ภาควิชาครุศาสตร์วิศวกรรม

คณะครุศาสตร์อุตสาหกรรม

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ใบรับรองปริญญานิพนธ์

หัวข้อปริญญานิพนธ์ โปรแกรมควบคุมเครื่องเจาะแผ่นวงจรพิมพ์อัตโนมัติ

SOFTWARE CONTROL AUTOMATIC PCB DRILLER

ชื่อนักศึกษา

1. นายชาญณรงค์	ปิ่นทอง	รหัสประจำตัว	40031409
2. นายธีรพล	หว่านณรงค์	รหัสประจำตัว	40031415
3. นายสมบูรณ์	กิจวัฒนาบูลย์	รหัสประจำตัว	40031431

หลักสูตร ครุศาสตร์อุตสาหกรรมบัณฑิต **สาขาวิชา** อิเล็กทรอนิกส์และคอมพิวเตอร์

อาจารย์ผู้ควบคุมปริญญานิพนธ์

1. อาจารย์โกศล ตราชู
2. อาจารย์สุรพงษ์ สิริพงศ์ดี

คณะกรรมการสอบปริญญานิพนธ์		ลายมือชื่อ
1. อาจารย์โกศล	ตราชู	.....
2. อาจารย์อำพล	ทองระอา	.....
3. อาจารย์ไพบูลย์	พวงวงศ์ตระกูล	.....
4. อาจารย์อมรชัย	ชัยชนะ	.....
5. อาจารย์สุระชัย	พิมพ์สาลี	.....

วัน เดือน ปีที่สอบ วันที่ 24 พฤศจิกายน 2541 เวลา 02.30 น.

สถานที่สอบ ห้อง ค.310 คณะครุศาสตร์อุตสาหกรรม

ภาควิชารับรองแล้ว

ลงชื่อ.....

(ผศ.ดร.ธีรพล เทพหัสดิน ณ อยุธยา)

หัวหน้าภาควิชาครุศาสตร์วิศวกรรม

วันที่ 12 เดือน ก.พ. พ.ศ. 42

ปริญญานิพนธ์

โปรแกรมควบคุมเครื่องเจาะแผ่นวงจรพิมพ์อัตโนมัติ

SOFTWARE CONTROL AUTOMATIC PCB DRILLER

นายชาญณรงค์ ปิ่นทอง
นายธีรพล หว่านณรงค์
นายสมบูรณ์ กิจวัฒนาบูลย์

ปริญญานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรครุศาสตร์อุตสาหกรรมบัณฑิต
สาขาวิชาอิเล็กทรอนิกส์และคอมพิวเตอร์
ภาควิชาครุศาสตร์วิศวกรรม คณะครุศาสตร์อุตสาหกรรม
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2541

การเขียนโปรแกรมอย่างมีโครงสร้างมีหลายวิธี วิธีที่นิยมใช้มีดังนี้

1. การเขียนโปรแกรมจากบนลงล่าง (Top-down programming)

วิธีนี้นิยมใช้กันอย่างแพร่หลาย โดยเริ่มจากการกำหดนส่วนของโปรแกรมหลักเสียก่อน โดยพยายามกำหนดคำสั่งออกเป็นส่วนๆ คำสั่งในแต่ละส่วนจะหมายถึงโปรแกรมย่อยหนึ่งโปรแกรม เมื่อเขียนโปรแกรมหลักเสร็จสิ้นแล้ว ก็จะเริ่มในส่วนโปรแกรมย่อย โดยการเขียนเป็นกลุ่มคำสั่งที่ทำหน้าที่ตามที่กำหนด ในกลุ่มคำสั่งก็อาจประกอบไปด้วยคำสั่งที่ภาษานั้นได้นิยามขึ้นหรืออาจเป็นโปรแกรมย่อยก็ได้ หากเป็นโปรแกรมย่อย ก็จะมีการเขียนโปรแกรมย่อยลงไปอีก การกระโดดและการวนรอบในการทำงาน ก็จะเกิดขึ้นภายในโปรแกรมย่อยนั้นๆ จะไม่มีการกระโดดข้ามโปรแกรมย่อยเป็นอันขาด

2. การเขียนโปรแกรมแบบโครงสร้างโมดูล (Modular programming)

สิ่งหนึ่งที่พบเมื่อได้เขียนโปรแกรมไปเป็นจำนวนหนึ่งก็คือจะต้องเขียนโปรแกรมย่อยที่ทำหน้าที่เดิมในโปรแกรมใหม่ หากแยกเอาโปรแกรมย่อยที่ได้เคยเขียนออกมาเก็บไว้ และนำไปใช้ในโปรแกรมใหม่ก็จะเป็นการลดเวลาในการเขียนโปรแกรมลงไปมาก นี่จึงเป็นที่มาของการโปรแกรมแบบโครงสร้างโมดูล โดยการเขียนโปรแกรมย่อย ซึ่งในกรณีนี้เราจะเรียกว่าโมดูล (module) และจัดโมดูลประเภทเดียวกันไว้เป็นกลุ่ม เมื่อเขียนโปรแกรมและต้องการการทำงานในรูปแบบที่ได้เขียนขึ้นแล้ว ก็จะนำโมดูลที่เขียนขึ้นรวมไว้กับโปรแกรม

3. การเขียนโปรแกรมจากล่างขึ้นบน (Bottom-up programming)

ในบางกรณีที่ไม่สามารถที่จะร่างโปรแกรมจากบนลงล่างได้ อาจเนื่องจากไม่ทราบลักษณะของการทำงานอย่างชัดเจน หรือยังไม่สามารถกำหนดตัวโปรแกรมย่อยได้ ก็อาจเริ่มจากการรวมคำสั่งเป็นโปรแกรมย่อยในระดับที่เรียกใช้โปรแกรมย่อยเหล่านั้นภายหลัง ลักษณะเช่นนี้เป็นการเขียนโปรแกรมจากส่วนย่อยไปหาส่วนหลัก จึงเรียกว่าเป็นโปรแกรมจากล่างขึ้นบน

ในการเขียนโปรแกรมจะใช้ทั้งสามวิธีนี้ร่วมกัน โดยใช้หลักการเขียนโปรแกรมจากบนลงล่าง ในภาพรวม ใช้การเขียนโปรแกรมแบบโครงสร้างโมดูล ในรูปของโมดูลมาตรฐานที่เราจะเขียนขึ้นเพื่อไว้ใช้ในภายหลังหรือใช้ในโมดูลที่เคยเขียนไว้ และการเขียนโปรแกรมจากล่างขึ้นบนในจุดย่อยต่างๆ ของโปรแกรม

4. การเขียนโปรแกรมเชิงวัตถุ (Object Oriented programming)

นอกจากมีวิธีการเขียนโปรแกรมในรูปแบบต่างๆ ที่กล่าวมาแล้ว ปัจจุบันได้มีการตื่นตัวในการเขียนโปรแกรมในรูปแบบเชิงวัตถุ การเขียนโปรแกรมในรูปแบบนี้ จะมองทุกๆ ส่วนของโปรแกรมให้เป็นวัตถุ (object) โดยการกำหนดวัตถุจากกลุ่มของการนิยามในโปรแกรมไว้และกำหนดลำดับความสำคัญในการเรียกใช้ มีลักษณะเป็นคลาส (class) ในแต่ละคลาสจะประกอบด้วย

ปริญญานิพนธ์

เรื่อง โปรแกรมควบคุมเครื่องเจาะแผ่นวงจรพิมพ์อัตโนมัติ

SOFTWARE CONTROL AUTOMATIC PCB DRILLER

วัตถุประสงค์ของปริญญานิพนธ์

1. ศึกษาการเขียนโปรแกรมภาษาซี
2. ศึกษาลักษณะและรูปแบบของไฟล์ที่เขียนวงจรโดยโปรแกรมโปรเทล
3. ศึกษาถึงวิธีการเขียนโปรแกรมติดต่อกับอุปกรณ์ภายนอก
4. ออกแบบโปรแกรมควบคุมการทำงานของเครื่องเจาะแผ่นวงจรพิมพ์อัตโนมัติ
5. สร้างโปรแกรมควบคุมการทำงานของเครื่องเจาะแผ่นวงจรพิมพ์อัตโนมัติ
6. ทดสอบการติดต่อของโปรแกรมควบคุมการทำงานของเครื่องเจาะแผ่นวงจรพิมพ์อัตโนมัติกับเครื่องเจาะแผ่นวงจรพิมพ์อัตโนมัติ
7. นำโปรแกรมควบคุมการทำงานของเครื่องเจาะแผ่นวงจรพิมพ์อัตโนมัติไปใช้งานจริงได้

ประโยชน์ที่คาดว่าจะได้รับ

1. มีความรู้เกี่ยวกับการเขียนโปรแกรมภาษาซี
2. มีความรู้เกี่ยวกับลักษณะและรูปแบบของไฟล์ที่เขียนวงจรโดยโปรแกรมโปรเทล
3. มีความรู้เกี่ยวกับการเขียนโปรแกรมควบคุมการทำงานของเครื่องเจาะแผ่นวงจรพิมพ์อัตโนมัติติดต่อสื่อสารผ่านคอมพิวเตอร์ไปยังเครื่องไมโครโปรเซสเซอร์
4. ได้ผังการทำงานของโปรแกรมควบคุมการทำงานของเครื่องเจาะแผ่นวงจรพิมพ์อัตโนมัติ
5. ได้โปรแกรมควบคุมการทำงานของเครื่องเจาะแผ่นวงจรพิมพ์อัตโนมัติ
6. ได้ทดสอบประสิทธิภาพของโปรแกรมควบคุมการทำงานของเครื่องเจาะแผ่นวงจรพิมพ์อัตโนมัติ
7. ได้โปรแกรมควบคุมการทำงานของเครื่องเจาะแผ่นวงจรพิมพ์อัตโนมัติไปใช้ในการควบคุมเครื่องเจาะแผ่นวงจรพิมพ์อัตโนมัติ

โปรแกรมควบคุมเครื่องเจาะแผ่นวงจรพิมพ์อัตโนมัติ

นายชาญณรงค์ ปิ่นทอง
นายธีรพล หว่านณรงค์
นายสมบูรณ์ กิจวัฒนาบูลย์

อาจารย์ที่ปรึกษา
อาจารย์โกศล ตราชู
อาจารย์สุรพงษ์ สิริพงศ์ดี

ปีการศึกษา 2541

บทคัดย่อ

ปริญญานิพนธ์ฉบับนี้ เสนอโปรแกรมควบคุมเครื่องเจาะแผ่นวงจรพิมพ์อัตโนมัติซึ่งเขียนด้วยภาษาซี โดยการนำข้อมูลการเจาะมาจากโปรแกรมโปรเทล (Protel) มาแปลงรูปแบบไฟล์ให้เป็นตำแหน่งของรูเจาะแล้วทำการส่งข้อมูลผ่านพอร์ตอนุกรม RS-232C ของเครื่องคอมพิวเตอร์ (PC) ไปยังไมโครคอนโทรลเลอร์ 8051 ของเครื่องเจาะแผ่นวงจรพิมพ์อัตโนมัติ ซึ่งจะทำงานได้ 2 โหมดการทำงานคือ การติดต่อสื่อสารข้อมูลกันระหว่างเครื่องคอมพิวเตอร์กับไมโครคอนโทรลเลอร์ 8051 ของเครื่องเจาะแผ่นวงจรพิมพ์อัตโนมัติและการติดต่อสื่อสารข้อมูลโดยตรงผ่านทางคีย์บอร์ดของเครื่องคอมพิวเตอร์ นอกจากนี้โปรแกรมควบคุมเครื่องเจาะแผ่นวงจรพิมพ์อัตโนมัตินี้ยังสามารถแสดงลายวงจรที่เขียนขึ้นโดยโปรแกรมโปรเทลด้วย เพื่อให้สะดวกในการเลือกไฟล์ที่จะส่งข้อมูล

SOFTWARE CONTROL AUTOMATIC PCB DRILLER

MR.CHANNARONG PINTHONG
MR.THEERAPON WANNARONG
MR.SOMBOON KITWATTANABOOL

ADVISORS

MR.KOSON TRACHU
MR.SURAPONG SIRIPONGDEE

1998

ABTRACT

This thesis presents the SOFTWARE CONTROL AUTOMATIC PCB DRILLER. It was written in C Language. It is takes data from Protel PCB program and convert into DRL format. Afterthat program will send data in DRL format to microcontroller (8051 chip) for controlling the AUTOMATIC PCB DRILLER via serial communication port.

The operating of this software has 2 functions. The first, drilling posittion will be send from PC to AUTOMATIC PCB DRILLER automatically and second is manual control directly from PC keyboard. Finally this program can display the PCB in graphics mode same as Protel PCB Program.

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้จะไม่สามารถสำเร็จลุล่วงไปได้ด้วยดีถ้ามีเพียงแต่คณะผู้จัดทำ หากแต่ยังมีความช่วยเหลือ และความอนุเคราะห์จากอาจารย์ทุกท่านในภาควิชาครุศาสตร์วิศวกรรม อีกทั้งยังความช่วยเหลือจากเพื่อน ๆ พี่ ๆ น้อง ๆ ในทุก ๆ สาขาวิชา

ขอบคุณเป็นพิเศษท่านอาจารย์โกศล ตราชูซึ่งได้ให้ความช่วยเหลือทั้งทางด้านฮาร์ดแวร์และซอฟต์แวร์ อีกทั้งยังคอยให้คำปรึกษา กำลังใจและความช่วยเหลืออื่น ๆ และขอขอบคุณเป็นพิเศษแด่ บิดา มารดา พี่น้องญาติผู้ใหญ่ ซึ่งให้ชีวิตและการเลี้ยงดูอุปการะคุณ และให้กำลังใจและให้กำลังทรัพย์ในการทำปริญญานิพนธ์ชิ้นนี้ คุณความดีครั้งนี้ขอมอบให้ บิดา มารดา และครูอาจารย์

สารบัญ

สารบัญ (ต่อ)

สารบัญ (ต่อ)

สารบัญตาราง

สารบัญรูปภาพ

สารบัญรูปภาพ (ต่อ)

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

ปัจจุบันในการที่จะสร้างสิ่งประดิษฐ์ทางอิเล็กทรอนิกส์ได้นั้นประกอบด้วยส่วนสำคัญ ๆ หลายส่วนด้วยกัน วงจรพิมพ์ก็เป็นส่วนสำคัญอีกส่วนหนึ่ง ในอดีตการผลิตแผ่นวงจรพิมพ์จะต้องทำการวาดลายวงจรและทำการกัดลายวงจรด้วยน้ำยากัดลายวงจรเอง รวมทั้งการเจาะแผ่นวงจรพิมพ์สำหรับใส่อุปกรณ์ด้วยสว่านมือ ซึ่งถ้าแผ่นวงจรพิมพ์มีขนาดใหญ่และหรือมีจำนวนมากจะทำให้เกิดความล่าช้าและผลผลิตที่ได้มีความแม่นยำต่ำ ดังนั้นจึงได้มีการคิดค้นเครื่องเจาะแผ่นวงจรพิมพ์อัตโนมัติซึ่งจะช่วยประหยัดเวลาในการเจาะรูสำหรับใส่อุปกรณ์และเพิ่มความแม่นยำได้มาก

เครื่องเจาะแผ่นวงจรพิมพ์อัตโนมัติประกอบด้วย 2 ส่วนใหญ่ ๆ คือฮาร์ดแวร์ (HARDWARE) และ ซอฟต์แวร์ (SOFTWARE) โดยในส่วนของฮาร์ดแวร์จะเป็นส่วนของเครื่องจักรกลต่าง ๆ และในส่วนของซอฟต์แวร์จะเป็นส่วนของโปรแกรมควบคุมการทำงานเครื่องเจาะแผ่นวงจรพิมพ์อัตโนมัติให้ทำงานได้ถูกต้อง แม่นยำและมีประสิทธิภาพโดยโปรแกรมที่ได้นี้เขียนขึ้นจากภาษา C

โปรแกรมควบคุมการทำงานเครื่องเจาะแผ่นวงจรพิมพ์อัตโนมัติจะทำหน้าที่อ่านข้อมูลจากโปรแกรมโปรเทล (Protel) แล้วทำการแปลงรูปแบบข้อมูลไฟล์สกุล PCB หรือ DRL ให้อยู่ในรูปแบบข้อมูลที่เครื่องเจาะแผ่นวงจรพิมพ์อัตโนมัติสามารถเข้าใจได้ จากนั้นจะทำการส่งข้อมูลที่ทำการแปลงแล้วให้กับเครื่องเจาะแผ่นวงจรพิมพ์อัตโนมัติ เพื่อให้เครื่องเจาะแผ่นวงจรพิมพ์อัตโนมัติได้เคลื่อนหัวเจาะไปตามข้อมูลที่ได้รับแล้วจึงทำการเจาะพร้อมตรวจสอบสภาพการทำงานของเครื่องเจาะแผ่นวงจรพิมพ์อัตโนมัติ

คณะผู้จัดทำมีแนวคิดที่จะจัดทำโครงงานโปรแกรมควบคุมเครื่องเจาะแผ่นวงจรพิมพ์อัตโนมัติเพื่อนำไปใช้ควบคุมเครื่องเจาะแผ่นวงจรพิมพ์อัตโนมัติและใช้ประโยชน์ในการศึกษา ซึ่งคณะผู้จัดทำได้มองเห็นว่าในกรณีที่เราต้องเจาะแผ่นวงจรพิมพ์ที่มีลายวงจรเดียวกันหรือต่างลายวงจรจำนวนหลายๆ แผ่น น่าที่จะมีวิธีการเจาะที่เที่ยงตรงแม่นยำและรวดเร็วและไม่ต้องออกแรงเอง ทางคณะผู้จัดทำจึงได้คิดพัฒนาโปรแกรมควบคุมเครื่องเจาะแผ่นวงจรพิมพ์อัตโนมัติขึ้นมา ซึ่งโปรแกรมควบคุมเครื่องเจาะแผ่นวงจรพิมพ์อัตโนมัติจะกล่าวโดยละเอียดในปริญญานิพนธ์ฉบับนี้ต่อไป

1.2 ขีดความสามารถของโครงงาน

1. สามารถอ่านไฟล์ข้อมูลโปรเทล .PCB หรือ .DRL ได้
2. สามารถทำการบันทึกไฟล์ทั้งแบบ .PCB หรือ .DRL ได้
3. สามารถเลือกพอร์ต COM1 หรือ COM2 ในการติดต่อกับเครื่องเจาะแผ่นวงจรพิมพ์ได้
4. สามารถเซต Boud rate , Parity bit , Data bit และ Stop bit เพื่อใช้ในการส่งข้อมูลได้
5. สามารถปรับเปลี่ยนมุมมองเป็นแบบการขยายเข้าหรือการขยายออกก็ได้
6. สามารถพิมพ์ข้อมูลต่าง ๆ ออกทางเครื่องพิมพ์ได้ทั้งในรูปของลายวงจรหรือตำแหน่งรูเจาะ
7. สามารถเลือกการทำงานแบบอัตโนมัติ, ควบคุมด้วยตัวเองและควบคุมตำแหน่งรูเจาะด้วย คีย์บอร์ด
8. สามารถเรียกดูวิธีใช้โปรแกรมได้
9. สามารถเรียกดูรายชื่อคณะผู้จัดทำได้

1.3 เนื้อหาโดยสังเขป

บทที่ 1 บทนำ ซึ่งจะกล่าวถึงความเป็นมาและความสำคัญของปัญหา ขีดความสามารถของโครงงานและเนื้อหาในส่วนของบทต่างๆ โดยสังเขป

บทที่ 2 ทฤษฎีและหลักการซึ่งจะกล่าวถึงหลักการพื้นฐานในการเขียนโปรแกรมในลักษณะวิธีการแก้ไขปัญหาคำสั่งเทียมโฟลว์ชาร์ตและการเขียนโปรแกรมโครงสร้าง ภาษาซีเบื้องต้น การเขียนโปรแกรมติดต่อกับระบบ โครงสร้างของระบบปฏิบัติการดอส การติดต่อกับพอร์ต การเขียนโปรแกรมแบบกราฟฟิกส์ รูปแบบไฟล์ข้อมูลโปรเทล

บทที่ 3 การออกแบบและสร้างซึ่งจะกล่าวถึงการออกแบบส่วนของเมนูส่วนของการติดต่อกับไฟล์ส่วนของการแสดงลายวงจรส่วนของการรับ-ส่งข้อมูลส่วนของโหมดการทำงานส่วนของการพิมพ์ไฟล์ข้อมูล .PCB และ .DRL ออกทางเครื่องพิมพ์ ส่วนของการติดต่อระหว่างโปรแกรมกับเครื่องเจาะแผ่นวงจรพิมพ์

บทที่ 4 การทดลองและผลการทดลองการทำงานของโปรแกรม ในการอ่านไฟล์ข้อมูล .PCB และ .DRL แล้วแสดงลายวงจร การส่งข้อมูลออกทางเครื่องพิมพ์ และการส่งข้อมูลออกทางเครื่องเจาะแผ่นวงจรพิมพ์

บทที่ 5 การสรุปและวิจารณ์ซึ่งจะกล่าวถึงปัญหาที่พบในการทำงาน การแก้ไขปัญหา ประโยชน์ที่ได้รับจากโครงงานและแนวทางในการพัฒนา

ภาคผนวก ก การใช้โปรแกรมควบคุมเครื่องเจาะแผ่นวงจรพิมพ์อัตโนมัติ

ภาคผนวก ข โปรแกรมภาษาซี

บทที่ 2
ทฤษฎีและหลักการ

2.1 หลักการพื้นฐานในการเขียนโปรแกรม

2.1.1 วิธีแก้ไขปัญหา

โดยทั่วไป เมื่อมีปัญหาที่จะต้องให้คอมพิวเตอร์ช่วยทำงานเกิดขึ้น ก็จะต้องมีแนวทางแก้ไขปัญหาที่เหมาะสม เพื่อให้การแก้ไขปัญหาเป็นไปอย่างมีประสิทธิภาพ

การแก้ไขปัญหาโดยคอมพิวเตอร์มีลำดับขั้นตอนดังนี้คือ

1. วิเคราะห์ปัญหา (Problem Analysis)

เมื่อมีปัญหาที่ต้องการแก้ไข สิ่งแรกที่เราจะต้องทำคือ พยายามหาเป้าหมายของปัญหา แบ่งปัญหาออกเป็นส่วนๆ การพิจารณาปัญหาจะเริ่มจาก

1.1 การกำหนดขอบเขตของปัญหา (Problem Definition) เราจะต้องกำหนดรายละเอียดของปัญหาออกมาโดยชัดเจน กำหนดขอบเขตการแก้ไข เพื่อเป็นแนวทางและกำกับในการตัดสินใจ ใช้ค่าคงที่หรือตัวแปรที่เหมาะสมต่อไป หากเราไม่กำหนดขอบเขต ก็จะเกิดความยุ่งยากในการตัดสินใจว่าจะต้องใช้แนวทางหรือข้อมูลอย่างไร

1.2 การกำหนดคุณสมบัติของข้อมูลเข้าและออกจากระบบ (Input/Output Specification) เมื่อกำหนดขอบเขตปัญหาได้แล้วเราก็จะกำหนดเส้นทางและวิธีการที่ข้อมูลจะเข้าและออกจากระบบ เช่น กรรมวิธีการรับค่าจากคีย์บอร์ด การใช้เมาส์ กำหนดปุ่มต่างๆ สำหรับการใช้งานในโปรแกรม ลักษณะการแสดงบนหน้าจอ และเครื่องพิมพ์ เป็นต้น

1.3 การกำหนดโครงสร้างข้อมูลที่ใช้ในโปรแกรม และข้อมูลที่เข้าและออกของโปรแกรม (Global Variable Definition) หลังจากนั้นเราจะกำหนดโครงสร้างข้อมูลที่ใช้ภายในโปรแกรม และชนิดของตัวแปรที่เข้าและออกจากระบบ

2. ออกแบบขั้นตอนการแก้ไขปัญหา (Algorithm Design)

เมื่อกำหนดคุณสมบัติของข้อมูลเข้าและออกแล้ว เราก็จะเริ่มหาวิธีการที่ใช้ในการแก้ไขปัญหา หรืออีกนัยหนึ่งก็คือ เขียนโครงสร้างของโปรแกรมขึ้นมา

2.1 ร่างแนวการแก้ไขปัญหา จะยึดหลักการต่อไปนี้

- การกำหนดวิธีการแก้ไขปัญหาโดยยึดหลักจากก่อนมาหลัง ในกรรมวิธีการแก้ไขปัญหาจะประกอบด้วยขั้นตอนย่อยๆ หลายขั้นตอนเรียงกัน ขั้นตอนที่ต้องทำก่อนก็จะต้องอยู่ก่อนขั้นตอนที่จะทำภายหลัง

- การแก้ไขปัญหาที่ยุ่งยาก จำเป็นจะต้องมองปัญหาในภาพรวมเสียก่อน แล้วในแต่ละจุดจะกำหนดรายละเอียดย่อยลงไป หากในจุดใดที่ยังไม่ชัดเจน ก็จะกำหนดย่อยลงไปจนชัดเจน

2.2 แปลงให้อยู่ในรูปอัลกอริทึม เมื่อได้โครงร่างแนวการแก้ไขปัญหาแล้ว ก็จะนำไปเขียนเป็นอัลกอริทึม (Algorithm) ซึ่งอาจเรียกได้ว่าเป็นภาษากลางของคอมพิวเตอร์ในขั้นตอนนี้เราจะมีการกำหนดตัวแปรภายในแต่ละกลุ่มคำสั่งที่จะต้องใช้กรรมวิธีในการคำนวณและวิธีการนำค่าเข้าและค่าออกอย่างชัดเจน

3. ใช้คอมพิวเตอร์ในการแก้ไขปัญหา (Computer Solution)

ปัญหาโดยส่วนใหญ่ ไม่จำเป็นต้องมีการใช้คอมพิวเตอร์ แต่ถ้าหากเป็นปัญหาที่ความซับซ้อน หรือต้องใช้ในการคำนวณซ้ำซาก ก็จะนำคอมพิวเตอร์มาใช้ในการแก้ปัญหา การแก้ปัญหาในขั้นตอนสุดท้ายมีดังนี้

3.1 เขียนโปรแกรม (Writing a program) หลังจากที่ได้อัลกอริทึมมาแล้ว ก็จะนำมาเปลี่ยนเป็นภาษาที่ต้องการเช่น ภาษาซี ภาษาปาสคาล และภาษาโคบอล เป็นต้น

3.2 สั่งให้โปรแกรมทำงาน (Run a program) เมื่อเขียนโปรแกรมเสร็จ ก็จะต้องมีการทดสอบทำงานเพื่อหาจุดผิดพลาด และหากมีที่ผิด ก็จะทำการแก้ไขจนกว่าจะตรวจสอบไม่พบอีก จุดที่ผิดนี้เราเรียกว่าบัก (bug) คำนี้มีที่มาจากความผิดพลาดในการทำงานของคอมพิวเตอร์

3.3 จัดทำคู่มือ (Documentation) เพื่อให้ผู้ใช้สามารถใช้งานโปรแกรมได้อย่างมีประสิทธิภาพ รวมถึงความสะดวกในการตรวจแก้ไขข้อผิดพลาด รวมทั้งการพัฒนาโปรแกรมในคราวต่อไป คู่มือการใช้งานแบ่งออกเป็น 2 ประเภท

- คู่มือที่ใช้อำนวยความสะดวกในการใช้งานโปรแกรม
- คู่มือที่ใช้อำนวยความสะดวกในการแก้ไขโปรแกรมและการพัฒนาโปรแกรม

4. บำรุงรักษาโปรแกรม (Maintalnance)

เมื่อได้จัดทำโปรแกรมพร้อมคู่มือเสร็จสิ้นแล้ว จำเป็นที่ผู้เขียนโปรแกรมจะต้องคอยตรวจสอบการใช้งานจริงของโปรแกรม เพื่อแก้ไขข้อผิดพลาดที่อาจจะเกิดขึ้นได้ในภายหลัง รวมทั้งการพัฒนาโปรแกรมให้เหมาะสมกับการใช้งานเมื่อเวลาผ่านไป

2.1.2 คำสั่งเทียม (Pseudocode)

การเขียนคำสั่งเทียมจะมีลักษณะคล้ายคลึงกับร่างที่เราร่างในขั้นต้น แต่จะมีรูปแบบชัดเจนและกำหนดตัวแปรแน่นอน

2.1.3 ผังการทำงาน (Flowchart)

ผังการทำงาน (Flowchart) คือแผนภาพที่ใช้อธิบายการทำงานโดยอาศัยรูปทรงต่างๆ ควบไปกับลูกศร โดยในแต่ละรูปจะหมายถึงการทำงานหนึ่งขั้นตอน และลูกศรจะแทนลำดับและทิศทางในการทำงานขั้นตอนต่างๆ

2.1.4 การเขียนโปรแกรมอย่างมีโครงสร้าง (Non-structureal programmimg)

การเขียนโปรแกรมอย่างมีโครงสร้างคือการเขียนโปรแกรมในลักษณะที่จะเรียงคำสั่งกันไปตั้งแต่คำสั่งแรกจนถึงคำสั่งสุดท้าย คำสั่งทั้งหมดจะอยู่กลุ่มคำสั่งเพียงกลุ่มเดียว และตัวแปรที่ใช้ในกลุ่มคำสั่งก็จะเป็นตัวแปรชุดเดียวกันทั้งหมด นั่นคือ หากโปรแกรมมีความยาวมากๆ ผู้เขียนโปรแกรมก็อาจมีความสับสนในการใช้ตัวแปร และอาจจะใช้ตัวแปรซ้ำกัน ทำให้เกิดความผิดพลาดได้ รวมทั้งการหาจุดผิดในตัวโปรแกรม ซึ่งจะต้องตรวจหาไปทีละบรรทัด เนื่องจากไม่สามารถเดาได้ว่า บริเวณที่ผิดพลาดนั้นอยู่จุดไหน

การเขียนโปรแกรมแบบไม่มีโครงสร้างนี้จะใช้ในโปรแกรมที่มีขนาดเล็กเท่านั้น ในโปรแกรมที่มีความซับซ้อนมากๆ เราจะใช้วิธีอื่น เพราะโปรแกรมที่ซับซ้อนมาก เราไม่สามารถตรวจสอบหาข้อผิดพลาดได้ง่าย เมื่อตรวจได้แล้ว อาจมีข้อผิดพลาดแบบเดียวกันเกิดขึ้นที่จุดอื่นของโปรแกรมอีก การแก้ไขจะจัดให้ชุดคำสั่งที่มีลักษณะเดียวกัน แต่มีข้อมูลต่างกันให้เป็นกลุ่มคำสั่งย่อยเดียวกัน การแก้ไขครั้งเดียว ก็จะแก้ไขข้อผิดพลาดได้ทั้งหมด

สำหรับข้อผิดพลาดที่ยังอาจเกิดได้อีกเราก็จะแก้ไขโดยการจัดรวมกลุ่มคำสั่งที่ทำหน้าที่เพื่อจุดประสงค์ใดจุดประสงค์หนึ่งเข้าด้วยกันเป็นกลุ่มคำสั่ง ในโปรแกรมที่เขียนโดยใช้รูปแบบนี้ ตัวโปรแกรมจะประกอบด้วยส่วนโปรแกรมหลัก (Main Program) และโปรแกรมย่อย (Subprogram) ซึ่งก็หมายถึงชุดคำสั่งหนึ่งชุด เมื่อมีการเรียกใช้โปรแกรมก็จะเริ่มทำงานในจุดเริ่มต้นของโปรแกรมหลัก ในโปรแกรมหลักก็จะมีการเรียกใช้โปรแกรมย่อย โดยการขานชื่อโปรแกรมย่อยเป็นคำสั่งหนึ่ง หรือเป็นฟังก์ชันทางคณิตศาสตร์ฟังก์ชันหนึ่ง หรือเป็นฟังก์ชันทั่วๆไป ในโปรแกรมย่อยก็อาจมีการเรียกใช้โปรแกรมย่อยลงไปอีกก็ได้ สำหรับการใช้ค่าตัวแปรนั้น ในโปรแกรมแต่ละส่วนก็จะมีชุดตัวแปรเป็นของตนเอง และจะมีชุดตัวแปรอีกชุดหนึ่งสำหรับใช้รวมกันทั้งหมด การทำเช่นนี้จึงทำให้สามารถแบ่งงานเขียนโปรแกรมออกเป็นส่วนๆ และให้บุคลากรหลายคนเขียนได้ ทำให้ลดเวลาในการพัฒนาโปรแกรมลงโปรแกรมที่ได้มีลักษณะโครงสร้างที่ง่ายแก่การเข้าใจมากขึ้นและเมื่อมีจุดผิดพลาดก็สามารถคาดเดาได้ว่า ส่วนผิดพลาดมาจากโปรแกรมย่อยใด และแก้ไขเฉพาะส่วนผิดพลาดนั้นได้ทันที และสามารถตรวจสอบเฉพาะส่วนโปรแกรมย่อยเพื่อหาจุดผิดพลาดได้อีกด้วย หากโปรแกรมใหม่ที่เราเขียนขึ้นมีการเรียกใช้งานการทำงาน ในลักษณะของโปรแกรมย่อยที่เคยเขียนขึ้นมาแล้ว ก็สามารถนำกลับมาใช้ใหม่ได้อีก ลดเวลาในการเขียนโปรแกรมได้

ปริญญานิพนธ์

เรื่อง โปรแกรมควบคุมเครื่องเจาะแผ่นวงจรพิมพ์อัตโนมัติ

SOFTWARE CONTROL AUTOMATIC PCB DRILLER

ผู้จัดทำ

1. นายชาญณรงค์ ปิ่นทอง
2. นายธีรพล หว่านณรงค์
3. นายสมบูรณ์ กิจวัฒนาบูลย์

อาจารย์ที่ปรึกษา

ลงนาม.....

(อาจารย์โกศล ตราชู)

ลงนาม.....

(อาจารย์สุรพงษ์ สิริพงศ์ดี)

หัวหน้าภาควิชาครุศาสตร์วิศวกรรม

ลงนาม.....

(ผศ.ดร.ธีระพล เทพหัสดิน ณ อยุธยา)

วัตถุมากมาย และวัตถุหนึ่งก็อาจจะสร้างมาจากวัตถุอื่นก็ได้ การเขียนโปรแกรมในรูปนี้จะช่วยให้เราสามารถแก้ไขไฟล์ต้นฉบับเพียงเล็กน้อย เพื่อนำมาใช้ในงานใหม่ที่คล้ายคลึงกันได้ และจากโครงสร้างการกำหนดตัวแปรและคำสั่งที่เป็นระบบอยู่แล้ว ทำให้หาจุดผิดพลาดได้ง่ายขึ้น แต่การเขียนโปรแกรมชนิดนี้มีข้อเสียตรงที่ขนาดของโปรแกรมที่ได้จากการแปลจะมีขนาดใหญ่กว่าวิธีอื่นและทำงานได้ช้ากว่า

2.2 ภาษาซีเบื้องต้น

2.2.1 โครงสร้างของโปรแกรมที่เขียนด้วยภาษาซี

ภาษาซีเป็นภาษาแบบโครงสร้างโมดูล เนื่องจากตัวโปรแกรมภาษาซีประกอบไปด้วยหลายๆโมดูล เรียงต่อเนื่องกันไป ในแต่ละโมดูลสามารถเรียกซึ่งกันและกันได้ และมีโมดูลชื่อ main สำหรับใช้เป็นทางเข้าออกโปรแกรม สามารถแบ่งโครงสร้างโปรแกรมได้เป็น 3 ส่วน คือ

1. ส่วนเรียกใช้โมดูลอื่น

เป็นส่วนที่บอกให้คอมไพเลอร์ไปดึงโมดูลอื่นๆ ที่กำหนดมาแปลร่วมด้วย โมดูลที่กำหนดขึ้นอาจเป็นโมดูลมาตรฐานที่มีในภาษาซี หรือเป็นโมดูลที่เขียนขึ้นมาใหม่ก็ได้ โมดูลเหล่านี้ต้องเขียนตามข้อกำหนดของภาษาซี แต่อาจไม่มีครบทั้ง 3 ส่วน ข้อแตกต่างระหว่างไฟล์โมดูลกับไฟล์ที่เป็นโปรแกรมก็คือ ในไฟล์ของโมดูลจะไม่มีโมดูล main

2. ส่วนกำหนดชื่อที่ใช้ในโปรแกรม

เป็นส่วนที่ใช้กำหนดค่าคงที่ ตัวแปร มาโคร และอื่นๆที่ต้องการ

3. ส่วนคำสั่ง

จะประกอบไปด้วยโมดูลต่างๆ เรียงกันไป โดยมีโมดูล main เป็นโมดูลที่ใช้เป็นทางเข้าออกของโปรแกรม

ทั้งสามส่วนนี้ อาจอยู่ปะปนกันไปก็ได้ โดยจะยึดหลักว่าส่วนไหนที่อยู่ก่อน จะถูกแปลก่อน และการเรียกหรืออ้างถึงชื่อใด จะต้องนิยามหรือกำหนดชื่อ หรือเรียกใช้โมดูลอื่นที่ได้กำหนดชื่อนั้นๆ ก่อนเสมอ

รูปแบบและองค์ประกอบของโปรแกรมภาษาซี

หมายเหตุ (Comments)

ปกติคอมไพเลอร์จะแปลทุกบรรทัดที่มีในโปรแกรม แต่ถ้าไม่ต้องการให้แปลส่วนไหนก็สามาถทำได้โดยการใส่เครื่องหมาย /* */ คร่อมกลุ่มคำสั่งดังกล่าว หรือใช้เครื่องหมาย // นำหน้าบรรทัดนั้นก็ได้ ข้อควรระวังคือ ห้ามใช้ comment ซ้อนกัน

Preprocessing Directives(#)

เป็นเครื่องหมายที่แจ้งให้คอมไพเลอร์ทำการใดๆ ก่อนที่จะมีการแปลโปรแกรมต่อไป คำสั่ง Preprocessing Directive จะนำหน้าด้วยเครื่องหมาย # มีคำสั่งดังนี้คือ

#define	#endif	#error
#ifndef	#elif	#if
#include	#else	#ifdef
#pragma	#undef	

ไดเรกทีฟ ที่น่าสนใจคือ

#include Directive ไดเรกทีฟนี้ทำหน้าที่แจ้งให้คอมไพเลอร์อ่านไฟล์อื่นเข้ามาแปลร่วมด้วย มีรูปแบบดังนี้ #include <filename> หรือ #include "filename"

การใช้เครื่องหมาย <> คร่อมชื่อ จะเป็นการแจ้งให้คอมไพเลอร์ค้นหาไฟล์จากไดเรกทอรีที่ได้กำหนดล่วงหน้าไว้ใน option directory แต่ถ้าใช้เครื่องหมาย " " จะเป็นการกำหนดให้ค้นหาจากไดเรกทอรีปัจจุบันหรือในไดเรกทอรีที่กำหนด

โดยปกติ ชื่อโมดูลของภาษาซี ตัวแปร ค่าคงที่ และมาโครพื้นฐาน จะรวมกันเป็นกลุ่มในไฟล์ที่เรียกว่า header file คือไฟล์ที่มีนามสกุล .h ใน header file แต่ละตัวจะเก็บโมดูลแยกกันเป็นส่วนการทำงานแต่ละส่วนสำหรับให้ผู้เขียนโปรแกรมเรียกใช้

#define Directive ไดเรกทีฟนี้ใช้กำหนดชื่อแทน คำ นิพจน์ คำสั่ง หรือ คำสั่งหลายคำสั่งได้ มีข้อกำหนดคือ #define ชื่อ ค่าที่ต้องการกำหนดแทน

หากต้องการเขียน "ค่าที่ต้องการกำหนดแทน" ในหลายบรรทัดสามารถใช้เครื่องหมาย \ ต่อท้ายได้

2.2.2 โครงสร้างทั่วไปในฟังก์ชัน

ส่วนคำสั่งของโปรแกรมภาษาซี ประกอบไปด้วยการนิยามโมดูล แต่ละโมดูลประกอบด้วยคำสั่งหลายคำสั่งเรียงกันเป็นกลุ่มและโมดูลแต่ละตัวจะมีชื่อกำกับเพื่อไว้ใช้เรียกทำงาน ในโปรแกรมภาษาซีจะมีกี่โมดูลก็ได้ แต่อย่างน้อยจะต้องมีหนึ่งโมดูลซึ่งมีชื่อว่า main() เป็นส่วนโปรแกรมหลักหรือโมดูลหลักเสมอ การเรียกชื่อโมดูลต่างๆ ในภาษาซีจะใช้คำว่า ฟังก์ชัน (function) ในการเขียนหรืออ้างถึงฟังก์ชันในภาษาซีจะเขียนลงท้ายด้วยเครื่องหมายวงเล็บ ( ) เสมอ ส่วน header files และกลุ่มฟังก์ชันที่ผู้เขียนโปรแกรมสร้างขึ้น จะใช้คำเรียกโดยรวมว่า โมดูล

กล่าวโดยสรุป ในส่วนคำสั่งของโปรแกรมภาษาซี จะประกอบไปด้วยการนิยามฟังก์ชันสำหรับใช้งานในโปรแกรมนั้นๆ จะมีฟังก์ชันอยู่ภายในส่วนนี้เป็นจำนวนเท่าใดก็ได้ แต่อย่างน้อยต้องมีหนึ่งฟังก์ชัน และฟังก์ชันนี้จะต้องมีชื่อว่า main()

ส่วนประกอบของฟังก์ชัน

1. ส่วนหัว(Heading)

เป็นส่วนที่ใช้นิยามชื่อฟังก์ชัน กำหนดชนิดและจำนวนตัวแปรที่ใช้ส่งค่าผ่านเข้าออก

2. ส่วนกลุ่มคำสั่ง (Compound Statements)

ส่วนนี้จะประกอบไปด้วยสามส่วนย่อยคือ

- ส่วนการกำหนดตัวแปรภายใน (Variable Declaration) ส่วนนี้ใช้สำหรับกำหนดตัวแปรสำหรับภายในฟังก์ชัน จากตัวอย่างนี้ไม่มีการกำหนดใดๆ

- ส่วนคำสั่ง (Statements) ประกอบด้วยคำสั่งต่างๆ เรียงกันไป คำสั่งในส่วนนี้มีได้หลายคำสั่ง หรือไม่มีเลยก็ได้ คำสั่งต้องลงท้ายด้วย ; (semicolon) เสมอ

- เครื่องหมาย { } ทำหน้าที่กำหนดขอบเขตของกลุ่มคำสั่ง โโยกลุ่มคำสั่งหนึ่งจะมีค่าเทียบเท่าคำสั่ง 1 คำสั่ง

2.2.3 การตั้งชื่อ (Identifiers)

ชื่อ (Identifiers) หรือในภาษาทั่วๆไปใช้คำว่า คำ (word) หมายถึงกลุ่มตัวอักษรที่ถูกนิยามขึ้น เพื่อใช้ภายในโปรแกรม ชื่อมีสองประเภทคือ

- คำสงวน (Keywords) คือคำที่ภาษา C กำหนดไว้ก่อนแล้วเป็นพื้นฐาน คำสงวนในภาษา C เช่น const , char , if , void เป็นต้น

- คำที่ผู้ใช้ตั้งชื่อใหม่ (User-defined word) คือ คำที่ผู้ใช้กำหนดขึ้นใหม่เพื่อใช้ในโปรแกรม ซึ่งต้องตรงตามหลักการตั้งชื่อ

2.2.4 สตรีม (Streams)

คือ สายข้อมูลที่ยาวต่อเนื่องกันติดต่อกันไป แหล่งกำเนิดข้อมูลและแหล่งรับข้อมูลเดียวกันอาจมีสตรีมที่ต่อถึงกันได้หลายสตรีม สตรีมบางตัวอาจจะติดต่อกับระบบภายนอกในรูปของทางเข้าหรือทางออก การไหลของสตรีม มีสองรูปแบบกว้างๆ ดังนี้

1. สตรีมที่แหล่งเก็บข้อมูลชั่วคราว สตรีมในภาษาซีส่วนใหญ่เป็นแบบนี้ เมื่อข้อมูลออกจากแหล่งกำเนิดแล้ว ข้อมูลในสตรีมจะถูกพักไว้ชั่วคราวในหน่วยความจำส่วนหนึ่งที่เรียกว่า บัฟเฟอร์ (Buffer) เมื่อแหล่งข้อมูลปลายทางต้องการอ่านข้อมูลก็จะมาอ่านจากส่วนนี้

2. สตรีมที่ไม่มีแหล่งเก็บข้อมูลชั่วคราวสตรีมในประเภทนี้ก็จะอาศัยความเร็วและความสามารถในการรับข้อมูลที่สูงกว่าจากแหล่งกำเนิด เมื่อมีข้อมูลไหลออกจากแหล่งกำเนิด ข้อมูลจะไปสู่แหล่งปลายทางทันทีโดยไม่มีการเก็บข้อมูลใดๆ ไว้ในสตรีม

2.2.5 ค่าคงที่ และแบบของตัวแปร

ข้อมูลในภาษาซี มีอยู่หลายชนิด แบ่งได้ 2 ชนิดใหญ่ๆ คือ

1. ข้อมูลชนิดจำนวนเต็ม คือ ตัวเลขไม่มีทศนิยม ชนิดข้อมูลที่นิยมใช้บ่อยๆ มีดังนี้

- ข้อมูลชนิดตัวอักษร (character) เป็นรหัสตัวเลขแทนตัวอักษร มีค่าระหว่าง –128 ถึง 127 ใช้พื้นที่ 1 ไบต์

- ข้อมูลชนิดจำนวนเต็ม (int) เป็นข้อมูลจำนวนเต็ม มีค่าระหว่าง -32768 ถึง 32767 ใช้พื้นที่ 2 ไบต์

- ข้อมูลชนิดจำนวนเต็มแบบเต็ม (long) ซึ่งเป็นข้อมูลจำนวนเต็ม ที่มีค่าระหว่าง –2147483648 ถึง 2147483647 ใช้พื้นที่ 4 ไบต์

2. ข้อมูลชนิดจำนวนจริง หรือข้อมูลชนิดมีทศนิยม มีดังนี้

- ข้อมูลชนิดจำนวนจริง (float) มีค่าระหว่าง 3.4E-38 ถึง 3.4E+38 ใช้พื้นที่เก็บ 4 ไบต์

- ข้อมูลชนิดจำนวนจริงแบบยาว (double) มีค่าระหว่าง 1.7E+308 ใช้พื้นที่ 8 ไบต์

ค่าคงที่ (constant)

ค่าคงที่คือค่าของข้อมูลที่มีจำนวนแน่นอน ค่าคงที่ในภาษาซี เขียนได้ 3 ลักษณะ คือ

- ค่าคงที่จำนวนเต็ม เขียนอยู่ในรูปตัวเลข อาจมีเครื่องหมายลบนำหน้า

- ค่าคงที่จำนวนจริง เขียนอยู่ในรูป a.bec หรือ a.bEc เช่น 3.2e5 , 7.0E-10

-ค่าคงที่หมายถึงรหัสแทนตัวอักษรคอมพิวเตอร์จะจดจำตัวอักษรในรูปเลขจำนวนเต็มโดยเขียนภายในเครื่องหมาย ‘ คร่อมตัวอักษร เช่น ‘A’ , “#’

ตัวแปร (Variable)

ตัวแปรหมายถึงชื่อเรียกแทนพื้นที่หน่วยความจำที่ใช้เก็บข้อมูล

ตัวแปรในภาษาซี มี 2 ประเภท คือ

1. ตัวแปรแบบ Global Variables อยู่นอกกลุ่มคำสั่ง กำหนดในจุดใดก็ได้ในนอกฟังก์ชัน แต่เพื่อความเหมาะสมจะเขียนรวมกันไว้ที่ต้นโปรแกรม

2. ตัวแปรแบบ Local Variables อยู่ในกลุ่มคำสั่งจะกำหนดก่อนเขียนคำสั่ง

ในการกำหนดค่าตัวแปร โปรแกรมจะจองหน่วยความจำสำหรับเก็บค่าไว้ แต่จะไม่มีการกำหนดค่าเริ่มต้นให้กับตัวแปรนั้น หากต้องการกำหนดค่าเริ่มต้น ในกรณีของ Global Variables เริ่มต้นทำคำสั่งใน Compound Statement ในกรณีของ Local Variables ให้ใส่เครื่องหมาย = ตามด้วยค่าคงที่หลังชื่อตัวแปรในบรรทัดของการกำหนดชื่อ

นิพจน์และคำสั่ง (Expression and Statements)

นิพจน์ (Expression) คือ การกระทำใดๆ เพื่อให้ได้ค่าของการกระทำออกมาหนึ่งค่า นิพจน์ประกอบไปด้วย ตัวถูกกระทำ (Operand) และตัวกระทำ (Operator) ซึ่งเขียนเรียงกันไป ตัวกระทำทางคณิตศาสตร์เบื้องต้นที่ควรรู้มีดังนี้ + , - , * , / , % เป็นต้น ลำดับในการคำนวณจะเรียงตามความสำคัญของ Operator ถ้าต้องการเรียงลำดับใหม่ จะใช้เครื่องหมายวงเล็บในการกำหนดลำดับการคำนวณก่อนหลัง

คำสั่ง (Statement) คือ ขั้นตอนในการทำงานหนึ่งขั้นตอน คำสั่งหนึ่งๆ ในภาษาซีจะต้องจบด้วยเครื่องหมาย ; (semicolon) เสมอ ยกเว้นกลุ่มคำสั่ง (Compound Statement)

คำสั่งว่าง (Null) คือ คำสั่งที่มีแต่เครื่องหมาย ; เพียงอย่างเดียว (ไม่มีชื่อใดๆ นำหน้า) คำสั่งว่างเป็นคำสั่งที่หมายถึงการไม่ทำอะไรเลย ถูกใช้เมื่อต้องการให้โปรแกรมกระทำในลักษณะไม่ทำอะไร เช่นการวนรอบ 10000 รอบโดยแต่ละรอบไม่ทำอะไร

กลุ่มคำสั่งว่าง (Null Compound Statements) คือกลุ่มคำสั่งที่ไม่มีการกำหนดค่าและคำสั่งใดๆ ภายในเลย

คำสั่งถ่ายค่า (Assignment Statements)

นิพจน์ถ่ายค่า (Assignment Expression) คือนิพจน์ที่มีตัวกระทำเป็นตัวกระทำถ่ายค่า (Assignment Operator) มีเครื่องหมายเป็น = นิพจน์ถ่ายค่าจะทำหน้าที่ถ่ายค่าจากตัวถูกกระทำทางขวา (Rvalue) ให้กับตัวแปรทางซ้าย (Lvalue) นิพจน์ถ่ายค่าเขียนได้ดังนี้

Lvalue = Rvalue

ค่าทางขวาอาจเป็น ค่าคงที่ ตัวแปร นิพจน์ ฟังก์ชัน หรือนิพจน์ถ่ายค่าก็ได้

คำสั่งถ่ายค่า (Assignment Statement) คือ นิพจน์ถ่ายค่าที่ลงท้ายด้วยเครื่องหมาย ;

ขั้นตอนในการเขียนโปรแกรมโดยอาศัยเทอร์โบซีและบอร์แลนด์ซีอย่างคร่าวๆ

โปรแกรมเทอร์โบซี (Turbo C) และบอร์แลนด์ซีพลัสพลัส (Borland C++) ประกอบไปด้วยตัวแปลภาษา (Complier) ตัวเชื่อม (Linker) ตัวช่วยตรวจสอบการทำงานของโปรแกรม (Debugger) และชุดพัฒนารวม (The Turbo C Integrated Development Environment เรียกย่อๆ ว่า IDE) ซึ่งภายใน IDE ประกอบด้วย เอดิเตอร์ ตัวแปลภาษา ตัวเชื่อม และตัวช่วยตรวจสอบการทำงาน ในการเขียนโปรแกรม เราจะใช้ IDE นี้เป็นตัวทำงานหลัก

ขั้นตอนในการทำงานอย่างคร่าวๆ มีดังนี้

1. เข้าโปรแกรม IDE โดยการเข้าไปในไดเรกทอรีที่มีโปรแกรม TC.EXE หรือ BC.EXE แล้วเรียกโปรแกรมดังกล่าว

2. เขียนโปรแกรมต้นฉบับ โดยการกดปุ่ม F10 และเลือกเมนู Edit เพื่อเข้าสู่เอดิเตอร์ของ IDE ในขณะนี้เราก็สามารถเขียนโปรแกรมภาษาซีได้ตามต้องการ

3. ทดลองแปลโปรแกรม โดยการกดปุ่ม F10 และเลือกเมนู Complie เลือกคำสั่ง Complie หรือใช้ปุ่ม Alt+F9 เพื่อแปลโปรแกรม แล้วตรวจดูข้อผิดพลาดที่เกิดขึ้นในขณะแปล การเปลี่ยนหน้าจอการทำงานระหว่างหน้าจอเอดิเตอร์กับหน้าจอที่ใช้รายงานข้อผิดพลาดใช้ปุ่ม F6

4. ทดลองสั่งให้ทำงาน เมื่อไม่มีข้อผิดพลาดใดๆ แล้ว เราจะสั่งให้โปรแกรมทำงานโดยการกดปุ่ม F10 แล้วเลือกเมนู Run เลือกคำสั่ง Run หรือใช้ปุ่ม Ctrl F9 เพื่อรันโปรแกรมแล้วตรวจดูการทำงานของโปรแกรม

2.2.6 โอเปอเรเตอร์ (Operator)

โอเปอเรเตอร์หรือตัวกระทำ ในภาษาซีมีอยู่หลายชนิดด้วยกัน โดยอาจแบ่งออกได้ดังนี้

1. ตัวกระทำถ่ายค่า (Assignment Operators)

2. ตัวกระทำบอกขนาด (Sizeof Operators)

3. ตัวกระทำทางคณิตศาสตร์(Arithmetic Operation)

4. ตัวกระทำทางตรรก (Logic Operations)

5. ตัวกระทำแบบบิตต่อบิต (Bitwise Operators)

6.ตัวกระทำลดรูปทางคณิตศาสตร์และตัวกระทำลดรูปทางบิตต่อบิต (Compound Assingment Operators)

7. ตัวกระทำเพิ่มค่าและตัวกระทำลดค่า (Increment and Decrement Operators)

8. ตัวกระทำสำหรับเลือกค่า (Condition Operators)

9. ตัวกระทำสำหรับจัดลำดับการให้ค่า (Comma Operators)

10.ตัวกระทำบอกตำแหน่งหน่วยความจำและตัวกระทำเข้าถึงข้อมูล (Address-of and Indirection Operators)

1.นิพจน์ และการแปลงค่าของตัวถูกกระทำ

นิพจน์ในภาษาซี มีรูปแบบดังนี้คือ

ตัวกระทำ ตัวถูกกระทำ

ตัวถูกกระทำ ตัวกระทำ ตัวถูกกระทำ

ตัวถูกกระทำ ตัวกระทำ ตัวถูกกระทำ ตัวกระทำ ตัวถูกกระทำ ....

