NT1011

3.11 COMUNICACIÓN SERIAL RS-485 PARA MCUs SIN SCI

PARA MICROCONTROLADORES SIN SCI

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

ÏNDICE

3.11.1 Introducción

Este documento explica la comunicación serial entre microcontroladores, para microcontroladores que no posean SCI[1]. La comunicación serial posee muchas ventajas, dependiendo del tipo de estándar que se implemente (RS – 232 o RS – 485), lo cierto es que las más notables son:

- El aumento de pines en los puertos: Al tener dos (2) o más microcontroladores, se sacrificarían pines, pero se obtendrá en aumento un beneficio de más pines.
- La simplicidad de código: Siendo el "software" compartido, el código estará reducido en una cantidad considerable para cada monopastilla.
- El aumento de la capacidad de los tipos de memoria: El sistema de micros esclavizados, genera obviamente a que la capacidad total de memoria aumente.

Actualmente, en Panamá, existe la compañía SINTEC (Sistemas Inteligentes de Panamá) en donde ha utilizado el estándar RS-485 para interconectar controladores. Algunos de estos lugares son:

Autoridad del Canal de Panamá, Nuevo Hospital Santo Tomás, Museo del Canal, Edificio de Postgrado de la USMA, World Trade Center Panamá, MEDCOM (nuevo edificio), Cable Onda (nuevo edificio).

Nota: Fuente proporcionada por Elías Lombardo Batista. Colaborador de SINTEC.

[1] Interfase de Comunicación Serial

3.11.2 Materiales

1. Microcontroladores: 2 × JL3
2. Resistores: 14 × 470Ω (Amarillo – Violeta – Chocolate)
3. Capacitores: 2 × 0.01 μF (104)
4. Displays: 2 × MAN71 (Ánodo Común), Jameco Part No: 24731
5. Decodificador: 2 × BCD a 7-segmentos, Jameco Part No: 24731
6. Tarjeta de Desarrollo TD68HC908JL3 o similar y su microcontrolador
7. Plantilla de proyectos: Breadboard JE27, Jameco Part No: 20811
8. Fuente de Poder
9. Alambres AWG 20
10. Pelador de Alambres
11. Circuito Integrado: DS75176B, Jameco Part No: 50964

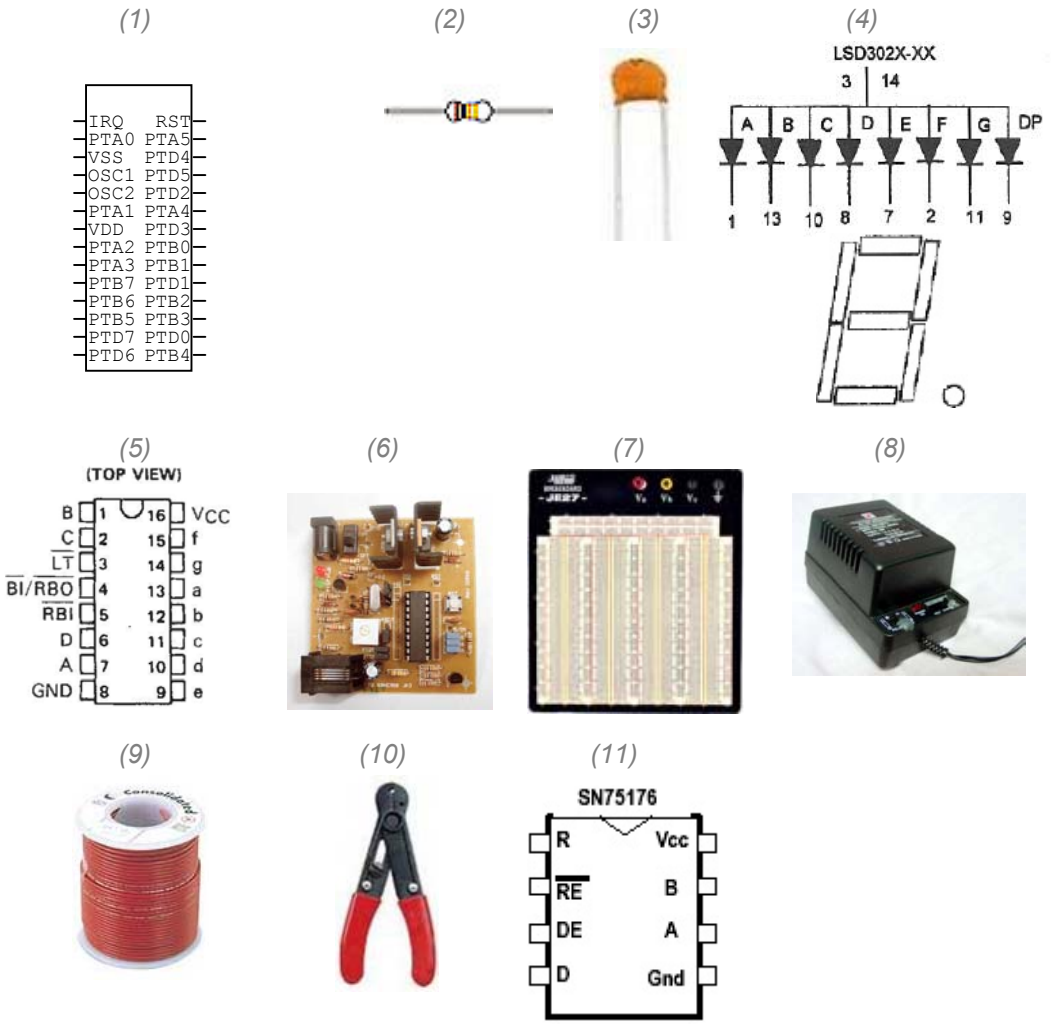
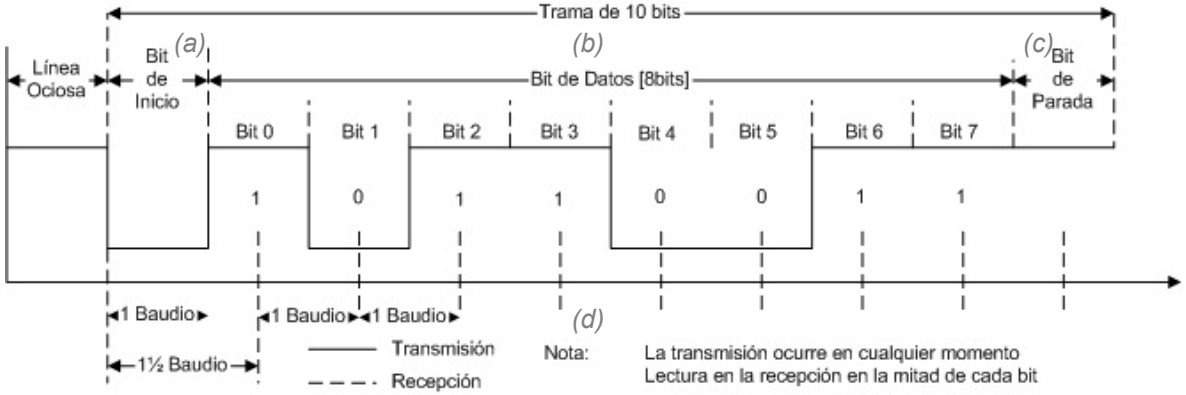


Figura 239. Materiales a Utilizar en el Esquema para RS485

3.11.3 Comunicación Serial Asíncrona

Cuando se habla de comunicaciones seriales asíncronas, directamente se piensa en el puerto serial de la PC. La palabra asíncrona viene debido a que el tipo de comunicación no es sincronizada por el reloj del sistema, sino, que puede ocurrir en cualquier instante. Cuando se transfiere la información, se da en grupos de bytes (ver figura 240(a)) o paquetes de información. Para una mejor referencia, ver la NT1002, sección 3.2.2. La velocidad con que conmuta un bit hacia otro, se llama *baudio* y corresponde a *una transición por segundo*; así, una rata de 9600 baudios, constituye un tiempo de 104.2 μs y una transferencia de 960 bytes (9600/10, asumiendo 1 bit de inicio, 8 de datos y 1 de parada).



Un baudio es la cantidad de transiciones por segundo que se da por cada bit,

Figura 240. Transmisión de un Byte de Datos Seriales. (a) Bit de Inicio. Un flanco de bajada avisa que inicia una nueva transmisión serial. (b) Bit de datos. Por lo general los bits de datos son ocho (8) y se transmite desde el bit menos significativo hasta el bit más significativo. (c) Bit de Parada. En el bit de parada, la línea es levantada, siempre es un uno (1) lógico y ocasionalmente varía entre uno (1), uno y medio (1½) y hasta dos (2) bits finales. (d) Recepción. En la ocasión de que el receptor detecte el flanco de bajada, se espera un baudio y medio para realizar la lectura; cada lectura se repetirá, esperando el baudio para fin de quedar en la mitad de cada bit y leer .

Tabla 82. Rata de Selección del Baudio

Rata de Baudio	Tiempo del Bit (μs)	#Bytes/seg.	Tiempo entre bytes (μs)
300	3 333.3	30	33 333
600	1 666.6	60	16 667
1200	833.3	120	8 333
2400	4 166.3	240	4 167
4800	208.3	480	2 083
9600	104.2	960	1 042
19200	52.1	1920	521
38400	26.0	3840	260
56000	17.9	5600	179

Nota: La tabla 82 fue extraida de la referencia 3.11.13.1 y se asume que existen 1 bit de inicio, 8 de datos y 1 de parada.

3.11.4 Estándar RS-485

En comunicaciones seriales existen varios estándares, este documento, explicará brevemente como se utilizará el RS-485, el cual es muy proliferado en la industria y desarrollado por la EIA (“Electronics Industry Association”), el mismo, mejora muchas de las desventajas que tiene el estándar RS-232 como lo es la longitud entre las unidades interconectadas. Las ventajas de este estándar se muestran en la tabla 83. Una interfase RS-485 consta de múltiples chips transmisores/receptores que proveen una salida diferencial (ver figura 241), algunos fabricantes las denotan como A y B, otros como D+ y D- (ver figura 239(11)) cuyos cables van trenzados (cada uno con su aislante). Cada elemento se conoce como unidad de carga ó UL y corresponde a un nodo. El estándar RS-485 solo especifica la capa de conexión física y el protocolo, tiempo y tipo de transmisión (serie o paralela) es abierto al diseñador.

Tabla 83. Ventajas del Estándar RS-485

Estándar RS-485
Rata de la Señal – Velocidades de Transmisión de hasta 10 MBits/seg.
Alta longitud de cable – Longitud de cable aproximada de 1200 metros (4000 pies)
Transmisión diferencial – Baja emisiones de ruido
Sistema Multipunto – Permite hasta 32 unidades (31 esclavos más 1 maestro)
Sistema Multimaestro – Se permite que el maestro cambie a esclavo
Nota: RS-485 es una comunicación estilo Half-Dúplex.

El “checksum” es la sumatoria de la operación lógica XOR a cada bit enviado. Esta se utiliza para verificación de errores cuando se envía paquetes de información.

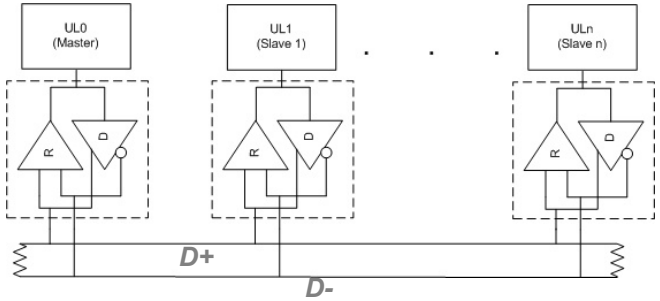


Figura 241 (superior). Capa Física para RS-485. Se recomienda tener resistores entre el inicio y el fin de las líneas de alrededor de los 120Ω para reducir las reflexiones de la línea de transmisión. El cable utilizado, recomendado, es 24 AWG y debe trenzarse para mayor reducir el ruido.

