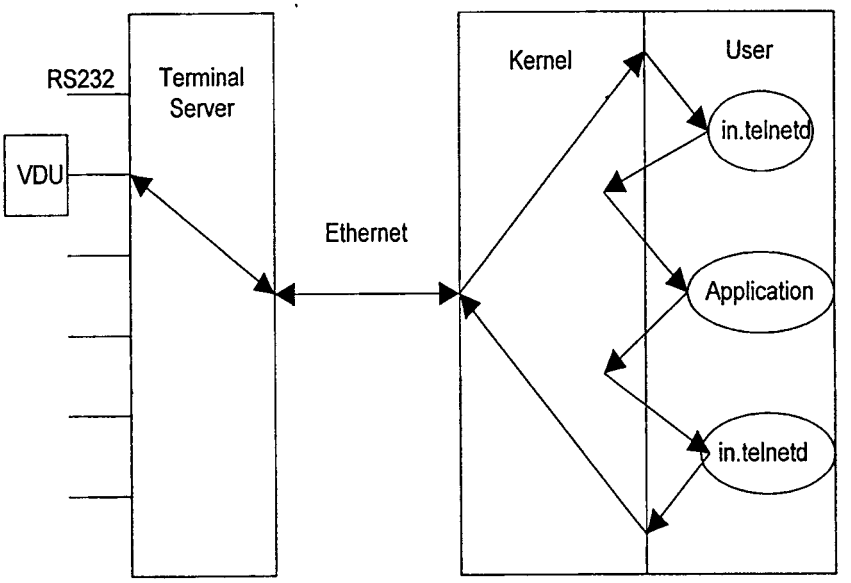
ตารางที่ 2

ตาราง Command code ของ Telnet

Command	Decimal Encoding	Meaning
IAC	255	บ่งบอกว่าตัวอักษรถัดไปเป็นรหัสควบคุม (ถ้าต้องการส่ง IAC ที่เป็นข้อมูล จะต้องส่ง IAC 2 ตัวติดต่อกัน คือส่ง IAC-IAC)
DON'T	254	ปฏิเสธ Request ที่จะกระทำใดๆ
DO	253	อนุญาตที่จะให้กระทำใดๆ
WON'T	252	ปฏิเสธการทำงานใดๆ
WILL	251	เห็นด้วยที่จะกระทำใดๆ
SB	250	เริ่มต้น Option Subnegotiation
GA	249	"go ahead" signal
EL	248	"erase line" signal
EC	247	"erase character" signal
AYT	246	"are you there" signal
AO	245	"abort output" signal
IP	244	"interrupt process" signal
BRK	243	"break" signal
DMARK	242	ส่วนของ SYNCH ใน data stream
NOP	241	No operation
SE	240	จบการทำงาน Option Subnegotiation
EOR	239	End of record

หลักการทำงานของเทอร์มินัลเซิฟเวอร์

การเชื่อมต่อเทอร์มินัลจำนวนมากในเน็ตเวิร์คนิยมใช้เทอร์มินัลเซิฟเวอร์ ซึ่งอาศัยโปรโตคอล telnet ทำการรับส่งอีเทอร์เน็ตแพ็คเก็ตระหว่าง telnet เซสชั่น ข้อมูลจะถูกส่งและรับโดย telnet daemon ที่โฮสของยูสเซอร์นั้นๆ การเรียกใช้บริการ telnet daemon ของโฮสทำได้โดยส่ง TCP แพ็คเก็ตและใช้พอร์ต TCP หมายเลข 23 ส่งไปที่โฮส การทำงานระหว่างเทอร์มินัลเซิฟเวอร์กับโฮส แสดงดังรูปที่ 21.



รูปที่ 21
แสดงการทำงานของเทอร์มินัลเซิฟเวอร์

1. ส่งอีเทอร์เน็ตแพ็คเก็ตจากเทอร์มินัลเซิฟเวอร์ไปที่โฮส
เทอร์มินัลเซิฟเวอร์รับข้อมูลจากพอร์ตอนุกรม และส่งไปในแพ็คเก็ต telnet บนอีเทอร์เน็ตเฟรม
2. ส่งแพ็คเก็ตไปที่ telnet daemon ของแต่ละยูสเซอร์บนเซิฟเวอร์

อีเทอร์เน็ตไดร์ฟเวอร์ส่งผ่านแพ็คเก็ตไปบน TCP/IP โปรโตคอลสแต็กถึง telnet daemon โปรเซส

3. ส่งข้อมูลไปที่โปรแกรมประยุกต์

telnet daemon ทำการแยกข้อมูลที่ส่งมาส่งต่อไปให้โปรแกรมประยุกต์โดยผ่านเทอร์มินัลเทียม (pseudo-terminal) และเมื่อโปรแกรมประยุกต์ทำงาน จะส่งอักขระใดๆ ต่อกลับมาที่เทอร์มินัลเทียม

4. ส่งอักขระเอาท์พุทไปที่ telnet daemon

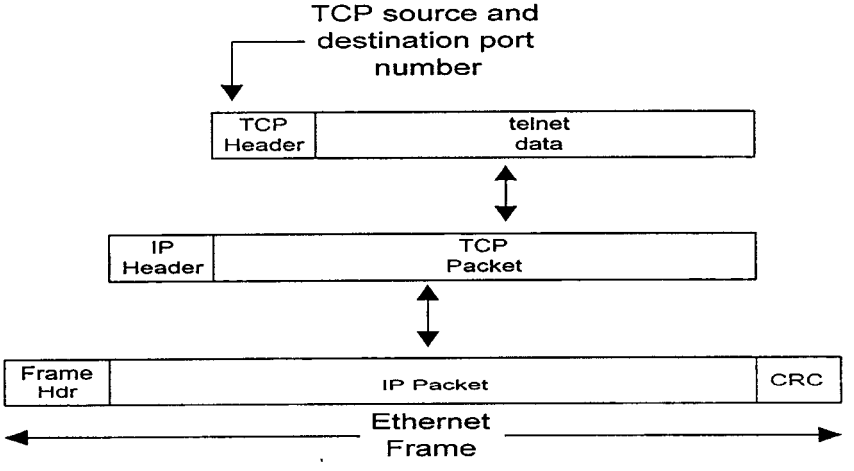
telnet daemon รับเอาท์พุทจากเทอร์มินัลเทียม

5. ส่งแพ็คเก็ต telnet จากโฮสกลับไปยังเทอร์มินัลเซิฟเวอร์

telnet daemon ทำการส่งแพ็คเก็ต telnet ผ่านไปยังอีเทอร์เน็ตเฟรม แล้วส่งกลับไปที่เทอร์มินัลเซิฟเวอร์ที่พอร์ตอนุกรมนั้นๆ

แพ็คเก็ตของ telnet ที่ส่งจะถูกบรรจุใน TCP แพ็คเก็ต โดยจะอาศัยหมายเลขพอร์ตของ source และ destination ของ TCP ที่เก็บอยู่ใน header ของแพ็คเก็ตในการแยกการติดต่อระหว่างพอร์ตอนุกรมบนเทอร์มินัลเซิฟเวอร์

TCP แพ็คเก็ตจะถูกบรรจุอยู่ใน IP ดาต้าแกรม และทั้งหมดนี้จะถูกรวมเข้ามาบรรจุในอีเทอร์เน็ตแพ็คเก็ตดังรูปที่ 22.



รูปที่ 22
แสดงลักษณะการบรรจุแพ็คเก็ตแต่ละชั้น

บทที่ 3
แพ็คเก็ตไดร์ฟเวอร์

อีเทอร์เน็ตเป็นโปรโตคอลที่องค์กรส่วนใหญ่เลือกใช้ในระบบเน็ตเวิร์ค การเขียนโปรแกรมประยุกต์ที่จะติดต่อผ่านทางอีเทอร์เน็ตจะต้องผ่านทางไดร์ฟเวอร์เฉพาะของการ์ดนั้น หมายถึง โปรแกรมประยุกต์เดียวกันจะไม่สามารถนำไปใช้กับการ์ดอีกชนิดหนึ่งได้ ทำให้มีปัญหาในการบำรุงรักษาระบบเมื่อมีการเปลี่ยนแปลงการ์ด เนื่องจากจะต้องแก้ไขระบบโปรแกรมแพ็คเก็ตไดร์ฟเวอร์เป็นวิธีหนึ่งในการแก้ปัญหา แพ็คเก็ตไดร์ฟเวอร์จะจัดเตรียมฟังก์ชั่นมาตรฐานในการเชื่อมต่อระหว่างโปรแกรมประยุกต์กับการ์ด ทำให้โปรแกรมประยุกต์สามารถทำงานได้บนเน็ตเวิร์คการ์ดต่างชนิดได้โดยที่ไม่ต้องมีการแก้ไขโปรแกรมใดๆ

แพ็คเก็ตไดร์ฟเวอร์จะจัดเตรียมฟังก์ชั่นไว้สำหรับให้โปรแกรมประยุกต์เข้าเรียกใช้ในระดับ Link level เพื่อให้เป็นอิสระไม่ขึ้นอยู่กับชนิดของฮาร์ดแวร์ และรูปแบบการเชื่อมต่อกันในเน็ตเวิร์ค แพ็คเก็ตไดร์ฟเวอร์จะเตรียมฟังก์ชั่นสำหรับการกำหนดชนิดของแพ็คเก็ต , ฟังก์ชั่นจบการทำงาน, ฟังก์ชั่นส่งแพ็คเก็ต , ฟังก์ชั่นเก็บสถิติบนเน็ตเวิร์ค และฟังก์ชั่นเรียกดูรายละเอียดเกี่ยวกับการเชื่อมต่อ

ระดับการให้บริการของ แพ็คเก็ตไดร์ฟเวอร์ ที่จะอธิบายนี้ ระดับแรกจะเป็นแพ็คเก็ตไดร์ฟเวอร์ระดับพื้นฐาน (Basic) ซึ่งมีฟังก์ชั่นให้ใช้น้อยกว่าแต่ใช้ง่ายและใช้เนื้อที่หน่วยความจำน้อยกว่า จะเป็นฟังก์ชั่นเกี่ยวกับการส่งและรับแพ็คเก็ต ส่วนไดร์ฟเวอร์ส่วนที่สองจะเป็นซุปเปอร์เซ็ตของระดับแรก จะมีฟังก์ชั่นเกี่ยวกับการเก็บสถิติการใช้เน็ตเวิร์ค ส่วนระดับสาม (High-performance) จะเป็นฟังก์ชั่นเกี่ยวกับการเพิ่มประสิทธิภาพการทำงานและการจูนนิ่ง ซึ่งในที่นี้จะอธิบายเฉพาะฟังก์ชั่นระดับพื้นฐานเท่านั้น

