Instrumentación de Tiempo Real

UNIVERSIDAD DE CANTABRIA

Necesidades de temporización en programas de tiempo real

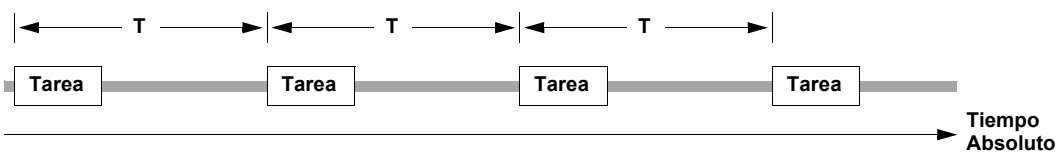
UNIVERSIDAD DE CANTABRIA

La temporización es uno de los aspectos más característicos de los sistemas de tiempo real

- **necesidad de realizar acciones con una periodicidad determinada o en un instante de tiempo prefijado**

Requerimientos temporales más habituales en sistemas de tiempo real:

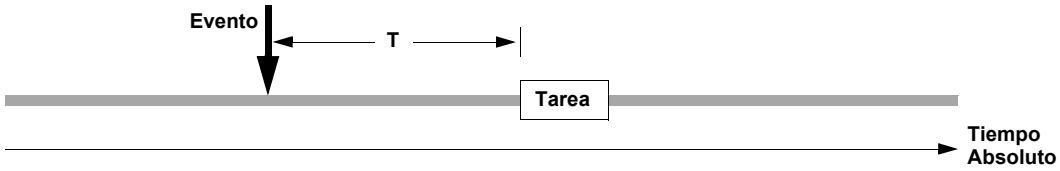
- **Realización de una tarea periódica**



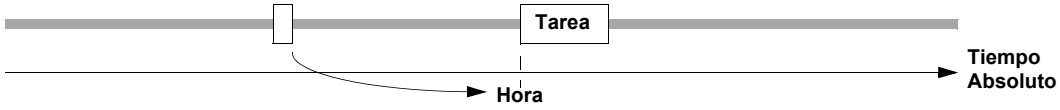
Necesidades de temporización en programas de tiempo real (cont.)

UNIVERSIDAD DE CANTABRIA

- **Tarea con tiempo relativo de respuesta**



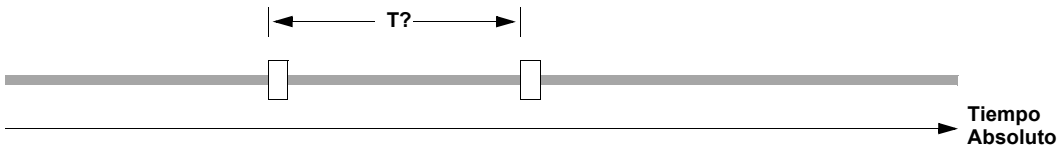
- **Tarea a realizar en un instante de tiempo absoluto**



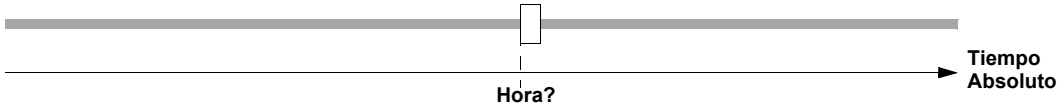
Necesidades de temporización en programas de tiempo real (cont.)

UNIVERSIDAD DE CANTABRIA

- **Medida de tiempo relativo**



- **Datación absoluta de eventos**



Recursos hardware para gestión del tiempo

UNIVERSIDAD DE CANTABRIA

Temporizador:

- contador de ciclos de un reloj de frecuencia fija y conocida
- cuando la cuenta llega a 0 se produce una interrupción

Reloj de tiempo real:

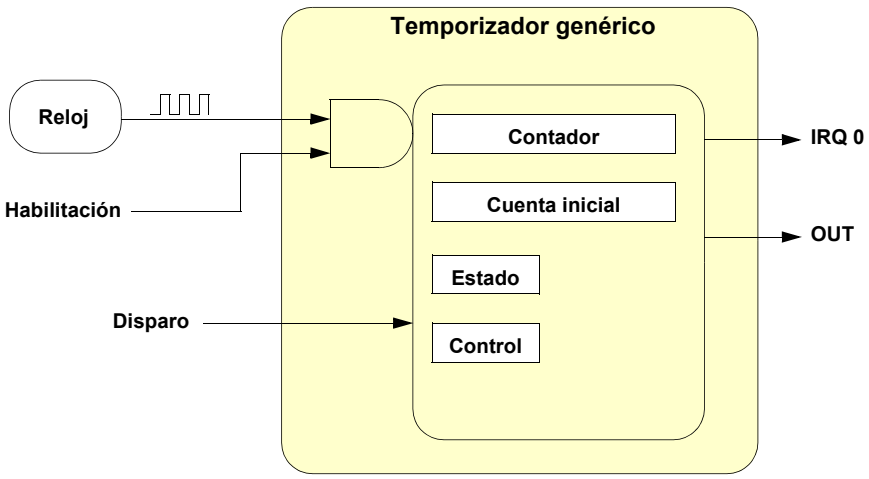
- reloj utilizado para obtener la hora y la fecha, normalmente con precisión de segundos
- tiene en cuenta cambios de hora "políticos" (compensación a fin de año y cambios de horario verano/invierno)

Reloj monótono:

- el valor leido siempre crece con el transcurso del tiempo
- utilizado por MaRTE OS (Time Stamp Counter)

Descripción de un temporizador genérico

UNIVERSIDAD DE CANTABRIA



Descripción de un temporizador genérico (cont.)

