Tema 10: Integración y Características Especiales

Reloj en Tiempo Real (RTC)

Características y Funcionamiento

- El Reloj en Tiempo Real (RTC) es un dispositivo que permite medir el tiempo para mantener un calendario y un reloj
  - Evitando el uso de un temporizador para esta función.
- El del STM32L152RB ha sido diseñado para un consumo mínimo, pensado para sistemas alimentados por batería
  - Proporciona segundos, minutos, horas, día del mes, mes, año, día de la semana y día del año
  - Posee un divisor de reloj para que se ajuste a distintas frecuencias de oscilador
    - Aunque por defecto está configurado para ser usado a través del LSE
  - No tiene ningún mecanismo para mantener la hora tras una pérdida de energía
  - Tiene un mecanismo de protección para evitar modificaciones no voluntarias de los registros de control
  - Contiene funcionalidades más avanzadas que no se van a ver en este curso

Guía de Funcionamiento

- **Procedimiento de inicialización:**
  - Se desprotegen los registros de control RTC→WPR
    - Se escribe primero 0xCA y seguidamente 0x53
    - Escribir cualquier cosa vuelve a bloquear los registros
    - El desbloqueo se mantiene
  - Se pone a 1 el bit INIT en el registro RTC→ISR
  - Se espera a que se ponga a 1 el bit INITF en el registro RTC→ISR
  - Se programa el divisor de reloj, tanto síncrono como asíncrono (RTC→PRER)
    - Se hace en dos pasos. Primero el síncrono (16 lsb) y luego el asíncrono (16msb)
    - Para el uso con el LSE se ponen de valores 255 para el síncrono y 127 para el asíncrono
  - Se escribe el valor de la hora inicial en el registro RTC→TR
  - Se escribe el valor de la fecha inicial en el registro RTC→DR
  - Se selecciona el modo de 12 o 24 horas en el bit FMT del registro RTC→CR
  - Se pone a 0 el bit INIT del registro RTC→ISR
  - Se protegen los registros (RTC→WPR = 0)
- **Procedimiento de consulta de la hora:**
  - Se leen en cualquier momento los registros RTC→TR para la hora y RTC→DR para la fecha.

RTC: Registros de Control

- RTC→CR – Control Register:
  - Sólo se va a utilizar el bit 6 (FMT) para indicar si es formato 24h (con un '0') o si es formato 12h (con un '1').
  - El resto de bits se deia a '0'

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								COE	OSEL[1:0]		POL	Reserv-ed	BKP	SUB1H	ADD1H
								rw	rw	rw	rw		rw	w	w

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSIE	WUTIE	ALRBIE	ALRAIE	TSE	WUTE	ALRBE	ALRAE	DCE	FMT	Reser ved	REFCKON	TSEDGE	WUCKSEL[2:0]		
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw		rw	rw	rw	rw	rw

- RTC->PRER – Prescaler Register:
  - Configura el preescalado asíncrono (PREDIV_A) y el síncrono (PREDIV-S). Para el uso con el LSE se ponen de valores 255 para el síncrono y 127 para el asíncrono
  - La frecuencia de funcionamiento del RTC será:

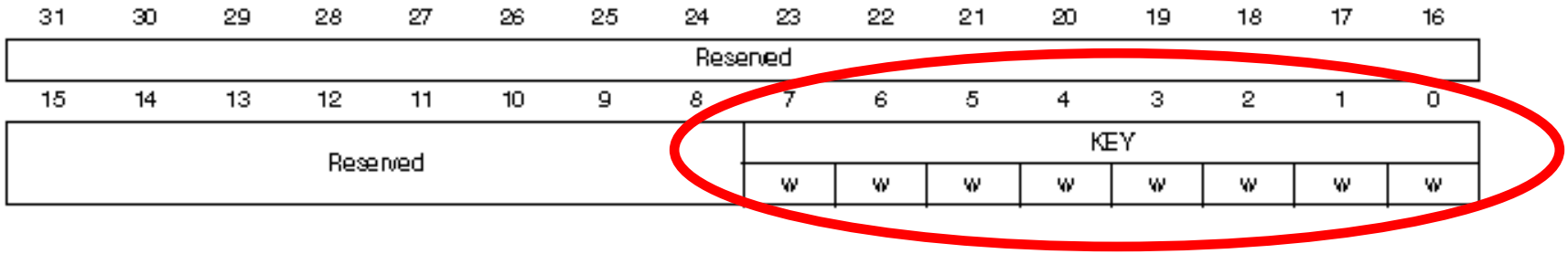
$$f_{CK\_SPRE} = \frac{f_{RTCCLK}}{(PREDIV\_S + 1) \times (PREVID\_A + 1)}$$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Resened									PREDIV_A[6:0]						
									rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Resened			PREDIV_S[12:0]												
			rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