1. การกำหนดชนิดการเชื่อมต่อของเน็ตเวิร์ค

ชนิดของการเชื่อมต่อของเน็ตเวิร์คประกอบด้วยคำ 3 คำ คือ Class , Type , Number Class ของ interface บ่งบอกชนิดของ media ได้แก่ DEC/Intel/Xerox (DIX) , Ethernet , IEEE 802.3 Ethernet , IEEE 802.5 Token Ring , ProNet-10 , Appletalk , Serial line เป็นต้น

ส่วนค่าที่สองเป็นชนิดของการ์ด ได้แก่ 3COM 3C503 หรือ 3C505 , Interlan NI5210 , Univation , BICC Data Networks ISOLan , Ungermann-Bass NIC เป็นต้น

ค่าสุดท้ายเป็นค่าของ interface number ถ้าเครื่องคอมพิวเตอร์ประกอบด้วยการ์ด interface มากกว่าหนึ่งการ์ด

ค่าต่างๆของ class และ type ดูได้จากภาคผนวก

2. การเริ่มต้นทำงานของไดร์ฟเวอร์

แพ็คเก็ตไดร์ฟเวอร์ ถูกเรียกขึ้นมาทำงานโดย Software interrupt ช่วง interrupt ระหว่าง 0x60 ถึง 0x80 ซึ่งค่า interrupt นี้จะสามารถกำหนดได้โดยใส่ค่า argument ในขณะติดตั้ง

เนื่องจากแพ็คเก็ตไดร์ฟเวอร์เป็นโปรแกรมประเภท TSR การที่จะตรวจสอบว่าการติดตั้งไดร์ฟเวอร์สามารถกระทำได้โดยการตรวจสอบที่ address ของ interrupt vector ที่ใช้บวกกับค่า 3 ว่ามี string "PKT DRVR" อยู่หรือไม่ ถ้ามีแสดงว่ามีการติดตั้งไดร์ฟเวอร์ที่ interrupt นั้นๆ แล้ว

3. การเขียนโปรแกรมติดต่อกับ Packet driver

Packet driver จะจัดเตรียมฟังก์ชั่นการทำงานต่างๆไว้โดยทาง Software Interrupt ในระหว่างการติดตั้งเรียบร้อยแล้ว การเรียกใช้แต่ละฟังก์ชั่นจะเลือกผ่านทางรีจิสเตอร์ AH แล้วเรียกไปที่ address ของ Interrupt นั้นๆ

ฟังก์ชั่นต่างๆในระดับพื้นฐานมีดังนี้

- ฟังก์ชั่น driver_info( ) รีจิสเตอร์ AH = 1

ฟังก์ชั่นนี้จะรายงานรายละเอียดของ Driver ได้แก่ เวอร์ชั่นของแพ็คเก็ตไดร์ฟเวอร์, class , type , number ของเน็ตเวิร์คการ์ด

- ฟังก์ชั่น access_type ( ) รีจิสเตอร์ AH = 2

ฟังก์ชั่นนี้ใช้กำหนดชนิดของแพ็คเก็ตที่จะใช้โดยผ่านไปที่ argument type เป็นความยาวของชนิดของแพ็คเก็ต receiver เป็นพอยเตอร์ชี้ไปที่รูทีนย่อย เมื่อมีการรับแพ็คเก็ตถ้า typelen

มีค่าเป็น 0 แสดงว่าต้องการรับแพ็คเก็ตทุกชนิด

เมื่อมีการรับแพ็คเก็ต ตัว Receiver จะถูก call 2 ครั้ง โดยแพ็คเก็ตครั้งแรกจะถูก call เพื่อร้องขอ buffer ในการ copy แพ็คเก็ตเข้าไป ณ ขณะนี้รีจิสเตอร์ AX จะเท่ากับ 0 จากนั้นจะ return ค่าพอยเตอร์ ของ buffer ไปที่รีจิสเตอร์ ES:DI ซึ่งเป็นแอดเดรสของรูทีนที่จะรับแพ็คเก็ต ถ้าไม่มี buffer จะส่งค่า 0:0 ไปที่รีจิสเตอร์ ES:DI และจะไม่ทำการ call ครั้งที่ 2

ในการ call ครั้งที่ 2 (AX= =1) จะทำการ copy ข้อมูลลงใน buffer นั้น จะอยู่ใน address DS:SI

- ฟังก์ชั่น release_type ( ) รีจิสเตอร์ AH = 3

ฟังก์ชั่นนี้ใช้ในการจบหรือยกเลิกการ access แพ็คเก็ตโดยกำหนดจาก handle ที่ได้มาจากฟังก์ชั่น access_type ( )

- ฟังก์ชั่น send_pkt ( ) รีจิสเตอร์ AH = 4

ฟังก์ชั่นนี้ใช้ในการส่งข้อมูลโดยเริ่มต้นที่ address ที่ buffer ชี้ไป ข้อมูลนี้จะรวมถึง header ต่างๆ และ mac หรือ LLC Information ด้วย จะต้องกำหนดรวมไปด้วย

- ฟังก์ชั่น terminate ( ) รีจิสเตอร์ AH = 5

ฟังก์ชั่นนี้ใช้ในการยกเลิกการทำงานของไดร์ฟเวอร์และจะทำการคืนทรัพยากรที่ใช้ไป ได้แก่ หน่วยความจำ

สำหรับรายละเอียดต่างๆ สามารถหาดูได้จาก PC/TCP packet Driver Specification[6]

บทที่ 4
การเขียนโปรแกรมติดต่อกับอะแดปเตอร์สื่อสารแบบอนุกรม

หัวใจสำคัญของอะแดปเตอร์การสื่อสารก็คือไอซี 8250 ซึ่งมีทั้งตัวกำหนด Baud Rate ที่สามารถควบคุมได้ในตัว และมีรีจีสเตอร์ที่ใช้ในงานควบคุมไว้ 7 รีจีสเตอร์ด้วยกัน นอกเหนือจากนั้นก็เป็นพวกไอซีซึ่งทำหน้าที่แยกตำแหน่งที่อยู่ของช่องทางออก และ crystal oscillator 1.8432 MHz

ดอสกำหนดหมายเลขของช่องทางเข้าออกสำหรับรีจีสเตอร์ควบคุมของ 8250 และความหมายการใช้งานของรีจีสเตอร์ใน 8250 ดังในตารางที่ 3.

ตารางที่ 3

แสดงช่องทางเข้าออกและการใช้งานของรีจีสเตอร์ใน 8250

ช่องการสื่อสารที่ 1	ช่องการสื่อสารที่ 2	รีจีสเตอร์	การใช้งาน
3F8	2F8	RBR DLL	- ใช้สำหรับที่พักข้อมูลทั้งรับและส่ง - ใช้เป็นตัวหารตัวที่มีค่าน้อยสำหรับหารความถี่ 1.8432 MHz เป็น Baud Rate
3F9	2F9	DLM IER	- ใช้เป็นตัวหารที่มีค่าสูงสำหรับหารความถี่ 1.8432 MHz ให้เป็น Baud Rate - ใช้สำหรับควบคุมการเกิดอินเตอร์รัพต์
3FA	2FA	IIR	- สำหรับบอกชนิดของอินเตอร์รัพต์
3FB	2FB	LCR	- สำหรับควบคุมพารามิเตอร์การรับส่ง - สำหรับการบ่งบอกการใช้งานของ DLL และ DLM

ตารางที่ 3 (ต่อ)

แสดงช่องทางเข้าออกและการใช้งานของรีจีสเตอร์ใน 8250

ช่องการสื่อสารที่ 1	ช่องการสื่อสารที่ 2	รีจีสเตอร์	การใช้งาน
3FC	2FC	MCR	- ใช้สำหรับควบคุมโมเด็ม
3FD	2FD	LSR	- สำหรับบอกสถานภาพของการรับส่ง
3FE	2FE	MSR	- สำหรับบอกสถานภาพของโมเด็ม

สำหรับหมายเลขของทางเข้าออกที่ XF8 และ XF9 ใช้สำหรับ 2 หน้าที่ โดยจะรวมกันเป็นตัวหาร (DLM และ DLL) ถ้าบิตที่ 8 ของ LCR เป็น 1 8250 สุ่มตัวอย่าง (Sampling) สัญญาณที่รับเข้ามา 16 เท่าของอัตราการรับส่งข้อมูล วงจรออสซิลเลเตอร์ในอะแดปเตอร์เองทำความถี่ 1.8432 MHz ป้อนให้กับ 8250 การกำหนดตัวหารเพื่อให้เกิดเป็น Baud Rate ที่ต้องการแสดงอยู่ในตารางที่ 4 ถ้าหากบิตที่ 8 ของ LCR เป็น 0

XF8 จะทำหน้าที่เป็นที่พักของข้อมูลทั้งรับและส่ง (2 รีจีสเตอร์ใช้หมายเลขของช่องทางเข้าออกอันเดียวกัน) ส่วน XF9 จะใช้สำหรับควบคุมการเกิดอินเตอร์รัพต์ โดยมีความหมายของการควบคุมดังแสดงในตารางที่ 5.

