รายการอ้างอิง

ภาษาไทย

ถวัลย์ ตั้งสิตานนท์. เปิดโฉม VGA ฮาร์ดแวร์. วารสารไมโครคอมพิวเตอร์ ฉบับที่ 55 (มกราคม-กุมภาพันธ์ 2533): 284-294.

ทวีศักดิ์ กออนันตกูล. จอคอมพิวเตอร์ไทยแบบต่าง ๆ. วารสารไมโครคอมพิวเตอร์ ฉบับที่ 41 (สิงหาคม 2531): 145-158.

โทรศัพท์แห่งประเทศไทย, องค์การ. กองบริการสอบถามและโทรศัพท์ทางไกล. ระบบ CDAS. กรุงเทพมหานคร: กองบริการสอบถามและโทรศัพท์ทางไกล องค์การโทรศัพท์แห่งประเทศไทย, 2534. (อัดสำเนา)

ยืน ภู่วรวรรณ. เทคโนโลยี ฮาร์ดแวร์ IBM PC. กรุงเทพมหานคร: หจก.เอช-เอน การพิมพ์, 2533.

ภาษาอังกฤษ

Black, U.D. Data Networks Concepts, Theory, and Practice. USA: Prentice-Hall., 1989.

Lafore, R. The Waite Group's C Programming Using Turbo C++. Indiana: SAMS., 1990.

ภาคผนวก

ภาคผนวก ก

การใช้งานโปรแกรมเลียนแบบเทอร์มินัลทีอีที

โปรแกรมเลียนแบบเทอร์มินัลทีอีที ประกอบด้วยแฟ้มข้อมูลต่าง ๆ ดังต่อไปนี้

1. CDAS.EXE โปรแกรมเลียนแบบเทอร์มินัลทีอีที
2. NORMAL.FON แฟ้มข้อมูลที่จัดเก็บรูปแบบอักขระ
3. EGAVGA.BGI ตัวขับกราฟิกของวงจรแสดงผลอีจีเอและวีจีเอ
4. READ.ME แฟ้มข้อมูลที่จัดเก็บวิธีการใช้งานเทอร์มินัลทีอีที

การออกแบบโปรแกรมเลียนแบบเทอร์มินัลทีอีที มีความมุ่งหมายให้ผู้ใช้มีความสะดวกในการใช้งานโปรแกรมโดยเมื่อผู้ใช้เรียกโปรแกรม CDAS.EXE แล้ว ผู้ใช้ต้องกำหนดค่าเริ่มต้นให้โปรแกรม ดังนี้

1. กำหนดวงจรสื่อสารแบบอนุกรม (COM PORT) ที่ต้องการใช้งาน โดยทั่วไปเครื่องไมโครคอมพิวเตอร์จะมีวงจรสื่อสารแบบอนุกรมให้เลือกใช้งานได้ 2 วงจรคือ COM1และ COM2

2. กำหนดชนิดของเครื่องไมโครคอมพิวเตอร์ที่นำมาใช้งาน ซึ่งได้แก่เครื่องไมโครคอมพิวเตอร์ที่มีหน่วยประมวลผลกลาง 80286, 80386 และ 80486

เมื่อผู้ใช้กำหนดค่าเริ่มต้นให้โปรแกรมแล้ว ผู้ใช้สามารถทำการต่อเชื่อมกับระบบซีดาสได้โดยตรง หากผู้ใช้ต้องการต่อเชื่อมกับระบบซีดาสผ่านอุปกรณ์สื่อสารข้อมูลโมเดม (MODEM)ผู้ใช้จะต้องกำหนดค่าตัวแปรเริ่มต้นให้กับโมเดมด้วยซึ่งได้แก่

1. กำหนดขนาดของข้อมูลเป็น 8 บิต
2. กำหนดบิตสิ้นสุดข้อมูลเป็น 1 บิต
3. กำหนดบิตตรวจสอบเป็น 0 (NONE)
4. กำหนดโหมดในการรับส่งข้อมูลเป็นอสมวาร
5. กำหนดความเร็วในการรับส่งข้อมูลตามความเร็วของโมเดมที่ใช้แต่ไม่เกิน 9600 บิตต่อวินาที

เมื่อโปรแกรมเลียนแบบเทอร์มินัลที่อีทีสามารถติดต่อกับระบบซีดาสได้แล้ว ผู้ใช้ต้องกดแป้น Home เพื่อเป็นการเข้าใช้งาน (SIGN ON) โดยระบบซีดาสจะแสดงผลดังนี้

```
TERMINAL (DEVICE NO.)


USER : _
PASSWORD :
```

ผู้ใช้จะต้องพิมพ์รหัสผู้ใช้ในเขตข้อมูลผู้ใช้ (User) แล้วกดแป้น Enter แล้วผู้ใช้ต้องพิมพ์รหัสผ่านในเขตข้อมูลรหัสผ่าน (PASSWORD) หลังจากนั้นจอแสดงผลจะแสดงผลดังนี้

```
FACILITY :
ENQUIRY
TEL NO. RETRIEVAL
XCHANGE TELEPHONE


SELECT BY ENTERING FIRST LETTER
ACTION : _
```

ผู้ใช้สามารถเลือกหน้าที่การทำงานต่าง ๆ ของระบบซีดาสได้โดยการเลือกอักษรตัวแรก เช่น "E" หรือ "T" หรือ "X" แล้วกดแป้น Enter

ถ้าผู้ใช้ต้องการค้นหาข้อมูลผู้ใช้โทรศัพท์จากชื่อ ให้เลือก "E" (ENQUIRY) จอแสดงผลจะแสดงข้อความ

```
NAME _                          ADDR                          DIR
```

ถ้าผู้ใช้ต้องการค้นหาข้อมูลจากหมายเลขโทรศัพท์ให้เลือก "T"(TEL NO. RETRIEVAL) จอแสดงผลจะแสดงข้อความ

```
                                AREA CODE _      TEL
```

ถ้าผู้ใช้ต้องการค้นหาข้อมูลหมายเลขโทรศัพท์เปลี่ยนแปลงให้เลือก "X" (XCHANGE TELEPHONE) จอแสดงผลจะแสดงข้อความ

```
            CHANGEOVER TELEPHONE LIST


            AREA CODE : _
            TEL NO. :


            NEW TEL NO. :    DATE CHANGE :
```

เมื่อผู้ใช้ต้องการเลิกการใช้งานจากระบบชีดาส ผู้ใช้จะต้องทำการออกจากระบบชีดาส (SIGN OFF) โดยกดแป้นฟังก์ชั่น F10 จอแสดงผลจะแสดงข้อความของเมนูหลัก

```
FACILITY :
ENQUIRY
TEL NO. RETRIEVAL
XCHANGE TELEPHONE


SELECT BY ENTERING FIRST LETTER
ACTION : _
```

และผู้ใช้จะต้องกดแป้น Enter จอแสดงผลจะแสดงข้อความ "GOOD BYE" ซึ่งเป็นการแสดงว่าผู้ใช้ได้ออกจากระบบซีดาสแล้ว

```
FACILITY :
ENQUIRY
TEL NO. RETRIEVAL
XCHANGE TELEPHONE

SELECT BY ENTERING FIRST LETTER
ACTION :
GOOD BYE
```

เมื่อผู้ใช้ต้องการพิมพ์ข้อมูลที่ปรากฏบนจอแสดงผลทางเครื่องพิมพ์ให้กดแป้น Alt และ P พร้อมกัน จอแสดงผลจะแสดงข้อความเพื่อเตือนให้ผู้ใช้รอจนกว่าการพิมพ์จะสิ้นสุด เมื่อการพิมพ์สิ้นสุดแล้วผู้ใช้สามารถใช้งานต่อไปได้

```
** WARNING **
PLEASE WAIT UNTIL FINISH
```

การเปลี่ยนโหมดการพิมพ์ระหว่างภาษาไทยและภาษาอังกฤษให้กดแป้น Alt และ Tพร้อมกัน โดยมุมซ้ายด้านบนของจอแสดงผลจะแสดงโหมดการพิมพ์ในขณะนั้นให้ทราบ

Thai Mode	F1-Name F3-Addr/Area F5-Dir/Tel Alt+T-Thai/Eng

ในขณะใดขณะหนึ่งผู้ใช้สามารถเรียกใช้ระบบช่วยเหลือ (Help) โดยกดแป้น Ctrl และ F1 พร้อมกัน ระบบช่วยเหลือจะแสดงหน้าที่ของแป้นฟังก์ชั่นทั้งหลายทำให้สะดวกในการใช้งาน

Help

Home	Press 'Home' key to sign on.
F1	Move cursor to the beginning of the 'NAME' field.
F2	More (PgDn).
F3	Move cursor to the beginning of the 'ADDR' field or 'AREA CODE' field.
F4	Back (PgUp).
F5	Move cursor to the beginning of the ''DIR' field or 'TEL' field. 'DIR' field consists of 1 or 3 letters. น - Business and residential file ร - Government file ท - Frequently Called Number (FCN) file
F7	Select the number of main record and then press F7 for more detail.
F8	Select the alternative spelling and then press F8
F9	Refreshes the screen with the empty format.
F10	Press F10 to main menu.
Enter	Indicates the input is complete and send the information to host computer.
Alt-P	Dump screen to printer.

