Embedded System Interfacing

Lecture 9
Timers

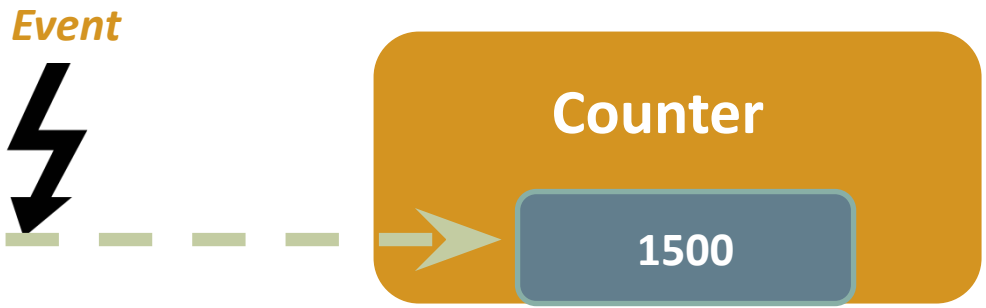
This material is developed by IMTSchool for educational use only
All copyrights are reserved

IMT SCHOOL

Introduction

What is a Counter ?

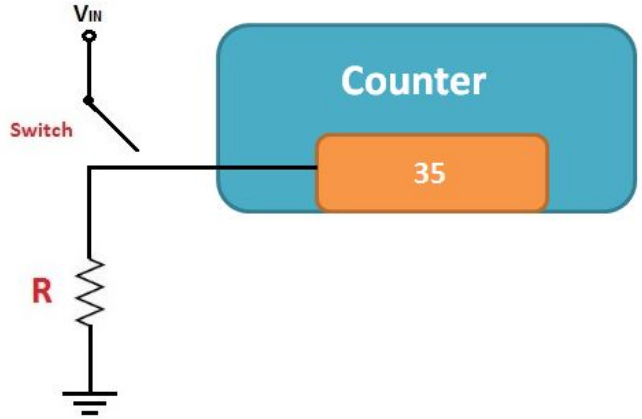
Simply counter is a peripheral that counts events in a specific register. These events are electrical signals like rising edge or falling edge. If these events are periodic (Comes every defined period) then this counter is counting time, hence it is called Timer.



IMT SCHOOL

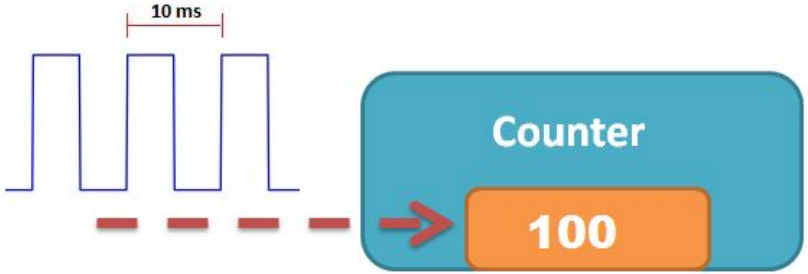
Example

Think of a peripheral that counts a switch presses, The Counter is connected to one terminal of the switch which gives a ***rising edge*** every press. The counter increments its register value by 1 count every press. Then if we read the counter register and found 35, it means that there are 35 switch presses have been done.



Example

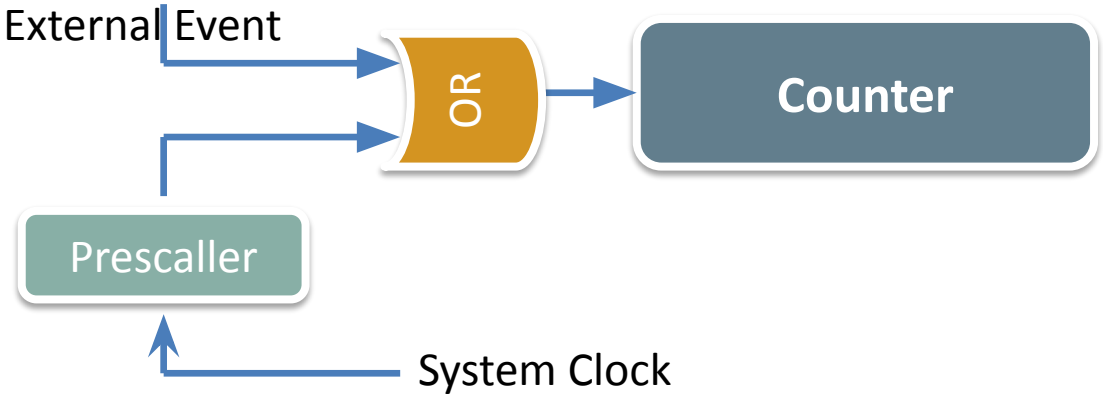
On the other hand, if the counter is connected to ***a periodic signal*** with a periodic time equals to 10 ms. The counter is configured to count every ***rising edge*** as the previous example. If we read the counter register and found 100, it means that 100 rising edge are happened. Because the signal is periodic and the it gives rising edge every 10 ms, then ***we can conclude that 1000 ms have been passed*** from the time we started the counter. This is the timer mode.



