**EXPERIMENT REPORT OF ASSEMBLY LANGUAGE**

Assignment 2 Experiment 2

NAME : ABID ALI

STUDENT ID :2019380141

DATE : 5/22/2021

SUBMITTED TO :PROFESSOR Yin LU

**Problem Description:**

**Chapter 3 Experiment 2 Simple IO and Lantern Control**

1. Do output with 8255 IO controller.

Now let us move to real hardware devices, although it is still an emulated one.

In this experiment, 8 LED lights are connected to the data bus the 8086 processor through port A of a piece of 8255, and a piece of 74LS138 is used to do address decoding. As is shown in figure3.2. The Y0 output of 74LS138 is assigned to 8255. And Address BUS A3 to A5, together A6 and A7 are used to be input of address decoder.

You are required to write a program to light the led one by one. You may start with D9, then light D8 and turn off D9, then move to the next D7, etc. When you reaches D1, please roll back to D2, and so on, till you turn on D9 again. And please do it repeatedly.

You may use logic shift instructions like SHL and SHR to generate the pattern code, and output pattern code through 8255.PortA to light the LEDs. And make use of a flag variable in your program to indicate on which direction should you shift the light.

**Goal:**

We are going to make a virtual device provided by the emu8086.Try to work with peripheral device.The virtual device can display can show up to 5 digits.

We use a word type to port 199 this is the IO address .

Each time we write a number to the port then a sub program is called “delay”,we need to wait for the display to be stable.

**Code:**

;========================================================

;Description: Program of Assignment 2 Experiment 2

;Author:[ABID ALI][2019380141]

;Date:[05/22/2021]

;==============================================================='

;This is the program for experiment2 assignment 1

;In this program, we try to display,8 LED lights are connected to the data bus

;start with D9, then light D8 and turn off D9, then move to the next D7, etc.

;When you reaches D1, please roll back to D2, and so on, till you turn on D9 again.

;==================================================================

.MODEL SMALL

.STACK 32

.DATA

PORT\_A EQU 40H ;

PORT\_CTRL EQU 46H

CTRLWORD\_8255 =10000000B

PATTERN\_CODE DB 01H

FLAG\_SHIFT DB 00H

.CODE ;THIS IS THE PROGRAM ENTRY POINT

MAIN PROC FAR ;LOAD THE DATA SEGMENT ADDRESS

MOV AX, @DATA ;ASSIGN VALUE TO DATA SEGMENT REGISTER

MOV DS, AX

;TODO:PROGRAM 8255

MOV DX, PORT\_CTRL ;MOVING THE VALUE OF PORT\_CTRL IN 8 BIT DX REGISTER

MOV AL, CTRLWORD\_8255 ;MOVING THE VALUE OF CTRLWORD\_8255 IN 8 BIT AL REGISTER

OUT DX, AL

;Sometesting:output portA

MOV DX, PORT\_A ;MOVING THE VALUE OF PORT\_A IN 8 BIT DX REGISTER

MOV AL, 0F5H ;MOVING THE VALUE (0F5H) IN 8 BIT AL REGISTER

OUT DX, AL

MOV AL, PATTERN\_CODE

MOV DX, PORT\_A

MOVE\_LEFT: ;LABEL OF MOVE\_LEFT

OUT DX,AL

CALL DELAY ;CALL DELAY SUB PROCEDURE

SHL AL, 1

CMP AL,10000000B ;COMPARING THE AL REGISTER WITHE THE VALUE

JZ MOVE\_RIGHT

JMP MOVE\_LEFT

MOVE\_RIGHT: ;LABEL OF MOVE\_RIGHT

OUT DX, AL

CALL DELAY ;CALL DELAY SUB PROCEDURE

SHR AL, 1

CMP AL, 00000001H

JZ MOVE\_LEFT

JMP MOVE\_RIGHT

MOV AX,4C00H ;SET UP TO

INT 21H ;RETURN TO DOS

MAIN ENDP

;;==================================================================

;SUBROUTINE:DELAY

;DELAY FOR SOME MILLISECONDS

DELAY PROC NEAR

PUSH BX;

PUSH CX;

MOV BX, 0FH

LOOP\_OUT:

MOV CX, 0FFH ;LABEL OF OUTERLOOP

LOOP\_INNER: ;LABEL OF INNERLOOP

LOOP LOOP\_INNER

DEC BX

JNZ LOOP\_OUT

POP CX ;POP CX FROM THE STACK

POP BX; ;POP BX FROM THE STACK

RET

DELAY ENDP ;ENDING POINT DELAY SEGMENT

END MAIN ;THIS IS THE PROGRAM EXIT POINT

**Explanation:**

There are lot of model available for this program but we take the .MODEL SMALL

We choose Stack size of 32.

