Timers - Counters

Andrés Benacerraf
Nahuel González

Introducción

El LPC1769 cuenta con **4 timers/counters de 32 bit incrementales** (cuenta ascendente).

Recordemos que un timer es un periférico que consta de un registro que va contando pulsos de una fuente de clock definida y por lo tanto aumentando su valor (en forma creciente en este caso, podría ser decreciente como lo es el Systick Timer –systick-).

Esta fuente de clock de la que hablamos, **puede ser el peripherial clock para el uso como timer o un clock externo** ingresado por las entradas de capture para el uso como contador (lo veremos más adelante).

Una obviedad pero puede ser la duda de algunos: el incremento del valor del timer y su funcionamiento en general es por hardware (excepto cuando debo configurarlo y cuando ocurre una interrupción), es decir, mi timer funciona, cuenta pulsos, incrementa su valor, ocurren o no eventos asociados, ***pero yo paralelamente en mi programa puedo estar haciendo otra cosa.***

Ahora bien, **estos timers** no son tan "comunes" ya que a diferencia de la mayoría de los timers en otras arquitecturas de microcontroladores (como 8051), **no generan ningún evento/interrupción cuando desbordan o se produce "overflow"** (con desborde/overflow nos referimos a cuando el timer llegó a la máxima cuenta de 0xFFFFFFFF y al próximo incremento vuelve a 0x00000000).

En nuestro caso, los timers simplemente cuentan, desbordan, y así sucesivamente mientras los mantengamos activados.

Por lo tanto, surgen algunas posibles preguntas que son:

- *¿Qué funcionalidades me ofrecen?*
- *¿Como los utilizo para generar interrupciones y otros eventos?*

Como respuesta, entran en juego **los registros de MATCH y CAPTURE**.

Se presentan como un juego de registros asociados a los timers que, dependiendo de como los configure, me pueden generar:

- *Interrupciones*
- *Modificar el valor lógico de ciertos pines del micro*
- *Incrementar la cuenta del timer/contador*
- *Resetearlo*
- *Detenerlo*
- *Capturar su valor en un momento dado*
- *Generar llamados al DMA (direct memory Access – acceso directo a memoria).*

Nota: El vínculo entre el DMA y timers/counters no será discutido en este documento.

Uso práctico

Ya que conocemos las herramientas que nos brindan los timers, seguramente aparecerán algunas dudas respecto a su uso práctico:

- ¿Uso el timer?
- ¿Uso el MATCH?
- ¿Uso el CAPTURE?
- ¿Uso el timer y el MATCH?
- ¿Uso el timer y el CAPTURE?
- ¿Uso el CAPTURE y el MATCH?
- ¿Uso el timer, el MATCH y el CAPTURE?

Con esta serie de preguntas, buscamos que les sirvan de disparador para poder evaluar que combinación y registros utilizar de acuerdo a lo que necesite realizar (aplicación).

Primero observemos que de las preguntas anteriores hay algunas que no tienen sentido, por lo que les proponemos analizarlas y descartarlas

Dado que el conjunto de registros de match y capture se relacionan con el timer a través de valores de cuenta, generación de interrupciones, etc. debemos concluir:

- **No puedo usar solamente el match.**
- **No puedo usar solamente el capture.**
- **No puedo usar solamente el capture y el match.**

Ahora que descartamos algunas de ellas, podemos seguir con las pendientes:

- ¿Uso el timer?
- ¿Uso el timer y el MATCH?
- ¿Uso el timer y el CAPTURE?
- ¿Uso el timer, el MATCH y el CAPTURE?

Evaluemos la primera:

¿Uso el timer?
Si bien puedo prender el timer solamente y hacer que incremente su valor, me sirve para muy poco ya que sin los registros de match y/o capture no puedo generar ningún evento (recordemos que al desbordar no genera ningún evento).

