Notas Técnicas de Uso y Aplicación

NT1006

3.6 INTERFACE A PANTALLAS MULTIPLEXADAS
DESPLIEGUE DE INFORMACION DEL ADC EN 3 PANTALLAS MULTIPLEXADAS

Preparado 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

3.6.1 Introducción

Anteriormente se tomó especial atención al control de una pantalla de LEDs del tipo visualizador de siete segmentos que se presta entre muchas situaciones a despliegue de información numérica. Con poco esfuerzo, puede utilizarse el mismo concepto y tomar varias pantallas para visualizar más caracteres.

La principal desventaja de utilizar pantallas de este tipo es su consumo. Según la hoja de datos del fabricante, a 25 °C, el consumo de cada pantalla puede alcanzar de 15 a 40 mA, dependiendo de la pantalla utilizada. Si se presume tener un grupo de 8 pantallas, en el mejor de los casos, su consumo será de 120 mA, suficiente como para que una aplicación de bajo consumo con pilas AA cese tempranamente.

Alternativamente se utiliza un método bastante efectivo y muy útil, denominado multiplexión, el mismo consiste en solo prender cada pantalla de momento mientras las otras permanecen apagadas, encender la siguiente y completar el ciclo. Así, se mejora el consumo. Como dato adicional, se debe encender la pantalla a una rata de parpadeo de 60 Hz.

La nota tratará de explicar los conceptos básicos de:

- Cálculo de la resistencia limitadora: calcular la resistencia limitadora del decodificador BCD a 7 segmentos.
- ¿Qué es un Multiplexor?: Nociones básicas de un selector de salidas.
- Hardware y Software: La nota termina con el esquemático y un software de aplicación que genera en tres (3) pantallas multiplexadas la visualización de la conversión del ADC.

3.6.2 Materiales

1. Microcontrolador: JL3 / JK3 / GP32
2. Tarjeta de Desarrollo TD68HC908JL3 o similar y su microcontrolador
3. Plantilla de proyectos: “Breadboard” JE27, Jameco Part No: 20811
4. Fuente de Poder
5. Alambres AWG 20
6. Pelador de Alambres
7. “Kit” de Aficionado: Pinzas del “Kit”, Jameco Part No: 99629
8. Integrado: Decodificador de BCD a 7 Segmentos, Jameco Part No.: 47790
   Multiplexor o Selector de Canales, Jameco Part No.: 46607
9. “Display”: Visualizador de 7 Segmentos, Jameco Part No.: 24731, MAN71
10. Resistores: Resistencias de 7 × 150Ω ¼ Watt, Chocolate, Verde, Negro.
    3 × 2.2kΩ ¼ Watt, Rojo, Rojo, Naranja
11. Transistores PNP: 3 x 2N2907, Jameco Part No.: 38279

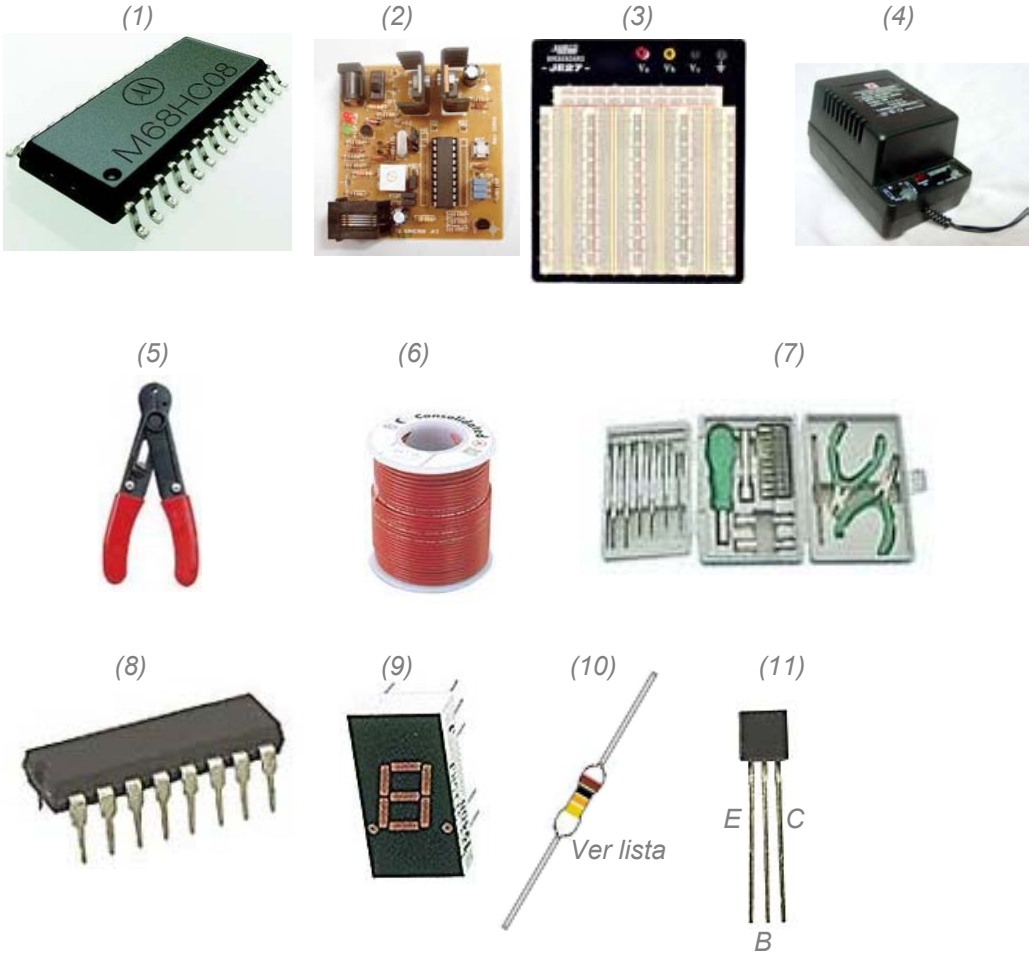


Figura 189. Listado de Materiales para Pantallas Multiplexadas

3.6.3 Descripción Física del "Hardware"

Para el siguiente párrafo, ayudarse por medio de la figuras de esta sección.

3.6.3.1 ¿Qué es un Multiplexor o Decodificador de Direcciones?

Imaginando un interruptor del cual se pueda seleccionar un canal de televisión diferente, un multiplexor, cuya abreviatura dada es MUX, tiene la misma función, selecciona una de las diversas señales de entrada y las pasa a la salida (ver figura 190(a)). En términos ingenieriles, un multiplexor o selector de acepta un conjunto de entradas y permite llevar de momento una a la salida; la entrada de selección, permite activar la salida correspondiente. En especial, el 74LS138 es un multiplexor que envía un estado bajo la salida correspondiente dependiendo del selector de entradas (ver tabla 75).

