สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

การตรวจสอบสภาวะ PLC โดยผ่านโมเด็ม
(STATUS INSPECTION OF PLC WITH MODEM)

นายวินิจฉัย นินทรกิจ
นายพันธุ์เทพ กูระเสถียร
นายจิรยุทธ โชติศิลากุล

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตร์บัณฑิต
สาขาวิศวกรรมการวัดคุม
คณะวิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2542

เลขหมู่.....
เลขทะเบียน.....36805.....
วัน, เดือน, ปี.....29 ส.ค. 2543.....

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ปริญญานิพนธ์ปีการศึกษา 2542
ภาควิชาเทคโนโลยีการวัดคุมทางอุตสาหกรรม
สาขาวิชาวิศวกรรมการวัดคุม
คณะวิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง การตรวจสอบสภาวะ PLC โดยผ่านโมเด็ม
STATUS INSPECTION OF PLC WITH MODEM

ผู้จัดทำ

นายวินิจฉัย	นินทรกิจ	รหัสประจำตัว 40013421
นายพันธุ์เทพ	กูระเสถียร	รหัสประจำตัว 40012094
นายจิรยุทธ	โชติศิลากุล	รหัสประจำตัว 40012080

..... อาจารย์ที่ปรึกษา
(อาจารย์เชื้อ นกอยู่)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หัวข้อปริญญานิพนธ์ : การตรวจสอบสภาวะ PLC โดยผ่านโมเด็ม
โดย : นายวินิจฉัย นินทรกิจ
: นายพันธุ์เทพ กูระเสถียร
: นายจิรยุทธ โชติศิลากุล
ภาควิชา : เทคโนโลยีการวัดคุมทางอุตสาหกรรม
อาจารย์ที่ปรึกษา : อ.เชื้อ นกอยู่
ปีการศึกษา : 2542

บทคัดย่อ

ในปัจจุบันได้มีการนำเอา PROGRAMMABLE LOGIC CONTROL มาใช้ในการควบคุมการทำงานของเครื่องจักร และนิยมนำเอาคอมพิวเตอร์มาต่อเชื่อมเพื่อใช้แสดงสภาวะการทำงานของเครื่องจักร ในส่วนการเชื่อมต่อระหว่าง PLC กับ COMPUTOR จะมีอยู่หลายวิธีตามลักษณะการใช้งาน ในการต่อเชื่อมด้วยโมเด็มผ่านเครือข่ายโทรศัพท์มักจะใช้สำหรับในการติดต่อระยะไกล แต่ในการเชื่อมต่อแบบนี้มักจะให้คอมพิวเตอร์ ทำหน้าที่จัดการระบบการเชื่อมต่อ จากคอมพิวเตอร์ผ่านเครือข่ายโทรศัพท์ไปสู่เครื่อง PLC ซึ่งจะต้องมีการเสียคู่สายโทรศัพท์ 1 คู่สายในการเชื่อมต่อเสมอ ในโครงงานนี้จะเสนอการเชื่อมต่อด้วยโมเด็มผ่านเครือข่ายโทรศัพท์ แต่จะใช้ PLC ทำหน้าที่ติดต่อและส่งข้อมูลสภาวะของเครื่องจักร มาให้คอมพิวเตอร์ตามเวลาที่กำหนดไว้ใน PLC จึงทำให้สามารถใช้คู่สายร่วมกับโทรศัพท์ทั่วไปได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Project Report Title : Status Inspection Of PLC With Modem
By : Mr. Winitchai Nintarakit
: Mr. Panthep Kurasathain
: Mr. Jirayut Chotsilakul
Department : Industrial Instrumentation Technology
Project Report : Mr. Chuae Nokyoo
Academic : 1999

Abstract

Now Programmable Logic Control has been generally used to control the machines. For machine monitoring, computer mostly be used to the connect with PLC. There are many type of connecting computer to PLC. One type of connecting is the dial-up telephone network which using computer to dial the number, connect checking transmit data to PLC and call data from PLC. One pair of telephone cable for communication is needed for this system.

In this project the PLC is used to dial the number, connect checking, transmit data to computer as the perial setting. In this system we can use the telephone and PLC modem for one pair of telephone cable.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญภาพ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หน้า

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

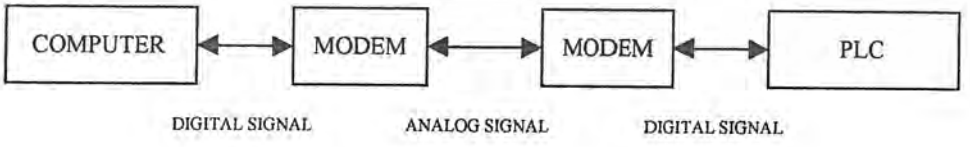
บทที่ 1
บทนำ

ในวงการอุตสาหกรรมปัจจุบัน ระบบควบคุมอัตโนมัติถูกนำมาใช้แทนแรงงานคนเป็นอันมาก เนื่องด้วยเหตุ และผลหลายประการ ที่เครื่องจักรควบคุมอัตโนมัติ ถูกเลือกมาใช้งาน อาทิเช่นความละเอียด ความเที่ยงตรง ความเร็ว และการทำงาน ภายใต้สิ่งแวดล้อมที่เป็นพิษซึ่งจะเป็นความได้เปรียบของเครื่องควบคุมอัตโนมัติ ถึงอย่างไรก็ตามการพัฒนาของระบบควบคุมอัตโนมัติ ก็ยังไม่จบลง เพื่อขจัดปัญหาในการเดินทาง ความคิดที่จะมีการควบคุมและติดตามผลในระยะไกล จึงเกิดขึ้นซึ่งปัจจุบันก็มีอยู่มาก เช่น สกาด้า หรือรี โมต คอนโทรลยูนิต แต่ในเครื่องนี้เราใช้ PLC เป็นตัวควบคุมและใช้คู่สายโทรศัพท์เป็นสื่อกลาง

เครื่องควบคุมแบบโปรแกรมได้ (PROGRAMMABLE LOGIC CONTROL) เป็นอุปกรณ์ที่ใช้กันแพร่หลาย และมีอุปกรณ์ใช้ร่วมกันเพื่อเพิ่มประสิทธิ์ภาพอีกมาก เครื่อง PLC ของ MODICON รุ่น 110 CPU 51200 เป็นรุ่นที่มีความสามารถได้กว้างขวางพอสมควรซึ่งมีความสามารถที่จะส่งและรับข้อมูลแบบอนุกรมได้ จึงสามารถที่จะติดต่อกับโมเด็ม เพื่อส่งข้อมูลผ่านสายโทรศัพท์

ที่เครื่องคอมพิวเตอร์ จะมีโมเด็มอีกตัวซึ่งทำหน้าที่รับข้อมูล จากเครื่องควบคุมกระบวนการเราสามารถที่จะดูสภาวะการทำงานของกระบวนการ และควบคุมสั่งงานจากหน้าจอคอมพิวเตอร์ได้

รูปที่ 1.1 แสดงการเชื่อมต่อของระบบ



รูปที่ 1.2 BLOCK DIAGRAM แสดงทิศทางสัญญาณ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 2
ทฤษฎีเนื้อหาประกอบโครงงาน

2.1 PLC (PROGRAMMABLE LOGIC CONTROL)

2.1.1 INTRODUCTION TO PROGRAMMABLE LOGIC CONTROL

ระบบควบคุมแบบซีเควนซ์ (SEQUENCE CONTROL) แต่เดิมประกอบด้วยไฟฟ้า เชิงกล (ELECTROMECHANICAL DEVICE) ซึ่งได้แก่ RELAY , TIMER , COUNTER แต่เดิมประกอบด้วยอุปกรณ์ ไฟฟ้าการทำงานของระบบควบคุมนี้จะมีอยู่ สองสภาวะด้วยกัน คือ เปิด กับ ปิด (ON , OFF) ระบบซีเควนซ์ที่ประกอบด้วยอุปกรณ์ ดังกล่าวนี้ มีขนาดใหญ่ ใช้กำลังงานสูง และเมื่อมีการเปลี่ยนแปลงการทำงานก็ต้องทำการสร้างวงจรควบคุมใหม่ ซึ่งไม่สามารถประยุกต์ใช้กับระบบควบคุมที่มีความยุ่งยากและซับซ้อน เหล่านี้เป็นต้น

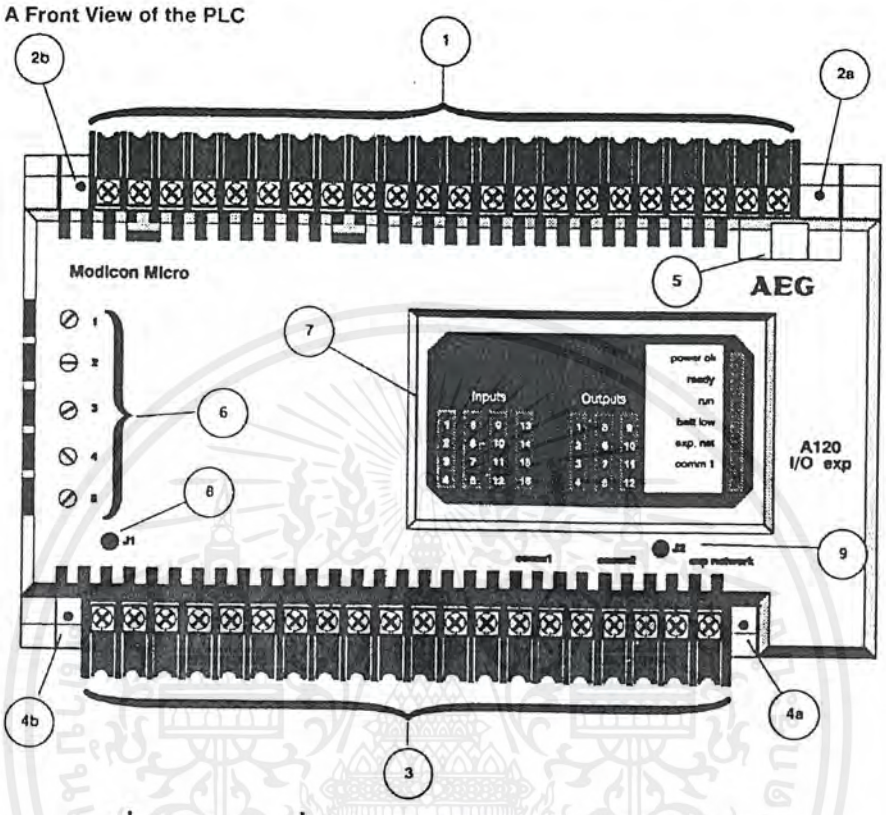
เนื่องจากเทคโนโลยีด้าน MICROPROCESSOR ในปัจจุบัน ได้เจริญก้าวหน้าไปมาก มีการนำ MICROCESSOR มาใช้กับการควบคุมแบบต่อเนื่อง ซึ่งก็คือเครื่องควบคุมแบบโปรแกรมได้ (PROGRAMMABLE SEQUENCE CONTROL) หรือเรียกสั้น ๆ ว่า PC ขึ้นมาใช้กับการควบคุมสำหรับงานด้านอุตสาหกรรมโดยเฉพาะ

PC จะมีส่วนที่เป็นอินพุทที่สามารถต่อใช้งานได้กับ ตัวตรวจจับต่าง ๆ (SENSOR) และส่วนเอาท์พุทจะต่อไป ควบคุมการทำงานของอุปกรณ์ หรือเครื่องจักรกล โดยสามารถสร้างวงจรหรือเงื่อนไข การทำงานของเครื่องจักรเหล่านี้ได้จากการป้อนโปรแกรมสั่งงานเป็นภาษาที่เรียกว่า LADDER DIAGRAM เข้าไป โปรแกรมนี้จะทำหน้าที่เหมือนกับวงจรรีเลย์ ตัวตั้งเวลา ตัวนับ และอื่น ๆ อีก โดยมีส่วนต่างๆ ดังต่อไปนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.1.2 PROGRAMMABLE LOGIC CONTROL MODICON 512/00 MICRO

ลักษณะทั่วไปของ MODICON 512/00 MICRO PLC



รูปที่ 2.1 ลักษณะทั่วไปของ MODICON 512/00 MICRO PLC

1. ส่วนขั้วต่อทางด้าน INPUT จะประกอบด้วยขั้วต่อ INPUT 16 ขั้ว และขั้วต่อ COMMON 2 ขั้ว ใช้ไฟ 24 VDC ทางด้าน INPUT

2. A/B ส่วนยึดขั้วต่อทางด้าน INPUT

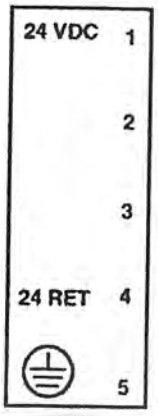
3. ส่วนขั้วต่อทางด้าน OUTPUT จะประกอบด้วยขั้วต่อ OUTPUT 12 ขั้ว ซึ่งต่อ COMMON 3 ขั้ว ใช้ไฟ 24 VDC ทางด้าน OUTPUT

4. A/B ส่วนยึดขั้วต่อทางด้าน OUTPUT

5. ส่วนที่ใส่ BATTERY

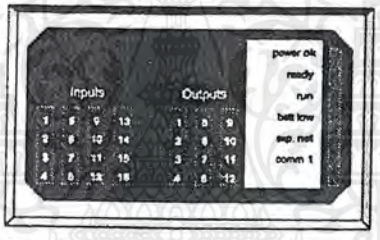
6. ส่วนขั้วต่อไฟเลี้ยง ประกอบด้วย ขั้วต่อไฟบวก (+) ,ลบ (-) 24 VDC และขั้วต่อกราวด์ ดังรูปที่ 2.2

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.2 ขั้วต่อไฟ SUPPLY

7. ส่วนแสดงผลของ CPU จะเป็น LED ที่ใช้แสดงผลเหมือน PLC โดยทั่วไป ดังจะได้ทราบความหมายจากตาราง



รูปที่ 2.3 แสดง LED ในตำแหน่งต่างๆ

หมวดแสดงผล

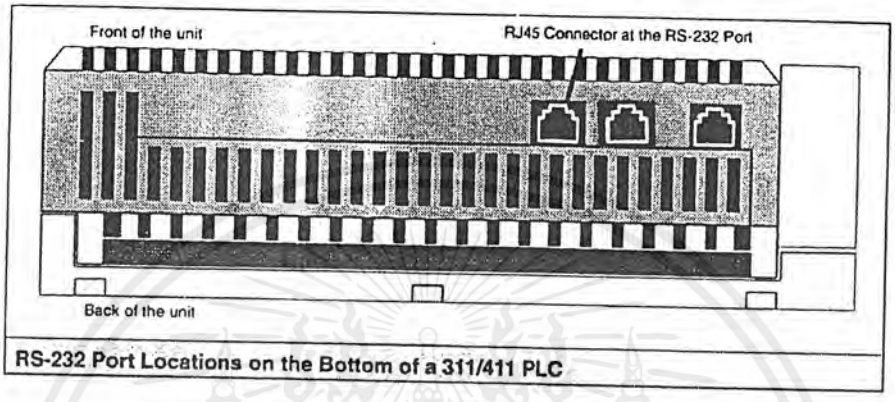
POWER OK	จะติดเมื่อจ่ายไปให้กับ CPU
READY	จะติดเมื่ออุปกรณ์ภายใน PLC พร้อมทุกส่วน
RUN	จะติดเมื่อ CPU ทำงาน
BATT LOW	จะติดเมื่อ BATTERY ภายในต้องการเก็บพลังงาน
EXP.NET	จะติดเมื่อมีการติดต่อทาง PORT I/O EXP LINK
COMM1	จะติดเมื่อมีการติดต่อทาง PORT COMM1
COMM2	จะติดเมื่อมีการติดต่อทาง PORT COMM2
INPUT	จะติดเมื่อมีไฟเลี้ยงที่ INPUT
OUTPUT	จะติดเมื่อมีการสั่งให้ OUTPUT ทำงาน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

8. ส่วนปรับระดับ GROUND TO CHASSIS GROUND

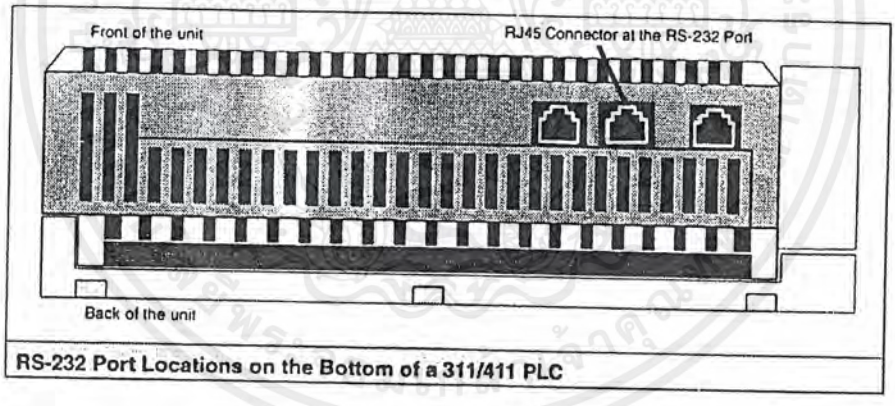
9. ส่วนปรับความต้านทานของ PLC

10. PORT COMM1 ใช้ RS-232 ในการติดต่อและใช้หัว CONNECT แบบ RJ45 ดังรูปที่ 2.4



รูปที่ 2.4 แสดง PORT COMM1

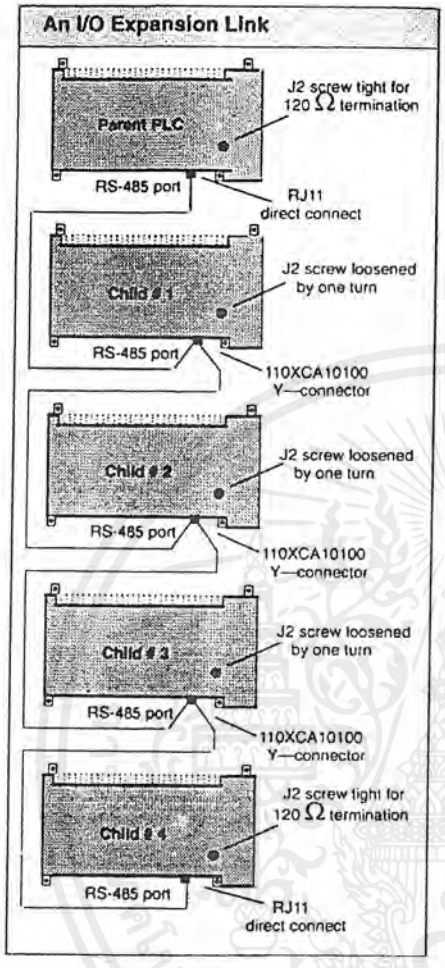
11. PORT COMM2 ใช้ RS-232 ในการติดต่อและใช้หัว CONNECT แบบ RJ45 ดังรูปที่ 2.5



รูปที่ 2.5 แสดง PORT COMM2

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

12. PORT EXP NETWORK

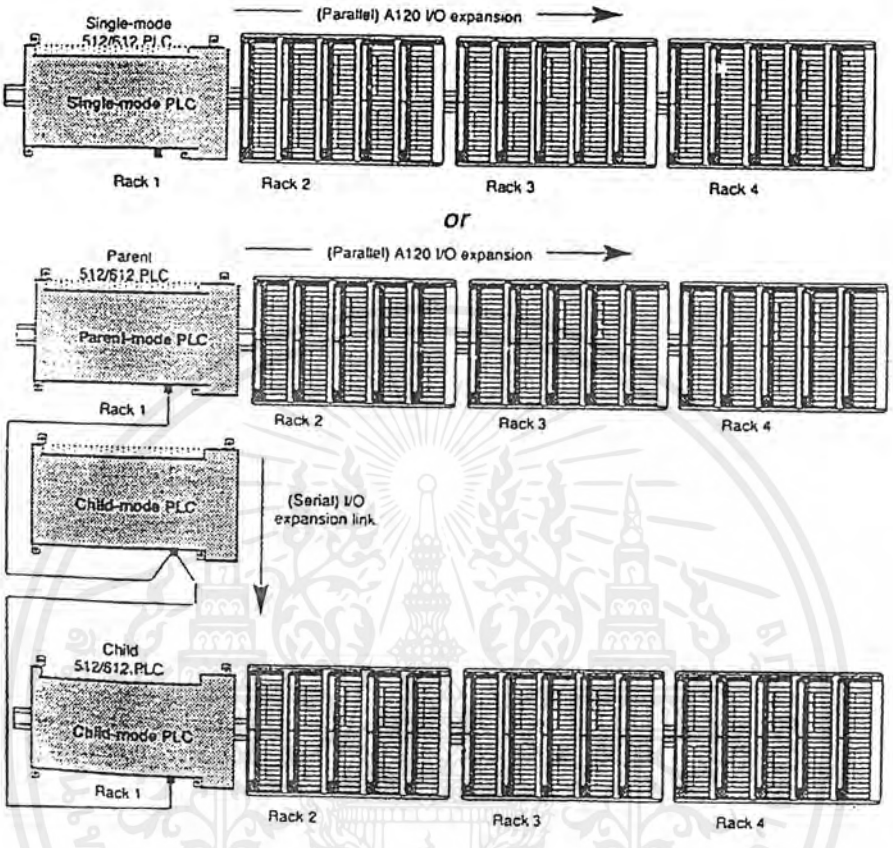


รูปที่ 2.6 แสดง I/O EXPANSION LINK

I/O EXPANSION LINK มีไว้สำหรับต่อ PARENT PLC กับ CHILD 1-4 ตัว ไว้สำหรับใช้งานด้วยกันและใช้หัว CONNECT แบบ RJ 11 ในการต่อกับ PORT โดยใช้สาย RS-485 ในการใช้งาน ดังรูปที่ 2.6

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

13. PORT A120 I/O EXP

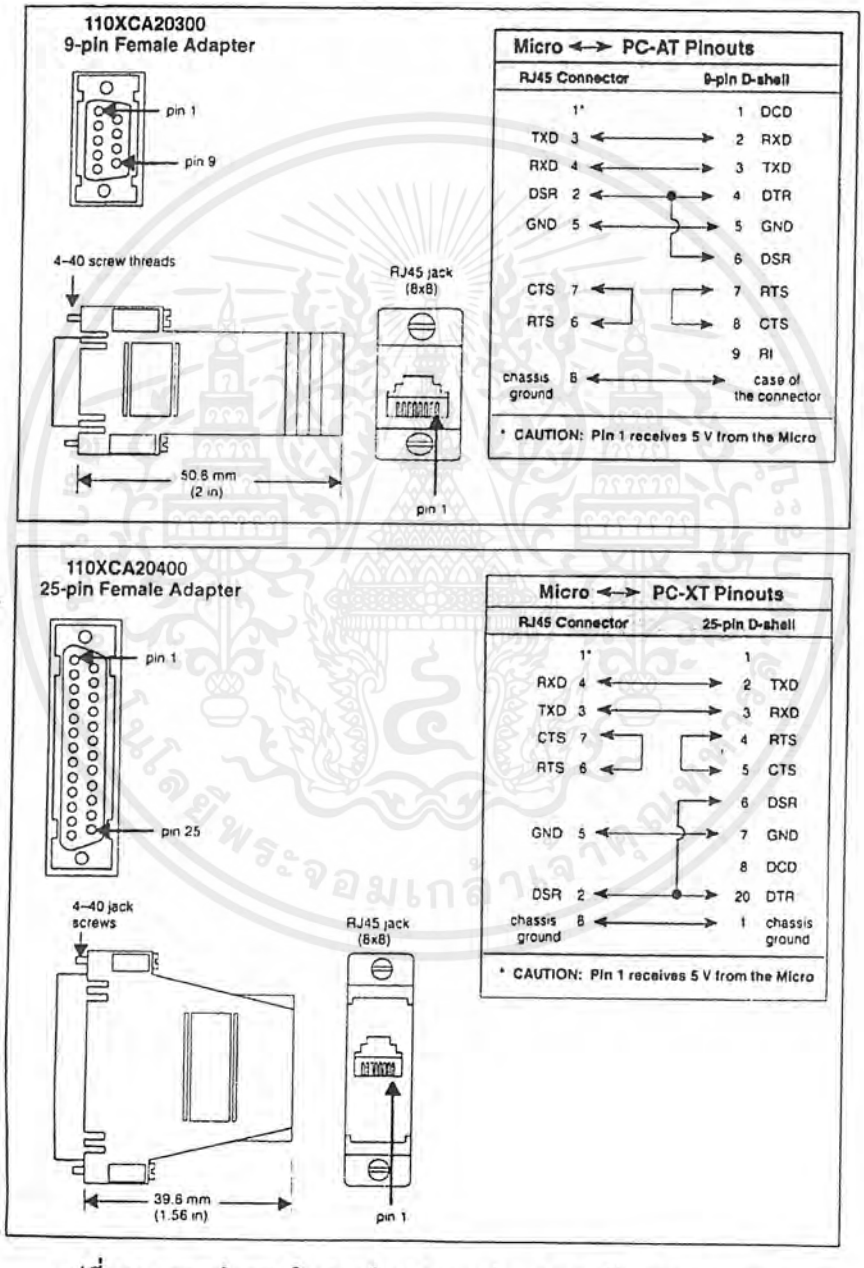


รูปที่ 2.7 แสดง PORT A120 I/O EXP

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.1.3 ลักษณะการต่อสายในการติดต่อสื่อสารในระบบแบบต่างๆ

