



UNIVERSITY OF RUHUNA

Faculty of Engineering

End-Semester 6 Examination in Engineering: May 2026

Module Number: EE6352 Module Name: Embedded System Design(C-18)

[Three Hours]

[Answer all questions]

-
- Q1. a) Compare the differences in between 16F84A and 16F877A microcontrollers [2 Marks]
- b) Briefly explain in few words the purpose of the address 0x0004 in the program memory of the PIC 16 series microcontrollers. [2 Marks]
- c) What is meant by the statement; PIC 16 series microcontrollers are 8-bit microcontrollers. [2 Marks]
- d) State the difference between a microprocessor and a microcontroller. [2 Marks]
- e) State the advantages of flash ROM, compared with other memory types [2 Marks]
- f) State the function of the EQU directive. [2 Marks]
- Q2. A matrix keypad consists of matrix switches (one switch for each key) as shown in Figure Q2 (a). In order to scan the keypad, each column is pulled to logic one in turn whilst each row is read in turn. Figure Q2 (b) demonstrates the flow chart of the scanning process. Once the key is identified it should be displayed on a 7 segment LED which is connected to a Microcontroller. The task of the program is to produce software which will scan the keypad and determine which key is pressed. Refer the table Q2 for coding.
- a) Draw a complete flow chart to demonstrate the overall operation of the keypad. [2 Marks]
- b) Write a algorithm to demonstrate the scanning and identification process. [3 Marks]
- c) Draw a complete circuit diagram with Microcontroller. [2 Marks]
- d) Write a program in assembly language to execute the given task. [3 Marks]

Q3. You are developing firmware for a low-power indoor air quality monitor using the ATmega328P, where a potentiometer connected to ADC0 (PC0) simulates the analog output of a gas sensor. The system reads the sensor value and sends it to a virtual terminal every 2 seconds via UART, triggered by a Timer1 Compare Match Interrupt. Timer1 is a 16-bit timer, and both ADC conversion and USART transmission occur inside the ISR. Refer to Figure Q3.1 for the setup.

You are given a partially completed C program in Figure Q3.3. Answer the following questions based on your understanding of the ATmega328P microcontroller and register configurations.

- a)
 - i) Define polling in embedded systems and briefly state its main disadvantage. Identify where polling is used in the given code in Figure Q3.3 and specify the corresponding line numbers [1.5 Marks]
 - ii) Select a prescaler, justify your selection with calculations, and complete line 49 by setting the appropriate TCCR1B prescaler bits. Refer to Figure Q3.2 and Table Q3.1 for assistance. [1.5 Marks]
 - iii) Complete lines 50 and 51 of the given code by calculating and setting the correct values for OCR1AH and OCR1AL to generate an interrupt exactly every 2 seconds. Show your full calculations clearly. (Note: In CTC mode, Timer1 compares its count with the OCR1A register.) [1 Mark]
- b)
 - i) Based on the provided C program in Figure Q3.3, calculate the baud rate of the microcontroller. Show all steps clearly. [1 Mark]
 - ii) A user observes that random or corrupted characters (junk values) are displayed on the virtual terminal instead of the expected output. List four (04) possible reasons why this issue could occur. [1 Mark]
 - iii) Configure the ATmega328P microcontroller to operate at its minimum possible theoretical baud rate. Clearly show all steps of your calculation. Also, modify the provided code at the appropriate place to match the new configuration. [1 Mark]
- c)
 - i) What is a Real-Time Operating System (RTOS) in the context of embedded systems? [1 Mark]
 - ii) List any four (04) basic functions of a real-time kernel in embedded systems. [1 Mark]
 - iii) In point form, outline the roles of the watchdog timer and bootloader in microcontroller applications. [1 Mark]

- Q4. a) State the difference between a microprocessor and a microcontroller. [1 Mark]
- b) Explain why a pull-up resistor is needed with a switch input. [1 Mark]
- c) Briefly explain why hardware timers are useful in MCUs. [1 Marks]
- d) Explain how a pre-scalar extends the timer period [1 Marks]
- e) Explain why the plain 7-segment LED display needs a code table. [2 Marks]
- f) What is I2C communication. Explain the structure of the I2C message. Write down the pros and cons of I2C communication (2 each). [2 Marks]
- g) Write the stepwise explanation of how SPI works. Explain CPOL and CPHA? [2 Marks]