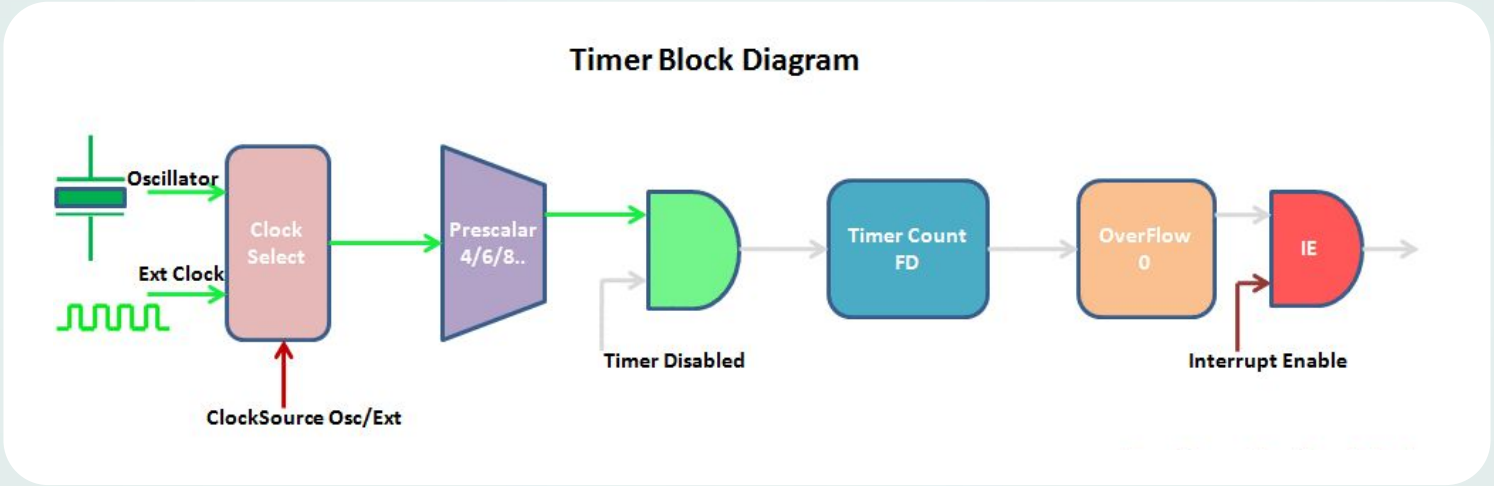
Example

From previous examples, we can say that a ***timer is a normal counter*** but counting a periodic signal. So that we can calculate the time from the value in the timer register.

In summary, the timer and the counter are the same peripheral, depending on the input we could classify if it is working as timer or as counter.



Example


Timer Block Diagram
Oscillator
Ext Clock
Clock Select
ClockSource Osc/Ext
Prescalar 4/6/8..
Timer Disabled
Timer Count FD
OverFlow 0
IE
Interrupt Enable

So..?

Timers are used everywhere. Without timers, you would end up nowhere! The range of timers vary from a few microseconds (like the ticks of a processor) to many hours (like the lecture classes)

counters are hardware mechanisms for counting some form of event like the clock pulses.

Timers If we know the time of one count ,so it will be easy to calculate the hole time by multiply the No of counts by the count time .

IMT SCHOOL

Timer Terminology

Resolution: Number of bits represents the timer register.

