LOOPING IN 8051 AND TIME DELAY CALCULATION

Machine Cycle

- 1 MC = 12 T-states
- 1 T-states = 1/XTAL
- 1MC = 12/XTAL

- For XTAL = 11.0592 MHz
- 1MC = 12 / 11.0592 = 1.085 μs

- For XTAL = 12 MHz
- 1MC = 12 / 12 = 1 μs

Some Instructions and Corresponding Machine Cycle

MOV Rn, # DATA	1MC
DEC Rn	2MC
DJNZ Rn, REL. ADD.	2MC
LJMP	2MC
SJMP	2MC
NOP	1MC
RET	2MC
LCALL	2MC
ACALL	2MC

Time delay using Single Loop

DELAY:	MOV R3, #0FFH		1
HERE:	DJNZ R3, HERE		2
	RET		2

Total MC = [{255 x 2} + 1 + 2] = 513

XTAL = 11.0592 MHz

1MC = 1.085 μs

Delay = 513 x 1.085 = 556.6 μs

Time delay using Nested Loop

```
DELAY:   MOV  R2, #0FFH        1
AGAIN:   MOV  R3, #0FFH        1
HERE:    DJNZ  R3, HERE        2
         DJNZ  R2, AGAIN       2
         RET                   2
```

Inner loop = [{255 x 2} + 1] = 511 MC

Outer loop = [{511 + 2} x 255] + 1 + 2 = 130818 MC

Total MC = 130818

XTAL = 11.0592 MHz

1MC = 1.085 µs

Delay = 130818 x 1.085 = 141937.5 µs = 142 ms.

Generation of Square wave with 50% duty cycle

```
        ORG 0000H
        LJMP MAIN
MAIN:   ORG 0030H
AGAIN:  CPL P1.3
        ACALL DELAY
        SJMP AGAIN
        END
```

Generation of Square wave of 25% duty cycle

```
        ORG 0000H
        LJMP MAIN
MAIN:   ORG 0030H
AGAIN:  SETB P1.3
        ACALL DELAY
        CLR P1.3
        ACALL DELAY
        ACALL DELAY
        ACALL DELAY
        SJMP AGAIN
        END
```

8051 timer/counter

Timers /Counters Programming

- The 8051 has 2 timers/counters: timer/counter 0 and timer/counter 1. They can be used as

1. The **timer** is used as a time delay generator.
   - The clock source is the **internal** crystal frequency of the 8051.
2. An event **counter**.
   - **External input** from input pin to count the number of events on registers.

Registers Used in Timer/Counter

- TH0, TL0, TH1, TL1
- TMOD (Timer mode register)
- TCON (Timer control register)

Timer Registers

TH0								TL0							
D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

Timer 0

TH1								TL1							
D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

Timer 1

TMOD Register

- Timer mode register: **TMOD**
  - An 8-bit register
  - Set the usage mode for two timers
    - Set lower 4 bits for Timer 0 (Set to 0000 if not used)
    - Set upper 4 bits for Timer 1 (Set to 0000 if not used)
  - Not bit-addressable

(MSB) (LSB)

GATE	C/T	M1	M0	GATE	C/T	M1	M0
Timer 1				Timer 0			

Gate

- Every timer has a mean of starting and stopping.
  - GATE=0
    - **Internal** control
    - The start and stop of the timer are controlled by way of **software.**
    - Set/clear the TR for start/stop timer.
  - GATE=1
    - **External** control
    - The another way of starting and stopping the timer by **software** and **an external source**.
    - Timer/counter is enabled only while the INT pin is high and the TR control pin is set (TR).

C/T (Clock/Timer)

- This bit is used to decide whether the timer is used as a delay generator or an event counter.
- C/T = 0 : timer
- C/T = 1 : counter

M1, M0

- M0 and M1 select the timer mode for timers 0 & 1.

