สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

ระบบควบคุมการผ่านเข้าออก

Access Control System

โดย

นายประจักษ์ จันทร์ใหม่มูล

วพ
21997
2548

เลขหมู่.....
เลขทะเบียน 62558
วัน,เดือน,ปี 19 ส.ค. 2549

b 11626161
i.....

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมโทรคมนาคม

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2548

ระบบควบคุมการผ่านเข้าออก

Access Control System

โดย

นายประจักษ์ จันทร์ใหม่มูล 45010437

อาจารย์ที่ปรึกษา

ผศ. เกรียงไกร วงศ์โรจนภรณ์

รศ.ดร. สุวิพล สิทธิชีวภาค

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมโทรคมนาคม

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2548

ผ่านการตรวจชิ้นงานแล้ว

(ลงชื่อ).....ผู้ตรวจ

ผ่านการตรวจรูปเล่มแล้ว

(ลงชื่อ).....ผู้ตรวจ

ปริญญานิพนธ์ปีการศึกษา 2548

ภาควิชาวิศวกรรมโทรคมนาคม

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง **ระบบควบคุมการผ่านเข้าออก**

Access Control System

ผู้จัดทำ

1. นายประจักษ์ จันทรีไหม่มูล 45010437

_____ อาจารย์ที่ปรึกษา

(ผศ. เกรียงไกร วงศ์โรจนภรณ์)

_____ อาจารย์ที่ปรึกษา

(รศ.ดร. สุวิพล สิทธิชีวภาค)

ระบบควบคุมการผ่านเข้าออก

Access Control System

โดย นายประจักษ์ จันทร์ใหม่มูล 45010437

อาจารย์ที่ปรึกษา ผศ.เกรียงไกร วงศ์โรจนภรณ์

รศ.ดร.สุวิพล สิทธิชีวภาค

บทคัดย่อ

ระบบควบคุมการผ่านเข้าออกนี้เป็นส่วนหนึ่งของระบบรักษาความปลอดภัยซึ่งทำงานโดยตัวอ่านบัตรทำการอ่านข้อมูลจากบัตรแล้วส่งสัญญาณไปประมวลผลที่ส่วนควบคุมการผ่านซึ่งทำหน้าที่ควบคุมการล็อคหรือปลดล็อคกลอนประตูไฟฟ้า โดยที่ส่วนควบคุมนี้ถูกกำหนดลักษณะการทำงานด้วยคอมพิวเตอร์และแจ้งสถานะการทำงานของระบบมายังคอมพิวเตอร์โดยผ่านเครือข่าย LAN

Abstract

Access Control System is a part of Security System. It works by the reader read the data in the card and transmits the signal to the access controller in order to lock or unlock the door. The access controller is set the property by the computer and It also reports the system status to the computer via Local Area Network.

ระบบควบคุมการผ่านเข้าออก

Access Control System

โดย นายประจักษ์ จันทร์ใหม่มูล 45010437

อาจารย์ที่ปรึกษา ผศ.เกรียงไกร วงศ์โรจนภรณ์

รศ.ดร.สุวิพล สิทธิชีวภาค

บทคัดย่อ

ระบบควบคุมการผ่านเข้าออกนี้เป็นส่วนหนึ่งของระบบรักษาความปลอดภัยซึ่งทำงานโดยตัวอ่านบัตรทำการอ่านข้อมูลจากบัตรแล้วส่งสัญญาณไปประมวลผลที่ส่วนควบคุมการผ่านซึ่งทำหน้าที่ควบคุมการล็อคหรือปลดล็อคกลอนประตูไฟฟ้า โดยที่ส่วนควบคุมนี้ถูกกำหนดลักษณะการทำงานด้วยคอมพิวเตอร์และแจ้งสถานะการทำงานของระบบมายังคอมพิวเตอร์โดยผ่านเครือข่าย LAN

Abstract

Access Control System is a part of Security System. It works by the reader read the data in the card and transmits the signal to the access controller in order to lock or unlock the door. The access controller is set the property by the computer and It also reports the system status to the computer via Local Area Network.

สารบัญ

หัวข้อ	หน้า
บทที่ 1 บทนำ	1
1.1 ความสำคัญและที่มา	1
1.2 วัตถุประสงค์ของโครงงาน	3
1.3 ขอบเขตของโครงงาน	3
บทที่ 2 ทฤษฎีและการออกแบบ	4
2.1 เครือข่ายท้องถิ่น (Local Area Network)	4
2.1.1 โครงสร้างของเครือข่าย (Network Topology)	4
2.1.2 เทคโนโลยีเครือข่าย	5
- อีเธอร์เน็ต (Ethernet)	5
- โทเคนริง (Token Ring)	6
- ATM (Asynchronous Transfer Mode)	6
2.2 อีเธอร์เน็ต (Ethernet)	6
2.2.1 สถาปัตยกรรม IEEE 802.3 อีเธอร์เน็ต	7
2.2.2 ฟอร์แมตเฟรมข้อมูล	9
2.3 โปรโตคอลทีซีพีไอพี	10
2.3.1 Internet Protocol (IP)	11
2.3.2 Transmission Control Protocol (TCP)	14
2.4 IP Addressing	18
2.4.1 ประเภทของหมายเลขไอพี	19
2.4.2 Private/Public Internet	20
2.4.3 ซับเน็ตและซับเน็ตมาส์ก (Subnet and Subnet Mask)	21
2.5 การสื่อสารข้อมูลแบบอนุกรม	22
2.5.1 ความเร็วของการสื่อสารข้อมูลแบบอนุกรม	22
2.5.2 รูปแบบของการส่งข้อมูลแบบอนุกรม	23
2.5.3 มาตรฐานพอร์ตอนุกรมแบบ RS-232	23
2.5.4 โค้ด (code)	27
2.6 ไมโครโปรเซสเซอร์ Rabbit 2000	28
2.6.1 โมดูลแร็บบิต RCM 2200	28
2.6.2 Prototyping Board	30
2.6.3 สายโปรแกรม (Programming Cable)	31
2.6.4 ส่วนของซอฟท์แวร์ที่ใช้ในการพัฒนาโมดูลแร็บบิต RCM 2200	32

หัวข้อ	หน้า
บทที่ 3 การคำนวณและการสร้าง	33
3.1 กลอนประตูไฟฟ้าดรอปโบลท์ (Drop Bolt)	33
3.2 เครื่องอ่านบัตรแม่เหล็ก (Magnetic Card Reader)	35
3.3 ส่วนควบคุมการผ่าน (Access Controller)	41
3.4 คอมพิวเตอร์ควบคุม	45
บทที่ 4 การทดลองและผลการทดลอง	46
4.1 ทดลองอ่านข้อมูลจากบัตรแม่เหล็กด้วยเครื่องอ่านบัตรแม่เหล็กที่สร้างขึ้นมา	46
ผลการทดลอง	50
4.2 ทดลองสื่อสารระหว่างส่วนควบคุมการผ่านกับคอมพิวเตอร์ควบคุม	51
ผลการทดลอง	54
4.3 ทดลองการทำงานของระบบควบคุมการผ่านเข้าออกในโหมดการทำงาน Manual	54
ผลการทดลอง	59
4.4 ทดลองการทำงานของระบบควบคุมการผ่านเข้าออกในโหมดการทำงาน Automatic	59
ผลการทดลอง	61
บทที่ 5 วิจารณ์และสรุป	62
5.1 ปัญหาที่พบ	63
5.2 แนวทางในการพัฒนาต่อไป	63
ภาคผนวก	64
บรรณานุกรม	

สารบัญรูปภาพ

หัวข้อ	หน้า
รูปที่ 1.1 : รูปแบบอย่างง่ายของ Access Control System	1
รูปที่ 1.2 : รูปแบบเบื้องต้นของระบบควบคุมการผ่านเข้าออกในโครงงานนี้	2
รูปที่ 2.1 : เครือข่ายท้องถิ่น	4
รูปที่ 2.2 : เปรียบเทียบระหว่าง IEEE Ethernet และแบบอ้างอิง OSI	8
รูปที่ 2.3 : ฟอร์แมตของอีเธอร์เน็ต	9
รูปที่ 2.4 : ฟอร์แมตของแพ็กเก็ตไอพี	12
รูปที่ 2.5 : ฟอร์แมตข้อมูลของแพ็กเก็ตทีซีพี	16
รูปที่ 2.6 : การแบ่งส่วนของหมายเลขไอพีและด็อตเดซิมอลโนเตชัน	19
รูปที่ 2.7 : การทำซับเน็ต	21
รูปที่ 2.8 : การต่ออุปกรณ์ภายนอกเข้ากับคอมพิวเตอร์แบบ Null Modem	25
รูปที่ 2.9 : การต่ออุปกรณ์ภายนอกเข้ากับคอมพิวเตอร์แบบ RS-232 โดยใช้สัญญาณเพียง3 เส้น	26
รูปที่ 2.10 : ตารางรหัสแอสกี้แทนตัวอักษร	27
รูปที่ 2.11 : ส่วนต่างๆภายในไมโครโปรเซสเซอร์แร็บบิต 2000	28
รูปที่ 2.12 : โมดูลแร็บบิต RCM 2200	29
รูปที่ 2.13 : การเชื่อมต่อโมดูลแร็บบิตเข้ากับเครือข่ายคอมพิวเตอร์	29
รูปที่ 2.14: RCM2200 I/O Pin outs	30
รูปที่ 2.15 : Prototyping Board	31
รูปที่ 3.1 : Block Diagram แสดงระบบรวม	33
รูปที่ 3.2 : ดรอปโบลท์ (Drop Bolt) เมื่อมองจากด้านหน้า	33
รูปที่ 3.3 : ดรอปโบลท์เมื่อมองจากด้านข้าง	34
รูปที่ 3.4 : ดรอปโบลท์สภาวะล็อคและสถานะของประตูคือปิดอยู่	34
รูปที่ 3.5 : ดรอปโบลท์ในสภาวะปลดล็อคและสถานะของประตูคือเปิดอยู่	35
รูปที่ 3.6 : วงจรขับดรอปโบลท์	35
รูปที่ 3.7 : วงจรไมโครคอนโทรลเลอร์ที่ใช้ควบคุมการทำงานของเครื่องอ่านบัตรแม่เหล็ก	39
รูปที่ 3.8 : การต่อใช้งานไอซี DS275 ในวงจรแปลงระดับลอจิก	40
รูปที่ 3.9 : การต่อใช้งานไอซี L293D ในวงจรขับสัญญาณควบคุม	40
รูปที่ 3.10 : การต่อโมดูลแร็บบิตเข้ากับส่วนอื่นๆ	41
รูปที่ 3.11 : แผนผังแสดงการเขียนโปรแกรมให้โมดูลแร็บบิต	42
รูปที่ 3.12 : แผนผังแสดงการเขียนโปรแกรมใน Manual Mode	43
รูปที่ 3.13 : แผนผังแสดงการเขียนโปรแกรมใน Automatic Mode	44
รูปที่ 4.1 : บัตรแม่เหล็กและเครื่องอ่านบัตรแม่เหล็กที่ใช้ในการทดลอง	47

รูปที่ 4.2 : การตั้งค่าการใช้งานโปรแกรมไฮเปอร์เทอมินอลในการทดลอง 48
รูปที่ 4.3 : หน้าต่างเริ่มต้นของโปรแกรมหลังจากตั้งค่าการใช้งานเสร็จ 48
รูปที่ 4.4 : แสดงข้อมูลที่อ่านได้ถูกต้อง 49
รูปที่ 4.5 : แสดงข้อมูลที่อ่านได้ถูกต้องจากบัตรแม่เหล็กอีกใบ 49
รูปที่ 4.6 : ข้อมูลที่อ่านได้เมื่อรูดผ่านหัวอ่านแบบสวนทางกับทิศทางในการรูดตามปกติ 50
รูปที่ 4.7 : แสดงความเร็วที่ได้จากการเชื่อมต่อ 51
รูปที่ 4.8 : แสดงการตั้งค่าไอพีแอดเดรสและเน็ตมาส์ค ของคอมพิวเตอร์ 52
รูปที่ 4.9 : ผลที่ได้จากคำสั่ง ping 52
รูปที่ 4.10 : ผลที่ได้จากคำสั่ง ping หลังจากเปลี่ยนไอพีแอดเดรส 53
รูปที่ 4.11 : ผลที่ได้จากการเชื่อมต่อโดยเทลเน็ต 53
รูปที่ 4.12 : แสดงการเชื่อมต่อระบบรวม 55
รูปที่ 4.13 : แสดงผลที่ได้จากการเชื่อมต่อโดยเทลเน็ตและเข้าสู่โหมด Manual 55
รูปที่ 4.14 : แสดงผลที่ได้หลังจากรูดบัตรแม่เหล็ก 56
รูปที่ 4.15.1 : แสดงผลที่ได้จากการตอบ “y” 56
รูปที่ 4.15.2 : แสดงผลที่ได้จากการตอบ “y” 57
รูปที่ 4.16.1 : แสดงผลที่ได้จากการตอบ “n” 57
รูปที่ 4.16.2 : แสดงผลที่ได้จากการตอบ “n” 58
รูปที่ 4.17 : แสดงผลที่ได้จากการตอบ “x” 58
รูปที่ 4.18 : แสดงผลที่ได้จากการเชื่อมต่อโดยเทลเน็ตและเข้าสู่โหมด Automatic 60
รูปที่ 4.19 : แสดงผลที่ได้หลังจากรูดบัตรแม่เหล็ก 60
รูปที่ 4.20 : แสดงผลที่ได้หลังจากรูดบัตรแม่เหล็ก 60
รูปที่ 5.1 : Block Diagram แสดงระบบรวม 62

สารบัญตาราง

หัวข้อ	หน้า
ตารางที่ 2.1 : หมายเลขพอร์ตของบางแอพพลิเคชัน	17
ตารางที่ 2.2 : หมายเลขไอพีเครือข่ายส่วนบุคคล	20
ตารางที่ 2.3 : ดีฟอลต์เน็ตมาส์กของไอพีแต่ละประเภท	22
ตารางที่ 2.4 : การจัดขาของคอนเน็กเตอร์พอร์ตอนุกรมตามมาตรฐาน RS– 232	24
ตารางที่ 3.1 แสดงข้อมูลของรหัส BCD สำหรับเครื่องอ่านบัตรแม่เหล็ก	38
ตารางที่ 3.2 : การกำหนดค่าให้กับคอมพิวเตอร์ควบคุม	45

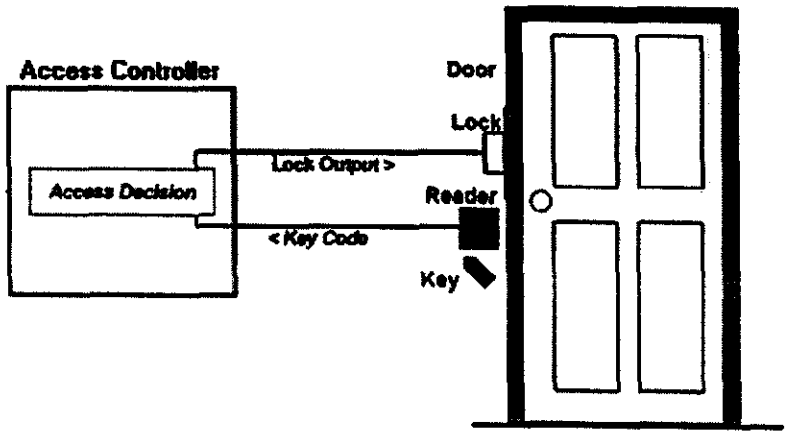
บทที่ 1
บทนำ

1.1 ความสำคัญและที่มา

ระบบควบคุมการผ่านเข้าออก หรือที่รู้จักกันในชื่อทางการค้าว่า "Access Control System" ซึ่งชื่ออาจจะฟังดูคล้ายกับ Access Control ในเรื่องของเครือข่ายท้องถิ่น (Local Area Network : LAN) แต่สำหรับในโครงงานนี้ "Access Control System" นั้นหมายถึงระบบควบคุมการผ่านเข้าออกที่มีการใช้งานกันอย่างแพร่หลายโดยเฉพาะในหน่วยงานที่ต้องการความปลอดภัยอย่างสูง ระบบจะอนุญาตให้เฉพาะผู้ที่มีสิทธิตามเงื่อนไขที่ตั้งไว้ สามารถผ่านเข้าไปได้

นิยามอย่างเป็นทางการของระบบนี้คือ การตรวจสอบการอนุญาตของการผ่านเข้าออก ณ จุดใดจุดหนึ่งโดยใช้อุปกรณ์ควบคุม หรือพูดกันง่ายๆก็คือ ระบบที่ใช้ตรวจสอบว่าใครผ่านเข้าออกที่ไหนในเวลาใด

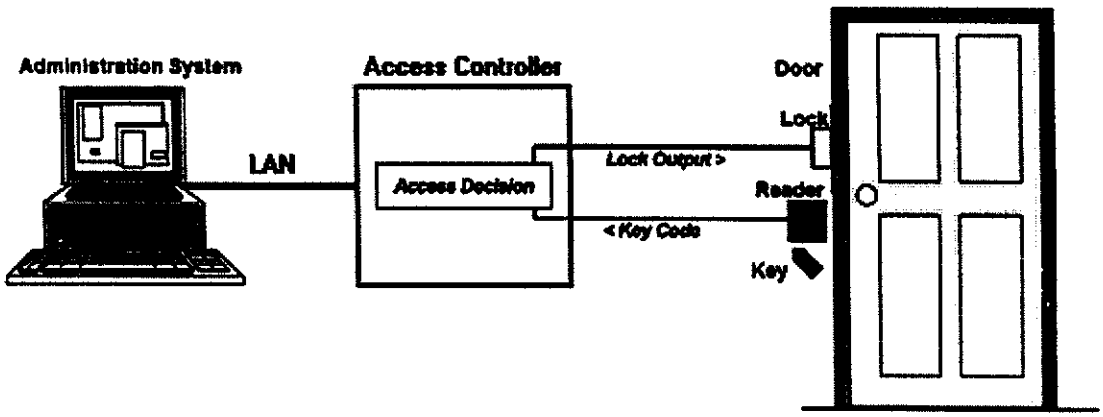
ระบบควบคุมการผ่านเข้าออก จัดว่าเป็นส่วนหนึ่งของระบบรักษาความปลอดภัยที่นับวันก็ยิ่งมีความจำเป็นต้องใช้กันมากขึ้น รูปแบบอย่างง่ายของระบบควบคุมการผ่านเข้าออกที่พบเห็นกันทั่วไปก็คือประตูเข้าออกหอพักหรืออาคารต่างๆที่ต้องใช้บัตรแม่เหล็กรูดผ่านเครื่องอ่านบัตรที่อยู่หน้าประตู ถ้าหากบัตรนั้นได้รับการอนุญาตให้ผ่านได้ผู้ถือบัตรจึงจะสามารถผ่านเข้าออกประตูนั้นได้ รูปแบบอย่างง่ายของระบบควบคุมการผ่านเข้าออก แสดงได้ดังรูปที่ 1.1



รูปที่ 1.1 : รูปแบบอย่างง่ายของระบบควบคุมการผ่านเข้าออก

ระบบตามรูปที่ 1.1 นั้นทำได้เพียงแค่ตัดสินใจที่จะอนุญาตให้ผู้ถือบัตรนั้นสามารถผ่านประตูไปได้หรือไม่โดยการตรวจสอบรหัสที่อ่านได้จากบัตร ถ้าหากบัตรนั้นได้รับการอนุญาตให้ผ่านได้ ส่วนควบคุมก็จะส่งสัญญาณไปปลดล็อคกลอนประตูไฟฟ้า ระบบตามรูปที่ 1.1 นี้ไม่สามารถบันทึกหรือตรวจสอบย้อนหลังได้ว่าในแต่ละช่วงเวลานั้นมีใครบ้างที่ผ่านเข้าออกประตูนี้

ระบบควบคุมการผ่านเข้าออก ที่มีจำหน่ายทั่วไปในท้องตลาดนั้น แม้จะมีประสิทธิภาพสูงและมีฟังก์ชันการใช้งานที่หลากหลาย ฟังก์ชันการใช้งานที่เพิ่มเข้ามามากมายนี้เป็นปัจจัยสำคัญที่ทำให้ราคาของระบบนั้นสูงตามไปด้วย และที่สำคัญฟังก์ชันการใช้ส่วนใหญ่ที่เพิ่มเข้ามาก็มักจะเกินความจำเป็นสำหรับผู้ใช้งานที่ต้องการระบบขนาดเล็ก ในโครงงานจึงได้จัดทำระบบควบคุมการผ่านเข้าออกที่มีประสิทธิภาพดีกว่าระบบในรูปที่ 1.1 และตัดฟังก์ชันการใช้งานที่เกินความจำเป็นสำหรับระบบขนาดเล็กออกไป ก็จะช่วยลดราคาของระบบลงไปได้มาก รูปแบบเบื้องต้นของระบบควบคุมการผ่านเข้าออกสำหรับโครงงานนี้แสดงได้ดังรูปที่ 1.2



รูปที่ 1.2 : รูปแบบเบื้องต้นของระบบควบคุมการผ่านเข้าออกในโครงงานนี้

ระบบตามรูปที่ 1.2 นั้นเกิดจากการเพิ่มเติมคอมพิวเตอร์ที่ใช้สำหรับควบคุมการทำงานโดยรวมของระบบเข้าไปในระบบดังรูปที่ 1.1 ซึ่งมีส่วนประกอบพื้นฐานของระบบดังนี้

1. Door Lock : เป็นกลอนประตูไฟฟ้าที่ใช้ล็อคประตู กลอนประตูไฟฟ้าที่นิยมใช้งานกันทั่วไปนั้นมีอยู่ 2 แบบคือ แบบใช้แรงแม่เหล็กไฟฟ้าเรียกว่า "Electro Magnetic Lock (EM Lock)" และแบบที่ใช้แกนล็อคเรียกว่า "Drop Bolt"
2. Reader : คือส่วนที่ใช้อ่านรหัสจากบัตรประจำตัวผู้ใช้เพื่อส่งไปวิเคราะห์ยังส่วนควบคุม สำหรับในโครงงานนี้จะใช้เป็นเครื่องอ่านบัตรแม่เหล็ก (Magnetic Card Reader)
3. Key : คือบัตรหรือสิ่งที่ใช้กำหนดสิทธิของผู้ใช้ ในโครงงานนี้จะใช้เป็นบัตรแม่เหล็ก
4. Access Controller : คือส่วนควบคุมการผ่านซึ่งเป็นส่วนที่รับสัญญาณจาก Reader เพื่อมาวิเคราะห์ว่าจะส่งสัญญาณไปปลดล็อคกลอนประตูไฟฟ้าหรือไม่ ตามเงื่อนไขที่ได้กำหนดไว้
5. Administration System : เป็นคอมพิวเตอร์ที่ทำหน้าที่ควบคุมและบันทึกการทำงานของระบบเพื่อเรียกดูย้อนหลังเมื่อมีปัญหาเกิดขึ้นภายในอาคาร และส่วนนี้ยังทำหน้าที่ควบคุมและกำหนดฟังก์ชันการทำงานของส่วนควบคุมการผ่านอีกด้วย

6. Local Area Network (LAN) : การเชื่อมต่อระหว่างส่วนควบคุมการผ่าน (Access Controller) และคอมพิวเตอร์ควบคุมเป็นแบบเครือข่ายท้องถิ่น ซึ่งใช้โปรโตคอลทีซีพีไอพี (TCP/IP : Transmission Control Protocol/Internet Protocol) ในการสื่อสารข้อมูลระหว่างกัน

1.2 วัตถุประสงค์ของโครงงาน

1. ศึกษาระบบการทำงานของเครือข่ายท้องถิ่น ที่ใช้โปรโตคอลทีซีพีไอพีในการสื่อสารข้อมูล
2. ศึกษาการสื่อสารข้อมูลอนุกรมแบบอะซิงโครนัส (Asynchronous) ของไมโครคอนโทรลเลอร์
3. ศึกษาการใช้งานโมดูลแร็บบิต (Rabbit Module) รุ่น RCM 2200 ซึ่งเป็นโมดูลที่สามารถเชื่อมต่อกับเครือข่ายท้องถิ่นได้โดยใช้ชุดคำสั่งภาษาไดนามิคซี (Dynamic C) ในการโปรแกรมการทำงานของโมดูล
4. นำความรู้ที่ได้ศึกษามาสร้างระบบควบคุมการผ่านเข้าออกจากการประยุกต์ใช้เครือข่ายท้องถิ่นที่มีอยู่

1.3 ขอบเขตของโครงงาน

โครงงานนี้ต้องสร้างเครื่องอ่านบัตรแม่เหล็กเพื่อใช้เป็นส่วนอ่านบัตรประจำตัวของผู้ใช้งานระบบควบคุมการผ่านเข้าออก เครื่องอ่านบัตรแม่เหล็กนี้จะใช้หัวอ่านบัตรแม่เหล็กทำงานร่วมกับไมโครคอนโทรลเลอร์ โดยเครื่องอ่านบัตรแม่เหล็กจะส่งหมายเลขบัตรไปให้โมดูลแร็บบิตโดยใช้การสื่อสารข้อมูลอนุกรมแบบอะซิงโครนัส โมดูลแร็บบิตจะทำหน้าที่ควบคุมการล็อคและปลดล็อคกลอนประตูไฟฟ้า โดยตรวจสอบหมายเลขบัตรที่ส่งเข้ามากับรายการหมายเลขบัตรที่ได้รับอนุญาตให้ผ่านได้ ถ้าหมายเลขบัตรที่ส่งเข้ามาอยู่ในรายการ โมดูลแร็บบิตก็จะส่งสัญญาณไปปลดล็อคกลอนประตูไฟฟ้า

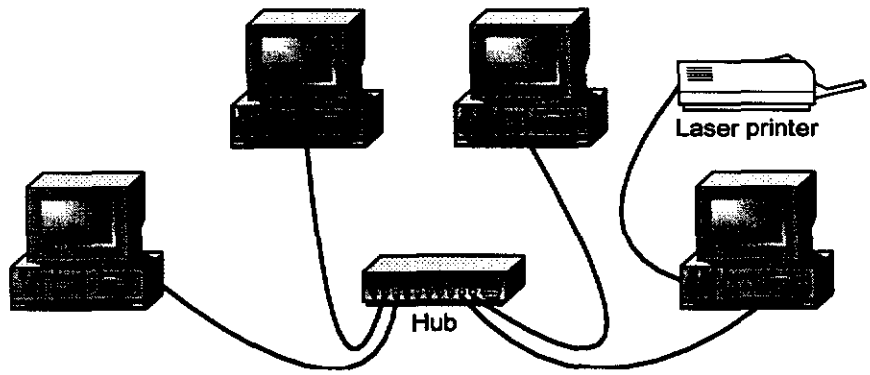
โมดูลแร็บบิตยังทำหน้าที่ส่งหมายเลขบัตรที่ได้รับมาไปให้คอมพิวเตอร์ผ่านทางเครือข่ายท้องถิ่นและใช้โปรโตคอลทีซีพีไอพีในการสื่อสาร เพื่อให้ผู้ดูแลระบบรับทราบและบันทึกข้อมูลไว้ อีกทั้งยังให้ผู้ดูแลระบบตอบกลับมาว่าจะให้หมายเลขบัตรที่ไม่ได้อยู่ในรายการสามารถผ่านได้หรือไม่ ถ้าผู้ดูแลระบบตอบกลับมาว่าให้ผ่านได้ โมดูลแร็บบิตก็จะปลดล็อคกลอนประตูไฟฟ้า แต่ถ้าผู้ดูแลระบบตอบกลับมาว่าไม่ให้ผ่าน โมดูลแร็บบิตก็จะไม่ปลดล็อคกลอนประตูไฟฟ้าและส่งสัญญาณเตือน ในการกำหนดการทำงานของโมดูลแร็บบิตนั้นจะใช้ชุดคำสั่งภาษาไดนามิคซีในการเขียนโปรแกรมการทำงานให้โมดูลแร็บบิตแล้วบันทึกไว้ในหน่วยความจำของโมดูลแร็บบิต

ในส่วนของโปรแกรมคอมพิวเตอร์ที่ใช้สื่อสารกับโมดูลแร็บบิตนั้นจะใช้โปรแกรมเทลเน็ต (Telnet) ซึ่งมีมาให้ในระบบปฏิบัติการไมโครซอฟท์วินโดวส์ (Microsoft Windows) เทลเน็ตเป็นโปรแกรมที่ใช้สำหรับเชื่อมต่อเพื่อใช้งานโฮสต์ (Host) จากระยะไกล เป็นแอพพลิเคชัน (Application) ที่ใช้โปรโตคอลทีซีพีไอพีในการสื่อสาร และทำงานที่พอร์ต (Port) 23

บทที่ 2
ทฤษฎีและการออกแบบ

2.1 เครือข่ายท้องถิ่น (Local Area Network : LAN)

เครือข่ายท้องถิ่นเป็นรากฐานของเครือข่ายคอมพิวเตอร์ทั่วไป กล่าวคือ เกือบทุกๆเครือข่ายต้องมีเครือข่ายท้องถิ่นเป็นองค์ประกอบ เครือข่ายท้องถิ่น อาจเป็นได้ตั้งแต่เครือข่ายแบบง่ายๆ เช่น มีคอมพิวเตอร์สองเครื่องเชื่อมต่อกันด้วยสายสัญญาณ ไปจนถึงเครือข่ายที่ซับซ้อน เช่น มีคอมพิวเตอร์เป็นร้อยๆเครื่องและมีอุปกรณ์เครือข่ายอื่นๆอีกมาก แต่ลักษณะสำคัญของเครือข่ายท้องถิ่นก็คือ เครือข่ายจะครอบคลุมพื้นที่จำกัด รูปที่ 2.1 แสดงเครือข่ายท้องถิ่นที่ประกอบด้วยเครื่องคอมพิวเตอร์สี่เครื่อง และมีเครื่องพิมพ์ที่แชร์กันใช้ เครื่องเซิร์ฟเวอร์ที่ใช้ในการจัดการเครือข่าย ซึ่งเครือข่ายจะรวมกันอยู่ในห้องปฏิบัติการ



รูปที่ 2.1 : เครือข่ายท้องถิ่น

2.1.1 โครงสร้างของเครือข่าย (Network Topology)

โครงสร้างของเครือข่ายจะอธิบายถึงการเชื่อมต่อคอมพิวเตอร์ตามลักษณะทางกายภาพหรือทางตรรกะ ซึ่งจะแสดงถึงตำแหน่งของคอมพิวเตอร์และอุปกรณ์เครือข่ายอื่นๆ และเส้นทางการเชื่อมต่อของอุปกรณ์เหล่านี้ โครงสร้างของเครือข่ายจะมีผลต่อสมรรถนะของเครือข่าย การเลือกใช้โครงสร้างของเครือข่ายต้องมีการวางแผนที่ดี เพราะโครงสร้างจะมีผลต่อชนิดของสายนำสัญญาณที่ใช้และเป็นตัวกำหนดลักษณะการสื่อสารกันระหว่างคอมพิวเตอร์ด้วย

- โครงสร้างแบบดาว (Star Topology) เป็นโครงสร้างที่เชื่อมต่อคอมพิวเตอร์แต่ละตัวเข้ากับคอมพิวเตอร์ศูนย์กลาง การรับส่งข้อมูลทั้งหมดจะต้องผ่านคอมพิวเตอร์ศูนย์กลางเสมอ มีข้อดีคือการเชื่อมต่อคอมพิวเตอร์เครื่องใหม่สามารถทำได้ง่ายและไม่กระทบกับเครื่องอื่นในระบบเลย แต่ข้อเสียคือมีค่าใช้จ่ายเกี่ยวกับสายสูงและถ้าคอมพิวเตอร์ศูนย์กลางเสียระบบเครือข่ายจะหยุดชะงักทั้งหมดทันที

- **โครงสร้างแบบบัส (Bus Topology)** เป็นโครงสร้างที่เชื่อมคอมพิวเตอร์แต่ละตัวด้วยสายเคเบิลที่ใช้ร่วมกัน ซึ่งสายเคเบิลหรือบัสนี้เปรียบเสมือนกับถนนที่ข้อมูลจะถูกส่งผ่านไปมาระหว่างแต่ละเครื่องได้ตลอดเวลา โดยไม่ต้องผ่านไปที่ศูนย์กลางก่อน โครงสร้างแบบนี้มีข้อดีที่ใช้สายน้อย และถ้ามีเครื่องเสียก็ไม่มีผลอะไรต่อระบบโดยรวม ส่วนข้อเสียก็คือตรวจหาจุดที่เป็นปัญหาได้ยาก
- **โครงสร้างแบบวงแหวน (Ring Topology)** เป็นโครงสร้างที่เชื่อมคอมพิวเตอร์ทั้งหมดเข้าเป็นวงแหวน ข้อมูลจะถูกส่งต่อๆกันไปในวงแหวนจนกว่าจะถึงเครื่องผู้รับที่ถูกต้อง ข้อดีของโครงสร้างแบบนี้คือ ใช้สายเคเบิลน้อยและสามารถตัดเครื่องที่เสียออกจากระบบได้ ทำให้ไม่มีผลต่อระบบเครือข่าย ข้อเสียคือหากมีเครื่องที่มีปัญหาอยู่ในระบบจะทำให้เครือข่ายไม่สามารถทำงานได้เลย และการเชื่อมต่อเครื่องเข้าสู่เครือข่ายอาจต้องหยุดระบบทั้งหมดลงก่อน

2.1.2 เทคโนโลยีเครือข่าย

เทคโนโลยีเครือข่ายท้องถิ่นมีหลายประเภท เช่น Ethernet, ATM, Token Ring, FDDI เป็นต้น แต่ที่นิยมกันมากที่สุดในปัจจุบันคือ อีเธอร์เน็ต (Ethernet) ซึ่งในอีเธอร์เน็ตเองยังจำแนกออกได้หลายประเภทขึ้นอยู่กับความเร็ว โครงสร้างของเครือข่ายและสายสัญญาณที่ใช้ เทคโนโลยีเครือข่ายท้องถิ่นแต่ละประเภทมีทั้งข้อดีข้อเสียที่แตกต่างกัน การเลือกใช้เทคโนโลยีเหล่านี้ควรเลือกให้เหมาะกับลักษณะการใช้งานเครือข่าย

- อีเธอร์เน็ต (Ethernet)

อีเธอร์เน็ตได้ถูกคิดค้นขึ้นตั้งแต่ทศวรรษ 1970 และยังคงเป็นเทคโนโลยีชั้นนำของเครือข่ายท้องถิ่น อีเธอเน็ตตั้งอยู่บนมาตรฐานการส่งข้อมูลหรือโปรโตคอลซีเอสเอ็มเอซีดี (CSMA/CD : Carrier Sense Multiple Access with Collision Detection) โปรโตคอลนี้ถูกใช้สำหรับการเข้าใช้สื่อกลางในการส่งสัญญาณที่แชร์กันระหว่างสถานีหรือโนด (Node) ต่างๆ ซึ่งมีขั้นตอนดังนี้ เมื่อโนดใดๆต้องการที่จะส่งข้อมูลจะต้องคอยฟังก่อน (Carrier Sense) ว่ามีโนดอื่นกำลังส่งข้อมูลอยู่หรือไม่ ถ้ามีให้รอจนกว่าโนดนั้นจะส่งข้อมูลเสร็จก่อน แล้วค่อยเริ่มส่งข้อมูล และในขณะที่กำลังส่งข้อมูลอยู่นั้นต้องคอยตรวจสอบว่ามีการชนกันของข้อมูลเกิดขึ้นหรือไม่ (Collision Detection) ถ้ามีการชนกันของข้อมูลเกิดขึ้นให้หยุดทำการส่งข้อมูลทันที และส่งสัญญาณแจ้งเตือนไปยังโนดอื่นๆว่ามีการชนกันของข้อมูลเกิดขึ้น แล้วค่อยเริ่มกระบวนการส่งข้อมูลใหม่อีกครั้ง เนื่องจากอีเธอร์เน็ตจะใช้สื่อกลางร่วมกัน ซึ่งเรียกว่า "บัส (BUS)" ฉะนั้นจึงมีโนดที่ส่งข้อมูลได้แค่โนดเดียวในขณะใดขณะหนึ่ง การชนกันของข้อมูลเกิดขึ้นเนื่องจากมีมากกว่าหนึ่งโนดที่ทำการส่งข้อมูลไปบนสื่อกลางในเวลาเดียวกัน ผลที่ได้คือ ข้อมูลจะกลายเป็นขยะหรืออ่านไม่ได้ทันที เมื่อมีจำนวนโนดเพิ่มมากขึ้นความน่าจะเป็นที่ข้อมูลจะชนกันก็เพิ่มขึ้นตามลำดับ

ตามมาตรฐานแล้วอีเธอร์เน็ตจะมีอัตราการส่งข้อมูลหรือแบนด์วิธที่ 10 Mbps (สิบล้านบิตต่อวินาที) ในขณะที่ฟาสต์อีเธอร์เน็ต (Fast Ethernet) มีการทำงานคล้ายๆกัน เพียงแต่มีอัตราข้อมูลที่สูงกว่า 10 เท่าหรือ 100 Mbps ส่วนกิกะบิตอีเธอร์เน็ต (Gigabit Ethernet) มีอัตราการส่งข้อมูลสูงสุดคือ

1,000 Mbps หรือ 1 Gbps และยังได้มีการพัฒนาอีเธอร์เน็ตที่ความเร็ว 10 Gbps ซึ่งเรียกว่า เท็นกิกะบิตอีเธอร์เน็ต (10G Ethernet)

นอกจากข้อแตกต่างในเรื่องของความเร็วแล้ว อีเธอร์เน็ตยังแบ่งย่อยออกเป็นแชร์อีเธอร์เน็ต (Shared Ethernet) และสวิตช์อีเธอร์เน็ต (Switched Ethernet) โดยแชร์อีเธอร์เน็ตมีการใช้ตัวกลางร่วมกัน คล้ายๆกับถนนมีเลนเดียว ดังนั้นจึงมีรถวิ่งบนถนนได้แค่คันเดียวในขณะใดขณะหนึ่ง ในความหมายเครือข่ายก็คือ ในขณะใดขณะหนึ่งจะมีแค่สถานีเดียวที่สามารถส่งข้อมูลได้ อุปกรณ์เครือข่ายที่ใช้สำหรับแชร์อีเธอร์เน็ตคือ ฮับ (Hub) ส่วนสวิตช์อีเธอร์เน็ตจะเปรียบได้กับถนนที่มีหลายเลน ดังนั้นจึงมีรถหลายคันที่สามารถวิ่งบนถนนได้ในเวลาเดียวกัน ส่วนอุปกรณ์ที่ใช้ในสวิตช์อีเธอร์เน็ตก็คือสวิตช์นั่นเอง

