การส่งโทรสารโดยผ่านแฟกซ์โมเด็ม

FACSIMILE TRANSMISSION BY FAXMODEM

โดย

นายศักดิศ สุวัชรังกูร

นายสันติ ลีอวนิชวงศ์

อาจารย์ที่ปรึกษา

อาจารย์เกรียงไกร วงศ์โรจนภรณ์

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตร์บัณฑิต

ภาควิชาโทรคมนาคม

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2537

ปริญญานิพนธ์ปีการศึกษา 2537

ภาควิชา วิศวกรรมโทรคมนาคม

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง การส่งโทรสารโดยผ่านแฟกซ์โมเด็ม

ผู้จัดทำ

1.นายศักดิศ สุวัชรังกูร 34107369

2.นายสันติ ลือวนิชวงศ์ 34107411

.....

(อาจารย์เกรียงไกร วงศ์โรจนภรณ์)

อาจารย์ที่ปรึกษา

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การส่งโทรสารโดยผ่านแฟกซ์โมเด็ม

FACSIMILE TRANSMISSION BY FAXMODEM

โดย นายศักดิศ สุวัชรังกูร
นายสันติ ลือวนิชวงศ์
อาจารย์ที่ปรึกษา
อาจารย์เกรียงไกร วงศ์โรจนภรณ์
ปีการศึกษา 2537

บทคัดย่อ

ในโครงงานจะกล่าวเกี่ยวกับการรับ/ส่งโทรสารโดยใช้แฟกซ์โมเด็มแทนการรับ/ส่งโทรสารโดยใช้เครื่องโทรสารธรรมดาในการรับ/ส่งโทรสารจะใช้เครื่องคอมพิวเตอร์เป็นตัวควบคุมขบวนการทำงาน ทั้งนี้โดยอาศัยการทำงานร่วมกับแฟกซ์โมเด็ม โดยจะใช้คำสั่ง Enhanced AT command Set (EIA TR 30.2.2/88) และคำสั่งของแฟกซ์โมเด็มระดับ 2 (SP-2388-A Class 2) ซอฟต์แวร์ที่เขียนขึ้นเพื่อการควบคุมการทำงานของแฟกซ์โมเด็มโดยพัฒนาให้ทำงานกับแฟกซ์โมเด็มรุ่นDiscovery 2496 ของ Datatronics และสามารถที่จะส่งไฟล์ที่เป็นรหัสภาษาไทยซึ่งได้จากจุฬาเวิร์ด (CU Writer) หรือ ราชวิถีเวิร์ด (Ratvitee Word)

ABSTRACT

In this project is said about facsimile transmission by using faxmodem not by using common fax machine.Facsimile transmisson is controled by computer and work with faxmodem. The computer control faxmodem by using Enhanced AT command set (EIA TR 30.2.2/88) and faxmodem class 2 command (SP-2388-A Class 2). The purpose of this project is to develope the software that control faxmodem to transmission facsimile with Discovery 2496 faxmodem and can send thai ascii file from CU writer or Ratvitee Word

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 1
บทนำ

ในปัจจุบันการติดต่อสื่อสารทางโทรสารเป็นที่นิยมใช้งานอย่างกว้างขวางเนื่องจากมีจุดเด่นอยู่หลายประการ เช่น สามารถรับส่งข้อมูลข่าวสารที่เป็นเอกสารได้ สามารถใช้งานร่วมกับโครงข่ายโทรศัพท์ได้ สามารถใช้งานได้สะดวกรวดเร็ว ดังนั้นเครื่องโทรสารจึงได้รับการพัฒนาอยู่ตลอดเวลา เพื่อให้สามารถตอบสนองต่อความต้องการทั้งด้านความเร็วและความถูกต้องแม่นยำของข่าวสาร

เพื่อให้การพัฒนาเป็นไปในแนวทางเดียวกัน ทางสมาคมอิเล็คโทรนิคอุตสาหกรรม EIA (The electronics Industry Association)แห่งสหรัฐจึงได้มีการกำหนดมาตรฐานการสื่อสารทางโทรสารขึ้นในปี ค.ศ. 1960 และต่อมาในปี ค.ศ. 1968 ได้กำหนดมาตรฐาน CCITT ขึ้นมาใหม่ และใช้เป็นแนวทางการพัฒนามาจนถึงปัจจุบัน จากมาตรฐาน CCITT นี้ได้แบ่ง เครื่องโทรสารออกเป็น 4 กลุ่มคือ

กลุ่มที่ 1 ใช้การผสมสัญญาณ(modulation) แบบ เอเอ็ม (AM) และ แบบเอฟเอ็ม (FM) โดยไม่มีกรรมวิธีพิเศษใดๆ มาบีบแถบความกว้างของความถี่ (bandwidth) ของการส่ง กลุ่มนี้ใช้เวลาในการส่งเอกสารขนาด A4 ประมาณ 6 นาที

กลุ่มที่ 2 พัฒนามาจากกลุ่มแรกโดยใช้เทคนิคการบีบแถบความกว้างของความถี่ (bandwidth) ซึ่งได้แก่ การเข้ารหัสแต่ไม่รวมถึงการประมวลผลเพื่อลดสัญญาณที่ไม่จำเป็น กลุ่มนี้ใช้เวลาในการส่งเอกสารขนาด A4 ประมาณ 3 นาที

กลุ่มที่ 3 ใช้การเข้ารหัส และลดขนาดข้อมูลที่ไม่จำเป็นของสัญญาณภาพก่อนจะทำการมอดดูเลทส่งออกไป และยังใช้การกดแบนด์วิธเพื่อลดเวลาลง โดยใช้เวลาในการส่งเอกสารขนาด A4 ประมาณ 1 นาที

กลุ่มที่ 4 สามารถส่งข้อมูลได้อย่างถูกต้อง ด้วยความเร็วสูง และมีความสามารถพิเศษเพิ่มขึ้น ใช้ในโครงข่ายข้อมูลสาธารณะ (Public Data Networks:PDN) และ ระบบ ISDN (Integrated Services Network)

นอกจากนี้ในปัจจุบันได้มีการนำเครื่อง คอมพิวเตอร์ มาใช้ในการทำงานอย่างกว้างขวาง ทั้งในระบบฐานการเก็บข้อมูล ,การคำนวณ ,การแสดงผล เพื่อให้เกิดความสะดวกในการใช้งานร่วมกับเครื่องโทรสารจึงได้มีการนำเทคโนโลยีทางเครื่องโทรสาร มารวมกับ ดาต้าโมเด็ม (data modem) ซึ่งจะสามารถทำการส่งข้อมูลข่าวสารจากเครื่องคอมพิวเตอร์ ไปยังเครื่องโทรสาร หรือ รับข้อมูลข่าวสารจากเครื่องโทรสารอื่นได้โดยตรง ซึ่งเรียกโมเด็มชนิดนี้ว่า แฟกซ์โมเด็ม (fax modem)

เพื่อให้การพัฒนาเป็นไปในแนวทางเดียวกัน ทางสมาคม EIA จึงได้มีการกำหนดมาตรฐานของแฟกซ์โมเด็มเป็น 3 ระดับ(class)

1. แฟกซ์โมเด็มระดับ 1 จะรองรับขั้นตอนการทำงานของ แฟกซ์กลุ่ม 3 เท่านั้น คอมพิวเตอร์จะต้องตอบสนองการทำงานในการเข้ารหัสของภาพตามมาตรฐาน T.4 และการจัดการในการส่งข้อมูลตามมาตรฐาน T.30

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2. แฟกซ์โมเด็มระดับ 2 รองรับขั้นตอนการทำงานของโทรสารกลุ่ม 3 และกลุ่ม 4 ส่วนขั้นตอนที่ซับซ้อนในมาตรฐาน T.30 จะรับผิดชอบโดยส่วนของฮาร์ดแวร์(hardware)ของแฟกซ์โมเด็มแทน

3. แฟกซ์โมเด็มระดับ 3 ยังอยู่ในระหว่างการศึกษาแต่มีแนวโน้มที่จะโอนหน้าที่ต่างๆจากคอมพิวเตอร์ ไปให้แฟกซ์โมเด็มมากขึ้น

ในโครงงานนี้จึงทำการเขียนโปรแกรมเพื่อมาควบคุมแฟกซ์โมเด็มโดยใช้กลุ่มคำสั่งสำหรับแฟกซ์โมเด็มระดับ 2 ในการส่งโทรสารตามมาตรฐานของการส่งโทรสารกลุ่ม 3 โดยสามารถที่จะส่งไฟล์ที่ได้จากเวิร์ดภาษาไทยเช่นจุฬาเวิร์ดหรือราชวิถีเวิร์ด รวมทั้งไฟล์รหัสแอสกี้ทั่วไป และยังได้เขียนโปรแกรมในส่วนของการรับโทรสารโดยจะใช้เป็นโหมด (mode) การทำงานเป็นแบบอัตโนมัติคือเมื่อมีเครื่องโทรสารปลายทางถูกเรียกโดยเครื่องโทรสารต้นทาง โปรแกรมจะทำการตอบรับโทรสารโดยอัตโนมัติทันทีแล้วเก็บเป็นไฟล์ฮัฟฟ์แมนลงในฮาร์ดดิสค์ (hard disk) หรือ ฟลอปปี้ดิสค์(floppy disk) หลังจากนั้นผู้ใช้สามารถจะดูรายละเอียดของโทรสารที่เรียกเข้ามาได้โดยจะนำไปแสดงเป็นจุดขาวดำบนหน้าจอของคอมพิวเตอร์

หมายเหตุ

มาตรฐาน T.4 เป็นมาตรฐานในการส่งโทรสารกลุ่ม 3 ของ CCITT

มาตรฐาน T.30 เป็นมาตรฐานในการกำหนดขบวนการส่งโทรสารกลุ่ม 3 ในโครงข่ายโทรศัพท์ทั่วไปของCCITT

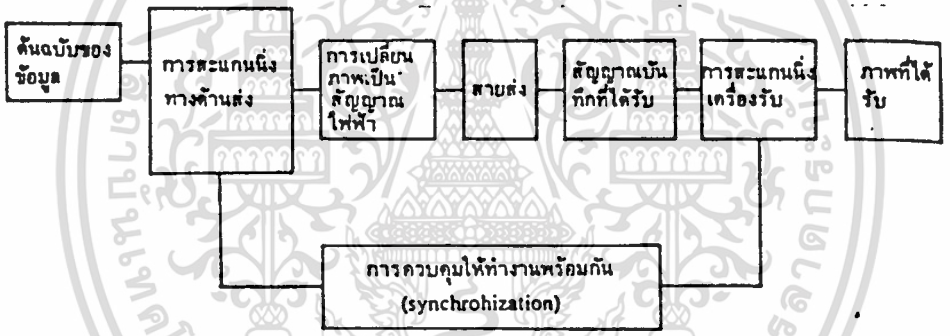
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 2
เทคโนโลยีของเครื่องโทรสาร

หลักการเบื้องต้นของเครื่องโทรสาร

เครื่องโทรสารทางด้านส่งจะส่งสัญญาณภาพต่างๆ โดยการสแกนภาพที่ต้องการส่งนั้นๆและเปลี่ยนภาพเหล่านั้นเป็นสัญญาณไฟฟ้า เพื่อส่งผ่านสายสัญญาณ แล้วทางด้านรับจะเปลี่ยนกลับโดยการสแกนเป็นภาพเดิมอีกครั้งหนึ่ง โดยทั่วๆไปเครื่องโทรสารด้านส่งจะส่งสัญญาณภาพ โดยอาศัยไดอะแกรมต่างๆของข้อมูลขาว,ดำ และทางด้านรับทำการสร้างภาพที่มีระดับขาวดำเหมือนของเดิม แต่อย่างไรก็ตามสิ่งที่สำคัญที่สุดในการสแกนก็คือ ทางด้านเครื่องส่งและเครื่องรับจะต้องมีความเร็วของการสแกนที่มีการทำงานสอดคล้องกัน (synchronization)

หลักการทำงานเบื้องต้นของเครื่องโทรสาร ดังแสดงในรูปที่ 2.1



รูปที่ 2.1 หลักการทำงานเบื้องต้นของเครื่องโทรสาร

1)สีดำ และสีขาวบนแผ่นข้อมูล หรือ ภาพจะถูกส่งเรียงแถว หรือ ที่เรียกว่าการสแกน(scanning) สำหรับเครื่องส่ง

2) ในเครื่องโทรสารจะอาศัยแสงที่ตกกระทบบนภาพ เพื่อให้ได้ข้อมูลที่มีความเข้มขาว,ดำ ซึ่งจะถูกเปลี่ยนเป็นพลังงานไฟฟ้าตามระดับของสีขาว,ดำ เรียกกระบวนการนี้ว่าการเปลี่ยนสัญญาณภาพ เป็นสัญญาณไฟฟ้า (photoelectric conversion)

3) พลังงานไฟฟ้าที่เปลี่ยนแล้วนี้จะส่งไปสายส่งเพื่อผ่านไปยังอุปกรณ์ส่งสัญญาณ(transmission)

4) การผสมผสานระหว่างสัญญาณขาว และ ดำ (modulation)

5) พลังงานไฟฟ้าทางด้านรับจะถูกเปลี่ยนกลับ เป็นความร้อนตามระดับของภาพความเข้ม ขาว,ดำ เพื่อให้ความร้อนแก่ตัวสไตลัส (stylus)ทางเครื่องรับ ซึ่งจะมากหรือน้อยขึ้นอยู่กับกระแสที่ได้รับ

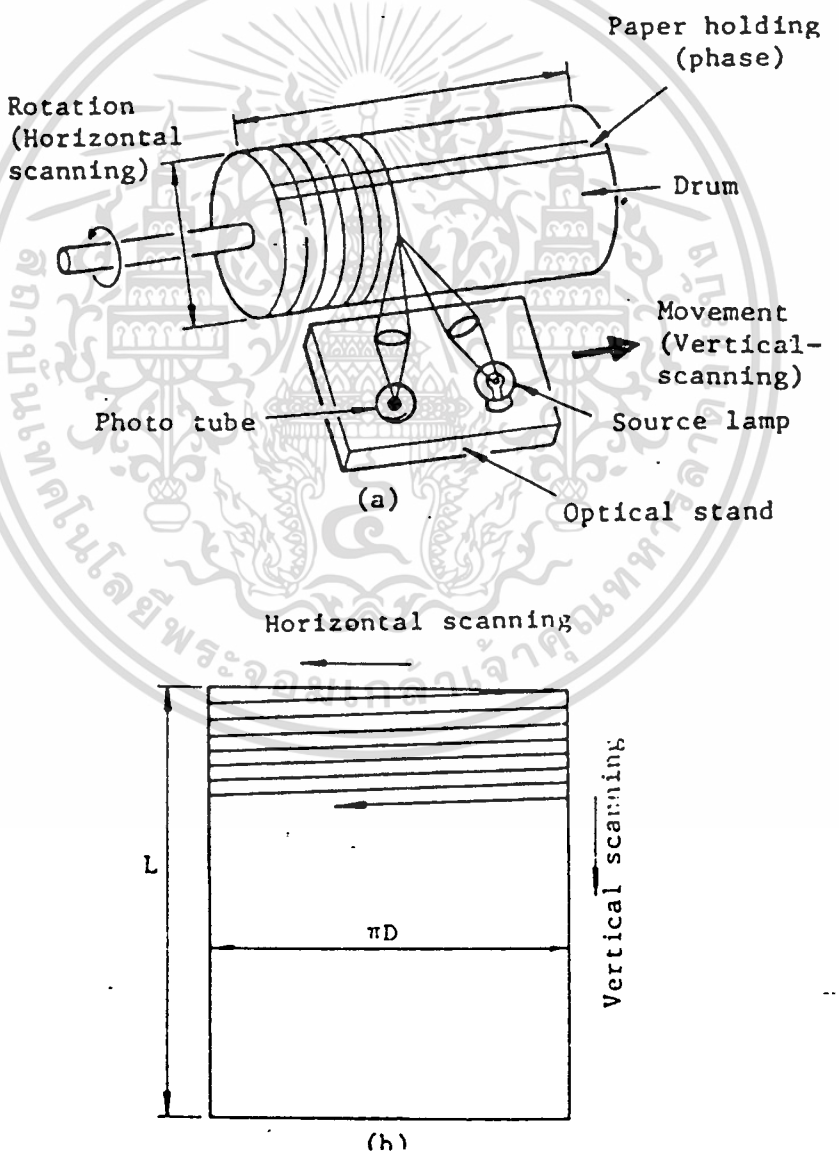
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

6) ทางด้านเครื่องรับจะสแกนภาพเหมือนกับ ภาพที่ทางเครื่องส่งส่งมาทุกประการ และจะทำงานพร้อม ๆ กัน (synchronization)

การทำงานทั้งหมดนี้ขึ้นอยู่กับการทำงานทางอิเล็คทรอนิกส์, ทางกลไก และ ทางเคมี ตลอดจนเทคโนโลยีอื่นๆ ที่นำมาใช้

<u>การสแกน(scanning)</u>

การสแกน คือ การแบ่งภาพเป็นไดเมนชั่นเพลน ซึ่งประกอบด้วยจุดภาพ(picture element) จำนวนมาก ดังรูป 2.2(a) การสแกนนี้มีทั้งทิศทางในแนวตั้งและแนวนอน ซึ่งเปรียบเทียบได้กับการสแกนของจอโทรทัศน์ ดังรูปที่ 2.2(b)



รูปที่ 2.2 แสดงกลไก และการทำงานของการสแกน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3) หลอดปล่อยประจุ ส่วนมากเป็นหลอดนีออนอาร์กอน หรือ ยีเลียมแก๊ส เป็นแหล่งกำเนิดแสงสำหรับเครื่องส่ง และเครื่องรับ ซึ่งตอบสนองกับความถี่ประมาณ 100 กิโลเฮิรตซ์ (KHz) และ ความสว่างสามารถจะปรับได้ด้วยกระแส เข้า และอายุการใช้งานสั้น(100 ถึง 500 ชั่วโมงการใช้งาน)

4) เลเซอร์ แสงเลเซอร์มีข้อดีคือ ง่ายต่อการปรับให้เป็นจุดเล็ก ๆ และมีพลังแสงที่แรงเพื่อการบันทึก มี 2 ระบบคือ อินเทอร์นอลโมดูเลชั่น(internal modulation) และอีกอย่างหนึ่งเรียกว่า เอ็กเทอร์นอลโมดูเลชั่น(external modulation)

การส่ง(transmission)

สัญญาณไฟฟ้าที่ได้จากการเปลี่ยนแปลงทางโฟโต้อีเล็คทริก จะขึ้นอยู่กับ ระบบการส่ง และ ช่องทางที่ใช้ในการส่ง เช่น ในสายเคเบิล ทางคลื่นวิทยุ เป็นต้น ส่วนทางด้านเครื่องรับก็จะรับสัญญาณภาพ และสแกนภาพเหมือนกับภาพเดิมอีกครั้งหนึ่ง

ระบบการส่งภาพจะมี 3 วิธีด้วยกันคือ

1) การส่งแบบเบสแบน (base band transmission)

2) การส่งแบบอนาล็อก (analog mod/demodulation)

3) การส่งแบบดิจิตอล (digital encoding-decoding)

การส่งแบบเบสแบน (base band transmission)

สัญญาณที่ได้รับประกอบด้วยไฟกระแสตรงเป็นหลัก ซึ่งจะยอมให้ความถี่ต่ำผ่านได้ และการส่งแบบเบสแบนนี้จะใช้ในการส่งที่ความถี่สูงโดยอาศัยสายโทรศัพท์

การส่งแบบอนาล็อก (analog mod/demodulation)

การส่งแบบนี้จะใช้สายโทรศัพท์เป็นตัวกลางในการส่งซึ่งอยู่ในย่านความถี่ 300-3400 เฮิรตซ์ (Hz) เหมือนกับการส่งเอฟดีเอ็ม (FDM:frequency division multiplex) สัญญาณที่จะส่งนี้จะรวมกับคลื่นพาหะ (carrier frequency) แล้วส่งออกไปยังสายส่งโทรศัพท์ ภาพที่รับได้ในเครื่องรับจะต้อง ทำการแยกคลื่นพาหะออกไป

การส่งแบบดิจิตอล (transmission digital encoding detecoding)

การส่งแบบดิจิตอลมีการทำงานเหมือนการส่ง ทีดีเอ็ม (TDM:time division multiplex) โดยมีความจำเป็นที่จะส่งสัญญาณภาพหลังจากเปลี่ยนเป็นสัญญาณ 2 ระดับ สัญญาณภาพที่เป็นแซมปลิ้ง (sampling) และฟอนติเซชั่น และ จะรับภาพทางด้านรับเหมือนกับภาพที่ส่งมาโดยตัวถอดรหัส (decoder)

การบันทึกภาพ (conversion for recording)

การบันทึกภาพมีหลายวิธีด้วยกัน โดยอาศัยการกระตุ้นภายนอกสำหรับการบันทึก เช่นการกระตุ้นด้วย กระแสไฟฟ้า ,พลังงานแสง ,พลังงานความร้อน และแรงอัด เป็นต้น

1) การบันทึกด้วยอีเลคโทรเซนซิทีป (eletrosensitive recording)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การสแกนมีความสำคัญมากในเครื่องโทรสาร การสแกนสามารถแบ่งได้เป็น 2 อย่างคือ การสแกนแบบกลไก และการสแกนแบบอิเล็คทรอนิกส์

การสแกนแบบกลไก แบ่งออกเป็น 3 อย่างคือ

1. สแกนทรงกระบอก (drum scanning)
2. สแกนรูปโค้ง (circular scanning)
3. สแกนบนพื้นราบ (flat-bed scanning)

การสแกนทรงกระบอก นี้มีข้อเสียมาก ภาพที่บันทึกในทางเครื่องรับไม่สามารถที่จะดูได้ แต่มีข้อดีสำหรับที่จะถอดกระดาษออกทีละครั้งเมื่อบันทึกภาพเสร็จแล้ว และเป็นหลักการเบื้องต้นที่จะอธิบายให้เข้าใจได้ง่ายทั้งเป็นเครื่องมือที่สะดวกและประหยัดอีกด้วย

การสแกนแบบรูปโค้ง ในระบบนี้การเคลื่อนของทั้งภาพส่งและกระดาษในเครื่องรับ จะมีลักษณะเป็นในรูปของครึ่งวงกลม โดยมีตัวสแกนก็เป็นตัวหมุน

การสแกนบนพื้นราบ ระบบนี้เหมาะในที่ทำงานของสำนักงาน ซึ่งการทำงานเหมือนกับเครื่องถ่ายสำเนา(copying machine)

สแกนแบบอิเล็คทรอนิกส์

อิเล็คทรอนิกส์สแกนเป็นระบบที่ใช้ความเร็วสูง และมีจุดเสียน้อยกว่าการสแกนแบบกลไก การสแกนแบบอิเล็คทรอนิคส์ นี้สามารถแบ่งเป็น 2 แบบใหญ่ๆ ด้วยกันคือ

1) การสแกนด้วยหลอดแคดโทดเรย์ สำหรับการส่งโทรสารส่วนใหญ่จะใช้หลอดฟิคอับทิวกับหลอ่ดฟลายอิ่งสปอดทิว แต่จะมีข้อเสียคือ ใช้งานได้เฉพาะอย่าง

2)โซลิตสเตทสแกน (solid state scan) เครื่องส่งส่วนใหญ่จะใช้หลอดโฟโตไดโอดเล็กๆ และอุปกรณ์ประเภทอินติเกรดเตดเซอร์กิต (IC) ทำให้เครื่องมีขนาดเล็กลงมากมีความเร็วในการทำงานดี ซึ่งอนาคตข้างหน้าจะนิยมใช้กันมาก

<u>การเปลี่ยนแปลงทางโฟโต้อีเล็คทริก (photoelectric conversion)</u>

ต้นฉบับของภาพที่ถูกสแกน จะแบ่งออกเป็นส่วนประกอบเล็กๆ ซึ่งส่วนประกอบเล็กๆ เหล่านี้จะถูกเปลี่ยนเป็นสัญญาณไฟฟ้า โดยอาศัยแสงจากแหล่งกำเนิดแสงไปตกกระทบ และสะท้อนกลับมา ซึ่งเรียกว่า การเปลี่ยนแปลงทางโฟโต้อีเล็คทริก

แหล่งกำเนิดแสง (light source)

แหล่งกำเนิดแสงโดยทั่วไปจะต้องมีความสว่าง และอายุการใช้งานทนทาน เมื่อใช้เป็นเครื่องส่งจะต้องตรวจสอบความยาวคลื่นของแสงที่ส่งออกไปยังภาพ สำหรับแหล่งกำเนิดแสงแบ่งออกได้ดังนี้

1) หลอดทังสเตน (tungsten lamp)เป็นหลอดเล็ก ๆ ที่นิยมใช้มาก ใช้กับกระแสไฟฟ้าซึ่งมีแรงเคลื่อนต่ำ แต่กระแสมาก

2) หลอดฟลูออรีสเซ่นต์ ในกรณีสแกนบนพื้นราบ บนพื้นผิวของภาพใช้กับไฟฟ้ากระแสตรงหรือความถี่สูง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ส่วนประกอบ และโครงสร้างของการดาษอีเลคโทรเซนซิทีป บนพื้นผิวของกระดาษจะถูกทาด้วยคาร์บอนสีดำ ซึ่งเหมือนกับสื่อตัวนำ เมื่อสไตลัท (stylus) หรือ เข็มปล่อยประจุไฟฟ้า จะทำให้สื่อตัวนำ (conductive material) ถูกทำลายเหลือแต่คาร์บอนสีดำ จะปรากฏภาพบนกระดาษบันทึก

กระดาษบันทึกเหล่านี้ส่วนมากใช้กับเครื่องแบบทรงกระบอกหมุน และเครื่องบันทึกหลายชนิดก็ใช้การดาษชนิดนี้เพราะสะดวก และการบันทึกทำได้อย่างรวดเร็ว ตามกระดาษทั่วไปใช้แรงเคลื่อนไฟฟ้าในการบันทึกประมาณ 150-200 โวลท์ ราว ๆ 20 มิลิแอมแปร์ แต่ชนิดใหม่นี้ จะใช้แรงเคลื่อนเพียง 60-70 โวลท์เท่านั้น

2) การบันทึกแบบอีเลคโทรสเตติก (electrostatic recording)

การบันทึกภาพแบบนี้เป็นที่นิยมใช้มาก เพราะประสิทธิภาพของการบันทึกดีมาก และยังบันทึกด้วยความเร็วสูง

3) เทอร์โมเซนซิทีฟเรคคอคดิ่ง (thermosensitive recording)

กระดาษเทอร์โมเซนติทีฟ แบ่งเป็น 2 ชนิด คือ ระบบทางฟิสิกส์ และระบบการเปลี่ยนแปลงทางเคมี ซึ่งคงจะเป็นที่นิยมใช้กับเครื่องโทรสารในอนาคต

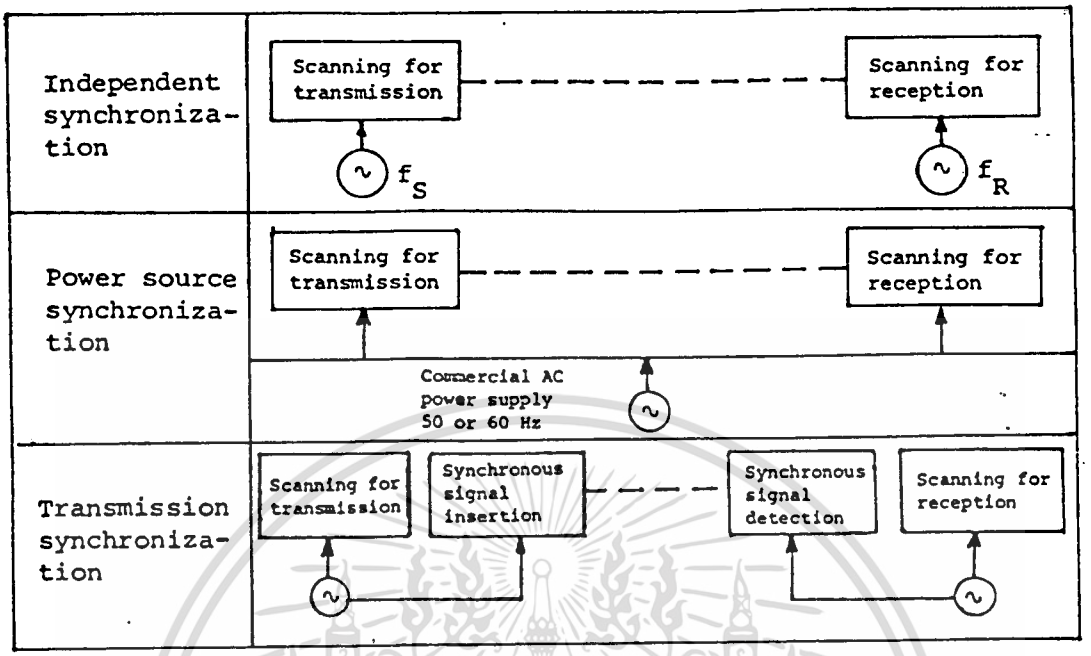
4) การบันทึกแบบอีเลคโทรเทอร์มอล (electrothermal recording)

ในระบบนี้ใช้วิธีบันทึกแบบให้พลังความร้อนให้กับกระดาษบันทึก โดยใช้แรงเคลื่อน 50-60 โวลท์ และใช้เวลาในการบันทึกเร็วประมาณ 3 นาที นอกจากนี้ยังมีวิธีการบันทึกอีกหลายชนิด เช่น อีเลคโทรไลด์ติกอิงค์เลช และกระดาษคราย์ซิลเวอร์ อีกด้วย

<u>การทำงานสอดคล้องกัน</u>

การสื่อสารทางเครื่องโทรสาร การสแกนทางด้านส่ง และทางด้านรับจะต้องทำงานสอดคล้องกัน ซึ่งสามารถแบ่งการทำงานที่สอดคล้องกันนี้ได้ 2แบบด้วยกันคือ ความสอดคล้องกันของการหมุน (rotation synchronization) และ ความสอดคล้องกันของเฟส (phase synchronization)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.3 แสดงวิธีการทำงานที่สอดคล้อง

1) ความสอดคล้องกันของการหมุน (rotation synchronization)

ความสอดคล้องกันของการหมุนได้จากการใช้มอเตอร์ที่หมุนพร้อมกัน (synchronization motor) ซึ่งมีวิธีการต่าง ๆ คือ

- ความสอดคล้องกันของการหมุนแบบอิสระ
- ความสอดคล้องกันของการหมุนแบบความถี่ของกำลังไฟฟ้า
- ความสอดคล้องกันของการหมุนแบบการส่งสัญญาณ

ความสอดคล้องกันของการหมุนแบบอิสระ มีออสซิเลเตอร์ที่ทำงานแน่นอนทั้งทางด้านส่งและด้านรับ โดยให้ความถี่ทั้งสองเท่ากัน (fS=fR) ดังรูปที่ 2.3

ความสอดคล้องกันของการหมุนแบบความถี่ของไฟฟ้ากำลัง ใช้กระแสไฟสลับของการไฟฟ้า 50 เฮิรตซ์ (Hz) ทั้งทางด้านส่ง และด้านรับ ซึ่งต้องเหมือนกัน ทำให้การหมุนพร้อมกัน

ความสอดคล้องกันของการหมุนแบบการส่งสัญญาณ ใช้วิธีการส่งสัญญาณเพื่อให้ทำงานสอดคล้องกัน (synchronization signal) ไปยังด้านรับพร้อมกันกับการสแกนภาพ ด้านรับจะแยกสัญญาณออกเพื่อนำไปควบคุมให้มีการหมุนพร้อมกันกับด้านส่ง

2) ความพร้อมกันของเฟส (phase synchronization)

สัญญาณซิงโครไนซ์นี้ใช้ชี้ตำแหน่งที่เริ่มต้นของการสแกน ซึ่งส่งไปก่อนสัญญาณภาพ ซึ่งจะทำให้ทั้งการหมุนพร้อมกันและการสแกนพร้อมกันทั้งสองด้าน

การเฟสซิ่งแบ่งออกเป็น 2 วิธีคือ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

-วิธีเฟสแทรคกิง (phase tracking)

-วิธีวัน-ชอท (one shot)

วิธีเฟสแทรคกิงก็คือ การเปลี่ยนแปลงของความเร็วทีละน้อย ทีละน้อยทั้งเครื่องส่ง และเครื่องรับ จนกระทั่งการสแกนได้ความเร็วคงที่

วิธีวัน-ชอท ทางด้านเครื่องรับจะหยุดเพื่อรอสัญญาณเฟสซิ่งจากเครื่องส่ง แล้วการสแกนจึงเริ่มทำงาน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

034820

บทที่ 3

มาตรฐานและคุณสมบัติของเครื่องโทรสารกลุ่ม 3

คุณสมบัติของเครื่องโทรสารกลุ่ม 3 ตามมาตรฐานของ CCITT แสดงดังต่อไปนี้

Item	Standard	Options		Small Copy (A5&A6)			
Scan Width							
in	8.46	10	11.9	4.2	5.9	5.9	4.2
mm	215	255	303	107	151	151	107
Pels/line	1728	2048	2432	854	1216	1728	1728
H /in	203			203	203	290	406
/mm	8			8	8	11.4	16
V /in	97.8	196		196/392	138/176	138/176	196/392
/mm	3.85	7.7		7.7/15.4	5.44/10.9	5.44/10.9	7.7/15.4
ms/line	20	0, 5, 10, 40					
Coding	Modified Huffman	Modified Read, Modified-Modified Read					
Modem							
Fax signal	V.27ter	V.29		V.17			
Bits/s	2400/4800	9600/7200		14400, 1200, 9600, 7200			
Handshake	V.21 (Ch2)	V.27ter					
Bits/s	300	2400					
Error Correction	None	Error Correction Mode					

ตารางที่ 3.1 คุณสมบัติของเครื่องโทรสารกลุ่ม 3 ตามมาตรฐานของ CCITT

คุณสมบัติทั่วไปของเครื่องโทรสารกลุ่ม 3

ความละเอียด

โดยจะมีความละเอียดของภาพที่ได้จากการรับส่ง เท่ากับ 203 พิคเซล(pixel)ต่อนิ้ว ในแนวนอน และ 98 พิคเซลต่อนิ้ว ในแนวตั้ง

เวลาที่ใช้ในการส่ง

เครื่องโทรสารกลุ่ม 3 จะใช้เวลาในการส่งแต่ละหน้าไม่แน่นอน ขึ้นอยู่กับจำนวนของจุดดำและรายละเอียดของข้อมูลที่ส่ง แต่โดยทั่วไปจะส่งข้อมูลได้โดยเฉลี่ย 1 หน้า ในเวลา 10-30 วินาที การเพิ่มขนาดความจุของหน่วยความจำในเครื่องโทรสารกลุ่ม 3 ทำให้สามารถส่งได้เร็วขึ้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ใช้งานง่าย

ผู้ใช้สามารถป้อนต้นฉบับได้อย่างง่ายดาย และเพียงกดปุ่มเพื่อทำการส่งไปยังที่หมายตามต้องการ โดยที่เครื่องโทรสารกลุ่ม 3 สามารถเรียกหมายเลขปลายทางอัตโนมัติได้ และสามารถตอบรับสัญญาณเรียกเข้าได้โดยอัตโนมัติเช่นกัน หลังจากรับข้อมูลเรียบร้อยแล้ว เครื่องจะวางสายและยกเลิกการติดต่อจนกว่าจะมีการเรียกครั้งใหม่เข้ามา

ใช้กับคู่สายโทรศัพท์ทั่วไปได้

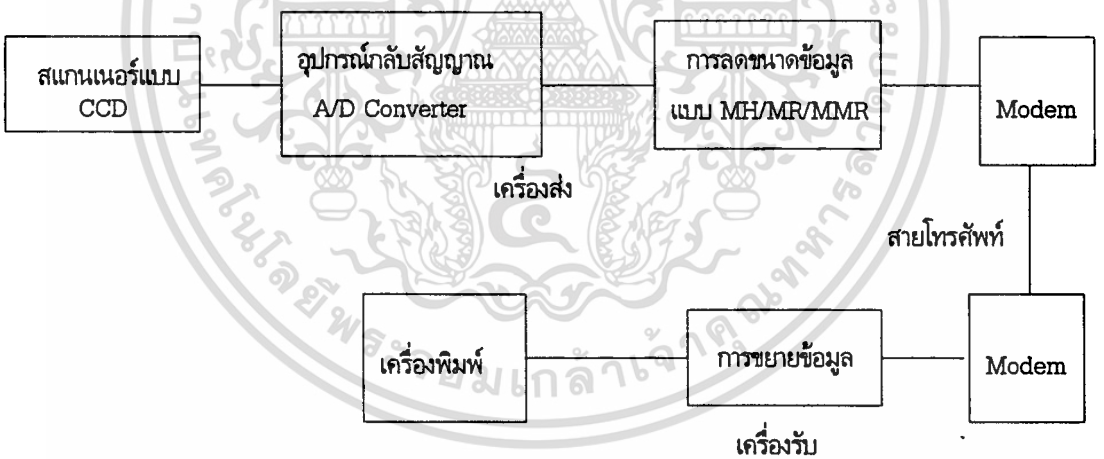
เครื่องโทรสารกลุ่ม 3 สามารถใช้กับสายโทรศัพท์ตามปกติได้ โดยเพียงแต่ต่อเครื่องโทรสารเข้ากับโทรศัพท์เท่านี้ ก็สามารถส่งข้อมูลไปยังปลายทางที่มีคู่สายโทรศัพท์ได้ตามต้องการ

โมเด็ม (modem) สามารถปรับความเร็วในการส่งได้

โมเด็ม (modem) ที่ใช้ในเครื่องโทรสารกลุ่ม 3 สามารถตรวจสอบสัญญาณการติดต่อระหว่างเครื่องโทรสารทั้ง 2 ข้างได้ และสามารถเลือกความเร็วที่เหมาะสมในการส่งข้อมูลนั้นๆได้โดยอัตโนมัติ โดยจะเลือกความเร็วสูงสุดที่เป็นไปได้ในการส่งข้อมูลระหว่างเครื่องโทรสารทั้ง 2 ด้าน

รายละเอียดของเครื่องโทรสารกลุ่ม 3

โครงสร้างของเครื่องโทรสารกลุ่ม 3



รูปที่ 3.1 โครงสร้างของเครื่องโทรสารกลุ่ม 3

โครงสร้างของเครื่องโทรสารกลุ่ม 3 ซึ่งแสดงดังบล็อคไดอะแกรมประกอบด้วย

-สแกนเนอร์แบบ ซีซีดี (CCD:Chargd Coupled Device) ทำหน้าที่ในการเปลี่ยนภาพให้เป็นไฟฟ้าโดยอาศัยโฟโต้เซนเซอร์ (photosensors) เป็นตัวตรวจจับความเข้มของจุดแต่ละจุดแล้วสร้างพัลส์ (pulse) ขึ้นมาแทนจุดแต่ละจุดนั้น ซึ่งใน 1 เส้นจะทำการตรวจจับ 1728 จุด ทำให้ได้พัลส์จำนวน 1728 ลูกเช่นกัน

-ตัวแปลงสัญญาณอนาล็อกเป็นดิจิตอล (A/D Converter) จะทำการเปลี่ยนสัญญาณอนาล็อกให้เป็นดิจิตอล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

-ลดขนาดข้อมูล อาจจะเป็นแบบโมดิฟายด์ ฮัฟฟ์แมน (HM:Modified Huffman),โมดิฟายด์ รีดดิ้ง(MR:Modified read) หรือ โมดิฟายด์ โมดิฟายด์ รีดดิ้ง (MMR:Modified Modified Reading) ซึ่งจะทำการลดจำนวนบิทของข้อมูลให้น้อยลง โดยการใช้รหัสตัวใหม่แทนข้อมูลเดิมทำให้ข้อมูลมีขนาดเล็กลงกว่าเดิมช่วยให้สามารถส่งข้อมูลได้เร็วขึ้นด้วย

-โมเด็ม (Modem) จะทำการเปลี่ยนสัญญาณที่ถูกเข้ารหัสเป็นดิจิตอลเรียบร้อยแล้ว ให้เป็นสัญญาณอนาล็อกอีกครั้งหนึ่ง เพื่อให้สามารถส่งไปบนสายโทรศัพท์

ทางด้านรับจะมีโมเด็มทำการเปลี่ยนสัญญาณอนาล็อกที่รับมาจากทางด้านส่ง ให้เป็นสัญญาณดิจิตอล แล้วส่งเข้าไปยังส่วนขยายข้อมูล ซึ่งจะทำการเปลี่ยนรหัสต่างๆ ให้อยู่ในรูปของจุดขาว,ดำ เพื่อให้เครื่องพิมพ์สามารถแสดงผลออกมาได้เหมือนต้นฉบับเดิม

การเข้ารหัส

ในเครื่องโทรสารกลุ่ม 3 กำหนดให้มีการเข้ารหัสเพื่อให้ข้อมูลที่ต้องส่งผ่านระบบสื่อสัญญาณมีจำนวนน้อยลง ช่วยให้การส่งเร็วขึ้น การเข้ารหัสมีให้เลือก 2 แบบคือ 1มิติ, 2มิติ แต่จะแสดงเฉพาะแค่รายละเอียดของการเข้ารหัสแบบ 1 มิติที่ใช้ในการทำโครงงานเรื่องนี้

การเข้ารหัสแบบ 1 มิติ

การเข้ารหัสแบ 1 มิติ ใช้หลักการเข้ารหัสแบบ โมดิฟายด์ ฮัฟฟ์แมน (MH:Modefied Huffman) ซึ่งข้อมูลที่มีค่าความน่าจะเป็นสูง(เกิดขึ้นบ่อย) จะถูกแทนด้วยรหัสที่มีความยาวน้อย ส่วนข้อมูลที่มีความน่าจะเป็นต่ำ(เกิดขึ้นน้อย) จะถูกแทนด้วยรหัสที่มีความยาวมาก ซึ่งในมาตรฐานของ CCITT ได้กำหนดรหัสที่ใช้ในการแทนมาอย่างแน่นอนแล้วดังตารางที่ 3.2, 3.3 และ 3.4

ในการเข้ารหัสจะเริ่มขึ้นด้วยจุดขาวก่อนเสมอ กรณีที่จุดแรกเป็นจุดดำ รหัสของจุดขาวที่มีความยาวเท่ากับ 0 จะถูกส่งออกไป ข้อมูลจุดขาวหรือจุดดำสามารถมีความยาวได้ถึง 1728 พิคเซล ซึ่งเป็นความยาวสูงสุดสำหรับ 1 เส้นสแกนมาตรฐาน รหัสที่ใช้แทนข้อมูลนี้มี 2 ชนิดคือ