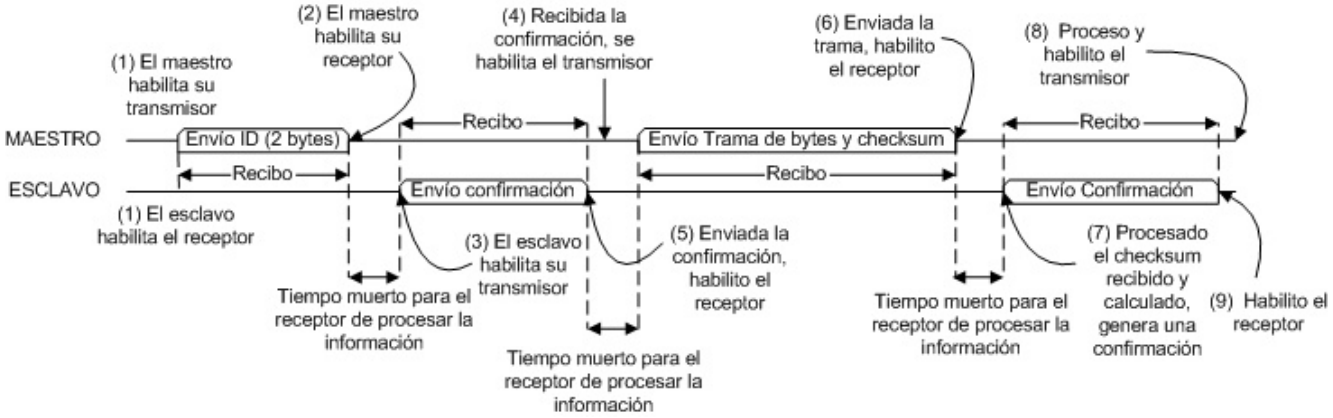


Figura 242 (izquierda). Protocolo creado para RS-485. El estándar no especifica un protocolo fijo, por consiguiente, se creo uno propio que consta de enviar ID como los dos primero bytes, y una trama de bytes que se puede configurar, esto por el transmisor. El receptor recibe el ID, envía confirmación, recibe trama y calcula-compara el “checksum” para luego tener una confirmación si la data es válida.

3.11.5 Descripción Rápida del Código

Esta rutina implementa la transmisión serial para cualquier microcontrolador 68HC908, ahora, para cualquier microcontrolador con puerto de interrupción externa dando prioridad a la recepción. La misma, fue calculada para un cristal de 4.9152 MHz, para otro cristal, recalcular el "delay" (retardo) de software de la sección 3.11.7. Lo ventajoso de la rutina es su reusabilidad y versatilidad de cambio de pin de transmisión.

El código utiliza un archivo tipo include[2] "SCI.inc", que no es más que una parte del programa. En este caso, este archivo consta de numerosas subrutina, de las cuales el usuario hará llamado a "SCITx485" y "SCIRx485", que mandan y reciben una trama de bytes en el formato de comunicación RS – 485, comprobando así que si se puede enviar y recibir desde microcontroladores que no posean módulo serial. También posee un retardo "SCIBaudDelay" que proporciona el tiempo de transmisión o recepción de un bit.

3.11.6 Subrutina de Transmisión y Recepción

Primeramente asegúrese que en su código de aplicación, está presente la librería SCI.inc, incluida como una línea de código de la siguiente manera:

```
#include '\FUNCTIONS\SCI.inc'       ; Incluye rutinas de RS-485
```

El llamado de la rutina de transmisión se da por medio de un lugar de la FLASH; imaginemos que queremos enviar la tabla NUM_NATURALES:

```
lda #1T                 ; ID del esclavo a contactar, ESCLAVO 1
ldhx #NUM_NATURALES     ; Puntero de la tabla de nums. naturales
jsr SCITx485            ; Envía ID, Trama y Checksum
```

El llamado de la rutina de recepción se da en cualquier parte del código y es totalmente libre de argumentos de entrada.

```
jsr SCIRx485            ; Recibe ID, Trama, Checksum y envía
                        ; confirmaciones
```

Para mayor información de el uso de estas subrutinas, entrada y salidas, favor, revisar el Apéndice G que habla sobre código reusable, sección del SCI para micros que no posean.

[2] Directiva del "assembler" para incluir una parte del programa como una línea.

3.11.7 Cálculo del Baudio

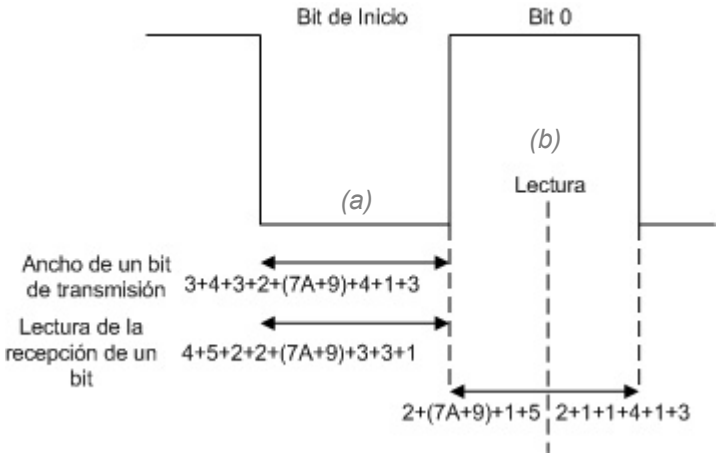


Figura 243. Calculo del Bit y Lectura. (a) Ancho del Bit. El ancho del bit se realiza a partir de subrutinas de retardo y consta de 7A+29 ciclos de máquina. (b) Lectura del Bit. Como la interrupción es asíncrona, el bit de inicio solo es muestreado una sola vez, consta de 7A+29 ciclos y luego, se realiza una lectura a 7A+17 ciclos, finalmente, se completa el resto del retardo del baudio que son de 10 a 12 ciclos de máquina. La operación de lectura se repite hasta llegar al bit de parada

Nota: 7·A + 9 es una subrutina de retardo sencilla, para configurar el baudio, Subrutina SCIBaudDelay.

Se puede calcular el valor de "A" requerido para señalizar la cantidad de baudios por una ecuación derivada, la cual es la siguiente

$$A = \frac{XTAL/(4 \times BAUD) - 29}{K}$$

Ecuación 19. Prescalar Selector de Ancho de Bit

$$\%\, error = 100 \times \left|1 - BAUD / BAUD_{calc}\right|$$

Ecuación 20. Cálculo del Porcentaje de Error

A selector de rata de baudio. Resultado en decimal
K es la constante de retardo. Para este caso particular *K = 7*
BAUD son la cantidad de bits por segundo a configurar
XTAL la frecuencia del cristal externo en Hertz

error es el error de transmisión
BAUD *cantidad de baudios deseados*
BAUDcalc baudios recalculados por la ecuación 20

Para la tabla 84, solo se pueden alcanzar los baudios de las velocidades tabuladas. En este caso se utilizó un cristal de 4.9152MHz.

Tabla 84. Tabla de Selector de Baudios

Baudios	Selector de Baudio	% de error
19200	#$05	0
9600	#$0E	0.78
4800	#$20	1.17
2400	#$45	0

3.11.8 IC 75176

El integrado 75176 es el integrado que comúnmente se utiliza para hacer la interfase de comunicación a 485. Algunos fabricantes de este IC son: Texas Instruments (SN75176) y National Semiconductors (DS75176); y se puede conseguir en el sitio http://www.jameco.com por un costo unitario aproximado de B/. 0.89.

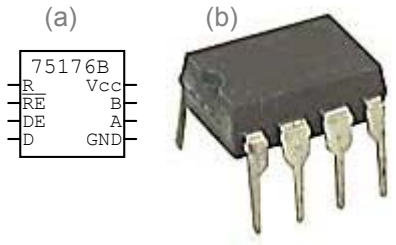


Figura 244 (izquierda). Encapsulado RS485. (De izquierda a derecha). (a) Esquemático del encapsulado. (b) Integrado de 8 pines utilizado para RS-485.

Tabla 85(a). Transmisor IC RS485

DRIVER (TRANSMISOR)			
INPUT D	ENABLE DE	OUTPUTS A	OUTPUTS B
H	H	H	L
L	H	L	H
X	L	Z	Z

Tabla 85(b). Receptor IC RS485

RECEIVER (RECEPTOR)		
DIFFERENTIAL INPUTS A – B	ENABLE $\overline{RE}$	OUTPUT R
$V_{ID} \geq 0.2$ V	L	H
-0.2 V > V_{ID} > 0.2 V	L	?
$V_{ID} \leq$ -0.2 V	H	Z
X	H	Z
OPEN	L	?

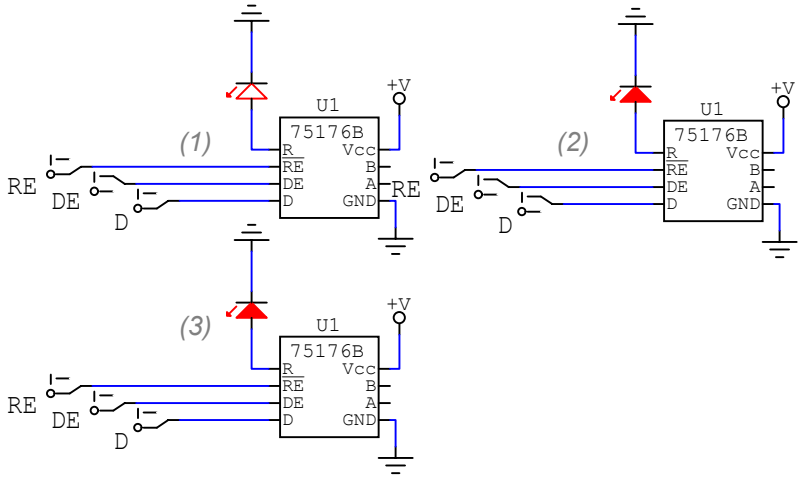


Figura 245. Prueba del integrado 75176.
(1) Transmisión y recepción habilitadas, led apagado, D = 0.
(2) Transmisión y recepción habilitadas, led encendido, D = 1.
(3) Observe que cuando s deshabilita el transmisor, la línea está en alto (LED = ON), es una buena condición para estándar seriales.

Tablas de verdad del transmisor/receptor, integrado 75176.

X = Condiciones no importa o irrelevante.
Z = Estado de alta impedancia o entradas / salidas flotantes.
? = Indeterminado, la respuesta es aleatoria.

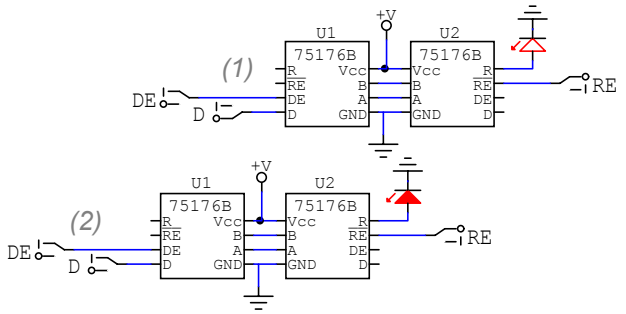


Figura 246. Transferencia entre puntos con dos integrados
Suponiendo que de un lado solo transmite y del otro solo recibe
Varie la señal en D (0 o 1) y note como se transfiere el dato a R que se visualiza por el LED.
(1) Transmito un 0 en U1 y recibo un 0 en U2 ($D_{U1} = R_{U2} = 0$).
(2) Transmito un 1 en U1 y recibo un 1 en U2 ($D_{U1} = R_{U2} = 1$).