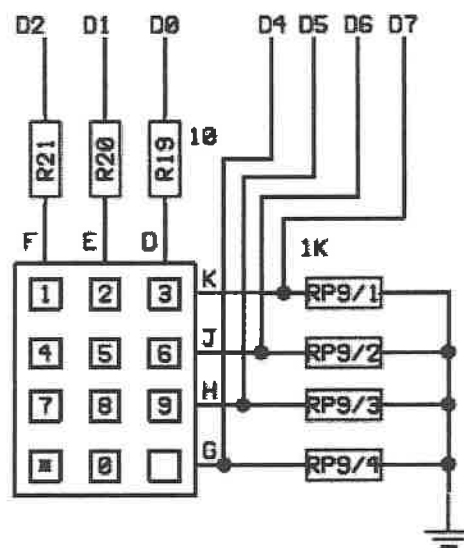


Figure Q2.(a)

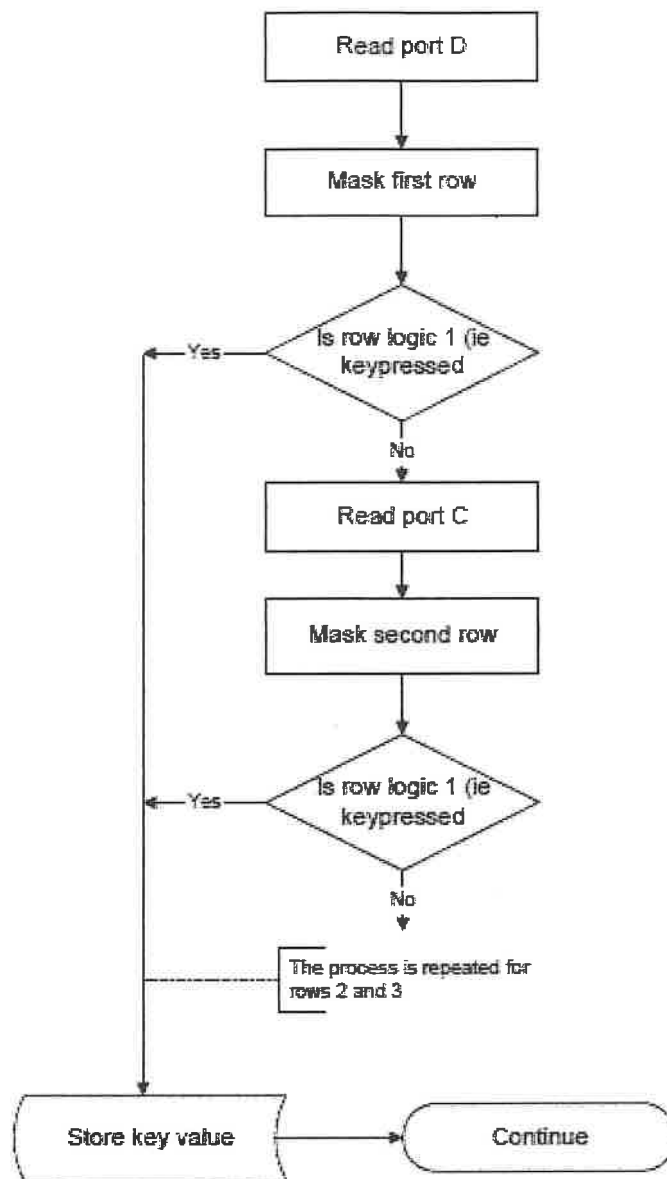


Figure Q2.(b)

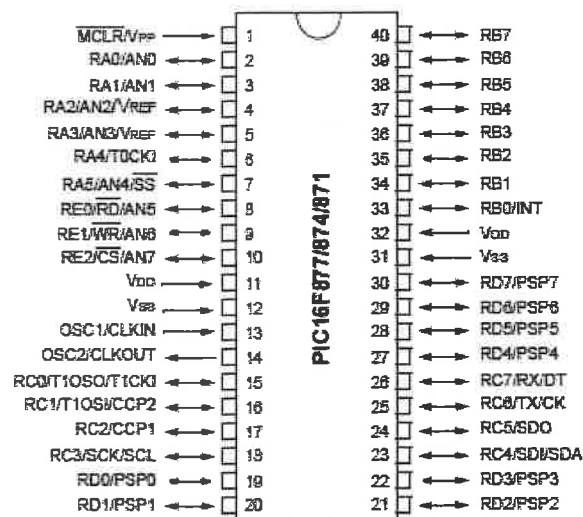


Figure Q2 (c): Pin diagram of 16F8XX.