Timer Tick Time: The time between to consecutive events, i.e. time needed to increment the timer register by 1

Timer Count: Number of counts needed by the timer to count from 0 till it reaches 0 again. It shall be equals to

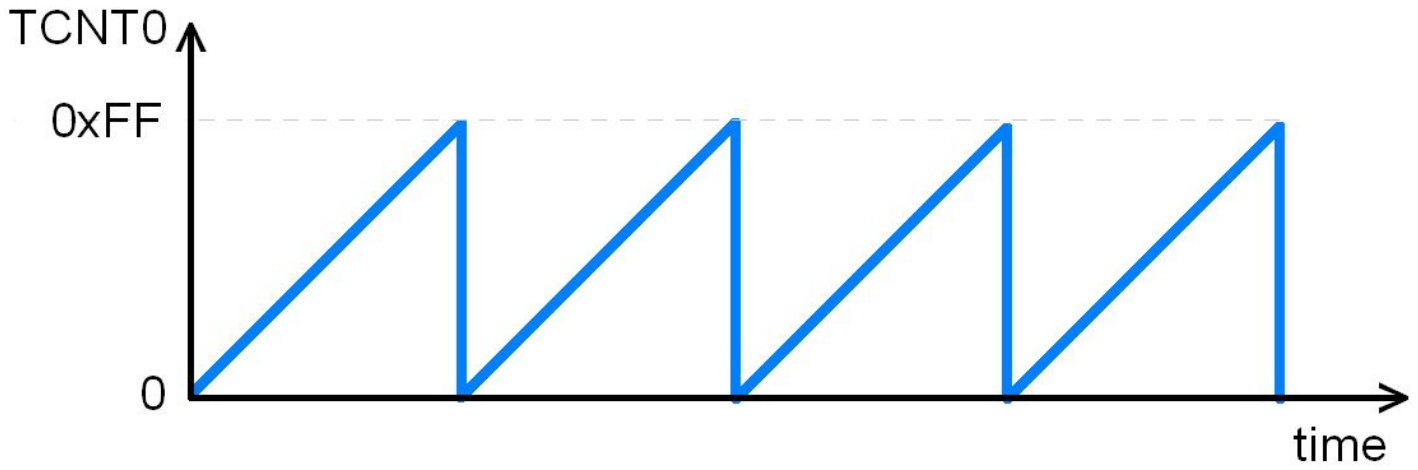
Timer Overflow: The time needed by the timer to count from 0 until it reaches 0 again.
It shall be equals to **Timer Count x Timer Tick Time**

Prescaler: Prescaler is a division factor for the system clock (Processor Clock) before it is applied to the timer.

iMT SCHOOL

Timer Overflow

Overflow timer is fire an interrupt (flag)
whenever the timer register overflows .



Timer Overflow

Overflow timer is fire an interrupt (flag)
whenever the timer register overflows .

In case of 16 bit timer register , the overflow flag
will be set when the value reaches 16535

Timer Calculations

- **Timer Clock =** $\frac{\text{System Clock}}{\text{Prescaller}}$
- **Timer Tick Time =** $\frac{1}{\text{Timer Clock}} = \frac{\text{Prescaller}}{\text{System Clock}}$

Example

Assume a microcontroller working on a frequency of 4 MHz, calculate the timer tick time assuming the timer is using a prescaller of 4.

Solution

Timer Tick Time = $\frac{4}{4 \times 10^{6}}$ = 1 microsecond.

i.e. this timer would be incremented every 1 micro second.

IMT SCHOOL

Timer Calculations

Timer Overflow Timer = Timer Count x Timer Tick Time

$$= 2^{\text{Resolution}} \times \frac{\text{Prescaller}}{\text{System Clock}}$$

Example

From the previous example, assuming the timer resolution is 8 bit, calculate the overflow time.

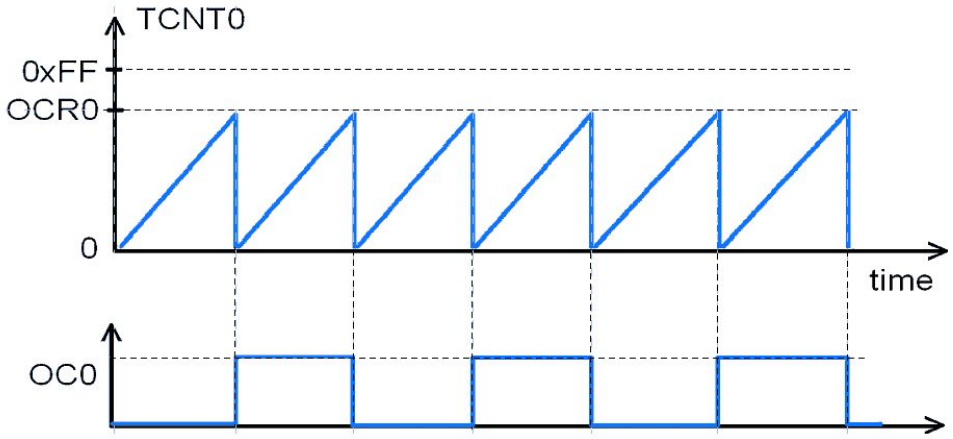
Solution

Overflow time = 1 microsecond x 2^8 = 256 microseconds.

