

**SERIAL  
PERIPHERAL  
INTERFACE  
(SPI)  
Módulo Serial  
Sincrónico**

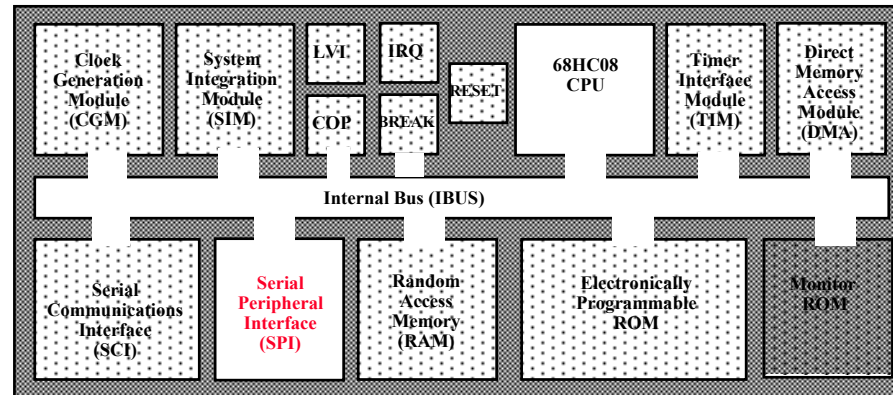
*Curso de Microcontroladores*

*Familia HC908 Flash de Motorola*

*Parte II*

ING. DANIEL DI LELLA DDFAE For Motorola Products

# SERIAL PERIPHERAL INTERFACE MODULE



- Las características del módulo SPI incluyen lo siguiente:
- **Operación Full-Duplex**
- **Modos Master y Slave**
- **Registros separados Transmit y Receive**
- **4 Frecuencias modo Master ( Máximo = Bus Frequency / 2 )**
- **Frecuencia Máxima modo Slave = Bus Frequency**
- **Serial Clock con Polaridad y Fase programables**
- **Flag de error “Bus Contention” ( contención del Bus )**
- **Flag “Overrun Error”**
- **2 habilitaciones de interrupción separadas :**
  - SPRF (SPI Receiver Full)**
  - SPTE (SPI Transmitter Empty)**
- **Modo programable “Wired-OR”**
- **Compatibilidad I2C (Inter-Integrated Circuit)**

# Registros I/O del SPI

Tres registros controlan y monitorean las operaciones del SPI :

- SPI Control Register (SPCR)
- SPI Status and Control Register (SPSCR)
- SPI Data Register (SPDR)

# Modos del SPI

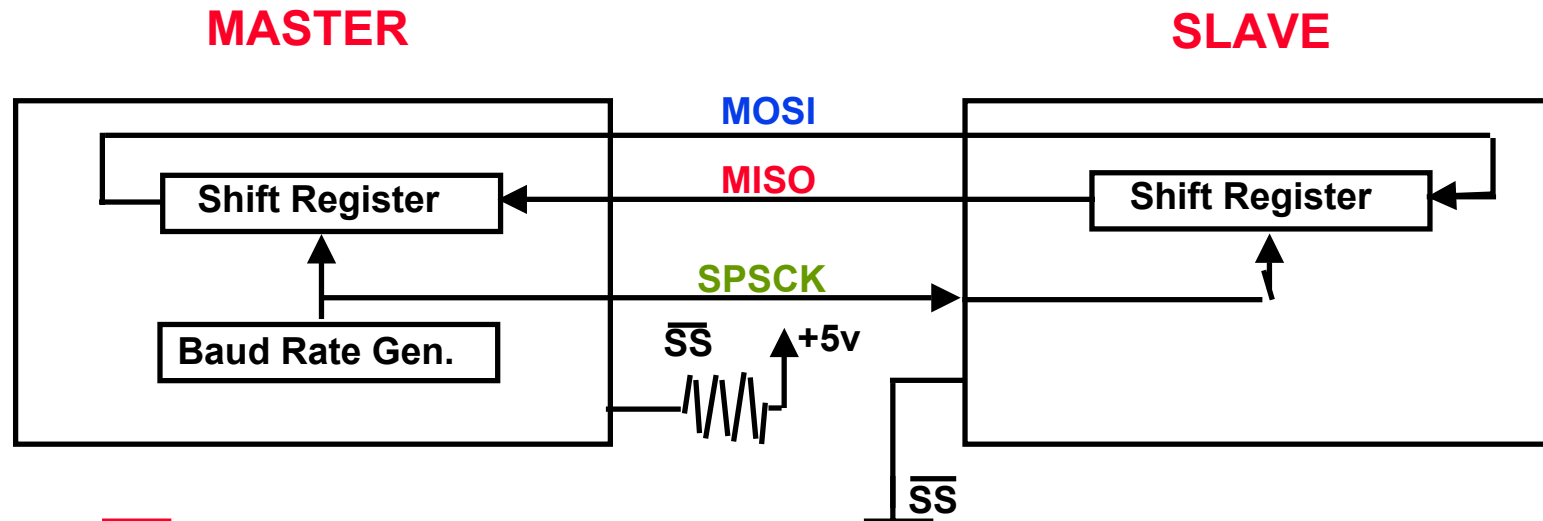
## Modo “Master” (Amo)

