NT0101

2.1 CONTROL DE EVENTOS TEMPORIZADOS – "TIMER MODULE"

GENERACIÓN DE RETARDOS DE "HARDWARE"

Preparado por: Elías Lombardo Batista
Y por: Rangel Alvarado
Estudiante Graduando de Lic. en Ing. Electromecánica
Universidad Tecnológica de Panamá
Panamá, Panamá
"e-mail": issaiass@cwpanama.net
"web site": http://www.geocities.com/issaiass/

ÍNDICE

2.1.1 Introducción

Existen muchas aplicaciones las cuales requieren el control de eventos temporizados, por ejemplo: refrescar una pantalla a intervalos de tiempo específicos, generación de una onda de determinada frecuencia, frecuencia variable y captura del ancho del pulso.

Normalmente, estas rutinas se implementan con bucles repetitivos, como se implementó en notas anteriores la función "**delay**" (NT0009), pero esta rutina es altamente inefectiva e imprecisa, si se requiere una estricta precisión en el "hardware".

La solución para desarrollar una "precisión al centavo" es el uso de contadores internos llamados temporizadores. Cuando un programa sirve de esta característica para generar retardos de tiempos, se dice que utiliza un "**Hardware Time Delay**".

El objetivo de esta nota es:

- Conocer el objetivo de un temporizador: como se ha explicado en párrafos anteriores, el temporizador controla eventos en los cuales se requiere una fina precisión en el "hardware" o disparo de una señal a un tiempo en específico.
- Familiarizarse con los registros básicos del módulo: por el momento, los registros a estudiarse solo sirven para temporizar eventos, pero el módulo completo, tiene el poder de generar capturas de ancho de pulso y variar el ancho del pulso, lo que se conoce como una PWM.
- Manejar la interrupción de desborde de sobre flujo: La interrupción de sobre flujo nos ayuda a controlar eventos temporizados, o mejor aún, a temporizar en grandes cuentas por medio del control de flujo del programa.
- Simular: Utilizar los "breakpoints" para visualizar la rutina de interrupción.

2.1.2 El Temporizador Interno

El temporizador es un contador de tiempo sincronizado por el cristal. La función del mismo es contar desde cero (0) hasta algún número solicitado, para luego reiniciar su conteo. Por ejemplo, si se pide al temporizador contar de cero (0) a siete (7), este entenderá que debe contar de la siguiente manera: 0, 1, 2, …, 7, 0, 1….

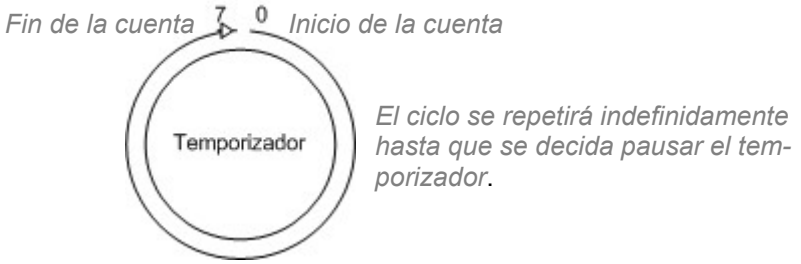


Figura 112. Analogía de un Temporizado. El temporizador es un contador libre que repite la cuenta cada vez que se desborde su límite máximo de conteo.

Nota: El temporizador cuenta con dos canales de dieciseis (16) bits:

Tabla 49. Canales del Temporizador

Canal	PIN JK3	PIN JL3	PIN GP32	PIN QT4
CH0	26	19	21	7
CH1	25	18	22	6

Los números en las columnas corresponden a la distribución de pines según cada microcontrolador.

2.1.3 Registro del Temporizador

Para realizar el temporizado, principalmente se utilizan tres registros; la cuenta (valor de conteo sucesivo), se almacena en los registros TCNT[H:L]; mientras que el módulo del contador (valor máximo de conteo), se almacena en los registros TMOD[H:L]; para que realice lo anterior se debe configurar el temporizador por medio del registro de estado y control TSC.

2.1.3.1 Registro de Estado y Control del TIM ("Timer Interface Module") - TSC

El registro de estado y control configura el temporizador para, habilitar o inhabilitar el temporizador, interrupciones del temporizador, reinicialo o hacer que su resolución varíe.

