บทที่ 6

การพัฒนาโปรแกรมส่งผ่านข้อมูลบนเครือข่ายท้องถิ่นแบบโทเกนริงบน LLC

6.1 เป้าหมายในการพัฒนาโปรแกรม

6.1.1 เพื่อแสดงการควบคุมการทำงานของอะแดปเตอร์การ์ดโทเกนริงด้วยซอฟต์แวร์ บน LLC

6.1.2 เพื่อแสดงขบวนการติดต่อกัน ส่งผ่านข้อมูลให้กัน และขบวนการยกเลิกการติดต่อ ระหว่างสถานีใดๆบนเครือข่ายท้องถิ่นแบบโทเกนริงในโพรโตคอลระดับเดียวกัน

6.1.3 ทำการติดต่อกับโพรโตคอล ในระดับ LLC

6.1.4 โปรแกรมสามารถแสดงสถานะต่างๆได้เช่น ข้อผิดพลาดต่างๆ ข้อมูลสถานภาพต่างๆของแผ่นวงจรอะแดปเตอร์ เป็นต้น

6.2 เครื่องมือและอุปกรณ์ประกอบต่างๆที่ใช้ในการพัฒนาโปรแกรม

6.2.1 ใช้ภาษา C ในการพัฒนาโปรแกรม โดยใช้ เทอร์โบซี คอมไพเล่อร์ รุ่นที่2.0 เป็นเครื่องมือ

6.2.2 เครื่องไมโครคอมพิวเตอร์ รุ่น IBM/PC XT คอมแพติเบิล ซึ่งมีความเร็วสัญญาณนาฬิกาขนาด 16 MHz มีหน่วยความจำแรมขนาด 640 กิโลไบต์ จำนวน 1 ตัว

เครื่องไมโครคอมพิวเตอร์ รุ่น IBM/PC AT คอมแพติเบิล ซึ่งมีความเร็วสัญญานนาฬิกา ขนาด 25 Mz มีหน่วยความจำแรมขนาด 1 เม็กกะไบต์ จำนวน 1 ตัว พร้อมฮาร์ดดิสก์ขนาด 80 เม็กกะไบต์

6.2.3 โทเกนริงแลนอินเตอร์เฟสการ์ดรุ่น Token Ring PC Adapter II ของบริษัท IBM จำนวน 2 แผ่น พร้อมสายเคเบิลแบบสายตีเกลียวคู่แบบมีสายนอกหุ้ม จำนวน 2 เส้น โดยที่มีการตั้งค่าต่างๆให้การ์ดทั้ง 2 แผ่นดังนี้ IRQ2 รอมแอดเดรส

CC000 แรมแอดเดรส D8000

6.2.4 อุปกรณ์ตัวรวมสาย MAU รุ่น 8228 ของบริษัท IBM จำนวน 1 ตัว

6.3 วิธีการเขียนโปรแกรมติดต่อกับ LLC โดยใช้ภาษาซี

การติดต่อในระดับ LLC กับแลนอินเตอร์เฟสการ์ดของ IBM จะต้องทำการเรียกโปรแกรมดีไวซ์ไดร์เวอร์ ชื่อ DXMCOMOD.SYS ก่อนโดย การใส่บรรทัดในไฟล์ CONFIG.SYS ดังนี้

DEVICE = DXMCOMOD.SYS

ซึ่งไดร์เวอร์ตัวนี้จะติดต่อกับ โปรแกรมระดับ MAC ที่อยู่ในฮาร์ดแวร์โดยตรง

การเรียกใช้งาน LLC จะต้องต้องค่าใน CCB (Command Control Block) ก่อน ซึ่งจะมีโครงสร้างดังนี้

```
struct command_control_block
{
  char   adapter;
  char   command;
  char   retcode;
  char   work;
  void   *queue;
  void   *post;
  void   *parameter;
};
```

โดยแต่ละฟิลด์มีความหมายดังนี้

adapter : 0x00 ถ้าเป็นอะแดปเตอร์ตัวแรก

0x01 ถ้าเป็นอะแดปเตอร์ตัวที่สองซึ่งปรกติจะใช้เป็นบริดจ์หรือเกตเวย์ของเน็ตเวิร์ค

command : จะใช้เก็บค่ารหัสคำสั่งที่เราต้องการให้ LLC ทำงาน

retcode : ให้ค่าส่งกลับหลังจากทำคำสั่งเสร็จ ซึ่งถ้ากำลังอยู่ในระหว่างทำงานจะมีค่าเป็น 0xFF และให้ค่าเป็น 0x00 แสดงว่าคำสั่งนั้นทำงานได้สมบูรณ์ ถ้าให้ค่าอื่นจะเป็นค่าแสดงความผิดพลาดต่างๆ

work : ใช้สำหรับใช้ภายใน LLC เอง

queue : ในขณะกำลังทำงานอยู่ อะแดปเตอร์การ์ดจะใช้ฟิลด์นี้เป็นการภายใน เมื่อทำคำสั่งเสร็จแล้วฟิลด์นี้จะเก็บค่าพอยต์เตอร์ไปยังคิวของ CCB ซึ่งจะทำอย่างนี้ก็ต่อเมื่อมีการสั่ง แบบ I เฟรม และมีการใช้ post ฟิลด์

post : เก็บค่าตัวชี้แบบไกลไปยังฟังก์ชันการบริการอินเตอร์รัปต์หลังจากที่ทำคำสั่งเสร็จแล้ว

parameter : เก็บค่าตัวชี้แบบระยะไกลไปยังบัฟเฟอร์ ที่เก็บค่าพารามิเตอร์ต่างๆที่จำเป็นต้องใช้ในแต่ละคำสั่ง

