บทที่ 3

ส่วนประกอบการทำงานและการพัฒนาโปรแกรม

ในบทที่แล้วได้กล่าวถึงทฤษฎีและความรู้พื้นฐานในการศึกษาและทำวิจัย สำหรับบทนี้จะกล่าวถึงส่วนประกอบที่สำคัญของโปรแกรมเลียนแบบเทอร์มินอล ส่วนประกอบการทำงานของส่วนเน็ตเวอร์ค ส่วนควบคุมโมเด็ม และการปรับเปลี่ยน โปรแกรมต้นฉบับ จากเทอร์โบซี (Turbo C) ไปเป็นไมโครซอฟต์ซี (Microsoft C)

หลักการออกแบบโปรแกรมเลียนแบบเทอร์มินอล

โปรแกรมเลียนแบบเทอร์มินอลนั้นเป็นโปรแกรมที่ต้องทำงานในลักษณะทำงานแบบทันที (real - time) ซึ่งจะต้องจัดการการทำงานของส่วนสำคัญ 2 ส่วน คือ ส่วนของแผงแป้นอักขระ (keyboard) และส่วนของการสื่อสาร (communication) ซึ่งทั้ง 2 ส่วนนี้ ต้องทำงานในลักษณะของการทำงานแบบทันที และจะต้องประสานกันอย่างเหมาะเจาะ นั่นคือ เมื่อมีอักขระเข้ามาถึงยังส่วนของการสื่อสาร โปรแกรมเลียนแบบเทอร์มินอลจะต้องพร้อมที่จะเก็บตัวอักขระนั้นไว้ เพื่อการประมวลผลอย่างอื่นต่อไป เช่น แสดงผลอักขระที่หน้าจอ การลบหน้าจอ การส่งเสียงทางลำโพง เป็นต้น และในขณะเดียวกัน โปรแกรมจะต้องพร้อมที่จะรับอักขระจากแผงแป้นอักขระพร้อม ๆ กันไปด้วย

ในการออกแบบโปรแกรมเลียนแบบเทอร์มินอลนั้น โดยทั่วไปสามารถออกแบบได้ใน 2 ลักษณะ คือ แบบการหยั่งสัญญาณ (polling) และแบบการขัดจังหวะ (interrupt) ซึ่งทั้ง 2 วิธีการนี้มีข้อดีข้อเสียต่างกัน

1. แบบการหยั่งสัญญาณ

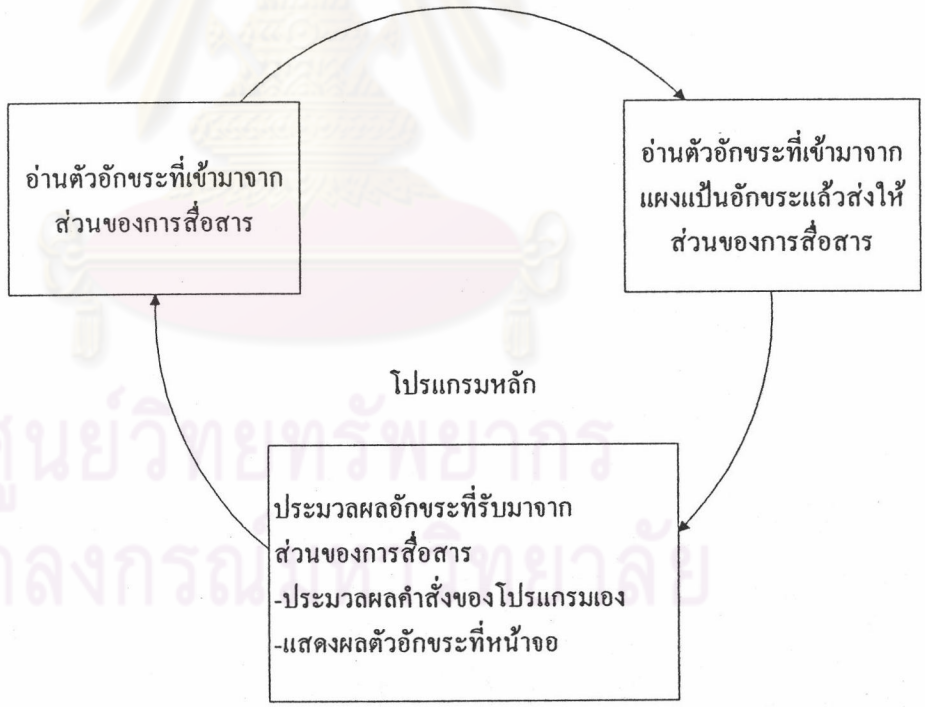
การออกแบบโปรแกรมแบบนี้จะมีหลักของโปรแกรมตามลำดับ ดังนี้

1.1 ให้บริการกับส่วนของการสื่อสาร และจัดการกับอักขระที่เข้ามา

1.2 ให้บริการส่วนของแผงแป้นอักขระ และจัดการกับชนิดของอักขระ

1.3 เขียนอักขระต่างๆ ไปยังจอภาพ หรือประมวลผลภาพตามอักขระควบคุม (Biggerstaff, 1986)

ลำดับการทำงานจะทำการวนซ้ำจนกระทั่งเลิกโปรแกรม ดังรูปที่ 3.1



รูปที่ 3.1 การออกแบบโปรแกรมเลียนแบบเทอร์มินอลแบบการหยั่งสัญญาณ

ในการออกแบบโปรแกรมเลียนแบบเทอร์มินอลในลักษณะการหยั่งสัญญาณนี้ สิ่งสำคัญคือ โปรแกรมหลักจะต้องเร็วพอที่จะบริการอักขระที่เข้ามา เพื่อไม่ให้เกิดการประมวลผลไม่ทันกัน (overrun) ดังนั้นการเขียนโปรแกรมประเภทนี้ต้องพยายามเขียนให้รหัสเครื่องสั้นที่สุดเท่าที่จะทำได้ อย่างไรก็ตามตัวอักขระบางตัวอาจจะมีการสูญหายได้ เนื่องจากในการหยั่งสัญญาณนั้น ไม่สามารถที่จะทราบได้แน่นอนว่าจะต้องสูญเสียเวลาไปทำงานส่วนอื่นนานเท่าไร ซึ่งในขณะนั้นมีอักขระเข้ามาจากการสื่อสาร ตัวอย่างเช่น ในกรณีที่มีการเพิ่มความสามารถในการรับส่งแฟ้มข้อมูลให้กับโปรแกรมเลียนแบบเทอร์มินอล เมื่อมีการรับส่งแฟ้มข้อมูลเกิดขึ้นก็จะต้องมีการอ่านและเขียนข้อมูลลงสู่จานบันทึกข้อมูลซึ่งกระทำโดยการขัดจังหวะของระบบ ซึ่งมีความสำคัญสูง ดังนั้นในขณะที่ เกิดการอ่านและเขียนข้อมูลอยู่นั้น ตัวอักขระที่เข้ามาบางส่วนของการสื่อสาร อาจมีการ สูญหายได้