M1	M0	Mode	Operating Mode
0	0	**0**	**13-bit timer** mode 8-bit THx + 5-bit TLx (x= 0 or 1)
0	1	**1**	**16-bit timer** mode 8-bit THx + 8-bit TLx
1	0	**2**	**8-bit auto reload** 8-bit auto reload timer/counter; THx holds a value which is to be reloaded into TLx each time it overflows.
1	1	**3**	Split timer mode

Example

Find the value for TMOD if we want to program timer 0 in mode 2, use 8051 XTAL for the clock source, and use instructions to start and stop the timer.

Solution:

timer 1 timer 0

TMOD= 0000 0010 Timer 1 is not used.

Timer 0, **mode 2**,

C/T = 0 to use XTAL clock source (timer)

gate = 0 to use internal (**software**) start and stop method.

TCON Register

- Timer control register: **TCON**
  - Upper nibble for timer/counter, lower nibble for interrupts
- **TR** (run control bit)
  - TR0 for Timer/counter 0; TR1 for Timer/counter 1.
  - TR is set by programmer to turn timer/counter on/off.
    - TR=0: off (stop)
    - TR=1: on (start)

(MSB) (LSB)

<table>
<tr><td>TF1</td><td>TR1</td><td>TF0</td><td>TR0</td><td>IE1</td><td>IT1</td><td>IE0</td><td>IT0</td></tr>
<tr><td colspan="2">Timer 1</td><td colspan="2">Timer0</td><td colspan="4">for Interrupt</td></tr>
</table>

TCON Register

- **TF** (timer flag, control flag)
  - TF0 for timer/counter 0; TF1 for timer/counter 1.
  - TF is like a carry. Originally, TF=0. When TH-TL roll over to 0000 from FFFFH, the TF is set to 1.
    - TF=0 : not reach
    - TF=1: reach
    - If we enable interrupt, TF=1 will trigger ISR.

(MSB) (LSB)

TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0
Timer 1		Timer0		for Interrupt			

Timer Mode 1

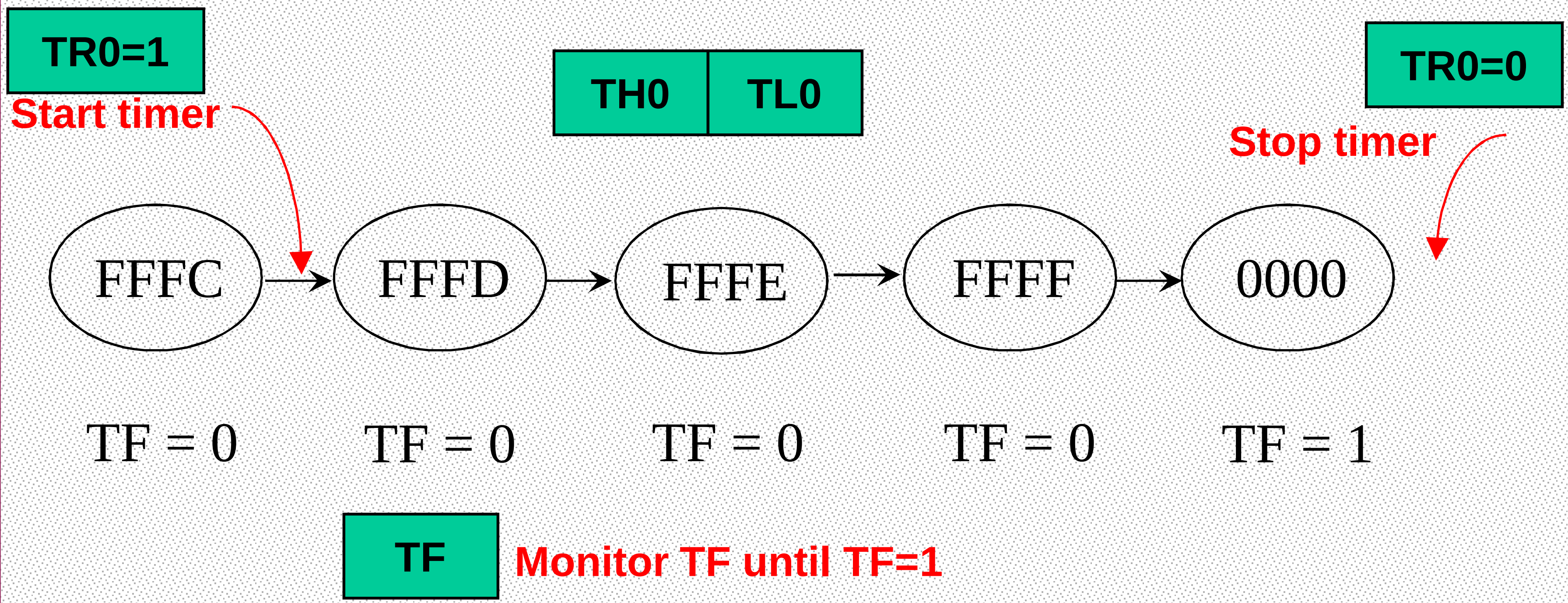
- **16-bit** timer (TH0 and TL0)
- TH0-TL0 is incremented continuously when TR0 is set to 1. And the 8051 stops to increment TH0-TL0 when TR0 is cleared.
- The timer works with the internal system clock. In other words, the timer counts up each machine cycle.
- When the timer (TH0-TL0) reaches its maximum of FFFFH, it rolls over to 0000, and TF0 is raised.
- Programmer should check TF0 and stop the timer 0.