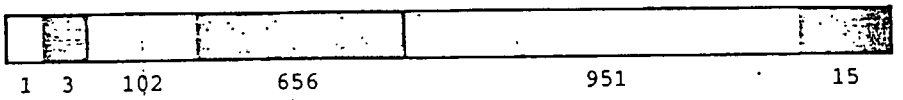
1. เทอร์มิเนตติ้งโค้ด (Terminating Code)ใช้แทนค่าพิคเซลที่มีความยาวตั้งแต่ 0 ถึง 63
2. เมคอัพโค้ด (Make-up Code) ใช้แทนค่าพิคเซลที่มีความยาวเป็นจำนวนเท่าของ 64 จนถึง1728

กรณีของพิคเซลซึ่งมีความยาวตั้งแต่ 64-1728 จะแทนด้วยเมคอัพโค้ดซึ่งมีค่าเท่ากับหรือน้อยกว่าความยาวของพิคเซลก่อน โดยความยาวของพิคเซลที่เหลือจากแทนโดยเมคอัพโค้ด จะมีค่าไม่เกิน 63 พิคเซล ซึ่งสามารถแทนค่าที่เหลือด้วยเทอร์มิเนตติ้งโค้ดได้

ตัวอย่างเช่น ถ้าพิคเซลซึ่งเป็นจุดขาวมีความยาวเท่ากับ 500 จะประกอบไปด้วยเมคอัพโค้ดความยาว 448 (0110 0100) และตามด้วยเทอร์มิเนตติ้งโค้ดความยาว 52 (0101 0101) และในกรณีที่จำนวนพิคเซลมีมากกว่า 1728 จุด ซึ่งใช้เมื่อขนาดของกระดาษใหญ่กว่า A4 สามารถแทนได้ด้วยรหัสในตารางที่ 3.4

ตัวอย่างการเข้ารหัสแบบ โมดิฟายด์ ฮัฟฟ์แมน แสดงดังรูปต่อไปนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



Color	Run Length	Makeup Codeword	Terminating Codeword
white	1	-	00 0111
black	3	-	10
white	102	1 1011	0001 0111
black	656	0 0000 0100 1010	00 0000 0111
white	951	0 1101 0011	0101 1000
black	15	-	0 0001 1000

รูปที่ 3.2 แสดงการเข้ารหัสแบบ โมดิฟายด์ ฮัฟฟ์แมน (Modified Huffman)

-จุดสิ้นสุดของเส้นสแกน (End-of-lind:EOL)

เมื่อสิ้นสุดการสแกนข้อมูลครบ 1 เส้น ข้อมูลจะต้องตามด้วยรหัส EOL เพื่อแสดงจุดสิ้นสุดของเส้นสแกน นอกจากนี้รหัส EOL ยังใช้นำหน้าเส้นสแกนเส้นแรกของแต่ละหน้าด้วย

รูปแบบของ EOL คือ 0000 0000 0001

-บิทเติม (Fill)

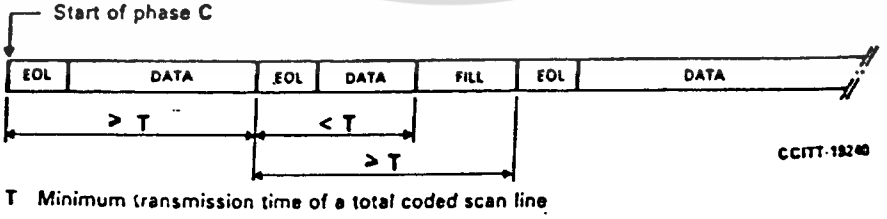
เป็นบิทที่เติมขึ้นระหว่างบิทของข้อมูลกับบิทสิ้นสุด เพื่อให้เวลาในการส่งข้อมูลในเส้นสแกนนั้นไม่ต่ำกว่าเวลาต่ำสุด (minimum time) ที่เครื่องพิมพ์ด้านรับสามารถทำงานได้ทัน

รูปแบบของบิทเติมคือ 0 ซึ่งจำนวนบิทสามารถแปรค่าได้

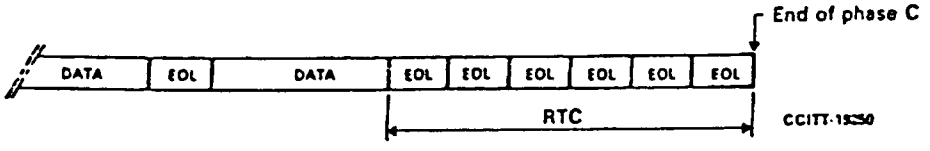
-รีเทิร์นทูคอนโทรล (Return to control:RTC)

เมื่อจบการส่งข้อมูลแต่ละแผ่นจะต้องตามด้วย รีเทิร์นทูคอนโทรล (RTC) เพื่อเป็นการกลับเข้าสู่โหมดการควบคุมเครื่องโทรสาร

รูปแบบของ RTC คือ EOL...EOL...EOL(EOL 6 ชุด)



รูปที่ 3.3 แสดงเส้นสแกนหลายเส้นที่จุดเริ่มต้นของหน้า



รูปที่ 3.4 แสดง รีเทิร์นทูคอนโทรล (RTC)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

White run length	Code word	Black run length	Code word
0	00110101	0	0000110111
1	000111	1	010
2	0111	2	11
3	1000	3	10
4	1011	4	011
5	1100	5	0011
6	1110	6	0010
7	1111	7	00011
8	10011	8	000101
9	10100	9	000100
10	00111	10	0000100
11	01000	11	0000101
12	001000	12	0000111
13	000011	13	00000100
14	110100	14	00000111
15	110101	15	000011000
16	101010	16	0000010111
17	101011	17	0000011000
18	0100111	18	0000001000
19	0001100	19	00001100111
20	0001000	20	00001101000
21	0010111	21	00001101100
22	0000011	22	00000110111
23	0000100	23	00000101000
24	0101000	24	00000010111
25	0101011	25	00000011000
26	0010011	26	000011001010
27	0100100	27	000011001011
28	0011000	28	000011001100
29	00000010	29	000011001101
30	00000011	30	000001101000
31	00011010	31	000001101001
32	00011011	32	000001101010
33	00010010	33	000001101011
34	00010011	34	000011010010
35	00010100	35	000011010011
36	00010101	36	000011010100
37	00010110	37	000011010101
38	00010111	38	000011010110
39	00101000	39	000011010111
40	00101001	40	000001101100
41	00101010	41	000001101101
42	00101011	42	000011011010
43	00101100	43	000011011011
44	00101101	44	000001010100
45	00000100	45	000001010101
46	00000101	46	000001010110
47	00001010	47	000001010111
48	00001011	48	000001100100
49	01010010	49	000001100101
50	01010011	50	000001010010
51	01010100	51	000001010011
52	01010101	52	000000100100
53	00100100	53	000000110111
54	00100101	54	000000111000
55	01011000	55	000000100111
56	01011001	56	000000101000
57	01011010	57	000001011000
58	01011011	58	000001011001
59	01001010	59	000000101011
60	01001011	60	000000101100
61	00110010	61	000001011010
62	00110011	62	000001100110
63	00110100	63	000001100111

ตารางที่ 3.2 เทอร์มิเนตติ้งโค้ด (Terminating codes)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

White run lengths	Code word	Black run lengths	Code word
64	11011	64	0000001111
128	10010	128	000011001000
192	010111	192	000011001001
256	0110111	256	000001011011
320	00110110	320	000000110011
384	00110111	384	000000110100
448	01100100	448	000000110101
512	01100101	512	0000001101100
576	01101000	576	0000001101101
640	01100111	640	0000001001010
704	011001100	704	0000001001011
768	011001101	768	0000001001100
832	011010010	832	0000001001101
896	011010011	896	0000001110010
960	011010100	960	0000001110011
1024	011010101	1024	0000001110100
1088	011010110	1088	0000001110101
1152	011010111	1152	0000001110110
1216	011011000	1216	0000001110111
1280	011011001	1280	0000001010010
1344	011011010	1344	0000001010011
1408	011011011	1408	0000001010100
1472	010011000	1472	0000001010101
1536	010011001	1536	0000001011010
1600	010011010	1600	0000001011011
1664	011000	1664	0000001100100
1728	010011011	1728	0000001100101
EOL	000000000001	EOL	000000000001

ตารางที่ 3.3 เมคอัพโค้ด (Make-up codes)

Run length (black and white)	Make-up codes
1792	00000001000
1856	00000001100
1920	00000001101
1984	000000010010
2048	000000010011
2112	000000010100
2176	000000010101
2240	000000010110
2304	000000010111
2368	000000011100
2432	000000011101
2496	000000011110
2560	000000011111

ตารางที่ 3.4 เมคอัพโค้ด (Make-up codes) ที่มีความยาวมากกว่า 1728 พิคเซล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 4

ขบวนการรับส่งข้อมูลของเครื่องโทรสารกลุ่มที่ 3

ตามมาตรฐานของ CCITT T.30 ได้แบ่งขั้นตอนในการรับส่งข้อมูลของเครื่องโทรสารกลุ่มที่ 3 ออกเป็น 5 เฟส

- เฟส A ขั้นตอนการติดต่อ (Call establishment)
- เฟส B ขั้นตอนการส่งข่าวสาร (Pre-message procedure)
- เฟส C ขั้นตอนการส่งข้อมูล (Message transmission)
- เฟส D ขั้นตอนหลังการส่งข่าวสาร (Post-message procedure)
- เฟส E ปลดสาย (Call release)

สำหรับขั้นตอนในการส่งโทรสารทั้ง 5 ขั้นตอนได้แสดงดังในรูปที่ 4.1

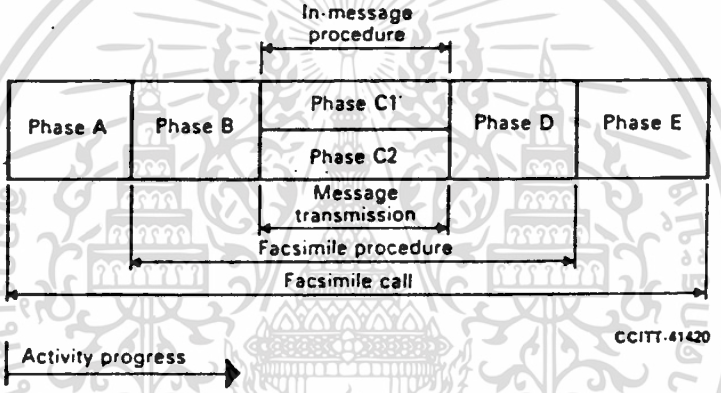


FIGURE 1/T.30

รูปที่ 4.1 แสดงขบวนการในการรับส่งข้อมูลของเครื่องโทรสารกลุ่มที่ 3

ช่วงการทำงานตั้งแต่เฟส A-E เรียกว่า แฟกซิไมล์ เซชชั่น (Facsimile session)

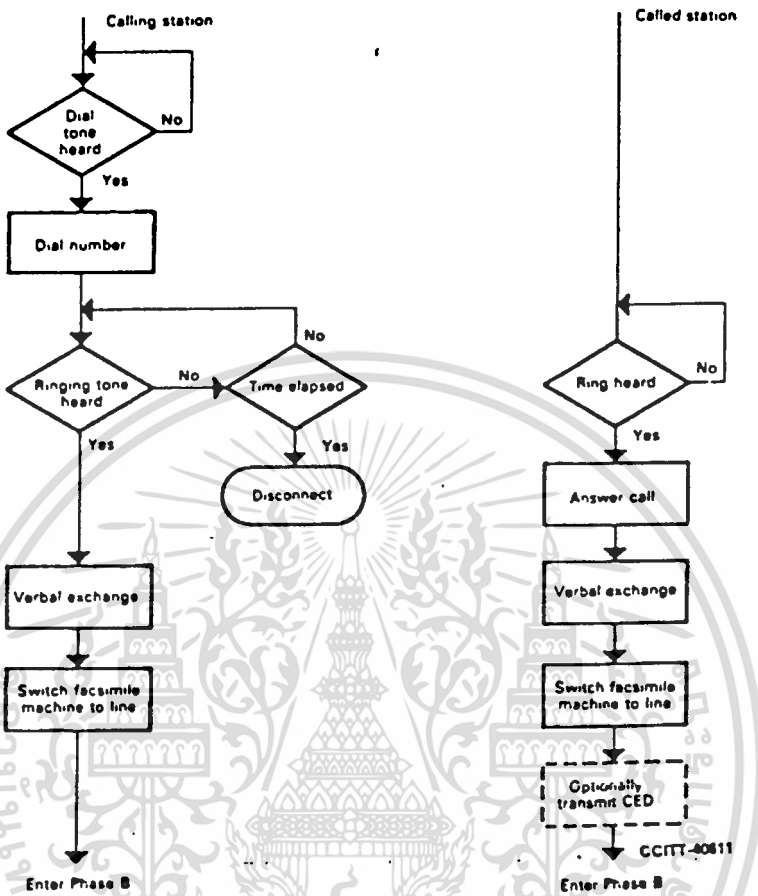
ช่วงการทำงานตั้งแต่เฟส B-D เรียกว่า แฟกซิไมล์ โพรซีเดอร์ (Facsimile procedure)

เฟส A ขั้นตอนการติดต่อ (Call establishment)

หรือเรียกว่า การเริ่มต้นการเรียก เป็นขั้นตอนที่จะเชื่อมโยงเครื่องโทรสารต้นทางเข้ากับเครื่องโทรสารปลายทาง โดยผ่านทางข่ายสายโทรศัพท์ อาจเป็นแบบควบคุมโดยโอเปอเรเตอร์ ให้โอเปอเรเตอร์ติดต่อกันได้แล้วจึงทำการส่งข้อมูล หรืออาจเป็นแบบอัตโนมัติ คือรับหรือส่งเอกสารโดยอัตโนมัติที่ด้านใดด้านหนึ่ง หรือสามารถรับอัตโนมัติและส่งอัตโนมัติได้ทั้งสองด้าน ดังรูป 4.2

ในขั้นแรกเครื่องโทรสารทางด้านผู้เรียกจะทำการเรียกไปยังเครื่องโทรสารปลายทางที่ต้องการติดต่อด้วยสัญญาณ ไดออลโทน (Dial tone) และส่งสัญญาณ CNG (Calling tone) ไป เพื่อเริ่มการติดต่อ ส่วนทางด้านถูกเรียกจะตอบสนองต่อการเรียกนั้นโดยส่งสัญญาณ CED (Called station identification) ซึ่งเป็นสัญญาณความถี่ 2100 เฮิรตซ์ กลับไปยังเครื่องโทรสารด้านผู้เรียกทุกๆ 3 วินาที

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4.2 แสดงขั้นตอนการติดต่อระหว่างเครื่องโทรสารระบบอัตโนมัติ

เฟส B ขั้นตอนการส่งข่าวสาร (Pre-message procedure)

เป็นขั้นตอนที่เครื่องโทรสารต้นทางเชื่อมโยงกับเครื่องโทรสารปลายทางแล้วต้องทำการตรวจสอบคุณสมบัติและความสามารถในการรับและส่งให้ตรงกัน ก่อนที่จะเริ่มส่งข่าวสาร มี 2 ขั้นตอนคือ

1. ส่วนการตรวจสอบสัญญาณ ทำการส่งสัญญาณเพื่อบอกให้ทราบว่าเป็นเครื่องโทรสารอยู่ในกลุ่มใด (Group identification) , พร้อมที่จะรับข้อมูลหรือไม่ (Confirmation for reception) , หมายเลขโทรศัพท์ของเครื่องโทรสารต้นทาง(Subscriber identification) รวมถึงความสามารถอื่นๆที่ไม่ใช่มาตรฐานของ CCITT (Nonstandard facilities identification) เป็นทางเลือก

2. ส่วนคำสั่ง ส่งสัญญาณเพื่อตรวจสอบระบบสื่อสัญญาณ DIS (Digital identification signal) ไปยังด้านผู้เรียกเพื่อถึงแสดงคุณสมบัติและความสามารถต่างๆ ของเครื่อง ซึ่งเมื่อทางด้านผู้เรียกได้รับสัญญาณนี้แล้วก็จะส่งสัญญาณ DCS (Digital command signal) มายังด้านผู้ถูกเรียกเพื่อแจ้งให้ทราบว่าได้เลือกใช้คุณสมบัติและความสามารถใดของเครื่องที่ถูกเรียกเป็นข้อกำหนดในการรับส่งข้อมูลครั้งนี้ ซึ่ง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เครื่องโทรสารด้านถูกเรียกก็จะต้องเลือกโหมดของตนเองให้เป็นไปตามข้อกำหนดนั้น จากนั้นเครื่องโทรสารด้านผู้เรียกจะส่งสัญญาณความเร็วสูงที่เรียกว่า เทรนนิ่ง (training) เพื่อตรวจสอบระบบสื่อสาร ซึ่งหลังจากเครื่องโทรสารด้านถูกเรียกได้รับสัญญาณเทรนนิ่งและเมื่อตรวจสอบเรียบร้อยแล้ว ก็จะส่งสัญญาณ CFR (Confirmation to receive) กลับไปยังเครื่องโทรสารต้นทางเพื่อยืนยันความพร้อมในการรับข้อมูล

เฟส C ขั้นตอนการส่งข้อมูล (Message transmission)

เฟส C ประกอบด้วย

เฟส C1 ส่งสัญญาณควบคุมข่าวสาร

เป็นขั้นตอนส่งสัญญาณควบคุมเพื่อการซิงโครไนซ์ การตรวจสอบข้อผิดพลาด การแก้ไขข้อผิดพลาด และตรวจสอบสถานะของระบบสื่อสัญญาณในขณะที่กำลังส่งข่าวสาร

เฟส C2 ส่งข่าวสาร

เป็นขั้นตอนการส่งข่าวสาร โดยมีรูปแบบตามแต่ชนิดของเครื่องโทรสารแต่ละกลุ่ม

เฟส D ขั้นตอนหลังการส่งข่าวสาร (Post-message procedure)

เครื่องโทรสารด้านผู้เรียกจะส่งสัญญาณ RTC (Return to control) ไปยังด้านถูกเรียกเพื่อทำการปรับโมเด็มให้กลับไปอยู่ที่ความเร็ว 300 บิท/วินาที จากนั้นจึงจะส่งสัญญาณ EOP (End of page) ซึ่งทางด้านถูกเรียกก็จะส่งสัญญาณ MCF (Message confirmation) กลับมาให้ทางด้านต้นทางเพื่อยืนยันการรับข้อมูลที่ถูกต้องเรียบร้อย ในกรณีที่มีการส่งข้อมูลหลายๆ หน้า เมื่อจบหน้าแรกจะมีการส่งสัญญาณอื่นๆที่แสดงว่ามีข้อมูลที่จะส่งต่อไปอีก แทนการส่งสัญญาณ EOP เช่น EOM (End of message) หรือ MPS (Multi page signal)

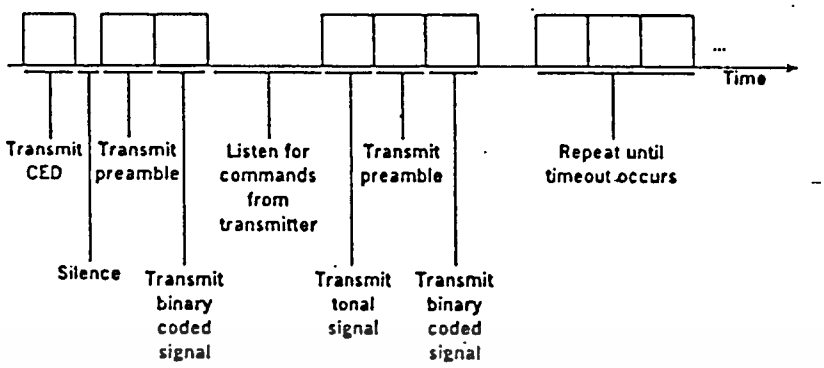
เฟส E ปลดสาย (Call release)

เป็นขั้นตอนเลิกการติดต่อระหว่างเครื่องโทรสารต้นทาง และเครื่องโทรสารปลายทาง อาจเป็นแบบใช้มือ (manual) หรือ แบบอัตโนมัติ (automatic) ก็ได้ โดยเครื่องโทรสารด้านผู้เรียกจะส่งสัญญาณ DCN (disconnect) ไปยังด้านปลายทางเพื่อยกเลิกการติดต่อ

ขั้นตอนการส่งโทรสาร(The Facsimile Procedure)

เครื่องโทรสารทางด้านรับทุกเครื่อง จะบอกคุณสมบัติของเครื่องให้ทางด้านผู้เรียกทราบได้ โดยการส่งขบวนสัญญาณและรหัสฐาน 2 ดังแสดงในรูปที่ 4.3

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4.3 รูปแบบสัญญาณซึ่งแสดงถึงคุณสมบัติของเครื่องโทรสาร
ที่ส่งโดยเครื่องโทรสารด้านรับไปยังเครื่องโทรสารด้านผู้เรียก

สัญญาณแรกที่ส่งออกไปคือ CED ประมาณ 1.8 -2.5 วินาที หลังจากเครื่องรับถูกต่อเข้ากับสายเรียบร้อย จะส่งสัญญาณความถี่ประมาณ 2100 เฮิรตซ์ โดยสัญญาณนี้จะถูกส่งอย่างน้อย 2.6 วินาที แต่ไม่เกิน 4 วินาที หลังจากนั้นเครื่องโทรสารด้านถูกเรียกจะหยุดส่งสัญญาณ 75 มิลลิวินาที ก่อนจะส่งสัญญาณตัวต่อไป

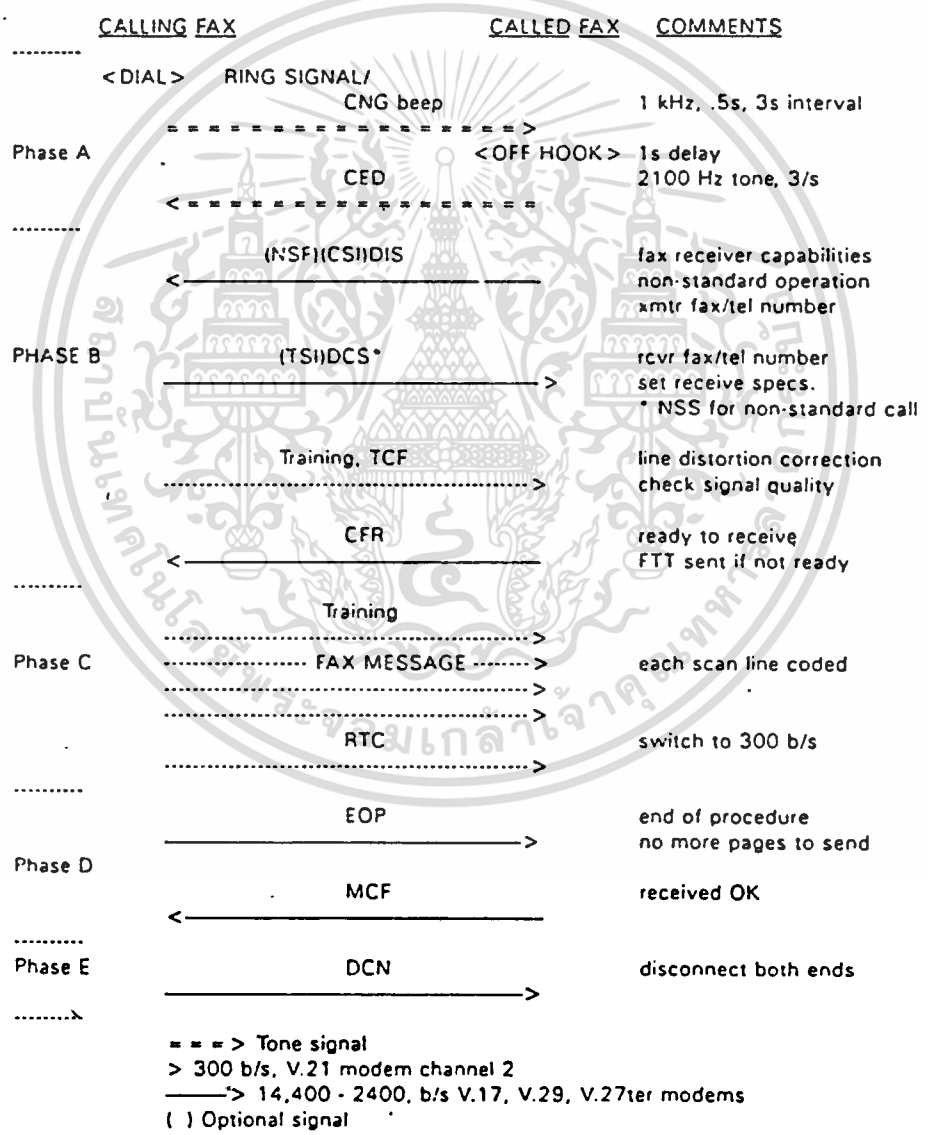
สัญญาณพรีแอมเบิล (preamble) เป็นสัญญาณที่จะส่งนำหน้าสัญญาณรหัสฐาน 2 ตัวอื่นๆเสมอทุกครั้งที่เริ่มต้นการส่งข้อมูลครั้งใหม่ สัญญาณพรีแอมเบิลเป็นสิ่งที่จะรับประกันว่าองค์ประกอบต่างๆในช่องสื่อสารเป็นไปตามกำหนด สำหรับเครื่องโทรสารกลุ่ม 3 ตามมาตรฐานได้กำหนดอัตราส่งสัญญาณพรีแอมเบิลไว้ 300 บิทต่อวินาที หรือ 2400 บิทต่อวินาที เป็นทางเลือก สัญญาณรหัสฐาน 2 ที่ถูกส่งตามหลังพรีแอมเบิลนี้จะส่งโดยใช้โครงสร้าง HDLC เฟรม (high level data link control) ซึ่งโครงสร้างพื้นฐานของ HDLC เฟรมจะได้กล่าวถึงในตอนหลัง ในระหว่างเฟส B สัญญาณดังกล่าวนี้จะนำข่าวสารพื้นฐาน และข่าวสารที่แสดงความสามารถของเครื่องโทรสารด้านรับ ซึ่งเครื่องโทรสารด้านส่งจะไม่สนใจข่าวสารที่แสดงถึงความสามารถที่มันไม่รู้จัก

หลังจากส่งสัญญาณรหัสฐาน 2 เครื่องโทรสารด้านรับจะรอรับสัญญาณตอบรับจากเครื่องโทรสารด้านส่งประมาณ 3 วินาที ถ้าไม่มีสัญญาณตอบรับกลับมา เครื่องโทรสารด้านรับจะส่งสัญญาณที่สอดคล้องกับแอพพาราตัส (apparatus) ของเครื่องโทรสารกลุ่ม 1 และ 2 ดังแสดงใน มาตรฐาน T.2 และ T.3 ทั้งนี้ขึ้นอยู่กับความสามารถของเครื่องโทรสารด้านรับ

ประมาณ 1.5 วินาที เครื่องโทรสารด้านรับจะส่งสัญญาณความถี่ 1650 เฮิรตซ์ สำหรับเครื่องโทรสารกลุ่ม 1 หรือความถี่ 1850 เฮิรตซ์ สำหรับกลุ่ม 2 ถ้าเครื่องสามารถรองรับได้ทั้งกลุ่ม 1 และกลุ่ม2จะส่งสัญญาณความถี่ 1650 เฮิรตซ์เป็นเวลา 1.5 วินาทีตามด้วยความถี่ 1850 เฮิรตซ์ ทันทีเป็นเวลา 0.75 วินาที ดังแสดงในรูปที่ 4.3 ซึ่งสัญญาณพรีแอมเบิล, รหัสฐาน 2 , การหยุดชั่วขณะและสัญญาณความถี่ต่างๆ จะถูกส่งซ้ำๆ จนกว่าเครื่องโทรสารด้านผู้เรียกจะส่งสัญญาณตอบรับหรือจนหมดเวลาของไทม์เอาท์

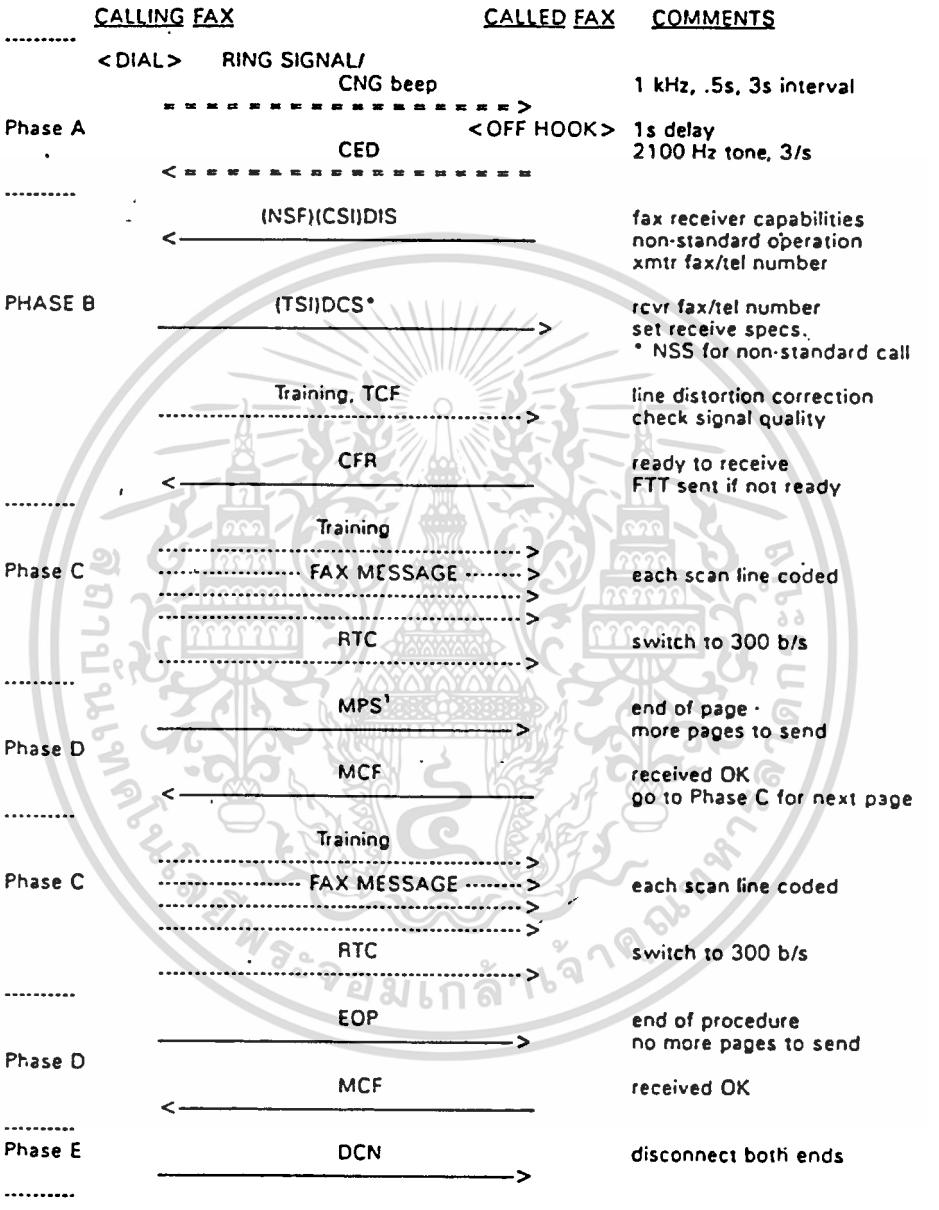
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สำหรับรูปแบบของสัญญาณในเฟส C แสดงไว้ในมาตรฐาน T.2 , T.3 และ T.4 และสัญญาณในเฟส D คำสั่งหลังการส่งข่าวสารจะถูกส่งจากด้านผู้ส่งไปยังด้านรับ และขึ้นอยู่กับความสามารถของเครื่องรับ รวมทั้งคุณสมบัติของการสื่อสาร ด้านส่งอาจจะส่งข้อความจำนวนหนึ่งต่อไปอีกก็ได้ เช่น คำสั่งแสดงการสิ้นสุดข้อความอาจจะถูกส่งออกไปแล้วเครื่องส่งกลับเข้าเฟส B อีกครั้งก็ได้ ,คำสั่งมัลติเพจ (multipage) แสดงถึงว่าเครื่องส่งจะกลับเข้าสู่เฟส C เป็นต้น ดังรูป 4.4 4.5 แสดงการรับส่งข้อมูลระหว่างเครื่องโทรสาร 1 หน้า และ มากกว่า 1 หน้าตามลำดับ



รูป 4.4 แสดงขั้นตอนการรับส่งข้อมูลระหว่างเครื่องโทรสาร จำนวน 1 หน้า

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



1. Some Group 3 fax machines use EOM incorrectly. If EOM is sent instead of MPS, the next page starts from Phase B, not C. More time is then needed to retrain. If EOM is sent instead of EOP, the fax machine will remain on line for 35 seconds and indicate an error.

รูปที่ 4.5 แสดงขั้นตอนการรับส่งข้อมูลระหว่างเครื่องโทรสารซึ่งมีจำนวนมากกว่า 1 หน้า

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เครื่องโทรสารต้นทาง	เครื่องโทรสารปลายทาง
1. ส่งสัญญาณ CNG เพื่อเริ่มการติดต่อ	
	2. ส่งสัญญาณ CED ตอบรับการติดต่อ
	3. ส่งสัญญาณ DIS
4. รับสัญญาณ DIS และคุณสมบัติของเครื่องปลายทาง	
5. ส่งสัญญาณ DCS	
	6. รับสัญญาณ DCS และจ้อกำหนดการส่งข้อมูล
	7. เลือกโหมดตามข้อกำหนดรับข้อมูล
8. ส่งสัญญาณ training เพื่อทดสอบคุณภาพของสาย	
	9. รับสัญญาณ training และตรวจสอบว่ามีความผิดพลาดหรือไม่
11. รับสัญญาณ CFR เข้าสู่โหมดการส่งข้อมูล	10. ถ้าไม่มีความผิดพลาดส่งสัญญาณ CFR
12. ส่งข้อมูลภาพ	
14. เมื่อส่งข้อมูลภาพหมดแล้ว ส่งสัญญาณเพื่อแสดงการสิ้นสุดของข้อมูลเช่น EOM, EOP, MPS เป็นต้น	13. รับข้อมูลภาพ
	15. รับสัญญาณแสดงการสิ้นสุดของข้อมูลเช่น EOM แล้วส่งสัญญาณ MCF ไปเพื่อยืนยันว่ารับข้อมูลเรียบร้อยแล้ว
16. รับสัญญาณ MCF แล้วส่ง DCN เพื่อยกเลิกการติดต่อ	

ตารางที่ 4.1 แสดงขั้นตอนการรับส่งข้อมูลเมื่อเครื่องโทรสารต้นทางเป็นผู้ส่งเอกสารให้ปลายทาง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เครื่องโทรสารต้นทาง	เครื่องโทรสารปลายทาง
1. ส่งสัญญาณ CNG เริ่มการติดต่อ	
	2. ส่งสัญญาณ CED ตอบรับการเรียก
4. รับสัญญาณ DIS และคุณสมบัติของเครื่องปลายทาง	3. ส่งสัญญาณ DIS
5. ส่งสัญญาณ DTC	
	6. รับสัญญาณ DTC และคุณสมบัติของเครื่องต้นทาง
	7.กำหนดคุณสมบัติที่จะใช้รับส่งข้อมูลที่ส่งไปกับสัญญาณ DCS
8. รับสัญญาณ DCS และเลือกโหมดตามข้อกำหนดรับข้อมูล	
	9. ส่งสัญญาณ training เพื่อตรวจสอบคุณภาพสาย
10. รับสัญญาณ training ตรวจสอบว่ามีความผิดพลาดหรือไม่	
11. ถ้าไม่มีความผิดพลาดส่งสัญญาณ CFR	
	12. รับสัญญาณ CFR เข้าสู่โหมดการส่งข้อมูล
	13. ส่งข้อมูล
14. รับข้อมูล	
	15. เมื่อส่งข้อมูลหมดแล้ว ส่งสัญญาณเพื่อแสดงว่าสิ้นสุดข้อมูลแล้ว ได้แก่ EOM, EOP, MPS เป็นต้น
16. รับสัญญาณการสิ้นสุดของข้อมูลเช่น EOM แล้วส่งสัญญาณ MCF เพื่อยืนยันการรับข้อมูล	
17. ส่งสัญญาณ DCN ยกเลิกการติดต่อ	

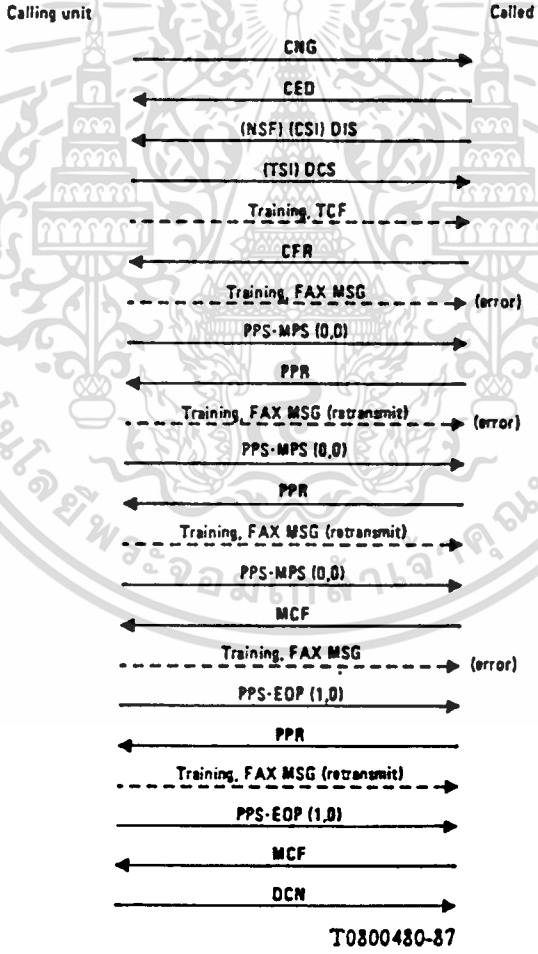
ตารางที่ 4.2 แสดงขั้นตอนการรับส่งข้อมูลเมื่อเครื่องโทรสารปลายทางเป็นผู้ส่งเอกสาร

<u>หมายเหตุ</u> ในกรณีที่เครื่องโทรสารกลุ่ม 3 บางเครื่องมีความสามารถและคุณสมบัติพิเศษที่นอกเหนือไปจากมาตรฐาน จะมีการส่งสัญญาณ NSF (Nonstardard facilities) และ CSI (Called subscriber identification) นำหน้าสัญญาณ DIS ซึ่ง NSF เป็นสัญญาณที่บอกให้ทราบว่า เครื่องโทรสาร

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ด้านรับมีความสามารถพิเศษ นอกเหนือไปจากมาตรฐานตามข้อกำหนด มาตรฐาน T. seies หลังจากด้านส่งได้รับสัญญาณ NSF แล้วจะส่งสัญญาณ NSS (Nonstard setup) กลับไปยังด้านรับเพื่อกำหนดให้เครื่องโทรสารด้านรับอยู่ในระบบคำสั่งพิเศษ อย่างไรก็ตาม แม้ว่าเครื่องโทรสารส่วนหนึ่งจะมีความสามารถพิเศษเฉพาะกลุ่มที่ผลิตจากบริษัทเดียวกัน แต่จะมีความสามารถสนองตอบต่อคำสั่งมาตรฐานได้เช่นกัน ดังรูป 4.6 แสดงการติดต่อที่มีความสามารถพิเศษนอกเหนือจากมาตรฐาน

สัญญาณ CSI จะเป็นการส่งรหัสสัญญาณหมายเลขโทรศัพท์ของทางด้านถูกเรียก ซึ่งเครื่องโทรสารบางเครื่องจะแสดงหมายเลขโทรศัพท์ให้เห็นได้อย่างชัดเจน ทำให้ผู้ใช้สามารถตรวจสอบได้ว่า ส่งข่าวสารไปยังเครื่องที่ต้องการหรือไม่ จากนั้นเครื่องโทรสารด้านส่งจะบันทึกหมายเลขดังกล่าวไว้ และส่งหมายเลขของตนไปยังเครื่องโทรสารด้านรับโดยสัญญาณ TSI (Transmit subscriber idenfication)



รูปที่ 4.6 แสดงขั้นตอนการติดต่อของเครื่องโทรสารต้นทางและปลายทางที่มีความสามารถพิเศษนอกเหนือไปจากมาตรฐาน ในการรับส่งข้อมูล และเกิดความผิดพลาดขึ้นในการติดต่อ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ในกรณีที่มีการรับส่งข้อมูลแบบ โพลลิ่ง (polling) ซึ่งเป็นลักษณะที่มีการส่งสัญญาณจากเครื่องโทรสารหลายๆเครื่อง เข้ามายังเครื่องโทรสารที่ศูนย์กลางนั้น หลังจากที่เครื่องโทรสารด้านผู้เรียกซึ่งอยู่ที่ศูนย์กลางได้รับสัญญาณตอบรับ DIS จากทางด้านถูกเรียกแล้ว ก็จะส่งสัญญาณ NSC (Nonstandard command) ไปและเครื่องโทรสารเครื่องอื่นๆจะมาติดต่อร่วมกัน จนกว่าจะได้ส่งข่าวสารตามกำหนดให้แก่เครื่องโทรสารศูนย์กลางเรียบร้อย สัญญาณ CIG (Calling subscriber identification) จะถูกส่งไปแทนสัญญาณ TSI โดยมีรหัสของหมายเลขโทรศัพท์ส่งไปด้วย ซึ่งรหัสดังกล่าวนี้ เครื่องโทรสารทางด้านส่งจะใช้ในการตรวจสอบว่าได้ส่งข่าวสารไปยังเครื่องโทรสารเครื่องที่ต้องการหรือไม่

คำสั่งและสัญญาณต่างๆที่ใช้ในการติดต่อระหว่างเครื่องโทรสารแสดงได้ดังตาราง 4.3,4.4 ต่อไปนี้