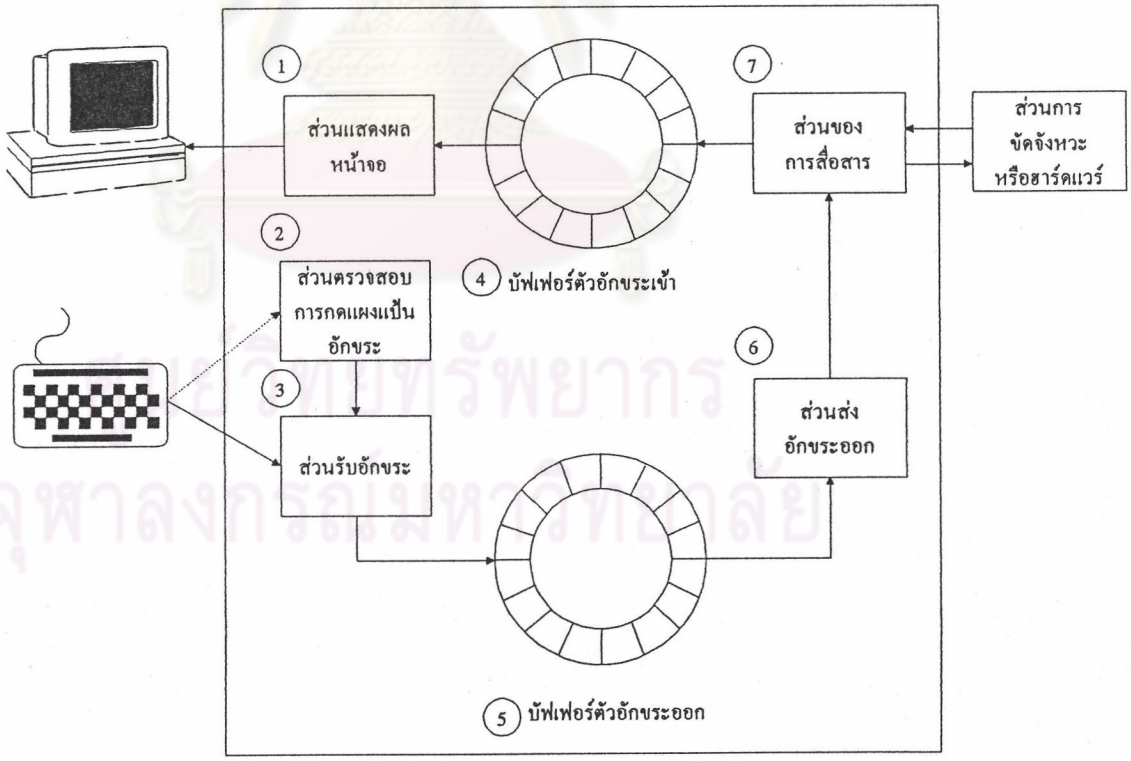
2. แบบการขัดจังหวะ

การออกแบบโปรแกรมเลียนแบบเทอร์มินอลแบบการขัดจังหวะนี้ จะถูกกระตุ้นให้ทำงานในการรับตัวอักขระจากส่วนของการสื่อสาร ส่วนของแผงแป้นอักขระ ตามการขัดจังหวะที่เกิดขึ้นเมื่อเกิดมีอักขระเข้ามา โปรแกรมจะถูกหยุดชั่วขณะเพื่อส่งการควบคุม ไปยังรูทีนบริการการขัดจังหวะนั้น เมื่อเสร็จสิ้นการทำงาน ก็จะโอนกลับการควบคุม มาสู่โปรแกรมหลักตามปกติในลักษณะเดิมที่ถูกหยุดไว้ โดยในส่วนของ รูทีนบริการ การขัดจังหวะจะเป็นรูทีนสั้น ๆ การขัดจังหวะจะเกิดขึ้นจากฮาร์ดแวร์ ตามเหตุการณ์ที่ เกิดขึ้น การสูญหายของอักขระจึงเป็นไปได้น้อยมาก ยกเว้นเสียแต่ว่าเครื่องคอมพิวเตอร์ นั้นถูกออกแบบมาโดยที่ไม่สามารถรองรับการทำงานที่มากขนาดนั้น หรือรูทีนบริการ การขัดจังหวะนั้นยาวเกินไป

ส่วนประกอบที่สำคัญของโปรแกรมเลียนแบบเทอร์มินอล

โปรแกรมเลียนแบบเทอร์มินอลสำหรับวิทยานิพนธ์นี้มีชื่อว่า CUTE ย่อมาจาก Chulalongkorn University Thai terminal Emulator มีความสามารถรับและแสดงผลภาษาไทย ใช้ได้กับการสื่อสารตามมาตราฐานอาร์เอส 232 ซี และระบบเครือข่ายที่มีโพรโตคอลทีซีพี/ไอพี โดยอาศัยแพกเก็ตไดร์เวอร์ ในการติดต่อกับเน็ตเวอร์คอินเตอร์เฟสการ์ด

ส่วนประกอบที่สำคัญของโปรแกรมเลียนแบบเทอร์มินอล CUTE ประกอบด้วยส่วนต่าง ๆ ดังรูปที่ 3.2



รูปที่ 3.2 แสดงส่วนประกอบการทำงานของโปรแกรมเลียนแบบเทอร์มินอล CUTE

1. ส่วนแสดงผลหน้าจอ

จะทำหน้าที่นำอักขระมาแสดงผลที่จอภาพ โดย สามารถแสดงผลภาษาไทย 3 ระดับได้ ในโหมดกราฟฟิก หรือปฏิบัติตามอักขระ ควบคุมตามคุณสมบัติของ VT220 การทำงานส่วนนี้อาศัยฟังก์ชั่นหรือ แมคโคร (macro) ที่สำคัญ คือ

1.1 ฟังก์ชั่น outfont

ฟังก์ชั่นนี้จะทำการตรวจสอบว่าอักขระเป็นภาษาไทยหรือภาษาอังกฤษ ตรวจสอบ ตำแหน่งที่จะนำอักขระไปแสดงผลที่จอภาพ แล้วจึงทำการย้ายแต่ละบิตของแถวลำดับตัวอักขระให้ปรากฏที่หน่วยความจำส่วนของจอภาพโดยตรง จอภาพก็จะปรากฏอักขระขึ้น ฟังก์ชั่น outfont มีรูปแบบการใช้งานดังนี้

```
outfont( unsigned char c );
```

c หมายถึงอักขระที่ต้องการนำไปแสดงผล

1.2 แมคโคร MK_FP

เป็นแมคโครในการย้ายค่าจุดในแถวลำดับของตัวอักขระไปปรากฏยังจอภาพโดยตรง โดยทำการย้ายไปยังหน่วยความจำของจอภาพโดยตรงทีละจุด แมคโครนี้ถูก กำหนดในแฟ้ม dos.h ของเทอร์โบซี 2.0 แต่ไม่ปรากฏในไมโครซอฟต์ซี 7 ดังนั้นต้องทำการกำหนดเองในโปรแกรม แมคโคร MK_FP มีรูปแบบการใช้งานดังนี้

```
#define  MK_FP( seg, ofs ) (( void far * ) \
            ((( unsigned long )( seg ) << 16 ) | ( unsigned )( ofs )))
```

seg หมายถึง แอดเดรสหน่วยความจำในส่วนเซกเมนต์ (segment)

ofs หมายถึง แอดเดรสหน่วยความจำในส่วนออฟเซ็ต (offset)