- Solamente un SPI “master” inicia la transmisión
- Los Datos son desplazados hacia afuera via la linea **Master Out Slave In (MOSI)**
- Los Datos son desplazados hacia adentro via la linea **Master In Slave Out (MISO)**
- La Transmisión **finaliza** después de **8 ciclos** del **serial clock (SPSCK)** , ya que se trata de una transmisión serial “sincronizada” con el “Clock” generado por el master y cada dato está sincronizado con dicho clock (8 pulsos de clock y no mas que ello).

## Modo “Slave” (Esclavo)

- Transferencia sincronizada al serial clock (**SPSCK**) desde el “Master”
- Los Datos son desplazados hacia adentro via la linea **Master Out Slave In (MOSI)**
- Los Datos son desplazados hacia afuera via la linea **Master In Slave Out (MISO)**

## Pin selección "Slave"



### Slave Select ( $\overline{SS}$ )

- **Modo Master**

- SS se mantiene alto ( "1" ) durante la transmisión (se fuerza a "1")
- Actua como una entrada de detección de error (si algo la fuerza a "0", hay un error !!)
- Puede ser una salida de propósitos generales.

- **Modo Slave**

- SS debe permanecer bajo ( "0" ) hasta que se completa la transmisión (forma de habilitar al dispositivo "slave")

0 = Habilita el "slave"

1 = Deshabilita el "slave"