ตัวถูกกระทำอาจเป็นได้ทั้ง ค่าคงที่ ตัวแปร นิพจน์ หรือฟังก์ชัน และ ตัวกระทำบางชนิดก็อาจจะกำหนดชนิดของตัวถูกกระทำ ผลของการกระทำของนิพจน์ จะให้ค่าออกมาหนึ่งค่าเสมอ ค่าที่ได้จะมีชนิดสอดคล้องกับตัวกระทำและตัวถูกกระทำภายในนิพจน์นั้นๆ หรืออาจจะกล่าวได้อย่างคร่าวๆ ว่า ชนิดของค่าที่ออกมาจากนิพจน์จะมีค่าชนิดเดียวกันกับชนิดของตัวถูกกระทำ เช่นถ้าตัวถูกกระทำมีชนิดเป็น int ค่าที่ได้จากนิพจน์ก็จะมีชนิดเป็น int ด้วย แต่สำหรับตัวกระทำบางตัวก็จะให้ค่าที่เปลี่ยนออกไป เช่นผลของการกระทำนิพจน์ทางตรรกะให้ค่าออกมามีชนิดเป็น int เสมอ ไม่ว่าตัวถูกกระทำมีชนิดเป็นชนิดใดก็ตาม

ในบางกรณี ค่าที่ได้จากนิพจน์อาจเป็นค่าที่มีชนิดคนละชนิดกับที่เราต้องการ เราอาจเปลี่ยนชนิดของค่านั้นๆ ให้มีชนิดตามที่เราต้องการได้โดยการเขียนชนิดของข้อมูลแบบใหม่ภายในวงเล็บ นำหน้านิพจน์นั้นๆ ดังรูปแบบต่อไปนี้

(แบบข้อมูลแบบใหม่) นิพจน์

(แบบข้อมูลแบบใหม่ที่ใช้ในการชี้หรือเข้าถึง *) นิพจน์ (ที่ให้ค่าการชี้)

ในกรณีของการเปลี่ยนค่าการชี้นี้สามารถกระทำได้กับ ค่าคงที่ตัวแปร และฟังก์ชันได้เช่นเดียวกับนิพจน์ทุกประการ

2. ตัวกระทำถ่ายค่า (Assingment Opertors)

ตัวกระทำถ่ายค่าในภาษาซีใช้เครื่องหมาย = ใช้ในการถ่ายค่าทางด้านขวาให้แก่ทางด้านซ้าย นิพจน์ที่ได้จากตัวกระทำถ่ายค่านี้ สามารถใส่เครื่องหมาย ; ลงท้ายนิพจน์ เพื่อเปลี่ยนนิพจน์ให้กลายเป็นคำสั่งได้ นอกจากนิพจน์ชนิดนี้แล้ว ยังมีนิพจน์เพิ่มค่า นิพจน์ลดค่า และนิพจน์ที่ใช้ Compound Assignment Operators

3. ตัวกระทำบอกขนาด (Sizeof Operators)

Sizeof Operators เป็นตัวกระทำใช้รายงานขนาดของหน่วยความจำที่จะต้องใช้เก็บค่า หรือรายงานขนาดของหน่วยความจำที่ตัวแปรหนึ่งๆ ต้องใช้ในการเก็บข้อมูล นิพจน์ที่ได้จาก sizeof มีรูปแบบดังนี้

Sizeof ค่าคงที่

Sizeof (แบบของตัวแปร)

ถึงแม้จะทราบอยู่แล้วว่า ค่าใดหรือตัวแปรใดต้องการพื้นที่เก็บเท่าใดก็ตาม การใช้ sizeof ก็ยังมีประโยชน์ โดยการใช้ sizeof จะทำให้โปรแกรมที่เขียนขึ้นนี้มีความเป็นมาตรฐานมากขึ้น เนื่องจากบนเครื่องคอมพิวเตอร์ต่างระบบกัน เช่น เครื่องมินิ เมนเฟรม เวอร์คสเตชัน หรืออื่นๆ จะมีการกำหนดพื้นที่ในการเก็บค่าตัวแปรบางชนิดแตกต่างกันออกไป การใช้ sizeof จะช่วยให้โปรแกรมที่

เขียนขึ้นสามารถนำไปแปลและทำงานบนเครื่องเหล่านั้นได้โดยไม่ต้องแก้ไข และนอกจากนั้น ยังช่วยอำนวยสะดวกในการหาขนาดของข้อมูลโครงสร้างที่ซับซ้อน

4.ตัวกระทำทางคณิตศาสตร์ (Arithmetic Operators)

ตัวกระทำทางคณิตศาสตร์ (Arithmetic Operators) คือ ตัวกระทำที่คำนวณทางด้านคณิตศาสตร์ ให้ค่าออกมามีชนิดสอดคล้องกับชนิดของตัวถูกกระทำ

นิพจน์ทางคณิตศาสตร์ (Arithmetic Expressions) คือ นิพจน์ที่มีตัวกระทำเป็นตัวกระทำทางคณิตศาสตร์

ในภาษาซีมีตัวกระทำทางคณิตศาสตร์อยู่ 5 ตัวด้วยกันคือ

ตารางที่2.1 ตัวกระทำทางคณิตศาสตร์

เครื่องหมาย	Name	ชื่อ	ใช้กับตัวแปร	ตัวอย่าง
*	Multiplication	การคูณ	ทุกชนิด	a * b
/	Division	การหาร	ทุกชนิด	a / b
%	Modulus	เศษการหาร	Int , long , char	a % b
+	Addition	การบวก	ทุกชนิด	a + b
-	Subtraction	การลบ	ทุกชนิด	a - b

ตัวกระทำ % เป็นตัวกระทำทางคณิตศาสตร์เพียงตัวเดียวที่กำหนดให้ใช้กับค่าจำนวนเต็มเท่านั้น ส่วนตัวอื่นๆ สามารถใช้ได้กับทุกค่า

5. ค่าทางตรรกและตัวกระทำทางตรรก (Logic Value and Logic Operators)

ค่าทางตรรก (Logic Value) คือ ค่าที่มีความหมายเป็นจริง หรือ เท็จ ในภาษาซีกำหนดให้ค่าทางตรรกจริง คือค่าจำนวนเต็มที่ไม่ใช่ศูนย์ โดยทั่วไปเราจะกำหนดให้ค่าจริงมีค่าเป็น 1

ตัวกระทำทางตรรก (Logic Operators) คือ นิพจน์ที่ให้ค่าผลของการกระทำค่าใดๆ เพื่อให้ได้ผลออกมาเป็นค่าทางตรรก

นิพจน์ทางตรรก (Logic Expressions) คือ นิพจน์ที่ให้ค่าผลของการกระทำในรูปของค่าทางตรรก

ในภาษาซีถือว่าค่าทางตรรกอันได้แก่ จริง และ เท็จ มีชนิดเป็น int ดังนั้น ผลของการกระทำทางตรรกจึงมีค่าเป็นจำนวนเต็ม และมีค่าได้เพียงสองค่าคือ 1 แทนจริง และ 0 แทนเท็จ ตัวกระทำทางลอจิกมีดังนี้คือ

ตารางที่ 2.2 ตัวกระทำทางตรรก

เครื่องหมาย	Name	ชื่อ	ใช้กับตัวแปร	ตัวอย่าง
>	Greater than	มากกว่า	ทุกชนิด	a > b
>=	Greater than or equal to	มากกว่าหรือเท่ากับ	ทุกชนิด	a >= b
<	Less than	น้อยกว่า	ทุกชนิด	a < b
<=	Less than or equal to	น้อยกว่าหรือเท่ากับ	ทุกชนิด	a <= b
==	Equal to	เท่ากับ	ทุกชนิด	a == b
!=	Not equal to	ไม่เท่ากับ	ทุกชนิด	a != b
&&	Logical AND	และ	จำนวนเต็ม	a && b
||	Logical OR	หรือ	จำนวนเต็ม	a || b
!	Not	นิเสธ	จำนวนเต็ม	!a

ข้อสังเกตของนิพจน์ทางตรรกก็คือ ผลของการกระทำของนิพจน์จะให้ค่าออกมาได้เพียงสองค่าเท่านั้นก็คือ 1 แทนค่าจริง และ 0 แทนค่าเท็จ ไม่ว่าตัวถูกกระทำจะเป็นค่าชนิดใด ผลลัพธ์จะเป็นชนิด int เสมอ

ตัวกระทำ && (และ) || (หรือ) และ ! (นิเสธ) มีจุดมุ่งหมายสำหรับใช้กระทำกับค่าทางตรรกโดยเฉพาะ แต่อาจนำตัวกระทำเหล่านี้มาใช้กับค่าจำนวนเต็มก็ได้ โดยจะถือว่า ค่าที่ไม่เป็นศูนย์คือ "จริง" และ ค่าศูนย์คือ "เท็จ"

ตัวกระทำทางตรรกที่เหลือจะใช้กับค่าชนิดใดก็ได้ แต่ตัวถูกกระทำทั้งสองของนิพจน์จะต้องเป็นค่าชนิดเดียวกัน

6. โครงสร้างการเก็บข้อมูลของตัวแปรจำนวนเต็ม

ภายในหน่วยความจำ เราจะมองข้อมูลอยู่ภายในรูปของตัวเลขฐานสอง โดยในแต่ละตำแหน่งหน่วยความจำจะเก็บได้ 1 ไบต์ หรือ 8 บิต เมื่อเราพิจารณาถึงตัวแปรจำนวนเต็มทั้งหกชนิดอันได้แก่ char , insigned char , int , unsign int , long , unsign long สำหรับตัวแปรชนิด unsigned char จะใช้พื้นที่เก็บข้อมูล 1 ไบต์ ตัวแปรชนิด unsigned int จะใช้พื้นที่เก็บ 2 ไบต์ และตัวแปรชนิด unsigned long จะใช้พื้นที่ 4 ไบต์ โดยจะจัดการเก็บหลักที่สูงกว่าไปไว้ในไบต์ถัดไป

ภายในแต่ละไบต์ของหน่วยความจำ ยังแบ่งออกเป็นบิต มีทั้งหมด 8 บิต โดยค่าในบิตทาง

ซ้ายจะเป็นหลักที่สูงกว่าทางขวา เรียกบิตหลักสูงสุดในไบต์สูงสุดของข้อมูลว่า Most Significant Bit (MSB) และเรียกบิตหลักต่ำสุดในไบต์ต่ำสุดของข้อมูลว่า Least Significant Bit (LSB)

สำหรับตัวแปรชนิด char int และ long นั้น มีการจัดวางข้อมูลในหน่วยความจำเช่นเดียวกันกับตัวแปรชนิด unsigned แต่มีความแตกต่างที่สำคัญคือ ในบิต MSB ของตัวแปรทั้งสามจะเก็บรหัสแทนเครื่องหมายบวกลบไว้ โดยกำหนดให้ค่า 0 แทนเครื่องหมายบวกค่า 1 แทนเครื่องหมายลบ และบิตที่เหลือจะเก็บค่าของข้อมูลไว้ ถ้าในบิต MSB เก็บค่า 1 บิตที่เหลือจะเก็บค่าข้อมูลอยู่ในรูป Two-complement (ซึ่งจะได้กล่าวในหัวข้อถัดไป) บิตที่เก็บรหัสแทนเครื่องหมายบวกลบนี้เรียกว่าบิตเครื่องหมาย (Signed bit) ดังเช่น 00000001
(เขียนในรูปเลขฐานสอง) จะแทนค่า 1 เนื่องจากบิตบนเป็น 0 (เป็นบวก) แต่ถ้าเป็น 11111110 จะแทนค่า –2 เนื่องจากบิตบนเป็น 1 (เป็นลบ)

ในการแปลงค่าข้อมูลจาก char ไปเป็น int หรือ long ซึ่งจะทำให้มีจำนวนบิตเก็บข้อมูลเพิ่มขึ้นนั้น ค่าในบิตใหม่ที่เพิ่มขึ้นมาจะได้จากค่าในบิตเครื่องหมาย ดังนั้น หากตัวแปร a ที่มีชนิดเป็น char เก็บค่า 10000010 (เขียนในรูปเลขฐานสอง) เมื่อนำตัวแปร b ที่มีชนิดเป็น int มารับค่าโดยมีการเปลี่ยนชนิดของค่าดังคำสั่ง

```
b=(int)a;
```

ค่าในตัวแปร b จะเป็น 1111111110000010 โดยค่า 1 ในบิตเครื่องหมายจะถูกนำมาใส่ในบิตใหม่ที่เพิ่มขึ้นดังกล่าว

7. ระบบเลขฐานสอง การกระทำพื้นฐานของระบบเลขฐานสอง และตัวกระทำแบบบิตต่อบิต (Bitwise Operators)

ในระบบเลขฐานสอง ค่าในแต่ละหลักจะมีค่าเพียงสองค่า คือ 1 และ 0 เท่านั้น เราสามารถเปลี่ยนค่าที่อยู่ในระบบเลขฐานสิบให้มาอยู่ในระบบเลขฐานสองได้โดยการใช้วิธีตั้งหารด้วยสอง

ในทำนองกลับกัน การเปลี่ยนค่าจากระบบเลขฐานสองกลับมายังระบบเลขฐานสิบกระทำได้โดยการใช้การคูณเลขกับค่าตัวเลขในแต่ละหลักของเลขฐานสอง โดยมีสูตรคือ

ค่าเลขฐานสิบ = 2^n * ค่าหลักที่ n + $2^{(n-1)}$ * ค่าหลักที่ (n-1) + … + 2^0 * ค่าหลักที่ 0

โดยให้หลักที่ 0 คือหลักต่ำสุดของค่าในระบบเลขฐานสอง

สำหรับการแปลงระหว่างเลขฐานสองกับเลขฐานสิบหกนั้นจะง่ายกว่ามาก เนื่องจากเมื่อแปลงค่า 1 หลักของเลขฐานสิบหก จะได้เลขฐานสอง 4 หลัก เราอาจเขียนตารางเทียบระหว่างเลขฐานสองและเลขฐานสิบหกได้ดังนี้

ตารางที่ 2.3 เทียบค่าเลขฐานสองกับเลขฐานสิบหก

เลขฐานสอง	เลขฐานสิบหก
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9
1010	A
1011	B
1100	C
1101	D
1110	E
1111	F

ในการแปลงจากเลขฐานสองให้เป็นเลขฐานสิบหกก็กระทำได้ในทางกลับกันคือ เราจะแบ่งเลขฐานสองออกเป็นกลุ่ม กลุ่มละสี่หลัก โดยเริ่มจากทางด้านขวาก่อน

การกระทำแบบบิตต่อบิต (Bitwise Operations)

การกระทำแบบบิตต่อบิต (Bitwise Operations) คือ การกระทำของเลขฐานสองในภาษาซี การกระทำจะเกิดเป็นคู่ๆ กับหลักแต่ละหลักภายในตัวถูกกระทำทั้งสอง

ตัวกระทำแบบบิตต่อบิต (Bitwise Operators) คือ ตัวกระทำที่ใช้ในการกระทำแบบบิตต่อบิต

การกระทำของเลขฐานสองนี้ ภาษาซีมีตัวกระทำที่ใช้อยู่หลายวิธีการด้วยกันดังนี้

- การบวก (Addition Operations)
- การกลับบิต (One-complement Operations)

- การหาค่าลบ (Two-complement Operations)
- การลบ (Subtraction Operations)
- การเลื่อนบิตข้อมูลไปทางซ้าย (Shift-left Operations)
- การเลื่อนบิตข้อมูลไปทางขวา (Shift-right Operations)
- การกระทำ AND (AND Operations)
- การกระทำ OR (OR Operations)
- การกระทำ Exclusive OR (XOR Operations)

การบวก (Addition Operations)

การบวกเลขฐานสอง มีลักษณะเช่นเดียวกันกับการบวกเลขฐานสิบ แต่มีค่าอยู่เพียงสองค่า คือ 0 กับ 1 ตารางแสดงการบวกของเลขฐานสองอาจเขียนได้ดังนี้

0	0	1	1
+	+	+	+
0	1	0	1
0	1	1	10 (ทด 1 ที่หลักต่อไป)

การกลับบิต (One-complement Operations)

การกลับบิต (One-complement Operations) เป็นการกลับค่าของเลขในแต่ละหลักโดยที่หลักใดมีค่าเป็น 0 ก็จะกลับเป็น 1 หลักใดมีค่าเป็น 1 ก็จะกลับเป็น 0

การหาค่าลบ (Two-complement Operations)

การหาค่าลบ (Two-complement Operations) คือการนำค่าเลขนั้นมากลับบิตและบวกด้วย 1

การลบ (Subtraction Operations)

การลบของเลขฐานสอง แตกต่างจากการหาค่าลบ การลบของเลขฐานสองจะมีตัวถูกกระทำสองตัว โดยการลบของเลขฐานสองก็คือการบวกค่าแรกด้วยค่าลบของตัวที่สองนั่นเอง เช่น 5-3 ก็คือ 5+(-3)

การเลื่อนบิตข้อมูลไปทางซ้าย (Shift-left Operations)

การเลื่อนบิตข้อมูลไปทางซ้าย เป็นการเลื่อนตัวเลขในหลักของเลขฐานสองไปทางซ้ายและเติมค่า 0 เข้าทางด้านขวา

การเลื่อนบิตข้อมูลไปทางขวา (Shift-right Operations)

การเลื่อนบิตข้อมูลไปทางขวาเป็นการกระทำในทางตรงข้ามการเลื่อนบิตข้อมูลไปทางซ้าย

การกระทำ AND (AND Operations)

การกระทำ AND ในลักษณะบิตต่อบิต จะเป็นการเทียบระหว่างหลักต่อหลัก โดยหากในหลักเดียวกันของตัวถูกกระทำทั้งสองมีค่าเป็น 1 ทั้งคู่ จะให้ค่าผลลัพธ์ในหลักนั้นเป็น 1

การกระทำ OR (OR Operations)

การกระทำ OR ในลักษณะบิตต่อบิต จะเป็นการเทียบระหว่างหลักต่อหลักเช่นเดียวกับการ AND แต่มีข้อแตกต่างในจุดที่หากในหลักเดียวกันของตัวถูกกระทำทั้งสองมีค่าเป็น 0 ทั้งคู่ จะให้ค่าผลลัพธ์ในหลักนั้นเป็น 0 ในกรณีอื่นจะให้ค่าเป็น 1

การกระทำ XOR (XOR Operations)

การกระทำ XOR ในลักษณะบิตต่อบิต จะให้ค่าเป็น 0 เมื่อในหลักเดียวกันของตัวถูกกระทำทั้งสองมีค่าเหมือนกัน และให้ค่าเป็น 1 เมื่อมีค่าต่างกัน

ตัวถูกกระทำแบบบิตต่อบิต จะต้องเป็นค่าจำนวนเต็มเสมอ และค่าที่ได้จากนิพจน์แบบบิตต่อบิตค่าออกมาเป็นจำนวนเต็มเช่นกัน ภาษาซีมีโอเปอร์เรเตอร์ใช้ในการกระทำแบบบิตต่อบิต ดังนี้

ตารางที่ 2.4 ข้อกำหนดของตัวถูกกระทำแบบบิตต่อบิต

เครื่องหมาย	Name	รูปทั่วไป	ตัวอย่าง	ผลลัพธ์
<<	Shift left	Aa << b	7 << 2	0x1c
>>	Shift right	Aa >> b	7 >> 2	0x01
&	AND	Aa & b	3 & 8	0
|	OR	Aa | b	3 | 8	0x0b
^	XOR	Aa ^ b	3 ^ 9	0x0a
~	NOT	~a	~0x08	0xf7

ตัวกระทำ ! (Logical NOT)มีความแตกต่างจาก ~ (Bitwise NOT) โดยที่ตัวกระทำ ! จะทำให้เรามองค่าที่จะมากระทำในลักษณะของตรรก ขณะที่การใช้ ~ จะทำให้เรามองค่าที่จะมากระทำในลักษณะของตรรก ขณะที่การใช้ ~ จะทำให้เรามองค่าที่มากระทำเป็นเลขจำนวนเต็ม เช่น !(1) จะมีค่าเป็น 0xfe ซึ่งก็คือการกลับบิตของเลขฐานสองนั่นเอง

8. การยุบนิพจน์ด้วย Compound Assignment Operators

ในคำสั่งถ่ายค่าที่มีตัวกระทำคณิตศาสตร์และบิตต่อบิตอยู่ในรูป

ตัวแปรที่ 1 = ตัวแปรที่ 1 operator ค่าที่ 2 (ค่าคงที่ ตัวแปร นิพจน์ หรืออื่นๆ)

จะสามารถลดรูปได้เป็น

ตัวแปรที่ 1 operator = ค่าที่ 2(ค่าคงที่ ตัวแปร นิพจน์ หรืออื่นๆ)

9. ตัวกระทำเพิ่มค่าและลดค่า (Increment and Decrement Operators)

ตัวกระทำเพิ่มค่าและลดค่า (Increment and Decrement Operators) ใช้สำหรับเพิ่มค่าหรือลดค่าตัวแปรชนิดจำนวนเต็ม เขียนอยู่ในรูป

++ ตัวแปร ตัวแปร++ - - ตัวแปร ตัวแปร- -

ข้อแตกต่างระหว่าง ++a กับ a++ และ - - a กับ a- - ก็คือในกรณีที่นิพจน์ชนิดนี้เขียนรวมกับนิพจน์อื่น หรืออยู่ในฟังก์ชันอื่น การเขียน ++a หรือ - -a จะเป็นการบังคับให้เพิ่มหรือลดค่าในตัวแปรก่อนที่จะทำนิพจน์หรือฟังก์ชันที่อยู่ร่วมในคำสั่งเดียวกัน แต่ถ้าเขียนเป็น a++ หรือ a- - ก็จะกระทำคำสั่งที่มีนิพจน์นี้อยู่ให้เสร็จสิ้นก่อน แล้วจึงค่อยเพิ่มหรือลดค่าในตัวแปรภายหลัง

10. ตัวกระทำสำหรับเลือกค่า (Conditional Operators)

ตัวกระทำสำหรับเลือกค่า (Conditional Operators) เป็นตัวกระทำใช้ในการเลือกการให้ค่าของนิพจน์ ตัวกระทำชนิดนี้ประกอบไปด้วยเครื่องหมาย ? และ : นิพจน์เลือกค่า (Conditios Expressions) จะเขียนอยู่ในรูป

นิพจน์1 , นิพจน์2 : นิพจน์3

นิพจน์เลือกค่ามีการให้ค่าดังนี้คือ หากค่าในนิพจน์1 มีค่าไม่เท่ากับ 0 (เป็นจริง) จะให้ค่านิพจน์เป็นไปตามนิพจน์ที่ 2 แต่ถ้าค่าในนิพจน์ 1 เป็น 0 จะให้ค่านิพจน์เป็นไปตามนิพจน์ที่ 3

11.ตัวกระทำจัดลำดับการให้ค่า (Comma Operators)

ตัวกระทำจัดลำดับการให้ค่า (Comma Operators) คือตัวกระทำที่ใช้เชื่อมนิพจน์หลายนิพจน์หรือคำสั่งหลายคำสั่ง ให้กลายเป็นคำสั่งเดียว การทำงานจะเกิดขึ้นเป็นลำดับจากนิพจน์ซ้ายไปขวา

ตัวกระทำนี้จะใช้เครื่องหมาย , คั่นระหว่างนิพจน์ย่อย มีรูปคือ

นิพจน์1 , นิพจน์2 , นิพจน์3 , ... , นิพจน์ n

12. ตัวกระทำบอกตำแหน่งหน่วยความจำ และตัวกระทำเข้าถึงข้อมูล (Address-of Operatprs and Indirection Operators)

เป็นตัวกระทำที่มีเครื่องหมาย & และ * อยู่หน้าตัวแปร

ตัวกระทำบอกตำแหน่งหน่วยความจำ (Address-of Operators - &) ใช้ในการรายงานค่าแอดเดรสเริ่มต้นของตัวแปรที่ใช้เก็บข้อมูล

ตัวกระทำเข้าถึงข้อมูล (Indirection Operators - *) ใช้ในการติดต่อกับข้อมูลในตำแหน่งหน่วยความจำที่เก็บไว้ในตัวชี้

2.2.7 คำสั่งในการตัดสินใจและวนรอบ

ในการเขียนโปรแกรมโดยทั่วไป มักจะมีส่วนที่เป็นการตัดสินใจเลือกกระทำคำสั่ง หรืออาจจะต้องมีการวนทำคำสั่งหลายๆ รอบ โดยแยกออกได้เป็นสองประเภทคือ

1. คำสั่งตัดสินใจ (Decision Statement)

คำสั่งเหล่านี้จะใช้ในการเลือกกระทำคำสั่งหรือยกเลิกการทำงาน โดยคำสั่งที่จะใช้ในการเลือกกระทำคำสั่งจะประกอบไปด้วย คำสงวนบอกชนิดของคำสั่ง ค่าที่ใช้ในการเลือกกระทำ และคำสั่งหรือกลุ่มคำสั่งที่ใช้ในการเลือกกระทำหรือไม่กระทำส่วนคำสั่งที่ใช้ในการยกเลิกการทำงานนั้น มักจะใช้ประกอบกับคำสั่งอื่นๆ เพื่อใช้ในการยกเลิกคำสั่งทำงานที่เหลืออยู่ คำสั่งตัดสินใจมีอยู่สองคำสั่ง ได้แก่คำสั่ง if และคำสั่ง switch-case คำสั่งใช้ในการยกเลิกการทำงานที่ควรรู้จักคือคำสั่ง break คำสั่ง continue และฟังก์ชัน exit()

2. คำสั่งวนรอบการทำงาน (Iteration Statement)

ใช้ในจุดที่ต้องการให้มีการทำคำสั่งชุดเดิมมากกว่าหนึ่งครั้ง คำสั่งวนรอบการทำงานนี้จะประกอบไปด้วย คำสงวนบอกชนิดของคำสั่ง ค่าที่ใช้ในการตรวจสอบการวนรอบ และคำสั่งหรือกลุ่มคำสั่งที่ต้องการให้กระทากว่าหนึ่งครั้ง คำสั่งวนรอบการทำงานมีอยู่สามคำสั่ง ได้แก่คำสั่ง while คำสั่ง do-while และคำสั่ง for

If Statements

คำสั่ง If (If Statements) คือ คำสั่งสำหรับใช้เบือกทำคำสั่งหรือกลุ่มคำสั่งโดยอาศัยการตัดสินใจจากค่าที่ใช้ตรวจสอบ มีรูปแบบดังนี้

If (value) statement1

If (value) statement1 else statement2

กฎ

1. ค่าที่ใช้ตรวจสอบ จะต้องมีเครื่องหมายวงเล็บคร่อมเสมอ ค่าดังกล่าวอาจใช้ ตัวแปร นิพจน์ หรือฟังก์ชัน ใส่ในจุดดังกล่าว

2. จะทำ statement 1 เมื่อค่าที่ได้จากการตรวจสอบมีค่าไม่เท่ากับ 0 (จริง) และทำ statement 2 เมื่อค่าที่ได้มีค่าเป็น 0 (เท็จ)

3. ค่าที่ได้จากการตรวจสอบจะต้องเป็นค่าจำนวนวเต็มเท่านั้น

4. ในกรณีจุดที่ statement 1 หรือ statement2 ต้องมีหลายคำสั่งจะใช้กลุ่มคำสั่งในจุดดังกล่าวแทน

5. ใช้คำ else โดดๆ โดยไม่มี if ไม่ได้

Switch-case Statement

คำสั่ง Switch (Switch Statements) คือคำสั่งที่ใช้เลือกตำแหน่งเริ่มต้นของคำสั่งที่จะทำงานภายในกลุ่มคำสั่งตามค่าที่ใช้ตรวจสอบ

จาก if statement เราจะเลือกกระทำคำสั่งได้อย่างมากสองทางเลือก ในกรณีที่ต้องการให้เลือกได้มากกว่านั้น เราจะใช้คำสั่ง switch แทน คำสั่ง switch มีรูปแบบดังนี้

switch *(value)*

{

case *value1 : statement;statement……..*

case *value2 : statement;statement……..*

……

case *valuen : statement;statement……..*

[default : *statement;statement……..*]

}

กฎ

1. ค่าที่ใช้ตรวจสอบจะต้องมีชนิดเป็นจำนวนเต็ม โดยอาจจะใช้ ตัวแปร นิพจน์ หรือฟังก์ชัน ในจุดดังกล่าวก็ได้

2. ตำแหน่งหลังเครื่องหมาย colon (:) หลังคำสงวน case คือ ตำแหน่งเริ่มต้นทำงานของคำสั่ง เมื่อค่าที่ได้จากการตรวจสอบมีค่าเท่ากับค่าคงที่ที่เขียนหลัง case

3. ในกรณีที่ไม่มีค่าใดที่อยู่หลัง case ตรงกับค่าที่ใช้ตรวจสอบโปรแกรมจะกระโดดไปทำตามคำสั่งที่อยู่หลังคำสงวน default หากไม่มี default case ก็จะไม่ทำคำสั่งใดๆ ในกลุ่มคำสั่งที่ตามหลังคำสงวน switch เลย

4. ในแต่ละ case สามารถมีคำสั่งได้มากกว่า 1 คำสั่ง หรือจะไม่มีเลยก็ได้ ในกรณีที่ไม่มีคำสั่ง โปรแกรมจะเริ่มกระทำในตำแหน่งเดียวกันของ case ถัดไป

Break Statements

คำสั่ง break (Break Statements) คือคำสั่งที่กำหนดให้หยุดการกระทำคำสั่งในกลุ่มคำสั่งของคำสั่ง switch case คำสั่ง while คำสั่ง do และคำสั่ง for มีรูปคำสั่งคือ

break;

คำสั่งนี้จะใช้ร่วมกับคำสั่ง switch คำสั่ง while คำสั่ง do-while และคำสั่ง for เพื่อใช้ในการกระโดดออกจากกลุ่มคำสั่งที่เป็นส่วนหนึ่งของคำสั่งดังกล่าว

While Statements

คำสั่ง while (While Statements) คือคำสั่งที่จะวนรอบทำงานจนกว่าค่าที่ใช้ตรวจสอบมีค่าเป็นเท็จ (มีค่าเป็น 0) มีรูปแบบทั่วไปดังนี้

while (value) statement

กฎ

1. ค่าที่ใช้ในการตรวจสอบต้องเป็นค่าจำนวนเต็ม ซึ่งอาจจะได้จาก ค่าคงที่ ตัวแปร นิพจน์ หรือ ฟังก์ชัน ก็ได้

2. จะมีการตรวจสอบก่อนที่จะทำ statement ในแต่ละรอบ

3. จะวนทำงานจนกว่าตรวจสอบได้ค่าเป็น 0 (ในรอบที่ตรวจสอบค่าที่ได้เป็น 0 จะไม่มีการทำงานตาม statement)

4. คำสั่งที่ใช้ทำงานจะต้องเป็นคำสั่งเดียวหรือเป็นกลุ่มคำสั่งเท่านั้น

Continue Statements

คำสั่ง continue (Continue Statements) คือคำสั่งที่ใช้ยกเลิกคำสั่งที่ยังเหลืออยู่ในกลุ่มคำสั่งที่อยู่ในส่วนหนึ่งของคำสั่งที่ใช้ในการวนรอบ อันได้แก่ คำสั่ง while คำสั่ง do-while และคำสั่ง for การยกเลิกคำสั่งที่อยู่ถัดไปภายในกลุ่มคำสั่งนี้จะทำให้เกิดการวนรอบใหม่ในทันที

คำสั่ง continue มีรูปแบบดังนี้

continue ;

Do-while Statements

คำสั่ง do-while (Do-while Statement or Do Statements) เป็นคำสั่งที่ใช้ในการสั่งวนรอบการทำงานจนกว่าค่าที่ใช้ตรวจสอบเป็นทเจ โดยจะทำงานตามคำสั่งหรือกลุ่มคำสั่งที่กำหนดก่อนตรวจสอบค่าในแต่ละ วนรอบ

คำสั่งนี้คล้ายคลึงกับคำสั่ง while แต่จะแตกต่างในการตรวจสอบค่าหลังจากการทำ statement ไปแล้ว มีรูปแบบทั่วไปดังนี้

do statement while (value) ;

กฎ

1. ค่าที่จะใช้ในการตรวจสอบต้องเป็นค่าจำนวนเต็ม โดยอาจจะเป็น ค่าคงที่ ตัวแปร นิพจน์ หรือ ฟังก์ชัน ก็ได้

2. ตรวจสอบหลังจากทำงานแล้ว

3. จะวนทำงานไปจนกว่าตรวจสอบค่าได้เป็น 0 (เป็นเท็จ)

4. คำสั่งที่ใช้ทำงานอาจเป็นคำสั่งเดียว หรือเป็นกลุ่มคำสั่งอย่างใดอย่างหนึ่ง

For Statements

คำสั่ง for (For Statements) คือคำสั่งที่ใช้ในการวนรอบเช่นเดียวกับ while แต่จะมีการกำหนดค่าเริ่มต้นภายในตัวคำสั่งไว้เลย มีรูปแบบทั่วไปดังนี้

for (statement1;value;statement2) statement3

กฎ

1. เมื่อโปรแกรมทำงานมาถึงคำสั่ง for ในครั้งแรก จะมีการทำงานตาม statement1 แล้วจึงตรวจสอบค่าว่าเป็นเท็จหรือไม่ ถ้าเป็นเท็จจะออกจากคำสั่ง for แต่ถ้าเป็นจริง ก็จะทำงานตาม statement3 แล้วจึงไปทำงานตาม statement2 แล้วกลับมาตรวจสอบ expression วนเวียนอย่างนี้เรื่อยไปจนกว่าตรวจสอบได้เป็นเท็จ การทำงานของคำสั่ง for อาจเขียนเป็น algorithm ได้ดังนี้

Algorithm for-statement

1. ทำ statement1
2. กระโดดไปข้อ 5
3. ทำ statement3
4. ทำ statement2
5. ถ้าค่าที่ใช้ตรวจสอบเป็นจริงให้กระโดดไปข้อ 3
6. return

2. statement1 เป็นคำสั่งถ่ายค่า ต้องมีคำสั่งเดียว และห้ามใช้กลุ่มคำสั่ง

3. statement2 เป็นคำสั่งที่ใช้เพิ่มค่าหรือลดค่าในตัวแปร หรือใช้ในการปรับค่าที่เกี่ยวเนื่องกับการวนรอบการทำงาน ต้องเป็นคำสั่งเดี่ยวเท่านั้น ห้ามใช้กลุ่มคำสั่ง ในกรณีที่ต้องการใช้หลายคำสั่งในจุดของ statement1 และ statement2 จะใช้ตัวกระทำจัดลำดับการให้ค่า (Comma Operations) ช่วยในจุดดังกล่าว

4. statement3 อาจเป็นคำสั่งเดี่ยวหรือเป็นกลุ่มคำสั่งก็ได้

5. ค่าที่ใช้ในการตรวจสอบต้องเป็นค่าชนิดจำนวนเต็ม ซึ่งอาจจะใช้ตัวแปร นิพจน์ หรือฟังก์ชัน ในจุดดังกล่าว

คำสั่งที่ใช้ออกจากโปรแกรม

โดยปกติ เมื่อจบฟังก์ชัน main( ) ก็จะถือว่าโปรแกรมทำงานเสร็จสิ้น โปรแกรมจะจบการทำงานและกลับสู่ระบบปฏิบัติการ แต่ถ้าผู้เขียนโปรแกรมต้องการให้ออกจากโปรแกรมในจุดอื่นจะกระทำได้โดยการใช้ฟังก์ชันใดฟังก์ชันหนึ่งต่อไปนี้

```
void abort(void);
void exit(int status);
```

ฟังก์ชัน abort( )เป็นฟังก์ชันที่ใช้ในการออกจากโปรแกรมอย่างไม่สมบูรณ์ เราจะใช้ฟังก์ชันนี้ในกรณีที่ต้องการสื่อความหมายให้ผู้ใช้โปรแกรมทราบว่า โปรแกรมดังกล่าวกลับสู่ดอสอย่างไม่สมบูรณ์ การใช้ฟังก์ชันนี้เราจะเขียนเป็นคำสั่งหนึ่งคำสั่ง

2.2.8 การสร้างฟังก์ชัน

รูปแบบของการกำหนดฟังก์ชันมีดังนี้

```
function type function_name (data type v)
{
        .
        statement ;
        .
}
```

การเรียกฟังก์ชันจะต้องระบุชื่อฟังก์ชันดังนี้

```
function_name (a) ;
```

เมื่อเรียกฟังก์ชันแล้วค่าของ v ซึ่งเป็นตัวแปรแท้ (actual variable) จะถูกส่งไปให้กับ a ซึ่งเป็นตัวแปรหุ่น (dummy variable) และเป็นชนิดเดียวกับตัวแปรแท้ ผลลัพธ์จากฟังก์ชันจะถูกส่งกลับไปยังโปรแกรมหลักเพียง 1 ค่า โดยเก็บไว้ใน function_name

การส่งค่ากลับไปยังโปรแกรมหลักจะต้องใช้คำสั่ง return นำหน้าตัวแปรหรือนิพจน์ที่ต้องการส่งค่ากลับ

การส่งค่าไปยังฟังก์ชันอาจจะส่งได้มากกว่า 1 ค่า แต่ฟังก์ชันจะส่งค่ากลับเพียงค่าเดียวเท่านั้น

การส่งค่าไปยังฟังก์ชันนอกจากจะส่งในรูปของตัวแปรแล้ว อาจจะส่งในรูปของค่าคงที่หรือนิพจน์ แต่โดยทั่วไปมักนิยมส่งในรูปของตัวแปรมากกว่า

ในเทอร์โบซีถ้าไม่ได้กำหนดชนิดของฟังก์ชันว่าเป็นข้อมูลชนิดใด จะถือว่าฟังก์ชันเป็นชนิดเลขจำนวนเต็ม เมื่อต้องการให้ฟังก์ชันส่งค่ากลับสู่โปรแกรมหลักเป็นข้อมูลชนิดอื่นๆ ซึ่งไม่ใช่ชนิดเลขจำนวนเต็ม จะต้องระบุให้เทอร์โบซีทราบโดยระบุไว้ก่อนฟังก์ชัน main( )

ตัวแปรโกลบอลและโลคอล

ตัวแปรแบบโกลบอลจะกำหนดนอกฟังก์ชันเสมอ จะสามารถนำไปใช้ได้ตลอดโปรแกรม หรือสามารถนำไปใช้ได้ในทุกฟังก์ชันที่มีอยู่ในโปรแกรม ตัวแปรแบบโกลบอลจะจองเนื้อที่ในหน่วยความจำตลอดเวลาในขณะที่โปรแกรมกำลังทำงาน

ตัวแปรแบบโลคอล ตัวแปรแบบนี้จะมีค่าเฉพาะภายในฟังก์ชันที่กำหนดตัวแปรนั้น ตัวแปรที่มีคุณสมบัติเป็นโลคอลถึงแม้จะมีชื่อเดียวกันทั้งในโปรแกรมหลักและนอกฟังก์ชัน ตัวแปรทั้งสองจะใช้หน่วยความจำแยกกัน โดยตัวแปรที่มีคุณสมบัติเป็นโลคอล จะใช้หน่วยความจำของเครื่องในส่วนของสแตก (stack) นอกจากนี้หน่วยความจำที่ตัวแปรแบบโลคอลใช้ จะถูกใช้เฉพาะเมื่อมีการเรียกฟังก์ชันเท่านั้น และเมื่อออกจากฟังก์ชันแล้วหน่วยความจำดังกล่าวจะถูกยกเลิก นั่นคือ ตัวแปรในโปรแกรมหลักซึ่งมีชื่อเดียวกันกับตัวแปรในฟังก์ชัน เมื่อตัวแปรตัวใดตัวหนึ่งมีค่าเปลี่ยนไปตัวแปรอีกตัวจะไม่เปลี่ยนตาม ตัวแปรดังกล่าวจะมีคุณสมบัติเป็นโลคอล

การส่งค่าแบบกำหนดค่าและแบบอ้างอิง

การส่งค่าระหว่างโปรแกรมหลักและฟังก์ชันแบ่งออกเป็น 2 แบบ คือ การส่งค่าแบบกำหนดค่า (pass by value) และการส่งค่าแบบอ้างอิง (pass by referance)

การส่งแบบกำหนดค่า เหมือนกับการส่งค่าทั่วๆ ไปในซี การส่งแบบกำหนดค่านั้นค่าของตัวแปรในโปรแกรมหลักที่ส่งไปในฟังก์ชันจะมีค่าไม่เปลี่ยนแปลง

การส่งค่าแบบอ้างอิง การส่งค่าแบบนี้ค่าของตัวแปรในโปรแกรมหลักสามารถจะถูกเปลี่ยนแปลงได้ในฟังก์ชันซึ่งต่างจากการส่งค่าแบบกำหนดค่า การส่งค่าแบบอ้างอิงนั้น ตัวแปรหุ่นที่กำหนดในฟังก์ชันจะเก็บแอดเดรสของตัวแปรหลักที่ส่งมาจากโปรแกรมหลัก ดังนั้นจึงสามารถเปลี่ยนแปลงค่าของตัวแปรในโปรแกรมหลักได้โดยใช้คุณสมบัติของพอยเตอร์

อาร์เรย์และฟังก์ชัน

เราไม่สามารถส่งทั้งอาร์เรย์ไปยังฟังก์ชันได้ จะส่งได้เฉพาะแอดเดรสของอาร์เรย์หรือพอยเตอร์เท่านั้น เมื่อเรียกฟังก์ชันโดยมีชื่ออาร์เรย์จะเป็นการส่งค่าพอยเตอร์ของตัวแปรอาร์เรย์ตัวแรกไปยังฟังก์ชัน นั่นคือในฟังก์ชันจะต้องมีพอยเตอร์สำหรับรับแอดเดรสของตัวแปรอาร์เรย์แต่ละตัว

ในการส่งอาร์เรย์แบบกำหนดค่านั้น เมื่อเปลี่ยนแปลงค่าของตัวแปรอาร์เรย์ที่ส่งมาจากโปรแกรมหลักแล้วจะต้องไม่เก็บในตัวแปรอาร์เรย์เดิม

คอมมานด์ไลน์อาร์กิวเมนต์

คอมมานด์ไลน์อาร์กิวเมนต์ (command-line arrument) คือข้อมูลซึ่งส่งไปยังโปรแกรมหลักโดยข้อมูลดังกล่าวจะต้องเขียนต่อจากชื่อโปรแกรมในขณะที่อยู่ในระบบจัดการ

คอมมานด์ไลน์อาร์กิวเมนต์อาจจะมีมากกว่า 1 อาร์กิวเมนต์ ซึ่งจะต้องเขียนแยกจากกันโดยการเว้นวรรคหรือ tab ในกรณีที่ต้องการให้ช่องว่างเป็นส่วนหนึ่งของข้อความ จะต้องเขียนข้อความภายในเครื่องหมาย " " เช่น

A>code ST138-R ST138-P ST138-T

มี 3 คอมมานด์ไลน์อาร์กิวเมนต์

A>massage " I love you."

มี 1 คอมมานด์ไลน์อาร์กิวเมนต์

ในเทอร์โบซี เราสามารถผ่านคอมมานด์ไลน์อาร์กิวเมนต์ไปยังฟังก์ชัน main() โดยผ่านทางอาร์กิวเมนต์ argc และ argv

argc จะเก็บจำนวนอาร์กิวเมนต์ซึ่งกำหนดในระบบจัดการ ดดยทั่วไปจะมีอย่างน้อย 1 เพราะชื่อของโปรแกรมจะเป็นอาร์กิวเมนต์ที่ 1 ค่าที่เก็บใน argc จะเป็นชนิดเลขจำนวนเต็มเสมอ ในตัวอย่างของ A>code ST138-R จะมีจำนวนอาร์กิวเมนต์ใน argc เป็น 2

argv เป็นพอยเตอร์ซึ่งชี้ไปที่อาร์เรย์สตริงก์ คอมมานด์ไลน์อาร์กิวเมนต์จะต้องเป็นสตริงก์เสมอ รูปแบบการกำหนด argv เป็นดังนี้

char *argv[ ] ;

การส่งพอยเตอร์จากฟังก์ชัน

เราสามารถส่งข้อมูลชนิดใดๆ จากฟังก์ชันกลับสู่โปรแกรมหลักหรือฟังก์ชัน main( ) ได้ ในเรื่องของอาร์เรย์ได้กล่าวถึงการผ่านพอยเตอร์หรือแอดเดรสไปยังฟังก์ชัน สำหรับการส่งพอยเตอร์กลับสู่โปรแกรมหลักสามารถใช้คำสั่งต่างๆ ได้คือ

การใช้คำสั่ง Call Stack และคำสั่ง Find Procedure

คำสั่งทั้งสองเป็นคำสั่งย่อยในคำสั่ง Debug ในเมนูหลัก คำสั่ง Call Stack จะแสดงชื่อของฟังก์ชันปัจจุบัน คำสั่งนี้จะมีผลเฉพาะขณะดีบักโปรแกรมในหน้าต่างเอดิต

สำหรับคำสั่ง Find Procedure จะใช้หาฟังก์ชันในโปรแกรม

2.2.9 การจัดการไฟล์ (File Manager)

ไฟล์หรือแฟ้มข้อมูล (Files) คือจุดที่ให้กำเนิดสตรีม หรือรับข้อมูลจากสตรีมเพื่อนำข้อมูลนั้นไปเก็บหรือใช้งานต่อไป

แบ่งไฟล์ออกเป็นสองประเภทคือ

1. ไฟล์ที่ทำหน้าที่รับข้อมูลเข้าหรือส่งข้อมูลออกจากระบบ

ไฟล์ประเภทนี้โดยส่วนใหญ่จะเป็นไฟล์ที่มีทิศทางเดียวกันคือ ส่งข้อมูลเข้าทางเดียว หรือส่งข้อมูลออกทางเดียวเท่านั้น ตัวอย่างเช่น คีย์บอร์ด เมาส์ เป็นไฟล์ชนิดส่งข้อมูลเข้าทางเดียว ส่วนจอภาพ เครื่องพิมพ์ เป็นไฟล์ชนิดส่งข้อมูลออกทางเดียว เป็นต้น

2. ไฟล์ที่อยู่ในหน่วยความจำสำรอง

ข้อมูล และโปรแกรมต่างๆที่อยู่ภายในระบบ อาจถูกเก็บไว้ในหน่วยความจำสำรอง เพื่อนำมาใช้งานในภายหลัง การเก็บข้อมูลและโปรแกรมที่อยู่ในหน่วยความจำสำรองนั้น จะถูกจัดอยู่เป็น

ส่วนๆ แต่ละส่วนก็คือกลุ่มข้อมูลหนึ่งกลุ่มหรือโปรแกรมซึ่งเราจะเรียกกลุ่มข้อมูลหรือโปรแกรมภายในหน่วยความจำสำรองว่าไฟล์เช่นเดียวกัน

โครงสร้างของไฟล์

เนื่องจากลักษณะของหน่วยความจำสำรองอินพุตและเอาต์พุตของระบบ มีความแตกต่างกันออกไปอย่างมาก เพื่อความสะดวกในการเขียนโปรแกรมเพื่อรับส่งข้อมูลเข้าออกกับหน่วยความจำสำรอง อินพุตและเอาต์พุตของระบบ จึงได้มีการกำหนดรูปแบบโครงสร้างข้อมูลมาตรฐานขึ้น โดยอาศัยการติดต่อกับโครงสร้างข้อมูลมาตฐานนี้ ผู้เขียนก็จะสามารถเขียนหรืออ่านกับฮาร์ดแวร์ที่มีลักษณะแตกต่างกันออกไปได้โดยง่าย

โครงสร้างข้อมูลนี้เอง ที่มีความหมายแท้จริงเป็นไฟล์ซึ่งจะประกอบไปด้วยข้อมูลหลักๆ สองกลุ่ม กลุ่มแรกจะมีลักษณะที่เหมือนกันสำหรับให้ผู้เขียนโปรแกรมเข้าใช้งาน ส่วนกลุ่มที่สองจะมีลักษณะแตกต่างกันออกไปตามชนิดของหน่วยความจำสำรองที่ใช้เก็บไฟล์เหล่านั้น หรืออินพุตเอาต์พุตที่ใช้จำลองเป็นไฟล์ กลุ่มข้อมูลในส่วนที่สองนี้ จะถูกจัดการโดยระบบปฏิบัติการ ดังนั้นเพื่อความสะดวกในการจัดการไฟล์ ระบบปฏิบัติการจะจัดข้อมูลของโครงสร้างไฟล์ไว้เป็นตารางเพื่อความสะดวกในการใช้งานของโปรแกรม การติดต่อกับไฟล์จึงง่ายขึ้น โดยอาศัยการอ้างถึงหมายเลขของตารางแทน หมายเลขที่ใช้แทนไฟล์นี้ เราเรียกว่า แฮนเดิล (Handles)

แฮนเดิล คือ หมายเลขที่ใช้แทนตารางสำหรับเก็บข้อมูลรายละเอียดของไฟล์

FILE คือ แบบข้อมูลชนิดไฟล์

ในระบบปฏิบัติการทั่วไป จะยอมให้เรากำหนดค่าจำนวนแฮนเดิลสูงสุดไว้ ดังเช่นในระบบปฏิบัติการดอส จะสามารถกำหนดได้จากไฟล์ CONFIG.SYS ในคำสั่ง FILES=n เป็นต้น

และเนื่องจากภาษาซี จะรับส่งข้อมูลกับไฟล์โดยอาศัยสตรีม โครงสร้างของสตรีมจึงมีชนิดเป็นไฟล์ด้วย และด้วยเหตุนี้ โครงสร้างของไฟล์จึงมีลักษณะคล้ายคลึงกับสตรีม นั่นคือ โครงสร้างของไฟล์ที่โปรแกรมมองเห็นจะประกอบไปด้วยหน่วยของข้อมูลที่เรียงกันไป โดยมีจุดสิ้นสุดที่แน่นอนที่หน่วยข้อมูลสุดท้ายของไฟล์

การเข้าถึงไฟล์ในภาษาซีสามารถกระทำได้ในสองวิธีคือ ใช้แฮนเดิลเป็นพารามิเตอร์ในฟังก์ชันเพื่อการเข้าหา หรือใช้สตรีมเป็นพารามิเตอร์ในฟังก์ชันในการเข้าหา แต่อันที่จริงแล้ว ไม่ว่าจะเข้าหาไฟล์ในรูปแบบใดทุกๆ ครั้งที่เปิดไฟล์ ก็จะมีการจองแฮนเดิลและมีการใช้สตรีมเพื่อเข้าหาไฟล์ทั้งสิ้น

การเขียนหรืออ่านหน่วยข้อมูลภายในไฟล์นั้น อาศัยตัวชี้ข้อมูลไฟล์ (File Pointers)

การจัดการไฟล์อาจแบ่งออกเป็นขั้นตอนได้ดังนี้คือ

1.เปิดไฟล์ ในการติดต่อกับไฟล์นั้น ในขั้นแรก ระบบจะยังไม่รับรู้ว่ามีไฟล์ใดๆ อยู่ภายในระบบ ยกเว้นไฟล์ที่เป็นแหล่งให้กำเนิดหรือรับข้อมูลจากสตรีมมาตรฐาน ซึ่งจะถูกเปิดโดยอัตโนมัติอยู่แล้ว การเปิดไฟล์กับหน่วยความจำสำรอง เป็นการกำหนดแฮนเดิลและหน่วยความจำส่วนหนึ่งซึ่งจะใช้สำหรับเป็นที่พักข้อมูลในขณะที่มีการเขียนหรืออ่านข้อมูล เนื่องจากการจัดการโครงสร้างหน่วยความจำสำรองโดยทั่วไป จะมีการเขียนและอ่านเป็นกลุ่มของข้อมูล เช่นในดิสก์ซึ่งจะแบ่งส่วนย่อยออกเป็นเซกเตอร์การเขียนและอ่านข้อมูลในดิสก์จะต้องกระทำคราวละหนึ่งเซ็กเตอร์เสมอในการจัดการไฟล์กับหน่วยความจำสำรองประเภทนี้จึงต้องมีหน่วยความจำในระบบส่วนหนึ่งที่กันไว้สำหรับใช้พักข้อมูลจากดิสก์ หน่วยความจำส่วนนี้เราจะเรียกว่า ไฟล์บัฟเฟอร์ (File Buffer) และโดยปกติ เมื่อเปิดไฟล์ ตัวชี้ข้อมูลไฟล์ก็จะถูกกำหนดให้ชี้ไปที่หน่วยข้อมูลแรกของไฟล์ด้วย

2. การจัดการข้อมูลไฟล์ เป็นการเข้าถึงข้อมูลต่างๆ ภายในโครงสร้างของไฟล์ รวมไปถึงการเขียนและอ่านข้อมูลจากหน่วยข้อมูลของไฟล์

- เขียนหรืออ่านหน่วยข้อมูลไฟล์ เป็นการใส่ข้อมูลลงในหน่วยข้อมูลของไฟล์หรืออ่านข้อมูลจากหน่วยข้อมูลของไฟล์

- เลื่อนตัวชี้ข้อมูลไฟล์ไปยังตำแหน่งที่ต้องการ โดยปกติเมื่อมีการเขียนหรืออ่านหน่วยข้อมูลภายในไฟล์ ตัวชี้ข้อมูลไฟล์จะเลื่อนไปชี้ยังหน่วยข้อมูลถัดไปโดยอัตโนมัติอยู่แล้ว แต่ในกรณีที่เราต้องการเข้าถึงตำแหน่งข้อมูลในจุดที่ไม่ใช่หน่วยถัดไปของข้อมูลที่เพิ่งอ่านหรือเขียนก็จำเป็นต้องเลื่อนตัวชี้ข้อมูลไฟล์เอง

- เขียนหรืออ่านข้อมูลโครงสร้างไฟล์ ดังที่ได้กล่าวมาแล้วว่า ข้อมูลในโครงสร้างไฟล์ประกอบไปด้วยสองส่วน คือส่วนที่มีลักษณะเหมือนกัน เพื่อความเป็นมาตรฐานในการติดต่อและส่วนที่สอง เป็นส่วนที่มีลักษณะเฉพาะขึ้นอยู่กับโครงสร้างของไฟล์ที่นอกเหนือไปจากการเขียนหรืออ่านหน่วยข้อมูลและการเลื่อนตัวชี้ข้อมูลไฟล์ ยกตัวอย่างเช่น การหาขนาดของไฟล์หรือการแก้ขนาดของไฟล์ การปรับความสามารถในการเข้าถึงไฟล์ เช่นเดียวกับคำสั่ง ATTRIB ในดอสเป็นต้น

- ปิดไฟล์ หลังจากที่ได้เขียนหรืออ่านข้อมูลเสร็จสิ้นแล้ว เราจะต้องปิดไฟล์เพื่อคืนแฮนเดิลที่จองมาให้แก่ระบบ สำหรับนำไปใช้เข้าถึงไฟล์อื่นๆ ต่อไป นอกจากนี้ ในกาณีของไฟล์ที่มีการพักข้อมูลในไฟล์บัฟเฟอร์การปิดไฟล์จะเป็นการนำข้อมูลที่ยังคงค้างอยู่ในไฟล์บัฟเฟอร์ให้ลงไปในไฟล์ทั้งหมด การออกจากโปรแกรมโดยไม่ได้ปิดไฟล์ จะทำให้ข้อมูลบางส่วนที่ยังคงค้างอยู่ในไฟล์บัฟเฟอร์ไม่ได้ถูกเขียนลงในไฟล์อย่างแท้จริง ซึ่งเป็นเหตุให้ข้อมูลบางส่วนในไฟล์อาจหายไปได้

การเปิดและปิดไฟล์

ฟังก์ชันที่ใช้ในการเปิดไฟล์ของภาษาซีคือ fopen( ) มีรูปแบบทั่วไปดังนี้

FILE *fopen(const char *filename,const char *mode)

ฟังก์ชัน fopen( ) ทำหน้าที่จองแฮนเดิลจากระบบปฏิบัติการและจองพื้นที่ไฟล์บัฟเฟอร์ตามชื่อ filename ที่กำหนด ซึ่งอาจจะเป็นค่าคงที่สตริงหรือเป็นตัวแปรสตริงก็ได้ เมื่อเปิดไฟล์ได้ตามต้องการ ก็จะมีการส่งค่าการชี้ข้อมูลไฟล์ หรืออาจเรียกว่าเป็นค่าสตรีมไปเชื่อมกับไฟล์ที่เปิด แต่ถ้าเปิดไฟล์ตามต้องการไม่ได้ ก็จะมีการส่งค่า NULL กลับ ฟังก์ชัน fopen( ) มีโหมดในการเปิดไฟล์ดังนี้

ตารางที่ 2.5 โหมดในการเปิดไฟล์

รหัสโหมด	ความหมาย
R	เปิดเพื่ออ่านไฟล์เก่า
W	เปิดเพื่อเขียนไฟล์ใหม่ หรือเขียนทับไฟล์เก่า
A	เปิดเพื่อเขียนต่อท้ายข้อมูลสุดท้ายของไฟล์เก่า
r+	เปิดเพื่ออ่านหรือเขียนทับไฟล์เก่า
w+	เปิดเพื่ออ่านหรือเขียนทับไฟล์เก่า หรือไฟล์ใหม่
a+	เปิดเพื่อเขียนต่อท้ายไฟล์เก่า หรือเขียนไฟล์ใหม่

นอกจากโหมดที่กล่าวมาแล้ว เรายังมีโหมดการเขียนอ่านอีกชุดหนึ่ง ซึ่งสามารถใช้ร่วมกันกับโหมดที่กล่าวมาแล้วได้ คือโหมด b หมายถึง ติดต่อกับไฟล์ที่มีหน่วยข้อมูลทั่วไป และ t หมายถึงการติดต่อกับไฟล์ที่มีหน่วยข้อมูลเป็น char เก็บข้อมูลอยู่ในรูปรหัสแอสกี ซึ่งก็คือ ไฟล์ตัวอักษร

ไฟล์ตัวอักษร (Text Files) คือไฟล์เก็บข้อมูลที่ประกอบขึ้นจากตัวอักษรซึ่งเขียนแทนด้วยรหัสแอสกี (ASCII) หน่วยข้อมูลแต่ละหน่วยจะมีชนิดเป็น char สตรม (Streams) (ตามความหมายที่เกี่ยวข้องกับไฟล์ในหน่วยความจำสำรอง) คือค่าการชี้ หรือค่าตำแหน่งหน่วยความจำเริ่มต้นของโครงสร้างข้อมูลไฟล์ซึ่งใช้เป็นที่พักข้อมูลภายในหน่วยความจำหลักของระบบ ระบบปฏิบัติการจะใช้พื้นที่เหล่านี้ในการติดต่อส่งข้อมูลกับไฟล์ในหน่วยความจำสำรองต่อไป

ไฟล์ตัวอักษรที่เรารู้จักกันดี ก็คือ ไฟล์ต้นฉบับ หรือ โปรแกรมต้นฉบับ ที่เราเขียนขึ้นจากเอดิเตอร์ทั่วๆ ไป

การกำหนดโหมด b และ t จะเขียนรวมอยู่กับโหมดการเขียนและอ่านเช่น rb wb+ เป็นต้น

ในกรณีที่ไม่สามารถให้ค่าการชี้ออกมาได้ เช่นไม่มีไฟล์ชื่อดังกล่าว ในกรณีของโหมด ra และ r+ หรือเกิดปัญหาอื่นๆ fopen( ) จะให้ค่าออกมาเป็น NULL เราจึงสามารถตรวจสอบไฟล์ว่ามีหรือไม่

หากเราต้องการเปลี่ยนโหมดของไฟล์ที่เรากำลังทำงานเสียใหม่อาจกระทำได้สองทางคือ ปิดไฟล์นั้นแล้วเปิดใหม่ในโหมดที่ต้องการ หรืออาจใช้ฟังก์ชัน freopen( ) ซึ่งมีโปรโตไทป์ดังนี้

```
FILE *freopen(const char *filename , const char *mode , FILE *stream);
```

หลังจากที่เราจัดการกับไฟล์ตามที่ต้องการเสร็จสิ้น เราก็จะต้องปิดไฟล์ เพื่อคืนไฟล์บัฟเฟอร์และแฮนเดิลให้แก่ระบบสำหรับใช้ในการติดต่อกับไฟล์อื่นๆ ต่อไป การปิดไฟล์นี้เราจะใช้ฟังก์ชัน fclose( ) มีโปรโตไทป์ดังรูป

```
int  fclose (FILE *stream);
```

การจัดการตัวชี้ข้อมูลไฟล์

เมื่อเราเปิดไฟล์ได้สำเร็จ ตัวชี้ข้อมูลไฟล์จะถูกกำหนดให้ชี้ไปยังหน่วยข้อมูลแรกของไฟล์ โดยปกติ ทุกครั้งหลังจากการเขียนหรืออ่านหน่วยข้อมูลแต่ละหน่วยในไฟล์ ตัวชี้ข้อมูลไฟล์จะเลื่อนไปยังหน่วยถัดไปโดยอัตโนมัติ แต่ถ้าผู้เขียนโปรแกรมต้องการเลื่อนตัวชี้ข้อมูลไฟล์ไปยังจุดอื่นที่ไม่ใช่ตำแหน่งหน่วยข้อมูลถัดไป ก็จะกระทำได้โดยการใช้ฟังก์ชัน fseek( ) ที่มีรูปแบบดังนี้

```
int  fseek(FILE *stream,long offset,int where)
```

เป็นการเลื่อนตัวชี้ข้อมูลไฟล์ไปยังตำแหน่งใหม่ ที่อยู่ห่างจากเดิม offset ไบต์ โดยทิศทางไปยังท้ายไฟล์จะเป็นบวก ส่วนทิศทางย้อนกลับมาทางต้นไฟล์จะมีค่าเป็นลบ ค่า where คือตำแหน่งอ้างอิงที่จะใช้ในการเลื่อนตัวชี้ข้อมูลไฟล์ ในภาษาซีได้มีการกำหนดค่าคงที่ใช้สำหรับบอกตำแหน่งอ้างอิงของข้อมูลไฟล์ไว้ดังนี้

ตารางที่ 2.6 ค่าคงที่ที่ใช้สำหรับบอกตำแหน่งอ้างอิงของข้อมูล

ชื่อค่าคงที่	ใช้แทนตำแหน่ง
SEEK_SET	ตำแหน่งเริ่มต้นไฟล์
SEEK_CUR	ตำแหน่งที่ตัวชี้ข้อมูลไฟล์ชี้ในปัจจุบัน
SEEK_END	ตำแหน่งท้ายไฟล์(ตำแหน่งสิ้นสุดไฟล์)

ที่ตำแหน่งท้ายสุดของไฟล์ในหน่วยความจำสำรองทุกไฟล์ จะเป็นรหัสบอกการจบของไฟล์ (End-of-file) ซึ่งเขียนโดยย่อในรูป EOF เราจะถือว่ารหัสตัวนี้จะเป็นหน่วยข้อมูลสุดท้ายของไฟล์ทุกไฟล์ การเขียนรูปแสดงเราจะใช้เครื่องหมาย ⊥ แทนรหัสบอกการจบของไฟล์เสมอ

เราจะใช้งานฟังก์ชัน fseek( ) ในรูปของคำสั่งดังตัวอย่าง

fseek (InStream , 0L ,SEEK_SET)

คือการเลื่อนตัวชี้ข้อมูลไฟล์ไปยังจุดเริ่มต้นไฟล์ โดยปกติเมื่อใช้คำสั่ง fopen( ) ตัวชี้ข้อมูลไฟล์ก็จะเลื่อนมายังตำแหน่งนี้โดยอัตโนมัติ