Analicemos el siguiente caso:

Podemos por ejemplo, ¿prender el timer, hacer que cuente los pulsos de PCLK y leer a TC (registro que tiene el valor actual del timer) en ciertos momentos de mi programa con el objetivo de supervisar el tiempo que tarda una función en ejecutarse?

En código:

```
int mi_funcion (void)
{
        int cuenta;
        T0TCR &= ~(0x10); // Activo el timer (Suponemos que estaba en reset anteriormente)
        ...
        ...
        /*
        cuerpo de mi función (Se supone que es una función extensa que demora cientos de ciclos de
        reloj en ejecutarse, es decir, la demora en realizarse no es afectada en gran parte por las
        demoras que me incluye cargar el valor de cuenta)
        */
        ...
        ...
        cuenta = TxTC;
        T0TCR |= 0x10; // Pongo en reset el timer.

        return cuenta;
}
```

Como respuesta a la pregunta anterior podemos decir que realizarlo es posible y puedo hacer otras tareas que involucren estrategias de este estilo, entonces:

¿Existe alguna limitación en el ejemplo anterior?

Lo que sucede es que estos métodos pierden precisión con tareas cortas y tengo el grave problema que **el timer desborda y no me avisa.**
Por lo que las temporizaciones a medir deberían medir como máximo el mayor valor de tiempo que puede contar el timer en un recorrido.

Finalmente, nos quedan 3 preguntas con aplicación práctica:

¿Uso el timer y el MATCH?

¡Si! Cuando el timer va contando y llega al valor cargado en el registro del match puedo:

- Generar una interrupción
- Cambiar el estado de pines (conmuntar, poner a 1, poner a 0 o dejarlo como está)
- Resetear el timer
- Incluso no hacer nada

¿Uso el timer y el CAPTURE?

¡Totalmente! Cuando aparece un evento definido por nosotros en las entradas de capture (flanco ascendente, flanco descendente) podemos:

- Generar una interrupción
- Capturar el valor del timer en ese momento
- Incrementar el contador.

¿Uso el timer, el MATCH y el CAPTURE?

Si! Es muy útil. Un ejemplo:

Supongamos que al timer lo queremos utilizar en modo contador, para contar los pulsos que llegan por un pin de captura que definimos y que provoque una interrupción cuando llega a un valor dado.

Con esta configuración, incrementamos el timer con cada pulso en la entrada de captura y e interrumpimos cuando **cuenta del contador sea igual al valor de match**.

Sería un contador de eventos externos enteramente por HW.

Estos son solo algunos ejemplos, como vemos el uso de timers nos permite evaluar su uso en muchas aplicaciones. Recordemos que salvo la generación de interrupciones, ***todos los demás eventos son por HW.*** Es decir, funcionan en mi programa mientras me ocupo de hacer otras cosas.

Funcionamiento Timer-Match

Los registros de MATCH me permiten generar algún evento cuando el timer/contador llegó al mismo valor que el registro MRn (registro donde cargo el valor al cual quiero generar el evento).

Como tanto el registro de cuenta del timer y el valor del match son de 32 bits, en el match puedo cargar valores de 0 a 0xFFFFFFFF.

Para cada timer, no tengo uno, sino 4 registros de match que los puedo utilizar **SIMULTÁNEAMENTE**.
Esto me permite generar *varios* eventos que pueden ser **SIMULTÁNEOS** para distintos valores de cuenta del timer.

Los eventos que puedo realizar con cada match son:

Generar una interrupción del timer asociado: dado un evento de match, puedo llamar a *TIMERn_IRQHandler* para una tarea específica.

Y/O

Generar el reset del timer asociado: dado un evento de match, puedo resetear el valor de cuenta del timer en caso de no querer que siga contando hasta 0xFFFF FFFF y desborde por si solo.

Y/O

Manejar un pin de salida: con esta función, podemos poner a 1, poner a 0, variar de estado un pin determinado o dejarlo sin variar.