Esc-Exit Help

สุดท้ายเมื่อผู้ใช้ต้องการออกจากโปรแกรมเลียนแบบเทอร์มินัลทีอีทีให้กดแป้น Alt และ X พร้อมกัน จอแสดงผลจะแสดงข้อความ

```
HAVE YOU ALREADY SIGNED OFF? (Y/N)
```

เพื่อเป็นการเตือนว่าได้ออกจากระบบซีดาสแล้วหรือไม่ ซึ่งผู้ใช้จะต้องทำการออกจากระบบซีดาสทุกครั้งก่อนออกจากโปรแกรมเลียนแบบเทอร์มินัลทีอีที

ภาคผนวก ข

รายละเอียดฮาร์ดแวร์ของเทอร์มินัลทีอีที

Hardware Description

Controller Board

- 1 serial I/O programmable port (ASYNC)
- 64K bytes RAM (32K for data buffer)
- 16K bytes by 12 bits (data 8 bits, attribute 4 bits) screen buffer
- 32K bytes EPROM program control
- Z80-A 4 MHz Microprocessor control
- Programmable real time clock (for BAUD rate generator)

Video Controller Board

Standard : TTL level monochrome display (9 pin D-type connector)
inverse video, highlight

Option : Composite monochrome (R.C.A. connector)
8 color R.G.B., color graphics bits image
600 x 200 pixels

Keyboard

Microprocessor electronic, capacitance switch

Monitor

- 12 inches diagonal, 90 degree deflection angle
- P39 phosphor, high resolution
- 20 MHz bandwidth, anti-glare screen
- TTL level (positive) data and horizontal/vertical drive
- Horizontal frequency 18.432 KHz

Display Format

6.5 x 8.9 inch viewing area

7 x 7 dots character matrix

80 columns by 24 rows (4 levels per row)

7680 character positions standard

Option :

80 columns by 36 rows (4 levels per row)