- โทเคนริง (Token Ring)

เครือข่ายแบบโทเคนริงซึ่งจะมีลักษณะการเชื่อมต่อแบบวงแหวนนี้ ถือได้ว่าเป็นเครือข่ายที่กำลังล้าสมัยเพราะมีการใช้งานน้อยลง โทเคนริงเป็นเครือข่ายที่นิยมใช้กันมากในการสร้างเครือข่ายช่วงสมัยแรกๆ เนื่องจากข้อดีของการส่งข้อมูลในเครือข่ายแบบนี้จะไม่มีการชนกันของข้อมูล เหมือนกับเครือข่ายแบบอีเธอร์เน็ต แต่ข้อเสียของเครือข่ายประเภทนี้อยู่ที่ความสามารถในการขยายเครือข่าย (Scalability) และการบริหารและจัดการเครือข่ายจะค่อนข้างยาก เครือข่ายประเภทนี้ยังมีใช้อยู่กับอุปกรณ์คอมพิวเตอร์ของบริษัท IBM ที่เป็นระบบเมนเฟรมและมินิคอมพิวเตอร์

- ATM (Asynchronous Transfer Mode)

ATM ย่อมาจาก "Asynchronous Transfer Mode" ไม่ได้หมายถึงตู้ ATM (Automatic Teller Machine) ที่เราใช้ถอนเงินสดจากธนาคาร แต่บางทีตู้ ATM ที่ใช้ถอนเงินสดอาจจะเชื่อมต่อกับศูนย์กลางด้วยระบบเครือข่ายแบบ ATM ก็ได้ ATM เป็นมาตรฐานการรับส่งข้อมูลที่กำหนดโดย ITU-T (International Telecommunication Union-Telecommunication Standard Sector) ซึ่งจะรวมบริการต่างๆ เช่น ข้อมูล เสียง วีดีโอ เข้าด้วยกันแล้วส่งเป็นเซลล์ (Cell) ข้อมูลที่มีขนาดเล็กและคงที่ เป็นเครือข่ายที่รองรับแบนด์วิธ (band width) ตั้งแต่ Mbps จนถึง Gbps ปัจจุบันยังมีการใช้ ATM ไม่มากเท่ากับอีเธอร์เน็ต แต่มีแนวโน้มว่า ATM อาจจะเป็นอีกทางเลือกหนึ่งที่นิยมในเครือข่ายในอนาคต

2.2 อีเธอร์เน็ต (Ethernet)

ปัจจุบันเป็นที่ยอมรับกันว่าอีเธอร์เน็ต เป็นเทคโนโลยีเครือข่ายที่เป็นฐานหลักของเทคโนโลยีสารสนเทศทั้งหมด เนื่องจากเป็นเทคโนโลยีเครือข่ายท้องถิ่นที่เป็นที่นิยมมากที่สุด อีเธอร์เน็ตมีอายุกว่า 30 ปีแล้ว และได้มีการพัฒนาอย่างต่อเนื่อง ด้วยเหตุนี้จึงเป็นการยากที่จะพัฒนาเทคโนโลยีใหม่มาแทนที่ได้ เทคโนโลยีนี้ได้ถูกพัฒนาและปรับปรุงภายใต้ความดูแลและรับผิดชอบของสถาบัน IEEE (Institute of Electrical and Electronics Engineer) โดยสิ่งที่สำคัญอย่างหนึ่งในการเปลี่ยนแปลงและปรุงคือการเพิ่มความเร็วในการรับส่งข้อมูลหรือแบนด์วิธ

ในการปรับปรุงครั้งแรกนั้นเป็นการปรับจากความเร็วเดิมที่ 10 Mbps เป็น 100 Mbps ซึ่งในการปรับปรุงครั้งนั้นได้มีการพัฒนาฟิสิคอลเลเยอร์ (Physical Layer) เพื่อให้สามารถรับส่งข้อมูลได้ที่ความเร็ว 100 Mbps และในการปรับปรุงฟิสิคอลเลเยอร์นี้ทำให้ต้องมีการปรับเปลี่ยนดาต้าลิงค์เลเยอร์ (Data Link Layer) เช่นกัน มาตรฐานใหม่นี้เรียกว่า "อีเธอร์เน็ตความเร็วสูง หรือฟาสต์อีเธอร์เน็ต" และได้รับความนิยมเหนือ ATM

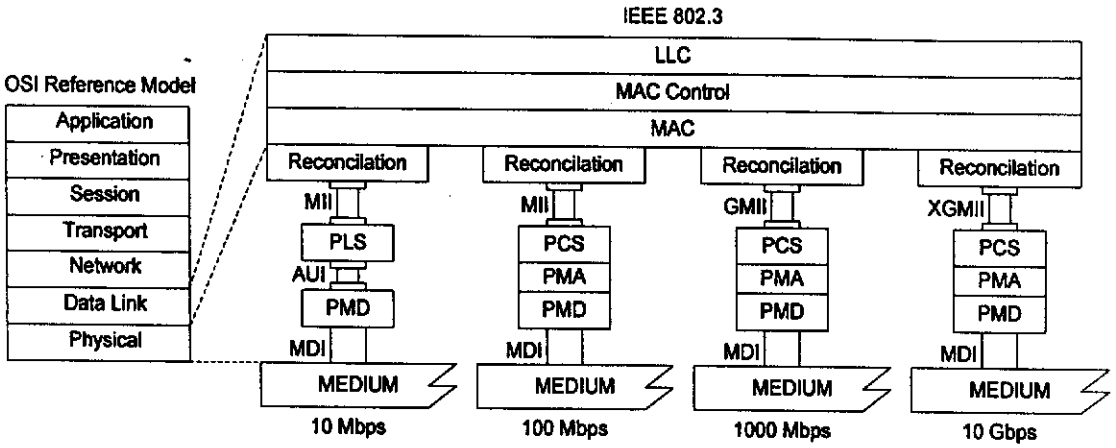
2.2.1 สถาปัตยกรรม IEEE 802.3 อีเธอร์เน็ต

ในสมัยแรกคำว่า "อีเธอร์เน็ต กับ ซีเอสเอ็มเอซีดี" มักจะหมายถึงระบบเครือข่ายชนิดเดียวกัน เนื่องจากอีเธอร์เน็ตจะใช้โปรโตคอลซีเอสเอ็มเอซีดีในการเข้าถึงสื่อกลางในการรับส่งข้อมูล แต่ปัจจุบันความหมายของอีเธอร์เน็ตได้เปลี่ยนไปเพราะได้มีการปรับปรุงเทคโนโลยี เช่น อีเธอร์เน็ตความเร็วสูงได้พัฒนาโปรโตคอลในฟิสิคอลเลเยอร์ใหม่ และมีการปรับเปลี่ยนกลไกในการเข้าใช้สื่อกลางเล็กน้อย สิ่งที่สำคัญที่สุดคือ การเพิ่มการรับส่งข้อมูลแบบฟูลล์ดูเพล็ก (Full Duplex) หรือการสื่อสารข้อมูลที่สามารถรับส่งข้อมูลได้ในเวลาเดียวกัน ซึ่งการรับส่งข้อมูลแบบนี้จะใช้สายคู่เกลียวบิดหนึ่งคู่ในการส่งข้อมูลและอีกหนึ่งคู่ในการรับข้อมูล เมื่อใช้เทคโนโลยีสวิตชิ่ง (Switching) ทำให้กำจัดปัญหาในการเข้าใช้สื่อกลางได้ เนื่องจากไม่จำเป็นต้องแชร์สื่อนำสัญญาณร่วมกัน อุปกรณ์เครือข่ายสามารถรับส่งข้อมูลได้ในอัตราที่สูงขึ้น ซึ่งประสิทธิภาพของเครือข่ายจะถูกจำกัดโดยสายสัญญาณที่ใช้มากกว่า ดังนั้นคำว่าซีเอสเอ็มเอซีดีก็จะใช้แทนคำว่าอีเธอร์เน็ตไม่ได้อีกต่อไป

อีกคำหนึ่งคือ บรอดแบนด์ (Broadband) กับเบสแบนด์ (Baseband) อีเธอร์เน็ตเกือบทุกประเภทจะเป็นแบบเบสแบนด์ มีบางประเภทที่เป็นบรอดแบนด์ เช่น 10Broad36 แต่มีการใช้เครือข่ายประเภทนี้จริงน้อยมาก และช่วงหลังๆ แทบจะไม่มีมาตรฐานอีเธอร์เน็ตที่เป็นแบบบรอดแบนด์เลย เพราะฉะนั้นเมื่อกล่าวถึงอีเธอร์เน็ตมักจะหมายถึงการส่งข้อมูลแบบเบสแบนด์

เพื่อไม่เป็นการสับสนกับชื่อต่างๆ จึงขอสรุปการเรียกชื่อดังนี้ เมื่อกล่าวถึงอีเธอร์เน็ตจะหมายถึงอีเธอร์เน็ตแบบดั้งเดิมที่มีความเร็วที่ 10 Mbps ส่วนคำว่าอีเธอร์เน็ตความเร็วสูงหรือฟาสต์อีเธอร์จะหมายถึงอีเธอร์เน็ตที่ความเร็ว 100 Mbps ส่วนกิกะบิตอีเธอร์เน็ตจะหมายถึงอีเธอร์เน็ตที่มีความเร็วที่ 1,000 Mbps หรือ 1 Gbps และสุดท้ายเท็นกิกะบิตอีเธอร์เน็ตนั้น จะหมายถึงอีเธอร์เน็ตที่ความเร็ว 10 Gbps หรือบางทีก็เรียกว่า 10 GbE ก็ได้เช่นกัน

คณะชุดทำงานของสถาบัน IEEE ได้ออกแบบอีเธอร์เน็ตโดยการแบ่งแยกหน้าที่ หรือฟังก์ชันของเครือข่ายท้องถิ่น ออกเป็นส่วนย่อยๆหรือเลเยอร์ (Layer) ตามลำดับขั้นตอนของเหตุการณ์ต่างๆที่เกิดขึ้นระหว่างการสื่อสารผ่านเครือข่าย รูปที่ 2.2 แสดงการเปรียบเทียบเลเยอร์ต่างๆของอีเธอร์เน็ตกับแบบอ้างอิง OSI ซึ่งจะเห็นได้ว่ามีทั้งเลเยอร์ย่อยและมีโมดูลในบางเลเยอร์



รูปที่ 2.2 : เปรียบเทียบระหว่าง IEEE Ethernet และแบบอ้างอิง OSI

ดาต้าลิงค์เลเยอร์ (Data Link Layer)

IEEE ได้แบ่งชั้นเชื่อมโยงข้อมูลหรือดาต้าลิงค์เลเยอร์ออกเป็น 2 เลเยอร์ย่อย คือ LLC (Logical Link Control) และ MAC (Medium Access Control) ทั้งสองเลเยอร์ย่อยนี้ถือได้ว่าเป็นหัวใจของอีเธอร์เน็ต เนื่องจากเป็นเลเยอร์ที่สร้างเฟรมข้อมูลและที่อยู่ (Address) เป็นชั้นที่ทำให้ข้อมูลส่งถึงปลายทางได้อย่างถูกต้องและในสองเลเยอร์นี้ยังรับผิดชอบเกี่ยวกับกลไกการตรวจสอบข้อผิดพลาดที่อาจเกิดขึ้นระหว่างการรับส่งข้อมูล และถ้ามีข้อผิดพลาดเกิดขึ้นก็จะเตรียมการในการส่งข้อมูลใหม่ โดยสรุปก็คือเป็นเลเยอร์ที่ควบคุมการรับส่งข้อมูล ถึงแม้ว่าจะไม่ใช่เลเยอร์ที่ส่งข้อมูลจริงๆก็ตาม เลเยอร์ที่ทำการรับส่งข้อมูลจริงๆก็คือ ฟิสิคอลเลเยอร์นั่นเอง

- Logical Link Control

LLC (Logical Link Control) เป็นเลเยอร์ที่อยู่ด้านบนของดาต้าลิงค์เลเยอร์ ซึ่งจะให้บริการกับโปรโตคอลของเลเยอร์บนในการเข้าใช้สื่อกลางหรือสายสัญญาณในการรับส่งข้อมูล ตามมาตรฐาน IEEE 802.3 แล้ว จะอนุญาตให้สถาปัตยกรรมของเครือข่ายท้องถิ่นที่ต่างกันสามารถทำงานร่วมกันได้ กล่าวคือ โปรโตคอลเลเยอร์บนไม่จำเป็นต้องทราบว่าฟิสิคอลเลเยอร์ใช้สายสัญญาณประเภทใดในการรับส่งข้อมูล เพราะ LLC จะรับผิดชอบแทนในการปรับเฟรมข้อมูลให้สามารถส่งไปได้ในสายสัญญาณประเภทนั้นๆ LLC เป็นเลเยอร์ที่แยกเน็ตเวิร์คเลเยอร์ (Network Layer) ออกจากการเปลี่ยนแปลงบ่อยๆ ของสถาปัตยกรรมเครือข่ายท้องถิ่น โดยโปรโตคอลของเลเยอร์ที่สูงกว่าไม่จำเป็นต้องสนใจว่าแพ็กเก็ตนั้นจะส่งผ่านเครือข่ายแบบอีเธอร์เน็ต โทเคนริง หรือ ATM และไม่จำเป็นต้องรู้ว่าการส่งผ่านข้อมูลในฟิสิคอลเลเยอร์จะใช้การรับส่งข้อมูลแบบใด ชั้น LLC จะจัดการเรื่องเหล่านี้ให้ทั้งหมด

- Media Access Control

MAC (Media Access Control) เป็นเลเยอร์ที่อยู่ล่างสุดของดาต้าลิงค์เลเยอร์ ซึ่งจะทำหน้าที่เชื่อมต่อกับฟิสิคอลเลเยอร์ และรับผิดชอบในการรับส่งข้อมูลให้สำเร็จและถูกต้อง โดยจะแบ่งหน้าที่ออกเป็นสองส่วนคือ การส่งข้อมูลและการรับข้อมูล

MAC จะทำหน้าที่ห่อหุ้มข้อมูลที่ส่งผ่านจากชั้น LLC และทำให้อยู่ในรูปของเฟรมข้อมูล ซึ่งเฟรมข้อมูลนี้จะประกอบด้วยที่อยู่และข้อมูลต่างๆที่จำเป็นสำหรับการส่งข้อมูลให้ถึงปลายทาง ชั้น MAC ยังรับผิดชอบในการสร้างกลไกสำหรับตรวจสอบข้อผิดพลาดของข้อมูลในเฟรมนั้นๆ ในระหว่างการรับส่งเฟรมด้วย นอกจากนี้ MAC ยังต้องตรวจสอบฟิสิคอลเลเยอร์ว่าช่องสัญญาณพร้อมสำหรับการส่งข้อมูลหรือไม่ ถ้าพร้อมเฟรมก็จะถูกส่งต่อไปยังฟิสิคอลเลเยอร์เพื่อทำการส่งไปตามสายสัญญาณต่อไป แต่ถ้ายังไม่พร้อมชั้น MAC ก็จะรอจนกว่าจะว่าง แล้วค่อยทำการส่งข้อมูล

หน้าที่สุดท้ายของชั้น MAC คือการตรวจสอบสถานภาพของเฟรมที่กำลังส่ง ว่ามีการชนกันของข้อมูลเกิดขึ้นหรือไม่ ถ้าหากมีการชนกันเกิดขึ้นก็จะหยุดส่งข้อมูล และเข้าสู่กลไกการรอด้วยช่วงเวลาที่เป็นเลขสุ่มเพื่อทำการส่งข้อมูลใหม่อีกครั้ง ซึ่งจะทำเช่นนี้ไปเรื่อยๆ จนกว่าจะทำการส่งข้อมูลได้สำเร็จ กระบวนการส่งข้อมูลที่ว่านี้เป็นทั้งข้อดีและข้อเสียของอีเธอร์เน็ต ข้อดีก็คือเป็นการรับรองให้แก่โปรโตคอลชั้นที่อยู่เหนือกว่ามั่นใจว่าข้อมูลจะถูกส่งไปถึงปลายทางอย่างแน่นอน แต่ในขณะเดียวกันข้อเสียก็คือการส่งข้อมูลอาจใช้เวลานานมากถ้ามีการใช้เครือข่ายมากๆ

2.2.2 ฟอร์แมตเฟรมข้อมูล

ข้อมูลที่อยู่ในเลเยอร์ที่สองนั้นจะถูกจัดให้อยู่ในรูปของเฟรม ซึ่งตามมาตรฐาน IEEE 802.3 แล้วรูปแบบของเฟรมจะเป็นดังรูปที่ 2.3 โดยตัวเลขที่แสดงข้างบนเป็นจำนวนไบต์ของฟิลด์นั้นๆ

7	1	6	6	2	46-1500	4
Preamble	SOF	Destination Address	Source Address	Type/ Length	Data	FCS

รูปที่ 2.3 : ฟอร์แมตของอีเธอร์เน็ต

รายละเอียดของฟิลด์ต่างๆ ของเฟรมอีเธอร์เน็ตแสดงในรูปที่ 2.3 ดังนี้

- Preamble : เป็นฟิลด์ที่มีบิตสลับกันระหว่าง 1 กับ 0 ซึ่งเป็นสัญญาณบอกสถานีฝ่ายรับว่ากำลังมีข้อมูลส่งมา ฟิลด์นี้มีความยาว 8 ไบต์ โดยรวมเอาไบต์ของ SOF เข้าด้วย
- SOF (Start-of-Frame) : เป็นไบต์สุดท้ายของพรีแอมเบิล (Preamble) ซึ่งไบต์นี้จะแตกต่างจากไบต์อื่นๆคือ 2 บิตสุดท้ายจะเป็น 1 ทั้งคู่ เพื่อเป็นสัญญาณสำหรับบอกจุดเริ่มต้นของเฟรมจริงๆ
- Destination/Source Address : หมายเลข หรือที่อยู่ของสถานีปลายทางและต้นทางซึ่งมีความยาวส่วนละ 48 บิต ส่วนใหญ่จะเรียกว่า "แมคแอดเดรส (MAC Address)" การกำหนดหมายเลขนี้จะ

ควบคุมโดย IEEE ซึ่งมีเกณฑ์คือ 24 บิตแรกเป็นหมายเลขที่กำหนดให้กับบริษัทผู้ผลิต NIC (Network Interface Card) และบริษัทนั้นจะเป็นผู้กำหนดอีก 24 บิตที่เหลือ การทำเช่นนี้เพื่อให้แน่ใจว่าในเครือข่ายหนึ่งๆจะไม่มีหมายเลขนี้ซ้ำกัน

- Type/Length : เป็นฟิลด์ที่บอกประเภทของโปรโตคอลของเลเยอร์ที่อยู่เหนือกว่า เช่น IP, IPX, IPv6, ARP, AppleTalk เป็นต้น
- Data : ส่วนนี้จะเป็นฟิลด์ที่เก็บข้อมูลซึ่งมีความยาวอย่างน้อยต้องไม่ต่ำกว่า 46 ไบต์ ถ้าต่ำกว่านี้จะต้องมีฟิลด์เสริม (Padding) เพื่อให้ข้อมูลมีอย่างน้อย 46 ไบต์ เหตุที่ต้องกำหนดความยาวขั้นต่ำนี้ก็เพื่อสำหรับการตรวจเช็คการชนกันของข้อมูล (Collision) ในระหว่างการรับส่งข้อมูล ส่วนความยาวสูงสุดคือ 1,500 ไบต์
- FCS (Frame Check Sequence) : ฟิลด์นี้มีความยาว 4 ไบต์ ซึ่งเป็นโค้ดสำหรับตรวจสอบข้อผิดพลาดแบบ CRC (Cyclic Redundancy Check) ของข้อมูลในเฟรม

2.3 โปรโตคอลทีซีพีไอพี

ปัจจุบันโปรโตคอลทีซีพีไอพีเป็นโปรโตคอลที่นิยมใช้ในเครือข่ายมากที่สุด เหตุผลหนึ่งที่โปรโตคอลชุดนี้เป็นที่นิยมมาก เนื่องจากหลายบริษัทที่ผลิตอุปกรณ์หรือซอฟท์แวร์ของเครือข่ายนำมาใช้เป็นมาตรฐาน และอีกอย่างทีซีพีไอพีเป็นโปรโตคอลพื้นฐานของเครือข่ายอินเตอร์เน็ต (Internet) ซึ่งเป็นเครือข่ายที่ใหญ่ที่สุดในโลก และเป็นเครือข่ายที่ทำให้คอมพิวเตอร์กลายเป็นส่วนที่สำคัญในชีวิตประจำวันของเราในปัจจุบัน ดังนั้นทีซีพีไอพีจึงได้กลายโปรโตคอลมาตรฐานที่ใช้ในองค์กรธุรกิจและรัฐบาล

ชุดโปรโตคอลทีซีพีไอพีได้ถูกพัฒนามานานแล้วกว่า 30 ปี ซึ่งเริ่มจากการวิจัยที่สนับสนุนโดยกระทรวงกลาโหมสหรัฐฯ จุดประสงค์ของการวิจัยนี้ก็เพื่อเชื่อมต่อคอมพิวเตอร์ที่ต่างแพลตฟอร์มกันให้สามารถสื่อสารกันผ่านเครือข่ายได้ ซึ่งสามารถทำได้โดยการแบ่งโปรโตคอลเป็นชั้นและเป็นการแยกการทำงานของแอพพลิเคชันของผู้ใช้ออกจากฮาร์ดแวร์ที่ใช้รับส่งข้อมูลผ่านเครือข่าย ชุดโปรโตคอลนี้จะมีการจัดรูปแบบที่แตกต่างจากแบบอ้างอิง OSI เล็กน้อย

เครือข่ายคอมพิวเตอร์ในปัจจุบันประกอบด้วยหลากหลายอุปกรณ์ และฮาร์ดแวร์ที่ผลิตโดยบริษัทต่างๆแต่อุปกรณ์เครือข่ายเหล่านี้มีระบบการสื่อสารข้อมูลที่เหมือนกัน เครือข่ายที่ใช้อุปกรณ์จากหลายบริษัทนี้สามารถทำงานร่วมกันได้ เนื่องจากอุปกรณ์แต่ละชิ้นผลิตตามมาตรฐานที่กำหนดโดยองค์กรกลาง โปรโตคอลทีซีพีไอพีเป็นมาตรฐานที่ได้รับความนิยมมากที่สุดสำหรับเครือข่ายในปัจจุบันเนื่องจากหลายเหตุผลดังนี้

- เป็นโปรโตคอลระบบเปิด (Open System) ที่ไม่มีบริษัทใดบริษัทหนึ่งเป็นเจ้าของลิขสิทธิ์และข้อกำหนดของโปรโตคอลจะถูกพัฒนาโดยองค์กรสาธารณะและมีการตีพิมพ์ให้ทราบ
- ทีซีพีไอพีถูกออกแบบมาเพื่อให้แพลตฟอร์มต่างกันสามารถสื่อสารกันได้ โปรแกรมบริการต่างๆ เช่น FTP (File Transfer Protocol) และเทลเน็ต เป็นโปรแกรมที่ไม่ขึ้นต่อระบบ เพียงแต่บริษัทนั้นๆ พัฒนาระบบของตัวเองให้สามารถรองรับทีซีพีไอพีได้ ก็สามารถสื่อสารกับระบบอื่นได้

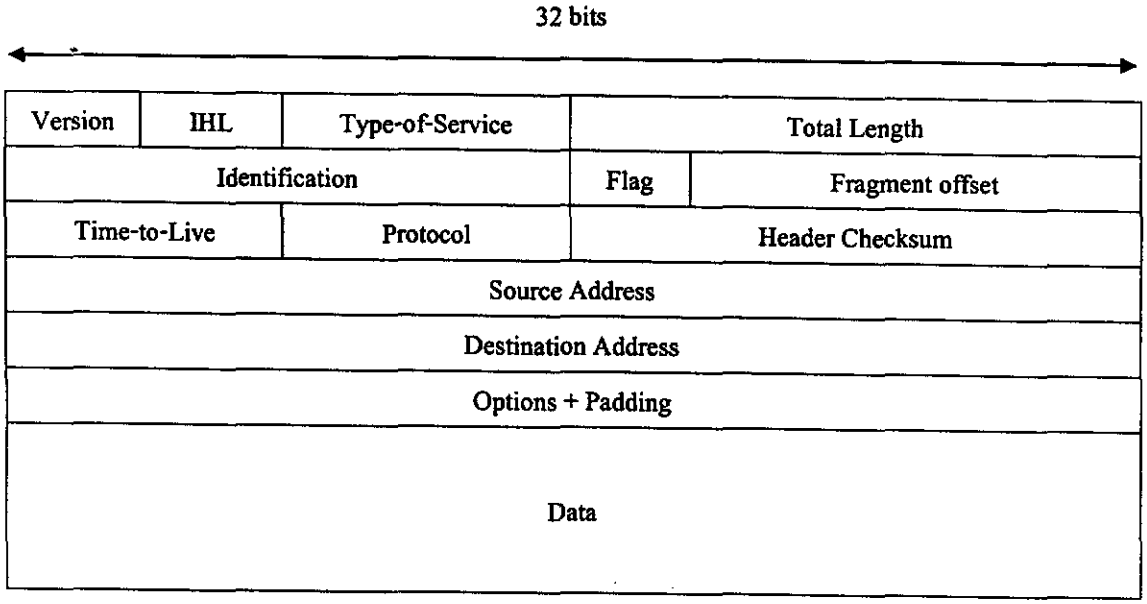
- โปรโตคอลทีซีพีไอพีได้ถูกพิสูจน์แล้วว่าเป็นโปรโตคอลที่แข็งแกร่ง มีประสิทธิภาพสูง และมีความสามารถในการขยายตัวสูง ด้วยการใช้งานในเครือข่ายอินเตอร์เน็ตซึ่งเป็นเครือข่ายที่ใหญ่ที่สุดในโลก
- โปรโตคอลทีซีพีไอพีได้กลายเป็นโปรโตคอลมาตรฐานกลางในการสื่อสารข้อมูลของคอมพิวเตอร์เนื่องจากเป็นภาษาที่ใช้ในระบบอินเตอร์เน็ต

ไม่ว่าจะด้วยเหตุผลอะไรก็แล้วแต่ที่ทำให้โปรโตคอลทีซีพีไอพีเป็นโปรโตคอลสุดยอดนิยมในปัจจุบันแต่สิ่งที่สำคัญที่สุดคือโปรโตคอลนี้ได้กลายเป็นมาตรฐานเครือข่ายๆไปโดยปริยาย ดังนั้นเครือข่ายสมัยใหม่จึงจำเป็นที่ต้องสร้างให้สามารถรองรับโปรโตคอลนี้

2.3.1 Internet Protocol (IP)

โปรโตคอลไอพี (IP) ทำหน้าที่เสมือนกับที่ทำการไปรษณีย์ กล่าวคือ โปรโตคอลไอพี จะทำหน้าที่จัดการเกี่ยวกับการรับส่งแพ็กเก็ต หรือบางทีก็เรียกว่า "ดาต้าแกรม (Datagram)" คือหน่วยข้อมูลที่รับมาจากโปรโตคอลที่อยู่ในเลเยอร์ที่สูงกว่า เช่น ทีซีพี (TCP) และยูดีพี (UDP) ถ้าโฮสต์ปลายทางอยู่คนละเครือข่ายกับโฮสต์ที่ส่งข้อมูล โปรโตคอลไอพีจะรับผิดชอบในการจัดเส้นทาง (Routing) ให้แพ็กเก็ตส่งไปยังเครือข่ายที่โฮสต์นั้นอยู่ ซึ่งในการจัดส่งแพ็กเก็ตข้ามเครือข่ายนั้น จะใช้เราท์เตอร์ (Router) ในการเชื่อมต่อเครือข่ายเหล่านั้นโดยทั่วไปแล้วอุปกรณ์ที่ทำหน้าที่รับส่งข้อมูลระหว่างเครือข่ายจะเรียกว่า "เราท์เตอร์" แต่บางทีอุปกรณ์ตัวนี้ก็จะเรียกว่า "เกตเวย์ (Gateway)" ซึ่งทำหน้าที่เป็นเสมือนประตูไปยังเครือข่ายอื่นๆ อย่างไรก็ตามทั้งเราท์เตอร์และเกตเวย์เป็นอุปกรณ์ที่ทำหน้าที่ในเลเยอร์ที่ 3 เหมือนกัน

โปรโตคอลไอพี เป็นโปรโตคอลที่ให้บริการแบบคอนเน็กชันเลสส์ (Connectionless) ซึ่งทำให้มีความน่าเชื่อถือน้อย เนื่องจากไม่มีการสร้างการเชื่อมต่อก่อนที่จะทำการรับส่งข้อมูล กล่าวคือในการส่งข้อมูลแต่ละครั้ง โฮสต์ที่ต้องการส่งข้อมูลจะไม่ทำการติดต่อโฮสต์ปลายทางเพื่อตกลงเกี่ยวกับการรับส่งข้อมูลก่อน แต่โฮสต์ที่ต้องการส่งข้อมูลจะทำการส่งแพ็กเก็ตออกไปทันที โดยที่คาดหวังว่าโฮสต์ปลายทางจะได้รับแพ็กเก็ตนั้นในที่สุด ดังนั้นความน่าเชื่อถือในการส่งข้อมูลจึงมีน้อยเพราะแพ็กเก็ตข้อมูลอาจสูญหายระหว่างทางหรือถ้าข้อมูลประกอบด้วยหลายแพ็กเก็ต แต่ละแพ็กเก็ตอาจเดินทางมาถึงปลายทางไม่เป็นลำดับได้ หรือมีการส่งแพ็กเก็ตซ้ำกันหรือแพ็กเก็ตส่งถึงล่าช้า การแก้ปัญหานี้จะปล่อยให้เป็นหน้าที่ของโปรโตคอลที่อยู่ในเลเยอร์ที่สูงกว่ารับผิดชอบ



รูปที่ 2.4 : ฟอร์แมตของแพ็กเก็ตไอพี

ฟอร์แมตของแพ็กเก็ตไอพี ประกอบด้วยหลายฟิลด์ดังแสดงในรูปที่ 2.4 ข้อมูลในส่วนหัวของแพ็กเก็ตไอพี มีดังนี้

- Version (4 บิต) : ข้อมูล 4 บิตแรกจะเป็นข้อมูลที่บอกถึงเวอร์ชันของโปรโตคอลทีซีพีไอพี ที่ใช้อยู่ ซึ่งในปัจจุบันจะใช้เวอร์ชัน 4 หรือเรียกสั้นๆว่า IPv4 ในอนาคตอันใกล้อาจจะมีการเปลี่ยนแปลงไปใช้เวอร์ชันใหม่คือ เวอร์ชัน 6 หรือ IPv6 เนื่องจากเวอร์ชัน 4 กำลังมีปัญหาเกี่ยวกับหมายเลขไอพีไม่เพียงพอต่อการใช้งาน
- Internet Header Length หรือ IHL (4 บิต) : เป็นตัวเลขที่บอกความยาวของข้อมูลในส่วนหัว (Header)
- Type of Service (8 บิต) : ในแต่ละบิตของข้อมูลส่วนนี้จะเป็นแฟล็ก (Flag) ที่แสดงถึงลำดับความสำคัญ (Precedence), ความล่าช้า (Delay), อัตราการส่งผ่าน (Throughput) และค่ากำหนดความเชื่อถือได้ของแพ็กเก็ตข้อมูลนี้
- Total Length (16 บิต) : ข้อมูลส่วนนี้จะบอกถึงความยาวของแพ็กเก็ตทั้งหมดซึ่งมีหน่วยเป็นไบต์ ซึ่งความยาวของแพ็กเก็ตนี้เป็นไปได้ตั้งแต่ 576 ถึง 65,536 ไบต์
- Identifier (16 บิต) : ถ้าดาต้าแกรมประกอบด้วยหลายแพ็กเก็ต หมายเลขนี้จะถูกกำหนดให้กับแต่ละแพ็กเก็ตย่อย ซึ่งแพ็กเก็ตย่อยแต่ละแพ็กเก็ตจะมีหมายเลขนี้ที่ไม่ซ้ำกันในช่วงเวลาหนึ่งๆ
- Flag (3 บิต) : เป็นฟิลด์ที่ใช้ในการจัดการเกี่ยวกับการแบ่งข้อมูลเป็นแพ็กเก็ตย่อย
- Fragment Offset (13 บิต) : เป็นค่าที่บอกจุดเริ่มต้นในส่วนของข้อมูลย่อย (Fragmented Content) ซึ่งเป็นตัวเลขที่บ่งบอกว่าแพ็กเก็ตย่อยนี้อยู่ห่างจากจุดเริ่มต้นของดาต้าแกรมทั้งหมดเท่าใด โดยจำนวนนี้มีหน่วยวัดเป็น 64 บิต

- Time to Live หรือ TTL (8 บิต) : แพ็กเก็ตจะไหลเวียนอยู่ในเครือข่ายได้ในช่วงเวลาหนึ่งเท่านั้น การกำหนดว่าแพ็กเก็ตแต่ละแพ็กเก็ตจะอยู่ได้ในเครือข่ายนานเท่าใดนั้น จะบอกเป็นจำนวนของ Hop หรือจำนวนครั้งที่ผ่านเราท์เตอร์ ทุกครั้งที่ผ่านเราท์เตอร์ค่า TTL จะลดลงทีละหนึ่ง เมื่อค่านี้เป็นศูนย์แพ็กเก็ตนี้ก็จะถูกละทิ้งไป
- Protocol (8 บิต) : เป็นข้อมูลที่บอกโปรโตคอลของชั้นที่เหนือกว่าเช่น ทีซีพี, ยูดีพี เป็นต้น
- Header Checksum (16 บิต) : เป็นข้อมูลส่วนที่ใช้ในการตรวจสอบข้อผิดพลาดในส่วนหัวของแพ็กเก็ต ซึ่งเมื่อผ่านอุปกรณ์เครือข่ายแต่ละครั้งจะทำการเช็คข้อผิดพลาดทุกครั้ง
- Source IP Address (32 บิต) : หมายเลขไอพีของเครื่องที่ส่งข้อมูล
- Destination IP Address (32 บิต) : หมายเลขไอพีของเครื่องปลายทาง
- Padding : เป็นเลข 0 ที่เพิ่มให้กับส่วนหัวของแพ็กเก็ตเพื่อให้ส่วนหัวมีความยาวที่หารด้วย 32 บิตลงตัว หรือเป็นข้อมูลที่เกี่ยวกับฟีเจอร์อื่นๆ เช่น การรักษาความปลอดภัย
- Data : ข้อมูลของโปรโตคอลที่อยู่สูงกว่าซึ่งความยาวจะไม่คงที่

หน้าที่หลักๆของโปรโตคอลไอพี มีอยู่ 3 อย่างดังต่อไปนี้

- Addressing

หน้าที่นี้หมายถึงการให้บริการในการติดตั้ง "ลอจิคัลแอดเดรส (Logical Address)"ให้กับเครื่องคอมพิวเตอร์ต่างๆที่ใช้โปรโตคอลไอพี เนื่องจาก ลอจิคัลแอดเดรสนี้จะไม่ได้ถูกกำหนดมาตายตัวหรือฝังมากับเน็ตเวิร์คการ์ด ดังนั้นมันจึงเป็นแอดเดรสที่ผู้ออกแบบหรือบริหารระบบเครือข่ายเป็นผู้ตั้งขึ้นมาเองและสามารถเปลี่ยนแปลงแก้ไขได้ตามใจชอบ ข้อดีของการมีลอจิคัลแอดเดรสหรือแอดเดรสในเน็ตเวิร์คเลเยอร์ (Network Layer) ก็คือ

- ทำให้เราสามารถออกแบบระบบเน็ตเวิร์คได้ง่ายขึ้น
- ทำให้ระบบเน็ตเวิร์คสามารถขยายเพิ่มเติมได้โดยง่าย
- ทำให้การแก้ไขปัญหาทำได้โดยง่าย

- Packaging

เป็นการจัดเตรียมแพ็กเก็ตไอพีให้อยู่ในสภาพที่พร้อมส่งไปยังเครื่องปลายทางโดยการนำเอาทีซีพีเซ็กเมนต์ (TCP Segment) หรือยูดีพีเซ็กเมนต์ (UDP Segment) จากเลเยอร์บนมาบรรจุไว้ในฟิลด์ดาต้า (Data) ของแพ็กเก็ตไอพี หากขนาดของเซ็กเมนต์ ใหญ่เกินกว่าจะส่งได้ภายในแพ็กเก็ตไอพี แพ็กเก็ตเดียวมันจะต้องแบ่งเซ็กเมนต์ ซอยย่อยออกและส่งไปในหลายๆแพ็กเก็ต จากนั้นก็ใส่ค่าฟิลด์ Destination Address และ Source Address ให้เป็นหมายเลขไอพีแอดเดรสปลายทางและต้นทางตามลำดับ และที่สำคัญก็คือ มันจะใส่ค่าฟิลด์ Protocol Number ลงไปด้วยตัวเลขค่าหนึ่งที่ระบุว่าเลเยอร์บนเป็นทีซีพีหรือยูดีพี (หมายเลข 6 สำหรับทีซีพีและ หมายเลข 17 สำหรับยูดีพี) แพ็กเก็ตไอพีหนึ่งๆ บางครั้งถูกเรียกว่า ดาต้าแกรม

- Routing

ความหมายของ Routing ก็คือ การหาเส้นทางในการส่งแพ็กเก็ตไปให้ถึงเครื่องปลายทางให้ได้ หลักสำคัญของการส่งแพ็กเก็ตโดยโปรโตคอลไอพีก็คือ มันจะส่งให้ดีที่สุด (Best Effort) โดยไม่รับประกันว่าข้อมูลจะถึงปลายทางหรือไม่ และจะปล่อยให้เป็นหน้าที่ของโปรโตคอลในระดับสูงกว่า (คือทีซีพี) เป็นผู้รับประกันให้

2.3.2 Transmission Control Protocol (TCP)

โปรโตคอลทีซีพี เป็นโปรโตคอลที่ให้บริการแบบคอนเน็กชันไอเรียนเต็ด (Connection-Oriented) ซึ่งเป็นการส่งข้อมูลที่เชื่อถือได้ โปรโตคอลทีซีพีจะส่งข้อมูลทั้งหมดจนสำเร็จ ซึ่งถ้าข้อมูลมีขนาดใหญ่ก็จะถูกแบ่งย่อยเป็นหลายแพ็กเก็ต โปรโตคอลทีซีพี จะทำหน้าที่ควบคุมการรับส่งแพ็กเก็ตข้อมูลย่อยๆเหล่านี้ สำหรับกลไกในการควบคุมการไหลของข้อมูลมีรายละเอียดดังนี้

