TMS320C642x DSP Multichannel Buffered Serial Port (McBSP) Interface

User's Guide

Literature Number: SPRUEN2B
September 2007

Texas Instruments

Contents

List of Figures

List of Tables

Read This First

About This Manual

Describes the operation of the multichannel buffered serial port (McBSP) interface in the TMS320C642x Digital Signal Processor (DSP). The McBSP consists of a data path and a control path that connect to external devices. Separate pins for transmission and reception communicate data to these external devices. The C642x CPU communicates to the McBSP using 32-bit-wide control registers accessible via the internal peripheral bus.

Notational Conventions

This document uses the following conventions.

- Hexadecimal numbers are shown with the suffix h. For example, the following number is 40 hexadecimal (decimal 64): 40h.
- Registers in this document are shown in figures and described in tables.
  - Each register figure shows a rectangle divided into fields that represent the fields of the register. Each field is labeled with its bit name, its beginning and ending bit numbers above, and its read/write properties below. A legend explains the notation used for the properties.
  - Reserved bits in a register figure designate a bit that is used for future device expansion.

Related Documentation From Texas Instruments

The following documents describe the TMS320C642x Digital Signal Processor (DSP). Copies of these documents are available on the Internet at www.ti.com. *Tip:* Enter the literature number in the search box provided at www.ti.com.

The current documentation that describes the C642x DSP, related peripherals, and other technical collateral, is available in the C6000 DSP product folder at: www.ti.com/c6000.

SPRUEM3 — *TMS320C642x DSP Peripherals Overview Reference Guide.* Provides an overview and briefly describes the peripherals available on the TMS320C642x Digital Signal Processor (DSP).

SPRAA84 — *TMS320C64x to TMS320C64x+ CPU Migration Guide.* Describes migrating from the Texas Instruments TMS320C64x digital signal processor (DSP) to the TMS320C64x+ DSP. The objective of this document is to indicate differences between the two cores. Functionality in the devices that is identical is not included.

SPRU732 — *TMS320C64x/C64x+ DSP CPU and Instruction Set Reference Guide.* Describes the CPU architecture, pipeline, instruction set, and interrupts for the TMS320C64x and TMS320C64x+ digital signal processors (DSPs) of the TMS320C6000 DSP family. The C64x/C64x+ DSP generation comprises fixed-point devices in the C6000 DSP platform. The C64x+ DSP is an enhancement of the C64x DSP with added functionality and an expanded instruction set.

SPRU871 — *TMS320C64x+ DSP Megamodule Reference Guide.* Describes the TMS320C64x+ digital signal processor (DSP) megamodule. Included is a discussion on the internal direct memory access (IDMA) controller, the interrupt controller, the power-down controller, memory protection, bandwidth management, and the memory and cache.

Trademarks

SPI is a trademark of Motorola, Inc..

Multichannel Buffered Serial Port (McBSP) Interface

1 Introduction

This document describes the operation of the multichannel buffered serial port (McBSP) interface in the TMS320C642x Digital Signal Processor (DSP). The McBSP consists of a data path and a control path that connect to external devices. Separate pins for transmission and reception communicate data to these external devices. The C642x CPU communicates to the McBSP using 32-bit-wide control registers accessible via the internal peripheral bus.

1.1 Purpose of the Peripheral

The primary use for the multichannel buffered serial port (McBSP) is for audio interface purposes. The McBSP is a specialized version of the McBSP peripheral used on other TI DSPs. The primary audio modes that are supported by the McBSP are the AC97 and IIS modes. In addition to the primary audio modes, the McBSP can be programmed to support other serial formats but is not intended to be used as a high-speed interface.

1.2 Features

The McBSP provides the following functions:

- Full-duplex communication
- Double-buffered data registers, which allow a continuous data stream
- Independent framing and clocking for receive and transmit
- Direct interface to industry-standard codecs, analog interface chips (AICs), and other serially connected analog-to-digital (A/D) and digital-to-analog (D/A) devices
- External shift clock or an internal, programmable frequency shift clock for data transfer

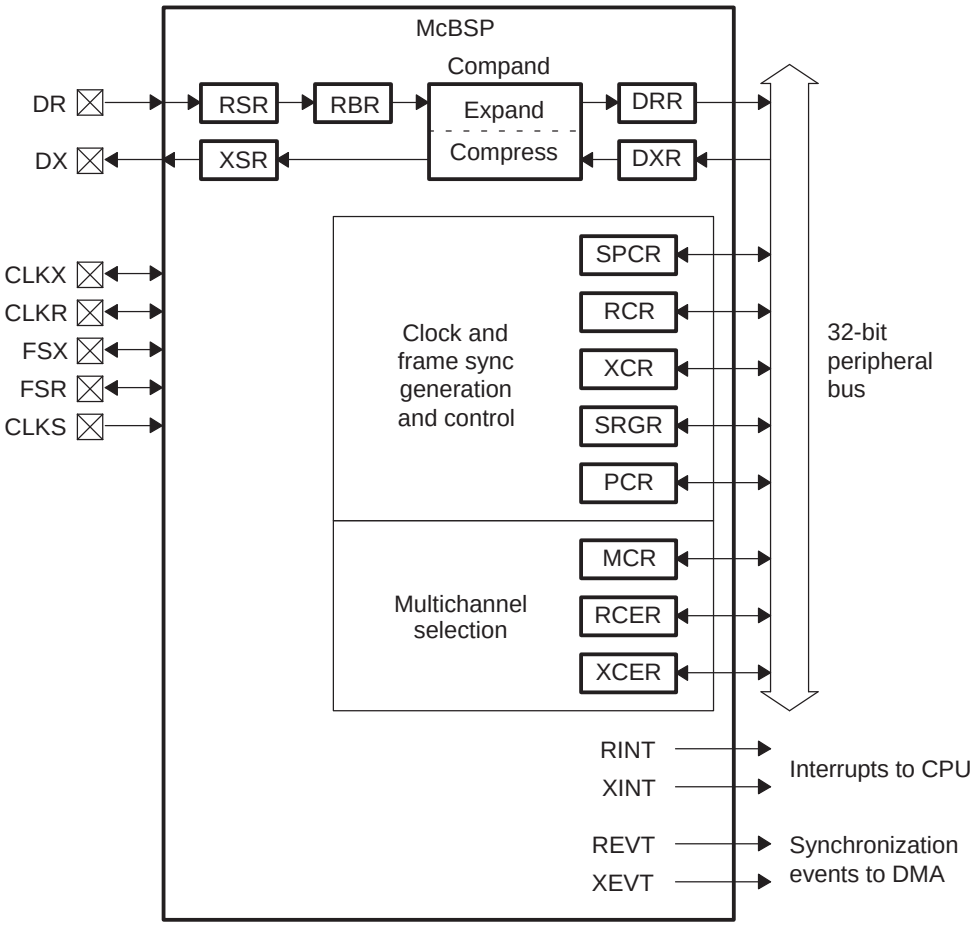
In addition, the McBSP has the following capabilities:

- Direct interface to:
  - T1/E1 framers
  - MVIP switching compatible and ST-BUS compliant devices including:
    - MVIP framers
    - H.100 framers
    - SCSA framers
  - IOM-2 compliant devices
  - AC97 compliant devices (the necessary multiphase frame synchronization capability is provided)
  - IIS compliant devices
  - SPI™ devices
- Multichannel transmit and receive of up to 128 channels
- A wide selection of data sizes, including 8, 12, 16, 20, 24, and 32 bits
- μ-Law and A-Law companding
- 8-bit data transfers with the option of LSB or MSB first
- Programmable polarity for both frame synchronization and data clocks
- Highly programmable internal clock and frame generation

1.3 Functional Block Diagram

The McBSP consists of a data path and control path, as shown in Figure 1

Figure 1. McBSP Block Diagram



1.4 Industry Standard Compliance Statement

The McBSP supports the following industry standard interfaces:

AC97— The AC97 standard specifies a 5-wire digital serial link between an audio codec device and its digital controller.

IIS— IIS is a protocol for transmitting two channels of digital audio data over a single serial connection. The IIS bus is an industry standard three-wire interface for streaming stereo audio between devices, typically between a CPU/DSP and a DAC/ADC.

2 Peripheral Architecture

This section describes the architecture of the McBSP.

2.1 Clock Control

The McBSP can use an internal or external clock source. Either clock source can be divided-down inside the McBSP to generate the actual interface bit clock frequency. For detailed timing information, see the device-specific data manual. Detailed information about how the interface clock and frame synchronization signals are generated is provided in Section 2.5.

2.2 Signal Descriptions

The signals used on the McBSP audio interface are listed in Table 1.

Table 1. McBSP Interface Signals

Pin	I/O/Z	Description
CLKR	I/O/Z	Receive clock - supplies or receives a reference clock for the receiver; or supplies a reference clock to the sample rate generator
CLKS	I	Supplies the input clock of the sample rate generator
CLKX	I/O/Z	Transmit clock - supplies or receives a reference clock for the transmitter; or supplies a reference clock to the sample rate generator
DR	I	Received serial data
DX	O/Z	Transmitted serial data
FSR	I/O/Z	Receive frame synchronization - control signal to synchronize the start of received data
FSX	I/O/Z	Transmit frame synchronization - control signal to synchronize the start of transmitted data

2.3 Pin Multiplexing

On the C642x DSP extensive pin multiplexing is used to accommodate the largest number of peripheral functions in the smallest possible package. Pin multiplexing is controlled using a combination of hardware configuration at device reset and software programmable register settings. Refer to the device-specific data manual to determine how pin multiplexing affects the McBSP.

2.4 Endianness Considerations

When the device is configured for big-endian mode, in order for the data to be placed in the right side of the register being accessed (that is, DXR or DRR), access to the McBSP registers must be performed as follows:

- **8-bit accesses:** An offset of 3h must be added to the address of the register being accessed. For example, the offset address of DRR becomes 3h (0 + 3h).
- **16-bit accesses:** An offset of 2h must be added to the address of the register being accessed. For example, the offset address of DRR becomes 2h (0 + 2h).
- **24-bit accesses** (only applicable to the EDMA, the CPU cannot perform 24-bit accesses): An offset of 1h must be added to the address of the register being accessed. For example, the offset address of DRR becomes 1h (0 + 1h).
- **32-bit accesses:** No offset is needed. For example, the offset address of DRR remains as 00h.

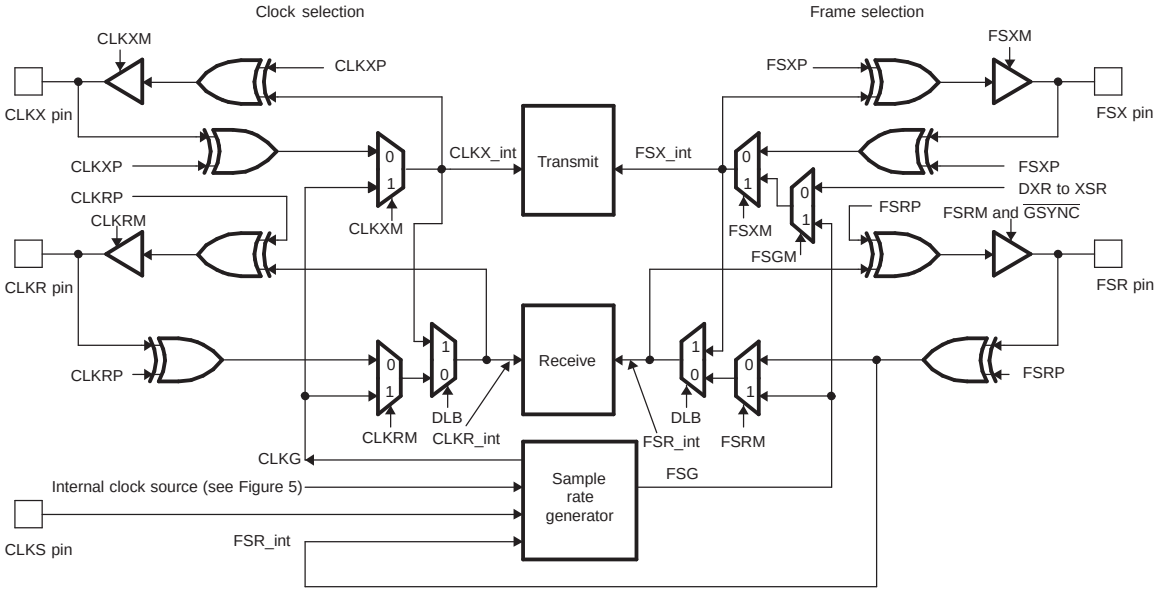
It is recommended to perform a 32-bit access to the control registers. This means that the same address offset is used for both big-endian and little-endian modes.

No offset is needed to access the McBSP registers when the device is configured in little-endian mode.

2.5 Clock, Frames, and Data

The McBSP has several ways of selecting clocking and framing for both the receiver and transmitter. Clocking and framing can be sent to both portions by the sample rate generator. Each portion can select external clocking and/or framing independently. Figure 2 is a block diagram of the clock and frame selection circuitry.

Figure 2. Clock and Frame Generation



2.5.1 Frame and Clock Operation

Receive and transmit frame sync pulses (FSR/X), and clocks (CLKR/X), can either be generated internally by the sample rate generator (see Section 2.5.2) or be driven by an external source. The source of frame sync and clock is selected by programming the mode bits, FS(R/X)M and CLK(R/X)M respectively, in the pin control register (PCR). FSR is also affected by the GSYNC bit in the sample rate generator register (SRGR), see Section 2.5.4.2 for details.

When FSR and FSX are inputs (FSXM = FSRM = 0), the McBSP detects them on the internal falling edge of clock, CLKR_int and CLKX_int, respectively (see Figure 2). The receive data arriving at the DR pin is also sampled on the falling edge of CLKR_int. These internal clock signals are either derived from external source via the CLK(R/X) pins or driven by the sample rate generator clock (CLKG) internal to the McBSP.

When FSR and FSX are outputs driven by the sample rate generator, they are generated (transition to their active state) on the rising edge of the internal clock, CLK(R/X)_int. Similarly, data on DX is output on the rising edge of CLKX_int.

The FSRP, FSXP, CLKRP, and CLKXP bits in PCR configure the polarities of FSR, FSX, CLKR, and CLKX. All frame sync signals (FSR_int and FSX_int) internal to the serial port are active high. If the serial port is configured for external frame synchronization (FSR/FSX are inputs to the McBSP) and FSRP = FSXP = 1, the external active (low) frame sync signals are inverted before being sent to the receiver signal (FSR_int) and transmitter signal (FSX_int). Similarly, if internal synchronization is selected (FSR/FSX are outputs and GSYNC = 0), the internal active (high) sync signals are inverted if the polarity bit FS(R/X)P = 1, before being sent to the FS(R/X) pin.Figure 2 shows this inversion using XOR gates.

On the transmit side, the transmit clock polarity bit, CLKXP, sets the edge used to shift and clock out transmit data. Data is always transmitted on the rising edge of CLKX_int (see Figure 3). If CLKXP = 1 and external clocking is selected (CLKXM = 0 and CLKX is an input), the external falling-edge-triggered input clock on CLKX is inverted to a rising-edge-triggered clock before being sent to the transmitter. If CLKXP = 1 and internal clocking is selected (CLKXM = 1 and CLKX is an output pin), the internal (rising-edge-triggered) clock, CLKX_int, is inverted before being sent out on the CLKX pin.

Similarly, the receiver can reliably sample data that is clocked (by the transmitter) with a rising-edge clock. The receive clock polarity bit, CLKRP, sets the edge used to sample received data. The receive data is always sampled on the falling edge of CLKR_int (see Figure 4). Therefore, if CLKRP = 1 and external clocking is selected (CLKRM = 0 and CLKR is an input pin), the external rising-edge triggered input clock on CLKR is inverted to a falling-edge clock before being sent to the receiver. If CLKRP = 1 and internal clocking is selected (CLKRM = 1), the internal falling-edge-triggered clock is inverted to a rising edge before being sent out on the CLKR pin.

In a system where the same clock (internal or external) is used to clock the receiver and transmitter, CLKRP = CLKXP. The receiver uses the opposite edge as the transmitter to ensure valid setup and hold times of data around this edge. Figure 4 shows how data clocked by an external serial device using a rising-edge clock can be sampled by the McBSP receiver with the falling edge of the same clock.

Figure 3. Transmit Data Clocking

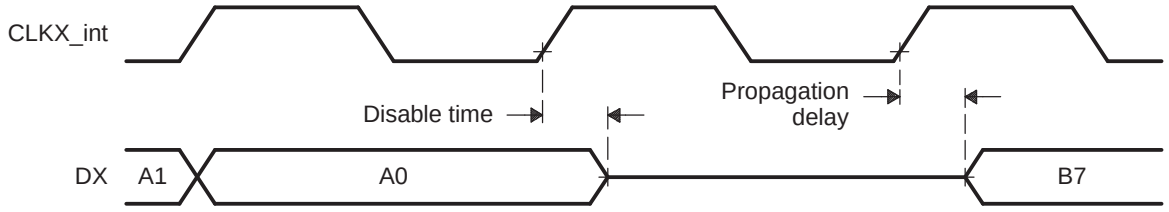
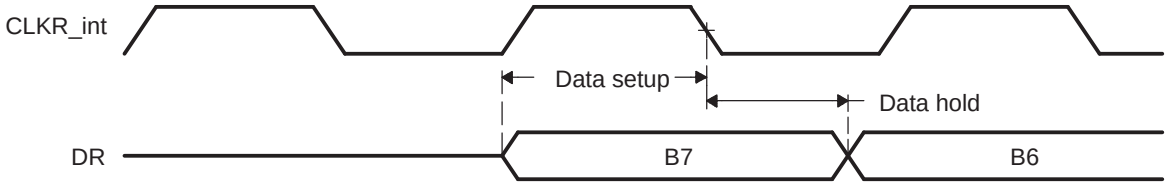


Figure 4. Receive Data Clocking



2.5.2 Sample Rate Generator Clocking and Framing

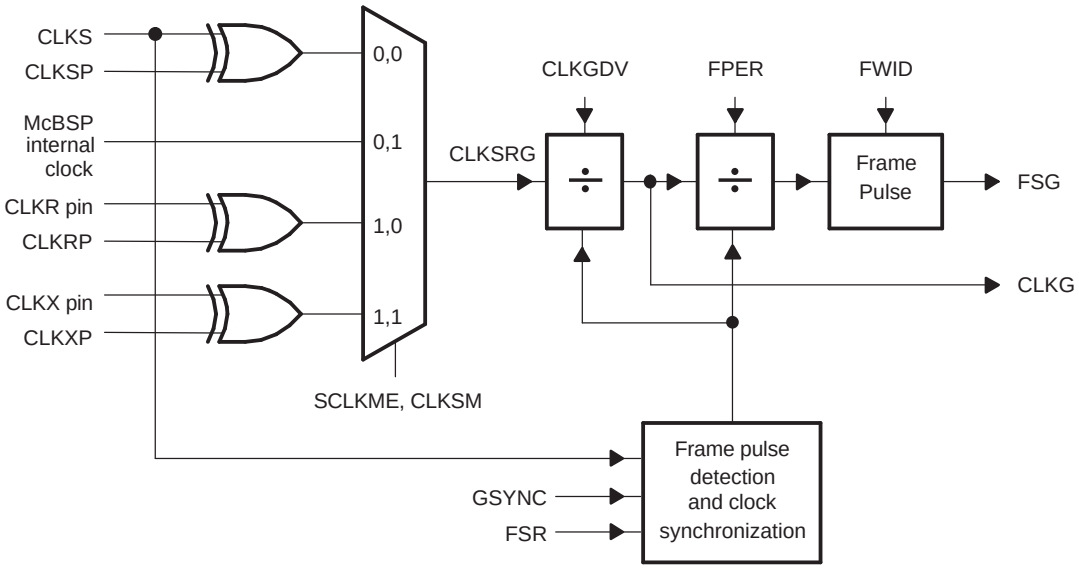
The sample rate generator is composed of a 3-stage clock divider that provides a programmable data clock (CLKG) and framing signal (FSG), as shown in Figure 5. CLKG and FSG are McBSP internal signals that can be programmed to drive receive and/or transmit clocking, CLK(R/X), and framing, FS(R/X). The sample rate generator can be programmed to be driven by an internal clock source or an internal clock derived from an external clock source.

The sample rate generator is not used when CLKX, FSX, CLKR, and FSR are driven by an external source. Therefore, the GRST bit in the serial port control register (SPCR) does not need to be enabled (GRST = 1) for this setup. The three stages of the sample rate generator circuit compute:

- Clock divide-down (CLKGDV): The number of input clocks per data bit clock
- Frame period (FPER): The frame period in data bit clocks
- Frame width (FWID): The width of an active frame pulse in data bit clocks

In addition, a frame pulse detection and clock synchronization module allows synchronization of the clock divide-down with an incoming frame pulse. The operation of the sample rate generator during device reset is described in Section 2.10.

Figure 5. Sample Rate Generator Block Diagram



2.5.3 Data Clock Generation

When the receive/transmit clock mode is set to 1 (CLK(R/X)M = 1 in the pin control register (PCR)), the data clocks (CLK(R/X)) are driven by the internal sample rate generator output clock, CLKG. You can select for the receiver and transmitter from a variety of data bit clocks including:

- The input clock to the sample rate generator, which can be either the internal clock source or a dedicated external clock source via the CLKX, CLKR, or CLKS pins. The McBSP internal clock is the CPU/6 clock. See Section 2.5.3.1 for details on the source of the McBSP internal clock.
- The input clock source (internal clock source or external clock CLKX/CLKR/CLKS) to the sample rate generator can be divided-down by a programmable value (CLKGDV bit in the sample rate generator register (SRGR)) to drive CLKG.

Regardless of the source to the sample rate generator, the rising edge of CLKSRG (see Figure 5) generates CLKG and FSG.

2.5.3.1 Input Clock Source Mode: CLKSM and SCLKME

The sample rate generator input clock signal can be driven from one of four sources selectable with the SCLKME bit in the pin control register (PCR) and the CLKSM bit in the sample rate generator register (SRGR), see Table 2.

Table 2. Choosing an Input Clock for the Sample Rate Generator With the SCLKME and CLKSM Bits

SCLKME Bit in PCR	CLKSM Bit in SRGR	Input Clock for Sample Rate Generator
0	0	Signal on CLKS pin
0	1	McBSP internal input clock
1	0	Signal on CLKR pin
1	1	Signal on CLKX pin

2.5.3.2 Sample Rate Generator Data Bit Clock Rate: CLKGDV

The first divider stage generates the serial data bit clock from the input clock. This divider stage uses a counter that is preloaded by the CLKGDV bit in the sample rate generator register (SRGR) and that contains the divide ratio value. The output of this stage is the data bit clock that is output on the sample rate generator output, CLKG, and that serves as the input for the second and third divider stages.

CLKG has a frequency equal to 1/(CLKGDV + 1) of the sample rate generator input clock. Thus, the sample rate generator input clock frequency is divided by a value between 1 to 256. The CLKGDV value chosen must result in a clock that meets the timing requirements/limitations specified in the device-specific data manual.

When CLKGDV is an odd value or equal to 0, the CLKG duty cycle is 50%. Note that an odd CLKGDV value means an even divide down of the source clock and an even CLKGDV value means an odd divide down of the source clock. When CLKGDV is an even value (2p), the high state duration is p + 1 cycles and the low state duration is p cycles. This is illustrated in Example 1, Example 2, and Example 3.

In the following examples:

S_{IN} = sample generator input clock period

f_{IN} = sample generator input clock frequency

S_G = CLKG period

f_G = CLKG frequency

The following equation is given above: $f_G = f_{IN}/(CLKGDV + 1)$; therefore, $S_G = (CLKGDV + 1) \times S_{IN}$.

Example 1. CLKGDV = 0

CLKGDV = 0

$S_G = (CLKGDV + 1) \times S_{IN} = (0 + 1) \times S_{IN} = S_{IN}$

Pulse width high = $S_{IN} \times (CLKGDV + 1)/2 = S_{IN} \times (0+ 1)/2 = 0.5 \times S_{IN}$

Pulse width low = $S_{IN} \times (CLKGDV + 1)/2 = S_{IN} \times (0+ 1)/2 = 0.5 \times S_{IN}$

Example 2. CLKGDV = 1

CLKGDV = 1

$S_G = (CLKGDV + 1) \times S_{IN} = (1 + 1) \times S_{IN} = 2 \times S_{IN}$

Pulse width high = $S_{IN} \times (CLKGDV + 1)/2 = S_{IN} \times (1 +1)/2 = S_{IN}$

Pulse width low = $S_{IN} \times (CLKGDV + 1)/2 = S_{IN} \times (1+ 1)/2 = S_{IN}$

Example 3. CLKGDV = 2

CLKGDV = 2

$S_G = (CLKGDV + 1) \times S_{IN} = (2 + 1) \times S_{IN} = 3 \times S_{IN}$

Pulse width high = $S_{IN} \times (CLKGDV/2 + 1) = S_{IN} \times (2/2 + 1) = 2 \times S_{IN}$

Pulse width low = $S_{IN} \times CLKGDV/2 = S_{IN} \times 2/2 = 1 \times S_{IN}$

2.5.3.3 Bit Clock Polarity: CLKSP

The external clock (CLKS) is selected to drive the sample rate generator clock divider by selecting CLKSM = 0 in the sample rate generator register (SRGR) and SCLKME = 0 in the pin control register (PCR). In this case, the CLKSP bit in SRGR selects the edge of CLKS on which sample rate generator data bit clock (CLKG) and frame sync signal (FSG) are generated. Since the rising edge of CLKSRG generates CLKG and FSG, the rising edge of CLKS when CLKSP = 0 or the falling edge of CLKS when CLKSP = 1 causes the transition on CLKG and FSG.

2.5.3.4 Bit Clock and Frame Synchronization

When the external clock (CLKS) is selected to drive the sample rate generator (CLKSM = 0 in SRGR and SCLKME = 0 in PCR), the GSYNC bit in SRGR can be used to configure the timing of CLKG relative to CLKS. GSYNC = 1 ensures that the McBSP and the external device to which it is communicating are dividing down the CLKS with the same phase relationship. If GSYNC = 0, this feature is disabled and CLKG runs freely and is not resynchronized. If GSYNC = 1, an inactive-to-active transition on FSR triggers a resynchronization of CLKG and the generation of FSG. CLKG always begins at a high state after synchronization. Also, FSR is always detected at the same edge of CLKS that generates CLKG, regardless of the length the FSR pulse. Although an external FSR is provided, FSG can still drive internal receive frame synchronization when GSYNC = 1. When GSYNC = 1, FPER does not matter, because the frame period is determined by the arrival of the external frame sync pulse.

Figure 6 and Figure 7 show this operation with various polarities of CLKS and FSR. These figures assume that FWID is 0, for a FSG = 1 CLKG wide.

These figures show what happens to CLKG when it is initially in sync and GSYNC = 1, as well as when it is not in sync with the frame synchronization and GSYNC = 1.

