Laboratory Report

Laboratory Exercise No.:	6	**Date Performed:**	11/09/2022
Laboratory Exercise Title:	Parallel I/O Interfacing		
Name of Student:	Paul John Toral	**Document Version:**	1.0

Activity #1

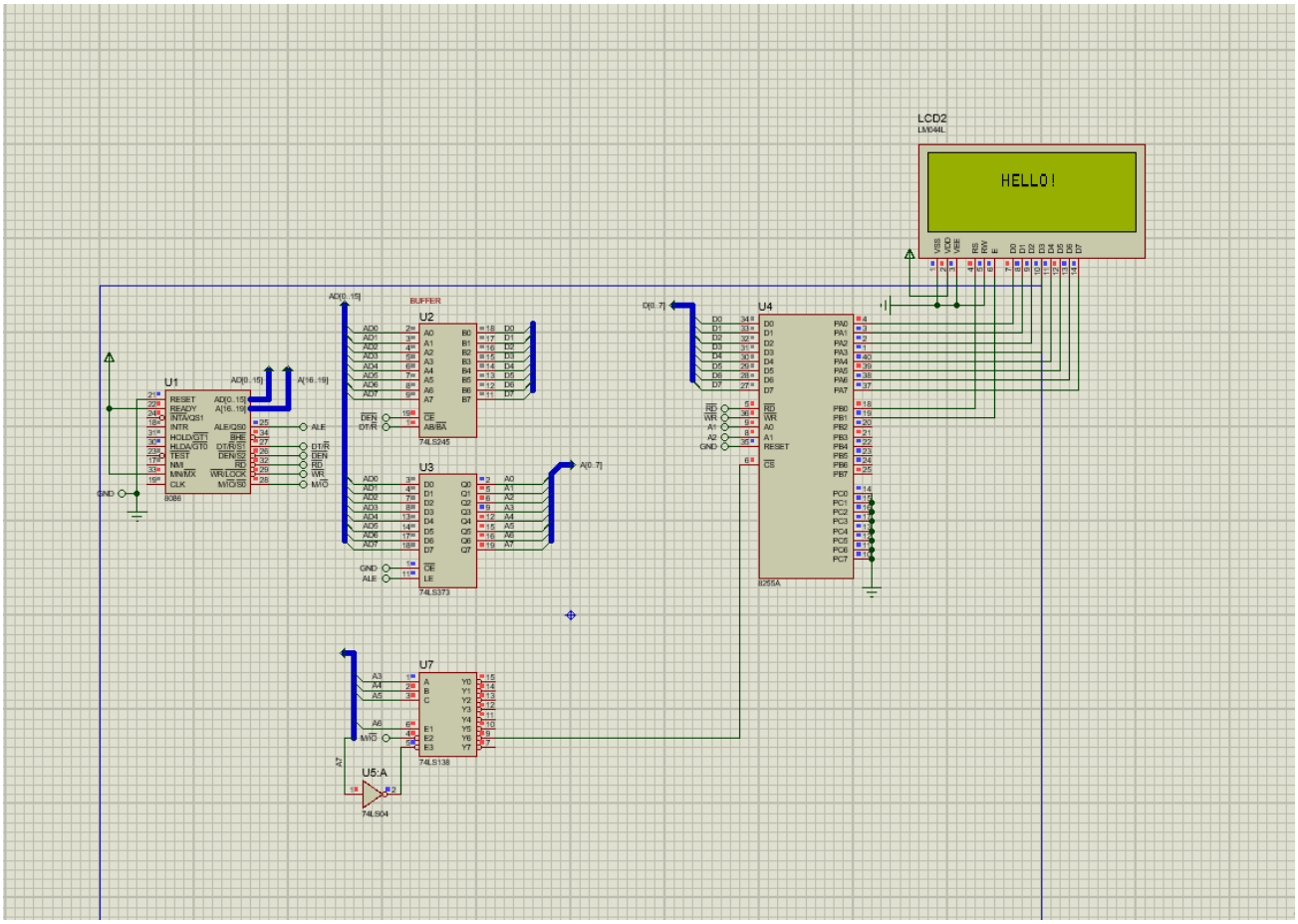

Figure 1: Schematic Diagram for Activity 1

Activity #2

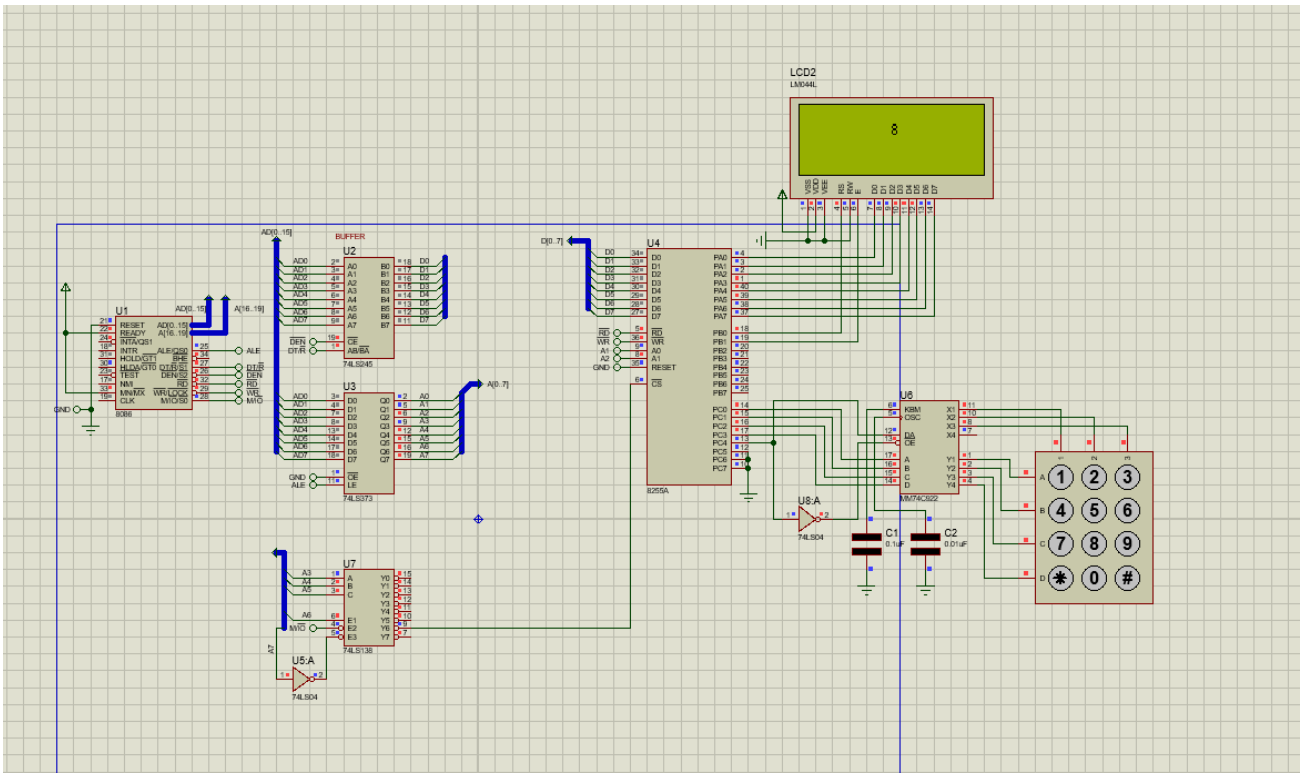

Figure 2: Schematic Diagram for Activity 2

Activity #3

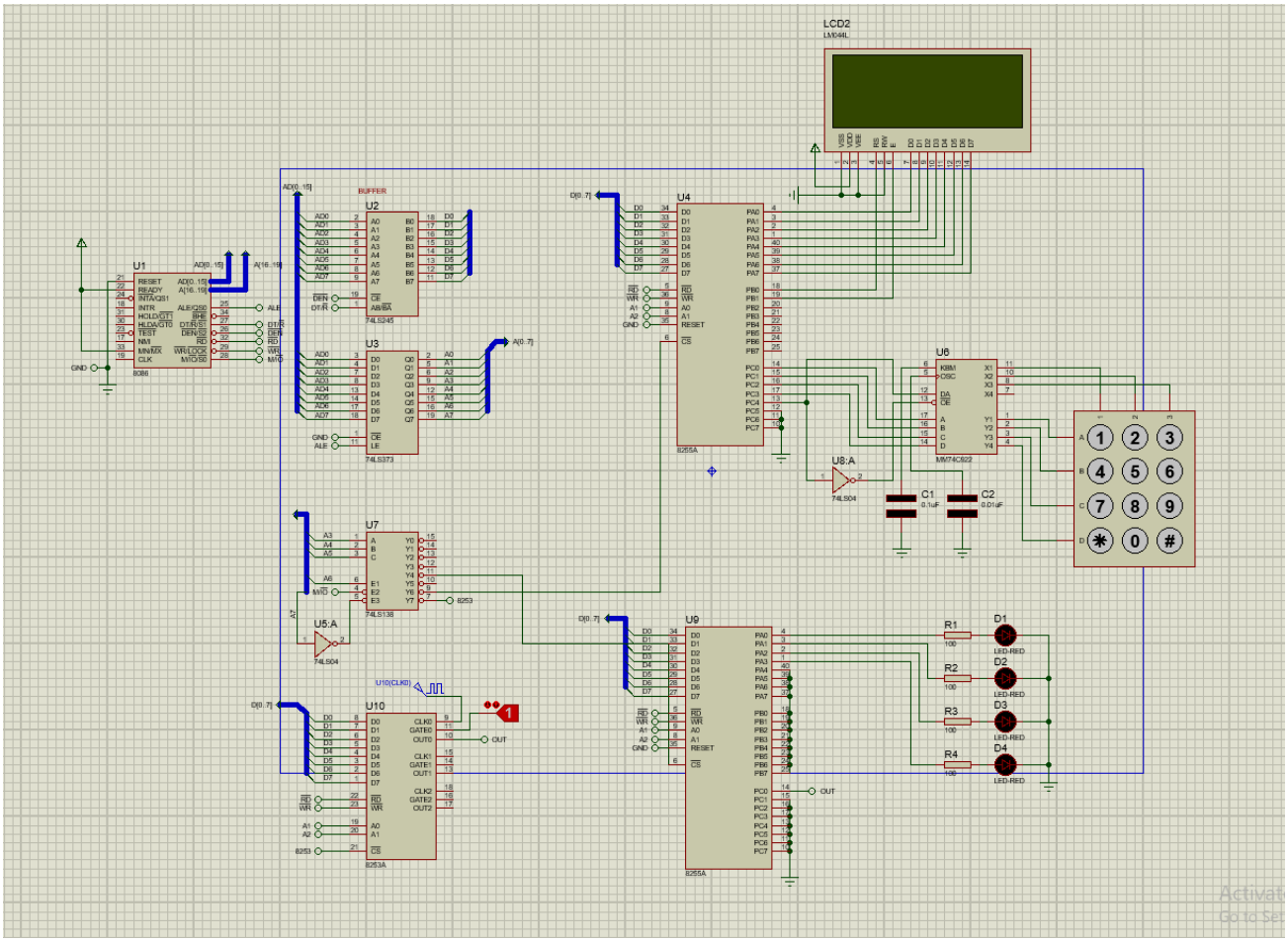

Figure 3: Schematic Diagram for Activity 3

Source Code #1

```
;===================================================================
; Main.asm file generated by New Project wizard
;
; Created:   Mon Oct 17 2022
; Processor: 8086
; Compiler:  MASM32
;
; Before starting simulation set Internal Memory Size
