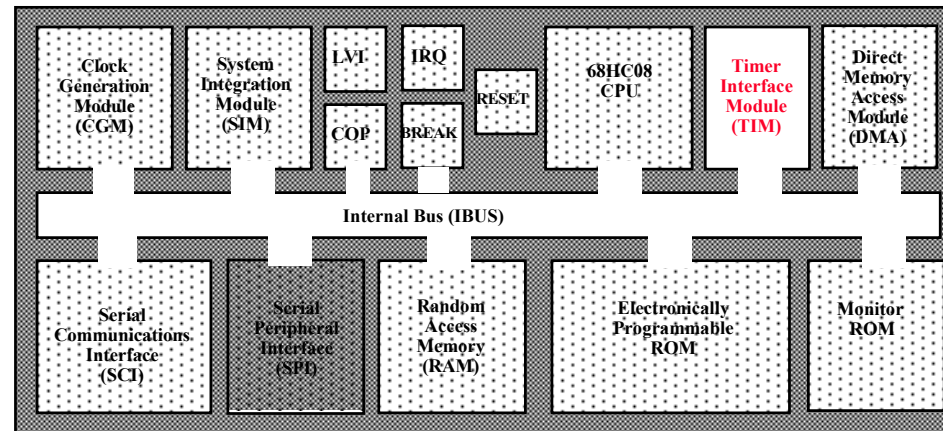


TIMER INTERFACE MODULE (TIM)

TIMER INTERFACE MODULE



Cuatro canales programables

- Input captures
 - x flanco ascendente, descendente, o cualquier flanco de disparo
- Output compares
 - Set, clear, o “toggle action” como nivel de salida
- Pulse width modulation (PWM) (Modulación por Ancho de Pulso)
 - Generación de señal “Buffereada o no Buffereada”

Entrada de Clock programable

- Clock del sistema con “prescaler” de 7 pasos.
Modo de operación “Free-running” o “Modulo up-count”
Acción “toggle” sobre cualquier pin de canal en “Overflow”
Stop y reset del TIM Counter !!!!!

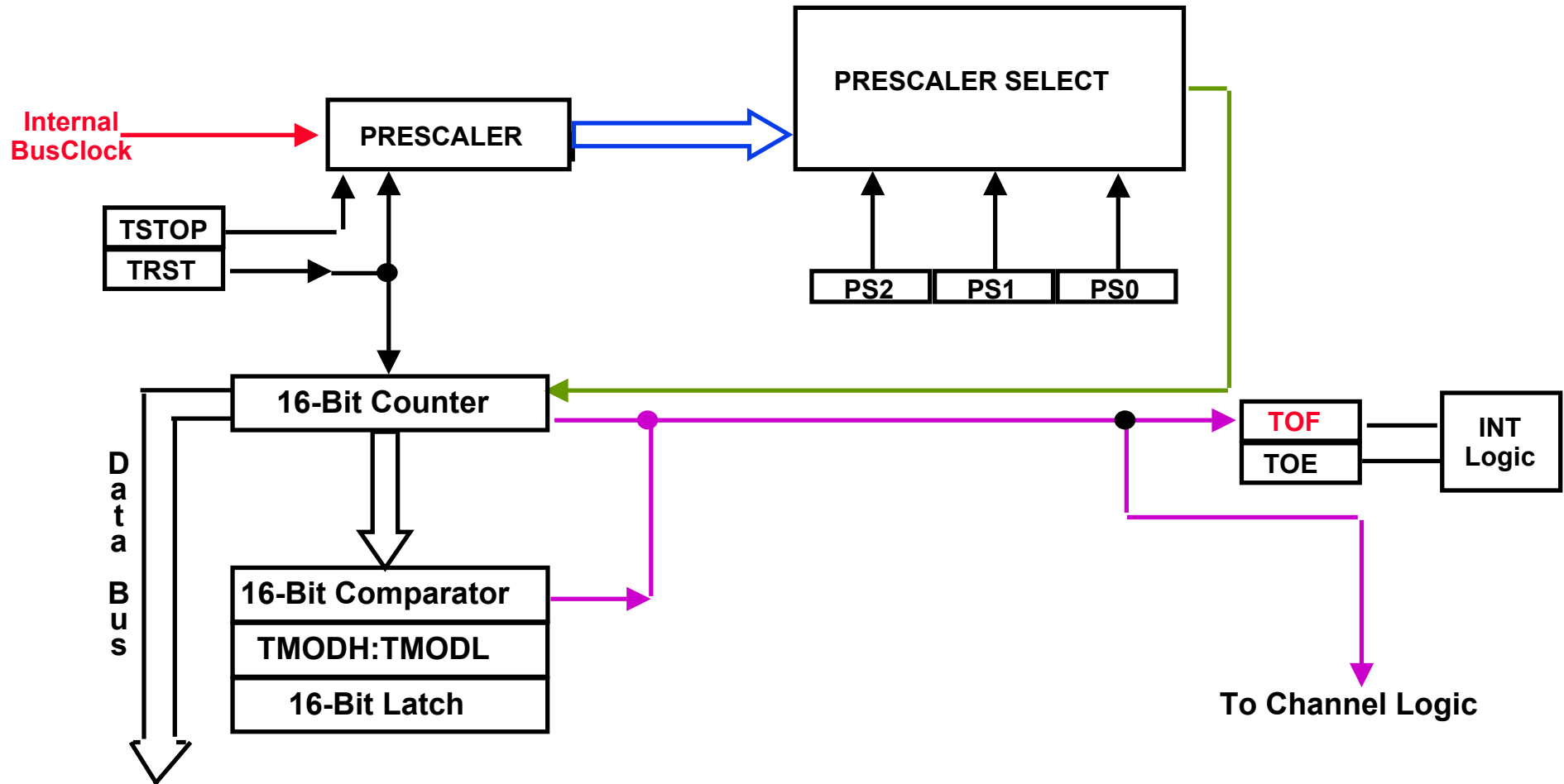
Curso de Microcontroladores

Familia HC908 Flash de Motorola

Parte II

ING. DANIEL DI LELLA DDFAE For Motorola Products

Diagrama en Bloques del Clock de referencia del TIM



El módulo TIM como timer simple solamente.....

Como se puede observar en el cuadro anterior, el módulo de timer del HC908, toma como referencia para su temporización el Clock interno del Bus (FBUS), que obviamente esta relacionado con la frecuencia del Xtal externo ($F_{xtal} / 4$) o bien con la frecuencia del oscilador interno en el caso que el dispositivo tenga la opción de oscilador interno o PLL.

El módulo posee un “prescaler” que divide “N” veces la frecuencia de referencia que entra al mismo, para de esta forma obtener mayor flexibilidad en los rangos de demoras a obtener. Este prescaler puede ser programado por medio de los bits **PS2, PS1, PS0** en el **Timer Status and Control Register (TSC)**.

