MIcrocontroller Projects

All About Electronics

Home

8051

projects

Lcd Interfacing

PIC Microcontroller Projects

c++ projects

Arduino

Xilinx

Knowledge based

About Me

Contact Me

How to use internal timer registers of 8051(89c51,89c52) microcontroller

8051(89c51,89c52) series microcontrollers have two build in timers Timer-0 and Timer-1. You can use them as event counters, generating Time-delays, generating Baud Rates for serial communication or as an interrupt counter. Both the timers 0 and 1 are 16-bit wide. Since the timers are 16-bit wide the maximum range they can count is 2^16 (65535 times) after this value timers overflows. Timers actually counts the cycles that are generated by the crystal oscillator connected on pin 18(XTAL1) & 19(XTAL2). To understand the working of timer's you should first know about the registers that are associated with timers.

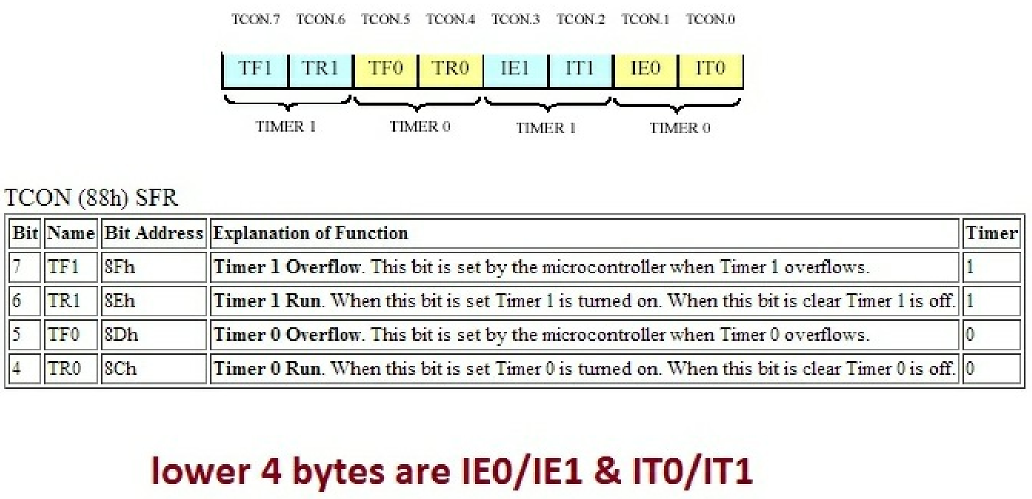
Timer Registers

- TCON (Timer Control register)
- TMOD (Timer Mode register)
- TH0/TL0 (Timer 0, 16-bit register, High bits goes to TH0, Low bits goes to TL0)
- TH1/TL1 (Timer 1, 16-bit register, High bits goes to TH1, Low bits goes to TL1)

TH0/TL0--------------TH1/TL1

These two registers TH and TL are timer high byte and timer low byte, 0 and 1 are the timers numbers. These are 16-bit registers. we Load our time delays in these registers. Since 8051(89c51,89c52) is an 8-bit microcontroller, we access These register in two bytes one byte for TH(timer high byte) and TL(timer low byte). TH and TL together makes 16-bits(TH 8-bit, TL8-bit). TH0 and TL0 are byte addressable only.

TCON(Timer control) Register



TCON (88h) SFR

Bit	Name	Bit Address	Explanation of Function	Timer
7	TF1	8Fh	**Timer 1 Overflow**. This bit is set by the microcontroller when Timer 1 overflows.	1
6	TR1	8Eh	**Timer 1 Run**. When this bit is set Timer 1 is turned on. When this bit is clear Timer 1 is off.	1
5	TF0	8Dh	**Timer 0 Overflow**. This bit is set by the microcontroller when Timer 0 overflows.	0
4	TR0	8Ch	**Timer 0 Run**. When this bit is set Timer 0 is turned on. When this bit is clear Timer 0 is off.	0

lower 4 bytes are IE0/IE1 & IT0/IT1

TCON (timer Control) register with its individual bits and sfr

88h is SFR of TCON register

- TR0/TR1: Timer 0/1 (Run control flag)
- TF0/TF1: Timer 0/1 (Timer overflow flag)