; in the 8086 model properties to 0x10000
;===================================================================

DATA SEGMENT
   PORTA EQU 0F0H;
   PORTB EQU 0F2H;
   PORTC EQU 0F4H;
   COM_REG EQU 0F6H;
   LCD_STR DB "HELLO!","$"
DATA ENDS


CODE    SEGMENT PUBLIC 'CODE'
        ASSUME CS:CODE
      MOV AX, DATA
      MOV DS, AX

      ORG 0000H

START:
   MOV DX, COM_REG ; set the address
   MOV AL, 10001001B
   OUT DX, AL ; send the command byte
   CALL INIT_LCD;
   LEA SI, LCD_STR

   MOV AL, 0C7H ; move cursor to 8th column of 2nd line
   CALL INST_CTRL ; send instruction to LCD

DISPLAY_STR:

   MOV AL, [SI]
   CMP AL, '$'
   JE EXIT
   CALL DATA_CTRL
   INC SI
   JMP DISPLAY_STR

INST_CTRL:
   PUSH AX ; preserve value of AL
   MOV DX, PORTA ; set port of LCD data bus (PORTA)
   OUT DX, AL ; write data in AL to PORTA
   MOV DX, PORTB ; set port of LCD control lines (PORTB)
   MOV AL, 02H ; E=1, RS=0 (access instruction reg)
   OUT DX, AL ; write data in AL to PORTB
```

```
   CALL DELAY_1MS ; delay for 1 ms
   MOV DX, PORTB ; set port of LCD control lines (PORTB)
   MOV AL, 00H ; E=0, RS=0
   OUT DX, AL ; write data in AL to PORTB
   POP AX ; restore value of AL
RET

DATA_CTRL:
   PUSH AX ; preserve value of AL
   MOV DX, PORTA ; set port of LCD data bus (PORTA)
   OUT DX, AL ; write data in AL to PORTA
   MOV DX, PORTB ; set port of LCD control lines (PORTB)
   MOV AL, 03H ; E=1, RS=1 (access data register)
   OUT DX, AL ; write data in AL to PORTB
   CALL DELAY_1MS ; delay for 1 ms
   MOV DX, PORTB ; set port of LCD control lines (PORTB)
   MOV AL, 01H ; E=0, RS=1
   OUT DX, AL ; write data in AL to PORTB
   POP AX ; restore value of AL
RET

INIT_LCD:
   MOV AL, 38H ; 8-bit interface, dual-line display
   CALL INST_CTRL ; write instruction to LCD
   MOV AL, 08H ; display off, cursor off, blink off
   CALL INST_CTRL ; write instruction to LCD
   MOV AL, 01H ; clear display
   CALL INST_CTRL ; write instruction to LCD
   MOV AL, 06H ; increment cursor, display shift off
   CALL INST_CTRL ; write instruction to LCD
   MOV AL, 0CH ; display on, cursor off, blink off
   CALL INST_CTRL ; write instruction to LCD
RET

DELAY_1MS:
   MOV BX, 02CAH
L1:
   DEC BX
   NOP
   JNZ L1
   RET
RET

EXIT:
CODE ENDS
END START
```

