จดหมายอิเลคทรอนิคส์ภาษาไทย

Thai Electronic Mail

โดย

เจษฎา ศิริยันต์ 34102073

สายัณห์ เลี้ยงบุญเลิศชัย 34108419

อาจารย์ที่ปรึกษา

อาจารย์เกรียงไกร วงศ์โรจนภรณ์

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร
วิศวกรรมศาสตรบัณฑิต ภาควิชาโทรคมนาคม
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2537

ปริญญานิพนธ์ ปีการศึกษา 2537
ภาควิชา โทรคมนาคม
คณะวิศวกรรมศาสตร์
เรื่อง จดหมายอิเลคทรอนิคส์ภาษาไทย

ผู้จัดทำ
นายเจษฎา ศิริยันต์ 34102073
นายสายัณห์ เลี้ยงบุญเลิศชัย 34108419

.....

(อาจารย์เกรียงไกร วงศ์โรจนภรณ์)
อาจารย์ที่ปรึกษา

จดหมายอิเลคทรอนิคส์ภาษาไทย

ผู้จัดทำ
นายเจษฎา ศิริยันต์ 34102073
นายสายัณห์ เลี้ยงบุญเลิศชัย 34108419

อาจารย์ที่ปรึกษา
อาจารย์เกรียงไกร วงศ์โรจนภรณ์
ปีการศึกษา 2537

บทคัดย่อ

ระบบจดหมายอิเลคทรอนิคส์เป็นระบบรับฝากข้อความและรับส่งข่าวสารบนระบบเครือข่ายคอมพิวเตอร์ ซึ่งมีลักษณะการใช้งานคล้ายกับการรับส่งจดหมายในชีวิตประจำวัน โดยจะทำให้ผู้ใช้สามารถรับส่งข่าวสารบนระบบเครือข่ายได้ทั่วโลกอย่างรวดเร็ว ผู้ใช้สามารถส่งข้อมูลไปยังผู้รับใดๆได้โดยเฉพาะไม่จำกัด และสามารถกำหนดกลุ่มผู้ใช้ร่วมกันได้ นอกจากนี้ยังสามารถส่งไฟล์ข้อมูลอื่นๆนอกเหนือจากข่าวสารที่เป็นข้อความปกติเช่น ข้อมูลภาพและเสียงได้อีกด้วย จากข้อดีต่างๆจดหมายอิเลคทรอนิคส์จึงจัดเป็นโปรแกรมที่ใช้งานมากที่สุดโปรแกรมหนึ่งในระบบเครือข่ายคอมพิวเตอร์ในปัจจุบัน

สำหรับในโครงงานฉบับนี้ เป็นการพัฒนาระบบจดหมายอิเลคทรอนิคส์ภาษาไทยบนระบบเครือข่ายแบบท้องถิ่นที่ใช้เน็ตแวร์เป็นระบบจัดการ โดยใช้โปรแกรมภาษาซีร่วมกับแอสแซมบลี อาศัยหลักการของระบบจดหมายอิเลคทรอนิคส์เป็นหลักในการเขียนโปรแกรม ภายใต้ข้อจำกัดของการเขียนโปรแกรมติดต่อกับระบบจัดการ(Application Programming Interface-API) ของเน็ตแวร์ ส่วนในส่วนติดต่อกับผู้ใช้ที่เป็นภาษาไทยนั้นใช้การดึงฟอนต์ของเวิร์ดจุฬามาใช้งาน ข่าวสารที่ส่งเป็นได้ทั้งข้อความภาษาไทยธรรมดาและข่าวสารในรูปจดหมายเสียง(voice mail) ระบบรักษาความปลอดภัยของระบบจดหมายทำได้โดยการตรวจสอบผู้รับและเข้ารหัสข่าวสารเพื่อป้องกันการลักลอบอ่านทำให้ได้ระบบจดหมายอิเลคทรอนิคส์ที่มีประสิทธิภาพดียิ่งขึ้น นอกจากนี้โปรแกรมที่ได้พัฒนาขึ้นยังครอบคลุมการสื่อสารข้อมูลระหว่างเครื่องคอมพิวเตอร์โดยผ่านโมเด็มได้อีกด้วย

คำนำ

ในปัจจุบันนี้เทคโนโลยีได้ก้าวหน้าไปอย่างรวดเร็ว การมีเครื่องคอมพิวเตอร์เป็นของตนเองและการเชื่อมต่อเครื่องคอมพิวเตอร์ร่วมกันเป็นเครือข่ายจึงไม่ใช่เรื่องยากหรือเกินความสามารถนัก เมื่อมีเครื่องคอมพิวเตอร์แล้วก็ย่อมต้องใช้มันให้เป็นประโยชน์คุ้มค่าและให้ความสะดวกสบายแก่เรามากที่สุด ระบบจดหมายอิเลคทรอนิคส์ก็เป็นสิ่งหนึ่งที่ถูกประดิษฐ์คิดค้นขึ้นมาเพื่อจุดประสงค์ดังกล่าวคือ อำนวยความสะดวกให้แก่ผู้มีคอมพิวเตอร์ที่เชื่อมต่ออยู่ภายในระบบเครือข่ายเป็นหลัก โดยทำให้สามารถติดต่อกันได้อย่างรวดเร็ว ความถูกต้องแม่นยำค่อนข้างสูง เป็นประโยชน์อย่างมากต่อผู้ใช้ โดยเฉพาะกับกิจการธุรกิจต่างๆ ที่ต้องอาศัยความรวดเร็วในการดำเนินการ

แต่เนื่องจากความซับซ้อนในเรื่องเกี่ยวกับระบบเครือข่าย ระบบจดหมายอิเลคทรอนิคส์ในแบบภาษาไทยจึงยังไม่มีการพัฒนาให้เป็นที่แพร่หลาย ทางผู้จัดทำจึงคิดว่าโครงงานฉบับนี้น่าจะมีประโยชน์ต่อผู้สนใจในระบบจดหมายอิเลคทรอนิคส์และเป็นพื้นฐานในการพัฒนาจดหมายอิเลคทรอนิคส์ภาษาไทยต่อไปได้ในอนาคต

ท้ายนี้ทางผู้จัดคิดว่าโครงงานฉบับนี้น่าจะเป็นประโยชน์ต่อผู้อ่านบ้างไม่มากก็น้อย และถ้าหากเกิดมีข้อผิดพลาดประการใด ทางผู้จัดทำต้องขออภัยไว้ ณ. ที่นี้ด้วย

ผู้จัดทำ

สารบัญ

สารบัญรูป

สารบัญตาราง

บทที่ 1
บทนำ

ความก้าวหน้าทางเทคโนโลยีในปัจจุบันทำให้ราคาของเครื่องคอมพิวเตอร์ลดลงอย่างรวดเร็ว จึงมีการใช้งานเครื่องคอมพิวเตอร์กันอย่างแพร่หลาย ในหลายหน่วยงานที่มีการใช้เครื่องคอมพิวเตอร์จำนวนมาก จะมีการนำเอาเครื่องคอมพิวเตอร์แต่ละเครื่องมาทำการเชื่อมต่อเข้าด้วยกัน เกิดเป็นระบบเครือข่ายคอมพิวเตอร์ โดยมีจุดประสงค์หลักเพื่อทำให้สถานีผู้ใช้ทุกเครื่องสามารถใช้ทรัพยากรต่างๆ ที่มีอยู่อย่างจำกัดของระบบเครือข่ายคอมพิวเตอร์ได้ เช่น เครื่องพิมพ์แบบเลเซอร์ จานแม่เหล็กขนาดใหญ่ และทำให้สถานีผู้ใช้แต่ละสถานีสามารถแลกเปลี่ยนข้อมูลข่าวสารระหว่างกันได้

ระบบจดหมายอิเลคทรอนิคส์ (Electronic Mail) เป็นระบบรับฝากข้อความ และรับส่งข่าวสารบนระบบเครือข่ายคอมพิวเตอร์ ซึ่งมีลักษณะการใช้งานคล้ายกับการรับส่งจดหมายในชีวิตประจำวัน กล่าวคือระบบจดหมายอิเลคทรอนิคส์จะมีศูนย์ไปรษณีย์กลาง และตู้รับส่งจดหมายเช่นกัน แต่ต้องใช้งานบนระบบเครือข่ายคอมพิวเตอร์ ซึ่งแต่ละระบบก็มีการจัดการที่แตกต่างกันออกไป แต่อย่างไรก็ตาม จดหมายอิเลคทรอนิคส์ก็สามารถแลกเปลี่ยนข่าวสารระหว่างระบบเครือข่ายคอมพิวเตอร์ที่มีการจัดการระบบแตกต่างกันได้

ระบบจดหมายอิเลคทรอนิคส์ทำให้ผู้ใช้สามารถรับส่งข่าวสารบนระบบเครือข่ายได้ทั่วโลกอย่างรวดเร็วและประหยัดค่าใช้จ่าย ระบบรักษาความปลอดภัยของระบบจดหมายอิเลคทรอนิคส์จะกระทำโดยการตรวจสอบผู้รับ และเข้ารหัสข่าวสารเพื่อป้องกันการลักลอบอ่าน ผู้ใช้สามารถกำหนดกลุ่มผู้รับข่าวสาร ซึ่งทำให้สามารถจัดการประชุมร่วมกันได้ ระบบจดหมายอิเลคทรอนิคส์สามารถจัดเก็บข่าวสารที่ได้รับให้เป็นหัวข้อ ส่งข่าวสารนั้นต่อไปยังผู้รับรายอื่นได้ทันที และสามารถส่งไฟล์ข้อมูลอื่นๆ นอกเหนือจากข่าวสารที่ต้องการรับส่งไปได้ รวมไปถึงไฟล์ข้อมูลเสียง (voice mail) ซึ่งจะทำให้การสื่อสารข้อมูลรวดเร็วและชัดเจนยิ่งขึ้น จากข้อดีต่างๆ ของระบบจดหมายอิเลคทรอนิคส์ทำให้ปัจจุบันระบบจดหมายอิเลคทรอนิคส์จัดเป็นโปรแกรมที่มีการใช้งานกันมากที่สุดในระบบเครือข่ายคอมพิวเตอร์

นอกจากการส่งข้อมูลข่าวสารภายในระบบเครือข่ายท้องถิ่นแล้ว ยังมีการสื่อสารข้อมูลระหว่างเครื่องคอมพิวเตอร์อีกประเภทที่ได้รับความนิยมมาก ก็คือการใช้โมเด็ม(modem)ในการสื่อสารและโอนย้ายข้อมูลระหว่างกัน ซึ่งสามารถทำได้สะดวกโดยไม่ต้องเข้าร่วมในเครือข่ายคอมพิวเตอร์ใดๆ

บทที่ 2
ระบบเครือข่ายคอมพิวเตอร์แบบท้องถิ่นที่ใช้เน็ตแวร์

แนวโน้มการใช้งานระบบเครือข่ายคอมพิวเตอร์ในประเทศไทยนั้น จะเห็นว่ามีการใช้ระบบเครือข่ายคอมพิวเตอร์แบบท้องถิ่น(Local Area Network) กันอย่างแพร่หลาย และมีการติดตั้งระบบเครือข่ายแบบนี้เพิ่มมากขึ้นเรื่อยๆ เนื่องจากมีค่าใช้จ่ายที่ไม่แพงมากนัก และสามารถขยายระบบออกไปได้อีกโดยไม่ลำบากนัก จึงเหมาะกับองค์กรที่มีคอมพิวเตอร์อยู่แล้ว และต้องการเชื่อมต่อคอมพิวเตอร์เหล่านั้นเข้าเป็นระบบเครือข่ายแบบท้องถิ่น

ในโครงงานนี้ได้เลือกพัฒนาระบบจดหมายอิเล็กทรอนิคส์ภาษาไทยเพื่อใช้กับระบบเครือข่ายคอมพิวเตอร์แบบท้องถิ่นแบบที่ใช้เน็ตแวร์(NetWare) ของบริษัทโนเวลล์เป็นระบบจัดการระบบเครือข่าย เหตุผลที่เลือกพัฒนาในระบบนี้ก็คือ ระบบเครือข่ายคอมพิวเตอร์แบบที่ใช้ยูนิกซ์(UNIX) เป็นระบบจัดการที่มีความสมบูรณ์และเป็นมาตรฐานอยู่แล้ว เนื่องจากส่วนจัดการของระบบจดหมายอิเลกทรอนิคส์นั้นรวมอยู่ในตัวเคอร์เนลของระบบจัดการแบบยูนิกซ์ การรับส่งจดหมายอิเล็กทรอนิคส์สามารถทำได้โดยง่าย ผู้ใช้สามารถรับส่งข่าวสารได้ทั่วโลก ส่วนระบบเครือข่ายคอมพิวเตอร์แบบท้องถิ่นนั้น ในประเทศไทยมีการใช้ระบบจัดการระบบเครือข่ายคอมพิวเตอร์แบบเน็ตแวร์มากที่สุด แต่ตัวระบบจัดการแบบเน็ตแวร์รุ่นก่อนๆ นั้นไม่มีส่วนจัดการของระบบจดหมายอิเล็กทรอนิคส์ การใช้งานจดหมายอิเล็กทรอนิคส์จะต้องซื้อชุดโปรแกรมเพิ่มเติมต่างหาก ถ้าหากสามารถจัดทำระบบจดหมายอิเล็กทรอนิคส์แพร่หลายมากขึ้น อย่างไรก็ตามเน็ตแวร์ตั้งแต่รุ่น 4.0 เป็นต้นไปก็มีการรวมระบบMHS(Messages Handling Service) ซึ่งเป็นส่วนจัดการข้อมูลของระบบจดหมายอิเล็กทรอนิคส์เข้าไปไว้ในระบบแล้ว แต่เนื่องจากระบบจัดการมีขนาดใหญ่มาก และเหมาะที่จะใช้งานกับบนระบบเครือข่ายแบบท้องถิ่นขนาดใหญ่เท่านั้น ทำให้ไม่ค่อยมีการใช้งานกันมากนัก

2.1) ลักษณะทั่วไปของระบบเครือข่ายคอมพิวเตอร์แบบท้องถิ่น (LOCAL AREA NETWORK)

ระบบเครือข่ายคอมพิวเตอร์แบบท้องถิ่นนี้เป็นการเชื่อมต่อคอมพิวเตอร์เข้าด้วยกันในระยะใกล้ๆ ไม่เกิน 10 กิโลเมตร ให้กลายเป็นระบบเครือข่ายคอมพิวเตอร์เพื่อที่จะสามารถส่งผ่านข่าวสารข้อมูลระหว่างกันได้ โดยใช้แพ็กเก็ตข่าวสาร ระบบเครือข่ายแบบท้องถิ่น จะประกอบไปด้วยส่วนต่างๆ คร่าวๆ ดังต่อไปนี้

- สถานีผู้ใช้ (Client) เป็นเครื่องคอมพิวเตอร์ซึ่งใช้งานกันทั่วไปโดยต่อเชื่อมอยู่ที่แต่ละโหนด(Node) ของระบบเครือข่ายคอมพิวเตอร์

- ระบบจัดการของสถานีผู้ใช้ เป็นโปรแกรมควบคุมสถานีผู้ใช้ให้สามารถติดต่อกับระบบเครือข่ายคอมพิวเตอร์ได้

- การ์ดเชื่อมต่อระบบเครือข่ายคอมพิวเตอร์ เป็นการ์ดติดต่อสำหรับเชื่อมต่อสถานีผู้ใช้เข้ากับระบบเครือข่ายคอมพิวเตอร์

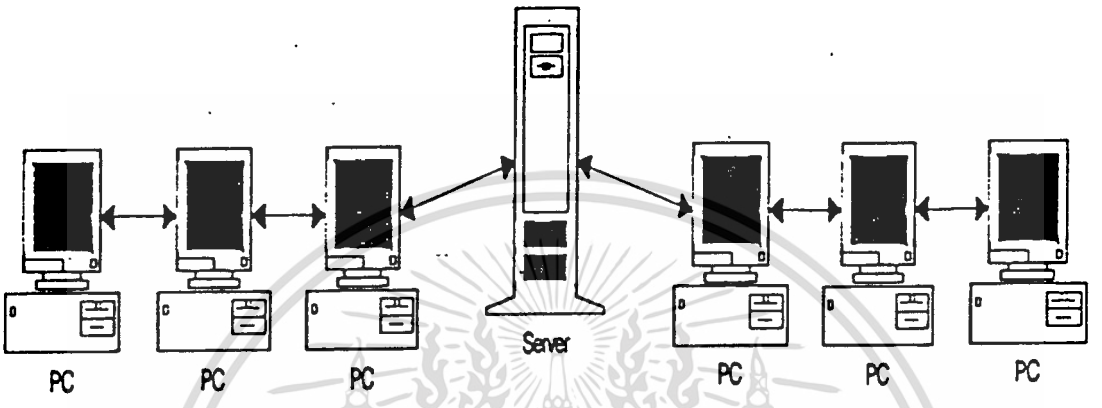
- ศูนย์บริการข้อมูล(Server) เป็นเครื่องคอมพิวเตอร์ศูนย์กลางของระบบเครือข่ายคอมพิวเตอร์ ที่มีทรัพยากรของระบบเครือข่ายคอมพิวเตอร์สำหรับให้สถานีผู้ใช้แต่ละเครื่องใช้งานร่วมกัน เช่น เครื่องพิมพ์แบบเลเซอร์ หรือ จานแม่เหล็กแบบแข็งความจุสูง เป็นต้น

- ระบบจัดการของศูนย์บริการข้อมูล เป็นโปรแกรมควบคุมตัวศูนย์บริการข้อมูลให้สามารถจัดการข้อมูลและทรัพยากรต่างๆ ตามที่สถานีผู้ใช้แต่ละเครื่องร้องขอมาได้

- สายเคเบิ้ล(Cable) สำหรับนำสัญญาณที่ใช้ในการเชื่อมต่อ ซึ่งมีหลายลักษณะที่แตกต่างกันออกไป เช่น สายสัญญาณแบบ UTP(Unshield-Twisted Pair) หรือ เส้นใยนำแสง

-เครื่องคอมพิวเตอร์แต่ละเครื่องในระบบเครือข่ายสามารถติดต่อสื่อสารกันได้โดยใช้แพ็กเก็ตข่าวสาร ในแพ็กเก็ตจะมีข้อมูลซึ่งบอกตำแหน่งของผู้ที่ส่งแพ็กเก็ตนั้นและผู้รับแพ็กเก็ต ซึ่งจะนำส่วนดังกล่าวมาใช้เพื่อการจัดเส้นทาง(Routing) การจัดระบบเครือข่ายคอมพิวเตอร์แบบท้องถิ่นโดยส่วนใหญ่แล้วจะต้องมีศูนย์บริการข้อมูลอย่างน้อย 1 เครื่อง อย่างไรก็ตามการทำงานต่างๆ จะกระทำที่เครื่องสถานีผู้ใช้นั้น ไม่ใช่บนศูนย์บริการข้อมูล และภายใต้ระบบเครือข่ายคอมพิวเตอร์แบบท้องถิ่นนั้นจะมีระบบจัดการที่ช่วยให้เครื่องสถานีผู้ใช้คอมพิวเตอร์แต่ละเครื่องสามารถใช้แฟ้มข้อมูลร่วมกัน การป้องกันเรคคอร์ด การกำหนดตั้งชื่อเครื่องและการส่งข่าวสารถึงสถานีผู้ใช้เครื่องอื่นๆ

ปัจจุบันคำว่า ระบบเครือข่ายคอมพิวเตอร์แบบท้องถิ่น มักจะหมายถึงระบบเครือข่ายคอมพิวเตอร์แบบที่มีสถานีผู้ใช้เป็นเครื่องไมโครคอมพิวเตอร์แบบส่วนบุคคล(Personal Computer - PC) ที่ใช้งานกันอยู่ทั่วไป อย่างไรก็ตามความหมายที่แท้จริงของแลน(LAN)นั้นรวมกว้างไปถึงระบบเครือข่ายคอมพิวเตอร์ที่มีสถานีผู้ใช้เป็นเครื่องคอมพิวเตอร์แบบใดๆ ก็ได้ โดยมีระยะทางการเชื่อมต่อไม่เกิน 10 กิโลเมตร



รูป 2.1 แสดงการจัดวางระบบของระบบเครือข่ายคอมพิวเตอร์แบบท้องถิ่น

2.2) การติดต่อสื่อสารของเน็ตแวร์

ซอฟต์แวร์สำหรับระบบเครือข่ายในเครื่องสถานีผู้ใช้แต่ละเครื่องจะมีการแบ่งการทำงานเป็นชั้นๆ ชั้นที่อยู่ต่ำที่สุดจะทำหน้าที่ติดต่อกับอุปกรณ์การเชื่อมต่อระบบเครือข่าย ในชั้นระดับบนที่สุดจะทำหน้าที่ติดต่อกับโปรแกรมของผู้ใช้งานระบบ และจัดการช่วยเหลือด้านการติดต่อโปรแกรมของผู้ใช้งานระบบกับกระบวนการการใช้งานระบบเครือข่าย ในซอฟต์แวร์แต่ละชั้นจะพัฒนาขึ้นโดยมีการกำหนดหน้าที่ในแต่ละชั้นไว้ก่อนล่วงหน้าแล้ว

ในการสื่อสารชั้นล่างสุดนั้นเครื่องคอมพิวเตอร์ในระบบเครือข่ายนั้นจะทำการติดต่อกับเครื่องอื่นๆ ในระบบ และกับศูนย์กลางบริการข้อมูล โดยการใช้แพ็กเก็จข่าวสาร หรือมักเรียกกันว่า เฟรม(Frame)ขั้นตอนการทำงานทั้งหมดของระบบเครือข่ายคอมพิวเตอร์แบบท้องถิ่นนั้นจะมีการรับส่งโดยใช้อุปกรณ์การเชื่อมต่อระบบเครือข่าย และการเรียกการใช้งานผ่านซอฟต์แวร์สำหรับอุปกรณ์นั้น

ซอฟต์แวร์สำหรับระบบเครือข่ายนั้น ทำการส่งเฟรมเพื่อการทำงานในหลายๆ หน้าที่ ดังนี้

- การเริ่มต้นเปิดช่องทางการสื่อสาร

- การส่งข้อมูล เช่น เรคคอร์ดจากแฟ้มข้อมูล ไปยังเครื่องคอมพิวเตอร์
- การตอบรับการรับรู้มาถึงของข้อมูล
- การกระจายข่าวสารไปยังเครื่องอื่นๆ
- การปิดช่องทางการสื่อสาร

Sender ID	Desti ID	Frame type	Data/massage	CRC

รูป2.2 แสดงรูปแบบพื้นฐานของเฟรมที่ใช้ในระบบเน็ตแวร์

การพัฒนาระบบเครือข่ายขึ้นมานั้น มีการกำหนดรูปแบบของเฟรมหลายรูปแบบแตกต่างกันไป แต่รูปแบบที่แสดงไว้นั้นเป็นข้อมูลที่มีการใช้เป็นประจำ ประกอบไปด้วย

- ตำแหน่งที่อยู่ระบบเครือข่ายของผู้ที่ส่งเฟรมนั้น
- ตำแหน่งที่อยู่ระบบเครือข่ายของผู้รับเฟรมนั้น
- ข้อมูลแสดงประเภทของข้อมูลภายในเฟรม
- ข้อมูลหรือข่าวสาร
- ข้อมูลตรวจสอบความถูกต้อง เช่น การตรวจสอบผลรวมหรือการตรวจสอบแบบ CRC

การรับส่งเฟรมจะกระทำโดยซอฟท์แวร์ระบบเครือข่าย ขั้นตอนการทำงานของเครื่องคอมพิวเตอร์จะเป็นตัวเรียกการทำงานนั้นเอง เมื่อมีการใช้แฟ้มข้อมูลที่อยู่ภายในศูนย์บริการข้อมูล หรือโดยการทำงานตามโปรโตคอลเช่น จากเน็ตไบออส(NetBIOS) หรือ IPX ที่จะส่งข่าวสารถึงเครื่องคอมพิวเตอร์เครื่องอื่นภายในระบบเครือข่าย

2.3)ส่วนประกอบของระบบเครือข่ายคอมพิวเตอร์ระดับท้องถิ่นแบบเน็ตแวร์

เน็ตแวร์เป็นการเชื่อมต่อคอมพิวเตอร์ส่วนบุคคลและอุปกรณ์อื่นๆ เช่น เครื่องพิมพ์ จานแม่เหล็กขนาดใหญ่ เข้าด้วยกัน ทำให้ผู้ใช้สามารถใช้อุปกรณ์และแฟ้มข้อมูลร่วมกัน ส่งข่าวสารสถานีผู้ใช้ไปยังอีกเครื่องหนึ่ง ระบบเครือข่ายเน็ตแวร์จะประกอบด้วยศูนย์บริการข้อมูล สถานีผู้ใช้ และอุปกรณ์ประกอบอื่นๆ ที่ต่อเชื่อมกับศูนย์บริการข้อมูล

ศูนย์บริการข้อมูล

ศูนย์บริการข้อมูลเป็นหัวใจของระบบเครือข่าย ทำหน้าที่จัดการจัดเก็บแฟ้มข้อมูลและการใช้แฟ้มข้อมูลร่วมกัน ควบคุมระบบรักษาความปลอดภัยของข้อมูล การทำงานติดต่อกันระหว่างสถานีผู้ใช้ โดยทั่วไปจะนำเครื่องคอมพิวเตอร์ส่วนบุคคลมาใช้เป็นเครื่องศูนย์บริการข้อมูล หรืออาจจะใช้เครื่องคอมพิวเตอร์ต่างชนิดกันก็ได้

ถ้ามีการใช้เครื่องคอมพิวเตอร์ชนิดอื่นๆ ทำหน้าที่เป็นศูนย์บริการข้อมูลจะต้องสามารถตรวจสอบการใช้งาน โดยจะต้องสามารถเชื่อมต่อกับระบบได้และเครื่องนั้นสามารถทำหน้าที่เป็นศูนย์บริการข้อมูลได้

ศูนย์บริการข้อมูลหลายตัวอาจจะเชื่อมต่อกันบนระบบเครือข่ายเดียวกันเป็นระบบเครือข่ายแบบหลายศูนย์บริการ(Multiserver) และหลายระบบเครือข่ายก็อาจเชื่อมต่อกันเพื่อทำเป็นระบบเครือข่ายร่วม (Internetwork) ได้

เน็ตแวร์จัดแบ่งความสามารถในการให้บริการข้อมูลของศูนย์บริการข้อมูลได้เป็น

- Dedicated server หรือ Client-server Model ระบบเครือข่ายแบบนี้จะต้องมีเครื่องคอมพิวเตอร์ที่ทำหน้าที่เป็นศูนย์บริการโดยเฉพาะ คอยบริการสถานีผู้ใช้ ลักษณะการต่อระบบเครือข่ายแบบนี้จะมีประสิทธิภาพมากกว่า และสามารถดูแลข้อมูลได้ง่ายกว่า มีระบบรักษาความปลอดภัยของข้อมูลสูง แต่ก็มีราคาแพงกว่ามาก

- Nondedicated server หรือ Peer-to-Peer Model ระบบเครือข่ายแบบนี้ คอมพิวเตอร์ที่ทำหน้าที่เป็นศูนย์บริการข้อมูลสามารถใช้งานอื่นได้ และสถานีผู้ใช้แต่ละเครื่องก็สามารถเรียกใช้ข้อมูลจากสถานีผู้ใช้อื่นๆ ได้ด้วย มีราคาถูก เหมาะกับองค์กรขนาดกลางและขนาดเล็ก

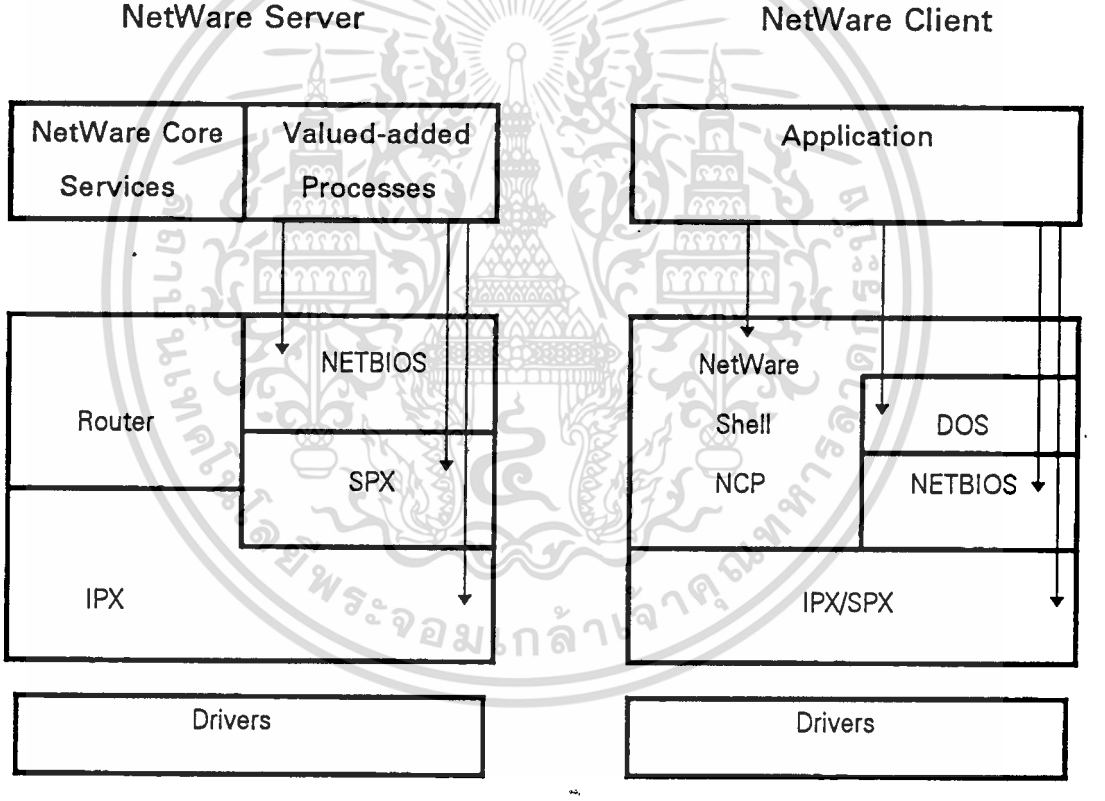
สถานีผู้ใช้

ผู้ใช้แต่ละคนสามารถสร้างเก็บและเรียกใช้งานแฟ้มข้อมูลที่อยู่ในศูนย์บริการข้อมูลได้จากสถานีของผู้ใช้ สถานีของผู้ใช้แต่ละคนจะติดต่อกับระบบจัดการเน็ตแวร์ของศูนย์บริการข้อมูลได้จะต้องโหลดซอฟท์แวร์ที่ติดต่อกับระบบเครือข่าย(Network Interface Software) เสียก่อน ส่วนของโปรแกรมที่จะติดต่อระหว่างระบบจัดการของสถานีผู้ใช้(โดยทั่วไปมักจะเป็น MS-DOS) กับระบบจัดการเน็ตแวร์ที่ศูนย์บริการข้อมูลนี้เรียกว่าเน็ตแวร์เชลล์(NetWare Shell) ซึ่งจะทำให้ผู้ใช้สามารถจะใช้งานศูนย์บริการข้อมูลได้ รูป 2.3 จะแสดงการทำงานระบบเครือข่าย การร้องขอโดยการเรียกใช้งานอินเทอร์รัพต์หมายเลข 21h จากโปรแกรมใดๆ จะถูกเชลล์ดักไว้ ถ้าเป็นการเรียกใช้ทรัพยากรที่เป็นของตัวเอง(local resource) จะถูกผ่านไปยังฟังก์ชั่นของดอสตามปกติโดยไม่มีการเรียกใช้งานระบบเครือข่าย แต่ถ้าตรวจพบว่าเป็นการร้องขอทรัพยากรของระบบเครือข่ายก็

จะเปลี่ยนทิศทางการร้องขอไปยังศูนย์บริการข้อมูลของเน็ตแวร์โดยใช้แพ็กเก็ตข่าวสารในการติดต่อสื่อสารระหว่างสถานีผู้ใช้กับศูนย์บริการข้อมูล

อุปกรณ์เชื่อมต่อระบบเครือข่าย

อุปกรณ์เชื่อมต่อระบบเครือข่ายอาจเป็นได้ทั้งแบบ ตรวจสอบการชน(Collision Sensing) หรือประเภท ตรวจสอบการผ่านมาของโทเคน (Token-passing) อุปกรณ์ทั้งสองประเภทประกอบไปด้วยอุปกรณ์ตรวจสอบสภาวะความพร้อมในการส่งเฟรม และตรวจจับเฟรมซึ่งจะส่งมาถึงตัวเอง

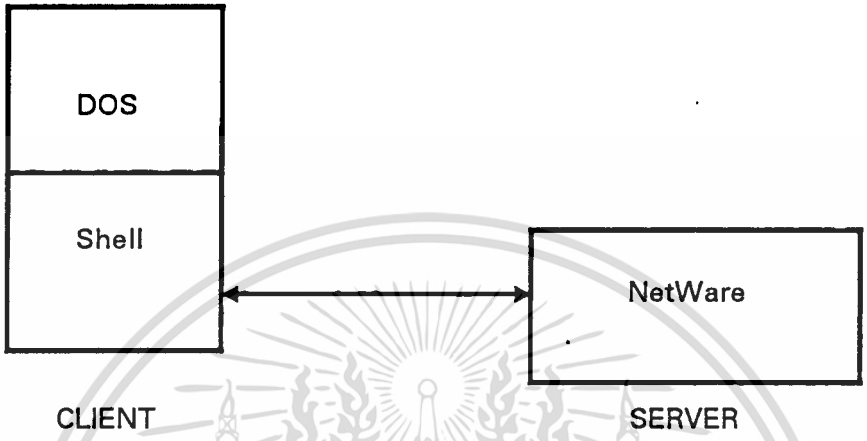


รูป 2.3 แสดงรูปแบบการติดต่อของสถานีผู้ใช้กับศูนย์บริการข้อมูล

2.4) การทำงานของเน็ตแวร์

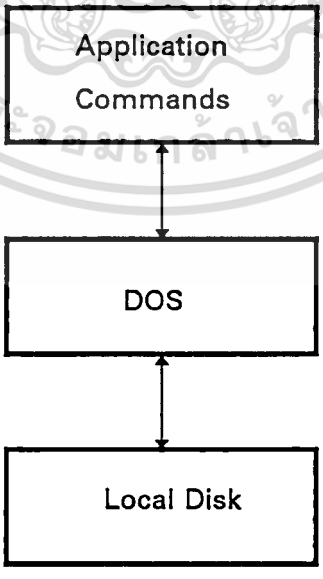
ระบบปฏิบัติการเน็ตแวร์สนับสนุนการใช้งานเครื่องไมโครคอมพิวเตอร์ที่ต่างกันหรือมีระบบปฏิบัติการที่ต่างกันโดยไม่ต้องมีการแบ่งเนื้อที่บนดิสก์ไฟล์ทั้งหมดและสถานีผู้ใช้ทุกสถานีสามารถใช้งานไดเรคทอรีร่วมกันได้

เน็ตแวร์ จะอยู่ที่ศูนย์บริการข้อมูลและดอสจะอยู่ที่สถานีผู้ใช้ เน็ตแวร์เชลล์จะอยู่ที่สถานีของผู้ใชเพื่อให้ดอสติดต่อกับเน็ตแวร์ได้



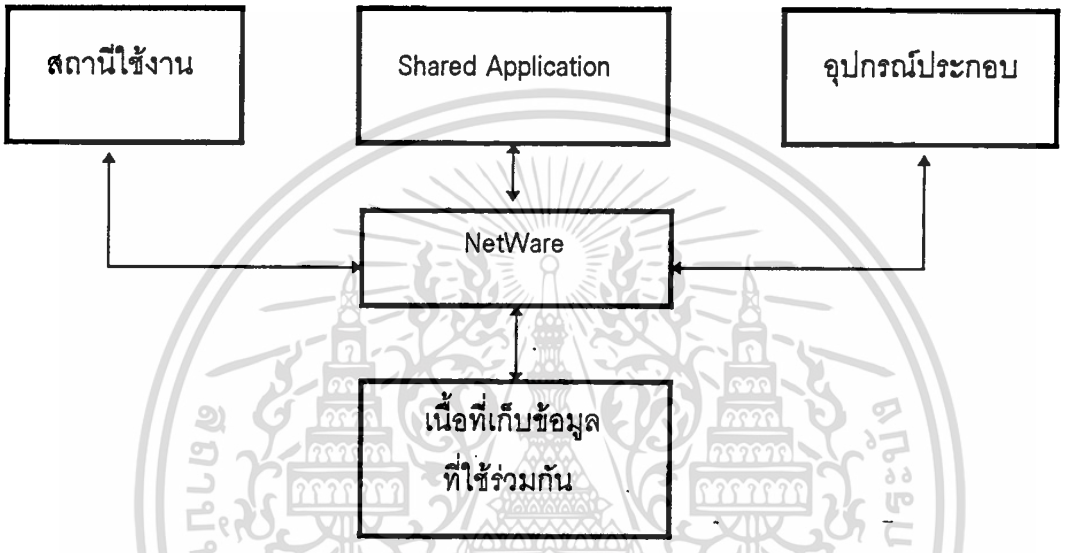
รูป 2.4 เชลล์ของเน็ตแวร์ทำหน้าที่ติดต่อระหว่างดอสของสถานีผู้ใช้งานกับศูนย์บริการข้อมูล

ดอส เน็ตแวร์ และเชลล์ จะทำงานร่วมกันเพื่อช่วยในการใช้งานข้อมูลและฮาร์ดแวร์ร่วมกัน ดอสออกแบบให้ประมวลผลโปรแกรมที่สถานีผู้ใช้ในสภาพแวดล้อมแบบผู้ใช้คนเดียว(single-user environment)



รูป 2.5 การทำงานของดอส (สภาวะแวดล้อมแบบมีผู้ใช้คนเดียว)

เน็ตแวร์ออกแบบมาให้ประมวลผลให้สภาพแวดล้อมที่มีผู้ใช้หลายคน (multiuser environment) ซึ่งจะช่วยในสการใช้งานอุปกรณ์ต่างๆร่วมกัน ใช้งานโปรแกรมร่วมกัน ควบคุมไดเรคทอรีที่ซับซ้อนและตารางแสดงตำแหน่งไฟล์ (File Allocation Table - FAT) ที่ศูนย์บริการข้อมูล



รูป 2.6 การทำงานของ เน็ตแวร์ (สภาพแวดล้อมแบบมีผู้ใช้หลายคน)

เน็ตแวร์นั้นจะไม่ใช้ดอสในสภาพแวดล้อมของระบบเครือข่าย แต่ดอสจะถูกใช้ที่สถานีผู้ใช้เท่านั้น เพื่อทำงานบนไดรฟ์ที่เป็นของตัวเองรวมทั้งโปรแกรมของผู้ใช้

เน็ตแวร์จะควบคุมการสร้างและจัดการข้อมูลบนศูนย์บริการข้อมูล ในขณะที่เน็ตแวร์เชลล์จะช่วยให้ดอสและเน็ตแวร์สามารถทำการติดต่อกันได้

ดอสจะเปรียบได้กับผู้ที่พูดและเข้าใจภาษาอังกฤษ ส่วนเน็ตแวร์ที่เป็นเหมือนผู้ที่เข้าใจและพูดเฉพาะภาษาไทย ดังนั้น เชลล์ จะเป็นเหมือนล่ามระหว่างดอสและเน็ตแวร์ ทำให้ดอสและเน็ตแวร์สามารถติดต่อกันได้

เชลล์ จะทำให้เน็ตแวร์ต่างไปจากระบบปฏิบัติการของระบบเครือข่ายอื่นๆ เพราะไม่จำกัดว่าสถานีของผู้ใช้ต้องใช้ดอสรุ่นไหน เน็ตแวร์จะสนับสนุนดอสตั้งแต่รุ่น 2.x ขึ้นไป หมายความว่าสถานีหนึ่งอาจใช้ดอส 2.0 และสถานีอื่นในระบบเครือข่ายเดียวกันอาจใช้ดอส 3.1 หรือ 3.2 ก็ได้

2.5) NetWare Core Protocol (NCP)

นอกจากเชลล์จะส่งการร้องขอการใช้งานดอสผ่านไปยังศูนย์บริการแล้ว เน็ตแวร์ยังอนุญาตให้โปรแกรมของผู้ใช้เรียกใช้การบริการที่เพิ่มขึ้นมาในเชลล์ด้วยการเรียกใช้ฟังก์ชั่น (function calls)ได้ การเรียกใช้เหล่านี้จะถูกกำหนดโดย NetWare Core Protocol (NCP) NCP มีเอกสารประกอบ และเปิดกว้างให้แก่ผู้เขียนโปรแกรม ซึ่ง NCP เป็นโปรโตคอลแบบจดทะเบียน (proprietary) ของเน็ตแวร์ โดยมีฟังก์ชั่นมากกว่า150ฟังก์ชั่นให้ผู้เขียนโปรแกรมเรียกใช้ได้

หลังจากเชลล์จัดการ NCP แล้ว มันจะส่งผ่านระบบการติดต่อสื่อสารไปยังศูนย์บริการ การร้องขอที่เข้ามาจะเริ่มกระบวนการของศูนย์บริการที่สัมพันธ์กับการเรียกใช้นั้นที่สถานีของผู้ใช้ กระบวนการแบบนี้เรียกว่ากระบวนการเรียกใช้ศูนย์บริการ(request-server-processes) ซึ่งจะเป็นการจัดการเกี่ยวกับการใช้ไฟล์ งานพิมพ์ การส่งข้อมูล(routing) ระบบรักษาความปลอดภัย และการบริการอื่นๆของศูนย์บริการข้อมูล

ฟังก์ชั่นของ NCP จะมีการบริการการรักษาความปลอดภัย การบำรุงรักษาไฟล์และไดเรคทอรี สิทธิการใช้ไฟล์ของผู้ใช้ การใช้งานศูนย์บริการข้อมูล บัญชีของผู้ใช้ การคิดค่าใช้จ่ายในการใช้งานศูนย์บริการข้อมูล การส่งข้อความระหว่างแต่ละสถานีในระบบ การจัดการกับเครื่องพิมพ์ของระบบเครือข่ายและลำดับงานพิมพ์ และฟังก์ชั่นอื่นๆอีกมากมาย

2.6) โปรแกรมเชื่อมโยงระบบเครือข่าย (API)

นอกจากผู้ใช้จะติดต่อกับศูนย์บริการข้อมูลของเน็ตแวร์ได้โดยใช้ฟังก์ชั่นของ NCP แล้ว เน็ตแวร์ยังมีโปรแกรมเชื่อมโยงระบบเครือข่าย (Application Programming Interface - API) อีกมากกว่า 20 ชนิด ซึ่ง APIs เหล่านี้จะสัมพันธ์กับฟังก์ชั่นของศูนย์บริการข้อมูล

เน็ตแวร์ 2.1 ได้เพิ่มประสิทธิภาพฟังก์ชั่นในการติดตามการใช้งานของผู้ใช้และฟังก์ชั่นในการตรวจสอบข้อผิดพลาดต่างๆ API ที่ใช้จัดการเกี่ยวกับระบบเครือข่ายและการตรวจสอบข้อผิดพลาดจะช่วยให้โปรแกรมได้รับสถิติจาก IPX/SPX ,ระบบดิสก์ และตารางการลอกอิน เป็นต้น

2.7) รูปแบบการจัดเก็บข้อมูลในหน่วยความจำของเน็ตแวร์

เน็ตแวร์มีรูปแบบการจัดเก็บข้อมูลในหน่วยความจำแตกต่างไปจากการจัดเก็บข้อมูลในสถาปัตยกรรมของไมโครโปรเซสเซอร์ตระกูลของอินเทล 80x86 ซึ่งใช้รูปแบบของการจัดเก็บข้อมูลในหน่วยความจำเป็นแบบสูงต่ำ(High-low Format) คือจะเก็บไบท์นัยสำคัญต่ำ (LSB) ไว้ก่อนไบท์นัยสำคัญสูง (MSB) เช่นข้อมูลค่า 0x1234 จะถูกจัดเก็บเป็น 0x3412 หรือค่า 0x12345678 จะ

ถูกจัดเก็บเป็น 0x87564312 ส่วนตัวเน็ตแวร์จะเก็บค่าข้อมูลเหล่านี้ไว้ในทางกลับกัน การเขียนโปรแกรมจึงต้องมีการสลับค่าไบท์สูง-ต่ำอยู่บ่อยๆ

2.8)ส่วนบริการฐานข้อมูลเครือข่าย (Bindery Services)

บินเดอรี(Bindery) เป็นโปรแกรมแบบฝังตัวที่อยู่ในแต่ละศูนย์บริการข้อมูล โดยทำหน้าที่เป็นฐานข้อมูลของระบบเครือข่าย ซึ่งประกอบไปด้วยรายการของออบเจคท์ (Objects) สำหรับระบบจดหมายอิเล็คทรอนิคส์จะมีการเรียกใช้การบริการของบินเดอรี เพื่อค้นหารายละเอียดของออบเจคท์ที่กำหนดบนเครือข่าย

ออบเจคท์ประกอบไปด้วยชื่อของออบเจคท์ และแบบของออบเจคท์ อันได้แก่ ผู้ใช้ กลุ่มของผู้ใช้ ไดเรคทอรี แถวรองงานพิมพ์ ศูนย์บริการที่ต่ออยู่และรายชื่ออื่นๆ บนระบบเครือข่าย แต่ละออบเจคท์จะมีคุณสมบัติประจำตัวที่เรียกว่า พรอพเพอร์ตี้ (property) เช่นข้อมูลที่เกี่ยวข้องกับออบเจคท์แบบผู้ใช้ (user) จะได้แก่ กลุ่มที่ผู้ใช้ใช้อยู่ ข้อมูลของการล็อกอิน และรหัสผ่าน ซึ่งก็จะมีพรอพเพอร์ตี้เป็น GROUP_I'M_IN, LOGIN_CONTROL และ PASSWORD แต่ละพรอพเพอร์ตี้จะมีค่า (value) ของมันอยู่ เช่น GROUP_I'M_IN จะเก็บรายชื่อของกลุ่มที่ผู้ใช้อยู่ และ PASSWORD จะเก็บรหัสผ่านของผู้ใช้ที่ผ่านการเข้ารหัสมาแล้ว

ออบเจคท์ในบินเดอรี(Bindery Objects)

ออบเจคท์เป็นองค์ประกอบพื้นฐานของบินเดอรี่ ซึ่งถูกกำหนดขึ้นโดย ชื่อและแบบของออบเจคท์ เช่นแบ่งเป็น ผู้ใช้ (user), กลุ่มผู้ใช้ (group), ศูนย์บริการข้อมูล(file server) หรืออะไรก็ตามที่ผู้พัฒนาต้องการ (ดังนั้นเราสามารถใช้ 2 ออบเจคท์ที่ชื่อ SERVER-1 เหมือนกัน โดยออบเจคท์หนึ่งสามารถใช้เป็นศูนย์บริการข้อมูล อีกออบเจคท์หนึ่งใช้เป็น ศูนย์บริการการพิมพ์ก็ได้

ความปลอดภัยของออบเจคท์(Object Security)

ความปลอดภัยของบินเดอรี่มีผลต่อ ความปลอดภัยของระบบเครือข่าย ดังนั้นจึงมีการจัดระดับชั้นความปลอดภัยขึ้นในบินเดอรีให้สำหรับแต่ละออบเจคท์ โดยกำหนดอยู่ในรูปสิทธิต่างๆ ในการอ้างถึงบินเดอรีดังแสดงในตาราง2.2

การตั้งค่าเกี่ยวกับความปลอดภัยสำหรับแต่ละออบเจคท์และพรอพเพอร์ตี้จะมี2 ลักษณะ คือ อนุญาตหรือปฏิเสธใน สิทธิ การอ่านข้อมูล กับ ควบคุมการบันทึกค่าสิทธิ(เพิ่มหรือลดค่าของพรอพเพอร์ตี้ต่างๆ) การตั้งค่าความปลอดภัยจะใช้เพียงไบต์เดียว; ใช้ 4 บิทด้านสูงเก็บค่าสิทธิ, ส่วนด้านต่ำใช้กำหนดสิทธิการอ่าน โดยค่าปกติของเน็ตแวร์คือ ชั้นลอกอินสำหรับการอ่านข้อมูล และชั้นผู้ดูแลสำหรับการบันทึกค่าสิทธิ

นอกจากเรื่องของความปลอดภัย (security field) ยังมีอีกหลายฟิลด์ที่เกี่ยวข้องกับแต่ละออบเจคท์ เช่น แฟล็ค, หมายเลขประจำเครื่อง, ชนิดของออบเจคท์ เป็นต้น

ตาราง 2.1 แสดงความปลอดภัยในการเข้าถึงบินเดอรี

ระดับการเข้าถึง	ผู้ที่ได้รับอนุญาต	ความหมาย
0	ทุกคน	ทุกออบเจคท์สามารถเข้าถึงได้ไม่ว่าจะมีการลอก อินเข้าสู่ศูนย์บริการข้อมูลหรือไม่ก็ตาม
1	Logged	อนุญาตให้เฉพาะลูกข่าย(client) ที่ลอกอินเข้าสู่ศูนย์บริการข้อมูลเท่านั้น
2	Object	อนุญาตให้เฉพาะลูกข่ายที่ลอกอินเข้าศูนย์บริการข้อมูลด้วยชื่อของออบเจคท์, ชนิด และรหัสผ่านที่กำหนดไว้เท่านั้น
3	Supervisor	อนุญาตให้เฉพาะ supervisor หรือออบเจคท์ที่เทียบเท่า เท่านั้น(ออบเจคท์จะมีฐานะเทียบเท่ากับsupervisor ได้ ก็ต่อเมื่อมีคุณสมบัติ SECURITY_EQUALS บรรจุอยู่ในหมายเลขของออบเจคท์ของ supervisor)
4	NetWare	มีเพียงเน็ตแวร์เท่านั้นที่สามารถเข้าถึงออเจคท์หรือพรอพเพอร์ตี้ในความปลอดภัยระดับนี้ได้

ชนิดของออบเจคท์(Bindery Object Types)

จะมีการกำหนดค่าขนาด 2 ไบต์ขึ้นมาเพื่อแสดงจุดประสงค์ของแต่ละออบเจคท์ในบินเดอรี โดยค่านี้จะถูกเซตให้เป็น 0 เมื่อไม่ทราบชนิดของออบเจคท์นั้น ชนิดของออบเจคท์ที่ถูกกำหนดขึ้นนี้จะเป็นสิ่งที่แสดงถึงจุดประสงค์ของออบเจคท์นั้น เช่น ผู้พัฒนาสร้างออบเจคท์ชนิด 9 ,ชื่อ Archive Server, ขึ้น เพื่อเป็นออบเจคท์ที่มีการปฏิบัติงานที่สอดคล้องกับโปรโตคอลของโนเวลล์ที่ชื่อ Archive Server

ออบเจทค์แฟลก(The Bindery Object Flag)

ออบเจคท์แฟลกจะมีขนาด 1 ไบต์ ใช้แสดงให้เห็นว่าออบเจคท์นั้นเป็นแบบสแตติค (ให้ค่าเป็น 0) หรือไดนามิก (ให้ค่าเป็น 1) โดยออบเจคท์แบบสแตติก จะคงอยู่ต่อไปจนกระทั่งถูกลบออกอย่างตั้งใจ ส่วนแบบไดนามิกนั้นจะถูกลบออกโดยอัตโนมัติเมื่อยกเลิกการทำงานของศูนย์บริการข้อมูล ผู้ใช้(user)เป็นตัวอย่างของออบเจคท์แบบสแตติก ส่วนศูนย์บริการแบบแอดเวอร์ไทซิง(Advertising file server) ถือเป็นตัวอย่างของออบเจคท์แบบไดนามิก

แฟลกกำหนดระดับความปลอดภัย(The Bindery Security Flag)

เป็นแฟลกที่มีขนาด 1 ไบต์ ใช้กำหนดสิทธิให้ผู้ใช้ว่าอยู่ในระดับชั้นใดในระดับชั้นความปลอดภัย

พรอพเพอร์ตี้ของออบเจคท์(Properties of Bindery Objects)

พรอพเพอร์ตี้เป็นตัวกำหนดความสัมพันธ์ระหว่างค่ากับออบเจคท์ว่าค่าควรจะสัมพันธ์กับออบเจคท์ใด โดยพรอพเพอร์ตี้เหล่านี้จะสอดคล้องกับลักษณะพิเศษของออบเจคท์หรือไม่ก็ได้ พรอพเพอร์ตี้ของเน็ตแวร์ที่ถูกกำหนดไว้ได้แก่ GROUP_I'M_IN, IDENTIFICATION และ SECURITY_EQUALS และเช่นเดียวกับออบเจคท์ พรอพเพอร์ตี้ก็จะถูกแสดงให้เห็นอยู่ในหลายๆ ฟิลด์ เช่น ชื่อ, แฟลก, ระดับความปลอดภัย เป็นต้น

พรอพเพอร์ตี้แฟลก(The Bindery Property Flag)

มีลักษณะเหมือนกับออบเจคท์แฟลก คือเป็นแฟลกขนาด 1 ไบต์ ใช้ระบุว่าพรอพเพอตี้นั้นเป็นแบบสแตติกหรือไดนามิกเช่นเดียวกัน โดยเมื่อออบเจคท์แบบไดนามิกถูกลบไป พรอพเพอร์ตี้และค่าก็จะถูกลบไปด้วย แต่จะต่างกับออบเจคท์แฟลกตรงที่ บิทสูงสุดจะถือเป็นแฟลกที่ใช้ระบุว่าพรอพเพอร์ตี้นี้เป็นแบบ item (0) หรือเป็นแบบ set (1) (set เป็นรายชื่อของหมายเลขของออบเจคท์ (Object ID) ขนาด1 4 ไบต์ ส่วน item เป็นส่วนฟิลด์ข้อมูลพิเศษซึ่งบรรจุข้อมูลของโครงสร้างใดๆ ไว้ เช่นรหัสผ่าน)

แฟลกแสดงระดับความปลอดภัย(The Bindery Security Flag)

มีลักษณะเหมือนกับออบเจคท์คือ เป็นแฟลกที่มีขนาด 1 ไบต์ ใช้กำหนดสิทธิในการอ่าน-บันทึกค่าพรอพเพอร์ตี้

โดยต่อไปนี้จะเป็นตัวอย่างของพรอพเพอร์ตี้ที่โนเวลล์กำหนดขึ้น

- ACCOUNT_SERVERS เป็นชุดของพรอพเพอร์ตี้ที่เก็บรายชื่อของทุกศูนย์บริการข้อมูลที่มีอำนาจต่อผู้ใช้ในการให้บริการ สิทธิในการเข้าถึงของพรอพเพอร์ตี้นี้คือสิทธิอ่านสำหรับผู้ลอกอิน

ทั่วไป ,และสิทธิเขียนสำหรับผู้ควบคุมเครือข่าย ออบเจคท์ของพรอพเพอร์ตี้นี้สามารถใช้เป็น ศูนย์บริการข้อมูล, ศูนย์บริการการพิมพ์, ศูนย์บริการฐานข้อมูล, ผู้ใช้ หรือออบเจคท์แบบอื่น ๆ

- ACCOUNT_LOGOUT เป็นพรอพเพอร์ตี้แบบสแตติกของศูนย์บริการข้อมูล ใช้เป็นคุณสมบัติที่บอกให้รู้ว่าแอคเคานท์ใดที่ทำการลอกเอาท์ออกจากศูนย์บริการข้อมูลนั้นแล้ว สิทธิในการเข้าถึงปกติจะเป็นระดับผู้ควบคุมทั้งการอ่านและการเขียน

- GROUP_MEMBERS เป็นเซตของพรอพเพอร์ตี้แบบสแตติก บรรจุชื่อของสมาชิกของกลุ่มผู้ใช้ สิทธิการเข้าถึงปกติจะอยู่ที่ผู้ลอกอินทั่วไปสำหรับการอ่านและอยู่ที่ผู้ควบคุมสำหรับการเขียน

- GROUP_I'M_IN มีลักษณะตรงข้ามกับ GROUP_MEMBERS โดยเป็นเซตของพรอพเพอร์ตี้แบบสแตติกที่บรรจุรายชื่อของกลุ่มที่ผู้ใช้เป็นเจ้าของอยู่ สิทธิการเข้าถึงปกติจะเป็นอ่านสำหรับผู้ลอกอินทั่วไป และเขียนสำหรับผู้ควบคุมเครือข่าย

- IDENTIFICATION เป็นพรอพเพอร์ตี้แบบสแตติกที่บรรจุชื่อเต็มของออบเจคท์ ไม่ว่าจะเป็นผู้ใช้หรือกลุ่มผู้ใช้ มีความยาวตั้งแต่ 0-128 ไบต์

- LOGIN_CONTROL แสดงคุณสมบัติที่เป็นชนิดของผู้ใช้ เช่นความยาวของรหัสผ่านที่ต้องการ, วันหมดอายุของรหัสผ่าน เป็นต้น

ค่าของข้อมูล(Value)

คือ ค่าของข้อมูลจริงที่ติดต่อกับโปรแกรม ค่าข้อมูลนี้จะเป็นแบบ item หรือ set ก็ขึ้นอยู่กับค่าพรอพเพอตี้แฟลก โดยค่าข้อมูลนี้สามารถมีขนาดใหญ่กว่า 128 ไบต์ของแพ็กเก็จตอบรับได้ ซึ่งแพ็กเก็จก็จะทำการเชื่อมต่อหลายๆ แพ็กเก็จเข้าด้วยกันเอง (เป็น Multiple Packet)

บินเดอรีไฟล์(Bindery Files)

เน็ตแวร์ 286 จะมีบินเดอรีที่ประกอบไปด้วย 2 ไฟล์หลักคือ NET$BIND.SYS เก็บค่าของออบเจคท์และพรอพเพอร์ตี้ กับ NET$VAL.SYS เก็บค่าของ value ส่วนเน็ตแวร์ 386 จะมี 3 บินเดอรีไฟล์ คือ NET$OBJ.SYS, NET$PROP.SYS และ NET$VAL.SYS โดยไฟล์เหล่านี้จะถูกกำหนดไว้เป็นไฟล์แบบ Hidden& System และถูกเก็บอยู่ในไดเรคทอรี SYS:SYSTEM ของแต่ละศูนย์บริการข้อมูล โดยมีศูนย์บริการทำหน้าที่ปิดเปิด

<u>ฟังก์ชั่นต่าง ๆในบินเดอรี API (Calls in The Bindery API)</u>

ข้อมูลจะแจ้งให้ทราบว่า ฟังก์ชั่นใดใช้ได้ในเน็ตแวร์เวอร์ชั่นใดบ้าง มีประโยชน์อย่างไรบ้าง แค่เพียงคร่าวๆ ดังต่อไปนี้

- OpenBindery (E3h 45h) เป็นฟังก์ชั่นที่มีใช้ในแอดวานซ์เน็ตแวร์ 1.0, 1.02, 2.0, 2.1และในเน็ตแวร์ 386 เวอร์ชั่นต่างๆ โดยจะทำหน้าที่เป็นบินเดอรี ซึ่งจะต้องถูกเรียกใช้หลังจากมีการปิดการใช้งานบินเดอรีโดยใช้ฟังก์ชั่น Close Bindery แล้ว เพราะบินเดอรีอื่นจะไม่สามารถทำงานได้ถ้ามีบินเดอรีที่ยังไม่ถูกปิด

- CloseBindery (E3h 44h) เป็นฟังก์ชั่นที่มีใช้ในแอดวานซ์เน็ตแวร์ 1.0, 1.02, 2.0, 2.1 และเน็ตแวร์ 386 เวอร์ชั่นต่างๆ ทำหน้าที่ปิดบินเดอรี โดยมีเพียงผู้ควบคุมระบบหรือผู้มีฐานะเทียบเท่าเท่านั้นที่สามารถปิดบินเดอรีได้

- CreateBinderyObject (E3h 32h) มีใช้ในเน็ตแวร์เวอร์ชั่นต่างๆ เหมือนฟังก์ชั่นที่กล่าวมาทำหน้าที่สร้างออบเจคท์ขึ้นในบินเดอรี โดยต้องให้โปรแกรมเมอร์เป็นผู้ระบุชื่อของออบเจคท์, ชนิด, ระดับความปลอดภัย และค่าในแฟลกต่างๆ ของออบเจคท์

- ScanBinderyObject (E3h 37h) ทำหน้าที่สแกนบินเดอรี ส่งกลับค่าในรูปของออบเจคท์และข้อมูลต่างๆ ที่เกี่ยวกับออบเจคท์ในบินเดอรี (ได้แก่ ID, ชื่อ, ชนิด และค่าในแฟลกต่างๆ)

บทที่ 3
หลักและส่วนประกอบของระบบจดหมายอิเลคทรอนิคส์

ระบบจดหมายอิเลคทรอนิคส์ (Electronic Mail หรือ E-mail) เป็นระบบรับส่งข่าวสารจากคอมพิวเตอร์เครื่องหนึ่งไปยังเครื่องคอมพิวเตอร์เครื่องอื่นๆ ที่เชื่อมต่อกันอยู่บนระบบเครือข่ายคอมพิวเตอร์ แรกเริ่มนั้น ระบบจดหมายอิเลคทรอนิคส์ถูกพัฒนามาจากการใช้งานในการติดต่อสื่อสารกันระหว่างนักวิจัยที่ใช้ระบบเครือข่ายคอมพิวเตอร์เดียวกัน แต่อยู่ต่างห้อง ต่างอาคารกัน ซึ่งการติดต่อข่าวสารโดยตรงจะไม่สะดวก จึงได้มีการพัฒนาระบบรับส่งข่าวสารโดยใช้ระบบจดหมายอิเลคทรอนิคส์ขึ้น การพัฒนาในช่วงแรกๆ จดหมายอิเลคทรอนิคส์จะมีการใช้งานจำกัด โดยผู้ใช้ทั้งสองฝ่ายจะต้องอยู่หน้าจอคอมพิวเตอร์เพื่อติดต่อสื่อสารกันโดยตรง ต่อมาได้มีการพัฒนาระบบรับฝากข่าวสารขึ้น เพื่อให้ระบบสามารถทำงานอยู่ได้อย่างอัตโนมัติ โดยระบบจะเตือนผู้รับว่ามีข่าวสารที่ยังไม่ได้อ่าน เมื่อผู้รับใช้งานระบบเครือข่ายคอมพิวเตอร์ในครั้งต่อมา

3.1)การใช้งานระบบจดหมายอิเลคทรอนิคส์

โดยทั่วไปแล้วระบบจดหมายอิเลคทรอนิคส์ของผู้ผลิตแต่ละบริษัทจะมีฟังก์ชั่นและการใช้งานที่คล้ายๆ กันดังต่อไปนี้

1.เข้าสู่ระบบจดหมายอิเลคทรอนิคส์โดยการป้อนชื่อผู้ใช้และรหัสผ่าน เราจะเห็นจดหมายอิเลคทรอนิคส์ที่เราได้รับเรียงตามลำดับ ซึ่งมักจะเป็นลำดับของเวลา โดยจดหมายอิเลคทรอนิคส์ฉบับล่าสุดจะอยู่อันดับแรก หรืออาจจะเรียงลำดับแบบอื่นๆ เช่น ชื่อผู้ส่งก็ได้ ข้อมูลที่แสดงจะประกอบไปด้วย ชื่อของจดหมายอิเลคทรอนิคส์ ชื่อผู้ส่ง แฟลกแสดงสถานะของจดหมายอิเลคทรอนิคส์ว่าอ่านหรือยังไม่อ่าน เป็นต้น