- การจัดการเกี่ยวกับเซสชั่น (Session)

เนื่องจากทีซีพีเป็นโปรโตคอลที่ให้บริการแบบคอนเน็กชันไอเรียนเต็ด ดังนั้นก่อนที่จะมีการส่งข้อมูล จำเป็นที่ต้องสร้างเซสชั่นเพื่อเชื่อมต่อกับโฮสต์ปลายทางก่อน เซสชั่นเป็นการสร้างการสนทนาอย่างเป็นรูปแบบระหว่างทั้งสองโฮสต์เพื่อใช้สำหรับการกู้คืนข้อมูลเมื่อเกิดข้อผิดพลาดระหว่างการรับส่งข้อมูล ขั้นตอนในการสร้างเซสชั่นนี้จะมีอยู่ 3 ขั้นตอนซึ่งบางทีก็เรียกว่า "ทรีเวย์แฮนเช็ค (Three-Way Handshake)"

1. โฮสต์ที่ต้องการส่งข้อมูลจะส่งแพ็กเก็ตไปยังโฮสต์ปลายทางเพื่อแจ้งให้ทราบว่าต้องการส่งข้อมูล
2. โฮสต์ปลายทางก็จะตอบตกลงกลับมาพร้อมทั้งรหัสที่จะใช้ในการรับส่งข้อมูล
3. โฮสต์ต้นทางก็จะส่งแพ็กเก็ตพร้อมรหัสที่ได้รับ เพื่อเป็นการยืนยันการเชื่อมต่อ

หลังจากที่ได้มีการสร้างเซสชั่นสำเร็จแล้วจึงจะเริ่มกระบวนการรับส่งข้อมูลจริงๆ ซึ่งการรับส่งข้อมูลแต่ละครั้งก็จะมีการยืนยันการรับข้อมูลจากโฮสต์ปลายทางทุกครั้ง เมื่อรับส่งข้อมูลเสร็จก็เป็นขั้นตอนการยกเลิกการเซสชั่น ซึ่งจะคล้ายๆกับการสร้างเซสชั่น

- การควบคุมการไหลและการกู้คืนข้อมูล

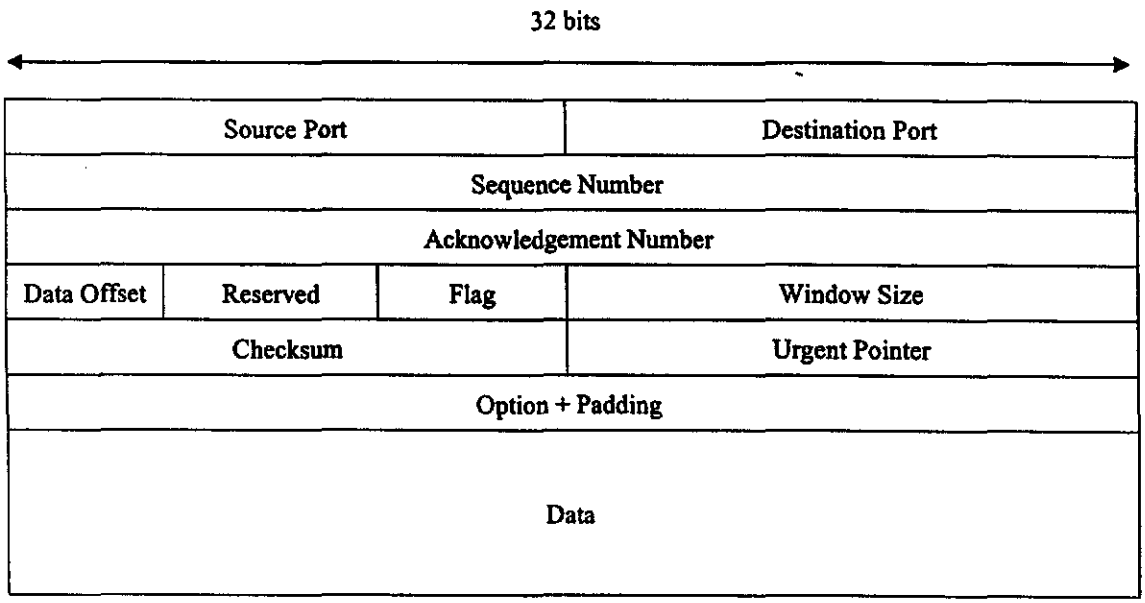
ในแต่ละเซสชั่น โฮสต์ฝ่ายรับต้องตอบกลับทุกๆแพ็กเก็ตที่ได้รับภายในเวลาที่กำหนด เพื่อเป็นการยืนยันการรับข้อมูลทุกๆแพ็กเก็ตที่ส่ง ฝ่ายรับจะทำการเช็คความถูกต้องของแพ็กเก็ตข้อมูลทุกครั้ง และแจ้งให้ทราบถึงผลการตรวจสอบนั้น ถ้าฝ่ายส่งไม่ได้รับการตอบกลับจากฝ่ายรับภายในเวลาที่กำหนดก็จะคาดเดาว่าแพ็กเก็ตสูญหายระหว่างทาง ฝ่ายส่งก็จะทำการส่งแพ็กเก็ตนั้นใหม่อีกครั้ง เพื่อทำให้มั่นใจได้ว่าข้อมูลทุกๆแพ็กเก็ตส่งถึงปลายทางอย่างสมบูรณ์ นอกจากนี้การแบ่งข้อมูลขนาดใหญ่ออกเป็นแพ็กเก็ตย่อยๆ ทีซีพีก็จะกำหนดหมายเลขลำดับ (Sequence Number) ให้แต่ละแพ็กเก็ต เพื่อใช้สำหรับการจัดรวมแพ็กเก็ตย่อยๆเหล่านั้นให้เป็นข้อมูลเหมือนเดิม นอกจากนี้หมายเลขลำดับยังใช้สำหรับการตรวจสอบว่า

ข้อมูลส่งถึงปลายทางครบทุกแพ็กเก็ตหรือไม่ กลไกการตอบกลับแพ็กเก็ตนั้นมีอยู่ 2 ประเภท ประเภทแรกคือ PAR (Positive Acknowledgement and Retransmission) กลไกการทำงานก็คือ เมื่อฝ่ายส่งทำการส่งแพ็กเก็ตหนึ่งก็จะรอการตอบกลับจากฝ่ายรับ แล้วค่อยส่งแพ็กเก็ตต่อไป ถ้าไม่ได้รับการตอบกลับภายในเวลาที่กำหนดก็จะส่งแพ็กเก็ตนั้นอีกครั้ง ปัญหาของกลไกนี้ก็คือ ถ้าข้อมูลประกอบด้วยหลายๆแพ็กเก็ต และการที่ฝ่ายรับต้องส่งแพ็กเก็ตตอบกลับต่อทุกๆแพ็กเก็ตที่ได้รับนั้นอาจเป็นการสิ้นเปลืองแบนด์วิธ และเป็นกระบวนการที่ไร้ประสิทธิภาพเนื่องจากฝ่ายส่งจะใช้เวลาในการรอมากกว่าการส่งข้อมูล กลไกที่สองจะแก้ปัญหานี้ซึ่งกลไกนี้จะเรียกว่า "สไลดิ้งวินโดว์ (Sliding Window)" กลไกการทำงานคือ ฝ่ายรับสามารถยืนยันการได้รับแพ็กเก็ตโดยส่งแพ็กเก็ตเดียวเพื่อแสดงการยืนยันการได้รับหลายๆแพ็กเก็ต วิธีนี้ช่วยลดจำนวนแพ็กเก็ตที่ต้องไหลเวียนในเครือข่าย และฝ่ายส่งสามารถส่งทีละหลายๆแพ็กเก็ตก่อนที่จะรอการตอบกลับ

เมื่อสร้างเซสชั่นสำเร็จ ขั้นตอนต่อไปคือการต่อรองเกี่ยวกับขนาดของวินโดว์ (Window Size) ขนาดของวินโดว์คือ จำนวนไบต์ที่ฝ่ายรับได้รับก่อนที่จะทำการตอบกลับหรือจำนวนไบต์ที่ฝ่ายส่งสามารถส่งได้ก่อนที่จะรอการตอบกลับ การทำงานของสไลดิ้งวินโดว์มีขั้นตอนดังนี้

1. เมื่อโฮสต์ต้องการที่จะส่งข้อมูล ทีซีพีจะย้ายข้อมูลไปไว้ที่บัฟเฟอร์ที่จะใช้ส่งข้อมูล ซึ่งข้อมูลส่วนนี้จะเรียกว่า "เซ็กเมนต์" ซึ่งแต่ละเซ็กเมนต์อาจจะถูกแบ่งย่อยเป็นหลายแพ็กเก็ตซึ่งแต่ละแพ็กเก็ตก็จะถูกกำหนดหมายเลขลำดับ
2. ทุกๆแพ็กเก็ตในเซ็กเมนต์จะถูกส่งต่อไปให้ส่วนไอพี เพื่อทำการส่งไปยังโฮสต์ปลายทาง
3. เซ็กเมนต์ข้อมูลจะยังคงถูกเก็บไว้ที่บัฟเฟอร์จนกว่าจะได้รับการตอบกลับจากโฮสต์ฝ่ายรับก่อนและโฮสต์ฝ่ายส่งจะตั้งเวลาเพื่อรอการตอบกลับ ถ้าโฮสต์ฝ่ายรับไม่ตอบกลับภายในเวลาที่กำหนดข้อมูลที่อยู่ในบัฟเฟอร์ก็จะถูกส่งใหม่อีกครั้ง
4. เมื่อแพ็กเก็ตเดินทางมาถึงฝ่ายรับ โฮสต์ฝ่ายรับก็จะใช้หมายเลขลำดับในการเรียบเรียงแพ็กเก็ตให้ได้เป็นเซ็กเมนต์เหมือนเดิม
5. เมื่อโฮสต์ฝ่ายรับได้รับแพ็กเก็ตครบและตรวจสอบแล้วว่าไม่มีข้อผิดพลาดใดๆก็จะส่งแพ็กเก็ตตอบกลับไปยังโฮสต์ฝ่ายส่งว่าได้รับข้อมูลครบหมดแล้ว
6. เมื่อโฮสต์ฝ่ายส่งได้รับการตอบกลับ เซ็กเมนต์ในบัฟเฟอร์ก็จะถูกลบทิ้งไปแล้วทำการส่งเซ็กเมนต์ถัดไป จนกว่าข้อมูลจะถูกส่งทั้งหมด

กระบวนการส่งข้อมูลแบบนี้จะทำให้มั่นใจได้ว่าข้อมูลจะส่งถึงปลายทางอย่างแน่นอนและถูกต้อง ซึ่งการให้บริการแบบนี้จะเรียกว่า "คอนเน็กชันโอเรียนเต็ด (Connection-Oriented)" นั่นเอง



รูปที่ 2.5 : ฟอร์แมตข้อมูลของแพ็กเก็ตทีซีพี

ข้อมูลในส่วนหัวของโปรโตคอลทีซีพี จะประกอบด้วยข้อมูลมากที่สุด 20 ไบต์และประกอบด้วยส่วนต่างๆ ดังแสดงในรูปที่ 2.5 ซึ่งแต่ละฟิลด์มีความหมายดังนี้

- TCP Source Port (16 บิต) : ส่วนนี้จะเป็นหมายเลขพอร์ตที่เป็นจุดเริ่มต้นการสื่อสาร หมายเลขพอร์ตเมื่อรวมกับหมายเลขไอพี จะเป็นที่อยู่ของการส่งข้อมูลกลับไป
- TCP Destination Port (16 บิต) : เป็นหมายเลขพอร์ตของเครื่องรับ ซึ่งพอร์ตนี้จะเป็นพอร์ตที่ใช้เชื่อมต่อกับแอพพลิเคชันที่จะนำข้อมูลที่ส่งไปให้นี้ไปประมวลผลต่อไป
- TCP Sequence Number (32 บิต) : เป็นหมายเลขที่บอกลำดับแพ็กเก็ตที่จะใช้โดยฝั่งเครื่องรับในการเรียงข้อมูลให้อยู่ในรูปแบบเดิม ในการส่งข้อมูลผ่านเครือข่ายที่สลับซับซ้อนนั้นแพ็กเก็ตแต่ละชุดอาจจะถูกส่งไปบนเส้นทางที่ต่างกัน ดังนั้นจึงเป็นไปได้ที่แพ็กเก็ตจะเดินทางมาถึงปลายทางไม่เป็นไปตามลำดับที่ส่ง หมายเลขนี้จะใช้ในการจัดเรียงแพ็กเก็ตเหล่านี้ให้อยู่ในลำดับเดิม
- TCP Acknowledgement Number (32 บิต) : เป็นหมายเลขลำดับแพ็กเก็ตถัดไปที่ทางฝั่งรับคาดหวัง ซึ่งเป็นการบอกเป็นนัยว่าแพ็กเก็ตที่มีหมายเลขลำดับก่อนหน้านี้ได้รับหมดแล้วนั่นเอง
- Data Offset (4 บิต) : เป็นตัวเลขที่บอกขนาดของข้อมูลส่วนหัว (TCP Header) ซึ่งมีหน่วยเป็น 32 บิต หรือ word
- Reserved (6 บิต) : ส่วนนี้จะถูกกำหนดให้เป็นศูนย์ตลอด ซึ่งข้อมูลส่วนนี้ไม่มีความหมายอะไร เพียงแต่เป็นการสงวนไว้ใช้ในอนาคตเมื่อมีการปรับปรุงโปรโตคอล
- Flag (6 บิต) : เป็นข้อมูลที่ใช้สำหรับควบคุมการรับส่งแพ็กเก็ต เช่น บิต SYN และ ACK ใช้สำหรับการสร้างการเชื่อมต่อ ส่วนบิต FIN เป็นการแจ้งการยกเลิกการเชื่อมต่อ เป็นต้น

- Window Size (16 บิต) : เป็นตัวเลขที่เครื่องปลายทางบอกให้เครื่องต้นทางทราบขนาดของวินโดวส์ที่เครื่องปลายทางสามารถรับข้อมูลได้
- Checksum (16 บิต) : เป็นข้อมูลที่ใช้ในการตรวจสอบข้อผิดพลาดของข้อมูลในส่วนหัว โดยเครื่องส่งจะทำการคำนวณค่าเช็คซัม (Checksum) ของข้อมูลส่วนหัว เมื่อเครื่องปลายทางได้รับข้อมูลก็จะทำการคำนวณเช็คซัมด้วยวิธีเดียวกัน แล้วทำการเปรียบเทียบค่าที่คำนวณได้กับค่าที่อยู่ในฟิลด์นี้ ถ้าเหมือนกันแสดงว่าไม่มีข้อผิดพลาดในข้อมูลที่ได้รับ
- Padding : เป็นข้อมูลส่วนเพิ่มเติมเพื่อให้ข้อมูลส่วนหัวมีจำนวนบิตที่หารด้วย 32 ลงตัว

- หมายเลขพอร์ต

คอมพิวเตอร์ที่ใช้โปรโตคอลทีซีพีไอพี ส่วนใหญ่จะมีแอพพลิเคชันหลายตัวที่ใช้โปรโตคอลทีซีพีไอพีในการสื่อสารกับเครื่องอื่น ซึ่งโปรโตคอลทีซีพีไอพี จะจัดส่งข้อมูลไปยังแอพพลิเคชันที่เหมาะสม เพื่อให้โปรโตคอลทีซีพีไอพีสามารถรองรับแอพพลิเคชันหลายแอพพลิเคชันในเครื่องเดียว จึงมีการใช้พอร์ตและซ็อกเก็ต (Socket) เพื่อช่วยในการแยกแยะแอพพลิเคชันต่างๆ

แอพพลิเคชันแต่ละตัวที่จะรับส่งข้อมูลผ่านเครือข่ายจะใช้หมายเลขพอร์ตตั้งแต่ 0 ถึง 65,535 ดังนั้นเพื่อให้การรับส่งข้อมูลถูกต้อง แอพพลิเคชันที่ทำงานในเครื่องเดียวกันจะต้องใช้หมายเลขพอร์ตที่ต่างกันเพื่อช่วยลดความสับสน แอพพลิเคชันที่นิยมใช้กันทั่วไปส่วนใหญ่จะถูกกำหนดให้ใช้หมายเลขพอร์ตใดพอร์ตหนึ่ง ซึ่งองค์กรที่ทำหน้าที่กำหนดหมายเลขนี้คือ IANA (Internet Assigned Numbers Authority) หมายเลขพอร์ตเหล่านี้จะถูกตีพิมพ์ใน RFC 1700 ซึ่งพอร์ตสำหรับแอพพลิเคชันที่นิยมใช้ทั่วไปได้แสดงในตารางที่ 2.1

พอร์ต	โปรโตคอล	แอพพลิเคชัน
20	TCP	FTP (Data)
21	TCP	FTP (Control)
23	TCP	Telnet
25	TCP	SMTP (Email)
53	TCP/UDP	DNS (Domain Name Server)
80	TCP	HTTP (Web Server)
110	TCP	POP3 (Email)
161	UDP	SNMP (Simple Network Management Protocol)

ตารางที่ 2.1 : หมายเลขพอร์ตของบางแอพพลิเคชัน

62558

โปรโตคอลทีซีพีไอพี จะแยกแยะแอพพลิเคชันที่ทำงานในแต่ละโฮสต์โดยใช้ข้อมูล 3 ส่วนต่อไปนี้

1. หมายเลขไอพีของโฮสต์นั้น
2. ประเภทของโปรโตคอลในชั้นทรานสปอร์ตเลเยอร์ (ทีซีพีหรือยูดีพี)
3. หมายเลขพอร์ตที่แอพพลิเคชันนั้นใช้

ตารางที่ 2.1 แสดงหมายเลขพอร์ตที่ถูกใช้โดยแอพพลิเคชันที่เป็นที่รู้จักกันโดยทั่วไป ส่วนเครื่องไคลเอนท์ (Client) ที่เชื่อมต่อกับเซิร์ฟเวอร์ (Server) จะใช้หมายเลขพอร์ตที่ต่างจากเซิร์ฟเวอร์ ซึ่งหมายเลขพอร์ตที่ใช้บนเครื่องไคลเอนท์จะถูกจัดการโดยระบบปฏิบัติการของเครื่องนั้น กล่าวคือถ้าไคลเอนท์จะทำการเชื่อมต่อกับเว็บเซิร์ฟเวอร์ หมายเลขพอร์ตที่เว็บเซิร์ฟเวอร์ใช้ก็คือ พอร์ต 80 แต่เครื่องไคลเอนท์จะใช้หมายเลขพอร์ตอื่นที่ว่างอยู่

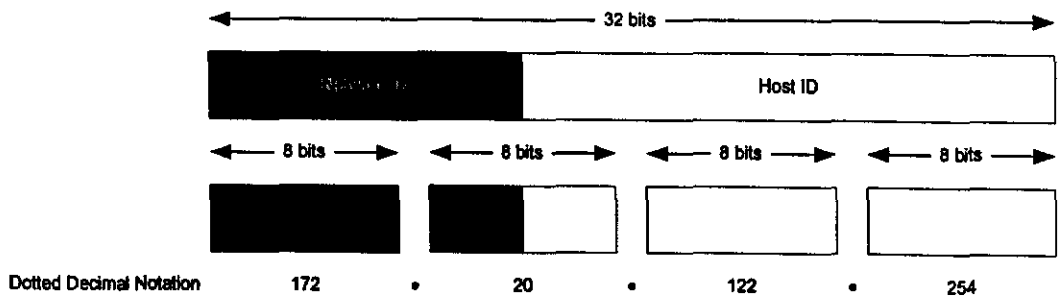
- Well Known Port

คำว่า "Well Known Port" จะหมายถึงหมายเลขพอร์ตที่ทางองค์กรกลางที่ชื่อ IETF (Internet Engineering Task Force) เป็นผู้กำหนดให้กับเซอร์วิสที่รู้จักกันดีและนิยมใช้กันแพร่หลายในเครือข่ายอินทราเน็ต (Intranet) และอินเตอร์เน็ต ค่าของ Well Known Port จะอยู่ระหว่าง 1 ถึง 1,024 ดังนั้นแอพพลิเคชันหรือเซอร์วิสอื่นที่เขียนขึ้นมาทำงานบนโปรโตคอลทีซีพีไอพี จึงสามารถตั้งค่าพอร์ตของตนเองได้ เป็นค่าที่มากกว่า 1,024 ขึ้นไป ตัวอย่างของ Well Known Port เช่น พอร์ตหมายเลข 80 เป็นของ HTTP พอร์ตหมายเลข 25 เป็นของ SMTP เป็นต้น หากต้องการทราบค่าของ Well Known Port ต่างๆที่นิยมใช้กัน สามารถเปิดดูได้จากไฟล์ชื่อ SERVICES สำหรับระบบปฏิบัติการวินโดวส์ NT 4.0, 2000, XP และ 2003 ไฟล์ SERVICES จะอยู่ที่โฟลเดอร์ C:\<winnt_root>\system32\driver\etc

2.4 IP Addressing

หมายเลขไอพี (IP Address) คือเลขที่บอกที่อยู่ของโนดหรือโฮสต์ที่อยู่ในเครือข่าย รวมถึงคอมพิวเตอร์และเราท์เตอร์ ที่อยู่บนระบบเครือข่าย หมายเลขนี้จะเป็นที่อยู่ในเลเยอร์ที่ 3 หรือเน็ตเวิร์คเลเยอร์ หมายเลขไอพีของแต่ละเครื่องที่อยู่ในเครือข่ายเดียวกันจะต้องไม่ซ้ำกัน

ปัจจุบันโปรโตคอลไอพีที่ใช้งานอยู่ในเครือข่ายอินเตอร์เน็ตจะเป็นเวอร์ชัน 4 หรือเรียกสั้นๆว่า IPv4 ซึ่งในเวอร์ชันนี้หมายเลขไอพีจะมีขนาด 32 บิต ซึ่งเป็นตัวเลขที่ยาวและยากต่อการจดจำ ดังนั้นเพื่อเป็นการง่ายหมายเลขไอพีจึงนิยมเขียนให้อยู่ในรูปแบบดือตเดซิมอล (Dotted Decimal Notation) การเขียนให้อยู่ในรูปแบบนี้จะทำได้โดยการจัดกลุ่มเลขฐานสองเป็น 4 กลุ่ม กลุ่มละ 8 บิต หลังจากนั้นให้แปลงเลขฐานสองแต่ละกลุ่มให้เป็นเลขฐานสิบแล้วเชื่อมเลขทั้งสี่กลุ่มด้วยจุด เนื่องจากหมายเลขไอพีที่เป็นฐานสิบนี้เป็นการแปลงมาจากเลขฐานสอง 8 บิต ดังนั้นเลขฐานสิบแต่ละส่วนจะต้องอยู่ระหว่าง 0 ถึง 255 เพราะฉะนั้นหมายเลขไอพีที่ถูกต้องจะอยู่ระหว่าง 0.0.0.0 ถึง 255.255.255.255



รูปที่ 2.6 : การแบ่งส่วนของหมายเลขไอพีและด็อตเดซิมอลโนเตชัน

2.4.1 ประเภทของหมายเลขไอพี

IPv4 ที่ใช้อยู่ในปัจจุบันจะแบ่งหมายเลขไอพีออกเป็น 5 ประเภท (Class) คือ A, B, C, D และ E โดยหมายเลขไอพีทั้ง 32 บิต จะถูกจัดให้เป็น 2 กลุ่มดังนี้คือ กลุ่มแรกจะเป็นตัวเลขที่ใช้บอกหมายเลขเครือข่าย (Network ID) และกลุ่มที่สองจะเป็นตัวเลขที่ใช้บอกหมายเลขโฮสต์ (Host ID) ที่อยู่ในเครือข่าย ข้อกำหนดที่ใช้ในการแบ่งประเภทของหมายเลขไอพีมีดังนี้

- Class A : บิตแรกจะเป็นเลข 0 เท่านั้นและส่วนที่บอกหมายเลขเครือข่าย คือ 8 บิตแรก ดังนั้นจะมีได้ทั้งหมด 126 เครือข่าย(หมายเลขเครือข่าย 0 จะไม่ใช้) ส่วนอีก 24 บิตที่เหลือจะเป็นเลขที่ใช้บอกหมายเลขโฮสต์ ดังนั้นในแต่ละเครือข่ายจะมีโฮสต์ได้ทั้งหมด 16,777,124 เครื่อง (หมายเลข 0.0.0 และ 255.255.255 จะไม่ใช้) เนื่องจากเครือข่ายมีจำนวนน้อยมากเมื่อเทียบกับจำนวนโฮสต์ ฉะนั้นหมายเลขไอพีประเภทนี้จึงไม่เหมาะสำหรับเครือข่ายขนาดใหญ่ ซึ่งประกอบด้วยหลายเครือข่ายเชื่อมต่อกัน เพราะในการส่งข้อมูลระหว่างเครือข่ายนั้นเราท์เตอร์จะใช้เฉพาะหมายเลขเครือข่ายเท่านั้น
- Class B : สองบิตแรกจะเป็น 10 เท่านั้น ส่วนหมายเลขเครือข่ายจะใช้ 16 บิตแรก ดังนั้นจะมีจำนวนเครือข่ายได้ทั้งหมด 16,382 เครือข่าย ส่วนอีก 16 บิตที่เหลือจะเป็นหมายเลขโฮสต์ ซึ่งจะทำให้แต่ละเครือข่ายมีโฮสต์ได้ทั้งหมด 65,534 เครื่อง
- Class C : มีบิตเริ่มต้นเป็น 110 ส่วนหมายเลขเครือข่ายจะใช้ 24 บิตแรก ซึ่งจะได้ทั้งหมด 2,097,152 เครือข่าย ส่วน 8 บิตสุดท้ายเป็นหมายเลขโฮสต์ ซึ่งมีทั้งหมด 254 เครื่อง
- Class D : เลขไอพีเริ่มต้นด้วย 1110 ซึ่งจะเป็นเลขไอพีที่ใช้สำหรับการมัลติคาสต์ หรือสำหรับการส่งข้อมูลแบบมีโฮสต์ปลายทางหลายเครื่อง แต่อาจจะอยู่คนละเครือข่ายกัน
- ประเภทสุดท้ายคือเลขไอพีที่เริ่มต้นด้วย 11110 เป็นหมายเลขไอพีที่สงวนไว้ใช้ในอนาคต หมายเลขเหล่านี้จะถูกกำหนดให้โดยศูนย์ข้อมูลเครือข่าย หรือ InterNIC (Internet Network Information Center)

จะเห็นได้ว่าในหมายเลขไอพีแต่ละประเภทจะมีหลายหมายเลขที่สงวนไว้ใช้สำหรับกรณีพิเศษ เช่น หมายเลข 0.0.0.0 จะใช้โดยโฮสต์ในขณะที่เริ่มเปิดเครื่อง เลขไอพีที่มีหมายเลขเครือข่ายเป็น 0 ทั้งหมดจะใช้อ้างอิงถึงเครือข่ายที่โฮสต์นั้นอยู่ เช่น ถ้าโฮสต์หนึ่งมีเลขไอพีเป็น 172.20.1.24 ซึ่งจะจัดอยู่ใน

ประเภท B โดยมีหมายเลขเครือข่ายเป็น 172.20 เมื่อโฮสต์นี้อ้างถึงหมายเลขไอพี 0.0.1.32 จะมีความหมายเช่นเดียวกับหมายเลข 172.20.1.32 เป็นต้น ส่วนหมายเลขไอพีที่ประกอบด้วยเลข 1 ทั้ง 32 บิตนั้นจะถูกใช้สำหรับการส่งข้อมูลแบบแพร่กระจาย (Broadcast) ในเครือข่ายนั้นๆ ส่วนเลขไอพีที่มีหมายเลขโฮสต์เป็นหนึ่งทั้งหมดนั้น จะใช้สำหรับการส่งข้อมูลแบบแพร่กระจายไปยังเครือข่ายนั้นๆ เช่นเลขไอพี 172.20.255.255 เป็นเลขที่ใช้สำหรับการส่งข้อมูลแบบแพร่กระจายไปยังโฮสต์ทุกเครื่องที่อยู่ในเครือข่าย 172.20 เป็นต้น ส่วนเลขหมาย 127.xx.yy.zz โดยเลข xx, yy และ zz จะเป็นเลขอะไรก็ได้ (แต่โดยทั่วไปจะนิยมใช้เป็นตัวเลข 127.0.0.1) นั้น จะใช้สำหรับการส่งข้อมูลไปยังตัวเอง (Loopback) ซึ่งข้อมูลจะไม่ถูกส่งออกไปในเครือข่าย แต่โฮสต์นั้นจะถือเสมือนว่าข้อมูลนั้นได้รับผ่านเครือข่าย

2.4.2 Private/Public Internet

การที่จะเชื่อมต่อคอมพิวเตอร์เข้ากับอินเตอร์เน็ตนั้นจำเป็นที่จะต้องร้องขอหมายเลขไอพีจาก InterNIC ซึ่งเป็นองค์กรที่รับผิดชอบเกี่ยวกับการแจกจ่ายหมายเลขไอพี และเป็นที่ทราบกันดีแล้วว่าหมายเลขไอพีของแต่ละเครื่องที่อยู่ในเครือข่ายเดียวกันจะต้องไม่ซ้ำกัน ดังนั้นถ้าเป็นเครือข่ายส่วนบุคคลที่ไม่มีการเชื่อมต่อเข้ากับอินเตอร์เน็ต ก็สามารถใช้หมายเลขไอพีอะไรก็ได้ และไม่ต้องร้องขอหมายเลขไอพีจาก InterNIC เพียงแค่กำหนดให้หมายเลขไอพีของโฮสต์ในเครือข่ายไม่ซ้ำกันก็พอ

อย่างไรก็ตาม เมื่อมีการเชื่อมต่อเครือข่ายส่วนบุคคลเข้ากับอินเตอร์เน็ต อาจจะทำให้หมายเลขไอพีที่ใช้ในเครือข่ายส่วนบุคคลไปซ้ำกับโฮสต์ที่อยู่ในอินเตอร์เน็ตแล้วก็ได้ ดังนั้นเพื่อป้องกันปัญหาดังกล่าว องค์กร IETF (Internet Engineering Task Force) ได้กำหนดหมายเลขไอพีบางกลุ่มให้เป็นหมายเลขไอพีส่วนบุคคล ดังแสดงในตารางที่ 2.2

ประเภท	ไอพีต้นทาง	ไอพีปลายทาง
A	10.0.0.0	10.255.255.255
B	172.16.0.0	172.31.255.255
C	192.168.0.0	192.168.255.255

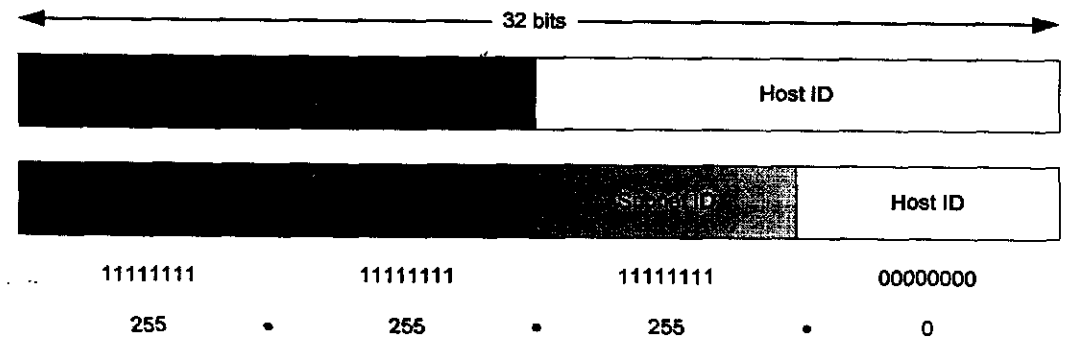
ตารางที่ 2.2 : หมายเลขไอพีเครือข่ายส่วนบุคคล

หมายเลขที่แสดงในตารางที่ 2.2 จะเป็นหมายเลขที่ไม่ใช้ในเครือข่ายอินเตอร์เน็ต กล่าวคือแพ็กเก็ตที่มีหมายเลขไอพีนี้จะไม่ถูกส่งต่อโดยเราท์เตอร์ของระบบอินเตอร์เน็ต ส่วนหมายเลขไอพีที่แจกจ่ายไปแล้ว สามารถดูรายละเอียดได้ที่ http://www.iana.org/assignments/ipv4-address-space

2.4.3 ซับเน็ตและซับเน็ตมาส์ค (Subnet and Subnet Mask)

เนื่องจากอินเตอร์เน็ตเติบโตเร็วมาก ทำให้มีปัญหาเกี่ยวกับการใช้หมายเลขไอพี เพราะในไม่ช้าหมายเลขไอพีที่มีทั้งหมด 32 บิต อาจถูกใช้หมดได้ นั่นคือต้องมีวิธีการใหม่ที่จะกำหนดหมายเลขไอพีให้กับโฮสต์ วิธีหนึ่งที่ใช้แก้ปัญหาก็คือ เพิ่มจำนวนบิตให้มีมากกว่า 32 บิต ซึ่งในขณะนี้กำลังมีการพัฒนาโปรโตคอลไอพีเวอร์ชันใหม่คือ IPv6 ซึ่งในเวอร์ชันนี้จะกำหนดให้หมายเลขไอพีมีขนาด 128 บิต และเมื่อมีการประกาศใช้แล้วจำนวนหมายเลขไอพีจะมีเพียงพออย่างแน่นอน แต่การที่จะอัพเกรดอินเตอร์เน็ตให้ใช้โปรโตคอลเวอร์ชันนี้ได้นั้นคงต้องใช้เวลาอีกนานพอสมควร เพราะเป็นการเปลี่ยนแปลงที่ใหญ่มาก

การแบ่งซับเน็ต จะเป็นอีกวิธีหนึ่งที่ช่วยให้การใช้หมายเลขไอพีมีความคุ้มค่ามากยิ่งขึ้น เนื่องจากหมายเลขไอพีบางประเภท เช่น คลาส B จะมีโฮสต์ได้ทั้งหมด 65,534 เครื่องซึ่งโดยส่วนใหญ่แล้วจะเป็นเครือข่ายที่ใหญ่เกินไปสำหรับองค์กรทั่วไป ดังนั้นจึงทำให้หมายเลขไอพีบางส่วนไม่ถูกใช้งาน การทำซับเน็ตเป็นการแก้ปัญหานี้ได้ โดยการแบ่งเป็นเครือข่ายย่อย เราท์เตอร์จะใช้เฉพาะส่วนที่เป็นหมายเลขเครือข่ายเท่านั้นในการจัดเส้นทางหรือเราท์ติ้ง สำหรับบริษัทใหญ่ๆที่ต้องการมีเครือข่ายย่อยจะมีวิธีที่ใช้แบ่งเครือข่ายใหญ่ให้เป็นเครือข่ายย่อยโดยวิธีที่เรียกว่า "ซับเน็ต (Subnet)" การแบ่งเป็นเครือข่ายย่อยๆ หรือซับเน็ตนั้นจะถูกจัดการโดยเราท์เตอร์หรือเกตเวย์ของเครือข่ายนั้นๆเท่านั้น โดยที่โฮสต์อื่นๆที่อยู่นอกเครือข่ายนี้จะมองเห็นเครือข่ายนี้เป็นเครือข่ายเดียวเท่านั้น



รูปที่ 2.7 : การทำซับเน็ต

การแบ่งเครือข่ายที่ใหญ่ให้เป็นเครือข่ายย่อย นั้นทำได้โดยการแบ่งเลขไอพีส่วนที่เป็นหมายเลขของโฮสต์มาเป็นหมายเลขของเครือข่าย ดังแสดงในรูปที่ 2.7 ส่วนซับเน็ตมาส์คคือตัวเลขที่บ่งชี้ว่าส่วนไหนของเลขไอพีเป็นหมายเลขเครือข่าย และส่วนไหนเป็นหมายเลขโฮสต์ ซับเน็ตมาส์คจะมีความยาวเท่ากับหมายเลขไอพีคือ 32 บิต ซึ่งจะเริ่มต้นด้วยเลขของเลข 1 เรียงกันและตามด้วยเลขของเลข 0 การคำนวณหมายเลขเครือข่ายย่อยจะทำได้โดยการ AND ระหว่างเลขซับเน็ตมาส์คและหมายเลขไอพีโดยดีฟอลต์ (Default) เน็ตมาส์คของหมายเลขไอพีแต่ละคลาสแสดงในตารางที่ 2.3

คลาส	เน็ตมาส์ค	ดีอตเดซิมอล
A	11111111 00000000 00000000 00000000	255.0.0.0
B	11111111 11111111 00000000 00000000	255.255.0.0
C	11111111 11111111 11111111 00000000	255.255.255.0

ตารางที่ 2.3 : ดีฟอลต์เน็ตมาส์คของไอพีแต่ละประเภท

2.5 การสื่อสารข้อมูลแบบอนุกรม

การสื่อสารข้อมูลแบบอนุกรมเป็นการรับหรือส่งข้อมูลในลักษณะกลุ่มของบิตครั้งละ 1 บิต เรียงลำดับเรื่อยไปจนสิ้นสุดแต่ในบางกรณีก็สามารถรับส่งข้อมูลครั้งละหลายๆบิตได้หากแต่จะต้องมีการตกลงกันระหว่างตัวส่งกับตัวรับว่า จะรับส่งข้อมูลคราวละกี่บิต ตัวรับจะต้องรอข้อมูลมาให้ครบทุกบิตเสียก่อนจึงทำการประมวลผล ส่งผลให้การสื่อสารข้อมูลแบบอนุกรมอาจจะมีความเร็วในการรับส่งข้อมูลต่ำกว่าแบบขนาน ในด้านจำนวนสายสัญญาณการรับส่งข้อมูลแบบอนุกรมจะใช้จำนวนสายที่น้อยกว่ามาก เนื่องจากการสื่อสารข้อมูลแบบขนานมีการโอนย้ายมาพร้อมกันจึงมีความจำเป็นต้องใช้จำนวนสายสัญญาณมากขึ้นตามจำนวนบิตของข้อมูลด้วยในขณะที่การสื่อสารแบบอนุกรมนั้นต้องการสายสัญญาณเพียงสองหรือสามเส้นเท่านั้นแต่อัตราในการรับส่งข้อมูลอาจต่ำกว่าแบบขนานทำให้ระยะทางในการสื่อสารข้อมูลแบบอนุกรมสามารถทำได้มากกว่าการสื่อสารแบบขนานดังนั้นการสื่อสารแบบขนานจึงไม่เหมาะในการสื่อสารกับอุปกรณ์ภายนอกเป็นระยะทางไกล ๆ เพราะจะทำให้สิ้นเปลืองค่าใช้จ่ายมาก

