ระบบตอบรับโทรศัพท์อัตโนมัติ

AUTOMATIC TELEPHONE ANSWER BY MICROCOMPUTER

นายคมสรรค์ ยศสินศักดิ์
นายพงศ์พิพัฒน์ วุฒาพาณิชย์

วัน เดือน ปี 30 ก.ค. 2540
เลขทะเบียน 036984
เลขเรียกหนังสือ T 38077 ค.159 ร

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรอุตสาหกรรมศาสตร์บัณฑิต
สาขาวิชาเทคโนโลยีโทรคมนาคม
ภาควิชาเทคนิคอุตสาหกรรม
คณะวิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2538

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า

หัวข้อปริญญานิพนธ์ ระบบตอบรับโทรศัพท์อัตโนมัติ

AUTOMATIC TELEPHONE ANSWER BY MICROCOMPUTER

ชื่อนักศึกษา 1. นายคมสรรค์ ยศสินศักดิ์ 37012044

2. นายพงศ์พิพัฒน์ วุฒาพาณิชย์ 37012073

อาจารย์ที่ปรึกษา อาจารย์กฤดากร กล่อมการ

ภาควิชา เทคนิคอุตสาหกรรม

ปีการศึกษา 2538

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง อนุมัติให้รับปริญญานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาอุตสาหกรรมศาสตร์บัณฑิต

คณะกรรมการสอบปริญญานิพนธ์

..... ประธานกรรมการ

(.....)

..... กรรมการ

(.....)

..... กรรมการ

(.....)

..... กรรมการ

(.....)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ระบบตอบรับโทรศัพท์อัตโนมัติ

โดย นายคมสรรค์ ยศสินศักดิ์
นายพงศ์พิพัฒน์ วุฒาพาณิชย์

อาจารย์ที่ปรึกษา อาจารย์ กฤดากร กล่อมการ

ปีการศึกษา 2538

.....

บทคัดย่อ

ระบบตอบรับโทรศัพท์อัตโนมัติ เป็นการนำไมโครคอมพิวเตอร์มาต่อร่วมกับ voice modem เพื่อให้สามารถรับโทรศัพท์ รวมทั้ง เก็บข้อมูลต่างๆที่ต้องการ ซึ่งแตกต่างจากเครื่องรับโทรศัพท์ที่มีขายอยู่ตามท้องตลาดตรงที่ สามารถรับคำสั่งบางคำสั่งจากการโทรศัพท์เข้ามาจากภายนอก เช่น เจ้าของบ้านสามารถโทรศัพท์ เข้ามาฟังข้อความได้ถึงแม้จะไม่อยู่บ้าน สามารถลบข้อความจากภายนอกได้เป็นต้น และยังสามารถพัฒนาไปสู่ศูนย์เพจเจอร์ได้ในอนาคต โดยอาศัย Visual Basic เป็น Solf ware ในการช่วยให้ไมรโครคอมพิวเตอร์ปฏิบัติงานต่างๆตามที่ผู้ใช้ต้องการ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

AUTOMATIC TELEPHONE ANSWER BY MICROCOMPETER

BY MR. KOMSAN YOESINSAK

MR. PONGPIPAT WUTAPANICH

ADVISOR MR. KITDAKORN KOMKARN

YEAR 1995

.....

ABSTRACT

This Thesis is concerned abount design Automatic telephone answer by using microcomputer operate with voice modem . The main different feature of this telephone answer is mass storage of voice by using harddisk and can access an Instruction form external telephone . The Soft Ware of this project has been developed by Visual Basic .

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กิตติกรรมประกาศ

การที่ปริญญานิพนธ์นี้สำเร็จลุล่วงไปด้วยดีนั้น คณะผู้จัดทำ จึงเรียนมาเพื่อขอขอบพระคุณท่านอาจารย์ที่ปรึกษาปริญญานิพนธ์เป็นอย่างมากที่ได้ให้ข้อเสนอแนะและแนวทางการแก้ปัญหาต่างๆรวมทั้งให้การสนับสนุนด้านเครื่องมือ และอุปกรณ์ที่ใช้ในการทดลองและขอขอบคุณเพื่อนทุกคน ที่ได้ให้คำแนะนำ และเป็นกำลังใจในการทำปริญญานิพนธ์ในครั้งนี้ และสุดท้ายนี้ขอขอบคุณเพื่อนร่วมงานทุกคน ที่ได้ให้ความร่วมมือ จนกระทั่งปริญญานิพนธ์นี้สำเร็จลุล่วงไปด้วยดี

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ

	หน้า
บทที่ 1 การสื่อสารข้อมูลแบบอนุกรม	1
1.1 มาตรฐานการเชื่อมต่อแบบ RS-232-C	1
1.2 สัญญาณข้อมูล	4
1.3 วงจรควบคุม	5
1.4 การส่งข้อมูลแบบอนุกรม	6
1.5 พอร์ตสื่อสารอนุกรมของเครื่อง PC	7
1.6 รหัส ASCII	9
บทที่ 2 รีจิสเตอร์ของ UART	11
บทที่ 3 เทคโนโลยีของโมเด็ม	20
3.1 เทคโนโลยีของโมเด็ม	20
3.2 ประวัติโดยย่อของโมเด็ม	21
3.3 มาตราฐานโมเด็ม	25
3.4 มาตราฐานการสื่อสารของโมเด็ม	26
3.5 ประเภทช่องโมเด็ม	30
3.6 โมเด็มกับการบีบอัดข้อมูล	32
3.7 การเชื่อมต่อโมเด็มเข้ากับคอมพิวเตอร์	34
บทที่ 4 ชุดคำสั่ง AT	35
บทที่ 5 แนะนำ Visual Basic	49
5.1 ความเป็นมา	50
5.2 ภาพรวมของ Visual Basic	51
5.3 หลักการโปรแกรมเชิงภาพของ Visual Basic	52
บทที่ 6 หลักการเขียนโปรแกรม	53
เอกสารอ้างอิง	59
ภาคผนวก	60

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญภาพ

หน้า

รูปที่ 1.1 แสดงการต่ออุปกรณ์สื่อสารผ่านทางพอร์ตอนุกรม RS-232-C 2

รูปที่ 1.2 สายนำสัญญาณที่ใช้ในมาตรฐาน RS-232-C 3

รูปที่ 1.3 แผนภาพการเชื่อมต่อโดยคร่าว ๆ ของชิปควบคุมการสื่อสารเบอร์ 8250 บนเครื่อง PC ทั่วไป 8

รูปที่ 1.4 ตัวอักษรพิเศษทั้ง 128 ตัวที่เพิ่มเติมขึ้นมาตามมาตรฐานของ IBM 9

รูปที่ 3.1 Microcom QX/V.32 33

รูปที่ 3.2 โมเด็มแบบ Rack-mounted 34

รูปที่ 4.1 กระบวนการทำงานของโมเด็มต้นทางปลายทาง 37

รูปที่ 5.1 ตัวอย่างแอพพลิเคชันที่พัฒนาด้วย Visual Basic 51

รูปที่ 5.2 การเขียนโปรแกรมแบบธรรมดากับแบบ Event Driven 53

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญตาราง

	หน้า
ตารางที่ 2.1 บิตต่างๆของรีจิสเตอร์ควบคุมสายสื่อสาร	12
ตารางที่ 2.2 การเลือกความยาวเวิร์ดในรีจิสเตอร์ควบคุมสายสื่อสาร	12
ตารางที่ 2.3 บิตต่างๆของรีจิสเตอร์ควบคุมโมเด็ม	13
ตารางที่ 2.4 บิตต่างๆของรีจิสเตอร์อีนาเบิลอินเตอร์รัปต์	14
ตารางที่ 2.5 ตัวหารอัตราบอดสำหรับอัตราบอดที่นิยมใช้	14
ตารางที่ 2.6 บิตต่างๆของรีจิสเตอร์แสดงสถานะสายสื่อสาร	16
ตารางที่ 2.7 บิตต่างๆของรีจิสเตอร์แสดงสถานะโมเด็ม	16
ตารางที่ 2.8 บิตต่างๆของรีจิสเตอร์จำแนกอินเตอร์รัปต์	17
ตารางที่ 4.1 พารามิเตอร์ที่ใช้ในการหมุนหมายเลขโทรศัพท์	40
ตารางที่ 4.2 คำสั่ง ATX	45
ตารางที่ 4.3 คำสั่ง AT&C	46
ตารางที่ 4.4 คำสั่ง AT&D	47
ตารางที่ 4.5 คำสั่ง AT&K	47

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 1
การสื่อสารข้อมูลแบบอนุกรม

เครื่อง PC สามารถที่จะติดต่อสื่อสารกับอุปกรณ์คอมพิวเตอร์อื่น ๆ ที่อยู่ห่างออกไปเป็นพัน ๆ ฟุตได้โดยติดต่อผ่านทางพอร์ตแบบสองทางที่ใช้เชื่อมเครื่อง PC ออกสู่โลกภายนอกโดยอาจจะต่อเข้ากับโมเด็ม เครื่อง PC เครื่องมินิคอมพิวเตอร์หรือเครื่องคอมพิวเตอร์เมนเฟรมก็ได้ และนอกจากนั้นพอร์ตอนุกรมยังเป็นช่องทางที่มักจะถูกใช้งานกับอุปกรณ์คอมพิวเตอร์อื่น ๆ ได้อีก เช่น เมาส์และเครื่องพิมพ์แบบอนุกรม

เครื่อง PC ทั่วไปจะมีพอร์ตขนานสำหรับใช้งานกับเครื่องพิมพ์ติดมาพร้อมกับเครื่องอยู่แล้ว การต่อแบบขนานนี้มีข้อดีคือ มีอัตราการส่งข้อมูลสูงกว่าการต่อแบบอนุกรม แต่ก็มีข้อเสียคือ มีข้อจำกัดในการต่อสายออกไปใช้งานไกล ๆ ซึ่งการใช้งานพอร์ตขนานปกตินั้นมักจะใช้งานอยู่ในระยะไม่เกินสิบฟุตจากเครื่องคอมพิวเตอร์เท่านั้น

1.1 มาตรฐานการเชื่อมต่อแบบ RS-232-C

การเชื่อมต่อแบบอนุกรมนั้นอาจจะกล่าวได้ว่าเป็นวิธีการเชื่อมต่อสื่อสารทั่วไปของอุปกรณ์ทางอิเล็กทรอนิกส์ต่าง ๆ ไม่ว่าจะมาจากผู้ผลิตรายใดก็ตาม แต่เนื่องจากว่าผู้ผลิตสินค้าทางด้านอิเล็กทรอนิกส์ต่าง ๆ ก็ออกแบบพอร์ตสื่อสารอนุกรมของตนขึ้นมา ทำให้มีความแตกต่างกันในรายละเอียดของพอร์ตหรือวิธีการในการรับส่งข้อมูล เพื่อให้แน่ใจว่าพอร์ตอนุกรมหนึ่ง ๆ นั้นสามารถที่จะติดต่อสื่อสารกับพอร์ตอนุกรมอื่น ๆ ได้โดยไม่มีปัญหาทางด้านเทคนิคองค์กรอุตสาหกรรมอิเล็กทรอนิกส์ของสหรัฐ (EIA) ก็ได้กำหนดมาตรฐานของการเชื่อมต่อ และคุณสมบัติของพอร์ตสื่อสารอนุกรมขึ้นมาในปี ค.ศ. 1969

ตัวอย่างองค์กร EIA เองนั้นนับได้ว่าเกี่ยวข้องโดยตรงกับอุตสาหกรรมอิเล็กทรอนิกส์ทั่วโลก ดังนั้นมาตรฐานดังกล่าวจึงกลายเป็นมาตรฐานสากลไปโดยปริยาย มาตรฐานอันหนึ่งที่ได้รับการรับรองโดย EIA แล้วก็คือมาตรฐานที่ 232 แบบ C หรือ เป็นที่รู้จักกันในชื่อของ RS-232-C (Regulation Standard No. 232 Type C) มาตรฐานการเชื่อมต่อแบบนี้มีความคล้ายคลึงกับมาตรฐาน CCITT V.24 และมีคุณสมบัติทางไฟฟ้าใกล้เคียงกับมาตรฐาน CCITT V.28 และนอกจากนั้นก็มีความใกล้เคียงกับมาตรฐาน Mil-Std-188C ของกระทรวงกลาโหมสหรัฐฯ ด้วย ซึ่งรายละเอียดของมาตรฐานต่าง ๆ นั้นจะกล่าวถึงอีกครั้งหนึ่ง

หลังจากนั้นในปี ค.ศ. 1970 EIA ได้กำหนดมาตรฐานใหม่ที่มีประสิทธิภาพสูงกว่ามาตรฐาน RS-232-C ออกมาใหม่ ซึ่งก็ได้แก่มาตรฐาน RS-449 , RS-422 และ RS-423 โดยที่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

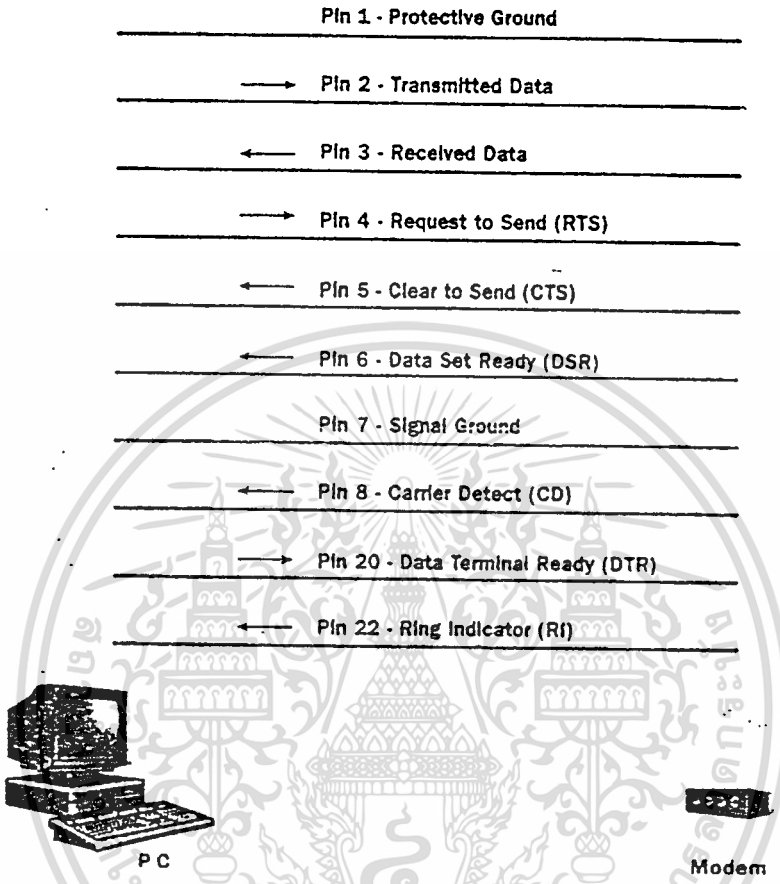
มาตรฐานใหม่เหล่านี้จะมีคุณสมบัติทางไฟฟ้าและทางกายภาพคล้ายๆ กันดังนั้นจึงถูกเรียกรวม ๆ ไปว่าเป็นกลุ่มมาตรฐาน RS-449 แต่อย่างไรก็ตามทางบริษัท IBM และบริษัทผู้ผลิตเครื่องคอมพิวเตอร์ตระกูล PC อื่น ๆ ก็ไม่ได้ให้ความสนใจมาตรฐานใหม่นี้เท่าไรนักถึงแม้ว่ามันจะไม่สามารถส่งข้อมูลได้เร็วและได้ระยะทางไกลกว่ามาตรฐาน RS-232-C ก็ตามมีเพียงแต่บริษัท Apple Computer เท่านั้นที่ได้นำเอามาตรฐานชุด RS-449 นี้ไปใช้กับผลิตภัณฑ์ Macintosh ของตน

มาตรฐาน RS-232-C นั้นแบ่งการเชื่อมต่อออกเป็นสองลักษณะคือการต่อกับเทอร์มินัล (DTE : Data Terminal Equipment) และการต่อกับอุปกรณ์สื่อสารข้อมูล (DCT : Data Communication Equipment) ซึ่งในกรณีปกตินั้น DCE จะต้องต่อเข้ากับ DCE เสมอยกตัวอย่าง เช่น การต่อกับโมเด็มเข้ากับเครื่อง PC โดยเครื่อง PC จะเป็นอุปกรณ์ DCE

รูปที่ 1.1 แสดงการต่ออุปกรณ์สื่อสารผ่านทางพอร์ตอนุกรม RS-232-C

การเชื่อมต่อตามมาตรฐาน RS-232-C นั้นโดยปกติจะคอนเน็กเตอร์รูปตัว D ชนิด 25 ขา กำหนดให้ปลายสายสัญญาณด้านหนึ่งที่เป็นคอนเน็กเตอร์ตัวผู้ใช้ต่ออยู่กับอุปกรณ์ DCE และปลายของสัญญาณอีกด้านหนึ่งจะต้องเป็นคอนเน็กเตอร์ตัวเมียใช้ต่ออยู่กับอุปกรณ์ DCE แต่ว่ากฎและมาตรฐานนั้นมักจะถูกเปลี่ยนแปลงอยู่เสมอ ดังนั้นผู้ผลิตอุปกรณ์คอมพิวเตอร์อื่น ๆ ก็ได้ออกแบบมาตรฐานคอนเน็กเตอร์แบบใหม่ออกมาเพื่อใช้แทนมาตรฐาน RS-232-C เดิม อย่างเช่น บริษัท IBM นั้นก็ได้ออกแบบคอนเน็กเตอร์แบบ 9 ขาขึ้นมาใช้แทนคอนเน็กเตอร์แบบ 25 ขาและได้กำหนดให้เป็นมาตรฐานบนเครื่อง IBM AT ซึ่งเครื่อง IBM ในช่วงนั้นยังเป็นผู้นำทางด้านการพัฒนาเครื่องคอมพิวเตอร์ส่วนบุคคลอยู่ ดังนั้นผู้ผลิตรายอื่น ๆ ก็ได้นำมาตรฐานคอนเน็กเตอร์แบบใหม่ของ IBM ไปใช้เป็นมาตรฐานในผลิตภัณฑ์ของตนด้วย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 1.2 สายนำสัญญาณที่ใช้ในมาตรฐาน RS-232-C นั้นจะมีทั้งหมด 9 เส้นประกอบด้วยวงจรข้อมูลสองวงจร คือขานำสัญญาณหมายเลข 2 และ 3 ซึ่งจะใช้ในการส่งผ่านข้อมูลระหว่างเครื่อง PC และโมเด็ม ขานำสัญญาณหมายเลข 1 และ 7 เป็นกราวด์ส่วนวงจรชุดที่สองจะประกอบไปด้วย ขานำสัญญาณหมายเลข 4 และ 5 เป็นวงจรควบคุมสัญญาณที่เรียกว่า RTS และ CTS ซึ่งจะใช้ในการควบคุมการไหลของข้อมูลระหว่าง PC และโมเด็ม (Hardware Flow Control) ขานำสัญญาณขา 8 เป็นขาสัญญาณ Carrier Detect (CD) ใช้บอก PC ว่าในขณะนั้นโมเด็มได้รับสัญญาณพาหะอยู่หรือไม่ ขานำสัญญาณหมายเลข 20 (DTR : Data Terminal Ready) ใช้บอกโมเด็มว่าในขณะนั้นเทอร์มินอล (หรือ PC) พร้อมที่จะติดต่อกับโมเด็มหรือไม่ ส่วนขานำสัญญาณหามายเลข 22 จะใช้เป็นสัญญาณแสดงว่าในขณะนั้นโมเด็มได้รับสัญญาณกระดิ่งหรือไม่ ซึ่งมักจะไม่ได้ถูกใช้งานมากนัก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ในการสื่อสารแบบอนุกรมนั้น การที่จะทำให้อุปกรณ์อิเล็กทรอนิกส์สื่อสารกันได้ อย่างน้อยก็จำเป็นจะต้องใช้สายไฟฟ้าเชื่อมต่อระหว่างอุปกรณ์สองตัวไม่ต่ำกว่าสองเส้น แต่จะเห็นว่ามาตรฐาน RS-232-C นั้นกำหนดให้คอนเน็กเตอร์มีขาสัญญาณถึง 25 ขา ซึ่งอาจจะดูเหมือนว่าจะดูฟุ่มเฟือยเกินความจำเป็นจนเกินไป เหตุผลที่ต้องใช้คอนเน็กเตอร์แบบ 25 ขานี้ก็เพราะว่าอุปกรณ์สื่อสารแบบอนุกรมตามมาตรฐานนั้น จะต้องมีวงจรไฟฟ้าที่เกี่ยวข้องกับการรับส่งข้อมูลอยู่สามวงจรด้วยกันก็คือ วงจรรับข้อมูล วงจรส่งข้อมูล และ วงจรควบคุม รูปที่1.2 จะแสดงผังการเชื่อมต่อแบบอนุกรมตามมาตรฐาน RS-232-C แบบคร่าว ๆ โดยจะแสดงเฉพาะการเชื่อมต่อสัญญาณที่สำคัญในการใช้งานเพียง 10 เส้นเท่านั้น จากรูปจะเห็นได้ว่า RS-232-C นั้นมีสัญญาณที่เกี่ยวข้องกับการสื่อสารแบบอนุกรมอยู่จำนวนมาก ดังที่จะกล่าวถึงรายละเอียดแยกเป็นประเภทดังต่อไปนี้

1.2 สัญญาณข้อมูล

ส่วนที่นับได้ว่าเป็นส่วนที่สำคัญที่สุดของมาตรฐาน RS-232-C ก็คือวงจรสำหรับส่งข้อมูล ซึ่งมีอยู่สองวงจรที่ทำหน้าที่ส่งข้อมูลจาก DTE ไป DCE และอีกวงจรหนึ่งก็จะทำหน้าที่ตรงกันข้ามกับวงจรแรกก็คือวงจรที่มีหน้าที่รับข้อมูลจาก DCE เพื่อส่งไปให้ DTE

จากรูปที่ 1.2 จะเห็นว่าเครื่อง PC นั้นจะส่งข้อมูลไปให้โมเด็มทางขานำสัญญาณหมายเลข 2 และรับข้อมูลกลับจากโมเด็มทางขานำสัญญาณหมายเลข 3 โดยใช้ขานำสัญญาณหมายเลข 7 เป็นกราวด์ร่วมของทั้งสองวงจร

1.2.1 สัญญาณทางไฟฟ้าที่ขานำสัญญาณหมายเลข 2 ,3 และ 7

มาตรฐาน RS-232-C นั้นจะให้ความสำคัญกับวิธีการส่งผ่านข้อมูลลงบนสายนำสัญญาณเป็นอย่างมาก โดยรูปแบบสัญญาณทางไฟฟ้าที่ใช้แทนข้อมูลนั้นก็เป็นรูปแบบสัญญาณง่าย ๆ คือสัญญาณสี่เหลี่ยมที่ถูกสร้างขึ้นจากไฟฟ้ากระแสตรงระดับแรงดันประมาณ 3 ถึง 25 โวลต์ แทนข้อมูล " 0 " และ -3 ถึง -25 โวลต์จะแทนข้อมูล "1" สำหรับช่วงระดับแรงดัน -3 ถึง 3 โวลต์นั้นจะต้องเป็นช่วงระดับแรงดันที่ใช้ในการแบ่งแยกระดับสถานะของสัญญาณระหว่างสถานะ "0" และสถานะ "1"

คุณสมบัติของสัญญาณแบบสี่เหลี่ยมนี้เมื่อถูกส่งออกมาในสายไฟฟ้าแบบธรรมดามักจะเกิดความผิดเพี้ยนได้ง่าย เนื่องจากข้อกำหนดของสายสัญญาณหรืออาจจะมีสาเหตุมาจากสัญญาณรบกวนจากภายนอกอื่น ๆ ทำให้การรับส่งข้อมูลมีระยะใช้งานอยู่ในช่วงที่จำกัด โดยเฉพาะการรับส่งข้อมูลที่มีความเร็วสูงนั้นคลื่นสี่เหลี่ยมจะเรียงชิดติดกัน จึงยิ่งทำให้มีโอกาสเกิดความผิดพลาดของการรับส่งข้อมูลได้ง่าย ดังนั้นมาตรฐาน RS-232-C ที่ความเร็วการส่งข้อมูล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

9600 BPS ก็จะมีระยะใช้งานได้ที่ระยะทางไม่น้อยกว่าห้าสิบฟุตเป็นต้น แต่อย่างไรก็ตามคุณภาพของสัญญาณก็จะขึ้นกับคุณภาพของสายนำสัญญาณด้วยเช่นกัน

1.8 วงจรควบคุม

วงจรควบคุมของ RS-232-C นั้นมีอยู่ทั้งหมดห้าวงจร ซึ่งแต่ละวงจรจะมีหน้าที่ในการสร้างสัญญาณควบคุมต่างๆ ขึ้นมาเพื่อทำให้เครื่อง PC และโมเด็มทราบสถานะในการทำงานของกันและกันว่าอยู่ในสถานะใด โดยที่สัญญาณในวงจรควบคุมนั้นจะมีลักษณะทางกายภาพเช่นเดียวกันกับสัญญาณที่ปรากฏอยู่ในวงจรข้อมูล แต่โมเด็มส่วนใหญ่นั้นไม่ได้ใช้วงจรควบคุมครบทุกวงจร นอกจากนี้ผู้ใช้ยังสามารถที่จะกำหนดให้โมเด็ม หรือ PC เพิกเฉยต่อสัญญาณควบคุมบางอย่างได้ตามต้องการ

ในการติดตั้งโมเด็มความเร็วสูงตั้งแต่ 9600 BPS ขึ้นไปนั้นการกำหนดพารามิเตอร์เกี่ยวกับสัญญาณควบคุมต่าง (Handshaking) เป็นเทคนิคสำคัญที่ช่วยแก้ปัญหา หรือช่วยเพิ่มความเร็วในการรับส่งข้อมูลได้ ซึ่งจะกล่าวถึงการโปรแกรม และกำหนดพารามิเตอร์ต่าง ๆ ของโมเด็ม ซึ่งการกำหนดพารามิเตอร์ของโมเด็มที่เกี่ยวกับสัญญาณควบคุมต่าง ๆ ให้สอดคล้องกับการทำงานนั้น ผู้ใช้ก็ควรจะมีความเข้าใจพื้นฐานเกี่ยวกับสัญญาณควบคุมต่าง ๆ เสียก่อน ซึ่งมีรายละเอียดดังต่อไปนี้

1.8.1 Request To Send และ Clear TO Send (ขานำสัญญาณหมายเลข 4 และ 5)

วงจร RTS และ CTS นั้นเป็นวงจรที่ใช้สำหรับการควบคุมการส่งสัญญาณข้อมูลระหว่างเครื่อง PC และโมเด็ม โดยโมเด็มจะส่งสัญญาณ CTS สถานะ “ON” ให้แก่ PC เมื่อโมเด็มพร้อมที่จะรับข้อมูล และ ส่งสัญญาณ CTS สถานะ “OFF” เมื่อโมเด็มนั้นไม่พร้อมที่จะรับข้อมูลจากเครื่อง PC ได้ ในทำนองเดียวกันเครื่อง PC ก็จะให้สัญญาณ RTS สถานะ “ON” เมื่อ PC พร้อมที่จะรับข้อมูล และส่งสัญญาณ RTS สถานะ “OFF” เมื่อ PC ไม่พร้อมที่จะรับข้อมูลจากโมเด็ม วงจรควบคุมการส่งผ่านข้อมูลระหว่างเครื่อง PC และโมเด็มนั้นเอง ซึ่งสัญญาณ RTS และ CTS นี้จะมีประโยชน์มากในการสื่อสารข้อมูลด้วยความเร็วสูงโดยสามารถป้องกันการล้นและการสูญหายของข้อมูลได้เป็นอย่างดี

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1.3.2 Data Terminal Ready (ขานำสัญญาณหมายเลข 20) และ Data Set Ready (ขานำสัญญาณหมายเลข)

สัญญาณ DTR จะใช้เป็นสิ่งที่บอกโมเด็มทราบว่าเครื่อง PC นั้นกำลังอยู่ในสถานะที่พร้อมที่จะติดต่อสื่อสารกับโมเด็มหรือไม่ และในกรณีเดียวกันสัญญาณ DSR นั้นก็จะใช้เป็นสิ่งที่บอกให้เครื่อง PC ทราบว่าโมเด็ม ก็พร้อมที่จะติดต่อกับ PC หรือไม่ โดยที่สัญญาณ DSR นั้นจะอยู่ในสถานะ "ON" ก็ต่อเมื่อโมเด็มได้รับสัญญาณ DTR แล้ว

1.3.3 Carrier Detect (ขานำสัญญาณหมายเลข 8) และ Ring Indicator (ขานำสัญญาณหมายเลข 22)

สัญญาณ CD นี้จะใช้เป็นสิ่งที่บอก PC ให้ทราบว่าโมเด็มนั้นกำลังเชื่อมต่อกับโมเด็มเครื่องอื่น ๆ และกำลังได้รับสัญญาณพาหะจากโมเด็มปลายทางส่วนสัญญาณ RI นั้นก็จะบอกเครื่อง PC ให้ทราบว่ามีสัญญาณกระดิ่งโทรศัพท์เรียกเข้ามาที่โมเด็ม ซึ่งโมเด็มส่วนใหญ่ในปัจจุบันก็มีวงจรที่ใช้ตอบรับโทรศัพท์โดยอัตโนมัติ (Auto-answer) อยู่ภายในตัวเองแล้วจึงไม่จำเป็นที่จะต้องใช้งานสัญญาณ RI นี้ แต่ในบางกรณีโปรแกรมบางประเภทก็อาจจะใช้สัญญาณ RI นี้เป็นตัวกำหนดให้เริ่มทำการรันโปรแกรมอื่น ๆ ได้เช่นกัน