3.11.9 Esquemático de la Aplicación

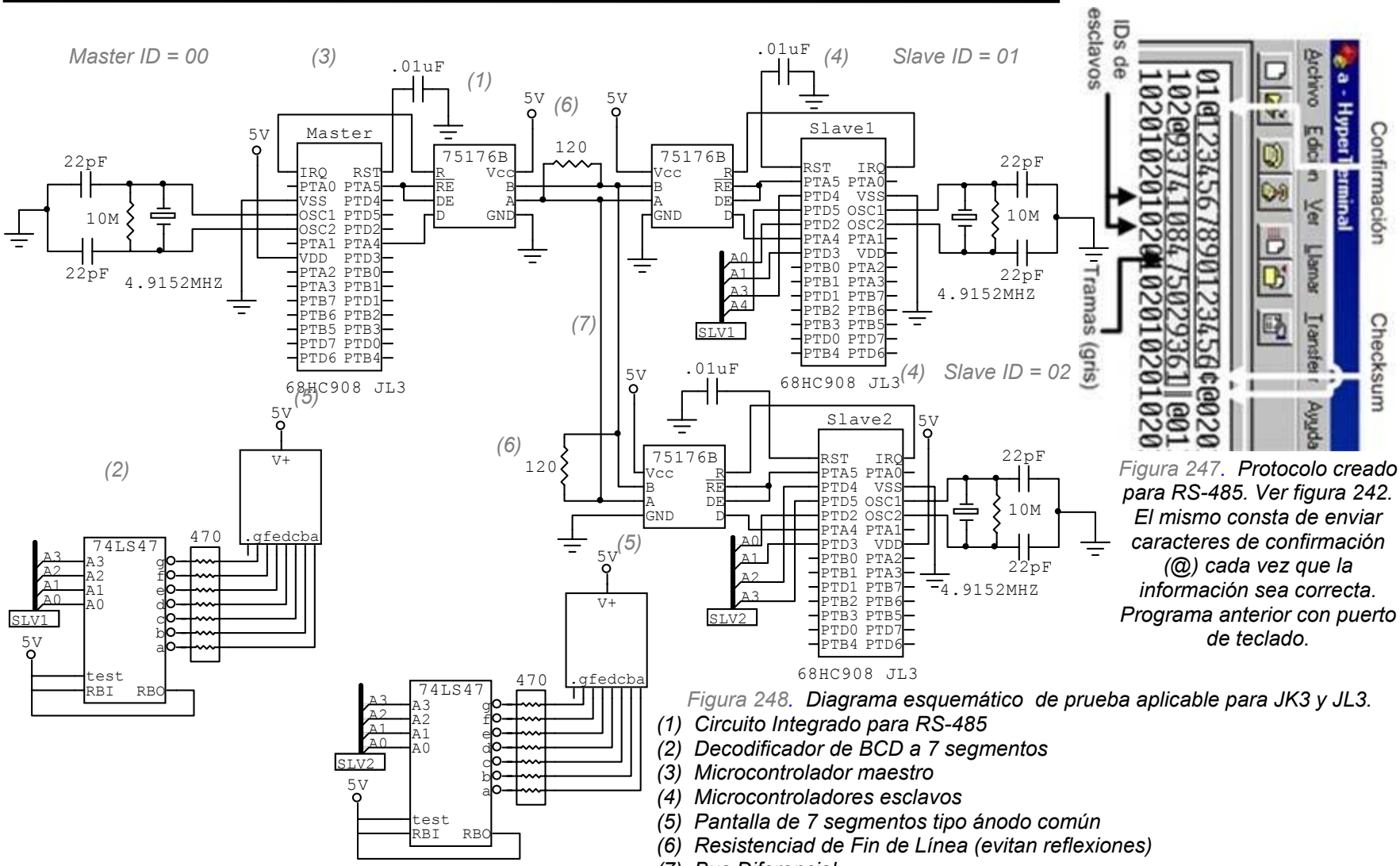


Figura 247. Protocolo creado para RS-485. Ver figura 242. El mismo consta de enviar caracteres de confirmación (@) cada vez que la información sea correcta. Programa anterior con puerto de teclado.

Figura 248. Diagrama esquemático de prueba aplicable para JK3 y JL3.

(1) Circuito Integrado para RS-485
(2) Decodificador de BCD a 7 segmentos
(3) Microcontrolador maestro
(4) Microcontroladores esclavos
(5) Pantalla de 7 segmentos tipo ánodo común
(6) Resistenciad de Fin de Línea (evitan reflexiones)
(7) Bus Diferencial

3.11.10 Diagrama de Flujo

El siguiente programa comunica serialmente tres (3) microcontroladores JL3, de los cuales dos (2) son esclavos y envía tramas que se desean desplegar en cada siete segmentos. Para el siguiente programa se asume un dominio total de las notas: NT0009 – Retardos, NT0010 – Puertos de E/S, NT0109 – IRQ1 y opcionalmente los apéndices. Este "software" se puede adecuar para cualquier microcontrolador.

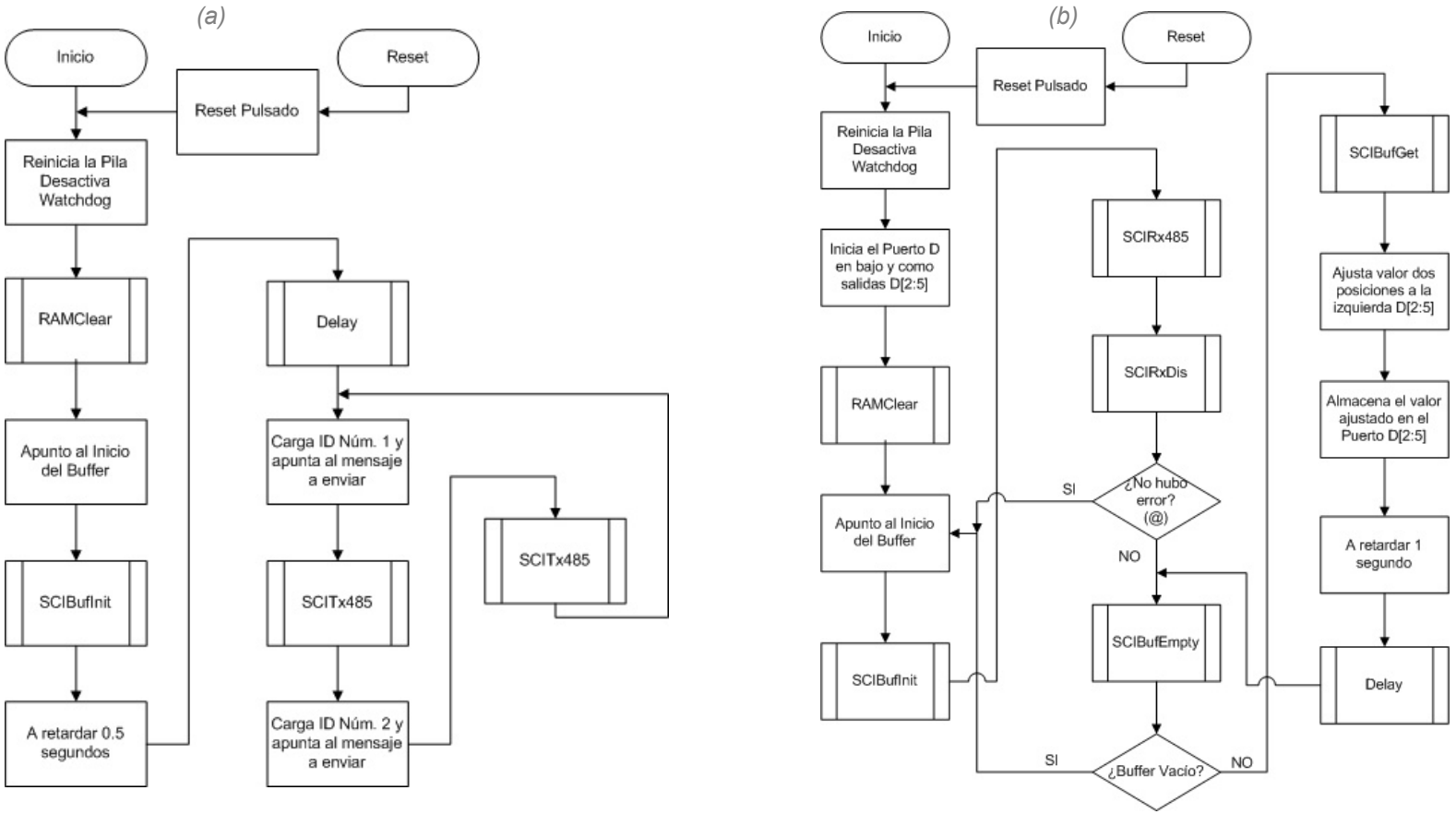


Figura 249. NT1011 – "SerialCom". (a) Rutina del Maestro. El maestro envía el ID y luego si existe confirmación envía la trama de "bytes" con el respectivo "checksum" o verificador de errores. (b) Rutina del Esclavo. El esclavo recibe el ID y luego de una confirmación al maestro recibe la trama la cual es desplegada secuencialmente en el Puerto D, bits dos a cinco.

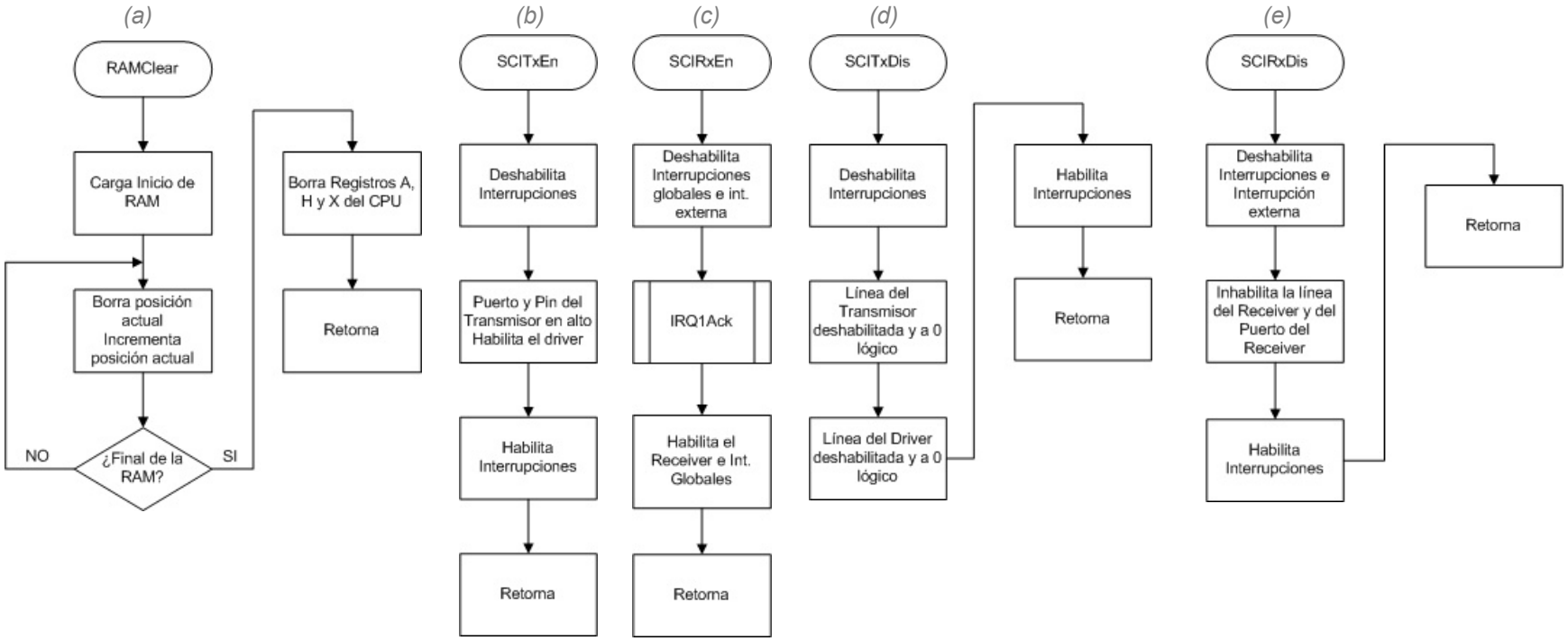


Figura 250. NT1011 – "SerialCom" - Subrutinas. (a) RAMClear. Borra la RAM del microcontrolador y registros A, H:X. (b) SCITxEn. Habilita el puerto destinado a transmisión en estado alto y predeterminado como salida; además habilita el "driver" del integrado 75176. (c) SCIRxEn. Habilita el "receiver" del integrado 75176. (d) SCITxDis Inhabilita el puerto destinado a transmisión, ahora destinado a ser entrada; además inhabilita el "driver" del integrado 75176. (e) SCIRxDis. Inhabilita el "receiver" del integrado 75176.

