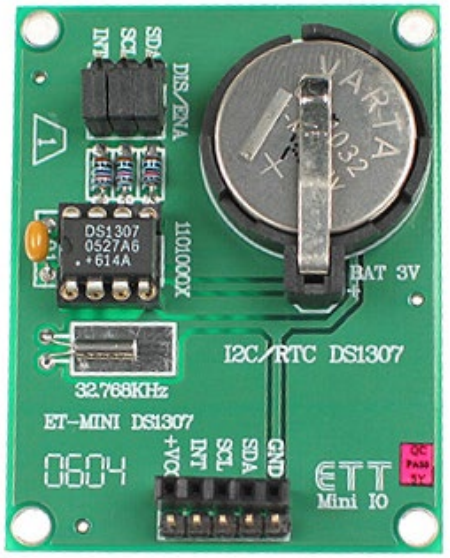
Reloj de Tiempo Real (RTC)

Basado en el material de Pablo Irrera Condines

Introducción

- Es un circuito que posee toda la funcionalidad para la gestión de fecha y hora.
- Están presentes en la mayoría de los dispositivos que necesitan guardar el tiempo en forma precisa.



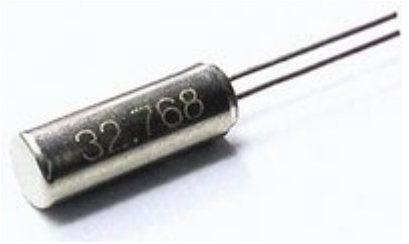
Introducción

Ventajas

- Bajo consumo de energía
- Libera de carga de trabajo al sistema principal para que pueda dedicarse a tareas más críticas.
- Suele ser más preciso que otros métodos.

Medición de tiempo

- Suele utilizarse un oscilador a cristal de 32,768KHz ya que con un divisor de frecuencia de 2^{15} se obtiene un clock de 1Hz (2^{15} = 32768)



