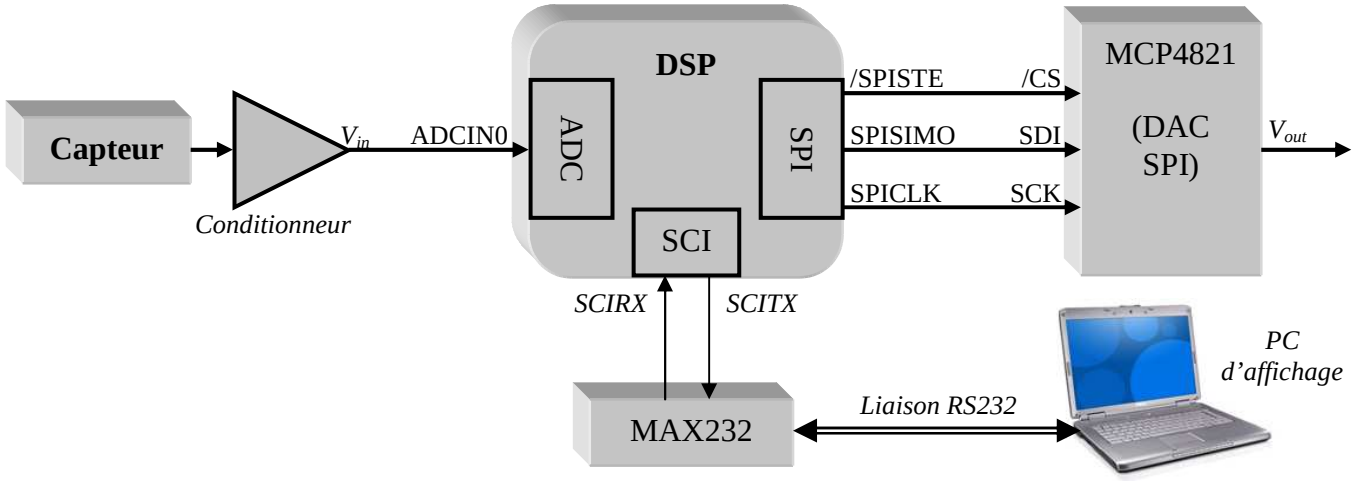
Médian MC43

Module de traitement temps réel de signaux analogiques

Le dispositif étudié réalise l'échantillonnage du signal analogique V_{in} délivré par un capteur et son conditionneur, puis le traitement numérique du signal échantillonné et enfin la restitution en analogique du signal traité à l'aide du convertisseur numérique analogique SPI (DAC MCP4821). Une IHM sur PC permet de communiquer avec le DSP par l'intermédiaire de la liaison série SCI (configuration du module et affichage de données).
Le calculateur est un DSP TMS320LF2407A cadencé à 40 MHz. Le principe de traitement s'organise de la manière suivante :

- le signal analogique V_{in} appliqué sur ADCIN0 est échantillonné à Fe = 100 kHz
- l'interruption liée à INT1 est déclenchée à chaque fin de séquence de conversion du module ADC et la fonction d'interruption associée réalise :
  - la lecture du résultat de conversion de V_{in} (stockage dans la variable Vin)
  - le calcul numérique de la tension V_{out} (fonction V_{out} = FILTRE(V_{in}) non étudiée ici)
  - la transmission de la valeur numérique V_{out} au DAC SPI MCP4821 (la fréquence SPI est de 10 MHz)
- les réceptions SCI sont traitées par interruption (INT5) et stockage en buffer circulaire
- la gestion des données émisses et reçues avec l'IHM est réalisée dans la fonction main()



I Conversion analogique numérique (6.5 points)

Question 1 (1 point) :
Le conditionneur est doté d'un filtre anti-repliement. Déterminer quelle doit être sa fréquence de coupure et expliquer quel serait le risque s'il n'y avait pas de filtre anti-repliement.
Question 2 (1 point) :
Le timer3 du DSP est utilisé pour permettre le déclenchement des conversions analogiques numériques lorsque son compteur atteint la valeur de la période T3PR. Déterminer le contenu du registre T3PR lors d'une configuration en mode comptage continu pour obtenir une fréquence d'échantillonnage Fe = 100 kHz.
Question 3 (1 point) :
Montrer que la durée d'une conversion est compatible avec la fréquence d'échantillonnage Fe.
Question 4 (1 point) :
Rappeler quel est l'intérêt d'utiliser le principe des interruptions matérielles pour traiter les résultats de conversion.
Question 5 (1 point) :
Indiquer (en assembleur) comment initialiser les vecteurs d'interruption pour associer l'interruption INT1 du DSP à la fonction : ***interrupt void interADC(void)***.
Question 6 (1.5 point) :
Ecrire en langage C la fonction ***interrupt void interADC(void)*** en y incluant les actions suivantes :

- réinitialisation des indicateurs d'interruption
- lecture du résultat de conversion (dans la variable globale *Vin*)
- instruction : Vout = FILTRE(Vin);

II Transmissions SPI (6.5 points)

Question 1 (1.5 point) :
Montrer que la durée de transmission de V_{out} au MCP4821 est compatible avec la fréquence d'échantillonnage Fe.
Question 2 (1 point) :
Déterminer les deux paramètres CLOCK POLARITY et CLOCK PHASE du contrôleur SPI pour assurer la compatibilité avec les timings du MCP4821.
Question 3 (2 point) :
Ecrire en langage C la fonction d'initialisation du contrôleur SPI : ***void initSPI(void)***.
Question 4 (2 point) :
Donner les instructions complétant la fonction d'interruption ***interrupt void interADC(void)*** pour réaliser le transfert de V_{out} vers le MCP4821 avec /GA=0 et /SHDN=1 (transfert et réinitialisation de l'indicateur de fin de transfert SPI).