Abbreviation	Function	Signal format	Reference
CED	Called station identification	2100 Hz	4.3.3.2
CFR	Confirmation to receive	X010 0001	5.3.6.1.4, 1)
		1850 or 1650 Hz for 3s	4.3.1.2
CRP	Command repeat	X101 1000	5.3.6.1.8, 2)
CIG	Calling subscriber identification	1000 0010	5.3.6.1.2, 2)
CNG	Calling tone	1100 Hz for 500 ms	4.3.3.3
CSI	Called subscriber identification	0000 0010	5.3.6.1.1, 2)
CTC	Continue to correct	X100 1000	A.4.1
CTR	Response to continue to correct	X010 0011	A.4.2
DCN	Disconnect	X101 1111	5.3.6.1.8, 1)
DCS	Digital command signal	X100 0001	5.3.6.1.3, 1)
DIS	Digital identification signal	0000 0001	5.3.6.1.1, 1)
DTC	Digital transmit command	1000 0001	5.3.6.1.2, 1)
EOM	End of message	X111 0001	5.3.6.1.6, 1)
		1100 Hz	4.3.2.4
EOP	End of procedure	X111 0100	5.3.6.1.6, 3)
EOR	End of retransmission	X111 0011	A.4.3
ERR	Response for end of retransmission	X011 1000	A.4.4
FCD	Facsimile coded date	0110 0000	A.2.2
FCF	Facsimile control field	–	5.3.6.1
FIF	Facsimile informtion field	–	5.3.6.2
FTT	Failure to train	X010 0010	5.3.6.1.4, 2)
GC	Group command	1300 Hz for 1.5-10.0 s	4.3.2.1
		2100 Hz for 1.5-10.0 s	
GI	Group identification	1650 or 1850 Hz	4.3.1.1
HDLC	High level data link control	–	5.3
LCS	Line conditioning signals	1100 Hz	4.3.2.2
MCF	Message confirmation	X011 0001	4.3.1.3
		1650 or 1850 Hz	
MPS	Multi-page signal	X111 0010	5.3.6.1.6, 2)
NSC	Non-standard facilities command	1000 0100	5.3.6.1.2, 3)
NSF	Non-standard facilities	0000 0100	5.3.6.1.1, 3)
NSS	Non-standard set-up	X100 0100	5.3.6.1.3, 3)

ตารางที่ 4.3 คำสั่งและสัญญาณต่างๆที่ใช้ในการติดต่อระหว่างเครื่องโทรสาร

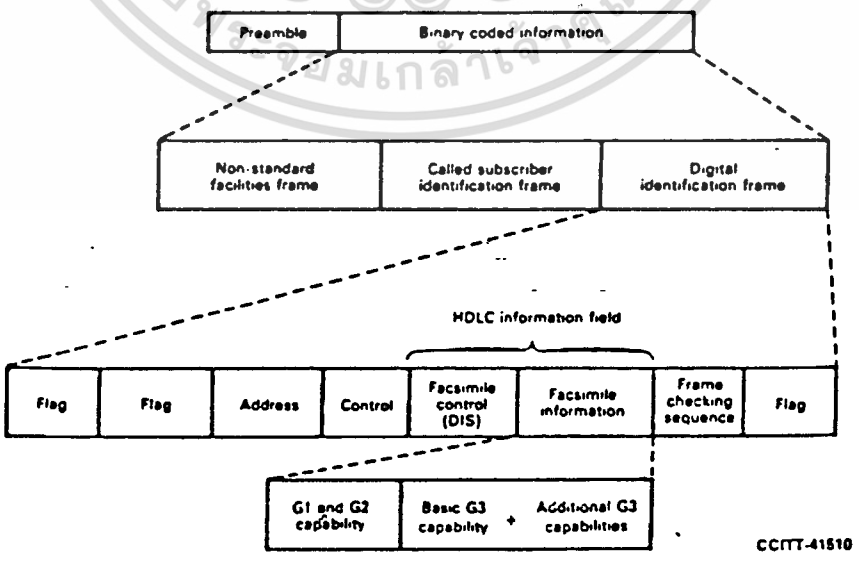
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Abbreviation	Function	Signal format	Reference
PIN	Procedural interrupt negative	X011 0100	5.3.6.1.7, 5)
PIP	Procedural interrupt positive	X011 0101	5.3.6.1.7, 4)
PIS	Procedure interrupt signal	462 Hz for 3 s	4.3.3.1
PPS	Partial page signal	X111 1101	A.4.3
PPR	Partial page request	X011 1101	A.4.4
PRI-EOM	Procedure interrupt-EOM	X111 1001	5.3.6.1.6, 4)
PRI-EOP	Procedure interrupt-EOP	X111 1100	5.3.6.1.6, 6)
PRI-MPS	Procedure interrupt-MPS	X111 1010	5.3.6.1.6, 5)
RCP	Return to control for partial page	0110 0001	A.2.2
RNR	Receive not ready	X011 0111	A.4.4
RR	Receive ready	X111 0110	A.4.3
RTN	Retrain negative	X011 0010	5.3.6.1.7, 3)
RTP	Retrain positive	X011 0011	5.3.6.1.7, 2)
TCF	Training check	Zeros for 1.5 s	5.3.6.1.3, 4)
TSI	Transmitting subscriber identification	X100 0010	5.3.6.1.3, 2)

ตารางที่ 4.4 แสดงคำสั่งและสัญญาณต่างๆที่ใช้ในการติดต่อระหว่างเครื่องโทรสาร

รูปแบบของ HDLC เฟรม (high level data link control)

ในการติดต่อรับส่งข้อมูลระหว่างเครื่องโทรสาร จะมีการใช้ HDLC เฟรม (high level data link control) ช่วยให้การรับส่งข้อมูลเป็นไปอย่างถูกต้องสมบูรณ์ ซึ่งสัญญาณรูปแบบ HDLC เฟรมประกอบไปด้วยส่วนต่างๆดังรูป 4.7 นี้



รูปที่ 4.7 แสดงรูปแบบของสัญญาณ HDLC ที่ใช้ติดต่อระหว่างเครื่องโทรสาร

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1. พรีแอมเบิล (Preamble)

เป็นส่วนแรกของขบวนสัญญาณเสมอเมื่อเป็นการเริ่มต้นการสื่อสาร ซึ่งพรีแอมเบิลจะช่วยให้แน่ใจได้ว่าสัญญาณที่อยู่ถัดมาๆมามีความถูกต้องตามเงื่อนไขที่เราต้องการ รูปแบบของพรีแอมเบิลคือ 0111 1110

2. แฟลก (flag)

มีรูปแบบไบนารี 8 บิท (0111 1110) ทำหน้าที่เป็นตัวชี้จุดเริ่มต้นและจุดสิ้นสุดของเฟรม และเป็นเฟรมซิงโครไนเซชั่น

3.ส่วนของทีอยู่ (address field)

มีรูปแบบไบนารี 8 บิท (1111 1111) ใช้ในการกำหนดการติดต่อในระบบการติดต่อแบบหลายๆ จุด

4. ส่วนควบคุม (control field)

มีรูปแบบไบนารี 8 บิท (1100 X000) ซึ่งสำหรับการส่งข้อมูลของเครื่องโทรสาร X จะมีค่าเป็น 1 เมื่อเป็นเฟรมสุดท้ายในขบวนการจัดการ

5. ส่วนของข้อมูล (information field)

ความยาวของฟิลด์นี้จะขึ้นอยู่กับข้อมูลที่บรรจุอยู่ภายใน ซึ่งได้แก่ คำสั่งควบคุมและข่าวสารที่แลกเปลี่ยนกันระหว่างเครื่องโทรสาร ตามข้อกำหนดของ มาตรฐาน T.30 ได้แบ่งฟิลด์นี้ออกเป็น 2 ส่วนได้แก่

5.1 ส่วนควบคุม FCF (Facsimile Control Field)

เป็นสัญญาณ 8 บิทหรือ 16 บิทแรกของส่วนข้อมูลซึ่ง FCF จะประกอบไปด้วยข้อมูลที่เกี่ยวกับชนิดของข่าวสารที่แลกเปลี่ยนกัน และหน้าที่เฟรม โดยมีรูปแบบดังต่อไปนี้

- บิทแรกจะถูกกำหนดให้เป็น 1 โดยสถานีด้านผู้เรียก
- บิทแรกจะถูกกำหนดให้เป็น 0 โดยสถานีด้านถูกเรียก
- บิทแรกจะมีค่าคงที่ไม่เปลี่ยนแปลงจนกว่าจะเข้าสู่ขั้นตอนก่อนการส่งข่าวสาร (เฟส B)

5.2 ส่วนของข้อมูส FIF (Facsimile Information Field)

ส่วนของข้อมูล FIF จะตามหลังส่วนควบคุม FCF โดยมีขนาด 8 บิท แสดงถึงขบวนการในการรับส่งข้อมูลของเครื่องโทรสารซึ่งได้แก่สัญญาณDIS,DCS,DTC,CSI,CIG,TST,NSC,NSF,VSS,CTC,PPS และPPR โดยรูปแบบของสัญญาณDIS,DTCและDCS แสดงดังตารางที่ 4.6 นี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Bit No.	DIS/DTC	DCS
1	Transmitter – T.2 operation	
2	Receiver – T.2 operation	Receiver – T.2 operation
3	T.2 IOC = 176	T.2 IOC = 176
4	Transmitter – T.3 operation	
5	Receiver – T.3 operation	Receiver – T.3 operation
6	Reserved for future T.3 operation features	
7	Reserved for future T.3 operation features	
8	Reserved for future T.3 operation features	
9	Transmitter – T.4 operation	
10	Receiver – T.4 operation	Receiver – T.4 operation
11, 12 (0,0) (0,1) (1,0) (1,1)	Data signalling rate V.27 *ter* fallback mode V.27 *ter* V.29 V.27 *ter* and V.29	Data signalling rate 2400 bit/s V.27 *ter* 4800 bit/s V.27 *ter* 9600 bit/s V.29 7200 bit/s V.29
13	Reserved for new modulation system	
14	Reserved for new modulation system	
15	Vertical resolution = 7.7 line/mm	Vertical resolution = 7.7 line/mm
16	Two-dimensional coding capability	Two-dimensional coding
17, 18 (0,0) (0,1) (1,0) (1,1)	Recording width capabilities 1728 picture elements along scan line length of 215 mm ± 1% 1728 picture elements along scan line length of 215 mm ± 1% and 2048 picture elements along scan line length of 255 mm ± 1% and 2432 picture elements along scan line length of 303 mm ± 1% 1728 picture elements along scan line length of 215 mm ± 1% and 2048 picture elements along scan line length of 255 mm ± 1% Invalid (see Note 7)	Recording width 1728 picture elements along scan line length of 215 mm ± 1% 2432 picture elements along scan line length of 303 mm ± 1% 2048 picture elements along scan line length of 255 mm ± 1% Invalid
19, 20 (0,0) (0,1) (1,0) (1,1)	Maximum recording length capability A4 (297 mm) Unlimited A4 (297 mm) and B4 (364 mm) Invalid	Maximum recording length A4 (297 mm) Unlimited B4 (364 mm) Invalid

ตารางที่ 4.6 แสดงรูปแบบและข้อกำหนดของสัญญาณ DIS,DTC และ DCS

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Bit No.	DIS/DTC	DCS
21, 22, 23 (0,0,0) (0,0,1) (0,1,0) (1,0,0) (0,1,1) (1,1,0) (1,0,1) (1,1,1)	Minimum scan line time capability at the receiver 20 ms at 3.85 l/mm: $T_{7.7} = T_{3.85}$ 40 ms at 3.85 l/mm: $T_{7.7} = T_{3.85}$ 10 ms at 3.85 l/mm: $T_{7.7} = T_{3.85}$ 5 ms at 3.85 l/mm: $T_{7.7} = T_{3.85}$ 10 ms at 3.85 l/mm: $T_{7.7} = 1/2\ T_{3.85}$ 20 ms at 3.85 l/mm: $T_{7.7} = 1/2\ T_{3.85}$ 40 ms at 3.85 l/mm: $T_{7.7} = 1/2\ T_{3.85}$ 0 ms at 3.85 l/mm: $T_{7.7} = T_{3.85}$	Minimum scan line time 20 ms 40 ms 10 ms 5 ms 0 ms
24	Extend field	Extend field
25	2400 bit/s handshaking	2400 bit/s handshaking
26	Uncompressed mode	Uncompressed mode
27	Error correction mode	Error correction mode
28	Set to "0"	Frame size 0 = 256 octets 1 = 64 octets
29	Error limiting mode	Error limiting mode
30	Reserved for G4 capability on PSTN	Reserved for G4 capability on PSTN
31	Unassigned	
32	Extend field	Extend field
33 (0) (1)	Validity of bits 17, 18 Bits 17, 18 are valid Bits 17, 18 are invalid	Recording width Recording width indicated by bits 17, 18 Recording width indicated by this field bit information
34	Recording width capability 1216 picture elements along scan line length of 151 mm ± 1%	Middle 1216 elements of 1728 picture elements
35	Recording width capability 864 picture elements along scan line length of 107 mm ± 1%	Middle 864 elements of 1728 picture elements
36	Recording width capability 1728 picture elements along scan line length of 151 mm ± 1%	Invalid
37	Recording width capability 1728 picture elements along scan line length of 107 mm ± 1%	Invalid
38	Reserved for future recording width capability	
39	Reserved for future recording width capability	
40	Extend field	Extend field

ตารางที่ 4.6(ต่อ) แสดงรูปแบบและข้อกำหนดของสัญญาณ DIS,DTC และ DCS

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

6. ส่วนของการตรวจสอบ (Frame Checking Sequence:FCS)

FCS มักจะถูกเรียกอีกชื่อหนึ่งว่า CRC (Cyclic Redundancy Code) ซึ่งใช้สำหรับการตรวจสอบข้อมูลในเฟรม มีขนาด 16 บิทซึ่งจะคำนวณได้จากจำนวนไบท์(byte)ทั้งหมดในเฟรมนั้นก่อนที่จะถึง FCS โดยที่โมเด็มจะสร้าง FCS เพื่อใช้ในการส่งข้อมูลใน 1 เฟรม และจะตรวจสอบ FCS ในเฟรม ที่รับมาจากเครื่องโทรสารปลายทาง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 5

แฟกซ์โมเด็ม (Fax Modem)

ถ้าจะมองด้านรูปลักษณ์ภายนอกที่มองเห็น เราไม่สามารถแยกความแตกต่างระหว่างแฟกซ์โมเด็มและดาต้าโมเด็มออกจากกันได้ และวิธีการผสมสัญญาณหรือวิธีการตรวจสอบความผิดพลาดที่ใช้ในดาต้าแฟกซ์โมเด็มก็คล้ายๆกับที่ใช้ในแฟกซ์โมเด็มเช่นกัน ข้อแตกต่างกันที่เด่นชัดคือ โปรโตคอล (protocol) ซึ่งควบคุมการติดต่อสื่อสาร แฟกซ์โมเด็มจะต้องสนับสนุนโปรโตคอลที่กำหนดไว้ในมาตรฐาน T.4,T.6 และ T.30 ของ CCITT กล่าวคือแฟกซ์โมเด็มจะต้องจัดหาคำสั่งใหม่ๆและตอบสนองต่อการเริ่มต้นที่จะนำไปสู่การส่งโทรสาร

โมเด็มที่ใช้ในเครื่องโทรสารกลุ่ม 3 ได้กำหนดไว้ตามมาตรฐานของ CCITT ดังตารางต่อไปนี้

Bits per Second	Baud Rate	Bits per Sample	Type	Carrier Frequency	Bandwidth in Hertz
14400	2400	6	V.17	1800	550-3050
12000	2400	5	V.17	1800	550-3050
9600	2400	4	V.29	1700	450-2950
7200	2400	3	V.29	1700	450-2950
4800	1600	3	V.27*ter*	1800	950-2650
2400	1200	2	V.27*ter*	1800	1150-2450

ตารางที่ 5.1 มาตรฐานของโมเด็มที่ใช้ในเครื่องโทรสารกลุ่มที่ 3 ตามข้อกำหนดของ CCITT

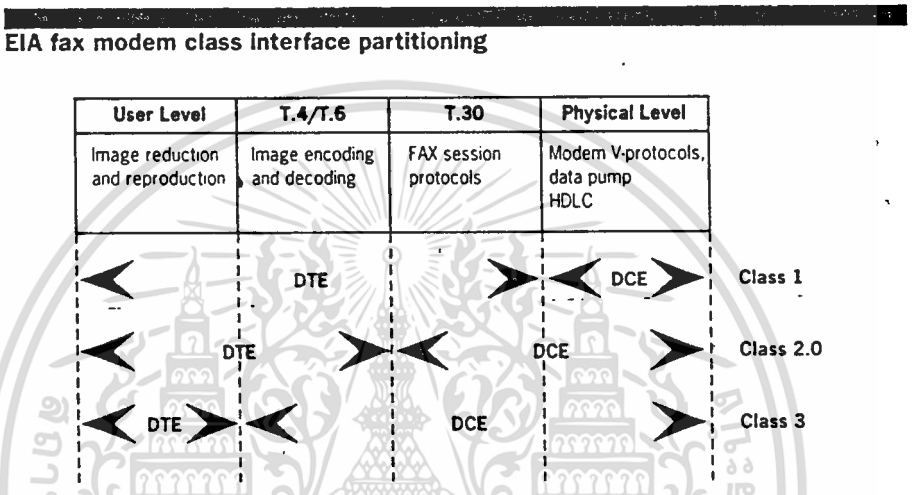
โมเด็มที่ใช้ในเครื่องโทรสารกลุ่ม 3 จะตรวจสอบการติดต่อระหว่างเครื่องโทรสารทั้ง 2 ด้านก่อนการส่งข้อมูล และโมเด็มจะพยายามส่งข้อมูลด้วยความเร็วสูงสุดเท่าที่จะเป็นไปได้ ส่วนใหญ่โมเด็มในเครื่องโทรสารได้ถูกกำหนดไว้ตามมาตรฐาน V.27*ter* และ V.29 แต่จะมีเครื่องโทรสารรุ่นใหม่ๆบางเครื่องที่จะใช้มาตรฐาน V.30 โดย V.27*ter* เป็นมาตรฐานเกี่ยวกับโมเด็มที่ส่งข้อมูลด้วยความเร็ว 4800/2400 บิทต่อวินาที ผ่านช่ายสายโทรศัพท์ โดยการผสมสัญญาณแบบเลื่อนเฟส โดยมีความถี่พาห์ที่ 1800±1 เฮิรตซ์ ส่วน V.29 เป็นข้อแนะนำในการส่งแบบ 9600 บิท/วินาที โดยความถี่พาห์เป็น 1700±1 เฮิรตซ์ และ V.33 จะส่งข้อมูลด้วยความเร็วสูงสุด 14400 บิท/วินาที ผสมสัญญาณแบบ QAM(TCM)

โดยส่วนใหญ่ในการส่งข้อมูลของเครื่องโทรสารกลุ่ม 3 นั้น จะเริ่มต้นที่ความเร็ว 9600 บิทต่อวินาที แต่ถ้าหากไม่สามารถส่งข้อมูลด้วยความเร็วนี้ได้ โมเด็มจะลดความเร็วในการส่งลงเป็น 7200,4800 หรือ 2400 บิทต่อวินาที แทนตามความเหมาะสม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การแบ่งระดับโมเด็มโดย EIA

ความสามารถของเครื่องโทรสารที่เพิ่มขึ้นจากโมเด็มปกติ ทำให้จำเป็นต้องมีคำสั่งใหม่ๆในการติดต่อ แต่ไม่มีบริษัทใดเป็นผู้กำหนดมาตรฐานนี้ ดังนั้นแฟกซ์โมเด็มจึงถูกพัฒนาโดยบริษัทต่างๆแตกต่างกันออกไป ทำให้ไม่สามารถติดต่อเชื่อมโยงกันได้ เพื่อแก้ไขปัญหาดังกล่าวทางสมาคม EIA จึงได้พัฒนามาตรฐานซึ่งกำหนดขั้นตอนและคำสั่งที่ใช้ระหว่าง คอมพิวเตอร์ (DTE) และ โมเด็ม (DCE) ขึ้น โดยแบ่งโมเด็ม ออกเป็น 3 ระดับตามความสามารถของโมเด็มในการนำไปสู่การโทรสารดังนี้



รูปที่ 5.1 แสดงการแบ่งระดับโมเด็มโดย EIA

แฟกซ์โมเด็มระดับ 1

EIA ได้กำหนดแฟกซ์โมเด็มระดับ 1 ไว้ใน EIA/TIA-578 ซึ่งโมเด็มระดับนี้จะหมายถึงแฟกซ์โมเด็มที่มีบริการเท่าที่จำเป็นต่อการรองรับขั้นตอนของโทรสารกลุ่ม 3 เท่านั้น ดังรูปที่ 5.1 คอมพิวเตอร์จะตอบสนองต่อการเข้ารหัสของภาพโดย มาตรฐาน T.4 และการจัดการในการส่งข้อมูลโดย มาตรฐาน T.30 แฟกซ์โมเด็มระดับ 1 จะมีบริการต่างๆต่อไปนี้

-GSTN interface

-หมุนหมายเลขอัตโนมัติ

-V series signal conversion (modulation)

-การรับส่งข้อมูล

-การใช้รูปแบบ HDLC เฟรม,ความชัดเจนของข้อมูล(transparency) และการตรวจสอบความผิดพลาด

-คำสั่งควบคุมและการตอบสนอง

การส่งโทรสารที่แฟกซ์โมเด็มระดับ 1 จะต้องใช้ภายใต้การควบคุมของซอฟต์แวร์ ต่างจากการส่งแบบใช้ดาต้าโมเด็ม ซึ่งการกำหนดเวลา,การเข้ารหัสและการลำดับการส่งต้องอาศัย มาตรฐาน T.30 ทำให้ไม่สามารถควบคุมการส่งโทรสารโดยใช้คำสั่งจากผู้ควบคุมเครื่องได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แฟกซ์โมเด็มระดับ 2

แฟกซ์โมเด็มระดับนี้มีความสามารถสูงแฟกซ์โมเด็มระดับแรก การแบ่งส่วนต่างๆก็จะเปลี่ยนไปดังแสดงในรูปที่ 5.1 ซึ่งขั้นตอนที่ยุ่งยากในการส่งโทรสารโดยใช้ มาตรฐาน T.30 จะถูกตัดออกจากคอมพิวเตอร์ไปเป็นหน้าที่ของโมเด็มแทน ซึ่งโมเด็มระดับ 2 นี้มีบริการต่างๆต่อไปนี้

-GSTN interface

-การหมุนหมายเลขอัตโนมัติ

-V series signal conversion

-มาตรฐาน T.30 protocol implementation

-การรายงานสภาวะการทำงาน

-การส่งข้อมูลเฟส C

-padding สำหรับเวลาในการสแกนที่น้อยที่สุด

-การตรวจสอบข้อมูลที่รับได้

-โปรโตคอลแบบแพคเก็ต (packet) สำหรับ DTE/DCE interface

แฟกซ์โมเด็มระดับ 2 จะเริ่มต้นการเรียก,จัดการขบวนการติดต่อสื่อสาร,ส่งข้อมูลภาพและหน้าที่อื่นๆระหว่างรูปแบบของ มาตรฐาน T.4 (เครื่องโทรสารกลุ่ม 3) และ มาตรฐาน T.6 (เครื่องโทรสารกลุ่ม 4) เครื่องคอมพิวเตอร์จะทำหน้าที่ในการจัดเตรียมและลดขนาดข้อมูลภาพสำหรับการสื่อสารและตีความข้อมูลที่ถูกลดขนาดมาแล้วในทางด้านรับ

แฟกซ์โมเด็มระดับ 3

การกำหนดมาตรฐานของแฟกซ์โมเด็มระดับ 3 ยังอยู่ในระหว่างการศึกษา แต่มีแนวโน้มในการโอนหน้าที่ต่างๆ ในขบวนการส่งโทรสารจากเครื่องคอมพิวเตอร์ไปให้กับโมเด็ม ดังแสดงในรูปที่ 5.1 จากมาตรฐาน T.30 และหน้าที่ทั่วไปของโมเด็มในการส่งข้อมูล แฟกซ์โมเด็มระดับ 3 จะเปลี่ยนข้อมูลภาพให้เป็นภาพที่ถูกลดขนาดตามมาตรฐาน T.4 และ T.6 ซึ่งโมเด็มอาจจะเป็นตัวขยายภาพทางด้านรับด้วยก็ได้

กลุ่มคำสั่งที่ใช้ควบคุมโมเด็ม

คำสั่งสำหรับควบคุมแฟกซ์โมเด็มระดับ 1 และ 2 โดย EIA ถูกออกแบบเพิ่มเติมจากกลุ่มคำสั่ง AT ซึ่งสายของอักขระที่ถูกส่งไปยังโมเด็มจะเรียกว่า แถวคำสั่ง (command line) และจะต้องขึ้นต้นด้วยอักษร AT หรือ at แถวคำสั่งประกอบด้วยสัญลักษณ์ใน รหัสแอสกี้ (ascii) และลงท้ายด้วยเครื่องหมายแสดงการสิ้นสุดคำสั่งที่เรียกว่า รหัสแอสกี้ 13 (carriage return)

มีเพียง บิทต่ำ 7 บิทของแต่ละอักษรเท่านั้นที่จะถูกตรวจสอบในขณะที่โมเด็มแปลความหมายของคำสั่ง คำสั่งที่เขียนด้วยอักษรตัวใหญ่หรือตัวเล็กจะมีความหมายเหมือนกัน ช่องว่างและเครื่องหมายควบคุมอื่นๆที่นอกเหนือไปจาก รหัสแอสกี้ 13 และ รหัสแอสกี้ 8 (ช่องว่าง) ที่ปรากฏในคำสั่งจะถูกละเลยไป และจะถูกแสดงค่าด้วย 0 ด้วยเหตุนี้แฟกซ์โมเด็มทุกเครื่องจึงต้องมี การควบคุมการไหล (flow control) XON/XOFF และอาจมี การควบคุมการไหล (flow control) อื่นๆอีกก็ได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

<u>คำสั่งสำหรับแฟกซ์โมเด็มระดับ 1</u>

คำสั่งแฟกซ์ของ EIA แต่ละคำสั่งจะขึ้นต้นด้วย +F ซึ่งมีรูปแบบทั่วไปอยู่ 3 รูปแบบตามการใช้งาน ได้แก่ ความสามารถ (capabilities), สถานะ (status) และการกำหนด (set)

ถ้าต้องการกำหนดขอบเขตความสามารถของโมเด็ม จะใช้รูปแบบดังนี้

+Fcommand=?

ซึ่ง *command* จะแสดงคำสั่งแฟกซ์ที่ใช้งานได้ โมเด็มจะตอบสนองโดยการเลือกค่าหรือช่วงของค่าที่มันสามารถรับได้ เช่นโมเด็มระดับ 1 อาจจะตอบสนองต่อแถวคำสั่งซึ่งรูปแบบ AT+FCLASS=? (เป็นการบอกว่าการติดต่อสื่อสารในครั้งนี้ใช้ระดับใด) ด้วยค่า 0,1 ผลตอบสนองนี้แสดงให้เห็นว่าโมเด็มสามารถถูกกำหนดให้เป็นระดับ 1 ได้

คำสั่งชุดที่ 2 คือ คำสั่งเกี่ยวกับสถานะ จะเป็นการกำหนดค่าขององค์ประกอบหรือทางเลือกในโครงสร้าง คำสั่งนี้มีรูปแบบดังนี้

+Fcommand?

ตัวอย่างการกำหนดรูปแบบการทำงานเช่น กำหนดสั่งว่า AT+FCLASS? โมเด็มซึ่งทำงานเหมือนเป็นแฟกซ์โมเด็มระดับ 1 จะตอบสนองต่อคำสั่งด้วย 1

คำสั่งชุดสุดท้าย ใช้สำหรับกำหนดค่าขององค์ประกอบหรือผ่านค่าองค์ประกอบที่ใช้ควบคุมการทำงานของโมเด็ม มีรูปแบบดังนี้

+Fcommand=val

ซึ่ง *command* จะแสดงถึงองค์ประกอบที่ต้องการกำหนดและ *val* แสดงถึงค่าขององค์ประกอบที่ต้องการ ทั้งนี้ขึ้นอยู่กับแต่ละคำสั่ง *val* อาจจะเป็นตัวเลขหรือตัวอักษรก็ได้ ซึ่งการกำหนดเป็นตัวเลขทั้งในคำสั่งมาตรฐาน AT ดาต้าโมเด็ม (data modem) และในคำสั่งแฟกซ์โมเด็มระดับ 1 จะแสดงด้วยเลขฐานสิบ

ค่าเริ่มต้นของแถวคำสั่งตัวสุดท้ายคือ รหัสแอสกี้ 13 (carriage return) หรือแถวคำสั่งอาจจะจบด้วยเครื่องหมาย ; ก็ได้ ยกเว้นคำสั่ง +FTS และ +FRS คำสั่งของโมเด็มระดับ 1 จะต้องเป็นคำสั่งสุดท้ายในบรรทัด (แม้ว่าจะได้กำหนดไว้อย่างชัดเจนใน EIA/TIA-578 แต่แฟกซ์โมเด็มระดับ 1 จำนวนมากก็ยอมรับคำสั่งหลายๆคำสั่งในบรรทัดเดียวกันโดยไม่มีการแบ่งแยกเป็นส่วน)

แฟกซ์โมเด็มอาจจะถูกกำหนดให้ตอบสนองต่อตัวอักษรหรือตัวเลขอย่างใดอย่างหนึ่งก็ได้ การตอบสนองต่อตัวอักษรจะตามด้วย รหัสแอสกี้ 13 (carriage return) และเครื่องหมายแสดงการขึ้นบรรทัดใหม่ ส่วนการตอบสนองต่อตัวเลขจะตามด้วย รหัสแอสกี้ 13 (carriage return) เท่านั้น และมีรหัสที่ใช้แสดงผลดังนี้คือ ตกลง(0), ติดต่อ(1), ไม่มีสัญญาณ(3), ผิดพลาด(4)

<u>คำสั่งสำหรับแฟกซ์โมเด็มระดับ 2</u>

คำสั่ง EIA ระดับ 2 นี้มีความจำเป็นต้องใช้รูปแบบตามข้อกำหนดเช่นเดียวกับระดับ 1 และจะขึ้นต้นคำสั่งด้วย +F เสมอ รูปแบบของคำสั่งมี 3 แบบเช่นเดียวกับระดับ 1 ดังกล่าวข้างต้น สำหรับระดับ 2 คำสั่งที่ใช้สำหรับกำหนดค่าขององค์ประกอบหรือผ่านองค์ประกอบที่ใช้ในการควบคุมการทำงานของโมเด็มอาจ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จะใช้ได้กับค่าเดียวหรือหลายค่าก็ได้ ซึ่งต่างจากระดับ 1 อย่างก็ตาม ในคำสั่งระดับ2ค่าคงที่จะต้องเป็นเลขฐาน 16 ค่าคงที่ต่างๆนี้สร้างจากขึ้นมาจาก 0 ถึง 9 (รหัสแอสกี้ 30h ถึง 39h) และ A ถึง F (รหัสแอสกี้ 41h ถึง 46h) เช่นค่าคงที่ 255 ในเลขฐานสิบมีค่าเท่ากับ FFh ในฐาน 16 จะถูกส่งตัวอักษร 2 ตัวคือ FF (หมายเหตุ:อักษร h ซึ่งแสดงให้ทราบว่าเป็นเลขฐาน 16 ในหนังสือจะไม่ถูกส่งออกไปด้วย)

กลุ่มของค่าคงที่ซึ่งประกอบด้วยสัญลักษณ์ใน แอสกี้ จะต้องครอบด้วยเครื่องหมาย (") ส่วนสตริงว่าง (null-string)จะมีเครื่องหมาย ("")

สำหรับการกำหนดค่าที่มีหลายๆค่า แต่ละค่าจะถูกแยกด้วยเครื่องหมาย "," เช่นผลตอบสนองคือ (0,2,4,8) ส่วนการกำหนดค่าเป็นช่วงจะแสดงด้วยเครื่องหมาย "-" เช่น กำหนดค่าอยู่ในช่วง 0-255 (ฐานสิบ) สามารถเขียนได้ว่า 0-FF เป็นต้น และคำสั่งในระดับ 2 ยังยอมรับการกำหนดค่าหลายแบบผสมกัน ประกอบด้วยค่าที่เป็นลำดับซึ่งแต่ละค่าจะอยู่ในวงเล็บและแบ่งด้วย "," โดยจะไม่สนใจช่องว่างระหว่างค่าเช่น(0,1,2), (0),(0-3)

คำสั่งระดับ 2 จะถูกทำจากซ้ายไปขวาและแต่ละคำสั่งจะถูกทำแยกจากกัน โดยไม่สนใจสิ่งที่ตามหลังแถวคำสั่ง ถ้าทุกคำสั่งสามารถทำได้ถูกต้อง จะแสดงด้วยรหัสที่บอกสถานะของคำสั่งสุดท้ายในบรรทัด แต่ถ้าเกิดความผิดพลาดขึ้นหรือพบคำสั่งที่ไม่ถูกต้องตามข้อกำหนด การประมวลผลคำสั่งจะหยุดลงและคำสั่งอื่นๆที่ยังไม่ได้ทำจะยกเลิกไป รหัสแสดงผลของแฟกซ์โมเด็มระดับ 2 มีดังนี้คือ

0 : OK

1 : CONNECT

2 : RING

3 : NO CARRIER

4 : ERROR

5 : NO DIALTONE

6 : BUSY

7 : NO ANSWER

การพัฒนาข้อกำหนดของแฟกซ์โมเด็มระดับ 2 ถูกนำโดยอนุกรรมการ TR-29.2 ในการเชื่อมต่อของเครื่องโทรสารระบบดิจิตอล ในเดือนสิงหาคม 1990 อนุกรรมการดังกล่าวได้ออกมาตรฐาน SP-2388-A โดยร่างชุดแรกเกี่ยวกับการนำไปสู่มาตรฐานแฟกซ์โมเด็มระดับ 2.0 TIA/EIA-592 แต่ต่อมาได้มีการออก SP-2388-B มาแทน ซึ่งมีบางส่วนที่ยังคงยึดถือตามมาตรฐานแฟกซ์โมเด็มระดับ 2 ที่ออกในเดือนสิงหาคม 1990 โดยคำสั่งเกี่ยวกับการจัดระดับจะใช้ดังต่อไปนี้

0 : ดาต้าโมเด็ม

1 : EIA/TIA-578

2 : SP-2388-A

2.0 : TIA/EIA-592

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตัวอย่างคำสั่งควบคุมแฟกซ์โมเด็มระดับ 2

+FAA(Select Fax Auto Answer Mode : โหมดตอบรับอัตโนมัติ)

รูปแบบ : AT+FAA = n

+FFA = 0 ตอบรับเมื่อแฟกซ์โมเด็มเลือกใช้เฉพาะโหมดที่กำหนดโดยคำสั่ง + FCLASS

+FAA = 1 ตรวจจับข้อมูลและการเรียกอัตโนมัติ และตอบรับตามความเหมาะสม

ค่าเริ่มต้น: +FAA = 0

ในโหมดตอบรับอัตโนมัติ โมเด็มจะปรับตัวเองโดอัตโนมัติและตอบรับทันที

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 6

พอร์ทสื่อสารอนุกรมบนคอมพิวเตอร์ (Personal Computer)

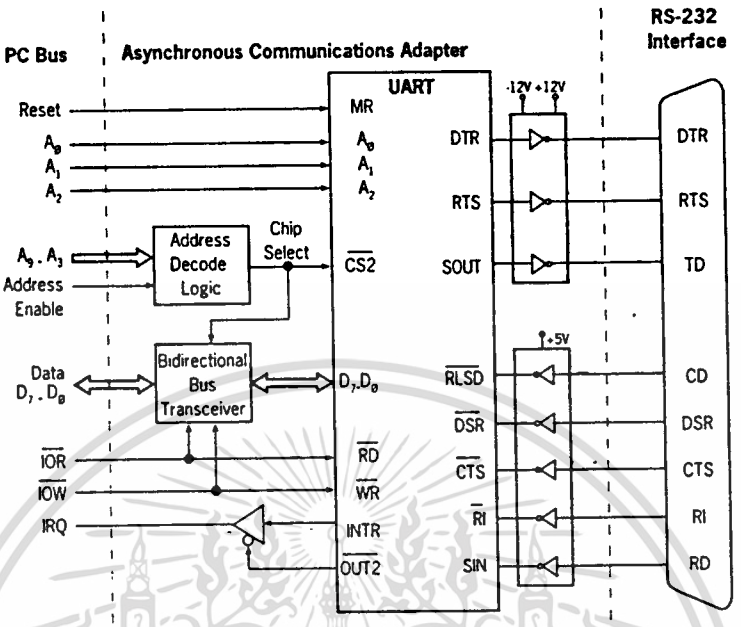
บอร์ดระบบของคอมพิวเตอร์โดยทั่วไปจะมีอะแดปเตอร์สื่อสารแบบอะซิงโครนัส (Asynchronous communication adapter) จะเป็นอะแดปเตอร์แบบพลั๊กอิน (plug in) สำหรับการเชื่อมต่อระหว่างไมโครโพรเซสเซอร์ (microprocessor) ของคอมพิวเตอร์ กับการเชื่อมต่อแบบ RS-232 อะแดปเตอร์นี้สามารถที่จะเซตค่า อินพุท/เอาท์พุท แอดเดรส (I/O address) ได้ 2 ค่าและฮาร์ดแวร์อินเตอร์รัพท์ (hardware interrupt) 1 คู่ อะแดปเตอร์จะสนับสนุน การเชื่อมต่อแบบกระแสลูปเทเลไทพ์ (current loop teletype interface) สำหรับขับกระแสให้กับเครื่องพิมพ์รุ่นเก่า IBM

IBM จะมีอะแดปเตอร์สื่อสารแบบอะซิงโครนัสรุ่นเดิมซึ่งเป็นการ์ดอะแดปเตอร์ในคอมพิวเตอร์รุ่น AT ซึ่งเป็นแบบอนุกรม/ขนาน ในการ์ดอันเดียวจะมีทั้งพอร์ทอนุกรมและพอร์ทขนาน สำหรับอะแดปเตอร์แบบอนุกรมจะแบ่งได้เป็น 3 ประเภท นั้นคือประเภทแรกจะเป็นอะแดปเตอร์ที่ประกอบด้วย 8250 หรือ 16450 UART เป็นอะแดปเตอร์ที่พื้นฐานสุดสำหรับการสื่อสารโดยทั่วไป ส่วนประเภทที่สองจะคล้ายประเภทแรกแต่จะใช้ 16550 และ FIFO (first in first out) บัฟเฟอร์ ใช้สำหรับการสื่อสารโมเด็มความเร็วสูงและสภาวะแวดล้อมการทำงานแบบหลายๆงาน (multitasking environments) แต่จะมีราคาแพงกว่าประเภทแรก และประเภทสุดท้ายประเภทที่สามจะประกอบด้วยอะแดปเตอร์แบบพิเศษคือ ตัวควบคุมพอร์ทอนุกรม (serial port controller) และ Hayes Enhanced Serial Interface (ESI) สำหรับตัวควบคุมประเภทที่ 3 นี้จะสนับสนุน การถ่ายเทข้อมูลแบบ DMA และมีการควบคุมการไหล (flow control) ของตัวเอง Hayes ESI เป็นโคโพรเซสเซอร์ (coprocessor) ทางการสื่อสารที่สมบูรณ์แบบสามารถใช้ไมโครโพเซสเซอร์ของตนเองในการจัดการสื่อสารโดยไม่ขึ้นกับประเภทของคอมพิวเตอร์

โครงสร้างของพอร์ทอนุกรม

บล๊อกไดอะแกรมของพอร์ทอนุกรมประเภทแรกที่เชื่อมต่อระหว่างบัสของคอมพิวเตอร์ (PC bus inteface) กับ การเชื่อมต่อRS-232 (Rs-232 interface) แสดงดังรูป 6.1

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูป 6.1 แสดงบล๊อกไดอะแกรมพอร์ทอนุกรม

<u>อินพุท/เอาท์พุท แอดเดรส</u> ใน IBM อะแดปเตอร์แบบอนุกรมจะมี อินพุท/เอาท์พุทแอดเดรส อยู่ 2 ช่วง สำหรับใช้โดยการ์ด เมื่อเราติดตั้งพอร์ทอนุกรมอันแรก (COM1) อะแดปเตอร์จะจอง 8 อินพุท/เอาท์พุทแอดเดรส โดยมีฐานแอดเดรสอยู่ที่ 3F8h (h แสดงว่าเป็นเลขฐาน 16) และฐานแอดเดรสของพอร์ทอนุกรมอันที่สอง (COM2) จะอยู่ที่ อินพุท/เอาท์พุทแอดเดรส 2F8h พอร์ทอนุกรมทั้งสองจะใช้แอดเดรส 3 บิทต่ำของอินพุท/เอาท์พุท แอดเดรสในคอมพิวเตอร์(PC I/O address) ในการระบุหมายเลขของรีจิสเตอร์ใน UART เช่น รีจิสเตอร์ตัวที่ 4 ของ พอร์ทอนุกรมอันที่สอง จะมีแอดเดรสตรงกับ 2F8h + 4 = 2FCh

หมายเหตุ COM1 และ COM2 เป็นชื่อของอุปกรณ์ที่ตั้งให้โดย MS-DOS สำหรับพอร์ทอนุกรมอันที่ 1 และอันที่ 2 ตามลำดับ ที่ติดตั้งอยู่ในระบบ เพราะโปรแกรมส่วนใหญ่จะติดต่อกับแฮร์ดแวร์โดยตรง ดังนั้นชื่อทั้ง 2 นี้จึงเป็นที่รู้จักโดยทั่วไปเพื่อการอ้างถึงแฮร์ดแวร์อะแดปเตอร์

ต่อมามีการเพิ่มพอร์ทอนุกรมเพิ่มเข้าอีกเพื่อสนับสนุนความต้องการใช้พอร์ทอนุกรมที่มีมากขึ้นโดยจะเพิ่มจากฐานแอดเดรส 3F8h และ 2F8h โดยอยู่ที่แอดเดรส 3E8h สำหรับพอร์ทอนุกรมอันที่ 3 (COM3) และ 2E8h สำหรับพอร์ทอนุกรมอันที่ 4 (COM4) ดังนั้นจึงสรุปความสัมพันธ์ระหว่างเบอร์พอร์ท และ อินพุท/เอาท์พุท แอดเดรส ดังตาราง 6.1

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Serial Port Number	UART I/O Base Address	Default IRQ
1	3F8h-3FFh	4
2	2F8h-2FFh	3
3	3E8h-3EFh	4
4	2E8h-2EFh	3

ตาราง 6.1 แสดงความสัมพันธ์ระหว่างเบอร์พอร์ทอนุกรมและ I/O แอดเดรส

การเชื่อมต่อบัสบนคอมพิวเตอร์ (PC Bus Interface) การอ่านและเขียนข้อมูลบนรีจิสเตอร์ของ UART อินพุท/เอาท์พุทแอดเดรส ของรีจิสเตอร์จะวางอยู่บนแอดเดรสบัสของคอมพิวเตอร์ คอมพิวเตอร์จะสโตรป (strobe) เส้นสัญญาณ IOR (I/O read) หรือ IOW (I/O write) เมื่อลอจิกถอดรหัสแอดเดรส(address decoding logic)ที่อยู่บนการ์ดอนุกรม (เป็น and gate) ตรวจสอบพบว่าเป็นแอดเดรสของตัวมันเองที่ถูกวางอยู่บนบัส มันจะทำการกระตุ้นสัญญาณเลือกชิป (chip select) ของ UART (CS2) เป็นการอ้างถึงรีจิสเตอร์ภายในของ UART ระบุโดยแอดเดรสของคอมพิวเตอร์ เส้นแอดเดรส A_2-A_0ซึ่งเป็น 3 บิทต่ำของ อินพุท/เอาท์พุทแอดเดรส