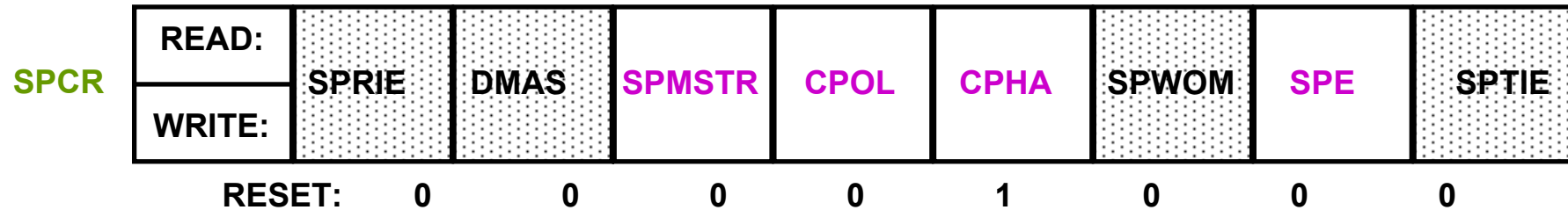
Curso de Microcontroladores

Familia HC908 Flash de Motorola

Parte II

ING. DANIEL DI LELLA DDFAE For Motorola Products

# SPI Control Register

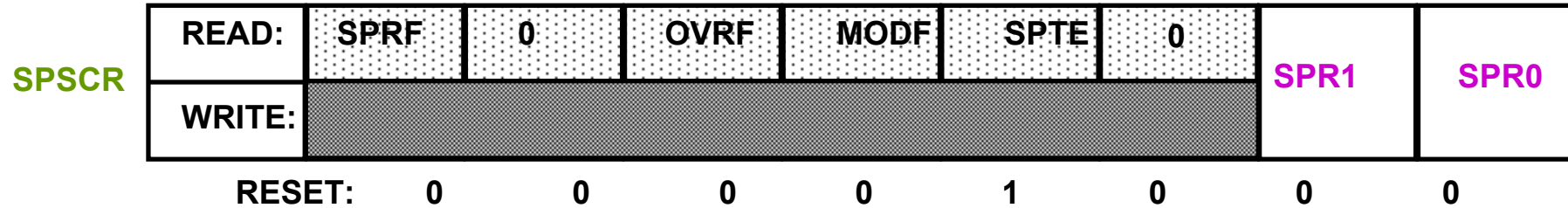


## SPI Control Register (SPCR)

- **SPI Master (SPMSTR)**
  - Selección del modo de operación master o slave
  - 1 = Modo Master
  - 0 = Modo Slave
- **SPI Enable (SPE)**
  - 1 = Habilita el SPI module
  - 0 = Deshabilita SPI module
- **SPI Master y Slave necesitan seteo de polaridad y fase de clock idénticas (1\*)**
- **Clock Polarity (CPOL)**
  - Determina el estado del clock cuando está “idle”
- **Clock Phase (CPHA)**
  - 1 = Comienza la captura de datos en el 2do. flanco del ciclo de clock
  - 0 = Comienza la captura de datos en el 1er. Flanco del ciclo de clock\*

**1\* NOTA:** Consultar en el manual de datos del MCU elegido y del dispositivo SPI a conectar, para determinar con exactitud el seteo de los bits CPOL y CPHA para una correcta comunicación entre ambos.

# Baud Rate del SPI

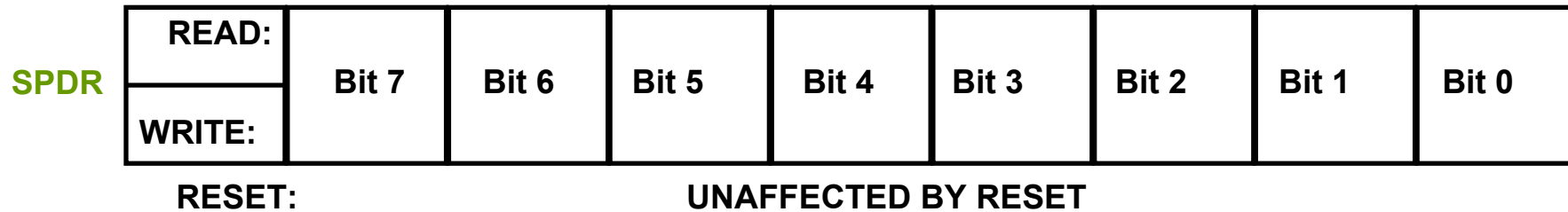


## SPI Status and Control Register (SPSCR)

- SPI rate select bits (**SPR1, SPR0**) (selección velocidad)
  - Setea la frecuencia del Master SPSCCK clock
  - No tiene efecto en los dispositivos "Slave"
  - **Baud Rate = CGMOUT / Baud Rate Divisor**

<b>SPR1:SPR0</b>	System Clock Divided By	Baud Rate (System Clock Freq. = 8 MHz)
00	2	<b>4 MHz</b>
01	8	1 MHz
10	32	250 KHz
11	128	62.5 KHz