11520 character positions option in Thai and English characters

Normal, inverse or highlight video

Communication Interface

RS-232C, 8 data bits, 1 stop bit, parity none

Baud Rate 110, 300, 600, 1200, 2400, 4800, 9600

Cursor Control

Cursor up, down, left, right, addressable cursor,

line feed, carriage return

Character Set

96 ASCII code

67 Thai characters and 10 Thai numeric extended

Keyboard

Detachable up to 1 metre from screen, low profile

83 keys typewriter key pad

11 keys numeric key pad

10 CDAS function keys

Switch selectable Thai or English key set on a sigle key press

Physical Characteristics

Processor

(W x D x H)	17.7 x 13.8 x 3.2 inches
Weight	7.8 pounds

Monitor

(W x D x H)	14.2 x 13.6 x 11.6 inches
Weight	19.3 pounds

Keyboard

(W x D x H)	17.7 x 7.7 x 1.5 inches
Weight	3.4 pounds

Power Requirements

115V or 220V	50 or 60 Hz

ภาคผนวก ค

ชุดคำสั่งภาษาต้นฉบับ (Source Program)

```
#include <graphics.h>
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>
#include <ctype.h>
#include <bios.h>
#include <dos.h>
#include <alloc.h>
#define   ROWS        256
#define   COLS         20
#define   COLOR        14          /*...BLUE */
#define   X             0
#define   Y            20
#define   INTR1      0x0C          /*...Interrupt UART COM1 */
#define   INTR2      0x0B          /*...Interrupt UART COM2 */
#define   INBUFSIZE  4096          /*...Buffer of received character */
#define   KBUFSIZE     16          /*...Keyboard buffer */
#define   lo(f) ((f) & 0xFF)
#define   hi(f) (lo(f >> 8))
typedef struct cbuf
{
```

```
int cb_id;
char *cb_begin;
char *cb_end;
char *cb_front;
char *cb_rear;
}  CBUF;
#define CBNULL  (CBUF *)0
static CBUF *inbuf;
static CBUF *kbuf;
unsigned char font[ROWS][COLS];
int  InputSt[2048];
unsigned x,y,maxx,maxy;
unsigned size;
unsigned size1,size2,size3;
unsigned int ch1,ch2,ch3;
int  z,banner,rch,keyed;
unsigned int Tmode,flag;
unsigned char f,code;
unsigned int RBR,THR,IER,LCR,MCR,LSR,INTR;
int    help_active=0;
float CPU;
FILE *fptr;
void *pagebuff;
char far *pagebuf1, *pagebuf2, *pagebuf3;
char far *pagebuf4, *pagebuf5, *pagebuf6;
int  getcbuf(CBUF *cbuf);
```

```
void dial_MODEM(void);
void Scroll(void);
void print_r3(unsigned char);
void screenD(void);
void inout(void);        /*...Interactive action of CDAS terminal */
void sure(int);
void help1(void);
void help(void);         /*...Display the banner */
void draw_LOGO(int,int,int);   /*...Draw TOT logo */
void Select_COM(void);         /*...Select communication port */
void display(int);
void show(int);
void displayCOM(void);
void displayCOM1(void);
void displayCOM2(void);
void displayCPU(void);
void displayCPU1(void);
void displayCPU2(void);
void displayCPU3(void);
char getcode(void);
void action(int);
void action1(int);
void opengraph(void);      /*...Open graphics system */
void draw_letter(unsigned char,unsigned,unsigned);
void savefont(void);       /*...Save font from NORMAL.FON */
void PageSignOn(void);     /*...Display page sign on */
```

```
void print_r1(unsigned char);          /*...Print font on screen */
void cursor(unsigned,unsigned);        /*...Display cursor */
void print_r2(unsigned char);          /*...Print font on screen */
int  keyin(int,char);                  /*...Type character */
int  Scdas(int,int);              /*...Convert code before sending */
unsigned char Rcdas(unsigned char);
void Screen(unsigned int);             /*...Display screen */
void Screen1(unsigned int);            /*...Display screen */
void clscr(int,int,int,int);           /*...Clear screen */
void beep(void);                       /*...Beep */
void shutdown(void);                   /*...Return old int to CPU */
void init_port(void);             /*...Initial communication port */
void Chk_TXempty(void);                /*...Check Transmit empty */
void Printer(void);                    /*...Dump screen to printer */
void interrupt (*oldhandler)(void); /*...Interrupt service routine */
void interrupt handler(void)
/*...Receiving a character from the port and put into the buffer */
{
unsigned char in_ch;
in_ch = inport(RBR);
putcbuf(in_ch,inbuf);
outportb(0x20,0x20);
}
static int cbufid = 0;
CBUF *newcbuf(size)
/*...Creating a circular queue buffer */
```

```
unsigned size;
{
CBUF *buf;
if((buf = (CBUF *)calloc((unsigned)1, sizeof(CBUF))) != CBNULL)
{
  char *p;
  if((p = (char *)malloc(size + 1)) != NULL)
  {
     buf->cb_id = cbufid++;
     buf->cb_begin = p;
     buf->cb_end = p + size;
     buf->cb_front = buf->cb_rear = p;
  }
  else      /*...can't allocate buffer */
  {
     free((char *)buf);
     buf = CBNULL;
  }
}
return(buf);
}

int putcbuf(c, cbuf)
/*...Putting a character into the buffer */
unsigned char c;
CBUF *cbuf;
```

```
{
register char *p;
if((p = cbuf->cb_rear + 1) > cbuf->cb_end)
  p = cbuf->cb_begin;
if(p != cbuf->cb_front)
{
 *p = c;
 cbuf->cb_rear = p;
 return((int)c);
}
else          /*...buffer full */
{
 return(-1);
}
}

int getcbuf(cbuf)
/*...Getting a character from the buffer */
CBUF *cbuf;
{
if(cbuf->cb_front != cbuf->cb_rear)
{
if(++(cbuf->cb_front) > cbuf->cb_end)
      cbuf->cb_front = cbuf->cb_begin;
return(*(cbuf->cb_front));
}
```

```
else            /*...buffer empty */
{
 return(-1);
}
}

int main(void)
/*...Main routine of CDAS emulation program */
{
ch1 = 0,ch2 = 0,ch3 = 0;
keyed = 0;
flag = 0;
z = 0;
Tmode = 1;
x = X;
y = Y;
savefont();
opengraph();
Select_COM();
inbuf = newcbuf(INBUFSIZE);
kbuf = newcbuf(KBUFSIZE);
oldhandler = getvect(INTR);
setvect(INTR, handler);
init_port();
dial_MODEM();
PageSignOn();
```

```
inout();
shutdown();
closegraph();
printf("    Thanks for using CDAS ");
delay(1000);
return (0);
}

void inout(void)
/*...Receive input from keyboard and outport to host */
/*...Receive input from port and display on the screen */
{
int sch,key,kchl,kchh,Tx;
int delstart;
char signoff;
do
{
     cursor(x,y);
     if(bioskey(1))
     {
     key = bioskey(0);
     putcbuf(lo(key),kbuf);
     putcbuf(hi(key),kbuf);
     }
     if((sch = getcbuf(kbuf)) != -1)
     {
```

```
    if((kchl = sch) != 0x00)
    {
kchh = getcbuf(kbuf);
if(kchl == 0x0D)            /*...ENTER */
{
   outportb(THR, 0x9E);
   Chk_TXempty();
   outportb(THR, 0xF9);
}
else
{
f = keyin(kchl,Tmode);
if(f == 0) continue;else
keyed = 1;
if((f > 64) && (f < 123))
{
    f = toupper(f);
}
code = Scdas(f,Tmode);
Chk_TXempty();
outportb(THR,code);
code = code & 0x00FF;
}
    }
    else
    {
```

```
kchh = getcbuf(kbuf);
switch(kchh)
{
   case 0x3B:                /*...F1 */
outportb(THR,0x9E);
Chk_TXempty();
outportb(THR,0xF1);
break;
   case 0x3C:                /*...F2 */
outportb(THR,0x9E);
Chk_TXempty();
outportb(THR,0xF8);
break;
   case 81:                  /*...PgDn */
outportb(THR,0x9E);
Chk_TXempty();
outportb(THR,0xF8);
break;
   case 0x3D:                /*...F3 */
outportb(THR,0x9E);
Chk_TXempty();
outportb(THR,0xF2);
break;
   case 0x3E:                /*...F4 */
outportb(THR,0x9E);
Chk_TXempty();
```

```
outportb(THR,0xF7);
break;
   case 73:                /*...PgUp */
outportb(THR,0x9E);
Chk_TXempty();
outportb(THR,0xF7);
break;
   case 0x3F:              /*...F5 */
outportb(THR,0x9E);
Chk_TXempty();
outportb(THR,0xF3);
break;
   case 0x40:              /*...F6 */
/* outportb(THR,0x9E);
Chk_TXempty();
outportb(THR,0xBA);   */
break;
   case 0x41:              /*...F7 */
outportb(THR,0x9E);
Chk_TXempty();
outportb(THR,0xF6);
break;
   case 0x42:              /*...F8 */
outportb(THR,0x9E);
Chk_TXempty();
outportb(THR,0xF5);
```

```
break;