Luego que el clock há sido dividido en el prescaler, ingresa a un contador de 16 bits de longitud. Este contador es del tipo “free-running” (de cuenta libre) con rango desde \$0000 a \$FFFF. Existe un comparador de 16 bits vinculado al contador y a un registro, también de 16 bits, denominado “**TMOD**” (Timer **M**odulo register) que está dividido en parte “alta” (**TMODH**) y parte “baja” (**TMODL**).

Durante el funcionamiento del timer, el contador es comparado permanentemente con el registro **TMOD** (**TMODH**, **TMODL**), por medio del comparador asociado a ellos, cuando la cuenta del mismo coincide con el valor almacenado en dicho registro, se produce “desborde” del timer o “Timer Overflow”, provocando las siguientes acciones:

- Se genera un Señal de “Timer Overflow Flag” (TOF), que nos indicará la condición de overflow del contador/comparador.
- Se genera un pedido de interrupción (Timer Overflow Interrupt) si esta se encuentra habilitada y se atiende la misma saltando a la dirección indicada por el vector respectivo (TIMx Overflow Vector).

El módulo TIM como timer simple solamente..... continua.

- Se produce un RESET automático del contador “Free – Running” forzándolo a \$0000.

Esta configuración permite operar el timer en modo “Free – Running”, o en modo “Modulo Up Counter”, según el valor del registro TMOD.

Si el registro TMOD (TMODH y TMODL) tiene un valor igual a \$FFFF, el timer funcionará en modo free-running, ya que al llegar a \$FFFF, recién allí, el comparador emitirá una señal de RESET del contador y de disparo del flag TOF y de la interrupción, si estuviera habilitada.

Si el registro TMOD, tiene un valor menor a \$FFFF, entonces el timer funcionará como “Módulo Up Counter”, ya que al llegar al valor establecido en TMOD, provocará el mismo efecto que en el modo “free – running”, lo que garantiza la generación de una temporización (delay) flexible y programable, a diferencia del típico modo “free – running” del timer de los HC705.

Timer Clock Reference

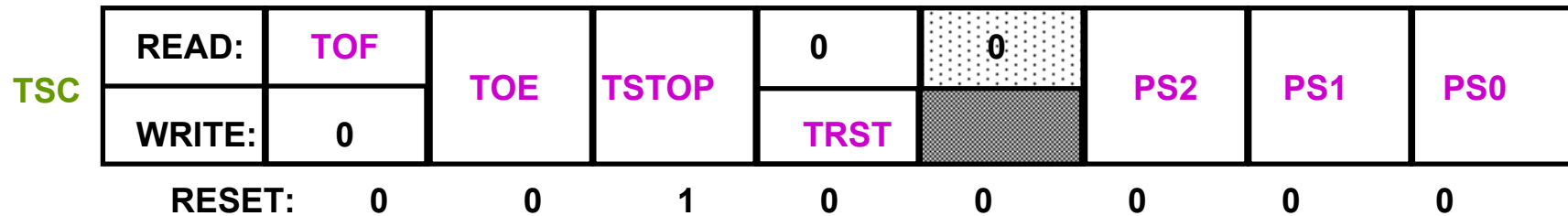
Los 2 canales de timer reciben sus clock reference desde:

- Free running counter
- Modulo up counter

Consisten de :

- Timer counter “ free running ” de solo lectura de 16 bits
- Modulo Register de 16 bit de lectura/escritura por software
- Comparador de 16 bits (timer counter vs. modulo register)
 - Cuando el contador concuerda con el modulo register..
 - Se setea el Timer Overflow Flag (TOF)
 - Resetea el contador a \$0000
 - Los contadores comienzan las cuentas nuevamente !!!!

Timer Status and Control Register (TSC)



Timer Status and Control Register (TSC)

- Clock select and prescaler bits (PS2-PS0)

- Timer Overflow Flag (TOF)

Seteado cuando el timer counter de 16 bits se resetea a \$0000

Limpiado por lectura del TSC y luego escritura de un "0" en el TOF

- Si ocurre un "overflow" durante la operación de limpieza, la escritura no tiene efecto.

1 = Timer ha sido reseteado

0 = Timer no ha sido reseteado todavía

- Timer Overflow interrupt Enable (TOE)

habilita interrupciones por "timer overflow"

1 = Habilita interrupción

0 = Deshabilita interrupción

- Timer Stop (TSTOP)

– Detiene el conteo del timer

1 = Timer stopped

0 = Timer active

- Timer Reset (TRST)

– Resetea el timer counter y el prescaler

– Limpieza automática después del counter reset

1 = Reset contador y prescaler

0 = No tiene efecto

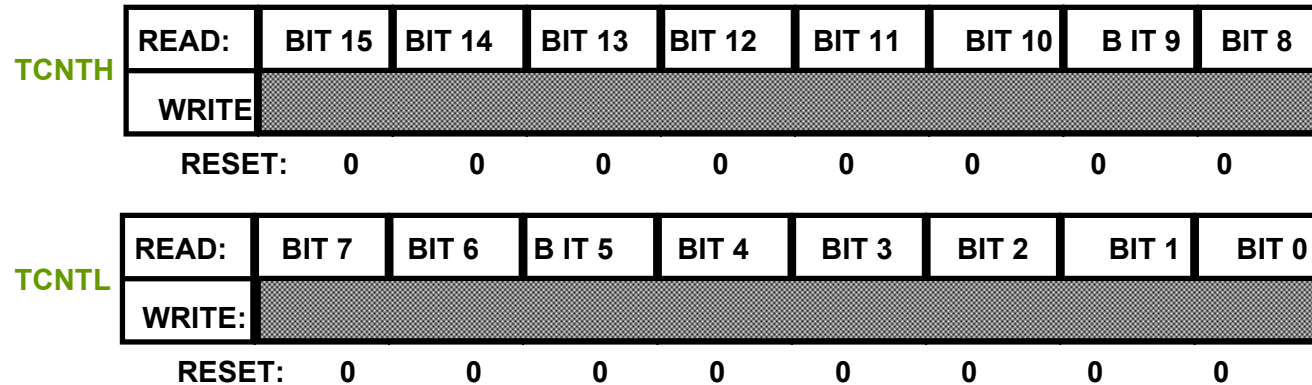
NOTA: Seteando ambos TSTOP y TRST

detiene el contador en \$0000 !!!!!

Timer Prescale Select Bits (PS0-2)

PS2	PS1	PS0	TIM Clock Source
0	0	0	Internal Bus Clock \div 1
0	0	1	Internal Bus Clock \div 2
0	1	0	Internal Bus Clock \div 4
0	1	1	Internal Bus Clock \div 8
1	0	0	Internal Bus Clock \div 16
1	0	1	Internal Bus Clock \div 32
1	1	0	Internal Bus Clock \div 64
1	1	1	no disponible !!