การสื่อสารข้อมูลแบบอนุกรมนั้นแบ่งออกได้เป็น 2 แบบคือการสื่อสารข้อมูลอนุกรมแบบซิงโครนัสและการสื่อสารข้อมูลอนุกรมแบบอะซิงโครนัสโดยการสื่อสารข้อมูลอนุกรมแบบซิงโครนัสนั้นจะมีสัญญาณนาฬิการวมอยู่กับการรับและส่งด้วย ตัวอย่างการส่งข้อมูลอนุกรมแบบซิงโครนัสได้แก่ คีย์บอร์ดของคอมพิวเตอร์ซึ่งสายเส้นหนึ่งจะเป็นของสัญญาณนาฬิกาส่วนสายอีกเส้นหนึ่งจะเป็นของข้อมูล ดังนั้นการติดต่อกันแบบซิงโครนัสนี้จะต้องใช้สายในการเชื่อมต่ออย่างน้อยที่สุด 3 เส้นคือสัญญาณนาฬิกา ข้อมูลและกราวด์ ส่วนการสื่อสารข้อมูลอนุกรมแบบอะซิงโครนัสนั้น สามารถรับและส่งข้อมูลไปในสายโดยไม่จำเป็นต้องมีสัญญาณนาฬิการวมอยู่ด้วยเหมือนกับการรับส่งข้อมูลแบบซิงโครนัสแต่จะใช้การกำหนดค่าสัญญาณนาฬิกาทั้งภาครับและภาคส่งให้มีค่าเท่ากัน ซึ่งเรียกสัญญาณนาฬิกาที่ใช้ในการกำหนดค่าให้ภาครับและภาคส่งนี้ว่า อัตราการถ่ายทอดข้อมูลหรือบอดเรต (Baudrate) มีหน่วยเป็น บิตต่อวินาที (bit per second : bps)

2.5.1 ความเร็วของการสื่อสารข้อมูลแบบอนุกรม

เนื่องจากการสื่อสารข้อมูลแบบอนุกรมเป็นการรับส่งข้อมูลในลักษณะกลุ่มของบิตข้อมูล (Bit Stream) ดังนั้น จึงต้องให้ความสนใจในการพิจารณาเรื่องอัตราเร็วในการรับส่งบิตเหล่านี้เป็นอันดับแรก โดยทั่วไปมักจะระบุกันในหน่วยของจำนวนบิตข้อมูลภายในเวลาหนึ่งวินาที เรียกว่า อัตราบอด ตามค่า

มาตรฐานเหล่านี้ ได้แก่ 110, 150, 300, 1200, 2400, 4800, 9600, 19200 บอด ข้อมูลทั้งแปดบิตนี้หากว่าถูกส่งออกมาด้วยอัตรา 9600 บอด จะใช้เวลาในการส่งข้อมูลหนึ่งบิตมีค่าเท่ากับ 1/9600 หรือ 104 us และเวลาในการส่งข้อมูลทั้งแปดบิตมีค่าเท่ากับ 8*104 หรือ 832 us

2.5.2 รูปแบบของการส่งข้อมูลแบบอนุกรม

การสื่อสารข้อมูลอนุกรมแบบอะซิงโครนัสจะใช้การแปลงข้อมูลขนานให้เป็นอนุกรมแล้วเพิ่มเติมบิตบางอย่างร่วมไปกับการส่งข้อมูลจริงซึ่งรูปแบบของข้อมูลที่ใช้ในการรับส่งข้อมูลอนุกรมแบบอะซิงโครนัสประกอบด้วย 4 ส่วนด้วยกันได้แก่

1. บิตเริ่มต้น (Start Bit) ซึ่งจะมีขนาด 1 บิต บิตเริ่มต้นมีหน้าที่สำหรับการบ่งบอกให้ทราบถึงตำแหน่งเริ่มต้นของบิตข้อมูล ตามปกติแล้วค่าของบิตเริ่มต้นจะเป็นระดับลอจิกต่ำ
2. บิตข้อมูลแบบอนุกรมจะมีขนาด 5, 6, 7 หรือ 8 บิต
3. บิตแสดงภาวะความเป็นเลขคู่หรือเลขคี่ของข้อมูลบิตที่เป็น 1 (Parity Bit) จะมีขนาด 1 บิต หรือไม่มี บิตนี้มีหน้าที่เพื่อการตรวจสอบความถูกต้องของข้อมูลโดยทั่วไปมักเรียกว่าบิตพาริตี้ และจะนำไปต่อท้ายบิตข้อมูล ค่าของบิตนี้ขึ้นอยู่กับจำนวนค่าของบิตที่เป็น 1 ซึ่งจะเป็นได้สองลักษณะคือ พาริตี้คู่ (Even Parity) หรือพาริตี้คี่ (Odd Parity) ตัวอย่างเช่น ระบบที่ติดต่อกันโดยระบุว่าจะใช้พาริตี้คู่ ทางด้านส่งจะนำค่าข้อมูลที่จะส่งมาพิจารณาหาจำนวนของบิตเป็น 1 หากเป็นเลขจำนวนคู่อยู่แล้ว ค่าของพาริตี้จะมีค่าเป็นศูนย์ แต่หากว่าจำนวนของบิตเป็น 1 เป็นเลขจำนวนคี่ ค่าของพาริตี้ก็จะมีค่าเป็น 1 ส่วนทางด้านรับก็จะทำการตรวจสอบจำนวนบิตที่เป็น 1 ของข้อมูลที่ได้รับมาทั้งหมดรวมทั้งบิตพาริตี้ ถ้ามีค่าเป็นเลขจำนวนคู่ แสดงว่าข้อมูลที่ได้รับเข้ามานี้ถูกต้องแต่หากไม่เป็นเลขจำนวนคู่แสดงว่าเกิดการผิดพลาดของข้อมูลขึ้น เป็นต้น
4. บิตสุดท้าย (Stop Bit) จะมีขนาด 1, 1.5 หรือ 2 บิต บิตสุดท้ายเป็นบิตที่เพิ่มเข้ามาเพื่อระบุถึงขอบเขตการสิ้นสุดของกลุ่มบิตข้อมูล บิตสุดท้ายสามารถโปรแกรมได้คือ 1, 1.5 หรือ 2 บิต ดังนั้นกรณีของการส่งข้อมูล 8 บิต หากข้อมูลถูกส่งออกไปด้วยอัตราเร็ว 9600 บอด เวลาโดยรวมในการส่งข้อมูลหนึ่งไบต์ จะมีค่าเป็น 12*104 หรือ 1.25 ms

2.5.3 มาตรฐานพอร์ตอนุกรมแบบ RS-232

มาตรฐานการเชื่อมต่อแบบอนุกรม RS-232 เป็นมาตรฐานอุตสาหกรรมที่ออกแบบมาเพื่อใช้กับการรับส่งข้อมูลอนุกรมแบบอะซิงโครนัส 2 ทิศทาง โดยมาตรฐาน ในอดีตนั้นถูกออกแบบมาเพื่อการส่งผ่านข้อมูลจากคอมพิวเตอร์ไปยังโมเด็มเพียงอย่างเดียวเท่านั้นเพื่อที่จะนำข้อมูลจากโมเด็มนี้สื่อสารผ่านสายโทรศัพท์ไปยังคอมพิวเตอร์อีกชุดหนึ่งที่อยู่ห่างไกลกัน โดยคณะกรรมวางมาตรฐานที่มีชื่อว่า EIA RS-232 มาตรฐานนี้ในช่วงแรกจะใช้คอนเน็กเตอร์ (Connector) เป็นแบบ DB-25 โดยกำหนดความยาวสูงสุดของสายสัญญาณไว้ที่ 50 ฟุต มีระดับสัญญาณตั้งแต่ -3 V ถึง -12 V แสดงว่ามีข้อมูล (MASK) และ +3 V ถึง +12 V แสดงเป็นช่องว่าง (Space)

มาตรฐาน RS-232 ได้กำหนดรูปแบบของอุปกรณ์ที่เชื่อมต่อข้อมูล (Data Terminal Equipment : DTE) กับวงจรข้อมูลปลายทาง (Data Circuit Terminating : DCE) ไว้ว่า อุปกรณ์ DTE จะต้องเป็นอุปกรณ์ที่มีการประมวลผลในตัวเช่น ไมโครคอนโทรลเลอร์หรือ ไมโครคอมพิวเตอร์ ซึ่งมีความสามารถในการสร้างข้อมูลแบบอนุกรมได้ ส่วนอุปกรณ์ DCE จะทำหน้าที่เป็นเพียงตัวรับข้อมูลที่ส่งมาจาก DTE เท่านั้น โดยการรับส่งข้อมูลระหว่างอุปกรณ์ทั้งสองจะกระทำผ่านมาตรฐาน RS-232

ข้อแตกต่างระหว่างอุปกรณ์ DTE และ DCE อย่างหนึ่งที่เห็นได้ชัดคือ คอนเน็กเตอร์ของ DTE เป็นตัวผู้ ส่วนคอนเน็กเตอร์ DCE เป็นตัวเมีย ซึ่งพอร์ตอนุกรมของคอมพิวเตอร์ที่ใช้กันอยู่ทั่วไปจะเป็นคอนเน็กเตอร์ตัวผู้ ส่วนคอนเน็กเตอร์ที่อยู่ในโมเด็มจะเป็นตัวเมีย

- คอนเน็กเตอร์สำหรับพอร์ต RS-232 และการเชื่อมต่อ

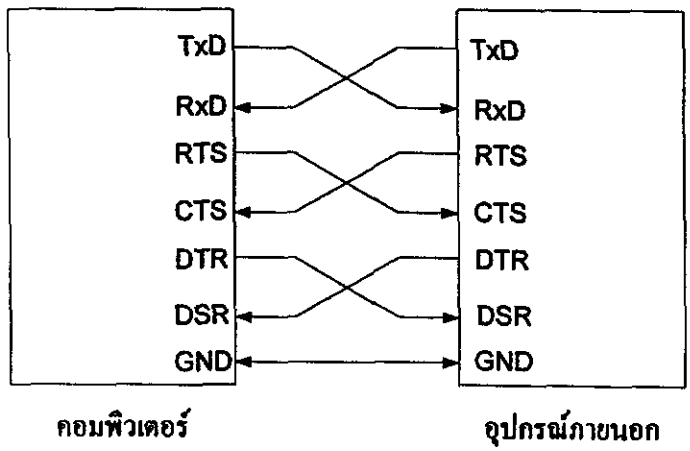
มาตรฐานการเชื่อมต่อแบบอนุกรม RS-232 จะใช้คอนเน็กเตอร์แบบ DB-25 ตัวผู้หรือแบบ DB-9 ตัวผู้ ซึ่งคอนเน็กเตอร์แบบ DB-25 จะมีขาต่อใช้งานเพียง 9 เส้นเช่นเดียวกับคอนเน็กเตอร์แบบ DB-9 เนื่องจากขาอื่นที่เคยใช้งานในอดีต ในปัจจุบันมีการใช้งานไม่มากนัก จึงถูกยกเลิกไป ดังตารางที่ 2.4

คอนเน็กเตอร์ DB-9	คอนเน็กเตอร์ DB-25	ชื่อของสายสัญญาณ	ชนิดของสายสัญญาณ
1	8	Data Carrier Detect :DCD	อินพุต
2	3	Receive Data :RD	อินพุต
3	2	Transmitted Data : TD	เอาท์พุต
4	20	Data terminal Ready: DTR	เอาท์พุต
5	7	Signal Ground : GND	-
6	6	Data Set Ready :DSR	อินพุต
7	4	Request to send :RTS	เอาท์พุต
8	5	Clear to send :CTS	อินพุต
9	22	Ring Indicator : RI	อินพุต

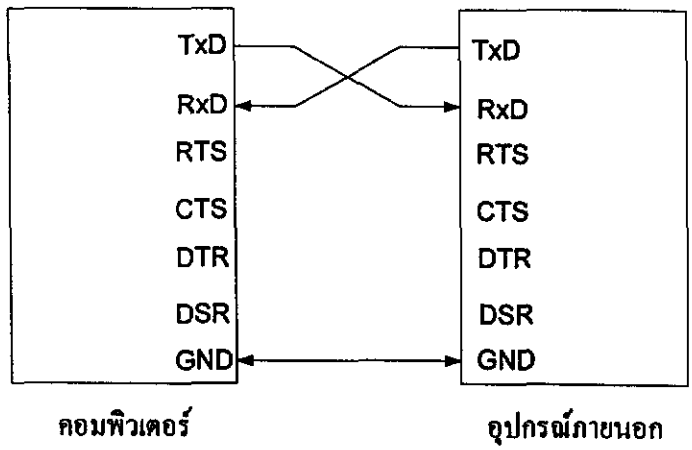
ตารางที่ 2.4 : การจัดขาของคอนเน็กเตอร์พอร์ตอนุกรมตามมาตรฐาน RS-232 ทั้งแบบ DB-25 และ DB-9

สำหรับการเชื่อมต่อคอมพิวเตอร์กับอุปกรณ์ภายนอกนั้นแสดงดังในรูป 2.8 ลูกศรในรูปแสดงถึงทิศทางของข้อมูล ในรูปเป็นการเชื่อมต่อแบบ Null modem หรือการเชื่อมต่อโดยตรงโดยไม่ต้องผ่านโมเด็ม โดยมีการตรวจสอบหรือแฮนด์เช็กเต็มรูปแบบ ส่วนในรูปที่ 2.9 เป็นการเชื่อมต่อในลักษณะที่ใช้สายสัญญาณเพียง 3 เส้น โดยเส้นหนึ่งสำหรับส่งข้อมูล อีกเส้นหนึ่งสำหรับรับข้อมูลและเส้นสุดท้ายเป็นกราวด์ สำหรับหน้าที่ในการทำงานแต่ละขาของพอร์ตอนุกรม RS-232 มีดังนี้

- Data Carrier Detect : DCD หรืออาจเรียกว่า "Carrier Detect : CD" ขานี้จะแอกทีฟเมื่อมีการส่งสัญญาณพาหะจากอุปกรณ์สื่อสารข้อมูลเช่น โมเด็ม สำหรับการใช้งานปรกติ ขานี้จะไม่ถูกใช้งานมากนัก
- Receive Data : RD ขานี้ใช้เพื่อรับข้อมูลอนุกรมที่เข้ามายังคอมพิวเตอร์ โดยนำข้อมูลที่อ่านได้เก็บไว้ในรีจิสเตอร์บัฟเฟอร์
- Transmitted Data หรือ TD ใช้ส่งข้อมูลออกจากคอมพิวเตอร์ โดยนำข้อมูลที่เก็บอยู่ในบัฟเฟอร์สำหรับส่งข้อมูล (Transmitted Buffer) ส่งออกไป
- Data terminal Ready : DTR เป็นขาที่ส่งสัญญาณออกจากคอมพิวเตอร์ให้อุปกรณ์ปลายทางรับรู้ว่าต้องการติดต่อด้วยโดยขา DTR นี้ต้องเชื่อมกับ DSR ของอุปกรณ์ปลายทางและขา DTR ของอุปกรณ์ปลายทางต้องเชื่อมกับขา DSR ของคอมพิวเตอร์ถ้าใช้การเชื่อมต่อเป็นแบบ Null Modem ซึ่งใช้สายในการเชื่อมต่อเพียงสามเส้นจะต้องต่อขา DTR และ DSR ของตัวมันเองเข้าด้วยกัน
- Signal Ground : GND กราวด์ของระบบ
- Data Set Ready : DSR ขานี้จะใช้คู่กับขา DTR เพื่อตรวจสอบการเชื่อมต่อกันระหว่างคอมพิวเตอร์กับอุปกรณ์ปลายทางซึ่งขา DSR นี้จะเป็นขาสำหรับรับข้อมูลจากภายนอกซึ่งถูกส่งมาจากขา DTR
- Request to send : RTS เป็นขาสำหรับส่งสัญญาณร้องขอให้ทางอุปกรณ์ปลายทางส่งข้อมูลกลับมายังคอมพิวเตอร์ โดยขาที่รับสัญญาณ RTS ก็คือขา CTS ในกรณีที่ใช้การเชื่อมต่อแบบ Null Modem จะต้องเชื่อมต่อขา RTS และ CTS ของตัวมันเองเข้าด้วยกัน เพื่อที่จะให้การรับและส่งข้อมูลเกิดขึ้นได้ตลอดเวลา
- Clear to send : CTS ขานี้จะคอยรับสัญญาณจากขา RTS เมื่อรับสัญญาณได้ ข้อมูลที่ขา TD จะถูกส่งออกไป ดังนั้นขานี้จึงถูกใช้เพื่อตรวจสอบการเชื่อมต่อว่าพร้อมที่จะรับข้อมูลหรือไม่
- Ring Indicator : RI ใช้ในการแสดงสถานะสัญญาณเรียกจากสัญญาณโทรศัพท์ ปกติในการสื่อสารโดยทั่วไปสายนี้จะไม่ถูกใช้งาน จะใช้งานก็ต่อเมื่อมีการเชื่อมต่อกับโมเด็มและโปรแกรมมีการตรวจสอบสัญญาณนี้เท่านั้น



รูปที่ 2.8 : การต่ออุปกรณ์ภายนอกเข้ากับคอมพิวเตอร์แบบ Null Modem



รูปที่ 2.9 : การต่ออุปกรณ์ภายนอกเข้ากับคอมพิวเตอร์แบบ RS-232 โดยใช้สัญญาณเพียง3 เส้น

- UART (Universal Asynchronous Receiver Transmitter)

UART ย่อมาจากคำว่า "Universal Asynchronous Receiver Transmitter" หมายถึงอุปกรณ์ที่ทำหน้าที่รับและส่งข้อมูลอนุกรมแบบอะซิงโครนัสนั้นเอง ถือว่าเป็นหัวใจสำคัญของการสื่อสารแบบอนุกรม

หน้าที่หลักของ UART คือทำหน้าที่แปลงข้อมูลที่อยู่ในรูปแบบขนานจากคอมพิวเตอร์ให้อยู่ในรูปแบบอนุกรมซิงโครนัสแล้วส่งออกไป และทำหน้าที่แปลงข้อมูลอนุกรมแบบอะซิงโครนัสที่ป้อนเข้ามายัง UART ให้เป็นแบบขนานก่อนที่จะส่งเข้าสู่คอมพิวเตอร์ ซึ่งนอกจาก UART จะส่งข้อมูลไปยังคอมพิวเตอร์แล้ว ยังแจ้งข้อมูลอื่นๆ ให้คอมพิวเตอร์รับทราบด้วยเช่น อัตราเร็วในการรับส่งข้อมูล, รูปแบบการส่งข้อมูล, ความผิดพลาดที่เกิดขึ้นระหว่างการถ่ายทอดข้อมูล เป็นต้น ภายในจะมีส่วนของวงจรสร้างบอดเรตแบบโปรแกรมได้ (programmable buadrate generator) โดยการกำหนดค่าตัวหารให้กับสัญญาณนาฬิกาของ UART โดยตัวหารนี้มีขนาด 16 บิต ดังนั้นจึงสามารถกำหนดตัวหารอยู่ในช่วง 1 ถึง 65,535 สามารถรับส่งข้อมูลได้ทั้งแบบฮาล์ฟดูเพล็กซ์ (Half duplex) และฟูลดูเพล็กซ์ โดยการส่งแบบฮาล์ฟดูเพล็กซ์เป็นการส่งแบบทิศทางเดียวส่วนการส่งแบบฟูลดูเพล็กซ์นั้นสามารถรับและส่งข้อมูลได้ในคราวเดียวกัน

- ระดับแรงดันที่ใช้งานสำหรับพอร์ตอนุกรม RS-232

มาตรฐานการสื่อสารข้อมูลของพอร์ตอนุกรม ได้ระบุช่วงระดับแรงดันสำหรับการทำงานของพอร์ตอนุกรมไว้ว่า ที่ลอจิก " 0 " จะมีระดับสัญญาณ +3V ถึง +15V ส่วนลอจิก " 1 " จะมีระดับสัญญาณ -3V ถึง -15V ด้วยเหตุนี้จึงทำให้ไม่สามารถที่จะนำขาเอาท์พุตใดๆต่อเข้ากับลอจิกเกตเพื่อใช้งานได้โดยตรง จะต้องผ่านวงจรเพื่อที่จะเปลี่ยนระดับแรงดันเสียก่อน โดยปกติจะใช้ไอซีพวก RS-232 transceiver ที่นิยมมากคือ MAX 232 หรือ ICL232 ไอซีกลุ่มนี้จะทำหน้าที่แปลงระดับแรงดันของ RS-232 ให้อยู่ในระดับทีทีแอล (TTL) โดยลอจิก " 0 " ซึ่งเดิมมีระดับสัญญาณ +3 V ถึง +15 V จะถูกแปลงเป็น

0 V ส่วนลอจิก " 1 " ซึ่งเดิมมีระดับสัญญาณ -3 V ถึง -15 V จะถูกแปลงเป็น 5 V ทั้งนี้เพื่อให้สามารถเชื่อมต่อกับอุปกรณ์ดิจิตอลอื่นๆที่ใช้ระดับแรงดันทีทีแอลได้

2.5.4 โค้ด (code)

รหัสหรือโค้ดคืออักษรหรือตัวเลขที่ใช้แทนการทำงานอย่างใดอย่างหนึ่งเช่น 76H หรือ 01110110b ถ้าเป็นรหัสคำสั่งของ Z-80 ก็จะมีความหมายว่า HALT มาตรฐานของรหัสมีอยู่หลายแบบเช่น Hex code, Binary Code, EBCDIC, ASCII ในที่นี้จะขอกล่าวถึงเฉพาะรหัสแอสกี้ (ASCII : American Standard Code for Information Interchange) เป็นรหัสที่พัฒนาขึ้นโดยสถาบันมาตรฐานแห่งชาติสหรัฐอเมริกา (American National Standard Institute : ANSI) สามารถใช้แทนข้อมูลอักขระและคำสั่งได้มากมายและเป็นโค้ดแบบ 7 บิต เช่นอักษร A, B, C แทนด้วยค่าดังนี้ 41H, 42H, 43H และมีการขยายเป็นรหัสแบบ 8 บิต โดยเพิ่มบิตสูงสุด (หรือบิตซ้ายสุด) ที่เติมเข้ามาอาจจะเพียงแต่เติม 0 เข้าไป หรือไม่ก็นำมาใช้เป็นพาริตี้บิต การอ่านตารางให้นำตัวเลขแนวนอน และแนวตั้งมาเรียงต่อกัน เช่น ตัวอักขระ Aก็คือ 41H ตัวเลข 1 คือ 31H เป็นต้น สมมติว่าต้องการเก็บคำว่า BINGO ก็จะได้เป็น 42H 49H 4EH 47H 4FH ตามลำดับ

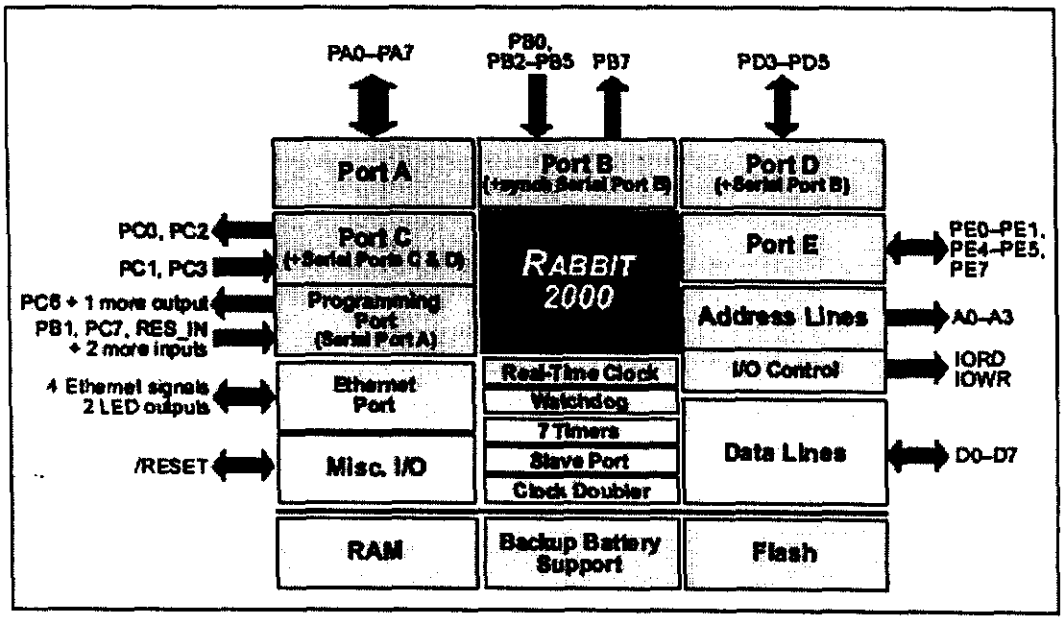
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NUL	DLE	SP	0	@	P		p								
1	SOH	DC1	!	1	A	Q	a	q								
2	STX	DC2	"	2	B	R	b	r								
3	ETX	DC3	#	3	C	S	c	s								
4	EOT	DC4	$	4	D	T	d	t								
5	ENQ	NAK	%	5	E	U	e	u								
6	ACK	SYN	&	6	F	V	f	v								
7	BEL	ETB	'	7	G	W	g	w								
8	BS	CAN	(	8	H	X	h	x								
9	HT	EM	)	9	I	Y	i	y								
A	LF	SUB	*	:	J	Z	j	z								
B	VT	ESC	+	;	K	[	k	{								
C	FF	FS	,	<	L	\	l	|							
D	CR	GS	-	=	M	]	m	}								
E	SO	RS	.	>	N	^	n	~								
F	SI	US	/	?	O	-	o	M								

รูปที่ 2.10 : ตารางรหัสแอสกี้แทนตัวอักษร

2.6 ไมโครโปรเซสเซอร์ Rabbit 2000

ไมโครโปรเซสเซอร์แร็บบิต (Rabbit) 2000 เป็นไมโครโปรเซสเซอร์ 8 บิต มีโครงสร้างเดียวกับไมโครโปรเซสเซอร์ Z80 และ Z180 แต่สามารถรองรับชุดคำสั่งที่เหมาะสมกับการใช้งานได้มากกว่าและตัดชุดคำสั่งที่ไม่จำเป็นออกไป มีชุดคำสั่งภาษาไดนามิคซีในการพัฒนาโปรแกรมโดยเฉพาะ และสามารถใช้ร่วมกับชุดคำสั่งภาษาแอสแซมบลี (Assembly) ได้ และที่สำคัญคือใช้เวลาในการประมวลผลชุดคำสั่งน้อยกว่าไมโครโปรเซสเซอร์ 8 บิต อื่นๆ อีกทั้งยังมีความสามารถด้านการติดต่อกับระบบเครือข่าย มีพอร์ตอินพุตและเอาท์พุตมากมาย

แร็บบิต 2000 จัดเป็นไมโครโปรเซสเซอร์ที่มีความกะทัดรัดและมีคุณสมบัติหลากหลาย อีกทั้งยังมีส่วนประกอบที่จำเป็น เช่น มี 40 I/O, 4 CMOS compatible serial ports มี Timer 8-bit 5 ตัว และ Timer 10-bit 1 ตัว กับชุดรีจิสเตอร์ 2 ชุด และ fast number-crunching clock สามารถเชื่อมต่อกับหน่วยความจำชนิด Flash และ SRAM ภายนอกตัวซีพียูได้ โดยเราสามารถที่จะทำการดาวน์โหลดและดีบัก(Debug) โปรแกรมให้กับซีพียูได้โดยใช้ซอฟท์แวร์ไดนามิคซี

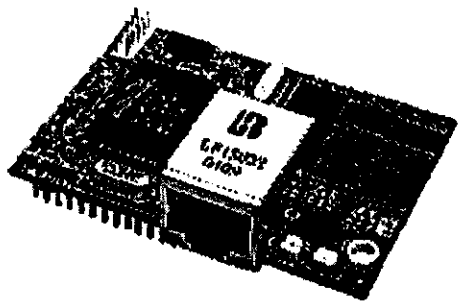


รูปที่ 2.11 : ส่วนต่างๆภายในไมโครโปรเซสเซอร์แร็บบิต 2000

2.6.1 โมดูลแร็บบิต RCM 2200

ลักษณะโครงสร้างของโมดูลแร็บบิต RCM 2200 เป็นชิ้นส่วนซึ่งประกอบด้วยไมโครโปรเซสเซอร์แร็บบิต 2000 ที่ถูกออกแบบให้มีความสามารถที่หลากหลาย มีพอร์ตอีเธอร์เน็ตสำหรับการเชื่อมต่อเครือข่ายท้องถิ่นและอินเตอร์เน็ต ระบบวงจรที่ใช้ในโมดูลแร็บบิต RCM 2200 ถูกออกแบบให้ง่ายต่อการใช้งาน หน่วยความจำที่ใช้เป็นแบบ Static RAM และ Flash Memory สัญญาณนาฬิกาที่ใช้งานในวงจรมี 2 แหล่งคือ Oscillator และ Time Keeping ใช้แหล่งจ่ายไฟ +5V ในการทำงาน

นอกจากนี้ยังสามารถติดต่อกับอุปกรณ์ภายนอก โดยผ่านทางพอร์ตที่จัดไว้บนคอนเน็กเตอร์ J4 และคอนเน็กเตอร์ J5

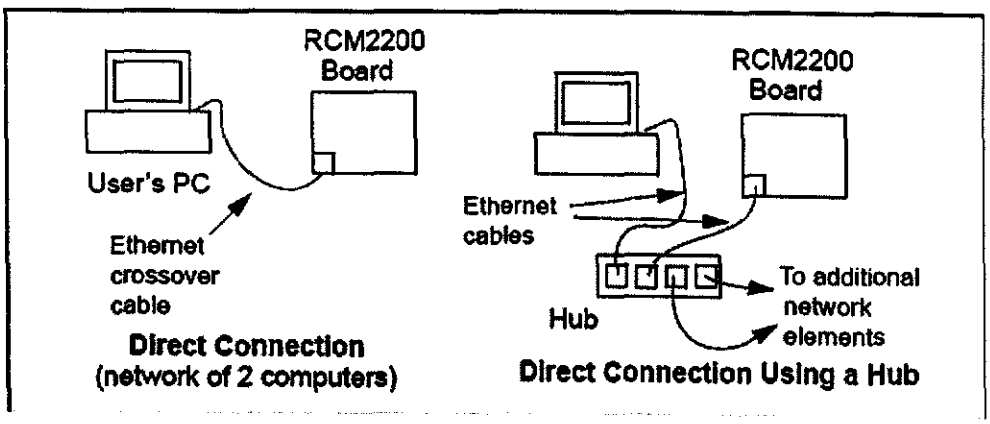


รูปที่ 2.12 : โมดูลแร็บบิต RCM 2200

ลักษณะที่สำคัญและความสามารถของโมดูลแร็บบิตรุ่น RCM2200

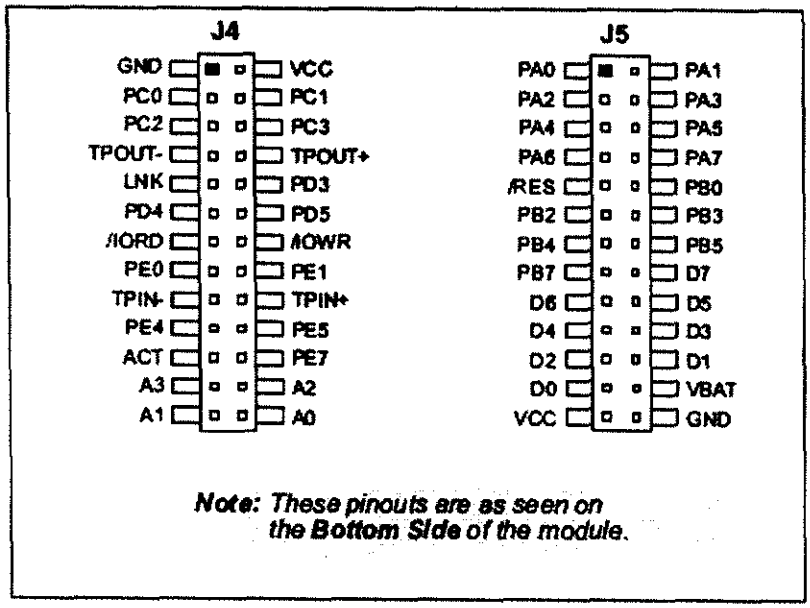
- ขนาดเล็ก 1.60'' x 2.30''x0.86'' หรือ (41mm x 58mm x 22mm)
- ไมโครโปรเซสเซอร์แร็บบิต 2000 ทำงานที่ความถี่ 22.1 MHz
- มีขาที่เป็นอินพุตหรือเอาท์พุต ทั้งหมด 26 ขา เป็นแบบขนาน มีโหมดการทำงานหลากหลาย
- มี Data Bus 8 เส้น
- มี Address Bus 4 เส้น
- มีหน่วยความจำแบบ Flash ขนาด 256 KB และแบบ Static RAM ขนาด 128 KB
- สัญญาณนาฬิกาให้จังหวะการทำงานที่ 29.5 MHz
- ใช้หัวต่อแบบ RJ-45 เป็น Ethernet Port

พอร์ตอนุกรมเป็นชนิด CMOS ซึ่งใช้อัตราบอดแบบอะซิงโครนัสได้ความถี่สูงสุดถึง 345,600 bps และบอดเรตที่ใช้ส่งแบบซิงโครนัสได้ความถี่สูงสุดถึง 138,240 bps



รูปที่ 2.13 : การเชื่อมต่อโมดูลแร็บบิตเข้ากับเครือข่ายคอมพิวเตอร์

RCM 2200 มีพอร์ตขนานสำหรับใช้งานทั้งหมด 5 พอร์ต พอร์ตละ 8 บิต โดยจะอยู่ในขาเชื่อมต่อของคอนเน็กเตอร์ J4 และคอนเน็กเตอร์ J5 โดยพอร์ตอินพุตและพอร์ตเอาท์พุตแบบสองทิศทางจะอยู่ที่ตำแหน่ง PA0–PA7, PD3–PD5, PE0–PE1, PE4, PE5 และ PE7



รูปที่ 2.14: RCM 2200 I/O Pin outs

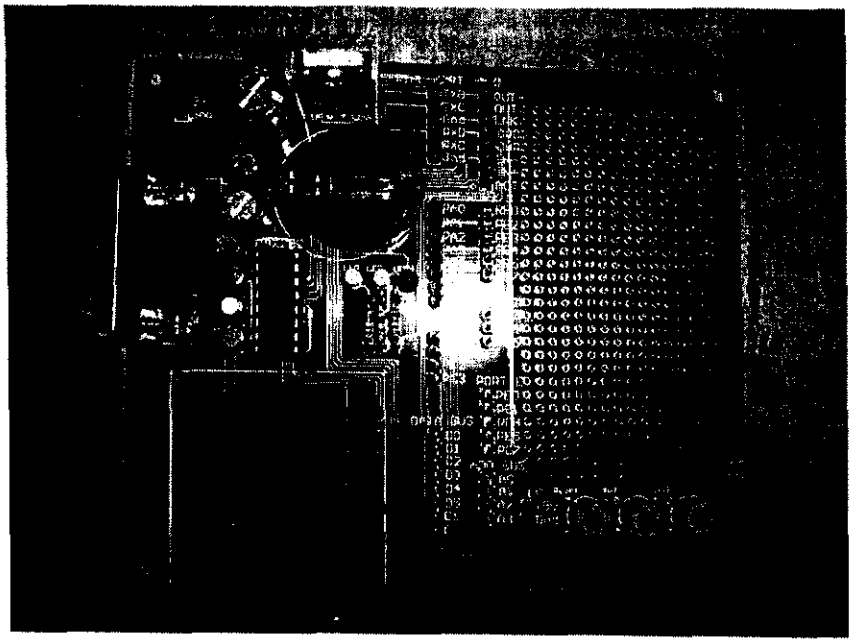
RCM 2200 ถูกออกแบบมาเพื่อให้เชื่อมต่อกับวงจรภายนอก เพื่อให้มีประสิทธิภาพในการใช้งานได้หลากหลายขึ้นอยู่กับการประยุกต์ใช้งาน โดยมีจุดเชื่อมต่อกับอุปกรณ์ภายนอกคือหัวเชื่อมต่อขนาด 26 ขา จำนวน 2 หัว ซึ่งก็คือคอนเน็กเตอร์ J4 และคอนเน็กเตอร์ J5 นั่นเอง

2.6.2 Prototyping Board

เป็นแผงวงจรที่ใช้เชื่อมต่อกับโมดูลแร็บบิตทางคอนเน็กเตอร์ J4 และคอนเน็กเตอร์ J5 ซึ่งจะช่วยเพิ่มความสะดวกในการใช้งานโมดูลแร็บบิต โดยมีการออกแบบให้สามารถทำงานในหน้าที่ต่างๆ ดังนี้

- เป็นแหล่งจ่ายไฟให้กับโมดูลแร็บบิต
- จัดขาเชื่อมต่อของพอร์ตต่างๆ ให้สามารถใช้งานได้สะดวก
- ทำหน้าที่เชื่อมต่อกับสายสัญญาณอื่นๆที่จะนำมาใช้กับโมดูลแร็บบิต
- ทำหน้าที่เชื่อมต่อกับหน่วยแสดงผล ด้วยจอ LCD ขนาด 16 x 2
- ทำหน้าที่เชื่อมต่อกับหน่วยรับค่า ด้วย Key Pad ขนาด 4 x 4

เพื่อศึกษาและทดลองการทำงานของโมดูลแร็บบิต จำเป็นต้องสร้าง Prototyping Board เพื่อใช้งานร่วมกับโมดูลแร็บบิตเพราะจะช่วยเพิ่มความสะดวกในการทำงานตามการประยุกต์ใช้งานที่ต้องการ



รูปที่ 2.15 : Prototyping Board

โดยหลักๆแล้ว Prototyping Board นั้นถูกสร้างขึ้นมาเพื่อให้สามารถใช้งานขาเชื่อมต่อของพอร์ตต่างๆได้สะดวกยิ่งขึ้น มีชุดจ่ายแรงดันไฟตรง +5V เพื่อเป็นแหล่งจ่ายไฟให้กับโมดูลแร็บบิตและมีคอนเน็กเตอร์ DB-9 เป็นจุดเชื่อมต่อกับพอร์ตอนุกรมซี (Serial Port C) และพอร์ตอนุกรมดี (Serial Port D) โดยผ่านวงจรแปลงระดับสัญญาณ RS-232

2.6.3 สายโปรแกรม (Programming Cable)

