TMS320x281x DSP Multichannel Buffered Serial Port (McBSP) Reference Guide

Literature Number: SPRU061A
June 2003

TEXAS INSTRUMENTS

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 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.

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.

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
		Telephony	www.ti.com/telephony
		Video & Imaging	www.ti.com/video
		Wireless	www.ti.com/wireless

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

Contents

1 Overview 1-1

Introduces the multichannel buffered serial port (McBSP).

1.1 Introduction to the McBSP 1-2
1.1.1 Key Features of the McBSP 1-2
1.1.2 Block Diagram of the McBSP Module With FIFO 1-4
1.1.3 McBSP Signals 1-5
1.2 Register Summary 1-7
1.3 McBSP Operation 1-10
1.3.1 Data Transfer Process of a McBSP 1-10
1.3.2 Companding (Compressing and Expanding) Data 1-11
1.3.3 Clocking and Framing Data 1-14
1.3.4 Frame Phases 1-18
1.3.5 McBSP Reception 1-21
1.3.6 McBSP Transmission 1-22
1.3.7 Interrupts and FIFO Events Generated by a McBSP 1-24
1.4 Sample Rate Generator of the McBSP 1-26
1.4.1 Clock Generation in the Sample Rate Generator 1-28
1.4.2 Frame Sync Generation in the Sample Rate Generator 1-31
1.4.3 Synchronizing Sample Rate Generator Outputs to an External Clock 1-32
1.4.4 Reset and Initialization Procedure for the Sample Rate Generator 1-34
1.4.5 Sample Rate Generator Clocking Examples 1-35
1.5 McBSP Exception/Error Conditions 1-39
1.5.1 Overrun in the Receiver 1-40
1.5.2 Unexpected Receive Frame-Sync Pulse 1-41
1.5.3 Overwrite in the Transmitter 1-44
1.5.4 Underflow in the Transmitter 1-45
1.5.5 Unexpected Transmit Frame-Sync Pulse 1-47

2 Multichannel Selection Modes 2-1

Describes the process for using multichannel selection.

2.1 Channels, Blocks, and Partitions 2-2
2.1.1 Multichannel Selection 2-2
2.1.2 Configuring a Frame for Multichannel Selection 2-2
2.1.3 Using Two Partitions 2-3
2.1.4 Using Eight Partitions 2-5

2.1.5 Receive Multichannel Selection Mode 2-7
2.1.6 Transmit Multichannel Selection Modes 2-8
2.1.7 Disabling/Enabling Versus Masking/Unmasking 2-9
2.1.8 Activity on McBSP Pins for Different Values of XMCM 2-10
2.1.9 Using Interrupts Between Block Transfers 2-12
2.2 A-bis Mode 2-13
2.2.1 A-bis Mode Receive Operation 2-13
2.2.2 A-bis Mode Transmit Operation 2-14
2.3 SPI Protocol 2-15
2.3.1 Clock Stop Mode 2-15
2.3.2 Bits Used to Enable and Configure the Clock Stop Mode 2-16
2.3.3 Clock Stop Mode Timing Diagrams 2-17
2.3.4 Procedure for Configuring a McBSP for SPI Operation 2-19
2.3.5 McBSP as the SPI Master 2-20
2.3.6 McBSP as an SPI Slave 2-22

3 Configure the Receiver and Transmitter 3-1
Describes how to configure the receiver and transmitter.

3.1 Receiver Configuration 3-2
3.1.1 Programming the McBSP Registers for the Desired Receiver Operation 3-2
3.1.2 Resetting and Enabling the Receiver 3-3
3.1.3 Clock Stop Mode 3-5
3.1.4 Receive Multichannel Selection and A-bis Modes 3-6
3.1.5 Choose 1 or 2 Phases for the Receive Frame 3-6
3.1.6 Set the Receive Companding Mode 3-9
3.1.7 Set the Receive Data Delay 3-11
3.1.8 Set the Receive Sign-Extension and Justification Mode 3-12
3.1.9 Set the Receive Interrupt Mode 3-14
3.1.10 Set the Receive Frame-Sync Mode 3-15
3.1.11 Set the Receive Frame-Sync Polarity 3-16
3.2 Transmitter Configuration 3-26
3.2.1 Programming the McBSP Registers for the Desired Transmitter Operation . 3-26
3.2.2 Resetting and Enabling the Transmitter 3-27
3.2.3 Set the Transmitter Pins to Operate as McBSP Pins 3-28
3.2.4 Enable/Disable the Digital Loopback Mode 3-28
3.2.5 Enable/Disable the Clock Stop Mode 3-29
3.2.6 Enable/Disable Transmit Multichannel Selection 3-30
3.2.7 Enable/Disable the A-bis Mode 3-31
3.2.8 Choose 1 or 2 Phases for the Transmit Frame 3-31
3.2.9 Set the Transmit Word Length(s) 3-31
3.2.10 Set the Transmit Frame Length 3-32
3.2.11 Set the Transmit Companding Mode 3-34
3.2.12 Set the Transmit Data Delay 3-36
3.2.13 Set the Transmit DXENA Mode 3-37

3.2.14 Set the Transmit Interrupt Mode 3-38
3.2.15 Set the Transmit Frame-Sync Mode 3-39
3.2.16 Set the Transmit Frame-Sync Polarity 3-40
3.2.17 Set the SRG Frame-Sync Period and Pulse Width 3-42
3.2.18 Set the Transmit Clock Mode 3-43
3.2.19 Set the SRG Clock Divide-Down Value 3-46
3.2.20 Set the SRG Clock Mode (Choose an Input Clock) 3-47

4 Emulation and Reset Considerations 4-1
Describes resetting and initializing the McBSP.

4.1 McBSP Emulation Mode 4-2
4.1.1 Resetting and Initializing a McBSP 4-3
4.2 Data Packing Examples 4-8
4.2.1 Data Packing Using Frame Length and Word Length 4-8
4.2.2 Data Packing Using Word Length and the Frame-Sync Ignore Function 4-9
4.3 GPIO Function 4-11

5 McBSP FIFO and Interrupts 5-1
Describes the FIFO interface logic.

5.1 McBSP FIFO Overview 5-2
5.2 McBSP Functionality and Limitation Under FIFO Mode 5-3
5.3 McBSP FIFO Operation 5-5
5.4 McBSP Receive Interrupt Generation 5-7
5.5 McBSP Transmit Interrupt Generation 5-9
5.5.1 FIFO Data Register Access Constraints 5-10
5.5.2 FIFO Error Flags 5-11
5.5.3 McBSP IDLE Mode 5-12
5.5.4 McBSP Reset Conditions 5-12
5.6 McBSP FIFO Register Descriptions 5-13

6 McBSP Registers 6-1
Describes the McBSP registers and bit descriptions.

6.1 Data Receive and Transmit Registers 6-2
6.1.1 Data Receive Registers (DRR2 and DRR1) 6-2
6.1.2 How Data Travels From the Data Receive (DR) Pin to the DRRs 6-2
6.1.3 Data Transmit Registers (DXR2 and DXR1) 6-3
6.1.4 How Data Travels From the DXRs to the Data Transmit (DX) Pin 6-3
6.2 Serial Port Control Registers (SPCR1 and SPCR2) 6-4
6.3 Receive Control Registers (RCR1 and RCR2) 6-8
6.4 Transmit Control Registers (XCR1 and XCR2) 6-11
6.5 Sample Rate Generator Registers (SRGR1 and SRGR2) 6-14
6.6 Multichannel Control Registers (MCR1 and MCR2) 6-17
6.7 Pin Control Register (PCR) 6-21
6.8 Receive Channel Enable Registers (RCERA – RCERH) 6-24

6.8.1 RCERs Used in the Receive Multichannel Selection Mode 6-26
6.8.2 RCERs Used in the A-bis Mode 6-27
6.9 Transmit Channel Enable Registers (XERA – XCERH) 6-29
6.9.1 XCERs Used in a Transmit Multichannel Selection Mode 6-30
6.9.2 XCERs Used in the A-bis Mode 6-32
6.10 Register Bit Summary 6-34

Figures

1–1. Block Diagram With FIFO Interface 1-4
1–2. McBSP Data Transfer Paths 1-10
1–3. Companding Processes 1-12
1–4. m-Law Transmit Data Companding Format 1-12
1–5. A-Law Transmit Data Companding Format 1-12
1–6. Two Methods by Which the McBSP Can Compand Internal Data 1-13
1–7. Clock Signal Control Waveform 1-14
1–8. 8-Bit Word Size Defined Waveform 1-15
1–9. One-word Frame Transfer 1-16
1–10. McBSP Operating at Maximum Packet Frequency 1-17
1–11. Single-Phase Frame for a McBSP Data Transfer 1-19
1–12. Dual-Phase Frame for a McBSP Data Transfer 1-19
1–13. Implementing the AC97 Standard With a Dual-Phase Frame 1-20
1–14. Timing of an AC97-Standard Data Transfer Near Frame Synchronization 1-20
1–15. McBSP Reception Physical Data Path 1-21
1–16. McBSP Reception Signal Activity 1-21
1–17. McBSP Transmission Physical Data Path 1-23
1–18. McBSP Transmission Signal Activity 1-23
1–19. Sample Rate Generator Clock Selection 1-26
1–20. Possible Inputs to the Sample Rate Generator and the Polarity Bits 1-29
1–21. CLKG Synchronization and FSG Generation When GSYNC = 1 and CLKGDV = 1 1-33
1–22. CLKG Synchronization and FSG Generation When GSYNC = 1 and CLKGDV = 3 . . . 1-34
1–23. ST-BUS and MVIP Clocking Example 1-36
1–24. Single-Rate Clock Example 1-37
1–25. Double-Rate Clock Example 1-38
1–26. Overrun in the McBSP Receiver 1-41
1–27. Overrun Prevented in the McBSP Receiver 1-41
1–28. Possible Responses to Receive Frame-Sync Pulses 1-42
1–29. An Unexpected Frame-Sync Pulse During a McBSP Reception 1-43
1–30. Proper Positioning of Frame-Sync Pulses 1-44
1–31. Data in the McBSP Transmitter Overwritten and Thus Not Transmitted 1-45
1–32. Underflow During McBSP Transmission 1-46
1–33. Underflow Prevented in the McBSP Transmitter 1-46
1–34. Possible Responses to Transmit Frame-Sync Pulses 1-47
1–35. An Unexpected Frame-Sync Pulse During a McBSP Transmission 1-48
1–36. Proper Positioning of Frame-Sync Pulses 1-49

2–1. Alternating Between the Channels of Partition A and the Channels of Partition B 2-4
2–2. Reassigning Channel Blocks Throughout a McBSP Data Transfer 2-5
2–3. McBSP Data Transfer in the 8-Partition Mode 2-7
2–4. Activity on McBSP Pins for the Possible Values of XMCM 2-11
2–5. A-bis Mode Receive Operation 2-13
2–6. A-bis Mode Transmit Operation 2-14
2–7. Typical SPI Interface 2-15
2–8. SPI Transfer With CLKSTP = 10b (no clock delay), CLKXP = 0, CLKRP = 0 2-18
2–9. SPI Transfer With CLKSTP = 11b (clock delay), CLKXP = 0, CLKRP = 1 2-18
2–10. SPI Transfer With CLKSTP = 10b (no clock delay), CLKXP = 1, CLKRP = 0 2-18
2–11. SPI Transfer With CLKSTP = 11b (clock delay), CLKXP = 1, CLKRP = 1 2-19
2–12. SPI Interface With McBSP as Master 2-20
2–13. SPI With McBSP Configured as Slave 2-22
3–1. Unexpected Frame-Sync Pulse With (R/X)FIG = 0 3-9
3–2. Unexpected Frame-Sync Pulse With (R/X)FIG = 1 3-9
3–3. Companding Processes for Reception and for Transmission 3-10
3–4. Range of Programmable Data Delay 3-11
3–5. 2-Bit Data Delay Used to Skip a Framing Bit 3-12
3–6. Data Clocked Externally Using a Rising Edge and Sampled by the McBSP Receiver on a Falling Edge 3-17
3–7. Frame of Period 16 CLKG Periods and Active Width of 2 CLKG Periods 3-19
3–8. Data Clocked Externally Using a Rising Edge and Sampled by the McBSP Receiver on a Falling Edge 3-22
3–9. Unexpected Frame-Sync Pulse With (R/X)FIG = 0 3-33
3–10. Unexpected Frame-Sync Pulse With (R/X)FIG = 1 3-34
3–11. Companding Processes for Reception and for Transmission 3-34
3–12. m-Law Transmit Data Companding Format 3-35
3–13. A-Law Transmit Data Companding Format 3-35
3–14. Range of Programmable Data Delay 3-36
3–15. 2-Bit Data Delay Used to Skip a Framing Bit 3-37
3–16. DX Delay When A-bis Mode is Off 3-38
3–17. DX Delays When A-bis Mode is On 3-38
3–18. Data Clocked Externally Using a Rising Edge and Sampled by the McBSP Receiver on a Falling Edge 3-42
3–19. Frame of Period 16 CLKG Periods and Active Width of 2 CLKG Periods 3-43
3–20. Data Clocked Externally Using a Rising Edge and Sampled by the McBSP Receiver on a Falling Edge 3-46
4–1. Four 8-Bit Data Words Transferred To/From the McBSP 4-8
4–2. One 32-Bit Data Word Transferred To/From the McBSP 4-9
4–3. 8-Bit Data Words Transferred at Maximum Packet Frequency 4-10
4–4. Configuring the Data Stream of 4–3 as a Continuous 32-Bit Word 4-10
5–1. Receive Interrupt Generation 5-7
5–2. Transmit Interrupt Generation 5-9
5–3. McBSP FIFO Transmit Register (MFFTX) 5-13
5–4. McBSP FIFO Receive Register (MFFRX) 5-14

5–5. McBSP FIFO Control Register (MFFCT) 5-15
5–6. McBSP FIFO Interrupt Register (MFFINT) 5-15
5–7. McBSP FIFO Status Register (MFFST) 5-16
6–1. Data Receive Registers (DRR2 and DRR1) 6-2
6–2. Data Transmit Registers (DXR2 and DXR1) 6-3
6–3. Serial Port Control 2 Register (SPCR2) 6-4
6–4. SPCR1 Register 6-6
6–5. Receive Control 2 Register (RCR2) 6-8
6–6. RCR1 Register 6-10
6–7. Transmit Control 2 Register (XCR2) 6-11
6–8. XCR1 Register 6-13
6–9. Sample Rate Generator 2 Register (SRGR2) 6-15
6–10. Sample Rate Generator 1 Register (SRGR1) 6-16
6–11. Multichannel Control 2 Register (MCR2) 6-17
6–12. Multichannel Control 1 Register (MCR1) 6-19
6–13. Pin Control Register (PCR) 6-21
6–14. Receive Channel Enable Register (RCERA/B) 6-24
6–15. RCER(A–G)–Receive Channel Enable Registers – A, C, E, G 6-25
6–16. RCER(B–H)–Receive Channel Enable Registers – B,D,F,H 6-25
6–17. Transmit Channel Enable Registers A. C. E. G (XCERA–XCERG) 6-29
6–18. Transmit Channel Enable Registers–B, D, F, H (XCERB–XCERH) 6-30

Tables

1–1. 28x Implementation Changes 1-3
1–2. McBSP Signal Summary 1-5
1–3. McBSP Register Summary 1-7
1–4. McBSP Register Bits That Determine the Number of Phases, Words, and Bits Per Frame 1-18
1–5. Interrupts and FIFO Events Generated by a McBSP 1-24
1–6. Sample Rate Generator Clock Options 1-27
1–7. Effects of DLB and CLKSTP on Clock Modes 1-28
1–8. Choosing an Input Clock for the Sample Rate Generator With the SCLKME and CLKSM Bits 1-29
1–9. Polarity Options for the Input to the Sample Rate Generator 1-30
2–1. Receive Channel Assignment and Control When Eight Receive Partitions Are Used . . . 2-6
2–2. Transmit Channel Assignment and Control When Eight Transmit Partitions Are Used . . 2-6
2–3. Selecting a Transmit Multichannel Selection Mode With the XMCM Bits 2-8
2–4. Bits Used to Enable and Configure the Clock Stop Mode 2-16
2–5. Effects of CLKSTP, CLKXP, and CLKRP on the Clock Scheme 2-17
2–6. Bit Values Required to Configure the McBSP as an SPI Master 2-21
2–7. Bit Values Required to Configure the McBSP as an SPI Slave 2-23
3–1. Reset State of Each McBSP Pin 3-4
3–2. Receive Signals Connected to Transmit Signals in Digital Loopback Mode 3-4
3–3. Effects of CLKSTP, CLKXP, and CLKRP on the Clock Scheme 3-5
3–4. How to Calculate the Length of the Receive Frame 3-8
3–5. Example: Use of RJUST Field With 12-Bit Data Value 0xABC 3-13
3–6. Example: Use of RJUST Field With 20-Bit Data Value 0xABCDE 3-13
3–7. Select Sources to Provide the Receive Frame-Synchronization Signal and the Effect on the FSR Pin 3-15
3–8. Select Sources to Provide the Receive Clock Signal and the Effect on the CLKR Pin 3-20
3–9. Reset State of Each McBSP Pin 3-28
3–10. Receive Signals Connected to Transmit Signals in Digital Loopback Mode 3-28
3–11. Effects of CLKSTP, CLKXP, and CLKRP on the Clock Scheme 3-30
3–12. How to Calculate Frame Length 3-32
3–13. How FSXM and FSGM Select the Source of Transmit Frame-Sync Pulses 3-39
3–14. How the CLKXM Bit Selects the Transmit Clock and the Corresponding Status of the CLKX Pin 3-44
4–1. McBSP Emulation Modes Selectable With the FREE and SOFT Bits of SPCR2 4-3
4–2. Reset State of Each McBSP Pin 4-3

5–1. McBSP FIFO Registers 5-2
5–2. McBSP Mode Selection 5-3
5–3. Receive Interrupt Sources and Signals in NonFIFO Mode and FIFO Mode 5-7
5–4. Transmit Interrupt Sources and Signals in NonFIFO Mode and FIFO Mode 5-9
5–5. Receive FIFO Read Order 5-10
5–6. Transmit FIFO Write Order 5-11
5–7. McBSP Error Flags 5-12
6–1. Use of the Receive Channel Enable Registers in theReceive Multichannel Selection Mode 6-26
6–2. Use of Receive Channel Enable Registers A and B in the A-bis Mode 6-28
6–3. Use of the Transmit Channel Enable Registers in a Transmit Multichannel Selection Mode 6-31
6–4. Use of Transmit Channel Enable Registers A and B in the A-bis Mode 6-33
6–5. Register Bit Summary (Base Address 0x00 7800) 6-34
6–6. FIFO Register Bit Descriptions (Base address 0x00 7800) 6-37

Chapter 1

Overview

The multichannel buffered serial port (McBSP) provides a direct serial interface between a digital signal processor (DSP) and other devices in a system. This guide describes the type of McBSP available on the 28x DSPs.

If you have used a McBSP in other TI DSPs, note that there are differences in this implementation, which are outlined in Table 1–1.

Topic **Page**

1.1 Introduction to the McBSP 1-2
1.2 Register Summary 1-7
1.3 McBSP Operation 1-9
1.4 Sample Rate Generator of the McBSP 1-25
1.5 McBSP Exception/Error Conditions 1-38

1.1 Introduction to the McBSP

The McBSP peripheral provides an interface between a 28x device and McBSP-compatible devices such as the VBAP, AIC, Combo Codecs. The external components attached to the McBSP can synchronously transmit/receive 8/16/32-bit serial data.

1.1.1 Key Features of the McBSP

The McBSP provides:

- ❏ Full-duplex communication
- ❏ Double-buffered transmission and triple-buffered reception, which allow a continuous data stream
- ❏ Independent clocking and framing for reception and for transmission
- ❏ 128 channels for transmission and for reception
- ❏ Multichannel selection modes that enable you to allow or block transfers in each of the channels
- ❏ DMA replaced with two 16-level, 32-bit FIFOs
- ❏ Support for A-bis mode
- ❏ Direct interface to industry-standard codecs, analog interface chips (AICs), and other serially connected A/D and D/A devices
- ❏ Support for external generation of clock signals and frame-synchronization (frame-sync) signals
- ❏ A programmable sample rate generator for internal generation and control of frame-sync signals
- ❏ Programmable internal clock and frame generation
- ❏ Programmable polarity for frame-synchronization and data clocks
- ❏ Support for SPI devices

- Support fractional T1/E1. 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 capability is provided.)
  - IIS-compliant devices
  - SPI devices
- A wide selection of data sizes: 8, 12, 16, 20, 24, and 32 bits
  Note: A value of the chosen data size is referred to as a *serial word* or *word* throughout the McBSP documentation. Elsewhere, *word* is used to describe a 16-bit value.
- The option of transmitting/receiving 8-bit data with the LSB or MSB first

The McBSP module in C28x devices is adapted from TI's family of McBSPs. Although it supports most of the McBSP applications, Table 1–1 lists some limitations to this implementation.

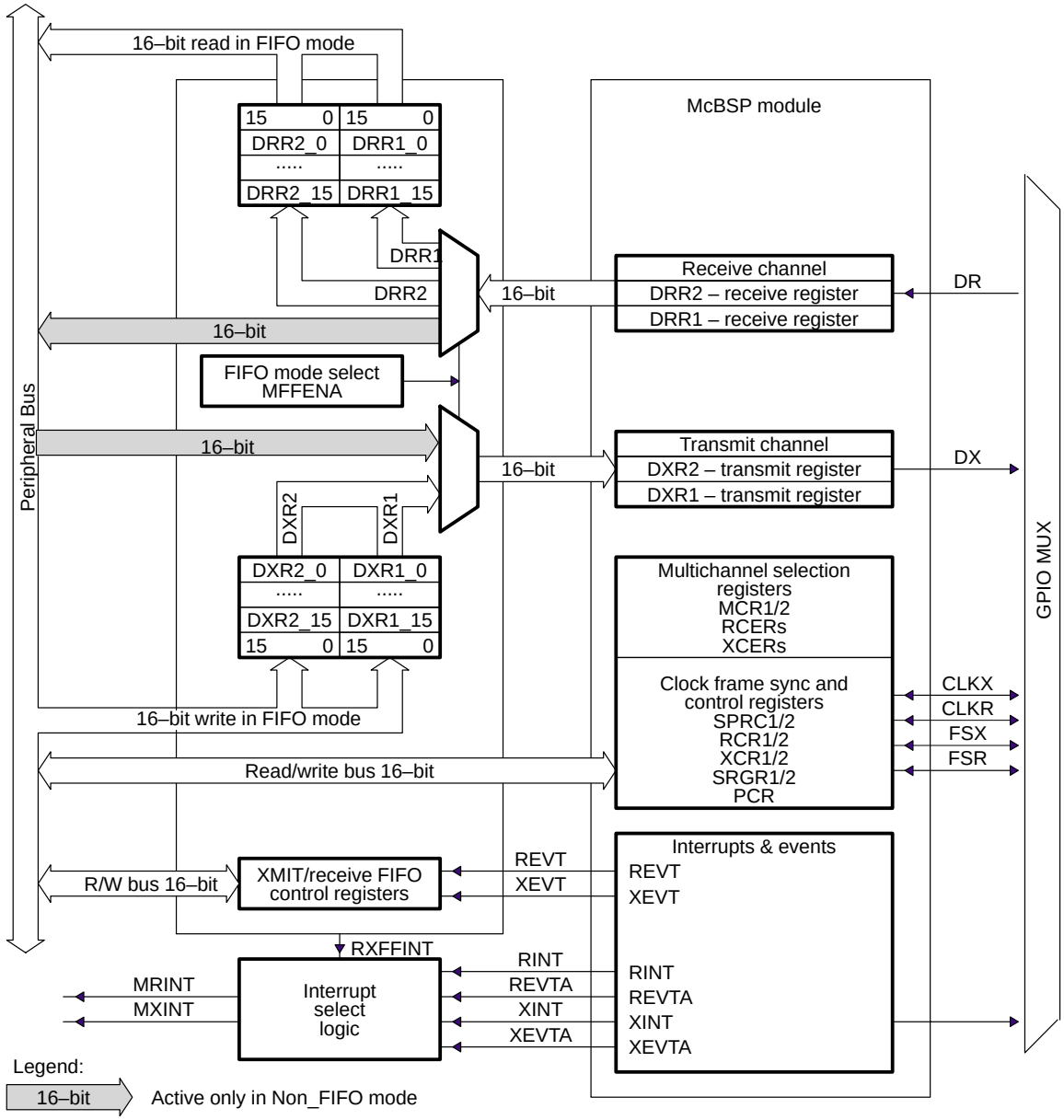
Table 1–1. 28x Implementation Changes

Features of TI's McBSP family	Implementation in 28x McBSP Module
DMA supports for data transfer	DMA replaced by two 32 X 16 level FIFOs
GPIO function on McBSP pins	Supported through the GPIO module available in 28x family.
Power down-mode using IDLE-EN bit in PCR register	The clock for the McBSP logic is controlled using bit 12 of the peripheral clock control register (PCLKCR).
CLKS as external shift clock	CLKS feature is not supported. CLKR/CLKX pin provide this feature.

1.1.2 Block Diagram of the McBSP Module With FIFO

The McBSP consists of a data-flow path and a control path connected to external devices by seven pins as shown in Figure 1–1.

Figure 1–1. Block Diagram With FIFO Interface



Data is communicated to devices interfaced with the McBSP via the data transmit (DX) pin for transmission and the data receive (DR) pin for reception. Control information in the form of clocking and frame synchronization is communicated via the following pins: CLKX (transmit clock), CLKR (receive clock), FSX (transmit frame sync), and FSR (receive frame sync).

DRR2, RBR2, RSR2, DXR2, and XSR2 are not used (written, read, or shifted) if the serial word length is 8 bits, 12 bits, or 16 bits. For larger word lengths, these registers are needed to hold the most significant bits.

The remaining registers are registers for controlling McBSP operation. Details about these registers are available in section 1.2 (page 1-7).

1.1.3 McBSP Signals

Table 1–2 describes the McBSP interface signals. Information on using these signals for general-purpose input/output is in section 4.3 on page 4-11.

Table 1–2. McBSP Signal Summary

Signal Name	Type †	Reset Status	Description
External Signals			**External Signals Names**
CLKX	I/O/Z	Input	Transmit clock
CLKR	I/O/Z	Input	Receive clock
DR	I	Input	Received serial data
DX	O/Z	High-z	Transmitted serial data
FSR	I/O/Z	Input	Receive frame synchronization
FSX	I/O/Z	Input	Transmit frame synchronization
CPU-Interrupt Signals			
MRINT			Receive interrupt to CPU or FIFO
MXINT			Transmit interrupt to CPU or FIFO
FIFO Events			
REVT			Receive synchronization event to FIFO
XEVT			Transmit synchronization event to FIFO

† I = input, O = output, Z = high impedance

Notes: 1) CLKS signal is not supported in this implementation.

Table 1–2. McBSP Signal Summary (Continued)

Signal Name	Type †	Reset Status	Description
REVTA			Receive synchronization in A-bis mode in the FIFO
XEVTA			Transmit synchronization in A-bis mode in the FIFO

† I = input, O = output, Z = high impedance

Notes: 1) CLKS signal is not supported in this implementation.

1.2 Register Summary

For each McBSP, the DSP contains the following registers. For the address of each register, see the data sheet for your device.

Table 1–3. McBSP Register Summary

Address Offset 0x000 xxh	16-Bit Access	Type R/W	Reset ValueHex	McBSP Communications Interface Register Description
Data Registers, Receive, Transmit				
–	–	–		McBSP Receive Buffer Register
–	–	–		McBSP Receive Shift Register
–	–	–		McBSP Transmit Shift Register
00	DRR2	R	0x0000	McBSP Data Receive Register2 ❑ Read First if the word size is greater than 16 bit else ignore DRR2
01	DRR1	R	0x0000	McBSP Data Receive Register1 ❑ Read second if the word size is greater than 16 bit else read DRR1 only
02	DXR2	W	0x0000	McBSP Data Transmit Register2 ❑ Write First if the word size is greater than 16 bit else ignore DXR2
03	DXR1	W	0x0000	McBSP Data Transmit Register1 ❑ Write second if the word size is greater than 16 bit else Write to DXR1 only
McBSP Control Registers				
04	SPCR2	R/W	0x0000	McBSP Serial Port Control Register2
05	SPCR1	R/W	0x0000	McBSP Serial Port Control Register1
06	RCR2	R/W	0x0000	McBSP Receive Control Register2
07	RCR1	R/W	0x0000	McBSP Receive Control Register1
08	XCR2	R/W	0x0000	McBSP Transmit Control Register2
09	XCR1	R/W	0x0000	McBSP Transmit Control Register1

† MFFST will read 0x000A if FSX/FSR pins are left unconnected; else, it will assume pin status at reset.

Table 1–3. McBSP Register Summary (Continued)

Address Offset 0x000 xxh	16-Bit Access	Type R/W	Reset ValueHex	McBSP Communications Interface Register Description
Multichannel Control Registers (Continued)				
0A	SRGR2	R/W	0x2000	McBSP Sample Rate Generator Register2
0B	SRGR1	R/W	0x0001	McBSP Sample Rate Generator Register1
0C	MCR2	R/W	0x0000	McBSP Multi–Channel Register2
0D	MCR1	R/W	0x0000	McBSP Multi–Channel Register1
0E	RCERA	R/W	0x0000	McBSP Receive Channel Enable Register Partition A
0F	RCERB	R/W	0x0000	McBSP Receive Channel Enable Register Partition B
10	XCERA	R/W	0x0000	McBSP Transmit Channel Enable Register Partition A
11	XCERB	R/W	0x0000	McBSP Transmit Channel Enable Register Partition A
12	PCR1	R/W	0x0000	McBSP Pin Control Register
13	RCERC	R/W	0x0000	McBSP Receive Channel Enable Register Partition C
14	RCERD	R/W	0x0000	McBSP Receive Channel Enable Register Partition D
15	XCERC	R/W	0x0000	McBSP Transmit Channel Enable Register Partition C
16	XCERD	R/W	0x0000	McBSP Transmit Channel Enable Register Partition D
17	RCERE	R/W	0x0000	McBSP Receive Channel Enable Register Partition E
18	RCERF	R/W	0x0000	McBSP Receive Channel Enable Register Partition F
19	XCERE	R/W	0x0000	McBSP Transmit Channel Enable Register Partition E
1A	XCERF	R/W	0x0000	McBSP Transmit Channel Enable Register Partition F
1B	RCERG	R/W	0x0000	McBSP Receive Channel Enable Register Partition G
1C	RCERH	R/W	0x0000	McBSP Receive Channel Enable Register Partition H
1D	XCERG	R/W	0x0000	McBSP Transmit Channel Enable Register Partition G
1E	XCERH	R/W	0x0000	McBSP Transmit Channel Enable Register Partition H

† MFFST will read 0x000A if FSX/FSR pins are left unconnected; else, it will assume pin status at reset.