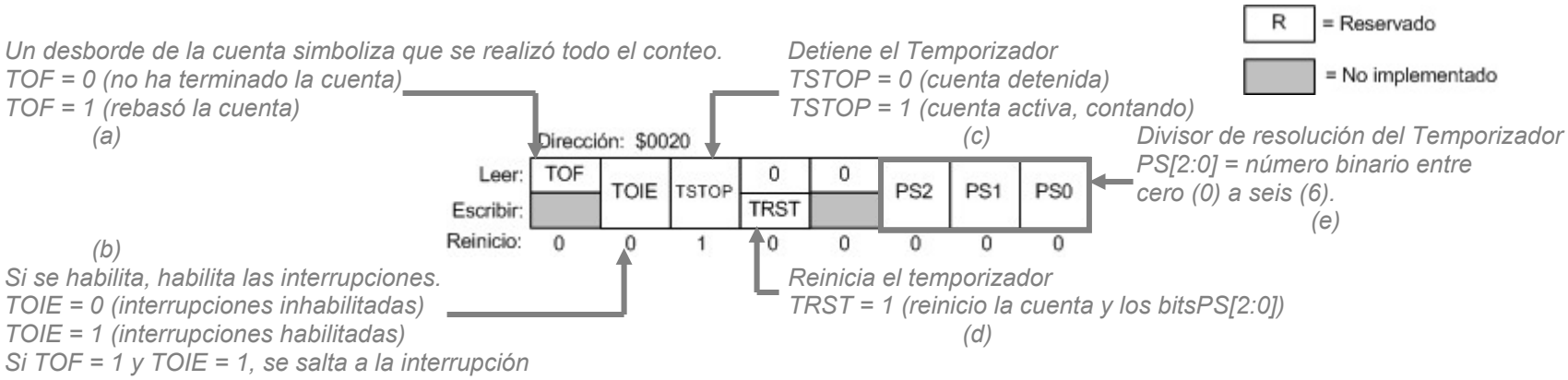


Figura 113. Registro de Estado y Control - TSC. (a) El bit TOF controla el desborde de la cuenta e interrumpe, solo si el bit TOIE está activo. (b) El bit TOIE controla la habilitación o deshabilitación de la interrupción del temporizador. (c) TSTOP reanuda o detiene la cuenta del temporizador. (d) TRST reinicia la cuenta y los valores prescalares PS[2:0] y finalmente. (e) PS[2:0] disminuye la resolución del canal.

2.1.3.2 Registro de Conteo del TIM – TCNT[H:L]

El registro de conteo, es un registro de solo lectura que almacena la cuenta del temporizador.

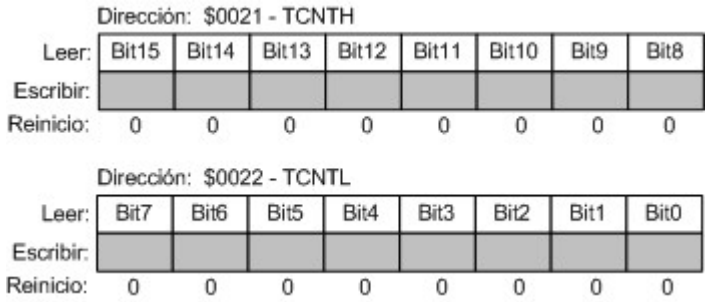
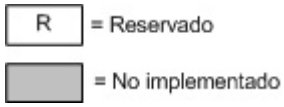


Figura 114. Registro de Conteo – TCNT[H:L]. Este registro actúa como un contador libre de solo lectura que deposita la cuenta del temporizador hasta alcanzar su desborde. Nota: En un programa, se debe leer primero el registro TCNTH y luego el registro TCNTL



2.1.3.3 Registro Contador del Módulo del TIM – TMOD[H:L]

El registro módulo contiene el valor máximo de la cuenta. Cuando el temporizador alcanza el valor máximo del módulo, la bandera TOF se levanta (1) y el conteo (TMOD[H:L]) se reinicia; adicionalmente, si la bandera TOF “y” TOIE están levantadas, se procede a seguir la rutina de interrupción.

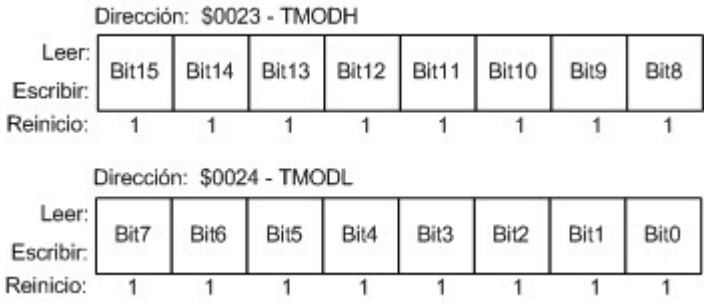


Figura 115. Registro Contador del Módulo – TMOD[H:L]. Escribiendo un valor en este registro, actúa como valor máximo de desborde del canal; cuando el valor de este registro es sobrepasado, se levanta la bandera TOF avisando un conteo completo y si está activado el bit TOIE, se procede a la búsqueda del vector de interrupción.