1.4 การส่งข้อมูลแบบอนุกรม

ข้อมูลที่ถูกส่งไปแบบอนุกรมนั้นโดยปกตินั้น โดยปรกติแล้วจะเป็นข้อมูล "0" หรือ "1" เรียงต่อกันเป็นชุด ๆ ซึ่งปกติแล้วข้อมูลที่ใช้สื่อสารกันทั่วไป จะประกอบไปด้วยข้อมูลจำนวนแปดบิตต่อหนึ่งตัวอักษร โดยข้อมูลแต่ละชุดนี้จะสามารถแทนตัวอักษรที่แตกต่างกันได้ถึง 256 แบบ ในเครื่อง PC ข้อมูลจะถูกเก็บอยู่ในรูปของข้อมูลขนาดแปดบิตหรือ 1 ไบต์อยู่แล้วไม่ว่าจะใช้ระบบปฏิบัติการ DOS , OS/2 หรือใช้เครื่อง PC ชนิด 32 บิตก็ตาม

ข้อมูลบางแบบนั้นอาจจะมีขนาดเพียงเจ็ดบิต อย่างเช่นรหัส ASCII ที่ใช้กันในคอมพิวเตอร์ทั่ว ๆ ไปนั้นก็เป็นรหัสขนาดเจ็ดบิตดังนั้นมันจึงใช้แทนตัวอักษรได้เพียง 128 ตัวเท่านั้น ยกตัวอย่างเช่นตัวอักษร A นั้นจะมีรหัส ASCII ที่ใช้แทนมันคือรหัสหมายเลข 65 ซึ่งจะขอกล่าวถึงเรื่องของรหัส ASCII กันอีกครั้งหนึ่งในหัวข้อต่อไป

ข้อมูลแบบอนุกรมจะถูกส่งไปตามสายนำสัญญาณโดยจะทยอยส่งออกไปทีละบิต ซึ่งบิตต่ำจะถูกส่งออกไปก่อนแล้วติดตามด้วยบิตที่สูงขึ้นมาเรื่อย ๆจนครบชุดข้อมูลหนึ่ง ๆ สมมุติว่าถ้าต้องการที่จะส่งตัวอักษร A ออกไปแบบอนุกรม ซึ่งตัวอักษร A นั้นมีรหัส ASCII ที่เป็นเลข

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ฐานสองคือ 01000001 ดังนั้นบิตที่ถูกส่งออกไป ก่อนก็คือบิตที่อยู่ทางขวาสุดซึ่งก็คือ 1 นั่นเอง และหลังจากนั้นก็เป็น 0 ตัวถัดมาทางซ้ายเรียงต่อ ๆ กันไปตามลำดับไปจนครบข้อมูลหนึ่งชุด

1.4.1 การสื่อสารแบบ Synchronous และ Asynchronous

การสื่อสารของเครื่อง PC ส่วนใหญ่นั้นเป็นแบบ Asynchronous ซึ่งหมายความว่าไม่มีการกำหนดช่วงเวลาตายตัวที่ใช้ส่งข้อมูลหนึ่งตัวอักษรออกไปที่พอร์ตอนุกรม นั่นก็หมายความว่าการส่งข้อมูลแต่ละครั้งนั้นจะต้องมีบิตเริ่มและบิตหยุดเป็นตัวกำหนดขอบเขตของข้อมูล บิตที่แสดงขอบเขตนี้จะถูกสร้างขึ้นโดยอัตโนมัติจากวงจรสื่อสาร แต่สำหรับการสื่อสารแบบ Synchronous นั้นบิตเริ่มและบิตหยุดไม่มีความจำเป็นเนื่องจากว่าได้กำหนดช่วงเวลาการส่งข้อมูลหนึ่งตัวอักษรเอาไว้แล้ว

เมื่อ PC ได้ทำการเชื่อมต่อเข้าเครื่องคอมพิวเตอร์โฮสอื่น ๆ อย่างเช่นศูนย์ BBS หรือศูนย์บริการข้อมูลอื่น ๆ เช่น CompuServe โมเด็มจะใช้การติดต่อแบบ Asynchonous เสมอ ส่วนการสื่อสารแบบ Synchronous นั้นจะใช้เฉพาะในงานพิเศษ ๆ ที่ต้องการความเร็วในการส่งข้อมูลสูงมาก เช่นการส่งข้อมูลระหว่าง LAN หรือ ระบบเครือข่าย Packet Switching ซึ่งวงจรสื่อสารที่พบในเครื่อง PC ทั่ว ๆ ไป นั้นไม่สามารถทำงานแบบ Synchronous ได้

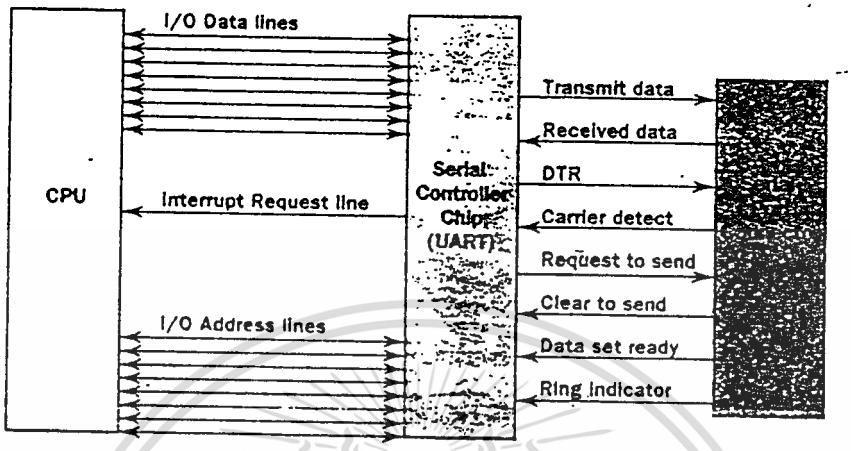
การสื่อสารแบบ Synchronous นั้นมีประสิทธิภาพการทำงานที่สูงกว่าแบบ Asynchronous มากแต่จะต้องการวงจรพิเศษที่ต้องใช้สำหรับการสื่อสารแบบ Synchronous โดยเฉพาะข้อมูลที่ส่งแบบ Synchronous นั้นจะไม่มีการขาดตอน โดยข้อมูลจะถูกแยกออกเป็นชุด ๆ จากสัญญาณนาฬิกาที่ส่งไปพร้อม ๆ กันในสายนำสัญญาณอีกเส้นหนึ่ง ดังนั้นจึงมีความเร็วในการส่งข้อมูลที่สูงมาก

1.5 พอร์ตสื่อสารอนุกรมของเครื่อง PC

ที่ได้กล่าวมาข้างต้นแล้วว่าคอนเน็กเตอร์แบบ RS-232-C นั้นจะมีอยู่สองแบบก็คือแบบ 25 ขาและแบบ 9 ขา แต่อย่างไรก็ตามการอินเตอร์เฟส RS-232-C เข้ากับเครื่อง PC นั้นไม่ได้มีเฉพาะข้อกำหนดทางด้านกายภาพของคอนเน็กเตอร์เท่านั้น ด้วยเหตุนี้เองการสื่อสารผ่านพอร์ตอนุกรมของ PC นั้น จำเป็นที่จะต้องมีชิปที่ใช้สำหรับควบคุมการสื่อสารข้อมูล โดยเฉพาะ ซึ่งชิปนี้อาจจะอยู่บนแผ่นวงจรหลัก หรือบนแผ่นวงจรควบคุมอุปกรณ์เอนกประสงค์ (Multi I/O Card) ก็ได้ โดยที่ชิปดังกล่าวที่ใช้กันทั่วไปก็คือชิปการควบคุมการสื่อสารเบอร์ 8250 ของบริษัท Western Digital และชิปที่ใหม่กว่าก็คือชิปเบอร์ 16550A หรืออาจจะเป็นชิปอื่น ๆ ที่มีคุณสมบัติ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เท่าเทียมกัน ตัวชิปประเภทนี้จะทำหน้าที่อินเตอร์เฟสข้อมูลแบบอนุกรมเข้ากับระบบบัสข้อมูลของ PC ตามแผนภาพที่แสดงไว้ในรูปที่ 1.3



รูปที่ 1.3 แผนภาพการเชื่อมต่อโดยคร่าว ๆ ของชิปควบคุมการสื่อสารเบอร์ 8250 บนเครื่อง PC ทั่วไป

เนื่องจากว่าชิปนี้จะเชื่อมต่ออยู่กับบัสข้อมูลของ PC และวงจรสื่อสารแบบอนุกรมในแต่ละพอร์ตก็จะถูกกำหนดให้มีแอดเดรส I/O ประจำพอร์ตดังนั้นจึงทำให้ CPU สามารถอ่านเขียนข้อมูลที่พอร์ตได้โดยตรงเมื่อ CPU ต้องการ

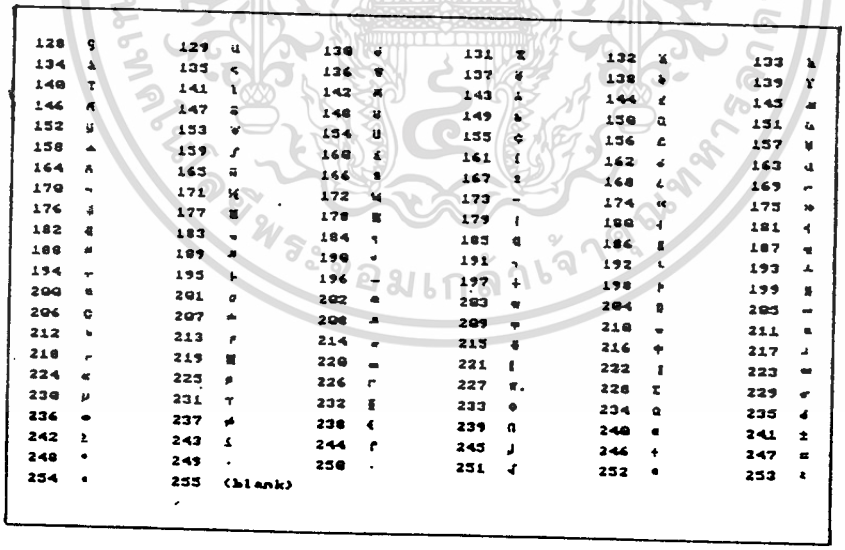
ในอดีตวิศวกรของบริษัท IBM ได้กำหนดให้เครื่อง IBM PC มีพอร์ตสื่อสารแบบอนุกรมได้เพียงสองพอร์ต และใช้งาน IRQ ได้เพียวสองความหมายเลขเท่านั้น ดังนั้นเครื่อง IBM PC และเครื่องที่เข้ากันได้กับ IBM PC อื่น ๆในยุคนั้นจึงสามารถใช้งานพอร์ตอนุกรมได้มากกว่าสองพอร์ต แต่ในปัจจุบันก็ได้มีการแก้ไขโดยการกำหนดให้เครื่องรุ่นใหม่อย่างเช่นเครื่อง IBM PC/AT หรือเครื่อง PS/2 สามารถมีจำนวนพอร์ตสื่อสารแบบอนุกรม (COM Port) ได้มากขึ้น และบางพอร์ตก็สามารถใช้ IRQ ร่วมกันได้โดยที่แต่ละพอร์ตจะมีหมายเลขแอดเดรสที่คล้ายกันและมี IRQ ให้ใช้พอร์ตละ 1 หมายเลข ซึ่งมาตรฐานของเครื่อง PC ทุกรุ่นจะกำหนดให้พอร์ตที่เรียกว่า COM1 นั้นใช้แอดเดรสที่ 3F8h และ พอร์ต COM2 ใช้แอดเดรสที่ 2F8h ส่วนด้านสัญญาณขัดจังหวะนั้น COM1 จะใช้ IRQ4 และ COM2 จะใช้ IRQ3 และ สำหรับเครื่อง PS/2 นั้นการกำหนดแอดเดรส และ IRQ ของทั้ง COM1 และ COM2 จะเป็นเช่นเดียวกันกับใน PC

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1.6 รหัส ASCII

ในขณะที่อ่านเอกสารหรือหนังสือต่าง ๆ ตาของคุณจะทำหน้าที่กวาดไปตามตัวหนังสือรูปแบบต่าง ๆ ที่พิมพ์ไว้บนกระดาษ และรวบรวมข้อมูลต่าง ๆ เข้ามาประมวลผลในสมอง แต่คอมพิวเตอร์จะรู้จักเพียงแค่ "0" และ "1" เหล่านี้นำมาแปลงให้เป็นตัวหนังสือที่มนุษย์เข้าใจได้ วิธีการหนึ่งที่เป็นนิยมก็คือกำหนดรหัสเลขฐานสองขึ้นมาใช้แทนตัวอักษรที่จะใช้งาน โดยตัวอักษรแต่ละตัวจะมีรหัสเลขฐานสองเฉพาะตัวของมันซึ่งมาตรฐานของรหัสเหล่านี้ที่เป็นที่นิยมใช้กันทั่วไปจะมีชื่อว่ารหัส ASCII (American Standard Code for Information Interchange) ดังแสดงไว้ที่ตารางชุดอักษร ASCII ในภาคผนวก ก จากตารางจะสังเกตได้ว่าอักษรทุกตัวไม่ว่าจะเป็นตัวอักษรที่พิมพ์ดีดหรือพิมพ์ไม่ได้ก็ตามจะต้องมีหมายเลขรหัสประจำตัวอักษรเสมอ

รหัส ASCII นั้นประกอบด้วยตัวอักษรและสัญลักษณ์ที่พิมพ์ได้ 96 ตัวและ 32 ตัวเป็นอักษรที่พิมพ์ไม่ได้รวมเป็นจำนวนทั้งสิ้น 128 ตัวอักษร ซึ่งใช้ขนาดข้อมูลจำนวน 7 บิตพอดี (27) แต่ IBM ก็ได้กำหนดชุดอักษรรหัส ASCII เพิ่มเติมขึ้นมาอีก 128 ตัว โดยใช้ข้อมูลขนาด 8 บิต ซึ่งชุดอักษรเพิ่มเติมนี้จะมีใช้งานในระบบของ IBM PC และ เครื่องที่เข้ากันได้เท่านั้น ดังแสดงไว้ในรูป 1.4



รูปที่ 1.4 ตัวอักษรพิเศษทั้ง 128 ตัวที่เพิ่มเติมขึ้นมาตามมาตรฐานของ IBM

การเกิดมาตรฐานของรหัส ASCII ขึ้นมานั้นทำให้เครื่องคอมพิวเตอร์และอุปกรณ์จากผู้ผลิตต่างกัน สามารถติดต่อกันได้โดยไม่มีปัญหารหัสไม่ตรงกัน เช่นเมื่ออ้างถึงรหัส ASCII

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หมายเลข 65 ก็จะหมายถึงตัวอักษร A ไม่ว่าจะเป็นอุปกรณ์ชนิดใดหรือเป็นอุปกรณ์จากผู้ผลิตราย ใด ๆ ก็ตามก็จะหมายถึงตัวอักษร A เสมอ

1.6.1 ตัวอักษรควบคุมของ ASCII

มาตรฐานรหัส ASCII นั้นถูกออกแบบขึ้นมาเพื่อใช้ทดแทนมาตรฐานเดิมที่มีชื่อว่า BAUDOT ซึ่งมีจำนวนบิตเพียง 5 บิตต่ออักษรซึ่งใช้งานกันอยู่กับเครื่องโทรพิมพ์แต่เนื่องจากการใช้ขนาดของข้อมูลเพียงห้าบิตนี่เอง ทำให้ตัวอักษรที่สามารถใช้ได้จึงมีจำนวนได้เพียง 32 ตัวเท่านั้น ซึ่งก็หมายความว่าจะแทนอักษรแบบตัวพิมพ์ใหญ่ได้เท่านั้น

คงยังจำกันได้ว่ารหัส ASCII นั้นมีตัวอักษรที่ไม่สามารถพิมพ์ได้ 32 ตัวซึ่งก็คือ ASCII หมายเลขตั้งแต่ 0 ถึง 31 เรียกว่ารหัสควบคุม ทั้งนี้ก็เพราะว่ารหัสในช่วงดังกล่าวนี้เดิมถูกแบบมาเพื่อใช้งานกับเครื่องโทรพิมพ์ที่อยู่ไกลออกไปเช่นการเลื่อนบรรทัดหรือเลื่อนหน้ากระดาษ ตัวอักษรในกลุ่มนี้ดูเหมือนว่าจะไม่มีประโยชน์สำหรับเครื่อง PC ในปัจจุบัน และบางตัวก็ได้ถูกกำหนดให้มีความหมายใหม่สำหรับการใช้งานเฉพาะอย่าง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 2

รีจิสเตอร์ของ UART

ในบทนี้จะกล่าวถึงรายละเอียดของ UART (Universal Asynchronous Reiver and Transmitter) ที่นิยมใช้ในการสื่อสารข้อมูลแบบอนุกรมบนพีซี ซึ่งจำเป็นสำหรับผู้ใช้ที่ต้องการเขียนโปรแกรมสื่อสารในระดับของระบบสื่อสารแบบอนุกรม

2.1 รีจิสเตอร์ของ UART

UART มีรีจิสเตอร์ภายในเช่นเดียวกับซีพียู รีจิสเตอร์ของมันแบ่งออกไปเป็น 3 ประเภท คือ

1. รีจิสเตอร์ควบคุม (Control register) ทำหน้าที่รับคำสั่งจากซีพียู
2. รีจิสเตอร์สถานะ (Status register) ทำหน้าที่เก็บสถานะของเหตุการณ์ที่เกิดขึ้นใน UART
3. รีจิสเตอร์พักข้อมูล (Buffer register) เป็นที่เก็บข้อมูลหรือรอการส่ง หรือรอการประมวลผล

1. รีจิสเตอร์ควบคุม

รีจิสเตอร์ควบคุมมี 4 ตัว คือ รีจิสเตอร์ควบคุมสายสื่อสาร (Line Control register) รีจิสเตอร์ควบคุมโมเด็ม (Modem Control register) รีจิสเตอร์อีนาเบิล อินเตอร์รัปต์ (Interrupt Enable register) และ รีจิสเตอร์ตัวหาอัตราบอด (Baud Rate Divisor Latches)

1.1 รีจิสเตอร์ควบคุมสายสื่อสาร

รีจิสเตอร์ควบคุมสายสื่อสารถูกใช้สำหรับตั้งพารามิเตอร์การสื่อสาร แต่ละบิต ของรีจิสเตอร์มีความหมายตามตารางที่ 12.1 และตารางที่ 12.2 แสดงความหมายของบิต 0 และ 1 การทำงานของแต่ละบิตสรุปได้ดังนี้

1. บิต 0 และ 1 กำหนดความยาวเวิร์ด
2. กำหนดจำนวนบิตจบ ถ้าบิตนี้เป็นศูนย์ จะใช้ หนึ่ง บิตจบ ถ้าบิตนี้เป็น หนึ่ง จะใช้สองบิตจบ ยกเว้นความยาวห้าบิต บิตจบจะยาวหนึ่งบิตครึ่ง
3. บิต 3 อีนาเบิล พาริตี้ ถ้าเป็นศูนย์คือไม่มีพาริตี้ ถ้าเป็นหนึ่งคือมีการใช้พาริตี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4. บิต 4 ถ้าเป็นหนึ่ง คือเลือกใช้พาริตี้คู่ และใช้พาริตี้คี่ถ้าบิตนี้เป็นศูนย์ บิต 4 จะไม่มีความหมายถ้าบิต 3 ไม่ถูกเซตด้วย

5. บิต 5 ถ้าถูกเซตจะใช้ศูนย์เป็นพาริตี้ ถ้าบิต 5 เป็นศูนย์พาริตี้จะเป็นหนึ่ง

6. บิต 6 ถูกใช้เพื่อสร้างคำสั่งเบรก เอาท์พุทจะถูกทำให้เป็นศูนย์จนกว่าบิต 6 จะเป็นศูนย์

7. บิต 7 จะถูกเรียกว่า บิตควบคุมการเข้าแลตช์ตัวหาร (Divisor Latch access bit หรือ DLAB) ถ้ามันถูกเซตเป็นหนึ่ง การดำเนินการอ่านและเขียน จะทำการแลตช์ตัวหาร (Divisor Latches) ของตัวสร้างอัตราบอด (Baud rate generator) ถ้ามันถูกเซตเป็นศูย์การเขียน และอ่านจะกระทำต่อรีจิสเตอร์พักข้อมูล หรือ รีจิสเตอร์อีนาเบิล อินเตอร์รัปต์

ตารางที่ 2.1 บิตต่างๆของรีจิสเตอร์ควบคุมสายสื่อสาร

บิต	ความหมาย
0	ความยาวเวิร์ด บิตนัยสำคัญต่ำสุด (Least significant bit)
1	ความยาวเวิร์ด บิตนัยสำคัญสูงสุด (Most significant bit)
2	สโตป
3	อีนาเบิลพาริตี้
4	เลือกพาริตี้
5	พาริตี้เป็นหนึ่ง
6	เบรก
7	บิตควบคุมการเข้าถึงแลตช์ตัวหาร (DLAB)

ตารางที่ 2.2 การเลือกความยาวเวิร์ดในรีจิสเตอร์ควบคุมสายสื่อสาร

บิต 0	บิต 1	ความยาวเวิร์ด
0	0	5
0	1	6
1	0	7
1	1	8

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1.2 รีจิสเตอร์ควบคุมโมเด็ม

รีจิสเตอร์ตัวนี้ควบคุมสัญญาณ แฮนเช็คกิ้ง ที่ถูกส่งออกจาก UART ความหมายแต่ละบิตถูกแสดงไว้ในตารางที่ 2.3

1. บิต 0 ถูกใช้เพื่อเซต Data Terminal Ready (TDR) ให้เป็นศูนย์ ถ้าบิตนี้เป็นศูนย์ DRT จะถูกเซตเป็นหนึ่ง

2. บิต 1 ถูกใช้ในลักษณะเดียวกับ บิต 0 แต่ควบคุม Request to Send (RTS)

3. บิต 2 และ 3 ควบคุมขา OUT 1 และ OUT 2 บนไอซีเอ็มพีซี บิต 3 ต้องถูกเซตเพื่ออีนาเบิล I/O แบบกระตุ้นด้วยอินเตอร์รัปต์

4. บิต 4 อีนาเบิล ภาวะการตรวจสอบการทำงาน

5. บิต 5 ถึง 7 ถูกเซตเป็นศูย์เสมอ

ตารางที่ 2.3 บิตต่างๆของรีจิสเตอร์ควบคุมโมเด็ม

บิต	ตัวย่อ	ชื่อ
0	DTR	Data Terminal ready
1	RTS	Request to Send
2	Out 1	User-defined output 1
3	Out 2	User-defined output 1
4	Loop	Test mode loopback

1.3 รีจิสเตอร์อีนาเบิลอินเตอร์รัปต์

UART สามารถส่งสัญญาณอินเตอร์รัปต์เมื่อมีเหตุการณ์เกิดขึ้น โดยใช้รีจิสเตอร์อีนาเบิลอินเตอร์รัปต์เป็นตัวบอก UART ว่าเหตุการณ์ใดควรทำให้เกิดอินเตอร์รัปต์

ถ้าโปรแกรมใช้วิธีการโพลถึง (polling) ไม่มีการอีนาบิลอินเตอร์รัปต์รีจิสเตอร์สถานะจะถูกโปรแกรมใช้เพื่อตรวจสอบว่ามีอะไรเกิดขึ้นแทน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตารางที่ 2.4 บิตต่างๆของรีจิสเตอร์อีนาเบิลอินเตอร์รัปต์

บิต	อินเตอร์รัปต์
0	ได้รับข้อมูล
1	รีจิสเตอร์พักข้อมูลส่งว่าง (Transmitter Holding register)
2	ได้รับสถานะสายสื่อสาร
3	ได้รับสถานะโมเด็ม
4-7	เป็นศูนย์เสมอ

1.4 ตัวหารอัตราบอด

อัตราบอดถูกตั้งโดยใส่ค่าตัวหารสัญญาณนาฬิกา (1.8432 Mhz) ลงในรีจิสเตอร์สองตัว ความถี่ของผลลัพธ์จากการหารจะเป็น 16 เท่าของอัตราบอด รีจิสเตอร์สองตัวนี้คือ DLL (Divisor latch least significant byte) และ DLM (Divisor latch most significant byte) ค่าตัวหารและอัตราบอดต่างๆ แสดงไว้ในตารางที่ 2.5

ตารางที่ 2.5 ตัวหารอัตราบอดสำหรับอัตราบอดที่นิยมใช้

อัตราบอด	เลขฐานสิบ	เลขฐานสิบหก	LSB	MSB
300	384	180	1	80
1200	96	60	0	60
2400	48	30	0	30
4800	24	18	0	18
9600	12	0C	0	0C

- อัตราบอดไม่ได้ถูกจำกัดไว้ที่ใช้กันตามปกติ ค่าที่ต้องการสามารถสร้างโดยกำหนดตัวหารที่เหมาะสม

2. รีจิสเตอร์สถานะ

รีจิสเตอร์สถานะมีสามตัว คือ รีจิสเตอร์แสดงสถานะของสายสื่อสาร (Line Status register) รีจิสเตอร์แสดงสถานะโมเด็ม (Modem Status register) และรีจิสเตอร์จำแนกอิน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เตอร์รัปต์ (Interrupt Identification registor) รีจิสเตอร์เหล่านี้บอกให้ซีพียูทราบว่าเกิดอะไรขึ้นในส่วนต่างๆของ UART

2.1 รีจิสเตอร์แสดงสถานะสายสื่อสาร

รีจิสเตอร์นี้ให้ข้อมูลเกี่ยวกับการรับการส่งข้อมูลความหมายของแต่ละบิตถูกแสดงไว้ในตารางที่ 2.6 โดยสรุปการทำงานได้ดังนี้

- Data Ready หมายถึงได้รับตัวอักษรจากภายนอก บิตนี้จะถูกเซตจนกว่าตัวอักษรนั้นจะถูกอ่านไปจากรีจิสเตอร์พักข้อมูลรับ (Receiver buffer register)

- Overrun Error หมายถึงได้รับตัวอักษรใหม่ก่อนที่ตัวอักษรก่อนหน้าจะถูกอ่านออกไป แสดงว่าตัวอักษรเข้ามาเร็วกว่าการประมวลผล

- Parity Error หมายความว่า ตัวอักษรที่ได้รับได้มีพาริตี้หรือไม่ถูกต้อง

- Framing Error หมายความว่า ไม่พบบิตจบตามที่ควรจะเป็น

- Break Interrupt หมายถึงได้รับสัญญาณเบรก

- Transmitter Holding Register Empty หมายความว่า UART พร้อมจะรับข้อมูลเพื่อส่งออกไป

- Transmitter Shift Register Empty หมายความว่า UART ไม่อยู่ในขั้นตอนการส่งตัวอักษร รีจิสเตอร์นี้ถูกใช้ในกระบวนการแปลงข้อมูลขนานกับอนุกรม ตามปกติซอฟต์แวร์สื่อสารไม่ได้ตรวจสอบสถานะของมัน

ตารางที่ 2.6 บิตต่างๆของรีจิสเตอร์แสดงสถานะสายสื่อสาร

บิต	ตัวย่อ	ชื่อ	ความหมาย
0	DR	Data ready	ได้รับตัวอักษรเข้ามาและถูกนำไปใส่ไว้ในรีจิสเตอร์พักข้อมูลรับ
1	OE	Overrun error	ได้รับตัวอักษรก่อนที่ตัวอักษรตัวก่อนจะถูกอ่านไปประมวลผล
2	PE	Parity error	บิตพาริตี้ของตัวอักษรที่เข้ามาไม่ถูกต้อง
3	FE	Framing error	ตัวอักษรที่เข้ามามีบิตจบไม่ถูกต้อง
4	BI	Break interrupt	ได้รับสัญญาณเบรก คือสายข้อมูลเข้าเป็นศูนย์นานกว่าความยาวสูงสุดของตัวอักษร

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตารางที่ 2.6 (ต่อ) บิตต่าง ๆ ของรีจิสเตอร์แสดงสถานะสายสื่อสาร

บิต	ตัวย่อ	ชื่อ	ความหมาย
5	THRE	Transmitter Holding register empty	UART พร้อมที่จะรับอักษรตัวใหม่เพื่อส่งออกไป
6	TSRE	Transmitter Shift register empty	TSR กำลังรอตัวอักษรจาก THR
7	[SPACE]		ถูกเซตเป็นศูนย์ตลอดเวลา

2.2 รีจิสเตอร์แสดงสถานะโมเด็ม

รีจิสเตอร์ตัวนี้ให้ข้อมูลเกี่ยวกับสายสัญญาณแฮนด์เช็คกิ้ง โดยแต่ละบิตมีความหมายตามตารางที่ 2.7

ตารางที่ 2.7 บิตต่างๆของรีจิสเตอร์แสดงสถานะโมเด็ม

บิต	ชื่อ	ความหมายถ้าถูกเซต
0	Delta CTS	สายสัญญาณ Clear to send มีการเปลี่ยนแปลง
1	Delta DSR	สายสัญญาณ Deta set ready มีการเปลี่ยนแปลง
2	TERI	สัญญาณ RI เปลี่ยนจาก on เป็น off
3	Delta RLSD	Received line signal detect มีการเปลี่ยนแปลง
4	CTS	Clear to send เป็นไฮ
5	DSR	Data set ready เป็นไฮ
6	RI	Ring indicator เป็นไฮ
7	RLSD	Received line signal detect เป็นไฮ

