Q1) The CPU is performing its task from PORTC to PORTD after the switch SW1 is pressed, the buzzer will start beeping with a certain frequency. The buzzer will only go off after completing 20 beep cycles or when SW2 is pressed.

```
LIST P=18F458
#include <p18f458.inc>
COUNT  EQU 0X21
END

ORG 0x00
GOTO MAIN

ORG 0x08          ;HIGH
BTFSC INTCON, INT0IF
GOTO SW1_ISR
RETFIE

ORG 0x18          ;LOW
BTFSC INTCON3, INT1IF
GOTO SW2_ISR
RETFIE

MAIN:
BCF TRISB,0
CLRF TRISC
CLRF TRISD
BCF PORTB,0
CLRF PORTC
CLRF PORTD
MOVFF PORTC,PORTD
BSF INTCON, INT0IE
BSF INTCON3, INT1IE
BSF INTCON, GIE

MAIN_LOOP:
GOTO MAIN_LOOP

SW1_ISR:
MOVLW 0x14  ;20 CYCLES
MOVWF COUNT
LOOP:
BSF PORTB, 0
CALL DELAY
BCF PORTB, 0
CALL DELAY
DECFSZ COUNT, F
GOTO LOOP
BCF INTCON, INT0IF
RETURN

SW2_ISR:
CLRF COUNT
BCF PORTB, 0
BCF INTCON3, INT1IF
RETURN

DELAY:
MOVLW 0xFF
DELAY_LOOP:
NOP
DECFSZ WREG, F
GOTO DELAY_LOOP
RETURN

END
```

Q2) Write PIC in the whole EEPROM with the following sequence
1st PIC --- 0-2
2nd PIC --- 5-7
3rd PIC --- 10-12
-
-
-
255

```
LIST P=18F458
#include <p18f458.inc>
COUNT EQU 0X20

ORG 0x00
GOTO MAIN

MAIN:
MOVLW 0XFF
MOVWF COUNT
MOVLW 0X00

WRITE_LOOP:
MOVWF EEADR

MOVLW A'P'
MOVWF EEDATA     ; Move to EEDATA for
EEPROM write
CALL WRITE_EEPROM
INCF EEADR, F     ; Increment address
DECFSZ COUNT,F
GOTO END_PROGRAM

MOVLW A'I'
MOVWF EEDATA     ; Move to EEDATA for
EEPROM write
CALL WRITE_EEPROM
INCF EEADR, F     ; Increment address
DECFSZ COUNT,F
GOTO END_PROGRAM

MOVLW A'C'
MOVWF EEDATA     ; Move to EEDATA for
EEPROM write
CALL WRITE_EEPROM
INCF EEADR, F     ; Increment address
DECFSZ COUNT,F
GOTO END_PROGRAM

INCF EEADR, F     ; Increment address      SKIP 1st
DECFSZ COUNT,F
GOTO END_PROGRAM
INCF EEADR, F     ; Increment address      SKIP
2nd
DECFSZ COUNT,F
GOTO END_PROGRAM
INCF EEADR, F     ; Increment address
DECFSZ COUNT,F
GOTO END_PROGRAM
GOTO WRITE_LOOP

WRITE_EEPROM:
BCF EECON1, EEPGD ; Access data EEPROM
BCF EECON1, CFGS  ; Access EEPROM (not
config registers)
BSF EECON1, WREN  ; Enable write operation
BCF INTCON,GIE

MOVLW 0x55
MOVWF EECON2
MOVLW 0xAA
MOVWF EECON2

BSF EECON1, WR
BSF INTCON,GIE
BCF EECON1, WREN  ; Disable write operation
RETURN

END_PROGRAM:
END
```