2. ส่วนตรวจสอบการกดแผงแป้นอักขระ

จะตรวจสอบว่ามีอักขระเข้ามาในบัฟเฟอร์แผงแป้นอักขระหรือไม่ โดยอาศัยการขัดจังหวะของแผงแป้นอักขระ การทำงานในส่วนนี้อาศัยฟังก์ชั่นของไมโครซอฟต์ซี

```
# include <bios.h>

unsigned _bios_keybrd (unsigned service);
```

service คือ ฟังก์ชั่นของแผงแป้นอักขระที่ต้องการใช้บริการ

การตรวจสอบการกดแผงแป้นอักขระนี้ ใช้ฟังก์ชั่นของ service เป็น _NKEYBRD_READY ซึ่งจะทำการตรวจสอบว่า มีการกดแผงแป้นอักขระแล้วเกิดอักขระขึ้นในบัฟเฟอร์ของแผงแป้นอักขระหรือไม่ ในกรณีที่มีอักขระอยู่ในบัฟเฟอร์ของแผงแป้นอักขระ จึงจะทำการอ่านค่าของอักขระและค่าของปุ่มที่กด (keystroke) จากบัฟเฟอร์อีกครั้ง

สาเหตุที่เลือกใช้ฟังก์ชั่น _bios_keybrd นี้ เนื่องจากฟังก์ชั่นนี้เป็นฟังก์ชั่นของระบบปฏิบัติการซึ่งใช้หลักการการขัดจังหวะ ซึ่งตรงกับการใช้ INT 0x16 ของ bios และสามารถตรวจสอบค่าของปุ่มที่กดได้อีกด้วย ซึ่งสามารถนำมาตรวจสอบเพื่อทำการ ประมวลผลในส่วนของโปรแกรมเอง เช่น ตรวจสอบค่าของปุ่มที่กดว่าเท่ากับ ALT-S ก็ทำการเรียกรูทีนส่วนกำหนดค่า (Setup) ขึ้นมา เพื่อให้ผู้ใช้ทำการกำหนดค่าต่าง ๆ ของโปรแกรมได้ตามต้องการ

3. ส่วนรับอักขระ

สืบเนื่องจากในส่วนของการตรวจสอบการกดแผงแป้น อักขระ เมื่อตรวจสอบพบว่ามีอักขระอยู่ในบัฟเฟอร์ของแผงแป้นอักขระ การทำงานส่วนนี้ จะทำการอ่านค่าอักขระและค่าของปุ่มที่กดจากบัฟเฟอร์ของแผงแป้นอักขระ เพื่อทำการส่ง ไปยังบัฟเฟอร์ตัวอักขระออก ฟังก์ชั่นที่เรียกใช้ในส่วนนี้เป็นฟังก์ชั่น _bios_keybrd เช่นกัน แต่เปลี่ยนค่า service เป็น _KEYBRD_READ ค่าที่อ่านจากบัฟเฟอร์ของแผงแป้นอักขระจะประกอบด้วย 2 ส่วน คือ ค่าของปุ่มที่กด และอักขระที่ได้จากการกด ค่าของปุ่มที่กดจะใช้ในการตรวจสอบเพื่อเรียกรูทีนประมวลผลในส่วนของ โปรแกรมเอง เช่น เรียกรายการการกำหนดค่า (setup menu) เป็นต้น สำหรับอักขระที่ อ่านได้จะส่งไปยังรูทีนการเปลี่ยนรหัสเป็นภาษาไทย ในกรณีที่โปรแกรมอยู่ในสถานะภาษาไทย แล้วจึงส่งไปเก็บในบัฟเฟอร์ตัวอักขระออก

4. บัฟเฟอร์ตัวอักขระเข้า

เป็นบัฟเฟอร์ของแถวลำดับที่เป็นการเรียงต่อเป็น แบบวงกลม (circular queue) ซึ่งจะมีตัวชี้ตำแหน่งเริ่มต้นของอักขระในบัฟเฟอร์ ตัวชี้ ตำแหน่งสุดท้ายของอักขระ และตัวบอกจำนวนอักขระที่มีอยู่ในบัฟเฟอร์ สาเหตุที่ต้องมี ตัวบอกจำนวนอักขระที่มีอยู่ในบัฟเฟอร์ก็เพื่อป้องกันการเข้าใจผิด ในกรณีที่ตัวชี้ตำแหน่งเริ่มต้นอักขระ และตำแหน่งสุดท้ายอยู่ในตำแหน่งเดียวกัน ซึ่งอาจจะหมายถึงไม่มีจำนวนอักขระอยู่ในบัฟเฟอร์เลย หรือมีอักขระอยู่ในบัฟเฟอร์จนเต็ม

การเพิ่มอักขระเข้าไปยังแถวลำดับที่เป็นการเรียงต่อเป็นแบบวงกลม จะต้องเพิ่มอักขระเข้าไปตอนท้ายของแถวลำดับ ปรับปรุงตัวชี้ตำแหน่งสุดท้าย และจำนวนอักขระ ให้ถูกต้องก่อนที่จะทำการเพิ่มอักขระเข้าไปยังบัฟเฟอร์นั้น จะต้องป้องกันการขัดจังหวะ ของระบบชั่วขณะ แล้วจึงทำการเปิดการขัดจังหวะอีกครั้ง เพื่อป้องกันไม่ให้ระหว่างที่ เพิ่มอักขระเข้าไปยังบัฟเฟอร์อักขระเข้า มีการขัดจังหวะอื่นเกิดขึ้น ฟังก์ชั่นในการปิดกั้น และเปิดการขัดจังหวะ เป็นฟังก์ชั่นของไมโครซอฟต์ซี คือ

```
# include <dos.h>

void _disable (void);
void _enable (void);
```

ทั้งสองฟังก์ชั่นจะเรียกใช้คำสั่งเครื่อง CLI และ STI ของรหัสเครื่อง 8086 ตามลำดับ

5. บัฟเฟอร์ตัวอักขระออก

เป็นบัฟเฟอร์ของแถวลำดับ เช่นเดียวกับบัฟเฟอร์ ตัวอักขระเข้า แต่จะเป็นบัฟเฟอร์ที่เก็บอักขระที่จะส่งออกไปยังส่วนของการสื่อสาร

6. ส่วนส่งอักขระออก

ทำหน้าที่ในการนำตัวอักขระในบัฟเฟอร์ตัวอักขระออก ส่งไปยังส่วนของการสื่อสาร และปรับปรุงค่าตัวชี้ตำแหน่งเริ่มต้น ตำแหน่งสุดท้ายของ อักขระ และจำนวนอักขระที่มีอยู่ในบัฟเฟอร์

7. ส่วนของการสื่อสาร

ทำหน้าที่เริ่มต้นและสิ้นสุดการสื่อสาร คอยตรวจสอบ เหตุการณ์และให้บริการด้านการสื่อสารต่าง ๆ เช่น ส่งอักขระออก รับอักขระเข้ามา เป็นต้น