บิต 1,2 และ 4 เรียกว่า เดลต้าบิต (Delta bits) เป็นตัวบอกว่ามีการเปลี่ยนแปลงนับจากการอ่านรีจิสเตอร์นี้ครั้งล่าสุด รีจิสเตอร์อีนาเบิล อินเตอร์รัปต์ สามารถถูกเซตเพื่อให้สร้างสัญญาณอินเตอร์รัปต์ เมื่อสถานะของโมเด็มเปลี่ยนไป แต่มีหลายเหตุการณ์สามารถทำให้เกิดการเปลี่ยนแปลงนี้ ดังนั้นเมื่อมีอินเตอร์รัปต์ประเภทนี้เกิดขึ้น โปรแกรมจะตรวจสอบเดลต้าบิตเพื่อดู

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ว่ามีอะไรเปลี่ยนแปลงไป แล้วจึงตอบสนองให้สอดคล้องกัน (เช่น หยุดการส่งชั่วคราว เมื่อสายสัญญาณแฮนด์เช็คกิ้งขาเข้าเป็นโล)

บิต 2 แสดงการเปลี่ยนแปลงเช่นกัน แต่ถูกเซตก็ต่อเมื่อสายสัญญาณ Ring indicator เปลี่ยนจาก 1 เป็น 0 เท่านั้น ส่วนบิต 1,3 และ 4 ถูกเซตเมื่อแรงดันไฟฟ้าบนสายสัญญาณนั้นเปลี่ยนแปลงในทิศทางใดๆ

อีก 4 บิตที่เหลือให้สถานะที่แท้จริงของสายสัญญาณแต่ละเส้น ค่าหนึ่งหมายความว่าสายสัญญาณเป็นไฮ คือยอมให้มีการสื่อสาร ค่าสูงหมายความว่าสายสัญญาณเป็นโล คือห้ามไม่ให้มีการติดต่อสื่อสาร

ไม่จำเป็นที่สายสัญญาณแฮนด์เช็คกิ้งขาเข้าต้องเป็นไฮจึงจะมีการส่งข้อมูลเป็นหน้าที่ของผู้เขียนโปรแกรมที่จะทดสอบสถานะของพวกมันเมื่อต้องการหรือจะไม่สนใจฮาร์ดแวร์แฮนด์เช็คกิ้งเลยก็ทำได้แต่อย่าลืมว่าฟังก์ชั้นการส่งข้อมูล ของ BIOS จะไม่ส่งข้อมูล สายสัญญาณทั้งคู่ไม่เป็นไฮเพราะมันตรวจสอบสถานะนี้

2.3 รีจิสเตอร์จำแนกอินเตอร์รัปต์

รีจิสเตอร์นี้ให้ข้อมูลเกี่ยวกับสถานะปัจจุบันของอินเตอร์รัปต์ที่ค้างอยู่บิต 0 จะถูกเซตเป็น 1 ถ้าไม่มีอินเตอร์รัปต์ค้างอยู่ ถ้ามันเป็นศูนย์ บิต 1 และ 2 จะเป็นตัวบอกว่ามีอินเตอร์รัปต์ใดค้างอยู่ตามตารางที่ 2.8 ส่วน บิต 3 ถึง 7 เป็นศูนย์เสมอ

ตารางที่ 2.8 บิตต่างๆของรีจิสเตอร์จำแนกอินเตอร์รัปต์

บิต 2	บิต 1	อินเตอร์รัปต์
1	1	ได้รับสถานะสายสื่อสาร
1	0	ได้รับข้อมูล
0	1	รีจิสเตอร์พักข้อมูลส่งว่าง
0	0	ได้รับสถานะโมเด็ม

3. รีจิสเตอร์พักข้อมูล

รีจิสเตอร์ประเภทที่สาม ใน UART คือรีจิสเตอร์พักข้อมูล ซึ่งมีอยู่สองตัวสำหรับข้อมูลที่รับเข้ามา และข้อมูลที่จะส่งออกไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.1 รีจิสเตอร์พักข้อมูลรับ

รีจิสเตอร์พักข้อมูลรับ (Receiver Buffer register) ทำหน้าที่เป็นตัวเก็บอักษรล่าสุดที่รับเข้ามาเมื่อตัวอักษรถูกอ่านออกไป รีจิสเตอร์แสดงสถานะจะแจ้งว่ารีจิสเตอร์พักข้อมูลรับว่าง จนกว่าตัวอักษรอื่นจะเข้ามาถ้าตัวอักษรตัวที่สองเข้ามาก่อนที่ตัวแรกจะถูกอ่านออกไป จะเกิดการผิดพลาดโอเวอร์รัน

3.2 รีจิสเตอร์พักข้อมูลส่ง

รีจิสเตอร์พักข้อมูลส่ง (Transmitter Holding register) ทำหน้าที่เก็บตัวอักษรที่จะถูกส่งเป็นตัวต่อไป โดยที่ตัวอักษรนี้ถูกใส่เข้ามาโดยโปรแกรมเมื่อตัวอักษรถูกส่งออกไปแล้ว รีจิสเตอร์แสดงสถานะสายสื่อสารจะแสดงให้ทราบ

4. การโปรแกรม UART แบบโพลลิ่ง

การโปรแกรม UART ให้ใช้วิธีโพลลิ่งและไม่มีอินเตอร์รัปต์ ขั้นตอนแรกคือการเซตรีจิสเตอร์อีนาเบิลอินเตอร์รัปต์ให้เป็นศูนย์ ในการส่งข้อมูลโปรแกรมต้องทำงานอยู่ในลูป (Loop) จนกว่าจะไม่มีข้อมูลที่จะส่ง รีจิสเตอร์แสดงสถานะสายสื่อสาร และแสดงสถานะโมเด็มจะถูกอ่านซ้ำๆ เพื่อตรวจสอบว่าสัญญาณแฮนด์เช็คกิ้งที่เหมาะสมเป็นไฮหรือไม่ และเมื่อรีจิสเตอร์พักข้อมูลส่งว่างโปรแกรมก็ส่งตัวอักษรตัวต่อไป

การทำซอฟแวร์แฮนเช็คกิ้งเป็นหน้าที่ของโปรแกรม เช่นถ้าใช้ XON/XOFF ก็ต้องทดสอบตัวอักษรที่รับเข้าได้ว่าเป็น XOFF หรือไม่ ถ้าใช่ก็ต้องหยุดส่งจนกว่าจะได้รับ XON จึงทำการส่งต่อไปได้

กระบวนการรับก็คล้ายกับกระบวนการส่ง ขั้นแรกต้องเซตสัญญาณแฮนเช็คกิ้งที่อุปกรณ์อีกฝ่ายต้องการ แล้วเข้าลูปเพื่อทดสอบรีจิสเตอร์แสดงสถานะสายสื่อสารว่ามีตัวอักษรถูกรับเข้ามาหรือไม่ ถ้ามีก็อ่านตัวอักษรนั้นจากรีจิสเตอร์พักข้อมูล

ในการสื่อสารแบบฟูลดูเพล็กซ์ การส่งและการรับต้องทำไปพร้อมกันโปรแกรมจึงต้องทดสอบทั้งตัวอักษรที่รับได้ และความพร้อมที่จะส่งข้อมูลมันควรทดสอบอินพุตจากแป้นพิมพ์เป็นส่วนหนึ่งของลูปด้วย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

5. การโปรแกรม UART แบบอินเตอร์รัปต์

การโปรแกรมคอมพิวเตอร์ให้ใช้ I/O แบบกระตุ้นด้วนอินเตอร์รัปต์ขั้นตอนแรกต้องติดตั้งส่วนของโค้ดที่เหมาะสมเรียกว่าตัวจัดการอินเตอร์รัปต์ (Interrupt handler) หลังจากเขียนโค้ดส่วนนี้แล้ว ใส่ตัวชี้ไปยังตัวจัดการอินเตอร์รัปต์ลงในตารางอินเตอร์รัปต์เวกเตอร์ แล้วจึงสั่งให้ UART สร้างสัญญาณอินเตอร์รัปต์ด้วยการเซตรีจิสเตอร์อีนาเบิ้ลอินเตอร์รัปต์ เมื่อมีเหตุการณ์เกิดขึ้น UART จะส่งสัญญาณไปอินเตอร์รัปต์ซีพียู ซีพียูจะค้นหาตัวชี้ไปยังตัวจัดการอินเตอร์รัปต์ จากนั้นจึงเรียกรูทีน (routine) ส่วนนั้นมาทำงาน

เมื่อตัวจัดการอินเตอร์รัปต์ได้รับอินเตอร์รัปต์ ตอนแรกมันจะรู้เพียงว่า UART ทำให้เกิดอินเตอร์รัปต์การหาสาเหตุของการเกิดอินเตอร์รัปต์จะดูจากรีจิสเตอร์จำแนกอินเตอร์รัปต์ ซึ่งอาจเป็น Data Ready ในกรณีนี้ตัวจัดการแล่ะอ่านตัวอักษรออกจากรีจิสเตอร์พักข้อมูลรับ โดยปกติตัวอักษรนั้นจะถูกใส่ไว้ในบัฟเฟอร์ชุดหนึ่ง ซึ่งจะถูกเข้าถึงโปรแกรมหลักในภายหลัง

อินเตอร์รัปต์อาจเกิดจากรีจิสเตอร์พักข้อมูลส่งว่าง ในกรณีนี้ตัวจัดการจะใส่ตัวอักษรที่จะถูกส่งตัวต่อไปลงในรีจิสเตอร์พักข้อมูลส่ง

ในการเขียนโปรแกรมแบบกระตุ้นด้วยอินเตอร์รัปต์ ตามปกติจะมีโปรแกรมทำงานในเวลาเดียวกับตัวจัดการอินเตอร์รัปต์ โปรแกรมหลักและตัวจัดการอินเตอร์รัปต์จะสื่อสารกันโดยการดึงและใส่ตัวอักษรลงในส่วนของบัฟเฟอร์ในหน่วยความจำ ซึ่งมักจะเป็นบัฟเฟอร์แบบวงกลม (Circular buffer)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของลิขสิทธิ์ทุกครั้งที่มีการนำไปใช้
036984

บทที่ 3
เทคโนโลยีของโมเด็ม

3.1 เทคโนโลยีของโมเด็ม

คำจำกัดความในการสื่อสารข้อมูลผ่านโมเด็มที่ควรจะทราบมีดังนี้

3.1.1 Baud

เป็นหน่วยการวัดที่ใช้วัดอัตราการส่งข้อมูลของโมเด็มตั้งขึ้นเพื่อเป็นเกียรติแก่ Frenchman Eneli Baudot บิดาของการสื่อสารข้อมูล (เนื่องจาก Baud นั้นเป็นหน่วยวัดอยู่แล้ว ดังนั้นการเขียนว่า Baud Rate จึงเป็นการเขียนที่ผิด) โมเด็มนั้นจะส่งข้อมูลโดยการส่งสัญญาณเสียงเป็นชุดๆ ไปตามสายโทรศัพท์ การเปลี่ยนแปลงระดับสัญญาณเสียงนั้นจะแทนสัญญาณดิจิตอล 0 และ 1 ซึ่งสามารถเทียบได้กับสัญลักษณ์จุดและขีดของรหัสมอร์ส แต่สิ่งที่แตกต่างกันก็คือ การเปลี่ยนแปลงสัญญาณเสียงนี้จะทำที่ความถี่สูงกว่ารหัสมอร์สมาก ดังนั้นหน่วย Baud ก็คือจำนวนครั้งของการติดและดับของสัญญาณเสียงในเวลาหนึ่งวินาทีนั่นเอง

โมเด็มรุ่นเก่าๆ นั้นจะส่งข้อมูลหนึ่งบิตต่อการเปลี่ยนแปลงของสัญญาณเสียงหนึ่งครั้ง หรือ 1 Bit ต่อ Baud ดังนั้นจึงทำให้คนส่วนใหญ่เข้าใจผิดกันไปว่า Baud นั้นก็คือจำนวนบิตต่อวินาที แต่ในความเป็นจริงแล้วไม่ได้เป็นเช่นนั้น โมเด็มรุ่นใหม่ๆ จะส่งข้อมูล 4 บิตหรือมากกว่านั้นต่อหนึ่ง Baud ยก ตัวอย่างเช่นมาตรฐานของโมเด็มที่นิยมใช้กันมากก็คือมาตรฐาน V.22bis ซึ่งทำงานที่ 600 Baud แต่มันจะส่งข้อมูล 4 บิต ต่อหนึ่ง Baud ดังนั้นโมเด็มที่ใช้มาตรฐานนี้จึงทำงานที่ 2400 บิตต่อวินาที

3.1.2 บิตต่อวินาที (BPS)

เป็นหน่วยวัดที่ใช้แสดงจำนวนบิตของข้อมูลที่ส่งออกไปจากโมเด็มภายในเวลาหนึ่งวินาที จากที่ได้ทราบมาก่อนหน้านี้แล้วว่าโมเด็ม 600 Baud นั้นสามารถส่งข้อมูลได้ 1200 หรือ 2400 BPS ซึ่งก็ขึ้นอยู่กับจำนวนบิตข้อมูลในหนึ่ง Baud

3.1.3 การย่อข้อมูล

การย่อข้อมูล (Data Compression) หมายถึงวิธีการจัดการกับข้อมูลให้มีขนาดเล็กลง ซึ่งมีหลักการพื้นฐานง่ายๆ ก็คือกำจัดข้อมูลที่ซ้ำกันออกไปให้มากที่สุด โดยอาจจะแทนข้อมูลดังกล่าว

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ในเชิงพาณิชย์ หรือเพื่อการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ด้วยรหัสพิเศษที่มีขนาดเล็กกว่าตัวข้อมูลจริง ซึ่งก็มีวิธีการย่อข้อมูลอยู่หลายวิธีที่นิยมใช้กันอยู่ในปัจจุบัน การย่อข้อมูลนี้สามารถทำได้จากโปรแกรมย่อข้อมูลในเครื่อง PC และจากวงจรย่อข้อมูลที่อยู่ในตัวโมเด็มเอง

3.1.4 การผสมสัญญาณ

การผสมสัญญาณ (Modulation) หมายถึงวิธีการที่จะใช้ในการส่งสัญญาณเสียงออกไปตามสายโทรศัพท์ โมเด็มในรุ่นแรกๆ นั้นจะใช้สัญญาณเสียงอยู่สองรูปแบบ ก็คือรูปแบบของสัญญาณ "ติด" และ "ดับ" เท่านั้น ซึ่งโมเด็มต้นทางก็จะใช้ความถี่ในย่านหนึ่ง และปลายทางก็จะใช้ความถี่อีกย่านหนึ่งรับส่งข้อมูลสวนกัน นั่นก็หมายความว่าจะใช้สัญญาณเสียง 2 คู่ คือ "ติด" และ "ดับ"(หมายถึงข้อมูลขนาด 1 บิต) ของโมเด็มต้นทางและ "ติด" และ "ดับ" ของโมเด็มปลายทาง แต่ในปัจจุบัน มาตรฐานใหม่ๆ ก็ได้กำหนดให้สัญญาณเสียงไม่ได้มีแต่เพียง "ติด" และ "ดับ" เท่านั้น แต่อาจจะมีรูปแบบที่ซับซ้อนขึ้น โดยจะใช้เทคนิคทางด้านเฟส และขนาด (Amplitude) เข้ามาแทนชุดข้อมูลที่มีขนาดมากกว่า 1 บิต อย่างเช่น 001, 010, 101 เป็นต้น จึงทำให้สัญญาณเสียงสามารถมีรูปแบบต่างๆ ได้มากถึง 512 แบบ ซึ่งถ้ารวมสัญญาณเสียงของโมเด็มทั้งด้านต้นทางและปลายทางแล้ว ก็จะมีจำนวนรูปแบบของสัญญาณ เสียงทั้งหมดในสายโทรศัพท์ถึง 512 คู่ทีเดียว

3.1.5 ทรูพุต

ทรูพุต (Through put) หมายถึงจำนวนข้อมูลที่สามารถส่งผ่านโมเด็มออกไปภายในหนึ่งวินาที ซึ่งจะอยู่ในหน่วยบิตต่อวินาที เมื่อใช้บอกคุณสมบัติของโมเด็ม และมีหน่วยเป็นจำนวนตัวอักษรต่อวินาทีเมื่อใช้ส่งแฟ้มข้อมูล ดังนั้น โมเด็มความเร็ว 9600 BPS เสมอไป ค่าทรูพุตนี้จะขึ้นอยู่กับคุณภาพของสายโทรศัพท์ในขณะนั้นๆ ว่าเป็นอย่างไร ถ้าหากการส่งมีปัญหาสัญญาณรบกวนมากค่าทรูพุตก็จะต่ำกว่า 9600 BPS และถ้าหากนำเอาเทคนิคการย่อข้อมูลแบบ V.42bis มาใช้งานด้วยแล้ว โมเด็มตัวนี้สามารถที่จะเพิ่มค่าทรูพุตขึ้นไปได้ถึง 19.2 kBPS หรือมากกว่านั้นได้

3.2 ประวัติโดยย่อของโมเด็ม

คนส่วนใหญ่นั้นคิดว่าการสื่อสารข้อมูลเพิ่งจะเริ่มมีขึ้นหลังจากการประดิษฐ์เครื่องคอมพิวเตอร์ขึ้นมา แต่ในความเป็นจริงแล้วเทคโนโลยีการสื่อสารข้อมูลนี้มีมานานกว่านั้นมาก โดยเริ่มตั้งแต่ปลายศตวรรษที่ 18 ในช่วงที่เครื่องโทรเลขจะต้องมีความรู้ความชำนาญสูงในการแปลสัญญาณต่างๆ ที่รับได้ออกมาเป็นสัญลักษณ์จุดและขีดเอาเองด้วยมือ แต่เครื่องโทรเลขของ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เอดิสันนั้นสามารถพิมพ์ข้อมูลที่ได้รับออกมาเป็นจุดและขีดได้ทันที ดังนั้นจึงช่วยลดความเบื่อหน่ายและความผิดพลาดของพนักงานโทรเลขไปได้มากทีเดียวพร้อมๆ กันนี้ Samuel Morse และ Emili Baudot ก็ได้พัฒนาเครื่องมือสื่อสารรุ่นใหม่อื่นๆ ออกมามากมาย ซึ่งต่อมาก็ได้พัฒนาเป็นเครื่องโทรพิมพ์ (TTY)ในที่สุด

เครื่องโทรพิมพ์เครื่องแรกนั้นถูกพัฒนาขึ้นมาโดย Edward Kleinschmidt ในช่วงต้นทศวรรษที่ 1900 ซึ่งเครื่องรุ่นนี้มีกลไกของเครื่องพิมพ์และแป้นพิมพ์อยู่ในเครื่องเดียวกัน เมื่อพนักงานกดปุ่มบนแป้นพิมพ์ตัวอักษรที่กดนั้นก็จะถูกพิมพ์ลงบนกระดาษที่เครื่องปลายทาง การส่งข้อมูลโดยเครื่องโทรพิมพ์นี้จะทำผ่านสายโทรศัพท์พิเศษที่ได้เช่าเหมาเอาไว้แล้ว (leased line) อุปกรณ์ชิ้นนี้ไม่ได้มีอุปกรณ์สื่อสารพิเศษใดๆ อย่างโมเด็มเลย มันส่งข้อมูลแต่ละตัวอักษรไปตามสายโทรศัพท์แบบเช่าเหมาโดยตรง แต่จากข้อมูลทางเทคนิคของสายโทรศัพท์นั้นถูกออกแบบมาเพื่อใช้งานกับเสียงของมนุษย์ ดังนั้นการใช้งานเครื่องโทรพิมพ์ผ่านทางสายโทรศัพท์นั้นจะต้องเป็นการต่อเชื่อมระหว่างจุดพิเศษเข้าด้วยกันเท่านั้น เนื่องจากว่าการทำงานของเครื่องมือชนิดนี้มีกลไกเป็นส่วนประกอบอยู่ค่อนข้างมากจึงทำให้ความเร็วในการส่งข้อมูลนั้นอยู่ที่ห้าสิบตัวอักษรต่อวินาทีซึ่งนับว่าช้ามาก

เครื่องโทรพิมพ์นี้ส่วนใหญ่มักจะใช้งานกันมากในธุรกิจทางด้านข่าวสารโดยสำนักข่าวรอยเตอร์ได้ติดตั้งเครื่องโทรพิมพ์ไว้ตามสาขาต่างๆ ทั่วโลกซึ่งส่วนใหญ่ก็ได้ถูกแทนที่ด้วยอุปกรณ์สื่อสารชนิดอื่นๆ ที่มีความเร็วในการส่งข้อมูลสูงกว่าไปแล้วในปัจจุบันแต่ก็มีบางส่วนที่ยังคงเป็นเครื่องโทรพิมพ์แบบเดิม ระบบเครือข่ายเครื่องโทรพิมพ์นี้เรารู้จักกันดีในชื่อเทเลกซ์ ซึ่งใช้งานได้เฉพาะตัวอักษรตัวพิมพ์ใหญ่เท่านั้น เครื่องเทเลกซ์นี้จะมีประโยชน์มากสำหรับบริษัทที่มีการติดต่อกับต่างประเทศบ่อยๆ ถึงแม้ว่าความเร็วในการทำงานของเทเลกซ์จะต่ำแต่มันก็เป็นที่นิยมกันมากเพราะว่ามันสะดวกในการติดต่อไปตามสถานที่ต่างๆ ทั่วโลกได้ตลอดยี่สิบสี่ชั่วโมงนั่นเอง

ในปี 1958 ทาง AT&T ได้สร้างบริการรูปแบบใหม่ขึ้นมาซึ่งก็คือบริการที่ชื่อว่า Dataphone ซึ่งทำให้สามารถส่งข้อมูลคอมพิวเตอร์ผ่านตามสายโทรศัพท์ทั่วๆ ไป ได้หัวใจของการให้บริการนี้ก็คืออุปกรณ์ที่ใช้ส่งข้อมูลที่มีชื่อว่า Dataphone Set ซึ่งเรารู้จักกันในชื่อของโมเด็มเครื่องอื่นๆ ได้เหมือนการใช้งานโทรศัพท์ปกติ ซึ่งการติดตั้งอุปกรณ์ดังกล่าวนี้ก็ไม่ยุ่งยากเพียงแต่ต่อเข้ากับสายโทรศัพท์ทั่วๆ ไปเท่านั้น เนื่องจากเครื่องโฮสคอมพิวเตอร์ในขณะนั้นส่วนใหญ่จะใช้เครื่องพิมพ์แบบกลไก ดังนั้นความเร็วขนาด 300 BPS ของเครื่อง DataPhone set ในยุคนั้นก็ยังเพียงพอต่อการใช้งาน

โมเด็มที่แท้จริงตัวแรกของโลกก็คือโมเด็มรุ่น Bell Dataphone 103 มีความเร็วในการรับส่งข้อมูล 300 BPS ซึ่งเป็นโมเด็มมาตรฐานที่โมเด็มทุกเครื่องที่มีอยู่ในปัจจุบันสามารถทำงานใน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ฟังก์ชันการทำงานของโมเด็มรุ่นนี้ได้ทั้งหมด และโมเด็มขนาดเล็กตัวแรกของโลกก็คือโมเด็มที่ออกแบบโดย Dale Heatherington จาก บริษัท Hayes Microcomputer Product ในราวๆ ปีทศวรรษที่ 70

การพัฒนาโมเด็มรุ่นใหม่ๆ นั้นมักจะออกมาจากสองบริษัทหลักก็คือบริษัท Racal-Vadic และ AT&T ซึ่งทั้งสองบริษัทนี้ก็แข่งขันกันพัฒนาโมเด็มรุ่นใหม่ๆ ออกมาเรื่อยๆ โดยวิศวกรของ AT&T นั้นได้ผลิตโมเด็มรุ่น Bell 212a ทำงานที่ความเร็ว 1200 BPS ขึ้นมา แต่โมเด็มรุ่นดังกล่าวนั้นมีปัญหาเกี่ยวกับการสะท้อนของสัญญาณกับเครื่องโทรศัพท์ธรรมดา ที่ติดตั้งด้วยกัน ดังนั้นการใช้งานโมเด็มรุ่นนี้จึงต้องต่อเข้ากับสายโทรศัพท์โดยตรงไม่สามารถใช้ร่วมกับเครื่องโทรศัพท์ได้

ในช่วงต้นๆ ของการพัฒนา โมเด็มยี่ห้อต่างๆ ที่ไม่ได้ผลิตโดย AT&T นั้นจะต้องเสียค่าลิขสิทธิ์วงจรให้แก่ทาง AT&T ทำให้ราคาของโมเด็มรุ่นที่เทียบเท่ากับโมเด็มรุ่น 212a นี้ราคาสูงกว่ารุ่น 103 มาก แต่เมื่อบริษัท Rockwell และ บริษัทผู้ผลิตรายอื่นๆ สามารถพัฒนาวงจรที่ใช้ทดแทนวงจรของ AT&T ได้ดังนั้นราคาของโมเด็มรุ่น 212a นี้เป็นที่นิยมก็คือโมเด็มมีความเร็วในการทำงานสูงกว่ารุ่น 103 ถึงสี่เท่านั่นเอง

ส่วนโมเด็มของผู้ผลิตรายอื่นๆ เช่นรุ่น 3400 ของ Vardic นั้นมีความเร็วในการส่งข้อมูล 1200 bps เช่นเดียวกัน แต่วงจรที่ใช้นั้นไม่สามารถใช้กับโมเด็มของ Bell ได้ดังนั้นจึงไม่เป็นที่นิยมของผู้ใช้ และจากที่ทราบมาแล้วจากบทที่สองว่า บริษัท Bell นั้นได้ผูกขาดการสื่อสารในสหรัฐมาช้านานดังนั้นอุปกรณ์ต่างๆ ที่ผลิตโดย Bell จึงกลายเป็นมาตรฐานของทวีปอเมริกาเหนือไปโดยปริยาย ซึ่งก็รวมถึงมาตรฐานของโมเด็มรุ่น Bell 103 และ 212a ด้วย ดังนั้นผู้ซื้อจึงไม่มีทางเลือกอื่นๆ ในการเลือกซื้ออุปการณ์โทรศัพท์เลย แต่หลังการแยกออกมาเป็นบริษัทย่อยๆ ในปี ค.ศ. 1984 AT&T ก็ได้สูญเสียการผูกขาดไปในทันที การแยกตัวของบริษัท AT&T ในครั้งนั้นทำให้มีผู้ประกอบการรายใหม่ๆ กระโดดเข้ามาทำธุรกิจอุปกรณ์ การสื่อสารมากขึ้น ซึ่งก็ทำให้ผู้ซื้อมีตัวเลือกสำหรับการซื้อมากขึ้นกว่าเดิม แต่ก็น่าเสียดายที่ AT&T ต้องแตกออกไปเป็นบริษัทย่อยๆ เสียก่อนจึงทำให้การนำระบบ ISDN เข้ามาใช้นั้นล่าช้าออกไปไม่อย่างนั้นเราคงได้ใช้ระบบ ISDN กันอย่างแพร่หลาย แล้วในปัจจุบัน

เนื่องจากว่า AT&T นั้นก็ยังไม่ได้กำหนดมาตรฐานของโมเด็มความเร็ว 2400 BPS เอาไว้ ดังนั้นโมเด็มรุ่นใหม่ๆ ที่ผลิตขึ้นในยุโรปจึงใช้มาตรฐานของยุโรปซึ่งก็คือ CCITT V.22 bis และโมเด็มรุ่นที่ใช้มาตรฐาน CCITT นี้ก็สนับสนุนการทำงานตามมาตรฐาน Bell 212a ไว้ด้วย ดังนั้นจึงเป็นที่นิยมใช้กันอย่างแพร่หลาย ทั้งในสหรัฐอเมริกาและยุโรป นอกจากนั้นมันยังมีราคาถูกและมีความเชื่อถือได้สูง จึงได้กลายเป็นมาตราฐานของโมเด็มความเร็ว 2400 BPS ไปในที่สุด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ถึงแม้ว่าโมเด็มมาตรฐาน CCITT V.22 bis นี้จะมีความเร็วในการสื่อสารสูงกว่าโมเด็มรุ่นก่อนๆ ถึงสองเท่า แต่ความเร็วในการส่งข้อมูลแค่ 2400 BPS นั้นยังไม่เพียงพอต่อการใช้งานอย่างแท้จริง ซึ่งมักจะต้องใช้ส่งข้อมูลจำนวนมากระหว่างศูนย์ข้อมูลต่างๆ ดังนั้นจึงต้องมีการพัฒนาโมเด็มรุ่นใหม่ๆ ที่มีความเร็วในการส่งข้อมูลสูงกว่านั้นขึ้นมาอีก โดยทาง CCITT ก็ได้ออกมาตรฐานโมเด็มความเร็ว 9600 BPS ประมาณปี 1984 แต่ผู้ผลิตโมเด็มรายอื่นๆ ไม่ได้ยอมรับมาตรฐานนี้ เนื่องจากเหตุผลทางการตลาดที่ว่า ผู้ผลิตรายใดที่สามารถยึดครองตลาดโมเด็มความเร็วสูงของตนออกมาโดยที่ไม่มีมาตรฐานใดๆ รองรับ ซึ่งโมเด็มความเร็วสูงนั้นจะต้องมีเทคนิคการผสมสัญญาณพิเศษหลายๆ อย่าง เพื่อที่จะให้ส่งข้อมูลออกไปได้คราวมากๆ แต่มีปัญหาหลักทางเทคนิคของสายโทรศัพท์ที่สามารถรับการเปลี่ยนแปลงสัญญาณเสียงได้แค่ 1600 Baud เท่านั้น ดังนั้นผู้ผลิตแต่ละรายจะต้องพัฒนาเทคนิคการย่อข้อมูลการผสมสัญญาณ และวงจรส่วนควบคุมความถูกต้องของข้อมูลขึ้นมาด้วย ทั้งนี้ก็เพราะโมเด็มความเร็วสูงนั้นจะมีความไวต่อสัญญาณรบกวนต่างๆ มากกว่าโมเด็มความเร็วต่ำ วงจรต่างๆ ที่เพิ่มขึ้นมานี้ทำให้โมเด็มรุ่นใหม่ๆ ยิ่งทวีความซับซ้อนมากขึ้นไปเรื่อยๆ จนถึงขั้นที่ต้องใช้ชิปไมโครโปรเซสเซอร์ เข้ามาควบคุมการทำงานของโมเด็ม

