รายการอ้างอิง

CCITT Recommendations I.310-I.470 , Blue Book , Volume III , Fascicle III.8 " ISDN Overall Network Aspects and Functions , ISDN User-Network Interfaces ", Ixth Plenary Assembly , Melbourne , 14-25 November 1988.

____. Q.930-Q.940 , Blue Book , Volume VI , Fascicle VI.11 " Digital Subscriber Signalling System No.1 (DSS1) , User-Network Management" , Ixth Plenary Assembly , Melbourne , 14-25 November 1988.

____. Q.920-Q.921 , Blue Book , Volume VI , Fascicle VI.10 ," Digital Subscriber Signalling System No.1 (DSS1) , Data Link Layer " Ixth Plinary Assimbly , Melbourne , 14-25 Novimber 1988.

G. Dicenet , Design and Prospects for the ISDN 2nd ed , Artech House , Inc., Norwood , 1987

Intel Corporation , MCS-51 Macro Assembler User's Guide for DOS Systems ,1986.

____. On-Chip Peripherals and Hardware Interfacing ,1989.

Pramode K. Verma , ISDN Systems Architecture Technology and Applications , AT&T Bell Laboratories , Prenticd-Hall International, 1990.

Siemens , ISDN Subscriber Access Controller ISAC-S PEB 2085. Version 02.92 , 1992

William Stallings , ISDN an Introduction. 2nd ed , New York: Macmillan, 1989.

