

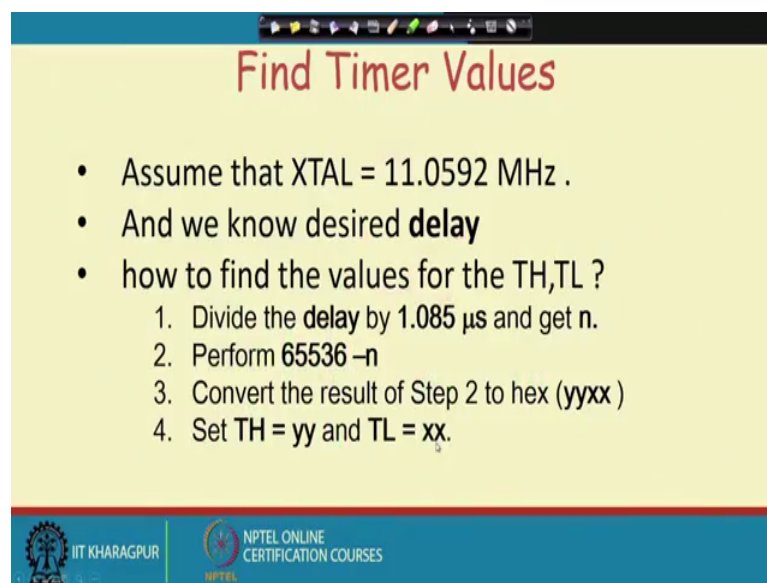
Microprocessors and Microcontrollers
Prof. Santanu Chattopadhyay
Department of E & EC Engineering
Indian Institute of Technology, Kharagpur

Lecture - 33
8051 Microcontroller (Contd.)

The next important thing that we have is to find out the what value should be loaded into the timer, the type of examples that we have seen so far, they are if we load this timer with some value. So, we can we can calculate what is the delay that is produced, but normally we need it in the other way that is for a particular application.



So, that amount of delay needed is known and for that we have to initialize the timer accordingly. So, what we need to do is just to reverse the calculation that we have done previously. So, suppose we have got this crystal frequency of 11.0592 megahertz and we know the desired delay, but what value should be loaded into TH and TL register.

(Refer Slide Time: 00:55)



Find Timer Values

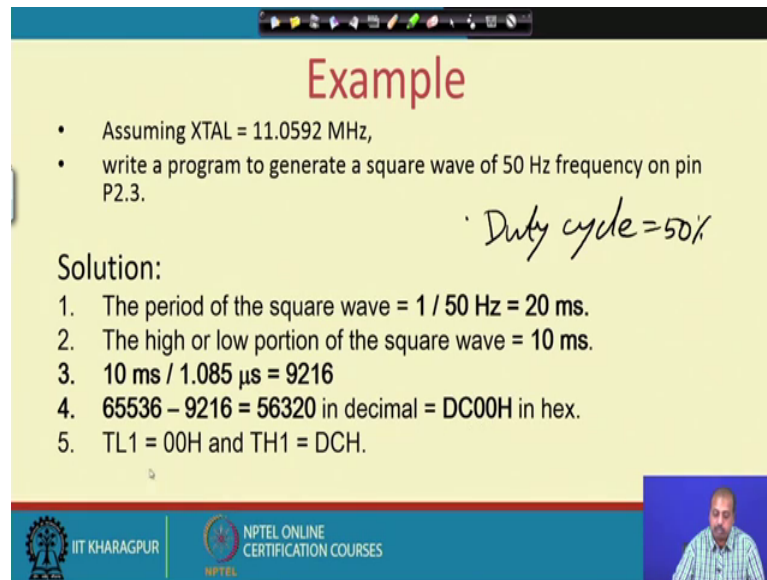
- Assume that XTAL = 11.0592 MHz .
- And we know desired **delay**
- how to find the values for the TH,TL ?
 1. Divide the **delay** by 1.085 μ s and get n.
 2. Perform 65536 -n
 3. Convert the result of Step 2 to hex (yyxx)
 4. Set TH = yy and TL = xx.

 IIT KHARAGPUR |  NPTEL ONLINE CERTIFICATION COURSES

So, first of all; so, 11.0592 so, you know that it will be divided by 12. So, this delay will be this value divided by 12. So, that will give us that with be the clock frequency, and as a result the delay of 1 clock is 1 clock period is 1.085 microsecond. So, this value 1.085 microsecond if you use it to divide the delay that you need. So, you will get the value n after you have got the value n. So, you just subtract 65536 minus n. So, you subtract this n from 65536, and we have to definitely you need to convert to hex. So, that way we get

this then this it will be converted to YYXX. So, suppose this is these are these are the higher ordered, it is the higher ordered byte this XX is the lower ordered byte accordingly, the TH should be loaded with YY and TL should be loaded with XX.