โมเด็มความเร็วสูงในปัจจุบันนั้นไม่ว่าจะเป็นยี่ห้อใดก็ตามก็จะต้องมีชิปไมโครโปรเซสเซอร์อยู่ในส่วนวงจรอย่างน้อยหนึ่งตัวเสมอ โมเด็มความเร็ว 9600 BPS ตัวแรกที่ออกมาสู่ท้องตลาดก็คือโมเด็มของ Telebit รุ่น Trail Blazer ใช้ไมโครโปรเซสเซอร์เบอร์ 68000 ของโมโตโรล่าทำงานกับ ROM 512K และ RAM 1MB จะเห็นได้ว่าโมเด็มรุ่นนี้มีคุณสมบัติทางเทคนิคบางอย่างดีกว่าเครื่องคอมพิวเตอร์ในยุคนั้นเสียอีก

ปัญหาหลักของโมเด็มจาก Telebit รุ่นนี้ ก็คือ ปัญหาทางด้านราคาซึ่งสูงมาก และโมเด็มรุ่นนี้ก็ใช้ได้กับโมเด็มของ Telebit เองเท่านั้น ซึ่งตลาดของโมเด็มรุ่นนี้ก็คือกลุ่มผู้ใช้ ระบบ UNIX เป็นส่วนใหญ่ หลังจากนั้นไม่นานบริษัท Hayes บริษัท U.S. Robotics และบริษัทอื่นๆ ก็ได้แนะนำโมเด็มความเร็ว 9600 BPS ของตนออกมาขายเช่นกันซึ่งแต่ละยี่ห้อจะสามารถใช้ได้กับโมเด็มในกลุ่มยี่ห้อของตนเองเท่านั้น

ถึงแม้ว่าจะมีโมเด็มความเร็วสูงออกมาให้เลือกซื้อมากมายหลายยี่ห้อแต่ผู้ซื้อเองก็ไม่กล้าซื้ออุปกรณ์ที่ยังไม่มีมาตรฐานรองรับ ดังนั้นผู้ผลิตโมเด็มต่างๆ จึงต้องกลับมาประชุมเพื่อหามาตรฐานสำหรับโมเด็มความเร็วสูงขึ้นมาอีกครั้งหนึ่ง ซึ่งก็คือมาตรฐาน CCITT V.32 และมาตรฐานที่ใหม่กว่าก็คือ V.32bis ซึ่งมีความเร็วในการส่งข้อมูลสูงขึ้นไปถึง 14400 BPS นั่นเอง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.3 มาตรฐานของโมเด็ม ทำไมจึงมีหลายมาตรฐาน

จากที่ทราบในช่วงต้นๆ แล้วว่า อัตราการเพิ่มของอุตสาหกรรมการสื่อสารนั้นเป็นไปอย่างรวดเร็วมาก ดังนั้นจึงมีมาตรฐานต่างๆ ออกมามากมาย บางมาตรฐานนั้นจะเน้นเทคนิคทางด้านการผสมสัญญาณ บางมาตรฐานก็จะเน้นเรื่องความถูกต้องของข้อมูลบางมาตรฐานนั้นก็จะเน้นที่วิธีการในการย่อข้อมูลโดยมากแล้วโมเด็มที่มีวางขายในปัจจุบันนั้นได้รวมเอามาตรฐานต่างๆ เหล่านี้เข้าไปในเครื่องๆ เดียวแล้ว ยกตัวอย่างเช่น โมเด็มรุ่น HST Dual Standard ของบริษัท U.S. Rkobotics รุ่น Courrier นั้นมีคุณสมบัติได้มาตรฐานเหล่านี้

- Bell 103 Modulation
- Bell 212a Modulation
- V.22bis Modulation
- V.23 Modulation
- V.32 และ V.32 bis Modulation
- USR-HST Modulation
- V.42 Error Control
- V.42 bis Data Compression
- MNP Class 2,3,4 Error Control
- MNP Class 5 Data Compression
- V.25 Answering Sequence
- V.54 Remote Testing Standard

รวมทั้งหมด 15 มาตรฐานในโมเด็มเครื่องเดียวกันแต่ในปัจจุบันนี้ก็มีมาตรฐานอีกชุดหนึ่งของ CCITT ที่ยังไม่ได้ตั้งชื่อเป็นทางการ แต่เป็นที่รู้จักกันในชื่อของ V.FAST ทำงานที่ความเร็ว 28.8 kBPS ซึ่งก็ขึ้นไปถึงขอบเขตการใช้งานสื่อสารข้อมูลผ่านสายโทรศัพท์แล้ว ดังนั้น V.FAST น่าจะเป็นมาตรฐานสุดท้ายของโมเด็มที่ทำงานแบบอนาล็อก ส่วนการสื่อสารที่เร็วกว่านั้นก็น่าจะเป็นการสื่อสารผ่านระบบ ISDN ซึ่งเป็นระบบดิจิตอลทั้งระบบ

ISDN นั้นตัวมันเองก็คือชุดข้อกำหนดมาตรฐานต่างๆ ของ CCITT นั่นเอง การปรากฏตัวของระบบ ISDN นั้นแทบจะทำให้โมเด็มนั้นกลายเป็นสิ่งล้าสมัยไปเลยทีเดียว ทั้งนี้ก็เพราะตัวระบบ ISDN ทำงานในระบบดิจิตอลอยู่แล้วดังนั้นจึงไม่มีความจำเป็นใดๆ ที่จะต้องเปลี่ยนข้อมูลดิจิตอลให้เป็นสัญญาณเสียงก่อนเหมือนกับที่ใช้อยู่ในโมเด็มปัจจุบัน แต่โครงข่าย ISDN ในปัจจุบันนั้นยังไม่ครอบคลุมพื้นที่ทั้งหมด ดังนั้นพื้นที่บางจุดจึงยังไม่สามารถใช้งานระบบนี้ได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

มาตรฐาน ISDN นั้นจะมีช่องนำสัญญาณอยู่สามช่องด้วยกัน คือช่องสัญญาณที่ใช้ความเร็ว 56 kBPS จำนวนสองช่องใช้สำหรับการสื่อสารข้อมูลทั้งทางดิจิตอลและเสียง ส่วนอีกหนึ่งช่องก็คือช่องสัญญาณที่มีความเร็ว 16 kBPS สำหรับสัญญาณควบคุม ซึ่งช่องสัญญาณนี้สามารถใช้ส่งรับข้อมูลดิจิตอลได้เช่นเดียวกันแต่ต้องเป็นสัญญาณความเร็วที่ไม่สูงมากนัก

ประวัติโดยย่อของโมเด็มจะไม่สมบูรณ์ ถ้าหากไม่มีการพูดถึงบริษัท Microcom ผู้ที่สร้างโปรโตคอลในการสื่อสารที่ชื่อว่า MNP (Microcom Networking Protocal)

MNP จะกล่าวถึงมาตรฐานในการควบคุมการผิดพลาดและวิธีการย่อข้อมูลในโมเด็ม โดยการจัดแบ่งมาตรฐานของ MNP จะใช้คำว่า Class ซึ่ง CCITT ก็ได้นำข้อกำหนดบางส่วนของ MNP ไปเป็นมาตรฐานอันหนึ่งของ CCITT ด้วย ดังนั้นจึงมีโมเด็มหลายยี่ห้อที่นำเอา MNP ไปใช้เพื่อเพิ่มความสามารถในการส่งข้อมูล โดยเฉพาะ MNP Class 5 นั้นเป็นวิธีการย่อข้อมูลที่มีประสิทธิภาพสูง ซึ่งผลก็คือ ถ้าหากโมเด็ม 2400 BPS เครื่องใดที่มี MNP5 อยู่ด้วยแล้วจะสามารถให้ทรูพุตสูงถึง 4800 BPS

3.4 มาตรฐานการสื่อสารของโมเด็ม

เนื่องด้วยสาเหตุจากการแย่งชิงตลาดโมเด็มความเร็วสูงที่ได้กล่าวมาแล้วข้างต้นจึงเป็นผลให้เกิดมาตรฐานใหม่ของโมเด็มออกมามากมายซึ่งก็ทำให้ผู้ใช้เกิดความสับสนในการเลือกซื้อโมเด็ม บางมาตรฐานก็มีใช้กันอยู่เฉพาะในทวีปอเมริกาเหนือ และบางมาตรฐานก็ใช้อยู่เฉพาะในยุโรปเท่านั้น ซึ่งมาตรฐานต่างๆ เหล่านี้ สามารถแบ่งออกได้เป็นสามกลุ่มใหญ่ๆ ตามวิธีการทำงานได้ดังนี้

1. กลุ่มมาตราฐานที่เกี่ยวกับการผสมสัญญาณ
2. กลุ่มมาตรฐานที่เกี่ยวกับการย่อขนาดข้อมูล
3. กลุ่มมาตรฐานที่เกี่ยวกับการควบคุมความผิดพลาดของข้อมูล โดยจะแยกพิจารณากันตามกลุ่มๆ ได้ดังนี้

3.4.1 มาตราฐานที่เกี่ยวกับการผสมสัญญาณ

เมื่อเทียบกับกลุ่มมาตรฐานอื่นๆ แล้ว การผสมสัญญาณนับได้ว่าเป็นมาตรฐานที่สำคัญที่สุดสำหรับโมเด็มทุกเครื่อง ก่อนอื่นเราจะต้องทราบก่อนว่าในการสื่อสารระหว่างโมเด็มแต่ละครั้งนั้นโมเด็มทั้งสองเครื่องจะต้องใช้เทคนิคการรับส่งข้อมูลแบบเดียวกัน เนื่องจากว่ามาตรฐานการรับส่งข้อมูลนั้นมีการพัฒนากันอยู่ตลอดเวลา ดังนั้นเพื่อให้มาตรฐานการรับส่งข้อมูลแบบใหม่นี้เป็นที่นิยมผู้พัฒนาจึงมักจะออกแบบให้มาตรฐานใหม่นี้สามารถใช้งานกับมาตรฐานเดิมได้ ยกตัวอย่างเช่นเมื่อใช้ให้โมเด็มที่ใช้มาตรฐาน Bell 212a ต่อไปยังโมเด็มที่ใช้มาตรฐาน V.22bis

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จะพยายามติดต่อกับโมเด็มตัวเรียกด้วยมาตรฐาน V.22 bis ซึ่งใหม่กว่าแต่ก็ไม่สำเร็จเพราะว่าโมเด็มตัวเรียกนั้นเป็นโมเด็มที่ใช้มาตรฐานเก่ากว่า ดังนั้นโมเด็มปลายทางที่เป็น V.22 bis จึงเปลี่ยนวิธีการติดต่อใหม่โดยกลับไปใช้มาตรฐาน Bell 212a

โมเด็มส่วนใหญ่นั้นจะใช้การสื่อสารแบบ Full Duplex ซึ่งหมายความว่า PC จะสามารถส่งและรับข้อมูลได้ในขณะเดียวกัน มีโมเด็มเพียงไม่กี่ตัวเท่านั้นที่ใช้การสื่อสารแบบ Half Duplex และนอกจากนี้โมเด็มส่วนใหญ่จะถูกออกแบบมาให้สื่อสารได้ในแบบ Asynchronous แต่โมเด็มบางรุ่นจะสามารถสื่อสารได้ทั้งแบบ Asynchronous และ Synchronous ในเครื่องเดียวกัน มาตรฐานการผสมสัญญาณที่นิยมใช้มีดังนี้

Bell 103	ข้อกำหนดของโมเด็มที่ใช้ความเร็ว 300 BPS , Full Duplex , Asynchronous
Bell 20	ข้อกำหนดของโมเด็มที่ใช้ความเร็ว 2400 BPS , Full Duplex , Synchronous ส่วนมากจะใช้กับเครื่องเมนเฟรมของ IBM
Bell 212	ข้อกำหนดของโมเด็มที่ใช้ความเร็ว 1200 BPS , Full Duplex , Asynchronous
V.22 bis	ข้อกำหนดของโมเด็มที่ใช้ความเร็ว 2400 BPS , Full Duplex , Synchronous/Asynchronous นิยมแพร่หลายกันมากที่สุด
V.29	ข้อกำหนดของโมเด็มที่ใช้ความเร็ว 4800/7200/9600 BPS , Full Duplex , Synchronous ซึ่งแฟกซ์/โมเด็ม GROUP III นั้นก็ใช้มาตรฐาน V .29 เช่นเดียวกัน
V.32	ข้อกำหนดของโมเด็มที่ใช้ความเร็ว 4800/9600 BPS , Full Duplex , Synchronous/Asynchronous
V.32 bis	ข้อกำหนดเพิ่มเติมของ V.32 โดยเพิ่มความเร็ว 7200 ,12000 และ 14400 BPS เข้าไป
Hayes Express	ชื่อเดิมก็คือ Hayes V-series 9600 ดัดแปลงมาจากมาตรฐาน V.32 ใช้การทำงานแบบ Half Duplex ในโมเด็มรุ่นเก่าของ Hayes smartmodem 9600 BPS ซึ่งโมเด็มรุ่นใหม่ของ Hayes ได้กลับมาใช้มาตรฐาน CCITT V.32 กันหมดแล้ว มาตรฐานนี้ไม่สามารถเชื่อมต่อกับโมเด็มของบรอษัทอื่น ก็ได้
USR HST	เป็นมาตรฐานเฉพาะของ U.S. Robotics มีความเร็วในการรับส่งข้อมูล 9600-14400 BPS แต่สามารถใช้ได้กับเครื่องยี่ห้อเดียวกันเท่านั้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.4.2 มาตรฐานที่เกี่ยวกับการย่อข้อมูล

โมเด็มความเร็วสูงในปัจจุบัน ส่วนใหญ่นั้นจะใช้เทคนิคการบีบขนาดข้อมูลให้เล็กลงเพื่อที่จะเพิ่มความเร็วในการส่งข้อมูลนี้จะทำให้โมเด็มนั้นมีค่าทรูพุตที่สูงขึ้น วิธีการย่อข้อมูลโดยทั่วไปแล้วจะใช้หลักการมองหาข้อมูลที่ซ้ำ ๆ กันและแทนที่ข้อมูลดังกล่าวด้วยรหัสพิเศษที่มีขนาดข้อมูลเล็กกว่าและหลังจากโมเด็มปลายทางได้รับรหัสพิเศษดังกล่าวแล้ววงจรบีบขนาดข้อมูลก็จะทำหน้าที่ถอดรหัสข้อมูลกลับมาอีกครั้งหนึ่งมาตรฐานการย่อข้อมูลที่เป็นที่นิยมกันก็มีอยู่ สองแบบด้วยกันคือ MNP และ CCITT

โดยข้อกำหนด CCITT V.42 bis นั้นออกแบบมาให้ใช้ร่วมกับมาตรฐานการควบคุมความผิดพลาดของข้อมูล (V.42) ซึ่งจะย่อขนาดข้อมูลประเภทข้อความได้ดีมาก และจะไม่ส่งผลกระทบใด ๆ ต่อแฟ้มข้อมูลที่ถูกย่อไว้แล้วส่วนมาตรฐานการย่อขนาดข้อมูลของ MNP นั้นมีอยู่สองมาตรฐานก็คือ MNP5 และ MNP7

วิธีการย่อข้อมูลของมาตรฐาน MNP5 นั้นใช้วิธีการเข้ารหัสของ Huffman และวิธี Run Lenght Method ตามความเหมาะสมของชนิดข้อมูลซึ่งจะได้อัตราส่วนการย่อข้อมูลประมาณ 2:1

การเข้ารหัสของ Huffman นั้นใช้หลักการง่าย ๆ คือตัวอักษรแต่ละตัวนั้นจะมีความถี่ในการใช้งานแตกต่างกัน ดังนั้นถ้าหากสามารถแทนตัวอักษรขนาดแปดบิตที่ใช้บ่อย ๆ ด้วยรหัสที่มีจำนวนบิตน้อยกว่าได้ก็จะลดขนาดของข้อมูลไปได้มากทีเดียวและแทนตัวหนังสืออื่น ๆ ที่ใช้น้อยครั้งก็แทนด้วยรหัสที่มีความยาวมากกว่าปกติเช่น 11 บิตเป็นต้น

วิธีการเข้ารหัสอีกแบบหนึ่งที่ใช้ใน MNP 5 ก็คือ Run lenght method โดยจะมองหาการซ้ำของตัวอักษรต่าง ๆ ในแฟ้มข้อมูล เมื่อMNP 5 ตรวจ พบว่ามีอักษรซ้ำ ๆ เรียวติดต่อกันอย่างน้อยสามตัว มันจะเรียกเอาวิธี RLL เข้ามาจัดการย่อข้อมูลทันที โดยส่งตัวอักษรที่ซ้ำไปก่อนแล้วตามด้วยเลขบอกจำนวนครั้งที่ซ้ำกัน ดังนั้นเทคนิค RLL นี้จึงเหมาะสำหรับงานบางประเภทเท่านั้นเช่นข้อมูลสเปรดชีต

โมเด็มบางรุ่นนั้นสามารถเลือกวิธีการทำงานได้โดยอิสระซึ่งบางครั้งการใช้ MNP ก็ไม่ได้ช่วยอะไรมากนักถ้าหากใช้งานกับข้อมูลที่ผ่านการย่อข้อมูลโดยโปรแกรมพวก PKZIP หรือ LHARC มาก่อนแล้วซึ่งบางครั้งอาจจะทำให้ค่าทรูพุตลดลงเสียด้วยซ้ำ

ส่วน MNP 7 นั้นเป็นวิธีการย่อข้อมูลแบบใหม่ที่ Microcom พัฒนาขึ้นมาใช้โดยจะมีอัตราส่วนการย่อข้อมูลประมาณ 3:1 นั่นก็หมายความว่าโมเด็มความเร็ว 2400 BPS ที่มี MNP 7 สามารถส่งข้อมูลได้เท่า ๆกับโมเด็มความเร็ว 7200 BPS นั้นเอง

แต่เมื่อเปรียบเทียบมาตรฐานการย่อข้อมูลต่าง ๆ แล้วจะพบว่าการย่อข้อมูลโดยมาตรฐาน V.42 bis นั้นมีเทคนิคการย่อข้อมูลที่ดีกว่า MNP 7

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.4.3 มาตราฐานที่เกี่ยวกับการควบคุมความผิดพลาดของข้อมูล

ข้อมูลที่ถูกส่งมาด้วยเร็วสูง ๆ นั้นมักถูกรบกวนจากสัญญาณรอบด้านได้ง่ายอยู่แล้ว ซึ่งสัญญาณรบกวนเพียงชั่วขณะสั้น ๆ นั้นจะทำให้ข้อมูลที่ส่งมาผิดพลาดไปอย่างมาก ดังนั้นเพื่อที่จะคงความถูกต้องของข้อมูลอยู่ด้วย ซึ่งในการสื่อสารครั้งหนึ่ง ๆ โมเด็มทั้งสองเครื่องจะต้องใช้วิธีตรวจสอบความถูกต้องของข้อมูลแบบเดียวกัน

เทคนิคต่าง ๆ ในกลุ่มมาตรฐานนี้จะเน้นหนักไปทางด้านการตรวจสอบและแก้ไปข้อมูลที่รับให้ถูกต้องได้ มันเพียงแต่ส่งสัญญาณกลับไปที่โมเด็มต้นทางให้ส่งข้อมูลชุดนั้นมาใหม่อีกครั้งหนึ่งเท่านั้น ซึ่งเทคนิคนี้มักจะถูกนำไปใช้คู่กันไปกับเทคนิคการผสมสัญญาณ ยกตัวอย่างเช่น มาตรฐานที่ใช้โมเด็มรุ่น V-Series ของ Hayes และรุ่น HST ของ U.S. robotics

มาตรฐานของการควบคุมความผิดพลาดข้อมูลนี้นิยมใช้กันก็มีแค่สองชนิด ชนิดแรกก็คือกลุ่มมาตรฐาน MNP ซึ่ง Class 1 ถึง 4 และ Class 10 นั้นเกี่ยวกับการควบคุมความผิดพลาดของข้อมูล ส่วน Class 5 ถึง 9 นั้นจะเป็นเรื่องของการย่อข้อมูล Class ที่นิยมใช้ในการตรวจสอบความผิดพลาดก็คือ Class 4 ใช้ในการสื่อสารปกติ และ MNP 10 สำหรับการสื่อสารที่มีสัญญาณรบกวนมาก อย่างเช่นการสื่อสารผ่านโทรศัพท์เคลื่อนที่

อีกกลุ่มหนึ่งก็คือกลุ่มมาตรฐานของ CCITT V.42 ซึ่งก็ได้รวมเอาเทคนิคของ MNP 4 เอาไว้แล้ว ดังนั้นจึงเป็นมาตรฐานที่นิยมกันมากกว่า MNP Class 4

การกำเนิดขึ้นของมาตรฐานการควบคุมสัญญาณผิดพลาดนี้ ทำให้การสื่อสารข้อมูลผ่านโมเด็มนั้นได้รับความนิยมมากกว่าเดิม โดยการทำงานทุกอย่างโมเด็มทั้งปลายทางและต้นทางจะตรวจสอบกันเองว่าข้อมูลถูกต้องหรือไม่โดยที่ผู้ใช้ไม่ต้องเข้าไปยุ่งด้วยเลยเพียงแต่ได้รับรายงานขั้นตอนการทำงานต่าง ๆ เท่านั้น นอกจากนี้ทั้ง MNP และ V.42 ก็มีความสามารถที่จะลดความเร็วในการรับส่งข้อมูลลงโดยอัตโนมัติถ้าหากพบว่ามันพบข้อมูลนั้นผิดพลาดมาก การลดความเร็วในการส่งนั้น เป็นวิธีการหนึ่งที่สามารถจำนวนข้อมูลที่ผิดพลาดได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.5 ประเภทของโมเด็ม

การเชื่อมต่อคอมพิวเตอร์สองเครื่องผ่านโมเด็มไปยังระบบโทรศัพท์ ไม่ใช่เรื่องราวทั้งหมดอย่างเด็ดขาด สิ่งที่ผู้ใช้รับรู้ก็คือ ข่าวสารหายเข้าไปในระบบโทรศัพท์ที่ปลายข้างหนึ่ง และปรากฏใหม่ที่ปลายอีกครั้งหนึ่ง ขณะเดียวกันสัญญาณอาจผ่านสายสัญญาณของระบบโทรศัพท์ท้องถิ่นไปยังระบบโทรศัพท์ทางไกล ถูกส่งขึ้นและลงจากดาวเทียมและผ่านเข้าไปในระบบโทรศัพท์ท้องถิ่นของผู้รับ การสื่อสารกับผู้อื่นผ่าน CompuServe อาจเกี่ยวกับช้องกับพนักงานเครือข่ายสองคนและเมนเฟรมคอมพิวเตอร์ของ CompuServe

โปรโตคอลของโมเด็มที่ใช้กันแพร่หลายในสหรัฐอเมริกา

1. โมเด็ม 300 -BPS

เมื่อสัญญาณแรงดันไฟฟ้าบวกและลบถูกส่งจากคอมพิวเตอร์ฝ่ายส่งไปให้โมเด็ม 300-BPS พวกมันจะถูกแปลงเป็นสองความถี่ ความถี่หนึ่งถูกใช้แทนแรงดันไฟฟ้าบวกและอีกความถี่หนึ่งแทนแรงดันไฟฟ้าลบ สำหรับการสื่อสารสองทางจะต้องใช้ความถี่แตกต่างกัน โดยใช้ความถี่สองความถี่สำหรับโมเด็มผู้เรียก (Originator modem) และสองความถี่ต่อไปนี้ถูกนิยามไว้ในมาตรฐาน Bell 103

- ผู้เรียกส่งลอจิก 0, แรงดันไฟฟ้าบวก : 1070 Hz
- ผู้เรียกส่งลอจิก 1, แรงดันไฟฟ้าลบ : 1270 Hz
- ผู้เรียกส่งลอจิก 0, แรงดันไฟฟ้าบวก : 2025 Hz
- ผู้เรียกส่งลอจิก 1, แรงดันไฟฟ้าลบ : 2225 Hz

เทคนิคนี้รู้จักกันในชื่อ Frequency Shift Keying หรือ FSK

2. โมเด็ม 1200-BPS

การสื่อสารของโมเด็มที่ 1200 BPS โดยทั่วไปทำตามระบบ Bell 212 A ซึ่งใช้เทคนิคของ Phase Modulation แทนการใช้ความถี่ที่ต่างกันสี่ความถี่เพื่อแทนค่าแต่ละบิต สัญญาณพาหะจะถูกมอดดูเลตทางเฟสเพื่อแทนค่าบิตต่าง ๆ

สัญญาณพาหะสามารถแทนค่าหนึ่งในสี่สถานะ และมันสามารถส่งข้อมูลได้สองบิตในครั้งหนึ่ง ตัวอย่างเช่น ถ้าเราเรียกสถานะทั้งสี่ว่า A ,B ,C และ D สถานะ A สามารถแทน 00 , สถานะ B แทน 01 , สถานะ C แทน 10 และ สถานะ D แทน 11 แม้ว่าปัจจัยทางเทคนิคจะจำกัดจำนวนครั้งที่สัญญาณสามารถเปลี่ยนจากสถานะหนึ่งไปอีกสถานะหนึ่งไว้ที่ 600 ครั้งต่อวินาที

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ข้อมูล 1200 บิต สามารถถูกส่งไปได้ในหนึ่งวินาที สถานะทั้งสี่ได้จากการใช้มุมเฟสที่ต่างกันสี่ เทคนิคนี้เรียกว่า Phase Shift Keying หรือ PSK สัญญาณพาหะที่ใช้มีสองความถี่โดยผู้เรียกใช้ 1200 Hz และผู้รับใช้ 2400 Hz

สังเกตุว่าโมเด็มที่อนุญาตให้ทำงานที่ 300 และ 1200 BPS จริง ๆ แล้วเป็นโมเด็มสองตัว รวมกัน คือ ตัวหนึ่งตามมาตรฐาน Bell 103 และตัวหนึ่งตามมาตรฐาน 212 A

3. โมเด็มตามมาตรฐาน V.32 9600 BPS

มาตรฐาน CCITT V.32 นิยามมาตรฐานสำหรับโมเด็ม 9200-BPS พร้อมกับคุณสมบัติ ดังนี้

- QAM (Quardrature Amplitude Modulation) ซึ่งผสมผสาน Phase และ Amplitude Modulation
- การเข้ารหัสแบบ Trillis ซึ่งทำให้มีการตรวจสอบและแก้ไขข้อผิดพลาด
- การทำงานแบบฟูลดูเพล็กซ์ (Full Duplex) หมายความว่าการส่งผ่านข้อมูลสามารถ เกิดขึ้นทั้งสองทิศทางที่ความเร็วเต็มที่พร้อมกัน

การสร้างตามเทคโนโลยีนี้มีราคาสูง โดยเฉพาะจากความต้องการฟูลดูเพล็กซ์ วงจรที่ซับ ซ้อนต้องถูกใช้เพื่อกำจัดการสะท้อน (Echo) ที่มีอยู่ตามธรรมชาติในระบบโทรศัพท์สาธารณะ บางส่วนของโมเด็มที่ใช้มาตรฐาน V.32 ได้แก่

- AT&T 2296A
- Hayes Ultra 96
- Microcom QX/V.32 C
- Telebit T2500
- VDS V.32
- US Robotics Courier V.32

4. โมเด็มความเร็วสูงแบบอื่น

มีบริษัทจำนวนหนึ่งได้พัฒนาเทคนิคเฉพาะตัว เพื่อให้ได้ความเร็ว 9600 BPS และสูงกว่า โมเด็มเหล่านี้มักจะถูกกว่าโมเด็ม V.32 ถ้าคุณต้องการใช้มันที่ความเร็วสูงสุดจะต้องมีโมเด็ม เหมือนกันทีปลายแต่ละข้างของการเชื่อมต่อ อย่างไรก็ตามโมเด็มพวกนี้มักจะสนับสนุนโปรโต คอลมาตรฐานที่มีความเร็วต่ำกว่าด้วย หมายความว่า ถ้าคุณมี Telebit Trailblazer Rlus ต่ออยู่กับ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ระบบโฮสต์ ผู้ใช้สามารถติดต่อที่ความเร็วสูงโดยใช้โมเด็ม Telebit ของตนเอง หรือที่ 1200 หรือ 2400 BPS โดยใช้โมเด็มมาตรฐานตัวอย่างบางส่วนของโมเด็มความเร็วสูงที่ไม่ใช้ V.32 ได้แก่

- Aayes V-series Smartmodem 9600
- USRobotics Courier HST
- Microcom AX/9624 c
- Telebit Trailblazer Plus

5. โมเด็มที่มีการแก้ไขข้อผิดพลาด

การตรวจสอบข้อผิดพลาดและการส่งซ้ำโดยอัตโนมัติได้ถูกสร้างไว้ในโมเด็มบางชนิด วิธีที่ใช้กันทั่วไปมีสองวิธี ตามที่บรรยายไว้ในมาตรฐาน V.42 ที่ออกโดย CCITT วิธีแรกเรียกว่า Link Acess Procedure for Modems (LAPM) วิธีที่สองแนะนำโดย Microcom Inc. เรียกว่า MNP 4 เช่นเดียวกันกับโมเด็มความเร็วสูงที่กล่าวถึงในหัวข้อก่อนคือต้อง แน่ใจว่าโมเด็มทั้งสองข้างใช้วิธีแก้ไขข้อผิดพลาดเหมือนกัน โมเด็มที่เข้ากับ V.42 ควรสนับสนุนทั้งสองวิธี