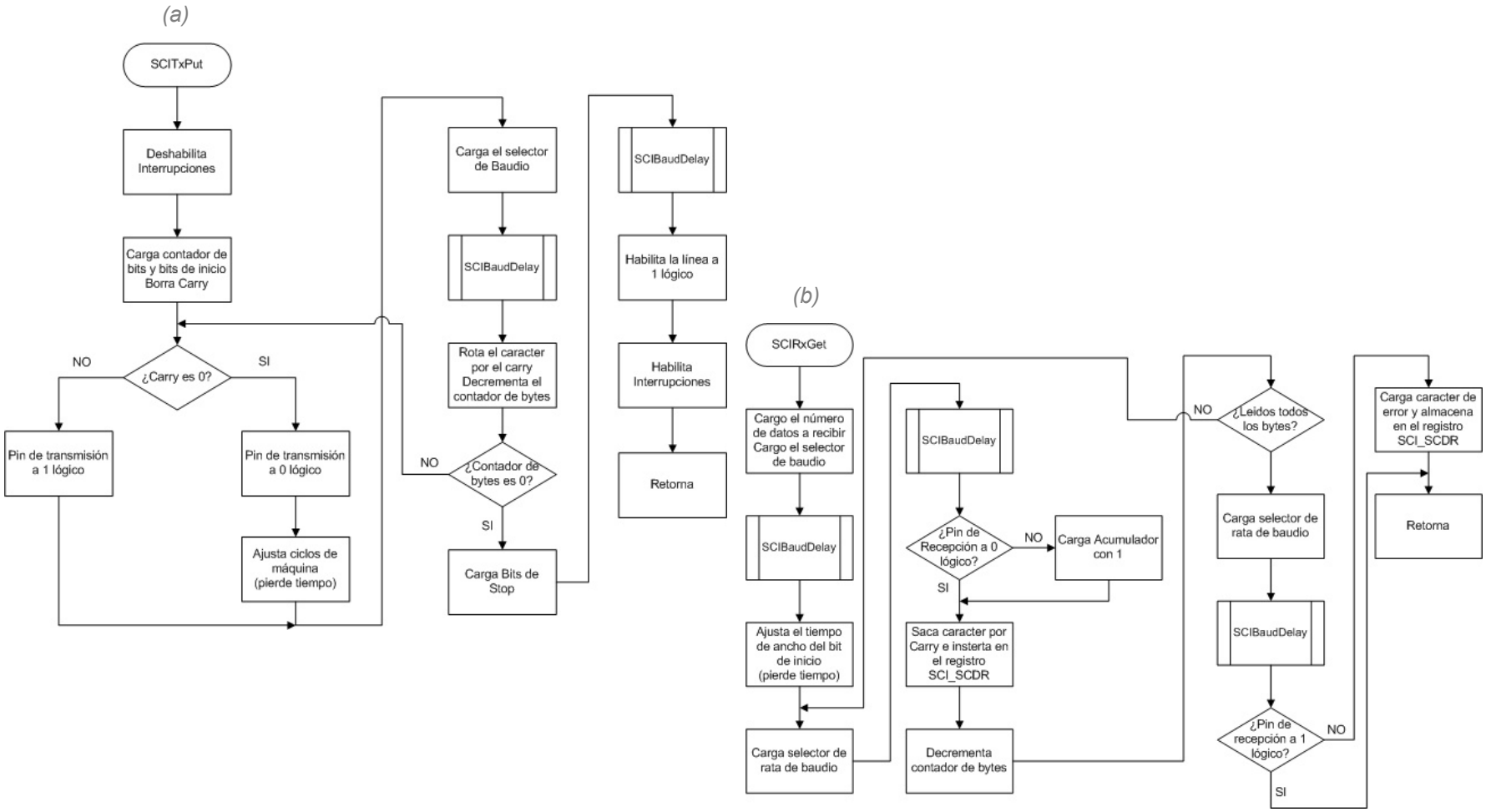


Figura 251. NT1011 – "SerialCom" – Subrutinas – Continuación. (a) SCITxPut. A la izquierda, se muestra la subrutina utilizada para enviar un (1) byte por el puerto serial o pin del puerto destinado a ser transmisor. (b) SCIRxGet. La subrutina lee, bit por bit, el caracter a recibir por el puerto serie o pin IRQ1 del microcontrolador.

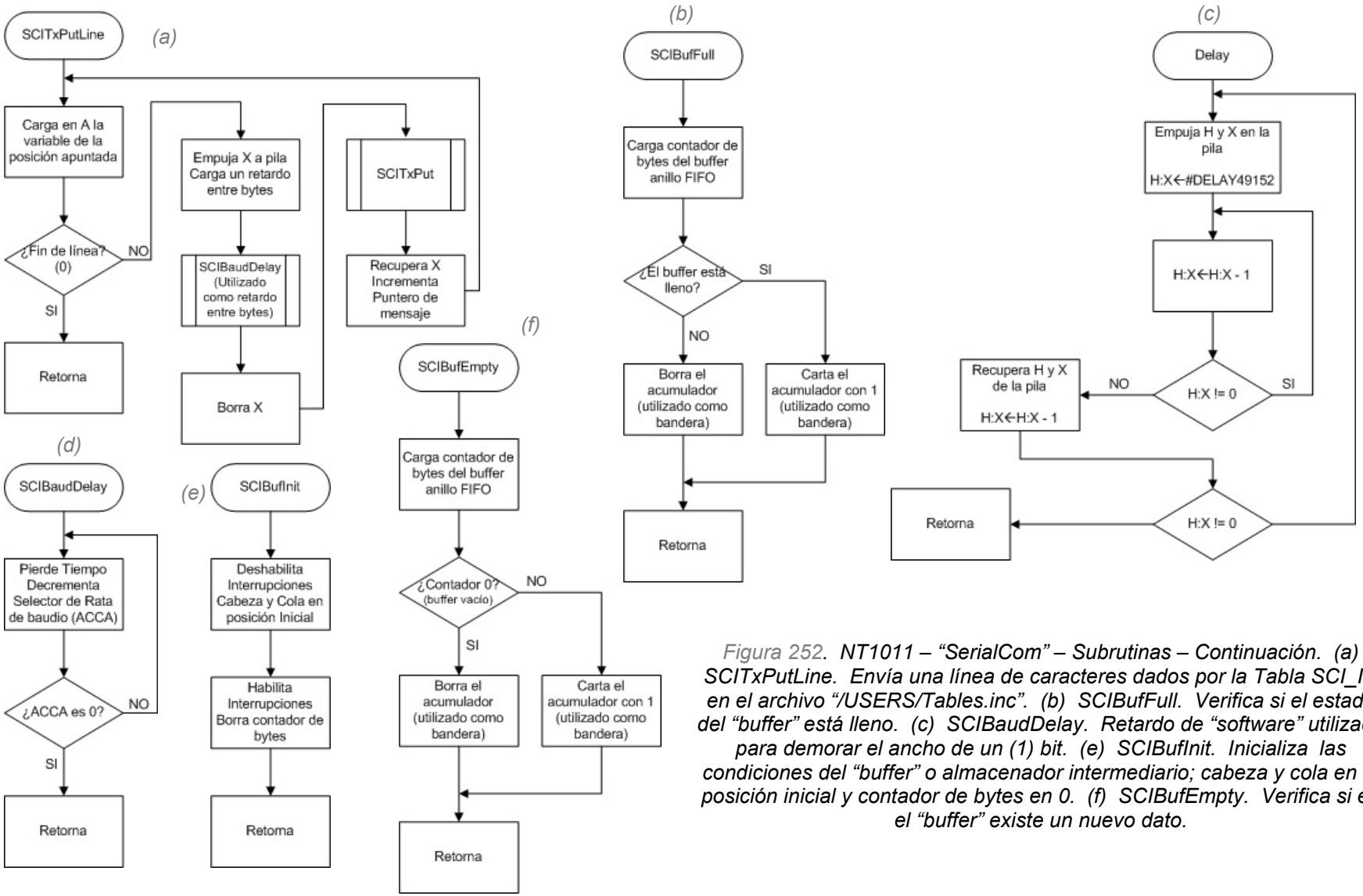


Figura 252. NT1011 – "SerialCom" – Subrutinas – Continuación. (a) SCITxPutLine. Envía una línea de caracteres dados por la Tabla SCI_ID en el archivo "/USERS/Tables.inc". (b) SCIBufFull. Verifica si el estado del "buffer" está lleno. (c) SCIBaudDelay. Retardo de "software" utilizado para demorar el ancho de un (1) bit. (e) SCIBufInit. Inicializa las condiciones del "buffer" o almacenador intermediario; cabeza y cola en la posición inicial y contador de bytes en 0. (f) SCIBufEmpty. Verifica si en el "buffer" existe un nuevo dato.

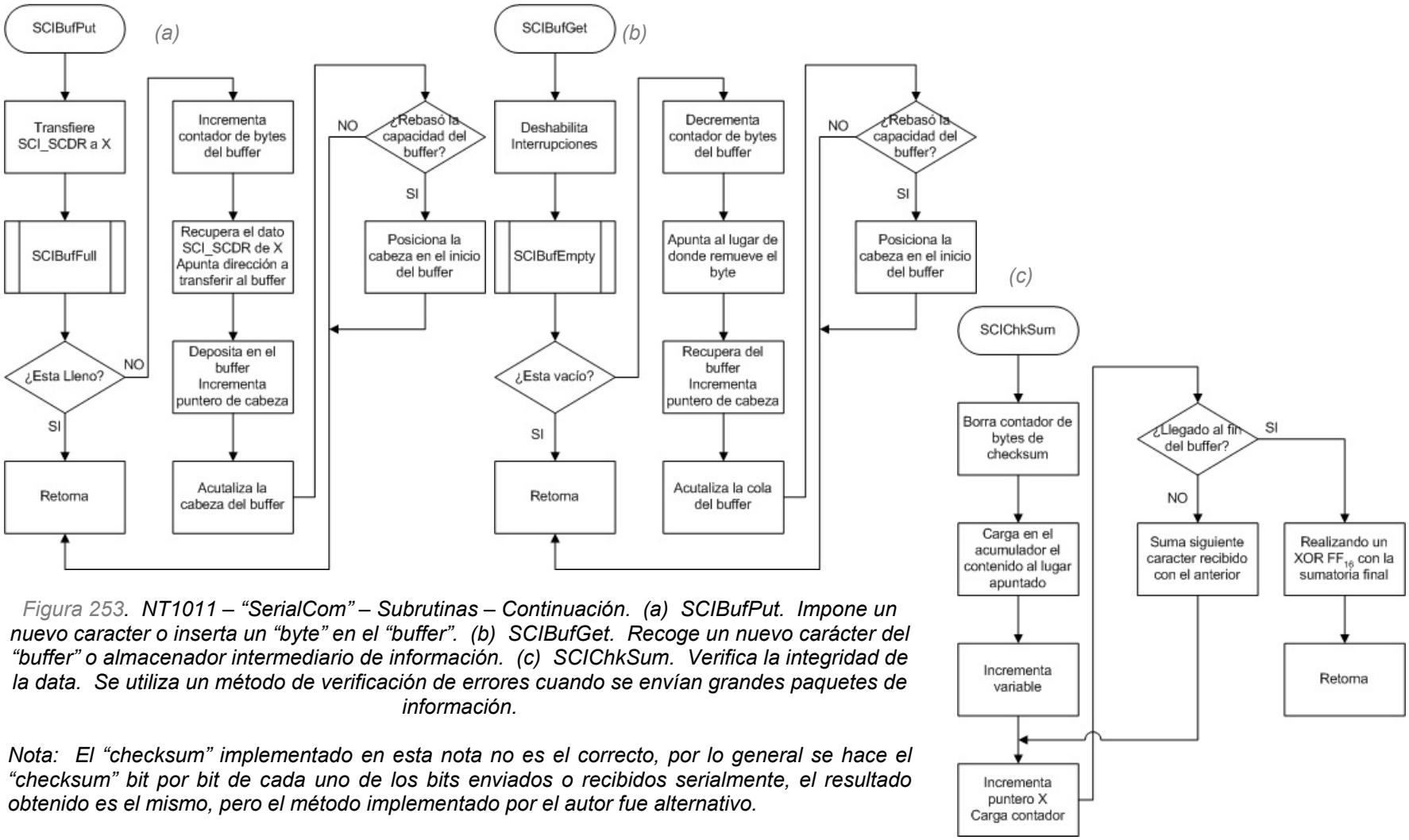


Figura 253. NT1011 – "SerialCom" – Subrutinas – Continuación. (a) SCIBufPut. Impone un nuevo caracter o inserta un "byte" en el "buffer". (b) SCIBufGet. Recoge un nuevo carácter del "buffer" o almacenador intermediario de información. (c) SCIChkSum. Verifica la integridad de la data. Se utiliza un método de verificación de errores cuando se envían grandes paquetes de información.

Nota: El "checksum" implementado en esta nota no es el correcto, por lo general se hace el "checksum" bit por bit de cada uno de los bits enviados o recibidos serialmente, el resultado obtenido es el mismo, pero el método implementado por el autor fue alternativo.