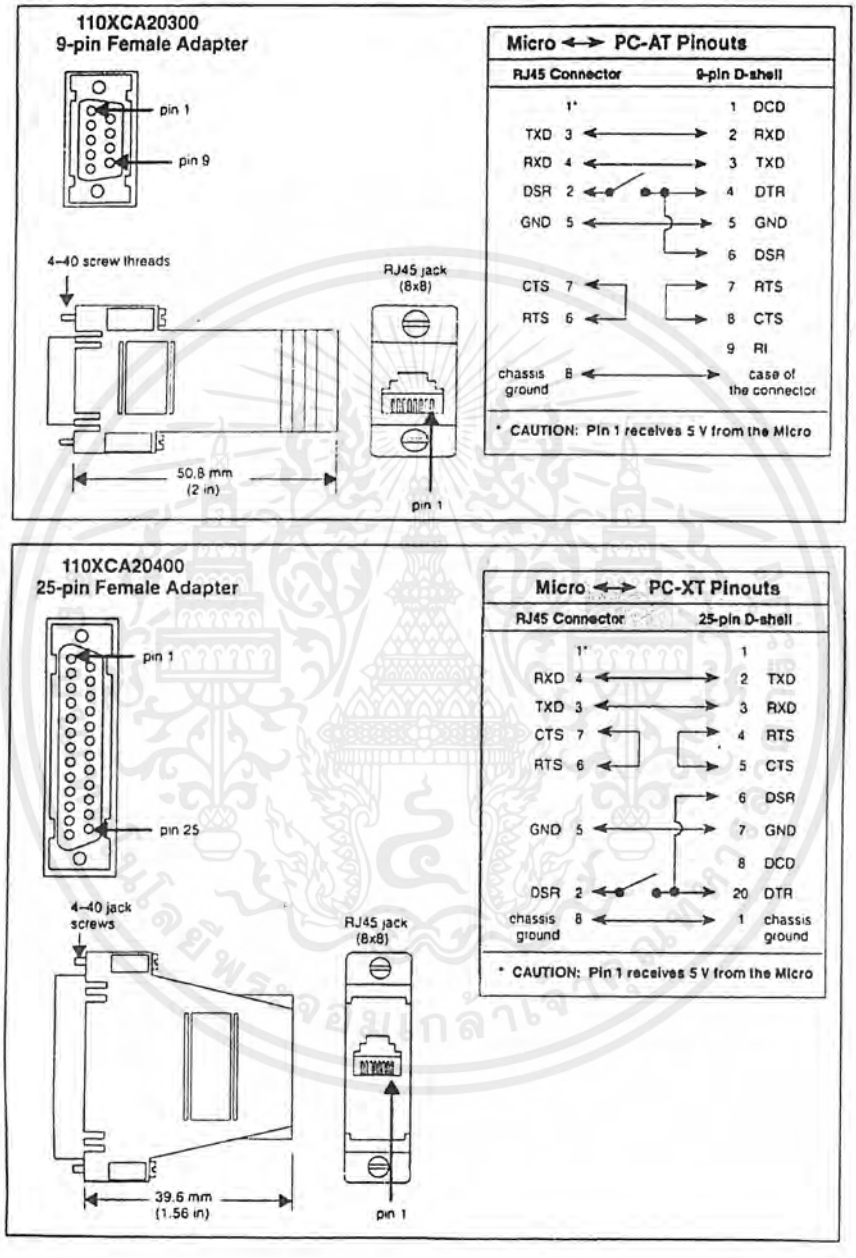
1. การต่อสายในการติดต่อระหว่าง PLC กับ COMPUTER จะใช้สาย RS-232 ในการติดต่อสื่อสารและใช้หัว CONNECT แบบ RJ45 ในการต่อเข้ากับ PORT COMM1 และ COMM2 ของ PLC โดยมีรูปแบบการต่อดังรูปที่ 2.8



รูปที่ 2.8 การต่อสายในการติดต่อระหว่าง PLC กับ COMPUTER

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2. การต่อสายในการติดต่อระหว่าง PLC กับ MODEM จะใช้สาย RS-232 ในการติดต่อสื่อสารและใช้หัว CONNECT แบบ RJ45 ในการต่อเข้ากับ PORT COMM1 และ COMM2 ของ PLC โดยมีรูปแบบการต่อดังรูปที่ 2.9



รูปที่ 2.9 การต่อสายในการติดต่อระหว่าง PLC กับ MODEM

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

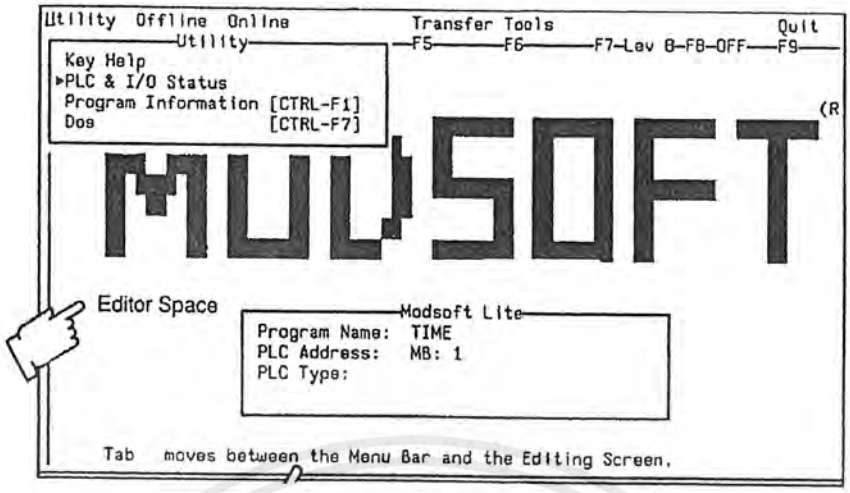
ดังรูปที่ 2.9 จะมี SWITCH ในการตัดต่อที่สาย DSR เพราะใน FUNCTION COMM จะต้องมีการตัดสัญญาณในสาย DSR ออก ดังนั้นเมื่อต่อสายในการติดต่อลักษณะต่างๆ เรียบร้อยก็สามารถนำมาต่อใช้ทั้งระบบได้ดังรูปที่ 2.10

รูปที่ 2.10 แสดงการเชื่อมต่อของระบบ

2.2 ส่วนของโปรแกรมควบคุม MODICON 512/00 MICRO PLC

โดยทั่วไปการที่เราจะใช้งานตัว PLC เราจะต้องทำการเขียน LADDER ให้กับตัว PLC โดยการคีย์โปรแกรมเข้าไป ซึ่งสามารถที่จะทำได้ทั้งการเขียนโดยเครื่องคีย์บันทึกภายนอกหรือโดยเขียนทางโปรแกรมเฉพาะของแต่ละ PLC ยี่ห้อนั้นๆ แต่สำหรับ MODICON PLC นั้นจะทำได้โดยการเขียนทางโปรแกรมโดยเฉพาะของ MODICON PLC ซึ่งก็เป็นโปรแกรมมาตรฐานตัวหนึ่งที่สามารถเขียนได้ทุกรุ่นของ PLC ยี่ห้อ MODICON ซึ่งโปรแกรมนั้นก็คือ โปรแกรม MODSOFT และในปัจจุบันก็มีหลายเวอร์ชั่นด้วยกันแต่ในรายงานนี้ใช้ VERSION 2.5 ในการเขียนโปรแกรม โดยโปรแกรมจะมีลักษณะหน้าจอดังรูปที่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.11 โปรแกรม MODSOFT VERSION 2.5

จากรูปหน้าจอจะแบ่งเป็น 3 ส่วน คือ

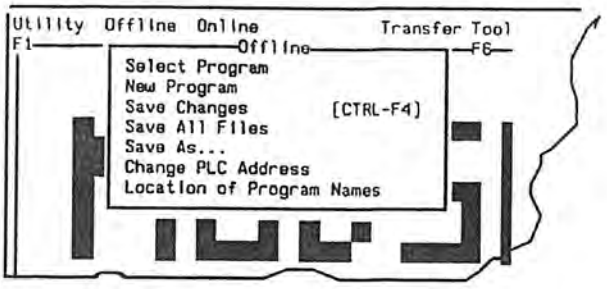
1. MAIN MENU LINE คือ ส่วนของเมนูหลักในการใช้ระบบของโปรแกรม
2. EDITOR SPACE คือ ส่วนของพื้นที่ในการเขียนโปรแกรม
3. TRACKING HELP LINE คือ ส่วนที่คอยบอกลักษณะใช้งานในแต่ละเมนู

ในการใช้งานหลักของโปรแกรม MODSOFT มี 3 MAIN MENU คือ

1. OFFLINE คือ ใช้ในการเขียนหรือแก้ไขโปรแกรมต่างๆ ที่เราต้องการนำไปใช้งานให้กับตัว PLC ซึ่งจะเก็บไว้ในตัวโปรแกรม MODSOFT
2. ONLINE คือ ใช้ในการติดต่อกับ PLC โดยต่อกับคอมพิวเตอร์ เราสามารถที่จะดูโปรแกรมในตัว PLC ที่เราติดต่ออยู่ได้และยังสามารถแก้ไขได้ทันทีตามที่เราต้องการ
3. TRANSFER คือ ใช้ในการจัดบันทึกโปรแกรมทั้งจาก PLC มาสู่โปรแกรม MODSOFT หรือจะบันทึกโปรแกรมจากโปรแกรม MODSOFT ไปสู่ PLC ก็ได้

ดังนั้นในการเขียนโปรแกรมที่เราต้องการ เราจะต้องใช้งานจะเริ่มเขียนในเมนู OFFLINE โดยในเมนู OFFLINE ประกอบด้วยดังรูปที่ 2.12

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.12 เมนู OFFLINE

ดังรูปที่ 2.12 ประกอบด้วย

SELECT PROGRAM คือ มีไว้สำหรับเลือกโปรแกรมที่เราเก็บไว้ในโปรแกรม MODSOFT

NEW PROGRAM คือ มีไว้สำหรับสร้างโปรแกรมใหม่

- SAVE CHANGE คือ มีไว้สำหรับ SAVE โปรแกรมที่มีการแก้ไข
- SAVE ALL FILE คือ มีไว้สำหรับ SAVE ทั้งโปรแกรมใหม่หรือที่มีการแก้ไข
- SAVE AS คือ มีไว้สำหรับ SAVE โปรแกรมพร้อมทั้งตั้งชื่อใหม่
- CHANGE PLC ADDRESS คือ มีไว้สำหรับ SET ค่าต่างๆ ในการติดต่อสื่อสารกับ PLC โดยจะมีค่าต่างๆ ให้ SET ดังรูปที่ 2.13

Address = 1-247

Protocal = Modbus 1
Modbus Plus
Default *

Mode = ASCII (7 data bits) RTU (8 data bits)

Parity = NONE EVEN ODD

Stop Bits = 1 or 2

Baud = 50 75 110 134
150 300 600 1200
1800 2000 2400 3600
4800 7200 9600 19200

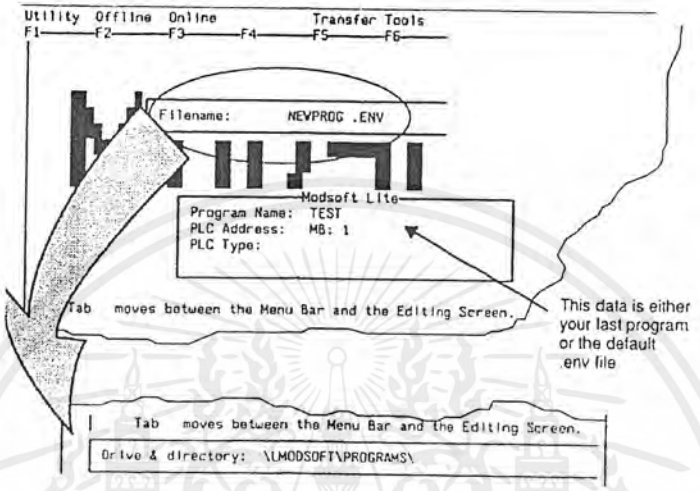
Device COM1 through COM8

รูปที่ 2.13 แสดง CHANGE PLC ADDRESS

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

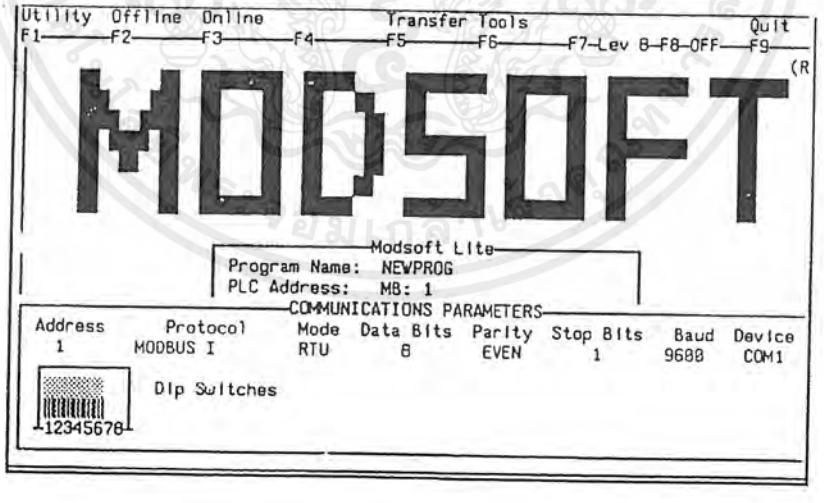
- LOCATION OF PROGRAM NAME คือ มีไว้สำหรับบอกที่อยู่ของ PROGRAM ที่เลือกใช้อยู่

ดังนั้นการเริ่มใช้งานเราจะต้องสร้างโปรแกรมใหม่ก่อนโดยใช้เมนูย่อย NEW PROGRAM ดังรูปที่ 2.14



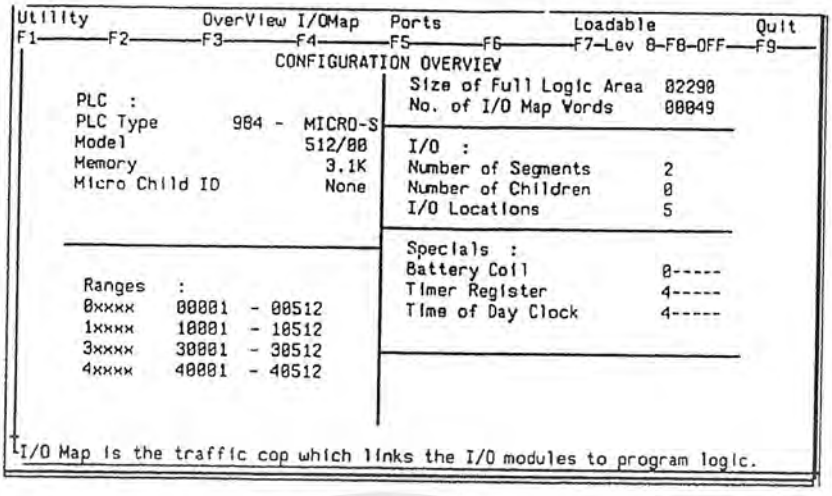
รูปที่ 2.14 การสร้างโปรแกรมใหม่โดยใช้เมนูย่อย NEW PROGRAM

เราจะเริ่มต้นโดยการสร้างชื่อโปรแกรม เมื่อสร้างชื่อโปรแกรมแล้วโปรแกรมจะให้เรา SET ค่าต่าง ๆ ดังรูปที่ 2.15

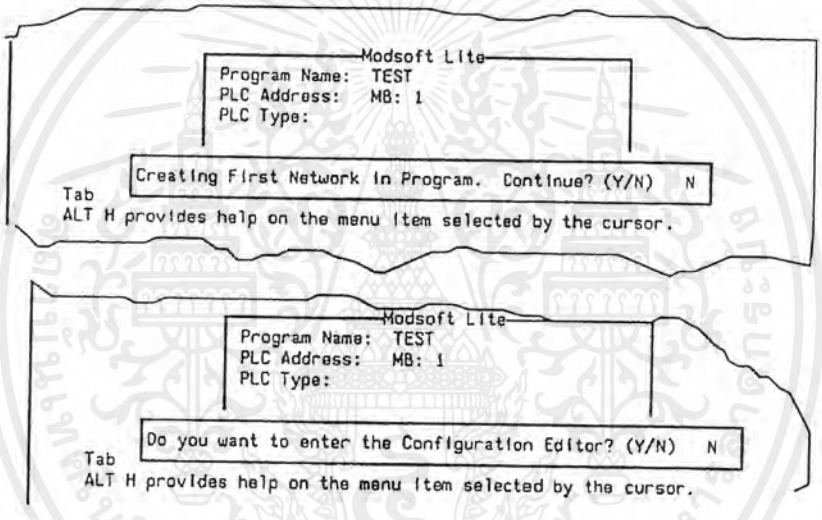


รูปที่ 2.15 แสดงค่าพารามิเตอร์

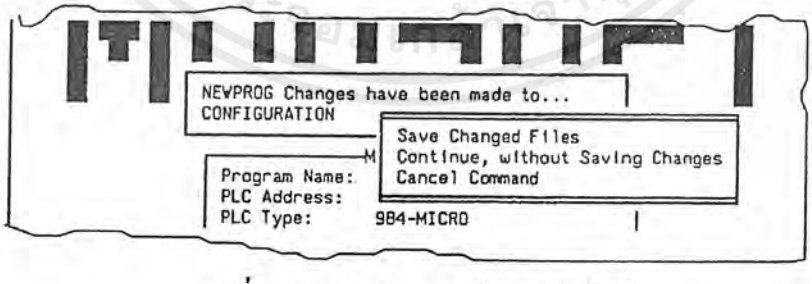
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.16 แสดงค่า CONFIGURATION



รูปที่ 2.17 แสดงการยืนยันการ SET ค่าใหม่

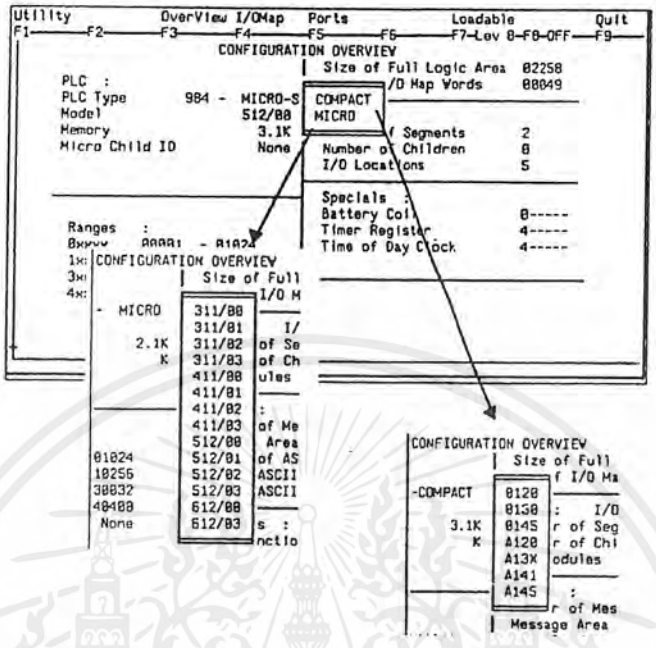


รูปที่ 2.18 การ SAVE โปรแกรมใหม่

เมื่อ SET ค่าต่างๆ ตามที่เราต้องการแล้ว เราก็สามารถ LADDER LOGIC เพื่อนำไปควบคุมการทำงานตามที่เราต้องการได้

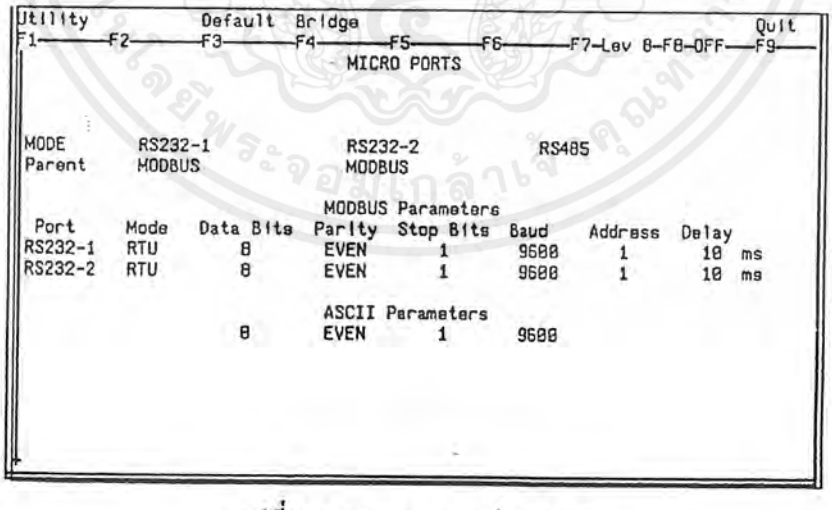
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การ SET ค่า CONFIGURATION ดังรูปที่ 2.19



รูปที่ 2.19 การ SET ค่า CONFIGURATION

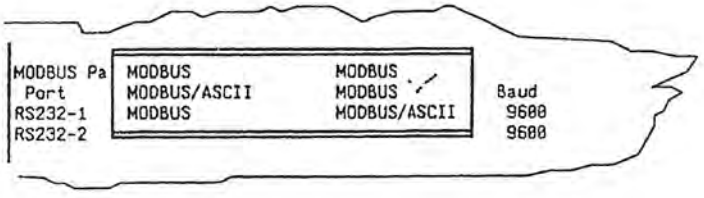
คือ จะต้อง SET ค่า ชนิดของ PLC จะมี 2 ชนิด คือ 1. COMPACT , 2. MICRO โดยในแต่ละชนิดจะมีให้เลือก CPU แต่ละชนิดดังรูป ในรายงานนี้เราจะต้องเลือกชนิด MICRO และรุ่น CPU 512/00 และยังจะต้อง SET ค่า PORT อีกดังรูปที่ 2.20



รูปที่ 2.20 การ SET ค่า PORT

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ดังรูปที่ 2.20 จะมีทั้งการ SET ค่าการติดต่อแต่ละ PORT รวมทั้งรูปแบบการติดต่อสื่อสารในขณะที่ใช้ FUNCTION COMM ซึ่งการ SET ค่าต่างๆ อยู่ในรูปที่ 2.21



รูปที่ 2.21 รูปแบบการ SET ค่า PORT

Autoconfiguration Parameters for a Single Mode Micro PLC

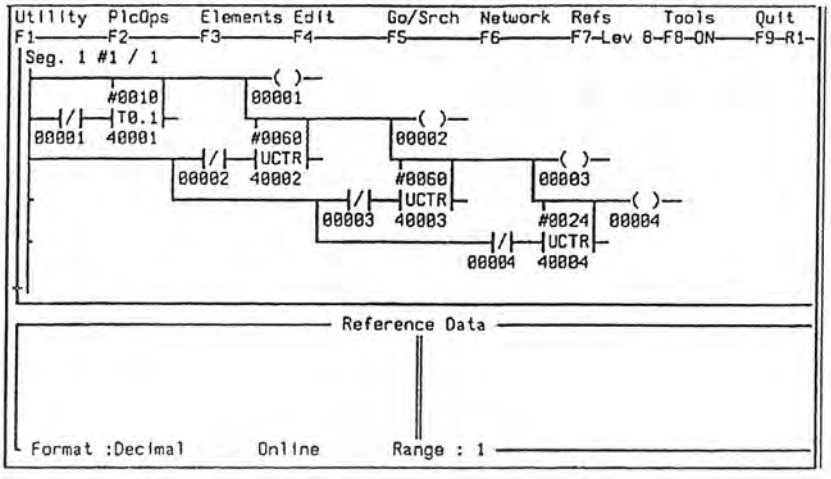
Parameter	110CPU Models	
	311 / 411	512 / 612
Number of 0x references	1024	1536
Number of 1x references	256	512
Number of 3x references	32	48
Number of 4x references	400	1872
Number of ladder logic segments	2 (the first for control logic and the second for subroutines)	2 (the first for control logic and the second for subroutines)
RS-232 port (comm 1)	Dedicated Modbus mode: 8-bit RTU communications, 9600 baud, even parity, 1 STOP bit, Modbus address #1	Dedicated Modbus mode: 8-bit RTU communications, 9600 baud, even parity, 1 STOP bit, Modbus address #1
RS-232 port (comm 2)	N/A	Dedicated Modbus mode: 8-bit RTU communications, 9600 baud, even parity, 1 STOP bit, Modbus address #1
RS-485 port (exp. link)	Dedicated ASCII 8-bit ASCII communications, 9600 baud, even parity, 1 STOP bit	Dedicated ASCII 8-bit ASCII communications, 9600 baud, even parity, 1 STOP bit

รูปที่ 2.22 AUTOCONFIGURATION PARAMETERS

เมื่อเราสร้างโปรแกรมใหม่แล้วเข้ามาอยู่ในโปรแกรมที่เราสร้างจะมีเมนูต่างๆ ดังรูปที่ 2.23

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง



รูปที่ 2.23 แสดง LADDER DIAGRAM ของโปรแกรม

เราสามารถเขียน LADDER LOGIC ได้ในเมนู ELEMENTS ในเมนู ELEMENTS จะมี LADDER LOGIC มากมาย ดังในรูปที่ 2.24