   case 0x43:                /*...F9 */
outportb(THR,0x9E);
Chk_TXempty();
outportb(THR,0xFB);
break;
   case 0x44:                /*...F10 */
outportb(THR,0x9E);
Chk_TXempty();
outportb(THR,0xBA);
break;
   case 0x63:                /*...Ctrl-F6 */
outportb(THR,0x9E);
Chk_TXempty();
outportb(THR,0xBA);
break;
   case 20:                  /*...Alt-T */
Tmode = !Tmode;
Screen(Tmode);
break;
   case 0x47:                /*...Home */
outportb(THR,0x00);
break;
   case 25:                  /*...Alt-P */
sure(2);
Printer();
```

```
 break;
    case 45:                    /*...Alt-X */
 sure(1);
 signoff = getch();
 if(toupper(signoff) == 'Y')
 {
   flag = 1;
   break;
 }
 else
 {
   putimage(170,175, pagebuff, COPY_PUT);
   free((void *)pagebuff);
   break;
 }
    case 94:                    /*...Ctrl+F1 */
 {
   help1();
   break;
 }
    default:
 break;
 }
     }
}
while((rch = getcbuf(inbuf)) != -1)
```

```
{
   delay(CPU);
   ch1 = rch & 0x00FF;
   if((ch1 < 32) || ((ch1 > 168) && (ch1 < 173)))
   {
     switch(ch1)
     {
     case 0x0007:    /*...BEEP */
     {
beep();
break;
     }
     case 0x000A:    /*...Linefeed(newline) */
     {
break;
     }
     case 0x000D:    /*...Carriage Return */
     {
x = 0;
y += Y;
if(y>maxy-20) { x=X; y=Y; clscr(0,20,maxx,463);}
break;
     }
     case 0x000C:    /*...Formfeed */
     {
clscr(0,0,maxx,maxy);
```

```
Screen(Tmode);
x = X;
y = Y;
break;
     }
     case 0x000F:
     {
x += 8;
break;
     }
     case 0x0010:    /*...get cursor position */
     {
delay(CPU+1);
rch = getcbuf(inbuf);
if ( (ch2 = rch & 0x00FF) > 80 ) ch2=81;
switch(ch2)
{
case 0x0005: /*...NAME field */
{
  Tmode = 0;
  Screen(Tmode);
  x = (ch2 * 8);
  clscr(48,20,256,40);
  rch = getcbuf(inbuf);
  ch3 = rch & 0x00FF;
  if(ch3 == 0)
```

```
  {
    y = Y;
    break;
  }
  else break;
}
case 0x0026: /*...ADDR field */
{
  Tmode = 0;
  Screen(Tmode);
  x = (ch2 * 8)+8;
  clscr(312,20,496,40);
  rch = getcbuf(inbuf);
  ch3 = rch & 0x00FF;
  if(ch3 == 0)
  {
    y = Y;
    break;
  }
  else break;
}
case 0x0044: /*...DIR field */
{
  Tmode = 0;
  Screen(Tmode);
  x = (ch2 * 8)+8;
```

```
  clscr(552,20,maxx-1,40);
  rch = getcbuf(inbuf);
  ch3 = rch & 0x00FF;
  if(ch3 == 0)
  {
    y = Y;
    break;
  }
  else break;
}
case 0x002D: /*...AREA CODE field */
{
  x = (ch2 * 8)+8;
  clscr(360,20,424,40);
  rch = getcbuf(inbuf);
  ch3 = rch & 0x00FF;
  if(ch3 == 0)
  {
    y = Y;
    break;
  }
  else break;
}
case 0x003B: /*...TEL field */
{
  x = (ch2 * 8)+8;
```

```
    clscr(472,20,maxx-1,40);
    rch = getcbuf(inbuf);
    ch3 = rch & 0x00FF;
    if(ch3 == 0)
    {
      y = Y;
      break;
    }
    else break;
  }
  case 0x0023:    /*...Xchange telephone */
  {
    x = ch2 * 8;
    delay(CPU+1);
    rch = getcbuf(inbuf);
    ch3 = rch & 0x00FF;
    if(ch3 == 0)
    {
      y = Y;
    }
    else if(ch3 == 0x0006)
    {
      y = 120;
      clscr(280,120,maxx-1,140);
    }
    else if(ch3 == 0x0007)
```

```
    {
      y = 140;
      clscr(280,140,maxx-1,160);
      clscr(1,160,maxx-1,180);
      clscr(1,220,maxx-1,240);
    }
    else if(ch3 == 0x000B)
    {
      y = 220;
    }
    break;
  }
  case 81:
  {
    getcbuf(inbuf);
    break;
  }
  default:
  {
    delay(CPU);
    rch = getcbuf(inbuf);
    ch3 = rch & 0x00FF;
    if(ch3 == 0)
    {
      y = Y;
    }
```

```
else if((ch3 != 0) && (ch3 != 23))
{
  y = ch3 * 20;
}
else if(ch3 == 23)
{ y = 457; }
if((ch2 == 0) && (y == 440))
{
    x = 32;
    clscr(552,440,592,456);
}
else if((ch2 == 0) && (y == 457))
{
    x = 0; y = 457;
}
else if((ch2 != 0) && (y == 457))
{
    x = (ch2 * 8); y = 457;
    clscr(x+8,460,maxx,479);
}
else if(ch2 == 0x0014)
{
    x = 160;
}
else
{   x = (ch2 * 8);    }
```

```
  break;
}
}/* switch ( ch2 ... */
      } /* case 0x0010 */
     case 0x00AC:       /*...Thai mode */
     {
Screen1(Tmode);
break;
     }
     case 0x00AB:       /*...English mode */
     {
Tmode = 1;
Screen1(Tmode);
clscr(x+8,y+5,maxx,y+15);
break;
     }
     case 0x00AA:     /*...print dot between field */
     {
delay(CPU);
rch = getcbuf(inbuf);
ch2 = rch & 0x00FF;
if(ch2 == 0x003E)
{
  while(x < 496)
  {
    x += 8;
```

```
      draw_letter(46,x,y);
    }
    delay(CPU);
    rch = getcbuf(inbuf);
    ch3 = rch & 0x00FF;
    if((ch3 == 0x0058) || (ch3 == 0x0078))
    {
      f = Rcdas(ch3);
      print_r1(f);
      continue;
    }
    else continue;
  }
  else if( ch2 == 0x0025)
  {
    while(x < 296)
    {
      x += 8;
      draw_letter(46,x,y);
    }
    continue;
  }
  else
  {
    while(x < ((ch2 * 8)-8))
    {
```

```
   x += 8;
   draw_letter(46,x,y);
  }
  break;
}
     }
     case 0x00A9:    /*...print field on fixed column */
     {
delay(CPU);
rch = getcbuf(inbuf);
if((ch2 = rch & 0x00FF) > 80 ) ch2 = 81;
switch(ch2)
{
case 0x0026:
{
  rch = getcbuf(inbuf);
  ch3 = rch & 0x00FF;
  if(ch3 == 0x00A9)
  {
     rch = getcbuf(inbuf);
     ch3 = rch & 0x00FF;
     x = ch3 * 8;
  }
  else if(ch3 == 0x0010)
  {
     rch = getcbuf(inbuf);
```

```
        ch3 = rch & 0x00FF;
        x = ch3 * 8;
        rch = getcbuf(inbuf);
        ch3 = rch & 0x00FF;
        if(ch3 == 23){ y = 457; }else
        y = ch3 * 20;
    }
    else
    {
        x = 304;
        f = Rcdas(ch3);
        print_r1(f);
    }
    break;
}
case 0x0027:
{
    rch = getcbuf(inbuf);
    ch3 = rch & 0x00FF;
    if(ch3 == 0x000D)
    {
        x = X;
        y += Y;
        getcbuf(inbuf);
    }
    else if(ch3 == 0x0010)
```

```
  {
     rch = getcbuf(inbuf);
     ch3 = rch & 0x00FF;
     if(ch3 == 0) x = 32; else x = ch3 * 8;
     rch = getcbuf(inbuf);
     ch3 = rch & 0x00FF;
     if(ch3 == 23) y = 457;else
     y = ch3 * 20;
  }
  else
  {
     x = 312;
     f = Rcdas(ch3);
     print_r1(f);
  }
  break;
}
case 0x004C:    /*...clear to end of line */
  if ( getcbuf(inbuf) == 0x0010 )
    {
      delstart=getcbuf(inbuf) & 0x00FF;
      if ( delstart > 80 || delstart < 0 ) break;
      clscr(delstart*8-8,y,maxx-1,y+20);
    }
  break;
case 81:
```

```
  getcbuf(inbuf);
  break;
default:
  print_r1(0);
  x = ch2 * 8;
  break;
}
break;
     }
     }
   }
   else  /*...print typed character and received character */
   {
       if((code == ch1) && (keyed == 1))
       {
   f = Rcdas(ch1);
   print_r2(f);
   keyed = 0;
       }
       else
       {
   f = Rcdas(ch1);
   print_r1(f);
   keyed = 0;
       }
       break;
```

```
   }
}
} while ((key != 0x002D) && (flag != 1));  /*...Exit to DOS */
}

void savefont(void)
/*...Loading font from normal.fon into font[][] */
{
int i,j;
fptr = fopen("normal.fon","rb");
for(i=0;i<ROWS;i++)
   for(j=0;j<COLS;j++)
     {
       fread(&font[i][j],1,1,fptr);
     }
fclose(fptr);
}

void opengraph(void)
/*...Openning graphics system and use VGA,EGA adapter card only */
{
int driver = DETECT;
int mode;
initgraph(&driver,&mode,"");
maxx = getmaxx();
maxy = getmaxy();
```

```
setbkcolor(1);
}

void draw_letter(unsigned char f,unsigned x,unsigned y)
/*...Drawing each font Thai/Eng from fon[][] */
{
int c,j,bit;
unsigned int mask;
for(c=0;c<20;c++)
{
    mask = 0x80;
    for(j=0;j<8;j++)
    {
bit = (*(*(font+f)+c) & mask) ? 1 : 0 ;
if ( bit == 1 )
   putpixel(x,y,COLOR);
   mask >>= 1;
   x++;
    }
x -= 8;
y++;
}
}
```

```
void clscr(int x1,int y1,int x2,int y2)
/*...Clear any places of screen graphics */
{
setviewport(x1,y1,x2,y2,1);
clearviewport();
setviewport(0,0,maxx,maxy,1);
}

void beep(void)
/*...Sound beep */
{
printf("\07");
}

int keyin(int ch,char Tmode)
/*...Type in Thai/Eng and return font(f) */