สายโปรแกรมใช้สำหรับเชื่อมต่อโมดูลแร็บบิตเข้ากับคอมพิวเตอร์เพื่อดาวน์โหลดชุดคำสั่งที่ใช้ควบคุมการทำงานของโมดูลแร็บบิตเข้าไปเก็บไว้ในหน่วยความจำของโมดูลแร็บบิต โดยการติดต่อผ่านพอร์ตอนุกรมเอ (Serial Port A) ของโมดูลแร็บบิต

2.6.4 ส่วนของซอฟท์แวร์ที่ใช้ในการพัฒนาโมดูลแร็บบิต RCM 2200

ซอฟท์แวร์ที่นำมาใช้ในการพัฒนาชุดคำสั่งสำหรับควบคุมการทำงานของโมดูลแร็บบิต อีกทั้งยังใช้สำหรับการโปรแกรมและดีบักก็คือซอฟท์แวร์ไดนามิคซีซึ่งมีคุณสมบัติต่างๆดังนี้

- ภาษาที่ใช้ในการพัฒนาคือภาษาไดนามิคซีซึ่งมีโครงสร้างภาษาเหมือนภาษาซี
- มีไลบรารี่ (Library) ที่สนับสนุนการเขียนโปรแกรมเพื่อเชื่อมต่อกับเครือข่าย
- สามารถโปรแกรมลงบนโมดูลแร็บบิต RCM 2200 โดยผ่านทางพอร์ตอนุกรม

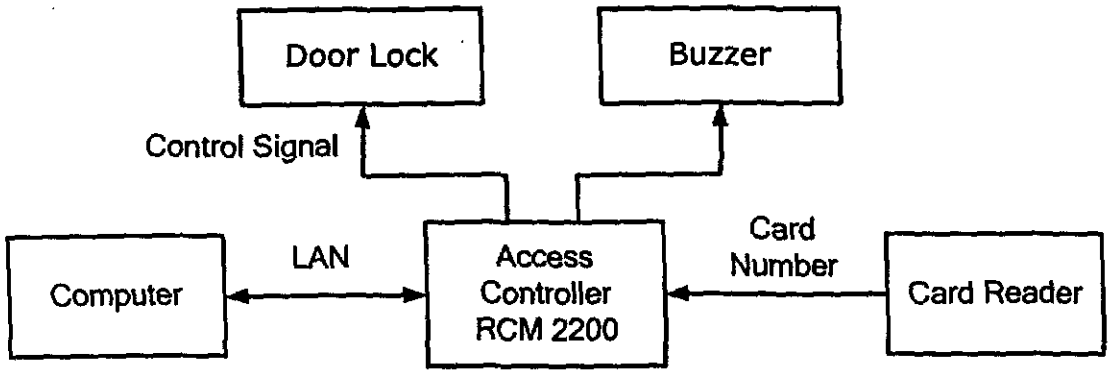
ซอฟท์แวร์ไดนามิคซี (Dynamic C)

ซอฟท์แวร์ไดนามิคซีนั้นมีการใช้งานในเครือข่ายตั้งแต่ปี 1989 ถูกออกแบบมาสำหรับการเขียนโปรแกรมที่เป็นระบบย่อยๆประกอบรวมกัน และซอฟแวร์ไดนามิคซียังมีความสามารถในการตรวจสอบตัวเองได้อย่างรวดเร็วโปรแกรมพื้นฐานของระบบมีข้อมูลประมาณ 1000 Bytes ที่ใช้ในการจัดเตรียมการดีบักและการติดต่อข้อมูลต่างๆ

ไดนามิคซีต้องการใช้ไบออส (BIOS) เพื่อใช้ในการตรวจสอบโปรแกรม เพื่อให้สะดวกต่อการใช้งาน ถ้าผู้ใช้หยุดการรันโปรแกรม และเปลี่ยนมาใช้โปรแกรมใหม่ไบออส ก็จะตรวจสอบการทำงานใหม่ตลอด ไดนามิคซีถูกออกแบบให้สามารถใช้งานร่วมกับภาษาแอสแซมบลีหรือใช้โปรแกรมภาษาซี การอินเตอร์รัพท์ (Interrupt) อาจเกิดขึ้นได้กับการเขียนโปรแกรมภาษาไดนามิคซีหรือภาษาแอสแซมบลี ซึ่งซอฟท์แวร์ไดนามิคซีเป็นซอฟท์แวร์ที่มีประสิทธิภาพในการเขียนโปรแกรมเพื่อควบคุมการทำงานของไมโครโปรเซสเซอร์แร็บบิต 2000 โดยซอฟท์แวร์มีโครงสร้างการใช้งานคล้ายคลึงกับภาษาซี ลักษณะของตัวซอฟท์แวร์ไดนามิคซีจะสนับสนุนการใช้งานชุดโปรโตคอลทีซีพีไอพีโดยจะมีไลบรารี่หลักที่สนับสนุนคือ DCRTCP.LIB, DNS.LIB, TCP.LIB, UDP.LIB, NET.LIB ส่วนในการติดต่อเน็ตเวิร์คเลเยอร์ของโปรโตคอลทีซีพีไอพีจะมีไลบรารี่ ARP.LIB กับ ICMP.LIB ทำให้ผู้ใช้ซอฟท์แวร์ไดนามิคซีมีทางเลือกในการที่จะพัฒนาโปรแกรมและบันทึกลงใน Flash Memory ขนาด 256 KB หรือใน Static RAM ขนาด 128 KB

ข้อเสียของการใช้ Flash Memory คือเมื่อมีการดีบักโปรแกรมเพื่อขัดจังหวะการทำงาน จะทำให้อินเตอร์รัพท์เกิดข้อผิดพลาด การทำงานของโปรแกรมก็จะหยุดทำงานด้วย

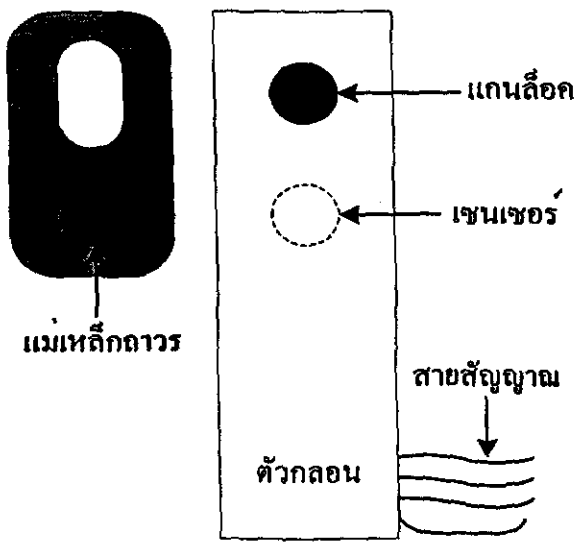
บทที่ 3
การคำนวณและการสร้าง



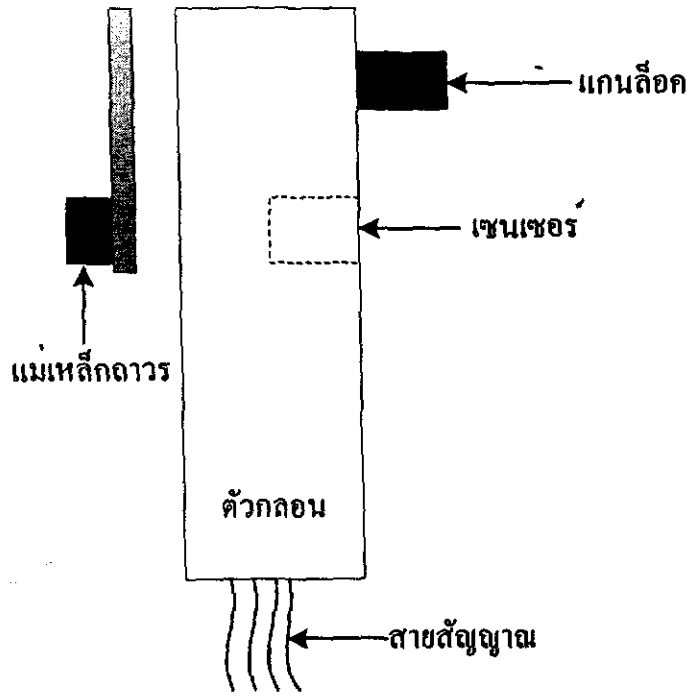
รูปที่ 3.1 : Block Diagram แสดงระบบรวม

3.1 Drop Bolt

คือกลอนประตูไฟฟ้าชนิดที่ใช้แกนล็อค ส่วนประกอบของกลอนชนิดนี้แบ่งออกเป็น 3 ส่วนคือ ส่วนที่เป็นแกนล็อค, ส่วนตัวกลอน และส่วนแม่เหล็กถาวรที่ใช้เป็นเซนเซอร์ ลักษณะของดรอปโบลท์ (Drop Bolt) แสดงในรูปที่ 3.2 ในตัวกลอนจะมีสายสัญญาณอย่างน้อย 4 เส้น โดยสาย 2 เส้นจะใช้สำหรับรับสัญญาณควบคุม (Control Signal) เพื่อทำการล็อคหรือปลดล็อค ส่วนสายที่เหลืออีก 2 เส้นจะเป็นสายที่ส่งสัญญาณแสดงสถานะของประตู (Door Status) ออกมา เพื่อบอกให้รู้ว่าขณะนี้ประตูเปิดหรือปิดอยู่

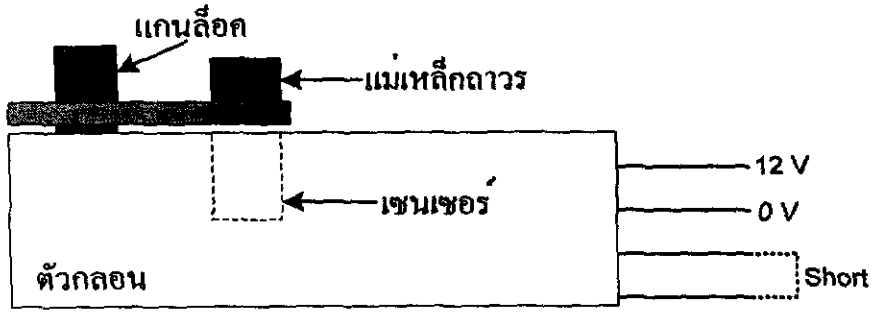


รูปที่ 3.2 : ดรอปโบลท์เมื่อมองจากด้านหน้า

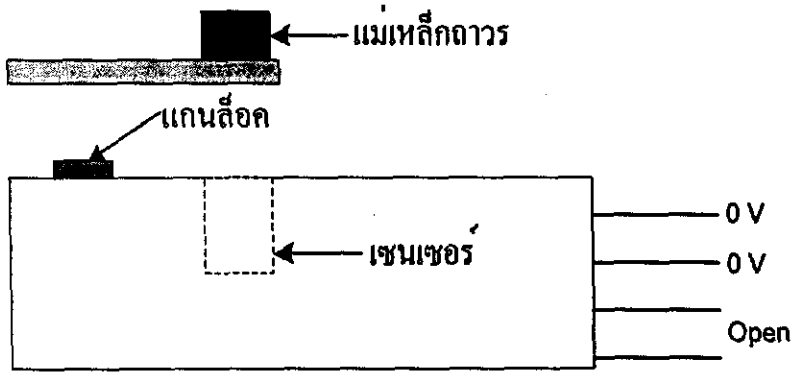


รูปที่ 3.3 : ครอปโบลท์เมื่อมองจากด้านข้าง

สัญญาณควบคุมจะใช้ VDC +12 V เมื่อป้อนสัญญาณนี้ให้ครอปโบลท์ส่วนแกนล็อคก็จะเด้งออกมาเพื่อทำการล็อค แต่ถ้าป้อนสัญญาณควบคุมเป็น 0 V ส่วนแกนล็อคก็จะยุบลงไปเพื่อทำการปลดล็อค ส่วนการทำงานของเซนเซอร์นั้นจะมีลักษณะเหมือนกับสวิตช์แม่เหล็ก กล่าวคือ เมื่อสวิตช์แม่เหล็กเข้าใกล้แม่เหล็กถาวร หน้าสัมผัส (Contact) ของสวิตช์จะปิดหรือเชื่อมถึงกันกัน แต่ถ้าสวิตช์ออกห่างแม่เหล็กถาวรหน้าสัมผัสของสวิตช์ก็จะเปิด ซึ่งหมายความว่าเมื่อประตูปิด สายสัญญาณแสดงสถานะของประตู ทั้งสองเส้นก็จะเชื่อมถึงกัน และเมื่อประตูเปิดสายสัญญาณแสดงสถานะของประตูทั้งสองเส้นก็จะเปิดหรือไม่เชื่อมถึงกัน

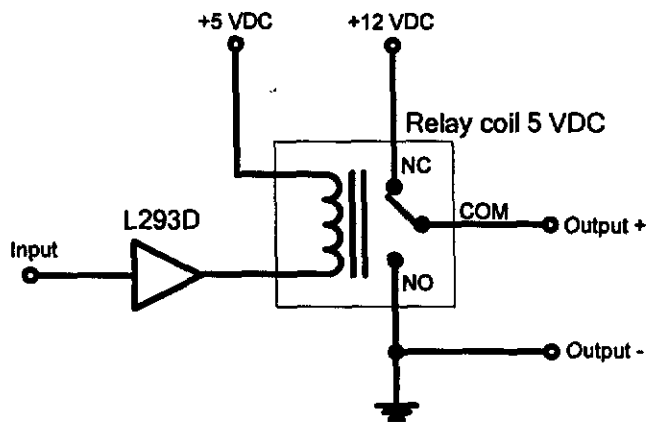


รูปที่ 3.4 : ครอปโบลท์สภาวะล็อคและสถานะของประตูคือปิดอยู่



รูปที่ 3.5 : ดรอปโบลท์ในสภาวะปลดล็อคและสถานะของประตูคือเปิดอยู่

สัญญาณควบคุมที่ส่งมายังดรอปโบลท์นั้นจะมาจากส่วนควบคุมการผ่าน โดยที่สัญญาณจากส่วนควบคุมการผ่าน นั้นจะมีระดับเป็น +5 V และ 0 V ซึ่งไม่เพียงพอที่จะขับดรอปโบลท์ได้จึงต้องมีวงจรขับเข้ามาช่วย ซึ่งสามารถสร้างได้ง่ายโดยใช้รีเลย์ คอยล์ 5 โวลต์ (Relay coil 5 VDC) ต่อร่วมกับไอซีเบอร์ L293D ซึ่งเป็นไอซีขับสัญญาณ ที่ต้องใช้ไอซีขับสัญญาณเข้ามาช่วยก็เพราะว่าสัญญาณจากส่วนควบคุมนั้นมีกำลังไม่เพียงพอที่จะขับให้รีเลย์ทำงาน



รูปที่ 3.6 : วงจรขับดรอปโบลท์

3.2 เครื่องอ่านบัตรแม่เหล็ก (Magnetic Card Reader)

Magnetic Card หรือบัตรแม่เหล็ก ที่เราคุ้นเคยกันดีในรูปแบบของบัตรเงินสด บัตร ATM บัตรเครดิต และบัตรอื่นๆอีกมากมายที่เป็นการ์ดแถบแม่เหล็ก ซึ่งบนตัวบัตรเองจะบันทึกข้อมูลและรายละเอียดต่างๆของบัตรไว้ในรูปของเส้นแรงแม่เหล็กภายในส่วนที่เป็นแถบแม่เหล็กในการ์ดโดยที่แถบแม่เหล็กที่อยู่บนการ์ด ซึ่งเรียกว่า "Track" นั้นตามปกติแล้วจะถูกแบ่งออกเป็น 3 ส่วนด้วยกัน ซึ่งในแต่ละส่วนจะใช้เก็บข้อมูลซึ่งมีความหนาแน่นและลักษณะของข้อมูลที่มีความแตกต่างกัน ดังนี้คือ

แถบแม่เหล็กที่	ความหนาแน่นในการบันทึก	การเข้ารหัสข้อมูล	จำนวนตัวอักษรสูงสุด	ลักษณะของข้อมูลที่เก็บในบัตร
Track 1	210 BPI	ALPHA	79	ชื่อเจ้าของบัตรและหมายเลขบัตร
Track 2	75 BPI	BCD	40	หมายเลขบัตรและวันหมดอายุ
Track 3	210 BPI	BCD	107	หมายเลขบัตรและรหัสพิเศษ

* ความหนาแน่นของการบันทึกข้อมูลมีหน่วยเป็น BPI (Byte Per Inch)

โดยเครื่องอ่านบัตรแม่เหล็ก รุ่น MCR-002 นี้จะอ่านได้เฉพาะข้อมูลที่บันทึกด้วยความหนาแน่น 75 BPI คือเฉพาะในส่วนของแถบแม่เหล็กแถบที่ 2 เท่านั้น โดยมีสัญญาณที่ใช้เชื่อมต่อเพื่ออ่านข้อมูลจากเครื่องอ่านบัตรแม่เหล็กทั้งหมด 5 สัญญาณ โดยเป็นสัญญาณเอาท์พุตส่งออกมา จากเครื่องอ่านบัตรแม่เหล็ก 3 สัญญาณ ส่วนอีก 2 สายที่เหลือก็คือไฟเลี้ยงและกราวด์ ที่ต้องต่อให้เครื่องอ่าน ดังนี้คือ

1. ไฟเลี้ยง (+VCC : สายสีดำ) เป็นแหล่งจ่ายไฟเลี้ยงให้กับเครื่องอ่านบัตรแม่เหล็ก ซึ่งต้องมีค่าเป็น +5 VDC โดยต่อจากภายนอกให้กับเครื่องอ่านบัตรแม่เหล็ก
2. กราวด์ (GND : สายสีน้ำตาล) เป็นจุดอ้างอิงของสัญญาณต่างๆระหว่างเครื่องอ่านบัตรแม่เหล็กกับบอร์ดไมโครคอนโทรลเลอร์หรืออุปกรณ์ที่จะนำมาเชื่อมต่อด้วย
3. สัญญาณข้อมูล (Data : สายสีแดง) เป็นสัญญาณข้อมูลเอาท์พุตที่ส่งออกมาจากเครื่องอ่านบัตรแม่เหล็ก ซึ่งเป็นสัญญาณของข้อมูลที่อ่านได้จากบัตรแม่เหล็ก โดยในการอ่านสัญญาณข้อมูลนี้ต้องอ่านแบบอนุกรมทีละบิตโดยต้องพิจารณาให้สัมพันธ์สอดคล้องกับสัญญาณนาฬิกา ของเครื่องอ่านบัตรด้วยซึ่งสัญญาณข้อมูลที่อ่านได้นี้จะกลับสภาวะกับสัญญาณจริง คือมีสภาวะตรงข้ามกันอยู่ ดังนั้น เมื่ออ่านสัญญาณจากเครื่องอ่านบัตรแม่เหล็กได้แล้วจะต้องกลับสภาวะของสัญญาณเสียก่อนจึงจะได้ข้อมูลที่ถูกต้อง
4. สัญญาณนาฬิกา (Clock : สายสีเหลือง) เป็นสัญญาณนาฬิกาเอาท์พุตที่ส่งออกมาจากเครื่องอ่านบัตรแม่เหล็ก ใช้เป็นสัญญาณอ้างอิงในการอ่านสัญญาณข้อมูลจากบัตร โดยการอ่านสัญญาณข้อมูลแต่ละบิตนั้นต้องอ่านในขณะที่สัญญาณนี้เป็นขอบขาลงเสมอ ซึ่งสัญญาณข้อมูลที่อ่านได้นั้นจะเริ่มต้นจาก บิตที่มีนัยสำคัญต่ำสุด (LSB) ก่อนเป็นอันดับแรก
5. สัญญาณ Present (Present : สายสีน้ำเงิน) เป็นสัญญาณสถานะเอาท์พุตที่ส่งออกมาจากเครื่องอ่านบัตรแม่เหล็ก ทำงานที่ลอจิกศูนย์ เมื่อสัญญาณนี้เป็นศูนย์จะเป็นการบอกให้ทราบว่าเครื่องอ่านบัตรแม่เหล็กเริ่มต้นส่งข้อมูลออกมา หรือมีการนำบัตรแม่เหล็กไปรูดผ่านหัวอ่านสัญญาณของเครื่องอ่านบัตรแม่เหล็ก เมื่อสัญญาณนี้หมดลง(กลับเป็นลอจิกหนึ่ง)แสดงว่าข้อมูลจากการอ่านในครั้งนั้นถูกส่งออกไปหมดแล้ว ซึ่งเราจะใช้ประโยชน์จากสัญญาณนี้เป็นสัญญาณอินเตอร์รัพท์ ให้ไมโครคอนโทรลเลอร์เพื่อทำการอ่านข้อมูล

การอ่านข้อมูลจากบัตรแม่เหล็ก

การใช้เครื่องอ่านบัตรแม่เหล็กเพื่อให้อ่านข้อมูลจากบัตรแม่เหล็กนั้น จะใช้วิธีการนำบัตรแม่เหล็กตรงส่วนที่เป็นแถบแม่เหล็กไปรูดผ่านหัวอ่านของเครื่องอ่านตามทิศทางและตำแหน่งที่กำหนดไว้เท่านั้น เครื่องอ่านบัตรแม่เหล็กนี้สามารถอ่านข้อมูลจากบัตรแม่เหล็กได้เฉพาะข้อมูลใน Track 2 เท่านั้น โดยฟอร์แม็ต (format) ข้อมูลในแถบแม่เหล็ก Track 2 ซึ่งใช้บันทึกสัญญาณด้วยความหนาแน่นของสนามแม่เหล็ก 75 BPI ซึ่งจะบรรจุจำนวนรหัสข้อมูลต่างๆใน Track นี้ได้สูงสุดไม่เกิน 40 ตัวอักษร โดยนับรวมรหัสควบคุมและรหัสตรวจสอบด้วย ซึ่งมีรายละเอียดดังนี้คือ

SS	PAN	FS	Addition Data	ES	LRC

SS (Start Sentinel) – รหัสข้อมูลที่แสดงการเริ่มต้นของข้อมูลในบัตรซึ่งมีรหัสข้อมูลเป็น 0BH (;)

PAN – ข้อมูลแสดงหมายเลขของบัตร ซึ่งมีจำนวนข้อมูลในส่วนนี้สูงสุดไม่เกิน 19 หลัก

FS (Field Separator) – รหัสข้อมูลแสดงการแบ่งแยกข้อมูล มีรหัสข้อมูลเป็น 0DH (=)

Additional Data – เป็นข้อมูลเพิ่มเติมอื่นๆของบัตร เช่น เดือน/ปี ที่ออกบัตร

ES (End Sentinel) – รหัสข้อมูลแสดงการสิ้นสุดของข้อมูลในบัตร มีรหัสข้อมูลเป็น 0FH (?)

LRC (Longitudinal Redundancy Check) – ข้อมูลตรวจสอบความผิดพลาด

ลักษณะการจัดเก็บข้อมูลของ Track 2 จะเก็บข้อมูลด้วยการเข้ารหัสแบบ "Modulo 5" ซึ่งในแต่ละชุดข้อมูลจะประกอบด้วยข้อมูล 5 บิต โดย 4 บิตแรกคือบิต D0 – D3 จะเป็นรหัสข้อมูลแบบ BCD ส่วนบิตที่ 5 คือบิต D4 เป็นพาริตี้บิตแบบคี่ เพื่อใช้ตรวจสอบความถูกต้องของข้อมูลในแต่ละชุดข้อมูลที่อ่านได้ ซึ่งหากค่าของพาริตี้บิตผิดพลาด แสดงว่าการอ่านข้อมูลนั้นล้มเหลว โดยเริ่มทำการอ่านข้อมูลเมื่อสัญญาณ Present เริ่มเป็นศูนย์ก่อน และทำการอ่านข้อมูลในทุกๆขอบขาลงของสัญญาณนาฬิกาเสมอ ซึ่งข้อมูลในส่วนก่อนเริ่มต้นและหลังจากสิ้นสุด ของการอ่านนี้จะมีค่าเป็นศูนย์นำหน้าและปิดท้ายข้อมูลจริงอยู่ ซึ่งข้อมูลในส่วนนี้เราไม่ต้องสนใจ แต่ให้ตรวจสอบและรอจนกว่าจะเริ่มเป็นข้อมูล Start บิต ซึ่งบิตเริ่มต้นของรหัส Start หรือ 0BH ต้องเริ่มด้วย 1 เป็นบิตแรกเสมอ ดังนั้นในการอ่านเราต้องรอจนกว่าจะพบสัญญาณข้อมูลที่มีค่าเป็น 1 (data = logic 0 เพราะกลับสภาวะกันอยู่) จึงเริ่มเก็บข้อมูลชุดละ 5 บิตไปเรื่อยๆ จนถึงรหัสจบ (0FH) ซึ่งเมื่อพบรหัสจบแล้วจะมีข้อมูลตามมาอีก 1ชุดซึ่งเป็นข้อมูลสำหรับตรวจสอบความผิดพลาดของการอ่านข้อมูลทั้งหมดใน Track 2 เรียกว่า LRC ซึ่งค่าของ LRC สามารถหาได้จากการนำเอาข้อมูลในแต่ละชุด(ไม่คิดพาริตี้บิต) ตั้งแต่เริ่มต้น จนสิ้นสุดมาทำการ XOR กับข้อมูลชุดถัดไปเรื่อยๆ ตามลำดับ ซึ่งหากผลลัพธ์สุดท้ายที่ได้ไม่เท่ากับค่าของ LRC ที่อ่านมาได้แสดงว่าการอ่านข้อมูลทั้งหมดล้มเหลว

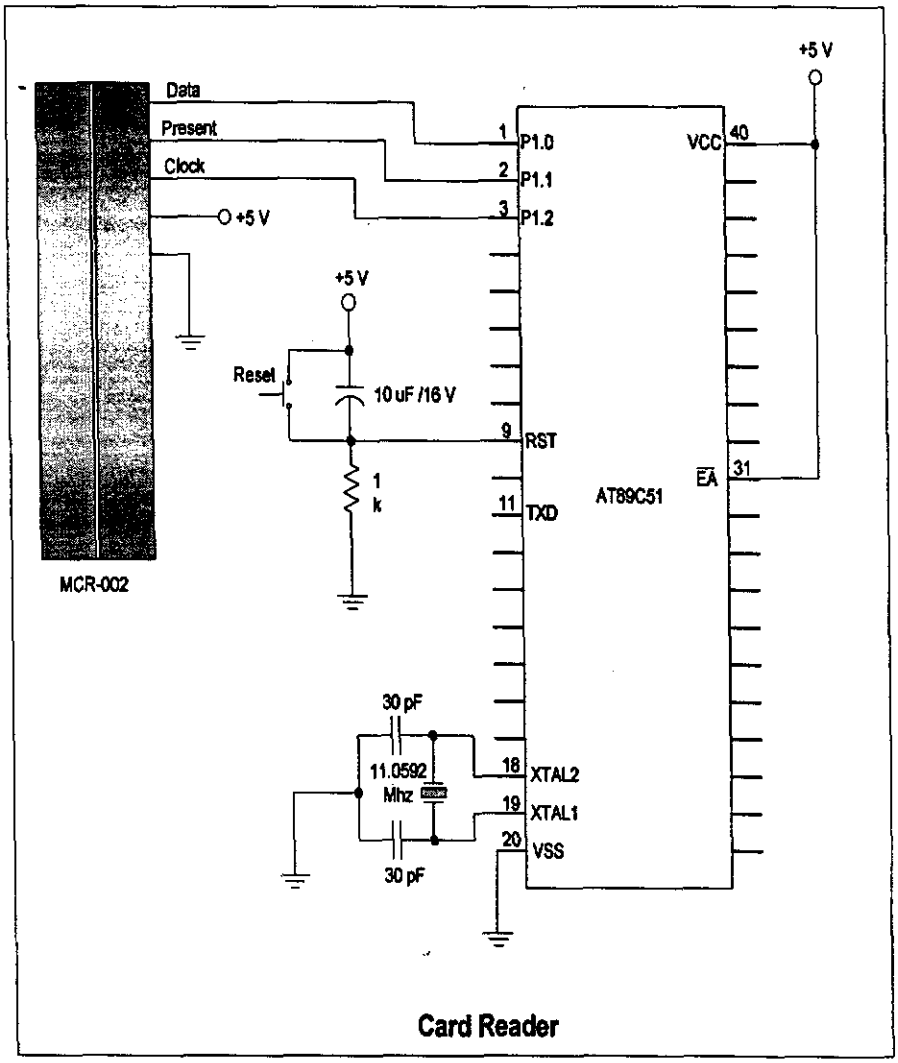
สำหรับข้อมูลของรหัส BCD ที่ใช้สำหรับเครื่องอ่านบัตรแม่เหล็กนี้จะเป็นข้อมูลชุดละ 5 บิต โดยเป็นข้อมูลจริง 4 บิตและเป็นรหัสตรวจสอบพาริตี้อีก 1 บิต ซึ่งมีจำนวนทั้งหมด 16 อักขระดังนี้คือ

Parity (D4)	D3	D2	D1	D0	Character	Function
1	0	0	0	0	0 (0H)	Data
0	0	0	0	1	1 (1H)	Data
0	0	0	1	0	2 (2H)	Data
1	0	0	1	1	3 (3H)	Data
0	0	1	0	0	4 (4H)	Data
1	0	1	0	1	5 (5H)	Data
1	0	1	1	0	6 (6H)	Data
0	0	1	1	1	7 (7H)	Data
0	1	0	0	0	8 (8H)	Data
1	1	0	0	1	9 (9H)	Data
1	1	0	1	0	: (AH)	Control
0	1	0	1	1	; (BH)	Start Sentinel
1	1	1	0	0	< (CH)	Control
0	1	1	0	1	= (DH)	Field Separator
0	1	1	1	0	> (EH)	Control
1	1	1	1	1	? (FH)	End Sentinel

ตารางที่ 3.1 : แสดงข้อมูลของรหัส BCD สำหรับเครื่องอ่านบัตรแม่เหล็ก

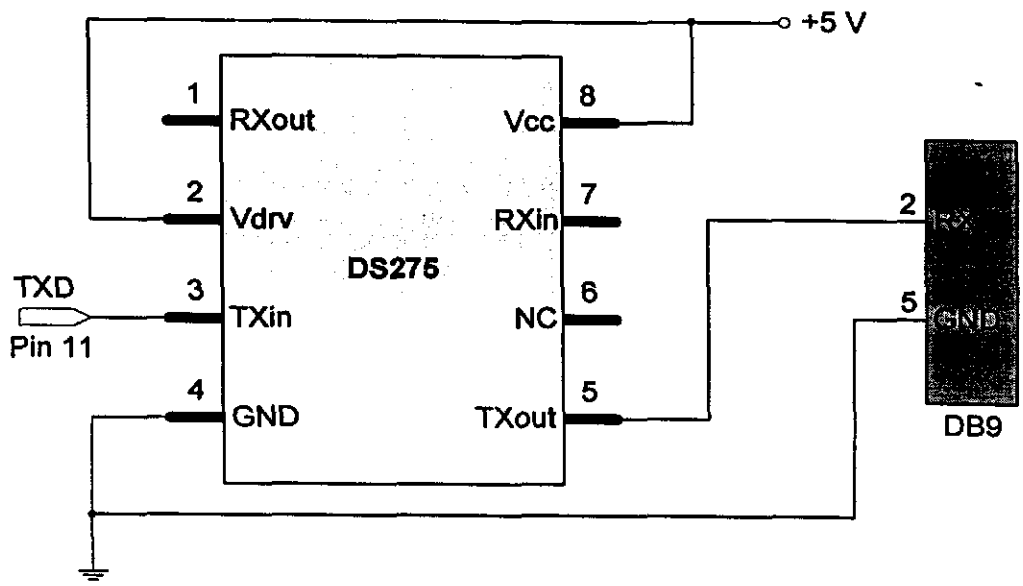
ในการอ่านค่าจากบัตรแม่เหล็กและตรวจสอบความถูกต้องของข้อมูลที่อ่านได้นั้น จะใช้ไมโครคอนโทรลเลอร์ เบอร์ AT89C51 นำมาต่อเป็นวงจรดังรูปที่ 3.7 แล้วทำการเขียนคำสั่งควบคุมการทำงานของไมโครคอนโทรลเลอร์ให้ทำการรับค่าและตรวจสอบค่าที่ส่งมาจากหัวอ่านบัตรแม่เหล็กจากนั้นก็แปลงเป็นรหัสแอสกี้โดยการนำข้อมูลที่อ่านได้มาบวกด้วยค่า 30H ก็จะได้ค่าที่ออกมาเป็นค่าตามตารางรหัสแอสกี้ เช่น หัวอ่านบัตรแม่เหล็กอ่านได้ค่า 2H ซึ่งหมายถึงข้อมูลที่เป็นเลข " 2 " ถ้าส่งค่า 2H นี้ไปให้คอมพิวเตอร์ คอมพิวเตอร์จะมองเห็นค่านี้เป็นข้อมูลที่ไม่ใช่เลข " 2 " เนื่องจากคอมพิวเตอร์ประมวลผลข้อมูลโดยใช้รหัสแอสกี้ ซึ่งข้อมูลที่เป็นเลข "2" ในรหัสแอสกี้นั้นมีค่าเท่ากับ 32H = 2H + 30H

จากนั้นไมโครคอนโทรลเลอร์ก็จะส่งค่าที่เป็นรหัสแอสกี้ออกไปทางพอร์ตสื่อสารข้อมูลอนุกรมแบบอะซิงโครนัสออกไป ให้ส่วนควบคุมการผ่านโดยผ่านทางขา 11 ของไมโครคอนโทรลเลอร์ เพื่อให้ส่วนควบคุมการผ่านนำข้อมูลไปใช้งานต่อไป ในขั้นตอนนี้จะใช้โปรแกรมไฮเปอร์เทอมินอล (HyperTerminal) ตรวจดูข้อมูลที่ไมโครคอนโทรลเลอร์ส่งออกมาทางพอร์ตสื่อสารข้อมูลอนุกรม

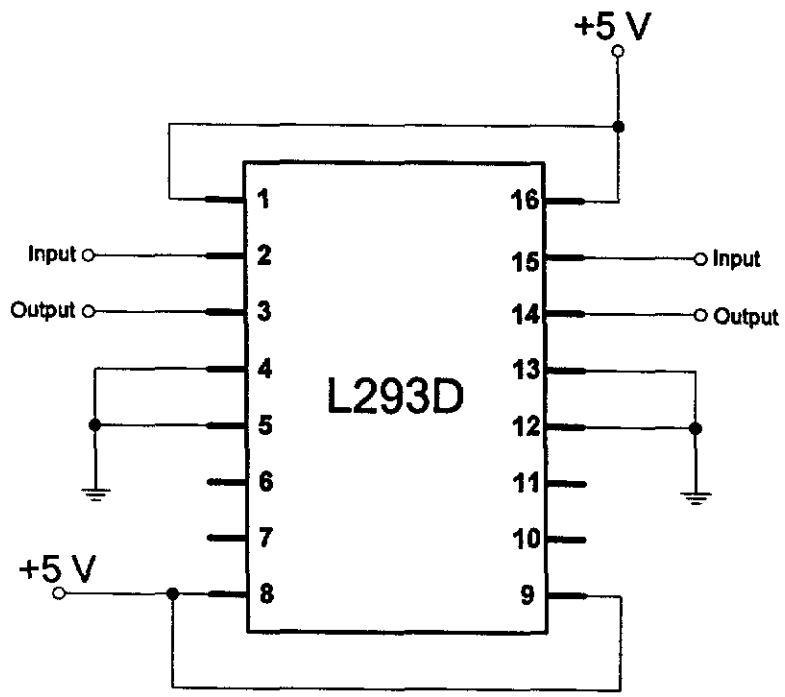


รูปที่ 3.7 : วงจรไมโครคอนโทรลเลอร์ที่ใช้ควบคุมการทำงานของเครื่องอ่านบัตรแม่เหล็ก

วงจรในรูปที่ 3.7 นั้นก็คือส่วนอ่านบัตรของระบบควบคุมการผ่านเข้าออก ในโครงงานนี้ สัญญาณที่เครื่องอ่านบัตรแม่เหล็กปล่อยออกมานั้นจะเกิดขึ้นในขณะที่มีการรูดบัตรแม่เหล็กผ่านหัวอ่านเท่านั้น สัญญาณเหล่านี้จะเกิดขึ้นและหายไปอย่างรวดเร็วจึงไม่สามารถใช้ออสซิลโลสโคปตรวจวัดได้ การจะดูข้อมูลที่เครื่องอ่านบัตรแม่เหล็กอ่านได้นั้นจะดูผ่านโปรแกรมไฮเปอร์เทอมินอล ที่มีมากับระบบปฏิบัติการไมโครซอฟท์วินโดวส์ ซึ่งจะต้องมีการเชื่อมต่อไมโครคอนโทรลเลอร์ของส่วนอ่านบัตรเข้ากับคอมพิวเตอร์ทางพอร์ตอนุกรม สัญญาณที่ส่งออกมาจากไมโครคอนโทรลเลอร์จะต้องผ่านวงจรแปลงระดับลอจิกเสียก่อนที่จะส่งให้กับคอมพิวเตอร์ เพราะระดับของลอจิกในคอมพิวเตอร์จะไม่เท่ากับระดับของลอจิกในไมโครคอนโทรลเลอร์ วงจรแปลงระดับลอจิกในโครงงานนี้จะใช้ไอซีเบอร์ DS275 ซึ่งมีการต่อใช้งานดังรูปที่ 3.8 แล้วส่งสัญญาณเอาท์พุตออกไปยังพอร์ตอนุกรมของคอมพิวเตอร์ ผ่านหัวต่อ DB-9



รูปที่ 3.8 : การต่อใช้งานไอซี DS275 ในวงจรแปลงระดับลอจิก

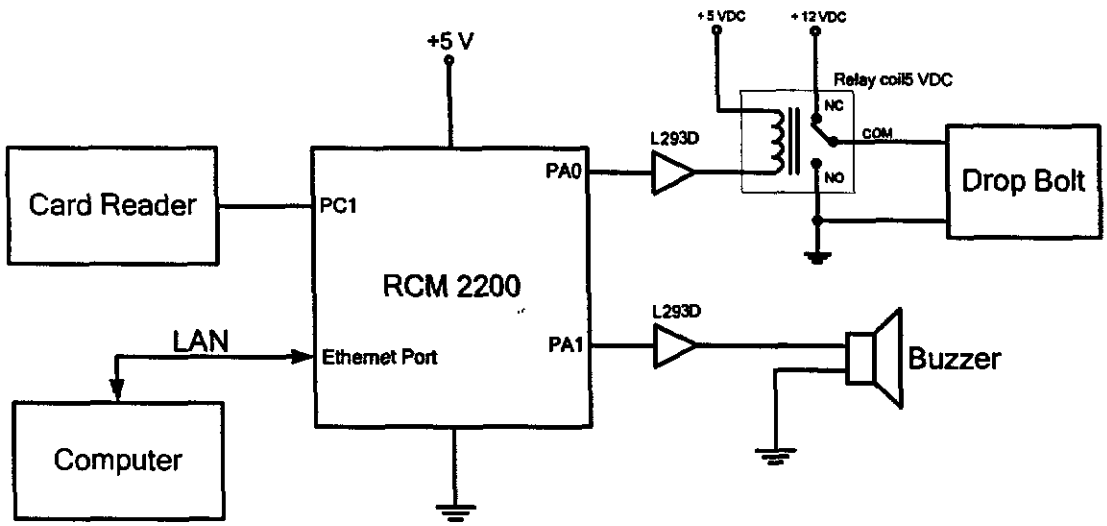


รูปที่ 3.9 : การต่อใช้งานไอซี L293D ในวงจรขับสัญญาณควบคุม

3.3 ส่วนควบคุมการผ่าน (Access Controller)