สัญญาณเลือกชิปจะทำให้สัญญาณข้อมูลของ UART ต่อกับบัสข้อมูลของคอมพิวเตอร์โดยการควบคุม ตัวรับ/ส่งที่มีบัสสองทิศทาง(bidirectional bus transceiver) ทิศทางการไหลของข้อมูลที่ผ่านตัวรับ/ส่งจะได้จากการตรวจสอบ สัญญาณ IOR และ IOW ของคอมพิวเตอร์ ซึ่งสัญญาณเหล่านี้จะต่อกับขา RD และ WR ของ UART

การเชื่อมต่ออินเทอร์รัพ (Interrupt Interface) สำหรับแต่ละพอร์ทอนุกรมจะมีสัญญาณอินเทอร์รัพท์โดยต่อสัญญาณ INTR เข้ากับบัสของคอมพิวเตอร์ IBM ได้กำหนดให้ IRQ 4 ใช้โดย COM1 และ IRQ 3 ใช้โดย COM2 ส่วนของ COM3 และ COM4 จะใช้ซ้ำโดยได้แสดงไว้ในตาราง 6.1 ดังนั้นในการใช้งานแบบอินเทอร์รัพท์จึงไม่สามารถที่เกิดการใช้สัญญาณ IRQ ซ้ำกันได้ในเวลาเดียวกัน ปัญหานี้ในการ์ดโครงข่าย (Network card) บางอย่างได้กำหนดให้พอร์ทอนุกรมสามารถใช้ IRQ 2,3,4,5 หรือ 7 ซึ่งการกำหนดดังกล่าวจะไม่เป็นไปตามมาตรฐานทั่วๆไปที่ใช้กันอยู่

จากบล็อกไดอะแกรมข้างต้นจะเห็นว่าสัญญาณ INTR จะไม่ต่อโดยตรงกับสัญญาณ IRQ ที่อยู่บนบัสของคอมพิวเตอร์ แต่จะมีลอจิกเกต (logic gate) คั่นกลางโดยคุมจากสัญญาณ OUT2 ของ UART สำหรับการควบคุมการเปิดปิดของเกต ดังนั้นในการที่จะทำให้เกิดการอินเทอร์รัพท์ขึ้นจึงต้องมีการโปรแกรมให้เซตบิท OUT2 ที่อยู่ในรีจิสเตอร์ควบคุมโมเด็ม (modem control register) เป็น 1

การเชื่อมต่อ RS-232 (RS-232 Interface) ขา DTR, RTS และ SOUT ของ UART จะเชื่อมต่อกับ RS-232 โดยผ่านตัวขับสัญญาณ (line drivers) สำหรับตัวขับสัญญาณนี้จะแปลงสัญญาณ 0V/+5V ของ UARTเป็น-12V/+12V ตามมาตรฐานของ RS-232 เช่นเดียวกันกับขาสัญญาณอินพุท CD, DSR,

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

CTS, RI และ RD ของ RS-232 ก็จะต่อกับขาของ UART ที่สอดคล้องกัน ในการออกแบบตัวขับสัญญาณของ RS-232 นั้นจะใช้ตัวขับสัญญาณที่มีลอจิกตรงข้ามคือมันจะทำการกลับค่าลอจิกอินพุทที่มีมาจาก UART

<u>รายละเอียดของรีจิสเตอร์ของ UART</u>

ในรายละเอียดของรีจิสเตอร์จะเป็นกล่าวถึงอย่างคร่าวๆถึงหน้าที่ของแต่ละบิทที่อยู่ภายในรีจิสเตอร์ซึ่งจะกล่าวถึงโดยอ้างฐานแอดเดรสของพอร์ท COM1 (3F8h-3FFh) ส่วนของพอร์ทอนุกรมอื่นๆก็มีส่วนประกอบของรีจิสเตอร์เหมือนกันเพียงแต่จะเริ่มจากฐานแอดเดรสต่างกัน

พอร์ท 3F8h-3FFh ของ COM1 (8250) ประกอบส่วนต่างๆดังนี้

พอร์ท 3F8h

ขณะเขียน เป็น รีจิสเตอร์ทรานสมิตเตอร์โฮลดิ้ง(transmitter holding)จะเก็บตัวอักขระที่จะส่งออกไปโดยที่บิท 0 จะส่งออกไปก่อน บิท 7-0 จะเป็นบิทข้อมูลเมื่อ DLAB = 0 (บิท Divisor Latch Access)

ขณะอ่าน เป็น รีจิสเตอร์รีซีฟเวอร์บัฟเฟอร์(receiver buffer) ใช้เก็บตัวอักขระที่รับเข้ามาโดยที่บิท 0 จะถูกรับเข้ามาก่อน บิท 7-0 จะเป็นบิทข้อมูลเมื่อ DLAB=0

ขณะอ่าน/เขียน เป็นไบต์ต่ำของแลทช์ตัวหาร (divisor latch) เมื่อ DLAB=1

พอร์ท 3F9h

ขณะอ่าน/เขียน เป็นไบต์สูงของแลทช์ตัวหารเมื่อ DLAB=1

ขณะอ่าน/เขียน เป็น รีจิสเตอร์อินเทอร์รัพท์อีเนเบิ้ล(interrupt enable) เมื่อ DLAB=0

บิท 7-4 สงวนไว้

บิท 3=1 เปิดการอินเทอร์รัพท์ของสถานะของโมเด็ม(modem status)

บิท 2=1 เปิดการอินเทอร์รัพท์ของสถานะของสายทางด้านรับ (receiver line status)

บิท 1=1 เปิดการอินเทอร์รัพท์การว่างของรีจิสเตอร์ทรานสมิตเตอร์โฮลดิ้ง

บิท 0=1 เปิดการอินเทอร์รัพท์ของการมีข้อมูลเข้ามา

พอร์ท 3FAh

ขณะอ่าน เป็นรีจิสเตอร์ระบุการเกิดอินเทอร์รัพท์(interrupt identification) ข่าวสารของการเกิดอินเทอร์รัพท์ต่างๆที่ค้างอยู่จะถูกเก็บไว้ในรีจิสเตอร์นี้ เมื่อมีการอินเทอร์รัพท์เกิดขึ้นจะมีเฉพาะอินเทอร์รัพท์ที่มีความสำคัญสูงสุดจะได้รับการบริการจากหน่วยประมวลผลกลางจนเสร็จก่อนอินเทอร์รัพท์ความสำคัญรองลงมาจึงได้รับการตอบสนอง

บิท 7-3 สงวนไว้

บิท 2-1 ระบุการเกิดอินเทอร์รัพท์ที่มีความสำคัญสูงสุดที่ค้างอยู่

=11 อินเทอร์รัพท์ของสถานะสายทางด้านรับ ความสำคัญสูงสุด

=10 อินเทอร์รัพท์ของการมีข้อมูลมา ความสำคัญอันดับสอง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

=01 อินเทอร์รัพท์การว่างของรีจิสเตอร์ทรานสมิตเตอร์โฮลดิ้ง ความสำคัญอันดับสาม

=00 อินเทอร์รัพท์ของสถานะของโมเด็ม ความสำคัญอันดับสี่

บิท 0 =0 มีอินเทอร์รัพท์ค้างอยู่ ค่าที่อยู่ในรีจิสเตอร์สามารถใช้ในการชี้ประเภทของอินเทอร์รัพท์ที่เกิดขึ้น

1 ไม่มีอินเทอร์รัพท์ค้างอยู่

พอร์ท 3FBh

ขณะอ่าน/เขียน รีจิสเตอร์ควบคุมสาย (line control register)

บิท 7= 1 เป็นบิทอ้างถึงแลทช์ตัวหาร (divisor latch access :DLAB)

0 เป็นการอ้างถึงรีซีฟเวอร์บัฟเฟอร์ , ทรานสมิตเตอร์โฮลดิ้ง หรือ รีจิสเตอร์อินเทอร์รัพท์อีเนเบิ้ล

บิท 6= 1 เซตการอิเนเบิ้ลเบรค(break enable)

บิท 5-4 ประเภทของพาริตี้

=00 พาริตี้คี่ (odd parity)

=01 พาริตี้คู่ (even parity)

=10 พาริตี้มาร์ค (mark parity)

=11 พาริตี้ว่าง (space parity)

บิท 3= 1 ให้มีบิทพาริตี้

บิท 2= 0 มี 1 บิทหยุด (stop bit)

1 ไม่มีบิทหยุด

บิท 1-0 ความยาวของเวิร์ด (word length)

00 มีความยาว 5 บิท

01 มีความยาว 6 บิท

10 มีความยาว 7 บิท

11 มีความยาว 8 บิท

พอร์ท 3FCh

ขณะอ่าน/เขียน เป็นรีจิสเตอร์ควบคุมโมเด็ม

บิท 7-5=000 สงวนไว้

บิท 4 =1 โหมดลูปแบ็ค(loopback mode) สำหรับการตรวจสอบพอร์ทอนุกรม โดยใช้เอาท์พุทของรีจิสเตอร์ทรานสมิตเตอร์ชิฟ(transmitter shift) ป้อนลูปแบ็คกลับไปยังอินพุทของรีจิสเตอร์รีซีฟเวอร์ชิฟ(receiver shift) ในโหมดนี้ข้อมูลที่ส่งออกไปจะได้รับในทันทีดังนั้นหน่วยประมวลผลกลางสามารถที่จะยืนยันถึงข้อมูลที่ส่งและรับจากพอร์ทอนุกรมว่าถูกต้องหรือไม่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บิท 3=1 ใช้กำหนดสัญญาณ $\overline{OUT1}$ ในการใช้งานตามวัตถุประสงค์ทั่วไป โดยขึ้นอยู่กับอุปกรณ์ที่จะนำมาใช้ จะทำให้เอาท์พุท $\overline{OUT1}$ = 0

บิท 2=1 ใช้กำหนดสัญญาณ $\overline{OUT2}$ ในการใช้งานตามวัตถุประสงค์ทั่วไป โดยขึ้นอยู่กับอุปกรณ์ที่จะนำมาใช้ จะทำให้เอาท์พุท $\overline{OUT2}$ = 0 นอกจากนี้ใน การสื่อสารแบบ อะซิงโครนัสของคอมพิวเตอร์ $\overline{OUT2}$ จะถูกกำหนดให้ใช้ในการขับ สัญญาณอินเทอร์รัพท์ของอุปกรณ์อินพุท/เอาท์พุท ด้วย

บิท 1=1 สถานะในบิทนี้ใช้ในการควบคุม สัญญาณ เอาท์พุท $\overline{RTS}$ ของ 8250 โดยทำให้ เอาท์พุท $\overline{RTS}$ = 0

บิท 0=1 สถานะในบิทนี้ใช้ในการควบคุม สัญญาณ เอาท์พุท $\overline{DTR}$ ของ 8250 โดยทำให้ เอาท์พุท $\overline{DTR}$ = 0

พอร์ท 3FDh

ขณะอ่าน เป็นรีจิสเตอร์สถานะของสาย

บิท 7=0 สงวนไว้

บิท 6=1 รีจิสเตอร์ทรานสมิตเตอร์ชิฟว่าง

บิท 5=1 รีจิสเตอร์ทรานสมิตเตอร์โฮลดิ้งว่างพร้อมที่จะรับตัวอักขระใหม่ที่จะส่งต่อไป

บิท 4=1 แสดงสถานะการเบรคอินเทอร์รัพท์(break interrupt)โดยมีสถานะเบรคเกิดขึ้นที่ขาอินพุท SIN

บิท 3=1 เกิดความผิดพลาดทางเฟรม (framing error)

บิท 2=1 เกิดความผิดพลาดทางพาริตี้ (parity error)

บิท 1=1 เกิดความผิดพลาดทางโอเวอร์รัน(overrun)

บิท 0=1 มีข้อมูลมาพร้อมแล้วที่รีจิสเตอร์รีซีฟเวอร์บัฟเฟอร์

พอร์ท 3FEh

ขณะอ่าน เป็นรีจิสเตอร์สถานะของโมเด็ม

บิท 7=1 DCD (data carrier detect) แสดงว่ามีสัญญาณอินพุท $\overline{DCD}$ เข้ามา

บิท 6=1 RI (ring indicator) แสดงว่ามีสัญญาณอินพุท $\overline{RI}$ เข้ามา

บิท 5=1 DSR (data set ready) แสดงว่ามีสัญญาณอินพุท $\overline{DSR}$ เข้ามา

บิท 4=1 CTS (clear to send) แสดงว่ามีสัญญาณอินพุท $\overline{CTS}$ เข้ามา

บิท 3=1 DDCD (delta data carrier detect) แสดงว่ามีการเปลี่ยนสถานะอินพุทของ $\overline{DCD}$ ไม่ว่าจะเปลี่ยนจากสถานะ 1 -> 0 หรือ 0 -> 1

บิท 2=1 TERI (trailing edge ring indicator) แสดงว่ามีการเปลี่ยนสถานะอินพุทของ $\overline{RI}$ จากสถานะสูง ไปสถานะต่ำ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บิท 1=1 DDSR (delta data set ready) แสดงว่ามีการเปลี่ยนสถานะอินพุท ของ $\overline{DTR}$ ไม่ว่าจะเปลี่ยนจาก 1 -> 0 หรือ 0 -> 1

บิท 0=1 DCTS (delta clear to send) แสดงว่ามีการเปลี่ยนสถานะอินพุท ของ $\overline{CTS}$ ไม่ว่าจะเปลี่ยนจาก 1 -> 0 หรือ 0 -> 1

- บิท 0-3 จะถูกรีเซตเมื่อหน่วยประมวลผลอ่านค่าในรีจิสเตอร์สถานะของโมเด็ม
- บิท 4 เป็น RTS ของรีจิสเตอร์สถานะของโมเด็มขณะที่มีการทดสอบลูปแบ็ค
- บิท 5 เป็น DTR ของรีจิสเตอร์สถานะของโมเด็มขณะที่มีการทดสอบลูปแบ็ค
- บิท 6 เป็น OUT1 ของรีจิสเตอร์สถานะของโมเด็มขณะที่มีการทดสอบลูปแบ็ค
- บิท 7 เป็น OUT2 ของรีจิสเตอร์สถานะของโมเด็มขณะที่มีการทดสอบลูปแบ็ค

พอร์ท 3FFh

ขณะอ่าน/เขียน เป็นรีจิสเตอร์แสครทช์ (scratch)ไม่มีหน้าที่อะไรพิเศษที่เกี่ยวกับ UART และสามารถใช้เป็นที่เก็บข้อมูลชั่วคราวขนาด 1 ไบท์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 7

หลักการทำงานของโปรแกรมรับ/ส่งโทรสาร

โปรแกรมทั้งหมดจะมีอยู่ทั้งหมด 5 ส่วนหลักดังนี้

1. โปรแกรมชื่อ SENDFAX.C

ทำหน้าที่หลักในการส่งโทรสารโดยจะให้ผู้ใช้ป้อนพารามิเตอร์ต่างๆเข้าไปในโปรแกรมได้แก่หมายเลขเครื่องโทรสารปลายทาง, ชื่อของไฟล์ที่จะส่ง, ID String และพารามิเตอร์ต่างๆเพื่อการเซ็ตค่าเริ่มต้นให้แก่พอร์ทอนุกรมรวมทั้งพารามิเตอร์ในการกำหนดคุณสมบัติในการส่งโทรสารว่าจะใช้ความเร็วเท่าไหร่, ความละเอียดเท่าไหร่ หรือใช้รหัสอะไรส่ง ในระหว่างการส่งข้อมูลจากคอมพิวเตอร์ไปยังแฟกซ์โมเด็มนั้นจะมีการควบคุมการไหลแบบ CTS/RTS โดยก่อนจะส่งข้อมูลนั้นจะมีการเช็คก่อนว่า CTS แอคทีฟหรือไม่ถ้าใช่ถึงจะส่งข้อมูลไปได้เพื่อให้ข้อมูลที่ส่งไปสัมพันธ์กับจำนวนของไบท์ในบัฟเฟอร์ของแฟกซ์โมเด็มว่ามีว่างพอที่จะรับข้อมูลเข้ามาได้เนื่องด้วยความเร็วของข้อมูลที่แฟกซ์โมเด็มรับเข้ามากับส่งออกไปนั้นมีความเร็วแตกต่างกันโดยบิทเรท (bit rate) จากคอมพิวเตอร์เป็น 19200 บิทต่อวินาทีส่วนบิทเรทที่ของแฟกซ์โมเด็มที่ส่งออกไปยังเครื่องโทรสารปลายทางเป็น 9600 บิทต่อวินาที

การเซ็ตค่าเริ่มต้นให้กับพอร์ทอนุกรมของคอมพิวเตอร์

เมื่อผู้ใช้ทำการเลือกพอร์ทอนุกรมเบอร์ใดแล้วโปรแกรมจะเลือกฐานแอดเดรสที่สอดคล้องกับเบอร์พอร์ทอนุกรมเช่นถ้าเลือก COM1 ก็จะได้ฐานแอดเดรส 3F8h หลังจากนั้นก็จะเซ็ตบิท DLAB เพื่อใส่ค่าตัวหารลงไปในแลทช์ตัวหารซึ่งขนาด 16 บิทดังนั้นจึงได้แบ่งออกเป็นไบท์ต่ำกับไบท์สูงสำหรับค่าตัวหารนั้นหาได้จากสูตร

$$\text{ตัวหาร} = \frac{1.8432 \times 10^{6}}{16 \times \text{บิทเรท}}$$

เมื่อเซ็ตบิทเรทเรียบร้อยแล้วก็จะรีเซ็ทบิท DLAB เพื่อจะกำหนดค่าพาริตี้บิท,บิทหยุด,ความยาวบิทของเวิร์ดรีจิสเตอร์ควบคุมสายสำหรับรายละเอียดของแต่ละบิทในรีจิสเตอร์นั้นได้กล่าวไว้ในบทที่ 6 มาแล้วจึงไม่ได้แสดงถึงการเซ็ตค่าในรีจิสเตอร์อย่างละเอียด และเซ็ตบิท 0 ของรีจิสเตอร์เปิดการอินเทอร์รัพท์เพื่อให้เกิดการอินเทอร์รัพท์เมื่อมีข้อมูลเข้ามาซึ่งในการรับข้อมูลจากพอร์อนุกรมจะใช้การอินเทอร์รัพท์รับข้อมูลเข้ามาแล้วมาเก็บในบัฟเฟอร์เมื่อโปรแกรมต้องการจะอ่านข้อมูลก็จะไปเช็คในบัฟเฟอร์แทนเนื่องจากถ้าใช้วิธีการไปเช็คที่พอร์ทโดยตรงอาจจะทำให้ข้อมูลบางไบท์ถูกทับได้เพราะไปอ่านไม่ทัน สำหรับโปรแกรมส่งจะเซ็ตให้พอร์ทอนุกรมมีบิทเรท 19200 บิทต่อวินาที, ไม่มีพาริตี้บิท, 1 บิทหยุดและมีความยาวเวิร์ด 8 บิท

การเซ็ตค่าเริ่มต้นให้กับแฟกซ์โมเด็ม

ในการเซ็ตค่าเริ่มต้นให้กับแฟกซ์โมเด็มก็จะทำการกำหนดพารามิเตอร์ต่างๆที่จะเป็นต้องใช้ในการส่งโทรสารให้กับแฟกซ์โมเด็มรู้ โดยใช้คำสั่งต่างๆเรียงลำดับดังต่อไปนี้

ATZ	เพื่อรีเซ็ตแฟกซ์โมเด็ม
ATE1V1	เพื่อเซ็ตให้มีการเอ็คโค (Echo) และมีการตอบสนองกลับเป็นตัว

หนังสือถ้าจะให้ตอบสนองกลับเป็นตัวเลขจะเซ็ตเป็น ATV0

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ATM1 X4 &D2 S7=120 S8=2 เพื่อเซ็ตให้ลำโพงของโมเด็มมีเสียงดังจนกว่าจะได้รับคลื่นพาหะ, ให้มีการอีเนเบิ้ล(enable)รหัสตอบสนองทุกอย่าง, ให้มีการวางหูลงเมื่อ DTR ไม่แอคทีฟ รวมทั้งเซ็ตเวลาในการรอคลื่นพาหะ(หน่วยเป็นวินาที)และการเซ็ตเวลาที่โมเด็มจะหยุดรอเมื่อเจอรหัส "," ในขณะที่หมุนหมายเลขโทรศัพท์อยู่

AT+FCLASS = 2 เพื่อเซ็ตแฟกซ์โมเด็มให้เป็นแฟกซ์โมเด็มระดับ 2(SP-2388A)

AT+FCR = 1 เพื่อแฟกซ์โมเด็มพร้อมจะรับข่าวสารในขณะที่มีการส่งแฟกซ์

AT+FDIS = ?,?,?,?,?,?,?,? เพื่อกำหนดพารามิเตอร์ในการตกลงกันระหว่างที่มีการส่งแฟกซ์ สำหรับของพารามิเตอร์ได้กล่าวไว้ในภาคผนวกคำสั่งแฟกซ์โมเด็มระดับ 2

<u>การส่งข้อมูลโทรสาร</u>

ในการส่งข้อมูลจะทำการอ่านข้อมูลจากไฟล์ที่จะส่งโดยที่ไฟล์ดังกล่าวต้องมีรูปแบบเป็น ฮัฟฟ์แมน 1 มิติ ที่ทำการเรียงข้อมูลใหม่จากบิท 0-7 เป็นบิท 7-0 ของไบท์ข้อมูลทุกๆไบท์เนื่องจากในขณะที่ส่งข้อมูลเมื่อแฟกซ์โมเด็มจะทำการส่งข้อมูลสลับจากบิทสูงไปยังบิทต่ำก่อนทำให้เมื่อถึงทางด้านรับ บิทที่รับเข้ามาจะกลับกันโดยบิทสูงที่ส่งเข้ามาก่อนทางด้านรับจะรับเข้ามาเป็นบิทต่ำดังนั้นเพื่อให้ทางด้านรับสามารถรับข้อมูลได้อย่างถูกต้องจึงต้องทำการสลับตำแหน่งของบิทดังกล่าวมาข้างต้น เมื่อหมุนหมายเลขเครื่องโทรสารปลายทางแล้วได้รับสัญญาณ "CED" และ "OK" ก็จะส่งคำสั่ง "AT+FDT" เพื่อเข้าสู่เฟส C ของการส่งโทรสารหลังจากนั้นจะรอจนแฟกซ์โมเด็มส่ง "CONNECT" มาให้ก็จะเป็นการเริ่มต้นส่งข้อมูลแต่ก่อนจะส่งไบท์ข้อมูลทุกครั้งจะต้องทำการตรวจสอบ CTS ทุกครั้งว่าเป็น 1 หรือไม่ถ้าใช่ก็แสดงว่าบัฟเฟอร์ของแฟกซ์โมเด็มว่างพร้อมที่จะรับข้อมูลเข้าไปแต่ถ้าเกิด CTS เป็น 0 อยู่ก็แสดงบัฟเฟอร์เต็มอยู่ให้หยุดส่งข้อมูลจนกว่า CTS จะเซ็ตเป็น 1 และข้อสำคัญอีกอย่างก็คือทุกครั้งที่เป็นไบท์ข้อมูลตรงกับรหัส DLE (Data link escape) มีค่าเท่ากับ 10h จะต้องทำการส่งไบท์นั่นซ้ำอีกครั้งเพื่อแฟกซ์โมเด็มตีความว่าเป็นไบท์ข้อมูลที่เป็นรหัส 10h ไม่ใช่รหัสคำสั่ง เมื่อส่งข้อมูลจนครบไฟล์ก็จะส่งรหัส DLE กับ รหัส 2Eh ไปยังแฟกซ์โมเด็มเพื่อให้รู้ว่าจบแผ่น(End of Page) แล้ว หลังจากนั้นก็จะวางสายลงพร้อมกับรีเซ็ตพอร์ทอนุกรมกับแฟกซ์โมเด็ม สำหรับรูป 7.1 จะเป็นการแสดงถึงการใช้คำสั่งควบคุมแฟกซ์โมเด็มในแต่ละขั้นตอนของการส่งโทรสาร

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

PC(DTE)	Fax Modem
ATDTs ->	
	<- "CED"
	<- "+FCON"
	<- "+FCSI:csi"
	<- "+FDIS:dis"
	<- "OK"
AT+FDT ->	
	<- "+DCS:dcs"
	<- "CONNECT"
<Image Data> -> <DLE> <2Eh> ->	
	<- "OK"
AT+FET=2 ->	
	<- "OK"
	<- "+FHNG:hng"
	<- "+FPTS:pts"
ATZ ->	Hang Up

รูป 7.1 แสดงการใช้คำสั่งควบคุมแฟกซ์โมเด็มในการส่งโทรสาร

2. โปรแกรมชื่อ RECVFAX.C

โปรแกรมจะทำการเซ็ตค่าเริ่มต้นให้กับพอร์ทอนุกรมกับแฟกซ์โมเด็มหลังจากนั้นเมื่อผู้ใช้เลือกให้โปรแกรมรอรับโทรสาร ก็จะมีการรอสัญญาณ "RING" ตอบกลับจากแฟกซ์โมเด็มเมื่อได้รับแล้วก็จะเริ่มส่งคำสั่งไปเซ็ตให้แฟกซ์โมเด็มเข้าสู่เฟส C หลังจากนั้นก็จะเป็นการรับข้อมูลโทรสารที่ส่งมาจากทางเครื่องโทรสารต้นทางเมื่อรับข้อมูลมา ก็จะเก็บลงไฟล์ที่มีรูปแบบของรหัสเป็นฮัฟฟ์แมน

การเซ็ตค่าเริ่มต้นให้กับพอร์ทอนุกรมของคอมพิวเตอร์

การเซ็ตค่าเริ่มต้นให้กับพอร์ทอนุกรมของคอมพิวเตอร์เหมือนกับในโปรแกรมส่งโทรสาร

การเซ็ตค่าเริ่มต้นให้กับแฟกซ์โมเด็ม

ในการเซ็ตค่าเริ่มต้นให้กับแฟกซ์โมเด็มก็จะทำการกำหนดพารามิเตอร์ต่างๆที่จะเป็นต้องใช้ในการรับโทรสารให้กับแฟกซ์โมเด็มรู้ โดยใช้คำสั่งต่างๆเรียงลำดับดังต่อไปนี้

ATZ เพื่อรีเซ็ตแฟกซ์โมเด็ม

AT&C1 เพื่อเซ็ตให้สัญญาณ CD เป็น 1 ก็ต่อเมื่อได้รับคลื่นพาหะเข้ามา

ATV1 M1 X4 &D2 S0=1 เพื่อให้โมเด็มมีการตอบสนองกลับเป็นตัวหนังสือตอบสนองเพื่อเซ็ตให้ลำโพงของโมเด็มมีเสียงดังจนกว่าจะได้รับคลื่นพาหะ, ให้มีการอีเนเบิ้ลรหัสตอบสนองทุกอย่าง, ให้มีการวางหูลงเมื่อ DTR ไม่แอคทีฟ และให้มีการตอบรับโทรศัพท์อัตโนมัติเมื่อได้รับสัญญาณกระดิ่ง 1 ครั้ง

AT+FDIS = ?,?,?,?,?,?,?,? เพื่อกำหนดพารามิเตอร์ในการตกลงกันระหว่างที่มีการส่งโทรสาร สำหรับชุดของพารามิเตอร์ได้กล่าวไว้ในภาคผนวกคำสั่งแฟกซ์โมเด็มระดับ 2

AT+FCLASS=2 เพื่อเซ็ตแฟกซ์โมเด็มให้เป็นแฟกซ์โมเด็มระดับ 2(SP-2388A)

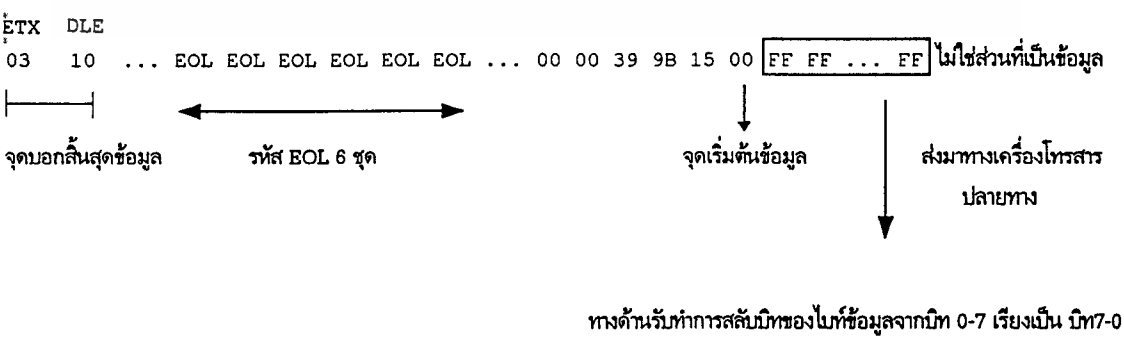
AT+FAA=0;+FCR=1 เพื่อแฟกซ์โมเด็มตอบรับโดยอัตโนมัติทันทีเมื่อมีการเรียกเข้าและกำหนดให้แฟกซ์โมเด็มสามารถที่จะรับข้อมูลข่าวสารได้ในระหว่างการรับโทรสาร

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การรับข้อมูลโทรสาร

หลังจากที่เซ็ตค่าเริ่มต้นให้กับพอร์ทอนุกรมกับแฟกซ์โมเด็มแล้ว โปรแกรมก็จะรอให้มีการโทรเข้ามา เมื่อใดที่เครื่องโทรสารต้นทางโทรเข้ามาแฟกซ์โมเด็มก็จะส่งสัญญาณ "RING" มาให้คอมพิวเตอร์ซึ่งเมื่อโปรแกรมตรวจสอบเจอสัญญาณดังกล่าวก็จะรอสัญญาณ "+FCON" ซึ่งแสดงว่าเริ่มมีติดต่อแลกเปลี่ยนข่าวสารระหว่างทางเครื่องโทรสารต้นทางและเครื่องโทรสารปลายทางต่อมาเมื่อได้รับ "OK" ทางโปรแกรมก็จะส่งคำสั่ง "AT+FDR" เพื่อให้แฟกซ์โมเด็มเข้าเฟส C ในการรับข้อมูล จนเมื่อได้รับสัญญาณ "CONNECT" แล้วแสดงว่าพร้อมจะรับโทรสาร โปรแกรมจะส่งรหัสค่า 12h เพื่อให้แฟกซ์โมเด็มเริ่มส่งข้อมูลโทรสารมาให้กับคอมพิวเตอร์โดยไบท์ข้อมูลช่วงแรกที่รับเข้ามานั้นจะไม่ใช่ไบท์ข้อมูลโดยจะมีค่าเป็น FFh โปรแกรมจะรอจนได้รับไบท์ข้อมูลที่มีค่า 0 ซึ่งแสดงว่าเป็นจุดเริ่มต้นของไบท์ข้อมูลที่แท้จริงหลังจากนั้นก็รับไบท์ข้อมูลมาเก็บไว้ในไฟล์ชั่วคราวที่ได้เปิดไว้จนเจอรหัส DLE(รหัสแอสกี้ 16) กับ ETX(รหัสแอสกี้ 3) ก็อันเป็นสิ้นสุดการรับข้อมูลจากทางเครื่องโทรสารต้นทางดังแสดงไว้ในรูป 7.2 โปรแกรมก็จะรอจนได้รับ "OK" ก็จะส่งคำสั่ง "AT+FDR" เพื่อรอ การเข้าสู่เฟส C เพื่อรับข้อความโทรสารในหน้าต่อมาแต่เนื่องจากโปรแกรมส่งได้ส่งโทรสารเพียงครั้งเดียวต่อเนื่องเพียงหน้าเดียว ไม่มีการส่งข้อความโทรสารครั้งละหลายๆหน้า เมื่อทางเครื่องโทรสารปลายทางได้รับคำสั่งนี้ก็จะตอบ "OK" และเมื่อถึงจุดนี้ก็เป็นอันสิ้นสุดการติดต่อกันระหว่างทางเครื่องโทรสารต้นทางกับเครื่องโทรสารปลายทางโปรแกรมก็จะสั่งให้แฟกซ์โมเด็มวางสายพร้อมกับรีเซ็ตแฟกซ์โมเด็มกับพอร์ทอนุกรม สำหรับรูปที่ 7.3 จะเป็นขั้นตอนในการส่งคำสั่งไปควบคุมแฟกซ์โมเด็มในระหว่างที่มีการรับโทรสารเข้ามา

หลังจากนั้นก็จะทำการอ่านข้อมูลที่ทำการเก็บไว้ในไฟล์ชั่วคราวมาทำการเรียงบิทใหม่จาก บิท 0-7 เป็นบิท 7-0 เพื่อให้สามารถใช้กับโปรแกรม VIEW.C ที่เขียนขึ้นมาเพื่อการแสดงรายละเอียดของโทรสารที่ได้รับมาแทนการพิมพ์ลงบนกระดาษโทรสาร ในขณะที่อ่านข้อมูลจากไฟล์ชั่วคราวก็จะทำการตรวจสอบรหัส EOL ไปด้วยถ้าพบรหัสนี้มากกว่า 2 ครั้งติดต่อกันก็แสดงว่าเป็นจุดสิ้นสุดของข้อมูล ต่อมาก็จะใส่รหัส EOL เพิ่มลงไปจนครบ 6 ชุด สำหรับข้อมูลที่ทำการสลับบิทแล้วจะเก็บลงไปอีกไฟล์หนึ่ง



รูป 7.2 แสดงการรับไบท์ข้อมูลมาแล้วเก็บลงไฟล์เป็นรหัสฮัฟฟ์แมนที่ทำการสลับบิทแล้ว

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

PC(DTE)	Fax Modem
	<- "RING"
	<- "+FCON"
	<- "+FDCS:dcs"
	<- "OK"
AT+FDR ->	
	<- "+FCFR"
	<- "+FDCS:dcs"
	<- "CONNECT"
<12h> ->	
	<- <Image Data>
	<- <DLE> <ETX>
	<- "+FPTS:pts"
	<- "+FET:n"
	<- "OK"
AT+FDR ->	
	<- "+FHNG:hng"
	<- "OK"
ATZ ->	Hang Up

รูป 7.3 แสดงขั้นตอนในการส่งคำสั่งไปควบคุมแฟกซ์โมเด็มในขณะรับโทรสาร

3. โปรแกรมชื่อ CODER.C

โปรแกรมนี้ใช้เพื่อการแปลงจากไฟล์ที่เป็นรหัสแอสกี้จากไฟล์จุฬาเวิร์ด, ราชวิถีเวิร์ด หรือไฟล์แอสกี้ของระบบปฏิบัติการดอส (DOS operating system) ทั่วๆไปให้เป็นรหัสฮัฟฟ์แมน 1 มิติเพื่อสำหรับใช้ในการส่งโทรสาร

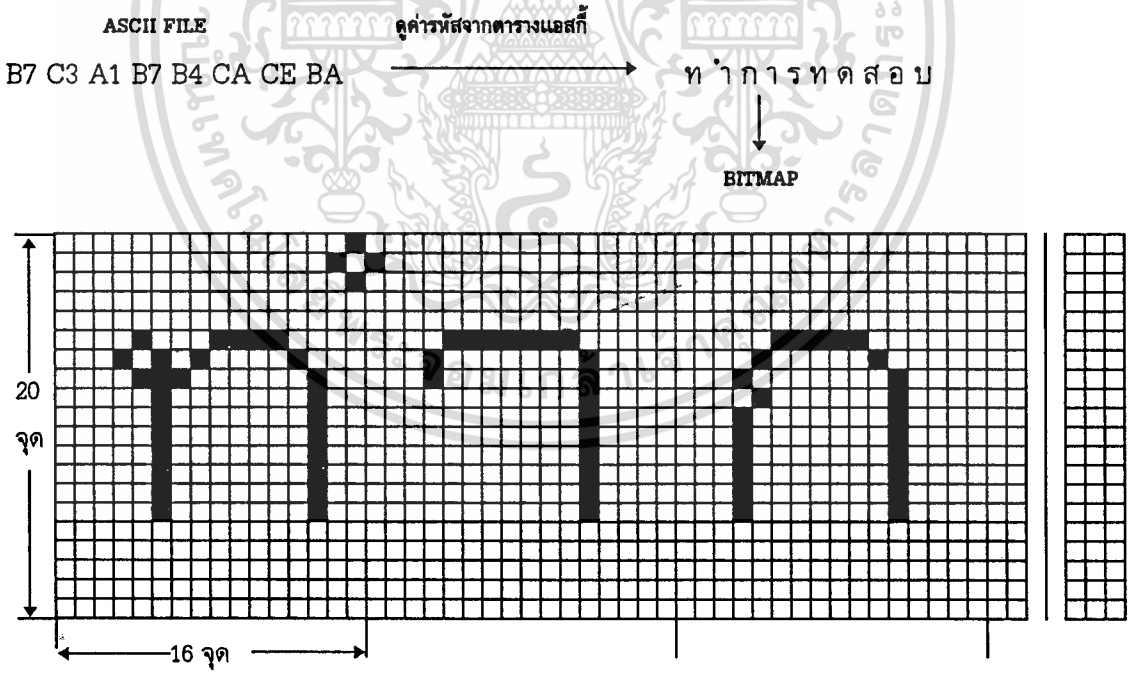
ในการแปลงจากรหัสแอสกี้เป็นรหัสฮัฟฟ์แมน 1 มิติจะทำการอ่านไฟล์แอสกี้ขึ้นมาโดยอ่านจนเจอรหัสขึ้นบรรทัดใหม่ (รหัสแอสกี้ 13) ก็นำมาแปลงเป็นรหัสฮัฟฟ์แมนครั้งหนึ่งโดยการเอารหัสแอสกี้มาหาในตารางแอสกี้ที่เป็นบิทแม็พของฟอนต์ไฟล์ชื่อ "normal2.fon" ซึ่งมีขนาด 16×20 จุด จะนำมาแปลงเป็นบิทแม็พต่อๆกันจนได้ขนาด 1728 จุด เป็นจำนวน 20 แถวแล้วนำเอาแต่ละแถวที่มีขนาด 1728 จุด มาเข้ารหัสฮัฟฟ์แมน 1 มิติซึ่งได้แสดงไว้ดังรูป 7.4 โดยที่รายละเอียดของการเข้ารหัสได้กล่าวไว้แล้วในบทที่ 4 จนครบ 20 แถวก็จะเสร็จการเข้ารหัสของแอสกี้ 1 แถวโดยได้กำหนดไว้ที่ 88 ตัวต่อบรรทัดถ้าในแถวดังกล่าวมีรหัสแอสกี้เกิน 88 ตัวก็จะปัดไปเป็นบรรทัดใหม่ซึ่งจะมีการเข้ารหัสอย่างนี้เรื่อยๆไปจนอ่านเจอรหัสแอสกี้ 1Ah ซึ่งเป็นรหัสจบไฟล์ของจุฬาเวิร์ดกับราชวิถีเวิร์ดส่วนในกรณีที่เป็นไฟล์แอสกี้ธรรมดาก็จะอ่านไปจนหมดไฟล์เพราะไม่มีรหัส 1Ah ปิดท้ายไฟล์

สำหรับตาราง 7.1 จะแสดงความสัมพันธ์ระหว่างรหัสแอสกี้และรหัสภาษาไทยโดยจะแทนที่รหัสภาษาไทยลงไปยังรหัสแอสกี้เดิมในส่วนที่เป็นรหัสที่ไม่ใช่อักษรภาษาอังกฤษ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

		ก	ข	ฃ	ค	ฅ	ฆ	ง	จ	ฉ	ช	ซ	ฌ	ญ	ฎ	ฏ
รหัสแอสกี้(ฐาน16)	A0	A1	A2	A3	A4	A5	A6	A7	A8	A9	AA	AB	AC	AD	AE	AF
	ฐ	ฑ	ฒ	ณ	ด	ต	ถ	ท	ธ	น	บ	ป	ผ	ฝ	พ	ฟ
รหัสแอสกี้(ฐาน16)	B0	B1	B2	B3	B4	B5	B6	B7	B8	B9	BA	BB	BC	BD	BE	BF
	ภ	ม	ย	ร	ฤ	ล	ฦ	ว	ศ	ษ	ส	ห	ฬ	อ	ฮ	ฯ
รหัสแอสกี้(ฐาน16)	C0	C1	C2	C3	C4	C5	C6	C7	C8	C9	CA	CB	CD	CE	CE	CF
	ะ	ั	า	ำ	ิ	ี	ึ	ื				้				
รหัสแอสกี้(ฐาน16)	D0	D1	D2	D3	D4	D5	D6	D7	D8	D9	DA	DB	DC	DD	DE	DF
	เ	แ	โ	ใ	ไ	ๅ	ๆ	็	่	้	๊	๋	์	๎	๏	
รหัสแอสกี้(ฐาน16)	E0	E1	E2	E3	E4	E5	E6	E7	E8	E9	EA	EB	EC	ED	EE	EF
	๐	๑	๒	๓	๔	๕	๖	๗	๘	๙						
รหัสแอสกี้(ฐาน16)	F0	F1	F2	F3	F4	F5	F6	F7	F8	F9	FA	FB	FC	FD	FE	FF

ตาราง 7.1 แสดงรหัสความสัมพันธ์ของรหัสแอสกี้กับรหัสภาษาไทย



รูป 7.4 แสดงการแปลงจากรหัสแอสกี้เป็นบิทแม็พ

จากรูป 7.4 ซึ่งแสดงเป็นบิทแม็พก็จะเก็บอยู่ในรูปรหัสไบนารีแล้วมาแปลงเป็นรหัสฮัฟฟ์แมน 1 มิติ ดังขั้นตอนข้างล่างดังนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เก็บในรูปรหัสไบนารี

00000000 00000001 00000000 00000000 00000000 00000000 ...

00000000 00000010 10000000 00000000 00000000 00000000 ...

00000000 00000001 00000000 00000000 00000000 00000000 ...

00000000 00000000 00000000 00000000 00000000 00000000 ...

00000000 00000000 00000000 00000000 00000000 00000000 ...

00001000 11110000 00001111 11100000 00000111 11000000 ...

00010101 00 00100 00010000 00010000 00001000 00100000 ...

.....