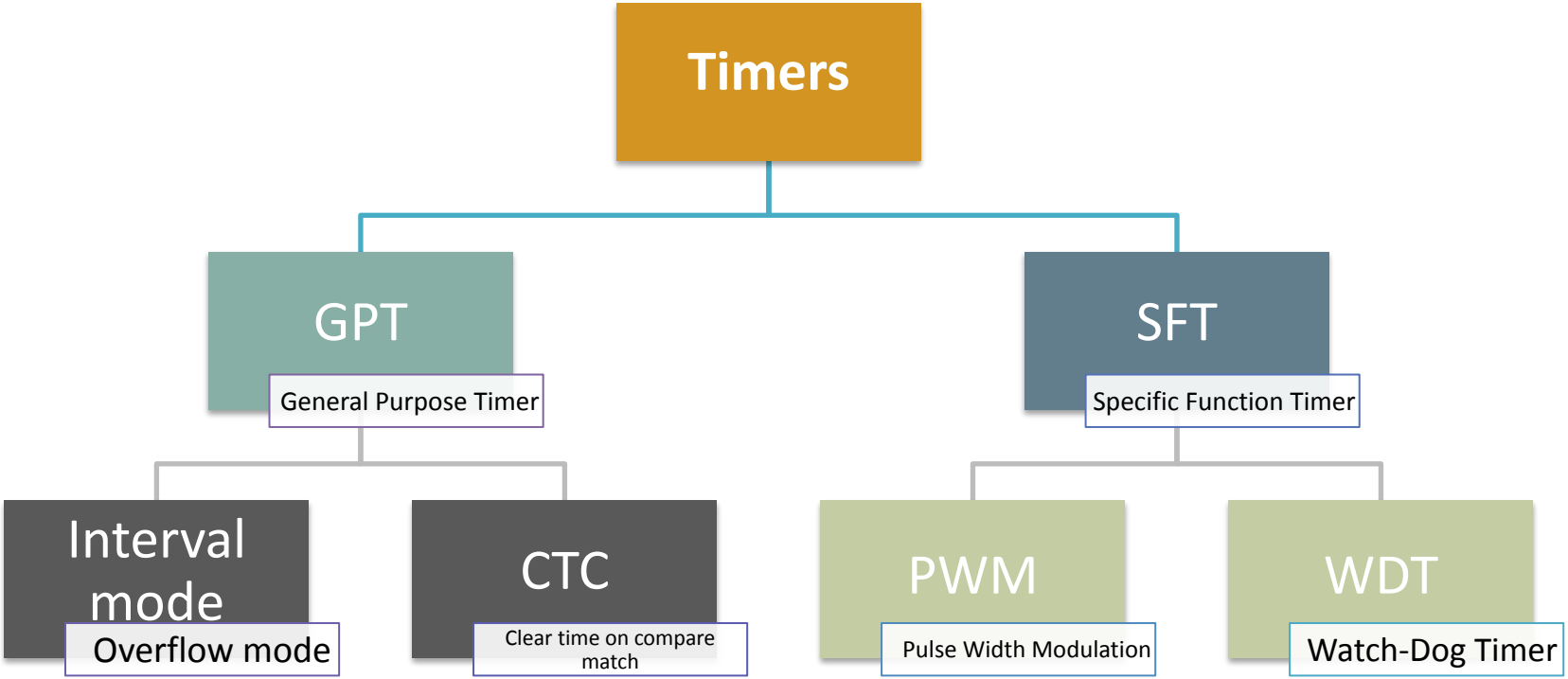
IMT SCHOOL

Timer CTC

- Compare Match Interrupt is fired by a timer whenever the value of the timer becomes equal to a certain predefined value.
- This predefined value is stored in a register known as the Output Compare Register.



Timers Types



IMT SCHOOL

IMT SCHOOL

Interval Timer

02

Normal Mode

Interval timer is a normal functioning timer that counts until it overflows. On overflow, the timer sets an overflow flag and fires an interrupt if enabled.

How to Calculate a certain time based on an interval timer … ?

Step 1 Calculate the overflow time of the used timer.