Table 1–3. McBSP Register Summary (Continued)

Address Offset 0x000 xxh	16-Bit Access	Type R/W	Reset ValueHex	McBSP Communications Interface Register Description
FIFO Mode Data Registers Applicable Only FIFO Mode				
00	DRR2	R	0x0000	McBSP Data Receive Register2 –Top of receive FIFO ❑ Read First FIFO pointers will not advance
01	DRR1	R	0x0000	McBSP Data Receive Register1–Top of receive FIFO ❑ Read second for FIFO pointers to advance
02	DXR2	W	0x0000	McBSP Data Transmit Register2–Top of transmit FIFO ❑ Write First FIFO pointers will not advance
03	DXR1	W	0x0000	McBSP Data Transmit Register1–Top of transmit FIFO ❑ Write second for FIFO pointers to advance
FIFO Control Registers				
20	MFFTX	R/W	0x2000	McBSP FIFO Transmit register
21	MFFRX	R/W	0x201F	McBSP FIFO Receive register
22	MFFCT	R/W	0x0000	McBSP FIFO Control register
23	MFFINT	R/W	0x0000	McBSP FIFO Interrupt register
24	MFFST	R/W	0x000x†	McBSP FIFO Status register

† MFFST will read 0x000A if FSX/FSR pins are left unconnected; else, it will assume pin status at reset.

The registers bits are summarized in Table 6–5.

1.3 McBSP Operation

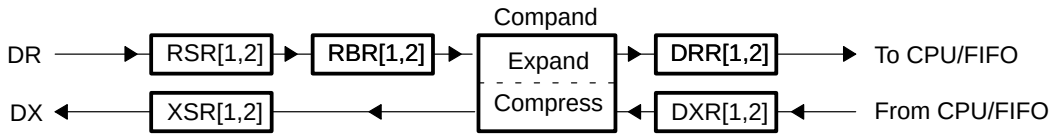
This section contains the following topics:

Topic	See ...
Data transfer process of a McBSP	Section 1.3.1
Companding (compressing and expanding) data	Section 1.3.2, page 1-11
Clocking and framing data	Section 1.3.3, page 1-14
Frame phases	Section 1.3.4, page 1-18
McBSP Reception	Section 1.3.5, page 1-21
McBSP Transmission	Section 1.3.6, page 1-22
Interrupts and FIFO events generated by a McBSP	Section 1.3.7, page 1-24

1.3.1 Data Transfer Process of a McBSP

Figure 1–2 shows a diagram of the McBSP data transfer paths. McBSP receive operation is triple buffered, and transmit operation is double buffered. The use of registers varies depending on whether the defined length of each serial word fits in 16 bits.

Figure 1–2. McBSP Data Transfer Paths



1.3.1.1 Data Transfer Process for Word Length of 8, 12, or 16 Bits

If the word length is 16 bits or smaller, only one 16-bit register is needed at each stage of the data transfer paths. The registers DRR2, RBR2, RSR2, DXR2, and XSR2 are not used (written, read, or shifted).

Receive data arrives on the DR pin and is shifted into receive shift register 1 (RSR1). Once a full word is received, the content of RSR1 is copied to receive buffer register 1 (RBR1), only if RBR1 is not full with previous data. RBR1 is then copied to data receive register 1 (DRR1), unless the previous content of DRR1 has not been read by the CPU. If the companding feature of the McBSP is implemented, the required word length is 8 bits and receive data is expanded into the appropriate format before being passed from RBR1 to DRR1. For more details about reception, see section 1.3.5 on page 1-21.

Transmit data is written by the CPU to data transmit register 1 (DXR1). If there is no previous data in transmit shift register (XSR1), the value in DXR1 is copied to XSR1; otherwise, DXR1 is copied to XSR1 when the last bit of the previous data is shifted out on the DX pin. If selected, the companding module compresses 16-bit data into the appropriate 8-bit format before passing it to XSR1. After transmit frame synchronization, the transmitter begins shifting bits from XSR1 to the DX pin. For more details about transmission, see section 1.3.6 on page 1-22.

1.3.1.2 Data Transfer Process for Word Length of 20, 24, or 32 Bits

If the word length is larger than 16 bits, two 16-bit registers are needed at each stage of the data transfer paths. The registers DRR2, RBR2, RSR2, DXR2, and XSR2 are needed to hold the most significant bits.

Receive data arrives on the DR pin and is shifted into RSR2 first and then into RSR1. Once the full word is received, the contents of RSR2 and RSR1 are copied to RBR2 and RBR1, respectively, only if RBR1 is not full. Then the contents of RBR2 and RBR1 are copied to DRR2 and DRR1, respectively, unless the previous content of DRR1 has not been read by the CPU. The CPU must read data from DRR2 first and then from DRR1. When DRR1 is read, the next RBR-to-DRR copy occurs. For more details about reception, see section 1.3.5 on page 1-21.

For transmission, the CPU must write data to DXR2 first and then to DXR1. When new data arrives in DXR1, if there is no previous data in XSR1, the contents of DXR2 and DXR1 are copied to XSR2 and XSR1, respectively; otherwise, the contents of the DXRs are copied to the XSRs when the last bit of the previous data is shifted out on the DX pin. After transmit frame synchronization, the transmitter begins shifting bits from the XSRs to the DX pin. For more details about transmission, see section 1.3.6 on page 1-22.

1.3.2 Companding (Compressing and Expanding) Data

Companding (COMpressing and exPANDing) hardware allows compression and expansion of data in either μ-law or A-law format. The companding standard employed in the United States and Japan is μ-law. The European companding standard is referred to as A-law. The specifications for μ-law and A-law log PCM are part of the CCITT G.711 recommendation.

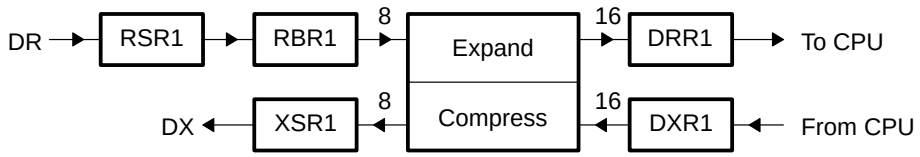
A-law and μ-law allow 13 bits and 14 bits of dynamic range, respectively. Any values outside this range are set to the most positive or most negative value. Thus, for companding to work best, the data transferred to and from the McBSP via the CPU must be at least 16 bits wide.

The μ-law and A-law formats both encode data into 8-bit code words. Companded data is always 8 bits wide; the appropriate word length bits

(RWDLEN1, RWDLEN2, XWDLEN1, XWDLEN2) must therefore be set to 0, indicating an 8-bit wide serial data stream. If companding is enabled and either of the frame phases does not have an 8-bit word length, companding continues as if the word length is 8 bits.

Figure 1–3 illustrates the companding processes. When companding is chosen for the transmitter, compression occurs during the process of copying data from DXR1 to XSR1. The transmit data is encoded according to the specified companding law (A-law or μ-law). When companding is chosen for the receiver, expansion occurs during the process of copying data from RBR1 to DRR1. The receive data is decoded to 2s-complement format.

Figure 1–3. Companding Processes

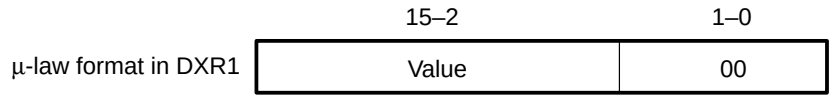


1.3.2.1 Companding Formats

For reception, the 8-bit compressed data in RBR1 is expanded to left-justified 16-bit data in DRR1. The receive sign-extension and justification mode specified in RJUST is ignored when companding is used.

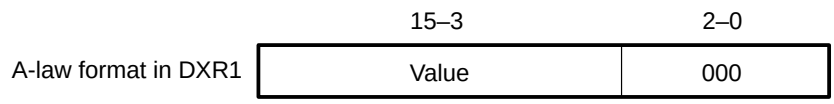
For transmission using μ-law compression, make sure the 14 data bits are left-justified in DXR1, with the remaining two low-order bits filled with 0s as shown in Figure 1–4.

Figure 1–4. μ-Law Transmit Data Companding Format



For transmission using A-law compression, make sure the 13 data bits are left-justified in DXR1, with the remaining three low-order bits filled with 0s as shown in Figure 1–5.

Figure 1–5. A-Law Transmit Data Companding Format



1.3.2.2 Capability to Compand Internal Data

If the McBSP is otherwise unused (the serial port transmit and receive sections are reset), the companding hardware can compand internal data. This can be used to:

- Convert linear to the appropriate μ-law or A-law format.
- Convert μ-law or A-law 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.

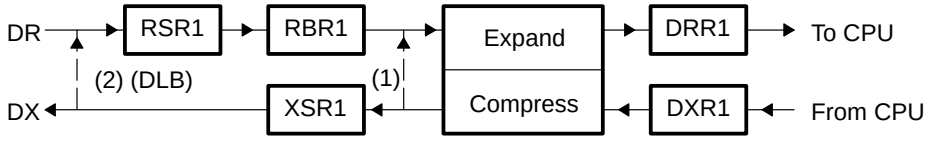
Figure 1–6 shows two methods by which the McBSP can compand internal data. Data paths for these two methods are used to indicate:

- When both the transmit and receive sections of the serial port are reset, DRR1 and DXR1 are connected internally through the companding logic. Values from DXR1 are compressed, as selected by XCOMPAND, and then expanded, as selected by RCOMPAND. Note that RRDY and XRDY bits are not set. However, data is available in DRR1 within four CPU clocks after being written to DXR1.

  The advantage of this method is its speed. The disadvantage is that there is no synchronization available to the CPU to control the flow. Note that DRR1 and DXR1 are internally connected if the (X/R)COMPAND bits are set to 10b or 11b (compand using μ-law or A-law).

- The McBSP is enabled in digital loopback mode with companding appropriately enabled by RCOMPAND and XCOMPAND. Receive and transmit interrupts (RINT when RINTM = 0 and XINT when XINTM = 0) or synchronization events (REVT and XEVT) allow synchronization of the CPU to these conversions. The time for this companding depends on the serial bit rate selected.

Figure 1–6. Two Methods by Which the McBSP Can Compand Internal Data



1.3.2.3 Reversing Bit Order: Option to Transfer LSB First

Normally, the McBSP transmit or receives all data with the most significant bit (MSB) first. However, certain 8-bit data protocols (that do not use companded data) require the least significant bit (LSB) to be transferred first. If you set XCOMPAND = 01b in XCR2, the bit ordering of 8-bit words is reversed (LSB first) before being sent from the serial port. If you set RCOMPAND = 01b in RCR2, the bit ordering of 8-bit words is reversed during reception. Similar to companding, this feature is enabled only if the appropriate word length bits are set to 0, indicating that 8-bit words are to be transferred serially. If either phase of the frame does not have an 8-bit word length, the McBSP assumes the word length is eight bits, and LSB-first ordering is done.

1.3.3 Clocking and Framing Data

This section explains basic concepts and terminology important for understanding how McBSP data transfers are timed and delimited.

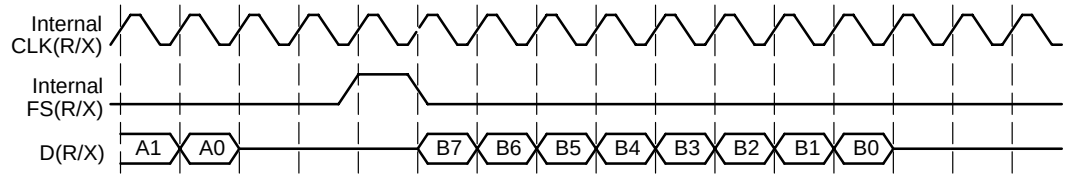
1.3.3.1 Clocking

Data is shifted one bit at a time from the DR pin to the RSR(s) or from the XSR(s) to the DX pin. The time for each bit transfer is controlled by the rising or falling edge of a clock signal.

The receive clock signal (CLKR) controls bit transfers from the DR pin to the RSR(s). The transmit clock signal (CLKX) controls bit transfers from the XSR(s) to the DX pin. CLKR or CLKX can be derived from a pin at the boundary of the McBSP or derived from inside the McBSP. The polarities of CLKR and CLKX are programmable.

In Figure 1–7, the clock signal controls the timing of each bit transfer on the pin.

Figure 1–7. Clock Signal Control Waveform



Note:

The McBSP cannot operate at a frequency faster than 1/2 the CPU clock frequency. When driving CLKX or CLKR at the pin, choose an appropriate input clock frequency. When using the internal sample rate generator for CLKX and/or CLKR, choose an appropriate input clock frequency and divide down value (CLKDV).

1.3.3.2 Serial Words

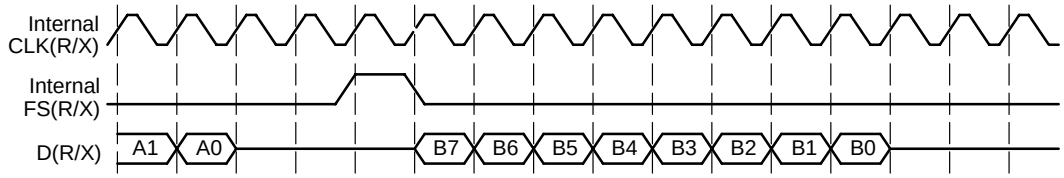
Bits traveling between a shift register (RSR or XSR) and a data pin (DR or DX) are transferred in a group called a **serial word**. You define how many bits are in a word.

Bits coming in on the DR pin are held in RSR until RSR holds a full serial word. Only then is the word passed to RBR (and ultimately to the DRR).

During transmission, XSR does not accept new data from DXR until a full serial word has been passed from XSR to the DX pin.

In Figure 1–8, an 8-bit word size was defined (see bits 7 through 0 of word B being transferred).

Figure 1–8. 8-Bit Word Size Defined Waveform



1.3.3.3 Frames and Frame Synchronization

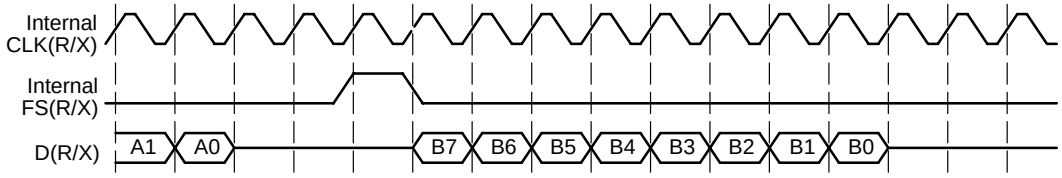
One or more words are transferred in a group called a **frame**. You define how many words are in a frame.

All of the words in a frame are sent in a continuous stream. However, there can be pauses between frame transfers. The McBSP uses frame-synchronization (frame-sync) signals to determine when each frame is received/transmitted. When a pulse occurs on a frame-sync signal, the McBSP begins receiving/transmitting a frame of data. When the next pulse occurs, the McBSP receives/transmits the next frame, and so on.

Pulses on the receive frame-sync signal (FSR) initiate frame transfers on DR. Pulses on the transmit frame-sync signal (FSX) initiate frame transfers on DX. FSR or FSX can be derived from a pin at the boundary of the McBSP or derived from inside the McBSP.

In Figure 1–9, a 1-word frame is transferred when a frame-sync pulse occurs.

Figure 1–9. One-word Frame Transfer



In McBSP operation, the inactive-to-active transition of the frame-synchronization signal indicates the start of the next frame. For this reason, the frame-sync signal may be high for an arbitrary number of clock cycles. Only after the signal is recognized to have gone inactive, and then active again, does the next frame synchronization occur.

1.3.3.4 Detecting Frame-Sync Pulses, Even in the Reset State

The McBSP can send receive and transmit interrupts to the CPU to indicate specific events in the McBSP. To facilitate detection of frame synchronization, these interrupts can be sent in response to frame-sync pulses. Set the appropriate interrupt mode bits to 10b (for reception, RINTM = 10b; for transmission, XINTM = 10b).

Unlike other serial port interrupt modes, this mode can operate while the associated portion of the serial port is in reset (such as activating RINT when the receiver is in reset). In this case, FSRM/FSXM and FSRP/FSXP still select the appropriate source and polarity of frame synchronization. Thus, even when the serial port is in the reset state, these signals are synchronized to the CPU clock and then sent to the CPU in the form of RINT and XINT at the point at which they feed the receiver and transmitter of the serial port. Consequently, a new frame-synchronization pulse can be detected, and after this occurs the CPU can take the serial port out of reset safely.

1.3.3.5 Ignoring Frame-Sync Pulses

The McBSP can be configured to ignore transmit and/or receive frame-synchronization pulses. To have the receiver or transmitter recognize frame-sync pulses, clear the appropriate frame-sync ignore bit (RFIG = 0 for the receiver, XFIG = 0 for the transmitter). To have the receiver or transmitter ignore frame-sync pulses until the desired frame length or number of words is reached, set the appropriate frame-sync ignore bit (RFIG = 1 for the receiver, XFIG = 1 for the transmitter). For more details on unexpected frame-sync pulses, see one of the following topics:

- *Unexpected Receive Frame-Sync Pulse* (page 1-40)
- *Unexpected Transmit Frame-Sync Pulse* (page 1-46)

You can also use the frame-sync ignore function for data packing (for more details, see section 4.2.2 on page 4-9).

1.3.3.6 Frame Frequency

The frame frequency is determined by the period between frame-synchronization pulses and is defined as shown by Equation 1–1.

Equation 1–1. Frame Frequency of a McBSP

$$\text{Frame Frequency} = \frac{\text{Clock Frequency}}{\text{Number of Clock Cycles Between Frame–Sync Pulses}}$$

The frame frequency may be increased by decreasing the time between frame-synchronization pulses (limited only by the number of bits per frame). As the frame transmit frequency increases, the inactivity period between the data packets for adjacent transfers decreases to zero.

1.3.3.7 Maximum Frame Frequency

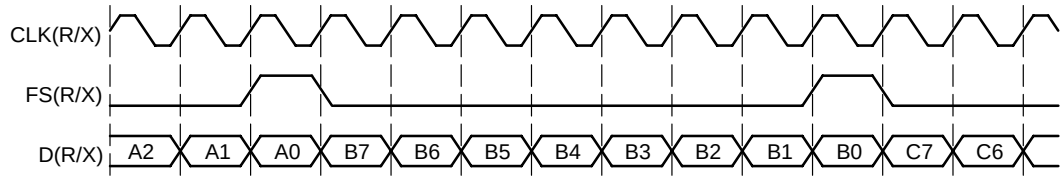
The minimum number of clock cycles between frame synchronization pulses is equal to the number of bits transferred per frame. The maximum frame frequency is defined as shown by Equation 1–2.

Equation 1–2. Maximum Frame Frequency of a McBSP

$$\text{Maximum Frame Frequency} = \frac{\text{Clock Frequency}}{\text{Number of Bits Per Frame}}$$

Figure 1–10 shows the McBSP operating at maximum packet frequency. At maximum packet frequency, the data bits in consecutive packets are transmitted contiguously with no inactivity between bits.

Figure 1–10. McBSP Operating at Maximum Packet Frequency



If there is a 1-bit data delay as shown in this figure, the frame-synchronization pulse overlaps the last bit transmitted in the previous frame. Effectively, this permits a continuous stream of data, making frame-synchronization pulses redundant. Theoretically, only an initial frame-synchronization pulse is required to initiate a multipacket transfer.

The McBSP supports operation of the serial port in this fashion by ignoring the successive frame-sync pulses. Data is clocked in to the receiver, or clocked out of the transmitter, during every clock cycle.

Note:

For XDATDLY = 0 (0-bit data delay), the first bit of data is transmitted asynchronously to the internal transmit clock signal (CLKX). For more details, see *Set the Transmit Data Delay* on page 3-36.

1.3.4 Frame Phases

The McBSP allows you to configure each frame to contain one or two phases. The number of words per frame, and the number of bits per word, can be specified differently for each of the two phases of a frame, allowing greater flexibility in structuring data transfers. For example, a user might define a frame as consisting of one phase containing two words of 16 bits each, followed by a second phase consisting of 10 words of 8 bits each. This configuration permits the user to compose frames for custom applications, or in general, to maximize the efficiency of data transfers.

1.3.4.1 Number of Phases, Words, and Bits Per Frame

Table 1–4 shows which bit fields in the receive control registers (RCR1 and RCR2) and in the transmit control registers (XCR1 and XCR2) determine the number of phases per frame, the number of words per frame, and number of bits per word for each phase, for the receiver and transmitter. The maximum number of words per frame is 128 for a single-phase frame and 256 for a dual-phase frame. The number of bits per word can be 8, 12, 16, 20, 24, or 32 bits.

Table 1–4. McBSP Register Bits That Determine the Number of Phases, Words, and Bits Per Frame

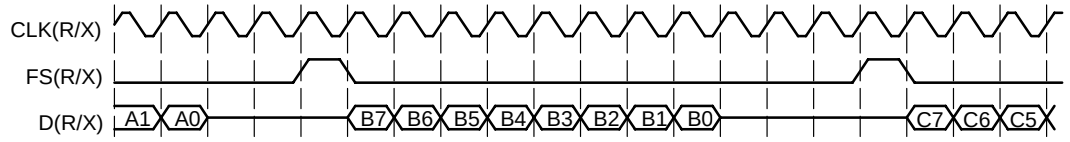
Operation	Number of Phases	Words Per Frame Set With ...	Bits Per Word Set With ...
Reception	1 (RPHASE = 0)	RFRLEN1	RWDLEN1
Reception	2 (RPHASE = 1)	RFRLEN1 and RFRLEN2	RWDLEN1 for phase 1 RWDLEN2 for phase 2
Transmission	1 (XPHASE = 0)	XFRLEN1	XWDLEN1
Transmission	2 (XPHASE = 1)	XFRLEN1 and XFRLEN2	XWDLEN1 for phase 1 XWDLEN2 for phase 2

1.3.4.2 Single-Phase Frame Example

Figure 1–11 shows an example of a single-phase data frame comprising one 8-bit word. Since the transfer is configured for one data bit delay, the data on the DX and DR pins are available one clock cycle after FS(R/X) goes active. The figure makes the following assumptions:

- (R/X)PHASE = 0: Single-phase frame
- (R/X)FRLEN1 = 0b: 1 word per frame
- (R/X)WDLEN1 = 000b: 8-bit word length
- (R/X)FRLEN2 and (R/X)WDLEN2 are ignored
- CLK(X/R)P = 0: Receive data clocked on falling edge; transmit data clocked on rising edge
- FS(R/X)P = 0: Active-high frame-sync signals
- (R/X)DATDLY = 01b: 1-bit data delay

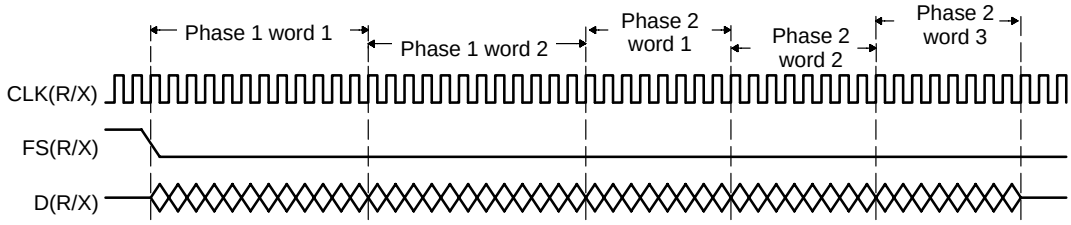
Figure 1–11. Single-Phase Frame for a McBSP Data Transfer



1.3.4.3 Dual-Phase Frame Example

Figure 1–12 shows an example of a frame where the first phase consists of 2 words of 12 bits each followed by a second phase of three words of 8 bits each. Note that the entire bit stream in the frame is contiguous. There are no gaps either between words or between phases.

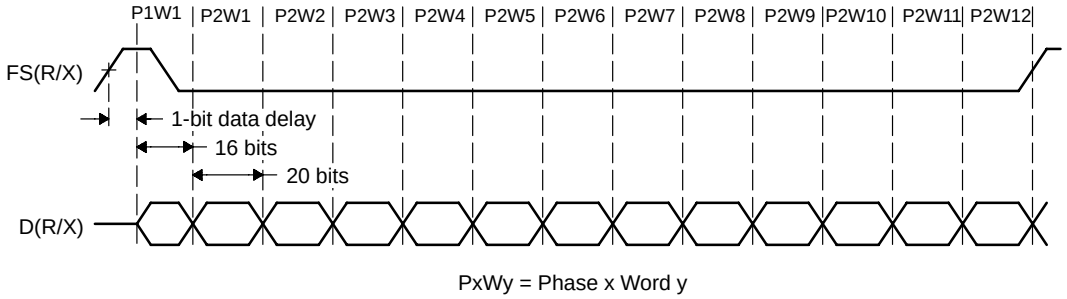
Figure 1–12. Dual-Phase Frame for a McBSP Data Transfer



1.3.4.4 Implementing the AC97 Standard With a Dual-Phase Frame

Figure 1–13 shows an example of the Audio Codec ‘97 (AC97) standard, which uses the dual-phase frame feature. Notice that words, not individual bits, are shown on the D(R/X) signal. The first phase (P1) consists of a single 16-bit word. The second phase (P2) consists of twelve 20-bit words. The phase configurations are listed after the figure.

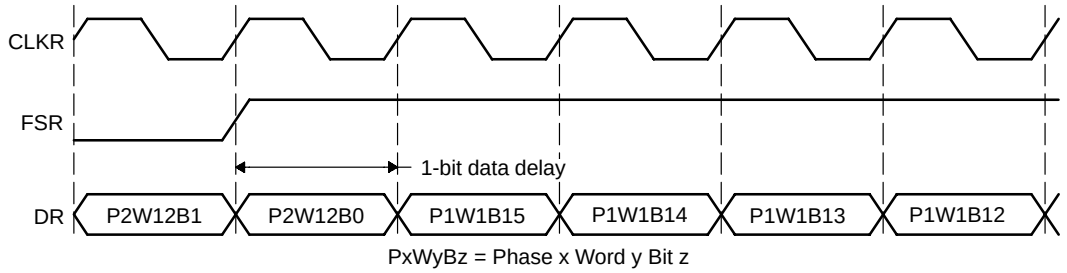
Figure 1–13. Implementing the AC97 Standard With a Dual-Phase Frame



- ❑ (R/X)PHASE = 1: Dual-phase frame
- ❑ (R/X)FRLEN1 = 0000000b: 1 word in phase 1
- ❑ (R/X)WDLEN1 = 010b: 16 bits per word in phase 1
- ❑ (R/X)FRLEN2 = 0001011b: 12 words in phase 2
- ❑ (R/X)WDLEN2 = 011b: 20 bits per word in phase 2
- ❑ CLKRP/CLKXP= 0: Receive data sampled on falling edge of internal CLKR / transmit data clocked on rising edge of internal CLKX
- ❑ FSRP/FSXP = 0: Active-high frame-sync signal
- ❑ (R/X)DATDLY = 01b: Data delay of 1 clock cycle (1-bit data delay)

Figure 1–14 shows the timing of an AC97-standard data transfer near frame synchronization. In this figure, individual bits are shown on D(R/X). Specifically, the figure shows the last two bits of phase 2 of one frame and the first four bits of phase 1 of the next frame. Regardless of the data delay, data transfers can occur without gaps. The first bit of the second frame (P1W1B15) immediately follows the last bit of the first frame (P2W12B0). Because a 1-bit data delay has been chosen, the transition on the frame-sync signal can occur when P2W12B0 is transferred.

Figure 1–14. Timing of an AC97-Standard Data Transfer Near Frame Synchronization

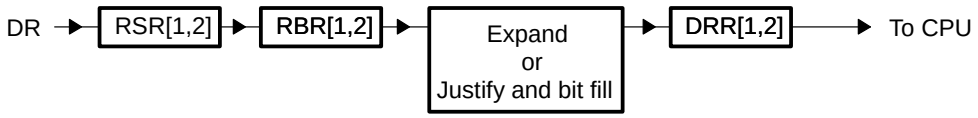


1.3.5 McBSP Reception

This section explains the fundamental process of reception in the McBSP. For details about how to program the McBSP receiver, see *Receiver Configuration* on page 3-2.

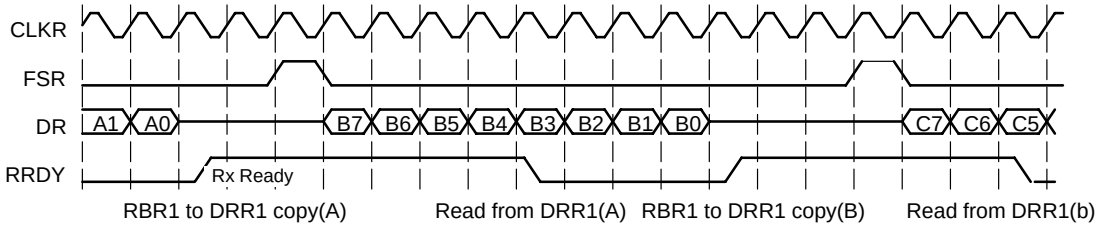
Figure 1–15 and Figure 1–16 show how reception occurs in the McBSP. Figure 1–15 shows the physical path for the data. Figure 1–16 is a timing diagram showing signal activity for one possible reception scenario. A description of the process follows the figures.

Figure 1–15. McBSP Reception Physical Data Path



RSR[1,2]: Receive shift registers 1 and 2
RBR[1,2]: Receive buffer registers 1 and 2
DRR[1,2]: Data receive registers 1 and 2

Figure 1–16. McBSP Reception Signal Activity



CLKR: Internal receive clock
FSR: Internal receive frame-sync signal
DR: Data on DR pin
RRDY: Status of receiver ready bit (high is 1)

The following process describes how data travels from the DR pin to the CPU:

1) The McBSP waits for a receive frame-sync pulse on internal FSR.

2) When the pulse arrives, the McBSP inserts the appropriate data delay that is selected with the RDATDLY bits of RCR2.

   In the preceding timing diagram (Figure 1–16), a 1-bit data delay is selected.

3) The McBSP accepts data bits on the DR pin and shifts them into the receive shift register(s).

   If the word length is 16 bits or smaller, only RSR1 is used. If the word length is larger than 16 bits, RSR2 and RSR1 are used, and RSR2 contains the

most significant bits. For details on choosing a word length, see *Set the Receive Word Length(s)*.

4) When a full word is received, the McBSP copies the contents of the receive shift register(s) to the receive buffer register(s), provided that RBR1 is not full with previous data.

   If the word length is 16 bits or smaller, only RBR1 is used. If the word length is larger than 16 bits, RBR2 and RBR1 are used, and RBR2 contains the most significant bits.

5) The McBSP copies the contents of the receive buffer register(s) into the data receive register(s), provided that DRR1 is not full with previous data. When DRR1 receives new data, the receiver ready bit (RRDY) is set in SPCR1. This indicates that receive data is ready to be read by the CPU.

   If the word length is 16 bits or smaller, only DRR1 is used. If the word length is larger than 16 bits, DRR2 and DRR1 are used, and DRR2 contains the most significant bits.

   If companding is used during the copy (RCOMPAND = 10b or 11b in RCR2), the 8-bit compressed data in RBR1 is expanded to a left-justified 16-bit value in DRR1. If companding is disabled, the data copied from RBR[1,2] to DRR[1,2] is justified and bit filled according to the RJUST bits.

6) The CPU reads the data from the data receive register(s). When DRR1 is read, RRDY is cleared and the next RBR-to-DRR copy is initiated.