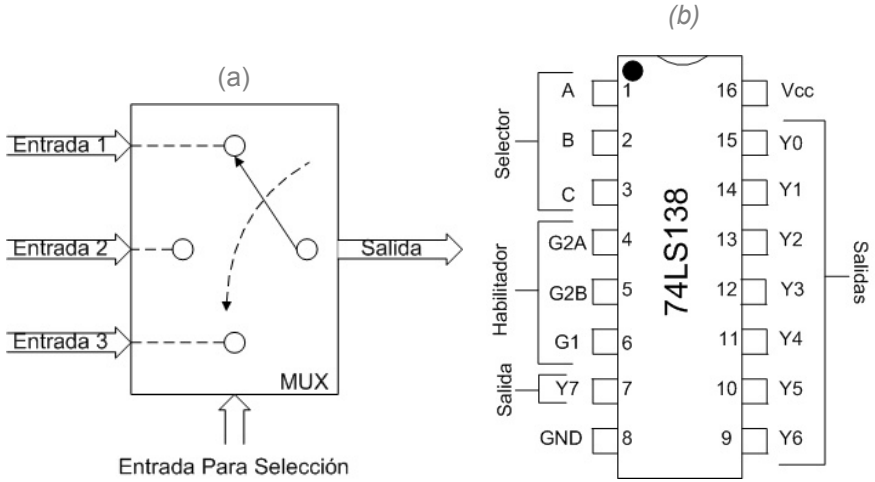


Tabla 75. Tabla de la Verdad del 74LS138

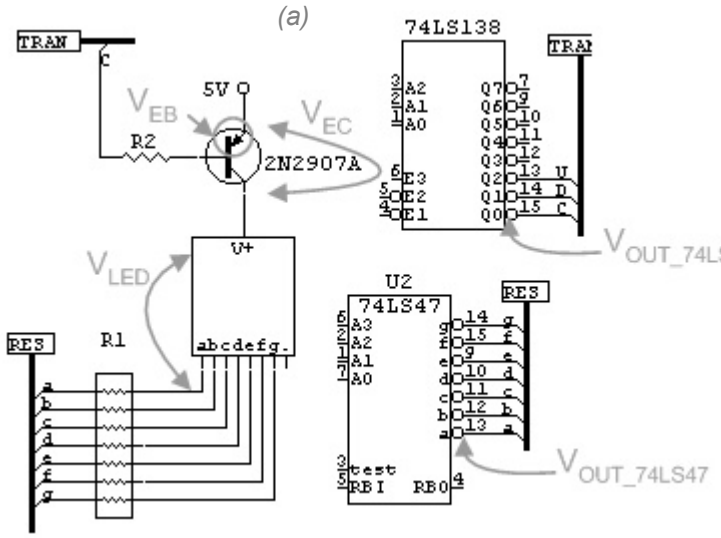
Entradas						Salidas							
G1	G2[A:B]		A	B	C	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7
X	H		X	X	X	H	H	H	H	H	H	H	H
L	X		X	X	X	H	H	H	H	H	H	H	H
H	L		L	L	L	L	H	H	H	H	H	H	H
H	L		L	L	H	H	L	H	H	H	H	H	H
H	L		L	H	L	H	H	L	H	H	H	H	H
H	L		L	H	H	H	H	H	L	H	H	H	H
H	L		H	L	L	H	H	H	H	L	H	H	H
H	L		H	L	H	H	H	H	H	H	L	H	H
H	L		H	H	L	H	H	H	H	H	H	L	H
H	L		H	H	H	H	H	H	H	H	H	H	L

H = Estado Alto, L = Estado Bajo, X = No importa.

Figura 190. Descripción de un Multiplexor. (a) Diagrama de Bloques. Un MUX o multiplexor toma una señal de entrada y la une a la salida, dependiendo del estado de la entrada de selección o selector. (b) "Hardware". Asignación de pines de un multiplexor 74 LS138. El estado de la entrada selectora decide el estado de la salida en estado bajo, todas las demás salidas de momento, quedarán en estado alto.

3.6.3.2 Cálculo de los Resistores Limitadores

La diferencia en controlar multiplexadamente pantallas a controlar una, está en su consumo, como fue mencionado en la introducción, estas pantallas consumen arriba de los 15 mA cada una y dependiendo de la cantidad a utilizar, si todas se mantienen encendidas a la vez, si consumo sube cerca o mayor a los 140 mA. Para evitar que nuestra fuente no entregue tanta cantidad de corriente, debemos encender cada pantalla a la vez a una rata de parpadeo de "n" × 60 Hz, en donde "n" es el número de pantallas a multiplexar, así el consumo total, será parecido al consumo de solo una pantalla. Brevemente se explica el cálculo de los resistores limitadores.



Calcule los resistores limitadores R_1 y R_2. Según sus hojas de datos (buscar la hoja de datos en http://www.jameco.com/ e insertar el número de parte dado en la lista de materiales):
V_{LED} es aproximadamente 1.7 V
$V_{EC(SAT)}$ es alrededor de 0.7 V
$V_{EB(SAT)}$ es de 1.3 V
Las corrientes de suministro escogidas: I_{BASE} = 1 mA e I_{LED} = 6 mA

$$R1 = \frac{(V_{CC} - V_{EC(SAT)} - V_{LED} - V_{OUT\_74LS47})}{N \times I_{LED}} \qquad R2 = \frac{(V_{CC} - V_{EB(SAT)} - V_{OUT\_74LS138})}{N \times I_{BASE}}$$

$$R1 = \frac{(5 - 0.4 - 1.7 - 0)V}{3 \times 6mA} \qquad R2 = \frac{(5 - 1.3 - 0)V}{3 \times 1mA}$$

$$R1 = 161\Omega \cong 150\Omega \qquad R2 = 1.23k\Omega \cong 1k\Omega$$

(b) (c)

Figura 191. Cálculo de Resistores Limitadores. (a) Sección de Esquema de Aplicación. Para el cálculo de componentes es necesario saber los valores dados por el fabricante, utilizando los valores para el peor de los casos. (b) Cálculo del Resistor R_1. El resultado depende de solo un LED y la corriente que se desea circule por el, para el caso, un resistor alrededor de 150Ω es suficiente. (c) Cálculo del Resistor R_2. Según los resultados, para conmutar el transistor, se puede colocar un resistor con un valor desde $1k\Omega$.

3.6.4 Esquemático de Aplicación

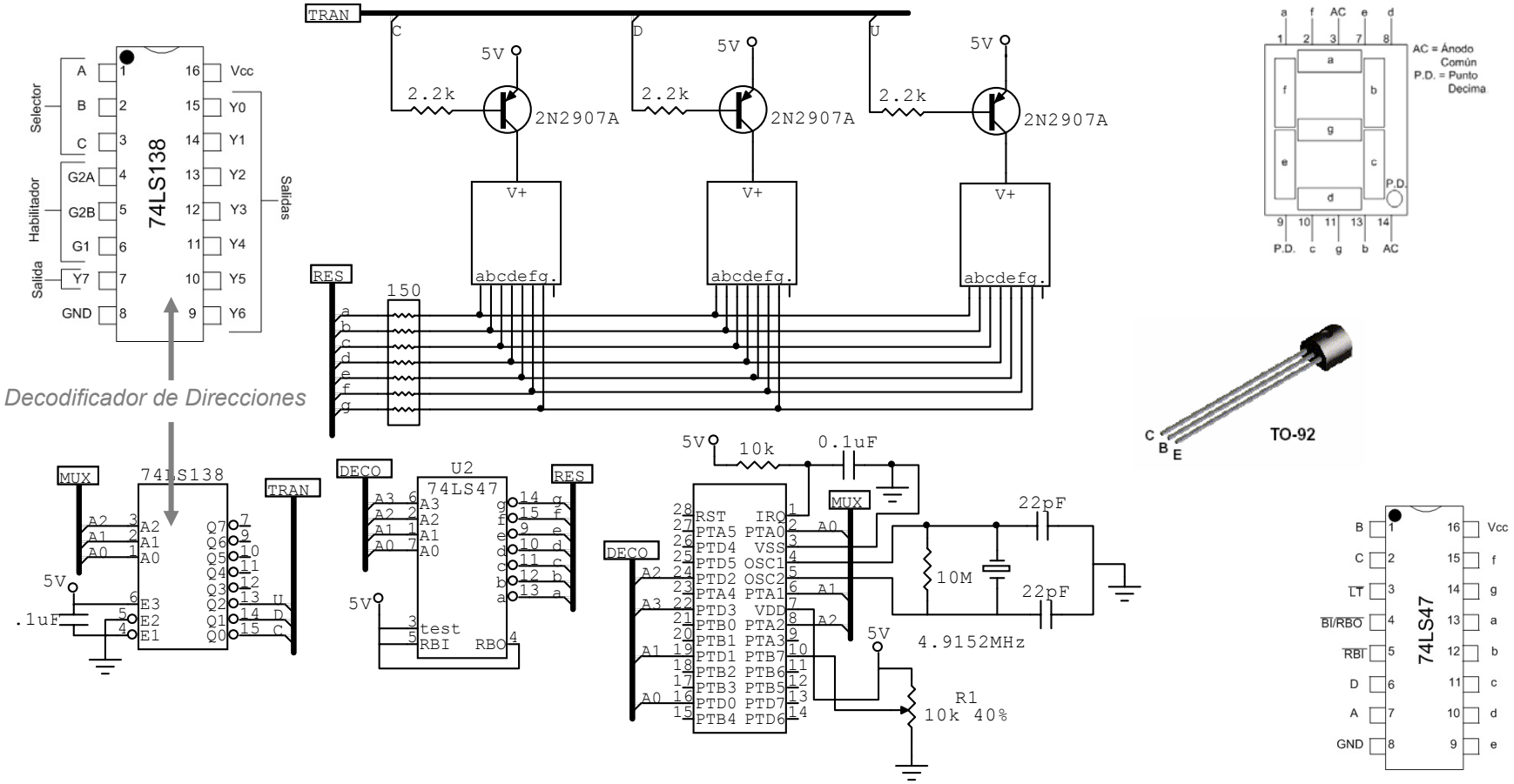


Figura 192. Esquema de Aplicación del Manejo de Siete Segmentos Multiplexado. Para la multiplexión de pantallas se utiliza un decodificador de direcciones 74LS138, de manera tal este encendieda una pantalla a su vez. Las pantallas deben refrescarse a una rata de "n" × 60 Hz, donde "n" es el número de pantallas a multiplexar. Adicionalmente se controla este esquema para visualizar en tres siete segmentos la información arrojada por el convertidor analógico digital.