ตารางที่ 4

ตัวหาร 1.8432 MHz เพื่อให้ได้ความถี่ 16 เท่าของ Baud Rate

Baud	ตัวหาร	
	ฐาน 10	ฐาน 16
300	384	180
600	192	0C0
1200	96	060
2400	48	030
4800	24	018
9600	12	00C
19200	6	6
38400	3	3
57600	2	2
115200	1	1

ตารางที่ 5

การคุมการกำเนิดสัญญาณอินเตอร์รัพต์ โดยการเขียนข้อมูลเข้าไปที่ช่องทางเข้าออกหมายเลข FX9

บิตที่	0	สร้างสัญญาณอินเตอร์รัพต์เมื่อรับข้อมูลได้ครบ อักขระ
บิตที่	1	สร้างสัญญาณอินเตอร์รัพต์เมื่อรีจีสเตอร์สำหรับส่งว่างลง
บิตที่	2	สร้างสัญญาณอินเตอร์รัพต์เมื่อมีข้อพลาดเกิดขึ้นจากการรับ
บิตที่	3	สร้างสัญญาณอินเตอร์รัพต์เมื่อมีสัญญาณ Modem Status

สำหรับรีจีสเตอร์ IIR ที่หมายเลขทางเข้าออก 3FA หรือ 2FA ใช้สำหรับบอกว่าอินเตอร์รัพต์ที่เกิดขึ้นเป็นอินเตอร์รัพต์ชนิดไหน ใน 4 ชนิดที่เราสามารถกำหนดให้เกิดได้โดยรีจีสเตอร์ IER ในโปรแกรมสำหรับจัดการอินเตอร์รัพต์จะต้องมาตรวจสอบรีจีสเตอร์ตัวนี้ก่อนเพื่อที่จะได้จัดการได้ถูกต้องว่าเกิดอินเตอร์รัพต์เพราะเหตุใด การอ่านค่าจากหมายเลขช่องทางเข้าออกนี้จะบอกได้โดยบิตที่ 0 จะเป็นตัวบอกว่าสัญญาณอินเตอร์รัพต์ยังอยู่หรือหายไป (0 ยังอยู่ 1 หายไป) ถ้าหากเกิดมีอินเตอร์-รัพต์หลายชนิดซ้อนกัน ชนิดที่มีความสำคัญที่สุดจะปรากฏในรีจีสเตอร์ดังนี้ ตารางที่ 6 บอกชนิดอินเตอร์รัพต์ที่เกิด พร้อมทั้งการกระทำที่จะทำให้สัญญาณอินเตอร์รัพต์หายไป

ตารางที่ 6

รีจีสเตอร์ IIR แสดงชนิดของอินเตอร์รัพต์ที่เกิดขึ้น

ลำดับความสำคัญ	ค่าที่อ่านได้ใน IIR	สาเหตุของการอินเตอร์รัพต์	ขบวนการทำให้สัญญาณอินเตอร์รัพต์หาย
1	0000110	- เกิดการเขียนทับข้อมูลเก่า - ตรวจสอบ ผิดพลาด - ผิดพลาดของช่วงอักขระ	อ่านรีจีสเตอร์ LSR ที่หมายเลขช่องทาง XFD
2	0000100	- เมื่อรับข้อมูลครบอักขระ	อ่านข้อมูลจาก RBR ที่หมายเลข XF8
ลำดับความสำคัญ	ค่าที่อ่านได้ใน IIR	สาเหตุของการอินเตอร์รัพต์	ขบวนการทำให้สัญญาณอินเตอร์รัพต์หาย
3	0000010	- เมื่อส่งข้อมูลออกไปหมด	เขียนข้อมูลลง RBR

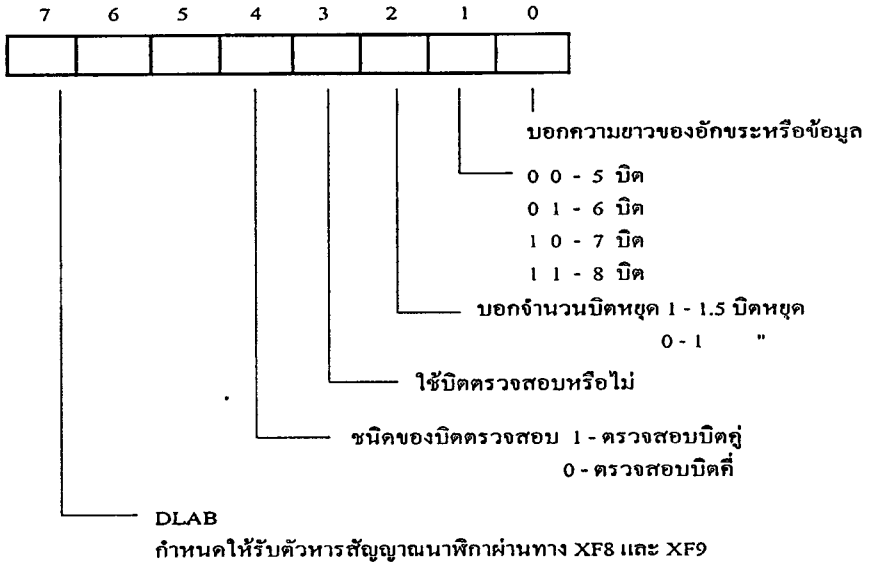
สาเหตุของการเกิดข้อผิดพลาดในการรับ เกิดขึ้นได้ 3 แบบ

1. เกิดการเขียนทับข้อมูลเก่าคืออ่านข้อมูลออกจาก RBR (หรือรีจีสเตอร์พักข้อมูล) ไม่ทันที่ของใหม่เขียนทับเข้ามา ข้อมูลเก่าจะหายไปเมื่อเกิดเหตุเช่นนี้ สถานภาพของการรับส่งใน LSR รีจีสเตอร์จะเป็นตัวบอกดังแสดงในรูปที่ 25 พร้อมกับเกิดสัญญาณอินเตอร์รัพต์ไปบอกซีพียู ถ้าเรายอมให้เกิดอินเตอร์รัพต์ชนิดนี้ (ซึ่งกำหนดได้โดยรีจีสเตอร์ LCR ดังที่กล่าวมาแล้ว)

2. เกิดการตรวจสอบผิดพลาดในการส่งข้อมูล เราสามารถแทรกบิตตรวจสอบเอาไว้ทุกอักขระที่ส่ง ถ้าหากฝ่ายรับตรวจสอบแล้วไม่ตรงตามที่ส่งมาฝ่ายรับจะถือว่าเกิดการผิดพลาด LSR รีจีสเตอร์จะเป็นผู้บ่งชี้และเกิดสัญญาณอินเตอร์รัพต์ทำนองเดียวกับการผิดพลาดในแบบที่ 1

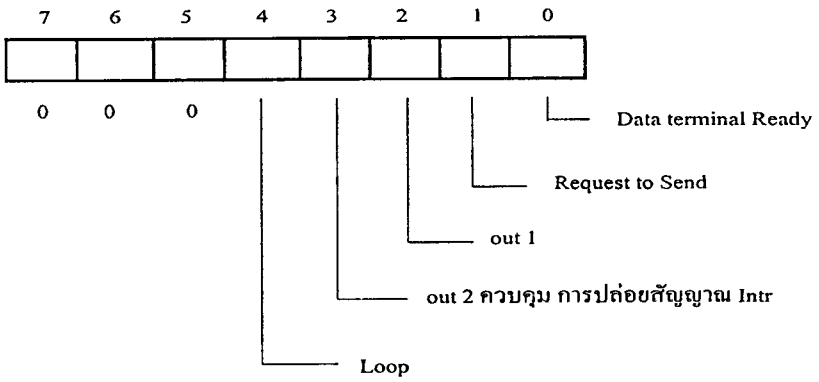
3. เมื่อเกิดการผิดพลาดในการหาขอบเขตของอักขระเรียกว่า Framing Error คือหาบิตเริ่มและบิตหยุดไม่พบ

การควบคุมพารามิเตอร์ของการรับส่งสามารถทำได้โดยการกำหนดค่าลงไป LCR รีจีสเตอร์ซึ่งอยู่ที่หมายเลขช่องทาง XFB โดยค่าใน XFB มีความหมายดังในรูปที่ 23

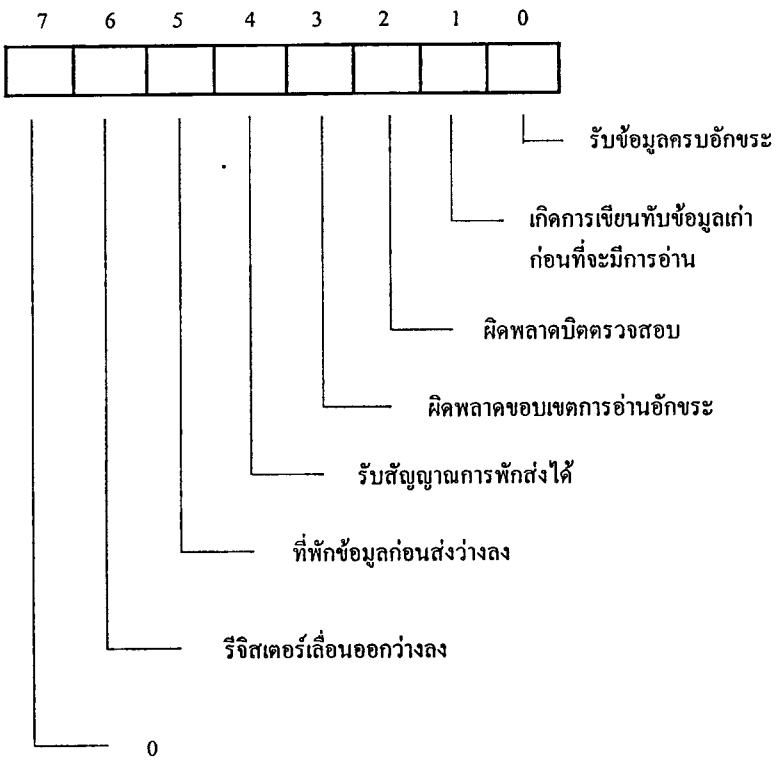


รูปที่ 23

แสดงการตั้งค่าพารามิเตอร์ของการสื่อสารที่หมายเลขช่องทางเข้าออก XFB



รูปที่ 24
แสดงการตั้งค่าควบคุมโมเด็ม XFC



รูปที่ 25
สถานภาพของการรับข้อมูล (หมายเลขช่องทางเข้าออก XFD)

การเขียนโปรแกรมติดต่อกับพอร์ตอนุกรมที่ 1 และ 2 จะต้องมีการกำหนดความเร็วของพอร์ตอนุกรมก่อน โดยแสดงเป็นภาษาซีได้ดังนี้

```
stat_com=inportb(0x3fb);          /* อ่านค่าพารามิเตอร์สำหรับควบคุมการ
                                     รับส่งที่ช่องทางเข้าออก XFB */
outportb(0x3fb,stat_com | 128);   /* เซ็ตบิตที่ 7 ของช่องทางเข้าออก XFB
                                     เพื่อควบคุมช่องทางเข้าออก XF8 และ XF9
                                     ในการกำหนดความเร็วของพอร์ตอนุกรม*/
outportb(0x3f8,Speed);            /* กำหนดค่าตัวหารจากตารางที่ 5 เพื่อกำหนด
outportb(0x3f9,00);                  ความเร็วของพอร์ตอนุกรม */
outportb(0x3fb,stat_com);         /* รีเซ็ตค่าเดิมของช่องทางเข้าออก XFB */
```