III Transmissions SCI (7 points)

Question 1 (1 point) :
Donner la longueur maximale d'une ligne de transmission RS232 et la vitesse de transmission maximale pour cette longueur de ligne, autorisées par la norme RS232.
Question 2 (2 points) :
Le module SCI est configuré dans le mode suivant : 19200 bauds, données de 8 bits, pas de bit de parité, 1 bit de STOP, RXINT activée en priorité faible (SCIRX Priority = 1). Ecrire en langage C la fonction d'initialisation du module SCI ***void initSCI(void)***.
Question 3 (1 point) :
Déterminer la vitesse de transmission utile maximale (bande passante) de la liaison SCI en octets par seconde en arrondissant le débit binaire à 20kbauds.
Question 4 (1 point) :
Ecrire en langage C la fonction d'émission d'un octet ***void putchar(unsigned int a)***.
Question 5 (1 point) :
Décrire en quelques lignes le principe de réception par interruption et stockage en buffer circulaire des données reçues.
Question 6 (1 point) :
Indiquer comment initialiser les vecteurs d'interruption pour associer l'interruption INT5 du DSP via RXINT à la fonction gérant le remplissage du buffer circulaire : ***interrupt void interSCI(void)***.

Registre de configuration du DSP : SCSR1

15	14	13	12	11	10	9	8
Reserved	CLKSRC	LPM1	LPM0	CLK PS2	CLK PS1	CLK PS0	Reserved
R-0	RW-0	RW-0	RW-0	RW-1	RW-1	RW-1	R-0
7	6	5	4	3	2	1	0
ADC CLKEN	SCI CLKEN	SPI CLKEN	CAN CLKEN	EVB CLKEN	EVA CLKEN	Reserved	ILLADR
RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	R-0	RC-0

Note: R = Read access, W = Write access, C = Clear, -0 = value after reset

Entrées/sorties numériques multiplexées du DSP

PIN FUNCTION SELECTED		MUX CONTROL REGISTER (name.bit #)
(MCRx.n = 1) Primary Function	(MCRX.N = 0) I/O	
SCITXD	***IOPA0***	MCRA.0
SCIRXD	***IOPA1***	MCRA.1
XINT1	***IOPA2***	MCRA.2
CAP1/QEP1	***IOPA3***	MCRA.3
CAP2/QEP2	***IOPA4***	MCRA.4
CAP3	***IOPA5***	MCRA.5
PWM1	***IOPA6***	MCRA.6
PWM2	***IOPA7***	MCRA.7

PIN FUNCTION SELECTED		MUX CONTROL REGISTER (name.bit #)
(MCRx.n = 1) Primary Function	(MCRX.N = 0) I/O	
W/R#	IOPC0	MCRB.0
BIO	IOPC1	MCRB.1
SPISIMO	***IOPC2***	MCRB.2
SPISOMI	***IOPC3***	MCRB.3
SPICLK	***IOPC4***	MCRB.4
SPISTE	***IOPC5***	MCRB.5
CANTX	***IOPC6***	MCRB.6
CANRX	***IOPC7***	MCRB.7

Module SCI du DSP

Address	Register Mnemonic	Bit Number 7	6	5	4	3	2	1	0	Register Name
7050h	SCICCR	STOP BITS	EVEN/ ODD PARITY	PARITY ENABLE	LOOP-BACK ENA	ADDR/ IDLE MODE	SCI CHAR2	SCI CHAR1	SCI CHAR0	Commu-nication control
7051h	SCICTL1	Reserved	RX ERR INT ENA	SW RESET	Reserved	TXWAKE	SLEEP	TXENA	RXENA	SCI control register1
7052h	SCIHBAUD	BAUD15 (MSB)	BAUD14	BAUD13	BAUD12	BAUD11	BAUD10	BAUD9	BAUD8	Baud rate (MSbyte)
7053h	SCILBAUD	BAUD7	BAUD6	BAUD5	BAUD4	BAUD3	BAUD2	BAUD1	BAUD0 (LSB)	Baud rate (LSbyte)
7054h	SCICTL2	TXRDY	TX EMPTY	Reserved				RX/BK INT ENA	TX INT ENA	SCI control register 2
7055h	SCIRXST	RX ERROR	RXRDY	BRKDT	FE	OE	PE	RXWAKE	Reserved	Receiver status
7057h	SCIRXBUF	RXDT7	RXDT6	RXDT5	RXDT4	RXDT3	RXDT2	RXDT1	RXDT0	Receiver data buffer
7058h	—	Reserved								—
7059h	SCITXBUF	TXDT7	TXDT7	TXDT5	TXDT4	TXDT3	TXDT2	TXDT1	TXDT0	Transmitter data buffer
705Fh	SCIPRI	Reserved	SCITX PRIORITY	SCIRX PRIORITY	SCI SOFT	SCI FREE	Reserved			Priority control

Configuration du débit binaire (vitesse de transmission): Registres SCIHBAUD et SCILBAUD

Registre SCIHBAUD

15	14	13	12	11	10	9	8
BAUD15 (MSB)	BAUD14	BAUD13	BAUD12	BAUD11	BAUD10	BAUD9	BAUD8
RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

Note: R = Read access, W = Write access, S = Set only, -0 = value after reset

Registre SCILBAUD

7	6	5	4	3	2	1	0
BAUD7	BAUD6	BAUD5	BAUD4	BAUD3	BAUD2	BAUD1	BAUD0 (LSB)
RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

Note: R = Read access, W = Write access, S = Set only, -0 = value after reset

Bits 15–0 **BAUD15–BAUD0**. SCI 16-bit baud selection.

Registers SCIHBAUD (MSbyte) and SCILBAUD (LSbyte) are concatenated to form a 16-bit baud value, BRR.

The internally-generated serial clock is determined by the CLKOUT signal and the two baud-select registers. The SCI uses the 16-bit value of these registers to select one of 64K serial clock rates for the communication modes.