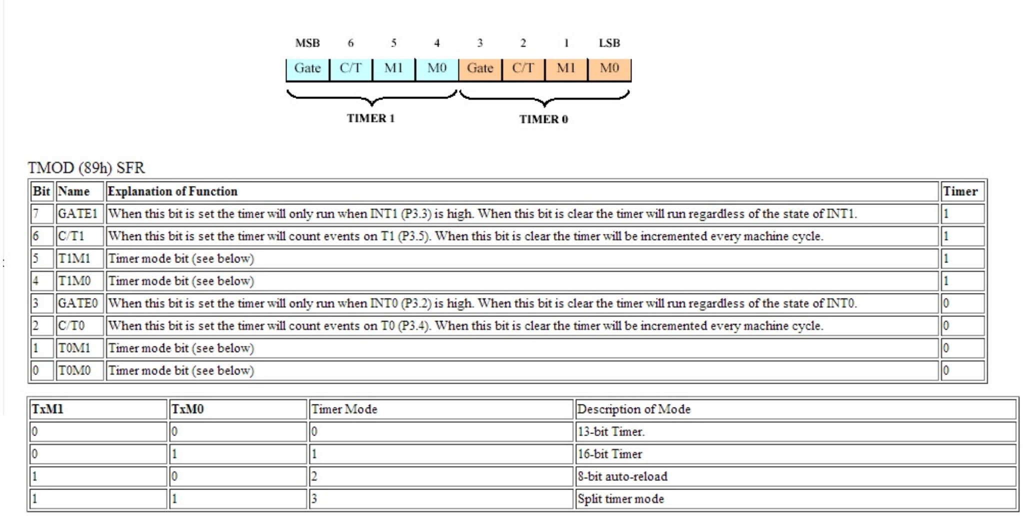
Lower Four bits are not shown in the above figure. These bits are

- IT0/IT1: Timer Interrupts. when IT0=1 or IT1=1 It specifies interrupt on falling edge and when IT0=0 or IT1=0 it specifies interrupt on rising edge.
- IE0/IE1: Used for external Interrupts.

Description

TCON is an 8-bit register. It's bits are used for generating interrupts internal or external. The most important bits of the timers, TRx and TFx are also in it. TRx(timer run) and TFx(timer overflow) bits which we use in almost all our timer applications are in it. When we initialize TRx with 1, TRx=1 it means that timer is running, When the specified time is over the timer it self make TFx=1 which means that the delay value is reached. Once TFx=1 stop the timer by initializing TRx with 0 TRx=0(Stop Timer). Know if we again want to run the timer make TRx=1. In the diagram you can see the SFR for TCON register, the bit's used for interrupt handling and the timer run and timer over flow bits. If you want to access the individual bits of the registers, you can access them by their names. You can also access whole register it self by its name. At the bottom of the page their is a small example explaining it.

TMOD(Timer mode)



TMOD (89h) SFR

Bit	Name	Explanation of Function	Timer
7	GATE1	When this bit is set the timer will only run when INT1 (P3.3) is high. When this bit is clear the timer will run regardless of the state of INT1.	1
6	C/T1	When this bit is set the timer will count events on T1 (P3.5). When this bit is clear the timer will be incremented every machine cycle.	1
5	T1M1	Timer mode bit (see below)	1
4	T1M0	Timer mode bit (see below)	1
3	GATE0	When this bit is set the timer will only run when INT0 (P3.2) is high. When this bit is clear the timer will run regardless of the state of INT0.	0
2	C/T0	When this bit is set the timer will count events on T0 (P3.4). When this bit is clear the timer will be incremented every machine cycle.	0
1	T0M1	Timer mode bit (see below)	0
0	T0M0	Timer mode bit (see below)	0