การตั้งค่า CCB นี้ก็เป็นหลักการเดียวกับ NCB และ ECB แต่การส่งผ่านพารามิเตอร์ต่างๆนั้นจะแตกต่างออกไป คือ ในแต่ละคำสั่งของ LLC จะมีการใช้ค่าพารามิเตอร์ไม่เหมือนกันและไม่เท่ากัน ดังนั้นจึงต้องใช้วิธีเก็บค่าตัวชี้ไปยังบัฟเฟอร์ที่เก็บตัวแปรแทน โดยบัฟเฟอร์เก็บพารามิเตอร์ของคำสั่งต่างๆจะมีลักษณะเป็นโครงสร้าง เช่น

```
struct dir_initialize_parameters
{
    unsigned int    bring_ups;
    unsigned int    aram_address;
    char    work[4];
    void  (*adapter_error);
    void  (*netw_status_error);
```

```
        void  (*pc_error);
    };
```

เมื่อมีการตั้งค่า CCB และ พารามิเตอร์ต่างๆแล้วการเรียกใช้งานก็สามารถทำโดยเรียกอินเตอร์รัปต์เบอร์ 5Ch เช่นเดียวกับ NETBIOS แต่จะแตกต่างกันที่ ไบต์แรกของ CCB และ NCB ฟังก์ชันที่ใช้เรียกอินเตอร์รัปต์เพื่อใช้งาน LLC เป็นดังนี้

```
extern int net_error;
void int_adapter(struct command_control_block *ccb,int wait)
{
     _ES = FP_SEG(ccb);
     _BX = FP_OFF(ccb);
     geninterrup(0x5c);
     if (wait == WAIT)
     {
        while (ccb.retcode == 0xFF
     }
     net_error = ccb.retcode;
}
```

จะเห็นว่าการเรียกฟังก์ชันนี้เราสามารถส่งผ่านตัวแปรเพื่อบอกว่า เป็นแบบคอย หรือ ไม่คอย ได้ ซึ่งถ้าเป็นแบบคอยมันจะวนลูปตรวจสอบค่า retcode ดูว่าทำงานเสร็จหรือยัง เมื่อเสร็จแล้วก็เก็บค่า retcode ไว้ในตัวแปร net_error ซึ่งเป็นตัวแปรภายนอกใช้ร่วมกัน เพื่อนำไปตรวจสอบความผิดพลาดที่เกิดขึ้นได้

ในการส่งผ่านข้อมูลโดยใช้ LLC นั้นต้องมีการสร้าง SAP แล้วสร้างลิงค์ เพื่อติดต่อกัน การติดต่อมีได้ทั้งแบบ เซสชัน และดาต้าแกรม ตามขั้นตอนที่กล่าวในบทที่แล้ว โดยการส่งข้อมูลโดยวิธีดาต้าแกรมนั้นต้องมีการสร้าง ส่วนหัวของ โทเกนริงเฟรมเองด้วย ซึ่งทำโดยการเรียกฟังก์ชันดังนี้

```
void   build_lan_header(char destination[6],char *buffer)
{
       memset(buffer,0,14)
       /* byte 0  = AS (access control)
          byte 1  = FC (frame control)
          bytes 2-7 = destination address */
       memcpy(&buffer[2], destination, 6);
       /* byte 8-13 = source address */
 }
```

เราเพียงแต่ตั้งค่า แอดเดรสปลายทางเท่านั้น ส่วนค่าอื่นๆเราจัดเนื้อที่ให้เท่านั้นแล้วตัวอะแดปเตอร์การ์ดจะตั้งค่าให้เอง

จุดที่สำคัญในการเรียกใช้งานระดับ LLC ก็คือการส่งผ่านค่าตัวแปรต่างๆไปให้ในการเรียกคำสั่งแต่ละคำสั่ง ซึ่งคำสั่งทั้งหมดมีถึง 30 กว่าคำสั่ง และแต่ละคำสั่งจะมีการส่งค่าตัวแปรแตกต่างกันไป ทั้งแบบซับซ้อนและแบบง่ายๆ และมีหลายคำสั่งจะทำการติดต่อในระดับ MAC โดยตรงเลย (IBM เรียกว่า ไดเร็กอินเตอร์เฟส) เช่น