UNIVERSIDAD DE CANTABRIA

Cuenta inicial: valor inicial que se carga en el contador al comienzo de cada cuenta

Contador: valor actual de la cuenta

- comienza al valor *Cuenta inicial* y se va decrementando con cada ciclo de reloj
- cuando su valor llega a 0 se produce una interrupción

Número de bits de la cuenta (N): este número indica el máximo valor que se puede contar (2^N)

- la máxima duración del intervalo para la que es posible programar el temporizador será 2^N * periodo del reloj

Habilitación: señal hardware que habilita o inhibe la cuenta

Descripción de un temporizador genérico (cont.)

UNIVERSIDAD DE CANTABRIA

Registro de estado: permite leer el estado del temporizador

- lo más relevante será si ha terminado la cuenta o no

Registro de control: permite establecer el modo de operación del temporizador

Modo de operación: establece los siguientes aspectos

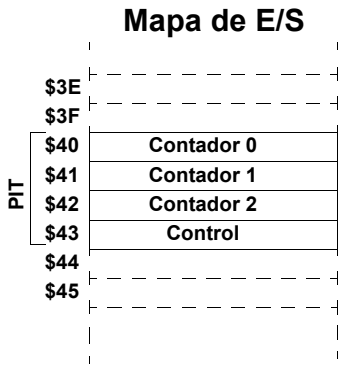
- ¿cómo se inicia la cuenta? por orden software, por señal externa, fin de cuenta anterior, ...
- ¿qué hacer cuando finaliza una cuenta? modificar salida, generar interrupción, ...
- ...

Temporizador programable (PIT) (8253/8254)

UNIVERSIDAD DE CANTABRIA

Contiene 3 temporizadores independientes, cada uno de ellos de de 16 bits

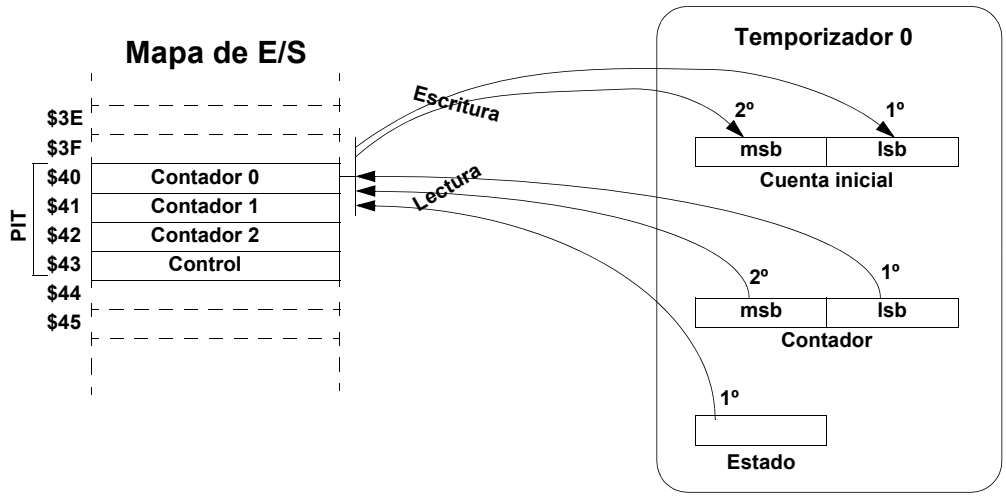
- **cuentan impulsos de un reloj de frecuencia 1.19318 MHz, lo que equivale a un periodo de 838.0965 ns**
- **máxima longitud de cuenta: 2^{16} * 838.0965 ≅ 55 ms**
- **su programación se realiza mediante 4 registros de E/S**



PIT
Estructura interna de registros

UNIVERSIDAD DE CANTABRIA

Mediante el registro destinado a cada temporizador es posible acceder a todos sus registros internos



PIT
Registros de E/S

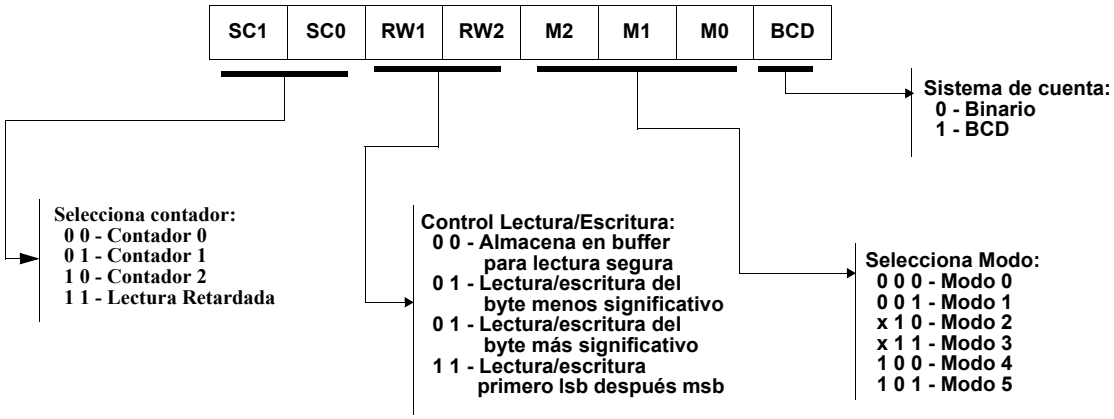
UNIVERSIDAD DE CANTABRIA

Dirección E/S	Tipo Operación	Función
$40	Escritura	Establece valor inicial del temporizador 0
$40	Lectura	Lee cuenta y/o estado del temporizador 0
$41	Escritura	Establece valor inicial del temporizador 1
$41	Lectura	Lee cuenta y/o estado del temporizador 1
$42	Escritura	Establece valor inicial del temporizador 2
$42	Lectura	Lee cuenta y/o estado del temporizador 2
$43	Escritura	Establece modo de control
$43	Lectura	Operación no permitida