เมื่อกำหนดความเร็วของพอร์ตอนุกรมเรียบร้อยแล้ว การส่งข้อมูลจะส่งทางช่องทางเข้าออก XF8 ซึ่งจะต้องเช็คบิตที่ 5 ของช่องทางเข้าออก XFD ก่อนดังนี้

```
while((inportb(0x3fd)&32)!=32);
    outportb(0x3fb,str[i]);
```

เช่นเดียวกัน การรับข้อมูลก็ต้องรับทางช่องทางเข้าออก XF8 ซึ่งจะต้องเช็คบิตที่ 0 ของช่องทางเข้าออก XFD ก่อนดังนี้

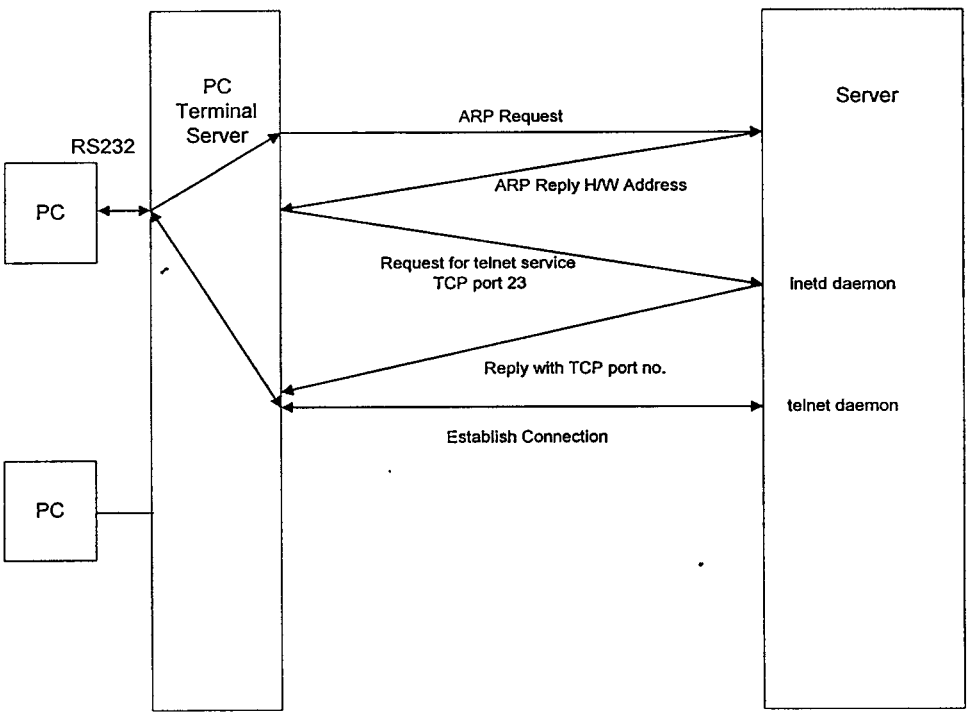
```
if((inportb(0x3fd)&1)==1){
    port_in1=inportb(0x3fb);
```

บทที่ 5

การดำเนินงาน

โปรแกรมบริการเทอร์มินัลระบบยูนิกซ์บนเครื่องคอมคอมพิวเตอร์ส่วนบุคคลนี้ พัฒนาโดยใช้ภาษาซี (Turbo C รุ่นที่ 2) ซึ่งใช้ในการติดต่อกับเน็ตเวิร์คการ์ดโดยผ่านแพ็คเก็ตไดร์ฟเวอร์

<u>การทำงานของโปรแกรม</u>



รูปที่ 26

แสดงการเริ่มต้นสร้างการติดต่อระหว่างเทอร์มินัลเซิฟเวอร์กับโฮส

จากรูปที่ 26 การทำงานของโปรแกรมจะทำงานดังนี้

- รับอินพุทจากพอร์ตอนุกรมซึ่งเป็น IP address ของเซิฟเวอร์แล้วจึงส่ง ARP request แพ็คเก็ตไปที่โฮสเพื่อหาฮาร์ดแวร์แอดเดรสในการส่ง TCP แพ็คเก็ต

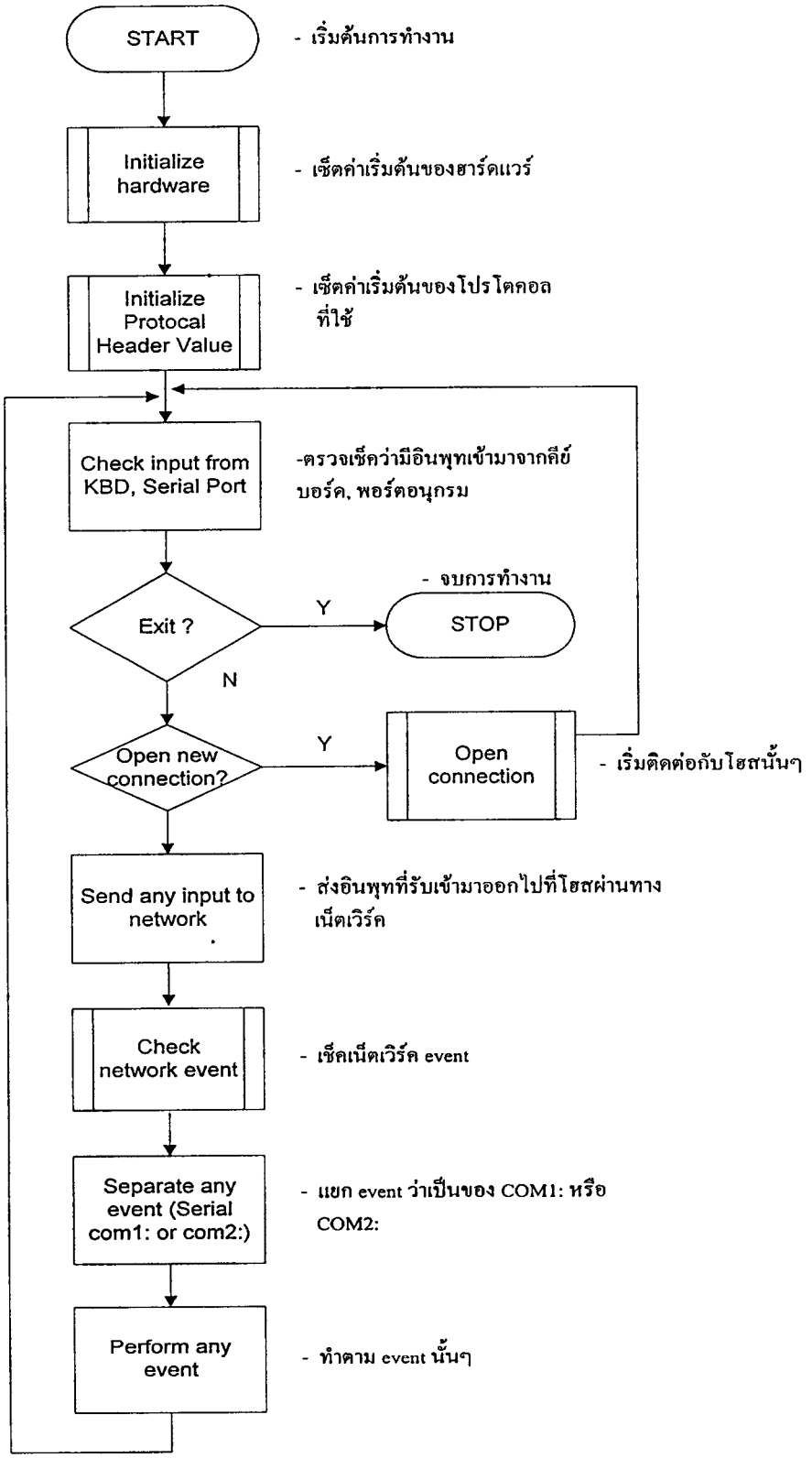
- ใช้ฮาร์ดแวร์แอดเดรสทำการส่งแพ็คเก็ตเพื่อเรียกบริการ telnet จากโฮสโดยกำหนดพอร์ต TCP ต้นทางให้กับพอร์ตอนุกรมที่ 1 เป็น 0, พอร์ตอนุกรมที่ 2 เป็น 1 และใช้พอร์ต TCP หมายเลขที่ 23 เป็นพอร์ตปลายทาง

- รับหมายเลขพอร์ต TCP จากแพ็คเก็ต TCP reply ของโฮส จากนั้นการติดต่อจึงเริ่มต้นขึ้น (Connection Establish)

1. โปรแกรมหลัก จะประกอบด้วยโปรแกรมย่อยต่างๆ ดังนี้

- โปรแกรมย่อย INITIALIZE HARDWARE
- โปรแกรมย่อย INITIALIZE PROTOCOL HEADER VALUE
- โปรแกรมย่อย OPEN CONNECTION
- โปรแกรมย่อย CHECK NETWORK EVENT

ขั้นตอนการทำงานของโปรแกรมหลักสามารถแสดงได้ตามขั้นตอนต่างๆดัง Flow chart



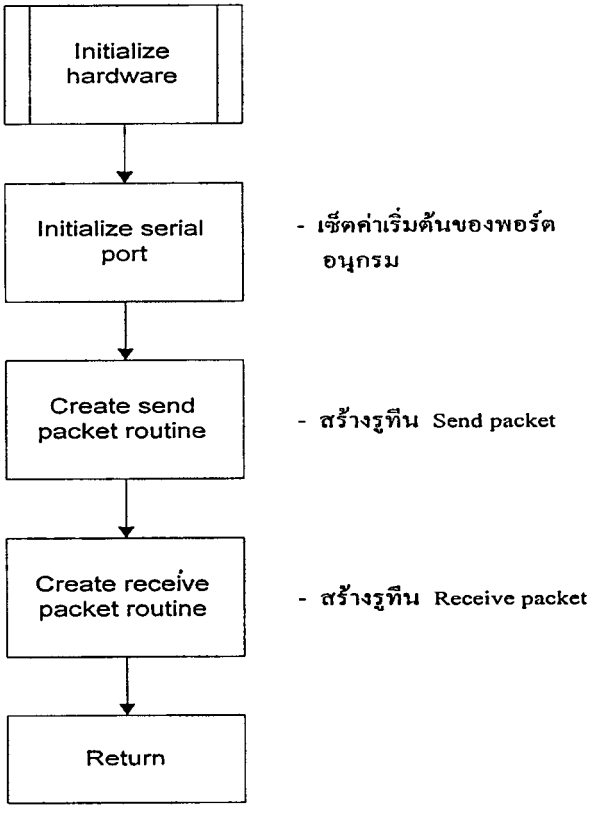
รูปที่ 27

แสดงการทำงานของโปรแกรมหลัก

จากรูปที่ 27 โปรแกรมจะทำการเซ็ตค่าเริ่มต้นของฮาร์ดแวร์ต่างๆ เช่น การสร้างรูทีนสำหรับให้บริการต่างๆ, เซ็ตค่า IP แอดเดรส, อีเทอร์เน็ตแอดเดรส และเซ็ตค่าเริ่มต้น header ของโปรโตคอลที่จำเป็นในการติดต่อ แล้วจึงจะทำการวนลูปเช็คอินพุทจากพอร์ตอนุกรมทั้งสองว่ามีอักขระส่งมาหรือไม่ ถ้ามีก็ส่งอักขระนั้นไปยังโฮสผ่านทางเน็ตเวิร์ค และเช่นกันถ้ามี event หรือเหตุการณ์ใดๆเกิดขึ้นในเน็ตเวิร์คก็จะส่งเหตุการณ์นั้นไปที่พอร์ตอนุกรมนั้นๆ เหตุการณ์ที่เกิดขึ้นนี้ ได้แก่ การ close connection, การมีอักขระส่งมาทางเน็ตเวิร์ค เป็นต้น

2. โปรแกรมย่อย INITIALIZE HARDWARE

แสดงได้ตามขั้นตอนต่างๆดัง Flow chart รูปที่ 28.



