

**INTERFACE DE
COMUNICACION
SERIE
(SCI)
Módulo Serial
Asincrónico**

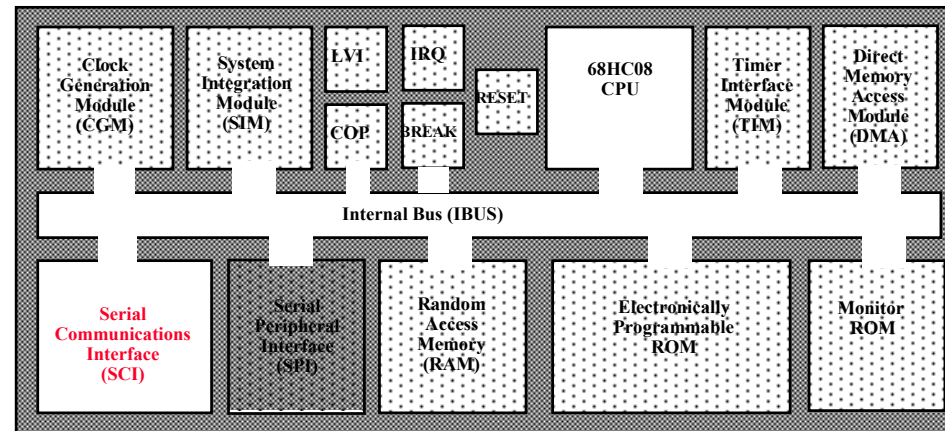
Curso de Microcontroladores

Familia HC908 Flash de Motorola

Parte II

ING. DANIEL DI LELLA DDFAE For Motorola Products

Serial Communication Interface Module



SCI

- Full duplex, Alta Velocidad, asincrónico programable 8 o 9 bit largo de caracter.
- Dos métodos de “ receiver wakeup”.
- Pedidos de Interrupción separados para Rx y Tx .
- Habilitación separada del Transmisor y el Receptor.
- Polaridad programable de la salida del Transmisor.

- Operación dirigida por INT'S con ocho Flags de interr.:

- Transmitter Empty
- Transmission Complete
- Receiver Full
- Idle Receiver Input
- Receiver Overrun
- Noise Error
- Framing Error
- Parity Error

- Modo de operación “Low power”

- Framing Error Detection en el receptor

- Checkeo de Paridad por Hardware

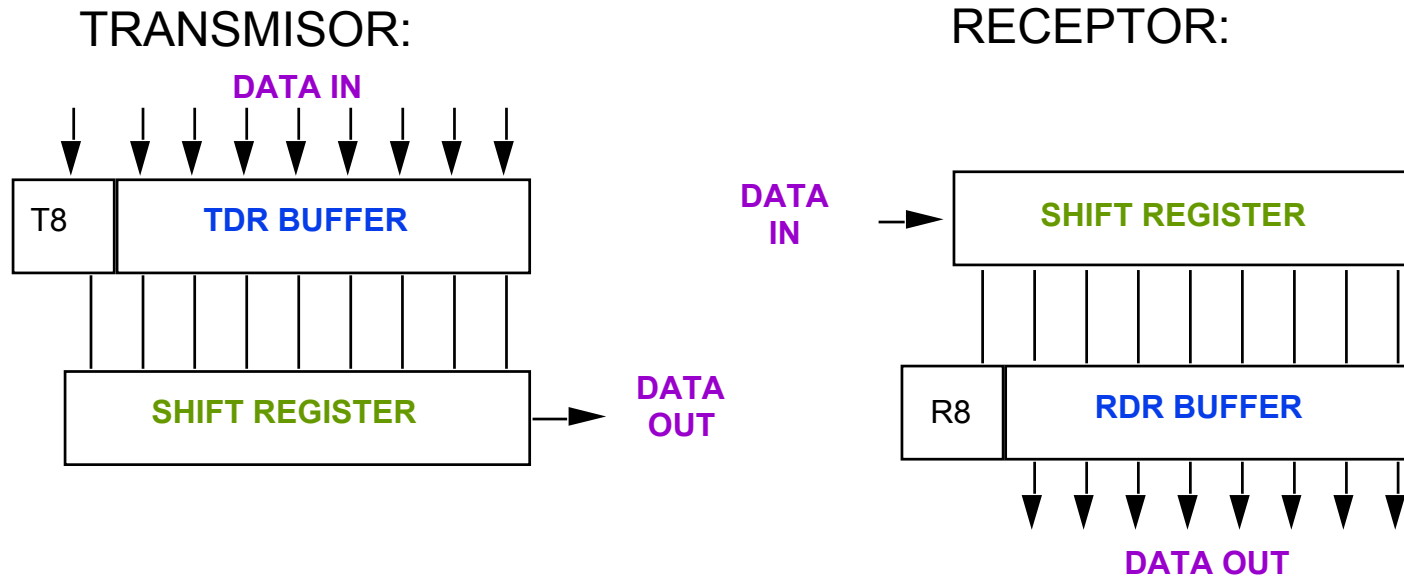
Curso de Microcontroladores

Familia HC908 Flash de Motorola

Parte II

ING. DANIEL DI LELLA DDFAE For Motorola Products

Sistema Doble “Buffer”



- **SCTE Flag** se setea cada vez que un nuevo dato es transferido desde el **TDR Buffer** al **transmit serial shift register**.

- **SCRF Flag** se setea cada vez que un nuevo dato es transferido desde el **serial shift register** al **RDR Buffer**.