(Refer Slide Time: 02:02).



Example

- Assuming XTAL = 11.0592 MHz,
- write a program to generate a square wave of 50 Hz frequency on pin P2.3.

Duty cycle = 50%

Solution:

1. The period of the square wave = $1 / 50 \text{ Hz} = 20 \text{ ms}$.
2. The high or low portion of the square wave = 10 ms.
3. $10 \text{ ms} / 1.085 \mu\text{s} = 9216$
4. $65536 - 9216 = 56320$ in decimal = DC00H in hex.
5. TL1 = 00H and TH1 = DCH.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, this way we can calculate it suppose we are trying to write a program that will generate a square wave of frequency 50 hertz. So, we want to generate a square wave of frequency 50 hertz on pin number P 2.3. So, pin number 2.3 and pin 2.3; so, those things are secondary now first of all we have to use the timer.

So, how this timer should be initialized. So, period of the square wave is 1 upon 50 hertz; so, that is 20 millisecond. So, both this high time and low time they are in 10 milliseconds. So, here the basic assumption is that the duty cycle is 50 percent; so, this entire assignment. So, we assume that this duty cycle is 50 percent. So, if duty cycle is not 50 percent, then also you can do some small modification to the program and get it done.

So, with that. So, if we just look into this program. So, this high time and low time they are same since it is 50 percent. So, it is 10 millisecond now for generating 10 millisecond. So, this 10-millisecond divided by 1.085 microsecond that gives us 9216. So, this 9216 minus. So, 65536 minus 9216 is 56320, because these if we initialize the timer with this value, then from this point onwards. So, it will be starting to count

upwards till FFFF and then it will overflow. So, it is calculated in this fashion. So, 56320 in hexadecimal. So, it is DC 0 0.

So, TL 1 should be loaded with this 0 0 value and TH 1 should be loaded with this DC x, value DC DCH value.

(Refer Slide Time: 03:54).

Example

```

MOV TMOD,#10H ;timer 1, mode 1
AGAIN: MOV TL1,#00 ;Timer value = DC00H
MOV TH1,#0DCH
SETB TR1 ;start
BACK: JNB TF1,BACK ;stop
CLR TR1 ;stop
CPL P2.3
CLR TF1 ;clear timer flag 1
SJMP AGAIN ;reload timer since
;mode 1 is not
;auto-reload
    
```

Timing Diagram: 80% duty cycle, period 1, pulse width 1/50.

Handwritten notes: $\frac{0.8}{50}$, $\frac{0.2}{50}$, SETB P2.3, CLR B P2.3, (M)

So, the program will be something like this, first we are using this timer 1 mode 1. So, for that the TMOD register is a 1 0 then this TL register is loaded with 0 0 TH registered is loaded with 0 DC and then we start the timer. So, SETB TR 1; so, it will start the timer and this P per 2.3 this port b 2 point 3 is having some value it maybe low maybe high. So, it does not matter. So, whatever be the value. So, it will be outputting that one only. So, after some time when this timer will expire. So, these instructions JNB TF 1 back. So, it is continually sensing this timer overflow and when that overflow will occur this TF 1 bit will be equal to 1.

So, in that case this loop this JNB check will be false. So, it will come to the next statement where it will clear the timer 1. So, it will clear timer 1 and then it will complement P 2.3. So, it will be complimenting the bit whatever was the output on that pin 2.3. So, it will be complemented so, if it was previously 1 now it will become 0, or if it was 0 previously it will become 1, then it is clearing the TF 1 and then it is. So, that flag is cleared and then SJMP again. So, it will jump back to this point, where it will be again loading the timer with the appropriate value, and accordingly it will work.

So, this program so, this generates this 50 percent duty cycle clock on this of frequency 50 hertz on the line P 2.3. So, if you need some other duty cycle then of course, it is slightly complex because then so, suppose we want say a duty cycle of say 80 percent that is, 80 percent time the clock should be high, 20 percent time it should be low then again 80 percent time it should be high.

So, if this is the situation then from here to here. So, this is that that 50-hertz frequency will be maintained. So, this is 1 by 50 second. So, that part is fine, but now you have to divide it into point 8 into 1.8 by 50 for to calculate the delay of this whole region, and point 2 by 50 to calculate the delay of this region.

