UNIVERSITY OF RUHUNA

Faculty of Engineering

End-Semester 6 Examination in Engineering: February 2020

Module Number: EE6304 **Module Name: Embedded System Design**

[Three Hours]

[Answer **all questions**, each question carries 10 marks]

Q1 a) Briefly describe two advantages of using a microcontroller in an embedded system than using discrete logic based circuits.

[1 Mark]

b) Briefly describe how an IDE is beneficial in embedded systems programming than using a separate text editor and a compiler.

[1 Mark]

c) i) Briefly describe the difference between a platform and an architecture with respect to a microcontroller.
ii) Using a figure, illustrate how an executable is made.
iii) Why is it said that "the executable is platform dependent"?

[3 Marks]

d) i) Briefly describe what an interrupt vector is.
ii) Using a figure, illustrate what happens when an interrupt occurs while a program is running in a microcontroller.
iii) List two types of interrupts associated with a microcontroller.

[3 Marks]

e) Briefly explain how you would guarantee the order of execution of three tasks in a multi-tasking environment.

[1 Mark]

f) What are the three main tasks of a kernel?

[1 Mark]

Q2 a) Consider the assembly code written for PIC16F877A microcontroller which is built in MPLAB-X IDE V5.15 given in Figure Q2.1.
Note: The register summary is given in Figure Q2.2 and Assembly instruction summary is given in Figure Q2.3.

i) Which line will be executed after line 12?
ii) What is the name of the register accessed in line 31 and the value set at the register after execution of the line?
iii) Briefly describe the operation of line 38 including name of the register accessed, value of the register after execution.
iv) Assume that the execution time of MOVLW, MOVWF, and GOTO instructions is 1 microsecond and DECFSZ is 2 μs. How long does it take for the program to come from line 14 to line 22 (including line 14 and excluding line 22)?

v) Briefly describe the key functionality of this program.

[5 Marks]

b) You have been asked to write an assembly code for PIC16F877A microcontroller to implement the following functionality.

```
If PORTC pin 0 is HIGH
  Set PORT A pin 4 LOW
Else
  Set PORT A pin 4 HIGH
```

Assuming that the data direction of the ports are already assigned, implement the assembly code to implement the given functionality.

Note: You can either use instructions from the list given in Figure Q2.3 or any branch instruction of your preference.

[3 Marks]

c) Explain functionality or write a simple pseudocode equivalent to the following assembly code.

```
START  MOVLW  0x0A
       MOVWF  0x20
LOOP   INCF   0x20
       INCFSZ 0x20, 0x01
       GOTO   LOOP
       END
```

[2 Marks]

Q3 You are given a microcontroller with 8-bit, 16-bit, and 32-bit timers, namely, Timer0, Timer1, and Timer2. Each has 5 prescale values (1, 8, 64, 256, and 1024). Each timer consists of a timer register and a compare register which can be programmed to generate timer overflow interrupt and compare interrupt. Further, the microcontroller is equipped with a 12-bit analog to digital converter (ADC) with four channels. The reference voltages V_{ref}^+ and V_{ref}^- are connected to 4 V and to the ground, respectively and the CPU speed is set to 20 MHz.

a) It is required to perform a time sensitive operation which requires a timer interrupt every 750 ms.

i) Identify the timer to be used in the above application and justify your choice.
ii) Identify the best prescale value to be used and justify your choice.
iii) What should be the value set at compare register in order to generate an interrupt every 750 ms?

[3 Marks]

b) What is the input voltage if the value obtained at the ADC output is 768?

[1 Mark]

c) You are required to program the microcontroller to read data from an analog sensor every 750 ms.

- Assume that the timer interrupt is already programmed to trigger at every 750 ms and a variable 'read_Flag' is set inside the timer interrupt service routine.
- A function 'read_ADC( )' is used to read the ADC output and the function returns the discrete value.
- A function 'display_Value()' is used to display a value of type double in an LCD screen. It takes a double as an input parameter and does not return anything. Note that the function will limit the display to three decimal digits by truncating the input double (eg. 1.234 V).

i) Use the given information to draw a simple flow chart to display the sensor output voltage at the LCD screen if it is greater than or equal to 2.5 V.

ii) What will be the value displayed at the LCD screen if the input voltage is 3V. State any assumptions you make.

[3 Marks]

d) Consider the following code to initialize the serial port of a PIC16F877A microcontroller, where the important registers are given in Figure Q3.

```
void USART_Init (void)
{
        TRISC = 0x80;
        TXSTA = 0x24;
        RCSTA = 0x90;
        SPBRG = 65;
}
```

i) Evaluate the baud rate given that
BRGH = 1 : High Speed SPBRG = (Fosc / (16 * BaudRate)) - 1
BRGH = 0 : Low Speed SPBRG = (Fosc / (64 * BaudRate)) - 1.

ii) Identify the settings which needs to be matched with the other device connected to this microcontroller in order to successfully communicate between the two devices.
Hint: Use the data given in Figure Q3.

[3 Marks]

Q4 a) Assume that you are assigned a task to design two embedded devices given below.
Device 1: a communication device to be installed in a train.
Device 2: a device be carried by a nature photographer.

i) Identify two design challenges for each of the device. If you identify more than two, write the most important two of them (do no write more).

ii) If the photographer is supposed to leave the device in a rough terrain to periodically capture photographs, briefly discuss on how you could guarantee the fail-safe operation of the device through design?

[3 Marks]

b) Selecting a proper microcontroller at the design stage is critical. Briefly explain why it is important to choose a microcontroller with a bootloader.

[2 Marks]

c) Consider a sensor module developed for a line following robot which is shown in Figure Q4, where S_A and S_B are the line sensors. Each of the sensor output is LOW if it detects the line and HIGH if the line is not detected.
The chosen microcontroller consist of two I/O ports including two hardware interrupt pins.
You are given a set of library functions including functions move_Left( ), move_Right( ), and move_Straight( ).

i) Name a (kernel) design strategy to be used in your program.

ii) Explain how you would design your program. You may use flow charts/block diagrams/codes to illustrate your answer.

iii) If the same line following robot is to be used in an environment where it has to follow a white line in a dark environment, draw and explain the changes you would do in your hardware if the microcontroller program is to remain unchanged.

[5 Marks]

```
11      RES_VECT  CODE    0x0000
12                GOTO    START
13
14      MYSUB     MOVLW   0xFF
15                MOVWF   0x21
16      LOOP1     MOVLW   0xFF
17                MOVWF   0x22
18      LOOP2     MOVLW   0x02
19                MOVWF   0x23
20      LOOP3     DECFSZ  0x23,0x01
21                GOTO    LOOP3
22                DECFSZ  0x22,0x01
23                GOTO    LOOP2
24                DECFSZ  0x21,0x01
25                GOTO    LOOP1
26                RETURN
27
28      START
29                BCF     0x03, 0x05
30                BCF     0x03, 0x06
31                CLRF    0x05
32                BSF     0x03, 0x05
33                MOVLW   0xC7
34                MOVWF   0x85
35                BCF     0x03, 0x05
36                BCF     0x03, 0x06
37      BLINK     MOVLW   0x28
38                IORWF   0x05, 0x01
39                CALL    MYSUB
40                MOVLW   0xDF
41                ANDWF   0x05, 0x01
42                CALL    MYSUB
43                END
```

Figure Q2.1

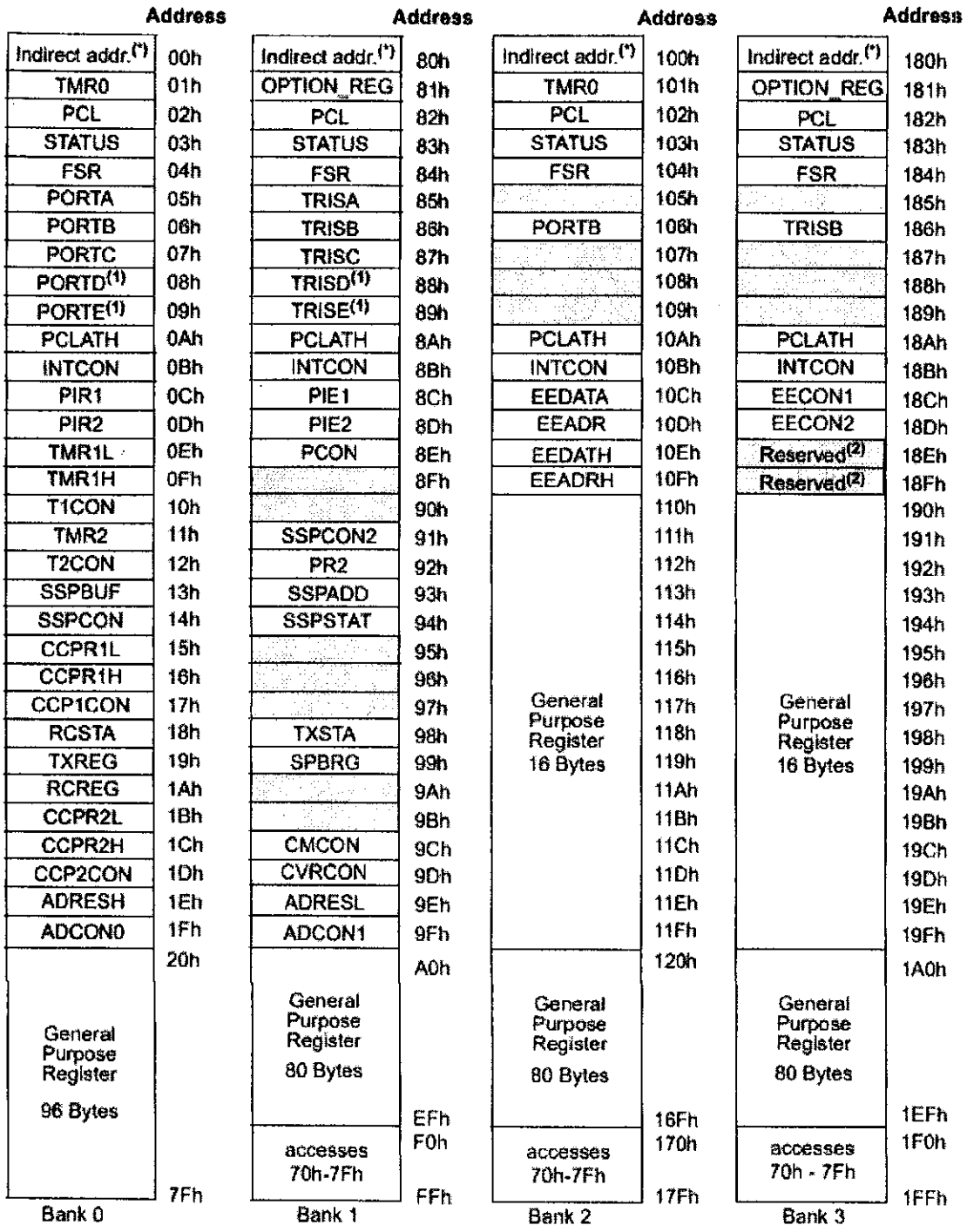

Figure Q2.2: Register Banks for PIC16F877A

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Details on page:
Bank 0											
00h(3)	INDF	Addressing this location uses contents of FSR to address data memory (not a physical register)								0000 0000	31, 150
01h	TMR0	Timer0 Module Register								xxxx xxxx	55, 150
02h(3)	PCL	Program Counter (PC) Least Significant Byte								0000 0000	30, 150
03h(3)	STATUS	IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C	0001 1xxx	22, 150
04h(3)	FSR	Indirect Data Memory Address Pointer								xxxx xxxx	31, 150
05h	PORTA	—	—	PORTA Data Latch when written: PORTA pins when read						--0x 0000	43, 150
06h	PORTB	PORTB Data Latch when written: PORTB pins when read								xxxx xxxx	45, 150
07h	PORTC	PORTC Data Latch when written: PORTC pins when read								xxxx xxxx	47, 150
08h(4)	PORTD	PORTD Data Latch when written: PORTD pins when read								xxxx xxxx	48, 150
09h(4)	PORTE	—	—	—	—	—	RE2	RE1	RE0	---- -xxx	49, 150
0Ah(1,3)	PCLATH	—	—	—	Write Buffer for the upper 5 bits of the Program Counter					---0 0000	30, 150
0Bh(3)	INTCON	GIE	PEIE	TMR0IE	INTE	RBIE	TMR0IF	INTF	RBIF	0000 000x	24, 150
0Ch	PIR1	PSPIF(3)	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000	26, 150
0Dh	PIR2	—	CMIF	—	EEIF	BCLIF	—	—	CCP2IF	-0-0 0--0	28, 150
0Eh	TMR1L	Holding Register for the Least Significant Byte of the 16-bit TMR1 Register								xxxx xxxx	60, 150
0Fh	TMR1H	Holding Register for the Most Significant Byte of the 16-bit TMR1 Register								xxxx xxxx	60, 150
10h	T1CON	—	—	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{T1SYNC}$	TMR1CS	TMR1ON	--00 0000	57, 150
11h	TMR2	Timer2 Module Register								0000 0000	62, 150
12h	T2CON	—	TOUTPS3	TOUTPS2	TOUTPS1	TOUTPS0	TMR2ON	T2CKPS1	T2CKPS0	-000 0000	61, 150
13h	SSPBUF	Synchronous Serial Port Receive Buffer/Transmit Register								xxxx xxxx	79, 150
14h	SSPCON	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000	82, 82, 150
15h	CCPR1L	Capture/Compare/PWM Register 1 (LSB)								xxxx xxxx	63, 150
16h	CCPR1H	Capture/Compare/PWM Register 1 (MSB)								xxxx xxxx	63, 150
17h	CCP1CON	—	—	CCP1X	CCP1Y	CCP1M3	CCP1M2	CCP1M1	CCP1M0	--00 0000	64, 150
18h	RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	112, 150
19h	TXREG	USART Transmit Data Register								0000 0000	118, 150
1Ah	RCREG	USART Receive Data Register								0000 0000	118, 150
1Bh	CCPR2L	Capture/Compare/PWM Register 2 (LSB)								xxxx xxxx	63, 150

Figure Q2.3: Special Function Register Summary for PIC16F877A

Mnemonic, Operands		Description	Cycles	14-Bit Opcode MSb LSb	Status Affected	Notes
BYTE-ORIENTED FILE REGISTER OPERATIONS						
ADDWF	f, d	Add W and f	1	00 0111 dfff ffff	C,DC,Z	1,2
ANDWF	f, d	AND W with f	1	00 0101 dfff ffff	Z	1,2
CLRF	f	Clear f	1	00 0001 lfff ffff	Z	2
CLRW	-	Clear W	1	00 0001 0xxx xxxx	Z	
COMF	f, d	Complement f	1	00 1001 dfff ffff	Z	1,2
DECF	f, d	Decrement f	1	00 0011 dfff ffff	Z	1,2
DECFSZ	f, d	Decrement f, Skip if 0	1(2)	00 1011 dfff ffff		1,2,3
INCF	f, d	Increment f	1	00 1010 dfff ffff	Z	1,2
INCFSZ	f, d	Increment f, Skip if 0	1(2)	00 1111 dfff ffff		1,2,3
IORWF	f, d	Inclusive OR W with f	1	00 0100 dfff ffff	Z	1,2
MOVF	f, d	Move f	1	00 1000 dfff ffff	Z	1,2
MOVWF	f	Move W to f	1	00 0000 lfff ffff		
NOP	-	No Operation	1	00 0000 0xx0 0000		
RLF	f, d	Rotate Left f through Carry	1	00 1101 dfff ffff	C	1,2
RRF	f, d	Rotate Right f through Carry	1	00 1100 dfff ffff	C	1,2
SUBWF	f, d	Subtract W from f	1	00 0010 dfff ffff	C,DC,Z	1,2
SWAPF	f, d	Swap nibbles in f	1	00 1110 dfff ffff		1,2
XORWF	f, d	Exclusive OR W with f	1	00 0110 dfff ffff	Z	1,2
BIT-ORIENTED FILE REGISTER OPERATIONS						
BCF	f, b	Bit Clear f	1	01 00bb bfff ffff		1,2
BSF	f, b	Bit Set f	1	01 01bb bfff ffff		1,2
BTFSC	f, b	Bit Test f, Skip if Clear	1 (2)	01 10bb bfff ffff		3
BTFSS	f, b	Bit Test f, Skip if Set	1 (2)	01 11bb bfff ffff		3
LITERAL AND CONTROL OPERATIONS						
ADDLW	k	Add Literal and W	1	11 111x kkkk kkkk	C,DC,Z	
ANDLW	k	AND Literal with W	1	11 1001 kkkk kkkk	Z	
CALL	k	Call Subroutine	2	10 0kkk kkkk kkkk		
CLRWDT	-	Clear Watchdog Timer	1	00 0000 0110 0100	$\overline{TO}$,$\overline{PD}$	
GOTO	k	Go to Address	2	10 1kkk kkkk kkkk		
IORLW	k	Inclusive OR Literal with W	1	11 1000 kkkk kkkk	Z	
MOVLW	k	Move Literal to W	1	11 00xx kkkk kkkk		
RETFIE	-	Return from Interrupt	2	00 0000 0000 1001		
RETLW	k	Return with Literal in W	2	11 01xx kkkk kkkk		
RETURN	-	Return from Subroutine	2	00 0000 0000 1000		
SLEEP	-	Go into Standby mode	1	00 0000 0110 0011	$\overline{TO}$,$\overline{PD}$	
SUBLW	k	Subtract W from Literal	1	11 110x kkkk kkkk	C,DC,Z	
XORLW	k	Exclusive OR Literal with W	1	11 1010 kkkk kkkk	Z	

Figure Q2.4: List of Assembly Instructions for PIC16F877A

BCF	Bit Clear f
Syntax:	[*label*] BCF f,b
Operands:	0 ≤ f ≤ 127 0 ≤ b ≤ 7
Operation:	0 → (f<b>)
Status Affected:	None
Description:	Bit 'b' in register 'f' is cleared.

BSF	Bit Set f
Syntax:	[*label*] BSF f,b
Operands:	0 ≤ f ≤ 127 0 ≤ b ≤ 7
Operation:	1 → (f<b>)
Status Affected:	None
Description:	Bit 'b' in register 'f' is set.

GOTO	Unconditional Branch
Syntax:	[*label*] GOTO k
Operands:	0 ≤ k ≤ 2047
Operation:	k → PC<10:0> PCLATH<4:3> → PC<12:11>
Status Affected:	None
Description:	GOTO is an unconditional branch. The eleven-bit immediate value is loaded into PC bits <10:0>. The upper bits of PC are loaded from PCLATH<4:3>. GOTO is a two-cycle instruction.

IORLW	Inclusive OR Literal with W
Syntax:	[*label*] IORLW k
Operands:	0 ≤ k ≤ 255
Operation:	(W) .OR. k → (W)
Status Affected:	Z
Description:	The contents of the W register are OR'ed with the eight-bit literal 'k'. The result is placed in the W register.

DECFSZ	Decrement f, Skip if 0
Syntax:	[*label*] DECFSZ f,d
Operands:	0 ≤ f ≤ 127 d ∈ [0,1]
Operation:	(f) - 1 → (destination); skip if result = 0
Status Affected:	None
Description:	The contents of register 'f' are decremented. If 'd' is '0', the result is placed in the W register. If 'd' is '1', the result is placed back in register 'f'. If the result is '1', the next instruction is executed. If the result is '0', then a NOP is executed instead, making it a 2 TCY instruction.

INCFSZ	Increment f, Skip if 0
Syntax:	[*label*] INCFSZ f,d
Operands:	0 ≤ f ≤ 127 d ∈ [0,1]
Operation:	(f) + 1 → (destination), skip if result = 0
Status Affected:	None
Description:	The contents of register 'f' are incremented. If 'd' is '0', the result is placed in the W register. If 'd' is '1', the result is placed back in register 'f'. If the result is '1', the next instruction is executed. If the result is '0', a NOP is executed instead, making it a 2 TCY instruction.

ANDWF	AND W with f
Syntax:	[*label*] ANDWF f,d
Operands:	0 ≤ f ≤ 127 d ∈ [0,1]
Operation:	(W) .AND. (f) → (destination)
Status Affected:	Z
Description:	AND the W register with register 'f'. If 'd' is '0', the result is stored in the W register. If 'd' is '1', the result is stored back in register 'f'.

IORWF	Inclusive OR W with f
Syntax:	[*label*] IORWF f,d
Operands:	0 ≤ f ≤ 127 d ∈ [0,1]
Operation:	(W) .OR. (f) → (destination)
Status Affected:	Z
Description:	Inclusive OR the W register with register 'f'. If 'd' is '0', the result is placed in the W register. If 'd' is '1', the result is placed back in register 'f'.

Figure Q2.5: Detailed Description of some Assembly Instructions for PIC16F877A

TXSTA	7	6	5	4	3	2	1	0
	CSRC	TX9	TXEN	SYNC	-	BRGH	TRMT	TX9D

CSRC: Clock Source Select bit
Asynchronous mode: Don't care.

TX9: 9-bit Transmit Enable bit
1 = Selects 9-bit transmission
0 = Selects 8-bit transmission

TXEN: Transmit Enable bit
1 = Transmit enabled
0 = Transmit disabled

SYNC: USART Mode Select bit
1 = Synchronous mode
0 = Asynchronous mode

BRGH: High Baud Rate Select bit
1 = High speed
0 = Low speed

TRMT: Transmit Shift Register Status bit
1 = TSR empty
0 = TSR full

TX9D: 9th bit of Transmit Data, can be Parity bit

RCSTA	7	6	5	4	3	2	1	0
	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D

SPEN: Serial Port Enable bit
1 = Serial port enabled
(configures RC7/RX/DT and RC6/TX/CK pins as serial port pins)
0 = Serial port disabled

RX9: 9-bit Receive Enable bit
1 = Selects 9-bit reception
0 = Selects 8-bit reception

SREN: Single Receive Enable bit
Asynchronous mode: Don't care.

CREN: Continuous Receive Enable bit
Asynchronous mode:
1 = Enables continuous receive
0 = Disables continuous receive

ADDEN: Address Detect Enable bit
Asynchronous mode 9-bit (RX9 = 1):
1 = Enables address detection, enables interrupt and load of the receive buffer when RSR is set
0 = Disables address detection, all bytes are received and ninth bit can be used as parity bit

FERR: Framing Error bit
1 = Framing error (can be updated by reading RCREG register and receive next valid byte)
0 = No framing error

OERR: Overrun Error bit
1 = Overrun error (can be cleared by clearing bit CREN)
0 = No overrun error
RX9D: 9th bit of Received Data (can be parity bit but must be calculated by user firmware)

Figure Q3: Important Registers for UART Configuration in PIC16F877A