Standard Ladder Logic Instructions (available on all Micro PLCs)	
Instruction	Description
Relay Logic	
-| |-	A normally open (N O) contact
-|/|-	A normally closed (N C) contact
-|↑|-	A positive transitional contact
-|↓|-	A negative transitional contact
-( )-	A normal coil
-(M)-	A memory-retentive coil
Counters	
UCTR	An up counter from 0 to a specified preset
DCTR	A down counter to 0 from a specified preset
Timers	
T1.0	A timer that increments in seconds
T0.1	A timer that increments in tenths of a second
T.01	A timer that increments in hundredths of a second
T1MS	A timer that increments in ms
Integer Math	
ADD	Addition
SUB	Subtraction or greater than/less than operations
MUL	Multiplication
DIV	Division
Data Move	
R→T	A register-to-table move
T→R	A table-to-register move
T→T	A table-to-table move
BLKM	A block move
FIN	A first-in operation to a queue
FOUT	A first-out operation from a queue
SRCH	A table search for a bit pattern in one of the registers

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้
36805

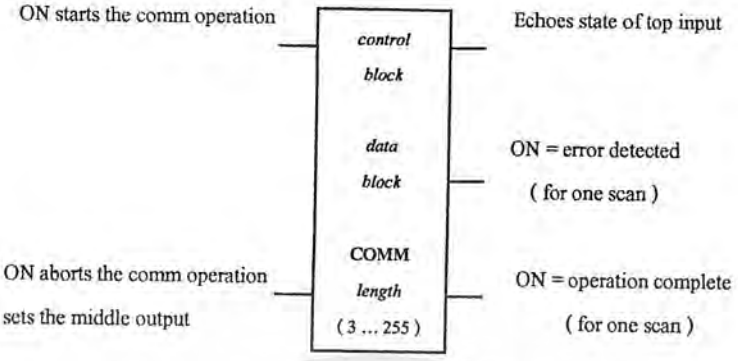
Standard Ladder Logic Instructions (continued)	
Instruction	Description
Data Matrix	
AND	A logical AND of two matrices
OR	A logical OR of two matrices
XOR	A logical exclusive OR of two matrices
COMP	A logical complement of the bit pattern in a matrix
CMPR	A logical compare of the bit patterns in two matrices
MBIT	A bit modify—i.e., changing the current (1, 0) value of the bit
SENS	A bit sense—i.e., reporting the current (1, 0) value of the bit
BROT	A bit rotation—i.e., shifting the bit positions left or right in a matrix
ASCII	
COMM	An ASCII read or write communication operation
Sequencing	
SCIF	Drum sequencing and input comparison operations
Subroutines	
JSR	Jumps the logic scan from control logic to a ladder logic subroutine programmed in the last segment
LAB	Labels the entry location for the called subroutine in the last segment
RET	Returns the logic scan to its previous place in logic prior to the JSR
CTIF	Sets up the high-speed inputs for interrupt and counter/timer operations
Other	
STAT	Checks and reports the health of the PLC and its I/O
SKP	Causes the logic scan to skip specified networks in the program

Enhanced Ladder Logic Instructions (available in specified 110CPU512 and 110CPU612 Models only)	
Instruction	Description
BLKT	A block-to-table move
TBLK	A table-to-block move
CKSM	Performs CRC-16, LRC, straight, or binary checksum operations
PID2	Performs proportional-integral-derivative control functions
EMTH	Performs extended math functions such as square root, process square root, log, antilog, and floating point operations

รูปที่ 2.24 STANDARD LADDER LOGIC INSTRUCTIONS

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โครงสร้างของ FUNCTION COMM 1120 (FUNCTION เขียน)



รูปที่ 2.25 โครงสร้างของ FUNCTION COMM 1120

รูปแบบของ CONTROL BLOCK

4X	1120	รูปแบบการเขียน
4X+1	ERROR	สภาวะการผิดผลาด
4X+2	20	จำนวนขอบเขต DATA
4X+3	0	ไม่ใช้
4X+4	0	ไม่ใช้
4X+5	1	ตัวเลขเครื่อง
4X+6	0	ไม่ใช้
4X+7	0	ไม่ใช้
4X+8	0	ไม่ใช้
4X+9	0	ไม่ใช้

รูปแบบของ DATA BLOCK คือ จะเริ่มตัวอักษรจาก ADDRESS ใน DATA BLOCK และมีขอบเขตตามที่กำหนดจากค่า K ใน BLOCK COMM โดยจะเขียนในหมวด HEXIDECIMAL จึงจะเขียนออกมาในรูปแบบ ASCII ตามที่เราต้องการได้ เมื่อเราเขียน LADDER LOGIC ต่างๆ ตามที่เราต้องการได้แล้ว เราก็จะต้องนำโปรแกรมนี้บันทึกลงสู่ MODICON 512/00 MICRO PLC โดยใช้เมนู TRANSFER โดยในเมนูจะประกอบด้วย 3 เมนูย่อย คือ

- PLC TO FILE
- VERIFY ALL
- FILE TO PLC

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เราสามารถส่งถ่ายข้อมูลจากโปรแกรมสู่ PLC ได้โดยเมนู FILE TO PLC โดยเราเลือกชื่อโปรแกรมที่เราต้องการ และเราบันทึกโปรแกรมสู่ PLC ได้ตามต้องการ PLC ก็จะเก็บข้อมูลที่เราบันทึกไว้ในหน่วยความจำ เราก็สามารถนำ PLC ไปใช้งานได้ตามต้องการได้

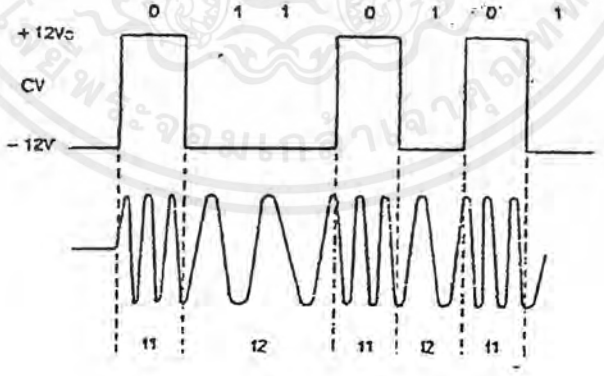
2.3 โมเด็ม (MODEM)

สาเหตุที่นำสัญญาณดิจิตอลส่งเข้าสู่เครือ ข่ายสื่อสาร โทรศัพท์โดยตรง ซึ่งเป็นการส่งข้อมูลในระยะที่ไกลมากๆ จะทำให้สัญญาณเกิดการเสียรูปทรง จึงแก้ปัญหาโดยการส่งข้อมูลไปตามสายส่งในแบบอนาลอกแทน จึงจำเป็นต้องทำการเปลี่ยนสัญญาณดิจิตอลให้อยู่ในรูปแบบที่เหมาะสมโดยการนำไปฝากกับสัญญาณพาหะ กระบวนการที่นำสัญญาณที่จะส่ง (Digital Signal) แฝงเข้าไปในสัญญาณพาหะ เรียกว่า โมดูเลท (Modulate) และเมื่อส่งไปตามสายส่งจนถึงผู้รับแล้วที่ผู้รับจะมีขบวนการในการที่จะแยกสัญญาณดิจิตอล ออกจากสัญญาณพาหะ ซึ่งกระบวนการนี้เรียกว่า ดีโมดูเลท (Demodolate)

2.3.1 มาตรฐานการผสมสัญญาณของโมเด็ม

การโมดูเลท แบ่งได้ 3 วิธี

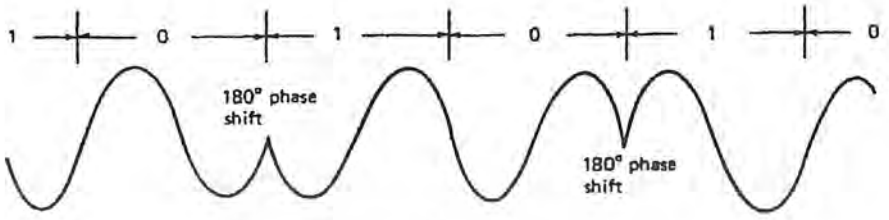
1. Frequency Shift Keying (FSK) เป็นระบบเก่า ใช้กับโมเด็มความเร็วต่ำ โดยแทน 0 และ 1 ด้วยความถี่ต่างกัน โดยผู้ส่งจะใช้ความถี่ 2 ความถี่ แทน 0 และ 1 ทางด้านส่ง และผู้รับจะใช้ความถี่ 2 ความถี่ แทน 0 และ 1 ทางด้านรับ เช่นกัน รวมใช้ความถี่ 4 ความถี่



รูปที่ 2.26 การผสมสัญญาณแบบ FSK

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2. Phase Shift Keying (PSK) ใช้หลักการแทนข้อมูล 0 และ 1 ด้วยการเปลี่ยนแปลงมุมของช่วงส่งสัญญาณในสายส่ง (Phase)



รูปที่ 2.27 การผสมสัญญาณแบบ PSK

3. Quadrature Amplitude Modulation (QAM) ใช้กับโมเด็มความเร็วสูง นิยมใช้กันเป็นการผสมสัญญาณที่ใช้ทั้งการเปลี่ยน Phase และขนาดของสัญญาณควบคู่กันไป

โมเด็มจำเป็นต้องมีมาตรฐานเพื่อให้ผู้ผลิตแต่ละบริษัทผลิตโมเด็มออกมาใช้รับส่งข้อมูลระหว่างกันได้ จึงทำให้เกิดมาตรฐานการส่งข้อมูลขึ้น เป็นไปตามองค์การมาตรฐานสื่อสารสากล หรือ CCITT ซึ่งมีรายละเอียดเกี่ยวกับ ความเร็วในการส่งข้อมูล ความถี่ที่ใช้และเทคนิคการส่งสัญญาณในสายส่ง โดยมีชื่อเหล่านี้ว่า V - Series

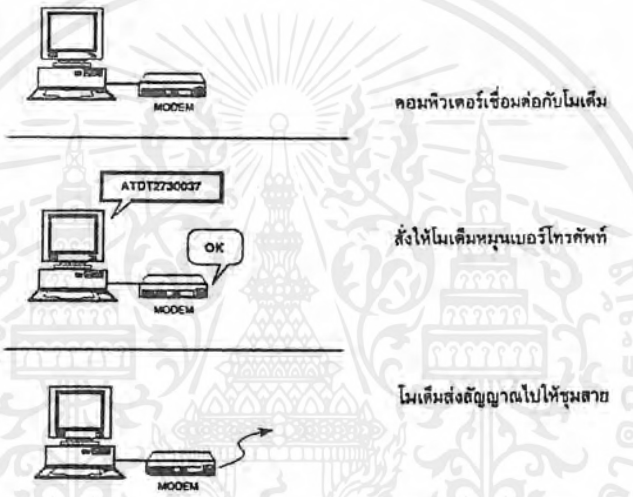
2.3.2 มาตรฐานของโมเด็มตาม CCITT V-Series

มาตรฐานที่ใช้กันอยู่ในปัจจุบันนี้เป็นไปตามที่องค์การมาตรฐานสื่อสารสากลหรือ CCITT เป็นผู้กำหนดขึ้น โดยมีชื่อเรียกแต่ละมาตรฐานของโมเด็มขึ้นต้นด้วยอักษร "V " และตามด้วยตัวเลข เราจึงเรียกมาตรฐานเหล่านี้ชื่อหนึ่งว่า V-Series นอกมาตรฐานของโมเด็มแล้ว CCITT ยังกำหนดมาตรฐานการสื่อสารอื่นๆ อีก เช่น มาตรฐานของการสื่อสารผ่านดาวเทียม มาตรฐานของโทรสาร (Facsimile) มาตรฐานการสื่อสารข้อมูลต่างๆ ทั้งในแบบดิจิตอลและอนาลอก รวมถึงมาตรฐานเกี่ยวกับระบบโทรศัพท์อีกด้วย มาตรฐานที่ CCITT เป็นผู้กำหนดได้รับการยอมรับกันทั่วโลก การติดต่อสื่อสารระหว่างประเทศจึงดำเนินไปได้อย่างมีปัญหา เนื่องจากทุกๆ คนต่างก็ทำตามมาตรฐานเดียวกัน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.3.3 มาตรฐานคำสั่งโมเด็ม

ปกติโมเด็มจะติดต่อกับเครื่องคอมพิวเตอร์ เพื่อทำหน้าที่รับส่งข้อมูลแล้ว ดังนั้นขณะที่โมเด็มยังไม่ได้ติดต่อปลายทางเพื่อส่งข้อมูล คอมพิวเตอร์สามารถส่งคำสั่งต่างๆ ให้โมเด็มได้โดยไม่รบกวนการส่งข้อมูลแต่อย่างใด ในขั้นแรกเมื่อเปิดสวิตช์ให้โมเด็มทำงาน สัญญาณที่โมเด็มได้รับจากคอมพิวเตอร์จะถือว่าเป็นคำสั่งทั้งหมดจนกว่าจะติดต่อกับโมเด็มปลายทางได้ และเมื่อโมเด็มทำการรับส่งข้อมูลผ่านสายส่ง สัญญาณต่างๆ ที่คอมพิวเตอร์ส่งให้โมเด็ม จะถือว่าเป็นข้อมูลทั้งหมด จนกระทั่งหยุดส่งข้อมูลโดยการเลิกการติดต่อกับปลายทาง หรือวางสายโทรศัพท์นั่นเอง โมเด็มก็กลับมาอยู่ในช่วงรับคำสั่งจากคอมพิวเตอร์อีกครั้ง



รูปที่ 2.28 คอมพิวเตอร์สั่งงานโมเด็มโดยใช้ AT command

คำสั่งโมเด็มจะควบคุมการทำงานที่จำเป็นทั้งหมดของโมเด็ม เช่น ตอบรับสัญญาณ โทรศัพท์ที่เรียกเข้ามา เลือกให้ทำงานแบบ Echo on หรือ Echo off ต่อเข้าสายโทรศัพท์หรือวางสายโทรศัพท์ที่เรียกเข้ามา เลือกให้ทำงานในแบบ Echo off ต่อเข้าสายโทรศัพท์หรือวางสายโทรศัพท์รีเซตโมเด็มสั่งให้โมเด็มหมุนโทรศัพท์ตามเบอร์ที่กำหนด ปรับพารามิเตอร์ต่างๆของโมเด็ม ซึ่งถ้าหากไม่ใช้คำสั่งโมเด็มแล้ว ผู้ใช้จะต้องกำหนดตัวแปรเหล่านั้น ด้วยวิธีการผลักสวิทช์บนโมเด็ม การใช้คำสั่งจึงสะดวกและง่ายต่อการใช้งานมาก ข้อดีอีกอันหนึ่งของการใช้คอมพิวเตอร์ส่งคำสั่งให้โมเด็มก็คือ ซอฟแวร์ติดต่อสื่อสาร สามารถปรับตัวแปรต่างๆ ของโมเด็มให้เป็นไปอย่างที่ต้องการ โดยที่ผู้ใช้ไม่ต้องรู้รายละเอียดใดๆ เลย โปรแกรมจะตัดการให้เสร็จและติดต่อส่งข้อมูลได้ทันที โปรแกรมคนละโปรแกรมก็จะปรับโมเด็มให้ทำงานต่างกันได้โดยไม่ต้องแก้ไขส่วนที่เป็นฮาร์ดแวร์ของโมเด็มเลย การใช้งานโมเด็ม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จึงมีความคล่องตัวมากกว่าการใช้สวิตซ์เลือกแบบเก่า ซึ่งถ้ามีการเปลี่ยนแปลงอะไรเราก็ต้องปรับสวิตซ์กันทีหนึ่งทุกครั้งไป และอาจเกิดความผิดพลาดได้ง่ายกว่าการใช้คำสั่งงานโมเด็ม

ภายในตัวโมเด็มจะมีหน่วยความจำพิเศษสำหรับเก็บตัวแปรในการเก็บการทำงานแทนที่สวิตช์แบบเก่าหน่วยความจำนี้จะยังเก็บค่าต่างๆ เอาไว้ได้ แม้ว่าจะปิดโมเด็มหรือดึงปลั๊กโมเด็มออกก็ตาม โมเด็มที่ใช้คำสั่ง Hayes เรียกหน่วยความจำส่วนนี้ว่า S-Register เอาไว้ใช้เก็บพารามิเตอร์ในการทำงานของโมเด็ม เช่น จำนวนครั้งที่จะตอบรับสัญญาณเรียกเข้า ช่วงเวลาสำหรับรอสัญญาณก่อนหมุนโทรศัพท์ ดังรูป

Smartmodem S-Register Set

Register	Description	Units	Range	Default
S0	Select ring to answer on	Rings	0-255	0
S1	Ring count (incremented with each ring)	Rings	0-255	0
S2	Escape-sequence character	ASCII	0-127	43(+)
S3	Carriage-return character	ASCII	0-127	13
S4	Line-feed character	ASCII	0-127	10
S5	Backspace character	ASCII	0-32,127	8
S6	Wait-time before blind dialing	Seconds	2-255	2
S7	Wait-time for carrier/dial tone	Seconds	1-255	30
S8	Length of pause for comma in number	Seconds	0-255	2
S9	Carrier-detect response time	.1 second	1-255	6
S10	Delay between carrier loss/hangup	.1 second	1-255	7/14
S11	Duration/spacing of DTMF tones	.001 second	50-255	70/195
S12	Escape-sequence guard time	.02 second	20-255	50
S16	Test mode			
S18	Select test timer	Seconds	0-255	0
S25	Data terminal ready change detect time	.01 second	0-255	5
S26	Select RTS to CTS delay	.01 second	0-255	1

รูปที่ 2.29 ตารางแสดง S- Register ของ Hayes

ซอฟต์แวร์ที่ใช้สื่อสารสามารถเปลี่ยนค่าตัวแปรเหล่านี้ชั่วคราว หรือเปลี่ยนค่าถาวรไปเลยก็ได้โดยใช้คำสั่งเก็บค่าตัวแปรเอาไว้ หน่วยความจำพิเศษนี้ บางชนิดใช้แบตเตอรี่เล็กๆ คอยจ่ายไฟให้เวลาที่เราปิดโมเด็ม เพื่อป้องกันค่าต่างๆหายไปจากหน่วยความจำ ดังนั้น เมื่อใช้โมเด็มไปนานๆ แบตเตอรี่ดังกล่าวจะหมดลง อาจจำเป็นต้องเปลี่ยนอันใหม่ให้ ไม่เช่นนั้นการทำงานของโมเด็มอาจผิดพลาดได้เนื่องจากค่าของตัวแปรใน หน่วยความจำหายไป โมเด็มบางชนิดเก็บค่าตัวแปรในหน่วยความจำแบบที่ไม่ต้องใช้แบตเตอรี่ จ่ายไฟสำรองให้โมเด็ม แบบนี้ก็ไม่จำเป็นต้องเปลี่ยนแบตเตอรี่ภายในให้ และโมเด็มบางแบบก็ไม่ยอมให้ เปลี่ยนค่าตัวแปรเหล่านี้อย่างถาวรโดยเก็บค่าต่างๆเอาไว้ใน ROMการ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เปลี่ยนค่าตัวแปรจะเป็นไปชั่วคราวเท่านั้น เมื่อปิดโมเด็มใหม่ ค่าต่างๆจะกลับเหมือนเดิมตามที่ผู้ผลิตกำหนดเอาไว้ใน ROM ของโมเด็มนั่นเอง

มาตรฐานคำสั่งโมเด็มของ Hayes

Hayes Command เป็นคำสั่งที่ใช้สั่งงานโมเด็ม มีอีกชื่อหนึ่งเรียกกันว่า AT Command เพราะคำสั่งทุกคำสั่งขึ้นต้นด้วยอักษร AT เสมอ เมื่อจบคำสั่งให้ปิดท้ายด้วย CR (Carriage Return) หรือกดปุ่ม Enter โมเด็มจะรับคำสั่งนั้นไปทำงานทันที และตอบว่า OK กลับมา คำสั่งเรียงตัวอักษร A ถึง Z จะมีดังนี้คือ

A Manual answer

คำสั่งในการตอบรับการเรียนเข้ามา ใช้ในกรณีที่ไม่ได้เป็นการตอบรับอัตโนมัติ

D Dial (หมุนเบอร์โทรศัพท์)

เป็นคำสั่งในการหมุนโทรศัพท์ คำสั่ง ATDn ใช้ในการหมุนเบอร์โทรศัพท์ (n แทนหมายเลขโทรศัพท์) การเว้นวรรค เครื่องหมายไฮโฟน (-) และวงเล็บ สามารถใช้เพื่อให้ดูเป็นระเบียบ ได้เพราะว่าโมเด็มจะไม่สนใจในเครื่องหมายเหล่านี้ ถ้าไม่บ่งบอกว่า เป็นสัญลักษณ์ระบบโทน หรือพัลส์ จะถือวาเป็นระบบโทน และอักษรที่ใช้ร่วมกับคำสั่ง D นี้อีกมาก ดังที่จะพบต่อไป

T และ P หมายถึงระบบโทนหรือพัลส์

H Hang up (วางหูโทรศัพท์)

คำสั่งนี้จะทำการวางหูโทรศัพท์

AT H

L ตั้งเสียงของลำโพง

ใช้คำสั่ง L ในการกำหนดเสียงของลำโพง

L เสียงค่อยมาก

L1 เสียงค่อย

L2 เสียงดังปานกลาง

L3 เสียงดังมาก

M ปิด-เปิด เสียง

คำสั่ง M ในการเปิดหรือปิดเสียง

M ปิดเสียงตลอดเวลา

M1 เปิดเสียงจนกระทั่งมีสัญญาณพาห์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

M2 เปิดเสียงตลอดเวลา

M3 ปิดเสียงขณะที่มีสัญญาณพาห์และระหว่างการหมุนโทรศัพท์

^M Carriage return

^M หมายถึงการขึ้นบรรทัดใหม่ที่ใช้เมื่อจบคำสั่งในแต่ละบรรทัด

Q ให้รหัสผลลัพธ์ (result code) ทำงานหรือหยุดทำงาน ใช้สั่งให้ผลลัพธ์ที่โมเด็มจะแสดงออกมาให้เห็นบนหน้าจอคอมพิวเตอร์ทำงานหรือไม่ทำงาน เช่น CONNECT , BUSY , RING และอื่นๆ Q1 จะทำให้โมเด็มไม่แสดงค่าเหล่านี้ออกมา แต่ QO จะเป็นการยอมให้มีการแสดงค่าเหล่านี้ ปกติค่าเริ่มต้นจะเป็น Q0

&F เรียกคำสั่งดั้งเดิมขึ้นมาใช้

&V เป็นการดูค่าต่างๆ เกี่ยวกับการใช้งานโมเด็มที่มีการตั้งค่าเอาไว้ (ซึ่งเก็บโดยการใช้คำสั่ง &W) รวมทั้งเลขหมายโทรศัพท์ที่เก็บไว้ด้วย

&W Write Configuration (ใช้เก็บคำต่างๆของโมเด็ม)การใส่ &W ไว้ที่ท้ายคำสั่งของบรรทัด จะเป็นการจัดเก็บค่าซึ่งมีการเปลี่ยนแปลงไปจากเดิมยกเว้นเลขหมายโทรศัพท์ เอาไว้

%C ให้การบีบอัดข้อมูลใน n ทำงานหรือไม่ทำงาน จะใช้คำสั่งนี้ได้ก็ต่อเมื่อมีการใช้การบีบขนาดข้อมูลใน MNP-5 บริการออนไลน์บางแห่งอาจจะให้คุณทำการดิสเอเบิล MNP-5 เพื่อให้การส่งข้อมูลมีความน่าเชื่อถือมากขึ้นมากขึ้นรูปแบบการใช้คือ

AT %Cn

เมื่อ n เป็น 0 การบีบข้อมูลใน MNP-5 จะถูกทำให้หยุดทำงานเมื่อ n เป็น 1 จะเป็นการทำงาน

S รีจิสเตอร์ (Register) โมเด็มที่เป็นเฮย์สคอมแพททิเบิลต่างๆ ที่เกี่ยวข้องกับโมเด็มไว้ในหน่วยความจำที่ชื่อ S รีจิสเตอร์ โดยรีจิสเตอร์เหล่านี้จะแบ่ง S0, S1, S2 และเรียงต่อไปเรื่อยๆ จำนวนของ S รีจิสเตอร์จะขึ้นอยู่กับแต่ละโมเด็ม แต่โมเด็มส่วนใหญ่จะมี 30 หรือมากกว่านี้สิ่งที่เก็บไว้ใน S รีจิสเตอร์บางตัวคือ

S0 จะกำหนดการตอบรับของโมเด็ม ถ้าค่า S0 เป็น 0 จะดิสเอเบิลการตอบรับอัตโนมัติ ส่วนค่าอื่นๆที่ไม่ใช่ 0 จะเป็นการกำหนดจำนวนครั้งของกระดิ่งโทรศัพท์ที่ดังก่อนที่โมเด็มจะตอบรับ

S1 นับจำนวนครั้งของกระดิ่งโทรศัพท์ (ในโหมดการตอบรับอัตโนมัติ)

S2 เก็บค่ารหัสแอสกี (ASCII) ของอักษร Escape Sequence โดยค่าเริ่มต้นที่ตั้งมาจากโรงงานคือ 43 ตรงกับรหัสแอสกีของเครื่องหมาย +

S3 เก็บค่ารหัส แอสกี ขึ้นบรรทัดใหม่ โดยค่าเริ่มต้นที่มาจากโรงงานคือ 13

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

S6 แสดงจำนวนเวลาที่รอก่อนจะหมุนโทรศัพท์เป็นวินาที จุดประสงค์ที่ต้องการมีการรอสัญญาณโทรศัพท์ (สัญญาณสายว่าง) ค่าเริ่มต้นที่มาจากโรงงาน คือ 2 วินาที

S7 แสดงจำนวนเวลาเป็นวินาทีที่จะรอเพื่อจะติดต่อกับปลายทาง ถ้าหากค่านี้เท่ากับ 30 วินาที โมเด็มจะทำการรอเป็นเวลา 30 วินาทีเพื่อที่จะให้มีการติดต่อเป็นผลสำเร็จ

S8 แสดงจำนวนเวลาเป็นวินาทีที่จะหยุดสำหรับคอมม่าแต่ละอัน (,) ที่คุณใส่ไว้ในการหมุนเลขโทรศัพท์ค่าเริ่มต้นที่มีเป็น 2 วินาที

การอ่านค่า S รีจิสเตอร์ ใช้ดูรายละเอียดใน S รีจิสเตอร์ โดยใช้คำสั่ง

AT Sn?

n คือตำแหน่งของ S รีจิสเตอร์ ที่คุณต้องการทราบ โดยปกติโมเด็มจะแสดงให้เก็บเป็นตัวเลข 3 ตัว และจะมีคำว่า " OK " ตามมาในบรรทัดต่อไปด้วย ดังตัวอย่างในการใช้คำสั่งเพื่อดูรายละเอียดในรีจิสเตอร์ S3 ต่อไปนี้

ATS3?

