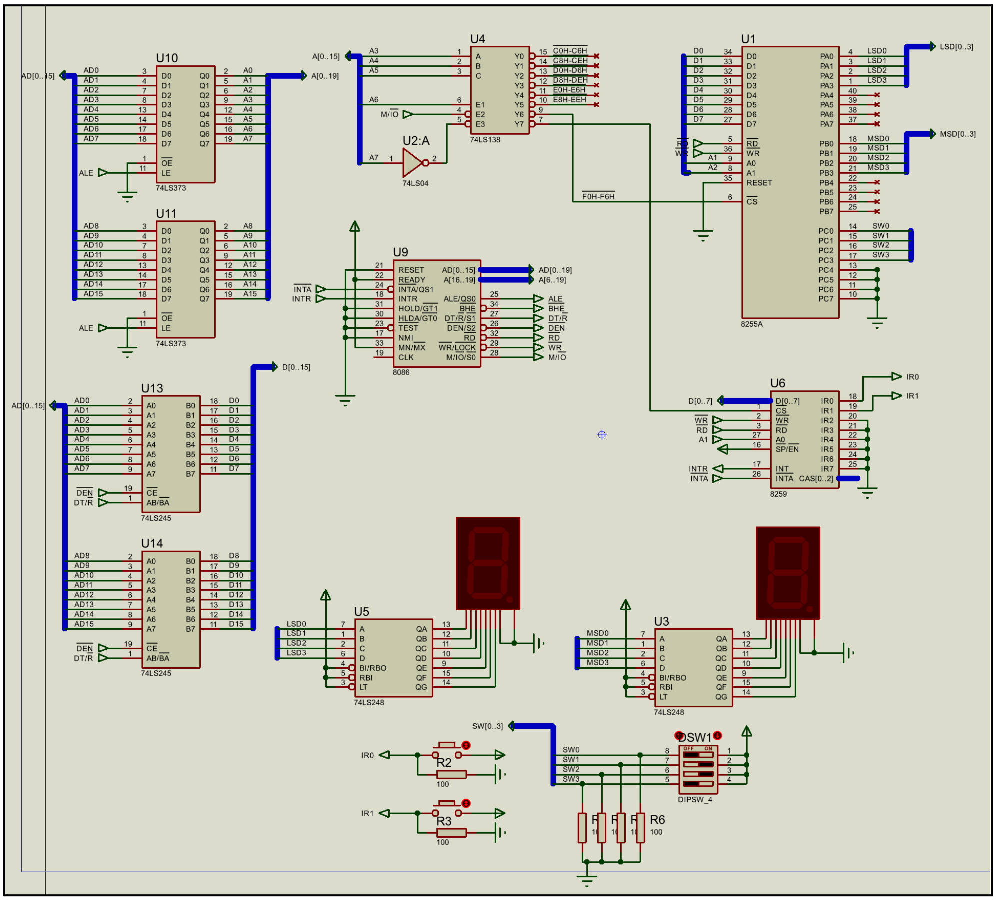
Department of Computer Engineering
Digital Hardware Systems
CpE 3104 - Microprocessors

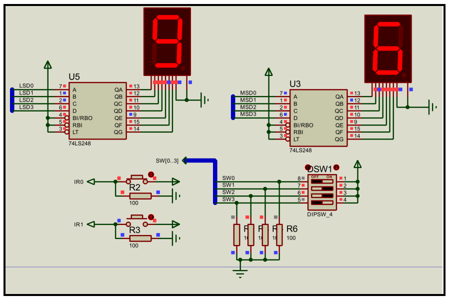
Laboratory Report 6

Laboratory Exercise No.:	7	**Date Performed:**	11/29/2025
Laboratory Exercise Title:	Hardware Interrupt Interfacing		
Name of Student(s):	Christian Jay Gallardo Jhon Fil Tizon	**Document Version:**	1

Activity #1

Proteus Simulation:





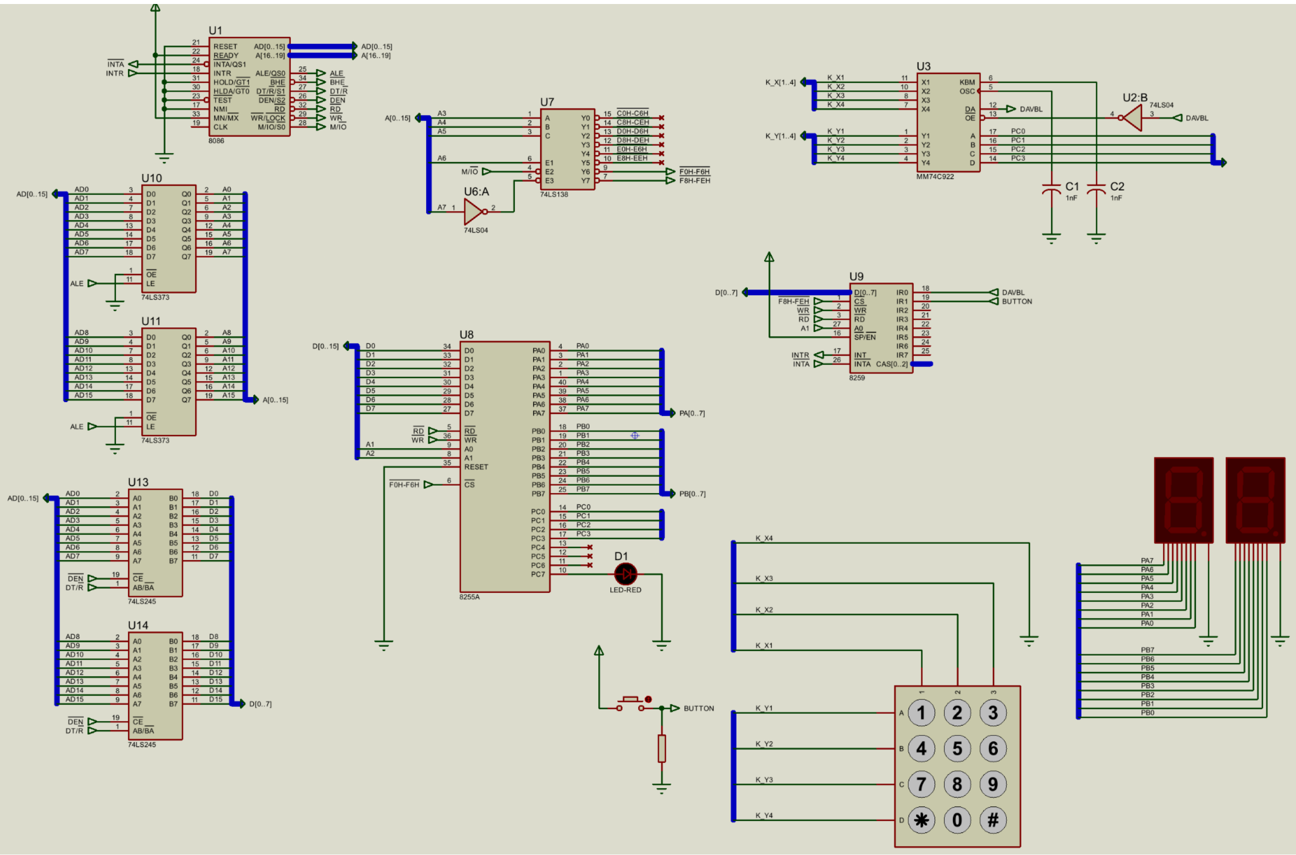
Assembly Code:

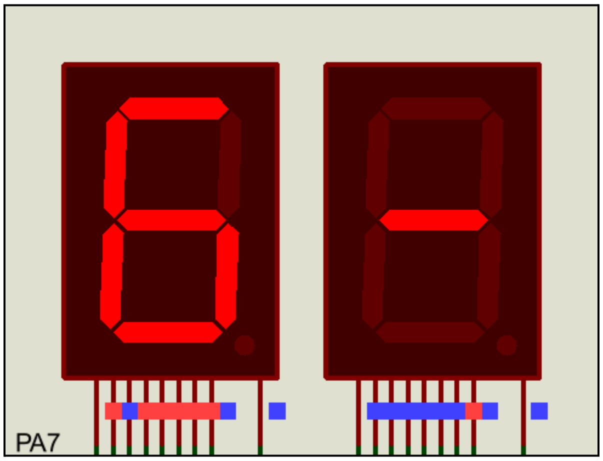
```
DATA SEGMENT
   ORG 03000H
   PORTA EQU 0F0H     ; PORTA address (ISR display)
   PORTB EQU 0F2H     ; PORTB address (foreground display)
   PORTC EQU 0F4H     ; PORTC address (DIP switch input)
   COM_REG EQU 0F6H   ; 8255 Command Register
   PIC1 EQU 0F8H      ; 8259 A1=0 (ICW1, OCW)
   PIC2 EQU 0FAH      ; 8259 A1=1 (ICW2,ICW4,OCW1)

   ICW1 EQU 13H       ; Edge triggered, single, ICW4 needed
   ICW2 EQU 80H       ; Vector base 80H-87H
   ICW4 EQU 03H       ; 8086 mode, auto EOI
   OCW1 EQU 0FCH      ; Unmask only IR0,IR1

DATA ENDS

STK SEGMENT STACK
   BOS DW 64 DUP(?)   ; Stack depth
   TOS LABEL WORD     ; Top of stack
STK ENDS

; ISR segments (separate from main CODE)
PROCED1 SEGMENT
ISR1 PROC FAR
   ASSUME CS:PROCED1, DS:DATA
   ORG 01000H
   PUSHF
   PUSH AX
   PUSH DX

   ; ISR1: Display '9' on PORTA
   MOV DX, PORTA
   MOV AL, 9
   OUT DX, AL

   POP DX
   POP AX
   POPF
   IRET
ISR1 ENDP
PROCED1 ENDS

PROCED2 SEGMENT
ISR2 PROC FAR
   ASSUME CS:PROCED2, DS:DATA
   ORG 02000H
   PUSHF
   PUSH AX
   PUSH DX
   ; ISR2: Display '0' on PORTA
   MOV DX, PORTA
   MOV AL, 0
   OUT DX, AL

   POP DX
   POP AX
   POPF
   IRET
ISR2 ENDP
PROCED2 ENDS

CODE SEGMENT PUBLIC 'CODE'
   ASSUME CS:CODE, DS:DATA, SS:STK
   ORG 08000H
START:
   MOV AX, DATA
   MOV DS, AX
   MOV AX, STK
   MOV SS, AX
   LEA SP, TOS
   CLI

   ; Program the 8255 (Mode 0: PORTA out, PORTB out, PORTC in)
   MOV DX, COM_REG
   MOV AL, 89H       ; PORTA out, PORTB out, PORTC in
   OUT DX, AL

   ; Program the 8259
   MOV DX, PIC1
   MOV AL, ICW1
   OUT DX, AL
   MOV DX, PIC2
   MOV AL, ICW2
   OUT DX, AL
   MOV AL, ICW4
   OUT DX, AL
   MOV AL, OCW1
   OUT DX, AL
   STI

   ; Store interrupt vectors (ES=0 for IVT)
   XOR AX, AX
   MOV ES, AX

   MOV AX, OFFSET ISR1
   MOV [ES:200H], AX
   MOV AX, SEG ISR1
   MOV [ES:202H], AX

   MOV AX, OFFSET ISR2
   MOV [ES:204H], AX
   MOV AX, SEG ISR2
```

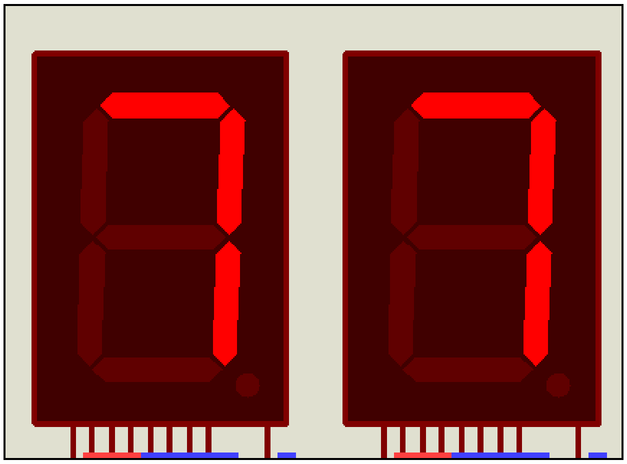
```
    MOV [ES:206H], AX

HERE:
    ; Foreground routine: Read PORTC lower nibble -> display on PORTB
    MOV DX, PORTC
    IN AL, DX
    AND AL, 0FH        ; Mask upper nibble (PC0-PC3 only)

    CMP AL, 09H
    JA show_zero

    MOV DX, PORTB      ; Valid 0-9: display directly
    OUT DX, AL
    JMP HERE

show_zero:
    MOV DX, PORTB      ; >9: display 0
    MOV AL, 0
    OUT DX, AL
    JMP HERE

CODE ENDS
END START
```

Activity #2

Proteus Simulation:







Assembly Code:

```
DATA    SEGMENT
   ORG 03000H

P6_COM_REG   EQU 0F6H
P6_PROGRAM   EQU 081H

ONE_DISPLAY  EQU 0F0H
TWO_DISPLAY  EQU 0F2H
KYPAD_INPUT  EQU 0F4H
LED_OUTPUT   EQU 0F4H
DEFAULT_SEG  EQU 002H
BLINK_DELAY  EQU 07000H

PIC0        EQU 0F8H
PIC1        EQU 0FAH
ICW1         EQU 013H
ICW2         EQU 080H
ICW4         EQU 003H
OCW1          EQU 0FCH

VADD0       EQU 0200H
VADD1       EQU 0204H

DISP_SEGS DB 11111100B, 01100000B, 11011010B
        DB 11110010B, 01100110B, 10110110B
        DB 10111110B, 11100000B, 11111110B
        DB 11110110B, 00000010B, 00000001B
        DB 00000001B, 00000001B, 00000001B
        DB 00000001B

SEG_STATE DB 0AH

KPD_TABLE DB 01H, 02H, 03H, 0AH, 04H, 05H, 06H, 0AH
        DB 07H, 08H, 09H, 0AH, 0AH, 00H, 0AH, 0AH

DATA    ENDS


STACK   SEGMENT PARA STACK
   DW 64 DUP(?)
TOS DW ?
STACK   ENDS


HANDLER0 SEGMENT
ISR0 PROC FAR
   ASSUME CS:HANDLER0, DS:DATA, SS:STACK
   ORG 01000H
   PUSHF
   PUSH AX
   PUSH BX
   PUSH CX
   PUSH DX
   PUSH SI

   MOV DX, LED_OUTPUT
   XOR AL, AL
   OUT DX, AL

   MOV DX, KYPAD_INPUT
   IN  AL, DX

   XOR BX, BX
   MOV BL, AL

   MOV SI, OFFSET KPD_TABLE
   MOV BL, [SI+BX]

   MOV SEG_STATE, BL

   MOV SI, OFFSET DISP_SEGS
   MOV AL, [SI+BX]

   MOV DX, ONE_DISPLAY
   OUT DX, AL

   POP SI
   POP DX
   POP CX
   POP BX
   POP AX
   POPF
   IRET
ISR0 ENDP
HANDLER0 ENDS


HANDLER1 SEGMENT
ISR1 PROC FAR
   ASSUME CS:HANDLER1, DS:DATA, SS:STACK
   ORG 02000H

   PUSHF
   PUSH AX
   PUSH BX
   PUSH DX
   PUSH SI

   XOR BX, BX
   MOV BL, SEG_STATE
   MOV SI, OFFSET DISP_SEGS
   MOV AL, [SI+BX]

   MOV DX, TWO_DISPLAY
   OUT DX, AL

   POP SI
   POP DX
   POP BX
   POP AX
   POPF
   IRET
ISR1 ENDP
HANDLER1 ENDS


CODE    SEGMENT PUBLIC 'CODE'
   ASSUME CS:CODE, DS:DATA, SS:STACK
   ORG 08000H

START:
   CLI

   MOV AX, DATA
   MOV DS, AX
   MOV AX, STACK
   MOV SS, AX
   LEA SP, TOS

   MOV DX, P6_COM_REG
   MOV AL, P6_PROGRAM
   OUT DX, AL

   MOV DX, ONE_DISPLAY
   MOV AL, DEFAULT_SEG
   OUT DX, AL

   MOV DX, TWO_DISPLAY
   MOV AL, DEFAULT_SEG
   OUT DX, AL

   MOV DX, PIC0
   MOV AL, ICW1
   OUT DX, AL

   MOV DX, PIC1
   MOV AL, ICW2
   OUT DX, AL

   MOV AL, ICW4
   OUT DX, AL

   MOV AL, OCW1
   OUT DX, AL

   XOR AX, AX
   MOV ES, AX

   MOV AX, OFFSET ISR0
   MOV [ES:VADD0], AX
   MOV AX, SEG ISR0
   MOV [ES:VADD0+2], AX

   MOV AX, OFFSET ISR1
   MOV [ES:VADD1], AX
   MOV AX, SEG ISR1
   MOV [ES:VADD1+2], AX

   STI

   XOR BX, BX
   MOV DX, LED_OUTPUT

MAIN_LOOP:
   MOV AL, BL
   OUT DX, AL

   MOV CX, BLINK_DELAY
DELAY_LOOP:
      NOP
      LOOP DELAY_LOOP

   NOT BX
   JMP MAIN_LOOP

   HLT

CODE    ENDS
      END START
```