Note:

If both DRRs are needed (word length larger than 16 bits), the CPU must read from DRR2 first and then from DRR1. As soon as DRR1 is read, the next RBR-to-DRR copy is initiated. If DRR2 is not read first, the data in DRR2 is lost.

When activity is not properly timed, errors can occur. See the following topics for more details:

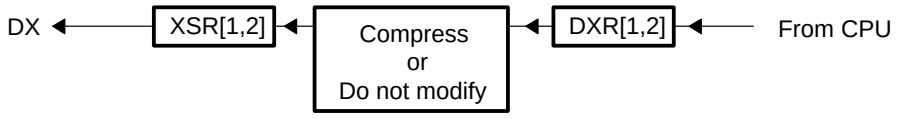
- *Overrun in the Receiver* (page 1-39)
- *Unexpected Receive Frame-Sync Pulse* (page 1-40)

1.3.6 McBSP Transmission

This section explains the fundamental process of transmission in the McBSP. For details about how to program the McBSP transmitter, see *Transmitter Configuration* on page 3-26.

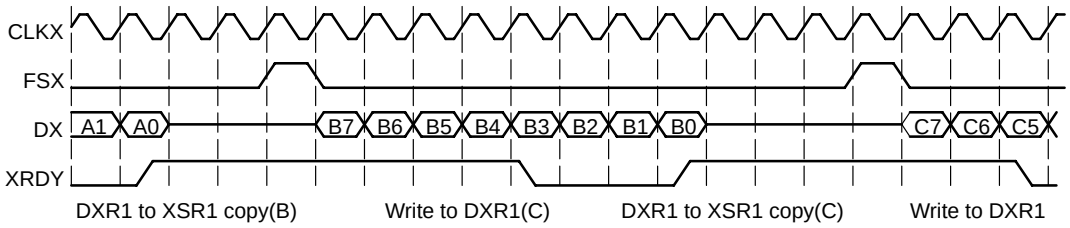
Figure 1–17 and Figure 1–18 show how transmission occurs in the McBSP. Figure 1–17 shows the physical path for the data. Figure 1–18 is a timing diagram showing signal activity for one possible transmission scenario. A description of the process follows the figures.

Figure 1–17. McBSP Transmission Physical Data Path



XSR[1,2]: Transmit shift registers 1 and 2 DXR[1,2]: Data transmit registers 1 and 2

Figure 1–18. McBSP Transmission Signal Activity



CLKX: Internal transmit clock
FSX: Internal transmit frame-sync signal
DX: Data on DX pin
XRDY: Status of transmitter ready bit (high is 1)

1) The CPU writes data to the data transmit register(s). When DXR1 is loaded, the transmitter ready bit (XRDY) is cleared in SPCR2 to indicate that the transmitter is not ready for new data.

   If the word length is 16 bits or smaller, only DXR1 is used. If the word length is larger than 16 bits, DXR2 and DXR1 are used, and DXR2 contains the most significant bits. For details on choosing a word length, see *Set the Transmit Word Length(s)* in Chapter 3.

Note:

If both DXRs are needed (word length larger than 16 bits), the CPU must load DXR2 first and then load DXR1. As soon as DXR1 is loaded, the contents of both DXRs are copied to the transmit shift registers (XSRs), as described in the next step. If DXR2 is not loaded first, the previous content of DXR2 is passed to the XSR2.

2) When new data arrives in DXR1, the McBSP copies the content of the data transmit register(s) to the transmit shift register(s). In addition, the transmit

ready bit (XRDY) is set. This indicates that the transmitter is ready to accept new data from the CPU.

If the word length is 16 bits or smaller, only XSR1 is used. If the word length is larger than 16 bits, XSR2 and XSR1 are used, and XSR2 contains the most significant bits.

If companding is used during the transfer (XCOMPAND = 10b or 11b in XCR2), the McBSP compresses the 16-bit data in DXR1 to 8-bit data in the μ-law or A-law format in XSR1. If companding is disabled, the McBSP passes data from the DXR(s) to the XSR(s) without modification.

3) The McBSP waits for a transmit frame-sync pulse on internal FSX.

4) When the pulse arrives, the McBSP inserts the appropriate data delay that is selected with the XDATDLY bits of XCR2.

   In the preceding timing diagram (Figure 1–18), a 1-bit data delay is selected.

5) The McBSP shifts data bits from the transmit shift register(s) to the DX pin.

When activity is not properly timed, errors can occur. See the following topics for more details:

- *Overwrite in the Transmitter* (page 1-43)
- *Underflow in the Transmitter* (page 1-44)
- *Unexpected Transmit Frame-Sync Pulse* (page 1-46)

1.3.7 Interrupts and FIFO Events Generated by a McBSP

The McBSP sends notification of important events to the CPU and FIFO via the internal signals shown in Table 1–5.

Table 1–5. Interrupts and FIFO Events Generated by a McBSP

Internal Signal	Description
RINT	Receive interrupt The McBSP can send a receive interrupt request to CPU based upon a selected condition in the receiver of the McBSP (a condition selected by the RINTM bits of SPCR1).
XINT	Transmit interrupt The McBSP can send a transmit interrupt request to CPU based upon a selected condition in the transmitter of the McBSP (a condition selected by the XINTM bits of SPCR2).

Table 1–5. Interrupts and FIFO Events Generated by a McBSP (Continued)

Internal Signal	Description
REVT	Receive synchronization event An REVT signal is sent to the FIFO when data has been received in the data receive registers (DRRs).
XEVT	Transmit synchronization event An XEVT signal is sent to the FIFO when the data transmit registers (DXRs) are ready to accept the next serial word for transmission.
REVTA	A-bis mode receive synchronization event If ABIS = 1 (A-bis mode is enabled) an REVTA signal is sent to the FIFO every 16 cycles.
XEVTA	A-bis mode transmit synchronization event If ABIS = 1 (A-bis mode is enabled) an XEVTA signal is sent to the FIFO every 16 cycles.

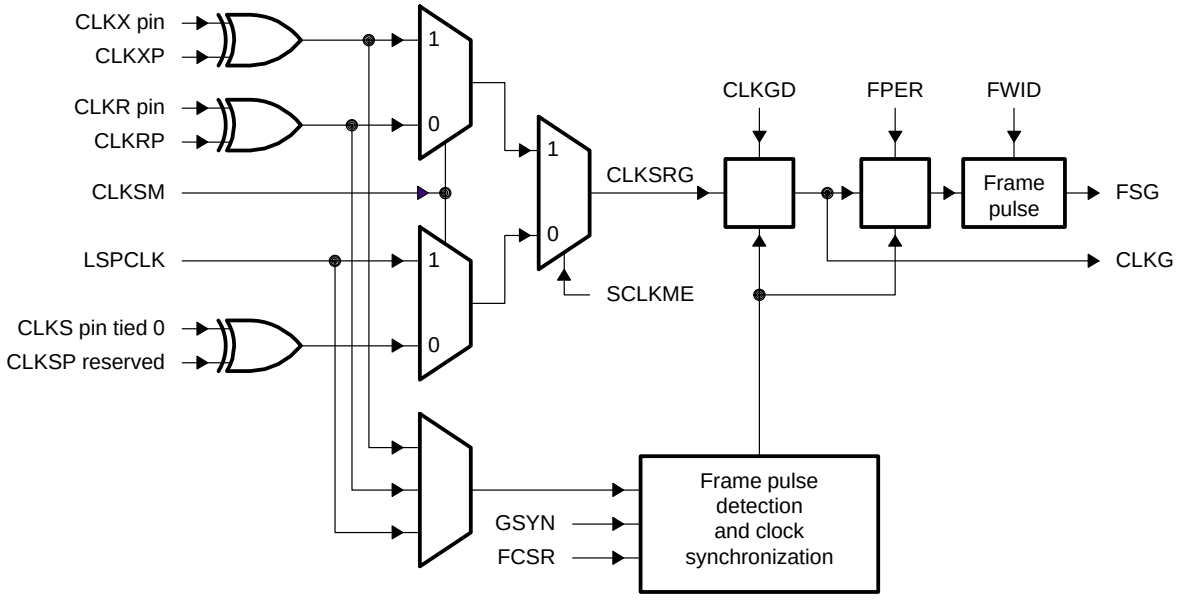
1.4 Sample Rate Generator of the McBSP

Note: General-purpose I/O Selection

Do not use the GPIO function using RIOEN/XIOEN bits 12 and 13 in the PCR register. This feature is not applicable to the 28x McBSP implementation; therefore, these bits are reserved on 28x devices.

The McBSP contains a sample rate generator module that can be programmed to generate internal data-clock (CLKG) and an internal frame-synchronization (FSG) signal. Figure 1–19 is a conceptual block diagram that shows the clock selection options available on the McBSP. The clock selection to the module is selectable using the SCLKME bit (PCR bit 7) and CLKSM bit (SRGR2 bit 13).

Figure 1–19. Sample Rate Generator Clock Selection



The McBSP contains a sample rate generator that can be used to generate an internal data clock (CLKG) and an internal frame-synchronization signal (FSG). CLKG can be used for bit shifting on the data receive (DR) pin and/or the data transmit (DX) pin. FSG can be used to initiate frame transfers on DR and/or DX.

The source clock for the sample rate generator (labeled CLKSRG in the diagram) can be supplied by the LSPCLK or by an external pin (CLKS, CLKX, or CLKR). The source is selected with the SCLKME bit of PCR and the CLKSM

bit of SRGR2. If a pin is used, the polarity of the incoming signal can be inverted with the appropriate polarity bit (CLKSP of SRGR2, CLKXP of PCR, or CLKRP of PCR).

The sample rate generator has a 3-stage clock divider that gives CLKG and FSG programmability. The three stages provide:

- Clock divide down. The source clock is divided according to the CLKGDV bits of SRGR1 to produce CLKG.
- Frame period divide down. CLKG is divided according to the FPER bits of SRGR2 to control the period from the start of a frame-sync pulse to the start of the next pulse.
- Frame-sync pulse width countdown. CLKG cycles are counted according to the FWID bits of SRGR1 to control the width of each frame-sync pulse.

In addition to the 3-stage clock divider, the sample rate generator has a frame-sync pulse detection and clock synchronization module that allows synchronization of the clock divide down with an incoming frame-sync pulse on the FSR pin. This feature is enabled or disabled with the GSYNC bit of SRGR2.

For details on getting the sample rate generator ready for operation, see the reset and initialization procedure on page 1-33.

1.4.1 Clock Generation in the Sample Rate Generator

The sample rate generator can produce a clock signal (CLKG) for use by the receiver, the transmitter, or both. Use of the sample rate generator to drive clocking is controlled by the clock mode bits (CLKRM and CLKXM) in the pin control register (PCR). When a clock mode bit is set to 1 (CLKRM = 1 for reception, CLKXM = 1 for transmission), the corresponding data clock (CLKR for reception, CLKX for transmission) is driven by the internal sample rate generator output clock (CLKG).

Note that the effects of CLKRM = 1 and CLKXM = 1 on the McBSP are partially affected by the use of the digital loopback mode and the clock stop (SPI) mode, respectively, as described in Table 1–6. The digital loopback mode (described in Chapter 3) is selected with the DLB bit of SPCR1. The clock stop mode is selected with the CLKSTP bits of SPCR1.

When using the sample rate generator as a clock source, make sure the sample rate generator is enabled ($\overline{\text{GRST}}$ = 1).

Table 1–6. Effects of DLB and CLKSTP on Clock Modes

Mode Bit Settings		Effect
CLKRM = 1	DLB = 0 (Digital loopback mode disabled)	CLKR is an output pin driven by the sample rate generator output clock (CLKG).
	DLB = 1 (Digital loopback mode enabled)	CLKR is an output pin driven by internal CLKX. The source for CLKX depends on the CLKXM bit.
CLKXM = 1	CLKSTP = 00b or 01b (Clock stop (SPI) mode disabled)	CLKX is an output pin driven by the sample rate generator output clock (CLKG).
	CLKSTP = 10b or 11b (Clock stop (SPI) mode enabled)	The McBSP is a master in an SPI system. Internal CLKX drives internal CLKR and the shift clocks of any SPI-compliant slave devices in the system. CLKX is driven by the internal sample rate generator.

1.4.1.1 Choosing an Input Clock

The sample rate generator must be driven by an input clock signal from one of the four sources selectable with the SCLKME bit of PCR and the CLKSM bit of SRGR2 (see Table 1–7). When CLKSM = 1, the minimum divide down value in CLKGDV bits should be 1. CLKGDV is described in section 1.4.1.3.

Note:

The McBSP cannot operate at a frequency faster than one-half the LSPCLK. Choose an input clock frequency and a CLKDV value such that CLKG is less than or equal to one-half the LSPCLK.

Table 1–7. Choosing an Input Clock for the Sample Rate Generator With the SCLKME and CLKSM Bits

SCLKME	CLKSM	Input Clock For Sample Rate Generator
0	0	Reserved
0	1	LSPCLK
1	0	Signal on CLKR pin
1	1	Signal on CLKX pin

1.4.1.2 Choosing a Polarity for the Input Clock

As shown in Figure 1–20, when the input clock is received from a pin, you can choose the polarity of the input clock. The rising edge of CLKSRG generates CLKG and FSG, but you can determine which edge of the input clock causes a rising edge on CLKSRG. The polarity options and their effects are described in Table 1–8.

Figure 1–20. Possible Inputs to the Sample Rate Generator and the Polarity Bits

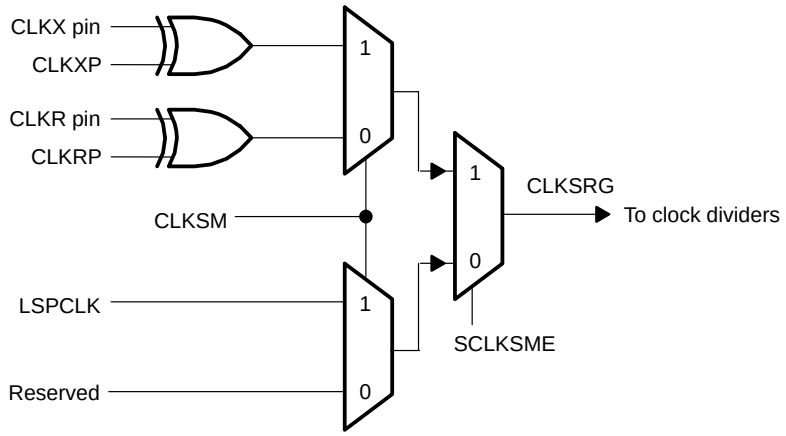


Table 1–8. Polarity Options for the Input to the Sample Rate Generator

Input Clock	Polarity Option	Effect
LSPCLK	Always positive polarity	Rising edge of CPU clock generates transitions on CLKG and FSG.
Signal on CLKR pin	CLKRP = 0 in PCR	Falling edge on CLKR pin generates transitions on CLKG and FSG.
	CLKRP = 1 in PCR	Rising edge on CLKR pin generates transitions on CLKG and FSG.
Signal on CLKX pin	CLKXP = 0 in PCR	Rising edge on CLKX pin generates transitions on CLKG and FSG.
	CLKXP = 1 in PCR	Falling edge on CLKX pin generates transitions on CLKG and FSG.

1.4.1.3 Choosing a Frequency for the Output Clock (CLKG)

The input clock (LSPCLK or external clock) can be divided down by a programmable value to drive CLKG. Regardless of the source to the sample rate generator, the rising edge of CLKSRG (see the sample rate generator diagram on page 1-26) generates CLKG and FSG.

The first divider stage of the sample rate generator creates the output clock from the input clock. This divider stage uses a counter that is preloaded with the divide down value in the CLKGDV bits of SRGR1. The output of this stage is the data clock (CLKG). CLKG has the frequency represented by the following equation.

$$\text{CLKG frequency} = \frac{\text{Input clock frequency}}{(\text{CLKGDV} + 1)}$$

Thus, the input clock frequency is divided by a value between 1 and 256. When CLKGDV is odd or equal to 0, the CLKG duty cycle is 50%. When CLKGDV is an even value, 2p, representing an odd divide down, the high-state duration is p+1 cycles and the low-state duration is p cycles.

Note:

The McBSP cannot operate at a frequency faster than one-half the LSPCLK. Choose an input clock frequency and a CLKDV value such that CLKG is less than or equal to one-half the LSPCLK.

1.4.1.4 Keeping CLKG Synchronized to an External Input Clock

When an external signal is selected to drive the sample rate generator (see section 1.4.1.1), the GSYNC bit in SRGR2 and the FSR pin can be used to configure the timing of the output clock (CLKG) relative to the input clock.

GSYNC = 1 ensures that the McBSP and an external device are dividing down the input clock with the same phase relationship. If GSYNC = 1, an inactive-to-active transition on the FSR pin triggers a resynchronization of CLKG and generation of FSG.

For more details about the synchronization, see section 1.4.3 on page 1-31.

1.4.2 Frame Sync Generation in the Sample Rate Generator

The sample rate generator can produce a frame-sync signal (FSG) for use by the receiver, the transmitter, or both.

If you want the **receiver** to use FSG for frame synchronization, make sure FSRM = 1. (When FSRM = 0, receive frame synchronization is supplied via the FSR pin.)

If you want the **transmitter** to use FSG for frame synchronization, you must set:

- FSXM = 1 in PCR: This indicates that transmit frame synchronization is supplied by the McBSP itself rather than from the FSX pin.
- FSGM = 1 in SRGR2: This indicates that when FSXM = 1, transmit frame synchronization is supplied by the sample rate generator. (When FSGM = 0 and FSXM = 1, the transmitter uses frame-sync pulses generated every time data is transferred from DXR[1,2] to XSR[1,2].)

In either case, the sample rate generator must be enabled ($\overline{\text{GRST}}$ = 1) and the frame-sync logic in the sample rate generator must be enabled ($\overline{\text{FRST}}$ = 0).

1.4.2.1 Choosing the Width of the Frame-Sync Pulse on FSG

Each pulse on FSG has a programmable width. You program the FWID bits of SRGR1, and the resulting pulse width is (FWID + 1) CLKG cycles, where CLKG is the output clock of the sample rate generator.

1.4.2.2 Controlling the Period Between the Starting Edges of Frame-Sync Pulses on FSG

You can control the amount of time from the starting edge of one FSG pulse to the starting edge of the next FSG pulse. This period is controlled in one of two ways, depending on the configuration of the sample rate generator:

- If the sample rate generator is using an external input clock and GSYNC = 1 in SRGR2, FSG pulses in response to an inactive-to-active transition on the FSR pin. Thus, the frame-sync period is controlled by an external device.
- Otherwise, you program the FPER bits of SRGR2, and the resulting frame-sync period is (FPER + 1) CLKG cycles, where CLKG is the output clock of the sample rate generator.

1.4.2.3 Keeping FSG Synchronized to an External Clock

When an external signal is selected to drive the sample rate generator (see section 1.4.1.1 on page 1-28), the GSYNC bit of SRGR2 and the FSR pin can be used to configure the timing of FSG pulses.

GSYNC = 1 ensures that the McBSP and an external device are dividing down the input clock with the same phase relationship. If GSYNC = 1, an inactive-to-active transition on the FSR pin triggers a resynchronization of CLKG and generation of FSG.

Section 1.4.3 has more details about the synchronization.

1.4.3 Synchronizing Sample Rate Generator Outputs to an External Clock

The sample rate generator can produce a clock signal (CLKG) and a frame-sync signal (FSG) based on an input clock signal that is either the CPU clock signal or a signal at the CLKS, CLKR, or CLKX pin. When an external clock is selected to drive the sample rate generator, the GSYNC bit of SRGR2 and the FSR pin can be used to control the timing of CLKG and the pulsing of FSG relative to the chosen input clock.

Make GSYNC = 1 when you want the McBSP and an external device to divide down the input clock with the same phase relationship. If GSYNC = 1:

- An inactive-to-active transition on the FSR pin triggers a resynchronization of CLKG and a pulsing of FSG.

- CLKG always begins with a high state after synchronization.
- FSR is always detected at the same edge of the input clock signal that generates CLKG, no matter how long the FSR pulse is.
- The FPER bits of SRGR2 are ignored because the frame-sync period on FSG is determined by the arrival of the next frame-sync pulse on the FSR pin.

If GSYNC = 0, CLKG runs freely and is not resynchronized, and the frame-sync period on FSG is determined by FPER.

1.4.3.1 Operating the Transmitter Synchronously With the Receiver

When GSYNC = 1, the transmitter can operate synchronously with the receiver, provided that:

- FSX is programmed to be driven by FSG (FSGM = 1 in SRGR2 and FSXM = 1 in PCR). If the input FSR has appropriate timing so that it can be sampled by the falling edge of CLKG, it can be used, instead, by setting FSXM = 0 and connecting FSR to FSX externally.
- The sample rate generator clock drives the transmit and receive clocking (CLKRM = CLKXM = 1 in PCR). Therefore, the CLK(R/X) pin should not be driven by any other driving source.

1.4.3.2 Synchronization Examples

Figure 1–21 and Figure 1–22 show the clock and frame-synchronization operation with various polarities of CLKS (the chosen input clock) and FSR. These figures assume FWID = 0 in SRGR1, for an FSG pulse that is 1 CLKG cycle wide. The FPER bits of SRGR2 are not programmed; the period from the start of a frame-sync pulse to the start of the next pulse is determined by the arrival of the next inactive-to-active transition on the FSR pin. Each of the figures shows what happens to CLKG when it is initially synchronized and GSYNC = 1, and when it is not initially synchronized and GSYNC = 1. The second figure has a slower CLKG frequency (it has a larger divide-down value in the CLKGDV bits of SRGR1).

Figure 1–21. CLKG Synchronization and FSG Generation When GSYNC = 1 and CLKGDV = 1

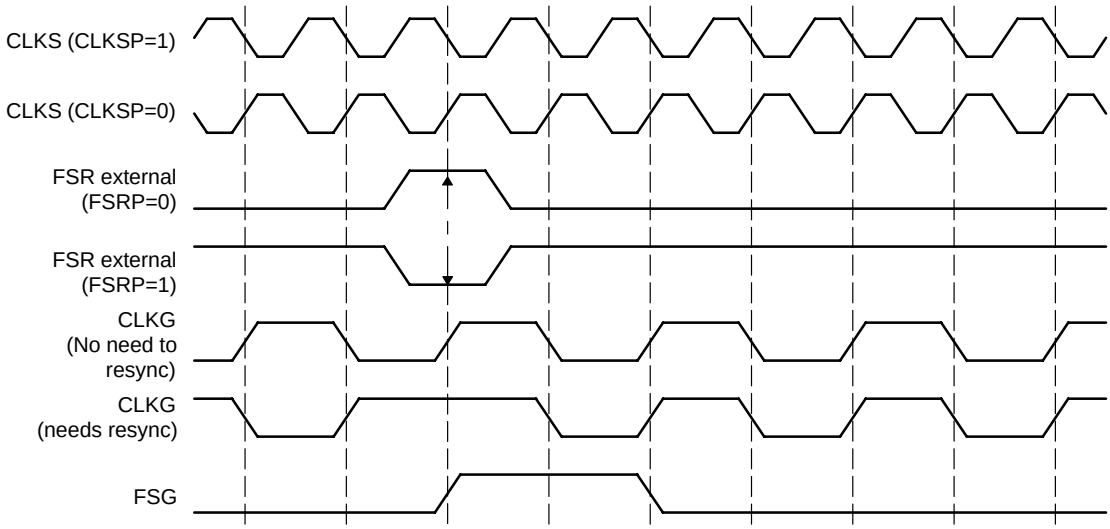
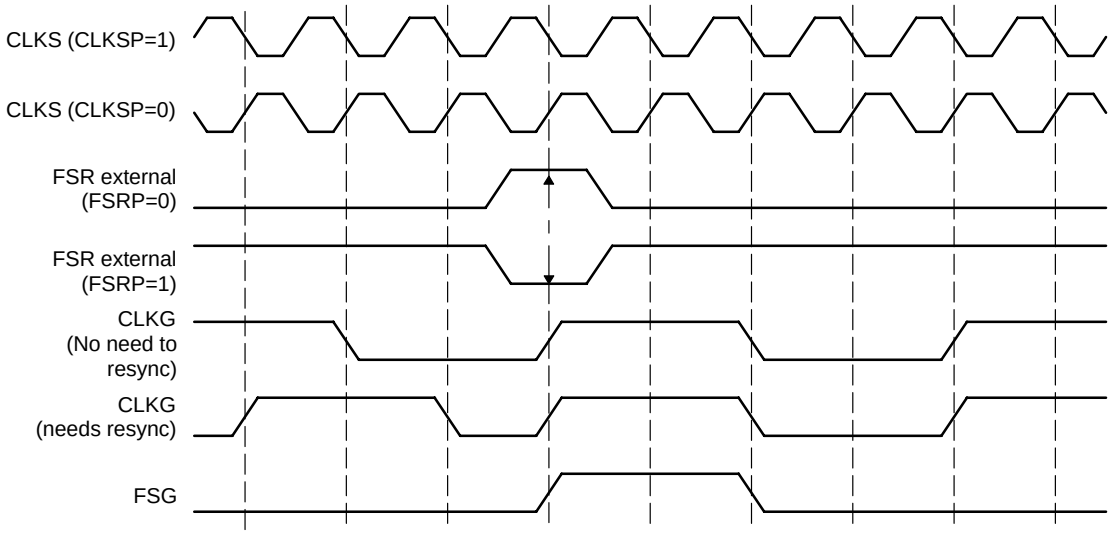


Figure 1–22. CLKG Synchronization and FSG Generation When GSYNC = 1 and CLKGDV = 3



1.4.4 Reset and Initialization Procedure for the Sample Rate Generator

To reset and initialize the sample rate generator:

1) Place the McBSP/sample rate generator in reset.

During a DSP reset, the sample rate generator, the receiver, and the transmitter reset bits ($\overline{\text{GRST}}$, $\overline{\text{RRST}}$, and $\overline{\text{XRST}}$) are automatically forced to 0. Otherwise, during normal operation, the sample rate generator can be reset by making $\overline{\text{GRST}}$ = 0 in SPCR2, provided that CLKG and/or FSG is not used by any portion of the McBSP. Depending on your system you may also want to reset the receiver ($\overline{\text{RRST}}$ = 0 in SPCR1) and reset the transmitter ($\overline{\text{XRST}}$ = 0 in SPCR2).

If $\overline{\text{GRST}}$ = 0 due to a DSP reset, CLKG is driven by the CPU clock divided by 2, and FSG is driven inactive-low. If $\overline{\text{GRST}}$ = 0 due to program code, CLKG and FSG are driven low (inactive).

2) Program registers that affect the sample rate generator.

Program the sample rate generator registers (SRGR1 and SRGR2) as required for your application. If necessary, other control registers can be loaded with desired values, provided the respective portion of the McBSP (the receiver or transmitter) is in reset.

After the sample rate generator registers are programmed, wait 2 CLKSRG cycles. This ensures proper synchronization internally.

3) Enable the sample rate generator (take it out of reset).

In SPCR2, make $\overline{\text{GRST}}$ = 1 to enable the sample rate generator.

After the sample rate generator is enabled, wait 2 CLKG cycles for the sample rate generator logic to stabilize.

On the next rising edge of CLKSRG, CLKG transitions to 1 and starts clocking with a frequency equal to

$$\text{CLKG frequency} = \frac{\text{Input clock frequency}}{(\text{CLKGDV} + 1)}$$

where the input clock is selected with the SCLKME bit of PCR and the CLKSM bit of SRGR2:

SCLKME	CLKSM	Input Clock For Sample Rate Generator
0	0	Reserved
0	1	LSPCLK
1	0	Signal on CLKR pin
1	1	Signal on CLKX pin

4) If necessary, enable the receiver and/or the transmitter.

If necessary, remove the receiver and/or transmitter from reset by setting $\overline{\text{RRST}}$ and/or XRST = 1.

5) If necessary, enable the frame-sync logic of the sample rate generator.

After the required data acquisition setup is done (DXR[1/2] is loaded with data), set $\overline{\text{FRST}}$ = 1 in SPCR2 if an internally generated frame-sync pulse is required. FSG is generated with an active-high edge after the programmed number of CLKG clocks (FPER + 1) have elapsed.

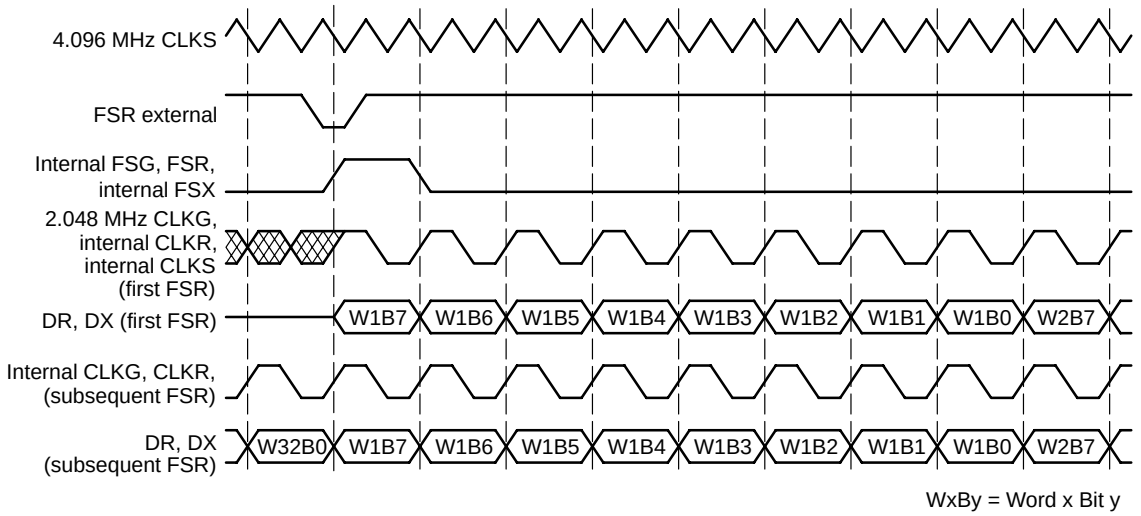
1.4.5 Sample Rate Generator Clocking Examples

This section shows three examples of using the sample rate generator to clock data during transmission and reception.

1.4.5.1 Double-Rate ST-Bus Clock

Figure 1–23 shows McBSP configuration to be compatible with the Mitel ST-Bus. Note that this operation is running at maximum frame frequency (described on page 1-17).

Figure 1–23. ST-BUS and MVIP Clocking Example



For this McBSP configuration:

- DLB = 0: Digital loopback mode off, CLKSTP = 00b: Clock stop mode off, and CLKRM/CLKXM = 1: Internal CLKR/CLKX generated internally by sample rate generator
- GSYNC = 1: Synchronize CLKG with external frame-sync signal input on FSR pin. CLKG is not synchronized until the frame-sync signal is active. FSR is regenerated internally to form a minimum pulse width.