The SCI baud rate is calculated using the following equation:

$$\text{SCI Asynchronous Baud} = \frac{\text{CLKOUT}}{(\text{BRR} + 1) \times 8}$$

Alternatively,

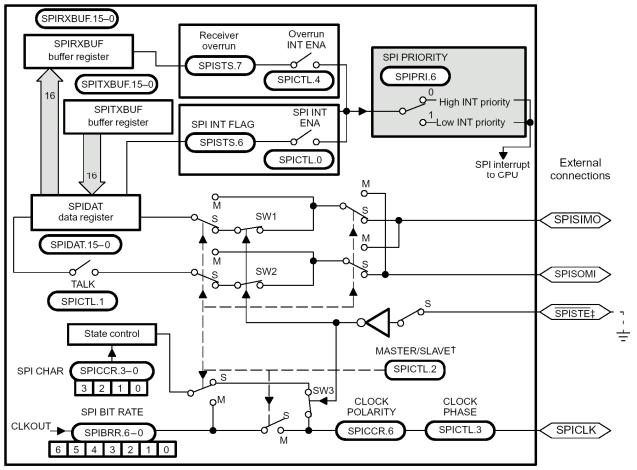
$$\text{BRR} = \frac{\text{CLKOUT}}{\text{SCI Asynchronous Baud} \times 8} - 1$$

Note that the above formulas are applicable only when $1 \le \text{BRR} \le 65535$. If BRR = 0, then

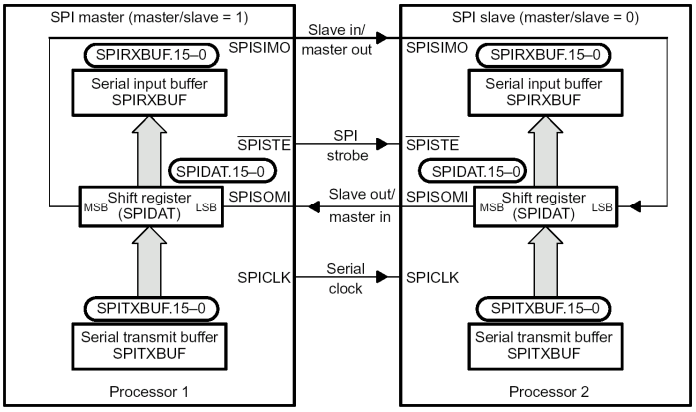
$$\text{SCI Asynchronous Baud} = \frac{\text{CLKOUT}}{16}$$

Where: BRR = the 16-bit value (in decimal) in the baud-select registers.

Module SPI du DSP



† The diagram is shown in slave mode.
‡ The $\overline{\text{SPISTE}}$ pin is driven low externally, meaning the data can be transmitted or received in this mode. Note that switches SW1, SW2, and SW3 are closed in this configuration. The "switches" are assumed to close when their "control signal" is high.



Registre de Contrôle de Configuration du Module SPI : SPICCR

7	6	5–4	3	2	1	0
SPI SW RESET	CLOCK POLARITY	Reserved	SPI CHAR3	SPI CHAR2	SPI CHAR1	SPI CHAR0
RW-0	RW-0	R-0	RW-0	RW-0	RW-0	RW-0

Note: R = Read access, W = Write access, -0 = value after reset

Bit 7 **SPI SW RESET**. SPI Software Reset. When changing configuration, you should clear this bit before the changes and set this bit before resuming operation. (See Section 9.4.5.1 on page 9-15.)

0 Initializes the SPI operating flags to the reset condition.

Specifically, the RECEIVER OVERRUN Flag bit (SPISTS.7), the SPI INT FLAG bit (SPISTS.6), and the TXBUF FULL Flag bit (SPISTS.5) are cleared. The SPI configuration remains unchanged. If the module is operating as a master, the SPICLK signal output returns to its inactive level.

1 SPI is ready to transmit or receive the next character.

When the SPI SW RESET bit is a 0, a character written to the transmitter will not be shifted out when this bit is set. A new character must be written to the serial data register.

Bit 6 **CLOCK POLARITY**. Shift Clock Polarity. This bit controls the polarity of the SPICLK signal. CLOCK POLARITY and CLOCK PHASE (SPICTL.3) control four clocking schemes on the SPICLK pin. See Section 9.4.4, *SPI Clocking Schemes*, on page 9-12.

0 Data is output on rising edge and input on falling edge. When no SPI data is sent, SPICLK is at low level.

The data input and output edges depend on the value of the CLOCK PHASE bit (SPICTL.3) as follows:

- ❏ CLOCK PHASE = 0: Data is output on the rising edge of the SPICLK signal; input data is latched on the falling edge of the SPICLK signal.
- ❏ CLOCK PHASE = 1: Data is output one half-cycle before the first rising edge of the SPICLK signal and on subsequent falling edges of the SPICLK signal; input data is latched on the rising edge of the SPICLK signal.

1 Data is output on falling edge and input on rising edge. When no SPI data is sent, SPICLK is at high level.

The data input and output edges depend on the value of the CLOCK PHASE bit (SPICTL.3) as follows:

- ❏ CLOCK PHASE = 0: Data is output on the falling edge of the SPICLK signal; input data is latched on the rising edge of the SPICLK signal.
- ❏ CLOCK PHASE = 1: Data is output one half-cycle before the first falling edge of the SPICLK signal and on subsequent rising edges of the SPICLK signal; input data is latched on the falling edge of the SPICLK signal.