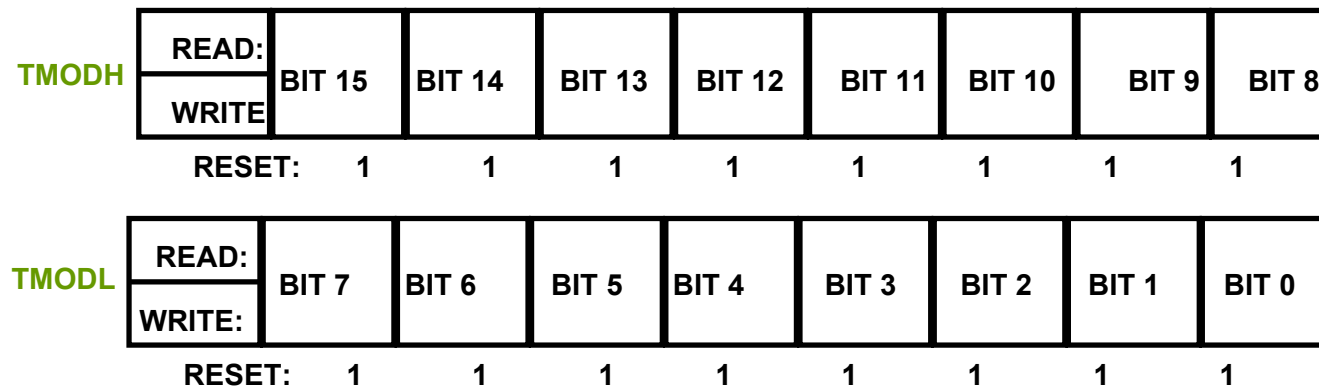
Timer Counter Register



Timer Counter Register (TCNTH, TCNTL)

- Contador de solo lectura “free running” de 16 bits
- Se lee el byte alto manteniendo el byte bajo “latcheado” hasta la lectura

Timer Modulo Register



Timer Modulo Register (TMODH, TMODL)

- El Contenido se compara con TCNTH, TCNTL para determinar el tiempo de reset
- Escribiendo el TMODH deshabilita TOF y overflow interrupts, hasta escribir el TMODL

Curso de Microcontroladores

Familia HC908 Flash de Motorola

Resolución y Rango del Timer

La resolución de Timer está determinada por el Clock del sistema y el valor del prescaler

$$\text{Resolución (sec)} = 1 \div (\text{Bus Clock} \div \text{prescaler})$$

El rango del Timer depende del valor en TMODH y TMODL

$$\text{Rango} = 0 \dots \text{Resolución} \times \text{valor de TMOD}$$

$$\text{Max Rango} = 0 \dots \text{Resolución} \times 65,535$$

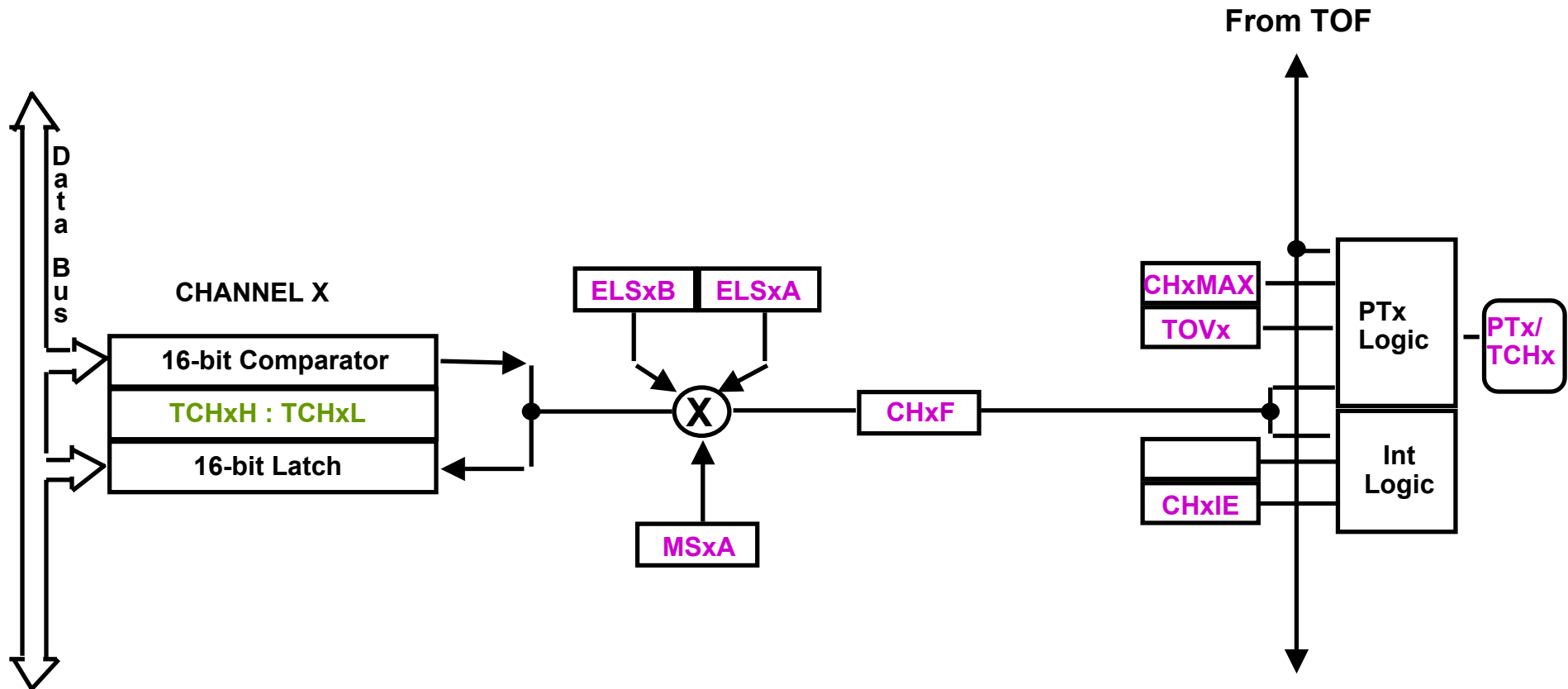
Ejemplo:

- Calculamos la resolución y el rango dado por un **Bus clock de 4 Mhz**, y un valor de **prescaler de 4 (010)**, y **TMOD = \$00FF**

$$\text{Resolución} = 1 \div (4 \text{ MHz} \div 4) = 1 \div 1 \text{ MHz} = 1 \mu\text{s}$$