So, that way you can. So, this delay this timers are to be loaded separately, first of all you cannot leave this P 2.3 to be added to have any arbitrary value at the beginning. So, if would first maybe we had set bit P 2.3 to say that this P 2.3 is high, and then you use the timer to produce the delay corresponding to that point 8 by 50. So, accordingly you have to start that you have to load that you have to calculate this delay value, to be loaded into TH 1 TL 1 pair and then you have to do that.

So, the then that delay will be produced after that when that after this instruction is executed after this overflow has occurred, then you have to clear that clear bit P 2.3, now the bit is low now that pole bit has become low and now you have to put a delay here for point 2 by 50, point 2 by 50 and again you should have that JNB check etcetera similar such similar type of check will be here, and then at the end you should go back to this point. So, that type of modification will be required. So, I think you can do it using some other duty cycle, instead of 50 percent using some other duty cycle comes.




(Refer Slide Time: 08:02)

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.

$$\begin{array}{c} \overbrace{1\text{FFF}}^{\leftarrow 4} \\ \leftarrow 8 \end{array}$$

 - $2^{13}-1=2000\text{H}-1=1\text{FFFH}$
- 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.

Now regarding the next mode that we will discuss; so, this is the mode 0. So, this is a restricted version of this timer mode 1. So, this is otherwise it is same as mode 0 mode 1, but only thing is that it is a 13-bit value. So, this 8 bit will be held in TH 0 register and 5 bits are held in the TL 0 register. So, counter can hold values between 0 0 0 0 to 1 FFF in TH 0, TL 1. \

So, since it is 13 bit; so, this FF this is sorry this is. So, since this is a 13 b 13 bit. So, these will constitute 8 bit, 8 plus this is another 4-bit 12 bit and this is 1. So, the total number is 13-bit number. So, you can have in 13 bit the maximum number that you can represent is 1 FFF.

So, when the count value reaches 1 FFF, after that it will overflow when it wants to go beyond that then the timer will overflow, and the overflow there is TF bit will be set. So, we set this initial. So, otherwise the operation is same. So, this that is 2 power 13 minus 1. So, that is 1 FFFH. So, otherwise the operation is same. So, we set the initial values TH 0 TL 0 to count up, and when the timer reaches the value 1 FFFH, it will rollover to 0 0 0 0 and TF 0 will be raised. So, otherwise this operation is similar.

(Refer Slide Time: 09:42)

The slide is titled "Timer Mode 2" in red text. It contains three bullet points: "8-bit timer.", "Auto-reloading", and "TL0 is incremented continuously when TR0=1.". Below the text, there is a small video inset of a man speaking. At the bottom, there are logos for IIT Kharagpur and NPTEL Online Certification Courses.

Then we look into mode 2. So, this is interesting mode because this has got some auto reload facility. So, first of all it is an 8-bit timer. So, your time delay that you can produce is small, because the delay values can be loaded is 00 to FF and it has to be loaded in the TH register only, TH TH 0 register only.

So, then auto reload; so, this will be; so, what. So, auto reload means whenever the timer will expire timer will overflow, then again, the value will be loaded from TH register to the TL register. So, TL 0 will be incremented continually and when TL 0 will overflow, then this whenever this TR 0 is 1, then TL 0 will be incremented and when it will be overflowing then this TF bit will be set, and also the value will be reloaded to this TL register.

(Refer Slide Time: 10:46)

The slide, titled "Steps of Mode 2", lists five steps for configuring timer 0 in mode 2. Step 1: "Chose mode 2 timer 0" with assembly code `MOV TMOD, #02H`. A handwritten diagram shows the TMOD register bits: 0000 for the first four bits and 0010 for the last two bits, with arrows pointing to the 10 in 0010. Step 2: "Set the original value to TH0." with assembly code `MOV TH0, #38H`. Step 3: "Clear the flag to TF0=0." with assembly code `CLR TF0`. Step 4: "After TH0 is loaded with the 8-bit value, the 8051 gives a copy of it to TL0" with the handwritten note `TL0=TH0=38H`. Step 5: "Start the timer." with assembly code `SETB TR0`. The slide footer includes the IIT KHARAGPUR logo and NPTEL ONLINE CERTIFICATION COURSES text.