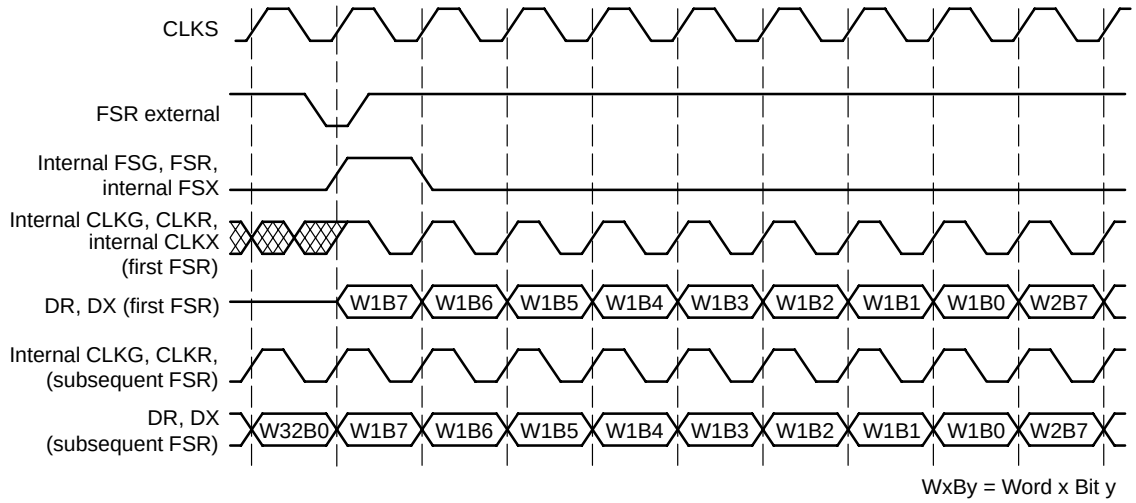
- ❑ SCLKME = 0 and CLKSM = 1: External clock signal at CLKS pin drives the sample rate generator
- ❑ CLKSP = 1: Falling edge of CLKS generates CLKG and thus internal CLK(R/X)
- ❑ CLKGDV = 1: Frequency of receive clock (shown as CLKR) is half CLKS frequency
- ❑ FSRP/FSXP = 1: Active-low frame-sync pulse
- ❑ RFRLEN1/XFRLEN1 = 11111b: 32 words per frame
- ❑ RWDLEN1/XWDLEN1 = 0: 8 bits per word
- ❑ RPHASE/XPHASE = 0: Single-phase frame and thus (R/X)FRLEN2 and (R/X)WDLEN2 are ignored
- ❑ RDATDLY/XDATDLY = 0: No data delay

1.4.5.2 Single-Rate ST-Bus Clock

The example in Figure 1–24 is the same as the double-rate ST-bus clock example in section 1.4.5.1 except that:

- ❑ CLKGDV = 0: CLKS drives internal CLK(R/X) without any divide down (single-rate clock).
- ❑ CLKSP = 0: Rising edge of CLKS generates CLKG and internal CLK(R/X)

Figure 1–24. Single-Rate Clock Example



The rising edge of CLKS is used to detect the external FSR pulse, which resynchronizes the internal McBSP clocks and generates a frame-sync pulse

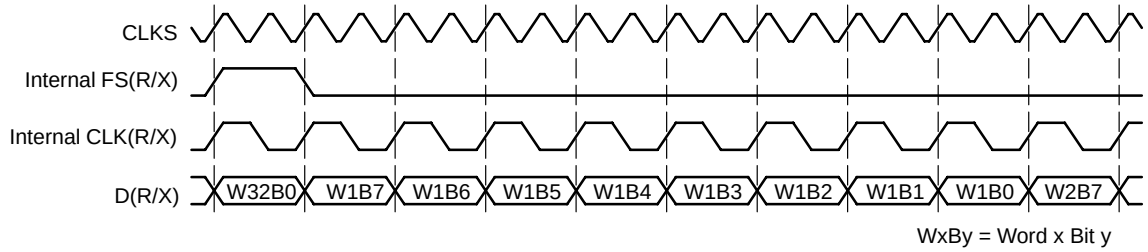
for internal use. The internal frame-sync pulse is generated so that it is wide enough to be detected on the falling edge of internal clocks.

1.4.5.3 Other Double-Rate Clock

The example in Figure 1–25 is the same as the double-rate ST-bus clock example in section 1.4.5.1 except that:

- CLKSP = 0: Rising edge of CLKS generates CLKG and thus CLK(R/X)
- CLKGDV = 1: Frequency of CLKG (and thus internal CLKR and internal CLKX) is half CLKS frequency
- FSRM/FSXM = 0: Frame synchronization is externally generated. The frame-sync pulse is wide enough to be detected.
- GSYNC = 0: CLKS drives CLKG. CLKG runs freely; it is not resynchronized by a pulse on the FSR pin.
- FSRP/FSXP = 0: Active-high input frame-sync signal
- RDATDLY/XDATDLY = 1: Data delay of one bit

Figure 1–25. Double-Rate Clock Example



1.5 McBSP Exception/Error Conditions

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

- **Receiver Overrun (RFULL = 1)**. This occurs when DRR1 has not been read since the last RBR-to-DRR copy. Consequently, the receiver does not copy a new word from the RBR(s) to the DRR(s), and the RSR(s) are now full with another new word shifted in from DR. Therefore, RFULL = 1 indicates an error condition wherein any new data that may arrive at this time on DR will replace the contents of the RSR(s), and thus, the previous word is lost. The RSR(s) continue to be overwritten as long as new data arrives on DR and DRR1 is not read. For more details about overrun in the receiver, see page 1-39.

- **Unexpected Receive Frame-Sync Pulse (RSYNCERR = 1)**. This occurs during reception when RFIG = 0 and an unexpected frame-sync pulse occurs. An unexpected frame-sync pulse is one that begins the next frame transfer before all the bits of the current frame have been received. Such a pulse causes data reception to abort and restart. If new data has been copied into the RBR(s) from the RSR(s) since the last RBR-to-DRR copy, this new data in the RBR(s) is lost. This is because no RBR-to-DRR copy occurs; the reception has been restarted. For more details about receive frame-sync errors, see page 1-40.

- **Transmitter Data Overwrite**. This occurs when the CPU overwrites data in the DXR(s) before the data is copied to the XSR(s). The overwritten data never reaches the DX pin. For more details about overwrite in the transmitter, see page 1-43.

- **Transmitter Underflow ($\overline{\text{XEMPTY}}$ = 0)**. If a new frame-sync signal arrives before new data is loaded into DXR1, the previous data in the DXR(s) is sent again. This will continue for every new frame-sync pulse that arrives until DXR1 is loaded with new data. For more details about underflow in the transmitter, see page 1-44.

- **Unexpected Transmit Frame-Synch Pulse (XSYNCERR = 1)**. This occurs during transmission when XFIG = 0 and an unexpected frame-sync pulse occurs. An unexpected frame-sync pulse is one that begins the next frame transfer before all the bits of the current frame have been transferred. Such a pulse causes the current data transmission to abort and restart. If new data has been written to the DXR(s) since the last DXR-to-XSR copy, the current value in the XSR(s) is lost. For more details about transmit frame-sync errors, see page 1-46.

1.5.1 Overrun in the Receiver

RFULL = 1 in SPCR1 indicates that the receiver has experienced overrun and is in an error condition. RFULL is set when all of the following conditions are met:

1) DRR1 has not been read since the last RBR-to-DRR copy (RRDY = 1).

2) RBR1 is full and an RBR-to-DRR copy has not occurred.

3) RSR1 is full and an RSR1-to-RBR copy has not occurred.

As described in the section on McBSP reception (page 1-21), data arriving on DR is continuously shifted into RSR1 (for word length of 16 bits or smaller) or RSR2 and RSR1 (for word length larger than 16 bits). Once a complete word is shifted into the RSR(s), an RSR-to-RBR copy can occur only if the previous data in RBR1 has been copied to DRR1. The RRDY bit is set when new data arrives in DRR1 and is cleared when that data is read from DRR1. Until RRDY = 0, the next RBR-to-DRR copy will not take place, and the data is held in the RSR(s). New data arriving on the DR pin is shifted into RSR(s), and the previous content of the RSR(s) is lost.

You can prevent the loss of data if DRR1 is read no later than 2.5 cycles before the end of the third word is shifted into the RSR1.

Important: If both DRRs are needed (word length larger than 16 bits), the CPU must read from DRR2 first and then from DRR1. As soon as DRR1 is read, the next RBR-to-DRR copy is initiated. If DRR2 is not read first, the data in DRR2 is lost.

Note that after the receiver starts running from reset, a minimum of three words must be received before RFULL is set. Either of the following events clears the RFULL bit and allows subsequent transfers to be read properly:

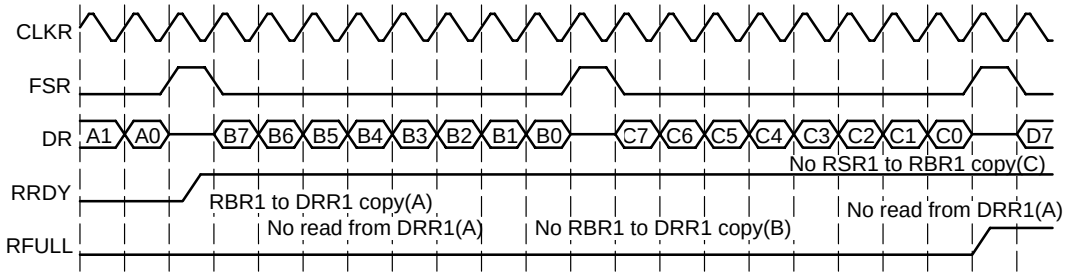
- The CPU reads DRR1.
- The receiver is reset individually ($\overline{\text{RRST}}$ = 0) or as part of a DSP reset.

Another frame-sync pulse is required to restart the receiver.

1.5.1.1 Example of the Overrun Condition

Figure 1–26 shows the receive overrun condition. Because serial word A is not read from DRR1 before serial word B arrives in RBR1, B is not transferred to DRR1 yet. Another new word (C) arrives and RSR1 is full with this data. DRR1 is finally read, but not earlier than 2.5 cycles before the end of word C. Therefore, new data (D) overwrites word C in RSR1. If DRR1 is not read in time, the next word can overwrite D.

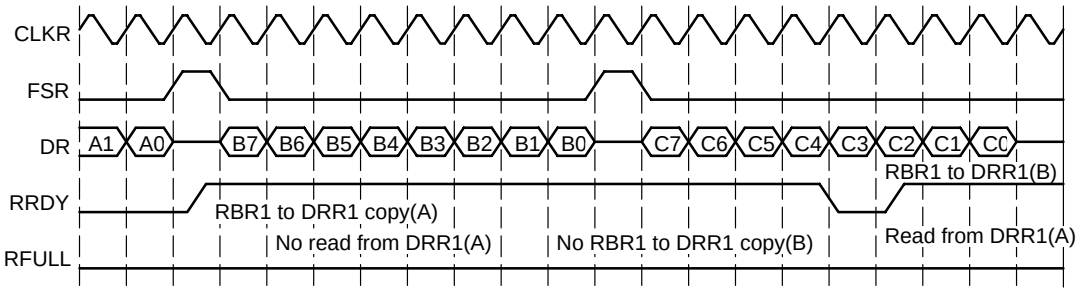
Figure 1–26. Overrun in the McBSP Receiver



1.5.1.2 Example of Preventing the Overrun Condition

Figure 1–27 shows the case where RFULL is set, but the overrun condition is prevented by a read from DRR1 at least 2.5 cycles before the next serial word (C) is completely shifted into RSR1. This ensures that an RBR1-to-DRR1 copy of word B occurs before receiver attempts to transfer word C from RSR1 to RBR1.

Figure 1–27. Overrun Prevented in the McBSP Receiver



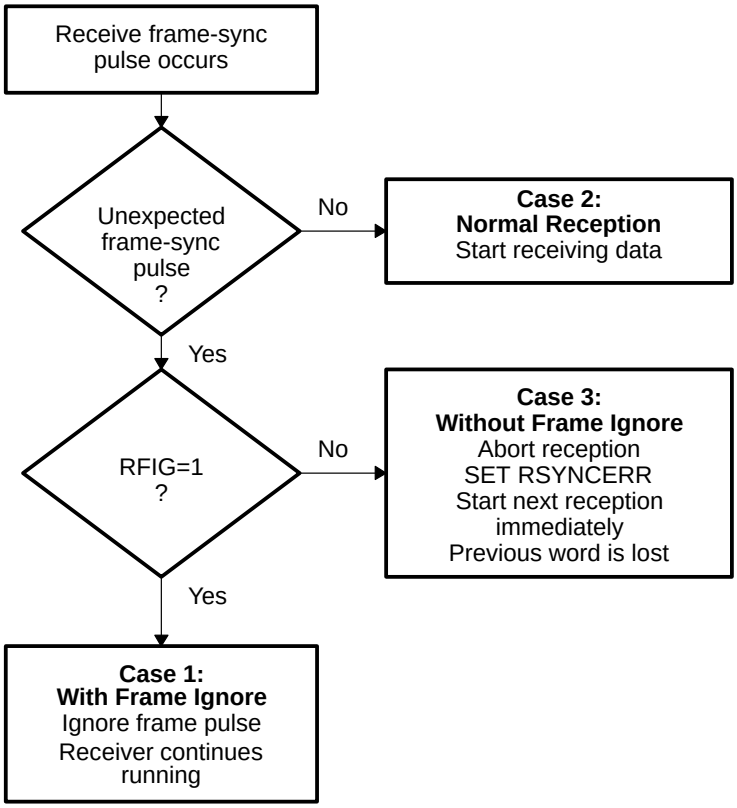
1.5.2 Unexpected Receive Frame-Sync Pulse

Section 1.5.2.1 shows how the McBSP responds to any receive frame-sync pulses, including an unexpected pulse. Sections 1.5.2.2 and 1.5.2.3 show an examples of a frame-sync error and an example of how to prevent such an error, respectively.

1.5.2.1 Possible Responses to Receive Frame-Sync Pulses

Figure 1–28 shows the decision tree that the receiver uses to handle all incoming frame-sync pulses. The figure assumes that the receiver has been started ($\overline{\text{RRST}}$ = 1 in SPCR1). Case 3 in the figure is the case in which an error occurs.

Figure 1–28. Possible Responses to Receive Frame-Sync Pulses



Any one of three cases can occur:

- **Case 1:** Unexpected internal FSR pulses with RFIG = 1 in RCR2. Receive frame-sync pulses are ignored, and the reception continues.

- **Case 2:** Normal serial port reception. Reception continues normally because the frame-sync pulse is not unexpected. There are three possible reasons why a receive operation might *not* be in progress when the pulse occurs:

  - The FSR pulse is the first after the receiver is enabled ($\overline{\text{RRST}}$ = 1 in SPCR1).

  - The FSR pulse is the first after DRR[1,2] is read, clearing a receiver full (RFULL = 1 in SPCR1) condition.

- The serial port is in the interpacket intervals. The programmed data delay for reception (programmed with the RDATDLY bits in RCR2) may start during these interpacket intervals for the first bit of the next word to be received. Thus, at maximum frame frequency, frame synchronization can still be received 0 to 2 clock cycles before the first bit of the synchronized frame.

❑ **Case 3:** Unexpected receive frame synchronization with RFIG = 0 (frame-sync pulses not ignored). Unexpected frame-sync pulses can originate from an external source or from the internal sample rate generator.

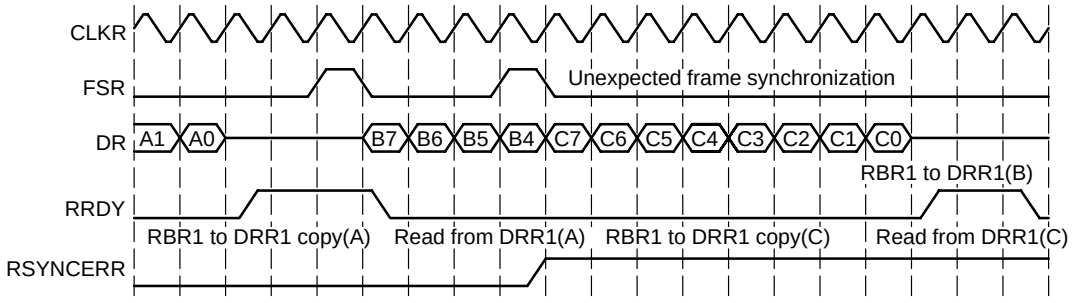
If a frame-sync pulse starts the transfer of a new frame before the current frame is fully received, this pulse is treated as an unexpected frame-sync pulse, and the receiver sets the receive frame-sync error bit (RSYNCERR) in SPCR1. RSYNCERR can be cleared only by a receiver reset or by a write of 0 to this bit.

If you want the McBSP to notify the CPU of receive frame-sync errors, you can set a special receive interrupt mode with the RINTM bits of SPCR1. When RINTM = 11b, the McBSP sends a receive interrupt (RINT) request to the CPU each time that RSYNCERR is set.

1.5.2.2 Example of an Unexpected Receive Frame-Sync Pulse

Figure 1–29 shows an unexpected receive frame-sync pulse during normal operation of the serial port, with time intervals between data packets. When the unexpected frame-sync pulse occurs, the RSYNCERR bit is set, the reception of data B is aborted, and the reception of data C begins. In addition, if RINTM = 11b, the McBSP sends a receive interrupt (RINT) request to the CPU.

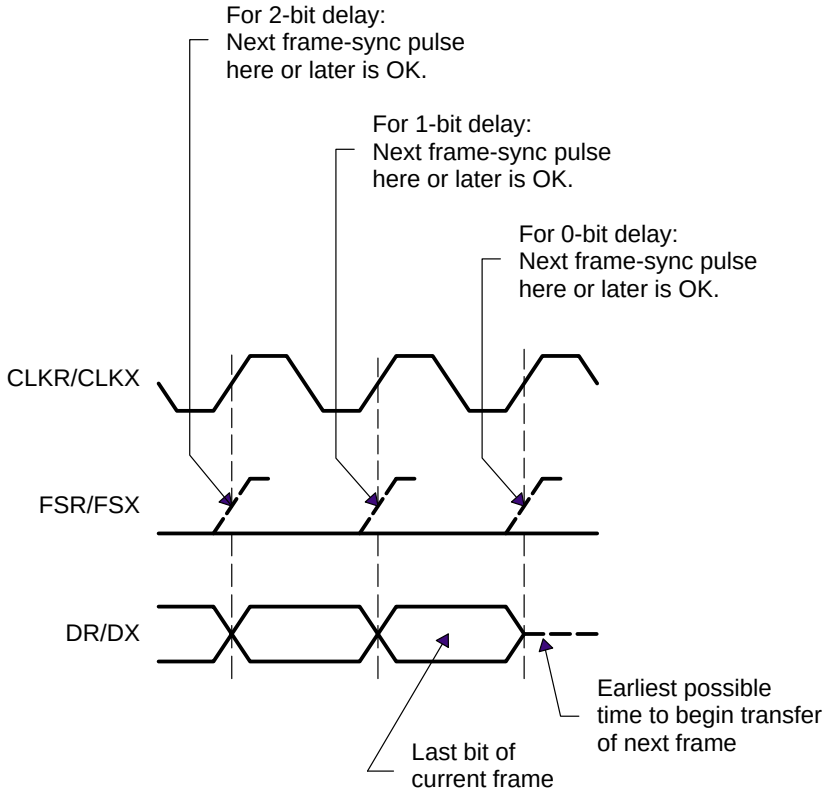
Figure 1–29. An Unexpected Frame-Sync Pulse During a McBSP Reception



1.5.2.3 Preventing Unexpected Receive Frame-Sync Pulses

Each frame transfer can be delayed by 0, 1, or 2 CLKR cycles, depending on the value in the RDATDLY bits of RCR2. For each possible data delay, Figure 1–30 shows when a new frame-sync pulse on FSR can safely occur relative to the last bit of the current frame.

Figure 1–30. Proper Positioning of Frame-Sync Pulses



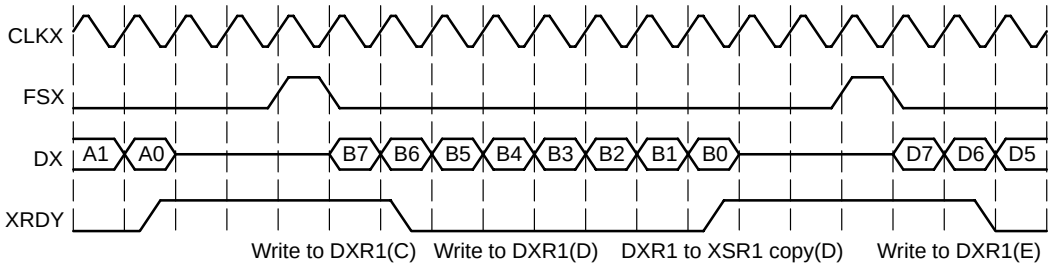
1.5.3 Overwrite in the Transmitter

As described in the section on McBSP transmission (page 1-22), after the CPU writes data to the DXR(s), the transmitter must then copy that data to the XSR(s) and then shift each bit from the XSR(s) to the DX pin. If new data is written to the DXR(s) before the previous data is copied to the XSR(s), the previous data in the DXR(s) is overwritten and thus lost.

1.5.3.1 Example of the Overwrite Condition

Figure 1–31 shows what happens if the data in DXR1 is overwritten before being transmitted. Initially, DXR1 is loaded with data C. A subsequent write to DXR1 overwrites C with D before C is copied to XSR1. Thus, C is never transmitted on DX.

Figure 1–31. Data in the McBSP Transmitter Overwritten and Thus Not Transmitted



1.5.3.2 Preventing Overwrites

You can prevent CPU overwrites by making the CPU:

- Poll for XRDY = 1 in SPCR2 before writing to the DXR(s). XRDY is set when data is copied from DXR1 to XSR1 and is cleared when new data is written to DXR1.
- Wait for a transmit interrupt (XINT) before writing to the DXR(s). When XINTM = 00b in SPCR2, the transmitter sends XINT to the CPU each time XRDY is set.

1.5.4 Underflow in the Transmitter

The McBSP indicates a transmitter empty (or underflow) condition by clearing the $\overline{\text{XEMPTY}}$ bit in SPCR2. Either of the following events activates $\overline{\text{XEMPTY}}$ ($\overline{\text{XEMPTY}}$ = 0):

- DXR1 has not been loaded since the last DXR-to-XSR copy, and all bits of the data word in the XSR(s) have been shifted out on the DX pin.
- The transmitter is reset (by forcing $\overline{\text{XRST}}$ = 0 in SPCR2, or by a DSP reset) and is then restarted.

In the underflow condition, the transmitter continues to transmit the old data that is in the DXR(s) for every new transmit frame-sync signal until a new value is loaded into DXR1 by the CPU.

Note:

If both DXRs are needed (word length larger than 16 bits), the CPU must load DXR2 first and then load DXR1. As soon as DXR1 is loaded, the contents of both DXRs are copied to the transmit shift registers (XSRs). If DXR2 is not loaded first, the previous content of DXR2 is passed to the XSR2.

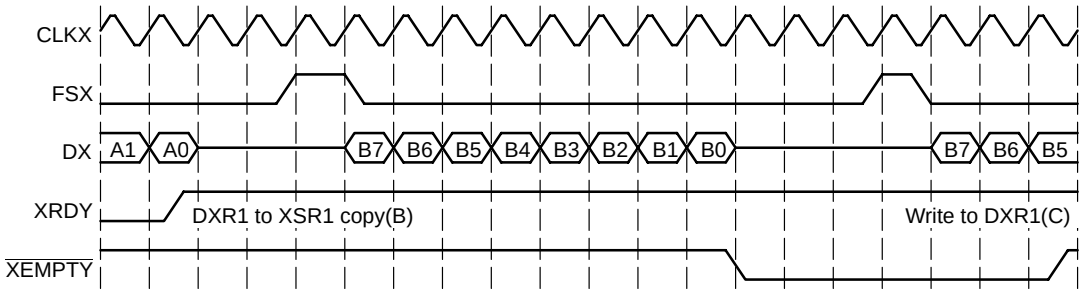
$\overline{\text{XEMPTY}}$ is deactivated ($\overline{\text{XEMPTY}}$ = 1) when a new word in DXR1 is transferred to XSR1. If FSXM = 1 in PCR and FSGM = 0 in SRGR2, the transmitter generates a single internal FSX pulse in response to a DXR-to-XSR copy. Otherwise, the transmitter waits for the next frame-sync pulse before sending out the next frame on DX.

When the transmitter is taken out of reset ($\overline{\text{XRST}}$ = 1), it is in a transmitter ready (XRDY = 1 in SPCR2) and transmitter empty ($\overline{\text{XEMPTY}}$ = 0) state. If DXR1 is loaded by the CPU before internal FSX goes active high, a valid DXR-to-XSR transfer occurs. This allows for the first word of the first frame to be valid even before the transmit frame-sync pulse is generated or detected. Alternatively, if a transmit frame-sync pulse is detected before DXR1 is loaded, zeros will be output on DX.

1.5.4.1 Example of the Underflow Condition

Figure 1–32 shows an underflow condition. After B is transmitted, DXR1 is not reloaded before the subsequent frame-sync pulse. Thus, B is again transmitted on DX.

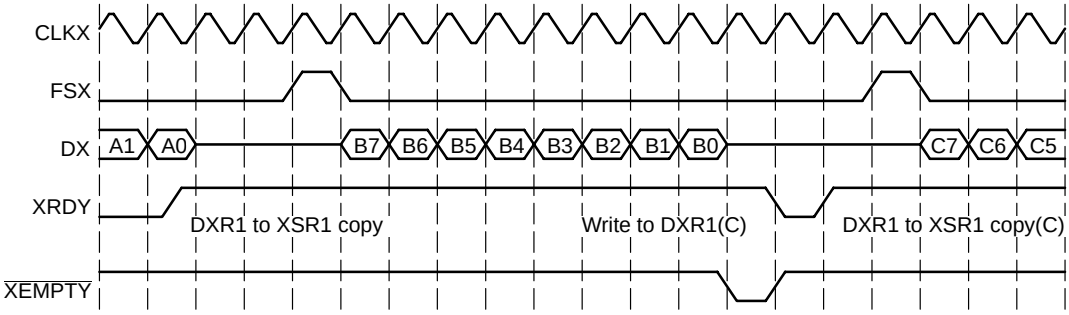
Figure 1–32. Underflow During McBSP Transmission



1.5.4.2 Example of Preventing the Underflow Condition

Figure 1–33 shows the case of writing to DXR1 just before an underflow condition would otherwise occur. After B is transmitted, C is written to DXR1 before the next frame-sync pulse. As a result, there is no underflow; B is not transmitted twice.

Figure 1–33. Underflow Prevented in the McBSP Transmitter



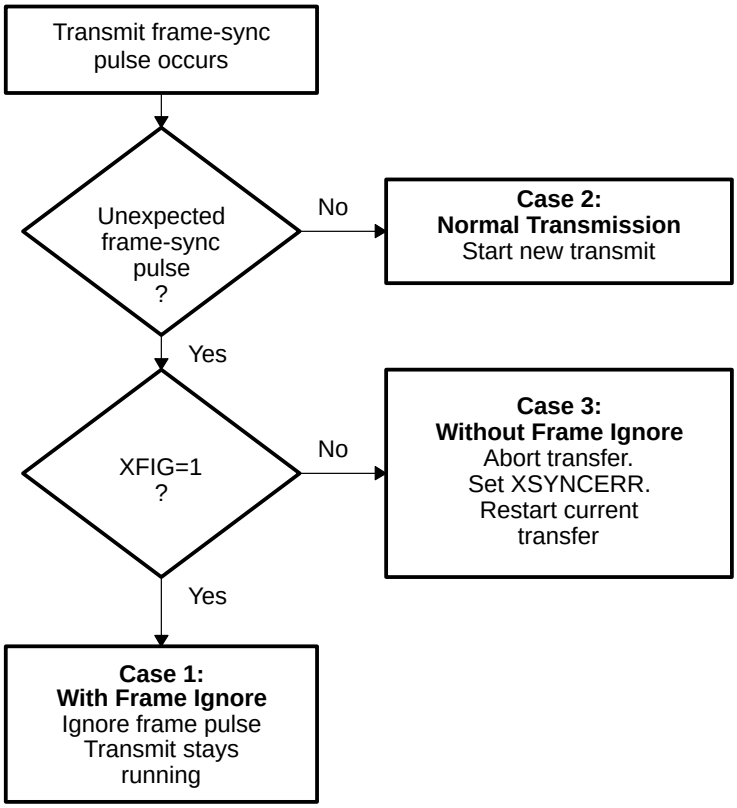
1.5.5 Unexpected Transmit Frame-Sync Pulse

Section 1.5.5.1 shows how the McBSP responds to any transmit frame-sync pulses, including an unexpected pulse. Sections 1.5.5.2 and 1.5.5.3 show an examples of a frame-sync error and an example of how to prevent such an error, respectively.

1.5.5.1 Possible Responses to Transmit Frame-Sync Pulses

Figure 1–34 shows the decision tree that the transmitter uses to handle all incoming frame-sync pulses. The figure assumes that the transmitter has been started ($\overline{\text{XRST}}$ = 1 in SPCR2). Case 3 in the figure is the case in which an error occurs.

Figure 1–34. Possible Responses to Transmit Frame-Sync Pulses



Any one of three cases can occur:

- **Case 1:** Unexpected internal FSX pulses with XFIG = 1 in XCR2. Transmit frame-sync pulses are ignored, and the transmission continues.

- **Case 2:** Normal serial port transmission. Transmission continues normally because the frame-sync pulse is not unexpected. There are two possible reasons why a transmit operations might *not* be in progress when the pulse occurs:

  This FSX pulse is the first after the transmitter is enabled ($\overline{\text{XRST}}$ = 1).

  The serial port is in the interpacket intervals. The programmed data delay for transmission (programmed with the XDATDLY bits of XCR2) may start during these interpacket intervals before the first bit of the previous word is transmitted. Thus, at maximum packet frequency, frame synchronization can still be received 0 to 2 clock cycles before the first bit of the synchronized frame.

- **Case 3:** Unexpected transmit frame synchronization with XFIG = 0 (frame-sync pulses not ignored). Unexpected frame-sync pulses can originate from an external source or from the internal sample rate generator.

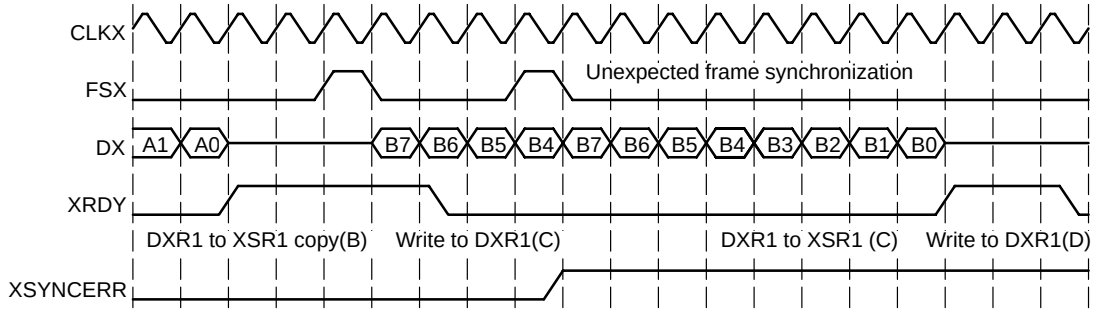
  If a frame-sync pulse starts the transfer of a new frame before the current frame is fully transmitted, this pulse is treated as an unexpected frame-sync pulse, and the transmitter sets the transmit frame-sync error bit (XSYNCERR) in SPCR2. XSYNCERR can be cleared only by a transmitter reset or by a write of 0 to this bit.