นำแต่ละแถวมาเข้ารหัส ฮัฟฟ์แมน

EOL รหัสบิทขาว 17ตัว +รหัสบิทดำ 1ตัว +...จนครบ 1728ตัว

EOL รหัสบิทขาว 16ตัว +รหัสบิทดำ 1ตัว +รหัสบิทขาว 1ตัว +รหัสบิทดำ 1ตัว +...จนครบ 1728ตัว

EOL รหัสบิทขาว 17ตัว +รหัสบิทดำ 1ตัว +...จนครบ 1728ตัว

EOL รหัสบิทขาว 1728 ตัว

EOL รหัสบิทขาว 1728 ตัว

EOL รหัสบิทขาว 4ตัว +รหัสบิทดำ 1ตัว +รหัสบิทขาว 3ตัว +รหัสบิทดำ 4ตัว +รหัสบิทขาว 8ตัว +รหัสบิทดำ 5ตัว +จนครบ 1728ตัว

EOL รหัสบิทขาว 3ตัว +รหัสบิทดำ 1ตัว +รหัสบิทขาว 1ตัว +รหัสบิทดำ 1ตัว +รหัสบิทขาว 1ตัว +รหัสบิทดำ 1ตัว +รหัสบิทขาว 4ตัว +รหัสบิทดำ 1ตัว +รหัสบิทขาว 5ตัว +รหัสบิทดำ 1ตัว +รหัสบิทขาว 7ตัว +รหัสบิทดำ 1ตัว +รหัสบิทขาว 8 ตัว +รหัสบิทดำ 1ตัว +รหัสบิทขาว 5ตัว +รหัสบิทดำ 1ตัว +จนครบ 1728 ตัว

4. โปรแกรมชื่อ CONVERT.C

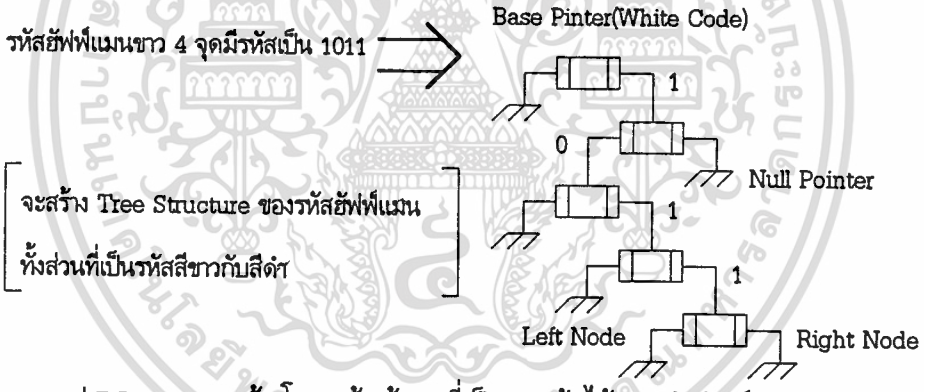
โปรแกรมนี้ใช้ในการแปลงฟอนต์ (font) ของโปรแกรมจุฬาเวิร์ดเพื่อมาใช้เป็นไฟล์บิทแม็พที่ไฟล์ฟอนต์สำหรับโปรแกรม CODER.C โดยใช้ฟอนต์จากไฟล์ชื่อ "normal.fon" ซึ่งส่วนประกอบภายในจะเก็บรูปแบบที่เป็นบิทแม็พของตัวอักษรแต่ละตัวไว้โดยรหัสแอสกี้จะมีตั้งแต่ 0-255 เป็นจำนวน 256 ตัวโดยจะเป็น20 ไบท์ต่อตัว ดังนั้นจะได้ขนาดทั้งหมดของไฟล์เป็น 5120 ไบท์ ขนาดของบิทแม็พของแต่ตัวอักษรจะเป็น8×20 จุด โดยที่แต่ละไบท์ของข้อมูลจะเก็บได้ 8 จุด รวม 20 ไบท์ต่อตัวอักษร สำหรับโปรแกรม CONVERT.C จะทำการขยายฟอนต์จากขนาด 8×20 จุด ต่อตัวอักษรเป็น 16×20 จุด ต่อตัวอักษร ดังนั้นจะต้องเก็บข้อมูล 16 ไบท์ต่อตัวอักษรโดยไฟล์ฟอนต์ที่แปลงขึ้นมาใหม่นี้จะเก็บลงในชื่อ "normal2.fon" มีขนาดของไฟล์เป็น 10240 ไบท์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

5. โปรแกรมชื่อ VIEW.C

โปรแกรมนี้ใช้สำหรับดูรายละเอียดภายในของไฟล์ที่เป็นรหัสฮัฟฟ์แมนที่ได้จากการแปลงของโปรแกรม CODER.C หรือจากการรับโทรสารโดยใช้โปรแกรม RECVFAX.C โดยในการแสดงผลนี้จะแสดงเป็นจุดขาวดำที่เป็นบิทแม็พตามรูปแบบของรหัสฮัฟฟ์แมนที่ในไฟล์ดังกล่าว โดยจะทำการอ่านข้อมูลไฟล์ที่เป็นฮัฟฟแมนที่ต้องการดูแล้วนำมาทำการสลับเรียงบิทก่อนโดยเรียงจากบิท 0-7 เป็น บิท 7-0 แล้วจึงไปเปรียบเทียบดูว่าตรงกับรหัสฮัฟฟ์แมนรหัสไหน ก็ได้ค่าของจำนวนจุดและเป็นสีขาวหรือดำ ดังนั้นสำหรับรหัสฮัฟฟ์แมน 1 แถวที่จบด้วยรหัส EOL จะประกอบด้วย 1728 จุด เมื่อแปลงเป็นไบท์ข้อมูลโดยใช้ 1 บิทต่อจุดก็จะได้ทั้งหมด 216 ไบท์แต่เนื่องจากในรหัสแอสกี้ 1 แถวจะแทนด้วยรหัสฮัฟฟ์แมน 20 แถวดังนั้นจะต้องเก็บทั้งหมด 216×20 = 4320 ไบท์ต่อรหัสแอสกี้ 1 แถว ต่อมาเมื่อแปลงเป็นบิทแม็พทั้งหมดก็จะทยอยนำแสดงผลบนจอโดยแต่ละครั้งจะแสดงจำนวนจุดได้มากที่สุด 640×480 จุด โดยโปรแกรมนี้จะสามารถกดปุ่มเลื่อนหน้าจอไปมาได้เพื่อความสะดวกในการเลือกดูตำแหน่งต่างๆ

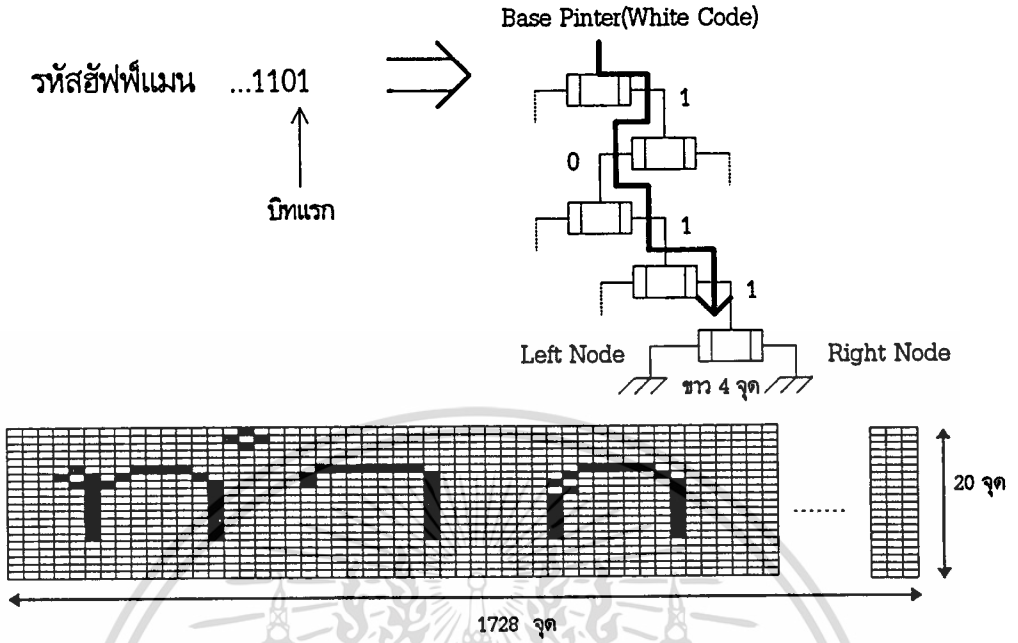
ในการแปลงกลับจากรหัสฮัฟฟ์แมนเป็นบิทแม็พนั้นโปรแกรมจะทำการสร้างโครงสร้างข้อมูลที่เป็นแบบต้นไม้ (Tree Structure) เพื่อใช้ในการถอดรหัสกลับเป็นบิทแม็พโดยได้แสดงการสร้างไว้ในรูป 7.5



รูป 7.5 แสดงการสร้างโครงสร้างข้อมูลที่เป็นแบบต้นไม้ของรหัสฮัฟฟ์แมน 1 มิติ

เมื่อได้สร้างโครงสร้างข้อมูลที่เป็นแบบต้นไม้ของรหัสฮัฟฟ์แมนของรหัสสีขาวและสีดำแล้วก็จะนำเอาไบท์ข้อมูลที่เป็นรหัสฮัฟฟ์แมนมาแล้วทำการเช็คไปทีละบิทโดยถ้าบิทใดเป็น 1 ก็จะไปตาม โหนดขวา (Right Node) ของโครงสร้างต้นไม้ส่วนในกรณีที่เป็น 0 ก็จะไปทางโหนดซ้าย (Left Node) ไปอย่างนี้เรื่อยๆจนกว่าจะพบว่าทั้งโหนดซ้ายและโหนดขวาจะเป็นโหนดว่าง ก็จะเป็นอันสิ้นสุดการหารหัสฮัฟฟ์แมน ซึ่งในแต่ละโหนดนั้นจะใส่ค่าจำนวนของจุดขาวหรือดำที่สัมพันธ์กับรหัสฮัฟฟ์แมนดังกล่าว ทำให้ทราบว่าเป็นจุดสีขาวหรือสีดำกี่จุด แล้วนำมาแสดงผลทางหน้าจอเป็น ในรูป 7.6 จะแสดงถึงวิธีการถอดรหัสฮัฟฟ์แมนจนได้เป็นค่าของจุดสีดำหรือขาวแล้วนำไปแสดงผลต่อทางหน้าจอของคอมพิวเตอร์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

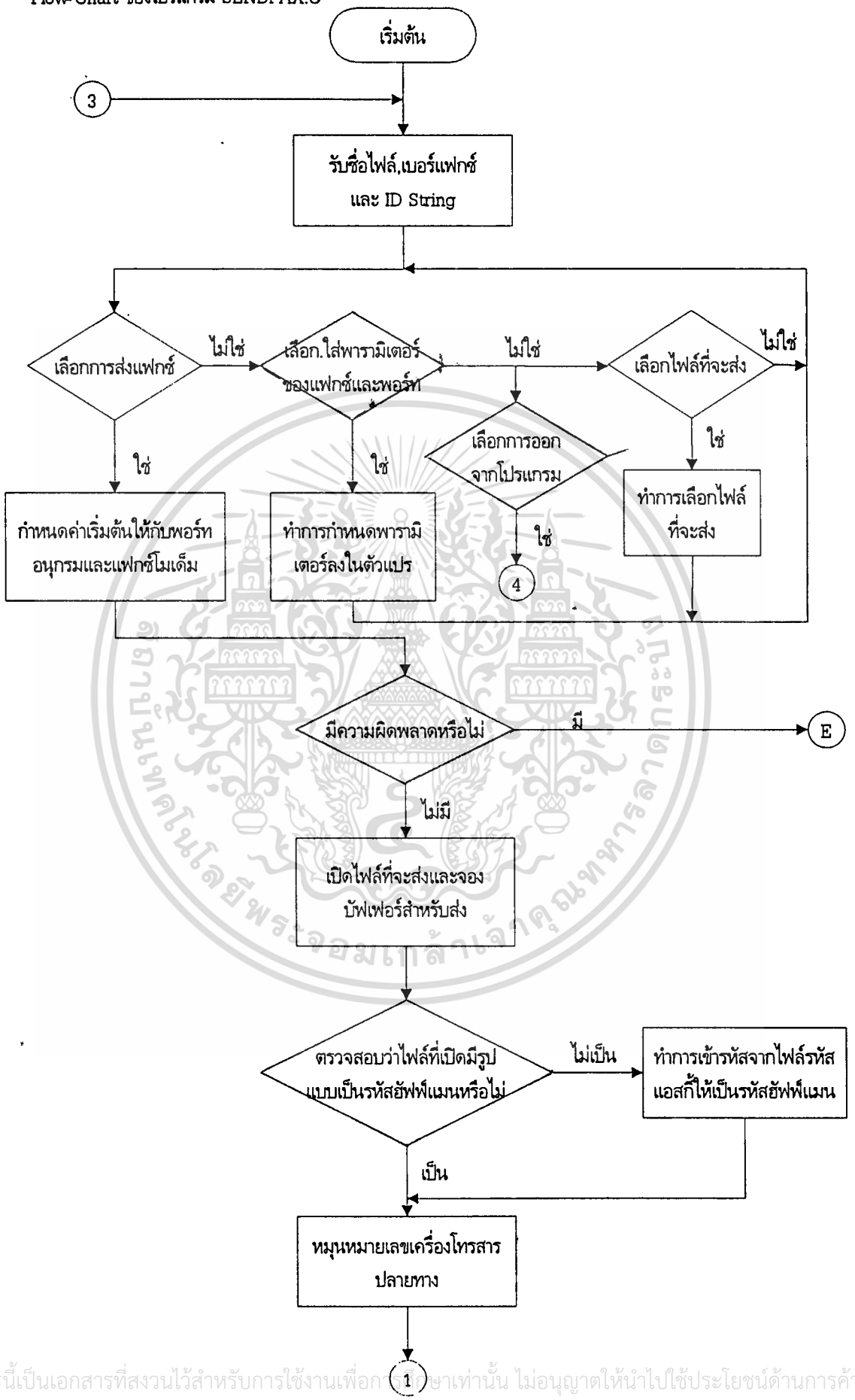


รูป 7.6 แสดงการถอดรหัสฮัฟฟ์แมนจนได้เป็นบิทแม็พเพื่อนำไปแสดงผลทางจอคอมพิวเตอร์

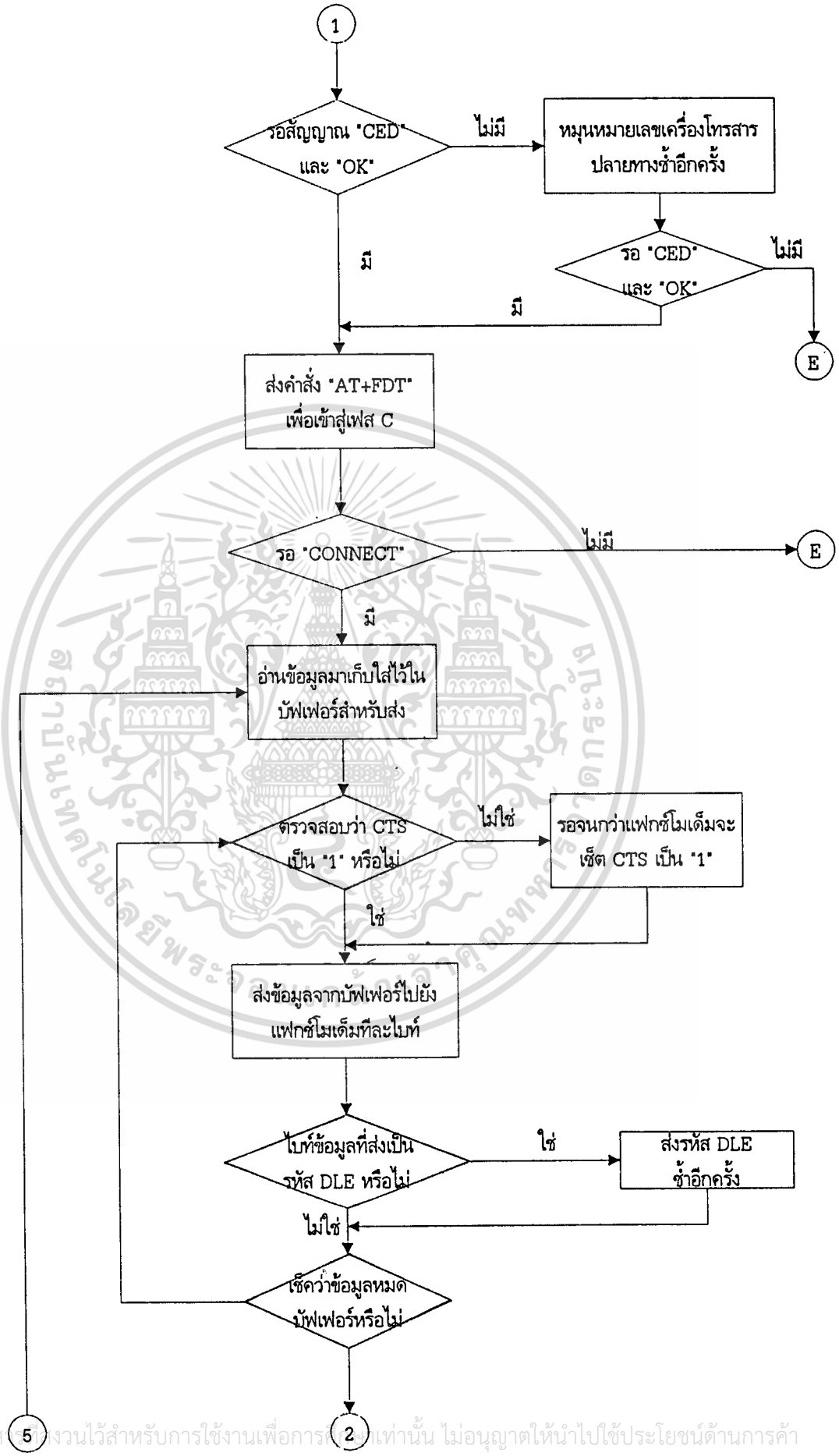
Flow Chart แสดงการทำงานของแต่ละโปรแกรม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

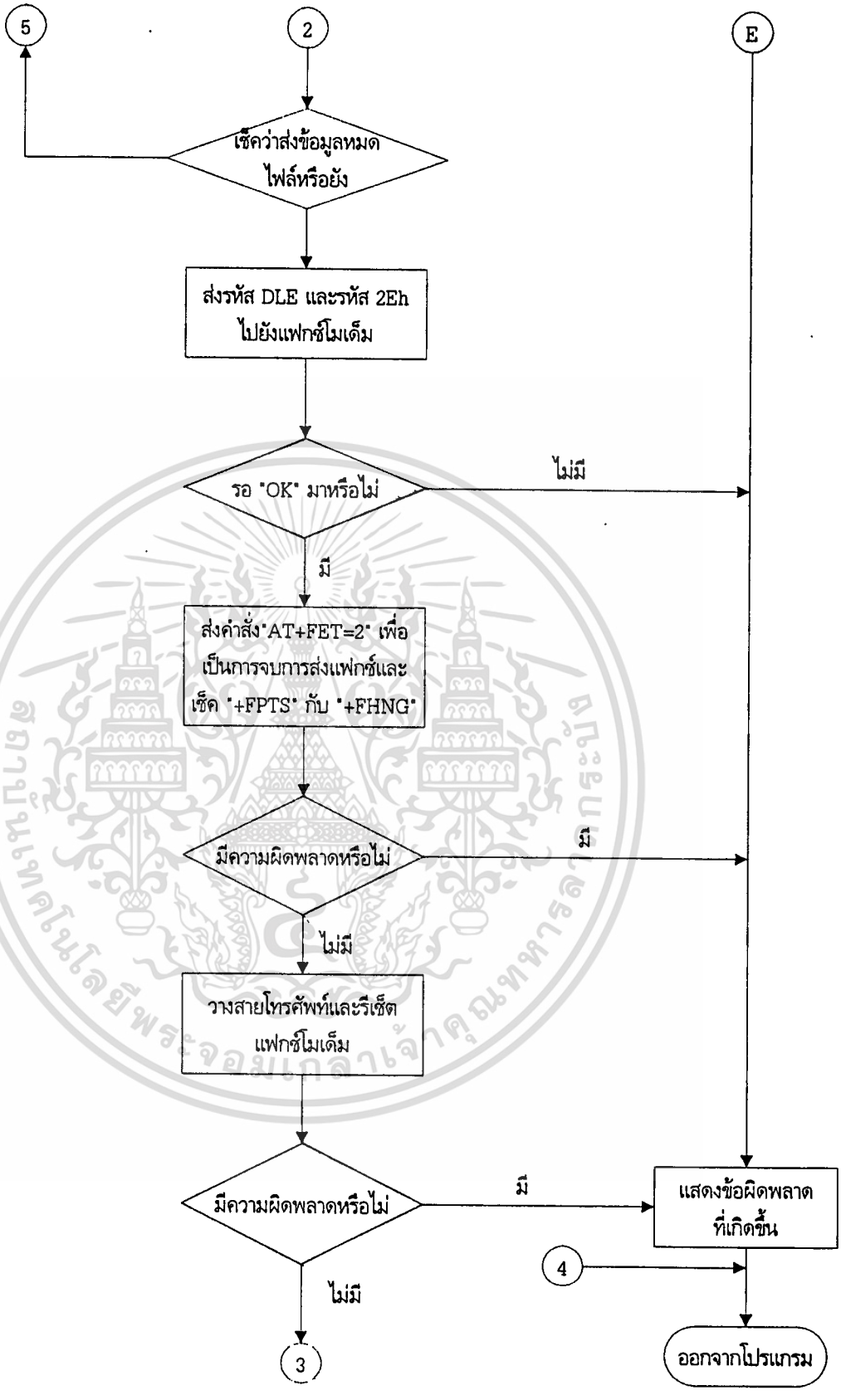
Flow-Chart ของโปรแกรม SENDFAX.C



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

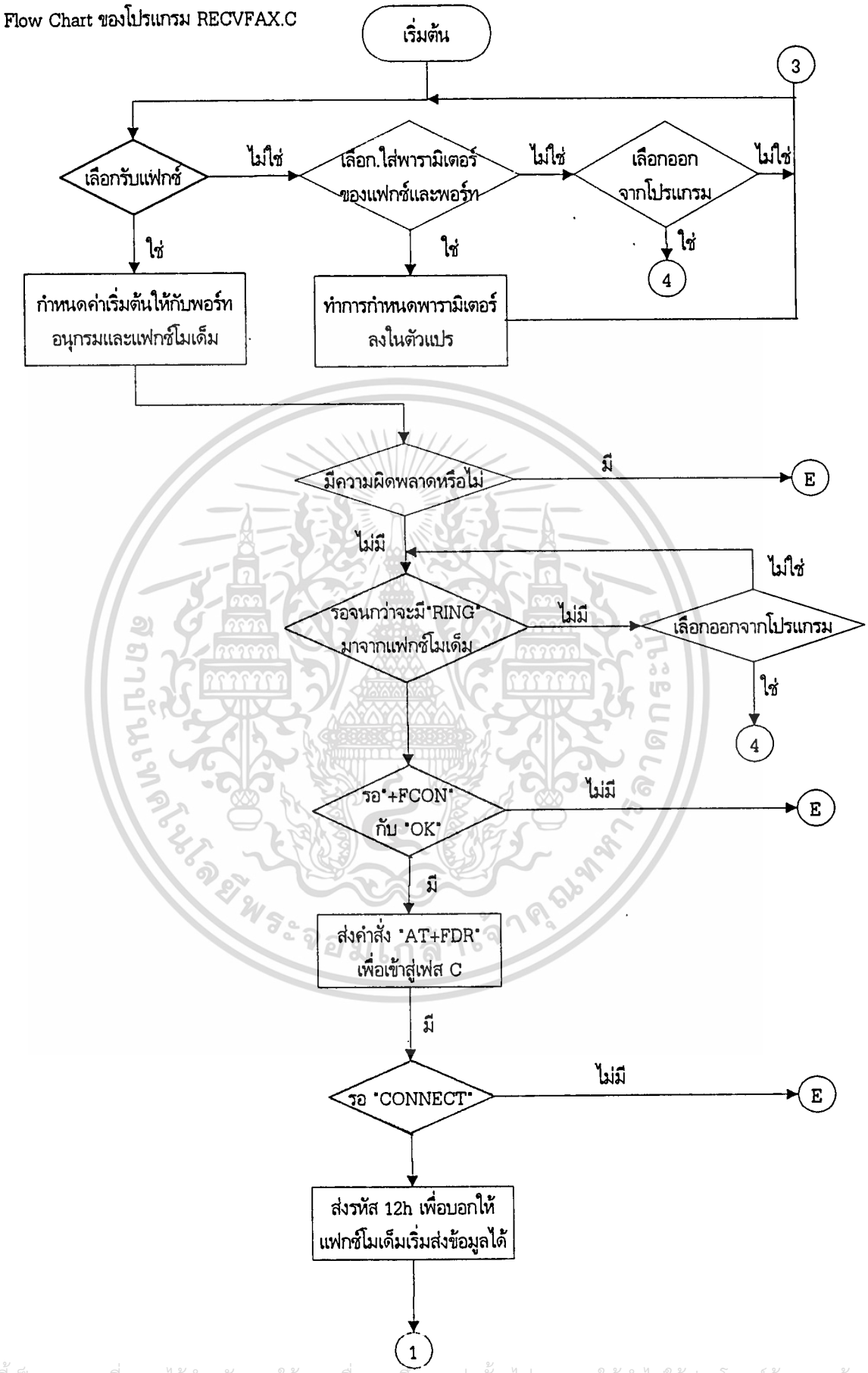


เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

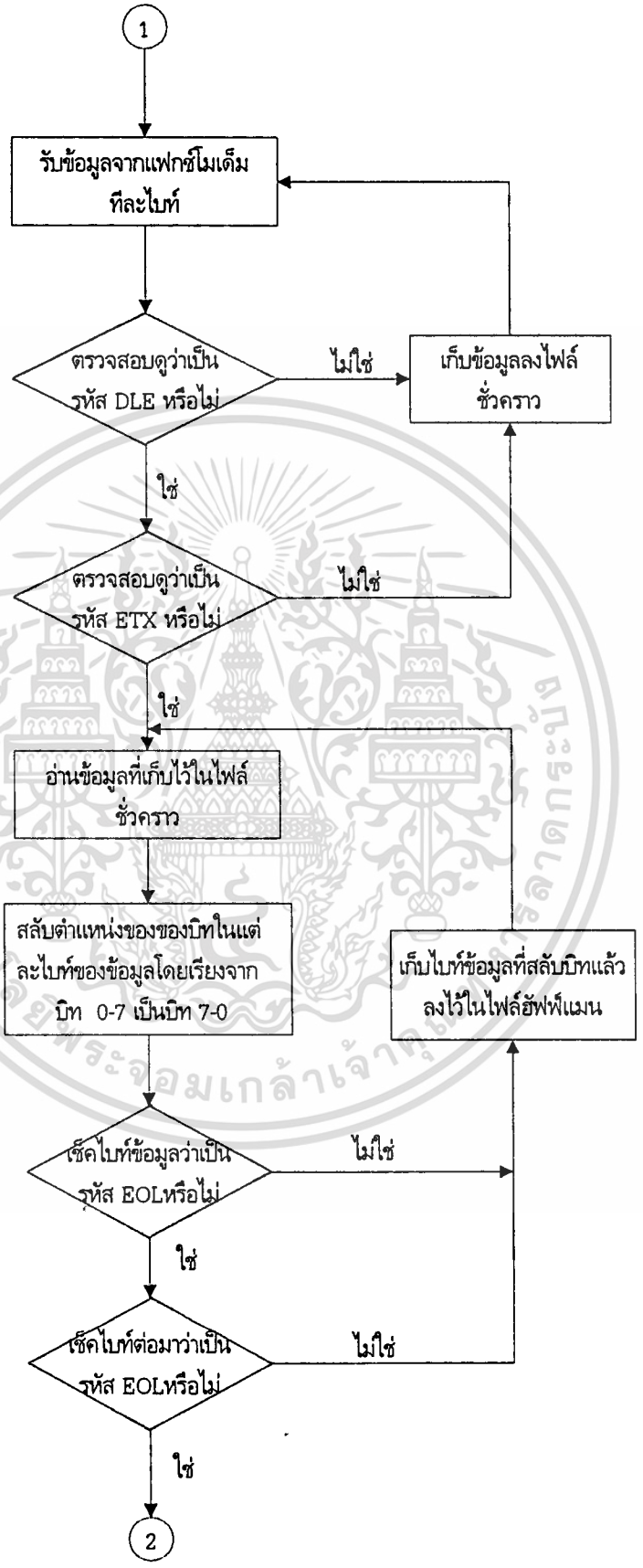


เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

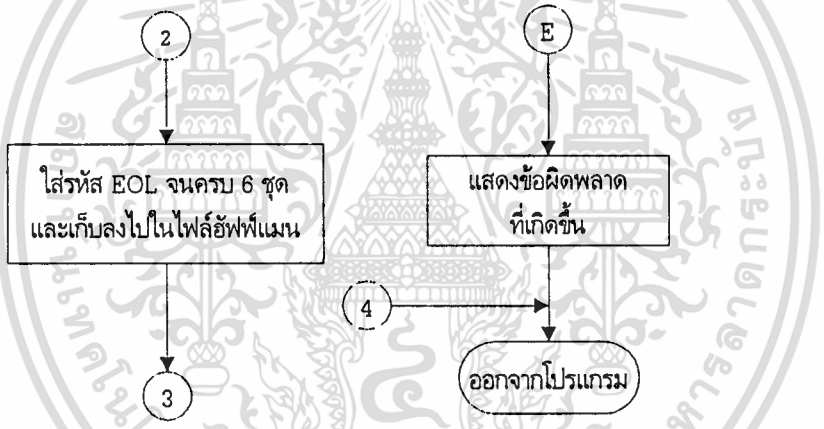
Flow Chart ของโปรแกรม RECVFAX.C



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

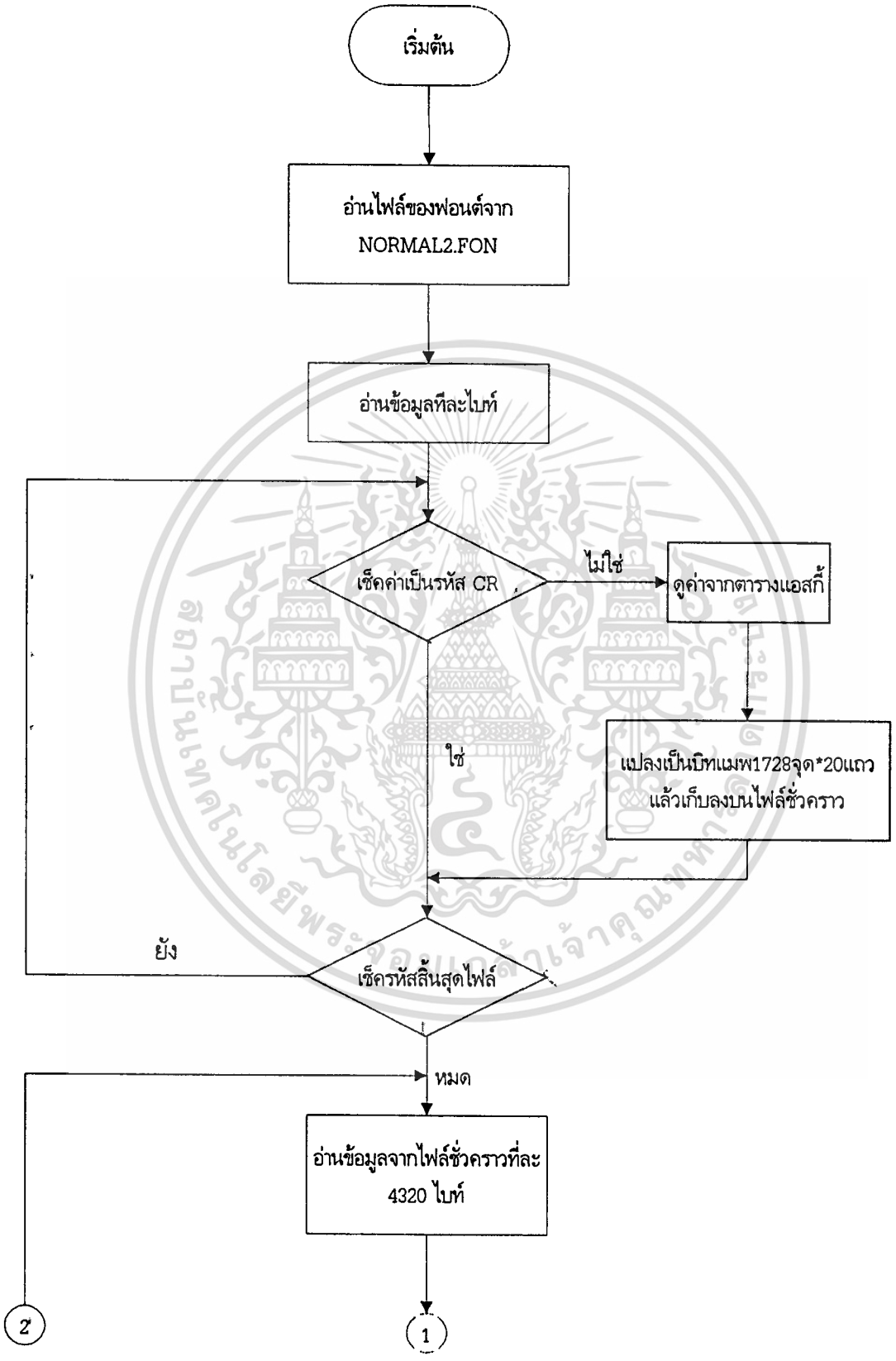


เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

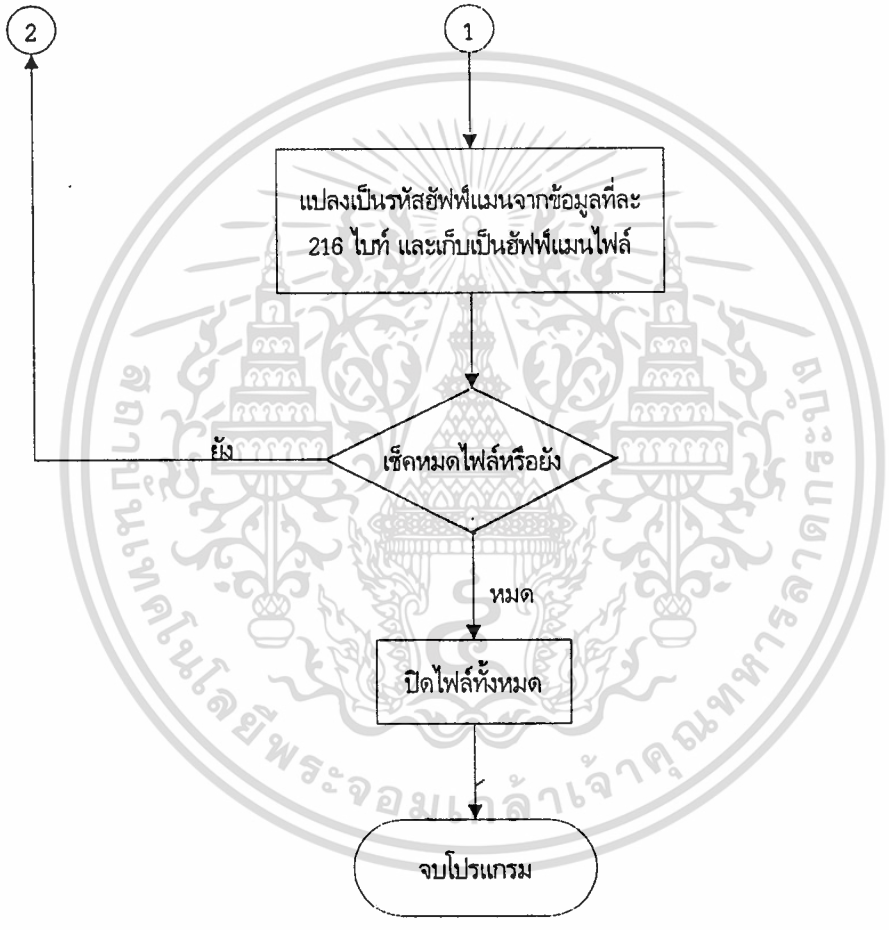


เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Flow Chart ของโปรแกรม CODER.C



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

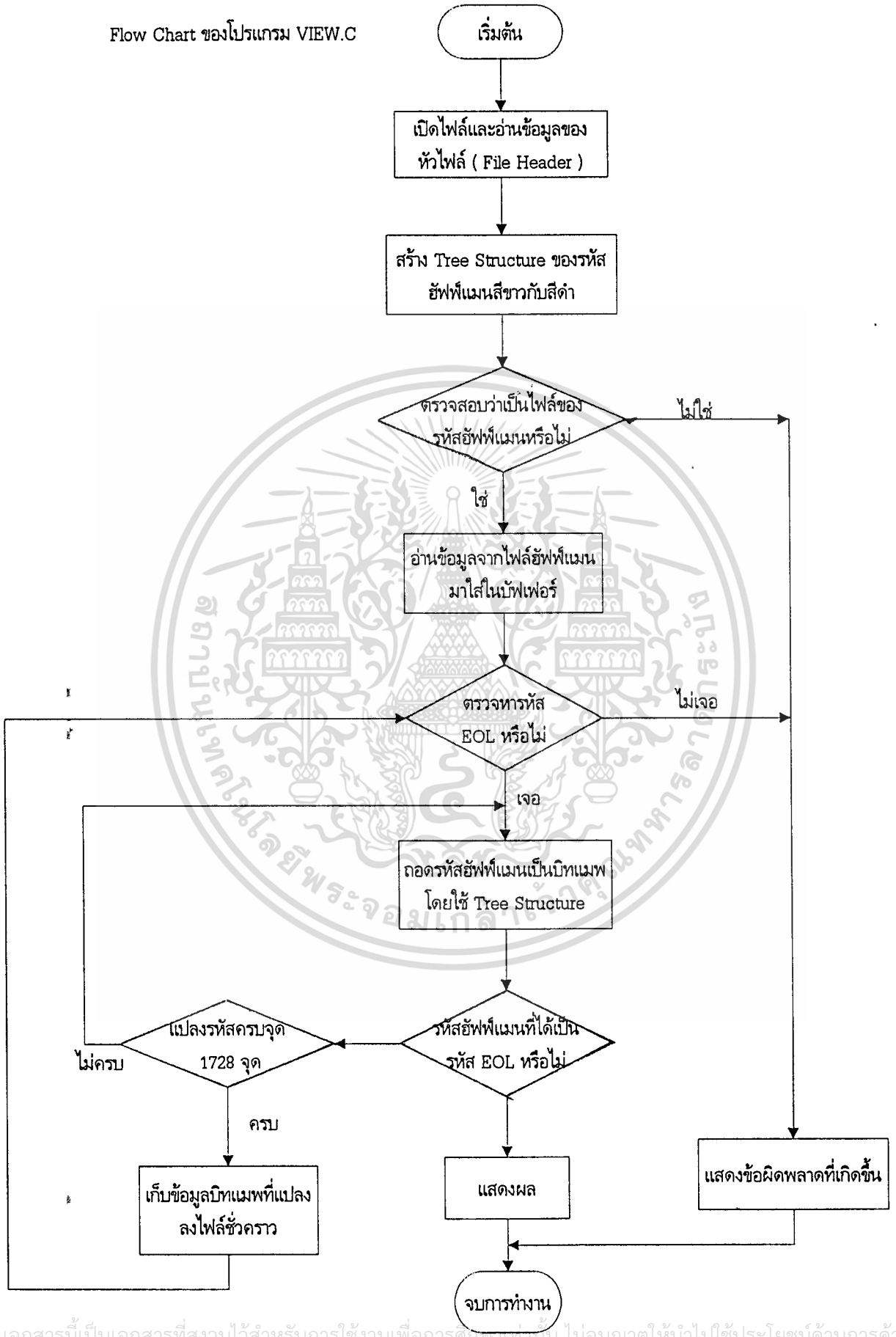


เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Flow Chart ของโปรแกรม CONVERT.C

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Flow Chart ของโปรแกรม VIEW.C



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 8

ผลการทดลองและสรุปผลการทดลองในการรับ/ส่งโทรสาร

ขั้นตอนการทดลองรับ/ส่งโทรสารจะแบ่งออกเป็น 3 แบบ

1.การส่งโทรสารจากแฟกซ์โมเด็มไปยังเครื่องโทรสารธรรมดา

ในการทดลองจะต่อพ่วงแฟกซ์โมเด็มเข้ากับคอมพิวเตอร์ในห้อง แล้วนำแฟกซ์โมเด็มมาต่อคู่สายโทรศัพท์เบอร์ภายใน 539 แล้วใช้โปรแกรม SENDFAX.C ที่เขียนขึ้นมาเองทำการส่งโทรสารไปยังโทรศัพท์เบอร์ภายใน 540 ซึ่งได้ต่อพ่วงกับเครื่องโทรสารกลุ่ม 3ไว้

2.การส่งโทรสารจากเครื่องโทรสารธรรมดาไปยังแฟกซ์โมเด็ม

ในการทดลองจะต่อพ่วงแฟกซ์โมเด็มเข้ากับคอมพิวเตอร์ในห้อง แล้วนำแฟกซ์โมเด็มมาต่อคู่สายโทรศัพท์เบอร์ภายใน 540 แล้วใช้โปรแกรม RECVFAX.C ที่เขียนขึ้นมาเองเพื่อทำการรับโทรสารที่ส่งมาจากเครื่องโทรสารกลุ่ม 3 เบอร์ภายใน 552

3.การรับ/ส่งโทรสารระหว่างแฟกซ์โมเด็มด้วยกันเอง

ในการทดลองจะต่อพ่วงแฟกซ์โมเด็มเข้ากับคอมพิวเตอร์ในห้อง แล้วนำแฟกซ์โมเด็มมาต่อคู่สายโทรศัพท์เบอร์ภายใน 539 แล้วใช้โปรแกรม SENDFAX.C ที่เขียนขึ้นมาเองทำการส่งโทรสารไปยังโทรศัพท์เบอร์ 540 ซึ่งได้ต่อพ่วงกับแฟกซ์โมเด็มและคอมพิวเตอร์ โดยใช้โปรแกรม RECVFAX.C ในการรับโทรสาร

ในการทดลองในขั้นตอนที่ 1 และ 3 ที่ส่งโทรสารโดยใช้แฟกซ์โมเด็มจะส่งไฟล์ที่มีข้อความดังต่อไปนี้

นี้เป็นการทดสอบการส่งแฟกซ์ โดยส่งผ่านแฟกซ์โมเด็ม (FCLASS 2)