TxM1	TxM0	Timer Mode	Description of Mode
0	0	0	13-bit Timer.
0	1	1	16-bit Timer
1	0	2	8-bit auto-reload
1	1	3	Split timer mode

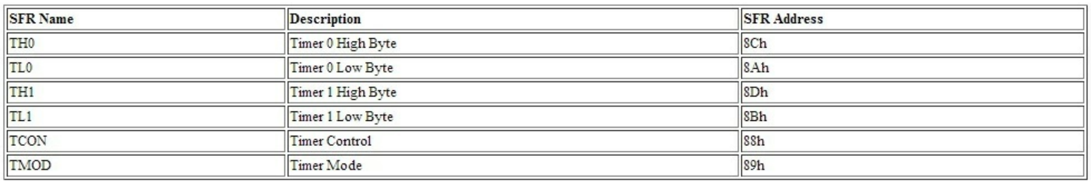
TMOD (Timer Mode) register bits and sfr.

- T1M0-T1M1----T0M1-T0M0: Selects the mode of the Timer.
- C/T: Selects Timer as External Counter or Internal Timer. If C/T is 1 Timer as a Counter is selected if 0 Timer as a time delay generator is selected.
- Gate0-Gate1: When Gate is 1, Timer runs only when INT0-INT1 Pin of 89c51 is high.

Description

TMOD is an eight bit register, it is bit addressable. Higher four bits (4 to 7) are related to Timer 1 where as the lower four bits (0 to 3) perform the exact same functions, but for timer 0. You can see their functions in the figure above.

SFR's for timer registers



SFR Name	Description	SFR Address
TH0	Timer 0 High Byte	8Ch
TL0	Timer 0 Low Byte	8Ah
TH1	Timer 1 High Byte	8Dh
TL1	Timer 1 Low Byte	8Bh
TCON	Timer Control	88h
TMOD	Timer Mode	89h

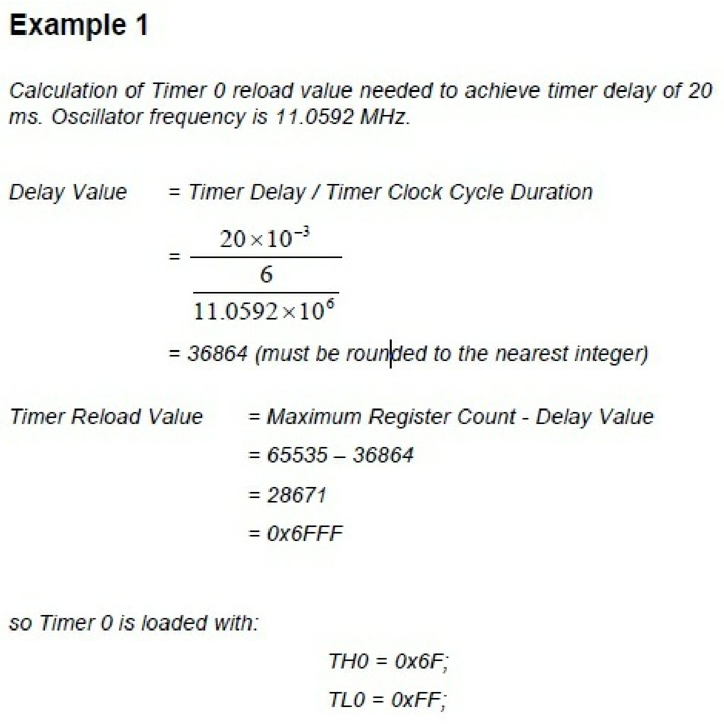
8051 Timers sfrs

Note we can access the registers by their names as well which make the programming easy and i am going to use them by their names and not sfr's.

