**Department of Computer Engineering** 

Digital Hardware Systems

*C CpE 3104 - Microprocessors*

**Laboratory Report**

|  |  |  |  |
| --- | --- | --- | --- |
| **Laboratory Exercise No.:** | 6 | **Date Performed:** | November 10, 2022 |
| **Laboratory Exercise Title:** | Parallel I/O Devices Interfacing | | |
| **Name of Student:** | Dumalagan, Danica Marie A. | **Document Version:** | v1.2.2 |

**Activity #1**

Chart, bar chart

Description automatically generated

Fig. 1.1. Schematic of parallel I/O devices interfaced to the 8086 configured to display “HELLO!” in the middle of the second line of the LCD constructed in Proteus Professional (v8.13).

; DUMALAGAN\_LE6-1

DATA SEGMENT

PORTA EQU 0F0H ; PORTA address

PORTB EQU 0F2H ; PORTB address

PORTC EQU 0F4H ; PORTC address

COM\_REG EQU 0F6H ; Command Register Address

; LCD message strings

MSG1 DB "HELLO!","$"

DATA ENDS

CODE SEGMENT PUBLIC 'CODE'

ASSUME CS:CODE

ORG 0000H

MOV SI, 0000H

XOR AX, AX

XOR BX, BX

XOR DX, DX

START:

; Configuring the 8255

MOV DX, COM\_REG ; set the address

MOV AL, 10001001B ; command byte

OUT DX, AL ; send the command byte

CALL INIT\_LCD ; initialize LCD

MOV AL, 0C7H ; set cursor location

CALL INST\_CTRL ; send instruction to LCD

LEA SI, MSG1 ; load string to display

CALL DISP\_STR ; call module to display string

ENDLESS:

JMP ENDLESS

; MODULE: Initialize LCD

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

; MODULE: Send instruction to LCD

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

; MODULE: Send data to LCD

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

; MODULE: Display string

DISP\_STR:

MOV AX, [SI]

CMP AL, '$'

JE EXIT

CALL DATA\_CTRL

INC SI

JMP DISP\_STR

RET

; MODULE: Delay for 1 millisecond

DELAY\_1MS:

MOV BX, 02CAH

L1:

DEC BX

NOP

JNZ L1

RET

; MODULE: Exit here

EXIT:

RET

CODE ENDS

END START

Code 1. Assembly program to display “HELLO!” In the middle of the second line of the LCD.

A picture containing chart

Description automatically generated

Fig. 1.2. Sample running simulation of Fig. 1.1.

**Activity #2**

Chart

Description automatically generated

Fig. 2.1. Schematic of parallel I/O devices interfaced to the 8086 configured to display the key being pressed in the middle of the LCD constructed in Proteus Professional (v8.13).

; DUMALAGAN\_LE6-2

DATA SEGMENT

PORTA EQU 0F0H ; PORTA address

PORTB EQU 0F2H ; PORTB address

PORTC EQU 0F4H ; PORTC address

COM\_REG EQU 0F6H ; Command Register Address

DATA ENDS

CODE SEGMENT PUBLIC 'CODE'

ASSUME CS:CODE

ORG 0000H

MOV SI, 0000H

XOR AX, AX

XOR BX, BX

XOR DX, DX

START:

; Configuring the 8255

MOV DX, COM\_REG ; set the address

MOV AL, 10001001B ; command byte

OUT DX, AL ; send the command byte

CALL INIT\_LCD ; initialize LCD

MOV AL, 0CAH ; set cursor location

CALL INST\_CTRL ; send instruction to LCD

CALL CHECK\_DAVBL ; check DAVBL

; MODULE: Check DAVBL

CHECK\_DAVBL:

MOV DX, PORTC ; set port of DAVBL(PORTC)

IN AL, DX ; read PORTC

TEST AL, 10H ; check if DAVBL is high

JZ CHECK\_DAVBL ; if low then check again

IN AL, DX ; read 4-bit keypad data

AND AL, 0FH ; mask upper nibble

CMP AL, 00H ; check if key pressed is 1 (00H)

JE D1 ; display 1

CMP AL, 01H ; check if key pressed is 2 (01H)

JE D2 ; display 2

CMP AL, 02H ; check if key pressed is 3 (02H)

JE D3 ; display 3

CMP AL, 04H ; check if key pressed is 4 (04H)

JE D4 ; display 4

CMP AL, 05H ; check if key pressed is 5 (05H)

JE D5 ; display 5

CMP AL, 06H ; check if key pressed is 6 (06H)

JE D6 ; display 6

CMP AL, 08H ; check if key pressed is 7 (08H)

JE D7 ; display 7

CMP AL, 09H ; check if key pressed is 8 (09H)