DIR_OPEN_ADAPTER

DIR_CLOSE_ADAPTER

เป็นต้น การส่งผ่านตัวแปรที่น่าสนใจบางตัว เช่น ตัวแปร access_piority ซึ่งจะเป็นการตั้งลำดับความสำคัญในการส่งเฟรมออกไปบน โทเกนริง ตัวแปรนี้จะส่งผ่านไปให้คำสั่ง DLC_OPEN_STAION ตอนที่เปิดเซสชันการติดต่อ ซึ่งตัวแปรนี้จะมีรูปแบบดังนี้ B'nnn00000' nnn คือค่าลำดับความสำคัญ ซึ่งตั้งได้ ตั้ง 0-3 ในสถานะที่เป็นเฟรมปรกติ ซึ่งปรกติจะเป็น 0

6.4 ขั้นตอนการติดต่อและส่งผ่านข้อมูล

ขบวนการติดต่อส่งผ่านข้อมูลในระดับ LLC นั้นมีขั้นตอนดังนี้

1. ทำขบวนการเริ่มต้นทั้งตัวรับและตัวส่ง โดยทำคำสั่งดังนี้

DIR_INTERRUPT จะเป็นการเรียกอินเตอร์รัปต์การ์ดเพื่อทดสอบว่ามีการติดตั้งการ์ดอยู่หรือเปล่า

DIR_INITIALIZE จะเป็นการรีเซตค่าเริ่มต้นให้กับตารางและบัฟเฟอร์ต่างๆที่เกี่ยวข้อง และทำขบวนการทดสอบระบบที่เรียกว่า 'Bring-up test'

DIR_OPEN_ADAPTER จะทำการเปิดการใช้งานอะแดปเตอร์ให้ติดต่อกับเครือข่ายท้องถิ่น และทำการตั้งค่าเริ่มต้นให้ตารางและบัฟเฟอร์ต่างๆใหม่

2. ทำการเปิดจุดแอคเซส หรือ SAP ทั้งตัวรับและตัวส่ง โดยคำสั่ง OPEN_SAP SAP นี้จะเป็นจุดทำให้การบริการแก่โพรโตคอลที่อยู่ระดับบนขึ้นไป และในการเปิด SAP จะต้องมีการเตรียมเนื้อที่เพื่อเป็นบัฟเฟอร์พูล (Buffer pool) ขนาด 4096 ไบต์ สำหรับใช้ในการรับส่งผ่าน SAP นั้นด้วย นอกจากนี้การเปิด SAP จะต้องมีการตั้งค่าหมายเลขให้ SAP นั้นด้วย ซึ่งหมายเลข SAP นี้ปรกติจะเป็นตัวบอกว่า SAP นั้นให้บริการโพรโตคอลอะไรในชั้นสูงขึ้นไป เช่น หมายเลข F0 จะให้บริการ NETBIOS เป็นต้น ซึ่งรายละเอียดได้กล่าวในบทที่ 5 แล้ว และเราสามารถเปิด SAP ได้พร้อมๆกันถึง 255 SAP หลังจากที่เราเปิด SAP แล้ว ค่าที่ส่งกลับมาจะเป็นค่า หมายเลขรหัสสถานี (Station_ID) ซึ่งจะใช้สำหรับอ้างอิงจุด SAP ที่จะส่งแพกเก็ตออกไป ส่วนปลายทางจะใช้อ้างอิงโดยหมายเลข SAP ปลายทางแทน

3. ตัวรับสั่งคำสั่ง RECEIVE วนรอรับ โดยที่จะสามารถรับได้ทั้งการส่งแบบดาต้าแกรม (UI เฟรม) และ แบบ เวอร์ชวลเซอร์กิต (I เฟรม) ในการรับจะต้องระบุหมายเลขรหัสสถานีปลายทางที่จะรับข้อมูลด้วย และเมื่อมีเฟรมข้อมูลส่งมาถึงก็จะมีการนำค่าจากบัฟเฟอร์รับมาแสดงผล

4. ตัวส่งที่จะส่งถ้าส่งแบบไม่มีหมายเลข (Unnumber Information) หรือที่เรียกว่าดาต้าแกรม จะต้องทำการสร้าง ส่วนหัวของเฟรมข้อมูลเอง โดยต้องจองที่ไว้มีขนาดเท่ากับส่วนหัวนั้น และ ต้องมีการให้ค่าแอดเดรสปลายทางซึ่งเป็นค่าที่เก็บไว้ในรอมของแผ่นวงจรอะแดปเตอร์ เป็นการถาวร ในการที่เราจะรู้ค่าแอดเดรสนี้