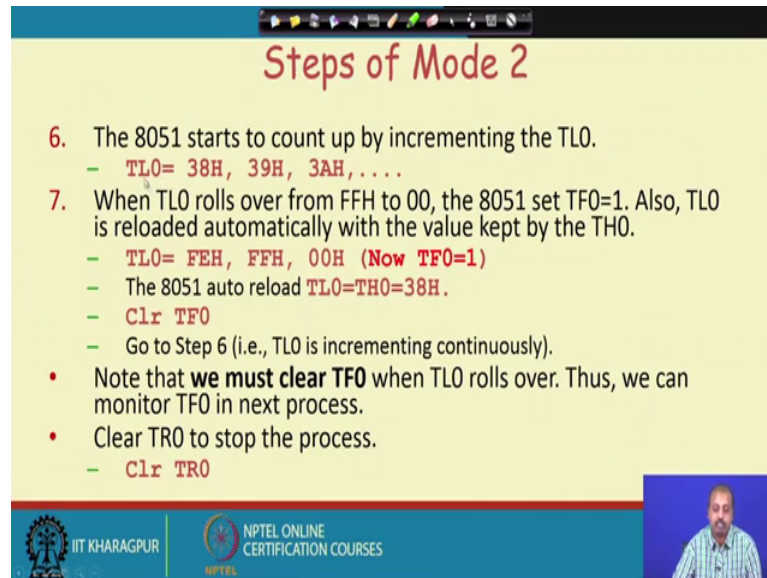
So, this is the operation first we have to choose mode 2. So, for timer 0; so, we are explaining with respect to timer 0. So, TMOD is 02. So, this TMOD register value is like this now. So, it is 00000010. So, this part is for timer 1.

So, this is the gate this is the C by t. So, it will resume your timer and these 2 bits will identify that we are going for mode 2, then we have to set the initial value in TH0. So, the original value. So, that is original value will be loaded into TH0. So, that is done by this instruction `mov TH0, #38h`, suppose we are loading this initial value as 38h, we have to clear the carry flag.

So, TF0 is cleared then this after TH0 is loaded with eight bit 8051 gives a copy of it to TL0 since it is mode 2. So, this is done implicitly this TH0 TL0 getting the value of TH0 get 38h. So, that is done automatically. So now, it will start the timer. So, you have to use this `SETB TR0` and it will be. So, it will be starting the time.

So, before once once we have set this timer mode to mode 2, and then then this TH0 registered is loaded with 38h. So, this TL0 will be automatically loaded with 308h. So, next you set this TR0 bit. So, the timer will start. So, this timer will start with this mode, with and this the upcounting will be like this.

(Refer Slide Time: 12:29)



Steps of Mode 2

- The 8051 starts to count up by incrementing the TL0.
 - TL0= 38H, 39H, 3AH,
- When TL0 rolls over from FFH to 00H, the 8051 sets TF0=1. Also, TL0 is reloaded automatically with the value kept by the TH0.
 - TL0= FEH, FFH, 00H (Now TF0=1)
 - The 8051 auto reloads 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 the next process.
- Clear TR0 to stop the process.
 - Clr TR0

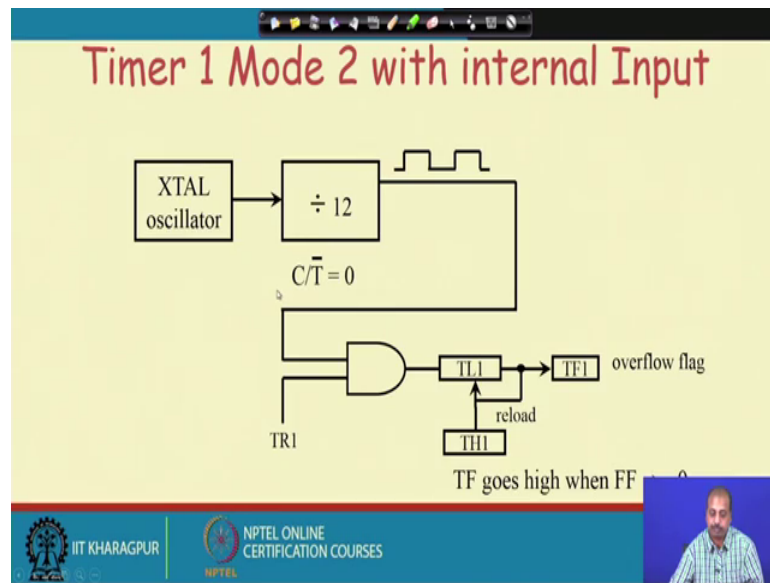
IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

The TL 0 is 38 39 3A 3B. So, it will go like this then after some time. So, it will go to FF and from FF it will roll off to 00. So, when it rolls like that from FF to 00 then the TF 0 bit will be set to 1, and then this TL 0 will be reloaded automatically with the value in the TH 0 register. So, TH 0 content is not lost. So, TH 0 was holding 38 h in our case. So, it will continue to hold 38 h. So, that value is not affected by this upcounting process.