ค่าที่ใช้เป็นพารามิเตอร์บอกการเลื่อนตัวชี้ข้อมูลไฟล์มีหน่วยเป็นไบต์ ไม่ใช่จำนวนหน่วยข้อมูล ซึ่งอันที่จริง หน่วยข้อมูลไฟล์หนึ่งหน่วยอาจมีขนาดมากกว่าหนึ่งไบต์ก็ได้ ดังนั้นในกรณีที่หน่วยข้อมูลมีขนาดเกินหนึ่งไบต์ เราจะต้องคูณด้วยขนาดของข้อมูล เพื่อให้เป็นค่าจำนวนไบต์ที่ต้องการเลื่อนไป

ฟังก์ชันสำหรับจัดการตัวชี้ข้อมูลไฟล์ยังมีอีกตัวหนึ่งคือ rewind() ใช้เลื่อนตัวชี้ข้อมูลไฟล์ไปยังจุดเริ่มต้นของไฟล์ มีค่าเทียบเท่ากับการใช้ฟังก์ชัน fseek(stream ,0L ,SEEK_SET) มีรูปทั่วไปคือ

void rewind(FILE *stream)

การเขียนและอ่านไฟล์

ในการเขียนหรืออ่านไฟล์นั้น ตำแหน่งที่หน่วยข้อมูลจะเข้าไปหรือออกจากไฟล์นั้น จะถูกกำหนดโดยตัวชี้ข้อมูลไฟล์ หลังจากการเขียนหรืออ่านแต่ละครั้ง ตัวชี้ข้อมูลไฟล์จะเลื่อนไปยังตำแหน่งไบต์ถัดไปจากไบต์สุดท้ายที่เราได้เขียนและอ่านไปแล้ว

ฟังก์ชันมาตรฐานที่ใช้อ่านและเขียนไฟล์ มีโปรโตไทป์ดังนี้

size_t fread(void *ptr,size_t n,FILE *stream);

size_t fwrite(void *ptr,size_t n,FILE *stream);

ฟังก์ชัน fread( )ใช้สำหรับอ่านข้อมูลจากไฟล์ผ่านทางสตรีม stream มาเก็บไว้ในหน่วยความจำหลัก ซึ่งมีตำแหน่งหน่วยความจำเริ่มต้นตามค่าการชี้ ptr กำหนดหน่วยข้อมูลของไฟล์แต่ละหน่วยมีขนาดเป็น size และอ่านมาจำนวน n ข้อมูล

ฟังก์ชัน fwirte( ) กระทำในทิศทางตรงกันข้ามคือ เขียนข้อมูลลงไปในไฟล์ผ่านทางสตรีม stream จากหน่วยความจำหลักที่มีตำแหน่งหน่วยความจำเริ่มต้นที่ ptr เขียนลงไปจำนวน n หน่วยข้อมูล แต่ละหน่วยมีขนาด size ไบต์ ฟังก์ชัน fread( ) และ fwrite( ) เมื่อทำงานเสร็จสิ้นแล้วก็จะส่งค่ากลับเป็นค่าจำนวนหน่วยข้อมูลที่ได้เขียนหรืออ่านจริง (ไม่ใช่จำนวนไบต์)

ฟังก์ชันทั้งสองนี้ สามารถกำหนดขนาดของหน่วยข้อมูลได้ตามที่ต้องการ จึงรองรับไฟล์ได้ทุกประเภท

นอกจากฟังก์ชัน fread( )และ fwrite( ) แล้ว ยังมีฟังก์ชันอื่นๆ ที่ใช้ในการเขียนและอ่านไฟล์ โดยผ่านทางสตรีมที่มีทศทางไปยังไฟล์ที่ต้องการ เช่น

```
int     fgetc(FILE *stream)
int     fputc(int c,FILE *stream)
char  *fgets(char  *s,int c,FILE *stream);
int     fputs(const char *s,FILE *stream);
```

ฟังก์ชัน fgetc( ) ใช้อ่านข้อมูลจำนวนหนึ่งไบต์จากสตรีม โดยค่าที่ส่งกลับก็คือค่าข้อมูลที่อ่านได้ ฟังก์ชัน fputc( ) ใช้สำหรับเขียนข้อมูลลงในสตรีม โดยมีค่าที่ส่งเข้าไปเป็นค่าข้อมูลที่จะเขียนและค่าตัวแปรสตรีม

ฟังก์ชัน fgets( )ใช้อ่านข้อมูลสตริงจากสตรีม โดยจะหยุดอ่านเมื่อพบรหัสขึ้นบรรทัดใหม่ (0x0a) หรือจะหยุดอ่านเมื่ออ่านได้เกินกว่าค่า c-1 และฟังก์ชัน fputs( ) ใช้เขียนข้อมูลสตริงลงในสตรีม

ในกรณีที่เป็นไฟล์ตัวอักษร แาจกระทำได้สองวิธีคือ กำหนดให้หน่วยข้อมูลมีชนิดเป็นตัวอักษร และเขียนหรืออ่านทีละหนึ่งตัวอักษร หรืออาจจะใช้ฟังก์ชัน fgets( ) หรือ fputs( ) ในการเขียนหรืออ่านเป็นบรรทัดก็ได้

ฟังก์ชันเกี่ยวกับการจัดการไฟล์ที่น่าสนใจ

```
int access(const char *filename,Int amode)
```

ความหมาย ตรวจสอบการเข้าถึงไฟล์

filename	ชื่อโปรแกรมและไดเรกทอรีที่กำหนด	
amode	=06	ตรวจว่าอนุญาตให้เขียนหรืออ่านได้หรือไม่
	=04	ตรวจว่าอนุญาตให้อ่านได้หรือไม่
	=02	ตรวจว่าอนุญาตให้เขียนได้หรือไม่
	=01	ตรวจว่าอนุญาตสั่งให้ทำงานได้หรือไม่
	=00	ตรวจว่ามีไฟล์หรือไม่
หมายเหตุ	ในระบบปฏิบัติการดอส amode 04 = 00 และ 06 = 02	
ค่าที่ส่งกลับ	=0	สำเร็จ
	= -1	ไม่สามารถหาไฟล์ได้

errno =ENOENT หาไฟล์หรือไดเรกทอรีไม่พบ

=EACCES ไม่สามารถจัดการกับไฟล์นั้นได้

ตัวแปร errno เป็นตัวแปร Global ซึ่งฟังก์ชันเกี่ยวกับไฟล์ในภาษาซีจะนำค่าไปเก็บไว้เมื่อทำงานตามฟังก์ชันเสร็จสิ้น เพื่อความสะดวก ภาษาซีจึงได้มีการกำหนดชื่อใช้เรียกแทนรหัสของค่าในตัวแปร errno ไว้

<dir.h> int chdir(const char *path);

ความหมาย เปลี่ยนไดเรกทอรีที่อยู่ปัจจุบัน

path ไดเรกทอรีใหม่ที่ต้องการเปลี่ยน (พารามิเตอร์เดียวกับคำสั่ง CD ในดอส)

ค่าที่ส่งกลับ = 0 สำเร็จ

= -1 ไม่สำเร็จ

errno = ENOENT หาไดเรกทอรีไม่พบ

<dos.h> INT_CHMOD(CONST CHAR *PATH , INT FUNC[, Int attrib]);

ความหมาย กำหนดค่าความสามารถในการใช้ไฟล์หรืออ่านค่าในการใช้ไฟล์ใช้กับ DOS

path ชื่อโปรแกรมและไดเรกทอรีที่ต้องการ

func = 0 อ่านค่าความสามารถในการใช้ไฟล์

= -1 กำหนดค่าความสามารถในการใช้ไฟล์

attrib ใช้ในกรณีที่ func = 1

=FA_RDONLY กำหนดให้อ่านได้เท่านั้น

=FA_HIDDEN Hidden file

=FA_SYSTEM System file

=FA_LABEL Volume Label

=FA_DIREC Directory

=FA_ARCH Archive

ค่าที่ส่งกลับ ค่า attrib เมื่อสำเร็จ

= -1 เมื่อไม่สำเร็จ

errno = ENOENT หาไฟล์หรือไดเรกทอรีไม่พบ

= EACCES ไม่สามารถจัดการกับไฟล์นั้นได้

ฟังก์ชัน int chmod(const char *path , Int amode);

<io.h> ความหมาย กำหนดค่าความสามารถในการใช้ไฟล์ ใช้กับ DOS หรือ UNIX

ค่าคงที่ path ชื่อโปรแกรมและไดเรกทอรีที่ต้องการ

<sys\stat.h> amode =S_IWRITE กำหนดให้เขียนได้เท่านั้น

=S_IREAD กำหนดให้อ่านได้เท่านั้น

=S_IREAD | S_IWRITE กำหนดให้อ่านหรือเขียนได้

ค่าที่ส่งกลับ = 0 เมื่อสำเร็จ

= -1 เมื่อไม่สำเร็จ

errno = ENOENT หาไฟล์หรือไดเรกทอรีไม่พบ

= EACCES ไม่สามารถจัดการกับไฟล์นั้นได้

int chsize(int handle , long size);

ความหมาย เปลี่ยนตามขนาดของไฟล์ ใช้กับระบบปฏิบัติการ DOS

handle ไฟล์แฮนเดิล

size ขนาดใหม่ที่ต้องการ

ค่าที่ส่งกลับ = 0 เมื่อสำเร็จ

= -1 เมื่อไม่สำเร็จ

errno = EACCES ไม่สามารถจัดการกับไฟล์นั้นได้

= EACCES แฮนเดิลไม่ถูกต้อง

void clearerr(FILE *stream);

ความหมาย ลบค่ารายงานความผิดพลาดของไฟล์นั้นทิ้ง

stream สตรีมของไฟล์ที่ต้องการ

int _close(Int handle); ใช้สำหรับ DOS

int close(int handle); ใช้สำหรับ DOS หรือ UNIX

ความหมาย ปิดไฟล์

handle แฮนเดิลของไฟล์ที่ต้องการปิด

หมายเหตุ ในการปิดไฟล์ จะไม่ได้ใส่รหัสบอกการจบของไฟล์ซึ่งมีค่าเท่ากับ 0x1a เป็นหน้าที่ของผู้เขียนโปรแกรมที่จะต้องใส่รหัสดังกล่าวก่อนปิดไฟล์ด้วยตนเอง

ค่าที่ส่งกลับ = 0 สำเร็จ

= -1 ไม่สามารถปิดได้

	errno	= EBADF	แฮนเดิลไม่ถูกต้อง
ฟังก์ชัน	int _creat(const char *path , Int attrib);		ใช้สำหรับ DOS
<io.h>	int creat(const char *path , Int amode);		ฟังก์ชันมาตรฐาน
ค่าคงที่	ความหมาย	เปิดไฟล์ใหม่ หรือเขียนทับไฟล์เก่า	
<dos.h>	path	ไดเรกทอรีและไฟล์ที่กำหนด	
สำหรับ _creat( )	attrib	ในกรณีของฟังก์ชัน _creat( )	
<sts\stat.h>		=FA_RDONLY	กำหนดให้ไฟล์ที่เขียนเป็นชนิดอ่านได้อย่างเดียว
		=FA_HIDDEN	กำหนดให้ไฟล์ที่เขียนเป็นชนิดHidden
		=FA_SYSTEM	กำหนดให้ไฟล์ที่เขียนเป็นชนิด System
		ในกรณีของฟังก์ชัน creat( )	
	amode	=S_IWRITE	กำหนดให้เขียนได้เท่านั้น
		=S_IREAD	กำหนดให้อ่านได้เท่านั้น
		=S_IREAD | S_IWRITE	กำหนดให้อ่านหรือเขียนได้
	ค่าที่ส่งกลับ	มากกว่าหรือเท่ากับ 0	ค่าแฮนเดิล
		น้อยกว่า 0	ไม่สามารถเปิดให้ตามต้องการได้
	errno	=ENOENT	หาไดเรกทอรีหรือไฟล์ที่ต้องการไม่ได้
		=EMFILE	แฮนเดิลไม่พอใช้ (แฮนเดิลหมดแล้ว)
		=EACCES	ไม่สามารถจัดการกับไฟล์นั้นได้
ฟังก์ชัน	int creatnew(const char *path , Int attrib);		ใช้สำหรับ DOS
<io.h>	ความหมาย	เปิดไฟล์ใหม่	
ค่าคงที่	path	ชื่อไดเรกทอรีและไฟล์ที่กำหนด	
<dos.h>	attrib	=FA_RDONLY	กำหนดให้ไฟล์ที่เขียนเป็นชนิดอ่านได้อย่างเดียว
		=FA_HIDDEN	กำหนดให้ไฟล์ที่เขียนเป็นชนิด Hidden
		=FA_SYSTEM	กำหนดให้ไฟล์ที่เขียนเป็นชนิด System
	ค่าที่ส่งกลับ	มากกว่าหรือเท่ากับ 0	ค่าแฮนเดิล
		= -1	ไม่สามารถเปิดให้ตามต้องการได้
	errno	= ENOENT	หาไดเรกทอรีหรือไฟล์ที่ต้องการไม่ได้
		= EMFILE	แฮนเดิลไม่พอใช้ (แฮนเดิลหมดแล้ว)

= EEXIST มีไฟล์นั้นแล้ว

int eof(Int handle); ใช้สำหรับ DOS

ความหมาย ตรวจสอบว่าตัวชี้ข้อมูลไฟล์ชี้ที่จุดสิ้นสุดไฟล์หรือไม่

handle แฮนเดิลของไฟล์ที่ต้องการตรวจสอบ

ค่าที่ส่งกลับ = 0 ปิดไฟล์และสตรีมได้สมบูรณ์

= 1 ถึงจุดสิ้นสุดของไฟล์แล้ว

errno = EBADF แฮนเดิลผิดพลาด

int fcolse(FILE *stream);

ความหมาย ปิดสตรีมและไฟล์ที่กำหนด

stream สตรีมที่ต้องการปิด

ค่าที่ส่งกลับ = 0 ปิดไฟล์และสตรีมได้สมบูรณ์

= EOF มีความผิดพลาดในการปิดสตรีม

int fcloseeall(void);

ความหมาย ปิดสตรีมทุกตัว ยกเว้นสตรีมมาตรฐาน

ค่าที่ส่งกลับ มากกว่า 0 จำนวนสตรีมที่ปิด

= EOF เกิดความผิดพลาดในการปิดสตรีม

FILE *fopen(Int handle , char *type);

ความหมาย เปิดสตรีมเพื่อใช้กับไฟล์ที่มีแฮนเดิลตามที่กำหนด

handle ค่าแฮนเดิลของไฟล์ที่เปิดอยู่แล้ว

type = r เปิดสตรีมเพื่ออ่าน

= w เปิดสตรีมเพื่อเขียน

= a เปิดสรีมเพื่อเขียนต่อท้าย หรือเปิดไฟล์ใหม่หากไฟล์ว่าง

= r+ เปิดสรีมจากไฟล์เก่าเพื่ออ่านหรือเขียน

= w+ เปิดไฟล์ใหม่หรือไฟล์เก่าเพื่อแก้ไข

= a+ เปิดไฟล์เพื่อเขียนต่อท้าย

ความหมาย เปิดไฟล์ร่วมกันกับไฟล์เดิม (เปิดแฮนเดิลใหม่กับไฟล์เดิมที่เปิดอยู่ก่อนแล้ว)

ค่าที่ส่งกลับ ค่าสตรีม

<stdio.h> int feof(FILE *stream);

ความหมาย ตรวจสอบว่าตัวชี้ข้อมูลไฟล์ชี้ที่จุดสิ้นสุดไฟล์หรือไม่

stream สตรีมของไฟล์ที่ต้องการตรวจสอบ

ค่าที่ส่งกลับ =0 ยังไม่สิ้นสุด

=ค่าอื่น ถึงจุดสิ้นสุดของไฟล์แล้ว

<stdio.h> int ferror(FILE *stream);

ความหมาย ตรวจสอบความผิดพลาดที่เกิดขึ้นในสตรีม

stream สตรีมของไฟล์ที่ต้องการตรวจสอบ

ค่าที่ส่งกลับ =0 ไม่มีความผิดพลาด

=ค่าอื่น มีความผิดพลาดเกิดขึ้น

หมายเหตุ ค่าความผิดพลาดจะถูกลบทิ้งเมื่อใช้ฟังก์ชัน rewind( ) หรือ clearerr( )

<stdio.h> int fflush(FILE *stream);

ความหมาย ดันข้อมูลที่ยังคงค้างอยู่ในสตรีมให้ลงไฟล์ให้หมด

stream สตรีมของไฟล์ที่ต้องการทำงาน

ค่าที่ส่งกลับ = 0 สำเร็จ

= EOF มีความผิดพลาดเกิดขึ้น

<io.h> int filelength (Int handle); ใช้สำหรับ DOS

ความหมาย หาความยาวของไฟล์

handle แฮนเดิลของไฟล์ที่ต้องการหาความยาว

ค่าที่ส่งกลับ ค่าบวก ค่าความยาวของไฟล์ มีหน่วยเป็นไบต์

= -1 ไม่สามารถหาความยาวได้

errno = EBADF แฮนเดิลผิดพลาด

<stdio.h> int fileno(FILE *stream);

ความหมาย หาค่าแฮนเดิลของไฟล์ที่สตรีมชี้อยู่

stream สตรีมของไฟล์ที่ต้องการหาแฮนเดิล

ค่าที่ส่งกลับ ค่าแฮนเดิลของไฟล์ที่สตรีมชี้อยู่

<stdio.h> int flushall(void);

ความหมาย ดันข้อมูลที่ยังค้างอยู่ในสตรีมทุกตัวให้ลงไฟล์ให้หมด และล้างข้อมูลในสตรีมที่ใช้สำหรับอ่านจากไฟล์ทุกตัว

ค่าที่ส่งกลับ มากกว่า 0 จำนวนสตรีมที่จัดการ

ค่าอื่น เกิดความผิดพลาด

<stdio.h> FILE *fopen(const char *filename , const char *mode);

ความหมาย เปิดไฟล์และเชื่อมสตรีม

filename ชื่อไดเรกทอรีและชื่อไฟล์ที่กำหนด

mode = r เปิดสตรีมเพื่ออ่าน

= w เปิดสตรีมเพื่อเขียน

= a เปิดสตรีมเพื่อเขียนต่อท้าย หรือเปิดไฟล์ใหม่หากไฟล์ว่าง

= r+ เปิดสตรีมจากไฟล์เก่าเพื่ออ่านหรือเขียน

= w+ เปิดไฟล์ใหม่หรือไฟล์เก่าเพื่อแก้ไข

= a+ เปิดไฟล์เพื่อเขียนต่อท้าย

_fmode (ตัวแปร global)

= O_BINARY ไฟล์ชนิดไบนารี

= O_TEXT ไฟล์ชนิดตัวอักษร

ค่าที่ส่งกลับ NULL ไม่สามารถเปิดไฟล์ได้ตามต้องการ

ค่าอื่นๆ ค่าแอดเดรสของสตรีมที่ต้องการ

<stdio.h> size_t fread(void *ptr , size_t size , size_t n , FILE *stream);

ความหมาย อ่านข้อมูลจากสตรีม

ptr ตัวชี้ที่ชี้ไปยังจุดเริ่มต้นของตำแหน่งหน่วยความจำที่ต้องการใช้เก็บค่าจากสตรีม

size ขนาดของหน่วยข้อมูลที่ต้องการอ่าน

n จำนวนของหน่วยข้อมูลที่ต้องการอ่าน

stream สตรีมที่ต้องการอ่าน

ค่าที่ส่งกลับ จำนวนข้อมูลที่อ่าน มีหน่วยเป็นจำนวนหน่วยข้อมูล

<stdio.h> FILE *fopen(const char *filename , const char *mode , FILE *stream);

ความหมาย เปิดไฟล์ใหม่อีกครั้งหนึ่ง

filename ชื่อไดเรกทอรีและชื่อไฟล์ที่กำหนด

mode เช่นเดียวกับ mode ของ fopen( )

stream สตรีมเก่าที่จะปิดลงไป

_fmode (ตัวแปร global)

= O_BINARY ไฟล์ชนิดไบนารี

= O_TEXT ไฟล์ชนิดตัวอักษร

ค่าที่ส่งกลับ NULL ไม่สามารถเปิดไฟล์ได้ตามต้องการ

ค่าอื่นๆ ค่าแอดเดรสของสตรีมใหม่ที่ต้องการ

<stdio.h> size_t fread(void *ptr , size_t size , size_t n , FILE *stream);

ความหมาย อ่านข้อมูลจากสตรีม

ptr ตัวชี้ที่ชี้ไปยังจุดเริ่มต้นของตำแหน่งหน่วยความจำที่ต้องการใช้เก็บค่าจากสตรีม

size ขนาดของหน่วยข้อมูลที่ต้องการอ่าน

n จำนวนของหน่วยข้อมูลที่ต้องการอ่าน

stream สตรีมที่ต้องการอ่าน

ค่าที่ส่งกลับ จำนวนข้อมูลที่อ่าน มีหน่วยเป็นจำนวนหน่วยข้อมูล

<stdio.h> FILE *freopen (const char *filename , const char *mode,File *stream);

ความหมาย เปิดไฟล์ใหม่อีกครั้งหนึ่ง

filename ชื่อไดเรกทอรีและชื่อไฟล์ที่กำหนด

mode เช่นเดียวกับ mode ของ fopen( )

stream สตรีมเก่าที่จะปิดลงไป

_fmode (ตัวแปร global)

= O_BINARY ไฟล์ชนิดไบนารี

= O_TEXT ไฟล์ชนิดตัวอักษร

ค่าที่ส่งกลับ NULL ไม่สามารถเปิดไฟล์ได้ตามต้องการ

ค่าอื่นๆ ค่าแอดเดรสของสตรีมใหม่ที่ต้องการ

<stdio.h> int fseek (FILE *stream , long offset , int where);

ความหมาย เลื่อนตัวชี้ข้อมูลไฟล์ไปยังตำแหน่งใหม่

stream สตรีมที่ต้องการเลื่อนตัวชี้ข้อมูลไฟล์

offset ค่าสำหรับใช้เลื่อน มีหน่วยเป็นไบต์

where = SEEK_SET เลื่อนจากตำแหน่งเริ่มต้น

= SEEK_CUR เลื่อนจากตำแหน่งปัจจุบัน

= SEEK_END เลื่อนจากท้ายสตรีม

ค่าที่ส่งกลับ = 0 เลื่อนสำเร็จ

ค่าอื่นๆ ไม่สามารถเลื่อนได้

<stdio.h> size_t fwrite(void *ptr , size_t size ,size_t n ,FILE *stream);

ความหมาย เขียนข้อมูลลงในสตรีม

ptr ตัวชี้ที่ชี้ไปยังจุดเริ่มต้นของตำแหน่งหน่วยความจำที่ต้องการใช้ส่งค่าไปยังสตรีม

size ขนาดของหน่วยข้อมูลที่ต้องการเขียน

n จำนวนของหน่วยข้อมูลที่ต้องการเขียน

stream สตรีมที่ต้องการเขียน

ค่าที่ส่งกลับ จำนวนข้อมูลที่เขียน มีหน่วยเป็นจำนวนหน่วยข้อมูล

<dir.h> int getcurdir(Int drive,char *directory); ใช้สำหรับ DOS

ความหมาย หาไดเรกทอรีปัจจุบัน

drive ไดรฟที่ต้องการหา 0 ไดรฟปัจจุบัน 1 ไดรฟ A 2=B 3=C ...

directory ไดเรกทอรีที่อ่านได้ ไม่รวมหมายเลขไดรฟ

ค่าที่ส่งกลับ = 0 สำเร็จ

= -1 มีข้อผิดพลาดเกิดขึ้น

<dir.h> *getcwd(char *buf , int buflen); ใช้สำหรับ DOS

ความหมาย หาไดเรกทอรีปัจจุบัน

buf ตัวชี้ที่ชี้ไปยังจุดเริ่มต้นของหน่วยความจำที่ใช้รับค่า

buflen ความยาวสูงสุดของหน่วยความจำที่จองไว้

ค่าที่ส่งกลับ ในกรณีที่ buf ไม่เป็น NULL จะให้ค่าเป็นค่าการชี้ของ buf หรือให้ค่าเป็น NULL ในกรณีที่เกิดความผิดพลาด แต่ถ้าใส่พารามิเตอร์ในจุดของ buf เป็น NULL ฟังก์ชันจะจองหน่วยความจำและใส่ไดเรกทอรีลงในหน่วยความจำที่จอง แล้วคืนค่าตำแหน่งหน่วยความจำเริ่มต้นมาให้ สามารถใช้ฟังก์ชัน free( ) เพื่อคืนหน่วยความจำในภายหลังได้

errno = ENODEV ไม่มีไดรฟดังกล่าว

= ENOMEM หน่วยความจำไม่พอ

= ERANGE ค่า buflen น้อยกว่าความยาวของไดเรกทอรีที่อ่านได้

<dir.h> void getdfree(unsigned char drive , struc dfree *dtable); ใช้สำหรับ DOS

ความหมาย หาพื้นที่ที่เหลือในดิสก์

drive หมายเลขไดรฟที่ต้องการหา 0 ไดรฟปัจจุบัน 1 ไดรฟ A 2=B 3=C ...

dtable ค่าการชี้โครงสร้าง

```
struct dfree{
        unsign df_avail;        จำนวนคลัสเตอร์ที่เหลือ
        unsign df_total;        จำนวนคลัสเตอร์ทั้งหมด
        unsigned df_bsec;       จำนวนไบต์ต่อเซกเตอร์
        unsigned df_dclus;      จำนวนเซกเตอร์ต่อคลัสเตอร์
};
```

<io.h> int getftime(Int handle,struc ftime *ftimep); ใช้สำหรับ DOS

ความหมาย หาค่าวันเวลาของไฟล์

handle แฮนเดิลของไฟล์ที่ต้องการหา

ftimep ค่าการชี้โครงสร้าง

```
struc ftime{
        unsigned ft_tsec :5;    หน่วยวินาทีคูณสอง
        unsigned ft_min  :6;    หน่วยนาที
        unsigned ft_hour :5;    หน่วยชั่วโมง
        unsigned ft_day  :5;    หน่วยวัน
        unsigned ft_month:4;    หน่วยเดือน
        unsigned ft_year :7;    หน่วยปีลบด้วย 1980
};
```

ค่าที่ส่งกลับ = 0 สำเร็จ

= -1 ไม่สามารถอ่านวันเวลาได้

errno = EINVFNC หมายเลขฟังก์ชันผิดพลาด

= EBADF แฮนเดิลผิดพลาด

<dir.h> int getdisk (void); ใช้สำหรับ DOS

ความหมาย หาหมายเลขดิกส์ที่กำลังทำงานปัจจุบัน

ค่าที่ส่งกลับ 0=A : 1=B : 2=C...

<io.h> int lseek(int handle , long offset ,Int where);

ความหมาย เลื่อนตัวชี้ข้อมูลไฟล์ให้ชี้ไปยังตำแหน่งใหม่

	handle	แฮนเดิลของไฟล์ที่ต้องการเลื่อนตัวชี้ข้อมูลไฟล์
	offset	ค่าสำหรับใช้เลื่อนไป มีหน่วยเป็นไบต์
	where	= SEEK_SET เลื่อนจากตำแหน่งเริ่มต้น
		= SEEK_CUR เลื่อนจากตำแหน่งปัจจุบัน
		= SEEK_END เลื่อนจากท้ายสตรีม
	ค่าที่ส่งกลับ	= 0 เลื่อนสำเร็จ
		ค่าอื่นๆ ไม่สามารถเลื่อนได้
	errno	= EBADF แฮนเดิลผิดพลาด
		= EINVAL พารามิเตอร์ผิดพลาด
<dir.h>	int mkdir(const char *path);	
	ความหมาย	สร้างไดเรกทอรีใหม่
	path	ไดเรกทอรีใหม่ที่ต้องการกำหนด
	ค่าที่ส่งกลับ	= 0 สร้างสำเร็จ
		= -1 ไม่สามารถสร้างได้
ฟังก์ชัน	int open(const char *path,Int access [,unsigned mode]);	
<io.h>	ความหมาย	เปิดไฟล์
ค่าคงที่	path	ไดเรกทอรีและไฟล์ที่ต้องการเปิด
<fcntl.h> และ	access	ในกรณีที่ไม่มีพารามิเตอร์ mode
<sys\stat.h>		= O_RDONLY อ่านอย่างเดียว
		= O_WRONLY เขียนอย่างเดียว
		= O_RDWR เขียนและอ่านได้
		ในกรณีที่มีพารามิเตอร์ mode
		= O_RDONLY ใช้สำหรับ UNIX
		= O_APPEND เปิดเพื่อเขียนต่อท้ายไฟล์
		= O_CREAT สร้างไฟล์ใหม่ หรือเปิดไฟล์เก่า
		= O_TRUNC ลบไฟล์เดิมที่พบให้เป็นไฟล์ว่าง
	mode	= S_IWRITE กำหนดให้เขียนได้เท่านั้น
		= S_IREAD กำหนดให้อ่านได้เท่านั้น
		= S_IREAD | S_WRITE กำหนดให้อ่านหรือเขียนได้
	ค่าที่ส่งกลับ	=มากกว่าหรือเท่ากับ 0 ค่าแฮนเดิล

= -1 เมื่อไม่สำเร็จ

errno =ENOENT หาไฟล์หรือไดเรกทอรีไม่พบ

=EACCES ไม่สามารถจัดการกับไฟล์นั้นได้

=EMFILE แฮนเดิลไม่พอ

=EINVACC พารามิเตอร์ไม่ถูกต้อง

<stdio.h> perror(const char *s);

ความหมาย พิมพ์รายงานความผิดพลาดไปยังสตรีม stderr

s ค่าการชี้ข้อมูลสตริงรายงานข้อผิดพลาด

<dir.h> int rmdir(const char *path);

ความหมาย ลบไดเรกทอรีทิ้ง

path ไดเรกทอรีที่ต้องการลบทิ้ง

ค่าที่ส่งกลับ = 0 สำเร็จ

= -1 ไม่สามารถเปลี่ยนชื่อไฟล์ได้

errno =ENOENT หาไฟล์หรือไดเรกทอรีไม่พบ

=EACCES ไม่สามารถจัดการกับไดเรกทอรีนั้นได้

หมายเหตุ ไดเรกทอรีที่ต้องการจะลบทิ้ง จะต้องไม่ไฟล์หรือไดเรกทอรีย่อยอยู่ภายใน ต้องไม่ใช่ไดเรกทอรีปัจจุบัน และต้องไม่ใช่ไดเรกทอรีราก (Root Directory)

<stdio.h> void setbuf(FILE *stream , char *buf);

ความหมาย เปลี่ยนบัฟเฟอร์ของสตรีมมายังที่กำหนดใหม่

stream สตรีมที่ต้องการกำหนดบัฟเฟอร์ใหม่

buf ค่าการชี้ไปยังบัฟเฟอร์ใหม่ หากมีค่าเป็น NULLสตรีมนั้นจะไม่มีบัฟเฟอร์

ฟังก์ชัน int setmode (Int handle , Int amode);

<io.h> ความหมาย เปลี่ยนโหมดของไฟล์

handle แฮนเดิลของไฟล์ที่ต้องการเปลี่ยนโหมด

amode = O_BINARY เปลี่ยนเป็นไฟล์ชนิดไบนารี (ไฟล์ทั่วไป)

= O_TEXT เปลี่ยนเป็นไฟล์ตัวอักษร

ค่าที่ส่งกลับ = 0 สำเร็จ

= -1 ไม่สำเร็จ

errno = EINVAL พารามิเตอร์ไม่ถูกต้อง

<stdio.h> int setvbuf(FILE *stream , char *buf ,Int type , size_t size);

ความหมาย กำหนดขนาดและชนิดของบัฟเฟอร์ที่เปลี่ยนใหม่

stream สตรีมของไฟล์ที่ต้องการเปลี่ยนบัฟเฟอร์

buf ค่าการชี้ไปยังบัฟเฟอร์ใหม่

type = _IOFBF full buffer mode

= _IOLBF line buffer mode

= _IONBF no buffer mode

size ขนาดของบัฟเฟอร์ มีค่าไม่เกิน 32767

ค่าที่ส่งกลับ = 0 สำเร็จ

= -1 ไม่สำเร็จ

<stdio.h> Int unlink(const char *filename);

<io.h> ความหมาย ลบไฟล์นั้นทิ้ง

<dos.h> filename ไฟล์ (ต้องใส่ทั้งชื่อไดรฟ ไดเรกทอรี และไฟล์) ที่ต้องการลบทิ้ง

ค่าที่ส่งกลับ = 0 ลบสำเร็จ

= -1 ไม่สามารถลบได้

errno = ENOENT หาไฟล์หรือไดเรกทอรีไม่พบ

= EACCES ไม่สามารถจัดการกับไฟล์นั้นได้

<io.h> int _write (Int handle , void *buf , unsigned len); สำหรับ DOS

int write(Int handle , void *buf , unsigned len);

ความหมาย เขียนข้อมูลลงไฟล์

handle แฮนเดิลของไฟล์ที่ต้องการเขียน

buf ค่าการชี้ไปยังข้อมูลที่ต้องการเขียน

len จำนวนข้อมูลที่เขียน มีหน่วยเป็นไบต์ มีค่าไม่เกิน 65535

ค่าที่ส่งกลับ จำนวนข้อมูลที่เขียนจริง มีหน่วยเป็นไบต์

errno = EACCES ไม่สามารถจัดการกับไฟล์นั้นได้

= EBADF แฮนเดิลไม่ถูกต้อง

2.3 การเขียนโปรแกรมติดต่อกับระบบ PC-System Programming

2.3.1 การติดต่อกับพอร์ต

พอร์ต (Port) คือฮาร์ดแวร์ที่ทำหน้าที่รับส่งข้อมูลกับอุปกรณ์อื่นใดของซีพียู

คอมพิวเตอร์ประกอบไปด้วยหน่วยประมวลผลกลาง หน่วยความจำ และหน่วยรับส่งข้อมูลเข้าออก ในเครื่องพีซี หน่วยประมวลผลกลางก็คือซีพียูในตระกูล 80xxx หน่วยความจำในที่นี้ก็คือหน่วยความจำหลักของระบบ การที่ซีพียูจะเขียนหรืออ่านข้อมูลกับส่วนอื่นใดที่นอกเหนือไปจากหน่วยความจำ จะต้องกระทำผ่านทางพอร์ตเสมอ

ในเครื่องพีซี การติดต่อควบคุมดิสก์ ซึ่งเป็นหน่วยความจำสำรอง การควบคุมจอภาพ เครื่องพิมพ์ เมาส์ ล้วนแต่ติดต่อผ่านพอร์ตทั้งสิ้น ยกเว้นในส่วนที่ต้องการความเร็วในการส่งถ่ายข้อมูลและมีปริมาณข้อมูลมาก เช่น การส่งข้อมูลระหว่างดิสก์กับหน่วยความจำหลัก การเขียนหรืออ่านจุดบนตอภาพ จะใช้วิธีการติดต่อแบบอื่น ในกรณีที่ผู้เขียนโปรแกรมต้องการจัดการอุปกรณ์เหล่านี้ด้วยตนเอง โดยไม่ผ่านระบบปฏิบัติการ หรือต้องการจัดการส่วนที่ระบบปฏิบัติการไม่มีความสามารถในการจัดการในส่วนนั้น ผู้เขียนจึงต้องเขียนหรืออ่านข้อมูลกับพอร์ตโดยตรง

ภาษาซีกำหนดโครงสร้างสำหรับติดต่อกับพอร์ตไว้ดังนี้

```
unsigned  inpw(unsigned portid);
int  inp(unsigned portid);
unsigned  outpw(unsigned portid,unsigned value);
int  outp(unsigned portid,int value);                    <conio.h>,
                                                          <dos.h>
int  inport(int  portid);
unsigned   char  inportb(int  portid);
void   outport (int portid,int value);
void  outportb(int  portid,unsigned char value);
```

มาโคร inpw() และ Inport() ทำหน้าที่อ่านข้อมูลขนาด 16 บิต โดยอ่านข้อมูล 8 บิตล่างจากพอร์ตหมายเลข portid และข้อมูล 8 บิตบน จากพอร์ตหมายเลข portid+1 โดยจะส่งค่ากลับเป็นข้อมูลที่อ่านได้

มาโคร inp() และ inportb() อ่านข้อมูลขนาด 8 บิตจากพอร์ตหมายเลข portid และส่ง 8 บิตบนไปที่ portid+1 ค่าที่มาโคร outpw() ส่งกลับก็คือข้อมูลที่ส่งออกไปนั่นเอง

มาโคร outp() และ outportb() ทำหน้าที่เขียนข้อมูล 8 บิตไปที่พอร์ตหมายเลข portid โดยมาโคร outp() จะส่งค่ากลับมาเป็นค่าข้อมูลที่ส่งออกไป

การสื่อสารอนุกรม

พอร์ตสื่อสารอนุกรมเป็นส่วนสำคัญส่วนหนึ่งที่มีอยู่บน คอมพิวเตอร์ส่วนบุคคล พอร์ตสื่อสารอนุกรมมีชื่อเรียกอีกอย่างว่า คอม-พอร์ต ผู้ออกแบบต้องการให้เป็นไปตามมาตรฐานการสื่อสารแบบอนุกรมที่เรียกว่า RS-232C พอร์ตนี้เป็นทางออกของข้อมูลที่ผู้ใช้ต้องการส่งหรือรับ กับระบบที่แตกต่างออกไปได้

พอร์ตสื่อสารอนุกรม RS-232C บนเครื่องไมโครคอมพิวเตอร์ขนาด 16 บิต มีโครงสร้างที่สามารถโปรแกรมด้วยการส่งรหัสคำสั่งให้กับชิปหลักได้ และสามารเลือกการสื่อสารด้วยกระแสวนรอบ หรือแบบแรงดัน (RS-232C) ได้ โดยใช้ไอซีเบอร์ 8250 เป็นหัวใจสำคัญ ต่อมาในไมโครคอมพิวเตอร์รุ่นที่เป็น AT ได้ใช้เบอร์ 16550 แทน ปกติเราจะเรียกว่า UART ภายในจะมีรีจิสเตอร์อยู่ 3 ประเภท

1. รีจิสเตอร์ควบคุม ทำหน้าที่รับคำสั่งจากไมโครโพรเซสเซอร์
2. รีจิสเตอร์สถานะ ทำหน้าที่เก็บสถานะของเหตุการณ์ที่เกิดขึ้นในตัว UART
3. รีจิสเตอร์พักข้อมูล ทำหน้าท่เป็นที่เก็บข้อมูลที่รอการส่งหรือรอการประมวลผล

รีจิสเตอร์ควบคุมมี 4 ตัวคือ

รีจิสเตอร์ควบคุมสายสื่อสาร

ถูกใช้สำหรับตั้งพารามิเตอร์การสื่อสารแต่ละบิตจะมีความหมายดังนี้

0,1 กำหนดความยาว Word

2 กำหนดบิตจบหากเป็น 0 ใช้หนึ่งบิตในการจบ หากเป็น 1 ใช้สองบิตในการจบ ยกเว้นว่าถ้าความยาวเวิร์ดเป็น 5 บิต บิตจบจะยาวหนึ่งบิตครึ่ง

3 กำหนดพาริตี้ถ้าหากเป็น 0 ไม่มีพาริตี้ ถ้าหากเป็น 1 มีการใช้พาริตี้

4 ถ้าหากเป็น 1 เป็นการเลือกใช้พาริตี้คู่ ถ้าหากเป็น 0 เป็นการเลือกใช้พาริตี้คี่ และบิตนี้จะไม่มีความหมายหากบิต 3 ไม่ถูกเซ็ต

5 ถ้าเป็น 1 หมายความว่าใช้ 0 เป็นพาริตี้ ถ้าเป็น 0 หมายความว่าใช้ 1 เป็นพาริตี้

6 ถูกใช้สร้างคำสั่ง break เอาต์พุตจะถูกทำให้เป็น 0 จนกว่าบิต 6 จะเป็น 0

7 ถูกเรียกว่าบิตควบคุมการเข้าถึงแลตช์ตัวหารถ้ามันมีค่าเท่ากับ 1 การดำเนินการอ่านเขียนจะกระทำกับแลตช์ตัวหารของตัวสร้างบอร์ด หากเป็น 0 การเขียนและอ่านจะกระทำกับรีจิสเตอร์พักข้อมูล หรือ รีจิสเตอร์ควบคุมการขัดจังหวะ

รีจิสเตอร์ควบคุมโมเด็ม

ถูกใช้ควบคุมสัญญาณแฮนด์เช็คกิ้งที่ถูกส่งออกมาจาก UART

0 ใช้ควบคุม Data Terminal Ready (DTR)

1 ใช้ควบคุม Request to Send (RTS)

2,3 ใช้ควบคุมขา OUT1&OUT2 ON IBM PC

4 เปิดสภาวะตรวจสอบการทำงาน

5-7 เป็น 0 ตลอดเวลา

รีจิสเตอร์ควบคุมการขัดจังหวะ

0 ได้รับข้อมูล

1 รีจิสเตอร์พักข้อมูลส่งว่าง

2 ได้รับสถานะสายสื่อสาร

3 ได้รับสถานะโมเด็ม

4-7 เป็น 0 เสมอ

รีจิสเตอร์ตัวหารอัตราบอด

อัตราบอดถูกตั้งโดยใส่ค่าตัวหารสัญญาณนาฬิกา(1.8432 MHz) ลงในรีจิสเตอร์ 2 ตัว ความถี่ของผลลัพธ์จากการหารจะเป็น 16 เท่าของอัตราบอด รีจิสเตอร์ 2 ตัวคือ DLL & DLM

รีจิสเตอร์สถานะมี 3 ตัว

รีจิสเตอร์แสดงสถานะการสื่อสารทำหน้าที่ให้ข้อมูลเกี่ยวกับการรับและส่งข้อมูล

0 DR Data Ready ได้รับตัวอักษรเข้ามาและถูกนำไปเก็บไว้ในรีจิสเตอร์พักข้อมูลภาครับ

1 OEOverrun error ได้รับตัวอักษรเข้ามาก่อนที่ตัวอักษรตัวเก่าถูกนำไปประมวลผล

2 PE Parity error บิตพาริตี้ของตัวอักษรที่รับเข้ามาไม่ถูกต้อง

3 FE Framing error ตัวอักษรที่รับเข้ามามีบิตจบไม่ถูกต้อง

4 BI Bieak Interrupt ได้รับสญญาณเบรก คือสายข้อมูลเข้าเป็นศูนย์ นานกว่าความยาวสูงสุดของตัวอักษร

5 THRE Transmitter Holding register empty UART พร้อมที่จะรับตัวอักษรตัวใหม่เพื่อที่จะส่งออกไป

6 TSRE Transmitter Shift register empty TSR กำลังรอรับตัวอักษรจาก THR

8 [SPACE] ถูกเซ็ตเป็น 0 อย่างถาวร

รีจิสเตอร์แสดงสถานะโมเด็ม

0 Delta CTS

1 Delta DSR

2 TERI

3 Delta RLSD

4 CTS

5 DSR

6 RI

7 RLSD

บิตที่ 1,2,4 เรียกว่า เดลต้าบิตใช้บอกว่ามีการเปลี่ยนแปลงนับจากการอ่านรีจิสเตอร์นี้ครั้งล่าสุด รีจิสเตอร์ควบคุมการอินเตอร์รัพต์สามารถ เซ็ตให้สร้างสัญญาณอินเตอร์รัพต์ เมื่อสถานะโมเด็มเปลี่ยนไป บิต 2 แสดงการเปลี่ยนแปลงเมื่อสายสัญญาณ RI ถูกเปลี่ยนจาก 1 เป็น 0 เท่านั้น

รีจิสเตอร์ควบคุมการอินเตอร์รัพต์

รีจิสเตอร์นี้ให้ข้อมูลเกี่ยวกับสถานะปัจจุบันของ อินเตอร์รัพต์ที่ค้างอยู่ บิต 0 จะถูกเซ็ตเป็น 1 หากไม่มีอินเตอร์รัพค้างอยู่ ถ้าเป็น 0 บิต 1-2 จะบอกว่ามีอินเตอร์รัพต์ตัวใดค้างอยู่ ส่วนบิต 3-7 เป็น 0 เสมอ

ตารางที่ 2.7 แสดงสถานะของอินเตอร์รัพต์

2	1	อินเตอร์รัพต์
0	0	ได้รับสถานะโมเด็ม
0	1	รีจิสเตอร์พักข้อมูลที่ส่งว่าง
1	0	ได้รับข้อมูล
1	1	ได้รับสถานะสายสื่อสาร

ตารางที่ 2.8 หมายเลขอินพุตเอาต์พุตพอร์ตของพอร์ตอนุกรม

อินพุตเอาต์พุตพอร์ต		รีจิสเตอร์	สถานะ DLAB
COM1	COM2		
3F8	2F8	บัฟเฟอร์ TX	0

ตารางที่ 2.8 (ต่อ) หมายเลขอินพุตเอาต์พุตพอร์ตของพอร์ตอนุกรม

อินพุตเอาต์พุตพอร์ต		รีจิสเตอร์	สถานะ DLAB
3F8	2F8	บัฟเฟอร์ RX	0
3F8	2F8	แลตซ์ตัวหาร LSB	1
3F9	2F9	แลตซ์ตัวหาร MSB	1
3F9	2F9	อีนาเบิลอินเตอร์รัพต์	
3FA	2FA	กำหนดอินเตอร์รัพต์	
3FB	2FB	ควบคุมสายสื่อสาร	
3FC	2FC	ควบคุมโมเด็ม	
3FD	2FD	แสดงสถานสายสื่อสาร	
3FE	2FE	แสดงสถานะโมเด็ม	

2.3.2 การจัดการอินเตอร์รัพต์

การขัดจังหวะ หรือ การอินเตอร์รัพต์ (Interrupt) คือการที่กระบวนการหนึ่งที่กำลังทำงานอยู่ ถูกสั่งให้หยุดการทำงาน ซึ่งอาจจะสั่งมาจากกระบวนการอื่นจากฮาร์ดแวร์ หรือจากกระบวนการตัวเองก็ได้ เพื่อทำการเริ่มต้นทำงานกระบวนการอีกกระบวนการ และเมื่อกระบวนการนี้ทำงานเสร็จสิ้นแล้ว กระบวนการที่ถูกขัดจังหวะก็จะทำงานต่อไป

โปรแกรมบริการอินเตอร์รัพต์ (Interrupt Service Routine – ISR) คือกระบวนการที่จะถูกเรียกขึ้นมาทำงาน เมื่อเกิดการร้องขออินเตอร์รัพต์

ตารางอินเตอร์รัพต์เวกเตอร์ (Interrupt Vector Table – IVT) คือตารางเก็บค่าตำแหน่งหน่วยความจำเริ่มต้นของฟังก์ชันที่จะเป็นกระบวนการทำงานตอบสนองการร้องขออินเตอร์รัพต์

อินเตอร์รัพต์ฟังก์ชัน (Interrupt Function)เป็นชื่ออีกชื่อหนึ่งของโปรแกรมบริการอินเตอร์รัพต์ ในมุมมองฟังก์ชันตามความหมายของภาษาซี หมายถึงกลุ่มของคำสั่งที่ทำหน้าที่ใดหน้าที่หนึ่ง ซึ่งจะเริ่มทำงานเมื่อได้รับการร้องขออินเตอร์รัพต์ (ตามหมายเลขที่ได้ติดตั้งอินเตอร์รัพต์ฟังก์ชันไว้)

ในขณะที่มีกระบวนการหนึ่งกำลังทำงานอยู่ในระบบ กระบวนการนั้นอาจถูกขัดจังหวะการทำงานได้จากความจำเป็นใดๆ ก็ตามในระบบ ที่ต้องการการจัดการสิ่งใดสิ่งหนึ่งเร่งด่วนกว่ากระบวนการที่กำลังทำงานอยู่ในปัจจุบัน การขัดจังหวะที่เกิดขึ้นจะทำให้กระบวนการที่กำลังทำงานอยู่นั้นหยุดการทำงานลงชั่วคราว เพื่อให้ระบบหันไปทำงานในกระบวนการที่จะใช้จัดการงาน

เร่งด่วนนั้น เมื่องานที่เร่งด่วนกว่าเสร็จสิ้นกระบวนการที่ถูกหยุดค้างไว้ก็จะทำงานต่อไป การถูกขัดจังหวะเพื่อไปกระทำงานที่เร่งด่วนกว่านี้เรียกว่า การอินเตอร์รัพต์ และกระบวนการที่ใช้จัดการงานที่เร่งด่วนกว่านี้เรียกว่า อินเตอร์รัพต์ฟังก์ชัน หรือ โปรแกรมบริการอินเตอร์รัพต์

สำหรับเครื่องพีซี แหล่งที่จะกำเนิดสัญญาณที่จะบอกการเกิดการอินเตอร์รัพต์นี้อาจมากจากฮาร์ดแวร์ เช่นเมื่อเกิดการหารด้วยศูนย์ ซีพียูจะส่งสัญญาณอินเตอร์รัพต์ เพื่อให้ระบบปฏิบัติการทราบว่า ค่าที่ได้จากการคำนวณนั้นเป็นอนันต์เป็นต้น นอกจากนี้ สัญญาณอินเตอร์รัพต์อาจมาจากกระบวนการที่กำลังทำงานอยู่ สั่งขัดจังหวะตนเอง อันเป็นวิธีที่เราจะใช้ในการเรียกอินเตอร์รัพต์ฟังก์ชันของระบบปฏิบัติการ และในกรณีสุดท้ายสัญญาณอินเตอร์รัพต์อาจมาจาก กระบวนการอื่นที่อยู่ในระบบ ซึ่งอาจจะกำลังทำงานไปพร้อมกับกระบวนการปัจจุบัน หรืออาจจะถูกสั่งให้เริ่มทำงานเมื่อมีสัญญาณร้องขออินเตอร์รัพต์ หรือมีการเรียกใช้กระบวนการในนามของฟังก์ชัน เราอาจแยกประเภทของการอินเตอร์รัพต์ตามแหล่งที่มาได้สองชนิดคือ ฮาร์ดแวร์อินเตอร์รัพต์ (Hardware Interrupt) เป็นอินเตอร์รัพต์ที่ถูกร้องขอโดยฮาร์ดแวร์ และซอฟต์แวร์อินเตอร์รัพต์ (Software Interrupt) จะถูกต้องขอจากกระบวนการใดกระบวนการหนึ่งที่กำลังทำงานภายในระบบ

เมื่อมีการร้องขออินเตอร์รัพต์ ไม่ว่าจะมาจากแหล่งใดก็ตาม จะมีการแจ้งหมายเลขอินเตอร์รัพต์มาด้วยเสมอ ค่าหมายเลขอินเตอร์รัพต์นี้จะถูกนำมาเปิดตารางอินเตอร์รัพต์เวกเตอร์ และนำค่าในตารางที่ได้มาเป็นค่าตำแหน่งหน่วยความจำเริ่มต้นของอินเตอร์รัพต์ฟังก์ชัน เพื่อใช้เป็นจุดกระโดดไปทำงานของระบบ

หลังจากที่ระบบทำงานตามอินเตอร์รัพต์ฟังก์ชันเสร็จสิ้นแล้ว ที่คำสั่งสุดท้ายของอินเตอร์รัพต์ฟังก์ชันจะมีคำสั่งที่ใช้กระโดดกลับไปยังจุดที่เรียกมา คำสั่งนี้จะทำให้ระบบกระโดดกลับไปทำงานในกระบวนการที่ค้างอยู่ต่อไป

ในตารางอินเตอร์รัพต์ จะมีอยู่ 256 หน่วย แต่ละหน่วยเป็นค่าการชี้ฟังก์ชันแบบไกล ด้วยเหตุนี้ เครื่องพีซีจึงมีหมายเลขอินเตอร์รัพต์ได้ตั้งแต่หมายเลข 0 ถึง 255 การส่งค่าไปยังอินเตอร์รัพต์ฟังก์ชัน หรือส่งค่ากลับ จะใช้การส่งผ่านค่าทางรีจิสเตอร์ รายละเอียดของอินเตอร์รัพต์ฟังก์ชันที่ใช้งานบนเครื่องพีซี สามารถอ่านได้จากคู่มือโปรแกรมระบบ

ภาษาซี มีฟังก์ชันและมาโครเกี่ยวกับการกระทำอินเตอร์รัพต์อยู่หลายตัว แต่ที่น่าสนใจมีดังนี้คือ

```
void    disable(void);
void    enable(void);
void    geninterrupt(int n);
```

มาโคร disable() ทำหน้าที่สั่งให้ระบบไม่รับอินเตอร์รัพต์ใดๆ ที่อาจจะเกิดขึ้น ใช้ในกรณีที่ไม่ต้องการให้เกิดอินเตอร์รัพต์ในขณะทำงานที่การอินเตอร์จะทำให้เกิดผลเสียหายได้ เช่นในขณะที่กำลังแก้ไขตารางอินเตอร์รัพต์เวกเตอร์ เป็นต้น การยกเลิกการสั่งงานของมาโคร disable () ทำได้โดยใช้มาโคร enable()

มาโคร geninterrupt() เป็นมาโครสำหรับกระทำซอฟต์แวร์อินเตอร์รัพต์ตามหมายเลขที่กำหนด ดังตัวอย่าง เป็นการกระทำอินเตอร์รัพต์เพื่อสั่งให้ระบบเริ่มต้นทำงานใหม่ โดยการใช้อินเตอร์รัพต์หมายเลข 19h มีรายละเอียดดังนี้

INT 19h Interrupt 19h Bootstrap Loader

ความหมาย สั่งให้อ่านไฟล์ระบบปฏิบัติการเข้ามาเริ่มต้นทำงานใหม่

ในความเป็นจริง ฟังก์ชันสำหรับใช้งานในระบบปฏิบัติการมีอยู่เป็นจำนวนมากมาย การที่จะทำให้ฟังก์ชันทั้งหมดใช้หมายเลขอินเตอร์รัพต์ให้พอเพียงจึงต้องมีการใช้หมายเลขอินเตอร์รัพต์ร่วมกันระหว่างฟังก์ชันทำงานในทำนองเดียวกัน โดยจะมีการส่งค่าผ่านทางรีจิสเตอร์เพื่อกำหนดหมายเลขฟังก์ชันย่อย ดังตัวอย่างการกระทำอินเตอร์รัพต์เพื่อสั่งเปลี่ยนโหมดการทำงานของจอภาพด้วยอินเตอร์รัพต์หมายเลข 10h ฟังก์ชัน 00h ดังนี้

Interrupt 10h Function 00h Set video state

ความหมาย เปลี่ยนโหมดจอภาพ

ค่าที่ส่งเข้าไป AH = 00h (หมายเลขฟังก์ชัน)

AL = หมายเลขโหมดที่ต้องการเปลี่ยน

อินเตอร์รัพต์หมายเลข 10h เป็นกลุ่มอินเตอร์รัพต์ฟังก์ชันสำหรับใช้จัดการจอภาพโดยเฉพาะ การเลือกทำงานในฟังก์ชันใดฟังก์ชันหนึ่งจะทำได้โดยส่งค่าหมายเลขฟังก์ชันย่อยทางรีจิสเตอร์ AH และส่งค่าพารามิเตอร์อื่นๆ ไปทางรีจิสเตอร์ตัวอื่น ดังฟังก์ชัน 00h ที่ส่งค่าหมายเลขโหมดที่ต้องการเปลี่ยนทางรีจิสเตอร์ AL

นอกจากการเขียนโปรแกรมเพื่อขออินเตอร์รัพต์จากอินเตอร์รัพต์ฟังก์ชันที่มีอยู่ในระบบแล้ว เราอาจเขียนอินเตอร์รัพต์ฟังก์ชันเองก็ได้ ฟังก์ชันที่เขียนขึ้นนี้จะทำงานในลักษณะของอินเตอร์รัพต์ฟังก์ชันจะต้องมีขั้นตอนต่อไปนี้อยู่ภายในลำดับขั้นตอนการทำงานของโปรแกรมคือ

1. กำหนดหมายเลขอินเตอร์รัพต์ที่ต้องการใช้สำหรับอินเตอร์รัพต์ฟังก์ชันที่เขียนขึ้นซึ่งอาจจะใช้หมายเลขที่ยังไม่มีอินเตอร์รัพต์ฟังก์ชันใดใช้ หรืออาจใช้ร่วมกับหมายเลขที่มีอินเตอร์รัพต์ใช้งานอยู่ก่อนแล้ว ซึ่งในกรณีหลัง เมื่อมีการร้องขออินเตอร์รัพต์ที่ไปเรียกอินเตอร์รัพต์ฟังก์ชันเดิมก็จะเกิดการกระโดดไปทำงานในอินเตอร์รัพต์ฟังก์ชันใหม่แทน ดังนั้น ในอินเตอร์รัพต์ฟังก์ชันตัว

ใหม่ของกรณีนี้ จะต้องมีการเรียกใช้อินเตอร์รัพต์ฟังก์ชันเดิมด้วย เพื่อให้อินเตอร์รัพต์ฟังก์ชันเดิมทำงาน

2. เก็บค่าการชี้อินเตอร์รัพต์ฟังก์ชันเดิมที่ใช้ โดยฟังก์ชัน getvect() ซึ่งมีโปรโตไทป์ดังนี้

void interrupt (*getvect (int n))(void);

คำสงวน Interrupt ที่ปรากฏอยู่ในโปรโตไทป์ เป็นการกำหนดให้ค่าการชี้ฟังก์ชัน getvect() ส่งกลับมานี้เป็นค่าการชี้อินเตอร์รัพต์ฟังก์ชัน ซึ่งหมายถึงค่าตำแหน่งหน่วยความจำเริ่มต้นของอินเตอร์รัพต์ฟังก์ชันนั่นเองฟังก์ชัน getvect() จะให้ค่าการชี้อินเตอร์รัพต์ฟังก์ชันจากอินเตอร์รัพต์หมายเลขที่กำหนดเป็นพารามิเตอร์ n ในฟังก์ชัน

3. กำหนดค่าการชี้อินเอตร์รัพต์ฟังก์ชันของตารางอินเตอร์รัพต์เวกเตอร์ในหมายเลขที่ต้องการด้วยค่าการชี้อินเตอร์รัพต์ฟังก์ชันที่สร้างขึ้น หลังจากที่อ่านค่าการชี้อินเตอร์รัพต์ในตารางอินเตอร์รัพต์เวกเตอร์ตามหมายเลขที่ต้องการด้วยฟังก์ชัน getvect() แล้ว เราก็จะกำหนดค่าการชี้อินเตอร์รัพต์ฟังก์ชันตัวใหม่ลงไปแทนในตัวเก่าที่เก็บค่าไว้ด้วยฟังก์ชัน setvect() เพื่อให้ตารางอินเตอร์รัพต์เวกเตอร์ในหมายเลขที่ต้องการชี้ไปยังอินเตอร์รัพต์ที่สร้างขึ้นแทนค่าเก่า ฟังก์ชัน setvect() มีรูปดังนี้

<dos.h> void setvect (int n,void interrupt (*isr) ());

ฟังก์ชัน setvect() ทำหน้าที่กำหนดค่าการชี้อินเตอร์รัพต์ฟังก์ชันใหม่ในตารางอินเตอร์รัพต์เวกเตอร์ตามหมายเลขอินเตอร์รัพต์ที่กำหนด ค่าที่ฟังก์ชัน setvect() รับไปเป็นพารามิเตอร์คือค่าหมายเลขอินเตอร์รัพต์ที่ต้องการเปลี่ยนค่าและค่าการชี้อินเตอร์รัพต์ฟังก์ชันชนิดไม่มีการส่งผ่านค่าเพราะอินเตอร์รัพต์ฟังก์ชันทั่วไปจะส่งผ่านค่าทางรีจิสเตอร์แทนการส่งผ่านสแต็กดังฟังก์ชันทั่วไป