/*...And also check any repeated typing */
{
int ThaiAscii[] = { 32,218,218,50,51,52,219,167,54,55,53,
                    57,193,162,227,189,168,197,45,47,192,
                    182,216,214,164,181,171,199,178,170,204,
                    198,49,196,218,169,175,174,226,172,231,
                    179,235,201,200,63,236,207,173,48,177,
                    166,184,234,206,34,41,237,40,186,92,
                    197,217,56,223,191,212,225,161,211,180,
                    224,233,195,232,210,202,183,215,185,194,
```

```
                    230,190,203,208,213,205,228,187,209,188,
                    176,124,44,42 };
if((Tmode == 0) && (ch>31))
{
   f =  ThaiAscii[(ch)-32];
   InputSt[z] = f;
   if((InputSt[z-1]>211 && InputSt[z-1]<220) &&
(InputSt[z]>211 && InputSt[z]<220))
     {
       InputSt[z] = 0;
       beep();
       cursor(x,y);
       return(0);
     }
   else if((InputSt[z-1]>230 && InputSt[z-1]<239) &&
(InputSt[z]>230 && InputSt[z]<239))
     {
       InputSt[z] = 0;
       beep();
       cursor(x,y);
       return(0);
     }
   else if((InputSt[z-1]>230 && InputSt[z-1]<239) &&
(InputSt[z]>215 && InputSt[z]<220))
     {
       InputSt[z] = 0;
```

```
        beep();
        cursor(x,y);
        return(0);
      }
    else if((InputSt[z-1]>230 && InputSt[z-1]<239) &&
(InputSt[z]>211 && InputSt[z]<216))
      {
        InputSt[z] = 0;
        beep();
        cursor(x,y);
        return(0);
      }
    else if(((InputSt[z-1] == 209) &&
(InputSt[z]>211 && InputSt[z]<219)) ||
((InputSt[z-1]>211 && InputSt[z-1]<219) &&
(InputSt[z] == 209)) || ((InputSt[z-1] == 209) &&
(InputSt[z] == 209)))
      {
        InputSt[z] = 0;
        beep();
        cursor(x,y);
        return(0);
      }
    else if((InputSt[z-1]>230 && InputSt[z-1]<239) &&
(InputSt[z] == 209))
      {
```

```
        InputSt[z] = 0;
        beep();
        cursor(x,y);
        return(0);
      }
    else
      {
        if(z<2048)
        z++;
        else z = 0;
        return (f);
      }
}
else if((Tmode == 0) && (ch == 8))
{
        f = ch;
        InputSt[z] = ch;
        if((InputSt[z-1]>215) && (InputSt[z-1]<219))
        {
    InputSt[z-1] = 0;
    clscr(x-8,y+17,x,y+20);
    cursor(x,y);
        }
        else if((InputSt[z-1] == 209) || ((InputSt[z-1]>211) &&
 (InputSt[z-1]<216)) || (InputSt[z-1] ==237) ||
 (InputSt[z-1] == 219))
```

```
        {
    InputSt[z-1] = 0;
    clscr(x-8,y,x,y+6);
    cursor(x,y);
        }
        else if((InputSt[z-1]>230) && (InputSt[z-1]<237))
        {
    InputSt[z-1] = 0;
    clscr(x-8,y,x,y+3);
    cursor(x,y);
        }
        else
        {
    InputSt[z-1] = 0;
    {
      clscr(x-8,y,x,y+20);
      x -= 8;
      cursor(x,y);
    }
        }
    z--;
    return f;
}
else if((Tmode == 1) && (ch > 31))
{
    f = ch;
```

```
    InputSt[z] = f;
    if(z<2048)
    z++;
    else z = 0;
    return f;
}
else if((Tmode == 1) && (ch == 8))
{
    f = ch;
    InputSt[z] = ch;
    {
       if(x>7)
       {
       clscr(x-8,y,x,y+20);
       x -= 8;
       cursor(x,y);
       }
       else
       {
       beep();
       x=0;
       cursor(x,y);
       return;
       }
    }
      InputSt[z-1] = 0;
```

```
        z--;
        return f;
}
}

int Scdas(int f,int Tmode)
/*...Convert CU code to GD asciicode before sending to host */
{
char Scdas[] = { 0xA1,0x22,0xA3,0xA4,0x25,0xA6,0xA7,0x28,
                 0x2D,0x2E,0x2F,0x30,0x31,0x32,0x33,0x34,
                 0x35,0x36,0x37,0x38,0x39,0x3A,0x3B,0xBC,
                 0x3D,0xBE,0x3F,0x40,0xC1,0xC2,0xC3,0xC4,
                 0xC5,0xC6,0xC7,0xC8,0xC9,0xCA,0xCB,0xCC,
                 0xCD,0xCE,0xCF,0xD0,0xD1,0xD2,0xD3,0xD4,
                 0xD5,0xD6,0xD7,0xD8,0xD9,0xDA,0xDB,0xDC,
                 0xDD,0xDE,0xDF,0xE0,0xE1,0xE2,0xE3,0xE4,
                 0xE5,0xE6,0xE7,0xE8,0xE9,0xEA,0xEB,0xEC,
                 0xED,0xEE,0xEF,0xF0,0xF1,0xF2,0xF3,0xF4,
                 0xF5,0xF6,0xF7,0xF8,0xF9,0xFA,0xFB,0xFC,
                 0xFD,0xFE };
if(Tmode == 0)
{
switch(f)
{
   case 161:
   code = Scdas[28];
```

```
break;
case 162:
code = Scdas[29];
break;
case 164:
code = Scdas[30];
break;
case 198:
code = Scdas[25];
break;
case 207:
code = Scdas[89];
break;
case 123:
code = Scdas[86];
break;
case 230:
code = Scdas[87];
break;
case 125:
code = Scdas[88];
break;
case 232:
code = Scdas[0];
break;
case 34:
```

```
code = Scdas[1];
break;
case 233:
code = Scdas[2];
break;
case 234:
code = Scdas[3];
break;
case 37:
code = Scdas[4];
break;
case 235:
code = Scdas[5];
break;
case 231:
code = Scdas[6];
break;
case 32:
code = 0x20;
break;
case 40:
code = Scdas[7];
break;
case 41:
code = 0x29;
break;
```

```
    case 44:
    code = 0x2C;
    break;
    case 236:
    code = Scdas[23];
    break;
    case 61:
    code = Scdas[24];
    break;
    case 63:
    code = Scdas[26];
    break;
    case 8:
    code = 0x7F;
    break;
    default:
    break;
}
if(f>165 && f<198)
    {
    code = Scdas[f-135];
    }
else if(f>198 && f<207)
    {
    code = Scdas[f-136];
    }
```

```
else if(f>207 && f<218)
   {
   code = Scdas[f-137];
   }
else if(f>223 && f<229)
   {
   code = Scdas[f-143];
   }
else if(f>44 && f<60)
   {
   code = Scdas[f-37];
   }
return code;
}
else if(Tmode == 1)
{
   if(f == 8)
   {
      code = 0x7F;
   }
   else
   {
      code = f;
   }
return code;
}
```

```
unsigned char Rcdas(unsigned char code)
{     /*...Convert GD asciicode to CU before displaying */
unsigned char  Rcdas[] = { 232,34,233,234,37,235,231,40,
                           45,46,47,48,49,50,51,52,
                           53,54,55,56,57,58,59,236,
                           61,198,63,64,161,162,164,166,
                           167,168,169,170,171,172,173,174,
                           175,176,177,178,179,180,181,182,
                           183,184,185,186,187,188,189,190,
                           191,192,193,194,195,196,197,199,
                           200,201,202,203,204,205,206,208,
                           209,210,211,212,213,214,215,216,
                           217,224,225,226,227,228,123,230,
                           125,207,255 };
 if((code > 160) && (code < 169))
 {
     f = Rcdas[code-161];
 }
 else if(code > 172)
 {
     f = Rcdas[code-165];
 }
 else if((code > 32) && (code < 128))
 {
     f = code;
 }
```

```
 else
 {
     f = code;
 }
     return f;
}

void Screen(unsigned int Tmode)
/*...Display terminal screen consist of Thai/Eng and Function key */
{
if(Tmode == 0)
{
   clscr(0,0,112,16);
   setcolor(MAGENTA);
   rectangle(0,0,112,16);
   setfillstyle(SOLID_FILL, MAGENTA);
   floodfill(56,8, MAGENTA);
   setcolor(LIGHTGREEN);
   outtextxy(20,4,"Thai Mode");
}
else if(Tmode == 1)
{
   clscr(0,0,112,16);
   setcolor(RED);
   rectangle(0,0,112,16);
   setfillstyle(SOLID_FILL, RED);
```

```
    floodfill(56,8, RED);
    setcolor(WHITE);
    outtextxy(10,4,"English Mode");
}
setcolor(YELLOW);
rectangle(112,0,maxx,16);
setfillstyle(SOLID_FILL, YELLOW);
floodfill(374,8, YELLOW);
setcolor(BLUE);
outtextxy(128,5,"  -Name");
outtextxy(204,5,"  -Addr/Area");
outtextxy(320,5,"  -Dir/Tel");
outtextxy(420,5,"   -Menu");
outtextxy(504,5,"     -Thai/Eng");
setcolor(RED);
outtextxy(128,5,"F1");
outtextxy(204,5,"F3");
outtextxy(320,5,"F5");
outtextxy(420,5,"F10");
outtextxy(504,5,"Alt+T");
setcolor(YELLOW);
rectangle(150,463,480,479);
setfillstyle(SOLID_FILL, YELLOW);
floodfill(315,471, YELLOW);
setcolor(BLUE);
outtextxy(165,468,"       -Help");
```

```
outtextxy(280,468,"     -Print");
outtextxy(385,468,"     -Exit");
setcolor(RED);