So, this when this TL 0 overflows, then this TL 0 will be loaded again with the TH 0. So now, when the TH 0 will become there is an overflow. So, this flag becomes equal to 1, and 8051 it goes into auto reload mode and this TL 0 and TH 0 they are loaded with 30 8 h. So, then we are clearing this TF 0. So, this; so now, this now it will again once you clear this TF 0. So, it will since TR 0 bit is already set. So, TR bit we have not reset. So, it will be again start this upcounting from that point. So, this is the point to note that we must clear TF 0 when TL 0 rolls over. So, that is important and then we. So, that we can know monitor TF 0 in the next process.

So, it is not automatic. So, it is not that as soon as it rolls off. So, TF 0 will be also be cleared. So, that does not happen. So, we have to do it manually, and then whenever we want to stop the process we have to set clear clear the TR 0 bit. So, that is as usual. So, clearing the TR 0 bit.

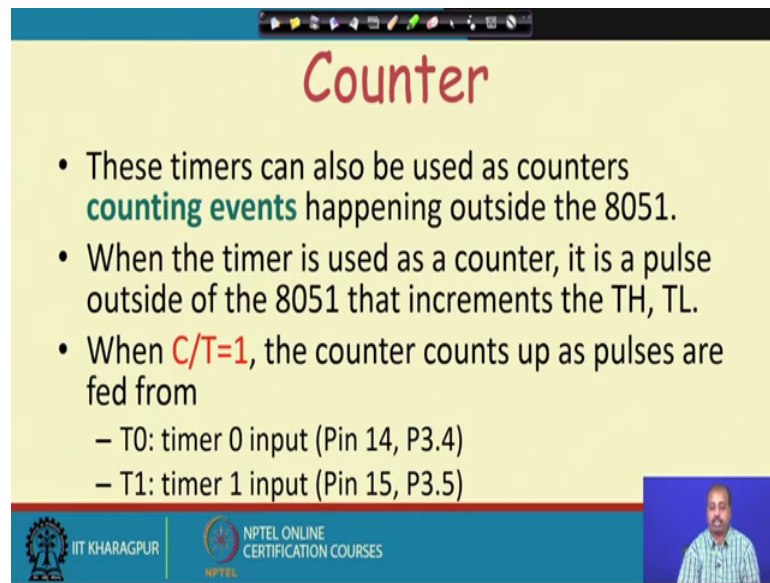
(Refer Slide Time: 14:24)



So, this is the operation see if. So, this crystal oscillator is there. So, it is divided by 12 to generate the clock frequency and then that is logically anded with that TR 1 bit. So, TR 1; so, this is with for timer 1. So, it is TL logically anded with TR 1. So, whenever TR 1 is high, then this clock will be reaching the timer then TL 1 is the timer that we have. So, TL 1 it will be starting to upcount and when that overflow occurs, then this TF 1 bit will be set and this reload will also be given to TH 1. So, TH or this reload instruction will come and accordingly TH 1, 1 value will be reloaded to TL 1 value.

So, TF will go high whenever this there is a rollover from FF to 0 0 and then this TF 1 is high and this reload is also done. So, that it will be, but we have to clear this TF 1 bit otherwise it may create some this maybe mistakenly take considered as another overflow of the timer, next we look into the counter operation.

(Refer Slide Time: 15:35)



The slide is titled "Counter" in a large, red, serif font. Below the title, there are three bullet points. The first bullet point states that these timers can be used as counters for counting events outside the 8051. The second bullet point explains that when used as a counter, a pulse from outside the 8051 increments the TH and TL registers. The third bullet point states that when the C/T bit is set to 1, the counter counts up as pulses are fed from two specific pins: T0 (timer 0 input, Pin 14, P3.4) and T1 (timer 1 input, Pin 15, P3.5). The slide has a yellow background and a blue footer containing the IIT Kharagpur and NPTEL logos.

- These timers can also be used as counters **counting events** happening outside the 8051.
- When the timer is used as a counter, it is a pulse outside of the 8051 that increments the TH, TL.
- When **C/T=1**, the counter counts up as pulses are fed from
 - T0: timer 0 input (Pin 14, P3.4)
 - T1: timer 1 input (Pin 15, P3.5)

So, this timer operation we have seen where we are getting the clock from the microcontroller crystal, and it is divided by 12 and that is used as the clock.

So, we can use this module this timer counter module also as a counter. So, how are you going to use it. So, these timers can be used for counting events that occur outside 8051, when the timer is used as a counter. So, it is a pulse outside the 8051 that will increment the TH TL registers. So, here this C by T bit should be set to 1. So, that now it is the count operation and we have for 8051, there are 2 pins 1 is called T 0 another is called T 1. So, they are actually pin number 14 and 15 of 8051, and they are multiplexed with the port bit 3s bit number port 3s bit number 4 and bit number 5.