ภาคผนวก ก

Software ที่ใช้กับ TE

```
LCD_PORT        EQU   0A000H  ; LCD ADDRESS
KBD_PORT        EQU   0C000H  ; KEYBOARD ADDRESS
KBD_BUFFER      EQU   07FECH  ; KEYBOARD BUFFER 20 BYTE LONG
KBD_PTR         EQU   07FEBH  ; KEEP LENGTH OF DATA IN KBD BUFFER
GET_KEY         EQU   07FEAH  ; RETURN COMMAND FOR MAIN HANDLING
LAST_CALL       EQU   07FD5H  ; SAVE LAST CALL INCLUDE POINTER
KBD_TIME        EQU   07FD2H  ; COUNTER FOR KBD TIME OUT (3 BYTES)
L1_TIME         EQU   07FCFH  ; COUNTER FOR TIME OUT T3 (3 BYTES)
TE_STATUS       EQU   07FCEH  ; STATUS OF TE
                              ;  XjighdefB
                              ; d: B1 CH STATUS  1=CH IN USE
                              ; e: B2 CH STATUS
                              ; f: D CH STATUS
                              ; g: L2 ESTABLISHED
                              ; h: SABME WAS SENT
                              ; i: POWER UP/DW MODE
                              ; j: LAYER 1 ACTIVATE
TE_TEI          EQU   07FCDH
L2_TIME         EQU   07FCFH  ; COUNTER FOR TIME OUT T202 (3 BYTES)
TIME_OUT_CNT    EQU   00005H  ; TIME OUT EACH VALUE CONTANT 1 SEC.
(NOT ADDRESS)
; ********** ISAC-S IOM-1 TE MODE REGISTER ************
RFIFO           EQU   08000H  ; RECEIVE FIFO 00-1FH R
XFIFO           EQU   08000H  ; TRANSMIT FIFO 00-1FH W
ISTA            EQU   08020H  ; INTR STATUS REG R
MASK            EQU   08020H  ; MASK REG W
STAR            EQU   08021H  ; STATUS REG R
```

```
CMD     EQU  08021H ; COMMAND REG W
MODE    EQU  08022H ; MODE REG R/W
TIMR    EQU  08023H ; TIMER REG R/W
EXIR    EQU  08024H ; EXTENDED INTR REG R
XAD1    EQU  08024H ; TRANSMIT ADDR 1 W
RBCL    EQU  08025H ; RECEIVE FRAME BYTE COUNT LOW R
XAD2    EQU  08025H ; TRANSMIT ADDR 2 W
SAPR    EQU  08026H ; RECEIVED SAPI REG R
SAP1    EQU  08026H ; SAPI1 W
RSTA    EQU  08027H ; RECEIVE STATUS REG R
SAP2    EQU  08027H ; SAPI2 REG W
TEI1    EQU  08028H ; TEI1 REG W
RHCR    EQU  08029H ; RECEIVE HDLC CTRL REG R
TEI2    EQU  08029H ; TEI2 REG W
RBCH    EQU  0802AH ; RECEIVE FRAME BYTE COUNT HIGH R
STAR2   EQU  0802BH ; STATUS REG 2 R/W
SPCR    EQU  08030H ; SERIAL PORT CONTROL REG R/W
CIRR    EQU  08031H ; COMMAND/INDICATION RECEIVE REG R
CIXR    EQU  08031H ; COMMAND/INDICATION TRANSMIT REG W
MOR     EQU  08032H ; MONITOR RECEIVE REG R
MOX     EQU  08032H ; MONITOR TRANSMIT REG W
SSCR    EQU  08033H ; SIP SIGNALLING CODE RECEIVE R
SSCX    EQU  08033H ; SIP SIGNALLING CODE TRANSMTI W
SFCR    EQU  08034H ; SIP FEATURE CTRL READ R
SFCW    EQU  08034H ; SIP FEATURE CTRL WRITE W
C1R     EQU  08035H ; CHANNEL REG 1 R/W
C2R     EQU  08036H ; CHANNEL REG 2 R/W
B1CR    EQU  08037H ; B1 CHANNEL REG R
STCR    EQU  08037H ; SYNC TRANSFER CTRL REG W
B2CR    EQU  08038H ; B2 CHANNEL REG R
```

```
ADF             EQU     08038H  ; ADDITIONAL FEATURE REG 1 W
ADF2            EQU     08039H  ; ADDITIONAL FEATURE REG 2 R/W
SQRR            EQU     0803BH  ; S,Q CH RECEIVE REG R
SQXR            EQU     0803BH  ; S,Q CH TRANSMIT REG W
; REGISTER R3 USE FOR MESSAGE UNIT
;       #04H    T200 TIME OUT (MESSAGE UNIT)
;       #08H    PROTOCOL ERROR (MESSAGE UNIT)
; REGISTER R5 USE FOR PRIMITIVES
;  LAYER 3 <--> LAYER 2
;       #00H  DL-ESTABLISH REQUEST
;       #01H  DL-ESTABLISH INDICATION
;       #02H  DL-ESTABLISH CONFIRM
;       #03H  DL-RELEASE REQUEST
;       #04H  DL-RELEASE INDICATION
;       #05H  DL-RELEASE CONFIRM
;       #06H  DL-DATA REQUEST           (HAVE MESSAGE UNIT)
;       #07H  DL-DATA INDICATION        (HAVE MESSAGE UNIT)
;       #08H  DL-UNIT DATA REQUEST      (HAVE MESSAGE UNIT)
;       #09H  DL-UNIT DATA INDICATION   (HAVE MESSAGE UNIT)
;  MANAGEMENT ENTITY <--> LAYER 2
;       #10H  MDL-ASSIGN REQUEST        (HAVE MESSAGE UNIT)
;       #11H  MDL-ASSIGN INDICATION     (HAVE MESSAGE UNIT)
;       #12H  MDL-REMOVE REQUEST        (HAVE MESSAGE UNIT)
;       #13H  MDL-ERROR INDICATION      (HAVE MESSAGE UNIT)
;       #14H  MDL-ERROR RESPOND         (HAVE MESSAGE UNIT)
;       #15H  MDL-UNIT DATA REQUEST     (HAVE MESSAGE UNIT)
;       #16H  MDL-UNIT DATA INDICATION  (HAVE MESSAGE UNIT)
;       #17H  MDL-XID REQUEST           (HAVE MESSAGE UNIT)
;       #18H  MDL-XID INDICATION        (HAVE MESSAGE UNIT)
;       #19H  MDL-XID RESPOND           (HAVE MESSAGE UNIT)
```

```
;       #17H  MDL-XID CONFIRM                (HAVE MESSAGE UNIT)
;   LAYER 2 <--> LAYER 1
;       #20H  PH-DATA REQUEST                (HAVE MESSAGE UNIT)
;       #21H  PH-DATA INDICATION             (HAVE MESSAGE UNIT)
;       #22H  PH-ACTIVATE REQUEST
;       #23H  PH-ACTIVATE INDICATION
;       #24H  PH-DEACTIVATE INDICATION
;   MANAGEMENT ENTITY <--> LAYER 1
;       #30H  MPH-ACTIVATE INDICATION
;       #31H  MPH-DEACTIVATE REQUEST
;       #32H  MPH-DEACTIVATE INDICATION
;       #33H  MPH-INFORMATION INDICATION     (HAVE MESSAGE UNIT)
; *************** JUMP ADDRESS DEFINED **********************
                ORG     0000H   ; SET START ADDRESS
                LJMP    START
                ORG     0003H   ; ISAC-S INTERRUPT ROUTINE
                LJMP    ISAC_INTR
                ORG     0013H   ; KEYBOARD INTRRUPT ROUTINE
                LJMP    KBD_INTR
;********************** MAIN *******************************
START:          CLR     IE.7
                MOV     DPTR,#TE_STATUS ; POWER UP
                MOV     A,#00100000B
                MOVX    @DPTR,A
                MOV     DPTR,#KBD_PORT
                MOVX    A,@DPTR         ; RESET KEYBOARD
                LCALL   CLR_KBDBF       ; CLEAR KEYBOARD BUFFER
                LCALL   C_LAST_KEY
                MOV     A,#00H
                MOV     DPTR,#KBD_TIME  ; RESET KBD TIME OUT COUNTER
```

```
MOVX    @DPTR,A
INC     DPTR
MOVX    @DPTR,A
INC     DPTR
MOVX    @DPTR,A          ; COUNTER TOTAL 3 BYTE
MOV     R2,#20H
LCALL   DELAY            ; LCD INITIALIZE
MOV     P1,#5FH          ; CLEAR PORT1
MOV     A,#01H           ; CLEAR
LCALL   LCDWI
MOV     A,#00111000B     ; INITIALIZE LCD FUNCTION SET
LCALL   LCDWI
MOV     A,#00001100B     ; DISPLAY ON
LCALL   LCDWI
MOV     A,#00000110B     ; ENTRY MODE SET
LCALL   LCDWI
MOV     A,#02H           ; HOME
LCALL   LCDWI
MOV     DPTR,#CLEAR      ; CLEAR SCREEN
MOV     R1,#00H
LCALL   LCD_DISPC
MOV     DPTR,#CLEAR
MOV     R1,#01H
LCALL   LCD_DISPC
MOV     PSW,#00H
SETB    TCON.2           ; SET EDGE DETECT OF INT1
CLR     TCON.0           ; SET LEVEL DETECT OF INT0
SETB    IE.0
CLR     IE.2             ; CLEAR KBD INTR.
SETB    IE.7             ; ENABLE INTERRUPT
```

```
                MOV     R3,#00H
                LCALL   INIT_2085
                MOV     A,#00H
                MOV     DPTR,#SAP1
                MOVX    @DPTR,A
                MOV     DPTR,#SAP2
                MOVX    @DPTR,A
                MOV     DPTR,#TEI1
                MOVX    @DPTR,A
                MOV     DPTR,#TEI2
                MOVX    @DPTR,A
; ********************************************************
; *            INITIALIZE COMPLETE                       *
; *    GO IN STANDBY MODE    BEGIN PROGRAM               *
; ********************************************************
PWR_DW:         LCALL   POWER_DW        ; SET POWER DOWN MODE
READ_CYCLE:     SETB    IE.2            ; ENABLE KBD INTR.
P_LP:           MOV     DPTR,#GET_KEY
                MOVX    A,@DPTR
                CJNE    A,#0EH,CHK_PRIMITIVE
                LCALL   CLR_KBDBF
                MOV     A,#00H
                MOV     DPTR,#KBD_TIME  ; RESET KBD TIME OUT COUNTER
                MOVX    @DPTR,A
                INC     DPTR
                MOVX    @DPTR,A
                INC     DPTR
                MOVX    @DPTR,A         ; COUNTER TOTAL 3 BYTE
                MOV     DPTR,#TE_STATUS ; INDICATE POWER UP MODE
                MOVX    A,@DPTR
```

```
                SETB    ACC.5
                MOVX    @DPTR,A
                LJMP    MAIN2
CHK_PRIMITIVE:  CJNE    R5,#24H,P_LP
                LJMP    PWR_DW
MAIN2:          MOV     DPTR,#MESS1
                MOV     R1,#00H
                LCALL   LCD_DISPC
                MOV     PTR,#CLEAR
                MOV     R1,#01H
                LCALL   LCD_DISPC
                MOV     A,#00001100B  ; DISPLAY ON
                LCALL   LCDWI
                MOV     R2,#0FFH
                LCALL   INKEY
                CJNE    R0,#01H,PWR_DW
                LCALL   L3_HANDLE
                LJMP    READ_CYCLE
; ****************** LAYER 3 HANDLE PROCEDURE ********************
L3_HANDLE:  MOV     R5,#00H       ; DL-ESTABLISH REQUEST
                LCALL   L2_HANDLE
                RET
; *************** LAYER 2 HANDLE PROCEDURE ******************
; PARAMETER R5 (PRIMITIVES);