รูปที่ 28

แสดงการทำงานของโปรแกรมย่อย INITIALIZE HARDWARE

ขั้นตอนนี้จะทำการเซ็ตค่าเริ่มต้นของพอร์ตอนุกรมทั้งสอง (วิธีการสามารถดูรายละเอียดได้จากบทที่ 4) และสร้างฟังก์ชั่นต่างๆที่จำเป็นในการรับส่งแพ็คเก็ตผ่านเน็ตเวิร์ค ได้แก่

- send packet (ส่งแพ็คเก็ต)
- receive packet (รับแพ็คเก็ต)

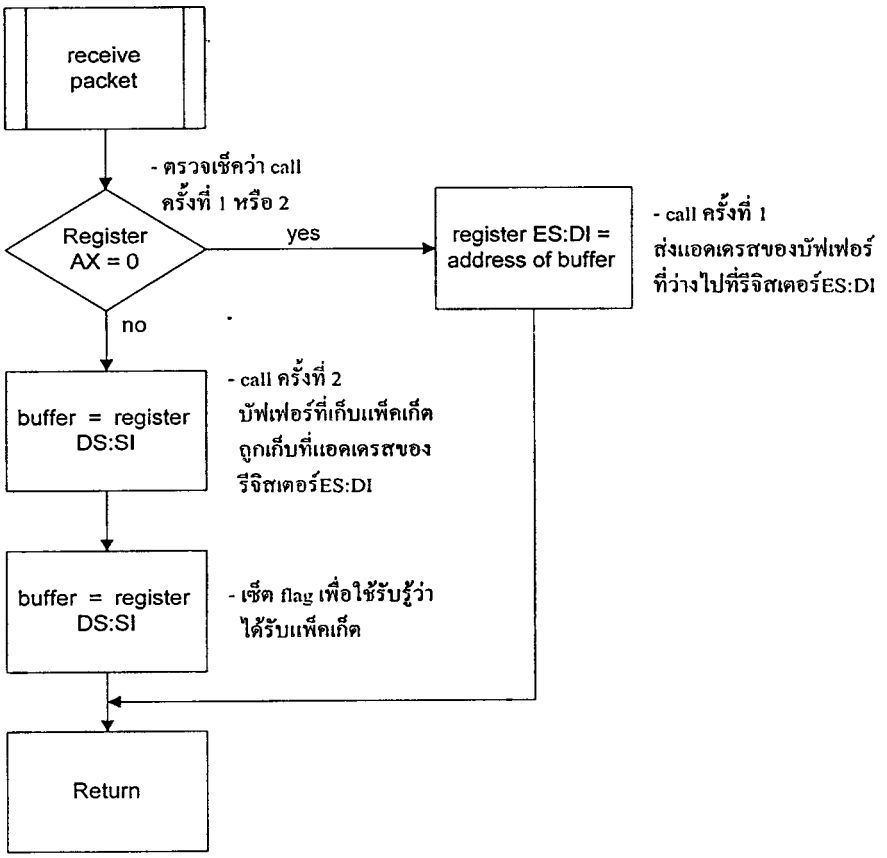
2.1 ฟังก์ชั่น send packet

ฟังก์ชั่นนี้จะใช้ฟังก์ชั่น send_packet() ของแพ็คเก็ตไดร์ฟเวอร์อีกที โดยการเซ็ตรีจิสเตอร์ AH ให้มีค่าเท่ากับ 4 , รีจิสเตอร์ DS:SI มีค่าเท่ากับแอดเดรสของข้อมูลแพ็คเก็ตที่จะส่ง และรีจิสเตอร์ CX มีค่าเท่ากับความยาวของแพ็คเก็ตที่จะส่ง ดังแสดงเป็นภาษาซีดังนี้

```
union REGS regs;
struct SREGS segregs;
        regs.x.ax = 0x0400;
        regs.x.si = FP_OFF(packet);
        regs.x.cx = strlen(packet);
        segregs.ds = FP_SEG(packet);
        int86x(packet_vector,&regs,&regs,&segregs);
```

2.2 ฟังก์ชั่น receive packet

ฟังก์ชั่นนี้จะถูกเรียกโดยแพ็คเก็ตไดร์ฟเวอร์โดยเมื่อแพ็คเก็ตไดร์ฟเวอร์รับแพ็คเก็ตเข้ามาจะเรียกไปที่แอดเดรสของฟังก์ชั่น receive packet ซึ่งฟังก์ชั่น receive packet จะทำงานดัง flow chart รูปที่ 29.



รูปที่ 29

แสดงการเรียกใช้ฟังก์ชั่น receive packet เมื่อแพ็คเก็ตไดร์ฟเวอร์ได้รับแพ็คเก็ต

การที่แพ็คเก็ตไดร์ฟเวอร์จะรับรู้แอดเดรสของฟังก์ชั่น receive packet ทำได้โดยการใช้ฟังก์ชั่น access_type() ของแพ็คเก็ตไดร์ฟเวอร์ ดังแสดงเป็นภาษาซีดังนี้

```
union REGS regs;
struct SREGS segregs;
        regs.x.ax  =  2 * 256 + if_class;     /*  AH = 2 , AL = 0xFF , class ของเน็ตเวิร์ค
                                                  อินเตอร์เฟซการ์ด */
        regs.x.bx  =  if_type;                /*  ชนิดของเน็ตเวิร์คอินเตอร์เฟซการ์ด */
        regs.x.dx  =  if_number;      /*  ลำดับของแพ็คเก็ตไดร์ฟเวอร์  */
        segregs.ds  =  FP_SEG(type);        /*  ชนิดของแพ็คเก็ต  IP = 0800 */
```

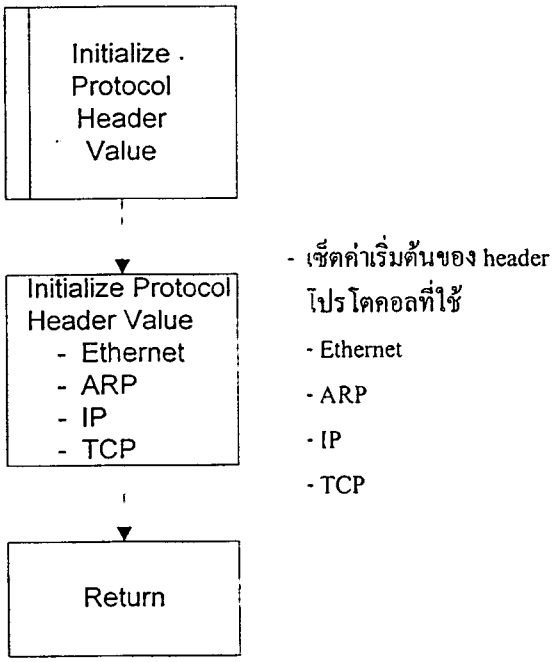
```
regs.x.si = FP_OFF(type);        /*                ARP = 0806 */
regs.x.cx = typelen;             /* มีค่าเป็น 0 หมายถึงรับทุกๆแพ็คเก็ต */
segregs.es = FP_SEG(receiver);   /* แอดเดรสของฟังก์ชั่น receive packet */
regs.x.di = FP_OFF(receiver);
int86x(packet_vector,&regs,&regs,&segregs);
```

สำหรับ class ของเน็ตเวิร์คอินเตอร์เฟซการ์ด, ชนิดของเน็ตเวิร์คอินเตอร์เฟซการ์ด และลำดับของแพ็คเก็ตไดร์ฟเวอร์ สามารถใช้ฟังก์ชั่น driver_info() ของแพ็คเก็ตไดร์ฟเวอร์ในการตรวจสอบ ดังแสดงเป็นภาษาซีดังนี้

```
union REGS regs;
struct SREGS segregs;
    regs.x.ax = 0x01ff;          /* AH = 1, AL = 0xFF */
    regs.x.bx = 0;               /* BX = handle */
    int86x(packet_vector,&regs,&regs,&segregs);
    packet_version = regs.x.bx;
    packet_class = regs.h.ch;
    packet_type = regs.x.dx;
    packet_ifnumber = regs.h.cl;
    packet_name = MK_FP(segs.ds,regs.x.si);
```

3. โปรแกรมย่อย INITIALIZE PROTOCOL HEADER VALUE

แสดงได้ตามขั้นตอนต่างๆดัง Flow chart รูปที่ 30.