ในส่วนควบคุมการผ่านนี้ได้ใช้โมดูลแร็บบิต RCM2200 เป็นหน่วยประมวลผลและควบคุมการทำงานของส่วนนี้ สำหรับรายละเอียดของโมดูลแร็บบิต RCM2200 นี้ได้กล่าวไว้แล้วในบทที่ 2 หน้าที่หลักๆของส่วนนี้คือรับหมายเลขบัตรที่ส่งมาจากเครื่องอ่านบัตรแม่เหล็ก แล้วทำการส่งต่อไปยังคอมพิวเตอร์โดยผ่านเครือข่ายท้องถิ่น จากนั้นก็รอการตอบกลับว่าหมายเลขบัตรที่รับมานี้ได้รับการอนุญาตให้ผ่านเข้ามาได้หรือไม่ ถ้าหมายเลขบัตรนี้ได้รับอนุญาตส่วนควบคุมการผ่าน ก็จะส่งสัญญาณไปปลดล็อคดรอปโบลท์แต่ถ้าหมายเลขบัตรนี้ไม่ได้รับอนุญาตส่วนควบคุมการผ่านก็จะส่งสัญญาณไปสั่งให้บัซเซอร์ (Buzzer) ส่งสัญญาณเสียงเพื่อแจ้งเตือนให้ผู้ใช้ได้รับทราบ

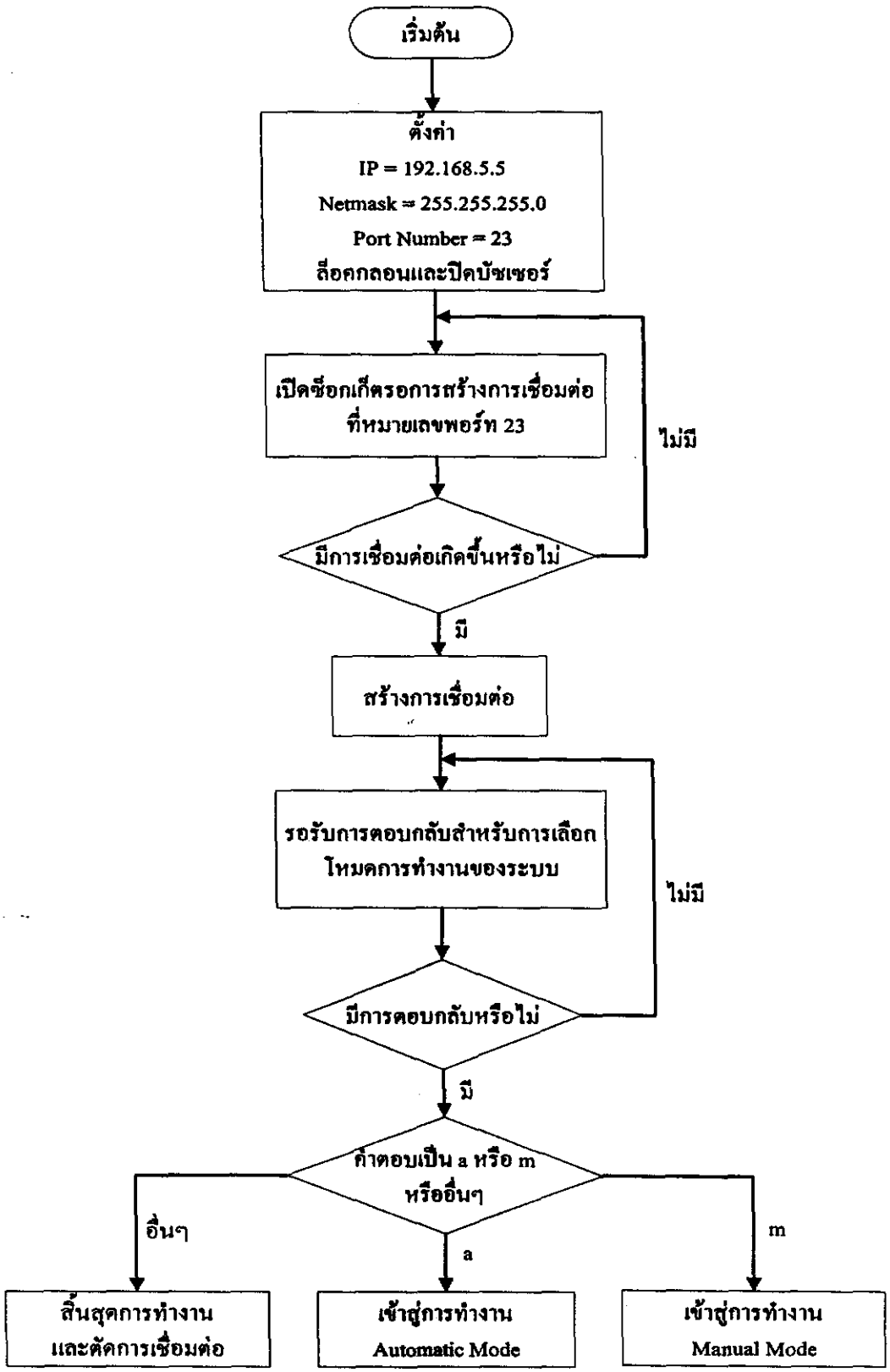
สำหรับชุดคำสั่งที่ใช้ควบคุมการทำงานของโมดูลแร็บบิตนั้นจะใช้ชุดคำสั่งภาษาไดนามิคซีซึ่งเป็นชุดคำสั่งที่พัฒนาขึ้นมาเพื่อใช้งานกับไมโครโปรเซสเซอร์แร็บบิต 2000 ซึ่งเป็นหน่วยประมวลผลของโมดูลแร็บบิต การต่อโมดูลแร็บบิตเข้ากับส่วนอื่นๆแสดงดังรูปที่ 3.10



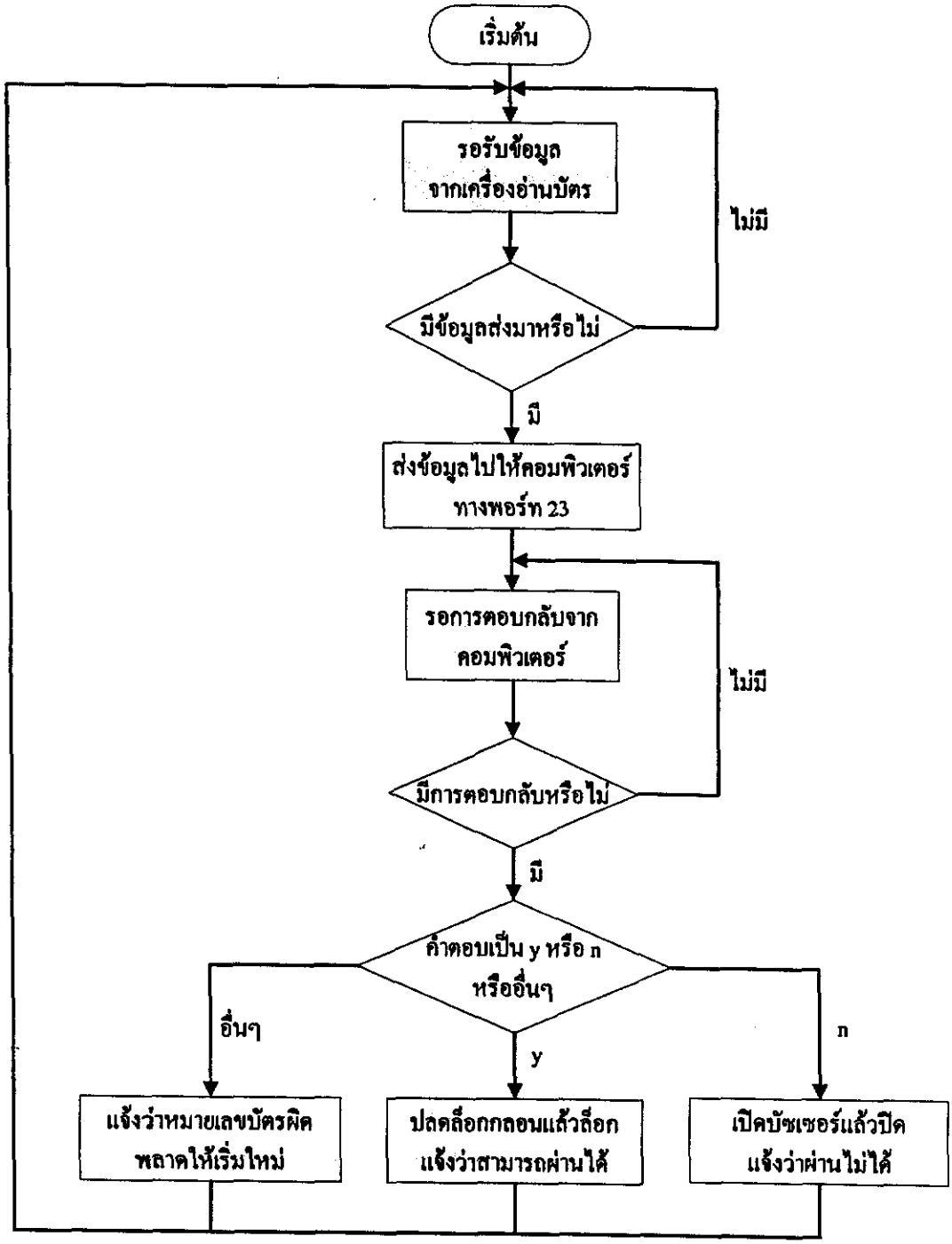
รูปที่ 3.10 : การต่อโมดูลแร็บบิตเข้ากับส่วนอื่นๆ

ขา PC1 ของโมดูลแร็บบิตนั้นก็คือขารับข้อมูลอนุกรมแบบอะซิงโครนัสของพอร์ตอนุกรมดี (Serial Port D) พอร์ตสื่อสารข้อมูลอนุกรมและพอร์ตขนานของโมดูลแร็บบิตจะใช้ขาเชื่อมต่อร่วมกัน ขา PC1 นี้จะต่ออยู่กับเครื่องอ่านบัตรเพื่อรับข้อมูลจากเครื่องอ่านบัตร ที่ขา PA0 ใช้เป็นขาเอาท์พุตเพื่อส่งสัญญาณไปควบคุมการทำงานของดรอปโบลท์โดยผ่านวงจรขับ และที่ขา PA1 ใช้เป็นขาเอาท์พุตเพื่อส่งสัญญาณไปควบคุมการทำงานของบัซเซอร์โดยผ่านไอซีขับสัญญาณ L293D ส่วนอีเธอร์เน็ตพอร์ตของโมดูลแร็บบิตนั้นจะเชื่อมต่อกับการ์ดแลนของคอมพิวเตอร์เป็นเครือข่ายท้องถิ่นโดยใช้สายคู่ตีเกลียว (Unshield Twisted Pair : UTP) ซึ่งเข้าหัวเชื่อมต่อทั้งสองด้านแบบไขว้กัน

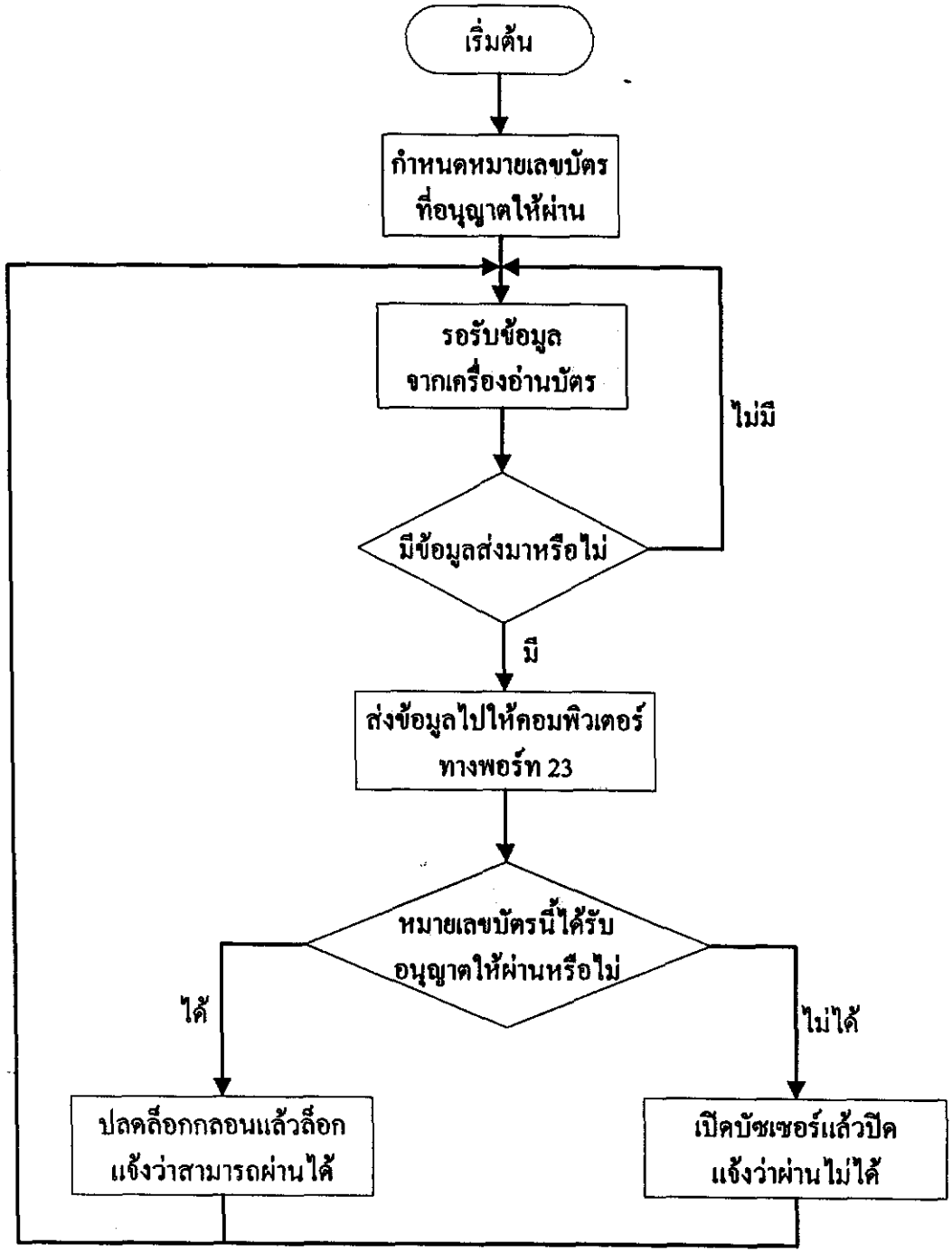
การเขียนโปรแกรมควบคุมการทำงานของโมดูลแร็บบิตจะเป็นไปตามแผนผังดังรูปที่ 3.11



รูปที่ 3.11 : แผนผังแสดงการเขียนโปรแกรมให้โมดูลแร็บบิต



รูปที่ 3.12 : แผนผังแสดงการเขียนโปรแกรมใน Manual Mode



รูปที่ 3.13 : แผนผังแสดงการเขียนโปรแกรมใน Automatic Mode

Manual Mode

การทำงานในโหมดนี้จะต้องมีผู้คอยกดคีย์ " y " หรือ " n " เพื่อตอบกลับไปให้ส่วนควบคุมการผ่านทราบว่าจะอนุญาตให้หมายเลขบัตรที่ได้รับมานี้ผ่านได้หรือไม่ ถ้าหากส่วนควบคุมการผ่านได้รับการตอบกลับเป็น y ก็จะดำเนินการปลดล็อคกลอนประตูไฟฟ้าในระยะเวลาสั้นๆหลังจากนั้นก็จะล็อคกลอนประตูไฟฟ้า ที่กลอนประตูไฟฟ้าจะมีวงจรหน่วงเวลาว่ากลอนจะปลดล็อคเป็นเวลาเท่าใด ซึ่งส่วนนี้สามารถปรับได้ที่กลอนประตูไฟฟ้า แต่ถ้าหากส่วนควบคุมการผ่านได้รับการตอบกลับเป็น n ก็จะดำเนินการเปิดบัซเซอร์เพื่อส่งสัญญาณเสียงแจ้งเจ้าของบัตรว่าไม่สามารถผ่านได้และไม่ปลดล็อคกลอนประตูไฟฟ้า หลังจากนั้นส่วนควบคุมการผ่านก็จะรอรับหมายเลขบัตรต่อไป

Automatic Mode

การทำงานในโหมดนี้จะต้องกำหนดรายการหมายเลขบัตรที่ได้รับอนุญาตให้ผ่านได้ ไว้ที่ส่วนควบคุมการผ่าน เมื่อมีหมายเลขบัตรเข้ามา ส่วนควบคุมการผ่านก็จะตรวจสอบหมายเลขบัตรนั้นกับรายการหมายเลขบัตรที่ได้รับอนุญาตให้ผ่านได้ ถ้าหากมีหมายเลขบัตรนั้นในรายการหมายเลขบัตรที่ได้รับอนุญาตให้ผ่านได้ ส่วนควบคุมการผ่านก็จะดำเนินการปลดล็อคกลอนประตูไฟฟ้า แต่ถ้าไม่มีหมายเลขบัตรนั้นในรายการหมายเลขบัตรที่ได้รับอนุญาตให้ผ่านได้ ส่วนควบคุมการผ่านก็จะดำเนินการเปิดบัซเซอร์เพื่อส่งสัญญาณเสียงแจ้งเตือนและไม่ปลดล็อคกลอนประตูไฟฟ้า หลังจากนั้นส่วนควบคุมการผ่านก็จะรอรับหมายเลขบัตรต่อไป

3.4 คอมพิวเตอร์ควบคุม

เป็นคอมพิวเตอร์ที่ใช้รับหมายบัตรจากโมดูลแร็บบิตโดยผ่านเครือข่ายท้องถิ่นและตอบกลับไปให้โมดูลแร็บบิตว่าจะอนุญาตให้ผ่านได้หรือไม่ การกำหนดค่าให้คอมพิวเตอร์ควบคุมมองเห็นโมดูลแร็บบิตได้นั้นจะต้องกำหนดค่าไอพีแอดเดรสของคอมพิวเตอร์ควบคุมให้มีหมายเลขเครือข่ายเหมือนกับหมายเลขเครือข่ายของโมดูลแร็บบิตและตั้งค่าเน็ตมาส์ค ให้เหมือนกับของโมดูลแร็บบิตด้วย ค่าต่างๆที่ต้องกำหนดให้กับคอมพิวเตอร์ควบคุมแสดงไว้ในตารางที่ 3.2

ส่วนที่ต้องกำหนด	คอมพิวเตอร์ควบคุม	โมดูลแร็บบิต
IP Address	161.246.18.99	161.246.18.98
Netmask	255.255.255.0	255.255.255.0

ตารางที่ 3.2 : การกำหนดค่าให้กับคอมพิวเตอร์ควบคุม

การสื่อสารระหว่างคอมพิวเตอร์ควบคุมกับโมดูลแร็บบิตนั้นจะใช้แอพลิเคชันเทลเน็ต ที่มีมาให้กับระบบปฏิบัติการวินโดวส์ เทลเน็ตเป็นแอพพลิเคชันที่ใช้สำหรับเชื่อมต่อเพื่อใช้งานโฮสต์จากระยะไกล (Remote) เทลเน็ตใช้โปรโตคอลทีซีพีไอพีในการสื่อสาร และทำงานที่พอร์ตหมายเลข 23

บทที่ 4
การทดลองและผลการทดลอง

4.1 ทดลองอ่านข้อมูลจากบัตรแม่เหล็กด้วยเครื่องอ่านบัตรแม่เหล็กที่สร้างขึ้นมา

ข้อมูลที่เครื่องอ่านบัตรแม่เหล็กนี้สามารถอ่านได้ก็คือข้อมูลในแถบแม่เหล็กแถบที่ 2 ซึ่งจะเป็นข้อมูลในส่วนของหมายเลขบัตรและวันหมดอายุของบัตร และเนื่องจากสัญญาณที่เครื่องอ่านบัตรแม่เหล็กจ่ายออกมาในขณะที่ทำการรูดบัตรแม่เหล็กผ่านหัวอ่านนั้น เกิดขึ้นและหายไปอย่างรวดเร็ว ทำให้ไม่สามารถใช้ออสซิลโลสโคปวัดสัญญาณได้ จึงต้องใช้วงจรไมโครคอนโทรลเลอร์เข้ามารับสัญญาณและประมวลผลเป็นข้อมูลออกมารวมทั้งตรวจสอบความถูกต้องของข้อมูลที่รับได้ แล้วจึงส่งออกไปยังโปรแกรมไฮเปอร์เทอมินอลเพื่อสังเกตผลที่ได้

อุปกรณ์ที่ใช้ในการทดลอง

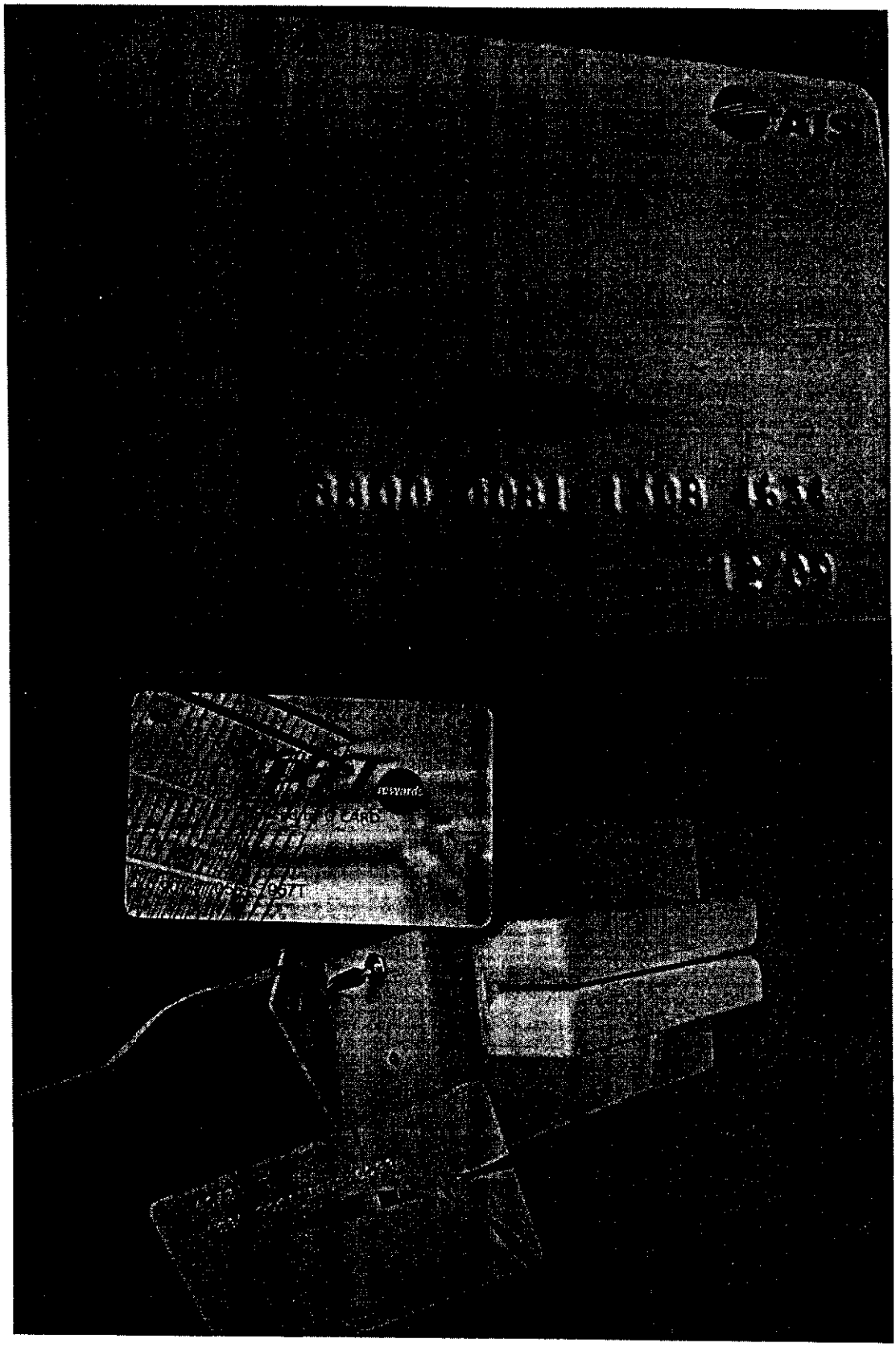
1. เครื่องอ่านบัตรแม่เหล็กที่สร้างขึ้นมาและบัตรแม่เหล็ก อย่างน้อย 2 ใบ
2. สายเชื่อมต่อทางพอร์ตอนุกรมระหว่างเครื่องอ่านบัตรแม่เหล็กกับคอมพิวเตอร์
3. คอมพิวเตอร์และโปรแกรมไฮเปอร์เทอมินอล

จุดประสงค์การทดลอง

1. ทดสอบการทำงานของเครื่องอ่านบัตรแม่เหล็กที่สร้างขึ้นมา
2. ดูข้อมูลที่บันทึกในบัตรแม่เหล็กตรงส่วนแถบแม่เหล็กแถบที่ 2 แต่ละใบ

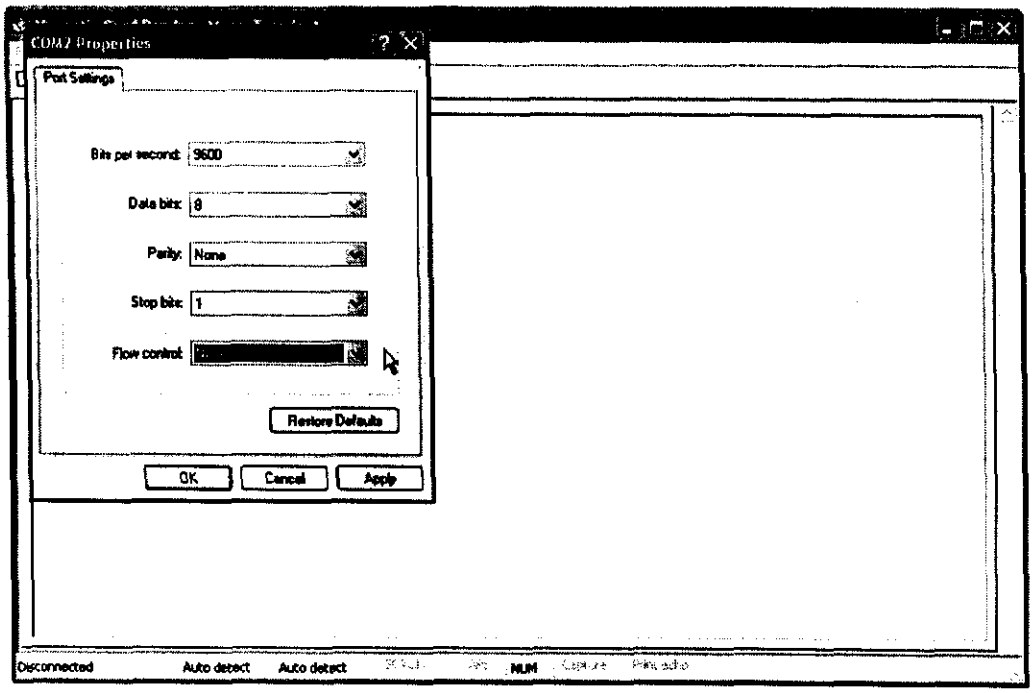
ขั้นตอนการทดลอง

1. เปิดการใช้งานวงจรเครื่องอ่านบัตรแม่เหล็กที่สร้างขึ้นมาและเตรียมบัตรแม่เหล็กที่ต้องการอ่านค่า จากนั้นทำการเชื่อมต่อสายสัญญาณข้อมูลจากเครื่องอ่านบัตรแม่เหล็กเข้ากับกับพอร์ตอนุกรมของคอมพิวเตอร์

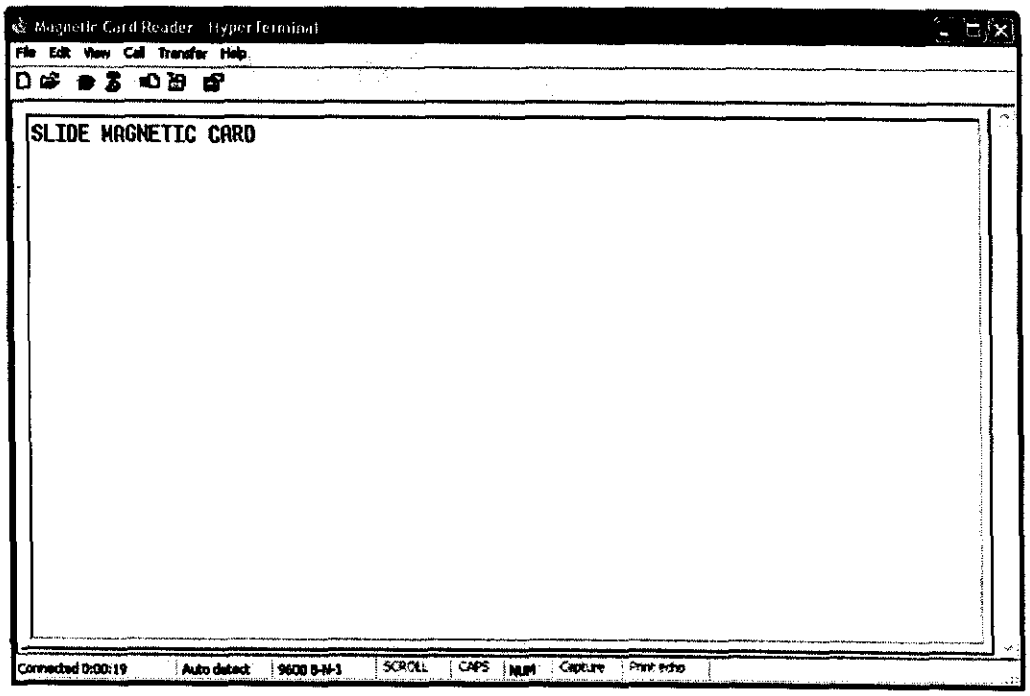


รูปที่ 4.1 : บัตรแม่เหล็กและเครื่องอ่านบัตรแม่เหล็กที่ใช้ในการทดลอง

2. เปิดโปรแกรมไฮเปอร์เทอมินอลขึ้นมา เพื่อใช้แสดงผลข้อมูลที่เครื่องอ่านบัตรแม่เหล็กสามารถอ่านได้ จากนั้นตั้งค่าการใช้งานตามรูปที่ 4.2 ก็จะได้หน้าต่างเริ่มต้นดังรูปที่ 4.3

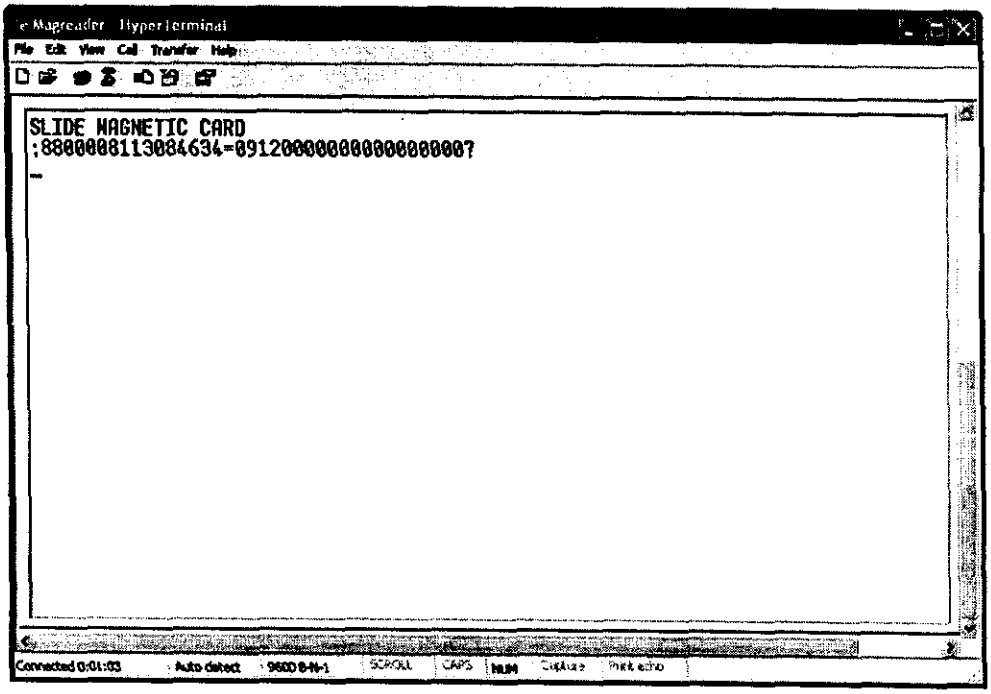


รูปที่ 4.2 : การตั้งค่าการใช้งานโปรแกรมไฮเปอร์เทอมินอลในการทดลอง



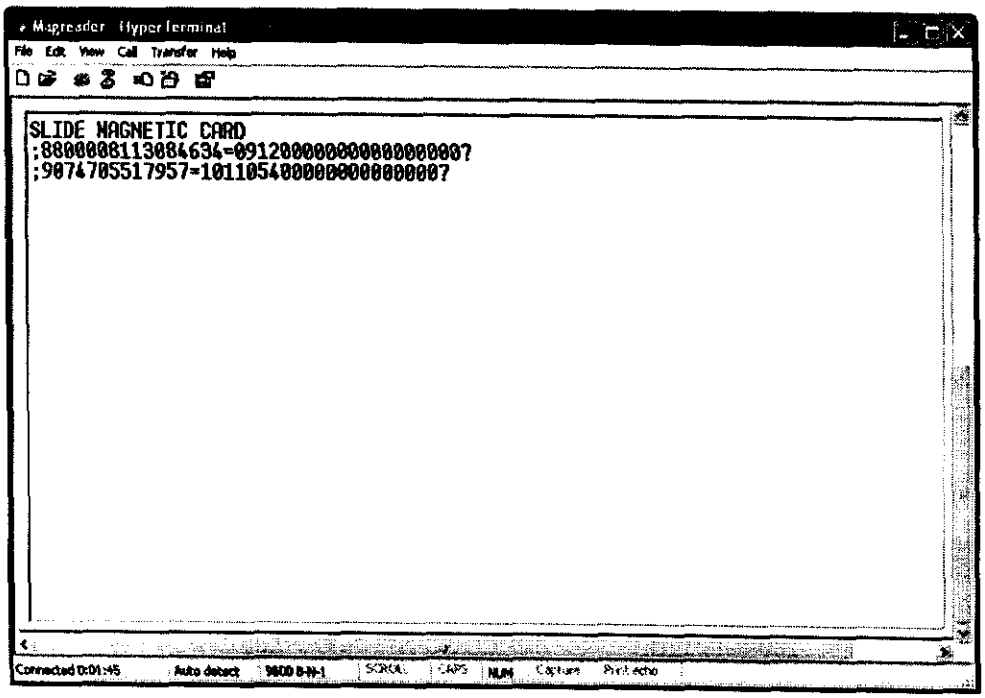
รูปที่ 4.3 : หน้าต่างเริ่มต้นของโปรแกรมหลังจากตั้งค่าการใช้งานเสร็จ

3. นำบัตรแม่เหล็กที่เตรียมไว้มารูดผ่านหัวอ่านของเครื่องอ่านบัตรแม่เหล็ก จะได้ผลดังรูปที่ 4.4



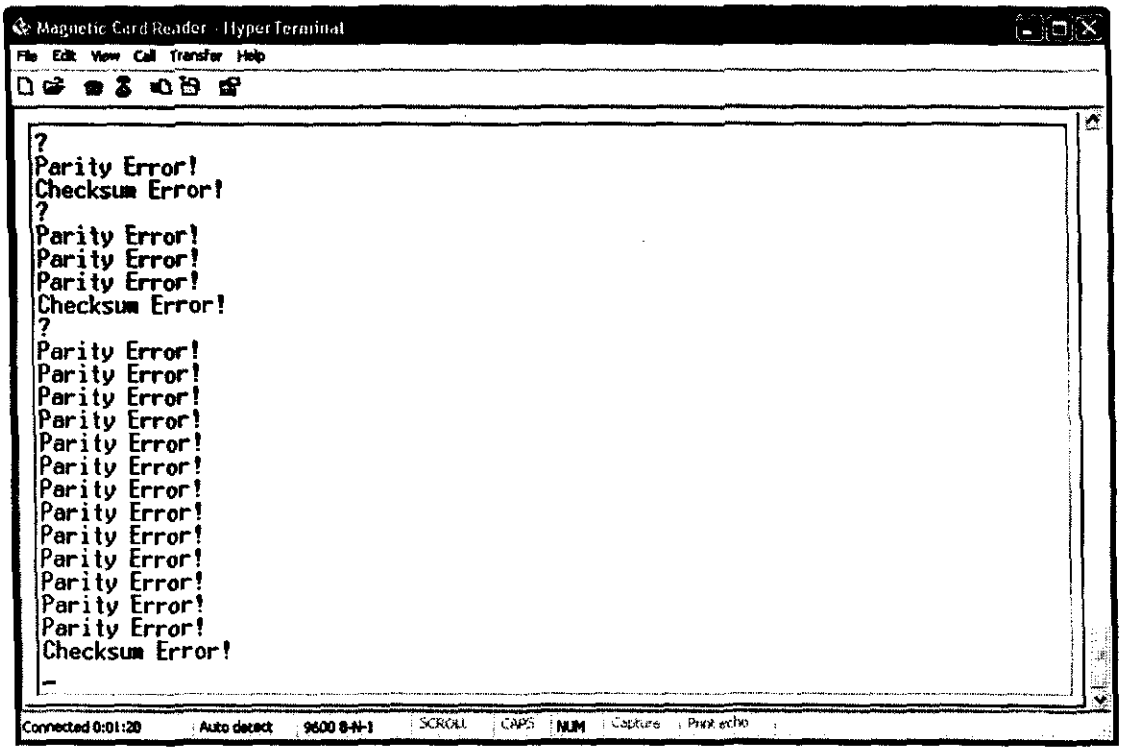
รูปที่ 4.4 : แสดงข้อมูลที่อ่านได้ถูกต้อง

4. นำบัตรแม่เหล็กอีกใบมารูดผ่านหัวอ่านของเครื่องอ่านบัตรแม่เหล็ก ก็จะได้ผลดังรูปที่ 4.5



รูปที่ 4.5 : แสดงข้อมูลที่อ่านได้ถูกต้องจากบัตรแม่เหล็กอีกใบ

5. ทำการรูดบัตรแม่เหล็กโดยทิศทางในการรูดผ่านหัวอ่านจะสวนทางกับทิศทางในการรูดตามปกติ



รูปที่ 4.6 : ข้อมูลที่อ่านได้เมื่อรูดผ่านหัวอ่านแบบสวนทางกับทิศทางในการรูดตามปกติ

ผลการทดลอง

ผลที่ได้จากการทดลองข้างต้นแสดงให้เห็นว่าข้อมูลที่อ่านได้จะขึ้นต้นด้วยสัญลักษณ์ " ; " และสิ้นสุดด้วยสัญลักษณ์ " ? " ซึ่งเป็นรูปแบบข้อมูลในแถบแม่เหล็กส่วนที่สองตามที่ได้กล่าวไปแล้วในบทที่ 3 ตรงส่วนของเครื่องอ่านบัตรแม่เหล็ก MCR-002 ดังนี้

SS (Start Sentinel) – รหัสข้อมูลที่แสดงการเริ่มต้นของข้อมูลในบัตรซึ่งมีรหัสข้อมูลเป็น 0BH (;)

ES (End Sentinel) – รหัสข้อมูลแสดงการสิ้นสุดของข้อมูลในบัตร มีรหัสข้อมูลเป็น 0FH (?)