ส่วนของการสื่อสารประกอบด้วยส่วนสำคัญ 2 ส่วน คือ

7.1 ส่วนที่ทำงานกับมาตรฐานอาร์เอส 232 ซี

การทำงานในส่วนนี้อาศัยหลักการขัดจังหวะ โดยมี ไอซี 8250 เป็นตัวควบคุมเกี่ยวกับการรับและส่งข้อมูล (สมนึก เจียมเจริญเดช, 2533)

7.2 ส่วนที่ทำงานกับเครือข่ายที่มีโพรโตคอลทีซีพี/ไอพี

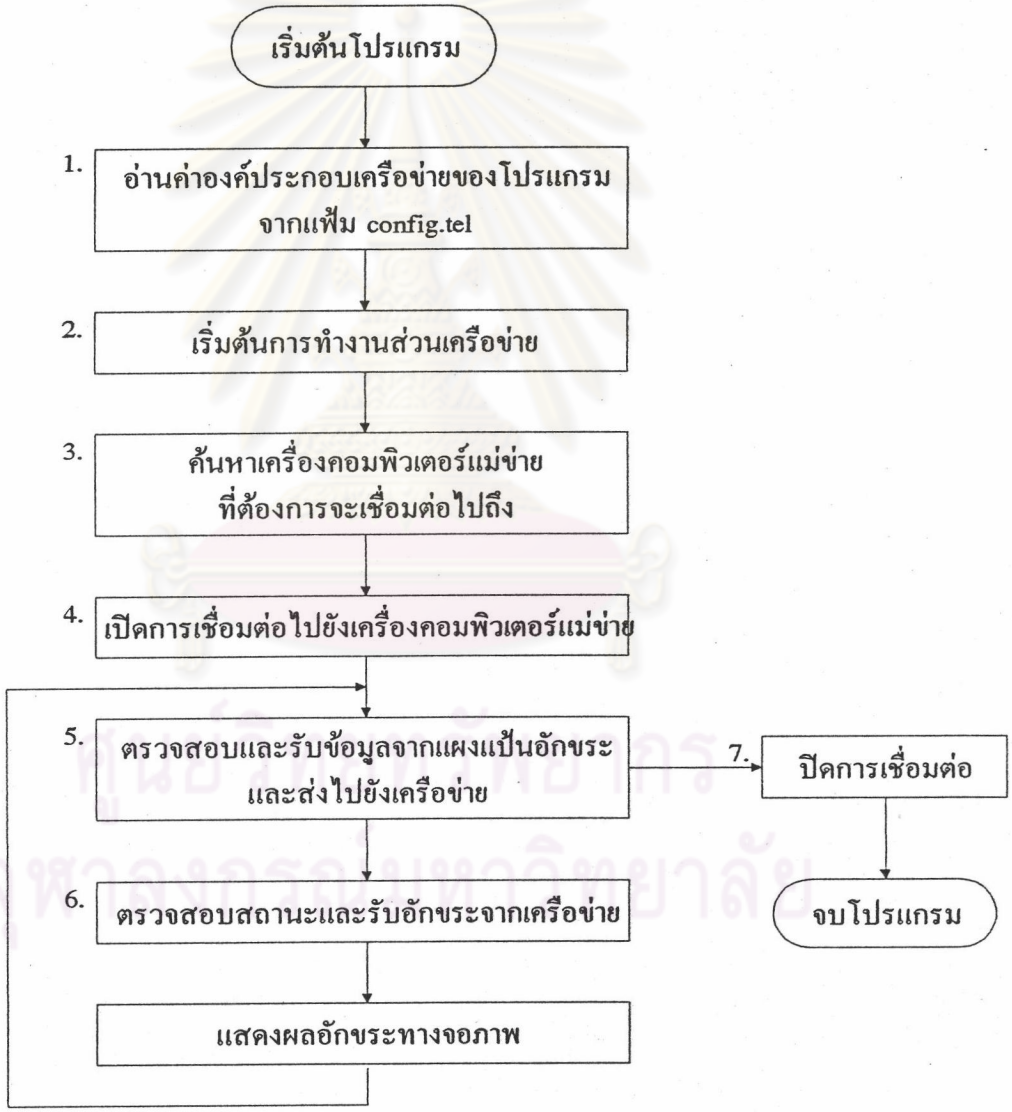
ส่วนของการสื่อสารที่ทำงานกับระบบเครือข่ายที่มีโพรโตคอลทีซีพี/ไอพี นั้น อาศัยโปรแกรมเฉพาะงานจากคลังของ NCSA Telnet version 2.307 ซึ่งทำหน้าที่ดูแลจัดการในระดับชั้นอินเตอร์เน็ต และทรานสปอร์ต รวมเรียกว่า กองซ้อนโพรโตคอลทีซีพี/ไอพี (TCP/IP Protocol Stack) ซึ่งทำงานร่วมกับแพกเก็ตไดรเวอร์ มาใช้ในการพัฒนาส่วนเชื่อมต่อนี้

8. ส่วนการขัดจังหวะหรือฮาร์ดแวร์

คือส่วนของฮาร์ดแวร์ที่ควบคุมการสื่อสาร ในกรณีของมาตรฐานอาร์เอส 232 ซี จะหมายถึง Asynchronous Communications Element (ACE) ในที่นี้จะหมายถึงชิพ 8250 แต่ถ้าเป็นลักษณะของเครือข่าย จะหมายถึง เน็ตเวอร์คอินเตอร์เฟสการ์ด ซึ่งอาจจะเป็นอีเทอร์เน็ต หรือโทคเคนริง หรืออื่น ๆ ที่มีแพกเก็ตไดรเวอร์เป็นตัวขับ

ส่วนของการสื่อสารที่ทำงานกับเครือข่ายที่มีโพรโตคอล ทีซีพี/ไอพี

ดังได้กล่าวไว้แล้ว การพัฒนาส่วนเชื่อมต่อโปรแกรมเลียนแบบเทอร์มินอล ภาษาไทยเข้ากับโพรโตคอล ทีซีพี/ไอพี นั้น อาศัยโปรแกรมเฉพาะงานจากคลังของ NCSA Telnet 2.307 ซึ่งประกอบด้วยฟังก์ชั่นต่าง ๆ ที่สำคัญ โดยจะกล่าวเป็นลำดับตามขั้นตอน ที่ใช้ประกอบลงในโปรแกรมเลียนแบบเทอร์มินอล CUTE ดังรูปที่ 3.3



รูปที่ 3.3 แสดงขั้นตอนการทำงานของโปรแกรม CUTE ในส่วนเชื่อมต่อกับโพรโตคอล ทีซีพี/ไอพี

1. อ่านค่าองค์ประกอบส่วนเครือข่ายของโปรแกรม

เป็นการอ่านค่าองค์ประกอบต่าง ๆ ที่จำเป็นสำหรับเครือข่ายของโพรโตคอลทีซีพี/ไอพี จากแฟ้ม config.tel เพื่อเก็บค่าเหล่านี้ไว้ในตัวแปรกำหนดค่า