L2_HANDLE:  MOV     R5,#22H ;PH-ACTIVATE REQUEST
                LCALL   L1_HANDLE
                RET
; *************** LAYER 1 HANDLE PRODECURE ******************
; PARAMETER R5 (PRIMITIVES);
L1_HANDLE:  MOV     R5,#23H       ; PH-ACTIVATE INDICATION
```

```
                MOV     DPTR,#TE_STATUS
                MOVX    A,@DPTR
                JB      ACC.6,STILL_ACT
                LCALL   ACTIVATE_L1
STILL_ACT:      RET
; ********* RESET TIMER T202 ************
RESET_T202:     MOV      A,#00H
                MOV     DPTR,#L2_TIME   ; RESET LAYER 1 TIME OUT COUNTER
                MOVX    @DPTR,A
                INC     DPTR
                MOVX    @DPTR,A
                INC     DPTR
                MOVX    @DPTR,A         ; COUNTER TOTAL 3 BYTE
                RET
; ********************* RESET T3 (I.430) ********************
; PARAMETER
; VAR A,DPTR
RESET_T3:       MOV      A,#00H
                MOV     DPTR,#L1_TIME   ; RESET LAYER 1 TIME OUT COUNTER
                MOVX    @DPTR,A
                INC     DPTR
                MOVX    @DPTR,A
                INC     DPTR
                MOVX    @DPTR,A         ; COUNTER TOTAL 3 BYTE
                RET
; ****************** INKEY PROCEDURE **********************
; PARAMETER R1 (DEFINE LENGHT) OR FFH (UNTIL ENTER PRESS)
; RETURN VALUE IN KBD_BUFFER, R0
;       00H     POWER DOWN MODE
;       01H     INKEY VALID
```

```
; VAR R2
INKEY:      MOV     A,R2
            PUSH    ACC
INKEY1:     CJNE    R2,#0FFH,CHK_LEN
            LJMP    NO_LEN
CHK_LEN:    MOV     DPTR,#KBD_PTR
            MOVX    A,@DPTR
            CLR     C
            SUBB    A,R2
            JZ      LEN_VALID
            LJMP    NO_LEN
LEN_VALID:  MOV     R0,#01H
            LJMP    END_KEY
NO_LEN:     MOV     DPTR,#GET_KEY
            MOVX    A,@DPTR
            MOV     R0,A
            CLR     C
            SUBB    A,#1EH          ; POWER DOWN KEY PRESSED
            MOV     R0,#0
            JZ      END_KEY
            MOV     DPTR,#TE_STATUS
            MOVX    A,@DPTR         ; IF CHANNEL IN USED THEN
            ANL     A,#00000111B    ;  NOT INCREASE COUNTER
            JNZ     NOT_CARRY
            MOV     DPTR,#KBD_TIME
            MOVX    A,@DPTR         ; DEC COUNTER 1st BYTE
            DEC     A
            MOVX    @DPTR,A
            JNZ     NOT_CARRY
            INC     DPTR            ; CARRY TO 2nd COUNTER BYTE
```

```
                MOVX    A,@DPTR
                DEC     A
                MOVX    @DPTR,A
                JNZ     NOT_CARRY
                INC     DPTR            ; CARRY TO 3rd COUNTER BYTE
                MOVX    A,@DPTR
                INC     A
                MOVX    @DPTR,A
                CJNE    A,#TIME_OUT_CNT,NOT_CARRY
                MOV     R0,#00H
                LJMP    END_KEY
NOT_CARRY:      MOV     DPTR,#GET_KEY
                MOVX    A,@DPTR
                CJNE    A,#0DH,NOT_CARRY1  ; WAIT FOR STRING AND ENTER
 PRESS
                MOV     R0,#01H
                LJMP    END_KEY
NOT_CARRY1:     CJNE    A,#0FH,INKEY1
              ; RESERVE FOR FUTURE USE
                LJMP    INKEY1
END_KEY:        POP     ACC
                MOV     R2,A
                RET
; ********* ACTIVATE LAYER 1 SUBROUTINE ********************
; PARAMETER
;       RETURN PRIMITIVE IN R5
;       23H PH-ACTIVATE INDICATION
;       24H PH-DEACTIVATE INDICATION
ACTIVATE_L1:    CLR     IE.2
                LCALL   RESET_T3
```

```
                LCALL   ENACLK_SBC
                MOV     R5,#00H
                MOV     R3,#00H
                MOV     A,#01100000B        ; SEND AR8 COMMAND
                MOV     DPTR,#CIXR
                MOVX    @DPTR,A
WAIT_INT:       MOV     DPTR,#L1_TIME
                MOVX    A,@DPTR             ; DEC COUNTER 1st BYTE
                DEC     A
                MOVX    @DPTR,A
                JNZ     STATE_ACT
                INC     DPTR                ; CARRY TO 2nd COUNTER BYTE
                MOVX    A,@DPTR
                DEC     A
                MOVX    @DPTR,A
                JNZ     STATE_ACT
                INC     DPTR                ;CARRY TO 3rd COUNTER BYTE
                MOVX    A,@DPTR
                INC     A
                MOVX    @DPTR,A
                CJNE    A,#TIME_OUT_CNT,STATE_ACT ; IF ACT L1 FAIL FOR 20
SEC  INDICATE TIME OUT ERROR.
                CJNE    R5,#01H,RESET_SBC
                LJMP    SEND_DIU
RESET_SBC:      MOV     A,#01000100B        ; SEND RESET COMMAND
                MOV     DPTR,#CIXR
                MOVX    @DPTR,A
                CJNE    R3,#00000110B,$     ; WAIT EI INDICATION
                LJMP    SEND_DIU
STATE_ACT:      CJNE    R5,#23H,WAIT_INT  ; IF STATE 07H THEN
```

```
                MOV     DPTR,#TE_STATUS
                MOVX    A,@DPTR
                SETB    ACC.6
                MOVX    @DPTR,A
                LJMP    RETURN_L1
SEND_DIU:       MOV     A,#01111100B        ; SEND DIU COMMAND
                MOV     DPTR,#CIXR
                MOVX    @DPTR,A
                CJNE    R3,#0FH,$           ; WAIT FOR DID INDICATE
                MOV     R5,#24H             ; PH-DEACTIVATE INDICATION
                MOV     DPTR,#TE_STATUS
                MOVX    A,@DPTR
                CLR     ACC.6
                MOVX    @DPTR,A
RETURN_L1:      SETB    IE.2
                RET
; ****************** ENABLE CLOCK **************
; PARAMETER
; VAR A,DPTR
ENACLK_SBC:     MOV     R3,#00H
                MOV     A,#10000000B
                MOV     DPTR,#SPCR
                MOVX    @DPTR,A
                MOV     A,#01000000B    ; SEND TIM COMMAND
                MOV     DPTR,#CIXR
                MOVX    @DPTR,A
                CJNE    R3,#00000111B,$ ; WAIT PU INDICATION
                MOV     A,#00H
                MOV     DPTR,#SPCR      ; RESET SPU BIT
                MOVX    @DPTR,A
```

```
                RET
; ************** INITIALIZE 2085 **************
; PARAMETER
; VAR A,DPTR
INIT_2085:      MOV     A,#0FFH
                MOV     DPTR,#MASK
                MOVX    @DPTR,A
                MOV     A,#00H
                MOV     DPTR,#ADF2
                MOVX    @DPTR,A         ; SELECT IOM-1 MODE
                MOV     A,#00111001B    ; SET AUTO MODE WITH 2 BYTE ADDR
                MOV     DPTR,#MODE      ; FIELDS INTERNAL TIMER ACTIVE HDLC
                MOVX    @DPTR,A         ; RECEIVER IN TE MODE
                MOV     A,#10100000B    ; SET ISAC TO RETRY 5 TIMES
                MOV     DPTR,#TIMR
                MOVX    @DPTR,A
                MOV     A,#01110000B    ; NOT USE SPECIAL FN REG
                MOV     DPTR,#STCR
                MOVX    @DPTR,A
                MOV     A,#00000000B
                MOV     DPTR,#ADF1
                MOVX    @DPTR,A
                MOV     R5,#00H         ; POWER DOWN STATE
                MOV     R3,#00H
                MOV     A,#00H
                MOV     DPTR,#MASK
                MOVX    @DPTR,A
                LCALL   ENACLK_SBC
                MOV     A,#01000100B    ; SEND RESET COMMAND
                MOV     DPTR,#CIXR
```

```
                MOVX    @DPTR,A
                CJNE    R3,#00000110B,$     ; WAIT EI INDICATE
                MOV     A,#01111100B        ; SEND DIU COMMAND
                MOV     DPTR,#CIXR
                MOVX    @DPTR,A
                CJNE    R3,#00001111B,$     ; WAIT FOR DID INDICATION
                RET
; ************** POWER DOWN ****************
; PARAMETER
; VAR DPTR,A,R1,R2
POWER_DW:       MOV     DPTR,#TE_STATUS  ; INDICATE POWER DOWN MODE
                MOVX    A,@DPTR
                CLR     ACC.5
                MOVX    @DPTR,A
                MOV     DPTR,#KBD_PORT
                MOVX    A,@DPTR
                MOV     DPTR,#GET_KEY
                MOV     A,#00H
                MOVX    @DPTR,A
                MOV     A,#00001011B    ; DISPLAY OFF
                LCALL   LCDWI
                RET
; ****************** DELAY SUB ******************
; PARAMETER=R2
; VAR=R2,R3,R4
DELAY:          MOV     A,R3
                PUSH    ACC
                MOV     A,R4
                PUSH    ACC
DELAY1:         MOV     R3,#0   ; DELAY 65.536 mS FOR EACH R2
```

```
DELAY       MOV     R4,#0
            DJNZ    R4,$
            DJNZ    R3,DELAY2
            DJNZ    R2,DELAY1
            POP     ACC
            MOV     R4,A
            POP     ACC
            MOV     R3,A
            RET
; ************ LCD DISPLAY IN CODE SEGMENT *****************
; PARAMETER=DPTR OF START BLOCK 20 BYTES IN PROGRAME MEMORY
;       R1=00H DISPLAY LINE 1, R1=01H DISPLAY LINE 2
; VAR=A, R0, R1, R7, DPTR
LCD_DISPC:  MOV     A,R0