การแก้ไขข้อผิดพลาดมีความสำคัญเมื่อใช้ความเร็วสูงขึ้น และสัญญาณรบกวนสามารถทำลายการสื่อสารข้อมูลได้ แต่ไม่จำเป็นเสมอไปที่ต้องสร้างการแก้ไขผิดพลาดไว้ในโมเด็ม ข้อผิดพลาดของข้อมูลเป็นเรื่องวิกฤตที่สุดเมื่อกำลังส่งผ่านไบนารีไฟล์ เช่น โปรแกรม เพราะไบต์ที่ไม่ถูกต้องหนึ่งไบต์อาจทำให้โปรแกรมไม่สามารถทำงานได้ อย่างไรก็ตาม ถ้าคุณกำลังถ่ายโอน
ไบนารีไฟล์ คุณอาจกำลังใช้โปรโตคอลถ่ายโอนไฟล์ซึ่งจัดการตรวจสอบข้อผิดพลาดและการส่งซ้ำด้วยตนเองอยู่แล้ว ในกรณีเช่นนี้การใช้โมเด็มที่มีการแก้ไขข้อผิดพลาดจะเป็นการซ้ำซ้อน

3.6 โมเด็มกับการบีบอัดข้อมูล

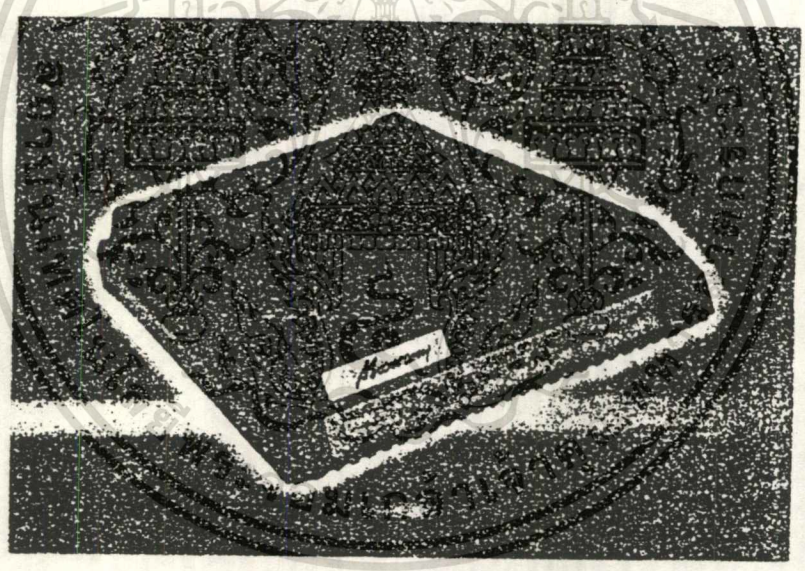
การบีบอัดข้อมูลเป็นเทคนิคทางซอฟแวร์ที่ถูกใช้เพื่อลดเวลาในการส่งข้อมูล การบีบอัดข้อมูลสามารถทำได้หลายวิธี ตัวอย่างเช่นเอกสารของเวิร์ดโปรเซสซิ่ง อาจมีช่องว่างอยู่เป็นจำนวนมาก ช่องว่างที่ติดต่อกันจำนวนหนึ่งสามารถถูกแทนด้วยรหัสพิเศษ ซึ่งขยายกลับคืนมาเป็นช่องว่างที่อีกด้านหนึ่ง วิธีนี้จะลดจำนวนตัวอักษรที่ถูกส่งและลดเวลาที่ใช้ในการส่งลงได้

การบีบอัดข้อมูลขณะนี้กำลังถูกสร้างไว้ในโมเด็ม ทำให้ได้ความเร็วและประสิทธิภาพของการส่งเพิ่มขึ้น มาตรฐานที่ใช้กันทั่วไป คือ V.42 bis กำหนดโดย CCITT และ MNP 5 พัฒนาโดย Microcom Inc. โมเด็มทั้งสองด้านต้องใช้วิธีเดียวกัน โมเด็มส่วนใหญ่เสนอการบีบอัดข้อมูลทั้งสองวิธี

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ในบางกรณีไม่จำเป็นต้องมีโมเด็ม MNP ที่แต่ละด้าน บางครั้ง MNP 5 ถูกใส่ไว้ในซอฟต์แวร์การสื่อสาร เพื่อที่มันจะสามารถใช้กับโมเด็มที่ไม่มี MNP ได้ การผสมผสาน MNP 5 กับโมเด็ม 9600 BPS V.32 สามารถให้ผลความเร็วประมาณ 18,000 BPS Microcoms มีเวอร์ชั่นเฉพาะตัวของ MNP เช่นกัน คือ MNP 9 ซึ่งเร็วขึ้นไปอีก รูปที่ 3.1 เป็นโมเด็ม Microcom QX/V.32 c ซึ่งให้ทรูพุต (Throughput) ได้ถึง 30,000 BPS ต่อไปนี้เป็นรายชื่อโมเด็มความเร็วสูงที่สนับสนุน MNP 5

- Microcom QX/V.32 (สนับสนุน MNP 9 ด้วยสำหรับ 30,000 BPS)
- Microcom QX/329 c (V.32 , MNP 5)
- USRobotics Courier V.32
- USRobotics Courier HST
- Hayes ULTRA Smartmodem 9600



รูปที่ 3.1 Microcom QX/V.32

เช่นเดียวกับกับการแก้ไขข้อผิดพลาด คือมันไม่จำเป็นเสมอไปที่ใส่การบีบอัดข้อมูลไว้ในโมเด็ม แม้ว่ามันสะดวกสบายน้อยกว่าก็ตาม เพราะว่าคุณสามารถบีบอัดไฟล์ไว้ก่อนที่จะส่งได้เสมอ (โดยการใช้โปรแกรมอย่าง PKZIP) และให้ผู้รับคลายการบีบอัดไฟล์ด้วยโปรแกรมแบบเดียวกัน

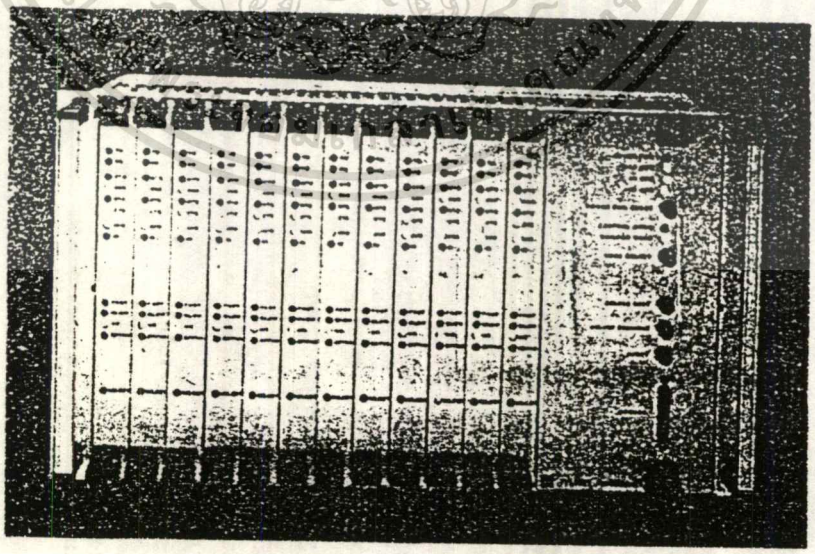
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.7 การเชื่อมต่อโมเด็มเข้ากับคอมพิวเตอร์

สำหรับคอมพิวเตอร์แล้ว โมเด็มภายนอก (External modem) เป็นเพียงอุปกรณ์อนุกรมอีกตัวหนึ่ง การเชื่อมต่อสายปหติสายทำตามปกติสำหรับข้อมูลที่ถูกส่งข้อมูลที่ได้รับซิกแนลกราวด์ และ แฮนด์เช็คกิ้ง

อาจมีการต่อสาย 8 , Carrier Detect (CD) เข้าด้วยเพื่อให้โมเด็มสามารถบอกคอมพิวเตอร์ให้รู้ว่ามีสัญญาณพาหะ และ Rng Indicator (RI) สาย 22 เพื่อบอกว่าโทรศัพท์กำลังส่งเสียงกริ่ง แรงดันไฟฟ้าบวกบน CD หมายถึงมีสัญญาณพาหะ แรงดันไฟฟ้าบวกบน RI หมายถึง โทรศัพท์กำลังส่งเสียงกริ่ง สัญญาณเหล่านี้จะถูกแยกแยะโดยคอมพิวเตอร์หรือไม่ขึ้นอยู่กับซอฟต์แวร์สื่อสารที่ใช้ โมเด็มบางชนิดส่งข้อความบนสายข้อมูลเพิ่มเติมจากการลดแรงดันไฟฟ้าบน CD เมื่อสัญญาณพาหะหายไปและส่งข่าวสาร เมื่อกริ่งโทรศัพท์ดังเพิ่มเติมจากการเพิ่มแรงดันไฟฟ้าบน RI ในกรณีของโมเด็มภายใน โมเด็มภายในถูกบรรจุไว้ในแผงวงจรที่ถูกเสียบเข้าโดยตรงในคอมพิวเตอร์ คอมพิวเตอร์จะมองเห็นเป็นแผงวงจรอนุกรมตัวหนึ่งเพื่อที่ซอฟต์แวร์จะไม่ต้องทราบว่าโมเด็มภายนอกหรือภายในถูกใช้อยู่โมเด็มภายนอกอาจเป็นแบบ Stand alone ดังแสดงในรูปที่ 3.1 หรือแบบ Rack-mounted ดังแสดงในรูปที่ 3.2

สังเกตุว่าสัญญาณฮาร์แวร์แฮนด์เช็คกิ้ง (DTR ,DSR , และอื่น ๆ) ถูกใช้เพื่อควบคุมการสื่อสารระหว่างคอมพิวเตอร์และโมเด็มพวกมันไม่ได้ถูกส่งไปตามสายไปยังโมเด็มและคอมพิวเตอร์ที่อยู่ห่างไกลดังนั้นเพื่อให้มีแฮนด์เช็คกิ้งระหว่างคอมพิวเตอร์สองเครื่อง ต้องใช้ซอฟต์แวร์แฮนด์เช็คกิ้งเข้าช่วย



รูปที่ 3.2 โมเด็มแบบ Rack-mounted

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 4

ชุดคำสั่งที่ใช้ในการสื่อสารข้อมูลผ่านโมเด็ม

ชุดคำสั่ง AT

โปรแกรมสื่อสารทั่วไปจะเป็นตัวดำเนินงานการติดต่อสื่อสารทั้งหมดระหว่างเครื่อง PC และโมเด็ม ผู้ใช้ส่วนมากมักจะไม่ค่อยสนใจการกำหนดคำสั่งต่างๆของโมเด็มจากโปรแกรมสื่อสารมากนัก เพียงแต่นำโมเด็มมาต่อเข้าที่พอร์ตสื่อสารอนุกรมของ PC แล้วก็เซตอัปชื่อและรุ่นของโมเด็มในการโปรแกรมสื่อสารให้ถูกต้องเท่านั้น จากนั้นก็สั่งให้โมเด็มหมุนเบอร์โทรศัพท์ออกด้วยปุ่มที่โปรแกรมสื่อสารกำหนด แต่ถ้าในกรณีที่เรียกโปรแกรมสื่อสารขึ้นมาแล้วปรากฏว่าโปรแกรมไม่สามารถติดต่อกับโมเด็มได้ ผู้ใช้ก็มักจะไม่ทราบวิธีการแก้ปัญหาเหล่านี้

ในกรณีเช่นนี้ สาเหตุอาจจะเนื่องมาจากการเซตอัปพารามิเตอร์ในโปรแกรมสื่อสาร หรือในตัวโมเด็มผิดพลาดไปเล็กน้อยทำให้ไม่เหมาะสมต่อสภาพแวดล้อมในการสื่อสาร ซึ่งสิ่งนี้เองที่ทำให้ผู้ใช้มีความจำเป็นจะต้องทราบรายละเอียดในการทำงาน และวิธีสั่งงานโมเด็ม

ในหัวข้อนี้ จะกล่าวถึงการใช้งานทุกคำสั่ง AT สำหรับโมเด็มทั่วๆไปที่ใช้มาตราฐานชุดคำสั่งนี้ โดยจะรวมทั้ง การตอบสนองคำสั่งของโมเด็ม (Result Code) การกำหนดค่ารีจิสเตอร์ s ต่างๆ (S-register) และการทำงานของรีจิสเตอร์เหล่านี้ ทั้งนี้จะอ้างอิงจากโมเด็มของ Hayes รุ่นที่มีความเร็ว 9600 BPS เป็นหลัก ซึ่งคำสั่งส่วนใหญ่จะใช้ได้กับโมเด็มของ Hayes รุ่นที่มีความเร็วต่ำกว่าได้ อย่างเช่น รุ่นความเร็ว 2400 , 1200 และ 300 BPS และยังสามารถใช้ได้กับโมเด็มของบริษัทอื่นที่มีความสามารถเทียบเท่ากับโมเด็มของ Hayes อีกด้วย , โดยคำสั่งต่างๆผู้ใช้สามารถศึกษาได้จากคู่มือของโมเด็มนั้นๆไปพร้อมๆกันซึ่งคำสั่งเหล่านี้มักจะไม่แตกต่างกันมากนัก

โมเด็มส่วนใหญ่จะถูกควบคุมการทำงานโดยชุดคำสั่งที่ส่งออกมาทางพอร์ตสื่อสารอนุกรม ซึ่งคำสั่งเหล่านี้สามารถสังเกตุได้จากขั้นตอนการเซตอัปประเภทโมเด็มในการสื่อสาร ผู้ใช้ต้องเลือกยี่ห้อ และรุ่นของโมเด็มให้ถูกต้อง ตามที่ใช้งาน เพื่อที่ว่าโปรแกรมสื่อสารจะได้คำสั่งเริ่มต้นที่ถูกต้องให้กับโมเด็ม แต่อย่างไรก็ตาม ชุดคำสั่งของโมเด็มจะมีคำสั่งที่เป็นมาตราฐานเดียวกัน โดยอ้างอิงจากชุดคำสั่งมาตรฐานของบริษัท Hayes และ บริษัท Hayes เองก็พยายามผลักดันให้ชุดคำสั่ง AT ของตน เป็นที่ยอมรับของคณะกรรมการ CCIT เพื่อพิจารณายอมรับเป็นชุดคำสั่งมาตรฐาน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ขั้นตอนการโทรศัพท์ออก

เมื่อผู้ใช้สั่งให้โมเด็มโทรศัพท์ไปยังโมเด็มปลายทาง กระบวนการต่างๆระหว่างโปรแกรมสื่อสาร และโมเด็มก็จะเริ่มขึ้น ถ้าหากผู้ใช้เคยใช้งานโมเด็มมาก่อนแล้ว ก็คงจะคุ้นเคยกับเสียงต่างๆ ที่ดังออกมาจากลำโพง ของโมเด็มเช่นเสียงไดอัลโทน ตามด้วยเสียง DTMF หรือ Pulse ในขณะทำการหมุนเลขหมายโทรศัพท์ปลายทาง เสียง Ringing Tone และเสียงสัญญาณพาหะโต้ตอบกันระหว่างโมเด็มต้นทาง และปลายทางหลังจากนั้นโมเด็มก็จะเงียบลงพร้อมกับเข้าสู่การเชื่อมต่อ (Connection) ซึ่งกระบวนการดังกล่าวนี้นับได้ว่า เป็นกระบวนการทำงานปกติที่ผู้ใช้คุณเคยรูปที่ 4.1 จะแสดงแผนผังระดับขั้นตอนการทำงาน โดยเริ่มตั้งแต่ขั้นตอนที่สั่งให้โมเด็มหมุนหมายเลขโทรศัพท์จนกระทั่งถึงการวางหูโทรศัพท์

จากรูปที่ 4.1 ผู้ใช้สามารถสังเกตุได้ว่าสิ่งที่ทำงานมากที่สุดในกระบวนการสื่อสารโมเด็มส่วนโปรแกรมสื่อสารจะมีหน้าที่เพียงส่งชุดคำสั่ง AT ไปให้กับโมเด็มเท่านั้น การกำหนดเวลาต่างๆในกระบวนการนี้ จะขึ้นอยู่กับข้อมูลที่อยู่ในรีจิสเตอร์ S ด้วยตัวอย่างเช่นรีจิสเตอร์ S7 จะเก็บค่าของเวลาที่โมเด็มจะลอเสียงตอบจากโมเด็มปลายทางเป็นต้น

การใช้งานชุดคำสั่ง AT

ขั้นตอนการเชื่อมต่อระหว่างโมเด็มต้นทาง และปลายทางที่ได้กล่าวไปแล้วนั้นโมเด็มทั้งคู่จำเป็นต้องมีพื้นฐานที่เหมือนกันบางประการ เช่น ความเร็ว รูปแบบของข้อมูลและโปรโตคอลในการรับส่งข้อมูล โดยการเชื่อมต่อโมเด็มในแต่ละครั้ง อย่างน้อยที่สุด ผู้ที่ใช้โมเด็มต้นทางจำเป็นจะต้องทราบว่าจะให้โมเด็มโทรศัพท์ออกด้วยหมายเลขใด ควรจะตั้งเวลาในการรอเสียงตอบจากโมเด็มปลายทางนานเท่าไร ระบบโทรศัพท์ของตนนั้นใช้ระบบ Tone หรือ Pulse และในขณะเดียวกันผู้ที่ใช้โมเด็มปลายทางก็จะต้องกำหนดให้โมเด็มรับทราบด้วยว่าจะให้โมเด็มรับสายโทรเข้าหลังจากที่มีเสียง Ringing กี่ครั้งและจะต้องรอสัญญาณได้ตอบนานเท่าไรเป็นต้น ซึ่งการดำเนินงานทั้งหมดนี้สามารถควบคุมได้โดยใช้ชุดคำสั่ง AT

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ขั้นตอน	ผู้ใช้	ซอฟต์แวร์	โมเดิมต้นทาง	โมเดิมปลายทาง
1	เลือกคำสั่ง "Dial" จากซอฟต์แวร์	เปิดสัญญาณ DRT เพื่อส่งคำสั่งหมุนหมายเลขไปยังโมเด็ม โดยใช้คำสั่ง: ATDT 555-1234	เปิดลำโพง ยกหูโทรศัพท์ รอสัญญาณให้หมุนหมายเลข	
2		รอ result codes จากโมเด็ม	รอการตอบรับจากปลายทาง ทั้งนี้ระยะเวลาการรอขึ้นกับการกำหนดค่ารีจิสเตอร์	
3				เสียงโทรศัพท์ดัง
4				ตอบรับสัญญาณ
5			รับรู้สัญญาณตอบรับและทำการส่งสัญญาณ Originate Carrier	
6			รับทราบวิธีการมอดูเลชันและความเร็ววงแต่ละฝ่าย	รับทราบวิธีการมอดูเลชันและความเร็ววงแต่ละฝ่าย
7			โมเด็มตกลงรับรู้โปรโตคอลการควบคุมความผิดพลาดและการบีบอัดข้อมูลของแต่ละฝ่าย	
8			ส่ง result code "Connect" ไปยัง PC เปิดลำโพงและเปิดสัญญาณ CD	
9		รับรู้ result code และสัญญาณ CD รายงานให้ผู้ใช้ทราบว่าการติดต่อได้เกิดขึ้นแล้ว		
10	เริ่มการติดต่อกับโฮสคอมพิวเตอร์	ดำเนินการสื่อสารและคอยดูสัญญาณที่ขาดหายไปจากหน้าจอสัญญาณ CD	ส่งและรับข้อมูล	ส่งและรับข้อมูล
11	การสื่อสารเสร็จสมบูรณ์เลือกคำสั่ง "Disconnect"	ปิดสัญญาณ DRT หรือส่ง+++ ตามด้วยคำสั่ง ATH		
12			วางสายโทรศัพท์	ยกเลิกสัญญาณ วางสายโทรศัพท์

รูปที่ 4.1 กระบวนการทำงานของโมเด็มต้นทางปลายทาง

เอกสารนี้เป็นเอกสาร[illegible]ให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ประเภทของชุดคำสั่ง AT

โดยพื้นฐานแล้ว เราสามารถแบ่งชุดคำสั่ง AT ออกเป็น 2 ประเภทใหญ่ๆได้ดังนี้

- ประเภทแรกคือชุดคำสั่งที่ใช้ในการปฏิบัติงานอย่างเช่น ATD (คำสั่งให้หมุนหมายเลขโทรศัพท์) หรือ ATH (คำสั่งให้วางสายโทรศัพท์) เป็นต้น

- ประเภทที่สองคือ ชุดคำสั่งที่ใช้กำหนดค่า หรือเปลี่ยนแปลงค่าต่างๆอย่างเช่น ATS7=90 เป็นการกำหนดค่าให้รีจิสเตอร์ S7 มีค่า 90 ซึ่งก็คือโมเด็มรอการตอบรับจากโมเด็มปลายทางเป็นเวลา 90 วินาที หลังจากนั้นหากยังไม่มีสัญญาณใดๆ ตอบกลับมาโมเด็มจะวางสายทันที ตามปกติแล้วโปรแกรมสื่อสารมักจะตั้งค่าของรีจิสเตอร์ S ต่างๆที่จำเป็นเอาไว้ตั้งแต่ตอนที่ผู้ใช้เรียกโปรแกรมนั้นๆขึ้นมา ผู้ใช้จะสามารถควบคุมและเปลี่ยนแปลงค่าของรีจิสเตอร์ S เหล่านี้ได้ในภายหลัง

การออนไลน์และออฟไลน์

สภาวะออฟไลน์นั้นสามารถเรียกได้อีกอย่างหนึ่งว่าสภาวะคำสั่ง (Command state) หมายถึงสภาวะที่ผู้ใช้สามารถจะส่งคำสั่งต่างๆไปยังโมเด็มได้ หรือจะอีกนับหนึ่งก็คือ สภาวะที่โมเด็มจะแปลความหมายของข้อมูลที่ได้รับมาจาก PC ให้เป็นคำสั่งเท่านั้น ซึ่งสภาวะนี้โมเด็มจะไม่ได้รับส่งข้อมูลกับโมเด็มปลายทางแต่จะสื่อสารกับ PC เท่านั้น ส่วนสภาวะออนไลน์หมายถึงสภาวะที่โมเด็มได้เชื่อมต่อกับโมเด็มปลายทางเป็นที่เรียบร้อยแล้ว ข้อมูลที่ส่งจาก PC ก็จะผ่านจากโมเด็มต้นทางไปยังโมเด็มปลายทางเสมอ ในกรณีนี้ถ้าหากผู้ใช้ต้องการจะส่งคำสั่งให้กับโมเด็มหรือต้องการจะให้โมเด็มกลับมาอยู่ในสภาวะออฟไลน์ เพื่อรับคำสั่งจาก PC ก็สามารถทำได้โดย 2 วิธีคือ

- ให้โมเด็มวางสายแล้วกลับมาอยู่ในสภาวะออนไลน์ใหม่

- ส่งชุดอักขระ Escape Sequence เข้าไปยังโมเด็ม ในขณะที่ออนไลน์

ซึ่งวิธีหลังมีความเหมาสมมากกว่าแพราะการเชื่อมต่อระหว่างโมเด็มจะยังคงดำเนินอยู่และหลังจากได้รับคำสั่งต่างๆให้กับโมเด็ม เป็นที่เรียบร้อยแล้ว ก็สามารถกลับเข้าอยู่ในสภาวะออนไลน์ได้เช่นเดิม วิธีการส่งชุดอักขระ Escape Sequence สำหรับโมเด็มที่เข้ากันได้กับโมเด็มของ Hayes คือให้รอ 1 วินาที (เรียกว่า Guard Time) แล้วกดปุ่ม + ติดกันสามครั้ง (+++) การที่ต้องรอ 1 วินาทีก่อนที่จะกดปุ่มเครื่องหมาย + เป็นสิ่งที่จำเป็นเนื่องจากว่าโมเด็มจะได้รู้ว่าอักษร + นั้นเป็นส่วนของอักขระ Escape Sequence ไม่ใช่ส่วนของข้อมูลที่จะต้องส่งไปโมเด็มปลายทาง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปแบบของชุดคำสั่ง AT

การใช้ชุดคำสั่ง AT จำเป็นต้องขึ้นต้นอักษร AT เสมอ และจะต้องจบลงด้วยการกด ENTER หรือ Cariage return ยกเว้นเฉพาะคำสั่ง A/ (ไม่ต้องการกดปุ่ม ENTER) ซึ่งหมายถึงให้โมเด็มกลับไปทำคำสั่งล่าสุดซ้ำอีกครั้งหนึ่งการขึ้นต้นด้วยอักษร AT ก็เพราะต้องการให้โมเด็มรับรู้ถึงความเร็ว และรูปแบบของอักษรคำสั่งที่ถูกส่งออกมาจากพอร์ตสื่อสารอนุกรมของ PC ซึ่งอย่าสับสนระหว่างความเร็วของพอร์ตสื่อสารอนุกรม (ความเร็วของ DTE) และความเร็วของโมเด็ม (ความเร็วของ DCE) ในกรณีนี้จะหมายถึงความเร็วของ DTE

ชุดคำสั่งเพิ่มเติม

ชุดคำสั่งแรกๆจะใช้ตัวอักษรตั้งแต่ A-Z เช่น ATA จนถึง ATZ ซึ่งก็หมายความว่าคำสั่งนี้ใช้ได้ไม่เกิน 26 คำสั่งเท่านั้น แต่โมเด็มในปัจจุบันมีความสามารถต่างๆเพิ่มมากขึ้นจึงได้มีการพัฒนาใช้ตัวอักษร & เป็นตัวเข้ามาประกอบ เช่น AT&A หมายถึงให้โมเด็มเรียกค่า Default Setting ขึ้นมาใช้งานและชุดตัวอักษรอื่นๆอีก เช่น % หรือ * เป็นต้น

การส่งหลายคำสั่งในครั้งเดียว

ในกรณีที่ต้องการส่งคำสั่งหลายคำสั่ง ผู้ใช้อาจจะส่งคำสั่งทั้งหมดในครั้งเดียว ซึ่งโมเด็มไม่สามารถรับเกิน 40 ตัวอักษรทั้งนี้จะนับตัวอักษร AT และ Cariage return ด้วย ผู้ใช้จะต้องพิมพ์คำสั่งทั้งหมดติดกันหรือเว้นช่องว่างแต่ละคำสั่งได้ดังตัวอย่าง ATS7=90 V1X4DT13055551234 ซึ่ง S7 = 90 หมายถึงใส่ค่า 90 ลงในรีจิสเตอร์ S7 , V1 หมายถึงอนุญาติให้โมเด็มส่ง Result Code ทุกครั้งที่ได้รับคำสั่งต่างๆ , X4 หมายถึงให้โมเด็มส่ง Result Code ออกมาได้บางคำสั่ง , DT1-305-555-1234 หมายถึงให้โมเด็มหมุนหมายเลขโทรศัพท์ 1-305-555-1234 โดยหมุนแบบระบบโทน

Data Terminal Ready

สัญญาณ Data Terminal Ready (DTR) คือสัญญาณไฟฟ้าที่ออกจาก RS-232-C ซึ่งเป็นสัญญาณที่ส่งออกมาจาก PC เพื่อให้โมเด็มทราบว่าขณะนั้น PC พร้อมจะติดต่อกับโมเด็มแล้ว ถ้าสัญญาณ DTR มีค่าเป็นศูนย์ขณะติดต่อกับโมเด็มปลายทาง ก็จะทำให้โมเด็มหยุดการสื่อสารและวางหูทันที ดังนั้นถ้าหากผู้ใช้ต้องการหยุดการสื่อสารก็อาจกำหนดให้โปรแกรมสื่อสารส่งสัญญาณ DTR ในลักษณะนี้ก็ได้ ซึ่งจะสะดวกกว่าวิธีการใช้คำสั่งแบบยุคต้นๆ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ชุดคำสั่ง AT ที่สำคัญ

ATA (รับโทรศัพท์)

คำสั่งนี้ทำให้โมเด็มรับสายโทรศัพท์ได้แต่จะใช้เท่าที่จำเป็นเท่านั้นโดยปกติแล้วผู้ใช้จะสามารถกำหนดให้โมเด็มรับสายทุกครั้งที่มีเสียง Ringing โดยตั้งค่าไว้ในรีจิสเตอร์ S0 แต่ในบางครั้งผู้ใช้อาจต้องการให้โมเด็มสามารถรับสายได้ทันทีทั้งๆที่ค่าในรีจิสเตอร์ไม่ได้ตั้งไว้ให้รับสายโทรศัพท์โดยการใช้คำสั่ง ATA ซึ่งหลังจากที่โมเด็มรับคำสั่งนี้โมเด็มก็จะยกหูโทรศัพท์ขึ้น (Off-hook) และจะส่งสัญญาณพาหะ (Carrier) ไปยังปลายทาง และรอสัญญาณตอบกลับมาถ้าไม่มีสัญญาณตอบกลับโมเด็มก็จะวางหู และกลับมาอยู่ในสภาวะคำสั่ง (Off-line) อีกครั้ง

ATD (หมุนหมายเลขโทรศัพท์)

คำสั่งนี้จะทำให้โมเด็มยกหูขึ้น และหมุนหมายเลขโทรศัพท์ออกด้วย หมายเลขที่กำหนดไว้ในพารามิเตอร์ของคำสั่งนี้ โดยมีรูปแบบดังตัวอย่าง ATD3267313 ซึ่งคำสั่ง ATD นี้ต้องการพารามิเตอร์บางอย่างทุกครั้งที่คุณต้องการใช้คำสั่งนี้ทุกครั้งเสมอ เช่น " T " (Tone) หรือ " , " ดังแสดงในตารางที่ 4.1