Figure 6. CLKG Synchronization and FSG Generation When GSYNC = 1 and CLKGDV = 1

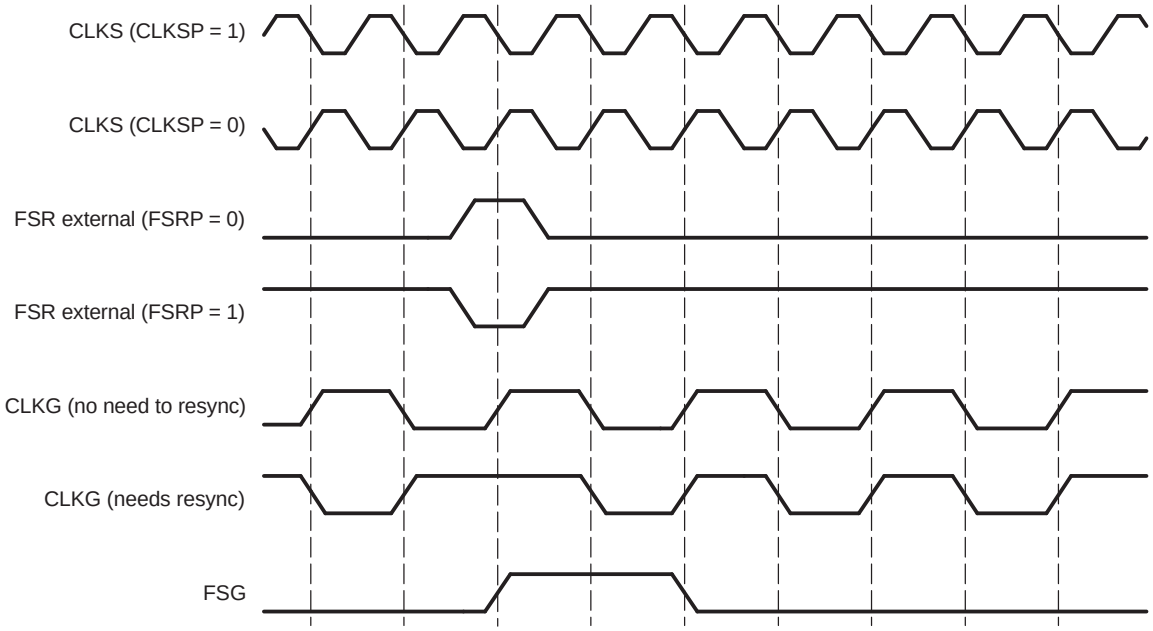
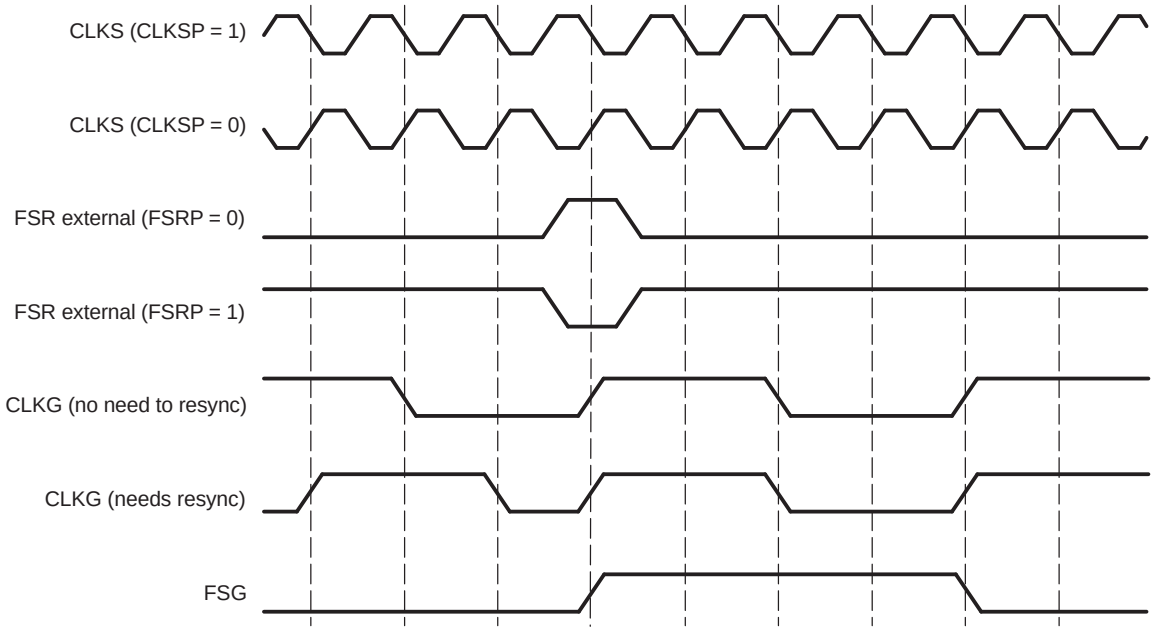


Figure 7. CLKG Synchronization and FSG Generation When GSYNC = 1 and CLKGDV = 3



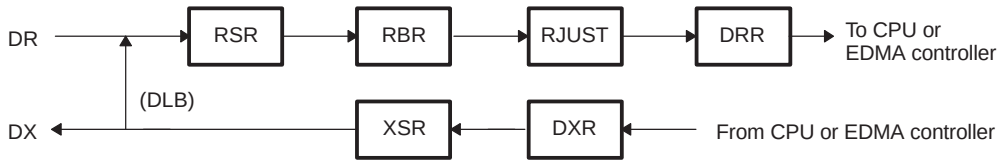
When GSYNC = 1, the transmitter can operate synchronously with the receiver, provided that the following conditions are met:

- FSX is programmed to be driven by the sample rate generator frame sync, FSG (FSGM = 1 in SRGR and FSXM = 1 in PCR).
- The sample-rate generator clock should drive the transmit and receive bit clock (CLK(R/X)M = 1 in SPCR). Therefore, the CLK(R/X) pin should not be driven by any other source.

2.5.3.5 Digital Loopback Mode: DLB

The DLB mode cannot be used when the McBSP is in clock stop mode (CLKSTP = 2h or 3h in the serial port control register (SPCR)). Setting DLB = 1 in SPCR enables digital loopback mode. In DLB mode, DR, FSR, and CLKR are internally connected through multiplexers to DX, FSX, and CLKX, respectively, as shown in Figure 2 and Figure 8. DLB mode allows testing of serial port code without using the external interface of the McBSP. CLKX and FSX must be enabled as outputs (CLKXM = FSXM = 1) in DLB mode.

Figure 8. Digital Loopback Mode



2.5.3.6 Receive Clock Selection: DLB, CLKRM

Table 3 shows how the digital loopback bit (DLB) in the serial port control register (SPCR) and the CLKRM bit in the pin control register (PCR) select the receiver clock. In digital loopback mode (DLB = 1), the transmitter clock drives the receiver. CLKRM determines whether the CLKR pin is an input or an output.

Table 3. Receive Clock Selection

DLB Bit in SPCR	CLKRM Bit in PCR	Source of Receive Clock	CLKR Function
0	0	CLKR acts as an input driven by the external clock and inverted as determined by CLKRP before being used.	Input.
0	1	The sample rate generator clock (CLKG) drives CLKR.	Output. CLKG inverted as determined by CLKRP before being driven out on CLKR.
1	0	CLKX_int drives the receive clock CLKR_int as selected and is inverted.	High impedance.
1	1	CLKX_int drives CLKR_int as selected and is inverted.	Output. CLKR (same as CLKX) is inverted as determined by CLKRP before being driven out.

2.5.3.7 Transmit Clock Selection: CLKXM

Table 4 shows how the CLKXM bit in the pin control register (PCR) selects the transmit clock and whether the CLKX pin is an input or output.

Table 4. Transmit Clock Selection

CLKXM Bit in PCR	Source of Transmit Clock	CLKX Function
0	The external clock drives the CLKX input pin. CLKX is inverted as determined by CLKXP before being used.	Input.
1	The sample rate generator clock (CLKG) drives the transmit clock.	Output. CLKG is inverted as determined by CLKXP before being driven out on CLKX.

2.5.3.8 Stopping Clocks

There are two methods to stop serial clocks between data transfers:

- Using the SPI CLKSTP mode where clocks are stopped between single-element transfers. This is described in Section 2.9.
- When the clocks are inputs to the McBSP (CLKXM or CLKRM = 0 in the pin control register (PCR)) and the McBSP operates in non-SPI mode (the clocks can be stopped between data transfers). If the external device stops the serial clock between data transfers, the McBSP interprets it as a slowed-down serial clock. Ensure that there are no glitches on the CLK(R/X) lines as the McBSP may interpret them as clock-edge transitions. Restarting the serial clock is equivalent to a normal clock transition after a slow CLK(R/X) cycle. Note that just as in normal operations, transmit under flow (XEMPTY) may occur if the DXR is not properly serviced at least three CLKX cycles before the next frame sync. Therefore, if the serial clock is stopped before DXR is properly serviced, the external device needs to restart the clock at least three CLKX cycles before the next frame sync to allow the DXR write to be properly synchronized. See Figure 29 for a graphical explanation on when DXR needs to be written to avoid underflow.

2.5.4 Frame Sync Generation

Data frame synchronization is independently programmable for the receiver and the transmitter for all data delay values. When the FRST bit in the serial port control register (SPCR) is set to 1 the frame generation logic is activated to generate frame sync signals, provided that FSGM = 1 in the sample rate generator register (SRGR). The frame sync programming options are:

- A frame pulse with a programmable period between sync pulses and a programmable active width specified in SRGR.
- The transmitter can trigger its own frame sync signal that is generated by a DXR-to-XSR copy. This causes a frame sync to occur on every DXR-to-XSR copy. The data delays can be programmed as required. However, maximum packet frequency cannot be achieved in this method for data delays of 1 and 2.
- Both the receiver and transmitter can independently select an external frame synchronization on the FSR and FSX pins, respectively.

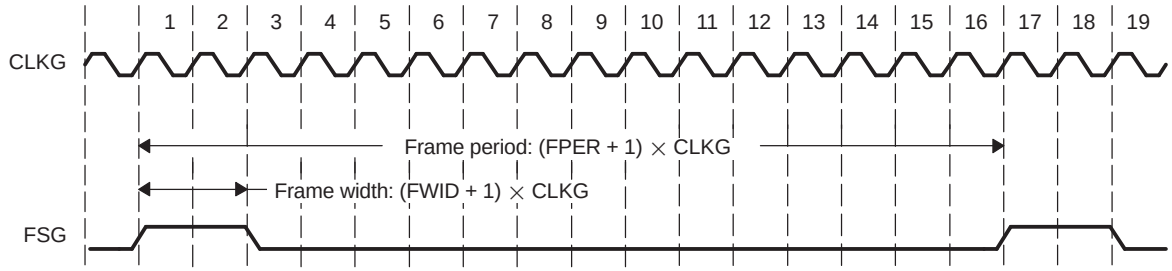
2.5.4.1 Frame Period (FPER) and Frame Width (FWID)

The FPER bits in the sample rate generator register (SRGR) are a 12-bit down-counter that can count down the generated data clocks from 4095 to 0. FPER controls the period of active frame sync pulses. The FWID bits in SRGR are an 8-bit down-counter. FWID controls the active width of the frame sync pulse.

When the sample rate generator comes out of reset, FSG is in an inactive (low) state. After this, when FRST = 1 in SPCR and FSGM = 1 in SRGR, frame sync signals are generated. The frame width value (FWID + 1) is counted down on every CLKG cycle until it reaches 0 when FSG goes low. Thus, the value of FWID + 1 determines an active frame pulse width ranging from 1 to 256 data bit clocks. At the same time, the frame period value (FPER + 1) is also counting down, and when this value reaches 0, FSG goes high again, indicating a new frame is beginning. Thus, the value of FPER + 1 determines a frame length from 1 to 4096 data bits. When GSYNC = 1 in SRGR, the value of FPER does not matter. Figure 9 shows a frame of 16 CLKG periods (FPER = 15 or 0000 1111b).

It is recommended that FWID be programmed to a value less than (R/X)WDLEN1/2.

Figure 9. Programmable Frame Period and Width



2.5.4.2 Receive Frame Synchronization Selection: DLB and FSRM

Table 5 shows how you can select various sources to provide the receive frame synchronization signal. Note that in digital loopback mode (DLB = 1 in the serial port control register (SPCR)), the transmit frame sync signal is used as the receive frame sync signal and that DR is internally connected to DX.

Note: FSR_int and FSX_int are shown in Figure 2.

Table 5. Receive Frame Synchronization Selection

DLB Bit in SPCR	FSRM Bit in PCR	GSYNC Bit in SRGR	Source of Receive Frame Synchronization	FSR Pin Function
0	0	X	External frame sync signal drives the FSR input pin, whose signal is then inverted as determined by FSRP before being used as FSR_int.	Input.
0	1	0	Sample rate generator frame sync signal (FSG) drives FSR_int, FRST = 1.	Output. FSG is inverted as determined by FSRP before being driven out on the FSR pin.
0	1	1	Sample rate generator frame sync signal (FSG) drives FSR_int, FRST = 1.	Input. The external frame sync input on FSR is used to synchronize CLKG and generate FSG.
1	0	0	FSX_int drives FSR_int. FSX is selected as shown in Table 6.	High impedance.
1	X	1	FSX_int drives FSR_int and is selected as shown in Table 6.	Input. External FSR is not used for frame synchronization but is used to synchronize CLKG and generate FSG since GSYNC = 1.
1	1	0	FSX_int drives FSR_int and is selected as shown in Table 6.	Output. Receive (same as transmit) frame synchronization is inverted as determined by FSRP before being driven out.

2.5.4.3 Transmit Frame Synchronization Selection: FSXM and FSGM

Table 6 shows how you can select the source of the transmit frame synchronization signal. The three choices are:

- External frame sync input
- The sample rate generator frame sync signal, FSG
- A signal that indicates a DXR-to-XSR copy has been made

Note: FSR_int and FSX_int are shown in Figure 2.

Table 6. Transmit Frame Synchronization Selection

FSXM Bit in PCR	FSGM Bit in SRGR	Source of Transmit Frame Synchronization	FSX Pin Function
0	X	External frame sync input on the FSX pin. This is inverted by FSXP before being used as FSX_int.	Input.
1	1	Sample rate generator frame sync signal (FSG) drives FSX_int. FRST = 1.	Output. FSG is inverted by FSXP before being driven out on FSX.
1	0	A DXR-to-XSR copy activates transmit frame sync signal.	Output. 1-bit-clock-wide signal inverted as determined by FSXP before being driven out on FSX.

2.5.4.4 Frame Detection

To facilitate detection of frame synchronization, the receive and transmit CPU interrupts (RINT and XINT) can be programmed to detect frame synchronization by setting the RINTM and XINTM bits in the serial port control register (SPCR) to 10b. The associated portion (receiver/transmitter) of the McBSP must be out of reset.

2.5.5 Data and Frames

2.5.5.1 Frame Synchronization Phases

Frame synchronization indicates the beginning of a transfer on the McBSP. The data stream following frame synchronization can have up to two phases, phase 1 and phase 2. The number of phases can be selected by the phase bit, (R/X)PHASE, in RCR and XCR. The number of elements per frame and bits per element can be independently selected for each phase via (R/X)FRLEN1/2 and (R/X)WDLEN1/2, respectively. Figure 10 shows a frame in which the first phase consists of two elements of 12 bits, each followed by a second phase of three elements of 8 bits each. The entire bit stream in the frame is contiguous; no gaps exist either between elements or phases. Table 7 shows the fields in the receive/transmit control registers (RCR/XCR) that control the frame length and element length for each phase for both the receiver and the transmitter. The maximum number of elements per frame is 128 for a single-phase frame and 256 elements in a dual-phase frame. The number of bits per element can be 8, 12, 16, 20, 24, or 32.

Note: For a dual-phase frame with internally generated frame synchronization, the maximum number of elements per phase depends on the word length. This is because the frame period, FPER, is only 12-bits wide and, therefore, provides 4096 bits per frame. Hence, the maximum number of 256 elements per dual- phase frame applies only when the WDLEN is 16 bits. However, any combination of element numbers and element size (defined by the FRLEN and WDLEN bits, respectively) is valid as long as their product is less than or equal to 4096 bits. This limitation does not apply for dual-phase with external frame sync.

Figure 10. Dual-Phase Frame Example

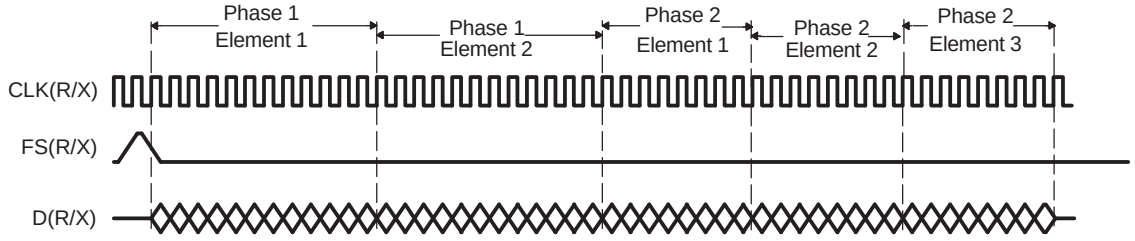


Table 7. RCR/XCR Fields Controlling Elements per Frame and Bits per Element

Serial Port	Frame Phase	RCR/XCR Field Control	
		Elements per Frame	Bits per Element
Receive	1	RFRLEN1	RWDLEN1
Receive	2	RFRLEN2	RWDLEN2
Transmit	1	XFRLEN1	XWDLEN1
Transmit	2	XFRLEN2	XWDLEN2

2.5.5.2 Frame Length: RFRLEN1/2 and XFRLEN1/2

Frame length specifies the maximum number of serial elements or logical time slots or channels that are available for transfer per frame synchronization signal. In multichannel selection mode, the frame length value is independent of, and perhaps different from, the actual number of channels that the DSP is programmed to receive or transmit per frame via the MCR, RCERE*n*, and XCERE*n* registers. See Section 2.8 for details on multichannel selection mode operation. The 7-bit (R/X)FRLEN1/2 bits in (R/X)CR support up to 128 elements per phase in a frame, as shown in Table 8. (R/X)PHASE = 0 selects a single-phase data frame, and (R/X)PHASE = 1 selects a dual-phase frame for the data stream. For a single-phase frame, the value of (R/X)FRLEN2 does not matter. Program the frame length fields with (*w* minus 1), where *w* represents the number of elements per frame. For Figure 10, (R/X)FRLEN1 = 1 or 000 0001b and (R/X)FRLEN2 = 2 or 000 0010b.

Table 8. Receive/Transmit Frame Length Configuration

(R/X)PHASE	(R/X)FRLEN1	(R/X)FRLEN2	Frame Length
0	$0 \le n \le 127$	x	Single-phase frame; (n + 1) elements per frame
1	$0 \le n \le 127$	$0 \le m \le 127$	Dual-phase frame; (n + 1) plus (m + 1) elements per frame

2.5.5.3 Element Length: RWDLEN1/2 and XWDLEN1/2

The (R/X)WDLEN1/2 fields in the receive/transmit control register (RCR and XCR) determine the element length in bits per element for the receiver and the transmitter for each phase of the frame, as indicated in Table 7. Table 9 shows how the value of these fields selects particular element lengths in bits. For the example in Figure 10, (R/X)WDLEN1 = 001b and (R/X)WDLEN2 = 000b. If (R/X)PHASE = 0, indicating a single-phase frame, (R/X)WDLEN2 is not used by the McBSP and its value does not matter.

Table 9. Receive/Transmit Element Length Configuration

(R/X)WDLEN1/2	Element Length (Bits)
000	8
001	12
010	16
011	20
100	24
101	32
110	Reserved
111	Reserved

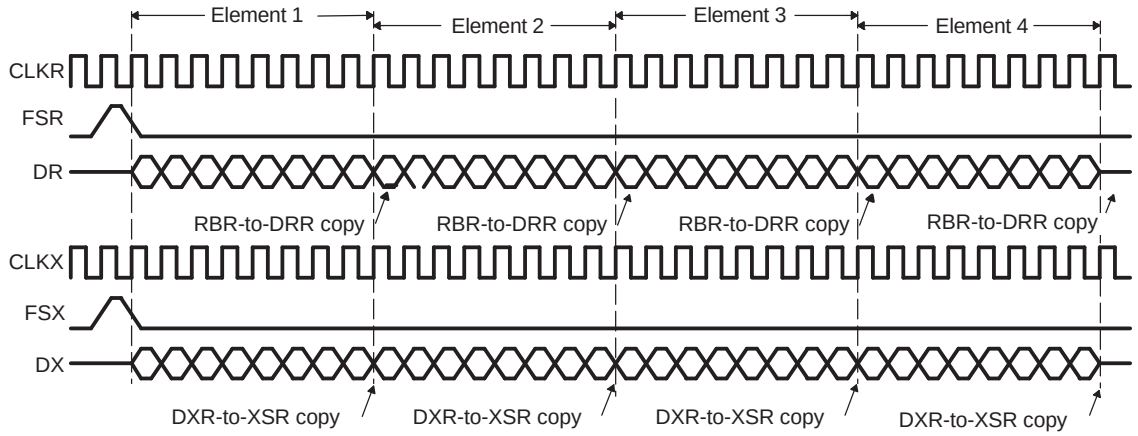
2.5.5.4 Data Packing Using Frame Length and Element Length

The frame length and element length can be manipulated to effectively pack data. For example, consider a situation in which four 8-bit elements are transferred in a single-phase frame, as shown in Figure 11. In this case:

- (R/X)PHASE = 0, indicating a single-phase frame
- (R/X)FRLEN1 = 000 0011b, indicating a 4-element frame
- (R/X)WDLEN1 = 000b, indicating 8-bit elements

In this example, four 8-bit data elements are transferred to and from the McBSP by the CPU or the EDMA controller. Four reads of DRR and four writes of DXR are necessary for each frame.

Figure 11. Single-Phase Frame of Four 8-Bit Elements

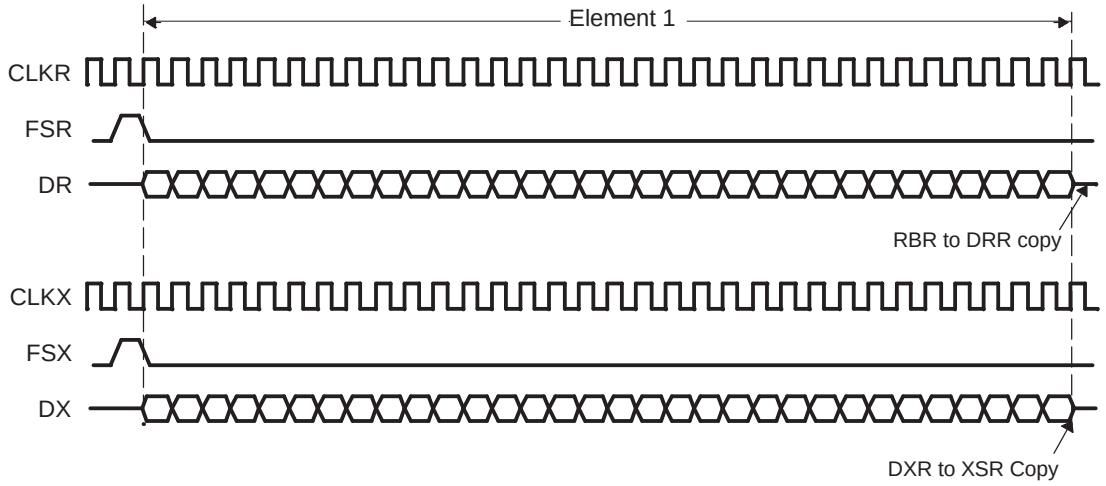


The example in Figure 11 can also be viewed as a data stream of a single-phase frame of one 32-bit data element, as shown in Figure 12. In this case:

- (R/X)PHASE = 0, indicating a single phase frame
- (R/X)FRLEN1 = 0, indicating a 1-element frame
- (R/X)WDLEN1 = 101b, indicating 32-bit elements

In this example, one 32-bit data element is transferred to and from the McBSP by the CPU or the EDMA controller. Thus, one read of DRR and one write of DXR is necessary for each frame. As a result, the number of transfers is one-fourth that of the previous example (Figure 11). This manipulation reduces the percentage of bus time required for serial port data movement.

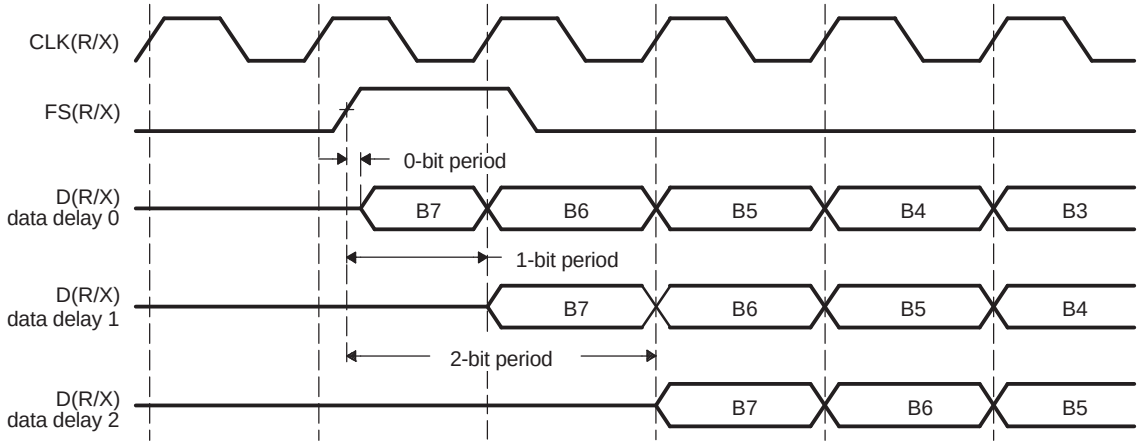
Figure 12. Single-Phase Frame of One 32-Bit Element



2.5.5.5 Data Delay: RDATDLY and XDATDLY

The start of a frame is defined by the first clock cycle in which frame synchronization is active. The beginning of actual data reception or transmission with respect to the start of the frame can be delayed if required. This delay is called data delay. RDATDLY (in RCR) and XDATDLY (in XCR) specify the data delay for reception and transmission, respectively. The range of programmable data delay is zero to two bit clocks ((R/X)DATDLY = 00b to10b), as shown in Figure 13. Typically, a 1-bit delay is selected because data often follows a 1-cycle active frame sync pulse.

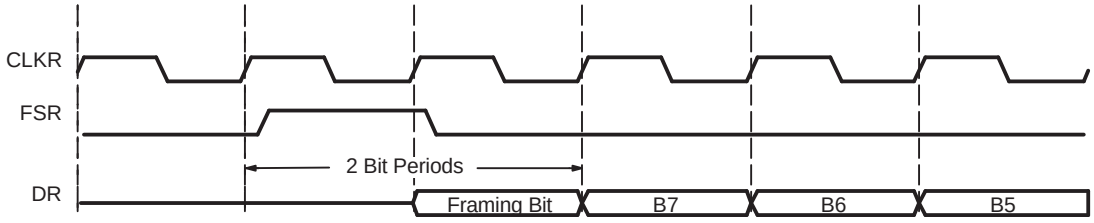
Figure 13. Data Delay



Normally a frame sync pulse is detected or sampled with respect to an edge of serial clock CLK(R/X). Thus, on a subsequent cycle (depending on data delay value), data can be received or transmitted. However, in the case of a 0-bit data delay, the data must be ready for reception and/or transmission on the same serial clock cycle. For reception, this problem is solved by receive data being sampled on the first falling edge of CLKR when an active (high) FSR is detected. However, data transmission must begin on the rising edge of CLKX that generated the frame synchronization. Therefore, the first data bit is assumed to be in the XSR and DX. The transmitter then asynchronously detects the frame synchronization, FSX goes active, and it immediately starts driving the first bit to be transmitted on the DX pin.

Another common operation uses a data delay of 2. This configuration allows the serial port to interface to different types of T1 framing devices in which the data stream is preceded by a framing bit. During the reception of such a stream with a data delay of two bits, the framing bit appears after a 1-bit delay and data appears after a 2-bit delay). The serial port essentially discards the framing bit from the data stream, as shown in Figure 14. In transmission, by delaying the first transfer bit, the serial port essentially inserts a blank period (high-impedance period) in place of the framing bit. Here, it is expected that the framing device inserts its own framing bit or that the framing bit is generated by another device. Alternatively, you may pull up or pull down DX to achieve the desired value.

Figure 14. 2-Bit Data Delay Used to Discard Framing Bit



2.5.5.6 Receive Data Justification and Sign Extension: RJUST

The RJUST bit in the serial port control register (SPCR) selects whether data in RBR is right- or left-justified (with respect to the MSB) in the DRR. If right justification is selected, RJUST further selects whether the data is sign-extended or zero-filled. Table 10 and Table 11 summarize the effect that various values of RJUST have on example receive data.