Steps of Mode 1

1. Choose mode 1 timer 0
   - `MOV TMOD,#01H`
2. Set the original value to TH0 and TL0.
   - `MOV TH0,#FFH`
   - `MOV TL0,#FCH`
3. You had better to clear the flag to monitor: TF0=0.
   - `CLR TF0`
4. Start the timer.
   - `SETB TR0`

Steps of Mode 1

5. The 8051 starts to count up by incrementing the TH0-TL0.
   - ❖ **TH0-TL0= FFFCH,FFFDH,FFFEH,FFFFH,0000H**



Steps of Mode 1

6. When TH0-TL0 rolls over from FFFFH to 0000, the 8051 set TF0=1.

   TH0-TL0= FFFEH, FFFFH, 0000H (Now TF0=1)

7. Keep monitoring the timer flag (TF) to see if it is raised.

   AGAIN: JNB TF0, AGAIN

8. Clear TR0 to stop the process.

   CLR TR0

9. Clear the TF flag for the next round.

   CLR TF0

10. Repeat from step 2 for continuous time delay.

Timer Delay Calculation for XTAL = 11.0592 MHz

(a) in hex

- (FFFF – YYXX + 1) × **1.085 μs**
- where YYXX are TH, TL initial values respectively.
- Notice that values YYXX are in hex.

(b) in decimal

- Convert YYXX values of the TH, TL register to decimal to get a NNNNN decimal number
- then (65536 – NNNNN) × 1.085 μs

Timer Mode 2

- 8-bit timer.
  - It allows only values of 00 to FFH to be loaded into TH0.
- Auto-reloading
- TL0 is incremented continuously when TR0=1.

Steps of Mode 2

1. Chose mode 2 timer 0

```
MOV TMOD,#02H
```

2. Set the original value to TH0.

```
MOV TH0,#38H
```

3. Clear the flag to TF0=0.

```
CLR TF0
```

4. After TH0 is loaded with the 8-bit value, the 8051 gives a copy of it to TL0.

TL0=TH0=38H

5. Start the timer.

```
SETB TR0
```

Steps of Mode 2

6. The 8051 starts to count up by incrementing the TL0.
   - **TL0= 38H, 39H, 3AH,....**
7. When TL0 rolls over from FFH to 00, the 8051 set TF0=1. Also, TL0 is reloaded automatically with the value kept by the TH0.
   - **TL0= FEH, FFH, 00H (Now TF0=1)**
   - The 8051 auto reload **TL0=TH0=38H.**
   - **Clr TF0**
   - Go to Step 6 (i.e., TL0 is incrementing continuously).