2.1.3.3 Ecuación del Módulo del Contador

Para programar nuestro temporizador un determinado período, se debe utilizar la siguiente ecuación:

$$TMOD[H:L] = \frac{t \cdot f_{xtal}}{2^{2+PS}}$$

Ecuación 5. Ecuación del Módulo del Temporizador

TMOD[H:L] = Módulo de conteo, frecuencia a la que se repite la cuente, valor máximo.
t = tiempo en segundos que durará la cuenta.
f_{xtal} = frecuencia del oscilador externo, cristal de cuarzo.
PS = valor del preescalar, divisor, retarda mayor tiempo la cuenta.

Ejemplo: Para un PS = 0 y un Cristal de 4.9152 MHz, ¿cuál será el valor a depositar en el registro módulo si se desea una base de tiempo de 1ms?.

$$TMOD[H:L] = \frac{t \cdot f_{xtal}}{2^{2+PS}} = \frac{0.001 \cdot 4.9152 \times 10^6}{2^{2+0}} = 1228.8_{10} = 4CD_{16}$$

Para una base de tiempo de 1 ms, se debe programar el registro TMOD con $4CD_{16}$.

2.1.4 Interrupción del Temporizador

Después de que la cuenta es mayor al módulo de conteo (TCNT[H:L] > TMOD[H:L]), el registro de control (TSC) habilita el bit de sobreflujo del canal (TOF) y si el bit de interrupciones (TOIE) está activo, se procede la búsqueda del vector de interrupción, desviando el flujo del programa hacia esta rutina.

Dirección del vector interrupción = $FFF2

Nota: Se recomienda en las primeras líneas del vector de interrupción, se borre el bit TOF (TOF = 0), indicando que el evento ha sido atendido; de lo contrario recaerá en un bucle una y otra vez sin parar.

Tabla 50. Vectór de Interrupción de Sobreflujo del Temporizador

Bandera	Máscara	Dirección	Vector (Dirección)
TOF	TOIE	FFF2	Sobreflujo del TIM (Alto)
		FFF3	Sobreflujo del TIM (Bajo)

2.1.5 Diagrama de Flujo

El siguiente programa utiliza una base de tiempo de 1ms para conmutar el estado del LED de la tarjeta, encendiéndolo y apagándolo por intervalos de un (1) segundo.

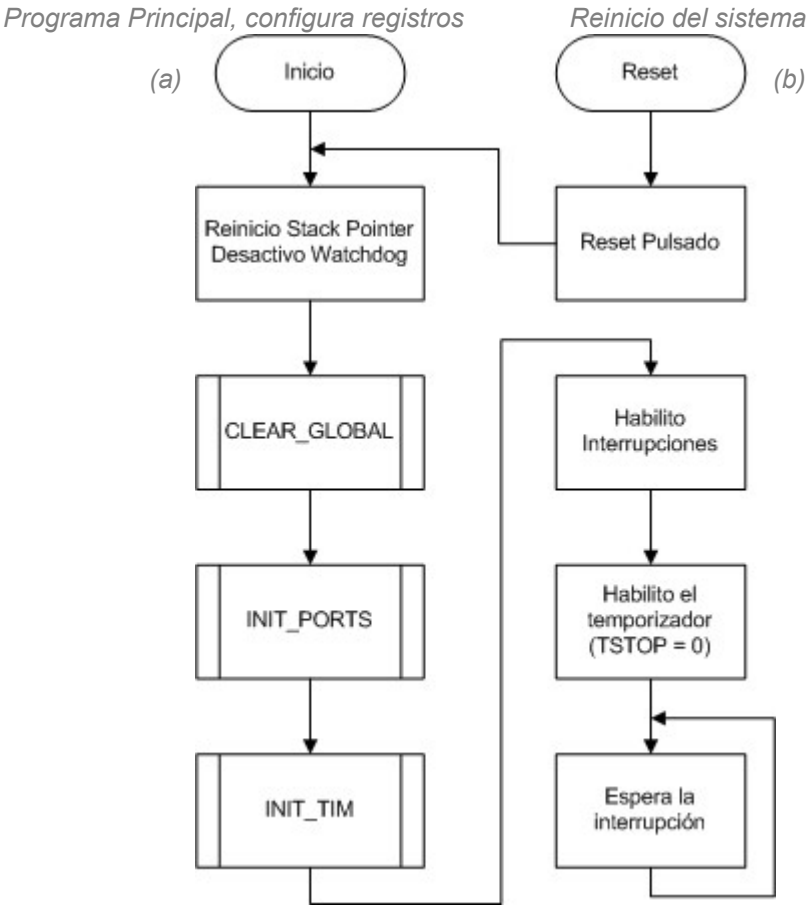


Figura 116. NT0101 – "Timer" - Temporizador. (a) Programa Principal. Inicializa variables, limpia registros, inicializa puertos y espera que ocurra la interrupción del temporizador. (b) Reinicio del sistema. Al presionar "reset", el sistema reinicia automáticamente.