PIT
Registro de control

UNIVERSIDAD DE CANTABRIA

Permite configurar el modo de operación de los contadores



Modo 0:
Ciclo simple

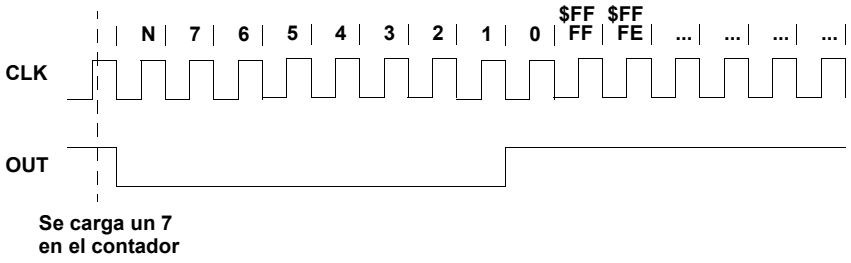
UNIVERSIDAD DE CANTABRIA

La cuenta se inicia:

- **al establecer el modo**
- **o al escribir un nuevo valor de cuenta inicial**

La salida se mantiene a nivel bajo hasta que finaliza la cuenta

El contador sólo cuenta si `Gate` vale 1



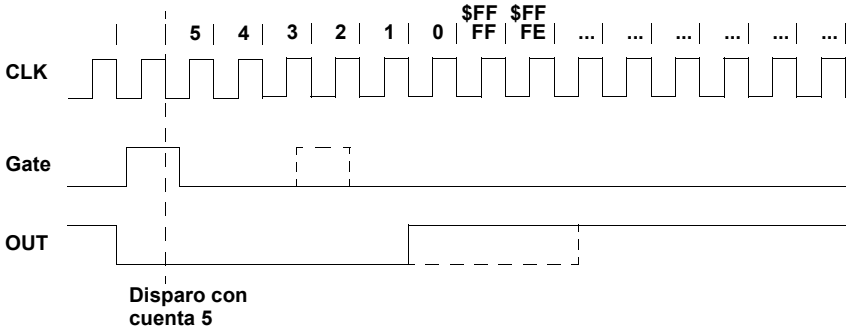
Modo 1:
Ciclo simple redisparable

UNIVERSIDAD DE CANTABRIA

La cuenta se inicia (o reinicia) en el primer flanco de bajada de la señal de reloj con `Gate` en estado alto

- **un cambio en el valor de cuenta inicial no tiene efecto hasta que se produzca un redisparo**

La salida se mantiene a nivel bajo hasta que finaliza la cuenta



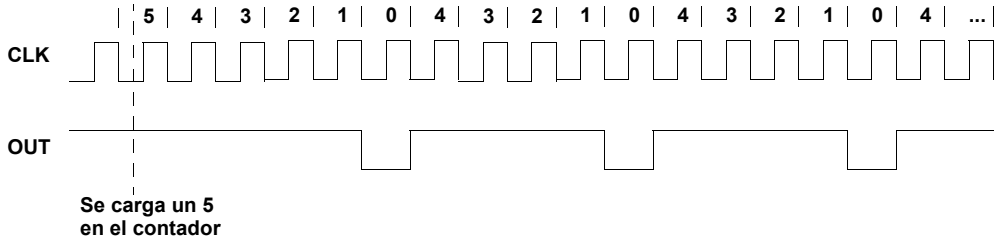
Modo 2:
Interrupción periódica

UNIVERSIDAD DE CANTABRIA

La cuenta se inicia:

- **al establecer el modo**
- **o al escribir un nuevo valor de cuenta inicial**
- **o cada vez que la cuenta llega a 0**

La salida se mantiene a nivel alto mientras la cuenta es distinta de 0 y a nivel bajo mientras la cuenta es 0



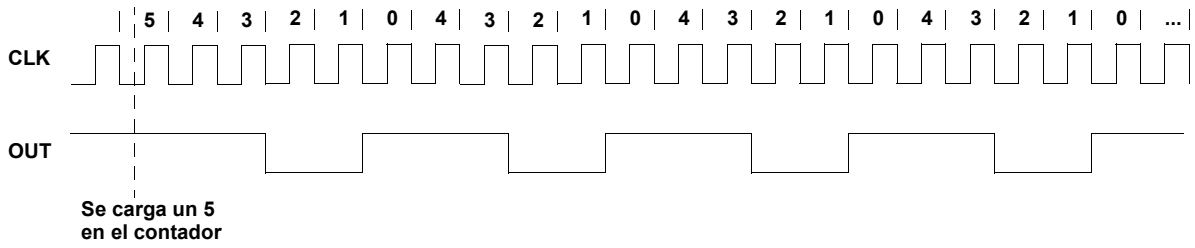
Modo 3:
Generador de onda cuadrada

UNIVERSIDAD DE CANTABRIA

Igual que el modo 2, salvo en el valor de la salida

Si el valor de cuenta inicial es N:

- **la salida permanece a nivel alto durante los `N/2` primeros ciclos (`(N+1)/2` si N es impar)**
- **la salida permanece a nivel bajo durante los `N/2` últimos ciclos (`(N-1)/2` si N es impar)**



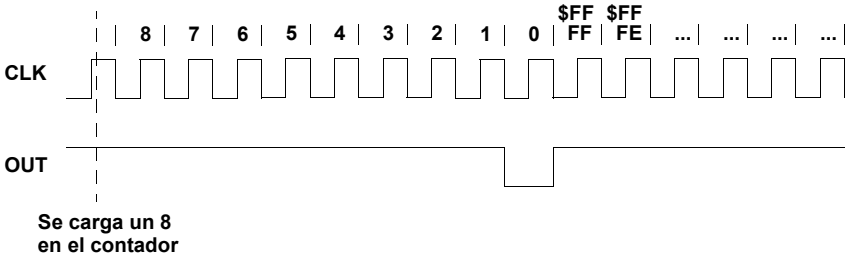
Modo 4:
Interrupción sobre fin de cuenta