- Note that **we must clear TF0** when TL0 rolls over. Thus, we can monitor TF0 in next process.
- Clear TR0 to stop the process.
  - **Clr TR0**

Timer Mode 0

- Mode 0 is exactly like mode 1 except that it is a **13-bit** timer instead of 16-bit.
  - 8-bit TH0
  - 5-bit TL0
- The counter can hold values between 0000 to 1FFF in TH0-TL0.
  - 2^{13}-1= 2000H-1=1FFFH
- We set the initial values TH0-TL0 to count up.
- When the timer reaches its maximum of 1FFFH, it rolls over to 0000, and TF0 is raised.

Example 01

- Find the frequency of a square wave generated on pin P1.0.

```
MOV   TMOD,#2H   ;Timer 0,mode 2
      MOV   TH0,#0
AGAIN:ACALL DELAY
      CPL   P1.0
      SJMP AGAIN
```

```
DELAY:SETB TR0          ;start
BACK: JNB   TF0,BACK    ;wait until TL0 ovrflw auto-reload
      CLR   TR0         ;stop
      CLR   TF0         ;clear TF
      RET
```

Solution:

T = 2 (256 × 1.085 μs) = 555.52 μs, and frequency = 1.8 KHz.

Example 02

- This program generates a square wave on pin P1.5 Using timer 1
- Find the frequency.(dont include the overhead of instruction delay)
- XTAL = 11.0592 MHz

```
       MOV  TMOD,#10H    ;timer 1, mode 1
AGAIN:MOV   TL1,#34H     ;timer value=3476H
       MOV  TH1,#76H
       SETB TR1          ;start
BACK:  JNB  TF1,BACK
       CLR  TR1          ;stop
       CPL  P1.5         ;next half clock
       CLR  TF1          ;clear timer flag 1
       SJMP AGAIN        ;reload timer1
```

Example 02

Solution:

FFFFH – 7634H + 1 = 89CCH = 35276 clock count

Half period = 35276 × 1.085 μs = 38.274 ms

Whole period = 2 × 38.274 ms = 76.548 ms

Frequency = 1/ 76.548 ms = 13.064 Hz.

Note

Mode 1 is not auto reload then the program must reload the TH1, TL1 register every timer overflow if we want to have a continuous wave.

Serial Communication

Types of Communication

- Simplex Communication
- Half Duplex Communication
- Full Duplex Communication

- Synchronous Communication
- Asynchronous Communication

- Parallel Communication
- Serial Communication

Start and stop bits

- When there is no transfer the signal is high
- Transmission begins with a start (low) bit
- LSB first
- Finally 1 stop bit (high)
- Data transfer rate (baud rate) is stated in bps