โดยที่บัตรแม่เหล็กแต่ละใบจะมีข้อมูลในแถบแม่เหล็กส่วนที่สองแตกต่างกันตามการบันทึกข้อมูลของผู้ผลิตบัตร สังเกตหมายเลขบัตรที่อ่านได้จากแถบแม่เหล็กจะเหมือนกับหมายเลขที่อยู่ด้านหน้าบัตร

ในการรูดบัตรแม่เหล็กผ่านหัวอ่านของเครื่องอ่านบัตรแม่เหล็ก ขอบบัตรจะต้องอยู่ชิดกับท้ายร่องสไลด์ตลอดการรูด จึงจะสามารถอ่านข้อมูลได้อย่างถูกต้อง แต่ถ้าขอบบัตรไม่อยู่ชิดกับท้ายร่องสไลด์ตลอดการรูด ข้อมูลที่อ่านได้จะเกิดความผิดพลาด แต่ในบางจังหวะก็ยังสามารถอ่านข้อมูลได้อย่างถูกต้อง และถ้าทำการรูดบัตรแม่เหล็กโดยทิศทางในการรูดผ่านหัวอ่านสวนทางกับทิศทางในการรูดตามปกติ ก็จะไม่มีโอกาสที่จะอ่านข้อมูลได้อย่างถูกต้องเนื่องจากเครื่องอ่านจะอ่านเจอรหัสข้อมูลแสดงการสิ้นสุดของข้อมูลในบัตรก่อนข้อมูลส่วนอื่นๆ

4.2 ทดลองสื่อสารระหว่างส่วนควบคุมการผ่านกับคอมพิวเตอร์ควบคุมโดยผ่านเครือข่ายท้องถิ่น

ทำการสื่อสารระหว่างส่วนควบคุมการผ่านกับคอมพิวเตอร์ควบคุมโดยผ่านเครือข่ายท้องถิ่น โดยใช้คำสั่ง ping ในโปรแกรม Command Prompt และทดลองการเชื่อมต่อผ่านเทลเน็ตโดยที่ยังไม่ได้ต่อเครื่องอ่านบัตรแม่เหล็กและดรอปโบลท์เข้ากับส่วนควบคุมการผ่าน

อุปกรณ์ที่ใช้ในการทดลอง

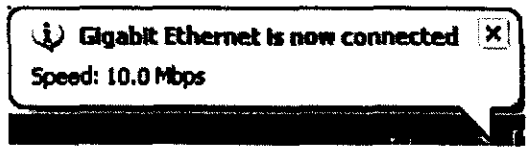
1. ส่วนควบคุมการผ่านที่สร้างขึ้นมาในโครงงาน
2. สายแลนแบบไขว้ (Cross Over Cable)
3. คอมพิวเตอร์ที่มี Network Interface Card (เรียกกันทั่วไปว่าการ์ดแลน)

จุดประสงค์การทดลอง

1. สร้างเครือข่ายท้องถิ่นอย่างง่าย
2. ตรวจสอบการสื่อผ่านเครือข่ายท้องถิ่นระหว่างส่วนควบคุมการผ่านที่สร้างขึ้นมากับคอมพิวเตอร์

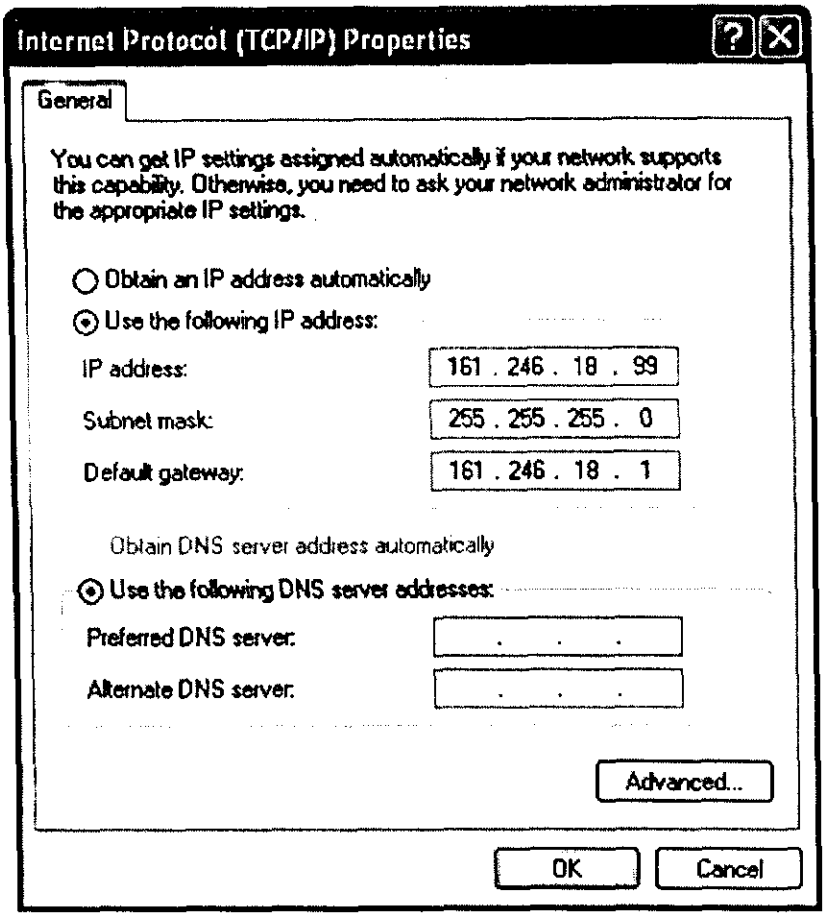
ขั้นตอนการทดลอง

1. เชื่อมต่อสายแลนเข้ากับพอร์ตอีเธอร์เน็ตของโมดูลแร็บบิต ในส่วนควบคุมการผ่านและคอมพิวเตอร์ พร้อมทั้งจ่ายไฟให้กับส่วนควบคุมการผ่าน



รูปที่ 4.7 : แสดงความเร็วที่ได้จากการเชื่อมต่อ

2. ตั้งค่าไอพีแอดเดรสของคอมพิวเตอร์เป็น 161.246.18.99 และตั้งค่าเน็ตมาส์ค เป็น 255.255.255.0

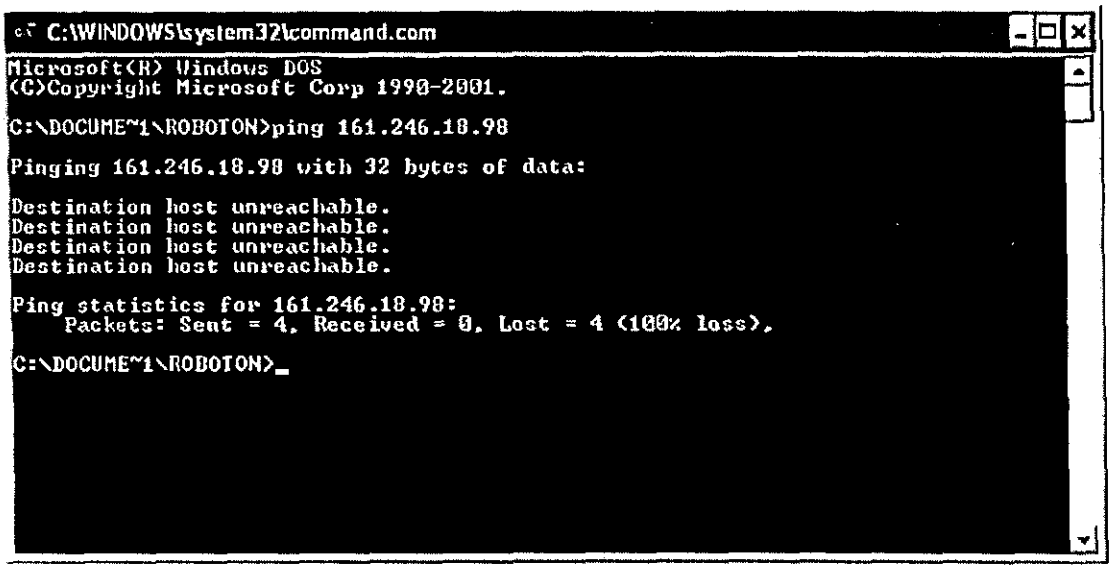


รูปที่ 4.8 : แสดงการตั้งค่าไอพีแอดเดรสและเน็ตมาส์ค ของคอมพิวเตอร์

3. เปิดโปรแกรม Command Prompt และพิมพ์คำสั่ง ping 161.246.18.98 สังเกตผลที่ได้

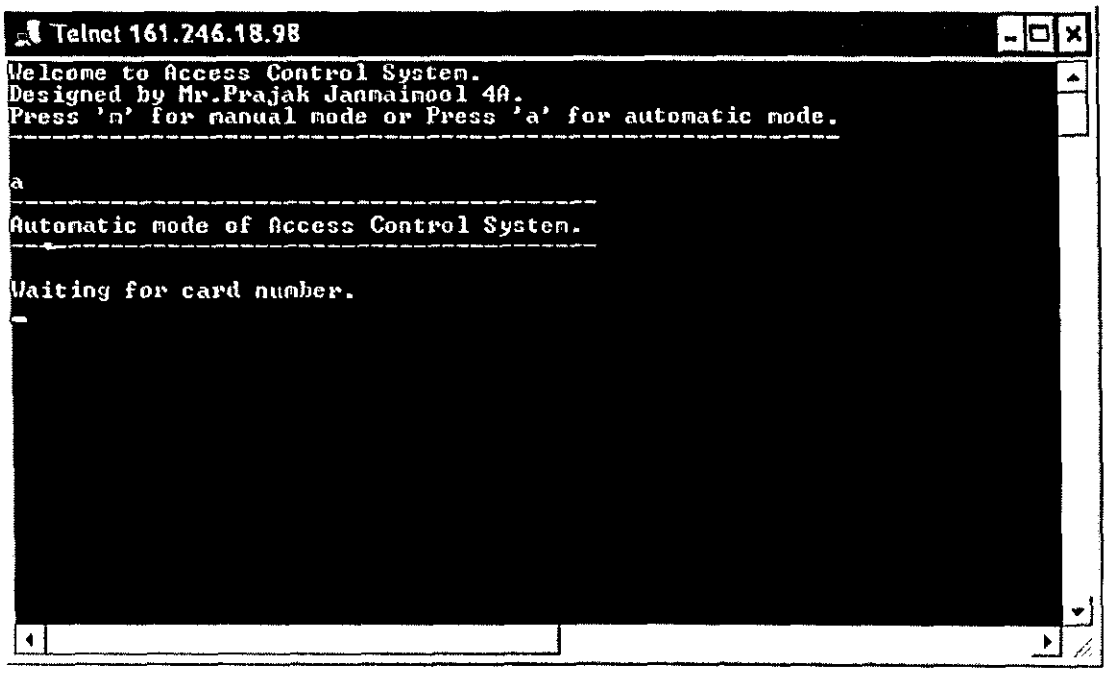
รูปที่ 4.9 : ผลที่ได้จากคำสั่ง ping

4. เปลี่ยนไอพีแอดเดรสของคอมพิวเตอร์เป็น 161.246.19.99 แล้วพิมพ์คำสั่ง ping 161.246.18.98



รูปที่ 4.10 : ผลที่ได้จากคำสั่ง ping หลังจากเปลี่ยนไอพีแอดเดรส

5. เปลี่ยนไอพีแอดเดรสของคอมพิวเตอร์กลับมาเป็น 161.246.18.99 แล้วพิมพ์คำสั่ง telnet 161.246.18.98 ในหน้าต่าง Command Prompt สังเกตผลที่ได้



รูปที่ 4.11 : ผลที่ได้จากการเชื่อมต่อโดยเทลเน็ต

ผลการทดลอง

ผลที่ได้จากการทดลองข้างต้นแสดงให้เห็นว่าสามารถเชื่อมต่อส่วนควบคุมการผ่านเข้ากับคอมพิวเตอร์เป็นเครือข่ายท้องถิ่นได้ ที่ความเร็ว 10 Mbps โดยสังเกตผลที่ได้จากคำสั่ง ping แสดงให้เห็นว่าสามารถเชื่อมต่อและรับส่งแพ็กเก็ตถึงกันได้ แต่เมื่อเปลี่ยนไอพีแอดเดรส ของคอมพิวเตอร์เป็น 161.246.19.99 กลับไม่สามารถเชื่อมกับส่วนควบคุมการผ่านได้เนื่องจากหมายเลขเครือข่ายของคอมพิวเตอร์ไม่ตรงกับหมายเลขเครือข่ายของส่วนควบคุมการผ่านซึ่งถือว่าอยู่คนละเครือข่ายกัน และเมื่อทดลองเชื่อมต่อโดยใช้เทลเน็ทผลปรากฏว่าสามารถเชื่อมต่อเข้าไปที่ส่วนเริ่มต้นของกระบวนการควบคุมการผ่านเข้าออกได้ แต่ไม่สามารถทำอะไรต่อไปได้นอกจากการเลือกโหมดการทำงานเพราะหลังจากนั้นโปรแกรมจะอยู่ในกระบวนการรอรับหมายเลขบัตรจากเครื่องอ่านบัตรแม่เหล็ก

4.3 ทดลองการทำงานของระบบควบคุมการผ่านเข้าออกในโหมดการทำงาน Manual

เชื่อมต่อกลอนประตูไฟฟ้าดรอปโบลท์และเครื่องอ่านบัตรแม่เหล็กรวมทั้งบัซเซอร์เข้ากับส่วนควบคุมการผ่านเพื่อทดสอบการทำงานของระบบรวม

อุปกรณ์ที่ใช้ในการทดลอง

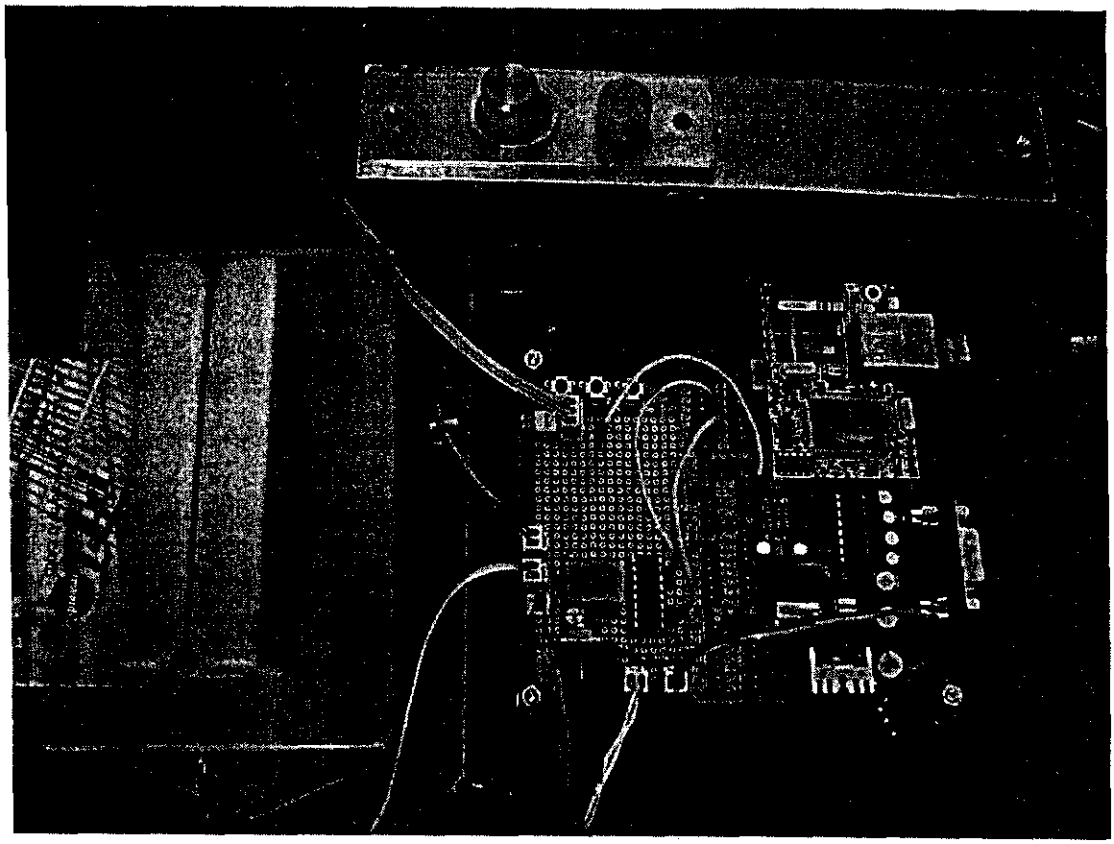
1. ส่วนควบคุมการผ่าน
2. เครื่องอ่านบัตรแม่เหล็กและบัตรแม่เหล็ก
3. กลอนประตูไฟฟ้าชนิดดรอปโบลท์และบัซเซอร์
4. คอมพิวเตอร์ควบคุม
5. สายแลนแบบไขว้

จุดประสงค์การทดลอง

1. เพื่อทดสอบการทำงานของระบบควบคุมการผ่านเข้าออกในโหมด Manual
2. หาจุดอ่อนของระบบเพื่อเป็นแนวทางในการพัฒนาต่อไป

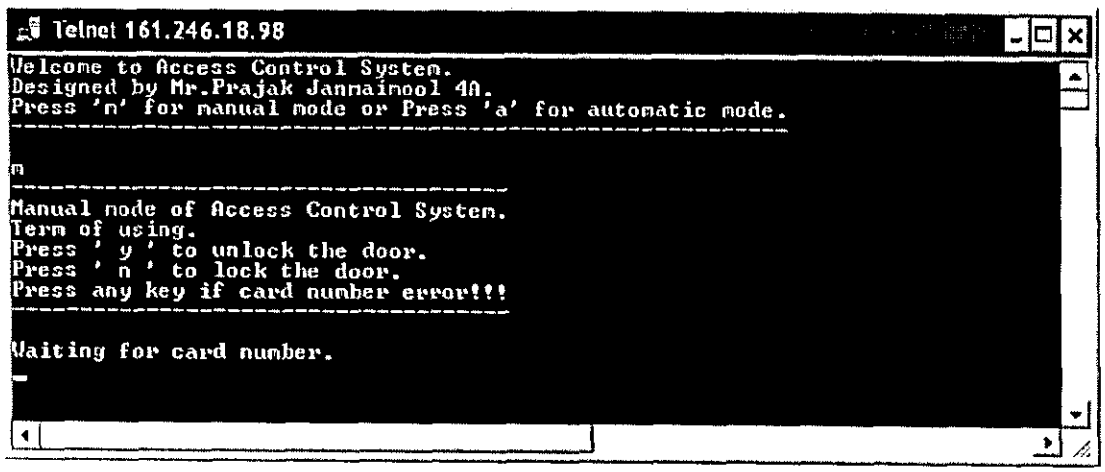
ขั้นตอนการทดลอง

1. เชื่อมต่อเครื่องอ่านบัตรแม่เหล็กและกลอนประตูไฟฟ้าดรอปโบลท์เข้ากับส่วนควบคุมการผ่าน เชื่อมต่อสายแลนระหว่างส่วนควบคุมการผ่านกับคอมพิวเตอร์ควบคุม กำหนดไอพีแอดเดรสของคอมพิวเตอร์ควบคุมเป็น 161.246.18.99 กำหนดเน็ตมาส์คเป็น 255.255.255.0 พร้อมทั้งจ่ายไฟให้กับส่วนควบคุมการผ่าน



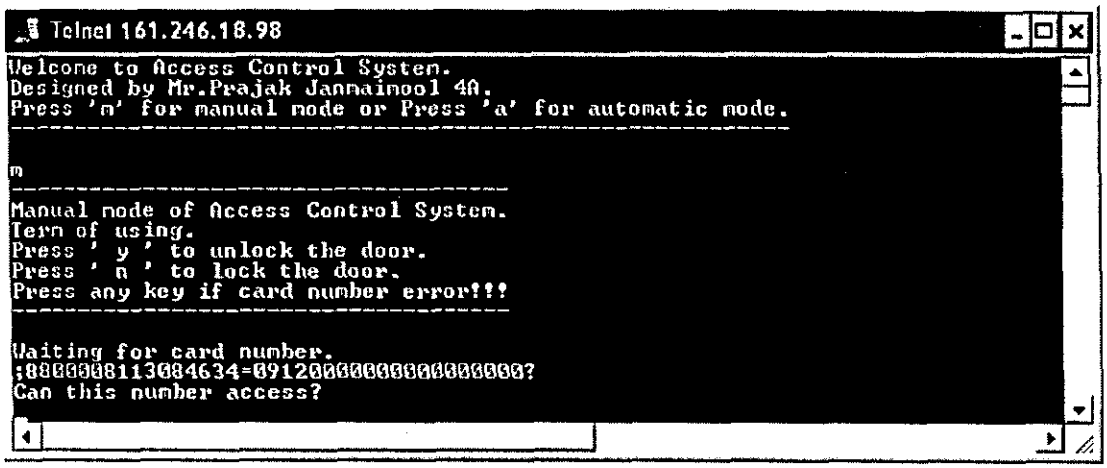
รูปที่ 4.12 : แสดงการเชื่อมต่อระบบรวม

2. เปิดโปรแกรม Command Prompt และพิมพ์คำสั่ง telnet 161.246.18.98 จากนั้นกดคีย์ "m" เพื่อเข้าสู่โหมดการทำงานแบบ Manual



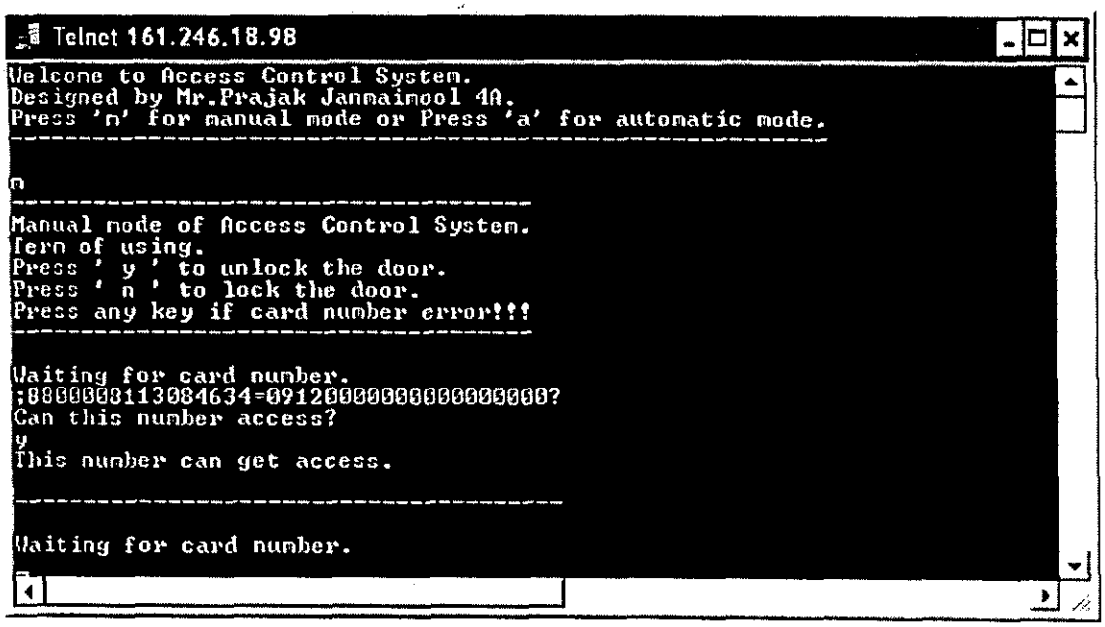
รูปที่ 4.13 : แสดงผลที่ได้จากการเชื่อมต่อโดยเทลเน็ตและเข้าสู่โหมด Manual

3. นำบัตรแม่เหล็กที่เตรียมไว้มารูดผ่านเครื่องอ่านบัตรแม่เหล็ก สังเกตผลที่ได้ในหน้าต่างโปรแกรมเทลเน็ต

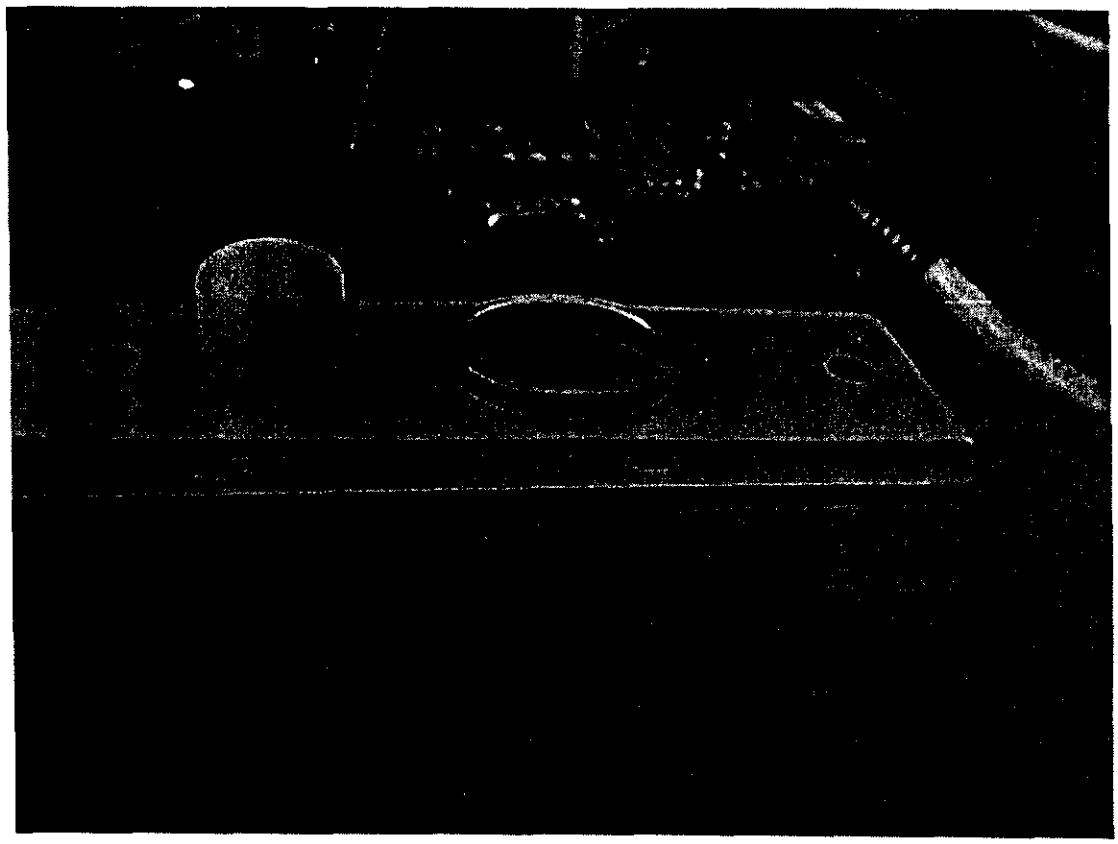


รูปที่ 4.14 : แสดงผลที่ได้หลังจากรูดบัตรแม่เหล็ก

4. พิมพ์ "y" ในหน้าต่างเทลเน็ตเพื่อตอบกลับไปให้ส่วนควบคุมการผ่านรับทราบว่าอนุญาตให้หมายเลขที่ส่งมานี้สามารถผ่านได้ สังเกตหน้าต่างโปรแกรมเทลเน็ตและกลอนประตูไฟฟ้า

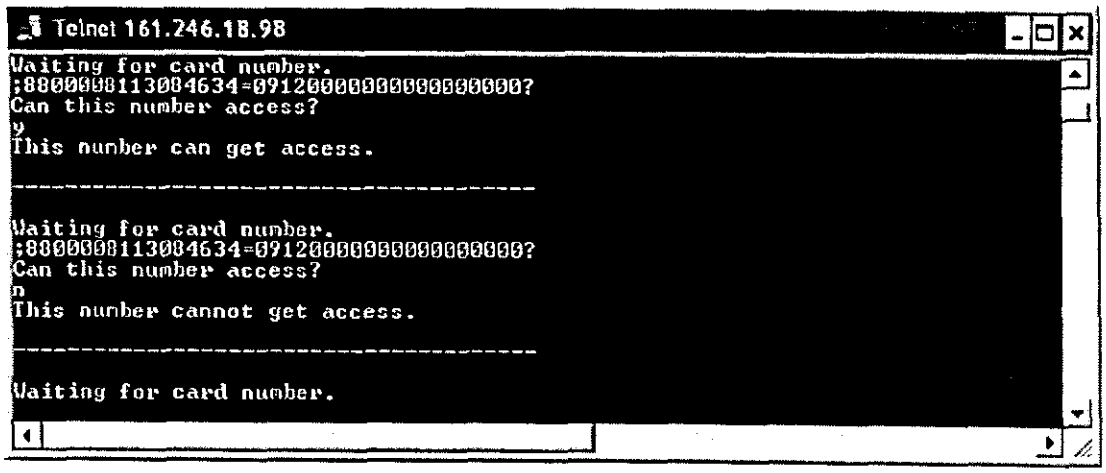


รูปที่ 4.15.1 : แสดงผลที่ได้จากการตอบ "y"

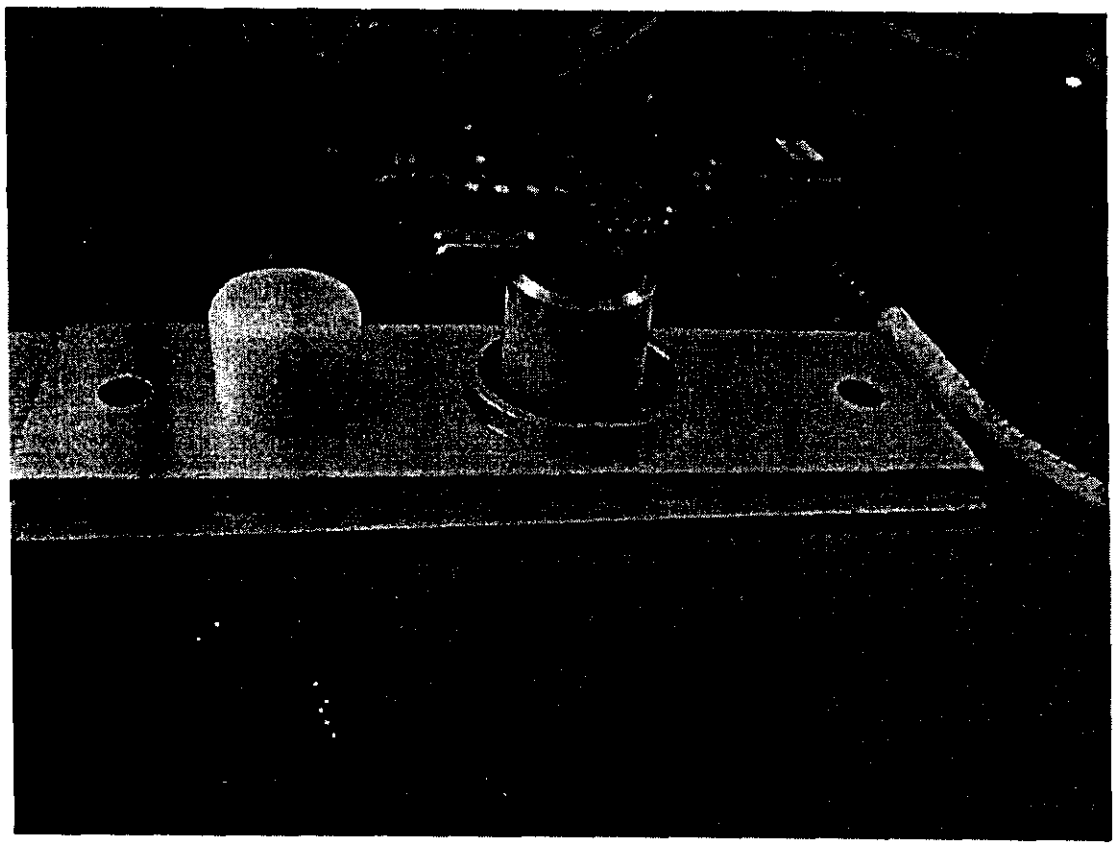


รูปที่ 4.15.2 : แสดงผลที่ได้จากการตอบ "y"

5. ทำตามขั้นตอนที่ 3. อีกครั้งแล้วพิมพ์ "n" ในหน้าต่างเทลเน็ตเพื่อตอบกลับไปให้ส่วนควบคุมการผ่านรับทราบว่าไม่อนุญาตให้หมายเลขบัตรที่ส่งมานี้ผ่าน สังเกตหน้าต่างโปรแกรมเทลเน็ตและกลอนประตูไฟฟ้า

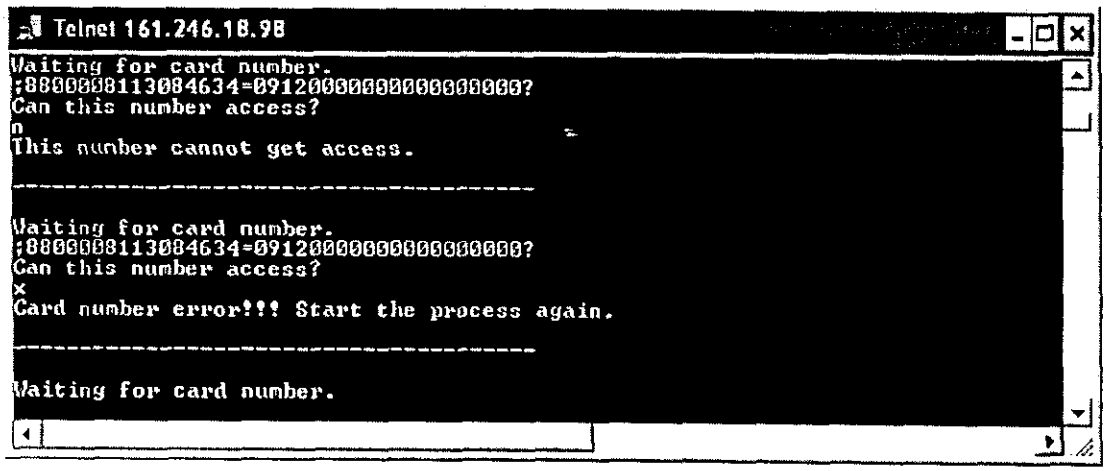


รูปที่ 4.16.1 : แสดงผลที่ได้จากการตอบ "n"



รูปที่ 4.16.2 : แสดงผลที่ได้จากการตอบ "n"

6. ทำตามขั้นตอนที่ 3. อีกครั้งแล้วพิมพ์ตัวอักษรใดๆก็ได้ในหน้าต่างเทลเน็ต 1 ตัวอักษร เพื่อตอบกลับไปให้ส่วนควบคุมการผ่านทำงานต่อไป สังเกตหน้าต่างโปรแกรมเทลเน็ตและกลอนประตูไฟฟ้า



รูปที่ 4.17 : แสดงผลที่ได้จากการตอบ "x"

ผลการทดลอง

จากผลการทดลองที่ได้จะเห็นได้ว่าระบบทำงานตามแผนผังการทำงานที่เขียนโปรแกรมให้โมดูลแร็บบิต คือส่วนควบคุมการผ่านที่ประกอบด้วยโมดูลแร็บบิตจะรอจนกว่าจะมีหมายเลขบัตรเข้ามาจึงจะดำเนินการต่อไป เมื่อมีหมายเลขบัตรเข้ามาส่วนควบคุมการผ่านก็จะส่งหมายเลขบัตรนั้นไปให้คอมพิวเตอร์ผ่านเครือข่ายท้องถิ่นและไปแสดงผลที่หน้าต่างโปรแกรมเทลเน็ทที่ใช้เชื่อมต่อกับส่วนควบคุมการผ่าน หลังจากนั้นส่วนควบคุมการผ่านก็จะรอรับการตอบกลับมาเทลเน็ต ถ้าเทลเน็ตตอบ "y" กลับไปให้ ส่วนควบคุมการผ่านก็จะส่งสัญญาณไปปลดล็อคกลอนประตูไฟฟ้าพร้อมทั้งส่งข้อความกลับไปบอกเทลเน็ตว่าหมายเลขบัตรนี้สามารถผ่านได้ และรอรับหมายเลขบัตรต่อไป แต่ถ้าเทลเน็ตตอบ "n" กลับไปให้ ส่วนควบคุมการผ่านก็จะไม่ปลดล็อคกลอนประตูไฟฟ้าพร้อมทั้งส่งข้อความกลับไปบอกเทลเน็ตว่า หมายเลขบัตรนี้ไม่สามารถผ่านได้ และรอรับหมายเลขบัตรต่อไป และในกรณีที่เทลเน็ตตอบอักษรใดๆกลับไปให้ ส่วนควบคุมก็จะไม่ปลดล็อคกลอนประตูไฟฟ้าพร้อมทั้งส่งข้อความกลับไปบอกเทลเน็ตว่า หมายเลขบัตรผิดพลาดให้เริ่มต้นกระบวนการใหม่ แล้วส่วนควบคุมก็รอรับหมายเลขบัตรต่อไป