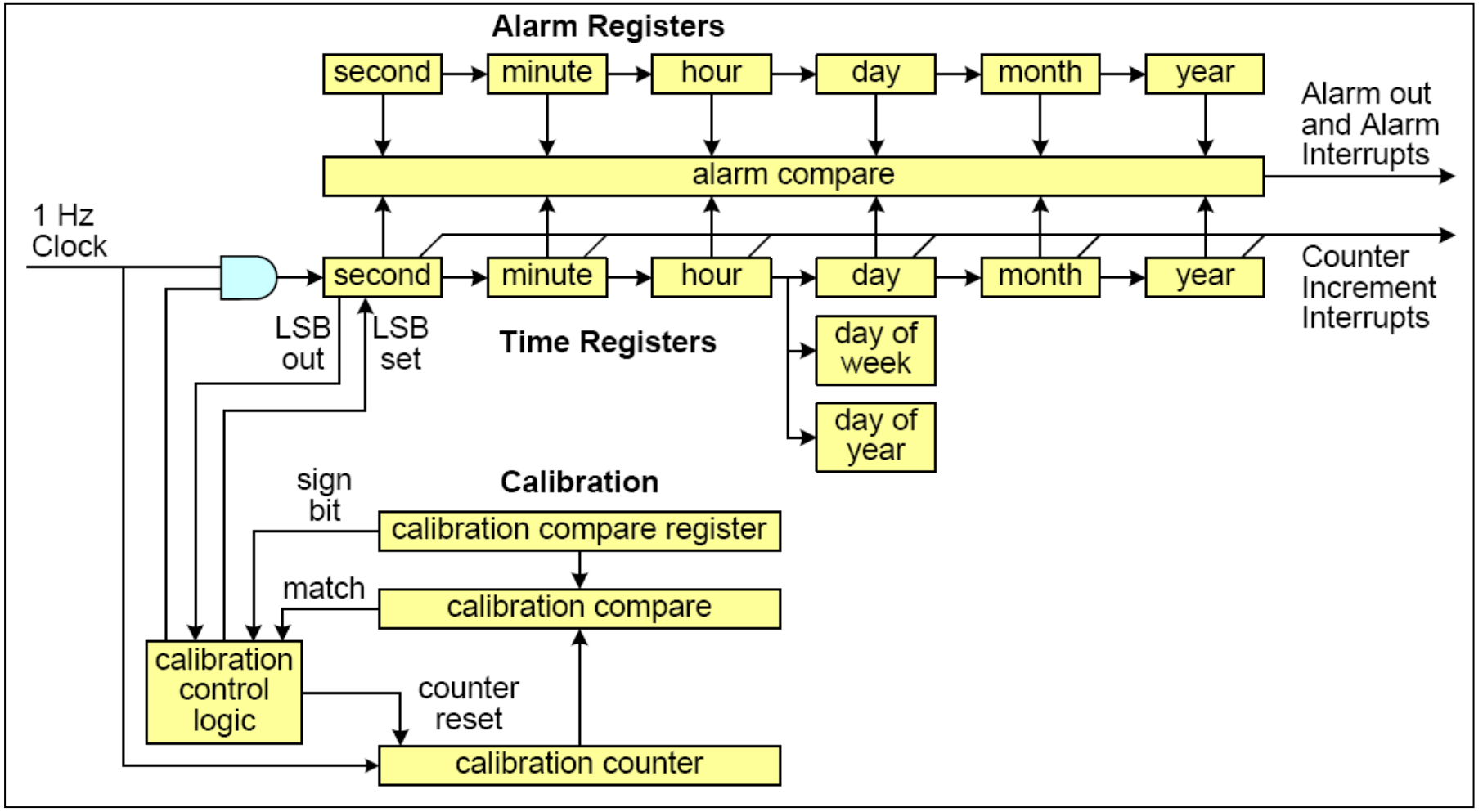
Introducción

Alimentación

- Normalmente se lo alimenta con una pila de litio tipo CR2032 (como en las PC).

- Como alternativa podemos utilizar un super-capacitor que se mantiene cargado mientras está encendido el equipo y que puede durar varios días con el equipo apagado.

Diagrama funcional en bloques



Registros

Table 507. Real-Time Clock register map

Name	Description	Access	Reset Value[1]	Address
Miscellaneous registers (see Section 27.6.2)				
ILR	Interrupt Location Register	R/W	0	0x4002 4000
CCR	Clock Control Register	R/W	NC	0x4002 4008
CIIR	Counter Increment Interrupt Register	R/W	0	0x4002 400C
AMR	Alarm Mask Register	R/W	0	0x4002 4010
RTC_AUX	RTC Auxiliary control register	R/W	0x8	0x4002 405C
RTC_AUXEN	RTC Auxiliary Enable register	R/W	0	0x4002 4058
Consolidated time registers (see Section 27.6.3)				
CTIME0	Consolidated Time Register 0	RO	NC	0x4002 4014
CTIME1	Consolidated Time Register 1	RO	NC	0x4002 4018
CTIME2	Consolidated Time Register 2	RO	NC	0x4002 401C
Time counter registers (see Section 27.6.4)				
SEC	Seconds Counter	R/W	NC	0x4002 4020
MIN	Minutes Register	R/W	NC	0x4002 4024
HOUR	Hours Register	R/W	NC	0x4002 4028
DOM	Day of Month Register	R/W	NC	0x4002 402C
DOW	Day of Week Register	R/W	NC	0x4002 4030
DOY	Day of Year Register	R/W	NC	0x4002 4034
MONTH	Months Register	R/W	NC	0x4002 4038
YEAR	Years Register	R/W	NC	0x4002 403C
CALIBRATION	Calibration Value Register	R/W	NC	0x4002 4040
General purpose registers (see Section 27.6.6)				
GPREG0	General Purpose Register 0	R/W	NC	0x4002 4044
GPREG1	General Purpose Register 1	R/W	NC	0x4002 4048
GPREG2	General Purpose Register 2	R/W	NC	0x4002 404C
GPREG3	General Purpose Register 3	R/W	NC	0x4002 4050
GPREG4	General Purpose Register 4	R/W	NC	0x4002 4054
Alarm register group (see Section 27.6.7)				
ALSEC	Alarm value for Seconds	R/W	NC	0x4002 4060
ALMIN	Alarm value for Minutes	R/W	NC	0x4002 4064
ALHOUR	Alarm value for Hours	R/W	NC	0x4002 4068
ALDOM	Alarm value for Day of Month	R/W	NC	0x4002 406C
ALDOW	Alarm value for Day of Week	R/W	NC	0x4002 4070
ALDOY	Alarm value for Day of Year	R/W	NC	0x4002 4074
ALMON	Alarm value for Months	R/W	NC	0x4002 4078
ALYEAR	Alarm value for Year	R/W	NC	0x4002 407C