            PUSH    ACC
            MOV     A,R7
            PUSH    ACC
            CJNE    R1,#00H,LCDLDS2
            MOV     A,#80H      ;SET LINE 1 LCD ADDRESS
            LJMP    LCDLDS3
LCDLDS2:    MOV     A,#0C0H
LCDLDS3:    MOV     R1,A
            PUSH    DPH
            PUSH    DPL
            LCALL   LCDWI
            POP     DPL
            POP     DPH
            MOV     R7,#20      ; 20 CHARS.
            MOV     R0,#0
LCDLDS1:    MOV     A,R0
```

```
                MOVC    A,@A+DPTR
                PUSH    DPH
                PUSH    DPL
                LCALL   LCDWD
                INC     R1
                MOV     A,R1
                LCALL   LCDWI
                POP     DPL
                POP     DPH
                INC     R0
                DJNZ    R7,LCDLDS1
                POP     ACC
                MOV     R7,A
                POP     ACC
                MOV     R0,A
                RET
; ********** LCD DISPLAY IN DATA SEGMENT **************
; PARAMETER=DPTR OF START BLOCK 20 BYTES IN DATA MEMORY
;       R1=00H DISPLAY LINE 1, R1=01H DISPLAY LINE 2
; VAR=A, R0, R1, R7, DPTR
LCD_DISPX:      MOV     A,R0
                PUSH    ACC
                MOV     A,R7
                PUSH    ACC
                CJNE    R1,#00H,LCDLDX2
                MOV     A,#80H          ; SET LINE 1 LCD ADDRESS
                LJMP    LCDLDX3
LCDLDX2:        MOV     A,#0C0H
LCDLDX3:        MOV     R1,A            ; PUT LCD ADDRESS TO R1
                PUSH    DPH
```

```
                PUSH    DPL
                LCALL   LCDWI
                POP     DPL
                POP     DPH
                MOV     R7,#20          ; 20 CHARS.
LCDLDX1:        MOVX    A,@DPTR
                PUSH    DPH
                PUSH    DPL
                LCALL   LCDWD
                INC     R1
                MOV     A,R1
                LCALL   LCDWI
                POP     DPL
                POP     DPH
                INC     DPTR
                DJNZ    R7,LCDLDX1
                POP     ACC
                MOV     R7,A
                POP     ACC
                MOV     R0,A
                RET
; ***************** LCDWI ************
; LCD WRITE INSTRUCTION
; PARAMETER=A
; VAR=A, DPTR
LCDWI:          MOV     DPTR,#LCD_PORT
                CLR     P1.0            ;RS=0
                CLR     P1.1
                CLR     P1.2
                MOVX    @DPTR,A
```

```
                SETB    P1.2
                NOP
                CLR     P1.2
                MOV     A,#0
LCDWI1:         DEC     A
                JNZ     LCDWI1
                SETB    P1.1
                RET
; ****************** LCDWD ***************
; LCD WRITE DATA
; PARAMETER=A
; VAR=A, DPTR
LCDWD:          MOV     DPTR,#LCD_PORT
                SETB    P1.0        ; RS=1
                CLR     P1.1
                CLR     P1.2
                MOVX    @DPTR,A
                SETB    P1.2
                NOP
                CLR     P1.2
                MOV     A,#0
LCDWD1:         DEC     A           ; WAIT FOR LCD WORKING
                JNZ     LCDWD1
                SETB    P1.1
                RET
; ******************** KBD HANDLE **********************
; PARAMETER
; VAR A,B,R0,R2,R7,DPTR
; RETURN DATA IN KEYBOARD BUFFER AND COMMAND IN GET_KEY
; BY THESE VALUE
```

```
;       00H   NO COMMAND (KBD INTR. HANDLING ITSELF)
;       0DH   VOICE-ENTER KEY PRESSED
;       0EH   POWER UP AND SET POWER STATUS
;       1EH   POWER DOWN AND CLEAR POWER STATUS
;       0FH   RESERVE
KBD_INTR:   CLR     IE.7            ; DISABLE ALL INTR.
            PUSH    DPH
            PUSH    DPL
            PUSH    ACC
            MOV     A,R2
            PUSH    ACC
            MOV     A,R7
            PUSH    ACC
            MOV     A,#00H
            MOV     DPTR,#KBD_TIME  ; RESET KBD TIME OUT COUNTER
            MOVX    @DPTR,A
            INC     DPTR
            MOVX    @DPTR,A
            INC     DPTR
            MOVX    @DPTR,A         ; COUNTER TOTAL 3 BYTE
            MOV     DPTR,#KBD_PORT  ; GET DATA FROM KEYBOARD
            MOVX    A,@DPTR
            MOV     R7,A
            ANL     A,#11110000B
            CJNE    A,#11110000B,LOW_KEY
            MOV     A,R7
            ANL     A,#0FH
            MOV     B,#02H
            DIV     AB
            ADD     A,#08H
```

```
                JMP     LOOK_UP
LOW_KEY:        MOV     A,R7
                RL      A
                RL      A
                RL      A
                RL      A
                ANL     A,#0FH
LOOK_UP:        MOV     DPTR,#SCAN_CODE
                MOVC    A,@A+DPTR
                MOV     R7,A                    ; R7 IS SCAN CODE
                CJNE    R7,#'A',NEXT1           ; POWER UP/DOWN
                MOV     DPTR,#TE_STATUS
                MOVX    A,@DPTR
                JNB     ACC.5,P_UP
                MOV     A,#1EH          ; POWER DOWN COMMAND
                MOV     DPTR,#GET_KEY
                MOVX    @DPTR,A
                MOV     DPTR,#TE_STATUS
                MOVX    A,@DPTR
                CLR     ACC.5           ; CLEAR POWER STATUS
                MOVX    @DPTR,A
                LJMP    EXIT1
P_UP:           MOV     A,#0EH          ; POWER UP COMMAND
                MOV     DPTR,#GET_KEY
                MOVX    @DPTR,A
                MOV     DPTR,#TE_STATUS
                MOVX    A,@DPTR
                SETB    ACC.5                   ; SET POWER STATUS
                MOVX    @DPTR,A
                LJMP    EXIT1
```

```
NEXT1:      MOV     DPTR,#TE_STATUS  ; CHECK POWER STATUS
            MOVX    A,@DPTR
            JNB     ACC.5,TEMP_EXIT  ; EXIT IF POWER DOWN
            LJMP    NEXT1_EXT        ; CJNE INDICATE ERROR 'DESTINATION
TEMP_EXIT:  LJMP    EXIT1 ;OUT OF RANGE' SO USE LJMP INSTEAD
NEXT1_EXT:  CJNE    R7,#'E',NEXT2    ; VOICE-ENTER KEY PRESSED
            LCALL   S_LAST_KEY
            MOV     DPTR,#GET_KEY
            MOV     A,#0DH           ; INDICATE VOICE-ENTER KEY
            MOVX    @DPTR,A
            LJMP    EXIT
NEXT2:      CJNE    R7,#'L',NEXT3    ; CLEAR KBD BUFFER
            LCALL   CLR_KBDBF
            MOV     DPTR,#GET_KEY
            MOV     A,#00H           ; INDICATE NO COMMAND
            MOVX    @DPTR,A
            LJMP    EXIT
NEXT3:      CJNE    R7,#'*',NEXT4    ; RECALL
            LCALL   R_LAST_KEY
            MOV     DPTR,#GET_KEY
            MOV     A,#00H           ; INDICATE NO COMMAND
            MOVX    @DPTR,A
            LJMP    EXIT
NEXT4:      CJNE    R7,#'#',NEXT5    ; CLEAR LAST KEY
            LCALL   C_LAST_KEY
            MOV     DPTR,#GET_KEY
            MOV     A,#00H           ; INDICATE NO COMMAND
            MOVX    @DPTR,A
            LJMP    EXIT1
NEXT5:      CJNE    R7,#'D',NEXT6    ; DATA-ENTER KEY PRESSED
```

```
                LCALL   S_LAST_KEY
                MOV     DPTR,#GET_KEY
                MOV     A,#0FH              ; INDICATE DATA-ENTER KEY
                MOVX    @DPTR,A
                LJMP    EXIT
NEXT6:          MOV     DPTR,#KBD_PTR       ; CHECK BUFFER IS FULL?
                MOVX    A,@DPTR
                CJNE    A,#20,KBD_EXCP
KBD_FULL:       MOV     DPTR,#KBD_ERR
                MOV     R1,#01H
                LCALL   LCD_DISPC
                MOV     R2,#10
                LCALL   DELAY
                MOV     R1,#01H
                MOV     DPTR,#KBD_BUFFER
                LCALL   LCD_DISPC
                LJMP    EXIT
KBD_EXCP:       MOV     DPTR,#KBD_PTR
                MOVX    A,@DPTR
                INC     A
                MOVX    @DPTR,A             ; INCREMENT KBD_PTR
                DEC     A                   ; BECAUSE START AT LOC. 0
                MOV     DPTR,#KBD_BUFFER
BUFFER_LP:      JZ      EXIT_LP             ; SEARCH FOR CHAR. LOCATION
                DEC     A
                INC     DPTR
                LJMP    BUFFER_LP
EXIT_LP:        MOV     A,R7                ; PUT SCAN CODE TO BUFFER
                MOVX    @DPTR,A
                MOV     DPTR,#GET_KEY
```

```
                MOV     A,#00H              ; INDICATE NO COMMAND
                MOVX    @DPTR,A
EXIT:           MOV     R1,#01H
                MOV     DPTR,#KBD_BUFFER
                LCALL   LCD_DISPX
EXIT1:          POP     ACC
                MOV     R7,A
                POP     ACC
                MOV     R2,A
                POP     ACC
                POP     DPL
                POP     DPH
                SETB    IE.7            ; ENABLE INTR.
                RETI
; ********** CLEAR KEYBOARD BUFFER *********
; PARAMETER
; VAR DPTR,A,R7,R1
CLR_KBDBF:      MOV     A,R7
                PUSH    ACC
                MOV     A,R1