  If you want the McBSP to notify the CPU of frame-sync errors, you can set a special transmit interrupt mode with the XINTM bits of SPCR2. When XINTM = 11b, the McBSP sends a transmit interrupt (XINT) request to the CPU each time that XSYNCERR is set.

1.5.5.2 Example of an Unexpected Transmit Frame-Sync Pulse

Figure 1–35 shows an unexpected transmit frame-sync pulse during normal operation of the serial port, with intervals between the data packets. When the unexpected frame-sync pulse occurs, the XSYNCERR bit is set and because no new data has been passed to XSR1 yet, the transmission of data B is restarted. In addition, if XINTM = 11b, the McBSP sends a transmit interrupt (XINT) request to the CPU.

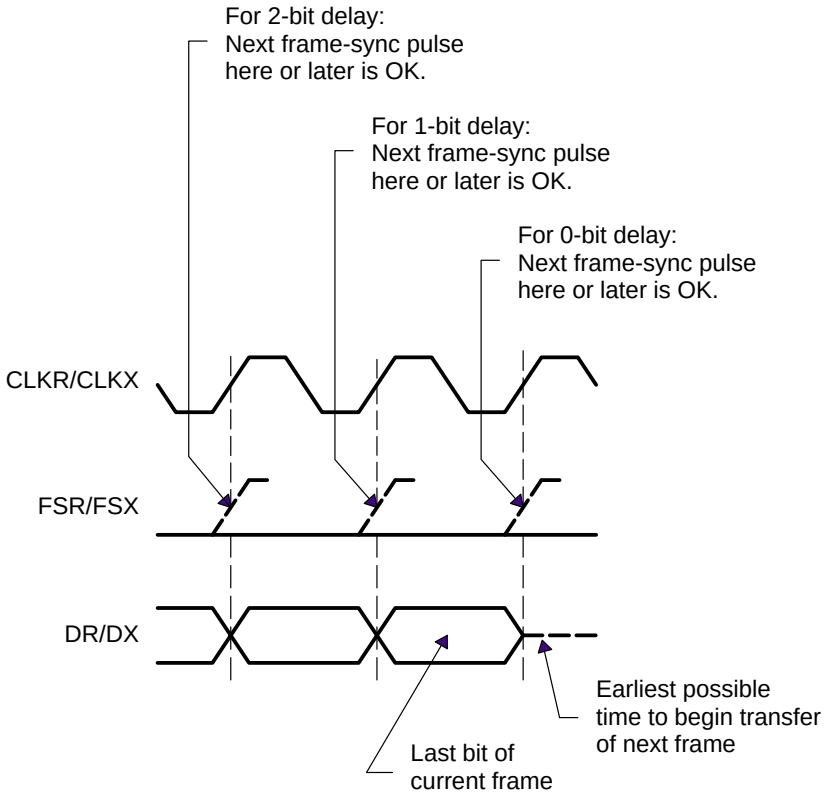
Figure 1–35. An Unexpected Frame-Sync Pulse During a McBSP Transmission



1.5.5.3 Preventing Unexpected Transmit Frame-Sync Pulses

Each frame transfer can be delayed by 0, 1, or 2 CLKX cycles, depending on the value in the XDATDLY bits of XCR2. For each possible data delay, Figure 1–36 shows when a new frame-sync pulse on FSX can safely occur relative to the last bit of the current frame.

Figure 1–36. Proper Positioning of Frame-Sync Pulses



Chapter 2

Multichannel Selection Modes

This chapter explains how to assign blocks to partitions for multichannel selection.

Topic **Page**

2.1 Channels, Blocks, and Partitions 2-2

2.2 A-bis Mode 2-13

2.3 SPI Protocol 2-15

2.1 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 0–15	Block 4: Channels 64–79
Block 1: Channels 16–31	Block 5: Channels 80–95
Block 2: Channels 32–47	Block 6: Channels 96–111
Block 3: Channels 48–63	Block 7: Channels 112–127

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–H).

2.1.1 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.

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.

The McBSP has one receive multichannel selection mode (described in section 2.1.5) and three transmit multichannel selection modes (described in section 2.1.6).

2.1.2 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 (in RFRLEN1/XFRLEN1) that includes the highest-numbered channel that will 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.

2.1.3 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.

2.1.3.1 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, 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.

For reception:

- ❑ Assign an even-numbered channel block (0, 2, 4, or 6) to receive partition A by writing to the RPABLK bits. In the receive multichannel selection mode (described in section 2.1.5), the channels in this partition are controlled by receive channel enable register A (RCERA).
- ❑ Assign an odd-numbered block (1, 3, 5, or 7) to receive partition B with the RPBBLK bits. In the receive multichannel selection mode, the channels in this partition are controlled by receive channel enable register B (RCERB).

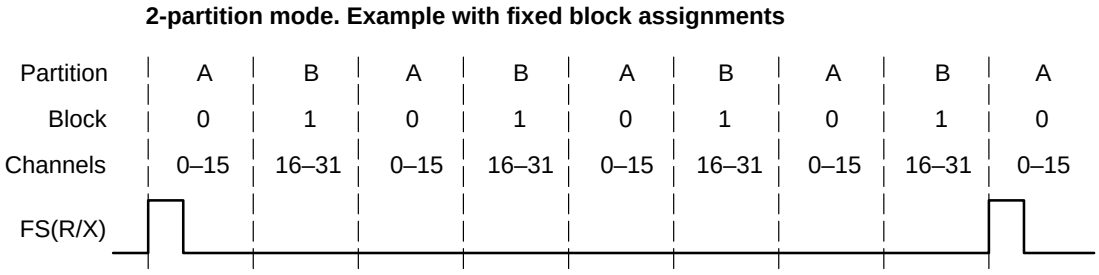
For transmission:

- ❑ Assign an even-numbered channel block (0, 2, 4, or 6) to transmit partition A by writing to the XPABLK bits. In one of the transmit multichannel selection modes (described in section 2.1.6), the channels in this partition are controlled by transmit channel enable register A (XCERA).
- ❑ Assign an odd-numbered block (1, 3, 5, or 7) to transmit partition B with the XPBBLK bits. In one of the transmit multichannel selection modes, the

channels in this partition are controlled by transmit channel enable register B (XCERB).

Figure 2–1 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 2–1. Alternating Between the Channels of Partition A and the Channels of Partition B



As explained next, you can dynamically change which blocks of channels are assigned to the partitions.

2.1.3.2 Reassigning Blocks During Reception/Transmission

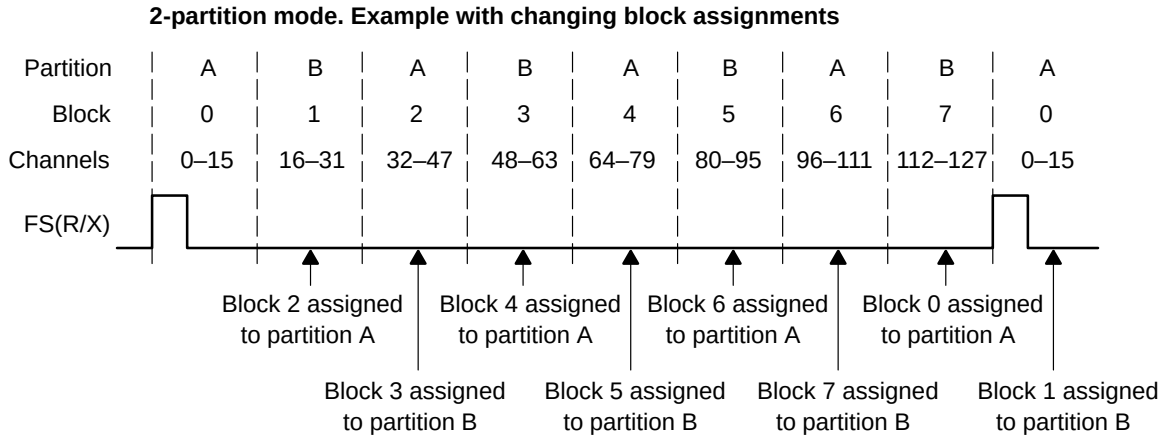
If you want to use more than 32 channels, you can change which channel blocks are assigned to partitions A and B during the course of a data transfer. However, these changes must be carefully timed. While a partition is being transferred, its the associated block assignment bits cannot be modified, and 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)CERA 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 2–2 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 2–2. Reassigning Channel Blocks Throughout a McBSP Data Transfer



2.1.4 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 channels 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 2–1 and Table 2–2. These assignments cannot be changed. The tables also show the registers used to control the channels in the partitions.

Table 2–1. Receive Channel Assignment and Control When Eight Receive Partitions Are Used

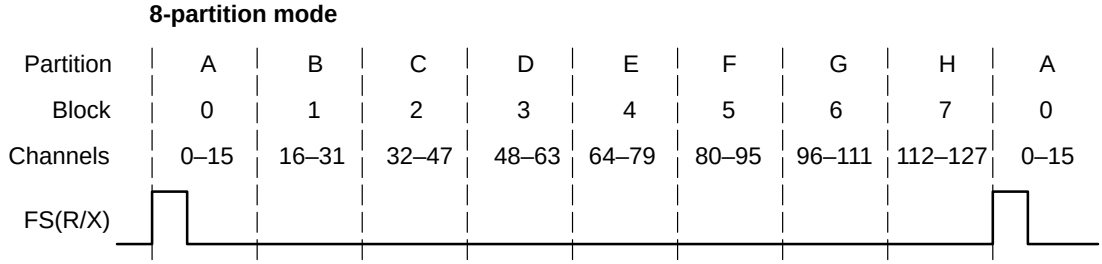
Receive Partition	Assigned Block of Receive Channels	Register Used For Channel Control
A	Block 0: channels 0 through 15	RCERA
B	Block 1: channels 16 through 31	RCERB
C	Block 2: channels 32 through 47	RCERC
D	Block 3: channels 48 through 63	RCERD
E	Block 4: channels 64 through 79	RCERE
F	Block 5: channels 80 through 95	RCERF
G	Block 6: channels 96 through 111	RCERG
H	Block 7: channels 112 through 127	RCERH

Table 2–2. 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	XCERA
B	Block 1: channels 16 through 31	XCERB
C	Block 2: channels 32 through 47	XCERC
D	Block 3: channels 48 through 63	XCERD
E	Block 4: channels 64 through 79	XCERE
F	Block 5: channels 80 through 95	XCERF
G	Block 6: channels 96 through 111	XCERG
H	Block 7: channels 112 through 127	XCERH

Figure 2–3 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 2–3. McBSP Data Transfer in the 8-Partition Mode



2.1.5 Receive Multichannel Selection Mode

The RMCM bit of MCR1 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 receive channel enable registers (RCERs). The way channels are assigned to the RCERs depends on the number of receive channel partitions (2 or 8), as defined by the RMCME bit of MCR1.
- ❏ If a receive channel is disabled, any bits received in that channel are passed only as far as the receive buffer register(s) (RBR(s)). The receiver does not copy the content of the RBR(s) to the DRR(s), and as a result, does not set the receiver ready bit (RRDY). Therefore, no receive FIFO event (REVT) is generated, and if the receiver interrupt mode depends on RRDY (RINTM = 00b), 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

2.1.6 Transmit Multichannel Selection Modes

The XMCM bits of XCR2 determine whether all channels or only selected channels are enabled and unmasked for transmission. More details on enabling and masking are in section 2.1.7. The McBSP has three transmit multichannel selection modes (XMCM = 01b, XMCM = 10b, and XMCM = 11b), which are described in the following table:

Table 2–3. Selecting a Transmit Multichannel Selection Mode With the XMCM Bits

XMCM	Transmit Multichannel Selection Mode
00b	No transmit multichannel selection mode is on. All channels are enabled and unmasked. No channels can be disabled or masked.
01b	All channels are disabled unless they are selected in the appropriate transmit channel enable registers (XCERs) If enabled, a channel in this mode is also unmasked. The XMCME bit of MCR2 determines whether 32 channels or 128 channels are selectable in XCERs.
10b	All channels are enabled, but they are masked unless they are selected in the appropriate transmit channel enable registers (XCERs). The XMCME bit of MCR2 determines whether 32 channels or 128 channels are selectable in XCERs.
11b	This mode is used for symmetric transmission and reception. All channels are disabled for transmission unless they are enabled for reception in the appropriate receive channel enable registers (RCERs). Once enabled, they are masked unless they are also selected in the appropriate transmit channel enable registers (XCERs). The XMCME bit of MCR2 determines whether 32 channels or 128 channels are selectable in RCERs and XCERs.

As an example of how the McBSP behaves in a transmit multichannel selection mode, suppose that XMCM = 01b (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

2.1.7 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(s) (DXR(s)) to the transmit shift registers (XSR(s)).

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 of SPCR2 is not set. Therefore, no transmit FIFO event (XEVT) is generated, and if the transmit interrupt mode depends on XRDY (XINTM = 00b in SPCR2), no interrupt is generated.

The $\overline{\text{XEMPTY}}$ bit of SPCR2 is not affected.

Unmasked channel A channel that is not masked. Data in the XSR(s) is shifted out on the DX pin.

2.1.8 Activity on McBSP Pins for Different Values of XMCM

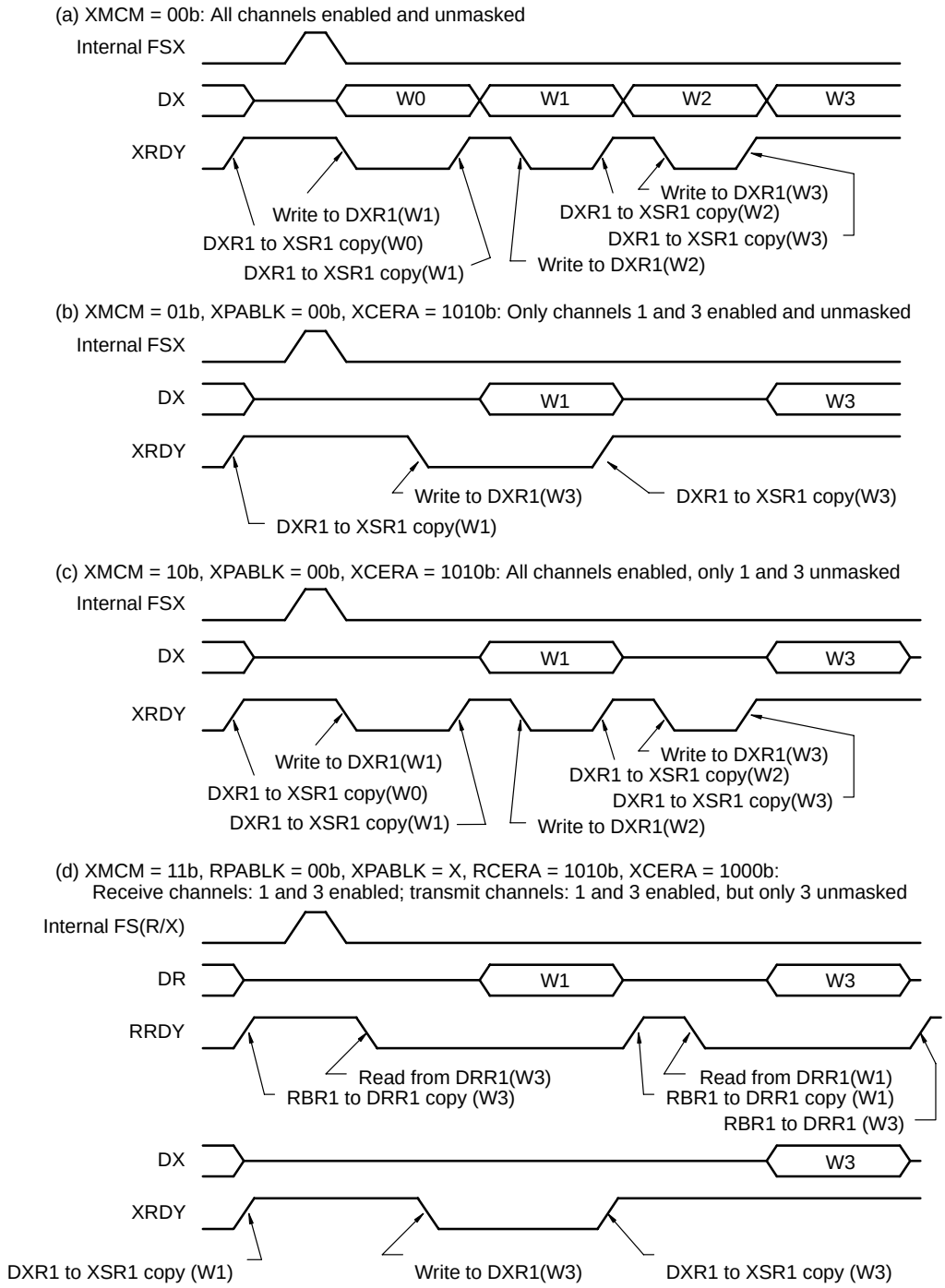
Figure 2–4 shows the activity on the McBSP pins for the various XMCM values. In all cases, the transmit frame is configured as follows:

- XPHASE = 0: Single-phase frame (required for multichannel selection modes)
- XFRLEN1 = 0000011b: 4 words per frame
- XWDLEN1 = 000b: 8 bits per word
- XMCME = 0: 2-partition mode (only partitions A and B used)

In the case where XMCM = 11b, 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 the figure, 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 2–4. Activity on McBSP Pins for the Possible Values of XMCM



2.1.9 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 RINTM = 01b. In any of the transmit multichannel selection modes, a transmit interrupt (XINT) request is generated at the end of each block transfer if XINTM = 01b. When RINTM/XINTM = 1b, no interrupt is generated unless a multichannel selection mode is on.

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.

2.2 A-bis Mode

In the A-bis mode (ABIS = 1 in SPCR1), the McBSP can receive and transmit up to 1024 bits on a PCM link. The receive section can extract all 1024 bits from a 1024-bit PCM frame according to a given bit-enable pattern, and generate an interrupt to the CPU when 16 enabled bits have been compacted into a word in DRR1, or when a receive frame is complete. In addition, the transmit section can expand up to 1024 bits into a 1024-bit PCM frame at a specific position, according to a given bit-enable pattern, and generate an interrupt when 16 enabled bits have been transmitted or a transmit frame is complete.

The bit-enable patterns are specified with channel enable registers A and B (RCERA and RCERB for reception, XCERA and XCERB for transmission). These registers have a different function than in the multichannel selection modes (described in section 2.1). Instead of indicating which channels will be enabled, these registers indicate which bits in the data stream will be enabled. A 1 in a given position in the (R/X)CER(A/B) register enables a corresponding bit in the receive/transmit data stream.

The A-bis mode requires a word length of 16 bits (for reception: RWDLEN1 = 010b in RCR1, for transmission: XWDLEN1 = 010b in XCR1). Otherwise, operation in the A-bis mode is undetermined.

2.2.1 A-bis Mode Receive Operation

In the A-bis mode, bits that are not enabled in the RCERA and RCERB registers are ignored and are not compacted in the receiver. Bits that are enabled are received and compacted. When 16 enabled bits have been received, the received word is copied from RSR1 to DRR1 and the McBSP generates an interrupt to the CPU. RCERA and RCERB alternate specifying the receive masking pattern for each of the 16 receive clocks. Figure 2–5 shows an example bit sequence for the receiver (in the figure, – indicates that the bit on the DR pin is ignored and thus is not passed to DRR1).

Figure 2–5. A-bis Mode Receive Operation

RCERA	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1															
RCERB																	0	0	1	0	0	1	1	1	0	0	1	0	0	1	1	1	
DR pin	1	0	1	1	0	1	0	1	0	0	1	1	0	1	0	1	0	1	1	1	0	1	1	0	1	1	0	1	1	0	1	1	
DRR1	–	0	–	1	–	1	–	1	–	0	–	1	0	–	–	1	–	–	1	–	–	1	1	0	–	–	0	–	–	0	1	1	75E3h

Note: – indicates that the bit on the DR pin is ignored and thus is not passed to DRR1.

2.2.2 A-bis Mode Transmit Operation

In the A-bis mode, only bits that are enabled in the XCERA and XCERB registers are transmitted out from the DX pin. Bits that are not enabled are not transmitted, and the DX pin is in the high-impedance state during that clock cycle. XCERA and XCERB alternate specifying the bit-enable pattern for each 16 clock cycles. When 16 enabled have been shifted out, the McBSP generates an interrupt to the CPU. Figure 2–6 shows an example bit sequence for the transmitter (in the figure, z indicates the high-impedance state).

Figure 2–6. A-bis Mode Transmit Operation

XCERA	0	1	1	1	1	1	0	0	0	1	1	0	0	1	1	1															
XCERB																	0	0	1	1	0	0	0	1	1	0	0	0	1	1	0	0
DXR1	1	0	1	1	0	1	0	1	0	0	0	1	1	1	1	1	1	0	1	1	0	1	1	0	1	1	0	0	0	0	0	0
DX pin	z	0	1	1	0	1	z	z	z	0	0	z	z	1	1	1	z	z	1	1	z	z	z	0	1	z	z	z	0	0	z	z

Note: z indicates the high-impedance state.

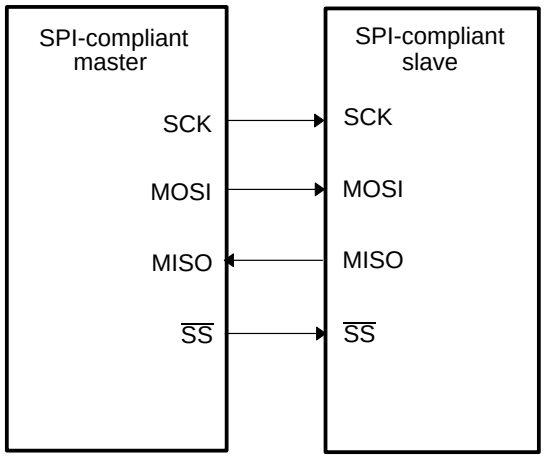
2.3 SPI Protocol

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 SS)

A typical SPI interface with a single slave device is shown in Figure 2–7.

Figure 2–7. 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 (device not sending out the clock).

In the absence of a dedicated slave-enable signal, communication between the master and slave is determined by the presence or absence of an active shift-clock. In such a configuration, the slave device must remain enabled at all times, and multiple slaves cannot be used.

2.3.1 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 signal ($\overline{SS}$).

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.

2.3.2 Bits Used to Enable and Configure the Clock Stop Mode

The bits required to configure the McBSP as an SPI device are introduced in Table 2–4. Table 2–5 shows how the various combinations of the CLKSTP bit and the polarity bits CLKXP and CLKRP create four possible clock stop mode configurations. The timing diagrams in section 2.3.3 show the effects of CLKSTP, CLKXP, and CLKRP.

Table 2–4. Bits Used to Enable and Configure the Clock Stop Mode

Bit Field	Description
CLKSTP bits of SPCR1	Use these bits to enable the clock stop mode and to select one of two timing variations. (See also Table 2–5.)
CLKXP bit of PCR	This bit determines the polarity of the CLKX signal. (See also Table 2–5.)
CLKRP bit of PCR	This bit determines the polarity of the CLKR signal. (See also Table 2–5.)
CLKXM bit of PCR	This bit determines whether CLKX is an input signal (McBSP as slave) or an output signal (McBSP as master).
XPHASE bit of XCR2	You must use a single-phase transmit frame (XPHASE = 0).
RPHASE bit of RCR2	You must use a single-phase receive frame (RPHASE = 0).
XFRLEN1 bits of XCR1	You must use a transmit frame length of 1 serial word (XFRLEN1 = 0).
RFRLEN1 bits of RCR1	You must use a receive frame length of 1 serial word (RFRLEN1 = 0).
XWDLEN1 bits of XCR1	The XWDLEN1 bits determine 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.
RWDLEN1 bits of RCR1	The RWDLEN1 bits determine 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.

Table 2–5. Effects of CLKSTP, CLKXP, and CLKRP on the Clock Scheme

Bit Settings	Clock Scheme
CLKSTP = 00b or 01b CLKXP = 0 or 1 CLKRP = 0 or 1	Clock stop mode disabled. Clock enabled for non-SPI mode.
CLKSTP = 10b CLKXP = 0 CLKRP = 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.
CLKSTP = 11b CLKXP = 0 CLKRP = 1	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.
CLKSTP = 10b CLKXP = 1 CLKRP = 0	High inactive state without delay: The McBSP transmits data on the falling edge of CLKX and receives data on the rising edge of CLKR.
CLKSTP = 11b CLKXP = 1 CLKRP = 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.

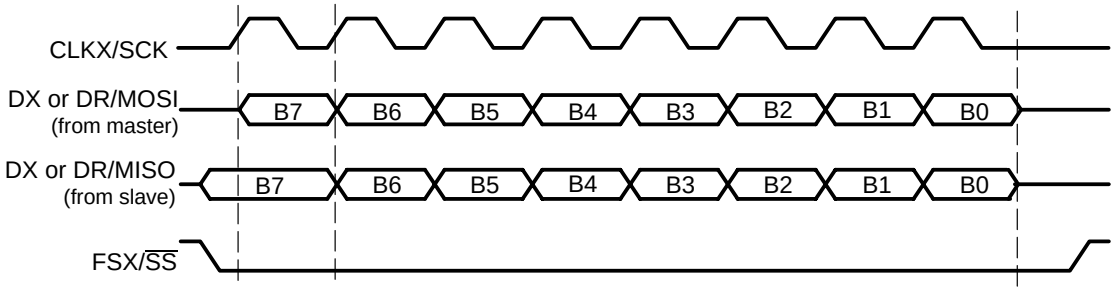
2.3.3 Clock Stop Mode Timing Diagrams

The timing diagrams for the four possible clock stop mode configurations are shown here. 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 of RCR1, and the transmit packet length is selected with the XWDLEN1 bits of XCR1. 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.

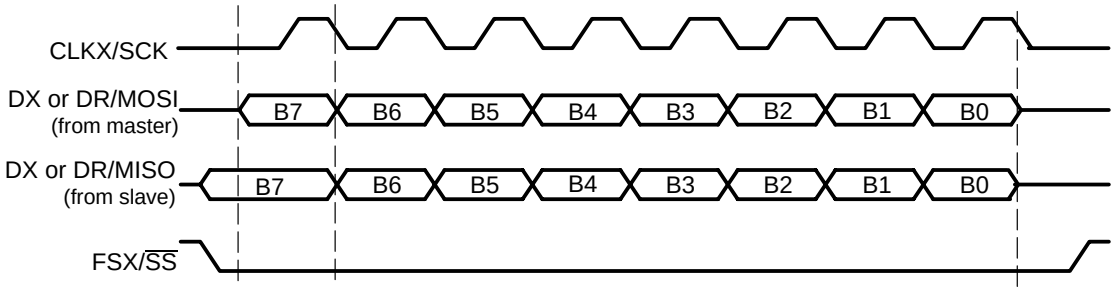
Figure 2–8. SPI Transfer With CLKSTP = 10b (no clock delay), CLKXP = 0, CLKRP = 0



Notes: 1) If the McBSP is the SPI master (CLKXM = 1), MOSI = DX. If the McBSP is the SPI slave (CLKXM = 0), MOSI = DR.

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

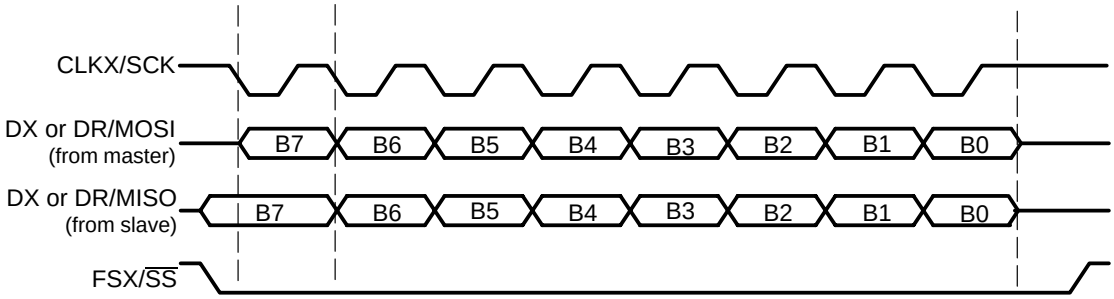
Figure 2–9. SPI Transfer With CLKSTP = 11b (clock delay), CLKXP = 0, CLKRP = 1



Notes: 1) If the McBSP is the SPI master (CLKXM = 1), MOSI = DX. If the McBSP is the SPI slave (CLKXM = 0), MOSI = DR.

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

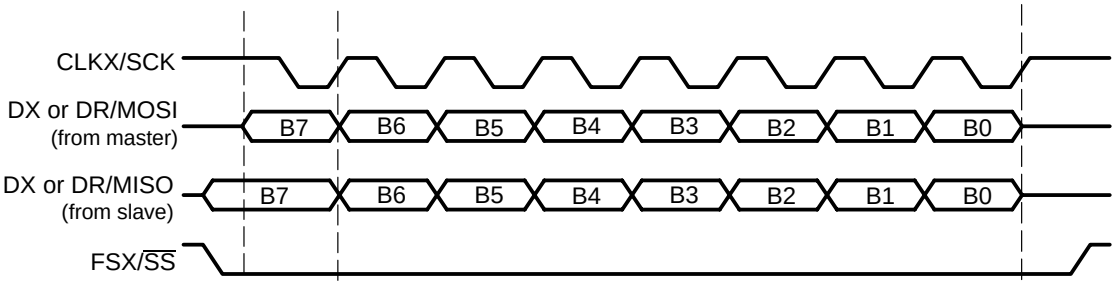
Figure 2–10. SPI Transfer With CLKSTP = 10b (no clock delay), CLKXP = 1, CLKRP = 0



Notes: 1) If the McBSP is the SPI master (CLKXM = 1), MOSI = DX. If the McBSP is the SPI slave (CLKXM = 0), MOSI = DR.

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

Figure 2–11. SPI Transfer With CLKSTP = 11b (clock delay), CLKXP = 1, CLKRP = 1



Notes: 1) If the McBSP is the SPI master (CLKXM = 1), MOSI=DX. If the McBSP is the SPI slave (CLKXM = 0), MOSI = DR.

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

2.3.4 Procedure for Configuring a McBSP for SPI Operation

To configure the McBSP for SPI master or slave operation:

1) Place the transmitter and receiver in reset.

   Clear the transmitter reset bit ($\overline{\text{XRST}}$ = 0) in SPCR2, to reset the transmitter. Clear the receiver reset bit ($\overline{\text{RRST}}$ = 0) in SPCR1, to reset the receiver.

2) Place the sample rate generator in reset.

   Clear the sample rate generator reset bit ($\overline{\text{GRST}}$ = 0) in SPCR2, to reset the sample rate generator.

3) Program registers that affect SPI operation.

   Program the appropriate McBSP registers to configure the McBSP for proper operation as an SPI master or an SPI slave. For a list of important bits settings, see one of the following topics:

   - *McBSP as the SPI Master* (page 2-20)
   - *McBSP as an SPI Slave* (page 2-22)

4) Enable the sample rate generator.