So, if you are using this 8051, for this counter operation if you are trying to use this counter facility then we cannot use port 3 for simple input output operation anyway. So, this T 0 and T 1. So, they are the 2 timers that we can use. So, this pins can be used for feeding the timer 0, and timer 1 and operating them in the counter mode.

(Refer Slide Time: 17:09)

Port 3 Pins Used For Timers 0 and 1

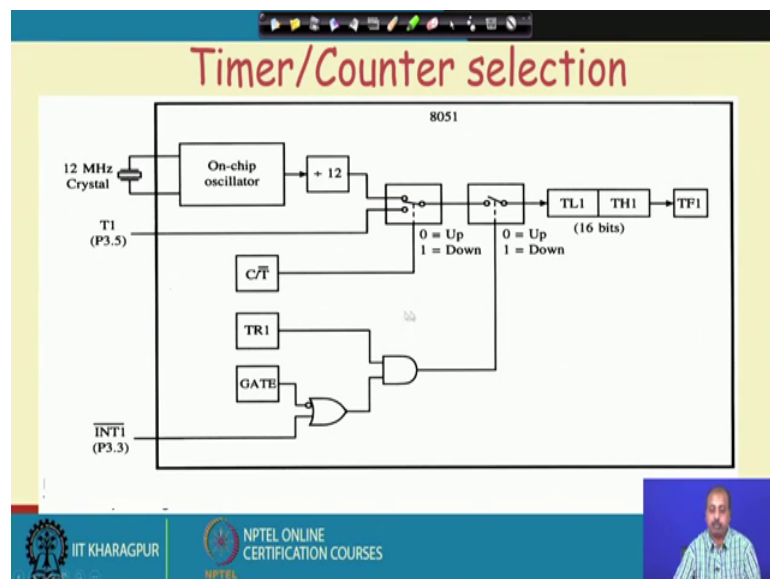
Pin	Port Pin	Function	Description
14	P3.4	T0	Timer/Counter 0 external input
15	P3.5	T1	Timer/Counter 1 external input

(MSB)	(LSB)						
GATE	CT=1	M1	M0	GATE	CT=1	M1	M0
Timer 1				Timer 0			



So, this 14 15 support pin is P 3.4 function is T 0, and timer counter 0 external input, and 15 is port pin P 3.5 function is T 1 and it is timer counter 1 external input. So, this structure we have already seen now the only difference that we have is this, C by T bit it should be set to 1 for using this timer as a counter and this this C by T bit should be set to 1 for using this timer 0 as a counter.

(Refer Slide Time: 17:43)



So, this is the whole schematic diagram. So, we have got this whenever you are going this going for the timer operation. So, this C by T bit is 0. So, when this C by T bit is 0,

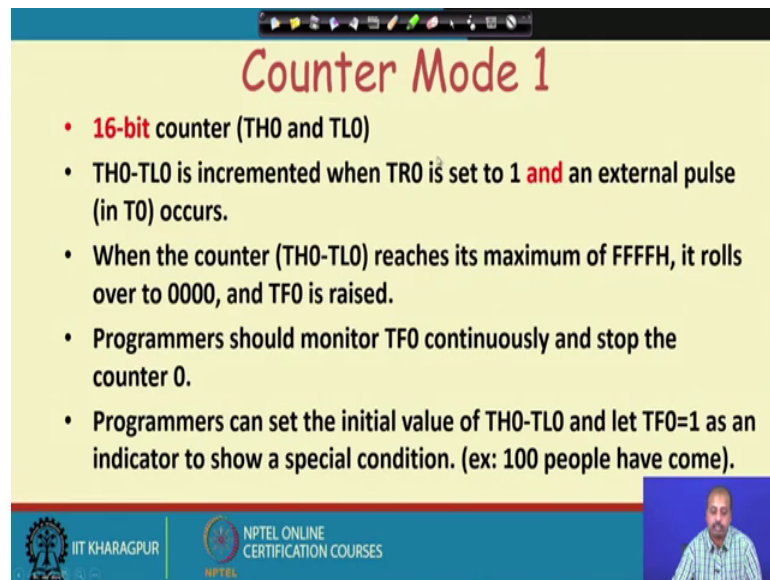
as if this switch is connected to the upward contact point as a result from the crystal oscillator whatever clock is coming. So, that is going to control the timer and of course, there is. So, and when this C by T is equal to 1, then this this switch this contact is established with the lower point you can say, and as if in that case the clock is coming from the pin T 1.