4.4 ทดลองการทำงานของระบบควบคุมการผ่านเข้าออกในโหมดการทำงาน Automatic

เชื่อมต่อกลอนประตูไฟฟ้าดรอปโบลท์และเครื่องอ่านบัตรแม่เหล็กรวมทั้งบัซเซอร์เข้ากับส่วนควบคุมการผ่านเพื่อทดสอบการทำงานของระบบรวม

อุปกรณ์ที่ใช้ในการทดลอง

1. ส่วนควบคุมการผ่าน
2. เครื่องอ่านบัตรแม่เหล็กและบัตรแม่เหล็ก
3. กลอนประตูไฟฟ้าชนิดดรอปโบลท์และบัซเซอร์
4. คอมพิวเตอร์ควบคุม
5. สายแลนแบบไขว้

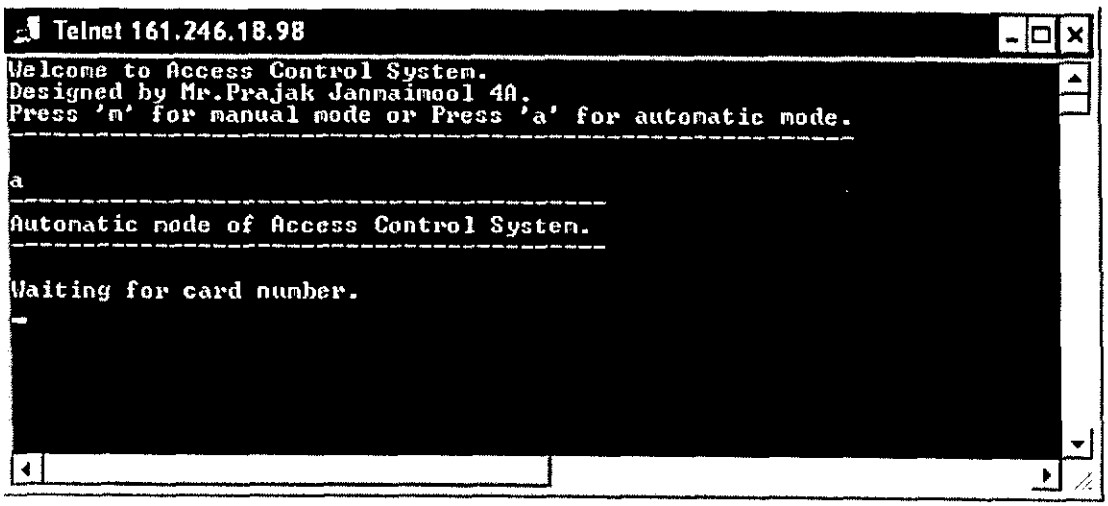
จุดประสงค์การทดลอง

1. เพื่อทดสอบการทำงานของระบบควบคุมการผ่านเข้าออกในโหมด Automatic
2. หาจุดอ่อนของระบบเพื่อเป็นแนวทางในการพัฒนาต่อไป

ขั้นตอนการทดลอง

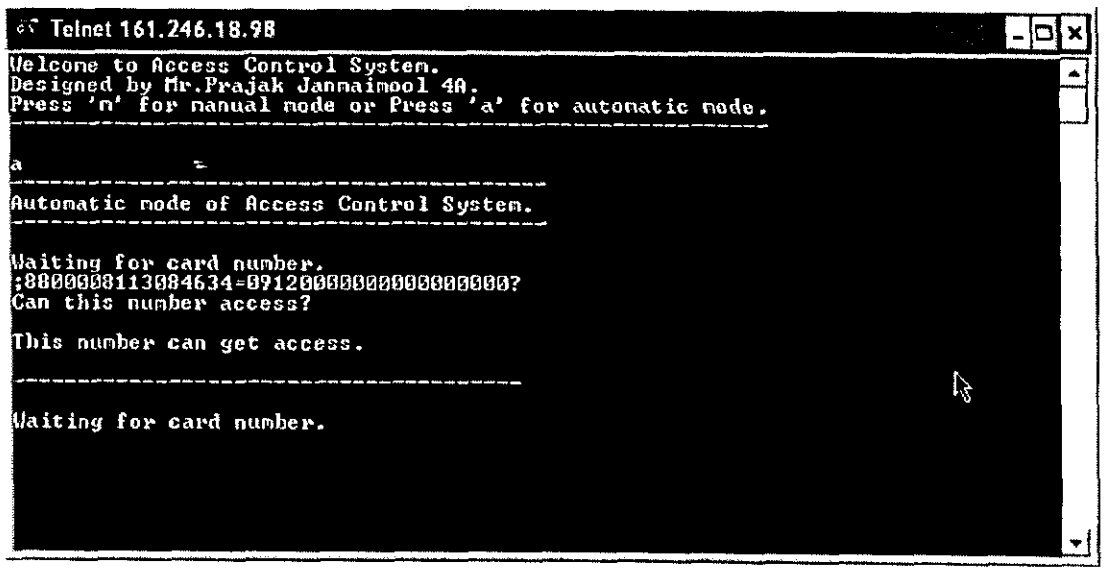
1. เชื่อมต่อเครื่องอ่านบัตรแม่เหล็กและกลอนประตูไฟฟ้าดรอปโบลท์เข้ากับส่วนควบคุมการผ่าน เชื่อมต่อสายแลนระหว่างส่วนควบคุมการผ่านกับคอมพิวเตอร์ควบคุม กำหนดไอพีแอดเดรสของคอมพิวเตอร์ควบคุมเป็น 161.246.18.99 กำหนดเน็ตมาส์คเป็น 255.255.255.0 พร้อมทั้งจ่ายไฟให้กับส่วนควบคุมการผ่าน

2. เปิดโปรแกรม Command Prompt และพิมพ์คำสั่ง telnet 161.246.18.98 จากนั้นกดคีย์ "a" เพื่อเข้าสู่โหมดการทำงานแบบ Automatic



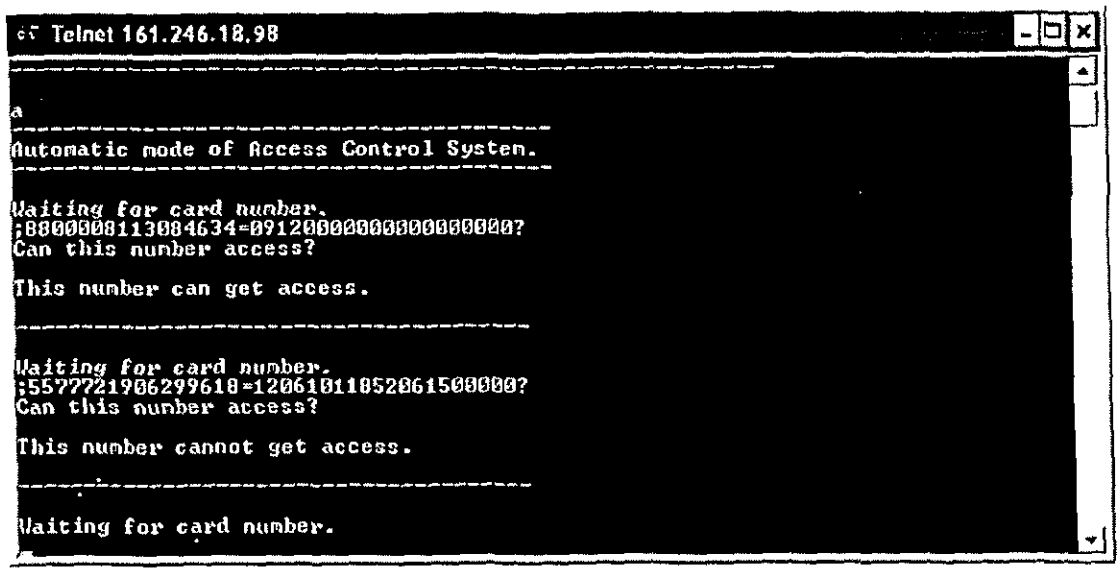
รูปที่ 4.18 : แสดงผลที่ได้จากการเชื่อมต่อโดยเทลเน็ตและเข้าสู่โหมด Automatic

3. นำบัตรแม่เหล็กที่มีหมายเลขบัตรอยู่ในรายการหมายเลขบัตรที่ได้รับอนุญาตให้ผ่านได้ มารูดผ่านเครื่องอ่านบัตรแม่เหล็ก สังเกตผลที่ได้ในหน้าต่างโปรแกรมเทลเน็ต



รูปที่ 4.19 : แสดงผลที่ได้หลังจากรูดบัตรแม่เหล็ก

4. นำบัตรแม่เหล็กที่ไม่มีหมายเลขบัตรอยู่ในรายการหมายเลขบัตรที่ได้รับอนุญาตให้ผ่านได้ มารูดผ่านเครื่องอ่านบัตรแม่เหล็ก สังเกตผลที่ได้ในหน้าต่างโปรแกรมเทลเน็ต

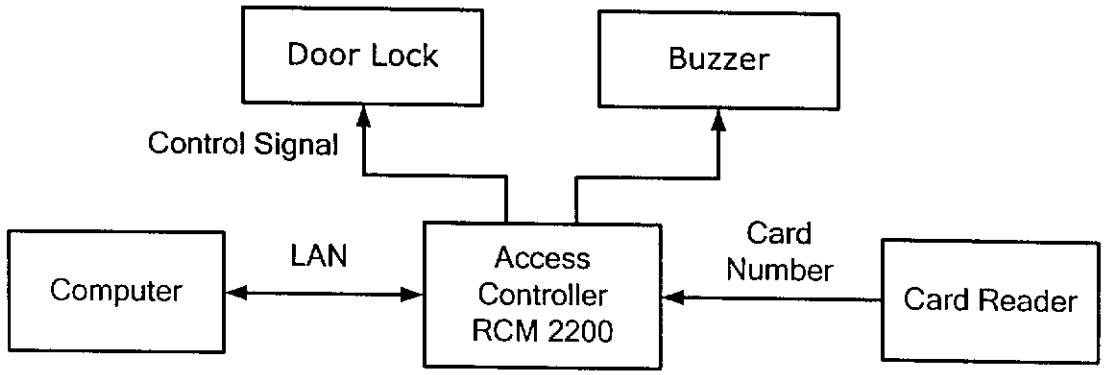


รูปที่ 4.20 : แสดงผลที่ได้หลังจากรูดบัตรแม่เหล็ก

ผลการทดลอง

จากผลการทดลองที่ได้จะเห็นได้ว่าระบบทำงานตามแผนผังการทำงานที่เขียนโปรแกรมให้โมดูลแร็บบิตในโหมดการทำงานแบบ Automatic คือส่วนควบคุมการผ่านที่ประกอบด้วยโมดูลแร็บบิตจะรอจนกว่าจะมีหมายเลขบัตรเข้ามาจึงจะดำเนินการต่อไป เมื่อมีหมายเลขบัตรเข้ามาส่วนควบคุมการผ่านก็จะส่งหมายเลขบัตรนั้นไปให้คอมพิวเตอร์ผ่านเครือข่ายท้องถิ่นและไปแสดงผลที่หน้าต่างโปรแกรมเทลเน็ทที่ใช้เชื่อมต่อกับส่วนควบคุมการผ่าน พร้อมทั้งตรวจสอบหมายเลขบัตรนั้นกับรายการหมายเลขบัตรที่ได้รับอนุญาตให้ผ่านได้ ถ้าหากมีหมายเลขบัตรนั้นในรายการหมายเลขบัตรที่ได้รับอนุญาตให้ผ่านได้ ส่วนควบคุมการผ่านก็จะดำเนินการปลดล็อคกลอนประตูไฟฟ้า แต่ถ้าไม่มีหมายเลขบัตรนั้นในรายการหมายเลขบัตรที่ได้รับอนุญาตให้ผ่านได้ ส่วนควบคุมการผ่านก็จะดำเนินการเปิดบัซเซอร์เพื่อส่งสัญญาณเสียงแจ้งเตือนและไม่ปลดล็อคกลอนประตูไฟฟ้า หลังจากนั้นส่วนควบคุมการผ่านก็จะรอรับหมายเลขบัตรต่อไป

บทที่ 5
วิจารณ์และสรุป



รูปที่ 5.1 : Block Diagram แสดงระบบรวม

จาก Block Diagram แสดงระบบรวม จะเห็นได้ว่าในขณะนี้โครงงานในส่วนต่างๆ ได้สำเร็จลุล่วงไปแล้ว และส่วนกลอนประตูไฟฟ้าชนิดแกนล็อคที่เรียกว่าดรอปโบลท์จะใช้เป็นกลอนประตูไฟฟ้าสำเร็จรูปที่มีจำหน่ายตามท้องตลาด จึงไม่ต้องสร้างชิ้นงานในส่วนนี้เพียงแต่ศึกษาวิธีการทำงานของอุปกรณ์ เพื่อให้นำมาใช้ในโครงงานได้อย่างมีประสิทธิภาพ สำหรับส่วนควบคุมการผ่าน (Access Controller) ซึ่งเป็นส่วนที่สำคัญมากซึ่งทำหน้าที่ประสานการทำงานของส่วนต่างๆเข้าด้วยกันนั้นสร้างขึ้นจากโมดูลแร็บบิต RCM 2200 ซึ่งใช้ชุดคำสั่งภาษาไดนามิคซีในการโปรแกรมการทำงานของโมดูล และในส่วนของคอมพิวเตอร์ควบคุมนั้นจะใช้โปรแกรมเทลเน็ตในการติดต่อและสื่อสารข้อมูลกับโมดูลแร็บบิต

จากผลการทดลองจะเห็นได้ว่าเครื่องอ่านบัตรแม่เหล็กนั้นสามารถทำงานได้อย่างดี และการสื่อสารระหว่างคอมพิวเตอร์ควบคุมกับโมดูลแร็บบิตโดยผ่านเครือข่ายท้องถิ่นอีเธอร์เน็ตก็เป็นไปได้ด้วยดี แต่ระบบควบคุมการผ่านเข้าออกในโครงงานนี้ยังขาดความยืดหยุ่น เนื่องจากในโหมดการทำงานแบบ Manual นั้นจะต้องมีคนคอยตรวจสอบหมายเลขบัตรและกดคีย์อนุญาตที่คอมพิวเตอร์ควบคุม ระบบจึงจะสามารถดำเนินไปได้ ทำให้ดูเหมือนว่าไม่ค่อยจะมีประโยชน์เท่าใดนัก และในโหมดการทำงานแบบ Automatic นั้นรายการหมายเลขบัตรที่ได้รับอนุญาตให้ผ่านได้จะถูกบันทึกให้กับโมดูลแร็บบิตได้เฉพาะในขั้นตอนการโปรแกรมการทำงานให้โมดูลแร็บบิตเท่านั้น ทำให้ยากต่อการปรับเปลี่ยน แต่อย่างไรก็ตามจุดประสงค์หลักของโครงงานก็เพื่อศึกษาและทำความเข้าใจเกี่ยวกับการสื่อสารผ่านเครือข่ายท้องถิ่นอีเธอร์เน็ตโดยใช้โปรโตคอลทีซีพีไอพี และสามารถควบคุมการทำงานของอุปกรณ์โดยผ่านเครือข่ายท้องถิ่นได้

ระบบควบคุมการผ่านเข้าออกที่เสร็จสมบูรณ์สามารถนำมาประยุกต์รวมกับระบบรักษาความปลอดภัยในรูปแบบอื่นๆเป็นเครือข่ายได้ซึ่งเหมาะกับสำนักงานหรือสถานที่ที่ต้องการจำกัดสิทธิในการผ่านเข้าออกในส่วนต่างๆและต้องการ การรักษาความปลอดภัยอย่างเข้มงวด

กล่าวโดยสรุปแล้วโครงงานนี้เป็นการประยุกต์ใช้เครือข่ายท้องถิ่นมาช่วยให้ระบบควบคุมการผ่านเข้าออกทำงานได้มีประสิทธิภาพมากขึ้น สามารถต่อร่วมกันเป็นเครือข่ายได้และในปัจจุบันอุปกรณ์ที่ใช้ในเครือข่ายท้องถิ่นอย่างอีเธอร์เน็ตมีราคาถูกลงมากและได้รับความนิยมอย่างแพร่หลายจึงประจวบเหมาะที่จะนำเทคโนโลยีเครือข่ายท้องถิ่นมาประยุกต์ใช้กับระบบอื่นๆเพื่อให้มีประสิทธิในการทำงานของระบบสูงขึ้น

ปัญหาที่พบ

1. เนื่องจากไมโครโปรเซสเซอร์แร็บบิต 2000 ยังไม่มีการใช้งานอย่างแพร่หลาย ทำให้ขาดตัวอย่างในการศึกษาและเป็นไมโครโปรเซสเซอร์ที่ยังไม่เคยใช้งานมาก่อนจึงใช้งานได้ไม่เต็มความสามารถของไมโครโปรเซสเซอร์
2. ไม่คุ้นเคยกับชุดคำสั่งภาษาไดนามิกส์ซี ทำให้แอพลิเคชันที่พัฒนาออกมาขาดความสมบูรณ์

แนวทางในการพัฒนาต่อไป

1. พัฒนาซอฟท์แวร์ที่ใช้งานเฉพาะสำหรับระบบควบคุมการผ่านเข้าออกที่สร้างขึ้นมา เพื่อตรวจสอบหมายเลขบัตรในรายการที่อนุญาตให้ผ่านได้และส่งคีย์กลับมาให้โมดูลแร็บบิตโดยอัตโนมัติ เพื่อจะได้ไม่ต้องใช้คนคอยกดคีย์ตอบกลับไปให้โมดูลแร็บบิตและสะดวกในการกำหนดรายการหมายเลขบัตรที่ได้รับอนุญาตให้ผ่านได้
2. จัดทำซอฟท์แวร์ฐานข้อมูลจัดเก็บรายการผู้ใช้งานระบบ เพื่อความสะดวกในการตรวจดูย้อนหลังในกรณีที่เกิดปัญหาภายในสถานที่ที่ใช้ระบบควบคุมการผ่านเข้าออก
3. ปรับปรุงโปรแกรมในส่วนของโมดูลแร็บบิตให้มีความสามารถมากขึ้นนอกเหนือจากการติดต่อกับคอมพิวเตอร์ผ่านเครือข่ายท้องถิ่น

บรรณานุกรม

1. จตุชัย แพงจันทร์, อนุโชต วุฒิพรพงษ์, เจาะระบบ Network ฉบับสมบูรณ์, ไอดีซีฯ พิมพ์ครั้งที่ 2 พ.ศ. 2547
2. ภัทรพล ศรีกัลยาณบุตร, รอบรู้เรื่อง Network บนวินโดวส์, บริษัท เอ.อาร์. อินฟอร์เมชัน แอนด์ พับลิเคชัน จำกัด, 2547
3. สมยศ จุณณะปิยะ, การประยุกต์ใช้งานไมโครคอนโทรลเลอร์, สถาบันเทคโนโลยีพระจอมเกล้า-เจ้าคุณทหารลาดกระบัง พิมพ์ครั้งที่ 5 พ.ศ. 2546
4. ประภาพร ช่างไม้, คู่มือการเขียนโปรแกรมภาษา C ฉบับผู้เริ่มต้น, อินโฟเพรส, 2545
5. http://www.zworld.com/
6. http://www.rabbitsemiconductor.com/

ภาคผนวก

ชุดคำสั่งที่ใช้ควบคุมการทำงานของไมโครคอนโทรลเลอร์ในวงจรเครื่องอ่านบัตรแม่เหล็กมีดังนี้

```
;*โปรแกรมเครื่องอ่านบัตรแม่เหล็ก*
DATA        BIT     P1.0        ; ข้อมูลที่อ่านได้
PRESENT     BIT     P1.1        ; สัญญาณควบคุมการอ่าน
CLOCK       BIT     P1.2        ; สัญญาณนาฬิกาของเครื่องอ่าน
            ORG     20H
FLAG:       DS      1           ; กำหนด Flag Buffer
CHECKSUM:   DS      1
BUFFER:     DS      40          ; กำหนดพื้นที่บัฟเฟอร์ 40 Byte
START       BIT     FLAG.0      ; บิตแสดงสถานะเริ่มต้น
STOP        BIT     FLAG.1      ; บิตแสดงสถานะหยุด
ERROR       BIT     FLAG.2      ; บิตแสดงสถานะผิดพลาด
PARITY      BIT     FLAG.3      ; บิตแสดงพาริตี้ผิดพลาด
            ORG     0000H
PROGRAM:    MOV     SP, #128-32 ; กำหนดพื้นที่ Stack 32 Byte
            MOV     FLAG, #0    ; เคลียร์ Flag
            SETB    DATA
            SETB    PRESENT
            SETB    CLOCK
SETSERIAL:  MOV     A, #0FDH    ; กำหนดบอดเรตเท่ากับ 9600 bps
            MOV     TH1, A
            MOV     TL1, A
            MOV     TMOD, #00100000B    ; TIMER1 MODE 1
            CLR     ES          ; ไม่ใช้ serial interrupt
            CLR     ET1         ; ไม่ใช้ timer1 interrupt
            SETB    TR1         ; timer1 เริ่มทำงาน
            MOV     SCON, #01010000B    ; ใช้ Serial mode 1
PROMPT:     LCALL   PRINT
            DB      'SLIDE MAGNETIC CARD', 0DH, 0AH, 00
PROCESS0:   LCALL   READ        ; เริ่มอ่านข้อมูลใน Track2
            MOV     R0, #BUFFER
```

```
                JB      ERROR, SERROR
                JB      PARITY, PERROR
                SJMP    PROCESS1
SERROR:         LCALL   PRINT           ; แจ้งเตือน Checksum Error
                DB      'Checksum Error!', 0DH,0AH,00
                CLR     ERROR
                SJMP    PROCESS0        ; กลับไปเริ่มกระบวนการอ่านข้อมูล
PERROR:         LCALL   PRINT           ; แจ้งเตือน Parity Error
                DB      'Parity Error!', 0DH,0AH,00
                CLR     PARITY
                SJMP    PROCESS0        ; กลับไปเริ่มกระบวนการอ่านข้อมูล
PROCESS1:       MOV     A,@R0           ; นำค่าในบัฟเฟอร์มาไว้ที่ A
                CJNE    A, #0FH, PROCESS2 ; ถ้ายังไม่สุดเฟรมให้ไปทำ PROCESS2
                ADD     A, #30H         ; แปลงเป็น ascii code
                LCALL   SENDDATA        ; ส่งข้อมูลออกทางพอร์ตอนุกรม
                SJMP    PROCESS3
PROCESS2:       ADD     A, #30H         ; แปลงเป็น ascii code
                LCALL   SENDDATA        ; ส่งข้อมูลออกทางพอร์ตอนุกรม
                INC     R0              ; เลื่อนไปไบท์ถัดไป
                SJMP    PROCESS1        ; กลับไปเก็บข้อมูลต่อ
PROCESS3:       MOV     A, #0DH         ; Field Separator
                LCALL   SENDDATA        ; ส่งข้อมูลออกทางพอร์ตอนุกรม
                MOV     A, #0AH         ; Control
                LCALL   SENDDATA        ; ส่งข้อมูลออกทางพอร์ตอนุกรม
                SJMP    PROCESS0        ; กลับไปเริ่มกระบวนการอ่านข้อมูล

;* เคลียร์บัฟเฟอร์ข้อมูล *
CLEAN0:         PUSH    B               ; นำค่าของ B ไปเก็บไว้ใน stack
                MOV     B, #40          ; จำนวนบัฟเฟอร์ที่จะเคลียร์
                MOV     R0, #BUFFER     ; R0 ชี้ไปที่บัฟเฟอร์ข้อมูล
CLEAN1:         CLR     A               ; A มีค่าเป็น 00H
                MOV     @R0, A          ; เคลียร์บัฟเฟอร์ข้อมูล
                INC     R0              ; R0 ชี้ตำแหน่งถัดไป
                DJNZ    B, CLEAN1       ; เคลียร์ข้อมูลจนกว่าจะครบ 40 ไบท์
```

```
        POP     B               ; นำค่าใน stack มาเก็บไว้ใน B
        RET                     ; กลับไปทำงานโปรแกรมหลัก

;* อ่านข้อมูลจากบัตรแม่เหล็ก *
READ:   LCALL   CLEAN0          ; เคลียร์บัฟเฟอร์
        MOV     CHECKSUM,#0     ; เคลียร์ checksum
        MOV     R1, #BUFFER-1   ; ชี้ไปยังข้อมูลที่บันทึก
        CLR     START           ; เคลียร์ start บิต
        CLR     STOP            ; เคลียร์ stop บิต
        CLR     ERROR           ; เคลียร์ error บิต
READ0:  LCALL   GET_CLK         ; จังหวะที่จะเก็บข้อมูล
        MOV     C, DATA         ; นำข้อมูลมาเก็บใน C
        CPL     C               ; กลับค่า C จะได้ข้อมูลจริง
        JB      DATA, READ0     ; วนลูปรอข้อมูล
        SETB    START           ; เริ่มอ่านข้อมูล
READ1:  MOV     R2, #5          ; อ่านข้อมูลทีละ 5 บิต
        CLR     A
        CLR     PARITY
        INC     R1              ; ชี้ไปยังข้อมูลถัดไป
READ2:  JB      START, READ3
        LCALL   GET_CLK         ; จังหวะที่จะเก็บข้อมูล
        MOV     C, DATA         ; นำข้อมูลมาเก็บใน C
        CPL     C               ; กลับค่า C จะได้ข้อมูลจริง
READ3:  RRC     A               ; เลื่อนบิตข้อมูลที่อ่านได้มาไว้ที่ A
        CLR     START
        DJNZ    R2, READ2       ; เก็บข้อมูลบิตถัดไปจนครบ 5 บิต
        RR      A
        RR      A
        RR      A
        JB      P, READ4        ; พาริตี้ดีผ่าน
        SETB    PARITY          ; พาริตี้ผิดพลาด
        SJMP    READ8
READ4:  ANL     A, #0FH         ; ไม่สนใจพาริตี้
        JB      STOP, READ5
```

```
                PUSH    ACC
                XRL     A, CHECKSUM         ; หาค่า Checksum จากข้อมูล
                MOV     CHECKSUM, A
                POP     ACC
READ5:          MOV     @R1, A              ; บันทึกข้อมูล
                JB      STOP, READ6         ; ไปทำส่วนถัดไปถ้าเก็บข้อมูลเสร็จแล้ว
                CJNE    A, #0FH, READ1      ; ถ้าเก็บข้อมูลยังไม่เสร็จให้ไปเก็บต่อ
                SETB    STOP
                SJMP    READ1
READ6:          CJNE    A, CHECKSUM, READ7
                CLR     ERROR               ; รหัส Checksum ถูกต้อง
                SJMP    READ8
READ7:          SETB    ERROR               ; รหัส Checksum ผิดพลาด
READ8:          RET

;* สัญญาณนาฬิกาของเครื่องอ่านบัตรแม่เหล็ก *
GET_CLK:        JB      CLOCK, $            ; รอสัญญาณขอบขาลง
                NOP
                NOP
                JNB     CLOCK, $            ; รอสัญญาณขอบขาขึ้น
                NOP
                NOP
                RET

;* ส่งข้อมูลออกทางพอร์ตอนุกรม *
SENDDATA:       CLR     TI          ; เคลียร์บิต TI เพื่อเริ่มส่งข้อมูล
                MOV     SBUF, A     ; นำค่าใน A ส่งออกพอร์ตอนุกรม
                JNB     TI, $       ; วนลูปรอจนกว่าจะส่งข้อมูลเสร็จ
                RET

;* ส่งข้อความออกทางพอร์ตอนุกรม*
PRINT:          POP     DPH
                POP     DPL
PRINT1:         CLR     A
```

```
        MOVC    A, @A+DPTR
        CJNE    A, #00H, PRINT2
        SJMP    PRINT3
PRINT2: LCALL   SENDDATA
        INC     DPTR
        SJMP    PRINT1
PRINT3: PUSH    DPL
        PUSH    DPH
        RET
        END
```

ชุดคำสั่งที่ใช้ควบคุมการทำงานของโมดูลแร็บบิตให้เป็นส่วนควบคุมการผ่าน

```
#define MY_IP_ADDRESS "161.246.18.98" // ส่วนกำหนดไอพีแอดเดรส
#define MY_NETMASK "255.255.255.0"    // ส่วนกำหนดเน็ตมาส์ค
#define MY_GATEWAY "161.246.18.1"     // ส่วนกำหนดเกตเวย์

#memmap xmem
#use "dcrtcp.lib"  // กำหนดไลบรารี่ที่จะใช้งาน

#define PORT 23  // กำหนดหมายเลขพอร์ตในการเชื่อมต่อเป็น 23
#define DINBUFSIZE 63  // กำหนดขนาดบัฟเฟอร์รับข้อมูลของซีเรียลพอร์ต D
#define DOUTBUFSIZE 63  // กำหนดขนาดบัฟเฟอร์ส่งข้อมูลของซีเรียลพอร์ต D
tcp_Socket mysock;  // ใช้ซ็อคเก็ตการเชื่อมต่อเป็นแบบทีซีพี

void initial_port() // ตั้งค่าพอร์ตเริ่มต้น
{
        // กลอนประตูไฟฟ้าต่ออยู่ที่ PA0, buzzer ต่ออยู่ที่ PA1
        // กลอนล็อคเมื่อ PA0 = 1 และปลดล็อคเมื่อ PA0 = 0
        // Buzzer เปิดเมื่อ PA1 = 1 และปิดเมื่อ PA1 = 0
        WrPortI( SPCR, & SPCRShadow, 0x84 );  // ตั้งค่าให้พอร์ต A เป็น output
        WrPortI( PADR, NULL, 0x01 );  // ล็อคกลอนและปิด Buzzer
        serDopen( 9600 );  // เริ่มการเชื่อมต่อทาง Serial Port D ที่ baud rate 9600
        sock_init();  // initial data structure และ enable ethernet chip
}

void open()  // ปลดล็อคกลอนและปิดบัซเซอร์
{
        int j;
        sock_puts( &mysock," " );  // ขึ้นบรรทัดใหม่ในหน้าต่าง telnet
        sock_puts( &mysock,"This number can get access." );  // แสดงข้อความ
        WrPortI( PADR, NULL, 0x00 );  // ปลดล็อคกลอนและปิด buzzer
        for( j=0; j<20000; j++ );  // time delay loop
        WrPortI( PADR, NULL, 0x01 );  // ล็อคกลอนและปิด buzzer
}
```

```
void close() // ล็อคกลอนและปิดบัชเซอร์
{
        int j;
        sock_puts( &mysock," " ); // ขึ้นบรรทัดใหม่ในหน้าต่าง telnet
        sock_puts( &mysock,"This number cannot get access." ); // แสดงข้อความ
        WrPortI( PADR, NULL, 0x03 ); // ล็อคกลอนและเปิด buzzer
        for( j=0; j<20000; j++ ); // time delay loop
        WrPortI(PADR, NULL, 0x01); // ล็อคกลอนและปิด Buzzer
}

void newline() // แสดงที่คั่นของการ Access แต่ละครั้ง
{
        tcp_tick(&mysock); // ประมวลผล Packetที่เข้ามาใน Socket
        sock_puts(&mysock," "); // ขึ้นบรรทัดใหม่ในหน้าต่าง telnet
        sock_puts(&mysock,"----------------------------------------"); // แสดงที่คั่น
        sock_puts(&mysock," "); // ขึ้นบรรทัดใหม่ในหน้าต่าง telnet
}

automatic() // โหมดการทำงานแบบ Automatic
{
        char buffer[41]; // buffer รับข้อมูลมีขนาด 41 ไบท์
        int n, i, t, status, a, b, c ;
        // รายการหมายเลขบัตรที่อนุญาตให้ผ่านได้
        const char dbasea[40] = ";57209000505056106304810130008100711911?";
        const char dbaseb[40] = ";5577721930094167=13011011125262600000?";
        const char dbasec[40] = ";8800008113084634=09120000000000000000?";
        i = 0;
        t = 0;

        tcp_tick(&mysock);
        sock_puts( &mysock,"----------------------------------------" );
        sock_puts( &mysock,"Automatic mode of Access Control System." );
        sock_puts( &mysock,"----------------------------------------" );
        sock_puts( &mysock," " );
```

```
while(tcp_tick(&mysock)) // ประมวลผล Packetที่เข้ามาใน Socket
{
        if (t==0)
        {
                sock_puts(&mysock,"Waiting for card number."); // แสดงข้อความ
                t=1;
        }

        while (tcp_tick(&mysock))
        {
                if ( (n = serDread(buffer, 41, 300)) > 0 ) // n คือจำนวนไบท์ที่อ่านได้
                break; // เมื่อได้รับข้อมูลแล้วให้ออกจากลูป
        }

        i = 0;
        while( i < n ) // วนลูปส่งข้อมูล n ไบท์ไปให้ telnet
        {
                sock_putc(&mysock,buffer[i]);  // ส่งข้อมูลไปให้ telnet
                i++;                           // ชี้ไปยังข้อมูลตัวต่อไป
        }

        sock_puts( &mysock,"Can this number access?" ); // แสดงข้อความ
        a = 0;
        b = 0;
        c = 0;
        for ( i=0; i<40; i++ ) // วนลูปตรวจสอบหมายเลขบัตร
        {
                if (buffer[i] == dbasea[i] )
                a++;
                if (buffer[i] == dbaseb[i] )
                b++;
                if (buffer[i] == dbasec[i] )
                c++;
        }
```

```
            if ( (a >= 39 ) || (b >= 39) || (c >= 39) )
            open(); // ปลดล็อคกลอนและปิดบัชเซอร์
            else
            close(); // ล็อคกลอนและเปิดบัชเซอร์
            newline(); // ขึ้นบรรทัดใหม่ในหน้าต่าง telnet
            t = 0; // เพื่อให้แสดงข้อความเริ่มต้นการ Access
            i = 0;
        }
}

manual() // โหมดการทำงานแบบ Manual
{
        char buffer[41];  // Buffer รับข้อมูลมีขนาด 41 ไบท์
        char ans[2]; // ตัวแปรรอรับคำตอบ
        int n, i, t, status;
        i = 0;
        t = 0;
        tcp_tick(&mysock);
        sock_puts( &mysock,"------------------------------------" );
        sock_puts( &mysock,"Manual mode of Access Control System." );
        sock_puts( &mysock,"Term of using." );
        sock_puts( &mysock,"Press ' y ' to unlock the door." );
        sock_puts( &mysock,"Press ' n ' to lock the door." );
        sock_puts( &mysock,"Press any key if card number error!!!" );
        sock_puts( &mysock,"------------------------------------" );
        sock_puts( &mysock," " );

        while(tcp_tick(&mysock)) // ประมวลผล Packet ที่เข้ามาใน Socket
        {
            if (t==0)
            {
                sock_puts(&mysock,"Waiting for card number."); // แสดงข้อความ
                t=1;
            }
```

```
while (tcp_tick(&mysock)) // ประมวลผล Packet ที่เข้ามาใน Socket
{
        if ( (n = serDread(buffer, 41, 300)) > 0 ) // n เท่ากับจำนวนไบท์ที่อ่านได้
        break; // เมื่อได้รับข้อมูลให้ออกจากลูป
}

i = 0;
while(i < n) // วนลูปส่งข้อมูล n ไบท์ไปให้ telnet
{
        sock_putc(&mysock,buffer[i]); // ส่งข้อมูลไปให้ telnet
        i++;                          // ชี้ไปยังข้อมูลตัวถัดไป
}
sock_puts( &mysock,"Can this number access?" ); // แสดงข้อความ
while( tcp_tick(&mysock) ) // ประมวลผล Packet ที่เข้ามาใน Socket
{
        if( sock_gets(&mysock, ans, 2) ) // รับคำตอบ
        break; // ออกจากลูปเมื่อได้รับคำตอบ
}

if ( (ans[0] == 'y') || (ans == "y") ) // ถ้าได้รับ y
open(); // ปลดล็อคกลอนและปิดบัซเซอร์
else if ( (ans[0] == 'n') || (ans == "n") ) // ถ้าได้รับ n
close(); // ล็อคกลอนและเปิดบัซเซอร์
else // ถ้าได้รับตัวอักษรใดๆ
{
        sock_puts(&mysock," ");
        sock_puts(&mysock,"Card number error!!! Start the process again.");
}
newline(); // แสดงที่คั่นในหน้าต่าง telnet
t = 0; // เพื่อให้แสดงข้อความเริ่มต้นการ Access
i = 0;
    }
}
```

```
main()
{
        char ans[2]; // ตัวแปรรับคำตอบการเลือกโหมด
        int status;
        initial_port(); // ตั้งค่าพอร์ตเริ่มต้น
        while(1)
        {
                //รอการติดต่อมาจาก Telnet Port 23, IP อะไรก็ได้ในเน็ตเวิร์คเดียวกัน
                tcp_listen(&mysock,PORT,0,0,NULL,0);
                //รอจนกว่า Connection จะสำเร็จ
                sock_wait_established(&mysock,0,NULL,&status);
                //กำหนดโหมด TCP เป็น ASCII Mode
                sock_mode(&mysock,TCP_MODE_ASCII);

                // ส่วนแสดงข้อความเริ่มต้นการใช้งานระบบ
                tcp_tick(&mysock); //
                sock_puts( &mysock,"Welcome to Access Control System." );
                sock_puts( &mysock,"Designed by Mr.Prajak Janmaimool 4A." );
                sock_puts( &mysock,"Press 'm' for manual mode or Press 'a' for automatic mode.");
                sock_puts( &mysock,"-------------------------------------------------------" );
                sock_puts( &mysock," " );

                while( tcp_tick(&mysock) ) // วนลูปรอรับคำตอบการเลือกโหมด
                {
                        if( sock_gets(&mysock, ans, 2) ) // รับคำตอบการเลือกโหมด
                        break; // ออกจากลูปเมื่อได้รับคำตอบการเลือกโหมด
                }
                tcp_tick(&mysock);
                sock_puts( &mysock," " );

                if ( (ans[0] == 'm') || (ans == "m") ) // ถ้าได้รับ m
                manual(); // เข้าสู่การทำงาน Manual mode
                else if ( (ans[0] == 'a') || (ans == "a") ) // ถ้าได้รับ a
                automatic(); // เข้าสู่การทำงาน Automatic mode
```

```
        sock_err: // ถ้าเกิด error ระหว่างการเชื่อมต่อจะมาทำที่ส่วนนี้
        switch(status)
        {
                case 1: // foreign host closed
                break;
                case -1: // timeout
                break;
        }
    }
}
```