Q3)As the main CPU task, the buzzer is beeping with a frequency of A MHz and LED1 is turned on. Once the switch is pressed the same buzzer will start beeping with a frequency of B MHz and LED2 would be turned on indicating the frequency change where A<B

```
LIST    P=18F458
#include <p18f458.inc>
COUNT EQU      0X20
ORG 0x00
GOTO MAIN

ORG 0x08
BTFSS INTCON,INT0IF
RETFIE
GOTO ISR

MAIN:
BSF TRISB, 0           ; RB0 as input (button)
BCF TRISB, 1           ; RB1 as output (LED1)
BCF TRISB, 2           ; RB2 as output (LED2)
BCF TRISB, 5           ; RB5 as output (buzzer)

MOVLW 0x20             ; Load initial count (for 50
MHz frequency)
MOVWF COUNT
BCF PORTB,5
BSF INTCON, GIE         ;
BSF INTCON, INT0IE
BSF INTCON,TMR0IE

LOOP:
BSF PORTB, 1           ; Turn on LED1
BCF PORTB, 2           ; Turn off LED2
BTG PORTB,5
CALL TMR0_50MHZ        ; Generate 50 MHz
signal
GOTO LOOP

ISR:
BCF PORTB, 1           ; Turn off LED1
BSF PORTB, 2           ; Turn on LED2
BTG PORTB,5
CALL TMR0_100MHZ       ; Generate 100 MHz
signal

DECFSZ COUNT, F
GOTO ISR               ; If COUNT is not zero, stay in
ISR

BCF INTCON, INT0IF
RETFIE

TMR0_50MHZ:
MOVLW 0X08
MOVWF T0CON
MOVLW 0XFF
MOVWF TMR0L
MOVLW 0XFF
MOVWF TMR0H
BCF INTCON,TMR0IF
BSF T0CON,TMR0ON
AGAIN:
BTFSS INTCON,TMR0IF
BRA AGAIN
BCF T0CON,TMR0ON
RETURN

TMR0_100MHZ:
MOVLW 0X08
MOVWF T0CON
MOVLW 0XFF
MOVWF TMR0L
MOVLW 0XFF
MOVWF TMR0H
BCF INTCON,TMR0IF
BSF T0CON,TMR0ON
AGAIN1:
BTFSS INTCON,TMR0IF
BRA AGAIN1
BCF T0CON,TMR0ON
RETURN

END
```

Q4) Detect the debouncing of a switch. If the switch is pressed for at least 90ms, then the LED1 will turn on. If the switch is pressed for less than 90ms, it will keep on monitoring the switch.

```
LIST P=18F458
#include <p18f458.inc>

DELAY_COUNT1 EQU 0X20
DELAY_COUNT2 EQU 0X21

ORG 0x00
GOTO MAIN

ORG 0x08
BTFSS INTCON,INT0IF
RETFIE
GOTO ISR

MAIN:
BSF TRISB, 0        ; RB0 as input for the switch
BCF TRISB, 1        ; RB1 as output for LED1
BSF INTCON, GIE
BSF INTCON, INT0IE     ; Enable RB0 external
interrupt

LOOP:
GOTO LOOP

ISR:
CALL DELAY_30MS      ; Wait for 30ms
BTFSS PORTB, 0       ; Check if RB0 is still pressed
RETFIE
CALL DELAY_30MS      ; Wait for 30ms
BTFSS PORTB, 0       ; Check if RB0 is still pressed
RETFIE
CALL DELAY_30MS      ; Wait for 30ms
BTFSS PORTB, 0       ; Check if RB0 is still pressed
RETFIE

BSF PORTB, 1         ; Turn on LED1
BCF INTCON, INT0IF
RETFIE

DELAY_30MS:
MOVLW 0X08
MOVWF T0CON
MOVLW 0X6D
MOVWF TMR0L
MOVLW 0X84
MOVWF TMR0H
BCF INTCON,TMR0IF
BSF T0CON,TMR0ON
AGAIN:
BTFSS INTCON,TMR0IF
BRA AGAIN
BCF T0CON,TMR0ON
RETURN

END
```