3.6.5 Diagrama de Flujo

El siguiente programa utiliza un Decodificador de Direcciones (74LS138) y un Decodificador de BCD a Siete Segmentos (74LS47) para realizar el multiplexado de tres pantallas a 180 Hz (60 Hz c/u) y desplegar la variación del resultado de la conversión del ADC7 (PTB7). Si desea conocer en detalles como se hizo esta nota ud. necesita de los documentos básicos, NT0011, NT0011, NT1005 y NT010; Apéndices B, C, I, M y N.

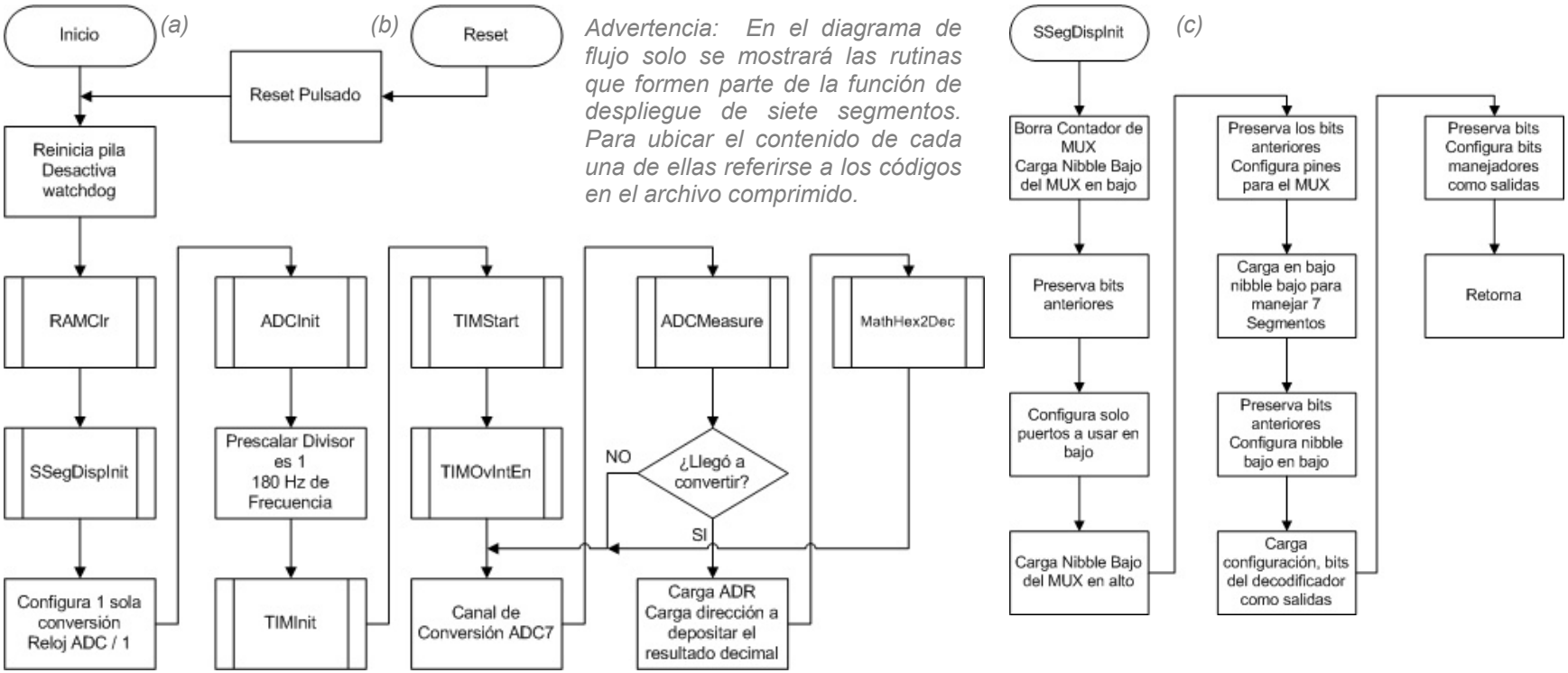


Figura 193. NT1006 – SSEGMUXD. (a) Programa Principal. Inicia el convertidor en el PTB7 y un temporizador a 180 Hz, el cual refrescará cada una de las tres pantallas a 60 Hz. Adicionalmente, se convierte el resultado del ADC a enteros de 0 a 9 que representan el valor escalado en decimal de la variación del potenciómetro, de 0 a 255. (b) Reinicio del Sistema. Al presionar el botón "Reset", sin importar lo que suceda de momento, el sistema saltará nuevamente al inicio. (c) SSegDispInit. Inicializa la pantalla de Siete Segmentos, carga el primer valor de la tabla y activa la primera pantalla.

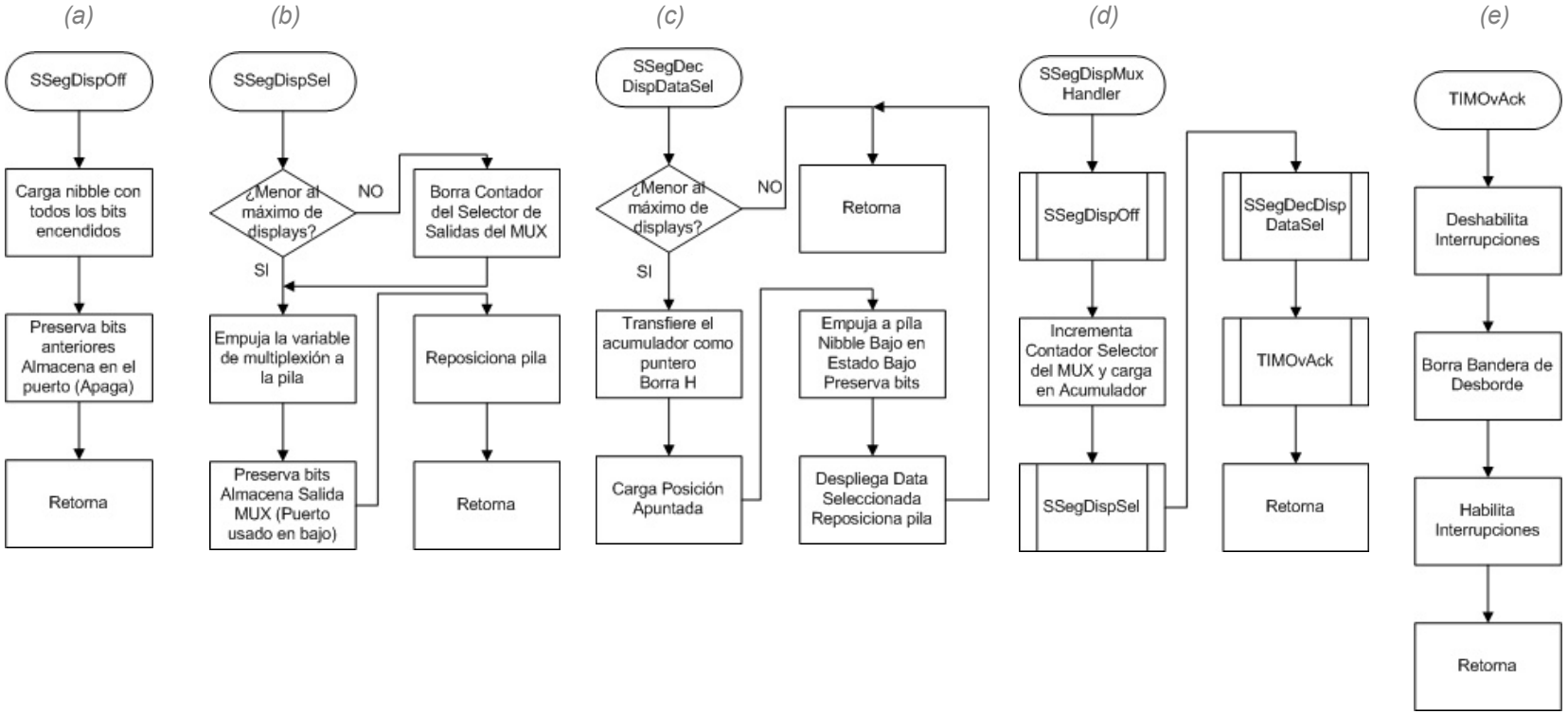


Figura 194. NT1006 – SSEGMUXD – Continuación. (a) SSegDispOff. Apaga el siete segmentos, envía a las entradas del decodificador de BCD a 7 Segmentos unos (1s) lógicos. (b) SSegDispSel. Selecciona la salida del multiplexor, multiplexa a la siguiente pantalla. (c) SSegDecDispDataSel. Selecciona el caracter decimal a desplegar en la pantalla, envía un valor BCD al decodificador 74LS47. (d) SSegDispMuxHandler. Apaga la pantalla y selecciona la siguiente pantalla e información a desplegar, finalmente, reconoce la interrupción del temporizador. (e) TIMOvAck. Reconoce la interrupción del temporizador.