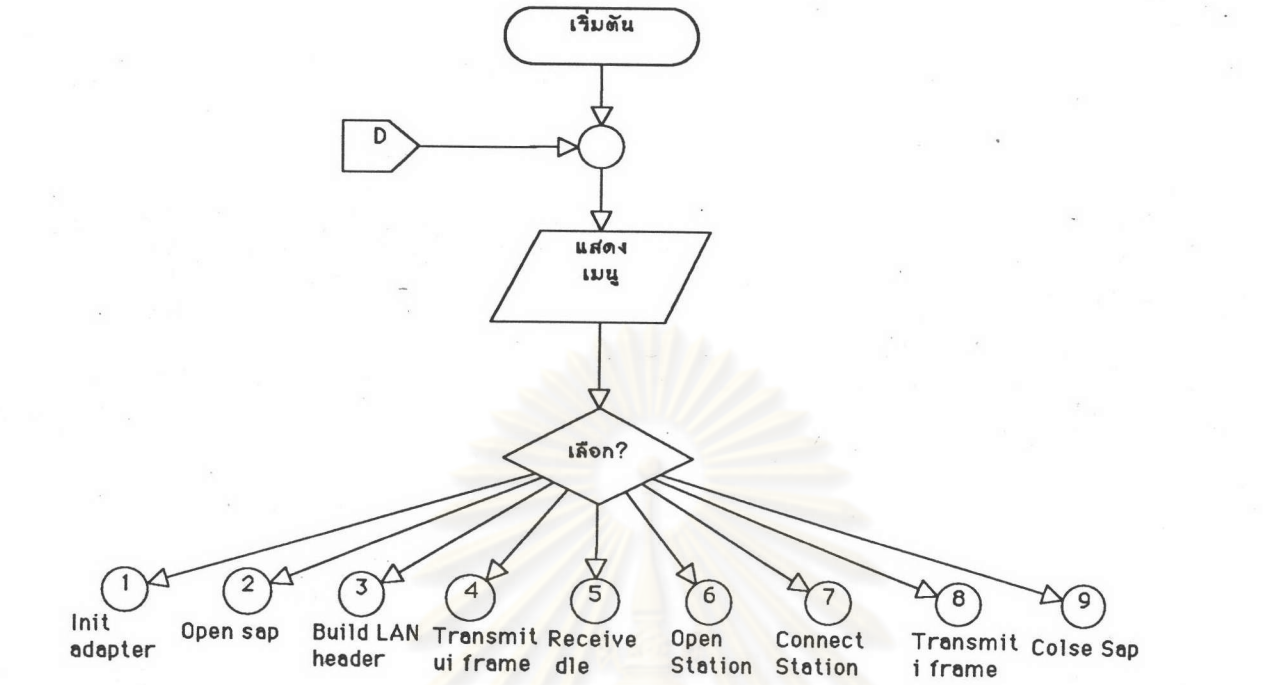
สามารถเรียกดูได้จาก คำสั่งเรียกดูสถานะของแผ่นวงจรเครือข่ายท้องถิ่นนั้นๆ นอกจากนี้เราต้องให้ค่า หมายเลขรหัสสถานีของสถานีส่ง ค่า SAP ปลายทาง และตัวข้อมูลที่จะส่งตามมา ซึ่งทั้งหมดนี้ทำโดยคำสั่ง TRANSMIT_I_FRAME

5. ในการที่จะส่งข้อมูลแบบมีหมายเลขกำกับ (Numbered Information) หรือที่เรียกว่า เวอร์ชวลเซอร์กิต จะต้องมีการสร้างลิงค์ขึ้นก่อนและเมื่อสร้างได้แล้ว จะให้สถานีที่ต่อกันอยู่นั้นส่งหรือรับก็ได้ ในการสร้างลิงค์นั้น ต้องมีขบวนการ OPEN_STATION และ CONNECT_STATION ก่อนโดยที่ในการ OPEN_STATION นั้น ต้องส่งผ่านค่าหมายเลขรหัสสถานีต้นทาง ค่าหมายเลข SAP ปลายทาง และ ค่าแอดเดรสปลายทางให้ ซึ่งการ OPEN_STATION นี้จะเป็นการสร้างลิงค์เชื่อมต่อระหว่าง SAP ของสองสถานี และเมื่อสร้างได้สำเร็จแล้วมันจะส่งค่า หมายเลขรหัสสถานีเชื่อมต่อ (Link Station ID) กลับมาให้ ซึ่งค่านี้จะใช้อ้างอิงในการใช้ลิงค์นั้นๆ เมื่อเราสร้างการเชื่อมต่อได้แล้ว เมื่อต้องการจะใช้ก็ต้องมีการเรียกติดต่อกับลิงค์นั้นโดยคำสั่ง CONNECT_STATION ซึ่งจะเป็นการเชื่อมต่อกันระหว่างโปรแกรมประยุกต์ของสถานีต้นทาง กับโปรแกรมประยุกต์ของสถานีปลายทาง ในการเรียกใช้คำสั่งนี้เราต้องให้ค่าหมายเลขรหัสสถานีเชื่อมต่อที่ได้มาจากการเรียก OPEN_SATION ด้วย เมื่อเราสร้างขบวนการทั้งหมดนี้เรียบร้อยแล้วก็จะสามารถส่งข้อมูลไปยังสถานีปลายทางได้โดยใช้คำสั่ง TRANSMIT_I_FRAME โดยที่เราจะส่งหรือรับผ่านทางลิงค์ที่สร้างขึ้นกี่ครั้งก็ได้ จนกระทั่งเราจะปิดลิงค์นั้น