ตารางที่ 4.1 พารามิเตอร์ที่ใช้ในการหมุนหมายเลขโทรศัพท์

พารามิเตอร์	ความหมาย
!	วางหูโทรศัพท์ชั่วคราว
,	หยุดรอเป็นเวลา 2 วินาทีขณะหมุนหมายเลข
;	หลังจากหมุนหมายเลขแล้วจะส่งไปที่สภาวะคำสั่ง
P	หมุนหมายเลขด้วยระบบ Pulse
R	เรียกโหมดรีเวิร์ส
S=x	เก็บหมายเลขโทรศัพท์ไว้ที่พื้นที่ x
T	หมุนหมายเลขด้วยระบบ Tone
W	รอฟังเสียงสัญญาณ"สายว่าง"

โดยปกติแล้วถ้าใช้คำสั่งเพียง ATD และตามด้วยหมายเลขจะหมายความว่าให้โมเด็มทำการหมุนหมายเลขโทรศัพท์ด้วยวิธีล่าสุดที่เคยหมุน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ATE (ปิดและเปิด)

Echo เป็นคำสั่งที่ใช้ในการควบคุมการแสดงตัวอักษรที่ผู้ใช้กำลังพิมพ์ออกมาทางจอภาพ ถ้า Echo เป็น off หมายความว่า เมื่อกดปุ่มใดๆแล้วไมาสามารถเห็นตัวอักษรบนหน้าจอในขณะที่พิมพ์เข้าไป ลักษณะการใช้คำสั่งเป็นดังนี้

ATE0 หรือ ATE : ECHO OFF

ATE1 : ECHO ON (DEFAULT)

ATH (ยกหูและวางหูโทรศัพท์)

คำสั่งนี้ทำให้โมเด็มยกหูวางหูโทรศัพท์ โดยมีรูปแบบดังนี้

ATH0 หรือ ATH : ให้โมเด็มวางหูโทรศัพท์

ATH1 : ให้โมเด็มยกหูโทรศัพท์

คำสั่ง ATH0 หรือ ATH มักจะใช้หลังจากที่ส่ง Escape Sequence (+++) ให้แก่โมเด็ม เพื่อให้โมเด็มเข้าสู่สภาวะคำสั่งซึ่งโปรแกรมสื่อสารทั่วๆไปมักจะใช้วิธีนี้เพื่อทำการหยุดทำการสื่อสาร (Disconnect)

ATL (ควบคุมความดังของเสียง)

คำสั่งนี้ใช้ควบคุมความดังของเสียงจากลำโพงของโมเด็มได้แต่โมเด็มบางรุ่นที่เป็นแบบ Pocket หรือกล่องเล็กๆ ใช้ต่ออยู่ภายนอก PC บางประเภทอาจไม่มีลำโพง ดังนั้นจึงไม่ควรใช้คำสั่งนี้

ATL0 หรือ ATL : หมายถึงให้มีเสียงเบาสุด

ATL1 : หมายถึงให้มีเสียงดังปานกลาง

ATL2 : หมายถึงให้มีเสียงดังที่สุด

ATM (ควบคุมการเกิดเสียง)

ลักษณะของคำสั่งมีรูปแบบดังนี้

ATM0 หรือ ATM : หมายถึงให้โมเด็มปิดเสียงทั้งหมด

ATM1 : หมายถึงให้โมเด็มส่งเสียงได้ขณะที่หมุนหมายเลขและปิดเสียงเมื่อเชื่อมต่อกับปลายทางเรียบร้อยแล้ว

ATM2 : หมายถึงอนุญาติให้โมเด็มส่งเสียงออกจากลำโพงตลอดเวลา

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ATM3 : หมายถึงให้โมเด็มปิดเสียงขณะที่หมุนหมายเลข และหลังจากนั้นก็ทำการเปิดเสียงจะกระทั่งได้รับสัญญาณพาหะจากโมเด็มปลายทางจึงจะปิดเสียงอีกครั้งหนึ่ง

ATO (กลับสู่สภาวะออนไลน์)

การใช้คำสั่งนี้ในสภาวะคำสั่ง (Command state) จะทำให้โมเด็มก็จะเข้าไปสู่สภาวะออนไลน์และผู้ใช้กดรหัส Escape Squence (+++) ออกมาสู่สภาวะคำสั่ง เพื่อออกคำสั่ง AT ต่างๆแก่โมเด็มแล้วต้องการกลับเข้าไปอยู่ในสภาวะออนไล์อีกครั้งหนึ่ง

ATQ (ควบคุม Result Code)

คำสั่งนี้ใช้ควบคุมการส่ง Result Code จากโมเด็มสู่ PC โดยปกติแล้วโมเด็มได้รับคำสั่งใดๆ ก็จะส่งชุดอักขระกลับออกมาอย่างเช่น "OK" "CONNECT" หรือ "ERROR" เป็นต้น รูปแบบการใช้คำสั่งมีต่อไปนี้

ATQ0 หรือ ATQ : อนุญาติให้โมเด็มส่ง Result code ออกมาได้

ATQ1 : ไม่อนุญาตให้โมเด็มส่ง Result code ออกมา

ATQ2 : ให้โมเด็มส่ง Result code ออกมา ถ้าเป็นโมเด็มตัวต้นทาง และไม่ให้ ส่ง Result code ออกมา ถ้าเป็นโมเด็มปลายทาง

ATS (คำสั่งตั้งค่าให้แก่รีจิสเตอร์ S)

ชุดคำสั่ง ATS จะช่วยให้คุณสามารถขอดูและเปลี่ยนแปลงค่าที่อยู่ในรีจิสเตอร์ S ต่างๆได้ สำหรับการตั้งค่ารีจิสเตอร์ S มีรูปแบบดังนี้

ATSx=y

โดยที่ x เป็นหมายเลขรีจิสเตอร์ S และ y หมายถึงค่าที่ต้องการบรรจุลงในรีจิสเตอร์ S นั้นๆดังตัวอย่างถ้าหากต้องการกำหนดให้รีจิสเตอร์ S ตัวที่ 11 มีค่าเท่ากับ 50 ก็ให้ใช้คำสั่งดังนี้ ATS11=50 และสำหรับการดูค่าในรีจิสเตอร์ S ก็จะมีาปแบบดังนี้ ATSx? หรืออาจใช้คำสั่ง AT&V เพื่อขอดูค่ารีจิสเตอร์ S ทั้งหมดรวมทั้งข้อมูลการเซตทั้งหมดที่เก็บเอาไว้ใน NVRAM สำหรับรีจิสเตอร์ S ที่สำคัญมีดังต่อไปนี้

S0 : ใช้เก็บค่าจำนวนครั้งของกระดิ่งโทรศัพท์ก่อนที่โมเด็มจะรับสายโทรศัพท์สามารถตั้งค่าให้สูงได้ถึง 255 (หมายถึงให้เสียงกระดิ่งดัง 255 ครั้งแล้วโมเด็มจึงจะรับโทรศัพท์) หรือถ้าตั้งเป็น 0 ก็จะมีความหมายว่าไม่ให้โมเด็มรับสายโทรเข้า

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

S1 : ใช้เก็บค่าจำนวนครั้งของเสียงกระดิ่งเป็นรีจิสเตอร์แบบอ่านได้อย่างเดียวไม่สามารถจะเปลี่ยนแปลงค่าได้โดยการใช้คำสั่ง AT ใดๆ

S2 : รีจิสเตอร์ตัวนี้ใช้เก็บค่ารหัส ASCII (ในลักษณะเลขฐาน 10) ที่ต้องการใช้เป็นรหัส Escape Sequence โดยปกติแล้วจะมีดีฟอลต์เป็น 43 ซึ่งจะเหมือนกับการกดอักษร +

S3 : รีจิสเตอร์ตัวนี้ใช้เก็บค่ารหัส ASCII (ในลักษณะเลขฐาน 10) ที่ต้องการใช้เป็นรหัส Carisge Return(CR) โดยปกติแล้วจะมีดีฟอลต์เป็น 13 ซึ่งจะเหมือนกับการกดปุ่ม ENTER

S4 : รีจิสเตอร์ตัวนี้ใช้เก็บค่ารหัส ASCII (ในลักษณะเลขฐาน 10) ที่ต้องการใช้เป็นรหัส Line Feed โดยปกติแล้วจะมีดีฟอลต์เป็น 10

S5 : รีจิสเตอร์ตัวนี้ใช้เก็บค่ารหัส ASCII (ในลักษณะเลขฐาน 10) ที่ต้องการใช้เป็นรหัส Backspace โดยปกติแล้วจะมีดีฟอลต์เป็น 8 ซึ่งจะเหมือนกับการกดปุ่ม Backspace

S6 : ใช้เก็บค่าของเวลาในการหยุดรอ ก่อนหมุนหมายเลขโทรศัพท์ออก มีหน่วยเป็นวินาที ซึ่งค่ารีจิสเตอร์ S6 นี่จะใช้งานก็ต่อเมื่อโมเด็มถูกสั่งให้ไม่สนใจฟังไดอัลโทน หรือคำสั่ง ATXn หลังจากนั้นเมื่อต้องการให้โมเด็มหมุนหมายเลขโทรศัพท์ออก โมเด็มก็จะไม่สนใจว่ามีเสียงไดอัลโทนหรือไม่และจะรอเป็นเวลา S6 วินาที จึงหมุนหมายเลขโทรศัพท์ออก ค่าในรีจิสเตอร์ S6 นี่สามารถมีได้ตั้งแต่ 1 ไปจนถึง 255 วินาที โดยมีดีฟอลต์เป็น 50 วินาที (เพื่อเข้าใจ ขอให้ดูที่คำอธิบายคำสั่ง ATD และ ATX ประกอบ)

S7 : ใช้เก็บค่าของเวลาในการรอฟังสัญญาณพาหะของโมเด็มปลายทางมีหน่วยเป็นวินาที หลังจากที่โมเด็มหมุนหมายเลขโทรศัพท์ออกหรือรับสายโทรศัพท์เข้าโมเด็มจะรอฟังสัญญาณพาหะเป็นเวลา S7 วินาที ซึ่งในกรณีที่ต้องการโทรศัพท์ระหว่างประเทศ ก็อาจจะต้องตั้งค่า S7 ให้มากกว่า 50 วินาที แล้วแต่ความเหมาะสม

S8 : ใช้เก็บค่าของเวลาในการรอ มีหน่วยเป็นวินาที หลังจากที่โมเด็มได้รับพารามิเตอร์ "," ในคำสั่ง ATD โมเด็มก็จะหยุดรอเป็นเวลา S8 วินาที ก่อนที่จะทำคำสั่ง หรือพารามิเตอร์ตัวต่อไป S8 นี้จะสามารถมีค่าได้ตั้งแต่ 2 ไปจนถึง 255 วินาที แต่โดยปกติแล้วจะมีดีฟอลต์เป็น 2 วินาที

S9 : ใช้เก็บค่าของช่วงเวลาที่โมเด็มรอรับทราบสัญญาณพาหะของโมเด็มปลายทาง มีหน่วยเป็น 1/10 วินาที โดยที่โมเด็มจะรอสัญญาณพาหะจากปลายทางภายในช่วงเวลาดังกล่าว แล้วจากนั้นโมเด็มก็ จะส่งสัญญาณ Carrier Detect (CD) ที่ขา 8 ของขั้วต่อ RS-232-C ไปยัง PC ผู้ใช้สามารถกำหนด ค่า S9 ได้ตั้งแต่ 1 ถึง 255

S10 : มีหน่วยเป็น 1/10 วินาที มีค่าได้ตั้งแต่ 1 ถึง 255 มักจะใช้อยู่ 2 กรณี ซึ่งกรณีแรกจะใช้เก็บช่วงเวลาที่โมเด็มจะทำการสัญญาณพาหะที่ขาดหายไปในขณะกำลังออนไลน์ ซึ่งอาจจะในช่วงเวลาดังกล่าว โมเด็มก็จะว่างสายทันที ส่วนกรณีที่สองก็คือจะใช้เก็บค่าของช่วงเวลาในการ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รอสัญญาณพาหะก่อนที่จะวางสาย เมื่อทำการวางสายเรียบร้อยแล้วจะให้โมเด็มตรวจสอบว่าการสื่อสารได้ตัดขาดเรียบร้อยแล้วหรือไม่ โดยปกติแล้วค่าดีฟอลต์เป็น 14 ซึ่งก็คือให้โมเด็มหยุดรอสัญญาณพาหะเป็นเวลา 1.4 วินาทีหลังจากการสื่อสารถูกตัวขาดไป และในทางกลับกันถ้าต้องการจะให้โมเด็มค้างสัญญาณ CD ได้ตลอดเวลา (เพื่อเป็นการหลอกเครื่อง PC) ก็สามารถตั้งค่าไว้เป็น 255 ได้เช่นกัน

S11 : ใช้เก็บค่าในช่วงเวลาในการหมุนโทรศัพท์แบบ Tone แต่ละหมายเลขมีหน่วยเป็นมิลลิวินาที ในขณะที่โมเด็มกำลังหมุนหมายเลขโทรศัพท์ออก ก็จะเว้นช่วงเวลาไว้ประมาณ S11 มิลลิวินาทีระหว่างหมายเลขโทรศัพท์แต่ละตัว ซึ่งค่าของ S11 นี้จะมีผลเฉพาะการหมุนโทรศัพท์แบบ Tone เท่านั้น (ATDT) ไม่มีผลต่อการหมุนแบบ Pulse (ATDP) ผู้ใช้สามารถกำหนดค่าใน S11 ได้ตั้งแต่ 50 ถึง 255 (0.05-0.255 วินาที) โดยมีดีฟอลต์เป็น 95 (0.095 วินาที)

S12 : ใช้เก็บค่าของช่วงเวลาที่โมเด็มหยุดรอ Escape Sequence มีหน่วยเป็น 1/50 วินาที ถ้าหากผู้ใช้ต้องการออกจากสภาวะออนไลน์ กลับเข้าสู่สภาวะคำสั่ง (Command state) จะต้องหยุดรอเป็นเวลา S12 วินาที แล้วจึงสามารถส่ง Escape Sequence ให้แก่โมเด็มได้ มิฉะนั้นโมเด็มก็จะไม่รับรู้และปล่อยให้ Escape Sequence เป็นข้อมูลธรรมดาผ่านสายโทรศัพท์ออกไป รีจิสเตอร์ S12 สามารถมีค่าตั้งแต่ 0 ไปจนถึง 255 (0-5.1 วินาที) โดยมีค่าดีฟอลต์เป็น 50 (1 วินาที)

ATV (กำหนด Verbose Mode)

คำสั่งนี้ใช้กำหนดชนิดของ Result code ว่าจะให้ส่งเป็นแบบตัวอักษร เป็น “OK”, “CONNECT” หรือเป็นแบบตัวเลข โดยมีรูปแบบดังนี้

ATV0 หรือ ATV : ให้โมเด็มส่ง Result code เป็นตัวเลข

ATV1 : ให้โมเด็มส่ง Result code เป็นตัวอักษรโดยมีดีฟอลต์เป็น V1

ATWn (กำหนด Result Code เพิ่มเติม)

โมเด็มสามารถแสดงค่า Result code เพิ่มเติมได้ ซึ่งเรียกว่า Negotiation Progress Codes ซึ่งมักจะมีใช้ในโมเด็มสนับสนุนโปรโตคอลควบคุม ความผิดพลาด อย่างเช่น MNP และ V.42

ข้อความของ Result code มักจะใช้แสดงการควบคุมความผิดพลาดและการบีบขนาดของข้อมูลระหว่างโมเด็มต้นทาง และปลายทาง ซึ่งโปรแกรมโดยทั่วไปจะไม่สนับสนุน Result code ประเภทนี้ จึงอาจจะทำให้ตีความผิดพลาดและส่งข้อความมาว่า “ERROR”

รูปแบบการใช้คำสั่ง

ATW0 หรือ ATW : ไม่อนุญาตให้โมเด็มส่ง Negotiation Progress Codes ให้แก่ PC

ATW1 : อนุญาตให้โมเด็มส่ง Negotiation Progress Codes ให้แก่ PC

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ATX (กำหนดชนิดของ Result code)

โดยทั่วไปโมเด็มจะมี Result code คือ "OK" , "RING" , "NO CARRIER" และ "ERROR" แต่โมเด็มที่ออกมาใหม่บางรุ่นจะมี Result code เพิ่มเติมดังตารางที่ 4.2

ตารางที่ 4.2 คำสั่ง ATX

คำสั่ง	รูปแบบ
ATX0	ใช้ result code ห้าข้อความคือ OK, CONNECT, RING, NO CARRIER และ ERROR เมื่อเลือกตัวเลือกนี้ สัญญาณหมุนหมายเลขและสัญญาณสายไม่ว่างจะไม่ทำงาน
ATX1	เหมือนกับคำสั่ง ATX0 แต่เพิ่ม result code ที่เกี่ยวกับความเร็วเข้าไป เช่น CONNECT 2400
ATX2	ทุกตัวเลือกของคำสั่ง ATX1 แต่เพิ่ม result code ข้อความ "No Dial Tone" เข้าไป
ATX3	ทุกตัวเลือกของคำสั่ง ATX1 แต่เพิ่ม result code ข้อความ "Busy" เข้าไป
ATX4	เลือก result code ได้ทุกตัว

ATZ (รีเซตโมเด็ม)

เป็นคำสั่งที่นำข้อมูลที่อยู่ใน NVRAM มาเก็บใน Active Profile เพื่อใช้งานได้ โดยมีรูปแบบดังต่อไปนี้

ATZ0 หรือ ATZ : โหลดข้อมูลข้อมูลการเซตจากโพรไฟล์ที่ 0 กลับมาเก็บไว้ในแอกทีฟโพรไฟล์

ATZ1 : โหลดข้อมูลการเซตจากโพรไฟล์ที่ 1กลับมาเก็บไว้ในแอกทีฟโพรไฟล์

AT&C (ควบคุมสัญญาณ CD)

คำสั่งนี้ใช้ควบคุมการส่งสัญญาณ CD ให้แก่ PC (ขา 8 ของคอนเน็กเตอร์ RS-232-C) ซึ่งสัญญาณนี้จะมีสถานะ "ON" อยู่ตลอดเวลาแม้ว่าจะไม่ได้เชื่อมต่อกับโมเด็มตัวอื่นๆ ผู้ใช้สามารถบังคับและควบคุมสัญญาณ CD ได้ โดยมีรูปแบบดังตารางที่ 4.3

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตารางที่ 4.3 คำสั่ง AT&C

คำสั่ง	รูปแบบ
AT&C0	สัญญาณ CD จะอยู่ในสถานะ "ON" เสมอ ไม่ว่าโมเด็มจะออน-ไลน์อยู่หรือไม่ก็ตาม
AT&C1	สัญญาณ CD จะอยู่ในสถานะ "ON" เมื่อโมเด็มออนไลน์ และมีการเชื่อมต่อกับโมเด็มตัวอื่นๆ เท่านั้น
AT&C2	สัญญาณ CD จะอยู่ในสถานะ "ON" เมื่ออยู่ในสภาวะคำสั่ง (Command State) แต่หลังจากที่โมเด็มทำการเชื่อมต่อเรียบร้อยแล้ว สัญญาณ CD จะมีสถานะตามความเป็นจริง
AT&C3 หรือ	สัญญาณ CD จะอยู่ในสถานะ "ON" เสมอไม่ว่าโมเด็มจะออน-ไลน์อยู่หรือไม่

AT&D (ควบคุมการรับรู้สัญญาณ DTR)

คำสั่งนี้ใช้ควบคุมการรับรู้สัญญาณ DTR (Data Terminal Ready) ที่ส่งออกมาจาก PC ซึ่ง จะเป็นสัญญาณที่ขา 20 ของคอนเน็กเตอร์ RS-232-C ปกติแล้ว โมเด็มจะคอยตรวจสอบว่า - สัญญาณนี้ อยู่ในสถานะ " ON" หรือไม่ ถ้าไม่โมเด็มก็จะไม่รับรู้ว่ามี PC ต่ออยู่กับโมเด็ม คำสั่ง AT&D นี้จะสามารถทำให้ผู้ใช้บังคับให้โมเด็มรับรู้ หรือไม่รับสัญญาณ DTR ได้ โดยมีรูปแบบดังตารางที่ 4.4

AT&K (เลือกชนิดของการควบคุม การไหลของข้อมูล)

จากที่กล่าวมาแล้วในข้างต้นที่เกี่ยวกับเรื่องของการควบคุมการไหลของข้อมูลทีมีทั้งแบบฮาร์ดแวร์และแบบซอฟต์แวร์ ซึ่งคำสั่ง AT&K จะทำให้คุณสามารถเลือกใช้การควบคุมการไหลชนิดต่างๆ ได้ โดยที่มีรูปแบบของคำสั่งดังตารางที่ 4.5

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตารางที่ 4.4 คำสั่ง AT&D

คำสั่ง	รูปแบบ
AT&D0	โมเด็มไม่รับรู้สัญญาณ DRT
AT&D1	โมเด็มรับรู้สัญญาณ DRT และจะไม่ตอบรับถ้าสัญญาณอยู่ในสถานะ "off" ถ้า DRT เปลี่ยนจาก "on" เป็น "off" ขณะออนไลน์โมเด็มจะยังคงยกหูโทรศัพท์ไว้ แต่จะเข้าสู่สภาวะคำสั่ง ซึ่งสามารถกลับสู่สถานะออนไลน์ได้ โดยใช้คำสั่ง ATO
AT&D2	โมเด็มรับรู้สัญญาณ DRT และจะไม่ตอบรับถ้าสัญญาณอยู่ในสถานะ "off" ถ้า DRT เปลี่ยนจาก "on" เป็น "off" ขณะที่ออนไลน์โมเด็มจะหยุดการติดต่อ และกลับไปสู่สภาวะคำสั่ง
AT&D3	โมเด็มรับรู้สัญญาณ DTR จะไม่ตอบรับถ้า DTR อยู่ในสถานะ "off" ถ้า DTR เปลี่ยนจาก "on" เป็น "off" ขณะที่ออนไลน์ โมเด็มจะหยุดการติดต่อและรีเซตตัวเอง

ตารางที่ 4.5 คำสั่ง AT&K

คำสั่ง	รูปแบบ
AT&K0	ไม่มีการใช้การควบคุมการไหลของข้อมูล ตัวอักษร XON และ XOFF จะสามารถส่งผ่านได้ ส่วนสัญญาณ RTS มีสถานะ "ON" ตลอดเวลา และไม่รับรู้สัญญาณ CTS
A&TK3	ใช้การควบคุมการไหลของข้อมูลแบบ RTS/CTS
AT&K4	ใช้การควบคุมการไหลของข้อมูลแบบ XON/XOFF
AT&K5	ใช้การควบคุมการไหลแบบ "Transparent" วิธีนี้ไม่ค่อยใช้กันมาก และจะถูกเลือกภายใต้ซอฟต์แวร์คอนโทรลเท่านั้น

จากตารางจะสังเกตได้ว่า คำสั่ง AT&K1 และ AT&K2 ไม่ได้ถูกกำหนดเอาไว้ เนื่องจากว่า Hayes เคยกำหนดให้AT&K1 และ AT&K2 เป็นคำสั่งเดียวกันกับ AT&K3 และ AT&K4 ตามลำดับ แต่ทว่าคู่มือการใช้โมเด็มของ Hayes ในปัจจุบันก็ได้ตัด AT&K1 และ AT&K2 ออกไป ดังนั้นเพื่อความแน่ใจ ก็ควรจะใช้คำสั่ง AT&K3 และ AT&K4 จะดีกว่าส่วนโมเด็มบางยี่ห้อก็อาจจะเลี่ยงปัญหานี้ โดยการเปลี่ยนไปใช้คำสั่งแบบอื่นอย่างเช่นโมเด็มของ UDS (Motorola) รุ่น 243oi จะใช้คำสั่ง AT\Q แทน AT&K เป็นต้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

AT&W (เก็บข้อมูลการเซตไว้ใน NVRAM)

คำสั่งนี้จะช่วยให้ผู้ใช้สามารถเก็บข้อมูลการตั้งค่าต่างๆ เอาไว้ใน NVRAM ซึ่งเป็นหน่วยความจำของโมเด็มได้ โดยที่ผู้ใช้จะต้องพิมพ์คำสั่งและตั้งค่ารีจิสเตอร์ S ต่างๆ ไว้ใน Active Profile เสียก่อนจากนั้นจึงสามารถใช้คำสั่ง AT&Wn เพื่อนำข้อมูลที่อยู่ใน Active Profile ไปเก็บไว้ใน User Profile ตัวที่ n ได้ หลังจากนั้นผู้ใช้ก็สามารถที่จะใช้คำสั่ง ATZ หรือ AT&Y เพื่อเรียกเอาข้อมูลจากโพรไฟล์นั้นๆ กลับมาเก็บไว้ใน Active Profile ได้เช่นเดิมรูปแบบการใช้คำสั่ง AT&W จะมีดังต่อไปนี้

AT&W0 หรือ AT&W : ให้เก็บข้อมูลการเซตจาก Active Profile ลงใน User Profile ชุดที่ 0

AT&W1 : ให้เก็บข้อมูลการเซตจาก Active Profile ลงใน User Profile ชุดที่ 1 เพื่อความเข้าใจโปรดดู ที่คำสั่ง ATZ และ AT&Y เพิ่มเติม

AT&Y (โหลดข้อมูลการเซตจาก NVRAM เมื่อเปิดเครื่อง)

คำสั่ง AT&T นี้จะมีหน้าที่ดึงเอาข้อมูลการเซตจาก User Profile ตำแหน่งที่ n กลับมาเก็บเอาไว้ในแอกทีฟโพรไฟล์ หลังจากที่เปิดเครื่องหรือหลังจากที่ป้อนไฟฟ้าให้แก่โมเด็ม โดยมีรูปแบบการใช้งานดังนี้

AT&Y0 หรือ AT&Y : ให้เรียก User Profile ตำแหน่งที่ 0 มาเก็บเอาไว้ในแอกทีฟโพรไฟล์ หลังจากเปิดเครื่อง

AT&Y1 : ให้โหลด User Profile ตำแหน่งที่ 1 มาเก็บเอาไว้ในแอกทีฟโพรไฟล์ หลังจากเปิดเครื่อง เพื่อความเข้าใจโปรดดูที่คำสั่ง AT&W และ ATZ เพิ่มเติม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 5

แนะนำ Visual Basic

แอพพลิเคชันบน Microsoft Windows ให้ความตื่นตาตื่นใจและความง่ายต่อการใช้งานกับผู้ใช้ด้วยการติดต่อกับผู้ใช้แบบกราฟฟิก มีภาพ แสง สี ที่ชวนสนใจกว่าการใช้งานในเท็กซ์โหมดบน DOS แต่เบื้องหลังของการพัฒนาแอพพลิเคชันขึ้นมานั้น มีเพียงผู้เขียนโปรแกรมเท่านั้นที่รู้ว่ายุ่งยากและลำบากเพียงใด โดยเฉพราะอย่างยิ่งในสมัยเริ่มแรก มีเพียงภาษา C และชุด Software Devolopment Kit (SDK) ของ Microsoft เท่านั้น ผู้พัฒนาต้องมีความรู้ความเข้าใจในภาษาและการทำงานของ Windows เป็นอย่างดี ที่สำคัญคือต้องรู้และเขียนโปรแกรมด้วนภาษา C ได้อย่างคล่องแคล่ว และเข้าใจการใช้พอยท์เตอร์และหลักการของภาษาอย่างถ่องแท้ สิ่งเหล่านี้นับว่าเป็นเรื่องลำบากมากสำหรับโปรแกรมเมอร์สมัครเล่น ในการที่จะสร้างแอพพลิเคชันสำหรับ Windows ขึ้นมาสักตัว

นั้นเป็นยุคแรกของการพัฒนาโปรแกรมบน Windows ต่อมารูปการณ์ได้เปลี่ยนไปเมื่อบริษัท Borland International หรือที่เรียกย่อๆ กันว่า Borland ได้นำเสนอผลิตภัณฑ์ที่ชื่อว่า Turbo Pascal for Windows ซึ่งให้ความง่ายในการเขียนโปรแกรมบน Windows ด้วยลักษณะคำสั่งและการใช้งานเหมือน Turbo Pascal เวอร์ชั่นบน DOS ที่มีนิยมใช้กันมากมายและมีความคุ้นเคยดี ถือได้ว่าเป็นการลดความซับซ้อนในการพัฒนาแอพพลิเคชันบน Windows ลงได้ระดับหนึ่ง นอกจากนี้ยังมีซอฟต์แวร์อีกหลายตัวด้วยกัน ที่มาช่วยในการสร้างแอพพลิเคชันบน Windows เช่น โปรแกรม Toolbooks หรือ WindowsMaker ที่ทำงานในลักษณะของ program generator คือผู้ใช้ออกแบบเมนู , องค์ประกอบต่างๆ ในวินโดว์และกำหนดเงื่อนไขในการทำงาน หลังจากนั้นซอฟต์แวร์พวกนี้จะสร้างซอร์สโปรแกรมในภาษา C มาให้ เพื่อให้ผู้ใช้แก้ไขและคอมไพล์เป็นไฟล์ที่ทำงานได้ต่อไป

ทั้ง Turbo Pascal for Windows, Toolbooks และกลุ่ม program generator ต่างก็เป็นเครื่องมือที่เป็นทางเลือกอื่นนอกเหนือจากภาษา C และชุด SDK ไม่ว่าจะเป็น Turbo Pascal for Windows หรือ Toolbooks ต่างก็ปลดตัวเองออกจากพันธนาการของ SDK โดยเฉพาะ Borland ถึงกับกล่าวว่าหากใช้ Turbo Pascal หรือ Turbo C for Windows แล้วผู้ใช้ก็ไม่จำเป็นต้องมายุ่งเกี่ยวกับ SDK อีกต่อไป