Table 10. Effect of RJUST Bit Values With 12-Bit Example Data ABCh

RJUST Bit in SPCR	Justification	Extension	Value in DRR
00	Right	Zero-fill MSBs	0000 0ABCh
01	Right	Sign-extend MSBs	FFFF FABCh
10	Left	Zero-fill LSBs	ABC0 0000h
11	Reserved	Reserved	Reserved

Table 11. Effect of RJUST Bit Values With 20-Bit Example Data ABCDEh

RJUST Bit in SPCR	Justification	Extension	Value in DRR
00	Right	Zero-fill MSBs	000A BCDEh
01	Right	Sign-extend MSBs	FFFA BCDEh
10	Left	Zero-fill LSBs	ABCD E000h
11	Reserved	Reserved	Reserved

2.5.5.7 32-Bit Bit Reversal: RWDREVRS, XWDREVRS

Normally all transfers are sent and received with the MSB first; however, you can reverse the receive/transmit bit ordering of a 32-bit element (LSB first) using the 32-bit reversal feature of the McBSP by setting all of the following:

- (R/X)WDREVRS = 1 in the receive/transmit control register (RCR/XCR).
- (R/X)COMPAND = 01b in RCR/XCR.
- (R/X)WDLEN(1/2) = 101b in RCR/XCR to indicate 32-bit elements.

When you set the register fields as above, the bit ordering of the 32-bit element is reversed before being received by or sent from the serial port. If the (R/W)WDREVRS and (R/X)COMPAND fields are set as above, but the element size is not set to 32-bit, operation is undefined.

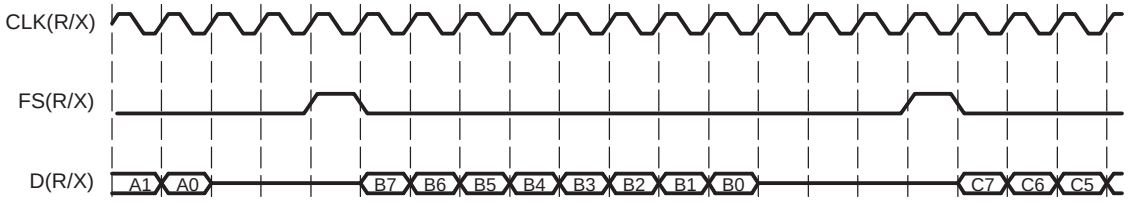
2.6 McBSP Standard Operation

During a serial transfer, there are typically periods of serial port inactivity between packets or transfers. The receive and transmit frame synchronization pulse occurs for every serial transfer. When the McBSP is not in the reset state and has been configured for the desired operation, a serial transfer can be initiated by programming (R/X)PHASE = 0 for a single-phase frame with the required number of elements programmed in (R/X)FRLEN1. The number of elements can range from 1 to 128 ((R/X)FRLEN1 = 00h to 7Fh). The required serial element length is set in the (R/X)WDLEN1 field in the (R/X)CR. If a dual-phase frame is required for the transfer, RPHASE = 1 and each (R/X)FRLEN1/2 can be set to any value between 0h and 7Fh.

Figure 15 shows a single-phase data frame of one 8-bit element. Since the transfer is configured for a 1-bit data delay, the data on the DX and DR pins are available one bit clock after FS(R/X) goes active. This figure, as well as all others in this section, use the following assumptions:

- (R/X)PHASE = 0, specifying a single-phase frame
- (R/X)FRLEN1 = 0b, specifying one element per frame
- (R/X)WDLEN1 = 000b, specifying eight bits per element
- (R/X)FRLEN2 = (R/X)WDLEN2 = Value is ignored
- CLK(R/X)P = 0, specifying that the receive data is clocked on the falling edge and that transmit data is clocked on the rising edge
- FS(R/X)P = 0, specifying that active (high) frame sync signals are used
- (R/X)DATDLY = 01b, specifying a 1-bit data delay

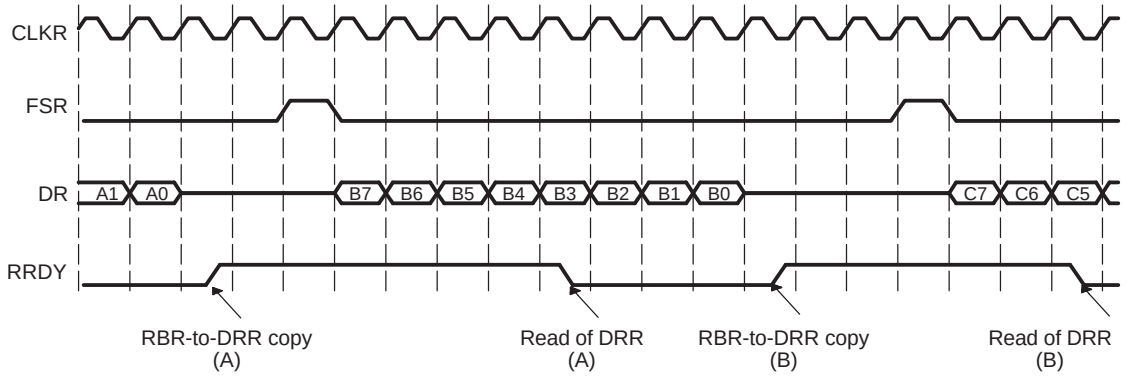
Figure 15. McBSP Standard Operation



2.6.1 Receive Operation

Figure 16 shows serial reception. Once the receive frame synchronization signal (FSR) transitions to its active state, it is detected on the first falling edge of the receivers CLKR. The data on the DR pin is then shifted into the receive shift register (RSR) after the appropriate data delay as set by the RDATDLY bit in the receive control register (RCR). The contents of RSR is copied to RBR at the end of every element on the rising edge of the clock, provided RBR is not full with the previous data. Then, an RBR-to-DRR copy activates the RRDY status bit in the serial port control register (SPCR) to 1 on the following falling edge of CLKR. This indicates that the receive data register (DRR) is ready with the data to be read by the CPU or the EDMA controller. RRDY is deactivated when the DRR is read by the CPU or the EDMA controller. See also Section 2.12.1.2 and Section 2.13.1.

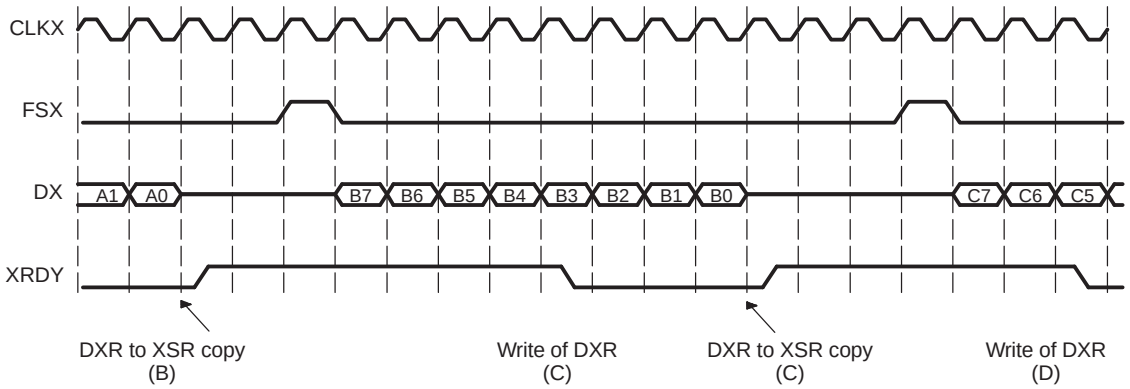
Figure 16. Receive Operation



2.6.2 Transmit Operation

Once transmit frame synchronization occurs, the value in the transmit shift register (XSR) is shifted out and driven on the DX pin after the appropriate data delay as set by the XDATDLY bit in the transmit control register (XCR). The XRDY bit in the serial port control register (SPCR) is activated after every DXR-to-XSR copy on the following falling edge of CLKX, indicating that the data transmit register (DXR) can be written with the next data to be transmitted. XRDY is deactivated when the DXR is written by the CPU or the EDMA controller. Figure 17 illustrates serial transmission. See Section 2.6.5.4 for information on transmit operation when the transmitter is pulled out of reset (XRST = 1). See also Section 2.12.1.3 and Section 2.13.2.

Figure 17. Transmit Operation



2.6.3 Maximum Frame Frequency

The frame frequency is determined by the following equation, which calculates the period between frame synchronization signals:

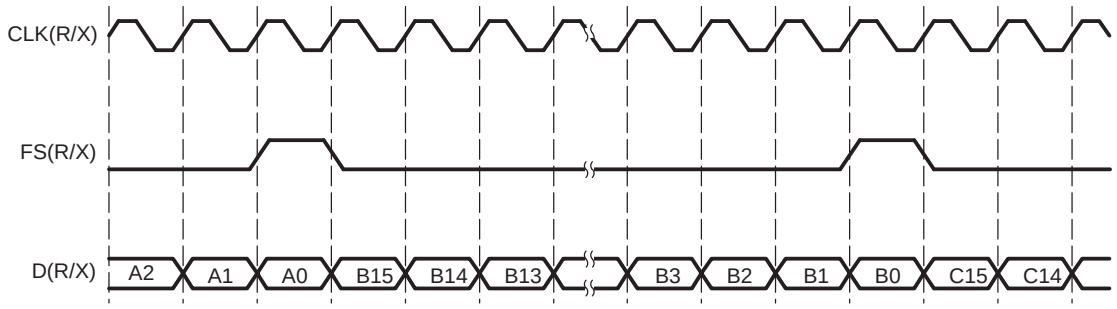
$$\text{Frame frequency} = \frac{\text{Bit-clock frequency}}{\text{Number of bit clocks between frame sync signals}}$$

The frame frequency may be increased by decreasing the time between frame synchronization signals in bit clocks (which is limited only by the number of bits per frame). As the frame transmit frequency is increased, the inactivity period between the data frames for adjacent transfers decreases to 0. The minimum time between frame synchronization pulses is the number of bits transferred per frame. This time also defines the maximum frame frequency, which is calculated by the following equation:

$$\text{Maximum frame frequency} = \frac{\text{Bit-clock frequency}}{\text{Number of bits per frame}}$$

Figure 18 shows the McBSP operating at maximum frame frequency. The data bits in consecutive frames are transmitted continuously with no inactivity between bits. If there is a 1-bit data delay, as shown, the frame synchronization pulse overlaps the last bit transmitted in the previous frame.

Figure 18. Maximum Frame Frequency for Transmit and Receive



Note: For (R/X)DATDLY = 0, the first bit of data transmitted is asynchronous to CLKX, as shown in Figure 13.

Maximum frame frequency may not be possible when the word length is 8-bits, depending on the clock divide value CLKGDV. The CPU or EDMA may not be able to service serial port requests that occur as frequently as every 8 bit clocks. This situation can be resolved by allowing additional space between words or choosing a slower bit clock (larger CLKGDV value).

2.6.4 Frame Synchronization Ignore

The McBSP can be configured to ignore transmit and receive frame synchronization pulses. The (R/X)FIG bit in (R/X)CR can be cleared to 0 to recognize frame sync pulses, or it can be set to 1 to ignore frame sync pulses. In this way, you can use (R/X)FIG either to pack data, if operating at maximum frame frequency, or to ignore unexpected frame sync pulses.

2.6.4.1 Frame Sync Ignore and Unexpected Frame Sync Pulses

RFIG and XFIG are used to ignore unexpected internal or external frame sync pulses. Any frame sync pulse is considered unexpected if it occurs one or more bit clocks earlier than the programmed data delay from the end of the previous frame specified by ((R/X)DATDLY). Setting the frame ignore bits to 1 causes the serial port to ignore these unexpected frame sync signals.

In reception, if not ignored (RFIG = 0), an unexpected FSR pulse discards the contents of RSR in favor of the incoming data. Therefore, if RFIG = 0, an unexpected frame synchronization pulse aborts the current data transfer, sets RSYNCERR in SPCR to 1, and begins the reception of a new data element. When RFIG = 1, the unexpected frame sync pulses are ignored.

In transmission, if not ignored (XFIG = 0), an unexpected FSX pulse aborts the ongoing transmission, sets the XSYNCERR bit in SPCR to 1, and reinitiates transmission of the current element that was aborted. When XFIG = 1, unexpected frame sync signals are ignored.

Figure 19 shows that element B is interrupted by an unexpected frame sync pulse when (R/X)FIG = 0. The reception of B is aborted (B is lost), and a new data element (C) is received after the appropriate data delay. This condition causes a receive synchronization error and thus sets the RSYNCERR bit. However, for transmission, the transmission of B is aborted and the same data (B) is retransmitted after the appropriate data delay. This condition is a transmit synchronization error and thus sets the XSYNCERR bit. Synchronization errors are discussed in Section 2.6.5.2 and Section 2.6.5.5.

Figure 19. Unexpected Frame Synchronization With (R/X)FIG = 0

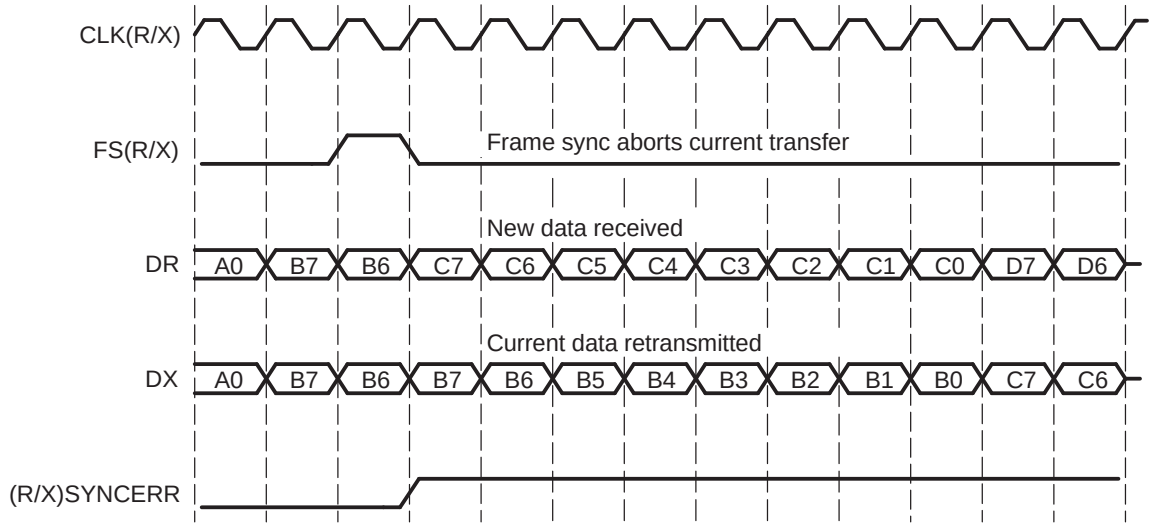
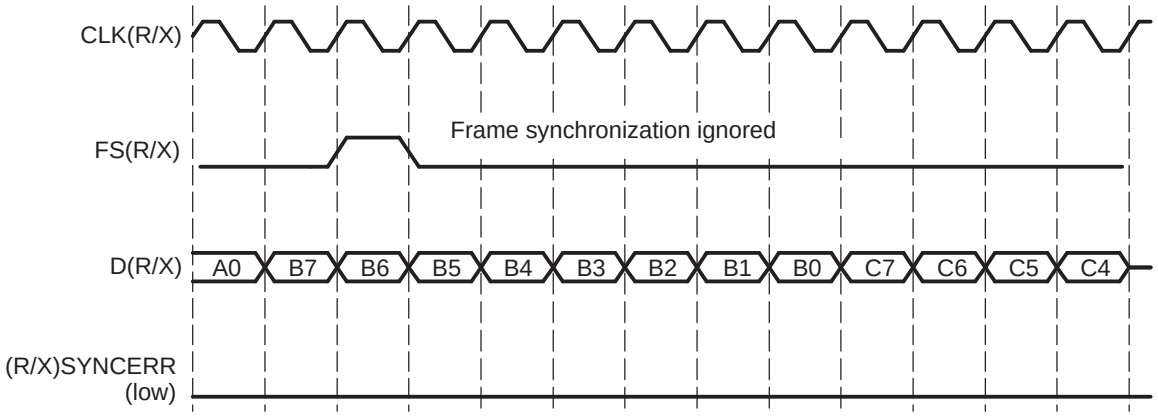


Figure 20 shows McBSP operation when unexpected internal or external frame synchronization signals are ignored by setting (R/X)FIG = 1. Here, the transfer of element B is not affected by an unexpected frame synchronization.

Figure 20. Unexpected Frame Synchronization With (R/X)FIG = 1



2.6.4.2 Data Packing using Frame Sync Ignore Bits

Section 2.5.5.4 describes one method of changing the element length and frame length to simulate 32-bit serial element transfers, thus requiring much less bus bandwidth than four 8-bit transfers require. This example works when there are multiple elements per frame.

Now consider the case of the McBSP operating at maximum packet frequency, as shown in Figure 21. Here, each frame has only a single 8-bit element. This stream takes one read transfer and one write transfer for each 8-bit element. Figure 22 shows the McBSP configured to treat this stream as a continuous stream of 32-bit elements. In this example, (R/X)FIG is set to 1 to ignore unexpected subsequent frames. Only one read transfer and one write transfer is needed every 32-bits. This configuration effectively reduces the required bus bandwidth to one-fourth of the bandwidth needed to transfer four 8-bit blocks.

Figure 21. Maximum Frame Frequency Operation With 8-Bit Data

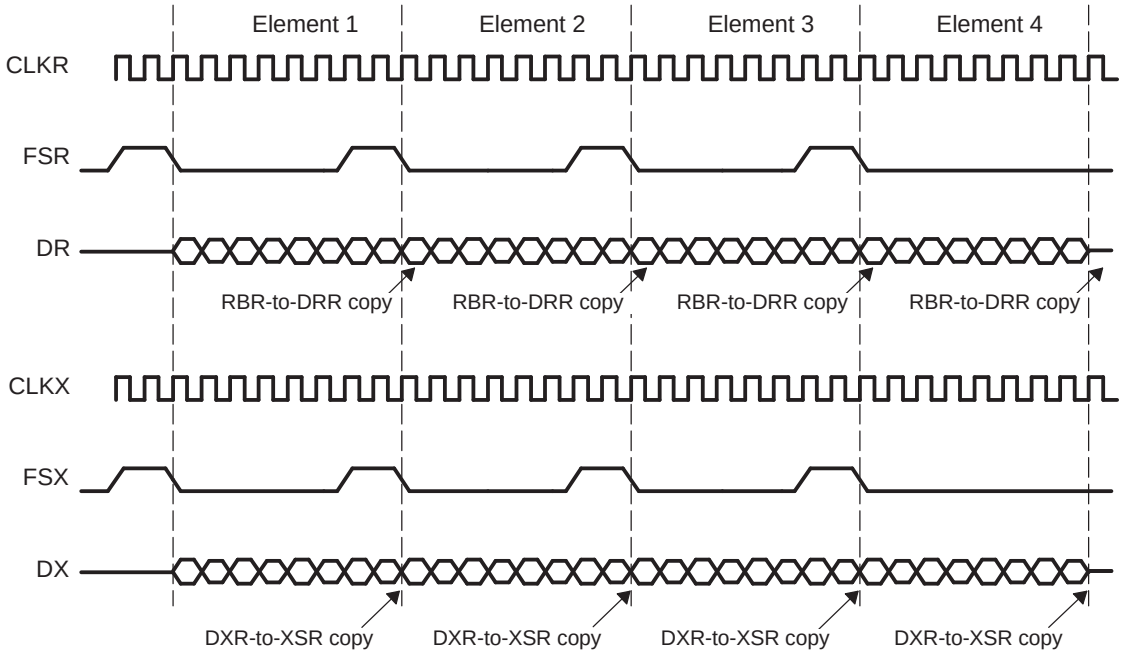
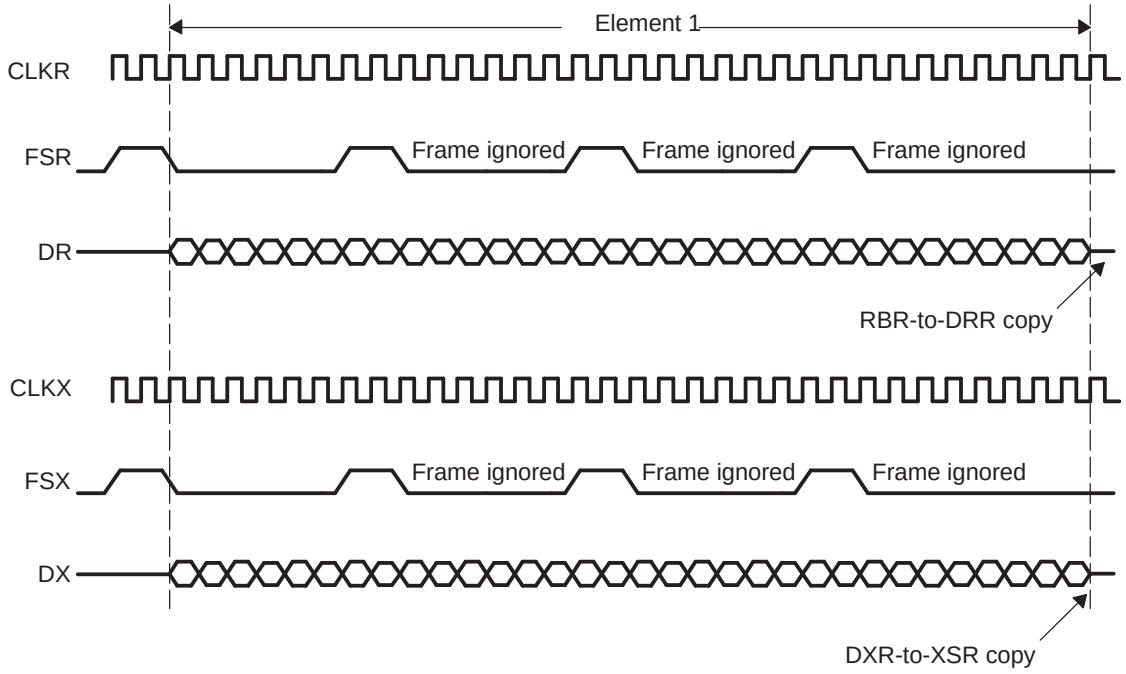


Figure 22. Data Packing at Maximum Frame Frequency With (R/X)FIG = 1



2.6.5 Serial Port Exception Conditions

There are five serial port events that can constitute a system error:

- Receive overrun (RFULL = 1 in SPCR)
- Unexpected receive frame synchronization (RSYNCERR = 1 in SPCR)
- Transmit data overwrite
- Transmit empty (XEMPTY = 0 in SPCR)
- Unexpected transmit frame synchronization (XSYNCERR = 1 in SPCR)

2.6.5.1 Receive Overrun: RFULL

RFULL = 1 in the serial port control register (SPCR) indicates that the receiver has experienced overrun and is in an error condition. RFULL is set when the following conditions are met:

- DRR has not been read since the last RBR-to-DRR transfer.
- RBR is full and an RBR-to-DRR copy has not occurred.
- RSR is full and an RSR-to-RBR transfer has not occurred.

The data arriving on DR is continuously shifted into RSR (Figure 23). Once a complete element is shifted into RSR, an RSR-to-RBR transfer can occur only if an RBR-to-DRR copy is complete. Therefore, if DRR has not been read by the CPU or the EDMA controller since the last RBR-to-DRR transfer (RRDY = 1), an RBR-to-DRR copy does not take place until RRDY = 0. This prevents an RSR-to-RBR copy. New data arriving on the DR pin is shifted into RSR, and the previous contents of RSR are lost. After the receiver starts running from reset, a minimum of three elements must be received before RFULL can be set, because there was no last RBR-to-DRR transfer before the first element.

This data loss can be avoided if DRR is read no later than two and a half CLKR cycles before the end of the third element (data C) in RSR, as shown in Figure 24.

Either of the following events clears the RFULL bit to 0 and allows subsequent transfers to be read properly:

- Reading DRR
- Resetting the receiver (RRST = 0) or the device

Another frame synchronization is required to restart the receiver.

Figure 23 shows the receive overrun condition. Because element A is not read before the reception of element B is complete, B is not transferred to DRR yet. Another element, C, arrives and fills RSR. DRR is finally read, but not earlier than two and one half cycles before the end of element C. New data D overwrites the previous element C in RSR. If RFULL is still set after the DRR is read, the next element can overwrite D if DRR is not read in time.

Figure 24 shows the case in which RFULL is set but the overrun condition is averted by reading the contents of DRR at least two and a half cycles before the next element, C, is completely shifted into RSR. This ensures that a RBR-to-DRR copy of data B occurs before the next element is transferred from RSR to RBR.

Figure 23. Serial Port Receive Overrun

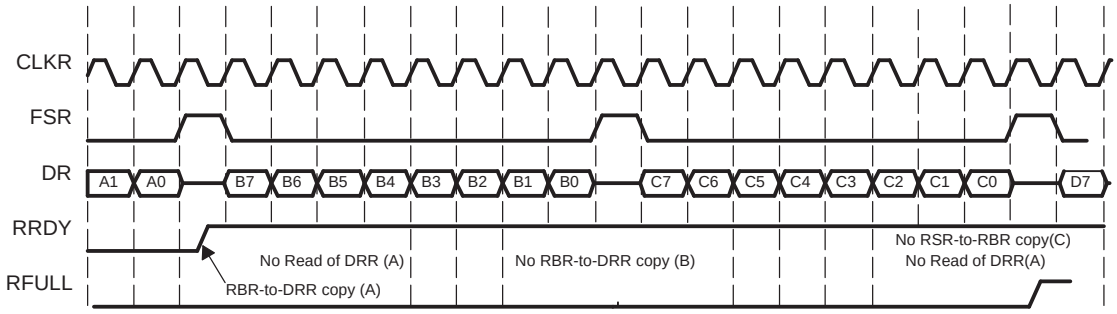
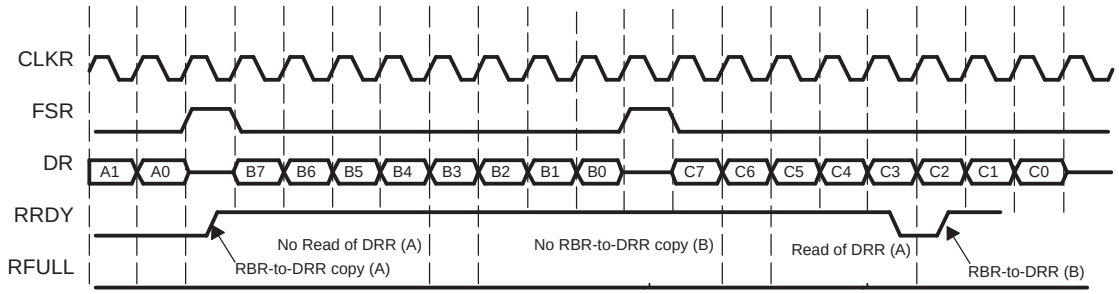


Figure 24. Serial Port Receive Overrun Avoided



2.6.5.2 Unexpected Receive Frame Synchronization: RSYNCERR

Figure 25 shows the decision tree that the receiver uses to handle all incoming frame synchronization pulses. The diagram assumes that the receiver has been activated (RRST = 1). Unexpected frame sync pulses can originate from an external source or from the internal sample rate generator. An unexpected frame sync pulse is defined as a sync pulse which occurs RDATDLY bit clocks earlier than the last transmitted bit of the previous frame. Any one of three cases can occur:

- Case 1: Unexpected FSR pulses with RFIG = 1. This case is discussed in Section 2.6.4.1 and shown in Figure 20. Here, receive frame sync pulses are ignored and the reception continues.
- Case 2: Normal serial port reception. There are three reasons for a receive not to be in progress:
  - This FSR is the first after RRST = 1.
  - This FSR is the first after DRR has been read clearing an RFULL condition.
  - The serial port is in the inter-packet intervals. The programmed data delay (RDATDLY) for reception may start during these inter-packet intervals for the first bit of the next element to be received. Thus, at maximum frame frequency, frame synchronization can still be received RDATDLY bit clocks before the first bit of the associated element.