รูปที่ 30

แสดงการทำงานของโปรแกรมย่อย INITIALIZE PROTOCOL HEADER VALUE

ขั้นตอนนี้เป็นการเซ็ตค่าเริ่มต้นของ header ของโปรโตคอลต่างๆที่จำเป็นในการรับส่งผ่านเน็ตเวิร์ค ได้แก่

- อีเทอร์เน็ต
- ARP
- IP
- TCP

รูปแบบการกำหนด string ในภาษาซีของโปรโตคอลที่ใช้เป็นดังนี้

3.1 โปรโตคอลอีเทอร์เน็ต

```
struct  ether {
        unsigned char dest[6],          /* ฮาร์ดแวร์แอดเดรสของผู้รับ  */
                      me[6],            /* ฮาร์ดแวร์แอดเดรสของผู้ส่ง  */
        unsigned int type;              /* ชนิดของอีเทอร์เน็ตแพ็คเก็ตมีค่า 0x0008 */
```

```
};
typedef  struct ether DLAYER;
```

3.2 โปรโตคอล ARP

```
struct arp {
        DLAYER d;               /* header แพ็คเก็ตของอีเทอร์เน็ต */
        unsigned int  hrd,      /* ชนิดของฮาร์ดแวร์, อีเทอร์เน็ต = 1 */
                      pro;      /* ชนิดของโปรโตคอล, ARP = 0806 */
        unsigned char hln,      /* ความยาวของฮาร์ดแวร์แอดเดรส, อีเทอร์เน็ต=6*/
                      pln;      /* ความยาวของโปรโตคอล, IP=4 */
        unsigned int op;        /* opcode, ARP request = 1 */
        unsigned char sha,      /* ฮาร์ดแวร์แอดเดรสของผู้ส่ง */
                      sip,      /* IP แอดเดรสของผู้ส่ง */
                      tha,      /* ฮาร์ดแวร์แอดเดรสของผู้รับ */
                      tip;      /* IP แอดเดรสของผู้รับ */
};
typedef struct arp ARPKT;
```

3.3 โปรโตคอล IP

```
struct iph {
        unsigned char versionandhdrlen;     /* version = 4, ความยาวของ  datagram
                                               haeader = 5 (32-bit words) */

        unsiged char service;               /* ชนิดการบริการของ IP */
        unsigned int tlen,                  /* ความยาวของแพ็คเก็ต IP */
                     ident,                 /* identification */
                     frags;                 /* fragmentation */
        unsigned char ttl,                  /* time to live */
```

```
                protocol;          /* ชนิดของข้อมูลใน datagram, TCP = 6 */
    unsigned int check;            /* header checksum = 0 */
    unsigned char ipsource[4],     /* IP addresses */
                  ipdest[4];
};
typedef struct iph IPLAYER;
```

3.4 โปรโตคอล TCP

```
struct tcph {
    unsigned int source,dest;      /* หมายเลขพอร์ต TCP ของสถานีต้นทาง
                                      และปลายทาง */
    unsigned long int seq,ack;     /* sequence, ACK numbers */
    unsigned char hlen,            /* ความยาวของ TCP header = 80 โดยใช้ 4
                                   บิตแรกเก็บ จะได้ความยาว = 5 (32-bits word) */
                  flags;           /* flag ควบคุม */
    unsigned int window,           /* ขนาดของข้อมูลที่ผู้รับสามารถรับได้ */
                 check,            /* ตรวจสอบความถูกต้องของข้อมูล */
                 urgent;           /* กำหนดตำแหน่งถัดไปของข้อมูลหลังจาก
                                      บิต urgent ถูกเซ็ต */
};
typedef struct tcph TCPLAYER;
struct tcp {
    DLAYER d;
    IPLAYER i;
    TCPLAYER t;
    union {
        unsigned char options[4];
        unsigned char data[1024];
    }
```

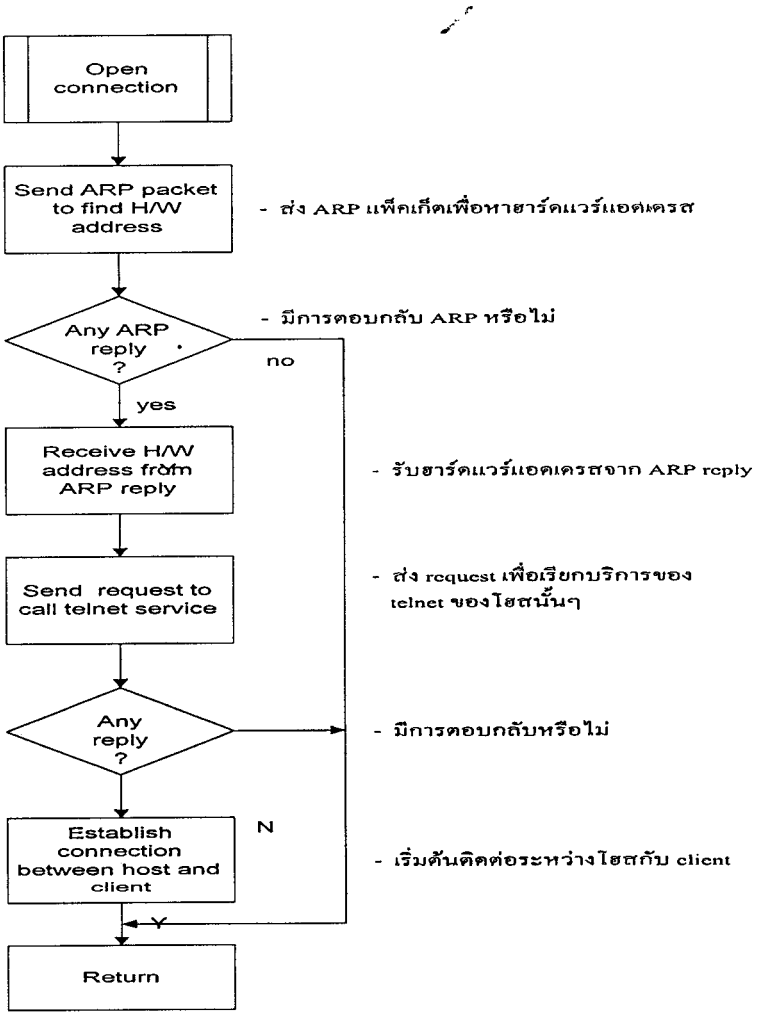
};

typedef struct tcp TCPKT;

สำหรับรายละเอียดของ header ของโปรโตคอลต่างๆ สามารถดูได้บทที่ 2.

4. โปรแกรมย่อย Open connection

แสดงได้ตามขั้นตอนต่างๆดัง Flow chart รูปที่ 31.



รูปที่ 31

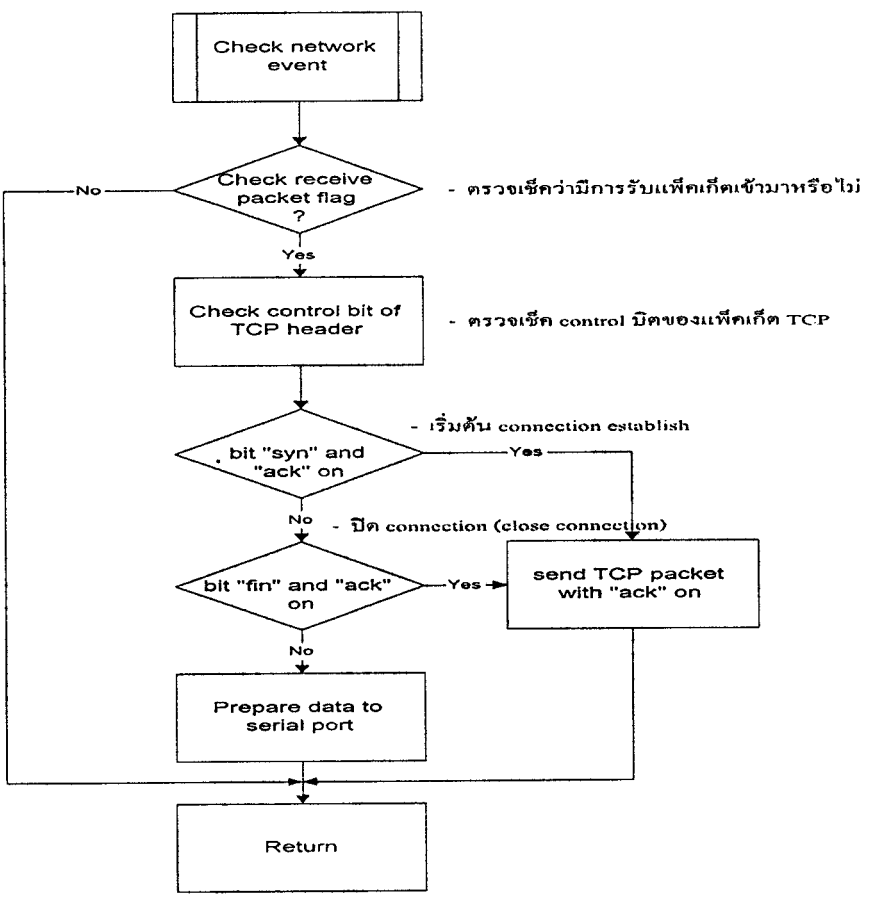
แสดงการทำงานของโปรแกรมย่อย OPEN CONNECTION

ขั้นตอนนี้ทำการส่ง ARP request แพ็คเก็ตไปที่เน็ตเวิร์ค เพื่อหาฮาร์ดแวร์แอดเดรส ของโฮสที่ต้องการจาก ARP reply แพ็คเก็ต หลังจากนั้น จึงส่ง telnet request แพ็คเก็ตโดยใช้ TCP พอร์ตหมายเลข 23 และกำหนด TCP พอร์ตหมายเลข 0 ให้กับพอร์ตอนุกรมที่ 1 , TCP พอร์ต หมายเลข 1 ให้กับพอร์ตอนุกรมที่ 2

เมื่อได้รับแพ็คเก็ตตอบกลับจากโฮส ซึ่งจะได้หมายเลข TCP พอร์ตของโฮส จึงทำการ เริ่มต้นติดต่อกับโฮสนั้นๆ (Establish connection) โดยส่ง แพ็คเก็ต TCP ที่ใช้พอร์ต TCP ที่ได้มา และที่ตั้งค่าไว้ให้กับพอร์ตอนุกรมนั้นๆ พร้อมกับเซ็ตบิตควบคุม syn เป็น "1"

5. โปรแกรมย่อย CHECK NETWORK EVENT

แสดงได้ตามขั้นตอนต่างๆดัง Flow chart รูปที่ 32.