Table Q2: Instruction set

Mnemonic, Operands	Description	Cycles	14-Bit Opcode			Status Affected	Notes		
			MSb	LSb					
BYTE-ORIENTED FILE REGISTER OPERATIONS									
ADDWF	f, d	Add W and f	1	00	0111	ffff	ffff	C,DC,Z	1,2
ANDWF	f, d	AND W with f	1	00	0101	ffff	ffff	Z	1,2
CLRF	f	Clear f	1	00	0001	1fff	ffff	Z	2
CLRWF	-	Clear W	1	00	0001	ffff	ffff	Z	
COMF	f, d	Complement f	1	00	1001	ffff	ffff	Z	1,2
DECf	f, d	Decrement f	1	00	0011	ffff	ffff	Z	1,2
DECFSZ	f, d	Decrement f, Skip if 0	1(2)	00	1011	ffff	ffff		1,2,3
INCF	f, d	Increment f	1	00	1010	ffff	ffff	Z	1,2
INCFSZ	f, d	Increment f, Skip if 0	1(2)	00	1111	ffff	ffff		1,2,3
IORWF	f, d	Inclusive OR W with f	1	00	0100	ffff	ffff	Z	1,2
MOVF	f, d	Move f	1	00	1000	ffff	ffff	Z	1,2
MOVWF	f	Move W to f	1	00	0000	1fff	ffff		
NOP	-	No Operation	1	00	0000	ffff	ffff		
RLF	f, d	Rotate Left f through Carry	1	00	1101	ffff	ffff	C	1,2
RRF	f, d	Rotate Right f through Carry	1	00	1100	ffff	ffff	C	1,2
SUBWF	f, d	Subtract W from f	1	00	0010	ffff	ffff	C,DC,Z	1,2
SWAPF	f, d	Swap nibbles in f	1	00	1110	ffff	ffff		1,2
XORWF	f, d	Exclusive OR W with f	1	00	0110	ffff	ffff	Z	1,2
BIT-ORIENTED FILE REGISTER OPERATIONS									
BCF	f, b	Bit Clear f	1	01	00bb	ffff	ffff		1,2
BSF	f, b	Bit Set f	1	01	01bb	ffff	ffff		1,2
BTFSZ	f, b	Bit Test f, Skip if Clear	1(2)	01	10bb	ffff	ffff		3
BTFSZ	f, b	Bit Test f, Skip if Set	1(2)	01	11bb	ffff	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	100x	kkkk	kkkk	Z	
CALL	k	Call subroutine	2	10	0kkk	kkkk	kkkk		
CLRWDTC	-	Clear Watchdog Timer	1	00	0000	0110	0100	TO,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	TO,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	