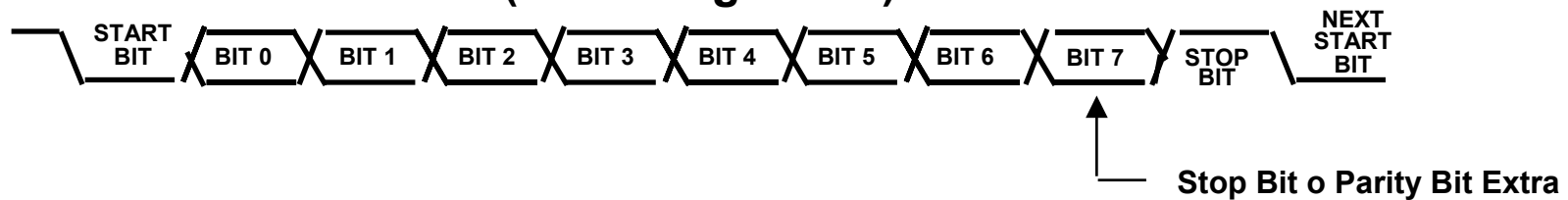
SCDR : Un SOLO registro que.....

- Al “escribir” el **SCDR** – Se escribe el **TDRx Buffer**
- Al “leer” el **SCDR** --- Se lee el **RDRx Buffer**

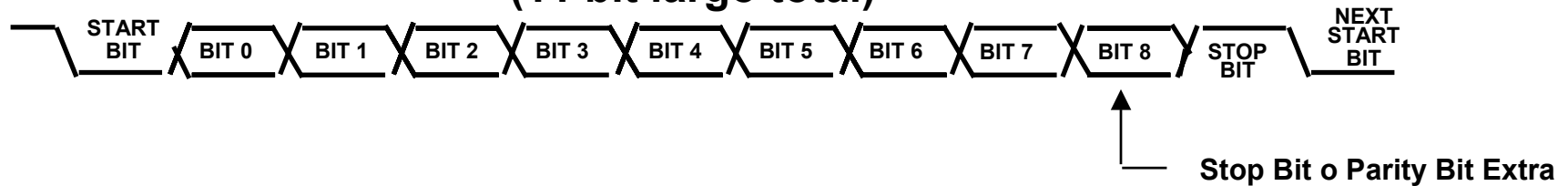
Lo básico en comunicaciones seriales...

Formato de Datos SCI

Formato de datos 8 - Bits (10 bit largo total)



Formato de Datos 9 - Bit (11 bit largo total)



Lo básico en comunicaciones seriales...

Caracteres de Datos Especiales:

Break - No tiene Start o Stop bits, existe como un “cero” lógico por un tiempo de 10 ó 11 bit (Formato de Datos de 8 o 9 bits respectivamente)

Idle - No tiene Start o Stop bits, existe como un “uno” lógico por un tiempo de 10 ó 11 bit (Formato de Datos de 8 o 9 bits respectivamente)

Preambulo - Un carácter “idle” de sincronismo

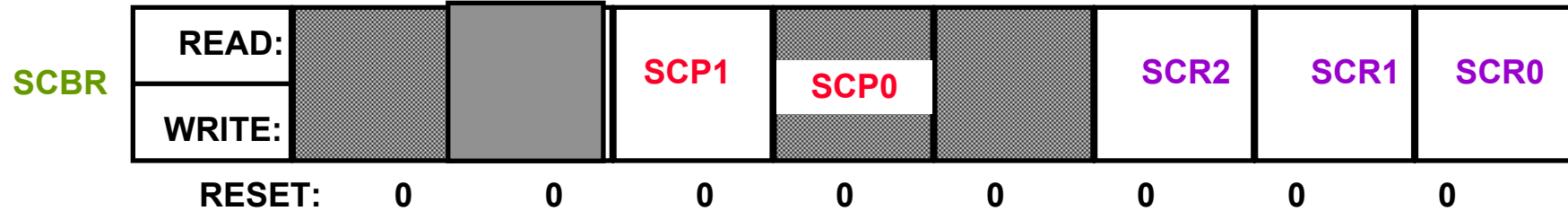
Registros I/O del SCI

Siete registros I/O controlan y monitorean la operación del SCI

- SCI Control Register 1 (SCC1)
- SCI Control Register 2 (SCC2)
- SCI Control Register 3 (SCC3)
- SCI Status Register 1 (SCS1)
- SCI Status Register 2 (SCS2)
- SCI Data Register (SCDR)
- SCI Baud Rate Register (SCBR)

El módulo de SCI de la flia. HC908, es una versión mejorada de los módulos de SCI de la flia. HC705. Las mejoras introducidas van desde un mejor manejo automático de la recepción de información serial con paridad (implementada por hardware interno) , mayor velocidad de TX / RX por medio de dos alternativas de Clocks de sincronismo (Fbus y External Clock) , hasta un mejor manejo de comunicaciones serial del tipo “Network” o Red con múltiples nodos.