โมเด็มจะตอบสนองดังนี้

013

OK

การเขียนลงใน S รีจิตตอร์ ใช้ในการเขียนค่าลงใน S รีจิสเตอร์โดยใช้คำสั่งดังนี้

AT Sn=value

โดย n คือตำแหน่งของรีจิสเตอร์ที่คุณต้องการเขียน และ Value คือค่าที่คุณต้องการใส่ในรีจิสเตอร์

ตัวอย่าง ในการเขียนค่า 3 ลงในรีจิสเตอร์ S0

AT S0=3

โมเด็มจะตอบสนองดังนี้

OK

ตัวอย่างรูปแบบคำสั่งที่ใช้งาน

ATA เป็นคำสั่งให้โมเด็มตอบรับสัญญาณโทรศัพท์ที่เรียกเข้ามา เมื่อโมเด็มทำคำสั่งนี้การติดต่อระหว่างปลายทางทั้งสองข้างจะเริ่มขึ้น และโมเด็มจะอยู่ในช่วงรับส่งข้อมูลไม่รับคำสั่งจากเครื่องคอมพิวเตอร์ จนกว่าจะเลิกการติดต่อกับปลายทางเสียก่อน

ตัวอย่าง การใช้งาน เช่น ATDT 2345678 9 ตามด้วย Return หรือ Enter เป็นตัวจบคำสั่ง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ATH เป็นคำสั่งให้โมเด็มวางสายโทรศัพท์ หรือ ปลดตัวออกจากสายโทรศัพท์และวางหูนั่นเอง ส่วนมากคำสั่งจะใช้เมื่อเลิกติดต่อกับปลายทางโดยสั่ง ATH เพื่อวางหูหลังจากส่งข้อมูลแล้ว ก่อนใช้คำสั่งนี้เราต้องสั่งให้โมเด็มกลับมาอยู่ในสภาวะรับคำสั่งเสียก่อน ซึ่งทำได้โดยส่งเครื่องหมาย + 3 ตัวไปให้โมเด็ม แต่ระยะห่างกันประมาณครึ่งวินาที โมเด็มจะถือว่าจบการส่งข้อมูล และรอรับคำสั่งจากเครื่องคอมพิวเตอร์ต่อไป

ATL ใช้คำสั่งควบคุมการทำงานของลำโพงภายในตัวโมเด็ม ซึ่งปรับความดังได้ 3 ระดับ นี้คือ ATL0 และ ATL1 เป็นคำสั่งรับความดังให้น้อยที่สุด ATL2 จะเป็นระดับความดังปานกลาง และ ATL3 สำหรับความดังของลำโพงมากที่สุด แต่โมเด็มบางแบบจะใช้ปุ่มหมุนปรับความดังของลำโพงภายในแทนก็ได้ ปกติเราก็ไม่ค่อยใช้คำสั่งนี้มากนัก

ATM ใช้สำหรับควบคุมการทำงานของลำโพงภายในโมเด็ม เช่น ATM0 จะสั่งไม่ให้ลำโพงภายในโมเด็มทำงานคือ เงียบตลอดเวลา ATM1 จะให้ลำโพงมีเสียงเฉพาะตอนที่ยังต่อกับปลายทางไม่ได้เท่านั้น เพื่อให้เราฟังเสียงโทรศัพท์ และสัญญาณตอบรับว่าไม่มีอะไรผิดปกติหรือเปล่า ATM2 จะสั่งให้ลำโพงมีเสียงตลอดเวลาแม้ว่าต่อกับปลายทางได้แล้วก็ตาม

ATO เป็นคำสั่งให้โมเด็มอยู่ในโหมด Online คือเปลี่ยนจากการรับคำสั่งจากเครื่องคอมพิวเตอร์มาเป็นการรับข้อมูลผ่านสายส่ง เหมือนกับการกดปุ่ม DATA บนตัวโมเด็มนั่นเอง

ATS ใช้สำหรับการเปลี่ยนแปลงค่าของ S-Register ซึ่งเป็นที่เก็บพารามิเตอร์ในการทำงานต่างๆ ของ Hayes เช่น S0 ใช้เก็บจำนวนครั้งของสัญญาณเรียกเข้า ก่อนที่โมเด็มจะทำการรับโทรศัพท์ ถ้าเราต้องการให้โมเด็มรับโทรศัพท์ที่เรียกเข้ามาโดยมีสัญญาณเรียกดัง 4 ครั้ง ก็สั่ง ATSO=4 เป็นต้น S-Register ของ Hayes แสดงในตารางประกอบว่าตัวไหนมีหน้าที่ทำอะไร นอกจากนี้คำสั่ง ATS ยังใช้เรียกดูค่าของ S-Register ได้อีกด้วยว่าค่าที่เก็บอยู่ในปัจจุบันเป็นเท่าไร โดยใช้คำสั่ง เช่น ATS0?
โมเด็มจะนำค่า ของ S0 ที่เก็บอยู่ในขณะนั้นมาแสดงให้เราดู

ATX เป็นคำสั่งโมเด็มแสดงผลลัพธ์ในการหมุนโทรศัพท์แบบต่างๆ เช่น ATX0 และ ATX1 จะเป็นคำสั่งของโมเด็มไม่ตรวจสอบ Dial Tone ก่อนหมุนโทรศัพท์ และไม่รับรู้สัญญาณที่ตอบกลับมาว่าปลายทางไม่ว่าง ATX2 จะให้โมเด็มตรวจสอบก่อนว่าสายโทรศัพท์มี Dial Tone ก่อนทำการหมุนโทรศัพท์ แต่ไม่รับรู้สัญญาณปลายทางไม่ว่าง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เหมือนเดิม ถ้าหากโมเด็มไม่ได้รับ Dial Tone จากชุมสายใน 5 นาที โมเด็มจะตอบกลับไปยังคอมพิวเตอร์ว่า " No Dial Tone " ATX3 จะให้โมเด็มรับรู้สัญญาณปลายทางไม่ว่าง ได้ และ ATX4 ซึ่งเป็นคำสั่งที่เราใช้ส่วนมาก จะให้โมเด็มตรวจสอบ Dial Tone ก่อนที่จะทำการหมุนโทรศัพท์ และรับรู้สัญญาณปลายทางไม่ว่างได้ด้วยคือเท่ากับ ATX2 รวมกับ ATX3 นั่นเอง

ภาวะคำสั่งและภาวะออนไลน์

โมเด็มจะอยู่ในภาวะใดภาวะหนึ่งคือ ภาวะคำสั่ง หรือภาวะออนไลน์ ขณะที่โมเด็มอยู่ในภาวะคำสั่งสามารถสั่งงานโมเด็มได้จากคอมพิวเตอร์ ตัวอย่างเช่น คำสั่งที่สั่งให้โมเด็มทำการหมุนหมายเลขโทรศัพท์ หรือตอบรับเมื่อกริ่งโทรศัพท์ดัง โดยอัตโนมัติคำสั่งจะถูกส่งให้กับโมเด็มไม่ใช่ถูกส่งออกไป

เมื่อการเชื่อมต่อเกิดขึ้นกับโมเด็มระยะไกล โมเด็มท้องถิ่นจะเข้าสู่ภาวะออนไลน์และจะไม่แปลงความหมายของข้อมูล ที่ถูกส่งให้กับโมเด็ม แต่จะส่งผ่านข้อมูลแทน ถ้าสัญญาณพาหะหายไป เช่น โมเด็มระยะไกลวางหูโทรศัพท์โมเด็มจะกลับสู่สภาวะคำสั่ง การอออกจากภาวะออนไลน์โดยไม่ต้องตัดการเชื่อมต่อ ทำได้โดยรอคอยเป็นช่วงเวลา Guard time (ค่าปริยายคือ 1 วินาที) พิมพ์ Escape Command (ค่าปริยายคือ +++) และรอสัก 1 นาที ก่อนที่ข้อมูลจะถูกแปลงเป็นคำสั่ง

คำสั่งที่ส่งโดยคอมพิวเตอร์ไปยังโมเด็มสามารถถูกส่งได้โดยซอฟแวร์การสื่อสาร หรือพิมพ์เข้าไปจากแป้นพิมพ์ โดยเอาท์พุทของแป้นพิมพ์ถูกเปลี่ยนทิศทางไปยังพอร์ตอนุกรม คำสั่งที่ถูกต้องส่งในภาวะคำสั่งควรถูกส่งด้วยเจ็ดบิตข้อมูล หนึ่งบิตพาริตี้หรือแปดบิตข้อมูล ไม่มีพาริตี้บิต ควรจะมีหนึ่งบิตจบ ถ้าไม่ได้สื่อสารด้วยอัตรา 110 bps ซึ่งที่อัตรานั้นควรใช้สองบิตจบ โมเด็มที่เข้ากับ Hayes สามารถตรวจสอบอัตราการส่งได้เองโดยอัตโนมัติ

รหัสผลลัพธ์

เมื่อโมเด็มได้รับคำสั่ง โมเด็มจะส่งผลลัพธ์ กลับมารหัสอาจอยู่ในรูปของข่าวสารข้อความหรือรหัสตัวเลข ถ้าคุณกำลังควบคุมโมเด็มผ่านทางซอฟแวร์ การใช้รหัสตัวเลขจะเหมาะสมกว่า ถ้าคุณกำลังควบคุมโมเด็มจากแป้นพิมพ์ ข่าวสารข้อความก็น่าจะดี ชนิดของรหัสผลลัพธ์สามารถเลือกได้โดยการใช้คำสั่งหรือการตั้งค่าสวิทช์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รหัสตัวเลข	รหัสข้อความ	ความหมาย
0	OK	คำสั่งถูกนำไปปฏิบัติ
1	CONNECT	เชื่อมต่อที่ 0-300 bps*
2	RING	ตรวจพบสัญญาณกริ่ง
3	NO CARRIER	ไม่พบโมเด็มระยะไกล
4	ERROR	ความผิดพลาดในคำสั่ง
5	CONNECT 1200	เชื่อมต่อที่ 1200 bps*
6	NO DIALTONE	ไม่พบสัญญาณให้หมุนบนสายโทรศัพท์ท้องถิ่น
7	BUSY	สายไม่ว่าง
8	NO ANSWER	ไม่มีการตอบสนองจากโมเด็มระยะไกล
9	CONNECT 2400	เชื่อมต่อที่ 2400 bps*
* ในโมเด็ม 1200 bps ถ้า X0 (ค่าโดยปริยาย) ถูกเซต รหัสผลลัพธ์ 1 ถูกใช้ สำหรับการเชื่อมต่อ 0-300 bps และ 1200 bps เพื่อประกันความเข้ากันได้ กับซอฟต์แวร์ที่เขียนมาสำหรับโมเด็ม 300 bps		

รูปที่ 2.30 รหัสคำสั่งของโมเด็ม Hayes

สภาวะการตอบรับโทรศัพท์

โมเด็มสามารถถูกตั้งให้ตอบรับโทรศัพท์อัตโนมัติ โดยสามารถกำหนดจำนวนครั้งของเสียงกริ่งก่อนที่โมเด็มจะตอบรับโดยการตั้งค่ารีจิสเตอร์ S0 เช่น ถ้าต้องการให้โมเด็ม รับหลังจากเสียงกริ่งครั้งที่ห้า ให้พิมพ์ว่า

ATS0=5

ค่าโดยปริยายคือ S0=0 ซึ่งบอกโมเด็มไม่ให้ตอบรับ มีสวิตช์ซึ่งสามารถทำให้ค่าปริยายเป็น S0=1 ซึ่งหมายความว่า มันจะตอบรับเมื่อมีเสียงกริ่งหนึ่งครั้ง ถ้าไม่สั่งเป็นอย่างอื่น

เมื่อโมเด็มตอบรับโทรศัพท์ มันจะส่งสัญญาณพาหะและรอคอยการตอบสนอง ถ้าไม่มีสัญญาณพาหะส่งกลับมาภายในช่วงเวลาที่ตั้งโดยรีจิสเตอร์ S7 มันจะวางสาย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การตรวจสอบอัตราบอดที่เข้ามาจะเป็นไปโดยอัตโนมัติ และโมเด็มจะส่งรหัสลัพธ์เพื่อบอกอัตราบอด คอมพิวเตอร์ของคุณต้องรู้จักรหัสผลลัพธ์นี้และปรับอัตราบอดของมันเพื่อจะสนองอัตราบอดที่เข้ามาแตกต่างกันโดยอัตโนมัติ

จะเห็นว่าคำสั่งตามมาตรฐานของ Hayes ใช้ง่ายและเข้าใจคำสั่งง่ายกว่ามาก เนื่องจากใช้ตัวอักษรสื่อความหมายได้ดีกว่า ทำให้จำคำสั่งได้ง่าย นอกจากนี้ผลลัพธ์ที่โมเด็มตอบกลับมายังคอมพิวเตอร์ตามแบบของ Hayes ยังเข้าใจได้ง่ายอีกด้วย เพราะว่าใช้ตัวอักษรภาษาอังกฤษตัวเต็มผู้ใช้อ่านแล้วเข้าใจทันที เช่น CONNECT , BUSY , NO DIAL TONE

2.4 พอร์ตอนุกรม (SERIAL PORT)

เนื่องจากในปัจจุบันมีการใช้งานตามมาตรฐานการเชื่อมต่อแบบ RS-232C กันอย่างแพร่หลาย ดังนั้นผู้เขียนจึงขอกล่าวรายละเอียดเฉพาะมาตรฐานการเชื่อมต่อแบบ RS-232C เท่านั้น ซึ่งเป็นมาตรฐานที่ถูกกำหนดโดย EIA ซึ่งเป็นองค์กรอุตสาหกรรมอิเล็กทรอนิกส์ของอเมริกา โดยแบ่งการเชื่อมต่อออกเป็น 2 ลักษณะ คือ DTE (DATA TERMINAL EQUIPMENT) และ DCE (DATA COMMUNICATION EQUIPMENT) ซึ่งโดยปกติ DTE จะต้องต่อเข้ากับ DCE เสมอ เช่น การต่อเครื่องคอมพิวเตอร์ (อุปกรณ์ DTE) เข้ากับโมเด็ม (อุปกรณ์ DCE) เป็นต้น

พอร์ตอนุกรม RS-232C จะเป็นพอร์ตของเครื่องคอมพิวเตอร์ที่มีขาต่อ (CONNECTOR) ทั้งประเภท 9 และ 25 ขา และเราเรียกกันว่าพอร์ต COM1 และ COM2 นั่นเอง ในความเป็นจริงพอร์ตอนุกรมไม่ได้ถูกควบคุมโดยตรงจาก CPU บนเมนบอร์ด แต่การสื่อสารทั้งหมดจะถูกจัดการโดยชิป UART (UNIVERSAL ASYNCHRONOUS RECEIVER/TRANSMITTER) อีกทีหนึ่ง ซึ่งในปัจจุบันเบอร์ที่ใช้กันมากที่สุดก็คือเบอร์ 16550C ซึ่งเป็นเวอร์ชั่นที่ได้รับการแก้ไขข้อผิดผลาดแล้ว ซึ่งชิป UART นี้จะทำหน้าที่ในการรับและส่งข้อมูลดังต่อไปนี้

2.4.1 การส่งข้อมูลและการรับข้อมูล

การส่งข้อมูล

- รับตัวอักษรจากเครื่องคอมพิวเตอร์
- แปลงตัวอักษรให้เป็นสายข้อมูลแบบบิต (เราเรียกว่าขบวนการ SERIALIZATION)
- สร้างเฟรมข้อมูลโดยการเพิ่มบิตที่จำเป็นสำหรับการสื่อสารและการตรวจสอบ เช่น บิต START , STOP และ PARITY เป็นต้น

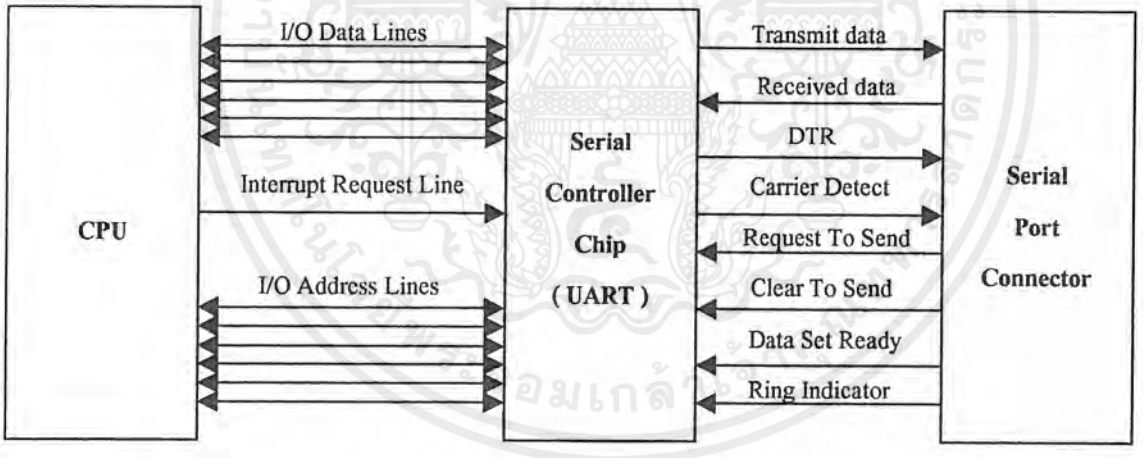
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- ส่งผ่านเฟรมข้อมูลที่สร้างขึ้นมาแล้วจากขั้นตอนที่ผ่านมา ด้วยความเร็วของโมเด็มหรือพอร์ตอนุกรม (BAUD RATE)
- แสดงสถานะความพร้อมที่จะรับข้อมูลตัวอักษรถัดไปให้กับเครื่องคอมพิวเตอร์

การรับข้อมูล

- รับตัวอักษรจากอินเทอร์เฟซ
- ตรวจสอบความถูกต้องของเฟรมข้อมูลตามมาตรฐานเฟรมที่กำหนด โดยถ้าหากเฟรมข้อมูลมีรูปแบบที่ไม่ถูกต้องก็จะมีการแจ้งข้อผิดผลาดทันที
- ตรวจสอบความถูกต้องของพาริตี้
- แปลงสายข้อมูลแบบบิตให้เป็นตัวอักษร
- ส่งตัวอักษรให้กับเครื่องคอมพิวเตอร์
- แสดงสถานะความพร้อมที่จะรับข้อมูลตัวอักษรถัดไปให้กับอินเทอร์เฟซ

สำหรับการเชื่อมต่อสายสัญญาณต่างๆ ระหว่าง CPU ของเมนบอร์ดของเครื่องคอมพิวเตอร์กับพอร์ตอนุกรมนั้น จะต้องกระทำผ่านทางชิป UART ซึ่งจะมีวิธีการเชื่อมต่อดังในรูปที่ 2.31



รูปที่ 2.31 แสดงการเชื่อมต่อสายสัญญาณระหว่าง CPU ของเครื่องคอมพิวเตอร์กับพอร์ตอนุกรม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จากรูปที่ 2.31 ผู้อ่านจะเห็นว่าการเชื่อมต่อระหว่าง UART กับพอร์ตอนุกรมนั้น มีสายสัญญาณมากมายที่ช่วยให้การสื่อสารมีความถูกต้องมากขึ้น โดยที่สายสัญญาณแต่ละเส้นมีความหมายดังต่อไปนี้

- TRANSMITTED DATA (TD)
  เป็นวงจรที่สร้างสัญญาณ TRANSMITTED DATA ซึ่งถูกส่งจาก DTE ไปยัง DCE ซึ่งสัญญาณที่ส่งออกมาจะเป็นโค้ดคำสั่งของโมเด็มหรือข้อมูลก็ได้
- RECEIVED DATA (RD)
  เป็นวงจรที่สร้างสัญญาณ RECEIVED DATA ซึ่งถูกส่งมาจาก DCE ไปยัง DTE ซึ่งสัญญาณที่ส่งออกมาอาจจะเป็นโค้ดคำสั่งของโมเด็มหรือข้อมูลก็ได้ ซึ่งสัญญาณที่มีทิศทางการส่งตรงข้ามกับสัญญาณ TRANSMITTED DATA
- DATA TERMINAL READY (DTR)
  สัญญาณ DTR จะถูกส่งจาก DTE ไปยัง DCE เพื่อเป็นการแจ้งความพร้อมในการสื่อสารให้โมเด็มได้ทราบ โดยถ้าหากโมเด็ม (อุปกรณ์ DCE) มีความสามารถในการตอบรับแบบอัตโนมัติ (AUTOMATICALLY ANSWER) โมเด็มก็จะสามารถตอบรับได้เฉพาะเมื่อสัญญาณ DTR อยู่ในสถานะ ON เท่านั้น
- CARRIER DETECT (CD)
  สัญญาณ CD จะถูกส่งจาก DCE ไปยัง DTE เพื่อเป็นการแจ้งว่าโมเด็มอยู่ในสถานะกำลังติดต่อกับโมเด็มตัวอื่น หรือโมเด็มกำลังได้รับสัญญาณที่พร้อมสำหรับการติดต่อสื่อสาร สำหรับสัญญาณ CARRIER DETECT นี้ผู้อ่านสามารถเรียกอีกชื่อได้ว่า RECEIVED LINE SIGNAL DETECTOR
- REQUEST TO SEND (RTS)
  สัญญาณ RTS จะถูกส่งจาก DTE ไปยัง DCE โดยเมื่อสัญญาณ RTS อยู่ในสถานะ ON ก็หมายถึงเครื่องคอมพิวเตอร์พร้อมที่จะรับข้อมูลจากโมเด็ม และในทางกลับกันถ้าหากสัญญาณ RTS อยู่ในสถานะ OFF ก็หมายถึง เครื่องคอมพิวเตอร์ไม่พร้อมที่จะรับข้อมูลจากโมเด็ม
- CLEAR TO SEND (CTS)
  สัญญาณ CTS จะถูกส่งจาก DCE ไปยัง DTE ซึ่งเป็นสัญญาณที่ทำหน้าที่ตรงข้ามกับสัญญาณ RTS โดยเมื่อสัญญาณ CTS อยู่ในสถานะ ON ก็หมายถึง โมเด็ม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

พร้อมที่จะรับข้อมูลจากเครื่องคอมพิวเตอร์และในทางกลับกันถ้าหากสัญญาณ CTS อยู่ในสถานะ OFF ก็หมายถึง โมเด็มไม่พร้อมที่จะรับข้อมูลจากเครื่องคอมพิวเตอร์

- DATA SET READY (DSR)

  สัญญาณ DSR จะถูกส่งจาก DCE ไปยัง DTE เพื่อเป็นการแจ้งความพร้อมในการสื่อสารจากโมเด็มให้กับเครื่องคอมพิวเตอร์ได้ทราบ โดยสัญญาณ DSR จะอยู่ในสถานะ ON ก็ต่อเมื่อโมเด็มได้รับสัญญาณ DTR เท่านั้น

- RING INDICATOR (RI)

  สัญญาณ RI จะถูกส่งจาก DCE ไปยัง DTE เพื่อเป็นการแจ้งให้เครื่องคอมพิวเตอร์ทราบว่า โมเด็มกำลังได้รับสัญญาณกระดิ่ง (RING SIGNAL) จากโมเด็มตัวอื่น โดยสัญญาณ RI จะอยู่ในสถานะ ON โมเด็มได้รับสัญญาณกระดิ่งและ OFF เมื่อโมเด็มไม่ได้รับสัญญาณกระดิ่ง เนื่องจากโมเด็มรุ่นใหม่ ๆ ในปัจจุบันมักจะมีความสามารถในการตอบรับแบบอัตโนมัติ (AUTOMATICALLY ANSWER) ดังนั้นจึงไม่มีความจำเป็นต้องใช้สัญญาณ RI อีกต่อไป

2.4.2 รูปแบบของเฟรมข้อมูล (DATA FORMAT)

ในการสื่อสารระหว่างโมเด็ม 2 ตัวนั้น โมเด็มทั้งสองจะต้องมีความเข้าใจถึงรูปแบบของเฟรมข้อมูลที่ตรงกัน ซึ่งการส่งข้อมูลแบบ ASYNCHRONOUS นั้นนอกจากข้อมูลที่จะต้องส่งผ่านแล้วโมเด็มยังต้องจัดการช่วงจังหวะในการส่งหรือรับข้อมูลระหว่างกันอีกด้วย โดยที่เฟรมข้อมูลในการส่งหรือรับจะประกอบด้วยบิตข้อมูลที่มีความหมายดังต่อไปนี้