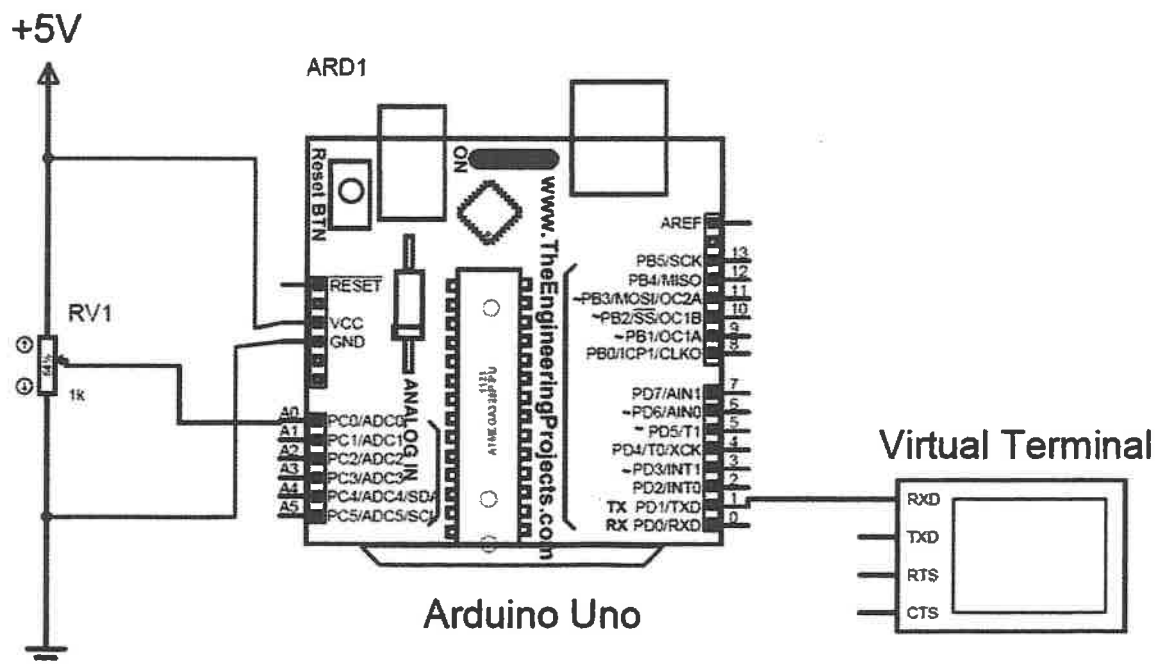


Figure Q3.1: Indoor Air Quality Monitor

Bit	7	6	5	4	3	2	1	0	
(0x81)	ICNC1	ICES1	-	WGM13	WGM12	CS12	CS11	CS10	TCCR1B
Read Write	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Figure Q3.2 TCCR1B – Timer/Counter1 Control Register

```

1  #define F_CPU 16000000UL
2  #include <avr/io.h>
3  #include <avr/interrupt.h>
4  #include <stdlib.h>
5
6  // ADC Initialization
7  void ADC_init() {
8      ADMUX = (1 << REFS0); // AVcc as reference, ADC0
          selected
9      ADCSRA = (1 << ADEN) | (1 << ADPS2) | (1 << ADPS1) |
          (1 << ADPS0); // Prescaler = 128
10 }
11
12 // ADC Read
13 uint16_t ADC_read() {
14     ADCSRA |= (1 << ADSC); // Start conversion
15     while (ADCSRA & (1 << ADSC)); // Wait for conversion
          to finish
16     return ADC;
17 }
18
19 // USART Initialization
20 void USART_init(unsigned int ubrr) {
21
22     /*
23     * USART is configured to operate in Asynchronous
          Normal Mode.
24     * UBRR is a 12-bit register (UBRR0H [4 bits] +
          UBRR0L [8 bits]),
25     */
26
27     UBRR0H = (unsigned char)(ubrr >> 8);
28     UBRR0L = (unsigned char)ubrr;
29     UCSRB = (1 << TXEN0); // Enable transmitter
30     UCSRC = (1 << UCSZ01) | (1 << UCSZ00); // 8-bit data
          , 1 stop bit, no parity
31 }
32
33 // Send a single char over UART
34 void USART_sendChar(char data) {
35     while (!(UCSR0A & (1 << UDRE0))); // Wait for
          transmit buffer to be empty
36     UDR0 = data; // Load data into buffer
37 }
38

```

```

29 // Send a string over UART
30 void UART_sendString(const char* str) {
31     while (*str) {
32         UART_sendChar(*str++);
33     }
34 }
35
36
37 // Timer1 Initialization
38 void Timer1_init() {
39     TCCR1B = (1 << WGM12) | ..... | ..... | .....? //
        CTC mode, Select suitable prescaler
40     OCR1AH = .....;
41     OCR1AL = .....;
42     TCNT1 = 0; // Initialize Timer1 counter to start from 0
43     TIMSK1 |= (1 << OCIE1A); // Enable Timer1 Compare
        Match A interrupt
44 }
45
46 ISR(TIMER1_COMPA_vect) {
47     uint16_t val = ADC_read();
48     char buffer[5];
49     itoa(val, buffer, 10); // Convert the integer ADC
        value 'val' into a decimal (base-10) string and
        store it in 'buffer'.
50
51     USART_sendString("Gas: ");
52     USART_sendString(buffer);
53     USART_sendString("\r\n");
54 }
55
56 // Main function
57 int main(void) {
58     ADC_init();
59     USART_init(51);
60     Timer1_init();
61     sei(); // Enable global interrupts
62
63     while (1){
64
65     }
66 }

```

Figure Q3.3: C Program for the Atmega328p-Based Indoor Air Quality Monitor System

Table Q3.1: Clock Select Bit Configuration for Timer1 Prescaler Selection in CTC

Prescaler	CS12	CS11	CS10
1	0	0	1
8	0	1	0
64	0	1	1
256	1	0	0
1024	1	0	1