                PUSH    ACC
                MOV     DPTR,#KBD_PTR
                MOV     A,#00H
                MOVX    @DPTR,A
                MOV     DPTR,#GET_KEY
                MOVX    @DPTR,A
                MOV     R7,#20
                MOV     DPTR,#KBD_BUFFER
                MOV     A,#' '
CLR_KBD:        MOVX    @DPTR,A
```

```
                INC     DPTR
                DJNZ    R7,CLR_KBD
                MOV     R1,#01H
                MOV     DPTR,#KBD_BUFFER
                LCALL   LCD_DISPX
                POP     ACC
                MOV     R1,A
                POP     ACC
                MOV     R7,A
                RET
; ************** SAVE KBD BUFFER AS LAST KEY *******
; SAVE KBD BUFFER LIKE LAST KEY AS BACK TO FRONT
; PARAMETER
; VAR DPTR,A,R0
S_LAST_KEY:     MOV     A,R0
                PUSH    ACC
                MOV     R0,#21
                MOV     DPTR,#KBD_PTR
LAST_LP1:       MOVX    A,@DPTR
                PUSH    ACC
                INC     DPTR
                DJNZ    R0,LAST_LP1
                MOV     R0,#21
                MOV     DPTR,#LAST_CALL
LAST_LP2:       POP     ACC
                MOVX    @DPTR,A
                INC     DPTR
                DJNZ    R0,LAST_LP2
                POP     ACC
                MOV     R0,A
```

```
                RET
; ******** RESTORE LAST KEY TO KBD BUFFER *******
; PARAMETER
; VAR DPTR,R0,A
R_LAST_KEY:  MOV     A,R0
             PUSH    ACC
             MOV     R0,#21
             MOV     DPTR,#LAST_CALL
LAST_LP3:    MOVX    A,@DPTR
             PUSH    ACC
             INC     DPTR
             DJNZ    R0,LAST_LP3
             MOV     R0,#21
             MOV     DPTR,#KBD_PTR
LAST_LP4:    POP     ACC
             MOVX    @DPTR,A
             INC     DPTR
             DJNZ    R0,LAST_LP4
             POP     ACC
             MOV     R0,A
             RET
; ********* CLEAR LAST KEY ********
; PARAMETER
; VAR DPTR,A,R0
C_LAST_KEY:  MOV     A,R0
             PUSH    ACC
             MOV     DPTR,#LAST_CALL
             MOV     R0,#20
             MOV     A,#' '
LAST_LP5:    MOVX    @DPTR,A
```

```
                INC     DPTR
                DJNZ    R0,LAST_LP5
                MOV     A,#00H
                MOVX    @DPTR,A
                POP     ACC
                MOV     R0,A
                RET
SCAN_CODE:  DB      '#E0*D987A654L321'
KBD_ERR:    DB      'Keyboard Buffer full'
; ***************** CHECK RECEIVE FROM RME INTR ***************
; RETURN VALUE IN R3
;       11H     I-FRAME MUST READ RFIFO AT OUT OF THIS ROUTINE
;       12H     ID CHECK REQUEST
;       13H     ID REMOVE
;       14H     DM DISCONNECT MODE
;       15H     UA UNNUMBERED ACKNOWLEDGE
;       16H     FRMR    FRAME REJECT
;       17H     SABME
;       18H     DISC
;       19H     TEI ASSIGN
;       1AH     RX DATA OVERFLOW
;       1BH     CRC ERROR
;       1CH     RX MESSAGE ABORT
;       F1H     UI FRAME INVALID
;       F2H     ID DENIED
; VAR A,R0,R1,DPTR
CHECK_FRAME: MOV    A,R0
                PUSH    ACC
                MOV     DPTR,#RSTA
                MOVX    A,@DPTR
```

```
                MOV     R0,A
                ANL     A,#01000000B            ; RX DATA OVERFLOW
                JZ      CRC_CHECK
                MOV     R3,01AH
CRC_CHECK:      MOV     A,R0
                ANL     A,#0010000B
                JZ      RAB_CHECK
                MOV     R3,#01BH                ; CRC ERROR
RAB_CHECK:      MOV     A,R0                    ; RX MESSAGE ABORT
                ANL     A,#0001000B
                JZ      CHECK_NEXT
                MOV     R3,#01CH
CHECK_NEXT:     MOV      DPTR,#RHCR
                MOVX    A,@DPTR
                CJNE    A,#00000011B,NOT_UI     ; INDICATE UI FRAME
                LJMP    UI_FRAME
NOT_UI:         LJMP    NOT_UI_FRAME
UI_FRAME:       MOV     DPTR,#RFIFO
                MOVX    A,@DPTR
                CJNE    A,#0FH,UI_ERROR
                INC
DPTR            ; CHECK Ri
                MOVX     A,@DPTR
                MOV      R0,A
                INC      DPTR
                MOVX     A,@DPTR
                MOV      R1,A
                PUSH     DPH
                PUSH     DPL
                ;MOV     DPTR,#RI1
```

```
                MOVX    A,@DPTR
                SUBB    A,R0
                JNZ     NOT_RI   ; IF NOT THIS Ri
                ;MOV    DPTR,#RI2
                MOVX    A,@DPTR
                SUBB    A,R1
                JNZ     NOT_RI
                LJMP    RI_VALID
UI_ERROR:       MOV     R3,#0F1H
                LJMP    UI_COMPLETE
NOT_RI:         MOV     A,R0
                ORL     A,R1
                JZ      RI_VALID_0
                LJMP    UI_COMPLETE
RI_VALID_0:     POP     DPL             ; Ri=0 INDICATE ID REMOVE OR
                POP     DPH             ;  ID CHECK REQUEST COMMAND
                INC     DPTR
                MOVX    A,@DPTR
                CJNE    A,#00000100B,ID_REMOVE
                INC     DPTR            ; INDICATE ID CHECK REQUEST
                MOVX    A,@DPTR
                MOV     R0,A
                MOV     DPTR,#TE_TEI
                MOVX    A,@DPTR
                SUBB    A,R0
                JNZ     UI_COMPLETE  ; IF TEI NOT IDENTIFY THEN
                MOV     R3,#12H         ;  NO RESPOND
                LJMP    UI_COMPLETE
ID_REMOVE:      CJNE    A,#00000110B,UI_COMPLETE
                INC     DPTR
```

```
                MOVX    A,@DPTR
                MOV     R0,A
                CJNE    A,#0FFH,SPEC_TEI  ; IF TEI=127 THEN NETWORK
                LJMP    REMOVED_TEI       ; NEED ALL TE TO REMOVE TEI
SPEC_TEI:       MOV     DPTR,#TE_TEI      ; IF REMOVED TEI SPECIFIED THEN
                MOVX    A,@DPTR           ;  COMPARE TO THIS TEI
                SUBB    A,R0
                JNZ     UI_COMPLETE
REMOVED_TEI:    MOV     R3,#13H           ;  REMOVE TEI
                MOV     DPTR,#TE_TEI
                MOV     A,#00H
                MOVX    @DPTR,A
                MOV     DPTR,#TEI1
                MOVX    @DPTR,A
                MOV     DPTR,#TEI2
                MOVX    @DPTR,A
                LJMP    UI_COMPLETE
RI_VALID:       POP     DPL
                POP     DPH
                INC     DPTR
                MOVX    A,@DPTR
                CJNE    A,#00000010B,ID_DENIED ; ID ASSIGN FROM NETWORK
                INC     DPTR
                MOVX    A,@DPTR
                MOV     DPTR,#TE_TEI
                MOVX    @DPTR,A                ; TEI ASSIGNED
                MOV     DPTR,#TEI1
                MOVX    @DPTR,A
                MOV     DPTR,#TEI2
                MOVX    @DPTR,A
```

```
                MOV     DPTR,#TE_TEI
                MOVX    @DPTR,A
                MOV     R3,#19H
                LJMP    RME_COMPLETE
ID_DENIED:      CJNE    A,#00000011B,UI_COMPLETE
                MOV     R3,#0F2H
UI_COMPLETE:    LJMP    RME_COMPLETE
NOT_UI_FRAME:   CJNE    A,#00011111B,UA_CHK     ; DM INDICATE
                MOV     R3,#14H
                LJMP    RME_COMPLETE
UA_CHK:         CJNE    A,#01110011B,FRMR_CHK   ; UA INDICATE
                MOV     R3,#15H
                LJMP    RME_COMPLETE
FRMR_CHK:       CJNE    A,#10010111B,SABME_CHK  ; FRAME REJECT INDICATE
                MOV     R3,#16H
                LJMP    RME_COMPLETE
SABME_CHK:      CJNE    A,#01111111B,DISC_CHK   ; SABME INDICATE
                MOV     R3,#17H
                LJMP    RME_COMPLETE
DISC_CHK:       CJNE    A,#01111111B,I_FRAME    ; DISC INDICATE
                MOV     R3,#18H
                LJMP    RME_COMPLETE
I_FRAME:        MOV     R3,#11H                 ; I FRAME INDICATE
                LJMP    RETURN_RME              ; RET COMMAND
RME_COMPLETE:   MOV     DPTR,#CMDR
                MOV     A,#10000000B
                MOVX    @DPTR,A
RETURN_RME:     RET
;  ************** ISAC-S INTERRUPT HANDLE ****************
; RETURN INDICATION IN R3
```

```
;       00H     RME INTR
;       01H     RX POOL INTR
;       02H     RX STATUS CHANGE
;       03H     TX POOL READY
;       04H     T200 TIME OUT (MESSAGE UNIT)
;       05H     SYNC TRANSFER INTR
;       06H     TX MESSAGE REPEAT
;       07H     TX DATA UNDERRUN
;       08H     PROTOCOL ERROR (MESSAGE UNIT)
;       09H     RECEIVE FRAME OVERFLOW
;       0AH     SYNC TRANSFER OVERFLOW
;       0BH     MONITOR STATUS
;       0CH     SUBSCRIBER AWAKE
;       0DH     WATCHDOG TIMER OVERFLOW
; VAR A,R3,R0,R5
ISAC_INTR:      CLR     IE.7
                PUSH    ACC
                MOV     A,R0
                PUSH    ACC
                MOV     DPTR,#ISTA      ; GET INTERRUPT FUNCTION
                MOVX    A,@DPTR
                MOV     R0,A            ; SAVE INTERRUPT FN TO R0