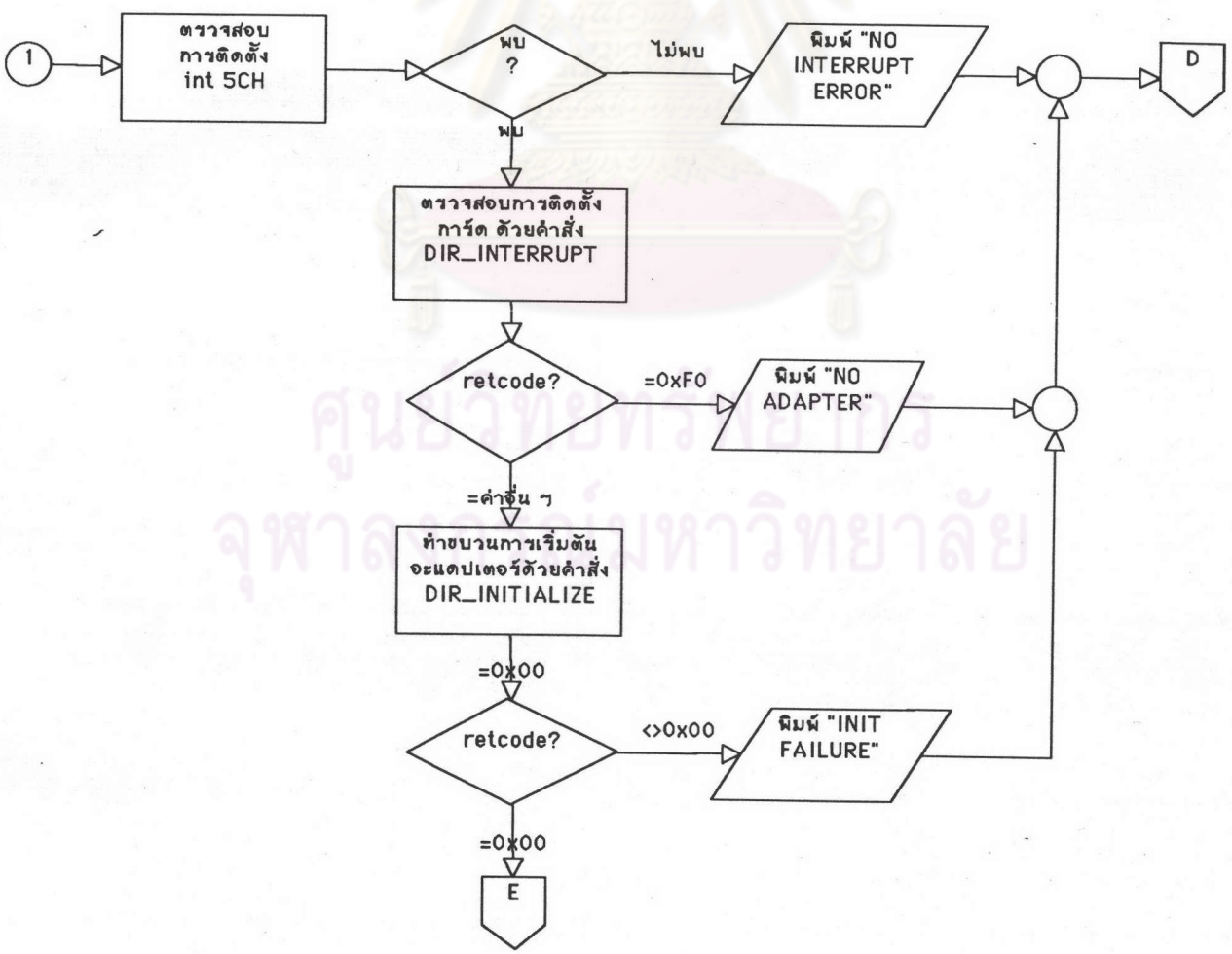
6. เมื่อเราจะเลิกส่งผ่านข้อมูลเราต้องทำการปิด SAP โดยคำสั่ง CLOSE_SAP ซึ่งเราจะบอกว่าจะปิด SAP ไหนโดยการให้ค่า หมายเลขรหัสสถานีที่ต้องการ และถ้า SAP นั้นมีลิงค์อยู่ ก็จะทำให้ลิงค์นั้นถูกปิดไปด้วย