2.ในการส่งจดหมายอิเลคทรอนิคส์ สามารถเลือกรายชื่อผู้รับได้จากไดเรคทอรีแสดงรายชื่อสมาชิกของผู้ใช้ระบบจดหมายอิเลคทรอนิคส์ (Directory Service) โดยสามารถเลือกส่งให้ผู้รับทุกคนได้เช่นเดียวกับระบบบุลเลติลบอร์ด หรือระบุผู้รับ ในบางระบบอาจจะมีฟังก์ชั่นที่สามารถติดต่อกันได้โดยตรง(online Mail) หรือทำการประชุมร่วม (Conference) ก็ได้

3.สร้างไฟล์จดหมายที่เป็นข้อความโดยการใช้ เท็กซ์อิดิเตอร์ (Text Editor) หรือ เวิร์ดโปรเซสเซอร์ (Word Processor) หรือ อิดิเตอร์ของจดหมายอิเลคทรอนิคส์เอง

4.ทำการส่งจดหมายอิเลคทรอนิคส์ โดยสามารถแนบไฟล์อื่นมาพร้อมกับจดหมายอิเลคทรอนิคส์ได้ ซึ่งชนิดของไฟล์และจำนวนไฟล์ที่สามารถส่งได้จะขึ้นอยู่กับการออกแบบระบบจดหมายอิเลคทรอนิคส์ของแต่ละบริษัท หลังจากทำการส่งแล้ว จดหมายอิเลคทรอนิคส์จะไปปรากฏที่ตู้รับจดหมาย (Mailbox) ของผู้รับทันที

5.ถ้าผู้รับกำลังใช้งานระบบเครือข่ายอยู่ในขณะนั้น จะมีเสียงเตือนหรือข้อความเตือนบอกว่า มีจดหมายอิเลคทรอนิคส์ส่งมา หรือถ้าผู้รับไม่ได้ใช้งานอยู่ เมื่อผู้ใช้เข้าใช้งานเครือข่ายในเวลาต่อมาก็จะมีสัญญาณเตือนผู้ใช้เช่นกัน

6.จดหมายอิเลคทรอนิคส์ที่ส่งมาสามารถอ่านได้โดยใช้เท็กซ์อิดิเตอร์ สามารถทำสำเนาไฟล์เพื่อเก็บไว้หรือส่งไปยังผู้รับคนอื่น ขึ้นอยู่กับฟังก์ชั่นและความสามารถของระบบจดหมายอิเลคทรอนิคส์ที่ใช้อยู่

3.2) ส่วนประกอบของระบบจดหมายอิเลคทรอนิคส์

ระบบจดหมายอิเลคทรอนิคส์จะประกอบไปด้วย 2 ส่วนหลัก ได้แก่

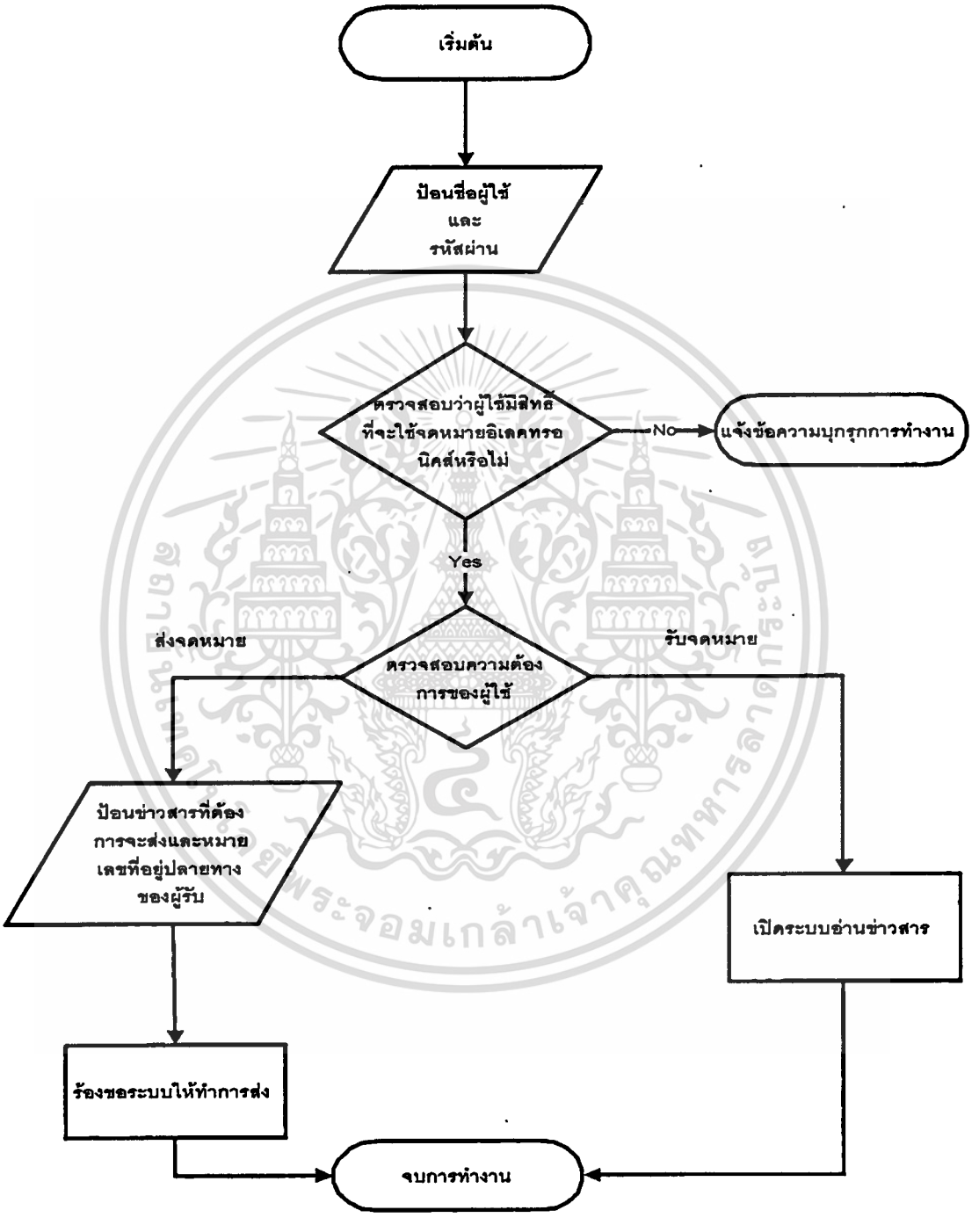
1. **ฟร้อนท์เอนท์** (Front End)**หรือยูสเซอร์ เอเจนท์** (User Agent - UA)**เป็น**ส่วนแสดงผลและรับคำสั่งจากผู้ใช้ซึ่งเป็นส่วนที่ผู้ใช้สามารถสัมผัสได้โดยตรง ผู้พัฒนาแต่ละบริษัทจะออกแบบให้ผู้ใช้สามารถใช้งานได้ง่าย (User friendly) ซึ่งมักจะประกอบไปด้วยเมนูที่มีฟังก์ชั่นการทำงานต่างๆ ดังนี้

กลุ่มฟังก์ชั่นพื้นฐาน

- Creat ใช้ในการสร้างจดหมายอิเลคทรอนิคส์ที่ใช้ในการส่ง
- Send ใช้ส่งจดหมายอิเลคทรอนิคส์ที่สร้างแล้ว
- Notification ทำหน้าที่เตือนเมื่อได้รับจดหมายอิเลคทรอนิคส์เข้ามาใหม่
- Inbox เป็นส่วนแสดงรายชื่อของจดหมายอิเลคทรอนิคส์ที่ได้รับ
- Read ใช้อ่านจดหมายอิเลคทรอนิคส์ สามารถเลื่อนอ่านข้อความในจดหมายอิเลคทรอนิคส์ได้ทั้งหมด
- Save ใช้ในการสำเนาไฟล์ลงแผ่นดิสก์

กลุ่มฟังก์ชั่นระดับปานกลาง

- Outbox เป็นส่วนแสดงรายชื่อจดหมายอิเลคทรอนิคส์ที่ส่งเพื่อยืนยันว่าได้ส่งจดหมายไปเรียบร้อยแล้ว
- Forward ทำหน้าที่ส่งจดหมายอิเลคทรอนิคส์ที่ได้รับมาต่อไปยังผู้รับคนอื่น



รูป 3.1 โฟลว์ชาร์ตแสดงการทำงานโดยทั่วไปของระบบจดหมายอิเลคทรอนิคส์

- Carbon Copy ทำหน้าที่ส่งจดหมายอิเลคทรอนิคส์ไปยังผู้รับคนอื่น แต่จดหมายอิเลคทรอนิคส์เดิมยังอยู่

- Attachment ทำหน้าที่แนบไฟล์อื่นมาด้วย

- Encryption ทำหน้าที่เข้ารหัสจดหมายอิเลคทรอนิคส์

- Message Priority ทำหน้าที่บอกระดับความเร่งด่วนของจดหมายอิเลคทรอนิคส์

- Mailing List แสดงรายชื่อผู้รับเพื่อความสะดวกในการส่ง

กลุ่มฟังก์ชั่นพิเศษ

- Viewing Attached File เพื่อตรวจดูไฟล์ที่แนบมา โดยอาจจะอยู่ในรูปแบบของไฟล์รูปภาพ หรือไฟล์แบบกระดาษทดอิเลคทรอนิคส์ หรือไฟล์แบบฐานข้อมูล

- Searching Message by Criteria ทำหน้าที่ค้นหาจดหมายอิเลคทรอนิคส์ตามวันเวลา ชื่อผู้ส่งหรือชื่อจดหมายอิเลคทรอนิคส์

- Rules-based Message Management ทำหน้าจัดการกับไฟล์ที่รับเก็บไว้

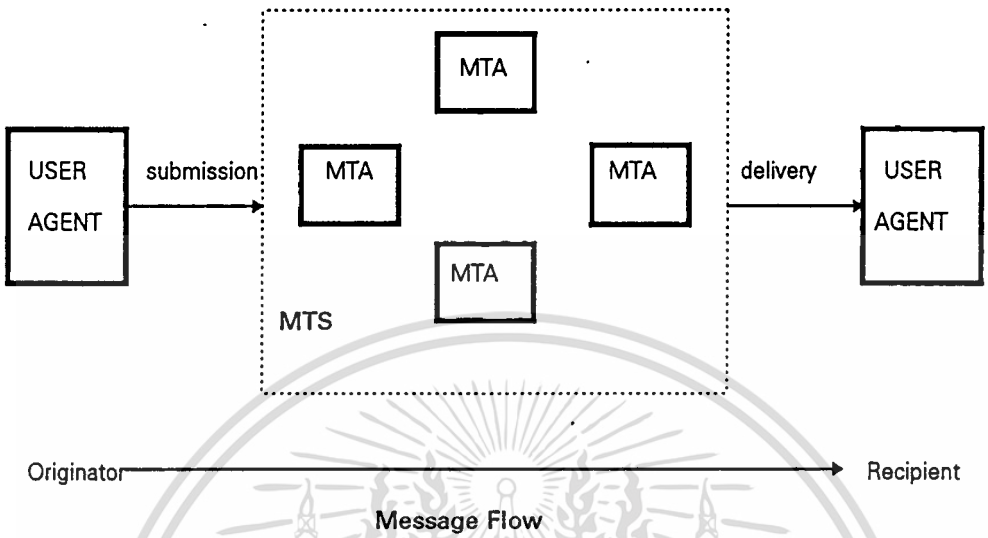
2. **แบ็คเอนด์** (Back End) **หรือ ทรานสปอร์ท เอเจนท์** (Transport Agent - TA) เป็นส่วนที่สำคัญที่สุดของระบบจดหมายอิเลคทรอนิคส์ เป็นสิ่งที่ใช้วัดความสามารถและความเชื่อถือได้ของระบบจดหมายอิเลคทรอนิคส์ มีหน้าที่รับ ส่งข่าวสารและจัดการระบบจดหมายอิเลคทรอนิคส์ ส่วนนี้ผู้ใช้จะมองไม่เห็นโดยตรง บางครั้งอาจจะเรียกว่า แมสเสจ ทรานสปอร์ท เอเจนท์ (Message Transport Agent) ซึ่งสามารถแบ่งออกเป็นส่วนย่อยได้หลายส่วน เช่น

- Transport Service ส่วนบริการส่งจดหมายอิเลคทรอนิคส์

- Directory Service ส่วนบริการไดเร็คทอรีเพื่อให้ระบบจดหมายอิเลคทรอนิคส์สามารถปรับเลขหมายปลายทางของระบบจดหมายอิเลคทรอนิคส์อื่นให้มีเลขที่อยู่เดียวกัน

- Message Store ส่วนเก็บข่าวสารของจดหมายอิเลคทรอนิคส์

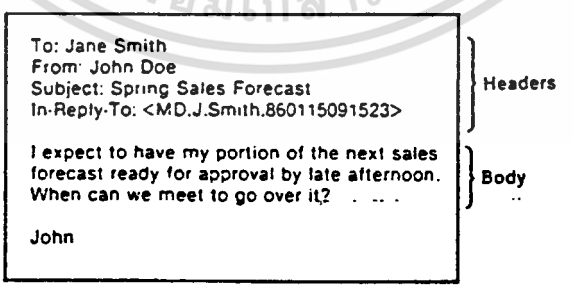
โดยบางครั้งต้องใช้งานหลายๆ ทรานสปอร์ท เอเจนท์ จึงจะสามารถส่งข่าวสารจากผู้ส่งไปยังผู้รับได้ เรียกระบบที่รวมหลายๆ ทรานสปอร์ท เอเจนท์เข้าด้วยกันนี้ว่า แมสเสจ ทรานสเฟอร์ ซิสเต็ม (Message Transfer System-MTS)



รูป 3.2 แสดงส่วนประกอบของระบบจดหมายอิเลคทรอนิคส์

3.3โครงสร้างของข่าวสารข้อมูลในระบบจดหมายอิเลคทรอนิคส์

ข่าวสารจะสามารถส่งไปถึงผู้รับได้จำเป็นต้องมีที่อยู่ของผู้รับ นอกจากนี้ยังต้องประกอบไปด้วยส่วนอื่นๆ อีก ซึ่งแยกออกเป็น ส่วนหัว(Header) ซึ่งประกอบไปด้วยข้อมูลดังในตารางที่ 3.1 และส่วนของข่าวสาร(Body)



รูป 3.3 แสดงโครงสร้างข้อมูลข่าวสารในระบบจดหมายอิเลคทรอนิคส์

ตาราง 3.1 แสดงส่วนหัวของโครงสร้างข้อมูลข่าวสาร

	รายละเอียด	ป้อนเข้าโดย
To:	ผู้รับจดหมายอิเลคทรอนิคส์	ผู้ส่ง
From:	ชื่อผู้ส่ง	ระบบ
Subject:	หัวข้อของข่าวสาร	ผู้ส่ง
Time:	เวลาขณะที่ทำการส่ง	ระบบ
Date:	วันขณะที่ทำการสร้าง	ระบบ

3.4) กลไกการส่งจดหมายอิเลคทรอนิคส์

กลไกการส่งจดหมายอิเลคทรอนิคส์ในแต่ละบริษัทที่ผลิตระบบจดหมายอิเลคทรอนิคส์จะมีส่วนที่แตกต่างกันออกไป จึงต้องมีการกำหนดมาตรฐานที่จะคอยกำหนดลักษณะการส่งจดหมายอิเลคทรอนิคส์เพื่อให้ระบบจดหมายอิเลคทรอนิคส์ที่แตกต่างกันสามารถรับส่งข้อความจากระบบจดหมายอิเลคทรอนิคส์อื่นได้ เช่น MHS และ X.400

MHS (Message Handling Service) เป็นระบบเปลี่ยนโปรโตคอลที่ใช้ในการติดต่อสื่อสารของระบบจดหมายอิเลคทรอนิคส์ บริษัทโนเวลล์ซึ่งเป็นผู้พัฒนาระบบจัดการเครือข่ายคอมพิวเตอร์แบบท้องถิ่นที่ชื่อเน็ตแวร์ ได้นำมาตรฐานแบบ MHS มาใช้กับตัวเน็ตแวร์ เนื่องจากตัวระบบจัดการแบบเน็ตแวร์นั้นเป็นมาตรฐานของระบบจัดการเครือข่ายแบบท้องถิ่นแบบที่ใช้ไมโครคอมพิวเตอร์แบบส่วนบุคคลเป็นสถานีผู้ใช้ ทำให้ MHS กลายเป็นมาตรฐานของการติดต่อสื่อสารจดหมายอิเลคทรอนิคส์บนระบบเครือข่ายแบบท้องถิ่นไปด้วย ถ้าต้องการใช้ MHS ในการส่งข่าวสารจากโหนดหนึ่งในเครือข่ายไปยังอีกโหนดที่อยู่บนอีกเครือข่ายหนึ่ง จะต้องมีเครื่องคอมพิวเตอร์ของระบบเครือข่ายตัวหนึ่งทำงานโปรแกรมจัดการการเชื่อมต่อของMHS(MHS Connectivity Manager) ซึ่งอาจจะเป็นศูนย์บริการข้อมูลหรือสถานีผู้ใช้เครื่องอื่นๆ ก็ได้ ซึ่งเรียกคอมพิวเตอร์เครื่องนี้ว่าโฮสต์(MHS Host) MHS ต้องการข้อมูลเกี่ยวกับผู้ใช้จดหมายอิเลคทรอนิคส์ทั้งที่อยู่บนระบบและอยู่นอกระบบที่ใช้งานอยู่ และลักษณะการให้บริการเช่น ไทรสาร หรือระบบจดหมายของMCI(MCI Mail) ซึ่งผู้ใช้ต้องกำหนดข้อมูลดังกล่าวให้ MHS ซึ่งหลังจากกระบวนการดังกล่าวก็จะทำให้ระบบจดหมายอิเลคทรอนิคส์ที่ใช้ MHS สามารถติดต่อสื่อสารกับระบบจดหมายอื่นๆ ได้

SMTP (Simple Mail Transfer Protocal) เป็นโปรโตคอลที่ใช้ระหว่างคอมพิวเตอร์เครือข่ายแต่ละสถานีบนระบบจัดการแบบยูนิกซ์ SMTP จัดเป็นโปรโตคอลการรับส่งจดหมายอิเลคทรอ

นิคส์บนโปรโตคอลแบบ TCP/IP (Transmission Control Protocal/Internet Protocal) ซึ่งเป็นมาตรฐานของการเชื่อมต่อเครือข่ายแบบอินเทอร์เนต, ระบบเครือข่ายแบบไกล ทำให้สามารถติดต่อส่งข่าวสารไปได้ทั่วโลก การอ้างหมายเลขที่อยู่ของผู้ส่งและผู้รับสามารถทำได้ 2 วิธี

วิธีแรกเรียกว่า Bang หรือ UUCP เมื่อต้องการส่งจดหมายอิเลคทรอนิคส์ ผู้ใช้จะต้องบอกเส้นทางการส่งทั้งหมดด้วย ซึ่งทำให้การใช้งานยุ่งยากมาก เช่น ถ้าผู้ใช้จดหมายอิเลคทรอนิคส์ที่อยู่ที่ระบบชื่อ elec ต้องการส่งข่าวสารผ่านระบบ tele ไปยังผู้ใช้ที่ชื่อ nick ที่อยู่บนระบบคอมพิวเตอร์ชื่อ comp ผู้ส่งจะต้องกำหนดเลขที่อยู่ปลายทางของจดหมายอิเลคทรอนิคส์เป็น tele/comp/nick ซึ่งหากมีหลายๆ ระบบอยู่บนเส้นทางจะยุ่งยากมากขึ้น

ส่วนวิธีที่สอง เรียกว่า DNS (Domain Name System) หรือ Internet Addressing การทำงานจะเก็บไดเรคทอรีที่อยู่ของผู้รับรวมทั้งเส้นทางการส่งไว้ ผู้ส่งจดหมายอิเลคทรอนิคส์จึงเพียงแต่ต้องรู้ชื่อผู้รับปลายทางและชื่อระบบที่ผู้รับปลายทางอยู่เท่านั้น เช่น nick@crsc.kmitl.ac.th หมายความถึงผู้รับชื่อ nick อยู่บน Host ชื่อ kmitl.ac.th อันหมายถึง kmitl ซึ่งเป็น academic ใน Thailand บนโดเมนย่อย crsc

X.400 เป็นโปรโตคอลของ CCITT ที่กำหนดขึ้นเป็นมาตรฐาน แต่มีจุดอ่อนที่ไม่มีระบบบริการไดเรคทอรี การส่งจดหมายอิเลคทรอนิคส์ด้วย X.400 จำเป็นต้องทราบหมายเลขที่อยู่ของผู้รับก่อน จึงจะสามารถส่งไปได้ จึงต้องใช้ X.500 Directory Stand ทำหน้าที่หาหมายเลขที่อยู่ของปลายทางให้ นอกจากนี้ X.400 ยังมีขนาดใหญ่และมีความสลับซับซ้อนสูง ทำให้การติดตั้งและการดูแลทำได้ยาก อย่างไรก็ตาม X.400 สามารถกำหนดลำดับความสำคัญของข่าวสารได้ เพื่อเป็นตัวบอก MTA ว่าข่าวสารนั้นควรจะถูกส่งเร็วแค่ไหน ข้อความที่ส่งไปสามารถเข้ารหัสได้ รูปแบบของข้อมูลที่ส่งไปได้มีมากมาย ไม่ว่าจะเป็นข้อความ ภาพ เสียง เพราะตามทฤษฎีแล้ว X.400 จะไม่มีข้อจำกัดในเรื่องของขนาดหรือปริมาณของข้อความที่อยู่ในข่าวสารแต่ละชุด

3.5) ระบบรักษาความปลอดภัยของระบบจดหมายอิเลคทรอนิคส์

เพื่อความปลอดภัยของข้อความในจดหมายอิเลคทรอนิคส์จึงมีความจำเป็นอย่างยิ่งที่จะป้องกันการเข้าถึงข้อมูลโดยไม่ได้รับอนุญาตจากผู้รับ จดหมายอิเลคทรอนิคส์แบ่งระดับการรักษาความปลอดภัยได้ ดังต่อไปนี้

1. ระบบรักษาความปลอดภัยของระบบเครือข่ายคอมพิวเตอร์ โดยทั่วไปแล้วระบบเครือข่ายจะมีระบบรักษาความปลอดภัยของตนเอง ก็คือ รหัสผ่านในตอนผู้ใช้ทำการลอกอิน (login) เข้าระบบเครือข่าย เพื่อเป็นการกำจัดและป้องกันผู้ที่ไม่มีสิทธิหรือไม่ได้รับอนุญาตให้ใช้ระบบเครือข่าย โดยมีผู้ควบคุมระบบเครือข่าย (Network Supervisor) เป็นผู้ให้สิทธิและเพิกถอนสิทธินี้

2. รหัสผ่านในการเรียกใช้จดหมายอิเลคทรอนิคส์ ในการเรียกใช้งานโปรแกรมจดหมายอิเลคทรอนิคส์ ผู้ใช้จำเป็นต้องป้อนรหัสผ่านของตนเอง ซึ่งจะเป็นรหัสผ่านคนละส่วนกับรหัสผ่านในการลอกอินเข้าเครือข่าย ระบบที่ดีจะมีการบันทึกความพยายามในการเข้ามาใช้งานเครือข่ายเมื่อป้อนรหัสผ่านที่ผิด และจำกัดจำนวนครั้งเพื่อให้ผู้ควบคุมเครือข่ายรู้ว่ามีผู้ใช้งานที่คนใดมีความเสี่ยงต่อความปลอดภัยในการใช้งานในระบบจดหมายอิเลคทรอนิคส์ และทำการเตือนให้ผู้ใช้ทำการเปลี่ยนรหัสผ่านของตน ในบางระบบที่อนุญาตให้ใช้การเข้าถึงระยะไกล (Remote Access) อาจมีการตรวจสอบหมายเลขโทรศัพท์ของผู้เรียกเพื่อตรวจสอบกับฐานข้อมูลที่มีอยู่เพื่อใช้ในการพิจารณาการเข้าสู่เครือข่ายของผู้เรียก

3. การเข้ารหัสจดหมายอิเลคทรอนิคส์ระบบจะทำการเข้ารหัสจดหมายอิเลคทรอนิคส์โดยอัตโนมัติเพื่อป้องกันการเข้าถึงของโปรแกรมเมอร์ที่มีความรู้เรื่องระบบดีพอ และผู้ควบคุมระบบ นอกจากนี้จดหมายอิเลคทรอนิคส์ที่มีความสำคัญมากๆ ซึ่งต้องการความปลอดภัยสูง ผู้ใช้อาจมีความจำเป็นต้องเข้ารหัสจดหมายอิเลคทรอนิคส์ของตนก่อนส่ง ทางด้านรับก็จะมีเฉพาะผู้ที่สามารถถอดรหัสจดหมายอิเลคทรอนิคส์ได้เท่านั้นที่สามารถเข้าใจข้อความในจดหมายอิเลคทรอนิคส์

ระบบจดหมายอิเลคทรอนิคส์เป็นระบบที่จะต้องทำงานบนระบบเครือข่ายคอมพิวเตอร์ ซึ่งโดยหลักการแล้ว คอมพิวเตอร์แต่ละเครื่องสามารถติดต่อเข้ากับคอมพิวเตอร์เครื่องอื่นๆ ได้ โดยจากการส่งผ่านข้อมูลในรูปแบบต่างๆ เช่น การแลกเปลี่ยนข้อมูลบนจานแม่เหล็ก การเชื่อมต่อด้วยสายเคเบิ้ลโดยใช้โปรแกรมโอนย้ายข้อมูล การเชื่อมต่อด้วยสายเคเบิ้ลและระบบจัดการระบบเครือข่ายคอมพิวเตอร์แบบท้องถิ่น หรือการใช้โทรศัพท์เรียกเข้าหาคอมพิวเตอร์เครื่องอื่นโดยใช้โมเด็มติดต่อผ่านโครงข่ายโทรศัพท์ เป็นต้น

การแลกเปลี่ยนข้อมูลบนจานแม่เหล็กเป็นวิธีง่ายๆ การใช้โปรแกรมโอนย้ายข้อมูลจะสามารถส่งผ่านข้อมูลขนาดใหญ่ได้อย่างรวดเร็ว แต่ในทางปฏิบัติแล้วจะไม่สามารถใช้งานได้อย่างต่อเนื่อง การใช้ระบบโทรคมนาคมจะมีความเหมาะสมสำหรับการเชื่อมต่อระยะไกล แต่จะมีอัตราการส่งผ่านข้อมูลต่ำ ส่วนระบบเครือข่ายคอมพิวเตอร์แบบท้องถิ่นจะมีความยุ่งยากใน

การติดตั้งและการบำรุงรักษา แต่สามารถใช้งานได้อย่างต่อเนื่องและมีอัตราการส่งผ่านข้อมูลที่สูงมาก

3.6) ระบบจดหมายอิเลคทรอนิคส์ในเครือข่ายท้องถิ่น

ระบบจดหมายอิเลคทรอนิคส์ที่ใช้กันในระบบเครือข่ายแบบ LAN นั้นจะประกอบไปด้วย

- โมดูลสำหรับศูนย์บริการข้อมูล (Server Module) เป็นส่วนที่ขึ้นอยู่กับระบบจัดการของศูนย์บริการข้อมูลว่าใช้ระบบจัดการแบบใด โดยทั่วไปจะมีส่วนบริการไดเรคทอรีซึ่งเป็นรายนามที่อยู่ของผู้ใช้ในระบบจดหมายอิเลคทรอนิคส์, E-mail engine ซึ่งเป็นฐานข้อมูลของจดหมายอิเลคทรอนิคส์สำหรับใช้เก็บข่าวสารต่างๆ และส่วนบริการการโอนย้าย สำหรับส่งข่าวสารและไฟล์ที่แนบส่งมาด้วย

- โมดูลสำหรับสถานีผู้ใช้ (Client Module) เป็นส่วนจัดการของแต่ละสถานีผู้ใช้ โดยทั่วไปจะมีส่วนติดต่อกับผู้ใช้ ซึ่งมักจะเป็นแบบเมนูให้ผู้ใช้คอมพิวเตอร์สามารถใช้งานจดหมายอิเลคทรอนิคส์ได้โดยง่าย

การใช้จดหมายอิเลคทรอนิคส์อยู่ภายในระบบเครือข่ายแบบท้องถิ่นวงเดียวกันโดยที่ไม่ได้มีการติดต่อกับระบบเครือข่ายท้องถิ่นวงอื่น จะไม่มีความซับซ้อนอะไรมากนัก กล่าวคือ เมื่อผู้ใช้ต้องการส่งจดหมาย ก็เพียงแต่พิมพ์ข้อความเข้าไปในคอมพิวเตอร์แล้วระบุชื่อผู้รับ จากนั้นก็ส่งจดหมายไป ตัวจดหมายจะถูกเก็บไว้อยู่ในตัวศูนย์บริการข้อมูล ซึ่งอาจทำเป็นตู้รับจดหมายของผู้รับ หรือตู้จดหมายส่วนกลางก็ได้ แล้วแจ้งให้ผู้รับมารับจดหมายไป แต่ในกรณีที่มีการรับส่งอยู่ห่างออกไปคนละเครือข่าย ลักษณะเช่นนี้จะเป็นการใช้จดหมายอิเลคทรอนิคส์ข้ามจังหวัด ข้ามประเทศ หรือที่เรียกว่าโกลบอลเน็ตเวอร์ค(Global Network) ซึ่งมีคุณค่าและความสำคัญมากกว่าการใช้จดหมายอิเลคทรอนิคส์แต่เพียงในกลุ่มเล็กๆมาก ในขณะเดียวกันความซับซ้อนของมันก็จะเพิ่มมากขึ้นด้วย การใช้จดหมายอิเลคทรอนิคส์ในลักษณะนี้ไม่สามารถเก็บข่าวสารทั้งหมดไว้ที่ศูนย์กลางเดียวกันได้ จำเป็นต้องส่งข่าวสารไปเก็บไว้ยังเครื่องของผู้รับโดยตรง หรือนำไปเก็บไว้ที่ศูนย์บริการข้อมูลของอีกเครือข่ายหนึ่งเลย ซึ่งก็เหมือนกับจดหมายที่ส่งระหว่างเมือง โดยเริ่มต้นจากผู้ส่งไปยังไปรษณีย์ท้องถิ่น แล้วต่อไปยังไปรษณีย์ของเมืองผู้รับ จากนั้นจึงค่อยส่งตรงไปยังเมืองผู้รับ

ระบบจดหมายอิเลคทรอนิคส์ที่ใช้กันบนระบบเครือข่ายนั้นจะมีลักษณะการทำงานบนยูสเซอร์ เอเจนท์ คล้ายคลึงกัน เนื่องจากต้องออกแบบมาให้ผู้ใช้สามารถใช้งานได้ง่าย แต่ในส่วนของแมสเสจ ทรานสปอร์ท เอเจนท์นั้นจะแตกต่างกันออกไป เพราะขึ้นอยู่กับระบบเครือข่ายที่ใช้

งานอยู่ ทำให้การใช้งานจดหมายอิเลคทรอนิคส์บนระบบเครือข่ายคอมพิวเตอร์แบบท้องถิ่นนั้นไม่เป็นที่นิยมแพร่หลายเท่ากับระบบเครือข่ายคอมพิวเตอร์ที่ใช้ยูนิกซ์ เนื่องจากสาเหตุหลักคือระบบเครือข่ายคอมพิวเตอร์แบบท้องถิ่นนั้นมีหลักการออกแบบมาเพื่อใช้งานกันในระยะทางใกล้ไม่เกิน 10 กิโลเมตร โดยทั่วไปแล้วจะมีการติดตั้งใช้งานกันภายในบริษัท ซึ่งจะอยู่ในตึก อาคารเดียวกัน โดยไม่สนใจที่จะติดต่อออกไปนอกระบบเครือข่ายมากนัก ในการใช้จดหมายอิเลคทรอนิคส์จึงมักจะใช้งานกันในเฉพาะระบบเครือข่ายคอมพิวเตอร์ที่ใช้งานกันอยู่มากกว่า

ระบบเครือข่ายคอมพิวเตอร์แบบท้องถิ่นที่ใช้งานกันนั้นมีลักษณะโครงสร้างระบบที่แตกต่างกันออกไปทั้งในด้านฮาร์ดแวร์ของระบบและซอฟต์แวร์ที่ใช้ควบคุมดูแลระบบ การพัฒนาระบบจดหมายอิเลคทรอนิคส์จึงมักจะขึ้นกับเทคโนโลยีของระบบเครือข่ายคอมพิวเตอร์นั้นด้วย ทำให้การแลกเปลี่ยนข่าวสารของจดหมายอิเลคทรอนิคส์มักทำกันได้ไม่ง่ายนัก การรับส่งจดหมายอิเลคทรอนิคส์ของสองระบบที่แตกต่างกันจำเป็นต้องมีรูปแบบบางอย่างร่วมกันเพื่อให้สามารถติดต่อกันได้ ข้อมูลที่เดินทางข้ามระบบเครือข่ายจะถูกตัดออกเป็นแพ็กเก็ต โดยมีส่วนต้นหรือส่วนหัวบอกว่าข้อมูลจะเดินทางไปทางไหน และมาจากไหน เสมือนว่าข่าวสารเหล่านี้ถูกบรรจุอยู่ในซองจดหมายอิเลคทรอนิคส์ ถ้าระบบทั้งสองสามารถสร้างและเปิดซองจดหมายเพื่ออ่านข้อมูลภายในได้ แสดงว่าทั้งสองระบบสามารถรับส่งจดหมายอิเลคทรอนิคส์ถึงกันได้ ปัจจุบันมีโปรโตคอลที่เป็นมาตรฐานของการติดต่อสื่อสารของจดหมายอิเลคทรอนิคส์ที่ใช้กันบนระบบเครือข่ายคอมพิวเตอร์แบบท้องถิ่นอยู่ 2 แบบใหญ่ๆ คือ MHS และ CCITT X.400

อย่างไรก็ตาม เนื่องจากโปรโตคอลทั้งสองแบบนั้นเป็นมาตรฐานกลางๆ ซึ่งในระบบจดหมายอิเลคทรอนิคส์จะทำให้ตัวโปรแกรมใหญ่ขึ้นโดยไม่จำเป็น ทำให้ผู้พัฒนาระบบจดหมายอิเลคทรอนิคส์ไม่ค่อยสนใจในโปรโตคอลดังกล่าวนั้นมากนัก แต่มักจะสร้างโปรโตคอลในการติดต่อสื่อสารมาใช้เองมากกว่า จดหมายอิเลคทรอนิคส์ที่มีใช้กันโดยส่วนใหญ่จึงมักจะไม่มีโปรโตคอลดังกล่าวรวมอยู่ในตัวระบบ แต่จะทำการขายเป็นชุดโปรแกรมเพิ่มเติมแยกต่างหาก หากต้องการแลกเปลี่ยนตัวจดหมายอิเลคทรอนิคส์กับระบบอื่นๆ ก็ทำได้โดยติดตั้งชุดโปรแกรมเชื่อมต่อเพิ่มเติมเข้าไปกับระบบจดหมายอิเลคทรอนิคส์

การแลกเปลี่ยนตัวจดหมายอิเลคทรอนิคส์บนระบบเครือข่ายหนึ่งๆ กับระบบเครือข่ายอื่นๆ สามารถทำได้โดย

- ถ้าอยู่บนระบบเครือข่ายคอมพิวเตอร์เดียวกันก็แลกเปลี่ยนจดหมายอิเลคทรอนิคส์ด้วยศูนย์บริการข้อมูลของระบบจดหมาย
- ถ้าอยู่คนละเครือข่ายก็จะใช้การเชื่อมต่อเครือข่ายเข้าช่วย (Internetwork Link)

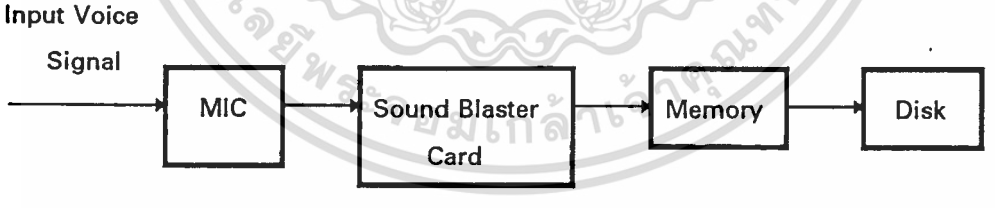
- ถ้าอยู่บนระบบเครือข่ายที่ห่างกันมากๆ หรือ ระบบเครือข่ายคอมพิวเตอร์ระยะไกลก็ติดต่อกันโดยใช้โมเด็ม

บทที่ 4
การโปรแกรมการ์ดเสียง

หลักการพื้นฐานของการ์ดเสียง(sound card)ทั่วไป ก็คือทำการแปลงเสียงที่เป็นสัญญาณอนาลอกให้เป็นข้อมูลดิจิตอลโดยใช้ ADC(Analog to Digital Converter) ในขณะทำการบันทึกเสียง และมีการแปลงจากข้อมูลดิจิตอลที่เก็บไว้ให้ออกเป็นสัญญาณเสียงในขณะอ่านข้อมูลด้วย DAC (Digital to Analog Converter)

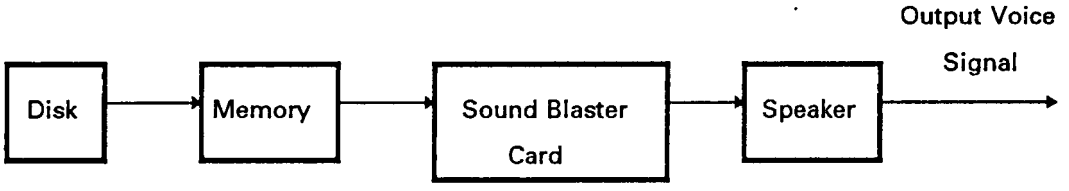
4.1) ลำดับขั้นตอนในการบันทึกและแปลงข้อมูลเป็นเสียง

สำหรับขั้นตอนในการบันทึกเสียง(record) ลงเป็นข้อมูลไบนารีผ่านการ์ดเสียงโดยทั่วไปจะเริ่มจากผ่านสัญญาณเสียงที่ต้องการบันทึกผ่านไมโครโฟนที่ต่ออยู่กับการ์ด เมื่อการ์ดได้รับเสียงก็จะแปลงจากสัญญาณอนาลอกให้เป็นข้อมูลดิจิตอล(ควบคุมโดยฟังก์ชั่นที่มีอยู่ในไดรเวอร์)เก็บลงในหน่วยความจำ เมื่อได้ข้อมูลครบตามที่ต้องการก็จะทำการบันทึกข้อมูลเสียงในหน่วยความจำนั้นลงเป็นไฟล์ในดิสก์ ซึ่งบล็อคไดอะแกรมการทำงานในการบันทึกเสียงสามารถแสดงได้ดังในรูปที่ 4.1



รูป 4.1 บล็อคไดอะแกรมแสดงขั้นตอนคร่าวๆของการบันทึกเสียงโดยใช้การ์ดเสียง

ส่วนการแปลงข้อมูลออกเป็นเสียงหรือการเล่นเสียง(playback) ก็ทำได้ในลักษณะตรงกันข้ามโดยใช้หลักการเดียวกัน คือเมื่อเริ่มมีคำสั่งเล่นเสียง การ์ดเสียงก็จะทำการอ่านข้อมูลจากไฟล์ออกมาเก็บไว้ที่หน่วยความจำก่อน หลังจากนั้นจึงทำการแปลงข้อมูลไบนารีที่อยู่ในหน่วยความจำออกให้เป็นสัญญาณอนาลอกโดยใช้ส่วน DACที่มีอยู่ในการ์ด(โดยใช้ฟังก์ชั่นที่มีอยู่ในไดรเวอร์ควบคุมการทำงานของการ์ดเสียง) ลำดับการทำงานคร่าวๆของการเล่นเสียงแสดงได้ดังรูป 4.2



รูป 4.2 บล็อคไดอะแกรมแสดงขั้นตอนการเล่นเสียงผ่านการ์ดเสียง

ทั้งขั้นตอนการบันทึกและเล่นเสียง จะถูกควบคุมและสั่งการโดยซอฟต์แวร์โดยกระทำผ่านฟังก์ชั่นต่างๆที่มีให้ในไดรเวอร์ของการ์ดเสียง

ในโครงงานนี้การ์ดเสียงที่ใช้คือ ซาวนด์บลาสเตอร์การ์ด (Sound Blaster Card) ของบริษัทครีเอทีพแล็บ (Creative Lab,Inc) ซึ่งจะมีลักษณะของข้อมูลเสียงและฟังก์ชั่นที่ใช้ควบคุมการทำงานของการ์ดดังนี้

4.2) โครงสร้างของข้อมูลเสียง(CT-Voice format)

รูปแบบของข้อมูลดิจิตอลที่แปลงได้จากซาวนด์บลาสเตอร์การ์ดของบริษัทครีเอทีฟแล็บนี้จะมีรูปแบบที่แน่นอน ไฟล์ที่เก็บข้อมูลรูปแบบนี้(Creative Voice file) จะมีส่วนขยายเป็นจุด voc โดยไฟล์จุด voc นี้จะแบ่งได้เป็น 2 ส่วนใหญ่ๆ คือ ส่วนหัว(Header) และส่วนข้อมูล(Data)

ส่วนหัวของไฟล์แบบ CT-Voice (CT-Voice Header Format)

ส่วนหัวนี้เป็นส่วนที่ใช้ระบุได้ว่าไฟล์นั้นเป็นไฟล์ที่มีรูปแบบเป็นของครีเอทีฟหรือไม่ โดยแต่ละไบต์จะมีความหมายดังนี้

- ไบต์ 00h-13h (0-19) : 19ไบต์แรกใช้ไฟล์จุด voc จะเป็นส่วนเก็บข้อความ "Creative Voice File" และอีกหนึ่งไบต์ที่เก็บค่า 1Ah ซึ่งใช้แสดงว่าไฟล์นี้เป็นไฟล์รูปแบบครีเอทีฟจริง

- ไบต์ 14h-15h (20-21) : สองไบต์นี้เก็บค่าออฟเซตของแอดเดรสของข้อมูลที่ได้แซมเปิลมาในลักษณะไบต์ต่ำ/ไบต์สูง ในที่นี้จะเก็บค่า 001Ah เนื่องจากส่วนหัวมีความยาว 1A ไบต์

- ไบต์ 16h-17h (22-23) : สองไบต์นี้เก็บเลขเวอร์ชั่นของรูปแบบ CT-Voice ในลักษณะไบต์ต่ำ/ไบต์สูง เช่นถ้าเป็นเวอร์ชั่น 1.10 ไบต์ 17h นี้จะเก็บค่า 10h และไบต์ 16hจะเก็บค่า 0Ah

- ไบต์ 18h-19h (24-25) : ค่าที่เก็บในสองไบต์นี้เป็นค่าคอมพลีเมนท์ของเลขเวอร์ชั่น(ในสองไบต์ตำแหน่งนี้) บวกกับ 1234h โดยเก็บในลักษณะไบต์ต่ำ/ไบต์สูง ซึ่งมีประโยชน์เพื่อใช้เทียบกับไบต์ 16h-17h เพื่อให้แน่ใจยิ่งขึ้นว่าไฟล์เป็นไฟล์ของครีเอทีฟแน่นอน

ต่อจาก 26 ไบต์นี้จะเป็นส่วนของบล็อคข้อมูล

บล็อคข้อมูล(Data Block)

ข้อมูลเสียงจะมีบล็อคข้อมูลในรูปแบบ CT-Voice นำหน้าข้อมูลจริงอยู่ 8 บล็อค รูปแบบของแต่ละบล็อคจะเหมือนกันหมดยกเว้นบล็อค 0 คือจะเริ่มต้นด้วยหมายเลขระบุบล็อค (0-7) ตามด้วย 3 ไบต์ระบุความยาวของบล็อคและจำนวนข้อมูลที่ใส่เพิ่มเข้ามา สำหรับความยาวนั้นไบต์แรกจะเป็นค่าต่ำสุด ส่วนไบต์ที่สามจะเป็นค่าสูงสุด

สำหรับรายละเอียดของบล็อคข้อมูลทั้งหมดจะมีอยู่ในภาคผนวก ก. โดยบล็อคการทำงานต่างๆเหล่านี้ในไฟล์แบบVOC จะเกิดขึ้นได้จากการใช้ฟังก์ชั่นต่างๆที่มีให้ในไดรเวอร์ CT-VOICE.DRV อยู่แล้วและการโปรแกรมซาวนด์บลาสเตอร์ก็ทำได้โดยใช้ฟังก์ชั่นเหล่านี้เช่นกัน

4.3) ฟังก์ชั่นต่าง ๆ ที่มีให้ในไดรเวอร์ของครีเอทีฟ(CT driver)

ฟังก์ชั่นต่างๆในไดร์ฟเวอร์จะถูกเรียกใช้โดยการกระโดดไปที่แอดเดรสเริ่มต้นของไดรเวอร์ พารามิเตอร์ทั้งหมดที่ต้องผ่านไปยังไดรเวอร์และผลของฟังก์ชั่นจะถูกส่งผ่านรีจิสเตอร์ของโปรเซสเซอร์ ฟังก์ชั่นต่างๆ ในไดรเวอร์ที่จำเป็นต่อการโปรแกรมควบคุมการทำงานของซาวนด์บลาสเตอร์การ์ดมีหลายฟังก์ชั่นเช่น ฟังก์ชั่นเริ่มต้นการทำงานของการ์ด, ฟังก์ชั่นบันทึกเสียง, ฟังก์ชั่นแปลงข้อมูลออกเป็นเสียงหรือเล่นเสียง, ฟังก์ชั่นตั้งค่าหมายเลขพอร์ตและอินเทอร์รัพต์เป็นต้น

โดยในโปรแกรมหลักจะทำการเรียกใช้ฟังก์ชั่นต่างๆเหล่านี้เพื่อควบคุมการทำงานของซาวนด์บลาสเตอร์ให้สอดคล้องกับโปรแกรมควบคุมระบบจดหมายอิเลคทรอนิคส์เพื่อให้สามารถสื่อสารด้วยข้อมูลเสียงหรือที่เรียกว่าจดหมายเสียง(voice mail)ได้

สำหรับรายละเอียดของฟังก์ชั่นต่างๆที่มีให้ในไดรเวอร์ของครีเอทีฟนี้มีแสดงอยู่ในส่วนของภาคผนวก ก.

บทที่ 5
การสื่อสารข้อมูลอนุกรมและโมเด็ม

5.1) การรับส่งข้อมูลอนุกรมด้วย UART

โปรแกรมภาษาระดับสูงจะถูกออกแบบมาเพื่อลดความซับซ้อนให้กับผู้ใช้ในการจัดการเกี่ยวกับไมโครโปรเซสเซอร์ ความคิดนี้ก็สามารถนำมาใช้ได้กับการสื่อสารอนุกรมทั่วไปด้วย ผลที่ได้ก็คือ ไมโครชิพพิเศษที่เก็บรวบรวมรายละเอียดต่างๆเกี่ยวกับการจัดการการสื่อสารแบบอนุกรม ซึ่งรู้จักกันในชื่อ Universal Asynchronous Receiver/Transceiver หรือUART นั่นเอง

หน้าที่ของ UART

การทำงานของUART ในแต่ละสถานะจะถูกควบคุมโดยพารามิเตอร์จากภายนอก โดยทั่วไปการทำงานรับส่งข้อมูลของ UART สามารถสรุปได้ดังนี้

เมื่อทำการส่งข้อมูล

- รับตัวอักษรจากพีซี(เป็นข้อมูลแบบขนาน)
- เปลี่ยนจากข้อมูลแบบขนานให้เป็นแบบอนุกรม
- สร้างเฟรมโดยเพิ่มบิตเริ่มต้น(start),บิตหยุด(stop) และบิตพาริตี้ที่จำเป็นเข้าในข้อมูล
- ส่งข้อมูลแต่ละบิตออกไปที่ส่วนเชื่อมต่อด้วยอัตราเร็วที่ต้องการ (baud rate)
- แจ้งกลับไปว่า พร้อมรับข้อมูลต่อไปแล้ว

เมื่อทำการรับข้อมูล

- รับข้อมูลอนุกรมจากส่วนอินเทอร์เฟส
- แยกเฟรมออกตามโครงสร้างที่กำหนด - บิตเริ่มต้น, ข้อมูล, พาริตี้และบิตหยุด ถ้าไม่เป็นไปตามโครงสร้างที่กำหนดไว้ก็จะรายงานความผิดพลาด (ถือเป็นเฟรมมิงเออร์เรอร์)
- จัดแยกข้อมูลในส่วนพาริตี้ ถ้าไม่ถูกต้องก็จะรายงานพาริตี้เออร์เรอร์
- แปลงจากข้อมูลอนุกรมที่เป็นบิตให้เป็นข้อมูลที่จัดอยู่ได้ในลักษณะอักขระ
- ส่งตัวอักษรเป็นลักษณะข้อมูลขนานไปยังพีซี
- แจ้งกลับว่า การรับตัวอักษรถูกต้อง

ลักษณะและความสามารถของ 8250 UART

เมื่อกล่าวถึง UART ก็มักจะนึกถึงชิพหลักคือ 8250 โดย 8250 จะเป็นตัวเชื่อมต่อระหว่างบัสข้อมูลและบัสแอดเดรสของพีซีกับโมเด็ม หรืออุปกรณ์ภายนอกอื่นๆ ในทางปฏิบัติ 8250 จะช่วยแบ่งเบาภาระการทำงานเกี่ยวกับการสื่อสารข้อมูลอนุกรมให้กับโปรเซสเซอร์ได้ เช่น การเพิ่ม, ตรวจสอบและย้ายเฟรมมิงบิต ก็เป็นหน้าที่ของ UART โดยปรกติพาริตี้บิตจะถูกสร้างขึ้นโดยอัตโนมัติระหว่างการส่งและก็จะมีการตรวจสอบให้โดยอัตโนมัติทางด้านรับเช่นกัน ซึ่งถ้าเกิดตรวจพบความผิดพลาดของพาริตี้ก็จะรายงานให้โปรเซสเซอร์ทราบ 8250ยังมีอินดิเคเตอร์แยกต่างหากเพื่อแสดงสถานะการทำงานขณะส่งและรับ โดยโปรเซสเซอร์จะสามารถอ่านและทำการโปรแกรมอินดิเคเตอร์เหล่านี้เพื่อสร้างอินเทอร์รัพต์ได้ นอกจากนี้ยังมีรีจิสเตอร์ภายในและชิพรีจิสเตอร์สำหรับการส่งและรับเพื่อช่วยจำกัดความจำเป็นที่จะต้องซิงค์กันอย่างแน่นอนระหว่างโปรเซสเซอร์กับข้อมูลอนุกรม

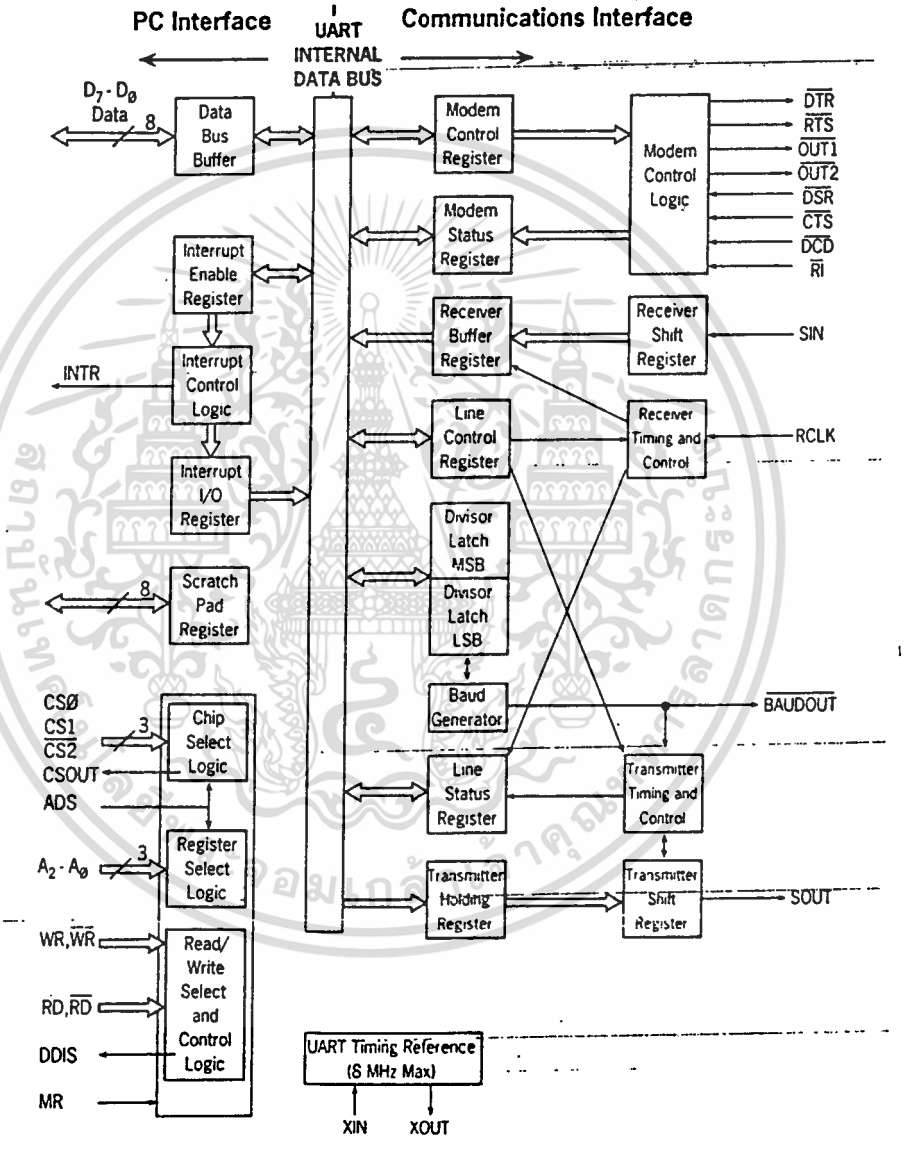
8250ยังมีตัวกำเนิดอัตราเร็วที่โปรแกรมได้(programmable buad generator)ทำหน้าที่หารความถี่ภายนอกซึ่งเป็นความถี่อ้างอิง(โดยมีค่ามากสุดได้ 8 เมกกะเฮิร์ต) โดยตัวหารจะมีค่าได้จาก 1ถึง 65535 สัญญาณที่ได้จะเป็นสัญญาณ 16เท่าของคล็อก เพื่อไปใช้ขับโลจิกภายในของส่วนส่ง และ 16เท่าของคล็อก เดียวกันนี้ยังสามารถใช้เป็นอินพุตสำหรับคล็อคของส่วนรับได้อีกด้วย

นอกจากนี้ 8250 ยังมีโลจิกที่ใช้จัดการกับสัญญาณควบคุมโมเด็ม(modem-control signal) ซึ่งจะต่อเข้ากับระบบอินเทอร์รัพต์ 8250จะดูที่สัญญาณควบคุมโมเด็มนี้และสามารถถูกโปรแกรมเพื่อรายงานการเปลี่ยนแปลงการใช้อินเทอร์รัพต์ได้อีกด้วย

บล็อคไดอะแกรม

เราสามารถแสดงส่วนประกอบภายในของ 8250 ได้เป็นส่วนของ ชุดวงจรเวลา, ส่วนควบคุม และโลจิกที่ใช้ในการเลือก รวมไปถึงรีจิสเตอร์ที่สามารถอ้างถึงได้อีก 11 ตัว ส่วนประกอบย่อยๆเหล่านี้เมื่อนำมารวมกันแล้วจะสามารถใช้เป็นส่วนอินเทอร์เฟสระหว่างโลจิกภายในของพีซีกับการสื่อสารแบบอนุกรมภายนอกได้ ส่วนประกอบย่อยๆเหล่านี้สามารถแทนได้เป็น 2 ส่วนที่ประกอบเข้าด้วยกันอยู่ด้วยบัสข้อมูล เพื่อใช้ในการแลกเปลี่ยนข้อมูลระหว่างกัน ดูได้ดังรูป 5.1

8250/16450 UART block diagram



รูป 5.1 แสดงบล็อคไดอะแกรมของ 8250 UART

รีจิสเตอร์ภายใน

การทำงานของ 8250 ทั้งการส่งและรับข้อมูลทั้งหมดจะถูกควบคุมผ่านรีจิสเตอร์ นั่นคือโปรแกรมเมอร์จะใช้การอ่านและเขียนที่รีจิสเตอร์เหล่านี้เพื่อควบคุมการทำงานทั้งหมดของ 8250 เพื่อให้ได้ผลเป็นการสื่อสารแบบอนุกรมตามที่ต้องการ

รีจิสเตอร์ทุกตัวใน 8250 มีขนาด 8 บิต แต่บางครั้งก็ไม่ได้ใช้ครบทั้ง 8 บิต (บิตที่ไม่ใช้จะมีการมาสค์ไว้ เพื่อไม่ให้ค่าในรีจิสเตอร์ผิดพลาด การใช้งานแต่ละรีจิสเตอร์ก็จะแตกต่างกัน และโปรแกรมเมอร์ต้องคำนึงถึงผลข้างเคียงที่จะเกิดขึ้นจากการอ่านและเขียนรีจิสเตอร์นั้นๆด้วย เช่น อาจทำให้เกิดการรีเซ็ตแฟล็กรายงานความผิดพลาด(error flag) รีจิสเตอร์ต่างๆใน 8250 มีดังนี้

-Tranamitter Holding Register (THR) ใช้สำหรับรับข้อมูลก่อนส่งออกไปยังเอาท์พุทอนุกรม (SOUT) ของ UART

-Receive Buffer Register (RBR) ใช้สำหรับรับข้อมูลจากอินพุตอนุกรม(SIN)ของ UART

-Line Control Register (LCR) ใช้สำหรับกำหนดรูปแบบของข้อมูลอะซิงโครนัสที่จะใช้ในการรับส่งข้อมูลและยังใช้สำหรับกำหนดค่าบอดของ 8250 ได้อีกด้วย

-Line Status Register (LSR) เป็นรีจิสเตอร์แสดงสถานะของการโอนย้ายข้อมูล โดยแต่ละบิตจะแสดงถึงสถานะที่แตกต่างกันไป

-Modem Control Register (MCR) เป็นรีจิสเตอร์ที่ใช้ควบคุมสัญญาณเอาท์พุทที่ส่งจาก 8250 ไปยังโมเด็ม

-Modem Status Register (MSR) ใช้แสดงให้เห็นถึงสภาวะปัจจุบันของโมเด็มและการเปลี่ยนแปลงที่เกิดขึ้นหลังจากการอ่าน MSR ครั้งสุดท้าย

-Divisor Latch Register เป็นรีจิสเตอร์ที่ใช้เก็บค่าตัวหารสำหรับหาค่าบอดที่จะใช้ในการรับส่งข้อมูล

สำหรับรายละเอียดโดยละเอียดของรีจิสเตอร์ภายในของ UART มีแสดงอยู่ในภาคผนวก ข.

5.2) โมเด็มและคำสั่งที่ใช้กับโมเด็ม

เราสามารถแบ่งการทำงานของโมเด็ม(ที่สามารถโปรแกรมได้) ออกเป็น2 โหมดคือ โหมดข้อมูล(data mode) ส่งข้อมูลระหว่าง DTE กับช่องสัญญาณสื่อสาร และโหมดคำสั่ง(command

mode) สำหรับการส่งคำสั่งและผลตอบสนองของคำสั่งนั้น โดยคำสั่งที่ใช้โปรแกรมเพื่อควบคุมการทำงานของโมเด็มนี้รู้จักกันในชื่อชุดคำสั่งเอที(.AT Command set)

ชุดคำสั่งที่ใช้นี้ได้ชื่อว่าชุดคำสั่งเอที ก็เพราะคำสั่งที่จะสั่งให้โมเด็มทำงานจะต้องเริ่มต้นด้วยอักษร 'AT' และโมเด็มจะต้องอยู่ในโหมดคำสั่งจึงจะรู้ว่า AT นี้คือจุดเริ่มต้นของคำสั่ง

ชุดของตัวอักษรที่ส่งให้โมเด็มในโหมดคำสั่งนี้เรียกว่าเป็นบรรทัดคำสั่ง(command line)โดยบรรทัดคำสั่งจะต้องเริ่มต้นด้วยอักษร AT แล้วตามด้วยหนึ่งคำสั่งหรือมากกว่าหรือไม่ม่ก็ได้ และจบบรรทัดด้วย carriage return(CR) รูปแบบของบรรทัดคำสั่งเป็นดังนี้

AT[[command [argument]]...] <CR>,

<u>ชนิดของคำสั่ง(Command Types)</u>

โมเด็มส่วนใหญ่จะถูกเซตให้ใช้ได้กับคำสั่งพื้นฐานทั่วไปสำหรับการสื่อสารแบบอนุกรม เพื่อเป็นการลดจำนวนคำสั่งที่ต้องใช้งานทั้งหมด ซึ่งเราสามารถแบ่งคำสั่งทั้งหมดนี้ได้เป็น 3 กลุ่มย่อยคือ Configuration, Actions และ Dial Command

1. คำสั่งจัดการลักษณะทั่วไป(Configuration Commands)

เป็นคำสั่งที่ใช้เปลี่ยนลักษณะการเชื่อมต่อกับผู้ใช้ของโมเด็มหรือใช้ระบุลักษณะการทำงานของโมเด็ม คำสั่งเหล่านี้จะไม่ทำให้โมเด็มมีการทำงานใดๆเกิดขึ้น แต่อาจจะมีผลให้การกระทำใดๆเกิดขึ้นได้ ตัวอย่างเช่นคำสั่ง Q เป็นคำสั่งที่ใช้ควบคุมว่าจะให้มีการส่งรหัสรับทราบคำสั่งกลับมาหรือไม่ หรือคำสั่ง V ใช้กำหนดว่าจะให้โมเด็มตอบสนองกลับมาในรูปตัวเลขหรือตัวอักษร เป็นต้น

2. คำสั่งสั่งการ(Action Commands)

คำสั่งแบบนี้จะมีผลทันทีต่อสถานะของโมเด็มหรือการเชื่อมต่อกับโมเด็มระยะไกล ตัวอย่างคำสั่งประเภทนี้เช่น คำสั่งH เป็นคำสั่งที่จะทำให้โมเด็มทำหน้าที่เลียนแบบการยกหูขึ้นและวางหูของโทรศัพท์ธรรมดา หรือคำสั่ง O จะทำหน้าที่เปลี่ยนโหมดของโมเด็มจากโหมดคำสั่งให้เป็นโหมดข้อมูล เป็นต้น

3. คำสั่งหมุน(Dial Commands)

คำสั่ง ATD เป็นคำสั่งที่ทำให้โมเด็มทำการหมุนโทรศัพท์ตามข้อมูลที่อยู่ในสตริงคำสั่งหมุน (dial string) เช่น ATD555-1212 เป็นคำสั่งให้โมเด็มยกหูขึ้น, รอเป็นเวลาเท่ากับที่ระบุไว้ในรีจิสเตอร์S6 แล้วจึงหมุนโทรศัพท์หมายเลข 555-1212(ไม่สนใจ -) หลังจากนั้นโมเด็มก็จะพยายามทำการเชื่อมต่อเป็นระยะเวลาเท่ากับที่ระบุไว้ในรีจิสเตอร์S7

สำหรับรายละเอียดโดยละเอียดของคำสั่งต่างๆของโมเด็มมีแสดงไว้อยู่ในภาคผนวก ข.