- บิตข้อมูล (DATA BIT)

  เมื่อชิป UART ได้รับตัวอักษรที่ส่งมาจากเครื่องคอมพิวเตอร์แล้ว ก็ต้องทำการแปลงด้วยตัวอักษรดังกล่าวให้เป็นสายข้อมูลชนิดบิตที่มีความยาวตั้งแต่ 5 ถึง 8 บิต ซึ่งเราเรียกขบวนการแปลงตัวอักษรให้เป็นสายข้อมูลชนิดบิตนี้ว่าขบวนการ SERIALIZATION จากนั้นโมเด็มก็จะทำการส่งแต่ละบิตไปยังโมเด็มปลายทาง โดยจะเริ่มต้นส่งจากบิตที่มีนัยสำคัญต่ำสุด (LEAST SIGNIFICANT BIT) ไปยังบิตที่มีนัยสำคัญสูงสุด (MOST SIGNIFICANT BIT)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- บิตเริ่มต้นข้อมูล (START BIT)

  ในการส่งข้อมูลแบบ ASYNCHRONOUS นั้น เราจะต้องมีวิธีการบอกโมเด็มให้ทราบถึงจุดเริ่มต้นของข้อมูลที่ต้องการส่ง ดังนั้นก่อนหน้าข้อมูลในทุก ๆ เฟรมจะต้องถูกนำหน้าด้วยบิตเริ่มต้นข้อมูล (START BIT) เสมอ

- บิตสิ้นสุดข้อมูล (STOP BIT)

  ในการส่งข้อมูลแบบ ASYNCHRONOUS นั้น ในกรณีที่บิตเริ่มต้นข้อมูลเกิดการสูญหายในระหว่างการส่งโมเด็มก็จะไม่สามารถทราบถึงจุดสิ้นสุดของสายข้อมูลบิตได้เลย นอกเสียจากโมเด็มจะตรวจพบบิตเริ่มต้นข้อมูลใหม่อีกครั้งเท่านั่น ดังนั้นจึงมีการเพิ่มบิตสิ้นสุดข้อมูล ต่อท้ายทุก ๆ ข้อมูลของแต่ละตัวอักษร เพื่อแจ้งการสิ้นสุดของสายข้อมูลบิต โดยที่การเรียงกันของบิตเริ่มต้นข้อมูล สายข้อมูลบิต และบิตสิ้นสุดข้อมูลนี้เราเรียกว่า เฟรมของข้อมูล (DATA FRAME)

- บิตพาริตี้ (PARITY BIT)

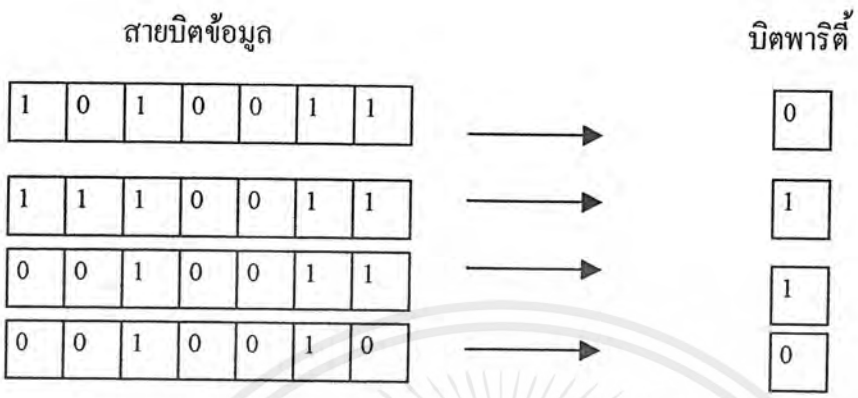
  เนื่องจากการส่งผ่านข้อมูลทางสายโทรศัพท์นั้น สามารถเกิดสัญญาณรบกวนได้ง่าย ด้วยเหตุนี้จึงเป็นไปได้มากที่สถานะของแต่ละบิตของข้อมูลที่ถูกส่งจะมีการเปลี่ยนแปลง เช่น จากบิต 0 เป็น บิต 1 เป็นต้น ในระหว่างการส่งผ่านข้อมูล ด้วยเหตุนี้จึงมีการคิดค้นวิธีที่ง่ายที่สุดที่ช่วยในการตรวจสอบความถูกต้องของสถานะของบิตข้อมูล ซึ่งเรียกกันว่า การตรวจสอบค่าพาริตี้ (PARITY CHECK) โดยในการส่งผ่านข้อมูลด้วยโปรโตคอล START / STOP นั้น ในการส่งตัวอักษรข้อความทั่วไปจะใช้เพียง 7 บิต ข้อมูลเท่านั้น ดังนั้นจึงมีการเพิ่มบิตพาริตี้ต่อสายเฟรมข้อมูลที่ถูกสร้างขึ้นมา ตามที่กล่าวมาแล้วในส่วนของบิตสิ้นสุดข้อมูล เพื่อตรวจสอบสถานะของผลบวกของบิตที่เป็น 1 ของสายบิตข้อมูลของแต่ละตัวอักษรสำหรับหลักการในการคำนวณค่าของบิตพาริตี้มีดังนี้

  1. ถ้าหากจำนวนของบิตที่มีค่าเท่ากับ 1 ของสายบิตข้อมูลมีค่าเป็นเลขคู่ (EVEN NUMBER) บิตพาริตี้ จะมีค่าเท่ากับ 0
  2. ถ้าหากจำนวนของบิตที่มีค่าเท่ากับ 1 ของสายบิตข้อมูลมีค่าเป็นเลขคี่ (ODD NUMBER) บิตพาริตี้ จะมีค่าเท่ากับ 1

เพื่อความเข้าใจขอให้ผู้อ่านดูตัวอย่างในรูปที่ 2.32 จะเห็นว่าถ้าหากจำนวนของบิตที่มีค่าเท่ากับ 1 ของสายบิตข้อมูลมีค่าเป็นเลขคู่ เช่น 2 , 4 หรือ 6 บิตพาริตี้จะมีค่าเท่ากับ 0 แต่ในทางกลับกัน ถ้าหากจำนวนบิตที่มีค่าเท่ากับ 1 ของสายบิตข้อมูลมีค่าเป็นเลขคี่ เช่น 1 , 3 , 5 หรือ 7 บิตพาริตี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จะมีค่าเท่ากับ 1



รูปที่ 2.32 แสดงการคำนวณบิตพาริตี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 3
การสร้างและประกอบโครงงาน

3.1 LADDER DIAGRAM

การควบคุมแบ่งออกเป็น 3 MODE คือ

1. MODE นับค่า คือ มีไว้สำหรับเริ่มต้นตั้งแต่เริ่มเดินเครื่อง PLC ก็จะมี SWITCH ไว้สำหรับตรวจสอบกระป๋อง เมื่อกระป๋องถูกเอาออกก็จะเริ่มลดค่ากระป๋องที่มีอยู่ในตู้ ลดลงทันทีโดยจะนับทีละกระป๋อง ดังนั้น MODE นี้ก็สามารถที่จะทราบค่ากระป๋องทั้งหมดที่เหลืออยู่ในตู้ใส่กระป๋อง

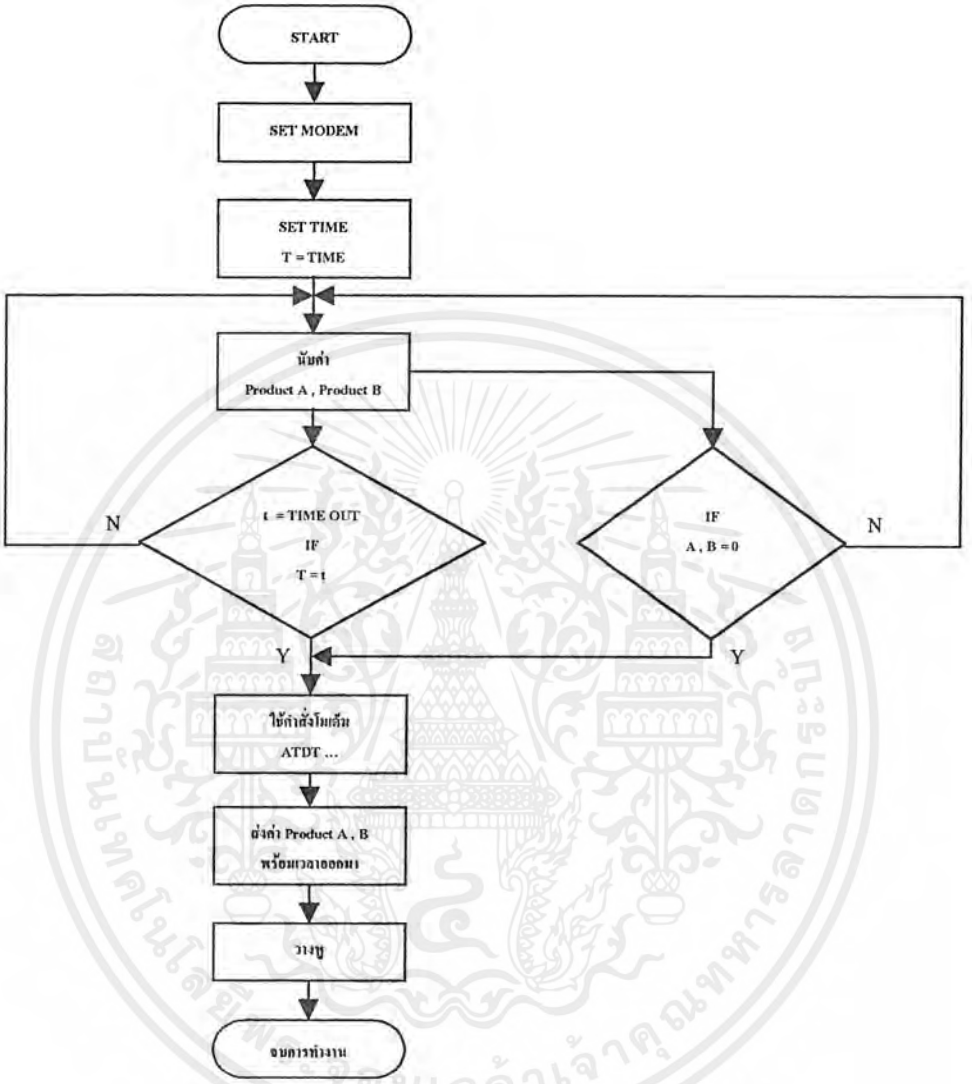
2. MODE เวลาและการตั้งเวลา คือ มีไว้สำหรับตั้งค่าเวลาให้ตรงกันกับเวลาที่ต้องการและยังสามารถที่จะตั้งเวลาให้เครื่องทำการส่งค่าน้ำทั้ง 2 ชนิดออกมาได้ตามต้องการ

3. MODE PRINT ค่าใน REGISTER ออกมาแสดง คือ มีไว้สำหรับส่งค่าใน REGISTER ที่ต้องการออกในรูปของข้อมูลเป็นแบบ ACSII โดย COMPUTER สามารถเข้าใจและสื่อสารข้อมูลแบบนี้ได้อย่างมีประสิทธิภาพ

ดังนั้นเมื่อรวมทั้ง 3 MODE เข้าด้วยกันก็จะสามารถที่จะสั่งให้ PLC ทำงานได้ตามต้องการดัง FLOW CHART ต่อไปนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

FLOW CHART แสดงการทำงานของ LADDER DIAGRAM



จาก FLOW CHART มีหลักการทำงานดังต่อไปนี้ โดยเริ่มจากการ SET MODEM แล้ว SET TIME โดยให้ตัวแปร T = TIME แล้วเริ่มนับค่า PRODUCT A และ B ในกรณีที่ PRODUCT A หรือ B มีค่าเท่ากับศูนย์ หรือค่า TIME OUT (t) มีค่าเท่ากับ T แล้วให้ใช้คำสั่งโมเด็มหมุนเบอร์โทรศัพท์ปลายทางจากนั้นก็ส่งค่า PRODUCT A และ B พร้อมเวลาออกมา แล้วทำการวางหู จบการทำงาน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.2 Modbus Protocal

Modbus Protocal เป็นโปรโตคอลในการสื่อสารข้อมูลระหว่าง Computer กับ PLC , PLC กับ PLC หรือระบบอื่นๆของ Modicon ซึ่งปัจจุบันนอกจากจะใช้กับ PLC แล้วยังใช้ได้กับอุปกรณ์อื่นๆอีก เช่น SCADA , FLOW COMPUTER เป็นต้น ทั้งนี้เพราะใช้งานง่ายและแพร่หลายกว่าโปรโตคอลอื่นๆ

Modbus เป็นระบบ Master-Slave โดยมาสเตอร์ 1 ตัว สามารถต่อเข้ากับสลาฟ ได้ 246 ตัว และมาสเตอร์เท่านั้นที่จะส่งคำถาม (Request Query) ไปยังสลาฟและสลาฟจะส่งคำตอบกลับมา

<u>การส่งข้อมูลระหว่าง Master-Slave</u>

1. มาสเตอร์จะส่งคำถาม ซึ่งมีรูปแบบดังนี้

Slave ID	Function Code	Start Addr (HO)	Start Addr (LO)	No.of Point (HO)	No.Of Point (LO)	CRC/LRC

2. สลาฟ (PLC) ที่มี ID ตรงกับที่มาสเตอร์ส่งคำถาม (Request) มา (จะมีเพียงตัวเดียวเท่านั้น) จะส่งคำตอบกลับไปยังมาสเตอร์ ซึ่งมีรูปแบบทั่วไปดังนี้

Slave ID	Function Code	Byte Count	……Data……..	CRC/LRC

<u>ตัวอย่างฟังก์ชั่นของ Modbus</u>

Function Code	Description
01	Read Coil status
02	Read input status
03	Read multiple registers
04	Read input registers
05	Write Coil
06	Write single register
15	Force multiple registers
16	Write multiple registers
23	read/write registers

etc.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ความหมายของแต่ละ Field มีดังนี้

1. SLAVE ID หมายถึง หมายเลขประจำตัวของอุปกรณ์ที่สลาฟ ซึ่งจะไม่ซ้ำกัน
2. FUNCTION CODE หมายถึง หมายเลขฟังก์ชั่นของ Modbus ใช้บอก Slave ว่า มาสเตอร์ต้องการทำอะไร หรือต้องการข้อมูลแบบใดจากสลาฟ ดังตัวอย่างฟังก์ชันข้างบน
3. STARING ADDR. (HO ,LO) หมายถึง ตำแหน่งเริ่มต้นของข้อมูลที่ต้องการอ่าน (HO ย่อมาจาก High order , LO ย่อมาจาก Low order)
4. NO.OF POINT (HO ,LO) หมายถึง จำนวนข้อมูลที่ต้องการจะนับจากตำแหน่งเริ่มต้นที่ระบุไปแล้ว
5. BYE COUNT หมายถึง จำนวนข้อมูลที่ส่งมาให้ว่ามีทั้งสิ้นกี่ไบต์ (จะเป็นส่วนของคำตอบบางฟังก์ชัน)
6. CRC / LRC หมายถึง ข้อมูลที่เพิ่มเข้าไปในการส่งข้อมูลเพื่อใช้ตรวจสอบความผิดพลาด

โหมดการส่งข้อมูล

การส่งข้อมูลทาง Modbus ทำได้สองแบบคือโหมด ASCII และโหมด RTU ซึ่งทั้งสองแบบมีข้อแตกต่างกันดังในตาราง

ตารางความแตกต่างระหว่างโหมด ASC กับ RTU

Description	ASCII (7 Bit)	RTU (8 Bit)
Coding	ASCII Code 0-9, A-F	8 Bit Binary
Number of Bits Per Character		
* Start bits	1	1
* Data bits	1	1
* Parity	1	1
* Stop bits	1 or 2	1 or 2
Error Check	LRC	CRC

สำหรับของดีของโหมด RTU คือ จำนวนไบต์น้อยกว่าทำให้ส่งข้อมูล (Data) ได้เร็ว ส่วนข้อดีของโหมด ASCII คือตรวจสอบได้ง่าย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ERROR DETECTION

ปกติความผิดพลาด (Error) จะมีอยู่สองลักษณะคือ Communication Error และ Operation Error ในกรณี Communication Error ถ้าสลาฟพบว่ามีข้อผิดพลาดมันจะไม่ส่งคำตอบ (Response) มาให้มาสเตอร์ และมาสเตอร์ก็จะส่งคำถามไปให้ใหม่ เมื่อมันรอคำตอบจนถึงเวลาที่ตั้งไว้ หรือเมื่อพบว่ามีความผิดพลาด สลาฟอาจจะส่งข้อความแจ้งความผิดพลาดกลับมาให้มาสเตอร์ก็ได้

การตรวจสอบความผิดพลาดมี 2 วิธี

1. Parity จะใช้วิธีเพิ่มจำนวนบิต 1 บิต เข้าไปในเฟรมของข้อมูลในแต่ละไบต์ที่ส่งไป อาจเป็น 1 หรือ 0 แล้วแต่ชนิดของพาริตี้ที่ใช้ ซึ่งวิธีนี้จะตรวจสอบได้เฉพาะความผิดพลาดที่เกิดขึ้นเพียง 1 บิต เท่านั้น ถ้าความผิดพลาดเกิดขึ้นมากกว่า 1 บิต จะไม่สามารถบอกได้

2. Redundancy Check วิธีนี้สามารถตรวจสอบความผิดพลาดที่เกิดขึ้นหลายบิตได้ มีอยู่ด้วยกันสองแบบคือ Cyclical Redundancy Check (CRC) ซึ่งใช้กับโหมด RTU และแบบ Longitudinal Redundancy check (LRC) ใช้กับโหมด ASCII

ในการส่งข้อมูลระหว่างกัน ตัวส่งจะส่งข้อมูลพร้อมกับ CRC/LRC ออกไป ส่วนตัวรับก็จะรับข้อมูล และคำนวณ CRC/LRC ออกมา แล้วเปรียบเทียบกับ CRC/LRC ที่ส่งมาว่าตรงกันหรือไม่ ถ้าไม่ตรงแสดงว่ามีการผิดพลาด

2.1 Cyclical Redundancy Check (CRC) ที่ได้จะมีขนาด 16 บิต และมีวิธีการคำนวณดังนี้

1. โหลดค่าคงที่ FFFFH (16 บิต) accumulator (ACC)
2. นำข้อมูล 8 บิต (ที่จะส่ง) มา EX-OR
3. Shift Right 1 บิต
4. ถ้าบิตที่ Shiff มาเป็น 1 ให้นำ A001H มา EX-OR กับ ACC แต่ถ้าบิตที่ Shiff มาเป็น 0 ไม่ต้องทำอะไร
5. ถ้ายังไม่ครบ 8 บิต ให้กลับไปทำข้อ 3
6. ใช้ค่าผลลัพธ์ที่ได้ใน ACC เป็นตัวตั้งแทน FFFFH ในการคำนวณ สำหรับข้อมูล 8 บิต ตัวต่อไปทำเช่นนี้จนครบทุกไบต์ของแพกเกจ

หมายเหตุ การคำนวณ CRC จะใช้เฉพาะข้อมูล 8 บิตเท่านั้น โดยบิต Start , Parity , Stop จะไม่นำมาคำนวณ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.2 Longitudinal Redundancy Check มีขนาด 8 บิต มีขั้นตอนการคำนวณดังนี้

Transmitter

1. หาค่าผลรวม (SUM) ของข้อมูลทุกไบต์ที่จะส่งไป
2. นำผลลัพธ์จากข้อ 1 มาหาค่า Complement อันดับ 2 จะได้เป็น LRC ส่งไปพร้อมกับข้อมูล

Receiver

1. รับค่าทั้งหมดมาแล้วหาผลรวม (SUM รวม LRC ด้วย)
2. ถ้าผลลัพธ์ที่ได้เป็น 00H แสดงว่าไม่มีการผิดพลาด

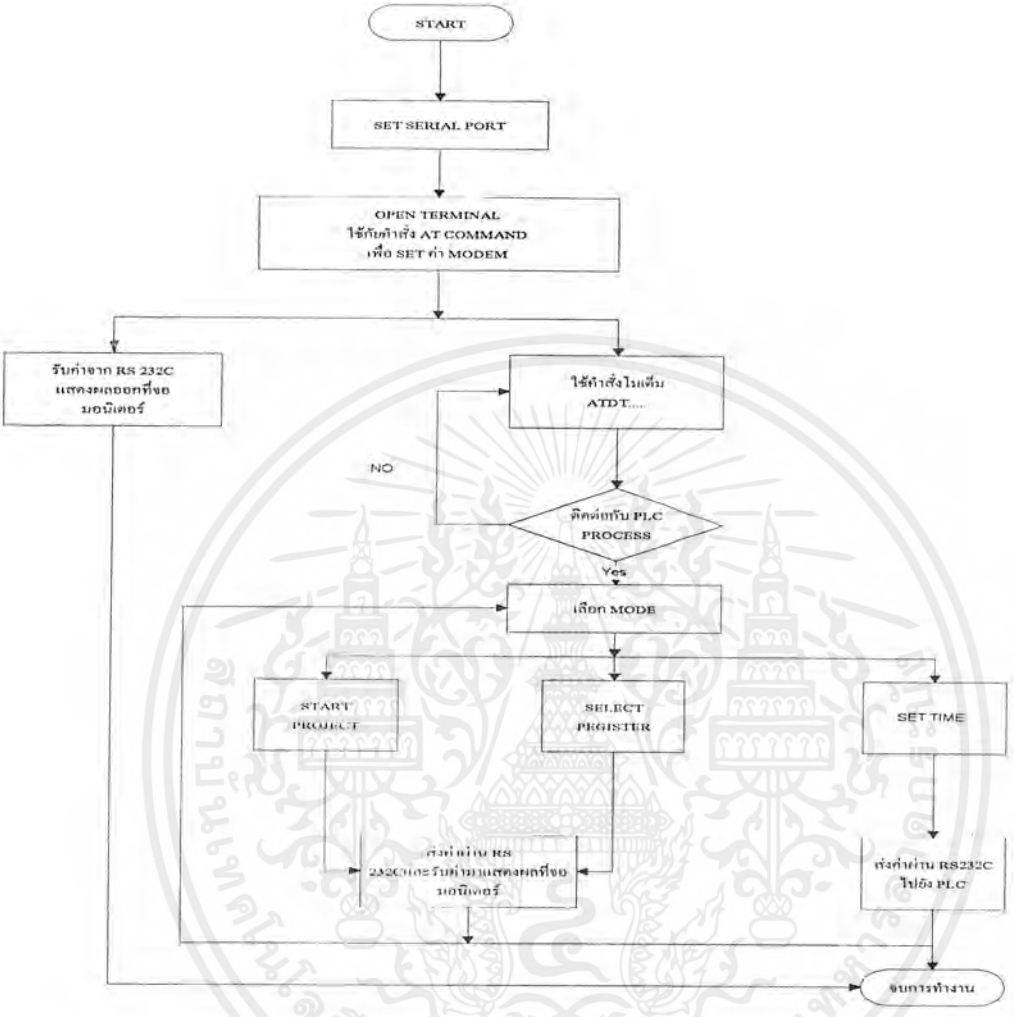
3.3 COMMUNICATION PROGRAM และ GRAPHICS DISPLAY

เป็น SOFTWARE ที่เขียนขึ้นมาจาก MICROSOFT VISAUL BASIC FOR WINDOWS VERSION 5.0 ทำหน้าที่ ติดต่อกับโมเด็ม และรับค่าที่ได้แสดงผลทางจอมอนิเตอร์ โดยลักษณะของโปรแกรมเป็นลักษณะของเทอมินัล เมื่อติดต่อกับปลายทางได้แล้ว คอมพิวเตอร์จะส่ง PROTOCAL ไปยัง PLC ซึ่ง PLC จะส่ง PROTOCAL ตอบกลับมา จากนั้น คอมพิวเตอร์ ก็จะทำการตรวจสอบ PROTOCAL ที่ส่งมาว่าถูกต้องหรือไม่ ถ้าถูกต้อง ก็จะทำการดึงค่าจาก PROTOCAL หลังจากแสดงผลออกจากทางจอมอนิเตอร์ ถ้า PROTOCAL ไม่ถูกต้อง ก็รอรับ ค่าใหม่

ในส่วนรายละเอียด ของ SOFT WARE ได้แสดงไว้ในผนวก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

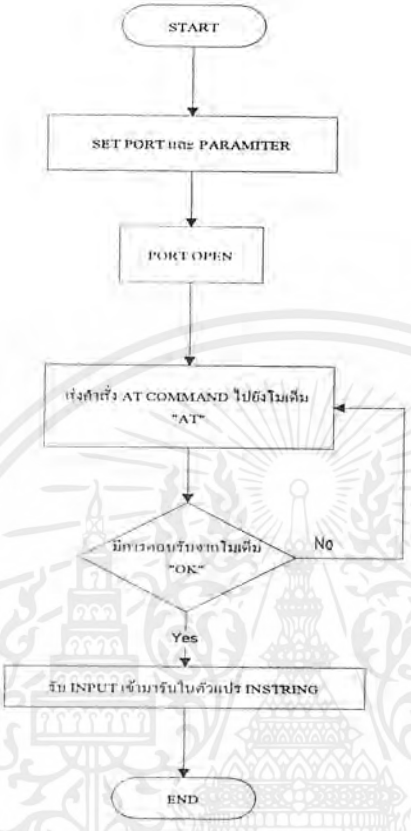
FLOW CHART แสดงการทำงานของโปรแกรมการติดต่อสื่อสาร