   To release the sample rate generator from reset, set the sample rate generator reset bit ($\overline{\text{GRST}}$ = 1) in SPCR2.

   Make sure that during the write to SPCR2, you only modify $\overline{\text{GRST}}$. Otherwise, you will modify the McBSP configuration you selected in the previous step.

5) Enable the transmitter and receiver.

After the sample rate generator is released from reset, wait two sample rate generator clock periods for the McBSP logic to stabilize.

If the CPU services the McBSP transmit and receive buffers, then you can immediately enable the transmitter ($\overline{\text{XRST}}$ = 1 in SPCR2) and enable the receiver ($\overline{\text{RRST}}$ = 1 in SPCR1).

The FIFO has to be configured first if it must write into the McBSP transmit buffer or read from the McBSP receive buffer. After the FIFO has been configured, make XRST = 1 and $\overline{\text{RRST}}$ = 1.

Note: In either case, make sure you only change $\overline{\text{XRST}}$ and $\overline{\text{RRST}}$ when you write to SPCR2 and SPCR1. Otherwise, you will modify the bit settings you selected earlier in this procedure.

After the transmitter and receiver are released from reset, wait two sample rate generator clock periods for the McBSP logic to stabilize.

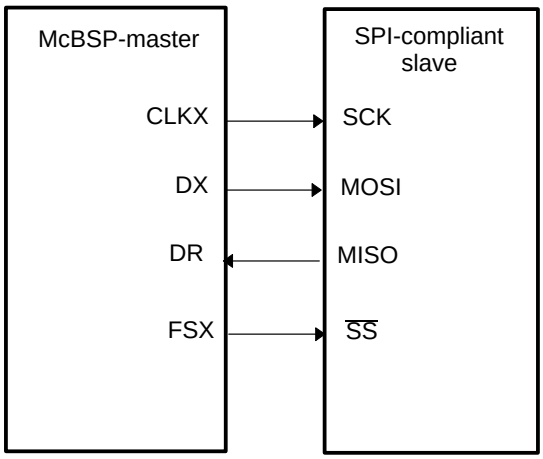
6) If necessary, enable the frame-sync logic of the sample rate generator.

After the required data acquisition setup is done (DXR[1/2] is loaded with data), set $\overline{\text{FRST}}$ = 1 if an internally generated frame-sync pulse is required (that is, if the McBSP is the SPI master).

2.3.5 McBSP as the SPI Master

An SPI interface with the McBSP used as the master is shown in Figure 2–12. 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.

Figure 2–12. SPI Interface With McBSP as Master



The register bit values required to configure the McBSP as a master are listed in the following table. After the table are more details about the configuration requirements.

Table 2–6. Bit Values Required to Configure the McBSP as an SPI Master

Required Bit Setting	Description
CLKSTP = 10b or 11b	The clock stop mode (without or with a clock delay) is selected.
CLKXP = 0 or 1	The polarity of CLKX as seen on the CLKX pin is positive (CLKXP = 0) or negative (CLKXP = 1).
CLKRP = 0 or 1	The polarity of CLKR as seen on the CLKR pin is positive (CLKRP = 0) or negative (CLKRP = 1).
CLKXM = 1	The CLKX pin is an output pin driven by the internal sample rate generator. Because CLKSTP is equal to 10b or 11b, CLKR is driven internally by CLKX.
SCLKME = 0 CLKSM = 1	The clock generated by the sample rate generator (CLKG) is derived from the CPU clock.
CLKGDV is a value from 0 to 255	CLKGDV defines the divide down value for CLKG.
FSXM = 1	The FSX pin is an output pin driven according to the FSGM bit.
FSGM = 0	The transmitter drives a frame-sync pulse on the FSX pin every time data is transferred from DXR1 to XSR1.
FSXP = 1	The FSX pin is active low.
XDATDLY = 01b RDATDLY = 01b	This setting provides the correct setup time on the FSX signal.

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 CPU 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 signal ($\overline{SS}$) on the FSX pin. If a slave-enable signal is required, the FSX pin must be configured as an output,

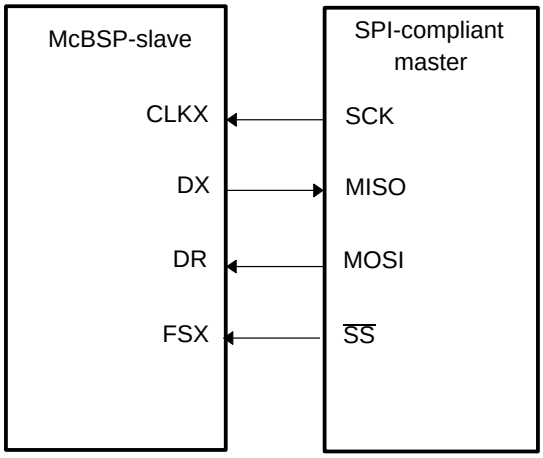
and the transmitter must be configured so that a frame-sync pulse is generated automatically each time a packet is transmitted (FSGM = 0). The polarity of the FSX pin is programmable high or low; however, in most cases the pin should be configured active-low.

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. To see the resulting waveform produced on the FSX pin, see the timing diagrams in section 2.3.3. 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.

2.3.6 McBSP as an SPI Slave

An SPI interface with the McBSP used as a slave is shown in Figure 2–13. When the McBSP is configured as a slave, DX is used as the MISO signal, and DR is used as the MOSI signal.

Figure 2–13. SPI With McBSP Configured as Slave



The register bit values required to configure the McBSP as a slave are listed in the following table. After the table are more details about the configuration requirements.

Table 2–7. Bit Values Required to Configure the McBSP as an SPI Slave

Required Bit Setting	Description
CLKSTP = 10b or 11b	The clock stop mode (without or with a clock delay) is selected.
CLKXP = 0 or 1	The polarity of CLKX as seen on the CLKX pin is positive (CLKXP = 0) or negative (CLKXP = 1).
CLKRP = 0 or 1	The polarity of CLKR as seen on the CLKR pin is positive (CLKRP = 0) or negative (CLKRP = 1).
CLKXM = 0	The CLKX pin is an input pin, so that it can be driven by the SPI master. Because CLKSTP = 10b or 11b, CLKR is driven internally by CLKX.
SCLKME = 0 CLKSM = 1	The clock generated by the sample rate generator (CLKG) is derived from the CPU 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 CPU clock by 2 before generating CLKG.
FSXM = 0	The FSX pin is an input pin, so that it can be driven by the SPI master.
FSXP = 1	The FSX pin is active low.
XDATDLY = 00b RDATDLY = 00b	These bits must be 0s 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 CPU 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 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 set to 0 for proper SPI slave operation. A value of 1 or 2 is undefined in the clock stop mode.

Chapter 3

Configure the Receiver and Transmitter

This chapter explains how to configure the receiver/transmitter and program the registers.

Topic **Page**

3.1 Receiver Configuration 3-2

3.2 Transmitter Configuration 3-26

3.1 Receiver Configuration

To configure the McBSP receiver, perform the following procedure:

1) Place the McBSP receiver in reset (see section 3.1.2).
2) Program the McBSP registers for the desired receiver operation (see section 3.1.1).
3) Take the receiver out of reset (see section 3.1.2).

3.1.1 Programming the McBSP Registers for the Desired Receiver Operation

The following is a list of important tasks to be performed when you are configuring the McBSP receiver. Each task corresponds to one or more McBSP register bit fields. Note that in the list, SRG is an abbreviation for sample rate generator.

- **Global behavior:**
  - Set the receiver pins to operate as McBSP pins
  - Enable/disable the digital loopback mode
  - Enable/disable the clock stop mode
  - Enable/disable the receive multichannel selection mode
  - Enable/disable the A-bis mode
- **Data behavior:**
  - Choose 1 or 2 phases for the receive frame
  - Set the receive word length(s)
  - Set the receive frame length
  - Enable/disable the receive frame-sync ignore function
  - Set the receive companding mode
  - Set the receive data delay
  - Set the receive sign-extension and justification mode
  - Set the receive interrupt mode
- **Frame-sync behavior:**
  - Set the receive frame-sync mode
  - Set the receive frame-sync polarity
  - Set the SRG frame-sync period and pulse width
- **Clock behavior:**
  - Set the receive clock mode
  - Set the receive clock polarity
  - Set the SRG clock divide-down value
  - Set the SRG clock synchronization mode
  - Set the SRG clock mode (choose an input clock)
  - Set the SRG input clock polarity

3.1.2 Resetting and Enabling the Receiver

The first step of the receiver configuration procedure is to reset the receiver, and the last step is to enable the receiver (to take it out of reset).

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

1) A DSP reset ($\overline{\text{RESET}}$ signal driven low) places the receiver, transmitter, and sample rate generator in reset. When the device reset is removed ($\overline{\text{RESET}}$ signal released), GRST = FRST = RRST = XRST = 0, keeping the entire serial port in the reset state.

2) The serial port transmitter and receiver can be reset directly using the RRST and XRST bits in the serial port control registers. The sample rate generator can be reset directly using the GRST bit in SPCR2.

To get this result ...	Use this bit ...
Enable the receiver	RRST (bit 0 of the SPCR1 register) set to 1
Disable the receiver (the reset state)	RRST set to 0
Reset the sample-rate generator	GRST (bit 6 of the SPCR2 register) set to 0
Enable the sample-rate generator	GRST set to 1
Reset the frame-sync logic	FRST (bit 7 of the SPCR2 register) = 0
Enable the frame-sync logic	FRST = 1

For details on the Serial Port Control Register 1 (SPCR1) and SPCR2 see Figure 6–3 (page 6-4) and Figure 6–4 (page 6-6). Table 3–1 shows the state of McBSP pins when the serial port is reset due to a DSP reset and a direct receiver/transmitter reset.

Table 3–1. Reset State of Each McBSP Pin

Pin	Possible State(s)	State Forced By DSP Reset	State Forced By Receiver/Transmitter Reset
			Receiver Reset ($\overline{RRST}$ = 0 and $\overline{GRST}$ = 1)
DR	I	Input	Input
CLKR	I/O/Z	Input	Known state if Input; CLKR running if output
FSR	I/O/Z	Input	Known state if Input; FSRP inactive state if output
			Transmitter Reset ($\overline{XRST}$ = 0 and $\overline{GRST}$ = 1)
DX	O/Z	High impedance	High impedance
CLKX	I/O/Z	Input	Known state if Input; CLKX running if output
FSX	I/O/Z	Input	Known state if Input; FSXP inactive state if output

Note: In Possible State(s) column, I = Input, O = Output, Z = High impedance

For more details about McBSP reset conditions and effects, see *Resetting and Initializing a McBSP* on page 4-3.

To get this result…	**Set this bit...**
Set the receiver pins to operate as McBSP pins in the reset state.	RIOEN (bit 12 of the PCR register) set to 0 0 is the only possible setting for this bit. 1 is a reserved function and must not be used.
Disable the digital-loopback mode	DLB (bit 15 of the SPCR1 register)
Enable the digital-loopback mode	DLB (bit 15 of the SPCR1 register)

In the digital loopback mode, the receive signals are connected internally through multiplexers to the corresponding transmit signals, as shown in Table 3–2. This mode allows testing of serial port code with a single DSP device; the McBSP receives the data it transmits.

Table 3–2. Receive Signals Connected to Transmit Signals in Digital Loopback Mode

This receive signal …	**Is fed internally by this transmit signal** …
DR (receive data)	DX (transmit data)
FSR (receive frame synchronization)	FSX (transmit frame synchronization)
CLKR (receive clock)	CLKX (transmit clock)

3.1.3 Clock Stop Mode

The CLKSTP bits in the serial port control registers determine whether the clock stop mode is on.

The clock stop mode supports the SPI master-slave protocol. If you will not be using the SPI protocol, you can clear the CLKSTP bits (12:11 in SPCR1) to disable the clock stop mode.

In the clock stop mode, the clock stops at the end of each data transfer. At the beginning of each data transfer, the clock starts immediately (CLKSTP = 10b) or after a half-cycle delay (CLKSTP = 11b). The CLKXP bit determines whether the starting edge of the clock on the CLKX pin is rising or falling. The CLKRP bit determines whether receive data is sampled on the rising or falling edge of the clock shown on the CLKR pin.

Table 3–3 summarizes the impact of CLKSTP, CLKXP, and CLKRP on serial port operation. Note that in the clock stop mode, the receive clock is tied internally to the transmit clock, and the receive frame-sync signal is tied internally to the transmit frame-sync signal.

Table 3–3. Effects of CLKSTP, CLKXP, and CLKRP on the Clock Scheme

Bit Settings	Clock Scheme
CLKSTP = 00b or 01b CLKXP = 0 or 1 CLKRP = 0 or 1	Clock stop mode disabled. Clock enabled for non-SPI mode.
CLKSTP = 10b CLKXP = 0 CLKRP = 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.
CLKSTP = 11b CLKXP = 0 CLKRP = 1	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.
CLKSTP = 10b CLKXP = 1 CLKRP = 0	High inactive state without delay: The McBSP transmits data on the falling edge of CLKX and receives data on the rising edge of CLKR.
CLKSTP = 11b CLKXP = 1 CLKRP = 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.

3.1.4 Receive Multichannel Selection and A-bis Modes

The RMCM bit determines whether the receive multichannel selection mode is on. For more details on this mode, see *Receive Multichannel Selection Mode* on page 2-7. For more details on the multichannel control registers, see Section 6.6 (page 6-17).

For more details on A-bis mode, see page 2-13. If A-bis mode is enabled, individual bits can be enabled or disabled during reception and transmission. For transmission, the bits are controlled by transmit channel enable registers A and B (XCERA and XCERB). For reception, the bits are controlled by receive channel enable registers A and B (RCERA and RCERB).

To get this result…	**Use this bit** …
Enable multichannel selection mode	RMCM bit in MCR1 to 1 (individual channels can be enabled or disabled with this setting)
Disable multichannel selection mode	Set RMCM to 0 so that all 128 channels are enabled
Enable A-bis mode	ABIS bit (number 6 in SPCR1) set to 1
Disable A-bis mode	ABIS set to 0

3.1.5 Choose 1 or 2 Phases for the Receive Frame

The RPHASE bit in the Receive Control Register 2 (RCR2) determines whether the receive data frame has one or two phases. See Section 6.3 for details of RCR1 and RCR2.

To get this result …	**Use this bit**…
Set receive frame to single phase	RPHASE bit in Receive FRCR2 set to 0
Set receive frame to dual-phase frame	RPHASE bit in RCR2 set to 1

3.1.5.1 Set the Receive Word Length(s)

The RWDLEN1 and RWDLEN2 bit fields determine how many bits are in each serial word in phase 1 and in phase 2, respectively, of the receive data frame.

To get this result …	Use this bit…	
Specify the length of every serial word in phase 1 of the receive frame	Set RWDLEN1 (bits 7:5 of RCR1) to one of the following:	
	000	8 bits
	001	12 bits
	010	16 bits
	011	20 bits
	100	24 bits
	101	32 bits
	11x	Reserved

Each frame can have one or two phases, depending on the value that you load into the RPHASE bit. If a single-phase frame is selected, RWDLEN1 selects the length for every serial word received in the frame. If a dual-phase frame is selected, RWDLEN1 determines the length of the serial words in phase 1 of the frame, and RWDLEN2 determines the word length in phase 2 of the frame with the same bit settings. For details on the receive control registers, see Section 6.3 (page 6-8).

3.1.5.2 Set the Receive Frame Length

The RFRLEN1 and RFRLEN2 bit fields determine how many serial words are in phase 1 and in phase 2, respectively, of the receive data frame.

To get this result …	Use this bit…	
Set receive frame length	Set RFRLEN1 (bits 14:8 of RCR1) to one of the following:	
	000 0000	1 word in phase 1
	000 0001	2 words in phase 1
	.	
	.	
	.	
	111 1111	128 words in phase 1

The receive frame length is the number of serial words in the receive frame. Each frame can have one or two phases, depending on value that you load into the RPHASE bit.

If a single-phase frame is selected (RPHASE = 0), the frame length is equal to the length of phase 1. If a dual-phase frame is selected (RPHASE = 1), the frame length is the length of phase 1 plus the length of phase 2:

The 7-bit RFRLEN fields allow up to 128 words per phase. See Table 3–4 for a summary of how to calculate the frame length. This length corresponds to the number of words or logical time slots or channels per frame-synchronization pulse.

Note: Program the RFRLEN fields with [*w minus 1*], where *w* represents the number of words per phase. For the example, if you want a phase length of 128 words in phase 1, load 127 into RFRLEN1.

Table 3–4. How to Calculate the Length of the Receive Frame

RPHASE	RFRLEN1	RFRLEN2	Frame Length
0	0 ≤ RFRLEN1 ≤ 127	Don't care	(RFRLEN1 + 1) words
1	0 ≤ RFRLEN1 ≤ 127	0 ≤ RFRLEN2 ≤ 127	(RFRLEN1 + 1) + (RFRLEN2 + 1) words

3.1.5.3 Enable/Disable the Receive Frame-Sync Ignore Function

The RFIG bit controls the receive frame-sync ignore function.

If there is an unexpected receive frame-sync pulse …	**Use this bit**…
Restart the frame transfer	Set RFIG (bit 2 in RCR2) to 0
Ignore the unexpected pulses	Set RFIG to 1

If a frame-synchronization (frame-sync) pulse starts the transfer of a new frame before the current frame is fully received, this pulse is treated as an unexpected frame-sync pulse.

When RFIG = 1, reception continues, ignoring the unexpected frame-sync pulses.

When RFIG = 0, an unexpected FSR pulse causes the McBSP to discard the contents of RSR[1,2] in favor of the new incoming data. Therefore, if RFIG = 0 and an unexpected frame-sync pulse occurs, the serial port:

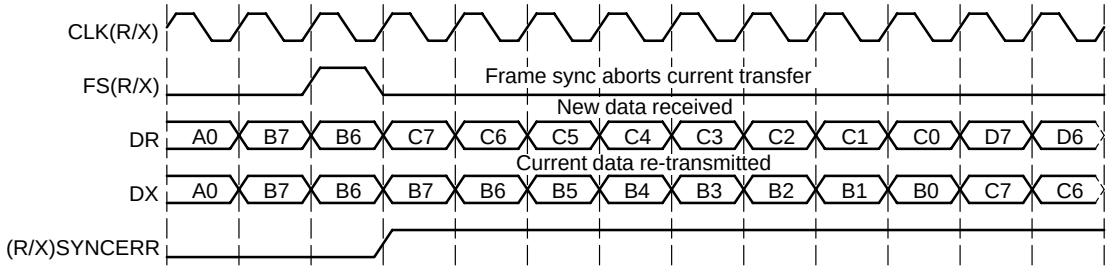
1) Aborts the current data transfer
2) Sets RSYNCERR in SPCR1 to 1
3) Begins the transfer of a new data word

For more details about the frame-sync error condition, see *Unexpected Receive Frame-Sync Pulse* on page 1-40.

3.1.5.4 Examples Showing the Effects of RFIG

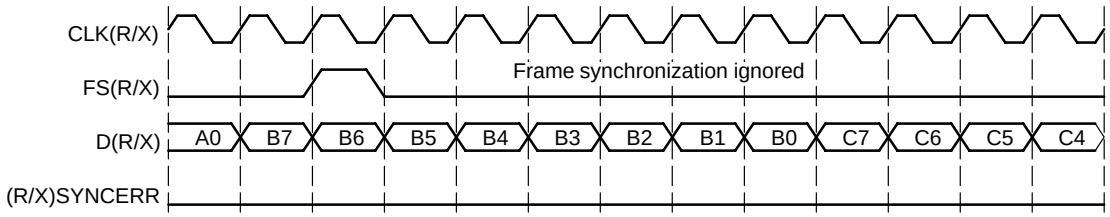
Figure 3–1 shows an example in which word B is interrupted by an unexpected frame-sync pulse when (R/X)FIG = 0. In the case of reception, the reception of B is aborted (B is lost), and a new data word (C in this example) is received after the appropriate data delay. This condition is a receive synchronization error, and thus sets the RSYNCERR bit.

Figure 3–1. Unexpected Frame-Sync Pulse With (R/X)FIG = 0



In contrast with Figure 3–1, Figure 3–2 shows McBSP operation when unexpected frame-sync signals are ignored (when (R/X)FIG = 1). Here, the transfer of word B is not affected by an unexpected pulse.

Figure 3–2. Unexpected Frame-Sync Pulse With (R/X)FIG = 1



3.1.6 Set the Receive Companding Mode

Companding (COMpressing and exPANDing) hardware allows compression and expansion of data in either μ-law or A-law format. The companding standard employed in the United States and Japan is μ-law. The European companding standard is referred to as A-law. The specifications for μ-law and A-law log PCM are part of the CCITT G.711 recommendation.

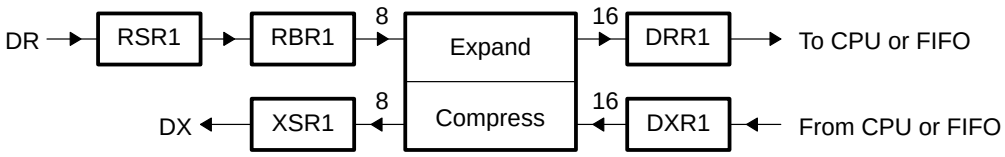
A-law and μ-law allow 13 bits and 14 bits of dynamic range, respectively. Any values outside this range are set to the most positive or most negative value. Thus, for companding to work best, the data transferred to and from the McBSP via the CPU or FIFO must be at least 16 bits wide.

The μ-law and A-law formats both encode data into 8-bit code words. Companded data is always 8 bits wide; the appropriate word length bits

(RWDLEN1, RWDLEN2, XWDLEN1, XWDLEN2) must therefore be set to 0, indicating an 8-bit wide serial data stream. If companding is enabled and either of the frame phases does not have an 8-bit word length, companding continues as if the word length is 8 bits.

Figure 3–3 illustrates the companding processes. When companding is chosen for the transmitter, compression occurs during the process of copying data from DXR1 to XSR1. The transmit data is encoded according to the specified companding law (A-law or μ-law). When companding is chosen for the receiver, expansion occurs during the process of copying data from RBR1 to DRR1. The receive data is decoded to 2s-complement format.

Figure 3–3. Companding Processes for Reception and for Transmission



For reception, the 8-bit compressed data in RBR1 is expanded to left-justified 16-bit data in DRR1. Note that the RJUST bit of SPCR1 is ignored when companding is used.

If the McBSP is otherwise unused (the serial port transmit and receive sections are reset), the companding hardware can compand internal data. See *Capability to Compand Internal Data* on page 1-13.

Normally, the McBSP transmit or receives all data with the most significant bit (MSB) first. However, certain 8-bit data protocols (that do not use companded data) require the least significant bit (LSB) to be transferred first. If you set RCOMPAND = 01b in RCR2, the bit ordering of 8-bit words is reversed during reception. Similar to companding, this feature is enabled only if the appropriate word length bits are set to 0, indicating that 8-bit words are to be transferred serially. If either phase of the frame does not have an 8-bit word length, the McBSP assumes the word length is eight bits, and LSB-first ordering is done.

The RCOMPAND bits determine whether companding or another data transfer option is chosen for McBSP reception. Modes other than 00b are enabled only when the appropriate RWDLEN is 000b, indicating 8-bit data.

To get this result …	**Use this bit**…
Receive companding mode	Set RCOMPAND (bits 4:3 in RCR2) to one of the following
	00 No companding, any size data, MSB received first

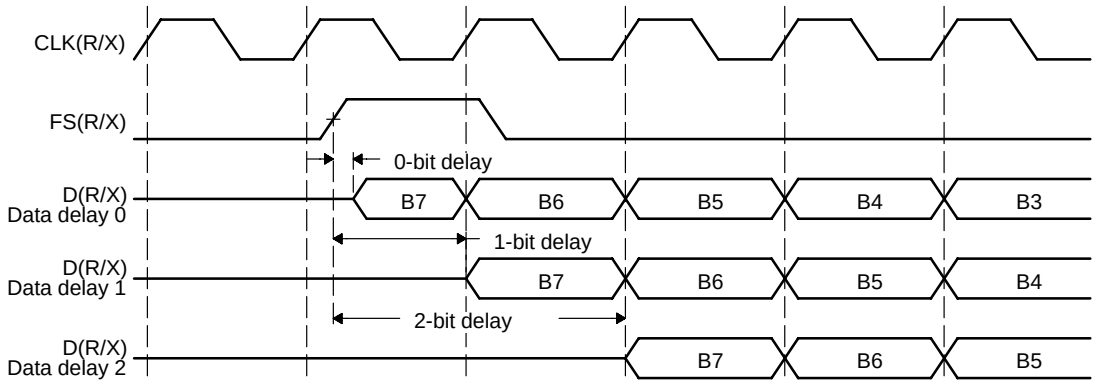
To get this result …	Use this bit…	
	01	No companding, 8-bit data, LSB received first (for details, scroll down to Option to Receive LSB First)
	10	μ-law companding, 8-bit data, MSB received first
	11	A-law companding, 8-bit data, MSB received first

3.1.7 Set the Receive Data Delay

The start of a frame is defined by the first clock cycle in which frame synchronization is found to be 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 specifies the data delay for reception. The range of programmable data delay is zero to two bit-clocks (RDATDLY = 00b–10b), as shown in Figure 3–4. In this figure, the data transferred is an 8-bit value with bits labeled B7, B6, B5, and so on. Typically a 1-bit delay is selected, because data often follows a 1-cycle active frame-sync pulse.

Figure 3–4. Range of Programmable Data Delay



3.1.7.1 0-Bit Data Delay

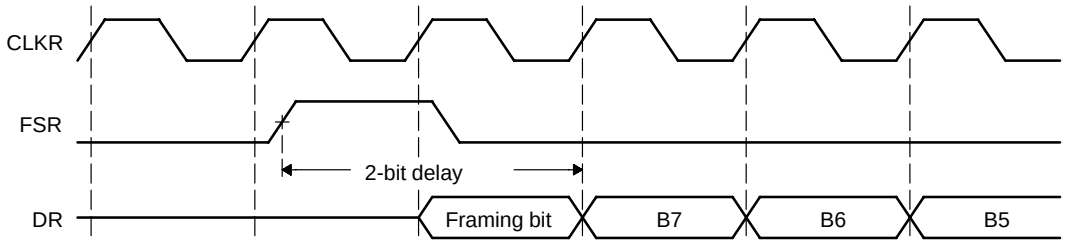
Normally, a frame-sync pulse is detected or sampled with respect to an edge of internal serial clock CLK(R/X). Thus, on the following cycle or later (depending on the data delay value), data may be received or transmitted. However, in the case of 0-bit data delay, the data must be ready for reception and/or transmission on the same serial clock cycle.

For reception this problem is solved because receive data is sampled on the first falling edge of CLKR where an active-high internal FSR is detected. However, data transmission must begin on the rising edge of the internal CLKX clock that generated the frame synchronization. Therefore, the first data bit is assumed to be present in XSR1, and thus on DX. The transmitter then asynchronously detects the frame-sync signal (FSX) going active high and immediately starts driving the first bit to be transmitted on the DX pin.

3.1.7.2 2-Bit Data Delay

A data delay of two bit periods allows the serial port to interface to different types of T1 framing devices where the data stream is preceded by a framing bit. During reception of such a stream with data delay of two bits (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 3–5. In this figure, the data transferred is an 8-bit value with bits labeled B7, B6, B5, and so on.

Figure 3–5. 2-Bit Data Delay Used to Skip a Framing Bit



The RDATDLY bits determine the length of the data delay for the receive frame.

To get this result …	**Use this bit**…	
Set the receive data delay	Set RDATDLY (bits 1:0 in RCR2 to:	
	00	0-bit data delay
	01	1-bit data delay
	10	2-bit data delay
	11	Reserved

3.1.8 Set the Receive Sign-Extension and Justification Mode

RJUST in SPCR1 selects whether data in RBR[1,2] is right- or left-justified (with respect to the MSB) in DRR[1,2] and how unused bits in DRR[1,2] are filled—with zeros or with sign bits.

Table 3–5 and Table 3–6 show the effects of various RJUST values. The first table shows the effect on an example 12-bit receive-data value 0xABC. The second table shows the effect on an example 20-bit receive-data value 0xABCDE.

Table 3–5. Example: Use of RJUST Field With 12-Bit Data Value 0xABC

RJUST	Justification	Extension	Value in DRR2	Value in DRR1
00b	Right	Zero fill MSBs	0x0000	0x0ABC
01b	Right	Sign extend data into MSBs	0xFFFF	0xFABC
10b	Left	Zero fill LSBs	0x0000	0xABC0
11b	Reserved	Reserved	Reserved	Reserved

Table 3–6. Example: Use of RJUST Field With 20-Bit Data Value 0xABCDE

RJUST	Justification	Extension	Value in DRR2	Value in DRR1
00b	Right	Zero fill MSBs	0x000A	0xBCDE
01b	Right	Sign extend data into MSBs	0xFFFA	0xBCDE
10b	Left	Zero fill LSBs	0xABCD	0xE000
11b	Reserved	Reserved	Reserved	Reserved

The RJUST bits determine whether data received by the McBSP is sign extended and how it is justified.

To get this result …	**Use this bit**…	
Set receive sign-extension and justification mode	Set RJUST (bits 14:13 in SPCR1 to:	
	00	Right justify data and zero fill MSBs in DRR[1,2]
	01	Right justify data and sign extend it into the MSBs in DRR[1,2]
	10	Left justify data and zero fill LSBs in DRR[1,2]
	11	Reserved

3.1.9 Set the Receive Interrupt Mode

The receive interrupt (RINT) signals the CPU of changes to the serial port status. Four options exist for configuring this interrupt. The options are set by the receive interrupt mode bits, RINTM, in SPCR1.

- RINTM = 00b. Interrupt on every serial word by tracking the RRDY bit in SPCR1. Note that regardless of the value of RINTM, RRDY can be read to detect the RRDY = 1 condition.

- RINTM = 01b. In the multichannel selection mode, interrupt after every 16-channel block boundary has been crossed within a frame and at the end of the frame. In any other serial transfer case, this setting is not applicable and, therefore, no interrupts are generated.

- RINTM = 10b. Interrupt on detection of receive frame-sync pulses. This generates an interrupt even when the receiver is in its reset state. This is done by synchronizing the incoming frame-sync pulse to the CPU clock and sending it to the CPU via RINT.

- RINTM = 11b. Interrupt on frame-synchronization error. Note that regardless of the value of RINTM, RSYNCERR can be read to detect this condition. For information on using RSYNCERR, see *Unexpected Receive Frame-Sync Pulse* on page 1-40. See section 5.4 for RINTM interrupt selection using FIFO logic.

The RINTM bits determine which event generates a receive interrupt request to the CPU.

To get this result …	**Use this bit**…	
Receive interrupt mode	Set RINTM (bits 5:4 in SPCR1 to:	
	00	RINT generated when RRDY changes from 0 to 1
	01	RINT generated by an end-of-block or end-of-frame condition in the receive multichannel selection mode
	10	RINT generated by a new receive frame-sync pulse
	11	RINT generated when RSYNCERR is set

3.1.10 Set the Receive Frame-Sync Mode

Table 3–7 shows how you may select various sources to provide the receive frame-synchronization signal and the effect on the FSR pin. The polarity of the signal on the FSR pin is determined by the FSRP bit.