5.3) โปรโตคอลและมาตรฐานการเชื่อมโยง

การติดต่อสื่อสารระหว่างสองหน่วยที่จะทำให้ประสบผลสมบูรณ์ได้ ก็ต่อเมื่อโปรโตคอลที่ใช้เป็นมาตรฐานเดียวกัน การเชื่อมโยงการสื่อสารจึงจำเป็นต้องอ้างอิงตามมาตรฐานใดมาตรฐานหนึ่ง ในระบบเครือข่ายของบูลเลตินบอร์ด(Bulletin Board) ของแต่ละแห่งที่สร้างกันขึ้น การเชื่อมต่อผ่านเข้าไปด้วยโมเด็ม เราจำเป็นต้องมีซอร์ฟแวร์เพื่อจำลองเทอร์มินอลเพื่อให้การรับส่งข้อมูลเป็นไปตามมาตรฐานที่กำหนด

โปรโตคอลสำหรับการโอนย้ายแฟ้มข้อมูล

แฟ้มข้อมูลถ้ามองในแง่ของการโอนย้ายแฟ้มข้อมูลจะมีอยู่ 2 ชนิดคือ แฟ้มที่เป็นข้อความ (text file) มีเฉพาะรหัสที่เป็นตัวอักษรเท่านั้น ไม่รวมถึงรหัสควบคุมที่นอกเหนือไปจากรหัสควบคุมการแสดงข้อความ เช่น LF, CR, FF, HT, VT ตัวอย่างเช่น แฟ้มที่สร้างขึ้นมาโดยใช้เวิร์ดสตาร์ แฟ้มอีกประเภทหนึ่งเป็นแฟ้มที่มีข้อมูลที่อาจจะมีค่าได้ตั้งแต่ 0-255 เราเรียกแฟ้มประเภทนี้ว่า แฟ้มเลขฐานสอง(Binary file) เช่นแฟ้มคำสั่งที่ลงท้ายด้วย com หรือ exe

การรับส่งแฟ้มประเภทข้อความนั้น ไม่มีปัญหาเรื่องที่จะทำให้ตัวโมเด็มเองหรือเทอร์มินอลตีความหมายผิดไป สำหรับแฟ้มประเภทไบนารีแล้ว รหัสข้อมูลภายในแฟ้มมีโอกาสเป็นไปได้ทุกชนิด หากส่งในลักษณะธรรมดาเหมือนกับข้อความอาจจะทำให้เทอร์มินอลตีความหมายผิดๆ ไปได้ ดังนั้นในการรับส่งแฟ้มข้อมูลไม่ว่าจะเป็นแฟ้มข้อความหรือแฟ้มเลขฐานสอง ทางที่ดีควรจะมีระเบียบกติกาสำหรับการรับส่งที่เป็นที่เข้าใจทั้งฝ่ายส่งและฝ่ายรับ ระเบียบกติกาที่ว่านี้ก็คือโปรโตคอลที่ใช้ในการรับส่งนั่นเอง

นอกจากจะช่วยในการรับส่งข้อมูลให้เป็นแบบแผนแล้ว โปรโตคอลควรจะต้องมีวิธีการในการ

- ตรวจสอบข้อผิดพลาดการรับส่ง
- มีขบวนการที่แน่นอนที่จะต้องปฏิบัติเมื่อมีข้อผิดพลาดในการรับส่ง
- มีวิธีการเรียงลำดับของกลุ่มข้อมูลที่จะส่ง
- มีวิธีการที่จะทำให้ข้อมูลประเภทแฟ้มเลขฐานสองไปถึงฝ่ายรับได้

โปรโตคอลในการรับส่งข้อมูลส่วนมากจะอยู่ในการสื่อสารอนุกรมแบบซิงโครนัส เช่น Bisync, Hdlc, Sdlc แต่เมื่อการรับส่งเป็นแบบอะซิงโครนัสมีปัญหาก็จำเป็นต้องมีโปรโตคอลในการ

รับส่งข้อมูลได้ โปรโตคอลสำหรับการรับส่งข้อมูลผ่านทางโมเด็มที่รู้จักกันดีคือ XMODEM, KERMIT, YMODEM และ ZMODEM เป็นต้น

โปรโตคอล ZMODEM

ZMODEM เป็นโปรโตคอลที่รวบรวมความสามารถของโปรโตคอล XMODEM, YMODEM และ KERMIT ในการโอนย้ายไฟล์ข้อมูล โดยมีลักษณะเป็นโปรโตคอลแบบสตรีมมิง(Streaming Prototcol) ซึ่งจะมีลักษณะพิเศษหลายประการเช่น การแจ้งและแก้ไขการชนกันของข้อมูล(crash recovery) ซึ่งจะช่วยให้ได้ข้อมูลที่ถูกต้องกลับคืนมาหลังเกิดการชน นอกจากนี้ ZMODEM ยังมีการตรวจสอบความผิดพลาดแบบ CRC (CRC Check Error) 16 และ 32 บิต เพื่อตรวจสอบความผิดพลาดขณะส่ง และถ้าเกิดมีสำเนาของไฟล์ที่เหมือนกันอยู่ทั้งสองด้านที่เชื่อมกันอยู่ เมื่อ ZMODEM ตรวจพบสภาพเช่นนี้ก็จะไม่ทำการโอนย้ายไฟล์นั้นให้ และเช่นเดียวกับ YMODEM คือจะมีการสงวนชื่อไฟล์และสิ่งที่สัมพันธ์กับไฟล์ที่โอนย้ายไว้

รูปแบบของการโอนย้ายไฟล์ข้อมูลของ ZMODEM จะเป็นแพคเก็จขนาดใหญ่ซึ่งประกอบด้วย เฟรมส่วนหัว และตามด้วยแพ็คเก็จข้อมูลย่อย เฟรมส่วนหัวจะประกอบด้วยชื่อไฟล์, ขนาดไฟล์ เป็นต้น ส่วนแพ็คเก็จข้อมูลย่อยจะประกอบด้วยข้อมูลย่อยจะประกอบด้วยข้อมูลควบคุม, ข้อมูลของไฟล์ที่โอนย้าย และข้อมูลตรวจสอบความผิดพลาด CRC

บทที่ 6
การทดลองและผลการทดลอง

6.1) การทดลอง

สำหรับโครงงานนี้การทดลองก็คือการพัฒนาโปรแกรมควบคุมระบบจดหมายอิเลคทรอนิคส์ภาษาไทย โดยโปรแกรมที่ได้พัฒนาขึ้นในการทดลองนี้มีส่วนประกอบที่สำคัญต่อการทำงานของโปรแกรม แบ่งได้เป็นหัวข้อดังนี้ คือ

1. ส่วนแสดงผลและเท็กซ์เอดิเตอร์ภาษาไทย
2. ส่วนของระบบจดหมายอิเลคทรอนิคส์
3. ส่วนการแปลงเสียง(play and record)
4. ส่วนเชื่อมต่อด้วยโมเด็ม

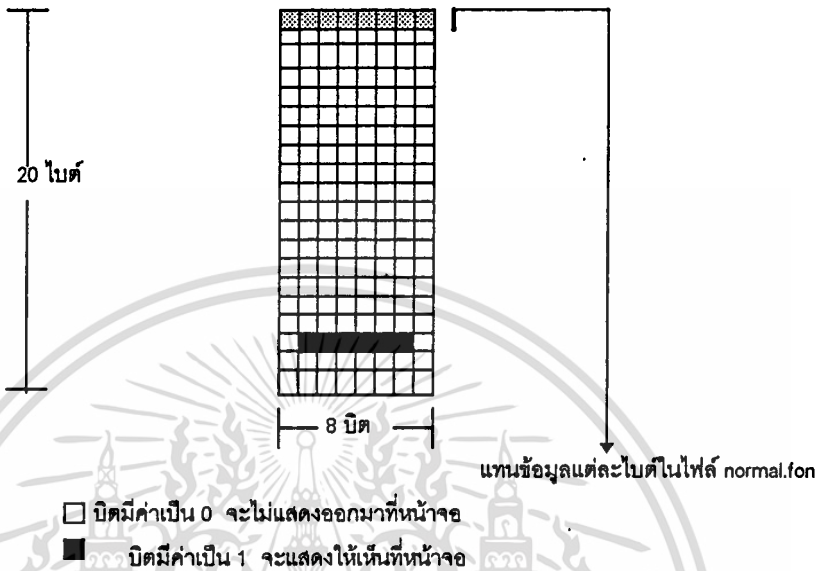
1.ส่วนแสดงผลและเท็กซ์เอดิเตอร์ภาษาไทย

1.1 ส่วนแสดงผลภาษาไทย

เพื่อความสะดวกในการใช้งานของผู้ใช้จึงได้พัฒนาจดหมายอิเลคทรอนิคส์ให้อยู่รูประบบภาษาไทยโดยจะใช้ได้เฉพาะในโปรแกรมของจดหมายอิเลคทรอนิคส์นี้เท่านั้น เมื่อออกจากโปรแกรมกลับเข้าสู่ดอส ระบบภาษาไทยก็จะถูกยกเลิกไป แต่ถ้าใช้วิธีเรียกใช้ไดรเวอร์ภาษาไทยเมื่อออกจากโปรแกรมไดร์เวอร์ภาษาไทยนั้นก็จะยังคงทำงานอยู่ และถ้าผู้ใช้ยกเลิกการทำงานของไดรเวอร์ภาษาไทยนั้นไม่เป็น ระบบภาษาไทยของไดร์เวอร์ก็จะเป็นผลเสียทำให้กลับไปรบกวนการทำงานของโปรแกรมอื่นๆบนดอสได้

เนื่องจากในรหัสแอสกี (ASCII) ไม่มีตัวอักษรภาษาไทยอยู่ การทำงานของระบบภาษาไทยทั่วไปจึงไม่สามารถทำงานในเท็กซ์โหมดได้ จึงจำเป็นต้องทำงานในกราฟฟิคโหมด โดยเซตให้ทำงานที่ความละเอียดของกราฟฟิคเป็น 640x480 โดยที่แต่ละตัวอักษรจะมีขนาด 8x20 (พิกเซล) ดังนั้นในหนึ่งหน้าจอจะบรรจุตัวอักษรได้ 24 แถว (480/20) แถวละ 80 ตัวอักษร (640/8) นั่นคือหนึ่งหน้าจอแสดงตัวอักษรได้ 80x24 ตัว อักษรแต่ละตัวจะถูกโหลดมาจากไฟล์ normal.fon ซึ่งเป็นไฟล์เก็บฟอนต์ตัวอักษรของเวิร์ดจุฬา (CW) ตัวอักษรแต่ละตัวในไฟล์นี้จะมีขนาด 20 ไบต์ กล่าวคือความละเอียดของแต่ละตัวอักษรจะเท่ากับ 8x20 บิต (1ไบต์เท่ากับ 8 บิต) แต่ละบิตคือแต่ละจุด(พิกเซล) ของกราฟฟิคส์บนจอ โดยถ้าบิตไหนมีค่าเป็น '1' ก็จะแสดงจุดที่ตำแหน่งนั้น

บนจอออกมา ถ้าบิตไหนเป็น '0' ก็จะไม่แสดงจุดที่ตำแหน่งนั้น แสดงลักษณะของหนึ่งตัวอักษรได้ดังรูป 6.1



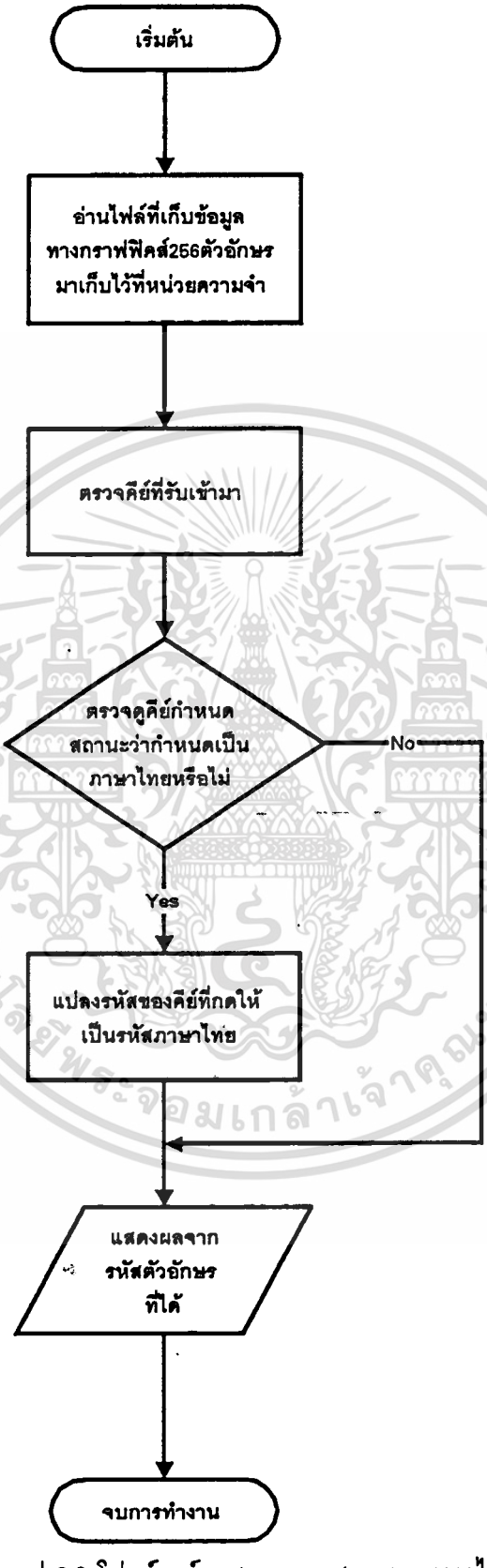
รูป 6.1 แสดงรายละเอียดของแต่ละตัวอักษรในไฟล์ normal.fon

จากรูป 6.1 ตัวอักษรที่แสดงบนหน้าจอ คือ '-' โดยไบต์ที่ 18 ของอักษรตัวนี้จะมีค่าเป็น 0111 1110 หรือ 7Eh ส่วนไบต์อื่นๆ จะมีค่าเป็น 00h หมด

การแสดงตัวอักษรแต่ละตัวสามารถทำได้โดยการอ่านทีละบิตแล้วแสดงออกมาทีละจุดจนเป็น 1 ตัวอักษร (8x20) หรือใช้วิธีแสดงตัวอักษรนั้นไว้ก่อนแล้วจึงเก็บข้อมูลทั้ง 8x20 จุดของทุกตัวอักษรไว้เป็นข้อมูลในลักษณะกราฟฟิคส์ โดยใช้ฟังก์ชั่นทางกราฟฟิคส์ เมื่อต้องการแสดงตัวอักษรนั้นก็จะอ่านข้อมูลที่เก็บไว้ออกมาเพียงครั้งเดียวก็สามารถแสดงตัวอักษรได้หนึ่งตัวอักษร เปรียบเทียบแล้วจะเห็นว่าวิธีหลังเหมาะจะนำมาใช้งานมากกว่าเพราะเสียเวลาในการทำงานน้อยกว่ามากแม้จะต้องเสียหน่วยความจำที่ใช้เก็บตัวอักษรเพิ่มขึ้นก็ตามเนื่องจากการเก็บข้อมูลทางกราฟฟิคของหน้าจอสำหรับโหมดกราฟฟิคที่ความละเอียด 640x480 นั้นต้องใช้หน่วยความจำขนาด 86 ไบต์ในการเก็บหนึ่งตัวอักษร (จากเดิมใช้ 8x20 หรือ 20 ไบต์ในการเก็บหนึ่งตัวอักษร) ส่วนการตอบสนองการกดคีย์ก็จะมีคีย์ที่เป็นตัวกำหนดภาษาที่จะแสดง ถ้ากำหนดเป็นภาษาไทยเมื่อกดคีย์ใดลงไปก็จะแปลงค่าของคีย์นั้นให้ตรงกับตัวอักษรภาษาไทยตัวนั้น

จากที่กล่าวมาสามารถสรุปขั้นตอนในการจัดเตรียมฟอนต์ภาษาไทยและขั้นตอนในการแสดงผลภาษาไทยได้ดังโฟลว์ชาร์ตในรูปที่ 6.2 และ 6.3

รูป 6.2 โฟลว์ชาร์ตแสดงขั้นตอนการเตรียมฟอนต์ภาษาไทย



รูป 6.3 โฟลว์ชาร์ตแสดงการแสดงผลภาษาไทย

1.2 ส่วนสร้างเท็กซ์เอดิเตอร์

สำหรับเท็กซ์เอดิเตอร์หรือส่วนที่ใช้สร้างหรือเปลี่ยนแปลงแก้ไขไฟล์ตัวอักษรนั้น จะทำการเก็บตัวอักษรที่ผู้ใช้คีย์ลงไปทีละบรรทัด โดยจะตรวจตำแหน่งที่เคอร์เซอร์อยู่ เมื่อมีการคีย์ตัวอักษรใดลงไปก็จะเก็บค่าของตัวอักษรนั้นลงในตำแหน่งที่เคอร์เซอร์อยู่และเก็บลงในหน่วยความจำ โดยถ้ามีการพิมพ์แทรกภายในบรรทัดนั้นก็จะเลื่อนค่าของตัวอักษรที่อยู่ที่ตำแหน่งของเคอร์เซอร์ออกไปแล้วแทรกค่าของตัวอักษรที่คีย์ลงไปที่ตำแหน่งที่เคอร์เซอร์อยู่ เมื่อคีย์ค่าลงไปจนสุดบรรทัดแล้วก็จะทำการจองหน่วยความจำให้กับบรรทัดต่อไป แล้วจึงเก็บค่าที่คีย์ลงในหน่วยความจำที่จองให้สำหรับบรรทัดใหม่

เพื่อให้เกิดประสิทธิภาพในการใช้หน่วยความจำเก็บค่าของตัวอักษรในเอดิเตอร์นี้จะใช้การลดจำนวนหน่วยความจำแบบไดนามิก คือเมื่อต้องการใช้จึงค่อยจองหน่วยความจำขนาดเท่ากับที่ต้องการ แต่การใช้หน่วยความจำแบบไดนามิกนี้จะต้องมีความระมัดระวังในการใช้เป็นอย่างมาก คือต้องไม่ใช้เกินจำนวนที่จองเอาไว้ และเมื่อใช้เสร็จแล้วต้องทำการคืนเนื้อที่หน่วยความจำนั้นให้แก่ระบบด้วย การคืนหน่วยความจำที่จองไว้ต้องคืนส่วนที่จองไว้ครั้งหลังสุดก่อนไล่ไปจนถึงหน่วยความจำที่จองไว้ครั้งแรกสุด สำหรับเท็กซ์เอดิเตอร์จะเก็บข้อมูลที่คีย์ทั้งหมดลงบนหน่วยความจำแบบแรมของเครื่อง ไม่ได้แบ่งเก็บลงดิสก์ (หรือที่เรียกว่าการแสวป(swap)ไฟล์) ดังนั้นจึงไม่สามารถสร้างไฟล์ที่มีขนาดใหญ่ได้ สำหรับเอดิเตอร์ที่สร้างขึ้นนี้สามารถสร้างเท็กซ์ไฟล์ได้ขนาดไม่เกิน 800 บรรทัด บรรทัดละไม่เกิน 320 ตัวอักษร ถ้ามากเกินกว่านี้จะไม่สามารถจองหน่วยความจำมาเก็บข้อมูลได้

สำหรับข้อควรระวังอีกอย่างหนึ่งในการใช้ระบบภาษาไทยคือในภาษาไทยมีการวางตัวอักษรอยู่ 3 ระดับคือ - ตัวอักษรระดับปกติ เช่น ก, ข, ค,.....

- ตัวอักษรที่อยู่ด้านบน ได้แก่วรรณยุกต์และสระบางตัว เช่น ่, ้, ิ
- ตัวอักษรที่อยู่ด้านล่าง เช่น สระ ุ, ู เป็นต้น

ดังนั้นการตรวจตำแหน่งของเคอร์เซอร์โดยการนับตัวอักษรจึงต้องไม่นับรวมสระหรือวรรณยุกต์ที่อยู่ด้านบนหรือด้านล่างของตัวอักษรปกติเข้าไปด้วย

ปัญหาอีกประการหนึ่งที่เกิดจากลักษณะของภาษาไทยที่สามารถเขียนอยู่ในตำแหน่งเดียวกันได้ (อยู่ด้านบนหรือด้านล่างได้ของตัวอักษรปกติได้) คือปัญหาการซ้อนทับกันของสีของตัวอักษรที่อยู่แสดงออกหน้าจอที่ตำแหน่งเดียวกัน เช่นคำว่า "รู้" ต้องใช้ค่าทางกราฟฟิคส์ของตัวอักษรสามค่า เพื่อแสดงผลออกทางหน้าจอที่ตำแหน่งเดียวกันโดยใช้ฟังก์ชั่นแสดงผลทาง

กราฟฟิคส์แบบวางซ้อน(XOR-PUT) และเนื่องจากสีของตัวอักษรกับสีของสระและวรรณยุกต์จะเป็นคนละสีกัน เมื่อนำมาวางซ้อนกันจะทำให้เกิดการผสมสีขึ้น ได้สีสุดท้าย(สีของคำว่า รู้)เปลี่ยนไปจากที่ต้องการ ดังนั้นการสร้างคำจึงต้องคำนึงถึงการผสมสีของตัวอักษร,วรรณยุกต์ และสระด้วย

2. ส่วนของระบบจดหมายอิเลคทรอนิคส์

โปรแกรมในส่วนระบบจดหมายอิเลคทรอนิคส์สามารถแบ่งได้เป็น 2 ส่วนใหญ่ๆอีกคือ เป็นส่วนเข้ารหัสและถอดรหัสข้อมูลในจดหมายกับส่วนขั้นตอนการรับส่งจดหมาย

2.1 ส่วนเข้ารหัสและถอดรหัสข้อมูลในจดหมาย

ระบบจดหมายอิเคทรอนิคส์ควรจะมีฟังก์ชั่นการทำงานในส่วนการเข้ารหัสในจดหมายที่ทำการส่ง เพื่อรักษาความปลอดภัยของข้อมูล ระบบจดหมายอิเลคทรอนิคส์?ที่จัดทำนี้ใช้การเข้ารหัสแบบเลื่อนข้อมูลวนรอบ (Shift Around) บล็อกข้อมูลขนาด 16x16 บิตหรือ 32 ไบต์ ใช้รหัสลับ(key) 32 ตัว ขนาดตัวละ 4 บิต ซึ่งแต่ละตัวอ้างข้อมูลได้ 16 รูปแบบ ลำดับการเข้ารหัสเป็นดังนี้

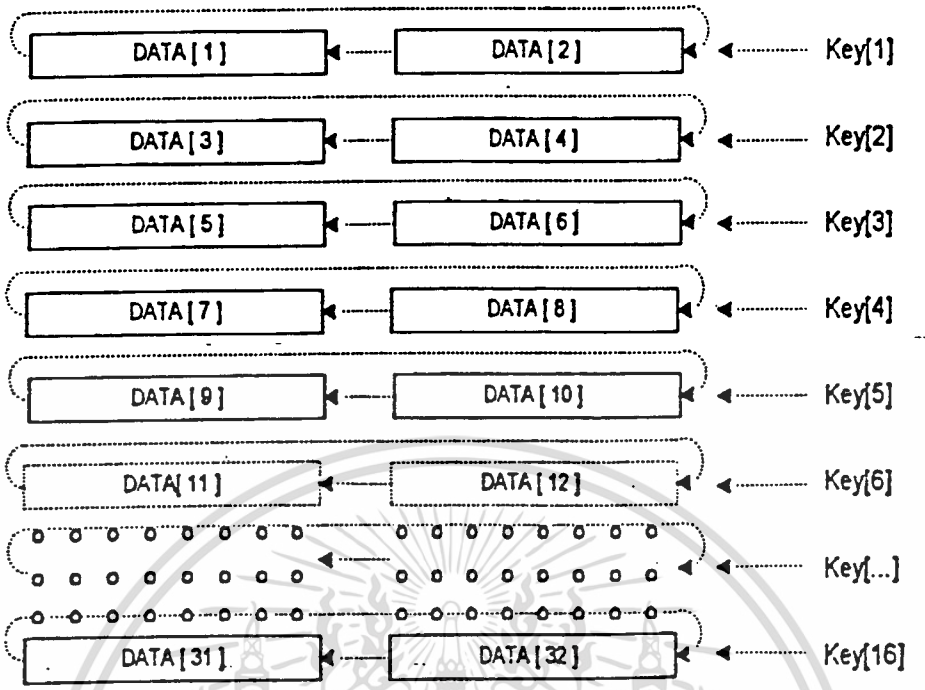
1.จัดบล็อกข้อมูลขนาด 16x16 บิต

2. เลื่อนข้อมูลไปทางซ้ายแบบวนรอบ โดยใช้รหัสลับ 16ตัวแรก

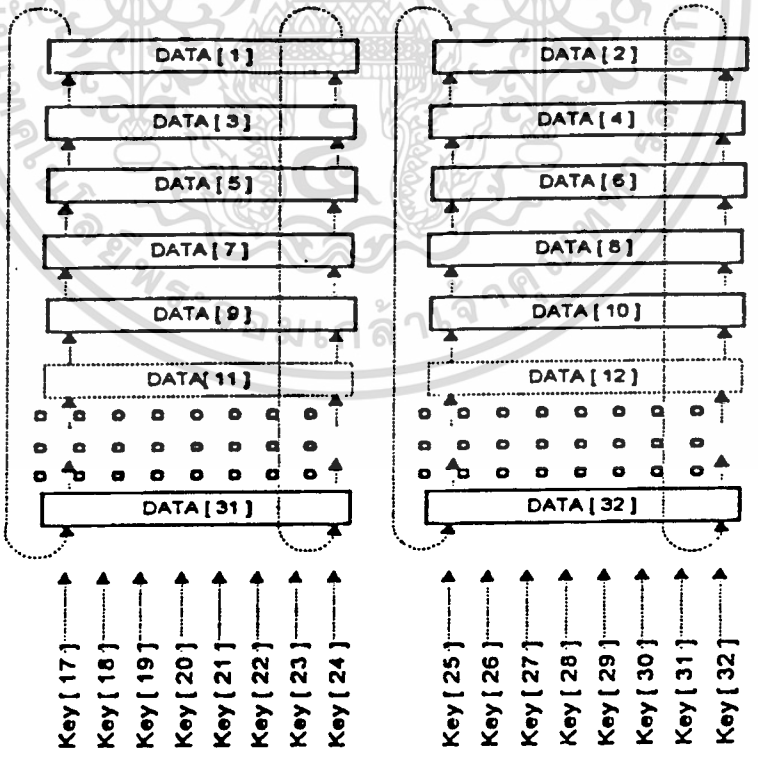
3. เลื่อนข้อมูลไปทางบนแบบวนรอบ โดยใช้รหัสลับ 16 ตัวหลัง

4. เก็บข้อมูลลงแฟ้ม

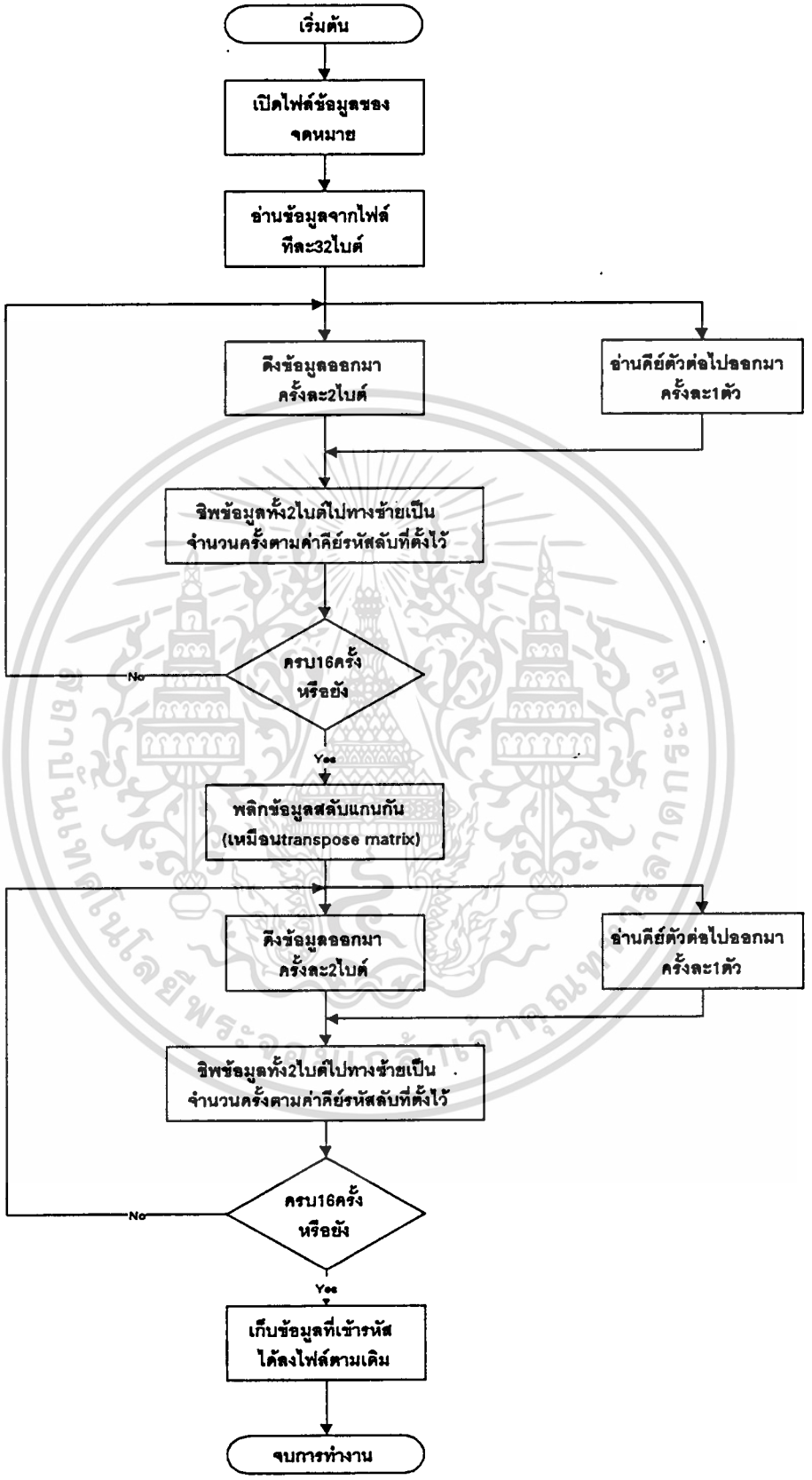
ส่วนการถอดรหัสก็จะกระทำในทางตรงข้ามกับการเข้ารหัส แสดงดังรูป 5.2 โอกาสที่จะถอดรหัสข้อมูลโดยไม่รู้รหัสลับนี้ ต้องใช้ความพยายามในการสุ่มรหัสลับ 32 ตัว ตัวละ16 ครั้งเป็นจำนวนทั้งหมด 16 ยกกำลัง 32 หรือ ประมาณ 3.028E32 ครั้ง(3.028 คูณ 10 ยกกำลัง 32) ถ้าสมมุติว่าการถอดรหัสแต่ละครั้งใช้เวลา 1 ไมโครวินาที ก็จะต้องใช้เวลาทั้งหมด 1.078E25 ปี



รูป 6.4 แสดงการเข้ารหัสบล็อคข้อมูลโดยการเลื่อนไปทางซ้ายแบบวนรอบ



รูป 6.5 แสดงการเข้ารหัสบล็อคข้อมูลโดยการเลื่อนขึ้นด้านบนแบบวนรอบ



รูป 6.6 โฟลว์ชาร์ตแสดงการทำงานของส่วนเข้าและถอดรหัสข้อมูลในจดหมาย

2.2 ขั้นตอนในการรับส่งจดหมาย

ก่อนที่จะกล่าวถึงขั้นตอนในการส่งจดหมายอิเลคทรอนิคส์จะขอกล่าวถึงลักษณะโครงสร้างของระบบเมล์หรือตู้จดหมายในเครือข่ายของโนเวลล์เน็ตแวร์ก่อน คือในระบบของเน็ตแวร์จะมีไดเรคทอรีสำหรับใช้เป็นตู้จดหมาย(mailbox) ของจดหมายในระบบแลนอยู่ไดเรคทอรีหนึ่งชื่อว่า MAIL ซึ่งจะอยู่ในโวลลุ่ม SYS: ของทุกๆศูนย์บริการข้อมูล โดยในไดเรคทอรี MAIL นี้จะประกอบด้วยไดเรคทอรีย่อยที่มีชื่อไดเรคทอรีเป็นหมายเลขออบเจคท์(object ID) ของผู้ใช้ทั้งหมดในศูนย์บริการข้อมูลนั้น ในไดเรคทอรี MAILนี้จะถูกกำหนดแอททริบิวต์หรือสิทธิสำหรับผู้ใช้ทั่วไปให้สร้าง (create) ไฟล์ได้เท่านั้น ไม่สามารถค้นหา (search) ไฟล์หรือไดเรคทอรีในไดเรคทอรีเมล์ได้ ดังนั้นเมื่อใช้คำสั่ง DIR ในไดเรคทอรี MAIL เราจะเห็นเฉพาะไดเรคทอรีที่ตรงกับหมายเลขออบเจคท์ของผู้ใช้เท่านั้น สำหรับไดเรคทอรีย่อยที่อยู่ใน /MAIL ก็เช่นกันคือทุกไดเรคทอรีจะถูกกำหนดแอททริบิวต์ให้ผู้ใช้ทั่วไปสามารถสร้างไฟล์หรือไดเรคทอรีได้เท่านั้น นอกจากผู้ใช้ที่เป็นเจ้าของไดเรคทอรีนี้เท่านั้นที่จะมีสิทธิ์ในไดเรคทอรีของตนทุกอย่าง คือสามารถสร้าง อ่าน เขียน ลบ และค้นหาไฟล์ต่างๆในไดเรคทอรีของตนได้ จากหลักการดังกล่าวสามารถนำมาสร้างเป็นระบบจดหมายอิเลคทรอนิคส์ได้

เริ่มแรกจะทำการแสกนดูบินเดอรี่ของศูนย์บริการข้อมูลทั้งหมดในเครือข่ายที่สามารถเชื่อมต่อได้ รวมทั้งบินเดอรี่ของผู้ใช้ทั้งหมดในศูนย์บริการข้อมูลที่กำลังเชื่อมต่ออยู่ โดยจะเก็บข้อมูลของศูนย์บริการข้อมูลและผู้ใช้เหล่านี้ไว้ เมื่อผู้ใช้ต้องการส่งจดหมายก็จะทำการสร้างส่วนต่างๆของจดหมายทั้งหมดลงในตู้จดหมายของผู้รับ ซึ่งก็คือไดเรคทอรีที่มีชื่อตรงกับหมายเลขออบเจคท์ของผู้รับ(หมายเลขออบเจคท์จะแสดงอยู่ในรูปของเลขฐานสิบหก) จะเห็นว่าผู้ส่งจำเป็นต้องทราบหมายเลขออบเจคท์ของผู้รับซึ่งสามารถดูได้จากบินเดอรี่ แต่เนื่องจากในไดเรคทอรีของผู้รับนั้น ผู้ส่งมีสิทธิเพียงแค่สร้างไฟล์ขึ้นมาใหม่เท่านั้น ไม่สามารถแก้ไขไฟล์เดิมหรือค้นหาไฟล์ได้ ดังนั้นการสร้างส่วนต่างๆของจดหมายลงยังไดเรคทอรีของผู้รับจำเป็นต้องทำให้ไฟล์ของส่วนประกอบของจดหมายมีชื่อไม่ซ้ำกับไฟล์ที่มีอยู่เดิมในไดเรคทอรีของผู้รับ

สำหรับการส่งจดหมายอิเลคทรอนิคส์ในโครงงานนี้ จะทำการแบ่งจดหมายที่ส่งในแต่ละครั้งเป็น 3 ส่วนใหญ่ๆ คือ

1. ส่วนหัวของจดหมาย (Header)
2. ส่วนข้อความในจดหมาย (Content)
3. ส่วนไฟล์ที่แนบไปกับจดหมาย (Attach file)

เพื่อให้การส่งจดหมายสามารถทำงานได้ จำเป็นต้องมีการตั้งชื่อส่วนประกอบต่างๆของจดหมายที่จะสร้างขึ้นที่ไดเรคทอรีเมล์บอกซ์ของผู้รับ โดยใช้หลักการดังนี้

1.ส่วนหัวจดหมาย

ที่ส่วนหัวของจดหมาย จะเก็บรายละเอียดต่างๆที่เกี่ยวกับจดหมายได้แก่ ชื่อผู้ส่ง, เวลาที่ส่ง, หัวเรื่องของจดหมาย, ความยาวจดหมาย, แฟลกแสดงสถานะต่างๆของจดหมาย เช่นถูกอ่านหรือยังหรือเป็นจดหมายใหม่หรือไม่ นอกจากนี้ยังมีรายละเอียดของไฟล์ที่ถูกส่งมาพร้อมจดหมายว่ามีไฟล์ถูกส่งมากับจดหมายหรือไม่ ถ้ามีก็จะมีรายละเอียดว่ามีกี่ไฟล์ ไฟล์อะไรบ้าง รวมทั้งยังมีรายละเอียดบ่งบอกว่าส่วนประกอบต่างๆของจดหมายถูกเก็บอยู่ในไฟล์ใดบ้าง โดยข้อมูลของส่วนหัวของจดหมายจะถูกเก็บลงในไฟล์ที่มีชื่อไฟล์ตามข้อกำหนดดังนี้

ในระบบดอส ชื่อไฟล์ถูกกำหนดให้มีความยาวสูงสุด 8 ตัวอักษร และส่วนขยายมีความยาวสูงสุด 3 ตัวอักษร เพื่อให้ชื่อไฟล์ของจดหมายไม่ซ้ำกับไฟล์ที่มีอยู่เดิมในไดเรคทอรีของผู้รับ จึงให้ ตัวอักษร 4 ตัวแรกของชื่อไฟล์เป็นตัวเลขฐานสิบหกของเวลาที่ทำการส่ง ส่วนตัวอักษรอีก 4 ตัวถัดมาจะใช้ตัวเลขแรนดอมในรูปเลขฐานสิบหกเช่นกัน สำหรับส่วนขยายนั้นสองตัวแรกจะใช้หมายเลขสองตัวแรกของหมายเลขออบเจคท์ของผู้ส่ง ตัวสุดท้ายจะใช้ตัวอักษร 'H' เพื่อระบุว่าเป็นไฟล์ของส่วนหัวของจดหมาย

จะเห็นว่าจากหลักการตั้งชื่อดังกล่าว จะทำให้ชื่อไฟล์ที่สร้างขึ้นใหม่มีโอกาสซ้ำกับชื่อไฟล์ที่มีอยู่เดิมแล้วน้อยมาก เพราะมีโอกาสน้อยมากที่ผู้ส่งคนเดียวกันจะส่งจดหมายมาในเวลาเดียวกันได้ แต่ถ้าบังเอิญเวลาที่ส่งซ้ำกันขึ้นมา ก็ยังมีส่วนที่แตกต่างไปอีกคือเลขแรนดอม 4 ตัวถัดมาสำหรับตัวอักษร 'H' ตัวสุดท้ายที่ส่วนขยายของไฟล์นั้นเป็นตัวที่จะแสดงให้ด้านรับค้นหาว่าเป็นไฟล์ของส่วนหัวของจดหมาย

2. ส่วนข้อความในจดหมาย

เป็นส่วนที่เก็บข้อความในจดหมายที่เข้ารหัสแล้ว หลักการตั้งชื่อไฟล์ของส่วนข้อความในจดหมายที่จะสร้างลงในไดเรคทอรีเมล์บอกซ์ของผู้รับนั้น ใช้หลักการเดียวกับการตั้งชื่อไฟล์ของส่วนหัวจดหมาย แต่ตัวอักษรตัวสุดท้ายจะใช้ตัวอักษร 'M' แทน เพื่อให้ชื่อไฟล์แตกต่างจากชื่อไฟล์ของส่วนหัวจดหมาย และเพื่อระบุว่าเป็นส่วนของข้อความในจดหมาย

3. ส่วนไฟล์ที่แนบไปพร้อมจดหมาย

ในโครงงานนี้เราสามารถส่งไฟล์ได้หลายๆไฟล์ไปพร้อมกับจดหมาย โดยในส่วนนี้จะแยกเก็บแต่ละไฟล์ที่ถูกส่งมาแยกจากกัน หลักการตั้งชื่อไฟล์ของส่วนนี้ก็ใช้หลักการเดียวกับสองส่วนที่ผ่านมา แต่ตัวอักษรตัวสุดท้ายของส่วนขยายจะใช้ค่าของลำดับของไฟล์ทั้งหมดที่แนบมากับ

จดหมายในรูปเลขฐานสิบหกแทนตัวอักษรดังสองส่วนที่กล่าวมาแล้ว(ตัวH และM) เนื่องจากเราใช้เลขฐานสิบหกเพื่อแสดงลำดับไฟล์ที่ตัวอักษรสุดท้ายของชื่อไฟล์ดังนั้นการส่งจดหมายแต่ละครั้งจึงส่งไฟล์ไปพร้อมกับจดหมายได้ครั้งละไม่เกิน 16 ไฟล์

เมื่อสร้างส่วนประกอบต่างๆของจดหมายดังกล่าวได้แล้ว และทำการสร้างไฟล์เหล่านั้นในไดเรคทอรีของผู้รับแล้วก็จะถือว่าเป็นการสิ้นสุดขั้นตอนในการส่ง โดยเมื่อส่งจดหมายไปยังไดเรคทอรีของผู้รับเรียบร้อยแล้วจะทำการส่งข้อความเตือนไปยังผู้รับให้ทราบว่าจดหมายได้ถูกส่งไปยังผู้รับแล้ว

สำหรับการส่งจดหมายไปยังผู้รับที่อยู่ต่างศูนย์บริการข้อมูลจำเป็นต้องมีขั้นตอนเพิ่มเติมดังนี้คือ

1.ตรวจสอบดูว่าศูนย์บริการข้อมูลที่ผู้ใช้ระบุเป็นศูนย์บริการข้อมูลที่เครือข่ายมองเห็นหรือไม่โดยใช้การแสกนดูที่บินเดอรีของศูนย์บริการข้อมูล

2. ถ้ามองเห็นศูนย์บริการข้อมูล ก็จะทำการแอทแทช(attach)ไปยังศูนย์บริการข้อมูลนั้นเพื่อทำการเปิดบินเดอรี ตรวจรายชื่อผู้ใช้ของศูนย์บริการข้อมูลนั้นเพื่อหาผู้รับที่ต้องการ (ใช้การแอทแทชแทนการลอกอิน เนื่องจากการลอกอินไปยังแอคเคานท์ใหม่จะทำการลอกเอาท์จากแอคเคานท์เดิมให้ก่อน แต่การแอทแทชจะเป็นเพียงการเชื่อมต่อไปยังศูนย์บริการข้อมูลนั้นเท่านั้น ไม่ได้ลอกเอาท์ออกจากแอคเคานท์เดิม)

3. ถ้าทำการแอทแทชไปยังศูนย์บริการข้อมูลนั้นได้ ก็จะทำการกำหนดพรีเฟอร์เรนซ์ (preference) ศูนย์บริการข้อมูลไปยังศูนย์บริการข้อมูลที่แอทแทชได้ (การกำหนดพรีเฟอร์เรนซ์ศูนย์บริการข้อมูลไปยังศูนย์บริการข้อมูลใดๆ จะทำให้เราสามารถเปิดและอ่านค่าบินเดอรีของศูนย์บริการข้อมูลนั้นได้ด้วย)

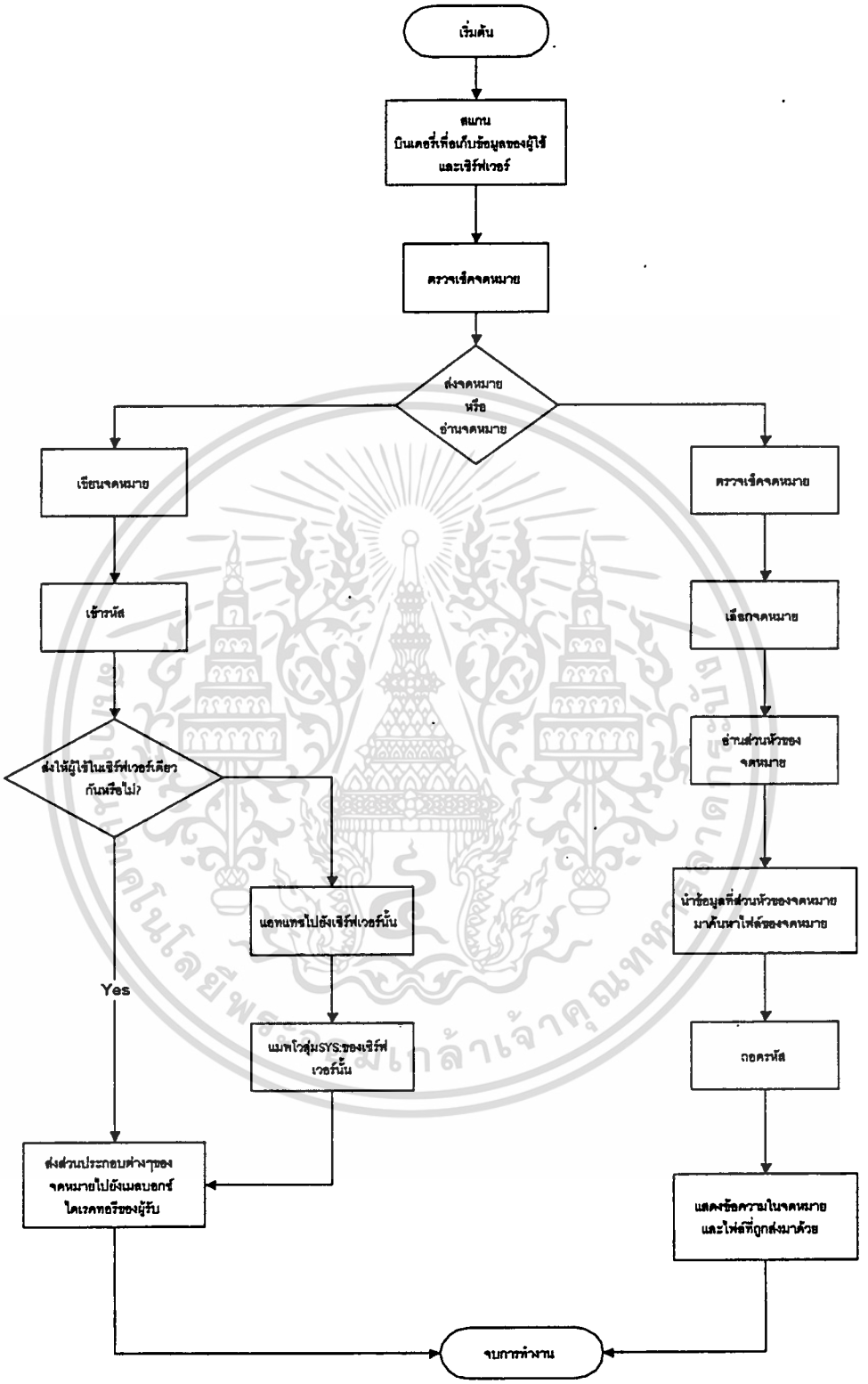
4. ทำการแสกนบินเดอรีของศูนย์บริการข้อมูลตัวนั้น โดยต้องเก็บหรือบันทึกค่าของบินเดอรีของศูนย์บริการข้อมูลเดิมของผู้ใช้ไว้ก่อน การแสกนบินเดอรีของศูนย์บริการข้อมูลของผู้รับจะทำให้ค้นหาได้ว่าผู้รับที่ระบุอยู่ในศูนย์บริการข้อมูลนั้นหรือไม่ ถ้ามีผู้รับอยู่ในศูนย์บริการข้อมูลนั้นก็จะสามารถหาหมายเลขออบเจคท์ของผู้รับนั้น เพื่อนำมาหาเมล์บอกซ์ไดเรคทอรีของผู้รับนั้นได้

5. ทำการแมพ(map) โวลุ่ม SYS: ของศูนย์บริการข้อมูลตัวนั้นเข้ามา เพื่อทำให้สามารถมองเห็นไดเรคทอรีเมล์บอกซ์ของศูนย์บริการข้อมูลรตัวนั้น จากนั้นจึงส่งส่วนต่างๆของจดหมายไปยังเมล์บอกซ์ไดเรคทอรีของผู้รับของศูนย์บริการข้อมูลนั้น พร้อมทั้งส่งข้อความไปเตือนทางด้านผู้รับว่ามีจดหมายถูกส่งไป

สำหรับทางด้านรับสามารถจะตรวจเช็คจดหมายที่ถูกส่งเข้ามาหาตนได้จากการตรวจดูที่ไฟล์ที่เป็นส่วนหัวของจดหมายอิเลคทรอนิคส์ในเมล์บอกซ์ไดเรคทอรีของตน กล่าวคือจะตรวจดูไฟล์ที่มีส่วนขยายตัวสุดท้ายเป็นตัว'H'และมีขนาดของไฟล์เท่ากับขนาดของส่วนหัวของจดหมาย (326 ไบต์) ว่ามีทั้งหมดเท่าไร จำนวนของจดหมายที่ถูกส่งเข้ามาและถูกเก็บในเมล์บอกซ์ทั้งหมด

หลังจากนั้นก็อ่านข้อมูลในส่วนหัวของจดหมายว่าใครส่งมา ส่งมาเมื่อไร มีไฟล์ถูกส่งมาพร้อมกับจดหมายบ้างหรือไม่ จากนั้นจึงทำการอ่านไฟล์ข้อความในจดหมายตามที่ระบุชื่อไฟล์ข้อความไว้ในส่วนหัวของจดหมาย ส่วนข้อความในจดหมายที่ถูกส่งมานั้นจะถูกเข้ารหัสมา จึงต้องทำการถอดรหัสเสียก่อนถึงจะสามารถอ่านข้อความในจดหมายนั้นได้ สำหรับไฟล์ที่ถูกส่งมาพร้อมกับจดหมายนั้น รายชื่อไฟล์ต่างๆ ที่ถูกส่งมาจะถูกเก็บอยู่ในส่วนหัวของจดหมาย ทั้งที่เป็นชื่อไฟล์ที่อยู่ในเมล์บอกซ์ไดเรคทอรีของผู้รับ และชื่อไฟล์เดิมที่ถูกส่งมา เราสามารถใช้ข้อมูลที่เก็บอยู่ในส่วนหัวของจดหมายนี้ทำการค้นหาและแปลงไฟล์ที่ถูกส่งมากับจดหมายเป็นไฟล์ที่เหมือนกับที่ส่งมาได้

สำหรับหลักการอย่างคร่าวๆของขั้นตอนการรับส่งจดหมายอิเลคทรอนิคส์นี้สามารถพิจารณาได้จากโฟลว์ชาร์ตในรูปที่ 6.7



รูป 6.7 โฟลว์ชาร์ตแสดงขั้นตอนการรับส่งจดหมายอิเลคทรอนิคส์

3. ส่วนแปลงเสียง

ส่วนแปลงเสียงนี้เป็นส่วนที่พัฒนาขึ้นมาเพื่อรองรับการรับส่งจดหมายเสียง (voice mail) โดยส่วนประกอบหลักของส่วนแปลงเสียงนี้คือซาวนด์บลาสเตอร์การ์ด การทดลองส่วนนี้ก็คือการเขียนโปรแกรมควบคุมการแปลงข้อมูลเป็นเสียง(playback)และบันทึกเสียง(record) โดยมีหลักการดังนี้

3.1 การแปลงข้อมูลเป็นเสียง

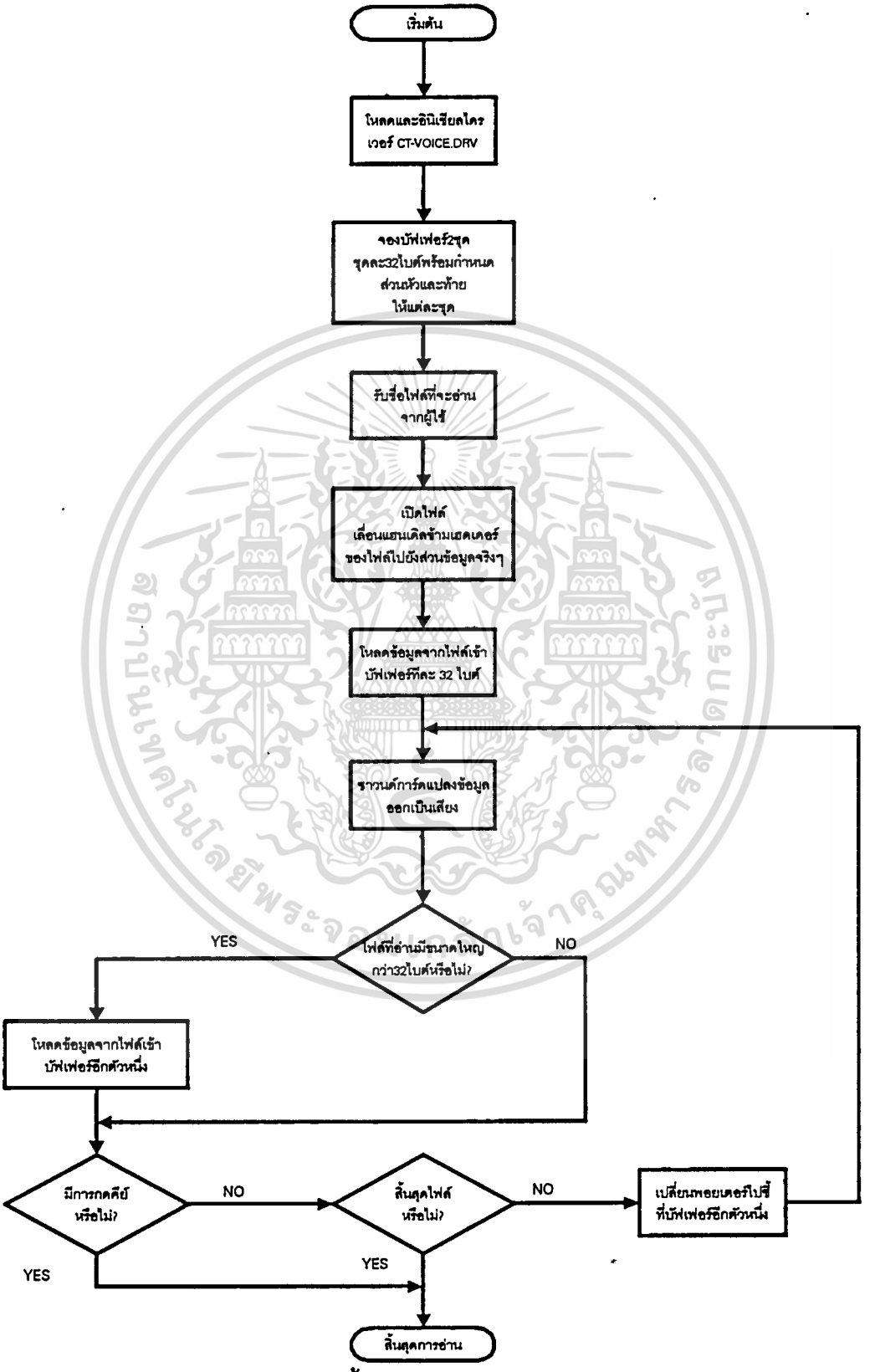
เริ่มต้นจะต้องโหลดไดรเวอร์ของซาวนด์บลาสเตอร์ (CT-VOICE.DRV) เข้ามาในหน่วยความจำและทำการอินิเชียล หลังจากนั้นให้จองบัฟเฟอร์สำหรับอ่านข้อมูลเพื่อแปลงเป็นเสียงไว้2 ชุด ชุดละ32ไบต์พร้อมทั้งกำหนดส่วนหัวและท้ายตามรูปแบบไฟล์ข้อมูลเสียงของบริษัทครีเอทีฟ(CT-format) ให้บัฟเฟอร์แต่ละชุดด้วย แล้วจึงไปรับชื่อไฟล์ที่ต้องการให้แปลงเป็นเสียงจากผู้ใช้ เลื่อนแฮนเดิลชี้ตำแหน่งข้อมูลในไฟล์ข้ามส่วนหัว(Header) ไปที่บล็อคข้อมูลจริง โหลดข้อมูลเข้ามายังบัฟเฟอร์เพื่อให้ซาวนด์บลาสเตอร์เริ่มแปลงข้อมูลออกเป็นเสียง

ขณะที่ซาวนด์บลาสเตอร์กำลังทำการอินเทอร์รัพต์เพื่อดึงข้อมูลจากในบัฟเฟอร์ออกมาแปลงเป็นเสียงอยู่นั้นก็จะมีการอ่านข้อมูลจากไฟล์ในฮาร์ดดิสก์เข้ามายังบัฟเฟอร์อีกตัวหนึ่ง เมื่อซาวนด์บลาสเตอร์อ่านข้อมูลจนหมดบัฟเฟอร์แล้วก็จะมีการเปลี่ยนตำแหน่งพอยเตอร์ให้ไปชี้ที่บัฟเฟอร์อีกชุดหนึ่งแทน และอ่านข้อมูลจากบัฟเฟอร์ชุดใหม่นี้แปลงเป็นเสียงต่อ โดยระหว่างนั้นก็จะมีการอ่านข้อมูลจากไฟล์ในฮาร์ดดิสก์เข้ามายังบัฟเฟอร์อีกตัวหนึ่ง วนทำเช่นนี้ไปเรื่อยๆจนหมดไฟล์(พบ end-of-file) หรือมีการกดคีย์ใดๆเพื่อยกเลิกการเล่นเสียง

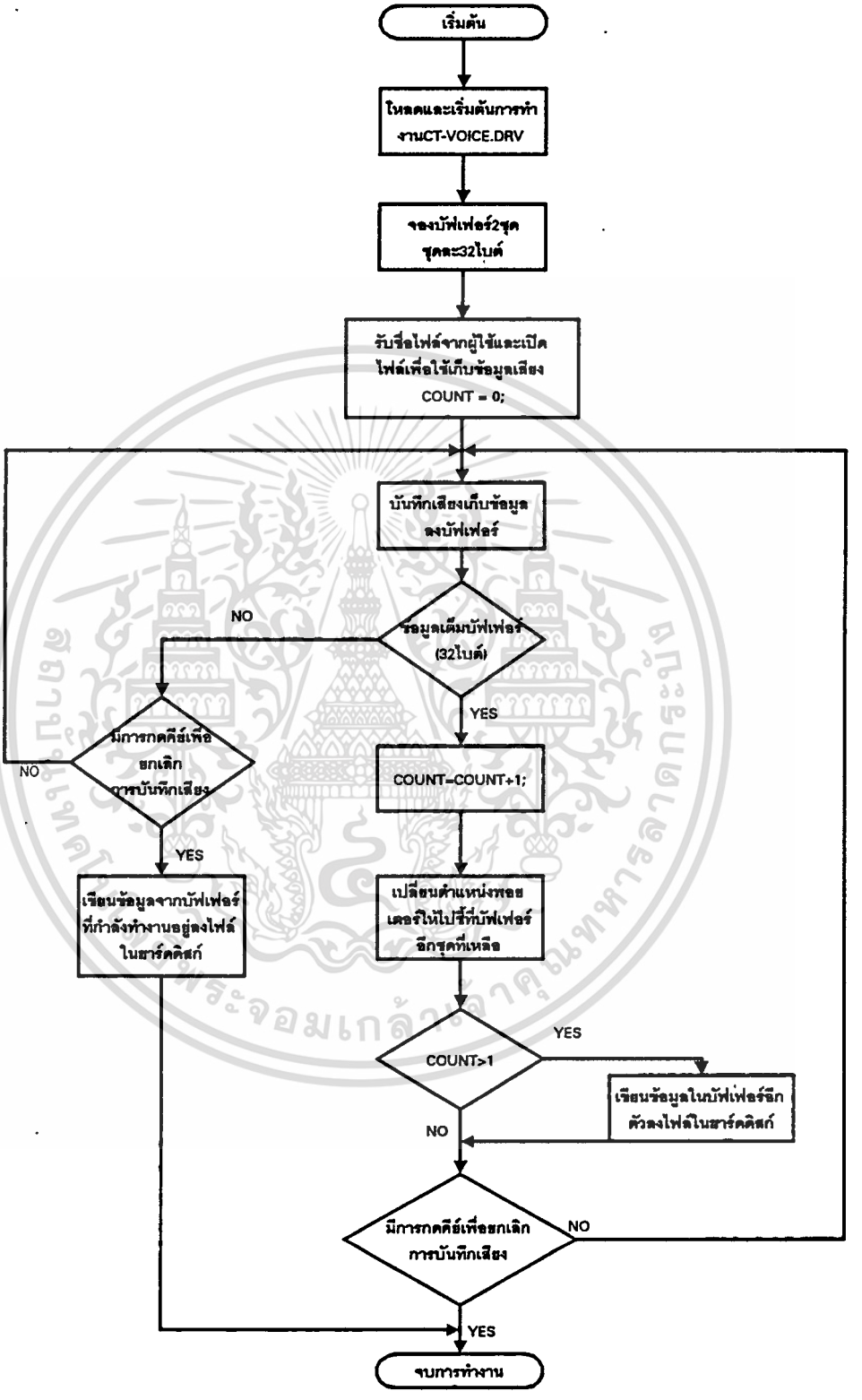
สำหรับหลักการคร่าวๆของการแปลงเสียงสามารถแสดงได้ดังรูป 6.8

3.2 ส่วนบันทึกเสียง

มีหลักการเช่นเดียวกับการแปลงข้อมูลเป็นเสียง คือจะต้องมีการโหลดและอินิเชียล CT-VOICE.DRV ก่อน หลังจากนั้นจองบัฟเฟอร์สำหรับบันทึกเสียงลงหน่วยความจำ 2 ชุด แล้วไปรับชื่อไฟล์ที่จะใช้เก็บข้อมูลเสียงจากผู้ใช้ จากนั้นเริ่มบันทึกเสียงเข้ามาเก็บลงในบัฟเฟอร์ตัวหนึ่งก่อน จนเต็มก็ให้ทำการเปลี่ยนตำแหน่งพอยเตอร์ไปชี้ที่บัฟเฟอร์อีกตัวหนึ่ง ขณะบันทึกเสียงอยู่นี้ก็จะมีการเขียนข้อมูลจากบัฟเฟอร์อีกชุดหนึ่งลงไฟล์ในฮาร์ดดิสก์ เมื่อบัฟเฟอร์เต็มก็จะมีการเปลี่ยนตำแหน่งพอยเตอร์ให้ไปชี้ที่อีกบัฟเฟอร์หนึ่ง(บัฟเฟอร์เดิม) แล้วทำการบันทึกเสียงต่อ (ระหว่างนี้ก็จะมีการเขียนข้อมูลจากอีกบัฟเฟอร์หนึ่งลงไฟล์ในฮาร์ดดิสก์) วนเช่นนี้ไปเรื่อยๆจนมีการกดคีย์ใดๆเพื่อยกเลิกการบันทึกเสียง