รูปที่ 32

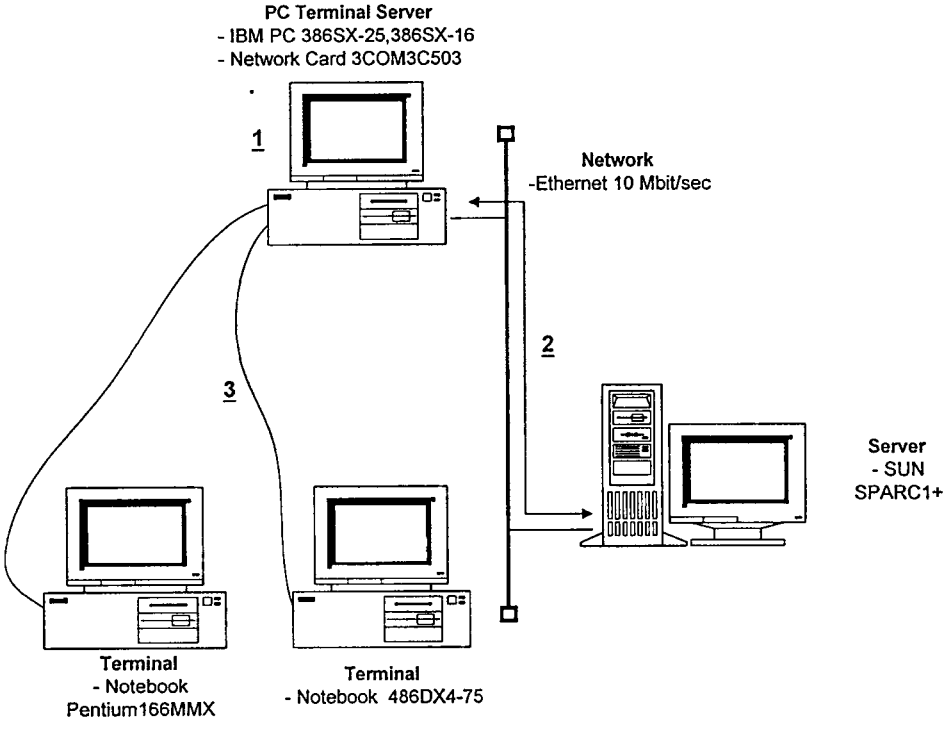
แสดงการทำงานของโปรแกรมย่อย CHECK NETWORK EVENT

จากรูปที่ 32 โปรแกรมย่อย check network event จะทำการเช็คว่ามีอินพุทแพ็คเก็ตเข้ามาหรือไม่ ถ้ามีก็จะตรวจเช็คบิตควบคุมของ TCP Header ว่าตรงตามเงื่อนไขต่างๆ เช่น มีการเริ่มต้นติดต่อกับโฮส (connection establish) หรือยกเลิกการติดต่อ (close connection)

บทที่ 6

การทดสอบ

การพัฒนาและทดสอบการวิจัยครั้งนี้ ใช้เครื่องไมโครคอมพิวเตอร์ IBM PC 386SX-25 compatable และ IBM PC 386SX-16 compatable ทำงานในส่วนของโปรแกรมบริการเทอร์มินัล, เน็ตเวิร์คอินเตอร์เฟซการ์ดของ 3COM 3C503, เครื่องเวิร์คสเตชั่น SUN SPARC 1+ ใช้ระบบปฏิบัติการ Solaris 2.4 ในส่วนของโฮสคอมพิวเตอร์, เครื่องไมโครคอมพิวเตอร์โน๊ตบุ๊ค 80486DX4-75 และ เพนเทียม 166MMX ระบบปฏิบัติการ Microsoft Window 95 ประมวลผลโปรแกรมประยุกต์ประเภทคอมมิวนิเคชั่น Reflection กำหนดคุณสมบัติของพอร์ตอนุกรมให้มี bit rate ต่างๆกันดังนี้ 9600 , 19200 , 38400 , 57600 , 115200 คุณสมบัติของข้อมูลเป็น 8,n,1 (8 บิต,no parity,1 stop บิต) ในส่วนของเทอร์มินัล และเน็ตเวิร์คแบบอีเทอร์เน็ตความเร็ว 10 เมกกะบิต ดังรูปที่ 33.



รูปที่ 33

แสดงรูปแบบของฮาร์ดแวร์ในการทดสอบ

ก่อนการเริ่มต้นทำงานจะต้องทำการโหลดโปรแกรมแพ็คเก็ตไดร์ฟเวอร์ ซึ่งในที่นี้ใช้เน็ตเวิร์คการ์ด 3COM 3C503 ดังตัวอย่าง

C:\> 3c503 0x60 3 0x300

C:\> termserv

- ค่า 0x60 หมายถึง หมายเลข software interrupt ที่ใช้ มีค่าตั้งแต่ 0x60 - 0x80
- ค่า 3 หมายถึง หมายเลข hardware interrupt ที่ใช้ มีค่าตั้งแต่ 2 - 5
- ค่า 0x300 หมายถึง หมายเลขช่องทางเข้าออกพื้นฐาน (I/O base address)

ค่าต่างๆ เหล่านี้ขึ้นอยู่กับชนิดของเน็ตเวิร์คการ์ด ซึ่งสามารถดูได้จากคู่มือของการ์ดแต่ละชนิด

การกำหนดชนิดของเทอร์มินัล (emulate type) สามารถกำหนดโดยหลังจากติดต่อ (connect) เข้าที่โฮสนั้นๆ แล้วจึงเซ็ตตัวแปร TERM ของโฮสนั้นๆ ให้เหมือนกับที่คอมมิวนิเคชั่นโปรแกรม

ลักษณะการทดสอบ

1. ทำการทดสอบโดยใช้คำสั่งในการ dump ไฟล์ขนาด 560,607 ไบต์ เปรียบเทียบเวลาในการทำงานจนสิ้นสุด แบ่งตามจำนวนพอร์ตอนุกรม 1 พอร์ต และ 2 พอร์ต โดยใช้คอมพิวเตอร์ส่วนบุคคล และ bit rate ของพอร์ตอนุกรมข้างต้น และจำนวนครั้งในการทดสอบ 12 ครั้ง แล้วเวลาเฉลี่ยในแต่ละเงื่อนไข ผลการทดสอบแสดงดังตารางที่ 7 และ 8.

ตารางที่ 7

แสดงผลการทดสอบโดยใช้ 386SX-25 เป็นเทอร์มินัลเซิฟเวอร์

รูปแบบ	bit rate	เวลาที่ใช้ (นาที)	ความเร็วในการทำงาน(บิต/วินาที)
ต่อเทอร์มินัล 1 พอร์ต	9,600	9:46	7,653
	19,200	5:04	14,752
	38,400	2:49	26,537
	57,600	1:59	37,688
	115,200	1:09	64,997
ต่อเทอร์มินัล 2 พอร์ต	9,600	19:39	3,803
	19,200	10:19	7,245
	38,400	5:44	13,037
	57,600	4:08	18,084
	115,200	2:35	28,934

ตารางที่ 8

แสดงผลการทดสอบโดยใช้ 386SX-16 เป็นเทอร์มินัลเซิฟเวอร์

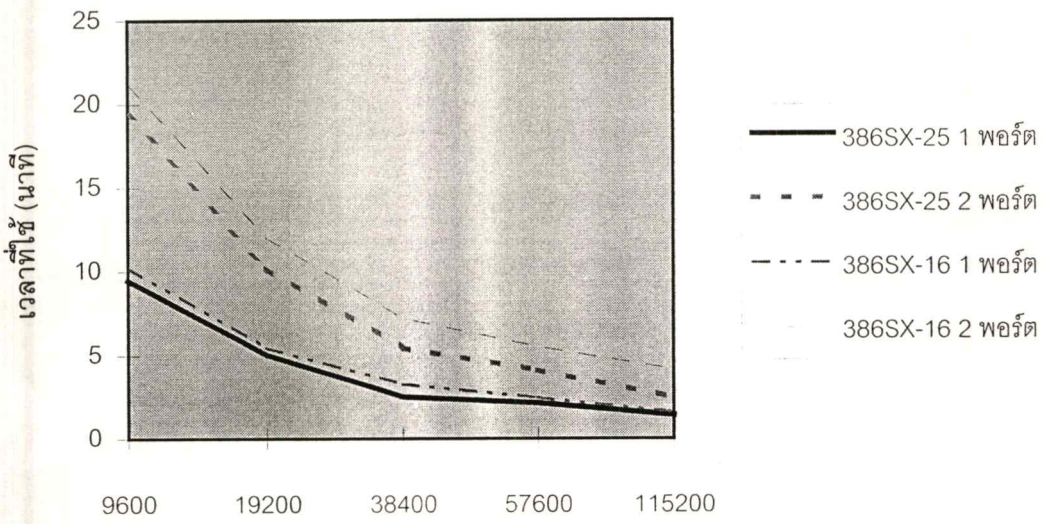
รูปแบบ	bit rate	เวลาที่ใช้ (นาที)	ความเร็วในการทำงาน(บิต/วินาที)
ต่อเทอร์มินัล 1 พอร์ต	9,600	10:24	7,187
	19,200	5:46	12,962
	38,400	3:25	21,877
	57,600	2:38	28,385
	115,200	1:41	44,404

ตารางที่ 8 (ต่อ)

แสดงผลการทดสอบโดยใช้ 386SX-16 เป็นเทอร์มินัลเซิฟเวอร์

รูปแบบ	bit rate	เวลาที่ใช้ (นาที)	ความเร็วในการทำงาน(บิต/วินาที)
ต่อเทอร์มินัล 2 พอร์ต	9,600	21:21	3,501
	19,200	12:00	6,228
	38,400	7:22	10,146
	57,600	5:49	12,850
	115,200	4:15	17,587

กราฟจากผลการทดสอบในตารางที่ 7 และ 8 แสดงในรูปที่ 34.



รูปที่ 34

กราฟแสดงผลการทดสอบ

2. ทดสอบการใช้งานโปรแกรมประยุกต์บนโฮส ได้แก่ vi กำหนดชนิดของเทอร์มินัลเป็น vt102, ansi, hp โดยเซ็ตค่า Environment ของ TERM บน Bourne Shell ดังนี้

TERM=vt102

export TERM

ทดสอบโดยทำงานบนโหมดทั้งสองของ vi ได้แก่ โหมดคำสั่ง (command mode) เช่น การเลื่อนเคอร์เซอร์ไปยังจุดต่างๆ และโหมดใส่ข้อความ (text entry mode) เช่น การคีย์ตัวอักษรใดๆ

3. ทดสอบความผิดพลาดในการรับส่งข้อมูล โดยทำการ dump ไฟล์ขนาด 51,031,617 ไบต์ด้วยคำสั่ง cat [filename] มาที่คอมพิวเตอร์ส่วนบุคคลที่ต่อเป็นเทอร์มินัล แล้วใช้ความสามารถของโปรแกรม emulation เก็บข้อมูลจากจอภาพลงฮาร์ดดิสก์ แล้วนำไฟล์ไปเปรียบเทียบกับไฟล์ที่โฮส โดยใช้คำสั่ง diff บนโฮส (เนื่องจากอักขระการขึ้นบรรทัดใหม่ในไฟล์ของยูนิกซ์ใช้ CR แต่ไฟล์ของ DOS ใช้ CR+LF ดังนั้นก่อนนำไฟล์ของเครื่องคอมพิวเตอร์ส่วนบุคคลที่ระบบปฏิบัติการ DOS มาเปรียบเทียบต้องใช้คำสั่ง dos2unix แปลงไฟล์ก่อน)

การทดสอบนี้ได้ทำการทดสอบกับเทอร์มินัล 2 ตัว ใช้ความเร็วของพอร์ตอนุกรม 115,200 บิต/วินาที จำนวน 4 ครั้ง ปรากฏว่าไม่มีความผิดพลาดของข้อมูลที่ส่ง ไฟล์ที่ได้รับมาจากเทอร์มินัลทั้งสองเหมือนกันกับไฟล์ที่โฮส

บทที่ 7
สรุปผลการวิจัยและข้อเสนอแนะ

<u>สรุปผลการวิจัย</u>

จากผลการทดสอบแยกตามลักษณะการทดสอบเป็นดังนี้ .