Selección del Baud Rate



SCI Baud Rate Register

- Selecciona el baud rate para el transmisor del SCI
- SCI baud rate prescaler bits (SCP1,SCP0)
 - Divide la frecuencia del BUS (fBus) por un múltiplo de 64
1, 3, 4, or 13
- SCI baud rate select bits (SCR2 - SCR0)
 - Selecciona el baud rate de transmisión desde el “prescaler output”
 - Divide la frecuencia de la salida del prescaler
1, 2, 4, 8, 16, 32, 64, o 128

Ejemplo de selección del Baud Rate :

Tabla Prescaler
(Frecuencia del BUS = 4.9152 MHz) (Como ejemplo)

SCP1	SCP0	Divisor	Máximo Baud Rate
0	0	1	76.80K Baud
0	1	3	25.833K Baud
1	0	4	19.20K Baud
1	1	13	5.908K Baud

Frecuencia del Bus = 4.9152 Mhz				Máximo Baud Rate (desde tabla prescaler)	
SCR2	SCR1	SCR0	Divisor	76.80K Baud	19.20K Baud
0	0	0	1	76.80K Baud	19.20K Baud
0	0	1	2	38.40K Baud	9600 Baud
0	1	0	4	19.20K Baud	4800 Baud
0	1	1	8	9600 Baud	2400 Baud
1	0	0	16	4800 Baud	1200 Baud
1	0	1	32	2400 Baud	600 Baud
1	1	0	64	1200 Baud	300 Baud
1	1	1	128	600 Baud	150 Baud

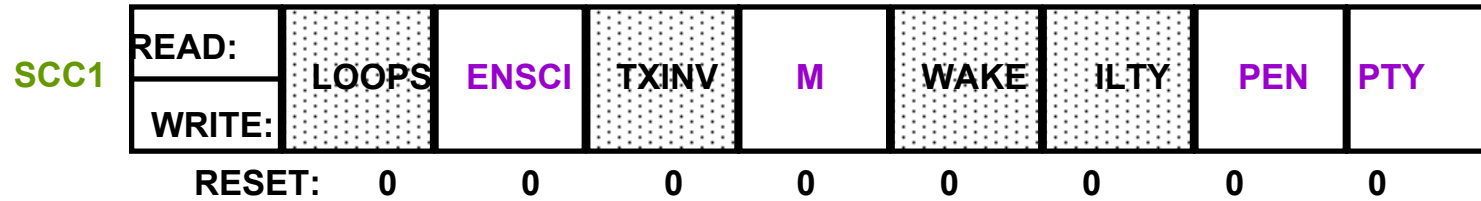
Curso de Microcontroladores

Familia HC908 Flash de Motorola

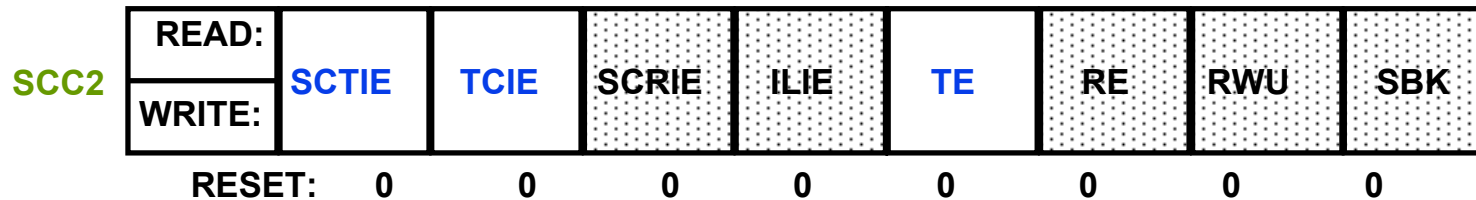
Parte II

ING. DANIEL DI LELLA DDFAE For Motorola Products

Interrupciones & configuración del Tx del SCI



- **SCI Enable (ENSCI)**
 - Habilita al SCI y al SCI baud rate generator
 - Permite al SCI ser deshab. para lower power
 - 1 = SCI habilitado
 - 0 = SCI deshabilitado
- **Character Length Select (M)**
 - 1 = 9-bit SCI characters
 - 0 = 8-bit SCI characters
- **Parity Enable (PEN)**
 - 1 = habilitado
 - 0 = Deshabilitado
- **Parity Type (PTY)**
 - 1 = Odd Parity
 - 0 = Even Parity



- **SCI Transmit Interrupt Enable (SCTIE)**
 - Interrupción cuando Tx Data Register pasa a vacío
 - El prox. byte puede ser cargado en el tx data register
 - 1 = Habilita interrupción
 - 0 = Deshabilita interrupción
- **Transmission Complete Interrupt Enable (TCIE)**
 - Interrupción cuando Tx está Completa
 - Un byte ha sido enviado
 - 1 = Habilita interrupción
 - 0 = Deshabilita interrupción
- **Transmitter Enable (TE)**
 - Habilita la operación de TX
 - Envía "preambulo"
 - 1 = Habilita el transmisor
 - 0 = Deshabilita el transmisor