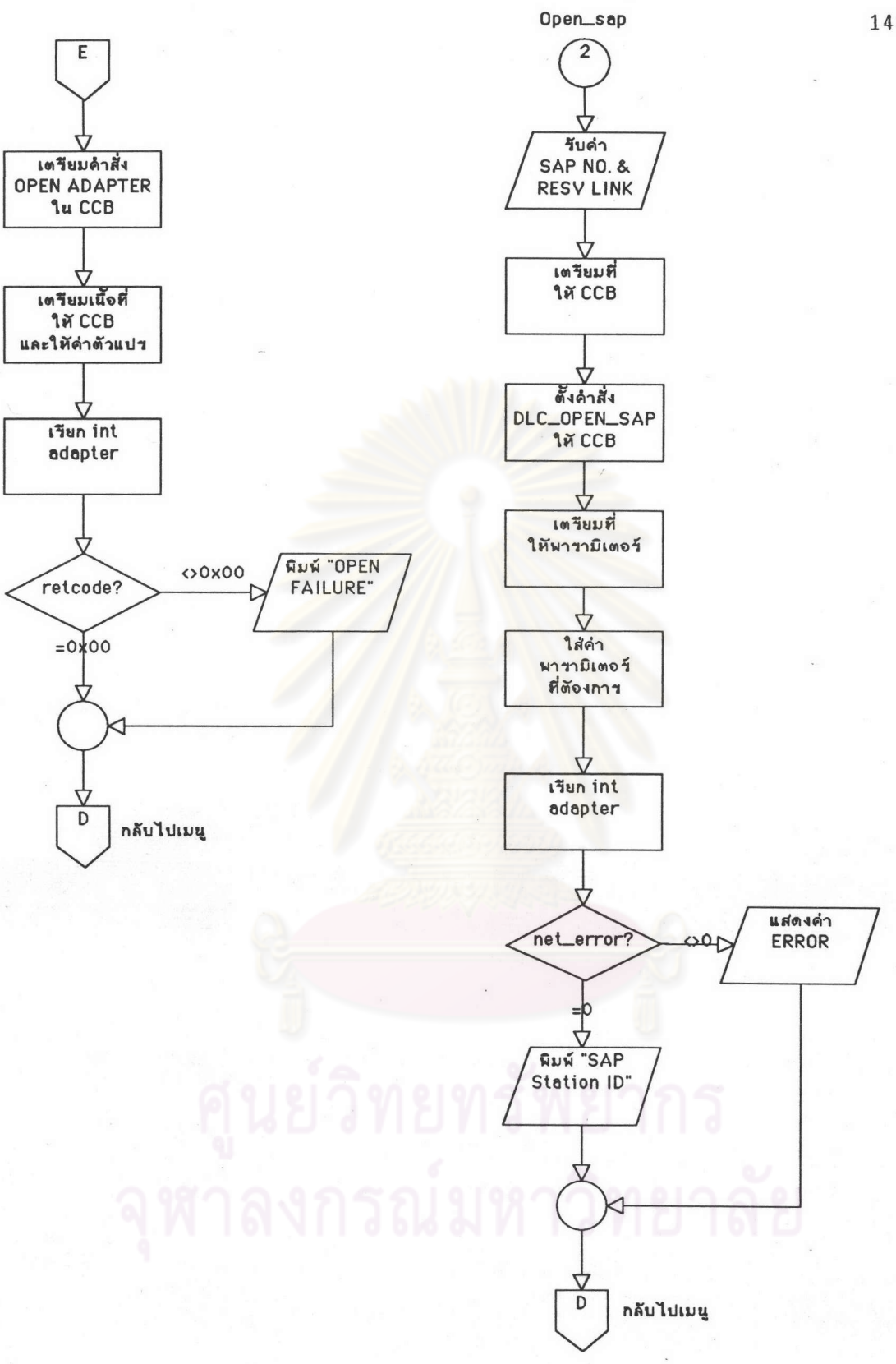
6.5 ผังงานแสดงโครงสร้างโปรแกรม LLC

โปรแกรม LLC