2. เริ่มต้นการทำงานส่วนเครือข่าย

เป็นส่วนกำหนดค่าเริ่มต้นต่าง ๆ ให้กับตัวแปร การกำหนดชนิดของฮาร์ดแวร์ที่ใช้ ไอพีแอดเดรส, ไอพีสับเน็ตมาส การเริ่มต้นลำดับเหตุการณ์ (event queue) ข้อมูลและข่าวสารในระหว่างการเชื่อมต่อ การทำงานของ RARP

ฟังก์ชั่น Snetinit

เป็นฟังก์ชั่นเริ่มต้นการทำงานกับส่วนเครือข่าย กำหนดลำดับเวลา (timer queue) ค่าที่ส่งกลับจะไม่เท่ากับ 0 ในกรณีที่มีข้อผิดพลาดเกิดขึ้น RARP จะถูกทำงานในฟังก์ชั่นนี้ และจะส่งค่ากลับเป็น -2 เมื่อ RARP ล้มเหลว ฟังก์ชั่น Snetinit มีรูปแบบการใช้ดังนี้

```
int  Snetinit ( )
```

ฟังก์ชั่น Snetinit ยังเรียกใช้ฟังก์ชั่นอื่นอีกดังต่อไปนี้

2.1 ฟังก์ชั่น Sreadhosts อ่านค่าองค์ประกอบของโฮส (host) จากแฟ้ม config.tel เพื่อเก็บไว้ใน โครงสร้างข้อมูลของ machinfo และกำหนดค่าทางเลือกบางประการสำหรับ ftp, rcp และ อื่น ๆ

2.2 ฟังก์ชั่น netinit กำหนดฮาร์ดแวร์ และกำหนดค่าเริ่มต้นให้กับตัวแปรต่างๆ เช่น ไอพีแอดเดรส, ไอพีสับเน็ตมาส

2.3 ฟังก์ชั่น Ssetgates กำหนดเน็ตมาส (netmask) ซึ่งอ่านค่าได้จากแฟ้ม config.tel และเปิด ftp และ rcp ถ้ามีการกำหนดการใช้ไว้ในแฟ้ม config.tel และกำหนดโฮสที่เป็นเกตเวย์ (gateway)

2.4 ฟังก์ชั่น neteventinit การเริ่มต้นลำดับเหตุการณ์ (event queue) เพื่อใช้ในการตรวจสอบ สถานะในระหว่างการเชื่อมต่อ

3. ค้นหาเครื่องคอมพิวเตอร์แม่ข่ายที่ต้องการจะเชื่อมต่อ

ในการเชื่อมต่อเพื่อเป็นเทอร์มินอลหนึ่งของเครื่องคอมพิวเตอร์แม่ข่าย ในการเรียกใช้โปรแกรมเลียนแบบเทอร์มินอลภาษาไทย CUTE นั้น จะต้องระบุชื่อของ เครื่องคอมพิวเตอร์แม่ข่ายซึ่งกำหนดไว้แล้วในแฟ้ม config.tel เป็นอาร์กิวเมนต์ การทำงาน ในส่วนนี้จึงเป็นการค้นหาเครื่องคอมพิวเตอร์แม่ข่ายที่ต้องการจะเชื่อมต่อในเครือข่าย ซึ่งเรียกใช้ฟังก์ชั่น Sgethost

3.1 ฟังก์ชั่น Sgethost

นำชื่อของเครื่องคอมพิวเตอร์แม่ข่าย หรือไอพีแอดเดรส ที่ต้องการจะเชื่อมต่อมา เพื่อหาส่วนประกอบหรือคุณลักษณะของเครื่องคอมพิวเตอร์แม่ข่ายนั้น ๆ การเรียกใช้ฟังก์ชั่น มีรูปแบบดังนี้

```
#include hostform.h

mp = Sgethost (name)
```

```
char *name;              /* name or IP number of machine to get
                            information about */
struct machinfo *mp;     /* Pointer to machine information record */
```

อาร์กิวเมนต์เป็นชื่อเครื่องคอมพิวเตอร์แม่ข่ายที่จะเชื่อมต่อ ส่วนค่าที่ส่งกลับจะเป็นตัวชี้ที่ชี้ไปยังโครงสร้างข้อมูลของเครื่องคอมพิวเตอร์แม่ข่ายนั้น ๆ และจะเท่ากับ NULL ในกรณีที่ไม่พบ ฟังก์ชั่นนี้จะเรียกใช้ฟังก์ชั่น Shostlook ในกรณีที่อาร์กิวเมนต์ ที่ใช้ไม่อยู่ในรูปของไอพีแอดเดรส โครงสร้างข้อมูลของ machinfo ประกอบด้วย

```
struct machinfo  {
    unsigned  char  *sname,      /* pointer to name of session */
              *hname,            /* pointer to name of that machine */
              *font,             /* font name, if we can do it */
              *ftpoptions,       /* options for FTP comman line (ALT-F) */
              hostip [4],        /* IP number of this machine */
              gateway,           /* gateway preference, start with 1 */
              nameserv,          /* nameserver preference, start with 1 */
              bksp,              /* backspace value */
              halfdup,           /* half duplex required */
              crmap,             /* Strange Berkeley 4.3 CR mode needed */
              fsize,             /* font size in points /*
              vtwrap,            /* flag on when need wrap mode */
              vtwidth,           /* how wide screen shouldd be for this session */
    char      nfcolor,           /* normal foreground */
              nbcolor,           /* normal background */
              bfcolor,           /* blink foreground */
              bbcolor,           /* blink background */
              ufcolor,           /* underline forground */
```

```
        ubcolor,            /* underline background */
int     clearsave,          /* whether to save cleared lines */
        port,               /* TCP port number to access, default=23
                               (telnet) */
        mno,                /* machine number for reference */
        mstart,             /* status of this machine entry */
        bkscroll,           /* how many lines to save */
        retrans,            /* initial retrans timeout */
        conto,              /* time out in seconds to wait for connect */
        window,             /* window, will be checked against buffers */
        maxseg,             /* maximum receivable segment size */
        mtu,                /* maximum transfer unit MTU (out) */
unsigned int mapoutflag : 1, /* flag to indicate whether output mapping is on
                                for this machine */
consoledebug : 2 ;          /* debugging levels for displaying information
                               on the console */
struct machinfo *next ;     /* surprise, its a linked list ! */
} ;
```

3.2 ฟังก์ชั่น Sdomain

ในการค้นหาเครื่องคอมพิวเตอร์แม่ข่าย เพื่อที่จะเชื่อมต่อไปถึงนั้น ยังสามารถค้นหาได้จาก โดเมนเนมเซอร์ฟเวอร์ (Domain Name Server) ซึ่งให้บริการค้นหาไอพีแอดเดรส ของเครื่องคอมพิวเตอร์แม่ข่ายที่จะต้องการเชื่อมต่อไปถึง โดยเรียกใช้ฟังก์ชั่น Sdomain ฟังก์ชั่นนี้จะค้นหาไอพีแอดเดรส จากเครื่องคอมพิวเตอร์ที่ทำหน้าที่เป็นเนมเซอร์ฟเวอร์ ซึ่งระบุไว้ในแฟ้ม config.tel หลังจากนั้น การค้นหาจะได้ค่าของไอพีแอดเดรส กลับมา โดยจะต้องตรวจสอบจากเหตุการณ์จากฟังก์ชั่น Sgetevent ซึ่งจะกล่าวถึงต่อไป การใช้งานฟังก์ชั่นนี้ มีรูปแบบดังนี้

```
Sdomain (name)
char *name;                    /* name of the machine to look up */
```