4. ก่อนที่จะจบการทำงานของโปรแกรม จะต้องคืนค่าการชี้อินเตอร์รัพต์ฟังก์ชันเดิมแก่ตารางอินเตอร์รัพต์ มิฉะนั้นเมื่อโปรแกรมจบการทำงาน มีการนำหน่วยความจำที่โปรแกรมนั้นเคยใช้ไปให้โปรแกรมหรือกระบวนการอื่น และมีการเรียกใช้อินเตอร์รัพต์หมายเลขนี้อีก จะทำให้เกิดการกระโดดไปทำงานยังพื้นที่ที่ไม่ใช้อินเตอร์รัพต์ฟังก์ชัน (เพราะได้นำข้อมูลใหม่ไปใส่แทนแล้ว) จะทำให้ระบบเสียหายได้

อินเตอร์รัพต์ฟังก์ชันในภาษาซีจะมีลักษณะเช่นเดียวกับฟังก์ชันที่มีการส่งค่าเข้าทางเดียวทั่วไป แต่จะมีโปรโตไทป์และส่วนหัวของฟังก์ชันดังนี้คือ

```
void    interrupt ชื่อฟังก์ชัน(unsigned bp,unsigned di,unsign si,
                               unsigned ds,unsigned es,unsign dx,
                               unsigned ex,unsigned bx,unsign ax,
                               unsigned ip,unsigned cs,unsign flag);
```

อินเตอร์รัพต์ฟังก์ชันที่สร้างขึ้นด้วยภาษาซีนั้น คอมไพเลอร์จะเพิ่มคำสั่งที่ใช้ในการเก็บค่ารีจิสเตอร์ต่างๆ ลงในสแต็ก และคืนค่ารีจิสเตอร์ต่างๆ จากสแต็กก่อนที่จะจบอินเตอร์รัพต์ฟังก์ชัน ดังนั้น เมื่อเรากำหนดชื่อของพารามิเตอร์อินเตอร์รัพต์ฟังก์ชันให้สอดคล้องกับตำแหน่งข้อมูลของรีจิสเตอร์ที่เก็บไว้ในสแต็ก เราก็อาจจะนำข้อมูลในตัวแปรเหล่านั้นมาใช้แทนการเรียกจากรีจิสเตอร์ก็ได้ แต่ประโยชน์ที่เห็นได้ชัดในการกำหนดพารามิเตอร์ก็คือ การใส่ค่าต่างๆ ลงในตัวแปรที่กำหนดในพารามิเตอร์ จะเท่ากับเป็นการส่งค่ากลับทางรีจิสเตอร์ด้วย เนื่องจากในการแปลนั้น คอมไพเลอร์จะเพิ่มคำสั่งการนำค่าในสแต็กกลับคืนให้รีจิสเตอร์ก่อนจบฟังก์ชัน ดังนั้นค่าที่เราใส่ลงในตัวแปร (ซึ่งอยู่ในสแต็ก) จึงเท่ากับเป็นการกำหนดค่าลงในรีสิสเตอร์ทางอ้อมด้วย กล่าวโดยสรุป การกำหนดพารามิเตอร์ในอินเตอร์รัพต์ฟังก์ชันจะช่วยให้เราสามารถส่งค่ากลับทางนีจิสเตอร์ไปยังกระบวนการที่เรียกมาได้

หากเราไม่ต้องการที่จะส่งค่ากลับทางรีจิสเตอร์ เราก็สามารถลดรูปโปรโตไทป์ของอินเตอร์รัพต์ฟังก์ชันเหลือดังรูป

void interrupt ชื่อฟังก์ชัน (votd);

ข้อควรจำในการเขียนโปรแกรมที่มีการกำหนดอินเตอร์รัพต์ฟังก์ชัน

1. ขนาดของอินเตอร์รัพต์ฟังก์ชันควรจะสั้นที่สุดเท่าที่จะทำได้ โดยยึดหลักที่ว่า เวลาที่เสียไปในการจัดการอินเตอร์รัพต์ฟังก์ชันทั้งหมดในการร้องขออินเตอร์รัพต์ครั้งหนึ่งๆ จะต้องน้อยกว่าช่วงเวลาในการ้องขอแต่ละครั้ง ยกตัวอย่างเช่น อินเตอร์รัพต์หมายเลข 1Ch จะถูกร้องขอทุกๆ 18.2 ครั้งต่อวินาที ดังนั้นช่วงเวลาในการ้องขอแต่ละครั้งจะมีค่าเท่ากับ 55 มิลลิวินาที เวลาในการทำงานของอินเตอร์รัพต์ฟังก์ชันทุกตัวที่จะถูกเรียกจากอินเตอร์รัพต์หมายเลขนี้จะต้องมีค่าไม่เกิน 55 มิลลิวินาที มิฉะนั้น กระบวนการหลักจะไม่มีโอกาสได้ทำงาน เพราะจะเสียเวลาในการทำอินเตอร์รัพต์หมด หากเวลารวมมีค่าเกินกว่าช่วงเวลาในการร้องขอ จะเกิดการร้องขออินเตอร์รพต์เดิมซ้ำในขณะที่ยังไม่จบการทำงานอินเตอร์รัพต์ ซึ่งถ้ายังคงเกิดสภาวะเช่นนี้เรื่อยไป จะทำให้สแต็กเต็มและระบบหยุดทำงานได้ เพราะการเรียกอินเตอร์รัพต์ฟังก์ชันแต่ละครั้ง จะมีการเก็บค่าตำแหน่งหน่วยความจำของคำสั่งที่ทำงานค้างไว้ในสแต็กเสมอ

สภาวะการเรียกฟังก์ชันตนเองซ้ำซ้อน (Re-entrance) คือสภาวะที่มีการเรียกใช้ฟังก์ชันเดิมในขณะที่ฟังก์ชันนั้นยังทำงานไม่เสร็จสิ้น

2. อย่าลืมที่จะเรียกใช้อินเตอร์รัพต์ฟังก์ชันเดิมในอินเตอร์รัพต์ฟังก์ชันใหม่ที่สร้างขึ้น มิฉะนั้นอินเตอร์รัพต์เดิมจะไม่ถูกจัดการ

3. อย่าลืมที่จะคืนค่าการชี้อินเตอร์รัพต์ฟังก์ชันเดิมให้แก่ตารางอินเตอร์รัพต์เวกเตอร์ก่อนออกจากโปรแกรม หากไม่คืนค่าการชี้แล้ว เมื่อมีการร้องขออินเตอร์รัพต์นี้อีก ก็จะเกิดการกระโดด

ไปยังจุดที่เคยเป็นตัวโปรแกรม ซึ่งหลังจากนั้นอาจจะถูกใช้งานโดยกระบวนการอื่น ทำให้ระบบเสียหายได้

2.3.3 การจัดการคีย์บอร์ด

ฟังก์ชัน bioskey ( ) เป็นฟังก์ชันที่ใช้ในการอ่านค่าสแกนจากคีย์บอร์ด แต่ฟังก์ชันตัวนี้มีข้อจำกัดที่ไม่สามารถอ่านปุ่ม F11 และ F12 ได้ เราอาจแก้ปัญหานี้ได้โดยการติดต่อกับอินเตอร์รัพต์ ฟังก์ชันที่ใช้ในการจัดการคีย์บอร์ดโดยตรง อาศัยอินเตอร์รัพต์หมายเลข 16h ฟังก์ชัน 10h ซึ่งมีรูปแบบทั่วไปดังนี้

INT 16h	Interrupt 16h	Function 10h Get enhanced keystroke and read
FUNC 10h	ความหมาย	อ่านค่ารหัสแอสกีและรหัสสแกนของคีย์บอร์ด
	ค่าที่ส่งเข้าไป	AH = 10h
	ค่าที่ส่งกลับ	AH = รหัสสแกน
		AL = รหัสแอสกี

นอกจากการอ่านค่ารหัสสแกนของคีย์บอร์ดโดยตรงแล้ว เราอาจปรับเปลี่ยนการแสดงไฟ LED บนคีย์บอร์ดได้ตามต้องการ โดยอาศัยการเปลี่ยนค่าในพื้นที่ที่ระบบปฏิบัติการใช้เก็บค่าข้อมูล Global ของระบบ ซึ่งจะมีจุดเริ่มต้นที่ 0040:0000 หน่วยความจำส่วนนี้เรียกว่า BIOS Data Area โดยในตำแหน่งหน่วยความจำ 0040:0017 จะใช้เก็บสถานะของการกดปุ่มพิเศษอันได้แก่ปุ่ม Insert ,Caps Lock , Num Lock , Alt , Ctrl , Shift Left และ Shift Right เมื่อเราเปลี่ยนค่าในบิตที่ 6 5 และ 4 ซึ่งจะมีไฟแสดงบนคีย์บอร์ด แล้วเรียกอินเตอร์รัพต์ฟังก์ชันหมายเลข 0x16 ในฟังก์ชันใดฟังก์ชันหนึ่ง ตัวฟังก์ชันจะทำการเปลี่ยนไฟบนคีย์บอร์ดให้สัมพันธ์กับค่า BIOS Data Area โดยอัตโนมัติ

ทุกครั้งที่มีการกดปุ่มหรือปล่อยปุ่มใดๆ บนคีย์บอร์ด จะมีข้อมูลหนึ่งชุดขนาด 8 บิตออกมาจากคีย์บอร์ด และจะเกิดการร้องขออินเตอร์รัพต์หมายเลข 9h อินเตอร์รัพต์ฟังก์ชันของระบบปฏิบัติการจะรับข้อมูลจากพอร์ตที่ต่อมาจากคีย์บอร์ด นำข้อมูลไปจัดการแปลงและใส่ลงในบัฟเฟอร์ต่อไป และรวมทั้งการตรวจสอบการกดปุ่ม Ctrl + Alt + Delete ด้วย

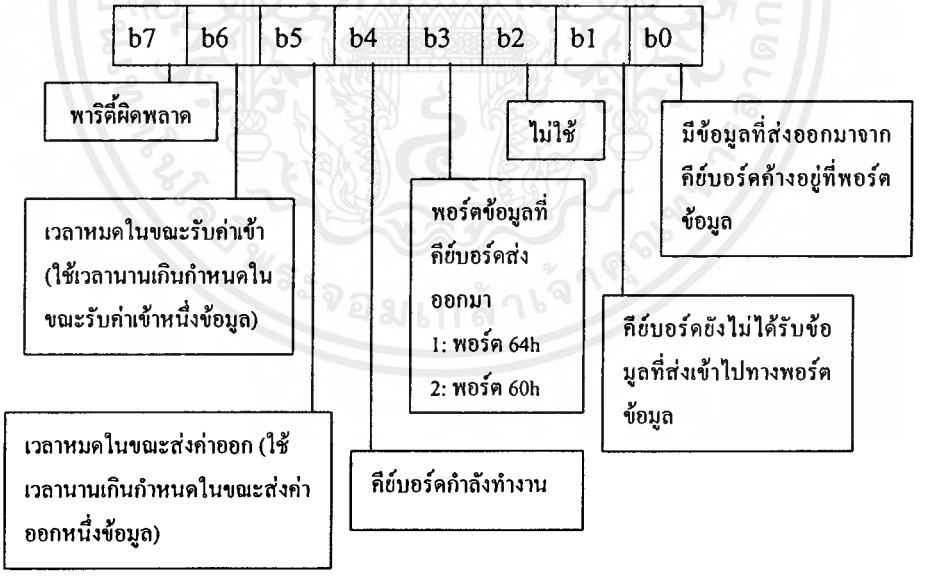
เมื่อผู้ใช้เครื่องกดปุ่มใดๆ จะเกิดข้อมูลออกมาหนึ่งค่า โดยที่บิตบนสุด หรือบิต 7 ของข้อมูลจะมีค่าเป็น 0 แต่ถ้าผู้ใช้ปล่อยปุ่มใดๆ จะเกิดข้อมูลออกมาหนึ่งค่าเช่นกัน แต่บิตบนสุดจะมีค่าเป็น 1 ข้อมูลที่ออกมาเมื่อผู้ใช้กดปุ่มจะเรียกว่า Make Code และข้อมูลที่ออกมาเมื่อผู้ใช้ปล่อยปุ่มเรียกว่า Break Code ข้อมูลเหล่านี้จะเข้ามาทางพอร์ตหมายเลข 60h

คอมพิวเตอร์จะรับรู้ปุ่มผสม ดังเช่น Alt+A ได้จากลำดับข้อมูลที่เข้ามาซึ่งเป็น Make Code ของปุ่ม Alt และตามด้วย Make Code ของปุ่ม A ซึ่งถูกกดภายหลัง ในทำนองเดียวกัน เราจึงเข้าใจ

ได้ว่าปุ่ม Ctrl+Alt+Delete นั้น พีซีจะรับรู้ในลักษณะของลำดับ Make Codeของปุ่ม Ctrl หรือของปุ่ม Alt ตัวใดตัวหนึ่งก่อน แล้วตามด้วย Make Code ของปุ่ม Delete

อินเตอร์รัพต์ฟังก์ชัน NewKeyInt() เป็นฟังก์ชันที่เราจะนำไปแทนอินเตอร์รัพต์หมายเลข 9h เดิมที่ระบบใช้ในการจัดการคีย์บอร์ด ฟังก์ชันที่เราเขียนขึ้นใหม่นี้ จะอ่านค่าข้อมูลที่คีย์บอร์ดส่งมา เพื่อนำไปแสดงบนจอภาพ ค่าที่คีย์บอร์ดส่งมานี้คือค่า Break Code และ Make Code ของคีย์บอร์ดแต่ละปุ่มที่ถูกกดหรือปล่อย

คำสั่งที่ทำหน้าที่บอกให้ส่วนจัดการฮาร์ดแวร์อินเตอร์รัพต์ของเครื่องพีซีรับรู้ว่า จบการกระทำอินเตอร์รัพต์ แล้ว โดยการส่งรหัสบอกการจบอินเตอร์รัพต์ (End-Of-Interrupt –EOI) ซึ่งมีค่า 20h ไปยังวงจรที่ใช้จัดการฮาร์ดแวร์อินเตอร์รัพต์ทางพอร์ตหมายเลข 20h คำสั่งนี้จะต้องเป็นคำสั่งสุดท้ายในส่วนการจัดการฮาร์ดแวร์อินเตอร์รัพต์ทุกตัว นอกจากเราจะอ่านข้อมูลจากคีย์บอร์ดผ่านทางพอร์ต 60h แล้ว เราอาจจะส่งข้อมูลไปยังคีย์บอร์ดเพื่อควบคุมการทำงานได้ โดยการส่งข้อมูลไปทางพอร์ตหมายเลข 60h และอ่านสถานะการทำงานจากพอร์ตหมายเลข 64h ข้อมูลที่ส่งไปยังคีย์บอร์ดจะประกอบไปด้วยคำสั่งและข้อมูลรายละเอียดของคำสั่งนั้น รายละเอียดของสถานะการทำงานที่อ่านได้จากคีย์บอร์ดมีดังนี้



รูปที่ 2.1 สถานะของคีย์บอร์ดที่อ่านได้จากพอร์ต 64h

ในการส่งข้อมูลหนึ่งข้อมูลไปยังคีย์บอร์ด จะเริ่มจากตรวจสอบว่ายังมีข้อมูลค้างอยู่ที่คีย์บอร์ดหรือไม่ โดยตรวจสอบในบิตที่ 1 ของสถานะของคีย์บอร์ด หากมีอยู่จะต้องรอจนกว่า

คีย์บอร์ดนำค่านั้นไปจัดการ เมื่อคีย์บอร์ดได้จัดการข้อมูลนั้นแล้ว บิตที่ 1 จะถูกกำหนดให้เป็น 0 หลังจากนั้นเราจะส่งข้อมูลเข้าไปทางพอร์ต 60h แล้รอข้อมูลที่คีย์บอร์ดส่งตอบกลับมา โดยตรวจสอบที่บิต 0 ซึ่งจะมีค่าเป็น 1 เมื่อคีย์บอร์ดตอบรับข้อมูล แล้วจึงอ่านข้อมูลจากพอร์ต 60h ตรวจสอบว่ามีค่าเป็น FAh หรือไม่ ค่าดังกล่าวแสดงว่าคีย์บอร์ดสามารถนำค่านั้นไปจัดการได้ แต่ถ้ามีข้อผิดพลาด คีย์บอร์ดจะส่งค่าอื่นออกมาแทน ค่าดังกล่าวแสดงว่าคีย์บอร์ดสามาถนำค่านั้นไปจัดการได้ แต่ถ้ามีข้อผิดพลาด คีย์บอร์ดจะส่งค่าอื่นออกมาแทน คำสั่งหนึ่งคำสั่งจะประกอบไปด้วยรหัสของคำสั่ง และตามด้วยข้อมูลที่เป็นพารามิเตอร์ของคำสั่งนั้น ตามปกติ เมื่อเรากดปุ่มคีย์บอร์ดค้างไว้สักครู่หนึ่ง ตัวอักษรที่ตรงกับปุ่มที่กดก็จะออกมาเป็นชุด ซึ่งเป็นผลมาจากการทำงานของคีย์บอร์ดเอง หลังจากที่คีย์บอร์ดส่ง Make Codeออกมาช่วงเวลาหนึ่ง หากยังไม่มีการปล่อยปุ่มที่กด คีย์บอร์ดจะเริ่มส่ง Make Code เดิมตามออกมาอีกเป็นชุด จนกว่าจะปล่อยปุ่มหรือมีการกดปุ่มอื่นเพิ่มลงไปอีก การส่งข้อมูล F3h เป็นคำสั่งกำหนดช่วงเวลาการส่ง Make Code และตามด้วยข้อมูล 8 บิต โดยบิตที่ 6 และ 5 จะเป็ฯการกำหนดช่วงเวลาหน่วง ส่วน 5 บิตล่างจะเป็นการกำหนดความเร็วในการส่ง Make Code ดังตารางต่อไปนี้

ตารางที่ 2.9 การกำหนดช่วงเวลาหน่วง

รหัสในบิตที่ 6 ตามด้วย 5		ช่วงเวลาหน่วง
0	0	¼ วินาที
0	1	½ วินาที
1	0	¾ วินาที
1	1	1 วินาที

ตารางที่ 2.10 การกำหนดความเร็วในการส่ง Make Code

z	ความเร็ว	รหัส	ความเร็ว	รหัส	ความเร็ว	รหัส	ความเร็ว
11111	2.0	10111	4.0	01111	8.0	00111	16.0
11110	2.1	10110	4.3	01110	8.6	00110	17.1
11101	2.3	10101	4.6	01101	9.2	00101	18.5
11100	2.5	10100	5.0	01100	10.0	00100	20.0
11011	2.7	10011	5.5	01011	10.9	00011	21.8

ตารางที่ 2.10 (ต่อ) การกำหนดความเร็วในการส่ง Make Code

z	ความเร็ว	รหัส	ความเร็ว	รหัส	ความเร็ว	รหัส	ความเร็ว
11010	3.0	10010	6.0	01010	12.0	00010	24.0
11001	3.3	10001	6.7	01001	13.3	00001	26.7
11000	3.7	10000	7.5	01000	15.0	00000	30.0

หมายเหตุ ความเร็วมีหน่วยเป็นจำนวนข้อมูลต่อวินาที

ฟังก์ชัน sendtoKBD( ) ทำหน้าที่ส่งข้อมูลหนึ่งตัวไปยังคีย์บอร์ด ภายในฟังก์ชันมีวนรอบชั้นนอกอยู่ 3 วนรอบ ใช้ในการส่งข้อมูลเดิมซ้ำ หากการส่งมีข้อผิดพลาดเกิดขึ้น โดยจะยอมให้ส่งซ้ำได้ไม่เกิน 3 ครั้ง หากผิดพลาดต่อเนื่องเกินกว่าที่กำหนด ฟังก์ชันจะส่งค่ากลับเป็น 0 ภายในวนรอบ เป็นขั้นตอนการส่งข้อมูลหนึ่งค่าไปยังคีย์บอร์ด เริ่มจากการตรวจสอบบิตที่ 1 รอจนกว่าจะมีค่าเป็น 0 โดยการใช้วนรอบ for เพื่อกำหนดจำนวนครั้งในการวนตรวจสอบ การทำเช่นนี้ช่วยทำให้โปรแกรมไม่หยุดรออย่างไม่มีกำหนด หากคีย์บอร์ดทำงานผิดพลาด หลังจากนั้นเป็นการส่งข้อมูลไปยังพอร์ตข้อมูลของคีย์บอร์ด และรอจนกว่าคีย์บอร์ดรับข้อมูลนั้นไปทำงาน แล้วจึงอ่านค่าจากพอร์ตข้อมูลตรวจสอบว่า คีย์บอร์ดได้รับข้อมูลไว้หรือไม่

2.3.4 การจัดการเมาส์

ในการจัดการเมาส์ ซึ่งเป็นอินพุตอีกตัวหนึ่งของระบบ เราจะติดต่อผ่านทางอินเตอร์รัพต์หมายเลข 33h ซึ่งเป็นอินเตอร์รัพต์สำหรับใช้จัดการเมาส์ อินเตอร์รัพต์ฟังก์ชันนี้ จะจัดการในเรื่องของการแสงดเมาส์บนจอภาพในโหมดตัวอักษรและโหมดภาพ (ไม่เกินวีจีเอ) การเปลี่ยนตำแหน่งของเมาส์ และขอบเขตของการแสดงเมาส์

เราอาจแบ่งขั้นตอนการจัดการเมาส์ได้ดังนี้

1. ตรวจสอบว่ามีเมาส์อยู่ในระบบหรือไม่ โดยใช้อินเตอร์รัพต์หมายเลข 33h ฟังก์ชัน 00h เพื่อตรวจสอบว่ามีไดรเวอร์ของเมาส์อยู่ในระบบหรือไม่ หากมีก็จะรีเซ็ตให้เริ่มทำงาน

2. กำหนดรูปแบบในการแสดงพื้นที่ในการแสดงและแสดงตัวชี้ของเมาส์บนจอ ในโหมดกราฟิกนั้น เราสามารถกำหนดรูปของเมาส์ได้ตามต้องการ โดยผ่านทางฟังก์ชัน 09h

3.สอบถามตำแหน่งและสถานะของปุ่ม ในจุดที่เราจะใช้รับข้อมูล เราอาจแสดงตัวเลือกบนจอภาพ และใช้คำสั่งวนรอบเพื่อเรียกใช้ฟังก์ชัน 03h ในการหาตำแหน่งของเมาส์ปัจจุบัน และสถานะของปุ่มที่ผู้ใช้กำลังกดอยู่

4.ซ่อนตัวชี้ของเมาส์ เมื่อจบการทำงาน หรือเมื่อไม่ต้องการให้ตัวชี้ของเมาส์อยู่บนหน้าจอ ผู้เขียนโปรแกรมควรซ่อนตัวชี้ของเมาส์ เพื่อไม่ให้เมาส์ยังคงทำงานอยู่ในระบบปฏิบัติการ หรือในโปรแกรมอื่น ที่ไม่มีการใช้เมาส์ อันอาจจะก่อความสับสนให้กับการใช้งานได้

ฟังก์ชันภายในอินเตอร์รัพต์หมายเลข 33h ที่น่าสนใจมีดังนี้

ฟังก์ชัน 00h รีเซ็ตเมาส์
ค่าที่ส่งเข้าไป AX = 0000h
ค่าที่ส่งกลับ AX = 0 ไม่มีไดร์เวอร์เมาส์อยู่
= 0FFFFh มีเมาส์ติดตั้งอยู่
BX = จำนวนปุ่มของเมาส์
= มากกว่า 2 ปุ่ม
= Microsoft mouse (and compatible)
= Mouse system mouse (3 bottons)
ฟังก์ชัน 01h แสดงตัวชี้ของเมาส์บนจอ
ค่าที่ส่งเข้าไป AX = 0001h
ฟังก์ชัน 02h ซ่อนตัวชี้ของเมาส์
ค่าที่ส่งเข้าไป AX = 0002h
ฟังก์ชัน 03h อ่านสถานะเมาส์
ค่าที่ส่งเข้าไป AX = 0003h
ค่าที่ส่งกลับ BX สถานะปุ่ม
บิต 0 ปุ่มซ้าย บิต 1 ปุ่มขวา บิต 2 ปุ่มกลาง
CX = ตำแหน่งแกน x ของเมาส์
DX = ตำแหน่งแกน y ของเมาส์
ฟังก์ชัน 04h กำหนดตำแหน่งเมาส์
ค่าที่ส่งเข้าไป AX = 0004h
CX = ตำแหน่งแกน x ของเมาส์
DX = ตำแหน่งแกน y ของเมาส์
ฟังก์ชัน 07h กำหนดช่วงแกน x ที่เมาส์จะเลื่อนไปได้
ค่าที่ส่งเข้าไป AX = 0008h
CX = ค่าเริ่มต้น
DX = ค่าสิ้นสุด

ฟังก์ชัน 08h กำหนดช่วงแกน x ที่เมาส์จะเลื่อนไปได้

ค่าที่ส่งเข้าไป AX = 0007h

CX = ค่าเริ่มต้น

DX = ค่าสิ้นสุด

ฟังก์ชัน 09h กำหนดตัวชี้ของเมาส์ในโหมดภาพ (ใช้ได้สูงสุดที่ VGA)

ค่าที่ส่งเข้าไป AX = 0009h

BX = ตำแหน่งของจุดอ้างอิงแกน x เมื่อเทียบกับเมาส์ (-16 ถึง 16)

CX = ตำแหน่งของจุดอ้างอิงแกน y เมื่อเทียบกับเมาส์ (-16 ถึง 16)

ES:DX ค่าการชี้ข้อมูลตัวชี้ของเมาส์ ประกอบไปด้วย

รูปพื้นเดิม (screen mask) 16 word (unsigned int)

ตัวชี้ของเมาส์ (cursor mask) 16 word (unsigned int)

ฟังก์ชัน 0Ah กำหนดตัวชี้ของเมาส์ในโหมดตัวอักษร

ค่าที่ส่งเข้าไป AX = 000Ah

BX เลือกเคอร์เซอร์ที่กำหนดโดย hardware หรือ software

= 0 Software

= 1 Hardware

CX = ค่าของ screen mask หรือจุดเริ่มต้นของ scan line ของเมาส์

DX = ค่าของ cursor mask หรือจุดสิ้นสุดของ scan line ของเมาส์

2.3.5 ฟังก์ชันเกี่ยวกับระบบที่น่าสนใจ

ในหัวข้อนี้เป็นฟังก์ชันเกี่ยวกับการจัดการฐานเวลา การจัดการดิสก์ และการทำงานทั่วไปที่น่าสนใจ

ฟังก์ชันเกี่ยวกับฐานเวลา

<time.h> time_t time(time_t *timer);

ความหมาย ให้ค่าเวลาปัจจุบันออกมาในหน่วยวินาที

timer ค่าการชี้ไปยังตำแหน่งหน่วยความจำที่ใช้รับค่า แต่โดยปกติจะกำหนดให้เป็น NULL

ค่าที่ส่งกลับ ค่าเวลาปัจจุบัน มีหน่วยเป็นวินาที นับจากเวลา 00:00:00 น. ของวันที่ 1 มกราคม ค.ศ.1970 มีหน่วยเทียบเท่า long

<time.h> int stime(time_t *tp);

ความหมาย กำหนดค่าเวลาของระบบ

tp จำนวนวินาทีนับตั้งแต่เวลา 00:00:00 น. ของวันที่ 1 มกราคม ค.ศ. 1970

ค่าที่ส่งกลับ 0

<dos.h> void getdate(struct date *datap);

void setdata (struct date *datap);

ความหมาย รายงานค่าวันเดือนปี และกำหนดค่าวันเดือนปี และกำหนดค่าวันเดือนปีของระบบ

datap ค่าการชี้ข้อมูลโครงสร้างที่ใช้รับค่าหรือกำหนดค่าวันเดือนปี

```
struc data{
        int da_year ; ปี
        int da_day ;  วัน
        int da_mon ;  เดือน  1 = มกราคม  2 = กุมภาพันธ์...
    } ;
```

void gettime(struct time *timep) ;

void settime (struct time *timep) ;

ความหมาย รายงานค่าเวลา และกำหนดค่าเวลาของระบบ

timep ค่าการชี้โครงสร้างที่ใช้รับค่าหรือกำหนดค่าเวลา

```
struct time{
        unsigned char ti_min ;  นาที
        unsigned char ti_hour ; ชั่วโมง
        unsigned char ti_hund ; 1/100 วินาที
        unsigned char ti_sec ;  วินาที
    } ;
```

clock_t clock(void);

ความหมาย บอกจำนวนช่วงเวลานับจากเริ่มต้นสั่งให้โปรแกรมทำงาน

ค่าที่ส่งกลับ มากกว่า 0 จำนวนคาบเวลาที่เกิดขึ้นนับตั้งแต่เริ่มต้น

= -1 ฟังก์ชันไม่สามารถทำงานได้ หรือค่าคาบเวลามีค่ามากเกินไป

หมายเหตุ ในภาษาซี กำหนดตัวแปร CLK_TCK สำหรับใช้เป็นตัวหารคาบเวลาที่ได้ เพื่อให้หน่วยกลายเป็นวินาที

ฟังก์ชันเกี่ยวกับการจัดการดิสก์

<dir.h> int getdisk(void);

int setdisk(drive);

ความหมาย รายงานหมายเลขไดรฟปัจจุบัน และเปลี่ยนช่องไดรฟ

drive หมายเลขดิสก์ไดรฟที่ต้องการเปลี่ยน 0=A : 1=B:...

ค่าที่ส่งกลับ หมายเลขไดรฟ 0=A : 1=B:...

<dos.h> char far *getdta(void) ;

void setdta(char far *dta);

ความหมาย อ่านและกำหนดตำแหน่งปัจจุบันในดิสก์ที่ระบบกำลังเขียนหรืออ่านข้อมูลอยู่

หมายเหตุ ในขณะที่กำลังเปิดไฟล์ในดิสก์ หากมีการใช้คำสั่งค้นหาชื่อไฟล์ในดิสก์ จะทำให้ค่าการชี้สำหรับการเขียนอ่านดิสก์ (Disk Transfer Address – DTA) มีค่าเปลี่ยนไป ฟังก์ชันที่จะทำให้เกิดผลดังกล่าวมีหลายฟังก์ชัน เช่น findfirst( ) และ findnext( ) เป็นต้น เราจึงต้องเก็บค่า DTA ของดิสก์ไว้ก่อน เมื่อกระทำฟังก์ชันดังกล่าวเสร็จสิ้นแล้ว ก็จะต้องเซ็ตค่า DTA คืนให้กับดิสก์

dta ค่า DTA ของดิสก์คอนโทรลเลอร์

ค่าที่ส่งกลับ ค่า DTA

<io.h> int getftime(Int handle , struct ftime *ftimep);

int setftime(Int handle , struct ftime *ftimep);

ความหมาย ค่าแฮนเดิลของไฟล์ที่ต้องการติดต่อ

ftimep ค่าการชี้ข้อมูลโครงสร้าง

```
struct ftime{
        unsigned ff_tsec : 5 ;    หน่วย 2 วินาที
        unsigned ff_min  : 6 ;    นาที
        unsigned ff_hour : 5 ;    ชั่วโมง
        unsigned ff_day  : 5 ;    วัน
        unsigned ff_month : 4 ;   เดือน
        unsigned ff_year : 7 ;    ปี ค.ศ.ลบด้วย 1970
```

};

ค่าที่ส่งกลับ = 0 สำเร็จ

= -1 ไม่สำเร็จ

errno = EINVFNC พารามิเตอร์ไม่ถูกต้อง

= EBADF แฮนเดิลไม่ถูกต้อง

int findfirst(const char *pathname , struct ffblk , Int attrib);

int findnext(struct ffblk *ffblk);

ความหมาย ฟังก์ชัน findfirst( ) ใช้หาชื่อไฟล์ตามที่ต้องการในไดเรกทอรีปัจจุบัน หลังจากใช้ฟังก์ชัน findfirst( ) แล้ว จะหาชื่อไฟล์อื่นที่มีคุณสมบัติเดียวกันได้อีก ด้วยฟังก์ชัน findnext( )

pathname ชื่อไฟล์และไดเรกทอรีที่ต้องการหา ชื่อไฟล์อาจจะอยู่ในรูป widegard (เจาะจงเป็นกลุ่มไฟล์) โดยใช้สัญญลักษณ์ ? และ * ได้ เช่น *.c?? เป็นต้น

ffblk ค่าการชี้ข้อมูลโครงสร้าง

```
struct ffblk {
    char ff_reserved[21] ;   สงวนไว้ (เราจะไม่ใช้)
    char ff_attrib;          สถานะของไฟล์
    int ff_ftime ;           เวลาของไฟล์
                             บิตที่ 0 ถึง 4 หน่วย 2 วินาที
                             บิตที่ 5 ถึง 10 นาที
                             บิตที่ 11 ถึง 15 ชั่วโมง
    int ff_date ;            วันของไฟล์
                             บิตที่ 0 ถึง 4 วัน
                             บิตที่ 5 ถึง 8 เดือน
                             บิตที่ 9 ถึง15 ปีค.ศ.ลบด้วย
                             1970
    long ff_fsize ;          ขนาดของไฟล์
    char ff_name [13] ;      ชื่อและนามสกุลของไฟล์
};
```

attrib = FA_RDONLY Read only
= FA_HIDDEN Hidden file
= FA_SYSTEM System file
= FA_LABEL Volume Label
= FA_DIREC Directory
= FA_ARCH Archive
การใส่ attrib ใน findfast( ) อาจใช้หลายตัวกระทำ OR กันได้

ค่าที่ส่งกลับ = 0 หาได้สำเร็จ
= -1 ไม่สามารถหาไฟล์ตามต้องการได้

errno = ENOENT หาไฟล์หรือไดเรกทอรีไม่พบ
= ENMFILE ไม่มีไฟล์ที่ต้องการอีกแล้ว

ฟังก์ชันเกี่ยวกับการทำงานทั่วไป

<dos.h> void ctrlbrk (*handler)(void);

ความหมาย กำหนดตัวชี้ฟังก์ชันสำหรับใช้จัดการเมื่อกดปุ่ม Ctrl+Break

handler ตัวชี้ฟังก์ชันที่ต้องการ

หมายเหตุ หากต้องการใช้งานฟังก์ชันนี้ ควรใช้เป็นคำสั่งแรกๆ ในโปรแกรม

<stdio.h> char *getenv(const char *name) ;

int putenv(const char *name) ;

ความหมาย อ่านหรือเพิ่มเติมสภาวะแวดล้อมของโปรแกรม

name คำสั่งในสภาวะแวดล้อมที่ต้องการ เช่น PATH

**environ คือตัวแปร global ชนิดอาร์เรย์ของตัวแปรสตริง เก็บค่าสภาวะแวดล้อมของโปรแกรม การแก้ไขจะต้องใช้ฟังก์ชัน putenv( ) แต่เราอาจใช้ตัวแปรนี้ในการรายงานสภาวะแวดล้อมทั้งหมดได้

int setcbrk (Int status);

ความหมาย กำหนดการยอมับการกดปุ่ม Ctrl+Break

status = 0 ไม่ตอบสนองปุ่ม
= 1 ตอบสนองการกดปุ่ม

ค่าที่ส่งกลับ ค่า status

void sleep(unsigned seconds) ;

ความหมาย หน่วงเวลาการทำงานของโปรแกรม

seconds เวลาที่หน่วง มีหน่วยเป็นวินาที

int bioscom(Int cmd ,char abyte , Int port);

ความหมาย จัดการพอร์ตอนุกรม (Serial port) ของระบบ

cmd = 0 กำหนดสถานะให้พอร์ตอนุกรม

= 1 ส่งข้อมูลออกไปยังพอร์ตอนุกรม

= 2 รับข้อมูลจากพอร์ตอนุกรม

= 3 อ่านสถานะปัจจุบันของพอร์ตอนุกรม

port = 0 พอร์ตอนุกรมหมายเลข 1 (COM1)

= 1 พอร์ตอนุกรมหมายเลข 2 (COM2)

abyte ในกรณีที่ cmd=0 ค่าใน abyte จะแบ่งออกเป็นสี่กลุ่ม ให้นำค่าใดค่าหนึ่งในแต่ละกลุ่มมากระทำ OR กัน เพื่อให้ได้สถานะที่ต้องการ

กลุ่มที่ 1	0x02 7 bit data	กลุ่มที่ 4	0x00 110 baud
	0x03 8 bit data		0x20 150 baud
กลุ่มที่ 2	0x00 1 stop bit		0x40 300 baud
	0x04 2 stop bit		0x60 600 baud
กลุ่มที่ 3	0x08 odd parity		0x80 1200 baud
	0x18 even parity		0xa0 2400 baud
			0xc0 4800 baud
			0xe0 9600 baud

ในกรณีที่ cmd=1 ค่าใน abyte คือข้อมูลขนาด 8 บิต ที่ต้องการส่งออก

ค่าที่ส่งกลับ แบ่งออกเป็น 2 ส่วนคือ 8 บิตบน และ 8 บิตล่าง

ค่าใน 8 บิตบน คือสถานะของพอร์ต มีดังนี้

bit 15 Timeout

bit 14 Transmit shift register empty

bit 13 Transmit holding register empty

bit 12 Break detect

bit 11 Framing error

bit 10 Parity error

bit 9 Overrun error

bit 8 Data ready

ในกรณีที่ cmd = 2 ค่า 8 บิตล่างจากเป็นค่าข้อมูลที่ได้รับจากพอร์ต

ในกรณีที่ cmd = 3 ค่า 8 บิตล่างจะเป็นสถานะของพอร์ต ดังนี้

bit 7 Received line signal detect

bit 6 Ring indicator

bit 5 Data set ready

bit 4 Clear to send

bit 3 Change in receive line signal indicator

bit 2 Trailing edge ring detector

bit 1 Change in data set ready

bit 0 Change in clear to send

รายละเอียดเกี่ยวกับความหมายของสถานะต่างๆ อ่านได้จากหนังสือเกี่ยวกับการติดต่อสื่อสารของคอมพิวเตอร์ และคู่มือการโปรแกรมระบบ

<bios.h> int biosdisk (int cmd , int drive , int head , int track ,
int sector , int nsectors , void *buffer);

ความหมาย จัดการดิสก์

ข้อควรระวัง ให้แน่ใจว่าเข้าใจคำสั่งและความหมายอย่างถ่องแท้ เพราะการใช้คำสั่งผิดพลาด จะก่อให้เกิดผลโดยตรงกับดิสก์อาจทำให้ดิสก์เสียหายได้

drive = 0 Floppy diskdrive ตัวที่ 1

= 1 Floppy diskdrive ตัวที่ 2

= 0x80 Harddisk ตัวที่ 1

= 0x81 Harddisk ตัวที่ 2

head หมายเลขหัวอ่าน = 0 หัวอ่านที่ 1 =1 หัวอ่านที่ 2

track แทร็คที่ต้องการ

sector	เซ็กเตอร์ที่ต้องการ	
nsectors	จำนวนเซ็กเตอร์ที่ต้องการกระทำการ	
buffer	บัฟเฟอร์สำหรับรับส่งข้อมูลกับดิสก์	
cmd	= 0	รีเซ็ตดิสก์ไดร์ฟ (พารามิเตอร์ตัวอื่นจะไม่มีผล)
	= 1	รายงานสถานะของดิสก์ที่เพิ่งใช้ (พารามิเตอร์อื่นไม่มีผล)
	= 2	อ่านข้อมูลจากดิสก์
	= 3	เขียนข้อมูลลงดิสก์
	= 4	ตรวจสอบข้อมูลในดิสก์
	= 5	ฟอร์แมตแทร็ก
หมายเหตุ	ยังมีโหมดการทำงานอื่นๆ อีกมากมาย รายละเอียดสามารถอ่านได้จากคู่มือคอมไพเลอร์	
ค่าที่ส่งกลับ	= 0x00	ทำงานได้สมบูรณ์
	= ค่าอื่น ๆ	มีความผิดพลาดเกิดขึ้น

2.4 การเขียนโปรแกรมกราฟฟิกโหมด

โดยปกติ การทำงานของโปรแกรมจะอยู่ในโปมดตัวอักษรหากเราจะต้องการให้โปรแกรมทำงานในโหมดภาพ จะต้องใช้ฟังก์ชั่นสำหรับเปลี่ยนโหมดให้มาอยู่ในโหมดภาพ และเมื่อต้องการเปลี่ยนกลับไปสู่โหมดตัวอักษร จะต้องใช้ฟังก์ชั่นปิดโหมดภาพ ฟังก์ชั่นที่ใช้ในการเปลี่ยนโหมดมีดังนี้คือ

```
void far initgraph(int *graphdriver,int far *graphmode,char far *pathtodriver);
void far closegraph(void);
```

ฟังก์ชั่น initgraph() เป็นฟังก์ชั่นสำหรับเปลี่ยนโหมดการทำงานมาเป็นโหมดภาพ ส่วนฟังก์ชั่น closegraph() ทำหน้าที่ปิดโหมดภาพให้กลับไปสู่โหมดตัวอักษร

พารามิเตอร์ใน initgraph() มีรายละเอียดดังนี้

graphdriver เป็นตัวชี้สำหรับชี้ไปยังหมายเลขโหมดของจอภาพที่ต้องการ โดยทั่วไป เราจะกำหนดให้เท่ากับ DETECT ซึ่งเป็นค่าคงที่ที่ได้นิยามไว้ใน graphics.h หมายถึงให้ใช้โหมดภาพที่มีความละเอียดสูงสุดที่การ์ดสามารถจะทำงานได้

graphmode เป็นตัวชี้สำหรับชี้ไปยังหมายเลขโหมดย่อย บอกระดับความละเอียดที่ต้องการในการใช้งานทั่วไปเราจะไม่กำหนดค่านี้เมื่อเราใช้ค่าคงที่ DETECT ในการกำหนดโหมดจอให้สูงสุดอยู่แล้ว

pathtodriver คือข้อมูลสตริง ที่หมายถึงชื่อไดเรกทอรี่ของไดรเวอร์จอภาพ ในการเขียนโปรแกรมภาษาซีและปาสคาลโดยใช้คอมไพเลอร์จากบริษัทเบอร์แลนด์ซีนั้น จะแยกโมดูลส่วนควบคุมการแสดงผลออกมาต่างหาก เราเรียกส่วนควบคุมนี้ว่า กราฟฟิกไดรเวอร์ (Graphic Driver) มีนามสกุลเป็น BGI และยังมีไฟล์รูปแบบตัวอักษรพิเศษที่มีนามสกุล CHR การกำหนดไดเรกทอรี่ในส่วนนี้จะทำให้โปรแกรมที่เขียนขึ้น เรียกกราฟฟิกไดรเวอร์และไฟล์รูปแบบตัวอักษรพิเศษจากไดเรกทอรี่ที่กำหนด แต่ถ้าเราไม่กำหนดค่าโปรแกรมจะค้นหาจากไดเรกทอรี่ปัจจุบันแทน ดังนั้น ผู้เขียนโปรแกรมจะต้องตรวจสอบไดเรกทอรี่ปัจจุบันว่ามีไฟล์นาสกุล BGI และไฟล์นามสกุล CHR ที่ต้องการหรือไม่ กราฟฟิกไดรเวอร์ที่เราจะใช้งานประจำ จะเป็นของจอ EGA และ VGA ซึ่งรวมอยู่ด้วยกันมีชื่อว่า EGAVGA.BGI

ในโหมดภาพมาตรฐานบนจอวีจีเอ ที่ได้จากการใช้ฟังก์ชั่น initgraph() ของภาษาซี จะมีความละเอียด 640 คูณ 480 จุด 16 สี ค่าโคออร์ดิเนตของมุมบนซ้ายของจอมีค่า (0,0) และค่ามุมล่างขวาจะมีค่า (639,749) จำนวนจุดในแกน x และ y เราสามารถใช้ฟังก์ชั่น getmaxx() และ getmaxy() หาค่าดังกล่าวได้ ฟังก์ชั่นทั้งสองมีรูปทั่วไปดังนี้

```
int far getmaxx(void);
int far getmaxy(void);
```

ในการเขียนโปรแกรมใช้งานจริง ผู้เขียนโปรแกรมอาจจะต้องการรวมกราฟิกไดรเวอร์ของภาษาซีเข้ากับตัวโปรแกรม เพื่อขจัดปัญหาที่โปรแกรมไม่สามารถทำงานได้ เนื่องจากไม่พบไฟล์กราฟฟิกไดรเวอร์ ขั้นตอนในการจัดการเริ่มจากสร้างไฟล์ EGAVGA.OBJ จากไฟล์ EGAVGA.BGI ด้วยโปรแกรม BGIOBJ.EXE ที่มีกับคอมไพเลอร์ โดยใช้คำสั่งดังนี้

```
bgiobj egavga.bgi egavga.obj EGAVGA_driver
```

พารามิเตอร์ตัวแรกของ BGIOBJ.EXE คือกราฟฟิกไดรเวอร์ที่เราต้องการเปลี่ยน พารามิเตอร์ตัวที่สองคือชื่อออบเจ็คต์ไฟล์ และพารามิเตอร์ตัวที่สามคือชื่อที่เราจะเรียกใช้ในโปรแกรม

ในการเขียนโปรแกรม เราจะใช้ฟังก์ชั่น registerbgidriver() เพื่อกำหนดให้คอมไพเลอร์เชื่อมตัวฟังก์ชั่นในไดรเวอร์เข้ากับตัวโปรแกรม เราจะต้องเปิดโปรเจคต์เพื่อเชื่อมออบเจ็คต์ที่ได้จากโปรแกรม BGIOBJ.EXE เข้ากับโปรแกรมที่เขียนขึ้น

ฟังก์ชั่นเกี่ยวกับการวาดภาพ

void far arc(int x,int y,int stangle,int endangle,int radius);

void far circle(int x,int radius);

void far pieslice(int x,inty,int stangle,int endangle,int radius);

ความหมาย	arc()	วาดส่วนของวงกลม
	circle()	วาดวงกลม
	pislice()	วาดส่วนของวงกลม และระบายภายใน ส่วนของวงกลม
(x,y)	จุดศูนย์กลาง	
stangle	มุมเริ่มต้น มีหน่วยเป็นองศา	
endangle	มุมสิ้นสุด มีหน่วยเป็นองศา	
radius	รัศมี	

void far bar(int lect,int top,int right,int bottom);

void far bar3d(int left,int top,int right,int bottom,int depth,int topflag);

void far rectangle(int lect,int topo,int right,int bottom);

ความหมาย	bar()	ระบายรูปสี่เหลี่ยมผืนผ้าทึบ
	bar3d()	ระบายรูปกล่องสามมิติ
	rectangle()	เขียนรูปสี่เหลี่ยมผืนผ้า
(left,top)	มุมบนซ้าย	
(right,bottom)	มุมล่างขวา	
depth	ค่าความลึกของกล่องสามมิติปกติจะใช้ค่า 25 เปอร์เซ็นต์ของความกว้าง	
topflag	= 0	ไม่ระบายส่วนบนของกล่อง
	ค่าอื่นๆ	ระบายส่วนบนของกล่อง

void far drawpoly(int numpoints,int far *polypoints);

void far fillpoly(int numpoints,int far *polypoints);

ความหมาย	drawpoly()	วาดรูปหลายเหลี่ยม
	fillpoly()	วาดรูปทึบหลายเหลี่ยม
numpoints	จำนวนจุดมุมของรูปหลายเหลี่ยม	
polypoints	อาร์เรย์ของโคออร์ดิเจตบอกจุดมุม มีรูปดังเช่น	
	{x0,y0,x1,y1,x2,y2,…….,xn,yx,x0,y0};	

void far ellipse(int x,int y,int stangle,int endangle,int xradius,int yradius);

void far fillellipse(int x,int y,int xradius,int yradius);

ความหมาย	ellipse()	วาดส่วนของวงรี
	fillellipse()	วาดวงรีและระบายสีภายใน
(x,y)	จุดศูนย์กลางวงรี(อยู่ระหว่างจุดศูนย์กลางย่อยสองสุดในวงรี)	
stangle	มุมเริ่มต้น	
endangle	มุมสิ้นสุด	
xradius	ค่ารัศมีในแกนแนวนอน	
yradius	ค่ารัศมีในแกนแนวตั้ง	

void far floodfill(int x,int y,int border);

ความหมาย	ระบายสีในบริเวณที่กำหนด
(x,y)	จุดเริ่มต้นของการระบาย จะต้องอยู่ภายในบริเวณที่กำหนด
border	สีที่แสดงขอบเขตของการระบาย

void far getimage(int left,int top,int right,int bottom,void far *bitmap);

void far putimage(int left,int top,void far *bitmap,int op);

unsigned far imagesize(int left,int top,int right,int bottom);

ความหมาย	getimage()	อ่านข้อมูลภาพบนจอในบริเวณที่กำหนดมาเก็บไว้ในตัวแปร
	putimage()	เขียนภาพบนจอจากที่เก็บไว้โดยฟังก์ชั่น getimage()
	imagesize()	หาขนาดของพื้นที่ ที่ต้องใช้เก็บภาพในบริเวณที่กำหนด
(left,top)	มุมบนซ้าย	
(right,bottom)	มุมล่างขวา	
bitmap	ค่าการชี้ไปยังพื้นที่หน่วยความจำที่ใช้เก็บภาพที่ได้	
op	= COPY_PUT	ใส่ข้อมูลลงไปตรงๆ
	= XOR_PUT	นำข้อมูลไป Exclusive OR กับข้อมูลบนจอ
	= OR_PUT	นำข้อมูลไป OR กับข้อมูลบนจอ
	= AND_PUT	นำข้อมูลไป AND กับข้อมูลบนจอ
	= NOT_PUT	ใส่ข้อมูลลงไป โดยจะกระทำ NOT กับตัวข้อมูลก่อนใส่ลงในจอ

unsigned far getpixel(int x,int y);

void far putpixel(int x,int y,int color);

ความหมาย getpixel() อ่านค่าสีจากจุดภาพที่กำหนด

putpixel() เขียนจุดภาพด้วยสีที่กำหนด

(x,y) จุดที่อ่านหรือเขียน

ค่าที่ส่งกลับ getpixel() ให้ค่าสีจากจุดที่กำหนด

void far line(int x1,int y1,int x2,int y2);

void far linerel(int dx,int dy);

void far lineto(int x,inty);

ความหมาย ลากเส้น

(x1,y1) จุดเริ่มต้น

(x2,y2) จุดสิ้นสุด

(dx,dy) จุดที่อยู่ห่างจากจุดเดิมเป็นระยะ dx ในแนวนอน และ dy ในแนวตั้ง

(x,y) จุดปลายทาง(ลากต่อจากตำแหน่งสุดท้ายของคำสั่งก่อน)

void far outtextxy(int x,int y,char far *textstring);

ความหมาย แสดงตัวอักษรบนจอภาพ

(x,y) จุดเริ่มต้นแสดงตัวอักษร

textstring ข้อมูลสตริงที่จะใช้แสดง

ฟังก์ชั่นเกี่ยวกับการควบคุมการแสดงภาพ

void far setactivepage(int page);

void far setvisualpage(int page);

ความหมาย setactivepage() กำหนดหมายเลขเฟรมภาพที่ใช้วาดภาพ

setvisualpage() กำหนดหมายเลขเฟรมภาพที่จะใช้แสดงบนจอ

page ค่าเฟรมภาพ ขึ้นอยู่กับโหมดของจอภาพแต่ละโหมด

void far cleardevice(void);

void far clearviewport(void);

void far getviewsettings(struct viewporttype far *viewport);

void far setviewport(int left,int top,int right ,int bottom,int clip);

ความหมาย cleardevice() ลบจอภาพ

clearviewport() ลบจอในส่วนที่กำหนด

getviewsettings() รายงานการกำหนดพื้นที่ทำงานในจอภาพ

setviewport(0 กำหนดพื้นที่ทำงานในจอภาพ

viewport ค่าการชี้ข้อมูลโครงสร้าง

```
struct viewporttype{
int left,top,right,bottom;
int clip;
};
```

clip = 0 วาดรูปตามปกติ

= ค่าอื่นๆ หากรูปที่วาดมีส่วนออกไปนอกพื้นทีทำงาน จะถูกตัดทิ้ง

int far graphresult(void);

ความหมาย รายงานรหัสผิดพลาดของการทำงานในโหมดภาพ

ค่าที่ส่งกลับ = 0 ไม่มีความผิดพลาด(ที่คำสั่งสุดท้าย)

ค่าอื่นๆ คำสั่งสุดท้ายมีความผิดพลาด

char far *grahperrormsg(int errcode);

ความหมาย ให้ข้อมูลสตริงอธิบายข้อผิดพลาด

errorcode ค่าที่ได้จาก graphresult()

int far getbkcolor(void);

void far setbkvolor(int volor);

int far getcolor(void);

void far setcolor(int color);

ความหมาย อ่านค่าสีพื้น กำหนดค่าสีพื้น อ่านค่าสีที่ใช้วาด และกำหนดสีที่ใช้วาด color และ ค่าที่ส่งกลับ เป็นค่ารหัสสี(ดูรหัสสีของตัวอักษรประกอบ)

void far getfillpattern(char far *userattern);

void far getfillsettings(struct fillsettingsytle far *fillinfo);

void far setfillpattern(char far *userpattern,int color);

void far setfillsytle(int pattern,int color);

ความหมาย getfillpattern() และ setfillpattern() รายงานและกำหนดรูปแบบการใส่ลายพื้นตามที่ผู้ใช้กำหนด

getfillsettings() รายงานเกี่ยวกับสถานะของลายพื้นและสีที่ได้กำหนด

setfillstyle() กำหนดลายพื้นมาตรฐาน

userpattern อาร์เรย์ขนาด 8 ไบต์ เช่น

```
char gray50[8] = {0xaa,0x55,0xaa,0x55,0xaa,0x55,0xaa,0x55};
```

color รหัสสีที่กำหนด(ดูรหัสสีตัวอักษร)

pattern มีหลายชนิดเช่น SLASH_FILL,BKSLASH_FILL,LTBKSLASH_FILL, EMPTY_FILL,SOLID_FILL,LINE_FILL,HATCH_FILL เป็นต้น

Fillinfo ค่าการชี้ข้อมูลโครงสร้าง

```
Struct fillsettingstyle{
        Int pattern;
        Int color;
};
```

```
void far getlinesettings(struct linesettingstyle far *lineinfo);
void far setlinestyle(int linestyle,unsigned pattern,int thickness);
```

ความหมาย รายงานและกำหนดลักษณะการลากเส้น

linelifo ค่าการชี้ข้อมูลโครงสร้าง

```
struct linesettingstyle{
        int linestyle;
        unsigned pattern;
        int thicknessl
};
```

linestyle = SOLID_LINE,DOTTED_LINE,CENTER_LINE,DASHED_LINE

pattern ใช้ในกรณีที่ linesytle กำหนดเป็น USERBIT_LINE ค่านี้จะแทนจุดเรียงกัน 16 จุด ค่าบิต 0 หมายถึงไม่ใส่จุด ค่าบิต 1 คือใส่จุด ช่วงของการใส่หรือไม่ใส่จุดนี้จะปรากฏบนเส้น

thickness ความหนาของเส้น

```
int far getmaxcolor(void);
```

ความหมาย รายงานค่ารหัสสีสูงสุดที่ใช้แสดงได้

```
void far getx(void);
int far gety(void);
```

ความหมาย รายงานตำแหน่งจุดอ้างอิงสำหรับการวาดปัจจุบัน

```
void far moverel(int dx,int dy);
void far moveto(int x,int y);
```

ความหมาย เลื่อนจุดอ้างอิงสำหรับการวาดไปยังจุดใหม่

(dx,dy) จุดอ้างอิงใหม่ซึ่งห่างจากจุดเดิม dx จุดในแนวนอน และ dy จุดในแนวตั้ง

(x,y) จุดใหม่

void far gettextsettings(struct textsettingstype far *texttypeinfo);

void far settextjustify(int horz,int vert);

void far settextstyle(int font,int direction,int charsize);

ความหมาย gettextsettings() รายงานค่ากำหนดของการแสดงตัวอักษร

settextjustify() จัดการวางตัวอักษร

settextstyle() กำหนดรูปแบบตัวอักษร

texttypeinfo ค่าการชี้ขอมูลโครงสร้าง

```
struct textsettingstype{
        int font;
        int direction;
        int charsize;
        int horiz;
        int vert;
};
```

horiz	= LEFT_TEXT	จัดชิดซ้าย
	= CENTER_TEXT	จัดวางกลาง
	= RIGHT_TEXT	จัดชิดขวา
vert	= BOTTOM_TEXT	จัดชิดล่าง
	= CENTER_TEXT	จัดชิดกลาง
	= TOP_TEXT	จัดชิดบน

หมายเหตุ ค่า horiz มีผลกับตัวอักษรที่วางแนวนอน ส่วน vert มีผลกับที่วางแนวตั้ง

font = DEFAULT_FONT ตัวอักษรปกติ

=TRIPLEX_FONT,SMALL_FONT,SANS_SERIF_FONT,
GOTHIC_FONT

direction	= HORIZ_DIR	วางตัวอักษรแนวนอน
	= VERT_DIR	วางตัวอักษรแนวตั้ง

charsize ขนาดตัวอักษรเป็นเท่าจากขนาดเดิม

void far setwritemode(int mode);

ความหมาย	กำหนดวิธีการวาดภาพบนจอ	
mode	= COPY_PUT	วาดตามปกติ
	= XOR_PUT	นำจุดบนจอมา Exclusive OR กับจุดใหม่

บทที่ 3

การออกแบบและการสร้าง

ในส่วนของโปรแกรมทั้งหมดจะมีส่วนประกอบที่สำคัญอยู่ 7 ส่วน คือ

1. ส่วนของเมนู
2. ส่วนของการติดต่อกับไฟล์
3. ส่วนของการแสดงลายวงจร
4. ส่วนของการรับส่งข้อมูล
5. ส่วนของการติดต่อกับเครื่องพิมพ์
6. ส่วนของการแสดงรายละเอียดการใช้งานโปรแกรมและรายละเอียดของผู้จัดทำ

3.1 การออกแบบส่วนของเมนู

เมนูจะเป็นส่วนที่ให้ผู้ใช้เลือกการใช้งาน ซึ่งมีลักษณะจอภาพดังนี้

รูปที่ 3.1 หน้าต่างการทำงานของโปรแกรม

จะประกอบไปด้วยเมนูหลัก 6 ส่วนและแต่ละส่วนก็จะมีเมนูย่อยอีกดังนี้

ส่วนของเมนู File จะมีเมนูย่อย 6 ส่วนคือ New ,Open ,Save ,Change Drv ,Dos และ Quit