So, this is one part distinguishing that counter and timer operation using C by T bit, second important thing that we have is the control of the gate and the TR bits, now if this gate is 0, then it does not depend on the interrupt whereas, if the gate is 1; that means, at this point you get is 0. So, it will be dependent on the INT pin now if the INT pin is 0, this this INT bar.

So, when; so, if it is 1 then you will get a 1 here and then this logical, you can say that if this this is 1 and this TR 1 is also 1 then only. So, this will be connected down. So, it will be operating. So, it will be. So, this point will be controlling this and if it is 0, in that case it will be controlled by it will be controlled by this line. So, that way we have got this time with this gate and TR bits controlling the as if it is controlling the second switch.

So, this is the totally logical diagram. So, it in reality in inside it might not be incremented like this, but this is a logical diagram for understanding, and ultimately whatever whether it comes from this internal clock frequency, or from this T 1 pint it is dependent on gate and all those things ultimately the clock reaches this timer. So, it maybe is the events that that is incremented as pulses or it may be the crystal frequency. So, that was this timer get it, and then it will start doing up counting and when it overflows it will go to this this overflow bit will be set. So, that is the whole operation of this timer counter selection.

(Refer Slide Time: 20:24)



Counter Mode 1

- **16-bit** counter (TH0 and TL0)
- TH0-TL0 is incremented when TR0 is set to 1 **and** an external pulse (in T0) occurs.
- When the counter (TH0-TL0) reaches its maximum of FFFFH, it rolls over to 0000, and TF0 is raised.
- Programmers should monitor TF0 continuously and stop the counter 0.
- Programmers can set the initial value of TH0-TL0 and let TF0=1 as an indicator to show a special condition. (ex: 100 people have come).

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

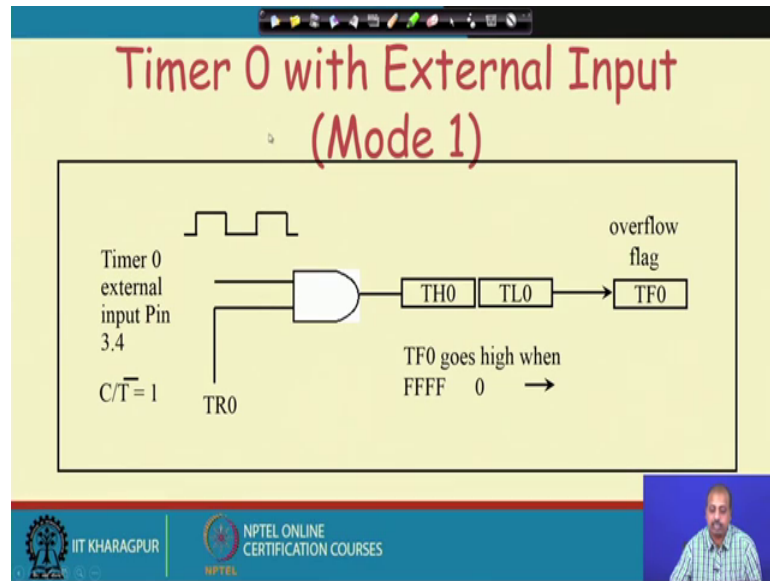
for counter mode 1 just like timer mode 1 we have got counter mode 1. So, it is 16 bit counter TH 0 and TL 0 will be used. So, TH 0 and TL 0 they will be incremented when TR 0 is set to 1, and an external pulse occurs. So, whenever this counter is enabled by means of this this TR bit and whenever this TR bit is 1 then only it will count the external events, and whenever if you do not want to count the external events then you can just in a in your program, you can set this to the corresponding TR bit to 0.

So, it will not count the external event, on the other hand if you when this count value will reach it is maximum, since we are using mode 1. So, it is a 16-bit operation. So, when it reaches the maximum FFFFH it will rollover to 0 0 0 0 0 and in that case the TF 0 will be raised, that as if 1 bunch of counting has been over. So, this timer or the counter is overflowing.

So, we can say that now the software that we are developing, it should take a note that it has already counted 65536 items, program should the programmer should monitor continually and stop the counter 0 now, and then you can you can set this initial value of the this TH 0 TL 0 as you want, but from that point onward it will start counting, and let TF 0 equal to 1 as an indicator to show the special condition, like say if we say that. So, if we want to count the number of people that have arrived in a room, and if we put a bar that after every 100 people that have come, we need to be notified that 100 people have entered.

So, what we have to do is we have to start the counter with the value FFFF minus 100 that way, and then we have to continue now when the. So, the counter will be upcounting from that value and when it crosses FFFF, then it will generate an overflow in TF 0 bit. So, we get an indication that an 100 people have arrived. So, this way this counters can be utilized for doing the operation.