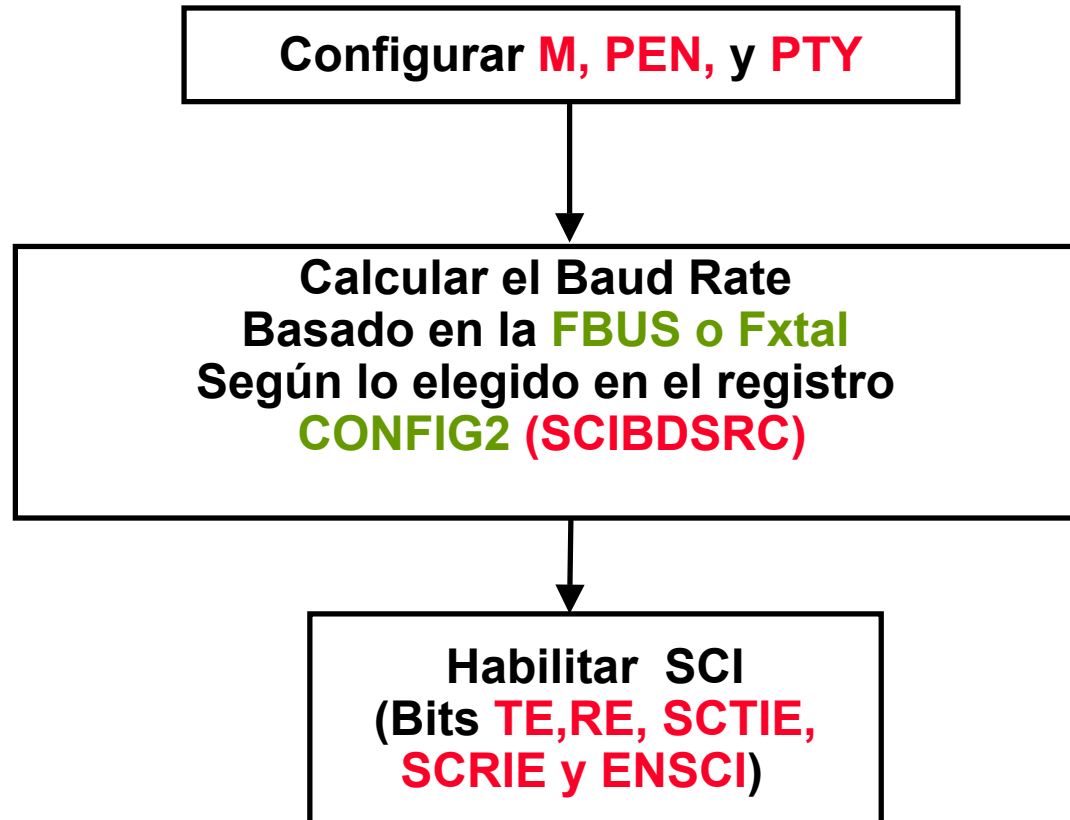
Selección del largo de caracter

M	PEN	PTY	Character Length
0	0	x	1 start + 8 (7) data + 1 (2) stop
1	0	x	1 start + 9 (8) data + 1 (2) stop
0	1	0	1 start + 7 data + Even +1 stop
0	1	1	1 start + 7 data + Odd +1 stop
1	1	0	1 start + 8 data + Even +1 stop
1	1	1	1 start + 8 data + Odd +1 stop

Con estos 3 bits (M,PEN, PTY), en el módulo de SCI, se puede setear el “largo” del caracter, si lleva paridad o no y de que tipo, y si se utiliza 1 o 2 bits de stop.

Se debe destacar que, en los módulos de SCI de los MCUs HC908, cuando se trabaja con paridad (par ó impar), la misma es generada (en la TX) y decodificada (en la RX) en forma automática por el mismo. No ocurre ello en los módulos de SCI de la flia. HC705, ya que si bién en la transmisión la paridad es generada en forma automática, en la recepción hay que implementar algoritmos, por soft, de decodificación de la paridad.

Diagrama de Flujo de la Inicialización



Flags de estado del Transmisor

SCS1	READ:	SCTE	TC	SCRF	IDLE	OR	NF	FE	PE
	WRITE:								
RESET:		1	1	0	0	0	0	0	0

SCI Status Register 1 (SCS1)

- **SCI Transmitter Empty (SCTE)**
 - Indica que el contenido del SCI Data Register há sido movido al Tx serial shift register
 - Limpiado por la lectura del SCS1 luego de escrito el SCDR
 - 1 = Data register vacio
 - 0 = Data register no vacio
- **Transmission Complete (TC)**
 - Indica que el SCDR está vacio y no hay transmisión en progreso
 - Limpiado por la lectura del SCS1 y luego la escritura del SCDR
 - 1 = No hay transmisión en progreso
 - 0 = Transmisión en progreso