UNIVERSIDAD DE CANTABRIA

La cuenta se inicia:

- **al establecer el modo**
- **o al escribir un nuevo valor de cuenta inicial**

La salida se mantiene a nivel alto salvo mientras la cuenta tiene el valor nulo



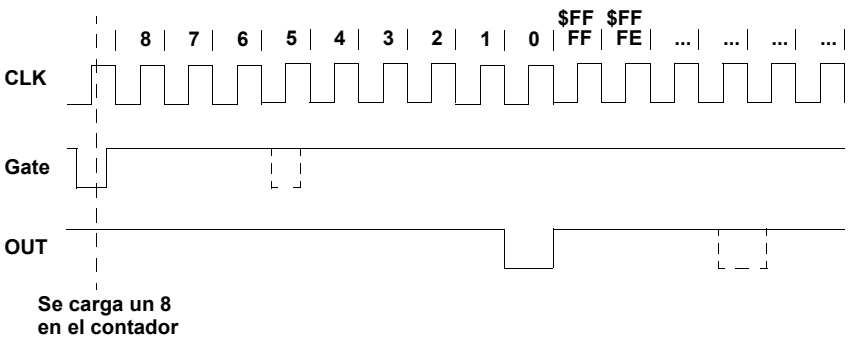
Modo 5:
Interrupción redisparable

UNIVERSIDAD DE CANTABRIA

la cuenta se inicia con cualquier flanco positivo de la señal `Gate`

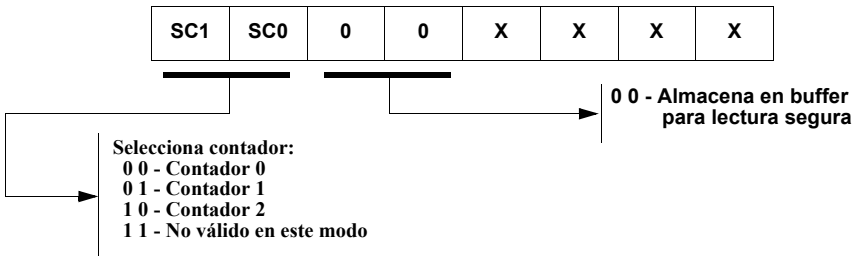
- **es, por tanto, redisparable**

La salida se mantiene a nivel alto salvo mientras la cuenta tiene el valor nulo



PIT: Lectura por almacenamiento previo en el buffer

UNIVERSIDAD DE CANTABRIA



Permite almacenar en un buffer el valor del contador para su lectura segura

- **el contador no detiene su cuenta**

La primera lectura sobre el registro del contador retorna el lsb y la segunda el msb de la cuenta almacenada

PIT: Lectura retrasada

UNIVERSIDAD DE CANTABRIA

Este método permite leer el valor de la cuenta y/o del estado de varios temporizadores en el mismo instante

- **Los bits del registro de control cambian de significado:**

1	1	$\overline{\text{CONT}}$	$\overline{\text{STAT}}$	T2	T1	T0	X

- ***CONT*: 0 => requiere almacenar el valor actual de la cuenta de los temporizadores que posteriormente se indiquen**
- ***STAT*: 0 => requiere almacenar el estado de los temporizadores que posteriormente se indiquen**
- ***T2*: almacena cuenta y/o estado del temporizador 2**
- ***T1*: almacena cuenta y/o estado del temporizador 1**
- ***T0*: almacena cuenta y/o estado del temporizador 0**

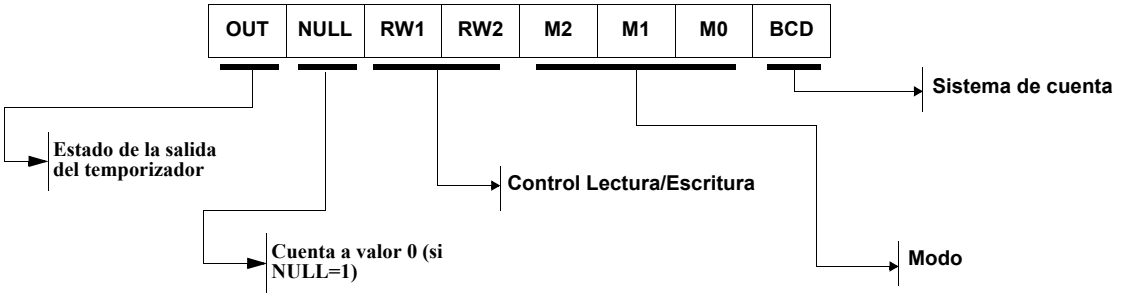
PIT: Lectura retrasada (cont.)

UNIVERSIDAD DE CANTABRIA

Tras configurar el modo de lectura retrasada, las sucesivas lecturas sobre el registro de un temporizador retornarán:

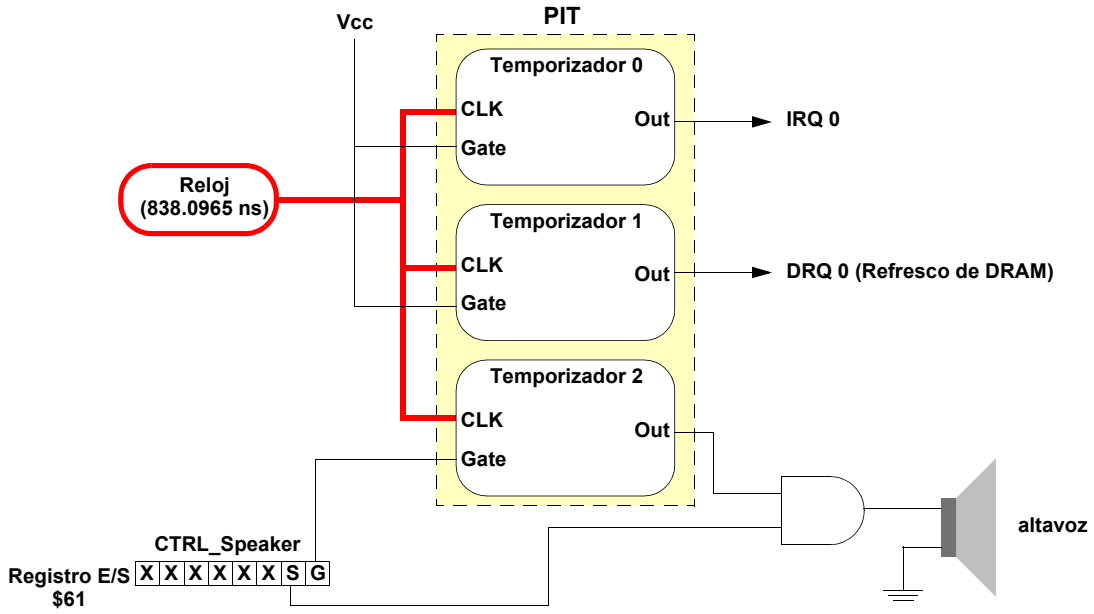
1. **Estado del contador (sólo si `STAT=0`)**
2. **lsb del valor de la cuenta (sólo si `CONT=0`)**
3. **msb del valor de la cuenta (sólo si `CONT=0`)**

El formato correspondiente al estado de un temporizador es:



PIT Conexión en el PC

UNIVERSIDAD DE CANTABRIA



Ejemplo de uso del PIT: Tarea periódica sencilla

UNIVERSIDAD DE CANTABRIA

Utiliza el temporizador 2 (normalmente utilizado por el altavoz) para esperar el periodo de tiempo deseado

```
with MaRTE_OS;
with Basic_Integer_Types; use Basic_Integer_Types;
with IO_Interface; use IO_Interface;

procedure Tarea_Periodica_PIT_Sin_Interrupciones is

   --  Registros del PIT
   Reg_Temp_0 : constant IO_Port := 16#40#;
   Reg_Temp_1 : constant IO_Port := 16#41#;
   Reg_Temp_2 : constant IO_Port := 16#42#;
   Reg_Ctrl   : constant IO_Port := 16#43#;

   --  Registro de control del altavoz
   Reg_Ctrl_Altavoz : constant IO_Port := 16#61#;
```


Ejemplo de uso del PIT: Tarea periódica sencilla (cont.)

UNIVERSIDAD DE CANTABRIA

```
   --  Frecuencia de reloj del PIT
   Frec_CLK_Hz : constant Float := 1_193_180.0;

   --  Periodo
   Periodo : constant Float := 0.020; -- 20 ms

   --  Número de cuentas en 20 ms
   Cuenta_Inicial : constant Unsigned_32 :=
                    Unsigned_32 (Periodo * Frec_CLK_Hz);

begin
   --  Configura temporizador 2 del PIT
   Outb (Reg_Ctrl, 2#10_11_000_0#); -- 10=>temporiz. 2,
      --  11=>lsb y después msb, 000=>modo 0, 0=>binario

   --  Deshabilita el altavoz y habilita la cuenta
   Outb (Reg_Ctrl_Altavoz, 2#000000_0_1#);
```

Ejemplo de uso del PIT: Tarea periódica sencilla

UNIVERSIDAD DE CANTABRIA

```
   loop
     -- Carga cuenta inicial (con lo que comienza cuenta)
     Outb (Reg_Temp_2,
           Unsigned_8 (Cuenta_Inicial and 16#FF#));--lsb
     Outb (Reg_Temp_2,
           Unsigned_8 (Cuenta_Inicial / 2**8)); -- msb

     --  Espera a que acabe la cuenta
     loop
       -- Requiere lect. retrasada del estado del temp. 2
       Outb (Reg_Ctrl, 2#11_1_0_100_0#);
       --  Sale cuando la señal "Out" está a nivel alto
       exit when (Inb (Reg_Temp_2) and 16#80#) /= 0;
     end loop;

     Realiza Tarea Periódica;
   end loop;
end Tarea_Periodica_PIT_Sin_Interrupciones;
```


Reloj de tiempo real (RTC) (MC146818)

UNIVERSIDAD DE CANTABRIA

- resolución de segundos
- puede operar en modo 12 o 24 horas
- contadores codificados en binario o en BCD
- calendario de 100 años (días de la semana, del mes, ...)
- compensaciones de tiempo a fin de año
- cambio de horario verano/invierno
- puede almacenar una alarma diaria y generar interrupción cuando expire

RTC
Registros internos

UNIVERSIDAD DE CANTABRIA

Número registro	Función	Número registro	Función
0	Segundo	7	Día_Mes
1	Segundo_Alarma	8	Mes
2	Minuto	9	Año
3	Minuto_Alarma	10	Registro A
4	Hora	11	Registro B
5	Hora_Alarma	12	Registro C
6	Día_Semana	13	Registro D

RTC
Lectura y cambio de hora

UNIVERSIDAD DE CANTABRIA

Los registros `Segundo`, `Minuto`, `Hora`, `Mes` y `Año`, pueden ser leídos o actualizados desde programa