# SPI Data Register

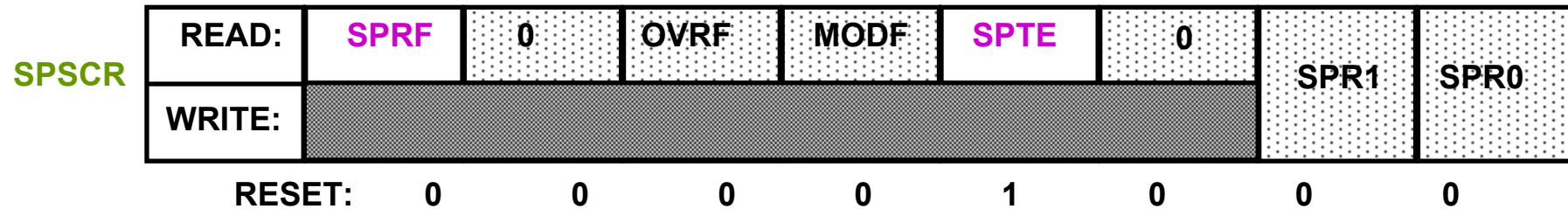


## SPI Data Register (SPDR)

- Buffer de **Lectura / Escritura** para los datos SPI
- Operación de escritura
  - Escribe datos al “transmit data register”
- Operación de lectura
  - Lee datos en el “receive data register”

Un SOLO buffer tanto para la lectura de datos (recepción) como para la escritura de datos (transmisión). Tener en cuenta que en una conexión SPI, los datos forman un “anillo” circular entre los buffers del dispositivo “master” y el dispositivo “slave”.

## Flags de estados del SPI

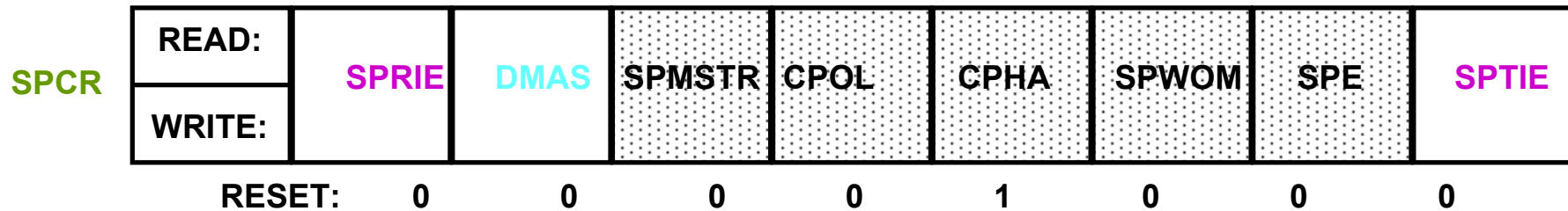


### SPI Status and Control Register (SPSCR)

- SPI Receiver Full (SPRF)
  - Seteado cuando un **byte** es desplazado desde el shift register al “receive data register”
  - **Limpiado por la lectura del SPSCR luego de la lectura del SPDR.**

1 = Receive data register full  
0 = Receive data register not full
- SPI Transmitter Empty (SPTE)
  - Seteado cuando un **byte** se transfiere desde el SPDR al shift register
  - **Limpiado por la lectura del SPDR register**
    - 1 = Transmit data register “vacío”
    - 0 = Transmit data register no “vacío”

# Interrupciones del SPI



## SPI Control Register (SPCR)

- SPI Receiver Interrupt Enable Bit (SPRIE)
  - Interrupción generada cuando se setea el flag SPRF
- SPI Transmit Interrupt Enable (SPTIE)
  - Interrupción generada cuando se setea el flag SPTE
  - 1 = Interrupción habilitada
  - 0 = Interrupción deshabilitada
- Direct Memory Access Select (DMAS)
  - No disponible en esta familia de HC08 !!!

# Inicialización

## Secuencia de inicialización del SPI

- 1) Inicializar la frecuencia de clock del SPI ( **SPR1 y SPR0 en SPSCR** )
- 2) Setear la configuración del clock ( **bits CPOL y CPHA en SPSCR** )
- 3) Seleccionar operación Master/Slave ( **SPMSTR en SPCR** )
- 4) Habilitar las interrupciones si es necesario ( **SPTIE, SPRIE en SPCR** )
- 5) Habilitar el sistema SPI ( **SPE en SPCR** )
  - **Deberá habilitarse el dispositivo “Master” antes del “Slaves” !!!**  
**(Para evitar falsas operaciones)**