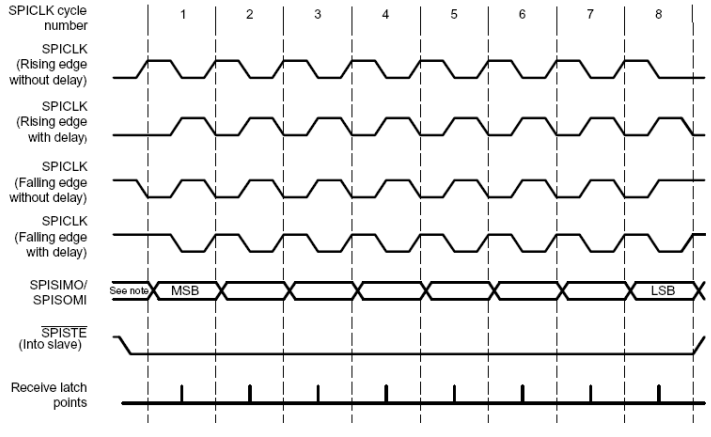
Bits 5–4 **Reserved**. Reads return zero; writes have no effect.

Bits 3–0 **SPI CHAR3–SPI CHAR0**. Character Length Control Bits 3–0. These four bits determine the number of bits to be shifted in or out as a single character during one shift sequence.

SPI CHAR3	SPI CHAR2	SPI CHAR1	SPI CHAR0	Character Length
0	0	0	0	1
0	0	0	1	2
0	0	1	0	3
0	0	1	1	4
0	1	0	0	5
0	1	0	1	6
0	1	1	0	7
0	1	1	1	8
1	0	0	0	9
1	0	0	1	10
1	0	1	0	11
1	0	1	1	12
1	1	0	0	13
1	1	0	1	14
1	1	1	0	15
1	1	1	1	16

Modes de configuration de l'horloge (CLOCK POLARITY et CLOCK PHASE)

SPICLK Scheme	CLOCK POLARITY (SPICCR.6)	CLOCK PHASE (SPICTL.3)
Rising edge without delay	0	0
Rising edge with delay	0	1
Falling edge without delay	1	0
Falling edge with delay	1	1



Note: Previous data bit

Registre de Contrôle du fonctionnement du module SPI : SPICTL

7–5	4	3	2	1	0
Reserved	OVERRUN INT ENA	CLOCK PHASE	MASTER/ SLAVE	TALK	SPI INT ENA
R-0	RW-0	RW-0	RW-0	RW-0	RW-0

Note: R = Read access, W = Write access, -0 = value after reset

Bits 7–5 **Reserved**. Reads return zero; writes have no effect.

Bit 4 **OVERRUN INT ENA**. Overrun Interrupt Enable. Setting this bit causes an interrupt to be generated when the RECEIVER OVERRUN Flag bit (SPISTS.7) is set by hardware. Interrupts generated by the RECEIVER OVERRUN Flag bit and the SPI INT FLAG bit (SPISTS.6) share the same interrupt vector.

0 Disable RECEIVER OVERRUN Flag bit (SPISTS.7) interrupts

1 Enable RECEIVER OVERRUN Flag bit (SPISTS.7) interrupts

Bit 3 **CLOCK PHASE**. SPI Clock Phase Select. This bit controls the phase of the SPICLK signal.

0 Normal SPI clocking scheme, depending on the CLOCK POLARITY bit (SPICCR.6)

1 SPICLK signal delayed by one half-cycle; polarity determined by the CLOCK POLARITY bit

CLOCK PHASE and CLOCK POLARITY (SPICCR.6) make four different clocking schemes possible (see Figure 9–3). When operating with CLOCK PHASE high, the SPI (master or slave) makes the first bit of data available after SPIDAT is written and before the first edge of the SPICLK signal, regardless of which SPI mode is being used.

Bit 2 **MASTER/SLAVE**. SPI Network Mode Control. This bit determines whether the SPI is a network master or slave. During reset initialization, the SPI is automatically configured as a network slave.

0 SPI configured as a slave.

1 SPI configured as a master.

Bit 1 **TALK**. Master/Slave Transmit Enable. The TALK bit can disable data transmission (master or slave) by placing the serial data output in the high-impedance state. If this bit is disabled during a transmission, the transmit shift register continues to operate until the previous character is shifted out. When the TALK bit is disabled, the SPI is still able to receive characters and update the status flags. TALK is cleared (disabled) by a system reset.

0 Disables transmission:

- ❑ Slave mode operation: If not previously configured as a general-purpose I/O pin, the SPISOMI pin will be put in the high-impedance state.
- ❑ Master mode operation: If not previously configured as a general-purpose I/O pin, the SPISIMO pin will be put in the high-impedance state.

1 Enables transmission

For the 4-pin option, ensure to enable the receiver's $\overline{\text{SPISTE}}$ input pin.

Bit 0 **SPI INT ENA**. SPI Interrupt Enable. This bit controls the SPI's ability to generate a transmit/receive interrupt. The SPI INT FLAG bit (SPISTS.6) is unaffected by this bit.

0 Disables interrupt

1 Enables interrupt

Registre de Configuration du débit binaire : SPIBRR

7	6	5	4	3	2	1	0
Reserved	SPI BIT RATE 6	SPI BIT RATE 5	SPI BIT RATE 4	SPI BIT RATE 3	SPI BIT RATE 2	SPI BIT RATE 1	SPI BIT RATE 0
R-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

Note: R = Read access, W = Write access, -0 = value after reset

Bit 7 **Reserved**. Reads return zero; writes have no effect.

Bits 6–0 **SPI BIT RATE 6–SPI BIT RATE 0**. SPI Bit Rate (Baud) Control. These bits determine the bit transfer rate if the SPI is the network master. There are 125 data-transfer rates (each a function of the CPU clock, CLKOUT) that can be selected. One data bit is shifted per SPICLK cycle. (SPICLK is the baud rate clock output on the SPICLK pin.)

If the SPI is a network slave, the module receives a clock on the SPICLK pin from the network master; therefore, these bits have no effect on the SPICLK signal. The frequency of the input clock from the master should not exceed the slave SPI's SPICLK signal divided by 4.

