```
// Author : Jose Goncalves
// jose.braga.pt@gmail.com

// Teacher : Nuno Peixoto
// University : IPCA
// 20.02.2014

// Board STM32F4
// blink onboard LED using CMIS


#include <stm32f4xx.h>




void Init_TIM2(void)
{
      TIM_TimeBaseInitTypeDef timerInitStructure;

      //Inicializa TIM2 peripheral clock : APB1
    RCC_APB1PeriphClockCmd(RCC_APB1Periph_TIM2, ENABLE);

    timerInitStructure.TIM_Prescaler = 42000;    // Prescaler  84MHz/42000 = 2Khz  => 2000 ciclos/sec
    timerInitStructure.TIM_CounterMode = TIM_CounterMode_Up;
    timerInitStructure.TIM_Period = 2000;
    timerInitStructure.TIM_ClockDivision = TIM_CKD_DIV1;
    timerInitStructure.TIM_RepetitionCounter = 0;
    TIM_TimeBaseInit(TIM2, &timerInitStructure);
    TIM_Cmd(TIM2, ENABLE);
}



void configure_PORTD_12()    // Configure PORTD , pin 12 ( LED 4 no PCB - verde )

{
      GPIO_InitTypeDef GPIO_InitStruct;

    //Inicializa GPIO peripheral clock : AHB1
    RCC_AHB1PeriphClockCmd (RCC_AHB1Periph_GPIOD, ENABLE);


    // Initilaise the GPIO port.

    GPIO_InitStruct.GPIO_Pin= GPIO_Pin_12;          // LED4 - Verde
    GPIO_InitStruct.GPIO_Mode= GPIO_Mode_OUT;       // pins as output
    GPIO_InitStruct.GPIO_Speed= GPIO_Speed_50MHz;  // GPIO modules clock speed
    GPIO_InitStruct.GPIO_OType= GPIO_OType_PP;      // pin type to push / pull (as opposed to open drain)
    GPIO_InitStruct.GPIO_PuPd= GPIO_PuPd_NOPULL;   // pullup/pulldown resistors inactive

    // GPIOD->MODER |= GPIO_Mode_OUT;          // GPIO_Mode_OUT  = 0x01
    // GPIOD->OSPEEDR |= GPIO_Speed_25MHz;     // GPIO_Speed_25MHz  = 0x01
    // GPIOD->OTYPER |= GPIO_OType_PP;         // GPIO_OType_PP = 0x00
    // GPIOD->PUPDR |= GPIO_PuPd_NOPULL;       // GPIO_PuPd_NOPULL = 0x00

    // this finally passes all the values to the GPIO_Initfunction which takes care of setting the corresponding bits.
    GPIO_Init(GPIOD, &GPIO_InitStruct);

}




int main ()

{

configure_PORTD_12();
Init_TIM2() ;


while(1)
    {
        int timerValue = TIM_GetCounter(TIM2);

              // 400/2000 . s = 1/5 . s = 200 ms
        if (timerValue == 400)
            GPIO_WriteBit(GPIOD, GPIO_Pin_12, Bit_RESET);
                    // GPIO_SetBits(GPIOD,GPIO_Pin_12);

                // 2000/2000 = 1s
        else if (timerValue == 2000)
            GPIO_WriteBit(GPIOD, GPIO_Pin_12, Bit_SET);
                    // GPIO_ResetBits(GPIOD,GPIO_Pin_12);
    }

}
```