                ANL     A,#00000100B    ; CHECK CISQ INTERRUPT
                JZ      RME_INTR
                LJMP    CISQ_INT
RME_INTR:       MOV     A,R0
                ANL     A,#10000000B    ; RECEIVE MESSAGE END INTR.
                JZ      RPF_INTR
                MOV     R3,#00H
                LCALL   CHECK_FRAME
```

```
                LJMP    INTR_COMPLETE
RPF_INTR:       MOV     A,R0
                ANL     A,#01000000B        ; RECEIVE POOL FULL INTR.
                JZ      RSC_INTR
                MOV     R3,#01H
                LJMP    INTR_COMPLETE
RSC_INTR:       MOV     A,R0
                ANL     A,#00100000B        ; RECEIVE STATUS CHANGE INTR.
                JZ      XPR_INTR
                MOV     R3,#02H
                LJMP    INTR_COMPLETE
XPR_INTR:       MOV     A,RO
                ANL     A,#00010000B        ; TRANSMITT POOL READY INTR.
                JZ      TIN_INTR
                MOV     R3,#03H
                LJMP    INTR_COMPLETE
TIN_INTR:       MOV     A,R0
                ANL     A,#00001000B    ; CHECK TIMER INTERRUPT
                JZ      SIN_INTR
                MOV     R3,#04H
                LJMP    INTR_COMPLETE
SIN_INTR:       MOV     A,R0
                ANL     A,#00000010B    ; SYNCHRONOUS TRANSFER INTRRUPT
                JZ      EXI_INTR
                MOV     R3,#05H
                LJMP    INTR_COMPLETE
EXI_INTR:       MOV     A,R0
                ANL     A,#00000001B    ; EXTENDED INTR.
                JZ      INTR_CMPT
                MOV     DPTR,#EXIR
```

```
                MOVX  A,@DPTR
                LJMP  XMR_INTR
INTR_CMPT:      LJMP  INTR_COMPLETE
XMR_INTR:       MOV   R0,A
                ANL   A,#10000000B   ; TRANSMIT MESSAGE REPEAT
                JZ    XDU_INTR
                MOV   R3,#06H
                LJMP  INTR_COMPLETE
XDU_INTR:       MOV   A,R0
                ANL   A,#01000000B   ; TRANSMIT DATA UNDERRUN
                JZ    PCE_INTR
                MOV   R3,#07H
                LJMP  INTR_COMPLETE
PCE_INTR:       MOV   A,R0
                ANL   A,#00100000B   ; PROTOCOL ERROR
                JZ    RFO_INTR
                MOV   R3,#08H
                LJMP  INTR_COMPLETE
RFO_INTR:       MOV   A,R0
                ANL   A,#00010000B   ; RECEIVE FRAME OVERFLOW
                JZ    SOV_INTR
                MOV   R3,#09H
                LJMP  INTR_COMPLETE
SOV_INTR:       MOV   A,R0
                ANL   A,#00001000B   ; SYNCHRONOUS TRANSFER OVERFLOW
                JZ    MOS_INTR
                MOV   R3,#0AH
                LJMP  INTR_COMPLETE
MOS_INTR:       MOV   A,R0
                ANL   A,#00000100B   ; MONITOR STATUS
```

```
                JZ      SAW_INTR
                MOV     R3,#0BH
                LJMP    INTR_COMPLETE
SAW_INTR:       MOV     A,R0
                ANL     A,#00000010B    ; SUBSCRIBER AWAKE (NOT USED)
                JZ      WOV_INTR
                MOV     R3,#0CH
                LJMP    INTR_COMPLETE
WOV_INTR:       MOV     A,R0
                ANL     A,#00000001B    ; WATCH DOG TIMER OVERFLOW
(NOT USED)
                JZ      INTR_COMPLETE
                MOV     R3,#0DH
INTR_COMPLETE:  POP     ACC
                MOV     R0,A
                POP     ACC
                SETB    IE.7
                RETI
; ********** CISQ INTERRUPT HANDLE **********
CISQ_INT:       LCALL   RESET_T3
                MOV     DPTR,#CIRR
                MOVX    A,@DPTR
                RR      A
                RR      A
                ANL     A,#0FH
                MOV     R3,A
                MOV     DPTR,#SQRR
                MOVX    A,@DPTR
                CJNE    R3,#00000111B,DIS       ; PU INDICATE
                MOV     DPTR,#MESS8
```

```
                MOV     R1,#00H
                LCALL   LCD_DISPC
                MOV     R5,#01H                 ; CHANGE TO PEND.ACT. STATE
                LJMP    INTR_COMPLETE
DIS:            CJNE    R3,#00000011B,RSYD      ; DIS INDICATE
                MOV     DPTR,#MESS5
                MOV     R1,#00H
                LCALL   LCD_DISPC
                MOV     DPTR,#MESS6
                MOV     R1,#01H
                LCALL   LCD_DISPC
                MOV     R5,#06H
                LJMP    INTR_COMPLETE           ; MUST RETURN CODE IN R3
RSYD:           CJNE    R3,#00000100B,ARD       ; RSYD INDICAATE
                CJNE    R5,#04H,RSYD_NEXT1      ; IF CURRENT STATE
                LJMP    STATE_F8                ; IS 04H OR 07H THEN GO TO
RSYD_NEXT1:     CJNE    R5,#07H,RSYD_NEXT2      ; LOST FRAME STATE
STATE_F8:       MOV     DPTR,#MESS11            ; ELSE GO TO UNSYNC STATE
                MOV     R1,#00H
                LCALL   LCD_DISPC
                MOV     R5,#05H                 ; CHANGE TO LOST FRAME STATE
                LJMP    INTR_COMPLETE
RSYD_NEXT2:     MOV     DPTR,#MESS9
                MOV     R1,#00H
                LCALL   LCD_DISPC
                MOV     R5,#03H                 ; CHANGE TO UNSYNC STATE
                LJMP    INTR_COMPLETE
ARD:            CJNE    R3,#00001000B,DR        ; ARD INDICATE
                MOV     DPTR,#MESS10
                MOV     R1,#00H
```

```
                LCALL   LCD_DISPC
                MOV     R5,#04H             ; CHANGE TO SYNC STATE
                LJMP    INTR_COMPLETE
DR:             CJNE    R3,#00000000B,SD    ; DR INDICATE
                MOV     DPTR,#MESS12
                MOV     R1,#00H
                LCALL   LCD_DISPC
                MOV     R5,#06H             ; CHANGE TO PEND.DEACT. STATE
                MOV     A,#01111100B        ; SEND DIU COMMAND
                MOV     DPTR,#CIXR
                MOVX    @DPTR,A
                LJMP    INTR_COMPLETE
SD:             CJNE    R3,#00000010B,AI8   ; SD INDICATE
                MOV     DPTR,#MESS15
                MOV     R1,#00H
                LCALL   LCD_DISPC
                MOV     R5,#07H             ; CHANGE TO ACT. STATE
                LJMP    INTR_COMPLETE
AI8:            CJNE    R3,#00001100B,EI    ; AI8 INDICATE
                MOV     DPTR,#MESS13
                MOV     R1,#00H
                LCALL   LCD_DISPC
                MOV     R5,#23H             ; CHANGE TO ACT. STATE
                LJMP    INTR_COMPLETE
EI:             CJNE    R3,#00000110B,DID   ;EI INDICATE
                LJMP    INTR_COMPLETE
DID:            CJNE    R3,#00001111B,AI10
                MOV     R5,#24H             ; CHANGE TO POWER DOWN STATE
                MOV     DPTR,#TE_STATUS     ; INDICATE POWER DOWN MODE
                MOVX    A,@DPTR
```

```
                CLR     ACC.6
                MOVX    @DPTR,A
                LJMP    INTR_COMPLETE
AI10:           CJNE    R3,#00001101B,OTHERS ; AI10 INDICATE
                MOV     DPTR,#MESS13
                MOV     R1,#00H
                LCALL   LCD_DISPC
                MOV     R5,#23H ;CHANGE TO ACT. STATE
                LJMP    INTR_COMPLETE
OTHERS:         MOV     DPTR,#MESS16
                MOV     R1,#00H
                LCALL   LCD_DISPC
                LJMP    INTR_COMPLETE
MESS1:          DB      ' Enter number:
MESS2:          DB      ' TEI assign failed.
MESS3:          DB      ' TE stand by.
MESS5:          DB      ' Fault Detected in TE
MESS6:          DB      ' or S Bus Disconnect
MESS7:          DB      ' Layer 1: Power Up'
MESS8:          DB      ' Layer 1: Pend.Act.'
MESS9:          DB      ' Layer 1: Unsync.
MESS10:         DB      ' Layer 1: Synchronize
MESS11:         DB      ' Layer 1: Lost Frame
MESS12:         DB      ' Layer 1: Pend.Deact.
MESS13:         DB      ' Layer 1  Activated
MESS4:          DB      ' Establishing L2 Link
MESS14:         DB      ' Physical Error
MESS17:         DB      ' Acitvate Failed
MESS15:         DB      ' Layer 1: Slip Detect
MESS16:         DB      ' Layer 1: Unknown int'
```

```
MESS18:     DB    ' Rx not Synchronize'
MESS19:     DB    ' Rx Synchronize
MESS20:     DB    ' Device Time Out'
MESS21:     DB    ' Tx message Repeat
MESS22:     DB    ' Tx Data Underrun
MESS23:     DB    ' Protocol Error
MESS24:     DB    ' Rx Frame Overflow
MESS25:     DB    ' Sync Transfer Overfw
MESS26:     DB    ' MONITOR Status
MESS27:     DB    ' Subscriber Awake'
MESS28:     DB    ' Watchdog Timer Ovflw
MESS29:     DB    ' Rx Data Overflow
MESS30:     DB    ' CRC Error
MESS31:     DB    ' Rx message abort
MESS32:     DB    ' Identity Denied.
MESS33:     DB    ' UI Frame Invalid
MESS34:     DB    ' Link Released
MESS35:     DB    ' TEI assigned.'
MESS36:     DB    ' IET
MESS37:     DB    ' TEI Removed
MESS38:     DB    ' Frame Reject
CLEAR:      DB
            END
```