    For this case, reception continues normally, because these are not unexpected frame sync pulses.
- Case 3: Unexpected receive frame synchronization with RFIG = 0 (unexpected frame not ignored). This case was shown in Figure 19 for maximum packet frequency. Figure 26 shows this case during normal operation of the serial port with time intervals between packets. Unexpected frame sync pulses are detected when they occur the value in RDATDLY bit clocks before the last bit of the previous element is received on DR. In both cases, RSYNCERR in SPCR is set. RSYNCERR can be cleared only by receiver reset or by writing a 0 to this bit in SPCR. If RINTM = 11b in SPCR, RSYNCERR drives the receive interrupt (RINT) to the CPU.

Note: Note that the RSYNCERR bit in SPCR is a read/write bit, so writing a 1 to it sets the error condition. Typically, writing a 0 is expected.

Figure 25. Decision Tree Response to Receive Frame Synchronization Pulse

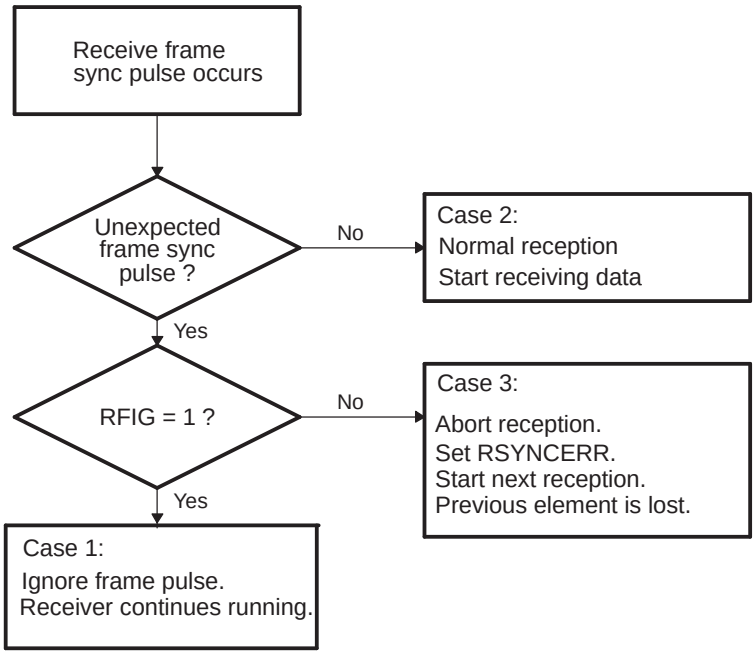
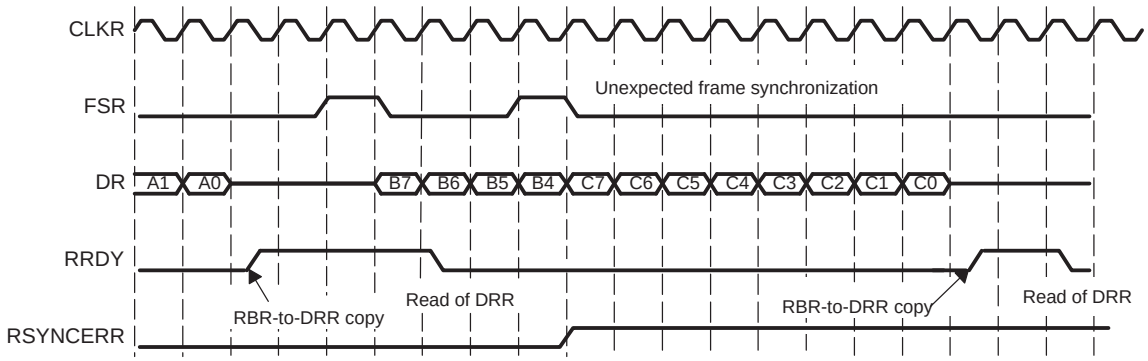


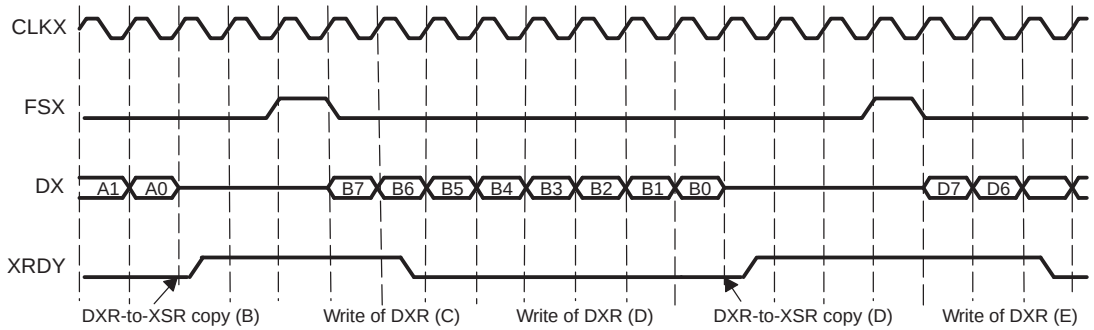
Figure 26. Unexpected Receive Frame Synchronization Pulse



2.6.5.3 Transmit With Data Overwrite

Figure 27 shows what happens if the data in DXR is overwritten before it is transmitted. Suppose you load the DXR with data C. A subsequent write to the DXR overwrites C with D before C is copied to the XSR. Thus, C is never transmitted on DX. The CPU can avoid overwriting data by polling XRDY before writing to DXR or by waiting for a programmed XINT to be triggered by XRDY (XINTM = 00b). The EDMA controller can avoid overwriting by synchronizing data writes with XEVT. See also Section 2.12.1.3.

Figure 27. Transmit With Data Overwrite



2.6.5.4 Transmit Empty: XEMPTY

XEMPTY indicates whether the transmitter has experienced underflow. Either of the following conditions causes XEMPTY to become active (XEMPTY = 0):

- During transmission, DXR has not been loaded since the last DXR-to-XSR copy, and all bits of the data element in XSR have been shifted out on DX.
- The transmitter is reset (XRST = 0 or the device is reset), and then restarted.

During underflow condition, the transmitter continues to transmit the old data in DXR for every new frame sync signal FSX (generated by an external device, or by the internal sample rate generator) until a new element is loaded into DXR by the CPU or the EDMA controller. XEMPTY is deactivated (XEMPTY = 1) when this new element in DXR is transferred to XSR. In the case when the FSX is generated by a DXR-to-XSR copy (FSXM = 1 in PCR and FSGM = 0 in SRGR), the McBSP does not generate any new frame sync until new data is written to the DXR and a DXR-to-XSR copy occurs.

When the transmitter is taken out of reset (XRST = 1), it is in a transmit ready (XRDY = 1) and transmit empty (XEMPTY = 0) condition. If DXR is loaded by the CPU or the EDMA controller before FSX goes active, a valid DXR-to-XSR transfer occurs. This allows for the first element of the first frame to be valid even before the transmit frame sync pulse is generated or detected. Alternatively, if a transmit frame sync is detected before DXR is loaded, 0s are output on DX.

Figure 28 shows a transmit underflow condition. After B is transmitted, B is retransmitted on DX if you fail to reload the DXR before the subsequent frame synchronization. Figure 29 shows the case of writing to DXR just before a transmit underflow condition that would otherwise occur. After B is transmitted, C is written to DXR before the next transmit frame sync pulse occurs.

Figure 28. Transmit Empty

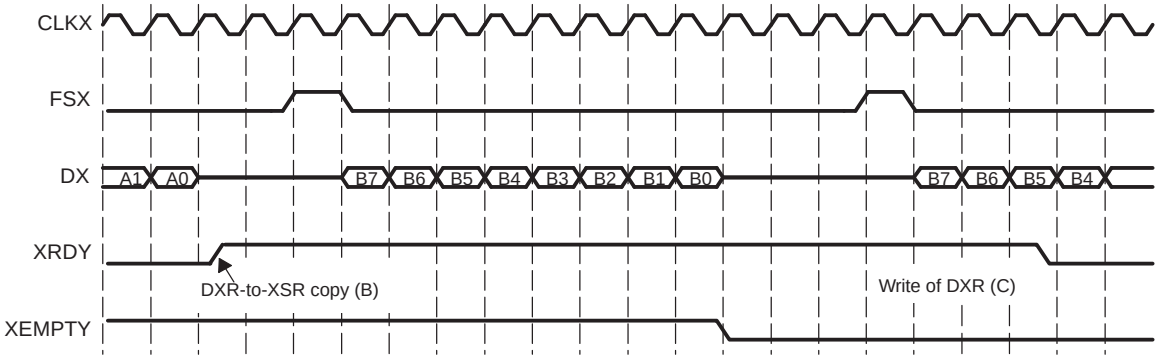
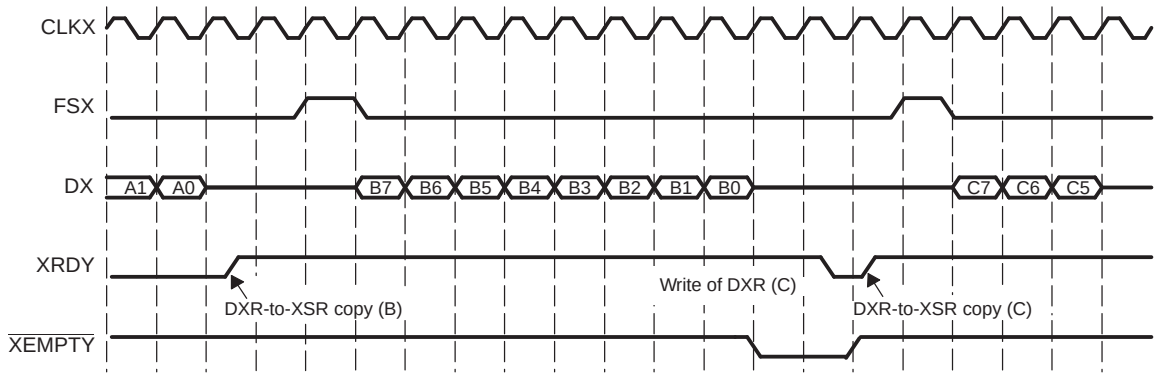


Figure 29. Transmit Empty Avoided



2.6.5.5 Unexpected Transmit Frame Synchronization: XSYNCERR

A transmit frame sync error (XSYNCERR) may occur the first time the transmitter is enabled (XRST = 1) after a device reset. To avoid this, after enabling the transmitter for the first time, the following procedure must be followed:

1. Wait for two CLKG cycles. The unexpected frame sync error (XSYNCERR), if any, occurs within this time period.
2. Disable the transmitter (XRST = 0). This clears any XSYNCERR.
3. Re-enable the transmitter (XRST = 1).

See also Section 2.11 for details on initialization procedure.

Figure 30 shows the decision tree that the transmitter uses to handle all incoming frame synchronization signals. The diagram assumes that the transmitter has been started (XRST = 1). An unexpected transmit frame sync pulse is defined as a sync pulse that occurs XDATDLY bit clocks earlier than the last transmitted bit of the previous frame. Any one of three cases can occur:

- Case 1: Unexpected FSX pulses with XFIG = 1. This case is discussed in Section 2.6.4.1 and shown in Figure 20. In this case, unexpected FSX pulses are ignored, and the transmission continues.
- Case 2: FSX pulses with normal serial port transmission. This situation is discussed in Section 2.6.2. There are two possible reasons for a transmit not to be in progress:
  - This FSX pulse is the first one to occur after XRST = 1.
  - The serial port is in the interpacket intervals. The programmed data delay (XDATDLY) may start during these interpacket intervals before the first bit of the next element is transmitted. Thus, if operating at maximum packet frequency, frame synchronization can still be received XDATDLY bit clocks before the first bit of the associated element.
- Case 3: Unexpected transmit frame synchronization with XFIG = 0. The case was shown in Figure 19 for frame synchronization with XFIG = 0 at maximum packet frequency. Figure 31 shows the case for normal operation of the serial port with interpacket intervals. In both cases, XSYNCERR in SPCR is set. XSYNCERR can be cleared only by transmitter reset or by writing a 0 to this bit in SPCR. If XINTM = 11b in SPCR, XSYNCERR drives the receive interrupt (XINT) to the CPU.

Note: The XSYNCERR bit in SPCR is a read/write bit, so writing a 1 to it sets the error condition. Typically, writing a 0 is expected.

Figure 30. Decision Tree Response to Transmit Frame Synchronization Pulse

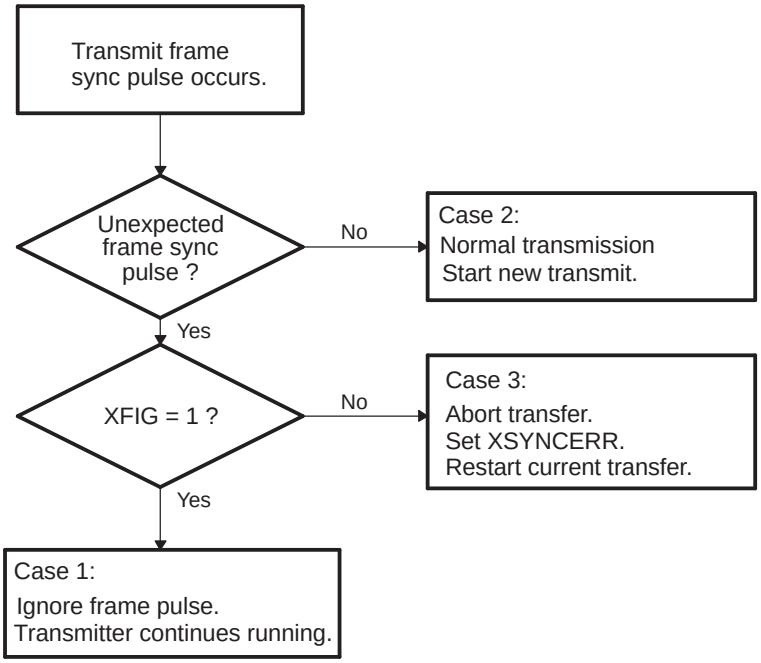
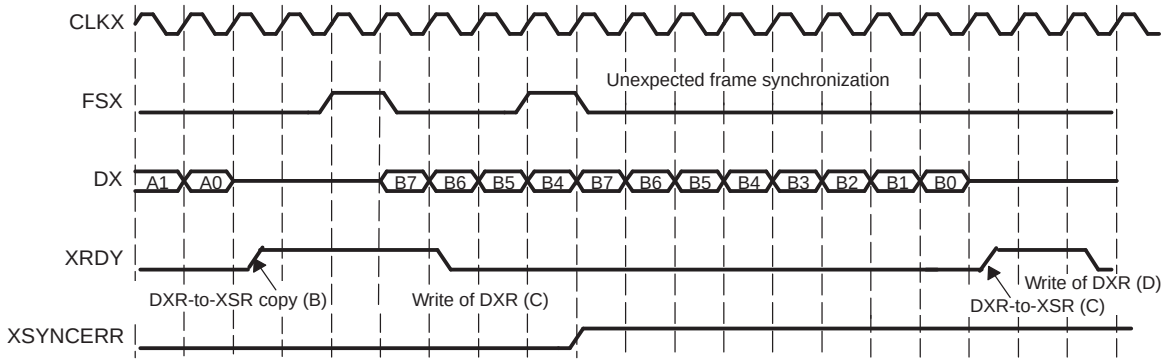


Figure 31. Unexpected Transmit Frame Synchronization Pulse



2.7 µ-Law/A-Law Companding Hardware Operation

Companding (compressing and expanding) hardware allows compression and expansion of data in either µ-law or A-law format. The specification for µ-law and A-law log PCM is part of the CCITT G.711 recommendation. The companding standard employed in the United States and Japan is µ-law and allows 14 bits of dynamic range. The European companding standard is A-law and allows 13 bits of dynamic range. Any values outside these ranges are set to the most positive or most negative value. Thus, for companding to work best here, the data transferred to and from the McBSP via the CPU or the EDMA controller must be at least 16 bits wide.

The µ-law and A-law formats encode data into 8-bit code elements. Companded data is always 8-bits-wide, so the appropriate (R/X)WDLEN1/2 must be cleared to 0, indicating an 8-bit serial data stream. If companding is enabled and either phase of the frame does not have an 8-bit element length, companding continues as if the element length is eight bits.

When companding is used, transmit data is encoded according to the specified companding law, and receive data is decoded to 2s-complement format. Companding is enabled and the desired format is selected by appropriately setting (R/X)COMPAND in the (R/X)CR. Compression occurs during the process of copying data from DXR to XSR and expansion occurs from RBR to DRR, as shown in Figure 32.

For transmit data to be compressed, it should be 16-bit, left-justified data, such as LAW16, as shown in Figure 33. The value can be either 13 or 14 bits wide, depending on the companding law. This 16-bit data is aligned in DXR, as shown in Figure 34.

For reception, the 8-bit compressed data in RBR is expanded to a left-justified 16-bit data, LAW16. This can be further justified to 32-bit data by programming the RJUST bits in the serial port control register (SPCR), as shown in Table 12.

Figure 32. Companding Flow

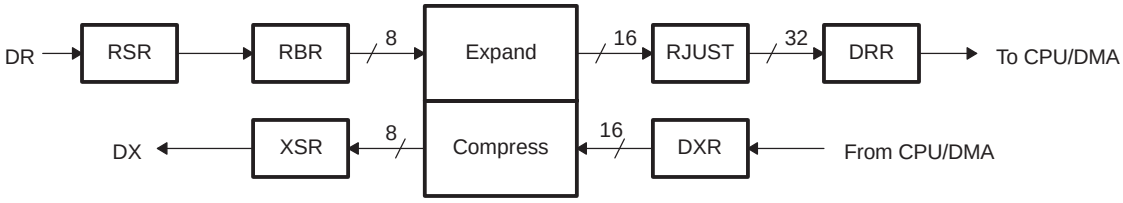


Figure 33. Companding Data Formats

LAW16	15	2	1	0
μ-Law	Value		0	0

LAW16	15	3	2	1	0
A-Law	Value		0	0	0

Figure 34. Transmit Data Companding Format in DXR

31	16	15	0
Don't care		LAW16	

Table 12. Justification of Expanded Data in DRR

	DRR Bits			
RJUST Bit in SPCR	**31**	**16**	**15**	**0**
00	0		LAW16	
01	sign		LAW16	
10	LAW16		0	
11	Reserved			

2.7.1 Companding Internal Data

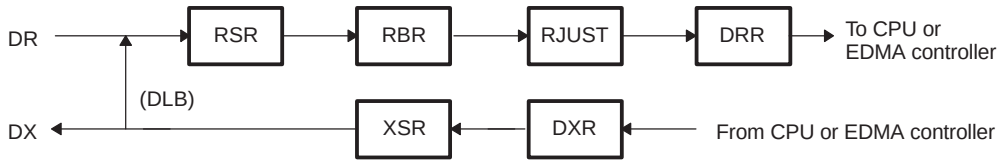
If the McBSP is unused, the companding hardware can compand internal data. This hardware can be used to:

- Convert linear data to the appropriate μ-law or A-law format.
- Convert μ-law or A-law data to the linear format.
- Observe the quantization effects in companding by transmitting linear data and compressing and re-expanding this data. This is useful only if both XCOMPAND and RCOMPAND enable the same companding format.

Figure 35 shows two methods by which the McBSP can compand internal data. Data paths for these two methods are indicated by (DLB) and (non-DLB) arrows.

- **Non-DLB:** When both the transmit and receive sections of the serial port are reset, DRR and DXR are internally connected through the companding logic. Values from DXR are compressed as determined by the XCOMPAND bits and then expanded as determined by the RCOMPAND bits. RRDY and XRDY bits are not set. However, data is available in DRR four internal McBSP clocks after being written to DXR. The advantage of this method is its speed. The disadvantage is that there is no synchronization available to the CPU and the EDMA controller to control the flow of data.
- **DLB:** The McBSP is enabled in digital loopback (DLB) mode with companding appropriately enabled by the RCOMPAND and XCOMPAND bits. Receive and transmit interrupts (RINT when RINTM = 0 and XINT when XINTM = 0) or synchronization events (REVT and XEVT) allow synchronization of the CPU or the EDMA controller to these conversions, respectively. Here, the time for this companding depends on the serial bit rate selected.

Figure 35. Companding of Internal Data



2.7.2 Bit Ordering

Normally, all transfers on the McBSP are sent and received with the MSB first. However, certain 8-bit data protocols (that do not use companded data) require the LSB to be transferred first. By setting the (R/X)COMPAND = 01b in (R/X)CR, the bit ordering of 8-bit elements is reversed (LSB first) before being sent to the serial port. Like the companding feature, this feature is enabled only if the appropriate (R/X)WDLEN1/2 bit is cleared to 0, indicating that 8-bit elements are to be serially transferred. A 32-bit bit reversal feature is also available, see Section 2.5.5.7

2.8 Multichannel Selection Modes

This section defines and provides the functions and all related information concerning the multichannel selection modes.

2.8.1 Channels, Blocks, and Partitions

A McBSP channel is a time slot for shifting in/out the bits of one serial word. Each McBSP supports up to 128 channels for reception and 128 channels for transmission. In the receiver and in the transmitter, the 128 available channels are divided into eight blocks that each contain 16 contiguous channels:

Block 0: Channels 0 through 15
Block 1: Channels 16 through 31
Block 2: Channels 32 through 47
Block 3: Channels 48 through 63
Block 4: Channels 64 through 79
Block 5: Channels 80 through 95
Block 6: Channels 96 through 111
Block 7: Channels 112 through 127

The blocks are assigned to partitions according to the selected partition mode. In the 2-partition mode, you assign one even-numbered block (0, 2, 4, or 6) to partition A and one odd-numbered block (1, 3, 5, or 7) to partition B. In the 8-partition mode, blocks 0 through 7 are automatically assigned to partitions, A through H, respectively.

The number of partitions for reception and the number of partitions for transmission are independent. For example, it is possible to use 2 receive partitions (A and B) and 8 transmit partitions (A through H).

2.8.2 Multichannel Selection

When a McBSP uses a time-division multiplexed (TDM) data stream while communicating with other McBSPs or serial devices, the McBSP may need to receive and/or transmit on only a few channels. To save memory and bus bandwidth, you can use a multichannel selection mode to prevent data flow in some of the channels. The McBSP has one receive multichannel selection mode and three transmit multichannel selection modes.

Each channel partition has a dedicated channel enable register. If the appropriate multichannel selection mode is on, each bit in the register controls whether data flow is allowed or prevented in one of the channels that is assigned to that partition.

2.8.3 Configuring a Frame for Multichannel Selection

Before you enable a multichannel selection mode, make sure you properly configure the data frame:

- Select a single-phase frame (RPHASE/XPHASE = 0). Each frame represents a TDM data stream.
- Set a frame length (RFRLEN1/XFRLEN1) that includes the highest-numbered channel that is to be used. For example, if you plan to use channels 0, 15, and 39 for reception, the receive frame length must be at least 40 (RFRLEN1 = 39). If XFRLEN1 = 39 in this case, the receiver creates 40 time slots per frame but only receives data during time slots 0, 15, and 39 of each frame.

Note: The frame-sync pulse can be generated internally by the sample rate generator or it can be supplied externally by another source. In a multichannel mode configuration with external frame-sync generation, the McBSP transmitter will ignore the first frame-sync pulse after it is taken out of reset. The transmitter will transmit only on the second frame-sync pulse. The receiver will shift in data on the first frame-sync pulse, regardless of whether it is generated internally or externally.

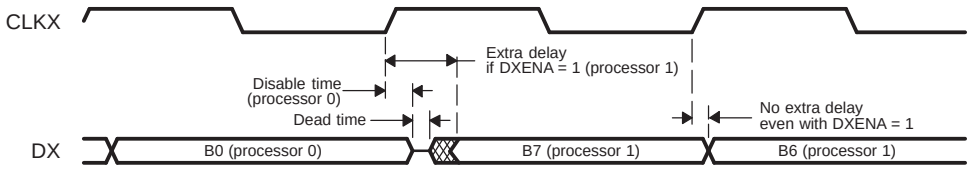
2.8.4 DX Enabler: DXENA

The DXENA bit in the serial port control register (SPCR) controls the high-impedance enable on the DX pin. When DXENA = 1, the McBSP enables extra delay for the DX pin turn-on time. This feature is useful for McBSP multichannel operations, such as in a time-division multiplexed (TDM) system. The McBSP supports up to 128 channels in a multichannel operation. These channels can be driven by different devices in a TDM data communication line, such as the T1/E1 line. In any multichannel operation where multiple devices transmit over the same DX line, you need to ensure that no two devices transmit data simultaneously, which results in bus contention. Enough dead time should exist between the transmission of the first data bit of the current device and the transmission of the last data bit of the previous device. In other words, the last data bit of the previous device needs to be disabled to a high-impedance state before the next device begins transmitting data to the same data line, as shown in Figure 36.

When two McBSPs are used to transmit data over the same TDM line, bus contention occurs if DXENA = 0. The first McBSP turns off the transmission of the last data bit (changes DX from valid to a high-impedance state) after a disable time specified in the data manual. As shown in Figure 36, this disable time is measured from the CLKX active clock edge. The next McBSP turns on its DX pin (changes from a high-impedance state to valid) after a delay time. Again, this delay time is measured from the CLKX active clock edge. Bus contention occurs because the dead time between the two devices is not enough. You need to apply alternative software or hardware methods to ensure proper multichannel operation in this case.

If you set DXENA = 1 in the second McBSP, the second McBSP turns on its DX pin after some extra delay time. This ensures that the previous McBSP on the same DX line is disabled before the second McBSP starts driving out data. The DX enabler controls only the high-impedance enable on the DX pin, not the data itself. Data is shifted out to the DX pin at the same time as in the case when DXENA = 0. The only difference is that with DXENA = 1, the DX pin is masked to a high-impedance state for some extra CPU cycles before the data is seen on the TDM data line. Therefore, only the first bit of data is delayed. Refer to the specific device datasheet for the exact amount of delay.

Figure 36. DX Timing for Multichannel Operation



2.8.5 Using Two Partitions

For multichannel selection operation in the receiver and/or the transmitter, you can use two partitions or eight partitions. If you choose the 2-partition mode (RMCME = 0 for reception, XMCME = 0 for transmission), McBSP channels are activated using an alternating scheme. In response to a frame-sync pulse, the receiver or transmitter begins with the channels in partition A and then alternates between partitions B and A until the complete frame has been transferred. When the next frame-sync pulse occurs, the next frame is transferred, beginning with the channels in partition A.

2.8.5.1 Assigning Blocks to Partitions A and B

For reception, any two of the eight receive-channel blocks can be assigned to receive partitions A and B (see Table 13), which means up to 32 receive channels can be enabled at any given point in time. Similarly, any two of the eight transmit-channel blocks (up 32 enabled transmit channels) can be assigned to transmit partitions A and B (see Table 14). You can dynamically change which blocks of channels are assigned to the partitions, see Section 2.8.5.2.

For reception:

- Assign an even-numbered channel block (0, 2, 4, or 6) to receive partition A by writing to the RPABLK bit in the multichannel control register (MCR). In the receive multichannel selection mode, the channels in this partition are controlled by the enhanced receive channel enable register partition A/B (RCERE0).
- Assign an odd-numbered block (1, 3, 5, or 7) to receive partition B with the RPBBLK bit in MCR. In the receive multichannel selection mode, the channels in this partition are controlled by the enhanced receive channel enable register partition A/B (RCERE0).