ตัวพยัญชนะภาษาไทย

ก ข ฃ ค ฅ ฆ ง จ ฉ ช ซ ฌ ญ ฎ ฏ
ฐ ฑ ฒ ณ ด ต ถ ท ธ น บ ป ผ ฝ พ
ฟ ภ ม ย ร ฤ ล ฦ ว ศ ษ ส ห ฬ อ
ฮ

ตัวอักษรภาษาอังกฤษ

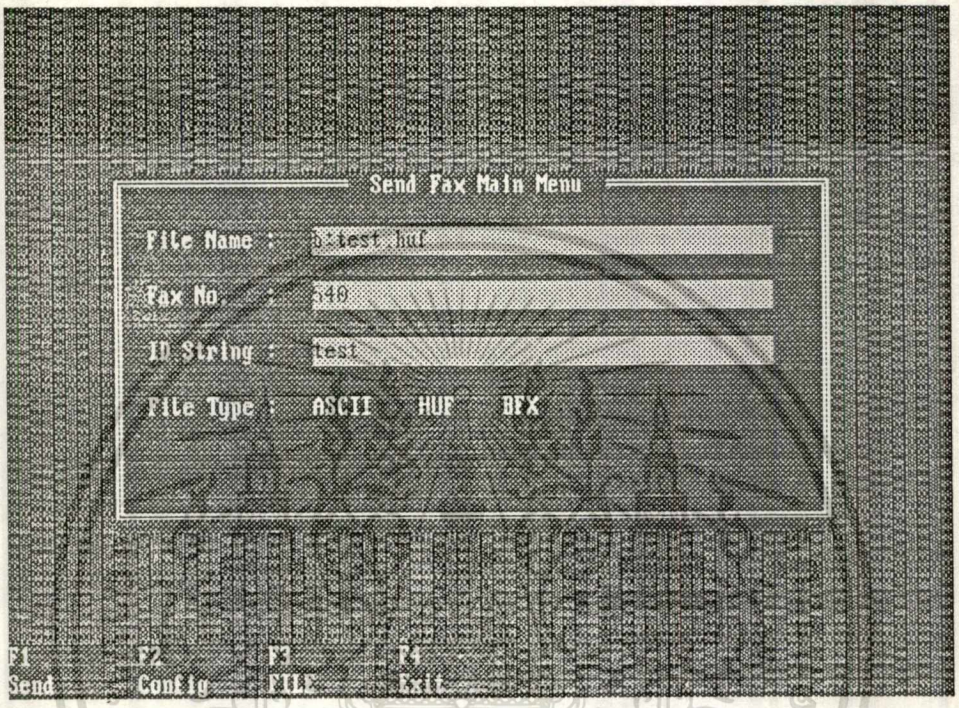
A a B b C c D d E e F f G g H h I i
J J K k L l M m N n O o P p Q q R r
S s T t U u V v W w X x Y y Z z

ผลการทดลอง

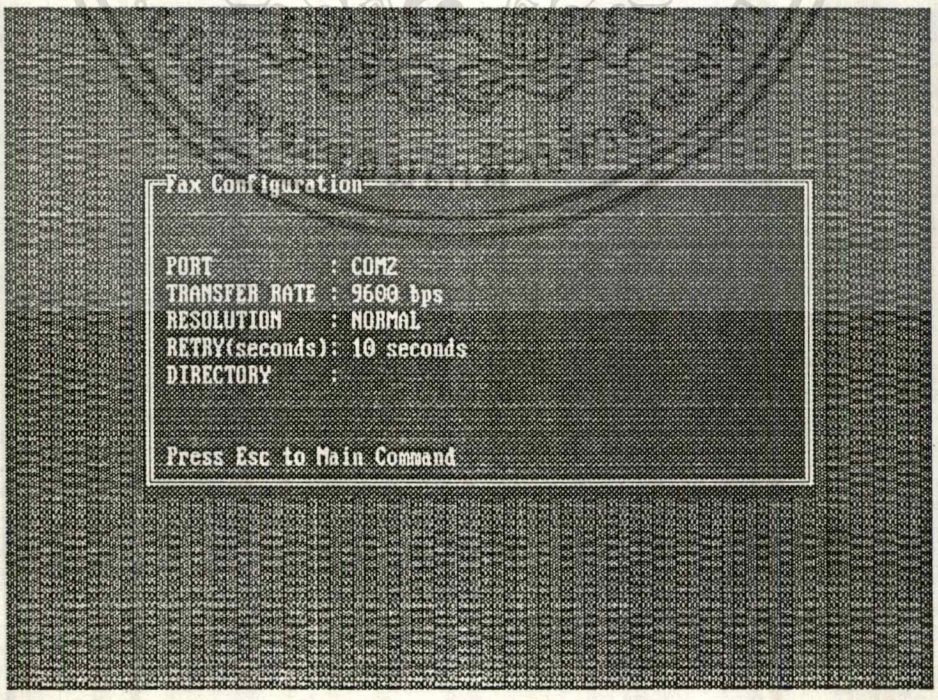
ผลที่ได้จากโปรแกรม SENDFAX.C ในขั้นตอนการทดลองที่ 1 และ 3 จะแสดงในส่วนหน้าจอที่โปรแกรมนี้กำลังทำงานอยู่จนจบขั้นตอนการส่งโทรสาร มี 3 รูป คือ รูป 8.1 จะแสดงส่วนที่เป็นเมนู

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

(MENU) หลักของโปรแกรมซึ่งให้ผู้ใช้ใส่พารามิเตอร์ต่างๆเช่นเบอร์โทรศัพท์ ,ชื่อไฟล์ เป็นต้น รูป 8.2 จะเป็นส่วนของการใส่พารามิเตอร์เกี่ยวกับพอร์ทอนุกรมและความเร็วในการส่งโทรสารหรือความละเอียดที่ใช้ รูป 8.3 จะเป็นส่วนที่โปรแกรมได้ทำการส่งโทรสารเรียบร้อยโดยไม่มีปัญหาเกิดขึ้น

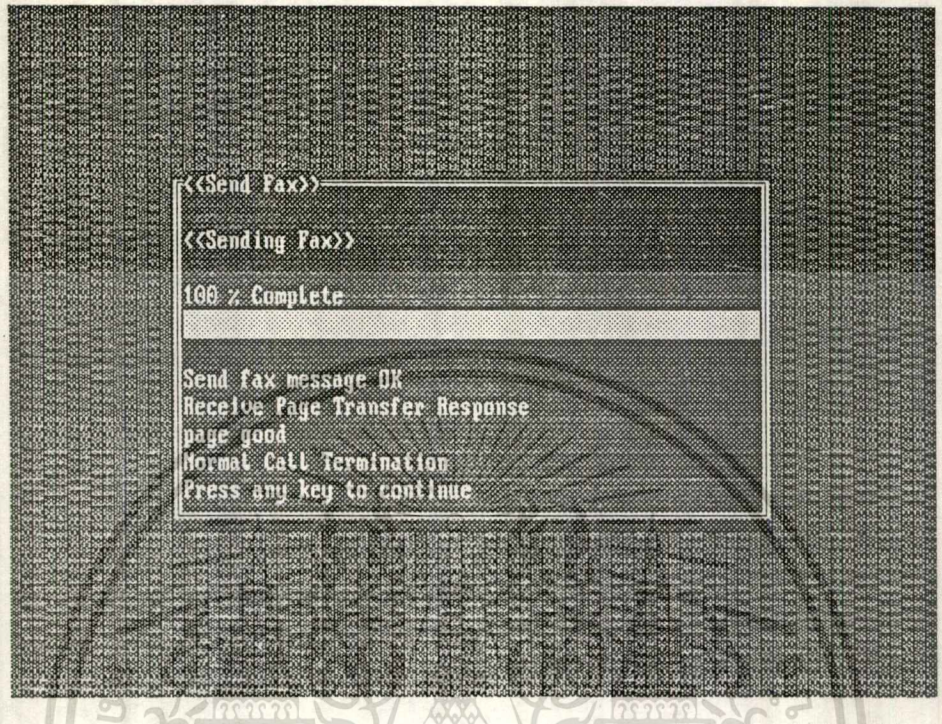


รูป 8.1 แสดงส่วนเมนูหลักของโปรแกรม SENDFAX.C



รูป 8.2 แสดงในส่วนที่ต้องใส่พารามิเตอร์ต่างๆในโปรแกรม SENDFAX.C

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูป 8.3 แสดงหน้าจอของโปรแกรม SENDFAX.C เมื่อทำการส่งโทรสารเสร็จ

ผลที่ได้จากการรับโทรสารโดยใช้เครื่องโทรสารกลุ่ม 3 ในขั้นตอนที่ 1 จะแสดงไว้ในรูป 8.4

นี่เป็นการทดสอบการส่งแฟกซ์ โดยส่งผ่านแฟกซ์โมเด็ม (FCLASS 2)

ตัวพยัญชนะภาษาไทย

ก ข ฃ ค ฅ ฆ ง จ ฉ ช ซ ฌ ญ ฎ ฏ
ฐ ฑ ฒ ณ ด ต ถ ท ธ น บ ป ผ ฝ พ
ฟ ภ ม ย ร ฤ ล ฦ ว ศ ษ ส ห ฬ อ
ฮ

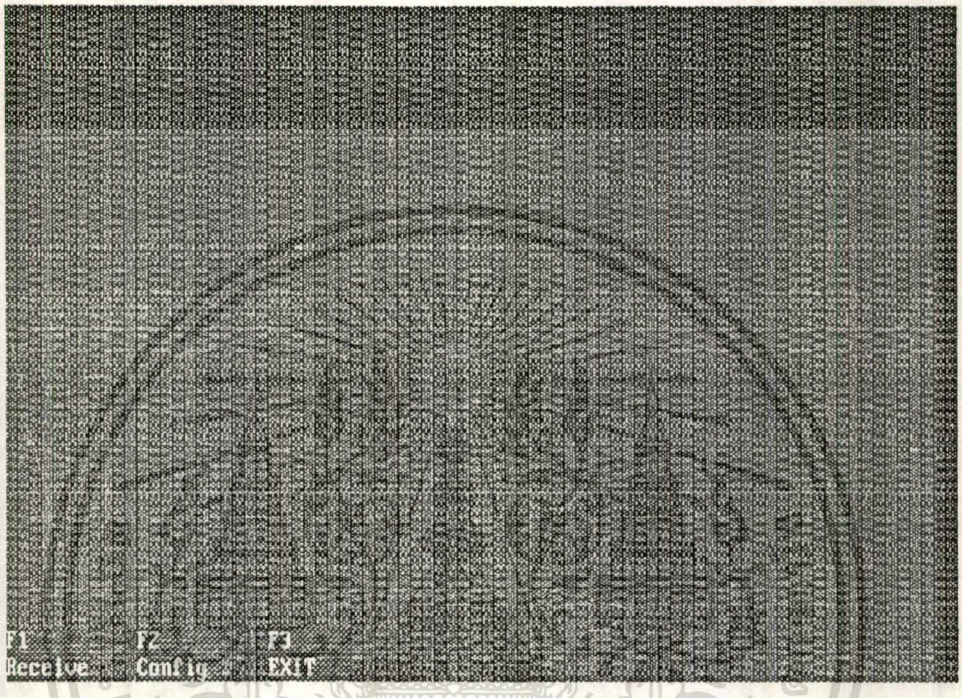
ตัวอักษรภาษาอังกฤษ

A a B b C c D d E e F f G g H h I i
J j K k L l M m N n O o P p Q q R r
S s T t U u V v W w X x Y y Z z

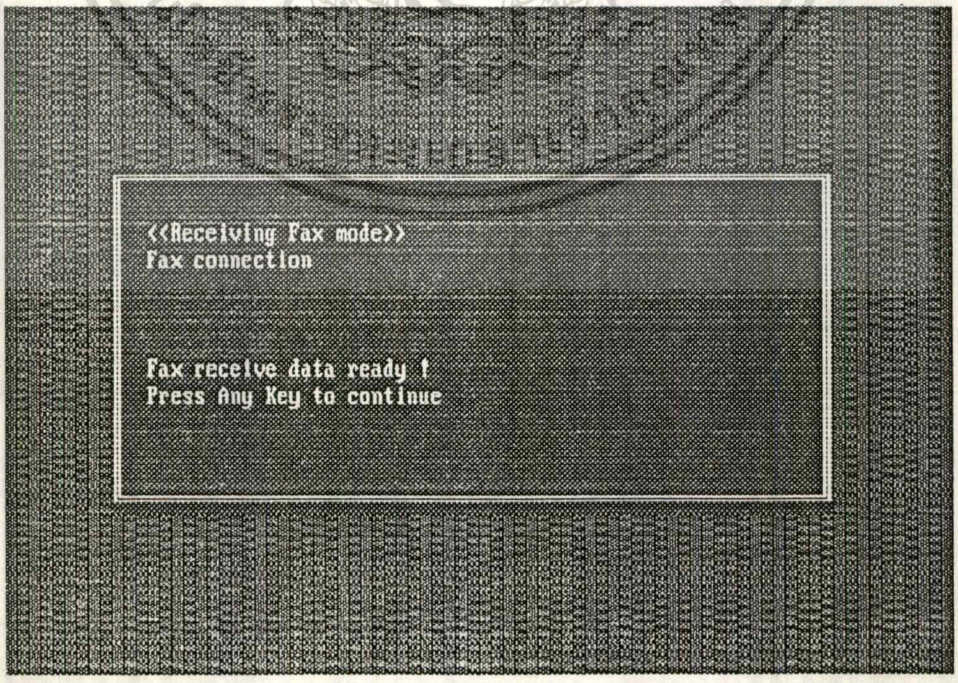
รูป 8.4 แสดงผลที่ได้จากการส่งโทรสารไปยังเครื่องโทรสาร

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ผลที่ได้จากโปรแกรม RECVFAX.C ในขั้นตอนที่ 2 และ 3 จะแสดงส่วนหน้าจอที่โปรแกรมนี้กำลังทำงานอยู่ในระหว่างขั้นตอนการรับโทรสาร มี 2 รูป คือ รูป 8.5 จะแสดงเมนูหลักของโปรแกรมรับโทรสาร และ รูป 8.6 แสดงการทำงานของโปรแกรมในช่วงที่สิ้นสุดการรับโทรสารโดยไม่มีปัญหาเกิดขึ้น



รูป 8.5 แสดงส่วนเมนูหลักของโปรแกรม RECVFAX.C



รูป 8.6 แสดงการทำงานของโปรแกรม RECVFAX.C เมื่อรับโทรสารเสร็จโดยไม่มีปัญหาเกิดขึ้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- ในการส่งโทรสารบางครั้งข้อมูลโทรสารจะมีลักษณะตรงกับคำสั่ง DLE ของโมเด็ม ซึ่งจะต้องทำการส่งข้อมูลนั้นซ้ำเพื่อแสดงว่าข้อมูลนั้นไม่ใช่คำสั่ง DLE ของโมเด็ม แต่เป็นส่วนของข้อมูลโทรสาร จึงจะสามารถรับส่งข้อมูลโทรสารนั้นได้ถูกต้อง

- ในส่วนของโปรแกรมรับ จะพบว่าในส่วนตอนเริ่มต้นและสิ้นสุดของข้อมูลโทรสารที่รับได้ จะมีส่วนที่ไม่ใช่ข้อมูลที่แท้จริงอยู่ซึ่งต้องทำการตัดออกก่อนแสดงผล

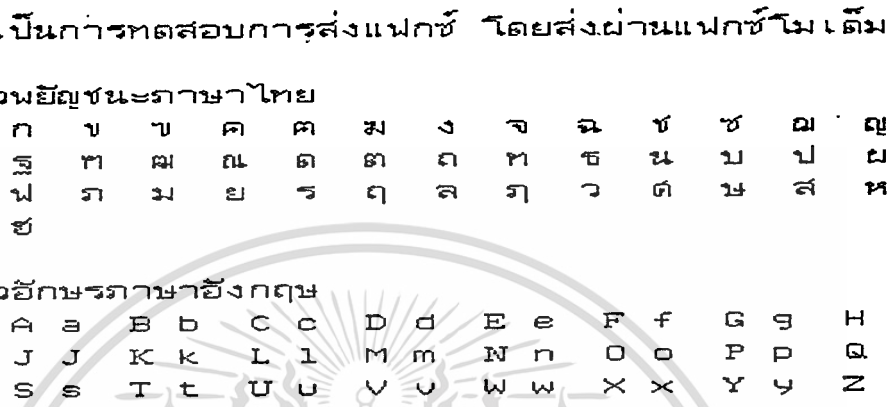
แนวทางการพัฒนา

โปรแกรมต่างๆ ในการทดลองนี้สามารถพัฒนาเพิ่มเติมส่วนต่างๆ อันจะอำนวยความสะดวกให้แก่ผู้ใช้ได้ เช่น

- เพิ่มส่วนของโปรแกรมที่ใช้พิมพ์โทรสารไฟล์ที่รับมาให้ออกทางเครื่องพิมพ์ได้
- เพิ่มส่วนอีดิเตอร์ (Editor) เพื่อใช้ในการพิมพ์ข้อความในการส่งโทรสาร
- พัฒนาโปรแกรมรับ ให้สามารถทำงานแบบฝังตัวในหน่วยความจำ(resident program)
- พัฒนาโปรแกรมรับ/ส่งโทรสารให้มีความสามารถด้านระบบ LAN(Local Area Network)โดยให้มีขีดความสามารถเป็นแฟกซ์เซิร์บเวอร์ (FAX SERVER) ได้ในอนาคต

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หลังจากที่ใช้โปรแกรม RECVFAX.C แล้วในขั้นตอนที่ 2 และ 3 แล้วไฟล์ที่รับมาได้จะใช้โปรแกรม VIEW.C มาแสดงผลว่าไฟล์ที่รับมาเป็นอย่างไร ดังรูป 8.7



รูป 8.7 แสดงผลจากการรับโทรสารมาแล้วใช้โปรแกรม VIEW.C มาดูรายละเอียดของไฟล์

สรุปผลการทดลอง

จากการทดลองนี้ได้ทำการรับส่งโทรสารระหว่างคอมพิวเตอร์กับเครื่องโทรสารกลุ่ม 3 และระหว่างคอมพิวเตอร์กับคอมพิวเตอร์ ซึ่งในการทดลองครั้งนี้ได้ทำการแก้ไขปัญหา และข้อผิดพลาด ต่างๆที่เกิดขึ้นในเทอมที่แล้ว ทำให้สามารถรับส่งโทรสารได้ถูกต้องและชัดเจนยิ่งขึ้น นอกจากนี้ยังทำการเขียนโปรแกรมเข้าและถอดรหัสฮัฟฟ์แมน เพื่ออำนวยความสะดวกในการส่งข้อมูลภาษาไทยจาก จุฬาเวิร์ด และราชวิถีเวิร์ดได้ด้วย โดยส่วนสำคัญในการทดลองครั้งนี้คือ ไทม์มิ่ง- (timing) ของแต่ละขั้นตอนในการทำงานจะต้องสอดคล้องกัน เช่น ช่วงเวลาในการส่งข้อมูลโทรสารแต่ละเส้นสแกน จะต้องไม่มากหรือน้อยกว่าช่วงเวลาที่กำหนดไว้(20 miliseconds - 5 seconds) นอกจากโปรแกรมที่เขียนไว้สำหรับรับ/ส่งโทรสารแล้วยังได้เขียนเพิ่มเติมไว้สำหรับเชื่อมต่อกับกลุ่มโครงงานชื่อ จดหมายอิเล็คทรอนิกส์ภาษาไทย โดยสามารถให้โปรแกรมจดหมายอิเล็คทรอนิกส์เรียกใช้โปรแกรม SENDFAX.C เพื่อทำการส่งโทรสารสำหรับไฟล์จดหมายอิเล็คทรอนิกส์นั้นจะนำมาผ่านการแปลงให้เป็นไฟล์รหัสฮัฟฟ์แมนโดยใช้โปรแกรม CODER.C ในการเชื่อมต่อกันได้กำหนดให้มีการส่งผ่านพารามิเตอร์มายังโปรแกรม SENDFAX.C ดังนี้

SENDFAX 1 ชื่อไฟล์หรือจดหมายอิเล็คทรอนิกส์ที่ส่ง เบอร์โทรศัพท์ปลายทาง

โปรแกรมจดหมายอิเล็คทรอนิกส์ภาษาไทยจะทำการเรียกโปรแกรม SENDFAX โดยส่งผ่านพารามิเตอร์มาตามที่ได้ตกลงกันไว้ก็จะทำให้สามารถที่ใช้โปรแกรมจดหมายอิเล็คทรอนิกส์มาส่งโทรสารได้

นอกจากนี้ยังพบปัญหาต่างๆ ระหว่างทำการทดลอง ได้แก่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก

คำสั่งแฟกซ์โมเด็มระดับ 2

+FAA(Select Fax Auto Answer Mode : โหมดตอบรับอัตโนมัติ)

รูปแบบ : AT+FAA = n

+FFA = 0 ตอบรับเมื่อแฟกซ์โมเด็มเลือกใช้เฉพาะโหมดที่กำหนดโดยคำสั่ง +FCLASS

+FAA = 1 ตรวจจับข้อมูลและการเรียกอัตโนมัติ และตอบรับตามความเหมาะสม

ค่าเริ่มต้น: +FAA = 0

ในโหมดตอบรับอัตโนมัติ โมเด็มจะปรับตัวเองโดอัตโนมัติและตอบรับทันที

+FAXERR (Fax T.30 Error Response: ผลแสดงความผิดพลาดตามมาตรฐานโทรสาร T.30)

รูปแบบ: AT+FAXERR = n

n คือ รหัสแสดงสถานะการวางสาย

แสดงถึง สาเหตุของการวางสาย และถูกกำหนดโดยโมเด็มที่อยู่ในระว่างการสิ้นสุดในแต่ละขั้นตอน โมเด็มจะปรับให้พารามิเตอร์นี้เป็น 0 เมื่อเริ่มต้นแต่ละขั้นตอน ค่าของ n แสดงในตารางผลตอบสนอง

+FHNG

+FCFR(Confirmation to Receive Response : การยืนยันการรับผลตอบสนอง)

รูปแบบ : AT+FCFR

โมเด็มท้องถิ่น (local modem) จะส่งผลตอบสนองนี้ไปยัง DTE หลังจากได้รับสัญญาณ DCS และ TFC ที่ถูกต้องจากเครื่องโทรสารที่อยู่ไกลออกไป

+FCLASS (Service Class Identification and Control : แสดงลักษณะและการควบคุม)

รูปแบบ :AT+FCLASS? :เป็นการถามถึงกลุ่มของคำสั่งที่ใช้ในขณะนี้

AT+FCLASS=? :เป็นการถามถึงความสามารถของโมเด็มในการใช้คำสั่งในระดับต่างๆ

AT+FCLASS=n :เป็นการกำหนดระดับของคำสั่งที่ใช้ในโมเด็ม

คำสั่งนี้ใช้สำหรับการกำหนดและถามโหมดการทำงานของแฟกซ์โมเด็มและความสามารถ ซึ่งโมเด็มจะตอบกลับดังนี้

1 : EIA/TIA-578 ระดับ 1

2 : กำหนดโดยโรงงาน (SP-2388-A ระดับ 2)

3 : TIA/EIA-592 ระดับ 2.0

+FCON(Fax Connection Response : ผลตอบสนองการติดต่อของเครื่องโทรสาร)

รูปแบบ : +FCON

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โมเด็มจะแสดงผลนี้เพื่อบอกให้รู้ว่า การติดต่อระหว่างเครื่องโทรสารประสบความสำเร็จ โดยจะถูกส่งโดยโมเด็มท้องถิ่นไปยัง DTE หลังจากที่ได้รับแฟลก (flag) แรกของ HDLC

+FCR(Set Capability to Receive : การกำหนดความสามารถในการรับ)

รูปแบบ : AT+FCR = n

+FCR = 0 โมเด็มจะไม่รับข้อมูลและไม่สามารถโพล (poll)

+FCR = 1 โมเด็มจะสามารถรับข้อมูลข่าวสารได้

พารามิเตอร์นี้จะถูกพิจารณาในระหว่างเฟส A และ D

+FDIS(Set Current Session Negotiation Parameter : การกำหนดพารามิเตอร์ตามการตกลงกันในขบวนการ)

รูปแบบ : AT+FDIS = *s*

s คือรหัสของพารามิเตอร์ย่อยในขบวนการ T.30

ค่าสั่งนี้จะเป็นการให้ DTE กำหนดค่าของ DIS เฟรมโดยตรงไปให้กับ DCE

Label	Function	Values	Description
VR	Vertical resolution	0	Normal, 98 lpi
		1	Fine, 196 lpi
BR	Data transfer speed[2]	0	2400 bps, V.27ter
		1	4800 bps, V.27ter
		2[1]	7200 bps, V.29 or V.17
		3[1]	9600 bps, V.29 or V.17
		4[1]	12000 bps, V.33 or V.17
		5[1]	14400 bps, V.33 or V.17
WD	Page width	0	1728 pixels in 215 mm
		1[1]	2048 pixels in 255 mm
		2[1]	2432 pixels in 303 mm
		3[1]	1216 pixels in 151 mm
		4[1]	864 pixels in 107 mm
LN	Page length	0	A4, 297 mm
		1[1]	B4, 364 mm
		2[1]	Unlimited length
DF	Data compression format	0	1-dimensional, modified Huffman

ตารางที่ ผ.1 แสดงค่า *s* ใน T.30

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Label	Function	Values	Description
		1[1]	1-dimensional, modified Read
		2[1]	2-dimensional, uncompressed
		3[1]	2-dimensional, modified Read
EC	Error correction	0	Disable ECM
		1[1]	Enable error correction mode, 64 bytes/frame
		2[1]	Enable error correction mode, 256 bytes/frame
BF	Binary file transfer	0	Disable binary file transfer
		1[1]	Enable binary file transfer
ST	Scan time per line	0	0 ms
		1	5 ms
		2	10 ms normal, 5 ms fine
		3	10 ms
		4	20 ms normal, 10 ms fine
		5	20 ms
		6	40 ms normal, 20 ms fine
		7	40 ms

[1] Optional

[2] The encoding specified for the DIS frame in CCITT T.30 does not allow all speeds to be specified exactly. The identification of some BR speeds will therefore be manufacturer-specific.

ตารางที่ ผ.1 (ต่อ)แสดงค่า *s* ใน T.30

+FDR(Receive Phase C Data : การรับข้อมูลในเฟส C)

รูปแบบ : AT+FDR

คำสั่งนี้เป็นการเริ่มต้นหรือการรับข้อมูลเฟส C ต่อไป โดยโมเด็มจะแสดงค่าพารามิเตอร์ที่ใช้ด้วย ID และ NSS เฟรมถ้าเป็นไปได้

+FDT(Transmit Phase C Data : การส่งข้อมูลในเฟส C)

รูปแบบ : AT+FDT [=] n

n = พารามิเตอร์ย่อย DF,VR,WD,LN

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

n	Class 1 Definition
	Call Placement and Termination
0	Normal and proper end of connection
1	Ring detect without successful handshake
2	Call aborted from +FK or <CAN>
3	No loop current
	Transmit Phase A and Miscellaneous Errors
10	Unspecified phase A error
11	No answer (T.30 T1 timeout)
	Transmit Phase B Hangup Codes
20	Unspecified phase B error
21	Remote cannot receive or send
22	Command received error in transmit phase B
23	Command received invalid command received
24	Response received error
25	DCS sent three times without response
26	DIS/DTC received three times; DCS not recognized
27	Failure to train
28	Response received invalid response
	Transmit Phase C Hangup Codes
40	Unspecified phase C error
43	DTE to DCE data underflow

ตารางที่ ผ.3 แสดงรหัสของการวางสาย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ในเฟส B คำสั่งนี้จะบอกให้โมเด็มทำการติดต่อกับเครื่องโทรสารที่อยู่ห่างออกไปและส่ง DCS ไปด้วย ส่วนในเฟส C จะเป็นการเริ่มต้นการส่งข้อมูลหรือให้ส่งข้อมูลต่อไป

ค่าของพารามิเตอร์ย่อยแสดงไว้ในตารางที่ ผ.1

+FET(End the Page or Document : สิ้นสุดหน้าหรือข้อมูล)

รูปแบบ : AT+FET = n

n = 0-15

เป็นคำสั่งแสดงการสิ้นสุดหน้าหรือข้อมูลที่ทำการส่ง ถ้าอยู่ในโหมดไม่มีการควบคุมความผิดพลาด (non-error-control operation) โมเด็มจะส่ง RTC ไปให้กับเครื่องโทรสารที่อยู่ไกล และเข้าสู่เฟส D โดยการส่งรหัส post-page ตามข้อกำหนดใน มาตรฐาน T.30

+FET(Post-Page Message Response : ผลตอบสนองข้อความแสดง post-message)

รูปแบบ : +FET : n

n แสดงดังตารางต่อไปนี้

PPM Code	T.30 Mnemonic	Description
0	[PPS-]MPS	Another page next, same document
1	[PPS-]EOM	Another document next
2	[PPS-]EOP	No more pages or documents
3	PPS-NULL	Another partial page next
4	[PPS-]PRI-MPS	Another page, procedure interrupt
5	[PPS-]PRI-EOM	Another document, procedure interrupt
6	[PPS-]PRI-EOP	All done, procedure interrupt

ตารางที่ ผ.2 แสดงค่าของ n

ผลตอบสนองนี้สร้างโดยโมเด็มท้องถิ่น โดยสร้างตามข่าวสาร post-message ที่รับมาจากเครื่องโทรสารที่อยู่ห่างออกไป และคำสั่งนี้จะทำหลังจากที่ทำคำสั่ง +FDR เรียบร้อยแล้ว

+FHNG(Call Termination Status Response : ผลตอบสนองสถานะการเรียกติดต่อ)

รูปแบบ : +FHNG : n

n คือรหัสในการวางสาย แสดงดังตารางที่ ผ.3

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

n	Class 1 Definition
	Transmit Phase D Hangup Codes
50	Unspecified phase D error
51	Response received error
52	No response to MPS repeated three times
53	Invalid response to MPS
54	No response to EOP repeated three times
55	Invalid response to EOP
56	No response to EOM repeated three times
57	Invalid response to EOM
58	Unable to continue after PIN or PIP
	Receive Phase B Hangup Codes
70	Unspecified receive phase B error
71	Response received error
72	Command received error
73	T.30 T2 timeout, expected page not received
74	T.30 T1 timeout after EOM received
	Receive Phase C Hangup Codes
90	Unspecified receive phase C error
91	Missing EOL after 5 seconds
92	Unused code
93	DCE to DTE buffer overflow
94	Bad CRC or frame (ECM or BFT modes)
	Receive Phase D Hangup Codes
00	Unspecified receive phase D error
01	Response received invalid response received
02	Command received invalid response received
03	Unable to continue after PIN or PIP

ตารางที่ ผ.3 (ต่อ)แสดงรหัสของการวางสาย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โดยผลตอบสนองนี้โมเด็มส่งให้กับ DTE เพื่อแสดงถึงสาเหตุในการวางสายและจะเก็บไว้ในพารามิเตอร์ +FAXERR ด้วย

+FLID(Set Local ID String : กำหนดค่า local ID string)

รูปแบบ : AT+FLID = *"s"*

s คือรหัสแอสกี้ 20 ตัว

เป็นการกำหนดค่า local ID ที่จะใช้ใน TSI/CSI เฟรม

+FPTS(Set Page Transfer Status : กำหนดสถานะการส่งข้อมูล)

รูปแบบ : AT+FPTS = *ppr*

ppr มีค่า 1-5 ,ค่าเริ่มต้น : 1

คำสั่งนี้จะกำหนดค่า post-message ซึ่งจะต้องดูจาก(copy-checking) หรือสัญญาณแสดงคุณภาพ

Value	Mnemonic	Definition
1	MCF	Page good
2	RTN	Page bad. Retrain requested.
3	RTP	Page good. Retrain requested.
4	PIN	Page bad. Interrupt requested.
5	PIP	Page good. Interrupt requested.

ตารางที่ ผ.4 แสดงค่าของ *ppr*

+FPTS(Receive Page Transfer Status Response : การรับผลตอบสนองสถานะของการส่งข้อมูล)

รูปแบบ : +FPTS : *ppr,lc*[,*blc,cblc*[,*lbc*]]

ppr : ค่าของข่าวสาร post-page ซึ่งกำหนดเพื่อให้ +FPTS ใช้ในการกำหนดคำสั่งแสดงสถานะการส่งข้อมูล

lc : ตำแหน่งของบรรทัด

blc : ตำแหน่งของบรรทัดที่เสีย

cblc : เรียงลำดับบรรทัดที่เสีย

lbc : ตำแหน่งของไบท์ที่เสียหาย

ผลตอบสนองนี้จะถูกส่งจากแฟกซ์โมเด็มไปยัง DCE เมื่อสิ้นสุดเฟส C เพื่อรายงานสถานะการส่งข้อมูล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

+FPTS(Transmit Page Transfer Status Status Response : การส่งผลตอบสนองสถานะการส่งข้อมูล)

รูปแบบ : +FPTS : *ppr*

ppr : ค่าของข่าวสาร post-page ซึ่งกำหนดเพื่อให้ +FPTS ใช้ในการกำหนดคำสั่งแสดงสถานะการส่งข้อมูล

โมเด็มจะส่งผลตอบสนองนี้ไปยัง DTE เพื่อแสดงถึงคุณภาพและข่าวสาร post-page จากเครื่องโทรสารที่อยู่ห่างออกไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รายละเอียดโปรแกรมต่างๆ