- **la escritura sólo puede realizarse con actualizaciones inhibidas (bit `SET` del `Registro B` a valor 1)**
- **la lectura debe realizarse fuera del intervalo de actualización que ocurre cada segundo. Hay 3 alternativas:**
  - **Chequeo del bit `UIP` del `Registro A`: cuando vale 0 se dispone de al menos 244μs para realizar la lectura**
  - **Sincronización con la interrupción de fin de ciclo de actualización: tras ella hay 999ms para realizar la lectura segura**
  - **Sincronización con la interrupción periódica: tras ella se dispone de la mitad del periodo de la interrupción**

RTC
Interrupciones y alarma

UNIVERSIDAD DE CANTABRIA

Existen tres fuentes independientes de interrupciones:

- ***Interrupción periódica*** **de frecuencia programable entre 2Hz y 32768KHz**
- ***Interrupción por fin de actualización***
- ***Interrupción por alarma*****: según el valor escrito en los registros `Segundo_Alarma`, `Minuto_Alarma` y `Hora_Alarma`**
  - **si en cada uno de estos registros se escribe un valor válido, la alarma se produce a la hora, minuto y segundo especificados**
  - **si en alguno de ellos se escribe un valor "comodín" (entre $C0 y $FF) la alarma se produce en cada hora y/o minuto y/o segundo**

RTC:
Registro A (lectura/escritura)

UNIVERSIDAD DE CANTABRIA

UIP	DV2	DV1	DV0	RS3	RS2	RS1	RS0

- ***UIP*****: actualización en progreso**
- ***DV2-DV1*****: frecuencia del reloj de entrada**
  - **en el PC ya se encuentra configurada al valor apropiado, por lo que no debemos cambiarlo**
- ***RS3-RS0*****: periodo de la interrupción periódica**
  - **Periodo = 0.030517578125 ms * $2^{RS3\ RS2\ RS1\ RS0\ -\ 1}$**
  - **Por ejemplo, si el valor escrito en RS3-RS0 es 2#1110# (14) el periodo vale:**

$$Periodo = 0.030517578125 * 2^{14\ -\ 1} = 250\ ms$$

RTC: Registro B (lectura/escritura)

UNIVERSIDAD DE CANTABRIA

SET	PIE	AIE	UIE	SQWE	DM	24/12	DSE

- *SET*: 1 => inhibe el ciclo de actualización
- *PIE*: 1 => habilita interrupciones periódicas
- *AIE*: 1 => habilita interrupciones por alarma
- *UIE*: 1 => habilita interrupciones por fin de actualización
- *SQWE*: 1 => habilita onda cuadrada de salida
- *DM*: 1 => contadores en binario, 0 => contadores en BCD
- *24/12*: 1 => modo 24 horas, 0 => modo 12 horas
- *DSE*: 1 => cambio de horario verano/invierno. Adelanto en el último domingo del mes de abril y retraso en el último domingo del mes de octubre

RTC: Registro C (sólo lectura)

UNIVERSIDAD DE CANTABRIA

IRQF	PF	AF	UF	0	0	0	0

- *IRQF*: interrupción pendiente, esto es:
  IRQF = (PF and PIE) or (AF and AIE) or (UF and UIE)
- *PF*: 1 => se ha verificado la condición de interrupción periódica (independientemente del valor de PIE)
- *AF*: 1 => se ha verificado la condición de interrupción por alarma (independientemente del valor de AIE)
- *UF*: 1 => se ha verificado la condición de interrupción por fin de actualización (independientemente del valor de UIE)

Todos los bits de este registro se ponen a 0 después de su lectura

RTC:
Registro D (sólo lectura)

UNIVERSIDAD DE CANTABRIA

VRT	0	0	0	0	0	0	0

- ***VRT*: RAM y tiempo válido. Se pone a cero cuando se detecta un fallo de alimentación.**

RTC
Registros de E/S y línea de IRQ

UNIVERSIDAD DE CANTABRIA

Dirección E/S	Tipo Operación	Función
$70	Escritura	Selecciona el registro interno al que se desea acceder
$71	Lectura/ Escritura	Accede (lee o escribe) el registro interno seleccionado

La interrupción que genera este dispositivo se produce a través de la línea IRQ8

- **tipo de interrupción:**
  `MaRTE_Hardware_Interrupts.RTC_INTERRUPT`

Ejemplo de uso del RTC: Lectura de la hora actual

UNIVERSIDAD DE CANTABRIA

```
with MaRTE_OS;
with Basic_Integer_Types; use Basic_Integer_Types;
with IO_Interface; use IO_Interface;
with Ada.Text_IO; use Ada.Text_IO;
with Ada.Integer_Text_IO; use Ada.Integer_Text_IO;
procedure Lectura_Hora_RTC is
   --  Registros de E/S del RTC
   RTC_Seleccion : constant IO_Port := 16#70#;
   RTC_Reg       : constant IO_Port := 16#71#;
   --  Registros internos del RTC
   RTC_SEG   : constant Unsigned_8 := 16#00#;
   RTC_MIN   : constant Unsigned_8 := 16#02#;
   RTC_HRS   : constant Unsigned_8 := 16#04#;
   RTC_REG_A : constant Unsigned_8 := 16#0a#;
   RTC_REG_B : constant Unsigned_8 := 16#0b#;
   RTC_REG_C : constant Unsigned_8 := 16#0c#;
   RTC_REG_D : constant Unsigned_8 := 16#0d#;
```


Ejemplo de uso del RTC: Lectura de la hora actual (cont.)

UNIVERSIDAD DE CANTABRIA

```
   --  Escribe en pantalla el valor del byte "B"
   --  codificado en código BCD
   procedure Put_BCD (B : Unsigned_8) is
   begin
      Put (Integer (B / 2**4), 1);
      Put (Integer (B and 16#0F#), 1);
   end Put_BCD;


   --  Lee el registro interno del RTC especificado
   --  por "Reg"
   function Lee_Registro_RTC (Reg : Unsigned_8)
                              return Unsigned_8 is
   begin
     Outb (RTC_Seleccion, Reg); -- selecciona registro
     return Inb (RTC_Reg); -- lee registro seleccionado
   end Lee_Registro_RTC;
```

Ejemplo de uso del RTC: Lectura de la hora actual (cont.)