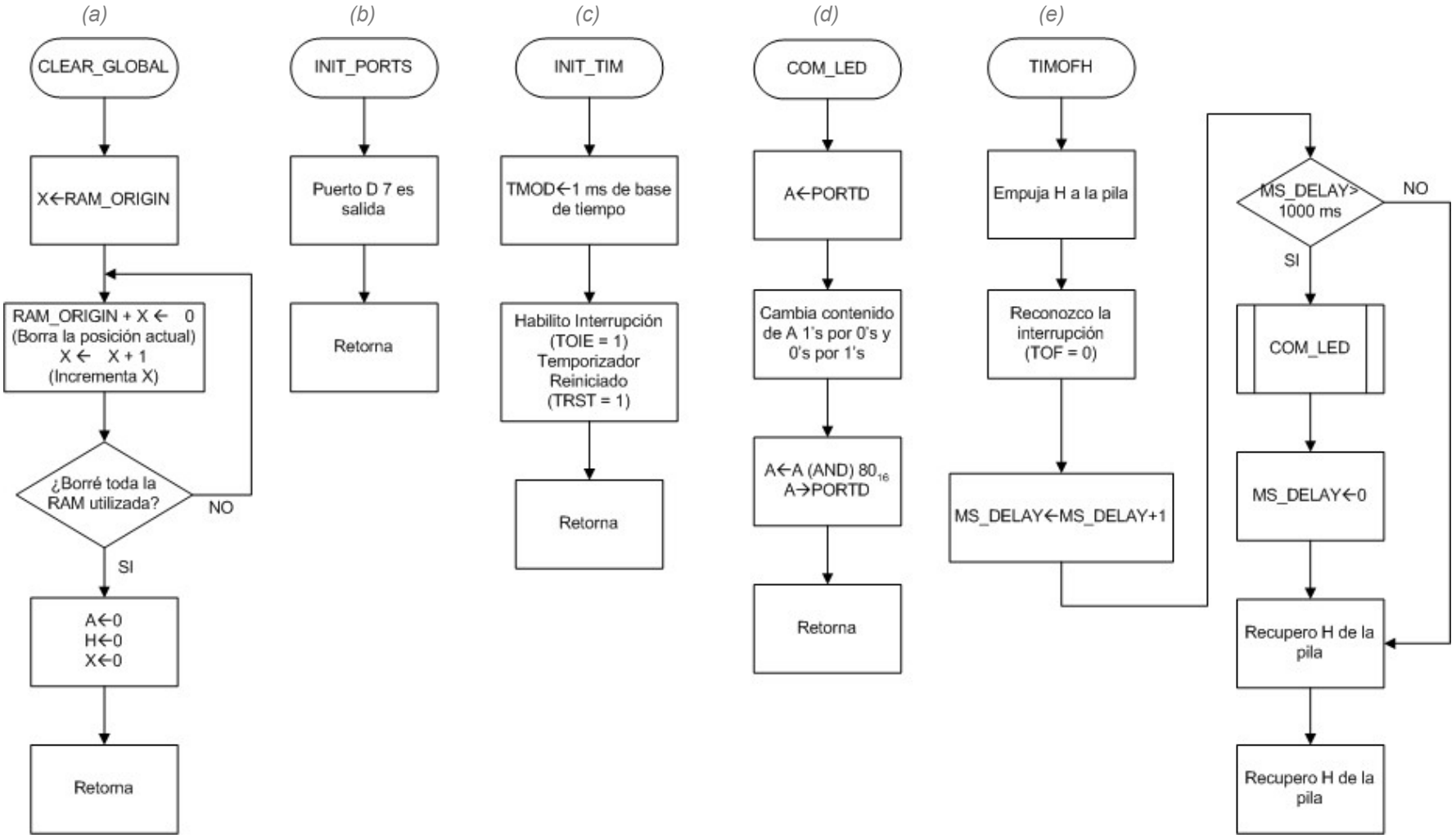


Figura 117. NT0101 – "Timer" – Temporizador – Subrutinas. (a) Clear_Global. Limpia la memoria RAM. (b) Init_Ports. Inicializa el Puerto D7 como salida. (c) Init_TIM. Carga el módulo con 1 ms de base de tiempo, habilita interrupciones y reinicia el temporizador. (d) Com_Led. Conmuta el estado del LED en PTD7. (e) TIMOFH. Interrupción del temporizador, al pasar 1000 ms, conmuta el estado del LED en el PTD7.