Note that in the digital loop back mode (DLB = 1), the transmit frame-sync signal is used as the receive frame-sync signal.

Also, in the clock stop mode, the internal receive clock signal (CLKR) and the internal receive frame-synchronization signal (FSR) are internally connected to their transmit counterparts, CLKX and FSX.

Table 3–7. Select Sources to Provide the Receive Frame-Synchronization Signal and the Effect on the FSR Pin

DLB	FSRM	GSYNC	Source of Receive Frame Synchronization	FSR Pin Status
0	0	0 or 1	An external frame-sync signal enters the McBSP through the FSR pin. The signal is then inverted as determined by FSRP before being used as internal FSR.	Input
0	1	0	Internal FSR is driven by the sample rate generator frame-sync signal (FSG).	Output. FSG is inverted as determined by FSRP before being driven out on the FSR pin.
0	1	1	Internal FSR is driven by the sample rate generator frame-sync signal (FSG).	Input. The external frame-sync input on the FSR pin is used to synchronize CLKG and generate FSG pulses.
1	0	0	Internal FSX drives internal FSR.	High impedance
1	0 or 1	1	Internal FSX drives internal FSR.	Input. If the sample rate generator is running, external FSR is used to synchronize CLKG and generate FSG pulses.
1	1	0	Internal FSX drives internal FSR.	Output. Receive (same as transmit) frame synchronization is inverted as determined by FSRP before being driven out on the FSR pin.

3.1.11 Set the Receive Frame-Sync Polarity

The FSRP bit determines whether frame-synchronization (frame-sync) pulses are active high or active low on the FSR pin.

Receive frame-sync pulses can be either generated internally by the sample rate generator (see section 1.4.2 on page 1-30) or driven by an external source. The source of frame sync is selected by programming the mode bit, FSRM, in PCR. FSR is also affected by the GSYNC bit in SRGR2. For information about the effects of FSRM and GSYNC, see *Set the Receive Frame-Sync Mode* on page 3-15. Similarly, receive clocks can be selected to be inputs or outputs by programming the mode bit, CLKRM, in the PCR.

When FSR and FSX are inputs (FSXM = FSRM= 0, external frame-sync pulses), the McBSP detects them on the internal falling edge of clock, internal CLKR, and internal CLKX, respectively. The receive data arriving at the DR pin is also sampled on the falling edge of internal CLKR. Note that these internal clock signals are either derived from external source via CLK(R/X) pins or driven by the sample rate generator clock (CLKG) internal to the McBSP.

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

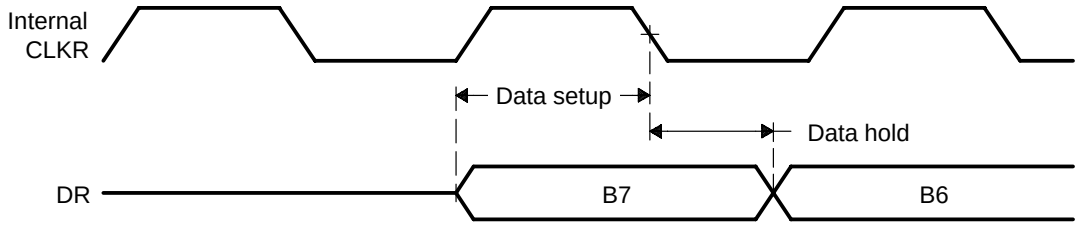
FSRP, FSXP, CLKRP, and CLKXP in the pin control register (PCR) configure the polarities of the FSR, FSX, CLKR, and CLKX signals, respectively. All frame-sync signals (internal FSR, internal FSX) that are internal to the serial port are active high. If the serial port is configured for external frame synchronization (FSR/FSX are inputs to McBSP), and FSRP = FSXP = 1, the external active-low frame-sync signals are inverted before being sent to the receiver (internal FSR) and transmitter (internal FSX). Similarly, if internal synchronization (FSR/FSX are output pins and GSYNC = 0) is selected, the internal active-high frame-sync signals are inverted, if the polarity bit FS(R/X)P = 1, before being sent to the FS(R/X) pin. The figure in the topic Clock and Frame Generation 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. Note that data is always transmitted on the rising edge of internal CLKX. 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 selected (CLKXM = 1 and CLKX is an output pin), the internal (rising-edge triggered) clock, internal CLKX, is inverted before being sent out on the CLKX pin.

Similarly, the receiver can reliably sample data that is clocked with a rising edge clock (by the transmitter). The receive clock polarity bit, CLKRP, sets the edge used to sample received data. Note that the receive data is always sampled on the falling edge of internal CLKR. 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 triggered 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 triggered clock before being sent out on the CLKR pin.

Note that CLKRP = CLKXP in a system where the same clock (internal or external) is used to clock the receiver and transmitter. The receiver uses the opposite edge as the transmitter to ensure valid setup and hold of data around this edge. Figure 3–6 shows how data clocked by an external serial device using a rising edge, may be sampled by the McBSP receiver on the falling edge of the same clock.

Figure 3–6. Data Clocked Externally Using a Rising Edge and Sampled by the McBSP Receiver on a Falling Edge



The FSRM (in PCR), GSYNC (in SRGR2), DLB and CLKSTP (both in SPCR1) determine the source for receive frame synchronization and the function of the FSR pin.

To get this result …	**Use this bit**…
Frame synchronization Supply receive frame synchronization by an external source via the FSR pin.	Set FSRM (bit 10 in PCR) to 0.
Receive frame synchronization is supplied by the sample rate generator. FSR is an output pin reflecting internal FSR, except when GSYNC = 1 in SRGR2.	Set FSRM (bit 10 in PCR) to 1
Set sample rate generator clock synchronization mode Use no clock synchronization. CLKG oscillates without adjustment, and FSG pulses every (FPER + 1) CLKG cycles.	Set GSYNC (bit 15 in SRGR2) to 0.

To get this result …	Use this bit…
Use clock synchronization. When a pulse is detected on the FSR pin: ❑ CLKG is adjusted as necessary so that it is synchronized with the input clock on the CLKS, CLKR, or CLKX pin. ❑ FSG pulses. FSG **only** pulses in response to a pulse on the FSR pin. The frame-sync period defined in FPER is ignored.	Set GSYNC to 1. For more details, see *Synchronizing Sample Rate Generator Outputs to an External Clock* on page 1-31.
Set digital loopback (DLB) mode. Disable DLB mode	Set DLB (bit 15 in SPCR1) to 0.
Enable digital loopback mode. The receive signals, including the receive frame-sync signal, are connected internally through multiplexers to the corresponding transmit signals.	Set DLB to 1.
Set clock stop mode Disable clock stop mode – normal clocking for non-SPI mode.	Set CLKSTP (bits 12:11 in SPCR1) to 0Xb
Enable clock stop mode without clock delay. The internal receive clock signal (CLKR) and the internal receive frame-synchronization signal (FSR) are internally connected to their transmit counterparts, CLKX and FSX.	Set CLKSTP (bits 12:11 in SPCR1) to 10b.
Clock stop mode enabled, with clock delay. The internal receive clock signal (CLKR) and the internal receive frame-synchronization signal (FSR) are internally connected to their transmit counterparts, CLKX and FSX.	Set CLKSTP (bits 12:11 in SPCR1) to 11b.

3.1.11.1 Set the SRG Frame-Sync Period and Pulse Width

The sample rate generator can produce a clock signal, CLKG, and a frame-sync signal, FSG. If the sample rate generator is supplying receive or transmit frame synchronization, you must program the bit fields FPER and FWID.

On FSG, the period from the start of a frame-sync pulse to the start of the next pulse is (FPER + 1) CLKG cycles. The 12 bits of FPER allow a frame-sync

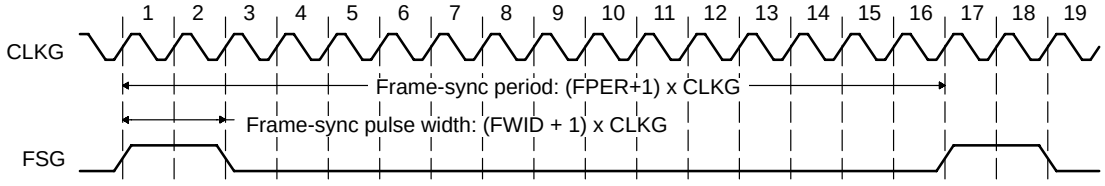
period of 1 to 4096 CLKG cycles, which allows up to 4096 data bits per frame. When GSYNC = 1, FPER is a don't care value.

Each pulse on FSG has a width of (FWID + 1) CLKG cycles. The eight bits of FWID allow a pulse width of 1 to 256 CLKG cycles. It is recommended that FWID be programmed to a value less than the programmed word length.

The values in FPER and FWID are loaded into separate down-counters. The 12-bit FPER counter counts down the generated clock cycles from the programmed value (4095 maximum) to 0. The 8-bit FWID counter counts down from the programmed value (255 maximum) to 0.

Figure 3–7 shows a frame-sync period of 16 CLKG periods (FPER = 15 or 00001111b) and a frame-sync pulse with an active width of 2 CLKG periods (FWID = 1).

Figure 3–7. Frame of Period 16 CLKG Periods and Active Width of 2 CLKG Periods



When the sample rate generator comes out of reset, FSG is in its inactive state. Then, when $\overline{\text{FRST}}$ = 1 and FSGM = 1, a frame-sync pulse is generated. The frame width value (FWID + 1) is counted down on every CLKG cycle until it reaches 0, at which time FSG goes low. At the same time, the frame period value (FPER + 1) is also counting down. When this value reaches 0, FSG goes high, indicating a new frame.

To get this result …	**Use this bit**…
Set the SRG frame-sync period and pulse width	FPER (bits 11:0 in SRGR2) set to range FPER + 1 (1 to 4096 CLKG cycles) For the frame-sync signal FSG, (FPER + 1) determines the period from the start of a frame-sync pulse to the start of the next frame-sync pulse.
Set the sample rate generator frame-sync pulse width	FWID (bits 15:8) in SRGR1 This field plus 1 determines the width of each frame-sync pulse on FSG.

3.1.11.2 Set the Receive Clock Mode

To get this result …	Use this bit…
Enable receive clock mode with digital loop-back (DLB) mode not set (DLB – 1)	Set CLKRM (bit 7 in PCR) to 0.
Enable receive clock mode with DLB mode not set (DLB – 1)	Set CLKRM (bit 7 in PCR) to 0.
Enable DLB mode	Set DLB (bit 15 in SPCR1) to 1
Disable DLB	Set DLB (bit 15 in SPCR1) to 0
Enable clock stop mode with clock delay	Set CLKSTP (bits 12:11 in SPCR1) to 11b.
Enable clock stop mode without clock delay	Set CLKSTP (bits 12:11 in SPCR1) to 10b.
Disable clock stop mode (normal clocking for non-SPI mode)	Set CLKSTP (bits 12:11 in SPCR1) to 0Xb.

3.1.11.3 Selecting a Source for the Receive Clock and a Data Direction for the CLKR Pin

Table 3–8 shows how you may select various sources to provide the receive clock signal and the effect on the CLKR pin. The polarity of the signal on the CLKR pin is determined by the CLKRP bit.

Note that in the digital loop back mode (DLB = 1), the transmit clock signal is used as the receive clock signal.

Also, in the clock stop mode, the internal receive clock signal (CLKR) and the internal receive frame-synchronization signal (FSR) are internally connected to their transmit counterparts, CLKX and FSX.

Table 3–8. Select Sources to Provide the Receive Clock Signal and the Effect on the CLKR Pin

DLB in SPCR1	CLKRM in PCR	Source of Receive Clock	CLKR Pin Status
0	0	The CLKR pin is an input driven by an external clock. The external clock signal is inverted as determined by CLKRP before being used.	Input
0	1	The sample rate generator clock (CLKG) drives internal CLKR.	Output. CLKG, inverted as determined by CLKRP, is driven out on the CLKR pin.

Table 3–8. Select Sources to Provide the Receive Clock Signal and the Effect on the CLKR Pin (Continued)

DLB in SPCR1	CLKRM in PCR	Source of Receive Clock	CLKR Pin Status
1	0	Internal CLKX drives internal CLKR.	High impedance
1	1	Internal CLKX drives internal CLKR.	Output. Internal CLKR (same as internal CLKX) is inverted as determined by CLKRP before being driven out on the CLKR pin.

Receive frame-sync pulses can be either generated internally by the sample rate generator (see section 1.4.2 on page 1-30) or driven by an external source. The source of frame sync is selected by programming the mode bit, FSRM, in PCR. FSR is also affected by the GSYNC bit in SRGR2. For information about the effects of FSRM and GSYNC, see *Set the Receive Frame-Sync Mode* on page 3-15. Similarly, receive clocks can be selected to be inputs or outputs by programming the mode bit, CLKRM, in the PCR.

When FSR and FSX are inputs (FSXM = FSRM= 0, external frame-sync pulses), the McBSP detects them on the internal falling edge of clock, internal CLKR, and internal CLKX, respectively. The receive data arriving at the DR pin is also sampled on the falling edge of internal CLKR. Note that these internal clock signals are either derived from external source via CLK(R/X) pins or driven by the sample rate generator clock (CLKG) internal to the McBSP.

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

FSRP, FSXP, CLKRP, and CLKXP in the pin control register (PCR) configure the polarities of the FSR, FSX, CLKR, and CLKX signals, respectively. All frame-sync signals (internal FSR, internal FSX) that are internal to the serial port are active high. If the serial port is configured for external frame synchronization (FSR/FSX are inputs to McBSP), and FSRP = FSXP = 1, the external active-low frame-sync signals are inverted before being sent to the receiver (internal FSR) and transmitter (internal FSX). Similarly, if internal synchronization (FSR/FSX are output pins and GSYNC = 0) is selected, the internal active-high frame-sync signals are inverted, if the polarity bit FS(R/X)P = 1, before being sent to the FS(R/X) pin. The figure in the topic Clock and Frame Generation shows this inversion using XOR gates.

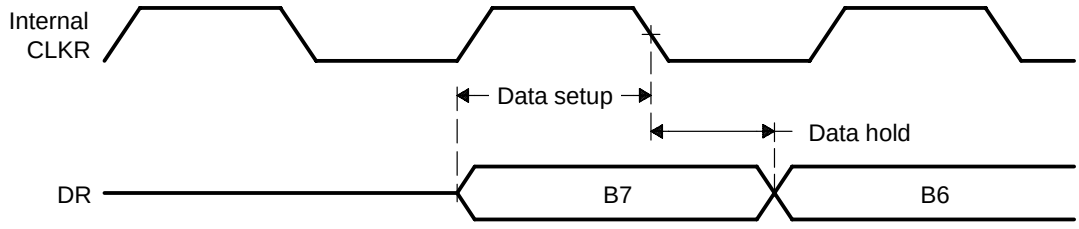
On the transmit side, the transmit clock polarity bit, CLKXP, sets the edge used to shift and clock out transmit data. Note that data is always transmitted on the

rising edge of internal CLKX. 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 selected (CLKXM = 1 and CLKX is an output pin), the internal (rising-edge triggered) clock, internal CLKX, is inverted before being sent out on the CLKX pin.

Similarly, the receiver can reliably sample data that is clocked with a rising edge clock (by the transmitter). The receive clock polarity bit, CLKRP, sets the edge used to sample received data. Note that the receive data is always sampled on the falling edge of internal CLKR. 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 triggered 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 triggered clock before being sent out on the CLKR pin.

Note that CLKRP = CLKXP in a system where the same clock (internal or external) is used to clock the receiver and transmitter. The receiver uses the opposite edge as the transmitter to ensure valid setup and hold of data around this edge. Figure 3–8 shows how data clocked by an external serial device using a rising edge, may be sampled by the McBSP receiver on the falling edge of the same clock.

Figure 3–8. Data Clocked Externally Using a Rising Edge and Sampled by the McBSP Receiver on a Falling Edge



3.1.11.4 Set the Receive Clock Polarity

To get this result …	**Use this bit**…
Sample receive data on falling edge of CLKR	Set CLRP (bit 0 in PCR) to 0
Sample receive data on rising edge of CLKR	Set CLRP (bit 0 in PCR) to 1

3.1.11.5 Set the SRG Clock Divide-Down Value

The first divider stage generates the serial data bit clock from the input clock. This divider stage utilizes a counter, preloaded by CLKGDV, that contains the divide ratio value.

The output of the first divider stage is the data bit clock, which is output as CLKG and which serves as the input for the second and third stages of the divider.

CLKG has a frequency equal to 1/(CLKGDV + 1) of sample rate generator input clock. Thus, the sample generator input clock frequency is divided by a value between 1 and 256. When CLKGDV is odd or equal to 0, the CLKG duty cycle is 50%. When CLKGDV is an even value, 2p, representing an odd divide-down, the high-state duration is p+1 cycles and the low-state duration is p cycles.

To get this result …	**Use this bit**…
Set the divide down value of the sample rate generator clock	Set CLKGDV (bits 7–0 of SRGR1) to generate the required SRG clock frequency. The default value is 1 (divide input clock by 2).
Set the SRG clock synchronization mode The sample rate generator clock (CLKG) is free running. CLKG oscillates without adjustment, and FSG pulses every (FPER + 1) CLKG cycles.	Set GYSYNC (bit 15 of SRGR2) to 0
Clock synchronization is performed. When a pulse is detected on the FSR pin: ❑ CLKG is adjusted as necessary so that it is synchronized with the input clock on the CLKR or CLKX pin. ❑ FSG pulses. FSG **only** pulses in response to a pulse on the FSR pin. The frame-sync period defined in FPER is ignored.	Set GYSYNC (bit 15 of SRGR2) to 1

GSYNC is used only when the input clock source for the sample rate generator is external—on the CLKR or CLKX pin.

For more details on using the clock synchronization feature, see *Synchronizing Sample Rate Generator Outputs to an External Clock* on page 1-31.

3.1.11.6 Set the SRG Clock Mode (Choose an Input Clock)

The sample rate generator can produce a clock signal (CLKG) for use by the receiver, the transmitter, or both, but CLKG is derived from an input clock. The preceding table shows the four possible sources of the input clock. For more details on generating CLKG, see *Clock Generation in the Sample Rate Generator* on page 1-27.

To get this result …	**Use this bit**…
Choose sample rate generator clock derived from CLKS pin	Set SCLKME (bit 7 of PCR) and CLKSM (bit 13 of SRGR2) both to 0.
Choose sample rate generator clock derived from CPU clock. (This is the condition forced by a DSP reset.)	Set SCLKME = 0 and CLKSM = 1.
Choose sample rate generator clock derived from CLKR pin	Set SCLKME = 1 and CLKSM = 0.
Choose sample rate generator clock derived from CLKX pin	Set SCLKME = 1 and CLKSM = 1.

3.1.11.7 Set the SRG Input Clock Polarity

To get this result …	**Use this bit**…
CLKXP determines the input clock polarity when the CLKX pin supplies the input clock (SCLKME = 1 and CLKSM = 1).	Set CLKXP (bit 1 in the PCR register) to 0.
Rising edge on CLKX pin generates transitions on CLKG and FSG.	
Falling edge on CLKX pin generates transitions on CLKG and FSG.	Set CLKXP to 1.

To get this result …	Use this bit…
CLKR Pin Polarity CLKRP determines the input clock polarity when the CLKR pin supplies the input clock (SCLKME = 1 and CLKSM = 0). Falling edge on CLKR pin generates transitions on CLKG and FSG.	Set CLKRP to 0.
Rising edge on CLKR pin generates transitions on CLKG and FSG.	Set CLKRP to 1.

3.1.11.8 Using CLKXP/CLKRP to Choose an Input Clock Polarity

The sample rate generator can produce a clock signal (CLKG) and a frame-sync signal (FSG) for use by the receiver, the transmitter, or both. To produce CLKG and FSG, the sample rate generator must be driven by an input clock signal derived from the CPU clock or from an external clock on the CLKS, CLKX, or CLKR pin. If you use a pin, choose a polarity for that pin by using the appropriate polarity bit (CLKXP for the CLKX pin and CLKRP for the CLKR pin). The polarity determines whether the rising or falling edge of the input clock generates transitions on CLKG and FSG.

3.2 Transmitter Configuration

To configure the McBSP transmitter, perform the following procedure:

1) Place the McBSP/transmitter in reset (see section 3.2.2).
2) Program the McBSP registers for the desired transmitter operation (see 3.2.1).
3) Take the transmitter out of reset (see section 3.2.2).

3.2.1 Programming the McBSP Registers for the Desired Transmitter Operation

The following is a list of important tasks to be performed when you are configuring the McBSP transmitter. Each task corresponds to one or more McBSP register bit fields. Note that in the list, SRG is an abbreviation for sample rate generator.

- **Global behavior:**
  - Set the transmitter pins to operate as McBSP pins
  - Enable/disable the digital loopback mode
  - Enable/disable the clock stop mode
  - Enable/disable transmit multichannel selection
  - Enable/disable the A-bis mode
- **Data behavior:**
  - Choose 1 or 2 phases for the transmit frame
  - Set the transmit word length(s)
  - Set the transmit frame length
  - Enable/disable the transmit frame-sync ignore function
  - Set the transmit companding mode
  - Set the transmit data delay
  - Set the transmit DXENA mode
  - Set the transmit interrupt mode
- **Frame-sync behavior:**
  - Set the transmit frame-sync mode
  - Set the transmit frame-sync polarity
  - Set the SRG frame-sync period and pulse width
- **Clock behavior:**
  - Set the transmit clock mode
  - Set the transmit clock polarity
  - Set the SRG clock divide-down value
  - Set the SRG clock synchronization mode
  - Set the SRG clock mode (choose an input clock)
  - Set the SRG input clock polarity

3.2.2 Resetting and Enabling the Transmitter

The first step of the transmitter configuration procedure is to reset the transmitter, and the last step is to enable the transmitter (to take it out of reset).

To get this result ...	Use this bit ...
Enable the serial port transmitter	XRST (bit 0 of the SPCR2 register) set to 1
Disable the serial port transmitter (the reset state)	XRST set to 0
Reset the sample-rate generator	GRST (bit 6 of the SPCR2 register) set to 0
Enable the sample-rate generator	GRST set to 1
Reset the frame-sync logic	FRST (bit 7 of the SPCR2 register) = 0
Enable the frame-sync logic	FRST = 1

3.2.2.1 Reset Considerations

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

1) A DSP reset ($\overline{\text{RESET}}$ signal driven low) places the receiver, transmitter, and sample rate generator in reset. When the device reset is removed ($\overline{\text{RESET}}$ signal released), GRST = FRST= RRST = XRST = 0, keeping the entire serial port in the reset state.

2) The serial port transmitter and receiver can be reset directly using the RRST and XRST bits in the serial port control registers. The sample rate generator can be reset directly using the GRST bit in SPCR2.

Table 3–9 shows the state of McBSP pins when the serial port is reset due to a DSP reset and a direct receiver/transmitter reset.

Table 3–9. Reset State of Each McBSP Pin

Pin	Possible State(s)	State Forced By DSP Reset	State Forced By Receiver/Transmitter Reset
			Receiver Reset (RRST = 0 and GRST = 1)
DR	I	Input	Input
CLKR	I/O/Z	Input	Known state if Input; CLKR running if output
FSR	I/O/Z	Input	Known state if Input; FSRP inactive state if output
			Transmitter Reset (XRST = 0 and GRST = 1)
DX	O/Z	High impedance	High impedance
CLKX	I/O/Z	Input	Known state if Input; CLKX running if output
FSX	I/O/Z	Input	Known state if Input; FSXP inactive state if output

For more details about McBSP reset conditions and effects, see *Resetting and Initializing a McBSP* on page 4-3.

3.2.3 Set the Transmitter Pins to Operate as McBSP Pins

To get this result…	Set this bit...
Set the receiver pins to operate as McBSP pins in the reset state.	RIOEN (bit 12 of the PCR register) set to 0 0 is the only possible setting for this bit. 1 is a reserved function and must not be used.

3.2.4 Enable/Disable the Digital Loopback Mode

In the digital loopback mode, the receive signals are connected internally through multiplexers to the corresponding transmit signals, as shown in Table 3–10. This mode allows testing of serial port code with a single DSP device; the McBSP receives the data it transmits.

Table 3–10. Receive Signals Connected to Transmit Signals in Digital Loopback Mode

This receive signal …	Is fed internally by this transmit signal …
DR (receive data)	DX (transmit data)
FSR (receive frame synchronization)	FSX (transmit frame synchronization)
CLKR (receive clock)	CLKX (transmit clock)

The DLB bit determines whether the digital loopback mode is on.

To get this result…	Set this bit...
Disable the digital-loopback mode	DLB (bit 15 of the SPCR1 register)
Enable the digital-loopback mode	DLB (bit 15 of the SPCR1 register)

3.2.5 Enable/Disable the Clock Stop Mode

The CLKSTP bits determine whether the clock stop mode is on.

To get this result…	Set this bit...
Disable clock stop mode (normal clocking for nonSPI mode)	CLKSTP (bits 11 and 12 of the SPCR1 register) set to 0Xb
Enable clock stop mode without clock delay	CLKSTP (bits 11 and 12 of the SPCR1 register)set to 10b
Enable clock stop mode with clock delay	CLKSTP (bits 11 and 12 of the SPCR1 register) set to 11b

The clock stop mode supports the SPI master-slave protocol. If you will not be using the SPI protocol, you can clear CLKSTP to disable the clock stop mode.

In the clock stop mode, the clock stops at the end of each data transfer. At the beginning of each data transfer, the clock starts immediately (CLKSTP = 10b) or after a half-cycle delay (CLKSTP = 11b). The CLKXP bit determines whether the starting edge of the clock on the CLKX pin is rising or falling. The CLKRP bit determines whether receive data is sampled on the rising or falling edge of the clock shown on the CLKR pin.

Table 3–11 summarizes the impact of CLKSTP, CLKXP, and CLKRP on serial port operation. Note that in the clock stop mode, the receive clock is tied internally to the transmit clock, and the receive frame-sync signal is tied internally to the transmit frame-sync signal.

Table 3–11. Effects of CLKSTP, CLKXP, and CLKRP on the Clock Scheme

Bit Settings	Clock Scheme
CLKSTP = 00b or 01b CLKXP = 0 or 1 CLKRP = 0 or 1	Clock stop mode disabled. Clock enabled for non-SPI mode.
CLKSTP = 10b CLKXP = 0 CLKRP = 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.
CLKSTP = 11b CLKXP = 0 CLKRP = 1	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.
CLKSTP = 10b CLKXP = 1 CLKRP = 0	High inactive state without delay: The McBSP transmits data on the falling edge of CLKX and receives data on the rising edge of CLKR.
CLKSTP = 11b CLKXP = 1 CLKRP = 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.

3.2.6 Enable/Disable Transmit Multichannel Selection

To get this result…	**Use this bit** …
Enable all channels	XMCM (bits 0 and 1 of the MCR2 register) to 00 (No channels can be disabled or masked with this setting.)
Disable all channels unless they are selected in the appropriate transit channel enable registers (XCERs).	Set XMCM to 01b. A channel is in this mode is also unmasked.
Enable all channels as masked unless they are selected in the appropriate transmit channel enable registers (XCERs).	Set XMCM to 10b.
Enable channels for symmetric transmissions and reception.	Set XMCM to 11b.

For more details, see *Transmit Multichannel Selection Modes* on page 2-8.

3.2.7 Enable/Disable the A-bis Mode

The ABIS bit determines whether the A-bis mode is on.

To get this result…	**Use this bit** …
Enable A-bis mode	ABIS (bit 6 in SPCR1) set to 1
Disable A-bis mode	ABIS set to 0

For more details, see *A-bis Mode* on page 2-13.

3.2.8 Choose 1 or 2 Phases for the Transmit Frame

To get this result …	**Use this bit**…
Set transmit frame to single phase	XPHASE (bit 15 in XCR2 register set to 0.
Set transmit frame to dual-phase frame	XPHASE bit in XCR2 set to 1.

3.2.9 Set the Transmit Word Length(s)

To get this result …	**Use this bit**…	
Specify the length of every transmit word in Frame Phase 1	Set XWDLEN1 and XWDLEN2 (bits 7:5 of XCR1 XCR2 registers, respectively) to one of the following:	
Transmit Word Length of Frame Phase 1	000b	8 bits
	001b	12 bits
	010b	16 bits
	011b	20 bits
	100b	24 bits
	101b	32 bits
	11xb	Reserved

Each frame can have one or two phases, depending on the value that you load into the RPHASE bit. If a single-phase frame is selected, XWDLEN1 selects the length for every serial word transmitted in the frame. If a dual-phase frame is selected, XWDLEN1 determines the length of the serial words in phase 1 of the frame, and XWDLEN2 determines the word length in phase 2 of the frame.

3.2.10 Set the Transmit Frame Length

The transmit frame length is the number of serial words in the transmit frame. Each frame can have one or two phases, depending on value that you load into the XPHASE bit.

If a single-phase frame is selected (XPHASE = 0), the frame length is equal to the length of phase 1. If a dual-phase frame is selected (XPHASE = 1), the frame length is the length of phase 1 plus the length of phase 2.

The 7-bit XFRLEN fields allow up to 128 words per phase. See Table 3–12 for a summary of how to calculate the frame length. This length corresponds to the number of words or logical time slots or channels per frame-synchronization pulse.

Table 3–12. How to Calculate Frame Length

XPHASE	XFRLEN1	XFRLEN2	Frame Length
0	0 ≤ XFRLEN1 ≤ 127	Don't care	(XFRLEN1 + 1) words
1	0 ≤ XFRLEN1 ≤ 127	0 ≤ XFRLEN2 ≤ 127	(XFRLEN1 + 1) + (XFRLEN2 + 1) words

Note: Program the XFRLEN fields with [*w minus 1*], where *w* represents the number of words per phase. For the example, if you want a phase length of 128 words in phase 1, load 127 into XFRLEN1.

To get this result …	**Use this bit**…
Set transmit frame length 1	Set XFRLEN1 (bits 14:8 of XRCR1) to one of the following: 000 0000 1 word in phase 1 000 0001 2 words in phase 1 . . . 111 1111 128 words in phase 1
Set transmit frame length 2	Set XFRLEN2 (bits 14:8 of XRCR2) to one of the following: 000 0000 1 word in phase 1 000 0001 2 words in phase 1 . . . 111 1111 128 words in phase 1

3.2.10.1 Enable/Disable the Transmit Frame-Sync Ignore Function

If a frame-synchronization (frame-sync) pulse starts the transfer of a new frame before the current frame is fully transmitted, this pulse is treated as an unexpected frame-sync pulse.

When XFIG = 1, normal transmission continues with unexpected frame-sync signals ignored.