# Transferencia -- Master a Slave

## Operación "Poleada" simple ( x polling )

- 1) Inicializar el SPI
- 2) Seleccionar el  $\overline{SS}$  en el dispositivo Slave ( hardware dependiente )
- 3) Escribir un byte en el SPDR
- 4) En espera de la activación del Flag SPI Transmitter Empty (SPTE)
- 5) Leer el SPDR
- 6) Liberar el  $\overline{SS}$  en dispositivo Slave ( hardware dependiente )

## Información Adicional Modo “Wired-Or”

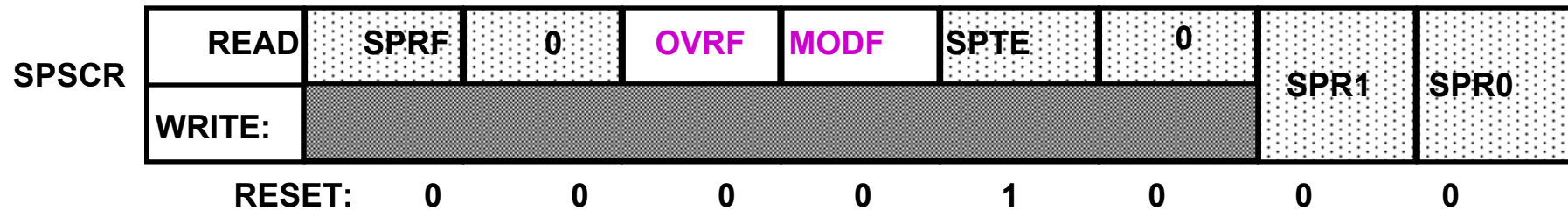


### SPI Control Register (SPCR)

- SPI Wired OR Mode (SPWOM)
  - Configura las salidas **MISO**, **MOSI**, y al **SPSCK** como “open-drain”
  - Permite sistemas de “Masters Múltiples”
  - Provee algo de protección contra el “CMOS latchup”.

## Información Adicional

### Flags de estados “Overflow y Mode Fault”



#### SPI Status and Control Register (SPSCR)

- Overflow flag (OVRF)
  - Se activa cuando el “Receive data register” no es leído antes de sobrescribirlo
  - El byte de datos entrante se pierde
    - El contenido del Data register no es afectado
  - Limpiado por la lectura del “data register”
- Mode Fault flag (MODF)
  - Modo Master solamente
  - Indica que otro master trata de acceder a ese dispositivo
  - Seteado cuando otro dispositivo pone el pin SS a “Low”
  - Limpiado por una escritura al SPSCR

## Sumario de Registros

SPCR	READ:	SPRIE	DMAS	SPMSTR	CPOL	CPHA	SPWOM	SPE	SPTIE
	WRITE:								

SPSCR	READ:	SPRF	0	OVRF	MODF	SPTE	0	SPR1	SPR0
	WRITE:								

SPDR	READ:	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	WRITE:								

```

*****
* ELECTROCOMPONENTES S.A. *
* CURSO EN LA WEB SOBRE MICROCONTROLADORES HC908 FLASH DE MOTOROLA *
* BUENOS AIRES, REPUBLICA ARGENTINA *
* *
* Título: SPI.ASM *
* ----- *
* "TRANSMISION DE DATOS SINCRONICA USANDO EL MODULO SPI" *
* *
* Fecha de creación: DICIEMBRE , 2001 *
* *
* Autor: ING. DANIEL DI LELLA - D.F.A.E for Motorola Products *
* Manager & Technical Consult *
* Descripción: *
* ----- *
* El código aquí descripto intenta explicar el funcionamiento del *
* Módulo de SPI para transmitir serialmente caracteres en código *
* ASCII. El prorama configura el SPI como un MASTER, manejado por *
* interrupciones. El periférico "Slave" es seleccionado por medio *
* de la línea PTB3 a un nivel BAJO. Entre cada transferencia de *
* 8 bits, la línea PTB3 se mantiene en nivel ALTO. *
* También el clock se mantiene BAJO y toma datos en el flanco de *
* Subida de este. El clock serial (SPCLK) se há seteado para no *
* Exceder 1 Mhz de baud rate.FBUS = 8MHZ (XTAL = 32MHZ) *
* El MCU utilizado en este ejemplo es un MC68HC908GP32, se puede *
* aplicar a cualquier otro MCU de la flia. HC908 con Módulo SPI *
* *****