In master mode, the SPI clock is generated by the SPI and is output on the SPICLK pin. The SPI baud rates are determined by the following formula:

SPI Baud-Rate Calculations

- ❑ For SPIBRR = 3 to 127:

$$\text{SPI Baud Rate} = \frac{\text{CLKOUT}}{(\text{SPIBRR} + 1)}$$

- ❑ For SPIBRR = 0, 1, or 2:

$$\text{SPI Baud Rate} = \frac{\text{CLKOUT}}{4}$$

where: CLKOUT = CPU clock frequency of the device
SPIBRR = Contents of the SPIBRR in the master SPI device

Registre d'état du module SPI : SPISTS

7	6	5	4–0
RECEIVER OVERRUN FLAG†‡	SPI INT FLAG†‡	TX BUF FULL FLAG‡	Reserved
RC-0	RC-0	RC-0	R-0

Note: R = Read access, C = Clear, -0 = value after reset
† The RECEIVER OVERRUN FLAG bit and the SPI INT FLAG bit share the same interrupt vector.
‡ Writing a 0 to bits 5, 6, and 7 has no effect.

Bit 7 **RECEIVER OVERRUN FLAG**. SPI Receiver Overrun Flag. This bit is a read/clear-only flag. The SPI hardware sets this bit when a receive or transmit operation completes before the previous character has been read from the buffer. The bit indicates that the last received character has been overwritten and therefore lost (when the SPIRXBUF was overwritten by the SPI module before the previous character was read by the user application). The SPI requests one interrupt sequence each time this bit is set if the OVERRUN INT ENA bit (SPICTL.4) is set high. The bit is cleared in one of three ways:

- ❑ Writing a 1 to this bit
- ❑ Writing a 0 to SPI SW RESET (SPICCR.7)
- ❑ Resetting the system

If the OVERRUN INT ENA bit (SPICTL.4) is set, the SPI requests only one interrupt upon the first occurrence of setting the RECEIVER OVERRUN Flag bit. Subsequent overruns will not request additional interrupts if this flag bit is already set. This means that in order to allow *new* overrun interrupt requests the user must clear this flag bit by writing a 1 to SPISTS.7 each time an overrun condition occurs. In other words, if the RECEIVER OVERRUN Flag bit is left set (not cleared) by the interrupt service routine, another overrun interrupt will not be immediately re-entered when the interrupt service routine is exited.

However, the RECEIVER OVERRUN Flag bit should be cleared during the interrupt service routine because the RECEIVER OVERRUN Flag bit and SPI INT FLAG bit (SPISTS.6) share the same interrupt vector. This will alleviate any possible doubt as to the source of the interrupt when the next byte is received.

Bit 6 **SPI INT FLAG**. SPI Interrupt Flag. SPI INT FLAG is a read-only flag. The SPI hardware sets this bit to indicate that it has completed sending or receiving the last bit and is ready to be serviced. The received character is placed in the receiver buffer at the same time this bit is set. This flag causes an interrupt to be requested if the SPI INT ENA bit (SPICTL.0) is set. This bit is cleared in one of three ways:

- ❑ Reading SPIRXBUF
- ❑ Writing a 0 to SPI SW RESET (SPICCR.7)
- ❑ Resetting the system

Bit 5 **TX BUF FULL FLAG**. SPI Transmit Buffer Full Flag. This read-only bit gets set to 1 when a character is written to the SPI Transmit buffer SPITXBUF. It is cleared when the character is automatically loaded into SPIDAT when the shifting out of a previous character is complete. It is cleared at reset.

Bits 4–0 **Reserved**. Reads return zero; writes have no effect.

ADC Control Register 2 (ADCTRL2)

15	14	13	12	11	10	9	8
EVB SOC SEQ	RST SEQ1/ STRT CAL	SOC SEQ1	SEQ1 BSY	INT ENA SEQ1 (Mode 1)	INT ENA SEQ1 (Mode 0)	INT FLAG SEQ1	EVA SOC SEQ1
RW-0	RS-0	RW-0	R-0	RW-0	RW-0	RC-0	RW-0
7	6	5	4	3	2	1	0
EXT SOC SEQ1	RST SEQ2	SOC SEQ2	SEQ2 BSY	INT ENA SEQ2 (Mode 1)	INT ENA SEQ2 (Mode 0)	INT FLAG SEQ2	EVB SOC SEQ2
RW-0	RS-0	RW-0	R-0	RW-0	RW-0	RC-0	RW-0

Note: R = Read access, W = Write access, S = Set only, C = Clear, -0 = value after reset

Bit 15 **EVB SOC SEQ**. EVB SOC enable for cascaded sequencer
(Note: This bit is active only in cascaded mode.)

0 No action

1 Setting this bit allows the cascaded sequencer to be started by an Event Manager B signal. The Event Manager can be programmed to start a conversion on various events. See chapter 6, Event Manager (EV), for details.

Bit 14 **RST SEQ1 / STRT CAL**. Reset Sequencer1/Start Calibration

Case: Calibration Disabled (Bit 3 of ADCTRL1) = 0

Writing a 1 to this bit will reset the sequencer immediately to an initial "pretriggered" state, i.e., waiting for a trigger at CONV00. A currently active conversion sequence will be aborted.

0 No action

1 Immediately reset sequencer to state CONV00

Case: Calibration Enabled (Bit 3 of ADCTRL1) = 1

Writing a 1 to this bit will begin the converter calibration process.

0 No action

1 Immediately start calibration process

Bit 13 **SOC SEQ1**. Start-of-conversion (SOC) trigger for Sequencer 1 (SEQ1). This bit can be set by the following triggers:

- ❏ S/W – Software writing a 1 to this bit
- ❏ EVA – Event Manager A
- ❏ EVB – Event Manager B (only in cascaded mode)
- ❏ EXT – External pin (i.e., the ADCSOC pin)

When a trigger occurs, there are three possibilities:

Case 1: SEQ1 idle and SOC bit clear
SEQ1 starts immediately (under arbiter control). This bit is set and cleared, allowing for any "pending" trigger requests.