ส่วนของเมนู โหมด จะมีเมนูย่อย 2 ส่วนคือ Auto, Manual และRemote

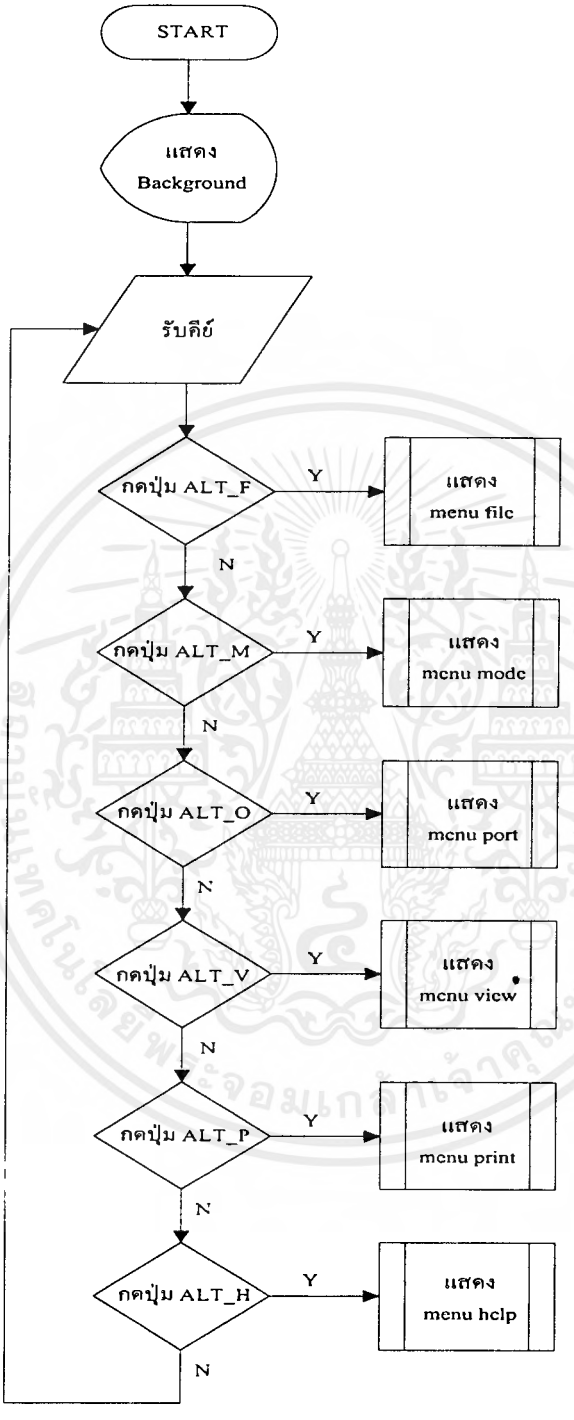
ส่วนของเมนูพอร์ตจะมีเมนูย่อย 2 ส่วนคือ Com และ เซตting ในส่วนของ Com ก็จะย่อยอีกเป็น2 ส่วนคือ Com 1 และ Com 2

ส่วนของเมนู View จะมีเมนูย่อย 2 ส่วน คือ Zoom in และ Zoom Out

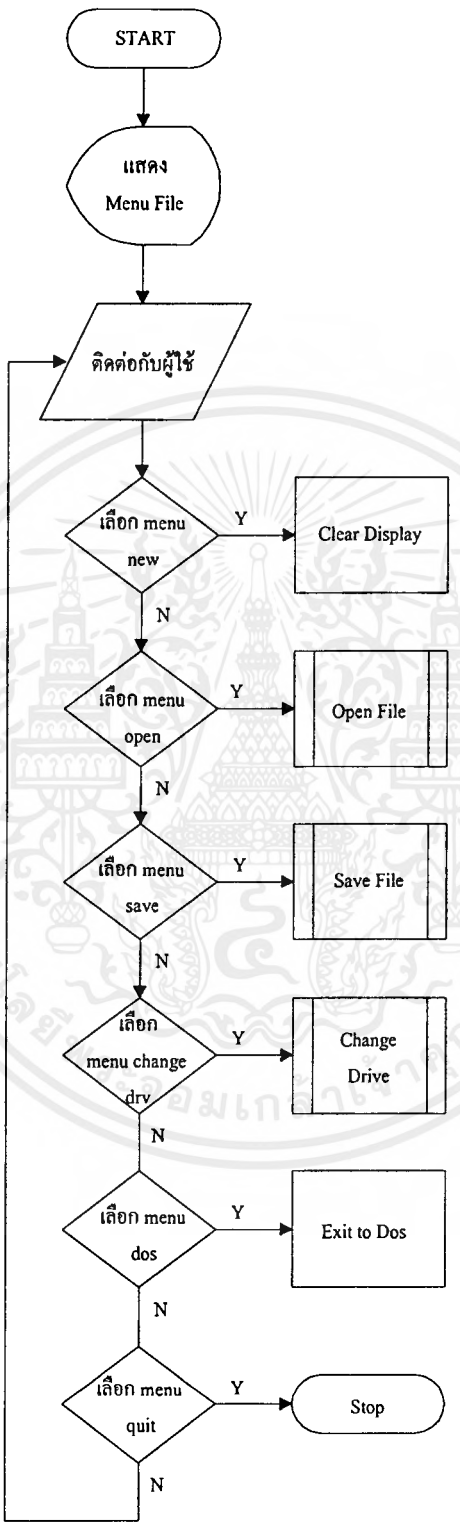
ส่วนของเมนู Print จะมีเมนูย่อย 2 ส่วนคือ Locate และ Circuit

ส่วนของเมนู Help จะมีเมนูย่อย 2 ส่วนคือ About และ Help

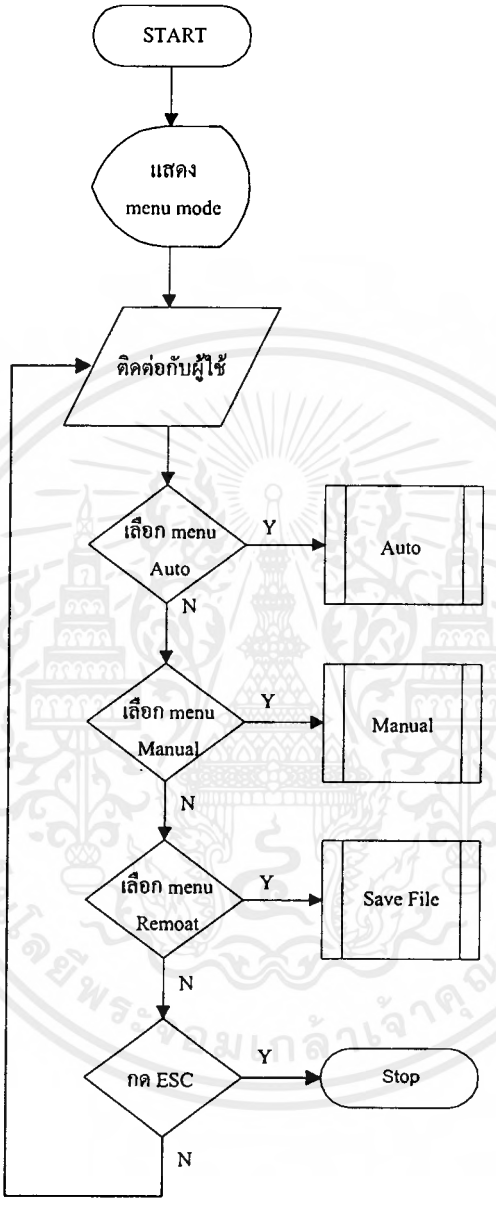
ในการเขียนโปรแกรมจะใช้กราฟฟิกส์โหมดจึงต้องมีการเปิดโหมดกราฟฟิกส์เพื่อใช้งานก่อน โดยใช้คำสั่งการเปิดโหมดกราฟฟิกส์ของภาษาซี คือ ฟังก์ชั่น initgraph แล้วเขียนรูปแบบของ Background ดังแสดงในรูปที่ 1 ในส่วนของเมนูที่ผู้ใช้ติดต่อใช้งาน จะมี 7 ส่วนหลัก ในด้านแนวนอน ซึ่งได้กล่าวในตอนต้น ส่วนเมนูย่อยของแต่ละเมนูหลักจะมีไม่เท่ากัน ซึ่งในการเขียนโปรแกรม ให้ไปทำงานตามเมนู ที่เราเลือก เราจะต้องดูจากเมนูหลักทางแกนนอนก่อน เราจึงนำเมนูย่อยที่เลือกทางแกนตั้งพิจารณา



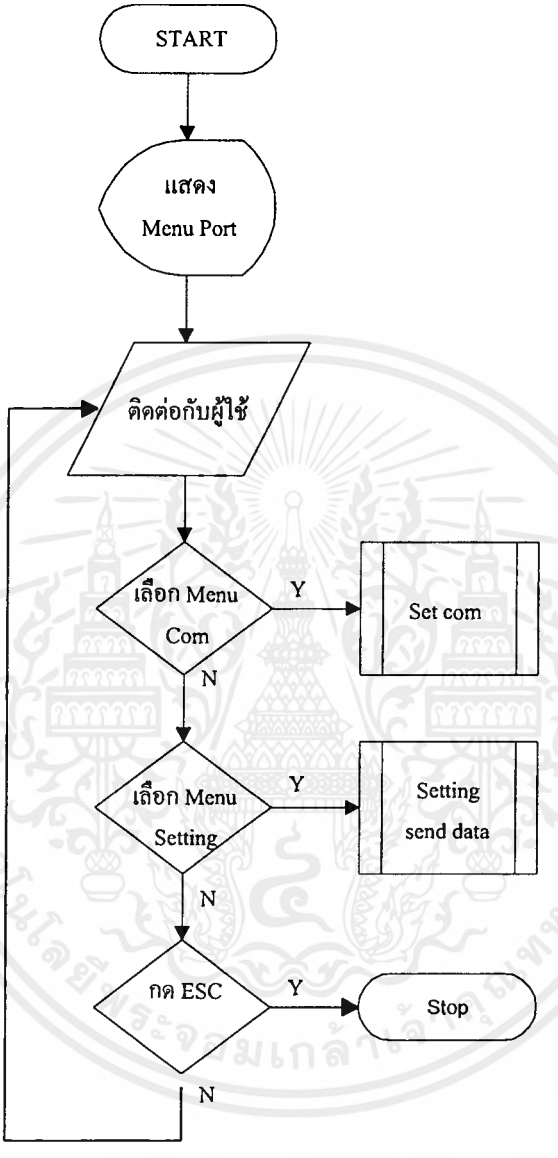
รูปที่ 3.2 ผังงานของโปรแกรมในส่วนของเมนู



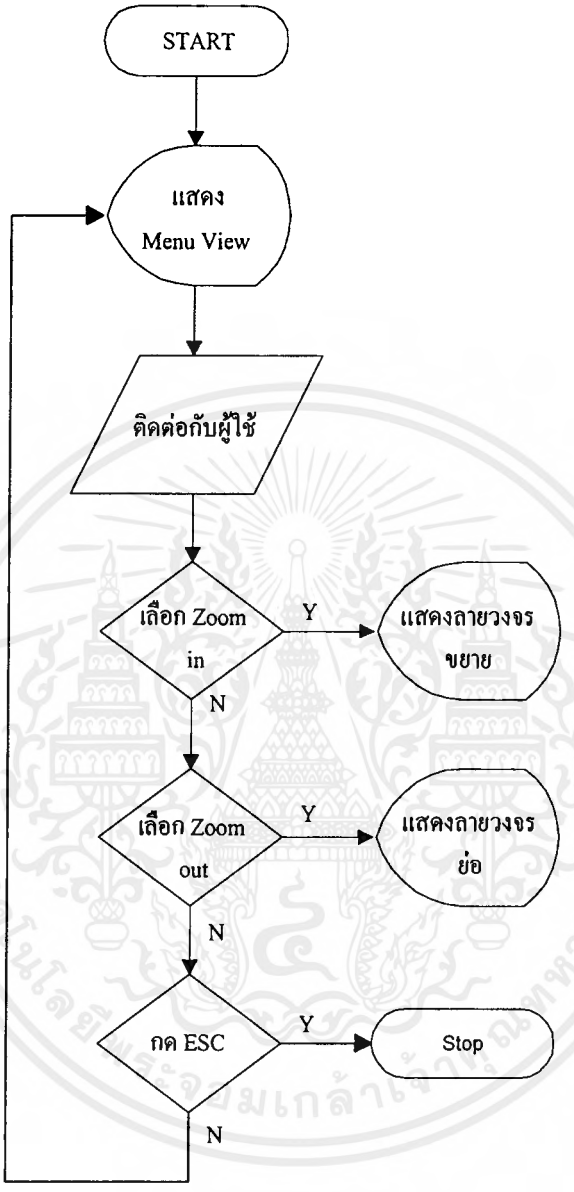
รูปที่ 3.3 ผังงานของโปรแกรมในส่วนของ เมนู File



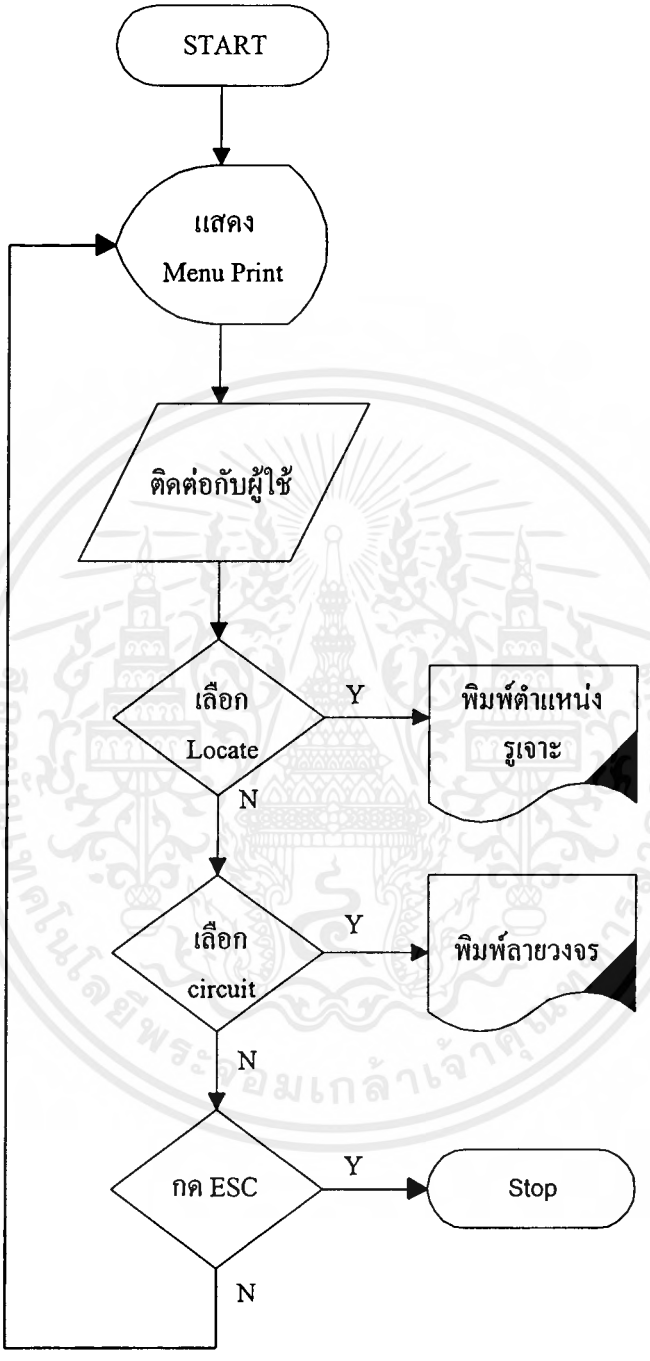
รูปที่ 3.4 ผังงานของโปรแกรมในส่วนของ เมนู Mode



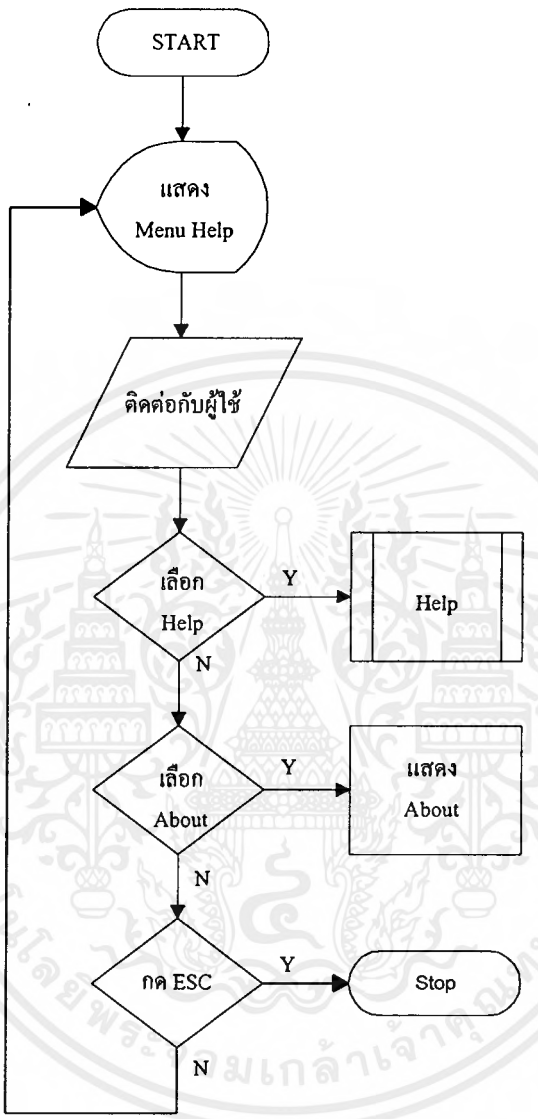
รูปที่ 3.5 ผังงานของโปรแกรมในส่วนของเมนูพอร์ต



รูปที่ 3.6 ผังงานของโปรแกรมในส่วนของเมนูView



รูปที่ 3.7 ผังงานของโปรแกรมในส่วนของเมนูPrint



รูปที่ 3.8 ผังงานของโปรแกรมในส่วนของ เมนูHelp

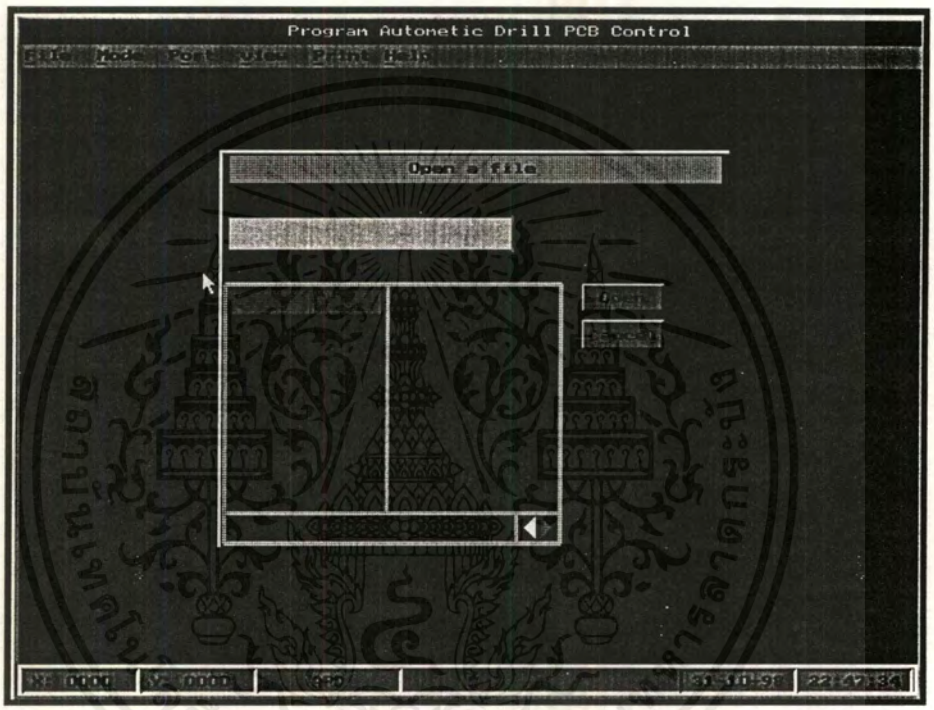
3.2 ส่วนของการติดต่อกับไฟล์

ในส่วนของการติดต่อกับไฟล์ จะแบ่งเป็น 3 ส่วนด้วยกัน คือ

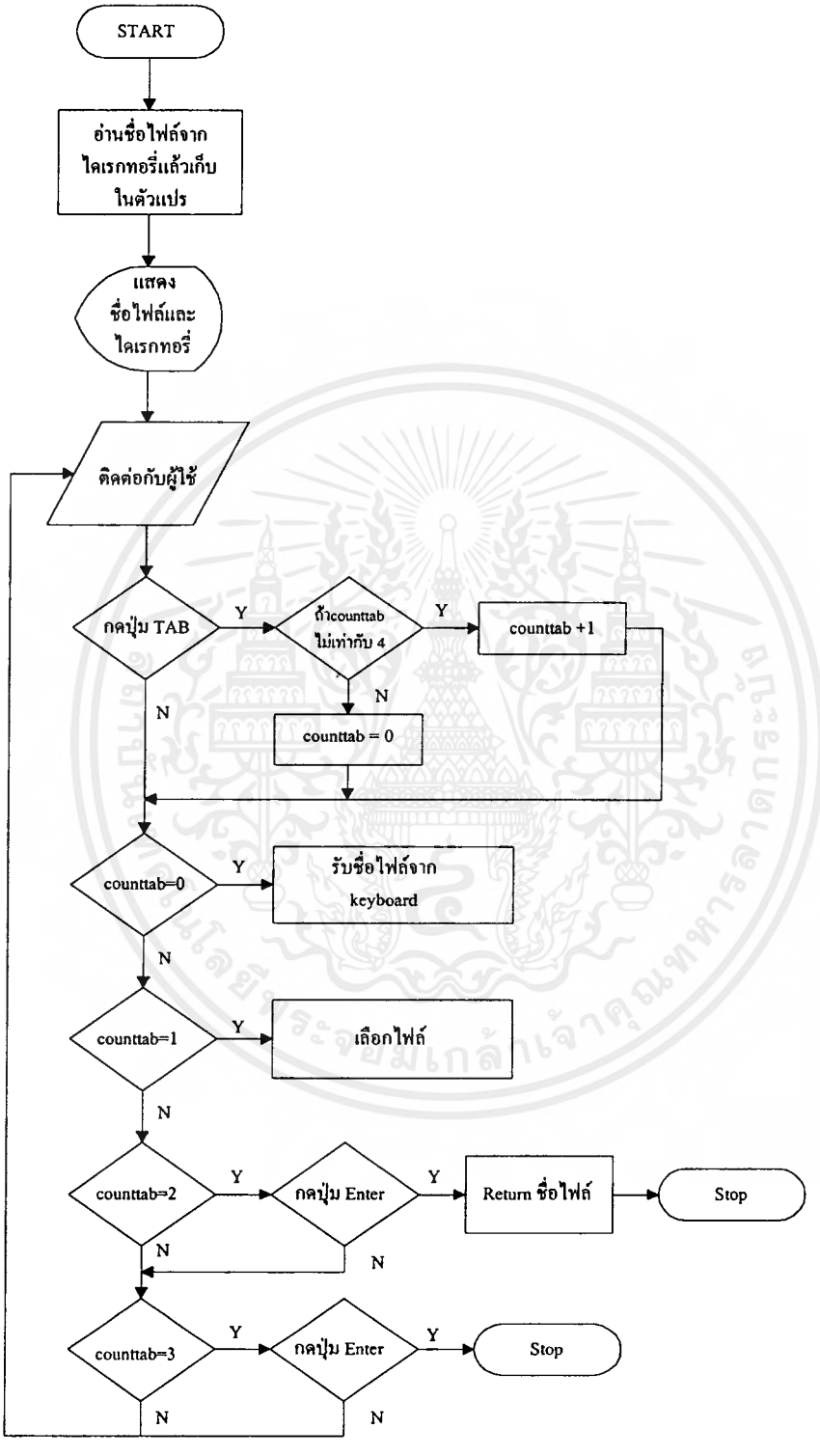
1. ส่วนของการเลือกไฟล์ที่ต้องการจะอ่านข้อมูล
2. ส่วนของการอ่านข้อมูลแล้วเก็บข้อมูล
3. ส่วนของการเปลี่ยนไดรฟ์ ที่ติดต่อใช้งาน

3.2.1 ส่วนของการเลือกไฟล์ที่ต้องการจะอ่านข้อมูล

ส่วนของการเลือกไฟล์จะเป็นส่วนที่ให้ผู้ใช้งานได้เลือกไฟล์ที่ต้องการจะนำข้อมูลนั้นใช้งาน ซึ่งส่วนของการเลือกไฟล์จะแบ่งเป็น 2 ส่วน คือ เลือกไฟล์โดยรับชื่อไฟล์จากคีย์บอร์ด และเลือกไฟล์โดยใช้คีย์ลูกศรเลือกชื่อไฟล์ดังแสดงดังรูปที่ 3.9



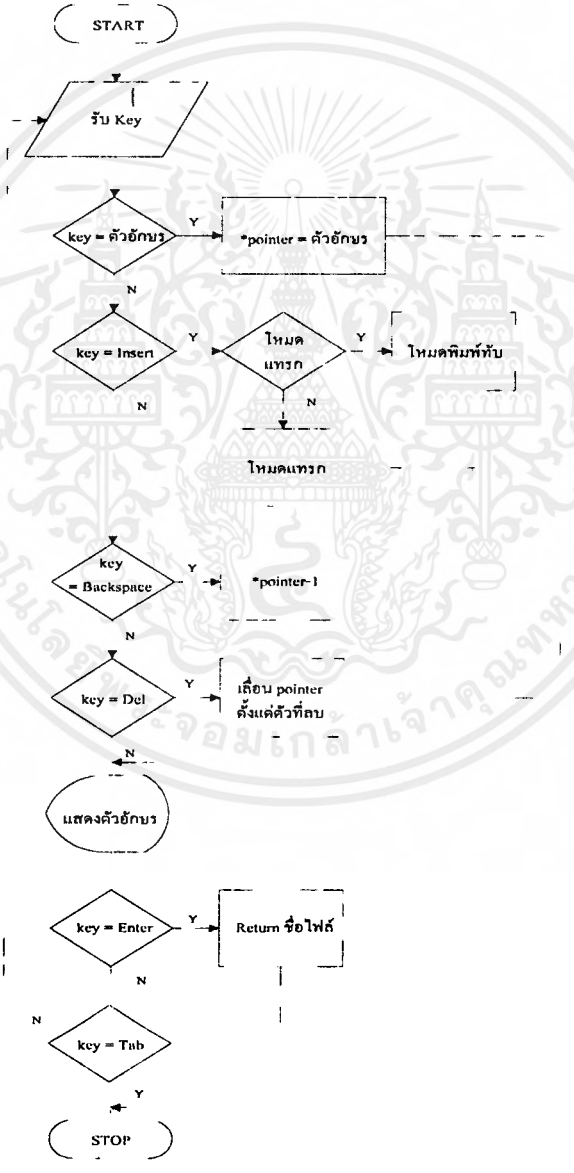
รูปที่ 3.9 ส่วนของการเปิดไฟล์



รูปที่ 3.10 ผังงานของโปรแกรม ส่วนของการเลือกไฟล์ที่ต้องการจะอ่านข้อมูล

1. เลือกไฟล์โดยรับชื่อไฟล์จากคีย์บอร์ด

จะเป็น Editor 1 บรรทัด เพื่อรับตัวอักษรที่รับจากคีย์บอร์ด เมื่อมีการพิมพ์ชื่อไฟล์แล้วกด Enter ก็จะเป็นการเลือกไฟล์นั้นขึ้นมาอ่านในการเขียนโปรแกรมจะต้องรับคีย์ครั้งละ 1 ตัวอักษร ถ้าตัวอักษรที่ได้รับเป็นตัวอักขระก็จะต้องมีการเก็บตัวอักขระที่ได้นั้นเป็นอาร์เรย์เพื่อเก็บเป็นชื่อไฟล์ แต่ถ้าตัวอักษรที่ได้เป็น คีย์ต่างๆ ได้แก่ Insert Backspace Spacebar หรือ Del ก็จะต้องทำตามคีย์ที่ได้รับ ซึ่งสามารถเขียนเป็น ผังงานของโปรแกรม ได้ดังข้างต้น



รูปที่ 3.11 ผังงานของโปรแกรมเลือกไฟล์โดยรับชื่อไฟล์จากคีย์บอร์ด

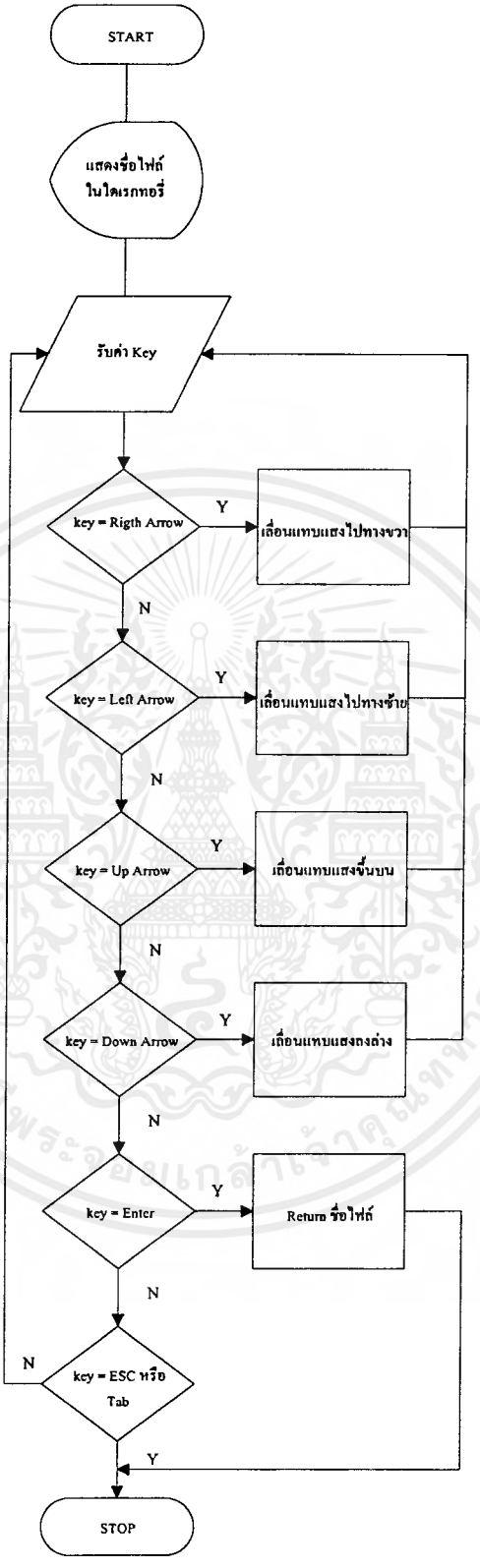
2. เลือกไฟล์โดยใช้คีย์ลูกศร

จะเป็นส่วนที่ทำหน้าที่อ่านชื่อไฟล์หรือไดเรกทอรี่ปัจจุบัน เก็บชื่อไฟล์และสถานะต่างๆ ไว้ แล้วแสดงชื่อไฟล์และไดเรกทอรี่ที่หน้าจอโดยเราสามารถเลือกไฟล์ที่ใช้งานได้ ซึ่งในการเขียนโปรแกรมเพื่ออ่านชื่อไฟล์และสถานะนำมาเก็บในตัวแปร จะมีปัญหาคือ จำนวนของไฟล์ในแต่ละไดเรกทอรี่มีจำนวนไม่เท่ากัน บางไดเรกทอรี่มีไฟล์อยู่หลายไฟล์ บางไดเรกทอรี่ก็มีไฟล์อยู่น้อย ดังนั้นในการจะใช้ตัวแปรเพื่อนำมาเก็บชื่อไฟล์และไดเรกทอรี่ เราจึงใช้วิธีการอ้างหน่วยความจำแบบไดนามิก (Dynamic Memory) คือถ้ามีไฟล์จึงจะกระทำการจองหน่วยความจำเพื่อเก็บชื่อไฟล์และสถานะต่างๆ ไว้ในตัวแปร สำหรับการอ้างหน่วยความจำแบบไดนามิกมีหลายวิธี แต่ในที่นี้จะใช้วิธีการจองหน่วยความจำและอ้างถึงแบบ Link List ซึ่งลักษณะของการอ้างหน่วยความจำแบบ Link List เป็นดังนี้คือ

ตัวแปรที่ใช้ในการเก็บข้อมูลเราจะกำหนดให้เป็นแบบโครงสร้าง (Structor) ภายในข้อมูลแบบโครงสร้างก็จะมีข้อมูลที่ใช้เก็บชื่อไฟล์และสถานะของไฟล์ และมีข้อมูลที่ใช้เก็บตำแหน่งข้อมูลถัดไปดังรูปข้างต้น ในแต่ละบล็อกข้อมูลซึ่งเราจะเรียกว่าโหนด (Node) จะมีการเชื่อมโยงกัน โดยในข้อมูลสุดท้าย จะให้ตำแหน่งข้อมูลถัดไปเป็น NULL

ในการออกแบบและเขียนโปรแกรม เมื่อมีการอ่านชื่อไฟล์ เราจะต้องเริ่มจองหน่วยความจำแบบโครงสร้างขึ้นมาก่อน จากนั้นจึงนำข้อมูลซึ่งได้แก่ชื่อไฟล์และสถานะต่าง ๆ มาเก็บไว้ในตัวแปรนั้น แล้วจึงกำหนด ตำแหน่งของข้อมูลถัดไปให้มีค่าเป็น NULL และเมื่อมีการค้นหาชื่อไฟล์พบอีก เราก็ต้องมีการเก็บข้อมูลของไฟล์นั้นต่อ โดยเราจะต้องกระทำการจองหน่วยความจำก่อน จากนั้นจึงเปลี่ยนตำแหน่งของข้อมูลถัดไปของโหนดก่อนหน้านี้ โดยเปลี่ยนให้เป็นตำแหน่งหน่วยความจำของข้อมูลถัดไปที่ได้จองหน่วยความจำไว้ จากนั้นจึงกำหนดตำแหน่งของข้อมูลถัดไปของโหนดใหม่ให้เป็น NULL ทำเช่นนี้ไปเรื่อย ๆ จนกว่าจะไม่พบไฟล์อีกแล้ว

และเมื่อเลิกใช้งานข้อมูลที่ได้จองหน่วยความจำไว้แล้ว ให้ทำการคืนหน่วยความจำทุกครั้ง เพื่อไม่ให้หน่วยความจำเต็ม ซึ่งจะทำให้โปรแกรมเกิดปัญหาได้



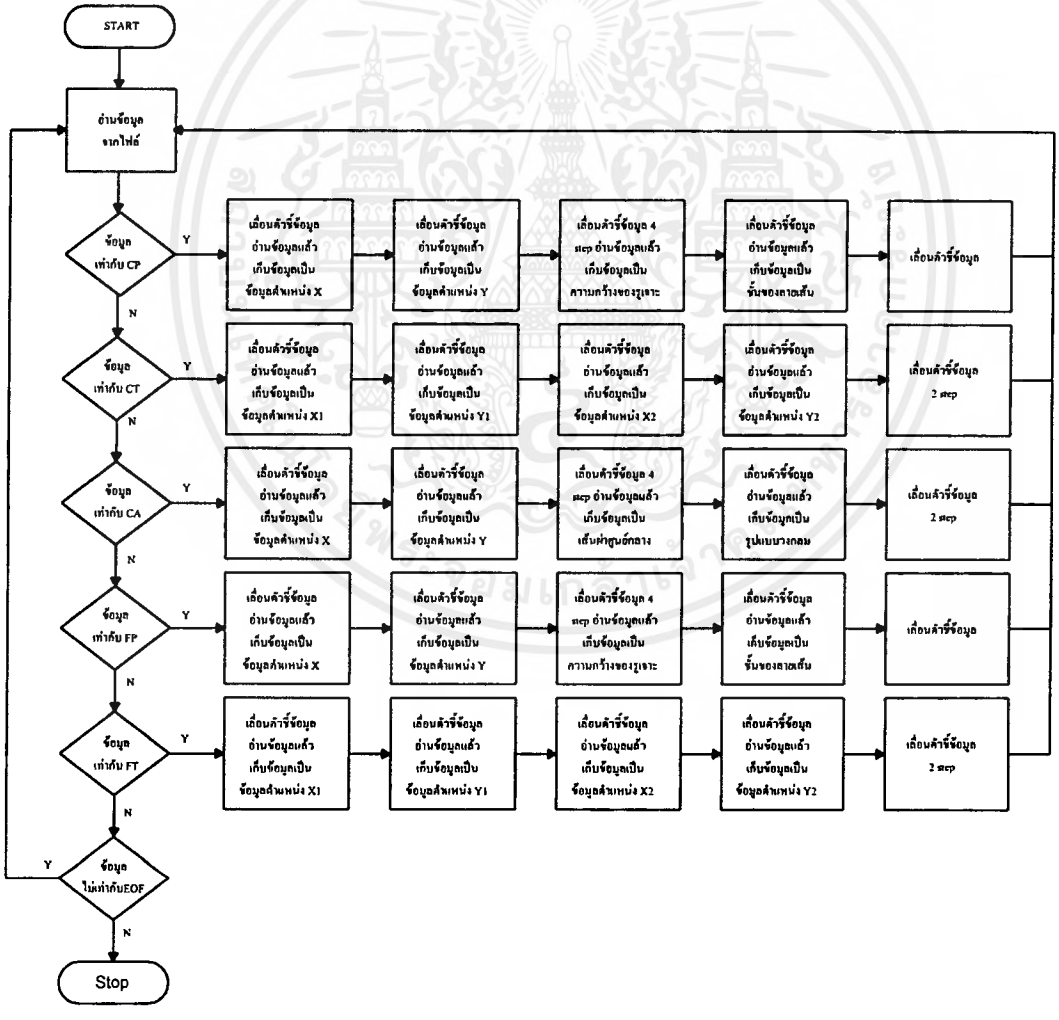
รูปที่ 3.12 ผังงานของโปรแกรม ของการเลือกไฟล์โดยใช้คีย์ลูกศร

3.2.2 ส่วนของการอ่านข้อมูลในไฟล์แล้วเก็บข้อมูล

จะอ่านข้อมูลจากไฟล์ที่เลือกแล้วนำข้อมูลที่ได้เก็บไว้ในตัวแปรเพื่อนำไปแสดงผลหรือส่งไปยังเครื่องเจาะ ไฟล์ข้อมูลที่ติดต่อใช้งานมี 2 รูปแบบ คือ

1. Format file pcb จะเป็นรูปแบบของไฟล์ที่จะจัดเก็บข้อมูลของลักษณะการวางอุปกรณ์และลักษณะของลายวงจร

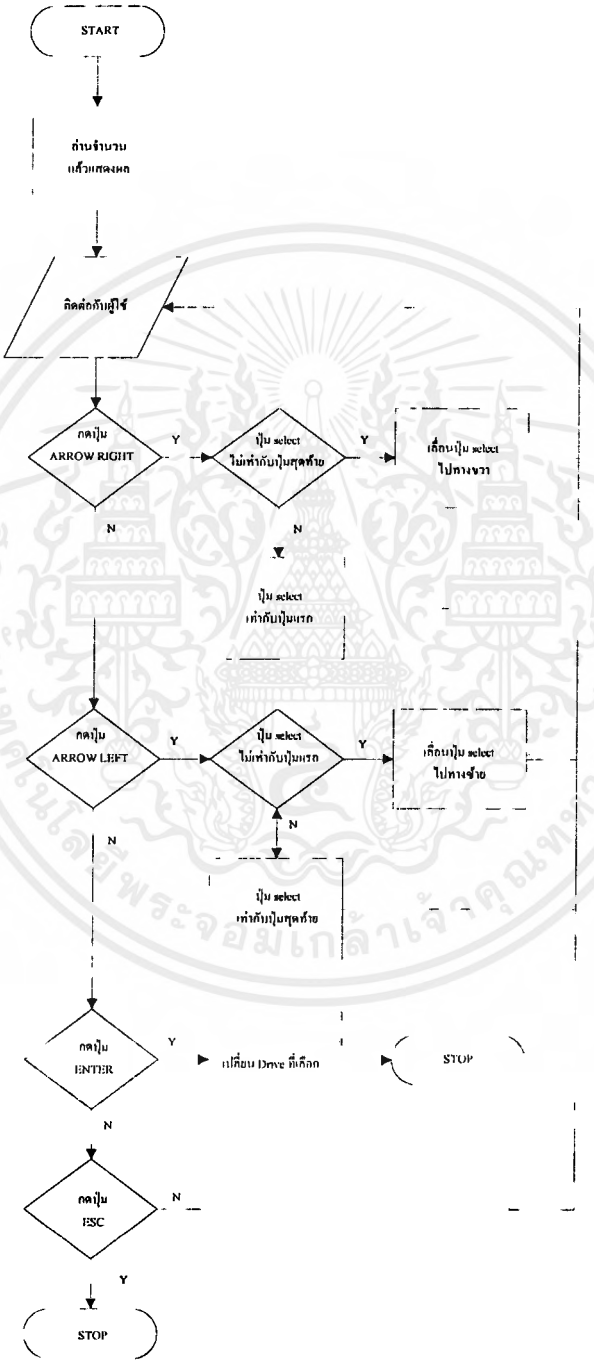
2. Format file drl จะเป็นรูปแบบของไฟล์ที่จะจัดเก็บข้อมูลของตำแหน่งของรูเจาะซึ่งในการอ่านข้อมูล ถ้าเป็นไฟล์ที่ drl โปรแกรมจะแสดงถึงตำแหน่งของรูเจาะทางจอภาพ แต่ถ้าเป็นไฟล์ pcb จะแสดงถึงการวางอุปกรณ์และแสดงลักษณะของลายวงจร



รูปที่ 3.13 ผังงานของโปรแกรม ของส่วนของการอ่านข้อมูลในไฟล์

3.2.3ส่วนของการเปลี่ยนไดรฟ์

เป็นส่วนที่ใช้เลือกช่องเก็บข้อมูลที่ต้องการติดต่อ



รูปที่ 3.14 ผังงานของโปรแกรมของส่วนของการเปลี่ยนไดรฟ์

3.3 ส่วนของการแสดงลายวงจร

เป็นส่วนที่ใช้แสดงลายวงจรซึ่งเขียนโดยโปรแกรมโปรเทลแสดงที่หน้าจอแสดงผล ซึ่งข้อมูลจะได้จากการอ่านข้อมูลในไฟล์ที่เป็น Format ไฟล์ PCB ซึ่งลายละเอียดส่วนของไฟล์ข้อมูลที่ได้ทำการศึกษาและนำมาใช้ในโปรแกรมมีดังนี้

PCB FILE 4 <---------- บอกว่าเป็นจุดเริ่มต้นของไฟล์ข้อมูล

COMP <---------- บอกว่าเป็นการเริ่มรายละเอียดของตัวอุปกรณ์

R4 <---------- ชื่อตัวอุปกรณ์

AXIAL0.4 <---------- ชนิดของตัวอุปกรณ์

7769 27866 60 0 10 7 <---------- ส่วนนี้ไม่ได้นำมาใช้ในโปรแกรม

7769 27896 60 0 10 7 <---------- ส่วนนี้ไม่ได้นำมาใช้ในโปรแกรม

7800 27800 1 1 2 <---------- ส่วนนี้ไม่ได้นำมาใช้ในโปรแกรม

CP <---------- บอกรายละเอียดในการวางตำแหน่งขาของอุปกรณ์

7800 27800 62 62 1 28 1 13

- 13 — กำหนดการแสดงรูเจาะ ถ้าเป็นเลขอื่นจะไม่แสดงจุดบัดกรี
- 1 — ชั้น(layer) ของลายเส้น
- 28 — ขนาดของรูเจาะ
- 1 — รูปร่างของจุดบัดกรีชั้น
- 62 — ความกว้างของจุดบัดกรี ขยายทางแกนตั้ง
- 62 — ความกว้างของจุดบัดกรี ขยายทางแกนนอน
- 27800 — ตำแหน่งของรูเจาะทางแกนตั้ง
- 7800 — ตำแหน่งของรูเจาะทางแกนนอน

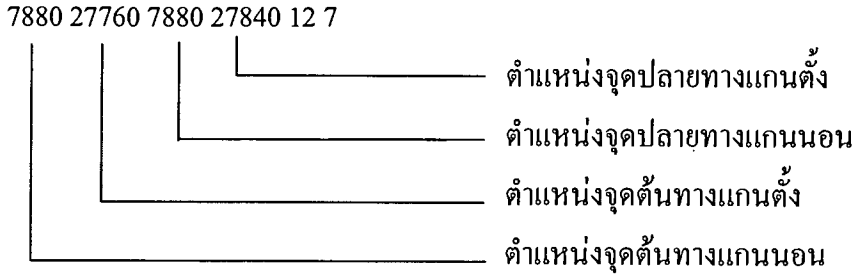
1 <---------- ลำดับที่ของขาอุปกรณ์

CP

8200 27800 62 62 1 28 1 13

2

CT <---------- บอกรายละเอียดเกี่ยวกับลักษณะรูปอุปกรณ์ที่เป็นเส้นตรง



CT

7880 27840 8120 27840 12 7

CT

8120 27760 8120 27840 12 7

CT

7880 27760 8120 27760 12 7

CT

7800 27800 7880 27800 12 7

CT

8120 27800 8200 27800 12 7

ENDCOMP

COMP

COMP

C1

RB.2/.4

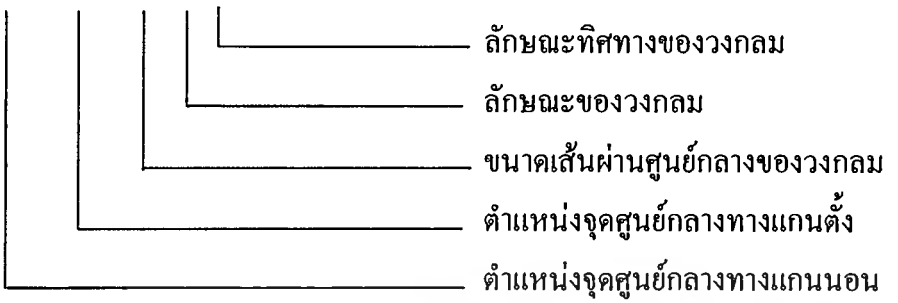
8650 28090 60 1 10 7

8620 28090 60 1 10 7

8880 28400 1 1 2

CA ←---------- บอกรายละเอียดเกี่ยวกับลักษณะรูปอุปกรณ์ที่เป็นรูปวงกลม

8880 28300 200 15 10 7



CP

8880 28400 62 62 1 28 1 13

1

CP

8880 28200 62 62 1 28 1 13

2

CT

8880 28500 8880 28600 12 7

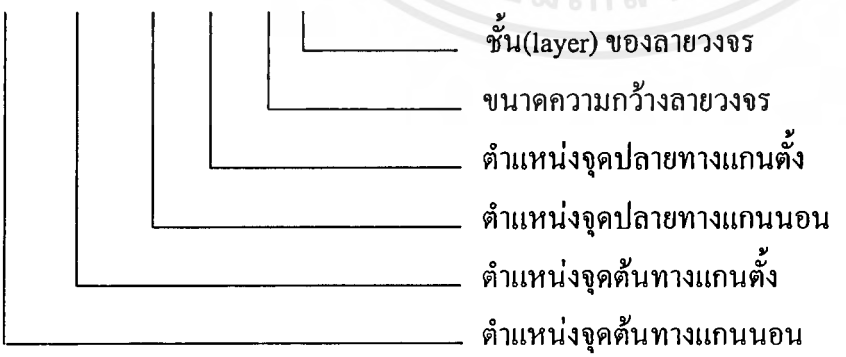
CT

8930 28550 8830 28550 12 7

ENDCOMP

FT <---------- บอกตำแหน่งลายวงจร

8200 28400 8880 28400 20 1



FT

7200 28400 7800 28400 20 1

FP

9600 28200 100 100 1 40 1 13

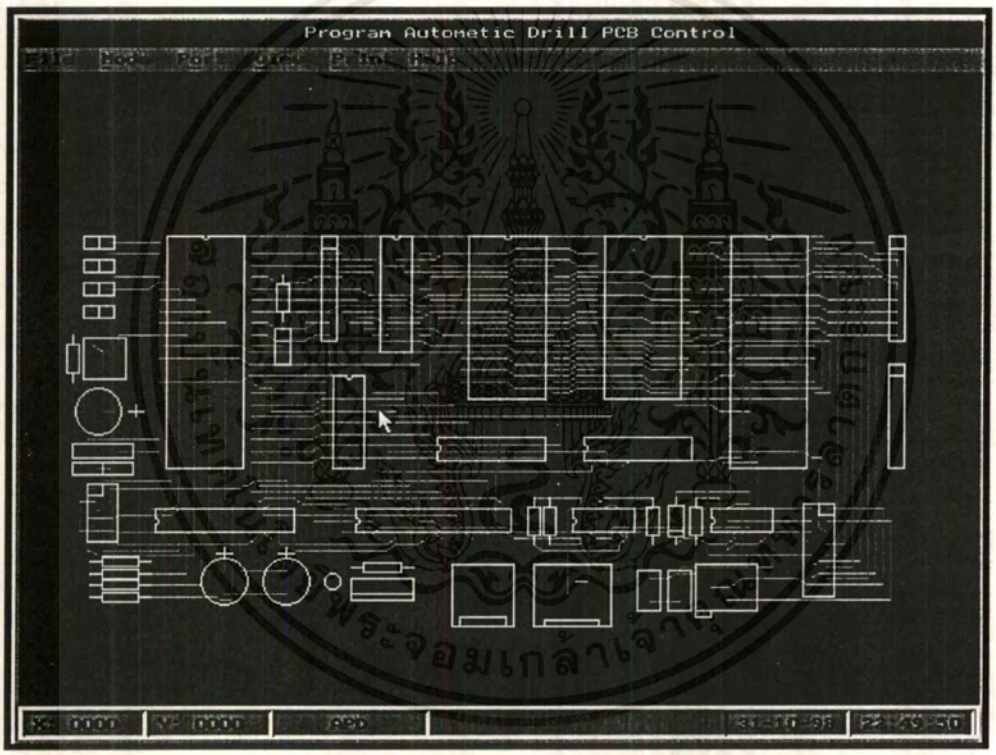
0

FP

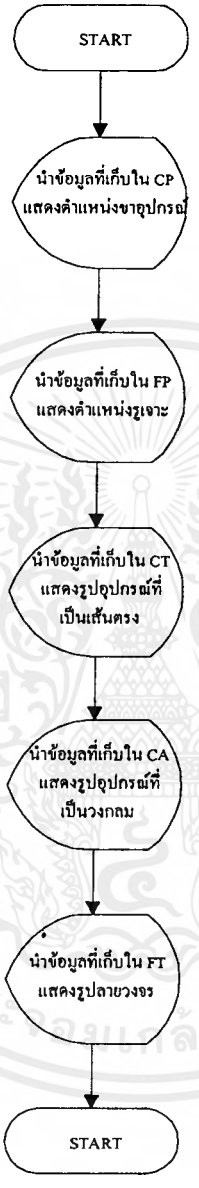
9600 28400 100 100 1 40 1 13

0

ENDPCB



รูปที่ 3.15 ลักษณะของลายวงจรบนโปรแกรม



รูปที่ 3.16 ผังงานของโปรแกรมแสดงลายวงจร

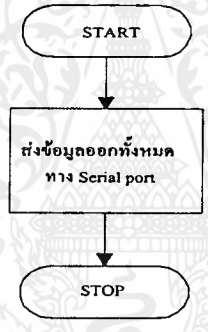
3.4 ส่วนของการรับส่งข้อมูล

เป็นส่วนที่จะต้องนำข้อมูลส่งไปยังเครื่องเจาะแผ่นวงจรพิมพ์ โดยการส่งข้อมูลจะส่งข้อมูลของตำแหน่งของรูเจาะออกทางพอร์ตอนุกรมโดยจะติดต่อกับกับพอร์ตโดยตรงคือถ้าต้องการติดต่อกับ

พอร์ต Com 1 จะต้องส่งข้อมูลออกพอร์ตหมายเลย 3F8 และถ้าติดต่อกับพอร์ต Com 2 จะต้องส่งข้อมูลออกพอร์ต หมายเลย 2F8 โดยเราสามารถเซตสถานะของสายสื่อสารได้ โดยการกำหนดที่รีจิสเตอร์ควบคุมสายสื่อสาร ซึ่งในโปรแกรมสามารถเลือกพอร์ตการติดต่อ และสามารถกำหนดสถานะการรับส่งข้อมูลได้ สำหรับ โหมด การทำงานในการรับส่งข้อมูลของ โปรแกรมกับเครื่องเป็น 3 ลักษณะคือ

3.4.1 การส่งข้อมูลโดยไม่ติดตามผล

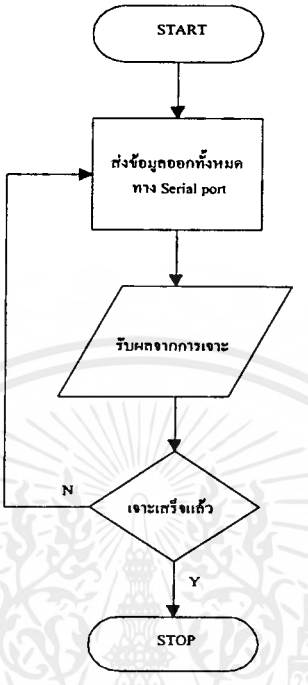
จะเป็นการส่งข้อมูลทั้งหมดไปยังเครื่องเจาะแผ่นวงจรพิมพ์โดยไม่ติดตามผลการทำงานของเครื่อง ซึ่งจะเป็นการส่งข้อมูลของจำนวน พัลซ์ในแต่ละระยะทางการเคลื่อนที่ไปยังเครื่องเจาะ ในการเขียนโปรแกรมเราจึงส่งข้อมูลของตำแหน่งเจาะออกทาง พอร์ต Serial ทั้งหมดทีเดียว จากนั้นก็จบโปรแกรมเลยโดยไม่ติดตามผลการทำงานของเครื่องเจาะ



รูปที่ 3.17 ผังงานของโปรแกรมของการส่งข้อมูลโดยไม่ติดตามผล

3.4.2 การส่งข้อมูลแล้วติดตามผลการทำงานของเครื่อง

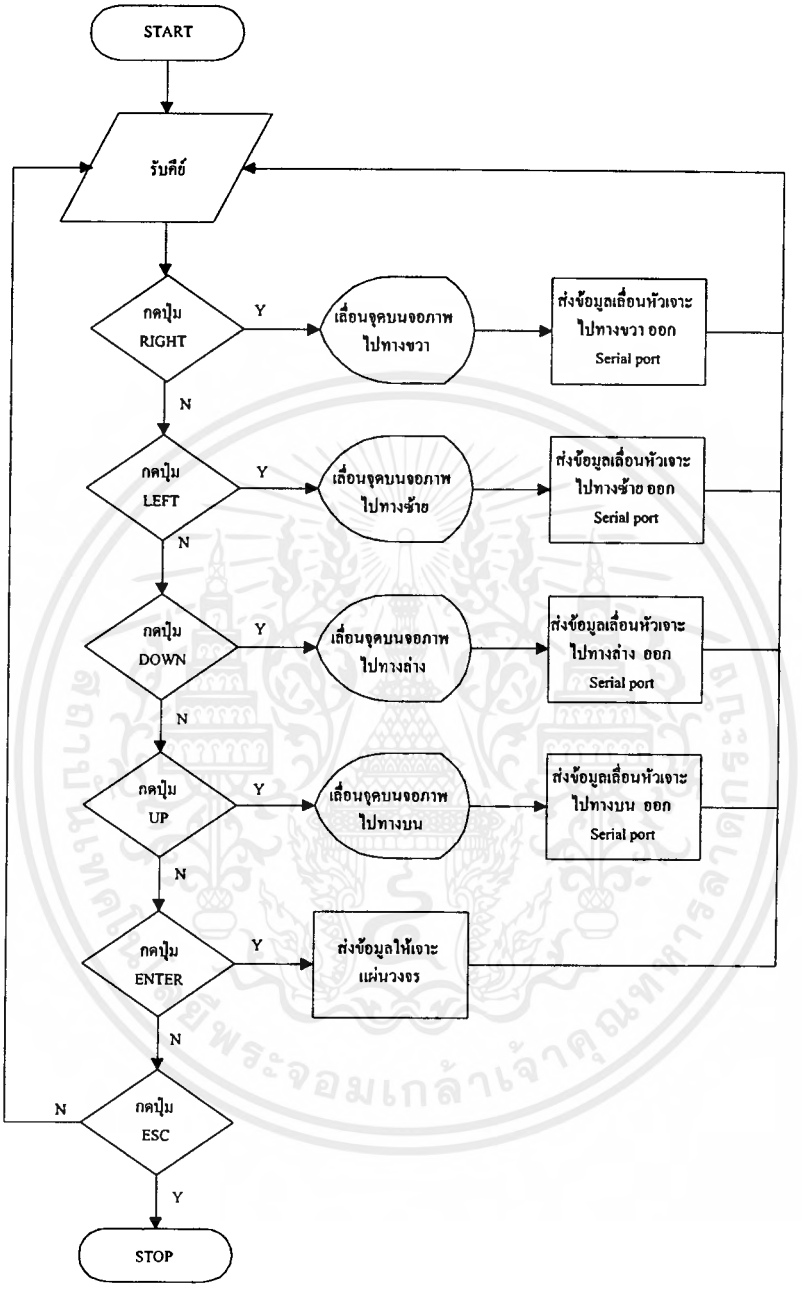
จะเป็นการส่งข้อมูลที่เป็นจำนวนพัลซ์ไปยังเครื่องเจาะแผ่นวงจรพิมพ์แล้วติดตามผลการทำงานของเครื่องเจาะแผ่นวงจรพิมพ์อยู่ตลอด ในการออกแบบเราจะส่งข้อมูลของตำแหน่งรูเจาะออกไปยังเครื่องทีละรู เมื่อเครื่องเจาะได้รับข้อมูลแล้วนำข้อมูลนั้นเลื่อนตำแหน่งไปยังตำแหน่งรูเจาะถ้าเจาะสำเร็จ ก็ต้องส่งค่ากลับให้โปรแกรมทราบจากนั้นโปรแกรมจึงส่งข้อมูลตำแหน่งถัดไปออกไป



รูปที่ 3.18 ผังงานของโปรแกรม ของการส่งข้อมูลแล้วติดตามผลการทำงานของเครื่อง

3.4.3 การควบคุมหัวเจาะของเครื่องเจาะแผ่นวงจรพิมพ์โดยคีย์บอร์ดของเครื่องคอมพิวเตอร์

ซึ่งในโปรแกรมจะแสดงถึงตำแหน่งของหัวเจาะที่เคลื่อนที่ ว่าเคลื่อนที่เป็นระยะทางเท่าใด ในการออกแบบเราจะส่งข้อมูลเลื่อนตำแหน่งของหัวเจาะก็ต่อเมื่อมีการกด คีย์ ลูกศร โดยตำแหน่งของหัวเจาะจะเลื่อนไปตามทิศทางของคีย์ลูกศร



รูปที่ 3.19 ผังงานของโปรแกรม ของการควบคุมหัวเจาะของเครื่องเจาะแผ่นวงจรพิมพ์โดย คีย์บอร์ด ของเครื่องคอมพิวเตอร์

3.5 ส่วนของการติดต่อกับเครื่องพิมพ์

จะแบ่งการพิมพ์ออกเป็น 2 โหมดการพิมพ์ คือ

1. โหมดตัวอักษรจะเป็นการพิมพ์ข้อมูลในโหมดของตัวอักษร

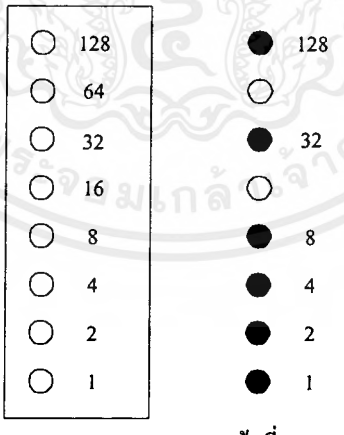
2. กราฟฟิกส์โหมดจะเป็นการพิมพ์ข้อมูลในโหมดของรูปภาพ

