Format of Tmod register

first(lower four bits) =t1(timer1)
second four=t0
if c/t(bar)=0 it acts as timer
if gate bit=1,we r gng to control hardware ckt
if gate bit=0,we r gng to control software ckt
1 machine cycle consist of 12 clock cycles
timer counts how many machine cycles
MODE1 INTIALIZING

gate	c/t-	m1	m0	gate	c/t-	M1	M0
0	0	0	0	0	0	0	1
TCON							TCON
HIGHER	NIBBLE	FOR CNTRL	TIMERS				
TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

M1	M0	Mode	Timer
0	0	Mode0	13 bit timer
0	1	Mode1	16 bit timer
1	0	Mode2	8 bit auto load
1	1	Mode 3	8 bit split timer

SPECIAL FUNCTION REGISTERS:
TL0 AND TH0 (TIMER ZERO LOWER BYTE AND HIGHER BYTE)

TCON(to start timer and to stop timer)
TR0=1 TO START
TR0=0 TO STOP
WHEN OVERFLOW OCCURS TFO BECOMES 1 (TF0=1)
WHEN COUNT EXCEEDS FFFF IT BECOMES COUNT 0 CARRY 1
for delay
ffff-1000=effff delay in terms of machines cycles
ffff-ffff=0000 for more delay

```
mov tmod,#01h
mov TL0,#00h
mov TH0,#00h
goto: mov p2,#0ffh
acall delay
```

```
mov p2,#00h
acall delay
sjmp goto
delay :setb TR0
here:jnb TF0,here
clr tf0
clr tr0
ret
end
```

2.using switch

```
mov tmod,#00h
mov TL0,#00h
mov TH0,#00h

check:          jnb p1.0,on
                mov p2,#00h(leds off state)
                acall delay
                sjmp check

on:             mov p2,#0ffh
                acall delay
                sjmp check
delay:          setb tr0
                jnb tf0,here
                clr tf0
                clr tr0
                ret
                end
```

2.using switch LED Scrolling

```
mov tmod,#00h
mov TL0,#00h
mov TH0,#00h
mov a,#01h
mov b,10h
check:          jnb p1.0,rolleft
                sjmp rolright

rolleft:        mov p2,@a
                rl a
                acall delay
                jnb p2.7,rolleft
                sjmp check


rolright:       mov p2,@b
                rr b
                acall delay
                jnb p2.0,rolright
                sjmp check

delay:          setb tr0
                jnb tf0,here
                clr tf0
                clr tr0
                ret
                end
```

```
/ ********** SERIAL COM TESTING PROGRAM  **********/

                MOV SCON,#50H
                MOV TMOD,#20H
                MOV TH1,#0FDH
                SETB TR1

                MOV A,#'S'
                MOV SBUF,A
                ACALL SEND
                MOV A,#'D'
                MOV SBUF,A
                ACALL SEND

SEND:                   JNB TI,SEND
                CLR TI
                RET
END
```

======================== TRANSFERRING THE DATA==================

```
                MOV SCON,#50H
                MOV TMOD,#20H
                MOV TH1,#0FDH
                SETB TR1

                MOV DPTR,#TEXT
NEXT:           CLR A
                MOVC A,@A+DPTR
                JZ EXIT
                ACALL SEND
                INC DPTR
                SJMP NEXT


SEND:                   MOV SBUF,A
WAIT:           JNB TI,WAIT
                CLR TI
                RET

EXIT:           SJMP EXIT
ORG 1000H
        TEXT: DB "HELLO WELCOME TO MY WORLD",00H
                END
```

================= RECEIVING THE DATA ================================

```
        MOV SCON,#50H
        MOV TMOD,#20H
        MOV TH1,#0FDH
        SETB TR1

BACK:           ACALL RECEIVE
        MOV P2,A
        SJMP BACK


RECEIVE:    JNB  RI, RECEIVE
          MOV A,SBUF
          CLR RI
         RET
END
```