Step 2 Compare the desired time with the overflow time, and follow one of the following possibilities.

IMT SCHOOL

Possibility 1

Desired Time equals to Overflow time

This is the happy scenario possibility, which normal doesn't happen. If you are lucky and the desired time is equals to overflow time, then just enable the overflow interrupt and take the desired action inside the overflow ISR.

```
ISR (OverFlow)
{
    /* Take Desired Action */
}
```

IMT SCHOOL

Possibility 2

Desired Time less than Overflow time

Step 1 Calculate the number of counts needed

$$\text{Number of counts needed} = \frac{\text{Desired Time}}{\text{Overflow Time}} \times \text{Overflow Count}$$

Step 2 Preload the timer register with the following Value:

Preload Value = Overflow Count – Number of counts needed

Step 3 Enable the timer overflow interrupt and take the desired action

See the following example to understand more …

IMT SCHOOL

Example

Assuming 8 bit timer using a prescaller of 4. The system frequency is 4 MHz and an action is needed to be taken after 64 microsecond. Do the needed calculations and write the code.

Overflow time = 256 microseconds (calculated in previous slides)

The desired time is 64 microseconds which is less than the overflow time. Then we calculate the needed number of counts:

Number of needed Counts = (64 / 256) x 256 = 64 count.

Preload value = 256 – 64 = 192.

If the timer register is initialized with the preload value (192), then it needs only 64 count to overflow, i.e. the overflow interrupt would fire after only 64 count. Which is the desired time.

Example

At Initialization

```
Timer_Register = 192;
```

Then

```
ISR (OverFlow)
{
    /* Initialize the timer register again to */
    /* keep the timer counting the same value */
    Timer_Register = 192;

    /* Take the desired action */
}
```

IMT SCHOOL

Possibility 3 case 1

Desired Time more than Overflow time

Step 1 Calculate the number of overflows needed

$$\text{Number of overflows} = \frac{\text{Desired Time}}{\text{Overflow Time}}$$

Assume that the number of the overflows needed is a *decimal value*, then define a variable and increments it inside the overflow interrupt till we reach the desired number of overflows, then take the desired action.

See the following example to understand more …

IMT SCHOOL

Example

Assuming 8 bit timer using a prescaller of 4. The system frequency is 4 MHz and an action is needed to be taken after 2.56 milliseconds. Do the needed calculations and write the code.

Overflow time = 256 microseconds (calculated in previous slides)

The desired time is 2.56 milliseconds which is more than the overflow time. Then we calculate the needed number of overflows:

Number of overflows = (2560 / 256) = 10 overflows.

Example

At Initialization

```
u8 Overflows_Count = 0;
```

Then

```
ISR (OverFlow)
{
    Overflows_Count++;

    if (Overflows_Count == 10)
    {
        /* Re-initialize the variable to 0 again */
        /* to count the 2560 microseconds again    */
        Overflows_Count = 0;

        /* Take the desired action */
    }
}
```

Possibility 3 case 2

Desired Time more than Overflow time

Step 1 Calculate the number of overflows needed

$$\text{Number of overflows} = \frac{\text{Desired Time}}{\text{Overflow Time}}$$

Assume that the number of the overflows has a floating value, then calculate the preload value for the floating part first, then calculate the number of the overflows for the decimal value.

See the following example to understand more …

IMT SCHOOL

Example

Assuming 8 bit timer using a prescaller of 4. The system frequency is 4 MHz and an action is needed to be taken after 2.62 milliseconds. Do the needed calculations and write the code.

Overflow time = 256 microseconds (calculated in previous slides)

The desired time is 2620 microseconds which is more than the overflow time.
Then we calculate the needed number of overflows:

Number of overflows = (2620/ 256) = 10.25 overflows.

First Calculate the preload value for the floating value,
Preload Value = 256 – (0.25 x 256) = 192 counts.

Second Calculate the number of overflows needed for the decimal value,
Number of overflows = 10 / 1 = 10 overflows.

IMT SCHOOL

Example

At Initialization

```
u8 Overflows_Count = 0   ;
Timer_Register     = 192 ;
```

Then

```
ISR (OverFlow)
{
    Overflows_Count++;

    if (Overflows_Count == 11)
    {
        /* Re-initialize the variable to 0 again */
        /* And the timer register to 64          */
        /* to count the 10.5 microseconds again  */
        Overflows_Count = 0;
        Timer_Register     = 192


        /* Take the desired action */
    }
}
```

Can you understand why it is 11 not 10 … ?