Source Code #2

```
;===================================================================
; Main.asm file generated by New Project wizard
;
; Created:   Mon Oct 17 2022
; Processor: 8086
; Compiler:  MASM32
;
; Before starting simulation set Internal Memory Size
; in the 8086 model properties to 0x10000
;===================================================================

DATA SEGMENT
   PORTA EQU 0F0H;
   PORTB EQU 0F2H;
   PORTC EQU 0F4H;
   COM_REG EQU 0F6H;

DATA ENDS


CODE    SEGMENT PUBLIC 'CODE'
        ASSUME CS:CODE
      MOV AX, DATA
      MOV DS, AX

      ORG 0000H




START:
   MOV DX, COM_REG ; set the address
   MOV AL, 10001001B
   OUT DX, AL ; send the command byte
   CALL INIT_LCD;


CHECK_DAVBL:

   MOV DX, PORTC
   IN  AL, DX
   TEST AL, 10H
   JZ CHECK_DAVBL
   IN AL, DX
   AND AL, 0FH
   PUSH AX
   CHECK_INPUT:
      CMP AL, 0CH          ; check if key pressed is *
       JE PRNT_AST
      CMP AL, 0EH          ; check if key pressed is #
      JE PRNT_OCTO
      CMP AL, 0DH          ; check if key pressed is 0
      JE PRNT_0
```

```
      CMP AL, 00H          ; check if key pressed is 1
      JE PRNT_1
      CMP AL, 01H          ; check if key pressed is 2
      JE PRNT_2
      CMP AL, 02H          ; check if key pressed is 3
      JE PRNT_3
      CMP AL, 04H          ; check if key pressed is 4
      JE PRNT_4
      CMP AL, 05H          ; check if key pressed is 5
      JE PRNT_5
      CMP AL, 06H          ; check if key pressed is 6
      JE PRNT_6
      CMP AL, 08H          ; check if key pressed is 7
      JE PRNT_7
      CMP AL, 09H          ; check if key pressed is 8
      JE PRNT_8
      CMP AL, 0AH          ; check if key pressed is 9
      JE PRNT_9
JMP CHECK_DAVBL

PRNT_AST:
   CALL CENTER
   MOV AL, '*'
   JMP PRINT_CHAR
PRNT_OCTO:
   CALL CENTER
   MOV AL, '#'
   JMP PRINT_CHAR
PRNT_0:
   CALL CENTER
   MOV AL, '0'
   JMP PRINT_CHAR
PRNT_1:
   CALL CENTER
   MOV AL, '1'
   JMP PRINT_CHAR
PRNT_2:
   CALL CENTER
   MOV AL, '2'
   JMP PRINT_CHAR
PRNT_3:
   CALL CENTER
   MOV AL, '3'
   JMP PRINT_CHAR
PRNT_4:
   CALL CENTER
   MOV AL, '4'
   JMP PRINT_CHAR
PRNT_5:
   CALL CENTER
   MOV AL, '5'
   JMP PRINT_CHAR
PRNT_6:
   CALL CENTER
```

```
   MOV AL, '6'
   JMP PRINT_CHAR
PRNT_7:
   CALL CENTER
   MOV AL, '7'
   JMP PRINT_CHAR
PRNT_8:
   CALL CENTER
   MOV AL, '8'
   JMP PRINT_CHAR
PRNT_9:
   CALL CENTER
   MOV AL, '9'
   JMP PRINT_CHAR


PRINT_CHAR:
   CALL DATA_CTRL
   POP AX
   JMP CHECK_DAVBL

CENTER:
   MOV AL, 0CAH
   CALL INST_CTRL
 RET

INST_CTRL:
   PUSH AX ; preserve value of AL
   MOV DX, PORTA ; set port of LCD data bus (PORTA)
   OUT DX, AL ; write data in AL to PORTA
   MOV DX, PORTB ; set port of LCD control lines (PORTB)
   MOV AL, 02H ; E=1, RS=0 (access instruction reg)
   OUT DX, AL ; write data in AL to PORTB
   CALL DELAY_1MS ; delay for 1 ms
   MOV DX, PORTB ; set port of LCD control lines (PORTB)
   MOV AL, 00H ; E=0, RS=0
   OUT DX, AL ; write data in AL to PORTB
   POP AX ; restore value of AL
RET

DATA_CTRL:
   PUSH AX ; preserve value of AL
   MOV DX, PORTA ; set port of LCD data bus (PORTA)
   OUT DX, AL ; write data in AL to PORTA
   MOV DX, PORTB ; set port of LCD control lines (PORTB)
   MOV AL, 03H ; E=1, RS=1 (access data register)
   OUT DX, AL ; write data in AL to PORTB
   CALL DELAY_1MS ; delay for 1 ms
   MOV DX, PORTB ; set port of LCD control lines (PORTB)
   MOV AL, 01H ; E=0, RS=1
   OUT DX, AL ; write data in AL to PORTB
   POP AX ; restore value of AL
RET
```

```
INIT_LCD:
   MOV AL, 38H ; 8-bit interface, dual-line display
   CALL INST_CTRL ; write instruction to LCD
   MOV AL, 08H ; display off, cursor off, blink off
   CALL INST_CTRL ; write instruction to LCD
   MOV AL, 01H ; clear display
   CALL INST_CTRL ; write instruction to LCD
   MOV AL, 06H ; increment cursor, display shift off
   CALL INST_CTRL ; write instruction to LCD
   MOV AL, 0CH ; display on, cursor off, blink off
   CALL INST_CTRL ; write instruction to LCD
RET

DELAY_1MS:
   MOV BX, 02CAH
L1:
   DEC BX
   NOP
   JNZ L1
   RET
RET

EXIT:
CODE ENDS
END START
```