For transmission:

- Assign an even-numbered channel block (0, 2, 4, or 6) to transmit partition A by writing to the XPABLK bit in the multichannel control register (MCR). In one of the transmit multichannel selection modes, the channels in this partition are controlled by the enhanced transmit channel enable register partition A/B (XCERE0).
- Assign an odd-numbered block (1, 3, 5, or 7) to transmit partition B with the XPBBLK bit in MCR. In one of the transmit multichannel selection modes, the channels in this partition are controlled by the enhanced transmit channel enable register partition A/B (XCERE0).

Table 13. Receive Channel Assignment and Control When Two Receive Partitions are Used

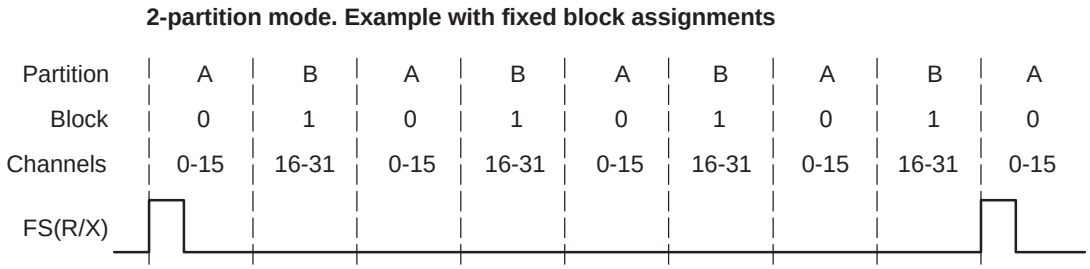
Receive Partition	Assigned Block of Receive Channels	RPABLK Bit in MCR	RPBBLK Bit in MCR	RCERE0 Bits
A	Block 0: channels 0 through 15	0	–	RCE0-RCE15
A	Block 2: channels 32 through 47	1h	–	RCE0-RCE15
A	Block 4: channels 64 through 79	2h	–	RCE0-RCE15
A	Block 6: channels 96 through 111	3h	–	RCE0-RCE15
B	Block 1: channels 16 through 31	–	0	RCE16-RCE31
B	Block 3: channels 48 through 63	–	1h	RCE16-RCE31
B	Block 5: channels 80 through 95	–	2h	RCE16-RCE31
B	Block 7: channels 112 through 127	–	3h	RCE16-RCE31

Table 14. Transmit Channel Assignment and Control When Two Transmit Partitions are Used

Transmit Partition	Assigned Block of Transmit Channels	XPABLK Bit in MCR	XPBBLK Bit in MCR	XCERE0 Bits
A	Block 0: channels 0 through 15	0	–	XCE0-XCE15
A	Block 2: channels 32 through 47	1h	–	XCE0-XCE15
A	Block 4: channels 64 through 79	2h	–	XCE0-XCE15
A	Block 6: channels 96 through 111	3h	–	XCE0-XCE15
B	Block 1: channels 16 through 31	–	0	XCE16-XCE31
B	Block 3: channels 48 through 63	–	1h	XCE16-XCE31
B	Block 5: channels 80 through 95	–	2h	XCE16-XCE31
B	Block 7: channels 112 through 127	–	3h	XCE16-XCE31

Figure 37 shows an example of alternating between the channels of partition A and the channels of partition B. Channels 0-15 have been assigned to partition A, and channels 16-31 have been assigned to partition B. In response to a frame-sync pulse, the McBSP begins a frame transfer with partition A and then alternates between partitions B and A until the complete frame is transferred.

Figure 37. Alternating Between the Channels of Partition A and the Channels of Partition B



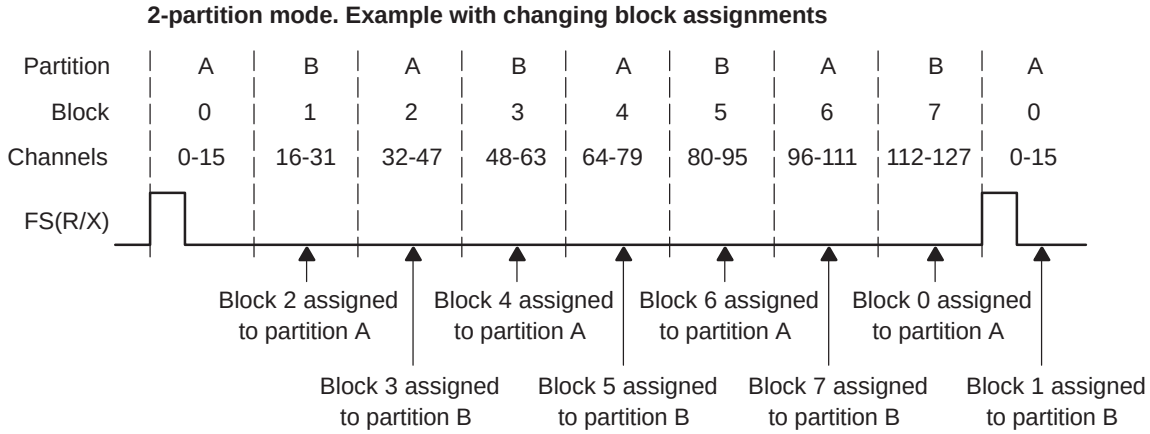
2.8.5.2 Reassigning Blocks During Reception/Transmission

If you want to use more than 32 channels, you can change which channel blocks are assigned to partitions A and B during the course of a data transfer. However, these changes must be carefully timed. While a partition is being transferred, it's the associated block assignment bits cannot be modified, and its associated channel enable register cannot be modified. For example, if block 3 is being transferred and block 3 is assigned to partition A, you cannot modify (R/X)PABLK to assign different channels to partition A, and you cannot modify (R/X)CERE0 to change the channel configuration for partition A. Several features of the McBSP help you time the reassignment:

- The block of channels currently involved in reception/transmission (the current block) is reflected in the RCBLK/XCBLK bits. Your program can poll these bits to determine which partition is active. When a partition is not active, it is safe to change its block assignment and channel configuration.
- At the end of every block (at the boundary of two partitions), an interrupt can be sent to the CPU. In response to the interrupt, the CPU can then check the RCBLK/XCBLK bits and update the inactive partition.

Figure 38 shows an example of reassigning channels throughout a data transfer. In response to a frame-sync pulse, the McBSP alternates between partitions A and B. Whenever partition B is active, the CPU changes the block assignment for partition A. Whenever, partition A is active, the CPU changes the block assignment for partition B.

Figure 38. Reassigning Channel Blocks Throughout a McBSP Data Transfer



2.8.6 Using Eight Partitions

For multichannel selection operation in the receiver and/or the transmitter, you can use eight partitions or two partitions. If you choose the 8-partition mode (RMCME = 1 for reception, XMCME = 1 for transmission), McBSP partitions are activated in the following order: A, B, C, D, E, F, G, H. In response to a frame-sync pulse, the receiver or transmitter begins with the channels in partition A and then continues with the other partitions in order until the complete frame has been transferred. When the next frame-sync pulse occurs, the next frame is transferred, beginning with the channels in partition A. In the 8-partition mode, the (R/X)PABLK and (R/X)PBBLK bits are ignored and the 16-channel blocks are assigned to the partitions as shown in Table 15 and Table 16. These assignments cannot be changed. Table 15 and Table 16 also show the registers used to control the channels in the partitions.

Table 15. Receive Channel Assignment and Control When Eight Receive Partitions are Used

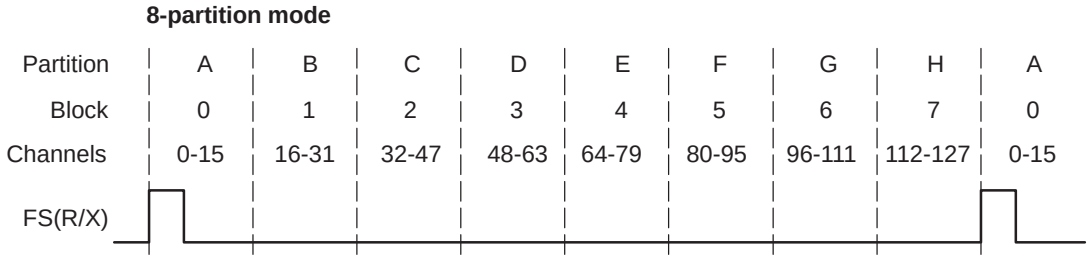
Receive Partition	Assigned Block of Receive Channels	Register Used For Channel Control
A	Block 0: channels 0 through 15	RCERE0
B	Block 1: channels 16 through 31	RCERE0
C	Block 2: channels 32 through 47	RCERE1
D	Block 3: channels 48 through 63	RCERE1
E	Block 4: channels 64 through 79	RCERE2
F	Block 5: channels 80 through 95	RCERE2
G	Block 6: channels 96 through 111	RCERE3
H	Block 7: channels 112 through 127	RCERE3

Table 16. Transmit Channel Assignment and Control When Eight Transmit Partitions are Used

Transmit Partition	Assigned Block of Transmit Channels	Register Used For Channel Control
A	Block 0: channels 0 through 15	XCERE0
B	Block 1: channels 16 through 31	XCERE0
C	Block 2: channels 32 through 47	XCERE1
D	Block 3: channels 48 through 63	XCERE1
E	Block 4: channels 64 through 79	XCERE2
F	Block 5: channels 80 through 95	XCERE2
G	Block 6: channels 96 through 111	XCERE3
H	Block 7: channels 112 through 127	XCERE3

Figure 39 shows an example of the McBSP using the 8-partition mode. In response to a frame-sync pulse, the McBSP begins a frame transfer with partition A and then activates B, C, D, E, F, G, and H to complete a 128-word frame.

Figure 39. McBSP Data Transfer in the 8-Partition Mode



2.8.7 Receive Multichannel Selection Mode

The RMCM bit in the multichannel control register (MCR) determines whether all channels or only selected channels are enabled for reception. When RMCM = 0, all 128 receive channels are enabled and cannot be disabled. When RMCM = 1, the receive multichannel selection mode is enabled. In this mode:

- Channels can be individually enabled or disabled. The only channels enabled are those selected in the appropriate enhanced receive channel enable register (RCERE*n*). The way channels are assigned to the RCERE*n* depends on the number of receive channel partitions (2 or 8), as defined by the RMCME bit in MCR.
- If a receive channel is disabled, any bits received in that channel are passed only as far as the receive buffer register (RBR). The receiver does not copy the content of the RBR to the DRR, and as a result, does not set the receiver ready bit (RRDY). Therefore, no DMA synchronization event (REVT) is generated, and if the receiver interrupt mode depends on RRDY (RINTM = 0), no interrupt is generated.

As an example of how the McBSP behaves in the receive multichannel selection mode, suppose you enable only channels 0, 15, and 39 and that the frame length is 40. The McBSP:

1. Accepts bits shifted in from the DR pin in channel 0.
2. Ignores bits received in channels 1-14.
3. Accepts bits shifted in from the DR pin in channel 15.
4. Ignores bits received in channels 16-38.
5. Accepts bits shifted in from the DR pin in channel 39.

2.8.8 Transmit Multichannel Selection Mode

The XMCM bit in the multichannel control register (MCR) determines whether all channels or only selected channels are enabled and unmasked for transmission. The McBSP has three transmit multichannel selection modes (XMCM = 1, XMCM = 2h, and XMCM = 3h), which are described in Table 17.

Table 17. Selecting a Transmit Multichannel Selection Mode With the XMCM Bits

XMCM Bit in MCR	Transmit Multichannel Selection Mode
0	No transmit multichannel selection mode is on. All channels are enabled and unmasked. No channels can be disabled or masked.
1h	All channels are disabled unless they are selected in the appropriate enhanced transmit channel enable register (XCERE*n*). If enabled, a channel in this mode is also unmasked. The XMCME bit in MCR determines whether 32 channels or 128 channels are selectable in XCERE*n*.
2h	All channels are enabled, but they are masked unless they are selected in the appropriate enhanced transmit channel enable register (XCERE*n*). The XMCME bit in MCR determines whether 32 channels or 128 channels are selectable in XCERE*n*.
3h	This mode is used for symmetric transmission and reception. All channels are disabled for transmission unless they are enabled for reception in the appropriate enhanced receive channel enable register (RCERE*n*). Once enabled, they are masked unless they are also selected in the appropriate enhanced transmit channel enable register (XCERE*n*). The XMCME bit in MCR determines whether 32 channels or 128 channels are selectable in RCERE*n* and XCERE*n*.

As an example of how the McBSP behaves in a transmit multichannel selection mode, suppose that XMCM = 1 (all channels disabled unless individually enabled) and that you have enabled only channels 0, 15, and 39. Suppose also that the frame length is 40. The McBSP:

1. Shifts data to the DX pin in channel 0.
2. Places the DX pin in the high-impedance state in channels 1–14.
3. Shifts data to the DX pin in channel 15.
4. Places the DX pin in the high-impedance state in channels 16–38.
5. Shifts data to the DX pin in channel 39.

2.8.8.1 Disabling/Enabling Versus Masking/Unmasking

For transmission, a channel can be:

- Enabled and unmasked (transmission can begin and can be completed).
- Enabled but masked (transmission can begin but cannot be completed).
- Disabled (transmission cannot occur).

The following definitions explain the channel control options:

Enabled channel	A channel that can begin transmission by passing data from the data transmit register (DXR) to the transmit shift register (XSR).
Masked channel	A channel that cannot complete transmission. The DX pin is held in the high-impedance state; data cannot be shifted out on the DX pin. In systems where symmetric transmit and receive provides software benefits, this feature allows transmit channels to be disabled on a shared serial bus. A similar feature is not needed for reception because multiple receptions cannot cause serial bus contention.
Disabled channel	A channel that is not enabled. A disabled channel is also masked. Because no DXR-to-XSR copy occurs, the XRDY bit in SPCR is not set. Therefore, no DMA synchronization event (XEVT) is generated, and if the transmit interrupt mode depends on XRDY (XINTM = 0 in SPCR), no interrupt is generated. The XEMPTY bit in SPCR is not affected.
Unmasked channel	A channel that is not masked. Data in the XSR is shifted out on the DX pin.

2.8.8.2 Activity on McBSP Pins for Different Values of XMCM

Figure 40 shows the activity on the McBSP pins for the various XMCM values. In all cases, the transmit frame is configured as follows:

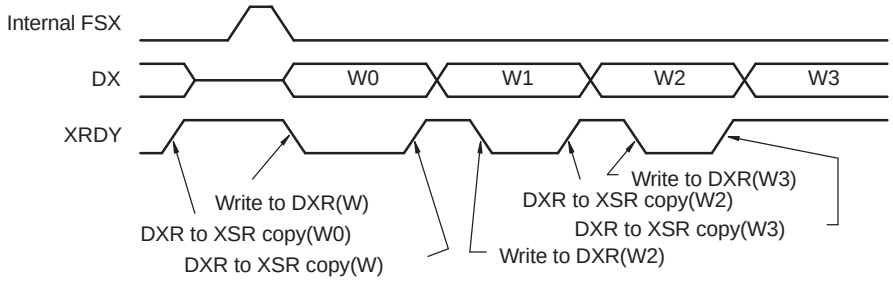
- In transmit control register (XCR):
  - XPHASE = 0: Single-phase frame (required for multichannel selection modes)
  - XFRLEN1 = 3h: 4 words per frame
  - XWDLEN1 = 0: 8 bits per word
- In multichannel control register (MCR):
  - XMCME = 0: 2-partition mode (only partitions A and B used)

In the case where XMCM = 3h, transmission and reception are symmetric, which means the corresponding bits for the receiver (RPHASE, RFRLEN1, RWDLEN1, and RMCME) must have the same values as XPHASE, XFRLEN1, and XWDLEN1, respectively.

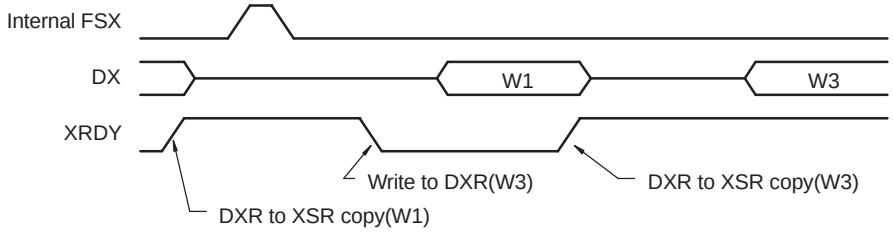
In Figure 40, the arrows showing where the various events occur are only sample indications. Wherever possible, there is a time window in which these events can occur.

Figure 40. Activity on McBSP Pins for the Possible Values of XMCM

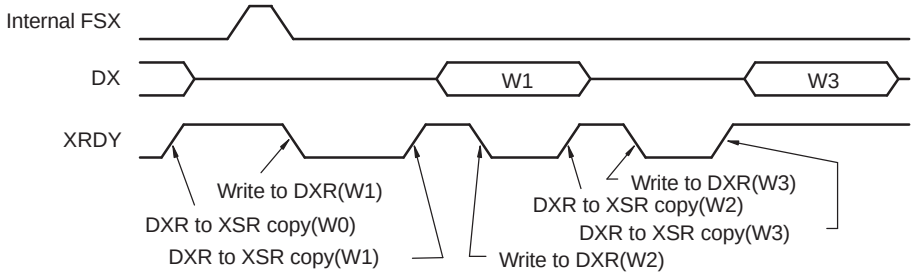
(a) XMCM = 0: All channels enabled and unmasked



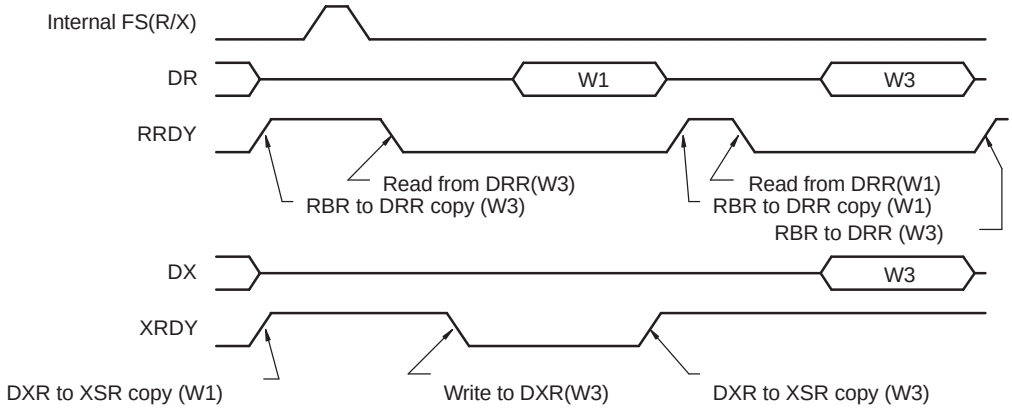
(b) XMCM = 1h, XPABLK = 0, XCERE0 = 0000 000Ah: Only channels 1 and 3 enabled and unmasked



(c) XMCM = 2h, XPABLK = 0, XCERE0 = 0000 000Ah: All channels enabled, only 1 and 3 unmasked



(d) XMCM = 3h, RPABLK = 0, XPABLK = x, RCERE0 = 0000 0008h, XCERE0 = 0000 000Ah: Receive channels: 1 and 3 enabled; transmit channels: 1 and 3 enabled, but only 3 unmasked



2.8.9 Using Interrupts Between Block Transfers

When a multichannel selection mode is used, an interrupt request can be sent to the CPU at the end of every 16-channel block (at the boundary between partitions and at the end of the frame). In the receive multichannel selection mode, a receive interrupt (RINT) request is generated at the end of each block transfer if the RINTM bit in the serial port control register (SPCR) is set to 1. In any of the transmit multichannel selection modes, a transmit interrupt (XINT) request is generated at the end of each block transfer if the XINTM bit in SPCR is set to 1. When RINTM/XINTM = 1, no interrupt is generated unless a multichannel selection mode is on.

These interrupt pulses are active high and last for 2 McBSP internal input clock cycles.

This type of interrupt is especially helpful if you are using the 2-partition mode and you want to know when you can assign a different block of channels to partition A or B.

2.9 SPI Operation Using the Clock Stop Mode

This section describes how the McBSP can communicate with one or more devices using the SPI protocol.

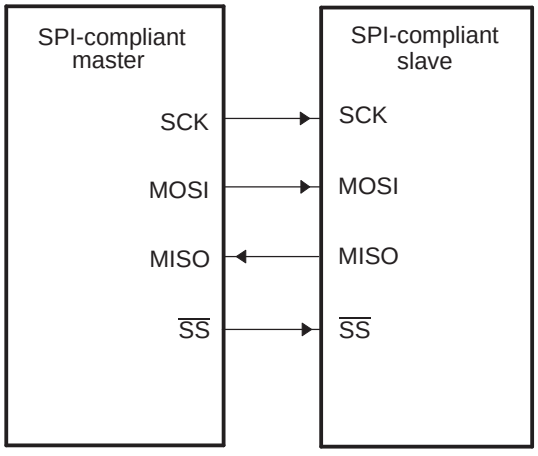
2.9.1 SPI Protocol

The SPI protocol is a master-slave configuration with one master device and one or more slave devices. The interface consists of the following four signals:

- Serial data input (also referred to as Master In—Slave Out, or MISO).
- Serial data output (also referred to as Master Out—Slave In, or MOSI).
- Shift-clock (also referred to as SCK).
- Slave-enable signal (also referred to as $\overline{SS}$).

A typical SPI interface with a single slave device is shown in Figure 41.

Figure 41. Typical SPI Interface



The master device controls the flow of communication by providing shift-clock and slave-enable signals. The slave-enable signal is an optional active-low signal that enables the serial data input and output of the slave device (the device not sending out the clock).

In the absence of a dedicated slave-enable ($\overline{SS}$) signal, communication between the master and slave is determined by the presence or absence of an active shift-clock. When the McBSP is operating in SPI master mode and the $\overline{SS}$ signal is not used by the slave SPI port, the slave device must remain enabled at all times, and multiple slaves cannot be used.

2.9.2 Clock Stop Mode

The clock stop mode of the McBSP provides compatibility with the SPI protocol. When the McBSP is configured in clock stop mode, the transmitter and receiver are internally synchronized, so that the McBSP functions as an SPI master or slave device. The transmit clock signal (CLKX) corresponds to the serial clock signal (SCK) of the SPI protocol, while the transmit frame-synchronization signal (FSX) is used as the slave-enable ($\overline{SS}$) signal.

The receive clock signal (CLKR) and receive frame-synchronization signal (FSR) are not used in the clock stop mode because these signals are internally connected to their transmit counterparts, CLKX and FSX.

2.9.3 Bits Used to Enable and Configure the Clock Stop Mode

Table 18 lists the bits required to configure the McBSP as an SPI device. Table 19 shows how the various combinations of the clock stop mode (CLKSTP) bit and the transmit clock polarity (CLKXP) bit create four possible clock stop mode configurations. The timing diagrams in Section 2.9.4 show the effects of CLKSTP and CLKXP.

Table 18. Bits Used to Enable and Configure the Clock Stop Mode

Register	Bit	Description
Serial port control register (SPCR)	CLKSTP	Enables the clock stop mode and to select one of two timing variations.
Pin control register (PCR)	CLKXP	Determines the polarity of the CLKX signal.
	CLKXM	Determines whether CLKX is an input signal (McBSP as slave) or an output signal (McBSP as master).
Receive control register (RCR)	RPHASE	You must use a single-phase receive frame (RPHASE = 0).
	RFRLEN1	You must use a receive frame length of 1 serial word (RFRLEN1 = 0).
	RWDLEN1	Determines the receive packet length. RWDLEN1 must be equal to XWDLEN1 because in the clock stop mode, the McBSP transmit and receive circuits are synchronized to a single clock.
Transmit control register (XCR)	XPHASE	You must use a single-phase transmit frame (XPHASE = 0).
	XFRLEN1	You must use a transmit frame length of 1 serial word (XFRLEN1 = 0).
	XWDLEN1	Determines the transmit packet length. XWDLEN1 must be equal to RWDLEN1 because in the clock stop mode, the McBSP transmit and receive circuits are synchronized to a single clock.

Table 19. Effects of CLKSTP and CLKXP Bits on the Clock Scheme

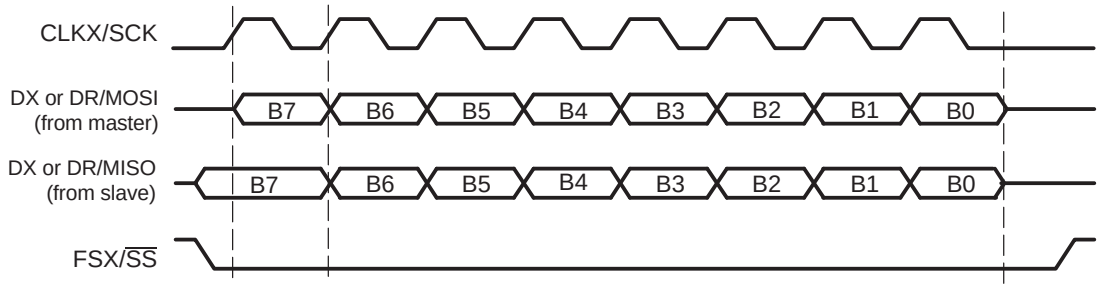
CLKSTP Bit in SPCR	CLKXP Bit in PCR	Clock Scheme
0 or 1	X	Clock stop mode is disabled. Clock is enabled for non-SPI mode.
		Low inactive state:
2h	0	Low inactive state without delay: The McBSP transmits data on the rising edge of CLKX and receives data on the falling edge of CLKR. See Figure 42.
3h	0	Low inactive state with delay: The McBSP transmits data one-half cycle ahead of the rising edge of CLKX and receives data on the rising edge of CLKR. See Figure 43.
		High inactive state:
2h	1	High inactive state without delay: The McBSP transmits data on the falling edge of CLKX and receives data on the rising edge of CLKR. See Figure 44.
3h	1	High inactive state with delay: The McBSP transmits data one-half cycle ahead of the falling edge of CLKX and receives data on the falling edge of CLKR. See Figure 45.

2.9.4 Clock Stop Mode Timing Diagrams

The timing diagrams for the four possible clock stop mode configurations are shown in Figure 42 through Figure 45. Notice that the frame-synchronization signal used in clock stop mode is active throughout the entire transmission as a slave-enable signal. Although the timing diagrams show 8-bit transfers, the packet length can be set to 8, 12, 16, 20, 24, or 32 bits per packet. The receive packet length is selected with the RWDLEN1 bits in RCR, and the transmit packet length is selected with the XWDLEN1 bits in XCR. For clock stop mode, the values of RWDLEN1 and XWDLEN1 must be the same because the McBSP transmit and receive circuits are synchronized to a single clock.

Note: Even if multiple words are consecutively transferred, the CLKX signal is always stopped and the FSX signal returns to the inactive state after a packet transfer. When consecutive packet transfers are performed, this leads to a minimum idle time of two bit-periods between each packet transfer.

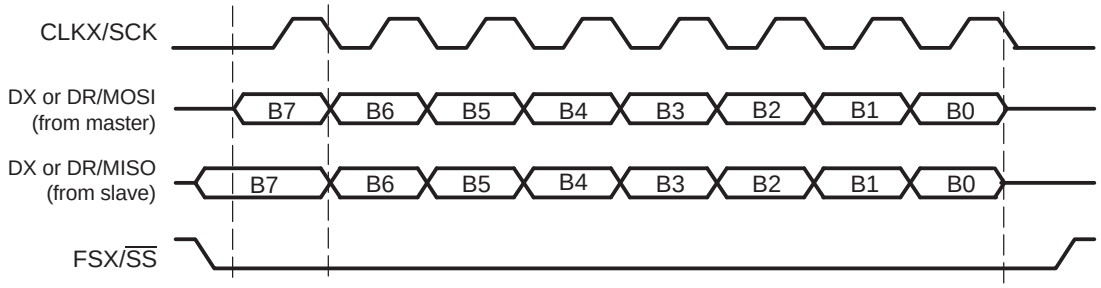
Figure 42. SPI Transfer with CLKSTP = 2h (no clock delay) and CLKXP = 0



1) If McBSP is SPI master (CLKXM = 1), MOSI = DX; if McBSP is SPI slave (CLKXM = 0), MOSI = DR.