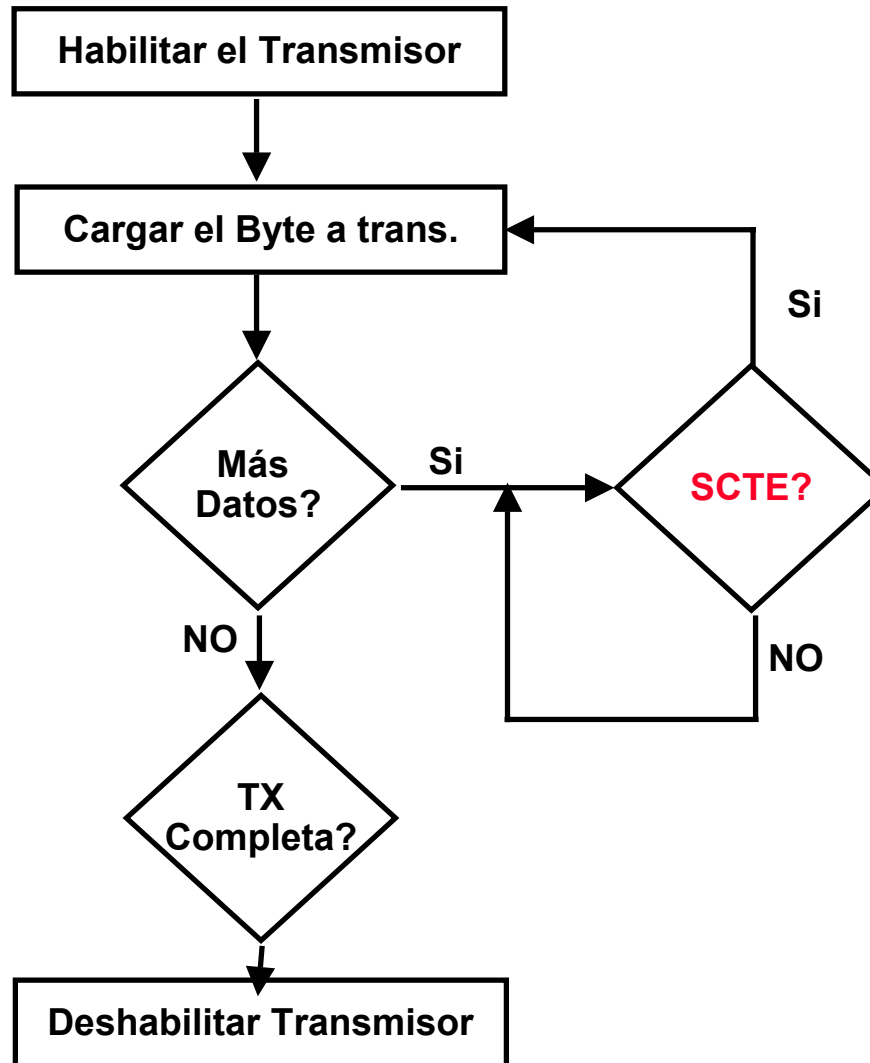
SCI Data Register

SCDR	READ:	R7	R6	R5	R4	R3	R2	R1	R0
	WRITE:	T7	T6	T5	T4	T3	T2	T1	T0
RESET:		UNAFECTED BY RESET							

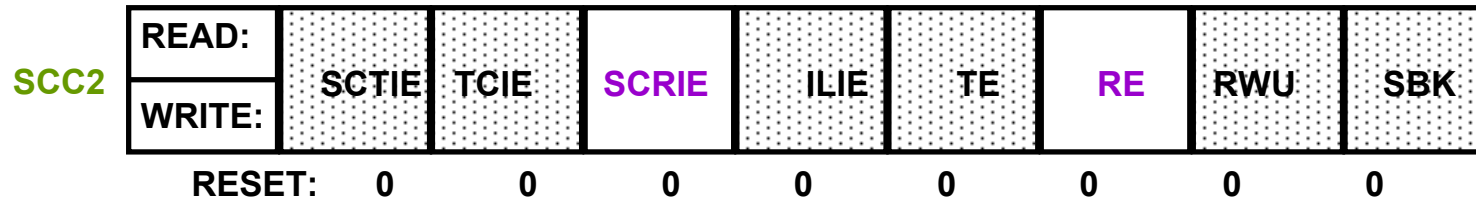
SCI Data Register (SCDR)

- Buffers de datos para el TX/RX shift registers
- Escribiendo en el SCDR se escribe datos a ser transmitidos
 - Inicia la operación de transmisión.

Diagrama de Flujo de Transmisión



Interrupciones & estados del RX del SCI



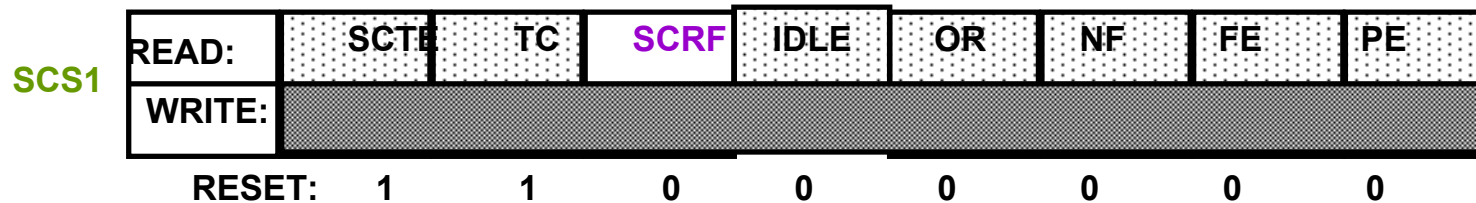
SCI Control Register 2 (SCC2)

SCI Receive Interrupt enable (SCRIE)

- Interrupción cuando se completa el Receive Data Register
- Se ha recibido un byte transmitido
 - 1 = Habilita la interrupción
 - 0 = Deshabilita la interrupción

• Receiver Enable (RE)

- Habilita la operación del receptor
 - 1 = Habilita el receptor
 - 0 = Deshabilita el receptor



SCI Status Register 1

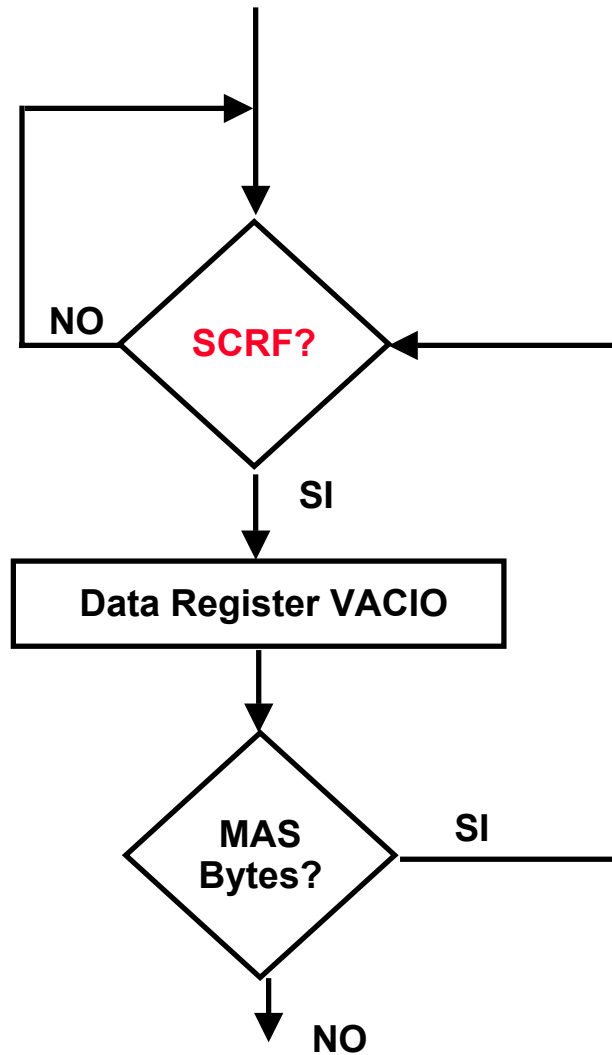
SCI Receiver Full (SCRF)