================= RECEPTION AND RE TRANSMITTING THE DATA ========

```
        MOV SCON,#50H
        MOV TMOD,#20H
        MOV TH1,#0FDH
        SETB TR1

BACK:           ACALL RECEIVE
```

```
            MOV P2,A
            ACALL SEND
            SJMP BACK


RECEIVE:    JNB  RI, RECEIVE
               MOV A,SBUF
               CLR RI
              RET

SEND:      MOV SBUF,A
WAIT: JNB TI,WAIT
            CLR TI
            RET
             END
===================== T0 INTERRUPT===========

            ORG 0000H
            LJMP MAIN

            0RG 000BH
            CPL P1.7
            RETI


MAIN:MOV IE,#82H
            MOV TMOD,#01H
            MOV TL0,#00H
            MOV TH0,#00H

BACK:             MOV P2,#00h
            acall delay
            MOV P2,#0FFH
            acall delay
            SJMP BACK

DELAY:
            SETB TR0                  ;;0000--0001-0002-0003-FFFF--+1-TF0--T0 INT---0000
WAIT: JNB TF0,WAIT
            CLR TR0
            CLR TF0
            RET
END

=================== SRL TI INT =====================


                        ORG 0000H
                        LJMP MAIN

                        ORG 0023H
                        MOV P1,#00H
```

```
                LJMP SRL

                ORG 0030H
MAIN:           MOV IE,#90H
                MOV SCON,#50H
                MOV TMOD,#20H
                MOV TH1,#0FDH
                SETB TR1

BACK:                   MOV P1,A
                SJMP BACK

SRL:            JNB TI,SRL
SEND:                   MOV SBUF,A
                CLR TI
                RETI

                END
```

/=============== SRL_RI_ INT ============================

```
                ORG 0000H
                LJMP MAIN

                ORG 0023H
                MOV P1,#00H
                LJMP SRL

                ORG 0030H
MAIN:           MOV IE,#90H
                MOV SCON,#50H
                MOV TMOD,#20H
                MOV TH1,#0FDH
                SETB TR1

BACK:                   MOV P1,A
                SJMP BACK

SRL:            JNB RI,SRL
                MOV A,SBUF
                CLR RI
                RETI
                END
```

/================== EXT INT0 - EDGE TRIGGER===================11/

```
                        ORG 0000H
                        LJMP MAIN
```

```
            ORG 0003H
            CPL P1.0
            RETI

            ORG 0030H
MAIN:       MOV IE,#81H
            SETB IT0
HERE:       SJMP HERE
            END
```

====================== APPLICATIONS ==================

LED INTERFACING: Light Emitting Diode

The LED is a PN-junction diode which emits light when an electric current passes through it in the forward direction. In the LED, the recombination of charge carrier takes place. The electron from the N-side and the hole from the P-side are combined and gives the energy in the form of heat and light.

APPLICATIOS:
----
* TV Backlighting
* Smartphone Backlighting
* LED displays
* Automotive Lighting
* Dimming of lights
* Indicators and signs
* lighting
* Biological detections

=================== SCROLL LEDS FROM LSB TO MSB =====================

```
      MOV TMOD,#10H
      MOV TL1,#00H
      MOV TH1,#00H

      MOV P2,#00h


      MOV A,#01H
BACK:       MOV P2,A
      ACALL DELAY
      ACALL DELAY
      RL A
      SJMP BACK

      DELAY:      SETB TR1
      WAIT: JNB TF1,WAIT
                  CLR TR1
                  CLR TF1
                  RET

      END
```

Assignment:
 scroll leds from msb to lsb
blink the leds in even and odd pattern
blinking

