Digital Assignment - 2

Allen Ben Philipose
18BIS0043
ECE3031 - C2

1. Write a program to perform the following:

a) Keep monitoring the P1.2 bit until it becomes high

b) When P1.2 becomes high, write value 45H to port 0

c) Send a high-to-low pulse to P2.3

ans

```
SETB   P1.2                    // Input
MOV    A, #45H
AGAIN: JNB    P1.2, AGAIN      // out when it is 1
       MOV    P0, A
       SETB   P2.3
       CLR    P2.3             // H-to-L
```

2. Write a program to get the 'x' value from P1 and send x2 to P2, continuously. $x2 = x^2$

ans

```
        ORG 0000H
            MOV DPTR, #300H        // Load table
            MOV A, #0FFH
            MOV P1, A
BACK:       MOV A, P1
            MOV A, @A+DPTR         // Get x² from table
            MOV P2, A
            SJMP BACK

        ORG 300H
        XSQR_TABLE:
        DB : 0,1,4,9,16,25,36,49,64,81
        END
```

3. Write a program to read the temperature and according to value, allocate them to

A = T if T = 75

R1 = T if T < 75

R2 = T if T > 75

ans

```
        MOV P1, #0FFH          // Input
        MOV A, P1
        CJNE A, #75, OVER      // Jump if A not 75
        SJMP EXIT
```

```
OVER:   JNC   NEXT      // CY=0, A>75
        MOV   R1,A      // Save in R1
        SJMP  EXIT      // else
NEXT:   MOV   R2,A      // save in R2
EXIT:   END
```

4. Assume XTAL = 11.0592 MHz. What value do we need to load the timer's register if we want to have a delay of 5ms? Show the program for timer 0 to create a pulse width of 5ms on P2.3.

ans

→ XTAL = 11.0592 MHz

→ 1 MC = 1/12, 11.0592/12 = 921.6 kHz = F

∴ T = 1/921.6 kHz = 1.085 µsec

→ 5ms / 1.085µs = 4608 clocks.

→ 65536 - 4608 = EE00H, TH = EE and TL = 00

```
        CLR   P2.3
        MOV   TMOD, #01     // Timer 0, 16-bit
```

```
HERE:  MOV  TL0, #00H          // #EE00H
       MOV  TH0, #0EEH
       SETB P2.3
       SETB TR0                // Start timer
AGAIN: JNB  TF0, AGAIN         // Monitor timer flag
       CLR  TR0
       CLR  TF0
```

5. XTAL = 11.0592 MHz, square wave of 2kHz frequency on pin P1.5

ans

→ T = 1/f = 1/2kHz = 500µs = Period

→ 500/2 = 250µs in high and low

→ 250µs/1.085µs = 230

→ 65536 - 230 = 65306 = FF1AH

TL = 1A, TH = FF

```
       MOV TMOD, #01           // Timer 0, 16-bit
AGAIN: MOV TL1, #1AH           // # FF1AH
       MOV TH1, #0FFH
       SETB TR1                // Start timer
BACK:  JNB  TF1, BACK
       CLR  TR1
       CLR  P1.5
       CLR  TF1
       SJMP AGAIN
```

60 classmate

6. Assuming the clock pulses are fed to pin T1, write a program for counter 1 in mod 2 to count the pulses and display the state of the TL1 count on P2, which connects to 8 LEDS

ans

```
        MOV TMOD,#60H        // Counter 1, mode 2
        MOV TH1,#0DH
        SETB P3.5
AGAIN:  SETB TR1
BACK:   MOV A,TL1
        MOV P2,A             // Display on port-2
        JNB TF1,BACK         // Repeat
        CLR TR1
        CLR TF1
        SJMP AGAIN
```

7. Write a program to transfer "YES" serially at 9600 baud, 8-bit data, 1-stop bit, do this continuously.

ans

```
MOV TMOD,#20H        // Timer 1, mod 2
MOV TH1,#-3          // 9600 Baud rate
MOV SCON,#50H        // 8-bit, 1-stop
SETB TR1
```

```
AGAIN:  MOV A,#"Y"
        ACALL TRANS
        MOV A,#"E"
        ACALL TRANS
        MOV A,#"S"
        ACALL TRANS
        SJMP AGAIN
TRANS:  MOV SBUF,A
HERE:   JNB TI, HERE      // wait for last bit
        CLR TI            // get ready for next byte
        RET
```

8. Receive bytes of data serially and put them in P1. set the baud rate to 4800, 8-bit data and 1 stop bit

ans

```
        MOV TMOD, #20H    // Timer 1, mod 2
        MOV TH1, #-6      // 4800 Baud rate
        MOV SCON, #50H    // 8-bit, 1-Stop
        SETB TR1
HERE:   JNB RI, HERE
        MOV A, SBUF       // Save incoming in A
        MOV P1, A
        CLR RI            // Get ready for next byte
        SJMP HERE
```

9. Assume that the 8051 port is connected to the com port of IBM PC, and weie using teminal.exe to send and receive data serially. P1 - LED, P2 - switches. Write program to:

a) Send PC "we ale ready"

b) Receive any data and put it on the LEDS

c) Get data from switches and send to PC serially

Do part (a) once and (b), (c) continuously with Baud rate = 4800

ans

```
        ORG  0000H
        MOV  P2, #0FFH
        MOV  TMOD, #20H        // Timer 1, mode 2
        MOV  TH1, #0FAH        // 4800 Baud rate
        MOV  SCON, #50H
        SETB TR1               // Start timer
        MOV  DPTR, #MD
H_1:    CLR  A
        MOV  A, @A+DPTR
        JZ   B_1               // If last character - out
        ACALL SEND             // else call transfer

        INC  DPTR
        SJMP H_1
```

```
B_1:  MOV  A, P2          // Read switch
      ACALL SEND
      ACALL RECV
      MOV P1, A           // Display LED
      SJMP B_1
SEND:  MOV SBUF, A        // Load data
H_2:  JNB  TI, H_2        // Stay till full transfered
      CLR TI
      RET

RECV:  JNB RI, RECV
       MOV A, SBUF
       CLR RI
       RET
MD:  DB  "WE ARE READY", 0
     END
```