**DATA SEGMENT**

PORT\_A EQU 40H ; PORT\_A IS NOT A VARIABLE, IT IS JUST SYNONYM FOR NUMBER 40H

PORT\_CTRL EQU 46H ; PORT\_CTRL IS NOT A VARIABLE, IT IS JUST SYNONYM FOR NUMBER 46H

CTRLWORD\_8255 =10000000B ;IN CTRLWORD\_8255 THE VALUE ASSIGNED 10000000B

PATTERN\_CODE DB 01H ;PATTERN\_CODE DATA BYTE IS 01H

FLAG\_SHIFT DB 00H ;FLAG\_SHIFT CODE DATA BYTE IS 00H

**CODE SEGMENT**

MAIN PROC FAR ;LOAD THE DATA SEGMENT ADDRESS

MOV AX, @DATA ;ASSIGN VALUE TO DATA SEGMENT REGISTER

MOV DS, AX

;TODO1:PROGRAM 8255

MOV DX, PORT\_CTRL ;MOVING THE VALUE OF PORT\_CTRL IN 8 BIT DX REGISTER

MOV AL, CTRLWORD\_8255 ;MOVING THE VALUE OF CTRLWORD\_8255 IN 8 BIT AL REGISTER

OUT DX, AL

;TODO2:

;Sometesting:output portA

MOV DX, PORT\_A ;MOVING THE VALUE OF PORT\_A IN 8 BIT DX REGISTER

MOV AL, 0F5H ;MOVING THE VALUE (0F5H) IN 8 BIT AL REGISTER

OUT DX, AL

MOV AL, PATTERN\_CODE

MOV DX, PORT\_A

MOVE\_LEFT: ;LABEL OF MOVE\_LEFT

OUT DX,AL

CALL DELAY ;CALL DELAY SUB PROCEDURE

SHL AL, 1 ;MOVES ALL OF BITS IN AL LEFT ONE PLACE

CMP AL,10000000B ; COMPARING THE AL REGISTER WITH THE VALUE

JZ MOVE\_RIGHT ; JUMP ZERO INSTRUCTION,IT'S A CONDITIONAL INSTRUCTION

;IT JUMP WHEN ZERO FLAG(ZF) IS SET 1

;JZ is commonly used to explicitly test for something being equal to zero

JMP MOVE\_LEFT

MOVE\_RIGHT: ;LABEL OF MOVE\_RIGHT

OUT DX, AL

CALL DELAY ;CALL DELAY SUB PROCEDURE

SHR AL, 1 ;MOVES ALL OF BITS IN AL RIGHT ONE PLACE

CMP AL, 00000001H

JZ MOVE\_LEFT

JMP MOVE\_RIGHT

MOV AX,4C00H ;SET UP TO

INT 21H ;RETURN TO DOS

MAIN ENDP

**SUBPROGRAM**

;SUBROUTINE:DELAY

;DELAY FOR SOME MILLISECONDS

DELAY PROC NEAR

PUSH BX;

PUSH CX;

MOV BX, 0FH

LOOP\_OUT:

MOV CX, 0FFH ;LABEL OF OUTERLOOP

LOOP\_INNER: ;LABEL OF INNERLOOP

LOOP LOOP\_INNER ;LOOP INSTRUCTION IS USED FOR LOOPING

DEC BX ;BX IS DECREMENTED

JNZ LOOP\_OUT ;JNZ IS JUMP NOT ZERO,IT'S A CONDITIONAL JUMP

;IT JUMPS TO LOOP\_OT IF ZERO FLAG(ZF) IS 'ZERO'