bps: bit per second

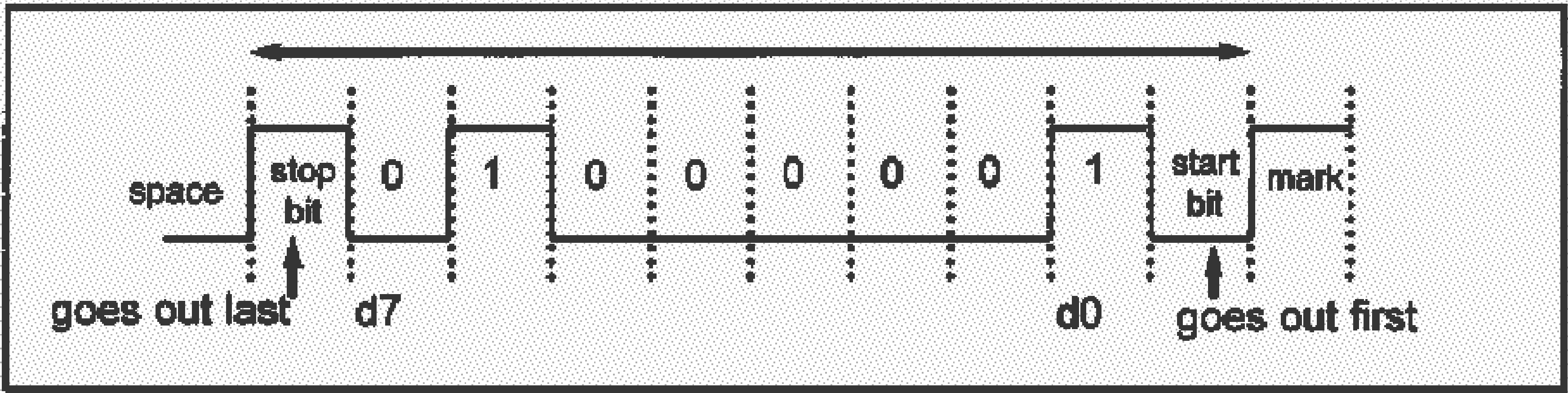


Figure 10-3. Framing ASCII "A" (41H)

How to communicate 8051 to PC

- Connect TXD to RXD and RXD to TXD from pc to 8051
- Use max232 to transform signal from TTL level to RS232 level
- The baud rate of the 8051 must matched the baud rate of the pc
- PC standard baud rate
  - 2400-4800-9600-14400-19200-28800-33600-57600
- Timer 1 mode 2 is used for bard rate generation

Example 10-1

With XTAL = 11.0592 MHz, find the TH1 value needed to have the following baud rates. (a) 9600 (b) 2400 (c) 1200

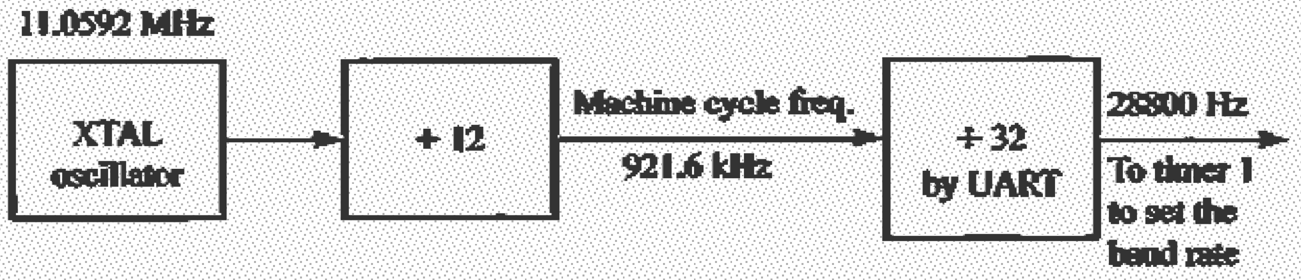
Solution:

With XTAL = 11.0592 MHz, we have:

The machine cycle frequency of the 8051 = 11.0592 MHz / 12 = 921.6 kHz, and 921.6 kHz/ 32 = 28,800 Hz is the frequency provided by UART to timer 1 to set baud rate.

(a) 28,800 / 3 = 9600 where –3 = FD (hex) is loaded into TH1
(b) 28,800 / 12 = 2400 where –12 = F4 (hex) is loaded into TH1
(c) 28,800 / 24 = 1200 where –24 = E8 (hex) is loaded into TH1

Notice that dividing 1/12th of the crystal frequency by 32 is the default value upon activation of the 8051 RESET pin. We can change this default setting. This is explained at the end of this chapter.