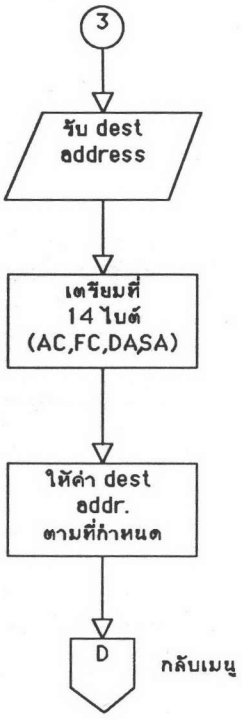


Init adapter

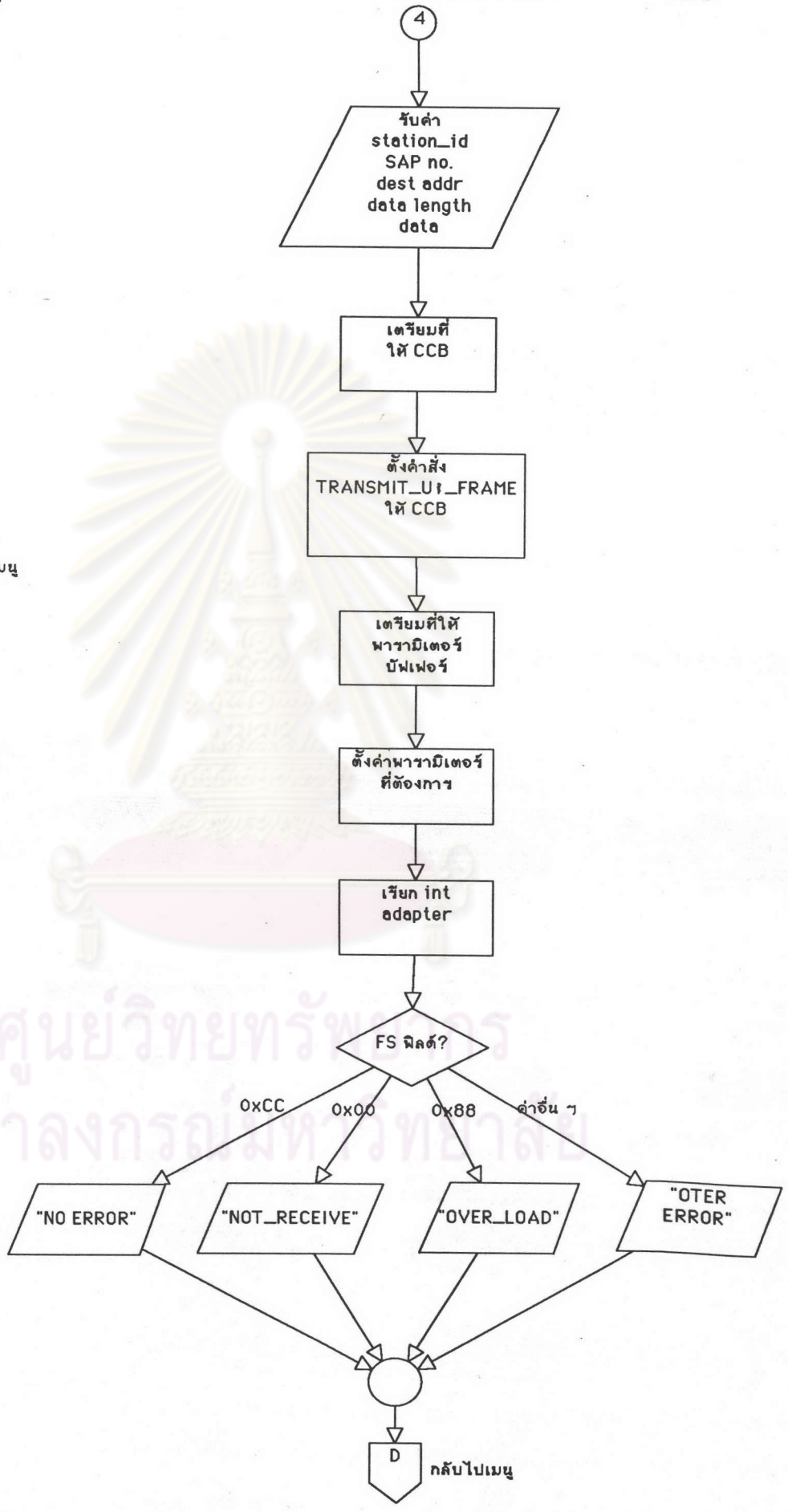


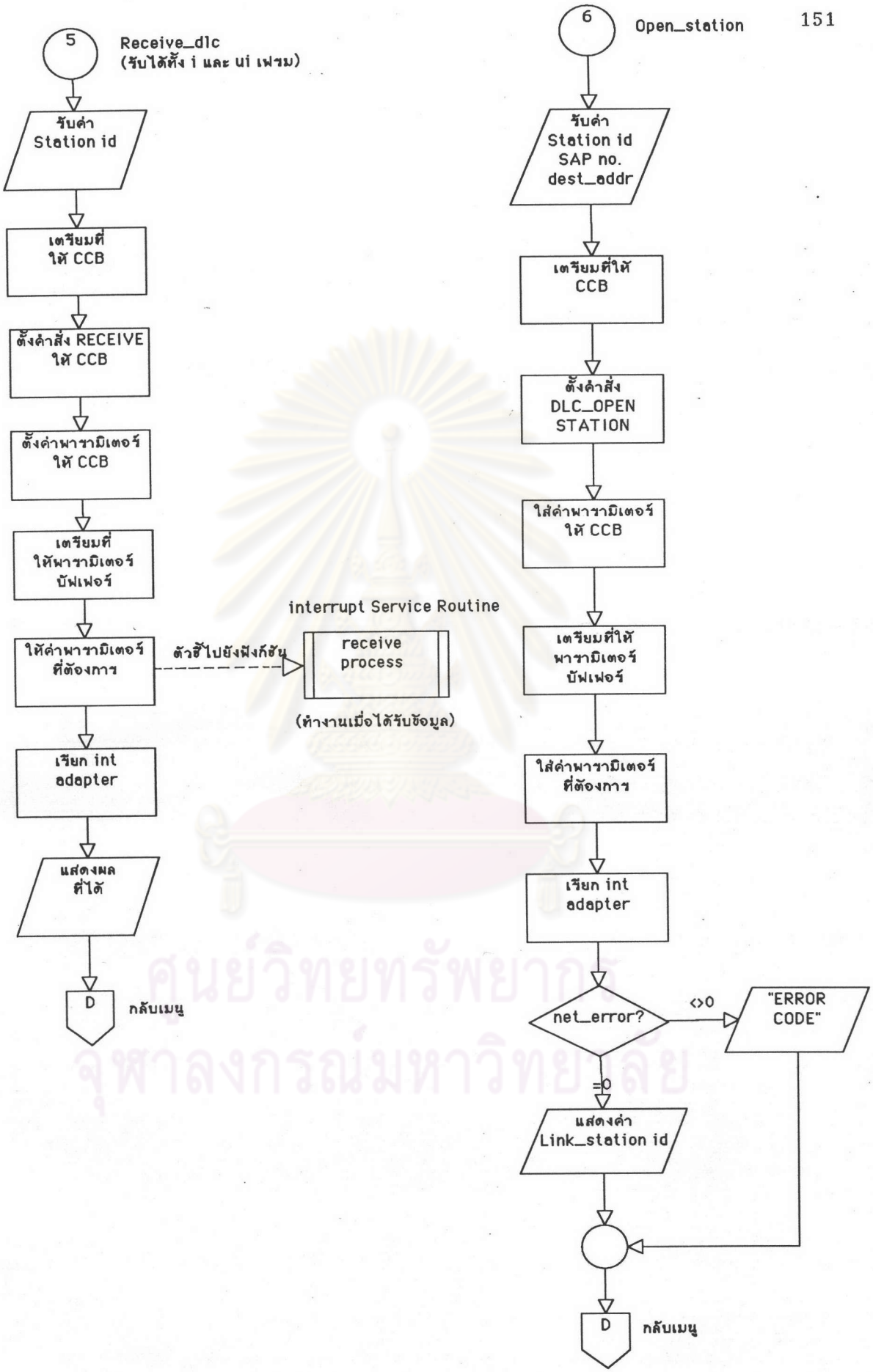


