DATA1 EQU 39H Data buffer

DATA2 EQU 40H

DATA3 EQU 41H

SEC EQU 42H ;second save address

MIN EQU 43H ;minute save address

HOUR EQU 44H ;hour save address

SDA BIT P1.0 ;Serial data input \ output

SCL BIT P1.1 ;Serial clock input

ORG 0000H

MOV 30H,#0

MOV 31H,#0

MOV 32H,#10

MOV 33H,#0

MOV 34H,#0

MOV 35H,#10

MOV 36H,#0

MOV 37H,#0

CLR P1.7

MOV R3,#00H ;Initialize the PIC

MOV R4,#00H

MOV DPTR,#TAB

CALL WDATA ;send data to PIC

CALL RDATA ;read data from PIC

; CALL CHUSHI ;Time initialization

;\*\*\*\*\*\*\*\*\*\*\*\*Timezone\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

SHIJIAN: CALL READ ;get the time from PIC

Q1: CALL ZH

CALL DISP

JNB P3.0,ANJIAN0 ;minute adjust

JNB P3.1,ANJIAN1 ;hour adjust

JMP SHIJIAN

;\*\*\*\*\*\*\*\*\*\*\*key dealing code\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;\*\*\*\*\*\*\*\*\*\*\*week setting code\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Data segment

Table dw t1,t2.t3,t4,t5,t6,t7;set the address format

T1 db ‘MON$’;this is Monday on the screen

T2 db ‘TUE$’;this is Tuesday on the screen

T3 db ‘WENN$’;this is Wednesday on the screen

T4 db ‘THU$’;this is Thursday on the screen

T5 db ‘FRI$’;this is Friday on the screen

T6 db ‘SAT$’;this is Saturday on the screen

T7 db ‘SUN$’;this is Sunday on the screen

Data ends

Code segment

assume ds:data,cs:code

Start:

mov ax,data

mov ds,ax

mov ah,08h;Interrupt input numbers 1-7

int 21h

sub al.30h; change ascii to number

dec al; subtract 1, if the input is 1, it means the first position

shl al,1; multiple with 2, to set the address

xor ah,ah; empty ah

mov bx,,ax

mov dx,[bx];Pass the first address of the output string to dx

mov ah,09h

Int 21h

mov ah,01h

Int 21h

mov ah,01h

Int 21h

mov ax,4c00h

Int 21h;code ends

;\*\*\*\*\*\*\*\*\*\*\*\*minute dealing code\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ANJIAN0: CALL DELAY1

JB P3.0,Q1 ;Press the key 1 to call the display

W2: JB P3.0,W1 ;Press the key 2 to add one

CALL DISP ;

JMP W2 ;return to main menu double check

W1: MOV A,MIN

ADD A,#1 ;Press the key 2 to add one in minute

DA A ;Decimal adjustment

CJNE A,#60H,W3

MOV A,#0

W3: MOV R3,#03H ;send the setted time to PIC

MOV R4,A

CALL WDATA

JMP SHIJIAN

;\*\*\*\*\*\*\*\*\*\*\*\*Time button dealing code\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ANJIAN1: CALL DELAY1

JB P3.1，Q1

W4: JB P3.1,W5

CALL DISP

JMP W4

W5: MOV A,HOUR

ADD A,#1

DA A

CJNE A,#24H,W6

MOV A,#0

W6: MOV R3,#04H

MOV R4,A

CALL WDATA

JMP SHIJIAN

;\*\*\*\*\*\*\*\*\*\*\*\*send the code to the PIC\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;R3=the address the information sent to,R4=the number to be sent

WDATA: CALL RWXT

MOV DATA2,R4

CALL XSZ

CALL STOP

RET

;\*\*\*\*\*\*\*\*\*\*\*\*read message from PIC\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

RDATA: CALL RWXT

CALL START

MOV DATA2,#0A1H

CALL XSZ

CALL DSZ

CALL STOP

RET

;\*\*\*\*\*\*\*\*\*\*\*\*send the information from the equipment address\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