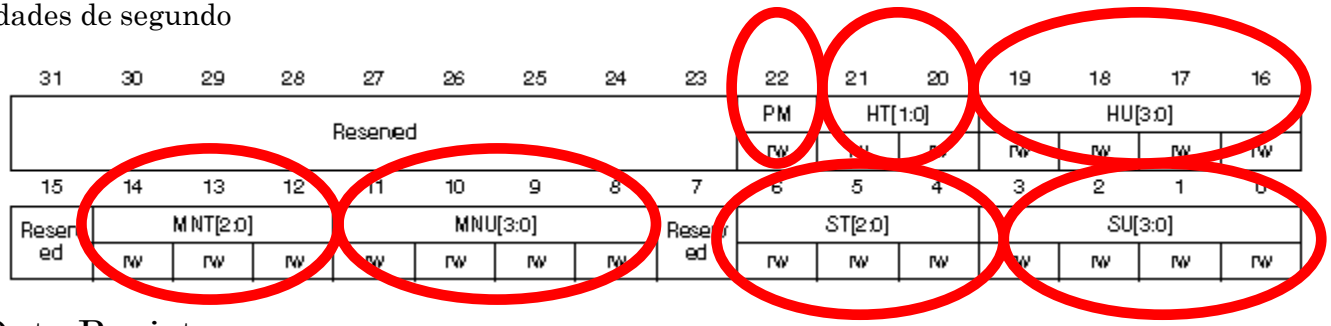
RTC: Registros de Control

- RTC→WPR – Write Protection Register:
  - Se utiliza como se ha comentado anteriormente
    - Primero se desprotegen los registros de control escribiendo primero 0xCA y seguidamente 0x53
    - Cuando se haya terminado de configurar todo, protegen los registros escribiendo 0x00

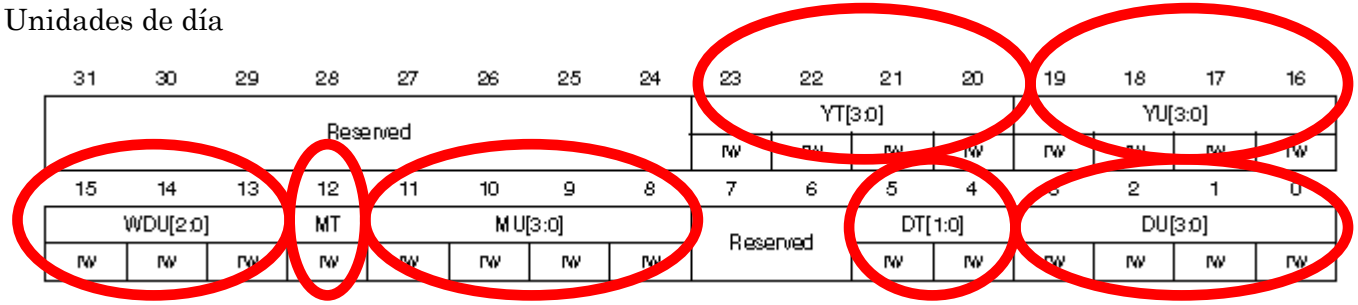


RTC: Registros de Datos

- RTC→TR – Time Register:
  - Contiene el valor de la hora, en la siguiente estructura (formato BCD):
    - PM: Con un 0 es formato de 24h y con un 1 es AM/PM
    - HT: Decenas de hora
    - HU: Unidades de hora
    - MINT: Decenas de minuto
    - MINU: Unidades de minuto
    - ST: Decenas de segundo
    - SU: Unidades de segundo