Para lograr esto, tengo que configurar los registros de match (observar más abajo en la lista de registros) y los registros PINSEL (la función de "salida manejada por match" es una función dedicada dentro de las 4 posibles de cada pin físico).
No puede ser cualquier pin disponible en el micro, son solo ciertos pines por cada timer y cada match.

Por ejemplo, el P1.29 puedo configurarlo como MAT**0.1** (fijarse en la tabla de PINSEL3) al seleccionar la función 3. Esto quiere decir que dicho pin puede ser manejado por match **1** del timer **0.**

Y/O

No hacer nada.

¡TODO ESTO POR CADA MATCH!

¡HAY 4 MATCH POR CADA TIMER!

Funcionamiento Timer-Capture

Los registros de captura me ofrecen la posibilidad de generar acciones dado un evento en alguna de las entradas de captura.

Es decir, hay dos entradas de captura por cada timer a las cuales les puedo definir un evento dado (flanco ascendente, flanco descendente o ambos) y también generar una interrupción.

Esto lo genero modificando grupos de 3 bits del registro TnCCR.

De ocurrir este evento en la entrada de captura, puedo:

- *Generar una interrupción del timer asociado*
- *Cargar el valor actual de cuenta del timer en alguno de los registros de captura (CR0 y CR1).*

Ahora bien, si en vez de hacer estas dos acciones previas quiero utilizar la entrada de captura para incrementar la cuenta del timer configurado como contador, debo poner el grupo de 3 bits anteriormente mencionado en "000" (Recomendado en el UM10360).

Después tengo que configurar el registro TnCTCR para modo "contador" y en ese modo, decirle por cual de las entradas de captura van a ingresar los pulsos y con qué tipo de flanco/s.

Descripción de registros

En esta sección siguiente veremos los registros que utilizaremos para configurar los timers, match y capture.

Encontrarán comentarios generales sobre cada registro, pero no se hará una descripción puntual de los bits de cada uno. La idea es que tengan este material de introducción y luego les proponemos que lean a nuestro querido UM10360.

Registros del timer

PCONP (0x400F C0C4)

Es el registro encargado de habilitar o deshabilitar el clock de ciertos periféricos. Con esto logramos que, al no poder operar dinámicamente, consuman muy poca energía.

En particular, para habilitar el clock de los timers 0-3 hay que poner a 1 los bits 1, 2, 22, 23 respectivamente.

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
13	PCCAN1	CAN Controller 1 power/clock control bit.	0
14	PCCAN2	CAN Controller 2 power/clock control bit.	0
15	PCGPIO	Power/clock control bit for IOCON, GPIO, and GPIO interrupts.	1
16	PCRIT	Repetitive Interrupt Timer power/clock control bit.	0

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

Bit	Symbol	Description	Reset value
17	PCMCPWM	Motor Control PWM	0
18	PCQEI	Quadrature Encoder Interface power/clock control bit.	0
19	PCI2C1	The I^2C1 interface power/clock control bit.	1
20	-	Reserved.	NA
21	PCSSP0	The SSP0 interface power/clock control bit.	1
22	PCTIM2	Timer 2 power/clock control bit.	0
23	PCTIM3	Timer 3 power/clock control bit.	0
24	PCUART2	UART 2 power/clock control bit.	0
25	PCUART3	UART 3 power/clock control bit.	0
26	PCI2C2	I^2C interface 2 power/clock control bit.	1
27	PCI2S	I^2S interface power/clock control bit.	0
28	-	Reserved.	NA
29	PCGPDMA	GPDMA function power/clock control bit.	0
30	PCENET	Ethernet block power/clock control bit.	0
31	PCUSB	USB interface power/clock control bit.	0

PCLKSEL0 (0x400F C1A8)

Con este registro lo que hacemos es seleccionar el peripherial clock (que queremos para nuestro timer).

