**ECE3301L Spring 2024 Midterm II**

Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Daniele Ricciardelli\_\_\_\_\_\_\_\_\_\_\_\_\_

1 Problem on definition/True/False – 10 questions 20 points p1

1 Problem on Timers – 10 points each 10 points p2

1 Problem on Timers – 10 points each 10 points p3

1 Problem on creating delay 15 points p4

1 Problem on writing simple instructions to setup Interrupt and Timers 15 points p5

1 Problem on writing code to handle External Interrupts 15 points p6

1 Problems writing code to handle GPIO 10 points p7

105 points

All questions were answered in paper but #6 code.

Problem 1 (20 points) Score: \_\_\_\_\_\_\_\_\_\_\_\_

1. True or false. After reset, all the external hardware interrupts are defaulted to negative edge-triggered.

Ans: \_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. What timer can only operate in 8-bit mode?

Ans: \_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Upon reset, what is the default speed of the PIC18F4321?

Ans: \_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Timer 2 has a prescale and a post-scale. Determine the maximum value for each and calculate the maximum combined scaling factor we can get using Timer 2?

Ans: \_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Which technique, interrupt or polling, will avoid tying up the microcontroller?

Ans: \_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. True or false. After reset, all the external hardware interrupts are defaulted to negative edge-triggered.

Ans: \_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. In the PIC18F4321, which port has only usable 3 GPIO pins?

Ans: \_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. What memory address in the interrupt vector table is assigned to low-priority interrupts?

Ans: \_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. True or false. Upon reset, some pins on PORTA, PORTB and PORTE are defaulted in digital mode

Ans: \_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. True or false. We can only use the internal oscillator inside the PIC18F microcontroller.

Ans: \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Problem 2 (10 points) Score: \_\_\_\_\_\_\_\_\_\_\_\_

Assuming that the PIC18F is running at the frequency of 8 MHz.

1. Find the **frequency** of the clock that is fed into the **Timer0** if a prescaler option of 256 is selected

Ans: Prescaled Timer Frequency = \_\_\_\_\_\_\_\_\_ (show calculation)

1. Based on the timer’s frequency on part a), if the registers TMR0L and TMR0H are loaded with the following values:

TMR0H = 0xB0;

TMR0L = 0x80;

What is the delay generated in term of milliseconds:

Ans: \_**Delay** = \_\_\_\_\_\_\_msec (Show all calculations)

Problem 3 (10 points) Score: \_\_\_\_\_\_\_\_\_\_\_\_

Assuming that the PIC18F4321 is operating at 8 Mhz system clock.

1. Find the frequency of the clock that is fed into the 16-bit **Timer3** if a prescale option of 8 is selected

**Ans: Prescale Timer Frequency** ( Show all calculations) **:**

1. Find the values of TMR3H and TMR3L needed to create a **time delay of 4 msec.** – Show all calculations

**TMR3H = 0x\_\_;**

**TMR3L = 0x\_\_;**

Problem 4 (15 points) Score: \_\_\_\_\_\_\_\_\_\_\_\_

1. Write a complete function called T3\_Delay() that generates the 4 msec delay from Problem 3). Make sure to setup all the required registers including the T3CON register.

T3CON:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
|  |  |  |  |  |  |  |  |

CODE:

void T3\_Delay()

{

}

1. Write a main program that will stay in a infinite loop whereas PORTA bit 3 is **toggled** after the 4 msec delay (call the function T3\_Delay()). This will generate a square wave with a period of 8 msec. Make sure to set TRISA and the OSCCON are programmed properly.

void main()

{

}

Problem 5 (15 points) Score: \_\_\_\_\_\_\_\_\_\_\_\_

1. Generate the same square wave of problem #4b) with 8 msec period using the delay generated from problem #3 and through Interrupt I/O method. The program should have the main program to initialize:
   1. All port directions (TRISx)
   2. System frequency (OSCCON)
   3. T3CON
   4. The T3 timer registers (problem #3)
   5. The proper settings of the IF, IE and GIE bits
   6. A while (1) loop to wait for interrupt
   7. An ISR routine when the T3 interrupt occurs. This routine is used to toggle the PORT A bit 3