UNIVERSIDAD DE CANTABRIA

```
procedure Lee_Hora_RTC
                  (Hor, Min, Seg : out Unsigned_8) is
begin
   -- espera fin de ciclo de actualización
   loop
      exit when (Lee_Registro_RTC (RTC_REG_A) and
                 16#80#) = 0;
   end loop;

   -- lee registros
   Hor := Lee_Registro_RTC (RTC_HRS);
   Min := Lee_Registro_RTC (RTC_MIN);
   Seg := Lee_Registro_RTC (RTC_SEG);
end Lee_Hora_RTC;
```


Ejemplo de uso del RTC: Lectura de la hora actual (cont.)

UNIVERSIDAD DE CANTABRIA

```
   Hor, Min, seg : Unsigned_8;
   Tecla : Character;

begin
   loop -- Lazo hasta que se pulsa la tecla 's'

      Get_Immediate (Tecla);
      exit when Tecla = 's';

      Lee_Hora_RTC (Hor, Min, Seg);
      Put ("Hora:"); Put_BCD (Hor); New_Line;
      Put ("Min:");  Put_BCD (Min); New_Line;
      Put ("Seg:");  Put_BCD (Seg); New_Line;

   end loop;

end Lectura_Hora_RTC;
```

Temporización utilizando recursos de bajo nivel

UNIVERSIDAD DE CANTABRIA

Servicios de temporización más habituales en tiempo real:

- realización de tareas periódicas
- medida de tiempos

Los sistemas operativos y lenguajes de programación avanzados facilitan el trabajo, proporcionando soporte para:

- tareas (y operación de suspensión de tareas)
- medida de tiempo con alta resolución

En ocasiones no se dispone de SO o lenguaje de programación avanzado

- normal en plataformas pequeñas y baratas (microcontrol.)
- hay que utilizar recursos de temporización de bajo nivel

Implementación de tareas periódicas utilizando el PIT

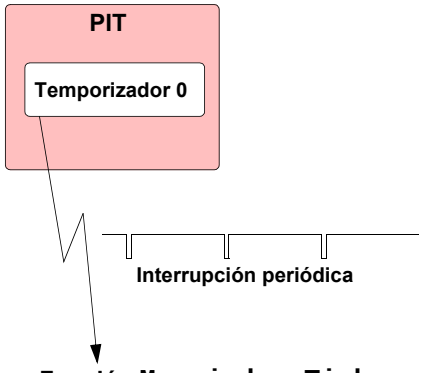
UNIVERSIDAD DE CANTABRIA

Implementación sencilla utilizando el temporizador en modo "generación de interrupción periódica" (tick)

- Se calcula el número de ticks correspondientes al periodo de cada tarea
- La función manejadora de la interrupción del tick:
  - incrementa el contador de ticks asociado a cada tarea
  - cuando se alcanza el número de ticks correspondiente al periodo de una tarea se pone su "flag" a "True"
- El programa principal:
  - realiza un lazo infinito en el que comprueba el valor de los "flags" asociados a las tareas
  - cuando un "flag" es "True" ejecuta la tarea asociada

Implementación de tareas periódicas utilizando el PIT (cont.)

UNIVERSIDAD DE CANTABRIA



Función `Manejador_Tick`

```
Para cada tarea:
  Incrementa Contador_Ticks;
  if Contador_Ticks =
     ticks en un periodo then
     Ejecuta_Tarea := True;
     Contador_Ticks := 0;
  end if;
```

Procedimiento Principal

```
loop
  if Ejecuta_Tarea_1 then
    Ejecuta_Tarea_1 := False;
    Ejecuta el código de la tarea 1;
  end if;

  if Ejecuta_Tarea_2 then
    Ejecuta_Tarea_2 := False;
    Ejecuta el código de la tarea 2;
  end if;

  ...

  if Ejecuta_Tarea_N then
    Ejecuta_Tarea_N := False;
    Ejecuta el código de la tarea N;
  end if;
end loop;
```


Implementación de tareas periódicas utilizando el PIT (cont.)

UNIVERSIDAD DE CANTABRIA

```
with MaRTE_OS;
with IO_Interface, MaRTE_Hardware_Interrupts;
use IO_Interface, MaRTE_Hardware_Interrupts;
with Ada.Text_IO; use Ada.Text_IO
with System;

procedure Tareas_Periodicas_PIT is

   --  Registros del PIT
   Reg_Temp_0 : constant IO_Port := 16#40#;
   Reg_Temp_1 : constant IO_Port := 16#41#;
   Reg_Temp_2 : constant IO_Port := 16#42#;
   Reg_Ctrl   : constant IO_Port := 16#43#;

   --  Periodo de Tick
   Frec_CLK_Hz  : constant Float := 1_193_180.0;
   Periodo_Tick : constant Float := Float (16#FFFF#) /
                                    Frec_CLK_Hz;
```

Implementación de tareas periódicas utilizando el PIT (cont.)

UNIVERSIDAD DE CANTABRIA

```
--  Datos gestión tarea 1
Ejecuta_Tarea_1 : Boolean := False;
pragma Volatile (Ejecuta_Tarea_1);

Periodo_Tarea_1 : constant Float := 1.0;
Ticks_En_Periodo_Tarea_1 : constant Integer :=
  Integer (Periodo_Tarea_1 / Periodo_Tick);
Contador_Tarea_1 : Integer := 0;

--  Datos gestión tarea 2
Ejecuta_Tarea_2 : Boolean := False;
pragma Volatile (Ejecuta_Tarea_2);

Periodo_Tarea_2 : constant Float := 3.0;
Ticks_En_Periodo_Tarea_2 : constant Integer :=
  Integer (Periodo_Tarea_2 / Periodo_Tick);
Contador_Tarea_2 : Integer := 0;
```


Implementación de tareas periódicas utilizando el PIT (cont.)

