Process:

First of all I needed to remember everything related to timers,

So I first watched timer videos of this play list:

https://www.youtube.com/watch?v=Sr7aYoWgMbg&list=PLoiqjtgvXf9e2VJk8GWEXwECPM_7JRwkE&index=32

Content:

Here I will be writing all the content I need to cover in the timer session:

First of all we need to revise what's a clk in a microprocessor:

Clk is the ticks that are digital square waves which make microprocess progress through its functionality like executing line-by-line code that helps the microprocessor to keep in track

In short, it's like a heart beat for us humans.

The clk has a period which we call a microprocessor clk or a tick which is usually very quick like some micro seconds or even faster.

$$clk\ Tick = \frac{1}{ftick}$$

Ok now let's talk about timers:

First of all it's a prephiral so we will find itsdriver in the MCALlayer

Like any other prephiral it has:

- HW circuit : that contains of system clk and prescaler

  A prescaler it's just a frequency divider that means it slows down the system clock

  So $ftick = \frac{fclk}{pre}$

- Register: where we can control this prephiral and we will be using systick register

The timer hardware circuit has two modes:

1. Timer mode: which is just a timer like a stop watch if the time run out it will do an interrupt so this is periodic and internal

2. Counter mode: Which just used as a counter here you could use like an external switch to count from the user so this is not periodic and external

Now as for the register:

It contains R= 8 or 10 bit for counting as it stores the # of ticks, and has an enable pin attached to an or gate which is used to determine which mode you will use

So we can say that the total time:

$$Ttotal = \#ticks * Ttick \;\; where \; \#ticks = 2^R \; , Tick = \frac{1}{ftick} = \frac{pre}{fclk}$$

$$Ttotatl = 2^R * \frac{pre}{fclk}$$

Moving on to the timer modes:

First we got the overflow (OV-F) or normal mode:

Where we begin at zero (if it counts up ++) till we hit the over flow number which is 2^R notice that we move from a number to another each tick

So for example if we have a required time for application = X ms we can have three cases:

X = Tov or less or bigger