CODE:

Problem 6 (15 points) Score: \_\_\_\_\_\_\_\_\_\_\_\_

Two push-button switches are connected respectively to INT0 and INT1 and 4 LEDs are connected to PORT C bits 0 to 3. Write a c program that will perform the following tasks:

1. Output initially the value of 0101 in binary those 4 LEDs (PORT C).
2. Every time the push-button is pressed (HIGH to LOW transition) on **INT0**, the first 2 LEDs (bits 0 and 1) of PORTC are toggled
3. Each time the push-button (HIGH to LOW transition)is pressed on **INT1**, the bits 2 and 3 of PORTC are toggled.

The c program will:

* Initialize the ports B and C directions.
* Initialize the operations so that the two lines INT0 and INT1 can work to interrupt
* Make sure the edge registers (INTEDGx) are properly programmed
* Initialize the Interrupt IF, IE bits
* Stay in a while(1) loop to wait for an interrupt
* When the interrupt occurs, make sure to have the high priority interrupt service routine to handle any occurrence of two INT0 and INT1 interrupts. These two ISR routines are used to change the output of the PORT C based on the requirements.
* Hint: Define each PORT C bit with a name to easily do the toggling of the bits.

CODE:

#include <xc.h>

#pragma config OSC = HS

#pragma config WDT = OFF

#pragma config LVP = OFF

#pragma config DEBUG = OFF

#define LED\_0\_1 0x03

#define LED\_2\_3 0x0C

// Initialize port settings

void initPorts(void) {

ADCON1 = 0x0F; // Set all PORTB and PORTC pins as digital

TRISB = 0x03; // Configure RB0 and RB1 as inputs

TRISC = 0x00; // Set all PORTC pins as outputs

PORTC = 0x05; // Initial value for PORTC

}

// Configure interrupts

void initInterrupts(void) {

INTCONbits.INT0IF = INTCON3bits.INT1IF = 0; // Clear interrupt flags for INT0 and INT1

INTCON2bits.INTEDG0 = INTCON2bits.INTEDG1 = 0; // Set INT0 and INT1 to trigger on falling edge

INTCONbits.INT0IE = INTCON3bits.INT1IE = 1; // Enable INT0 and INT1 interrupts

RCONbits.IPEN = INTCONbits.GIE = INTCONbits.PEIE = 1; // Enable priority levels, global and peripheral interrupts

}

// Interrupt service routine for high priority interrupts

void \_\_interrupt(high\_priority) HighPriorityISR(void) {

if (INTCONbits.INT0IF) { // Check for INT0 interrupt

PORTC ^= LED\_0\_1; // Toggle LEDs 0 and 1

INTCONbits.INT0IF = 0; // Clear the interrupt flag for INT0

}

if (INTCON3bits.INT1IF) { // Check for INT1 interrupt

PORTC ^= LED\_2\_3; // Toggle LEDs 2 and 3

INTCON3bits.INT1IF = 0; // Clear the interrupt flag for INT1

}

}

// Main function

void main(void) {

initPorts(); // Call initialization for ports

initInterrupts(); // Call initialization for interrupts

while (1) {} // Infinite loop to keep the program running

}

Problem 7 (10 points) Score: \_\_\_\_\_\_\_\_\_\_\_\_

The PIC18F4321 microcontroller is required to output a BCD digit (0 to 9) to a **common-anode** 7- segment display connected to bits 0 through 6 of PORTD. The PIC18F4321 inputs the BCD number via four switches connected to **bits 4 through 7** of PORTB. Write a C language program that will continuously read the four switches and based on the binary code represented by those four switches display a BCD digit (0 to 9) on the seven segment display. Display a blank if the input number is higher than 9.

Use the array developed in the labs for the 7-segment codes.

Make sure to have all the proper direction (TRIS) registers initialized.

CODE: