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

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

// Board STM32F4
// blink onboard LED without using any API


// PERIPH_base + AHB1_PERIPH_base + GPIOD_base = 0x40000000  + 0x20000 + 0x0C00 =  0x40020C00
// GPIOD
# define GPIOD_MODER     0x40020C00   // MODER Address Offset : 0x00
# define GPIOD_OTYPER    0x40020C04   // OTYPER Address Offset : 0x04
# define GPIOD_OSPEEDER  0x40020C08   // OSPEEDER Address Offset : 0x08
# define GPIOD_PUPDR     0x40020C0C   // PUPR Address Offset : 0xC0
# define GPIOD_ODR       0x40020C14   // ODR Output value  Address Offset : 0x14
# define RCC_GPIOD       0x40023830   // RCC Register Clock Control AHB1

// TIMER 2
# define TIM2_CR1        0x40000000   // TIM2 Control Register 1
# define TIM2_Count      0x40000024   // TIM2 Counter
# define TIM2_Prescaler  0x40000028   // TIM2 Prescale Register
# define TIM2_ARR        0x4000002C   // TIM2 Auto-Reload Register
# define RCC_TIM2        0x40023840   // RCC Register Clock Control APB1


void Init_TIM2(void) // --------------------------------------------------------------------
{
     // Initialize TIM2 peripheral clock : APB1
       *(volatile unsigned long*)RCC_TIM2 |= 0x1 ;

       // Control Register 1  -  Reset value 0x0000
       *(volatile unsigned long*)TIM2_CR1 |=     1<<0 ;   // Bit 0     Counter enable
     //*(volatile unsigned long*)TIM2_CR1 |=  (1<<1);   // Bit 1     Update Enable
       //*(volatile unsigned long*)TIM2_CR1 |=   1<<2 ;   // Bit 2     Update Request Source
     //*(volatile unsigned long*)TIM2_CR1 |=   1<<3 ;   // Bit 3     One Pulse Mode
     //*(volatile unsigned long*)TIM2_CR1 |=  (1<<4);   // Bit 4     Counter used as upcounter
     //*(volatile unsigned long*)TIM2_CR1 |=  (1<<5);   // Bits 6:5  Center-aligned mode selection
     //*(volatile unsigned long*)TIM2_CR1 |=  (1<<6);   // Bits 6:5  Center-aligned mode selection
     //*(volatile unsigned long*)TIM2_CR1 |=  (1<<7);   // Bit 7     Auto-Reload
     //*(volatile unsigned long*)TIM2_CR1 |=  (1<<8);   // Bits 9:8  Clock division
     //*(volatile unsigned long*)TIM2_CR1 |=  (1<<9);   // Bits 9:8  Clock division

       *(volatile unsigned long*)TIM2_Prescaler = 42000 ; // Prescaler  84MHz/42000 = 2Khz  => 2000 ciclos/sec
                                                 // Specifies the prescaler value used to divide the TIM clock.
                                                 // This parameter can be a number between 0x0000 and 0xFFFF

        *(volatile unsigned long*)TIM2_ARR = 2000  ;       //  Specifies the prescaler value used to divide the TIM clock.
                                                 //  This parameter can be a number between 0x0000 and 0xFFFF
}



void configure_portD_12()   // --------------------------------------------------------------------
{
    //Inicializa GPIOD peripheral clock : AHB1
     // port mode register   01 : Output
    *(volatile unsigned long*)RCC_GPIOD |= ~(1 << 4);

    // pin 12 PORTD  - LED Verde
    *(volatile unsigned long*)GPIOD_MODER    &=  ~(1 << 25)  ;  // port mode register   01 : Output
    *(volatile unsigned long*)GPIOD_MODER    |=    1 << 24   ;
    *(volatile unsigned long*)GPIOD_OTYPER   &=  ~(1 << 12)  ;  // port type register    0 : Push Pull
    *(volatile unsigned long*)GPIOD_OSPEEDER &=  ~(3 << 2*12);  // port speed register  00 : low speed
    *(volatile unsigned long*)GPIOD_PUPDR    &=  ~(3 << 2*12);  //                      00 : No Pull
}



int main ()
{
   configure_portD_12() ;
   Init_TIM2() ;

   while(1)
      {

        if((*(volatile unsigned long*)TIM2_Count) == 400)        // 400/2000 . s = 1/5 . s = 200 ms
                  *(volatile unsigned long*)GPIOD_ODR &=  ~(1 << 12) ; // reset port D

        else if ((*(volatile unsigned long*)TIM2_Count) == 2000)  // 2000/2000 = 1s
                  *(volatile unsigned long*)GPIOD_ODR |=  (1 << 12) ;   // set port D
      }
}


// ver STM32F4_technical_overview.pdf
// http://www.farrellf.com/projects/hardware/2012-08-11_STM32F4_Basics:_Timers_(Part_1)/
// http://www.k-space.org/Class_Info/   - abrir file stm32 lecture 5   atenção não é F4
```