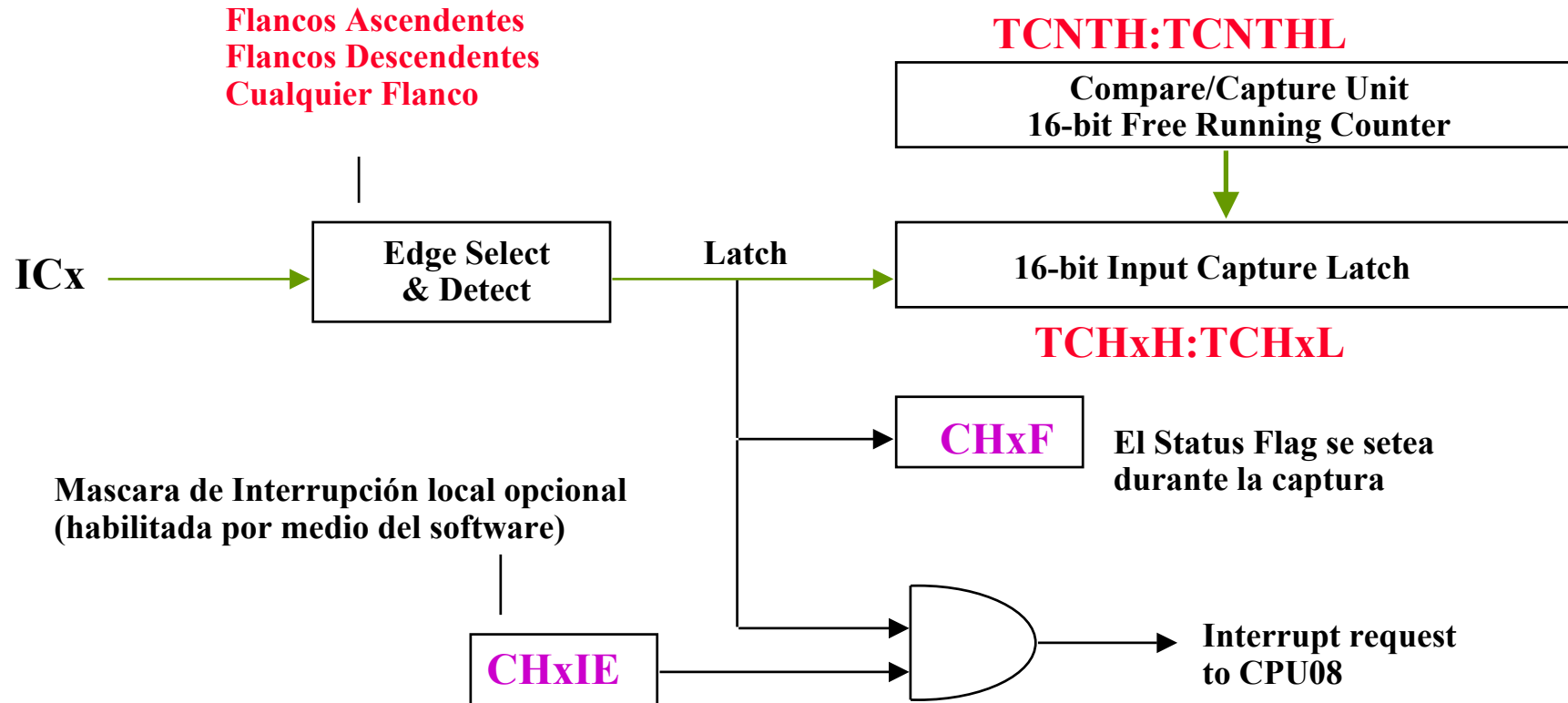
$$\text{Rango} = 1 \mu\text{s} \times \$00FF = 1 \mu\text{s} \times 255 = 255 \mu\text{s}$$

Diagrama en Bloques del Timer del Canal

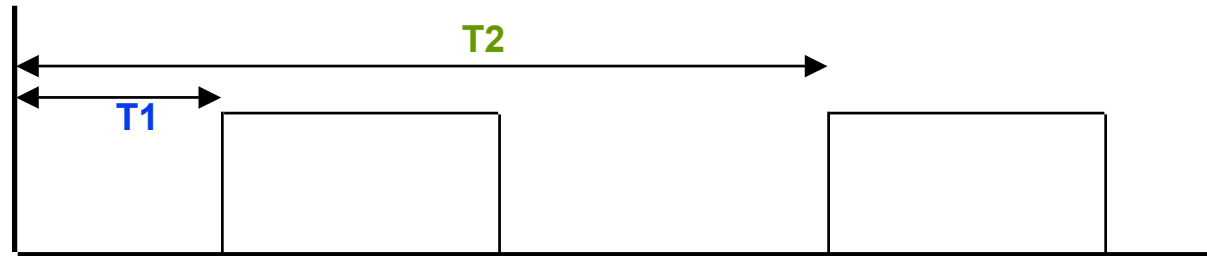


Función "Input Capture"

- Provee un mecanismo para capturar el tiempo en el cuál ocurre un evento externo



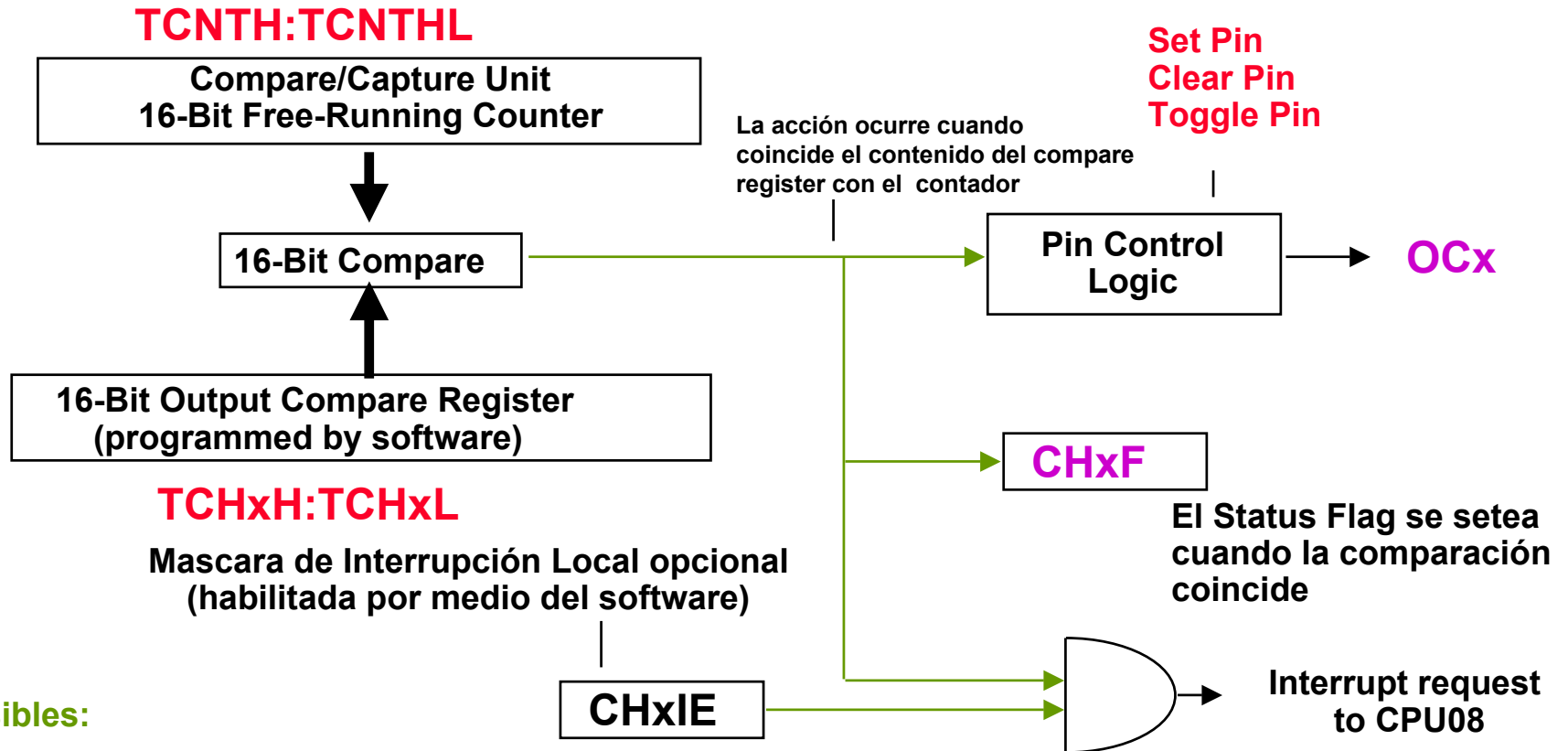
Ejemplo de Input Capture - Medición del Ancho de Pulso -