จาก FLOW CHART มีหลักการทำงานดังต่อไปนี้ โดยเริ่มจากทำการ SET SERIAL PORT แล้ว OPEN TERMINAL ใช้คำสั่ง AT COMMAND เพื่อ SET ค่าให้ MODEM แล้วเลือกว่าจะรับค่าจาก RS-232C และแสดงผลที่จอมอนิเตอร์ หรือจะใช้คำสั่งให้โมเด็มหมุนเบอร์ปลายทาง ถ้าใช้คำสั่งให้โมเด็มหมุนเบอร์ปลายทาง (ATDT .....) กรณีที่ติดต่อไม่ได้ ให้ทำการติดต่อใหม่ ถ้าติดต่อได้แล้ว ก็ทำการเลือก MODE โดยถ้าทำการเลือก MODE START PROJECT หรือ MODE SELECT REGISTER จะเป็นการส่งค่าผ่าน RS-232C และรับค่ามาแสดงผลที่จอมอนิเตอร์ หรือถ้าเลือก MODE SET TIME ก็จะเป็นการส่งค่าผ่าน RS-232C ไปยัง PLC แล้วจบการทำงาน (ลักษณะจะเป็นไปตาม FLOW CHART)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

FLOW CHART แสดงการทำงานของโปรแกรมที่ใช้ติดต่อกับโมเด็ม



จาก FLOW CHART มีหลักการทำงานดังต่อไปนี้ โดยเริ่มจากการ SET PORT และ ค่า PARAMITER จากนั้นก็ทำการเปิด PORT แล้วส่งคำสั่ง AT COMMAND ไปยังโมเด็ม ถ้าไม่มีการตอบรับจากโมเด็ม ก็ส่งคำสั่ง AT COMMAND ไปยังโมเด็มอีก จนกว่าจะมีการตอบรับจากโมเด็ม แล้วนำค่าที่ได้ เข้ามาไว้ในตัวแปร INSTRING แล้วจบการทำงาน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตัวอย่าง โปรแกรมที่ใช้ติดต่อกับโมเด็ม โดยใช้ MICROSOFT VISUAL BASIC VERSION 5.0

```
Private Sub Form_Load()
        'Use COM2.
        MSComm1.CommPort = 2
        '2400 Baud,Even parity,7 Data Bit and 1 Stop  Bit
        MSComm1.Setting = "2400,E,7,1"
        'Tell the Control to read entire buffer when Input is used
        MSComm1.InputLen = 0
        'Open the port
        MSComm1.PortOpen = True
        'Send the attention command to the modem.
        MSComm1.Output = "AT" + Chr$(13)
        'Wait for data to come back to the serial port
        Do
                Dummy = DoEvents()
        Loop Until MSComm1.InBufferCount >=2
        'Read the "OK" response data in the serial port
        InString$ = MSComm1.Input.
        'Close the serial port.
        MSComm1.PortOpen = False
End Sub.
```


เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 4
การทดลองโครงงาน

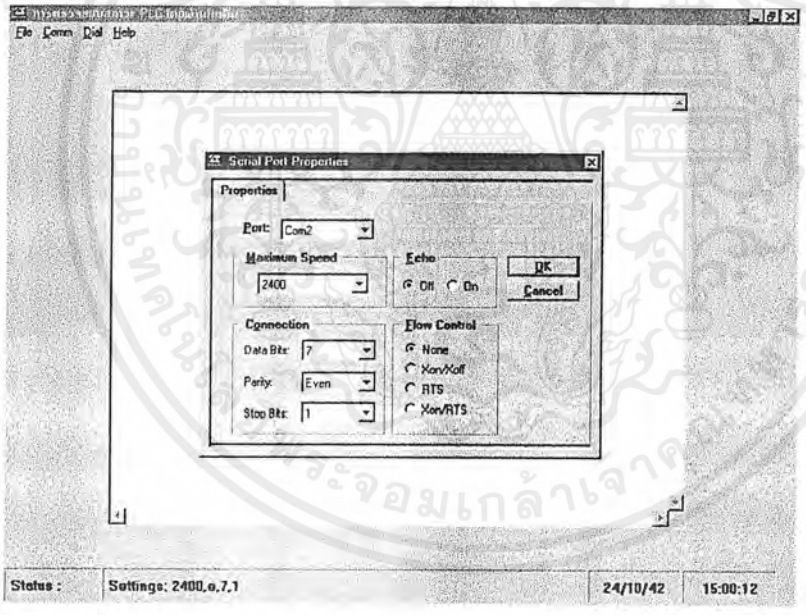
ในการทดลองโปรแกรมการตรวจสอบสภาวะ PLC โดยผ่านโมเด็ม ผลการทดลองเป็นที่ น่าพอใจโดยสรุปเป็นขั้นการทำงานดังนี้

แบ่งออกเป็น 2 ระบบคือ

1. ระบบที่ให้คอมพิวเตอร์เริ่มดำเนินการติดต่อสื่อสาร
2. ระบบที่ให้ PLC เริ่มดำเนินการติดต่อสื่อสาร

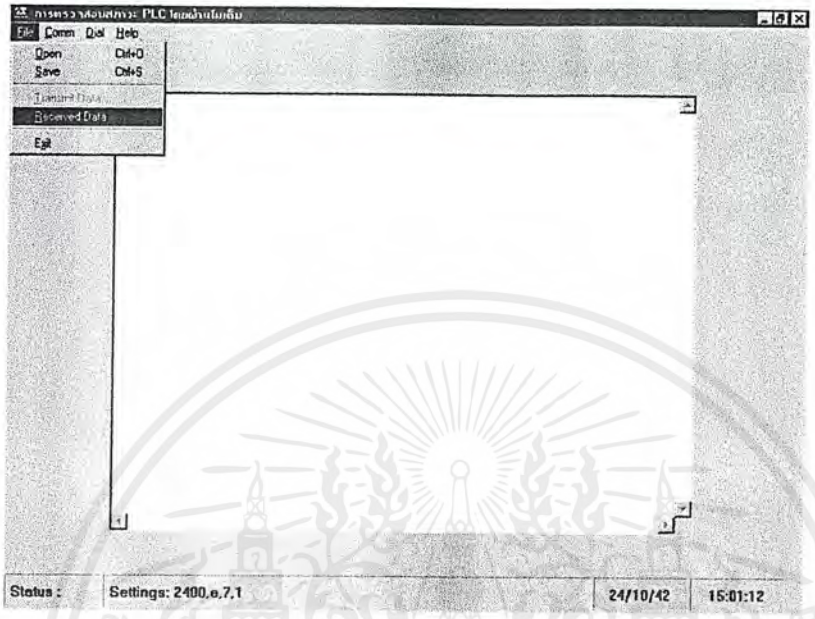
ในระบบที่ 1. มีขั้นตอนการทดลองดังนี้

4.1.1) ทำการ Set Serial Port โดยจาก Menu Comm ทำการเลือก Properties หรือ กดคีย์ F9



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

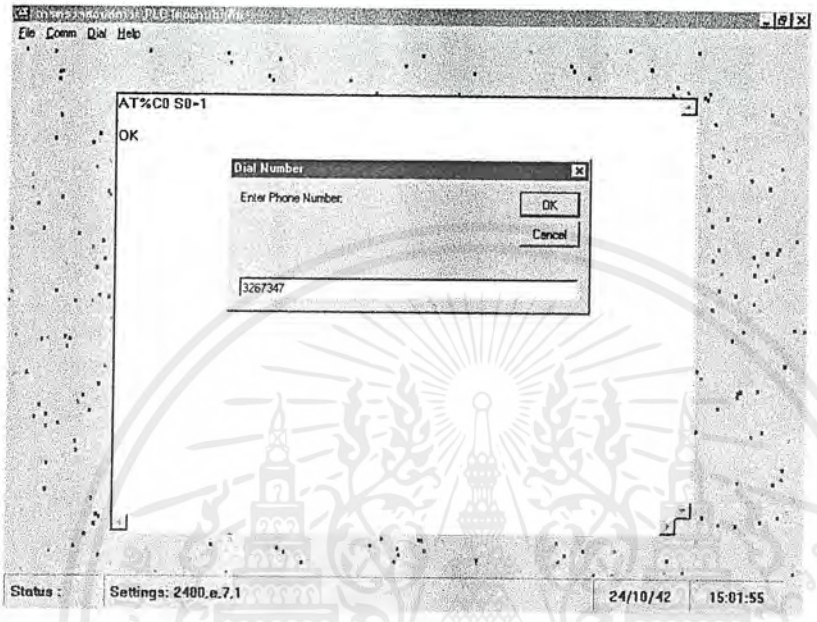
4.1.2) จาก Menu ทำการเลือก Received Data เป็นการเปิด Port Set Parameter ต่างๆให้แก่โมเด็มก่อนมีการใช้งาน



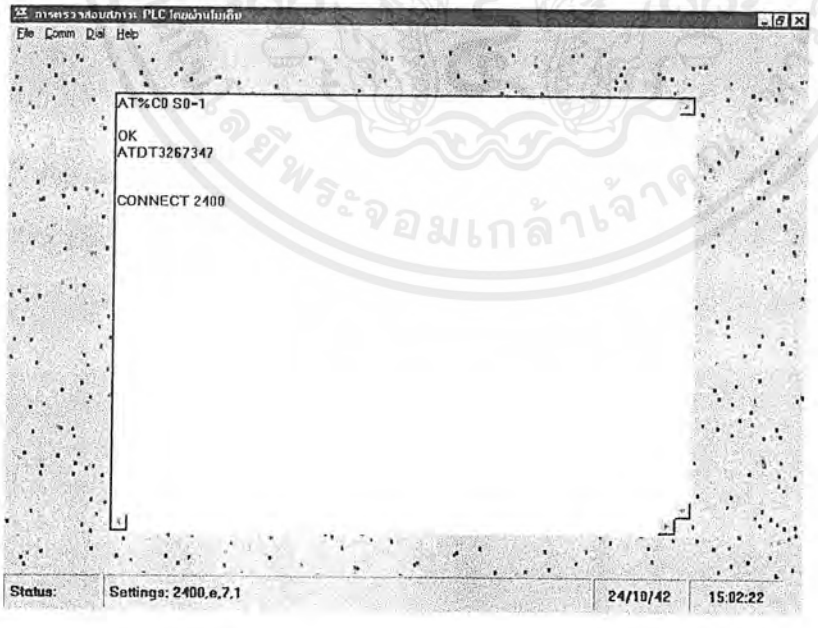
Text Box จะแสดงข้อความ Parameter และ คำว่า OK ซึ่งหมายความว่า Mode Received Data พร้อมใช้งาน ถ้า PLC เกิดปัญหาทำการหมุนเบอร์โทรศัพท์เข้ามา ก็สามารถรับข้อความที่ PLC ส่งมา

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.1.3) ถ้าต้องการติดต่อกับ PLC ก็ทำการหมุนหมายเลขโทรศัพท์ของ PLC เครื่องที่ต้องการ โดยจาก Menu Dial เลือก Dial Number หรือกดคีย์ F5 ทำการใส่หมายเลขโทรศัพท์ จากนั้นรอการติดต่อจากโมเด็มปลายทาง

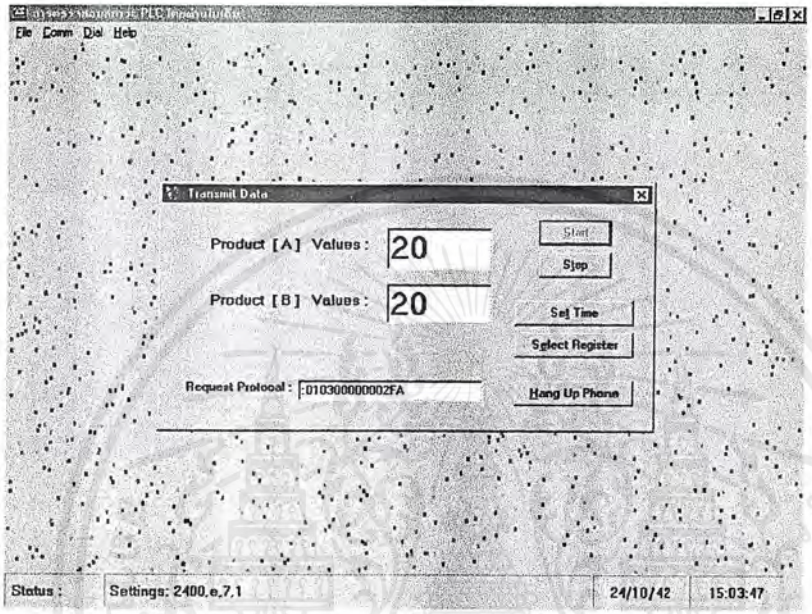


4.1.4) เมื่อติดต่อได้แล้ว จะปรากฏข้อความ " CONNECT 2400 "



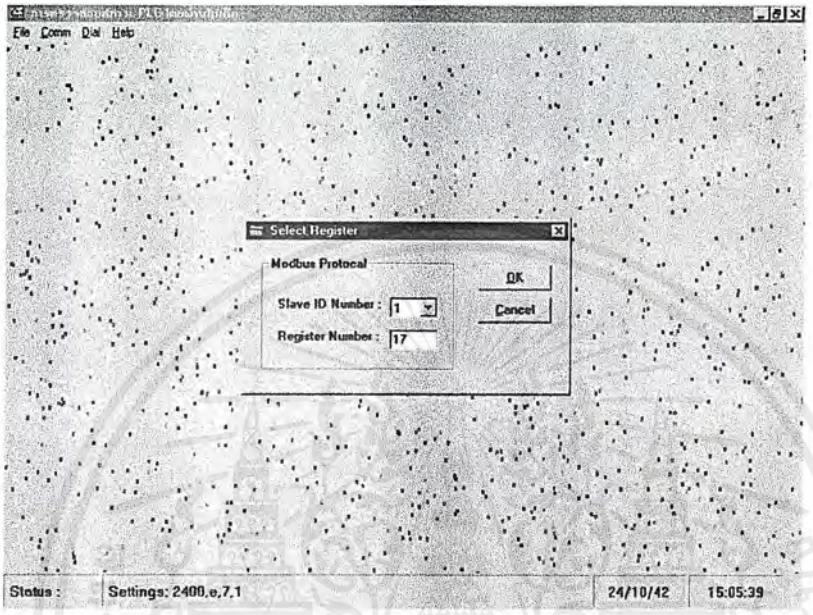
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

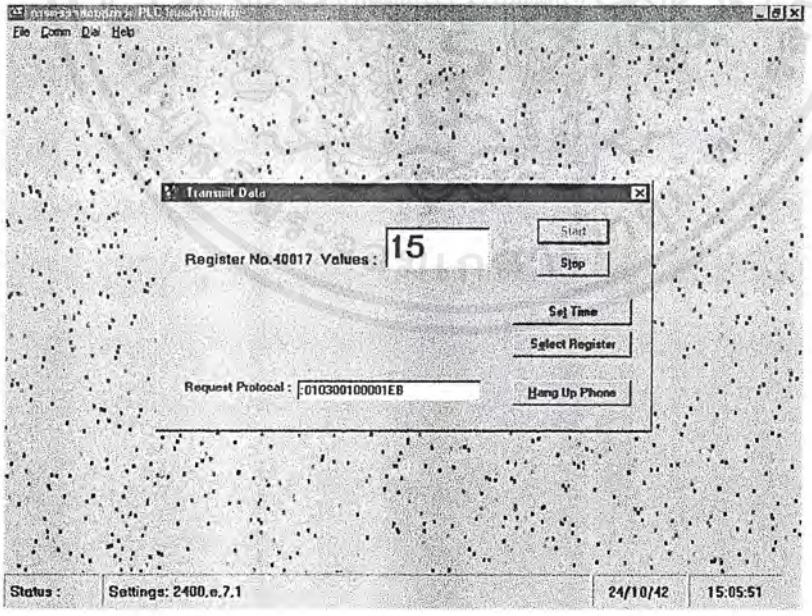
4.1.5) จาก Menu File เลือก Mode Transmit Data จะปรากฏ Form Transmit Data ถ้าคลิกปุ่ม Start จะเป็นการดูค่าใน Register ของ PLC ใน Project ซึ่งในที่นี่เป็น Product A และ Product B



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

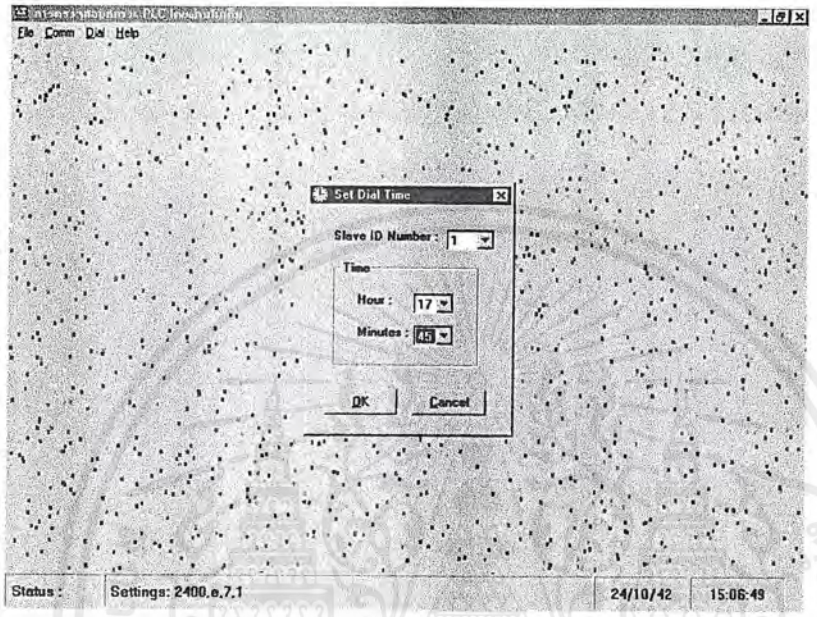
4.1.6) ถ้าต้องการดูค่าใน Register อื่นๆ ก็สามารถทำได้โดย คลิกปุ่ม Select Register ก็จะปรากฏ Form Select Register ให้ใส่ว่า Slave ID Number และค่า Register ที่ต้องการ จากนั้น คลิกปุ่ม OK แล้ว คลิกปุ่ม Start อีกครั้ง

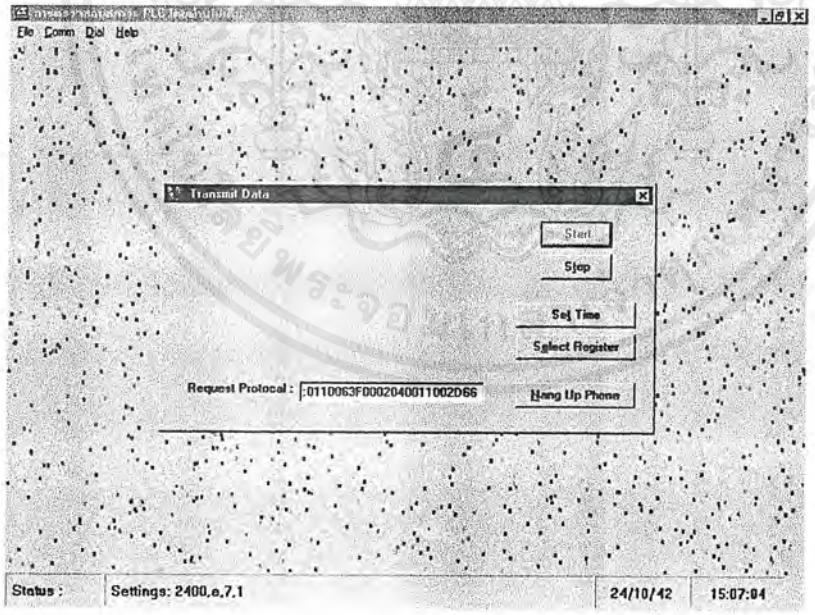




เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.1.7) สามารถ Set ค่า Time ให้แก่ PLC แต่ละเครื่อง หมุนเบอร์โทรศัพท์กลับมารายงานผลได้ตามเวลาที่ต้องการ โดย คลิกปุ่ม Set Time จะปรากฏ Form Set Dial Time ให้ใส่ค่า Slave ID Number และค่าเวลาที่ต้องการคลิกปุ่มจากนั้น คลิกปุ่ม OK แล้ว คลิกปุ่ม Sart อีกครั้ง





จากตัวอย่าง เป็น การ SET ให้ PLC โทรกลับมารายงานผลในเวลา 17.45 น.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

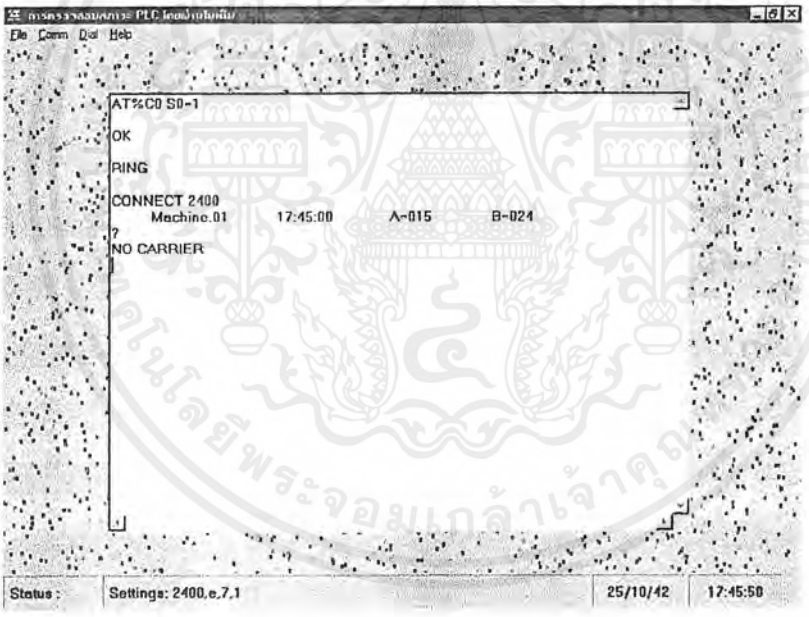
4.1.8) เมื่อดูค่าต่างๆหรือ Set Time เสร็จเรียบร้อยแล้ว ให้คลิกปุ่ม Hang Up Phone เพื่อเป็นการวางสายโทรศัพท์ ยกเลิกการติดต่อ

ระบบที่ 2. มีขั้นตอนการทดลองดังนี้

4.2.1) เมื่อต้องการรอรับค่าที่ PLC ส่งกลับมาโดย

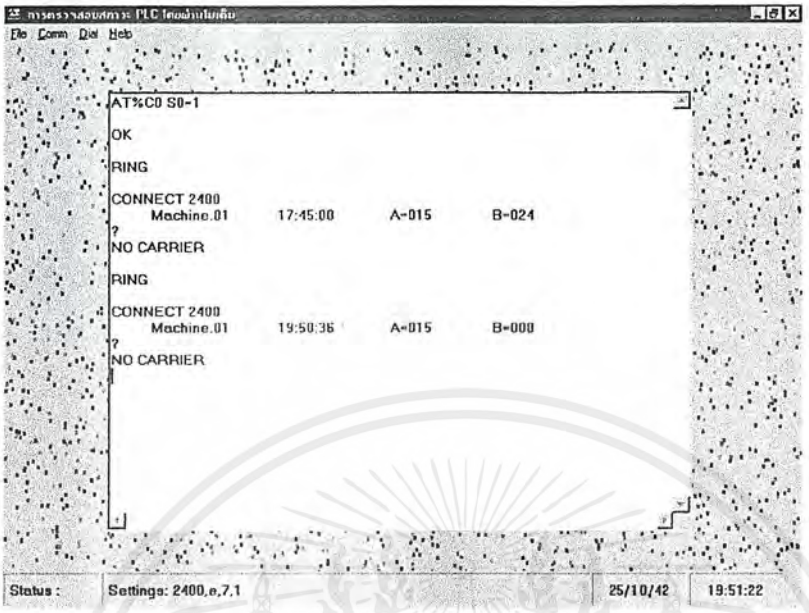
จาก Menu File เลือก Mode Received Data กรณี เมื่อถึงเวลาที่ Set Time ให้กับ PLC แต่ละเครื่อง PLC จะทำการหมุนหมายเลขโทรศัพท์เข้ามา เพื่อติดต่อกับคอมพิวเตอร์แล้วจะรายงานผลซึ่งมีรูป แบบดังนี้

- Slave ID Number ของ PLC แต่ละเครื่อง
- เวลาที่ทำการส่ง Data เข้ามา
- ค่า Data ใน Register