- Indica que el dato en el "receiver shift register" ha sido transferido al SCI receive data register
- Limpiado por la lectura del SCS1 y luego la lectura del SCD
 - 1 = Dato recibido disponible
 - 0 = Dato no disponible

Curso de Microcontroladores

Familia HC908 Flash de Motorola

Diagrama de Flujo de RX

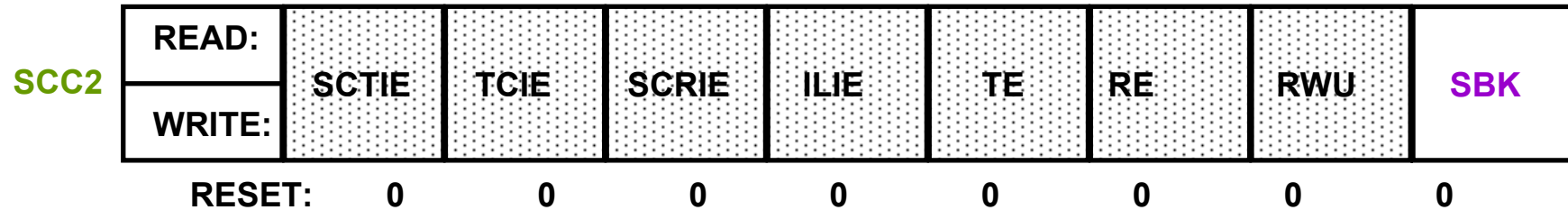


El Receptor debe estar siempre habilitado !!!

Si quiero habilitar/deshabilitar el receptor, entonces..

- La recepción de datos debe ser sincronizada sino se corre el riesgo de perder información al tener el RX deshabilitado.

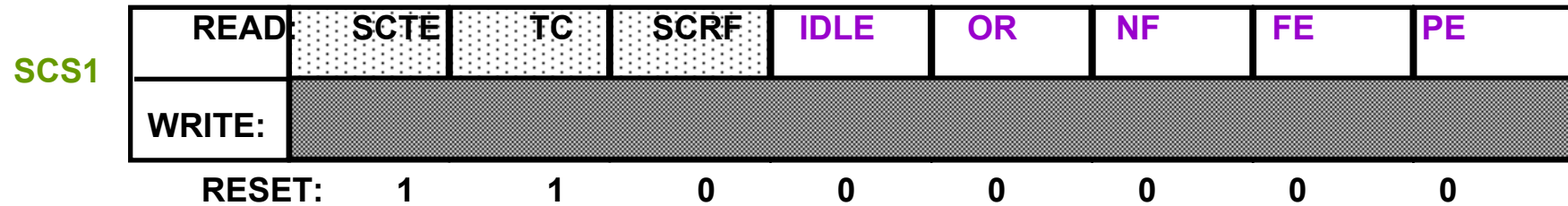
Enviando señal de “Break”



SCI Control Register 2

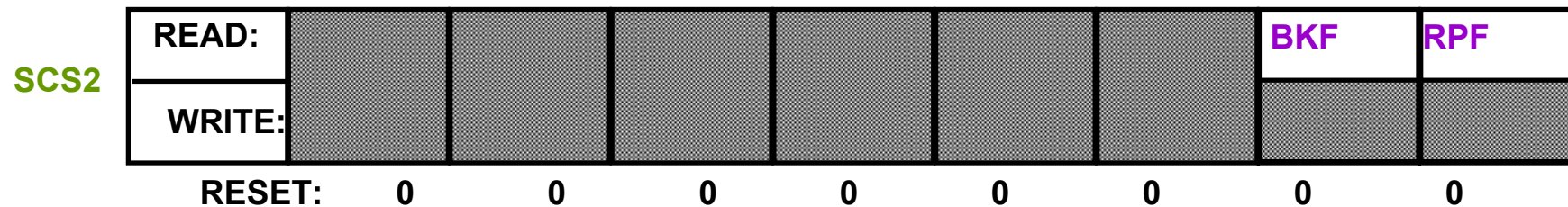
- Send Break (SBK)
 - Usado para “ganar” control del bus
 - 1 = se transmite “break character”
 - 0 = no se transmite “break character”
- Usado en redes con topología BUS de “N” nodos.