2.1.6 Código

```
;*************************************************************
; ARCHIVO      :  NT0101 - Timer - Temporizador - 24 03 04.asm
; PROPÓSITO    :  Genera por medio de una base de tiempo de 1ms el conmutado de
;                 un LED (PTD7) a intervalos de 1 segundo.
;                 - Observar la configuración del canal del temporizador
;                 - Generar con base de tiempo, eventos mayores a 1 segundo
;                 P.D.:  no hay que hacer ninguna conexión, esta
;                 internamente cableado.
; NOTA         :
;                 1 - Observar el resultado de la temporización en el LED PTD7
;                 2 - Observar el resultado del registro TIM en las localidades
;                 TSC, TMOD[H:L] y corrida del conteo en TCNT[H:L].
; REFERENCIA:
;                 Advanced Information of MCU68HC908JK1, JK3, JL3...
;                 http://www.freescale.com/files/microcontrollers/
;                 doc/data_sheet/MC68HC08JL3.pdf
;                 Pág. 107 @ 109 - Descripción Funcional
;                 Pág. 115 - Interrupción
;                 Págs. 117 @ 121. - Información de registros
;
;
; LENGUAJE    : IN-LINE ASSEMBLER
; ------------------------------------------------------------------------
; HISTORIAL
;  DD MM YY
;  10 03 04        Creado.
;  03 09 04        Modificado.
;*************************************************************
;Pasos para iniciar el TIM:
;
;           1 - Definir del registro TSC o registro de
;           control del TIM:
;              ¿Necesito o no interrupciones? (TOIE)
;              ¿Cuál que valor de divisor ajustar? (PS[2:0])
;           2 - Definir el registro TMOD[H:L] como módulo máximo de
;              conteo TMOD[H:L] = t*fxtal/(2^(2+PS))
;           3 - Habilitar Interrupciones globales (cli)
;           4 - Habilitar el temporizador (TSTOP = 0)
;           5 - Configurar el vector interrupción ($FFF2)
;           6 - Escribir el código de la interrupción y su retorno (rti)
;*************************************************************
$SET    ICS08                   ; ICS08 = 1, Vamos a simular en la pastilla
                                ; la velocidad de simulación es menor en la
                                ; PC.
;$SETNOT  ICS08                 ; ICS08 = 0, Vamos a programar la pastilla
                                ; la aplicación debe correr en tiempo real
$IF ICS08
MS1000          equ 0005        ; Constante de Retardo  - Simulación
$ELSEIF
MS1000          equ 1000T       ; 1000 milisegundos de retardo
$ENDIF
```

```
;=========================================
;                Definiciones del Usuario
;=========================================
COPD          equ 0T              ; Bit 0 del registro CONFIG1
MS1           equ $04CD           ; 1 milisegundo de base de tiempo
BIT7          equ 7T              ; TSC, Bit de Sobreflujo del Temporizador, Bit
                                  ; 7 ON
BIT5          equ 5T              ; TSC, Bit de Inicio de Conteo, Bit 5 ON
BIT7MASK      equ $80             ; Máscara para el Bit 7 del Puerto
RAM_ORIGIN    equ $0080           ; Inicio de la memoria RAM
RAM_USED      equ $00FE           ; Fin de limpieza de la RAM
TOF           equ 7T              ; TSC, Bit de sobreflujo del temporizador, Bit
                                  ; 7, ON
TOIE          equ %01000000       ; TSC, Bit de Interrupción habilitada, Bit 6,
                                  ; ON
TRST          equ %00010000       ; TSC, Bit de Reinicio de conteo, Bit 4 ON
TSTOP         equ %00100000       ; TSC, Bit de pausa, Bit 5 ON

;=========================================
;          Mapa de Memoria del Microcontrolador
;=========================================
;=========================================
;                Registro de E/S
;=========================================
PORTD         equ $0003           ; Registro del Puerto D
DDRD          equ $0007           ; Registro de Direccionamiento del Puerto D

;=========================================
;              Registro de Configuraciones
;=========================================
CONFIG1       equ $001F           ; Vectores de configuración

;=========================================
;              Registro de Temporizador
;=========================================
TSC           equ $0020           ; Dirección, registro de estado y control del
                                  ; TIM
TCNTH         equ $0021           ; TCNT, Registro  almacenador de cuenta del
                                  ; módulo, registro alto.
TCNTL         equ $0022           ; TCNT, Registro  almacenador de cuenta del
                                  ; módulo, registro bajo.
TMODH         equ $0023           ; TMODH, Registro de cuenta del módulo,
                                  ; registro alto.
TMODL         equ $0024           ; TMODL, Registro de cuenta del módulo,
                                  ; registro bajo.

;=========================================
;                Memoria RAM
;=========================================
RAM           equ $0080           ; Inicio de la Memoria RAM
MS_DELAY      equ RAM             ; La RAM almacena la variable de retardo
```

```
;=============================================
;                       Memoria FLASH
;=============================================
FLASH_STARTequ $EC00            ; Puntero - Mem.FLASH


;=============================================
;                       Vectores de Usuario
;=============================================
TIMOFH      equ $FFF2           ; Sobreflujo del TIM (Alto)
RESET_VEC   equ $FFFE           ; Puntero del RESET

;=============================================
; OBJETIVO    : Inicio de Codif. del Ensam-
;               blador en Memoria FLASH.