- RTC→DR – Date Register:
  - Contiene el valor de la fecha, en la siguiente estructura (formato BCD):
    - YT: Decenas de año
    - YU: Unidades de año
    - WDU: Tres bits que indican el día de la semana (000-prohibido; 001-lunes; …; 111-domingo)
    - MT: Decenas de mes
    - MU: Unidades de mes
    - DT: Decenas de día
    - DU: Unidades de día



RTC: Registros de Estado

- RTC→ISR – Initialization and Status Register:
  - De todos los bits sólo interesan:
    - INIT - Bit de control:
      - 1 – Se pone el RTC en modo inicialización
      - 0 – Se quita el modo de inicialización, actualizando los nuevos valores y arrancando el RTC
    - INITF - Bit de estado:
      - 0 – El modo de inicialización está desactivado.
      - 1 – El modo de inicialización está activado y ya se puede escribir en los registros.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	Res.	TAMP1F	TSOVF	TSF	WUTF	ALRBF	ALRAF	INIT	INITF	RSF	INITS	Res.	WUTWF	ALRBWF	ALRAWF
		rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rw	r	rc_w0	r		r	r	r

EJEMPLO: CONFIGURACIÓN Y USO RTC (1)

- Se muestra la fecha (23/03/16) y la hora (empezando a las 18:30:20) alternativamente cada segundo)

```
#include "stm32l1xx.h"
#include "Biblioteca_SDM.h"
#include "Utiles_SDM.h"

int main(void){
   unsigned char cadena[6];
   unsigned valor;
   Init_SDM();
   Init_LCD();
   LCD_Limpia();

   // Inicializa el RTC

   // Desprotege los registros del RTC (obligatorio el valor 0xCA y luego 0x53)
   RTC->WPR=0xCA;
   RTC->WPR=0x53;

   // INIT = 1 en RTC->ISR (bit 7 del registro)
   RTC->ISR |= (1<<7);

   // Espera a INITF = 1 (bit 6 del registro)
   while ((RTC->ISR & (1<<6))==0);

   // Programa el RTC_PRER (la parte síncrona, siempre 255 con el LSB)
   RTC->PRER=255;

   // Programa el RTC_PRER (la parte asíncrona, siempre 127 con el LSB)
   RTC->PRER|=127<<16;

   // Carga el valor inicial en RTC->TR y RTC->DR y el formato de hora (bit FMT en RTC->CR)
   RTC->TR = 0x00183020; // PM = 0 (formato de 24 horas), Hora = 18, Minutos = 30, Segundos = 20
   RTC->DR = 0x00166323; // Año = 16, Día de la semana = 011 (miércoles), Mes = 03, Día = 23
   RTC->CR = 0x00000000; // FNMT = 0 ->  Modo de 24h

   // INIT = 0, o sea, se quita el modo de inicialización, se actualizan los nuevos valores y se arranca el RTC
   RTC->ISR &= ~(1<<7);

   // Vuelvo a proteger los registros (obligatorio el valor 0x00)
   RTC->WPR=0;

   while ((RTC->ISR & (1<<6))!=0);   // Si esta el modo de inicialización activado, espero
```

EJEMPLO: CONFIGURACIÓN Y USO RTC (2)