POP CX ;POP CX FROM THE STACK

POP BX; ;POP BX FROM THE STACK

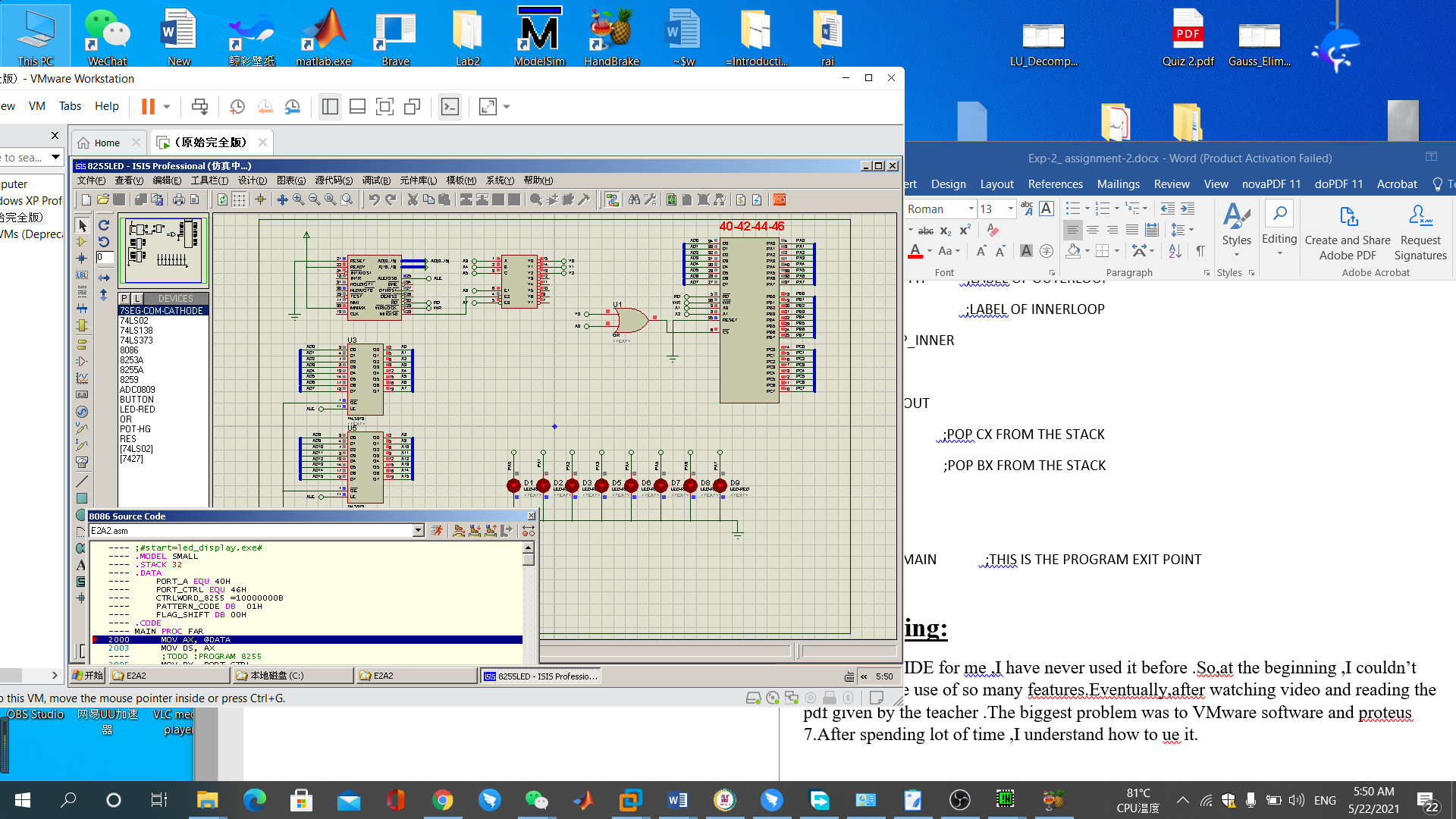
RET ;IT'S THE RETURN INSTRUCTION

DELAY ENDP ;;ENDING POINT DELAY SEGMENT

END MAIN ;THIS IS THE PROGRAM EXIT POINT

**Debugging:**

This is a new IDE for me ,I have never used it before .So,at the beginning ,I couldn’t understand the use of so many features.Eventually,after watching video and reading the pdf given by the teacher .The biggest problem was to VMware software and proteus 7.After spending lot of time ,I understand how to use it.



**Attachment:**

1) Experiment-2(assignment-2).m4v

2) Exp-2\_ assignment 2.asm

3) Exp-2\_ assignment 2.pdf

**Acknowledgement:**

I complete this assignment by myself by using online videos and taking help from online.The most useful help from teacher’s hint given in question ,the theory class and the lecture note from the practical class.