โปรแกรม SENDFAX.C

```
#include <stdio.h>
#include <conio.h>
#include <string.h>
#include <bios.h>
#include <dos.h>
#include <alloc.h>
#include <stdlib.h>
#include <time.h>
#include <process.h>
#include <dir.h>
#include <stdarg.h>
#include <errno.h>
#include "d:\abcfax\data\project\keycode.h"
#include "d:\abcfax\data\project\serial.h"
#include "d:\abcfax\data\project\console.c"
#include "d:\abcfax\data\project\config.c"
#define Num_file 60
struct files {
               char name[13];
               char attr;
             }Files[Num_file];
int Retry_Dial(char *str);
void send_fax(void);
void send_page(FILE *send_file);
int  CheckFile(char *InputFile,int *check,int param);
int openfile(char *fileName);
void delmarker(int cur_x,int cur_y,int cur_file);
void putmarker(int cur_x,int cur_y,int cur_file);
void showscreen(int start,int end,int position);
int readfile(int start,char a[30]);
int readkey(char *a);
void findbksp(char *a);
void operate(void);

int last,first,total,index=0,E_Mail_Use=0;
char Fax_No[50],FileName[50],ID_String[50],path[50],*Send_buf;
long fsize;
FILE *FaxFile;
/*=============================================*/
/*argument format : (1) (filename) (dial No.)
  for use to link with e-mail project*/
void main(int argc,char *argv[])
{
 StrHead Head[4]={{13,9,"File Name :    "},{13,11,"Fax No.    :    "},
13,13,"ID String :    "},{13,15,"File Type :   ASCII    HUF    BFX"}};
   int ix,Is_huffman;
   char answer[3][50],ch;
   if(argc > 1 && argc <= 4)
   {
    if(strchr(argv[1],'1') != NULL)
      E_Mail_Use = 1;
   }
   initscreen();
   Savescr=(char *)farmalloc(4000);
   Send_buf=(char *)malloc(SendBufSize);
      if(Send_buf == NULL || Savescr == NULL)
        {
         QuitSerialComm();
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        printf("\nError in allocate memory\n");
        exit(0);
      }
   ReadConfig();
   MakeBorder(1,1,80,25,LIGHTGRAY,BLACK,2,'°');
   MakeBorder(10,7,70,19,WHITE,BLUE,1,' ');
   attr = WHITE+(BLUE <<4);
   if(!E_Mail_Use)
   {
    putline(30,7,"  Send Fax Main Menu  ");
    putline(1,24,"F1      ");
    putline(1,25,"Send    ");
    putline(12,24,"F2      ");
    putline(12,25,"Config  ");
    putline(23,24,"F3      ");
    putline(23,25,"FILE    ");
    putline(34,24,"F4      ");
    putline(34,25,"Exit    ");
    for(ix=0;ix <4;ix++)
    {
      putline(Head[ix].x,Head[ix].y,Head[ix].str);
      attr=BLUE +(LIGHTGRAY<<4);
        if(ix != 3)
        {
         memset(&answer[ix],0,50);
         putline(Head[ix].x+14,Head[ix].y,"
");
         attr=WHITE+(BLUE <<4);
        }
    }
    ix=0;
    while(1)
       {
        attr= BLUE +(LIGHTGRAY<<4);
        if(getstr(Head[ix].x+14,Head[ix].y,40,&answer[ix][0],&ch))
         {
            if(answer[0][0] != 0 && ix == 0)
             {
             if(CheckFile(&answer[0][1],&Is_huffman,1))
               {
                  old_attr = attr;
                  attr = WHITE+(BLUE <<4);
                  putline(27,15,"ASCII    HUF    BFX");
                  attr = old_attr;
               switch(ch){
                   case UP   : ix = 1; break;
                   case DOWN : ix = 2; break;
                  }
               }
             }
            switch(ch){
                 case UP   :  if(ix == 0)
                              {
                              ix = 2;
                              break;
                              }
                             ix--; break;
                 case DOWN :  if(ix == 2)
                              {
                              ix = 0;
                              break;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                                }
                                ix++; break;
                        case F1   :  Save_scr();
                                strcpy(FileName,&answer[0][1]);
                                strcpy(Fax_No,&answer[1][1]);
                                strcpy(ID_String,&answer[2][1]);
                              if(CheckFile(FileName,&Is_huffman,0) == 0)
                                {
                                 send_fax();
                                 QuitSerialComm();
                                }
                                Restore_scr();
                                break;
                          case F2  :Save_scr();
                                old_attr = attr;
                                SetConfig();
                                SaveConfig();
                                Restore_scr();
                                      attr = old_attr; break;
                          case F3  :Save_scr();
                                operate();
                                Restore_scr();
                                answer[0][0]=(char)strlen(FileName);
                                putline(Head[0].x+14,Head[0].y,"
");
                                putline(Head[0].x+14,Head[0].y,FileName;
                                strcpy(&answer[ix][1],FileName);
                                if(answer[0][0] != 0)
                                  CheckFile(FileName,&Is_huffman,1);
                                break;
                          case F4  :QuitSerialComm();
                                free(Send_buf);
                                textattr(LIGHTGRAY);
                                free(Savescr);
                                clrscr();
                                exit(0); break;
                        } /*end of switch*/
                    }else{  /* after press enter */
                      ix++;
                      if(ix == 1)
                       if(answer[0][0] != 0)
                         {
                         if(CheckFile(&answer[0][1],&Is_huffman,1))
                          ix--;
                         }else{
                              old_attr = attr;
                              attr = WHITE+(BLUE <<4);
                              putline(27,15,"ASCII    HUF    BFX");
                              attr = old_attr;
                              }
                       if(ix == 3)
                        ix = 0;
                    }
                  } /*end of while*/
              }else{
                strcpy(FileName,argv[2]);
                strcpy(Fax_No,argv[3]);
                ID_String[0] = 0;
                if(CheckFile(FileName,&Is_huffman,1) == 0)
                 if(CheckFile(FileName,&Is_huffman,0) == 0)
                  {
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        send_fax();
        QuitSerialComm();
       }
   }
   textattr(LIGHTGRAY);
   free(Send_buf);
   free(Savescr);
   clrscr();
}
void send_fax(void)
{
    char temp[60];
      MakeBorder(1,1,80,25,LIGHTGRAY,BLACK,2,'°');
      MakeBorder(15,7,66,19,WHITE,BLUE,1,' ');
      fsize = filesize(FaxFile)-26;
      fseek(FaxFile,26L,SEEK_SET);
      attr = WHITE+(BLUE <<4);
      putline(16,7,"<<Send Fax>>");
      attr= attr+BLINK;
      putline(16,9,"<<Initial Communication port>>");
      InitSerialComm(PORT,B19200,D8 | S1 | PNONE);
      delay(2000);
      putline(16,9,"<<Initial Fax Modem>>          ");
      send_command("ATZ\r");
      Wait_For(OK,2,1);
      delay(800);
      send_command("ATE1V1\r");
      Wait_For(OK,2,1);
      send_command("ATM1X4&D2S7=120S8=2\r");
      Wait_For(OK,4,1);
      sprintf(temp,"AT+FLID=\"%s\"\r",ID_String);
      send_command(temp);
      Wait_For(OK,2,1);
      send_command("AT+FCR=1;+FCLASS=2\r");
      Wait_For(OK,2,1);
      sprintf(temp,"AT+FDIS=%d,%d,0,2,0,0,0,5\r",RESOLUTION,TRANSFER_
RATE);
      send_command(temp);
      Wait_For(OK,4,1);
      if(Error == FAIL)
      {
        old_attr = attr;
        attr = YELLOW+(BLUE<<4)+BLINK;
        putline(16,17,"No Response from Modem");
        attr = WHITE+(BLUE<<4);
        putline(16,18,"Press any key to continue");
        getkey();
        attr=old_attr;
       return;
      }
      sprintf(temp,"ATDT%s\r",Fax_No);
      putline(16,9,"<<Dialing>>                    ");
      send_command(temp);
      Wait_For("\r\nCED",25,1);
      if(Error == FAIL)
       if(!Retry_Dial(temp))
            return;
      attr = WHITE+(BLUE <<4) + BLINK;
      Wait_For(OK,40,0);
      putline(16,9,"<<Sending Fax>>                ");
      attr = attr-BLINK;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
      putline(16,11,"  0 % Complete");
      putline(16,12,"±±±±±±±±±±±±±±±±±±±±±±±±±±±±±±±±±±±±±±±±±±±±±±±±±
±±");
      putline(16,14,"Press ESC to stop sending fax message");
      attr = YELLOW+(BLUE <<4);
      delay(20);
      send_command("AT+FDT\r");
      Wait_For("CONNECT",30,0);
      send_page(FaxFile);
}
int Retry_Dial(char *str)
{
   char temp[4][20]={"NO CARRIER","NO DIALTONE","BUSY","NO ANSWER"};
   int i;
   clock_t start;
   double t;
   for(i=0;i < 4;i++)
     if(strstr(Response,temp[i])!=NULL)
      {
        putline(16,10,"%s",temp[i]);
        putline(16,11,"It will Redial again in %d seconds",RETRY);
        break;
      }
   QuitSerialComm();
   InitSerialComm(PORT,B19200,D8 | S1 | PNONE);
   attr = WHITE+(BLUE <<4);
   start=clock();
      while(1)
       {
        t = (clock()-start)/CLK_TCK;
        if(t > RETRY)
         break;
        putline(16,12,"time remain : %2.0f",RETRY-t);
       }
      putline(16,12,"                          ");
      send_command(str);
      Wait_For("\r\nCED",25,1);
      if(Error == FAIL)
       {
         for(i=0;i < 4;i++)
            if(strstr(Response,temp[i])!=NULL)
             break;
         putline(16,10,"                                        ");
         putline(16,11,"                                        ");
         if(i != 4)
           putline(16,12,"%s",temp[i]);
         getkey();
         return 0;
       }
      return 1;
}
void send_page(FILE *send_file)
{
 char ch,*temp,check;
 int fhng,fpts,remain,i,ix,iy;
 long fcount=0;
 /*Wait until /CTS = 0*/
  while(1){
     {
       check=inportb(CommAddr + MSR);
       if(check & 0x10)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        break;
       }
 }
 while(( remain=fread(Send_buf,1,SendBufSize,send_file)) != 0 )
  {
   for(i = 0 ;i < remain; i++)
    {
      if(kbhit())
       {
        ch=getch();
        if(ch == 0x1B)
          {
           attr=WHITE+(BLUE<<4);
          putline(16,14,"User break (Press any key to continue)");
           getkey();
           return;
          }
       }
      check=inportb(CommAddr +MSR);
      if(!(check & 0x10))
      {
       while(1)
        {
         check=inportb(CommAddr + MSR);
         if(check & 0x10)
           break;
        }
      }
      tx(Send_buf[i]);
      if(Send_buf[i] == DLE)
       {
         check=inportb(CommAddr +MSR);
         if(!(check & 0x10))
          {
           while(1)
           {
             check=inportb(CommAddr + MSR);
             if(check & 0x10)
             break;
           }
          }
          tx(DLE);
       }
    } /*end loop for*/
     fcount+=remain;
     iy = fcount*100/fsize;
     putline(16,11,"%3d",iy);
     for(ix = 0; ix < iy/2;ix++)
       put(16+ix,12,'Û');
  } /*End loop while*/
  tx(DLE);
  tx(0x2E); /* end of page*/
  Wait_For(OK,2,1);
  attr=WHITE+(BLUE <<4);
  putline(16,14,"Send fax message OK                    ");
  send_command("AT+FET=2\r");
  Wait_For(OK,10,1);
  temp=strstr(Response,"+FPTS");
  if(temp == NULL)
   fpts = -1;
  else
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
     sscanf(temp+7,"%d",&fpts);
    temp=strstr(Response,"+FHNG");
    if(temp == NULL)
     fhng = -1;
    else
     sscanf(temp+7,"%d",&fhng);
    send_command("+++");
    Wait_For(OK,3,1);
    send_command("ATZ\r");
    Wait_For(OK,3,1);
       /*Check +FHNG & +FPTS*/
    putline(16,15,"Receive Page Transfer Response");
    switch(fpts){
        case -1: putline(16,16,"+FPTS Response Not found"); break;
        case 1 : putline(16,16,"page good"); break;
        case 2 : putline(16,16,"page bad,retrain requested"); break;
        case 3 : putline(16,16,"page good,retrain requested"); break;
        case 4 : putline(16,16,"page bad,interrupt requested"); break;
        case 5 : putline(16,16,"page good,interrupt requested");
break;
       }
    switch(fhng){
        case -1: putline(16,17,"+FHNG Response Not found"); break;
        case 0 : putline(16,17,"Normal Call Termination"); break;
        default: putline(16,17,"Error Call Termination");
       }
    putline(16,18,"Press any key to continue");
    getkey();
    fclose(FaxFile);
}

int CheckFile(char *InputFile,int *check,int param) /*param =1 check
file*/
{                                        /*return 0 noerror ,1 error*/
  char cmd[50],output[40],*ptr;
  unsigned length,ix,temp;
  if((FaxFile=fopen(InputFile,"rb")) == NULL)
   {
      old_attr = attr;
      attr = YELLOW+(BLUE<<4)+BLINK;
      putline(27,16,"Error in opening File");
      attr = WHITE+(BLUE<<4);
      putline(27,17,"Press any key to continue");
      getkey();
      putline(27,16,"                        ");
            putline(27,17,"                              ");
      attr=old_attr;
      return 1;
   }
  if(param)
  {
    fread(output,1,20,FaxFile);
    old_attr=attr;
    attr=YELLOW+(BLUE <<4);
    if((strstr(output,"HUFFMAN_FILE") != NULL))
     {
      *check=1;
       if(!E_Mail_Use)
        putline(36,15,"HUF");
     }else if((strstr(output,"BIT  FAX") != NULL))
     {
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
      *check=1;
      if(!E_Mail_Use)
        putline(43,15,"BFX");
    }else
    {
      *check=0;
      if(!E_Mail_Use)
       putline(27,15,"ASCII");
    }
    fclose(FaxFile);
    attr=old_attr;
    return 0;
  }else if(*check == 0)
   {
     fclose(FaxFile);
     length=strlen(InputFile);
     if(length >= 14)
     ix=length-12;
     else
     ix = 0;
     if((ptr=strchr(InputFile+ix,':')) != NULL)
     ix = ptr-InputFile;
     temp = ix;
     while(1)
      {
          ptr=strchr(InputFile+ix,'\\');
          if(ptr == NULL)
            break;
          else
           {
            ix = ptr-InputFile;
            if(ix == temp)
              ix++;
            else
              temp = ix;
           }
      }
    if((ptr=strchr(InputFile+ix,'.')) != NULL)
     {
      temp = ptr-InputFile;
       for(length = 0;length < temp-ix;length++)
        cmd[length] = InputFile[ix+length];
       cmd[length] =0;
       sprintf(output,"%s.huf",cmd);
     }
    else
      sprintf(output,"%s.huf",InputFile+ix);
    old_attr = attr;
    attr = YELLOW+(BLUE<<4)+BLINK;
    putline(27,16,"Waiting For Convert File");
    if(spawnl(P_WAIT,"coder.exe","coder.exe",InputFile,output,NULL) !
= 0)
     {
       putline(27,16,"                          ");
       switch(errno){
            case ENOENT : putline(27,16,"CODER.EXE not found");
                          break;
            case EINVAL : putline(27,16,"Invalid argument");  break;
            case ENOMEM : putline(27,16,"Out of memory");  break;
            case E2BIG  : putline(27,16,"Arg list too long"); break;
            case ENOEXEC: putline(27,16,"Exec format error"); break;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            default : putline(27,16,"Error  occuring\n");
          }
      attr = WHITE+(BLUE<<4);
      putline(27,17,"Press any key to continue");
      getkey();
      putline(27,16,"                            ");
      putline(27,17,"                         ");
      attr=old_attr;
      return 1;
    }
   if((FaxFile=fopen(output,"rb")) == NULL)
     {
     putline(27,16,"Error in openning File");
           attr = WHITE+(BLUE<<4);
     putline(27,17,"Press any key to continue");
           getkey();
           putline(27,16,"                          ");
           putline(27,17,"                        ");
     attr=old_attr;
     return 1;
       }
  }
  return 0;
}

void operate(void)
{ int y=0;
   MakeBorder(1,1,80,25,LIGHTGRAY,BLACK,2,'°');
   MakeBorder(10,1,56,24,WHITE,BLUE,1,' ');
   textattr(WHITE +(BLUE<<4));
   gotoxy(10,3);
   cprintf("Ìíííííííííííííííííííííííííííííííííííííííííííííí¹");
   gotoxy(23,24);
   cprintf("<<press esc to exit>>");
   for(;y==0;)
      {   textattr(WHITE+(BLUE<<4));
          gotoxy(11,2);
            cprintf("                                        ");
            y=openfile(FileName);
            _setcursortype(_NORMALCURSOR);
      }
}
int openfile(char *filename)
{
   int cur_x=1,cur_y=1,cur_file=0;
   int key,e,cur=0;
   cur_x=1;
   cur_y=1;
   cur_file=0;
   last=0;first=0;total=0;
   gotoxy(11,2);
   cprintf(">");
   gotoxy(13,2);
   if(readkey(path)==1)
     return(2);
      _setcursortype(_NOCURSOR);
   /*************LIST FILE FORM DIRECTORY***************/
   if(readfile(0,path)==0)
      return(0);
   showscreen(0,last,0);
   /********************READ KEYBOARD******************/
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
   for(e=1;e==1;)
        {  key=bioskey(0);
           switch(key)
             {  case 7181 :if(Files[cur_file].attr==0x10)
             /* enter.*/    { findbksp(path);
                             If(strcmp(Files[cur_file].name,"..\\")
==0)
                              { findbksp(path);
                                strcat(path,"\\");
                                index=strlen(path);
                                return(0);
                              }
                             if(strcmp(Files[cur_file].name,".\\")
==0)
                              {  strcat(path,"\\");
                                 index=strlen(path);
                                 return(0);
                              }
                             strcat(path,"\\");
                             strcat(path,Files[cur_file].name);
                             index=strlen(path);
                             return(0);
                           }
                          else
                           {  findbksp(path);
                             strcpy(filename,path);
                             strcat(filename,"\\");
                             strcat(filename,Files[cur_file].name);
                             e=2;       /* Enter */
                             index=strlen(path);
                              break;
                           }
  /* Esc */ case 283    : return(0);
  /* up  */ case 18432  : if (cur<3)
                           break;
                          delmarker(cur_x,cur_y,cur_file);
                          cur_file=cur_file-3;
                          cur=cur-3;
                          cur_y--;
                          if(cur_y<1)
                           {  cur_y=1;
                             first=first-3;
                             readfile(first,path);
                             cur_file=cur_file+3;
                             showscreen(0,last,cur_file);
                             break;
                           }
                          putmarker(cur_x,cur_y,cur_file);
                          gotoxy(20,25);
                          break;
  /* down */ case 20480 : if (cur+3>total)
                          break;
                         delmarker(cur_x,cur_y,cur_file);
                         cur_y++;
                         cur_file=cur_file+3;
                         cur=cur+3;
                         if(cur_y>20)
                          {  cur_y=20;
                             first=first+3;
                             readfile(first,path);
                             cur_file=cur_file-3;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                                showscreen(0,last,cur_file);
                               }
                              putmarker(cur_x,cur_y,cur_file);
                              gotoxy(20,25);
                              break;
  /* right */ case 19712 : if ((cur_x==3)||(cur>=total))
                                break;
                              delmarker(cur_x,cur_y,cur_file);
                              cur_x++;
                              cur_file++;
                              cur++;
                              putmarker(cur_x,cur_y,cur_file);
                              gotoxy(20,25);
                              break;
  /* left */  case 19200 : if (cur_x==1)
                                break;
                              delmarker(cur_x,cur_y,cur_file);
                              cur_x--;
                              cur--;
                              cur_file--;
                              putmarker(cur_x,cur_y,cur_file);
                              gotoxy(20,25);
                              break;
/* page down */ case 20736 : if ((cur-cur_file+Num_file)>total)
                              break;
                              first=first+60;
                              readfile(first,path);
                              cur=cur+60;
                             if(cur>total)
                              { cur_file=0;
                                cur=total-last;
                                cur_y=1;
                                cur_x=1;
                              }
                             readfile(first,path);
                             showscreen(0,last,cur_file);
                             break;
/* page up */ case 18688 :if (cur<3)
                             break;
                             first=first-60;
                             cur=cur-60;
                             if(first<0)
                              { first=0;
                                cur_file=0;
                                cur_x=1;
                                cur_y=1;
                                cur=0;
                               }
                             readfile(first,path);
                             showscreen(0,last,cur_file);
                             break;
            }  /* end switch case */
        } /* end if(kbhit),for(e==0) */

  return(1);
}
void showscreen(int start,int end,int position)
{ int a,b,x=11,y=4;
    b=end-start;
    for(a=0;a<60;a++)
    { textattr(WHITE+(BLUE<<4));
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        gotoxy(x,y);
        cprintf("              ");
        if(a<=b)
          {  if( start==position)
             textattr(BLUE+(LIGHTGRAY<<4));
             gotoxy(x,y);
             cprintf("%s",Files[start++].name);
          }
        x=x+15;
        if( x>51 )
              {  x=11;
                 y++;
              }
      }
}
int readfile(int start,char a[])
{  struct ffblk ffblk;
   int done,b,file_index=0;
   done = findfirst(a,&ffblk,FA_DIREC);
   if (done == -1)
       return(0);
   total=0;
   while (!done)
     {  b=0;
      if ((total>=start)&(file_index<60))
      {
       while ((Files[file_index].name[b]=ffblk.ff_name[b]) != '\0')
          b++ ;
       Files[file_index].attr=ffblk.ff_attrib;
       if(Files[file_index].attr==0x10)
          {  Files[file_index].name[b++]='\\';
             Files[file_index].name[b]='\0';
          }
       last=file_index++;
      }
      total++;
      done = findnext(&ffblk);
     }
  total=total-1;
  return(1);
}
void putmarker(int cur_x,int cur_y,int cur_file)
{
   gotoxy(11+(cur_x-1)*15,4+cur_y-1);
   textattr(BLUE+(LIGHTGRAY<<4));
   cprintf("%s",Files[cur_file].name);
}
void delmarker(int cur_x,int cur_y,int cur_file)
{  textattr(WHITE+(BLUE<<4));
   gotoxy(11+(cur_x-1)*15,4+cur_y-1);
   cprintf("                ");
   gotoxy(11+(cur_x-1)*15,4+cur_y-1);
   cprintf("%s",Files[cur_file].name);

}
int readkey(char *a)
 { int x,y;
   cprintf("%s",a);
   x=wherex();y=wherey();
   for (;a[index-1]!=13;)
      {
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
      if((a[index]=getch())!=0)
      {
       if(a[index]==27)
        {
           a[index]=NULL;
         return(1);
        }
       if((a[index]==8)&&(index>0))
         { gotoxy(x-1,y);
           cprintf(" ");
           gotoxy(--x,y);
           index--;
         }
       else
         { if ((a[index]!=8)&(index<40))
            { cprintf("%c",a[index] );
             x=wherex();
             index++;
            }
         }
      }else
        getch();
     }
    if(index<=0) index=1;
      index--;
      a[index]='\0';
      return(0);
}
void findbksp(char *a)
 {
     for(index=0;a[index]!=NULL;index++);
       for(;((a[index]!='\\'));index--)
         if(index==0)
       { strcpy(a, "X:\\");          /* fill string with form of
                                       response: X:\ */
         a[0] = 'A' + getdisk();  /* replace X with current drive
                                       letter */
         getcurdir(0, a+3);       /* fill rest of string with current
                                       directory */
         return;
       }
       a[index]=NULL;
      return;
  }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โปรแกรม RECVFAX.C

```
#include <stdio.h>
#include <conio.h>
#include <string.h>
#include <bios.h>
#include <dos.h>
#include <alloc.h>
#include <stdlib.h>
#include <dir.h>
#include <time.h>
#include <stdarg.h>
#include "d:\abcfax\data\project\keycode.h"
#include "d:\abcfax\data\project\serial.h"
#include "d:\abcfax\data\project\console.c"
#include "d:\abcfax\data\project\config.c"
typedef unsigned char byte;
void Receive_Fax(void);
byte inverse[256]={0x00,0x80,0x40,0xC0,0x20,0xA0,0x60,0xE0,0x10,
0x90,0x50,0xD0,0x30,0xB0,0x70,0xF0,0x08,0x88,0x48,0xC8,0x28,0xA8,
0x68,0xE8,0x18,0x98,0x58,0xD8,0x38,0xB8,0x78,0xF8,0x04,0x84,0x44,
0xC4,0x24,0xA4,0x64,0xE4,0x14,0x94,0x54,0xD4,0x34,0xB4,0x74,0xF4,
0x0C,0x8C,0x4C,0xCC,0x2C,0xAC,0x6C,0xEC,0x1C,0x9C,0x5C,0xDC,0x3C,
0xBC,0x7C,0xFC,0x02,0x82,0x42,0xC2,0x22,0xA2,0x62,0xE2,0x12,0x92,
0x52,0xD2,0x32,0xB2,0x72,0xF2,x0A,0x8A,0x4A,0xCA,0x2A,0xAA,0x6A,
0xEA,0x1A,0x9A,0x5A,0xDA,0x3A,0xBA,0x7A,0xFA,0x06,0x86,0x46,0xC6,
0x26,0xA6,0x66,0xE6,0x16,0x96,0x56,0xD6,0x36,0xB6,0x76,0xF6,0x0E,
0x8E,0x4E,0xCE,0x2E,0xAE,0x6E,0xEE,0x1E,0x9E,0x5E,0xDE,0x3E,0xBE,
0x7E,0xFE,0x01,0x81,0x41,0xC1,0x21,0xA1,0x61,0xE1,0x11,0x91,0x51,
0xD1,0x31,0xB1,0x71,0xF1,0x09,0x89,0x49,0xC9,0x29,0xA9,0x69,0xE9,
0x19,0x99,0x59,0xD9,0x39,0xB9,0x79,0xF9,0x05,0x85,0x45,0xC5,0x25,
0xA5,0x65,0xE5,0x15,0x95,0x55,0xD5,0x35,0xB5,0x75,0xF5,0x0D,0x8D,
0x4D,0xCD,0x2D,0xAD,0x6D,0xED,0x1D,0x9D,0x5D,0xDD,0x3D,0xBD,0x7D,
0xFD,0x03,0x83,0x43,0xC3,0x23,0xA3,0x63,0xE3,0x13,0x93,0x53,0xD3,
0x33,0xB3,0x73,0xF3,0x0B,0x8B,0x4B,0xCB,0x2B,0xAB,0x6B,0xEB,0x1B,
0x9B,0x5B,0xDB,0x3B,0xBB,0x7B,0xFB,0x07,0x87,0x47,0xC7,0x27,0xA7,
0x67,0xE7,0x17,0x97,0x57,0xD7,0x37,0xB7,0x77,0xF7,0x0F,0x8F,0x4F,
0xCF,0x2F,0xAF,0x6F,0xEF,0x1F,0x9F,0x5F,
0xDF,0x3F,0xBF,0x7F,0xFF};
char EOL_CODE[10] ={0,0x80,0,0x08,0x80,0,0x08,0x80,0,0x80};
const byte bits[8]={0x80,0x40,0x20,0x10,8,4,2,1};
char Fax_No[50],FileName[50],ID_String[50],path[50];
char writebuf[CommBufSize],readbuf[CommBufSize];
FILE *FaxFile;

/*=============================================*/

void main(void)
{
   int ch;
   initscreen();
   Savescr=(char *)farmalloc(4000);
   ReadConfig();
   MakeBorder(1,1,80,25,LIGHTGRAY,BLACK,2,'°');
   attr = WHITE+(BLUE <<4);
   putline(1,24,"F1     ");
   putline(1,25,"Receive");
   putline(12,24,"F2     ");
   putline(12,25,"Config ");
   putline(23,24,"F3     ");
   putline(23,25,"EXIT   ");
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    while(1){
       ch = getkey();
       if(ch & 0x100)
        switch(ch & 0xFF)
         {
           case F1 : Save_scr();
                     Receive_Fax();
                     QuitSerialComm();
                     Restore_scr();
                     break;
           case F2 : Save_scr();
                     SetConfig();
                              SaveConfig();
                     Restore_scr();
                     break;
           case F3 : textattr(LIGHTGRAY);
                     clrscr();
                     free(Savescr);
                     exit(0);
         }
    }
}
void Receive_Fax(void)
{
       volatile int i,ch,iy,remain,ix;
       long file_pos ,size ;
       char temp[60],*fname="FAX0XXXXXX",*str=NULL;
       FILE *receive_file,*temp_file;
       file_pos = size = 0;
 // create file name
       str = mktemp(fname);
       sprintf(temp,"%s%s",_DIRECTORY,str);
 // set 19200 bps, 8 bit,No stop bit ,No parity bit
       MakeBorder(1,1,80,25,LIGHTGRAY,BLACK,2,'°');
       MakeBorder(10,7,70,19,WHITE,BLUE,1,' ');
       attr = WHITE+(BLUE<<4)+BLINK;
       putline(13,9,"<<Initial Communication port>>");
       InitSerialComm(PORT,B19200,D8 | S1 | PNONE);
       delay(500);
       attr -= BLINK;
       putline(13,9,"<<Initial Fax Modem>>          ");
       temp_file = tmpfile();
       receive_file=fopen(temp,"wb");
       if(receive_file == NULL || temp_file == NULL)
         {
             attr+=BLINK;
             putline(13,14,"Error in Opening file          ");
             attr-=BLINK;
             putline(13,15,"Press any key to exit    ");
             getkey();
             return;
         }
       send_command("ATZ\r");
       Wait_For(OK,2,1);
       delay(800);
       send_command("AT&C1\r");
       Wait_For(OK,2,1);
       send_command("ATV1M1X4&D2S0=1\r");
       Wait_For(OK,4,1);
       sprintf(temp,"AT+FDIS=%d,%d,0,2,0,0,0,5\r",RESOLUTION,TRANSFER_
RATE);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
send_command(temp);
Wait_For(OK,4,1);
send_command("AT+FCLASS=2\r");
Wait_For(OK,4,1);
send_command("AT+FAA=0;+FCR=1\r");
Wait_For(OK,4,1);
if(Error == FAIL)
{
  old_attr = attr;
  attr = YELLOW+(BLUE<<4)+BLINK;
  putline(13,14,"No Response from Modem          ");
  attr = WHITE+(BLUE<<4);
  putline(13,15,"Press any key to continue");
  getkey();
  attr=old_attr;
 return;
}
putline(13,9,"<<Receiving Fax mode>>          ");
putline(13,14,"Press ESC to exit receiving mode");
 while(1)
  {
   if(kbhit())
     {
       ch = getkey();
       if(ch == ESC)
       {
        fclose(receive_file);
        fclose(temp_file);
        return;
       }
     }
   Wait_For(RING,1,1);
   if(Error == PASS)
      break;
    Error = PASS;
  }
  putline(13,10,"Fax is coming !");
  Wait_For("+FCON",30,0);
  putline(13,10,"Fax Negotiation");
  Wait_For(OK,30,0);
  delay(100);
  send_command("AT+FDR\r");
  Wait_For(CONNECT,30,0);
  putline(13,10,"Fax connection ");
  tx(0x12);
  Curr_Pos = CommDataPos;
  ix = 0;
  // neglect some byte data for 00h ,FFh
 while(ix < 100){
      if(Curr_Pos != CommDataPos)
       {
        Curr_Pos++;
        ix++;
        if(Curr_Pos == CommBufSize)
            Curr_Pos = 0;
       }
   }
 //Wait until receive 00h to begin receive real data
 while(1){
      if(Curr_Pos != CommDataPos)
       {
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            if(CommBuf[Curr_Pos] == 0)
                break;
            Curr_Pos++;
            if(Curr_Pos == CommBufSize)
                Curr_Pos = 0;
           }
        }
       // begin to receive real data from remote fax station
      i = 0;
      while(1){
       if(Curr_Pos != CommDataPos)
        {
         writebuf[i] = CommBuf[Curr_Pos++];
         switch(writebuf[i])
            {
             case DLE : iy=1; break;
             case ETX : iy++; break;
             default  : iy = 0;
            }
         i++;
         size++;
         if(Curr_Pos == CommBufSize)
             Curr_Pos = 0;
        }
       if(kbhit())
        {
            ch=getkey();
            if (ch==ESC)
             {
              fclose(receive_file);
              fclose(temp_file);
              return;
             }
        }
       if(i == CommBufSize)
         {
             i = 0;
             fwrite(writebuf,CommBufSize,1,temp_file);
         }
        if(iy==2)           //end of receive data
            break;
        }
        if(i < CommBufSize)
         fwrite(writebuf,1,i,temp_file);
        // end loop while
      //receive data until DLE(10h) ETX(03h) it will be end of data
from fax
        Wait_For(OK,20,0);
        delay(50);
        send_command("AT+FDR\r");
        Wait_For(OK,20,0);
        QuitSerialComm();
        /*
         first byte fill with ZERO value byte and then swap bits
following byte
         util found that Buf_Pos from receive file less than 100
bytes.
         After that find EOL codes for 2 times appearance then it
will be end of
         data file and fill with 4 EOL codes and End of convert
receive data file
```

เอกสารนี้ใช้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
          to Huffman files.
         */
         memset(writebuf,0,CommBufSize);
         fwrite(writebuf,1,30,receive_file);
         i=0;
         ix = 0;
         fseek(temp_file,0L,SEEK_SET);
         while(((remain=fread(readbuf,1,CommBufSize,temp_file))!=0))
         {
          for(iy=0;iy < remain ;iy++)
          {
               switch(readbuf[iy])
                {
                   case 0x80 :case 0x40 :case 0x20 :case 0x10:
                   case 0x08 :case 0x04 :case 0x02 :case 0x01: ix++;
break;
                   case 0   : ix = 1; break;
                   default : ix = 0;
                }
               file_pos++;
               if(ix == 2)
               {
                if((size - file_pos) < 100)
                   break;
            ix = 0;
           }
           writebuf[i++] = inverse[readbuf[iy]];
           if(i == CommBufSize)
           {
             i = 0;
             fwrite(writebuf,1,CommBufSize,receive_file);
           }
          }
          if(ix == 2)
            break;
         }
         if( i < CommBufSize)
          fwrite(writebuf,1,i,receive_file);
         fwrite(EOL_CODE,1,10,receive_file);
         fclose(temp_file);
         fseek(receive_file,0L,SEEK_SET);
         fprintf(receive_file,"HUFFMAN_FILE");
         fclose(receive_file);
         putline(13,14,"Fax receive data ready !        ");
         putline(13,15,"Press Any Key to continue        ");
         ch=getkey();
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
int compare[16]={0x8000,0x4000,0x2000,0x1000,0x800,0x400,
0x200,0x100,0x80,0x40,0x20,0x10,8,4,2,1};
byte bits[8]={0x80,0x40,0x20,0x10,8,4,2,1};
byte inverse[256]={0x00,0x80,0x40,0xC0,0x20,0xA0,0x60,0xE0,0x10,
0x90,0x50,0xD0,0x30,0xB0,0x70,0xF0,0x08,0x88,0x48,0xC8,0x28,0xA8,
0x68,0xE8,0x18,0x98,0x58,0xD8,0x38,0xB8,0x78,0xF8,0x04,0x84,0x44,
0xC4,0x24,0xA4,0x64,0xE4,0x14,0x94,0x54,0xD4,0x34,0xB4,0x74,0xF4,
0x0C,0x8C,0x4C,0xCC,0x2C,0xAC,0x6C,0xEC,0x1C,0x9C,0x5C,0xDC,0x3C,
0xBC,0x7C,0xFC,0x02,0x82,0x42,0xC2,0x22,0xA2,0x62,0xE2,0x12,0x92,
0x52,0xD2,0x32,0xB2,0x72,0xF2,x0A,0x8A,0x4A,0xCA,0x2A,0xAA,0x6A,
0xEA,0x1A,0x9A,0x5A,0xDA,0x3A,0xBA,0x7A,0xFA,0x06,0x86,0x46,0xC6,
0x26,0xA6,0x66,0xE6,0x16,0x96,0x56,0xD6,0x36,0xB6,0x76,0xF6,0x0E,
0x8E,0x4E,0xCE,0x2E,0xAE,0x6E,0xEE,0x1E,0x9E,0x5E,0xDE,0x3E,0xBE,
0x7E,0xFE,0x01,0x81,0x41,0xC1,0x21,0xA1,0x61,0xE1,0x11,0x91,0x51,
0xD1,0x31,0xB1,0x71,0xF1,0x09,0x89,0x49,0xC9,0x29,0xA9,0x69,0xE9,
0x19,0x99,0x59,0xD9,0x39,0xB9,0x79,0xF9,0x05,0x85,0x45,0xC5,0x25,
0xA5,0x65,0xE5,0x15,0x95,0x55,0xD5,0x35,0xB5,0x75,0xF5,0x0D,0x8D,
0x4D,0xCD,0x2D,0xAD,0x6D,0xED,0x1D,0x9D,0x5D,0xDD,0x3D,0xBD,0x7D,
0xFD,0x03,0x83,0x43,0xC3,0x23,0xA3,0x63,0xE3,0x13,0x93,0x53,0xD3,
0x33,0xB3,0x73,0xF3,0x0B,0x8B,0x4B,0xCB,0x2B,0xAB,0x6B,0xEB,0x1B,
0x9B,0x5B,0xDB,0x3B,0xBB,0x7B,0xFB,0x07,0x87,0x47,0xC7,0x27,0xA7,
0x67,0xE7,0x17,0x97,0x57,0xD7,0x37,0xB7,0x77,0xF7,0x0F,0x8F,0x4F,
0xCF,0x2F,0xAF,0x6F,0xEF,0x1F,0x9F,0x5F,0xDF,0x3F,0xBF,0x7F,0xFF};
byte font[256][20][2],buf[20][216],Flag,temp[200],*readbuf,*writebuf;
int WriteBuf_Pos,Buf_Pos,WriteBuf_bit,Buf_bit,remain,count,Code_bit,
    Buf_count;
int i,j,k,x,y,row,Ch_count,End=NO;
long FileSize,size=1;
FILE *InputFile,*OutputFile,*fontfile;
void main(int argc,char *argv[])
{
 if(argc < 3)
   {
    printf("Usage : CODER [input file] [output file]\n");
    printf("Convert Document files to Huffman format files\n%c",7);
    exit(0);
   }
 if((fontfile=fopen("normal2.fon","rb")) == NULL)
   {
    printf("Error in open fontfile\n");
    exit(0);
   }
  fread(&font,sizeof(font),1,fontfile);
  fclose(fontfile);
 if(((InputFile=fopen(argv[1],"rb"))==NULL) || ((OutputFile=fopen
(argv[2],"wb"))==NULL))
   {
    printf("Error in open file \n");
    exit(0);
   }
 readbuf=(byte *)malloc(BufSize);
 writebuf=(byte *)malloc(BufSize);
  if(readbuf == NULL || writebuf == NULL)
   {
    printf("Error in allocate memory\n");
    exit(0);
   }
   WriteBuf_Pos = WriteBuf_bit = Ch_count=0;
   FileSize=filesize(InputFile);
   memset(writebuf,0,BufSize);
   fwrite(writebuf,1,1024,OutputFile);
```

เอกสารนี้เป็นสิ่งที่ได้รับการคุ้มครองลิขสิทธิ์ ใช้เพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โปรแกรม CODER.C

```
#include <stdio.h>
#include <conio.h>
#include <string.h>
#include <stdlib.h>
#include <dos.h>
#define TabSize 8
#define CR  13
#define TAB 9
#define YES 1
#define NO  0
#define PEL 216
#define BLACK 1
#define WHITE 0
#define BufSize 4096
typedef unsigned char byte;
typedef struct{
               byte count;
               int code;
             }HUFF_CODE;
long filesize(FILE *send_file);
void Ascii_to_Pel(void);
void Onedimention(void); /*White 0 , Black 1*/
void To_Huffman(HUFF_CODE *AnyCode);
HUFF_CODE Terminate[64][2]={{8,0x3500,10,0X0DC0},{6,0x1C00,3,0X4000},
{4,0x7000,2,0XC000},{4,0x8000,2,0X8000},{4,0xB000,3,0X6000},
{4,0XC000,4,0X3000},{4,0XE000,4,0X2000},{4,0XF000,5,0X1800},
{5,0X9800,6,0X1400},{5,0XA000,6,0X1000},{5,0X3800,7,0X0800},
{5,0X4000,7,0X0A00},{6,0X2000,7,0X0E00},{6,0X0C00,8,0X0400},
{6,0XD000,8,0X0700},{6,0XD400,9,0X0C00},{6,0XA800,10,0X05C0},
{6,0XAC00,10,0X0600},{7,0X4E00,10,0X0200},{7,0X1800,11,0X0CE0},
{7,0X1000,11,0X0D00},{7,0X2E00,11,0X0D80},{7,0X0600,11,0X06E0},
{7,0X0800,11,0X0500},{7,0X5000,11,0X02E0},{7,0X5600,11,0X0300},
{7,0X2600,12,0X0CA0},{7,0X4800,12,0X0CB0},{7,0X3000,12,0X0CC0},
{8,0X0200,12,0X0CD0},{8,0X0300,12,0X0680},{8,0X1A00,12,0X0690},
{8,0X1B00,12,0X06A0},{8,0X1200,12,0X06B0},{8,0X1300,12,0X0D20},
{8,0X1400,12,0X0D30},{8,0X1500,12,0X0D40},{8,0X1600,12,0X0D50},
{8,0X1700,12,0X0D60},{8,0X2800,12,0X0D70},{8,0X2900,12,0X06C0},
{8,0X2A00,12,0X06D0},{8,0X2B00,12,0X0DA0},{8,0X2C00,12,0X0DB0},
{8,0X2D00,12,0X0540},{8,0X0400,12,0X0550},{8,0X0500,12,0X0560},
{8,0X0A00,12,0X0570},{8,0X0B00,12,0X0640},{8,0X5200,12,0X0650},
{8,0X5300,12,0x0520},{8,0X5400,12,0x0530},{8,0X5500,12,0x0240},
{8,0X2400,12,0x0370},{8,0X2500,12,0x0380},{8,0X5800,12,0x0270},
{8,0X5900,12,0x0280},{8,0X5A00,12,0x0580},{8,0X5B00,12,0x0590},
{8,0X4A00,12,0X02B0},{8,0X4B00,12,0X02C0},{8,0X3200,12,0X05A0},
{8,0X3300,12,0X0660},{8,0X3400,12,0X0670}},
MakeUp[27][2]={{5,0XD800,10,0X03C0},{5,0X9000,12,0X0C80},
{6,0X5C00,12,0X0C90},{7,0X6E00,12,0X05B0},{8,0X3600,12,0X0330},
{8,0X3700,12,0X0340},{8,0X6400,12,0X0350},{8,0X6500,13,0X0360},
{8,0X6800,13,0X0368},{8,0X6700,13,0X0250},{9,0X6600,13,0X0258},
{9,0X6680,13,0X0260},{9,0X6900,13,0X0268},{9,0X6980,13,0X0390},
{9,0X6A00,13,0X0398},{9,0X6A80,13,0X03A0},{9,0X6B00,13,0X03A8},
{9,0X6B80,13,0X03B0},{9,0X6C00,13,0X03B8},{9,0X6C80,13,0X0290},
{9,0X6D00,13,0X0298},{9,0X6D80,13,0X02A0},{9,0X4C00,13,0X02A8},
{9,0X4C80,13,0X02D0},{9,0X4D00,13,0X02D8},{6,0X6000,13,0X0320},
{9,0X4D80,13,0X0328}};
/*MakeUp code = 64,128,192,256,320,384,448,512,576,640,704,768,832,
896,960,1024,1088,1152,1216,1280,1344,1408,1472,1536,1600,1664,1728*/

HUFF_CODE EOL={12,0x0010},Zero={1,0};
```

เอกสารนี้มีลิขสิทธิ์ที่ใช้ได้เพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
while(((remain=fread(readbuf,1,BufSize,InputFile))!=0))
{
  for(i=0;i < remain ;i++,size++)
  {
     if(readbuf[i] >=32)
      {
       if(readbuf[i] != 141)
              temp[Ch_count++]=readbuf[i];
       else if(readbuf[i+1] == 0x0A)
        {       /* RW format  8Dh 0Ah,CW format 8Dh 0Dh 0Ah*/
            temp[Ch_count++] = CR;
            i++;
            size++;
        }
      }
     else
      {
     switch(readbuf[i]){
       case TAB  : for(x=0;x < TabSize;x++)
                   temp[Ch_count++] = 32;
                 break;
    /*for link with email project*/
       case 0x0A : temp[Ch_count++] = CR;
                 size++;
                 break;
    /*-----------------------------------------*/
       case CR   : temp[Ch_count++] = CR; /*new line*/
                 i++;
                 size++;
                 break;
       case 0x1A : temp[Ch_count++] = CR;
                 if(temp[Ch_count-2] == 0x9E) /*format cw.16*/
                   temp[Ch_count-2] = 32;
                 End = YES;
                 i = remain;
                 break;
           }
     }
    if(size == FileSize)
      {
       temp[Ch_count++] = CR;
       End =YES;
       }
      if(temp[Ch_count-1] == CR)
       {
         Ch_count=0;
         Ascii_to_Pel();
         Onedimention();
         /*if character/line >88 begin new line*/
         if((temp[Ch_count-1] != CR))
           {
            Ascii_to_Pel();
            Onedimention();
           }
         Ch_count=0;
       }
    }/*end loop for*/
   if(End)
     break;
  } /*end loop while*/
for(count=0;count <6;count++)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    To_Huffman(&EOL);
 if(WriteBuf_Pos < BufSize)    /*Clear all data pending in buffer*/
    fwrite(writebuf,1,WriteBuf_Pos,OutputFile);
 free(writebuf);
 free(readbuf);
 fseek(OutputFile,0L,SEEK_SET);
 fprintf(OutputFile,"HUFFMAN_FILE");
 fclose(InputFile);
 fclose(OutputFile);
}
void Onedimention(void)
{
  int PIXEL[1729],ix,Change;
  div_t l;
 for(row=0;row < 20;row++)
  {       /*insert EOL code*/
   Buf_count=0;
   Change = 0;
   To_Huffman(&EOL);
   Flag=WHITE;
   count=0;
   for(Buf_Pos=0;Buf_Pos < PEL;Buf_Pos++)
   {
    if(buf[row][Buf_Pos] == 0)
     {
       if(Flag == WHITE)
         count+=8;
       else
        {
          PIXEL[Change++] =count;
          Flag = WHITE;
          count = 8;
        }
     }
    else
      for(Buf_bit = 0;Buf_bit < 8;Buf_bit++)
          {
            if(buf[row][Buf_Pos] & bits[Buf_bit]) /*black pixel*/
              {
                if(Flag == WHITE)
                  {
                    PIXEL[Change++]=count;
                    Flag = BLACK;
                    count=0;
                  }
                   count++;
              }else{     /*white pixel*/
                 if(Flag == BLACK)
                  {
                    PIXEL[Change++]=count;
                 Flag = WHITE;
                 count=0;
                }
                  count++;
              }
          } /*end loop for Buf_bit*/
    }/*end loopfor Buf_Pos*/
    PIXEL[Change++] = count;
    Flag = WHITE;
    for(ix=0;ix < Change;ix++)
    {
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
      if(PIXEL[ix] <=63)
        To_Huffman(&Terminate[PIXEL[ix]][Flag]);
      else{
            count = PIXEL[ix];
            j=0;
            do{
                l=div(count,(j+1)*64);
                j++;
              }while((l.quot != 1) || (l.rem > 63));
            To_Huffman(&MakeUp[j-1][Flag]);
            count=l.rem;
            To_Huffman(&Terminate[count][Flag]);
           }
      if(Flag == BLACK)
        Flag = WHITE;
      else
        Flag = BLACK;
    }
   if(Buf_count < 28)
     {
      while(Buf_count != 28)
      To_Huffman(&Zero);
     }
  } /*end loop for row*/
}
void To_Huffman(HUFF_CODE *AnyCode)
{
  for(Code_bit=0;Code_bit < (*AnyCode).count;Code_bit++)
   {
     if((*AnyCode).code & Compare[Code_bit])
       writebuf[WriteBuf_Pos] |=bits[WriteBuf_bit];
     WriteBuf_bit++;
     if(WriteBuf_bit == 8)
      {
         /*Swap bit LSB->MSB*/
        writebuf[WriteBuf_Pos] = inverse[writebuf[WriteBuf_Pos]];
       WriteBuf_bit = 0;
       WriteBuf_Pos++;
       Buf_count++;
       if(WriteBuf_Pos == BufSize)
         {
           fwrite(writebuf,BufSize,1,OutputFile);
           WriteBuf_Pos = 0;
           memset(writebuf,0,BufSize);
         }
      }
   }
}
void Ascii_to_Pel(void)
{
 int k;
 for(row=0;row <20; row++)
  {
     k=Ch_count;
     for(j=0;j < 20;j++)
       buf[row][j]=0; /* insert front tab*/
     x=0;
     while(x !=88) /*ascii 88 characters*/
      {
         switch(temp[k]) {
           case 209 :case 212 :case 213 :case 214 :case 215 :
```

เอกสารนี้เป็นเอกสารที่สงวนลิขสิทธิ์ สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            case 216 :case 217 :case 218 :case 231 :case 232 :
            case 233 :case 234 :case 235 :case 236 :case 237 :
                      buf[row][j-2] |=font[temp[k]][row][0];
                      buf[row][j-1] |=font[temp[k]][row][1];
                      break;
            case CR : for(;x <88;x++)
                       for(y=0;y <2;y++)
                          buf[row][j++] = 0;
                      break;
           default  : for(y=0;y <2;y++)
                       buf[row][j++]=font[temp[k]][row][y];
                      x++;
                      break;
                  }
            k++;
        }
      for(j=196;j < 216;j++)
        buf[row][j]=0; /*insert back tab*/
  }
  Ch_count=k;
}
long filesize(FILE *send_file)
{
  long curpos, length;
     curpos = ftell(send_file);
     fseek(send_file, 0L, SEEK_END);
     length = ftell(send_file);
     fseek(send_file, curpos, SEEK_SET);
     return length;
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โปรแกรม CONVERT.C

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
char bits[8] = {0x80,0x40,0x20,0x10,8,4,2,1};
char patt[4] = {0xC0,0x30,0x0C,0x03};
void main(int argc, char *argv[])
{
   FILE *fontfile,*outfile;
   char font[256][20];
   char ch[256][20][2];
   int ix,iy,bit;
   if(argc < 3)
   {
     printf("Usage: CONVERT [fontfile] [faxfontfile]\n");
     printf("Convert cuwriter font to fax font%c\n",7);
     exit(0);
   }
   memset(ch,0,10240);
   if((fontfile = fopen(argv[1],"rb"))==NULL || (outfile = fopen(argv
[2],"wb"))== NULL)
    exit(0);
   fread(font,5120,1,fontfile);
   fclose(fontfile);
   for(ix = 0;ix < 256;ix++)
    for(iy = 0;iy < 20;iy++)
    {
      for(bit = 0; bit < 4;bit++)
       if(font[ix][iy] & bits[bit])
         ch[ix][iy][0] |=patt[bit];
      for(; bit < 8;bit++)
       if(font[ix][iy] & bits[bit])
         ch[ix][iy][1] |=patt[bit-4];
    }
    fwrite(ch,10420,1,outfile);
    fclose(outfile);
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โปรแกรม VIEW.C