Source Code #3

```
;===================================================================
; Main.asm file generated by New Project wizard
;
; Created:   Thu Nov 10 2022
; Processor: 8086
; Compiler:  MASM32
;
; Before starting simulation set Internal Memory Size
; in the 8086 model properties to 0x10000
;===================================================================

DATA SEGMENT
   PORTA EQU 0F0H;
   PORTB EQU 0F2H;
   PORTC EQU 0F4H;
   COM_REG EQU 0F6H;

   PORTA2 EQU 0E0H;
   PORTB2 EQU 0E2H;
   PORTC2 EQU 0E4H;
   COM_REG2 EQU 0E6H;

   LOAD_CTR0 EQU 0F8H      ; counter 0 address
   TIMER_REG EQU 0FEH      ; 8253 Command register address

   DISPENSE_PROMPT DB "Dispensing...","$"
   READY DB "Get Your Drink :D","$"
   L_CK DB "[1] Coke Large","$"
   M_CK DB "[2] Coke Medium","$"
   L_SPR DB "[3] Sprite Large","$"
   M_SPR DB "[4] Sprite Medium","$"
DATA ENDS


CODE    SEGMENT PUBLIC 'CODE'
        ASSUME CS:CODE
      MOV AX, DATA
      MOV DS, AX
      ORG 0000H




START:
   MOV DX, COM_REG ; set the address for 1st 8255
   MOV AL, 10001001B
   OUT DX, AL ; send the command byte

   MOV DX, COM_REG2 ; set the address for 2nd 8255
   MOV AL, 10001001B
   OUT DX, AL ; send the command byte
   CALL INIT_LCD;
```

```
   MOV DX, TIMER_REG
   MOV AL, 00111000B
   OUT DX, AL
   DISPLAY:
   CALL CLEAR_SCREEN
   ;printing first line;
   LEA SI, L_CK
   MOV AL, 081H
   CALL INST_CTRL
   CALL PRINT_STR
   ;printing  second line;
   LEA SI, M_CK
   MOV AL, 0C1H
   CALL INST_CTRL
   CALL PRINT_STR
   ;printing third line;
   LEA SI, L_SPR
   MOV AL, 095H
   CALL INST_CTRL
   CALL PRINT_STR
   ;printing fourth line;
   LEA SI, M_SPR
   MOV AL, 0D5H
   CALL INST_CTRL
   CALL PRINT_STR



CHECK_DAVBL:

   MOV DX, PORTC
   IN  AL, DX
   TEST AL, 10H
   JZ CHECK_DAVBL
   IN AL, DX
   AND AL, 0FH
   PUSH AX
   CHECK_INPUT:
      CMP AL, 00H          ; check if key pressed is 0
      JE GET_L_CK
      CMP AL, 01H          ; check if key pressed is 1
      JE GET_M_CK
      CMP AL, 02H          ; check if key pressed is 2
      JE GET_L_SPR
      CMP AL, 04H          ; check if key pressed is 3
      JE GET_M_SPR
JMP CHECK_DAVBL


GET_L_CK:
   CALL CLEAR_SCREEN
   LEA SI,  DISPENSE_PROMPT
   CALL LINE_2
   CALL PRINT_STR
```

```
   MOV CL, 07H
    JMP LED_1
GET_M_CK:
    CALL CLEAR_SCREEN
   LEA SI,  DISPENSE_PROMPT
   CALL LINE_2
   CALL PRINT_STR
   MOV CL, 04H
    JMP LED_2

GET_L_SPR  :
    CALL CLEAR_SCREEN
   LEA SI,  DISPENSE_PROMPT
   CALL LINE_2
   CALL PRINT_STR
   MOV CL, 07H
    JMP LED_3

GET_M_SPR:
    CALL CLEAR_SCREEN
   LEA SI,  DISPENSE_PROMPT
   CALL LINE_2
   CALL PRINT_STR
   MOV CL, 04H
    JMP LED_4



LED_1:
   MOV DX, PORTA2
   MOV AL, 00000001B
   OUT DX,AL
   CALL DISPLAY_COUNT
   CALL FINISHED
   POP AX
   JMP CHECK_DAVBL

LED_2:
   MOV DX, PORTA2
   MOV AL, 00000010B
   OUT DX,AL
   CALL DISPLAY_COUNT
   CALL FINISHED
   POP AX
   JMP CHECK_DAVBL

 LED_3:
   MOV DX, PORTA2
   MOV AL, 00000100B
   OUT DX,AL