Case 2: SEQ1 busy and SOC bit clear
Bit is set signifying a trigger request is pending. When SEQ1 finally starts after completing current conversion, this bit will be cleared.

Case 3: SEQ1 busy and SOC bit set
Any trigger occurring in this case will be ignored (lost).

0 Clears a pending SOC trigger.
Note: If the sequencer has already started, this bit will automatically be cleared, and hence, writing a zero will have no effect; i.e., an already started sequencer cannot be stopped by clearing this bit.

1 Software trigger – Start SEQ1 from currently stopped position (i.e., Idle mode)

Note:

The RST SEQ1 (ADCTRL2.14) and the SOC SEQ1 (ADCTRL2.13) bits should not be set in the same instruction. This will reset the sequencer, but will not start the sequence. The correct sequence of operation is to set the RST SEQ1 bit first, and the SOC SEQ1 bit in the following instruction. This ensures that the sequencer is reset and a new sequence started. This sequence applies to the RST SEQ2 (ADCTRL2.6) and SOC SEQ2 (ADCTRL2.5) bits also.

Bit 12 **SEQ1 BSY**. SEQ1 Busy

This bit is set to a 1 while the ADC autoconversion sequence is in progress. It is cleared when the conversion sequence is complete.

0 Sequencer is Idle (i.e., waiting for trigger)

1 Conversion sequence is in progress

After a start-of-sequence (SOC) is initiated, four NOPs need to be executed before polling the SEQ1 or SEQ2 BSY bit.

Example code:

```
ADC_LOOP1:
       LDP    #ADCTRL1>>7

       SPLK   #0100000000000000b,ADCTRL2   ;Reset for SEQ1

       SPLK   #0010000000000000b,ADCTRL2   ;SOC for SEQ1

       NOP                                 ;Wait for Busy
                                           ;bit to set.
       NOP
       NOP
       NOP

CHK_EOS1:
       BIT    ADCTRL2, 3          ; Wait for SEQ1 Busy bit
                                  ; to clear

       BCND   CHK_EOS1, TC        ; If TC=1, keep looping.
```

A better approach would be to check the INT FLAG SEQn bit for end-of-sequence. This does not require NOPs, as the bit should already be cleared prior to starting a sequenced conversion. To reiterate, the NOPs are required only when polling the SEQn BUSY bit; interrupt-driven conversions do not have this requirement.

Bits 11–10 **INT ENA SEQ1**. Interrupt-mode-enable control for SEQ1

Bit 11	Bit 10	Operation Description
0	0	Interrupt is Disabled
0	1	Interrupt **Mode 1** Interrupt requested immediately when INT FLAG SEQ1 flag is set
1	0	Interrupt **Mode 2** Interrupt requested only if INT FLAG SEQ1 flag is already set. If clear†, INT FLAG SEQ1 flag is set and INT request is suppressed. (This mode allows Interrupt requests to be generated for every other EOS.)
1	1	Reserved

† This means that the last completed sequence is the first of the two sequences needed to assert an interrupt.

Bit 9 **INT FLAG SEQ1.** ADC interrupt flag bit for SEQ1

This bit indicates whether an interrupt event has occurred or not. This bit must be cleared by the user writing a 1 to it.

0 No interrupt event

1 An interrupt event has occurred.

Bit 8 **EVA SOC SEQ1.** Event Manager A SOC mask bit for SEQ1

0 SEQ1 cannot be started by EVA trigger.

1 Allows SEQ1/SEQ to be started by Event Manager A trigger. The Event Manager can be programmed to start a conversion on various events. See chapter 6, *Event Manager (EV)*, for details.

Bit 7 **EXT SOC SEQ1.** External signal start-of-conversion bit for SEQ1

0 No action

1 Setting this bit enables an ADC autoconversion sequence to be started by a signal from the ADCSOC device pin.

Bit 6 **RST SEQ2.** Reset SEQ2

0 No action

1 Immediately resets SEQ2 to an initial "pretriggered" state, i.e., waiting for a trigger at CONV08. A currently active conversion sequence will be aborted.

Bit 5 **SOC SEQ2.** Start-of-conversion trigger for Sequencer 2 (SEQ2)
(Only applicable in dual-sequencer mode; ignored in cascaded mode.)

This bit can be set by the following triggers:

- ❏ S/W – Software writing of 1 to this bit
- ❏ EVB – Event Manager B

When a trigger occurs, there are three possibilities:

Case 1: SEQ2 idle and SOC bit clear
SEQ2 starts immediately (under arbiter control) and the bit is cleared, allowing for any pending trigger requests.

Case 2: SEQ2 busy and SOC bit clear
Bit is set signifying a trigger request is pending. When SEQ2 finally starts after completing current conversion, this bit will be cleared.

Case 3: SEQ2 busy and SOC bit set
Any trigger occurring in this case will be ignored (lost).

0 Clears a Pending SOC trigger.
Note: If the sequencer has already started, this bit will automatically be cleared, and hence, writing a zero will have no effect; i.e., an already started sequencer cannot be stopped by clearing this bit.

1 Software trigger – Start SEQ2 from currently stopped position (i.e., Idle mode)

Bit 4 **SEQ2 BSY.** SEQ2 Busy

This bit is set to a 1 while the ADC autoconversion sequence is in progress. It is cleared when the conversion sequence is complete.

0 Sequencer is idle (i.e., waiting for trigger).

1 Conversion sequence is in progress.

Bits 3–2 **INT ENA SEQ2.** Interrupt-mode-enable control for SEQ2

Bit 3	Bit 2	Operation Description
0	0	Interrupt is Disabled
0	1	Interrupt **Mode 1** Interrupt requested immediate on INT FLAG SEQ2 flag set
1	0	Interrupt **Mode 2** Interrupt requested only if INT FLAG SEQ2 flag is already set. If clear†, INT FLAG SEQ2 flag is set and INT request is suppressed. (This mode allows Interrupt requests to be generated for every other EOS)
1	1	Reserved

† This means that the last completed sequence is the first of the two sequences needed to assert an interrupt.