JE D8 ; display 8

CMP AL, 0AH ; check if key pressed is 9 (0AH)

JE D9 ; display 9

CMP AL, 0CH ; check if key pressed is \* (0CH)

JE D\_ASTERISK ; display \*

CMP AL, 0DH ; check if key pressed is 0 (0DH)

JE D0 ; display 0

CMP AL, 0EH ; check if key pressed is # (0EH)

JE D\_POUND ; display #

CALL DELAY\_1MS

JMP CHECK\_DAVBL

; MODULES to display the keypad key pressed

D1:

MOV AL, 0CAH ; set cursor location

CALL INST\_CTRL ; send instruction to LCD

MOV AL, '1' ; display ‘1‘

JMP CONT

D2:

MOV AL, 0CAH ; set cursor location

CALL INST\_CTRL ; send instruction to LCD

MOV AL, '2' ; display ‘2‘

JMP CONT

D3:

MOV AL, 0CAH ; set cursor location

CALL INST\_CTRL ; send instruction to LCD

MOV AL, '3' ; display ‘3‘

JMP CONT

D4:

MOV AL, 0CAH ; set cursor location

CALL INST\_CTRL ; send instruction to LCD

MOV AL, '4' ; display ‘4‘

JMP CONT

D5:

MOV AL, 0CAH ; set cursor location

CALL INST\_CTRL ; send instruction to LCD

MOV AL, '5' ; display ‘5‘

JMP CONT

D6:

MOV AL, 0CAH ; set cursor location

CALL INST\_CTRL ; send instruction to LCD

MOV AL, '6' ; display ‘6‘

JMP CONT

D7:

MOV AL, 0CAH ; set cursor location

CALL INST\_CTRL ; send instruction to LCD

MOV AL, '7' ; display ‘7‘

JMP CONT

D8:

MOV AL, 0CAH ; set cursor location

CALL INST\_CTRL ; send instruction to LCD

MOV AL, '8' ; display ‘8‘

JMP CONT

D9:

MOV AL, 0CAH ; set cursor location

CALL INST\_CTRL ; send instruction to LCD

MOV AL, '9' ; display ‘9‘

JMP CONT

D0:

MOV AL, 0CAH ; set cursor location

CALL INST\_CTRL ; send instruction to LCD

MOV AL, '0' ; display ‘0‘

JMP CONT

D\_ASTERISK:

MOV AL, 0CAH ; set cursor location

CALL INST\_CTRL ; send instruction to LCD

MOV AL, '\*' ; display ‘\*‘

JMP CONT

D\_POUND:

MOV AL, 0CAH ; set cursor location

CALL INST\_CTRL ; send instruction to LCD

MOV AL, '#' ; display ‘#‘

JMP CONT

CONT:

CALL DATA\_CTRL

CALL DELAY\_1MS

JMP CHECK\_DAVBL

; MODULE: Endless loop

ENDLESS:

JMP ENDLESS

; MODULE: Initialize LCD

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

; MODULE: Send instruction to LCD

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

; MODULE: Send data to LCD

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

; MODULE: Delay for 1 millisecond

DELAY\_1MS:

MOV BX, 02CAH

L1:

DEC BX

NOP

JNZ L1

RET

; MODULE: Exit here

EXIT:

RET

CODE ENDS

END START

Code 2. Assembly program to display the key being pressed in the middle of the LCD.

Chart

Description automatically generated

Fig. 2.2. Sample running simulation of Fig. 2.1 on startup.

Chart, bar chart

Description automatically generated

Fig. 2.3. Sample running simulation of Fig. 2.1 when “8” is pressed on the simulation keypad.

Chart, bar chart

Description automatically generated

Fig. 2.4. Sample running simulation of Fig. 2.1 when “#” is pressed on the simulation keypad.

**Activity #3**

Diagram, schematic

Description automatically generated

Fig. 3.1. Schematic of parallel I/O devices interfaced to the 8086 of an LCD and numeric keypad controls a soft drink dispenser constructed in Proteus Professional (v8.13).

; DUMALAGAN\_LE6-3

DATA SEGMENT

; Port addresses of the first 8255

PORTA EQU 0F0H ; PORTA address of the first 8255

PORTB EQU 0F2H ; PORTB address of the first 8255

PORTC EQU 0F4H ; PORTC address of the first 8255

COM\_REG1 EQU 0F6H ; Command Register Address of the first 8255

; Port addresses of the second 8255

PORTD EQU 0F8H ; PORTA address of the second 8255

PORTE EQU 0FAH ; PORTB address of the second 8255

PORTF EQU 0FCH ; PORTC address of the second 8255

COM\_REG2 EQU 0FEH ; Command Register Address of the second 8255

; 8253 timer addresses

PORT\_T EQU 0E8H

COM\_REGT EQU 0EEH

; Message prompts

MENU1 DB "[1] Coke Large","$"