รูป 6.8 โฟลว์ชาร์ตแสดงขั้นตอนการแปลงข้อมูลเป็นเสียงของจดหมายเสียง



รูป 6.9 โฟลว์ชาร์ตแสดงขั้นตอนการบันทึกเสียงของจดหมายเสียง

เมื่อมีการกดคีย์ใดๆ ก็จะมีการนำเวลาขณะทำการบันทึกเสียงทั้งหมดมาคำนวณหาเป็นขนาดไฟล์(จำนวนไบต์)ของข้อมูลเสียงที่ทำการบันทึกไป เพื่อนำมาใส่ในส่วนหัวและเพิ่มส่วนท้าย เพื่อให้ได้เป็นไฟล์ข้อมูลเสียงตามรูปแบบไฟล์ข้อมูลของครีเอทีฟ เพื่อให้สามารถนำไปแปลงออกเป็นเสียงได้

สำหรับหลักการคร่าวๆของขั้นตอนการบันทึกเสียงนี้สามารถพิจารณาได้จากโฟลว์ชาร์ตในรูป 6.9

จากนั้นจึงนำทั้งส่วนแปลงเสียงและส่วนบันทึกเสียงมาเชื่อมต่อเข้ากับส่วนรับส่งจดหมายอิเลคทรอนิคส์และส่วนแสดงผลดังที่ได้กล่าวมาแล้ว ก็จะได้เป็นจดหมายเสียงหรือวอยซ์เมล์เพิ่มเข้ามาเป็นอีกส่วนหนึ่งของจดหมายอิเลคทรอนิคส์ภาษาไทย โดยเมื่อผู้ส่งพูดผ่านไมโครโฟนที่ต่ออยู่กับซาวนด์บลาสเตอร์ ซาวนด์บลาสเตอร์ก็จะทำการแปลงเสียงที่ได้เก็บลงเป็นไฟล์ไบนารี(โดยใช้ส่วนบันทึกเสียงที่กล่าวมา) แล้วจึงให้ส่วนส่งจดหมายจัดการส่งไฟล์เสียงนี้ไปยังผู้รับตามที่ต้องการ ทางด้านผู้รับเมื่อรับไฟล์เสียงนี้มาแล้วก็จะเรียกให้ส่วนแปลงเสียงแปลงข้อมูลไบนารีให้ออกเป็นสัญญาณเสียง

4. ส่วนเชื่อมต่อด้วยโมเด็ม

สำหรับการส่งจดหมายอิเลคทรอนิคส์ผ่านทางโมเด็มถือว่าเป็นอีกทางเลือกหนึ่งของการส่งจดหมายอิเลคทรอนิคส์สำหรับผู้ที่ไม่ได้อยู่ในระบบเครือข่ายแลน กล่าวคือ หากต้องการติดต่อกับผู้ใช้อื่นที่อยู่นอกขอบเขตของเครือข่าย ก็สามารถทำได้โดยใช้โมเด็มส่งข้อมูลผ่านทางสายโทรศัพท์ไปยังโมเด็มของผู้รับ โครงสร้างของระบบจดหมายอิเลคทรอนิคส์ผ่านทางโมเด็มโดยทั่วไปนั้น ผู้ใช้โมเด็มส่วนใหญ่จะติดต่อกับศูนย์บูลลิตินบอร์ด(Bulletin Board-BBS) ซึ่งทำหน้าที่เหมือนตู้เก็บจดหมายและคอยรับส่งจดหมาย โดยมีซอฟต์แวร์จัดการทางโมเด็มกับผู้ที่ติดต่อเข้ามา และฟังก์ชั่นต่างๆให้ผู้ติดต่อเข้ามาเลือกใช้

สำหรับในการทดลองนี้ ได้ทำการส่งจดหมายอิเลคทรอนิคส์ผ่านโมเด็มเช่นกัน แต่ไม่ได้ส่งผ่านศูนย์ BBS แต่จะใช้คอมพิวเตอร์เครื่องใดเครื่องหนึ่งในระบบแลนที่มีโมเด็มเป็นตัวคอยจัดการส่งจดหมายไปยังผู้รับตามที่ต้องการ กล่าวคือ โปรแกรมจะกำหนดเครื่องที่มีโมเด็มให้เป็นศูนย์ไปรษณีย์ของระบบจดหมาย ผู้ใช้ในระบบแลนที่ต้องการส่งจดหมายอิเลคทรอนิคส์ไปให้ยังผู้รับที่อยู่นอกระบบ จะต้องส่งจดหมายมาที่ศูนย์ไปรษณีย์ของระบบแลนนี้ก่อน ที่ศูนย์กลางนี้จะมีการกำหนดช่วงเวลาในการตรวจจดหมายในตู้ไปรษณีย์และส่งจดหมายไปยังผู้รับตามเบอร์โทรศัพท์ที่กำหนดไว้ที่จดหมายตามลำดับก่อนหลังที่ผู้ส่งส่งเข้ามาเก็บไว้ที่ตู้จดหมาย สำหรับโปรโตคอลใน

การส่ง ในการทดลองครั้งนี้ใช้โปรโตคอล ZMODEM ซึ่งเป็นโปรโตคอลที่นิยมใช้กันมากที่สุดในปัจจุบัน

ขั้นตอนในการทดลองในการส่งจดหมายอิเลคทรอนิคส์ผ่านโมเด็มมีดังนี้

1. ทำการกำหนดค่าเริ่มต้นการทำงานของพอร์ตอนุกรมของคอมพิวเตอร์ (โดยการเซตที่ 8250 UART) โดยค่าเริ่มต้นที่กำหนดให้กับการติดต่อสื่อสารผ่านพอร์ตอนุกรมได้แก่ จำนวนบิตต่อชุดข้อมูล, จำนวนบิตหยุด, พาริตี้ และอัตราเร็วในการส่งข้อมูล

2. ทำการรีเซตและติดต่อกับโมเด็มผ่านทางพอร์ตอนุกรม โดยโมเด็มจะรับคำสั่งจากคอมพิวเตอร์ในรูปแบบ AT Command ซึ่งจะสามารถสั่งให้โมเด็มทำงานต่างๆรวมทั้งสามารถกำหนดค่าต่างๆของการทำงานของโมเด็มผ่านทางคำสั่งเหล่านี้ได้ด้วย

3. เมื่อกำหนดค่าต่างๆของโมเด็มแล้ว เมื่อต้องการจะส่งก็ใช้คำสั่งรูปแบบ AT นี้สั่งให้โมเด็มหมุนโทรศัพท์ไปยังเบอร์ที่ต้องการได้

4. เมื่อทางด้านรับได้รับสัญญาณโทรศัพท์ก็จะมีการเชื่อมต่อกันระหว่างทั้งสองด้านผ่านสายโทรศัพท์ หลังจากที่มีการเชื่อมต่อกันแล้วข้อมูลที่ถูกส่งให้กับโมเด็มหลังจากนี้จะถือเป็นข้อมูลทั้งหมด

5. เมื่อโมเด็มทั้งสองด้านเชื่อมต่อกันแล้ว จะสามารถส่งข้อมูลไปถึงกันได้ สำหรับการทดลองนี้ทำการส่งข้อมูลโดยใช้โปรโตคอล ZMODEM โดยการทำงานทั้งหมดในการสร้างเฟรมให้อยู่ในโปรโตคอล ZMODEM รวมทั้งการตรวจเช็คค่า CRC นั้น สามารถใช้ฟังก์ชั่นสำเร็จซึ่งรวมอยู่ในไลบรารี่การติดต่อสื่อสาร การทำงานทั้งหมดจึงถูกทำโดยฟังก์ชั่นในไลบรารีนั้น

6. เมื่อสถานีที่เป็นศูนย์กลางจดหมาย ส่งจดหมายฉบับหนึ่งเสร็จแล้วก็จะส่งจดหมายลำดับต่อไปจนครบตามช่วงเวลาที่กำหนดไว้ สำหรับในการทดลองสถานีศูนย์กลางจดหมายจะส่งจดหมายในตู้ไปรษณีย์ตามเวลาที่ตกลงกันไว้ทั้งสองด้านคือด้านส่งและด้านรับ ด้านรับก็จะตอบรับจดหมายที่ช่วงเวลาหนึ่งที่ตกลงกันไว้

สำหรับทางด้านรับนั้น ในการทดลองได้เขียนโปรแกรมรับข้อมูลโดยใช้โปรโตคอล ZMODEM ขึ้น โดยมีขั้นตอนในการกำหนดค่าเริ่มต้นของพอร์ตอนุกรมและโมเด็มเช่นเดียวกันกับทางด้านส่ง และส่งคำสั่งตอบรับอัตโนมัติ (Auto Answer) ให้กับโมเด็ม โดยถ้ามีสัญญาณโทรศัพท์เข้ามาทางด้านรับก็จะตอบรับการเรียกนั้นโดยอัตโนมัติ และถ้าเป็นโมเด็มด้วยกันเรียกเข้ามาทางด้านรับก็จะทำการเชื่อมต่อกับทางด้านรับแล้วจะทำการรอรับข้อมูลในโปรโตคอล ZMODEM เมื่อรับไฟล์ของจดหมายในการเรียกนั้นเสร็จเรียบร้อยแล้ว ด้านส่งจะทำการยกเลิก

การเชื่อมต่อหรือยกหูโทรศัพท์ (hang up) แล้วส่งจดหมายฉบับต่อไป ส่วนทางด้านรับก็จะอยู่ในโหมดตอบรับอัตโนมัติเพื่อรอรับจดหมายต่อไปจนกว่าจะครบตามช่วงเวลาที่กำหนดไว้

6.2) ผลการทดลอง

สำหรับผลการทดลองที่ได้จากการพัฒนาโปรแกรม โดยการรันโปรแกรมจะได้ผลการทำงานดังต่อไปนี้

1. เมื่อเรียกโปรแกรมจดหมายอิเลคทรอนิคส์ภาษาไทย โปรแกรมจะตรวจเช็คดูว่าผู้ใช้เรียกใช้โปรแกรมบนระบบแลนหรือไม่ โดยการอ่านค่าบินเดอรี ถ้าไม่สามารถอ่านค่าบินเดอรีได้ก็แสดงว่าไม่ได้เรียกใช้โปรแกรมบนระบบแลน โปรแกรมจะไม่สามารถทำงานได้ ก็จะยกเลิกการทำงาน แต่ถ้าสามารถอ่านค่าบินเดอรีได้ก็แสดงว่าอยู่บนระบบแลน โปรแกรมก็จะทำงานต่อไป

2. ต่อจากนั้นก็จะเก็บข้อมูลของศูนย์บริการทั้งหมด และผู้ใช้บริการทั้งในศูนยปัจจุบันเพื่อใช้ในการค้นหาผู้รับจดหมาย และเมล์บอกซ์ไดเรคทอรีของผู้รับ

3. โปรแกรมสามารถสร้างและส่งจดหมายได้ โดยมีเอดิเตอร์ภาษาไทย ซึ่งสามารถเขียนหรือสร้างไฟล์ข้อความภาษาไทยและภาษาอังกฤษได้ โดยมีความสามารถในการตัดคำเมื่อสิ้นสุดบรรทัดโดยอัตโนมัติ นอกจากนี้ยังสามารถโหลดไฟล์อื่นที่เป็นไฟล์ข้อความ เช่นไฟล์ข้อความทั่วไปหรือไฟล์ข้อความภาษาไทยของเวิร์ดจุฬามาประกอบเป็นข้อความในจดหมายได้ เมื่อทำการเขียนจดหมายเสร็จเรียบร้อยแล้ว ก็สามารถส่งไปยังผู้รับได้ โดยผู้รับอาจอยู่ในศูนย์บริการเดียวกันหรืออยู่ต่างศูนย์บริการข้อมูลก็ได้ แต่ถ้าอยู่ต่างศูนย์ก็จะต้องเป็นศูนย์บริการข้อมูลที่เชื่อมต่อและมองเห็นได้โดยศูนย์บริการปัจจุบัน สำหรับในระบบเครือข่ายแลนของสถาบันเทคโนโลยีพระจอมเกล้า เจ้าคุณทหารลาดกระบัง มีศูนย์บริการที่เชื่อมต่อกันอยู่ทั้งหมดประมาณ 17-20 ศูนย์ ในบางครั้งที่ไม่สามารถส่งจดหมายไปยังผู้รับที่อยู่ในศูนย์บริการอื่นได้อาจเกิดเนื่องจากศูนย์บริการนั้นไม่เปิดบริการสำหรับผู้ใช้ทั่วไปหรือที่เรียกว่า guest จดหมายจึงไม่สามารถส่งได้โดยตรง ผู้ใช้ต้องทำการเชื่อมต่อไปยังผู้ให้บริการนั้นเอง โดยต้องสามารถเข้าถึงผู้ใช้คนใดคนหนึ่งในศูนย์บริการข้อมูลนั้นได้ จึงจะสามารถส่งจดหมายข้ามศูนย์บริการข้อมูลนั้นได้

4. สำหรับทางด้านรับก็จะมีเสียงเตือนเมื่อมีจดหมายเข้ามา เมื่อผู้รับกดคีย์เพื่อรับจดหมาย ก็จะมีการเก็บข้อมูลของจดหมายนั้นไว้ แล้วแสดงให้ผู้ใช้ดูถ้าผู้ใช้มาตรวจดูจดหมายของตน ผู้ใช้สามารถพิมพ์จดหมายออกทางเครื่องพิมพ์และสามารถตรวจดูว่ามีการส่งไฟล์มาพร้อมกับจดหมายหรือไม่ ถ้ามีผู้ใช้ก็เลือกเก็บไฟล์เหล่านั้นมาไว้ที่ไดเรคทอรีของตนได้

5. ในการส่งจดหมายสามารถที่จะส่งไฟล์ไปพร้อมกับจดหมายได้ โดยสามารถส่งได้มากที่สุด 5 ไฟล์ (จำกัดไว้ 5 ไฟล์เพื่อประหยัดเนื้อที่ของเมล์ไดเรคทอรีและเวลาในการส่ง)

6. สำหรับผู้ใช้ที่มีการ์ดเสียงอยู่ที่เครื่องก็สามารถส่งจดหมายอิเลคทรอนิคส์เป็นเสียงไปยังผู้รับที่มีการ์ดเสียงเหมือนกันได้ โดยจะทำการอัดเสียงและบีบอัดไฟล์ของเสียงนั้นก่อนจึงจะทำ

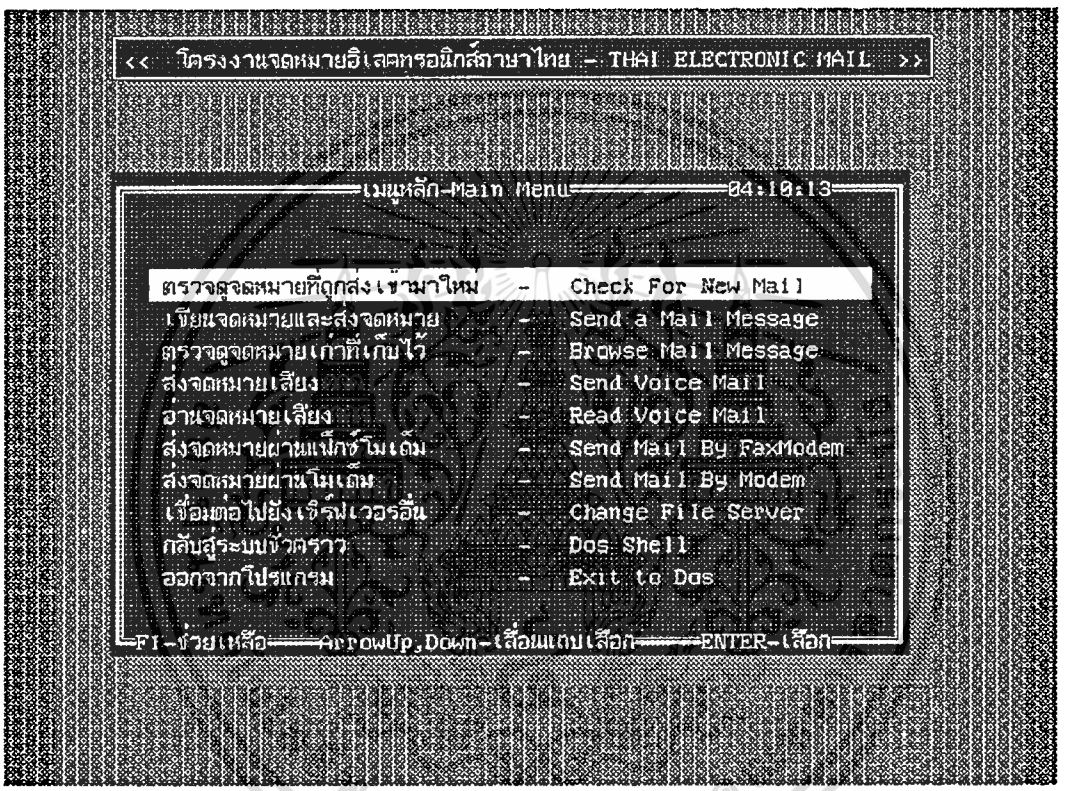
การส่งไปยังด้านผู้รับ ทางด้านผู้รับก็จะตรวจหาจดหมายเสียง ถ้าเครื่องด้านผู้รับมีการ์ดเสียงอยู่ โปรแกรมก็จะทำการขยายจดหมายเสียงที่บีบอัดออกมาก่อน แล้วจึงแปลงข้อมูลนั้นออกเป็นเสียงโดยใช้การ์ดเสียง

เสียงที่อัดและเล่นกลับออกมานั้นจะสามารถเซตค่าอัตราการแซมปลิ้งได้ตามความต้องการเนื่องจากอัตราแซมปลิ้งมีผลโดยตรงต่อจำนวนข้อมูลที่การ์ดเสียงแปลงมาและความชัดเจนของเสียงขณะเล่นกลับ สำหรับการส่งจดหมายเสียงนี้ส่วนใหญ่จะใช้สำหรับเสียงพูดของคนธรรมดาจึงไม่ต้องอัตราแซมปลิ้งสูงมาก เพียงความถี่ 4 กิโลเฮิร์ตซก็พอแล้ว แต่เนื่องจากอัตราแซมปลิ้งต่ำสุดของการ์ดเสียงที่ใช้ในการทดลองนี้อยู่ 5 กิโลเฮิร์ตซ จึงใช้อัตราแซมปลิ้ง 5 กิโลเฮิร์ตซนี้สำหรับการบันทึกและเล่นเสียง ซึ่งข้อมูลที่ได้จากการลองจับเวลาดูขณะบันทึกเสียงทางด้านส่งเป็นดังนี้

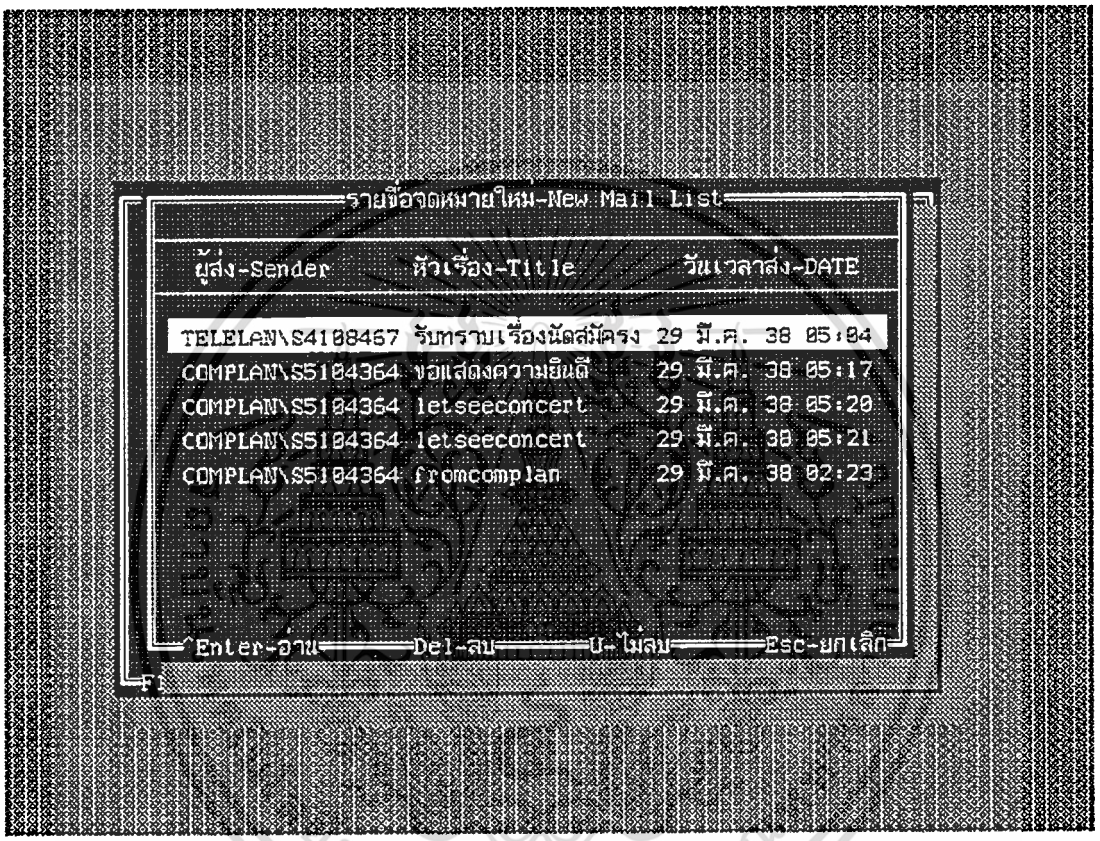
อัตราแซมปลิ้ง (กิโลเฮิร์ตซ)	เวลา (วินาที)	ขนาดข้อมูล (ไบต์)	ผ่านการบีบอัดแล้ว (ไบต์)
5	30	144,931	321
5	60	291,031	461
5	150	738,301	581,745

7. สำหรับในส่วนของการเชื่อมต่อกับโมเด็ม โปรแกรมสามารถส่งจดหมายไปเก็บไว้ที่ศูนย์กลางจดหมายได้โดยผ่านโมเด็ม สถานีที่เป็นศูนย์กลางจดหมายผ่านโมเด็มก็จะมีบริการตรวจดูว่ามีจดหมายที่จะส่งหรือไม่ตามเวลาที่กำหนดไว้ ถ้ามีก็จะส่งตามลำดับของจดหมายที่เข้ามาเก็บที่ศูนย์กลาง หากในช่วงเวลาที่กำหนดยังส่งจดหมายไม่หมดก็จะรอส่งครั้งต่อไป ส่วนทางด้านรับจะมีการกำหนดกับทางศูนย์กลางไว้ว่า จะมีการส่งจดหมายในช่วงเวลาใด เมื่อถึงช่วงเวลานั้นก็จะทำการรอรับจดหมายที่จะถูกส่งมาถึงตน โดยการรับส่งจดหมายจะใช้โปรโตคอลในการรับส่งข้อมูลผ่านโมเด็มที่นิยมใช้กันในปัจจุบันคือ ZMODEM

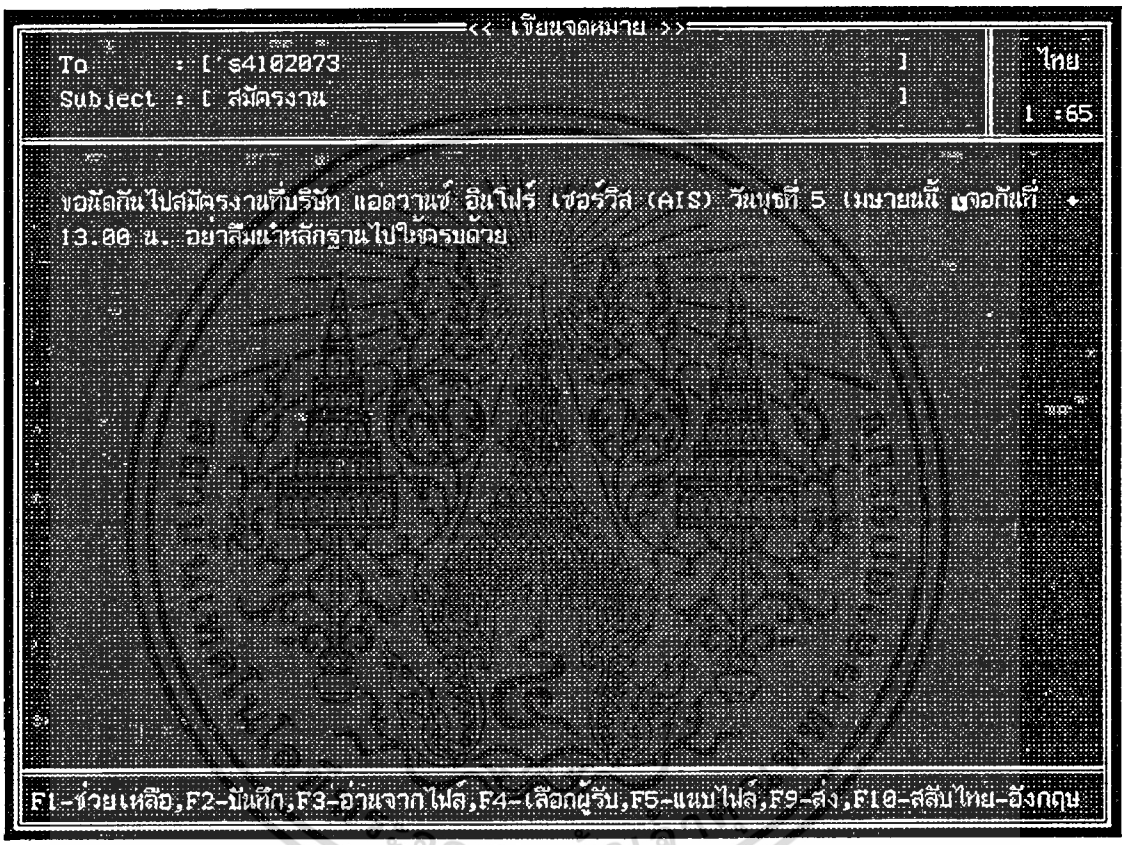
8. นอกจากนี้โปรแกรมยังสามารถส่งจดหมายผ่านแฟกซ์โมเด็ม ซึ่งเป็นผลที่ได้มาจากการเรียกใช้โปรแกรมของโครงงานรับส่งแฟกซ์โมเด็ม เพื่อเป็นการเพิ่มประสิทธิภาพการทำงาน โดยสามารถส่งจดหมายที่สร้างขึ้นโดยเอดิเตอร์ โปรแกรมจดหมายอิเลคทรอนิคส์ภาษาไทยนี้ไปยังเครื่องแฟกซ์ของผู้รับได้



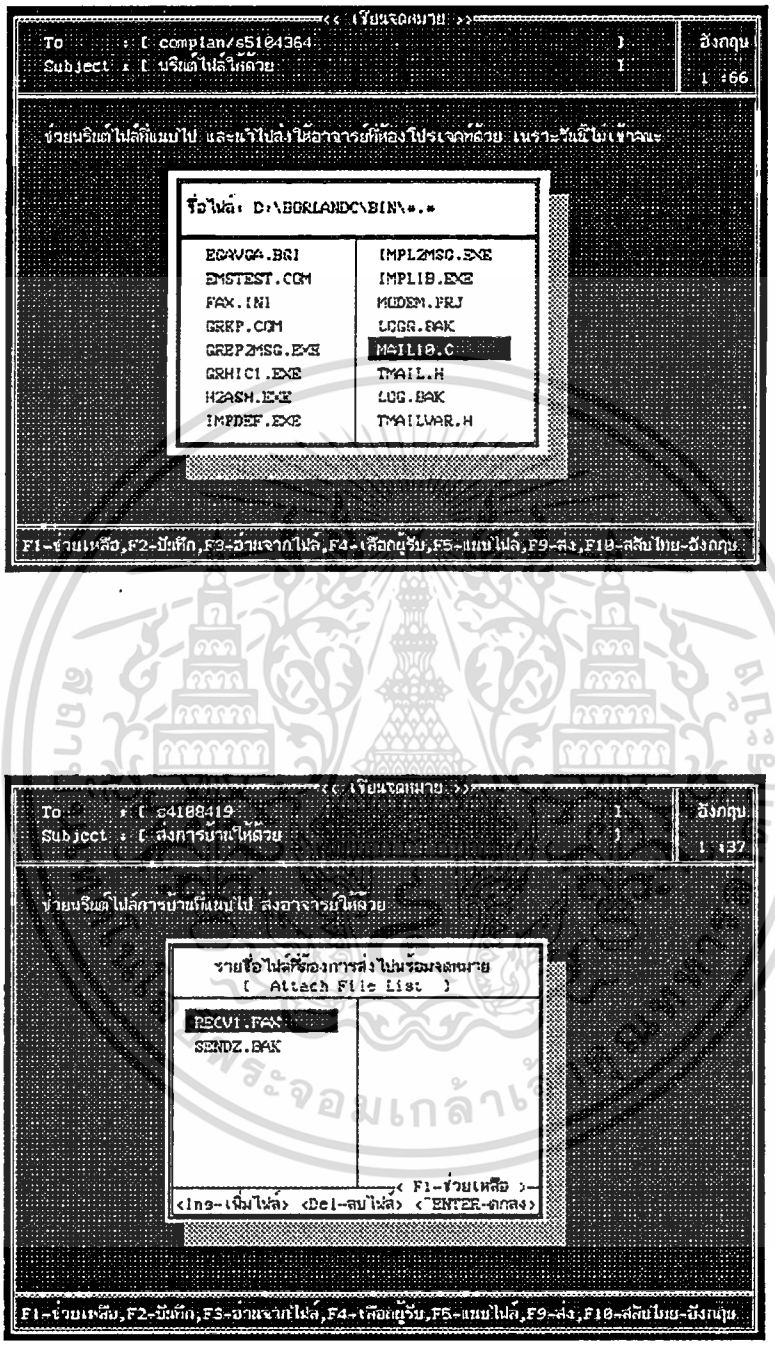
รูป 6.10 แสดงรูปหน้าจอเมนูหลักของระบบจดหมายอิเลคทรอนิคส์



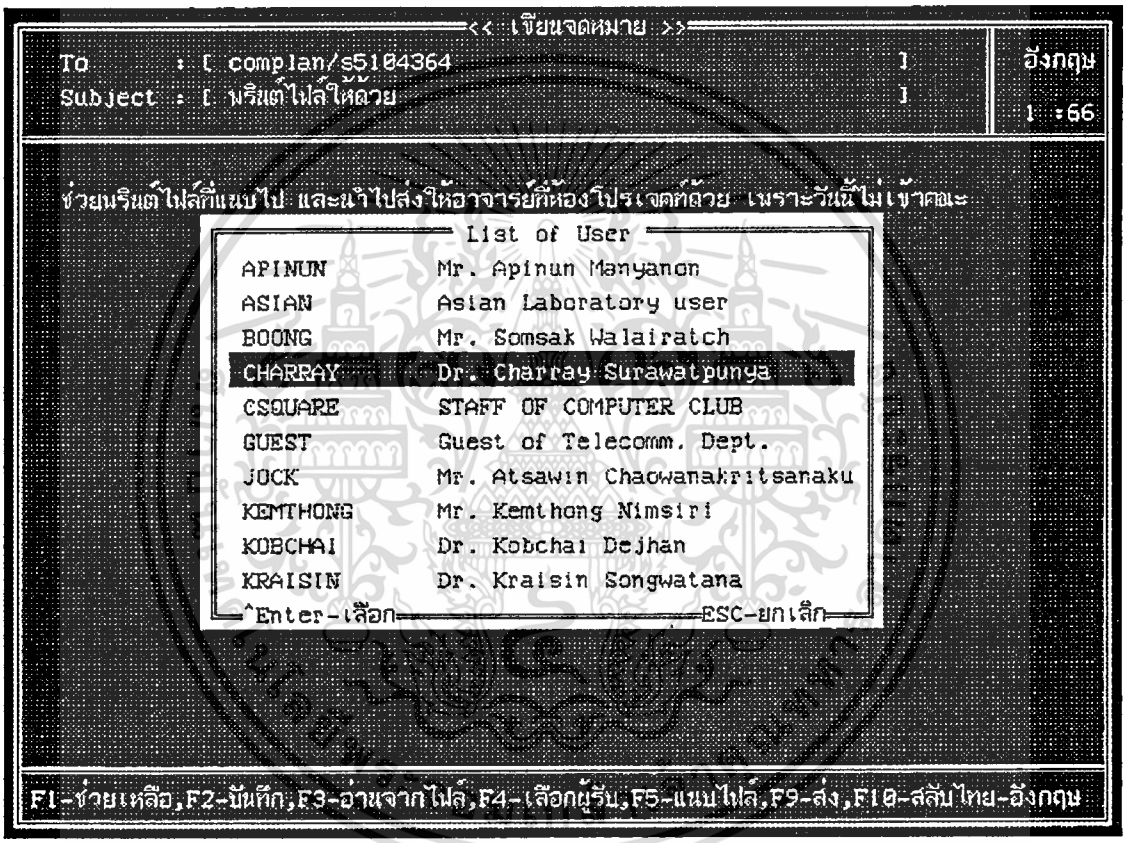
รูป 6.11 หน้าจอแสดงรายชื่อจดหมายใหม่ที่เข้ามา



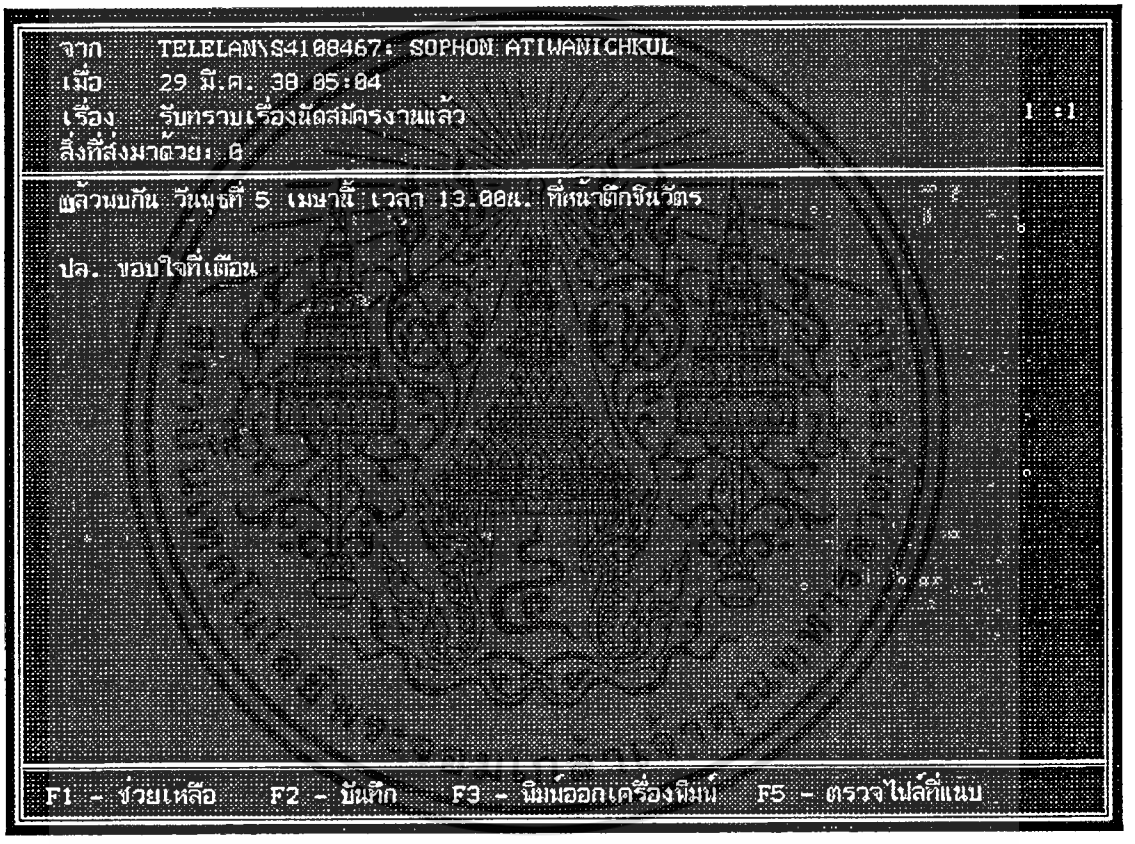
รูป 6.12 แสดงหน้าจอขณะเขียนและส่งจดหมาย



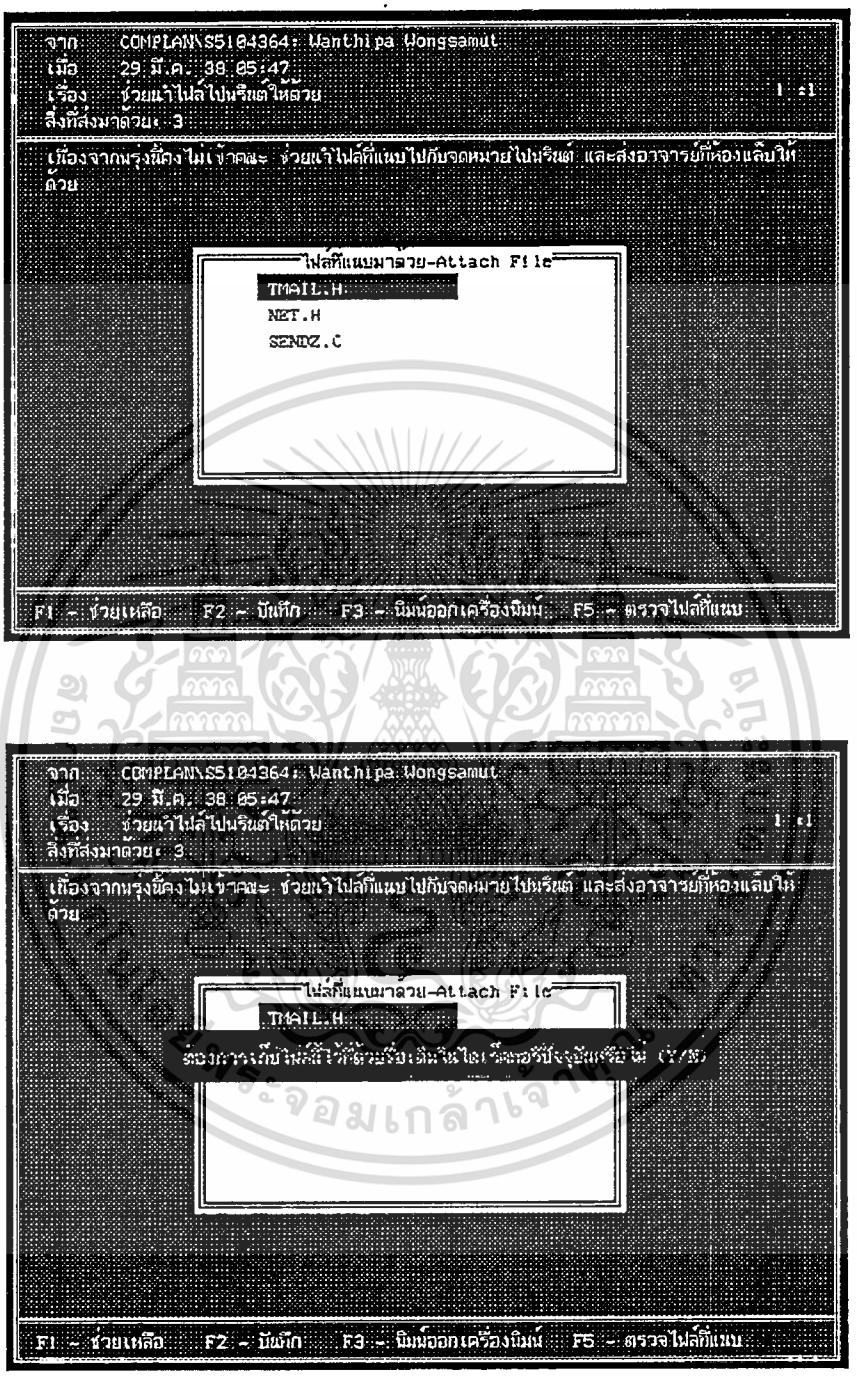
รูป 6.13 แสดงหน้าจอขณะส่งจดหมายพร้อมแนบไฟล์



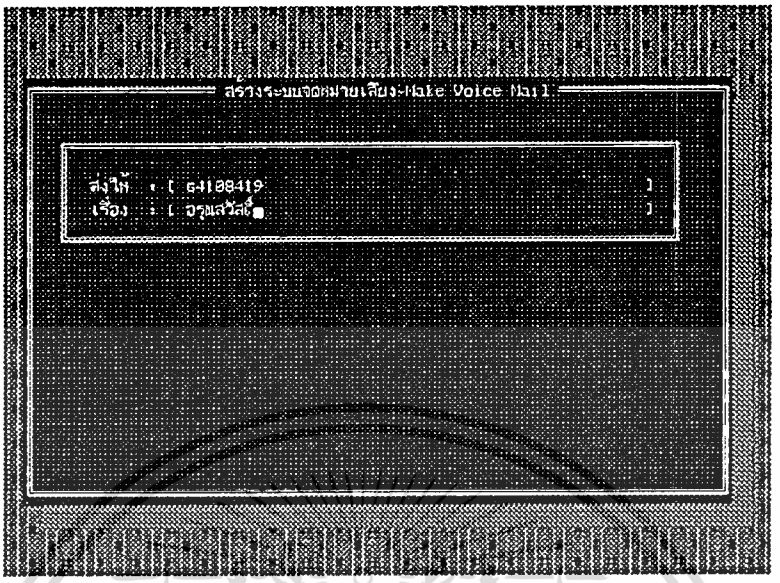
รูป 6.14 แสดงหน้าจอเลือกผู้รับขณะส่งจดหมาย

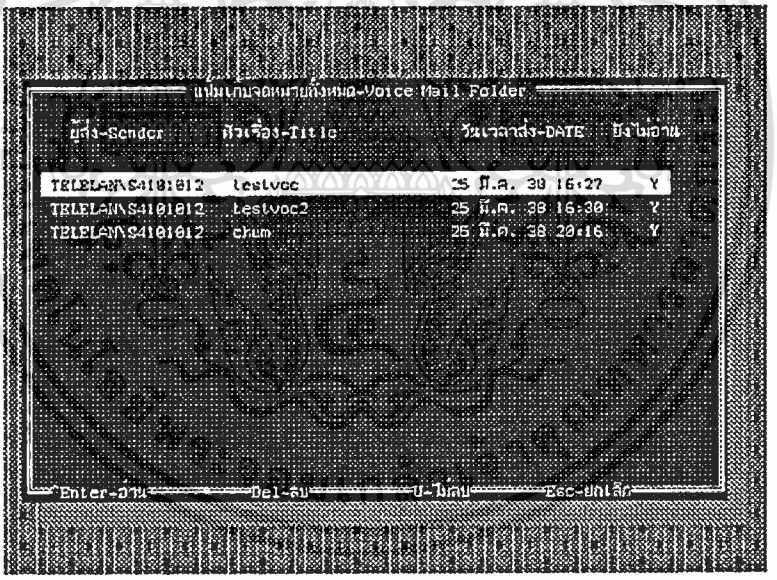


รูป 6.15 แสดงหน้าจอขณะอ่านจดหมาย

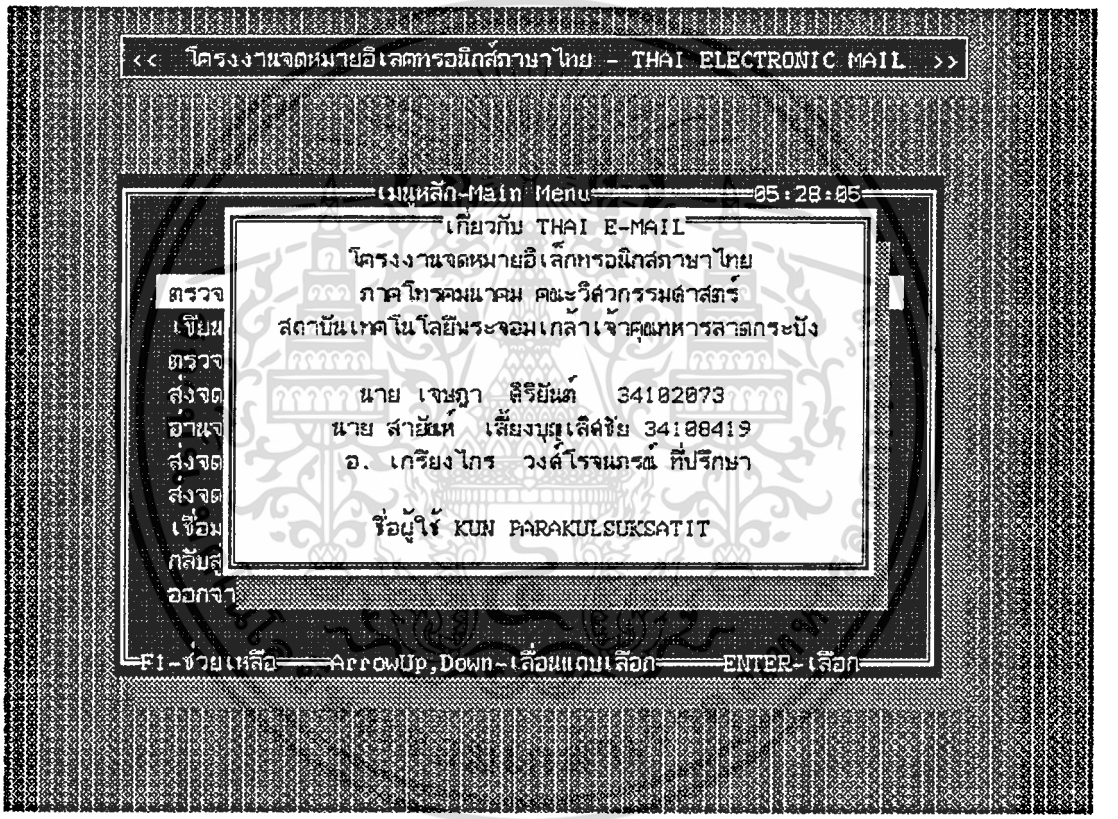


รูป 6.16 แสดงหน้าจอขณะอ่านจดหมายที่มีไฟล์แนบมาด้วย





รูป 16.17 แสดงหน้าจอขณะทำการส่งและอ่านจดหมายเสียง



รูป 16.18 แสดงหน้ารายละเอียดเกี่ยวกับจดหมายอิเลคทรอนิคส์ภาษาไทย

บทที่ 7
สรุปผลและแนวทางพัฒนา

จากการพัฒนาโปรแกรมระบบจดหมายอิเล็คทรอนิคส์ภาษาไทย ที่ได้ทำมาทั้งหมด สามารถสรุปผลการทำงานและแนวทางการพัฒนาได้ดังต่อไปนี้

1. การเขียนโปรแกรมบนระบบแลน ส่วนที่ติดต่อกับระบบจัดการของระบบเครือข่ายนั้น ส่วนใหญ่จะใช้การอินเทอร์รัพท์ แม้บางครั้งจะเขียนโปรแกรมตามคู่มือ API ของระบบเน็ตแวร์อย่างถูกต้องแล้วแต่ผลไม่ออกมาตามต้องการก็ไม่สามารถตรวจสอบหาข้อผิดพลาด (Debug) ได้ว่าผิดตรงส่วนใด

2. การพัฒนาโปรแกรมให้สามารถส่งข้ามศูนย์บริการ หากศูนย์บริการใดไม่เปิดให้ผู้ใช้ทั่วไปเข้าไปใช้ได้ก็ไม่สามารถส่งจดหมายไปให้ผู้ใช้ในศูนย์บริการนั้นได้ ในการทดลองทำการแก้ไขโดยการให้ผู้ใช้เชื่อมต่อไปยังศูนย์บริการนั้นก่อนจึงจะสามารถส่งจดหมายข้ามไปศูนย์บริการนั้นได้ ถ้าไม่สามารถเข้าไปใช้ศูนย์บริการนั้นก็ไม่สามารถส่งจดหมายได้

3. ในส่วนของเอดิเตอร์ หากสามารถพัฒนาเพิ่มเติมฟังก์ชั่นที่สำคัญๆของเอดิเตอร์เช่น การค้นหาคำ การกำหนด ย้ายบล็อคข้อมูล และการตัด และทำสำเนาข้อมูลเป็นต้น จะช่วยให้ส่วนเขียนจดหมายมีประสิทธิภาพยิ่งขึ้น

4. สำหรับแนวทางพัฒนาที่คิดว่าน่าจะเป็นประโยชน์และสามารถทำได้คือ ปรับปรุงระบบจดหมายอิเล็คทรอนิคส์ให้สามารถรับส่งกับโปรแกรมจดหมายอิเล็คทรอนิคส์อื่นๆได้ หรือสามารถติดต่อรับส่งจดหมายกับเครือข่ายอินเทอร์เน็ตได้ ในส่วนของการส่งจดหมายผ่านโมเด็มหากสามารถหา API ในการติดต่อกับศูนย์บริการให้ศูนย์บริการข้อมูลทำหน้าที่เป็นศูนย์กลางส่งจดหมายผ่านโมเด็มแทนสถานีผู้ใช้สถานีใดสถานีหนึ่งได้จะช่วยเพิ่มประสิทธิภาพมากยิ่งขึ้น

5. ในส่วนของการส่งจดหมายผ่านแฟ็กซ์โมเด็มที่ได้นำโครงงาน"รับส่งแฟ็กซ์โมเด็ม"มาใช้ประกอบนั้น สามารถนำมาพัฒนาต่อได้ในลักษณะเดียวกับการส่งจดหมายผ่านโมเด็มที่ได้กล่าวมาข้างต้น

กิตติกรรมประกาศ

โครงงานจดหมายอิเลคทรอนิคส์ภาษาไทยฉบับนี้จะไม่สามารถสำเร็จลงได้ ถ้าขาดคำแนะนำและความช่วยเหลือจาก อาจารย์เกรียงไกร วงศ์โรจนภรณ์ - อาจารย์ที่ปรึกษา และ อาจารย์ นภัทร์ สระเอี่ยม- ผู้ควบคุมระบบแลนของภาควิชาโทรคมนาคม

จึงขอขอบคุณบุคคลผู้มีพระคุณทั้งสองท่านมา ณ. ที่นี้ด้วย

ผู้จัดทำ

หนังสืออ้างอิง

1) Debra Deutsch,"Digital Communication/Electronic Mail Systems",Prentice Hall,Inc.,1990

2) Charles G. Rose,"Programer's Guide to NetWare,McGraw-Hill,Inc.,1990

3) User's manual,"Essential Communications",South Mountain Software,Inc.,1991

4) Atloz,"The Sound Blaster Book",Abacus,1991

5) Robert L Hummel,"Programer 's Technical Reference, Data and Fax Communication", Ziff-Davis Press Emryvilly California,1993

6) ธันวา ศรีประมง,"การเขียนโปรแกรมภาษาซีทางวิศวกรรม",มหาวิทยาลัยมหานคร, 1994

ภาคผนวก

ภาคผนวก ก.

โครงสร้างของบล็อคข้อมูลเสียงและฟังก์ชั่นที่มีให้ในไดรเวอร์การ์ดเสียง

1) **ส่วนของบล็อคข้อมูลตามโครงสร้างของบริษัทครีเอทีฟ**

บล็อค 0 : บล็อคจบ(End Block) ใช้แสดงว่าจบส่วนของข้อมูลแล้ว ถ้าเจอบล็อคนี้เอาท์พุทของข้อมูลแบบ voc ระหว่างที่กำลังเล่นอยู่จะหยุดทันที ดังนั้นตำแหน่งของบล็อคนี้จึงควรอยู่ที่ท้ายสุดของไฟล์แบบ voc นี้

รูปแบบของบล็อคจบ

ชนิดของบล็อค 1 ไบต์ = 0

ขนาดบล็อค ไม่มี

จำนวนข้อมูล ไม่มี

บล็อค 1 : บล็อคแสดงข้อมูลใหม่ (New Voice Block) เป็นบล็อคที่มีการใช้งานบ่อยที่สุด สิ่งที่บรรจุอยู่ในบล็อคนี้ได้แก่ ข้อมูลที่ถูกแซมเปิลมา, ขนาดของบล็อค, ไบต์แสดงอัตราการแซมปลิงสำหรับตอนบันทึกเสียง(SR byte) และไบต์แสดงอัตราการบีบอัดข้อมูลเสียง(Pack byte) สำหรับอัตราการแซมปลิงในไบต์ SR สามารถนำมาคำนวณหาอัตราแซมปลิงจริงๆ ได้จากสูตร

อัตราแซมปลิงจริง = -1,000,000 DIV (อัตราแซมปลิงในไบต์ SR - 256)

ค่าในไบต์บีบอัดข้อมูล ถ้าเป็น 0 แสดงว่าไม่มีการบีบอัด(นั่นคือข้อมูล 1 ค่าประกอบด้วย 8 บิต), ถ้าเป็น 1 อัตราการบีบอัดจะเป็น 2:1, ถ้าเป็น 2 แสดงว่าอัตราการบีบอัดจะเป็น 3:1 และค่า 3 จะแสดงว่าอัตราการบีบอัดเป็น 4:1

รูปแบบของบล็อคแสดงข้อมูลใหม่

ชนิดของบล็อค 1 ไบต์ = 1

ความยาวบล็อค 3 ไบต์

ไบต์ SR 1 ไบต์

ไบต์บีบอัด 1 ไบต์ = 0, 1, 2, 3

ขนาดข้อมูล x ไบต์

บล็อค 2 : บล็อคข้อมูลเสียงย่อย (Subsequent Voice Block) ใช้สำหรับแบ่งข้อมูลออกเป็นบล็อคเล็กๆ ซึ่งจะถูกใช้เมื่อข้อมูลเสียงมีขนาดใหญ่จนไม่สามารถใส่เข้าหน่วยความจำได้ในครั้งเดียว

รูปแบบบล็อคข้อมูลเสียงย่อย

ชนิดของบล็อค 1 ไบต์ = 2

ความยาวบล็อค 3 ไบต์

ขนาดข้อมูล x ไบต์

บล็อค 3 : บล็อคเงียบ (Silence Block) สามารถใช้กำหนดช่วงหยุด หรือช่วงเงียบในขณะเล่นเสียงได้ โดยการกำหนดค่า 0 ให้ตามจำนวนไบต์ที่ต้องการได้(โดยใช้วอยซ์เอดิเตอร์)

รูปแบบบล็อคเงียบ

ชนิดของบล็อค 1 ไบต์ = 3

ขนาดของบล็อค 3 ไบต์ = 3

ช่วงหยุด 2 ไบต์

อัตราแซมเปิล 1 ไบต์

บล็อค 4 : บล็อคมาร์คเกอร์ (Marker Block) เมื่อพบบล็อคมาร์คเกอร์นี้ในรูทีนเล่นเสียงของไดรฟ์เวอร์ CT-VOICE.DRV ค่าของไบต์มาร์คเกอร์จะถูกก็อบปี้ลงหน่วยความจำในตำแหน่งที่ระบุไว้โดยไดรฟ์เวอร์ และด้วยวอยซ์เอดิเตอร์คุณจะสามารถแบ่งข้อมูลขนาดใหญ่ให้เป็นบล็อคเล็กๆได้ โดยการใส่บล็อคมาร์คเกอร์ลงไปที่ตำแหน่งต่างๆจะไม่มีผลต่อการแปลงข้อมูลกลับเป็นเสียง เพียงแต่จะทำให้สามารถหาจุดที่กำลังอ่านข้อมูลเสียงอยู่ได้ง่ายขึ้น

รูปแบบของบล็อคมาร์คเกอร์

ชนิดของบล็อค 1 ไบต์ = 4

ความยาวของบล็อค 3 ไบต์ = 2

มาร์คเกอร์ 2 ไบต์

บล็อค 5 : บล็อคข้อความ (Message Block) เราสามารถแทรกข้อความที่เป็นแบบรหัสแอสกีลงในไฟล์แบบ VOC ได้โดยใช้บล็อคนี้ โดยข้อความแบบแอสกีนี้ต้องจบด้วยค่า 0 ในไบต์สุดท้ายของข้อความเสมอ

รูปแบบของบล็อคข้อความ

ชนิดของบล็อค 1 ไบต์ = 5

ความยาวของบล็อค 3 ไบต์

ข้อมูลแบบแอสกี x ไบต์

อักษรจบข้อความ 1 ไบต์ = 0

บล็อค 6 : บล็อคทำซ้ำ (Repeat Block) ใช้ระบุการอ่านข้อมูลซ้ำตามจำนวนครั้งที่ระบุไว้ที่เคาน์เตอร์ เช่นถ้าค่าในเคาน์เตอร์เป็น 4 แสดงว่าจะมีการอ่านข้อมูลนั้นทั้งหมด 5 ครั้ง (เล่นปกติ 1 กับอ่านซ้ำอีก 4 ครั้ง) โดยต้องใช้ร่วมกับบล็อค 7

รูปแบบของบล็อคทำซ้ำ

ชนิดของบล็อค	1 ไบต์ = 6
ความยาวของบล็อค	3 ไบต์ = 2
เคาน์เตอร์	2 ไบต์

บล็อค 7 : บล็อคจบการทำซ้ำ (Repeat End Block) ใช้ระบุว่าข้อมูลระหว่างบล็อค 6 และ 7 จะเป็นข้อมูลที่ต้องอ่านซ้ำ

รูปแบบของบล็อคจบการทำซ้ำ

ชนิดของบล็อค	1 ไบต์ = 7
ความยาวของบล็อค	3 ไบต์ = 0

2) รายละเอียดของฟังก์ชั่นที่มีให้ในไดรเวอร์ CT-VOICE.DRV

ฟังก์ชั่น 0 (BX=0) : หาเวอร์ชั่นของไดรเวอร์

ฟังก์ชั่นนี้จะส่งกลับค่าเลขเวอร์ชั่นของไดรเวอร์ CT-VOICE.DRV ที่กำลังใช้งานอยู่ หลังจากเรียกฟังก์ชั่นใช้งานแล้วค่าเลขเวอร์ชั่นจะถูกเก็บไว้ในรีจิสเตอร์ AX (โดย AH เก็บค่าเลขหน้าทศนิยมและAL เก็บเลขหลังจุด)

ฟังก์ชั่นหมายเลข 0 - หาเวอร์ชั่นของไดรเวอร์

อินพุท	BX = 00
เอาท์พุท	AH = เลขหน้าจุดทศนิยม
	AL = เลขหลังจุดทศนิยม

ฟังก์ชั่น 1 (BX=1) : ตั้งค่าแอดเดรสพอร์ต

ค่าแอดเดรสพอร์ตมาตรฐานจะถูกกำหนดไว้ในไฟล์ CT-VOICE.DRV อยู่ที่ตำแหน่งออฟเซตที่ 30h มีขนาดสองไบต์ ถ้าไม่มีการเรียกใช้ฟังก์ชั่น 1 นี้ก่อน ค่าแอดเดรสพอร์ตก็จะเป็นตามค่ามาตรฐานนี้ แต่ถ้าไดรเวอร์ต้องทำงานกับแอดเดรสพอร์ตอื่น ก็จะต้องระบุค่าใหม่ไว้ในรีจิสเตอร์ AX โดยต้องทำก่อนเรียกใช้ฟังก์ชั่น 3 มิฉะนั้นค่าที่ระบุไปใหม่จะไม่มีผล ค่าแอดเดรสที่ใช้ได้อาจเป็น 210h, 220h, 230h, 240h, 250h และ 260h ขึ้นอยู่กับเวอร์ชั่นของการ์ดที่ใช้

ฟังก์ชั่น 1 - ตั้งค่าแอดเดรสพอร์ต

อินพุท BX = 01

AX = แอดเดรสพอร์ต

เอาท์พุท ไม่มี

หมายเหตุ ต้องเรียกใช้ก่อนฟังก์ชั่น 3

ฟังก์ชั่น 2 (BX=2) : ตั้งค่าอินเทอร์รัพต์

ฟังก์ชั่นนี้มีลักษณะคล้ายฟังก์ชั่น 1 โดยค่าปกติของหมายเลขอินเทอร์รัพต์ใน CT-VOICE .DRV จะถูกกำหนดไว้ที่ออฟเซต 32h ขนาด1ไบต์ เช่นเดียวกับฟังก์ชั่น 1 คือถ้าจะกำหนดหมายเลขอินเทอร์รัพต์ใหม่ด้วยฟังก์ชั่นนี้ต้องกระทำก่อนการเริ่มต้น(initial)ซาวนด์การ์ด โดยมีหลายค่าที่ใช้ได้ (ได้แก่อินเทอร์รัพต์หมายเลข2, 3, 5 และ7) ขึ้นอยู่กับเวอร์ชั่นของการ์ด

ฟังก์ชั่น 2 - ตั้งค่าหมายเลขอินเทอร์รัพต์

อินพุท BX = 02

AX = หมายเลขอินเทอร์รัพต์

เอาท์พุท ไม่มี

หมายเหตุ ต้องเรียกใช้ฟังก์ชั่น 3 ก่อน

ฟังก์ชั่น 3 (BX=3) : เริ่มต้นการทำงานไดรเวอร์

ใช้เริ่มต้นการทำงานของซาวนด์การ์ดสำหรับการทำงานต่อไปของฟังก์ชั่นอื่นๆ โดยจะมีการตรวจสอบพารามิเตอร์ทั้งหมดว่ากำหนดไว้ถูกต้องหรือไม่แล้วส่งผลกลับว่าไว้ในรีจิสเตอร์AX

โดย - ถ้า AX=0 แสดงว่าไม่มีข้อผิดพลาดเกิดขึ้น แอดเดรสพอร์ตและหมายเลข IRQ ก็ถูกต้อง

- ถ้า AX=1 แสดงว่าไม่สามารถหาซาวนด์บลาสเตอร์การ์ดได้พบ

- AX=2 แสดงว่าเกิดความผิดพลาดขึ้นที่ I/O นั่นคือตั้งค่าแอดเดรสพอร์ตไว้ไม่ถูกต้อง

- AX=3 แสดงว่าเกิดความผิดพลาดจากอินเทอร์รัพต์ นั่นคือหมายเลขอินเทอร์รัพต์ที่ตั้งไว้ไม่ถูกต้อง

ฟังก์ชั่น 3 - เริ่มต้นการทำงานไดรเวอร์

อินพุท BX = 03

เอาท์พุท AX = 0 , ไม่เกิดข้อผิดพลาด

AX = 1 , ไม่พบซาวนด์บลาสเตอร์การ์ด

AX = 2 , แอดเดรสพอร์ตผิดพลาด

AX = 3 , หมายเลขอินเทอร์รัพต์ผิดพลาด

ฟังก์ชั่น 4 (BX=4) : เปิด/ปิดเสียงที่เอาท์พุต

ฟังก์ชั่นนี้ทำหน้าที่เปิด/ปิดเอาท์พุตของซาวนด์บลาสเตอร์การ์ดตามค่าที่ใส่ให้ในรีจิสเตอร์ AX: โดย AX=0 จะเป็นการปิดเสียงแต่เอาท์พุตและจะเปิดเสียงเอาท์พุตเมื่อ AX=1 ตามปกติ ฟังก์ชั่น 4 นี้จะถูกเรียกจากฟังก์ชั่น 3 เพื่อเปิดเสียงเอาท์พุตและถูกเรียกจากฟังก์ชั่น 9 เพื่อปิดเอาท์พุตโดยอัตโนมัติอยู่แล้ว

ฟังก์ชั่น 4 - เปิด/ปิดเสียงเอาท์พุต

อินพุท	BX = 04
	AL = 0 , ปิด
	AH = 1 , เปิด
เอาท์พุท	ไม่มี

ฟังก์ชั่น 5 (BX=5) : ตั้งค่าแอดเดรสของเวิร์ดสถานะ

ฟังก์ชั่นนี้จะทำให้เราสามารถควบคุมการทำงานของซาวนด์บลาสเตอร์การ์ดได้ โดยค่าออฟเซต:แอดเดรสจะถูกส่งไปยังไดรเวอร์โดยผ่านรีจิสเตอร์คู่ ES:DI ค่าสถานะการทำงานปัจจุบันก็จะถูกเขียนลงที่ตำแหน่งนี้เสมอ โดยค่าอื่นๆที่ไม่ใช่ 0 จะแสดงว่าการ์ดกำลังทำงานบางอย่าง (เขียนหรืออ่าน)

ฟังก์ชั่น 5 - ตั้งค่าแอดเดรสของเวิร์ดสถานะ

อินพุท	BX = 06
	ES:DI = แอดเดรสของเวิร์ดสถานะ
เอาท์พุท	ไม่มี

ฟังก์ชั่น 6 (BX=6) : แปลงข้อมูลออกเป็นเสียง

ฟังก์ชั่นนี้จะแปลงข้อมูลที่อัดไว้จากหน่วยความจำให้ออกเป็นเสียง โดยใช้รีจิสเตอร์คู่ ES:DI ผ่านค่าเซกเมนท์:ออฟเซตของข้อมูลบล็อคแรกที่จะแปลง เมื่อเริ่มแปลงข้อมูล ค่าเวิร์ดสถานะก็จะถูกเซตให้เป็น FFFFh โดยอัตโนมัติจนกระทั่งยกเลิกการแปลง(จบการทำงานของฟังก์ชั่น 6)ค่าเวิร์ดสถานะจึงกลายเป็น 0 เหมือนเดิม

ฟังก์ชั่น 6 - แปลงข้อมูลออกเป็นเสียง

อินพุท	BX = 06
	ES:DI = แอดเดรสของข้อมูลเสียง
เอาท์พุท	ไม่มี

ฟังก์ชั่น 7 (BX=7) : บันทึกเสียง

ฟังก์ชั่นนี้จะทำให้สัญญาณเสียงดิจิตอลที่การ์ดแปลงได้โอนย้ายเข้าสู่หน่วยความจำผ่านทางแชนเนล DMA รีจิสเตอร์ AX จะเก็บค่าอัตราแซมปลิง(มีค่าได้ระหว่าง 4,000 ถึง 44,100 เฮิร์ตซ แต่ก็ต้องขึ้นกับเวอร์ชั่นของการ์ดด้วย) ความยาวของข้อมูลที่จะบันทึกเข้าไปจะถูกกำหนดไว้ที่รีจิสเตอร์คู่ DX:CX และก็จะมีรีจิสเตอร์คู่ ES:DI เก็บแอดเดรสเริ่มต้นของหน่วยความจำที่จะนำข้อมูลไปเก็บก่อนย้ายลงสู่ฮาร์ดดิสก์

ในขณะทำการบันทึกเสียง ค่าเวิร์ดสถานะก็จะมีค่าเป็นอื่นที่ไม่ใช่ 0 จนเมื่อบัฟเฟอร์ในหน่วยความจำหรือข้อมูลที่บันทึกถึงความยาวที่ระบุไว้ใน DX:CX เวิร์ดสถานะจึงจะมีค่ากลับเป็น 0 เหมือนเดิม

ฟังก์ชั่น 7 - บันทึกเสียง	
อินพุท	BX = 07
	AX = อัตราการแซมปลิงข้อมูล
	DX:CX = แอดเดรสข้อมูลเสียงในหน่วยความจำ
เอาท์พุท	ไม่มี

ฟังก์ชั่น 8 (BX=8) : หยุดการทำงานของการ์ด

ฟังก์ชั่นนี้จะอินเทอร์รัพต์การทำงานทุกอย่างของซาวนด์การ์ดไม่ว่าจะเป็นการบันทึกหรือเล่น เวิร์ดสถานะจะถูกรีเซ็ตเป็น 0

ฟังก์ชั่น 8 - หยุดการทำงานของการ์ด	
อินพุท	BX = 08
เอาท์พุท	ไม่มี
หมายเหตุ	เวิร์ดสถานะมีค่าเป็น 0

ฟังก์ชั่น 9 (BX=9) : ถอนการติดตั้งไดรเวอร์

ฟังก์ชั่นนี้จะถอนการติดตั้งซอฟต์แวร์ไดรเวอร์ของซาวนด์บลาสเตอร์ออกจากหน่วยความจำ ซึ่งจะมีผลให้การเชื่อมต่อกับลำโพงถูกตัดขาดไปโดยอัตโนมัติ

ฟังก์ชั่น 9 - ถอนการติดตั้งไดรเวอร์	
อินพุท	BX = 09
เอาท์พุท	ไม่มี

จากโครงสร้างและฟังก์ชั่นต่างๆของการ์ดเสียง เราสามารถนำมาประยุกต์ใช้ร่วมกับโปรแกรมภาษาซีและแอสแซมบลีเพื่อทำการบันทึกหรือเล่นเสียงได้

ภาคผนวก ข.