2) If McBSP is SPI master (CLKXM = 1), MISO = DR; if McBSP is SPI slave (CLKXM = 0), MISO = DX.

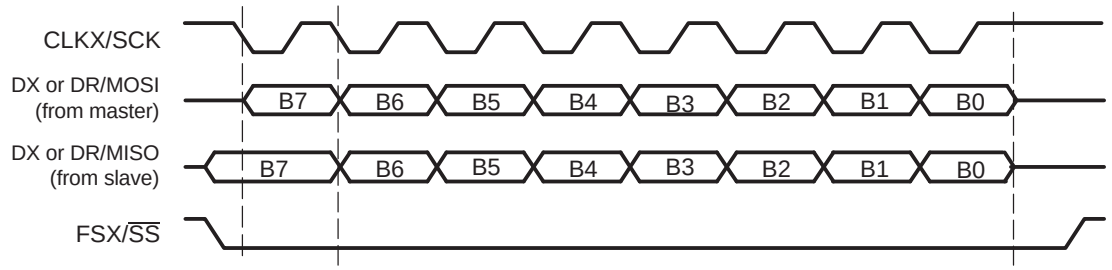
Figure 43. SPI Transfer With CLKSTP = 3h (clock delay) and CLKXP = 0



1) If McBSP is SPI master (CLKXM = 1), MOSI = DX; if McBSP is SPI slave (CLKXM = 0), MOSI = DR.

2) If McBSP is SPI master (CLKXM = 1), MISO = DR; if McBSP is SPI slave (CLKXM = 0), MISO = DX.

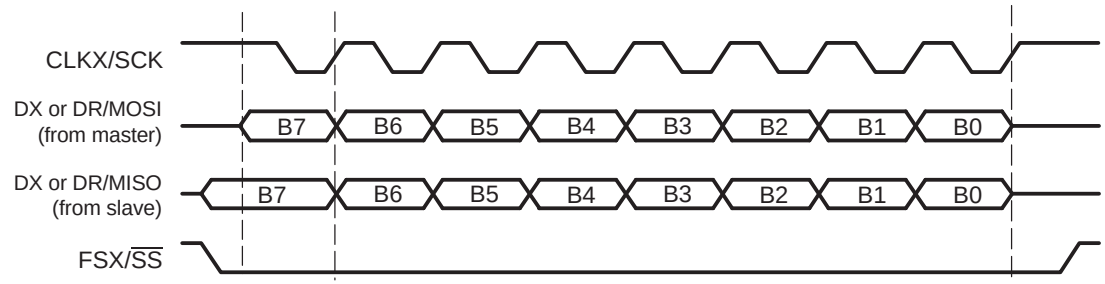
Figure 44. SPI Transfer With CLKSTP = 2h (no clock delay) and CLKXP = 1



1) If McBSP is SPI master (CLKXM = 1), MOSI = DX; if McBSP is SPI slave (CLKXM = 0), MOSI = DR.

2) If McBSP is SPI master (CLKXM = 1), MISO = DR; if McBSP is SPI slave (CLKXM = 0), MISO = DX.

Figure 45. SPI Transfer With CLKSTP = 3h (clock delay) and CLKXP = 1



1) If McBSP is SPI master (CLKXM = 1), MOSI = DX; if McBSP is SPI slave (CLKXM = 0), MOSI = DR.

2) If McBSP is SPI master (CLKXM = 1), MISO = DR; if McBSP is SPI slave (CLKXM = 0), MISO = DX.

2.9.5 Configuring a McBSP for SPI Operation

Perform the following procedure to configure the McBSP for SPI master or slave operation:

1. Place the transmitter and receiver in reset:
   a. To reset the transmitter, clear the transmitter reset (XRST) bit in SPCR to 0.
   b. To reset the receiver, clear the receiver reset (RRST) bit in SPCR to 0.
2. To reset the sample rate generator, clear the sample rate generator reset (GRST) bit in SPCR to 0.
3. Program the appropriate McBSP registers to configure the McBSP for proper operation as an SPI master or an SPI slave.
4. To release the sample rate generator from reset, set the sample rate generator reset (GRST) bit in SPCR to 1. Make sure that during the write to SPCR, you only modify the GRST bit; otherwise, you will modify the McBSP configuration you selected in the previous step.
5. After the sample rate generator is released from reset, wait two sample rate generator clock periods for the McBSP logic to stabilize.
6. Enable the transmitter and receiver:
   a. If the CPU services the McBSP transmit and receive buffers, then you can immediately enable the transmitter (XRST = 1 in SPCR) and enable the receiver (RRST = 1 in SPCR).
   b. If the DMA controller services the McBSP transmit and receive buffers, then you must first configure the DMA controller (this includes enabling the channels that service the McBSP buffers). When the DMA controller is ready, set XRST = 1 and RRST = 1.
   c. In either case, make sure you only change the XRST and RRST bits when you write to SPCR; otherwise, you will modify the bit settings you selected earlier in this procedure.
7. After the transmitter and receiver are released from reset, wait two sample rate generator clock periods for the McBSP logic to stabilize.
8. If necessary, enable the frame-sync logic of the sample rate generator. After the required data acquisition setup is done (DXR is loaded with data), set the FRST bit in SPCR to 1 if an internally generated frame-sync pulse is required (that is, if the McBSP is the SPI master).

2.9.6 McBSP as the SPI Master

An SPI interface with the McBSP used as the master is shown in Figure 46. When the McBSP is configured as a master, the transmit output signal (DX) is used as the MOSI signal of the SPI protocol, and the receive input signal (DR) is used as the MISO signal. The register bit values required to configure the McBSP as a master are listed in Table 20. After the table are more details about the configuration requirements.

Figure 46. McBSP as the SPI Master

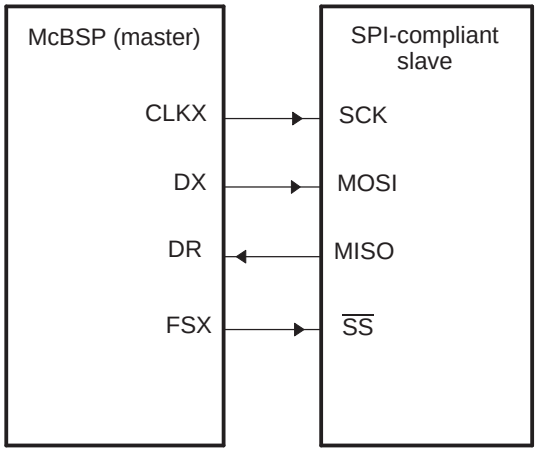


Table 20. Bit Values Required to Configure the McBSP as an SPI Master

Register	Required Bit Setting	Description
Serial port control register (SPCR)	CLKSTP = 2h or 3h	The clock stop mode (without or with a clock delay) is selected.
Pin control register (PCR)	FSXM = 1	The FSX pin is an output pin driven according to the FSGM bit in SRGR.
	CLKXM = 1	The CLKX pin is an output pin driven by the internal sample rate generator. Because CLKSTP is equal to 2h or 3h, CLKR is driven internally by CLKX.
	SCLKME = 0	SCLKME bit is used in conjunction with the CLKSM bit in SRGR to determine the source for the input clock. The clock generated by the sample rate generator (CLKG) is derived from the McBSP internal input clock.
	FSXP = 1	The FSX pin is active low.
	CLKXP = 0 or 1	The polarity of CLKX as seen on the CLKX pin is positive (CLKXP = 0) or negative (CLKXP = 1).
Sample rate generator register (SRGR)	CLKSM = 1	CLKSM bit is used in conjunction with the SCLKME bit in PCR to determine the source for the input clock. The clock generated by the sample rate generator (CLKG) is derived from the McBSP internal input clock.
	FSGM = 0	The transmitter drives a frame-sync pulse on the FSX pin every time data is transferred from DXR to XSR.
	CLKGDV = 0 to FFh	CLKGDV defines the divide-down value for CLKG.
Transmit control register (XCR)	XDATDLY = 1	This setting provides the correct setup time on the FSX signal, 1-bit data delay.
Receive control register (RCR)	RDATDLY = 1	This setting provides the correct setup time on the FSX signal, 1-bit data delay.

When the McBSP functions as the SPI master, it controls the transmission of data by producing the serial clock signal. The clock signal on the CLKX pin is enabled only during packet transfers. When packets are not being transferred, the CLKX pin remains high or low depending on the polarity used.

For SPI master operation, the CLKX pin must be configured as an output. The sample rate generator is then used to derive the CLKX signal from the McBSP internal input clock. The clock stop mode internally connects the CLKX pin to the CLKR signal so that no external signal connection is required on the CLKR pin, and both the transmit and receive circuits are clocked by the master clock (CLKX).

The data delay parameters of the McBSP (XDATDLY and RDATDLY) must be set to 1 for proper SPI master operation. A data delay value of 0 or 2 is undefined in the clock stop mode.

The McBSP can also provide a slave-enable ($\overline{SS}$) signal on the FSX pin. If a slave-enable signal is required, the FSX pin must be configured as an output, and the transmitter must be configured so that a frame-sync pulse is generated automatically each time a packet is transmitted (FSGM = 0). The polarity of the FSX pin is programmable high or low; however, in most cases the pin should be configured active-low.

When the McBSP is configured as described for SPI-master operation, the bit fields for frame-sync pulse width (FWID) and frame-sync period (FPER) are overridden, and custom frame-sync waveforms are not allowed. The signal becomes active before the first bit of a packet transfer, and remains active until the last bit of the packet is transferred. After the packet transfer is complete, the FSX signal returns to the inactive state.

2.9.7 McBSP as the SPI Slave

An SPI interface with the McBSP used as a slave is shown in Figure 47. When the McBSP is configured as a slave, DX is used as the MISO signal, and DR is used as the MOSI signal. The register bit values required to configure the McBSP as a slave are listed in Table 21.

Figure 47. McBSP as an SPI Slave

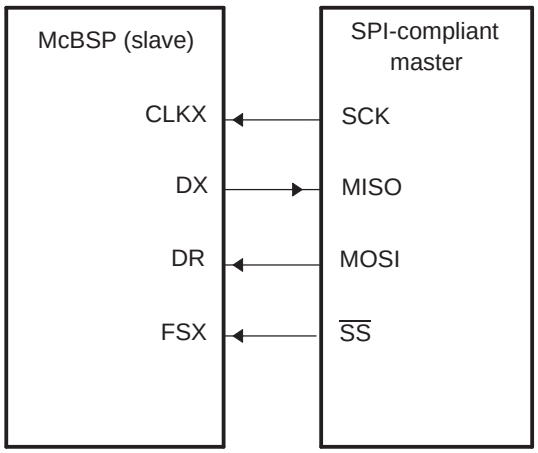


Table 21. Bit Values Required to Configure the McBSP as an SPI Slave

Register	Required Bit Setting	Description
Serial port control register (SPCR)	CLKSTP = 2h or 3h	The clock stop mode (without or with a clock delay) is selected.
Pin control register (PCR)	FSXM = 0	The FSX pin is an input pin, so that it can be driven by the SPI master.
	CLKXM = 0	The CLKX pin is an input pin, so that it can be driven by the SPI master. Because CLKSTP is equal to 2h or 3h, CLKR is driven internally by CLKX.
	SCLKME = 0	SCLKME bit is used in conjunction with the CLKSM bit in SRGR to determine the source for the input clock. The clock generated by the sample rate generator (CLKG) is derived from the McBSP internal input clock. (The sample rate generator is used to synchronize the McBSP logic with the externally-generated master clock.)
	FSXP = 1	The FSX pin is active low.
	CLKXP = 0 or 1	The polarity of CLKX as seen on the CLKX pin is positive (CLKXP = 0) or negative (CLKXP = 1).
Sample rate generator register (SRGR)	CLKSM = 1	CLKSM bit is used in conjunction with the SCLKME bit in PCR to determine the source for the input clock. The clock generated by the sample rate generator (CLKG) is derived from the McBSP internal input clock. (The sample rate generator is used to synchronize the McBSP logic with the externally-generated master clock.)
	CLKGDV = 1	The sample rate generator divides the McBSP internal input clock by 2 before generating CLKG.
Transmit control register (XCR)	XDATDLY = 0	This bit must be 0 for SPI slave operation.
Receive control register (RCR)	RDATDLY = 0	This bit must be 0 for SPI slave operation.

When the McBSP is used as an SPI slave, the master clock and slave-enable signals are generated externally by a master device. Accordingly, the CLKX and FSX pins must be configured as inputs. The CLKX pin is internally connected to the CLKR signal, so that both the transmit and receive circuits of the McBSP are clocked by the external master clock. The FSX pin is also internally connected to the FSR signal, and no external signal connections are required on the CLKR and FSR pins.

Although the CLKX signal is generated externally by the master and is asynchronous to the McBSP, the sample rate generator of the McBSP must be enabled for proper SPI slave operation. The sample rate generator should be programmed to its maximum rate of half the McBSP internal input clock rate. The internal sample rate clock is then used to synchronize the McBSP logic to the external master clock and slave-enable signals.

The McBSP requires an active edge of the slave-enable ($\overline{SS}$) signal on the FSX input for each transfer. This means that the master device must assert the slave-enable signal at the beginning of each transfer, and deassert the signal after the completion of each packet transfer; the slave-enable signal cannot remain active between transfers.

The data delay parameters of the McBSP must be cleared to 0 for proper SPI slave operation. A value of 1 or 2 is undefined in the clock stop mode.

2.10 Resetting the Serial Port: RRST, XRST, GRST, and RESET

Device reset or McBSP reset: When the McBSP is reset by device reset or McBSP reset, the state machine is reset to its initial state. All counters and status bits are reset. This includes the receive status bits RFULL, RRDY, and RSYNCERR, and the transmit status bits XEMPTY, XRDY, and XSYNCERR in the serial port control register (SPCR).

The serial port can be reset in the following two ways:

- Device reset (RESET pin is low) places the receiver, the transmitter, and the sample rate generator in reset. When the device reset is removed (RESET = 1), FRST = GRST = RRST = XRST = 0 in SPCR, keeping the entire serial port in the reset state.
- The serial port transmitter and receiver can be independently reset by the XRST and RRST bits in SPCR. The sample rate generator is reset by the GRST bit in SPCR.

Table 22 shows the state of the McBSP pins when the serial port is reset by these methods.

Table 22. Reset State of McBSP Pins

Pin	Direction	Device Reset ($\overline{\text{RESET}}$ = 0)	McBSP Reset
			Receiver Reset (RRST = 0 and GRST = 1)
DR	I	Input	Input
CLKR	I/O/Z	Input	Known state if input; CLKR if output
FSR	I/O/Z	Input	Known state if input; FSRP(inactive state) if output
CLKS	I	Input	Input
			Transmitter Reset (XRST = 0 and GRST = 1)
DX	O/Z	High impedance	High impedance
CLKX	I/O/Z	Input	Known state if input; CLKX if output
FSX	I/O/Z	Input	Known state if input; FSXP(inactive state) if output
CLKS	I	Input	Input

2.10.1 Software Reset Considerations

McBSP reset: When the receiver and transmitter reset bits, RRST and XRST in SPCR, are written with 0, the respective portions of the McBSP are reset and activity in the corresponding section stops. All input-only pins, such as DR, and all other pins that are configured as inputs are in a known state. FS(R/X) is driven to its inactive state (same as its polarity bit, FS(R/X)P) if it is an output. If CLK(R/X) are programmed as outputs, they are driven by CLKG, provided that GRST = 1. The DX pin is in the high-impedance state when the transmitter is reset. During normal operation, the sample rate generator can be reset by writing a 0 to GRST. The sample rate generator should only be reset when not being used by the transmitter ro the receiver. In this case, the internal sample rate generator clock CLKG, and its frame sync signal (FSG) is driven inactive (low). When the sample rate generator is not in the reset state (GRST = 1), FSR and FSX are in an inactive state when RRST = 0 and XRST = 0, respectively, even if they are outputs driven by FSG. This ensures that when only one portion of the McBSP is in reset, the other portion can continue operation when FRST = 1 and frame sync is driven by FSG.

Sample-rate generator reset: As mentioned previously, the sample rate generator is reset when the device is reset or when its reset bit, GRST in SPCR, is written with 0.

Emulator software reset: In the event of an emulator software reset initiated from the DSP, the McBSP register values are reset to their default values.

2.10.2 Hardware Reset Considerations

When the McBSP is reset due to device reset, the entire serial port (including the transmitter, receiver, and the sample rate generator) is reset. All input-only pins and 3-state pins should be in a known state. The output-only pin, DX, is in the high-impedance state. When the device is pulled out of reset, the serial port remains in the reset condition (RRST = XRST = FRST = GRST = 0 in SPCR).

2.11 McBSP Initialization Procedure

The McBSP initialization procedure varies depending on the specific system setup. Section 2.11.1 provides a general initialization sequence.

The transmitter and the receiver of the McBSP can operate independently from each other. Therefore, they can be placed in or taken out of reset individually by modifying only the desired bit in the registers without disrupting the other portion. The steps in the following sections discuss the initialization procedure for taking both the transmitter and the receiver out of reset. To initialize only one portion, configure only the portion desired.

The McBSP internal sample rate generator and internal frame sync generator are shared between the transmitter and the receiver. Table 23 and Table 24 describe their usage base upon the clock and frame sync configurations of the receiver and transmitter, respectively.

Table 23. Receiver Clock and Frame Configurations

CLKR Source	FSR Source	Comment on Configuration
Internal	Internal	The McBSP internal sample rate generator and internal frame sync generator are used by the receiver.
External	Internal	The McBSP internal sample rate generator and internal frame sync generator are used by the receiver.
Internal	External	The McBSP internal sample rate generator is used but the internal frame sync generator is not used by the receiver.
External	External	The McBSP internal sample rate generator and internal frame sync generator are not used by the receiver.

Table 24. Transmitter Clock and Frame Configurations

CLKX Source	FSX Source	Comment on Configuration
Internal	Internal	The McBSP internal sample rate generator is used by the transmitter. The transmitter can generate frame sync FSX in one of two ways. First, it can generate FSX by using the internal frame sync generator (FSGM = 1). Alternatively, it can generate FSX upon each DXR-to-XSR copy (FSGM = 0). In this case, the internal frame sync generator can be kept in reset (FRST = 0) if it is not used by the receiver. You can follow the general initialization sequence in Section 2.11.1.
External	Internal	The McBSP internal sample rate generator and internal frame sync generator are not used by the transmitter. This configuration is only valid with FSGM = 0 where the McBSP transmitter generates FSX upon each DXR-to-XSR copy. You can follow the general initialization sequence in Section 2.11.1.
Internal	External	The McBSP internal sample rate generator is used by the transmitter but the internal frame sync generator is not.
External	External	The McBSP internal sample rate generator and internal frame sync generator are not used by the transmitter.

2.11.1 General Initialization Procedure

This section provides the general initialization procedure.

1. With the McBSP still in reset (Power and Sleep Controller (PSC) in the default state):
   a. Perform the necessary device pin multiplexing setup (see the device-specific data manual).
   b. Program the VDD3P3V_PWDN register in the System Module to power up the IO pins for the McBSP (see the device-specific data manual).

2. Ensure that no portion of the McBSP is using the internal sample rate generator signal CLKG and the internal frame sync generator signal FSG (GRST = FRST = 0 in SPCR). The respective portion of the McBSP needs to be in reset (XRST = 0 and/or RRST = 0 in SPCR).
3. Program the control registers as required. Ensure the internal sample rate generator and the internal frame sync generator are still in reset (GRST = FRST = 0). Also ensure the respective portion of the McBSP is still in reset in this step (XRST = 0 and/or RRST = 0).
4. Wait for proper internal synchronization. If the external device provides the bit clock, wait for two CLKR or CLKX cycles. If the McBSP generates the bit clock as a clock master, wait for two CLKSRG cycles. In this case, the clock source to the sample rate generator (CLKSRG) is selected by the CLKSM bit in SRGR and the SCLKME bit in PCR.
5. Skip this step if the bit clock is provided by the external device. This step only applies if the McBSP is the bit clock master and the internal sample rate generator is used.
   a. Start the sample rate generator by setting the GRST bit to 1. Wait two CLKG bit clocks for synchronization. CLKG is the output of the sample rate generator.
   b. On the next rising edge of CLKSRG, CLKG transitions to 1 and starts clocking with a frequency equal to 1/(CLKGDV + 1) of the sample rate generator source clock CLKSRG.
6. Skip this step if the transmitter is not used. If the transmitter is used, a transmit sync error (XSYNCERR) may occur when it is enabled for the first time after device reset. The purpose of this step is to clear any potential XSYNCERR that occurs on the transmitter at this time:
   a. Set the XRST bit to 1 to enable the transmitter.
   b. Wait for any unexpected frame sync error to occur. If the external device provides the bit clock, wait for two CLKR or CLKX cycles. If the McBSP generates the bit clock as a clock master, wait for two CLKG cycles. The unexpected frame sync error (XSYNCERR), if any, occurs within this time period.
   c. Disable the transmitter (XRST = 0). This clears any outstanding XSYNCERR.
7. Setup data acquisition as required:
   a. If the EDMA is used to service the McBSP, setup data acquisition as desired and start the EDMA in this step, before the McBSP is taken out of reset.
   b. If CPU interrupt is used to service the McBSP, enable the transmit and/or receive interrupt as required.
   c. If CPU polling is used to service the McBSP, no action is required in this step.
8. Set the XRST bit and/or the RRST bit to 1 to enable the corresponding section of the McBSP. The McBSP is now ready to transmit and/or receive.
   a. If the EDMA is used to service the McBSP, it services the McBSP automatically upon receiving the XEVT and/or REVT.
   b. If CPU interrupt is used to service the McBSP, the interrupt service routine is automatically entered upon receiving the XINT and/or RINT.
   c. If CPU polling is used to service the McBSP, it can do so now by polling the XRDY and/or RRDY bit.
9. If the internal frame sync generator is used (FSGM = 1), proceed to the additional steps to turn on the internal frame sync generator. Initialization is complete if any one of the following is true:
   a. The external device generates frame sync FSX and/or FSR. The McBSP is now ready to transmit and/or receive upon receiving external frame sync.
   b. The McBSP generates transmit frame sync FSX upon each DXR-to-XSR copy. The internal frame sync generator is not used (FSGM = 0).

   The following additional steps to turn on the internal frame sync generator apply only if FSGM = 1:
10. Skip this step if the transmitter is not used. If the transmitter is used, ensure that DXR is serviced before you start the internal frame sync generator. You can do so by checking XEMPTY = 1 (XSR is not empty) in SPCR.
11. Set the FRST bit to 1 to start the internal frame sync generator. The internal frame sync signal FSG is generated on a CLKG active edge after 7 to 8 CLKG clocks have elapsed.

2.11.2 Special Case: External Device is the Transmit Frame Master

Care must be taken if the transmitter expects a frame sync from an external device. After the transmitter comes out of reset (XRST = 1), it waits for a frame sync from the external device. If the first frame sync arrives very shortly after the transmitter is enabled, the CPU or EDMA controller may not have a chance to service the data transmit register (DXR). In this case, the transmitter shifts out the default data in the transmit shift register (XSR) instead of the desired value, which has not yet arrived in DXR. This causes problems in some applications, as the first data element in the frame is invalid. The data stream appears element-shifted (the first data word may appear in the second channel instead of the first).

To ensure proper operation when the external device is the frame master, you must assure that DXR is already serviced with the first word when a frame sync occurs. To do so, you can keep the transmitter in reset until the first frame sync is detected. Software is setup such that it will only take the transmitter out of reset (XRST = 1) promptly after detecting the first frame sync. This assures that the transmitter does not begin data transfers at the data pin during the first frame sync period. This also provides almost an entire frame period for the DSP to service DXR with the first word before the second frame sync occurs. The transmitter only begins data transfers upon receiving the second frame sync. At this point, DXR is already serviced with the first word.

2.11.2.1 How to Detect First Frame Sync

Although the McBSP is capable of generating an interrupt to the CPU upon the detection of frame synchronization (XINTM = 2h and/or RINTM = 2h in the serial port control register (SPCR)), the McBSP requires the associated portion (receiver/transmitter) of the McBSP to be out of reset in order for the interrupt to be generated. Therefore, instead of directly using the McBSP interrupt to detect the first frame sync, you can use the GPIO peripheral. This can be achieved by connecting the frame sync signal to a GPIO pin. Software can either poll the GPIO pin to detect the first frame sync or program the GPIO peripheral to generate an interrupt to the CPU upon detecting the first frame sync edge. For more information on the GPIO peripheral, see the *TMS320C642x DSP General-Purpose Input/Output (GPIO) User's Guide* (SPRUEM8).

The following are some recommended GPIO pin(s) on the C642x DSP that you can use to detect the first McBSP external frame sync:

- **GPIO pin located near the McBSP pins.** Connect the external frame sync to both the McBSP FSX/FSR pin(s) and the dedicated GPIO pin.
- **GPIO pin multiplexed with the McBSP FSX signal.** Note that on the C642x DSP, the GPIO pins (of the GPIO peripheral) are multiplexed with the McBSP pins. Software can program the C642x DSP pin multiplexing register (PINMUX) to default these pins to the GPIO function, and only switch them to the McBSP function upon detecting the first frame sync. This method is only recommended if the external device is both the frame sync and clock master; that is, the external device drives both the FSX and CLKX signals. This method is not recommended if the McBSP is the clock master (driving CLKX and/or CLKR), as the “on-the-fly” pin multiplexed switching can cause a glitch on the CLKX/CLKR pin. For more details on pin multiplexing, see the device-specific data manual.

2.11.2.2 Initialization Procedure When External Device is Frame Sync Master

The initialization procedure assumes the following:

- Using a GPIO pin multiplexed with the McBSP FSX signal. If a dedicated GPIO pin is used instead, skip step 1 and step 8b.
- Software polls the GPIO pin to detect the first frame sync. If the GPIO interrupt is used instead to detect the first frame sync, step 8 can be performed within an interrupt service routine (ISR).

1. On the C642x DSP, the GPIO and McBSP signals are multiplexed together. Start by programming the pin multiplexing register (PINMUX) to select the GPIO function on the GPIO/McBSP multliplexed pins. Program the GPIO peripheral so that these pins function as GPIO inputs.
2. Ensure that no portion of the McBSP is using the internal sample rate generator signal CLKG and the internal frame sync generator signal FSG (GRST = FRST = 0 in SPCR). The respective portion of the McBSP needs to be in reset (XRST = 0 and/or RRST = 0 in SPCR).