outtextxy(165,468,"Ctrl+F1");
outtextxy(280,468,"Alt+P");
outtextxy(385,468,"Alt+X");
}

void Screen1(unsigned int Tmode)
{
        if(Tmode == 0)
{
   clscr(0,0,112,16);
   setcolor(MAGENTA);
   rectangle(0,0,112,16);
   setfillstyle(SOLID_FILL, MAGENTA);
   floodfill(56,8, MAGENTA);
   setcolor(LIGHTGREEN);
   outtextxy(20,4,"Thai Mode");
}
else if(Tmode == 1)
{
   clscr(0,0,112,16);
   setcolor(RED);
   rectangle(0,0,112,16);
   setfillstyle(SOLID_FILL, RED);
```

```
   floodfill(56,8, RED);
   setcolor(WHITE);
   outtextxy(10,4,"English Mode");
}
setcolor(YELLOW);
rectangle(112,0,maxx,16);
setfillstyle(SOLID_FILL, YELLOW);
floodfill(374,8, YELLOW);
setcolor(BLUE);
outtextxy(128,5,"  -Name");
outtextxy(204,5,"  -Addr/Area");
outtextxy(320,5,"  -Dir/Tel");
outtextxy(420,5,"   -Menu");
outtextxy(504,5,"     -Thai/Eng");
setcolor(RED);
outtextxy(128,5,"F1");
outtextxy(204,5,"F3");
outtextxy(320,5,"F5");
outtextxy(420,5,"F10");
outtextxy(504,5,"Alt+T");
}

void cursor(unsigned n,unsigned m)
/*...Display cursor anywhere in graphics mode */
{
 if ( n > maxx || m > maxy ) return;
```

```
 setcolor(YELLOW);
 line(n,m+19,n+7,m+19);
 line(n,m+20,n+7,m+20);
 clscr(n,m+19,n+7,m+20);
}

void print_r1(unsigned char f)
  /*...Displaying the recieving code on the screen */
{
  if((x>maxx-8) && (y == 440) && !((f == 209) || (f>211 && f<220) ||
       (f>230 && f<239)))
  { x=64; y=457; clscr(150,463,480,479); }
  if((x>maxx-8) && (y >= maxy-20)) return;
  if((f == 255) || (f == 32))
  {
     x += 8;
  }
  else if((f == 209) || (f>211 && f<220) ||
       (f>230 && f<239))
  {
     draw_letter(f,x,y);
  }
  else if((f>32 && f<127))
  {
     x += 8;
     draw_letter(f,x,y);
```

```
  }
  else if((f>160 && f<209) || (f>209 && f<212) ||
(f>219 && f<231))
  {
     x += 8;
     draw_letter(f,x,y);
  }
  else
  {
      f = code;
      draw_letter(0,x,y);
  }
}

void init_port(void)
/*...Initial communication port (COM) */
{
int in;
in = inportb(0x21);
in = in & 0xE7;
outportb(0x21,in);
outportb(LCR,0x80);
outportb(THR,0x0C);
outportb(IER,0x00);
outportb(LCR,0x03);
outportb(MCR,0x0B);
```

```
outportb(IER,0x01);
}

void shutdown(void)
/*...Set INTR back to old INTR vector and shutdown */
{
unsigned char mask1;
mask1 = inportb(0x21);
mask1 = mask1 |0x18;
outportb(0x21,mask1);
setvect(INTR, oldhandler);
}

void print_r2(unsigned char f)
/*...Displaying the typing code that eco from host */
{
       if(f == 255)
{
    x += 8;
    cursor(x,y);
}
else if(f == 32)
{
    x += 8;
    if(x>=maxx) { beep(); x=maxx-8; cursor(x,y); }else
    cursor(x,y);
```

```
}
else if((f == 209) || (f > 211 && f < 220) ||
(f > 230 && f < 239))
{
    x -= 8;
    draw_letter(f,x,y);
    x += 8;
    if(x>=maxx-1) { beep(); x=maxx-8; clscr(x,y+5,x+8,y+15);
    cursor(x,y); }else cursor(x,y);
}
else if((f > 32 && f < 127))
{
    draw_letter(f,x,y);
    x += 8;
    if(x>=maxx-1) { beep(); x=maxx-8; clscr(x,y+5,x+8,y+15);
    cursor(x,y); }else cursor(x,y);
}
else if((f > 160 && f < 209) || (f >209 && f < 212) ||
(f > 219 && f < 231))
{
    draw_letter(f,x,y);
    x += 8;
    if(x>=maxx-1) { beep(); x=maxx-8; clscr(x,y+5,x+8,y+15);
    cursor(x,y); }else cursor(x,y);
}
}
```

```
void PageSignOn(void)
/*...Showing page that teach how to sign on */
{
clscr(0,0,maxx,maxy);
setcolor(YELLOW);
rectangle(240,195,370,235);
setfillstyle(SOLID_FILL, RED);
floodfill(300,212, YELLOW);
clscr(240,195,370,235);
setcolor(YELLOW);
rectangle(230,190,380,240);
setfillstyle(SOLID_FILL, RED);
floodfill(310,212, YELLOW);
clscr(230,190,380,240);
setcolor(YELLOW);
rectangle(220,185,390,245);
setfillstyle(SOLID_FILL, RED);
floodfill(310,212, YELLOW);
clscr(220,185,390,245);
setcolor(YELLOW);
rectangle(210,180,400,250);
setfillstyle(SOLID_FILL, RED);
floodfill(310,212, YELLOW);
clscr(210,180,400,250);
setcolor(YELLOW);
rectangle(200,175,410,255);
```

```
setfillstyle(SOLID_FILL, RED);
floodfill(310,212, YELLOW);
outtextxy(220,200,"NOW READY TO SIGN ON !");
outtextxy(240,220,"** PRESS Home **");
setcolor(BLUE);
rectangle(200,175,410,255);
setcolor(YELLOW);
rectangle(85,315,260,350);
setfillstyle(SOLID_FILL, GREEN);
floodfill(172,332, YELLOW);
outtextxy(100,330,"THE WAY TO SIGN ON");
outtextxy(220,370,"USER :  TOT   <Enter>");
outtextxy(220,390,"PASSWORD :  TOT   <Enter>");
setcolor(BLUE);
rectangle(85,315,260,350);
}

void Chk_TXempty(void)
/*...Check the status of Transmitter holding register empty */
 /*...before sending another character */
{
int Tx;
do
{
  Tx = inportb(LSR);
  Tx = Tx & 0x20;
```

```
} while(!Tx);
}

void Select_COM(void)
/*...Select communication port(COM1/COM2), CPU (286/386/486) */
{
int l;
int x1=505,y1=115;
int TRUE = 1,TRUE1 = 1;
int curpos = 0,curpos1 = 0;
setcolor(RED);
rectangle(45,35,590,180);
setfillstyle(SOLID_FILL, RED);
floodfill(300,100, RED);
setcolor(WHITE);
outtextxy(80,60,"This CDAS emulation program can be run on IBM PC or");
outtextxy(80,80,"IBM PC compatible and use VGA or EGA adapter card.");
outtextxy(80,100,"Please use modem speed 2400 bps and set to 8 data ");
outtextxy(80,120,"bits , 1 stop bit , parity bit none and ASYNC mode.");
outtextxy(80,145,"      *****  Created by                              *****");
setcolor(YELLOW);
rectangle(278,141,437,155);
setfillstyle(SOLID_FILL, RED);
floodfill(356,150, YELLOW);
outtextxy(50,145,"                                  NUTTAVUDH SATRAVAHA");
setcolor(RED);
```

```
rectangle(278,141,437,155);
setcolor(LIGHTMAGENTA);
outtextxy(50,160,"                Telephone Organization of Thailand");
for(l=0;l<55;l++)
{
   draw_LOGO(l,x1,y1);
   y1++;
}
setcolor(YELLOW);
rectangle(50,40,585,175);
rectangle(52,42,583,173);
printf("\07");
displayCOM();
while(TRUE)
{
switch(getcode())
    {
    case 72:
       if(curpos>0)
       {
 --curpos;
 display(curpos);
 break;
       }
       else break;
    case 80:
```

```
        if(curpos<1)
        {
++curpos;
display(curpos);
break;
        }
        else break;
    case '\r':
        action(curpos);
        TRUE = 0;
        break;
    }
}
displayCPU();
while(TRUE1)
{
   switch(getcode())
   {
   case 72:
      if(curpos1>0)
      {
--curpos1;
show(curpos1);
break;
      }
      else break;
```

```
   case 80:
      if(curpos1<2)
      {
++curpos1;
show(curpos1);
break;
      }
      else break;
   case '\r':
      action1(curpos1);
      TRUE1 = 0;
      break;
   }
}
setcolor(MAGENTA);
rectangle(270,320,480,365);
setfillstyle(SOLID_FILL, MAGENTA);
floodfill(370,340, MAGENTA);
setcolor(LIGHTCYAN);
outtextxy(280,330,"NOW, DIAL MODEM PLEASE !    ");
outtextxy(280,350,"  <  Press any key  >      ");
getch();
}
```

```
void display(int pos)
/*...Display menu for COM1/COM2 */
{
if(pos == 0)
{
   clscr(100,215,150,230);
   clscr(100,245,150,260);
   displayCOM1();
}
if(pos == 1)
{
   clscr(100,215,150,230);
   clscr(100,245,150,260);
   displayCOM2();
}
}

void show(int pos)
{
if(pos == 0)
{
   clscr(200,270,250,285);
   clscr(200,295,250,310);
   displayCPU1();
}
if(pos == 1)
```

```
{
   clscr(200,270,250,285);
   clscr(200,295,250,310);
   clscr(200,315,250,330);
   displayCPU2();
}
if(pos == 2)
{
   clscr(200,295,250,310);
   clscr(200,315,250,330);
   displayCPU3();
}
}

char getcode(void)
/*...Receive the selection choice and return to Select_COM() */
{
int keyx;
if((keyx=getch()) == 0)
    return(getch());
else if(keyx == '\r')
    return(keyx);
else
    return(0);
}
```

```
void action(int pos)
/*...Address of UART register */
/*...for COM1 or COM2 */
{
switch(pos)
{
  case 0:                  /*...COM1 */