   CALL DISPLAY_COUNT
   CALL FINISHED
   POP AX
   JMP CHECK_DAVBL
```

```
 LED_4:
   MOV DX, PORTA2
   MOV AL, 00001000B
   OUT DX,AL
   CALL DISPLAY_COUNT
   CALL FINISHED
   POP AX
   JMP CHECK_DAVBL
PRINT_STR:
      MOV AX, [SI]    ; store character
      CMP AL, '$'     ; check if end
      JE DELAY_1MS    ; jump if end
      CALL DATA_CTRL  ; display character
      INC SI          ; increment to next character in the array
      JMP PRINT_STR        ; loop back procedure

DISPLAY_COUNT:
      CALL LINE_3
      MOV AL, 030H
      ADD AL, CL
      CALL DATA_CTRL
      MOV AL, 's'
      CALL DATA_CTRL
      CALL DELAY_1S
      DEC CL
      CMP CL, 00H
      JNE DISPLAY_COUNT
RET

FINISHED:
   CALL CLEAR_SCREEN
   LEA SI, READY
   MOV AL, 0C1H
   CALL INST_CTRL
   CALL PRINT_STR
   MOV DX, PORTA2
   MOV AL, 00000000B
   OUT DX,AL
   CALL DELAY_1S
   JMP DISPLAY
RET
CENTER:
   MOV AL, 0CAH
   CALL INST_CTRL
 RET

LINE_2:
   MOV AL, 0C4H
   CALL INST_CTRL
 RET

 LINE_3:
   MOV AL, 09EH
```

```
   CALL INST_CTRL
 RET



INST_CTRL:
   PUSH AX ; preserve value of AL
   MOV DX, PORTA ; set port of LCD data bus (PORTA)
   OUT DX, AL ; write data in AL to PORTA
   MOV DX, PORTB ; set port of LCD control lines (PORTB)
   MOV AL, 02H ; E=1, RS=0 (access instruction reg)
   OUT DX, AL ; write data in AL to PORTB
   CALL DELAY_1MS ; delay for 1 ms
   MOV DX, PORTB ; set port of LCD control lines (PORTB)
   MOV AL, 00H ; E=0, RS=0
   OUT DX, AL ; write data in AL to PORTB
   POP AX ; restore value of AL
RET

DATA_CTRL:
   PUSH AX ; preserve value of AL
   MOV DX, PORTA ; set port of LCD data bus (PORTA)
   OUT DX, AL ; write data in AL to PORTA
   MOV DX, PORTB ; set port of LCD control lines (PORTB)
   MOV AL, 03H ; E=1, RS=1 (access data register)
   OUT DX, AL ; write data in AL to PORTB
   CALL DELAY_1MS ; delay for 1 ms
   MOV DX, PORTB ; set port of LCD control lines (PORTB)
   MOV AL, 01H ; E=0, RS=1
   OUT DX, AL ; write data in AL to PORTB
   POP AX ; restore value of AL
RET

INIT_LCD:
   MOV AL, 38H ; 8-bit interface, dual-line display
   CALL INST_CTRL ; write instruction to LCD
   MOV AL, 08H ; display off, cursor off, blink off
   CALL INST_CTRL ; write instruction to LCD
   MOV AL, 01H ; clear display
   CALL INST_CTRL ; write instruction to LCD
   MOV AL, 06H ; increment cursor, display shift off
   CALL INST_CTRL ; write instruction to LCD
   MOV AL, 0CH ; display on, cursor off, blink off
   CALL INST_CTRL ; write instruction to LCD
RET
;==========1 milisecond timer ============;
DELAY_1MS:
   MOV BX, 02CAH
L1:
   DEC BX
   NOP
   JNZ L1
   RET
```

```
RET
;==========1 second timer ============;
DELAY_1S:
     MOV DX, LOAD_CTR0
      MOV AL, 0A0H
      OUT DX, AL
      MOV AL, 0FH
      OUT DX, AL

      TIMER:
       MOV DX, PORTC2
       IN AX, DX
       MOV AH, 00H
       AND AL, 00000001B ; MASK OFF
       CMP AL, 00H
       JNE TIMER
RET

CLEAR_SCREEN:
   MOV AL, 01H
   CALL INST_CTRL
 RET
EXIT:
CODE ENDS
END START
```