IMT SCHOOL

AVR Timers

Avr Timers

In ATMEGA32, we have three different kinds of timers:

TIMER0

- 8-bit timer
- CTC mode
- Phase Correct PWM
- 10-bit Clock Prescaler
- Overflow and Compare Match Interrupt Sources

TIMER1

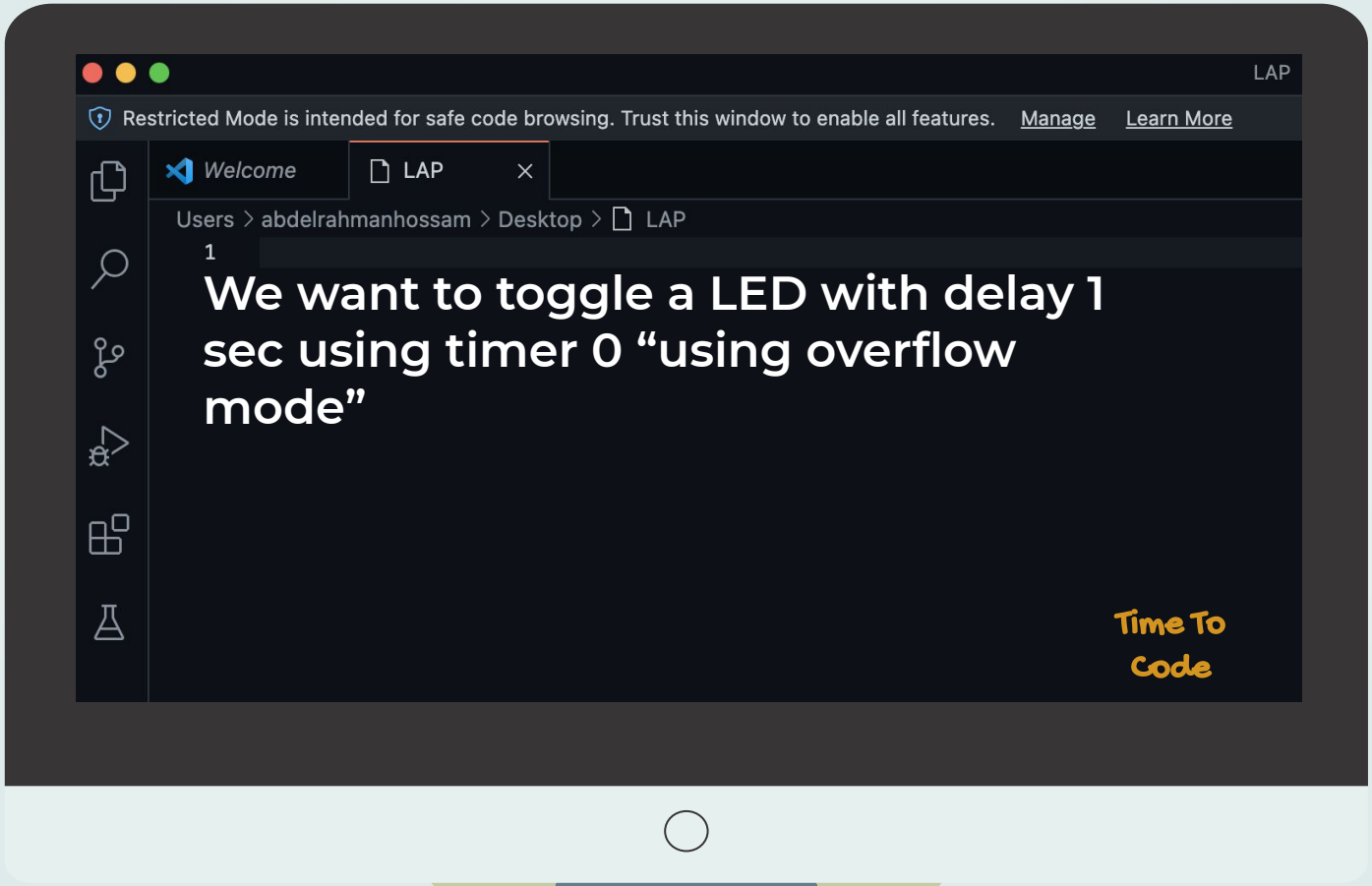
- 16-bit timer
- Two Independent Output Compare Units
- One Input Capture Unit
- Phase Correct (PWM)
- Variable PWM Period
- Four Independent Interrupt Sources