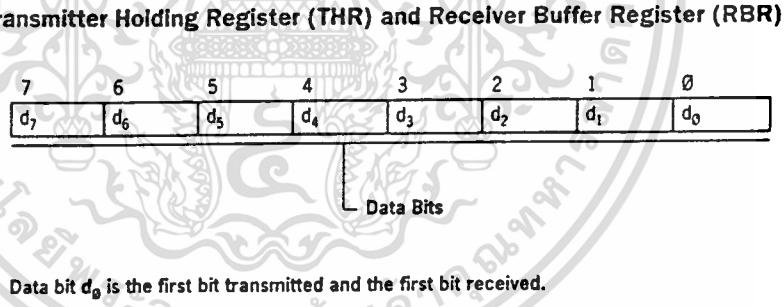
รีจิสเตอร์ภายในของ 8250 และคำสั่งที่ใช้กับโมเด็ม

รีจิสเตอร์ภายในของ 8250

- Transmitter Holding Register (THR) ขบวนการส่งข้อมูลอนุกรมจะเริ่มต้นขึ้นเมื่อโปรเซสเซอร์มีการเขียนข้อมูล(1ไบต์) ลงในรีจิสเตอร์ THR ของ 8250 หลังจากนั้น 8250 ก็จะย้ายข้อมูลนั้นไปยังtransmitter shift register(TSR) ถึงตอนนี้จะได้ไบต์เป็นอนุกรมโดยมีพาริตี้และเฟรมมิงบิตเพิ่มเข้ามา ขบวนของบิตที่ได้นี้จะถูกส่งไปยังเอาท์พุทอนุกรม(SOUT) ของ UART

THR หรือที่รู้จักกันว่าเป็น transmitter buffer register (TBR) จะถูกอ้างถึงผ่านรีจิสเตอร์ 0 THR นี้เป็นรีจิสเตอร์แบบเขียนได้อย่างเดียว(write-only) โดยค่าที่เขียนลง THR จะไม่เป็นไปตามลำดับ

รูปแบบของ THR แสดงได้ดังรูป 1ข. LSB ของตัวอักษรที่จะส่งจะถูกเขียนลงที่บิต 0 (ซึ่งถือเป็น LSB ของรีจิสเตอร์ด้วย) หรือนั่นคือ LSB ของอักษรก็คือบิตแรกที่จะส่งออกไปยังเส้นทางอนุกรม



รูป 1ข. แสดงรูปแบบของTHR และRBR

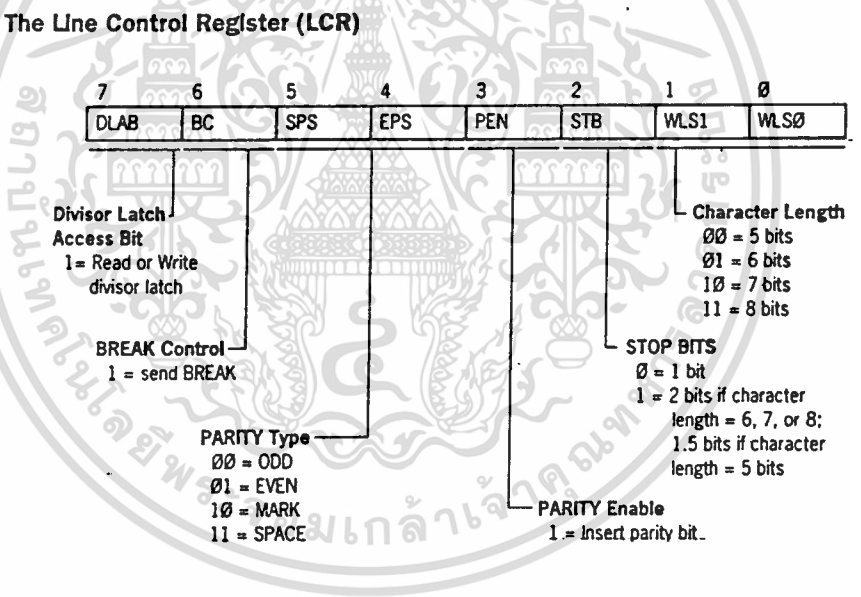
- Receiver Buffer Register (RBR) อินพุทอนุกรม(SIN) ของ 8250 จะเป็นที่รับข้อมูลอนุกรมของ 8250 ซึ่งจะมีการตัดเฟรมมิงและพาริตี้บิตออกเหลือเพียงแต่บิตข้อมูลส่งไปยัง Receiver Shift Register (RSR) โดยบิตข้อมูลที่รับเข้ามาจะมาประกอบรวมกันเป็นตัวอักษรที่ RSR นี้ หลังจากนั้นตัวอักษรที่ได้ก็จะถูกส่งต่อไปยัง RBR ซึ่งเป็นที่ที่โปรเซสเซอร์จะมาอ่านข้อมูลไป ข้อสังเกต: จำนวนบิตที่ใช้เพื่อประกอบกันเป็นตัวอักษรจะถูกกำหนดโดย RSR

ความยาวของตัวอักษรสามารถเป็นได้ตั้งแต่ 5-8 บิต (บิตที่ไม่ได้ใช้ก็จะถูกมาสค์ไว้โดยโปรเซสเซอร์ก่อนนำไปใช้ การอ้างถึง RBR ทำได้โดยผ่านรีจิสเตอร์ 0 แอดเดรสตำแหน่งเดียวกับ

THR แต่ RBR ถูกกำหนดให้ใช้เป็นรีจิสเตอร์สำหรับอ่านเท่านั้น และถึงแม้จะใช้รีจิสเตอร์แอดเดรสของ UART เดียวกัน แต่ THR และRBR ก็มีช่วงเวลาสำหรับข้อมูล(data space) ที่ต่างกัน เช่น การเขียนข้อมูลลงTHR ขณะที่ยังมีข้อมูลที่รับเข้ามาแต่ยังไม่ได้อ่านอยู่ใน RBR ก็สามารถทำได้โดยไม่มีผลกระทบต่อข้อมูลที่รับเข้ามานั้น นอกจากนี้ยังใช้ THR รวมกับ Divisor Latch Register เพื่อเซตค่าบอดของ 8250 ได้อีกด้วย โดยรูปแบบของ RBR แสดงได้ดังในรูป 1ข. เช่นกัน

- Line Control Register (LCR) LCR แบ่งออกเป็น 7 ฟิลด์ รูปแบบของข้อมูลอะซิงโครนัสที่จะใช้ในการรับและส่งจะถูกกำหนดขึ้นโดยค่าที่กำหนดลงในรีจิสเตอร์นี้ นอกจากนี้เมื่อใช้รวมกับ Divisor Latch Register ยังสามารถใช้เซตค่าบอดของ 8250 ได้อีกด้วย

LCR จะถูกอ้างถึงได้โดยผ่านรีจิสเตอร์ 3 ของ UART และสามารถใช้ได้ทั้งอ่านและเขียน รูป 2ข. แสดงรูปแบบของ LCR



รูป 2ข. แสดงรูปแบบของ Line Control Register

บิต 1-0 : 2 บิตนี้ใช้กำหนดจำนวนบิตข้อมูลในแต่ละเฟรมที่จะส่งหรือรับ ความยาวข้อมูลนี้เป็นได้ตั้งแต่ 5-8 บิต โดยเลือกค่าของสองบิตนี้ตามความยาวที่ต้องการ(ดูได้ดังในรูป2ข.)

บิต 2 : ใช้กำหนดจำนวนของบิตหยุด(STB) ที่จะเพิ่มเข้าไปในแต่ละเฟรมข้อมูลระหว่างการส่ง โดยถ้ากำหนดให้บิตนี้เป็น 0 จะไปเพิ่ม 1 บิตหยุด และถ้าเป็น 1 จะไปเพิ่มให้ 2 บิตหยุดกับข้อมูลที่จะทำการส่ง แต่ถ้าข้อมูลที่จะส่งนั้นมีความยาว 5 บิตและเลือกใช้ 1 บิตหยุด บิตหยุดจริงที่จะถูกสร้างระหว่างการส่งจะเป็น 1.5 บิต

บิต 5-3 : ใช้เพื่อกำหนดว่าจะให้มีพาริตี้บิตระหว่างการส่งและมีการตรวจสอบเช็คพาริตี้ระหว่างการรับหรือไม่(พาริตี้บิตจะอยู่ระหว่างข้อมูลบิตสุดท้ายกับบิตหยุดแรก) โดยมีชื่อเรียกเฉพาะสำหรับทั้งสามบิตคือ บิตที่3-Parity Enable (PEN), บิตที่4-Even Parity Select (EPS) และบิตที่5-Stick Parity Select bit (SPS) ค่าและความหมายของแต่ละบิตแสดงได้ดังนี้

Bit5 (SPS)	Bit4 (EPS)	Bit3 (PEN)	Parity
x	x	0	NONE : ไม่มีพาริตี้บิต, ไม่มีการตรวจพาริตี้บิตทางด้านรับ
0	0	1	ODD : เป็นพาริตี้คี่ คือบิตข้อมูลและพาริตี้จะมีเลข1อยู่เป็นคี่
0	1	1	EVEN : เป็นพาริตี้คู่ คือบิตข้อมูลและพาริตี้จะมีเลข1อยู่เป็นคู่
1	0	1	MARK : พาริตี้บิต จะมีค่าเป็น 1 เสมอ
1	1	1	SPACE : พาริตี้บิต จะมีค่าเป็น 0 เสมอ

บิต 6 : เป็น Break Control(BC) การเซตค่าของบิตนี้จะเป็นการควบคุมสัญญาณเอาท์พุทของ 8250 โดยตรง และยังใช้สร้างสัญญาณ BREAK ได้อีกด้วย โดยเมื่อบิต6นี้มีค่าเป็น 1 SOUT จะถูกบังคับให้อยู่ในสถานะ SPACE และจะยังคงอยู่ในสถานะนี้จนกว่าจะมีการเซต 0 มายังที่บิตนี้

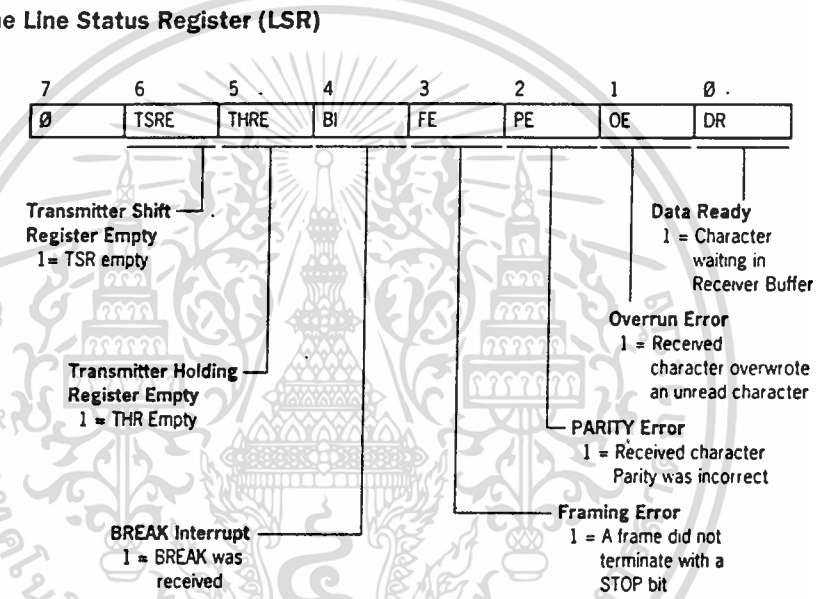
บิต 7 : Divisor Latch Access Bit (DLAB) เมื่อเซตบิตนี้เป็น 0 จะเป็นการอ้างเข้าไปถึง THR, RBR และIER ส่วนเมื่อเซตเป็น 1 จะเป็นการเคลียร์ค่าในรีจิสเตอร์เหล่านี้เพื่อใช้อ้างถึงแลทช์ตัวหาร ไบต์สำคัญสูงสุดและต่ำสุดของตัวหารกำหนดบอดอาจจะถูกอ้างถึงโดยผ่านรีจิสเตอร์ 0และ1ของ UART ก็ได้ตามลำดับ

- Line Status Register (LSR) : แบ่งได้เป็น 6 ฟิลด์ แสดงสถานะของการโอนย้ายข้อมูล โปรเซสเซอร์สามารถตรวจสอบค่าของฟิลด์เหล่านี้ได้โดยการอ่าน LSR และตรวจสอบทีละบิต โดยเมื่อมีการอ่านรีจิสเตอร์นี้แล้วจะมีการเคลียร์ค่าของบางฟิลด์ให้โดยอัตโนมัติ ดังนั้นโปรแกรมเมอร์ควรมีการเก็บค่าของ LSR เมื่อได้อ่านแล้ว

การอ้างเข้าสู่ LSR กระทำผ่านรีจิสเตอร์ 5 ของ UART และปกติจะใช้เป็นรีจิสเตอร์ที่อ่านได้อย่างเดียว รูป 3ข. แสดง LSR โดยบิตที่7จะถูกเซตให้เป็น 0 ตลอด

บิต0 : Data Ready(DR) เมื่อ 8250 รับข้อมูลเข้ามาและส่งไปไว้ยัง RBR แล้ว DRบิตนี้จะมีค่าเป็น 1 และเมื่อมีการอ่านข้อมูลจาก RBR โดยอัตโนมัติ บิตDRนี้จะถูกเคลียร์เป็น 0

บิต1 : Overrun Error(OE) ส่วนรับข้อมูลของ 8250 จะทำงานอย่างต่อเนื่อง ถ้าข้อมูลที่รับเข้ามา(รับเก็บไว้ใน RSR ก่อน) ถูกส่งไปที่ RBR ก่อนที่ข้อมูลจะถูกโปรเซสเซอร์อ่านไป ก็จะทำให้ข้อมูลเดิมเสียหายได้ สภาวะเช่นนี้เรียกว่าความผิดพลาดจากโอเวอร์รันทางด้านรับ(receiver overrun error) เมื่อมีการตรวจพบ 8250ก็จะเซตให้บิต OE นี้เป็น 1 และจะกลับเป็น 0 โดยอัตโนมัติเมื่อโปรเซสเซอร์อ่านค่าใน LSR



รูป 3ข. แสดงรูปแบบของ Line Status Register(LSR)

บิต2 : Parity Error(PE) เมื่อรับข้อมูลเข้ามาจะมีการตรวจสอบพาริตี้บิตตามค่าพารามิเตอร์ที่เซตไว้ใน LCR ถ้าพาริตี้บิตจากข้อมูลที่รับเข้ามาไม่ถูกต้อง บิตPEก็จะถูกเซตเป็น1 โดยจะยังคงเป็น 1 แม้ว่าหลังจากนั้นจะได้ข้อมูลที่มีพาริตี้ถูกแล้วก็ตาม โดยจะถูกเคลียร์เป็น 0เมื่อมีการอ่าน LSRเท่านั้น

บิต3 : Framing Error(FE) ถ้าข้อมูลที่รับเข้ามาไม่มีอย่างน้อย 1 บิตหยุด หรือถ้าบิตที่ตามข้อมูลสุดท้ายหรือพาริตี้บิตมาไม่ใช่บิตหยุด 8250จะเซตบิต FE นี้เป็น 1 เมื่อมีการอ่าน LSR เท่านั้นบิตนี้จึงจะถูกเคลียร์เป็น 0

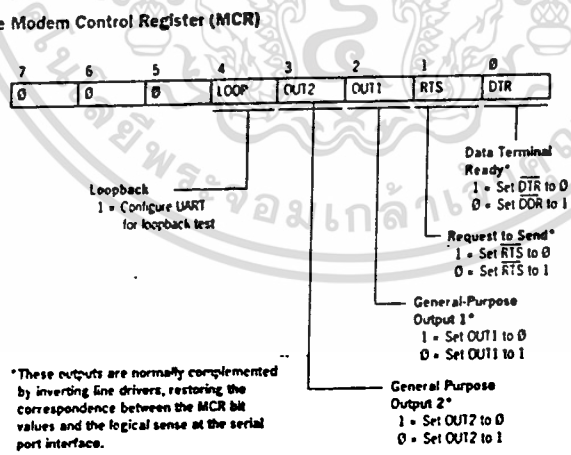
บิต4 : Break Interrrupt(BI) 8250จะเซตบิตนี้เป็น 1 เมื่อเกิดสภาวะ BREAK ขึ้นที่อินพุตอนุกรม บิตนี้จะถูกเคลียร์เป็น 0เมื่อโปรเซสเซอร์ได้อ่าน LSR แล้วเท่านั้น

บิต5 : Transmitter Holding Register Empty(THRE) บิตนี้จะถูกเซตเป็น 1 เมื่อ8250 ได้ทำการย้ายข้อมูลจาก THR ไปไว้ใน TSR ซึ่งก็แสดงว่า THR พร้อมที่จะรับข้อมูลชุดใหม่แล้ว แต่ถ้ามีการเขียนข้อมูลชุดใหม่ลงใน THR ขณะที่บิต THRE เป็น 0 อยู่จะเท่ากับเป็นการเขียนทับข้อมูลเดิมใน THR ซึ่งยังไม่ถูกส่งออกไป สภาวะเช่นนี้เรียกว่าความผิดพลาดโอเวอร์รันทางด้านส่ง (Transmitter overrun error)

THRE จะถูกเคลียร์เป็น 0 เมื่อโปรเซสเซอร์ทำการเขียนข้อมูลลงยัง THR นั่นคือค่าของบิต THRE จะขึ้นอยู่กับสถานะของ THR โดยการที่ THRว่างก็ไม่ได้หมายความว่า TSR จะว่างหรือการส่งข้อมูลอนุกรมได้หยุดลง

บิต6 : Transmitter Shift Register Empty(TSRE) เมื่อข้อมูลใน TSR ถูกส่งออกไปแล้วไม่มีข้อมูลชุดใหม่พร้อมที่จะถูกส่งจาก THR 8250จะเซตบิต TSRE นี้เป็น 1 และเมื่อ 8250 ส่งข้อมูลจาก THR ไปยัง TSR บิตTSRE นี้ก็จะถูกเคลียร์เป็น 0 ทันที

- Modem Control Register(MCR) MCR จะควบคุมสัญญาณเอาท์พุทที่ส่งจาก 8250 ไปยังโมเด็ม โดยเป็นรีจิสเตอร์แบบอ่าน/เขียน ซึ่งอ้างผ่านรีจิสเตอร์ 4 ของUART แบ่งออกได้เป็น 5 ฟิลด์ดังรูป 4.5 (บิต7,6,5 ถูกเซตให้เป็น 0 เสมอ)



รูป 4ข. แสดงรูปแบบของ Modem Control Register

บิต0 : Data Terminal Ready(DTR) สัญญาณ DTR จะถูกส่งจาก 8250 ไปยังอุปกรณ์ภายนอกที่เชื่อมต่ออยู่ ค่าที่เขียนลงที่บิตนี้จะไปควบคุมเอาท์พุทของสัญญาณ $\overline{DTR}$ ของ 8250 โดย

ตรง คือเมื่อเซตให้บิต DTR เป็น1 จะได้เอาท์พุต $\overline{DTR}$เป็น 0 และเมื่อเซตบิต DTR เป็น 0 จะได้เอาท์พุต$\overline{DTR}$ เป็น 1

บิต1 : Request To Send(RTS) สัญญาณ RTS จะถูกส่งจาก 8250 ไปยังอุปกรณ์ภายนอก เช่นกัน เมื่อเซตให้บิตนี้เป็น 0 จะทำให้ $\overline{RTS}$ เป็น 1 และเมื่อบิตนี้เป็น 1 $\overline{RTS}$ จะมีค่าเป็น 0

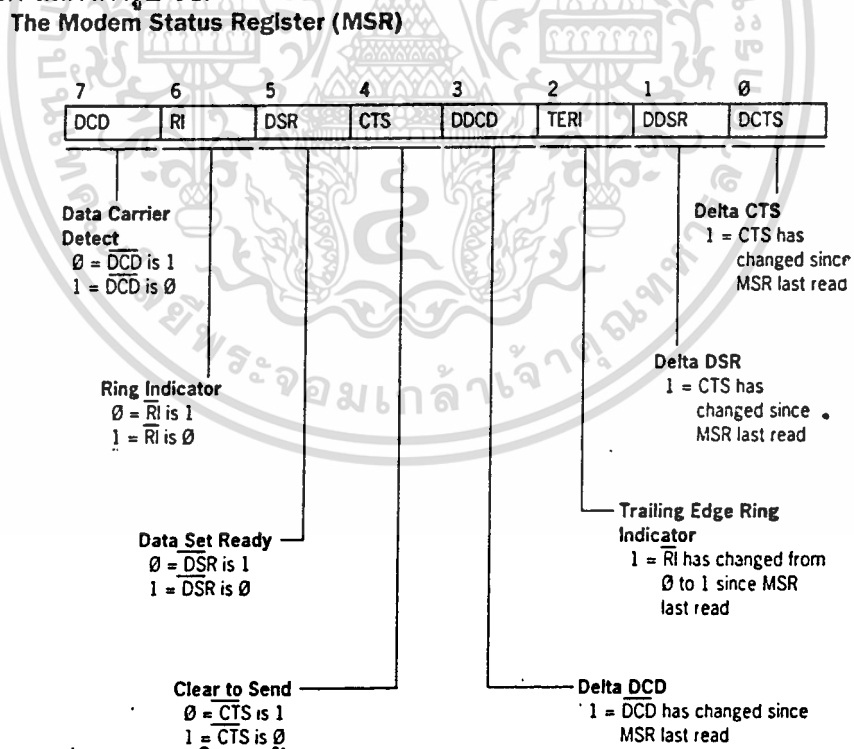
บิต2 : General-Purpose Output(OUT1) สัญญาณ OUT1 เป็นเอาท์พุทที่นำไปใช้ประโยชน์ได้ทั่วไป โดยถ้าเซตให้บิตนี้เป็น 0 จะได้เอาท์พุทออกที่ $\overline{OUT1}$เป็น1และเมื่อบิตเป็น0จะได้ $\overline{OUT1}$=1

บิต3 : GeneralPurpose Output(OUT2) ลักษณะเดียวกับบิต2 เพียงแต่ผลออกที่ $\overline{OUT2}$

บิต4 : Loopback(LOOP) เป็นบิตควบคุมการทำงานของลูปแบ็ค ซึ่งเป็นความสามารถอย่างหนึ่งที่8250มีอยู่แล้ว

- Modem Status Register (MSR) MSR แบ่งออกเป็น 7 ฟิลด์ทำหน้าที่แสดงให้เห็นถึงสภาวะปัจจุบันของโมเด็ม และการเปลี่ยนแปลงที่เกิดขึ้นหลังจากการอ่าน MSR ครั้งสุดท้าย

การอ้างถึง MSR จะอ้างผ่านรีจิสเตอร์ 6 ของUART และสามารถเขียนและอ่านได้ โครงสร้างของ MCR แสดงดังรูป 5ข.

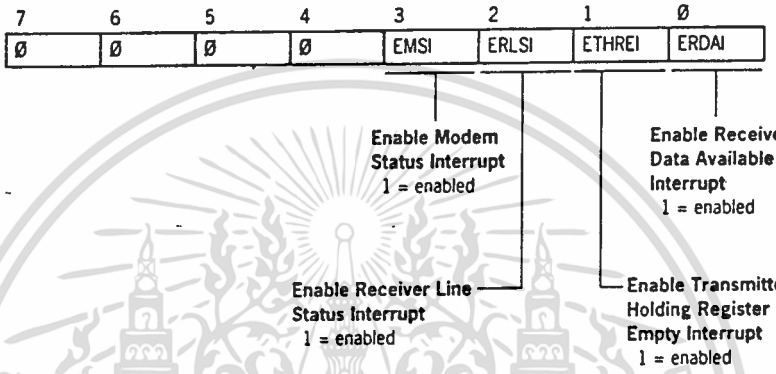


รูป 5ข. แสดงโครงสร้างของ Modem Status Register(MSR)

- Interrupt Enable Register(IER) IER แสดงอินเทอร์รัพต์ทั้ง4ชนิดที่ 8250 สามารถสร้างได้ โดยเมื่อมีอินเทอร์รัพต์เกิดขึ้น(enable) ก็จะมีการรายงานให้ทราบใน IIR และจะไปแอคทีพเอาท์พุทอินเทอร์รัพต์ของ 8250 ด้วย

การอ้างถึง IER จะกระทำผ่านรีจิสเตอร์ 1 ของUART โดยเป็นรีจิสเตอร์แบบอ่าน/เขียน โครงสร้างของ IER แสดงให้ดูได้ดังรูป 6ข. (บิต4-7 ถูกเซตให้เป็น 0 เสมอ)

The Interrupt Enable Register (IER)



รูป 6ข. แสดงโครงสร้างของ Interrupt Enable Register(IER)

บิต0 : Enable Received Data Available Interrupt(ERDAI) เมื่อ ERDAI เป็น 0 อินเทอร์รัพต์จะถูกดิสเอเบิล และเมื่อเซตบิตนี้เป็น 1 ก็จะเป็นการเอ็นเนเบิล RDAอินเทอร์รัพต์ โดยอินเทอร์รัพต์จะเกิดขึ้นทุกครั้งที่ข้อมูลที่รับเข้ามาได้ถูกย้ายจาก RSR ไปยัง RBR

บิต1 : Enable Transmitter Holding Register Empty Interrupt(ETHREI) เซตบิตนี้เป็น 1 จะเป็นการเอ็นเนเบิล THREอินเทอร์รัพต์ และอินเทอร์รัพต์นี้จะเกิดขึ้นทุกครั้งที่ข้อมูลถูกย้ายจาก THR ไปยัง TSR

บิต2 : Enable Line Status Interrupt (ERLSI) เซตบิตนี้เป็น1 จะเอ็นเนเบิล RLSอินเทอร์รัพต์ และอินเทอร์รัพต์นี้จะถูกส่งออกไปเมื่อบิตOE,PE,FE หรือBIบิตใดบิตหนึ่งหรือมากกว่าถูกเซตเป็น 1

บิต3 : Enable Modem Status Interrupt(EMSI) เซตบิตนี้เป็น 1 จะเอ็นเนเบิลอินเทอร์รัพต์ MSโดยอินเทอร์รัพต์นี้จะเกิดขึ้นเมื่อบิต DCTS, DDSR, TERI หรือDDCDถูกเซตเป็น 1

- Interrupt Identification Register (IIR) อินเทอร์รัพต์4ระดับซึ่งระบุไว้โดย IER จะถูกจัดลำดับก่อนหลังให้โดย 8250 คือถ้ามีอินเทอร์รัพต์เกิดขึ้นมากกว่า 1ชนิด อินเทอร์รัพต์ที่มีความสำคัญสุดเท่านั้นที่จะถูกแจ้งมายัง IIR ระดับการอินเทอร์รัพต์4ระดับได้แก่

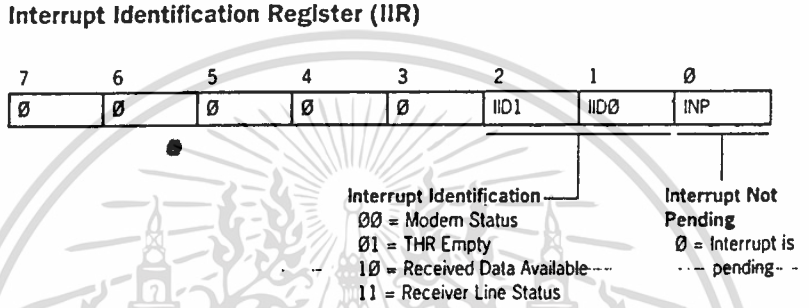
1. Receiver Line Status(ความสำคัญสูงสุด)

2. Received Data Ready

3. Transmitter Holding Register Empty

4. Modem Status (ความสำคัญต่ำสุด)

การอ้างถึง IIR ทำได้โดยผ่านรีจิสเตอร์ 2 ของUART โดย IIR นี้เป็นรีจิสเตอร์แบบอ่านเท่านั้น โครงสร้างของ IIR แสดงได้ดังในรูป 7ข.(บิต 3-7 ถูกเซตให้เป็น 0 ตลอด)



รูป 7ข. แสดงโครงสร้างของ Interrupt Identification Register (IIR)

บิต0 : Interrupt Not Pending (INP) กรณีที่ไม่มีการขออินเทอร์รัพต์ บิตนี้จะมีค่าเป็น 1 แต่ถ้าเกิดมีอินเทอร์รัพต์จาก 8250 ขึ้นบิตนี้ก็จะมีค่าเป็น 0 ทันที อินเทอร์รัพต์ที่เข้ามาอาจมีมากกว่าหนึ่งได้ แต่จะมีเพียงอินเทอร์รัพต์ที่มีความสำคัญสูงสุดเท่านั้นที่จะได้เข้าไปอยู่ในฟิลด์ IID

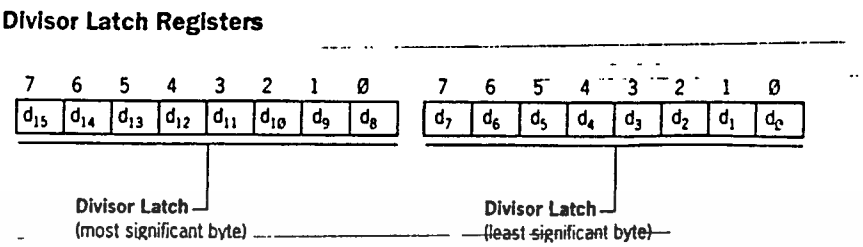
บิต2-1 : Interrupt ID Field (IID) ถ้า INP มีค่าเป็น 0 ฟิลด์ IID จะบรรจุอินเทอร์รัพต์ที่มีความสำคัญสูงสุด

- Divisor Latch Register : 8250 มีบอดเจเนเรเตอร์ที่โปรแกรมได้ ทำหน้าที่หาค่าเวลาอ้างอิงจากภายนอกโดยใช้ตัวหารที่มีค่าตั้งแต่ 1-65535 ผลที่ได้ก็จะเป็นสัญญาณ16เท่าของคล็อก ซึ่งจะนำไปใช้เป็นสัญญาณนาฬิกาให้กับส่วนส่งข้อมูลที่อยู่ภายใน ค่าตัวหารที่จะทำให้ได้ค่าบอดตามต้องการหาได้จาก

$$\text{ตัวหาร} = f/16 \times \text{บอดเรต}$$

โดย f เป็นค่าเวลาอ้างอิงภายนอกหน่วย เฮิร์ตซ ซึ่งพีซีส่วนใหญ่จะใช้คริสตอล1.8432 เมกกะเฮิร์ตซเป็นสัญญาณอ้างอิงจากภายนอก

ค่าตัวหารนี้จะถูกเขียนลง 8250 โดยแยกเป็น 2 ไบต์คือ ไบต์สูงและไบต์ต่ำ คืออยู่ในรูปแลทช์ตัวละ 8 บิต 2 ตัว รูป 8ข. แสดงรูปแบบของแลทช์ที่ใช้เก็บตัวหาร(divisor latch)



รูป 8ข. แสดงรูปแบบของแลทช์เก็บค่าตัวหาร

การอ้างถึงแลทช์เก็บค่าตัวหารจะอ้างโดยผ่านแอดเดรสของรีจิสเตอร์ 0(สำหรับไบต์ต่ำ) และรีจิสเตอร์ 1(สำหรับไบต์สูง) ซึ่งเป็นแอดเดรสเดียวกับของรีจิสเตอร์ของบัฟเฟอร์ด้านรับ และอินเทอร์รัพต์เอ็นเนเบิลรีจิสเตอร์ ในการตรวจดูว่ารีจิสเตอร์ภายในคู่ใดที่ใช้รับข้อมูล UARTจะตรวจไปที่บิต7ของ LCR ซึ่งเรียกว่า Divisor Latch Access Bit(DLAB) โดยถ้าบิตนี้เป็น 1 แสดงว่าข้อมูลที่เขียนลงรีจิสเตอร์0 และ1ของ UART จะถูกส่งไปที่แลทช์เก็บค่าตัวหาร แต่ถ้า DLAB เป็น 0 ข้อมูลที่เขียนจะถูกส่งไปยัง THR และRBR ตามปกติ

ขั้นตอนในการโหลดแลทช์เก็บค่าตัวหาร เป็นดังนี้

1. เขียน 1 ลงบิต7 ของ LCR
2. เขียนค่าตัวหาร 16 บิตลงรีจิสเตอร์ 0 และ 1 ของ UART
3. เขียน 0 ลงบิต7 ของ LCR

โมเด็มส่วนใหญ่จะถูกเซตให้ใช้ได้กับคำสั่งพื้นฐานทั่วไปสำหรับการสื่อสารแบบอนุกรม เพื่อเป็นการลดจำนวนคำสั่งที่ต้องใช้งานทั้งหมด ซึ่งเราสามารถแบ่งคำสั่งทั้งหมดนี้ได้เป็น 3 กลุ่มย่อยคือ Configuration, Actions และ Dial Command

1. คำสั่งจัดการลักษณะทั่วไป(Configuration Commands)

เป็นคำสั่งที่ใช้เปลี่ยนลักษณะการเชื่อมต่อกับผู้ใช้ของโมเด็มหรือใช้ระบุลักษณะการทำงานของโมเด็ม คำสั่งเหล่านี้จะไม่ทำให้โมเด็มมีการทำงานใดๆเกิดขึ้น แต่อาจจะมีผลให้การกระทำใดๆเกิดขึ้นได้ โดยคำสั่งที่จะกล่าวถึงต่อไปนี้เป็นคำสั่งที่ใช้สำหรับการจัดลักษณะโมเด็มแบบพื้นฐาน

- E : Modem Command Echo : เมื่อโมเด็มอยู่ในโหมดคำสั่ง คำสั่ง E (echo) จะเป็นตัวกำหนดว่าจะให้คำสั่งที่โมเด็มรับมาถูกส่งกลับไปยัง DTE หรือไม่

0 — Command echo disabled (default)

1 — Command echo enabled

- M : Monitor(speaker) Control : เป็นคำสั่งใช้เอ็นเนเบิลหรือดิสเอเบิลลำโพงของเครื่อง ตามความต้องการที่ระบุมา ซึ่งจะอนุญาตให้ผู้ใช้ควบคุมลำโพงตั้งแต่เริ่มหมุนจนถึงการสนทนากัน

0 — Always off

1 — On until call established (delfault)

2 — Always on

- Q : Enable or Disable Modem Response : เป็นคำสั่งที่ใช้ควบคุมว่าจะให้มีการส่งรหัสรับทราบคำสั่ง (result codes acknowledging commands) และมีการรายงานสถานะการเรียกจากโมเด็มไปยังDTEหรือไม่

0 — Modem response enable(default)

1 — Modem response disabled

- V : Result Code Format : คำสั่ง V (verbose) ใช้กำหนดว่าจะให้โมเด็มตอบสนองกลับมาในรูปของตัวเลขหรือตัวอักษร

0 — Use numeric character responses

1 — Use word response

ค่าปกติจะเป็น 1 คือจะตอบสนองเป็นข้อความ โดยผลตอบสนองพื้นฐานที่ใช้กันมีดังนี้

Numeric	Word
0	OK
1	Connect
2	Ring
3	No Carrier
4	Error
6	No Dialing
7	Busy
8	No Answer

- X : Select Result Code Set : ทั่วไปแล้วคำสั่ง X จะทำให้เกิดความเข้ากันได้กับโมเด็มรุ่นก่อนๆ โดยข้อความที่โมเด็มจะส่งไปยัง DTE และการรายงานสภาวะ(โดยโมเด็ม)จะถูกควบคุมโดยคำสั่งดังนี้

0 — Connect

1 — Connect bps

2 — Connect bps, No Dial tone

3 — Connect bps, Busy

4 — Connect bps, Busy, No Dial tone

เมื่อใช้คำสั่ง ATX0 โมเด็มจะทำตัวเป็นเหมือนโมเด็มดั้งเดิมคือจะหมุนหมายเลขโดยไม่สนใจไดออลโทนและสัญญาณ busy และจะมีข้อความ'Connect' ส่งกลับมาเมื่อเชื่อมต่อได้แล้ว

ATX1 จะช่วยเชื่อมต่อโมเด็มพร้อมกับกำหนดอัตราเร็วของข้อมูลที่ใช้ในการติดต่อกันด้วย ตำแหน่ง ส่วน bps ก็คือค่าตัวเลขแสดงอัตราเร็วในการส่งสัญญาณข้อมูล

แต่ถ้าโมเด็มได้รับคำสั่ง ATX2 โมเด็มจะพยายามดีเท็คไดออลโทนก่อนการหมุนหมายเลข ถ้าตรวจไม่พบไดออลโทนภายใน 5 วินาที ข้อความ "No Dial tone" จะถูกส่งกลับมาแทน

คำสั่ง ATX3 ไม่สนใจไดออลโทนแต่จะดีเท็คสัญญาณ busy เพื่อหยุดการเรียก ส่วน ATX4 จะดีเท็คทั้งสองอย่าง

2. คำสั่งสั่งการ(Action Commands)

คำสั่งแบบนี้จะมีผลทันทีต่อสถานะของโมเด็มหรือการเชื่อมต่อกับโมเด็มระยะไกล

- A : Answer an Incoming Call : คำสั่งนี้จะทำให้โมเด็มยกหูขึ้นทันทีและปล่อยโทนตอบรับออกจากโมเด็ม ตามปกติการตอบสนองของโมเด็มต่อสัญญาณเรียกเข้าจะถูกกำหนดด้วย การเซตค่าของรีจิสเตอร์ S0 ถ้าสามารถทำการเชื่อมต่อได้ โมเด็มจะตอบกลับมาว่า 'Connect' แต่ถ้าไม่ได้ก่อนเวลาที่ระบุไว้ในรีจิสเตอร์ S7 โมเด็มจะตอบกลับมาเป็น 'No Carrier' แทน

- H : Switch Hook Control : คำสั่งทำหน้าที่เลียนแบบการยกหูขึ้นและการวางหูของโทรศัพท์ธรรมดา ATH0 จะทำให้โมเด็มหยุดการเชื่อมต่อที่กระทำอยู่แล้ววางหูทันที

0 — Go on-hook (hang up)

1 — Go off-hook (on-line)

- Zn : Soft Reset : (n เป็นหมายเลขconfigurationที่จะทำการรีเซ็ต) คำสั่งนี้จะทำให้โมเด็มวางหู ตัดการเชื่อมต่อทั้งหมดที่ติดต่ออยู่ คำสั่งที่ต่อท้ายในบรรทัดเดียวกันก็จะไม่มีผล

- O : Return to on-line state : คำสั่ง ATO ทำหน้าที่เปลี่ยนโมเด็มจากโหมดคำสั่งให้เป็นโหมดข้อมูล(on-line) และสามารถกลับไปยังโหมดคำสั่งได้โดยใช้อักษรเอสเคป(escape character)

3. คำสั่งหมุน(Dial Commands)

คำสั่ง ATD เป็นคำสั่งที่ทำให้โมเด็มทำการหมุนโทรศัพท์ตามข้อมูลที่อยู่ในสตริงคำสั่งหมุน (dial string) เช่น ATD555-1212 เป็นคำสั่งให้โมเด็มยกหูขึ้น, รอเป็นเวลาเท่ากับที่ระบุไว้ในรีจิสเตอร์S6 แล้วจึงหมุนโทรศัพท์หมายเลข 555-1212(ไม่สนใจ -) หลังจากนั้นโมเด็มก็จะพยายามทำการเชื่อมต่อเป็นระยะเวลาเท่ากับที่ระบุไว้ในรีจิสเตอร์S7

- P : Pulse Dialing : เป็นคำสั่งให้โมเด็มใช้ระบบ make/break ในการหมุนหมายเลขกับโทรศัพท์แบบหมุน

- T : Tone Dialing : เป็นคำสั่งให้โมเด็มหมุนหมายเลขกับโทรศัพท์แบบกด(โดยใช้โทนต่างๆประกอบกัน) คำสั่งTและPนี้จะเป็นการเซตโหมดที่จะใช้ในการหมุนโดยจะอยู่ต่อจากATD เสมอ

S Registers

รีจิสเตอร์ S เป็นหน่วยความจำภายในโมเด็มที่ใช้เก็บข้อมูลเกี่ยวกับพารามิเตอร์ที่ใช้ในการทำงานต่างๆ ซึ่งจะทำให้เราได้รับข้อมูลของโมเด็มและสามารถใช้ข้อมูลเหล่านี้ในการทดสอบโมเด็มได้อีกด้วย

ชุดคำสั่ง AT ประกอบด้วย 2 คำสั่งที่ใช้เพื่อตรวจสอบและจัดการกับรีจิสเตอร์ S รูปแบบคำสั่งที่ใช้เพื่อตรวจสอบค่าปัจจุบันที่เก็บอยู่ในรีจิสเตอร์คือ Sr? โดย r คือหมายเลขรีจิสเตอร์ โมเด็มจะตอบสนองคำสั่งนี้เป็นค่าที่อยู่ในรีจิสเตอร์นั้นในรูปเลขฐานสิบ 3 หลัก ส่วนการกำหนดค่าของรีจิสเตอร์ S สามารถทำได้โดยใช้คำสั่ง Sr = n โดย r คือหมายเลขรีจิสเตอร์และ n คือค่าที่ต้องการเซตให้รีจิสเตอร์(n มีค่าตั้งแต่ 0 ถึง 255) โดยรีจิสเตอร์ S ที่สำคัญๆมีดังนี้

S0 : Ring to Answer On : การกำหนดค่าที่ไม่เป็น 0 ให้รีจิสเตอร์ S0 จะทำให้โมเด็มอยู่ในโหมดตอบรับอัตโนมัติ โดยค่าในรีจิสเตอร์นี้จะเป็นตัวกำหนดจำนวนครั้งของริงที่จะเกิดขึ้นก่อนโมเด็มจะยกหูขึ้น เพื่อตอบรับการเรียกที่เรียกเข้ามา แต่ถ้ากำหนดให้รีจิสเตอร์นี้เป็น 0 ก็จะปิดการตอบรับอัตโนมัติ

S1 : Number of Ring Counter : ขณะที่ริงที่เข้ามาถูกนับ ค่าในS1จะเปลี่ยนไปตามค่านับนั้น และเมื่อค่าในรีจิสเตอร์นี้เท่ากับค่าใน S0 โมเด็มก็จะทำการตอบโทรศัพท์โดยอัตโนมัติ

S6 : Blind Dialing Wait Time : ค่าใน S6 จะเป็นตัวกำหนดจำนวนวินาทีที่โมเด็มจะรอหลังจากยกหูก่อนที่จะทำการหมุนหมายเลขแรกของหมายเลขโทรศัพท์ในคำสั่งหมุน โดยโมเด็มส่วนใหญ่จะเซตค่านี้ไว้ที่ 2 วินาที

S7 : Carrier Wait Time : ค่าใน S7 เป็นค่าที่บอกให้โมเด็มรู้ว่าจะต้องรอเป็นเวลากี่วินาทีหลังการหมุน จึงจะเริ่มสร้างการเชื่อมต่อได้ ถ้าโมเด็มตรวจไม่พบแคร์เรียร์ภายในเวลาที่กำหนดก็จะวางหูและส่งข้อความ "No Carrier" โมเด็มส่วนใหญ่จะตั้งค่าเวลานี้ไว้ที่ 30 วินาที

ภาคผนวก ค.