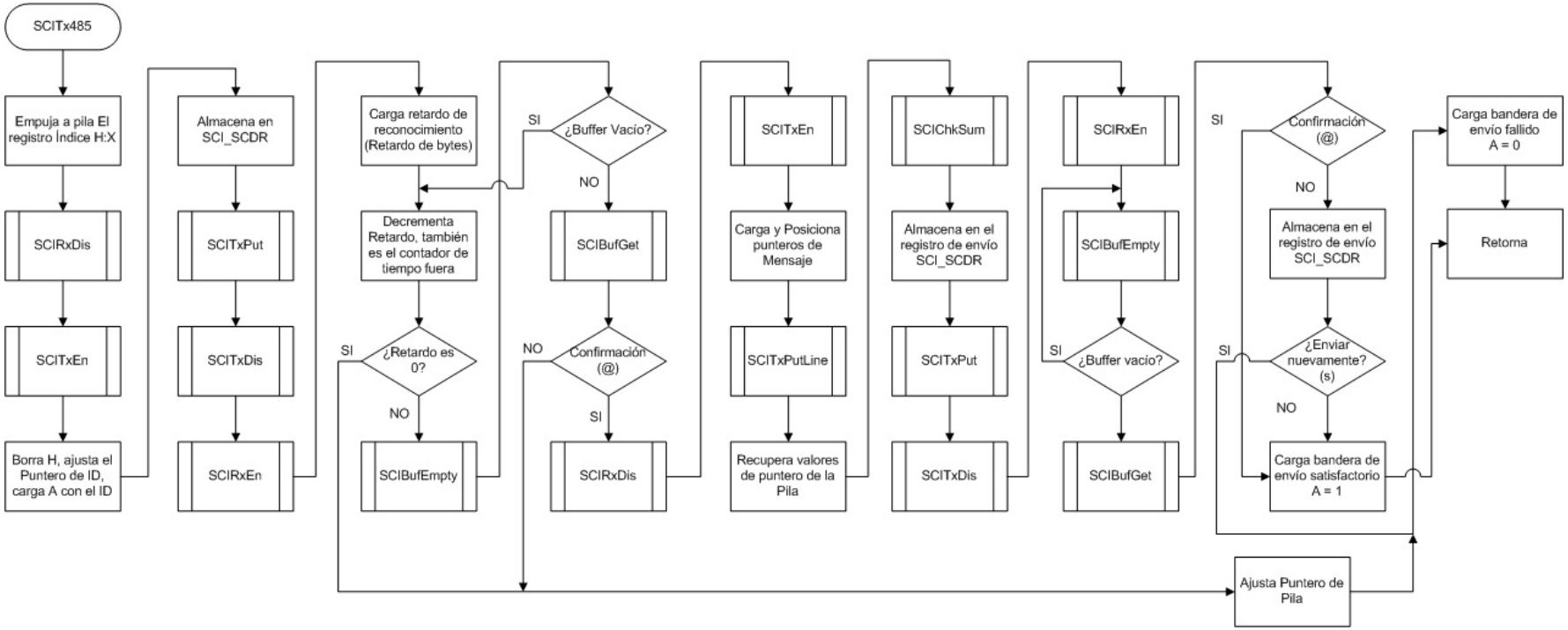


Figura 254. NT1011 – "SerialCom" – Subrutinas – Continuación. SCITx485. Subrutina utilizada para enviar un protocolo de comunicación creado para el estándar RS-485.

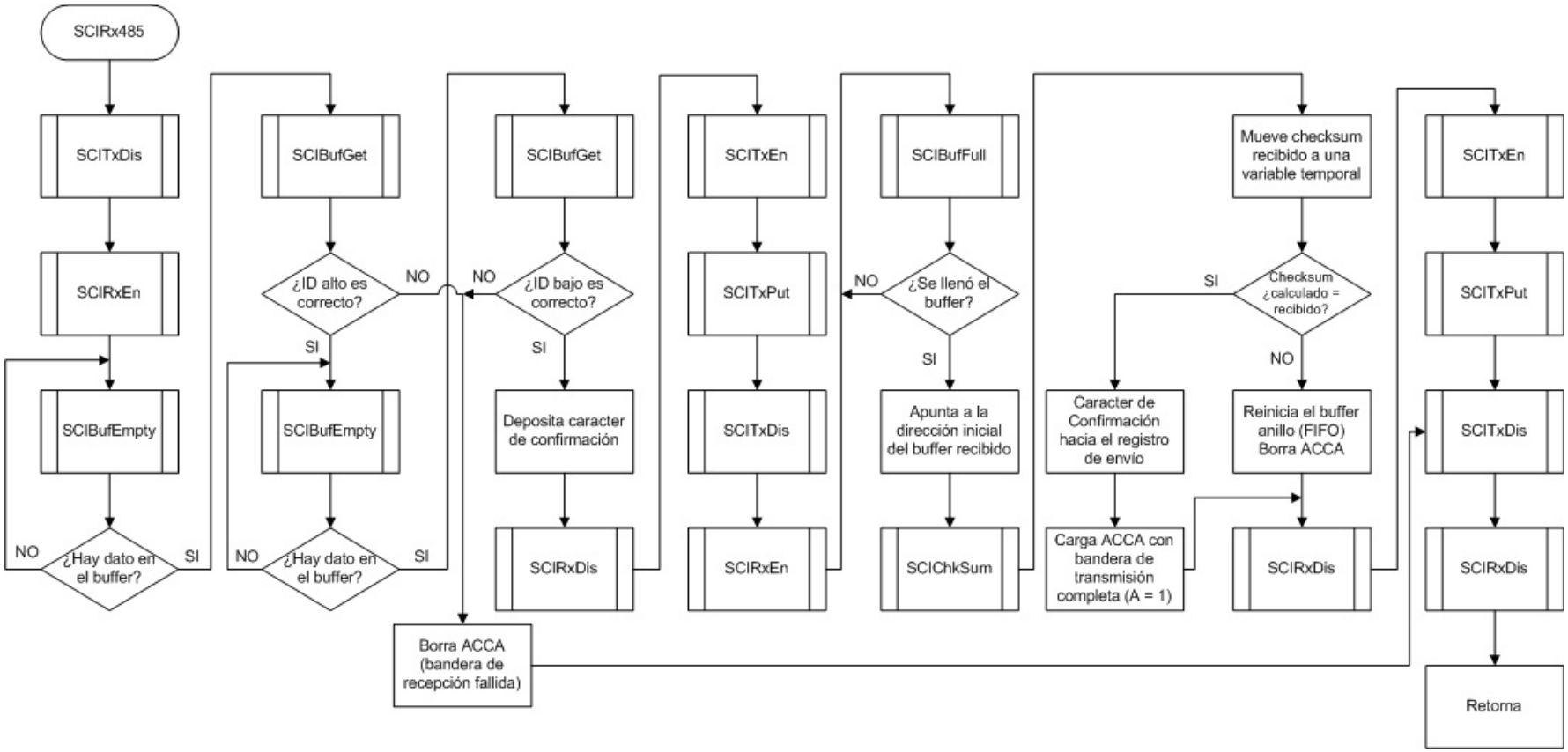


Figura 255. NT1011 – "SerialCom" – Subrutinas – Continuación. SCIRx485. Subrutina utilizada para recibir el protocolo creado para RS-485.

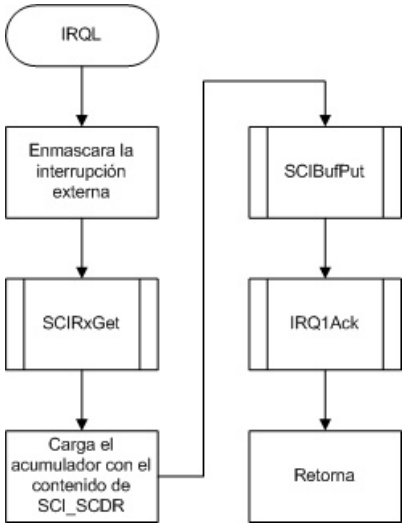


Figura 256. NT1011 – "SerialCom" – Subrutinas – Subrutina de Interrupción. IRQL contiene las subrutinas de interrupción encargada de recibir un "byte" y depositarlo en el "buffer" de recepción.

3.11.11 Código

```
;===================================================================
; ARCHIVO      :  NT1011 – SerialCom – master – 22 10 04.asm
; PROPÓSITO    :  Enviar una trama en formato RS-485, tipo repetitiva
; NOTA         :  Utiliza funciones de Interrupción Externa y SCI
;
; REFERENCIA:  NT1011 - SerialCom - 20 10 04
;
; LENGUAJE     :  IN-LINE ASSEMBLER
; -------------------------------------------------------------------
; HISTORIAL
;   DD MM AA
;   05  10   04    Creado.
;   22  10   04    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     :  Transmite ID, Trama y Check
;                 sum en formato RS-485
;===============================================
START
        rsp                         ; reinicia el puntero de pila
        bset BIT0,CONFIG1           ; deshabilita la función del watchdog
        jsr RAMClear                ; Borra RAM
        ldhx #SCI_FIFO              ; Apunta a la dirección inicial del buffer
        jsr SCIBufInit              ; Inicializa el buffer anillo estilo FIFO
        ldhx #500T                  ; Retardar 500 ms para ...
        jsr Delay                   ; ... esperar al esclavo.
SCI_485_1011.A
        lda #1T                     ; ID #0
        ldhx #MENSAJE               ; Puntero de mensaje
        jsr SCITx485                ; Transmite en formato RS-485
        lda #2T                     ; ID #1
        ldhx #MENSAJE1              ; Puntero de mensaje
        jsr SCITx485                ; Transmite en formato RS-485
        bra SCI_485_1011.A          ; Transmite nuevamente

;===============================================
;    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 82. NT1011 – "SerialCom" – "master" – 22 10 04.asm. Rutina principal del maestro, inicialmente retarda quinientos milisegundos (500 ms) para darle oportunidad a los esclavos a escuchar el llamado del maestro, para luego el maestro enviar tramas respectivas a los esclavos uno (ID = 01) y dos (ID = 02).

```
;=================================================================
; ARCHIVO      : NT1011 – SerialCom – slave – 22 10 04.asm
; PROPÓSITO    : Recibir una trama en formato RS-485, tipo repetitiva y des
;                plegarla en el 7-segmentos
; NOTA         : Utiliza funciones de Interrupción Externa y SCI
;
; REFERENCIA:  NT1011 - SerialCom - 20 10 04
;
; LENGUAJE     : IN-LINE ASSEMBLER
; ---------------------------------------------------------------------
; HISTORIAL
;   DD MM AA
;   05  10    04   Creado.
;   22  10    04   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     :  Transmite ID, Trama y Check
;                 sum en formato RS-485
;===========================================
START                                   ; Label de Inicio
        rsp                             ; reinicia el puntero de pila
        bset BIT0,CONFIG1               ; deshabilita la función del watchdog
        mov #0,PORTD                    ; 7-SEG = 0
        mov #{DDRD2|DDRD3|DDRD4|DDRD5},DDRD
                                        ; PTD[2:5] = salidas
        jsr RAMClear                    ; Borra RAM
SCI_RESET1011.A
        ldhx #SCI_FIFO                  ; Apunta a la dirección inicial del buffer
        jsr SCIBufInit                  ; Inicializa el buffer anillo estilo FIFO
        jsr SCIRx485                    ; Recibe en 485
        jsr SCIRxDis                    ; Deshabilita receptor
        lda SCI_SCDR                    ; Carga registro de datos
        cmp #'@'                        ; Compara con recepción satisfactoria
        bne SCI_RESET1011.A             ; Si no es igual, reiniciar
SCI_NEXT1011.B
        jsr SCIBufEmpty                 ; Verifica si el buffer está vacío
        bne SCI_RESET1011.A             ; SI, Pedir nuevos datos
        jsr SCIBufGet                   ; NO, Recoger 1 byte del buffer
        sub #'0'                        ; Restar 0 ASCII
        rola                            ; Rotar a la izq.
        rola                            ; Rotar a la izq.
        sta PORTD                       ; Enviar al puerto

        ldhx #1000T                     ; 1 seg
        jsr Delay                       ; Retardar
        bra SCI_NEXT1011.B              ; Desplegar siguiente
;===========================================
;   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 83. NT1011 – "SerialCom" – "slave" – 22 10 04.asm. Rutina principal del esclavo, la misma, despliega una trama de "bytes" recibida por un almacenador intermediario, siempre y cuando se de una recepción completa.

```
;*****************************************************************
; 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

;*****************************************
; RAMCLEAR    : 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
;*****************************************
RAMClear                        ; Borra la RAM y registros
      ldx #RAM_ORG              ; Carga con el origen

RAM_EMPTY1011.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_EMPTY1011.A       ; Si no concuerda entonces sigue limpiando
      clra                      ; Borra A
      clrh                      ; Borra H
      clrx                      ; Borra X
      rts                       ; retorna
```