==

SWITCHES:
NONC---   normally open normally closed
SPST
SPDT
DPST
DPDT

push buttons

KEYBOARDS
RELAY
ISOLATOR
IR

```
MOV P1,#0FFH                          ---- SET I/P AS 1
MOV P2,#00H                           ---- SET O/P AS 0
                                1---- OPEN
                                0=== CLOSED
task1:   RURN ON all leds iF THE SWItch is closed,if not turn off all leds

        MOV P1,#0FFH
        MOV P2,#00H

BACK:           JNB P1.0, ON_LEDS
        MOV P2,#00H
        SJMP BACK

ON_LEDS: MOV P2,#0FFH
        SJMP BACK
END


        BLINK THE LEDS WHEN YOU GET INT0

--------------------------------- INTO LED BLINKING --------------------
ORG 0000H
        LJMP MAIN

        ORG 0003H
        LJMP TASK

ORG 0030H
MAIN:MOV P2,#00H
```

```
        MOV IE,#81H

STAY:   SJMP STAY


TASK:   MOV R7,#05H
NEXT:   MOV P2,#0FFH
        ACALL DELAY
        MOV P2,#00H
        ACALL DELAY
        DJNZ R7,NEXT
        RETI

DELAY:
        MOV R0,#0FFH
        MOV R1,#0FFH
        MOV R2,#03H
L1:     DJNZ R0,L1
        DJNZ R1,L1
        DJNZ R2,L1
        RET
END
----------------------------------------------------------

============== if switch is connected to vcc ================
        MOV P1,#00H
        MOV P2,#00H

BACK:           JB P1.0, ON_LEDS
        MOV P2,#00H
        SJMP BACK

ON_LEDS: MOV P2,#0FFH
        SJMP BACK
END
----------------------------------------------------------


        DC MOTOR INTERFACING:
=====================================================
--------------------- REMOTE CAR--------------


                L1 EQU P2.0
                L2 EQU P2.1
                N1 EQU P2.2
                N2 EQU P2.3
                S1 EQU P1.0
                S2 EQU P1.1

                MOV P1,#0FFH
                MOV P2,#00H
```

```
CHECK:                  JNB S1,CLK
                JNB S2,ANT_CLK
                SJMP CHECK

CLK:            SETB L1
                CLR L2
                SETB N1
                CLR N2
                SJMP CHECK
ANT_CLK:                CLR L1
                SETB L2
                CLR N1
                SETB N2
                SJMP CHECK
                END
```

===================== VEHICLE CONTROL USING WIRELESS COM =================

```
        GSM
        WIFI
        BLUETOOTH
        IOT             (SERIAL)
        IR

        F -----FORWARD
        B -----BACKWARD (REVERSE)
        L ----- LEFT
        R ----- RIGHT
        S ----- STOP
```

====================== CODE ====================================

```
                L1 EQU P2.0
                L2 EQU P2.1
                N1 EQU P2.2
                N2 EQU P2.3
                S1 EQU P1.0
                S2 EQU P1.1

                MOV P2,#00H

                MOV SCON,#50H
                MOV TMOD,#20H
                MOV TH1,#0FDH
                SETB TR1

                MOV DPTR,#TEXT1
                ACALL SEND
MAIN:           ACALL RECEIVE
                        ACALL TRANS

                CJNE A,#'F',REVERSE
                SETB L1
                CLR L2
                SETB N1
                CLR N2
                MOV DPTR,#T2
                ACALL SEND
                SJMP MAIN

REVERSE:        CJNE A,#'B', LEFT
                CLR L1
                SETB L2
                CLR N1
                SETB N2
                MOV DPTR,#T3
                ACALL SEND
                SJMP MAIN

LEFT:   CJNE A,#'L', RIGHT
                MOV P2,#01H
                MOV DPTR,#T4
                ACALL SEND
                SJMP MAIN
RIGHT:          CJNE A,#'R',STOP
                MOV P2,#04H
                MOV DPTR,#T5
                ACALL SEND
                SJMP MAIN
STOP: CJNE A,#'S',MAIN
                MOV P2,#00H
                MOV DPTR,#T6
                ACALL SEND
                SJMP MAIN
SEND:CLR A
                MOVC A,@A+DPTR
                JZ EXIT
                ACALL TRANS
                INC DPTR
                SJMP SEND
EXIT:  RET

RECEIVE:                JNB  RI,RECEIVE
                MOV A,SBUF
                CLR RI
                RET
TRANS:                  MOV SBUF,A
HERE:JNB TI,HERE
                CLR TI
                RET

        ORG 1000H
TEXT1:DB "   VEHICLE CONTROL USING BLUETOOTH TECH    ",00H
```

```
          T2:DB "     VEHICLE MOVING IN FORWARD DIRECTION     ",00H
                    T3:DB "     VEHICLE MOVING IN BACKWARD DIRECTION    ",00H
                              T4:DB "    VEHICLE MOVING IN LEFT DIRECTION       ",00H
                                        T5:DB "    VEHICLE MOVING IN RIGHT DIRECTION  ",00H
                                                  T6:DB "   VEHICLE STOP        ",00H


END

==========================================================================================
```