1. Configurar el canal del timer para “ input capture” , “ rising edge”
2. Tiempo de Captura **T1**
3. Tiempo de Captura **T2**
4. **Periodo = T2 - T1**

Función "Output Compare"

- Provee un mecanismo para sacar una señal a un tiempo específico



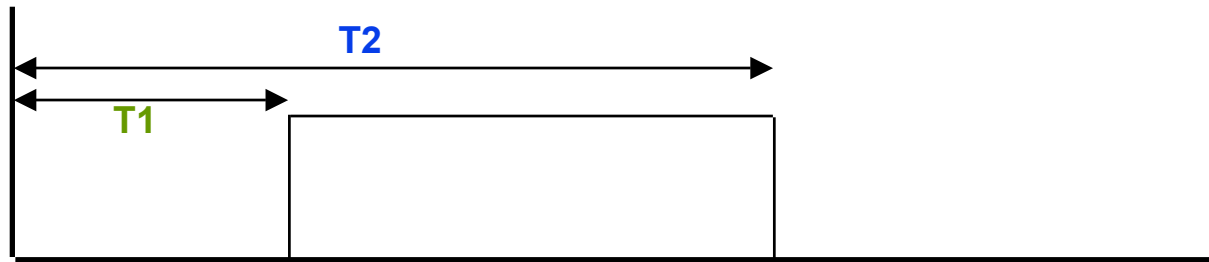
Usos Posibles:

- Generación de formas de ondas o pulsos
- Indicador de tiempo transcurrido (a un circuito externo)
- Disparo de eventos externos

Curso de Microcontroladores

Familia HC908 Flash de Motorola

Ejemplo de “Output Compare” - Generación de Pulsos -



1. Setear el valor a comparar **T1**
2. Configurar el canal de timer para “output compare”, setear la salida
3. Setear el valor a comparar **T2**
4. Configurar el canal de timer para “output compare”, limpiar la salida
5. **Ancho de pulso generado** = **T2** - **T1**

Acción “Timer Overflow”

TSCx	READ:	CHxF	CHxIE	MSxB	MSxA	ELSxB	ELSxA	TOVx	CHxMAX
	WRITE:	0							
RESET:		0	0	0	0	0	0	0	0

x = número de canal 0, 1.

Timer Channel Status and Control Registers (TSCx)

- **Toggle on Overflow (TOVx)**
 - Tiene control solo en modo “Output Compare” y PWM
 - No tiene efecto cuando el canal está configurado como “input capture”
 - Normalmente usado en generación de PWMs
- 1 = Cambia el estado de la salida cuando hay “Timer Overflow ”
- 0 = No hace nada cuando hay “Timer Overflow ”

Channel Timer Registers

TCHxH	READ:	BIT 15	BIT 14	BIT 13	BIT 12	BIT 11	BIT 10	BIT 9	BIT 8
	WRITE:								

RESET: INDETERMINADO DESPUES DEL RESET

TCHxL	READ:	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
	WRITE:								

RESET: INDETERMINADO DESPUES DEL RESET

x = número de canal 0, 1.

Timer Channel registers (TCHx)

- **Input Capture**
 - El valor del “Timer register ” es memorizado cuando aparece un “input capture”
- **Output Compare**
 - Valor a comparar con el timer

Selección de Modo, Flanco, y Nivel

TSCx

READ:	CHxF								
WRITE:	0	CHxIE	MSxB	MSxA	ELSxB	ELSxA	TOVx	CHxMAX	

RESET: 0 0 0 0 0 0 0 0

x = número del canal 0, 1.

MSxB:MSxA	ELSxB:ELSxA	Mode	Configuration
X0	00	Output Preset	Pin under port control; initial output lvl high
X1	00	Output Preset	Pin under port control; initial output lvl low
00	01	Input Capture	Capture on Rising Edge Only
00	10	Input Capture	Capture on Falling Edge Only
00	11	Input Capture	Capture on ANY (rising or falling) Edge
01	01	Output Compare	Toggle output line on Output Compare
01	10	Or	Clear output line to 0 on output compare
01	11	PWM	Set output line to 1 on output compare
1X	01	Buffered Output	Toggle output on compare
1X	10	Compare Or	Clear output on compare
1X	10	Buffered PWM	Set output on compare

Interrupciones & Estados del canal

TSCx	READ:	CHxF	CHxIE	MSxB	MSxA	ELSxB	ELSxA	TOVx	CHxMAX
	WRITE:	0							
RESET:	0	0	0	0	0	0	0	0	0

x = número de canal 0, 1.

Timer Channel Status and Control Registers (TSCx)

- **Channel x Interrupt Enable (CHxE)**
 - Habilita las interrupciones del TIM al CPU sobre el canal x.
 - 1 = Canal x CPU interrupt requests habilitado
 - 0 = Canal x CPU interrupt requests deshabilitado

- **Channel Status Flag (CHxF)**
 - **Input Capture**
 - Seteado cuando ocurre un flanco activo
 - **Output Compare**
 - Seteado cuando el valor en el “TIM counter registers” coincide con el valor en el TIM channel register
 - Limpiado por la lectura del “status register “ luego de escribir un “ 1” en CHxF
 - 1 = Input capture o output compare en canal x
 - 0 = No evento capture o output compare en canal x

Curso de Microcontroladores

Familia HC908 Flash de Motorola

Señal PWM Unbuffered

(el más conocido de los modos de generación de PWM...)