```

*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 del HC908GP32
* -----
*  TABLE : DATOS A SER TRANSMITIDOS
* -----
          ORG      $40          ;Los DATOS comienzan en $40 (comienzo de la RAM)

DATA     FCB      "Como usar el módulo de SPI en los HC908 FLASH"
          FCB      "En forma sencilla y sin sufrir por ello."
          DB       0

* -----
*  MAIN PROGRAM (PROGRAMA PRINCIPAL)
* -----
          ORG      $8000       ;Comienzo de la memoria de programa en FLASH
MAIN:
          BSR      INIT        ; Subrutina para inicializar los registros del SPI
          BSR      TRANSMIT    ; Subrutina para comenzar con la transmisión
DONE     WAIT     ; Espero (Wait) hasta la interrupción
          INCX     ; Incremento el puntero "X" al próximo byte
          BRA     DONE        ; Siempre salto a DONE
FINISH   BRA     FINISH      ; Finalizó la transmisión de todos los DATOS

```

```

* -----
*           SUB-RUTINA INIT:
* -----

INIT:

    BSET    3,PTB        ; Setea el chip-select pin(PTB3) al estado "high"
    BSET    3,DDRB       ; Setea el chip-select pin(PTB3) como una "output"

    LDA     #$0C         ; Configuro PORT D como input/ouput según..
    STA     DDRD         ; MOSI, SPCK = output, MISO, SS* = Input

    LDA     #$01         ; Selección del serial clock baud rate = 1 MHz
    STA     SPSCR        ; lo almaceno en el SPSCR register

    LDA     #$A0         ; Configuro el SPI (SPCR): habilito interrupciones SPI,
    STA     SPCR         ; DMA NO habilitado, SPMSTR=1, CPOL=0, CPHA=0, SPE=0

    LDX     #DATA        ; Uso el registro "X" como un puntero al primer caracter

    LDA     SPSCR        ; 1er paso p/ limpiar SPRF Flag, Read SPSCR
    LDA     SPDR         ; 2do paso p/ limpiar SPRF Flag, Read SPDR

    BSET    1,SPCR       ; Habilito el módulo de SPI (SPE=1)
    CLI                    ; Limpio el I flag, habilito las interrupciones del CPU
    RTS

```

```

* -----
*      TRANSMIT SUBROUTINE (SUB-RUTINA DE TRANSMISION)
* -----
TRANSMIT:
    LDA    0,X          ; Cargo el Acumulador con el "NUEVO" carácter a enviar
    CBEQA  #$0,FINISH  ; Detecto si el último carácter(0) está siendo transmitido
    BCLR   3,PTB       ; Selecciono PTB3 = Low para seleccionar el periférico
    STA    SPDR        ; Guardo el Acumulador(dato) a ser transmitido
                    ;

    RTS

* -----
*      SPI_TRANSMIT: INTERRUPT SERVICE ROUTINE
* -----
TSPI_VEC

    BSET   3,PTB       ; De-selecciono el periférico (PTB3 = high)
    LDA    SPSCR       ; 1er paso p/ limpiar SPIF Flag, Read SPSCR
    LDA    SPDR        ; 2do paso p/ limpiar SPIF Flag, Read SPDR
    BSR    TRANSMIT    ; Llamo a la Subrutina de transmisión p/ transmitir el prox. Carac.
    RTI                ; Return from Interrupt

```

```

* -----
*   SPI VECTORS: Transmit Vector (Vector de Transmisión)
* -----
      ORG    $FFE8      ; Inicializo el "SPI vector" en $FFF8
      FDB    TSPI_VEC

* -----
*   Inicializo el "RESET" Vector
* -----
      ORG    $FFFE      ; Inicialización del RESET vector en $FFFE
      FDB    MAIN

      END              ; Fin del programa !!! (no es necesaria esta línea, es solo a
                       ; los efectos didacticos)

```