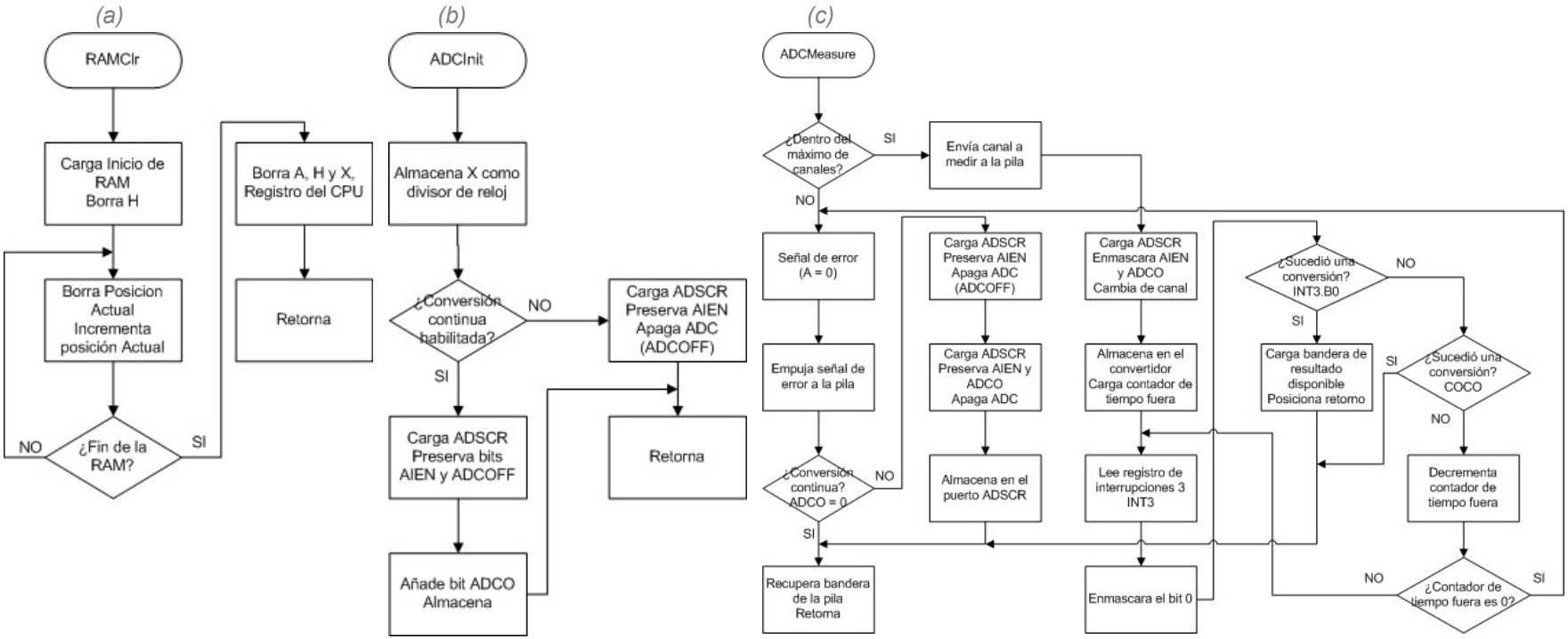


Figura 195. NT1006 – SSEGMUXD – Continuación. (a) RAMClr. Borra todos los bytes de la memoria RAM y los registros del CPU. (b) ADCInit. Rutina que inicializa el ADC dado el divisor y el tipo de conversión. (c) ADCMeasure. Rutina que mide un canal del ADC y retorna una bandera que simboliza si la conversión fue completa o no.

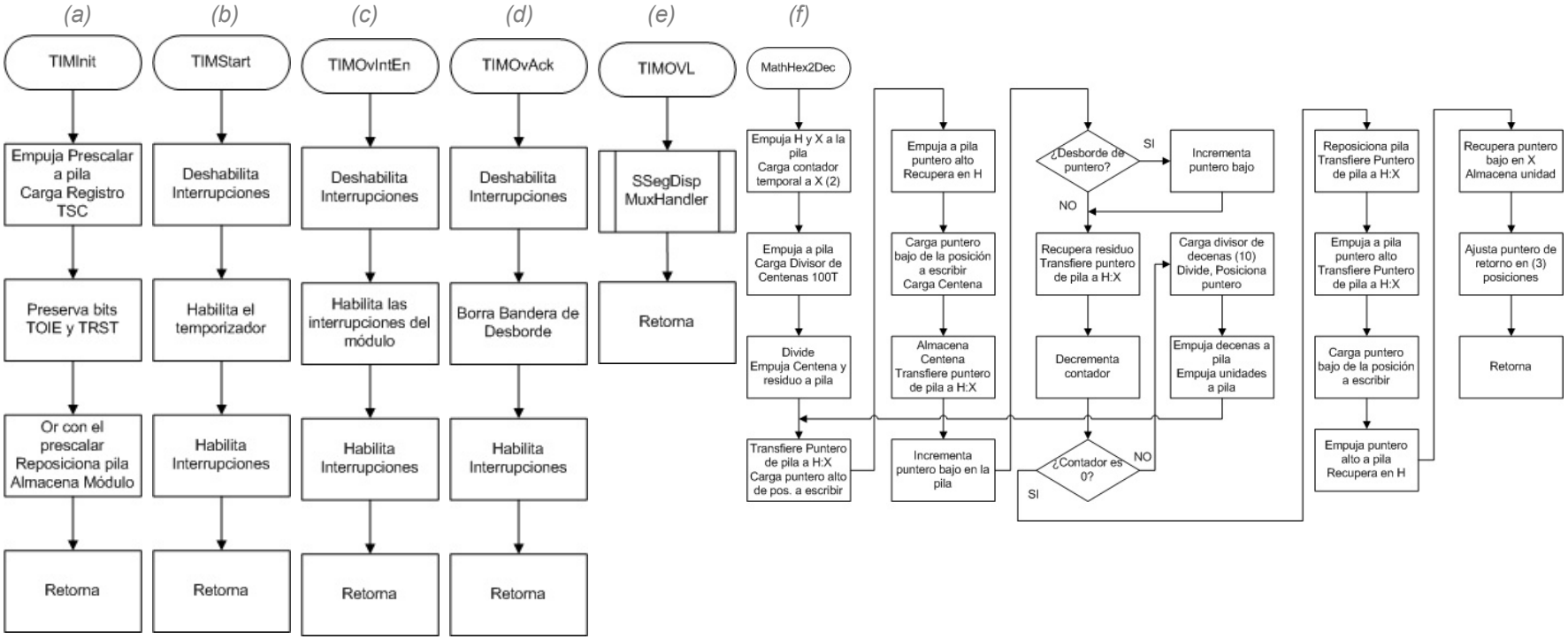


Figura 196. NT1006 – SSEGMUXD – Continuación. (a) TIMInit. Inicializa el temporizador dado el divisor preescalar y el módulo del contador. (b) TIMStart. Rutina que arranca la cuenta del módulo temporizador. (c) TIMOvIntEn. Habilita las interrupciones de sobreflujo del temporizador. (d) TIMOvAck. Rutina que reconoce la interrupción del temporizador. (e) TIMOVL. Subrutina de interrupción que se encarga de multiplexión. (f) MathHex2Dec. Rutina que transforma un número hexadecimal de ocho bits a su equivalente decimal y descarga en una posición especificada.

