ภาคผนวก

ภาคผนวก ก

ตัวโปรแกรมภาษาแอสเซมบลี (assembly) ซึ่งถูกบันทึกไว้ในหน่วยความจำชนิด ROM เบอร์ 27C64 ชื่อไฟล์ W1024.asm

```
;works with w1024.c

binum   equ 50h
bcdhi   equ 51h
bcdlo   equ 52h
d1_3    equ 53h
d1_2    equ 54h
d1_1    equ 55h
d1      equ 56h
binum1  equ 57h
binum2  equ 58h
count   equ 59h
rflag bit 20h.1
sflag bit 20h.0
aflag bit 20h.2

        org 0000h
        sjmp main
        org 0003h
        ljmp adc
        org 0023h
        ljmp serial

main:
    ;start initialization----------
      mov sp,#30h
```

```
mov tmod,#022h    ;25 ;init timer1 mode 2,timer 0 mode 1,leave t0 as is
push acc
      mov r6,#060h    ;
      ; wait for signal from PC where 255 is sent to start
    wait0:
      mov binum2,#05h ; these two lines help establish the beginnings
      acall send      ; ( send sig. to PC before recieving conditions)
      clr rflag
      mov binum,#0fh
    wait:
      jnb rflag,$  :for 255 from PC
      mov r0,binum
      cjne r0,#0ffh,noteq
      mov binum2,#0ah  ;send back A to acknowlege
      acall send
      pop acc
      pop psw
      ret  ; sjmp wait1
    noteq:
      mov binum2,#0ffh ;send $ff to PC
      acall send  ;asking for ready seignal ($FF)
      sjmp wait
;-------------------------------------------------------
get_time:
      push psw
      push acc
      clr rflag   ;wait for tl0 & th0
      jnb rflag,$ ;wait !!!
      clr rflag   ;get tl0
      mov count,binum ;store at count
;     jnb rflag,$  ;wait for th0
;     clr rflag    ;get it
;     mov r7,binum ;store at r7
      pop acc
```

```
      pop psw
      ret
;-------------------------------------------------------------
get_data:
      push psw
      push acc
      ;now for adc ----------
loo:      mov dptr,#4000h ;store data on RAM
loop:
       mov r6,count   ;try delay loop
       djnz r6,$     ;
      setb p3.3
      clr p3.3
      setb p3.3
       clr p3.4        ; to read
       mov a,p1
movx @dptr,a    ;a --> dptr
      inc dptr ;16 bit-inc
      mov r0,dph
      cjne r0,#044h,loop ;store 512*2=1024  bytes on RAM
setb p3.4
      pop acc
      pop psw
      ret
;-------------------------------------------------------------
send_data:
      push psw
      push acc
      mov dptr,#4000h  ;to send and display data
    again1:
      movx a,@dptr
      mov binum2,a
      acall send
      inc dptr
```

```
      mov r1,dph
      cjne r1,#044h,again1 ;512*2 = 1024 points
      pop acc
      pop psw
      ret
;-------------------------------------------------------------
delay: push acc
       mov r5,#0fh
L5:    djnz r5,$
       pop acc
       ret
;-------------------------------------------------------------
adc:    push acc
        push psw
        push dpl
        push dph
        mov r0,#60h     ;point at data
        mov dptr,#0e000h ;add. toactivate y7(top page), for 0804
     more:
        movx a,@dptr    ;read 0804
        mov @r0,a       ;store at $60++
        inc r0
        inc dptr
        pop dph
        pop dpl
        pop psw
        pop acc
        reti
;-------------------------------------------------------------
serial: push psw
       push acc
       jnb ti,nxt   ;is bit transmit set
       clr ti       ;clear it ,clear status of transmit
       clr sflag     ;(busy) flag
```

```
nxt:   jnb ri,bye   ;if it is not a transmit process
       mov a,sbuf   ;it should be a recieve process
       mov binum,a  ;get data
       clr ri     ;clear recieve flag
       setb rflag
bye:   pop acc
       pop psw
       reti
;-------------------------------------------------------------
send:  push psw
       push acc     ;save acc
       mov a,binum2
again: mov sbuf,a   ;send what is in a
       setb sflag   ;set flag for busy status
       clr rflag    ;clr reciev flag
slp:   jb sflag,slp  ;wait for interrupt to change flag
echo:  jnb rflag,$
       mov a,binum2
       cjne a,binum,again ;check with echo
       pop acc     ;o.k.
       pop psw
       ret
;-------------------------------------------------------------
       end
```