;=============================================
        org FLASH_START         ; Inicio Mem. FLASH

;=============================================
; OBJETIVO    : Configura el TIM para generar
;               una rata de parpadeo de 1
;               segundo.
;=============================================
START
        rsp                     ; Inic.Stack = $00ff
        bset COPD,CONFIG1       ; Desactiva watchdog
        jsr CLEAR_GLOBAL        ; Borra los registros; al
                                ; simulador no le simpatizan
                                ; registros no inicializados.
        jsr INIT_PORTS          ; Subr,Inic. PUERTO
        jsr INIT_TIM            ; Inicializa TIM
        cli                     ; Habilita Interrupciones
        bclr BIT5,TSC           ; Inicia el temporizado (TSTOP = 0)
ESPERA
        wait                    ; Espera la interrupción.
        bra ESPERA              ; Salta al modo de bajo consumo
```

```
;===========================================
; CLEAR_GLOBAL:  Borra la Ram utilizada y re-
;                gistros inherentes
; OBJETIVO     : Borra registros
; ENTRADA      : RAM_ORIGIN, RAM_USED
; SALIDA       : A, H:X y RAM en 0
; REGISTROS
; AFECTADOS    : RAM, H:X, A
; EJEMPLO      :
; Entrada      : RAM_ORIGIN = 0080
;                RAM_USED = 0090
;                X = 7
; Salida       : 0080 @ 0087 = 0
;===========================================
CLEAR_GLOBAL                    ; Borra registros a usar
        ldx #RAM_ORIGIN         ; Carga con el origen
FILL_EMPTY
        clr ,x                  ; rellena con "0" la posición actual
        incx                    ; incrementa puntero de RAM
        cphx #RAM_USED          ; Compara hasta el final deseado
        bne FILL_EMPTY          ; Si no concuerda entonces sigue limpiando
        clra                    ; Borra A
        clrh                    ; Borra H
        clrx                    ; Borra X
        rts                     ; retorna

;===========================================
; INIT_PORTS   : Inicializa variables y regis
;                tros.
; OBJETIVO     : Inicializa los registros de
;                direccionamiento.
;                PORTD7 = OUTPUT
; ENTRADA      : Ninguna
; SALIDA       : Ninguna
; REGISTROS
; AFECTADOS    : DDRD
;===========================================
INIT_PORTS
        bset BIT7,DDRD          ; Fija PD7 = Salida
        rts                     ; retorna
```

```
;===============================================
; INIT_TIM      : Inicializa el TIM
; OBJETIVO      : Inicialización del tempori-
;                 zador.
;                 Base de tiempo de 1ms
; ENTRADA       : Ninguna
; SALIDA        : Ninguna
; REGISTROS
; AFECTADOS     : TSC, TMODH, TMODL, HX
;===============================================
INIT_TIM
        ldhx #MS1                   ; Programa H:X para cargar al módulo con
                                    ; 1ms de retardo
        sthx TMODH                  ; Almacena 1ms de retardo en el módulo
        mov #{TOIE|TRST|TSTOP},TSC; Reseteo el temporizador, habilito
                                    ; interrupciones
                                    ; PS[2:0] = 0, el temporizador se encuentra
                                    ; detenido por defecto.
        rts                         ; retorna

;===============================================
; COM_LED       : Apaga o prende el LED depen-
;                 diendo del estado anterior.
; OBJETIVO      : Si el led del puerto D, bit 7
;                 estuvo apagado, lo enciende ó
;                 visceversa.
; ENTRADA       : Ninguna
; SALIDA        : ACCA = Estado del LED PTD7
; REGISTROS
; AFECTADOS     : ACCA, PORTD
;===============================================
COM_LED
        lda PORTD                   ; Carga ACCA con el contenido del Puerto D
        coma                        ; Cambia 1's por 0's y 0's por 1's
        and #BIT7MASK               ; Enmascara el bit 7
        sta PORTD                   ; ACCA lo almacena en el puerto D
        rts                         ; retorna
```

```
;===========================================
; TIMOFL       : Interrupción del TIM
; OBJETIVO     : Conmutar el estado del LED a
;                intervalos de tiempo de 1 seg
; ENTRADA      : Ninguna
; SALIDA       : Ninguna
; REGISTROS
; AFECTADOS    : TSC, HX, RAM
;===========================================
TIMOFL
        pshh                    ; Guarda H en la pila
        bclr BIT7,TSC           ; Reconozco la interrupción, TOF = 0
        ldhx MS_DELAY           ; Carga la variable
        aix #1                  ; Incrementa H:X en 1
        sthx MS_DELAY           ; MS_DELAY <- MS_DELAY+1
        cphx #MS1000            ; Compara con 1000 ms
        bne OUTTIM              ; ¿Ha pasado el segundo?
        jsr COM_LED             ; SI, conmuta el LED
        clr MS_DELAY            ; Borra variable de retardo, byte alto
        clr MS_DELAY+1          ; Borra variable de retardo, byte bajo
OUTTIM                          ; sale del temporizador
        pulh                    ; Recupera H de la pila
        rti                     ; NO, retorna de la interrupción

;===========================================
; OBJETIVO     : Inicializa el Vector de Reset
;                Arranque del programa en la
;                memoria Flash y temporización
;                de 1 segundo.
;===========================================
;===== Vector del temporizador, sobreflujo - TIM =======
        org TIMOFH              ; Puntero Vec - TIM
        dw TIMOFL               ; Desborde del contador
;===== Vector de Reinicio de Sistema =======
        org RESET_VEC           ; Puntero Vec - RESET
        dw START                ; al darse reset salta a Start
```