SBUF register

```
MOV    SBUF,#'D'    ;load SBUF=44H, ASCII for 'D'
MOV    SBUF,A       ;copy accumulator into SBUF
MOV    A,SBUF       ;copy SBUF into accumulator
```

Serial control (SCON) Register

SM0	SM1	SM2	REN	TB8	RB8	TI	RI

SM0 (SCON.7) : mode specifier

SM1 (SCON.6) : mode specifier

SM2 (SCON.5) : used for multi processor communication

REN (SCON.4) : receive enable

TB8 (SCON.3) : transmit bit (9th)

RB8 (SCON.2) : receive bit (9th)

TI (SCON.1) : transmit interrupt flag

RI (SCON.0) : receive interrupt flag

Mode of operation

SM0	SM1	MODE	operation	transmit rate
0	0	0	shift register	fixed
0	1	1	8 bit UART	variable (timer1)
1	0	2	9 bit UART	fixed
1	1	3	9 bit UART	variable (timer1)

What is SMOD

- Bit 7 of PCON register
- If SMOD=1 double baud rate
- PCON is not bit addressable
- How to set SMOD

```
Mov a, pcon
Setb acc.7
Mov pcon,a
```

MSB							LSB
SMOD	—	—	—	GF1	GF0	PD	IDL

Serial example(1)

An example of sending a message.

```
;initialization
        MOV TMOD,#20H
        MOV TH1,#-12
        MOV SCON,#52H
;begin to trnasmit
        SETB TR1
AGAIN1: MOV A,#‘A'
        CALL TRANSS
        MOV A,#‘E'
        CALL TRANSS
        MOV A,#‘I'
        CALL TRANSS
        MOV A,#'E'
        CALL TRANSS
        SJMP AGAIN1
;seial transmiting subroutine
TRANSS: MOV SBUF,A
AGAIN2: JNB TI,AGAIN2
        CLR TI
        RET
        END
```

Power control register

MSB							LSB
SMOD	—	—	—	GF1	GF0	PD	IDL

BIT	SYMBOL	FUNCTION
PCON.7	SMOD	Double Baud rate bit. When set to a 1 and Timer 1 is used to generate baud rate, and the Serial Port is used in modes 1, 2, or 3.
PCON.6	—	Reserved.
PCON.5	—	Reserved.
PCON.4	—	Reserved.
PCON.3	GF1	General-purpose flag bit.
PCON.2	GF0	General-purpose flag bit.
PCON.1	PD	Power-Down bit. Setting this bit activates power-down operation.
PCON.0	IDL	Idle mode bit. Setting this bit activate idle mode operation.

Idle mode

- An instruction that sets PCON.0 causes Idle mode
  - Last instruction executed before going into the Idle mode
  - the internal CPU clock is gated off
  - Interrupt, Timer, and Serial Port functions act normally.
  - All of registers , ports and internal RAM maintain their data during Idle
- RST signal clears the IDL bit directly

Power-Down Mode

- Last instruction executed before going into the power down mode
- the on-chip oscillator is stopped.
- all functions are stopped, the contents of the on-chip RAM and Special Function Registers are maintained.
- The reset that terminates Power Down

8051

Interrupts

Steps in executing an interrupt

- Finish current instruction and saves the PC on stack.
- Jumps to a fixed location in memory depend on type of interrupt
- Starts to execute the interrupt service routine until RETI (return from interrupt)
- Upon executing the RETI the microcontroller returns to the place where it was interrupted. Get pop PC from stack

Interrupt Sources

- 8051 has 6 sources of interrupts
  - Reset
  - Timer 0 overflow
  - Timer 1 overflow
  - External Interrupt 0
  - External Interrupt 1
  - Serial Port events (buffer full, buffer empty, etc)

Interrupt Vectors

Each interrupt has a specific place in code memory where program execution (interrupt service routine) begins.

External Interrupt 0:	0003h
Timer 0 overflow:	000Bh
External Interrupt 1:	0013h
Timer 1 overflow:	001Bh
Serial :	0023h
Timer 2 overflow(8052+)	002bh

Note: that there are only 8 memory locations between vectors.