3.5.1 แบบโหมดของตัวอักษร

เราสามารถสั่งพิมพ์ได้โดยตรงใช้คำสั่ง Printf(STDPRN,ข้อมูลตัวอักษร) เราจึงสามารถสั่งพิมพ์ได้โดยการส่งข้อมูลตัวอักษรออกไป

3.5.2 แบบ กราฟฟิกส์โหมด

เราจะต้องทราบลักษณะของเครื่องพิมพ์ว่าเป็นเครื่องพิมพ์แบบใด ซึ่งในโปรแกรมในที่นี้ จะเป็นการพิมพ์โดยใช้เครื่องพิมพ์ ประเภทหัวเข็ม ซึ่งจะเป็นการพิมพ์แบบ 9 หัวเข็ม ในเครื่องพิมพ์แบบ 24 หัวเข็ม สามารถทำงานเหมือนเครื่องพิมพ์แบบ 9 เข็ม ซึ่งในการสั่งงานของหัวเข็มเราจะใช้ 8 เข็ม ในการสร้างจุด ซึ่งเข็มแต่ละเล่มจะมีค่าประจำเข็มที่แน่นอนอยู่ เข็มล่างสุดมีค่าเป็น 1 เข็มถัดขึ้นมามีค่าประจำเข็มเป็น 2 เข็มถัดมาเป็น 2,8,16,32,64 และเข็มบนสุดมีค่าเป็น 128 ดังภาพ



รูปที่ 3.20 ค่าประจำตำแหน่งหัวเข็มของเครื่องพิมพ์

สมมุติถ้าต้องการให้เข็มในแนวที่ 1 ทำงาน ค่าที่ต้องส่งให้เครื่องพิมพ์คือ 1+2+4+8+32+128 = 175 ถ้าต้องการให้เข็มทุกเข็ม พิมพ์ข้อมูลจั้นค่าที่ต้องส่งให้เครื่องพิมพ์คือ

1+2+4+8+16+32+64+128 = 255

สรุปแล้ว เราสามารถกำหนดค่าของเข็มได้ทั้งหมดรวม 256 ค่าคือ ค่า 0 ไปจนถึง ค่า 255

ในภาวะปกติเครื่องพิมพ์จะทำงานในภาวะโหมดตัวอักษร หรือพิมพ์ข้อความ การที่จะให้เครื่องพิมพ์ พิมพ์ภาพกราฟฟิกส์ เราจะต้องใช้คำสั่งเปลี่ยนโหมดการพิมพ์ให้เป็นโหมดกราฟฟิกส์โดยคำสั่งนี้มีอยู่ในคู่มือที่ให้มากับเครื่องพิมพ์โดยจะบอกไว้ 3 รูปแบบ

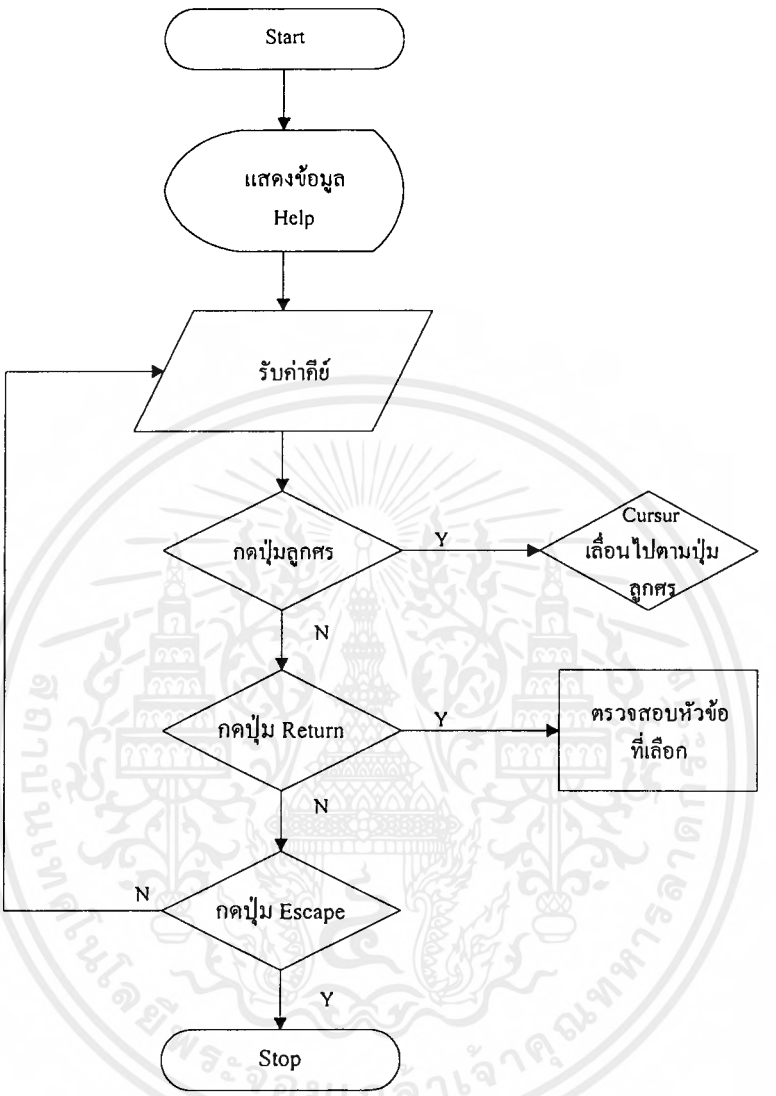
การออกแบบจากโปรแกรมจะสามารถเลือกการพิมพ์เป็น 2 แบบ คือ

1. Locate จะเป็นการพิมพ์ตำแหน่งรูเจาะทั้งหมดออกทางเครื่องพิมพ์ ซึ่งเราสามารถสั่งพิมพ์โดยใช้โหมดตัวอักษรได้เลย

2. Circuit จะเป็นการพิมพ์ลายวงจรที่ได้ทำการโหลดจากไฟล์ PCB ออกทางเครื่องพิมพ์ ซึ่งในการสั่งพิมพ์เราจะต้องสั่งพิมพ์โดยใช้วิธีกราฟฟิกส์โหมด

3.6 ส่วนของการแสดงรายละเอียดการใช้งานโปรแกรมและรายละเอียดของผู้จัดทำ

ในส่วนของการแสดงรายละเอียดการใช้งานโปรแกรม จะเป็นส่วนที่จะบอกถึงการใช้งานโปรแกรม ซึ่งภายในส่วนของการแสดงรายละเอียดการใช้งานโปรแกรม เราสามารถที่จะเลือกหัวข้อที่ต้องการจะดูได้โดยการใช้การเลื่อนเคอร์เซอร์ไปยังหัวข้อนั้น หรือใช้เมาส์คลิก ในการออกแบบเราจะต้องกำหนดหัวข้อที่สามารถดูย่อยได้อีกก่อน แล้วจึงแสดงผลให้ทราบ เมื่อมีการกดคีย์ลูกศร เคอร์เซอร์จะเลื่อนไปตามคีย์นั้น ซึ่งเราจะต้องมีการตรวจสอบตำแหน่งของเคอร์เซอร์อยู่ตลอดว่า ตำแหน่งของเคอร์เซอร์นั้นอยู่ในบริเวณของหัวข้อที่เรากำหนดหรือไม่ ถ้าอยู่เมื่อมีการกดปุ่ม Enter เราจะต้องลบหน้าต่างเดิมออก แล้วแสดงข้อมูลของการใช้งานโปรแกรมถัดไปตามหัวข้อนั้น



รูปที่ 3.21 ผังงานของโปรแกรมในส่วนของการแสดงรายละเอียดการใช้งานโปรแกรม

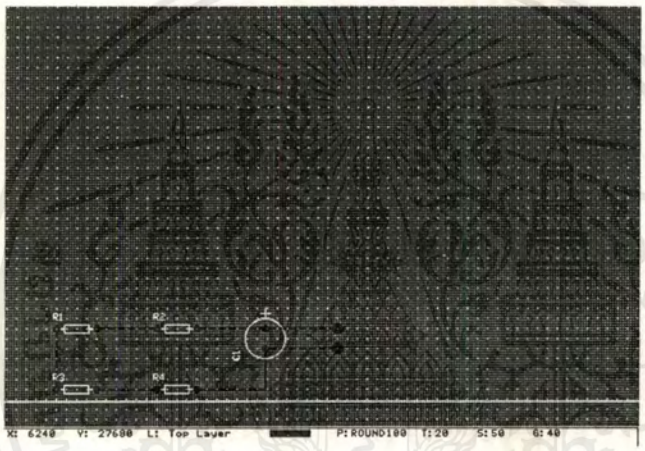
ในส่วนของ About จะเป็นส่วนที่ใช้แสดงข้อมูลของอาจารย์ที่ปรึกษาและข้อมูลของนักศึกษาที่จัดทำโปรแกรม

บทที่ 4

การทดลองและผลการทดลอง

4.1 ทดสอบการทำงานของโปรแกรม

ทำการเปิดโปรแกรมโปรเทลแล้วทำการวาดวงจรจากโปรแกรมโปรเทล (Protel) แล้วบันทึกไฟล์เป็น .PCB ดังรูปที่ 4.1



รูปที่ 4.1 รูปวงจรวาดจากโปรแกรมโปรเทล

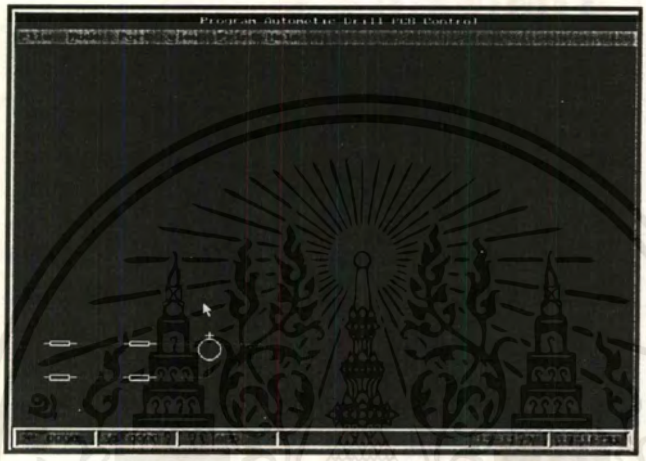
ทำการเปิดโปรแกรมควบคุมเครื่องเจาะแผ่นวงจรพิมพ์จะปรากฏ ชื่อของโปรแกรมที่หน้าจอคอมพิวเตอร์ดังรูปที่ 4.2

รูปที่ 4.2 ชื่อของโปรแกรมควบคุมเครื่องเจาะ

ทำการเปิดไฟล์ TEST1.PCB จากคำสั่ง Open Files

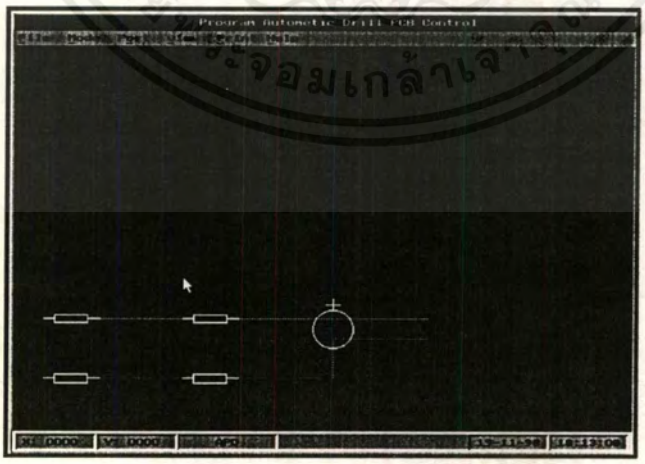
4.2 การทดสอบการอ่านไฟล์ .PCB แล้วแสดงลายวงจร

ทำการเปิดไฟล์ TEST1.PCBดังรูปที่ 4.3



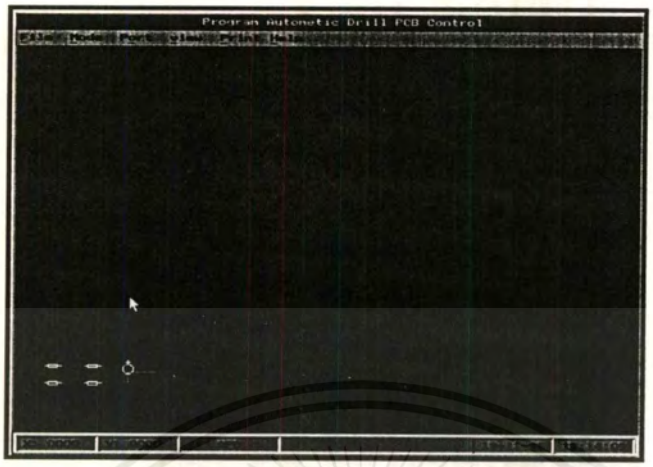
รูปที่ 4.3 รูปวงจรจากไฟล์ TEST1.PCB

ทำการขยายภาพวงจร โดยใช้คำสั่ง ZOOM INดังรูปที่ 4.4



รูปที่ 4.4 การขยายขนาดภาพวงจร

ทำการลดขนาดภาพวงจร โดยใช้คำสั่ง ZOOM OUTดังรูปที่ 4.5



รูปที่ 4.5 การลดขนาดภาพวงจร

ทำการบันทึกไฟล์เป็น .DRL

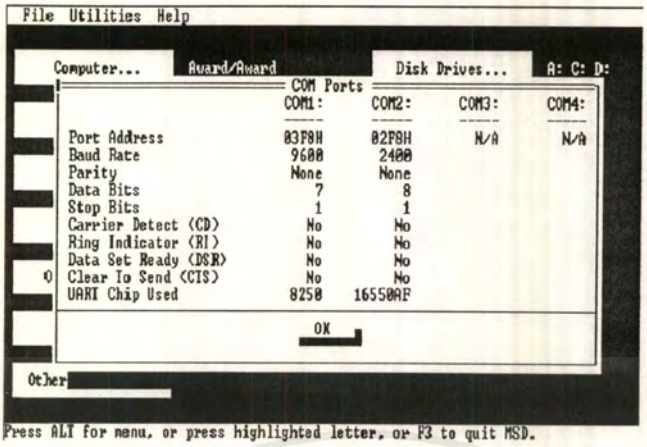
ทำการเปิดไฟล์ TEST1.DRLดังรูปที่ 4.6

รูปที่ 4.6 ตำแหน่งรูเจาะจากไฟล์ TEST1.DRL

4.3 การทดสอบการส่งข้อมูลออกทางเครื่องพิมพ์

ทำการพิมพ์ลายวงจรออกทางเครื่องพิมพ์

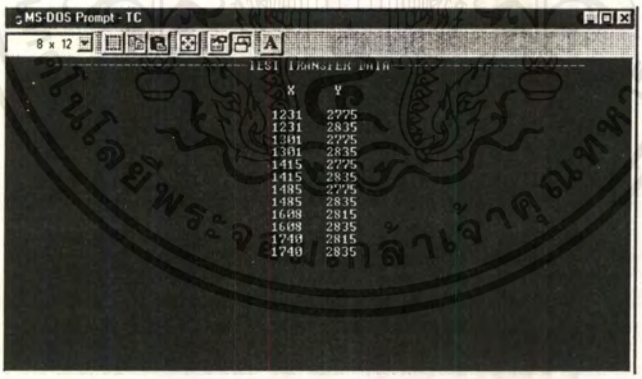
ทำการพิมพ์ตำแหน่งรูเจาะออกทางเครื่องพิมพ์



รูปที่ 4.9 การตรวจเช็คการเซตอัตราความเร็วในการส่งข้อมูล

ในการเลือกใช้ COM2 การเซตอัตราความเร็วในการส่งข้อมูลหรืออื่น ๆ สามารถทำได้ลักษณะเดียวกันกับ COM1

ทำการทดสอบการส่งข้อมูลโดยใช้การติดต่อระหว่างคอมพิวเตอร์ 2 เครื่องโดยผ่านพอร์ตอนุกรมโดยส่งข้อมูลจากไฟล์ TEST1.DRL ไปยังคอมพิวเตอร์อีกเครื่องดังรูปที่ 4.10



รูปที่ 4.10 การทดสอบการส่งข้อมูลผ่านพอร์ตอนุกรม

บทที่ 5
บทสรุป ปัญหา แนวทางแก้ไขและพัฒนา

5.1 บทสรุป

โปรแกรมควบคุมเครื่องเจาะแผ่นวงจรพิมพ์ในปริญานิพนธ์ฉบับนี้สร้างขึ้นเพื่อควบคุมเครื่องเจาะแผ่นวงจรพิมพ์ ซึ่งในการใช้งานจริง ๆ นั้น ต้องอาศัยทั้งทางด้าน ฮาร์ดแวร์และซอฟต์แวร์ ที่มีความสัมพันธ์กัน โดยฮาร์ดแวร์จะเป็นตัวเครื่องเจาะแผ่นวงจรที่สามารถเคลื่อนที่ได้ใน 3 แนว คือ แนวแกน X แนวแกน Y และแนวแกน Z โดยรับข้อมูลในตำแหน่งที่ต้องการเจาะจากซอฟต์แวร์ควบคุมส่งมา ซึ่งภาษาโปรแกรมที่ใช้เขียนโปรแกรมนั้นจะใช้ภาษาซี เนื่องจากต้องการให้โปรแกรมใช้งานได้จริงในโรงงานอุตสาหกรรม ที่มักนิยมใช้งานบนดอสอยู่และใช้เนื้อที่ในการเก็บข้อมูลไม่มากนัก นอกจากนี้ภาษาซียังสะดวกในการเขียนโปรแกรมเพื่อติดต่อกับระบบเนื่องจากสามารถเขียนโปรแกรมรวมกับภาษาแอสแซมบลี้ได้

ในส่วนของโปรแกรมที่ใช้ควบคุมเครื่องเจาะแผ่นวงจรพิมพ์นั้นจะสามารถควบคุมเครื่องได้เป็น 2 ลักษณะคือ ให้สามารถเคลื่อนหัวเจาะแผ่นวงจรพิมพ์ด้วยการกดคีย์บอร์ดและอีกลักษณะหนึ่งคือ นำตำแหน่งของรูเจาะที่ได้ในไฟล์ที่ถูกเขียนโดยโปรเทล เป็นตัวเลื่อนตำแหน่งของหัวเจาะ คือ เราสามารถเขียนลายวงจรโดยโปรแกรมโปรเทลแล้วนำมารเปิดไฟล์ที่ได้แล้วจึงส่งข้อมูลไปยังเครื่องเจาะแผ่นวงจรพิมพ์ เหตุผลที่ใช้ไฟล์ที่เขียนวงจรด้วยโปรเทลก็เพื่อให้เกิดความสะดวกเนื่องจากในการออกแบบลายวงจรนิยมที่จะออกแบบโดยใช้โปรแกรมโปรเทล จากนั้นจึงนำลายวงจรที่ออกแบบไปกัดลายวงจร นำแผ่นวงจรที่กัดลายวงจรแล้วนำมาเจาะรู เพื่อวางอุปกรณ์ เราจึงเขียนโปรแกรมให้อ่านไฟล์จากโปรเทล เพื่อให้สามารถเจาะแผ่นวงจรพิมพ์ได้เลย จึงทำให้สะดวกยิ่งขึ้น

จากการทดลองโปรแกรมควบคุมเครื่องเจาะแผ่นวงจรพิมพ์ ซึ่งก็สามารถอ่านไฟล์โปรเทลและนำลายวงจรมาแสดงที่หน้าจอได้เพื่อให้ได้ทราบว่าเป็นไฟล์ที่ต้องการใช้งานถูกต้องจึงส่งข้อมูลออกไปยังเครื่องเจาะแผ่นวงจรพิมพ์เพื่อให้เครื่องเจาะทำงานต่อไป

แนวทางแก้ไข

สำหรับโปรแกรมที่มีปัญหาเมื่อรันบนวินโดว์ยังไม่สามารถแก้ไขได้ถ้าแก้ไขก็ต้องเปลี่ยนไปเขียนโปรแกรมบนวินโดว์หรือไม่ก็ต้องพยายามไม่ใช้อินเตอร์รัพต์ของดอส

5.3 ประโยชน์ที่ได้รับจากโครงงาน

จากการทำโครงงานในครั้งนี้ได้รับประโยชน์ในด้านต่าง ๆ มากมายไม่ว่าจะเป็นในด้านของการทำงานซึ่งจะมีประโยชน์ต่อไปในอนาคตเมื่อต้องทำงานจริง ประสบการณ์ในการทำงานที่ได้รับผิดชอบ และประโยชน์ที่ได้รับจากการเขียนโปรแกรมซึ่งสามารถแยกเป็นหัวข้อ คือ

1. มีความชำนาญในการคิดและเขียนโปรแกรมมากขึ้น
2. มีความสามารถในการเขียนโปรแกรมภาษาซีเป็นอย่างดี
3. ได้ทราบถึง Format ไฟล์ในรูปแบบอื่น เช่น Format ไฟล์แบบ BMP หรือ Format ไฟล์ของ protel ซึ่งทำให้เราสามารถสร้าง Format อื่นที่ต้องการได้
4. มีความสามารถในการใช้ โปรแกรม Protel For Dos มากขึ้น
5. สามารถเขียนโปรแกรมโดยอาศัยอินเตอร์รัพต์ของดอสได้
6. ได้ทราบถึงฟังก์ชันต่าง ๆ ที่มีมากมายหลายฟังก์ชันของภาษาซีมากขึ้น
7. ได้รู้จำการคิดและแก้ไขปัญหาเมื่อเกิดปัญหาขึ้นได้
8. ได้รู้จักการรับผิดชอบจริงในการทำงานที่ได้รับมอบหมาย
9. ได้รู้จักการทำงานร่วมกับผู้อื่น
10. รู้จักการวางแผนการทำงานอย่างมีระบบ
11. ได้ประสบการณ์ในด้านการบริหารเวลา คือ ถ้าเราขาดความรับผิดชอบก็จะทำให้เกิดการผิดพลาดในการบริหารเวลาได้
12. ได้รู้จักการค้นคว้า หาความรู้และหาประสบการณ์ด้วยตนเองได้

5.4 แนวทางการพัฒนา

1. เนื่องจากโปรแกรมเป็นโปรแกรมมีทำงานบนดอสจึงค่อนข้างล้าสมัยและบางครั้งเมื่อรันโปรแกรมบนวินโดว์แล้วมีปัญหาถ้ามีการพัฒนาต่อไปควรเขียนโปรแกรมเป็นโปรแกรมบนวินโดว์
2. โปรแกรมสามารถอ่านข้อมูลของไฟล์ PCB ที่เขียนโดย Protel For Dos ได้อย่างเดียวจึงควรมีการพัฒนาให้สามารถเขียนโปรแกรมที่อ่านข้อมูลได้ทั้งบนดอส และบนวินโดว์
3. ควรมีการแสดงรายละเอียดของอุปกรณ์เมื่อมีการคลิ๊กเมาส์ที่ตัวอุปกรณ์
4. สามารถที่จะกำหนดตำแหน่งที่ต้องการให้เจาะแผ่นได้โดยการกำหนดจุดบนหน้าจอ

5. เพิ่ม Hotkey ให้กับโปรแกรมพร้อมทั้งมี Help บอกหน้าที่ของ Hotkey

6. เพิ่มการป้องกันและแก้ไขเมื่อโปรแกรมทำงานผิดพลาดให้ครบ

7. สามารถพิมพ์รายงานการเจาะออกทางเครื่องพิมพ์ได้

8. สามารถพิมพ์ตำแหน่งของรูเจาะจริง ๆ ออกทางเครื่องพิมพ์ได้

9. เพิ่มการกำหนดขนาดของรูเจาะเพื่อให้สามารถเปลี่ยนขนาดของดอกสว่านได้

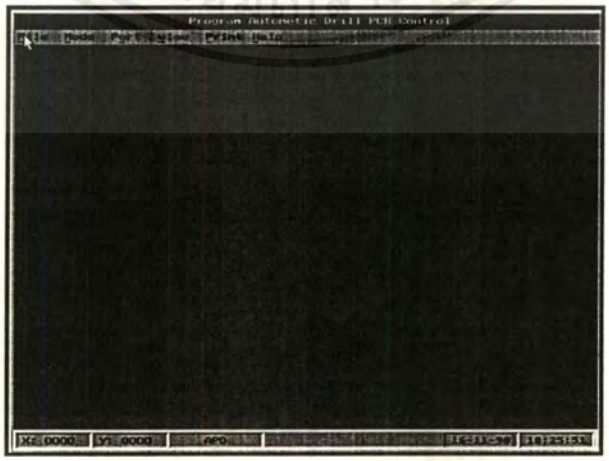
ภาคผนวก ก
คู่มือการใช้งานโปรแกรม

ภาคผนวก ก

โปรแกรมควบคุมเครื่องเจาะแผ่นวงจรพิมพ์อัตโนมัติเป็นโปรแกรมที่ถูกสร้างขึ้นและพัฒนาเพื่อใช้ในการควบคุมเครื่องเจาะแผ่นวงจรพิมพ์อัตโนมัติโดยนักศึกษา เมื่อทำการเปิดโปรแกรมจะเห็นชื่อของโปรแกรมดังรูปที่ ก.1

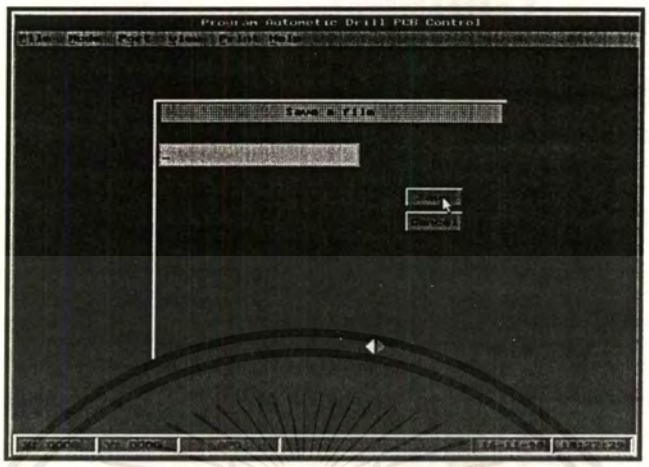
รูปที่ ก.1 ชื่อของโปรแกรม

จากนั้นเมื่อทำการกดคีย์ใดๆ หรือรอประมาณ 5 วินาที โปรแกรมจะแสดงหน้าต่างเมนูดังรูปที่ ก.2



รูปที่ ก.2 หน้าต่างเมนูของโปรแกรม

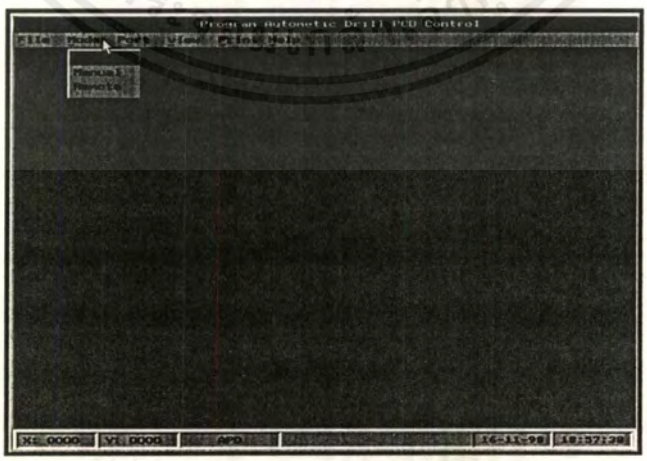
- **Save** เป็นคำสั่งที่ใช้ในการบันทึกไฟล์



รูปที่ ก.5 แสดงหน้าต่างเมนู Save files

- **Change Drv** เป็นคำสั่งที่ใช้ในการเปลี่ยนไดเร็กทรอรี่
- **Dos** เป็นคำสั่งที่ใช้ในการหยุดโปรแกรมชั่วคราวแล้วไปทำงานในโหมดของดอส
- **Quit** เป็นคำสั่งที่ใช้จบการทำงานของโปรแกรม

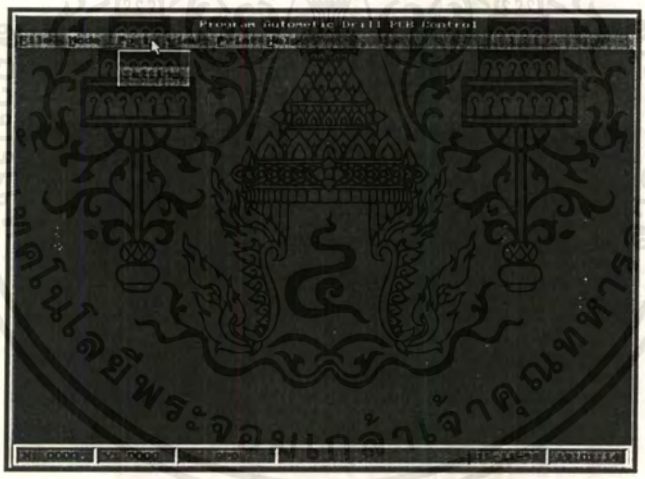
MODE เป็นคำสั่งที่ใช้ควบคุมการส่งข้อมูลไปยังเครื่องเจาะแผ่นวงจรพิมพ์อัตโนมัติ โดยสามารถส่งข้อมูลแบบอัตโนมัติโดยการอ่านข้อมูลจากไฟล์ ส่งข้อมูลแบบควบคุมด้วยคีย์บอร์ด



รูปที่ ก.6 หน้าต่างเมนู Mode

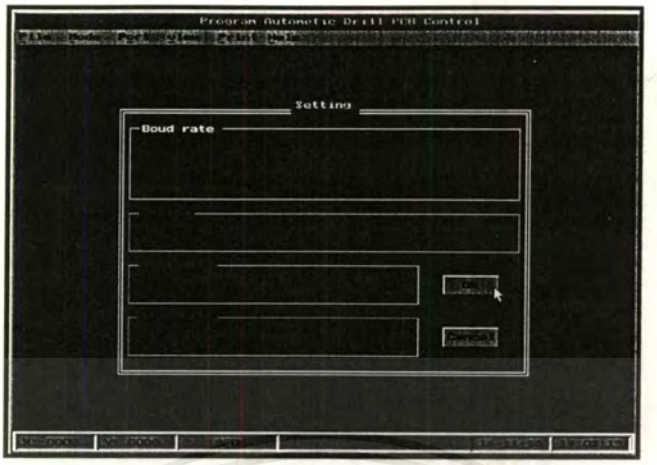
- **Auto** เป็นคำสั่งที่ใช้ควบคุมการส่งข้อมูลไปยังเครื่องเจาะแผ่นวงจรพิมพ์อัตโนมัติแบบอัตโนมัติโดยการอ่านข้อมูลจากไฟล์สกุล PCB หรือ DRL โดยไม่ตรวจเช็คสภาวะของเครื่องเจาะ
- **Manual** เป็นคำสั่งที่ใช้ควบคุมการส่งข้อมูลไปยังเครื่องเจาะแผ่นวงจรพิมพ์อัตโนมัติแบบอัตโนมัติโดยการอ่านข้อมูลจากไฟล์สกุล PCB หรือ DRL โดยตรวจเช็คสภาวะของเครื่องเจาะ
- **Remote** เป็นคำสั่งที่ใช้ควบคุมการส่งข้อมูลไปยังเครื่องเจาะแผ่นวงจรพิมพ์อัตโนมัติ แบบควบคุมตำแหน่งรูเจาะด้วยคีย์บอร์ด

PORT เป็นคำสั่งที่ใช้เลือกพอร์ตในการส่งข้อมูล การเซ็ทอัตราความเร็วในการส่งข้อมูล, ลักษณะบิตตรวจสอบ, จำนวนบิตข้อมูล รวมถึงบิตหยุดด้วย



รูปที่ ก.7 หน้าต่างเมนู Port

- **Port** เป็นคำสั่งที่ใช้ในการเลือกพอร์ต COM 1 และ COM 2
- **Setting** เป็นคำสั่งที่ใช้ในการเซ็ทคุณสมบัติของการส่งข้อมูล เช่น อัตราเร็วในการส่งข้อมูล (Baud rate) , ลักษณะของบิตตรวจสอบ (Parity Bit) , จำนวนบิตข้อมูล (Data Bit) และบิตหยุด (Stop Bit) เป็นต้น



รูปที่ ก.8 หน้าต่าง Setting port

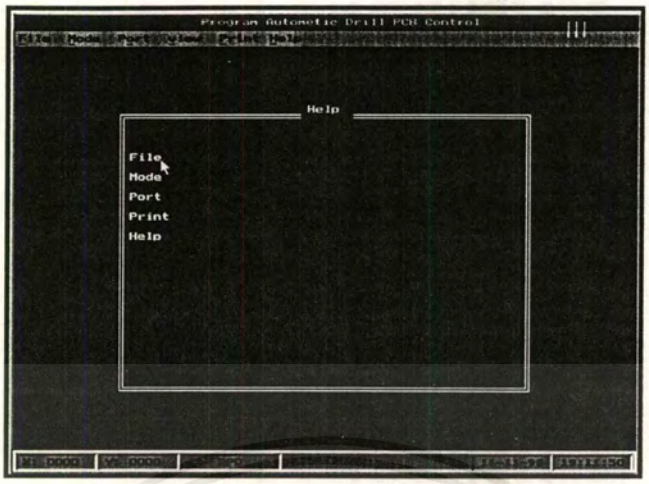
VIEW เป็นคำสั่งที่ใช้ในการย่อหรือขยาย ขนาดของรูปภาพ

- **Zoom Out** เป็นคำสั่งที่ใช้ในการย่อขนาดของรูปภาพ
- **Zoom In** เป็นคำสั่งที่ใช้ในการขยายขนาดของรูปภาพ

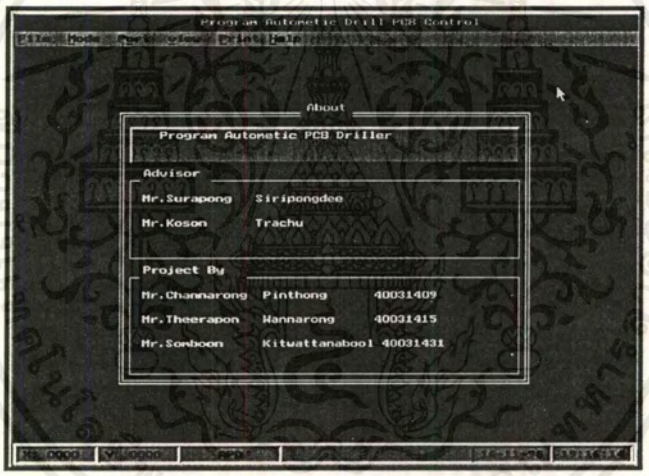
รูปที่ ก.9 หน้าต่างเมนู View

PRINT เป็นคำสั่งที่ใช้ในการสั่งข้อมูลออกทางเครื่องพิมพ์ (PRINTER)

- **Locate** เป็นคำสั่งที่ใช้ในการพิมพ์ข้อมูลเฉพาะตำแหน่งรูเจาะออกทางเครื่องพิมพ์
- **Circuit** เป็นคำสั่งที่ใช้ในการพิมพ์รูปวงจรออกทางเครื่องพิมพ์



รูปที่ ก.12 เมนู Help



รูปที่ ก.13 เมนู About

ภาคผนวก ข

โปรแกรมภาษาซี

ภาคผนวก ข

ไฟล์ about.c

```
void about(int x1,int y1,int x2,int y2)
{
        struct textsettingstype textinfo;
        char color=5,ch;
        union REGS regs;
        gettextsettings(&textinfo);
        settextstyle(2,0,6);
        setfillstyle(1,color);
        bar(x1,y1,x2,y2);
        setcolor(15);
        rectangle(x1+10,y1+10,x2-10,y2-10);
        rectangle(x1+13,y1+13,x2-13,y2-13);
        bar(x1-26+(x2-x1)/2,y1,x1+26+(x2-x1)/2,y1+15);
        setcolor(0);
        outtextxy(x1-20+(x2-x1)/2,y1,"About");
        setcolor(11);
        outtextxy(x1-19+(x2-x1)/2,y1+1,"About");
        bottomup(x1+20,y1+25,x2-20,y1+62,2,7);
        settextstyle(2,0,7);
        setcolor(2);
        outtextxy(x1+48,y1+28,"Program Autometic PCB Driller");
        setcolor(9);
        outtextxy(x1+49,y1+29,"Program Autometic PCB Driller");
        setcolor(15);
        outtextxy(x1+50,y1+30,"Program Autometic PCB Driller");
        rectangle(x1+20,y1+80,x2-20,y1+160);
```

```
 setcolor(0);
 rectangle(x1+21,y1+81,x2-21,y1+159);
 rectangle(x1+19,y1+79,x2-19,y1+161);
 setfillstyle(1,color);
 bar(x1+30,y1+65,x1+100,y1+85);
 setcolor(0);
 settextstyle(2,0,6);
 setcolor(3);
 outtextxy(x1+33,y1+68,"Advisor");
 setcolor(15);
 outtextxy(x1+34,y1+69,"Advisor");
 outtextxy(x1+33,y1+95,"Mr.Surapong   Siripongdee");
 outtextxy(x1+33,y1+120,"Mr.Koson     Trachu");
 rectangle(x1+20,y1+180,x2-20,y1+280);
 setcolor(0);
 rectangle(x1+21,y1+181,x2-21,y1+279);
 rectangle(x1+19,y1+179,x2-19,y1+281);
 setfillstyle(1,color);
 bar(x1+30,y1+165,x1+135,y1+185);
 setcolor(3);
 outtextxy(x1+33,y1+168,"Project By");
 setcolor(15);
 outtextxy(x1+34,y1+169,"Project By");
 outtextxy(x1+33,y1+195,"Mr.Channarong  Pinthong     40031409");
 outtextxy(x1+33,y1+220,"Mr.Theerapon   Wannarong    40031415");
 outtextxy(x1+33,y1+245,"Mr.Somboon     Kitwattanabool 40031431");
 settextstyle(textinfo.font,textinfo.direction,textinfo.charsize);
 showmouse();
 do
 {
```

```
            ch=getnw();
            regs.x.ax=3;
            int86(0x33,&regs,&regs);
        }while(ch=='\x0'&& !(regs.x.bx&1));
        hidemouse();
}
```

ไฟล์ help.c

```
char choice_help[6][10]={"File","Mode","Port","Print","Help"};
void backgroundhelp(int x1,int y1,int x2,int y2)
{
        setcolor(0);
        settextstyle(0,0,1);
        outtextxy(x1+20,y1+30,"Help for program Autometic PCB Driller");
        setcolor(YELLOW);
        outtextxy(x1+20,y1+50,choice_help[0]);
        outtextxy(x1+20,y1+70,choice_help[1]);
        outtextxy(x1+20,y1+90,choice_help[2]);
        outtextxy(x1+20,y1+110,choice_help[3]);
        outtextxy(x1+20,y1+130,choice_help[4]);
        setcolor(0);
        outtextxy(x1+68,y1+50,"manage with file");
        outtextxy(x1+68,y1+70,"Select mode to control");
        outtextxy(x1+68,y1+90,"Select port to out data");
        outtextxy(x1+68,y1+110,"Out data to printer");
        outtextxy(x1+68,y1+130,"Help for use program");
        showmouse();
}
```

```
movecursor(int x1,int y1,int x2,int y2)
{
        char subkeyhelp;
        int subcursorx=x1+20,subcursory=y1+32,xmouse,ymouse;
        union REGS regs;
        setcolor(0);
        outtextxy(subcursorx,subcursory,"_");
        do
        {
                do
                {
                        subkeyhelp=getnw();
                        regs.x.ax=3;
                        int86(0x33,&regs,&regs);
                        xmouse=regs.x.cx;
                        ymouse=regs.x.dx;
                        delay(50);
                }while(subkeyhelp=='\x0'&& !(regs.x.bx&1));
                if((regs.x.bx&1) && (xmouse<x1 || xmouse>x2 || ymouse<y1 || ymouse>y2))
                        subkeyhelp=ESC;
                setcolor(5);
                switch(subkeyhelp)
                {
                        case DOWN :     if(subcursory<y2-30)
                                        {
                                                outtextxy(subcursorx,subcursory,"_");
                                                subcursory+=20;
                                        }
                                        break;
                        case UP :       if(subcursory>y1+30)
```

```
                                {
                                        outtextxy(subcursorx,subcursory,"_");
                                        subcursory-=20;
                                }
                                break;
                        case RIGHT :    if(subcursorx<x2-28)
                                {
                                        outtextxy(subcursorx,subcursory,"_");
                                        subcursorx+=8;
                                }
                                break;
                        case LEFT :     if(subcursorx>x1+20)
                                {
                                        outtextxy(subcursorx,subcursory,"_");
                                        subcursorx-=8;
                                }
                                break;
                }
                setcolor(0);
                outtextxy(subcursorx,subcursory,"_");
        }while(subkeyhelp!=ESC);
}

help_file(int x1,int y1,int x2,int y2)
{
        hidemouse();
        setfillstyle(1,5);
        setcolor(5);
        bar(x1+15,y1+20,x2-15,y2-20);
        setcolor(0);
```

```
        outtextxy(x1+20,y1+30,"Manage with file and menu file consist of");
        outtextxy(x1+20,y1+50,"New       - Clear display.");
        outtextxy(x1+20,y1+70,"Open      - Open file.File have 2 case");
        outtextxy(x1+20,y1+90,"          case 1 format file is *.DRL,");
        outtextxy(x1+20,y1+110,"          it is data of locate drill.");
        outtextxy(x1+20,y1+130,"          case 2 format file is *.PCB,");
        outtextxy(x1+20,y1+150,"          it is data of circuit line.");
        outtextxy(x1+20,y1+170,"Save      - Save file is *.DRL format.");
        outtextxy(x1+20,y1+190,"Change Drv - Change drive.");
        outtextxy(x1+20,y1+210,"Dos       - Exit to dos but close this program.");
        outtextxy(x1+20,y1+230,"          Type exit to return program.");
        outtextxy(x1+20,y1+250,"Quit      - Quit out program.");
        showmouse();
        movecursor(x1,y1,x2,y2);
        setfillstyle(1,5);
        setcolor(5);
        hidemouse();
        bar(x1+15,y1+20,x2-15,y2-20);
        backgroundhelp(x1,y1,x2,y2);
}

help_mode(int x1,int y1,int x2,int y2)
{
        hidemouse();
        setfillstyle(1,5);
        setcolor(5);
        bar(x1+15,y1+20,x2-15,y2-20);
        setcolor(0);
        outtextxy(x1+20,y1+30,"Select mode to control Autometic PCB Drill");
        outtextxy(x1+20,y1+50,"Mode have 3 style");
```

```
        outtextxy(x1+20,y1+70,"Auto   - Send data to Autometic PCB Drill");
        outtextxy(x1+20,y1+90,"         and pursue work of machine");
        outtextxy(x1+20,y1+110,"Manual - Send data to Autometic PCB Drill");
        outtextxy(x1+20,y1+130,"         and do not pursue work of machine");
        outtextxy(x1+20,y1+150,"Remote - Move tip drill with arrow key on");
        outtextxy(x1+20,y1+170,"         keyboard");
        showmouse();
        movecursor(x1,y1,x2,y2);
        hidemouse();
        setfillstyle(1,5);
        setcolor(5);
        bar(x1+15,y1+20,x2-15,y2-20);
        backgroundhelp(x1,y1,x2,y2);
}

help_port(int x1,int y1,int x2,int y2)
{
        hidemouse();
        setfillstyle(1,5);
        setcolor(5);
        bar(x1+15,y1+20,x2-15,y2-20);
        setcolor(0);
        outtextxy(x1+20,y1+30,"Select port to use is Com1 or Com2");
        showmouse();
        movecursor(x1,y1,x2,y2);
        hidemouse();
        setfillstyle(1,5);
        setcolor(5);
        bar(x1+15,y1+20,x2-15,y2-20);
        backgroundhelp(x1,y1,x2,y2);
```

```
}

help_print(int x1,int y1,int x2,int y2)
{
        hidemouse();
        setfillstyle(1,5);
        setcolor(5);
        bar(x1+15,y1+20,x2-15,y2-20);
        setcolor(0);
        outtextxy(x1+20,y1+30,"Print line circuit or locate hole.If select");
        outtextxy(x1+20,y1+50,"locate will print locate hole to printer or");
        outtextxy(x1+20,y1+70,"select circuit will print line circuit to");
        outtextxy(x1+20,y1+90,"printer");
        showmouse();
        movecursor(x1,y1,x2,y2);
        hidemouse();
        setfillstyle(1,5);
        setcolor(5);
        bar(x1+15,y1+20,x2-15,y2-20);
        backgroundhelp(x1,y1,x2,y2);
}

help_help(int x1,int y1,int x2,int y2)
{
        hidemouse();
        setfillstyle(1,5);
        setcolor(5);
        bar(x1+15,y1+20,x2-15,y2-20);
        setcolor(0);
        outtextxy(x1+20,y1+30,"Help have 2 submenu");
```

```
        outtextxy(x1+20,y1+50,"Help  - Help for use program");
        outtextxy(x1+20,y1+70,"About - Display about adviser name and creator");
        outtextxy(x1+20,y1+90,"       name.");
        showmouse();
        movecursor(x1,y1,x2,y2);
        hidemouse();
        setfillstyle(1,5);
        setcolor(5);
        bar(x1+15,y1+20,x2-15,y2-20);
        backgroundhelp(x1,y1,x2,y2);
}

void pro_help(int x1,int y1,int x2,int y2)
{
        struct textsettingstype textinfo;
        int cursorx=x1+20,cursory=y1+32,bufkey[2],xmouse,ymouse;
        char color=5,statekey=0;
        char keyhelp;
        union REGS regs;
        gettextsettings(&textinfo);
        settextstyle(2,0,6);
        setfillstyle(1,color);
        bar(x1,y1,x2,y2);
        setcolor(15);
        rectangle(x1+10,y1+10,x2-10,y2-10);
        rectangle(x1+13,y1+13,x2-13,y2-13);
        bar(x1-26+(x2-x1)/2,y1,x1+26+(x2-x1)/2,y1+15);
        setcolor(0);
        outtextxy(x1-20+(x2-x1)/2,y1,"Help");
        setcolor(11);
```

```
outtextxy(x1-19+(x2-x1)/2,y1+1,"Help");
backgroundhelp(x1,y1,x2,y2);
setcolor(0);
outtextxy(cursorx,cursory,"_");
showmouse();
do
{
        do
        {
                keyhelp=getnw();
                regs.x.ax=3;
                int86(0x33,&regs,&regs);
                xmouse=regs.x.cx;
                ymouse=regs.x.dx;
        }while(keyhelp=='\x0'&& !(regs.x.bx&1));
        if(regs.x.bx&1)
        {
                if(xmouse>x1 && xmouse<x2 && ymouse>y1 && ymouse<y2)
                        keyhelp=RETURN;
                else
                        keyhelp=ESC;
        }
        setcolor(5);
        bufkey[0]=cursory;
        switch(keyhelp)
        {
                case DOWN :     if(cursory<y2-30)
                                {
                                        outtextxy(cursorx,cursory,"_");
                                        cursory+=20;
```

```
                    bufkey[1]=1;
                }
                break;
        case UP :   if(cursory>y1+30)
                {
                    outtextxy(cursorx,cursory,"_");
                    cursory-=20;
                    bufkey[1]=2;
                }
                break;
        case RIGHT: if(cursorx<x2-28)
                {
                    outtextxy(cursorx,cursory,"_");
                    cursorx+=8;
                }
                break;
        case LEFT : if(cursorx>x1+20)
                {
                    outtextxy(cursorx,cursory,"_");
                    cursorx-=8;
                }
                break;
}
if(keyhelp==RETURN)
{
    if(cursory==(y1+52) || (xmouse>x1+20 && xmouse<x1+60 &&
    ymouse>y1+45 && ymouse<y1+65))
    {
        help_file(x1,y1,x2,y2);
    }
```

```
            if(cursory==(y1+72)||(xmouse>x1+20 && xmouse<x1+60 &&
            ymouse>y1+65 && ymouse<y1+85))
            {
                    help_mode(x1,y1,x2,y2);
            }
            if(cursory==(y1+92) || (xmouse>x1+20 && xmouse<x1+60 &&
            ymouse>y1+85 && ymouse<y1+105))
            {
                    help_port(x1,y1,x2,y2);
            }
            if(cursory==(y1+112)||(xmouse>x1+20 && xmouse<x1+60 &&
            ymouse>y1+105 && ymouse<y1+125))
            {
                    help_print(x1,y1,x2,y2);
            }
            if(cursory==(y1+132)||(xmouse>x1+20 && xmouse<x1+60 &&
            ymouse>y1+125 && ymouse<y1+145))
            {
                    help_help(x1,y1,x2,y2);
            }
}
setcolor(0);
outtextxy(cursorx,cursory,"_");
if(cursorx>=x1+20 && cursorx<x1+60 && cursory >= y1+52 &&
cursory<=y1+132)
{
        if(cursory>=y1+52 && cursory<y1+72)
        {
                if(statekey==1)
                {
```

```
                if(bufkey[1]==2)
                {
                        setcolor(5);
                        setfillstyle(1,5);
                        bar(x1+15,bufkey[0]-5,x1+60,bufkey[0]+10);
                        setcolor(YELLOW);
                        outtextxy(x1+20,bufkey[0]-2,choice_help[1]);
                }
        }
        setcolor(BLUE);
        setfillstyle(1,BLUE);
        bar(x1+15,cursory-5,x1+60,cursory+10);
        setcolor(YELLOW);
        outtextxy(x1+20,y1+50,choice_help[0]);
        statekey=1;
}
if(cursory>=y1+72 && cursory < y1+92)
{
        if(statekey==1)
        {
                setcolor(5);
                setfillstyle(1,5);
                bar(x1+15,bufkey[0]-5,x1+60,bufkey[0]+10);
                setcolor(YELLOW);
                if(bufkey[1]==1)
                        outtextxy(x1+20,bufkey[0]-2,choice_help[0]);
                else
                        outtextxy(x1+20,bufkey[0]-2,choice_help[2]);
        }
        setcolor(BLUE);
```

```
            setfillstyle(1,BLUE);
            bar(x1+15,cursory-5,x1+60,cursory+10);
            setcolor(YELLOW);
            outtextxy(x1+20,y1+70,choice_help[1]);
            statekey=1;
}
if(cursory>=y1+92 && cursory < y1+112)
{
            if(statekey==1)
            {
                        setcolor(5);
                        setfillstyle(1,5);
                        bar(x1+15,bufkey[0]-5,x1+60,bufkey[0]+10);
                        setcolor(YELLOW);
                        if(bufkey[1]==1)
                                    outtextxy(x1+20,bufkey[0]-2,choice_help[1]);
                        else
                                    outtextxy(x1+20,bufkey[0]-2,choice_help[3]);
            }
            setcolor(BLUE);
            setfillstyle(1,BLUE);
            bar(x1+15,cursory-5,x1+60,cursory+10);
            setcolor(YELLOW);
            outtextxy(x1+20,y1+90,choice_help[2]);
            statekey=1;
}
if(cursory>=y1+112 && cursory < y1+132)
{
            if(statekey==1)
            {
```

```
                setcolor(5);
                setfillstyle(1,5);
                bar(x1+15,bufkey[0]-5,x1+60,bufkey[0]+10);
                setcolor(YELLOW);
                if(bufkey[1]==1)
                        outtextxy(x1+20,bufkey[0]-2,choice_help[2]);
                else
                        outtextxy(x1+20,bufkey[0]-2,choice_help[4]);
        }
        setcolor(BLUE);
        setfillstyle(1,BLUE);
        bar(x1+15,cursory-5,x1+60,cursory+10);
        setcolor(YELLOW);
        outtextxy(x1+20,y1+110,choice_help[3]);
        statekey=1;
}
if(cursory>=y1+132 && cursory < y1+152)
{
        if(statekey==1)
        {
                setcolor(5);
                setfillstyle(1,5);
                bar(x1+15,bufkey[0]-5,x1+60,bufkey[0]+10);
                setcolor(YELLOW);
                if(bufkey[1]==1)
                        outtextxy(x1+20,bufkey[0]-2,choice_help[3]);
                else
                        outtextxy(x1+20,bufkey[0]-2,choice_help[5]);
        }
        setcolor(BLUE);
```

```
                setfillstyle(1,BLUE);
                bar(x1+15,cursory-5,x1+60,cursory+10);
                setcolor(YELLOW);
                outtextxy(x1+20,y1+130,choice_help[4]);
                statekey=1;
        }
}
else
{
        if(statekey==1)
        {
                setcolor(5);
                setfillstyle(1,5);
                bar(x1+15,bufkey[0]-5,x1+60,bufkey[0]+10);
                setcolor(YELLOW);
                if((bufkey[0]-(y1+40))/10==1)
                        outtextxy(x1+20,bufkey[0]-2,choice_help[0]);
                if((bufkey[0]-(y1+40))/10==3)
                        outtextxy(x1+20,bufkey[0]-2,choice_help[1]);
                if((bufkey[0]-(y1+40))/10==5)
                        outtextxy(x1+20,bufkey[0]-2,choice_help[2]);
                if((bufkey[0]-(y1+40))/10==7)
                        outtextxy(x1+20,bufkey[0]-2,choice_help[3]);
                if((bufkey[0]-(y1+40))/10==9)
                        outtextxy(x1+20,bufkey[0]-2,choice_help[4]);
                if((bufkey[0]-(y1+40))/10==11)
                        outtextxy(x1+20,bufkey[0]-2,choice_help[5]);
        }
        statekey=0;
}
```

```
        }while(keyhelp!=ESC);
        hidemouse();
        settextstyle(textinfo.font,textinfo.direction,textinfo.charsize);
}
```