Listado 11. NT0101 – "Timer" – Temporizador. El programa principal mantiene en modo de bajo consumo ("wait") al microcontrolador, mientras espera la rutina de interrupción donde decide por medio de una variable que lleva el conteo, la cantidad de milisegundos que han pasado.

2.1.7 Simulación

En esta sección, refiérase a la figura 118.

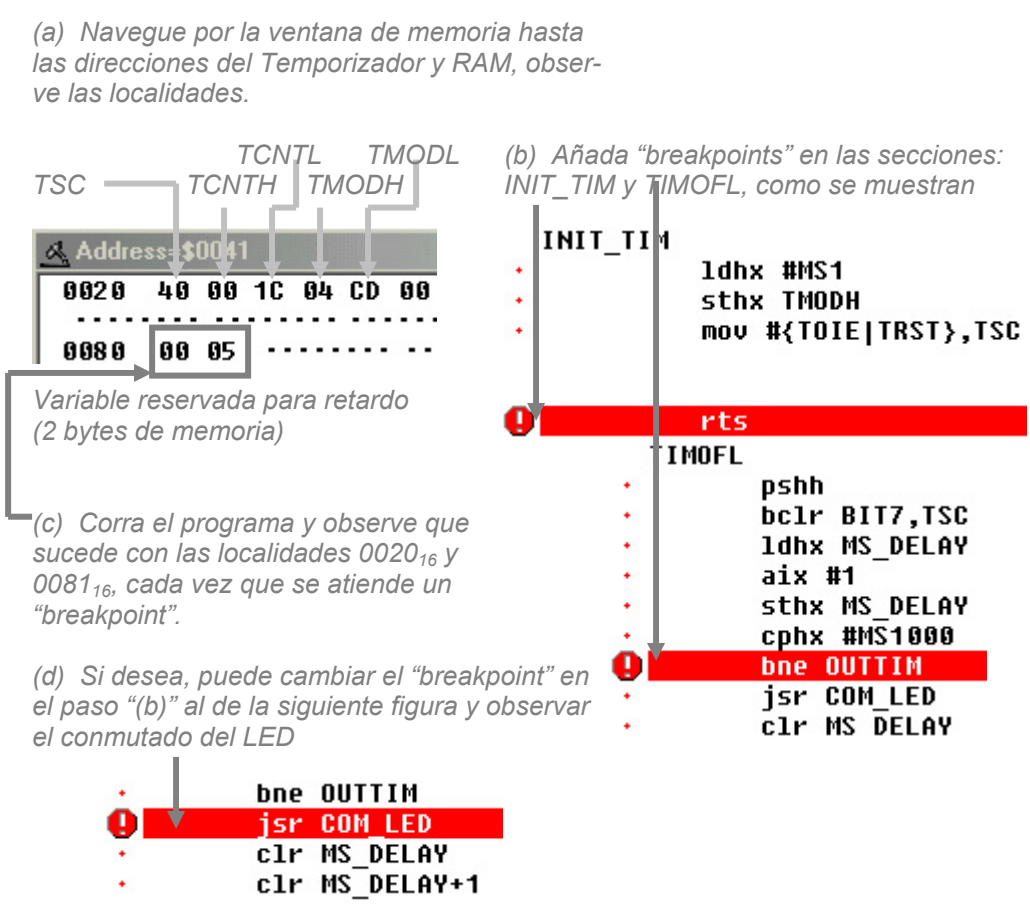


Figura 118. Simulación – NT0101. (a) Ventana de Memoria. Navegue y observe los registros principales del temporizador. (b) Ventana de código. Utilice los "breakpoints" para detener el flujo del programa en la rutina de inicialización e interrupción. (c) Ventana de Memoria – variable de retardo. Al darse la interrupción observe la variación de la variable de retardo. (d) Ventana de Código – Interrupción. Si se desea saber cuando ocurre el conmutado del LED, puede cambiarse el breakpoint de la interrupción sobre la rutina de conmutado del LED.

(a) Inicie WinIDE.

(b) Cargue el archivo NT0101 – Timer – Temporizador 24 03 04.asm

(c) Si desea simular, solo compile.
(c.1) Inicie el simulador
(c.2) Situe "breakpoints" en las rutinas de iniciación del temporizador e interrupción.
(c.3) Corra la simulación
(c.4) Al detenerse en el "breakpoint", observe el cambio en el registro del temporizador.
(c.5) Nuevamente corra su aplicación y observe la rutina de interrupción como opera.
(c.6) Si desea obtener mayor claridad, puede correr su programa paso a paso.

(d) Si desea programar, cambie la línea de compilación condicional por:

```
;$SET    ICS08          ; ICS08 = 1, Vamos a simular en la pastilla
                        ; la velocidad de simulación es menor en la
                        ; PC.
$SETNOT  ICS08          ; ICS08 = 0, Vamos a programar la pastilla
                        ; la aplicación debe correr en tiempo real
```

(e) Compíle luego del cambio y proceda a programar su pastilla. Ver NT0009, sección 1.9.5.

2.1.8 Conclusión