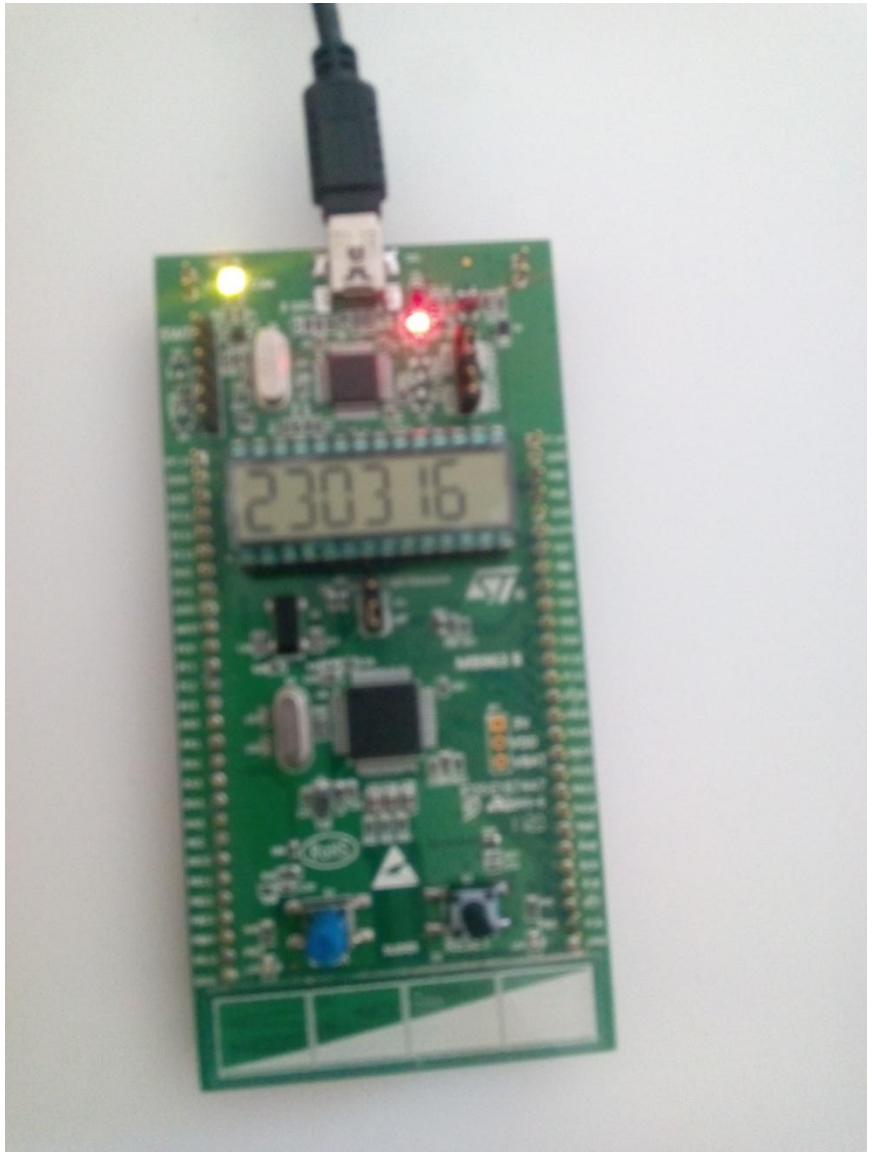
- Se muestra la fecha (23/03/16) y la hora (empezando a las 18:30:20) alternativamente cada segundo)