โปรแกรมระบบจดหมายอิเลคทรอนิคส์

```
/*
        TMail.h

        Definitions for TMail Project's constant and enumeration types.

*/

#ifndef     __TMAIL_H_
#define     __TMAIL_H_

#include <alloc.h>
#include <conio.h>
#include <graphics.h>
#include <dos.h>
#include <dir.h>
#include <io.h>
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <process.h>
#include <d:\mailprj\tmailh.h>

#define MAXMAIL             20      /* Max. Mail in Mailbox   for Show */
#define MAXATTACH           14      /* Max. Attach File shown for Select*/
#define MAXSEND             20      /* Max. Receiver shown for Select */
#define MAXATTACHFILE   5
#define MAXRECEIVER     5

#define FALSE           0
#define TRUE            1

#define MSTARTROW   3       /* START ROW of Mail Window      */
#define MSTARTCOL   2       /* START COLumn of Mail Window   */
#define MSTOPROW    12      /* STOP ROW of Mail Window       */
#define MSTOPCOL    79      /* STOP COLumn of MAil Window    */
#define MROWWIDTH   (MSTOPROW-MSTARTROW+1)      /* ROW Mail Window WIDTH      */
#define MCOLWIDTH   (MSTOPCOL-MSTARTCOL+1)      /* COLumn Mail window WIDTH   */

#define STARTROW    6       /* START ROW of Edit Window      */
#define STARTCOL    5       /* START  COLumn of Edit Window  */
#define STOPROW             21      /* STOP ROW of Edit Window       */
#define STOPCOL             75      /* STOP COLumn of Edit Window    */
```

```
#define ROWWIDTH (STOPROW-STARTROW+1)   /* ROW WIDTH of Edit Window      */
#define COLWIDTH (STOPCOL-STARTCOL+1)   /* COLumn WIDTH of Edit Window   */

#define MAXMENU                 4       /* Max. Theme of Mainmenu        */
#define MAXSUBMENU              4       /* Max. Theme of Submenu         */
#define STARTOFSUBMENU          3       /* Start Row of Submenu          */

/*
#define FOREGROUND    YELLOW
#define BACKGROUND    BLUE */

#define FOREGROUND    MAGENTA
#define BACKGROUND    WHITE

#define MAXOPENFILE   20
#define STRLEN        320


enum Font {
        OLDFONT = 0 , NEWFONT
};

enum Cursor {
        ENGCURSOR = 252, THAICURSOR
};

enum MaskShow  {
        MARKSELECT = 254, MARKDELETE
};

typedef unsigned char     fontpic[256][20];

/*********************************************************      */
/*      Keyboard Scan Code ,they have 2 low byte as 0x00  */
/*********************************************************      */
enum  ScanCode {
        KIns                    = 0x52,
        KDel                    = 0x53,
        KHome                   = 0x47,
        KEnd                    = 0x4F,
        KPgUp                   = 0x49,
        KPgDown                 = 0x51,
        KArrowUp    = 0x48,
        KArrowDown = 0x50,
        KArrowLeft   = 0x4B,
        KArrowRight = 0x4D,
```

```
        KCtrlArrowLeft  = 0x4B,
        KCtrlArrowRight = 0x74,
        KShiftTab     = 0X0F,
        KAltA                   = 0x1E,
        KAltB                   = 0x30,
        KAltC                   = 0x2E,
        KAltD                   = 0x20,
        KAltE                   = 0x12,
        KAltF                   = 0x21,
        KAltG                   = 0x22,
        KAltH                   = 0x23,
        KAltI                   = 0x17,
        KAltJ                   = 0x24,
        KAltK                   = 0x25,
        KAltL                   = 0x26,
        KAltM                   = 0x32,
        KAltN                   = 0x31,
        KAltO                   = 0x18,
        KAltP                   = 0x19,
        KAltQ                   = 0x10,
        KAltR                   = 0x13,
        KAltS                   = 0x1F,
        KAltT                   = 0x14,
        KAltU                   = 0x16,
        KAltV                   = 0x2F,
        KAltW                   = 0x11,
        KAltX                   = 0x2D,
        KAltY                   = 0x15,
        KAltZ                   = 0x2C
};

enum KFunction {
        KF1 = 0x3B, KF2, KF3, KF4, KF5, KF6, KF7, KF8, KF9, KF10
};

enum KAltFunction {
        KAltF1 = 0x68, KAltF2, KAltF3, KAltF4, KAltF5, KAltF6, KALtF7, KALtF8, KAltF9, KAltF10
};

enum KCtrlFunction {
        KCtrlF1 = 0x5E, KCtrlF2, KCtrlF3, KCtrlF4, KCtrlF5, KCtrlF6, KCtrlF7, KCtrlF8, KCtrlF9, KCtrlF10
};

enum KAltNumber {
        KAlt1 = 0x78, KAlt2, KAlt3, KAlt4, KAlt5, KAlt6, KALt7, KAlt9, KAlt0
};
```

```
/*********************************************************/
/*          ASCII Code ,use only 2 low byte as key          */
/*********************************************************/
enum KCtrl { /* Not use KCtrlI equal to TAB          */
        KCtrlA = 0x01,  KCtrlB, KCtrlC, KCtrlD, KCtrlE, KCtrlF, KCtrlG,
        KCtrlH, KCtrlI, KCtrlJ, KCtrlK, KCtrlL, KCtrlM, KCtrlN, KCtrlO,
        KCtrlP, KCtrlQ, KCtrlR, KCtrlS, KCtrlT, KCtrlU, KCtrlV, KCtrlW,
        KCtrlX, KCtrlY, KCtrlZ
};

enum ASCIICode {
        KESC = 0x1B, KBACKSPACE         = 0x08, KTAB = 0x09, KENTER = 0x0D, KSPACEBAR = 0x20
};

typedef struct {
        char        sender[12], subject[35], date[19], flagdel, flagnew;
        unsigned int No_in;
} MAILTYPE;

typedef     union {
        unsigned word_field;
        struct {
                int lowbit :1;
                int another :14;
                int hibit :1;
        } bit_field;
} BITFORM;

typedef     union {
        unsigned  word_field;
        struct {
                unsigned char lowbyte, hibyte;
        } byte_field;
} BLOCKFORM;

unsigned char           P[256][86],PR[256][86];

unsigned char           DeliverColor(unsigned char data, int pos,
        unsigned char foreground, unsigned char background);
int         FindLevel(unsigned char ch);
void        terminate(const char *st);
unsigned char           EngToThai(unsigned char ch);
void        ClearScreen(int left, int top, int right, int bottom, unsigned char color);
int         GetGraphX( void );      /* Get X Position in Graphic Mode    */
```

```
int         GetGraphY(void);          /* Get Y Position in Graphic Mode      */
int         GetGraphMaxX( void );     /* Get Max. Position of X in Graphic Mode          */
int         GetGraphMaxY(void);
struct  viewporttype GetTextViewPort(void);          /* Get Margin of Window in Text Mode */
int         GetTextX( void );         /* Get Position of X in Text Mode      */
int         GetTextY( void );         /* Get Position of Y in Text Mode      */
void        GotoXY(unsigned char x, unsigned char y);
void        GotoXYShift(unsigned char x, unsigned char y, char s);
int         GetTextMaxX( void );      /* Get Max. Position of X in Text Mode */
int         GetTextMaxY(void);
void    Block(int left,int top,int right,int bottom,int thick,unsigned char color);
void    BlockReverse(int left, int top, int right, int bottom);
void        TmailGetFont(char *filename,char f);
void    WriteCharShiftRvs( unsigned char ch,char s,char frvs );
void    WriteChar( unsigned char ch );
void    WriteShiftRvs( unsigned char *st,char s,char frvs );
void    Write( unsigned char *st );
void    Putbkmenu(int left,int top,int right,int bottom);
void    Highlight(char *st,int x,int y,int stopx,char frvs);
int     SelectMenu(void);
int     CountChar( char *st );
void    mmenu(char jump);
void    DosShell(void);
void    MakeMail(void);
void        AddString( void );        /* Allocate Mmory of String at the Last */
void        BeepAlarm( void );        /* Warn about an Error     */
void        ClearChar(int x, int y,char frvs);       /* Clear at Position               */
void        DeleteCharacter( void );
unsigned char           EngToThai(unsigned char ch);        /* change ASCII from Eng to Thai     */
void        ImageGet(int left, int top, int right, int bottom);     /* Get from Screen     */
void        ImageGetH(int left, int top, int right, int bottom);    /* Get from Screen     */
char        GetKey( void );           /* Wait for key Rressing with Twingcle Curcor*/
void        InsertCharacter(char ch);
void        InsertNewLine( void );
int         IsThaiMode( void );                 /* Check Thai Mode or Not          */
int         PosInScreen(char *st, int stringpos);   /* Map Position from String to Screen  */
int         PosInString(char *st, int screenpos);   /* Map Position from Screen to String  */
void        ImagePut(int left, int top); /* Reshow onScreen                */
void        ImagePutH(int left, int top);           /* Reshow onScreen                 */
void        RemoveString( void );
void        ScrollDown( void );
void        ScrollPageDown( void );
void        ScrollPageUp( void );
void        ScrollScreen(int top, int bottom, int totop);
void        ScrollUp( void );
void        ShowScreen(int startrow, int stoprow);
```

```
void        ToggleTypeMode( void );          /* Toggle Eng or Thai Mode        */
void        WriteCharXOR(unsigned char ch);
int         IsThaiLetter( unsigned char ch );
void    DelChar(void);
void    WriteThrough( unsigned char *st );
int     SaveFile( char * filename );
void    ReadFromFile(void);
void    GetInfo(int startx, int stopx, int y, char *Ans, char *msg , char fjmp, char rvs );
void    ShowFile(int left, int top, int right, int bottom, char *path, char *dir );
void    SelectAttach( int left, int top, int right, int bottom );
void    DecrypteFile(char * pathfilename, char * outfile);
void    EncrypteFile( char * pathfilename );
void    TransposeBlock(BLOCKFORM dcrypte[16]);
int     CopyFile( const char *InputFile, const char *OutputFile );
void    CheckMail(void);
void    SendMail( LONG RxID, char * Subject, char * MailFileName, char AttachFileName[14][40], int AttachFileCount, BYTE maillength, char
flagdif, char *serv, char voc );
unsigned ShiftRightAround(unsigned int input, int key);
unsigned ShiftLeftAround( unsigned int input, int key );

char    menu[10][60]={
                    "ส่งจดหมายเสียง          -  Send Voice Mail",
                    "อ่านจดหมายเสียง         -  Read Voice Mail",
                    "ส่งจดหมายผ่านแฟกซ์โมเด็ม    -  Send Mail By FaxModem",
                    "ตรวจดูจดหมายที่ถูกส่งเข้ามาใหม่  -  Check For New Mail",
                    "เขียนจดหมายและส่งจดหมาย    -  Send a Mail Message",
                    "แนบไฟล์ส่งไปพร้อมกับจดหมาย   -  Send File via Mail",
                    "ตรวจดูจดหมายเก่าที่เก็บไว้    -  Browse Mail Message",
                    "เกี่ยวกับอิเลคทรอนิคส์เมล์ภาษาไทย -  About Thai E-MAIL",
                    "กลับสู่ระบบชั่วคราว        -  Dos Shell",
                    "ออกจากโปรแกรม          -  Exit to Dos"
                };
void        AboutTmail(char *username);          /* Detail of this Software and user name */
void        AddString( void );          /* Allocate Mmory of String at the Last */
void        BeepAlarm( void );          /* Warn about an Error    */
void        BlockReverse(int left, int top, int right, int bottom);
//void      ClearChar(int x, int y);    /* Clear at Position                  */
void        ClearChar(int x, int y,char frvs);          /* Clear at Position                  */
void        ClearScreen(int left, int top, int right, int bottom, unsigned char color);          /* Clear Block          */
int         ComposeMail( void );        /* Edit text file for mail letter          */
void CompressMailBox( void );

void        Constructor(void);          /* Initialize All                     */
int         CountChar(char *st);        /* Max. Position on Screen            */
int         CreateFileList(char *path); /* return amount of file in that path */
void        CreateGraph( void );
```

```
int         CreateMenu( void );
void        DecrypteFile(char *pathfilename, char *outfile);
void        DeleteCharacter( void );
unsigned char       DeliverColor(unsigned char data, int pos,
            unsigned char foreground, unsigned char background);
void        Destructor(void);       /* Close All                                   */
/* put message with title of block    */
void        DialogBox(char *st);
void        EncrypteFile(char *pathfilename);
unsigned char       EngToThai(unsigned char ch);        /* change ASCII from Eng to Thai    */
int         FindLevel(unsigned char ch);        /* find level of Thai char   */
void        GetFont(char *filename);  /* retrieve nez font                   */
int         GetGraphX( void );      /* Get X Position in Graphic Mode      */
int         GetGraphY(void);        /* Get Y Position in Graphic Mode      */
int         GetGraphMaxX( void );   /* Get Max. Position of X in Graphic Mode      */
int         GetGraphMaxY(void);     /* Get Max. Position of Y in Graphic Mode      */
void        GetImage(int left, int top, int right, int bottom);     /* Get from Screen     */
char        GetKey( void );         /* Wait for key Rressing with Twingcle Curcor*/
char  * GetString(char *title, int length);  /* Get String that has length byte */
int         GetTextMaxX( void );    /* Get Max. Position of X in Text Mode */
int         GetTextMaxY(void);      /* Get Max. Position of Y in Text Mode */
struct  viewporttype GetTextViewPort(void);         /* Get Margin of Window in Text Mode */
int         GetTextX( void );       /* Get Position of X in Text Mode      */
int         GetTextY( void );       /* Get Position of Y in Text Mode      */
void        GotoXY(unsigned char x, unsigned char y);       /* Goto (x,y) in Text Mode */
void        InitializeMailList( void );
void        InsertCharacter(char ch);
void        InsertNewLine( void );
int         IsThaiLetter(unsigned char ch);     /* Check Thai Letter or Not    */
int         IsThaiMode( void );             /* Check Thai Mode or Not      */
int       *MaxLength(char st[][10], int maxpos);
int         OpenFile(char *filename, int MailID);
void        OpenFont( void );
void        OpenGraph( void );
int         PosInScreen(char *st, int stringpos);   /* Map Position from String to Screen  */
int         PosInString(char *st, int screenpos);   /* Map Position from Screen to String  */
void        PutImage(int left, int top); /* Reshow onScreen             */
void        RemoveString( void );
void        RevokeMemory( void );
int         SaveFile(char *filename);  /* Save File with Encryption        */
void        ScrollDown( void );
void        ScrollPageDown( void );
void        ScrollPageUp( void );
void        ScrollScreen(int top, int bottom, int totop);
void        ScrollUp( void );
int         SelectfromArray(int flagselect);/* Set Margin of Show Window   */
```



```
            WORD    Length;
            BYTE    Function;
            WORD    ObjectType;
```

```
        BYTE    ObjectNameLength;
        BYTE    ObjectName[48];
        LONG    SequenceNumber;
        BYTE    PropertyNameLength;
        BYTE    PropertyName[16];
} scanPropReq;                  /*      Scan Property Requestor         */

typedef struct {
        WORD    Length;
        BYTE    PropertyName[16];
        BYTE    PropertyFlags;
        BYTE    PropertySecurity;
        LONG    SequenceNumber;
        BYTE    PropertyHasValue;
        BYTE    MoreProperties;
} scanPropRep;                  /*      Scan Property Reply             */

typedef struct  {
        WORD    Length;
        BYTE    Function;
        WORD    ObjectType;
        BYTE    ObjectNameLength;
        BYTE    ObjectName[48];
        BYTE    SegmentNumber;
        BYTE    PropertyNameLength;
        BYTE    PropertyName[16];
} readPropValueReq;             /*      Read Property Requestor         */

typedef struct {
        WORD    Length;
        BYTE    PropertyValue[128];
        BYTE    MoreSegments;
        BYTE    PropertyFlags;
} readPropValueRep;             /*      Read Property Reply             */

typedef struct {
        BYTE    ObjectName[48];
        BYTE    ObjectFullName[80];
        LONG    ObjectID;
} AccountFrameType;             /* Object Bindery structer              */

int ScanBinderyObject( scanBindReq *Request, scanBindRep *Reply );
int ScanProperty( scanPropReq *Request, scanPropRep *Reply );
int ReadPropertyValue( readPropValueReq *Request, readPropValueRep *Reply );
int countBindery( void );
int openBindery( void );
```

```
BYTE GetConnectionNumber( void );
char *GetUserName( BYTE ConnectionNumber );
void GetSysTime(void);
int sort_function( const AccountFrameType *a, const AccountFrameType *b );
void SortBindery(void);
int openBindery();
int GetMailID(const char *UserName, LONG UserID );

extern AccountFrameType *AccountData;          /* Global varibale define in TMAILN.C   */
extern WORD             AccountNumber;         /*        get information about bindery  */
extern BYTE  *ServerName[48];
extern WORD     ServerNumber;
extern BYTE  far  *Sname;
#define MaxSubLength    64
#define      MaxMail              20

typedef struct {
        //char      idle[20];
        char    Sendfullname[80];     /* From Sender                              */
        char      Sendname[30];
        BYTE      fNewMail;           /* Mail is new Flag                         */
        BYTE      fNotRead;           /* Mail isn't read Flag                     */
        BYTE      fDelMail;           /* Mail has deleted Flag   */
        time_t    lTime;                    /* System Time                        */
        BYTE      byMailLength;             /* Mail Length in line number         */
        char      MailContentID[13];  /* Mail Content File                        */
        char      szMailSubject[MaxSubLength];    /* Subject in Mail              */
        BYTE      byAttach;           /* Number of Attach file   */
        char      szAttachName[5][13];/*          Attach File   */
        char      szAttachID[5][13];  /* Attach file ID          */
} MailFrameType;

char *MailTime( time_t mailTime );
void CheckMail( void );
LONG  LongSwitch( BYTE *before );
#endif
```

```
/*
        TMailVar.h

        Definitions for TMail Project's global variable.

        Programmer: Kitti Khumronrith       33100020
                    Napat Sra-ium           33100161
                    Telecommunication Engineering' 29 KMITL

        Copyright (C) 1994 by RoboCRAB International
        All Rights Reserved.
*/

#ifndef __TMAILVAR_H_
#define __TMAILVAR_H_

#include "d:\mailprj\TMail.H"
#include "d:\mailprj\TMailN.h"

char        filelist[MAXOPENFILE][13];/* Use in CreateFileList Function      */
int         gxpos, gypos;             /* Graph Position = (0,0,639,479)      */
int         txpos, typos;             /* Text  Position = (1,1,80,24)        */
enum Font   fonttype = OLDFONT;
/*enum Font fonttype = NEWFONT; */
enum Cursor cursortype = THAICURSOR;
unsigned char       P[256][86];   /* Buffer Font of 256 Characters       */
unsigned char       far *VidRam = (unsigned char far *) 0xA0000000UL;

char        screen[5][20] = {     /* Title of each Window */
        " จดหมาย ", " แฟมแฟ้ม ", " ส่งจดหมาย ", " อ่านจดหมาย ", " เขียนจดหมาย "
};

int         upscreen = 0, downscreen = 3;   /* Default Index           */
int         submenupos[MAXMENU], /* Array of Submenu [1..MAXSUBMENU]   */
            submenumax[MAXMENU],          /* Array of Maximun Submenu Position */
            submenulength[MAXMENU],       /* Array of Maximum Length of Submenu String */
            tmenupos[MAXMENU];    /* Position in Text Mode of Started Menu */

int         LASTMAIL = 15;

MAILTYPE maillist[MAXMAIL];               /* buffer of mail detail           */
char        attachlist[MAXATTACH][13];    /* whole file in current directory */
char        sendlist[MAXSEND][25];        /* total mail receiver use account of LAN */

int         flaglistmail = FALSE;         /* List only one File              */
```

```
int         oldflagselect = -1;                 /* Initialize Value in SelectformArray function      */
int         listindex = 1;                      /* Index of Maillist Variable                        */
int         attacharray[MAXATTACHFILE];         /* Array of Index of Attached                        */
int receiverarray[MAXRECEIVER];         /* Array of Index of Send to                         */

char far *sid[2600];                    /* String Index of the 128 Line                      */
int         c_id,                               /* Curent Index to the First String in Screen */
            e_id = 0;                   /* End String Index                                  */
int         menupos = 0;                        /* Menu Position                                     */
int         flagrestoremenu = FALSE;
int         oldfileindex = -1;          /* Save old viewed File                              */
int         savefileindex = 1;          /* Save File Index Use to Name of File .CML */
unsigned char far *ScrBuffPtr, *screen_buffptr;/* Screen Buffer ptr                      */
unsigned int graph_seg = 0xA000;        /* Graphic system segment of VGA                     */

char        *progname;              /* point to argv[0] of main function                 */

unsigned key[32] = {        /* Key for Encryption */
            1, 1, 4, 7, 7,15, 3, 2,13, 4, 2,12, 9,11, 5,10,
            9, 5,10,15, 2, 8, 0,11, 5, 7, 6, 2, 7,11, 4, 6
};



int AttachFlag = 0;
int MailLength = 0;
char SubjectName[81] = "-_-";

extern AccountFrameType *AccountData;
extern WORD             AccountNumber;

MailFrameType           *MailFrame; /* Mail header                                       */
MailFrameType   *VocMailFrame;
int                     AllMail = 0,    /* All Mail in mailbox                           */
                        NotRead = 0,            /* Mail has not been read */
                        NewMail = 0;            /* Mail is new mail                      */

int                     AllVocMail = 0,         /* All Voice Mail in mailbox                         */
                        VocNotRead = 0,         /* Voice Mail has not been read          */
                        NewVocMail = 0;         /* Voice Mail is new mail                */

long LocalID;/* Current user's NetWare Bindery object ID       */

char        LoMail[40],                 /* SYS:MAIL\MAI : Mail content       */
                        LoBox[40],              /* SYS:MAIL\MBG : Mail header        */
                        LoAtt[40];              /* SYS:MAIL\ATT : LoAtt name         */
```

```
#endif
```

```
#include <io.h>
#include <stdio.h>
#include <stdlib.h>
#include <dos.h>
#include <bios.h>
#include <fcntl.h>
#include <mem.h>
#include <time.h>
#include <conio.h>
#define  FALSE  0
#define  TRUE   1
#define  SPRATE 4000
#define  DWORD  unsigned long
#define  WORDS   unsigned int
#define  BYTES   unsigned char
#define  VOCPTR char far*

const char voctool_version[] = "V1.5";
const BYTES voc_breakend      = 0;
const BYTES voc_breaknow      = 1;
const long eachread          = 0x8000;
const long eachwrite         = 0x8000;
  WORDS    voc_status_word     = 0;
  WORDS    voc_err_stat        = 0;
  char     vocfile_header[]    =
                          "Creative Voice File\0x1A\0x1A\0\0x0A\1\0x29\0x11\1";
  BYTES    voc_paused          = 0;
  BYTES    vocdriver_installed  = 0;
  WORDS    vocdriver_version    = 0;
  BYTES    vocfile_headerlength = 0;
  VOCPTR   voc_ptr_to_driver    = 0;

  char huge*buffptr1;
  char huge*buffptr2;
  VOCPTR  buffaddr1;
  VOCPTR  buffaddr2;
  int     filehandle;
  long    filesize;
  WORDS    byte_read;
  WORDS       byte_write ;
  char huge *filepointer;

long     cal_read_times(void);
void     set_headerrd(VOCPTR buffpoint);
void     set_header_block(VOCPTR hbuff);
void     set_new_header(VOCPTR newbuffptr);
```

```
void      set_tailerrd(VOCPTR tailbuff);
VOCPTR    _voc_for_wrbuffer(char *voicefile);
void      write_to_file(VOCPTR bufferpointer);
void      set_tailerwt(VOCPTR bufadr);
void      set_headerwt(int num,VOCPTR bufadr,unsigned int lastwrite);
void              _voc_rec(char far *bufferaddress);
void      _print_voc_errmessage(void);
VOCPTR    _voc_get_buffer(char *voicefile);
WORDS     _voc_getversion(void);
void      _voc_setport(WORDS portnumber);
void      _voc_setirq(WORDS irqnumber);
WORDS     _voc_init_driver(void);
void      _voc_deinstall_driver(void);
void      _voc_set_speaker(BYTES OnOff);
void      _voc_output(char far *bufferaddress);
void      _voc_stop(void);
void      textnumerror();
int       ReadVoiceMail(char *filename);
int       WriteVoiceMail(char *filename);
```

```
#include "d:\mailprj\vocmail.h"
#include <string.h>
#include <graphics.h>
//#include "d:\mailprj\tmail.h"
//#include "d:\mailprj\tmailvar.h"
void _print_voc_errmessage(void)
/* INPUT: None; OUTPUT: None; PURPOSE: Displays SB error as text. */
{
  char msg[40];
  switch (voc_err_stat) {
            case 100 : strcpy(msg," CT-VOICE.DRV driver file not loaded");break;
            case 110 : strcpy(msg," No memory free for driver file ");break;
            case 120 : strcpy(msg," Wrong driver file ");break;
            case 200 : strcpy(msg," VOC file not found ");break;
            case 210 : strcpy(msg," No memory free for VOC file ");break;
            case 220 : strcpy(msg," File is not in VOC format ");break;
            case 300 : strcpy(msg," Memory allocation error occurred ");break;
            case 101 : strcpy(msg," No Sound Blaster card found ");break;
            case 410 : strcpy(msg," Wrong port address used ");break;
            case 420 : strcpy(msg," Wrong interrupt used ");break;
            case 500 : strcpy(msg," No loop in preparation ");break;
            case 510 : strcpy(msg," No sample in the output ");break;
            case 520 : strcpy(msg," No paused sample available ");break;
  }
  setfillstyle(SOLID_FILL,BLUE);
  bar(90,260,450,330);
  Block(90,260,450,330,1,WHITE);
  Block(93,263,447,327,0,WHITE);
  GotoXY(14,15);
  Write(msg);
  getch();
  setfillstyle(SOLID_FILL,BLUE);
  bar(90,260,450,330);
}

VOCPTR _voc_get_buffer(char *voicefile)
/* INPUT: Filename as string; OUTPUT: Pointer to buffer with VOC data;
 * PURPOSE: Loads a file into memory and returns the value pointer to
                        the data upon being successfully loaded. */
{
            WORDS     blocksize;
            WORDS     byte_read ;
            BYTES     check1,check2;
            WORDS     segment1;
            WORDS     segment2;
            long                pos;
```

```
/* VOC file not found */
   if (_dos_open(voicefile,O_RDONLY,&filehandle) != 0)
                {
                 voc_err_stat = 200;
                 return(FALSE);
                }
   filesize = filelength(filehandle);
   blocksize = (eachread + 18L /*header16+tailer2*/+15L) /16;
/* Insufficient memory for VOC file */
   check1 = _dos_allocmem(blocksize,&segment1);
   if (check1 != 0)
                {
                 voc_err_stat = 210;
                 return(FALSE);
                }
   FP_SEG(buffaddr1) = segment1;
   FP_OFF(buffaddr1) = 0;

   check2 = _dos_allocmem(blocksize,&segment2);
   if(check2!=0)
                { voc_err_stat = 211 ;
                     return(FALSE);
                }
   FP_SEG(buffaddr2) = segment2;
   FP_OFF(buffaddr2) = 0;
   buffptr1 = (char huge*)buffaddr1;
   buffptr2 = (char huge*)buffaddr2;
   pos = tell(filehandle);
/* Load sample file */
   if(filesize <= 0x8000)
                { if(pos==0)
                          lseek(filehandle,16,0);
                        _dos_read(filehandle,buffptr1+16,filesize,&byte_read);
                        buffptr1 += byte_read;
                }
   if(filesize > 0x8000)
                { if(pos==0)       /*|-------|*/
                          lseek(filehandle,16,0);  /* | */
                        _dos_read(filehandle,buffptr1+16,0x8000,&byte_read);
                        buffptr1 += byte_read;     /* |-> this num is depend on vocfile format*/
                }                        /* if rec from CREATIVE LAB = 32 */
   set_headerrd(buffaddr1);          /* if rec from record funct in sb16 = 42*/
   set_headerrd(buffaddr2);          /* if rec from C  = 16*/
   set_tailerrd(buffaddr1);
   set_tailerrd(buffaddr2);
/*   do
```

```
                {
                 _dos_read(filehandle,filepointer,0xFFFF,&byte_read);
                 filepointer += byte_read ;
                } while (byte_read == 0xFFFF);
                                        _dos_close(filehandle);
 File not in VOC format
  if ((bufferaddress[0] != 'C') || (bufferaddress[1] != 'r'))
                {
                 voc_err_stat = 220;
                 return(FALSE);
                }
 Load successful */
  voc_err_stat = 0;
  vocfile_headerlength = (BYTES)buffaddr1[20];
  return(buffaddr1);
}

WORDS _voc_getversion(void)
/* INPUT: None; OUTPUT: DRV ver. no.; PURPOSE: Determines DRV ver. no. */
{
  WORDS   vdummy;
  asm {
                mov     bx,0
                call    voc_ptr_to_driver
                mov     vdummy, ax
                }
  return(vdummy);
}

void _voc_setport(WORDS portnumber)
/* INPUT: Port address number; OUTPUT: None;
 * PURPOSE : Sets port address before initialization. */
{
  asm {
                mov   bx,1
                mov   ax,portnumber
                call  voc_ptr_to_driver
                }
}

void _voc_setirq(WORDS irqnumber)
/* INPUT: Interrupt number; OUTPUT: None;
 * PURPOSE: Sets the interrupt number before initialization. */
{
  asm {
                mov   bx,2
```

```
                mov     ax,irqnumber
                call    voc_ptr_to_driver
                }
}
WORDS _voc_init_driver(void)
/* INPUT: None; OUTPUT: Error msg. no., depending on results
 * PURPOSE: Initializes driver software. */
{
                int             filehandle;
                unsigned long   filesize;
                WORDS           blocksize;
                char huge       *filepointer;
                WORDS           byte_read;
                BYTES           check;
                WORDS           segment;
                WORDS           vseg;
                WORDS           vofs;
                WORDS           vout;
                FILE            *fp;
                int             test = FALSE,i=0;
                char            st[80];
  vocdriver_installed = FALSE;
/* Driver file not found */
  if (_dos_open("CT-VOICE.DRV",O_RDONLY,&filehandle) != 0)
                {
                voc_err_stat = 100;
                return(FALSE);
                }
  system("mem /c > c:\\checkcb.mem");
  if( (fp = fopen("c:\\checkcb.mem","rb"))== NULL ){
                voc_err_stat = 101;
                return(FALSE);
  }
  while( !feof(fp) ){
                while( !feof(fp)&&( st[i++]=(char)fgetc(fp) ) != 0x0A );
                st[i]=0;
                i=0;
                if( strstr(st,"CTSB16") ){
                                test=TRUE;
                                break;
                }
  }
  fclose(fp);
  unlink("c:\\checkcb.mem");
  if( test == FALSE ){
                                voc_err_stat = 101;
```

```
                return FALSE;
    }

    filesize = filelength(filehandle);
    blocksize = (filesize + 15L) /16;
  /* Insufficient memory */
    check = _dos_allocmem(blocksize,&segment);
    if (check != 0)
                {
                 voc_err_stat = 110;
                 return(FALSE);
                }
    FP_SEG(voc_ptr_to_driver) = segment;
    FP_OFF(voc_ptr_to_driver) = 0;
    filepointer = (char huge*)voc_ptr_to_driver;
  /* Load driver file */
    do
                {
                 _dos_read(filehandle,filepointer,0x8000,&byte_read);
                 filepointer += byte_read ;
                } while (byte_read == 0x8000) ;
    _dos_close(filehandle) ;
  /* Driver file doesn't start with "CT"? Not a CT-Voice driver */
    if ((voc_ptr_to_driver[3] != 'C') || (voc_ptr_to_driver[4] != 'T'))
                {
                 voc_err_stat = 120;
                 _dos_freemem(segment);
                 return(FALSE);
                }
  /* Determine driver version */
    vocdriver_version = _voc_getversion();
  /* Start driver */
    vseg = FP_SEG(&voc_status_word);
    vofs = FP_OFF(&voc_status_word);
    asm {
                mov     bx,3
                call    voc_ptr_to_driver
                mov     vout,ax
                mov     bx,5
                mov     es,vseg
                mov     di,vofs
                call    voc_ptr_to_driver
                }
  /* No Sound Blaster card found */
    if (vout == 1)
                {
```

```
                voc_err_stat = 400;
                _dos_freemem(segment);
                return(FALSE);
               }
/* Wrong port address used */
  if (vout == 2)
               {
                voc_err_stat = 410;
                _dos_freemem(segment);
                return(FALSE);
               }
/* Wrong interrupt number used */
  if (vout == 3)
               {
                voc_err_stat = 420;
                _dos_freemem(segment);
                return(FALSE);
               }
  vocdriver_installed = TRUE;
  return(TRUE);
}
void _voc_deinstall_driver(void)
/* INPUT: None; OUTPUT: None; PURPOSE: Switches off driver, releases memory. */
{
  if (vocdriver_installed)
               {
                asm {
               mov   bx,9
               call  voc_ptr_to_driver
                }
                _dos_freemem(FP_SEG(voc_ptr_to_driver));
               }
}
void _voc_set_speaker(BYTES on_off)
/* INPUT: TRUE for speaker on, FALSE for speaker off; OUTPUT: None;
 * PURPOSE: Toggles SB card output. */
{
  asm {
               mov      bx,4
               mov      al,on_off
               call     voc_ptr_to_driver
                }
}
void _voc_output(VOCPTR bufferaddress)
/* INPUT: Pointer to sample data; OUTPUT: None; PURPOSE: Plays back sample. */
{
```

```
  WORDS  vseg;
  WORDS  vofs;
  _voc_set_speaker(TRUE);
  vseg = FP_SEG(bufferaddress);
  if((bufferaddress[0] == 'c') || (bufferaddress[1] == 'r'))
              vofs = FP_OFF(bufferaddress) + vocfile_headerlength ;
              else vofs = FP_OFF(bufferaddress);
  asm {
              mov     bx,6
              mov     es,vseg
              mov     di,vofs
              call    voc_ptr_to_driver
              }
}

void _voc_stop(void)
/* INPUT: None; OUTPUT: None; PURPOSE: Stops a sample. */
{
  asm {
              mov     bx,8
              call    voc_ptr_to_driver
              }
}

long cal_read_times(void)
{  long  read_times = 0;

           read_times = (filesize/eachread) + 1L;
           return(read_times);
}



void set_headerrd(VOCPTR buffpoint)
{          long  pow5,b5,b4,b3,b2,b1,b0,b5b4,b3b2,b1b0;

           pow5 = 1048576;
           b5 = eachread/pow5;
           b4 = eachread%pow5/65536L;
           b3 = eachread%pow5%65536L/4096L;
           b2 = eachread%pow5%65536L%4096L/256L;
           b1 = eachread%pow5%65536L%4096L%256L/16L;
           b0 = eachread%pow5%65536L%4096L%256L%16L;
           b5b4 = b5*16L+b4;
           b3b2 = b3*16L+b2;
           b1b0 = b1*16L+b0;
```

```
        _fmemset(buffpoint,9,1);
        _fmemset(buffpoint+1,b1b0,1);
        _fmemset(buffpoint+2,b3b2,1);
        _fmemset(buffpoint+3,b5b4,1);
        set_header_block(buffpoint+4);
}

void set_tailerrd(VOCPTR tailbuff)
{  _fmemset(tailbuff+32785L,0,1);
        _fmemset(tailbuff+32786L,0,1);
}

void set_header_block(VOCPTR hbuff)
{       _fmemset(hbuff,0x88,1);
        _fmemset(hbuff+1,0x13,1);
        _fmemset(hbuff+2,0,1);
        _fmemset(hbuff+3,0,1);
        _fmemset(hbuff+4,0x08,1);
        _fmemset(hbuff+5,0x01,1);
        _fmemset(hbuff+6,0,1);
        _fmemset(hbuff+7,0,1);
        _fmemset(hbuff+8,0,1);
        _fmemset(hbuff+9,0,1);
        _fmemset(hbuff+10,0,1);
        _fmemset(hbuff+11,0,1);
}

void set_new_header(VOCPTR newbuffptr)
{       long more,b0,b1,b2,b3,b3b2,b1b0;

        more = (filesize%eachread) - 4L; /*00 00 00 00 at the bottom of vocfile*/
        b3 = more/4096L;
        b2 = more%4096L/256L;
        b1 = more%4096L%256L/16L;
        b0 = more%4096L%256L%16L;
        b3b2 = b3*16L+b2;
        b1b0 = b1*16L+b0;
        _fmemset(newbuffptr,9,1);
        _fmemset(newbuffptr+1,b1b0,1);
        _fmemset(newbuffptr+2,b3b2,1);
        _fmemset(newbuffptr+3,0,1);
        set_header_block(newbuffptr+4);
}

VOCPTR _voc_for_wrbuffer(char *voicefile)
```

```
{
  WORDS blocksize;
  BYTES check1;
  BYTES check2;
  WORDS segment1;
  WORDS segment2;
/* VOC file not found */
  if (_dos_creat(voicefile,_A_NORMAL,&filehandle) != 0)
              {
               voc_err_stat = 200;
               return(FALSE);
              }
 /*  filesize = 0x6000;*/
  blocksize  = (eachwrite + 20L + 15L) /16;
/* Insufficient memory for VOC file */
  check1 = _dos_allocmem(blocksize,&segment1);
  if (check1 != 0)
               {
               voc_err_stat = 210;
               return(FALSE);
               }
  FP_SEG(buffaddr1) = segment1;
  FP_OFF(buffaddr1) = 0;

  check2 = _dos_allocmem(blocksize,&segment2);
  if(check2 != 0)
               {
               voc_err_stat = 211;
               return(FALSE);
               }
  FP_SEG(buffaddr2) = segment2;
  FP_OFF(buffaddr2) = 0;

  voc_err_stat = 0;
  vocfile_headerlength = (BYTES)buffaddr1[20];
  return(buffaddr1);
}

void set_tailerwt(VOCPTR bufadr)
{   _fmemset(bufadr,0,1);
              _fmemset(bufadr+1,0,1);
              _fmemset(bufadr+2,0,1);
              _fmemset(bufadr+3,0,1);
              _dos_write(filehandle,bufadr,4,&byte_write);
}
```

```
void set_headerwt(int num,VOCPTR bufadr,unsigned int lastwrite)
{    unsigned long  size,pow5,b0,b1,b2,b3,b4,b5,b5b4,b3b2,b1b0;

                pow5 = 1048576;
                size = (num*32768L)+20L - 8L + (unsigned long)lastwrite;
                b5 = size/pow5;
                b4 = size%pow5/65536L;
                b3 = size%pow5%65536L/4096L;
                b2 = size%pow5%65536L%4096L/256L;
                b1 = size%pow5%65536L%4096L%256L/16L;
                b0 = size%pow5%65536L%4096L%256L%16L;
                b5b4 = b5*16L+b4;
                b3b2 = b3*16L+b2;
                b1b0 = b1*16L+b0;
                _fmemset(bufadr,9,1);
                _fmemset(bufadr+1,b1b0,1);
                _fmemset(bufadr+2,b3b2,1);
                _fmemset(bufadr+3,b5b4,1);

                _fmemset(bufadr+4,0x88,1);
                _fmemset(bufadr+5,0x13,1);
                _fmemset(bufadr+6,0,1);
                _fmemset(bufadr+7,0,1);
                _fmemset(bufadr+8,0x08,1);
                _fmemset(bufadr+9,0x01,1);
                _fmemset(bufadr+10,0,1);
                _fmemset(bufadr+11,0,1);
                _fmemset(bufadr+12,0,1);
                _fmemset(bufadr+13,0,1);
                _fmemset(bufadr+14,0,1);
                _fmemset(bufadr+15,0,1);
                lseek(filehandle,0L,SEEK_SET);
                _dos_write(filehandle,bufadr,16,&byte_write);
                bufadr += byte_write;
}


void _voc_rec(VOCPTR bufferaddress)
/* INPUT: Pointer to sample data; OUTPUT: None; PURPOSE: Plays back sample. */
{
  WORDS  vseg;
  WORDS  vofs;
  _voc_set_speaker(TRUE);
  vseg = FP_SEG(bufferaddress);
  vofs = FP_OFF(bufferaddress);
  asm {
```

```
                mov     bx,7
                mov     ax,SPRATE
                mov     es,vseg
                mov     di,vofs
  /*   mov   dx,0x60*/
                mov             cx,32788   /*0x8000*/
                call    voc_ptr_to_driver
                 }
}

void textnumerror()
/* INPUT   : None, the data come from voc_err_stat; OUTPUT: None;
 * PURPOSE : Displays SB error as text; includes error no. and level. */
{
//   printf(errmsg" Error # %d: ",voc_err_stat);
   _print_voc_errmessage();
   //exit(voc_err_stat);
}

int WriteVoiceMail(char *filename)
{
  VOCPTR voc_buffer = 0;
  VOCPTR voc_buffer1 = 0;
  VOCPTR voc_buffer2 = 0;
  VOCPTR voc_buffer_temp = 0;
  char   ch;
  int    times=0,x;
  clock_t start,end;
  float  diff_tm;
  unsigned int num_write_bytes;
  long pos;

  if (_voc_init_driver() == FALSE){
                textnumerror();
                switch( voc_err_stat ){
                        case 100:
                        case 110:
                        case 120:
                        case 400:
                        case 410:
                        case 420:
                        case 101:
                                                break;
                        case 200:
                                                _voc_deinstall_driver();
                                                break;
```

```
                }
                goto  exitwrite;
}
voc_buffer1  = _voc_for_wrbuffer(filename);
voc_buffer2  = buffaddr2;
voc_buffer = voc_buffer1;
if(voc_buffer == NULL){
                textnumerror();
                switch( voc_err_stat ){
                        case 210:
                                                _dos_freemem(FP_SEG(voc_buffer1));
                                                break;
                        case 211:
                                                _dos_freemem(FP_SEG(voc_buffer2));
                                                break;
                }
                _voc_deinstall_driver();
                goto exitwrite;
}
setfillstyle(SOLID_FILL,BLUE);
bar(78,231,490,284);
GotoXYShift( 16,13,4 );
WriteShiftRvs("Press 'r' key for record or anykey for exit: ",4,0);
ch = getche();
printf("\n\n");
switch (ch) {
                case 'r':
                case 'R':
                        bar(78,231,490,284);
                        GotoXYShift( 16,13,4 );
                        WriteShiftRvs("Press a key to stop record the sound...",4,0);
                        do{start = 0;
                                pos = tell(filehandle);
                                if(pos==0)
                                lseek(filehandle,16L,SEEK_SET);
                                _voc_rec(voc_buffer);
                                start = clock();
                                times = times+1;
                                if(times > 1)
                                 {
                                if(voc_buffer==voc_buffer2)
                                  _dos_write(filehandle,voc_buffer1+16,0x8000,&byte_write);
                                if(voc_buffer==voc_buffer1)
                                  _dos_write(filehandle,voc_buffer2+16,0x8000,&byte_write);
                                  }
                        /*if value of status word is 0,a sample isn't being processed*/
```

```
                    if(voc_buffer==voc_buffer1)
                      voc_buffer_temp = voc_buffer2;
                    if(voc_buffer==voc_buffer2)
                      voc_buffer_temp = voc_buffer1;
                    voc_buffer = voc_buffer_temp;
                    do{}while((voc_status_word != 0)&&( !(x=kbhit()) ));
                }while(x==0);
                    end = clock();
                    diff_tm = (end - start)/CLK_TCK;
                    //printf("times is %f\n",diff_tm);
                    num_write_bytes = (unsigned int)((diff_tm*32788)/6.593407);
                    if(voc_buffer==voc_buffer1)
                    voc_buffer_temp = voc_buffer2;
                    if(voc_buffer==voc_buffer2)
                    voc_buffer_temp = voc_buffer1;
                    voc_buffer = voc_buffer_temp;
                    _dos_write(filehandle,voc_buffer+16,num_write_bytes-16,&byte_write);

                    set_tailerwt(voc_buffer);
                    set_headerwt(times-1,voc_buffer,num_write_bytes-16);
                    _dos_close(filehandle);
                  break;
            default :
                  break;
            }
/* Free VOC file memory */
  _dos_freemem(FP_SEG(voc_buffer1));
  _dos_freemem(FP_SEG(voc_buffer1));
  _voc_deinstall_driver();
  return 0;
exitwrite:
  return 1;
}

int ReadVoiceMail(char * filename)
{

  VOCPTR voc_buffer = 0;
  VOCPTR voc_buffer1 = 0;
  VOCPTR voc_buffer2 = 0;
  VOCPTR voc_buffer_temp = 0;
  char ch;
  long position = 0;
  WORDS seg;
  WORDS off;
  long times_for_read = 0;
```

```
if (_voc_init_driver() == FALSE) {
                        textnumerror();
                        switch( voc_err_stat ){
                          case 120:
                          case 400:
                          case 410:
                          case 420:
                          case 100:
                          case 110:
                          case 101:
                                                break;
                          case 200:
                                                _voc_deinstall_driver();
                                                break;
                        }
                        goto exitread;
}
voc_buffer1 = _voc_get_buffer(filename);
voc_buffer2 = buffaddr2;
voc_buffer = voc_buffer1;
if(voc_buffer == NULL){
          textnumerror();
          switch( voc_err_stat ){
                    case 210:
                                                _dos_freemem(FP_SEG(voc_buffer1));
                                                break;
                    case 211:
                                                _dos_freemem(FP_SEG(voc_buffer1));
                                                break;
          }
          _voc_deinstall_driver();
          goto exitread;
}
times_for_read = cal_read_times();
setfillstyle(SOLID_FILL,BLUE);
bar(90,260,450,330);
Block(90,260,450,330,1,WHITE);
Block(93,263,447,327,0,WHITE);
GotoXY(14,15);
Write("(p) for playback or ant key for exit ");
ch = getch();
switch(ch) {
          case 'p':
          case 'P':
                                setfillstyle(SOLID_FILL,BLUE);
```

```
bar(90,260,450,330);
Block(90,260,450,330,1,WHITE);
Block(93,263,447,327,0,WHITE);
GotoXY(14,15);
Write("Press any key to stop this pretty sound...");
if(filesize > 0x8000)
 {
        do{
                _voc_output(voc_buffer);
                if(voc_buffer==voc_buffer1)
                _dos_read(filehandle,voc_buffer2+16,0x8000,&byte_read);
                if(voc_buffer==voc_buffer2)
                _dos_read(filehandle,voc_buffer1+16,0x8000,&byte_read);
                times_for_read = times_for_read - 1;

                do{} while((voc_status_word != 0)&&(kbhit()==0)) ;
                if(kbhit()!=0)
                 { _voc_stop();
                        break;
                 }
                if(voc_buffer==voc_buffer1)
                        voc_buffer_temp = voc_buffer2;
                if(voc_buffer==voc_buffer2)
                        voc_buffer_temp = voc_buffer1;
                voc_buffer = voc_buffer_temp;
                if(times_for_read==1)
                        set_new_header(voc_buffer);

                position = tell(filehandle);
                }while((times_for_read != 1)&&(1!=eof(filehandle)));
                _voc_output(voc_buffer);
                do{}while((voc_status_word != 0)&&(kbhit()==0));
                if(kbhit()!=0)
                        { _voc_stop();
                          break;
                         }
 }

if(filesize <= 0x8000)
 {  _voc_output(voc_buffer) ;
        do{}while((voc_status_word!=0)&&(kbhit()==0));
        if(kbhit() != 0)
                { _voc_stop();
                  break;
                 }
 }
```

```
                                        setfillstyle(SOLID_FILL,BLUE);
                                        bar(90,260,450,330);
                                        _dos_close(filehandle);

                                        break;
                case 'x':break;
                                }
                _dos_freemem(FP_SEG(buffaddr1));
                _dos_freemem(FP_SEG(buffaddr2));
                _voc_deinstall_driver();
                return 0;
exitread:
}
```

```
#include "D:\mailpr\tmain.h"
#define USER        0x0100
#define FILE_SERVER 0x0400

//extern AccountFrameType              *AccountData;
//extern WORD              AccountNumber;
struct connectionIDTableType{
            BYTE  SlotInUse;
            BYTE  ServerOrder;
            BYTE  Network[4];
            BYTE  Node[6];
            BYTE  Socket[2];
            BYTE  ReceiveTimeout[2];
            BYTE  RouterNode[6];
            BYTE  PacketSequence;
            BYTE  ConnectionNumber;
            BYTE  ConnectionStatus;
            BYTE  MaxTimeout[2];
            BYTE  Reserved[5];
      };

BYTE far *GetConnectionID( void );
BYTE far *GetFileServerName( void );
void ShowConnectionTable( struct connectionIDTableType far *Table );
void ShowNameTable( BYTE far *serverNameTable );
void PrintNum( BYTE *num, int count );
void FarPrintNum( BYTE far *num, int count );
int  AttachToFileServer( BYTE freeSlot );
BYTE GetPreferredConnectionID( void );
void Attach( char *Server );
void SetPreferredConnectionID( BYTE newConn );
void ScanServer( void );
int  LoginToFileServer( WORD type, char *name, char *password );
int ScanBinderyObject( scanBindReq *Request, scanBindRep *Reply )
{
            union REGS  regs;
            struct SREGS sregs;

            Request->Function = 0x37;
            Request->Length          = sizeof( scanBindReq ) - 2;
            Reply->Length            = sizeof( scanBindRep ) - 2;

            sregs.ds  = FP_SEG((void far *) Request);
            regs.x.si = FP_OFF((void far *) Request);
            sregs.es  = FP_SEG((void far *) Reply);
            regs.x.di = FP_OFF((void far *) Reply);
```

```
        regs.h.ah = 0xE3;
        int86x(0x21,&regs,&regs,&sregs);
        return regs.h.al;
}

int ScanProperty( scanPropReq *Request, scanPropRep *Reply )
{
        union REGS  regs;
        struct SREGS sregs;

        Request->Function = 0x3C;
        Request->Length          = sizeof( scanPropReq ) - 2;
        Reply->Length            = sizeof( scanPropRep ) - 2;

        sregs.ds  = FP_SEG((void far *) Request);
        regs.x.si = FP_OFF((void far *) Request);
        sregs.es  = FP_SEG((void far *) Reply);
        regs.x.di = FP_OFF((void far *) Reply);

        regs.h.ah = 0xE3;
        int86x(0x21,&regs,&regs,&sregs);
        return regs.h.al;
}

int ReadPropertyValue( readPropValueReq *Request, readPropValueRep *Reply )
{
        union REGS  regs;
        struct SREGS sregs;

        Request->Function = 0x3D;
        Request->Length          = sizeof( readPropValueReq ) - 2;
        Reply->Length            = sizeof( readPropValueRep ) - 2;

        sregs.ds  = FP_SEG((void far *) Request);
        regs.x.si = FP_OFF((void far *) Request);
        sregs.es  = FP_SEG((void far *) Reply);
        regs.x.di = FP_OFF((void far *) Reply);

        regs.h.ah = 0xE3;
        int86x(0x21,&regs,&regs,&sregs);
        return regs.h.al;
}

int countBindery(void)
```

```
{
        scanBindReq ScanBindRequest;
        scanBindRep ScanBindReply;
        register int count=0;
    volatile int result;

        memset( &ScanBindRequest, 0, sizeof( ScanBindRequest ) );
        memset( &ScanBindReply,   0, sizeof( ScanBindReply ) );

        ScanBindRequest.LastObjectID        = 0xFFFFFFFFUL;
        ScanBindRequest.ObjectType          = 0x0100;
        ScanBindRequest.ObjectName[0]       = '*';
        ScanBindRequest.ObjectNameLength = 1;

        while((result=ScanBinderyObject( &ScanBindRequest, &ScanBindReply )) != 0xFC ) {
                if( result ) {
                        return 0;
                }
                count++;
                ScanBindRequest.LastObjectID = ScanBindReply.ObjectID;
        }
        return count;
}

BYTE GetConnectionNumber(void)
{
        _AX = 0xDC00;
        geninterrupt(0x21);
        return _AL;
}

char *GetUserName( BYTE ConnectionNumber )
{
        union REGS  regs;
        struct SREGS        sregs;

        struct  {
                WORD    Length;
                BYTE    RequestType;
                BYTE    ConnectionNumber;
        } RequestStruct;

        struct  {
                WORD    Length;
                BYTE    ObjectID[4];
                WORD    ObjectType;
```

```
        BYTE    ObjectName[48];
        BYTE    LoginTime[7];
        WORD    Reserved;
    } ResponseStruct;

    RequestStruct.Length = sizeof(RequestStruct) - sizeof(WORD);
    RequestStruct.RequestType = 0x16;
    RequestStruct.ConnectionNumber = ConnectionNumber;

    ResponseStruct.Length = sizeof(ResponseStruct)-sizeof(WORD);

    sregs.ds  = FP_SEG((void far *) &RequestStruct);
    regs.x.si = FP_OFF((void far *) &RequestStruct);
    sregs.es  = FP_SEG((void far *) &ResponseStruct);
    regs.x.di = FP_OFF((void far *) &ResponseStruct);

    regs.h.ah = 0xE3;
    int86x(0x21, &regs, &regs, &sregs);
    return (regs.h.al) ? "" : ResponseStruct.ObjectName;
}

void GetSysTime(void)
{
    union REGS  regs;
    struct SREGS        sregs;

    struct {
        BYTE year, month, day, hour, minute, second, dayofweek;
    } reply;
    struct time timevar;
    struct date datevar;

    sregs.ds  = FP_SEG((void far *) &reply);
    regs.x.dx = FP_OFF((void far *) &reply);

    regs.h.ah = 0xE7;
    int86x(0x21,&regs,&regs,&sregs);

    timevar.ti_hour=reply.hour;
    timevar.ti_min=reply.minute;
    timevar.ti_sec=reply.second;
    timevar.ti_hund=0;
    settime(&timevar);

    datevar.da_year=1900+reply.year;
    datevar.da_mon=reply.month;
```

```
        datevar.da_day=reply.day;
        setdate(&datevar);
}

int sort_function( const AccountFrameType *a, const AccountFrameType *b )
{
        return( strcmp(a->ObjectName, b->ObjectName) );
}

void SortBindery(void)
{
        qsort( AccountData, AccountNumber, sizeof(AccountFrameType), sort_function );
}

int openBindery( void )
{
        scanBindReq ScanBindRequest;
        scanBindRep ScanBindReply;
        scanPropReq ScanPropRequest;
        scanPropRep ScanPropReply;
        readPropValueReq ReadPropRequest;
        readPropValueRep ReadPropReply;
        int             count=0;

        memset( &ScanBindRequest, 0, sizeof( ScanBindRequest ) );
        memset( &ScanBindReply,   0, sizeof( ScanBindReply ) );
        memset( &ScanPropRequest, 0, sizeof( ScanPropRequest ) );
        memset( &ScanPropReply,   0, sizeof( ScanPropReply ) );
        memset( &ReadPropRequest, 0, sizeof( ReadPropRequest ) );
        memset( &ReadPropReply,   0, sizeof( ReadPropReply ) );

        ScanBindRequest.LastObjectID = 0xFFFFFFFFUL;
        ScanBindRequest.ObjectType   = 0x0100;              /* User */

        AccountNumber = countBindery();
        if((AccountData = (AccountFrameType *) farmalloc(AccountNumber*sizeof(AccountFrameType)))==NULL)
                return 0;
        memset( AccountData, 0, AccountNumber*sizeof(AccountFrameType) );

        ScanBindRequest.ObjectName[0] = '*';
        ScanBindRequest.ObjectNameLength = 1;

        while(ScanBinderyObject( &ScanBindRequest, &ScanBindReply ) != 0xFC ) {
                AccountData[count].ObjectID = LongSwitch(&ScanBindReply.ObjectID);// & 0x0000FFFF;
                strcpy(AccountData[count].ObjectName,ScanBindReply.ObjectName);
                if ( ScanBindReply.ObjectHasProperties ) {
```

```
                                ScanPropRequest.ObjectType = ScanBindReply.ObjectType;
                                ScanPropRequest.ObjectNameLength = sizeof( ScanPropRequest.ObjectName );
                                strcpy( ScanPropRequest.ObjectName, ScanBindReply.ObjectName );
                                ScanPropRequest.SequenceNumber = 0xFFFFFFFFUL;
                                strcpy(ScanPropRequest.PropertyName,"IDENTIFICATION");
//                              strcpy(ScanPropRequest.PropertyName,"*");
                                ScanPropRequest.PropertyNameLength = strlen(ScanPropRequest.PropertyName);
                                ScanProperty( &ScanPropRequest, &ScanPropReply );
                                ReadPropRequest.ObjectType = ScanBindReply.ObjectType;
                                ReadPropRequest.ObjectNameLength = sizeof( ReadPropRequest.ObjectName );
                                strcpy( ReadPropRequest.ObjectName, ScanBindReply.ObjectName );
                                ReadPropRequest.SegmentNumber = 1;
                                ReadPropRequest.PropertyNameLength = strlen( ScanPropReply.PropertyName );
                                movmem( &ScanPropReply.PropertyName, &ReadPropRequest.PropertyName,
                                        ReadPropRequest.PropertyNameLength );
                                strcpy(AccountData[count].ObjectFullName,
                                (ReadPropertyValue( &ReadPropRequest, &ReadPropReply ) != 0) ? "":
                                        ReadPropReply.PropertyValue );
                        }

                        ScanBindRequest.LastObjectID = ScanBindReply.ObjectID;
                        count++;
                }

        SortBindery();
        return 1;
}

int GetMailID(const char *UserName, LONG UserID )
{
        register int    i;

        if(UserName[0] == '\0') {
                for(i=0; i<AccountNumber; i++)
                        if(UserID == AccountData[i].ObjectID)
                                return i;
        } else {
                for(i=0; i<AccountNumber; i++)
                        if(!strcmp(UserName,AccountData[i].ObjectName))
                                return i;
        }
        return -1;
}

char *MailTime( time_t mailTime )
{
```

```
        static char    temp[19];
        static char *MName[12] = { "ม.ค.", "ก.พ.","มี.ค.","เม.ย.","พ.ค.","มิ.ย.",
                "ก.ค.","ส.ค.","ก.ย.","ต.ค.","พ.ย.","ธ.ค." };
        struct tm*timetm;

        timetm = localtime( &mailTime );
        sprintf(temp, "%d %s %d %02d:%02d",
                timetm->tm_mday, MName[timetm->tm_mon],
                timetm->tm_year-57, timetm->tm_hour, timetm->tm_min);
        return temp;
}

LONG  LongSwitch( BYTE *before )
{
    BYTE  after[4];

    after[0] = before[3];
        after[1] = before[2];
        after[2] = before[1];
        after[3] = before[0];

        return *( (LONG *)after );
}
void Attach( char *Server )
{
//    readPropValueReq ReadPropRequest;
//    readPropValueRep ReadPropReply;
//        struct connectionIDTableType far *connectionIDTable, far *tablePtr;
        BYTE  far *serverNameTable, far *serverTablePtr, far *iNetAddr;
//        int    result, i, j, freeSlot = 0;
//    BYTE   oldConnID;


/*    memset( &ReadPropRequest, NULL, sizeof( ReadPropRequest ) );
        memset( &ReadPropReply, NULL, sizeof( ReadPropReply ) );

        ReadPropRequest.ObjectType = FILE_SERVER;
        strcpy( ReadPropRequest.ObjectName, strupr( Server ) );
        ReadPropRequest.ObjectNameLength = sizeof( ReadPropRequest.ObjectName );
        ReadPropRequest.SegmentNumber = 1;
        strcpy( ReadPropRequest.PropertyName, "NET_ADDRESS" );
        ReadPropRequest.PropertyNameLength = 11;
        ReadPropRequest.Function = 0x3D;
//    ReadPropRequest->Length   = sizeof(struct readPropValueReq)-2;
        ReadPropRequest.Length   = sizeof( ReadPropRequest.ObjectName ) + 11 + 6;
        ReadPropReply.Length     = sizeof( readPropValueRep )-2;
```

```
            result = ReadPropertyValue( &ReadPropRequest, &ReadPropReply );
            if( result ){
            printf("Error reading property value of NET_ADDRESS on %s, code = %X\n", Server,result);
            exit(1);
            }
            else{
            printf("Server %s Found. Address = ", Server );
            PrintNum( ReadPropReply.PropertyValue, 4 );
            printf(":");
            PrintNum( ReadPropReply.PropertyValue+4, 6 );
            printf(":");
            PrintNum( ReadPropReply.PropertyValue+10,2 );
            printf("\n");
            }
*/
//          printf("WORKSTATION TABLES UPON ENTRY TO PROGRAM:\n");
//          connectionIDTable = GetConnectionID();
//          ShowConnectionTable( connectionIDTable );
            serverNameTable = GetFileServerName();
            ShowNameTable(serverNameTable);
            //Sname = GetFileServerName();
            //ShowNameTable(Sname);
/*  printf("press any key to setup register for attach to %s...",Server );
            getch();

    tablePtr = connectionIDTable;
    for( i = 0; i < 8; i++, tablePtr++ ){
            iNetAddr = tablePtr->Network;
            for( j = 0; j < 12; j++ )
              if( iNetAddr[j] != ReadPropReply.PropertyValue[j] )
                break;
            if( j == 12 ){
              printf("\nCannot attach, already connected to %s.\n",Server );
              exit(1);
            }
            if( !freeSlot )
              if( !tablePtr->SlotInUse )
                freeSlot = i;
    }

    if( !freeSlot ){
            printf("\n All 8 server slots are in use, cannot attach.\n");
            exit(1);
    }
```

```
serverTablePtr = serverNameTable + ( freeSlot*48 );
tablePtr = connectionIDTable + freeSlot;
tablePtr->SlotInUse = 0xFF;
tablePtr->ServerOrder = freeSlot + 1;

iNetAddr = tablePtr->Network;
for( i = 0; i <12; i++ )
        iNetAddr[i] = ReadPropReply.PropertyValue[i];
      i=0;
do
          serverTablePtr[i] = Server[i];
while( Server[i++] );

printf("\nWORKSTASION ADDRESS JUST BEFORE ATTACH CALL:\n");
ShowConnectionTable( connectionIDTable );
ShowNameTable( serverNameTable );

printf("press any key to attempt to attach to %s..\n",Server );
getch();

result = AttachToFileServer( freeSlot+1 );
switch( result ){
        case 0x00:  printf("Successfully attach to fileserver %s\n",Server );
                    break;
        case 0xF8:  printf("Already attach to fileserver %s\n", Server );
                    break;
        case 0xF9:  printf("Server has no more free connection slot\n");
                    exit(1);
        case 0xfA:  printf("No more server slot available\n");
                    exit(1);
        case 0xFC:  printf("File server unknown\n");
                    exit(1);
        case 0xFE:  printf("Server bindery locked\n");
                    exit(1);
        case 0xFF:  printf("No response from server (illegal server address)\n");
                    exit(1);
        default :   printf("Error attaching to server, code = %X\n",result );
}

printf("WORKSTATION TABLES AFTER ATTACH CALL:\n");
ShowConnectionTable( connectionIDTable );
ShowNameTable( serverNameTable );
//   oldConnID = GetPreferredConnectionID();

SetPreferredConnectionID( freeSlot+1 );
```

```
//    SetPreferredConnectionID( oldConnID );
*/
    return 0;
}

/*
int ReadPropValue( struct readPropValueReq *Request, struct readPropValueRep *Reply )
{
 //   Request->Function = 0x3D;
   // Request->Length   = sizeof(struct readPropValueReq)-2;
    //Reply->Length     = sizeof(struct readPropValueRep)-2;

    _SI = (unsigned)Request;
    _DI = (unsigned)Reply;
    _ES = _DS;
    _AH = 0xE3;

    geninterrupt( 0x21 );
    return _AL;
}
*/

BYTE  far *GetConnectionID( void )
{
    _AH = 0xEF;
    _AL = 3;
    geninterrupt( 0x21 );
    return MK_FP(_ES,_SI);
}


BYTE far *GetFileServerName( void )
{
    _AH = 0xEF;
            _AL = 4;
    geninterrupt( 0x21 );
    return MK_FP(_ES,_SI);
}

void ShowConnectionTable( struct connectionIDTableType far *Table )
{
    int    i;
    struct connectionIDTableType far *T = Table;

    printf("\n        Workstation connection ID table\n");
```

```
            printf("InUse Order Net      Node          Socket Timeout   Router    Seq Conn Status\n");
            for( i = 0; i < 8; i++, T++ ){
            printf("%02X    %02X    ",T->SlotInUse,T->ServerOrder );
            FarPrintNum( T->Network, 4 );  printf(" ");
            FarPrintNum( T->Node, 6 );     printf(" ");
            FarPrintNum( T->Socket, 2 );   printf("   ");
            FarPrintNum( T->ReceiveTimeout, 2 );  printf("  ");
            FarPrintNum( T->RouterNode, 6 );  printf("  ");
            printf("%02X  %02X   %02X\n",T->PacketSequence, T->ConnectionNumber,T->ConnectionStatus);
    }
}


void ShowNameTable( BYTE far *serverNameTable )
{

    int   i;

            Sname = serverNameTable;
/*
            printf("\n     Workstation File Server Name Table\n");
            printf("Slot   Server Name\n");
            for( i = 0; i < 8; i++ )
            printf("%1d          %Fs\n",i,Sname+48*i);
*/
}

void PrintNum( BYTE *num, int count )
{

    int   i;
    for( i = 0; i < count; i++ )
            printf("%02X",num[i]);
}

void FarPrintNum( BYTE far *num, int count )
{

    int   i;
    for( i = 0; i < count; i++ )
            printf("%02X",num[i]);
}

int  AttachToFileServer( BYTE freeSlot )
{
```

```
    _AH = 0xF1;
    _AL = 0;
    _DL = freeSlot;

    geninterrupt( 0x21);

    return _AL;
}

BYTE GetPreferredConnectionID( void )
{
    _AH = 0xF0;
    _AL = 1;

    geninterrupt( 0x21 );

    return _AL;
}

void SetPreferredConnectionID( BYTE newConn )
{
    _AH = 0xF0;
    _AL = 0;
    _DL = newConn;

    geninterrupt( 0x21 );

    return _AL;
}

void  ScanServer( void )
{
        scanBindReq ScanBindRequest;
        scanBindRep ScanBindReply;
        register int i=0;
        volatile int result;

        memset( &ScanBindRequest, 0, sizeof( ScanBindRequest ) );
        memset( &ScanBindReply,   0, sizeof( ScanBindReply ) );

        ServerNumber = 0;
        ScanBindRequest.LastObjectID        = 0xFFFFFFFFUL;
        ScanBindRequest.ObjectType          = 0x0400;
        ScanBindRequest.ObjectName[0]       = '*';
        ScanBindRequest.Obj
```

```
        while((result=ScanBinderyObject( &ScanBindRequest, &ScanBindReply )) != 0xFC ) {
                if( result ) {
                        terminate("Error in scanning file server");
                }
                if( ( ServerName[i]=(BYTE *)farmalloc(48) )==NULL )
                   terminate("Error in alloc memory for server");
                ScanBindRequest.LastObjectID = ScanBindReply.ObjectID;
                strcpy( (char *)ServerName[i++],(char *)(ScanBindReply.ObjectName) );
                ServerNumber++;
        }
}
int  LoginToFileServer( WORD type, char *name, char *password )
{
    BYTE  Request[181];
    WORD   Reply=0;

 //   Request[0] = 0xB3;
    Request[1] = 0;
    Request[2] = 0x14;
    *( (WORD *) (Request+3) ) = type;
    Request[5] = strlen( name );
    strcpy( Request+6, strupr(name) );
    Request[ 6 + Request[5] ] = strlen(password);
    strcpy(Request+6+Request[5]+1, strupr(password) );
    Request[0] = 5+strlen(password)+strlen(name);

    _SI = (unsigned)Request;
    _DI = (unsigned)&Reply;
    _ES = _DS;
    _AH = 0xE3;

    geninterrupt( 0x21 );
    return _AL;
}
```

```
#include <bios.h>
#include <alloc.h>
#include <conio.h>
#include <graphics.h>
#include <dos.h>
#include <dir.h>
#include <io.h>
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <ctype.h>
#include <process.h>
#include <time.h>
#include <fcntl.h>
#include <sys\stat.h>
#include <d:\mailpr\tmail.h>
#include <d:\mailpr\tmailn.h>
#include <d:\mailpr\tmailvar.h>

#define FALSE           0
#define TRUE            1
#define STARTROW        6           /* START ROW of Edit Window        */
#define STARTCOL        5           /* START  COLumn of Edit Window    */
#define STOPROW                     21          /* STOP ROW of Edit Window         */
#define STOPCOL                     75          /* STOP COLumn of Edit Window      */
#define ROWWIDTH (STOPROW-STARTROW+1)   /* ROW WIDTH of Edit Window        */
#define COLWIDTH (STOPCOL-STARTCOL+1)   /* COLumn WIDTH of Edit Window     */
#define STRLEN                      320
#define FOREGROUNDR  MAGENTA
#define BACKGROUNDR  WHITE
#define FOREGROUND   WHITE
#define BACKGROUND   BLUE
//#define MAXATTACH      14

#define MaxSubLength    64
#define     MaxMail                 20

AccountFrameType        *AccountData;
WORD        AccountNumber;
BYTE  *ServerName[48];
WORD     ServerNumber;
BYTE  LocalName[48];
BYTE  LocalFullName[80];
BYTE  far *Sname;

char far *sid[2600];                . /* String Index of the 128 Line                */
```

```
int  lflag;

unsigned char far *ScrBuffPtr, *hlpbuff;
int     attcount = 0;
char    attachfile[MAXATTACH][40];
int     recvcount= 0;
char    RecvName[MAXRECEIVER][20];

void   ShowAtt(int mailno);
void   DialogBox(char *st);
int    SelectRecv(void);
void   ChNewMail(int flag);
void   CheckReadMail(int flag);
int    ViewMails( int mailno );
void   WriteBlock(int left, int top, int right, int bottom, char flagdouble, char flagbottom, char *title);
int    OpenFiles(char * filename, int MailID);
void   About(char *username);
void   IDToLowHi( LONG ID, char *pathID );
void   ReCheckMail(void);
void   SwapImage(int left, int top, int right, int bottom,char *file);
void   RestoreImage(int left, int top, int right, int bottom,char *file,char del);
int    functionsort( const void *a, const void *b );
int    functionsort( const void *a, const void *b);
void   Warning( char *msg );
void   SaveBindery(void);
void   GetBindery(void);
void   strcopy(char *str1,char *str2,char *str3);
void   strtoupr(char *str);
void   Putbackground(void);
void   WriteVoice(void);
void   CheckVocMail(void);
void   ReadVoice();
void   CompressVocMailBox( void );
void   MakeFaxModemMail(void);

int         FindLevel( unsigned char ch )
{
            switch (ch) {