MENU2 DB "[2] Coke Medium","$"

MENU3 DB "[3] Sprite Large","$"

MENU4 DB "[4] Sprite Medium","$"

DMSG1 DB "Dispensing...","$"

DMSG2 DB " S","$"

DMSG3 DB "Enjoy your drink!","$"

DATA ENDS

CODE SEGMENT PUBLIC 'CODE'

ASSUME CS:CODE

ORG 0000H

MOV SI, 0000H

XOR AX, AX

XOR BX, BX

XOR DX, DX

START:

; Configuring the first 8255

MOV DX, COM\_REG1 ; set the address

MOV AL, 89H ; command byte

OUT DX, AL ; send the command byte

; Configuring the second 8255

MOV DX, COM\_REG2 ; set the address

MOV AL, 082H ; command byte

OUT DX, AL ; send the command byte

; Configuring the 8253 timer

MOV DX, COM\_REGT ; set the address

MOV AL, 038H ; command byte

OUT DX, AL ; send the command byte

CALL INIT\_LCD ; initialize LCD

CALL SHOW\_MENU ; show menu options

CALL CHECK\_DAVBL ; check DAVBL

JMP ENDLESS

; MODULE: Show menu options

SHOW\_MENU:

; Line 1 Menu

MOV AL, 080H ; set cursor location

CALL INST\_CTRL ; send instruction to LCD

LEA SI, MENU1 ; load string message to be dsiplayed

CALL DISP\_STR ; display string

XOR AX, AX

; Line 2 Menu

MOV AL, 0C0H ; set cursor location

CALL INST\_CTRL ; send instruction to LCD

LEA SI, MENU2 ; load string message to be dsiplayed

CALL DISP\_STR ; display string

XOR AX, AX

; Line 3 Menu

MOV AL, 094H ; set cursor location

CALL INST\_CTRL ; send instruction to LCD

LEA SI, MENU3 ; load string message to be dsiplayed

CALL DISP\_STR ; display string

XOR AX, AX

; Line 4 Menu

MOV AL, 0D4H ; set cursor location

CALL INST\_CTRL ; send instruction to LCD

LEA SI, MENU4 ; load string message to be dsiplayed

CALL DISP\_STR ; display string

XOR AX, AX

RET

; MODULE: Check DAVBL

CHECK\_DAVBL:

MOV DX, PORTC ; set port of DAVBL(PORTC)

IN AL, DX ; read PORTC

TEST AL, 10H ; check if DAVBL is high

JZ CHECK\_DAVBL ; if low then check again

IN AL, DX ; read 4-bit keypad data

AND AL, 0FH ; mask upper nibble

CMP AL, 00H ; check if key pressed is 1 (00H)

JE COKE\_L ; dispense Coke Large

CMP AL, 01H ; check if key pressed is 2 (01H)

JE COKE\_M ; dispense Coke Medium

CMP AL, 02H ; check if key pressed is 3 (02H)

JE SPRITE\_L ; dispense Sprite Large

CMP AL, 04H ; check if key pressed is 4 (04H)

JE SPRITE\_M ; dispense Sprite Medium

CALL DELAY\_1MS

JMP CHECK\_DAVBL

; MODULES for each menu option

COKE\_L:

CALL DISPENSING ; display "Dispensing..."

MOV CX, 07H ; set timer to 7 seconds

MOV DX, PORTD

MOV AL, 0001B ; set target LED to logic-1

CALL LED\_CTRL ; send instruction to LED

JMP START ; go back to start function

RET

COKE\_M:

CALL DISPENSING ; display "Dispensing..."

MOV CX, 04H ; set timer to 4 seconds

MOV DX, PORTD

MOV AL, 0010B ; set target LED to logic-1

CALL LED\_CTRL ; send instruction to LED

JMP START ; go back to start function

RET

SPRITE\_L:

CALL DISPENSING ; display "Dispensing..."

MOV CX, 07H ; set timer to 7 seconds

MOV DX, PORTD

MOV AL, 0100B ; set target LED to logic-1

CALL LED\_CTRL ; send instruction to LED

JMP START ; go back to start function

RET

SPRITE\_M:

CALL DISPENSING ; display "Dispensing..."

MOV CX, 04H ; set timer to 4 seconds

MOV DX, PORTD

MOV AL, 1000B ; set target LED to logic-1

CALL LED\_CTRL ; send instruction to LED

JMP START ; go back to start function

RET

; MODULE: Dispensing

DISPENSING:

CALL INIT\_LCD ; initialize LCD

MOV AL, 0C4H ; set cursor location

CALL INST\_CTRL ; send instruction to LCD

LEA SI, DMSG1 ; load string message to be displayed

CALL DISP\_STR ; display "Dispensing..."