Para cada timer, tenemos 4 PCLK posibles:
Como el PCLK es un derivado del CCLK (recordemos que el CCLK es el system clock, es decir, la salida del PLL. Para Info2, CCLK = 100 Mhz) podemos hacer que sea: CCLK ó CCLK/2 ó CCLK/4 ó CCLK/8.

Ojo, que quede claro, **no hay un solo PCLK que va a todos los periféricos,** sino que cada periférico tiene su propio PCLK.

Lo que si hay que decir es que todos los PCLK, si bien pueden tener distinta frecuencia, son sincrónicos.

Para elegir PCLK = CCLK, escribir "01" en los bits 3:2 (timer0), 5:4(timer1), 15:14(timer2), 17:16(timer3).

T[0/1/2/3]IR - (0x4000 4000, 0x4000 8000, 0x4009 0000, 0x4009 4000)

Registros donde se encuentran los flags de las fuentes de interrupción para un timer dado.

Siempre que alguno de estos flags esté **activado (en alto)**, se generará una interrupción. Los flags se limpian escribiendo un 1 en cada uno.

Recordemos, por cada timer, hay una sola función de interrupción (TIMER3_IRQHandler para el timer3), **pero muchas fuentes pueden provocarla**.

Estas fuentes pueden ser alguno o varios de los 4 registros de match y/o una o ambas entradas de captura.

Bit	Symbol	Description	Reset Value
0	MR0 Interrupt	Interrupt flag for match channel 0.	0
1	MR1 Interrupt	Interrupt flag for match channel 1.	0
2	MR2 Interrupt	Interrupt flag for match channel 2.	0
3	MR3 Interrupt	Interrupt flag for match channel 3.	0
4	CR0 Interrupt	Interrupt flag for capture channel 0 event.	0
5	CR1 Interrupt	Interrupt flag for capture channel 1 event.	0
31:6	-	Reserved	-

Nota: es sumamente importante detectar cual fue la fuente de interrupción. Al ingresar a la rutina de atención debo ver cuales de los flags están en 1 para hacer las funciones correspondientes de cada uno.

Para profundizar: vean el ejemplo hecho en clase (subido al aula virtual).

T[0/1/2/3]CTCR - (0x4000 4070, 0x4000 8070, 0x4009 0070, 0x4009 4070)

En este registro lo que hago es configurar si el timer/contador estará en modo TIMER o modo CONTADOR, y en caso de que lo configure como contador, configuro que pin de captura utilizará para contar los pulsos y con qué tipo de flanco/s.

Table 428. Count Control Register (T[0/1/2/3]CTCR - addresses 0x4000 4070, 0x4000 8070, 0x4009 0070, 0x4009 4070) bit description

Bit	Symbol	Value	Description	Reset Value
1:0	Counter/ Timer Mode		This field selects which rising PCLK edges can increment the Timer's Prescale Counter (PC), or clear the PC and increment the Timer Counter (TC).	00
		00	Timer Mode: the TC is incremented when the Prescale Counter matches the Prescale Register. The Prescale Counter is incremented on every rising PCLK edge.	
		01	Counter Mode: TC is incremented on rising edges on the CAP input selected by bits 3:2.	
		10	Counter Mode: TC is incremented on falling edges on the CAP input selected by bits 3:2.	
		11	Counter Mode: TC is incremented on both edges on the CAP input selected by bits 3:2.	
3:2	Count Input Select		When bits 1:0 in this register are not 00, these bits select which CAP pin is sampled for clocking.	00
		00	CAPn.0 for TIMERn	
		01	CAPn.1 for TIMERn	
		10	Reserved	
		11	Reserved	
			Note: If Counter mode is selected for a particular CAPn input in the TnCTCR, the 3 bits for that input in the Capture Control Register (TnCCR) must be programmed as 000. However, capture and/or interrupt can be selected for the other 3 CAPn inputs in the same timer.	
31:4	-	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

Importante: La frecuencia máxima de las transiciones en las entradas de CAP debe ser mayor a ½ de PCLK. .