3. Program the sample rate generator register (SRGR) and other control registers as required. Ensure the internal sample rate generator and the internal frame sync generator are still in reset (GRST = FRST = 0 in SPCR). Also ensure the respective portion of the McBSP is still in reset in this step (XRST = 0 and/or RRST = 0 in SPCR).
4. Wait for proper McBSP internal synchronization:
   a. If the external device provides the bit clock, wait for two CLKR or CLKX cycles. Skip step 5.
   b. If the McBSP generates the bit clock as a clock master, wait for two CLKSRG cycles. In this case, the clock source to the sample rate generator (CLKSRG) is selected by the CLKSM bit in SRGR.
5. Skip this step if the bit clock is provided by the external device. This step only applies if the McBSP is the bit clock master and the internal sample rate generator is used.
   a. Start the sample rate generator by setting the GRST bit in SPCR to 1. Wait two CLKG bit clocks for synchronization. CLKG is the output of the sample rate generator.
   b. On the next rising edge of CLKSRG, CLKG transitions to 1 and starts clocking with a frequency equal to 1/(CLKGDV + 1) of the sample rate generator source clock CLKSRG.
6. A transmit sync error (XSYNCERR) may occur when it is enabled for the first time after device reset. The purpose of this step is to clear any potential XSYNCERR that occurs on the transmitter at this time:
   a. Set the XRST bit in SPCR to 1 to enable the transmitter.
   b. Wait for any unexpected frame sync error to occur. If the external device provides the bit clock, wait for two CLKR or CLKX cycles. If the McBSP generates the bit clock as a clock master, wait for two CLKG cycles. The unexpected frame sync error (XSYNCERR), if any, occurs within this time period.
   c. Disable the transmitter (XRST = 0). This clears any outstanding XSYNCERR.
7. Setup data acquisition as required:
   a. If the EDMA controller is used to service the McBSP, setup data acquisition as desired and start the EDMA controller in this step, before the McBSP is taken out of reset.
   b. If the CPU interrupt is used to service the McBSP, no action is required in this step.
   c. If CPU polling is used to service the McBSP, no action is required in this step.
8. Poll the GPIO pin (through reading the appropriate registers in the GPIO peripheral) to detect the first transmit frame sync from the external device. Upon detection of the first frame sync, perform the following in this order:
   a. Set the XRST bit and/or the RRST bit to 1 to enable the respective portion of the McBSP. The McBSP is now ready to transmit and/or receive.
   b. Program PINMUX to switch the GPIO/McBSP multiplexed pins to the McBSP function.
9. Service the McBSP:
   a. If CPU polling is used to service the McBSP in normal operations, it can do so upon exit from the ISR.
   b. If the CPU interrupt is used to service the McBSP in normal operations, upon XRDY interrupt service routine is entered. The ISR should be setup to verify that XRDY = 1 and service the McBSP accordingly.
   c. If the EDMA controller is used to service the McBSP in normal operations, it services the McBSP automatically upon receiving the XEVT and/or REVT.
10. Upon detection of the second frame sync, DXR is already serviced and the transmitter is ready to transmit the valid data. The receiver is also serviced properly by the DSP.

2.12 Interrupt Support

The McBSP can send both receive and transmit interrupts to the DSP controller. For more details on the Interrupt Controller, see the *TMS320C64x+ DSP Megamodule Reference Guide* (SPRU871).

2.12.1 Interrupt Events and Requests

The RRDY and XRDY bits in the serial port control register (SPCR) indicate the ready state of the McBSP receiver and transmitter, respectively. Writes and reads from the serial port can be synchronized by any of the following methods:

- Polling RRDY and XRDY bits in SPCR
- Using the events sent to the EDMA controller (REVT and XEVT)
- Using the interrupts to the CPU (RINT and XINT) that the events generate

Reading DRR and writing to DXR affects RRDY and XRDY, respectively.

2.12.1.1 Interrupt Events: RINT and XINT

The receive interrupt (RINT) and transmit interrupt (XINT) signals inform the DSP CPU of changes to the serial port status. Three options exist for configuring these interrupts. These options are set by the receive/transmit interrupt mode bits (RINTM and XINTM) in SPCR. The possible values of the mode, and the configurations they represent, are:

- (R/X)INTM = 00b: Interrupt on every serial element by tracking the (R/X)RDY bits in SPCR.
- (R/X)INTM = 01b: Interrupt at the end of a subframe (16 elements or less) within a frame. See Section 2.8.9 for more details.
- (R/X)INTM = 10b: Interrupt on detection of frame synchronization pulses. The associated portion (receiver/transmitter) of the McBSP must be out of reset.
- (R/X)INTM = 11b: Interrupt on frame synchronization error. Note that if any of the other interrupt modes are selected, (R/X)SYNCERR may be read when servicing the interrupts to detect this condition. See Section 2.6.5.2 and Section 2.6.5.5 for more details on synchronization error.

2.12.1.2 Receive Ready Status: RINT and RRDY

RRDY = 1 indicates that the RBR contents have been copied to DRR and that the data can now be read by either the CPU or the EDMA controller. Once that data has been read by either the CPU, RRDY is cleared to 0. Also, at device reset or serial port receiver reset (RRST = 0), the RRDY bit is cleared to 0 to indicate that no data has been received and loaded into DRR. RRDY directly drives the McBSP receive interrupt (RINT) to the CPU if RINTM = 00b (default value) in SPCR.

2.12.1.3 Transmit Ready Status: XINT and XRDY

XRDY = 1 indicates that the DXR contents have been copied to XSR and that DXR is ready to be loaded with a new data word. When the transmitter transitions from reset to non-reset (XRST transitions from 0 to 1), XRDY also transitions from 0 to 1 indicating that DXR is ready for new data. Once new data is loaded by the CPU, the XRDY bit is cleared to 0. However, once this data is copied from DXR to XSR, the XRDY bit transitions again from 0 to 1. The CPU can write to DXR although XSR has not yet been shifted out on DX. XRDY directly drives the McBSP transmit interrupt (XINT) to the CPU if XINTM = 00b (default value) in SPCR.

Note: If the polling method is used to service the transmitter, the CPU should wait for one McBSP bit clock (CLKX) before polling again to write the next element in DXR. This is because XRDY transitions occur based on bit clock and not CPU clock. The CPU clock is much faster and can cause false XRDY status, leading to data errors due to over-writes.

2.12.2 Interrupt Multiplexing

The RINT and XINT interrupts generated by the McBSP peripheral to the CPU are multiplexed with other interrupt sources. Refer to the device-specific data manual to determine how pin multiplexing affects the McBSP.

2.13 EDMA Event Support

2.13.1 Receive Ready Status: REVT and RRDY

RRDY = 1 in the serial port control register (SPCR) indicates that the RBR contents have been copied to DRR and that the data can now be read by the EDMA controller. Once that data has been read by the EDMA controller, RRDY is cleared to 0. Also, at device reset or serial port receiver reset (RRST = 0 in SPCR), the RRDY bit is cleared to 0 to indicate that no data has been received and loaded into DRR. RRDY directly drives the McBSP receive event to the EDMA controller (via REVT).

For detailed information on using the EDMA to read or write to the McBSP, see the *TMS320C642x DSP Enhanced Direct Memory Access (EDMA) Controller User's Guide* (SPRUEM5).

2.13.2 Transmit Ready Status: XEVT and XRDY

XRDY = 1 in the serial port control register (SPCR) indicates that the DXR contents have been copied to XSR and that DXR is ready to be loaded with a new data word. When the transmitter transitions from reset to non-reset (XRST transitions from 0 to 1 in SPCR), XRDY also transitions from 0 to 1 indicating that DXR is ready for new data. Once new data is loaded by the EDMA controller, the XRDY bit is cleared to 0. However, once this data is copied from DXR to XSR, the XRDY bit transitions again from 0 to 1. The EDMA controller can write to DXR although XSR has not yet been shifted out on DX. XRDY directly drives the transmit synchronization event to the EDMA controller (via XEVT).

For detailed information on using the EDMA to read or write to the McBSP, see the *TMS320C642x DSP Enhanced Direct Memory Access (EDMA) Controller User's Guide* (SPRUEM5).

Note: If the polling method is used to service the transmitter, the CPU should wait for one McBSP bit clock (CLKX) before polling again to write the next element in DXR. This is because XRDY transitions occur based on bit clock and not CPU clock. The CPU clock is much faster and can cause false XRDY status, leading to data errors due to over-writes.

2.14 Power Management

The McBSP can be placed in reduced power modes to conserve power during periods of low activity. The power management of the peripheral is controlled by the processor Power and Sleep Controller (PSC). The PSC acts as a master controller for power management for all of the peripherals on the device.

In order for the McBSP to be placed in power-down mode by the PSC, ensure that the XRDY and RRDY flags in the serial port control register (SPCR) are cleared by performing the following steps:

1. Place the McBSP in reset by clearing the XRST, RRST, FRST, and GRST bits to 0 in SPCR.
2. If EDMA is being used to service the transmitter and/or the receiver, disable the associated EDMA channels. For detailed information on EDMA usage, see the *TMS320C642x DSP Enhanced Direct Memory Access (EDMA) Controller User's Guide* (SPRUEM5).
3. Switch the McBSP clocks and frames to internal clock source:
   a. Set the CLKSM and FSGM bits to 1 in the sample rate generator register (SRGR).
   b. Set the CLKXM, CLKRM, FSXM, and FSRM bits to 1 in the pin control register (PCR).
   c. Clear the SCLKME bit to 0 in PCR.
4. Bring the McBSP out of reset by setting the XRST, RRST, and GRST bits to 1 in SPCR.
5. Wait for two CLKSRG cycles for proper internal synchronization.
6. Write a dummy data value to the data transmit register (DXR) in order to clear the first XRDY flag.
7. Wait for at least one McBSP bit clock, since once the first dummy data value is internally copied from DXR to XSR, the XRDY flag transitions again from 0 to 1.
8. Write a second dummy data value to DXR in order to clear the second XRDY flag.
9. Check the RRDY flag in SPCR and if set to 1, read the data receive register (DRR) and discard the data to clear the RRDY flag.
10. Place the McBSP in power-down mode by issuing the proper PSC commands. For detailed information on power management procedures using the PSC, see the *TMS320C642x DSP Power and Sleep Controller (PSC) User's Guide* (SPRUEN8).

Note: After waking up the McBSP from a power-down mode using the proper PSC commands, remember to reconfigure the SPCR, SRGR, and PCR registers to the clock and frame combination that they were in before entering the power-down sequence and discard the two dummy data values that were used to clear the XRDY flags. If EDMA is used, re-enable the corresponding EDMA channels.

2.15 Emulation Considerations

The FREE and SOFT bits are special emulation bits in the serial port control register (SPCR) that determine the state of the McBSP when an emulation suspend event occurs in the emulator. An emulation suspend event corresponds to any type of emulator access to the DSP, such as a hardware or software breakpoint, a probe point, or a printf instruction.

Table 25 shows the effects of the FREE and SOFT bits on the response of the McBSP to emulation suspend events.

Table 25. McBSP Emulation Modes Selectable With the FREE and SOFT Bits of SPCR

FREE Bit in SPCR	SOFT Bit in SPCR	McBSP Emulation Mode
0	0	Immediate stop mode (reset condition). The transmitter and receiver stop immediately in response to an emulation suspend event.
0	1	Soft stop mode. When an emulation suspend event occurs, the transmitter stops after completion of the current word. The receiver is not affected.
1	0 or 1	Free run mode. The transmitter and receiver continue to run when an emulation suspend event occurs.

3 Registers

Table 26 lists the memory-mapped registers for the McBSP. See the device-specific data manual for the memory address of these registers. All other register offset addresses not listed in Table 26 should be considered as reserved locations and the register contents should not be modified.

The McBSP control registers are accessible by the DSP CPUs. You should halt the McBSP before making changes to the serial port control register (SPCR), receive control register (RCR), transmit control register (XCR), and pin control register (PCR). Changes made to these registers without halting the McBSP could result in an undefined state.

Table 26. McBSP Registers

Offset	Acronym	Register Name	Section
-	RBR[1]	Receive buffer register	-
-	RSR[1]	Receive shift register	-
-	XSR[1]	Transmit shift register	-
0h	DRR[2][3]	Data receive register	Section 3.1
4h	DXR[3]	Data transmit register	Section 3.2
8h	SPCR	Serial port control register	Section 3.3
Ch	RCR	Receive control register	Section 3.4
10h	XCR	Transmit control register	Section 3.5
14h	SRGR	Sample rate generator register	Section 3.6
18h	MCR	Multichannel control register	Section 3.7
1Ch	RCERE0	Enhanced receive channel enable register partition A/B	Section 3.8
20h	XCERE0	Enhanced transmit channel enable register partition A/B	Section 3.9
24h	PCR	Pin control register	Section 3.10
28h	RCERE1	Enhanced receive channel enable register partition C/D	Section 3.8
2Ch	XCERE1	Enhanced transmit channel enable register partition C/D	Section 3.9
30h	RCERE2	Enhanced receive channel enable register partition E/F	Section 3.8
34h	XCERE2	Enhanced transmit channel enable register partition E/F	Section 3.9
38h	RCERE3	Enhanced receive channel enable register partition G/H	Section 3.8
3Ch	XCERE3	Enhanced transmit channel enable register partition G/H	Section 3.9

(1) The RBR, RSR, and XSR are not directly accessible via the CPUs or the EDMA controller.
(2) The CPUs and EDMA controller can only read this register; they cannot write to it.
(3) The DRR and DXR are accessible via the CPUs or the EDMA controller.

3.1 Data Receive Register (DRR)

The data receive register (DRR) contains the value to be written to the data bus. The DRR is shown in Figure 48 and described in Table 27.

See the device-specific data manual for the memory address of these registers. Both the CPUs and the EDMA can access DRR in all the memory-mapped locations. An access to *any* EDMA bus location is equivalent to an access to DRR of the corresponding McBSP.

Figure 48. Data Receive Register (DRR)

31	16
DR	
R-0	

15	0
DR	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 27. Data Receive Register (DRR) Field Descriptions

Bit	Field	Value	Description
31-0	DR	0-FFFF FFFFh	Data receive register value to be written to the data bus.

3.2 Data Transmit Register (DXR)

The data transmit register (DXR) contains the value to be loaded into the data transmit shift register (XSR). The DXR is shown in Figure 49 and described in Table 28.

See the device-specific data manual for the memory address of these registers. DXR is accessible via the peripheral bus and via the EDMA bus. Both the CPUs and the EDMA can access DXR in all the memory-mapped locations.

Figure 49. Data Transmit Register (DXR)

31	16
DX	
R-0	

15	0
DX	
R-0	

LEGEND: R = Read only; -*n* = value after reset

Table 28. Data Transmit Register (DXR) Field Descriptions

Bit	Field	Value	Description
31-0	DX	0-FFFF FFFFh	Data transmit register value to be loaded into the data transmit shift register (XSR).

3.3 Serial Port Control Register (SPCR)

The serial port is configured via the serial port control register (SPCR) and the pin control register (PCR). The SPCR contains McBSP status control bits. The SPCR is shown in Figure 50 and described in Table 29.

Figure 50. Serial Port Control Register (SPCR)

31						26	25	24
Reserved							FREE	SOFT
R-0							R/W-0	R/W-0
23	22	21			20	19	18	17	16
FRST	GRST	XINTM				XSYNCERR	XEMPTY	XRDY	XRST
R/W-0	R/W-0	R/W-0				R/W-0	R-0	R-0	R/W-0
15	14			13	12	11	10		8
DLB	RJUST				CLKSTP		Reserved		
R/W-0	R/W-0						R-0		
7	6	5			4	3	2	1	0
DXENA	Reserved	RINTM				RSYNCERR	RFULL	RRDY	RRST
R/W-0	R-0	R/W-0				R/W-0	R-0	R-0	R/W-0

LEGEND: R = Read only; R/W = Read/ Write; -n = value after reset

Table 29. Serial Port Control Register (SPCR) Field Descriptions

Bit	Field	Value	Description
31-26	Reserved	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
25	FREE		Free-running enable mode bit. This bit is used in conjunction with SOFT bit to determine state of serial port clock during emulation halt.
		0	Free-running mode is disabled. During emulation halt, SOFT bit determines operation of McBSP.
		1	Free-running mode is enabled. During emulation halt, serial clocks continue to run.
24	SOFT		Soft bit enable mode bit. This bit is used in conjunction with FREE bit to determine state of serial port clock during emulation halt. This bit has no effect if FREE = 1.
		0	Soft mode is disabled. Serial port clock stops immediately during emulation halt, thus aborting any transmissions.
		1	Soft mode is enabled. During emulation halt, serial port clock stops after completion of current transmission.
23	FRST		Frame-sync generator reset bit.
		0	Frame-synchronization logic is reset. Frame-sync signal (FSG) is not generated by the sample-rate generator.
		1	Frame-sync signal (FSG) is generated after (FPER + 1) number of CLKG clocks; that is, all frame counters are loaded with their programmed values.
22	GRST		Sample-rate generator reset bit.
		0	Sample-rate generator is reset.
		1	Sample-rate generator is taken out of reset. CLKG is driven as per programmed value in sample-rate generator register (SRGR).
21-20	XINTM	0-3h	Transmit interrupt (XINT) mode bit.
		0	XINT is driven by XRDY (end-of-word).
		1h	Reserved
		2h	XINT is generated by a new frame synchronization.
		3h	XINT is generated by XSYNCERR.
19	XSYNCERR		Transmit synchronization error bit. Writing a 1 to XSYNCERR sets the error condition when the transmitter is enabled (XRST = 1). Thus, it is used mainly for testing purposes or if this operation is desired.
		0	No synchronization error is detected.
		1	Synchronization error is detected.

Table 29. Serial Port Control Register (SPCR) Field Descriptions (continued)

Bit	Field	Value	Description
18	XEMPTY		Transmit shift register empty bit.
		0	XSR is empty.
		1	XSR is not empty.
17	XRDY		Transmitter ready bit.
		0	Transmitter is not ready.
		1	Transmitter is ready for new data in DXR.
16	XRST		Transmitter reset bit resets or enables the transmitter.
		0	Serial port transmitter is disabled and in reset state.
		1	Serial port transmitter is enabled.
15	DLB		Digital loop back mode enable bit.
		0	Digital loop back mode is disabled.
		1	Digital loop back mode is enabled.
14-13	RJUST	0-3h	Receive sign-extension and justification mode bit.
		0	Right-justify and zero-fill MSBs in DRR.
		1h	Right-justify and sign-extend MSBs in DRR.
		2h	Left-justify and zero-fill LSBs in DRR.
		3h	Reserved
12-11	CLKSTP	0-3h	Clock stop mode bit. In SPI mode, operates in conjunction with CLKXP bit of pin control register (PCR).
		0	Clock stop mode is disabled. Normal clocking for non-SPI mode.
		1h	Reserved
			In SPI mode with data sampled on rising edge (CLKXP = 0):
		2h	Clock starts with rising edge without delay.
		3h	Clock starts with rising edge with delay.
			In SPI mode with data sampled on falling edge (CLKXP = 1):
		2h	Clock starts with falling edge without delay.
		3h	Clock starts with falling edge with delay.
10-8	Reserved	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
7	DXENA		DX enabler bit. See Section 2.8.4 for details on the DX enabler bit.
		0	DX enabler is off.
		1	DX enabler is on.
6	Reserved	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
5-4	RINTM	0-3h	Receive interrupt (RINT) mode bit.
		0	RINT is driven by RRDY (end-of-word).
		1h	Reserved
		2h	RINT is generated by a new frame synchronization.
		3h	RINT is generated by RSYNCERR.
3	RSYNCERR		Receive synchronization error bit. Writing a 1 to RSYNCERR sets the error condition when the receiver is enabled (RRST = 1). Thus, it is used mainly for testing purposes or if this operation is desired.
		0	No synchronization error is detected.
		1	Synchronization error is detected.
2	RFULL		Receive shift register full bit.
		0	RBR is not in overrun condition.
		1	DRR is not read, RBR is full, and RSR is also full with new word.
1	RRDY		Receiver ready bit.
		0	Receiver is not ready.
		1	Receiver is ready with data to be read from DRR.

Table 29. Serial Port Control Register (SPCR) Field Descriptions (continued)

Bit	Field	Value	Description
0	RRST		Receiver reset bit resets or enables the receiver.
		0	The serial port receiver is disabled and in reset state.
		1	The serial port receiver is enabled.

3.4 Receive Control Register (RCR)

The receive control register (RCR) configures parameters of the receive operations. The RCR is shown in Figure 51 and described in Table 30.

Figure 51. Receive Control Register (RCR)

31	30 24	23 21	20 19	18	17 16
RPHASE	RFRLEN2	RWDLEN2	RCOMPAND	RFIG	RDATDLY
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

15	14 8	7 5	4	3 0
Reserved	RFRLEN1	RWDLEN1	RWDREVRS	Reserved
R-0	R/W-0	R/W-0	R/W-0	R-0

LEGEND: R = Read only; R/ W = Read/Write; -n = value after reset

Table 30. Receive Control Register (RCR) Field Descriptions

Bit	Field	Value	Description
31	RPHASE		Receive phases bit.
		0	Single-phase frame
		1	Dual-phase frame
30-24	RFRLEN2	0-7Fh	Specifies the receive frame length (number of words) in phase 2.
		0	1 word in phase 2
		1h	2 words in phase 2
		2h	3 words in phase 2
		...	...
		7Fh	128 words in phase 2
23-21	RWDLEN2	0-7h	Specifies the receive word length (number of bits) in phase 2.
		0	Receive word length is 8 bits.
		1h	Receive word length is 12 bits.
		2h	Receive word length is 16 bits.
		3h	Receive word length is 20 bits.
		4h	Receive word length is 24 bits.
		5h	Receive word length is 32 bits.
		6h-7h	Reserved
20-19	RCOMPAND	0-3h	Receive companding mode bit. Modes other than 00 are only enabled when RWDLEN1/2 bit is 000 (indicating 8-bit data).
		0	No companding, data transfer starts with MSB first.
		1h	No companding, 8-bit data transfer starts with LSB first.
		2h	Compand using µ-law for receive data.
		3h	Compand using A-law for receive data.
18	RFIG		Receive frame ignore bit.
		0	Receive frame-synchronization pulses after the first pulse restarts the transfer.
		1	Receive frame-synchronization pulses after the first pulse are ignored.

Table 30. Receive Control Register (RCR) Field Descriptions (continued)

Bit	Field	Value	Description
17-16	RDATDLY	0-3h	Receive data delay bit.
		0	0-bit data delay
		1h	1-bit data delay
		2h	2-bit data delay
		3h	Reserved
15	Reserved	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
14-8	RFRLEN1	0-7Fh	Specifies the receive frame length (number of words) in phase 1.
		0	1 word in phase 1
		1h	2 words in phase 1
		2h	3 words in phase 1
		...	...
		7Fh	128 words in phase 1
7-5	RWDLEN1	0-7h	Specifies the receive word length (number of bits) in phase 1.
		0	Receive word length is 8 bits.
		1h	Receive word length is 12 bits.
		2h	Receive word length is 16 bits.
		3h	Receive word length is 20 bits.
		4h	Receive word length is 24 bits.
		5h	Receive word length is 32 bits.
		6h-7h	Reserved
4	RWDREVRS		Receive 32-bit bit reversal enable bit.
		0	32-bit bit reversal is disabled.
		1	32-bit bit reversal is enabled. 32-bit data is received LSB first. RWDLEN1/2 bit should be set to 5h (32-bit operation); RCOMPAND bit should be set to 1h (transfer starts with LSB first); otherwise, operation is undefined.
3-0	Reserved	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.

3.5 Transmit Control Register (XCR)

The transmit control register (XCR) configures parameters of the transmit operations. The XCR is shown in Figure 52 and described in Table 31.

Figure 52. Transmit Control Register (XCR)

31	30 24	23 21	20 19	18	17 16
XPHASE	XFRLEN2	XWDLEN2	XCOMPAND	XFIG	XDATDLY
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

15	14 8	7 5	4	3 0
Reserved	XFRLEN1	XWDLEN1	XWDREVRS	Reserved
R-0	R/W-0	R/W-0	R/W-0	R-0

LEGEND: R = Read only; R/ W = Read/Write; -n = value after reset

Table 31. Transmit Control Register (XCR) Field Descriptions

Bit	Field	Value	Description
31	XPHASE		Transmit phases bit.
		0	Single-phase frame
		1	Dual-phase frame
30-24	XFRLEN2	0-7Fh	Specifies the transmit frame length (number of words) in phase 2.
		0	1 word in phase 2
		1h	2 words in phase 2
		2h	3 words in phase 2
		...	...
		7Fh	128 words in phase 2
23-21	XWDLEN2	0-7h	Specifies the transmit word length (number of bits) in phase 2.
		0	Transmit word length is 8 bits.
		1h	Transmit word length is 12 bits.
		2h	Transmit word length is 16 bits.
		3h	Transmit word length is 20 bits.
		4h	Transmit word length is 24 bits.
		5h	Transmit word length is 32 bits.
		6h-7h	Reserved
20-19	XCOMPAND	0-3h	Transmit companding mode bit. Modes other than 00 are only enabled when XWDLEN1/2 bit is 000 (indicating 8-bit data).
		0	No companding, data transfer starts with MSB first.
		1h	No companding, 8-bit data transfer starts with LSB first.
		2h	Compand using μ-law for transmit data.
		3h	Compand using A-law for transmit data.
18	XFIG		Transmit frame ignore bit.
		0	Transmit frame-synchronization pulses after the first pulse restarts the transfer.
		1	Transmit frame-synchronization pulses after the first pulse are ignored.
17-16	XDATDLY	0-3h	Transmit data delay bit.
		0	0-bit data delay
		1h	1-bit data delay
		2h	2-bit data delay
		3h	Reserved
15	Reserved	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.

Table 31. Transmit Control Register (XCR) Field Descriptions (continued)

Bit	Field	Value	Description
14-8	XFRLEN1	0-7Fh	Specifies the transmit frame length (number of words) in phase 1.
		0	1 word in phase 1
		1h	2 words in phase 1
		2h	3 words in phase 1
		...	...
		7Fh	128 words in phase 1
7-5	XWDLEN1	0-7h	Specifies the transmit word length (number of bits) in phase 1.
		0	Transmit word length is 8 bits.
		1h	Transmit word length is 12 bits.
		2h	Transmit word length is 16 bits.
		3h	Transmit word length is 20 bits.
		4h	Transmit word length is 24 bits.
		5h	Transmit word length is 32 bits.
		6h-7h	Reserved
4	XWDREVRS		Transmit 32-bit bit reversal feature enable bit.
		0	32-bit bit reversal is disabled.
		1	32-bit bit reversal is enabled. 32-bit data is transmitted LSB first. XWDLEN1/2 bit should be set to 5h (32-bit operation); XCOMPAND bit should be set to 1h (transfer starts with LSB first); otherwise, operation is undefined.
3-0	Reserved	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.

3.6 Sample Rate Generator Register (SRGR)

The sample rate generator register (SRGR) controls the operation of various features of the sample rate generator. The SRGR is shown in Figure 53 and described in Table 32.

Figure 53. Sample Rate Generator Register (SRGR)

31	30	29	28	27 ... 16
GSYNC	CLKSP	CLKSM	FSGM	FPER
R/W-0	R/W-0	R/W-1	R/W-0	R/W-0

15 ... 8	7 ... 0
FWID	CLKGDV
R/W-0	R/W-1

LEGEND: R/W = Read/Write; -n = value after reset

Table 32. Sample Rate Generator Register (SRGR) Field Descriptions

Bit	Field	Value	Description
31	GSYNC		Sample-rate generator clock synchronization bit is only used when the external clock (CLKS) drives the sample-rate generator clock (CLKSM = 0).
		0	The sample-rate generator clock (CLKG) is free running.
		1	The sample-rate generator clock (CLKG) is running; however, CLKG is resynchronized and frame-sync signal (FSG) is generated only after detecting the receive frame-synchronization signal (FSR). Also, frame period (FPER) is a don't care because the period is dictated by the external frame-sync pulse.
30	CLKSP		CLKS polarity clock edge select bit is only used when the external clock (CLKS) drives the sample-rate generator clock (CLKSM = 0).
		0	Rising edge of CLKS generates CLKG and FSG.
		1	Falling edge of CLKS generates CLKG and FSG.
29	CLKSM		Sample rate generator input clock mode bit. The sample rate generator can accept an input clock signal and divide it down according to CLKGDV to produce an output clock signal, CLKG. The frequency of CLKG is: *CLKG frequency = Input clock frequency/(CLKGDV + 1)* CLKSM is used in conjunction with the SCLKME bit in the serial port control register (SPCR) to determine the source for the input clock. A DSP reset selects the McBSP internal input clock as the input clock and forces the CLKG frequency to 1/2 the McBSP internal input clock frequency.
		0	The input clock for the sample rate generator is taken from the CLKR pin when selected as shown below (when the SCLKME bit in SPCR = 1): **CLKSM** / **SCLKME** / **Input Clock for Sample Rate Generator** 0 / 0 / Signal on CLKS pin 0 / 1 / Signal on CLKR pin
		1	The input clock for the sample rate generator is taken from the McBSP internal input clock or from the CLKX pin, depending on the value of the SCLKME bit in SPCR: **CLKSM** / **SCLKME** / **Input Clock for Sample Rate Generator** 1 / 0 / McBSP internal input clock 1 / 1 / Signal on CLKX pin
28	FSGM		Sample-rate generator transmit frame-synchronization mode bit is only used when FSXM = 1 in PCR.
		0	Transmit frame-sync signal (FSX) is generated on every DXR-to-XSR copy. When FSGM = 0, FWID bit and FPER bit are ignored.
		1	Transmit frame-sync signal (FSX) is driven by the sample-rate generator frame-sync signal (FSG).
27-16	FPER	0-FFFh	Frame period value plus 1 specifies when the next frame-sync signal becomes active. Range is 1 to 4096 sample-rate generator clock (CLKG) periods.
15-8	FWID	0-FFh	Frame width value plus 1 specifies the width of the frame-sync pulse (FSG) during its active period.