Bit 1 **INT FLAG SEQ2.** ADC interrupt flag bit for SEQ2

This bit indicates whether an interrupt event has occurred or not. This bit must be cleared by the user writing a 1 to it.

0 No interrupt event.

1 An interrupt event has occurred.

Bit 0 **EVB SOC SEQ2.** Event Manager B SOC mask bit for SEQ2

0 SEQ2 cannot be started by EVB trigger.

1 Allows SEQ2 to be started by Event Manager B trigger. The Event Manager can be programmed to start a conversion on various events. See chapter 6, *Event Manager (EV)*, for details.

Timers du DSP

Registre de contrôle général des timers GPTCONA

15	14	13	12–11	10–9	8–7
Reserved	T2STAT	T1STAT	Reserved	T2TOADC	T1TOADC
RW-0	R-1	R-1	RW-0	RW-0	RW-0

6	5–4	3–2	1–0
TCOMPOE	Reserved	T2PIN	T1PIN
RW-0	RW-0	RW-0	RW-0

Note: R = Read access, W = Write access, -n = value after reset

Bit 15 **Reserved**. Reads return zero; writes have no effect.

Bit 14 **T2STAT**. GP timer 2 Status. Read only.
- 0 Counting downward
- 1 Counting upward

Bit 13 **T1STAT**. GP timer 1 Status. Read only.
- 0 Counting downward
- 1 Counting upward

Bits 12–11 **Reserved**. Reads return zero; writes have no effect.

Bits 10–9 **T2TOADC**. Start ADC with timer 2 event.
- 00 No event starts ADC
- 01 Setting of underflow interrupt flag starts ADC
- 10 Setting of period interrupt flag starts ADC
- 11 Setting of compare interrupt flag starts ADC

Bits 8–7 **T1TOADC**. Start ADC with timer 1 event.
- 00 No event starts ADC
- 01 Setting of underflow interrupt flag starts ADC
- 10 Setting of period interrupt flag starts ADC
- 11 Setting of compare interrupt flag starts ADC

Bit 6 **TCOMPOE**. Compare output enable. If $\overline{\text{PDPINTx}}$ is active this bit is set to zero.
- 0 Disable all GP timer compare outputs (all compare outputs are put in the high-impedance state)
- 1 Enable all GP timer compare outputs

Bits 5–4 **Reserved**. Reads return zero; writes have no effect.

Bits 3–2 **T2PIN**. Polarity of GP timer 2 compare output.
- 00 Forced low
- 01 Active low
- 10 Active high
- 11 Forced high

Bits 1–0 **T1PIN**. Polarity of GP timer 1 compare output.
- 00 Forced low
- 01 Active low
- 10 Active high
- 11 Forced high

Registre de contrôle individuel des timers : TxCON (x=1, 2, 3 ou 4)

15	14	13	12	11	10	9	8
Free	Soft	Reserved	TMODE1	TMODE0	TPS2	TPS1	TPS0
RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

7	6	5	4	3	2	1	0
T2SWT1/ T4SWT3†	TENABLE	TCLKS1	TCLKS0	TCLD1	TCLD0	TECMPR	SELT1PR/ SELT3PR†
RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0	RW-0

Note: R = Read access, W = Write access, -0 = value after reset
† Reserved in T1CON andT3CON

Bits 15–14 **Free, Soft.** Emulation control bits.
- 00 Stop immediately on emulation suspend
- 01 Stop after current timer period is complete on emulation suspend
- 10 Operation is not affected by emulation suspend
- 11 Operation is not affected by emulation suspend

Bit 13 **Reserved**. Reads return zero, writes have no effect.

Bits 12–11 **TMODE1–TMODE0.** Count Mode Selection.
- 00 Stop/Hold
- 01 Continuous-Up/-Down Count Mode
- 10 Continuous-Up Count Mode
- 11 Directional-Up/-Down Count Mode

Bits 10–8 **TPS2–TPS0.** Input Clock Prescaler.

000	x/1	100	x/16
001	x/2	101	x/32
010	x/4	110	x/64
011	x/8	111	x/128

x = device (CPU) clock frequency

Bit 7 **T2SWT1.** In the case of EVA, this bit is T2SWT1. (GP timer 2 start with GP timer 1.) Start GP timer 2 with GP timer 1's timer enable bit. This bit is reserved in T1CON.
T4SWT3. In the case of EVB, this bit is T4SWT3. (GP timer 4 start with GP timer 3.) Start GP timer 4 with GP timer 3's timer enable bit. This bit is reserved in T3CON.
- 0 Use own TENABLE bit
- 1 Use TENABLE bit of T1CON (in case of EVA) or T3CON (in case of EVB) to enable and disable operation ignoring own TENABLE bit

Bit 6 **TENABLE**. Timer enable.
- 0 Disable timer operation (the timer is put in hold and the prescaler counter is reset)
- 1 Enable timer operations

Bits 5–4 **TCLKS1, TCLKS0.** Clock Source Select.