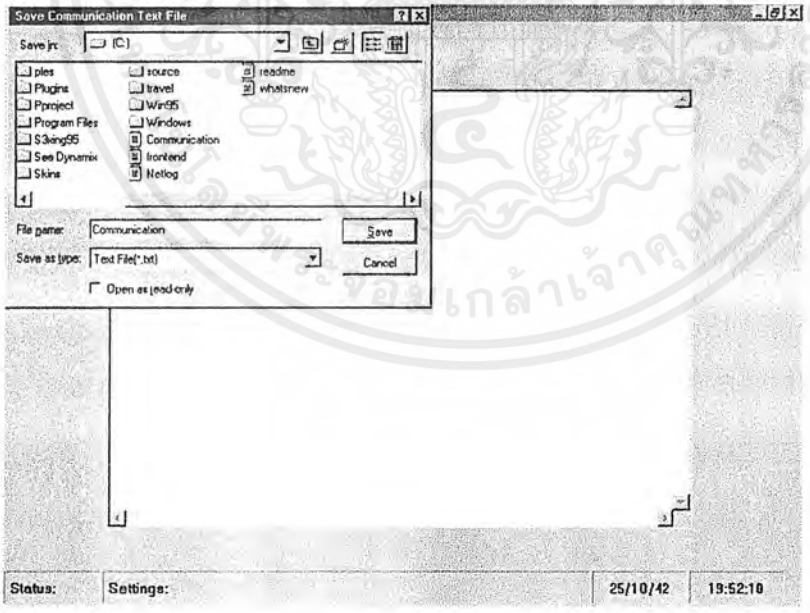


จากตัวอย่าง ข้อความที่ส่งเข้ามาคือ เป็น PLC เครื่องที่ 01 , เวลาที่ส่งข้อมูล 17.45 , ค่า Data Product A = 15 และค่า Product B = 24 หรือกรณีที่ PRODUCT B เกิดหมด PLC ก็จะทำการหมุนหมายเลขโทรศัพท์ เข้ามา เพื่อติดต่อกับ PLC แล้วจะรายงานผล เช่นเดียวกัน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



4.2.2) ข้อมูลที่ PLC ส่งมา สามารถจะ Save เก็บข้อมูลเป็น Text File และสามารถเปิด Text File ดังกล่าวได้



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่5
บทสรุปและวิจารณ์

ปริญญานิพนธ์ฉบับนี้ ได้ดำเนินงานบรรลุตามวัตถุประสงค์ คือสามารถพัฒนาการใช้งานของ PLC ให้มีความสะดวกในการตรวจสอบสภาวะ PLC จากที่ที่ห่างจาก PLC เป็นระยะทางไกล ๆ ได้โดยมีโมเด็มและเครือข่ายโทรศัพท์ทำหน้าที่เชื่อมโยงสัญญาณ และสามารถจัดทำให้ PLC หรือคอมพิวเตอร์เป็นตัวเริ่มการติดต่อสื่อสาร ส่งและรับข้อมูลได้ นอกจากนี้ PLC จะทำการแจ้งผลผ่านโมเด็มและเครือข่ายโทรศัพท์เมื่อมีการทำงานของกระบวนการเปลี่ยนแปลง ซึ่งจากการทดลองในปริญญานิพนธ์ฉบับนี้ ได้พบว่าอาจจะสามารถพัฒนาปริญญานิพนธ์ฉบับนี้ต่อไปได้อีก โดยการพัฒนาให้ PLC เริ่มการติดต่อสื่อสารและสามารถส่งผลมาให้คอมพิวเตอร์ แล้วให้คอมพิวเตอร์สามารถแก้ไขกระบวนการหรือจัดการกับการทำงานของ PLC ได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก

A: LADDER PROGRAM

B: COMMUNICATION PROGRAM

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก A

LADDER PROGRAM

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Configuration

CONFIGURATION OVERVIEW

PLC :		Size of Full Logic Area	02121
PLC Type	MICRO-S	No. of I/O Map Words	00153
Model	512/00	I/O :	
System Memory	3.1K	Number of Segments	2
Micro Child ID	None	Number of Children	0
		I/O Locations	18
		Specials :	
Ranges :		Battery Coil	00081
0xxxx	00001 - 01536	Timer Register	40011
1xxxx	10001 - 10512	Time of Day Clock	40012 - 40019
3xxxx	30001 - 30048		
4xxxx	40001 - 41872		

CONFIGURATION EXTENSION BLOCKS

No Configuration Extensions found

MICRO PORTS

MODE	RS232-1	RS232-2	RS485
Single	MODBUS/ASCII	MODBUS	Not Used

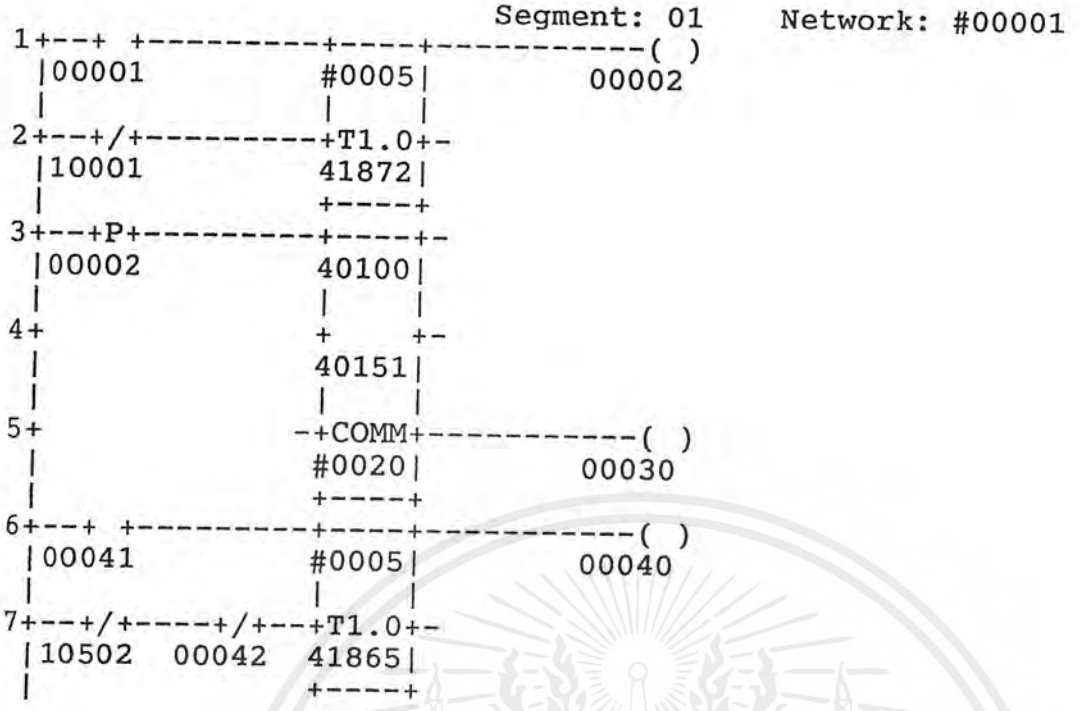
MODBUS Parameters

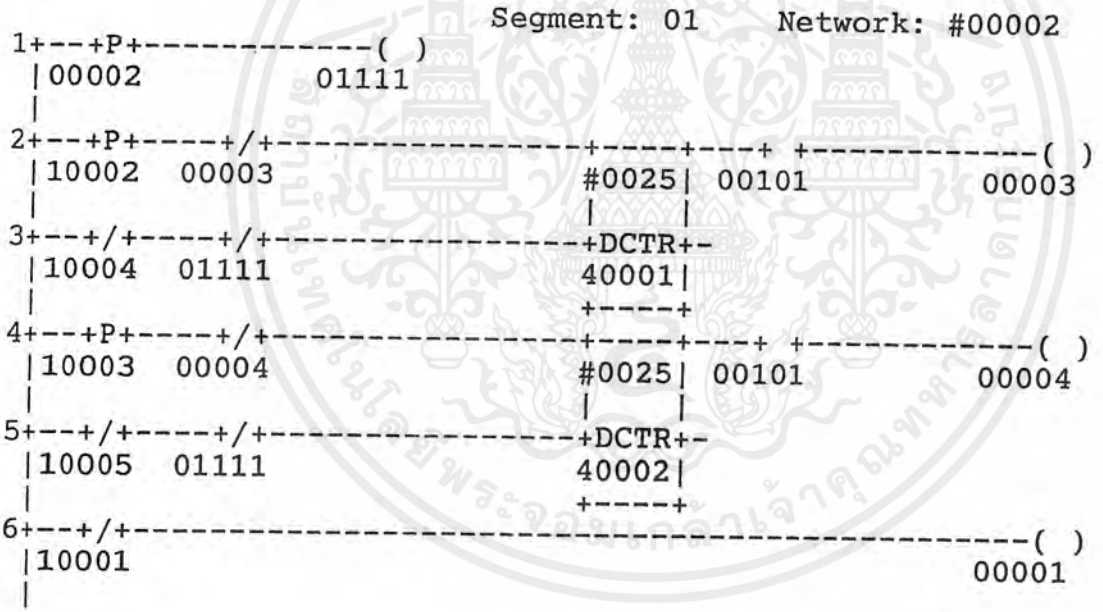
Port	Mode	Data Bits	Parity	Stop Bits	Baud	Address	Delay
RS232-1	ASCII	7	EVEN	1	2400	1	10 ms
RS232-2	RTU	8	EVEN	1	9600	1	10 ms

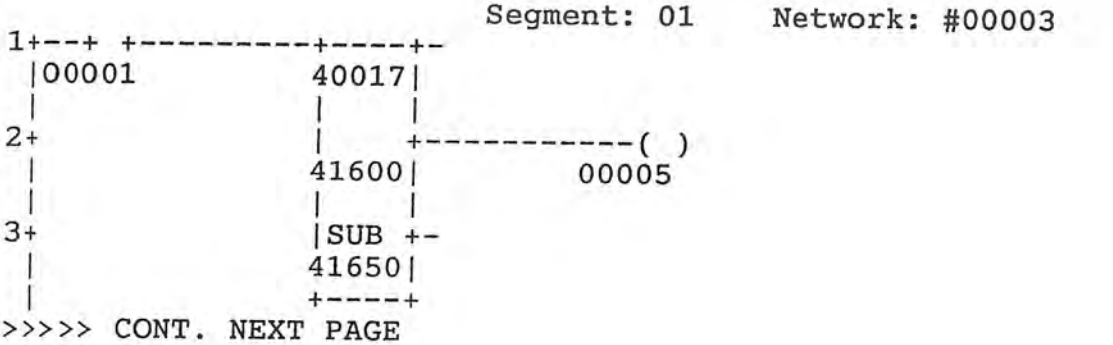
ASCII Parameters

7	EVEN	1	2400

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



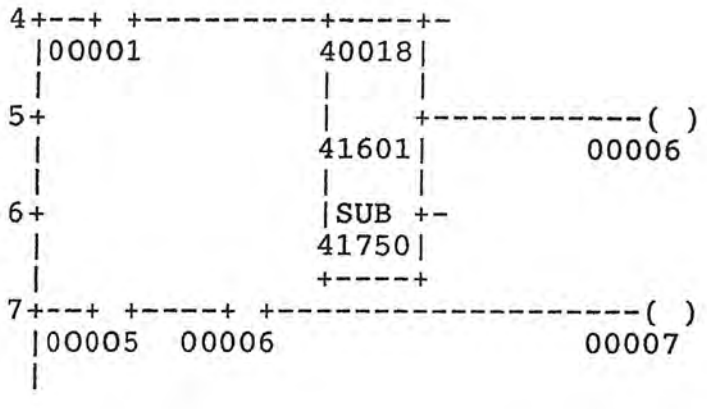




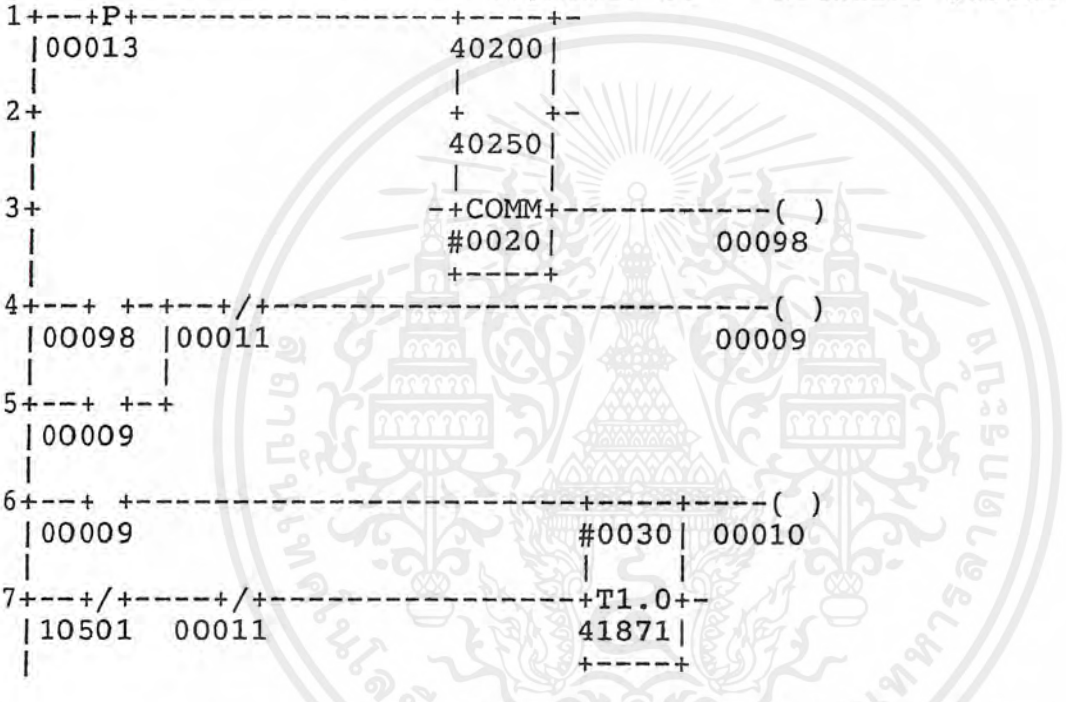
>>>>> CONT. NEXT PAGE

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

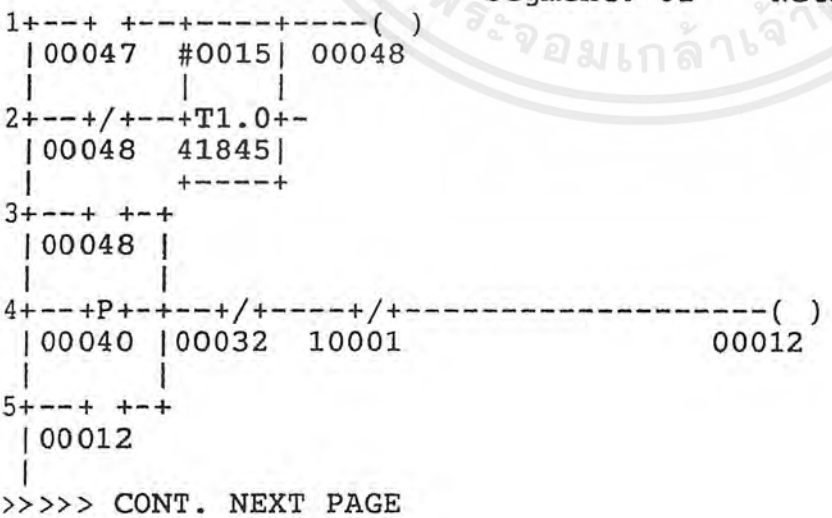
<<<<< FROM PREV PAGE



Segment: 01 Network: #00004



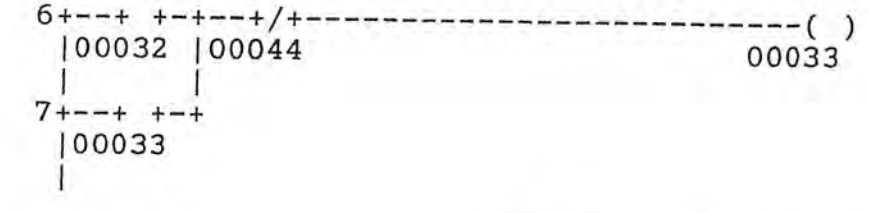
Segment: 01 Network: #00005



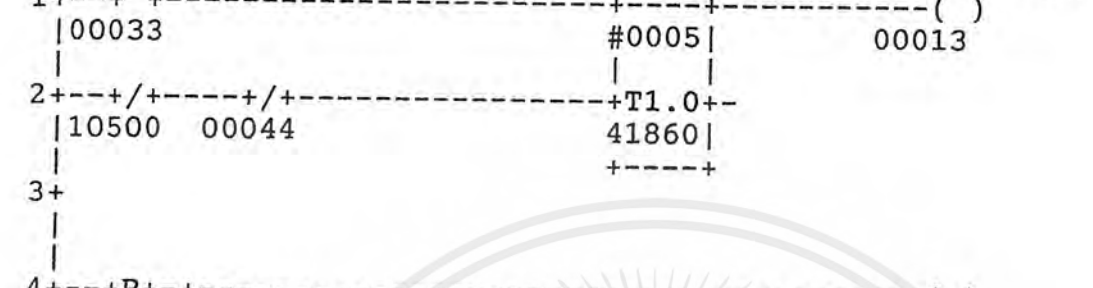
>>>>> CONT. NEXT PAGE

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

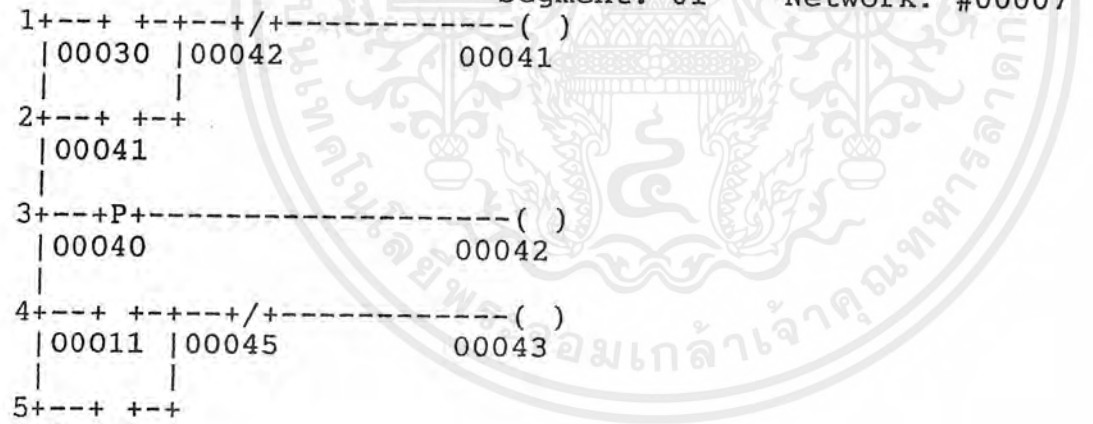
<<<<< FROM PREV PAGE



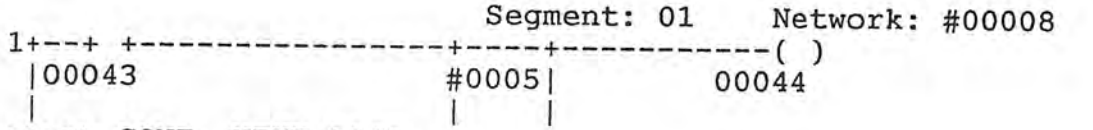
Segment: 01 Network: #00006



Segment: 01 Network: #00007



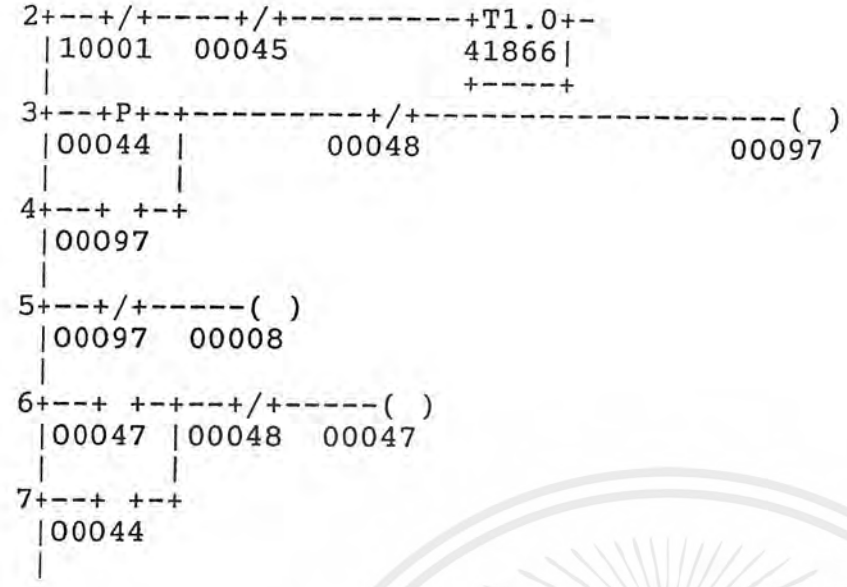
Segment: 01 Network: #00008

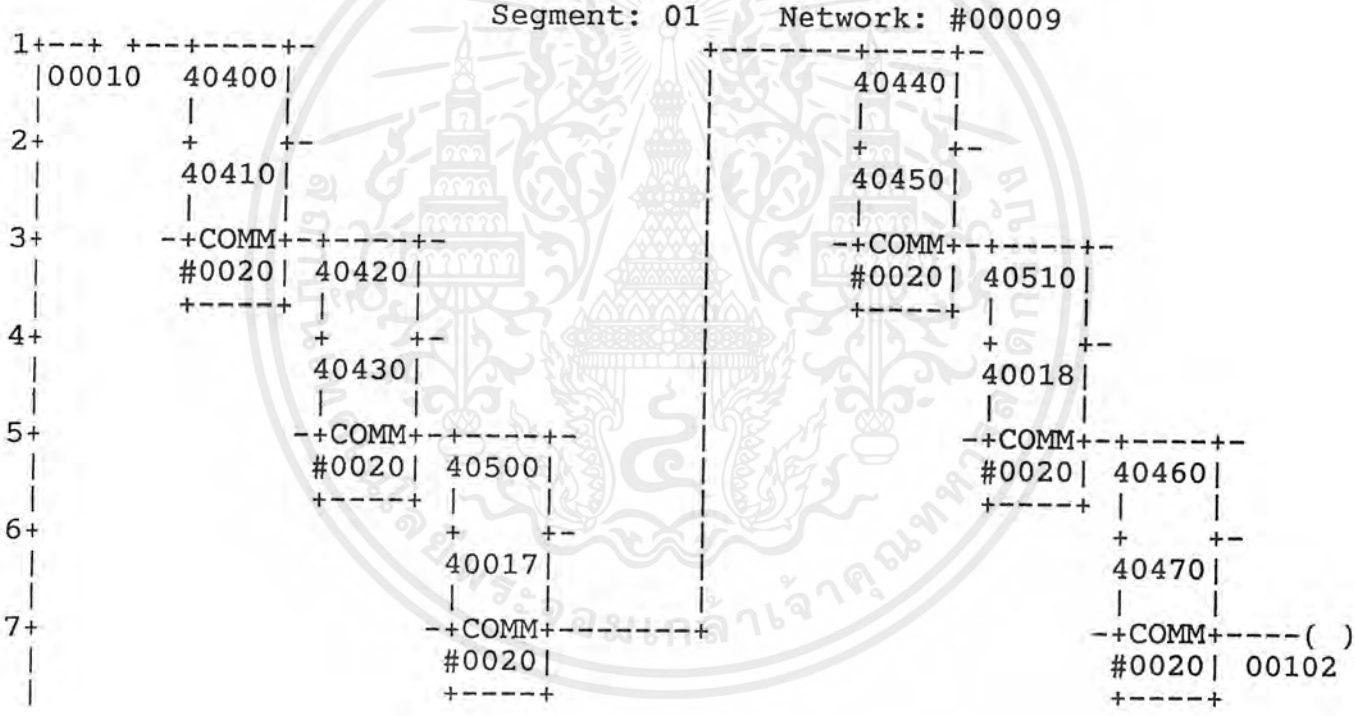


>>>>> CONT. NEXT PAGE

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

<<<<< FROM PREV PAGE

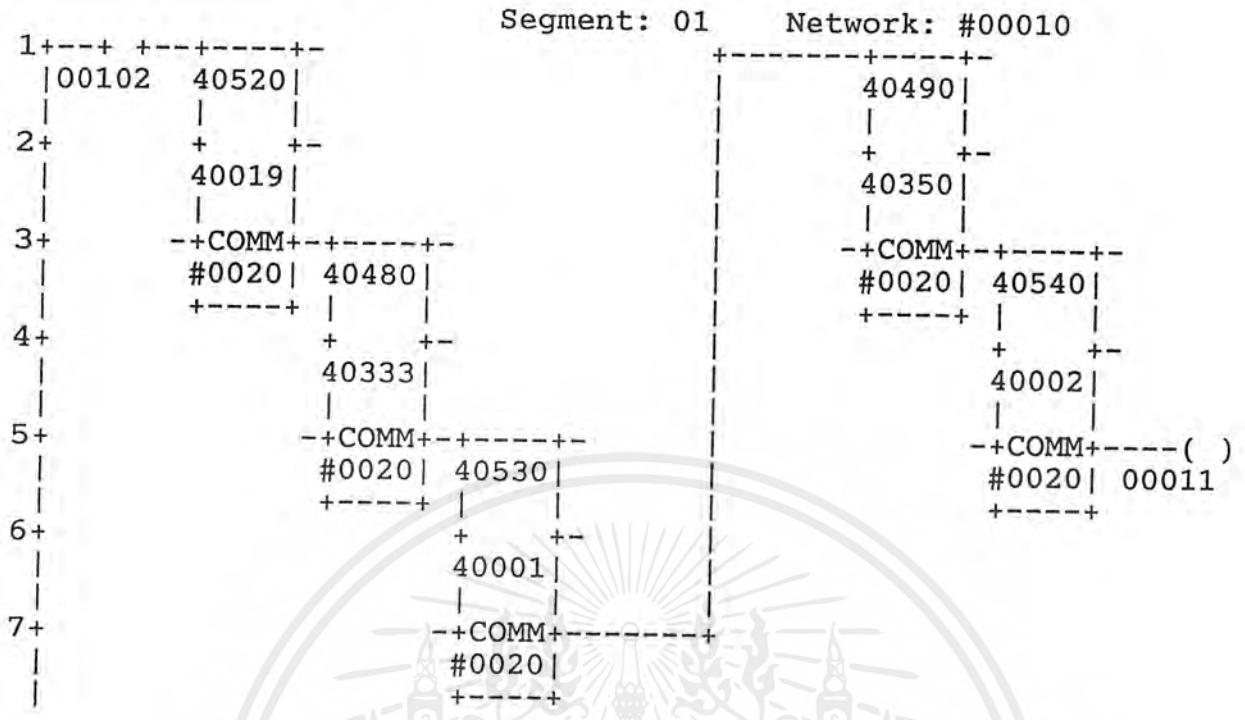




>>>>> CONT. NEXT PAGE

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

<<<<< FROM PREV PAGE



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก B

COMMUNICATION PROGRAM

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
Form1

Option Explicit

Dim Num$
Dim BufInt%
Dim Echo As Boolean


Private Sub Form_Load()
    Form7.Show
    Form1.Visible = False
    Term.Text = ""
    Term.ForeColor = vbBlue
    StatusBar1.Panels("Settings").Text = "Settings: " & MSComm1.Settings
End Sub