Table 32. Sample Rate Generator Register (SRGR) Field Descriptions (continued)

Bit	Field	Value	Description
7-0	CLKGDV	0-FFh	Sample-rate generator clock (CLKG) divider value is used as the divide-down number to generate the required sample-rate generator clock frequency.

3.7 Multichannel Control Register (MCR)

The multichannel control register (MCR) has control and status bits for multichannel selection operation in the receiver (with an R prefix) and the same type of bits for the transmitter (with an X prefix). The MCR is shown in Figure 54 and described in Table 33. This I/O-mapped register enables you to:

- Enable all channels or only selected channels for reception (RMCM).
- Choose which channels are enabled/disabled and masked/unmasked for transmission (XMCM).
- Specify whether two partitions (32 channels at a time) or eight partitions (128 channels at a time) can be used (RMCME for reception, XMCME for transmission).
- Assign blocks of 16 channels to partitions A and B when the 2-partition mode is selected (RPABLK and RPBBLK for reception, XPABLK and XPBBLK for transmission).
- Determine which block of 16 channels is currently involved in a data transfer (RCBLK for reception, XCBLK for transmission).

Figure 54. Multichannel Control Registers (MCR)

31	26	25	24	23	22	21	20	18	17	16
Reserved		XMCME	XPBBLK		XPABLK		XCBLK		XMCM	
R-0		R/W-0	R/W-0		R/W-0		R-0		R/W-0	

15	10	9	8	7	6	5	4	2	1	0
Reserved		RMCME	RPBBLK		RPABLK		RCBLK		Reserved	RMCM
R-0		R/W-0	R/W-0		R/W-0		R-0		R-0	R/W-0

LEGEND: R = Read only; R/ W = Read/Write; -n = value after reset

Table 33. Multichannel Control Register (MCR) Field Descriptions

Bit	Field	Value	Description
31-26	Reserved	0	Reserved.
25	XMCME		Transmit multichannel partition mode bit. XMCME is only applicable if channels can be individually disabled/enabled or masked/unmasked for transmission (XMCM is nonzero). XMCME determines whether only 32 channels or all 128 channels are to be individually selectable.
		0	2-partition mode. Only partitions A and B are used. You can control up to 32 channels in the transmit multichannel selection mode selected with the XMCM bit. **If XMCM = 1 or 2h:** assign 16 channels to partition A with the XPABLK bit and assign 16 channels to partition B with the XPBBLK bit. **If XMCM = 3h:** (for symmetric transmission and reception), assign 16 channels to receive partition A with the RPABLK bit and assign 16 channels to receive partition B with the RPBBLK bit.
		1	8-partition mode. All partitions (A through H) are used. You can control up to 128 channels in the transmit multichannel selection mode selected with the XMCM bit. You control the channels with the appropriate enhanced transmit channel enable register (XCERE*n*): XCERE0: Channels 0 through 31 XCERE1: Channels 32 through 63 XCERE2: Channels 64 through 95 XCERE3: Channels 96 through 127

Table 33. Multichannel Control Register (MCR) Field Descriptions (continued)

Bit	Field	Value	Description
24-23	XPBBLK	0-3h	Transmit partition B block bit. XPBBLK is only applicable if channels can be individually disabled/enabled and masked/unmasked (XMCM is nonzero) and the 2-partition mode is selected (XMCME = 0). Under these conditions, the McBSP transmitter can transmit or withhold data in any of the 32 channels that are assigned to partitions A and B of the transmitter. The 128 transmit channels of the McBSP are divided equally among 8 blocks (0 through 7). When XPBBLK is applicable, use the XPBBLK bit to assign one of the odd-numbered blocks (1, 3, 5, or 7) to partition B; use the XPABLK bit to assign one of the even-numbered blocks (0, 2, 4, or 6) to partition A. If you want to use more than 32 channels, you can change block assignments dynamically. You can assign a new block to one partition while the transmitter is handling activity in the other partition. For example, while the block in partition A is active, you can change which block is assigned to partition B. The XCBLK bit is regularly updated to indicate which block is active. When XMCM = 3h (for symmetric transmission and reception), the transmitter uses the receive block bits (RPABLK and RPBBLK) rather than the transmit block bits (XPABLK and XPBBLK).
		0	Block 1: channels 16 through 31
		1h	Block 3: channels 48 through 63
		2h	Block 5: channels 80 through 95
		3h	Block 7: channels 112 through 127
22-21	XPABLK	0-3h	Transmit partition A block bit. XPABLK is only applicable if channels can be individually disabled/enabled and masked/unmasked (XMCM is nonzero) and the 2-partition mode is selected (XMCME = 0). Under these conditions, the McBSP transmitter can transmit or withhold data in any of the 32 channels that are assigned to partitions A and B of the transmitter. See the XPBBLK bit description for more information about assigning blocks to partitions A and B.
		0	Block 0: channels 0 through 15
		1h	Block 2: channels 32 through 47
		2h	Block 4: channels 64 through 79
		3h	Block 6: channels 96 through 111
20-18	XCBLK	0-7h	Transmit current block indicator. XCBLK indicates which block of 16 channels is involved in the current McBSP transmission.
		0	Block 0: channels 0 through 15
		1h	Block 1: channels 16 through 31
		2h	Block 2: channels 32 through 47
		3h	Block 3: channels 48 through 63
		4h	Block 4: channels 64 through 79
		5h	Block 5: channels 80 through 95
		6h	Block 6: channels 96 through 111
		7h	Block 7: channels 112 through 127

Table 33. Multichannel Control Register (MCR) Field Descriptions (continued)

Bit	Field	Value	Description
17-16	XMCM	0-3h	Transmit multichannel selection mode bit. XMCM determines whether all channels or only selected channels are enabled and unmasked for transmission.
		0	Transmit multichannel selection is off. All channels are enabled and unmasked. No channels can be disabled or masked.
		1h	All channels are disabled unless they are selected in the appropriate enhanced transmit channel enable register (XCERE*n*). If enabled, a channel in this mode is also unmasked.
			The XMCME bit determines whether 32 channels or 128 channels are selectable in XCERE*n*.
		2h	All channels are enabled, but they are masked unless they are selected in the appropriate enhanced transmit channel enable register (XCERE*n*).
			The XMCME bit determines whether 32 channels or 128 channels are selectable in XCERE*n*.
		3h	This mode is used for symmetric transmission and reception.
			All channels are disabled for transmission unless they are enabled for reception in the appropriate enhanced receive channel enable register (RCERE*n*). Once enabled, they are masked unless they are also selected in the appropriate enhanced transmit channel enable register (XCERE*n*).
			The XMCME bit determines whether 32 channels or 128 channels are selectable in RCERE*n* and XCERE*n*.
15-10	Reserved	0	Reserved.
9	RMCME		Receive multichannel partition mode bit. RMCME is only applicable if channels can be individually disabled/enabled for reception (RMCM = 1). RMCME determines whether only 32 channels or all 128 channels are to be individually selectable.
		0	2-partition mode. Only partitions A and B are used. You can control up to 32 channels in the receive multichannel selection mode (RMCM = 1).
			Assign 16 channels to partition A with the RPABLK bit and assign 16 channels to partition B with the RPBBLK bit.
			You control the channels with the enhanced receive channel enable register partition A/B (RCERE0).
		1	You can control up to 128 channels in the receive multichannel selection mode.
			You control the channels with the appropriate enhanced receive channel enable register (RCERE*n*):
			RCERE0: Channels 0 through 31
			RCERE1: Channels 32 through 63
			RCERE2: Channels 64 through 95
			RCERE3: Channels 96 through 127
8-7	RPBBLK	0-3h	Receive partition B block bit.
			RPBBLK is only applicable if channels can be individually disabled/enabled (RMCM = 1) and the 2-partition mode is selected (RMCME = 0). Under these conditions, the McBSP receiver can accept or ignore data in any of the 32 channels that are assigned to partitions A and B of the receiver.
			The 128 receive channels of the McBSP are divided equally among 8 blocks (0 through 7). When RPBBLK is applicable, use the RPBBLK bit to assign one of the odd-numbered blocks (1, 3, 5, or 7) to partition B; use the RPABLK bit to assign one of the even-numbered blocks (0, 2, 4, or 6) to partition A.
			If you want to use more than 32 channels, you can change block assignments dynamically. You can assign a new block to one partition while the receiver is handling activity in the other partition. For example, while the block in partition A is active, you can change which block is assigned to partition B. The RCBLK bit is regularly updated to indicate which block is active.
			When XMCM = 3h (for symmetric transmission and reception), the transmitter uses the receive block bits (RPABLK and RPBBLK) rather than the transmit block bits (XPABLK and XPBBLK).
		0	Block 1: channels 16 through 31
		1h	Block 3: channels 48 through 63
		2h	Block 5: channels 80 through 95
		3h	Block 7: channels 112 through 127

Table 33. Multichannel Control Register (MCR) Field Descriptions (continued)

Bit	Field	Value	Description
6-5	RPABLK	0-3h	Receive partition A block bit. RPABLK is only applicable if channels can be individually disabled/enabled (RMCM = 1) and the 2-partition mode is selected (RMCME = 0). Under these conditions, the McBSP receiver can accept or ignore data in any of the 32 channels that are assigned to partitions A and B of the receiver. See the RPBBLK bit description for more information about assigning blocks to partitions A and B.
		0	Block 0: channels 0 through 15
		1h	Block 2: channels 32 through 47
		2h	Block 4: channels 64 through 79
		3h	Block 6: channels 96 through 111
4-2	RCBLK	0-7h	Receive current block indicator. RCBLK indicates which block of 16 channels is involved in the current McBSP reception.
		0	Block 0: channels 0 through 15
		1h	Block 1: channels 16 through 31
		2h	Block 2: channels 32 through 47
		3h	Block 3: channels 48 through 63
		4h	Block 4: channels 64 through 79
		5h	Block 5: channels 80 through 95
		6h	Block 6: channels 96 through 111
		7h	Block 7: channels 112 through 127
1	Reserved	0	Reserved.
0	RMCM		Receive multichannel selection mode bit. RMCM determines whether all channels or only selected channels are enabled for reception.
		0	All 128 channels are enabled.
		1	Multichannel selection mode. Channels can be individually enabled or disabled. The only channels enabled are those selected in the appropriate enhanced receive channel enable register (RCERE*n*). The way channels are assigned to RCERE*n* depends on the number of receive channel partitions (2 or 8), as defined by the RMCME bit.

3.8 Enhanced Receive Channel Enable Registers (RCERE0-RCERE3)

The enhanced receive channel enable register (RCERE*n*) is shown in Figure 55 and described in Table 34. The RCERE*n* is used to enable any of 128 elements for receive. RCERE0 is the only register used in normal mode (up to 32 channels can be selected in partitions A and B, RMCME = XMCME = 0 in MCR). RCERE0-RCERE3 are used when in enhanced mode (up to 128 channels can be selected in all partitions, RMCME = XMCME = 1 in MCR).

The receive multichannel partition mode (RMCME) bit in the multichannel control register (MCR) is only applicable if channels can be individually disabled/enabled for reception (RMCM = 1). The RMCME bit determines whether only 32 channels or all 128 channels are to be individually selectable:

- **When RMCME = 0:** Only partitions A and B are used. RCERE0 is used to enable any of the 32 elements for a receive. Of the 32 elements, 16 channels belong to a subframe in partition A and 16 channels belong to a subframe in partition B. The RCE0-RCE15 bits enable elements within the 16-channel elements in partition A and the RCE16-RCE31 bits enable elements within the 16-channel elements in partition B. The 16 channels in partition A are assigned with the RPABLK bit in MCR and the 16 channels in partition B are assigned with the RPBBLK bit in MCR.
- **When RMCME = 1:** All partitions are used. RCERE0 is used to enable any of the 32 elements in channels 0 through 31 for a receive. Of the 32 elements, channels 0 to 15 belong to a subframe in partition A and channels 16 to 31 belong to a subframe in partition B. The RCE0-RCE15 bits enable elements within the 16-channel elements in partition A and the RCE16-RCE31 bits enable elements within the 16-channel elements in partition B.

Section 3.8.1 shows the 128 channels in a multichannel data stream and their corresponding enable bits in RCERE*n*.

Figure 55. Enhanced Receive Channel Enable Register *n* (RCERE*n*)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RCE31	RCE30	RCE29	RCE28	RCE27	RCE26	RCE25	RCE24	RCE23	RCE22	RCE21	RCE20	RCE19	RCE18	RCE17	RCE16
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RCE15	RCE14	RCE13	RCE12	RCE11	RCE10	RCE9	RCE8	RCE7	RCE6	RCE5	RCE4	RCE3	RCE2	RCE1	RCE0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; -n = value afer reset

Table 34. Enhanced Receive Channel Enable Register *n* (RCERE*n*) Field Descriptions

Bit	Field	Value	Description
31-0	RCE*n*		Receive channel enable bit. For receive multichannel selection mode (RMCM = 1 in MCR).
		0	Disable the channel that is mapped to RCE*n*.
		1	Enable the channel that is mapped to RCE*n*.

3.8.1 RCERE*n* Used in the Receive Multichannel Selection Mode

For multichannel selection operation, the assignment of channels to the enhanced receive channel enable register (RCERE*n*) depends on whether 32 or 128 channels are individually selectable, as defined by the RMCME bit in the multichannel control register (MCR). For each of these two cases, Table 35 shows which block of channels is assigned to each RCERE*n* used. For each RCERE*n*, Table 35 shows which channel is assigned to each of the bits.

Table 35. Use of the Receive Channel Enable Registers

Number of selectable channels	Block Assignments		Channel Assignments	
	RCERE*n*	Block assigned[1]	Bit in RCERE*n*	Channel assigned[1]
32	RCERE0	Channels *n* to (*n* + 15)	RCE0	Channel *n*
(RMCME = 0)		The block of channels (0, 2, 4, or 6) is selected with the RPABLK bit in MCR	RCE1	Channel (*n* + 1)
			...	...
			RCE15	Channel (*n* + 15)
		Channels *m* to (*m* + 15)	RCE16	Channel *m*
		The block of channels (1, 3, 5, or 7) is selected with the RPBBLK bit in MCR	RCE17	Channel (*m* + 1)
			...	...
			RCE31	Channel (*m* + 15)
128	RCERE0	Block 0	RCE0	Channel 0
(RMCME = 1)			...	...
			RCE15	Channel 15
		Block 1	RCE16	Channel 16
			...	...
			RCE31	Channel 31
	RCERE1	Block 2	RCE0	Channel 32
			...	...
			RCE15	Channel 47
		Block 3	RCE16	Channel 48
			...	...
			RCE31	Channel 63
	RCERE2	Block 4	RCE0	Channel 64
			...	...
			RCE15	Channel 79
		Block 5	RCE16	Channel 80
			...	...
			RCE31	Channel 95
	RCERE3	Block 6	RCE0	Channel 96
			...	...
			RCE15	Channel 111
		Block 7	RCE16	Channel 112
			...	...
			RCE31	Channel 127

(1) *n* is any even-numbered block 0, 2, 4, or 6. *m* is any odd-numbered block 1, 3, 5, or 7.

3.9 Enhanced Transmit Channel Enable Registers (XCERE0-XCERE3)

The enhanced transmit channel enable register (XCERE*n*) is shown in Figure 56 and described in Table 36. The XCERE*n* is used to enable any of 128 elements for transmit. XCERE0 is the only register used in normal mode (up to 32 channels can be selected in partitions A and B, RMCME = XMCME = 0 in MCR). XCERE0-XCERE3 are used when in enhanced mode (up to 128 channels can be selected in all partitions, RMCME = XMCME = 1 in MCR).

The transmit multichannel partition mode (XMCME) bit in the multichannel control register (MCR) is only applicable if channels can be individually disabled/enabled or masked/unmasked for transmission (XMCM is nonzero). The XMCME bit determines whether only 32 channels or all 128 channels are to be individually selectable:

- **When XMCME = 0:** Only partitions A and B are used. XCERE0 is used to enable any of the 32 elements for a transmit. Of the 32 elements, 16 channels belong to a subframe in partition A and 16 channels belong to a subframe in partition B. The XCE0-XCE15 bits enable elements within the 16-channel elements in partition A and the XCE16-XCE31 bits enable elements within the 16-channel elements in partition B. You can control up to 32 channels in the transmit multichannel selection mode selected with the XMCM bit in MCR.
- **When XMCME = 1:** All partitions are used. XCERE0 is used to enable any of the 32 elements in channels 0 through 31 for a transmit. Of the 32 elements, channels 0 to 15 belong to a subframe in partition A and channels 16 to 31 belong to a subframe in partition B. The XCE0-XCE15 bits enable elements within the 16-channel elements in partition A and the XCE16-XCE31 bits enable elements within the 16-channel elements in partition B.

Section 3.9.1 shows the 128 channels in a multichannel data stream and their corresponding enable bits in XCERE*n*.

Figure 56. Enhanced Transmit Channel Enable Register *n* (XCERE*n*)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
XCE31	XCE30	XCE29	XCE28	XCE27	XCE26	XCE25	XCE24	XCE23	XCE22	XCE21	XCE20	XCE19	XCE18	XCE17	XCE16
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
XCE15	XCE14	XCE13	XCE12	XCE11	XCE10	XCE9	XCE8	XCE7	XCE6	XCE5	XCE4	XCE3	XCE2	XCE1	XCE0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; -n = value afer reset

Table 36. Enhanced Transmit Channel Enable Register *n* (XCERE*n*) Field Descriptions

Bit	Field	Value	Description
31-0	XCE*n*		Transmit channel enable bit. The role of this bit depends on which transmit multichannel mode is selected with the XMCM bit in MCR:
			When XMCM = 1h (all channels disabled unless selected):
		0	Disable and mask the channel that is mapped to XCE*n*.
		1	Enable and unmask the channel that is mapped to XCE*n*.
			When XMCM = 2h (all channels enabled but masked unless selected):
		0	Mask the channel that is mapped to XCE*n*.
		1	Unmask the channel that is mapped to XCE*n*.
			When XMCM = 3h (all channels masked unless selected):
		0	Mask the channel that is mapped to XCE*n*. Even if this channel is enabled by the corresponding enhanced receive channel enable bit, this channel's data cannot appear on the DX pin.
		1	Unmask the channel that is mapped to XCE*n*. Even if this channel is also enabled by the corresponding enhanced receive channel enable bit, full transmission can occur.

3.9.1 XCER*En* Used in the Transmit Multichannel Selection Mode

For multichannel selection operation, the assignment of channels to the enhanced transmit channel enable register (XCERE*n*) depends on whether 32 or 128 channels are individually selectable, as defined by the XMCME bit in the multichannel control register (MCR). For each of these two cases, Table 37 shows which block of channels is assigned to each XCERE*n* used. For each XCERE*n*, Table 37 shows which channel is assigned to each of the bits.

When XMCM = 3h (symmetric transmission and reception), the transmitter uses the enhanced receive channel enable register (RCERE*n*) to enable channels and uses XCERE*n* to unmask channels for transmission.

Table 37. Use of the Transmit Channel Enable Registers

	Block Assignments		Channel Assignments	
Number of selectable channels	**XCERE*n***	**Block assigned[(1)]**	**Bit in XCERE*n***	**Channel assigned[(1)]**
32	XCERE0	Channels *n* to (*n* + 15)	XCE0	Channel *n*
(XMCME = 0)		When XMCM = 1h or 2h, the block of channels (0, 2, 4, or 6) is selected with the XPABLK bit in MCR. When XMCM = 3h, the block of channels (0, 2, 4, or 6) is selected with the RPABLK bit in MCR.	XCE1	Channel (*n* + 1)
			...	...
			XCE15	Channel (*n* + 15)
		Channels *m* to (*m* + 15)	XCE16	Channel *m*
		When XMCM = 1h or 2h, the block of channels (1, 3, 5, or 7) is selected with the XPBBLK bit in MCR. When XMCM = 3h, the block of channels (1, 3, 5, or 7) is selected with the RPBBLK bit in MCR.	XCE17	Channel (*m* + 1)
			...	...
			XCE31	Channel (*m* + 15)
128	XCERE0	Block 0	XCE0	Channel 0
(XMCME = 1)			...	...
			XCE15	Channel 15
		Block 1	XCE16	Channel 16
			...	...
			XCE31	Channel 31
	XCERE1	Block 2	XCE0	Channel 32
			...	...
			XCE15	Channel 47
		Block 3	XCE16	Channel 48
			...	...
			XCE31	Channel 63
	XCERE2	Block 4	XCE0	Channel 64
			...	...
			XCE15	Channel 79
		Block 5	XCE16	Channel 80
			...	...
			XCE31	Channel 95

(1) *n* is any even-numbered block 0, 2, 4, or 6. *m* is any odd-numbered block 1, 3, 5, or 7.

Table 37. Use of the Transmit Channel Enable Registers (continued)

Number of selectable channels	Block Assignments		Channel Assignments	
	XCERE*n*	Block assigned[1]	Bit in XCERE*n*	Channel assigned[1]
	XCERE3	Block 6	XCE0	Channel 96
			...	...
			XCE15	Channel 111
		Block 7	XCE16	Channel 112
			...	...
			XCE31	Channel 127

3.10 Pin Control Register (PCR)

The serial port is configured via the serial port control register (SPCR) and the pin control register (PCR). The PCR contains McBSP status control bits. The PCR is shown in Figure 57 and described in Table 38.

Figure 57. Pin Control Register (PCR)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
R-0															

15	14	13	12	11	10	9	8
Reserved		Reserved[1]	Reserved[1]	FSXM	FSRM	CLKXM	CLKRM
R-0		R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

7	6	5	4	3	2	1	0
SCLKME	Reserved[1]	Reserved	Reserved	FSXP	FSRP	CLKXP	CLKRP
R/W-0	R-0	R/W-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R = Read only; R/W = Rear/Write; -n = value afer reset

[1] If writing to this field, always write the default value of 0 to ensure proper McBSP operation.

Table 38. Pin Control Register (PCR) Field Descriptions

Bit	Field	Value	Description
31-14	Reserved	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
13-12	Reserved	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect. If writing to this field, always write the default value of 0 to ensure proper McBSP operation.
11	FSXM		Transmit frame-synchronization mode bit.
		0	Frame-synchronization signal is derived from an external source.
		1	Frame-synchronization signal is determined by FSGM bit in SRGR.
10	FSRM		Receive frame-synchronization mode bit.
		0	Frame-synchronization signal is derived from an external source. FSR is an input pin.
		1	Frame-synchronization signal is generated internally by the sample-rate generator. FSR is an output pin.

Table 38. Pin Control Register (PCR) Field Descriptions (continued)

Bit	Field	Value	Description
9	CLKXM		Transmit clock mode bit.
			CLKSTP bit in SPCR is 0:
		0	CLKX is an input pin and is driven by an external clock.
		1	CLKX is an output pin and is driven by the internal sample-rate generator.
			In SPI mode when CLKSTP bit in SPCR is a non-zero value:
		0	McBSP is a slave and clock (CLKX) is driven by the SPI master in the system. CLKR is internally driven by CLKX.
		1	McBSP is a master and generates the clock (CLKX) to drive its receive clock (CLKR) and the shift clock of the SPI-compliant slaves in the system.
8	CLKRM		Receive clock mode bit.
			Digital loop back mode is disabled (DLB = 0 in SPCR):
		0	CLKR is an input pin and is driven by an external clock.
		1	CLKR is an output pin and is driven by the internal sample-rate generator.
			Digital loop back mode is enabled (DLB = 1 in SPCR):
		0	Receive clock (not the CLKR pin) is driven by transmit clock (CLKX) that is based on CLKXM bit. CLKR pin is in high-impedance state.
		1	CLKR is an output pin and is driven by the transmit clock. The transmit clock is based on CLKXM bit.
7	SCLKME		Sample rate generator input clock mode bit. The sample rate generator can produce a clock signal, CLKG. The frequency of CLKG is:
			CLKG frequency = Input clock frequency/(CLKGDV + 1)
			SCLKME is used in conjunction with the CLKSM bit in the sample rate generator register (SRGR) to select the input clock.
			A DSP reset selects the McBSP internal input clock as the input clock and forces the CLKG frequency to 1/2 the McBSP internal input clock frequency.
		0	The input clock for the sample rate generator is taken from the McBSP internal input clock as shown below (when the CLKSM bit in SRGR = 1):
			SCLKME / **CLKSM** / **Input Clock for Sample Rate Generator** 0 / 0 / Signal on CLKS pin 0 / 1 / McBSP internal input clock
		1	The input clock for the sample rate generator is taken from the CLKR pin or from the CLKX pin, depending on the value of the CLKSM bit in SRGR:
			SCLKME / **CLKSM** / **Input Clock for Sample Rate Generator** 1 / 0 / Signal on CLKR pin 1 / 1 / Signal on CLKX pin
6	Reserved	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect. If writing to this field, always write the default value of 0 to ensure proper McBSP operation.
5-4	Reserved	0	Reserved. The reserved bit location is always read as 0. A value written to this field has no effect.
3	FSXP		Transmit frame-synchronization polarity bit.
		0	Transmit frame-synchronization pulse is active high.
		1	Transmit frame-synchronization pulse is active low.
2	FSRP		Receive frame-synchronization polarity bit.
		0	Receive frame-synchronization pulse is active high.
		1	Receive frame-synchronization pulse is active low.
1	CLKXP		Transmit clock polarity bit.
		0	Transmit data sampled on rising edge of CLKX.
		1	Transmit data sampled on falling edge of CLKX.
0	CLKRP		Receive clock polarity bit.
		0	Receive data sampled on falling edge of CLKR.
		1	Receive data sampled on rising edge of CLKR.

Appendix A Revision History

Table A-1 lists the changes made since the previous version of this document.

Table A-1. Document Revision History

Reference	Additions/Modifications/Deletions
Table 30	Changed Description of RFRLEN2 bit.
	Changed Description of RFRLEN1 bit.
Table 31	Changed Description of XFRLEN2 bit.
	Changed Description of XFRLEN1 bit.

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

TI products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in such safety-critical applications.

TI products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are specifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military specifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at the Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are designated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated products in automotive applications, TI will not be responsible for any failure to meet such requirements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		**Applications**	
Amplifiers	amplifier.ti.com	Audio	www.ti.com/audio
Data Converters	dataconverter.ti.com	Automotive	www.ti.com/automotive
DSP	dsp.ti.com	Broadband	www.ti.com/broadband
Interface	interface.ti.com	Digital Control	www.ti.com/digitalcontrol
Logic	logic.ti.com	Military	www.ti.com/military
Power Mgmt	power.ti.com	Optical Networking	www.ti.com/opticalnetwork
Microcontrollers	microcontroller.ti.com	Security	www.ti.com/security
RFID	www.ti-rfid.com	Telephony	www.ti.com/telephony
Low Power Wireless	www.ti.com/lpw	Video & Imaging	www.ti.com/video
		Wireless	www.ti.com/wireless

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265