     RBR = 0x3F8;          /*...Receiver Buffer Register */
     THR = 0x3F8;          /*...Transmitter Holding Register */
     IER = 0x3F9;          /*...Interrupt Enable Register */
     LCR = 0x3FB;          /*...Line Control Register */
     MCR = 0x3FC;          /*...Modem Control Register */
     LSR = 0x3FD;          /*...Line Status Register */
     INTR = INTR1;         /*...Interrupt COM1 */
     break;
  case 1:                  /*...COM2 */
     RBR = 0x2F8;          /*...Receiver Buffer Register */
     THR = 0x2F8;          /*...Transmitter Holding Register */
     IER = 0x2F9;          /*...Interrupt Enable Register */
     LCR = 0x2FB;          /*...Line Control Register */
     MCR = 0x2FC;          /*...Modem Control Register */
     LSR = 0x2FD;          /*...Line Status Register */
     INTR = INTR2;         /*...Interrupt COM2 */
     break;
}
}
```

```
void action1(int pos1)
/*...Delay CPU-X86 for number of milliseconds */
/*...for each CPU can receive every characters from host */
{
switch(pos1)
{
case 0:                    /*...Delay CPU-286 */
     CPU = 4.86;           /*...milliseconds */
     break;
case 1:                    /*...Delay CPU-386 */
     CPU = 5.86;
     break;
case 2:                    /*...Delay CPU-486 */
     CPU = 6.86;
     break;
}
}

void displayCOM(void)
{
setcolor(DARKGRAY);
rectangle(90,190,280,270);
line(90,210,280,210);
setfillstyle(SOLID_FILL, YELLOW);
floodfill(185,200, DARKGRAY);
outtextxy(100,197, "PLEASE SELECT COM PORT" );
```

```
setfillstyle(SOLID_FILL, LIGHTGRAY);
floodfill(185,240, DARKGRAY);
setcolor(WHITE);
rectangle(100,215,150,230);
setfillstyle(SOLID_FILL, RED);
floodfill(125,220, WHITE);
outtextxy(100,220, " COM1 " );
outtextxy(100,250, " COM2 " );
setcolor(YELLOW);
line(90,210,280,210);
setcolor(BLUE);
rectangle(90,190,280,270);
}

void displayCOM1(void)
{
setcolor(WHITE);
rectangle(100,215,150,230);
setfillstyle(SOLID_FILL, RED);
floodfill(125,220, WHITE);
outtextxy(100,220, " COM1 " );
setfillstyle(SOLID_FILL, LIGHTGRAY);
floodfill(125,250, LIGHTGRAY);
outtextxy(100,250, " COM2 " );
}
```

```
void displayCOM2(void)
{
setfillstyle(SOLID_FILL, LIGHTGRAY);
floodfill(125,220, LIGHTGRAY);
outtextxy(100,220, " COM1 " );
setcolor(WHITE);
rectangle(100,245,150,260);
setfillstyle(SOLID_FILL, RED);
floodfill(125,250, WHITE);
outtextxy(100,250, " COM2 " );
}

void displayCPU(void)
{
setcolor(RED);
rectangle(180,240,405,350);
line(180,260,405,260);
setfillstyle(SOLID_FILL, YELLOW);
floodfill(290,250, RED);
outtextxy(195,247, "PLEASE SELECT CPU TO USE" );
setfillstyle(SOLID_FILL, GREEN);
floodfill(290,330, RED);
setcolor(WHITE);
rectangle(200,270,250,285);
setfillstyle(SOLID_FILL, BROWN);
floodfill(225,280, WHITE);
```

```
outtextxy(200,275, " 80286 " );
outtextxy(200,300, " 80386 " );
outtextxy(200,320, " 80486 " );
setcolor(BLUE);
rectangle(180,240,405,350);
setcolor(YELLOW);
line(180,260,405,260);
setcolor(LIGHTGRAY);
line(180,240,280,240);
line(180,240,180,270);
}

void displayCPU1(void)
{
setcolor(WHITE);
rectangle(200,270,250,285);
setfillstyle(SOLID_FILL, BROWN);
floodfill(225,280, WHITE);
outtextxy(200,275, " 80286 " );
setfillstyle(SOLID_FILL, GREEN);
floodfill(225,305, GREEN );
outtextxy(200,300, " 80386 " );
setfillstyle(SOLID_FILL, GREEN);
floodfill(225,320, GREEN);
outtextxy(200,320, " 80486 " );
}
```

```
void displayCPU2(void)
{
setfillstyle(SOLID_FILL, GREEN);
floodfill(225,280, GREEN);
outtextxy(200,275, " 80286 " );
setcolor(WHITE);
rectangle(200,295,250,310);
setfillstyle(SOLID_FILL, BROWN);
floodfill(225,305, WHITE);
outtextxy(200,300, " 80386 " );
setfillstyle(SOLID_FILL, GREEN);
floodfill(225,320, GREEN);
outtextxy(200,320, " 80486 " );
}

void displayCPU3(void)
{
setfillstyle(SOLID_FILL, GREEN);
floodfill(225,280, GREEN);
outtextxy(200,275, " 80286 " );
setfillstyle(SOLID_FILL, GREEN);
floodfill(225,305, GREEN);
outtextxy(200,300, " 80386 " );
setcolor(WHITE);
rectangle(200,315,250,330);
setfillstyle(SOLID_FILL, BROWN);
```

```
floodfill(225,320, WHITE);
outtextxy(200,320, " 80486 " );
}

void Printer(void)
/*...Dump screen to printer */
/*...by pressing Alt-P */
{
char m;
int i,j,k,Msb,Lsb;
setviewport(0,0,maxx,maxy,0);
fprintf(stdprn, "\x1B%c%c",0x41,7);
Lsb = (maxx+1) & 0x00FF;
Msb = (maxx+1) >> 8;
for(j=0;j<=maxy/8;j++)
{
   fprintf(stdprn, "\x1B*%c%c%c",6, Lsb, Msb);
   for(i=0;i<=maxx;i++)
   {
      m = 0;
      for(k=0;k<8;k++)
      {
m <<= 1;
if(getpixel(i,j*8+k)) m++;
      }
      fprintf(stdprn, "%c", m);
```

```
   }
fprintf(stdprn, "\x0D\x0A");
}
fprintf(stdprn, "\f");
}

void sure(int banner)
{
size = (unsigned int)imagesize(170,175,470,255);
pagebuff = (void *)malloc(size);
getimage(170,175,470,255, pagebuff);
clscr(170,175,470,255);
setcolor(RED);
rectangle(170,175,470,255);
setfillstyle(SOLID_FILL, YELLOW);
floodfill(374,212, RED);
if(banner == 1)
{
  outtextxy(185,210,"HAVE YOU ALREADY SIGNED OFF? (Y/N)");
  setcolor(RED);
  rectangle(172,177,468,253);
  rectangle(175,180,465,250);
  setcolor(BLUE);
  rectangle(170,175,470,255);
  beep();
}
```

```
else if(banner == 2)
{
  outtextxy(264,185,"** WARNING **");
  outtextxy(224,210,"PLEASE WAIT UNTIL FINISH");
  outtextxy(245,235,"< PRESS ANY KEY >");
  setcolor(RED);
  rectangle(172,177,468,253);
  rectangle(175,180,465,250);
  setcolor(BLUE);
  rectangle(170,175,470,255);
  beep();
  getch();
  clscr(170,175,470,255);
  putimage(170,175, pagebuff, COPY_PUT);
  free((void *)pagebuff);
}
}

void help(void)
{
clscr(100,60,540,420);
setcolor(LIGHTGRAY);
rectangle(100,60,540,420);
setfillstyle(SOLID_FILL, LIGHTGRAY);
floodfill(320,240, LIGHTGRAY);
setcolor(RED);
```

```
line(105,73,291,73);
line(105,75,291,75);
line(340,73,535,73);
line(340,75,535,75);
line(105,73,105,413);
line(535,73,535,413);
line(105,413,269,413);
line(405,413,535,413);
outtextxy(125,70,"                    Help");
outtextxy(125,90,"Home   Press 'Home' key to sign on.");
outtextxy(125,105,"F1    Move cursor to the beginning of the 'NAME'");
outtextxy(125,120,"      field.");
outtextxy(125,135,"F2    More (PgDn)");
outtextxy(125,150,"F3    Move cursor to the beginning of the 'ADDR'");
outtextxy(125,165,"      or 'AREA CODE' field.");
outtextxy(125,180,"F4    Back (PgUp)");
outtextxy(125,195,"F5    Move cursor to the beginning of the 'DIR'");
outtextxy(125,210,"      or 'TEL' field.");
outtextxy(125,225,"      'DIR' field consists of 1 or 3 letters.");
draw_letter(185,189,231);
outtextxy(125,240,"        - Business and Residential File");
draw_letter(195,189,246);
outtextxy(125,255,"        - Goverment File");
draw_letter(183,189,261);
outtextxy(125,270,"        - Frequently Called Number (FCN) File");
outtextxy(125,285,"F7    Select the number of main record and then");
```

```
outtextxy(125,300,"        'F7' for more detail.");
outtextxy(125,315,"F8    Select the alternative spelling and 'F8'.");
outtextxy(125,330,"F9    Refreshes the screen with an empty format.");
outtextxy(125,345,"F10   Press 'F10' key to main menu.");
outtextxy(125,360,"Enter   Indicates the input is complete and send");
outtextxy(125,375,"        the information to host computer.");
outtextxy(125,390,"Alt+P   Dump screen to printer.");
draw_letter(69,277,400);
draw_letter(115,285,400);
draw_letter(99,293,400);
outtextxy(125,409,"                          - Exit Help");
while(getch() != 27); return;
}

void draw_LOGO(int l,int x,int y)
{
     int logo[55][7] = { {0x00,0x00,0x00,0x20,0x00,0x00,0x00},
                         {0x00,0x00,0x01,0x74,0x00,0x00,0x00},
                         {0x00,0x00,0x01,0x74,0x00,0x00,0x00},