RWXT: MOV DATA3,R3

CALL START

MOV DATA2,#0A0H

CALL XSZ

MOV DATA2,DATA3

CALL XSZ

RET

;\*\*\*\*\*\*\*\*\*\*\*\*send the initialized time to PIC\*\*\*\*\*\*\*

/\*CHUSHI: MOV DPTR,#TDATA

MOV R5,#3

MOV R0,#00H

CHUSHI1: MOV A,R0

MOVC A,@A+DPTR

MOV R3,A

INC R0

MOV A,R0

MOVC A,@A+DPTR

MOV R4,A

INC R0

CALL WDATA

DJNZ R5,CHUSHI1

RET

TDATA: DB 02H,43H,03H,15H,04H,55H \*/

;\*\*\*\*\*\*\*\*\*\*\*\*the screen read the hour minute and second information code\*\*\*\*\*\*\*\*\*\*\*\*\*\*

READ: MOV R3,#02H

CALL RDATA

MOV SEC,A ;read second

MOV R3,#03H

CALL RDATA

MOV MIN,A ;read minute

MOV R3,#04H

CALL RDATA

MOV HOUR,A

ANL HOUR,#3FH ;read hour

;\*\*\*\*\*\*\*\*\*\*\*\*write a byte to PIC\*\*\*\*\*\*\*\*\*

XSZ: MOV R2,#08H

MOV A,DATA2

XSZ0: RLC A

JNC XSZ1

SETB SDA

JMP XSZ2

XSZ1: CLR SDA

XSZ2: CALL CLOCK\_LHL

DJNZ R2,XSZ0

CALL CLOCK\_LHL

RET

;\*\*\*\*\*\*\*\*\*\*\*\*read a byte from PIC\*\*\*\*\*\*\*\*\*

DSZ: MOV R2,#08H

DSZ0: SETB SDA

JB SDA,DSZ1

CLR C

SJMP DSZ2

DSZ1: SETB C

DSZ2: RLC A

CALL CLOCK\_LHL; the clock function

DJNZ R2,DSZ0

MOV DATA1,A

CALL CLOCK\_LHL

RET

;\*\*\*\*\*\*\*\*\*\*\*\*start \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

START: SETB SDA

SETB SCL

NOP

NOP

CLR SDA

NOP

NOP

CLR SCL

RET

;\*\*\*\*\*\*\*\*\*\*\*\*stio\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

STOP: CLR SDA

SETB SCL

NOP

SETB SDA

NOP

CLR SCL

RET

;\*\*\*\*\*\*\*\*\*\*\*\*Shift pulse\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

CLOCK\_LHL: CLR SCL

NOP

SETB SCL

NOP

CLR SCL

NOP

RET

;\*\*\*\*\*\*\*\*\*\*\*\*Convert hexadecimal to decimal and send it to the display area\*\*\*\*

ZH: MOV A,SEC ;Take seconds to store the value

MOV B,#10H ;Convert hexadecimal to decimal

DIV AB

MOV 31H,A ;seconds storage for (compressed bcd code)

MOV 30H,B ;seconds storage

MOV A,MIN

MOV B,#10H

DIV AB

MOV 34H,A

MOV 33H,B

MOV A,HOUR

MOV B,#10H

DIV AB

MOV 37H,A

MOV 38H,B

RET

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*date display\*\*\*\*\*\*\*\*\*\*\*\*\*

START: MOV AX,0001H ;Set the display mode to 40\*25 color text mode

INT 10H

MOV AX,DATA

MOV DS,AX

MOV ES,AX

MOV BP,OFFSET SPACE

MOV DX,0B00H

MOV CX,1000

MOV BX,0040H

MOV AX,1300H

INT 10H

MOV BP,OFFSET PATTERN ;Show rectangular bar

MOV DX,0B00H

MOV CX,120

MOV BX,004EH

MOV AX,1301H

INT 10H

LEA DX,STR ;

MOV AH,9

INT 21H

MOV AH,1 ;Enter a single character from the keyboard

INT 21H

CMP AL,44H ;AL='D'？

JNE A

CALL DATE ;show the date

A: CMP AL,54H ;AL='T'？

JNE B

CALL TIME ;show the time

B: CMP AL,51H ;AL='Q'？

JNE START

MOV AH,4CH ;back to doc condition

INT 21H

DATE PROC NEAR ;Display date subroutine

DISPLAY:MOV AH,2AH ;show the date

INT 21H

MOV SI,0

MOV AX,CX

MOV BX,100

DIV BL

MOV BL,AH

CALL BCDASC1 ;The date value is converted into the corresponding ASCII code characte