Listado 84. NT1011 – "SerialCom" – RAM.inc. Rutinas de RAM, por el momento, existe solo la rutina de limpiado total de la memoria RAM.

```
;=================================================================
; ARCHIVO      : SCI.inc
; PROPÓSITO    : Funciones de comunicación serial RS232-RS485 para micros sin
;                SCI.
;
;           Para configurar el módulo SCI
;           1     - Configurar SCI_BAUD dado por la tabla
;           2     - Especificar el número de datos SCI_DATA_BITS
;           3     - Especificar el número de bits de parada SCI_STOP_BITS
;           4     - Especificar el ID del dispositivo, caracteres alto/bajo
;                   SCI_ID_H, SCI_ID_L
;           5     - Especificar la dirección del byte a transmitir SCI_SCDR
;           6     - Especificar la dirección del contador de byte de checksum
;           7     - Epecificar el registro de direccionamiento, puerto, pin
;                   SCI_TX_DDR, SCI_TX_PORT, SCI_TX_PIN, SCI_DRE_DDR,
;                   SCI_DRE_PORT, SCIR_DRE_PIN
;           8     - El máximo de caracteres del buffer SCI_MAX_CHAR
;           9     - La dirección del contador del buffer SCI_FIFO_CTR
;           10    - La dirección de la cabeza del buffer SCI_FIFO_HD
;           11    - La dirección de la cola del buffer SCI_FIFO_TL
;           12    - La dirección de el inicio del buffer SCI_FIFO
;
; LENGUAJE     : IN-LINE ASSEMBLER
; ---------------------------------------------------------------------
; HISTORIAL
;  DD MM AA
;  05  10  04   Creado.
;  31  10  04   Modificado.
;=================================================================

;=========================================
;         CONSTANTES & MACROS
;=========================================

;=========================================
;           SELECTOR DE BAUDIOS DEL TX/RX
;               PARA UN XTAL DE 4.9152 MHz
;=========================================
; BAUDIOS    NO. SELECTOR DE BAUD.      % DE ERROR ABSOLUTO
; 19200            #$05                      0.00
;  9600            #$0E                      0.78
;  4800            #$20                     1.17
;  2400            #$45                     0.00

;=========================================
;    SELECCIONE EL FORMATO DE TRANSMISIÓN
;=========================================
SCI_BAUD        equ $05          ; 19200 baudios
SCI_DATA_BITS   equ 8T           ; 8 bits de datos
SCI_STOP_BITS   equ 1T           ; 1 bit de parada
```

```
;===========================================
;  UBIQUE EL IDENTIFICADOR DE SU DISPOSITIVO
;===========================================
SCI_ID_H        equ '0'         ; ID.H = 0
SCI_ID_L        equ '1'         ; ID.L = 1

;===========================================
;         UBIQUE EL DELAY DE RECONOCIMIENTO
;===========================================
SCI_ACK_DELAY   equ $30         ; Mínimo ($28), Máximo ($FF)

;===========================================
;     REUBIQUE LOS REGISTROS DE CONTROL
;===========================================
SCI_SCDR        equ $80         ; Reserva 1 byte para el dato enviado
                                ; por serial
SCI_CHK_CTR     equ $81         ; Reserva 1 byte para conteo de
                                ; chksum, es temporal

;===========================================
;  DEFINA EL PUERTO Y REGISTROS DE TRANSMISIÓN
;===========================================
SCI_TX_DDR      equ DDRA        ; Puerto del Registro de
                                ; Direccionamiento de Transmisión
SCI_TX_PORT     equ PORTA       ; Puerto de Transmisión
SCI_TX_PIN      equ BIT4        ; Pin del Puerto al que pertenece la
                                ; Recepción

;===========================================
; DEFINA PUERTOS Y REG. DE HABILIT. DEL DRIVE
;===========================================
SCI_DRE_DDR     equ DDRA        ; Puerto del Registro de
                                ; Direccionamiento del Pin Enable de
                                ; TX/RX
SCI_DRE_PORT    equ PORTA       ; Puerto de Habilitación del IC (DE,
                                ; RE)

SCI_DRE_PIN     equ BIT5        ; Pin del Puerto del Habilitador del
                                ; TXD
                                ; ... Driver or Receiver Enabled

;===========================================
; DEFINA TAMAÑO Y UBICACIÓN DEL BUFFER ANILLO
;===========================================
SCI_MAX_CHAR    equ 16T         ; Máximo de caracteres del buffer
SCI_FIFO_CTR    equ $A1         ; Contador de bytes del buffer
SCI_FIFO_HD     equ $A2         ; Cabeza del Buffer
SCI_FIFO_TL     equ $A4         ; Cola del Buffer
SCI_FIFO        equ $A6         ; Dirección Inicial del Buffer Anillo
SCI_FIFO_MAX    equ {SCI_FIFO+SCI_MAX_CHAR}
                                ; Máximo Puntero del buffer
```

```
;===========================================
; SCITXEN      : Habilita el transmisor del
;                módulo SCI, IC 75176 y el
;                pin del puerto del transmisor
; OBJETIVO     : Transmisor habilitado
; ENTRADA      : Ninguna
; SALIDA       : Ninguna
; REGISTROS
; AFECTADOS    : CCR, DDRX, PORTX
;===========================================
SCITxEn
        sei                                 ; Deshabilita interrupciones
        bset SCI_TX_PIN,SCI_TX_PORT         ; Línea del transmisor a 1 lógico
        bset SCI_TX_PIN,SCI_TX_DDR          ; Habilita el transmisor
        bset SCI_DRE_PIN,SCI_DRE_PORT       ; Línea del driver 485 a 1 lógico
        bset SCI_DRE_PIN,SCI_DRE_DDR        ; Habilita el driver
        cli                                 ; Habilita interrupciones
        rts                                 ; retorna

;===========================================
; SCIRXEN      : Habilita el receptor del mó-
;                dulo SCI, y el pin del puerto
;                del módulo receptor.
; OBJETIVO     : Receptor habilitado
; ENTRADA      : Ninguna
; SALIDA       : Ninguna
; REGISTROS
; AFECTADOS    : CCR, DDRX, PORTX, INTSCR
;===========================================
SCIRxEn
        sei                                 ; Deshabilita interrupciones
        bset BIT1,INTSCR                    ; Enmascara interrupcion externa
        jsr IRQ1Ack                         ; Reconoce la int. externa
        bclr SCI_DRE_PIN,SCI_DRE_PORT       ; Habilita receptor
        bset SCI_DRE_PIN,SCI_DRE_DDR        ; Habilita receptor
        cli                                 ; Habilita interrupciones
        rts                                 ; Retorna
```

```
;=========================================
; SCITXDIS     : Deshabilita el transmisor del
;                módulo SCI, IC 75176 y el
;                pin del puerto del transmisor
; OBJETIVO     : Transmisor inhabilitado
; ENTRADA      : Ninguna
; SALIDA       : Ninguna
; REGISTROS
; AFECTADOS    : CCR, DDRX, PORTX
;=========================================
SCITxDis
        sei                                 ; Deshabilita interrupciones
        bclr SCI_TX_PIN,SCI_TX_PORT         ; Línea del transmisor a 0 lógico
        bclr SCI_TX_PIN,SCI_TX_DDR          ; Deshabilita el transmisor
        bclr SCI_DRE_PIN,SCI_DRE_PORT       ; Línea del driver 485 a 0 lógico
        bclr SCI_DRE_PIN,SCI_DRE_DDR        ; Inhabilita el driver
        cli                                 ; Habilita interrupciones
        rts                                 ; retorna

;=========================================
; SCIRXDIS     : Habilita el receptor del mó-
;                dulo SCI, y el pin del puerto
;                del módulo receptor.
; OBJETIVO     : Receptor inhabilitado
; ENTRADA      : Ninguna
; SALIDA       : Ninguna
; REGISTROS
; AFECTADOS    : CCR, DDRX, PORTX, INTSCR
;=========================================
SCIRxDis
        sei                                 ; Deshabilita interrupciones
        bset BIT1,INTSCR                    ; Inhabilita interrupción externa
        bclr SCI_DRE_PIN,SCI_DRE_PORT       ; Inhabilita receptor
        bset SCI_DRE_PIN,SCI_DRE_DDR        ; Inhabilita receptor
        cli                                 ; Inhabilita interrupciones
        rts                                 ; Retorna
```

```
;===========================================
; SCITXPUT     : Envía un byte por serial
; OBJETIVO     : Enviar un caracter
; ENTRADA      : SCI_SCDR    contiene el ca-
;                racter a enviar
;                por serial
; SALIDA       : Ninguna
; REGISTROS
; AFECTADOS    : CCR, DDRX, PORTX, X
;===========================================
SCITxPut                                    ; Label de Enviar dato
        sei                                 ; [2] Inhabilita interrupciones
        ldx #{1+SCI_DATA_BITS}              ; [2] cargo el registro x (START + N-
                                            ; BITS-DATA)
        clc                                 ; [1] limpio el bit de carry
SCI_DATA1011.C                              ; Label de bit de datos

        bcc SCI_START1011.A                 ; [3] La primera vez salta al bit de
                                            ; inicio
        bset SCI_TX_PIN,SCI_TX_PORT         ; [4] Levanta la línea dependiendo si C
                                            ; = 1
        bra SCI_CHAR1011.B                  ; [3] Salta a recibir otro bit del dato
SCI_START1011.A                             ; Label de bit inicio
        bclr SCI_TX_PIN,SCI_TX_PORT         ; [4] la primera vez este es el bit de
                                            ; inicio
        brn *                               ; [3] pierde tiempo
SCI_CHAR1011.B                              ; Label del byte
        lda #SCI_BAUD                       ; [2] Carga con el baudio seleccionado
        jsr SCIBaudDelay                    ; [7a+4+5] 'a' viene dado por la tabla al
                                            ; principio,  define el baudio
        ror SCI_SCDR                        ; [4]Al rotarlo va saliendo por el carry
                                            ; un "1" o "0"
        decx                                ; [1]Decremento la cuenta de X (Start
                                            ; + dato)
        bne SCI_DATA1011.C                  ; [3]Si X es diferente de 0 salta a data
                                            ; bit
SCI_STOP1011.D                              ; Label del Bit de parada
        lda #{SCI_BAUD*SCI_STOP_BITS}       ; [2]Carga con el baudio respectivo
        jsr SCIBaudDelay                    ; [7a+9] Espero medio baudio
        bset SCI_TX_PIN,SCI_TX_PORT         ; [4] Habilita nuevamente la línea en
                                            ; alto
        cli                                 ; [4] Habilita interrupciones
        rts                                 ; [3] Regreso de la subrutina
```

```
;===========================================
; SCIRXGET    : Recibe un byte por serial
; OBJETIVO    : Recibir un caracter
; ENTRADA     : Ninguna
; SALIDA      : SCI_SCDR   contiene el ca-
;                          racter a recibi-
;                          do por serial.
; REGISTROS
; AFECTADOS   : CCR, DDRX, PORTX, X
;===========================================
SCIRxGet                        ; Label de Recibir byte
        ldx #SCI_DATA_BITS      ; [2] Cargo X con 8 para los bits de
                                ; datos
        lda #SCI_BAUD           ; [2] A retardar 1/2 baudio
        jsr SCIBaudDelay        ; [7*a+4+5] Espero medio baudio
        brn *                   ; [3] Pierde tiempo
        brn *                   ; [3] Pierde tiempo
        clc                     ; [1] Borro el carry
SCI_READ1011.E
        lda #SCI_BAUD           ; [2] Carga selector del baudio
        brn *                   ; [3]
        jsr SCIBaudDelay        ; [7*a+9] Demora
        bil SCI_ZERO1011.F      ; [3]
        lda #1                  ; [2] NO, Carga acumulador con 1
SCI_ZERO1011.F
        nop                     ; [1] Pierde tiempo

        rora                    ; [1] Roto lo que hay en porta y sale
                                ; por el carrry
        ror SCI_SCDR            ; [4] Lo llevo a dato
        decx                    ; [1] Decremento X hasta 0
        bne SCI_READ1011.E      ; [3] Si no es cero entonces salto al
                                ; otro bit de datos
SCI_STOP1011.G
        lda #SCI_BAUD           ; [2] Cargo con el número del baudio
                                ; seleccionado
        jsr SCIBaudDelay        ; [7*a+9] Salto a esperar el baudio
        brn *                   ; [3] saltar nunca
        bih SCI_DONE1011.H      ; [3] Si está en alto
        lda #'\'                ; [2] Caracter de Error = 7C
        sta SCI_SCDR            ; [3] Lo almaceno en dato
SCI_DONE1011.H
        rts                     ; [4] Regreso de la subrutina
```

```
;===========================================
; SCITXPUTLINE
;                : Envía una línea de bytes por
;                  serial
; OBJETIVO       : Inicializa TXD o RXD
; ENTRADA        : H:X   contiene la posición
;                        inicial de la cadena
;                        a enviar.
; SALIDA         : Ninguna
; REGISTROS
; AFECTADOS      : CCR, DDRX, PORTX
;===========================================
SCITxPutLine
        lda ,x                          ; Carga el valor apuntado
        beq SCI_OUT1011.I               ; Si es 0 (fin de línea), salir
        sta SCI_SCDR                    ; De lo contrario, almacena en el
                                        ; registro del SCI
        pshx                            ; Empuja H
        ldx #SCI_ACK_DELAY              ; NOTA:  Retardo de reconocimiento
                                        ; [omitir estas dos líneas]
        dbnzx *                         ; NOTA:  Salta en sí hasta ser 0   [si
                                        ; solo se transmite   ]
        clrx                            ; Borra X
        jsr SCITxPut                    ; Transmite 1 byte
        pulx                            ; Recupera X
        aix #1                          ; Apunta al siguiente caracter
        bra SCITxPutLine                ; Envía el siguiente
SCI_OUT1011.I                           ; Etiqueta de Salir
        rts                             ; Retorna

;===========================================
; SCIBaudDelay
;                : Retardo programable de SCI
; OBJETIVO       : Retarda 7*a+4
; ENTRADA        : A    es el baudio programado
; SALIDA         : Ninguna
; REGISTROS
; AFECTADOS      : ACCA
;===========================================
SCIBaudDelay                            ; Label de la subrutina del baudio
        nop                             ; [1]-|
        nop                             ; [1] |-> 7*a-
        tsta                            ; [1] |      |
        deca                            ; [1] |      |->7*a+4... Delay de baudio
        bne SCIBaudDelay                ; [3]-      |
        rts                             ; [4]---------
```

```
;===========================================
; SCIBUFINIT   : Inicializa el buffer anillo
; OBJETIVO     : Cabeza y cola en posición i-
;                nicial; contador en cero.
; ENTRADA      : Ninguna
; SALIDA       : Ninguna
; REGISTROS
; AFECTADOS    : CCR, H:X
;===========================================
SCIBufInit
        sei                             ; Deshabilita interrupciones
        sthx SCI_FIFO_HD                ; Cabeza y cola apuntando...
        sthx SCI_FIFO_TL                ; ... al inicio del buffer
        cli                             ; Habilita interrupciones
        clr SCI_FIFO_CTR                ; Borra contador
        rts                             ; Retorna

;===========================================
; SCIBUFEMPTY
;              : Verifica si el buffer está
;                vacío.
; OBJETIVO     : Pregunta si hay un nuevo con-
;                tenido del buffer.
; ENTRADA      : Ninguna
; SALIDA       : A = 1        si el buffer está
;                               vacío.
;                A = 0        si el buffer no está
;                               lleno.
; REGISTROS
; AFECTADOS    :  ACCA
;===========================================
SCIBufEmpty
        lda SCI_FIFO_CTR                ; Carga contador de bytes del buffer
        bne SCI1011.J                   ; No es 0?, SI, salir
        lda #1                          ; Carga 1, buffer vacío
        bra SCI1011.K                   ; Salir

SCI1011.J
         clra                           ; NO, Carga 0, buffer no vacío
SCI1011.K
         rts                            ; Retorna
```

```
;===============================================
; SCIBUFFULL :  Verifica si el buffer está
;               lleno.
; OBJETIVO   :  Verifica si se agotó la capa-
;               cidad del buffer.
; ENTRADA    :  Ninguna
; SALIDA     :  A = 1        si el buffer está
;                            lleno.
;               A = 0        si el buffer no está
;                            lleno.
; REGISTROS
; AFECTADOS  :  ACCA
;===============================================
SCIBufFull
        lda SCI_FIFO_CTR            ; Carga contador de bytes del buffer
        cmp #SCI_MAX_CHAR           ; Compara con la capacidad máxima
                                    ; del buffer
        bge SCI1011.L               ; Sobrepasa?, SI, activar bandera
        clra                        ; NO, Carga 0, buffer no lleno
        bra SCI1011.M               ; Salir
SCI1011.L
         lda #1                     ; Carga 1, buffer lleno
SCI1011.M
        rts                         ; Retorna

;===============================================
; SCIBUFPUT  :  Impone 1 byte en el buffer
; OBJETIVO   :  Arroja 1 byte en el bufer
; ENTRADA    :  A            es el SCI_SCDR o dato
;                            a enviar al buffer
; SALIDA     :  Ninguna
; REGISTROS
; AFECTADOS  :  ACCA, H:X, CCR
;===============================================
SCIBufPut
        tax                         ; Transfiere SCI_SCDR a X
        jsr SCIBufFull              ; Verifica capacidad del buffer
        bne SCI1011.N               ; Si está lleno, salir
        inc SCI_FIFO_CTR            ; incrementa contador
        txa                         ; Recupera el dato
        ldhx SCI_FIFO_HD            ; Apunta al lugar a tranferir al buffer
        sta ,x                      ; Almacena en la posición apuntada
        aix #1                      ; incrementa el puntero
        sthx SCI_FIFO_HD            ; Actualiza cabeza
        cphx #SCI_FIFO_MAX          ; Compara con la capacidad máxima
        bne SCI1011.N               ; NO, no son iguales, salir
        ldhx #SCI_FIFO              ; SI, Carga posición inicial

        sthx SCI_FIFO_HD            ; Apunta cabeza al inicio nuevamente
SCI1011.N
        rts                         ; retorna
```

```
;===========================================
; SCIBUFGET  :  Recoge 1 byte en el buffer
; OBJETIVO   :  Arroja 1 byte en el bufer
; ENTRADA    :  Ninguna
; SALIDA     :  A = 0          si no se recupera
;                              del buffer... ó
;               A              contiene el byte re-
;                              cuperado del buffer
; REGISTROS
; AFECTADOS  :  ACCA, H:X, CCR
;===========================================
SCIBufGet
        sei                             ; Deshabilita interrupciones
        jsr SCIBufEmpty                 ; Verifica si hay un nuevo dato en el
                                        ; buffer
        bne SCI1011.O                   ; Si está vacío, salir
        dec SCI_FIFO_CTR                ; decrementa contador
        ldhx SCI_FIFO_TL                ; Apunta al lugar a remover el byte del
                                        ; buffer
        lda ,x                          ; Recupera de la posición apuntada
        aix #1                          ; incrementa el puntero
        sthx SCI_FIFO_TL                ; Actualiza cola
        cphx #SCI_FIFO_MAX              ; Compara con la capacidad máxima
        bne SCI1011.P                   ; NO, no son iguales, salir
        ldhx #SCI_FIFO                  ; SI, Carga posición inicial
        sthx SCI_FIFO_TL                ; Apunta cola al inicio nuevamente
        bra SCI1011.P                   ; salir
SCI1011.O
        clra                            ; si el buffer está lleno, retorna 0
SCI1011.P
        cli                             ; Habilita interrupciones
        rts                             ; retorna
```

```
;===========================================
; SCICHKSUM  :  Verifica la integridad de la
;               información
; OBJETIVO   :  Verifica el paquete
; ENTRADA    :  H:X          contiene la posición
;                            inicial de la cadena
;                            a calcular.
; SALIDA     :  A            contiene la integri-
;                            dad de la información
; REGISTROS
; AFECTADOS  :  ACCA, H:X, RAM
;===========================================
SCIChkSum
        clr SCI_CHK_CTR             ; Borra contador
        lda ,x                      ; Carga posición inicial
SCI1011.R
         aix #1                     ; Incrementa puntero
        inc SCI_CHK_CTR             ; Incrementa variable
        psha                        ; Empuja A
        lda SCI_CHK_CTR             ; Carga contador

        cmp #SCI_MAX_CHAR           ; compara con el máximo
        pula                        ; Recupera A
        beq SCI_OUT1011.Q           ; Si es igual, salir
        add ,x                      ; Sumar Siguiente con anterior
        bra SCI1011.R               ; Repetir
SCI_OUT1011.Q                       ; Etiqueta de Salir
        eor #$FF                    ; XOR para completar CheckSum
        rts                         ; Retorna
```

```
;=========================================
; SCITx485      : Envía en formato RS-485
; OBJETIVO      : Envía una trama en formato
;                 RS-485 con forma
;                 ID-TRAMA-CHECKSUM
;                 ID          = 2 bytes
;                 TRAMA       = Definida por el
;                               macro SCI_MAX_CHAR
;                 CHECKSUM    = 1 byte
; ENTRADA       : A       contiene la posición
;                         del ID dado por una
;                         tabla SCI_ID.
;                 H:X     es la dirección o
;                         puntero del mensaje
;                         a enviar.
; SALIDA        : A = 1 si el envío es satis-
;                         factorio.
;                 A = 0 si el envío no fue
;                         satisfactorio
; NOTA          : A = 0 puede verificar
;                         el registro SCI_SCDR y si el
;                         mismo es 's', hubo un error
;                         en envío.
; REGISTROS
; AFECTADOS     : ACCA, H:X, RAM
;=========================================
SCITx485
        pshh                            ; Empuja H a pila
        pshx                            ; Empuja X a pila
        jsr SCIRxDis                    ; Deshabilita receptor
        jsr SCITxEn                     ; Habilita transmisor
        clrh                            ; Borra H
        ldx #2T                         ; A = 2
        mul                             ; Calcula puntero
        tax                             ; Transfiere puntero a X
        lda SCI_ID,x                    ; Carga ID.H
        sta SCI_SCDR                    ; Almacena en el reg. de datos
        pshx                            ; Preserva puntero en pila

        jsr SCITxPut                    ; Envía
        ldx #SCI_ACK_DELAY              ; Retardo de reconocimiento
        dbnzx *                         ; Salta en sí hasta ser 0
        pulx                            ; Recupera puntero en pila
        aix #1                          ; Apunta a ID.L
        lda SCI_ID,x                    ; Carga ID.L
        sta SCI_SCDR                    ; Almacena en el reg. de datos
        jsr SCITxPut                    ; Envía
        jsr SCITxDis                    ; Deshabilita transmisor
        jsr SCIRxEn                     ; Habilita receptor
        ldx #SCI_ACK_DELAY              ; Retardo de reconocimiento
SCI_GET1011.S
        decx                            ; decrementa contador
        cpx #0                          ; compara contra 0
```

```
            beq SCI_OUT1011.U           ; SI, es igual, salir
            jsr SCIBufEmpty             ; Dato en buffer?
            bne SCI_GET1011.S           ; NO, Esperar
            jsr SCIBufGet               ; SI, Recoger el dato
            cmp #'@'                    ; Compara con caracter de atención
            beq SCI_FRAME1011.T         ; SI, es igual, enviar trama
            bra SCI_OUT1011.U           ; NO, Salir
SCI_FRAME1011.T
            jsr SCIRxDis                ; Deshabilita receptor
            jsr SCITxEn                 ; Habilita Transmisor
            ldx 1,SP                    ; X = Puntero bajo
            lda 2,SP                    ; A = Puntero alto
            psha                        ; Empuja A
            sei                         ; Inhabilita interrupciones
            pulh                        ; Recupera en H = Puntero alto
            jsr SCITxPutLine            ; Envía Trama
            pulx                        ; Recupera Puntero Bajo
            sei                         ; Inhabilita interrupciones
            pulh                        ; Recupera Puntero Alto
            jsr SCIChkSum               ; Calcula Checksum
            sta SCI_SCDR                ; Almacena en el registro de envío
            jsr SCITxPut                ; Envía Checksum
            jsr SCITxDis                ; Deshabilita transmisor
            jsr SCIRxEn                 ; Habilita Receptor
SCI_GET1011.V
            jsr SCIBufEmpty             ; Dato en Buffer?
            bne SCI_GET1011.V           ; NO, Esperar
            jsr SCIBufGet               ; SI, Recoger dato
            cmp #'@'                    ; Compara con caracter de
                                        ; confirmación
            beq SCI_OUT1011.W           ; SI, es igual, salir
            sta SCI_SCDR                ; Almacena para verificar si se envía
                                        ; nuevamente
            cmp #'s'                    ; NO, Compara con caracter de envío
                                        ; nuevamente
            bne SCI_OUT1011.W           ; SI, salir
            bra SCI_OUT1011.Z           ; NO, Enviar error
SCI_OUT1011.W
            lda #1                      ; A = 1, envío satisfactorio
            bra SCI_OUT1011.Y           ; Salir
SCI_OUT1011.U
            ais #2                      ; Ajusta valor de pila
SCI_OUT1011.Z
            clra                        ; A = 0, No es el ID

SCI_OUT1011.Y
            rts                         ; Retorna
```

```
;===========================================
; SCIRx485      : Recibe en formato RS-485
; OBJETIVO      : Recibe una trama en formato
;                 RS-485 con forma
;                 ID-TRAMA-CHECKSUM
;                 ID          = 2 bytes
;                 TRAMA       = Definida por el
;                               macro SCI_MAX_CHAR
;                 CHECKSUM    = 1 byte
; ENTRADA       : Ninguna
; SALIDA        : A = 1       si la recep. es
;                             satisfactoria
;                 A = 0       si la recep. no fue
;                             satisfactorio
; NOTA          : Si A = 0,   se puede verificar
;                             el registro SCI_SCDR y si
;                             el mismo es 's', hubo un
;                             error en envío.
; REGISTROS
; AFECTADOS     : ACCA, H:X, RAM
;===========================================
SCIRx485
        jsr SCITxDis                    ; Deshabilita transmisor
        jsr SCIRxEn                     ; Habilita el receptor
SCI_GET1011.A1
        jsr SCIBufEmpty                 ; No hay dato en buffer?
        bne SCI_GET1011.A1              ; SI no hay, esperar
        jsr SCIBufGet                   ; Existen datos, recoger
        cmp #SCI_ID_H                   ; Comparar con ID alto
        beq SCI_GET1011.C1              ; SI, es igual, seguir
        clra                            ; A = 0, Avisa error
        bra SCI_OUT1011.B1              ; NO, diferente, salir
SCI_GET1011.C1
        jsr SCIBufEmpty                 ; No hay dato en buffer?
        bne SCI_GET1011.C1              ; SI no hay, esperar
        jsr SCIBufGet                   ; Existen datos, recoger
        cmp #SCI_ID_L                   ; Comparar con ID alto
        beq SCI_RX_FRAME1011.D1         ; SI, es igual, seguir
        clra                            ; A = 0, Avisa error
        bra SCI_OUT1011.B1              ; NO, salir
SCI_RX_FRAME1011.D1
        mov #'@',SCI_SCDR               ; Enviar caracter de Confirmación
        jsr SCIRxDis                    ; Deshabilita receptor
        jsr SCITxEn                     ; Habilitar transmisor
        jsr SCITxPut                    ; Enviar
        jsr SCITxDis                    ; Deshabilita transmisor
        jsr SCIRxEn                     ; Habilita receptor
SCI_GET1011.E1
        jsr SCIBufFull                  ; Buffer Lleno?
        beq SCI_GET1011.E1              ; NO, esperar
        ldhx #SCI_FIFO                  ; SI, Carga posición del FIFO
        jsr SCIChkSum                   ; Calcula Checksum
        mov SCI_SCDR,SCI_CHK_CTR        ; Checksum recibido a var. temporal
```

```
        cmp SCI_CHK_CTR         ; Compara checksum calculado con
                                ; recibido
        beq SCI_OUT1011.G1      ; SI, son iguales, salir
        mov #'s',SCI_SCDR       ; Caracter de enviar nuevamente
        jsr SCIBufInit          ; Reinicia el buffer
        clra                    ; A = 0, Avisa error
        bra SCI_OUT1011.F1      ; Salir y decidir si se recibe
                                ; nuevamente
SCI_OUT1011.G1
        mov #'@',SCI_SCDR       ; Caracter de confirmación
        lda #1                  ; A = 1, Envío satisfactorio
SCI_OUT1011.F1
        jsr SCIRxDis            ; Deshabilita receptor
        jsr SCITxEn             ; Habilita el transmisor
        jsr SCITxPut            ; Envía
SCI_OUT1011.B1
        jsr SCITxDis            ; Deshabilita transmisor
        jsr SCIRxDis            ; Deshabilita el receptor
        rts                     ; Retorna
```