ภาคผนวก ข

Software ที่ใช้กับ LT-S

```
LCD_PORT        EQU   0A000H      ;LCD ADDRESS
TIME_OUT_CNT    EQU   00014H      ;TIME OUT EACH VALUE
                                  ;CONTAIN 1 SEC.(NOT ADDRESS)
STATE_L1        EQU   07FFFH      ;LAYER1 CURRENT STATE
                                  ;STORE BY R5
                                  ;#00H STATE G1  DEACTIVATED
                                  ;#01H STATE G2  SYNCHRONIZED
                                  ;#02H STATE G3  ACTIVATED
                                  ;#03H STATE G4  PEND. DEACT.
                                  ;#04H STATE G5  UNACKN.
REQUEST         EQU   07FFEH      ;ACTIVATED REQUEST FROM
                                  ;#00H  LT_S
                                  ;#01H  TE
L1_TIME         EQU   07FFBH      ;COUNTER FOR ACT. TIME OUT
                                  ;(3 BYTES)
;************* ISAC-S IOM-1 LT_S MODE REGISTER *************
RFIFO           EQU   08000H      ;RECEIVE FIFO 00-1FH R
XFIFO           EQU   08000H      ;TRANSMIT FIFO 00-1FH W
ISTA            EQU   08020H      ;INTR STATUS REG R
MASK            EQU   08020H      ;MASK REG W
STAR            EQU   08021H      ;STATUS REG R
CMDR            EQU   08021H      ;COMMAND REG W
MODE            EQU   08022H      ;MODE REG R/W
TIMR            EQU   08023H      ;TIMER REG R/W
EXIR            EQU   08024H      ;EXTENDED INTR REG R
XAD1            EQU   08024H      ;TRANSMIT ADDR 1 W
RBCL            EQU   08025H      ;RECEIVE FRAME BYTE COUNT
```

```
                                    ;LOW R
XAD2        EQU   08025H            ;TRANSMIT ADDR 2 W
SAPR        EQU   08026H            ;RECEIVED SAPI REG R
SAP1        EQU   08026H            ;SAPI1 W
RSTA        EQU   08027H            ;RECEIVE STATUS REG R
SAP2        EQU   08027H            ;SAPI2 REG W
TEI1        EQU   08028H            ;TEI1 REG W
RHCR        EQU   08029H            ;RECEIVE HDLC CTRL REG R
TEI2        EQU   08029H            ;TEI2 REG W
RBCH        EQU   0802AH            ;RECEIVE FRAME BYTE
                                    ;COUNT HIGH R
STAR2       EQU   0802BH            ;STATUS REG 2 R/W
SPCR        EQU   08030H            ;SERIAL PORT CONTROL
                                    ;REG R/W
CIRR        EQU   08031H            ;COMMAND/INDICATION
                                    ;RECEIVE REG R
CIXR        EQU   08031H            ;COMMAND/INDICATION
                                    ;TRANSMIT REG W
MOR         EQU   08032H            ;MONITOR RECEIVE REG R
MOX         EQU   08032H            ;MONITOR TRANSMIT REG W
SSCR        EQU   08033H            ;SIP SIGNALLING CODE
                                    ;RECEIVE R
SSCX        EQU   08033H            ;SIP SIGNALLING CODE
                                    ;TRANSMTI W
SFCR        EQU   08034H            ;SIP FEATURE CTRL READ R
SFCW        EQU   08034H            ;SIP FEATURE CTRL WRITE W
C1R         EQU   08035H            ;CHANNEL REG 1 R/W
C2R         EQU   08036H            ;CHANNEL REG 2 R/W
B1CR        EQU   08037H            ;B1 CHANNEL REG R
STCR        EQU   08037H            ;SYNC TRANSFER CTRL REG W
```

```
B2CR            EQU   08038H      ;B2 CHANNEL REG R
ADF1            EQU   08038H      ;ADDITIONAL FEATURE REG 1 W
ADF2            EQU   08039H      ;ADDITIONAL FEATURE REG 2 R/W
SQRR            EQU   0803BH      ;S,Q CH RECEIVE REG R
SQXR            EQU   0803BH      ;S,Q CH TRANSMIT REG W
;REGISTER R3 USE FOR INDICATED COMMAND BY THESE VALUE
;     0XH  CISQ INTRRUPT
;     10H  RME INTERRUPT
;     20H  RPF INTR.
;     30H  RSC INTR.
;     40H  XPR INTR.
;     50H  TIN INTR.
;     60H  SIN INTR.
;     80H  XMR INTR. (EXIR INTR.)
;     90H  XDU INTR. (EXIR INTR.)
;     A0H  PCE INTR. (EXIR INTR.)
;     B0H  RFO INTR. (EXIR INTR.)
;     C0H  SOV INTR. (EXIR INTR.)
;     D0H  MOS INTR. (EXIR INTR.)
;     E0H  SAW INTR. (EXIR INTR.)
;     F0H  WOV INTR. (EXIR INTR.)
; ********** JUMP ADDRESS DEFINED **********
      ORG   0000H       ; SET START ADDRESS
      LJMP  START
      ORG   0003H       ; ISAC-S INTERRUPT ROUTINE
      LJMP  ISAC_INTR
; ********** MAIN **********
START:    MOV     IE,#00H     ; MASK ALL INTERRUPT
          MOV     R2,#20H
          LCALL   DELAY       ; LCD INITIALIZE
```

```
MOV     P1,#5FH         ; CLEAR PORT1
MOV     A,#01H          ; CLEAR
LCALL   LCDWI
MOV     A,#00111000B    ; INITIALIZE LCD FUNCTION SET
LCALL   LCDWI
MOV     A,#00001100B    ; DISPLAY ON
LCALL   LCDWI
MOV     A,#00000110B    ; ENTRY MODE SET
LCALL   LCDWI
MOV     A,#02H          ; HOME
LCALL   LCDWI
MOV     PSW,#00H
SETB    IE.0            ; ENABLE 2085 INTR.
CLR     IE.2            ; CLEAR INTR1
CLR     TCON.0          ; SET LEVEL DETECT OF INT0
CLR     TCON.2
MOV     R3,#00H
LCALL   INIT_2085
MOV     DPTR,#MESS1
MOV     R1,#00H
LCALL   LCD_DISPC
MOV     DPTR,#CLEAR
MOV     R1,#01H
LCALL   LCD_DISPC
SETB    IE.7            ; ENABLE INTERRUPT
LCALL   DEACTIVATE
MOV     A,#01H          ; SET ACTIVATED REQUEST
                        ; FROM LT_S
MOV     DPTR,#REQUEST
MOVX    @DPTR,A
```

```
; ************************************************************
; *              INITIALIZE COMPLETE                         *
; *                BEGIN PROGRAM                             *
; ************************************************************
CHK_REQ:    MOV     DPTR,#REQUEST     ; CHECKING REQUEST FROM
            MOVX    A,@DPTR           ; LT_S(#00H) OR  TE(#01H)
            CJNE    A,#00H,CHK_TE
                                      ; IF EQUAL,LT_S REQUEST ACT.
            LCALL   ACTIVATE_L1       ; THEN ACTIVATE LAYER1
            CJNE    R5,#00H,LAYER2
                                      ; IF RECEIVE TIME OUT ERROR
            LCALL   DEACTIVATE        ; THEN DEACTIVATE LT_S
            LJMP    CHK_REQ
LAYER2:     MOV     R2,#100
            LCALL   DELAY             ; INSERT LAYER2
            LCALL   DEACTIVATE
            LJMP    CHK_REQ
CHK_TE:     CJNE    R5,#02H,ACT_TIME_OUT
                                      ; IF EQU,TE REQUEST FOR ACT.
            MOV     R2,#50            ; INSERT LAYER2
            LCALL   DELAY
            LCALL   DEACTIVATE
            MOV     R2,#50
            LCALL   DELAY
            LJMP    CHK_REQ
ACT_TIME_OUT: CJNE  R5,#00H,ACT_L1_FRM_TE
            LJMP    CHK_REQ
ACT_L1_FRM_TE:MOV   DPTR,#L1_TIME
            MOVX    A,@DPTR
            DEC     A                 ;decrease counter first byte
```

```
                MOVX    @DPTR,A
                JNZ     CHK_REQ
                INC     DPTR            ;carry to 2nd counter byte
                MOVX    A,@DPTR
                DEC     A
                MOVX    @DPTR,A
                JNZ     CHK_REQ
                INC     DPTR            ;carry to 3rd counter byte
                MOVX    A,@DPTR
                INC     A
                MOVX    @DPTR,A
                CJNE    A,#TIME_OUT_CNT,CHK_REQ
                                        ;if fail for 20sec indicate
                LCALL   TIME_OUT        ;timeout  error
                LCALL   DEACTIVATE
                LJMP    CHK_REQ
;****************** ACTIVATE  LT_S  LAYER 1 ********************
; RETURN VALUE IN R5
;               02H     ACTIVATE COMPLETE
;               00H     ACTIVATE FAIL
ACTIVATE_L1:  CJNE      R5,#00H,STILL_ACT  ;If state=00H then activate lt_s
                LCALL   RESET_T3
                MOV     A,#01100000B    ;send ARD command
                MOV     DPTR,#CIXR
                MOVX    @DPTR,A
WAIT_INT:       MOV     DPTR,#L1_TIME
                MOVX    A,@DPTR
                DEC     A               ;decrease counter first byte
                MOVX    @DPTR,A
                JNZ     STATE_ACT
```

```
                INC     DPTR            ;carry to 2nd counter byte
                MOVX    A,@DPTR
                DEC     A
                MOVX    @DPTR,A
                JNZ     STATE_ACT
                INC     DPTR            ;carry to 3rd counter byte
                MOVX    A,@DPTR
                INC     A
                MOVX    @DPTR,A
                CJNE    A,#TIME_OUT_CNT,STATE_ACT
                                        ;if fail for 20sec indicate
                LCALL   TIME_OUT        ;timeout  error
                MOV     R5,#00H         ;if time_out error then
                                        ;give stste#00H
                LJMP    STILL_ACT
STATE_ACT:      CJNE    R5,#02H,WAIT_INT ;if state=#02H,activate
                                        ;layer1 complete
STILL_ACT :     RET
; *** TIME OUT INDICATE FROM LAYER 1 COUNTER AND TIN INTR. **
TIME_OUT:       CLR     IE.7
                MOV     DPTR,#MESS5
                MOV     R1,#00H
                LCALL   LCD_DISPC
                MOV     DPTR,#MESS6
                MOV     R1,#01H
                LCALL   LCD_DISPC
                MOV     DPTR,#CLEAR
                MOV     R1,#01H
                LCALL   LCD_DISPC
                SETB    IE.7
```

```
                RET
;********************** INITIALIZE   2085 ***********************
; PARAMETER
; VAR A,DPTR
INIT_2085:      MOV         A,#0FFH
                MOV         DPTR,#MASK