Flags de estados adicionales



SCI Status Register 1

- Receiver idle (IDLE)
- Noise Flag (NF)
- Parity Error (PE)
- Receiver overrun (OR)
- Framing Error (FE)



SCI Status Register 2

- Break (BKF)
 - Seteado cuando se detecta un “break character”
 - Limpiado por lectura del SCS2 luego de leer el SCDR
- Reception in Progress (RPF)
 - Seteado durante la búsqueda del “start bit”
 - Reseteado después de que es detectado un “stop bit o falso start bit”

Curso de Microcontroladores

Familia HC908 Flash de Motorola

Sumario de Registros

SCC1	READ:	LOOPS	ENSCI	TXINV	M	WAKE	ILTY	PEN	PTY
	WRITE:								
SCC2	READ:	SCTIE	TCIE	SCRIE	ILIE	TE	RE	RWU	SBK
	WRITE:								
SCC3	READ:	R8	T8	DMARE	DMATE	ORIE	NEIE	FEIE	PEIE
	WRITE:								
SCS1	READ:	SCTE	TC	SCRf	IDLE	OR	NF	FE	PE
	WRITE:								
SCS2	READ:						BKF	RPF	
	WRITE:								
SCDR	READ:	R7	R6	R5	R4	R3	R2	R1	R0
	WRITE:	T7	T6	T5	T4	T3	T2	T1	T0
SCBR	READ:		SCP1	SCP0			SCR2	SCR1	SCR0
	WRITE:								

Curso de Microcontroladores

Familia HC908 Flash de Motorola

```

*****
* ELECTROCOMPONENTES S.A. *
* Curso en la WEB de MCUs HC908 FLASH de Motorola *
* BUENOS AIRES, REPUBLICA ARGENTINA *
* *
* Title: SCI.ASM *
* ----- *
* "Transmisión de Datos Asincrónicas - Usando el módulo de SCI" *
* *
* Creado en : Noviembre 2001 *
* *
* Autor: ING. DANIEL DI LELLA D.F.A.E for Motorola Products *
* *
* Descripción: *
* ----- *
* El código aquí contenido usa el módulo de SCI para transmitir *
* Caracteres en forma serial usando interrupciones: *
* Cuando el "Transmission Data Register" (SCTE) está vacío, se *
* tendrá que atender una interrupción la cuál nos indicará que el *
* SCDR está disponible para escribir otro. La transmisión se realiza *
* A un Baud Rate de 9600. Mientras que el SCI está siendo solamente *
* Usado para la transmisión, el Data register no será usado *
* Bi-direccionalmente para recibir información *
* Sin embargo, la rutina de servicio , en el caso que un *
* "Receive Interrupt" ocurra, la misma está también incluida *
* Frecuencia de Bus = 4,9152 Mhz (xtal ext.= Fbus x 4 = 19,6608Mhz *
* *****

```

Curso de Microcontroladores

Familia HC908 Flash de Motorola

Parte II

ING. DANIEL DI LELLA DDFAE For Motorola Products

```

INCLUDE 'EQUATES.ASM'      ;Equates para todos los registros (ver archivo adjunto)
* User Variables
DEST    EQU    $70          ;Destino para los caracteres recibidos
* Bit Equates
FLAG    EQU    6
* -----
*  TABLE : DATOS A SER TRANSMITIDO
* -----

        ORG    $40          ; Origen de la tabla en $40

TABLE   FCB    'Demo del Modulo SCI en el HC908GP32 o similares'
        FCB    'transmision de caracteres por pin TX'
        DB     0            ; Byte usado para testear el fin de datos

* -----
*          MAIN PROGRAM
* -----

        ORG    $8000       ;Comienzo de la ROM (FLASH), comienzo programa en $8000 (HC908GP32)
MAIN:
        BSR    INIT        ; Subrutina para Inicializar los registros del SCI
        BSR    TRANSMIT    ; Subrutina para comenzar la transmisi3n
DONE    WAIT           ; Espera hasta la interrupci3n
        INCX           ; Incremento "X" al pr3ximo byte
        BRA    DONE       ; Salto a DONE
FINISH  BRA    FINISH     ; Finalizada la transmisi3n de todos los datos desde TABLE

```

Curso de Microcontroladores

Familia HC908 Flash de Motorola

Parte II

ING. DANIEL DI LELLA DDFAE For Motorola Products

```

* -----
*           SUBROUTINE INIT:
* -----

INIT:  SEI                ; Seteo I flag, para deshabilitar las interrupciones del CPU

        LDA    #$03      ; Cargo Acumulador para BAUD RATE = 9600 (Fbus = 4,9152Mhz)
        STA    SCBR      ; Descargo el valor en el SCI Baud Rate Reg.
        LDA    #$C0      ; Cargo Acumulador con valor : Formato de datos de 8-bit
        STA    SCC1      ; "Loop Mode" habilitado, SCI habilitado,
*                ; Descargo el valor al SCI Control Register 1
        LDA    #$00      ; Inicializo SCC3 para deshabilitar todas las interrupciones
        STA    SCC3      ; SCI Control Register 3

        LDX    #TABLE    ; Se usa "X" como un puntero para la TABLA (TABLE).

        LDA    SCS1      ; 1ER paso para limpiar TDRE & RDRF flags: Read SCS1
        STA    SCD       ; 2DO paso para limpiar TDRE & RDRF flags: Acceso a SCD register

        LDA    #$8C      ; Cargo Acc. con valor para:No Interrupciones excepto:
        STA    SCC2      ; Transmit interrupt, SCI Transmit & Receive enabled
*                ; Descargo valor en SCI control Register 2

        CLI                ; Limpio I flag, habilito interrupciones del CPU

        RTS