หลังจากนั้นจึงเรียกใช้ฟังก์ชั่น Slooknum เพื่อจะได้ค่าตัวชี้ ที่ชี้ไปยังโครงสร้างของ machinfo อีกที

4. เปิดการเชื่อมต่อไปยังเครื่องคอมพิวเตอร์แม่ข่าย

หลังจากที่ได้ตัวชี้ตำแหน่งที่ชี้ไปยังโครงสร้างข้อมูลของเครื่องคอมพิวเตอร์แม่ข่ายดังกล่าวแล้ว ในลำดับต่อไปจะเป็นการเปิดการเชื่อมต่อไปยังเครื่องคอมพิวเตอร์แม่ข่ายนั้น โดยจะทำการเชื่อมต่อไปยังช่องทางการสื่อสาร (port), คือ ทีซีพีพอร์ต 23 (TCP port 23) ซึ่งเป็นหมายเลขพอร์ตของเทลเน็ต (telnet port) ฟังก์ชั่นที่ทำการเรียกใช้ คือ Snetopen

ฟังก์ชั่น Snetopen

ทำหน้าที่เปิดการเชื่อมต่อไปยังเครื่องคอมพิวเตอร์แม่ข่ายและพอร์ตที่กำหนด ซึ่งจะให้ค่าที่ส่งกลับมาเป็นตัวบอกพอร์ต (port descriptor) เพื่อใช้ในการอ่านและเขียน โดยฟังก์ชั่น netread และ netwrite ในภายหลัง มีการกำหนดเวลาสำหรับระยะเวลาการสร้างการเชื่อมต่อ (connection timeout) ซึ่งจะถูกใช้ภายหลัง หลังจากที่เปิดการเชื่อมต่อแล้วยังไม่สามารถที่จะเข้าถึงหรือใช้งานได้ จนกว่าจะได้รับผลที่ตอบกลับจากเหตุการณ์ ในการเรียกใช้ฟังก์ชั่น Sgetevent เพื่อจะทราบว่า การเปิดการเชื่อมต่อสำเร็จหรือไม่ การเรียกใช้ฟังก์ชั่น มีรูปแบบดังนี้

```
pn = Snetopen (mp, tport)
struct machininfo *mp ;
int tport ;
int pn;
```

mp หมายถึงตัวชี้ที่ชี้ไปยังโครงสร้างข้อมูลแบบ machinfo

tport หมายถึงช่องทางการสื่อสาร หรือ ทีซีพีพอร์ต

pn หมายถึงตัวบอกพอร์ต

5. ตรวจสอบและรับข้อมูลจากแผงแป้นอักขระและส่งไปยังเครือข่าย

การตรวจสอบและรับข้อมูลจากแผงแป้นอักขระ อาศัยการขัดจังหวะของแผงแป้นอักขระ ซึ่งได้กล่าวไว้แล้วในหัวข้อส่วนประกอบที่สำคัญของโปรแกรมเลียนแบบเทอร์มินอล ส่วนการตรวจสอบแผงแป้นอักขระ หลังจากที่ข้อมูลจากแผงแป้นอักขระถูกเก็บไว้ในส่วนบัฟเฟอร์ตัวอักขระเข้าแล้ว จึงจะถูกส่งไปยังส่วนของการสื่อสาร ซึ่งประกอบด้วยฟังก์ชั่น netwrite ซึ่งมีรูปแบบการใช้ดังนี้

```
cnt = netwrite (pn, buf, len)
int pn ;                    /* port number from netopen( ) */
char *buf ;                 /* pointer to data space to write from */
int len ;                   /* length of data to try to write */
int cnt ;                   /* return # of bytes written, -1 on closed
                               connection, 0 for waiting */
```

การใช้งานของฟังก์ชั่นจะเหมือนกับการใช้ฟังก์ชั่น write ของระบบยูนิกส์ แต่แทนที่จะเป็นตัวบอกแฟ้มข้อมูล (file descripter) ก็กลายเป็นตัวบอกพอร์ตที่ได้จากการเปิดการเชื่อมต่อด้วยฟังก์ชั่น netopen ความหมายของแต่ละอาร์กูเมนต์ ประกอบด้วย

pn หมายถึงตัวบอกพอร์ตที่ได้จากฟังก์ชั่น netopen

*buf หมายถึงตัวชี้ที่ชี้ไปยังบัฟเฟอร์ที่ส่งข้อมูลออกไป

len ความยาวของข้อมูลที่พยายามจะส่ง

cnt ค่าส่งกลับที่จะบอกถึงจำนวนไบท์ที่ส่งไปได้จริง
ถ้าเท่ากับ -1 หมายถึงการเชื่อมต่อถูกปิดลง

ถ้าเท่ากับ 0 แสดงว่าเกิดการรอ

6. ตรวจสอบสถานะและรับอักขระจากเครือข่าย

การตรวจสอบสถานะของตัวอักขระที่จะเข้ามา ข้อผิดพลาดที่เกิดขึ้น หรือ สถานะการเชื่อมต่อของเครือข่ายนั้น โปรแกรมเฉพาะงานจากคลังของ NCSA Telnet ได้กำหนดเป็นลักษณะของชั้น (classes) และเหตุการณ์ (events) เพื่อจัดการหรือ ตอบสนองเหตุการณ์ดังกล่าวได้ถูกต้อง การตรวจสอบอาศัยฟังก์ชั่น Sgetevent

6.1 ฟังก์ชั่น Sgetevent

เป็นฟังก์ชั่นที่ทำงานในลักษณะของการประมวลผลเบื้องหลัง (background processing) ของ ftp, rcp และการค้นหาโดเมนเนม (domain name) เหตุการณ์ต่าง ๆ จะถูกส่งโดยฟังก์ชั่น netputevent และโดยรูทีนอื่นในระดับล่างลงไป การใช้งานฟังก์ชั่นนี้มีรูปแบบดังนี้

```
#include  netevent.h

thevent = Sgetevent (class, theclass, dat)
int class;          /* Classes to search for (or combination) */
int *theclass;      /* Actual class of the event returned
                       (return value) */
int *dat;           /* Data which tags the event */
int thevent;        /* Which event has occurred */
```

อาร์กิวเมนต์ตัวแรก *class หมายถึงชั้นที่ต้องการจะสอบถาม ซึ่ง สามารถที่จะ สอบถามหลาย ๆ ชั้นที่สนใจได้ โดยใช้การ OR ในการส่งค่าให้กับฟังก์ชั่น

อาร์กิวเมนต์ตัวที่ 2 *theclass เป็นค่าที่ส่งกลับมาของชั้นเหตุการณ์ ที่เกิดขึ้นจริง