Pueden profundizar con la lectura de UM10360

T[0/1/2/3]CR - (0x4000 4004, 0x4000 8004, 0x4009 0004, 0x4009 4004)

Registro utilizado para habilitar/deshabilitar la cuenta del timer y resetearlo. Para habilitar la cuenta y quitarlo del estado de reset igualar este registro a 0x01.

T0PR - T3PR, (0x4000 400C, 0x4000 800C, 0x4009 000C, 0x4009 400C)

Registro del prescaler. El prescaler es un divisor de clock que está entre PCLK y la entrada de conteo del timer. Es decir, el PCLK tiene que pasar por el prescaler antes de llegar al timer. Si hago que el prescaler divida por 1, el timer contará directamente el clock PCLK.
El valor en TnPR + 1 será el divisor que afectará a PCLK. Por lo tanto, si no se quiere afectar a PCLK, igualar este registro a 0x00.

T0PC - T3PC (0x4000 4010, 0x4000 8010, 0x4009 0010, 0x4009 4010)

Valor de cuenta del prescaler. Cada vez que este registro desborda, la cuenta del timer se incrementa en 1.

Registros de MATCH

T[0/1/2/3]MCR - (0x4000 4014, 0x4000 8014, 0x4009 0014, 0x4009 4014)

Con este registro lo que hago es controlar 3 de las posibles acciones que puede realizar el match en caso de darse el evento.

Table 429. Match Control Register (T[0/1/2/3]MCR - addresses 0x4000 4014, 0x4000 8014, 0x4009 0014, 0x4009 4014) bit description

Bit	Symbol	Value	Description	Reset Value
0	MR0I	1	Interrupt on MR0: an interrupt is generated when MR0 matches the value in the TC.	0
		0	This interrupt is disabled	
1	MR0R	1	Reset on MR0: the TC will be reset if MR0 matches it.	0
		0	Feature disabled.	
2	MR0S	1	Stop on MR0: the TC and PC will be stopped and TCR[0] will be set to 0 if MR0 matches the TC.	0
		0	Feature disabled.	

Las acciones disponibles son:
- Resetear el timer
- Detenerlo
- Generar una interrupción del timer asociado.

Recordemos que podemos realizar cualquier combinación simultánea de estas tres funciones.

Ejemplo:

¿Cómo se configura el timer usar match 0 y 1 para hacer dos temporizaciones distintas e interrumpir?

- Habilito clock para Timer 0
- Elijo fuente de clock para timers
- Configuro los tiempos de match 0 y 1
- Configuro para que match 0 interrumpa y match 1 interrumpa y resetee el timer
- Habilito las interrupciones de timer 0
- Reseteo el contador del timer
- Arranco el timer

MR[0/1/2/3]

En estos 4 registros cargo los valores de cuenta del timer a los cuales quiero que se produzcan los eventos de match.

Generic Name	Description	Access	Reset Value[1]	TIMERn Register/ Name & Address
MR0	Match Register 0. MR0 can be enabled through the MCR to reset the TC, stop both the TC and PC, and/or generate an interrupt every time MR0 matches the TC.	R/W	0	T0MR0 - 0x4000 4018 T1MR0 - 0x4000 8018 T2MR0 - 0x4009 0018 T3MR0 - 0x4009 4018
MR1	Match Register 1. See MR0 description.	R/W	0	T0MR1 - 0x4000 401C T1MR1 - 0x4000 801C T2MR1 - 0x4009 001C T3MR1 - 0x4009 401C
MR2	Match Register 2. See MR0 description.	R/W	0	T0MR2 - 0x4000 4020 T1MR2 - 0x4000 8020 T2MR2 - 0x4009 0020 T3MR2 - 0x4009 4020
MR3	Match Register 3. See MR0 description.	R/W	0	T0MR3 - 0x4000 4024 T1MR3 - 0x4000 8024 T2MR3 - 0x4009 0024 T3MR3 - 0x4009 4024
CCR	Capture Control Register. The CCR controls which edges of the capture inputs are used to load the Capture Registers and whether or not an interrupt is generated when a capture takes place.	R/W	0	T0CCR - 0x4000 4028 T1CCR - 0x4000 8028 T2CCR - 0x4009 0028 T3CCR - 0x4009 4028