El temporizador tiene como función generar retardos programables de "hardware", pero precisos en ejecución sin pérdidas de tiempo, el mismo es un contador libre de dieciseis (16) bits que opera bajo interrupciones.

Los registros básicos poseen acciones de control que temporizan el evento, mientras que la interrupción atiende el sobreflujo del canal. Fácilmente este registro es programable por una ecuación derivada que controla el período del temporizado.

En la simulación, se encarga de situar "breakpoints" para referir secciones importantes del programa como la inicialización y la interrupción del temporizador para observar el flujo del programa.

Finalmente, el programa realiza el mismo concepto de notas anteriores, pero utiliza el tiempo de pérdida de una rutina de retardo de "software" por una de "hardware", dejando libre al CPU para realizar cualquier otra acción de control.

2.1.9 Referencias

2.1.9.1 Temporizadores Integrados
(a) http://www.monografias.com/trabajos14/temporizador/temporizador.shtml

2.1.9.2 Manual de Referencia del Módulo del Temporizador
(a) http://www.freescale.com/files/microcontrollers/doc/ref_manual/TIM08RM.pdf

Págs. 39 – 54. Uso de los registros generales

2.1.9.3 Información Avanzada sobre el Microcontrolador
(a) http://www.freescale.com/files/microcontrollers/doc/data_sheet/MC68HC08JL3.pdf

Pág. 107 – 109 - Descripción Funcional
Pág. 115 – Interrupción de desborde del temporizador
Págs. 117 – 121. - Información de registros

2.1.9.4 Manual de Referencia del CPU
(a) http://www.freescale.com/files/microcontrollers/doc/ref_manual/CPU08RM.pdf

Pág. 188, Instrucción WAIT.

2.1.9.5 Página "web" sobre esta Nota Técnica
(a) http://www.geocities.com/issaiass/

2.1.10 Problemas Propuestos

2.1.10.1 Utilice el temporizador y los registros de entrada y salida para que el "jumper" conectado a PTB5 defina el inicio de una temporización de 1 segundo.

2.1.10.2 Temporice para 10 seg. de encendido y 3 seg. de apagado.

2.1.10.3 Genere un temporizado de 10 Hz de frecuencia (TMOD[H:L]) con un PS = 2. Visualice la salida en el LED PTD7.