```
#include "d:\abcfax\data\project\keycode.h"
#include <stdio.h>
#include <conio.h>
#include <alloc.h>
#include <dos.h>
#include <graphics.h>
#include <stdlib.h>
#include <mem.h>
#include <string.h>
#define NO  0
#define YES 1
#define PEL 216
#define BLACK 1
#define WHITE 0
#define BufSize 4320

typedef unsigned char byte;
typedef struct ptr
{
  byte bit;
  int count;
  struct ptr *Left,*Right;
}NODE;
typedef struct
{
   byte bit;
   int code;
}HUFF_CODE;
NODE *creat_list(void);
void Init_head(void);
void Add_Left_node(NODE *p);
void Add_Right_node(NODE *p);
void Gen_Tree(NODE *Head,int color);
void show(int x1,int x2,int y,int idx);
void GO_LeftRight(int x1,int x2,int xincr,void far *buf[4]);
void GO_UpDown(int y1,int y2,int yincr,void far *buf[4]);
long filesize(FILE *input_file);
void Check_HuffmanCode(void);
void Convert_to_bitmap(void);
void Check_Eol(void);
void Check_bit(void);
void Check_Code(void);
void Error(void);
/*White 0 , Black 1*/
HUFF_CODE Terminate[64][2]={{8,0x3500,10,0X0DC0},{6,0x1C00,3,0X4000},
{4,0x7000,2,0XC000},{4,0x8000,2,0X8000},{4,0xB000,3,0X6000},
{4,0XC000,4,0X3000},{4,0XE000,4,0X2000},{4,0XF000,5,0X1800},
{5,0X9800,6,0X1400},{5,0XA000,6,0X1000},{5,0X3800,7,0X0800},
{5,0X4000,7,0X0A00},{6,0X2000,7,0X0E00},{6,0X0C00,8,0X0400},
{6,0XD000,8,0X0700},{6,0XD400,9,0X0C00},{6,0XA800,10,0X05C0},
{6,0XAC00,10,0X0600},{7,0X4E00,10,0X0200},{7,0X1800,11,0X0CE0},
{7,0X1000,11,0X0D00},{7,0X2E00,11,0X0D80},{7,0X0600,11,0X06E0},
{7,0X0800,11,0X0500},{7,0X5000,11,0X02E0},{7,0X5600,11,0X0300},
{7,0X2600,12,0X0CA0},{7,0X4800,12,0X0CB0},{7,0X3000,12,0X0CC0},
{8,0X0200,12,0X0CD0},{8,0X0300,12,0X0680},{8,0X1A00,12,0X0690},
{8,0X1B00,12,0X06A0},{8,0X1200,12,0X06B0},{8,0X1300,12,0X0D20},
{8,0X1400,12,0X0D30},{8,0X1500,12,0X0D40},{8,0X1600,12,0X0D50},
{8,0X1700,12,0X0D60},{8,0X2800,12,0X0D70},{8,0X2900,12,0X06C0},
{8,0X2A00,12,0X06D0},{8,0X2B00,12,0X0DA0},{8,0X2C00,12,0X0DB0},
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
{8,0X2D00,12,0X0540},{8,0X0400,12,0X0550},{8,0X0500,12,0X0560},
{8,0X0A00,12,0X0570},{8,0X0B00,12,0X0640},{8,0X5200,12,0X0650},
{8,0X5300,12,0x0520},{8,0X5400,12,0x0530},{8,0X5500,12,0x0240},
{8,0X2400,12,0x0370},{8,0X2500,12,0x0380},{8,0X5800,12,0x0270},
{8,0X5900,12,0x0280},{8,0X5A00,12,0x0580},{8,0X5B00,12,0x0590},
{8,0X4A00,12,0X02B0},{8,0X4B00,12,0X02C0},{8,0X3200,12,0X05A0},
{8,0X3300,12,0X0660},{8,0X3400,12,0X0670}},
MakeUp[27][2]={{5,0XD800,10,0X03C0},{5,0X9000,12,0X0C80},
{6,0X5C00,12,0X0C90},{7,0X6E00,12,0X05B0},{8,0X3600,12,0X0330},
{8,0X3700,12,0X0340},{8,0X6400,12,0X0350},{8,0X6500,13,0X0360},
{8,0X6800,13,0X0368},{8,0X6700,13,0X0250},{9,0X6600,13,0X0258},
{9,0X6680,13,0X0260},{9,0X6900,13,0X0268},{9,0X6980,13,0X0390},
{9,0X6A00,13,0X0398},{9,0X6A80,13,0X03A0},{9,0X6B00,13,0X03A8},
{9,0X6B80,13,0X03B0},{9,0X6C00,13,0X03B8},{9,0X6C80,13,0X0290},
{9,0X6D00,13,0X0298},{9,0X6D80,13,0X02A0},{9,0X4C00,13,0X02A8},
{9,0X4C80,13,0X02D0},{9,0X4D00,13,0X02D8},{6,0X6000,13,0X0320},
{9,0X4D80,13,0X0328}};
/*MakeUp code =
64,128,192,256,320,384,448,512,576,640,704,768,832,896,960,1024,
1088,1152,1216,1280,1344,1408,1472,1536,1600,1664,1728*/


byte inverse[256]={0x00,0x80,0x40,0xC0,0x20,0xA0,0x60,0xE0,0x10,
0x90,0x50,0xD0,0x30,0xB0,0x70,0xF0,0x08,0x88,0x48,0xC8,0x28,0xA8,
0x68,0xE8,0x18,0x98,0x58,0xD8,0x38,0xB8,0x78,0xF8,0x04,0x84,0x44,
0xC4,0x24,0xA4,0x64,0xE4,0x14,0x94,0x54,0xD4,0x34,0xB4,0x74,0xF4,
0x0C,0x8C,0x4C,0xCC,0x2C,0xAC,0x6C,0xEC,0x1C,0x9C,0x5C,0xDC,0x3C,
0xBC,0x7C,0xFC,0x02,0x82,0x42,0xC2,0x22,0xA2,0x62,0xE2,0x12,0x92,
0x52,0xD2,0x32,0xB2,0x72,0xF2,x0A,0x8A,0x4A,0xCA,0x2A,0xAA,0x6A,
0xEA,0x1A,0x9A,0x5A,0xDA,0x3A,0xBA,0x7A,0xFA,0x06,0x86,0x46,0xC6,
0x26,0xA6,0x66,0xE6,0x16,0x96,0x56,0xD6,0x36,0xB6,0x76,0xF6,0x0E,
0x8E,0x4E,0xCE,0x2E,0xAE,0x6E,0xEE,0x1E,0x9E,0x5E,0xDE,0x3E,0xBE,
0x7E,0xFE,0x01,0x81,0x41,0xC1,0x21,0xA1,0x61,0xE1,0x11,0x91,0x51,
0xD1,0x31,0xB1,0x71,0xF1,0x09,0x89,0x49,0xC9,0x29,0xA9,0x69,0xE9,
0x19,0x99,0x59,0xD9,0x39,0xB9,0x79,0xF9,0x05,0x85,0x45,0xC5,0x25,
0xA5,0x65,0xE5,0x15,0x95,0x55,0xD5,0x35,0xB5,0x75,0xF5,0x0D,0x8D,
0x4D,0xCD,0x2D,0xAD,0x6D,0xED,0x1D,0x9D,0x5D,0xDD,0x3D,0xBD,0x7D,
0xFD,0x03,0x83,0x43,0xC3,0x23,0xA3,0x63,0xE3,0x13,0x93,0x53,0xD3,
0x33,0xB3,0x73,0xF3,0x0B,0x8B,0x4B,0xCB,0x2B,0xAB,0x6B,0xEB,0x1B,
0x9B,0x5B,0xDB,0x3B,0xBB,0x7B,0xFB,0x07,0x87,0x47,0xC7,0x27,0xA7,
0x67,0xE7,0x17,0x97,0x57,0xD7,0x37,0xB7,0x77,0xF7,0x0F,0x8F,0x4F,
0xCF,0x2F,0xAF,0x6F,0xEF,0x1F,0x9F,0x5F,
0xDF,0x3F,0xBF,0x7F,0xFF};
HUFF_CODE EOL ={12,0x0010};
const int Compare[16]={0x8000,0x4000,0x2000,0x1000,0x800,
          0x400,0x200,0x100,0x80,0x40,0x20,0x10,8,4,2,1};
const byte bits[8]={0x80,0x40,0x20,0x10,8,4,2,1};
byte *writebuf,*readbuf;
int   WriteBuf_Pos,Buf_Pos,WriteBuf_bit,Buf_bit,count,Total_bit,Code,
      Code_bit;
int remain,Flag,END = NO;
byte far *Dat[24];
NODE *White_head,*Black_head,*Base;
FILE *source,*tempfile;
void main(int argc,char *argv[])
{
 int gd=VGA,gm=VGAHI,line,Cur_line,dy,EndX,BeginX,ix;
 byte ch,StepX,StepY,temp[30];
 long fsize,fremain;
 void far *ptr[4];
 if(argc < 2)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    {
     printf("Usage : VIEW [Huffman file] \r\n%c",7);
     exit(0);
    }
   source=fopen(argv[1],"rb");
   tempfile=tmpfile();
   if(source == NULL || tempfile == NULL)
    {
     printf("Error in opening file\n");
     exit(0);
    }
   remain=fread(temp,1,20,source);
   if((strstr(temp,"HUFFMAN_FILE") == NULL) && (strstr(temp,"BIT
FAX") == NULL))
    {
     printf("Not a Huffman file\n");
     exit(0);
    }
   for(ix=0;ix < 24;ix++)
   if((Dat[ix] = (byte far *) farmalloc(BufSize)) == NULL)
    {
     printf("Error in allocating memory\n");
     exit(0);
    }
   fseek(source,26L,SEEK_SET);
   writebuf=(byte *)malloc(BufSize);
   readbuf=(byte *)malloc(BufSize);
   if(writebuf == NULL || readbuf == NULL)
    {
     printf("Error in allocate memory\n");
     exit(0);
    }
   clrscr();
   textattr(128+YELLOW);
   cprintf("Wait for processing data");
   memset(writebuf,0,BufSize);
   Init_head();
   WriteBuf_Pos = WriteBuf_bit = Code = 0;
   Buf_bit = Buf_Pos = Total_bit= 0;
   remain = fread(readbuf,1,BufSize,source);
   readbuf[Buf_Pos] = inverse[readbuf[Buf_Pos]];
   Check_Eol();
   while(1)
     {
       Check_HuffmanCode();
       if(Total_bit == 1728)
       {
        if(END)
         break;
        Check_Eol();
        Total_bit = 0;
       }
     }
   if(WriteBuf_Pos <  BufSize)
      fwrite(writebuf,1,WriteBuf_Pos,tempfile);
   fclose(source);
   free(readbuf);
   memset(writebuf,0,BufSize);
   fsize = filesize(tempfile);
   fremain = fsize - (fsize/BufSize)*BufSize;
   if(fremain != 0)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  fwrite(writebuf,1,BufSize - fremain,tempfile);
  free(writebuf);
  line=filesize(tempfile)/BufSize;
  Cur_line = line;
  dy = line;
  EndX = 80;
  BeginX = 0;
  if(line > 24)
   {
    Cur_line = 24;
    dy = 24;
   }
  fseek(tempfile,0L,SEEK_SET);
  initgraph(&gd,&gm,"d:\\borlandc\\bgi");
  setfillstyle(SOLID_FILL,EGA_WHITE);
  for(ix=0;ix < dy;ix++)
   {
    fread(&Dat[ix][0],4320,1,tempfile);
    show(BeginX,EndX,ix,0);
   }
 while(1){
    ch = getch();
    if(ch == 0)
     {
      ch =getch();
      switch(ch){
       case LEFT      :if(EndX == 80)
                       break;
                      GO_LeftRight(0,16,155,ptr);
                      EndX-=2;
                      BeginX-=2;
                      for(ix=0;ix < dy;ix++)
                        show(BeginX,BeginX+2,ix,0);
                      break;
       case Ctrl_LEFT :if(EndX == 80)
                        break;
                      if((EndX-20) < 80)
                        StepX = EndX - 80;
                      else
                        StepX = 20;
                      GO_LeftRight(0,StepX*8,((640-StepX*8)/4)-1,ptr);
                      EndX-=StepX;
                      BeginX-=StepX;
                      for(ix=0;ix <dy;ix++)
                        show(BeginX,BeginX+StepX,ix,0);
                      break;
       case RIGHT      :if(EndX == 214)
                        break;
                      GO_LeftRight(16,0,155,ptr);
                      EndX+=2;
                      BeginX+=2;
                      for(ix=0;ix < dy;ix++)
                        show(EndX-2,EndX,ix,624);
                      break;
       case Ctrl_RIGHT:if(EndX == 214)
                        break;
                      if((EndX+20) > 214)
                        StepX = 214 - EndX;
                      else
                        StepX = 20;
                      GO_LeftRight(StepX*8,0,((640-StepX*8)/4)-1,ptr);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                EndX+=StepX;
                BeginX+=StepX;
                 for(ix=0;ix < dy;ix++)
                   show(EndX-StepX,EndX,ix,640-StepX*8);
                 break;
case DOWN     : if(line <= 24 || Cur_line == line)
                   break;
              for(ix=0;ix < 23;ix++)
                _fmemcpy(&Dat[ix][0],&Dat[ix+1][0],4320);
              fseek(tempfile,Cur_line*4320L,SEEK_SET);
              fread(&Dat[23][0],1,4320,tempfile);
              GO_UpDown(20,0,114,ptr);
                   show(BeginX,EndX,23,0);
                   Cur_line++;
                    break;
case PGDN       :  if(line <= 24 || Cur_line == line)
                      break;
              if((Cur_line+24) > line)
               {
                StepY = line-Cur_line;
                for(ix=0;ix < (24-StepY);ix++)
                 _fmemcpy(&Dat[ix][0],&Dat[ix+StepY][0],4320);
                 GO_UpDown(StepY*20,0,((480-StepY*20)/4)-1,ptr);
                }
              else
                StepY = 24;
              Cur_line+=StepY;
              fseek(tempfile,(Cur_line-StepY)*4320L,SEEK_SET);
              for(ix=0;ix < StepY;ix++)
              {
                fread(&Dat[24-StepY+ix][0],1,4320,tempfile);
                show(BeginX,EndX,24-StepY+ix,0);
              }
              break;
case UP         :  if(line <= 24 || Cur_line == 24)
              break;
              for(ix=0;ix < 23;ix++)
               _fmemcpy(&Dat[23-ix][0],&Dat[22-ix][0],4320);
              fseek(tempfile,(Cur_line-25)*4320L,SEEK_SET);
              fread(&Dat[0][0],1,4320,tempfile);
              GO_UpDown(0,20,114,ptr);
              show(BeginX,EndX,0,0);
              Cur_line--;
              break;
case PGUP       :  if(line <= 24 || Cur_line == 24)
            break;
            if((Cur_line - 24) < 24)
            {
            StepY = Cur_line - 24;
            for(ix=0;ix < (24-StepY);ix++)
            _fmemcpy(&Dat[23-ix][0],&Dat[23-ix-StepY][0],4320);
            GO_UpDown(0,StepY*20,((480-StepY*20)/4)-1,ptr);
            }else
                StepY = 24;
            fseek(tempfile,(Cur_line-24-StepY)*4320L,SEEK_SET);
            for(ix=0;ix < StepY;ix++)
             {
               fread(&Dat[ix][0],1,4320,tempfile);
               show(BeginX,EndX,ix,0);
             }
            Cur_line-=StepY;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                    break;
          }
        }else{
         if(ch == ESC)
           break;
        }
    }
    closegraph();
    for(ix=0;ix < 24;ix++)
     farfree(Dat[ix]);
}
NODE *creat_list(void)
{
    NODE *p;
    if((p =(NODE *)malloc(sizeof(NODE))) == NULL)
     {
      printf("Error in creat list\n");
      exit(0);
     }
     p->bit = 0xFF;
     p->count=0xFFFF;
     p->Left =NULL;
     p->Right=NULL;
     return p;
}
void Init_head(void)
{
    White_head=creat_list();
    Black_head=creat_list();
    Gen_Tree(White_head,WHITE);
    Gen_Tree(Black_head,BLACK);
}

void Add_Left_node(NODE *p)
{
    NODE *ptr;
     ptr=creat_list();
     p->Left = ptr;
}
void Add_Right_node(NODE *p)
{
    NODE *ptr;
     ptr=creat_list();
     p->Right=ptr;
}
void Gen_Tree(NODE *Head,int color)
{
    int i_code,i_bit;
    NODE *index;
    for(i_code=0;i_code < 65;i_code++)
     {
      index = Head;
      for(i_bit=0;i_bit < Terminate[i_code][color].bit;i_bit++)
       {
         if(Terminate[i_code][color].code & Compare[i_bit])
          {
            if(index->Right == NULL) /* bit = 1*/
              Add_Right_node(index);
            index = index->Right;
          }
         else
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        {
          if(index->Left == NULL) /* bit = 0*/
            Add_Left_node(index);
           index = index->Left;
        }
      }
     if(i_code !=64)
       index->count = i_code;
     else
       index->count = 0x0FFF;
     index->bit   = Terminate[i_code][color].bit;
    }
   for(i_code=0;i_code < 27;i_code++)
    {
     index = Head;
     for(i_bit=0;i_bit < MakeUp[i_code][color].bit;i_bit++)
      {
        if(MakeUp[i_code][color].code & Compare[i_bit])
         {
           if(index->Right == NULL) /* bit = 1*/
             Add_Right_node(index);
           index = index->Right;
         }
        else
         {
          if(index->Left == NULL) /* bit = 0*/
            Add_Left_node(index);
           index = index->Left;
         }
      }
     index->count = (i_code+1)*64;
     index->bit   = MakeUp[i_code][color].bit;
    }

}
void show(int x1,int x2,int y,int idx)
{
  int ix,iy,X_CO,Y_CO,bit;
  bar(idx,y*20,idx+(x2-x1)*8-1,y*20+19);
  for(Y_CO=y*20,iy=0;iy <20 ;iy++,Y_CO++)
   for(X_CO=idx,ix=x1;ix < x2;ix++)
    {
     for(bit = 0;bit <8;bit++)
      {
       if(Dat[y][iy*216+ix] & bits[bit])
        putpixel(X_CO,Y_CO,EGA_BLACK);
       X_CO++;
      }
    }
}
void GO_LeftRight(int x1,int x2,int xincr,void far *buf[4])
{
   unsigned size;
   int xstart, xend,block;
   size = imagesize(0,0,xincr,479);
   xstart=x1;
   xend =x1+xincr;
   for (block=0; block<=3; block++)
   {
      if ((buf[block] = farmalloc(size)) == NULL)
      {
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        closegraph();
        printf("Error: not enough heap space \n");
        for(block=0;block < 24;block++)
         farfree(Dat[block]);
        exit(1);
       }
       getimage(xstart,0,xend,479,buf[block]);
       xstart = xend+1;
       xend += xincr+1;
    }
    xstart=x2;
    for (block=0; block<=3; block++)
    {
       putimage(xstart,0,buf[block],0);
       farfree(buf[block]);
       xstart =xstart+xincr+1;
    }
}
void GO_UpDown(int y1,int y2,int yincr,void far *buf[4])
{
    unsigned size;
    int ystart, yend,block;
    size = imagesize(0,0,639,yincr);
    ystart=y1;
    yend =y1+yincr;
    for (block=0; block<=3; block++)
    {
       if ((buf[block] = farmalloc(size)) == NULL)
       {
        closegraph();
        printf("Error: not enough heap space \n");
        for(block=0;block < 24;block++)
         farfree(Dat[block]);
        exit(1);
       }
       getimage(0,ystart,639,yend,buf[block]);
       ystart = yend+1;
       yend += yincr+1;
    }
    ystart=y2;
    for (block=0; block<=3; block++)
    {
       putimage(0,ystart,buf[block],0);
       farfree(buf[block]);
       ystart =ystart+yincr+1;
    }
}
void Check_HuffmanCode(void)
{
   char ch;
   NODE *temp;
   count=0;
   Code_bit = 0;
   if(Flag == WHITE)
      temp = White_head;
   else
      temp = Black_head;
   Base =temp;
   while(1){
    Check_Code();
    Code_bit++;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
   if(Code_bit >= 14)
   {
    printf("\nError in decode HUFFMAN CODE\n");
      ch = getch();
      END = YES;
    Total_bit =1728;
    return;
   }
   if(Base->count !=-1)
    if(Base->bit == Code_bit)
     {
      count = Base->count;
      if(count > 63) .
/*first code is MakeUp code ,follow with Terminate code*/
      {
        if(count == 0x0FFF)
          {
           Total_bit = 1728;
           END = YES;
           return;
          }
      Code_bit = 0;
      Base =temp;
      while(1)
       {
          Check_Code();
          Code_bit++;
         if(Base->count != -1)
          if(Base->bit == Code_bit)
           {
             count += Base->count;
           Total_bit += count;
           Convert_to_bitmap();
           return;
            }
       }
      }
      else
      {
      Total_bit += count;
      Convert_to_bitmap();
      return;
      }
       }
  } /*end loop while*/

}
void Convert_to_bitmap(void)
{
  int ix;
   if(Flag == WHITE)
   {
    for(ix = 0;ix < count;ix++)
     {
      WriteBuf_bit++;
      if(WriteBuf_bit == 8)
       {
          WriteBuf_bit = 0;
          WriteBuf_Pos++;
          if(WriteBuf_Pos ==  BufSize)
          {
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                fwrite(writebuf,BufSize,1,tempfile);
                  WriteBuf_Pos = 0;
                memset(writebuf,0,BufSize);
              }
          }
        }
        Flag = BLACK;
      }
     else{
      for(ix = 0;ix <count;ix++)
       {
        writebuf[WriteBuf_Pos] |=bits[WriteBuf_bit++];
        if(WriteBuf_bit == 8)
         {
            WriteBuf_bit = 0;
            WriteBuf_Pos++;
            if(WriteBuf_Pos == BufSize)
             {
               fwrite(writebuf,BufSize,1,tempfile);
               WriteBuf_Pos = 0;
               memset(writebuf,0,BufSize);
             }
         }
       }
      Flag = WHITE;
     }
}

void Check_Eol(void)
{
    Code = 0;
    for(Code_bit=0;Code_bit < 12;Code_bit++)
      Check_bit();
    if(Code != EOL.code)
      while(1) /* if not found then search until code_bit is '1'*/
       {
        Check_bit();
        Code_bit++;
        if(Code_bit == 16)
            Code_bit = 0;
        if(Code)
           break;
       }
    Flag = WHITE;
}
void Check_bit(void)
{
       char ch;
        if(readbuf[Buf_Pos] & bits[Buf_bit++])
         Code |= Compare[Code_bit];
        if(Buf_bit == 8)
         {
       Buf_bit = 0;
       Buf_Pos++;
       if(Buf_Pos == remain)
       {
        Buf_Pos = 0;
        remain = fread(readbuf,1,BufSize,source);
        if(remain == 0)
         {
           printf("\nError in decode HUFFMAN CODE\n");
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
          ch = getch();
         }
       }
       readbuf[Buf_Pos] = inverse[readbuf[Buf_Pos]];
         }
}
void Check_Code(void)
{
   if(readbuf[Buf_Pos] & bits[Buf_bit++])
     Base=Base->Right;
   else
     Base=Base->Left;
   if(Buf_bit == 8)
      {
       Buf_bit = 0;
       Buf_Pos++;
       if(Buf_Pos == remain)
        {
         Buf_Pos = 0;
         remain = fread(readbuf,1,BufSize,source);
        }
       readbuf[Buf_Pos] = inverse[readbuf[Buf_Pos]];
       }
}
long filesize(FILE *input_file)
{
  long curpos, length;
     curpos = ftell(input_file);
     fseek(input_file, 0L, SEEK_END);
     length = ftell(input_file);
     fseek(input_file, curpos, SEEK_SET);
     return length;
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โปรแกรม KEYCODE.H

```
#define ESC 0x1B
#define ENTER   0x0D
#define BACKSPACE 0x08
/*extend key read 00 first */
#define PGDN 0X51
#define PGUP 0X49
#define LEFT 0X4B
#define RIGHT  0X4D
#define UP     0X48
#define DOWN   0X50
#define F1  0X3B
#define F2  0X3C
#define F3  0X3D
#define F4  0X3E
#define F5  0X3F
#define F6  0X40
#define F7  0X41
#define F8  0X42
#define F9  0X43
#define F10 0X44
#define Alt_F1  0X68
#define Alt_F2  0X69
#define Alt_F3  0X6A
#define Alt_F4  0X6B
#define Alt_F5  0X6C
#define Alt_F6  0X6D
#define Alt_F7  0X6E
#define Alt_F8  0X6F
#define Alt_F9  0X70
#define Alt_F10 0X71
#define Ctrl_F1  0X5E
#define Ctrl_F2  0X5F
#define Ctrl_F3  0X60
#define Ctrl_F4  0X61
#define Ctrl_F5  0X62
#define Ctrl_F6  0X63
#define Ctrl_F7  0X64
#define Ctrl_F8  0X65
#define Ctrl_F9  0X66
#define Ctrl_F10 0X67
#define Ctrl_LEFT 0x73
#define Ctrl_RIGHT 0x74
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โปรแกรม SERIAL.H

```
#define COM1PORT  0x03F8
#define COM2PORT  0x02F8
#define COM3PORT  0x03E8
#define COM4PORT  0X02E8
#define BUF  0
#define DLL  0
#define IER  1
#define DLH  1
#define IIR  2
#define LCR  3
#define MCR  4
#define LSR  5
#define MSR  6
#define CommBufSize  1024
#define SendBufSize  1024
#define PASS  0
#define FAIL  1
#define EOI  0x20
#define COM1  1
#define COM2  2
#define COM3  3
#define COM4  4
#define IRQ4FLAG  0x10  /* INT 0x0C */
#define IRQ3FLAG  0x08  /* INT 0x0B */
#define COM1COM3INT  0x0C
#define COM2COM4INT  0x0B
#define IMR  0x21
#define B1200  96
#define B2400  48
#define B4800  24
#define B9600  12
#define B19200  6
#define D8  3
#define D7  2
#define S1  0
#define S2  4
#define PNONE 0
#define PODD  0x80
#define PEVEN  0x18
#define OK "\r\nOK\r\n"
#define CONNECT "\r\nCONNECT\r\n"
#define RING "\r\nRING\r\n"
#define NO_CARRIER "\r\nNO CARRIER\r\n"
#define NO_DIALTONE "\r\nNO DIALTONE\r\n"
#define BUSY "\r\nBUSY\r\n"
#define XON 0x11
#define DLE 0x10
#define ETX 0x03
typedef struct {
               int x;
               int y;
               char str[35];
               }StrHead;
void interrupt (*old)();
void interrupt GetData();
void QuitSerialComm(void);
unsigned char CommBuf[CommBufSize],Response[60];
int  CommAddr,CommDataPos,Curr_Pos,installed = 0,Error;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
void tx(char ch)
{
  unsigned int i;
  for(i=0;i < 65535L;i++)
   {
    if ( inportb(CommAddr + LSR) & 0x60)
      {
         outportb(CommAddr,ch);
         return;
      }
   }
   QuitSerialComm();
   printf("\nError in THR not empty\n");
   exit(0);
}

void send_command(char *cmd)
{
  int i;
  for(i=0; cmd[i] != NULL ;i++)
    tx(cmd[i]);
}
void Wait_For(char *check,float secs,int param)
{
  clock_t start;
  int i=0;
  memset(Response,0,60);
  start=clock();
  if(Error == FAIL)
   return;
  while(1)
      {
         if(Curr_Pos != CommDataPos)
         {
           Response[i++] = CommBuf[Curr_Pos++];
           if(Curr_Pos == CommBufSize)
            Curr_Pos = 0;
           if(strstr(Response,check) != NULL)
              {
              Error =PASS;
              return;
              }
          }
         if( (clock() - start)/CLK_TCK > secs)
         {
             if(param == 0)
              {
               QuitSerialComm();
               printf("\nNo response from modem\n");
               exit(0);
              }else{
               Error=FAIL;
               return;
            }
         }
    } /*End loop  */
}

void InitSerialComm(int comno,char baudrate,char settings)
{
  int i;
```

เอกสารนี้มีลิขสิทธิ์การที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
memset(CommBuf,0,CommBufSize);
CommDataPos = Curr_Pos = 0;
Error = PASS;
switch( comno )
{
  case COM1 : CommAddr = COM1PORT; break;
  case COM2 : CommAddr = COM2PORT; break;
  case COM3 : CommAddr = COM3PORT; break;
  case COM4 : CommAddr = COM4PORT; break;
}
outportb(CommAddr+LCR,settings | 0x80);
outportb(CommAddr+DLL,baudrate);
outportb(CommAddr+DLH,0);
outportb(CommAddr+LCR,settings & 0x7F);
outportb(CommAddr+IER,1); /* enable int : Rx Ready */
outportb(CommAddr+MCR,0x0B); /* OUT2(3),RTS(1),DTR(1) */
for(i=0;i < 6 ;i++)
  inportb( CommAddr + i);  /*Clear any pending interrupt*/
if(!installed)
   switch( comno )
  {
  case COM1 : case COM3 :
         old = getvect( COM1COM3INT );
         setvect( COM1COM3INT , GetData );
         outportb( IMR ,inportb(IMR) & (~IRQ4FLAG) );
         break;

  case COM2 : case COM4 :
         old = getvect( COM2COM4INT );
         setvect( COM2COM4INT , GetData);
         outportb( IMR , inportb(IMR) & (~IRQ3FLAG) );
         break;
  }
  installed = comno;
}
void QuitSerialComm(void)
{
  if ( !installed )
      return;

  switch( installed )
  {
      case COM1 : case COM3 :
            setvect( COM1COM3INT , old );
            outportb( IMR , inportb(IMR) | IRQ4FLAG );
            break;

      case COM2 : case COM4 :
            setvect( COM2COM4INT , old );
            outportb( IMR , inportb(IMR) | IRQ3FLAG );
            break;
  }
  outportb(CommAddr+MCR,0);
  delay(800);
  installed = 0;
}

void interrupt GetData()
{
       CommBuf[ CommDataPos++ ] = inportb( CommAddr );
    if ( CommDataPos == CommBufSize )
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        CommDataPos = 0;
      outportb( 0x20 , EOI );
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โปรแกรม CONFIG.C

```
int PORT,TRANSFER_RATE,RESOLUTION,RETRY;
char _DIRECTORY[40];

/*
format file fax.ini
Fax Configuration file<EOF>
PORT ? ; 1-4 PORT COM1-COM4
TRANSFER_RATE ? ;(0-3 =>2400,4800,7200 or 9600 bps)
RESOLUTION ? ;(0-1 => Normal or Fine)
RETRY ??? ;(10,20,30,40,or 50 Time to retry new phone call (in
seconds)
DIRECTORY ???????????;Path directory for store fax data
  */
void ReadConfig(void)
{
  FILE *configfile;
  char param[150],*temp;
  if((configfile=fopen("FAX.INI","rb")) == NULL)
   {
        /* set default parameter*/
         PORT = COM2;
         TRANSFER_RATE = 3;
         RESOLUTION = 0;
         RETRY = 30;
         memset(_DIRECTORY,0,40);
   }
  else{
      memset(param,0,150);
      while(fread(param,150,1,configfile) != 0);
      if((temp=strstr(param,"PORT")) == NULL)
        PORT = COM2;
      else
        sscanf(temp+5,"%d",&PORT);
      if((temp=strstr(param,"TRANSFER_RATE")) == NULL)
        TRANSFER_RATE = 3;
      else
        sscanf(temp+14,"%d",&TRANSFER_RATE);
      if((temp=strstr(param,"RESOLUTION")) == NULL)
        RESOLUTION = 0;
      else
        sscanf(temp+11,"%d",&RESOLUTION);
      if((temp=strstr(param,"RETRY")) == NULL)
        RETRY = 30;
      else
        sscanf(temp+5,"%d",&RETRY);
      if((temp=strstr(param,"DIRECTORY")) != NULL)
        sscanf(temp+9,"%s",_DIRECTORY);
      fclose(configfile);
   }
}
void SetConfig(void)
{
   StrHead Head[]={{14,10," PORT          : "},{14,11," TRANSFER RATE
: "},{14,12," RESOLUTION    : "},{14,13," RETRY(seconds): "},{14,14,"
DIRECTORY     : "}};
   int ix,iy;
   char temp[40],ch;
   temp[0] = 0;
   MakeBorder(1,1,80,25,LIGHTGRAY,BLACK,2,'°');
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
MakeBorder(13,7,70,18,WHITE,BLUE,1,' ');
attr = WHITE+(BLUE <<4);
textattr(WHITE +(BLUE <<4));
gotoxy(15,7);
cprintf("Fax Configuration");
gotoxy(15,17);
cprintf("Press Esc to Main Command");
for(ix=0 ; ix < 5 ;ix++)
  {
   gotoxy(Head[ix].x,Head[ix].y);
   cprintf("%s",Head[ix].str);
  }
gotoxy(Head[0].x+17,Head[0].y);
cprintf("COM%d",PORT);
gotoxy(Head[1].x+17,Head[1].y);
switch(TRANSFER_RATE){
   case 0 : cprintf("2400 bps"); break;
   case 1 : cprintf("4800 bps"); break;
   case 2 : cprintf("7200 bps"); break;
   case 3 : cprintf("9600 bps"); break;
   }
gotoxy(Head[2].x+17,Head[2].y);
if(RESOLUTION == 0)
  cprintf("NORMAL");
else cprintf("FINE");
gotoxy(Head[3].x+17,Head[3].y);
cprintf("%d seconds",RETRY);
gotoxy(Head[4].x+17,Head[4].y);
if(_DIRECTORY[0] != 0)
{
 cprintf("%s",_DIRECTORY);
 temp[0] = (char)strlen(_DIRECTORY);
 strcpy(temp+1,_DIRECTORY);
}
ix = 0;
while(1)
 {
    gotoxy(Head[ix].x+17,Head[ix].y);
    ch=getch();
    if(ch == ESC)
         return;
    if(ch == 0)
      {
        ch=getch();
        switch(ch){
           case UP   : ix--;
                       iy=0;
                       if(ix < 0)
                        ix = 4;    break;
           case DOWN : ix++;
                       iy=0;
                       if(ix > 4)
                        ix = 0;    break;
           case LEFT : iy = -1; break;
           case RIGHT: iy = 1; break;
        }
       switch(ix){
           case 0 : PORT += iy;
                    if(PORT <1)
                       PORT = 4;
                    if(PORT > 4)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                PORT = 1;
              gotoxy(Head[ix].x+17,Head[ix].y);
              cprintf("COM%d",PORT);  break;
      case 1 : TRANSFER_RATE +=iy;
              if(TRANSFER_RATE <0)
                  TRANSFER_RATE = 3;
              if(TRANSFER_RATE >3)
                  TRANSFER_RATE = 0;
              gotoxy(Head[ix].x+17,Head[ix].y);
              switch(TRANSFER_RATE){
                  case 0 : cprintf("2400"); break;
                  case 1 : cprintf("4800"); break;
                  case 2 : cprintf("7200"); break;
                  case 3 : cprintf("9600"); break;
                  }  break;
     case 2 : RESOLUTION +=iy;
              if(RESOLUTION <0)
                  RESOLUTION =1;
              if(RESOLUTION >1)
                  RESOLUTION =0;
              gotoxy(Head[ix].x+17,Head[ix].y);
              if(RESOLUTION == 0)
                  cprintf("NORMAL");
              else cprintf("FINE  "); break;
    case 3 : RETRY = RETRY + 10*iy;
               if(RETRY <10)
                RETRY = 50;
               if(RETRY >50)
                RETRY = 10;
              gotoxy(Head[ix].x+17,Head[ix].y);
              cprintf("%d",RETRY); break;
    case 4 : if(getstr(Head[ix].x+17,Head[ix].y,38, temp,&ch))
             {
              switch(ch){
                case ESC: if(temp[0] != 0)
                          {
                           if(temp[temp[0]] != '\\')
                             sprintf(_DIRECTORY,"%s\\",temp+1);
                           else  sprintf(_DIRECTORY,"%s",temp+1);
                          }else _DIRECTORY[0] = 0;
                           return;
                case UP  : ix--;
                           iy=0;
                           if(ix < 0)
                             ix = 4;
                            break;
                case DOWN: ix++;
                           iy=0;
                           if(ix > 4)
                             ix = 0;
                           break;
              }
             }else{
                 ix = 0;
                 iy = 0;
             }
                if(temp[0] != 0)
                 {
                  if(temp[temp[0]] != '\\')
                      sprintf(_DIRECTORY,"%s\\",temp+1);
                  else sprintf(_DIRECTORY,"%s",temp+1);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                        }else _DIRECTORY[0] = 0;
                }
            }
        }  /*end loop while*/
}
void SaveConfig(void)
{
  FILE *configfile;
  if((configfile=fopen("FAX.INI","wb")) == NULL)
   {
     printf("Error in open config file\n");
     exit(0);
   }
 fprintf(configfile,"Fax Configuration file%c\n",0x1A);
 fprintf(configfile,"PORT %d\nTRANSFER_RATE%d\n",PORT,TRANSFER_RATE);
 fprintf(configfile,"RESOLUTION %d\n",RESOLUTION);
 fprintf(configfile,"RETRY %d\nDIRECTORY %s",RETRY,_DIRECTORY);
 fclose(configfile);
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โปรแกรม CONSOLE.C

```
int  getmode(void);
char far *screen;
char attr,WhereX,WhereY,old_attr,*Savescr;
void initscreen(void)
{
  if(getmode)
      screen=(char far *)0xb8000000L;
  else screen=(char far *)0xb0000000L;
}
int getmode(void)
{
 char far *mode =(char far *)0x400049L;
 if(*mode==7) return 0;
 else if((*mode>0) && (*mode<=3))
        return 1;
      else return -1;
}
void put(int CO_X,int CO_Y,char ch)
{
 char far *p=screen;
  p += 2*(CO_X-1) + (CO_Y-1)*160;
  *p++ = ch;
  *p = attr;
}
void putline(int CO_X,int CO_Y,char *fmt,...)
{
 char far *p=screen;
 va_list argptr;
 char s[80];
 register int i;
 va_start(argptr,fmt);
 vsprintf(s,fmt,argptr);
 p += 2*(CO_X-1) +(CO_Y-1)*160;
 for(i=0;s[i] != NULL;i++)
  {
   *p++ = s[i];
   *p++ = attr;
  }
 va_end(argptr);
}
void Save_scr(void)
{
 int ix;
 for(ix=0 ;ix <4000;ix++)
   Savescr[ix] = *(screen +ix);
 WhereX = wherex();
 WhereY = wherey();
}
void Restore_scr(void)
{
 int ix;
 for(ix=0 ;ix <4000;ix++)
  *(screen+ix) = Savescr[ix];
 gotoxy(WhereX,WhereY);
}
int getkey(void)
{
   char ch;
   ch=getch();
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    if(ch != 0)
     return(ch);
    else{
     ch = getch();
     return(ch+0x100);
    }
}

int getstr(int x,int y,int length,char *str,char *s)
{  /*extend key return 1,return 0 if already read string */
  int ch,idx;
  idx = str[0];
  gotoxy(x+str[0],y);
  while(1){
   ch=getkey();
   if(ch & 0x0100)
    {
      *s =(char)(ch & 0x00FF);
      return 1;
    }
   else{
      if(ch == ENTER)
      {
       str[idx+1] = 0;
       return 0;
      }
      if(ch == ESC)
      {
       str[idx+1] = 0;
       *s = ESC;
       return 1;
      }
      if(ch == BACKSPACE)
       {
          if(wherex() != x)
           {
            idx--;
            str[0]=(char)idx;
            str[idx+1]=0;
            gotoxy(x+idx,y);
            put(x+idx,y,' ');
           }
       }else{
           idx++;
           if(idx != length)
           {
             str[0]=(char)idx;
             str[idx] =(char)ch;
             put(x+idx-1,y,str[idx]);
             gotoxy(x+idx,y);
           }else  idx--;
       }
    }
   }
}

void MakeBorder(int x1,int y1,int x2,int y2,int fore,int back,int
i,char a)
{  /*0,1 draw border,2 no border */
 int ix,iy;
 char ch[2][6]={{'Ú','Ä','¿','³','À','Ù'},{'É','Í','»','º','È','¼'}};
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
old_attr =attr;
attr=fore+(back << 4);
if(i != 2)
 {
  put(x1,y1,ch[i][0]);
  for (ix=x1+1; ix<x2; ix++)
      put(ix,y1,ch[i][1]);
  put(x2,y1,ch[i][2]);
  for (iy=y1+1; iy<y2; iy++)
     {
       put(x1,iy,ch[i][3]);
       put(x2,iy,ch[i][3]);
     }
  put(x1,y2,ch[i][4]);
  for (ix=x1+1; ix<x2; ix++)
     put(ix,y2,ch[i][1]);
  put(x2,y2,ch[i][5]);
  for(iy = y1+1;iy <y2;iy++)
    for(ix = x1+1;ix <x2;ix++)
      put(ix,iy,a);
 }
 else
 {
  for(iy = y1;iy <=y2;iy++)
    for(ix = x1;ix <=x2;ix++)
      put(ix,iy,a);
 }
 attr = old_attr;
}
long filesize(FILE *send_file)
{
  long curpos, length;
     curpos = ftell(send_file);
     fseek(send_file, 0L, SEEK_END);
     length = ftell(send_file);
     fseek(send_file, curpos, SEEK_SET);
     return length;
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กิตติกรรมประกาศ

ในการทำโครงงานวิจัยในครั้งนี้ ผู้จัดทำขอขอบพระคุณ อาจารย์เกรียงไกร วงค์โรจนภรณ์ อาจารย์ที่ปรึกษา ที่ได้กรุณาให้คำแนะนำ ในเรื่องเอกสารข้อมูลที่เป็นประโยนช์ในการทดลองนี้ ตลอดจนแนวทางในการแก้ไขปัญหาต่างๆ ที่เกิดขึ้น รวมทั้งความช่วยเหลือในการจัดหาอุปกรณ์ที่ใช้ในการทดลอง และ ขอขอบประคุณอาจารย์ สุคนธ์ น้ำเพ็ชร ที่ได้กรุณาเอื้อเฟื้อให้ใช้สถานที่ และเครื่องโทรสารในการทดลอง ,อาจารย์ปราโมทย์ วาดเขียน ที่ได้กรุณาให้คำแนะนำในการเขียนปริญญานิพนธ์ฉบับนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เอกสารอ้างอิง

[1] CCITT. Standard tranmission of group 3 facsimile apparatus ,Recommendation T.4

[2] CCITT. Procedure for document facsimile transmission in the general switched telephone network, Recommendation T.30

[3] Robert L. Hummel , Programmer 's Technical Reference : Data and FAX communications. ,ziff-Davis Press Emryvilly, califomia , 1993

[4] User 's manual DISCOVERY 2400 bps modem, Datatronics , 1991

[5] Joe Campbell, C Programmer 's guide to Serial Communication, Howard W. Sams Company , 1987

[6] สุคนธ์ น้ำเพ็ชร,เทคโนโลยี โทรพิมพ์และโทรสาร ,2533

[7] กฤษณ์ คงพัฒนะโยธิน และกัลยา ปุรสาธิต, การส่งโทรสารโดยเครื่องคอมพิวเตอร์ผ่านแฟกซ์โมเด็ม, ปริญญานิพนธ์, 2536

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า