UNIVERSIDAD DE CANTABRIA

```
--  Manejador de la interrupción del Tick
function Manejador_Tick (Area : in System.Address;
                         Intr : in Hardware_Interrupt)
                        return Handler_Return_Code is
begin
  Contador_Tarea_1 := Contador_Tarea_1 + 1;
  if Contador_Tarea_1 = Ticks_En_Periodo_Tarea_1 then
     Ejecuta_Tarea_1 := True;
     Contador_Tarea_1 := 0;
  end if;
  Contador_Tarea_2 := Contador_Tarea_2 + 1;
  if Contador_Tarea_2 = Ticks_En_Periodo_Tarea_2 then
     Ejecuta_Tarea_2 := True;
     Contador_Tarea_2 := 0;
  end if;
  return POSIX_INTR_HANDLED_NOTIFY;
end Manejador_Tick;
```

Implementación de tareas periódicas utilizando el PIT (cont.)

UNIVERSIDAD DE CANTABRIA

```
begin
   --  Configura temporizador 2 del PIT
   Outb (Reg_Ctrl, 2#00_11_010_0#); -- 00=>temporiz. 0,
      --  11=>lsb y después msb, 010=>modo 2, 0=>binario

   --  Carga cuenta inicial (con lo que empieza a contar)
   Outb (Reg_Temp_0, 16#FF#); -- lsb
   Outb (Reg_Temp_0, 16#FF#); -- msb

   --  Instala manejador y habilita interrupción
   if Associate (TIMER_INTERRUPT,
                 Manejador_Tick'Unrestricted_Access,
                 System.Null_Address, 0) /= 0 then
      Put_Line ("Error Associate");
   end if;
   if Unlock (TIMER_INTERRUPT) /= 0 then
      Put_Line ("Error Unlock");
   end if;
```


Implementación de tareas periódicas utilizando el PIT (cont.)

UNIVERSIDAD DE CANTABRIA

```
   loop

      if Ejecuta_Tarea_1 then
         Ejecuta_Tarea_1 := False;
         Put_Line ("Tarea Periódica 1");
      end if;

      if Ejecuta_Tarea_2 then
         Ejecuta_Tarea_2 := False;
         Put_Line ("Tarea Periódico 2");
      end if;

   end loop;

end Tareas_Periodicas_PIT;
```

Estrategias de medida de tiempos

UNIVERSIDAD DE CANTABRIA

Utilizando el RTC

- resolución de segundo

Utilizando un contador del PIT

- resolución de 838.0965 ns
- máxima duración del intervalo a medir ≅ 55 ms

Utilizando la estrategia de tick periódico

- llevar un contador con el número de ticks desde el arranque del programa
- hora actual := contador * periodo tick
- resolución del periodo del tick

Medida de tiempo utilizando el contador 2 del PIT

UNIVERSIDAD DE CANTABRIA

```
with MaRTE_OS;
with Basic_Integer_Types; use Basic_Integer_Types;
with IO_Interface; use IO_Interface;
with Ada.Text_IO; use Ada.Text_IO;

procedure Mide_Tiempo_Con_PIT is

   --  Registros del PIT
   Reg_Temp_0 : constant IO_Port := 16#40#;
   Reg_Temp_1 : constant IO_Port := 16#41#;
   Reg_Temp_2 : constant IO_Port := 16#42#;
   Reg_Ctrl   : constant IO_Port := 16#43#;

   --  Registro de habilitación del temp. 2 y altavoz
   Reg_Ctrl_Altavoz : constant IO_Port := 16#61#;

   --  Frecuencia de reloj del PIT
   Frec_CLK_Hz : constant Float := 1_193_180.0;
```

Medida de tiempo utilizando el contador 2 del PIT (cont.)

UNIVERSIDAD DE CANTABRIA

```
   Estado, Cuenta_Lo, Cuenta_Hi : Unsigned_8;
   Cuenta : Unsigned_16;
   Retraso : Float;
begin
   --  Configura temporizador 2 del PCT
   --  temp 2, lsb y después msb, modo 0, binario
   Outb (Reg_Ctrl, 2#10_11_000_0#);
   --  Carga cuenta inicial (máximo valor: 16#FFFF#)
   Outb (Reg_Temp_2, 16#FF#); -- lsb
   Outb (Reg_Temp_2, 16#FF#); -- msb
   --  Deshabilita el altavoz y habilita la cuenta
   Outb (Reg_Ctrl_Altavoz, 2#000000_0_1#);
   --  Trozo de código a medir su duración
   ...;
   --  Configura lectura retrasada de estado y cuenta
   Outb (Reg_Ctrl, 2#11_0_0_100_0#);
```


Medida de tiempo utilizando el contador 2 del PIT (cont.)

UNIVERSIDAD DE CANTABRIA

```
   --  Lee estado y valor de la cuenta
   Estado := Inb (Reg_Temp_2);
   Cuenta_Lo := Inb (Reg_Temp_2);
   Cuenta_Hi := Inb (Reg_Temp_2);
   Cuenta := Unsigned_16 (Cuenta_Hi) * 2**8 +
             Unsigned_16 (Cuenta_Lo);

   if (Estado and  16#80#) /= 0 then
      --  Ha llegado al final de la cuenta
      Put_Line ("Tiempo demasiado largo");
   else
      Retraso := Float (16#FFFF# - Cuenta) / Frec_CLK_Hz;
      Put_Line ("Retraso:" & Float'Image (Retraso));
   end if;

end Mide_Tiempo_Con_PIT;
```