[1] Reset values apply only to a power-up of the RTC block, other types of reset have no effect on this block. Since the RTC is powered whenever either of the $V_{DD(REG)(3V3)}$, or V_{BAT} supplies are present, power-up reset occurs only when both supplies were absent and then one is turned on. Most registers are not affected by power-up of the RTC and must be initialized by software if the RTC is enabled. The Reset Value reflects the data stored in used bits only. It does not include reserved bits content.

Macros para manejar los registros

```
#define     RTC             ( ( register_t  * ) 0x40024000UL )
#define     RTCILR          RTC[0].dword
#define     RTCCCR          RTC[2].dword
#define     RTCCIIR         RTC[3].dword
#define     RTCAMR          RTC[4].dword
#define     RTC_AUX         RTC[23].dword
#define     RTC_AUXEN       RTC[22].dword
#define     RTCCTIME0       RTC[5].dWord
#define     RTCCTIME1       RTC[6].dWord
#define     RTCCTIME2       RTC[7].dWord
#define     RTCSEC          RTC[8].dword
#define     RTCMIN          RTC[9].dword
#define     RTCHOUR         RTC[10].dword
#define     RTCDOM          RTC[11].dword
#define     RTCDOW          RTC[12].dword
#define     RTCDOY          RTC[13].dword
#define     RTCMONTH        RTC[14].dword
#define     RTCYEAR         RTC[15].dword
#define     RTCCALIBRATION  RTC[16].dword
#define     RTCGPREG0       RTC[17].dword
#define     RTCGPREG1       RTC[18].dword
#define     RTCGPREG2       RTC[19].dword
#define     RTCGPREG3       RTC[20].dword
#define     RTCGPREG4       RTC[21].dword
#define     RTCALSEC        RTC[24].dword
#define     RTCALMIN        RTC[25].dword
#define     RTCALHOUR       RTC[26].dword
#define     RTCALDOM        RTC[27].dword
#define     RTCALDOW        RTC[28].dword
#define     RTCALDOY        RTC[29].dword
#define     RTCALMON        RTC[30].dword
#define     RTCALYEAR       RTC[31].dword
```

Configuración

- **Habilitar PCONP** (Alimentación del periférico RTC)
- **Configurar CCR** (Registro de control de RTC)
- **Configurar RTC_AUXEN** (Registro de interrupción en caso de falla del RTC)
- **Configurar Hora**
- **Configurar CIIR** (Habilitar interrupciones periódicas del RTC)
- **Configurar interrupciones por alarma** (En caso de requerirlo)
- **Empezar a contar**
- **Habilitar interrupciones** (A nivel microcontrolador → NVIC)

Habilitar PCONP

If a peripheral control bit is 1, that peripheral is enabled. If a peripheral control bit is 0, that peripheral's clock is disabled (gated off) to conserve power. For example if bit 19 is 1, the I^2C1 interface is enabled. If bit 19 is 0, the I^2C1 interface is disabled.

Important: valid read from a peripheral register and valid write to a peripheral register is possible only if that peripheral is enabled in the PCONP register!

Table 46. Power Control for Peripherals register (PCONP - address 0x400F C0C4) bit description

Bit	Symbol	Description	Reset value
0	-	Reserved.	NA
1	PCTIM0	Timer/Counter 0 power/clock control bit.	1
2	PCTIM1	Timer/Counter 1 power/clock control bit.	1
3	PCUART0	UART0 power/clock control bit.	1
4	PCUART1	UART1 power/clock control bit.	1
5	-	Reserved.	NA
6	PCPWM1	PWM1 power/clock control bit.	1
7	PCI2C0	The I^2C0 interface power/clock control bit.	1
8	PCSPI	The SPI interface power/clock control bit.	1
9	PCRTC	The RTC power/clock control bit.	1
10	PCSSP1	The SSP 1 interface power/clock control bit.	1
11	-	Reserved.	NA
12	PCADC	A/D converter (ADC) power/clock control bit. **Note:** Clear the PDN bit in the AD0CR before clearing this bit, and set this bit before setting PDN.	0

```
PCONP |= (1 << 9); //Habilitamos el clock del RTC
```