MOV AL,BL

CALL BCDASC1

INC SI

MOV AL,DH

CALL BCDASC1

INC SI

MOV AL,DL

CALL BCDASC1

MOV BP,OFFSET DBUFFER1

MOV DX,0C0DH

MOV CX,20

MOV BX,004EH

MOV AX,1301H

INT 10H

MOV AH,02H ;Set cursor position

MOV DX,0300H

MOV BH,0

INT 10H

MOV BX,0018H

REPEA: MOV CX,0FFFFH ;time delay

REPEAT:LOOP REPEAT

DEC BX

JNZ REPEA

MOV AH,01H ;Read keyboard buffer characters to AL register

INT 16H

JE DISPLAY

JMP START

MOV AX,4C00H

INT 21H

RET

DATE ENDP

TIME PROC NEAR ;Display time subroutine

DISPLAY1:MOV SI,0

MOV BX,100

DIV BL

MOV AH,2CH ;get the time information

INT 21H

MOV AL,CH

CALL BCDASC ;

INC SI

MOV AL,CL

CALL BCDASC

INC SI

MOV AL,DH

CALL BCDASC

MOV BP,OFFSET DBUFFER

MOV DX,0C0DH

MOV CX,20

MOV BX,004EH

MOV AX,1301H

INT 10H

MOV AH,02H

MOV DX,0300H

MOV BH,0

INT 10H

MOV BX,0018H

RE: MOV CX,0FFFFH

REA: LOOP REA

DEC BX

JNZ RE

MOV AH,01H

INT 16H

JE DISPLAY1

JMP START

MOV AX,4C00H

INT 21H

RET

TIME ENDP

BCDASC PROC NEAR ;

PUSH BX

CBW

MOV BL,10; change the date information to decimal expression

DIV BL

ADD AL,'0'

MOV DBUFFER[SI],AL

INC SI

ADD AH,'0'

MOV DBUFFER[SI],AH

INC SI

POP BX

RET

BCDASC ENDP

BCDASC1 PROC NEAR ;Subroutine for converting date value into ASCII code character

PUSH BX

CBW

MOV BL,10

DIV BL

ADD AL,'0'

MOV DBUFFER1[SI],AL

INC SI

ADD AH,'0'

MOV DBUFFER1[SI],AH

INC SI

POP BX

RET

BCDASC1 ENDP

CODE ENDS

END START

;\*\*\*\*\*\*\*\*\*\*\*\*time display\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

DISP: MOV DPTR,#TAB

MOV R0,#30H

MOV R1,#0FEH

MOV R2,#08 ;run for 8 times

LOOP: MOV P2,#0FFH ;