            case 231:  /*  */
            case 232:  /*  */
            case 233:  /*  */
            case 234:  /*  */
            case 235:  /*  */
            case 236:  /*  */
                        return 3;
            case 209:  /*  */
```

```
		case 212:  /* ◌ั */
		case 213:  /* ◌ิ */
		case 214:  /* ◌ี */
		case 215:  /* ◌ึ */
				return 2;
		case 216:  /* ◌ื */
		case 217:  /* ◌ุ */
		case ENGCURSOR:
		case THAICURSOR:
		case MARKSELECT:
		case MARKDELETE:
				return 0;
		case 218:
		case 219:
		case 237:
		case 238:  /* other not use */
				return - 1;
		default: return 1;
		}
}

void TmailGetFont( char * filename,char font)
{
		FILE		* fp;
		register int	i;

		if ( (fp= fopen(filename, "rb")) == NULL )
				terminate("\nOpen Font File Error");
		for (i = 0; i < 256; i++)
				if(!font)
				  fread(P[i], 86, 1, fp);	/* imagesize(0,0,7,19) == 86		*/
				else
				  fread(PR[i], 86, 1, fp);
		fclose(fp);
}

unsigned char		DeliverColor(unsigned char data, int pos,
		unsigned char foreground, unsigned char background)
{
		if ((data & (128 >> pos)) > 0) /* if bit set return foreground */
				return foreground;
		else
				return background;
}
void terminate(const char *st)
{
```

```
        register int i;
        if (e_id != 0) {
          for (i = e_id; i > 0; i--)
                RemoveString();
        }
        for( i = ServerNumber-1; i >= 0; i-- )
                farfree( ServerName[i] );
        closegraph();
        perror(st);
        exit(1);
}
unsigned char           EngToThai( unsigned char ch )
{
        static unsigned char    _EngToThai_[] = {
                 32,  44,  46, 242, 243, 244, 163, 167, 246, 247, 245, 249, 193, 162, 227, 189,
                168, 197,  47,  45, 192, 182, 216, 214, 164, 181, 171, 199, 178, 170, 204, 198,
                241, 196, 218, 169, 175, 174, 226, 172, 231, 179, 235, 201, 200,  63, 236, 207,
                173, 240, 177, 166, 184, 234, 206,  34,  41, 237,  40, 186,  96, 197, 217, 248,
                 32, 191, 212, 225, 161, 211, 180, 224, 233, 195, 232, 210, 202, 183, 215, 185,
                194, 230, 190, 203, 208, 213, 205, 228, 187, 209, 188, 176, 229,  44, 165,  32
        };
        return ((ch < 32)  ? ch : _EngToThai_[ch-32]);
}
void ClearScreen(int left, int top, int right, int bottom, unsigned char color)
{
        struct fillsettingstype fs;

        left = (left - 1) * 8;
        right = right * 8 - 1; /* change coordinate to graphic mode */
        top = (top - 1) * 20;
        bottom = bottom * 20 - 1;

        getfillsettings(&fs);
        setfillstyle(SOLID_FILL, color); /* set color of bar */
        bar(left, top, right, bottom);
        setfillstyle(fs.pattern, fs.color); /* pop old status*/
}

void GotoXY( unsigned char x, unsigned char y )
{
        struct viewporttype vp;

        vp = GetTextViewPort();
        x = x + vp.left - 1;
        y = y + vp.top - 1;
        x = ( x > vp.right )        ? vp.right : x;
```

```
        x = ( x < vp.left )     ? vp.left  : x;
        y = ( y > vp.bottom )   ? vp.bottom : y;
        y = ( y < vp.top )      ? vp.top : y;

        gxpos = (x - 1) * 8;
        gypos = (y - 1) * 20;
        txpos = x;
        typos = y;
}
void GotoXYShift( unsigned char x, unsigned char y, char s )
{
        struct viewporttype vp;

        vp = GetTextViewPort();
        x = x + vp.left - 1;
        y = y + vp.top - 1;
        x = ( x > vp.right )    ? vp.right : x;
        x = ( x < vp.left )     ? vp.left  : x;
        y = ( y > vp.bottom )   ? vp.bottom : y;
        y = ( y < vp.top )      ? vp.top : y;

        gxpos = (x - 1) * 8;
        gypos = (y - 1) * 20+s;
        txpos = x;
        typos = y;
}

struct viewporttype GetTextViewPort()
{
        struct viewporttype vp;

        getviewsettings(&vp);
        vp.left = vp.left / 8 + 1;
        vp.right = (vp.right + 1) / 8;
        vp.top = vp.top / 20 + 1;
        vp.bottom = (vp.bottom + 1) / 20;
        return vp;
}



void SetTextViewPort(int left, int top, int right, int bottom)
{
        left = (left - 1) * 8;
        right = right * 8 - 1;
        top = (top - 1) * 20;
```

```
        bottom = bottom * 20 - 1;
        setviewport(left, top, right, bottom, 1);
}

int GetGraphX()
{
        struct viewporttype vp;

        getviewsettings(&vp);
        return(gxpos - vp.left);
}
int     GetGraphY()
{
        struct viewporttype vp;

        getviewsettings(&vp);
        return(gypos - vp.top);
}
int     GetGraphMaxX()
{
        struct viewporttype vp;

        getviewsettings(&vp);
        return(vp.right - vp.left + 1);
}
int     GetGraphMaxY()
{
        struct viewporttype vp;

        getviewsettings(&vp);
        return(vp.bottom - vp.top + 1);
}

void WriteCharShiftRvs( unsigned char ch,char s,char frvs )
{
        if ( FindLevel(ch) != 1 ) {
                if (GetTextX() == 0)  /* if the first position of line */
                        if( !frvs )
                           putimage(GetGraphMaxX() - 8, GetGraphY() - 20, P[ch],XOR_PUT);
                        else
                           putimage(GetGraphMaxX() - 8, GetGraphY() - 20, PR[ch],XOR_PUT);
                else
                        if( !frvs )
                           putimage(GetGraphX() - 8, GetGraphY(), P[ch],XOR_PUT);
                        else
                           putimage(GetGraphX() - 8, GetGraphY(), PR[ch],XOR_PUT);
```

```
        } else {        /* write normal level character */
                        if( !frvs )
                            putimage(GetGraphX(), GetGraphY(), P[ch], COPY_PUT);
                        else
                            putimage(GetGraphX(), GetGraphY(), PR[ch], COPY_PUT);
                        if (GetTextX() == COLWIDTH)

                                GotoXYShift(1, GetTextY() + 1,s);
                        else
                                GotoXYShift(GetTextX() + 1, GetTextY(),s);
        }
}
void WriteChar( unsigned char ch )
{
        if ( FindLevel(ch) != 1 ) {
                        if (GetTextX() == 1)  /* if the first position of line */
                                putimage(GetGraphMaxX() - 8, GetGraphY() - 20, P[ch],XOR_PUT);
                        else
                                putimage(GetGraphX() - 8, GetGraphY(), P[ch],XOR_PUT);
        } else {        /* write normal level character */
                        putimage(GetGraphX(), GetGraphY(), P[ch], COPY_PUT);
                        if (GetTextX() == COLWIDTH)
                                GotoXY(1, GetTextY() + 1);
                        else
                                GotoXY(GetTextX() + 1, GetTextY());
        }
}
int             GetTextMaxX()
{
        struct viewporttype vp;

        vp = GetTextViewPort();
        return(vp.right - vp.left + 1);
}
int             GetTextMaxY()
{
        struct viewporttype vp;

        vp = GetTextViewPort();
        return(vp.bottom - vp.top + 1);
}

void Write( unsigned char *st )
{
        while (*st)
                        WriteChar(*st++);
```

```
}

void WriteThrough( unsigned char *st )
{
        int find = FALSE;
        while (*st){
          if(  ( GetTextX()==COLWIDTH )&&( GetTextY()==GetTextMaxY() )&&( FindLevel(*st)==1 ) )
                find = TRUE;
                WriteChar(*st++);
          if( find == TRUE ){
                if( *st && ( FindLevel(*st) != 1 ) ){
                  putimage(GetGraphMaxX()-8,GetGraphMaxY()-20,P[*st],XOR_PUT);
                        st++;

                        if( *st && ( FindLevel(*st) != 1 ) ){
                        putimage(GetGraphMaxX()-8,GetGraphMaxY()-20,P[*st],XOR_PUT);
                        st++;
                        }
                        if( *st ){
                        c_id++;
                        ScrollScreen(2,GetTextMaxY(),1);
                        ClearScreen(1,GetTextMaxY(),COLWIDTH,GetTextMaxY(),BACKGROUND);
                        GotoXY(1,GetTextMaxY());
                        }
                }
          }
        }
}
void WriteShiftRvs( unsigned char *st,char s,char frvs )
{
        while (*st)
                WriteCharShiftRvs(*st++,s,frvs);
}
int GetTextX()
{
        struct viewporttype vp;

        vp = GetTextViewPort();
        return(bxpos - vp.left + 1);
}
int GetTextY()
{
        struct viewporttype vp;

        vp = GetTextViewPort();
```

```
        return(typos - vp.top + 1);
}
void mmenu(char jump)
{
        register int i;
        int x,y;
        if(jump)
                goto jumptohere;
        SetTextViewPort(1,1,80,24);
        Putbackground();
        setfillstyle(SOLID_FILL,DARKGRAY);
        bar(73,24,573,51);
        setfillstyle(SOLID_FILL,BACKGROUND);
        bar(65,16,565,43);
        Block(66,17,564,42,0,WHITE);
        GotoXYShift(10,2,0);
        WriteShiftRvs("<<  โครงงานจดหมายอิเลคทรอนิกส์ภาษาไทย - THAI ELECTRONIC MAIL  >>",0,0);
        setfillstyle(SOLID_FILL,DARKGRAY);
        bar(76,116,564,417);
jumptohere:
        SetTextViewPort(9,6,69,20);
        ClearScreen(1,1,61,15,BACKGROUND);
        Block(4,9,482,292,1,WHITE);
        Block(7,12,479,289,0,WHITE);
        GotoXY(20,1);
        Write("เมนูหลัก-Main Menu");
        for(i=0; i<10; i++){
         GotoXY(5,i+4);
         Write(menu[i]);
        }
        GotoXY( 3, 15 );
        Write("F1-ช่วยเหลือ");
        GotoXY( 17, 15 );
    Write("ArrowUp,Down-เลื่อนแถบเลือก");
        GotoXY( 46,15 );
    Write("ENTER-เลือก");
}
void Block(int left,int top,int right,int bottom,int thick,unsigned char color)
{
                                        setfillstyle(SOLID_FILL,color);
                                        bar(left,top,right,top+thick);
                                        bar(left,bottom-thick,right,bottom);
                                        bar(left,top,left+thick,bottom);
                                        bar(right-thick,top,right,bottom);
}
void BlockReverse(int left, int top, int right, int bottom)
```

```
{
        unsigned char           far *bitimage;

        left = (left - 1) * 8;
        right = right * 8 - 1;
        top  = (top - 1) * 20;
        bottom = bottom * 20 - 1;
        if ((bitimage = (unsigned char far *) farmalloc(imagesize(left, top, right, bottom))) == NULL)
                terminate("\nMemory allocation fail in BlockReverse");
        getimage(left, top, right, bottom, bitimage);
        putimage(left, top, bitimage, NOT_PUT);
        farfree(bitimage);
}
void Putbkmenu(int left,int top,int right,int bottom)
{
                        register int i,j;

                        for(i = top; i <= bottom; i++)
    for(j = left; j <= right ;j++){
                        GotoXY(j,i);
                        putimage(gxpos,gypos,P[250],COPY_PUT);
    }
}
void  Highlight(char *st,int startx,int y,int stopx,char frvs)
{
                        GotoXY(startx,y);
                        WriteCharShiftRvs(' ',0,frvs);
                        WriteShiftRvs(st,0,frvs);
                        while(GetTextX() < stopx)
                    WriteCharShiftRvs(' ',0,frvs);
}

int         CountChar( char *st )
{
        register int    count = 0;

        while ( *st )
                if (FindLevel(*st++) == 1)
                        count++;
        return count;
}
int SelectMenu(void)
{
        unsigned char pos=0;
        time_t tim;
        struct tm *tm;
```

```
        char timestr[9];
        Highlight(menu[pos],4,4+pos,59,1);
        while(1){

          if( kbhit() ){
                switch( getch()){
                  case KArrowUp :
                                        Highlight(menu[pos],4,4+pos,59,0);
                                        pos=( (pos==0) ? 9 : pos-1 );
                                        Highlight(menu[pos],4,4+pos,59,1);
                                        break;
                  case KArrowDown :
                                        Highlight(menu[pos],4,4+pos,59,0);
                                        pos = ( (pos==9) ? 0 : pos+1 );
                                        Highlight(menu[pos],4,4+pos,59,1);
                                        break;
                  case KESC :
                                        return 9;
                  case KENTER :
                                        Highlight(menu[pos],4,4+pos,59,0);
                                        return pos;
                  default :
                                        break;
                }
          }
                time(&tim);
          tm = localtime(&tim);
          sprintf(timestr,"%02d:%02d:%02d",tm->tm_hour,tm->tm_min,tm->tm_sec);
          GotoXY(48,1);
          Write(timestr);
        }
}

void DosShell()
{
        int             gd = VGA, gm = VGAHI;

        closegraph();
                system("command.com/k prompt[SHELL FROM MAIL]$P$G");
        initgraph(&gd, &gm, "");
        if (graphresult() != grOk)
                terminate("\nTMail can't initialize graphics system.");

        mmenu(0);
}
```

```
void DelChar(void)
{
    int posx, posy,start;
    int index, count, len;
    int i, j;
    char temp[STRLEN];

    posx = GetTextX();
    posy = GetTextY();
    index= c_id+posy-1;
    if (GetTextX() <= CountChar(sid[index])) {
            strcpy(temp, sid[index]);
            start = PosInString(temp, posx);
            j = 0;
            for (i = PosInString(temp, posx + 1); i <= strlen(temp); i++)
                    sid[index][start++] = temp[j++] = temp[i];
            ClearScreen(posx,posy,COLWIDTH,posy,BACKGROUND);
            GotoXY(posx,posy);
            if( CountChar(sid[index]) > COLWIDTH )
                    temp[PosInString(temp,COLWIDTH-posx+2)] = '\0';
            WriteThrough(temp);
            GotoXY(posx,posy);
            iflag = 3;
    }
    else{
            if(index == e_id)
                    BeepAlarm();
            else{
                    if( posy < ROWWIDTH ){
                     ClearScreen(1,posy+1,COLWIDTH,posy+1,BACKGROUND);
                     WriteThrough(sid[index+1]);
                    }

                    GotoXY(posx,posy);
                    if( (CountChar(sid[index]) + CountChar(sid[index+1])) <= COLWIDTH ){
                        strcat(sid[index],sid[index+1]);
                        for(i = index + 2; i <= e_id; i++)
                                strcpy(sid[i-1],sid[i]);
                        if( posy < ROWWIDTH )
                                ClearScreen(1,posy+1,COLWIDTH,GetTextMaxY(),BACKGROUND);
                        sid[e_id][0] = '\0';
                        RemoveString();
                        if( posy < ROWWIDTH )
                                ShowScreen(posy+1,GetTextMaxY());
                        GotoXY(posx,posy);
```

```
                }
                else{
                    len = strlen(sid[index]);
                    strcat(sid[index],sid[index+1]);
                    sid[index][PosInString(sid[index],COLWIDTH+1)] = '\0';
                    count = strlen(sid[index+1]);
                    for(i = start = strlen(sid[index])-len; i <= count; i++)
                            sid[index+1][i-start] = sid[index+1][i];
                    if( posy >= ROWWIDTH ){
                            GotoXY(1,posy);
                            WriteThrough( sid[index] );
                            GotoXY( posx, posy );
                    }
                }
                flag = 1;
            }
        }
}

void InsertCharacter(char ch)
{
    int   index, posx, posy, posxx, posyy;
    int   level, len;
    int   posinstr,i,j;
    char  temp[STRLEN] = "";

    posx = GetTextX();
    posy = GetTextY();
    index = posy+c_id-1;
    level = FindLevel(ch);

    posinstr = PosInString( sid[ index ],posx );
    if( level != 1 ){
            if( ( posx == 1 )&&( index != 1 ) ){
                    if( CountChar(sid[index-1]) <= COLWIDTH ){
                            if( ( !IsThaiLetter(sid[index-1][strlen(sid[index-1])-1]) )&&( FindLevel(sid[index-1][strlen(sid[index-1])-1])
== 1 ) ){
                                BeepAlarm();
                                return;
                            }
                            if( level == FindLevel(sid[index-1][strlen(sid[index-1])-1]) ){
                                WriteChar(sid[index-1][strlen(sid[index-1])-1]);
```

```
                    sid[index-1][strlen(sid[index-1])-1] = ch;
                    if( posy !=1 )
                            WriteChar(ch);
                    return;
                }

                if( ( level == FindLevel(sid[index-1][strlen(sid[index-1])-2]) )&&( FindLevel(sid[index-1][strlen(sid[index-
1])-1]) != 1 ) ){
                    WriteChar(sid[index-1][strlen(sid[index-1])-2]);
                    sid[index-1][strlen(sid[index-1])-2] = ch;
                    if( posy != 1 )
                            WriteChar(ch);
                    return;
                }

                sid[index-1][len = strlen(sid[index-1])] = ch;
                sid[index-1][len+1] = '\0';
                if( posy != 1 )
                    WriteChar(ch);
                return;
            }
            else{
                BeepAlarm();
                return;
            }
        }

        if( ( (posx == 1)&&(index == 1) )||( !IsThaiLetter(sid[index][posinstr-1] )&&( FindLevel(sid[index][posinstr-1]) == 1 ) )){
            BeepAlarm();
            return;
        }

        if( level == FindLevel(sid[index][posinstr-1]) ){
            WriteChar(sid[index][posinstr-1]);
            sid[index][posinstr-1] = ch;
            WriteChar(ch);
            return;
        }

        if( ( level == FindLevel(sid[index][posinstr-2]) )&&( FindLevel(sid[index][posinstr-1]) != 1 ) ){
            WriteChar(sid[index][posinstr-2]);
            sid[index][posinstr-2] = ch;
            WriteChar(ch);
            return;
        }
```

```
                for( i = strlen(sid[index]); i >= posinstr; i-- )
                        sid[index][i+1] = sid[index][i];
                sid[index][posinstr] = ch;
                WriteChar(ch);
                return;

        }
        else{
                for( i = strlen(sid[index]); i >= posinstr; i-- )
                        sid[index][i+1] = sid[index][i];
                sid[index][posinstr] = ch;
        }
        WriteChar(ch);
        iflag = 3;

        posxx = GetTextX();
        posyy = GetTextY();

        if( ( posxx < posx ) ){
                AddString();
                for( i = e_id; i > index; i-- )
                        strcpy(sid[i+1],sid[i]);
                if( strlen(sid[index]) > PosInString(sid[index],COLWIDTH+1)-1 ){
                        j = 0;
                        for( i =  PosInString(sid[index],COLWIDTH+1); sid[index][i] != '\0'; i++ )
                           sid[index+1][j++] = sid[index][i];
                        sid[index+1][j] = '\0';
                        sid[index][PosInString(sid[index],COLWIDTH+1)] = '\0';
                }
                if( posy ==ROWWIDTH ){
                        c_id++;
                        ScrollScreen(2,posy-1,1);
                        ShowScreen(posy-1,posy);
                        GotoXY(posxx,posyy);
                        iflag = 2;
                        return;
                }
                else{
                        ScrollScreen(posy+1,GetTextMaxY()-1,posy+2);
                        for( i = PosInString(sid[index+1],posxx); (sid[index+1][i] != '\0')&&( i < PosInString(sid[index+1],COLWIDTH+1) );
i++)
                           WriteChar(sid[index+1][i]);
                        GotoXY(posxx,posyy);
                        iflag =1;
                        return;
                }
```

```
        }

        j = 0;
        for( i = PosInString(sid[index],posxx); (sid[index][i] != '\0')&&( (CountChar(sid[index]) <= COLWIDTH) ? TRUE : ( i < PosInString(sid
[index],COLWIDTH+1) ) ); i++)
                temp[j++] = sid[index][i];
        WriteThrough(temp);
        GotoXY(posxx,posyy);

}



void InsertNewLine(void)
{

    int  posx, posy;
    int  index, start;
    int  i, j;

    posx = GetTextX();
    posy = GetTextY();
    index = posy + c_id - 1;

    AddString();
    for(i = e_id; i > index; i--)
            strcpy(sid[i+1],sid[i]);
    start = PosInString(sid[index],posx);
    for(i = start,j = 0; sid[index][i] != '\0'; i++,j++)
            sid[index+1][j] = sid[index][i];
    sid[index+1][j] = '\0';
    sid[index][start] = '\0';
    if( posy ==ROWWIDTH ){
            ScrollScreen(2,posy-1,1);
            c_id++;
            ShowScreen(posy-1,posy);
            GotoXY(1,posy);
            lflag = 2;
    }
    else{
            ScrollScreen(posy+1,GetTextMaxY()-1,posy+2);
            ShowScreen(posy,posy+1);
            GotoXY(1,posy+1);
            lflag = 4;
    }
```

```
}

void DeleteCharacter(void)
{
    int  posx, posy;
    int  index, posinstr;
    int  i, j;
    int  count;
    char  temp[STRLEN], delch;
    int  find = FALSE;

    posx = GetTextX();
    posy = GetTextY();
    index = posy + c_id - 1;
    posinstr = PosInString(sid[index],posx);

    if( posx != 1 ){
            delch = sid[index][posinstr-1];
            for( i = posinstr; sid[index][i] != '\0'; i++ )
              sid[index][i-1] = temp[i-posinstr] = sid[index][i];
            sid[index][i-1] = temp[i-posinstr] = '\0';
            if( FindLevel(delch) == 1 ){
              ClearScreen(posx-1,posy,COLWIDTH,posy,BACKGROUND);
              GotoXY(posx-1,posy);
              if( CountChar(sid[index]) > COLWIDTH )
                      temp[PosInString(temp,COLWIDTH-posx+3)] = '\0';
              WriteThrough(temp);
            }
            else{
              WriteChar(delch);
            }

            if( FindLevel(delch) == 1 )
               GotoXY(posx-1,posy);
            flag = 3;
    }
    else{
            if( index == 1 ){
               BeepAlarm();
               return;
            }
            strcpy(temp,sid[index]);
            count = CountChar(sid[index-1]);
            if( count > COLWIDTH ){
               j = 0;
               for( i = PosInString(sid[index-1],COLWIDTH+1); sid[index-1][i] != '\0'; i++ )
```

```
                    sid[index][j++] = sid[index-1][i];
                sid[index][j] = '\0';
                sid[index-1][PosInString(sid[index-1],COLWIDTH+1)] = '\0';
                index++;
                count = CountChar(sid[index-1]);
                find = TRUE;
            }
            strcat(sid[index-1],temp);
            if( !find ){
                for( i = index+1; i <= e_id; i++ )
                    strcpy(sid[i-1],sid[i]);
                sid[e_id][0] = '\0';
                RemoveString();
            }
            if( posy != 1 ){
                if( !find ){
                    ScrollScreen(posy+1,GetTextMaxY(),posy);
                    ShowScreen(GetTextMaxY(),GetTextMaxY());
                    ShowScreen(posy-1,posy-1);
                    GotoXY(count+1,posy-1);
                }
                else{
                    ShowScreen(posy,posy);
                    GotoXY(count+1,posy);
                }
            }
            else{
                if( !find )
                    c_id--;
                ShowScreen(1,1);
                GotoXY(count+1,1);
            }
            flag = 1;
        }

    }


    void RemoveString()
    {
            farfree(sid[e_id--]);
    }

    void AddString()
    {
            if ( (sid[++e_id] = (char far * )farmalloc(STRLEN)) == NULL ){
```

```
                terminate("\nMemory Allocation Error in AddString");
        }

        side[e_id][0] = '\0';           /* initialize new string      */
}

char    GetKey()
{
        char        ch = 0;

        while (!kbhit())            {
                WriteCharXOR(cursortype);
                delay(100);
                ch++;
        }
        if (ch % 2 == 1) { /* clear cursor if shows */
                WriteCharXOR(cursortype);
        }
        return getch();
}


void WriteCharXOR( unsigned char ch )
{
        if ( ch >= ENGCURSOR  ) {
                putimage(GetGraphX(), GetGraphY(), P[ch], XOR_PUT);
/*              if (GetTextX() == COLWIDTH)
                        GotoXY(1, GetTextY() + 1);
                else
                        GotoXY(GetTextX() + 1, GetTextY());  */
        }
}

void ToggleTypeMode()
{
        cursortype = (cursortype == THAICURSOR) ? ENGCURSOR : THAICURSOR;
}

void ImageGet(int left, int top, int right, int bottom)
{
        left  = (left - 1) * 8;
        right = (right * 8) - 1;
        top   = (top - 1) * 20;
        bottom = (bottom * 20) - 1;

        if ((ScrBuffPtr = (unsigned char far * )farmalloc(imagesize(left, top, right, bottom))) == NULL){
```

```
                terminate("Error in ImageGet");
        }
        getimage(left, top, right, bottom, ScrBuffPtr);
}

void ImagePut(int left, int top)
{
        left = (left - 1) * 8;
        top = (top - 1) * 20;
        putimage(left, top, ScrBuffPtr, COPY_PUT);
        farfree(ScrBuffPtr);
}
void ImageGetH(int left, int top, int right, int bottom)
{
        left  = (left - 1) * 8;
        right = (right * 8) - 1;
        top   = (top - 1) * 20;
        bottom = (bottom * 20) - 1;

        if ((hlpbuff = (unsigned char far * )farmalloc(imagesize(left, top, right, bottom))) == NULL){
                terminate("Error in ImageGet");
        }
        getimage(left, top, right, bottom, hlpbuff);
}

void ImagePutH(int left, int top)
{
        left = (left - 1) * 8;
        top = (top - 1) * 20;
        putimage(left, top, hlpbuff, COPY_PUT);
        farfree(hlpbuff);
}

void ClearChar(int x, int y,char frvs)
{
        struct fillsettingstype fs;

        getfillsettings(&fs);
        ClearScreen(x, y, x, y, (frvs) ? BACKGROUNDR : BACKGROUND);
}

int     PosInString( char *st, int screenpos )
{
        register int    i = 0, count = 0;

        while (count != screenpos)
```

```
                        if (FindLevel(st[i++]) == 1)
                                count++;
        return (i - 1);
}

int     PosInScreen( char *st, int position )
{
        register int    i = 0, j = 0;

        while (i <= position) {
                if (FindLevel(st[i]) == 1)
                        j++;
                i++;
        }
        return (j);
}

void BeepAlarm()
{
        //putchar('\007'); /* sound */
        sound(250);
        delay(80);
        nosound();
        sound(350);
        delay(120);
        nosound();
}

void ScrollUp()
{
        int     oldx;

        oldx = GetTextX();
        c_id--;
        ScrollScreen(1,ROWWIDTH - 1, 2);        /* insert line and show only one */
        ShowScreen(1, 1);               /* string at the first line      */
        GotoXY(oldx, 1);
}

void ScrollScreen(int top, int bottom, int totop)
{
        int upper, lower;

        if( totop < top ){
            lower = ( top+7 >= bottom ) ? bottom : top+7;
            ImageGet(1, top, COLWIDTH, lower);
```

```
                ImagePut(1, totop);
                if( lower != bottom ){
                        upper = lower + 1;
                        lower = ( upper+7 >= bottom ) ? bottom : upper+7;
                        ImageGet(1, upper, COLWIDTH, lower);
                        ImagePut(1, totop+8);
                }
                if( lower != bottom ){
                        upper = lower + 1;
                        ImageGet(1, upper, COLWIDTH, bottom);
                        ImagePut(1, totop+16);
                }
                ClearScreen(1, bottom-top+totop+1, COLWIDTH, bottom, BACKGROUND);
        }
        if( totop > top ){
                upper = ( bottom - 7 <= top ) ? top : bottom-7;
                ImageGet(1, upper, COLWIDTH, bottom);
                ImagePut(1, upper+totop-top);
                if( upper != top ){
                        lower = upper - 1;
                        upper = ( lower - 7 <= top ) ? top :  lower-7;
                        ImageGet(1, upper, COLWIDTH, lower);
                        ImagePut(1, upper+totop-top);
                }
                if( upper != top ){
                        lower = upper - 1;
                        upper = ( lower - 7 <= top ) ? top :  lower-7;
                        ImageGet(1, upper, COLWIDTH, lower);
                        ImagePut(1, upper+totop-top);
                }
                ClearScreen(1, top, COLWIDTH, totop-1, BACKGROUND);
        }
}

void ShowScreen( int startrow, int stoprow )
{
        int             currentrow, id;
        static char     temp[STRLEN];

        currentrow = startrow;
        id = c_id;
        if (stoprow == ROWWIDTH)  /* if stopped row equals to ROWWIDTH then delete the last line */
                ClearScreen(1,ROWWIDTH, COLWIDTH,ROWWIDTH, BACKGROUND);

        while (currentrow <= stoprow && id <= e_id-startrow+1) {   /* show string on screen */
                GotoXY(1, currentrow++);                /* from started line to    */
```

```
                ClearScreen(GetTextX(), GetTextY(), COLWIDTH, GetTextY(), BACKGROUND);
                if (CountChar(sid[id+startrow-1]) > COLWIDTH) { /* stopped line */
                        strcpy(temp, sid[id+startrow-1]);  /* if string length more than */
                        temp[PosInString(sid[id+startrow-1],COLWIDTH+1)] = '\0';       /* COLWIDTH cut only COLWIDTH
*/
                        WriteThrough(temp);
                } else
                        WriteThrough(sid[id+startrow-1]);
                id++;
        }
}

void ScrollDown()
{
        int     oldx;

        oldx = GetTextX();
        c_id++;
        ScrollScreen(2,ROWWIDTH, 1); /* delete the first line , screen */
        GotoXY(1, ROWWIDTH);            /* scroll up and show string     */
        ShowScreen(ROWWIDTH, ROWWIDTH);   /* at the last line            */
        GotoXY(oldx, ROWWIDTH);
}

void ScrollPageUp()
{
        int     maxy, posx, posy;

        maxy =ROWWIDTH;
        posx = GetTextX();
        posy = GetTextY();
        c_id -= maxy - 1;
        if (c_id < 1)
                c_id = 1;
        ClearScreen(1, 1, COLWIDTH, maxy, BACKGROUND);
        ShowScreen(1, maxy);
        GotoXY(posx, posy);
}

void ScrollPageDown()
{
        int     maxy, posx, posy;

        maxy =ROWWIDTH;
        posx = GetTextX();
        posy = GetTextY();
```

```
        c_id += maxy - 1;
        ClearScreen(1, 1, COLWIDTH, maxy, BACKGROUND);
        ShowScreen(1, maxy);
        if( (e_id-c_id<2*(maxy-1)) && (posy > e_id-c_id+1) )
          GotoXY(posx,e_id-c_id+1);
        else
          GotoXY(posx, posy);
}

int     IsThaiMode()
{
        return (cursortype == THAICURSOR);
}

int     IsThaiLetter( unsigned char ch )
{
        return (ch >= 161 && ch <= 206);        /* ch in [ก..ฺ] */
}

int SaveFile( char * filename )
{
        FILE        * fp;
        int handle,r;
        register int   i;
        fp = fopen( filename,"wb+" );
        fclose(fp);
        if ((handle = _open(filename, O_RDWR )) == -1){
                exit(1);
        }
        else {
                for (i = 1; i <= e_id; i++) {           /* write data from array  */
                    if( sid[i][( r = strlen(sid[i]) )-1] != '\n' ){
                            sid[i][r] = '\n';
                            sid[i][r+1] = '\0';
                    }
                    r=_write(handle, sid[i], strlen(sid[i]));
                }
                if (_close(handle) != 0) /* if error of save file exit */
                        terminate("Can't save file");
        }
        return e_id;
}
```

```
void ReadFromFile(void)
{

    int   posx,posy;
    int   i,j,index;
    int   olde_id;
    FILE  *fp;
    char  path[80];
    char  dir[MAXPATH];
    char  far *eid[600];
    int   k = 0;
//    char    oldpath[MAXPATH];

    posx  = GetTextX();
    posy  = GetTextY();
    index = posy + c_id - 1;

    ImageGet( 14, 3, 55, 15 );
//    getcwd(oldpath, MAXPATH);
    ShowFile( STARTCOL+13, STARTROW+2, STARTCOL+54, STARTROW+14, path, dir );
//    chdir( oldpath );
    ImagePut( 14, 3 );
    if( strlen( dir ) != 3  )
     strcat( dir,"\\");
    strcat( dir, path );
    strcpy( path, dir );
    if( path[0] == '\0' ){
                GotoXY(posx,posy);
                return;
    }
    if( ( fp=fopen( path, "rb" ) ) == NULL ){
                ImageGet(5,6,65,8);
                ClearScreen(5,6,65,8,BACKGROUNDR);
                GotoXY(6,7);
                WriteShiftRvs(" ไม่สามารถเปิดไฟล์ที่ระบุได้ กดคีย์ใดๆเพื่อทำงานต่อ",0,1);
                getch();
                ImagePut(5,6);
                GotoXY(posx,posy);
                return;
     }
    olde_id = e_id;
    j    = 0;
    if ( (eid[++j] = (char far * )farmalloc(STRLEN)) == NULL ){
                terminate("\nMemory Allocation Error in AddString");
    }
```

```
    eid[j][0] = '\0';
    i    = 0;
    while( !feof(fp)&&( k < 570 ) ){
            eid[j][i] = fgetc(fp);

            if( ( eid[j][i] == '\n' )||( eid[j][i] == 0x1A ) ){
                    eid[j][i] = '\0';
                    if ( (eid[++j] = (char far * )farmalloc(STRLEN)) == NULL ){

                        break;
                    }
                    eid[j][0] = '\0';
                    i = 0;
                    k++;
            }
            else
                    i++;
    }
    farfree(eid[j--]);
    fclose(fp);
    for( i = 0; i < j; i++){
            if ( (sid[++e_id] = (char far * )farmalloc(STRLEN)) == NULL ){
                    for(  i = j; i > 0; i-- )
                       farfree(eid[j--]);
                    terminate("\nMemory Allocation Error in AddString");
            }
            sid[e_id][0] = '\0';        /* initialize new string      */


    }
    for(  i = olde_id; i >= index; i--){

             strcpy(sid[i+j],sid[i]);
    }
            for( i = j-1; i >= 0; i--){
             strcpy(sid[index+i],eid[i+1]);
             farfree(eid[i+1]);
             j--;

            }
            ShowScreen(posy,ROWWIDTH);
            GotoXY(1,posy);
}

void GetInfo(int startx, int stopx, int y, char *Ans, char *msg ,char fjmp ,char rvs)
{
```

```
struct viewporttype vp;
char   temp[50]="";
int    posx, posxx, level;
char   key;
int    start;
int    i,j;
char   delch;
int    posinstr, len1;
int    rxno;

vp = GetTextViewPort();
strcpy(Ans,msg);
SetTextViewPort(startx-1,y,stopx-1,y);
ClearScreen( 1, 1, stopx-startx-1, 1 ,(rvs) ? BACKGROUNDR : BACKGROUND );
GotoXY(1,1);
WriteShiftRvs(Ans,0,rvs);
if( fjmp ){
    SetTextViewPort(vp.left,vp.top,vp.right,vp.bottom);
    return;
}
while(1){
    if ((key = GetKey()) == 0) {
            switch (key = getch())  {
                case KF2  :
                        if( ( rxno=SelectRecv() ) != 500 ){
                          WriteShiftRvs(AccountData[rxno].ObjectName,0,rvs);
                          strcpy( Ans,AccountData[rxno].ObjectName );
                        }
                        return;

                case KF10 :
                        posx = GetTextX();
                        ToggleTypeMode();
                        GotoXY(posx,1);
                        break;
              case KHome :
                        GotoXY(1, 1);
                        break;
              case KEnd :
                        GotoXY(CountChar(Ans) + 1, 1);
                        break;
              case KDel :
                        if (GetTextX() <= CountChar(Ans)) {
                                posx = GetTextX();
                                ImageGet(posx + 1, 1, CountChar(Ans), 1);
                                ImagePut(posx, 1);
```

```
                        ClearChar(CountChar(Ans), 1,rvs);
                        strcpy(temp, Ans);
                        start = PosInString(temp, posx);
                        for (i = PosInString(temp, posx + 1); i <= strlen(temp); i++)
                                Ans[start++] = temp[i];
                }
                break;
        case KArrowLeft :  /* left arrow key  */
                if (GetTextX() != 1)
                        GotoXY(GetTextX() - 1, 1);
                else
                        BeepAlarm();
                break;
        case KArrowRight :  /* right arrow key -> */
                if (GetTextX() < strlen(Ans))
                        GotoXY(GetTextX() + 1, 1);
                else
                        BeepAlarm();
                break;
        }
}
else {
        if ( IsThaiMode() )
                key = EngToThai(key);
        switch (key) {
                case KBACKSPACE:
                        posx = GetTextX();
                        posinstr = PosInString(Ans,posx);

                        if( posx != 1 ){
                          delch = Ans[posinstr-1];
                          for( i = posinstr; Ans[i] != '\0'; i++ )
                            Ans[i-1] = temp[i-posinstr] = Ans[i];
                          Ans[i-1] = temp[i-posinstr] = '\0';
                          if( FindLevel(delch) == 1 ){
                            ClearScreen(posx-1,1,stopx-startx+1,1,( rvs ) ? BACKGROUNDR : BACKGROUND);
                            GotoXY(posx-1,1);
                            WriteShiftRvs(temp,0,rvs);
                          }
                          else{
                            WriteCharShiftRvs(delch,0,rvs);
                          }

                          if( FindLevel(delch) == 1 )
                            GotoXY(posx-1,1);
                        }
```

```
                else{
                  BeepAlarm();
                }

                break;

        case KENTER:
                len1 = strlen( Ans );

                for( i = 0; i < len1; i++){
                  if( Ans[i] == ' ' ){
                                for( j = i+1; j <= len1; j++ )
                                  Ans[j-1] = Ans[j];
                                len1--;
                                i--;
                  }
                }
                SetTextViewPort(vp.left,vp.top,vp.right,vp.bottom);
                return ;

        case KESC:  /* Key ESC to save file and exit */
                strcpy(Ans,"");
                SetTextViewPort(vp.left,vp.top,vp.right,vp.bottom);
                return ;

        default:
                posx = GetTextX();
                level = FindLevel(key);

                posinstr = PosInString( Ans,posx );
                if( level != 1 ){
                  if( ( posx == 1 )||( !IsThaiLetter(Ans[posinstr-1] )&&( FindLevel(Ans[posinstr-1]) == 1 )
)){
                    BeepAlarm();
                    break;
                  }

                  if( level == FindLevel(Ans[posinstr-1]) ){
                    WriteCharShiftRvs(Ans[posinstr-1],0,rvs);
                    Ans[posinstr-1] = key;
                    WriteCharShiftRvs(key,0,rvs);
                    break;
                  }

                  if( ( level == FindLevel(Ans[posinstr-2]) )&&( FindLevel(Ans[posinstr-1]) != 1 ) ){
                    WriteCharShiftRvs(Ans[posinstr-2],0,rvs);
```

```
                    Ans[posinstr-2] = key;
                    WriteCharShiftRvs(key,0,rvs);
                    break;
                  }

                  for( i = strlen(Ans); i >= posinstr; i-- )
                    Ans[i+1] = Ans[i];
                  Ans[posinstr] = key;
                  WriteCharShiftRvs(key,0,rvs);
                  break;

                }
                else{
                  for( i = strlen(Ans); i >= posinstr; i-- )
                    Ans[i+1] = Ans[i];
                  Ans[posinstr] = key;
                }
                WriteCharShiftRvs(key,0,rvs);

                posxx = GetTextX();
                j = 0;
                strcpy(temp,"");
                for( i = PosInString(Ans,posxx); Ans[i] != '\0'; i++)
                  temp[j++] = Ans[i];
                WriteShiftRvs(temp,0,rvs);
                GotoXY(posxx,1);

                break;
              }
            }
          }
        }
}


void ShowFile(int left, int top, int right, int bottom, char *path ,char *dir)
{
        struct  viewporttype vp;
        struct  ffblk fblk, buf[16];
        int     done, i, j;
        char    buffer[MAXPATH];
        int     begin, end, pos;
        char    *ptr, mwait = 0;
        char    oldpath[MAXPATH];

        vp = GetTextViewPort();
        SetTextViewPort( 1, 1, 80, 24 );
```

```
                ClearScreen( left+2, top+1, right, bottom, LIGHTGRAY );
                ClearScreen( left, top, right-2 ,bottom-1, BACKGROUNDR);
                Block( left*8+2, top*20-10, right*8-28, bottom*20-30, 0 , BLUE );
                Block( left*8-1, top*20-13, right*8-25, bottom*20-27, 1 , BLUE );
                Block( left*8+2, top*20+35, right*8-28, top*20+35, 0 , BLUE );
                Block( left*8+2, top*20+37, right*8-28, top*20+37, 0 , BLUE );
                Block( (left+19)*8-3,top*20+35,(left+19)*8-3,bottom*20-30, 0 , BLUE );
                GotoXY( left+2, top+1 );
                WriteShiftRvs( "ชื่อไฟล์:",0,1);
                getcwd(buffer, MAXPATH);
                strcpy( oldpath,buffer );
ddd:
                getcwd(buffer, MAXPATH);
                if( strlen(buffer)==3 )
                    buffer[2] = '\0';
                strcat( buffer, "\\*.*" );
                pos   = 0;
                begin = 1;
                end   = 0;
                done  = findfirst("*.*",&fblk,FA_DIRECIFA_ARCH);
                if( !done && !strcmp(fblk.ff_name,".") )
                    done = findnext(&fblk);
                for( i = 0; ( i < 16 )&&( !done ); i++ ){
                        buf[i] = fblk;
                        if( buf[i].ff_attrib == 0x10 )
                          strcat(buf[i].ff_name,"\\");
                        end++;
                        done = findnext(&fblk);
                }
                for( j = 0; ( j < 16 )&&( j < end ); j++ ){
                    GotoXY( ( j < 8 )? left+4 : left+22, top+3+((j>=8)?j%8:j));
                    WriteShiftRvs( "              ", 0, 1 );
                    GotoXY( ( j < 8 )? left+4 : left+22, top+3+((j>=8)?j%8:j));
                    WriteShiftRvs( buf[j].ff_name, 0, 1 );
                }
bbb:            GetInfo( left+10, left+38, top+1, path, buffer, nwait,1 );
                nwait = 0 ;
                if( !access(path,0) ){
                    findfirst(path,&fblk,FA_DIRECIFA_ARCH);
                    if( fblk.ff_attrib == 0x10 ){
                            GetInfo( left+10, left+38, top+1, buffer, buffer, 1,1 );
                            strcpy( path, buffer );
                            goto aaa;
                    }
                    strcpy(attachfile[0],path);
                }
```

```
            else{
aaa:            end = 0;
                done = findfirst(path,&fblk,FA_DIRECIFA_ARCH);
                if( done ){
                        GetInfo( left+10, left+38, top+1, buffer, buffer, 1, 1 );
                        done = findfirst(buffer,&fblk,FA_DIRECIFA_ARCH);
                }
                if( !done && !strcmp(fblk.ff_name,".") )
                        done = findnext(&fblk);
                for( i = 0; ( i < 16 )&&( !done ); i++ ){
                        buf[i] = fblk;
                        if( buf[i].ff_attrib == 0x10 )
                           strcat(buf[i].ff_name,"\\");
                        end++;
                        done = findnext(&fblk);
                }
                for( j = 0; ( j < 16 )&&( j < end ); j++ ){
                        GotoXY( ( j < 8 )? left+4 : left+22, top+3+((j>=8)?j%8:j));
                        WriteShiftRvs( "              ", 0, 1 );
                        GotoXY( ( j < 8 )? left+4 : left+22, top+3+((j>=8)?j%8:j));
                        WriteShiftRvs( buf[j].ff_name, 0, 1 );
                }
                for( j = end; j <16; j++ ){
                        GotoXY( ( j < 8 )? left+4 : left+22, top+3+((j>=8)?j%8:j));
                        WriteShiftRvs( "              ", 0, 1 );
                }
                Highlight( buf[0].ff_name, left+3, top+3, left+18, 0 );
                while(1){
                        //if( kbhit() )
                          switch(getch()){
                                case KArrowUp :
                                        if( pos != 0 ){
                                                Highlight( buf[pos].ff_name, ( pos >= 8 ) ? left+21 : left+3, ( pos >= 8 ) ? top+3+
(pos%8) : top+3+pos, ( pos >= 8 ) ? left+36 : left+18, 1);
                                                pos--;
                                                Highlight( buf[pos].ff_name, ( pos >= 8 ) ? left+21 : left+3, ( pos >= 8 ) ? top+3+
(pos%8) : top+3+pos, ( pos >= 8 ) ? left+36 : left+18, 0);
                                        }
                                        else{
                                                if( begin != 1 ){
                                                   Highlight( buf[pos].ff_name, ( pos >= 8 ) ? left+21 : left+3, ( pos >= 8 ) ? top+3+
(pos%8) : top+3+pos, ( pos >= 8 ) ? left+36 : left+18, 1);
                                                   end  = 0;
                                                   done = findfirst(path,&fblk,FA_DIRECIFA_ARCH);
                                                   if( !done && !strcmp(fblk.ff_name,".") )
                                                           done = findnext(&fblk);
```

```
                    for( i = 0; i < begin-9; i++ )
                        done = findnext(&fblk);
                    for( i = 0; ( i < 16 )&&( !done ); i++ ){
                        buf[i] = fblk;
                        if( buf[i].ff_attrib == 0x10 )
                            strcat(buf[i].ff_name,"\\");
                        end++;
                        done = findnext(&fblk);
                    }
                    begin -= 8;
                    for( j = 0; ( j < 16 )&&( j <= end ); j++ ){
                        GotoXY( ( j < 8 ) ? left+4 : left+22, top+3+((j>=8)?j%8:j));
                        WriteShiftRvs( "              ", 0, 1 );
                        GotoXY( ( j < 8 ) ? left+4 : left+22, top+3+((j>=8)?j%8:j));
                        WriteShiftRvs( buf[j].ff_name, 0, 1 );
                    }
                    Highlight( buf[7].ff_name, left+3, top+10, left+18, 0 );
                    pos = 7;
                }
                else{
                    Highlight( buf[pos].ff_name, ( pos >= 8 ) ? left+21 : left+3, ( pos >= 8 ) ? top+3+
(pos%8) : top+3+pos, ( pos >= 8 ) ? left+36 : left+18, 1);
                    goto  bbb;
                    }
                }
                break;
            case KArrowDown :
            if( pos < end-1 ){
                Highlight( buf[pos].ff_name, ( pos >= 8 ) ? left+21 : left+3, ( pos >= 8 ) ? top+3+
(pos%8) : top+3+pos, ( pos >= 8 ) ? left+36 : left+18, 1);
                pos++;
                Highlight( buf[pos].ff_name, ( pos >= 8 ) ? left+21 : left+3, ( pos >= 8 ) ? top+3+
(pos%8) : top+3+pos, ( pos >= 8 ) ? left+36 : left+18, 0);
            }
            else{
                if( ( pos == end-1 )&&( pos == 15 ) ){
                Highlight( buf[pos].ff_name, ( pos >= 8 ) ? left+21 : left+3, ( pos >= 8 ) ? top+3+
(pos%8) : top+3+pos, ( pos >= 8 ) ? left+36 : left+18, 1);
                end  = 0;
                done = findfirst(path,&fblk,FA_DIRECIFA_ARCH);
                if( !done && !strcmp(fblk.ff_name,".") )
                    done = findnext(&fblk);
                for( i = 0; i < begin+7; i++ )
                    done = findnext(&fblk);
                for( i = 0; ( i < 16 )&&( !done ); i++ ){
                    buf[i] = fblk;
```

```
                    if( buf[i].ff_attrib == 0x10 )
                        strcat(buf[i].ff_name,"\\");
                    end++;
                    done = findnext(&fblk);
                }
                begin += 8;
                for( j = 0; ( j < 16 )&&( j <= end ); j++ ){
                    GotoXY( ( j < 8 ) ? left+4 : left+22, top+3+((j>=8)?j%8:j));
                    WriteShiftRvs( "              ", 0, 1 );
                    GotoXY( ( j < 8 ) ? left+4 : left+22, top+3+((j>=8)?j%8:j));
                    WriteShiftRvs( buf[j].ff_name, 0, 1 );
                }
                for( j = end; j < 16; j++ ){
                    GotoXY( ( j < 8 ) ? left+4 : left+22, top+3+((j>=8)?j%8:j));
                    WriteShiftRvs( "              ", 0, 1 );
                }
                Highlight( buf[8].ff_name, left+21, top+3, left+36, 0 );
                pos = 8;
              }
              else
                BeepAlarm();
            }
            break;
        case KArrowLeft :
            if( pos > 7 ){
                Highlight( buf[pos].ff_name, ( pos >= 8 ) ? left+21 : left+3, ( pos >= 8 ) ? top+3+
(pos%8) : top+3+pos, ( pos >= 8 ) ? left+36 : left+18, 1);
                pos -= 8;
                Highlight( buf[pos].ff_name, ( pos >= 8 ) ? left+21 : left+3, ( pos >= 8 ) ? top+3+
(pos%8) : top+3+pos, ( pos >= 8 ) ? left+36 : left+18, 0);
            }
            else{
              if( begin != 1 ){
                Highlight( buf[pos].ff_name, ( pos >= 8 ) ? left+21 : left+3, ( pos >= 8 ) ? top+3+
(pos%8) : top+3+pos, ( pos >= 8 ) ? left+36 : left+18, 1);
                end  = 0;
                done = findfirst(path,&fblk,FA_DIRECIFA_ARCH);
                if( !done && !strcmp(fblk.ff_name,".") )
                    done = findnext(&fblk);
                for( i = 0; i < begin-9; i++ )
                    done = findnext(&fblk);
                for( i = 0; ( i < 16 )&&( !done ); i++ ){
                    buf[i] = fblk;
                    if( buf[i].ff_attrib == 0x10 )
                      strcat(buf[i].ff_name,"\\");
                    end++;
```

```
                    done = findnext(&fblk);
                }
                begin -= 8;
                for( j = 0; ( j < 16 )&&( j <= end ); j++ ){
                    GotoXY( ( j < 8 ) ? left+4 : left+22, top+3+((j>=8)?j%8:j));
                    WriteShiftRvs( "              ", 0, 1 );
                    GotoXY( ( j < 8 ) ? left+4 : left+22, top+3+((j>=8)?j%8:j));
                    WriteShiftRvs( buf[j].ff_name, 0, 1 );
                }
                Highlight( buf[pos].ff_name, left+3, top+3+pos, left+18, 0 );
            }
            else
                BeepAlarm();
            }
            break;
        case KArrowRight :
            if( pos < 8 ){
                if( end < 8 ){
                BeepAlarm();
                break;
                }
                Highlight( buf[pos].ff_name, ( pos >= 8 ) ? left+21 : left+3, ( pos >= 8 ) ? top+3+
(pos%8) : top+3+pos, ( pos >= 8 ) ? left+36 : left+18, 1);
                pos = ( pos > (end-1)%8 ) ? end -1 : pos+8;
                Highlight( buf[pos].ff_name, ( pos >= 8 ) ? left+21 : left+3, ( pos >= 8 ) ? top+3+
(pos%8) : top+3+pos, ( pos >= 8 ) ? left+36 : left+18, 0);
            }
            else{
                if( !findnext(&fblk) ){
                Highlight( buf[pos].ff_name, ( pos >= 8 ) ? left+21 : left+3, ( pos >= 8 ) ? top+3+
(pos%8) : top+3+pos, ( pos >= 8 ) ? left+36 : left+18, 1);
                end  = 0;
                done = findfirst(path,&fblk,FA_DIRECIFA_ARCHI);
                if( !done && !strcmp(fblk.ff_name,".") )
                    done = findnext(&fblk);
                for( i = 0; i < begin+7; i++ )
                    done = findnext(&fblk);
                for( i = 0; ( i < 16 )&&( !done ); i++ ){
                    buf[i] = fblk;
                    if( buf[i].ff_attrib == 0x10 )
                        strcat(buf[i].ff_name,"\\");
                    end++;
                    done = findnext(&fblk);
                }
                begin += 8;
                for( j = 0; ( j < 16 )&&( j <= end ); j++ ){
```

```
                                        GotoXY( ( j < 8 ) ? left+4 : left+22, top+3+((j>=8)?j%8:j));
                                        WriteShiftRvs( "          ", 0, 1 );
                                        GotoXY( ( j < 8 ) ? left+4 : left+22, top+3+((j>=8)?j%8:j));
                                        WriteShiftRvs( buf[j].ff_name, 0, 1 );
                                    }
                                    for( j = end; j < 16; j++ ){
                                        GotoXY( ( j < 8 ) ? left+4 : left+22, top+3+((j>=8)?j%8:j));
                                        WriteShiftRvs( "          ", 0, 1 );
                                    }
                                    pos = ( pos > end-1 ) ? end-1 : pos;
                                    Highlight( buf[pos].ff_name, left+21, top+3+pos%8, left+36, 0 );
                                }
                                else
                                    BeepAlarm();
                            }
                            break;
                        case KESC :
                            strcpy( path, "" );
                            goto ccc ;
                        case KENTER :
                            if( buf[pos].ff_attrib != 0x10 ){
                                strcpy( path, buf[pos].ff_name );
                                getcwd(dir, MAXPATH);
                                goto ccc;
                            }
                            else{
                                Highlight( buf[pos].ff_name, ( pos >= 8 ) ? left+21 : left+3, ( pos >= 8 ) ? top+3+
(pos%8) : top+3+pos, ( pos >= 8 ) ? left+36 : left+18, 1);
                                ptr = strrchr( buf[pos].ff_name, 0x5C);;
                                *ptr = '\0';
                                chdir( buf[pos].ff_name );
                                nwait++;
                                goto ddd;
                            }
                        default :
                            break;
                    }

                }
            }
ccc:
            chdir( oldpath );
            SetTextViewPort( vp.left, vp.top, vp.right, vp.bottom );

}
```

```
void SelectAttach(  int left, int top, int right, int bottom )
{
        struct viewporttype vp;
        int             i, pos = 0;
        char            dir[MAXATTACH][MAXPATH];
//      char            oldpath[MAXPATH];
        char            *ptr, temp[15];
        for( i = 0; i < attcount; i++ ){
            ptr = strrchr( attachfile[i], '\\');
            strcpy( temp, ptr+1 );
            *ptr = '\0';
            strcpy( dir[i], attachfile[i] );
            strcpy( attachfile[i], temp );
        }
Startatt:
        vp = GetTextViewPort();
        SetTextViewPort( 1, 1, 80, 24 );
        ClearScreen( left+2, top+1, right, bottom, LIGHTGRAY );
        ClearScreen( left, top, right-2 , bottom-1, BACKGROUNDR);
        Block( left*8-3, top*20-13, right*8-23, bottom*20-27, 0 , BLUE );
        Block( left*8-6, top*20-16, right*8-20, bottom*20-24, 1 , BLUE );
        GotoXYShift( left+5, top, 13 );
        WriteShiftRvs( "รายชื่อไฟล์ที่ต้องการส่งไปพร้อมจดหมาย", 13, 1 );
        GotoXYShift( left+8, top+1, 9 );
        WriteShiftRvs( "[  Attach File List  ]", 9, 1 );
        Block( left*8-3, top*20+30, right*8-23, top*20+30, 0, BLUE );
        GotoXYShift( left+1, bottom-2, 13 );
        WriteShiftRvs( "<Ins-เพิ่มไฟล์> <Del-ลบไฟล์> <^ENTER-ตกลง>", 13, 1 );
        Block( left*8-3, bottom*20-50, right*8-23, bottom*20-50, 0, BLUE );
        Block( (left+19)*8, top*20+30, (left+19)*8, bottom*20-50, 0, BLUE );
        GotoXYShift( right-17, bottom-3, 18);
        WriteShiftRvs( "< F1-ช่วยเหลือ >", 18, 1 );

        for( i = 0; (i < attcount); i ++ ){
            GotoXY( ( i >= 7 )? left+23 : left+3, top+3+(i%7) );
            WriteShiftRvs( attachfile[i], 0, 1 );
        }
        if( attcount > 0 )
            Highlight( attachfile[pos], left+2, top+3+pos, left+18, 0 );
        while( 1 ){
          switch( getch() ){
            case KIns :    if( attcount < MAXATTACH ){
                                    SetTextViewPort( vp.left, vp.top, vp.right, vp.bottom );
                                    ShowFile( STARTCOL+13, STARTROW+2, STARTCOL+54, STARTROW+14,
attachfile[attcount], dir[attcount] );
                                    if( attachfile[attcount][0] != 0 )
```

```
                                        attcount++;
                                    for( i = 0; i < attcount-1; i++ )
                                        if( !strcmp(attachfile[i],attachfile[attcount-1]) ){
                                            attcount--;
                                            break;
                                    }
                                    goto Startatt;
                                }
                                break;
                case KArrowUp : if( attcount > 0 ){
                                    Highlight( attachfile[pos], ( pos >= 7 ) ? left+22 : left+2, top+3+(pos%7), ( pos >= 7
) ? left+38 : left+18, 1 );
                                    pos = ( pos == 0 ) ? attcount-1 : pos -1 ;
                                    Highlight( attachfile[pos], ( pos >= 7 ) ? left+22 : left+2, top+3+(pos%7), ( pos >= 7
) ? left+38 : left+18, 0 );
                                }
                                break;
                case KArrowDown :
                                if( attcount > 0 ){
                                    Highlight( attachfile[pos], ( pos >= 7 ) ? left+22 : left+2, top+3+(pos%7), ( pos >= 7
) ? left+38 : left+18, 1 );
                                    pos = ( pos == attcount-1 ) ? 0 : pos+1 ;
                                    Highlight( attachfile[pos], ( pos >= 7 ) ? left+22 : left+2, top+3+(pos%7), ( pos >= 7
) ? left+38 : left+18, 0 );
                                }
                                break;
                case KArrowRight :
                case KArrowLeft  :
                                if( attcount > 0 ){
                                    Highlight( attachfile[pos], ( pos >= 7 ) ? left+22 : left+2, top+3+(pos%7), ( pos >= 7
) ? left+38 : left+18, 1 );
                                    pos = ( pos < 7 ) ? ( ( pos+7 >= attcount )?( (attcount<7) ? pos : attcount-1 ) :
pos+7 ) : pos - 7 ;
                                    Highlight( attachfile[pos], ( pos >= 7 ) ? left+22 : left+2, top+3+(pos%7), ( pos >= 7
) ? left+38 : left+18, 0 );
                                }
                                break;
                case KDel :     if( attcount > 0 ){
                                    Highlight( attachfile[pos], ( pos >= 7 ) ? left+22 : left+2, top+3+(pos%7), ( pos >= 7
) ? left+38 : left+18, 1 );
                                    for( i = pos; i < attcount; i++ )
                                        strcpy( attachfile[i], attachfile[i+1] );
                                    attcount--;
                                    for( i = 0; (i < attcount); i ++ ){
                                        GotoXY( ( i >= 7 )? left+23 : left+3, top+3+(i%7) );
                                        WriteShiftRvs( "                    ", 0, 1 );
```

```
                                        GotoXY( ( i >= 7 )? left+23 : left+3, top+3+(i%7) );
                                        WriteShiftRvs( attachfile[i], 0, 1 );
                                    }
                                    GotoXY( ( attcount >= 7 )? left+23 : left+3, top+3+(attcount%7) );
                                    WriteShiftRvs( "                    ", 0, 1 );
                                    if( ( pos == attcount )&&( pos ) )
                                        pos--;
                                    Highlight( attachfile[pos], ( pos >= 7 ) ? left+22 : left+2, top+3+(pos%7), ( pos >= 7
) ? left+38 : left+18, 0 );
                                }
                                break;

                case 10  :
                case KESC :     for( i = 0; i < attcount; i++ ){
                                    if( strlen(dir[i]) != 3 )
                                        strcat( dir[i], "\\" );
                                    strcat( dir[i], attachfile[i] );
                                                                        strcpy( attachfile[i], dir[i] );
                                }
                                SetTextViewPort( vp.left, vp.top, vp.right, vp.bottom );
                                return;
                    default   :  break;
                }
            }
}

void SendMail( LONG RxID, char * Subject, char * MailFileName, char AttachFileName[14][40], int AttachFileCount, BYTE maillength, char
flagdif, char *server, char voc )
{
            FILE                * fp;
            MailFrameType mailframe;
            volatile register int  i;
            char          rxbox[100], rxmail[100], rxatt[100];
            char                  drive[3], dir[66], name[9], ext[5];
            BYTE          LowID;
            int           randnum;

            memset(&mailframe, 0, sizeof(MailFrameType));
            mailframe.byMailLength = maillength;
            sprintf(mailframe.Sendname,"%s\\%s",ServerName[0],LocalName);
            strcpy(mailframe.SendfullName,LocalFullName);
//          MailFrame.lTrID = LocalID;
            mailframe.fNewMail = mailframe.fNotRead = TRUE;
            mailframe.fDelMail = FALSE;
            strcpy(mailframe.szMailSubject, Subject);
```

```
        GetSysTime();                           // Get current NetWare server 's time
        time(&mailframe.lTime);    // and Set to MailHeader data
        LowID = (BYTE)( LongSwitch( &LocalID ) & 0x00FFUL );
        randomize();
        randnum = random(0xFFFE);
        if( flagdif ){
          if( voc )
                sprintf(nxbox, "%s\\SYS:MAIL\\%lX\\%04lX%04X.%02X", server, RxID, mailframe.lTime&0x0000
FFFF,randnum,LowID );
            else
                sprintf(nxbox, "%s\\SYS:MAIL\\%lX\\%04lX%04X.%02XH", server, RxID, mailframe.lTime&0x0000
FFFF,randnum,LowID );
        }
        else{
          if( voc )
                sprintf(nxbox, "SYS:MAIL\\%lX\\%04lX%04X.%02X", RxID, mailframe.lTime&0x0000FFFF,randnum,LowID );
          else
                sprintf(nxbox, "SYS:MAIL\\%lX\\%04lX%04X.%02XH", RxID, mailframe.lTime&0x0000FFFF,randnum,LowID );
        }
        if( (fp=fopen(nxbox, access(nxbox, 0) ? "wb":"a+b")) == NULL )
                terminate("Cannot access mailbox");
        if( flagdif )
          sprintf(nxmai, "%s\\SYS:MAIL\\%lX\\%04lX%04X.%02XM", server, RxID, mailframe.lTime&0x0000FFFF,randnum,LowID );
        else
          sprintf(nxmai, "SYS:MAIL\\%lX\\%04lX%04X.%02XM", RxID, mailframe.lTime&0x0000FFFF,randnum,LowID );

        sprintf(mailframe.MailContentID, "%04lX%04X.%02XM", mailframe.lTime&0x0000FFFF,randnum,LowID );

        mailframe.byAttach = AttachFileCount;
        for(i = 0; i < mailframe.byAttach; i++) {
                if( flagdif )
                  sprintf(nxatt, "%s\\SYS:MAIL\\%lX\\%04lX%04X.%02X%1X",  server, RxID,
mailframe.lTime&0x0000FFFF,randnum, LowID, i );
                else
                  sprintf(nxatt, "SYS:MAIL\\%lX\\%04lX%04X.%02X%1X",  RxID, mailframe.lTime&0x0000FFFF,randnum, LowID, i
);
                sprintf(mailframe.szAttachID[i], "%04lX%04X.%02X%1X", mailframe.lTime&0x0000FFFF,randnum, LowID, i );
                fnsplit( AttachFileName[i], drive, dir, name, ext);
                sprintf( mailframe.szAttachName[i], "%s%s", name,ext );
                //strcpy(mailframe.szAttachName[i], AttachFileName[i]);
                CopyFile( AttachFileName[i], nxatt);
        }

        fwrite(&mailframe, sizeof(MailFrameType), 1, fp);
        fclose(fp);
```

```
        CopyFile(MailFileName, rxmail);          // Copy MailFileName to RxMailBox
}


int CopyFile( const char *InputFile, const char *OutputFile )
{
        FILE        *fpi, *fpo;
        char        *buffer;
        long        fl;

        if((fpi=fopen(InputFile, "rb")) == NULL)
                return 0;
        if((fpo=fopen(OutputFile, "wb"))== NULL)
                return 0;

        fl= filelength(fileno(fpi));
        while( fl/3000 >= 1 ){
          if((buffer=(char *) farmalloc((unsigned long)3000)) == NULL)
                return 0;
          fread (buffer, sizeof(char), 3000U, fpi);
          fwrite(buffer, sizeof(char), 3000U, fpo);
          farfree(buffer);
          fl -= 3000;
          }

        if((buffer=(char *) farmalloc((unsigned long)fl)) == NULL)
                return 0;
        fread (buffer, sizeof(char),(size_t) fl, fpi);
        fwrite(buffer, sizeof(char),(size_t) fl, fpo);
        farfree(buffer);

        fclose(fpi);
        fclose(fpo);

        return 1;
}

unsigned ShiftLeftAround( unsigned int input, int key )
{
        BITFORM data;
        register int highbit,i;

        data.word_field = input;

        for (i=0; i<key; i++) {
```

```
                highbit = data.bit_field.hibit;
                data.word_field <<= 1;
                data.bit_field.lowbit = highbit;
        }
        return data.word_field;
}


unsigned ShiftRightAround(unsigned int input, int key)
{
        return ShiftLeftAround(input, 16-key);
}


void TransposeBlock(BLOCKFORM dcrypte[16])
{
        unsigned int d[16][16];
        register int temp,i,j;

        for (i=0; i<16; i++)        // copy data to buffer d[i][j]
                for (j=0; j<16; j++)
                        d[i][j] = ( dcrypte[i].word_field & 1 << j );

        for (i=0; i<15; i++) // transpose column to row and row to column
                for (j=i+1; j<16; j++) {
                        temp = d[i][j];
                        d[i][j] = d[j][i];
                        d[j][i] = temp;
                }

        for (i=0; i<16; i++) { // copy buffer back to output
                for (temp=j=0; j<16; j++) {
                        if (d[i][j] != 0)
                                temp |=  (1 << j);
                        dcrypte[i].word_field = temp;
                }
        }
}


void EncrypteFile( char * pathfilename )
{
        int             i, j, size;
        FILE * fp1, * fp2;
        char message[32],  outfile[13], *template="TMAILXXXXXX";
        BLOCKFORM               dcrypte[16];
```

```
        strcpy( outfile, template );
        mktemp( outfile );
        if ( (fp1 = fopen(pathfilename, "rb+")) == NULL )
                terminate("\nOpen File Error");
        if ( (fp2 = fopen(outfile, "wb+")) == NULL )
                terminate("\nOpen File Error");

        while ( (size = fread(message, sizeof( char ), 32, fp1) ) > 0 ) {
                if (size < 32)
                        for (i=size;i<32;i++)
                                message[i] = ' ';

                for (i = 0, j = 0; i < 16; i++, j += 2) {
                        dcrypte[i].byte_field.lowbyte = message[j];
                        dcrypte[i].byte_field.hibyte = message[j + 1];
                }

                for (i = 0; i < 16; i++)
                        dcrypte[i].word_field = ShiftLeftAround(dcrypte[i].word_field, key[i]);
                TransposeBlock(dcrypte);

                for (i = 0; i < 16; i++)
                        dcrypte[i].word_field = ShiftLeftAround(dcrypte[i].word_field, key[i+16]);
                TransposeBlock(dcrypte);

                for (i = 0, j = 0; i < 16; i++, j += 2) {
                        message[j] = dcrypte[i].byte_field.lowbyte;
                        message[j+1] = dcrypte[i].byte_field.hibyte;
                }
                fwrite(message, sizeof( char ), 32, fp2);
        }
        fclose(fp1);
        fclose(fp2);

        unlink(pathfilename);       // delete input file
        rename(outfile, pathfilename); // change name temp file to input file
}

void DecrypteFile(char * pathfilename, char * outfile)
{
        int  i, j, size;
        FILE * fp1, * fp2;
        char message[32];
```

```
        BLOCKFORM       dcrypte[16];

        if ( (fp1 = fopen(pathfilename, "rb+")) == NULL )
                terminate("\nOpen File Error");
        if ( (fp2 = fopen(outfile, "wb+")) == NULL )
                terminate("\nOpen File Error");
        while ( (size = fread(message, sizeof( char ), 32, fp1) ) > 0 ) {

                if (size < 32)
                        for (i = size ; i < 32; i++) message[i] = ' ';

                for (i = 0, j = 0; i < 16; i++, j+=2) {
                        dcrypte[i].byte_field.lowbyte = message[j];
                        dcrypte[i].byte_field.hibyte = message[j + 1];
                }
                TransposeBlock(dcrypte);
                for (i = 15; i >= 0; i--)
                        dcrypte[i].word_field = ShiftRightAround(dcrypte[i].word_field, key[i+16]);

                TransposeBlock(dcrypte);
                for (i = 15; i >= 0; i--)
                        dcrypte[i].word_field = ShiftRightAround(dcrypte[i].word_field, key[i]);

                for (i = 0, j = 0; i < 16; i++, j+=2) {
                        message[j] = dcrypte[i].byte_field.lowbyte;
                        message[j + 1] = dcrypte[i].byte_field.hibyte;
                }
                fwrite(message, sizeof( char ), 32, fp2);