TIMER2

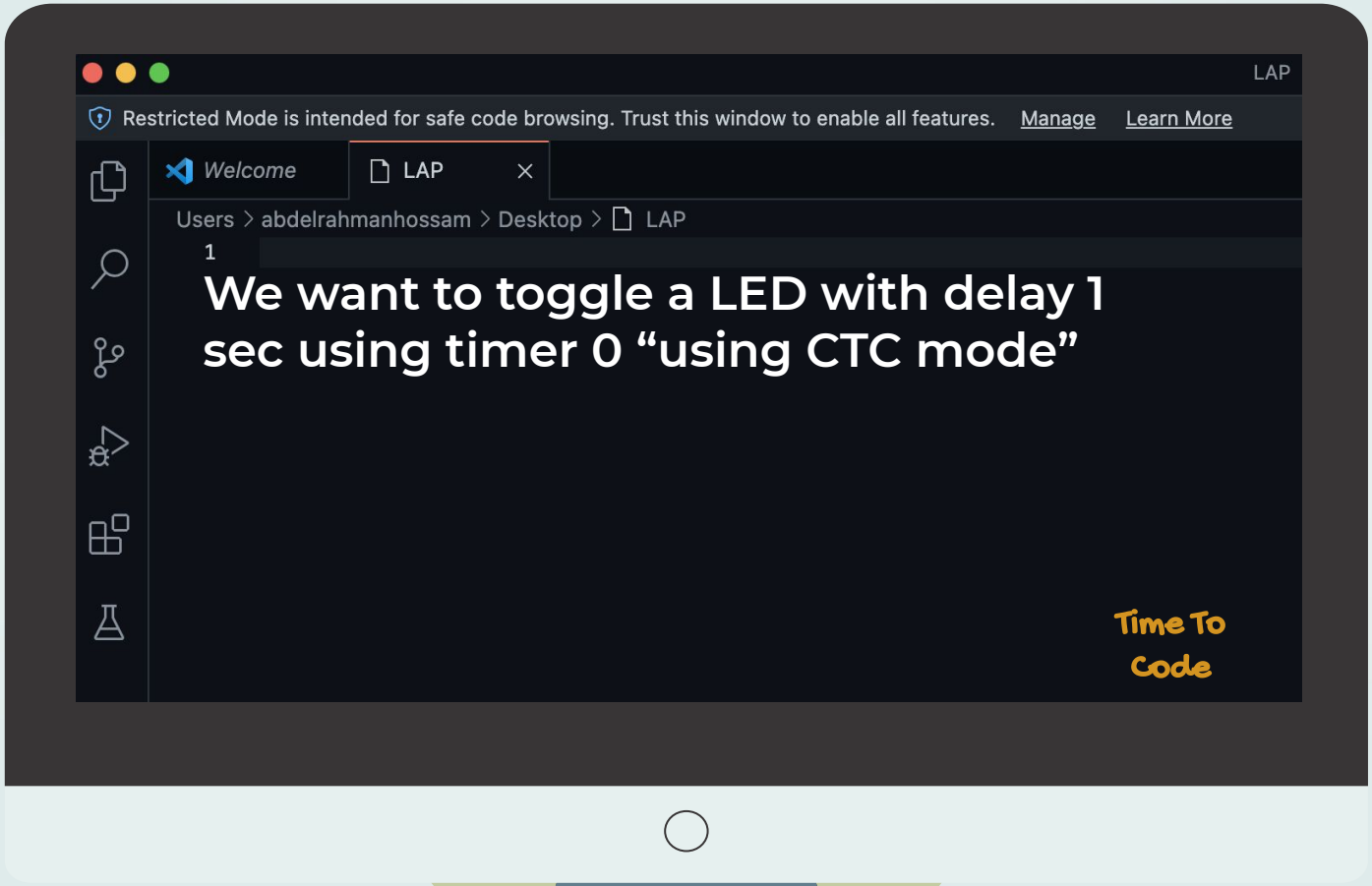
- 8-bit timer
- CTC mode
- Phase Correct PWM
- 10-bit Clock Prescaler
- Overflow and Compare Match Interrupt Sources
- Allows clocking from External 32 kHz