Interrupt Enable (IE) register

- All interrupt are disabled after reset
- We can enable and disable them bye IE

D7 D0

EA	--	ET2	ES	ET1	EX1	ET0	EX0

EA	IE.7	Disables all interrupts.
--	IE.6	No implemented, reserved for future use
ET2	IE.5	Enables or disables timer 2 overflow interrupt
ES	IE.4	Enables or disables the serial port interrupt
ET1	IE.3	Enables or disables timer 2 overflow interrupt
EX1	IE.2	Enables or disables external interrupt 1
ET0	IE.1	Enables or disables timer 0 overflow interrupt
EX0	IE.0	Enables or disables external interrupt

Enabling and disabling an interrupt

- by bit operation
- Recommended in the middle of program

```
SETB  EA     SETB  IE.7     ;Enable All
SETB  ET0    SETB  IE.1     ;Enable Timer0 ovrf
SETB  ET1    SETB  IE.3     ;Enable Timer1 ovrf
SETB  EX0    SETB  IE.0     ;Enable INT0
SETB  EX1    SETB  IE.2     ;Enable INT1
SETB  ES     SETB  IE.4     ;Enable Serial port
```

- by mov instruction
- Recommended in the first of program

```
MOV IE, #10010110B
```

External interrupt type control

- By low nibble of Timer control register TCON
- IE0 (IE1): External interrupt 0(1) edge flag.
  - set by CPU when external interrupt edge (H-to-L) is detected.
  - Cleared by CPU when RETI executed.
- IT0 (IT1): interrupt 0 (1) type control bit.
  - Set/cleared by software
  - IT=1 edge trigger
  - IT=0 low-level trigger

(MSB)							(LSB)
TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0
Timer 1		Timer0		for Interrupt			

Interrupt Priorities

- What if two interrupt sources interrupt at the same time?
- The interrupt with the highest PRIORITY gets serviced first.
- All interrupts have a power on default priority order.
  1. External interrupt 0 (INT0)
  2. Timer interrupt0 (TF0)
  3. External interrupt 1 (INT1)
  4. Timer interrupt1 (TF1)
  5. Serial communication (RI+TI)
- Priority can also be set to “high” or “low” by IP reg.

Interrupt Priorities (IP) Register

---	---	PT2	PS	PT1	PX1	PT0	PX0

IP.7: reserved

IP.6: reserved

IP.5: timer 2 interrupt priority bit(8052 only)

IP.4: serial port interrupt priority bit

IP.3: timer 1 interrupt priority bit

IP.2: external interrupt 1 priority bit

IP.1: timer 0 interrupt priority bit

IP.0: external interrupt 0 priority bit

Interrupt Priorities Example

---	---	PT2	PS	PT1	PX1	PT0	PX0

- `MOV IP , #00000100B` **or** `SETB IP.2` **gives priority order**
  1. Int1
  2. Int0
  3. Timer0
  4. Timer1
  5. Serial
- `MOV IP , #00001100B` **gives priority order**
  1. Int1
  2. Timer1
  3. Int0
  4. Timer0
  5. Serial

Example

- A 10khz square wave with 50% duty cycle

```
        ORG    0              ;Reset entry point
        LJMP   MAIN           ;Jump above interrupt

        ORG    000BH          ;Timer 0 interrupt vector
T0ISR:CPL    P1.0             ;Toggle port bit
        RETI                  ;Return from ISR to Main program

        ORG 0030H             ;Main Program entry point
MAIN:   MOV    TMOD,#02H      ;Timer 0, mode 2
        MOV    TH0,#F0H       ;50 us delay
        SETB   TR0            ;Start timer
        MOV    IE,#82H        ;Enable timer 0 interrupt
HERE:   SJMP   HERE           ;Do nothing just wait
        END
```