T[0/1/2/3]EMR - (0x4000 403C, 0x4000 803C, 0x4009 003C, 0x4009 403C)

Es el registro que utilizo para controlar si el match cambia de estado, pone a 1, pone a 0 o deja invariante un pin asociado con funcion especial MAT. Leer detalle de bits en UM10360.

Registros de CAPTURE

T[0/1/2/3]CCR - (0x4000 4028, 0x4000 8028, 0x4009 0028, 0x4009 4028)

Sirve para controlar los eventos de capture que ocurren en las entradas configuradas como CAP.

Para cada entrada puedo capturar el valor del timer (y guardarlo en CR0 o CR1) en flanco ascendente/descendente y/o generar una interrupción.

Table 430. Capture Control Register (T[0/1/2/3]CCR - addresses 0x4000 4028, 0x4000 8020, 0x4009 0028, 0x4009 4028) bit description

Bit	Symbol	Value	Description	Reset Value
0	CAP0RE	1	Capture on CAPn.0 rising edge: a sequence of 0 then 1 on CAPn.0 will cause CR0 to be loaded with the contents of TC.	0
		0	This feature is disabled.	
1	CAP0FE	1	Capture on CAPn.0 falling edge: a sequence of 1 then 0 on CAPn.0 will cause CR0 to be loaded with the contents of TC.	0
		0	This feature is disabled.	
2	CAP0I	1	Interrupt on CAPn.0 event: a CR0 load due to a CAPn.0 event will generate an interrupt.	0
		0	This feature is disabled.	
3	CAP1RE	1	Capture on CAPn.1 rising edge: a sequence of 0 then 1 on CAPn.1 will cause CR1 to be loaded with the contents of TC.	0
		0	This feature is disabled.	
4	CAP1FE	1	Capture on CAPn.1 falling edge: a sequence of 1 then 0 on CAPn.1 will cause CR1 to be loaded with the contents of TC.	0
		0	This feature is disabled.	
5	CAP1I	1	Interrupt on CAPn.1 event: a CR1 load due to a CAPn.1 event will generate an interrupt.	0
		0	This feature is disabled.	
31:6	-		Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

Table 427. Timer Control Register (TCR, TIMERn: TnTCR - addresses 0x4000 4004, 0x4000 8004, 0x4009 0004, 0x4009 4004) bit description

Bit	Symbol	Description	Reset Value
0	Counter Enable	When one, the Timer Counter and Prescale Counter are enabled for counting. When 1, the counters are disabled.	0
1	Counter Reset	When one, the Timer Counter and the Prescale Counter are synchronously reset on the next positive edge of PCLK. The counters remain reset until TCR[1] is returned to zero.	0
31:2	-	Reserved, user software should not write ones to reserved bits. The value read from a reserved bit is not defined.	NA

CR0 - CR1

Dispongo de este conjunto de registros por cada timer. Es aquí donde guardo las capturas realizadas.

Ejemplo:

¿Cómo se configura el timer para funcionar como captura y generar una interrupción?

- Elijo con PINSEL la funcionalidad de CAP para el pin que voy a usar
- Habilito clock para Timer 0
- Elijo fuente de clock para timers
- Configuro el Timer/Counter como timer y para capturar en flanco descendente de CAP0.0 (P1.26)
- Habilito la interrupción de captura del timer 0
- Habilitor las interrupciones de timer 0
- Reseteo el contador del timer
- Arranco el timer