Listado 85. NT1011 – "SerialCom" – SCI.inc. Rutinas de Comunicación Serial para microcontroladores sin SCI. Para mayor información, referirse a la NT0026, sección de Comunicación Serial.

```
;=========================================
; DELAY       :  Genera un retardo de tiempo
; OBJETIVO    :  Retardo de tiempo, base 1ms
; ENTRADA     :  H:X = Retardo en ms
; SALIDA      :  H:X = 0
; REGISTROS
; AFECTADOS   :  H:X
; USO         :
;                MIN = H:X = 1T
;                MÁX = H:X = 65535T
;                ldhx #500T
;                jsr Delay ; retarda 0.5 seg
;=========================================
Delay
        cphx #0                         ; [3] Compara con 0
        beq DELAY0009.A                 ; [3] Salir si es 0
        pshx                            ; [2] Salva X en la pila
        pshh                            ; [2] Salva H en la pila
        ldhx #DELAY49152                ; [3] Carga constante de bucle fino
Delay0009.B
        aix #-1                         ; [2] Decrementa H:X en 1
        cphx #0                         ; [3] LLegó a cero (0)
        bne Delay0009.B                 ; [3] Si no es igual, salta a Delay0
        pulh                            ; [2] Si es igual, recupera H de la pila
        pulx                            ; [2] Recupera X de la pila
        aix #-1                         ; [2] Decrementa H:X en 1
        cphx #0                         ; [3] LLegó a cero (0)
        bne Delay                       ; [3] Si no es igual, salta a Delay
DELAY0009.A
        rts                             ; [4] retorna
```