5	4	Source
0	0	Internal
0	1	External
1	0	Reserved
1	1	QEP Circuit† (in case of Timer 2/Timer 4)
		Reserved (in case of Timer 1/Timer 3)

† This option is valid only if SELT1PR = 0

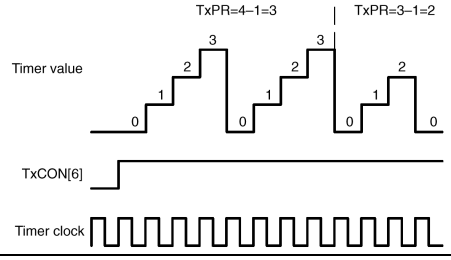
Bits 3–2 **TCLD1, TCLD0**. Timer Compare Register Reload Condition.
- 00 When counter is 0
- 01 When counter value is 0 or equals period register value
- 10 Immediately
- 11 Reserved

Bit 1 **TECMPR**. Timer compare enable.
- 0 Disable timer compare operation
- 1 Enable timer compare operation

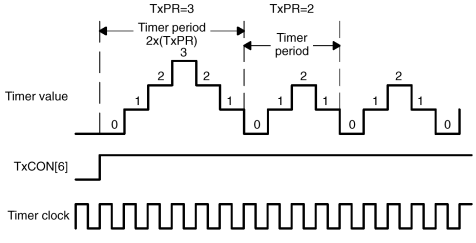
Bit 0 **SELT1PR**. In the case of EVA, this bit is SELT1PR (Period register select). When set to1 in T2CON, the period register of Timer 1 is chosen for Timer 2 also, ignoring the period register of Timer 2. This bit is a reserved bit in T1CON.
SELT3PR. In the case of EVB, this bit is SELT3PR (Period register select). When set to1 in T4CON, the period register of Timer 3 is chosen for Timer 4 also, ignoring the period register of Timer 4. This bit is a reserved bit in T3CON.
- 0 Use own period register
- 1 Use T1PR (in case of EVA) or T3PR (in case of EVB) as period register ignoring own period register

Modes de fonctionnement des timers (défini par les bits 12-11 de TxCON) :

Mode comptage continu



Mode comptage/décomptage continu

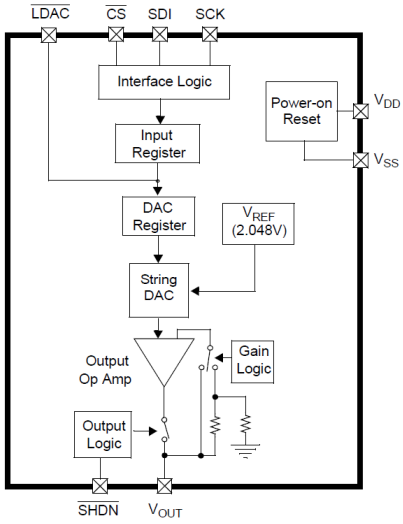


MICROCHIP MCP4801/4811/4821

8/10/12-Bit Voltage Output Digital-to-Analog Converter with Internal V_{REF} and SPI Interface

Features

- MCP4801: 8-Bit Voltage Output DAC
- MCP4811: 10-Bit Voltage Output DAC
- MCP4821: 12-Bit Voltage Output DAC
- Rail-to-Rail Output
- SPI Interface with 20 MHz Clock Support
- Simultaneous Latching of the DAC Output with $\overline{LDAC}$ Pin
- Fast Settling Time of 4.5 µs
- Selectable Unity or 2x Gain Output
- 2.048V Internal Voltage Reference
- 50 ppm/°C V_{REF} Temperature Coefficient
- 2.7V to 5.5V Single-Supply Operation
- Extended Temperature Range: -40°C to +125°C



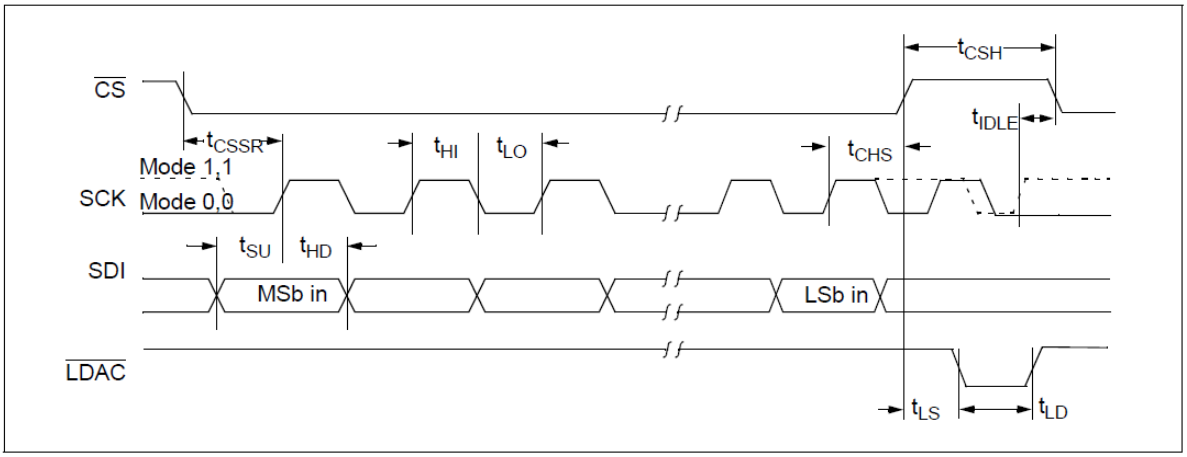


FIGURE 1-1: *SPI Input Timing Data.*

REGISTER 5-1: WRITE COMMAND REGISTER FOR MCP4821 (12-BIT DAC)

W-x	W-x	W-x	W-0	W-x	W-x	W-x	W-x	W-x	W-x	W-x	W-x	W-x	W-x	W-x	W-x
0	—	$\overline{GA}$	$\overline{SHDN}$	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
bit 15															bit 0

bit 15 (1) 0 = Write to DAC register
1 = Ignore this command

bit 14 — Don't Care

bit 13 **$\overline{GA}$:** Output Gain Selection bit
1 = 1x ($V_{OUT} = V_{REF} * D/4096$)
0 = 2x ($V_{OUT} = 2 * V_{REF} * D/4096$), where internal V_{REF} = 2.048V.

bit 12 **$\overline{SHDN}$:** Output Shutdown Control bit
1 = Active mode operation. V_{OUT} is available.
0 = Shutdown the device. Analog output is not available. V_{OUT} pin is connected to 500 kΩ (typical).

bit 11-0 **D11:D0:** DAC Input Data bits. Bit x is ignored.

Legend			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	1 = bit is set	0 = bit is cleared	x = bit is unknown