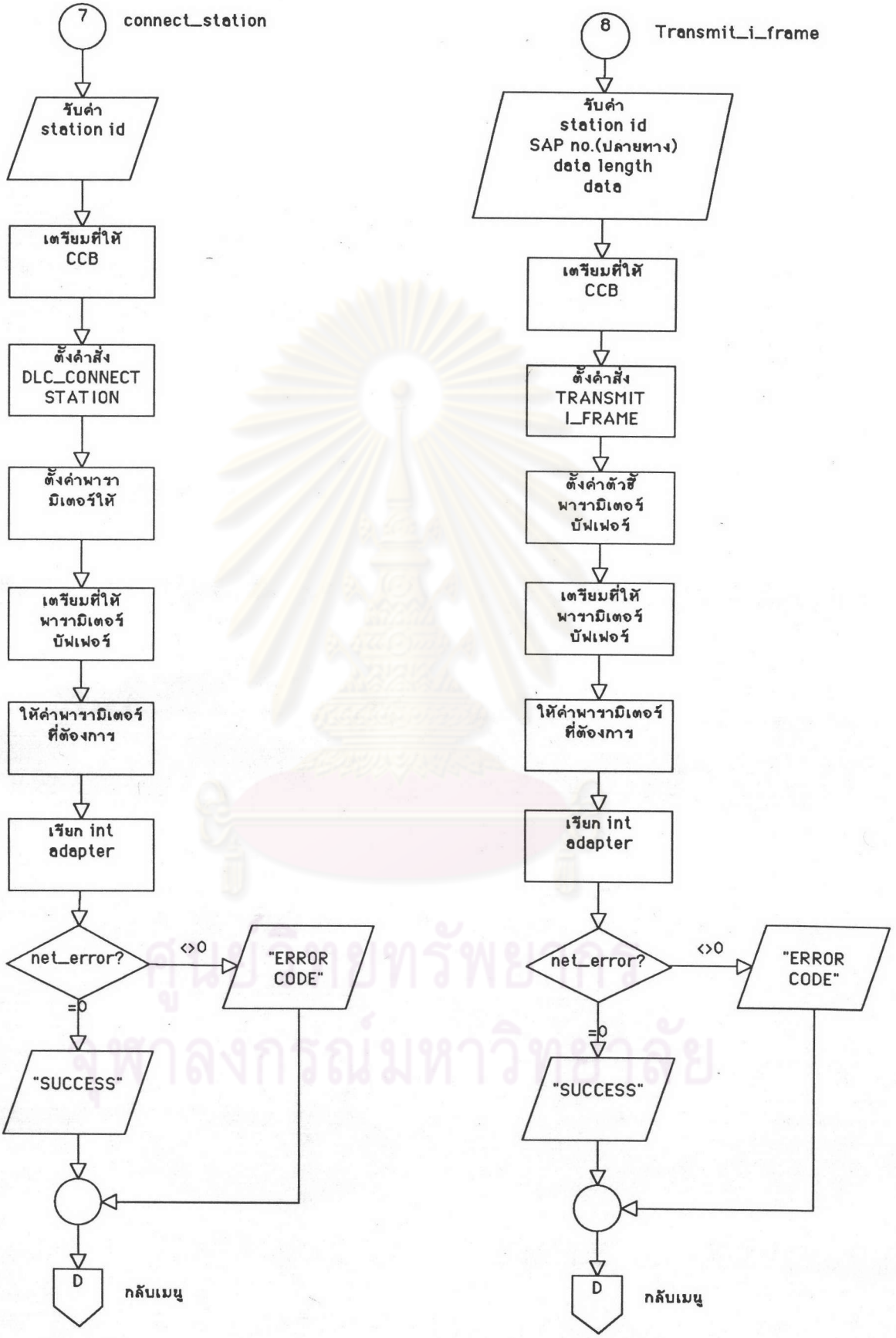
Build_lan_header



Transmit_ui_frame







Close_Sap