3.6.6 Código

Advertencia, del siguiente programa solo se listará las partes necesarias para que la rutina opere de manera apropiada, si desea ver el resto del contenido de un archivo cabecera incompleto, dirigirse al archivo comprimido. También, referirse a las notas, NT0011, NT0011, NT1005 y NT010; Apéndices B, C, I, M y N para un mejor entendimiento.

```
;===================================================================
; ARCHIVO      : NT1006 - SSEGMUXD - 03 01 05.asm
; PROPÓSITO    : Adquiere una señal del ADC7 y la despliega en 3 siete seg-
;                mentos multiplexando la información, adicionalmente, existe
;                un temporizador que multiplexa los displays a una rata de
;                N*60 Hz, en donde N es el número de pantallas.
; NOTAS        : Ninguna
;
;
;
; REFERENCIA:  NT1006 - SSEGMUXD - 03 01 05.doc
;
; LENGUAJE     : IN-LINE ASSEMBLER
; ------------------------------------------------------------------------
; HISTORIAL
;   DD MM AA
;   26 12 04       Creado.
;   03 01 05       Modificado.
;===================================================================

;=========================================
;            Cabecera de Macros, Const. y Memoria
;=========================================
$include '\MAP\includes.equ'                ; Definiciones de usuario y mapa de memoria
```

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

;===========================================
; OBJETIVO     :  Convierte el valor Hexadeci
;                 mal resultado de la conver-
;                 sión en Decimal y despliega
;                 en 3 siete segmentos.
;===========================================
START
        rsp                                 ; Inic.Stack = $00ff
        bset BIT0,CONFIG1                   ; Desactiva Watchdog
        jsr RAMClr                          ; Borra RAM y registros
        jsr SSegDispInit                    ; Inicializa pantalla
        clra                                ; Una sola conversión
        clrx                                ; ADICLK = XTAL/(2^(0+2))
        jsr ADCInit                         ; Inicializa ADC
        clra                                ; DIV1
        ldhx #$1AAA                         ; Desborde a 3*60 = 180 Hz
        jsr TIMInit                         ; Inicializa TIM
        jsr TIMStart                        ; Inicia temporizado
        jsr TIMOvIntEn                      ; Habilita interrupciones
LEDAGAIN0101.AA
        lda #7T                             ; ADC7 = PTB7
        jsr ADCMeasure                      ; Mide el ADC
        beq LEDAGAIN0101.AA                 ; ¿No hubo medición?, salir
        lda ADR                             ; Carga Resultado de conversión
        ldhx #SSegDecDispData               ; Lugar a depositar la información
        jsr MathHex2Dec                     ; Transforma a Decimal
        ldhx #SSegDecDispData               ; Carga lugar a depositar resultado decimal
        bra LEDAGAIN0101.AA                 ; Captura una nueva conversión

;===========================================
;             Declaración y definición de funciones
;===========================================
$include '\FUNCTIONS\INCLUDES.inc' ; Incluye Funciones

;===========================================
;           Declaración y Definición de interrupciones
;===========================================
$include '\INTERRUPTS\interrupt.inc'    ; Incluye interrupciones del microcontrolador
```

Listado 39. NT1006 – SSEGMUXD. El programa inicia el temporizador a 180 Hz y captura las conversiones del ADC, PTB7, para desplegarlo en tres (3) siete segmentos

```
;=================================================================
; ARCHIVO      : SSEG.inc
; PROPÓSITO    : Librería para utilidad de Siete segmentos.
; LENGUAJE     : IN-LINE ASSEMBLER
; NOTAS        : UTILIZA UNA ESTRUCTURA EN RAM QUE UBICA LA
;                INFORMACIÓN A DESPLEGAR EN LOS SIETE SEGMENTOS.
;                UTILIZAR LA RUTINA "SSEGMUXHANDLER" CON UN
;                TEMPORIZADOR A
;                N_DISPLAYS * 60 Hz.
; ----------------------------------------------------------------
; HISTORIAL
;  DD MM AA
;  26  05  03   Creado.
;  25  12  04   Modificado.
;=================================================================
; INICIALIZACIÓN
;          : Configurar...
;                1 - El nibble para el multiplexor o el decodificador de
;                    teclado:
;                    $SET, Selecciona la configuración y uso del nibble alto
;                    $SETNOT, Selecciona el nibble bajo
;                2 - Las características de operación del Multiplexor
;                        2.1 - Puerto de Multiplexión = SSEG_MUX_PORT
;                        2.2 - Registro del Puerto del Multiplexor = SSEG_MUX_DDR
;                        2.3 - Máximo de caracteres a multiplexar = SSEG_MAX_DISP
;                3 - Las características del Decodificador de BCD-7Segmentos
;                        3.1 - Puerto del Decodificador =
;                              SSEG_DEC_PORT
;                        3.2 - Registro del Puerto del Decodificador =
;                              SSEG_DEC_DDR
;                4 - La estructura de despliegue de información Ubicando la
;                        Dirección en la memoria RAM de:
;                        4.1 - Variable de control del Multiplexor = SSegMuxCtr
;
; ADVERTENCIA :        La estructura en RAM, depende de la ubicación de LedCtr
;                      y usa "el número de caracteres a multiplexar" + 2, definida
;                      por la ubicación (es decir, empezando) por la variable
;                      SSegMuxCtr.
;                      LA RUTINA "SSEGMUXHANDLER", DEBE IR
;                      ADMINISTRADA POR UN TEMPORIZADOR INICIADO A
;                      "N * 60 Hz", EN DONDE "N" ES LA CANTI-
;                      DAD DE 7 SEGMENTOS A UTILIZAR.
;
;=================================================================

;$SET    MUX_NIBBLE_HI          ; Configura para nibble alto del puerto de MUX
;$SET    DEC_NIBBLE_HI          ; Configura para nibble alto del puerto de DEC
$SETNOT MUX_NIBBLE_HI          ; Configura para nibble bajo del puerto de
                               ; MUX
$SETNOT DEC_NIBBLE_HI          ; Configura para nibble bajo del puerto de
                               ; DEC
```

```
;===========================================
;                  Constantes & Macros
;===========================================
;===========================================
;            DEFINA PUERTO DE MULTIPLEXIÓN
;===========================================
SSEG_MUX_PORT     equ PORTA     ; Puerto del Multiplexor
SSEG_MUX_DDR      equ DDRA      ; Registro del Puerto del Multiplexor
SSEG_MAX_DISP     equ 3T        ; Máximo de caracteres a multiplexar

;===========================================
;              DEFINA PUERTO DE CONTEO
;===========================================
SSEG_DEC_PORT     equ PORTD     ; Puerto del Decodificador
SSEG_DEC_DDR      equ DDRD      ; Registro del Puerto del Decodificador

;===========================================
; DEFINA LUGAR DE LA ESTRUCTURA DE DESPLIEGUE
;===========================================
SsegMuxCtr            equ $80         ; Variable de control del Multiplexor
SsegDecDispData       equ SSegMuxCtr+1
                                      ; Inicio de data a desplegar en 7 segmentos
SsegDecDispDataTop    equ {SSegDecDispData + SSEG_MAX_DISP}
                                      ; Fin de la data a desplegar en 7 segmentos

;===========================================
;  CONSTANTES DE MULTIPLEXOR Y DECODIFICADOR
;===========================================
SSEG_MUXNIBHIOFF  equ $8F       ; Nibble Alto en estado bajo
SSEG_MUXNIBLOOFFequ $F8         ; Nibble Bajo en estado alto
SSEG_MUXNIBHIOUT equ $70        ; Nibble Alto son salidas
SSEG_MUXNIBLOOUTequ $07         ; Nibble Bajo son salidas

SSEG_DECNIBHIOFF  equ $0F       ; Nibble Alto en estado bajo
SSEG_DECNIBLOOFF equ $F0        ; Nibble Bajo en estado alto
SSEG_DECNIBHIOUT equ $F0        ; Nibble Alto son salidas
SSEG_DECNIBLOOUTequ $0F         ; Nibble Bajo son salidas
SSEG_DECNIBHIHI   equ $F0       ; Nibble Alto en alto
SSEG_DECNIBLOHI   equ $0F       ; Nibble Bajo en alto
```