Cualquier canal puede generar Unbuffered PWM

- Se usa “output compare”
- La salida cambia de estado basada en el “timer overflow”

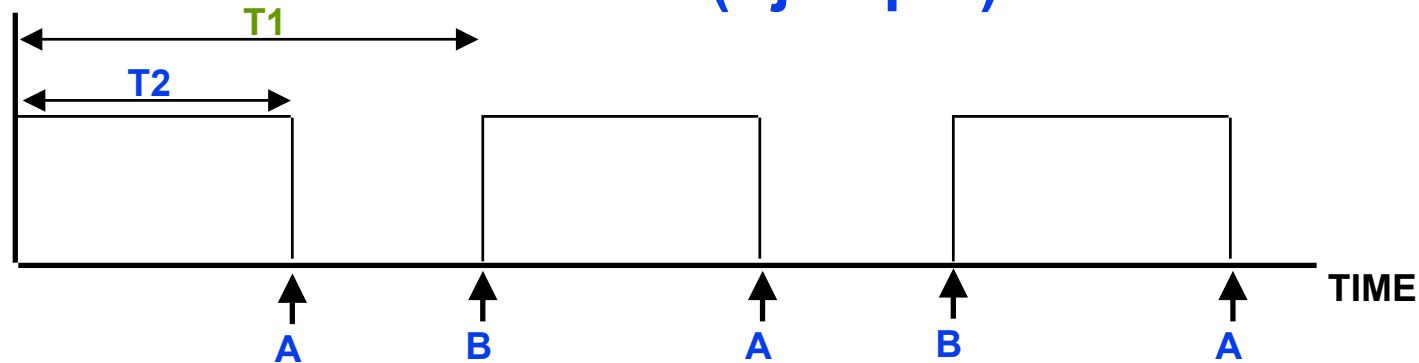
El período PWM puede fijarse por:

- Valor de la cuenta del Modulo (Modulo count value)
- La salida del “Clock prescaler”

La duración del ancho del pulso puede fijarse por:

- Valor del “Timer compare register”
 - El canal del timer configurado para forzar el pin de salida para complementar nivel del ancho de pulso

Generación de la señal PWM (Ejemplo)



A = Output compare, limpia la salida cuando ocurre **B** = Timer overflow, “cambia” la salida

T1 = Período PWM = comandado por el Timer overflow

- **Calcular el valor del prescaler y $TMOD = T1$**

T2 = Ancho del Pulso = Valor del “Output compare”

Ejemplo: Quiero un PWM con un 50% de duty cycle y período de 100 μ s desde un system clock de 4 Mhz.

Selecciono un prescaler de 4, Resolución = $1 \div (4 \text{ MHz} \div 4) = 1\mu$ s

$TMOD = T1 \div \text{Resolución} = 100\mu\text{s} \div 1\mu\text{s} = 100$

$TCHxH:TCHxL = T2 = \text{Duty cycle} \times TMOD = 50\% \times 100 = 50$

Inicialización del Unbuffered PWM

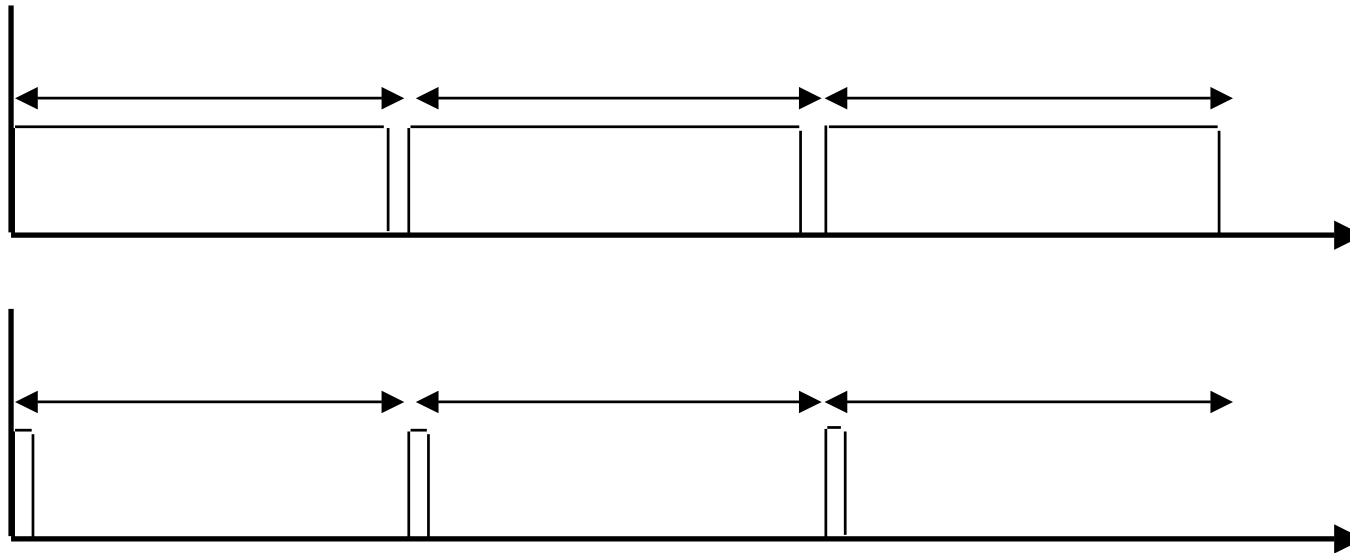
- 1) Parar y Resetear el timer
- 2) Seleccionar el valor del “timer counter modulo” y el “ timer clock prescaler” para proveer período PWM requerido.
- 3) Cargar el “ Timer compare register” con el valor del ancho del pulso
- 4) Configurar el canal del timer para operación “output compare”
- 5) Seleccionar la función “timer counter toggle on overflow”
- 6) Configurar el canal del timer para forzar el pin de salida para complementar el nivel del ancho de pulso
 - **No debe ser usado “Toggle on output compare”**
- 7) Habilitar el timer

Limitación para el Unbuffered PWM

El metodo anterior para cambiar el ancho del pulso **funciona bien en la mayoría de los casos,**
excepto cuando el cambio en el ancho del pulso es muy grande !!

Ejemplo:

No se podría cambiar desde un duty cycle del 99% al 1%



Que es un Buffered PWM?