แต่ทุกคนคงไม่ลืมว่าหนึ่งในภาษาแรกๆ ที่ผู้สนใจการเขียนโปรแกรมจะเรียนกันก็คือภาษา BASIC ซึ่งมีจุดเด่นตรงความง่ายสดวกต่อการเขียนและเรียนรู้ และทาง Microsolf เองคงจะรู้ถึงจุดดีนี้เป็นอย่างดี อีกทั้งยังรู้ว่าแนวโน้มของผู้ใช้จะนิยมพัฒนาโปรแกรมขึ้นมาใช้เองมีมาก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ขึ้น หรือนักพัฒนามืออาชีพก็คงต้องการความสะดวกและง่ายในการพัฒนาแอพพลิเคชันบน Windows มากขึ้นนอกจากนี้ตัวของ Bill Gates ประธานบริษัท Microsoft เองก็มีจุดกำเนิดในวงการคอมพิวเตอร์ด้วยตัวแปลภาษา Basic จึงไม่แปลกอะไรที่ Microsoft จะมีตัวแปล Basic สำหรับการพัฒนาแอพพลิเคชันบน Windows โดยให้ชื่อว่า Visual Basic ที่ง่ายต่อการใช้งาน ในขณะเดียวกันก็ง่ายต่อการเขียนโปรแกรมตามแบบฉบับของ Basic ด้วย และที่สำคัญคือไม่ต้องใช้ชุด SDK อีก

5.6 ความเป็นมา

Visual Basic เวอร์ชั่นแรกที่ออกมานั้นเป็น เวอร์ชั่นบน Windows ออกสู่สายตาผู้ใช้เมื่อปี 1991 ในเวอร์ชั่น 1.0 นั้นยังไม่มีความสามารถอะไรมากนัก เป็นเพียงเครื่องมืออย่างง่ายสำหรับการสร้างแอพพลิเคชันบน Windows มากกว่า องค์ประกอบหรือ Object ที่ใช้ได้ก็มีแค่องค์ประกอบพื้นฐานของ Windows เช่น Text Box , List Box เท่านั้น แม้แต่การใช้งานแบบ Multiple Document Interface (MDI) คือมีวินโดว์ ย่อยๆในวินโดว์หลักก็ไม่สามารถทำได้ แต่อย่างไรก็ตาม นับว่า Visual Basic 1.0 ประสบความสำเร็จและได้รับการต้อนรับจากผู้ใช้เป็นอย่างดีในแง่ของการใช้ และหลักการโปรแกรมแบบ Visual จน Microsoft ต้องทำ Visual Basic for DOS ออกมาด้วย จนถึงขณะนี้ทั้ง Visual Basic for Windows และ DOS ออกมาถึงเวอร์ชั่น 3.0 แล้ว ในแต่ละเวอร์ชั่นมีการพัฒนาปรับปรุงความสามารถเรื่อยๆ เช่นสนับสนุน MDI , OLE และมี Object มากขึ้นปรับปรุง Debugger จนใน เวอร์ชั่น 3.0 ได้เพิ่มความสามารถด้านการทำงานกับฐานข้อมูล,สนับสนุน OLE 2.0 และอื่นๆอีกหลายประการ

สำหรับในรายงานฉบับนี้ จะกล่าวถึงเฉพาะ Visual Basic for Windows เท่านั้น ยังไรก็ตามโดยหลักการและการใช้งานทั่วไปแล้ว จะเหมือนกับ Visual Basic for Dos เช่นกัน

ในอนาคตคาดว่า Visual Basic จะมีบทบาทต่อผู้ใช้ Windows มากขึ้นเพราะต่อไปแอพพลิเคชัน ของ Microsoft ทุกตัวบน Windows จะมีภาษามาโครของ Word for Windows , Excel หรือ โปรแกรมอื่นของ Microsoft แต่ละตัวก็จะมีการเรียนรู้ VBA เพียงอย่างเดียวทั้งนี้เนื่องจากความง่ายของภาษา ประสิทธิภาพ และความนิยมของผู้ใช้ร่วมทั้งเป็นการสร้างมาตรฐานของภาษาสำหรับแอพพลิเคชัน บน Windows อีกด้วยในลักษณะเดียวกันกับภาษา REXX ของ OS/2 นอกจากนี้ยังมีข่าวการพัฒนา Visual Basic บน แพลตฟอร์มอื่นออกมาเป็นระยะๆด้วย ฉนั้นถือว่าการเรียนรู้ Visual Basic จะทำให้เราก้าวไปสู่การใช้แอพพลิเคชัน สำเร็จรูปอย่างมีประสิทธิภาพมากขึ้น นอกเหนือจากการพัฒนาแอพพลิเคชัน ขึ้นมาใช้เองอย่างไรก็ตาม VBA จะเป็นเพียงการดึงบางคำสั่ง และฟังก์ชั่นของ Visual Basic ขึ้นมาใช้เท่านั้น

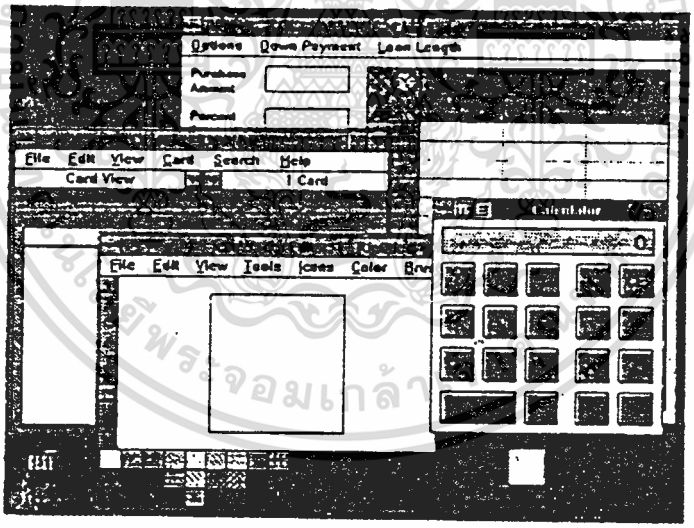
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

5.2 ภาพรวมของ Visual Basic

Visual Basic มีสภาพแวดล้อมสำหรับการพัฒนาโปรแกรมบนวินโดว์ ประกอบด้วยเครื่องมือต่างๆ ครบถ้วนไม่ว่าจะเป็น ส่วนของการออกแบบ User Interface , ส่วนออกแบบเมนู (Menu designer) , การสร้างรายงาน (Report writer) , อิดิเตอร์สำหรับป้อนโปรแกรม และ Debugger เพื่อการตรวจหาข้อผิดพลาดในโปรแกรม องค์ประกอบเหล่านี้นับว่าเอื้ออำนวยต่อการทำงานของโปรแกรมเมอร์เป็นอย่างมาก

ในด้านของตัวภาษา Visual Basic ได้นำไวยากรณ์ของ BASICA และ GW-BASIC มาใช้โดยสนับสนุนความสามารถเดิมเกือบทั้งหมดนอกจากนี้ยังได้เพิ่มการโปรแกรมแบบมีโครงสร้างของ QuickBasic ซึ่งคล้ายกับภาษาที่มีโครงสร้างเช่น Pascal หรือ C เข้าไปด้วย

นอกจานี้ยังมีการเพิ่มคำสั่งและฟังก์ชั่นเกี่ยวกับ Object และการเรียกฟังก์ชั่นของระบบปฏิบัติการ (API) เพื่อให้การทำงานกว้างขึ้นรวมทั้งสนับสนุนความสามารถของระบบเช่น OLE , DDE และการใช้งานคลิปบอร์ดเป็นต้น



รูปที่ 5.1 ตัวอย่างแอพพลิเคชันที่พัฒนาด้วย Visual Basic

ด้วยความสามารถของตัวภาษา Visual Basic ผู้ใช้สามารถนำไปสร้างแอพพลิเคชันได้หลายประเภทไม่ว่าจะเป็นโปรแกรมวาดภาพ , การคำนวณทางการเงินหรือแม้แต่โปรแกรม Cardfile ซึ่งเป็นโปรแกรมมาตรฐาน ใน Windows โดยไม่ต้องใช้ชุด SDK เลย

Visual Basic แต่ละเวอร์ชั่นจะมี 2 Edition คือ Standard และ Professional Edition ซึ่งข้อแตกต่างคือในชุด Professional นั้นจะมี Custom Control (Osbjectที่สามารถนำมาใช้ในฟอร์ม)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

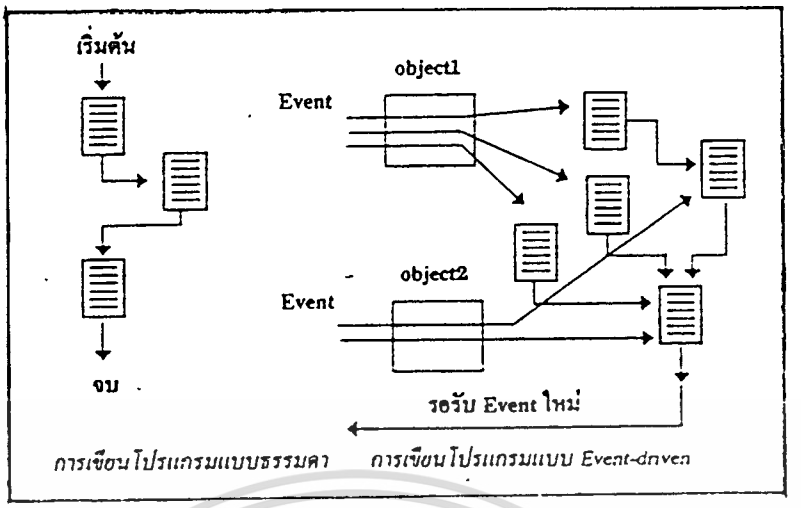
มากกว่าและจะมีเครื่องมืออื่นๆ เช่น Help Compiler สำหรับการสร้างข้อความอธิบายการใช้ Setup Kit เพื่อทำส่วนของการติดตั้งแอพพลิเคชั่น (Instoller) และ Report writer มาให้พร้อมกับข้อมูลเพิ่มเติมอื่นสำหรับผู้พัฒนา แต่ในด้วนความสามารถของภาษาแล้วจะเหมือนกันทั้ง 2 Edition

5.3 หลักการโปรแกรมเชิงภาพของ Visual Basic

ใน Visual Basic นั้นการพัฒนาและการเขียนโปรแกรมจะเป็นไปในอีกรูปแบบหนึ่ง กล่าวคือ ในหารเขียนโปรแกรมแบบเดิมนั้นเราจะต้องออกแบบหน้าจอ ระบุตำแหน่งการแสดงผล คิดหาขั้นตอนการทำงาน และอื่นๆ จากนั้นจึงทำการเขียนโปรแกรม โปรแกรมที่ได้จะอธิบายและสั่งงานคอมพิวเตอร์ไปเป็นลำดับไป แต่ใน Visual Basic จะใช้หลักการของภาพ และการมองเห็น โดยเริ่มจากการออกแบบ วินโดว์ย่อย หรือที่ใน Visual Basic เรียกว่าฟอร์ม ในฟอร์มจะประกอบด้วยสิ่งต่างๆที่เราทำงานด้วย หรือเรียกว่าเป็น Object เช่นข้อความ , ช่องรับข้อความ , Scroll Bar หรือ ปุ่ม (Buttom) เมื่อกำหนดสิ่งเหล่านี้ครบตามความต้องการแล้วจึงระบุว่าองค์ประกอบแต่ละอย่างจะทำงานอย่างไร โดยเขียนโปรแกรมย่อยๆปะเข้าไปกับ Object เหล่านี้ ที่ต้องทำแบบนี้เพราะว่าการทำงานใน Windows เป็นแบบที่เรียกว่า Event-Driven คือขึ้นอยู่กับเหตุการณ์ (Event) การเขียนโปรแกรมแบบเดิมคือสั่งงานตามลำดับจะยุ่งยากมาก หรือบางกรณีอาจทำไม่ได้เลย เพราะอย่าลืมว่าในขณะใดขณะหนึ่งนั้น ในระบบไม่ใช่จะมีเพียงแอพพลิเคชั่นของเราเท่านั้นที่รันอยู่ Windows จะต้องนัดการกับทุก แอพพลิเคชั่นที่ทำงานในขณะนั้นทั้งหมดไปพร้อมๆกัน (เนื่องจากเป็นระบบแบบ Multitasking หรือรันหลายงานได้ในเวลาเดียวกัน) ในขณะที่โปรแกรมแสดงหน้าจอสำหรับรับอินพุท ผู้ใช้อาจพิมพ์ข้อมูลเข้าไปแต่โดยดี หรืออาจอยู่ไม่สุขใช้เมาส์เลื่อยไปคลิ๊กตรงนั้นตรงนี้ได้โดยอิสระ ทำให้ยากที่จะเขียนโปรแกรมธรรมดาให้คอยดักเส้นทางการทำงานในการรับอินพุทว่าจะเกิดอะไรขึ้นตรงไหนได้ จึงต้องใช้รูปแบบการโปรแกรมในลักษณะ Event Driven ดังกล่าวซึ่ง Object แต่ละตัวจะมีเหตุการณ์เกิดขึ้นกับมันได้หลายอย่าง ถ้าเราสนใจเฉพาะเหตุการณ์ใดก็เขียนโปรแกรมสั่งงานให้คอยดักหรือทำงานเฉพาะเหตุการณ์นั้นๆได้ เช่น ถ้าสิ่งที่เราสนใจเป็นปุ่มควบคุมและเราต้องการให้ทำงานเมื่อคลิ๊ก หรือ ดับเบิ้ลคลิ๊ก ก็ระบุว่า หากมีการคลิ๊กที่ปุ่มควบคุมนี้โหรแกรมจะต้องทำอน่างไร หรือถ้ามีการ ดับเบิ้ลคลิ๊กจะต้องทำอย่างไร ส่วนเหตุการณ์อื่นที่ไม่ได้ระบุไว้จะไม่มีผลต่อ Object นั้น

จากแนวทางนี้จะเห็นว่ามีลักษณะเป็นธรรมชาติมาก คือกำหนดหน้าตาของจอภาพ หรือส่วนติดต่อผู้ใช้แล้วระบุว่าถ้าเกิดเหตุการณ์อย่างนี้กับสิ่งนี้จะต้องทำอย่างไรซึ่งโปรแกรมที่เราจะเขียนกันนี้ก็คือส่วนที่จะบอกว่า “จะต้องทำอย่างไร” นั้นเอง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 5.2 การเขียนโปรแกรมแบบธรรมดากับแบบ Event Driven

นอกจาก Object จะมีการตอบสนองต่อเหตุการณ์ต่างๆที่กำหนดแล้วยังมีอีกเรื่องหนึ่งที่จะขอกล่าวคือทุก Object จะมีลักษณะหรือคุณสมบัติ (Property) ของตัวเองเช่นช่องรับข้อความ (Text Box) จะมีชื่อ , ความกว้าง , ความสูง , สี โดยเราสามารถอ้างถึง หรือเปลี่ยนคุณสมบัติเหล่านี้ได้ขณะที่โปรแกรมทำงานอยู่เป็นต้นว่าหากไม่มีการป้อนข้อมูลจะให้แสดงด้วยสีหนึ่งหรืออาจไม่แสดงบนจอภาพเลย

ในการที่จะกระทำสิ่งใดสิ่งหนึ่งกับ Object นั้น จะมีสิ่งที่เรียกว่า Methodซึ่งเปรียบเปรียบเหมือนเป็นกระบวนการทำงานของ Object ซึ่ง Object แต่ละแบบก็อาจจะมี Method ที่แตกต่างกันออกไป เช่น ถ้าต้องการสั่งให้เลื่อนตำแหน่งของข้อความ (Label) ก็จะมีกระบวนการ หรือ Method ชื่อ Move ของ Label เพื่อทำงานนี้โดยสั่งว่า Label.Move = ตำแหน่งที่จะย้ายไปโดยการสั่งพิมพ์ก็มี Method ชื่อ Print เป็นต้น ถ้าพูดไปแล้ว Method ก็คล้ายๆกับ คำสั่งที่ใช้กับ Object แต่ละชนิดนั้นเอง

โดยสรุปแล้วรูปแบบของหลักการของ Visual Baic ก็คือเริ่มจากการออกแบบจอภาพและเขียนโปรแกรมสำหรับแต่ละ Event ปะเข้าไปยัง Object ต่างๆให้ทำงานตามเหตุการณ์ที่เกิดขึ้น โดย Object จะมีคุณสมบัติเฉพาะที่สามารถเปลี่ยนแปลงได้ของตัวเอง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 6
หลักการเขียนโปรแกรม

การเขียนโปรแกรมจะแยกออกเป็น 2 โปรแกรม

- โปรแกรมบันทึกข้อมูล < RECORD.FRM >
- โปรแกรมบันทึกเสียงพูด < VOICE.FRM >

โปรแกรมบันทึกข้อมูล

ส่วนของการจัดการฐานข้อมูลจะใช้ ส่วนควบคุมที่มีชื่อว่า Data Control ที่มีอยู่ในตัว Visual Basic 3.0 โดยมีชื่อไฟล์ที่ทำการเก็บข้อมูลว่า DATA.MDB และมีชื่อเรคคอร์ดว่า Infomation และเนื่องจากใช้วิธีการเขียนแบบ ตามเหตุการณ์ที่เกิดขึ้น (event driven) ดังนั้นเราจึงสามารถแยกโปรแกรมออกเป็น 4 ส่วน คือ

- ส่วนเพิ่มข้อมูล

เมื่อมีการกดปุ่มเพิ่มข้อมูล โปรแกรมจะทำการเปลี่ยนหน้าจอใหม่เพื่อทำการรอรับ ข้อมูล ถ้าทำการกดปุ่มตกลง จะทำการ Update ข้อมูลลงในฐานข้อมูลเพิ่ม แต่ถ้ากดปุ่มยก เลิก จะทำการ Refresh ข้อมูลใหม่ และกลับไปที่หน้าจอหลัก

- ส่วนลบข้อมูล

เมื่อมีการกดปุ่มลบข้อมูล โปรแกรมจะทำการลบข้อมูลที่อยู่ในขณะนั้น ถ้าต้องการลบข้อมูลที่ต้องการจะต้องทำการค้นหาข้อมูลนั้นก่อน

- ส่วนค้นหาข้อมูล

เมื่อมีการกดปุ่มค้นหาข้อมูล โปรแกรมจะให้กดหมายเลขที่ต้องการค้นหา ถ้ากดปุ่ม OK โปรแกรมจะทำการค้นหาข้อมูลโดยเริ่มจากข้อมูลแรก และถ้าหาไม่พบก็จะฟ้องว่าไม่มีหมายเลขเรียก xxxxxx อยู่ในเรคคอร์ด แต่ถ้ากดปุ่ม Cancel จะทำกลับสู่โปรแกรม หลัก

- ส่วนออกจากโปรแกรม

เมื่อมีการกดปุ่มออกโปรแกรม จะทำการออกจากโปรแกรม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โปรแกรมบันทึกเสียง

โปรแกรมนี้ยังไม่สมบูรณ์ เนื่องจากยังไม่สามารถหาข้อมูลในการควบคุมการพูด และบันทึกเสียง ดังนั้นจึงใช้การจำลองสถานการณ์จริง โดยทำการควบคุมโปรแกรม Quick Link โดยใช้ Function บน Windows มาควบคุม คือ

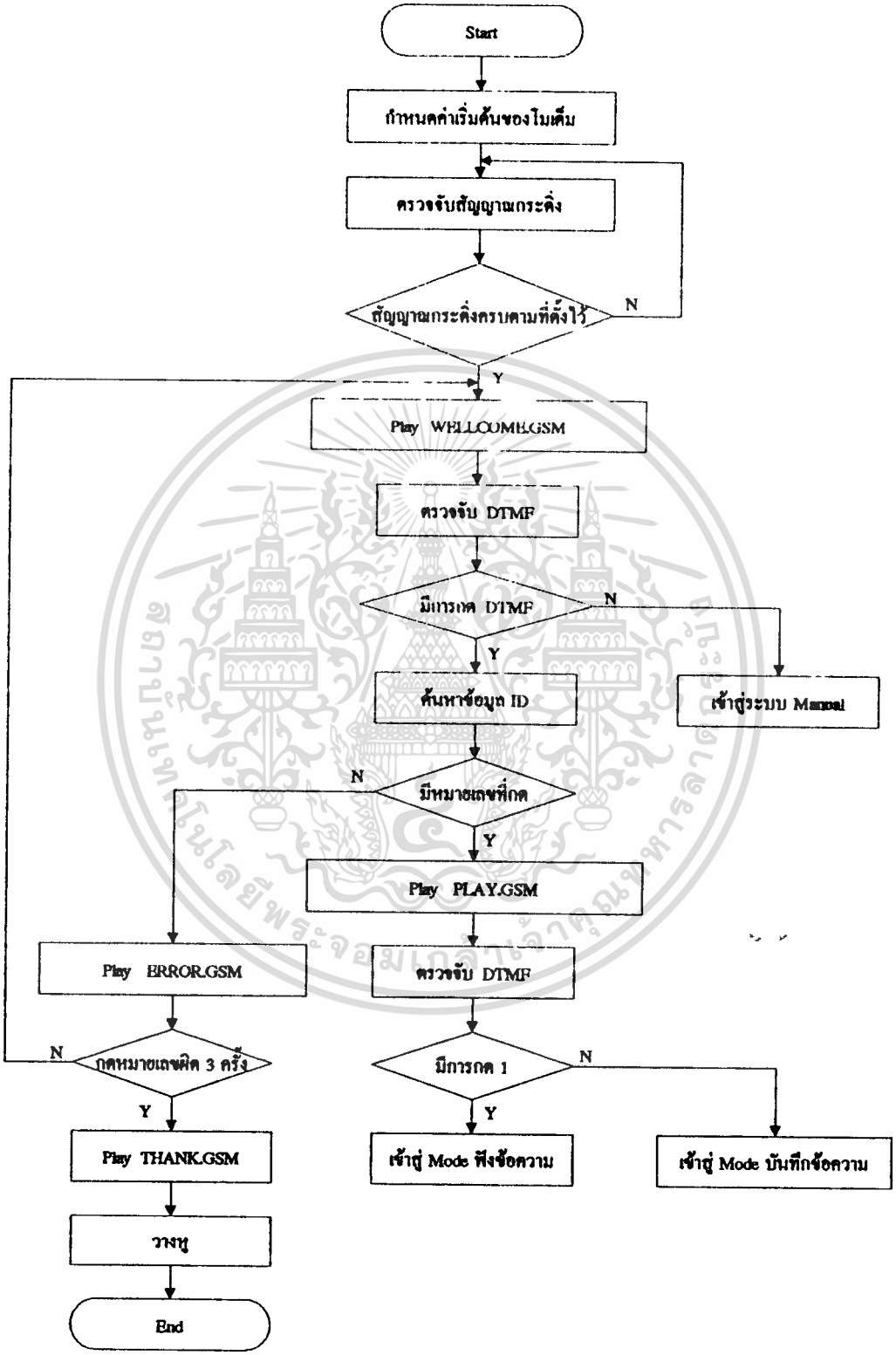
- Function GetFocus ใช้ในการเก็บค่า Handle ของโปรแกรม ที่กำลังถูก Focus อยู่
- Function SetFocusAPI ใช้ในการ Set ให้โปรแกรมที่มีค่า Handle นั้นอยู่ ถูก Focus

เริ่มต้นจากการลดให้โปรแกรม VB และ โปรแกรม Manager ให้เป็น Minimize เพื่อการเก็บและคืนค่า Handle จะไม่ยุ่งยาก ซึ่งอาจจะเกิดการผิดพลาดได้ จากนั้นทำการ Load โปรแกรม MCMAIL.EXE และทำการเก็บค่า Handle ไว้ในชื่อ MCMail_Handle จากนั้นทำการควบคุมคล้ายการกดคีย์ โดยใช้คำสั่ง SendKeys เพื่อทำให้โปรแกรมทำงานอย่างอัตโนมัติ

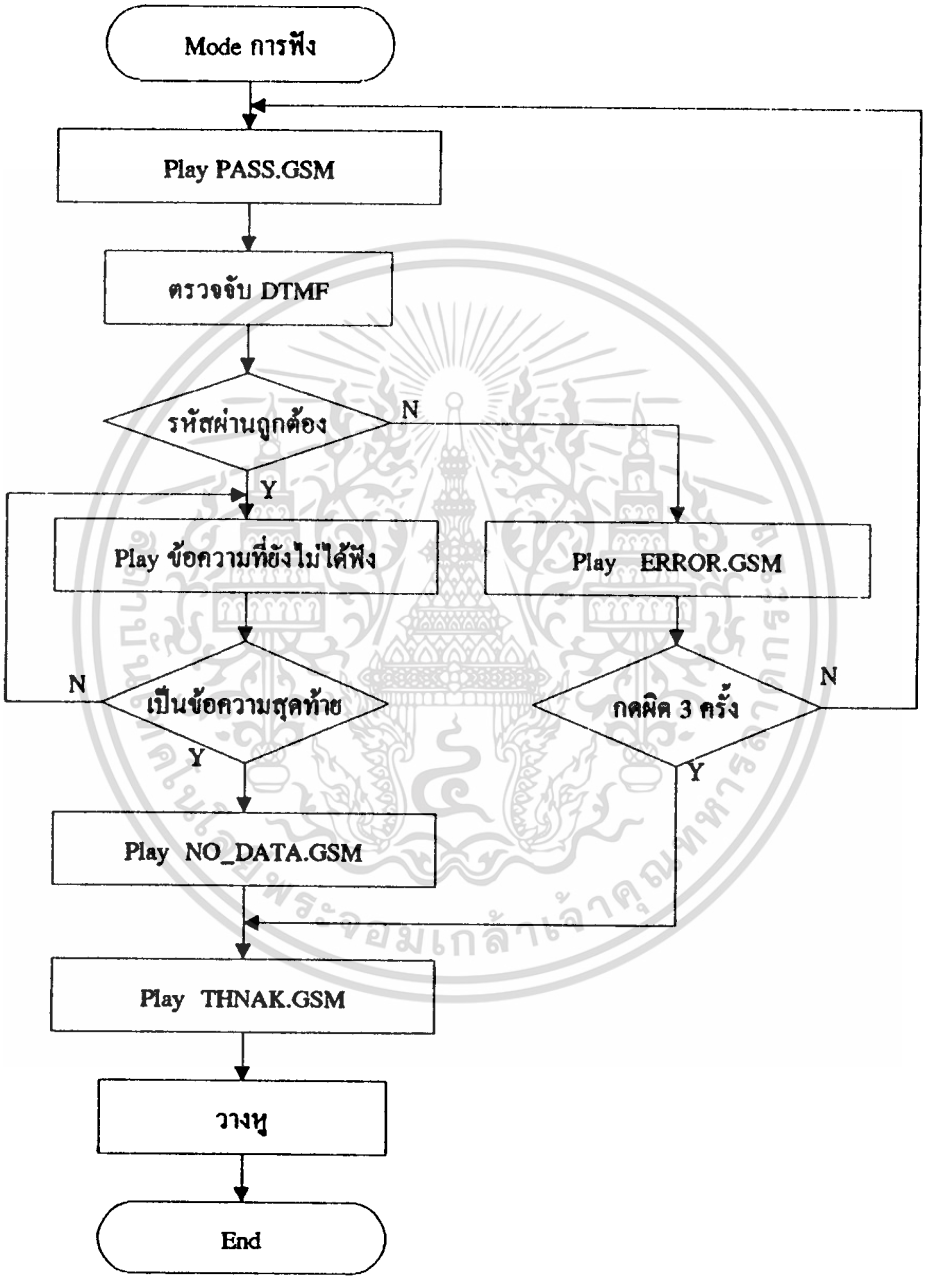
<u>หมายเหตุ</u> เนื่องจากโปรแกรมจะทำการ Play จาก GREETING.GSM และจะทำการบันทึกลง MAIL0001.GSM ดังนั้นถ้าต้องการ Play หรือ Record จะต้องทำการเปลี่ยนชื่อให้เป็นชื่อดังกล่าวเสียก่อน และเนื่องจากการใช้คำสั่ง SendKeys จะทำให้โปรแกรมรับการกดคีย์เร็วเกินไปจึงต้องทำการหน่วงเวลาไปบ้างเพื่อให้โปรแกรมนั้นสามารถรับคีย์ได้ทัน

อย่างไรก็ตาม ส่วนของ FLOW CHART ยังคงมีลักษณะเดียวกัน ดังนั้นในการพัฒนาต่อจึงสามารถพัฒนาได้ตาม FLOW CHART ที่ให้มาในหน้าถัดไป

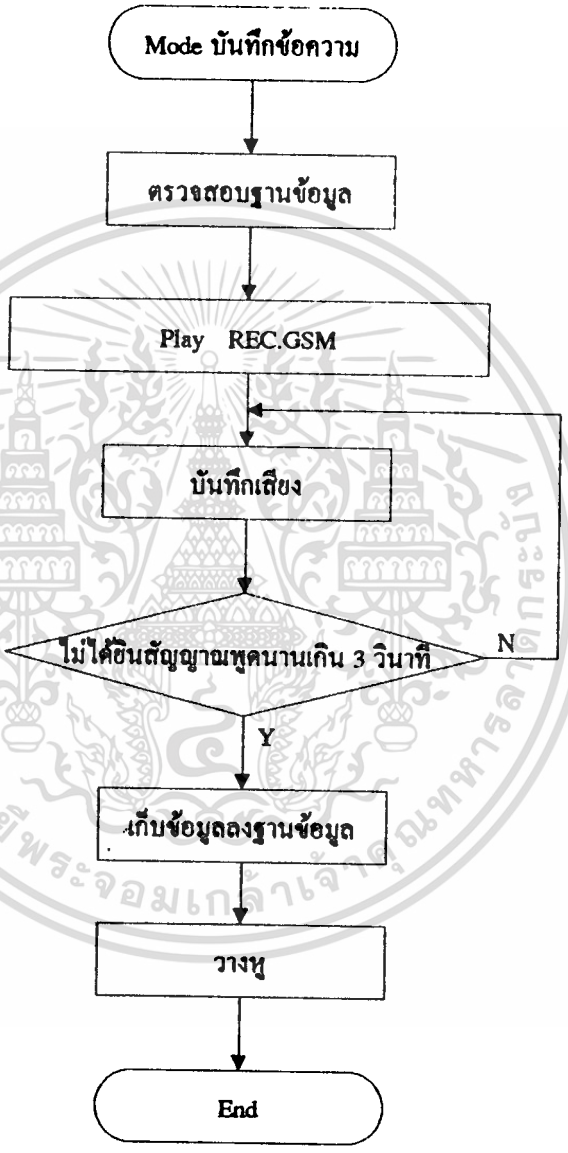
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บรรณานุกรม