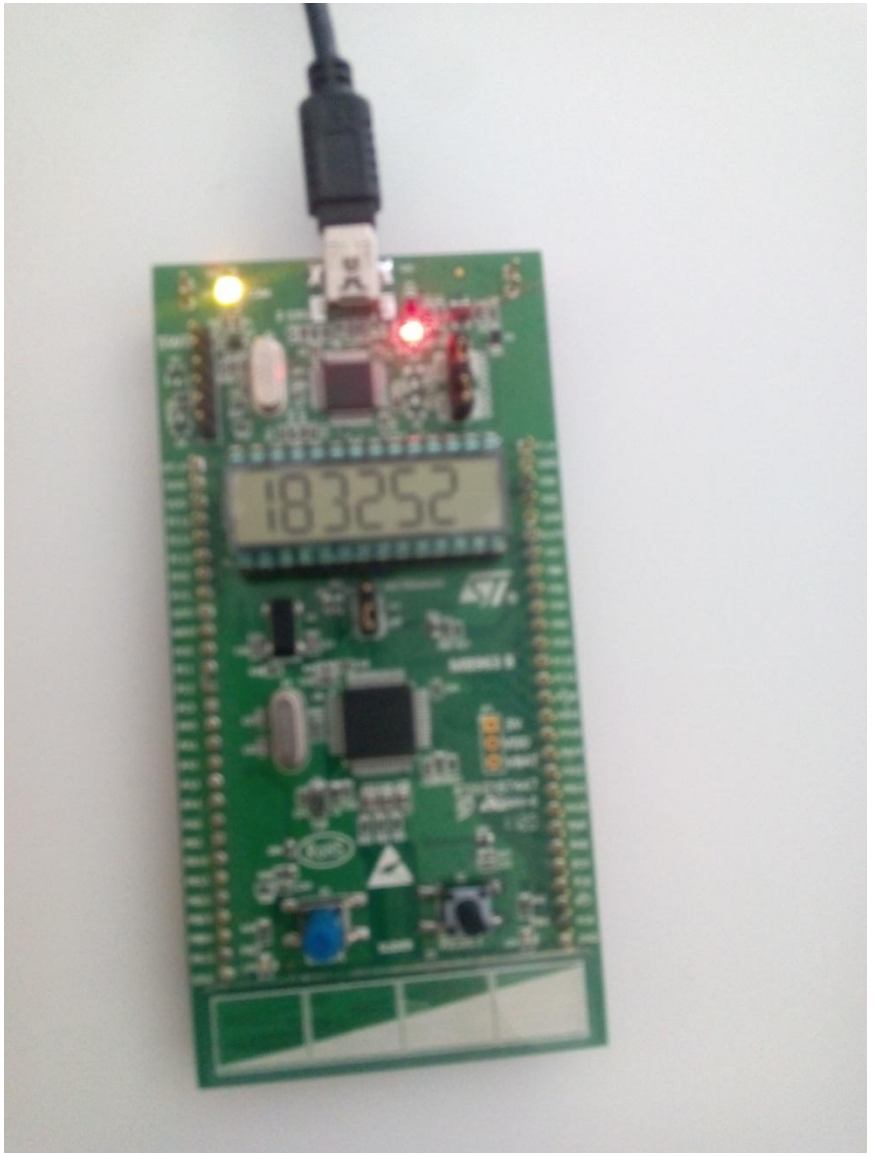
```
while (1) {

    // Lee y muestra la hora
    valor = RTC->TR;
    cadena[5]=(valor & 0x0000000F)+'0';
    valor = valor >> 4;
    cadena[4]=(valor & 0x00000007)+'0';
    valor = valor >> 4;
    cadena[3]=(valor & 0x0000000F)+'0';
    valor = valor >> 4;
    cadena[2]=(valor & 0x00000007)+'0';
    valor = valor >> 4;
    cadena[1]=(valor & 0x0000000F)+'0';
    valor = valor >> 4;
    cadena[0]=(valor & 0x00000003)+'0';
    valor = valor >> 4;
    LCD_Texto(cadena);
    espera(10000000);    // Espera un tiempo para mostrar la fecha

    // Lee y muestra la fecha
    valor = RTC->DR;
    cadena[1]=(valor & 0x0000000F)+'0';
    valor = valor >> 4;
    cadena[0]=(valor & 0x00000003)+'0';
    valor = valor >> 4;
    cadena[3]=(valor & 0x0000000F)+'0';
    valor = valor >> 4;
    cadena[2]=(valor & 0x00000001)+'0';
    valor = valor >> 4;
    cadena[5]=(valor & 0x0000000F)+'0';
    valor = valor >> 4;
    cadena[4]=(valor & 0x0000000F)+'0';
    valor = valor >> 4;
    LCD_Texto(cadena);
    espera(10000000);   // Espera un tiempo para mostrar la hora
    }
 }
```

PRUEBA DEL EJEMPLO EXPLICADO





Tema 10: Integración y Características Especiales

Watchdog

Características Generales

- Un Watchdog es un mecanismo interno de control del microcontrolador para provocar el reset del sistema en caso de que se detecte que el micro ha perdido el control
- El sistema de Watchdog tiene las siguientes características:
  - Resetea el chip internamente si no se actualiza periódicamente
  - Se habilita por software, pero sólo se resetea por reset o por una interrupción de Watchdog
  - Un uso incorrecto o incompleto, provoca también el reset
  - Utiliza internamente un temporizador

Tema 10: Integración y Características Especiales

Bajo Consumo

Bajo Consumo

- Al crear un sistema electrónico, uno de los requisitos principales es que su consumo sea mínimo:
  - Por motivos de calificación energética
  - Cuando el sistema es portátil y ha de ser alimentado con baterías
- Todos los microcontroladores actuales tienen la posibilidad de definir distintos modos de bajo consumo:
  - Totalmente operativo (máximo consumo)
  - Determinados periféricos desconectados
  - Todos los periféricos desconectados (salvo algún PIN)
  - La mayor parte de la CPU desconectada (mínimo consumo)
- El ubicar al dispositivo en uno de los modos de bajo consumo, lo realiza el programador de la aplicación cuando se cumplen determinados requisitos
- El "despertar" al micro de un estado de bajo consumo se realiza mediante una IRQ (por ejemplo EINT)