MOV A,@R0

MOVC A,@A+DPTR

MOV P0,A

MOV A,R1

RR A

MOV P2,A

MOV R1,A

INC R0

CALL DELAY

DJNZ R2,LOOP

RET

;\*\*\*\*\*\*\*\*\*\*\*\*time delay program\*\*\*\*\*\*\*\*\*\*\*

DELAY: MOV 20H,#50

D1: MOV 21H,#20

DJNZ 21H,$

DJNZ 20H,D1

RET

DELAY1: MOV 20H,#250

D2: MOV 21H,#200

DJNZ 21H,$

DJNZ 20H,D2

RET

TAB: DB 3FH,06H,5BH,4FH,66H,6DH,7DH,07H,7FH,6FH,40H

ORG 0000H

LJMP MAIN

ORG 000BH

LJMP TIME

;××××× main program part: ×××××

ORG 0100H

MAIN:MOV SP,#50H

MOV 21H,#00H ;Minute　BIN

MOV 22H,#00H ;Hour　BIN

MOV 23H,#01H

MOV 24H,#01H

MOV 25H,#00H

MOV 30H,#00H

MOV 31H,#00H

MOV 32H,#00H

MOV 33H,#00H

MOV 34H,#00H

MOV 35H,#00H

MOV 36H,#01H

MOV 37H,#00H

MOV 38H,#01H

MOV 39H,#00H

MOV TMOD, #01H; 16-bit counter

MOV TH0, #03CH; Assign initial value of count

MOV TL0,#0B0H

MOV IE,#10000111B

SETB TR0; T0 starts counting

MOV R2,#14H

MOV P2,#0FFH

LOOP: LCALL TIMEPRO

LCALL DISPLAY1

JB P1.1,M1

LCALL SETTIME; call set time program

LJMP LOOP

M1: JB P1.2, M2

LCALL SETATIME; call set time program

LJMP LOOP

M2: JB P1.4, M4

LCALL LOOKATIME; call to set the alarm time program

M4: LJMP LOOP

DELAY:MOV R4,#030H ;delay time

DL00:MOV R5,#0FFH

DL11:MOV R6,#9H

DL12: DJNZ R6, DL12

DJNZ R5,DL11

DJNZ R4,DL00

RET

;×××××Set time program:×××××

SETTIME:

L0: LCALL DISPLAY1; call time allows the program

MM1: JB P1.2,L1

MOV C,P1.2

JC MM1

LCALL DELAY1; call delay

JC MM1

MSTOP1: MOV C,P1.2

JNC MSTOP1; Judge whether P1.2 is released? Release to continue

LCALL DELAY1; call delay

MOV C,P1.2

JNC MSTOP1

INC 22H; hour increase by 1

MOV A,22H

CJNE A, #18H, GO12; Determine whether the hour is up to 24 o'clock? Not to continue loop

MOV 22H, #00H; hour reset

MOV 34H,#00H

MOV 35H,#00H

LJMP L0

L1: JB P1.3, L2

MOV C,P1.3

JC L1

LCALL DELAY1; delay

JC L1

MSTOP2: MOV C,P1.3

JNC MSTOP2; Judge whether P1.3 is released? Release to continue

LCALL DELAY1; call delay

MOV C,P1.3

JNC MSTOP2

INC 21H; Increase by one minute

MOV A,21H

CJNE A,#3CH,GO11

MOV 21H,#00H ;Minute reset

MOV 32H,#00H

MOV 33H,#00H

LJMP L0

GO11:MOV B,#0AH; Divide the content in A into high and low parts

DIV AB

MOV 32H,B

MOV 33H,A

LJMP L0

GO12: MOV B,#0AH

DIV AB

MOV 34H,B

MOV 35H,A

LJMP L0

L2: JB P1.4, L0

MOV C,P1.4

JC L2

LCALL DELAY1; call delay

MOV C,P1.4

JC L2

STOP1: MOV C, P1.4; Determine whether the button P1.4 is released?

JNC STOP1

LCALL DELAY1; call delay

MOV C,P1.4

JNC STOP1

LJMP LOOP

;×××××Set the alarm time×××××

SETATIME:LCALL DISPLAY2; run at call time

N0:LCALL DISPLAY2

MM2: JB P1.3, N1; Judge whether P1.3 is pressed?

MOV C,P1.3

JC MM2

LCALL DELAY1

JC MM2

MSTOP3: MOV C, P1.3; Determine whether P1.3 is released?

JNC MSTOP3

LCALL DELAY1

MOV C,P1.3

JNC MSTOP3

INC 24H; set hour increase by 1

MOV A,24H

CJNE A,#24,GO22

MOV 24H, #00H; clock reset

MOV 38H,#00H

MOV 39H,#00H

LJMP N0

N1: JB P1.1, N2; Judge whether P1.1 is pressed?

MOV C,P1.1

JC N1

LCALL DELAY1

JC N1

MSTOP4: MOV C, P1.1; Determine whether P1.1 is released?

JNC MSTOP4

LCALL DELAY1

MOV C,P1.1

JNC MSTOP4

INC 23H; Set the alarm minute to increase by 1

MOV A,23H

CJNE A, #60, GO21; Determine whether A reaches 60 points?

MOV 23H,#00H ;Minute reset

MOV 36H,#00H

MOV 37H,#00H

LJMP N0

GO21:MOV B,#0AH; Divide the content in A into high and low parts

DIV AB

MOV 36H,B

MOV 37H,A

LJMP N0

GO22: MOV B,#0AH

DIV AB

MOV 38H,B

MOV 39H,A

LJMP N0

N2: JB P1.4, N0; Judge whether P1.4 is pressed?