                MOVX        @DPTR,A
                MOV         A,#00H
                MOV         DPTR,#ADF2
                MOVX        @DPTR,A             ;SELECT IOM-1 MODE
                MOV         A,#00100000B        ;SET SPCR REG, SLD NO OPERATE
                MOV         DPTR,#SPCR
                MOVX        @DPTR,A
                MOV         A,#00111000B        ;SET AUTO MODE WITH 2 BYTE
                                                ADDR
                MOV         DPTR,#MODE          ;FIELDS INTERNAL TIMER
                                                ACTIVE HDLC
                MOVX        @DPTR,A             ;RECEIVER IN LT_S MODE
                MOV         A,#10100000B        ;SET ISAC TO RETRY 5 TIMES
                MOV         DPTR,#TIMR
                MOVX        @DPTR,A
                MOV         A,#01110000B        ;NOT USE SPECIAL FN REG
                MOV         DPTR,#STCR
                MOVX        @DPTR,A
                MOV         A,#00000000B
                MOV         DPTR,#ADF1
                MOVX        @DPTR,A
                MOV         R5,#00H
                MOV         A,#00H
                MOV         DPTR,#MASK
```

```
                MOVX        @DPTR,A
                RET
;***************** DEACTIVATE LT_S ********************
DEACTIVATE:  MOV        A,#01000000B        ;send DR command
                MOV         DPTR,#CIXR
                MOVX        @DPTR,A
                CJNE        R3,#0FH,$           ; WAIT DIU INDICATION
                MOV         A,#01111100B        ;send DID command
                MOV         DPTR,#CIXR
                MOVX        @DPTR,A
                RET
; ******************* DELAY SUB ******************
; PARAMETER=R2
; VAR=R2,R3,R4
DELAY:          MOV         A,R3
                PUSH        ACC
                MOV         A,R4
                PUSH        ACC
DELAY1:         MOV         R3,#0               ; DELAY 65.536 mS FOR EACH R2
DELAY2:         MOV         R4,#0
                DJNZ        R4,$
                DJNZ        R3,DELAY2
                DJNZ        R2,DELAY1
                POP         ACC
                MOV         R4,A
                POP         ACC
                MOV         R3,A
                RET
; ************ LCD DISPLAY IN CODE SEGMENT *****************
; PARAMETER=DPTR OF START BLOCK 20 BYTES IN PROGRAME MEMORY
```

```
;       R1=00H DISPLAY LINE 1, R1=01H DISPLAY LINE 2
; VAR=A, R0, R1, R7, DPTR
LCD_DISPC:  MOV     A,R0
            PUSH    ACC
            MOV     A,R7
            PUSH    ACC
            CJNE    R1,#00H,LCDLDS2
            MOV     A,#80H          ; SET LINE 1 LCD ADDRESS
            LJMP    LCDLDS3
LCDLDS2:    MOV     A,#0C0H
LCDLDS3:    MOV     R1,A
            PUSH    DPH
            PUSH    DPL
            LCALL   LCDWI
            POP     DPL
            POP     R7,#20          ; 20 CHARS.
            MOV     R0,#0
LCDLDS1:    MOV     A,R0
            MOVC    A,@A+DPTR
            PUSH    DPH
            PUSH    DPL
            LCALL   LCDWD
            INC     R1
            MOV     A,R1
            LCALL   LCDWI
            POP     DPL
            POP     DPH
            INC     R0
            DJNZ    R7,LCDLDS1
            POP     ACC
```

```
                MOV     R7,A
                POP     ACC
                MOV     R0,A
                RET
; ********** LCD DISPLAY IN DATA SEGMENT **************
; PARAMETER=DPTR OF START BLOCK 20 BYTES IN DATA MEMORY
;       R1=00H DISPLAY LINE 1, R1=01H DISPLAY LINE 2
; VAR=A, R0, R1, R7, DPTR
LCD_DISPX:  MOV     A,R0
            PUSH    ACC
            MOV     A,R7
            PUSH    ACC
            CJNE    R1,#00H,LCDLDX2
            MOV     A,#80H          ; SET LINE 1 LCD ADDRESS
            LJMP    LCDLDX3
LCDLDX2:    MOV     A,#0C0H
LCDLDX3:    MOV     R1,A            ; PUT LCD ADDRESS TO R1
            PUSH    DPH
            PUSH    DPL
            LCALL   LCDWI
            POP     DPL
            POP     DPH
            MOV     R7,#20          ; 20 CHARS.
LCDLDX1:    MOVX    A,@DPTR
            PUSH    DPH
            PUSH    DPL
            LCALL   LCDWD
            INC     R1
            MOV     A,R1
            LCALL   LCDWI
```

```
                POP     DPL
                POP     DPH
                INC     DPTR
                DJNZ    R7,LCDLDX1
                POP     ACC
                MOV     R7,A
                POP     ACC
                MOV     R0,A
                RET
; ***************** LCDWI *************
; LCD WRITE INSTRUCTION
; PARAMETER=A
; VAR=A, DPTR
LCDWI:          MOV     DPTR,#LCD_PORT
                CLR     P1.0 ;RS=0
                CLR     P1.1
                CLR     P1.2
                MOVX    @DPTR,A
                SETB    P1.2
                NOP
                CLR     P1.2
                MOV     A,#0
LCDWI1:         DEC     A
                JNZ     LCDWI1
                SETB    P1.1
                RET
; ****************** LCDWD ***************
; LCD WRITE DATA
; PARAMETER=A
; VAR=A, DPTR
```

```
LCDWD:      MOV     DPTR,#LCD_PORT
            SETB    P1.0 ; RS=1
            CLR     P1.1
            CLR     P1.2
            MOVX    @DPTR,A
            SETB    P1.2
            NOP
            CLR     P1.2
            MOV     A,#0
LCDWD1:     DEC     A               ; WAIT FOR LCD WORKING
            JNZ     LCDWD1
            SETB    P1.1
            RET
; ********************** RESET T3 (I.430) *****************
; RESET TIMER T3 (CCITT I.430)
; PARAMETER
; VAR A,DPTR
RESET_T3:   MOV     DPTR,#L1_TIME
            MOV     A,#00H
            MOVX    @DPTR,A
            INC     DPTR
            MOVX    @DPTR,A
            INC     DPTR
            MOVX    @DPTR,A
            MOV     DPTR,#CLEAR
            MOV     R1,#01H
            LCALL   LCD_DISPC
            RET
; ******************** ISAC-S INTERRUPT HANDLE ***************
; RETURN INDICATION IN R3
```

```
; VAR A,R3,R0,R5
ISAC_INTR:  CLR     IE.7          ;DISABLE 2085 INTERRUPT.
            PUSH    ACC
            MOV     A,R0
            PUSH    ACC
            MOV     DPTR,#ISTA    ;GET INTERRUPT FUNCTION
            MOVX    A,@DPTR
            MOV     R0,A          ;SAVE INTERRUPT FN TO R0
            ANL     A,#00000100B  ;CHECK CISQ INTERRUPT
            JNZ     CISQ_INT
            MOV     A,R0
            ANL     A,#00001000B  ;CHECK TIMER INTRRUPT
            JNZ     TIN_INT
; ************** OTHER INTR. HANDLE ************************
INTR_COMPLETE:POP   ACC
            MOV     R0,A
            POP     ACC
            SETB    IE.7          ;ENABLE 2085 INTERRUPT
            RETI
;************************ TIN INTERRUPT HANDLE **************
TIN_INT:    MOV     R3,#50H
            LJMP    INTR_COMPLETE
;************************ CISQ  INTERRUPT HANDLE ************
CISQ_INT:   LCALL   RESET_T3
            MOV     DPTR,#CIRR
            MOVX    A,@DPTR
            RR      A
            RR      A
            ANL     A,#0FH
            MOV     R3,A
```

```
        MOV     DPTR,#SQRR
        MOVX    A,@DPTR
        CJNE    R3,#00001111B,AIU ;DIU indicate
        MOV     DPTR,#MESS2
        MOV     R1,#00H
        LCALL   LCD_DISPC
        MOV     R5,#00H
        LJMP    INTR_COMPLETE
AIU:    CJNE    R3,#00001100B,ARU ;AIU INDICATE
        MOV     R5,#02H
        MOV     DPTR,#MESS3
        MOV     R1,#00H
        LCALL   LCD_DISPC
        LJMP    INTR_COMPLETE
ARU:    CJNE    R3,#00001000B,RSYU
        MOV     DPTR,#MESS5
        MOV     R1,#00H
        LCALL   LCD_DISPC
        MOV     R5,#01H
        LJMP    INTR_COMPLETE
RSYU:   CJNE    R3,#00000100B,LSL   ;RECEIVER NOT SYNC
                                    ;INDICATE
        MOV     R5,#02H
        MOV     DPTR,#MESS6
        MOV     R1,#00H
        LCALL   LCD_DISPC
        MOV     R2,#5
        LCALL   DELAY
        LJMP    INTR_COMPLETE
LSL:    CJNE    R3,#00000001B,OTHERS
```

```
                                    ;NO RECEIVE SIGNAL
                                    ;INDICATE
            MOV     R5,#02H
            MOV     DPTR,#MESS7
            MOV     R1,#00H
            LCALL   LCD_DISPC
            MOV     R2,#5
            LCALL   DELAY
            LJMP    INTR_COMPLETE
OTHERS:     MOV     DPTR,#MESS8
            MOV     R1,#00H
            LCALL   LCD_DISPC
            MOV     R2,#5
            LCALL   DELAY
            LJMP    INTR_COMPLETE
MESS1:      DB      'Initialize Complete.'
MESS2:      DB      'Layer 1: Deactivated'
MESS3:      DB      'Layer 1: Activated'
MESS4:      DB      'Layer 1: Pend.Deact.'
MESS5:      DB      'Layer 1: Synchronize'
MESS6:      DB      'Layer 1: Rx not Sync'
MESS7:      DB      'Layer 1: No Signal'
MESS8:      DB      'Layer 1: Unknown Int'
CLEAR:      DB'
            END
```

ประวัติผู้เขียน

นายใหญ่ ภาวนานนท์ เกิดเมื่อวันที่ 3 พฤษภาคม พ.ศ. 2512 ที่กรุงเทพมหานคร สำเร็จการศึกษา ระดับปริญญาตรี วิศวกรรมศาสตร์บัณฑิต สาขาไฟฟ้าสื่อสาร ภาควิชาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย ในปีการศึกษา 2535 และเข้าศึกษาต่อในหลักสูตร วิศวกรรมศาสตร์มหาบัณฑิต สาขาไฟฟ้าสื่อสาร ที่จุฬาลงกรณ์ มหาวิทยาลัย ในปีการศึกษา 2536 จนถึงปัจจุบัน