                         {0x00,0x00,0x09,0x04,0x80,0x00,0x00},
                         {0x00,0x00,0x05,0x05,0x00,0x00,0x00},
                         {0x00,0x00,0x02,0x02,0x00,0x00,0x00},
                         {0x00,0x01,0xFC,0xA9,0xFC,0x00,0x00},
                         {0x00,0x00,0x01,0x24,0x00,0x00,0x00},
                         {0x00,0x00,0x06,0x73,0x00,0x00,0x00},
                         {0x00,0x08,0x39,0x74,0xE1,0x00,0x00},
```

```
{0x00,0x0C,0x7C,0x0a,0xf1,0x80,0x00},
{0x00,0x1C,0x72,0x02,0xF1,0xC0,0x00},
{0x00,0x3C,0x24,0xF9,0x21,0xE0,0x00},
{0x00,0x3D,0x0C,0x01,0x85,0xE0,0x00},
{0x00,0x3F,0x89,0xFC,0x8F,0xE0,0x00},
{0x00,0x3F,0x80,0x00,0x0F,0xE0,0x00},
{0x00,0xBF,0x13,0xFE,0x47,0xE8,0x00},
{0x01,0xDA,0x28,0x00,0xA2,0xDC,0x00},
{0x01,0xD8,0x50,0x20,0x50,0xDC,0x00},
{0x01,0xCC,0xA4,0x71,0x29,0x9C,0x00},
{0x01,0xE2,0x47,0xDF,0x12,0x3C,0x00},
{0x01,0xF1,0x00,0x00,0x08,0x7C,0x00},
{0x01,0xFA,0x88,0x00,0x94,0xFC,0x00},
{0x00,0xFA,0x90,0x00,0x54,0xF8,0x00},
{0x00,0xFA,0x83,0x07,0x14,0xF8,0x00},
{0x00,0x62,0xA7,0x8F,0x34,0x30,0x00},
{0x01,0x60,0x23,0xDE,0x20,0x34,0x00},
{0x01,0x28,0x21,0xDC,0x20,0xA4,0x00},
{0x80,0x07,0x20,0x70,0x27,0x00,0x08},
{0xFD,0xF7,0xA0,0x70,0x2F,0x7D,0xF8},
{0x00,0x03,0xA0,0xD8,0x2E,0x00,0x00},
{0x7D,0xF1,0xA1,0x04,0x2C,0x7D,0xF0},
{0x00,0x01,0x20,0x00,0x24,0x00,0x00},
{0x3E,0xF1,0x28,0x00,0x64,0x7B,0xE0},
{0x00,0x02,0x2A,0x80,0xA2,0x00,0x00},
{0x1E,0xF0,0x2A,0x2A,0x20,0x7B,0xC0},
```

```
                    {0x00,0x02,0x22,0xAA,0xA2,0x00,0x00},
                    {0x0F,0x79,0x20,0x21,0x24,0x77,0x00},
                    {0x00,0x00,0xA0,0x82,0x24,0x00,0x00},
                    {0x03,0xDC,0x90,0x28,0x25,0xDC,0x00},
                    {0x00,0x00,0x48,0x00,0x48,0x00,0x00},
                    {0x00,0xF6,0x27,0xDF,0x93,0x30,0x00},
                    {0x00,0x00,0x10,0x00,0X20,0x00,0x00},
                    {0x00,0x6A,0x8F,0xDF,0xCA,0xC0,0x00},
                    {0x00,0x00,0x00,0x00,0x00,0x00,0x00},
                    {0x00,0x10,0x01,0xAA,0x81,0x00,0x00},
                    {0x00,0x08,0x10,0x00,0X42,0x00,0x00},
                    {0x00,0x05,0x05,0x55,0x04,0x00,0x00},
                    {0x00,0x00,0xE8,0x00,0x70,0x00,0x00},
                    {0x00,0x00,0x44,0x71,0x20,0x00,0x00},
                    {0x00,0x00,0x06,0x73,0x00,0x00,0x00},
                    {0x00,0x00,0x03,0x26,0x00,0x00,0x00},
                    {0x00,0x00,0x03,0x8E,0x00,0x00,0x00},
                    {0x00,0x00,0x01,0x8C,0x00,0x00,0x00},
                    {0x00,0x00,0x00,0x88,0x00,0x00,0x00} };
int c,j,bit;
unsigned mask;
for(j=0;j<7;j++)
{
   mask = 0x80;
   for(c=0;c<8;c++)
   {
```

```
        bit = (*(*(logo+l)+j) & mask) ? 1 : 0 ;
        if(bit == 1)
   putpixel(x,y,LIGHTGREEN);
   mask >>= 1;
   x++;
     }
 }
}

void help1(void)
{
int hhit;
size1 = imagesize(100,60,540,180);
size2 = imagesize(100,181,540,300);
size3 = imagesize(100,301,540,420);
if ( coreleft() < (unsigned long)size1+size2+size3+1024UL )
  { setcolor(RED);outtextxy(200,237,"ERROR! MEMORY NOT ENOUGH");
    return ; }
pagebuf1 = malloc(size1);
getimage(100,60,540,180, pagebuf1);
pagebuf2 = malloc(size2);
getimage(100,181,540,300, pagebuf2);
size3 = imagesize(100,301,540,420);
pagebuf3 = malloc(size3);
getimage(100,301,540,420, pagebuf3);
clscr(100,100,540,340);
```

```
do
{
setcolor(LIGHTGRAY);
rectangle(100,100,540,340);
setfillstyle(SOLID_FILL, LIGHTGRAY);
floodfill(320,220, LIGHTGRAY);
setcolor(RED);
line(105,113,291,113);
line(105,115,291,115);
line(340,113,535,113);
line(340,115,535,115);
line(105,113,105,333);
line(535,113,535,333);
line(105,333,269,333);
line(405,333,490,333);
line(526,333,535,333);
outtextxy(125,110,"                    Help");
outtextxy(115,130,"Enquiry      Search by name.");
outtextxy(115,145,"              This method has many ways.");
outtextxy(115,160,"               -search by full name.");
outtextxy(115,175,"               -search by primary name.");
outtextxy(115,190,"               -search by primary name and");
outtextxy(115,205,"                partial secondary name.");
outtextxy(115,220,"               -search by partial primary name");
outtextxy(115,235,"                and secondary name.");
outtextxy(115,250,"               -search by partial primary name");
```

```
outtextxy(115,265,"                    and partial secondary name.");
outtextxy(115,285,"Tel No. Retrieval   Search by telephone number.");
outtextxy(115,305,"Xchange Telephone   Search for new telephone no.");
draw_letter(69,277,320);
draw_letter(115,285,320);
draw_letter(99,293,320);
outtextxy(125,329,"                        - Exit Help");
draw_letter(80,492,320);
draw_letter(103,500,320);
draw_letter(68,508,320);
draw_letter(110,516,320);
hhit = getch();
if(hhit == 0)
{
   hhit = getch();
          if(hhit == 81)
   {  help();break;  }
}
}
while(hhit != 27);
   putimage(100,60, pagebuf1, COPY_PUT);
   putimage(100,181, pagebuf2, COPY_PUT);
   putimage(100,301, pagebuf3, COPY_PUT);
   free(pagebuf1);
   free(pagebuf2);
   free(pagebuf3);
```

```
}

void dial_MODEM(void)
{
int sch,key,kchl,kchh;
int X1=8;
int Y1=20;
clscr(0,0,maxx,maxy);
screenD();
setcolor(YELLOW);
outtextxy(0,30," You can dial modem by keyboard command.");
outtextxy(0,45,"     ATDTxxxxxxx        ** for touch tone dial **");
outtextxy(0,60,"     ATDPxxxxxxx        ** for pulse dial **");
outtextxy(0,75,"     ATDxWTxxxxxxx      ** for PABX line **");
outtextxy(0,100," <AT Command>");
x = 112; y = 90;
do
{
    cursor(x,y);
    if(bioskey(1))
    {
       key = bioskey(0);
       putcbuf(lo(key),kbuf);
       putcbuf(hi(key),kbuf);
    }
    if((sch = getcbuf(kbuf)) != -1)
```

```
  {
     kchl = sch;
     getcbuf(kbuf);
     if(kchl == 0x0D)
     {  outportb(THR,0x0D);  }
     else
     {
f = keyin(kchl,1);
keyed = 1;
if(f==32){x+=8;cursor(x,y);}
code = f;
Chk_TXempty();
outportb(THR, code);
code = code & 0x00FF;
     }
  }
  while((rch = getcbuf(inbuf)) != -1)
  {
     ch1 = rch & 0x00FF;
     switch(ch1)
     {
case 0x000C:
     clscr(0,20,maxx,maxy);
     screenD();
     x=X1; y=Y1;
     cursor(x,y);
```

```
        break;
 case 0x000D:
        x=X1; y += Y1;
        if(y>maxy-20)
        {
clscr(0,20,maxx,maxy); screenD();
x=X1; y=Y1;
        }
        break;
 default:
        if((code == ch1) && (keyed == 1))
        {
 f = Rcdas(ch1);
 print_r3(f);
 keyed = 0;
        }
        else
        {
 f = Rcdas(ch1);
 print_r3(f);
 keyed = 0;
        }
        break;
 }
         }
}while(key != 0x011B);
```

```
}

void print_r3(unsigned char f)
{
if((f>32) && (f<128))
{
   if(y>maxy-20) return;
   draw_letter(f,x,y);
   x += 8;
   cursor(x,y);
}
}

void screenD(void)
{
setcolor(LIGHTGRAY);
line(3,9,250,9);
line(360,9,636,9);
line(3,12,250,12);
line(360,12,636,12);
line(3,9,3,474);
line(3,474,420,474);
line(605,474,636,474);
line(636,9,636,474);
        setcolor(YELLOW);
outtextxy(250,8,"  DIAL MODEM");
```

```
outtextxy(250,10,"  _________");
outtextxy(250,470,"                          'Esc'- Exit to Sign on");
}
```

ประวัติผู้เขียน

นายณัฐวุฒิ ศาสตราวาหะ เกิดเมื่อวันที่ 19 พฤษภาคม พ.ศ.2508 ที่จังหวัดสมุทรปราการ สำเร็จการศึกษา วิศวกรรมศาสตรบัณฑิต สาขาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้า เจ้าคุณทหารลาดกระบัง เมื่อปี พ.ศ.2529 และเข้าศึกษาต่อในหลักสูตร วิทยาศาสตรมหาบัณฑิต สาขาวิทยาศาสตร์คอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย เมื่อ พ.ศ. 2532 ปัจจุบันทำงานที่ องค์การโทรศัพท์แห่งประเทศไทย ตำแหน่งนายช่างอันดับ 2 ทำหน้าที่ หัวหน้าหน่วยควบคุมระบบการค้นหาเลขหมายด้วยคอมพิวเตอร์