Private Sub mnuClosePort_Click()
    If MSComm1.PortOpen Then
          MSComm1.PortOpen = False
    End If

    Form1.Timer1.Enabled = False
    Form1.Cls
    StatusBar1.Panels("Settings").Text = "Settings: "
    StatusBar1.Panels("Status").Text = "Status: "
End Sub


Private Sub mnuDialPhone_Click()
    On Local Error Resume Next
    Static Num$
       Num$ = InputBox$("Enter Phone Number:", "Dial Number", Num$)
          If Num$ = "" Then Exit Sub
          If Not MSComm1.PortOpen Then
            MSComm1.PortOpen = True
          End If
          StatusBar1.Panels("Settings").Text = "Settings: " & MSComm1.Settings
          'Dial the dialNumber.
          MSComm1.Output = "ATDT" & Num$ & Chr$(13) & Chr(10)
          'Enable hang up button and menu item.
          mnuHangUp.Enabled = True
          mnuProject.Enabled = True
          mnuOpenPort.Enabled = False
End Sub


Private Sub mnuExit_Click()
    If MSComm1.PortOpen Then
          MSComm1.PortOpen = False
    End If
    Call mnuFileSave_Click
    Unload Form1
    Unload Form2
    Unload Form3
    Unload Form4
    Unload Form5
    Unload Form6
    Unload Form7
End Sub


Private Sub mnuFileOpen_Click()
    Dim Cancel As Boolean
    Dim iFileNum As Integer
    Dim DataVar As String
    Dim filesize As Long
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    On Error GoTo ErrHandler
    Cancel = False
    CommonDialog1.DialogTitle = " Open Communication Text File "
    CommonDialog1.ShowOpen
    If Not Cancel Then
       iFileNum = FreeFile
       Open CommonDialog1.FileName For Input As #iFileNum
       filesize = LOF(iFileNum)
       DataVar = Input(filesize, #iFileNum)
       Term.Text = DataVar
       Close #iFileNum
    End If
Exit Sub

ErrHandler:
    If Err.Number = cdlCancel Then
      Cancel = True
      Resume Next
    End If
End
End Sub


' Toggle the DTREnable property to hang up the line.
Private Sub mnuHangup_Click()
    On Error Resume Next
    MSComm1.Output = "~~~+++~~~ATH0" ' Send hangup string
    BufInt% = MSComm1.DTREnable      ' Save the current setting.
    MSComm1.DTREnable = True         ' Turn DTR on.
    MSComm1.DTREnable = False        ' Turn DTR off.
    MSComm1.DTREnable = BufInt%      ' Restore the old setting.
    mnuHangUp.Enabled = False
    ' If port is actually still open, then close it
    If MSComm1.PortOpen Then
          MSComm1.PortOpen = False
    End If

    Form1.Timer1.Enabled = False
    Form1.Cls
    mnuHangUp.Enabled = False
    mnuProject.Enabled = False
    mnuDial.Enabled = True
    mnuOpenPort.Enabled = True
    StatusBar1.Panels("Settings").Text = "Settings: "
    StatusBar1.Panels("Status").Text = "Status: "
End Sub


Private Sub mnuHelp_Click()
    Form3.Show
End Sub


Private Sub mnuOpenPort_Click()
    If Not MSComm1.PortOpen Then
          MSComm1.PortOpen = True
    End If

    Form1.Timer1.Enabled = True
    StatusBar1.Panels("Settings").Text = "Settings: " & MSComm1.Settings
    MSComm1.Output = "AT%C0 S0=1" + Chr$(13) + Chr$(10)
End Sub
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
Private Sub mnuProject_Click()
    Term.Visible = False
    Form4.Text2 = ""
    Form4.Show
End Sub


Private Sub mnuProperties_Click()
    Form2.Show
End Sub


Private Sub mnuFileSave_Click()
    Dim Cancel As Boolean
    Dim iFileNum As Integer

    On Error GoTo ErrHandler
    Cancel = False
    CommonDialog1.DialogTitle = " Save Communication Text File "
    CommonDialog1.ShowSave
    If Not Cancel Then
       iFileNum = FreeFile
       Open CommonDialog1.FileName For Output As #iFileNum
       Print #iFileNum, Form1.Term.Text
       Close #iFileNum
    End If
Exit Sub

ErrHandler:
    If Err.Number = cdlCancel Then
      Cancel = True
      Resume Next
    End If
End
End Sub


Private Static Sub MSComm1_OnComm()
    Dim EVMsg$
    Dim ERMsg$

    ' Branch according to the CommEvent property.
    Select Case MSComm1.CommEvent
    'Event messages
    Case comEvReceive
        Dim Buffer As Variant

        Buffer = MSComm1.Input
        ShowData Term, (StrConv(Buffer, vbUnicode))
    Case comEvSend
    Case comEvCTS: EVMsg$ = "Change in CTS Detected"
    Case comEvDSR: EVMsg$ = "Change in DSR Detected"
    Case comEvCD: EVMsg$ = "Change in CD Detected"
    Case comEvRing: EVMsg$ = "The Phone is Ringing"
    Case comEvEOF: EVMsg$ = "End of File Detected"

    ' Error messages.
    Case comBreak: ERMsg$ = "Break Received"
    Case comCDTO: ERMsg$ = "Carrier Detect Timeout"
    Case comCTSTO: ERMsg$ = "CTS Timeout"
    Case comDSRTO: ERMsg$ = "DSR Timeout"
    Case comFrame: ERMsg$ = "Framing Error"
    Case comOverrun: ERMsg$ = "Overrun Error"
    Case comRxOver: ERMsg$ = "Receive Buffer Overflow"
    Case comRxParity: ERMsg$ = "Parity Error"
    Case comTxFull: ERMsg$ = "Transmit Buffer Full"
    Case Else: ERMsg$ = "Unknown error or event"
    End Select
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    If Len(EVMsg$) Then
        ' Display event messages in the status bar.
        StatusBar1.Panels("Status").Text = "Status: " & EVMsg$
        EVMsg$ = ""
    ElseIf Len(ERMsg$) Then
        ' Display event messages in the status bar.
        StatusBar1.Panels("Status").Text = "Status: " & ERMsg$
        ' Display error messages in an alert message box.
        Beep
        BufInt% = MsgBox(ERMsg$, 1, "Click Cancel to quit, OK to ignore.")
        ERMsg$ = ""
        ' If the user clicks Cancel (2)
        If BufInt% = 2 Then
           MSComm1.PortOpen = False
        End If

    End If
End Sub


Private Static Sub ShowData(Term As Control, Data As String)
    On Error Resume Next
    Dim TermSize As Long, i

    ' Make sure the existing text doesn't get too large.
    TermSize = Len(Term.Text)
    If TermSize >= 16384 Then
       Term.Text = Mid$(Term.Text, 4097)
       TermSize = Len(Term.Text)
    End If

    ' Point to the end of Term's data.
    Term.SelStart = TermSize

    ' Filter/handle BACKSPACE characters.
    Do
       i = InStr(Data, Chr$(8))
       If i Then
          If i = 1 Then
             Term.SelStart = TermSize - 1
             Term.SelLength = 1
             Data = Mid$(Data, i + 1)
          Else
             Data = Left$(Data, i - 2) & Mid$(Data, i + 1)
          End If
       End If
    Loop While i

    ' Eliminate line feeds.
    Do
       i = InStr(Data, Chr$(10))
       If i Then
          Data = Left$(Data, i - 1) & Mid$(Data, i + 1)
       End If
    Loop While i

    ' Make sure all carriage BufInt%urns have a line feed.
    i = 1
    Do
       i = InStr(i, Data, Chr$(13))
       If i Then
          Data = Left$(Data, i) & Chr$(10) & Mid$(Data, i + 1)
          i = i + 1
       End If
    Loop While i

    ' Add the filtered data to the SelText property.
    Term.SelText = Data
End Sub
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
Private Sub Term_Change()
    Form4.Text2.Text = Term.Text
End Sub


Private Sub Term_KeyPress(KeyAscii As Integer)
    If MSComm1.PortOpen Then
        ' Send the keystroke to the port.
        MSComm1.Output = Chr$(KeyAscii)

        ' Unless Echo is on, there is no need to
        ' let the text control display the key.
        If Not Echo Then
            KeyAscii = 0
        End If
    End If
End Sub


Private Sub Timer1_Timer()
    Dim R As Integer
    Dim G As Integer
    Dim B As Integer
    Dim Xi As Single
    Dim Yi As Single

    Randomize
    R = Rnd * 255
    G = Rnd * 255
    B = Rnd * 255
    Xi = Rnd * Form1.ScaleWidth
    Yi = Rnd * Form1.ScaleHeight
    Form1.PSet (Xi, Yi), RGB(R, G, B)
End Sub


Private Sub Timer2_Timer()
    StatusBar1.Panels("CTime").Text = Format(Now, "hh:mm:ss")
End Sub
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
Form2

Option Explicit

Dim iFlow%
Dim Echo As Boolean


Private Sub CmdCancel_Click()
    Unload Form2
End Sub


Private Sub CmdOK_Click()
    Dim OldPort%, NewPort%
    Dim ReOpenPort As Boolean

    On Error Resume Next
    OldPort% = Form1.MSComm1.CommPort
    NewPort% = cboPort.ListIndex + 1

    'If port number changes ,Close the old port first.
    If NewPort% <> OldPort% Then
        ReOpenPort = False
        If Form1.MSComm1.PortOpen Then
            Form1.MSComm1.PortOpen = False
            ReOpenPort = True
        End If

       'Set new port number.
       Form1.MSComm1.CommPort = NewPort%

       If Err = 0 Then
           If ReOpenPort Then
              Form1.MSComm1.PortOpen = True
           End If
       End If
       If Err Then
           MsgBox Error$, vbExclamation
           Form1.MSComm1.CommPort = OldPort%
           Exit Sub
       End If
    End If

    Form1.MSComm1.Settings = (cboBaudRate.Text) & "," &
                             Mid$(cboParity.Text, 1, 1) & "," & _
                             (cboDataBits.Text) & "," & _
                             (cboStopBits.Text)

    If Err Then
        MsgBox Error$, vbExclamation
        Exit Sub
    End If

   Form1.MSComm1.Handshaking = iFlow%
    If Err Then
        MsgBox Error$, vbExclamation
        Exit Sub
    End If

    Unload Form2
    Form1.StatusBar1.Panels("Settings").Text = "Settings: " & _
    Form1.MSComm1.Settings
End Sub
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
Private Sub EchoOff_Click()
    Echo = 0
End Sub


Private Sub EchoOn_Click()
    Echo = True
End Sub


Private Sub Form_Load()
    Dim Settings As String, Length%
    'Get current port
    cboPort.ListIndex = Form1.MSComm1.CommPort - 1
    Settings = LCase$(Form1.MSComm1.Settings)
    Length% = Len(Settings)

    'Get current baud
    cboBaudRate.Text = Left$(Form1.MSComm1.Settings, Length% - 6)

    'Get current parity
    Select Case Mid$(Form1.MSComm1.Settings, Length% - 4, 1)
       Case "n": cboParity.ListIndex = 0
       Case "e": cboParity.ListIndex = 1
       Case "o": cboParity.ListIndex = 2
    End Select

    'Get data bits
    cboDataBits.Text = Mid$(Form1.MSComm1.Settings, Length% - 2, 1)

    'Get stop bits
    cboStopBits.Text = Right$(Form1.MSComm1.Settings, 1)

    optFlow(Form1.MSComm1.Handshaking).Value = True

    If Echo Then
        EchoOn.Value = True
    Else
        EchoOff.Value = True
    End If
End Sub


Private Sub optFlow_Click(Index As Integer)
   iFlow% = Index
End Sub



Form3

Option Explicit


Private Sub CmdCloseHelp_Click()
    Unload Form3
End Sub
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
Form4

Option Explicit

Dim LH, HD, LHA, HHA, Ans As String
Dim X, s, LD, A, B As String
Dim BufInt%
Dim Echo As Boolean


Private Sub CmdCancelProject_Click()
    Text3.Text = ""
    Text4.Text = ""
    Unload Form4
    Form1.Term.Visible = True

    On Error Resume Next
    Form1.MSComm1.Output = "~~~+++~~~ATH0" ' Send hangup string
    BufInt% = Form1.MSComm1.DTREnable      ' Save the current setting.
    Form1.MSComm1.DTREnable = True         ' Turn DTR on.
    Form1.MSComm1.DTREnable = False        ' Turn DTR off.
    Form1.MSComm1.DTREnable = BufInt%      ' Restore the old setting.
    Form1.mnuHangUp.Enabled = False
    ' If port is actually still open, then close it
    If Form1.MSComm1.PortOpen Then
        Form1.MSComm1.PortOpen = False
    End If

    Form1.Timer1.Enabled = False
    Form1.Cls
    Form1.mnuHangUp.Enabled = False
    Form1.mnuProject.Enabled = False
    Form1.mnuDial.Enabled = True
    Form1.mnuOpenPort.Enabled = True
    Form1.StatusBar1.Panels("Settings").Text = "Settings: "
    Form1.StatusBar1.Panels("Status").Text = "Status: "
End Sub


Private Sub CmdOption_Click()
    Text1.Text = ""
    Text3.Text = ""
    Text4.Text = ""
    Unload Form4
    Form5.Show
End Sub


Private Sub CmdSetTime_Click()
    Text1.Text = ""
    Text3.Text = ""
    Text4.Text = ""
    Unload Form4
    Form6.Show
End Sub


Private Sub Command1_Click()
    Dim N, M, T, Z As String
    Dim InString$
    Dim i As Integer
    Dim L As Long

    Command1.Enabled = False
    Command2.Enabled = True
    Echo = True
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    On Local Error Resume Next
    If Not Form1.MSComm1.PortOpen Then
        Form1.MSComm1.PortOpen = True
    End If
    Form1.MSComm1.InputLen = 0
    Call ChkLRC
    Do While Form1.MSComm1.PortOpen
        Form1.MSComm1.Output = Text1.Text + Chr$(13) + Chr$(10)
            BufInt% = DoEvents()
        InString$ = Form1.Term.Text
        Form1.Term.Text = ""
        i = InStr(InString$, Chr$(10))
        Text2.MaxLength = i
        Text2.Text = InString$
        L = 0
        Do While L < 100000
           L = L + 1
        Loop
        Call ChkResponse
        Select Case Ans
           Case "00"
             Z = Text2.Text
             T = Left$(Z, 1)
             Select Case T
                 Case ":"
                 N = Val("&H" + Mid(Z, 8, 4))
                 M = Val("&H" + Mid(Z, 12, 4))
                 Case "0"
                 N = Val("&H" + Mid(Z, 7, 4))
                 M = Val("&H" + Mid(Z, 11, 4))
             End Select
             If N <= -1 Then
                N = 65536 + N
             End If
             Text3.Text = N

             If M <= -1 Then
               M = 65536 + M
             End If
             Text4.Text = M
        End Select
    Loop
End Sub


Private Sub ChkLRC()
    Dim LDA, HDA, LRC As String
    Dim h1$, h2$

    X = Text1.Text
    s = Len(X)
    LD = 0
    HD = 0
    For s = 0 To s - 1 Step 2
         A = Val("&H" + Mid(X, s + 2, 1))
         LD = A + LD
         B = Val("&H" + Mid(X, s + 1, 1))
         HD = B + HD
    Next s
    LH = Hex(LD)
    Call Chksize
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    Select Case LHA
        Case "0": LDA = 15
        Case "1": LDA = 14
        Case "2": LDA = 13
        Case "3": LDA = 12
        Case "4": LDA = 11
        Case "5": LDA = 10
        Case "6": LDA = 9
        Case "7": LDA = 8
        Case "8": LDA = 7
        Case "9": LDA = 6
        Case "A": LDA = 5
        Case "B": LDA = 4
        Case "C": LDA = 3
        Case "D": LDA = 2
        Case "E": LDA = 1
        Case "F": LDA = 0
    End Select
    Select Case HHA
        Case "0": HDA = 15
        Case "1": HDA = 14
        Case "2": HDA = 13
        Case "3": HDA = 12
        Case "4": HDA = 11
        Case "5": HDA = 10
        Case "6": HDA = 9
        Case "7": HDA = 8
        Case "8": HDA = 7
        Case "9": HDA = 6
        Case "A": HDA = 5
        Case "B": HDA = 4
        Case "C": HDA = 3
        Case "D": HDA = 2
        Case "E": HDA = 1
        Case "F": HDA = 0
    End Select
    LDA = LDA + 1
    LD = LDA
    HD = HDA
    LH = Hex(LD)
    Call Chksize
    h1$ = HHA
    h2$ = LHA
    LRC = h1$ & h2$
    Text1.Text = ":" + X + LRC
End Sub


Private Sub ChkResponse()
    X = Text2.Text
    s = Len(X)
    LD = 0
    HD = 0
    For s = 0 To s - 1 Step 2
        A = Val("&H" + Mid(X, s + 3, 1))
        LD = A + LD
        B = Val("&H" + Mid(X, s + 2, 1))
        HD = B + HD
    Next s
    LH = Hex(LD)
    Call Chksize
    Ans = HHA + LHA
End Sub
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
Private Sub Chksize()
    Dim Size, LHT, LDT As String
    Dim HH$

    LHA = Right$(LH, 1)
    Size = Len(LH)
    If Size = 2 Then
        LHT = Left$(LH, 1)
        LDT = Val("&H" & LHT)
        HD = HD + LDT
        HH$ = Hex(HD)
        HHA = Right$(HH$, 1)
    End If
    If Size = 1 Then
        HH$ = Hex(HD)
        HHA = Right$(HH$, 1)
    End If
End Sub


Private Sub Command2_Click()
    Command1.Enabled = True
    Command2.Enabled = False
    Text1.Text = ""
    Text3.Text = ""
    Text4.Text = ""
End Sub


Private Sub Form_Load()
    Text3.ForeColor = vbBlue
    Text4.ForeColor = vbBlue
End Sub



Form5

Option Explicit

Dim IDNo$, ID$, RNo$, R$, RR$
Dim i As Integer


Private Sub CmdOKOption_Click()
    Form4.Show
    Form4.Label1.Visible = False
    Form4.Label4.Visible = False
    Form4.Text4.Visible = False
    Form4.Text3.Text = ""

    IDNo$ = Combo1.Text
    If IDNo$ = "" Then
       Unload Form5
       Form4.Show
       Form4.Label1.Visible = True
       Form4.Label4.Visible = True
       Form4.Text4.Visible = True
    Else
      ID$ = Hex(IDNo$)
      i = Len(ID$)
      If i = 1 Then
         ID$ = "0" & ID$
      Else
         ID$ = ID$
      End If
    End If
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    RNo$ = Text1.Text
    If RNo$ = "" Then
       Unload Form5
       Form4.Show
       Form4.Label1.Visible = True
       Form4.Label4.Visible = True
       Form4.Text4.Visible = True
    Else
      RNo$ = RNo$ - 1
      R$ = Hex(RNo$)
      Call Laba13
      Form4.Text1.Text = ID$ & "03" & R$ & "0001"
      RNo$ = RNo$ + 1
      R$ = Hex(RNo$)
      Call Laba13
      R$ = Val("&H" & R$)
      Call Laba13
      Form4.Label3.Caption = "Register No." & "4" & R$ & "  Values :"
      Unload Form5
    End If
End Sub


Private Sub CmdCancelOption_Click()
    Unload Form5
    Form4.Show
    Form4.Text1.Text = "010300000002"
    Form4.Text3.Text = ""
    Form4.Text4.Text = ""
End Sub


Private Sub Laba13()
    i = Len(R$)
    If i = 1 Then
         R$ = "0" & "0" & "0" & R$
    End If
    If i = 2 Then
         R$ = "0" & "0" & R$
    End If
    If i = 3 Then
         R$ = "0" & R$
    End If
    If i = 4 Then
         R$ = R$
     End If
End Sub
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
Form6

Option Explicit

Dim IDNo$, ID$, HHour$, Hour$, MMin$, Min$
Dim i As Integer


Private Sub CmdCancelTime_Click()
    Unload Form6
    Form4.Show
    Form4.Text1.Text = "010300000002"
    Form4.Text3.Text = ""
    Form4.Text4.Text = ""
End Sub


Private Sub CmdOKTime_Click()
    Form4.Show
    Form4.Label1.Visible = False
    Form4.Label4.Visible = False
    Form4.Text3.Visible = False
    Form4.Text4.Visible = False

    IDNo$ = Combo1.Text
    If IDNo$ = "" Then
       Unload Form6
       Form4.Show
       Form4.Label1.Visible = True
       Form4.Label4.Visible = True
       Form4.Text3.Visible = True
       Form4.Text4.Visible = True
    Else
      ID$ = Hex(IDNo$)
      i = Len(ID$)
      If i = 1 Then
         ID$ = "0" & ID$
      Else
         ID$ = ID$
      End If
    End If

    HHour$ = Combo2.Text
    If HHour$ = "" Then
       Unload Form6
       Form4.Show
       Form4.Label1.Visible = True
       Form4.Label4.Visible = True
       Form4.Text3.Visible = True
       Form4.Text4.Visible = True
    Else
      Hour$ = Hex(HHour$)
      i = Len(Hour$)
      If i = 1 Then
         Hour$ = "0" & Hour$
      Else
         Hour$ = Hour$
      End If
    End If

    MMin$ = Combo3.Text
    If MMin$ = "" Then
       Unload Form6
       Form4.Show
       Form4.Label1.Visible = True
       Form4.Label4.Visible = True
       Form4.Text3.Visible = True
       Form4.Text4.Visible = True
    Else
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        Min$ = Hex(MMin$)
        i = Len(Min$)
        If i = 1 Then
           Min$ = "0" & Min$
        Else
           Min$ = Min$
        End If
        Form4.Text1.Text = ID$ & "10063F000204" & "00" & Hour$ & "00" & Min$
        Unload Form6
    End If
End Sub


Form7

Option Explicit


Private Sub Command1_Click()
    Form1.Show
    Unload Form7
End Sub


Private Sub Command2_Click()
    Unload Form1
    Unload Form7
End Sub
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้คงไม่สามารถสำเร็จลุล่วงไปได้ ถ้าไม่ได้รับความช่วยเหลือจากอาจารย์เชื้อ นกอยู่ และ อาจารย์ท่านอื่นๆที่คอยให้คำปรึกษาและช่วยแก้ไขปัญหาต่างๆด้วยดีตลอดมา

ขอขอบคุณ พ่อและแม่ ที่คอยเป็นกำลังใจ และให้ทุนทรัพย์ในการทำโครงงานนี้

ขอขอบคุณ การไฟฟ้าฝ่ายผลิตแห่งประเทศไทยที่คอยให้คำปรึกษาและช่วยแก้ไขปัญหาต่างๆ

ขอขอบคุณ คุณวรรธณะ จตุรกิตตินันท์ บริษัทออโต้อินโฟ จำกัด

ขอขอบคุณ เพื่อนๆนักศึกษาภาควิชาเทคโนโลยีการวัดคุมทางอุตสาหกรรม ที่คอยให้คำปรึกษาพร้อมทั้งกำลังใจ

สุดท้ายขอขอบคุณภาควิชาเทคโนโลยีการวัดคุมทางอุตสาหกรรม สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ที่ได้เอื้อเฟื้อในเรื่องเครื่องมือและอุปกรณ์ต่างๆในการทำโครงงานนี้ให้สำเร็จลุล่วงไปด้วยดี

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บรรณานุกรม

1. ราบินเดอร์ ศรีกิจจาภรณ์ , " คู่มือการใช้งาน Visual Basic สำหรับวินโดวส์ ", บริษัทซีเอ็ดยูเคชั่น , 407 หน้า , 2538.
2. บัญชา ยงฤทธิธิกุล , " เริ่มแรกกับ MODEM " , บริษัทซีเอ็ดยูเคชั่น , 318 หน้า , 2538.
3. เอกวิทย์ จิตรดา , " MODBUS PROTOCAL " , เทคนิค165 , หน้า 65-68 , 2538.
4. สุทธิศักดิ์ พงศ์ธนาพานิช , " Visual Basic 5.0 Professional การใช้คำสั่งและคอนโทรล Active X " , ซีเอ็ดยูเคชั่น , 1144 หน้า , 2541.
5. ธาริน สิทธิธรรมชารี , " Microsoft Visual Basic Version 5.0 " , บริษัท ส.เอเชียพลส (1989) จำกัด , 2539.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้