Configurar CCR

27.6.2.2 Clock Control Register (CCR - 0x4002 4008)

The clock register is a 4-bit register that controls the operation of the clock divide circuit. Each bit of the clock register is described in Table 510. All NC bits in this register should be initialized when the RTC is first turned on.

Table 510. Clock Control Register (CCR - address 0x4002 4008) bit description

Bit	Symbol	Value	Description	Reset value
0	CLKEN		Clock Enable.	NC
		1	The time counters are enabled.	
		0	The time counters are disabled so that they may be initialized.	

Table 510. Clock Control Register (CCR - address 0x4002 4008) bit description *...continued*

Bit	Symbol	Value	Description	Reset value
1	CTCRST		CTC Reset.	0
		1	When one, the elements in the internal oscillator divider are reset, and remain reset until CCR[1] is changed to zero. This is the divider that generates the 1 Hz clock from the 32.768 kHz crystal. The state of the divider is not visible to software.	
		0	No effect.	
3:2	-		Internal test mode controls. These bits must be 0 for normal RTC operation.	NC
4	CCALEN		Calibration counter enable.	NC
		1	The calibration counter is disabled and reset to zero.	
		0	The calibration counter is enabled and counting, using the 1 Hz clock. When the calibration counter is equal to the value of the CALIBRATION register, the counter resets and repeats counting up to the value of the CALIBRATION register. See Section 27.6.4.2 and Section 27.6.5.	
31:5	-		Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

(1) Sin calib.	**(0) Normal**	**(0) Normal**	**(1) Resteo mientras configuro**	**(0) CLKEN dehabilitado**

```
CCR = 0x12; // Divisor de clock y el contador en reset. Calibracion deshab.
```

Configurar hora

Table 519. Time Counter registers

Name	Size	Description	Access	Address
SEC	6	Seconds value in the range of 0 to 59	R/W	0x4002 4020
MIN	6	Minutes value in the range of 0 to 59	R/W	0x4002 4024
HOUR	5	Hours value in the range of 0 to 23	R/W	0x4002 4028
DOM	5	Day of month value in the range of 1 to 28, 29, 30, or 31 (depending on the month and whether it is a leap year).[1]	R/W	0x4002 402C
DOW	3	Day of week value in the range of 0 to 6[1]	R/W	0x4002 4030
DOY	9	Day of year value in the range of 1 to 365 (366 for leap years)[1]	R/W	0x4002 4034
MONTH	4	Month value in the range of 1 to 12	R/W	0x4002 4038
YEAR	12	Year value in the range of 0 to 4095	R/W	0x4002 403C

A nivel aplicación

```
Fijar_tiempo(2020,9,7,11,0,0);
// 11:00:00 7/9/2020
```

A nivel drivers

```
YEAR = anio;
MONTH = mes;
DOM = dia;
HOUR = hora;
MIN = minutos;
SEC = segundos;
```

Nota: Los registros son R/W por lo que se puede generar una primitiva que obtenga la hora

Configurar interrupciones (CIIR)

27.6.2.3 Counter Increment Interrupt Register (CIIR - 0x4002 400C)

The Counter Increment Interrupt Register (CIIR) gives the ability to generate an interrupt every time a counter is incremented. This interrupt remains valid until cleared by writing a 1 to bit 0 of the Interrupt Location Register (ILR[0]).