IMT SCHOOL

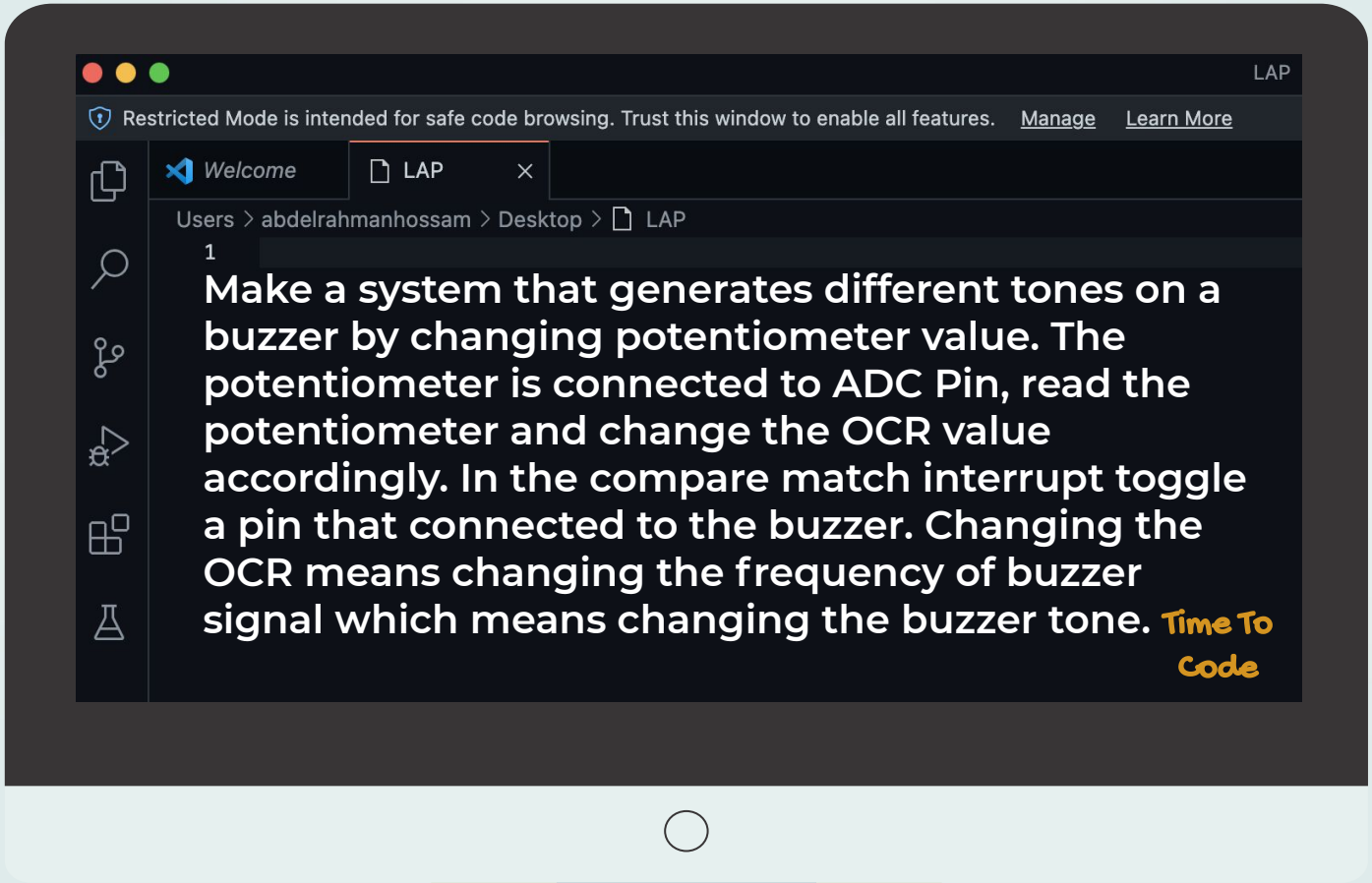
LAb 1


LAP
Restricted Mode is intended for safe code browsing. Trust this window to enable all features.
Manage
Learn More
Welcome
LAP
Users > abdelrahmanhossam > Desktop > LAP
1
We want to toggle a LED with delay 1 sec using timer 0 “using overflow mode”
Time To Code

LAb 2


LAP
Restricted Mode is intended for safe code browsing. Trust this window to enable all features.
Manage
Learn More
Welcome
LAP
Users > abdelrahmanhossam > Desktop > LAP
1
We want to toggle a LED with delay 1 sec using timer 0 “using CTC mode”
Time To Code

LAb 3



IMT SCHOOL

The End

IMT SCHOOL

www.imtschool.com

www.facebook.com/imaketechnologyschool/

This material is developed by IMTSchool for educational use only

All copyrights are reserved