ไฟล์ readfile.c

```
#include <dir.h>
#include<string.h>
#include<alloc.h>
#ifdef __cplusplus
        #define __CPPARGS ...
#else
        #define __CPPARGS
#endif
int failcheck2=0;
int showmouse();
hidemouse();
int showmouse();
int mousedrv();
char getnw();
void interrupt(*Oldint2)(__CPPARGS);
void interrupt Myint2(__CPPARGS)
{
        failcheck2=1;
}
char chkdsk(int disk);
int boxerror();
setnewint();
setoldint();
```

```
int consx1 =150, consy1= 100, consx2= 500, consy2= 370,mouseindex=0;
char *bufname;
char value[10];
ftriangle(int x1,int y1,char color);
btriangle(int x1,int y1,char color);
struct far singlylink
{
        char data[13];
        unsigned long size;
        struct singlylink *next;
};
struct far singlylink *headnode,*walknode,*newnode;
void *savepic;
unsigned sizepic;

setnewint2()
{
        Oldint2=getvect(0x24);
        setvect(0x24,Myint2);
}

setoldint2()
{
        setvect(0x24,Oldint2);
}

void addnode(char dat[],unsigned long datsize)
{
        newnode=(struct singlylink far*)farmalloc(sizeof(struct singlylink));
        walknode->next=newnode;
```

```
        walknode=newnode;
        strcpy(newnode->data,dat);
        newnode->size=datsize;
        newnode->next=NULL;
}

void cursor()
{
        setcolor(8);
        setlinestyle(0,0,2);
        line(getx(),gety()+8,getx()+5,gety()+8);
        setcolor(1);
}

void clrtext(char *strname,int subcount,int subindex,int x,int y)
{
        char buf[2],i;
        hidemouse();
        buf[1]='\0';
        if(subcount>=23) subcount-=23;
        else subcount=0;
        setfillstyle(1,11);
        bar(x,y,x+190,y+14);
        setcolor(1);
        for(i=subindex;i<subindex+23;i++)
        {
                buf[0]=strname[i];
                buf[1]='\0';
                outtextxy(x,y,buf);
                x+=8;
```

```
        }
        showmouse();
}

void clrcursor(int x1,int y1)
{
        hidemouse();
        setcolor(11);
        setlinestyle(0,0,11);
        line(x1,y1+8,x1+5,y1+8);
        setcolor(1);
        showmouse();
}

char *Openfile()
{
        char ch1[2],ch,*name,*buf="\v";
        int i=0,j,xtext=157,ytext=151,count=0,check1=0,check=0,ins=0,index=0;
        int xbox=157,ybox=151,xmouse,ymouse;
        union REGS iregs;
        hidemouse();
        buf=(char *)malloc(3);
        strcpy(buf,"\v");
        name=(char *)malloc(100);
        for(j=0;j<100;j++)
                name[j]=0;
        moveto(xtext,ytext);
        cursor();
        showmouse();
        do
```

```
{
        do
        {
                ch=getnw();
                iregs.x.ax=3;
                int86(0x33,&iregs,&iregs);
                xmouse=iregs.x.cx;
                ymouse=iregs.x.dx;
                showmouse();
                if(iregs.x.bx&1 && xmouse>152 && xmouse<385 && ymouse>190
                && ymouse<ybox+345)
                { mouseindex=1;hidemouse();return(name); }
                if(iregs.x.bx&1 && xmouse>400 && xmouse<455 && ymouse>190
                && ymouse<208)
                {mouseindex=2;hidemouse();return(name);}
                if(iregs.x.bx&1 && xmouse>400 && xmouse<455 && ymouse>213
                && ymouse<231)
                {mouseindex=3;hidemouse();ch=ESC;}
        }while(ch=='\x0' && !(iregs.x.bx&1));
        if(ch==0)
        {
                ch=getnw();
                switch(ch)
                {
                        case 75: if(xtext>=xbox)
                                {
                                        clrcursor(xtext,ytext);
                                        xtext-=8;
                                        if(xtext<=xbox+192 && count>0)
                                                count--;
```

```
                if(xtext<xbox && index > 0)
                {
                        index--;
                        clrtext(name,count,index,xbox,ybox);
                }
                if(xtext<xbox)    xtext+=8;
                moveto(xtext,ytext);
        }break;
case 77: if(xtext < (xbox+(check*8)) && xtext<xbox+182)
        {
                clrcursor(xtext,ytext);
                xtext+=8;count++;
                if(xtext>xbox+182 && name[count]!='\0')
                {
                        index++;
                        clrtext(name,count,index,xbox,ybox);
                        xtext-=8;
                }
                moveto(xtext,ytext);
        }break;
case 83: for(j=(count);*(name+j)!='\0';j++)
                *(name+j)=*(name+j+1);
        check--;
        clrtext(name,count,index,xbox,ybox);
        break;
case 0x52 : if(ins==0)
                ins=1;
        else ins=0;
```

```
                        break;
                }
        }
        else
        {
                if((ch>=33 && ch <126) && (i<24))
                {
                        check1=check-count;
                        if(ins==1)
                        {
                                for(j=0;j<=check1;j++)
                                {
                                        *(name+check-j+1)=*(name+check-j);
                                }
                        *(name+count)=ch;
                        }
                        else
                        {
                                *(name+count)=ch;
                        }
                        check=strlen(name);
                        setcolor(7);
                        clrcursor(xtext,ytext);
                        if(xtext<=xbox+190)      xtext+=8;
                        if(xtext>xbox+190 && ins==0)   {index++;xtext-=8;}
                        count++;
                        moveto(xtext,ytext);
                        clrtext(name,count,index,xbox,ybox);
                }
        }
```

```
            if(ch==8)
            {
                  if(count>0)
                  {
                        for(j=(count-1);*(name+j)!='\0';j++)
                              *(name+j)=*(name+j+1);
                        clrcursor(xtext,ytext);
                        count--;
                        if(xtext>xbox)    xtext-=8;
                        i--;
                        moveto(xtext,ytext);
                        clrtext(name,count,index,xbox,ybox);
                        if(count==index && index !=0)   index--;
                  }
            }
            cursor();
      }while(ch!=13 && ch!=27 && ch!='\t' );
      if(ch=='\t')        { mouseindex=1;return(name);}
      if(ch==RETURN)
      {
            strcat(name,buf);
            free(buf);
            return(name);
      }if(ch==ESC)    return("EOF");
}

int checkstatedrv()
{
      int subcasesel,drvinit;
      setnewint2();
```

```
        chkdsk(getdisk()+1);
        if(failcheck2==1)
        {
                do
                {
                        if(failcheck2==1)
                        {
                                failcheck2=0;
                                subcasesel=boxerror();
                        }
                        if(subcasesel==0)       chkdsk(getdisk()+1);
                }while(failcheck2==1);
                if(subcasesel==0)       setdisk(getdisk());
                else return(0);
        }setoldint2();
}

int readnamefile(char *wildcard)
{
        struct ffblk ffblk;
        int done,counter=0,statedrv;
        statedrv=checkstatedrv();
        if(statedrv==0)  return(0);
        headnode=(struct singlylink far*)farmalloc(sizeof(struct singlylink));
        headnode->next=NULL;
        walknode=headnode;
        done = findfirst("*.*",&ffblk,0x10);
```

```
        while(!done)
        {
                if(ffblk.ff_attrib==16 || ffblk.ff_attrib==17)
                {
                        addnode(ffblk.ff_name,ffblk.ff_fsize);
                        counter++;
                }done = findnext(&ffblk);
        }
        done=findfirst(wildcard,&ffblk,0);
        while (!done)
        {
                if(ffblk.ff_fsize!=0)
                {
                        addnode(ffblk.ff_name,ffblk.ff_fsize);
                        counter++;
                }done=findnext(&ffblk);
        }walknode=headnode->next;
        return(counter);
}

void boxbar(int x1,int y1,int x2,int y2)
{
        setfillstyle(1,7);
        bar(x1-2,y1-2,x2+2,y2+2);
        setfillstyle(1,11);
        bar(x1,y1,x2,y2);
}

void boxbars(int x1,int y1,int x2,int y2)
{
```

```
        setfillstyle(1,10);
        bar(x1-2,y1-2,x2+2,y2+2);
        setfillstyle(1,11);
        bar(x1,y1,x2,y2);
}

void bigboxbars(int x1,int y1,int x2,int y2)
{
        setlinestyle(0,0,3);
        setcolor(10);
        line(x1+115,y2-180,x1+115,y2-23);
        rectangle(x1+2,y2-180,x1+235,y2-23);
        rectangle(x1+205,y2-23,x1+235,y2-2);
        rectangle(x1+2,y2-23,x1+235,y2-2);
}

void bottom(int x1,int y1,int x2,int y2)
{
        setfillstyle(1,7);
        bar(x1,y1,x2,y2);
        setcolor(15);
        setlinestyle(0,15,2);
        line(x1,y1,x2,y1);
        line(x1,y1,x1,y2-1);
        setcolor(8);
        setlinestyle(0,8,2);
        line(x1+1,y2,x2,y2);
        line(x2,y1+1,x2,y2);
}
```

```
void showfile(int xwinfile1,int ywinfile1,int xwinfile2,int ywinfile2,char wincolor,int subopen_save)
{
        char textbar[2][12]={"Open a file","Save a file"};
        char textbottom[2][6]={"Open","Save"};
        setcolor(7);
        setfillstyle(1,15);
        bar(xwinfile1-2,ywinfile1-2,xwinfile2,ywinfile2);
        setfillstyle(1,wincolor);
        bar(xwinfile1,ywinfile1,xwinfile2,ywinfile2);
        setcolor(1);
        setlinestyle(0,0,3);
        line(xwinfile1+115,ywinfile2-180,xwinfile1+115,ywinfile2-23);
        rectangle(xwinfile1+2,ywinfile2-180,xwinfile1+235,ywinfile2-23);
        rectangle(xwinfile1+205,ywinfile2-23,xwinfile1+235,ywinfile2-2);
        rectangle(xwinfile1+2,ywinfile2-23,xwinfile1+235,ywinfile2-2);
        setcolor(1);
        outtextxy(xwinfile1+5,ywinfile2-240,"Name");
        setcolor(1);
        outtextxy(xwinfile1+5,ywinfile2-192,"File");
        boxbar(xwinfile1+5,ywinfile2-225,xwinfile1+200,ywinfile2-205);
        setfillstyle(1,10);
        bar(xwinfile1+5,ywinfile2-268,xwinfile2-5,ywinfile2-250);
        outtextxy(xwinfile1+(xwinfile2-xwinfile1)/2-45,ywinfile2-263,textbar[subopen_save]);
        bottom(xwinfile1+250,ywinfile2-180,xwinfile1+305,ywinfile2-162);
        outtextxy(xwinfile1+262,ywinfile2-174,textbottom[subopen_save]);
        bottom(xwinfile1+250,ywinfile2-155,xwinfile1+305,ywinfile2-136);
        outtextxy(xwinfile1+255,ywinfile2-148,"Cancel");
}
```

```
void outtextcolor(int xx,int yy,int pointfile,char atrib)
{
        int j;
        if(atrib==0)
        {
                setcolor(3);
                setfillstyle(1,3);
        }else
        {
                setcolor(13);
                setfillstyle(1,13);
        }
        bar(xx-5,yy-5,xx+100,yy+10);
        walknode=headnode->next;
        for(j=1;j<pointfile;j++)
        {
                walknode=walknode->next;
        }
        setcolor(0);
        outtextxy(xx,yy,walknode->data);
}

ftriangle(int x1,int y1,char color)
{
        int tri[]={x1,y1,x1,y1+14,x1+7,y1+7,x1,y1};
        setcolor(color);
        setfillstyle(1,color);
        fillpoly(3,tri);
}
```

```
btriangle(int x1,int y1,char color)
{
        int tri[]={x1-4,y1,x1-4,y1+14,x1-11,y1+7,x1-4,y1};
        setcolor(color);
        setfillstyle(1,color);
        fillpoly(3,tri);
}

void clrbigbox(int x1,int y1,int x2,int y2,char mode)
{
        setfillstyle(1,3);
        if(mode!=1)     bar(x1+2,y2-180,x1+235,y2-2);
        setcolor(mode);
        setlinestyle(0,0,3);
        line(x1+115,y2-180,x1+115,y2-23);
        rectangle(x1+2,y2-180,x1+235,y2-23);
        rectangle(x1+205,y2-23,x1+235,y2-2);
        rectangle(x1+2,y2-23,x1+235,y2-2);
}

void pshiftdisplay(int x1,int y1,int subcount,int subpointcount)
{
        int i;
        clrbigbox(consx1,consy1,consx2,consy2,10);
        setcolor(0);
        walknode=headnode->next;
        for(i=1;i<=(subpointcount*10) && i<=subcount;i++)
        {
                walknode=walknode->next;
```

```
        }
        for(i=1;i<=(subcount-(subpointcount*10)) && x1<=300 ;i++)
        {
                outtextxy(x1,y1,walknode->data);
                y1+=15;
                if(y1>335)        { x1+=120;y1=200; }
                walknode=walknode->next;
        }
}

void nshiftdisplay(int x1,int y1,int subcount,int subpointcount)
{
        int i;
        clrbigbox(consx1,consy1,consx2,consy2,10);
        setcolor(0);
        walknode=headnode->next;
        for(i=1;i<=(subpointcount*10) && i<=subcount;i++)
        {
                walknode=walknode->next;
        }
        for(i=1;i<(subcount-(subpointcount*10)) && x1<=300;i++)
        {
                outtextxy(x1,y1,walknode->data);
                y1+=15;
                if(y1>335)        { x1+=120;y1=200; }
                walknode=walknode->next;
        }
}
```

```
void freemem()
{
        int i;
        do
        {
                walknode=headnode;
                headnode=headnode->next;
                farfree(walknode);
        }while(headnode!=NULL);
}

void clrfilename()
{
        setfillstyle(1,3);
        bar(consx1+5,consy2-15,consx1+200,consy2-4);
}

char *selectfile(int init,char *wild,int open_save)
{
        int xx1=160,yy1=200,i,count,pointer=1,pointcount=0,temp,xmouse,ymouse;
        int lp1=0,lp2=0,countmouse=0;
        char key=0,*buf="\v";
        int consxx1=xx1,consyy1=yy1;
        union REGS iregs;
        strcpy(value,"");
        strcpy(buf,"\v");
        setcolor(7);
        setfillstyle(1,6);
        hidemouse();
        do
```

```
{
        xx1=160;yy1=200;pointer=1;pointcount=0;
        showfile(consx1,consy1,consx2,consy2,3,open_save);
        count=readnamefile(wild);
        if(count>10)
                ftriangle(consx1+222,consy2-20,13);
        else
                ftriangle(consx1+222,consy2-20,15);
        btriangle(consx1+222,consy2-20,15);
        if(count==0)    return("");
        walknode=headnode->next;
        setcolor(0);
        for(i=1;i<=count && xx1<=300;i++)
        {
                outtextxy(xx1,yy1,walknode->data);
                yy1+=15;
                if(yy1>335)     { xx1+=120;yy1=200; }
                walknode=walknode->next;
        }
        xx1=consxx1;yy1=consyy1;
        if(init==0)         { freemem();return(0); }
        if(headnode->next==NULL)        { freemem();return(0); }
        outtextcolor(xx1,yy1,pointer,1);
        ltoa(walknode->size,value,10);
        setcolor(1);
        outtextxy(consx1+120,consy2-15,value);
        outtextxy(consx1+5,consy2-15,walknode->data);
        bigboxbars(consx1,consy2,consx1,consy2);
        showmouse();
        do
```

```
{
        _AX=5;
        _BX=0;
        geninterrupt(0x33);
        countmouse=_BX;
        do
        {
                key=getnw();
                iregs.x.ax=3;
                int86(0x33,&iregs,&iregs);
                xmouse=iregs.x.cx;
                ymouse=iregs.x.dx;
                showmouse();
        }while(key=='\x0' && !(iregs.x.bx&1));
        if(iregs.x.bx&1) delay(175);
        _AX=5;
        _BX=0;
        geninterrupt(0x33);
        countmouse=_BX;
        hidemouse();
        if(iregs.x.bx&1 && xmouse>157 && xmouse<347 && ymouse>151
                && ymouse<165)
        {
                mouseindex=0;hidemouse();outtextcolor(xx1,yy1,pointer,0);
                strcpy(value,"1");return("");
        }
        if(iregs.x.bx&1 && xmouse>400 && xmouse<455 && ymouse>190
          && ymouse<208)
        {
                mouseindex=2;hidemouse();key=RETURN;
```

```
}
if(iregs.x.bx&1 && xmouse>400 && xmouse<455 && ymouse>213
  && ymouse<231)
{
        mouseindex=3;key=ESC;hidemouse();
}       if(count<11) countloop1=1;else countloop1=2;
for(lp1=0;lp1<countloop1;lp1++)
{
        if(lp1==0 && countloop1==1)
                countloop2=count+1;
        if(lp1==0 && countloop1==2)
                countloop2=11;
        if(lp1==1 && countloop1==2)
        {
                if(pointer>(count-20) && pointside==(count/10)-1)
                        countloop2=(count%10)+1;
                else countloop2=11;
        }
        for(lp2=0;lp2<countloop2;lp2++)
        {
                if(iregs.x.bx&1 && xmouse>152+ (lp1*120)
                  && xmouse<272+(lp1*120)
                  && ymouse>180+(lp2*15)
                  && ymouse<180+((lp2+1)*15))
                {
                        outtextcolor(xx1,yy1,pointer,0);
                        xx1=consxx1+(lp1*120);
                        yy1=consyy1-30+(15*(lp2+1));
                        pointer=(pointside*10)+lp2+ (lp1*10);
                        outtextcolor(xx1,yy1,pointer,1);
```

```
                clrfilename();
                setcolor(1);
                outtextxy(consx1+5,consy2-15
                ,walknode->data);
                ltoa(walknode->size,value,10);
                outtextxy(consx1+120,consy2-15,value);
                if(countmouse>1)
                        key=RETURN;
                }
        }
}
consx1+222,consy2-20,15);
if(iregs.x.bx&1 && xmouse>consx1+222 && xmouse<consx1+22+7
&& ymouse>consy2-20 && ymouse<consy2-6)
        key = RIGHT;
switch(key)
{
        case LEFT :     if(xx1>consxx1)
                        {
                                outtextcolor(xx1,yy1,pointer,0);
                                xx1-=120;
                                pointer-=10;
                                outtextcolor(xx1,yy1,pointer,1);
                        }
                        else
                        {
                                if(pointer>10)
                                {
                                        xx1=consxx1;
                                        temp=yy1;
```

```
                                        yy1=consyy1;
                                        pointcount--;
                                        nshiftdisplay(xx1,yy1,count,
                                        pointcount);
                                        pointer-=10;
                                        yy1=temp;
                                        outtextcolor(xx1,yy1,
                                        pointer,1);
                                    }
                    }clrfilename();
                    setcolor(1);
                    outtextxy(consx1+5,consy2-15,walknode->
                    data);
                    ltoa(walknode->size,value,10);
                    outtextxy(consx1+120,consy2-15,value);
                    if(pointer+(count%10)-1<count)
                            ftriangle(consx1+222,consy2-20,13);
                    else
                            ftriangle(consx1+222,consy2-20,15);
                    if(pointer-10>=1)
                            btriangle(consx1+222,consy2-20,13);
                    else
                            btriangle(consx1+222,consy2-20,15);
            }break;
case RIGHT :    if(xx1<consxx1+120 && pointer<count)
            {
                    if(pointer+10<=count)
                    {
                            outtextcolor(xx1,yy1,pointer,0);
                            xx1+=120;pointer+=10;
```

```
                outtextcolor(xx1,yy1,pointer,1);
        }else
        {
                if(pointer+(count%10)-1<count)
                {
                        outtextcolor(xx1,yy1,pointer,
                        0);
                        xx1=consxx1;yy1=consyy1;
                        pointer=count;
                        xx1+=120;
                        yy1=consyy1+(((count %10)-
                        1)*15);
                        outtextcolor(xx1,yy1,pointer,
                        1);
                }
        }
    }else
    {
        if(pointer+9<count)
        {
                xx1=consxx1;
                temp=yy1;yy1=consyy1;
                pointcount++;
                pshiftdisplay(xx1,yy1,count,
                pointcount);
                pointer+=10;
                xx1+=120;yy1=temp;
                outtextcolor(xx1,yy1,pointer,1);
        }else
        {
```

```
                    if(pointer+(count%10)-1<count
                    && count%10 !=0)
                    {
                            xx1=consxx1;yy1=consyy1;
                            pointcount++;
                            pshiftdisplay(xx1,yy1,count,
                            pointcount);
                            pointer=count;
                            xx1+=120;
                            yy1=consyy1+(((count %10)-
                            1)*15);
                            outtextcolor(xx1,yy1,pointer,
                            1);
                    }
                }
        }clrfilename();
        setcolor(1);
        outtextxy(consx1+5,consy2-15,walknode->data);
        ltoa(walknode->size,value,10);
        outtextxy(consx1+120,consy2-15,value);
        if(pointer+(count%10)-1<count )
        {
                if((count%10)==0)
                {
                        if(pointer+(count%10)+10<=count)
                                ftriangle(consx1+222,consy2-
                                20,13);
                        else
                                ftriangle(consx1+222,consy2-
                                20,15);
```

```
                    }else
                        ftriangle(consx1+222,consy2-
                        20,13);
                }else
                    ftriangle(consx1+222,consy2-20,15);
                if(pointer-10>=1)
                    btriangle(consx1+222,consy2-20,13);
                else
                    btriangle(consx1+222,consy2-20,15);
            }break;
case DOWN :  if(yy1>consyy1+120 && pointer<count)
            {
                if(yy1>consyy1+120 && xx1>=consxx1+120)
                {
                    xx1=consxx1;yy1=consyy1;
                    pointcount++;
                    pshiftdisplay(xx1,yy1,count,
                    pointcount);
                    pointer++;
                    xx1+=120;
                    outtextcolor(xx1,yy1,pointer,1);
                }else
                {
                    outtextcolor(xx1,yy1,pointer,0);
                    xx1+=120;
                    yy1=consyy1;
                    pointer++;
                    outtextcolor(xx1,yy1,pointer,1);
                }
            }else
```

```
{
        if(pointer <count)
        {
                outtextcolor(xx1,yy1,pointer,0);
                yy1+=15;
                pointer++;
                outtextcolor(xx1,yy1,pointer,1);
        }
}clrfilename();
setcolor(1);
outtextxy(consx1+5,consy2-15,walknode->data);
ltoa(walknode->size,value,10);
outtextxy(consx1+120,consy2-15,value);
if(pointer+(count%10)-1<count)
{
        if((count%10)==0)
        {
                if(pointer+(count%10)+10<=count)
                        ftriangle(consx1+222,consy2-
                        20,13);
                else
                        ftriangle(consx1+222,consy2-
                        20,15);
        }else
                ftriangle(consx1+222,consy2-20,13);
}else
        ftriangle(consx1+222,consy2-20,15);
if(pointer-10>=1)
        btriangle(consx1+222,consy2-20,13);
else
```

```
                    btriangle(consx1+222,consy2-20,15);
                break;
case UP : if(yy1<=consyy1)
        {
                if(xx1>=consxx1+120)
                {
                        outtextcolor(xx1,yy1,pointer,0);
                        yy1=consyy1+135;xx1-=120;
                        pointer--;
                        outtextcolor(xx1,yy1,pointer,1);
                }else
                {
                        if(pointer>1)
                        {
                                xx1=consxx1;temp=yy1;
                                yy1=consyy1;
                                pointcount--;
                                nshiftdisplay(xx1,yy1,count,
                                pointcount);
                                pointer--;
                                yy1=consyy1+135;
                                outtextcolor(xx1,yy1,pointer,1);
                        }
                }
        }else
        {
                if(yy1>consyy1)
                {
                        outtextcolor(xx1,yy1,pointer,0);
                        yy1-=15;
```

```
                        pointer--;
                        outtextcolor(xx1,yy1,pointer,1);
                }
        }clrfilename();
        setcolor(1);
        outtextxy(consx1+5,consy2-15,walknode->data);
        ltoa(walknode->size,value,10);
        outtextxy(consx1+120,consy2-15,value);
        if(pointer+(count%10)-1<count)
        {
                if((count%10)==0)
                {
                        if(pointer+(count%10)+10<=count)
                                ftriangle(consx1+222,consy2-20,13);
                        else
                                ftriangle(consx1+222,consy2-20,15);
                }else
                        ftriangle(consx1+222,consy2-20,13);
        }else
                ftriangle(consx1+222,consy2-20,15);
        if(pointer-10>=1)
                btriangle(consx1+222,consy2-20,13);
        else
                btriangle(consx1+222,consy2-20,15);
        break;
    }
}while(key!='\t' && key!=27 && key!=13);
if(strcmp(value,"0")==0 && key!=27 && key!='\t')
        chdir(walknode->data);
```

```
            hidemouse();
        }while(strcmp(value,"0")==0 && key!='\t' && key!=27 && (key!=RETURN ||
        mouseindex!=2));
        hidemouse();
        outtextcolor(xx1,yy1,pointer,0);
        strcpy(bufname,walknode->data);
        freemem();
        if(key=='\t')
        {
            mouseindex++;strcpy(value,"1");
            return(bufname);
        }
        if(key==RETURN)
        {
            strcat(bufname,buf);
            return(bufname);
        }
        if(key==ESC)    return("EOF");
}

char *OPENfile(int subos)
{
        char select=0,*subfilename="",*bufclear,textbottom[2][5]={"Open","Save"};
        int caseselect=0,ini=0,point,i,xmouse,ymouse,mouse=0;
        union REGS iregs;
        showmouse();
        subfilename=(char *)malloc(13);
        bufclear=(char *)malloc(13);
        subfilename=selectfile(ini,"*.pcb",subos);
        while(strcmp(subfilename,"EOF")!=0 && strcmp(bufclear,"\v")!=0)
```

```
{
        mouse=0;
        switch(caseselect)
        {
                case 0 : hidemouse();
                         boxbars(consx1+5,consy2-225,consx1+200,consy2-205);
                         showmouse();
                         subfilename=Openfile();
                         setfillstyle(1,7);
                         bar(consx1+3,consy2-227,consx1+202,consy2-203);
                         setfillstyle(1,11);
                         bar(consx1+5,consy2-225,consx1+200,consy2-205);
                         if(mouseindex==1)
                         {       caseselect=1;mouse=1;showmouse();       }
                         if(mouseindex==2)
                         {       caseselect=2;mouse=1;showmouse();       }
                         if(mouseindex==3)
                         {       caseselect=3;mouse=1;showmouse();       }
                         if(mouse==0)    select='\t';
                         showmouse();
                        break;
                case 1 : if(strcmp(subfilename,"")==0)
                                 strcpy(subfilename,"*.pcb");
                         subfilename=selectfile(1,subfilename,subos);
                         showmouse();
                         if(subfilename==0)
                                 outtextxy(consx1+5,consy2-174,"No file");
                         clrbigbox(consx1,consy1,consx2,consy2,1);
                         if(mouseindex==0)
                         {       caseselect=0;mouse=1;showmouse();       }
```

```
        if(mouseindex==2)
        {       caseselect=2;mouse=1;showmouse();       }
        if(mouseindex==3)
        {       caseselect=3;mouse=1;showmouse();       }
        if(mouse==0)    select='\t';
        showmouse();break;
case 2 : hidemouse();
        setcolor(15);outtextxy(consx1+262,consy2-174,textbottom
        [subos]);
        showmouse();
        do
        {
                select=getnw();
                iregs.x.ax=3;
                int86(0x33,&iregs,&iregs);
                xmouse=iregs.x.cx;
                ymouse=iregs.x.dx;
                showmouse();
        }while(select=='\x0'&& !(iregs.x.bx&1));
        hidemouse();
        if(iregs.x.bx&1 && xmouse>157 && xmouse<347 &&
        ymouse>151 && ymouse<165)
        {
                caseselect=0;mouseindex=0;
        }
        if(iregs.x.bx&1 && xmouse>152 && xmouse<385 &&
        ymouse>190 && ymouse<151+345)
        {
                caseselect=1;mouseindex=1;
        }
```

```
if(iregs.x.bx&1 && xmouse>400 && xmouse<455 &&
ymouse>213 && ymouse<231)
{
        hidemouse();return("");
}
 switch(select)
 {
        case RETURN : hidemouse();return(subfilename);
        case 27 : hidemouse();return("");
        case 80 : caseselect=3;mouseindex++;break;
 }
setcolor(8);outtextxy(consx1+262,consy2-174,textbottom
[subos]);
 break;
case 3 : hidemouse();
setcolor(15);outtextxy(consx1+255,consy2-148,"Cancel");
 showmouse();
 do
 {
        select=getnw();
        iregs.x.ax=3;
        int86(0x33,&iregs,&iregs);
        xmouse=iregs.x.cx;
        ymouse=iregs.x.dx;
        showmouse();
 }while(select=='\x0'&& !(iregs.x.bx&1));
if(iregs.x.bx&1 && xmouse>157 && xmouse<347
&& ymouse>151 && ymouse<165)
{
        caseselect=0;mouseindex=0;hidemouse();
```

```
                    }
                    if(iregs.x.bx&1 && xmouse>152 && xmouse<385
                    && ymouse>190 && ymouse<151+345)
                    {
                            caseselect=1;mouseindex=1;hidemouse();
                    }
                    if(iregs.x.bx&1 && xmouse>400 && xmouse<455
                    && ymouse>190 && ymouse<208)
                    {
                            caseselect=2;mouseindex=2;hidemouse();
                    }
                    if(mouse==1 && caseselect==3)           { return(""); }
                    if(select==13 || select==27)        { hidemouse();return(""); }
                    if(select==72)   { caseselect=2;mouseindex--; }
                    setcolor(8);outtextxy(consx1+255,consy2-148,"Cancel");
                    hidemouse();
                    break;
            }
            point=(strlen(subfilename)-2)+1;
            strcpy(bufclear,subfilename);
            for(i=0;i<point;i++)
                    *bufclear++;
            if(strcmp(subfilename,"EOF")!=0 && select=='\t')
            {
                    caseselect++;mouseindex++;
            }
            if(caseselect==4)           { caseselect=0;mouseindex=0; }
            select=0;
    }
    if(strcmp(bufclear,"\v")==0)
```

```
        {
                subfilename+=(point);
                *subfilename='\0';
                subfilename-=(point);
        }
        hidemouse();
        return(subfilename);
        free(subfilename);
        free(bufclear);
}
char *Ofile(int os)
{
        char i,rname[13];
        sizepic=imagesize(consx1-2,consy1-2,consx2,consy2);
        savepic=malloc(sizepic);
        getimage(consx1-2,consy1-2,consx2,consy2,savepic);
        bufname=(char *)malloc(15);
        showmouse();
        strcpy(bufname,OPENfile(os));
        if(strcmp(bufname,"EOF")==0)    strcpy(bufname,"");
        putimage(consx1-2,consy1-2,savepic,COPY_PUT);
        setlinestyle(0,0,1);
        free(savepic);
        if(strcmp(value,"0")==0)            return("");
        return(bufname);
}
```

ไฟล์ remote.c

```
void showpoint(int x1,int y1,char attrib)
```

```
{
        if(attrib==1)
        {       setcolor(4);
                setfillstyle(1,4);
                pieslice(x1,y1,0,360,3);
                setcolor(0);
                setfillstyle(1,0);
                pieslice(x1,y1,0,360,1);
        }
}
void remote()
{
        char ch;
        int pointx=40,pointy=400;
        char strx[10],stry[10];
        unsigned pointsize;
        void *savepoint;
        pointsize=imagesize(pointx-5,pointy-5,pointx+5,pointy+5);
        savepoint=malloc(pointsize);
        getimage(pointx-5,pointy-5,pointx+5,pointy+5,savepoint);
        showpoint(pointx,pointy,1);
        bottomdown(176,455,276,472,2,7); //apd
        outtextxy(176+50-24,460,"REMOTE");
        do
        {
                do
                {
                        ch=getch();
                        switch(ch)
                        {case UP :      if(pointy>50)
```

```
                {
                        putimage(pointx-5,pointy-
                        5,savepoint,COPY_PUT);
                        pointy--;
                        getimage(pointx-5,pointy-
                        5,pointx+5,pointy+5,savepoint);
                        showpoint(pointx,pointy,1);
                }break;
case DOWN :     if(pointy<400)
                {
                        putimage(pointx-5,pointy-
                        5,savepoint,COPY_PUT);
                        pointy++;
                        getimage(pointx-5,pointy-
                        5,pointx+5,pointy+5,
                        savepoint);
                        showpoint(pointx,pointy,1);
                }break;
case LEFT :     if(pointx>40)
                {
                        putimage(pointx-5,pointy-
                        5,savepoint,COPY_PUT);
                        pointx--;
                        getimage(pointx-5,pointy-
                        5,pointx+5,pointy+5,savepoint);
                        showpoint(pointx,pointy,1);
                }break;
case RIGHT :    if(pointx<600)
                {
```

```
                                putimage(pointx-5,pointy-
                                5,savepoint,COPY_PUT);
                                pointx++;
                                getimage(pointx-5,pointy-
                                5,pointx+5,pointy+5,savepoint);
                                showpoint(pointx,pointy,1);
                        }break;
                }
                setcolor(7);
                setfillstyle(1,7);
                bar(40,460,90,470);//x
                bar(122,460,172,470); //y
                sprintf(strx,"X: %04d",pointx);
                sprintf(stry,"Y: %04d",pointy);
                setcolor(0);
                outtextxy(20,460,strx);
                outtextxy(102,460,stry);
        }while(ch!=ESC && ch!=RETURN);
        if(ch==RETURN)          { }
    }while(ch!=ESC);
    putimage(pointx-5,pointy-5,savepoint,COPY_PUT);
    free(savepoint);
    bottomdown(176,455,276,472,2,7);
    bottomdown(13,455,93,472,2,7);
    bottomdown(95,455,175,472,2,7);
    outtextxy(16,460,"X: 0000");
    outtextxy(98,460,"Y: 0000");
    outtextxy(176+50-12,460,"APD");
}
```

ไฟล์ kbdio.c

```
#define AT          0X40
#define UP          0X48
#define DOWN        0X50
#define RIGHT       0X4D
#define LEFT        0X4B
#define SPACE       0X20
#define RETURN      0X0D
#define TAB         0X09
#define ESC         0X1B
#define DEL         0X53
#define PGDN        0X51
#define PGUP        0X49
#define ALT_A       0X1E
#define ALT_B       0X30
#define ALT_C       0X2E
#define ALT_D       0X20
#define ALT_E       0X12
#define ALT_F       0X21
#define ALT_G       0X22
#define ALT_H       0X23
#define ALT_I       0x17
#define ALT_J       0x24
#define ALT_K       0x25
#define ALT_L       0x26
#define ALT_M       0X32
#define ALT_N       0x31
#define ALT_O       0X18
#define ALT_P       0x19
```

```
#define ALT_Q       0X10
#define ALT_R       0X13
#define ALT_S       0X1F
#define ALT_T       0X14
#define ALT_U       0X16
#define ALT_V       0X2F
#define ALT_W       0X11
#define ALT_X       0X2D
#define ALT_Y       0X15
#define ALT_Z       0X2C
#define CTRL_A      0X01
#define CTRL_B      0X02
#define CTRL_C      0X03
#define CTRL_D      0X04
#define CTRL_E      0X05
#define CTRL_F      0X06
#define CTRL_G      0X07
#define CTRL_H      0X08
#define CTRL_I      0X09
#define CTRL_J      0X0A
#define CTRL_K      0X0B
#define CTRL_L      0X0C
#define CTRL_M      0X0D
#define CTRL_N      0X0E
#define CTRL_O      0X0F
#define CTRL_P      0X10
#define CTRL_Q      0X11
#define CTRL_R      0X12
#define CTRL_S      0X13
#define CTRL_T      0X14
```

```
#define CTRL_U      0X15
#define CTRL_V      0X16
#define CTRL_W      0X17
#define CTRL_X      0X18
#define CTRL_Y      0X19
#define CTRL_Z      0X1A
#define F1          0X3B
#define F2          0X3C
#define F3          0X3D
#define F4          0X3E
#define F5          0X3F
#define F6          0X40
#define F7          0X41
#define F8          0X42
#define F9          0X43
#define F10         0X44
#define F11         0X85
#define F12         0X86
```

ไฟล์ showtitl.c

```
void SetBMPColor(void);
void PrintBMP(int xx,int yy);
void ShowBMP(void);

struct BMPHEAD
{
        char id[2];
        long filesize;
        int reserved[2];
```

```
        long headersize;
        long infosize;
        long width;
        long depth;
        int biplanes;
        int bits;
        long bicompression;
        long bisizeimage;
        long bixpelspermeter;
        long biypelspermeter;
        long bicirused;
        long bicirimportant;
};
struct BMPHEAD bmphead;
struct palettetype palette;
FILE *fbmp;

void ShowBMP(void)
{
        int x,y;
        if((fbmp=fopen("..\\winlogo.bmp","rb"))==0)
        {
                printf("Error Can't open file graphics");
                exit(0);
        }
        fseek(fbmp,0,SEEK_SET);
        fread(&bmphead,sizeof(bmphead),1,fbmp);
        x=320 -(bmphead.width/2);
        y=240 - (bmphead.depth/2);
        SetBMPColor();
```

```
        PrintBMP(x,y);
        fclose(fbmp);
}

void SetBMPColor(void)
{
        int i=0;
        int r,g,b;
        fseek(fbmp,sizeof(bmphead),SEEK_SET);
        while((ftell(fbmp)<=bmphead.headersize)&&(i<16))
        {
                r=getc(fbmp)>>2;
                g=getc(fbmp)>>2;
                b=getc(fbmp)>>2;
                getc(fbmp);
                setrgbpalette(i,b,g,r);
                palette.colors[i]=i;
                i++;
        }
        palette.size=16;
        setallpalette(&palette);
}

void PrintBMP(int xx,int yy)
{
        char ch,color;
        int x,y,i=0;
        int szline;
        szline=((bmphead.width+7)/8)*8;
        xx=xx;
```

```
        yy=yy+bmphead.depth;
        fseek(fbmp,bmphead.headersize,SEEK_SET);
        for(y=0;y<bmphead.depth;y++)
        {
                for(x=0;x<(szline);x++)
                {
                        if(i==0)
                        {
                                ch=getc(fbmp);
                                color=ch>>4;
                                i=1;
                        }else
                        {
                                color=(ch & 0x0f);
                                i=0;
                        }if(x<bmphead.width)
                                putpixel(xx+x,yy-y,color);
                }i=0;
        }
}

showtitle()
{
        int countdelay=0;
        ShowBMP();
        do
        {
                countdelay++;
                delay(200);
        }while(!kbhit() && countdelay<40);
```

```
        setallpalette(getdefaultpalette());
        cleardevice();
        closegraph();
}
```

ไฟล์ printcir

```
void printbyte(char ch)
{
        biosprint(0 ,ch ,0);
}
char getpoint (int x, int y)
{
        union REGS r;
        r.h.ah=13;
        r.x.dx=y;
        r.x.cx=x;
        r.h.bh=0;
        int86(16,&r,&r);
        return r.h.al;
}

void set_graphic (int cols,int density)
{
        union
        {       unsigned char c[2];
                unsigned int i;
        } u;
        char den_code;
        u.i=cols;
```

```
        switch(density)
        {       case 0 :  den_code =75;break;
                case 1:   den_code =76;break;
                case 2 :  den_code =90;break;
        }
        printbyte(27); printbyte(65); printbyte(8);
        printbyte(27); printbyte(den_code); printbyte(u.c[0]); printbyte(u.c[1]);
}
void print_scr(int startx,int starty,int endx,int endy,int pagex,int pagey,int density)
{
        register int i,x,y,px;
        int cols,color,sum;
        printbyte(27);printbyte(108);printbyte(5);
        endx++; endy++;
        cols=endx-startx;
        for(; pagey>=0; pagey--)  printbyte('\n');
        for(y = starty; y<endy; y+=8)
        {
                for(px=0; px<pagex; px++)  printbyte(' ');
                set_graphic(cols, density);
                for(x = startx; x<endx; x++)
                {
                        sum = 0;
                        for(i=0; i<8; i++)
                        {
                                if(y+i < endy)
                                {
                                        color = getpoint(x, y+i);
                                        if(color!=8)    sum+=1<<(7-i);
                                }
```

```
                }
                printbyte(sum);
            }
            printbyte('\n');
        }
    }
    void reset (void)
    {
        printbyte(27); printbyte(64);
    }
```

ไฟล์ changd.c

```
#ifdef __cplusplus
    #define __CPPARGS ...
#else
    #define __CPPARGS
#endif
int failcheck=0;
setnewint(void);
setoldint(void);
char chkdsk(int disk);
int boxerror(int subcheckmouse);
void changedrv(int subcheckmouse);
void interrupt(*Oldint)(__CPPARGS);
void interrupt Myint(__CPPARGS)
{
    failcheck=1;
}
```

```
setnewint(void)
{
        Oldint=getvect(0x24);
        setvect(0x24,Myint);
}

setoldint(void)
{
        setvect(0x24,Oldint);
}

void bottomup(xx1,yy1,xx2,yy2,tic,c1)
{
        int i;
        char c;
        c=getcolor();
        setfillstyle(1,c1);
        bar(xx1,yy1,xx2,yy2);
        for(i=0;i<tic;i++)
        {
                setcolor(15);
                line(xx1+i,yy1+i,xx1+i,yy2-i);
                line(xx1+i,yy1+i,xx2-i,yy1+i);
                setcolor(8);
                line(xx2-i,yy1+i,xx2-i,yy2-i);
                line(xx1+i,yy2-i,xx2-i,yy2-i);
        }
        setcolor(c);
}
```

```
void bottomdown(xx1,yy1,xx2,yy2,tic)
{
        int i;
        char c;
        c=getcolor();
        setfillstyle(1,7);
        bar(xx1,yy1,xx2,yy2);
        for(i=0;i<tic;i++)
        {
                setcolor(8);
                line(xx1+i,yy1+i,xx1+i,yy2-i);
                line(xx1+i,yy1+i,xx2-i,yy1+i);
                setcolor(15);
                line(xx2-i,yy1+i,xx2-i,yy2-i);
                line(xx1+i,yy2-i,xx2-i,yy2-i);
        }
        setcolor(c);
}

char chkdsk(int disk)
{
        _AH=0x36;
        _DL=disk;
        geninterrupt(0x21);
        return(_AX);
}

int countdrv()
{
        int i=0,drv=0;
```

```
        do
        {
                i=drv;
                _AH=8;
                _DL=i;
                geninterrupt(0x13);
                drv++;i++;
        }while(_BL!=0);
        return(i);
}

int boxerror(int subcheckmouse)
{
        int x1=210,y1=200,i=0,xmouse,ymouse;
        char textcheck[2][7]={"Retry","Cancel"};
        char keycheck=0;
        union REGS regs;
        void *saveboxerror;
        unsigned sizeboxerror;
        sizeboxerror=imagesize(x1,y1,x1+250,y1+100);
        saveboxerror=malloc(sizeboxerror);
        getimage(x1,y1,x1+250,y1+100,saveboxerror);
        setfillstyle(1,1);
        bar(x1,y1,x1+250,y1+100);
        setcolor(15);
        rectangle(x1+2,y1+2,x1+248,y1+98);
        rectangle(x1+4,y1+4,x1+246,y1+96);
        setcolor(4);
        outtextxy(x1+25,y1+15,"Insert disk or disk error");
        bottomup(x1+30,y1+50,x1+85,y1+70,1,7);
```

```
bottomup(x1+165,y1+50,x1+220,y1+70,1,7);
setcolor(15);
outtextxy(x1+36,y1+56,"Retry");
setcolor(0);
outtextxy(x1+168,y1+56,"Cancel");
showmouse(subcheckmouse);
do
{
        keycheck=getnw();
        regs.x.ax=3;
        int86(0x33,&regs,&regs);
        xmouse=regs.x.cx;
        ymouse=regs.x.dx;
        if(regs.x.bx&1 && xmouse>x1+30 && xmouse<x1+85 && ymouse>y1+50
          && ymouse <y1+70)
        {
                i=0;
                keycheck=RETURN;
        }
        if(regs.x.bx&1 && xmouse>x1+165 && xmouse<x1+220 && ymouse>y1+50
          && ymouse <y1+70)
        {
                i=1;
                keycheck=RETURN;
        }
        switch(keycheck)
        {
                case LEFT : setcolor(0);
                                        outtextxy(x1+36+(i*132),y1+56,textcheck[i]);
                                        i--;
```

```
                                        if(i<0)i=1;
                                        setcolor(15);
                                        outtextxy(x1+36+(i*132),y1+56,textcheck[i]);
                                        break;
                        case TAB        :
                        case RIGHT :setcolor(0);
                                        outtextxy(x1+36+(i*132),y1+56,textcheck[i]);
                                        i++;
                                        if(i>1)i=0;
                                        setcolor(15);
                                        outtextxy(x1+36+(i*132),y1+56,textcheck[i]);
                                        break;
                }
        }while(keycheck!=RETURN && keycheck!=ESC);
        hidemouse(subcheckmouse);
        putimage(x1,y1,saveboxerror,COPY_PUT);
        free(saveboxerror);
        return(i);
}

void changedrv(int subcheckmouse)
{
        int disk[10],
maxdrive=0,tempmaxdrive,i=2,j=0,count=0,initd,x,y,casesel,totaldrv,bufinitd;
        int xmouse,ymouse,countmouse=0;
        char bufdisk[1],ch=0,checkdrv,bufnamedisk[10];
        void *savepopchanged;
        unsigned sizechanged;
        union REGS regs;
        initd=getdisk();
```

```
bufinitd=initd;
setnewint();
setdisk(0);
totaldrv=countdrv();
for(j=0;j<totaldrv;j++)
{
        bufnamedisk[j]=getdisk()+'A';
        maxdrive++;
        setdisk(j);
}
if(initd>1)
{
        bufinitd=initd;
        initd-=maxdrive;
}
setdisk(i);
do
{
        bufnamedisk[maxdrive]=getdisk()+'A';
        i++;maxdrive++;
        setdisk(i);
}while(bufnamedisk[maxdrive-1]!=bufnamedisk[maxdrive-2]);
maxdrive--;
x=((640/2)-(maxdrive*30)/2);
y=(480/2)-25;
sizechanged=imagesize(x,y,x+(maxdrive*30),y+25);
savepopchanged=malloc(sizechanged);
getimage(x,y,x+(maxdrive*30),y+25,savepopchanged);
setcolor(7);
setfillstyle(1,11);
```

```
bar(x,y,x+(maxdrive*30),y+25);
setcolor(15);
rectangle(x+2,y+2,x+(maxdrive*30)-2,y+23);
setcolor(0);
bufdisk[1]='\0';
for(i=0;i<maxdrive;i++)
{
        bufdisk[0]=bufnamedisk[i];
        bottomup(x+(i*30)+4,y+5,x+(i*30)+25,y+20,1,7);
        outtextxy(x+(i*30)+11,y+10,bufdisk);
}
setcolor(2);
count=initd;
bottomdown(x+(count*30)+4,y+5,x+(count*30)+25,y+20,1,7);
setcolor(0);
bufdisk[0]=bufnamedisk[count];;
outtextxy(x+(count*30)+11,y+10,bufdisk);
showmouse(subcheckmouse);
delay(50);
do
{
        if(regs.x.bx&1) delay(175);
        _AX=5;
        _BX=0;
        geninterrupt(0x33);
        do
        {
                ch=getnw();
                regs.x.ax=3;
                int86(0x33,&regs,&regs);
```

```
            xmouse=regs.x.cx;
            ymouse=regs.x.dx;
        }while(ch=='\x0' && !(regs.x.bx&1));
        if(regs.x.bx&1) delay(175);
        _AX=5;
        _BX=0;
        geninterrupt(0x33);
        countmouse=_BX;
        if(countmouse>1)
            ch=RETURN;
        if(regs.x.bx&1)
        {
            for(i=0;i<maxdrive;i++)
            {

                if(xmouse>x+(i*30)+4&&xmouse<x+(i*30)+25
                  &&ymouse>y+5 && ymouse<y+20)
                {
                    hidemouse(subcheckmouse);
                    bottomup(x+(count*30)+4,y+5,x+(count*30)+25,
                    y+20,1,7);
                    bufdisk[0]=bufnamedisk[count];;
                    setcolor(0);
                    outtextxy(x+(count*30)+11,y+10,bufdisk);
                    count=i;
                    if(count>=maxdrive)count=0;
                    bottomdown(x+(count*30)+4,y+5,x+count*30)+25,
                    y+20,1,7);
                    setcolor(0);
                    bufdisk[0]=bufnamedisk[count];;
```

```
                        outtextxy(x+(count*30)+11,y+10,bufdisk);
                        showmouse(subcheckmouse);
                }
        }
        if(xmouse<x||xmouse>x+(maxdrive*30)+30||ymouse<y
        ||ymouse>y+25)
                ch=ESC;
}
switch(ch)
{
        case RIGHT      :       bottomup(x+(count*30)+4,y+5,
                                x+(count*30)+25,y+20,1,7);
                                bufdisk[0]=bufnamedisk[count];;
                                setcolor(0);
                                outtextxy(x+(count*30)+11,y+10,bufdisk);
                                count++;
                                if(count>=maxdrive)count=0;
                                bottomdown(x+(count*30)+4,y+5,
                                x+(count*30)+25,y+20,1,7);
                                setcolor(0);
                                bufdisk[0]=bufnamedisk[count];;
                                outtextxy(x+(count*30)+11,y+10,bufdisk);
                                break;
        case LEFT       :       bottomup(x+(count*30)+4,y+5,x+
                                (count*30)+25,y+20,1,7);
                                bufdisk[0]=bufnamedisk[count];;
                                bufdisk[1]='\0';
                                setcolor(0);
                                outtextxy(x+(count*30)+11,y+10,bufdisk);
                                count--;
```

```
                                        if(count<0)count=maxdrive-1;
                                        bottomdown(x+(count*30)+4,y+5,x+
                                        (count*30)+25,y+20,1,7);
                                        setcolor(0);
                                        bufdisk[0]=bufnamedisk[count];;
                                        outtextxy(x+(count*30)+11,y+10,bufdisk);
                                        break;
        }
}while(ch!=ESC && ch!=RETURN);
if(ch==RETURN)
{
        if(bufnamedisk[count]-'A'<2) chkdsk(bufnamedisk[count]-'A'-1);
        if(failcheck==1)
        {
                do
                {
                        if(failcheck==1)
                        {
                                failcheck=0;
                                casesel=boxerror(subcheckmouse);
                        }
                        if(casesel==0) chkdsk(count+1);
                }while(failcheck==1);
                if(casesel==0) setdisk(bufnamedisk[count]-'A');
                else    setdisk(bufinitd);
        }
        else
                setdisk(bufnamedisk[count]-'A');
}
else
```

```
            setdisk(bufinitd);
        hidemouse(subcheckmouse);
        putimage(x,y,savepopchanged,COPY_PUT);
        free(savepopchanged);
        setoldint();
}
```

ไฟล์ mouse.c

```
#include <dos.h>
int mousedrv()
{
        union REGS regs;
        regs.x.ax=0;
        int86(0x33,&regs,&regs);
        return(_AX);
}

int showmouse(int checkmouse)
{
        union REGS regs;
        if(!checkmouse)return(0);
        regs.x.ax=1;
        int86(0x33,&regs,&regs);
}

int hidemouse(int checkmouse)
{
        union REGS regs;
        if(!checkmouse)return(0);
        regs.x.ax=2;
```

```
        int86(0x33,&regs,&regs);
}

void setspeedmouse(int speed)
{
        union REGS regs;
        regs.x.ax=15;
        regs.x.cx=speed/2;
        regs.x.dx=speed;
        int86(0x33,&regs,&regs);
}
```