Listado 86. NT1011 – "SerialCom" – DELAY.inc. Archivo que contiene las funciones de retardo programables.

```
;=====================================================================
; ARCHIVO      :  TABLES.inc
; PROPÓSITO    :  Tablas de búsqueda predefinidas por el usuario
; LENGUAJE     :  IN-LINE ASSEMBLER
; ---------------------------------------------------------------------
; HISTORIAL
;   DD MM AA
;   06   09   04   Creado.
;   06   09   04   Modificado.
;=====================================================================

;=============================================
;       IDENTIFICADORES PARA ESCLAVOS
;=============================================
SCI_ID
      db '00','01','02',0                       ; IDs de esclavos

;=============================================
;            TABLA DE MENSAJES
;=============================================
MENSAJE
      db '1234567890123456',0              ; Mensaje
MENSAJE1
      db '9374108475029361',0
```

Listado 87. NT1011 – "SerialCom" – TABLES.inc. Tablas de búsqueda de mensajes a enviar.

```
;=================================================================
; 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.
;  30  10  04   Modificado.
;=================================================================

;=========================================
;                Interrupción Externa
;=========================================
;=========================================
; IRQL         : Interrupción externa
; OBJETIVO     : Recibe 1 byte del serial.
; ENTRADA      : Ninguna
; SALIDA       : Ninguna
; REGISTROS
; AFECTADOS    : INTSCR
;=========================================
IRQL                                    ; IRQ (Bajo)
      bset BIT1,INTSCR                  ; [4] Enmascara la int.
      jsr SCIRxGet                      ; [5] Recibe
      lda SCI_SCDR                      ; Carga caracter a insertar en
                                        ; el buffer
      jsr SCIBufPut                     ; Impone en el buffer
      jsr IRQ1Ack                       ; Reconoce la int.
      rti                               ; Retorna
```

Listado 88. NT1011 – "SerialCom" – INTERRUPTSJL3.inc. Plantilla para interrupciones de microcontroladores. Para mayor información, referirse a la NT0026.

```
;=====================================================================
; 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 Interrupción Externa
;=============================================
        org IRQH                        ; IRQ (Alto)
        dw IRQL                         ; IRQ (Bajo)

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

Listado 89. NT1011 – SerialCom – VECTORSJL3.inc. Activa los vectores de interrupción a utilizar.

3.11.12 Conclusión

En el estándar RS-485 no se hace mención específica del protocolo, más bien de la capa física de conexión. Es por esto que fue usado un protocolo creado por el autor para transmisión serial, basado en el esquema de transmisión 8N1, este consta de un bit de inicio, 8 bits de datos y de uno a dos bits de parada. La transmisión y recepción implementados es estilo “half dúplex” y con la ayuda del integrado 75176, se puede lograr más de 1.2km entre conexión de dispositivos y esclavizar entre sí más de 32 unidades además del maestro.

El propósito de la industria es utilizar este tipo de transmisión de data multipunto indiscriminadamente mientras se pueda y los costos lo justifiquen, por lo general, la red RS485, está distribuida entre los esclavos. Por otro lado, este no es el único protocolo de red que se utiliza unitariamente o entre conjunto. Aunque RS-485 es uno de los protocolos más utilizados, el TCP/IP es mucho más viable y más rápido, además que no requiere de mucho cableado, otros también mezclan un sistema BACNet.

Si se requiere de algún fabricante que trabaje bajo este tipo de estándares se recomienda visitar el sitio de la compañía “Johnson Controls”, el cual utiliza el protocolo protocolo denominado como “Johnson Controls N2”.

3.11.13 Referencias

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

Pág. 30 - Registros de Puertos de Entrada – Salida y de Direccionamiento de Datos.
Págs. 137 – 147 – Información de los puertos de Entrada/Salida
Págs. 149 a 153 – Módulo de Interrupción Externa (IRQ1)

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

Pág. 105, Instrucción BIH
Pág. 128, Instrucción BIL

3.11.13.3 Página de Microcontroladores de esta nota
(a) http://www.geocities.com/issaiass/index.htm

3.11.13.4 "Embedded Systems Building Blocks, Second Edition" - "Complete and Ready-to-Use Modules in C"

Autor: *Jean J. Labrosse*
Recurso: *Capítulo 11 – Asyncronous Serial Communications – Págs. 409 a 420*

3.11.13.5 Estándar RS-232
(a) http://www.euskalnet.net/shizuka/rs232.htm
(b) http://www.ctv.es/pckits/tpseriee.html
(c) http://www.geocities.com/neonluzia/siteperifericos/rs232.htm
(d) http://www.disca.upv.es/aperles/web51/modulos/modulo4/m_comunic.html
(e) http://148.237.96.15/noticiero/tips/interface_rs_232.htm

Información sobre el estándar, conectores y el puerto serie de la PC, niveles lógicos y conexiones DTC y DCE.

(f) http://www.astrosurf.com/cavadore/Hardware/rs232ex/max232.pdf

Data del fabricante de la Max 232; dispositivo para elevar niveles TTL.

3.11.13.6 Estándar RS-485

(a) http://www.jameco.com/Jameco/Products/ProdDS/50964.pdf

Data del fabricante e integrado para RS-485.

(b) http://www.jdrichards.com/rs485.html

Página de RS-485 para un microcontrolador Parallax por medio de Tiny Basic.

(c) http://www.bb-elec.com/bbelec/literature/tech/485appnote.pdf

(d) http://www.grantronics.com.au/docs/TIA-EIA-485.pdf

(e) http://www.epanorama.net/links/tele_interface.html#rs485

Notas de Aplicación sobre el estándar RS485, diagramas de conexión, especificación de la capa física, topología, tipo de cable, velocidad de transmisión, aterrizaje, longitud de cable.

(f) http://www.rs485.com/pfaq.html

Protocolos de Comunicación serial y el RS485. Esta empresa vende productos de soporte para la implementación practica de aplicaciones basadas en el RS485 (Ver pagina principal)

(g) http://www.obvius.com/documentation/faq/modbus.html

Aspectos prácticos de la instalación de un sistema RS485 de una empresa

(h) http://www.electroind.com/support/faq.html

Aspectos prácticos de la instalación de un sistema RS485 de una empresa

(i) http://www.bb-europe.com/tech_articles/faq_check_2wire_rs485.asp

Como verificar el funcionamiento de un sistema RS485 de dos hilos

(j) http://www.codebluecommunications.com/Documents/cb_serial_port_adapter_gen2_faq_1_0.pdf

RS485 y la comunicación inalámbrica basada en uno de los protocolos que se utilizan en la industria “Bluetooth”.

3.11.13.7 Edificios Inteligentes
(a) http://www.jci.com/cg/what_flash.asp

Sitio de la empresa Jonson Controls

(b) http://www.coggan.com/intelligentbuildings.html

Vínculos / Información sobre edificios inteligentes

(c) http://www.energymanagement.umich.edu/utilities/energy_management/BAS_FAQ.html

Sobre automatización de edificios desde una perspectiva practica

(d) http://www.smarthomeforum.com/start/faq.asp?ID=2

FAQ sobre el protocolo BacNet

(e) http://doityourself.com/computers/x10.htm

FAQ sobre el protocolo X10 que se usa en la automatización de residencias.

(f) http://www.automationfaq.com/fom-serve/cache/240.html

FAQ sobre el cableado con cable tipo Categoría 5 o CAT5 en residencias

3.11.13.7 "Buffer" Anillo
(g) http://delta.cs.cinvestav.mx/~pmejia/softeng/44

Buffer Anillo para transmisión o recepción de información

3.11.14 Problemas Propuestos

3.11.14.1 Realizar un "software" que monitoree un sistema inteligente de una red de sensores de temperatura y los despliegue en un conjunto de siete segmentos.

3.11.14.2 Controle el encendido y apagado de luces de una red inteligente.

3.11.14.3 Realice un sistema de Monitoreo de la propuesta 3.11.14.1, pero ahora con intefase hacia la PC. Nota: Utilizar una MAX232.