```
;===========================================
; SSEGDISPINIT
;                 : Inicializa el módulo de 7
;                   Segmentos, Leds.
; OBJETIVO        : Inicializa el módulo de
;                   multiplexión de 7 segmentos,
;                   leds
; ENTRADA         : Ninguna
; SALIDA          : Ninguna
; REGISTROS
; AFECTADOS       : ACCA, SSEG_MUX_PORT,
;                   SSEG_MUX_DDR, SSEG_DEC_PORT,
;                   SSEG_DEC_DDR, SsegMuxCtr
;                   (STRU)
; NOTAS           : Asegúrese de inicializar
;                   correctamente el número de
;                   displays a multiplexar.
;===========================================
SSegDispInit
        clr SSegMuxCtr                  ; Borra contador del multiplexor
$IF  MUX_NIBBLE_HI
        lda #SSEG_MUXNIBHIOFF           ; Nibble Alto en estado bajo
$ELSEIF
        lda #SSEG_MUXNIBLOOFF           ; Nibble Bajo en estado bajo
$ENDIF
        and SSEG_MUX_PORT               ; Preservar bits
        sta SSEG_MUX_PORT               ; Puertos a usar en bajo
$IF  MUX_NIBBLE_HI
        lda #SSEG_MUXNIBHIOUT           ; Nibble alto son salidas
$ELSEIF
        lda #SSEG_MUXNIBLOOUT           ; Nibble bajo son salidas
$ENDIF
        ora SSEG_MUX_DDR                ; Preserva bits
        sta SSEG_MUX_DDR                ; Puertos a usar son salidas
$IF  DEC_NIBBLE_HI
        lda #SSEG_DECNIBHIOFF           ; Nibble alto en bajo
$ELSEIF
        lda #SSEG_DECNIBLOOFF           ; Nibble bajo en bajo
$ENDIF
        and SSEG_DEC_PORT               ; Preserva los bits anteriores
$IF  DEC_NIBBLE_HI
        ora {SSegDecDispData < 4}       ; Carga el primer dato a desplegar
$ELSEIF
        ora SSegDecDispData             ; Carga el primer dato a desplegar
$ENDIF
        sta SSEG_DEC_PORT               ; Configura los bits del puerto a usar
                                        ; en bajo
$IF  DEC_NIBBLE_HI
        lda #SSEG_DECNIBHIOUT           ; Configura salidas del nibble alto a
                                        ; usar
$ELSEIF
        lda #SSEG_DECNIBLOOUT           ; Configura salidas del nibble bajo a
```

```
                                        ; usar
                                        ; Carga nibble
$ENDIF
        ora SSEG_DEC_DDR                ; Preserva bits
        sta SSEG_DEC_DDR                ; Configura bits de Decodificador
                                        ; como salidas
        rts                             ; SI, Salir

;=============================================
; SSEGDISPOFF
;               : Apaga el display de siete
;                 Segmentos
; OBJETIVO      : Envía la data $0F al display
;                 para apagar todos los seg-
;                 mentos.
; ENTRADA       : Ninguna
; SALIDA        : Ninguna
; REGISTROS
; AFECTADOS     : ACCA, SSEG_DEC_PORT
;=============================================
SSegDispOff
$IF  DEC_NIBBLE_HI
        lda #SSEG_DECNIBHIHI            ; Nibble alto en alto
$ELSEIF
        lda #SSEG_DECNIBLOHI            ; Nibble bajo en alto
$ENDIF
        ora SSEG_DEC_PORT               ; A apagar los displays
        sta SSEG_DEC_PORT               ; Almacena en el puerto
        rts                             ; Apaga los displays
```

```
;=============================================
; SSEGDISPSEL
;                 : Selecciona el Display a ser
;                   encendido
; OBJETIVO        : Enciende el display actual
;                   dado el número del display
; ENTRADA         : ACCA    contiene el número
;                   del canal actual
;                   a multiplexarse
; SALIDA          : Ninguna
; REGISTROS
; AFECTADOS       : ACCA, SSegMuxCtr (RAM) , SP
;=============================================
SSegDispSel
        cmp #SSEG_MAX_DISP          ; Compara con el máximo de displays
                                    ; a multiplexar
        blo SSEGCTRCLR0102.B        ; ¿Faltan por multiplexar?, SI, Saltar
        clr SsegMuxCtr              ; No, reiniciar multiplexión
        clra                        ; contador reiniciado
SSEGCTRCLR0102.B
$IF  MUX_NIBBLE_HI
        nsa                         ; Cambia nibble bajo por alto
$ENDIF
        psha                        ; Empuja variable de multiplexión
$IF  MUX_NIBBLE_HI
        lda #SSEG_MUXNIBHIOFF       ; Cambia las líneas del multiplexor,
                                    ; nibble alto, por 0s
$ELSEIF
        lda #SSEG_MUXNIBLOOFF       ; Cambia las líneas del multiplexor,
                                    ; nibble bajo, por 0s
$ENDIF
        and SSEG_MUX_PORT           ; Preserva bits
        ora 1,SP                    ; Suma contenido del puerto
        sta SSEG_MUX_PORT           ; Puertos a usar en bajo
        pula                        ; Reposiciona pila
        rts                         ; Retorna
```

```
;===========================================
; SSEGDECDISPDATASEL
;                 : Selecciona data a desplegar
; OBJETIVO        : Despliega la información exis
;                   tente en la estructura de
;                   RAM.
; ENTRADA         : ACCA    es el puntero de la
;                   información del men-
;                   saje a desplegar
; SALIDA          : Ninguna
; REGISTROS
; AFECTADOS       : ACCA, SSegMuxCtr (RAM)
;===========================================
SSegDecDispDataSel
        cmp #SSEG_MAX_DISP          ; Compara con el máximo de pantallas
        bge SSEGOUT0102.C           ; ¿Llegó al máximo?
        tax                         ; Transfiere comp puntero
        clrh                        ; Borra H
        lda SSegDecDispData,x       ; Carga tabla
$IF  DEC_NIBBLE_HI
        nsa                         ; Cambia nibble bajo por alto
$ENDIF
        psha                        ; Empuja caracter a desplegar
                                    ; a la pila
$IF  DEC_NIBBLE_HI
        lda #SSEG_DECNIBHIOFF       ; Nibble alto en estado bajo
$ELSEIF
        lda #SSEG_DECNIBLOOFF       ; Nibble bajo en estado bajo
$ENDIF
        and SSEG_DEC_PORT           ; Preserva bits
        ora 1,SP                    ; Almacena
        sta SSEG_DEC_PORT           ; Almacena en el puerto
        pula                        ; Reposiciona pila
SSEGOUT0102.C
        rts                         ; Retorna
```

```
;===============================================
; SSEGDISPMUXHANDLER
;                 : Multiplexa el display de
;                   siete segmentos.
; OBJETIVO        : Decide el display a encender
;                   y reconoce la interrupción
;                   del temporizador.
; ENTRADA         : Ninguna
; SALIDA          : Ninguna
; REGISTROS
; AFECTADOS       : ACCA, SSegMuxCtr (RAM)
;===============================================
SSegDispMuxHandler
        jsr SSegDispOff             ; Apaga el display
        inc SSegMuxCtr              ; Carga el display a encender
        lda SSegMuxCtr              ; Carga contador
        jsr SSegDispSel             ; Selecciona el display a encender
        jsr SSegDecDispDataSel      ; Selecciona el número a desplegar
        jsr TIMOvAck                ; Reconoce la interrupción del
                                    ; temporizador
        rts                         ; Retorna
```

Listado 40. NT1006 – SSEGMUXD – SSEG.inc. Archivo de funciones de Manejo de Siete Segmentos con un decodificador de BCD a Siete Segmentos 74LS47 y un Decodificador de Direcciones 74LS138.

```
;=================================================================
; ARCHIVO      : ADC.inc
; PROPÓSITO    : Inclusión de la función de inicialización y medición de ADC
; LENGUAJE     : IN-LINE ASSEMBLER
; ----------------------------------------------------------------------
; HISTORIAL
;   DD MM AA
;   30  08  04   Creado.
;   30  08  04   Modificado.
;=================================================================

;=========================================
;                    Constantes & Macros
;=========================================
ADC_MAX_CH      equ 12T             ; Máximo número de canales del MUX
ADC_TO          equ 15T             ; Tiempo fuera de Conversión

;=========================================
; ADCINIT      : Inicializa el convertidor
; OBJETIVO     : Inicializa el ADC con su di-
;                visor y tipo de conversión
; ENTRADA      : A  es el tipo de conversión
;                X  contiene el valor del
;                divisor del reloj de en-
;                trada del ADC
; SALIDA       : Ninguna
; REGISTROS
; AFECTADOS    : ADSCR, ADICLK, ACCA, X
;=========================================
ADCInit
        stx ADICLK                  ; Almacena divisor del ADC
        cmp #1                      ; ¿A habilitar conv. continua?
        beq ADC0011.E               ; Si no es igual, apagar ADC
        lda ADSCR                   ; Carga el ADSCR
        and #AIEN                   ; Habilita interrupciones
        ora #ADCOFF                 ; Apaga el ADC
        bra ADC0011.F               ; Salir
ADC0011.E
        lda ADSCR                   ; A = ADSCR
        and #{AIEN|ADCOFF}          ; Preserva bits de int. y canal
        ora #ADCO                   ; añade bit ADCO = 1
ADC0011.F
        sta ADSCR                   ; Almacena en el registro del ADC
        rts                         ; Retorna
```