Usa 2 output compare registers para controlar una sola salida

- Supera la “sincronización” y limitación en el ancho del pulso de los unbuffered PWMs
- Se pueden “linkear” los Canales 0 para formar un Buffered PWM

Seleccionado por los bits MS0B y/o MS2B

- “Linkea” las salidas de los canales de timer como si fueran I/O
 - Bajo el control del DDR y el data register
 - Independiente del seteo de TSC1

Operación del HC08 Buffered PWM

Configurar el canal 0 como para “unbuffered PWM”

- Excepto MSxB bit que es puesto a set en TSCx register

El ancho del pulso inicial debe ser cargado en el output compare register del canal 0

Subsecuentes valores de ancho de pulso son escritos en el canal linkeado inactivo en cualquier tiempo

- Escribiendo en el compare register del canal inactivo habilita ese canal
- Los cambios en el control de salida cambian después del prox. “counter overflow “
 - Auto sincronización

PWM

Buffered/Unbuffered

Unbuffered PWMs

- **Ventajas**
 - Consistente, Forma de onda PWM no asistida
 - Periodo y duty cycle programable
- **Desventajas**
 - Deben sincronizarse cambios con el duty cycle

Buffered PWMs

- **Ventajas**
 - Cambios no sincronizados con el duty cycle, auto sincronismo
- **Desventajas**
 - Require 2 canales de timer
 - Debe mantenerse vigilado el canal inactivo

Sumario de Registros

TSC	TOF	TOIE	TSTOP	TRST	0	PS2	PS1	PS0
TDMA	0	0	0	0	DMAS3	DMAS2	DMAS1	DMAS0
TCNTH	BIT 15	BIT 14	BIT 13	BIT 12	BIT 11	BIT 10	BIT 9	BIT 8
TCNTL	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
TMODH	BIT 15	BIT 14	BIT 13	BIT 12	BIT 11	BIT 10	BIT 9	BIT 8
TMODL	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
TCS0	CH0F	CH0IE	MS0B	MS0A	ELS0B	ELS0A	TOV0	CH0MX
TCH0H	BIT 15	BIT 14	BIT 13	BIT 12	BIT 11	BIT 10	BIT 9	BIT 8
TCH0L	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
	⋮						⋮	
TCSn	CHnF	CHnIE	0	MSnA	ELSnB	ELSnA	TOVn	CHnMX
TCHnH	BIT 15	BIT 14	BIT 13	BIT 12	BIT 11	BIT 10	BIT 9	BIT 8
TCHnL	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0

Curso de Microcontroladores

Familia HC908 Flash de Motorola

```

*****
* ELECTROCOMPONENTES S.A. *
* CURSO EN LA WEB FLIA. MICROCONTROLADORES HC908 FLASH DE MOTOROLA *
* BUENOS AIRES - REPUBLICA ARGENTINA *
* *
* TITULO: TMR_OVF.ASM *
* "Timer Overflow - Contador de 30 segundos" *
* Creado en: Diciembre del 2001 *
* Autor: ING. DANIEL DI LELLA - D.F.A.E For Motorola Products *
* Descripción: *
* ----- *
* Este programa usa la función "Timer Overflow" para incrementar *
* Un contador binario(PTB)cada 30 segundos. Este limpia el bit de *
* Status TOF en el "Timer Status Register" (TSC) e incrementa una *
* Variable en RAM "COUNT". Si el valor de esta variable es igual a *
* $00E4, entonces los 30 segundos han transcurrido. Una vez que esto *
* ha sucedido, un contador binario que usa el PORTB es incrementado, *
* el contador en RAM es limpiado, y el proceso comienza nuevamente. *
* Si los 30 segundos no han transcurrido, el contador es incrementado*
* Y luego se espera por el próximo Flag de interrupción de Overflow *
* Hasta que se alcanza el número de "overflows" requeridos para los *
* 30 segundos. La resolución del Timer, es inversamente proporcional *
* Al clock del sistema y al valor del prescaler: *
* Resolución (sec) = 1 / (Clock del Sistema / prescaler) *
* El Rango del Timer dependerá del valor en TMODH y TMDL. *
* De esta forma... *
* Rango del Timer = 0 .... Resolución * TMOD (valor) *
* Rango Máximo = 0 .... Resolución * 65535 *

```

Curso de Microcontroladores

Familia HC908 Flash de Motorola

```

* Nota:Porque el contador es de 16 bits de longitud y está precedido *
* Por un Prescaler variable, el contador dá vuelta cada 65,535 a *
* 4,194,240 ciclos internos de clock, dependiendo del valor en el *
* Prescaler. *
* La resolución del Timer con un clock de sistema de 8 Mhz es de *
* 125nS-8uS, dependiendo del valor del Prescaler. El Máximo de *
* cuentas por overflow para un contador de 16-bits es $FFFF = 65535 *
* *
* Para este ejemplo ser usará TMOD=$FFFF, Timer Clock source = *
* Clock del sistema / 16, y un clock de 8 MHz *
* RESOLUCION = 1 / (8MHz/16) = 1 / .5 MHz = 2us *
* RANGO = 2 uS * $FFFF = 2 * 65535 = 2 * 65535 uS *
* Cada "timer overflow" = 65535 * 2uS = 0.13107 sec. *
* *
* Para los 30 segundos: 1 overflow = 0.13107 sec *
* *
* Número de overflows = 30 / 0.13107 = 228.88($E4) overflows *
* *
*****

```

```

INCLUDE 'EQUATES.ASM'      ; Equates para todos los registros del HC908GP32

*      VARIABLES DE USUARIO Y REGISTROS

FLAG   EQU      7
COUNT EQU      $40      ; Timer TOF (Timer Overflow Flag) variable del contador en RAM
OUTPUT EQU      $42      ; Dirección del valor a sacar por el Port B

      ORG      $8000      ; Comienzo de la Flash, Comienzo del programa ppal. en $8000

MAIN:  BSR      INIT      ; Salto a la sub-rutina "Init" para inicializar variables & timer
DONE  WAIT
      BRA      DONE      ; Salto siempre a DONE

```

```
* -----  
*      Sub-rutina INIT:  
* -----
```

```
INIT:  CLR      COUNT      ; Limpio valor "COUNT" para inicializarlo a cero  
       CLR      OUTPUT     ; Limpio Valor "OUTPUT"  
  
       LDA      OUTPUT     ; Cargo Acumulador con valor OUTPUT  
       STA      PTB        ; Guardo Acc. en Port B para encender LED's  
  
       LDA      #$FF       ; Inicializo el PORTB como outputs (salidas)  
       STA      DDRB       ; poniendo 1's en DDRB  
  
       LDA      TSC        ; Secuencia requerida para limpiar el TOF flag: (1) Leo el TSC reg.  
       BCLR     FLAG,TSC   ; (2) Escribo "0" en el TOF flag
```

```
* Para este ejemplo: seteo el Modulo a $FFFF
```

```
       LDHX     #$FFFF     ; Cargo H:X con el valor del modulo  
       STHX     TMODH     ; Descargo el valor al TMOD register  
  
       MOV      #$54,TSC   ; Habilito interrupción por overflow  
*                               ; Reseteo & habilito el contador, seteo prescaler = Clock / 16  
  
       CLI                               ; Limpio el I flag, habilito interrupciones del CPU  
       RTS                               ; Return from Sobroutine
```