Table 511. Counter Increment Interrupt Register (CIIR - address 0x4002 400C) bit description

Bit	Symbol	Description	Reset value
0	IMSEC	When 1, an increment of the Second value generates an interrupt.	0
1	IMMIN	When 1, an increment of the Minute value generates an interrupt.	0
2	IMHOUR	When 1, an increment of the Hour value generates an interrupt.	0
3	IMDOM	When 1, an increment of the Day of Month value generates an interrupt.	0
4	IMDOW	When 1, an increment of the Day of Week value generates an interrupt.	0
5	IMDOY	When 1, an increment of the Day of Year value generates an interrupt.	0
6	IMMON	When 1, an increment of the Month value generates an interrupt.	0
7	IMYEAR	When 1, an increment of the Year value generates an interrupt.	0
31:8	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

```
CIIR = 0x01; // Cada un segundo me da una interrupcion.
```

Configurar alarmas (1)

27.6.2.4 Alarm Mask Register (AMR - 0x4002 4010)

The Alarm Mask Register (AMR) allows the user to mask any of the alarm registers. Table 512 shows the relationship between the bits in the AMR and the alarms. For the alarm function, every non-masked alarm register must match the corresponding time counter for an interrupt to be generated. The interrupt is generated only when the counter comparison first changes from no match to match. The interrupt is removed when a one is written to the appropriate bit of the Interrupt Location Register (ILR). If all mask bits are set. then the alarm is disabled.

Table 512. Alarm Mask Register (AMR - address 0x4002 4010) bit description

Bit	Symbol	Description	Reset value
0	AMRSEC	When 1, the Second value is not compared for the alarm.	0
1	AMRMIN	When 1, the Minutes value is not compared for the alarm.	0
2	AMRHOUR	When 1, the Hour value is not compared for the alarm.	0
3	AMRDOM	When 1, the Day of Month value is not compared for the alarm.	0
4	AMRDOW	When 1, the Day of Week value is not compared for the alarm.	0
5	AMRDOY	When 1, the Day of Year value is not compared for the alarm.	0
6	AMRMON	When 1, the Month value is not compared for the alarm.	0
7	AMRYEAR	When 1, the Year value is not compared for the alarm.	0
31:8	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

Pongo en 1 solamente los valores por los cuáles voy a comparar para la alarma.

Configurar alarmas (2)

27.6.7 Alarm register group

The alarm registers are shown in Table 522. The values in these registers are compared with the time counters. If all the unmasked (See Section 27.6.2.4 “Alarm Mask Register (AMR - 0x4002 4010)” on page 574) alarm registers match their corresponding time counters then an interrupt is generated. The interrupt is cleared when a 1 is written to bit 1 of the Interrupt Location Register (ILR[1]).

Table 522. Alarm registers

Name	Size	Description	Access	Address
ALSEC	6	Alarm value for Seconds	R/W	0x4002 4060
ALMIN	6	Alarm value for Minutes	R/W	0x4002 4064
ALHOUR	5	Alarm value for Hours	R/W	0x4002 4068
ALDOM	5	Alarm value for Day of Month	R/W	0x4002 406C
ALDOW	3	Alarm value for Day of Week	R/W	0x4002 4070
ALDOY	9	Alarm value for Day of Year	R/W	0x4002 4074
ALMON	4	Alarm value for Months	R/W	0x4002 4078
ALYEAR	12	Alarm value for Years	R/W	0x4002 407C

Escribo los valores de tiempo con los cuáles voy a comparar el tiempo actual.

Nota: En el ejemplo no utilizaremos alarmas

Arranco RTC – Reconfiguro CCR

(1) Sin calib.	(0) Normal	(0) Normal	(0) NO RESETEO	(1) CLKEN HABILITADO

```
CCR = 0x11; // Habilitamos el RTC.
```

Habilito interrupciones del NVIC

Table 52. Interrupt Set-Enable Register 0 register (ISER0 - 0xE000 E100)