(Refer Slide Time: 12:49)





So, this is timer 0 with external operation mode 1 operation. So, this is the C by T pin is C by T bar pin is set to 1. So, that way it is a counter operation. So, TR 0; so, this is, so, this time this external pin P 3.4. So, it is logically ended here. So, it goes to TH 0 TL 0 pair. So, otherwise it is same. So, when it goes from FFFF to 0 it will overflow and this TF 0 will be getting the value.

(Refer Slide Time: 23:24).

Counter Mode 2

- 8-bit **counter**.
 - It allows only values of 00 to FFH to be loaded into TH0.
- Auto-reloading
- TL0 is incremented if TR0=1 **and** external pulse occurs.





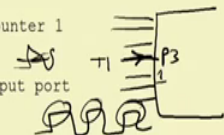
Next we will look into this mode 2 operation of counter. So, it is an mode 2 just like mode 2 timer operation. So, mode 2 counter operation is also an 8-bit operation, and it allows only values 00 to ff to be loaded into TH0. So, it is a 8-bit value everything is same it is auto reload TL0 will be incremented if TR0 equal to 1, and this external pulse occurs. So, then only this TL0 value will be incremented. So, that way it will so, when there will be overflow again the value will be loaded from th0 to TL0 and it will continue like that.

(Refer Slide Time: 24:05)

Example

Assuming that clock pulses are fed into pin T1, write a program for counter 1 in mode 2 to count the pulses and display the state of the TL1 count on P2.

```
MOV TMOD,#01100000B ;mode 2, counter 1
MOV TH1,#0
SETB P3.5 ;make T1 input port
AGAIN: SETB TR1 ;start
BACK: MOV A,TL1
MOV P2,A ;display in P2
JNB TF1,Back ;overflow
CLR TR1 ;stop
CLR TF1 ;make TF=0
SJMP AGAIN ;keep doing it
```



So, this is an example so, we assume that the clock pulses are fed into pin T 1, and we want to write a program for counter 1 in mode 2 to count the pulses and display the state of the T 1 of the TL TL 1 count or the this port P 2 on port 2.

So, the first thing is that we want to have this mode 2 counter 1. So, counter 1 has to be set to mode 2. So, the first instruction. So, this is actually doing that. So, if we divide it. So, this is for 0 counter timer 0. So, this gate bit is 0. So, this is a counter operation. So, C by T is 1, and then this is mode 2. So, this is the mode 2 setting next, we will initialize this TH 1 to 0 because we just want to count how many pulses have come. So, that way we initialize this is this to 0 and then SETB P 3.5. So, so SETB P 3.5; so, this will make this T 1-bit input.

So, this is another very interesting thing that I should tell you. So, you remember that in case of 8051 whenever you have got a port. So, this P 3; so, this bits are there now this bits if you are trying to read the value, first of all you have to write a 1, whenever you are trying to read a 8051 port. So, you have to first write a 1 there. So, this is because of that structure of that port consisting of the transistor and that cell of that and that port latch. So, this we have seen long back. So, basically; so, that has to be done.

So, this instruction is very, very important. So, if you miss this, then it may be that previously this this port was having a value 0, and it continues to have the value 0. So, it will not be able to read the port even if some value is coming on that line. So, this SETB P 3.5 is very important. So, this will make this T 1 configured as input port, because this T 1 will be coming as input to the microcontroller now.

So, after that we set bit TR 1. So, start the timer. So, when you start the timer. So, this as soon as you have loaded TH 1. So, this value is also loaded in the TL register TL 1 register, and then. So now, we can continually see. So, pulses will be coming on this line so, and now the pulses will be coming on this T 1 pin sorry this say it is coming like this on the TL 1 pin and on each such pulse is counted as a 1.

So, that is done automatically by the processor. So, as a programmer what we do we move the content of TL 1 registered to a, and then move the content of a register to P 2. So, that way continually we can display the number of pulses that have arrived on T 1 pin onto the port P 2, then we check whether there is an overflow, because it will count up to ff only.

So, in that case; so, if there is a jump on not bit TF 1 backs as long as there is no overflow. So, it gets the next value and updates and once it overflow. So, you want to stop it. So, you want this clear TR 1. So, we stop this timer. So, we stop make the overflow bit 0 and SJMP again. So, we start the process again. So, that way we can continually go on displaying the numbers number of pulses coming from 0 0 to FF, and then again 0 0 to FF. So, it can go on continually by this program.

So, this is useful for mode 2 operation in the counter.