ไฟล์ Menu.c

```
/* ============== Program Autometic Driller ============== */
/*  program name : menu.c                                   */
/* ========================================================= */
#include<graphics.h>
#include<conio.h>
#include<stdlib.h>
#include<stdio.h>
#include<dos.h>
#include<bios.h>
#include "c:\tee\project\kbdio.c"
#include "c:\tee\project\readfile.c"
#include "c:\tee\project\changed.c"
//#include "c:\tee\mouse.c"
#include "c:\tee\project\about.c"
#include "c:\tee\project\remouat.c"
//#include "c:\tee\help.c"
//#include "c:\tee\showtitl.c"
```

```
#include "c:\tee\printcir.c"
#define COM1 0x3f8
#define COM2 0x2f8
#define COM1check 0x3fd
#define COM2check 0x2fd
extern int mousedrv();
extern int showmouse(int checkmouse);
extern int hidemouse(int checkmouse);
extern void backgroundhelp(int x1,int y1,int x2,int y2,int subcheckmouse);
extern movecursor(int x1,int y1,int x2,int y2);
extern help_file(int x1,int y1,int x2,int y2,int subcheckmouse);
extern help_mode(int x1,int y1,int x2,int y2,int subcheckmouse);
extern help_port(int x1,int y1,int x2,int y2,int subcheckmouse);
extern help_print(int x1,int y1,int x2,int y2,int subcheckmouse);
extern help_help(int x1,int y1,int x2,int y2,int subcheckmouse);
extern void pro_help(int x1,int y1,int x2,int y2,int subcheckmouse);
extern void SetBMPColor(void);
extern void PrintBMP(int xx,int yy);
extern void ShowBMP(void);
extern void showtitle(void);
FILE *pcbfile,*savepcb;
char file[6][11]={"New","Open","Save","Change Drv","Dos","Quit"};
char mode[3][8]={"Auto","Manual","Remote"};
char port[2][9]={"Port","Setting"};
char portcom[2][5]={"Com1","Com2"};
char view[2][10]={"Zoom out","Zoom in"};
char print[2][8]={"Locate","Circuit"};
char help[2][6]={"Help","About"};
char menu_main[6][6]={"File","Mode","Port","view","Print","Help"};
char *filename="";
```

```
int  comport=0,bx=0,by=0,p=0,d=0,s=0,checksetport=0,pointdrl=0;
void *savepopmenu,*savepopmenu2;
unsigned menusize,menusize2;
int countcp=0,countct=0,countca=0,countft=0,countfp=0,checkmouse=1;
int maxx=0,minx=32000,maxy=0,miny=32000,checkfiledrl=0,checkfilepcb=0;
int *cp,*ct,*ca,*ft,*fp,*datadrill[2],portselect=COM1,portselectcheck=COM1check;
void countpcb(void);
readfile(char *namefile);
void sortfile(void);
void about(int x1,int y1,int x2,int y2,int subcheckmouse);
int date_report(void);
int time_report(void);
void countpcb(void)
{
        char ch;
        countcp=0;countct=0;countca=0;countft=0;countfp=0;
        maxx=0;minx=32000;maxy=0;miny=32000;
        do{
                ch=fgetc(pcbfile);
                switch(ch)
                {       case 'C': ch=fgetc(pcbfile);
                                  if(ch=='P')
                                          countcp++;
                                  if(ch=='T')
                                          countct++;
                                  if(ch=='A')
                                          countca++;
                                  break;
                        case 'F': ch=fgetc(pcbfile);
                                  if(ch=='T')
```

```
                                countft++;
                            if(ch=='P')
                                countfp++;
                            break;
                    }
            }while(ch!=EOF);
}
int readpcbfile(char *namefile)
{
        int x=0,y=0,i=0,j=0;
        char buf[5],ch;
        if((pcbfile=fopen(namefile,"r"))==0)
                return(0);
        checkfilepcb=1;
        countpcb();
        fseek(pcbfile,0l,SEEK_SET);
        cp=(int *)malloc(countcp*16);
        ct=(int *)malloc(countct*12);
        ca=(int *)malloc(countca*12);
        ft=(int *)malloc(countft*12);
        fp=(int *)malloc(countfp*16);
        do{
                ch=fgetc(pcbfile);
                switch(ch)
                {
                        case 'C' : ch=fgetc(pcbfile);
                                if(ch=='P')
                                {
                                        fgetc(pcbfile);
                                        do{
```

```
                do{
                        ch=fgetc(pcbfile);
                        buf[i]=ch;
                        i++;
                }while(ch!=' ' && ch!='\n' );
                *cp=atoi(buf);
                if(j==0)
                {
                        if(*cp>maxx)    maxx=*cp;
                        if(*cp<minx)    minx=*cp;
                }
                if(j==1)
                {
                        if(*cp>maxy)    maxy=*cp;
                        if(*cp<miny)    miny=*cp;
                }
                i=0;cp++;j++;
        }while(ch!='\n');
        j=0;
}
if(ch=='T')
{
        fgetc(pcbfile);
        do{
                do{
                        ch=fgetc(pcbfile);
                        buf[i]=ch;
                        i++;
                }while(ch!=' ' && ch!='\n' );
                *ct=atoi(buf);
```

```
                if(j==0)
                {
                        if(*ct>maxx) maxx=*ct;
                        if(*ct<minx) minx=*ct;
                }
                if(j==1)
                {
                        if(*ct>maxy) maxy=*ct;
                        if(*ct<miny) miny=*ct;
                }
                i=0;ct++;j++;
        }while(ch!='\n');
        j=0;
}
if(ch=='A')
{
        fgetc(pcbfile);
        do{
                do{
                        ch=fgetc(pcbfile);
                        buf[i]=ch;
                        i++;
                }while(ch!=' ' && ch!='\n' );
                *ca=atoi(buf);
                if(j==0)
                {
                        if(*ca>maxx) maxx=*ca;
                        if(*ca<minx) minx=*ca;
                }
                if(j==1)
```

```
                {
                        if(*ca>maxy) maxy=*ca;
                        if(*ca<miny) miny=*ca;
                }
                i=0;*ca++;j++;
        }while(ch!='\n');
        j=0;
    }break;
case 'F' : ch=fgetc(pcbfile);
    if(ch=='T')
    {
        fgetc(pcbfile);
        do{
            do{
                ch=fgetc(pcbfile);
                buf[i]=ch;
                i++;
            }while(ch!=' ' && ch!='\n' );
            *ft=atoi(buf);
            if(j==0)
            {
                if(*ft>maxx) maxx=*ft;
                if(*ft<minx) minx=*ft;
            }
            if(j==1)
            {
                if(*ft>maxy) maxy=*ft;
                if(*ft<miny) miny=*ft;
            }
            i=0;ft++;j++;
```

```
                }while(ch!='\n');
                j=0;
            }
            if(ch=='P')
            {
                fgetc(pcbfile);
                do{
                    do{
                        ch=fgetc(pcbfile);
                        buf[i]=ch;
                        i++;
                    }while(ch!=' ' && ch!='\n' );
                    *fp=atoi(buf);
                    if(j==0)
                    {
                        if(*fp>maxx) maxx=*fp;
                        if(*fp<minx) minx=*fp;
                    }
                    if(j==1)
                    {
                        if(*fp>maxy) maxy=*fp;
                        if(*fp<miny) miny=*fp;
                    }
                    i=0;fp++;j++;
                }while(ch!='\n');
                j=0;
            }
        }
    }while(ch!=EOF);
    fclose(pcbfile);
```

```
        return(1);
}

void showpcb(int mod)
{
        int x1,y1,x2,y2,i,j,r;
        float xf=50.0,yf=66.66,rf=124;
        cp-=(countcp*8);
        ct-=(countct*6);
        ca-=(countca*6);
        ft-=(countft*6);
        fp-=(countfp*8);
        xf/=mod;
        yf/=mod;
        rf/=mod;
        setcolor(4);
        setfillstyle(1,4);
        for(i=0;i<countcp;i++)
        {
                x1=40+(((*cp-minx)/xf));
                cp++;
                y1=400-(((*cp-miny)/yf));
                cp++;
                r=(*cp/rf);
                if(r<2) r=1;
                pieslice(x1,y1,0,360,r);
                cp+=6;
        }
        for(i=0;i<countfp;i++)
        {
```

```
        x1=40+(((*fp-minx)/xf));
        fp++;
        y1=400-(((*fp-miny)/yf));
        fp++;
        r=(*fp/rf);
        if(r<2) r=1;
        pieslice(x1,y1,0,360,r);
        fp+=6;
}
setcolor(12);
for(i=0;i<countft;i++)
{
        x1=40+(((*ft-minx)/xf));
        ft++;
        y1=400-(((*ft-miny)/yf));
        ft++;
        x2=40+(((*ft-minx)/xf));
        ft++;
        y2=400-(((*ft-miny)/yf));
        ft++;
        r=(*ft/rf);
        ft++;
        if(*ft==1)          setcolor(12);
        else if(*ft==6) setcolor(13);
        else
                setcolor(14);
        line(x1,y1,x2,y2);
        ft++;
}
setcolor(14);
```

```
for(i=0;i<countct;i++)
{
        x1=40+(((*ct-minx)/xf));
        ct++;
        y1=400-(((*ct-miny)/yf));
        ct++;
        x2=40+(((*ct-minx)/xf));
        ct++;
        y2=400-(((*ct-miny)/yf));
        ct++;
        r=(*ct/rf);
        line(x1,y1,x2,y2);
        ct+=2;
}
setfillstyle(0,14);
for(i=0;i<countca;i++)
{
        x1=40+(((*ca-minx)/xf));
        ca++;
        y1=400-(((*ca-miny)/yf));
        ca++;
        r=(*ca/rf);
        if(r<2) r=1;
        ca++;
        switch(*ca)
        {
                case 15 : r=r*1.9;
                          circle(x1,y1,r);
                          break;
                case 3 : r=r*2.9;
```

```
                    arc(x1,y1,0,180,r);
                    break;
                case 6 : r=r*2.9;
                    arc(x1,y1,90,270,r);
                    break;
                case 9 : r=r*2.9;
                    arc(x1,y1,270,90,r);
                    break;
                case 12 : r=r*2.9;
                    arc(x1,y1,180,360,r);
                    break;
            }ca+=3;
        }
}
int opengraph(void)
{
        int driver = DETECT,mode,errorcode;
        errorcode = registerbgidriver(EGAVGA_driver);
        if(errorcode<0)
            exit(1);
        initgraph(&driver,&mode,"c:\\tc\\bgi");
}

char getnw(void)
{
        time_report();
        _AH=6;
        _DL=0xff;
        geninterrupt(0x21);
        return(_AL);
```

```
}
int date_report(void)
{
        struct date date;
        char today[15];
        getdate(&date);
        sprintf(today,"%02d-%02d-%02d",date.da_day,date.da_mon,date.da_year%100);
        setcolor(0);
        outtextxy(478,460,today);
        return(0);
}
int time_report(void)
{
        struct time time;
        char now[15];
        delay(20);
        gettime(&time);
        sprintf(now,"%02d:%02d:%02d",time.ti_hour,time.ti_min,time.ti_sec);
        setfillstyle(1,7);
        bar(559,460,625,468);
        setcolor(0);
        outtextxy(559,460,now);
        delay(20);
        return(0);
}

void background(void)
{
        struct date d;
        char dd[2],mm[2],yy[2];
```

```
opengraph();
setcolor(15);
setfillstyle(1,15);
bar(5,5,635,475);
setcolor(7);
setfillstyle(1,7);
bar(7,7,635,475);
setfillstyle(1,1);
bar(9,9,633,25);
setfillstyle(1,8);
bar(10,42,633,472);
outtextxy(195,13,"Program Autometic Drill PCB Control");
setcolor(0);
outtextxy(13,30,menu_main[0]);
line(13,39,18,39);
outtextxy(63,30,menu_main[1]);
line(63,39,68,39);
outtextxy(113,30,menu_main[2]);
line(121,39,126,39);
outtextxy(163,30,menu_main[3]);
line(163,39,168,39);
outtextxy(213,30,menu_main[4]);
line(213,39,218,39);
outtextxy(263,30,menu_main[5]);
line(263,39,268,39);
setcolor(15);
setfillstyle(1,15);
bar(10,452,633,472);
setcolor(7);
setfillstyle(1,7);
```

```
        bar(12,454,631,470);
        bottomdown(13,455,93,472,2,7);//x
        bottomdown(95,455,175,472,2,7); //y
        bottomdown(176,455,276,472,2,7); //apd
        bottomdown(469,455,549,472,2,7);
        bottomdown(550,455,630,472,2,7);
        setcolor(0);
        outtextxy(20,460,"X: 0000");
        outtextxy(102,460,"Y: 0000");
        outtextxy(176+50-12,460,"APD");
        date_report();
        time_report();
}

void menu_file(int xx,int yy)
{
        menusize=imagesize(xx-1,yy,xx+91,yy+97);
        savepopmenu=malloc(menusize);
        getimage(xx-1,yy,xx+91,yy+97,savepopmenu);
        setcolor(15);
        setfillstyle(1,15);
        bar(xx,yy,xx+90,yy+96);
        setcolor(7);
        setfillstyle(1,7);
        bar(xx+1,yy+1,xx+90,yy+96);
        setcolor(0);
        outtextxy(xx+6,yy+6,file[0]);
        outtextxy(xx+6,yy+21,file[1]);
        outtextxy(xx+6,yy+36,file[2]);
        outtextxy(xx+6,yy+51,file[3]);
```

```
        outtextxy(xx+6,yy+66,file[4]);
        outtextxy(xx+6,yy+81,file[5]);
}
void clrmenu_file(int xx,int yy)
{
        putimage(xx-1,yy,savepopmenu,COPY_PUT);
        free(savepopmenu);
}
void menu_mode(int xx,int yy)
{
        menusize=imagesize(xx-1,yy,xx+72,yy+52);
        savepopmenu=malloc(menusize);
        getimage(xx-1,yy,xx+72,yy+52,savepopmenu);
        setcolor(15);setfillstyle(1,15);
        bar(xx,yy,xx+71,yy+51);
        setcolor(7);setfillstyle(1,7);
        bar(xx+1,yy+1,xx+71,yy+51);
        setcolor(0);
        outtextxy(xx+6,yy+6,mode[0]);
        outtextxy(xx+6,yy+21,mode[1]);
        outtextxy(xx+6,yy+36,mode[2]);
}

void clrmenu_mode(int xx,int yy)
{
        putimage(xx-1,yy,savepopmenu,COPY_PUT);
        free(savepopmenu);
}

void menu_port(int xx,int yy)
```

```
{
        menusize=imagesize(xx-1,yy,xx+72,yy+37);
        savepopmenu=malloc(menusize);
        getimage(xx-1,yy,xx+72,yy+37,savepopmenu);
        setcolor(15);
        setfillstyle(1,15);
        bar(xx,yy,xx+71,yy+36);
        setcolor(7);setfillstyle(1,7);
        bar(xx+1,yy+1,xx+71,yy+36);
        setcolor(0);
        outtextxy(xx+6,yy+6,port[0]);
        outtextxy(xx+6,yy+21,port[1]);
}

void clrmenu_port(int xx,int yy)
{
        putimage(xx-1,yy,savepopmenu,COPY_PUT);
        free(savepopmenu);
}

void menu_portsub(int xx,int yy)
{
        menusize2=imagesize(xx-1,yy,xx+72,yy+37);
        savepopmenu2=malloc(menusize);
        getimage(xx-1,yy,xx+72,yy+37,savepopmenu2);
        setcolor(15);setfillstyle(1,15);
        bar(xx,yy,xx+71,yy+36);
        setcolor(7);setfillstyle(1,7);
        bar(xx+1,yy+1,xx+71,yy+36);
        setcolor(0);
```

```
        outtextxy(xx+6,yy+6,portcom[0]);
        outtextxy(xx+6,yy+21,portcom[1]);
        setcolor(0);
        circle(xx+55,yy+8,3);
        setfillstyle(1,4);
        circle(xx+55,yy+8+(comport*16),3);
        floodfill(xx+55,yy+8+(comport*16),0);
}
void clrmenu_portsub(int xx,int yy)
{
        putimage(xx-1,yy,savepopmenu2,COPY_PUT);
        free(savepopmenu2);
}

void menu_view(int xx,int yy)
{
        menusize=imagesize(xx-1,yy,xx+72,yy+37);
        savepopmenu=malloc(menusize);
        getimage(xx-1,yy,xx+72,yy+37,savepopmenu);
        setcolor(15);setfillstyle(1,15);
        bar(xx,yy,xx+71,yy+36);
        setcolor(7);setfillstyle(1,7);
        bar(xx+1,yy+1,xx+71,yy+36);
        setcolor(0);
        outtextxy(xx+6,yy+6,view[0]);
        outtextxy(xx+6,yy+21,view[1]);
}

void clrmenu_view(int xx,int yy)
{
```

```
        putimage(xx-1,yy,savepopmenu,COPY_PUT);
        free(savepopmenu);
}

void menu_print(int xx,int yy)
{
        menusize=imagesize(xx-1,yy,xx+72,yy+37);
        savepopmenu=malloc(menusize);
        getimage(xx-1,yy,xx+72,yy+37,savepopmenu);
        setcolor(15);setfillstyle(1,15);
        bar(xx,yy,xx+71,yy+36);
        setcolor(7);setfillstyle(1,7);
        bar(xx+1,yy+1,xx+71,yy+36);
        setcolor(0);
        outtextxy(xx+6,yy+6,print[0]);
        outtextxy(xx+6,yy+21,print[1]);
}

void clrmenu_print(int xx,int yy)
{
        putimage(xx-1,yy,savepopmenu,COPY_PUT);
        free(savepopmenu);
}


void menu_help(int xx,int yy)
{
        menusize=imagesize(xx-1,yy,xx+72,yy+37);
        savepopmenu=malloc(menusize);
        getimage(xx-1,yy,xx+72,yy+37,savepopmenu);
```

```
        setcolor(15);setfillstyle(1,15);
        bar(xx,yy,xx+71,yy+36);
        setcolor(7);setfillstyle(1,7);
        bar(xx+1,yy+1,xx+71,yy+36);
        setcolor(0);
        outtextxy(xx+6,yy+6,help[0]);
        outtextxy(xx+6,yy+21,help[1]);
}
void clrmenu_help(int xx,int yy)
{
        putimage(xx-1,yy,savepopmenu,COPY_PUT);
        free(savepopmenu);
}

void bar_file(int checkx,int checky)
{
        int x=14,y=45,yp;
        yp=44;y=45+(checky*15);
        setfillstyle(1,9);
        switch(checkx)
        {       case 0:  bar(14,y,102,y+15);
                         setcolor(0);
                         outtextxy(19,y+5,file[checky]);
                         break;
                case 1:  bar(64,y,133,y+15);
                         setcolor(0);
                         outtextxy(69,y+5,mode[checky]);
                         break;
                case 2:  bar(114,y,183,y+15);
                         setcolor(0);
```

```
                outtextxy(119,y+5,port[checky]);
                break;
        case 3: bar(164,y,233,y+15);
                setcolor(0);
                outtextxy(169,y+5,view[checky]);
                break;
        case 4: bar(214,y,283,y+15);
                setcolor(0);
                outtextxy(219,y+5,print[checky]);
                break;
        case 5 : bar(264,y,333,y+15);
                setcolor(0);
                outtextxy(269,y+5,help[checky]);
    }
}

void clrbar_file(int checkx,int checky)
{
    int x=14,y=45,yp;
    yp=44;y=45+(checky*15);
    setfillstyle(1,7);
    switch(checkx)
    {   case 0: bar(14,y,102,y+15);
                setcolor(0);
                outtextxy(19,y+5,file[checky]);
                break;
        case 1: bar(64,y,133,y+15);
                setcolor(0);
                outtextxy(69,y+5,mode[checky]);
                break;
```

```
                case 2:  bar(114,y,183,y+15);
                         setcolor(0);
                         outtextxy(119,y+5,port[checky]);
                         break;
                case 3:  bar(164,y,233,y+15);
                         setcolor(0);
                         outtextxy(169,y+5,view[checky]);
                         break;
                case 4:  bar(214,y,283,y+15);
                         setcolor(0);
                         outtextxy(219,y+5,print[checky]);
                         break;
                case 5:  bar(264,y,333,y+15);
                         setcolor(0);
                         outtextxy(269,y+5,help[checky]);
        }
}

void bar_main(int mainpoint)
{
        int x=9,y=27,sizex=40;
        switch(mainpoint)
        {       case 0 : x=9;break;
                case 1 : x=59;break;
                case 2 : x=109;break;
                case 3 : x=159;sizex=45;break;
                case 4 : x=209;sizex=45;break;
                case 5 : x=259;break;
        }
        setfillstyle(1,9);
```

```
        bar(x,y,x+sizex,y+13);
        setcolor(0);
        outtextxy(x+4,y+3,menu_main[mainpoint]);
        if(mainpoint==2)        x+=8;
        line(x+4,39,x+9,39);
}

void clrbar_main(int mainpoint)
{
        int x=9,y=27,sizex=40;
        switch(mainpoint)
        {       case 0 : x=9;break;
                case 1 : x=59;break;
                case 2 : x=109;break;
                case 3 : x=159;sizex=45;break;
                case 4 : x=209;sizex=45;break;
                case 5 : x=259;break;
        }
        setfillstyle(1,7);
        bar(x,y,x+sizex,y+13);
        setcolor(0);
        outtextxy(x+4,y+3,menu_main[mainpoint]);
        if(mainpoint==2)        x+=8;
        line(x+4,39,x+9,39);
}

void clrpopmenu(int menucheck)
{
        switch(menucheck)
        {       case 0 : clrmenu_file(13,44);break;
```

```
                case 1 : clrmenu_mode(63,44);break;
                case 2 : clrmenu_port(113,44);break;
                case 3 : clrmenu_view(163,44);break;
                case 4:  clrmenu_print(213,44);break;
                case 5:   clrmenu_help(263,44);break;
        }
}

void popmenu(int menucheck)
{
        switch(menucheck)
        {       case 0 : menu_file(13,44);break;
                case 1 : menu_mode(63,44);break;
                case 2 : menu_port(113,44);break;
                case 3 : menu_view(163,44);break;
                case 4 : menu_print(213,44);break;
                case 5 : menu_help(263,44);break;
        }
}

void setport(int xx,int yy)
{
        char keyport;
        int y=45,yp=44,maincheckx=0,mainchecky=0,xmouse,ymouse;
        union REGS regs;
        menu_portsub(xx+50,44);
        checksetport=1;
        setfillstyle(1,9);
        bar(xx+51,y,xx+120,y+15);
        setcolor(0);
```

```
outtextxy(xx+55,y+5,portcom[mainchecky]);
setcolor(0);
circle(xx+105,yp+8,3);
setfillstyle(1,4);
circle(xx+105,yp+24,3);
floodfill(xx+105,yp+8+(comport*16),0);
showmouse(checkmouse);
delay(100);
do{
        keyport=getnw();
        regs.x.ax=3;
        int86(0x33,&regs,&regs);
        xmouse=regs.x.cx;
        ymouse=regs.x.dx;
        if(regs.x.bx&1)
        {
                if(xmouse>xx+51 && xmouse<xx+120 && ymouse>y && ymouse
                <y+15)
                {
                        hidemouse(checkmouse);
                        setfillstyle(1,7);
                        bar(xx+51,y+(mainchecky*15),xx+120,y+15+
                        (mainchecky*15));
                        setcolor(0);
                        outtextxy(xx+55,y+(mainchecky*15)+5,portcom
                        [mainchecky]);
                        setcolor(0);
                        circle(xx+105,yp+8+(mainchecky*15),3);
                        setfillstyle(1,7);
```

```
                floodfill(xx+105,yp+8+(comport*16),0);
                mainchecky=0;comport=0;
                if(mainchecky>1)        mainchecky=0;
                setfillstyle(1,9);
                bar(xx+51,y+(mainchecky*15),xx+120,y+15+
                (mainchecky*15));
                setcolor(0);
                outtextxy(xx+55,y+5+(mainchecky*15),portcom
                [mainchecky]);
                setcolor(0);
                circle(xx+105,yp+8+(mainchecky*15),3);
                setfillstyle(1,4);
                floodfill(xx+105,yp+8+(comport*16),0);
                showmouse(checkmouse);
        }
        else
        if(xmouse>xx+51 && xmouse<xx+120&& ymouse>y+15 && ymouse
        <y+30)
        {
                hidemouse(checkmouse);
                setfillstyle(1,7);
                bar(xx+51,y+(mainchecky*15),xx+120,y+15+
                (mainchecky*15));
                setcolor(0);
                outtextxy(xx+55,y+(mainchecky*15)+5,portcom
                [mainchecky]);
                setcolor(0);
                circle(xx+105,yp+8+(mainchecky*15),3);
                setfillstyle(1,7);
                floodfill(xx+105,yp+8+(comport*16),0);
```

```
                    mainchecky=1;comport=1;
                    if(mainchecky>1)        mainchecky=0;
                    setfillstyle(1,9);
                    bar(xx+51,y+(mainchecky*15),xx+120,y+15+
                    (mainchecky*15));
                    setcolor(0);
                    outtextxy(xx+55,y+5+(mainchecky*15),portcom
                    [mainchecky]);
                    setcolor(0);
                    circle(xx+105,yp+8+(mainchecky*15),3);
                    setfillstyle(1,4);
                    floodfill(xx+105,yp+8+(comport*16),0);
                    showmouse(checkmouse);
            }
            else
                    keyport=ESC;
}
switch(keyport)
{       case DOWN :   hidemouse(checkmouse);
                      setfillstyle(1,7);
                      bar(xx+51,y+(mainchecky*15),xx+120,y+15+
                      (mainchecky*15));
                      setcolor(0);
                      outtextxy(xx+55,y+(mainchecky*15)+5,portcom
                      [mainchecky]);
                      setcolor(0);
                      circle(xx+105,yp+8+(mainchecky*15),3);
                      setfillstyle(1,4);
                      floodfill(xx+105,yp+8+(comport*16),0);
                      mainchecky++;
```

```
                if(mainchecky>1)        mainchecky=0;
                setfillstyle(1,9);
                bar(xx+51,y+(mainchecky*15),xx+120,y+15+
                (mainchecky*15));
                setcolor(0);
                outtextxy(xx+55,y+5+(mainchecky*15),portcom
                [mainchecky]);
                setcolor(0);
                circle(xx+105,yp+8+(mainchecky*15),3);
                setfillstyle(1,4);
                floodfill(xx+105,yp+8+(comport*16),0);
                showmouse(checkmouse);
                break;
case UP :       hidemouse(checkmouse);
                setfillstyle(1,7);
                bar(xx+51,y+(mainchecky*15),xx+120,y+15+
                (mainchecky*15));
                setcolor(0);
                outtextxy(xx+55,y+(mainchecky*15)+5,portcom
                [mainchecky]);
                setcolor(0);
                circle(xx+105,yp+8+(mainchecky*15),3);
                setfillstyle(1,4);
                floodfill(xx+105,yp+8+(comport*16),0);
                mainchecky--;
                if(mainchecky<0) mainchecky=1;
                setfillstyle(1,9);
                bar(xx+51,y+(mainchecky*15),xx+120,y+15+
```

```
                                        (mainchecky*15));
                                        setcolor(0);
                                        outtextxy(xx+55,y+5+(mainchecky*15),portcom
                                        [mainchecky]);
                                        setcolor(0);
                                        circle(xx+105,yp+8+(mainchecky*15),3);
                                        setfillstyle(1,4);
                                        floodfill(xx+105,yp+8+(comport*16),0);
                                        showmouse(checkmouse);
                                        break;
                        case SPACE :    hidemouse(checkmouse);
                                        setcolor(0);setfillstyle(1,7);
                                        floodfill(xx+105,52+(comport*16),0);
                                        comport=mainchecky;
                                        setcolor(0);setfillstyle(1,4);
                                        floodfill(xx+105,52+(comport*16),0);
                                        showmouse(checkmouse);
                                        break;
                }
        }while(keyport!=ESC);
        if(comport==0)
        {       portselect=COM1;
                portselectcheck=COM1check;
        }
        if(comport==1)
        {       portselect=COM2;
                portselectcheck=COM2check;
        }
        hidemouse(checkmouse);
        clrmenu_portsub(xx+50,44);
```

```
        checksetport=0;
}

int showdrl(void)
{
        int y1=100,j;
        char temp[2][10];
        setcolor(0);
        outtextxy(50,80,"X      Y");
        for(j=pointdrl;datadrill[0][j]!=NULL && y1<400;j++)
        {
                itoa(datadrill[0][j],temp[0],10);
                itoa(datadrill[1][j],temp[1],10);
                outtextxy(40,y1,temp[0]);
                outtextxy(100,y1,temp[1]);
                y1+=15;
        }
        if(datadrill[0][j]!=NULL)
        {
                setcolor(4);
                outtextxy(420,400,"Press Page Down to down");
        }
        else
        {
                setcolor(8);
                setfillstyle(1,8);
                bar(410,400,630,420);
        }pointdrl=j;
}
```

```
int readandshowdrl(char *namefile)
{
        int i=0,j,countdrl=0;
        char far *strdrl;
        fclose(pcbfile);
        if((pcbfile=fopen(namefile,"r"))==0)
                return(0);
        checkfiledrl=1;
        while((fgets(strdrl,80,pcbfile)) != NULL)
        {       countdrl++;
        }
        fseek(pcbfile,0l,SEEK_SET);
        datadrill[0]=(int *)malloc(countdrl*2);
        datadrill[1]=(int *)malloc(countdrl*2);
        while((fgets(strdrl,80,pcbfile)) != NULL)
        {       sscanf(strdrl,"%d %d",&datadrill[0][i],&datadrill[1][i]);
                i++;
        }
        datadrill[0][i]=NULL;
        fclose(pcbfile);
        showdrl();
}

void sortfile(void)
{
        int i=0,j=0,k=0,tem[2];
        cp-=(countcp*8);
        fp-=(countfp*8);
        datadrill[0]=(int *)malloc((countcp+countfp)*2);
        datadrill[1]=(int *)malloc((countcp+countfp)*2);
```

```
for(k=0;k<countcp;k++)
{       datadrill[0][i]=((*cp++)/26)*4.7169;
        datadrill[1][i]=((*cp)/31.3)*3.125;
        cp+=7;i++;
}
for(k=0;k<countfp;k++)
{       datadrill[0][i]=((*fp++)/26)*4.7169;
        datadrill[1][i]=((*fp)/31.3)*3.125;
        fp+=7;i++;
}
datadrill[0][i]=NULL;
 for(k=0;k<i;k++)
 {      for(j=k;j<i;j++)
        {       if(datadrill[0][k]>datadrill[0][j])
                {       tem[0]=datadrill[0][k];
                        tem[1]=datadrill[1][k];
                        datadrill[0][k]=datadrill[0][j];
                        datadrill[1][k]=datadrill[1][j];
                        datadrill[0][j]=tem[0];
                        datadrill[1][j]=tem[1];
                }
        }
}
 for(k=0;k<i;k++)
{       for(j=k;j<i;j++)
        {       if ((datadrill[1][k]>datadrill[1][j])&&(datadrill[0][k]==datadrill[0][j]))
                {       tem[0]=datadrill[0][k];
                        tem[1]=datadrill[1][k];
                        datadrill[0][k]=datadrill[0][j];
                        datadrill[1][k]=datadrill[1][j];
```

```
                    datadrill[0][j]=tem[0];
                    datadrill[1][j]=tem[1];
                }
            }
        }
    }

void savefile(char *namesave)
{
    int k;
    sortfile();
    savepcb=fopen(namesave,"w");
    for(k=0;datadrill[0][k]!=NULL;k++)
    {   fprintf(savepcb,"%d %d",datadrill[0][k],datadrill[1][k]);
        fprintf(savepcb,"\n");
    }
    fclose (savepcb);
}

void printdrilltext()
{
    char temp[2][10];
    int j;
    FILE *fp;
    fp=fopen("LPT1","wb");
    fprintf(fp,"\tNo.\t x\t y");
    fprintf(fp,"\n");
    for(j=0;datadrill[0][j]!=NULL;j++)
    {   fprintf(fp,"\t%d\t%d\t",j,datadrill[0][j]);
        fprintf(fp,"\t%d",datadrill[1][j]);
```

```
                fprintf(fp,"\n");
        }
        fclose(fp);
}

void printcircuit(void)
{
        print_scr(40,50,600,460,0,0,0);
        reset();
}

void setbaud(long baudrate)
{       unsigned int divisor;
        unsigned char lsb,msb;
        divisor=115200/baudrate;
        msb=divisor>>8;
        lsb=(divisor<<8)>>8;
        if(comport==0)
        {       outport(0x3FB,128);
                outport(0x3F8,lsb);
                outport(0X3F9,msb);
        }else
        {       outport(0x2FB,128);
                outport(0x2F8,lsb);
                outport(0X2F9,msb);
        }
}

void comparm (int parity,int stop,int databits)
{
```

```
        unsigned char parmbyte;
        parmbyte=databits-5;
        if (stop==2)        parmbyte |= 4;
        if (parity != 0)    parmbyte |= 8;
        if (parity == 2)    parmbyte |= 16;
        if(comport==0)  outport(0x3FB,parmbyte);
        else                outport(0x2FB,parmbyte);
}

int initport(void)
{
        int bitset=0x00;
        int boudstop,i;
        int b;
        b=(bx*3)+by;
        boudstop=b;
        if(b==0)            b=110;
        else
        {       b=150;
                for(i=0;i<boudstop-1;i++)
                        b=b*2;
        }s++;
        setbaud(b);
        comparm (p,s,d+7);
}

void setting(int x1,int y1,int x2,int y2)
{
        struct textsettingstype textinfo;
        char color=5,keyset=0;
```

```
int i,j,pointset=0,xmouse,ymouse,init[5];
char label[5][10]={"Boud rate","Parity","data bits","stop bits"};
union REGS regs;
init[0]=bx;
init[1]=by;
init[2]=p;
init[3]=d;
init[4]=s;
gettextsettings(&textinfo);
settextstyle(2,0,6);
setfillstyle(1,color);
bar(x1,y1,x2,y2);
setcolor(15);
rectangle(x1+10,y1+10,x2-10,y2-10);
rectangle(x1+13,y1+13,x2-13,y2-13);
bar(x1-34+(x2-x1)/2,y1,x1+34+(x2-x1)/2,y1+15);
setcolor(0);
outtextxy(x1-30+(x2-x1)/2,y1,"Setting");
setcolor(11);
outtextxy(x1-29+(x2-x1)/2,y1+1,"Setting");
settextstyle(textinfo.font,textinfo.direction,textinfo.charsize);
rectangle(x1+20,y1+30,x2-20,y1+105);
setcolor(0);
rectangle(x1+21,y1+31,x2-21,y1+104);
rectangle(x1+19,y1+29,x2-19,y1+106);
setfillstyle(1,color);
bar(x1+30,y1+20,x1+112,y1+35);
setcolor(0);
setcolor(3);
outtextxy(x1+33,y1+27,label[0]);
```

```
setcolor(0);
outtextxy(x1+50,y1+45,"110");
outtextxy(x1+50,y1+65,"150");
outtextxy(x1+50,y1+85,"300");
outtextxy(x1+200,y1+45,"600");
outtextxy(x1+200,y1+65,"1200");
outtextxy(x1+200,y1+85,"2400");
outtextxy(x1+350,y1+45,"4800");
outtextxy(x1+350,y1+65,"9600");
outtextxy(x1+350,y1+85,"19200");
setcolor(11);
rectangle(x1+20,y1+120,x2-20,y1+160);
setcolor(0);
rectangle(x1+21,y1+121,x2-21,y1+159);
rectangle(x1+19,y1+119,x2-19,y1+161);
setfillstyle(1,color);
bar(x1+30,y1+115,x1+85,y1+135);
setcolor(0);
setcolor(3);
outtextxy(x1+33,y1+115,label[1]);
setcolor(0);
outtextxy(x1+50,y1+135,"None");
outtextxy(x1+200,y1+135,"odd");
outtextxy(x1+350,y1+135,"even");
setcolor(11);
rectangle(x1+20,y1+175,x2-120,y1+215);
setcolor(0);
rectangle(x1+21,y1+176,x2-121,y1+214);
rectangle(x1+19,y1+174,x2-119,y1+216);
setfillstyle(1,color);
```

```
bar(x1+30,y1+170,x1+108,y1+190);
setcolor(0);
setcolor(3);
outtextxy(x1+33,y1+170,label[2]);
setcolor(0);
outtextxy(x1+50,y1+190,"7");
outtextxy(x1+200,y1+190,"8");
setcolor(11);
rectangle(x1+20,y1+230,x2-120,y1+270);
setcolor(0);
rectangle(x1+21,y1+231,x2-121,y1+269);
rectangle(x1+19,y1+229,x2-119,y1+271);
setfillstyle(1,color);
bar(x1+30,y1+225,x1+108,y1+245);
setcolor(0);
setcolor(3);
outtextxy(x1+33,y1+225,label[3]);
setcolor(0);
outtextxy(x1+50,y1+245,"1");
outtextxy(x1+200,y1+245,"2");
setcolor(0);
for(i=0;i<3;i++)
{       circle(x1+40,y1+(i*20)+48,3);
        circle(x1+40,y1+(i*20)+48,2);
        circle(x1+190,y1+(i*20)+48,3);
        circle(x1+190,y1+(i*20)+48,2);
        circle(x1+340,y1+(i*20)+48,3);
        circle(x1+340,y1+(i*20)+48,2);
}
for(i=0;i<3;i++)
```

```
{       circle(x1+40,y1+138+(i*55),3);
        circle(x1+40,y1+138+(i*55),2);
        circle(x1+190,y1+138+(i*55),3);
        circle(x1+190,y1+138+(i*55),2);
}
for(i=0;i<1;i++)
{       circle(x1+340,y1+138+(i*55),3);
        circle(x1+340,y1+138+(i*55),2);
}
bottomup(x1+335,y1+185,x1+390,y1+205,1,7);
bottomup(x1+335,y1+240,x1+390,y1+260,1,7);
outtextxy(x1+354,y1+191,"OK");
outtextxy(x1+340,y1+246,"Cancel");
setcolor(15);setfillstyle(1,color);
rectangle(x1+20,y1+30,x2-20,y1+105);
bar(x1+30,y1+20,x1+112,y1+35);
outtextxy(x1+33,y1+27,label[0]);
setcolor(0);setfillstyle(1,0);
pieslice(x1+40+(bx*150),y1+48+(by*20),0,360,2);
setcolor(0);setfillstyle(1,0);
pieslice(x1+40+(bx*150),y1+48+(by*20),0,360,2);
pieslice(x1+40+(p*150),y1+138,0,360,2);
pieslice(x1+40+(d*150),y1+193,0,360,2);
pieslice(x1+40+(s*150),y1+248,0,360,2);
showmouse(checkmouse);
do
{       keyset=getnw();
        regs.x.ax=3;
        int86(0x33,&regs,&regs);
        xmouse=regs.x.cx;
```

```
ymouse=regs.x.dx;
for(i=0;i<3;i++)
{       for(j=0;j<3;j++)
        {       if(regs.x.bx&1 && xmouse>x1+37+(j*150) && xmouse
                <x1+82+(j*150) && ymouse>y1+(i*20)+42 && ymouse<y1+
                (i*20)+53)
                {       hidemouse(checkmouse);
                        setcolor(color);
                        setfillstyle(1,color);
                        pieslice(x1+40+(bx*150),y1+48+(by*20),0,360,2);
                        by=i;bx=j;
                        setcolor(0);
                        setfillstyle(1,0);
                        pieslice(x1+40+(bx*150),y1+48+(by*20),0,360,2);
                        showmouse(checkmouse);
                        delay(100);
                }
        }
}
for(i=0;i<3;i++)
{       for(j=0;j<3;j++)
        {       if(regs.x.bx&1 && xmouse>x1+37+(i*150) && xmouse
                <x1+82+(i*150) && ymouse>y1+(j*55)+133 && ymouse<y1+
                (j*55)+141)
                {
                        if(xmouse>x1+37+(300)&&xmouse<x1+82+(300)
                        && ymouse>y1+(55)+133 && ymouse<y1+
                        (110)+141)
                        {
```

```
                    }else
                    {       hidemouse(checkmouse);
                            setcolor(color);
                            setfillstyle(1,color);
                            if(j==0)
                            {       pieslice(x1+40+(p*150),y1+138+
                                    (j*55),0,360,2);
                                    p=i;
                            }
                            if(j==1)
                            {       pieslice(x1+40+(d*150),y1+138+
                                    (j*55),0,360,2);
                                    d=i;
                            }
                            if(j==2)
                            {       pieslice(x1+40+(s*150),y1+138+
                                    (j*55),0,360,2);
                                    s=i;
                            }
                            setcolor(0);setfillstyle(1,0);
                            pieslice(x1+40+(i*150),y1+138+
                            (j*55),0,360,2);
                            showmouse(checkmouse);
                            delay(100);
                    }
                }
            }
        }
        if(regs.x.bx&1&&xmouse>x1+335 && xmouse<x1+390 && ymouse>y1+185
        && ymouse<y1+205)
```

```
        keyset=RETURN;
if(regs.x.bx&1&&xmouse>x1+335 && xmouse<x1+390 && ymouse>y1+240
&& ymouse<y1+260)
        keyset=ESC;
switch(keyset)
{       case TAB :      setcolor(11);
                        setfillstyle(1,color);
                        if(pointset==0)
                        {       rectangle(x1+20,y1+30,x2-20,y1+105);
                                bar(x1+30,y1+20,x1+112,y1+35);
                                outtextxy(x1+33,y1+27,label[0]);
                        }
                        if(pointset==1)
                        {       rectangle(x1+20,y1+120,x2-20,y1+160);
                                bar(x1+30,y1+115,x1+85,y1+130);
                                outtextxy(x1+33,y1+115,label[1]);
                        }
                        if(pointset==2)
                        {       rectangle(x1+20,y1+175,x2-120,y1+215);
                                bar(x1+30,y1+170,x1+108,y1+185);
                                outtextxy(x1+33,y1+170,label[2]);
                        }
                        if(pointset==3)
                        {       rectangle(x1+20,y1+230,x2-120,y1+270);
                                bar(x1+30,y1+225,x1+108,y1+240);
                                outtextxy(x1+33,y1+225,label[3]);
                        }
                        if(pointset==4)
                        {       setcolor(0);
                                outtextxy(x1+354,y1+191,"OK");
```

```
}
if(pointset==5)
{
        setcolor(0);
        outtextxy(x1+340,y1+246,"Cancel");
}
pointset++;
if(pointset>=6)  pointset=0;
setcolor(15);setfillstyle(1,color);
if(pointset==0)
{       rectangle(x1+20,y1+30,x2-20,y1+105);
        bar(x1+30,y1+20,x1+112,y1+35);
        outtextxy(x1+33,y1+27,label[0]);
}
if(pointset==1)
{       rectangle(x1+20,y1+120,x2-20,y1+160);
        bar(x1+30,y1+115,x1+85,y1+130);
        outtextxy(x1+33,y1+115,label[1]);
}
if(pointset==2)
{       rectangle(x1+20,y1+175,x2-120,y1+215);
        bar(x1+30,y1+170,x1+108,y1+185);
        outtextxy(x1+33,y1+170,label[2]);
}
if(pointset==3)
{       rectangle(x1+20,y1+230,x2-120,y1+270);
        bar(x1+30,y1+225,x1+108,y1+240);
        outtextxy(x1+33,y1+225,label[3]);
}
if(pointset==4)
```

```
                {       setcolor(15);
                        outtextxy(x1+354,y1+191,"OK");
                }
                if(pointset==5)
                {       setcolor(15);
                        outtextxy(x1+340,y1+246,"Cancel");
                }
                break;
case DOWN :     if(pointset==0)
                {       if(by<2)
                        {       setfillstyle(1,color);
                                setcolor(color);
                                pieslice(x1+40+(bx*150),y1+48+
                                (by*20),0,360,2);
                                by++;
                                setcolor(0);setfillstyle(1,0);
                                pieslice(x1+40+(bx*150),y1+48+
                                (by*20),0,360,2);
                        }
                }
                break;
case UP   :     if(pointset==0)
                {       if(by>0)
                        {
                                setfillstyle(1,color);
                                setcolor(color);
                                pieslice(x1+40+(bx*150),y1+48+
                                (by*20),0,360,2);
                                by--;
                                setcolor(0);setfillstyle(1,0);
```

```
                                pieslice(x1+40+(bx*150),y1+48+
                                (by*20),0,360,2);
                        }
                }
                break;
case RIGHT :    if(pointset==0)
                {       if(bx<2)
                        {       setfillstyle(1,color);setcolor(color);
                                pieslice(x1+40+(bx*150),y1+48+
                                (by*20),0,360,2);
                                bx++;
                                setcolor(0);setfillstyle(1,0);
                                pieslice(x1+40+(bx*150),y1+48+
                                (by*20),0,360,2);
                        }
                }
        if(pointset==1)
        {       if(p<2)
                {       setfillstyle(1,color);
                        setcolor(color);
                        pieslice(x1+40+(p*150),y1+138,0,
                        360,2);
                        p++;
                        setcolor(0);setfillstyle(1,0);
                        pieslice(x1+40+(p*150),y1+138,0,
                        360,2);
                }
        }if(pointset==2)
        {       if(d<1)
```

```
                {       setfillstyle(1,color);setcolor(color);
                        pieslice(x1+40+(d*150),y1+193,0,
                        360,2);
                        d++;
                        setcolor(0);
                        setfillstyle(1,0);
                        pieslice(x1+40+(d*150),y1+193,0,
                        360,2);
                }
        }
        if(pointset==3)
        {       if(s<1)
                {       setfillstyle(1,color);setcolor(color);
                        pieslice(x1+40+(s*150),y1+248,0,
                                360,2);
                                s++;
                                setcolor(0);setfillstyle(1,0);
                                pieslice(x1+40+(s*150),y1+248,0,
                                360,2);
                        }
                }break;
case LEFT :     if(pointset==0)
                {       if(bx>0)
                        {       setfillstyle(1,color);
                                setcolor(color);
                                pieslice(x1+40+(bx*150),y1+48+
                                (by*20),0,360,2);
                                bx--;
                                setcolor(0);setfillstyle(1,0);
                                pieslice(x1+40+(bx*150),y1+48+
```

```
                    (by*20),0,360,2);
            }
    }
    if(pointset==1)
    {       if(p>0)
            {       setfillstyle(1,color);setcolor(color);
                    pieslice(x1+40+(p*150),y1+138,0,
                    360,2);
                    p--;
                    setcolor(0);setfillstyle(1,0);
                    pieslice(x1+40+(p*150),y1+138,0,
                    360,2);
            }
    }
    if(pointset==2)
    {       if(d>0)
            {       setfillstyle(1,color);setcolor(color);
                    pieslice(x1+40+(d*150),y1+193,0,
                    360,2);
                    d--;
                    setcolor(0);setfillstyle(1,0);
                    pieslice(x1+40+(d*150),y1+193,0,
                    360,2);
            }
    }
    if(pointset==3)
    {       if(s>0)
            {       setfillstyle(1,color);setcolor(color);
                    pieslice(x1+40+(s*150),y1+248,0,
```

```
                                                    360,2);
                                                    s--;
                                                    setcolor(0);setfillstyle(1,0);
                                                    pieslice(x1+40+(s*150),y1+248,0,
                                                    360,2);
                                            }
                                    }break;
                            }
        }while(keyset!=ESC && keyset != RETURN);
        if(keyset==RETURN)
        initport();
        if(keyset==ESC)
        {       bx=init[0];
                by=init[1];
                p=init[2];
                d=init[3];
                s=init[4];
        }
        settextstyle(textinfo.font,textinfo.direction,textinfo.charsize);
        hidemouse(checkmouse);
}

autodrill()
{
        int j,statein=0,in=0;
        statein=inport(portselectcheck);
        outport(portselectcheck,0);
        bottomdown(176,455,276,472,2,7);
        outtextxy(176+50-32,460,"TRANSFER");
        for(j=0;datadrill[0][j]!=NULL;j++)
```

```
        {       outport(portselect,datadrill[0][j]/255);
                do
                {       statein=inport(portselectcheck);
                        if(statein&1)   in=inport(portselect);
                }while(!(statein&1) || in !=1);
                outport(portselect,datadrill[0][j]%255);
                do
                {       statein=inport(portselectcheck);
                                if(statein&1) in=inport(portselect);
                }while(!(statein&1)|| in !=1);
                outport(portselect,datadrill[1][j]/255);
                do
                {       statein=inport(portselectcheck);
                        if(statein&1) in=inport(portselect);
                }while(!(statein&1)|| in !=1);
                outport(portselect,datadrill[1][j]%255);
                do
                {       statein=inport(portselectcheck);
                        if(statein&1) in=inport(portselect);
                }while(!(statein&1)|| in !=1);
        }
        bottomdown(176,455,276,472,2,7);
        outtextxy(176+50-12,460,"APD");
}

void manual(void)
{
        int j,statein=0,in=0;
        statein=inport(portselectcheck);
        outport(portselectcheck,0);
```

```
        bottomdown(176,455,276,472,2,7);
        outtextxy(176+50-32,460,"TRANSFER");
        for(j=0;datadrill[0][j]!=NULL;j++)
        {       outport(portselect,datadrill[0][j]/255);
                do
                {       statein=inport(portselectcheck);
                        if(statein&1)     in=inport(portselect);
                }while(!(statein&1) || in !=1);
                outport(portselect,datadrill[0][j]%255);
                do
                {       statein=inport(portselectcheck);
                                if(statein&1) in=inport(portselect);
                }while(!(statein&1)|| in !=1);
                outport(portselect,datadrill[1][j]/255);
                do
                {       statein=inport(portselectcheck);
                        if(statein&1) in=inport(portselect);
                }while(!(statein&1)|| in !=1);
                outport(portselect,datadrill[1][j]%255);
                do
                {       statein=inport(portselectcheck);
                        if(statein&1) in=inport(portselect);
                }while(!(statein&1)|| in !=1);
        }
        bottomdown(176,455,276,472,2,7);
        outtextxy(176+50-12,460,"APD");
}

int gotosub(int casex,int casey,int ratio)
{
```

```
int key,length;
switch(casex)
{       case 0 : casex=0;break;
        case 1 : casex=6;break;
        case 2 : casex=9;break;
        case 3 : casex=11;break;
        case 4 : casex=13;break;
        case 5 : casex=15;break;
}
key=casex+casey;
switch(key)
{       case 0 : setcolor(8);
                 setfillstyle(1,8);
                 bar(10,50,630,440);
                 if(checkfilepcb==1 || checkfiledrl==1)
                 {
                        free(ct); free(cp);
                        free(ca); free(ft);
                        free(fp); free(datadrill[0]);
                        free(datadrill[1]);
                        checkfilepcb=0;checkfiledrl=0;pointdrl=0;
                 }
                 break;
        case 1 : strcpy(filename,Ofile(0,checkmouse));
                 if(strcmp(filename,"")!=0)
                 {       length=strlen(filename);
                        filename+=(length-4);
                        if(strcmp(filename,".PCB")==0 || strcmp(filename,".pcb")==0)
                        {       if(checkfilepcb==1)
                                {
```

```
                free(ct); free(cp);
                free(ca);free(ft);
                free(fp);free(datadrill[0]);
                free(datadrill[1]);checkfilepcb=0;
                pointdrl=0;
        }
        if(checkfiledrl==1)
        {       free(datadrill[0]);
                free(datadrill[1]);
                checkfiledrl=0;pointdrl=0;
        }
        setcolor(8);setfillstyle(1,8);
        bar(10,50,630,440);
        filename-=(length-4);
        if(readpcbfile(filename)!=0)
        {       sortfile();showpcb(ratio);
        }
} else if(strcmp(filename,".DRL")==0 ||
        strcmp(filename,".drl")==0)
        {       if(checkfilepcb==1)
                {       free(ct); free(cp);
                        free(ca);free(ft);
                        free(fp);free(datadrill[0]);
                        free(datadrill[1]);
                        checkfilepcb=0;
                        pointdrl=0;
                }
                if(checkfiledrl==1)
                {       free(datadrill[0]);
                        free(datadrill[1]);
```

```
                                checkfiledrl=0;pointdrl=0;
                            }
                            setcolor(8);setfillstyle(1,8);
                            bar(10,50,630,440);
                            filename-=(length-4);
                            readandshowdrl(filename);
                        }
                }
                free(filename);
                break;
case 2 : strcpy(filename,Ofile(1,checkmouse));
        if(strcmp(filename,"")!=0)
                savefile(filename);
        break;
case 3 : changedrv(checkmouse);break;
case 4: closegraph();
        system("command.com");
        opengraph();
        background();
        if(checkfilepcb==1)
                showpcb(ratio);
        else if(checkfiledrl==1)
                readandshowdrl(filename);
        break;
case 5 : cleardevice();closegraph();exit(0);hidemouse(checkmouse);break;
case 6 : outtextxy(10,10,"auto");break;
case 7 : manual();break;
case 8 : setcolor(8); setfillstyle(1,8);
        bar(10,50,630,440);
        remote();break;
```

```
case 9 : setport(138,44);break;
case 10 :menusize=imagesize(105,100,535,400);
        savepopmenu=malloc(menusize);
        getimage(105,100,535,400,savepopmenu);
        setting(105,100,535,400);
        putimage(105,100,savepopmenu,COPY_PUT);
        free(savepopmenu);
        break;
case 11 :hidemouse(checkmouse);
        if(checkfilepcb==1)
        {       if(ratio>1)
                {       ratio--;
                        setcolor(8);setfillstyle(1,8);
                        bar(10,50,630,440);
                        showpcb(ratio);
                }
        }
        break;
case 12 :hidemouse(checkmouse);
        if(checkfilepcb==1)
        {       if(20+(maxx-minx)/(50/ratio)<530 && (maxy-miny)/(66/ratio)
                <350)
                        ratio++;
                setcolor(8);setfillstyle(1,8);
                bar(10,50,630,440);
                showpcb(ratio);
                showmouse(checkmouse);
        } break;
case 13 :if(checkfiledrl==1)        printdrilltext();break;
case 14 :if(checkfilepcb==1)        printcircuit();break;
```

```
                case 15 :menusize=imagesize(105,100,535,400);
                        savepopmenu=malloc(menusize);
                        getimage(105,100,535,400,savepopmenu);
                        pro_help(105,100,535,400,checkmouse);
                        putimage(105,100,savepopmenu,COPY_PUT);
                        free(savepopmenu);
                        break;
                case 16 :menusize=imagesize(105,100,535,400);
                        savepopmenu=malloc(menusize);
                        getimage(105,100,535,400,savepopmenu);
                        about(105,100,535,400,checkmouse);
                        putimage(105,100,savepopmenu,COPY_PUT);
                        free(savepopmenu);
                        break;
        }return(ratio);
}

void clrkey()
{
        _AH=0x0c;
        geninterrupt(0x21);
}

int menu(void)
{
        char key=0,checkkey;
        int maincheckx=0,checkmenu=0,mainchecky=0,ratiopcb=1,xmouse,ymouse;
        union REGS regs;
        checkmouse=mousedrv();
        background();
```

```
showmouse(checkmouse);
key=0;checkmenu=0;
while(key!=RETURN)
{       while(key!=ESC)
        {       key=getnw();
                regs.x.ax=3;
                int86(0x33,&regs,&regs);
                xmouse=regs.x.cx;
                ymouse=regs.x.dx;
                showmouse(checkmouse);
                if(regs.x.bx&1)
                {       if(xmouse>11 && xmouse<51 && ymouse >20 && ymouse
                        <40)
                        {       key=ALT_F;
                                delay(100);
                        }else if(xmouse>61 && xmouse<101 && ymouse >20 &&
                        ymouse<40)
                        {       key=ALT_M;
                                delay(100);
                        }
                        else if(xmouse>111 && xmouse<151 && ymouse >20
                        && ymouse<40)
                        {       key=ALT_O;
                                delay(100);
                        }
                        else if(xmouse>151 && xmouse<201 && ymouse >20 &&
                        ymouse<40)
                        {       key=ALT_V;
                                delay(100);
                        }
```

```
        else if(xmouse>211 && xmouse<251 && ymouse >20 &&
        ymouse<40)
        {       key=ALT_P;
                delay(100);
        }
        else if(xmouse>261 && xmouse<301 && ymouse >20 &&
        ymouse<40)
        {       key=ALT_H;
                delay(100);
        }
        else
                key=ESC;
}
if(regs.x.bx & 1 && xmouse>14 && xmouse<102&& ymouse>45 &&
ymouse<60 && checkmenu==1 && mainchèckx==0)
{       maincheckx=0;mainchecky=0;
        key=RETURN;
}
if(regs.x.bx&1 && xmouse>14 && xmouse<102  && ymouse>60 &&
ymouse<75 && checkmenu==1&& maincheckx==0)
{       maincheckx=0;mainchecky=1;
        key=RETURN;
}
if(regs.x.bx&1 && xmouse>14 && xmouse<102 && ymouse>75  &&
ymouse<90 && checkmenu==1&& maincheckx==0)
{       maincheckx=0;mainchecky=2;
        key=RETURN;
}
if(regs.x.bx&1 && xmouse>14 && xmouse<102 && ymouse>90
&& ymouse<105 && checkmenu==1&& maincheckx==0)
```

```
{       maincheckx=0;mainchecky=3;
        key=RETURN;
}
if(regs.x.bx&1 && xmouse>14 && xmouse<102 && ymouse>105
&& ymouse<120 && checkmenu==1&& maincheckx==0)
{       maincheckx=0;mainchecky=4;
        key=RETURN;
}
if(regs.x.bx&1 && xmouse>14 && xmouse<102 && ymouse>120
&& ymouse<135 && checkmenu==1&& maincheckx==0)
{       maincheckx=0;mainchecky=5;
        key=RETURN;
}
if(regs.x.bx&1 && xmouse>64 && xmouse<133 && ymouse>45
&& ymouse<60 && checkmenu==1&& maincheckx==1)
{       maincheckx=1;mainchecky=0;
        key=RETURN;
}
if(regs.x.bx&1 && xmouse>64 && xmouse<133 && ymouse>60
&& ymouse<75 && checkmenu==1&& maincheckx==1)
{       maincheckx=1;mainchecky=1;
        key=RETURN;
}
if(regs.x.bx&1 && xmouse>64 && xmouse<133 && ymouse>75
&& ymouse<90 && checkmenu==1&& maincheckx==1)
{       maincheckx=1;mainchecky=2;
        key=RETURN;
}
if(regs.x.bx&1 && xmouse>114 && xmouse<163 && ymouse>45
&& ymouse<60 && checkmenu==1&& maincheckx==2)
```

```
{       maincheckx=2;mainchecky=0;
        key=RETURN;
}
if(regs.x.bx&1 && xmouse>114 && xmouse<163 && ymouse>60
&& ymouse<75 && checkmenu==1&& maincheckx==2)
{       maincheckx=2;mainchecky=1;
        key=RETURN;
}
if(regs.x.bx&1 && xmouse>164 && xmouse<213 && ymouse>45
&& ymouse<60 && checkmenu==1&& maincheckx==3 &&
checksetport!=1)
{       maincheckx=3;mainchecky=0;
        key=RETURN;
}
if(regs.x.bx&1 && xmouse>164 && xmouse<213 && ymouse>60
&& ymouse<75 && checkmenu==1 &&maincheckx==3 &&
checksetport!=1)
{       maincheckx=3;mainchecky=1;
        key=RETURN;
}
if(regs.x.bx&1 && xmouse>214 && xmouse<283 && ymouse>45
&& ymouse<60 && checkmenu==1&& maincheckx==4)
{       maincheckx=4;mainchecky=0;
        key=RETURN;
}
if(regs.x.bx&1 && xmouse>214 && xmouse<283 && ymouse>60
&& ymouse<75 && checkmenu==1 &&maincheckx==4)
{       maincheckx=4;mainchecky=1;
        key=RETURN;
}
```

```
if(regs.x.bx&1 && xmouse>264 && xmouse<333 && ymouse>45
&& ymouse<60 && checkmenu==1&& maincheckx==5)
{       maincheckx=5;mainchecky=0;
        key=RETURN;
}
if(regs.x.bx&1 && xmouse>264 && xmouse<333 && ymouse>60
&& ymouse<75 && checkmenu==1 &&maincheckx==5)
{       maincheckx=5;mainchecky=1;
        key=RETURN;
}
switch(key)
{       case ALT_F :    hidemouse(checkmouse);
                        if(checkmenu==1)
                        {       clrbar_main(maincheckx);
                                clrpopmenu(maincheckx);
                        }
                        checkmenu=1;
                        maincheckx=0;mainchecky=0;
                        popmenu(maincheckx);
                        bar_file(maincheckx,mainchecky);
                        showmouse(checkmouse);
                        break;
        case ALT_M :    hidemouse(checkmouse);
                        if(checkmenu==1)
                        {       clrbar_main(maincheckx);
                                clrpopmenu(maincheckx);
                        }
                        checkmenu=1;
                        maincheckx=1;mainchecky=0;
                        popmenu(maincheckx);
```

```
                bar_file(maincheckx,mainchecky);
                showmouse(checkmouse);
                break;
case ALT_O :    hidemouse(checkmouse);
                if(checkmenu==1)
                {       clrbar_main(maincheckx);
                        clrpopmenu(maincheckx);
                }
                checkmenu=1;
                maincheckx=2;mainchecky=0;
                popmenu(maincheckx);
                bar_file(maincheckx,mainchecky);
                showmouse(checkmouse);
                break;
case ALT_V :    hidemouse(checkmouse);
                if(checkmenu==1)
                {       clrbar_main(maincheckx);
                        clrpopmenu(maincheckx);
                }
                checkmenu=1;
                maincheckx=3;mainchecky=0;
                popmenu(maincheckx);
                bar_file(maincheckx,mainchecky);
                showmouse(checkmouse);
                break;
case ALT_P :    hidemouse(checkmouse);
                if(checkmenu==1)
                {       clrbar_main(maincheckx);
                        clrpopmenu(maincheckx);
                }
```

```
                checkmenu=1;
                maincheckx=4;mainchecky=0;
                bar_main(maincheckx);
                popmenu(maincheckx);
                bar_file(maincheckx,mainchecky);
                showmouse(checkmouse);
                break;
case ALT_H :    hidemouse(checkmouse);
                if(checkmenu==1)
                {       clrbar_main(maincheckx);
                        clrpopmenu(maincheckx);
                }
                checkmenu=1;
                maincheckx=5;mainchecky=0;
                bar_main(maincheckx);
                popmenu(maincheckx);
                bar_file(maincheckx,mainchecky);
                showmouse(checkmouse);
                break;
case F3 :       hidemouse(checkmouse);
                clrbar_file(maincheckx,mainchecky);
                clrbar_main(maincheckx);
                clrpopmenu(maincheckx);
                filename=Ofile(0,checkmouse);
                checkmenu=0;
                showmouse(checkmouse);
                break;

case LEFT :     hidemouse(checkmouse);
                if(checkmenu==1)
```

```
{       if(maincheckx==3)
                mainchecky=comport;
        else mainchecky=0;
                clrbar_main(maincheckx);
        clrpopmenu(maincheckx);
        maincheckx--;
        if(maincheckx<0)
                maincheckx=5;
        bar_main(maincheckx);
        popmenu(maincheckx);
        bar_file(maincheckx,mainchecky);
}
showmouse(checkmouse);
break;
case RIGHT :    hidemouse(checkmouse);
if(checkmenu==1)
{       if(maincheckx==1)
                mainchecky=comport;
        else mainchecky=0;
                clrbar_main(maincheckx);
        clrpopmenu(maincheckx);
        maincheckx++;
        if(maincheckx>5)
                maincheckx=0;
        bar_main(maincheckx);
        popmenu(maincheckx);
        bar_file(maincheckx,mainchecky);
}
showmouse(checkmouse);
break;
```

```
case DOWN :    hidemouse(checkmouse);
               if(checkmenu==1)
               {       clrbar_file(maincheckx,mainchecky);
                       mainchecky++;
                       if(maincheckx==0)
                               if(mainchecky>5)
                                       mainchecky=0;
                       if(maincheckx==1)
                               if(mainchecky>2)
                                       mainchecky=0;
                       if(maincheckx==2)
                               if(mainchecky>1)
                                       mainchecky=0;
                       if(maincheckx==3)
                               if(mainchecky>1)
                                       mainchecky=0;
                       if(maincheckx==4)
                               if(mainchecky>1)
                                       mainchecky=0;
                       bar_file(maincheckx,mainchecky);
               }
               showmouse(checkmouse);
               break;
case UP  :     hidemouse(checkmouse);
               if(checkmenu==1)
               {       clrbar_file(maincheckx,mainchecky);
                       mainchecky--;
                       if(maincheckx==0)
                               if(mainchecky<0)
                                       mainchecky=5;
```

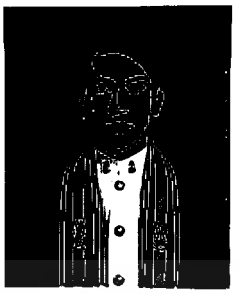
```
                    if(maincheckx==1)
                            if(mainchecky<0)
                                    mainchecky=2;
                    if(maincheckx==2)
                            if(mainchecky<0)
                                    mainchecky=1;
                    if(maincheckx==3)
                            if(mainchecky<0)
                                    mainchecky=1;
                    if(maincheckx==4)
                            if(mainchecky<0)
                                    mainchecky=1;
                    bar_file(maincheckx,mainchecky);
            }
            showmouse(checkmouse);
            break;
case PGUP :  hidemouse(checkmouse);
            if(checkfilepcb==1)
            {       if(20+(maxx-minx)/(50/ratiopcb)<530
                    && (maxy-miny)/(66/ratiopcb)<350)
                            ratiopcb++;
                    setcolor(8);setfillstyle(1,8);
                    bar(10,50,630,440);
                    showpcb(ratiopcb);
                    showmouse(checkmouse);
            }else if(checkfiledrl==1 && pointdrl>20)
            {       if(pointdrl%20!=0)
                    {       pointdrl=pointdrl-
                            (pointdrl%20);
                            pointdrl-=20;
```

```
                    }else
                    {
                        if(pointdrl<=20)  pointdrl=0;
                        else     pointdrl-=40;
                    }
                    setcolor(8);setfillstyle(1,8);
                    bar(10,50,630,440);
                    showdrl();
                }break;
case PGDN :     hidemouse(checkmouse);
                if(checkfilepcb==1)
                {       if(ratiopcb>1)
                        {       ratiopcb--;
                                setcolor(8);
                                setfillstyle(1,8);
                                bar(10,50,630,440);
                                showpcb(ratiopcb);
                        }
                }
                else if(checkfiledrl==1 && datadrill[0]
                [pointdrl]!=NULL)
                {       setcolor(8);
                        setfillstyle(1,8);
                        bar(10,50,630,440);
                }
                showmouse(checkmouse);
                break;
case RETURN :   if(checkmenu==1)
                {       hidemouse(checkmouse);
                        If(maincheckx!=2 || mainchecky!=0)
```

```
                                        {       clrbar_file(maincheckx,mainchecky);
                                                clrbar_main(maincheckx);
                                                clrpopmenu(maincheckx);
                                        }
                                        if(maincheckx==0 && mainchecky==1)
                                                ratiopcb=1;
                                        if(checkmenu==1)
                                        ratiopcb=gotosub(maincheckx,mainchecky,rati
                                        opcb);
                                        if(maincheckx!=2 || mainchecky!=0)
                                                checkmenu=0;
                                        showmouse(checkmouse);
                                }
                                break;
                        }
                }
        if(key==ESC && checkmenu==1)
        {       hidemouse(checkmouse);
                checkmenu=0;
                clrbar_main(maincheckx);
                clrpopmenu(maincheckx);
                key=0;showmouse(checkmouse);
        }
        else
                key=0;
        }
}
void main()
{       int mousecheck=1;
        filename=(char *)malloc(13);
```

```
        filename="";
        opengraph();
        showtitle();
        mousecheck=menu();
        cleardevice();
        closegraph();
        clrscr();
}
```

ประวัติผู้แต่ง



ชื่อผู้ทำปริญญานิพนธ์	นายชาญณรงค์ ปิ่นทอง
วันเดือนปีเกิด	17 กันยายน พ.ศ. 2521
สถานที่เกิด	จังหวัด อ่างทอง
ภูมิลำเนาเดิม	91/3 หมู่ 4 ตำบลหลักแก้ว อำเภอวิเศษชัยชาญ จังหวัดอ่างทอง
ที่อยู่ปัจจุบัน	91/3 หมู่ 4 ตำบลหลักแก้ว อำเภอวิเศษชัยชาญ จังหวัดอ่างทอง
โทรศัพท์	-

ประวัติการศึกษา

ประถมศึกษา	โรงเรียนวัดคลองสำโรง
มัธยมศึกษาตอนต้นโรงเรียน	โรงเรียนวิเศษไชยชาญ"ตันติวิทยาภูมิ"
ประกาศนียบัตรวิชาชีพ (ปวช.)	วิทยาลัยเทคนิคอ่างทอง
ประกาศนียบัตรวิชาชีพชั้นสูง (ปวส.)	สถาบันเทคโนโลยีราชมงคลวิทยาเขต-เทคนิคนนทบุรี
ปริญญาตรี	สาขาวิชา อิเล็กทรอนิกส์และคอมพิวเตอร์ คณะครุศาสตร์อุตสาหกรรม
ผลงานที่ได้รับรางวัล	-
ทุนการศึกษา	-
คติพจน์	กระทำความดีให้ถึงที่สุด

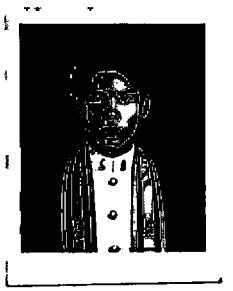
ประวัติผู้แต่ง

ชื่อผู้ทำปริญญานิพนธ์	นายธีรพล หว่านณรงค์
วันเดือนปีเกิด	25 กรกฎาคม พ.ศ. 2520
สถานที่เกิด	จังหวัด สมุทรปราการ
ภูมิลำเนาเดิม	43/6 หมู่ 11 ตำบลคลองด่าน อำเภอบางบ่อ จังหวัดสมุทรปราการ
ที่อยู่ปัจจุบัน	43/6 หมู่ 11 ตำบลคลองด่าน อำเภอบางบ่อ จังหวัดสมุทรปราการ
โทรศัพท์	3301108,3301419

ประวัติการศึกษา

ประถมศึกษา	โรงเรียนวัดมงคลโคธาวาส
มัธยมศึกษาตอนต้นโรงเรียน	โรงเรียนสมุทรปราการ
ประกาศนียบัตรวิชาชีพ (ปวช.)	วิทยาลัยเทคนิคสมุทรปราการ
ประกาศนียบัตรวิชาชีพชั้นสูง (ปวส.)	สถาบันเทคโนโลยีราชมงคลวิทยาเขต- เทคนิคกรุงเทพฯ
ปริญญาตรี	สาขาวิชา อิเล็กทรอนิกส์และ คอมพิวเตอร์ คณะครุศาสตร์อุตสาหกรรม
ผลงานที่ได้รับรางวัล	-
ทุนการศึกษา	-
คติพจน์	หวังให้ไกลแล้วไปให้เลย

ประวัติผู้แต่ง



ชื่อผู้ทำปริญญานิพนธ์	นายสมบูรณ์ กิจวัฒนาบูลย์
วันเดือนปีเกิด	21 กุมภาพันธ์ พ.ศ. 2521
สถานที่เกิด	กรุงเทพมหานคร
ภูมิลำเนาเดิม	285/7 หมู่ 2 แขวงคลองถนน เขตบางเขน กรุงเทพมหานคร
ที่อยู่ปัจจุบัน	285/7 หมู่ 2 แขวงคลองถนน เขตบางเขน กรุงเทพมหานคร
โทรศัพท์	5238178,5322062

ประวัติการศึกษา

ประถมศึกษา	โรงเรียนประเทืองทิพย์วิทยา
มัธยมศึกษาตอนต้นโรงเรียน	โรงเรียนฤทธิยะวรรณาลัย
ประกาศนียบัตรวิชาชีพ (ปวช.)	วิทยาลัยเทคนิคปทุมธานี
ประกาศนียบัตรวิชาชีพชั้นสูง (ปวส.) เทคนิคกรุงเทพฯ	สถาบันเทคโนโลยีราชมงคลวิทยาเขต
ปริญญาตรี	สาขาวิชา อิเล็กทรอนิกส์และคอมพิวเตอร์ คณะครุศาสตร์อุตสาหกรรม
ผลงานที่ได้รับรางวัล	-
ทุนการศึกษา	-
คติพจน์	ทำวันนี้ให้ดีที่สุดแล้วพรุ่งนี้จะดีเอง