```
;=============================================
; ADCMEASURE
;               : Decide si el canal es hábil
; OBJETIVO      : Canal disponible o no
; ENTRADA       : A  contiene el canal a medir
; SALIDA        : A  decide si el canal está
;                  disponible o no
; REGISTROS
; AFECTADOS     : ADSCR, ADICLK, ACCA, X
; NOTA          : SALIDA = 0, Señal de error
;                  SALIDA = 1, Lectura Dispo-
;                  nible
;=============================================
ADCMeasure
        cmp #ADC_MAX_CH                 ; ¿Dentro del rango de multiplexión?
        bge ADCOUT0011.D                ; NO, salir
        psha                            ; Envía a la pila
        lda ADSCR                       ; Carga Registro ADC
        and #{AIEN|ADCO}                ; Enmascara los bits altos
        ora 1,SP                        ; Cambia de canal
        sta ADSCR                       ; Almacena en el convertidor
        ldx #ADC_TO                     ; Contador de Tiempo Fuera
ADCLOOP0011.B
        lda INT3                        ; leer INT3
        bit #BIT0M                      ; Enmascara BIT0
        bne AREAD0011.A                 ; Hay interrupción, leer
        brset COCO,ADSCR,AREAD0011.A    ; Si termina la conversión, leer
        decx                            ; ADC_TO = ADC_TO - 1 (TO =
                                        ; TimeOut)
        cpx #0                          ; ¿contador = 0?
        bne ADCLOOP0011.B               ; Salta y pregunta nuevamente
        ais #1                          ; Posiciona la dirección de retorno
ADCOUT0011.D
        clra                            ; Señal de error
        bra ADCOUT0011.C                ; salir del ADC
AREAD0011.A
        lda #1                          ; Resultado Disponible
        ais #1                          ; Posiciona la dirección de retorno
ADCOUT0011.C
        psha                            ; empuja a la pila
        brset BIT5,ADSCR,ADC0011.I      ; Si ADCO = 0, salir
        lda ADSCR                       ; Carga ADSCR
        and #{AIEN|ADCO}                ; Preserva bits
        ora #ADCOFF                     ; apaga el ADC
        sta ADSCR                       ; Almacena en el registro
ADC0011.I
        pula                            ; recupera de la pila
        rts                             ; Retorna
```

Listado 41. NT1006 – SSEGMUXD – ADC.inc. Archivo de funciones de ADC, solo se listan las funciones utilizadas ADCInit que inicia el ADC y ADCMeasure que mide un canal.

```
;*****************************************************************
; ARCHIVO      : RAM.inc
; PROPÓSITO    : Funciones de uso de la RAM
; NOTAS        : ADVERTENCIA - Se debe especificar el inicio y el origen de
;                RAM de su microcontrolador por medio de los
;                macros RAM_ORG y RAM_END
;
; LENGUAJE     : IN-LINE ASSEMBLER
; -------------------------------------------------------------------
; HISTORIAL
;  DD MM AA
;  05  10  04   Creado.
;  06  10  04   Modificado.
;*****************************************************************

;*****************************************
;                CONSTANTES & MACROS
;*****************************************
RAM_ORG      equ $0080                  ; Inicio de la memoria RAM
RAM_END      equ $00FF-1                ; Fin de limpieza de la RAM

;*****************************************
; RAMCLR       : Borra la Ram utilizada y re-
;                gistros inherentes
; OBJETIVO     : Borra registros
; ENTRADA      : Ninguna
; SALIDA       : A, H:X y RAM en 0
; REGISTROS
; AFECTADOS    : RAM, H:X, A
;*****************************************
RAMClr                                  ; Borra la RAM y registros
        clrh                            ; Borra H
        ldx #RAM_ORG                    ; Carga con el origen
RAM_EMPTY0000.A
        clr ,x                          ; rellena con "0" la posición actual
        aix #1                          ; incrementa puntero de RAM
        cphx #RAM_END                   ; Compara hasta el final deseado
        bne RAM_EMPTY0000.A             ; Si no concuerda entonces sigue
                                        ; limpiando
        clra                            ; Borra A
        clrh                            ; Borra H
        clrx                            ; Borra X
        rts                             ; retorna
```

Listado 42. NT1006 – SSEGMUXD – RAM.inc. Archivo de funciones de RAM, borra el contenido de la RAM.

```
;*******************************************************************
; ARCHIVO      : MATH.inc
; PROPÓSITO    : Librería para ruinas matemáticas.
; LENGUAJE     : IN-LINE ASSEMBLER
; NOTAS        : Ninguna
; -------------------------------------------------------------------
; HISTORIAL
;   DD MM AA
;   26  12  04   Creado.
;   26  12  04   Modificado.
;*******************************************************************

;===========================================
; MATHHEX2DEC
;               : Convierte un byte a Decimal
; OBJETIVO      : Convierte un caracter hexa-
;                 decimal a su correspondiente
;                 decimal.
; ENTRADA       : ACCA   es el caracter a
;                 convertir a decimal
;                 HX       puntero donde se ubi
;                          ca el resultado de
;                          la información
; SALIDA        : 3 Posiciones correspondientes
;                 al número equivalente son
;                 depositadas desde el ingreso
;                 del registro H:X.
;                 Una para las CENTENA
;                 La siguiente para las DECENA
;                 La siguiente para las UNIDAD
; REGISTROS
; AFECTADOS     : ACCA, H:X, SP, RAM
;===========================================
MathHex2Dec
        sei                             ; Inhabilita interrupciones
        pshx                            ; Empuja byte bajo
        pshh                            ; Empuja byte alto
        ldx #2T                         ; Carga contador temporal
        pshx                            ; Empuja a pila
        ldx #100T                       ; CCCA dividir Centenas
        div                             ; Divide
        psha                            ; Empuja Centena a la pila
        pshh                            ; Empuja Residuo a pila
MATHNEXT0126.B
        tsx                             ; Transfiere SP a H:X
        lda 3,x                         ; Carga Puntero Alto de la posición a
                                        ; escribir
        psha                            ; Empuja a pila puntero alto
        pulh                            ; Recupera en H para la posición a
                                        ; escribir
        ldx 4,x                         ; Carga Puntero Bajo de la posición a
                                        ; escribir
```

```
        lda 2,SP                    ; Carga Centena
        sta ,x                      ; Almacena Centena
        tsx                         ; Transfiere SP a H:X
        inc 4,x                     ; Incrementa puntero bajo en la pila
        tst 4,x                     ; Comprueba si es cero
        bne MATHNEXT0126.A          ; ¿0?, NO, Seguir
        inc 3,x                     ; SI, Incrementar puntero alto
MATHNEXT0126.A
        pula                        ; Recupera Residuo
        tsx                         ; Transfiere SP a H:X
        dec 1,x                     ; Decrementa contador
        tst 1,x                     ; Prueba si es 0
        beq MATHOUT0126.C           ; Salir si es cero
        ldx #10T                    ; A dividir Decenas
        div                         ; Divide
        ais #1T                     ; Reposiciona puntero
        psha                        ; Empuja Decenas a la pila
        pshh                        ; Empuja Unidades a la pila
        bra MATHNEXT0126.B          ; Siguiente caracter
MATHOUT0126.C
        ais #1T                     ; Reposiciona pila
        tsx                         ; Transfiere SP a H:X
        ldx 2,x                     ; Carga Puntero Bajo de la posición a
                                    ; escribir
        pshx                        ; Empuja a pila puntero alto
        tsx                         ; Transfiere SP a H:X
        ldx 2,x                     ; Carga Puntero Bajo de la posición a
                                    ; escribir
        pshx                        ; Empuja puntero Alto de la posición a
                                    ; escribir
        pulh                        ; Recupera en H
        pulx                        ; Recupera puntero Bajo en X
        sta ,x                      ; Almacena Unidad
        cli                         ; Habilita interrupciones
        ais #3T                     ; Ajusta Puntero de Retorno
        rts                         ; Retorna
```

Listado 43. NT1006 – SSEGMUXD – MATH.inc. Archivo de funciones matemáticas en ensamblador, MathHex2Dec transforma un hexadecimal a Decimal.