1. จากกราฟบทที่ 6 จะพบว่าปัจจัยที่มีผลต่อความเร็วในการทำงานของเทอร์มินัลเซิฟเวอร์แบ่งตามลักษณะของการประมวลผลดังนี้

- ช่วงแยกข้อมูลอีเทอร์เน็ตแพ็คเก็ตของแต่ละพอร์ตอนุกรม ขึ้นอยู่กับความเร็วของซีพียูที่ใช้ในการทำงานเป็นเทอร์มินัลเซิฟเวอร์
- ช่วงกระจายข้อมูลไปยังแต่ละพอร์ตอนุกรม ขึ้นอยู่กับความเร็วของพอร์ตอนุกรมและจำนวนพอร์ตอนุกรมที่ต่ออยู่ ทั้งนี้เนื่องจากการส่งข้อมูลทางพอร์ตอนุกรมจะต้องมีการตรวจเช็คบัฟเฟอร์ก่อนที่จะส่ง เพื่อป้องกันการเขียนทับของข้อมูล

2. การทำงานของเทอร์มินัลโดยใช้งานโปรแกรมประยุกต์บนโฮส เช่น vi สามารถทำงานได้อย่างถูกต้อง เนื่องจากเทอร์มินัลเซิฟเวอร์ไม่ได้ทำการเปลี่ยนแปลงค่าของคีย์ใดๆที่ส่งไปที่โฮส เพราะโฮสมีสามารถสนับสนุนการทำงานของเทอร์มินัลชนิดนั้นๆได้โดยตั้งค่าสภาวะแวดล้อม TERM ให้ตรงกับชนิดของเทอร์มินัล ทั้งนี้ขึ้นอยู่กับโฮสนั้นๆด้วย

3. โปรแกรมบริการเทอร์มินัลนี้ ไม่มีการตรวจสอบความผิดพลาดในการรับส่งทางเน็ตเวิร์ค ผลการทดสอบในส่วนการตรวจสอบความผิดพลาดแล้วไม่เกิดความผิดพลาดขึ้น เนื่องจากรูปแบบและฮาร์ดแวร์ที่ใช้ในการทดสอบมีขอบเขตจำกัด จำนวนโหนดในเน็ตเวิร์คมีเพียง 2 โหนด หากทดสอบในสภาวะแวดล้อมของเน็ตเวิร์คที่มีขนาดใหญ่และมีการใช้งานเน็ตเวิร์คคับคั่งก็อาจจะเกิดความผิดพลาดได้

ข้อเสนอแนะ

ข้อเสนอแนะของโปรแกรมนี้มีดังนี้

1. ควรพัฒนาให้สามารถใช้กับเน็ตเวิร์คหลายๆ เน็ตเวิร์ค และสามารถ connect เข้าไปที่โฮสที่ต่างเน็ตเวิร์คได้

2. ควรพัฒนาให้มีการตรวจสอบความผิดพลาดในขณะรับส่งผ่านเน็ตเวิร์ค โดยอาศัยฟังก์ชั่นในการตรวจสอบความผิดพลาดของโปรโตคอล TCP

3. ควรพัฒนาให้สามารถใช้กับโมเด็ม โดยสามารถใช้โมเด็มเข้ามาต่อที่พอร์ตอนุกรมทั้งสอง ทำให้สามารถ connect ผ่านทางโมเด็มได้

4. ควรขยายความสามารถในการให้บริการพอร์ตอนุกรมจาก 2 พอร์ต เป็น 8 พอร์ต หรือ 10 พอร์ต

บรรณานุกรม

[1] คมกริช ศิริแสงชัยกุล. "TCP/IP กับ ISO ใครจะแน่กว่ากัน" บิสิเนสคอมพิวเตอร์แมกกาซีน. ปีที่ 5, ฉบับที่ 49 (มีนาคม 2536) : 156.

[2] ไพศาล สงวนหมู่. "มาตรฐานการสื่อสารข้อมูลและการพัฒนายูทิลิตี้" ไมโครคอมพิวเตอร์. ฉบับที่ 35 (มกราคม 2531) : 147-152.

[3] ไพศาล สงวนหมู่ และยืน ภู่วรวรรณ. การสื่อสารข้อมูลและไมโครคอมพิวเตอร์เน็ตเวิร์ค. กรุงเทพฯ: ซีเอ็ดยูเคชั่น, 2536.

[4] Borland International. Turbo C Reference Guide Version 2.0. Scotts Valley : Borland International, 1988.

[5] Comer, Douglas E. Internetworking with TCP/IP Volume I; principle, protocols, and architecture. Englewood Cliffs : PRENTICE-HALL, 1991.

[6] FTP Software Inc. PC/TCP packet Driver Specification : Sun. Soe. clarkson. edu via anonymous ftp, 1989.

[7] Rieken, Bill and Lyle Weiman. Adventure in UNIX Network Applications Programming. Palo Alto : John Wiley and Sons, 1992.

[8] Stevens, Richard W. TCP/IP Illustrated.california:Addison-wesley,1996.

[9] Sun Microsystems Inc. Advanced System Administration. Mountain View : Sun Microsystem Inc, 1992.

ภาคผนวก

ชนิดและคลาสของเน็ตเวิร์คการ์ด

ชนิดและคลาสของการชื่อมต่อขึ้นอยู่กับโปรโตคอลและเน็ตเวิร์คการ์ดที่ใช้กับแพ็คเก็ตไดร์ฟเวอร์มีค่าดังนี้

DEC/Intel/Xerox "Bluebook" Ethernet

Class 1

ชนิดของเน็ตเวิร์คการ์ด	Type
3COM 3C500/3C501	1
3COM 3C505	2
Interlan Ni5010	3
BICC Data Networks 4110	4
BICC Data Networks 4117	5
MICOM-Interlan NP600	6
Ungermann-Bass PC-NIC	8
Univation NC-516	9
TRW PC-2000	10
Interlan Ni5210	11
3COM 3C503	12
3COM 3C523	13
Western Digital WD8003	14
Spider System S4	15
Torus Frame Level	16
10NET Communications	17
Gateway PC-bus	18
Gateway AT-bus	19

Gateway ACA-bus	20
IMC Pcnic	21
IMC Pcnic II	22
IMC Pcnic 8bit	23
Tigan Communications	24
Micromatic Research	25
Clarksom "Multiplexor"	26
D-Link 8–bit	27
D-Link 16–bit	28
D-Link PS/2	29
Research Machines 8	30
Research Machines 16	31
Research Machines MCA	32
Redix Microsys. EXM1 16-bit	33
Interlan Ni9210	34
Interlan Ni6510	35
Vestra LANMASTER 16-bit	36
Vestra LANMASTER 8-bit	37
Allied Telesis PC/XT/AT	38
Allied Telesis NEC PC-98	39
Allied Telesis Fujitsu FMR	40
Ungermann-Bass NIC/PS2	41
Tiara LANCard/E AT	42
Tiara LANCard/E MC	43
Tiara LANCard/E TP	44
Spider Comm. SpiderComm8	45
Spider Comm. SpiderComm16	46
AT&T Starlan NAU	47
AT&T Starlan-10 NAU	48
AT&T Ethernet NAU	49

Intel smart card	50

ProNET-10

Class 2

ชนิดของเน็ตเวิร์คการ์ด	Type
Proteon p1300	1
Proteon p1800	2

IEEE 802.5/ProNET-4

Class 3

ชนิดของเน็ตเวิร์คการ์ด	Type
IBM Token ring adapter	1
Proteon p1340	2
Proteon p1344	3
Gateway PC-bus	4
Gateway AT-bus	5
Gateway MCA-bus	6

KISS

Class 10

IEE 802.3 w/802.2 hdrs

Class 11

Type as given under DIX Ethernet

FDDI w/802.0 hdrs

Class 12

Internet X.25

Class 13

ชนิดของเน็ตเวิร์คการ์ด	Type
Western Digital	1
Frontier Technology	2

N.T. LANSTAR (encapsulating DIX)

Class 14

Omninet

Class 4

Appletalk

Class 5

Serial line

Class 6

ชนิดของเน็ตเวิร์คการ์ด	Type
Clarkson 8250-SLIP	1
Clarkson "Multiplexor"	2

Starlan

Class 7

ArcNet

Class 8

ชนิดของเน็ตเวิร์คการ์ด	Type
Datapoint RIM	1

AX.25

Class 9

ชนิดของเน็ตเวิร์คการ์ด	Type
NT LANSTAR/8	1
NT LANSTAR/MC	2

ประวัติผู้เขียน

นายอภิรมย์ แซ่เล้า เกิดวันที่ 2 เมษายน 2513 ที่จังหวัดกรุงเทพฯ สำเร็จการศึกษาวิทยาศาสตรบัณฑิต (วิทยาการคอมพิวเตอร์) จากสถาบันเทคโนโลยีพระจอมเกล้าพระนครเหนือ ปีการศึกษา 2533 และเข้าทำงานในตำแหน่งโปรแกรมเมอร์ ที่ฝ่ายระบบสารสนเทศ ธนาคารไทยพาณิชย์ 1 ปี และตำแหน่งวิศวกรระบบที่บริษัท โอเพ่น คอมพิวติ้ง เทคโนโลยีส์ จำกัด 4 ปี ปัจจุบันทำงานในตำแหน่ง Customer Support Engineer ที่บริษัท ฮิวเล็ตแพคการ์ด (ประเทศไทย) จำกัด