10. Send message "The Earth is but one country" to serial port. SW connected to pin P1.2. Monitor and set baud rate:

SW = 0, 4800 Baud rate

SW = 1, 9600 Baud rate

XTAL = 11.0592 MHz, 8-bit data, 1-stop bit

ans

```
        SW   BIT  P1.2
        ORG  0000H
MAIN:   MOV  TMOD, #20H
        MOV  TH1, #-6          // 4800 Baud rate
        MOV  SCON, #50H
        SETB TR1
        SETB SW
S1:     JNB  SW, SLOWSP
        MOV  A, PCON           // Read PCON
        SETB ACC.7             // Set SMOD high for 9600
        MOV  PCON, A           // Write PCON
        SJMP OVER

SLOWSP: MOV  A, PCON
        SETB ACC.7             // Set SMOD low for 4800
        MOV  PCON, A
OVER:   MOV  DPTR, #MESS1
FN:     CLR  A
        MOVC A, @A+DPTR
        JZ   S1
        ACALL SENDCOM
        INC  DPTR
        SJMP FN
```

```
SENDCOM:  MOV SBUF, A
HERE:     JNB TI, HERE
          CLR TI
          RET
MESS1:    DB "The Earth is but one country", 0
          END
```

11. Continuously get 8-bit from P0 and sends it to P1 while simultaneously creating a square wave of 200μs period on pin P2.1. Use timer 0 to create the square wave. XTAL = 11.0592 MHz

ans

```
          ORG 0000H
          LJMP MAIN
          ORG 000BH          // Timer 0 interrupt
          CPL P2.1
          RETI               // Return from ISR
          ORG 0030H
MAIN:     MOV TMOD, #02H     // Timer 0, mode 2
          MOV P0, #0FFH      // Input port
          MOV TH0, #-92
          MOV IE, 82H
          SETB TR0
```

```
BACK:  MOV A, P0
       MOV P1, A
       SJMP BACK
       END
```

12. INT1 pin is connected to a switch that is high. Whenever it goes low, LED on, which is connected to P1.3. As long as the switch is in low, the LED should stay on.

ans

```
       ORG  0000H
       LJMP MAIN
       ORG  0013H          // INT1 ISR
       SETB P1.3
       MOV R3, #255
BACK:  DJNZ R3, BACK       // Keep LED ON
       CLR P1.3
       RETI

       ORG 0030H
MAIN:  MOV IE, 84H         // Enable external INT1
HERE:  SJMP HERE           // Until Interrupted
       END
```

13. Pin 3.3 (INT1) is connected to a pulse generator, write a program in which the falling edge of the pulse will send high to P1.3

ans.

```
        ORG  0000H
        LJMP MAIN
        ORG  0013H          // INT1 ISR
        SETB P1.3
        MOV R3,#255
BACK:   DJNZ R3, BACK
        CLR  P1.3
        RETI                // Return from ISR
        ORG  0030H
MAIN:   SETB TCON.2         // Edge triggered
        MOV IE, 84H         // Enable external INT1
HERE:   SJMP HERE
        END
```

14. Read data from P1 and continuously write to P2, while giving a copy to serial COM port to be transfered serially. Baud rate = 9600
XTAL = 11.0592 MHz

ans

```
        ORG   0000H
        LJMP  MAIN
        ORG   0023H        // Serial INT ISR
        LJMP  SERIAL
        ORG   0030H

MAIN:   MOV   P1,#0FFH
        MOV   TMOD,#20H    // Timer 1, auto reload
        MOV   TH1,#0FDH    // 9600 Baud rate
        MOV   SCON,#50H    // Ren enabled, 8-bit
        MOV   IE,  90H     // Enable serial int
        SETB  TR1
BACK:   MOV   A,P1
        MOV   SBUF,A       // Copy sent
        MOV   P2,A
        SJMP  BACK
        ORG   0100H
SERIAL: JB    TI,TRANS
        MOV   A,SBUF
        CLR   RI
        RETI               // return from ISR
TRANS:  CLR   RI
        RETI
        END
```

15. Write a program using interrupts for:

a) Receive data serially and send to P0

b) P1 port read and transmit serial, a copy given to P2

c) Make timer 0 to generate a square wave of 5kHz frequency on P0.1

XTAL = 11.0592MHz Baud rate = 4800

ans

```
        ORG 0000H
        LJMP MAIN
        ORG 000BH          // ISR for timer 0
        CPL P0.1
        RETI

        ORG 0023H
        LJMP SERIAL
        ORG 0030H
MAIN:   MOV P1,#0FFH       // Input port
        MOV TMOD,#22H      // Timer 1, mod 2
        MOV TH1,#0F6H      // 4800 Baud Rate
        Mov SCON,#50H      // Ren enabled
        Mov TH0,#-92       // 5kHz wave
```

```
        MOV IE, 92H
        SETB TR1          // start both timers
        SETB TR1
BACK:   MOV A, P1
        MOV SBUF, A       // giving copy
        MOV P2, A
        SJMP BACK

        ORG 0100H
SERIAL: JB TI, TRANS
        MOV A, SBUF
        MOV P0, A
        CLR RI
        RETI
TRANS:  CLR TI
        RETI
        END
```