อาร์กิวเมนต์ตัวที่ 3 *dat เป็นข้อมูลที่ประกอบเพิ่มเติมของชั้นหรือเหตุการณ์นั้น

ค่าที่ส่งกลับมาของการเรียกใช้ฟังก์ชั่นจะเป็นเหตุการณ์ที่เกิดขึ้น

ในกรณี ที่ไม่เกิดเหตุการณ์ใด ๆ ค่าดังกล่าวจะเท่ากับค่า 0 ค่าและความหมายของแต่ละชั้น และ เหตุการณ์ ถูกกำหนดไว้ในแฟ้ม netevent.h ซึ่งประกอบด้วย

```
#define  USERCLASS  1
#define  ICMPCLASS  2
#define  ERRCLASS   4
#define  SCLASS     8
#define  CONCLASS   0x10
#define  ERR1       1    /* an error message is waiting,
                            ERRCLASS */
#define  IREDIR     1    /* ICMP redirect, ICMPCLASS */
#define  CONOPEN    1    /* connection has opened, CONCLASS */
#define  CONDATA    2    /* there is data available on this
                            connection */
#define  CONCLOSE   3    /* the other side has closed its side of the
                            connection */
#define  CONFAIL    4    /* connection open attempt has failed */
#define  UDPDATA    1    /* UDP data has arrived on listening port,
                            USERCLASS */
#define  DOMOK      2    /* domain name ready */
```

```
#define  DOMFAIL    3     /* domain name lookup failed */
#define  FTPCOPEN   20    /*FTP command connection has opened*/
#define  FTPCLOSE   21    /*FTP command connection has closed*/
#define  FTPBEGIN   22    /* FTP transfer beginning, dat=1 for get,
                             0 for put */
#define  FTPEND     23    /* FTP transfer ending */
#define  FTPLIST    24    /* FTP file listing taking place */
#define  FTPUSER    25    /* FTP user name has been entered */
#define  FTPPWOK    26    /* FTP password verified */
#define  FTPPWNO    27    /* FTP password failed */
#define  RCPBEGIN   30    /* RCP beginning */
#define  RCPEND     31    /* RCP ending */
#define  UDPTO      1     /* UDP request from DOMAIN timed out
                             SCLASS */
#define  FTPACT     2     /* FTP transfer is active, keep sending */
#define  TCPTO      3     /* TCP for DOMAIN timed out */
#define  RCPACT     4     /* rcp is active, needs CPU time */
#define  RETRYCON   5     /*retry connection packet, might be last*/
```

ความหมายพอสังเขปของแต่ละชั้นและเหตุการณ์ประกอบด้วย

6.1.1 ชั้น USERCLASS ประกอบด้วยเหตุการณ์ต่าง ๆ ที่สำคัญดังนี้

6.1.1.1 เหตุการณ์ DOMOK

เป็นเหตุการณ์ที่บอกถึงโดเมนเนมเซอร์เวอร์ได้ค้นหาไอพีแอดเดรสได้ และส่งมาให้ในอาร์กูเมนต์ dat

6.1.1.2 เหตุการณ์ DOMFAIL

เมื่อเหตุการณ์นี้เกิดขึ้นแสดงว่าการค้นหาชื่อของโดเมนเนมเซอร์เวอร์ไม่สำเร็จ และต่อมาจะมีเหตุการณ์ของการผิดพลาดในชั้น ERRCLASS ตามมาด้วย ซึ่งจะกล่าวถึงต่อไป

6.1.2 ชั้น ERRCLASS ประกอบด้วยเหตุการณ์เดียวคือ

เหตุการณ์ ERR1

จะเกิดขึ้นเมื่อการเชื่อมต่อมีข้อผิดพลาดขึ้น พร้อมทั้งมีตัวเลข ส่งกลับมาที่อาร์กูเมนต์ dat ซึ่งสามารถใช้ฟังก์ชั่น neterrstring ในการแสดงข้อความการผิดพลาดออกมาที่จอภาพได้

6.1.3 ชั้น CONCLASS เป็นชั้นที่แสดงถึงสถานะการเชื่อมต่อว่าเป็น อย่างไร ซึ่งประกอบด้วยเหตุการณ์ที่สำคัญดังนี้

6.1.3.1 เหตุการณ์ CONOPEN

หมายถึงมีการเปิดการเชื่อมต่อเกิดขึ้น ซึ่งเป็นผลของการใช้คำสั่ง Snetopen ค่าที่ส่งกลับมาที่อาร์กูเมนต์ dat จะเป็นค่าของตัวบอกพอร์ตที่เกิดการเชื่อมต่อ

6.1.3.2 เหตุการณ์ CONDATA

มีข้อมูลเข้ามาจากพอร์ตที่ทำการเปิดการเชื่อมต่ออยู่ ค่าอาร์กูเมนต์ dat ยังคงเป็นค่าของตัวบอกพอร์ตที่มีข้อมูลมาถึง เมื่อเกิดเหตุการณ์ดังกล่าว ขึ้นจึงจะทำการอ่านข้อมูลจากบัฟเฟอร์ของทีซีพี ด้วยฟังก์ชั่น netread ซึ่งจะกล่าวถึง รายละเอียดต่อไป

6.1.3.3 เหตุการณ์ CONFAIL

เป็นเหตุการณ์ที่บอกถึงการเชื่อมต่อล้มเหลว เนื่องจากใช้ระยะ เวลาในการพยายามเชื่อมต่อนานเกินกว่าเวลากำหนด เมื่อเกิดเหตุการณ์นี้จะต้องทำการต่อไป

6.1.3.4 เหตุการณ์ CONCLOSE

เป็นเหตุการณ์ที่เกิดจากการปิดการเชื่อมต่อของเครื่องคอมพิวเตอร์แม่ข่ายซึ่งกำลังเชื่อมต่ออยู่ เมื่อเกิดเหตุการณ์นี้ จะต้องทำการอ่านข้อมูลที่ถูกส่งมาแล้วให้หมดก่อนด้วยฟังก์ชั่น netread จนกระทั่งค่าที่ส่งกลับมาเป็น 0 หรือ -1 จึงจะทำการปิดการเชื่อมต่อด้วยฟังก์ชั่น netclose ต่อไป

6.2 ฟังก์ชั่น netread

เป็นฟังก์ชั่นในการอ่านข้อมูลที่เข้ามาจากบัฟเฟอร์ของ ทีซีพี การใช้งานฟังก์ชั่นนี้ จะต้องตรวจสอบเหตุการณ์ที่เกิดขึ้นก่อน ดังหัวข้อ 6.1.3.1 สำหรับการทำงานของโปรแกรม CUTE นี้ จะทำการวนอ่านข้อมูลจากบัฟเฟอร์ของ ทีซีพี แล้วมาใส่ไว้ในบัฟเฟอร์ของส่วนข้อมูลเข้าอีกที ลักษณะการใช้งานฟังก์ชั่นนี้ มีรูปแบบดังนี้

```
cnt = netread (pn, buf, len)
int pn;      /* port number from netopen( ) */
char *buf;   /* pointer to data space to read into */
int len;     /* maximum length of data to read */
int cnt;     /* returns # of bytes read, -1 on closed connection, 0 for
                waiting */
```