MOV AL, 09EH ; set cursor location

CALL INST\_CTRL ; send instruction to LCD

LEA SI, DMSG2 ; load string message to be displayed

CALL DISP\_STR ; display "S"

XOR CX, CX

RET

; MODULE: Control LED

LED\_CTRL:

OUT DX, AL ; turn LED on

; display seconds value

CMP CX, 07H

JE D7

CMP CX, 06H

JE D6

CMP CX, 05H

JE D5

CMP CX, 04H

JE D4

CMP CX, 03H

JE D3

CMP CX, 02H

JE D2

CMP CX, 01H

JE D1

RESUME:

CALL DELAY\_1S

DEC CX

CMP CX, 00H

JNZ LED\_CTRL

CALL D\_ENJOY

RET

; MODULE: Dispensing timer

DELAY\_1S:

MOV DX, PORT\_T ; access 8253 timer

MOV AL, 0A0H

OUT DX, AL

MOV AL, 0FH

OUT DX, AL

LOCK\_INPUT:

MOV DX, PORTE

IN AX, DX

XOR AH, AH

AND AL, 01H

CMP AL, 00H ; checks if remaining time is 0

JNE LOCK\_INPUT

RET

; MODULES to display the number in the number of seconds remaining

D1:

MOV AL, 09EH ; set cursor location

CALL INST\_CTRL ; send instruction to LCD

MOV AL, '1' ; display ‘1‘

JMP CONT

D2:

MOV AL, 09EH ; set cursor location

CALL INST\_CTRL ; send instruction to LCD

MOV AL, '2' ; display ‘2‘

JMP CONT

D3:

MOV AL, 09EH ; set cursor location

CALL INST\_CTRL ; send instruction to LCD

MOV AL, '3' ; display ‘3‘

JMP CONT

D4:

MOV AL, 09EH ; set cursor location

CALL INST\_CTRL ; send instruction to LCD

MOV AL, '4' ; display ‘4‘

JMP CONT

D5:

MOV AL, 09EH ; set cursor location

CALL INST\_CTRL ; send instruction to LCD

MOV AL, '5' ; display ‘5‘

JMP CONT

D6:

MOV AL, 09EH ; set cursor location

CALL INST\_CTRL ; send instruction to LCD

MOV AL, '6' ; display ‘6‘

JMP CONT

D7:

MOV AL, 09EH ; set cursor location

CALL INST\_CTRL ; send instruction to LCD

MOV AL, '7' ; display ‘7‘

JMP CONT

CONT:

CALL DATA\_CTRL

CALL DELAY\_1MS

JMP RESUME

; MODULE: Display "Enjoy your drink!"

D\_ENJOY:

CALL INIT\_LCD ; initialize LCD

MOV AL, 0C2H ; set cursor location

CALL INST\_CTRL ; send instruction to LCD

LEA SI, DMSG3 ; load string message to be displayed

CALL DISP\_STR ; display "Enjoy your drink!"

CALL DELAY\_1S

RET

; MODULE: Endless loop

ENDLESS:

JMP ENDLESS

; MODULE: Initialize LCD

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 CLR\_LCD ; 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

; MODULE: Clear LCD

CLR\_LCD:

MOV AL, 01H ; clear display

CALL INST\_CTRL ; write instruction to LCD

; MODULE: Send instruction to LCD

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

; MODULE: Send data to LCD

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

; MODULE: Display string

DISP\_STR:

MOV AX, [SI]

CMP AL, '$'

JE EXIT

CALL DATA\_CTRL

INC SI

JMP DISP\_STR

RET

; MODULE: Delay for 1 millisecond

DELAY\_1MS:

MOV BX, 02CAH

L1:

DEC BX

NOP

JNZ L1

RET

; MODULE: Exit here

EXIT:

RET

CODE ENDS

END START

Code 3. Assembly program of an LCD and numeric keypad controls a soft drink dispenser.

Diagram, schematic

Description automatically generated

Fig. 3.2. Sample running simulation of Fig. 3.1 on startup.

Diagram, schematic

Description automatically generated

Fig. 3.3. Sample running simulation of Fig. 3.1 when “9” is pressed on the simulation keypad.

Diagram, schematic

Description automatically generated

Fig. 3.4. Sample running simulation of Fig. 3.1 when “3” is pressed on the simulation keypad.

Diagram, schematic

Description automatically generated

Fig. 3.5. Sample running simulation of Fig. 3.1 when “2” is pressed on the simulation keypad.

Diagram, schematic

Description automatically generated

Fig. 3.6. Sample running simulation of Fig. 3.1 upon dispensing completion.