How to calculate values for TH and TL registers?

Timer Clock Cycle Duration = 6/oscillator frequency

Timer clock cycle duration is the time in which one cycle is provided by the oscillator, and in a second thousand 's of cycle's are provided by the oscillator. Divide the delay needed by the Timer clock cycle duration. For example I need a delay of 20ms, Divide 20ms by Timer Clock Cycle duration. Now negate the result of previous calculation from 65535, which is the maximum count by the timer with out over flow. Now transform the obtained decimal value in hexadecimal, and load the two high bytes in TH and low bytes in TL.



Example 1

Calculation of Timer 0 reload value needed to achieve timer delay of 20 ms. Oscillator frequency is 11.0592 MHz.

Delay Value = Timer Delay / Timer Clock Cycle Duration

$$= \frac{20\times10^{-3}}{\dfrac{6}{11.0592\times10^{6}}}$$

= 36864 (must be rounded to the nearest integer)

Timer Reload Value = Maximum Register Count - Delay Value

= 65535 – 36864

= 28671

= 0x6FFF

so Timer 0 is loaded with:

TH0 = 0x6F;

TL0 = 0xFF;

Calculating Delay for 8051(89c51,89c52) microcontroller

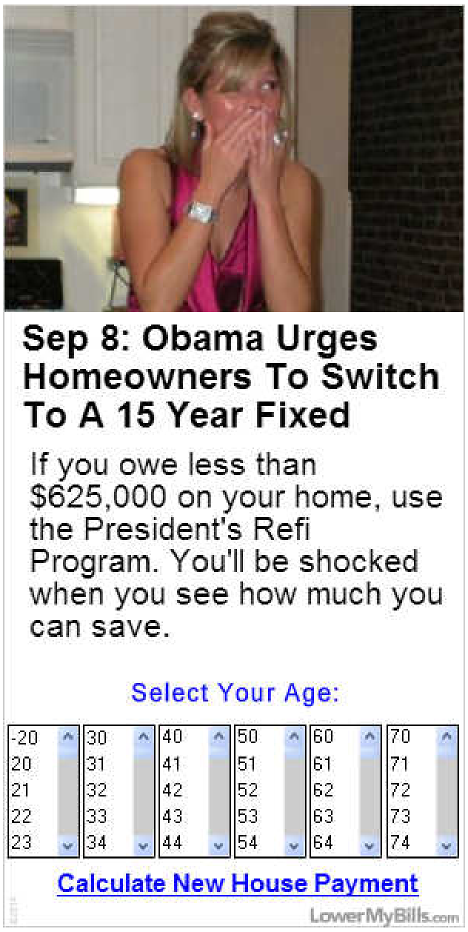
8051 Microcontroller Delay generation Steps

1. Select Timer Mode in TMOD Register. Specify which Timer you are using .
2. Load values for TLx and THx registers.
3. Run the timer by initilizing TRx with 1 TR=1.
4. Keep monitoring the timer flag (TFx). Tf it is raised Stop the timer by making TRx=0;.
5. Clear the TFx flag for the next count.

Simple Delay function

```
void Delay(void)
{
TMOD = 0x01;      //Timer 0 is in use. 16-bit Timer Mode is selected.
TL0 = 0x6F;       //Load value for TLx register
TH0 = 0xFF;       //Load value for THx register
TR0 = 1;          //Run Timer-0
while(!TF0)       //Poll TFx
TR0 = 0;          //If TF=1 stop the timer by making TR=0
TF0 = 0;          //Make TF=0 for next counting
}
```

Some Projects Created using same Delay method shown above



Digital Clock using 8051(89c51,89c52) Microcontroller.

Generating One Second Delay using Timers of 8051(89c51,89c52) microcontroller.

Generating One Minute Delay using Timers of 8051(89c51,89c52) microcontroller.

Download PDF Document for 8051(89c51,89c52) Microcontroller Timers Registers