MOV C,P1.4

JC N2

LCALL DELAY1

MOV C,P1.4

JC N2

STOP2: MOV C, P1.4; Determine whether P1.4 is released?

JNC STOP2

LCALL DELAY1

MOV C,P1.4

JNC STOP2

LJMP LOOP

TIMEPRO:MOV A,21H

MOV B,23H

CJNE A, B, BK; Judge whether the minute runs to the minute of the set alarm?

MOV A,22H

MOV B,24H

CJNE A, B, BK; Determine whether the clock is running to the set alarm clock?

SETB 25H.0

MOV C,25H.0

JC XX

XX: LCALL TIMEOUT; call the time alarm response program

BK:RET

TIMEOUT:

X1: LCALL BZ; call the speaker response program

LCALL DISPLAY2

CLR 25H.0

JB P1.4, X1; Judge whether P1.4 is pressed?

LCALL DELAY

CLR 25H.0

LJMP DISPLAY1

BZ: CLR P3.7; speaker response program

MOV R7, #250; Response delay time

T2: MOV R6,#124

T3: DJNZ R6, T3

DJNZ R7, T2

SETB P3.7

RET

LOOKATIME:LCALL DISPLAY2; call time to run the program

MM: JB P1.4, LOOKATIME; Judge whether the button P1.4 is pressed

MOV C,P1.4

JC MM

LCALL DELAY1

MOV C,P1.4

JC MM

STOP3: MOV C,P1.4

JNC STOP3

LCALL DELAY1

MOV C,P1.4

JNC STOP3

LJMP LOOP

DELAY1: MOV R4,#14H; time delay

DL001: MOV R5,#0FFH

DL111: DJNZ R5,DL111

DJNZ R4,DL001

RET

;×××××time running program×××××

TIME: PUSH ACC; On-site protection

PUSH PSW

MOV TH0,#03CH; Assign initial value

MOV TL0,#0B0H

DJNZ R2,RET0

MOV R2,#14H

MOV A,20H

CLR C

MOV 30H,#0

MOV 31H,#0

MOV A,21H

INC A; Minutes add 1

CJNE A, #3CH, GO2; Determine whether the minute reaches 60 minutes?

MOV 21H, #0H; reset to 60 minutes

MOV 32H,#0

MOV 33H,#0

MOV A,22H

INC A; clock increments by 1

CJNE A, #18H, GO3; Determine whether the clock reaches 24 o'clock?

MOV 22H, #00H; reset at 24

MOV 34H,#0

MOV 35H,#0

AJMP RET0

GO1: MOV 20H,A

MOV B,#0AH

DIV AB

MOV 31H,A

MOV 30H,B

AJMP RET0

GO2: MOV 21H,A

MOV B,#0AH

DIV AB

MOV 33H,A

MOV 32H,B

AJMP RET0

GO3: MOV 22H,A

MOV B,#0AH

DIV AB

MOV 35H,A

MOV 34H,B

AJMP RET0

RET0: POP PSW; restore scene

POP ACC

RETI

;××××× running part×××××

DISPLAY1: MOV R0,#30H

MOV R3,#0FEH

MOV A, R3

PLAY1: MOV P2,A

MOV A,@R0

MOV DPTR,#DSEG1; The first address of the table is sent to DPTR

MOVC A,@A+DPTR

MOV P0,A

LCALL DL1

MOV P2, #0FFH; Send high level to P2

MOV A, R3

RL A;

JNB ACC.6, LD1

INC R0

MOV R3,A

LJMP PLAY1; call the lookup table program

LD1: RET

DISPLAY2: PUSH ACC; field protection

PUSH PSW

MOV R0,#36H

MOV R3,#0FBH

MOV A, R3

PLAY2: MOV P2,A

MOV A,@R0

MOV DPTR,#DSEG1; The first address of the table is sent to DPTR

MOVC A,@A+DPTR ;Check ASCII special code

MOV P0, a

LCALL DL1

MOV P2, #0FFH; Send high level to P2

MOV A, R3

RL A

JNB ACC.6, LD2

INC R0

MOV R3,A

LJMP PLAY2

LD2: POP PSW; restore the scene

POP ACC

RET

;×××××delay time×××××

DL1: MOV R7,#02H; Delay time

DL: MOV R6,#020H

DL6: DJNZ R6,$

DJNZ R7,DL

RET

DSEG1: DB 3FH,06H,5BH,4FH,66H

DB 6DH,7DH,07H,7FH,6FH

END

S\_SEG SEGMENT STACK

DB 256 DUP(?)