        }
        fclose(fp1);
        fclose(fp2);
}


void CheckMail(void)
{
        FILE            *fp;
        register int i=0;
        struct ffblk ffblk;
        int         done;
        char        *fname[13];

        sprintf(LoBox, "SYS:MAIL\\%lX\\*.??H", LocalID);
        done = findfirst(LoBox,&ffblk,0);
        while (!done){
          if( ( fname[i] = (char *)malloc(13) ) == NULL )
```

```
                terminate("Alloc memory in mailbox system error");
            strcpy(fname[i++],ffblk.ff_name);
            done = findnext(&ffblk);
        }
        if( (AllMail = i) == 0 )
            return;
        if( (MailFrame=(MailFrameType *) calloc( AllMail, sizeof(MailFrameType))) == NULL )
            terminate("Alloc memory in mailbox system error");
        for(i = 0; i < AllMail; i++){
            sprintf(LoBox, "SYS:MAIL\\%lX\\%s", LocalID,fname[i]);
            if( (fp=fopen(LoBox,"rb")) == NULL )
                terminate("Error in opening mailbox");
            fread(&MailFrame[i],sizeof( MailFrameType ), 1, fp);
            fclose(fp);
        }
        for(i = 0; i < AllMail; i++){  /* Check all mail          */
            if(MailFrame[i].fNewMail){    /* for New mail          */
//              MailFrame[i].fNewMail = FALSE;
                    NewMail++;
            }
            if(MailFrame[i].fNotRead){          /* end mail that not read yet*/
//              MailFrame[i].fNotRead = FALSE;
                    NotRead++;
            }
        }
        for( i = AllMail-1; i >= 0; i-- )
            free( fname[i] );
        i=0;
}
void CheckVocMail(void)
{
        FILE            *fp;
        register int i=0;
        struct ffblk ffblk;
        int         done;
        char        *fname[13];

        sprintf(LoBox, "SYS:MAIL\\%lX\\*.??", LocalID);
        done = findfirst(LoBox,&ffblk,0);
        while (!done){
            if( ( fname[i] = (char *)malloc(13) ) == NULL )
                    terminate("Alloc memory in mailbox system error");
            strcpy(fname[i++],ffblk.ff_name);
            done = findnext(&ffblk);
        }
        if( (AllVocMail = i) == 0 )
```

```
        return;
    if( (VocMailFrame=(MailFrameType *) calloc( AllVocMail, sizeof(MailFrameType))) == NULL )
        terminate("Alloc memory in mailbox system error");
    for(i = 0; i < AllVocMail; i++){
        sprintf(LoBox, "SYS:MAIL\\%lX\\%s", LocalID,fname[i]);
        if( (fp=fopen(LoBox,"rb")) == NULL )
                terminate("Error in opening voice mailbox");
        fread(&VocMailFrame[i],sizeof( MailFrameType ), 1, fp);
        fclose(fp);
    }
    for(i = 0; i < AllVocMail; i++){            /* Check all mail          */
        if(VocMailFrame[i].fNewMail){   /* for New mail    */
//          MailFrame[i].fNewMail = FALSE;
                NewVocMail++;
        }
        if(VocMailFrame[i].fNotRead){       /* and mail that not read yet*/
//          MailFrame[i].fNotRead = FALSE;
                VocNotRead++;
        }
    }
    for( i = AllVocMail-1; i >= 0; i-- )
        free( fname[i] );
    i=0;
}

void ReCheckMail(void)
{
    FILE            *fp;
    register int i=0;
    struct ffblk ffblk;
    int         done;
    char        *fname[13];
    int         OldMail;

    sprintf(LoBox, "SYS:MAIL\\%lX\\*.??H", LocalID);
    done = findfirst(LoBox,&ffblk,0);
    while (!done){
        if( ( fname[i] = (char *)malloc(13) ) == NULL )
                terminate("Alloc memory in mailbox system error");
        strcpy(fname[i++],ffblk.ff_name);
        done = findnext(&ffblk);
    }
    if( i == AllMail )
        return;

    OldMail = AllMail;
```

```
        //AllMail = i;
        if( (AllMail = i) == 0 )
          return;

        if( (MailFrame=(MailFrameType *) realloc( MailFrame, AllMail*sizeof(MailFrameType) )) == NULL )
          terminate("Alloc memory in mailbox system error");
        for(i = OldMail; i < AllMail; i++){
          sprintf(LoBox, "SYS:MAIL\\%lX\\%s", LocalID,fname[i]);
          if( (fp=fopen(LoBox,"rb")) == NULL )
                terminate("Error in opening mailbox");
          fread(&MailFrame[i],sizeof( MailFrameType ), 1, fp);
          fclose(fp);
        }
        for(i = OldMail; i < AllMail; i++){        /* Check all mail          */
          if(MailFrame[i].fNewMail){   /* for New mail        */
//          MailFrame[i].fNewMail = FALSE;
                NewMail++;
          }
          if(MailFrame[i].fNotRead){          /* and mail that not read yet*/
//          MailFrame[i].fNotRead = FALSE;
                NotRead++;
          }
        }
        for( i = AllMail-1; i >= 0; i-- )
          free( fname[i] );
        i=0;
}


int  SelectRecv(void)
{
        struct viewporttype vp;
        int             posx, posy;
        char             temp[40], show[10][50];
        int             i, j, k;
        int             start = 0, stop = 0, pos = 0;

        posx = GetTextX();
        posy = GetTextY();
        vp = GetTextViewPort();
        SetTextViewPort( 1, 2, 80, 24 );
        ImageGet( 15, 6, 65, 18 );
        ClearScreen( 17, 7, 65, 18, BLUE );
        ClearScreen( 15, 6, 63, 17, BACKGROUNDR );
        Block( 14*8+6, 5*20+8, 63*8-6, 17*20-7, 0, BLUE );
        Block( 14*8+8, 5*20+10, 63*8-8, 17*20-9, 0, BLUE );
```

```
GotoXYShift( 33, 6, 0 );
WriteShiftRvs( " List of User ", 0, 1 );
GotoXY( 18, 17 );
WriteShiftRvs( "^Enter-เลือก", 0, 1 );
GotoXY( 51, 17 );
WriteShiftRvs( "ESC-ยกเลิก", 0, 1 );
for( i = 0; ( i < 10 )&&( i < AccountNumber ); i++ ){
    GotoXY( 18, 7+i );
    WriteShiftRvs( AccountData[i].ObjectName,  0, 1 );
    strcpy( show[i], AccountData[i].ObjectName );
    j = strlen( AccountData[i].ObjectName );
    for( k = 0; k+j < 14; k++ )
            show[i][j+k] = ' ';
    show[i][j+k] = '\0';
    strcpy( temp, AccountData[i].ObjectFullName );
    if( strlen(temp) > 30 )
            temp[30] = '\0';
    strcat( show[i], temp );
    GotoXY( 32, 7+i );
    WriteShiftRvs( temp, 0, 1 );
    stop++;
  }
  Highlight( show[pos], 17,  7+pos,  62, 0 );
  start++;
  while( 1 ){
                    switch( getch() ){

/*                                  case KPgUp :
                                        if( start != 1 ){
                                            if(*/
                                  case KArrowUp :
                                        if( pos > 0 ){
                                            Highlight( show[pos], 17,  7+pos,  62, 1 );
                                            pos--;
                                            Highlight( show[pos], 17,  7+pos,  62, 0 );
                                        }
                                        else{
                                            if( start != 1 ){
                                              //Highlight( show[pos], 17,  7+pos,  62, 1 );
                                              for( i = stop-start-1; i >= 0; i-- )
                                                      strcpy( show[i+1], show[i] );
                                              strcpy( show[0], AccountData[start-2].ObjectName );
                                              j = strlen( AccountData[start-2].ObjectName );
                                              for( k = 0; k+j < 14; k++ )
                                                      show[0][j+k] = ' ';
                                              show[0][k+j] = '\0';
```

```
                                strcpy( temp, AccountData[start-2].ObjectFullName );
                                if( strlen(temp) > 30 )
                                        temp[30] = '\0';
                                strcat( show[0], temp );

                                for( i = 0; i <= stop-start ; i++ ){
                                        GotoXY( 17, 7+i );
                                        WriteShiftRvs( "
",  0, 1 );

                                        GotoXY( 18, 7+i );
                                        WriteShiftRvs( show[i], 0, 1 );
                                }
                                Highlight( show[pos], 17,  7+pos,  62, 0 );
                                start--;
                                stop--;
                        }
                        else
                                BeepAlarm();
                        }
                        break;

                case KArrowDown :
                                if( pos < stop-start ){
                                Highlight( show[pos], 17,  7+pos,  62, 1 );
                                pos++;
                                Highlight( show[pos], 17,  7+pos,  62, 0 );
                                }
                                else{
                                if( stop != AccountNumber ){
                //Highlight( show[pos], 17,  7+pos,  62, 1 );
                for( i = 1; i <= stop-start; i++ )
                                        strcpy( show[i-1], show[i] );
                strcpy( show[stop-start], AccountData[stop].ObjectName );
                j = strlen( AccountData[stop].ObjectName );
                for( k = 0; k+j < 14; k++ )
                                        show[stop-start][j+k] = ' ';
                show[stop-start][k+j] = '\0';
                strcpy( temp, AccountData[stop].ObjectFullName );
                if( strlen(temp) > 30 )
                                        temp[30] = '\0';
                strcat( show[stop-start], temp );

                for( i = 0; i < stop-start ; i++ ){
                                        GotoXY( 17, 7+i );
                                        WriteShiftRvs( "
",  0, 1 );
```

```
                                        GotoXY( 18, 7+i );
                                        WriteShiftRvs( show[i], 0, 1 );
                                }
                                Highlight( show[stop-start], 17,  7+pos,  62, 0 );
                                start++;
                                stop++;
                                        }
                                        else
                                BeepAlarm();
                                        }
                                        break;

                                case 10  :
                                        ImagePut( 15, 6 );
                                        SetTextViewPort( vp.left, vp.top, vp.right, vp.bottom );
                                        GotoXY( posx, posy );
                                        return start+pos-1;

                                case KESC :
                                        ImagePut( 15, 6 );
                                        SetTextViewPort( vp.left, vp.top, vp.right, vp.bottom );
                                        GotoXY( posx, posy );
                                        return 500;
                                default  :  break;
                        }
                }
}

void   ChNewMail(int flag)
{
        int   i, j = 0;
        int   *new;
        int   pos = 0;
        int   start = 0, stop = 0;
        char  key;


        SetTextViewPort(1,1,80,24);
        if( flag )
        Putbkmenu(9,1,75,3);
        else
        Putbackground();
        setfillstyle(SOLID_FILL,DARKGRAY);
        bar(92,116,548,397);
        SetTextViewPort(11,6,67,19);
```

```
ClearScreen(1,1,57,14,BACKGROUND);
Block(4,9,450,272,1,WHITE);
Block(7,12,447,269,0,WHITE);
GotoXY( 16, 1 );
Write("รายชื่อจดหมายใหม่-New Mail List");
GotoXY( 4, 19 );
Write("^Enter-อ่าน");
GotoXY( 21, 19 );
Write("Del-ลบ");
GotoXY( 34, 19 );
Write("U-ไม่ลบ");
GotoXY( 47, 19 );
Write("Esc-ยกเลิก");
Block( 7, 34, 447, 65, 0, WHITE );
GotoXY( 5, 3 );
Write("ผู้ส่ง-Sender      หัวเรื่อง-Title      วันเวลาส่ง-DATE");

new = ( int * )calloc( NewMail, sizeof(int) );
for( i = 0, j = 0; (i < AllMail)&&(j < NewMail); i++ ){
        if( MailFrame[i].fNewMail ){
        new[j++] = i;

    }
}

for( i = 0; (i < NewMail)&&(i < 9); i++ ){
    GotoXY( 4, 5+i );
    //Write( AccountData[GetMailID('', MailFrame[new[i]].lTrID)].ObjectName );
    Write( MailFrame[new[i]].Sendname );
            GotoXY( 21, 5+i );
            Write( MailFrame[new[i]].szMailSubject );
            GotoXY( 39, 5+i );
            Write( MailTime(MailFrame[new[i]].lTime));
            stop++;
    }

    start++;
    if( NewMail ){
        //Highlight(AccountData[GetMailID('', MailFrame[new[0]].lTrID)].ObjectName,3,5,15,1);
        Highlight(MailFrame[new[0]].Sendname,3,5,20,1);
        Highlight(MailFrame[new[0]].szMailSubject,20,5,38,1);
        Highlight(MailTime(MailFrame[new[0]].lTime),38,5,56,1);

        pos = 1;
        while(1){
```

```
if( (key = getch()) == 0 )
  switch( key = getch() ){

  case KDel :
                    MailFrame[new[pos+start-2]].fDelMail = TRUE;
                    break;

  case KArrowUp :
                    if( pos != 1 ){
                              Highlight(MailFrame[new[pos+start-2]].Sendname,3,4+pos,20,0);
                              Highlight(MailFrame[new[pos+start-2]].szMailSubject,20,4+pos,38,0);
                              Highlight(MailTime(MailFrame[new[pos+start-2]].lTime),38,4+pos,56,0);
                              pos--;
                              Highlight(MailFrame[new[pos+start-2]].Sendname,3,4+pos,20,1);
                              Highlight(MailFrame[new[pos+start-2]].szMailSubject,20,4+pos,38,1);
                              Highlight(MailTime(MailFrame[new[pos+start-2]].lTime),38,4+pos,56,1);
                    }
                    else{
                              if( start != 1 ){
                              start--;
                              Highlight(MailFrame[new[pos+start-2]].Sendname,3,4+pos,20,1);
                              Highlight(MailFrame[new[pos+start-2]].szMailSubject,20,4+pos,38,1);
                              Highlight(MailTime(MailFrame[new[pos+start-2]].lTime),38,4+pos,56,1);
                              for( i = 0; ( i < stop-start )&&( i < 9 ); i++ ){
                                        Highlight(MailFrame[new[start+i]].Sendname,3,6+i,20,0);
                                        Highlight(MailFrame[new[start+i]].szMailSubject,20,6+i,38,0)
;
                                        Highlight(MailTime(MailFrame[new[start+i]].lTime),38,6+i,56,0);
                              }
                              if( stop-start >= 10 )
                                        stop--;
                    }
                    else
                              BeepAlarm();
                    }
                    break;
  case  KArrowDown :
                    if( ( pos != 10 )&&( pos+start-1 < NewMail ) ){
                              Highlight(MailFrame[new[pos+start-2]].Sendname,3,4+pos,20,0);
                              Highlight(MailFrame[new[pos+start-2]].szMailSubject,20,4+pos,38,0);
                    Highlight(MailTime(MailFrame[new[pos+start-2]].lTime),38,4+pos,56,0);
                    pos++;
                              Highlight(MailFrame[new[pos+start-2]].Sendname,3,4+pos,20,1);
                              Highlight(MailFrame[new[pos+start-2]].szMailSubject,20,4+pos,38,1);
                    Highlight(MailTime(MailFrame[new[pos+start-2]].lTime),38,4+pos,56,1);
                    }
```

```
                                        else{
                                            if( ( stop < NewMail )&&( pos+start-1 < NewMail ) ){
                                                    for( i = 0; i < stop-start; i++ ){
                                                            Highlight(MailFrame[new[start+i]].Sendname,3,5+i,20,0);
                                                            Highlight(MailFrame[new[start+i]].szMailSubject,20,5+i,38,0);
                                                      Highlight(MailTime(MailFrame[new[start+i]].lTime),38,5+i,56,0);
                                                    }
                                                    start++;
                                                    Highlight(MailFrame[new[pos+start-2]].Sendname,3,4+pos,20,1);
                                                    Highlight(MailFrame[new[pos+start-2]].szMailSubject,20,4+pos,38,1);
                                                    Highlight(MailTime(MailFrame[new[pos+start-2]].lTime),38,4+pos,56,1);
                                                    stop++;
                                                }
                                                else
                                                    BeepAlarm();
                                            }

                                            break;
                    }
                else
                    switch( key ){
                            case 'U' :
                            case 'u' :
                                            MailFrame[new[pos+start-2]].fDelMail = FALSE;
                                            break;
                            case KESC :

                                            SetTextViewPort(1,1,80,24);
                                            return;
                            case KENTER :
                                            ViewMails( new[start+pos-2] );
                                            MailFrame[ new[start+pos-2] ].fNewMail = FALSE;
                                            NewMail--;
                                            free( new );
                                            ChNewMail( 0 );
                                            return;
                            default :
                                            break;
                      }

                  }
                }
                else
        getch();
```

```
    NewMail = 0;
    if( new != NULL )
        free(new);
}

void CompressMailBox( void )
{
    FILE        * fp;
    char    *fname[13];
    int         i=0,j,real;//, OldMail;
    int     done;
    struct ffblk ffblk;

    /* ReRead mail header file to check New Incomming Mail          */
    sprintf(LoBox, "SYS:MAIL\\%lX\\*.??H", LocalID);
    done = findfirst(LoBox,&ffblk,0);
    while (!done && (i < AllMail) ){
      if( ( fname[i] = (char *)malloc(13) ) == NULL )
            terminate("Alloc memory in mailbox system error");
      strcpy(fname[i++],ffblk.ff_name);
      done = findnext(&ffblk);
    }
    real=i;

    for(i = 0; i < AllMail; i++) {
        if(MailFrame[i].fDelMail) {   /* Deleted Mail          */
            sprintf(LoBox, "SYS:MAIL\\%lX\\%s", LocalID, fname[i]);
            unlink(LoBox);
            sprintf(LoMai, "SYS:MAIL\\%lX\\%s", LocalID,MailFrame[i].MailContentID);
            unlink(LoMai);
            /* Update mail's attach file */
            for(j = 0; j < MailFrame[i].byAttach; j++) {
                sprintf(LoAtt, "SYS:MAIL\\%lX\\%s", LocalID, MailFrame[i].szAttachID[j]);
                unlink(LoAtt);
            }
        } else {

            sprintf(LoBox, "SYS:MAIL\\%lX\\%s", LocalID,fname[i]);
            if( (fp=fopen(LoBox,"wb")) == NULL )
              terminate("Error in opening mailbox");
            fwrite(&MailFrame[i], sizeof(MailFrameType), 1, fp);
            fclose(fp);
        }
    }
    for( i=real-1; i>=0; i-- )
```

```
                free(fname[i]);

}

void CompressVocMailBox( void )
{
        FILE        * fp;
        char    *fname[13];
        int         i=0,j,real;//, OldMail;
        int     done;
        struct ffblk ffblk;

        /* ReRead mail header file to check New Incomming Mail          */
        sprintf(LoBox, "SYS:MAIL\\%D\\*.??", LocalID);
        done = findfirst(LoBox,&ffblk,0);
        while (!done && (i < AllVocMail) ){
          if( ( fname[i] = (char *)malloc(13) ) == NULL )
                terminate("Alloc memory in mailbox system error");
          strcpy(fname[i++],ffblk.ff_name);
          done = findnext(&ffblk);
        }
        real=i;

        for(i = 0; i < AllVocMail; i++) {
                if(VocMailFrame[i].fDelMail) {          /* Deleted Mail         */
                        sprintf(LoBox, "SYS:MAIL\\%D\\%s", LocalID, fname[i]);
                        unlink(LoBox);
                        sprintf(LoMai, "SYS:MAIL\\%D\\%s", LocalID,VocMailFrame[i].MailContentID);
                        unlink(LoMai);
                        /* Update mail's attach file */
                        for(j = 0; j < VocMailFrame[i].byAttach; j++) {
                                sprintf(LoAtt, "SYS:MAIL\\%D\\%s", LocalID, VocMailFrame[i].szAttachID[j]);
                                unlink(LoAtt);
                        }
                } else {

                        sprintf(LoBox, "SYS:MAIL\\%D\\%s", LocalID,fname[i]);
                        if( (fp=fopen(LoBox,"wb")) == NULL )
                          terminate("Error in opening mailbox");
                        fwrite(&VocMailFrame[i], sizeof(MailFrameType), 1, fp);
                        fclose(fp);
                }
        }
        for( i=real-1; i>=0; i-- )
                free(fname[i]);
```

```
}

void   CheckReadMail(int flag)
{
        int   i;
        int   pos = 0;
        int   start = 0, stop = 0;
        char  key;

        SetTextViewPort(1,1,80,24);
        if( flag )
          Putbkmenu(9,1,75,3);
        else{
                Putbackground();
        }
        ClearScreen(5,5,78,22,DARKGRAY);
        SetTextViewPort(3,4,77,21);
        ClearScreen(1,1,74,18,BACKGROUND);
        Block(4,8,588,351,1,WHITE);
        Block(7,11,585,348,0,WHITE);
        GotoXYShift(21,1,0);
        WriteShiftRvs(" แฟ้มเก็บจดหมายทั้งหมด-Mail Folder ",0,0);
        //Block(160,18,392,18,0,WHITE);
        GotoXYShift( 6, 2, 15 );
        WriteShiftRvs("ผู้ส่ง-Sender      หัวเรื่อง-Title          วันเวลาส่ง-DATE   ยังไม่อ่าน", 15, 0 );
        GotoXY( 4, 18 );
        Write("^Enter-อ่าน");
        GotoXY( 25, 18 );
        Write("Del-ลบ");
        GotoXY( 42, 18 );
        Write("U-ไม่ลบ");
        GotoXY( 56, 18 );
        Write("Esc-ยกเลิก");
        for( i = 0; ( i < 10 )&&( i < AllMail ); i++ ){
           GotoXY( 4, 5+i );
           Write( MailFrame[i].Sendname );
           GotoXY( 23, 5+i );
           Write( MailFrame[i].szMailSubject );
           GotoXY( 46, 5+i );
           Write( MailTime(MailFrame[i].lTime) );
           GotoXY( 67, 5+i );
                   WriteChar( MailFrame[i].fNotRead ? 'Y' : 'N' );
                   GotoXY( 71, 5+i );
                   WriteChar( MailFrame[i].fDelMail ? MARKSELECT : ' ' );
                   stop++;
        }
```

```
start++;
if( AllMail ){
        Highlight(MailFrame[0].Sendname,3,5,22,1);
        Highlight(MailFrame[0].szMailSubject,22,5,45,1);
        Highlight(MailTime(MailFrame[0].lTime),45,5,66,1);
        Highlight(MailFrame[0].fNotRead ? "Y" : "N",66,5,69,1);
        GotoXY( 70, 5 );
        Write( MailFrame[0].fDelMail ? "#" : " ");

        pos = 1;
        while(1){
         if( (key = getch()) == 0 )
           switch( key = getch() ){

         case KDel :
                        MailFrame[pos+start-2].fDelMail = TRUE;
                        GotoXY( 70,4+pos );
                        WriteChar(' ');
                        WriteChar(MARKSELECT);
                        break;

         case KArrowUp :
                        if( pos != 1 ){

                                Highlight(MailFrame[pos+start-2].Sendname,3,4+pos,22,0);
                                Highlight(MailFrame[pos+start-2].szMailSubject,22,4+pos,45,0);
                                Highlight(MailTime(MailFrame[pos+start-2].lTime),45,4+pos,66,0);
                                Highlight(MailFrame[pos+start-2].fNotRead ? "Y" : "N",66,4+pos,69,0);
                                GotoXY( 70, 4+pos );
                                //WriteChar( MailFrame[pos+start-2].fDelMail ? MARKSELECT : ' ' );
                                Write( MailFrame[pos+start-2].fDelMail ? "#" : " ");
                                pos--;
                                Highlight(MailFrame[pos+start-2].Sendname,3,4+pos,22,1);
                                Highlight(MailFrame[pos+start-2].szMailSubject,22,4+pos,45,1);
                                Highlight(MailTime(MailFrame[pos+start-2].lTime),45,4+pos,66,1);
                                Highlight(MailFrame[pos+start-2].fNotRead ? "Y" : "N",66,4+pos,69,1);
                                GotoXY( 70, 4+pos );
                                //WriteChar( MailFrame[pos+start-2].fDelMail ? MARKSELECT : ' ' );
                                Write( MailFrame[pos+start-2].fDelMail ? "#" : " ");

                        }
                        else{
                                if( start != 1 ){
                                start--;
                                Highlight(MailFrame[pos+start-2].Sendname,3,4+pos,22,1);
                                Highlight(MailFrame[pos+start-2].szMailSubject,22,4+pos,45,1);
                                Highlight(MailTime(MailFrame[pos+start-2].lTime),45,4+pos,66,1);
```

```
                                        Highlight(MailFrame[pos+start-2].fNotRead ? 'Y' : 'N',66,4+pos,69,1);
                                        GotoXY( 70, 4+pos );
                                        //WriteChar( MailFrame[pos+start-2].fDelMail ? MARKSELECT : ' ' );
                                        Write( MailFrame[pos+start-2].fDelMail ? '#' : ' ');
                                        for( i = 0; ( i < stop-start )&&( i < 9 ); i++ ){
                                                Highlight(MailFrame[start+i].Sendname,3,6+i,22,0);
                                                Highlight(MailFrame[start+i].szMailSubject,22,6+i,45,0);
                                                Highlight(MailTime(MailFrame[start+i].lTime),45,6+i,66,0);
                                                Highlight(MailFrame[start+i].fNotRead ? 'Y' : '
N',66,6+i,69,0);
                                                GotoXY( 70, 6+i );
                                                //WriteChar( MailFrame[start+i].fDelMail ? MARKSELECT : '
' );
                                                Write( MailFrame[start+i].fDelMail ? '#' : ' ');
                                        }
                                        if( stop-start >= 10 )
                                                stop--;
                                        }
                                        else
                                        BeepAlarm();
                                }
                                break;
                        case KArrowDown :
                                if( ( pos != 10 )&&( pos+start-1 < AllMail ) ){
                                        Highlight(MailFrame[pos+start-2].Sendname,3,4+pos,22,0);
                                        Highlight(MailFrame[pos+start-2].szMailSubject,22,4+pos,45,0);
                                        Highlight(MailTime(MailFrame[pos+start-2].lTime),45,4+pos,66,0);
                                        Highlight(MailFrame[pos+start-2].fNotRead ? 'Y' : 'N',66,4+pos,69,0);
                                        GotoXY( 70, 4+pos );
                                        //WriteChar( MailFrame[pos+start-2].fDelMail ? MARKSELECT : ' ' );
                                        Write( MailFrame[pos+start-2].fDelMail ? '#' : ' ');
                                        pos++;
                                        Highlight(MailFrame[pos+start-2].Sendname,3,4+pos,22,1);
                                        Highlight(MailFrame[pos+start-2].szMailSubject,22,4+pos,45,1);
                                        Highlight(MailTime(MailFrame[pos+start-2].lTime),45,4+pos,66,1);
                                        Highlight(MailFrame[pos+start-2].fNotRead ? 'Y' : 'N',66,4+pos,69,1);
                                        GotoXY( 70, 4+pos );
                                        //WriteChar( MailFrame[pos+start-2].fDelMail ? MARKSELECT : ' ' );
                                        Write( MailFrame[pos+start-2].fDelMail ? '#' : ' ');
                                }
                                else{
                                        if( ( stop < AllMail )&&( pos+start-1 < AllMail ) ){
                                        for( i = 0; i < stop-start; i++ ){
                                                Highlight(MailFrame[start+i].Sendname,3,5+i,22,0);
                                                Highlight(MailFrame[start+i].szMailSubject,22,5+i,45,0);
                                                Highlight(MailTime(MailFrame[start+i].lTime),45,5+i,66,0);
```

```
                                        Highlight(MailFrame[start+i].fNotRead ? "Y" : "N",66,5+i,69,0)
;
                                        GotoXY( 70, 5+i );
                                        //WriteChar( MailFrame[start+i].fDelMail ? MARKSELECT : ' '
);
                                        Write( MailFrame[start+i].fDelMail ? "#" : " ");
                                    }
                                    start++;
                                    Highlight(MailFrame[pos+start-2].Sendname,3,4+pos,22,1);
                                    Highlight(MailFrame[pos+start-2].szMailSubject,22,4+pos,45,1);
                                    Highlight(MailTime(MailFrame[pos+start-2].lTime),45,4+pos,66,1);
                                    Highlight(MailFrame[pos+start-2].fNotRead ? "Y" : "N",66,4+pos,69,1);
                                    GotoXY( 70, 4+pos );
                                    //WriteChar( MailFrame[pos+start-2].fDelMail ? MARKSELECT : ' ' );
                                    Write( MailFrame[pos+start-2].fDelMail ? "#" : " ");
                                    stop++;
                                    }
                                    else
                                    BeepAlarm();
                                }
                                break;
                            }
                        else
                         switch( key ){
                        case 'U' :
                        case 'u' :
                                MailFrame[pos+start-2].fDelMail = FALSE;
                                GotoXY( 70,4+pos );
                                WriteChar(' ');
                                break;
                        case KESC :
                                SetTextViewPort(1,1,80,24);
                                return;
                        case KENTER :
                                ViewMails( start+pos-2 );
                                CheckReadMail( 0 );
                                return;
                        default :
                                break;
                         }
                    }
                }
```

```
        getch();
}
void DialogBox(char *st)
{
        int left=20,right=60,top=10,bottom=11;
        char title[] = " ข้อผิดพลาด ",
                recommend[] = "กด Enter เพื่อทำงานต่อ";
        struct viewporttype vp;

        vp = GetTextViewPort();
        SetTextViewPort(1,1,80,24);
        ImageGet(left-1,top-1,right+1,bottom+1);
        ClearScreen(left-1,top-1,right+1,bottom+1,BACKGROUND);
        WriteBlock(left-1,top-1,right+1,bottom+1,FALSE,TRUE,title);
        GotoXY((80-CountChar(st))/2+1,top); Write(st);
        GotoXY((80-CountChar(recommend))/2+1,bottom); Write(recommend);
        BlockReverse(left-1, top-1, right+1, bottom+1);
        while ( getch() != KENTER );
        ImagePut(left-1,top-1);
        SetTextViewPort(vp.left,vp.top,vp.right,vp.bottom);
}
void WriteBlock(int left, int top, int right, int bottom, char flagdouble, char flagbottom, char *title)
{
        char    startx, starty;

        startx = (left + right - CountChar(title)) / 2;
        starty = top;
        ClearScreen(left + 1, top, right - 1, top, BACKGROUND);
        left = (left - 1) * 8 + 4;
        right = right * 8 - 4;
        top = (top - 1) * 20 + 10;
        bottom = bottom * 20 - 10;
        setcolor(BACKGROUND);
        line(left + 1, top + 2, right - 1, top + 2);
        setcolor(FOREGROUND);

        if (left != right) {
                line(left, top, right, top);
                line(left, top, left, bottom);
        }
        line(right, top, right, bottom);
        if (flagbottom)
                line(left, bottom, right, bottom);
        if (flagdouble)
                line(left, top + 2, right, top + 2);
        if ( title[0] != '\0' ) { /* title != "" */
```

```
                left = GetTextX();
                top = GetTextY();
                ClearScreen(startx, starty, startx + CountChar(title) - 1, starty, BACKGROUND);
                GotoXY(startx, starty);
                Write(title);
                GotoXY(left, top);
        }
}

int   ViewMails( int mailno )
{
        char   temp1[128], temp2[80], temp3[80], temp4[80];
        int    posx = 0;
        int    posy = 0;
        int    count;
        int    i;
        char   row[5],col[5];
        char   key;
        char   filename[40];

        SetTextViewPort(1,1,80,24);
        ClearScreen(1,1,80,24,BACKGROUND);
        GotoXYShift(3,23,6);
        WriteShiftRvs(" F1 - ช่วยเหลือ   F2 - บันทึก   F3 - พิมพ์ออกเครื่องพิมพ์   F5 - ตรวจไฟล์ที่แนบ ",6,0);
        sprintf(temp1, " จาก   %s: ", MailFrame[mailno].Sendname);
        strcat(temp1, MailFrame[mailno].Sendfullname);
        sprintf(temp2, " เมื่อ   ");
        strcat(temp2, MailTime(MailFrame[mailno].lTime));
        sprintf(temp3, " เรื่อง   ");
        strcat(temp3, MailFrame[mailno].szMailSubject);
        sprintf(temp4," สิ่งที่ส่งมาด้วย: %d", MailFrame[mailno].byAttach);

        GotoXYShift( 3, 1, 12 );
        WriteShiftRvs( temp1, 12, 0 );
        GotoXYShift( 3, 2, 12 );
        WriteShiftRvs( temp2, 12, 0 );
        GotoXYShift( 3, 3, 12 );
        WriteShiftRvs( temp3, 12, 0 );
        GotoXYShift( 3, 4, 12 );
        WriteShiftRvs( temp4, 12, 0 );
        GotoXYShift(76,3,10);
        WriteCharShiftRvs(':',10,0);
        Block(6,9,633,473,1,WHITE);
        Block(9,12,630,470,0,WHITE);
```

```
bar(9,95,630,95);
bar(9,97,630,98);
bar(9,442,630,442);
bar(9,439,630,440);

sprintf(LoMai, "SYS:MAIL\\%lX\\%s", LocalID, MailFrame[mailno].MailContentID);

   if ( OpenFiles(LoMai, mailno) == 0 )   /* open file and put in buffer */
            return 0;
SetTextViewPort(STARTCOL, STARTROW, STOPCOL, STOPROW);

if (e_id > GetTextMaxY())
   ShowScreen(1, GetTextMaxY());      /* show on screen */
else
   ShowScreen(1, e_id);

GotoXY(posx,posy);

for (;;) {                              /* wait for pressed key */
   gcvt((posx=GetTextX()),2,col);
            gcvt( (posy=GetTextY())+c_id-1 ,(posy+c_id-1 < 100) ? 2 : 3,row);
   SetTextViewPort(1,1,80,24);
   GotoXYShift(73,3,10);
   WriteShiftRvs("   ",10,0);
   GotoXYShift((posy+c_id-1 < 100) ? 74 : 73,3,10);
   WriteShiftRvs(row,10,0);
   GotoXYShift(77,3,10);
   WriteShiftRvs("  ",10,0);
   GotoXYShift(77,3,10);
   WriteShiftRvs(col,10,0);
   SetTextViewPort(STARTCOL, STARTROW, STOPCOL, STOPROW);
   GotoXY(posx,posy);
   if ((key = GetKey()) == 0) {
     switch (key = getch())  {

            case KF3 :  posx=GetTextX();
                        posy=GetTextY();
                        if (!(biosprint(2, 0, 0) & 0x80 && (biosprint(2, 0, 0) & 0x29) == 0)) {
                                DialogBox("เครื่องพิมพ์ไม่ถูกต้องไม่สามารถพิมพ์ได้");
                        }
                                else{
                                 ImageGet( 25,5,45,7 );
                                 ClearScreen(25,5,45,7, BACKGROUNDR);
                                 GotoXY(27,6);
                                 WriteShiftRvs("กำลังพิมพ์ออกเครื่องพิมพ์",0,1);
```

```
                    strcpy(filename, tmpnam(NULL));
                    SaveFile(filename);
                    i = spawnlp(P_WAIT, "TMAILP.EXE", "TMAILP.EXE", filename, NULL);
                    if(i == -1)
                        DialogBox("เครื่องพิมพ์ไม่ถูกต้องไม่สามารถพิมพ์ได้");
                    unlink(filename);
                }
                GotoXY(posx,posy);
                break;

            case KF5 :  if( MailFrame[mailno].byAttach ){
                    posx=GetTextX();
                    posy=GetTextY();
                    ShowAtt(mailno);
                    GotoXY(posx,posy);
                    }
                break;
            case KHome :
                GotoXY(1, GetTextY());
                break;

            case KEnd :
                GotoXY(CountChar(sid[c_id+GetTextY()-1]) + 1, GetTextY());
                break;

            case KArrowLeft :  /* left arrow key  */
                if ( (posx=GetTextX()) != 1)
                    GotoXY(posx - 1, GetTextY());
                else{
                    if( c_id+(posy=GetTextY())-1  != 1 ){
                        if( posy != 1 )
                            GotoXY( ( (count=CountChar(sid[posy+c_id-2])) >= COLWIDTH ) ?
COLWIDTH : count+1,posy-1);
                        else{
                            ScrollUp();
                            Iflag = 2;
                            GotoXY( ( (count=CountChar(sid[posy+c_id-1])) >= COLWIDTH ) ?
COLWIDTH : count+1,posy);
                        }
                    }
                    else
                        BeepAlarm();
                }
                break;

            case KArrowRight :  /* right arrow key -> */
```

```
                    if ( (posx=GetTextX()) < COLWIDTH ){
                        if( posx <= (count=CountChar(sid[(posy=GetTextY())+c_id-1])) )
                            GotoXY(posx + 1, posy);
                        else{
ArrowRightposition:         if( posy+c_id-1 != e_id ){
                                if( posy != ROWWIDTH )
                                    GotoXY(1,posy+1);
                                else{
                                    ScrollDown();
                                    iflag = 2;
                                    GotoXY(1,posy);
                                }
                            }
                            else
                                BeepAlarm();
                        }
                    }
                    else
                        goto ArrowRightposition;
                    break;

                case KArrowUp :     /* press up-arrow key to scroll up */
                    posx = GetTextX();
                    posy = GetTextY();
                    if( c_id+posy > 2 )
                        count = CountChar(sid[c_id+posy-2]);
                    if ( posy != 1 )
                        GotoXY( (posx > count+1) ? ( (count+1 > COLWIDTH) ? COLWIDTH : count+1 ) :
posx, posy - 1);
                    else
                        if (GetTextY() <= 1 && c_id != 1){
                            ScrollUp();
                            iflag = 2;
                            GotoXY( (posx > count+1) ? ( (count+1 > COLWIDTH) ? COLWIDTH : count+1 )
: posx, posy);
                        }
                        else
                            BeepAlarm();   /* beep if scroll up the first line */
                    break;

                case KArrowDown :   /* press down-arrow key to scroll down */
                    posx = GetTextX();
                    posy = GetTextY();
                    if( c_id+posy <= e_id )
                    count = CountChar(sid[c_id+posy]);
                    if (posy < ROWWIDTH && posy+c_id-1 < e_id )
```

```
                                GotoXY( (posx > count+1) ? ( (count+1 > COLWIDTH) ? COLWIDTH : count+1 ) :
posx, posy+1);
                            else
                                if ( GetTextY() >= ROWWIDTH - 1 && e_id - c_id >= ROWWIDTH ){
                                    ScrollDown();
                                    Iflag = 2;
                                    GotoXY( (posx > count+1) ? ( (count+1 > COLWIDTH) ? COLWIDTH : count+1 )
: posx, posy);
                                }
                                else
                                    BeepAlarm();   /* beep if scrolldown the lastline */
                            break;

                        case KPgUp :
                            if (c_id == 1) {
                                if (GetTextY() == 1)
                                    BeepAlarm();
                                else
                                    GotoXY(GetTextX(), 1);
                            } else
                                ScrollPageUp();
                            break;

                        case KPgDown :
                            if( e_id-c_id <ROWWIDTH-1 ){
                                if( GetTextY() == e_id-c_id+1 )
                                    BeepAlarm();
                                else
                                    GotoXY(GetTextX(),e_id-c_id+1);
                                }
                            else
                                ScrollPageDown();
                            break;
                    }
                }
                else {
                  switch (key) {
                        case KESC:          /* Key ESC to save file and exit */
                                            if (e_id != 0) {
                                                for (i = e_id; i > 0; i--)
                                                    RemoveString();
                                            }
                                            MailFrame[mailno].fNotRead = FALSE;
                                            return 1;
                        default:
                            break;
```

```
            }
          }
        }

}

int OpenFiles(char * filename, int MailID)
{
        FILE * fp;
        char        outfile[40], *template="TMAILXXXXXX";  /* buffer character read from file */
        static      char Dummy[STRLEN];
        int         i,j;  /* index of other string */

        strcpy( Dummy, template );
        mktemp( Dummy );
        sprintf(outfile,"D:\\MAIL\\MAIL\\%s",Dummy);

        DecrypteFile(filename, outfile);

        if ( (fp = fopen(outfile, "rt")) == NULL ) {
                GotoXY(STARTCOL, GetTextY());
                Write("Open File Error...");
                return 0;
        }

        for(j = 0; j<MailFrame[MailID].byMailLength; j++) {
                fgets(Dummy, 128, fp);
                fgetc(fp);
        }

        rewind(fp);

        c_id = 1;
        e_id = i = 0; /* initialize array of string */

        AddString();

        for(i=0; i<MailFrame[MailID].byMailLength; i++) {
                fgets(sid[e_id],STRLEN,fp);
                sid[e_id][strlen(sid[e_id])-1] = '\0'; /* destruct '\n' */
                AddString();
```

```
        }

        if (fclose(fp) != 0)
                terminate("Close file error");

        unlink(outfile);
        return 1;   /* open and close not error */
}

void About(char *username)
{
        static char     title[9][60] = {
                "โครงงานจดหมายอิเล็กทรอนิกสภาษาไทย",
                "ภาคโทรคมนาคม คณะวิศวกรรมศาสตร์",
                "สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง",
                "",
                "นาย เจษฎา  ศิริยันต์   34102073",
                "นาย สายัณห์  เดี้ยงบุญเลิศชัย 34108419",
                "อ. เกรียงไกร  วงศ์โรจนภรณ์ ที่ปรึกษา",
                "",
                ""
        };

        int             maxlen, i, startx = 7, starty = 8;

        if(username[0] == '\0')
                strcpy(username, "* Un known *");
        sprintf(title[8], "ชื่อผู้ใช้ %s", username);
        for (i = 1, maxlen = CountChar(title[0]); i < 9; i++)
                if (maxlen < CountChar(title[i]))
                        maxlen = CountChar(title[i]);
        startx = (80-maxlen)  / 2 - 2;
        SetTextViewPort(1, 1, 80, 24);
        ImageGet(startx-1, starty-1, 80-startx + 3, starty + 10);
        ClearScreen(startx, starty, 80-startx + 3, starty + 10, DARKGRAY);
        ClearScreen(startx-1, starty-1, 80-startx + 2, starty + 9, BACKGROUNDR);
        Block( (startx-1)*8-5, (starty-1)*20-15, (80-startx + 2)*8-4, (starty + 9)*20-5,1,LIGHTBLUE);
        Block( (startx-1)*8-2, (starty-1)*20-12, (80-startx + 2)*8-7, (starty + 9)*20-8,0,LIGHTBLUE);
        GotoXY( startx+15, starty-1 );
        WriteShiftRvs("เกี่ยวกับ THAI E-MAIL",0,1);
        for (i = 0; i < 9; i++) {
                GotoXY((80 - CountChar(title[i])) / 2 + 1, starty + i);
                WriteShiftRvs(title[i],0,1);
        }

        /*      BlockReverse(startx-1,starty-1,80-startx+2,starty+8); */
```

```
        getch(); /* wait for press key */
        ImagePut(startx-1, starty-1);
        SetTextViewPort(9,6,69,20);
}

void    ShowAtt(int mailno)
{
        int  i;
        int  pos;
        char key;
        char ans;
        char path[40];

        ImageGet( 15, 5, 55, 13 );
        ClearScreen( 15,5,55,13,BACKGROUNDR);
        Block( 115,85,435,255,0,BLUE );
        Block( 118,88,432,252,1,BLUE);
        GotoXY(25,5);
        WriteShiftRvs("ไฟล์ที่แนบมาด้วย-Attach File",0,1);
        for( i=0; i<MailFrame[mailno].byAttach&&i<MAXATTACHFILE; i++ ){
            GotoXY( 22, 6+i);
            WriteShiftRvs(MailFrame[mailno].szAttachName[i],0,1);
        }
        if(MailFrame[mailno].byAttach){
            Highlight(MailFrame[mailno].szAttachName[0],21,6,40,0);
            pos = 1;
            while(1){
                if( (key = getch()) == 0 )
                switch( key = getch() ){
                    case KArrowUp :
                                        Highlight(MailFrame[mailno].szAttachName[pos-1],21,5+pos,40,1);
                                        if( pos != 1 )
                                            pos--;
                                        else
                                                pos = MailFrame[mailno].byAttach > MAXATTACHFILE ?
MAXATTACHFILE : MailFrame[mailno].byAttach;
                                        Highlight(MailFrame[mailno].szAttachName[pos-1],21,5+pos,40,0);
                                        break;

                    case KArrowDown :
                                        Highlight(MailFrame[mailno].szAttachName[pos-1],21,5+pos,40,1);
                                        if( pos != 5 ){
                                                if( pos == MailFrame[mailno].byAttach )
                                                pos = 1;
                                            else
                                                pos++;
```

```
                                        }
                                        else
                                            pos = 1;
                                        Highlight(MailFrame[mailno].szAttachName[pos-1],21,5+pos,40,0);
                                        break;
                                }
                            else
                                switch(key){
                                    case KENTER :
/*                                      ImageGetH(12,7,64,8);
                                        ClearScreen(12,7,64,8,WHITE);
                                        GotoXYShift( 13,7,10 );
                                        WriteShiftRvs("ต้องการเก็บไฟล์นี้ไว้ที่ด้วยชื่อเดิมในไดเร็คทอรีปัจจุบันหรือไม่ ( Y )
",10,0);
                                        if( (ans=getch()) == 'y' || ans == 'Y' ){
                                            sprintf( path,"D:\\MAIL\\ATT\\%s",MailFrame[mailno].szAttachID[pos-1]);
                                            CopyFile(path,MailFrame[mailno].szAttachName[pos-1]);
                                        }
                                        ImagePutH(12,7);
                                        ImagePut( 15, 5 );
                                        return;
                                    case KESC :
                                        ImagePut( 15, 5 );
                                        return;
*/
                                        SwapImage( 15, 5, 55, 13,"temp00.swp" );
                                        ImageGet(12,7,64,8);
                                        ClearScreen(12,7,64,8,BACKGROUNDR);
                                        GotoXYShift( 12,7,10 );
                                        WriteShiftRvs("  ต้องการเก็บไฟล์นี้ไว้ที่ด้วยชื่อเดิมในไดเร็คทอรีปัจจุบันหรือไม่ (Y/N)
",10,1);
                                        BlockReverse(12,7,64,8);
                                        if( (ans=getch()) == 'y' || ans == 'Y' ){
                                         sprintf( path,"SYS:MAIL\\%DX\\%s",LocalID,MailFrame[mailno].szAttachID[pos-1]);
                                         CopyFile(path,MailFrame[mailno].szAttachName[pos-1]);
                                        }
                                        ImagePut(12,7);
                                        RestoreImage( 15, 5, 55, 13,"temp00.swp",1 );
                                        goto endShow;
                                    case KESC :
                                        ImagePut( 15, 5 );
                                        return;
                                }
                        }
                }
                getch();
```

```
                ImagePut( 15, 5 );
endShow:
                ;
}



void SwapImage(int left, int top, int right, int bottom,char *file)
{
                FILE  *fp;

                left  = (left - 1) * 8;
                right = (right * 8) - 1;
                top   = (top - 1) * 20;
                bottom = (bottom * 20) - 1;

                if( ( fp=fopen(file,"wb") ) == NULL )
                                terminate("Error in Opening Swap File");
                fwrite(ScrBuffPtr,imagesize(left, top, right, bottom),1,fp);
                fclose(fp);
                farfree(ScrBuffPtr);
}


void RestoreImage(int left, int top, int right, int bottom,char *file,char del)
{
                FILE  *fp;

                left  = (left - 1) * 8;
                right = (right * 8) - 1;
                top   = (top - 1) * 20;
                bottom = (bottom * 20) - 1;

                if( ( fp=fopen(file,"rb") ) == NULL )
                                terminate("Error in Opening Swap File");

                if ((ScrBuffPtr = (unsigned char far * )farmalloc(imagesize(left, top, right, bottom))) == NULL)
                                terminate("Error in ImageGet");
                fread(ScrBuffPtr,imagesize(left, top, right, bottom),1,fp);
                putimage(left, top, ScrBuffPtr, COPY_PUT);
                fclose(fp);
                if( del )
                                unlink(file);
                farfree(ScrBuffPtr);
}
```

```
int functionsort( const void *a, const void *b )
{
        return( strcmp((char *)a,(char *)b) );
}
void SaveBindery(void)
{
        FILE        * fp;
        register int   i;

        if ( (fp= fopen("c:\\bintemp", "wb")) == NULL )
          terminate("\nOpen File to save bindery Error");
        fwrite( &AccountNumber, sizeof(WORD), 1, fp );
        for( i =0; i < AccountNumber; i++ ){
          fwrite( &AccountData[i], sizeof(AccountFrameType), 1, fp );
          //fwrite( AccountData+i*sizeof(AccountFrameType)+48, 80, 1, fp );
          //fwrite( AccountData+i*sizeof(AccountFrameType)+128, 4, 1, fp );
        }
        fclose(fp);
//      farfree(AccountData);
}

void  GetBindery(void)
{
        FILE        * fp;
        register int   i;

        if ( (fp= fopen("c:\\bintemp", "rb")) == NULL )
          terminate("\nOpen File to save bindery Error");
        fread( &AccountNumber, sizeof(WORD), 1, fp );
        if((AccountData = (AccountFrameType *) farmalloc(AccountNumber*sizeof(AccountFrameType)))==NULL)
          terminate("Error in GetBindery");
        for( i =0; i < AccountNumber; i++ )
          fread( &AccountData[i], sizeof(AccountFrameType), 1, fp );
        fclose(fp);
        unlink("c:\\bintemp");
}

void  Warning( char *msg )
{

        ImageGet(12,9,64,10);
        ClearScreen(12,9,64,10,BACKGROUND);
        Block(89,160,510,199,1,FOREGROUND);
        Block(92,163,507,196,0,FOREGROUND);

        GotoXYShift( 12+(52-(int)strlen(msg))/2,9,10 );
```

```
        WriteShiftRvs(msg,10,0);
        BlockReverse(12,9,64,10);
        getch();
        ImagePut(12,9);
}

void strcopy(char *str1,char *str2,char *str3 )
{
        while( *str3 != '\0' ){
          *str1 = *str3;
          *str2 = *str3;
          str1++;
          str2++;
          str3++;
        }
        *str1 = 0;
        *str2 = 0;
}

void strtoupr(char *str)
{
  while( *str != 0 ){
          *str = toupper( *str );
          str++;
  }
}
void Putbackground(void)
{
        FILE *fp;

        if( ( fp=fopen("wall.bkg","rb") ) == NULL )
                terminate("Error in Opening Swap File");

        if ((ScrBuffPtr = (unsigned char far * )farmalloc(imagesize(0, 0, 639, 159))) == NULL)
                terminate("Error in ImageGet");
        fread(ScrBuffPtr,imagesize(0, 0, 639, 159),1,fp);
        putimage(0, 0, ScrBuffPtr, COPY_PUT);
        putimage(0, 160, ScrBuffPtr, COPY_PUT);
        putimage(0, 320, ScrBuffPtr, COPY_PUT);
        fclose(fp);
        farfree(ScrBuffPtr);
}
void WriteVoice()
{
        char    key, filename[20] ;
        char    recv[80],*ptr;
```

```
int             i,ndif;
int   posx,posy,count;
char  row[5],col[5];
char  title[MaxSubLength];
char  temp[80];
LONG  RXID;
int   rxno;
char  difserv[MAXRECEIVER][20],difsvname[MAXRECEIVER][20];
int   *result;
int   j,k;
char  *command;
char  *msg;

SetTextViewPort(1,1,80,24);
Putbackground();
ClearScreen(5,5,78,22,DARKGRAY);
SetTextViewPort(3,4,77,21);
ClearScreen(1,1,74,18,BACKGROUND);
Block(4,8,588,351,1,WHITE);
Block(7,11,585,348,0,WHITE);
GotoXYShift(21,1,0);
WriteShiftRvs(" สร้างระบบบอดหมายเสียง-Make Voice Mail ",0,0);
Block(32,55,547,137,1,WHITE);
Block(35,58,544,134,0,WHITE);
cursortype = ENGCURSOR;
GotoXY(6,5);
Write(" ส่งให้ : [                                    ]");
GotoXY(6,6);
Write(" เรื่อง : [                                    ]");
GetInfo( 21,66,8,recv,"",0,0 );
strtoupr(recv);
if( ptr = strchr(recv,',') ){
        while( ( ptr = strchr(recv,',') )&&(recvcount <= 5) ){
          *ptr = 14;
          strcpy(temp,recv);
          *ptr = 0;
          strcpy(RecvName[recvcount++],recv);
          strcpy(recv,temp);
          ptr = strchr(temp,14)+1;
          strcpy(recv,ptr);
        }
        if( recvcount < 5 )
          strcpy(RecvName[recvcount++],recv);
}
else{
        strcpy(RecvName[recvcount++],recv);
```

```
}
for( i=0,ndif=0; i<recvcount; i++ )
        if( strchr(RecvName[i],'/')||strchr(RecvName[i],'\\') ){
          strcpy(difserv[ndif++],RecvName[i]);
          RecvName[i][0]=0;
        }
qsort( difserv, ndif, 20, functionsort);
for( i = 0; i < ndif; i++ ){
        ptr = ( strchr(difserv[i],'/') ? strchr(difserv[i],'/') : strchr(difserv[i],'\\') );
        strcpy(difsvname[i],ptr+1);
        *ptr = 0;
}
GetInfo( 21,66,9,title,"",0,0 );
sprintf(filename, "%d.cml", savefileindex++);

//Block(72,225,497,290,1,WHITE);
//Block(75,228,494,287,0,WHITE);
//GotoXYShift( 15,13,4 );
if( WriteVoiceMail(filename) )
        return;
for( i=0; i<recvcount; i++ ){
        if( RecvName[i][0] ){
                for( j = 0; j < AccountNumber; j++ ){
                        if( !strcmp( RecvName[i],(char *)AccountData[j].ObjectName ) ){
                                //RXID = AccountData[ GetMailID( RecvName[i],0 ) ].ObjectID;
                                RXID = AccountData[j].ObjectID;
                                SendMail( RXID, title, filename, attachfile, attcount, (BYTE)MailLength, 0,
"",1 );
                                break;
                        }
                        else
                                if( j == AccountNumber-1 ){
                                  sprintf(msg,"User : %s is not on this server",RecvName[i]);
                                  Warning(msg);
                                }
                }
        }
        else
          Warning("You 've not entered receiver");
}

        if( ndif ){
                k = 0;
                SaveBindery();
                for( i=0; i<ndif; i++ ){
                                if( !ill(strcmp(difserv[i],difserv[i-1])) ){
```

```
for( j=0; ( j<ServerNumber ); j++ ){
        if( !strcmp( difserv[i],(char *)ServerName[j] ) ){
          //sname = GetFileServerName();
          Attach('');
          for( k=0; k<8; k++ )
                if( !strcmp(difserv[i],Sname+48*k) ){
                        k = 15;
                        break;
                }

          if( k != 15 ){
                sprintf(command,"Attach %s/guest >
c:\\tempfile",difserv[i]);
                system(command);
                unlink("c:\\tempfile");
          }

          //Sname = GetFileServerName();
          Attach('');
          for( k=0; k<8; k++ ){
                if( !strcmp(difserv[i],Sname+48*k) ){
                 SetPreferredConnectionID( k+1 );
                 farfree( AccountData );
                 if(!openBindery())
                        terminate( "Can't open other
server's bindery");
                 k = 100;
                 break;
                }
          }
          if( k != 100 ){
                sprintf(msg,"Error can't attach to server : %
s",difserv[i]);
                Warning(msg);
          }
          break;
        }
        else
          if( j == (ServerNumber-1) ){
                sprintf(msg,"Server : %s is not exist",difserv[i]);
                Warning(msg);
          }
}

if( k == 100 ){
```

```
                                        sprintf(command,"map root *3:=%s\\sys:  >
c:\\temp001",difserv[i]);
                                        system(command);
                                        unlink("c:\\temp001");
                                        //SaveBindery();

                                }
                                else
                                        continue;
                        }
                        else{
                                if( k != 100 ){
                                        sprintf(msg,"Can't attach to server : %s or this server's not
exist",difserv[i]);
                                        Warning(msg);
                                        continue;
                                }
                        }
                        for( j = 0; j < AccountNumber; j++ ){
                                if( !strcmp( difsvname[i],(char *)AccountData[j].ObjectName ) ){
                                        RXID = AccountData[j].ObjectID;
                                        SendMail( RXID, title, filename, attachfile, attcount, (BYTE)
MailLength, 1, difserv[i],1 );
                                        break;
                                }
                        }
                        if( strcmp( difserv[i],difserv[i+1] )&&( i < ndif-1 ) ){
                                sprintf(command,"logout %s >c:\\tempfile",difserv[i] );
                                system(command);
                                unlink("c:\\tempfile");
                        }
                }
                farfree(AccountData);
                GetBindery();
        }
        SetPreferredConnectionID( 10 );
        attcount = 0;
        recvcount = 0;
        unlink(filename);

}


void ReadVoice()
{
```

```
int  i;
int  pos = 0;
int  start = 0, stop = 0;
char  key, *cmd;
char  path[80],ren[80];

SetTextViewPort(1,1,80,24);
Putbackground();
ClearScreen(5,5,78,22,DARKGRAY);
SetTextViewPort(3,4,77,21);
ClearScreen(1,1,74,18,BACKGROUND);
Block(4,8,588,351,1,WHITE);
Block(7,11,585,348,0,WHITE);
GotoXYShift(18,1,0);
WriteShiftRvs(" แฟ้มเก็บจดหมายทั้งหมด-Voice Mail Folder ",0,0);
//Block(160,18,392,18,0,WHITE);
GotoXYShift( 6, 2, 15 );
WriteShiftRvs("ผู้ส่ง-Sender     หัวเรื่อง-Title          วันเวลาส่ง-DATE   ยังไม่อ่าน", 15, 0 );
GotoXY( 4, 18 );
Write("^Enter-อ่าน");
GotoXY( 25, 18 );
Write("Del-ลบ");
GotoXY( 42, 18 );
Write("U-ไม่ลบ");
GotoXY( 56, 18 );
Write("Esc-ยกเลิก");
for( i = 0; ( i < 7 )&&( i < AllVocMail ); i++ ){
        GotoXY( 4, 5+i );
        Write( VocMailFrame[i].Sendname );
        GotoXY( 23, 5+i );
        Write( VocMailFrame[i].szMailSubject );
        GotoXY( 46, 5+i );
        Write( MailTime(VocMailFrame[i].lTime) );
        GotoXY( 67, 5+i );
        WriteChar( VocMailFrame[i].fNotRead ? 'Y' : 'N' );
        GotoXY( 71, 5+i );
        WriteChar( VocMailFrame[i].fDelMail ? MARKSELECT : ' ' );
        stop++;
}
start++;
if( AllVocMail ){
        Highlight(VocMailFrame[0].Sendname,3,5,22,1);
        Highlight(VocMailFrame[0].szMailSubject,22,5,45,1);
        Highlight(MailTime(VocMailFrame[0].lTime),45,5,66,1);
        Highlight(VocMailFrame[0].fNotRead ? "Y" : "N",66,5,69,1);
        GotoXY( 70, 5 );
```

```
Write( VocMailFrame[0].fDelMail ? '#' : ' ');
pos = 1;
while(1){
 if( (key = getch()) == 0 )
   switch( key = getch() ){

  case KDel :
                        VocMailFrame[pos+start-2].fDelMail = TRUE;
                        GotoXY( 70,4+pos );
                        WriteChar(' ');
                        WriteChar(MARKSELECT);
                        break;

  case KArrowUp :
                        if( pos != 1 ){

                                Highlight(VocMailFrame[pos+start-2].Sendname,3,4+pos,22,0);
                                Highlight(VocMailFrame[pos+start-2].szMailSubject,22,4+pos,45,0);
                                Highlight(MailTime(VocMailFrame[pos+start-2].lTime),45,4+pos,66,0);
                                Highlight(VocMailFrame[pos+start-2].fNotRead ? 'Y' : 'N',66,4+pos,69,0)
;
                                GotoXY( 70, 4+pos );
                                //WriteChar( VocMailFrame[pos+start-2].fDelMail ? MARKSELECT : ' ' );
                                Write( VocMailFrame[pos+start-2].fDelMail ? '#' : ' ');
                                pos--;
                                Highlight(VocMailFrame[pos+start-2].Sendname,3,4+pos,22,1);
                                Highlight(VocMailFrame[pos+start-2].szMailSubject,22,4+pos,45,1);
                                Highlight(MailTime(VocMailFrame[pos+start-2].lTime),45,4+pos,66,1);
                                Highlight(VocMailFrame[pos+start-2].fNotRead ? 'Y' : 'N',66,4+pos,69,1)
;
                                GotoXY( 70, 4+pos );
                                //WriteChar( VocMailFrame[pos+start-2].fDelMail ? MARKSELECT : ' ' );
                                Write( VocMailFrame[pos+start-2].fDelMail ? '#' : ' ');

                        }
                        else{
                                if( start != 1 ){
                                start--;
                                Highlight(VocMailFrame[pos+start-2].Sendname,3,4+pos,22,1);
                                Highlight(VocMailFrame[pos+start-2].szMailSubject,22,4+pos,45,1);
                                Highlight(MailTime(VocMailFrame[pos+start-2].lTime),45,4+pos,66,1);
                                Highlight(VocMailFrame[pos+start-2].fNotRead ? 'Y' : 'N',66,4+pos,69,1);
                                GotoXY( 70, 4+pos );
                                Write( VocMailFrame[pos+start-2].fDelMail ? '#' : ' ');
                                //WriteChar( VocMailFrame[pos+start-2].fDelMail ? MARKSELECT : '-' );
                                for( i = 0; ( i < stop-start )&&( i < 9 ); i++ ){
                                        Highlight(VocMailFrame[start+i].Sendname,3,6+i,22,0);
```

```
                                        Highlight(VocMailFrame[start+i].szMailSubject,22,6+i,45,0);
                                        Highlight(MailTime(VocMailFrame[start+i].lTime),45,6+
i,66,0);
                                        Highlight(VocMailFrame[start+i].fNotRead ? "Y" : "
N",66,6+i,69,0);
                                        GotoXY( 70, 6+i );
                                        Write( VocMailFrame[i+start].fDelMail ? "#" : " ");
                                        //WriteChar( VocMailFrame[start+i].fDelMail ?
MARKSELECT : ' ' );
                                }
                                if( stop-start >= 7 )
                                        stop--;
                                }
                                else
                                BeepAlarm();
                        }
                        break;
                case  KArrowDown :
                        if( ( pos != 7 )&&( pos+start-1 < AllVocMail ) ){
                                Highlight(VocMailFrame[pos+start-2].Sendname,3,4+pos,22,0);
                                Highlight(VocMailFrame[pos+start-2].szMailSubject,22,4+pos,45,0);
                                Highlight(MailTime(VocMailFrame[pos+start-2].lTime),45,4+pos,66,0);
                                Highlight(VocMailFrame[pos+start-2].fNotRead ? "Y" : "N",66,4+pos,69,0)
;
                                GotoXY( 70, 4+pos );
                                Write( VocMailFrame[pos+start-2].fDelMail ? "#" : " ");
                                //WriteChar( VocMailFrame[pos+start-2].fDelMail ? MARKSELECT : ' ' );
                                pos++;
                                Highlight(VocMailFrame[pos+start-2].Sendname,3,4+pos,22,1);
                                Highlight(VocMailFrame[pos+start-2].szMailSubject,22,4+pos,45,1);
                                Highlight(MailTime(VocMailFrame[pos+start-2].lTime),45,4+pos,66,1);
                                Highlight(VocMailFrame[pos+start-2].fNotRead ? "Y" : "N",66,4+pos,69,1)
;
                                GotoXY( 70, 4+pos );
                                Write( VocMailFrame[pos+start-2].fDelMail ? "#" : " ");
                                //WriteChar( VocMailFrame[pos+start-2].fDelMail ? MARKSELECT : ' ' );
                        }
                        else{
                                if( ( stop < AllVocMail )&&( pos+start-1 < AllVocMail ) ){
                                for( i = 0; i < stop-start; i++ ){
                                        Highlight(VocMailFrame[start+i].Sendname,3,5+i,22,0);
                                        Highlight(VocMailFrame[start+i].szMailSubject,22,5+i,45,0);
                                        Highlight(MailTime(VocMailFrame[start+i].lTime),45,5+i,66,0);
                                        Highlight(VocMailFrame[start+i].fNotRead ? "Y" : "
N",66,5+i,69,0);
                                        GotoXY( 70, 5+i );
```

```
                                        Write( VocMailFrame[start+i].fDelMail ? "#" : " ");
                                        //WriteChar( VocMailFrame[start+i].fDelMail ? MARKSELECT
        : ' ' );
                                    }
                                    start++;
                                    Highlight(VocMailFrame[pos+start-2].Sendname,3,4+pos,22,1);
                                    Highlight(VocMailFrame[pos+start-2].szMailSubject,22,4+pos,45,1);
                                    Highlight(MailTime(VocMailFrame[pos+start-2].lTime),45,4+pos,66,1);
                                    Highlight(VocMailFrame[pos+start-2].fNotRead ? "Y" : "N",66,4+pos,69,1);
                                    GotoXY( 71, 4+pos );
                                    Write( VocMailFrame[pos+start-2].fDelMail ? "#" : " ");
                                    //WriteChar( VocMailFrame[pos+start-2].fDelMail ? MARKSELECT : ' ' );
                                    stop++;
                                    }
                                    else
                                    BeepAlarm();
                                }
                                break;
                            }
                            else
                             switch( key ){
                            case 'U' :
                            case 'u' :
                                    VocMailFrame[pos+start-2].fDelMail = FALSE;
                                    GotoXY( 70,4+pos );
                                    WriteChar(' ');
                                    break;
                            case KESC :
                                    SetTextViewPort(1,1,80,24);
                                    return;
                            case KENTER :
                                    sprintf(path, "SYS:MAIL\\%X\\%s", LocalID,VocMailFrame[start+pos-2].MailContentID);
                                    ReadVoiceMail(path);
                                    if(kbhit())
                                        getch();
                                    return;
                            default :
                                    break;
                             }
                           }
                    }
                    else
                      getch();
            }
```