Curso de Microcontroladores

Familia HC908 Flash de Motorola

```

* -----
*   Timer Overflow Interrupt Service routine (RUTINA DE T.O.I)
* -----
TOVVEC  INC     COUNT      ; Incremento "COUNT"
        LDA     COUNT      ; Cargo Acumulador con "COUNT"
        CBEQA  #$E4,SEC_30 ; Comparo para saber si se está cerca del valor de 30 segundos
*
*                               ; en el número de Overflows (E4)
CONT:   LDA     TSC        ; Secuencia requerida para limpiar el TOF flag: (1) Leo el TSC reg.
        BCLR   FLAG,TSC   ; (2) Escribo "0" en el TOF flag
        RTI     ; Return from Interrupt (vuelvo a main !!)

* -----
*   SEC_30 Service routine (Rutina de 30 segundos cumplidos)
* -----
SEC_30: INC     OUTPUT     ; Incremento la variable OUTPUT
        CLR     COUNT     ; 30 sec. CUMPLIDOS luego hago cero el contador y comienzo nuevamente
        LDA     OUTPUT    ; Cargo Acumulador con valor OUTPUT
        STA     PTB       ; Descargo el Acc. en el Port B para encender los LED's
        BRA     CONT      ; Salto siempre a CONT: para limpiar el TOF flag

```

```

*-----
*      ;VECTORS (vectores)
*-----
      ORG      $FFEC      ; Inicializo el Timer Overflow Vector
      FDB      TOVVEC     ; en $FFF4
*-----
*      Inicializo el "RESET" Vector
*-----
      ORG      $FFFE      ; Inicializo el reset Vector en $FFFE
      FDB      MAIN
*-----
      END                ; End of program (solo a titulo ilustrativo)

```

```

*****
* ELECTROCOMPONENTES S.A. *
* CURSO EN LA WEB, FLIA MICROCONTROLADORES HC908 FLASH DE MOTOROLA *
* BUENOS AIRES - REPUBLICA ARGENTINA *
* *
* TITULO: OUTCOMP.ASM *
* ----- *
* USO DEL "OUTPUT COMPARE" PARA GENERAR UNA ONDA CUADRADA *
* CREADO EN: DICIEMBRE DEL 2001 *
* Autor: ING. DANIEL DI LELLA D.F.A.E For Motorola Products *
* Descripción: *
* ----- *
* Este programa genera una onda cuadrada, 50% duty cycle, en el *
* TCH0 - Output compare 0. Esto asume un Bus clock de 8.0 MHz *
* Los registros de control son inicializados para habilitar la *
* Interrupción de TCH0, la acción adecuada del pin y también el valor*
* TCH0H y TCH0L para la comparación . La cuenta correcta debe ser *
* calculada para alcanzar la frecuencia y el duty cycle adecuados *
* Por ejemplo: Para un ciclo de un 50%, y una señal de 1KHz *
* Cada período debe consistir en 4000 cuentas o 2000 cuentas "high" *
* Y 2000 cuentas "low". *
* En esencia un $03E8 se suma para generar una frecuencia de 1 kHz. *
* *

```

```

*                                                                 *
* EJEMPLO DE CALCULO: para obtener las cuentas del periodo      *
* Bus clock = 8 MHz, IC/OC resolución = 1/(Bus clock/Prescaler) *
* Si el Prescaler = 4                                           *
* Luego la resolución del Output compare es 0.5 uS            *
*                                                                 *
* Para un 1KHz, 50% duty cycle                                  *
*                                                                 *
* CALCULO: Para 1000Hz 1/F = T = 1/1000 = 1mS                  *
* F para "output compare" = Prescaler/Bus clock = 2 MHz        *
*                                                                 *
* |<----- 1mS ----->| Número de clocks = F * D            *
*                                                                 *
* |_____| |_____| |_____| | therefore                          *
* |_____| |_____| |_____| |                                     *
*                                                                 *
* |<.5ms>| Número de clocks = (2 MHz) * (.5mS) = 1000 = $03E8*
*                                                                 *
*-----*

```

```

INCLUDE 'EQUATES.ASM' ; Equates para todos los registros HC908GP32
* Bit Equates
FLAG EQU 7
ORG $8000 ; Comienzo de la FLASH, comienzo del programa ppal. en $8000
MAIN:
BSR TIMERINIT ; Sub-rutina usada para inicializar las interrup. del timer:
; Output compare channel
BRA * ; Salta a si mismo
* -----
* Sub-rutina TIMERINIT: (INICIALIZACION DEL TIMER)
* -----
TIMERINIT:
LDX #$00 ; Inicializo el registro X, requerido por el simulador

LDA #$00 ; Cargo el Acumulador A con el valor luego;
STA TCH0H ; Escribo el TCHxH para "Inhibir" cualquier Output compare
MOV #$A,TCH0L ; Escribiendo el TCHxL "Habilito" el Output Compare

LDA #$D4 ; Deshabilito las interrupciones por overflow ,
STA TSC0 ; Toggle Pin en output compare, habilito interrupción en TCH0

MOV #$13,TSC ; Deshabilito Timer Overflow interrupt
* ; Timer counter activo, Prescaler y timer
* ; counter limpios, Prescaler = Bus clock / 16

CLI ; Limpio el I flag, habilito interrupciones del CPU
RTS ; Return from Subroutine

```

Curso de Microcontroladores

Familia HC908 Flash de Motorola

Parte II

ING. DANIEL DI LELLA DDFAE For Motorola Products

```

* -----
* Output Compare channel interrupt service routine (solo entra
* en función cuando una comparación ocurre .
* -----
* Para limpiar el CH0F flag: 1) Leo el TSC0 cuando
* CH0F está puesto a "1" y luego 2) Escribo un cero logico en CH0F.
OC0_ISR BSR      SQWAVE
        BCLR    FLAG,TSC0 ; Limpio flag por la escritura de un cero logico
        RTI     ; Return from Interrupt
* -----
*      SUB-RUTINA:  SQWAVE
* -----
SQWAVE:
        LDA     TSC0      ; Leo el flag en TSC0 Register
        LDA     TCH0L     ; Cargo Acc. Con valor inicial "Low" del compare register
        ADD     #$A       ; Sumo el valor del tiempo HIGH al byte low de TCH0L,
*                          ; para setear la próxima transición,
        TAX     ; Descargo el resultado a la localización temporaria (X)
*                          ; Esto se hace porque al escribir el TCH0L habilito
*                          ; el output compare, y NO se quiere habilitar ello
*                          ; hasta que No se tenga el byte HIGH (TCH0H)
        LDA     TCH0H     ; Cargo Acc. con TCH0H
        ADC     #$0       ; Sumo "0", más el Carry desde la adición previa
        STA     TCH0H     ; Descargo el resultado (Valor High) a TCH0H
        STX     TCH0L     ; Escribo el "low byte" del valor a comparar en TCH0L
*                          ; esto también habilita el OC (OUTPUT COMPARE)
        RTS     ; return from Subroutine

```

Curso de Microcontroladores

Familia HC908 Flash de Motorola

Parte II

ING. DANIEL DI LELLA DDFAE For Motorola Products

```

* -----
*           ;VECTORS (vectores)
* -----

          ORG    $FFF4    ; Inicializo Output Compare Vector para el canal 0
          FDB    OC0_ISR  ; en $FFF4

* -----
* Inicializo el "RESET" Vector
* -----

          ORG    $FFFE    ; Inicializo el reset Vector en $FFFE
          FDB    MAIN

          END            ; End of program (solo a titulo ilustrativo)

```

FIN CAPITULO 12 !!!