ลักษณะการใช้งาน จะเหมือนกับฟังก์ชั่น read ของระบบยูนิกส์ แต่แทนที่จะเป็นตัวบอกแฟ้มข้อมูล (file description) ก็กลายเป็นตัวบอกพอร์ตที่ได้จากการเปิดการเชื่อมต่อด้วยฟังก์ชั่น netopen ความหมายของแต่ละอาร์กูเมนท์ ประกอบด้วย

pn หมายถึงตัวบอกพอร์ต

*buf หมายถึงตัวชี้ที่ชี้ไปยังบัฟเฟอร์ที่จะเก็บข้อมูลจากการอ่านชั่วคราว

len ความยาวของข้อมูลที่พยายามจะอ่าน

cnt ค่าส่งกลับซึ่งจะบอกถึงจำนวนไบท์ที่อ่านได้จริง ถ้าเท่ากับ -1 หมายถึง การเชื่อมต่อถูกปิดลง ถ้าเป็น 0 แสดงว่าเกิดการรอ

7. ปิดการเชื่อมต่อ

เป็นส่วนที่ทำการปิดการเชื่อมต่อ เพื่อจะหยุดการรับส่งข้อมูล ซึ่งประกอบด้วยฟังก์ชั่นต่อไปนี้

7.1 ฟังก์ชั่น netclose

เป็นฟังก์ชั่นในการปิดการเชื่อมต่อ คล้าย ๆ กับการปิดแฟ้มข้อมูล ข้อสำคัญคือ ต้องทำการผลักข้อมูลที่ส่งไปได้เรียบร้อยเสียก่อน ด้วยฟังก์ชั่น netpush สำหรับรูปแบบการใช้งาน netclose ประกอบด้วย

```
netclose(pn)
int pn;                    /* port number from netopen( ) */
```

อาร์กูเมนท์ pn หมายถึงตัวบอกพอร์ตที่ได้ทำการเปิดไว้แล้ว

7.2 ฟังก์ชั่น netshut

เป็นฟังก์ชั่นในการหยุดการติดต่อสื่อสาร ปิดการขัดจังหวะ ฟังก์ชั่นนี้จึงเป็นฟังก์ชั่นสุดท้ายก่อนการเลิกโปรแกรม

ส่วนควบคุมการทำงานของโมเด็ม

ดังได้กล่าวไว้แล้วว่า โปรแกรมเลียนแบบเทอร์มินอลภาษาไทย CUTE นี้ สามารถทำงานได้กับมาตรฐานอาร์เอส 232 ซี เมื่อต้องการที่จะใช้เทอร์มินอลในระยะไกล จำเป็นที่จะต้องใช้โมเด็มในการเชื่อมต่อ ดังนั้น เพื่อความสะดวกในการใช้งานโปรแกรม จึงได้เพิ่มเติมส่วนควบคุมการทำงานของโมเด็มเข้าไป โดยอาศัยวิธีการส่งคำสั่ง AT ซึ่งเป็นมาตรฐานที่โมเด็มสามารถรับรู้ได้ วิธีการก็คือ เพิ่มส่วนที่ให้ผู้ใช้สามารถป้อนและเก็บหมายเลขโทรศัพท์และชื่อสถานที่ หรือคำบรรยายของหมายเลขโทรศัพท์นั้น หลังจากนั้นเมื่อต้องการที่จะหมุนโมเด็มเพื่อติดต่อระยะไกล ไปยังเครื่องคอมพิวเตอร์แม่ข่าย ก็เพียงแต่ส่งคำสั่ง AT ในการหมุนโทรศัพท์ และตามด้วยหมายเลขโทรศัพท์ที่ต้องการ ที่ได้กำหนดไว้แล้ว

การปรับเปลี่ยนโปรแกรมต้นฉบับจากเทอร์โบซี ไปเป็นไมโครซอฟต์ซี

สืบเนื่องจากโปรแกรมเลียนแบบเทอร์มินอลภาษาไทย CUTE ในรุ่นแรกนั้น พัฒนาโดยใช้เทอร์โบซี ซึ่งทำงานกับมาตรฐานอาร์เอส 232 ซี นั้น เมื่อนำโปรแกรมต้นฉบับเดิมมาพัฒนา เพื่อให้สามารถติดต่อกับเครือข่ายที่มีโพรโตคอลทีซีพี/ไอพี ประสบปัญหาเนื่องจากโปรแกรมเฉพาะงานจากคลังของ NCSA Telnet นั้น พัฒนาโดยใช้ไมโครซอฟต์ซี ดังนั้น เพื่อให้สามารถประกอบโปรแกรม 2 ส่วนเข้าด้วยกัน จึงจำเป็นที่จะต้องเลือกเปลี่ยนโปรแกรมต้นฉบับ ให้เข้ากับโปรแกรมเฉพาะงานจากคลังของ NCSA Telnet สาเหตุที่ทำการเปลี่ยนโปรแกรมต้นฉบับจากเทอร์โบซี ไปเป็นไมโครซอฟต์ซี แทนที่จะเปลี่ยนจากไมโครซอฟต์ซี มาเป็นเทอร์โบซี ก็เพราะว่า ประการแรกการเปลี่ยน โปรแกรมเฉพาะงานจากคลังของ NCSA Telnet อาจทำให้เกิดการผิดพลาดขึ้นได้ ซึ่งยากต่อการแก้ไขกว่าการที่จะเปลี่ยนโปรแกรมต้นฉบับของ CUTE เอง ประการที่สอง ไมโครซอฟต์ซีเป็นตัวแปลภาษาที่ใหม่กว่าและค่อนข้างจะให้คำสั่งกระทำการ (execute code) ที่ดีกว่า

ฟังก์ชั่นหลัก ๆ ในการเปลี่ยนจากเทอร์โบซี ไปเป็นไมโครซอฟต์ซี นั้น ส่วนใหญ่จะเกี่ยวข้องกับฟังก์ชั่นการทำงานกับจอภาพ ซึ่งตัวแปลภาษาแต่ละตัวจะออกแบบการใช้งานและการเชื่อมต่อกับฟังก์ชั่นต่างกัน

สำหรับเทอร์โบซี นั้น ใช้หลักการของตัวขับภาพกราฟฟิก (graphics device driver) ซึ่งจะแยกการทำงานส่วนที่ขึ้นกับฮาร์ดแวร์ออกมา และจะถูกเรียกใช้ในเวลาที่ดำเนินงาน (run time) โปรแกรมเท่านั้น ซึ่งเก็บอยู่ในแฟ้มที่ลงท้ายด้วย .BGI

จากการปรับเปลี่ยนโปรแกรมจากเทอร์โบซี ไปเป็นไมโครซอฟต์ซี พอที่จะสรุปเปรียบเทียบได้ดังนี้

เทอร์โบซี 2.0	ไมโครซอฟต์ซี 7
#include <alloc.h>	#include <malloc.h>
#include <dir.h>	ไม่มี
#include <mem.h>	#include <memory.h>
#include <graphics.h>	#include <graph.h>
detectgraph	_getvideoconfig
getvect	_dos_getvect
outportb	outp
inportb	inp
disable	_disable
enable	_enable
setcolor	_setcolor
closegraph	_setvideomode (_DEFAULTMODE)
bioskey	_bios_keybrd
setvisualpage	_setvisualpage
clearviewport	_clearcreen (_GVIEWPORT)
setactivepage	_setactivepage
setbkcolor	_setbkcolor
rectangle	_rectangle