S\_SEG ENDS

D\_SEG SEGMENT

COUNT DB 0 ;Timing unit, the initial value is 0

TENM DB ‘ 0’ ;10 minutes timing unit, initial value is 0

MINUTE DB ‘0:’ ;Time division unit, initial value is 0

TENS DB ‘0’ ;10 second timing unit, initial value is 0

SECOND DB ‘0.’ ;second timing unit, initial value is 0

HAOM DB ‘0’; 10 millisecond timing unit, the initial value is 0

HAO DB ’8’,’$’; millisecond timing unit, the initial value is 8

D\_SEG ENDS

C\_SEG SEGMENT

ASSUME CS:C\_SEG ,SS:S\_SEG

START: MOV AX,D\_SEG

MOV DS,AX

CLI; turn off the interrupt first to get INT1CH

MOV AX, 351CH; call 35H system function

INT 21H; return ES: BX = interrupt vector (segment: offset)

PUSH BX; save the original INT1CH interrupt vector in the stack

PUSH ES

STI; open interrupt to make the keyboard work

MOV AH,1; wait for button to be pressed

INT 16H

CLI; turn off interrupt

MOV DX, SEG TIMER; set new interrupt vector

MOV DS,DX

MOV DX,OFFSET TIMER; DS: DX = new interrupt vector (segment: offset)

MOV AX,251CH

INT 21H

STI; Open the interrupt again to make the keyboard and INT1CH work

CHECK: MOV AH ,1; check whether there is a key code

INT 16H

JZ DISPLAY1; if there is no code to read, jump to display (DISPLAY system reserved)]

MOV AH, 0; read it without code

INT 16H

CMP AL, 51H; is it "Q"

JE OVER; Yes, return to DOS

DISPLAY1: MOV AX, D\_SEG; No, just display

MOV DS, AX

ASSUME DS: D\_SEG

LEA DX, TENM; DS: DX=display string address

MOV AH, 9; display mm: ss.msms

INT 21H

JMP CHECK; return to CHECK, the loop continues

OVER: CLI

POP DS; retrieve the INT1CH original vector from the stack

POP DX

MOV AX, 251CH; set INT1CH, restore the original vector

INT 21H

STI; open interrupt

MOV AX, 4C00H; return to DOS

INT 21H

; The following is the interrupt service subroutine using INT 1CH

TIMER PROC FAR

PUSH AX

MOV AX, D\_SEG

MOV DS, AX

ASSUME DS :D\_SEG

INC COUNT; Increase the timing unit by 1

CMP COUNT, ‘2’; No to 110 milliseconds

JL EXIT; If not, return

MOV COUNT, ‘0’; when it arrives, the timing unit is cleared to 0

INC HAOM; 10 millisecond timing unit increases by 1

CMP HAOM ,’9’ ;to 1 second no

JLE EXIT; If not, return

INC SECOND; When it arrives, the second timing unit will increase by 1

MOV HAOM, ‘0’; 10 millisecond timing unit is cleared to 0

CMP SECOND, ‘9’; to 10 seconds no

JLE EXIT; If not, return

MOV SECOND, ‘0’; when it arrives, the second timing unit is cleared to 0

INC TENS; 10 seconds timing unit increases by 1

CMP TENS, ‘6’; to 60 seconds no

JL EXIT; If not, return

MOV TENS, ‘0’; to, the 10 second timing unit is cleared to 0

INC MINUTE; cleared to 0 by the time unit

CMP MINUTE, ‘9’; to 10 points no

JLE EXIT; If not, return

MOV MINUTE, ‘0’; when it arrives, the sub-timing unit is clear

INC TENM; 10 minutes time unit increases by 1

CMP TENM, ‘6’; to 60 points no

JL EXIT; If not, return

MOV TENM, ‘0’; when it arrives, the 10 minute timing unit is cleared to 0, and the timing is restarted

EXIT: POP AX

IRET

TIMER ENDP

C\_SEG ENDS

END START