```

```

* -----
*      TRANSMIT SUBROUTINE
* -----
TRANSMIT:
      LDA      0,X          ; Carga Acumulador con carácter a enviar
      CBEQA   #$0,FINISH  ; Detecto si el último caracter(0) há sido transmitido
      STA     SCD          ; Escribo "Nueva" Dato al SCD register p/ iniciar transmisión,
*                                     ; también el 2do paso p/ limpiar TDRE flag, Acceso al SCD register
      RTS

* -----
*      SCI_RECEIVE: INTERRUPT SERVICE ROUTINE
* -----
RSCI_INT
      BRSET   FLAG,SCS1,TSCI_INT ; Si el Requerimiento es para una Transmisión
*                                     ; Luego ir a "Transmit Service Routine"
      LDA     SCS1          ; 1er paso p/ limpiar TDRE Flag, Read SCS1 ó
*                                     ; 1er paso p/ limpiar RDRF Flag, Read SCS1

      LDA     SCD          ; Cargar Acumulador con el Dato Recibido.
      STA     DEST,X       ; Descargar el dato recibido a DEST($70),X => C0

      RTI                  ; Return from Interrupt (vuelvo a ppal)

```

```

* -----
*   SCI_TRANSMIT: INTERRUPT SERVICE ROUTINE
* -----

TSCI_INT
    LDA    SCS1        ; ler paso p/ limpiar TDRE Flag, Read SCS1
    BSR    TRANSMIT    ; Salto a Subrutina de Transmisión p/ el próx. byte a transferir
    RTI                    ; Return from Interrupt

* -----
*   SCI VECTORS
* -----

    ORG    $FFE2        ; Inicializo SCI: "Receive" y "Transmit" vectors
    FDB    TSCI_INT     ; en dirección $FFE2
    FDB    RSCI_INT     ; en dirección $FFE4

* -----
*   Inicialización del "RESET" Vector
* -----

    ORG    $FFFE        ; Inicialización del RESET vector en $FFFE
    FDB    MAIN

    END                    ; Fin del programa (para el compilador no es necesaria esta última
*                          ; línea es solo a modo ilustrativo)

```

EQUATES.ASM (equates para el HC908GP32/GP20)

```
; 68HC908GP32 Equates (Equate invocadas desde el archivo "EQUATES.ASM" en la instrucción
;
; INCLUDE del compilador)
PTA EQU $0000 ; Ports and data direction
PORTA EQU $0000
PTB EQU $0001
PORTB EQU $0001
PTC EQU $0002
PORTC EQU $0002
PTD EQU $0003
PORTD EQU $0003
DDRA EQU $0004
DDRB EQU $0005
DDRC EQU $0006
DDRD EQU $0007
PTE EQU $0008
PORTE EQU $0008
DDRE EQU $000C

PTAPUE EQU $000D ; Port pull-up enables
PTCPUE EQU $000E
PTDPUE EQU $000F

SPCR EQU $0010 ; SPI (Synchronous communications)
SPSCR EQU $0011
SPDR EQU $0012
```

Curso de Microcontroladores

Familia HC908 Flash de Motorola

Parte II

ING. DANIEL DI LELLA DDFAE For Motorola Products

```

SCC1    EQU $0013    ; SPI (Asynchronous communications)
SCC2    EQU $0014
SCC3    EQU $0015
SCS1    EQU $0016
SCS2    EQU $0017
SCDR    EQU $0018
SCBR    EQU $0019

INTKBSCR EQU $001a    ; Keyboard interrupt control/status
INTKBIER EQU $001b

TBCR    EQU $001c    ; Time base module

INTSCR  EQU $001d    ; IRQ status/control

CONFIG2 EQU $001e    ; System configuration
CONFIG1 EQU $001f
T1SC    EQU $0020    ; Timer 1
T1CNTH  EQU $0021
T1CNTL  EQU $0022
T1MODH  EQU $0023
T1MODL  EQU $0024
T1SC0   EQU $0025
T1CH0H  EQU $0026
T1CH0L  EQU $0027
T1SC1   EQU $0028
T1CH1H  EQU $0029
T1CH1L  EQU $002a

```

Curso de Microcontroladores

Familia HC908 Flash de Motorola

```

T2SC    EQU $002b    ; Timer 2
T2CNTH  EQU $002c
T2CNTL  EQU $002d
T2MODH  EQU $002e
T2MODL  EQU $002f
T2SC0   EQU $0030
T2CH0H  EQU $0031
T2CH0L  EQU $0032
T2SC1   EQU $0033
T2CH1H  EQU $0034
T2CH1L  EQU $0035

PCTL    EQU $0036    ; Phase lock loop (for crystals)
PBWC    EQU $0037
PMSH    EQU $0038
PMSL    EQU $0039
PMRS    EQU $003A
PMDS    EQU $003B

ADSCR   EQU $003C    ; A to D converter
ADR     EQU $003D
ADCLK   EQU $003E

```

```

SBSR    EQU $fe00    ; System integration
SRSR    EQU $fe01
SUBAR   EQU $fe02
SBFCR   EQU $fe03

INT1    EQU $fe04    ; Interrupt status
INT2    EQU $fe05
INT3    EQU $fe06

FLTCR   EQU $fe07    ; Flash test/programming
FLCR    EQU $fe08

BRKH    EQU $fe09    ; Hardware breakpoint
BRKL    EQU $fe0a
BRKSCR  EQU $fe0b

LVISR   EQU $fe0c    ; Low voltage detect

FLBPR   EQU $ff7e    ; Flash boot protect

COPCTL  EQU $ffff    ; COP (Computer operating properly) control

```

```

; (C) opyright P&E Microcomputer Systems, 1998
; You may use this code freely as long as this copyright notice
; is included.

```