Q5) As the main CPU task LED1 is ON continuously. Write a code to turn the LED1 OFF and LED2 ON with only the correct sequence of 231. As soon as the wrong switch is pressed, the system should return to its main task and not proceed further.

```
LIST P=18F458
#INCLUDE <P18F458.INC>

ORG 0X00
GOTO MAIN

ORG 0X08
BTFSS INTCON, RBIF
RETFIE
GOTO CHECK2

MAIN:
CLRF PORTB          ; CLEAR PORTB
CLRF PORTC          ; CLEAR PORTC
BSF TRISB, 4        ; SET RB4 AS INPUT (SWITCH
1)
BSF TRISB, 5        ; SET RB5 AS INPUT (SWITCH
2)
BSF TRISB, 6        ; SET RB6 AS INPUT (SWITCH
3)
BCF TRISC, 0        ; SET RC0 AS OUTPUT (LED1)
BCF TRISC, 1        ; SET RC1 AS OUTPUT (LED2)
BSF PORTC,0             ; TURN LED1 ON

BSF INTCON, GIE
BSF INTCON, RBIE

MAIN_LOOP:
GOTO MAIN_LOOP

CHECK2:
BTFSS PORTB, 5
GOTO RESET_SEQUENCE
GOTO CHECK1

CHECK1:
BTFSS PORTB, 4
GOTO RESET_SEQUENCE
GOTO CHECK3

CHECK3:
BTFSS PORTB, 6
GOTO RESET_SEQUENCE
GOTO SEQUENCE_CORRECT

SEQUENCE_CORRECT:
BCF PORTC,0             ; TURN OFF LED1
BSF PORTC,1             ; TURN ON LED2
CALL DELAY
BCF PORTC,1
GOTO RESET_SEQUENCE

RESET_SEQUENCE:
BCF INTCON, RBIF
RETFIE

DELAY:
MOVLW 0XFF
DELAYLOOP:
DECFSZ WREG, F
GOTO DELAYLOOP
RETURN

END
```

Q6) Make a traffic light system in which RED is on for 5m sec, Yellow is on for 2m sec and Green is on for 5m sec. There is a pedestrian button which when pressed will make the red light freeze for 10m sec and then the normal sequence continues for the traffic light

```
LIST P=18F458
#INCLUDE <P18F458.INC>

ORG 0X00
GOTO MAIN

ORG 0X08
BTFSS INTCON,RBIF
RETFIE
GOTO PEDESTRIAN_MODE

MAIN:
CLRF PORTB
CLRF PORTC
BSF TRISB, 4         ; SET RB4
AS INPUT (PEDESTRIAN
BUTTON)
BCF TRISC, 0         ; SET RC0
AS OUTPUT (RED LIGHT)
BCF TRISC, 1         ; SET RC1
AS OUTPUT (YELLOW LIGHT)
BCF TRISC, 2         ; SET RC2
AS OUTPUT (GREEN LIGHT)
BSF INTCON, GIE
BSF INTCON, RBIE

MAIN_LOOP:
CALL RED_LIGHT       ; RED
LIGHT FOR 5ms
CALL YELLOW_LIGHT    ;
YELLOW LIGHT FOR 2ms
CALL GREEN_LIGHT     ;
GREEN LIGHT FOR 5ms
GOTO MAIN_LOOP

RED_LIGHT:
BSF PORTC,0
MOVLW 0X08
MOVWF T0CON
MOVLW 0XCF
MOVWF TMR0L
MOVLW 0X2C
MOVWF TMR0H
BCF INTCON,TMR0IF
BSF T0CON,TMR0ON
CALL AGAIN
BCF PORTC,0
RETURN

YELLOW_LIGHT:
BSF PORTC,1
MOVLW 0X08
MOVWF T0CON
MOVLW 0XEC
MOVWF TMR0L
MOVLW 0X78
MOVWF TMR0H
BCF INTCON,TMR0IF
BSF T0CON,TMR0ON
CALL AGAIN
BCF PORTC,1
RETURN

GREEN_LIGHT:
BSF PORTC,2
MOVLW 0X08
MOVWF T0CON
MOVLW 0XCF
MOVWF TMR0L
MOVLW 0X2C
MOVWF TMR0H
BCF INTCON,TMR0IF
BSF T0CON,TMR0ON
CALL AGAIN
BCF PORTC,2
RETURN

PEDESTRIAN_MODE:
BSF PORTC,0
MOVLW 0X08
MOVWF T0CON
MOVLW 0X9E
MOVWF TMR0L
MOVLW 0X58
MOVWF TMR0H
BCF INTCON,TMR0IF
BSF T0CON,TMR0ON
CALL AGAIN
BCF PORTC,0
BCF INTCON,RBIF
RETFIE

AGAIN:
BTFSS INTCON,TMR0IF
BRA AGAIN
BCF T0CON,TMR0ON
RETURN

END
```