1. จิรพัฒน์ จันทร์เจิดศักดิ์, วีระ นพนิราพาธ, **เขียนโปรแกรมบน Microsoft Windows**, บริษัท ซีเอ็ดยูเคชั่น จำกัด, 2521
2. จิรศักดิ์ เหลืองอุไร, **คัมภีร์การใช้งาน การสื่อสารอนุกรมบน PC**, บริษัท ซีเอ็ดยูเคชั่น จำกัด, 2538
3. ราบินเดอร์ ศรีกิจจาภรณ์, **คู่มือการใช้งาน Visual Basic สำหรับวินโดวส์**, บริษัท ซีเอ็ดยูเคชั่น จำกัด, 2538
4. วรวิทย์ ตันติโภคิน, นภดล ชายธีระเดช, **การเขียนโปรแกรมบนวินโดวส์ด้วย Microsoft Visual Basic ภาคปฏิบัติ**, บริษัท ซีเอ็ดยูเคชั่น จำกัด, 2537
5. สุรชัย เพิ่มสินทวี, **นำทางสู่การใช้งาน การสื่อสารข้อมูลด้วยระบบ MODEM**, บริษัท ซีเอ็ดยูเคชั่น จำกัด, 2537

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

RECORD.FRM

VERSION 2.00

Begin Form Form1

BorderStyle	=	3 'Fixed Double
Caption	=	"บันทึกข้อมูล"
ClientHeight	=	4305
ClientLeft	=	1485
ClientTop	=	1710
ClientWidth	=	7485
Height	=	4710
Left	=	1425
LinkTopic	=	"Form1"
ScaleHeight	=	4305
ScaleWidth	=	7485
Top	=	1365
Width	=	7605

Begin Data Data1

Caption	=	"พลิกหน้า"
Connect	=	""
DatabaseName	=	"D:\VB\DATA\DATA.MDB"
Exclusive	=	0 'False
Height	=	270
Left	=	2160
Options	=	0
ReadOnly	=	0 'False
RecordSource	=	"Infomation"
Top	=	3000
Width	=	3255

End

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
Begin CommandButton Command4
        Caption     =     "ออกโปรแกรม"
        Height      =     495
        Left        =     5880
        TabIndex    =     17
        Top         =     3600
        Width       =     1215
End

Begin CommandButton Command3
        Caption     =     "ค้นหาข้อมูล"
        Height      =     495
        Left        =     4080
        TabIndex    =     16
        Top         =     3600
        Width       =     1215
End

Begin CommandButton Command2
        Caption     =     "ลบข้อมูล"
        Height      =     495
        Left        =     2280
        TabIndex    =     15
        Top         =     3600
        Width       =     1215
End
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
Begin CommandButton Command1
        Caption     =     "เพิ่มข้อมูล"
        Height      =     495
        Left        =     480
        TabIndex    =     14
        Top         =     3600
        Width       =     1215
End

Begin Label Label14
        Caption     =     " "
        DataField   =     "M_Code"
        DataSource  =     "Data1"
        Height      =     375
        Left        =     4920
        TabIndex    =     13
        Top         =     2520
        Width       =     1935
End

Begin Label Label13
        Caption     =     " "
        DataField   =     "Tel"
        DataSource  =     "Data1"
        Height      =     375
        Left        =     1320
        TabIndex    =     12
        Top         =     2520
        Width       =     2175
End
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
Begin Label Label12
        Caption     =     " "
        DataField   =     "Post"
        DataSource  =     "Data1"
        Height      =     255
        Left        =     1680
        TabIndex    =     11
        Top         =     2160
        Width       =     1215
End

Begin Label Label11
        Caption     =     " "
        DataField   =     "Address"
        DataSource  =     "Data1"
        Height      =     615
        Left        =     1200
        TabIndex    =     10
        Top         =     1560
        Width       =     5775
End

Begin Label Label10
        Caption     =     " "
        DataField   =     "Name"
        DataSource  =     "Data1"
        Height      =     255
        Left        =     1800
        TabIndex    =     9
        Top         =     960
        Width       =     5175
End
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
Begin Label Label9
        Caption     =     " "
        DataField   =     "Password"
        DataSource  =     "Data1"
        Height      =     255
        Left        =     5280
        TabIndex    =     8
        Top         =     360
        Width       =     1575
End

Begin Label Label8
        Caption     =     " "
        DataField   =     "ID"
        DataSource  =     "Data1"
        Height      =     255
        Left        =     1680
        TabIndex    =     7
        Top         =     360
        Width       =     1815
End

Begin Label Label7
        Caption     =     "รหัสเครื่อง"
        Height      =     375
        Left        =     3960
        TabIndex    =     6
        Top         =     2520
        Width       =     975
End
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
Begin Label Label6
        Caption     =     "โทรศัพท์"
        Height      =     375
        Left        =     480
        TabIndex    =     5
        Top         =     2520
        Width       =     735
End

Begin Label Label5
        Caption     =     "รหัสไปรษณีย์"
        Height      =     255
        Left        =     480
        TabIndex    =     4
        Top         =     2160
        Width       =     1095
End

Begin Label Label4
        Caption     =     "ที่อยู่"
        Height      =     255
        Left        =     480
        TabIndex    =     3
        Top         =     1560
        Width       =     495
End
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
Begin Label Label3
        Caption     =     "ชื่อ - นามสกุล"
        Height      =     255
        Left        =     480
        TabIndex    =     2
        Top         =     960
        Width       =     1095
End

Begin Label Label2
        Caption     =     "รหัสผ่าน"
        Height      =     255
        Left        =     4320
        TabIndex    =     1
        Top         =     360
        Width       =     735
End

Begin Label Label1
        Caption     =     "หมายเลขเรียก"
        Height      =     255
        Left        =     480
        TabIndex    =     0
        Top         =     360
        Width       =     1095
End

End
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
Sub Command1_Click ()          ' เพิ่มข้อมูล
    Unload Form1
    Form2.Show
    Form2.Data1.Recordset.AddNew          ' เพิ่ม Record ใหม่
End Sub

Sub Command2_Click ()          ' ลบข้อมูล
    Dim Msg, Title, ID As String
    Dim Response, Count As Integer
    If Not Data1.Recordset.EOF Then          'ตรวจว่าเป็นข้อมูลสุดท้ายหรือไม่
        If Not Data1.Recordset.BOF Then          'ตรวจว่าเป็นข้อมูลแรกหรือไม่
            ID = Label8.Caption
            Data1.Recordset.MoveLast
            Count = Data1.Recordset.RecordCount          ' นับจำนวน Record
        ' ค้นหาตาม ID ตั้งแต่ Record แรก
            Data1.Recordset.FindFirst "ID = " + Chr$(39) + ID + Chr$(39)
        ' แสดงข้อความ ตอบตกลง หรือ ยกเลิก
            Msg =  "คุณต้องการลบข้อมูลของหมายเลขเรียก [ " + ID + " ] หรือไม่?"
            Title  = "ลบข้อมูล"
            Response = MsgBox(Msg, 1, Title)
        ' ตรวจสอบการตอบตกลง
            If  Response = 1 Then
                Data1.Recordset.Delete
                Data1.Recordset.MoveNext
                If Data1.Recordset.EOF Then
                    If Count = 1 Then
                        Data1.Recordset.AddNew
                    Else Data1.Recordset.MoveLast
                    End If
                End If
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                End If
            End If
        Else
            MsgBox "ขณะนี้ไม่มีข้อมูลอยู่ในเรคคอร์ด"
        End If
End Sub

Sub Command3_Click ()                    ' ค้นหาข้อมูล
        Dim Criteria As String
        Criteria = InputBox("กรุณาป้อนหมายเลขเรียกที่ต้องการค้นหา")
        Data1.Recordset.FindFirst "ID=" + Chr$(39) + Criteria + Chr$(39)
' ไม่มีข้อมูลใน Record
        If Data1.Recordset.NoMatch Then
            MsgBox "ไม่มีหมายเลขเรียก [ " + Criteria + " ]อยู่ในเรคคอร์ด"
        End If
End Sub

Sub Command4_Click ()                    ' ออกจากโปรแกรม
        End
End Sub
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ADD.FRM

VERSION 2.00

Begin Form Form2

BorderStyle	=	3 'Fixed Double
Caption	=	"เพิ่มข้อมูล"
ClientHeight	=	4320
ClientLeft	=	1215
ClientTop	=	2040
ClientWidth	=	7650
Height	=	4725
Left	=	1155
LinkTopic	=	"Form2"
ScaleHeight	=	4320
ScaleWidth	=	7650
Top	=	1695
Width	=	7770

Begin TextBox Text10

DataField	=	"Point"
DataSource	=	"Data1"
Height	=	375
Left	=	1200
TabIndex	=	18
Text	=	" "
Top	=	3120
Visible	=	0 'False
Width	=	735

End

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
Begin TextBox Text9
        DataField       =       "Tail"
        DataSource      =       "Data1"
        Height          =       495
        Left            =       360
        TabIndex        =       17
        Text            =       " "
        Top             =       3600
        Visible         =       0   'False
        Width           =       615
End

Begin TextBox Text8
        DataField       =       "Head"
        DataSource      =       "Data1"
        Height          =       375
        Left            =       360
        TabIndex        =       16
        Text            =       " "
        Top             =       3120
        Visible         =       0   'False
        Width           =       615
End

Begin CommandButton Command2
        Caption         =       "ยกเลิก"
        Height          =       495
        Left            =       3960
        TabIndex        =       8
        Top             =       3480
        Width           =       1215
End
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
Begin CommandButton Command1
        Caption     =     "ตกลง"
        Height      =     495
        Left        =     2280
        TabIndex    =     7
        Top         =     3480
        Width       =     1215
End

Begin Data Data1
        Caption     =     "Data1"
        Connect     =     ""
        DatabaseName =    "D:\VB\DATA\DATA.MDB"
        Exclusive   =     0   'False
        Height      =     270
        Left        =     6000
        Options     =     0
        ReadOnly    =     0   'False
        RecordSource =    "Infomation"
        Top         =     3240
        Visible     =     0   'False
        Width       =     1215
End

Begin TextBox Text7
        DataField   =     "M_Code"
        DataSource  =     "Data1"
        Height      =     375
        Left        =     6120
        MaxLength   =     15
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        TabIndex    =    6
        Text        =    " "
        Top         =    2640
        Width       =    1095
End

Begin TextBox Text6
        DataField   =    "Tel"
        DataSource  =    "Data1"
        Height      =    375
        Left        =    3840
        MaxLength   =    15
        TabIndex    =    5
        Text        =    " "
        Top         =    2640
        Width       =    1215
End

Begin TextBox Text5
        DataField   =    "Post"
        DataSource  =    "Data1"
        Height      =    375
        Left        =    1560
        MaxLength   =    10
        TabIndex    =    4
        Text        =    " "
        Top         =    2640
        Width       =    1335
End
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
Begin TextBox Text4
        DataField       =       "Address"
        DataSource      =       "Data1"
        Height          =       735
        Left            =       960
        MaxLength       =       100
        MultiLine       =       -1  'True
        TabIndex        =       3
        Text            =       " "
        Top             =       1560
        Width           =       6255
End

Begin TextBox Text3
        DataField       =       "Name"
        DataSource      =       "Data1"
        Height          =       375
        Left            =       1560
        MaxLength       =       40
        TabIndex        =       2
        Text            =       " "
        Top             =       960
        Width           =       4935
End

Begin TextBox Text2
        DataField       =       "Password"
        DataSource      =       "Data1"
        Height          =       375
        Left            =       5040
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        MaxLength   =   10
        TabIndex    =   1
        Text        =   " "
        Top         =   360
        Width       =   1455
End

Begin TextBox Text1
        DataField   =   "ID"
        DataSource  =   "Data1"
        Height      =   375
        Left        =   1560
        MaxLength   =   10
        TabIndex    =   0
        Text        =   " "
        Top         =   360
        Width       =   2295
End

Begin Label Label7
        Caption     =   "รหัสเครื่อง"
        Height      =   375
        Left        =   5160
        TabIndex    =   15
        Top         =   2640
        Width       =   975
End
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
Begin Label Label6
        Caption     =     "โทรศัพท์"
        Height      =     375
        Left        =     3000
        TabIndex    =     14
        Top         =     2640
        Width       =     735
End

Begin Label Label5
        Caption     =     "รหัสไปรษณีย์"
        Height      =     255
        Left        =     360
        TabIndex    =     13
        Top         =     2640
        Width       =     1095
End

Begin Label Label4
        Caption     =     "ที่อยู่"
        Height      =     255
        Left        =     360
        TabIndex    =     12
        Top         =     1560
        Width       =     495
End
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
Begin Label Label3
        Caption     =     "ชื่อ - นามสกุล"
        Height      =     255
        Left        =     360
        TabIndex    =     11
        Top         =     960
        Width       =     1095
End

Begin Label Label2
        Caption     =     "รหัสผ่าน"
        Height      =     255
        Left        =     4200
        TabIndex    =     10
        Top         =     360
        Width       =     735
End

Begin Label Label1
        Caption     =     "หมายเลขเรียก"
        Height      =     255
        Left        =     360
        TabIndex    =     9
        Top         =     360
        Width       =     1095
End

End
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
Sub Command1_Click ()                    ' เพิ่มข้อมูล
        On Error GoTo Rec_Err:           ' ถ้ามีการ Eror ไปที่ Label ชื่อ Rec_Err
        Text8.Text = 0
        Text9.Text = 0
        Text10.Text = 0
        Form2.Data1.Recordset.Update     ' บันทึกข้อมูลใหม่
        Unload Form2
        Form1.Show
        Exit Sub
Rec_Err:
        MsgBox ("มีข้อมูล [ " + Text1.Text + " ] อยู่แล้ว ไม่สามารถเพิ่มข้อมูลได้ ")
        Unload Form2
        Form1.Show
        Resume Next                      ' กลับไปทำงานต่อหลังคำสั่ง On Error
End Sub

Sub Command2_Click ()          ' ยกเลิก
        Data1.Refresh                    ' อ่านข้อมูลใน Record ใหม่
        Unload Form2
        Form1.Show                       ' กลับไปที่ Form 1
End Sub
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

VOICE.FRM

VERSION 2.00

Begin Form Form1

Caption	=	"ระบบฝากข้อความ"
ClientHeight	=	3915
ClientLeft	=	1305
ClientTop	=	1425
ClientWidth	=	5055
Height	=	4320
Left	=	1245
LinkTopic	=	"Form1"
ScaleHeight	=	3915
ScaleWidth	=	5055
Top	=	1080
Width	=	5175

Begin SSPanel Panel3D7

BackColor	=	&H00C0C0C0&
BevelInner	=	2 'Raised
BevelOuter	=	0 'None
DataField	=	"Password"
DataSource	=	"Data1"
Font3D	=	3 'Inset w/light shading
FontBold	=	-1 'True
FontItalic	=	0 'False
FontName	=	"MS Sans Serif"
FontSize	=	9.75
FontStrikethru	=	0 'False
FontUnderline	=	0 'False
Height	=	495

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        Left            =       3480
        TabIndex        =       8
        Top             =       2520
        Width           =       1455
End

Begin SSPanel Panel3D5
        BackColor       =       &H00C0C0C0&
        BevelInner      =       2  'Raised
        BvelOuter       =       0  'None
        DataField       =       "Name"
        DataSource      =       "Data1"
        Font3D          =       3  'Inset w/light shading
        FontBold        =       -1  'True
        FontItalic      =       0   'False
        FontName        =       "MS Sans Serif"
        FontSize        =       9.75
        FontStrikethru  =       0   'False
        FontUnderline   =       0   'False
        Height          =       495
        Left            =       360
        TabIndex        =       9
        Top             =       2520
        Width           =       2895
End
Begin MSComm Comm1
        Interval        =       1000
        Left            =       0
        Top             =       120
End
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
Begin VScrollBar VScroll1
        Height          =       615
        Left            =       4440
        Max             =       10
        Min             =       1
        TabIndex        =       7
        Top             =       960
        Value           =       1
        Width           =       255
End

Begin SSCommand Command3D1
        Caption         =       "เริ่มต้น"
        Font3D          =       3  'Inset w/light shading
        FontBold        =       -1  'True
        FontItalic      =       0  'False
        FontName        =       "MS Sans Serif"
        FontSize        =       9.75
        FontStrikethru  =       0  'False
        FontUnderline   =       0  'False
        Height          =       615
        Left            =       1680
        TabIndex        =       6
        Top             =       1800
        Width           =       1695
End
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
Begin S SPanel Panel3D6
        Align          =     2  'Align Bottom
        BackColor      =     &H00C0C0C0&
        BevelInner     =     2  'Raised
        BevelOuter     =     0  'None
        Caption        =     "พระจอมเกล้าลาดกระบัง"
        Font3D         =     3  'Inset w/light shading
        FontBold       =     -1  'True
        FontItalic     =     0   'False
        FontName       =     "Angsana New"
        FontSize       =     18
        FontStrikethru =     0   'False
        FontUnderline  =     0   'False
        Height         =     840
        Left           =     0
        TabIndex       =     5
        Top            =     3075
        Width          =     5055
End

Begin SSPanel Panel3D4
        BackColor      =     &H00C0C0C0&
        BevelInner     =     2  'Raised
        BevelOuter     =     0  'None
        Caption        =     "ครั้ง"
        Font3D         =     3  'Inset w/light shading
        FontBold       =     -1  'True
        FontItalic     =     0   'False
        FontName       =     "MS Sans Serif"
        FontSize       =     9.75
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        FontStrikethru  =       0  'False
        FontUnderline   =       0  'False
        Height          =       615
        Left            =       3720
        TabIndex        =       4
        Top             =       960
        Width           =       615
End

Begin SSPanel Panel3D3
        BackColor       =       &H00C0C0C0&
        BevelInner      =       2  'Raised
        BevelOuter      =       0  'None
        BevelWidth      =       2
        Caption         =       "3"
        Font3D          =       3  'Inset w/light shading
        FontBold        =       -1 'True
        FontItalic      =       0  'False
        FontName        =       "MS Sans Serif"
        FontSize        =       9.75
        FontStrikethru  =       0  'False
        FontUnderline   =       0  'False
        Height          =       615
        Left            =       3120
        TabIndex        =       3
        Top             =       960
        Width           =       495
End
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
Begin SSPanel Panel3D2
        BackColor       =       &H00C0C0C0&
        BevelInner      =       2  'Raised
        BevelOuter      =       0  'None
        Caption         =       "ทำงานหลังสัญญาณกระดิ่ง"
        Font3D          =       3  'Inset w/light shading
        FontBold        =       -1  'True
        FontItalic      =       0  'False
        FontName        =       "MS Sans Serif"
        FontSize        =       9.75
        FontStrikethru  =       0  'False
        FontUnderline   =       0  'False
        Height          =       615
        Left            =       480
        TabIndex        =       2
        Top             =       960
        Width           =       2535
End

Begin SSPanel Panel3D1
        BackColor       =       &H00C0C0C0&
        BevelInner      =       2  'Raised
        BevelOuter      =       0  'None
        Caption         =       "VOICE MAIL"
        Font3D          =       3  'Inset w/light shading
        FontBold        =       -1  'True
        FontItalic      =       0  'False
        FontName        =       "MS Sans Serif"
        FontSize        =       9.75
        FontStrikethru  =       0  'False
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        FontUnderline  =     0  'False
        Height         =     615
        Left           =     1680
        TabIndex       =     1
        Top            =     240
        Width          =     1815
End

Begin SSFrame Frame3D1
        Font3D         =     0  'None
        Height         =     3855
        Left           =     0
        TabIndex       =     0
        Top            =     0
        Width          =     5055

        Begin Data Data1
                Caption        =     "Data1"
                Connect        =     ""
                DatabaseName   =     "D:\VB\DATA\DATA.MDB"
                Exclusive      =     0  'False
                Height         =     270
                Left           =     0
                Options        =     0
                ReadOnly       =     0  'False
                RecordSource   =     "Infomation"
                Top            =     600
                Visible        =     0  'False
                Width          =     1140
        End
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
Begin Label Label2
        DataField       =       "Point"
        DataSource      =       "Data1"
        Height          =       255
        Left            =       0
        TabIndex        =       10
        Top             =       2280
        Visible         =       0   'False
        Width           =       855
End

Begin Label Label1
        DataField       =       "Head"
        DataSource      =       "Data1"
        Height          =       255
        Left            =       0
        TabIndex        =       11
        Top             =       1560
        Visible         =       0   'False
        Width           =       855
End

End

End
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
' ขอใช้ Function GetFocus และ SetFocousAPI ที่อยู่ใน USER.DLL
Declare Function GetFocus Lib "User" ( ) As Integer
Declare Function SetFocusAPI Lib "User" Alias "SetFocus" (ByVal hWnd As Integer) As Integer
' ประกาศตัวแปรให้เป็นตัวแปรแบบ Global
Dim Num_Ring As Integer
Dim Result As Integer
Dim Form_Handle As Integer
Dim MCMail_Handle As Integer
Dim MCStatus_Handle As Integer
Dim DTMF As String

Sub Command3D1_Click ( )
        Dim FName As String
        Dim TName As String
        Dim Dir1  As String
        Dim Dir2  As String
        Dim Key   As String
        Dim Count As Integer
        Dim H  As Integer
        Dim P As Integer
        Form_Handle = GetFocus( )      ' เก็บ Handle ของ Application ที่กำลัง Focus อยู่
        FName = "Greeting.gsm"
        TName = "Temp.gsm"
        Dir1 = "\qlmcw\Admin"
        Dir2 = "\voice"
        ChDir Dir1
        FileCopy FName, TName
        TName = "Mail*.GSM"
        On Error Resume Next
        Kill Tname
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
ChDir Dir2
TName = "Wellcome.gsm"
FileCopy TName, Dir1 + "\" + FName
Play                                    ' Play  Wellcome.gsm
Delay 1
Count = 0
Do
        DTMF1                   ' รอการกด Key และค้นหาข้อมูลใน Database
        If DTMF = "" Then               ' ถ้าสามารถไม่สามารถค้นหาได้
                TName = "Error.gsm"
                FileCopy TName, Dir1 + "\" + FName
                Play                    ' Play Error.gsm
                Count = Count + 1
        Else Exit Do
        End If
Loop Until Count >= 3                   ' รอรับการกด จนครบ 3 ครั้ง
If Count >= 3 Then                      ' กดหมายเลขเรียกผิด จนครบ 3 ครั้ง
        TName = "Thank.gsm"
        FileCopy TName, Dir1 + "\" + FName
        Play                            ' Play Thank.gsm
        Result = SetFocusAPI(Form_Handle)    ' ให้ Focus ไปที่ Form
        Exit Sub
End If
' เก็บข้อมูลเกี่ยวกับการบันทึกลง Record
H = Val(Label1.Caption)
P = Val(Label2.Caption)
TName = "play.gsm"
FileCopy TName, Dir1 + "\" + FName
Play                                    ' Play  play.gsm
Key = InputBox("กรุณากด 1 เพื่อฟังข้อความ")    ' รอการกด Key
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
If Key = "1" Then                          ' ตรวจสอบการกดหมายเลข 1
    Count = 0
    TName = "Pass.gsm"
    FileCopy TName, Dir1 + "\" + FName
    Play                                   ' Play  Pass.gsm
    Do
        Key = InputBox("กรุณากดรหัสผ่าน")
        If Key = Panel3D7.Caption Then     ' Password Correct
        Do
            If P = H Then  ' ตรวจสอบว่าหมดข้อความในการฟังหรือไม่
                P = 0
                H = 0
                TName = "No_Data.gsm"
                FileCopy TName, Dir1 + "\" + FName
                Play                       ' Play  No_Data.gsm
                TName = "Thank.gsm"
                FileCopy TName, Dir1 + "\" + FName
                Play                       ' Play Thank.gsm
                Exit Do
            End If
            P = P + 1
            TName = DTMF + "." + Str$(P)
            FileCopy TName, Dir1 + "\" + FName
            Play                   ' Play ข้อความที่ยังไม่ได้ฟัง
            Kill Tname             ' ลบข้อความนั้นทิ้ง
        Loop
        Label1.Caption = Str$(H)
        Label2.Caption = Str$(P)
        Data1.Recordset.Update     ' บันทึกข้อมูลต่างๆ
        Exit Do
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            Else
                TName = "Err_Pass.gsm"
                FileCopy TName, Dir1 + "\" + FName
                Play                    ' Play Err_Pass.gsm
                Count = Count + 1       ' นับการกดรหัสผิดเพิ่มขึ้น
            End If
        Loop Until Count >= 3
Else
        Result = SetFocusAPI(MCStatus_Handle)       ' Focus ไปที่ MC Mail
        SendKeys "{ESC}", True                      ' กด Key Esc
        Delay 1
        TName = "Rec.gsm"
        FileCopy TName, Dir1 + "\" + FName
        Result = SetFocusAPI(MCMail_Handle)
        SendKeys "%ot", True                        'เข้าสู่ Mode การบันทึก
        Delay 5
        Old = GetFocus( )
        N = Old
        Do                                          ' รอจนกว่าจะบันทึกข้อความจบ
            Dummy = DoEvents( )
            N = GetFocus( )
        Loop Until N <> Old
        H = H + 1
        TName = DTMF + "." + Str$(H)
        ChDir Dir1
        FileCopy "Mail0001.gsm", DIR2 + "\" + Tname      ' บันทึก File
        Kill "Mail0001.gsm"
        Delay 1
        SendKeys "^g", True
        Delay 2
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            Label1.Caption = Str$(H)
            Label2.Caption = Str$(P)
            Data1.Recordset.Update
            MCStatus_Handle = GetFocus( )
            Result = SetFocusAPI(Form_Handle)
        End If
        If Count >= 3 Then
            TName = "Thank.gsm"
            FileCopy TName, Dir1 + "\" + FName
            Play                        ' Play Thank.gsm
            Result = SetFocusAPI(Form_Handle)
            Exit Sub
        End If
        Data1.Recordset.MoveFirst       ' ให้กลับไปที่ Record แรก
        Result = SetFocusAPI(Form_Handle)   ' ให้ Focus ไปที่ Form
End Sub

Sub Delay (D As Integer)                ' หน่วงเวลา D วินาที
    Dim Dummy As Integer
    T = Timer
    Do
      Dummy = DoEvents( )
    Loop Until Timer > T + D
End Sub

Sub DTMF1 ( )           ' รอการกดKey พร้อมทั้งตรวจสอบว่ามีหมายเลขอยู่ในฐานข้อมูลหรือไม่
    Result = SetFocusAPI(Form_Handle)
    DTMF = InputBox("กรุณาป้อนหมายเลขเรียกที่ต้องการเรียก : ")
    Search
End Sub
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
Sub Form_Load ( )                    ' เริ่มต้นโปรแกรม
    Num_Ring = 4
    Init                             ' กำหนดค่าเริ่มต้นของ Program
    Init_Menu                        ' กำหนดค่าเริ่มต้นของ Menu
End Sub

Sub Form_Unload (Cancel As Integer)          ' จบโปรแกรม
    Result = SetFocusAPI(MCStatus_Handle)
    SendKeys "{ESC}", True
    Delay 1
    SendKeys "%fx", True                     ' ปิดโปรแกรม MCMail
    End
End Sub

Sub Init ( )
    On Error Resume Next
    AppActivate "Program Manager"
    SendKeys "%{ }{DOWN 3}{Enter}", True          ' ลดขนาด Program Manager
    AppActivate "Microsoft Visual Basic"
    SendKeys "%{ }{DOWN 3}{Enter}", True          ' ลดขนาดโปรแกรม Virsual Basic
    MCMail_Handle = Shell("D:\QLMCW\MCMAIL.EXE", 1)   ' เรียกโปรแกรม MCMail
    If Err Then                                   ' ถ้ามีการเปิดอยู่แล้ว
      AppActivate "MCMAIL"                        ' ให้ MCMail ทำงาน
    End If
    Delay 1
    MCMail_Handle = GetFocus( )                   ' เก็บค่า Handle ของโปรแกรม MCMail
    SendKeys "^g", True
    Delay 1
    MCStatus_Handle = GetFocus( )
    Delay 1
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    SendKeys "%{TAB}", True
End Sub

Sub Init_Menu ( )
    Panel3D3.Caption = Str$(Num_Ring)
    VScroll1.Value = Num_Ring
    SendKeys "%{TAB}{TAB}", True
End Sub

Sub Play ( )
    Dim Size As Long
    Result = SetFocusAPI(MCStatus_Handle)
    SendKeys "%p", True
    Size = FileLen("\qlmcw\admin\Greeting.gsm")
    TD = (Size - 8) / 2030
    T = Timer
    Do                                  ' รอจนกว่าจะพูดจบ
      Dummy = DoEvents( )
    Loop Until Timer > T + TD + 1
End Sub

Sub Search ( )                          ' ค้นหาข้อมูลในฐานข้อมูล
    Data1.Recordset.FindFirst "ID=" + Chr$(39) + DTMF + Chr$(39)
    If Data1.Recordset.NoMatch Then     ' มีข้อมูลหรือไม่
      DTMF = ""
    End If
End Sub

Sub VScroll1_Change ( )                 ' มีการกด Scoll
    Num_Ring = VScroll1.Value
    Panel3D3.Caption = Str$(Num_Ring)
End Sub
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้