```
;=====================================================================
; ARCHIVO      :  TIM.inc
; PROPÓSITO    :  Inclusión de la función de inicialización/utilitario de TIM
; LENGUAJE     :  IN-LINE ASSEMBLER
; ---------------------------------------------------------------------
; HISTORIAL
;   DD MM AA
;   00   08   04   Creado.
;   04   09   04   Modificado.
;=====================================================================

;=============================================
;                       Constantes & Macros
;=============================================
TIM_MAX_CH          equ 2T              ; Máximo de canales por temporizador
TIM_OP_LV_HI        equ 0T              ; Pin inicialmente en Alto
TIM_OP_LV_LO        equ $10             ; Pin inicialmente en Bajo

TIM_IC_RISE         equ $04             ; Captura Flanco de Subida
TIM_IC_FALL         equ $08             ; Captura Flanco de Bajada
TIM_IC_RISE_FALL    equ $0C             ; Captura ambos flancos

TIM_OC_UN_TGL       equ $14             ; Conmuta al comparar, Unb.
TIM_OC_UN_CLR       equ $18             ; Bajo al comparar, Unb.
TIM_OC_UN_SET       equ $1C             ; Alto al comparar, Unb.
TIM_OC_BU_TGL       equ $24             ; Conmuta al comparar, Buf.
TIM_OC_BU_CLR       equ $28             ; Bajo al comparar, Buf.
TIM_OC_BU_SET       equ $2C             ; Alto al comparar, Buf.

TIM_PWM_UN_TGL      equ $16             ; PWM Conmuta, Unb.
TIM_PWM_UN_CLR      equ $1A             ; PWM Bajo al comp., Unb.
TIM_PWM_UN_SET      equ $1E             ; PWM Alto al comp., Unb.
TIM_PWM_BU_TGL      equ $26             ; PWM Conmuta, Buf.
TIM_PWM_BU_CLR      equ $2A             ; PWM Bajo al comp, Buf.
TIM_PWM_BU_SET      equ $2E             ; PWM Alto al comp. Buf.
```

```
;=========================================
; TIMINIT       : Inicializa el Módulo TIM
; OBJETIVO      : Inicializa el TIM y configura
;                 divisor y Módulo
; ENTRADA       : A  valor del prescalar
;                 HX contiene el valor del
;                 registro TMOD[H:L]
; SALIDA        : Ninguna
; REGISTROS
; AFECTADOS     : TSC, TMOD[H:L], ACCA, SP
;=========================================
TIMInit
        psha                                ; Empuja A a la pila
        lda TSC                             ; Carga TSC
        and #{TOIE|TSTOP}                   ; Preserva Bits
        ora 1,SP                            ; or con el prescalar
        sta TSC                             ; Almacena en TSC
        ais #1                              ; reubica puntero de retorno
        sthx TMODH                          ; Almacena Módulo de contador
        rts                                 ; Retorna

;=========================================
; TIMOVINTEN    : Habilita interrupciones del
;                 módulo temporizador
; OBJETIVO      : TOIE = 1
; ENTRADA       : Ninguna
; SALIDA        : Ninguna
; REGISTROS
; AFECTADOS     : TSC, CCR
;=========================================
TIMOvIntEn
        sei                                 ; Deshabilita Interrupciones
        bset BIT6,TSC                       ; Habilita int. del módulo
        cli                                 ; Habilita interrupciones
        rts                                 ; retorna


;=========================================
; TIMStart      : Inicia el módulo TIM
; OBJETIVO      : Apaga el bit de parada
; ENTRADA       : Ninguna
; SALIDA        : Ninguna
; REGISTROS
; AFECTADOS     : CCR, TSC
;=========================================
TIMStart
        sei                                 ; Inhabilita Interrupciones
        bclr BIT5,TSC                       ; TSTOP = 0, TIM Hábil
        cli                                 ; Habilita Interrpciones
        rts                                 ; Retorna
```

```
;===========================================
; TIMOVACK    :  Borra bandera de sobreflujo
;                del TIM
; OBJETIVO    :  TSC_TOF = 0
; ENTRADA     :  Ninguna
; SALIDA      :  Ninguna
; REGISTROS
; AFECTADOS   :  TSC, CCR
;===========================================
TIMOvAck
        sei                             ; Deshabilita interrupciones
        bclr BIT7,TSC                   ; borra desborde
        cli                             ; Habilita interrupciones
        rts                             ; retorna
```

Listado 44. NT1006 – SSEGMUXD – TIM.inc. Archivo de funciones del temporizador. TIMInit inicializa el temporizador con un prescalar y tiempo determinado por el usuario, TIMStart inicia el temporizado mientras que TIMOvAck reconoce la interrupción. Solo se listan las funciones utilizadas.

```
;=================================================================
; ARCHIVO      : INTERRUPTSJL3.inc
; PROPÓSITO    : Plantilla de Funciones de Interrupciones para
;                JL3/JK1/JK3/Serie QT/QY
; LENGUAJE     : IN-LINE ASSEMBLER
; ---------------------------------------------------------------------
; HISTORIAL
;   DD MM AA
;   22  08  04   Creado.
;   06  12  04   Modificado.
;=================================================================

;=========================================
;          Interrupción de Temporizador del Sistema
;=========================================
;=========================================
; TIMOFL       : Servicio de interrupción del
;                temporizador del sistema
; OBJETIVO     : Reconoce la interrupción y
;                retorna al programa principal
; ENTRADA      : Ninguna
; SALIDA       : Ninguna
; REGISTROS
; AFECTADOS    : Ninguno
;=========================================
TIMOFL                                  ; TIM Sobreflujo (Bajo)
      jsr SSegDispMuxHandler            ; Reconoce la interrupción y
                                        ; multiplexa las pantallas
      rti                               ; NO, retorna de la interrupción
```

Listado 45. NT1006 – SSEGMUXD – INTERRUPTSJL3.inc. Solamente es listada la interrupción del temporizador, la cual es utilizada para refrescar las pantallas.

```
;=====================================================================
; ARCHIVO      :  VECTORSJL3.inc
; PROPÓSITO    :  Definir el vector de búsqueda de cada interrupción
; LENGUAJE     :  IN-LINE ASSEMBLER
; ---------------------------------------------------------------------
; HISTORIAL
;   DD MM AA
;   22   08   04   Creado.
;   06   12   04   Modificado.
;=====================================================================

;=============================================
;                Vector de Temporizador del Sistema
;=============================================
        org TIMOFH                          ; Sobreflujo del TIM (Alto)
        dw TIMOFL                           ; Sobreflujo del TIM (Bajo)

;=============================================
;      Vector de Reinicio del Sistema
;=============================================
        org RESET_VEC                       ; Puntero Vec - RESET
        dw START                            ; al darse reset salta a Start
```

Listado 46. NT1006 – SSEGMUXD – VECTORSJL3.inc. Listado solamente del vector utilizado, temporizador.

3.6.7 Conclusión

Los siete segmentos son dispositivos que contienen un grupo de LEDs, si se mantiene la pantalla encendida, causa que el consumo se “dispare”, es decir, aumente notablemente. Para un grupo de más de un (1) siete segmentos, el consumo puede ser tan alto, que se puede exceder la capacidad de la fuente, causando en el peor de los casos que la fuente ceda y fallezca.

Para reducir tal problema, se utiliza la técnica de multiplexado de pantallas, que decide de momento la pantalla a encender. Así, se reduce el consumo y se puede especificar en un diseño una fuente de menor requerimiento. La manera de que las pantallas den el efecto de estar todas encendidas depende de que tan frecuente se encienda y apague una de otra. La frecuencia mínima a la que se recomienda es 60 Hz, pero existen personas que utilizan un mínimo de 40 Hz para el refrescado de las pantallas.

3.6.8 Referencias

3.6.8.1 "Embedded Systems Building Blocks, Second Edition" - "Complete and Ready-to-Use Modules in C"
Autor: Jean J. Labrosse
Recurso: Capítulo 04 – Multiplexed LED Displays

3.6.8.2 Sistemas Digitales, Principios y Aplicaciones
Autor: Ronald J. Tocci
Recurso: Capítulo 9 – Sección 9.7, Multiplexores

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

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

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

NT0026 – Módulos – Rutinas reusables para programación.

3.6.8.6 Multiplexión de Siete Segmentos
(a) http://www.mcmanis.com/chuck/robotics/fpga/projects/5/multi-led.html
(b) http://claymore.engineer.gvsu.edu/~jackh/eod/mechtron/mechtron-77.html

3.6.8.7 Nota de Aplicación Sobre el Cálculo de Resistores para LEDs
(a) http://www.freescale.com/files/microcontrollers/doc/app_note/AN1238.pdf

AN1238 – "HC05 MCU LED Drive Techniques Using the MC68HC705J1A"

3.6.9 Problemas Propuestos

3.6.9.1 Utilice estas rutinas para crear un contador de botellas, en donde cada vez que las botellas toquen el interruptor, se incremente la cuenta.

3.6.9.2 Realice un reloj Digital utilizando la técnica de multiplexado de pantallas de siete segmentos.