================== STEPPER MOTOR INTERFACING
================================

FOR ANTI CLOCKWISE:

```
          MOV A,#33H
MAIN:     MOV P2,A
          ACALL DELAY
          RL A
          SJMP MAIN



DELAY:              MOV R0,#0FFH
          MOV R1,#0FFH
          MOV R2,#0AH
L1:       DJNZ R0,L1
          DJNZ R1,L1
          DJNZ R2,L1
          RET

END
```

FOR CLOCKWISE:

```
          MOV A,#99H
MAIN:     MOV P2,A
          ACALL DELAY
          RR A
          SJMP MAIN



DELAY:              MOV R0,#0FFH
          MOV R1,#0FFH
          MOV R2,#0AH
L1:       DJNZ R0,L1
          DJNZ R1,L1
          DJNZ R2,L1
```

```
                RET

END
```

==================== RELAY ================== 22.05.2020

HOME AUTOMATION
==================

```
                M---->N  = ENTRY
                N---->M  = EXIT

                M EQU P1.0
                N EQU P1.1
                T EQU P2.0

                MOV P1,#0FFH
                MOV P2,#00H

CHECK:                  JNB M,ENTER
                JNB N, LEAVE
                SJMP CHECK

ENTER:                  JNB N,LAMP_ON
                SJMP ENTER

LAMP_ON:        SETB T
                SJMP CHECK

LEAVE:                  JNB M,LAMP_OFF
                SJMP LEAVE

LAMP_OFF:       CLR T
                SJMP CHECK
END
```
==

TASK: HOME AUTOMATION WITH LDR

7 SEGMENT DISPLAY:
=== 23.05.2020

```
                GFEDCBA
                11111001-----1
                ----------------2 hex values
```

bcd 7 segment display driver(7447)
4input 8 output lines

send direct values to the bcd driver, internally it will translate them into their equvalent hex vaules.

==========WAVEFORM GENERATION=================

GENERATE A SQUAREWAVE:

```
-----------------------------------
BACK:       SETB P2.0
           ACALL DELAY
             ACALL DELAY
           CLR P2.0
           ACALL DELAY
           SJMP BACK

DELAY:
                 MOV R0,#0FFH
                 MOV R1,#0FFH
L1:     DJNZ R0,L1
      DJNZ R1,L1
                  RET
                  END
-----------------------------------------------------
```

```
 BACK:        SETB P2.0-----------------1
        ACALL DELAY1 ----------TD  70%
 CLR P2.0  -----------------0
ACALL DELAY2  ----------30 %
SJMP BACK


DELAY1:
DELAY2:

END
```

====================================

LCD INTERFACING:

16x2 LCD

It consist of 16 pins

3 are control pins
8 are data lines

rs--reg selection-
0-----cmd reg
1-----data reg
rw -- read/write
rw-- 0
en-- enable--- sending data or cmds , enable lcd
1 TD 0

PART1----initilization
PART2----main prog to display something

;; INIITLIZATION

38H----16X2 LCD
06H---INCREMENT THE CURSOR POSITION
0EH--- DISPLAY ON CURSOR OFF
01H --- TO CLEAR THE LCD PREVIOUS DATA
80H --- TO PRINT DATA FROM 1ST ROW & 1ST COLUMN
0C0H-- 2ND ROW AND 1ST COL

```
RS EQU P1.0
RW EQU P1.1
EN EQU P1.2
LCD EQU P2

MOV A,#38H
ACALL CMD
MOV A,#06H
ACALL CMD
MOV A,#0EH
ACALL CMD
MOV A,#01H
ACALL CMD
MOV A,#80H
ACALL CMD

MOV DPTR,#TEXT1
BACK:CLR A
MOVC A,@A+DPTR
JZ STOP
ACALL SEND
INC DPTR
SJMP BACK

STOP: SJMP STOP

CMD:MOV LCD,A
CLR RS
CLR RW
SETB EN
ACALL DELAY
CLR EN
RET

SEND:     MOV LCD,A
          SETB RS
```

```
CLR RW
SETB EN
ACALL DELAY
CLR EN
RET


DELAY:  MOV R0,#0FFH
MOV R1,#0FFH
L1: DJNZ R0,L1
DJNZ R1,L1
RET

ORG 1000H
TEXT1: DB "SOMETHING",00H
END
```