Bit	Name	Function
0	ISE_WDT	Watchdog Timer Interrupt Enable. Write: writing 0 has no effect, writing 1 enables the interrupt. Read: 0 indicates that the interrupt is disabled, 1 indicates that the interrupt is enabled.
1	ISE_TIMER0	Timer 0 Interrupt Enable. See functional description for bit 0.
2	ISE_TIMER1	Timer 1. Interrupt Enable. See functional description for bit 0.
3	ISE_TIMER2	Timer 2 Interrupt Enable. See functional description for bit 0.
4	ISE_TIMER3	Timer 3 Interrupt Enable. See functional description for bit 0.
5	ISE_UART0	UART0 Interrupt Enable. See functional description for bit 0.
6	ISE_UART1	UART1 Interrupt Enable. See functional description for bit 0.
7	ISE_UART2	UART2 Interrupt Enable. See functional description for bit 0.
8	ISE_UART3	UART3 Interrupt Enable. See functional description for bit 0.
9	ISE_PWM	PWM1 Interrupt Enable. See functional description for bit 0.
10	ISE_I2C0	I^2C0 Interrupt Enable. See functional description for bit 0.
11	ISE_I2C1	I^2C1 Interrupt Enable. See functional description for bit 0.
12	ISE_I2C2	I^2C2 Interrupt Enable. See functional description for bit 0.
13	ISE_SPI	SPI Interrupt Enable. See functional description for bit 0.
14	ISE_SSP0	SSP0 Interrupt Enable. See functional description for bit 0.
15	ISE_SSP1	SSP1 Interrupt Enable. See functional description for bit 0.
16	ISE_PLL0	PLL0 (Main PLL) Interrupt Enable. See functional description for bit 0.
17	ISE_RTC	Real Time Clock (RTC) Interrupt Enable. See functional description for bit 0.
18	ISE_EINT0	External Interrupt 0 Interrupt Enable. See functional description for bit 0.
19	ISE_EINT1	External Interrupt 1 Interrupt Enable. See functional description for bit 0.

```
ISER0 = (1 << 17);
```

¡Ya está configurado el RTC y funcionando!

¿Cómo sabemos si interrumpió por alarma o por contador?

Interrupciones

27.6.2.1 Interrupt Location Register (ILR - 0x4002 4000)

The Interrupt Location Register is a 2-bit register that specifies which blocks are generating an interrupt (see Table 509). Writing a one to the appropriate bit clears the corresponding interrupt. Writing a zero has no effect. This allows the programmer to read this register and write back the same value to clear only the interrupt that is detected by the read.

Table 509. Interrupt Location Register (ILR - address 0x4002 4000) bit description

Bit	Symbol	Description	Reset value
0	RTCCIF	When one, the Counter Increment Interrupt block generated an interrupt. Writing a one to this bit location clears the counter increment interrupt.	0
1	RTCALF	When one, the alarm registers generated an interrupt. Writing a one to this bit location clears the alarm interrupt.	0
31:21	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

```
if(ILR & 0x01) // Si interrumpio por counter
    {
    ILR |= 0x01; //Limpio el flag
    //HAGO ALGO
    }
if(ILR & 0x02) // Si interrumpio por alarma
    {
    ILR |= 0x02; //Limpio el flag
    //HAGO ALGO
    }
```

Inicialización del RTC

```
void Init_RTC(void)
{
	PCONP |= (1 << 9); //Habilitamos el clock del RTC

	CCR = 0x12;
	// Ponemos el divisor de clock y el contador en reset. Calibracion deshab.

	RTC_AUXEN &= ~(0x01 << 4); // Deshabilito la interrupcion por fail del clock.

	Fijar_tiempo(2020,9,7,11,0,0);
	// Fijamos la hora en 00:00:00. Primeros tres argumentos
	// Año, mes, dia. Ultimos tres argumentos: horas, minutos, segundos.

	CIIR = 0x01; // Cada un segundo me da una interrupcion.
	CCR = 0x11;
	// Habilitamos el RTC. A partir de que se ejecuta esta linea es que empieza a contar.

	ISER0 = (1 << 17);
}
```

Fijar_tiempo

```
void Fijar_tiempo(uint32_t anio, uint8_t mes, uint8_t dia, uint8_t hora, uint8_t
minutos, uint8_t segundos)
{
	YEAR = anio;

	MONTH = mes;

	DOM = dia;

	HOUR = hora;

	MIN = minutos;

	SEC = segundos;
}
```

ISR RTC

```
void RTC_IRQHandler(void)
{
	// Si interrumpio por counter
	if (ILR & 0x01)
	{
		ILR |= 0x01;
		if(GetPIN(PORT0,22))
		{
			SetPIN(PORT0,22,0);
		}
		else
		{
			SetPIN(PORT0,22,1);
		}
	}
	// Si interrumpio por alarma
	if (ILR & 0x02)
	{

	}
}
```