When XFIG = 0 and an unexpected frame-sync pulse occurs, the serial port:

1) Aborts the present transmission

2) Sets XSYNCERR to 1 in SPCR2

3) Re-initiates transmission of the current word that was aborted

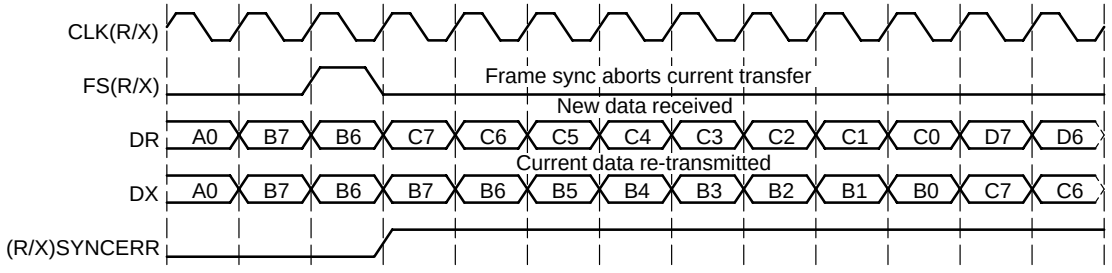
See section 5.5 for XINTM interrupt selection into FIFO logic.

If there is an unexpected receive frame-sync pulse …	**Use this bit**…
Allow an unexpected transmit frame sync pulse to restart the frame transfer	Set XFIG (bit 2 in XCR2) to 0
Ignore the unexpected pulses	Set XFIG to 1

3.2.10.2 Examples Showing the Effects of XFIG

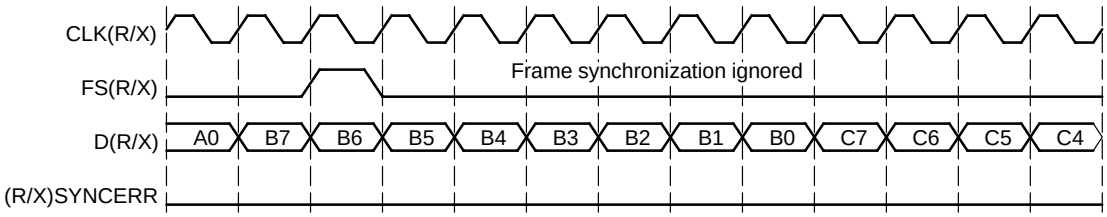
Figure 3–9 shows an example in which word B is interrupted by an unexpected frame-sync pulse when (R/X)FIG = 0. In the case of transmission, the transmission of B is aborted (B is lost). This condition is a transmit synchronization error, and thus sets the XSYNCERR bit. No new data has been written to DXR[1,2], and therefore, the McBSP transmits B again.

Figure 3–9. Unexpected Frame-Sync Pulse With (R/X)FIG = 0



In contrast with Figure 3–9, Figure 3–10 shows McBSP operation when unexpected frame-sync signals are ignored (when (R/X)FIG = 1). Here, the transfer of word B is not affected by an unexpected frame-sync pulse.

Figure 3–10. Unexpected Frame-Sync Pulse With (R/X)FIG = 1



3.2.11 Set the Transmit Companding Mode

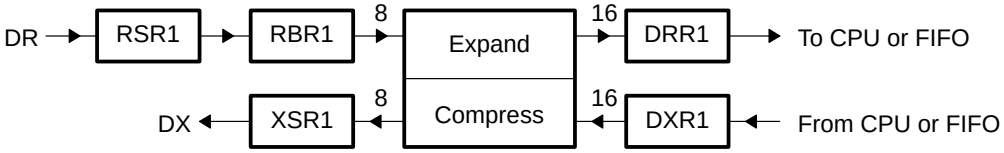
Companding (COMpressing and exPANDing) hardware allows compression and expansion of data in either μ-law or A-law format. The companding standard employed in the United States and Japan is μ-law. The European companding standard is referred to as A-law. The specifications for μ-law and A-law log PCM are part of the CCITT G.711 recommendation.

A-law and μ-law allow 13 bits and 14 bits of dynamic range, respectively. Any values outside this range are set to the most positive or most negative value. Thus, for companding to work best, the data transferred to and from the McBSP via the CPU or FIFO must be at least 16 bits wide.

The μ-law and A-law formats both encode data into 8-bit code words. Companded data is always 8 bits wide; the appropriate word length bits (RWDLEN1, RWDLEN2, XWDLEN1, XWDLEN2) must therefore be set to 0, indicating an 8-bit wide serial data stream. If companding is enabled and either of the frame phases does not have an 8-bit word length, companding continues as if the word length is 8 bits.

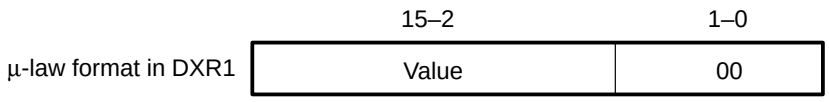
Figure 3–11 illustrates the companding processes. When companding is chosen for the transmitter, compression occurs during the process of copying data from DXR1 to XSR1. The transmit data is encoded according to the specified companding law (A-law or μ-law). When companding is chosen for the receiver, expansion occurs during the process of copying data from RBR1 to DRR1. The receive data is decoded to 2s-complement format.

Figure 3–11. Companding Processes for Reception and for Transmission



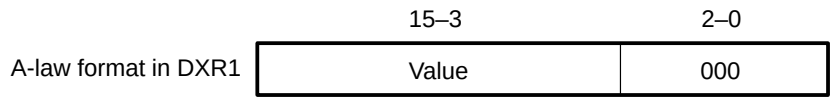
For transmission using μ-law compression, make sure the 14 data bits are left-justified in DXR1, with the remaining two low-order bits filled with 0s as shown in Figure 3–12.

Figure 3–12. μ-Law Transmit Data Companding Format



For transmission using A-law compression, make sure the 13 data bits are left-justified in DXR1, with the remaining three low-order bits filled with 0s as shown in Figure 3–13.

Figure 3–13. A-Law Transmit Data Companding Format



If the McBSP is otherwise unused (the serial port transmit and receive sections are reset), the companding hardware can compand internal data. See section 1.3.2.2 on page 1-13.

Normally, the McBSP transmit or receives all data with the most significant bit (MSB) first. However, certain 8-bit data protocols (that do not use companded data) require the least significant bit (LSB) to be transferred first. If you set XCOMPAND = 01b in XCR2, the bit ordering of 8-bit words is reversed (LSB first) before being sent from the serial port. Similar to companding, this feature is enabled only if the appropriate word length bits are set to 0, indicating that 8-bit words are to be transferred serially. If either phase of the frame does not have an 8-bit word length, the McBSP assumes the word length is eight bits, and LSB-first ordering is done.

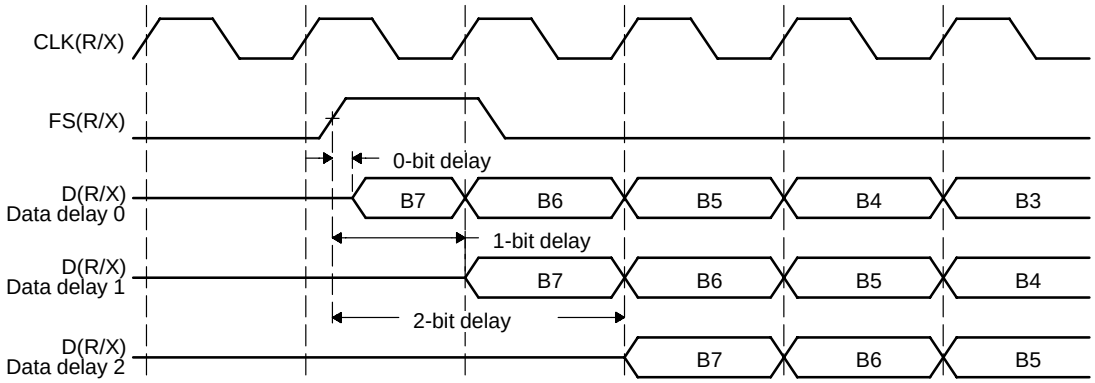
To get this result …	**Use this bit**…	
Transmit companding mode	Set XCOMPAND (bits 4:3 in XCR2) to one of the following:	
	00	No companding, any size data, MSB transmitted first
	01	No companding, 8-bit data, LSB transmitted first
	10	μ-law companding, 8-bit data, MSB received first
	11	A-law companding, 8-bit data, MSB received first

3.2.12 Set the Transmit Data Delay

The start of a frame is defined by the first clock cycle in which frame synchronization is found to be 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.

XDATDLY specifies the data delay for transmission. The range of programmable data delay is zero to two bit-clocks (XDATDLY = 00b–10b), as described in Figure 3–14. In this figure, the data transferred is an 8-bit value with bits labeled B7, B6, B5, and so on. Typically a 1-bit delay is selected, because data often follows a 1-cycle active frame-sync pulse.

Figure 3–14. Range of Programmable Data Delay



3.2.12.1 0-Bit Data Delay

Normally, a frame-sync pulse is detected or sampled with respect to an edge of serial clock internal CLK(R/X). Thus, on the following cycle or later (depending on the data delay value), data may be received or transmitted. However, in the case of 0-bit data delay, the data must be ready for reception and/or transmission on the same serial clock cycle.

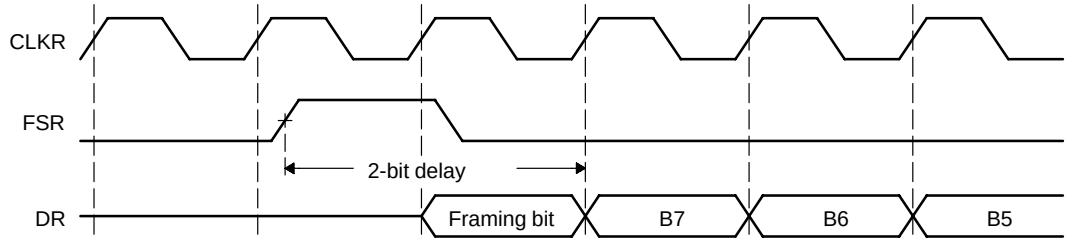
For reception this problem is solved because receive data is sampled on the first falling edge of CLKR where an active-high internal FSR is detected. However, data transmission must begin on the rising edge of the internal CLKX clock that generated the frame synchronization. Therefore, the first data bit is assumed to be present in XSR1, and thus DX. The transmitter then asynchronously detects the frame synchronization, FSX, going active high, and immediately starts driving the first bit to be transmitted on the DX pin.

3.2.12.2 2-Bit Data Delay

A data delay of two bit periods allows the serial port to interface to different types of T1 framing devices where the data stream is preceded by a framing

bit. During reception of such a stream with data delay of two bits (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 the following figure. In this figure, the data transferred is an 8-bit value with bits labeled B7, B6, B5, and so on.

Figure 3–15. 2-Bit Data Delay Used to Skip a Framing Bit



To get this result …	**Use this bit**…	
Set the transmit data delay	Set XDATDLY (bits 1:0 in XCR2 to:	
	00	0-bit data delay
	01	1-bit data delay
	10	2-bit data delay
	11	Reserved

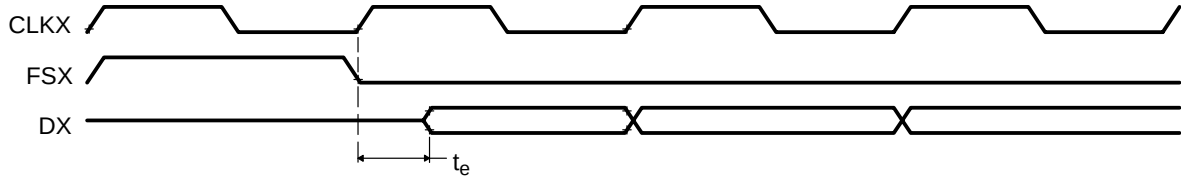
3.2.13 Set the Transmit DXENA Mode

The DXENA bit controls the delay enabler on the DX pin. Set DXENA to enable an extra delay for turn-on time (for the length of the delay, see the data sheet for your TMS320C55x DSP). Note that this bit does not control the data itself, so only the first bit is delayed, unless the A-bis mode is on. In the A-bis mode, any bit can be delayed because any bit can go from the high-impedance state to the valid state.

If you tie together the DX pins of multiple McBSPs, make sure DXENA = 1 to avoid having more than one McBSP transmit on the data line at one time.

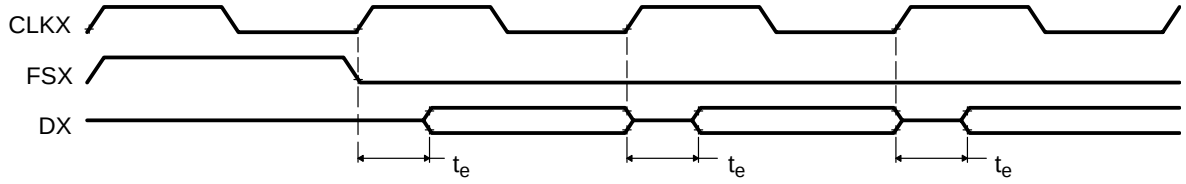
The following two figures show the timing of the DX pin for DXENA = 1. The first figure shows the effect of the DX delay enabler when the A-bis mode is off. The second figure shows the effect of the DX delay enabler when the A-bis mode is on.

Figure 3–16. DX Delay When A-bis Mode is Off



Note: te = extra delay for turn on time with DXENA = 1

Figure 3–17. DX Delays When A-bis Mode is On



Note: te = extra delay for turn on time with DXENA = 1

To get this result …	**Use this bit**…
Turn DX delay enabler mode off	Set DXENA (bit 7 in SPCR1) to 0
Turn DX delay enabler mode on	Set DXENA (bit 7 in SPCR1) to 1

3.2.14 Set the Transmit Interrupt Mode

The transmitter interrupt (XINT) signals the CPU of changes to the serial port status. Four options exist for configuring this interrupt. The options are set by the transmit interrupt mode bits, XINTM, in SPCR2.

- ❑ XINTM = 00b. Interrupt on every serial word by tracking the XRDY bit in SPCR2. Note that regardless of the value of XINTM, XRDY can be read to detect the XRDY = 1 condition.
- ❑ XINTM = 01b. In any of the transmit multichannel selection modes, interrupt after every 16-channel block boundary has been crossed within a frame and at the end of the frame. In any other serial transfer case, this setting is not applicable and, therefore, no interrupts are generated.
- ❑ XINTM = 10b. Interrupt on detection of each transmit frame-sync pulse. This generates an interrupt even when the transmitter is in its reset state. This is done by synchronizing the incoming frame-sync pulse to the CPU clock and sending it to the CPU via XINT.
- ❑ XINTM = 11b. Interrupt on frame-synchronization error. Note that regardless of the value of XINTM, XSYNCERR can be read to detect this

condition. For more information on using XSYNCERR, see *Unexpected Transmit Frame-Sync Pulse* on page 1-46.

To get this result …	Use this bit…	
Set the transmit interrupt mode	Set XINTM (bits 5:4 in SPCR2 to:	
	00	XINT generated when XRDY changes from 0 to 1
	01	XINT generated by an end-of-block or end-of-frame condition in the transmit multichannel selection mode
	10	XINT generated by a new transmit frame-sync pulse
	11	XINT generated when XSYNCERR is set

3.2.15 Set the Transmit Frame-Sync Mode

Table 3–13 shows how FSXM and FSGM select the source of transmit frame-sync pulses. The three choices are:

- ❏ External frame-sync input
- ❏ Sample rate generator frame-sync signal (FSG).
- ❏ Internal signal that indicates a DXR-to-XSR copy has been made

Table 3–13 also shows the effect of each bit setting on the FSX pin. The polarity of the signal on the FSX pin is determined by the FSXP bit.

Table 3–13. How FSXM and FSGM Select the Source of Transmit Frame-Sync Pulses

FSXM	FSGM	Source of Transmit Frame Synchronization	FSX Pin Status
0	0 or 1	An external frame-sync signal enters the McBSP through the FSX pin. The signal is then inverted by FSXP before being used as internal FSX.	Input
1	1	Internal FSX is driven by the sample rate generator frame-sync signal (FSG).	Output. FSG is inverted by FSXP before being driven out on FSX pin.
1	0	A DXR-to-XSR copy causes the McBSP to generate a transmit frame-sync pulse that is 1 cycle wide.	Output. The generated frame-sync pulse is inverted as determined by FSXP before being driven out on FSX pin.

If the sample rate generator creates a frame-sync signal (FSG) that is derived from an external input clock, the GSYNC bit determines whether FSG is kept synchronized with pulses on the FSR pin. For more details, see *Synchronizing Sample Rate Generator Outputs to an External Clock* on page 1-31.

In the clock stop mode (CLKSTP = 10b or 11b), the McBSP can act as a master or as a slave in the SPI protocol. If the McBSP is a master and must provide a slave-enable signal ($\overline{SS}$) on the FSX pin, make sure that FSXM = 1 and FSGM = 0, so that FSX is an output and is driven active for the duration of each transmission. If the McBSP is a slave, make sure that FSXM = 0, so that the McBSP can receive the slave-enable signal on the FSX pin.

To get this result …	**Use this bit**…
To supply transmit frame synchronization mode by an external source via the FSX pin	Set FSXM (bit 11 in PCR) to 0.
To have the McBSP supply the transmit frame synchronization mode by way of the FSGM bit of SRGR2.	Set FSXM to 1.
Have the McBSP generate a transmit frame-sync pulse when the content of DXR is copied to XSR	Set FSGM (bit 12 of SRGR2) to 0. This mode is used when FSXM = 1.
Use frame-sync pulses generated by the sample rate generator	Set FSGM (bit 12 of SRGR2) to 1.

3.2.16 Set the Transmit Frame-Sync Polarity

Transmit frame-sync pulses can be either generated internally by the sample rate generator (see section 1.4.2 on page 1-30) or driven by an external source. The source of frame sync is selected by programming the mode bit, FSXM, in PCR. FSX is also affected by the FSGM bit in SRGR2. For information about the effects of FSXM and FSGM, see *Set the Transmit Frame-Sync Mode* on page 3-39). Similarly, transmit clocks can be selected to be inputs or outputs by programming the mode bit, CLKXM, in the PCR (see *Set the Transmit Clock Mode* on page 3-43).

When FSR and FSX are inputs (FSXM = FSRM= 0, external frame-sync pulses), the McBSP detects them on the internal falling edge of clock, internal CLKR, and internal CLKX, respectively. The receive data arriving at the DR pin is also sampled on the falling edge of internal CLKR. Note that these internal clock signals are either derived from external source via CLK(R/X) pins or driven by the sample rate generator clock (CLKG) internal to the McBSP.

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

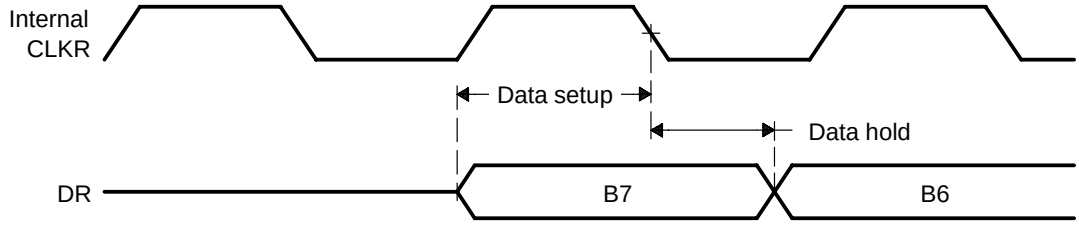
FSRP, FSXP, CLKRP, and CLKXP in the pin control register (PCR) configure the polarities of the FSR, FSX, CLKR, and CLKX signals, respectively. All frame-sync signals (internal FSR, internal FSX) that are internal to the serial port are active high. If the serial port is configured for external frame synchronization (FSR/FSX are inputs to McBSP), and FSRP = FSXP = 1, the external active-low frame-sync signals are inverted before being sent to the receiver (internal FSR) and transmitter (internal FSX). Similarly, if internal synchronization (FSR/FSX are output pins and GSYNC = 0) is selected, the internal active-high frame-sync signals are inverted, if the polarity bit FS(R/X)P = 1, before being sent to the FS(R/X) pin. The figure in the topic Clock and Frame Generation 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. Note that data is always transmitted on the rising edge of internal CLKX. 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 selected (CLKXM = 1 and CLKX is an output pin), the internal (rising-edge triggered) clock, internal CLKX, is inverted before being sent out on the CLKX pin.

Similarly, the receiver can reliably sample data that is clocked with a rising edge clock (by the transmitter). The receive clock polarity bit, CLKRP, sets the edge used to sample received data. Note that the receive data is always sampled on the falling edge of internal CLKR. 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 triggered 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 triggered clock before being sent out on the CLKR pin.

Note that CLKRP = CLKXP in a system where the same clock (internal or external) is used to clock the receiver and transmitter. The receiver uses the opposite edge as the transmitter to ensure valid setup and hold of data around this edge. Figure 3–18 shows how data clocked by an external serial device using a rising edge, may be sampled by the McBSP receiver on the falling edge of the same clock.

Figure 3–18. Data Clocked Externally Using a Rising Edge and Sampled by the McBSP Receiver on a Falling Edge



To get this result …	Use this bit…
Set frame-synchronization pulse FSX to active high	Set FSXP (bit 3 in the PCR register) to 0
Set frame-synchronization pulse FSX to active low	Set FSXP (bit 3 in the PCR register) to 1

3.2.17 Set the SRG Frame-Sync Period and Pulse Width

To get this result …	Use this bit…
Set the sample rate generator frame-sync period	For the frame-sync signal FSG, (FPER + 1) determines the period from the start of a frame-sync pulse to the start of the next frame-sync pulse. Set FPER (bits 11–0 in the SRGR2 register) from 1 to 4096 cycles.
Set the sample rate generator frame-sync pulse width	FWID (bits 15–8 in SRGR1) plus 1 determines the width of each frame-sync pulse (from 1 to 256 CLKG cycles).

The sample rate generator can produce a clock signal, CLKG, and a frame-sync signal, FSG. If the sample rate generator is supplying receive or transmit frame synchronization, you must program the bit fields FPER and FWID.

On FSG, the period from the start of a frame-sync pulse to the start of the next pulse is (FPER + 1) CLKG cycles. The 12 bits of FPER allow a frame-sync period of 1 to 4096 CLKG cycles, which allows up to 4096 data bits per frame. When GSYNC = 1, FPER is a don’t care value.

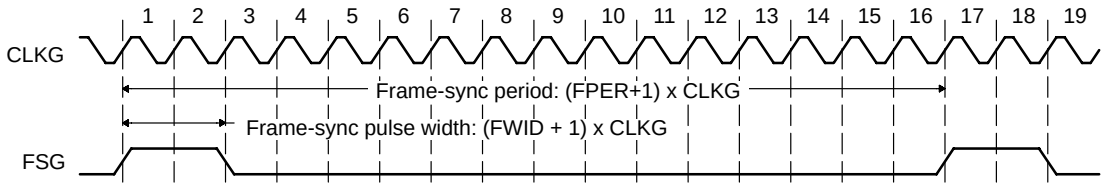
Each pulse on FSG has a width of (FWID + 1) CLKG cycles. The eight bits of FWID allow a pulse width of 1 to 256 CLKG cycles. It is recommended that FWID be programmed to a value less than the programmed word length.

The values in FPER and FWID are loaded into separate down-counters. The 12-bit FPER counter counts down the generated clock cycles from the

programmed value (4095 maximum) to 0. The 8-bit FWID counter counts down from the programmed value (255 maximum) to 0.

Figure 3–19 shows a frame-sync period of 16 CLKG periods (FPER = 15 or 00001111b) and a frame-sync pulse with an active width of 2 CLKG periods (FWID = 1).

Figure 3–19. Frame of Period 16 CLKG Periods and Active Width of 2 CLKG Periods



When the sample rate generator comes out of reset, FSG is in its inactive state. Then, when $\overline{\text{FRST}}$ = 1 and FSGM = 1, a frame-sync pulse is generated. The frame width value (FWID + 1) is counted down on every CLKG cycle until it reaches 0, at which time FSG goes low. At the same time, the frame period value (FPER + 1) is also counting down. When this value reaches 0, FSG goes high, indicating a new frame.

3.2.18 Set the Transmit Clock Mode

To get this result …	**Use this bit**…
Set the transmitter to get its clock signal from an external source via the CLKX pin.	Set CLKXM (bit 9 in the PCR register) to 0
Set the CLKX output pin driven by the sample rate generator	Set CLKXM (bit 9 in the PCR register) to 1

3.2.18.1 Selecting a Source for the Transmit Clock and a Data Direction for the CLKX Pin

Table 3–14 shows how the CLKXM bit selects the transmit clock and the corresponding status of the CLKX pin. The polarity of the signal on the CLKX pin is determined by the CLKXP bit.

Table 3–14. How the CLKXM Bit Selects the Transmit Clock and the Corresponding Status of the CLKX Pin

CLKXM in PCR	Source of Transmit Clock	CLKX Pin Status
0	Internal CLKX is driven by an external clock on the CLKX pin. CLKX is inverted as determined by CLKXP before being used.	Input
1	Internal CLKX is driven by the sample rate generator clock, CLKG.	Output. CLKG, inverted as determined by CLKXP, is driven out on CLKX.

If the sample rate generator creates a clock signal (CLKG) that is derived from an external input clock, the GSYNC bit determines whether CLKG is kept synchronized with pulses on the FSR pin. For more details, see *Synchronizing Sample Rate Generator Outputs to an External Clock* on page 1-31.

In the clock stop mode (CLKSTP = 10b or 11b), the McBSP can act as a master or as a slave in the SPI protocol. If the McBSP is a master, make sure that CLKXM = 1, so that CLKX is an output to supply the master clock to any slave devices. If the McBSP is a slave, make sure that CLKXM = 0, so that CLKX is an input to accept the master clock signal.

3.2.18.2 Set the Transmit Clock Polarity

To get this result …	**Use this bit**…
Sample transmit data on the rising edge of CLKX	Set CLKXP (bit 1 of PCR) to 0.
Sample transmit data on the falling edge of CLKX	Set CLKXP (bit 1 of PCR) to 1.

Transmit frame-sync pulses can be either generated internally by the sample rate generator (see section 1.4.2 on page 1-30) or driven by an external source. The source of frame sync is selected by programming the mode bit, FSXM, in PCR. FSX is also affected by the FSGM bit in SRGR2. For information about the effects of FSXM and FSGM, see *Set the Transmit Frame-Sync Mode* . Similarly, transmit clocks can be selected to be inputs or outputs by programming the mode bit, CLKXM, in the PCR (see *Set the Transmit Clock Mode* on page 3-43).

When FSR and FSX are inputs (FSXM = FSRM= 0, external frame-sync pulses), the McBSP detects them on the internal falling edge of clock, internal CLKR, and internal CLKX, respectively. The receive data arriving at the DR pin

is also sampled on the falling edge of internal CLKR. Note that these internal clock signals are either derived from external source via CLK(R/X) pins or driven by the sample rate generator clock (CLKG) internal to the McBSP.

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

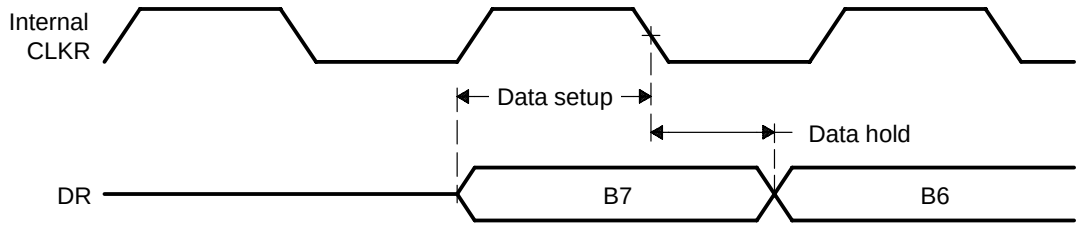
FSRP, FSXP, CLKRP, and CLKXP in the pin control register (PCR) configure the polarities of the FSR, FSX, CLKR, and CLKX signals, respectively. All frame-sync signals (internal FSR, internal FSX) that are internal to the serial port are active high. If the serial port is configured for external frame synchronization (FSR/FSX are inputs to McBSP), and FSRP = FSXP = 1, the external active-low frame-sync signals are inverted before being sent to the receiver (internal FSR) and transmitter (internal FSX). Similarly, if internal synchronization (FSR/FSX are output pins and GSYNC = 0) is selected, the internal active-high frame-sync signals are inverted, if the polarity bit FS(R/X)P = 1, before being sent to the FS(R/X) pin. The figure in the topic Clock and Frame Generation 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. Note that data is always transmitted on the rising edge of internal CLKX. 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 selected (CLKXM = 1 and CLKX is an output pin), the internal (rising-edge triggered) clock, internal CLKX, is inverted before being sent out on the CLKX pin.

Similarly, the receiver can reliably sample data that is clocked with a rising edge clock (by the transmitter). The receive clock polarity bit, CLKRP, sets the edge used to sample received data. Note that the receive data is always sampled on the falling edge of internal CLKR. 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 triggered 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 triggered clock before being sent out on the CLKR pin.

Note that CLKRP = CLKXP in a system where the same clock (internal or external) is used to clock the receiver and transmitter. The receiver uses the opposite edge as the transmitter to ensure valid setup and hold of data around this edge. Figure 3–20 shows how data clocked by an external serial device using a rising edge, may be sampled by the McBSP receiver on the falling edge of the same clock.

Figure 3–20. Data Clocked Externally Using a Rising Edge and Sampled by the McBSP Receiver on a Falling Edge



3.2.19 Set the SRG Clock Divide-Down Value

To get this result …	**Use this bit**…
Set the divide down value of the sample rate generator clock	Set CLKGDV (bits 7–0 of SRGR1) to generate the required SRG clock frequency. The default value is 1 (divide input clock by 2).

The first divider stage generates the serial data bit clock from the input clock. This divider stage utilizes a counter, preloaded by CLKGDV, that contains the divide ratio value.

The output of the first divider stage is the data bit clock, which is output as CLKG and which serves as the input for the second and third stages of the divider.

CLKG has a frequency equal to 1/(CLKGDV + 1) of sample rate generator input clock. Thus, the sample generator input clock frequency is divided by a value between 1 and 256. When CLKGDV is odd or equal to 0, the CLKG duty cycle is 50%. When CLKGDV is an even value, 2p, representing an odd divide-down, the high-state duration is p+1 cycles and the low-state duration is p cycles.

3.2.19.1 Set the SRG Clock Synchronization Mode

GSYNC is used only when the input clock source for the sample rate generator is external—on the CLKR or CLKX pin.

For more details on using the clock synchronization feature, see *Synchronizing Sample Rate Generator Outputs to an External Clock* on page 1-31.

To get this result …	Use this bit…
The sample rate generator clock (CLKG) is free running. CLKG oscillates without adjustment, and FSG pulses every (FPER + 1) CLKG cycles.	Set GSYNC (bit 15 of SRGR2) to 0
Clock synchronization is performed. When a pulse is detected on the FSR pin: ❑ CLKG is adjusted as necessary so that it is synchronized with the input clock on the CLKR or CLKX pin. ❑ FSG pulses. FSG **only** pulses in response to a pulse on the FSR pin. The frame-sync period defined in FPER is ignored.	Set GSYNC (bit 15 of SRGR2) to 1

3.2.20 Set the SRG Clock Mode (Choose an Input Clock)

To get this result …	Use this bit…
Sample rate generator clock derived from CLKS pin	Set SCLKME (bit 7 of PCR) = 0 and CLKSM (bit 13 of SRGR2) = 0
Sample rate generator clock derived from CPU clock (This is the condition forced by a DSP reset.)	SCLKME = 0 CLKSM = 1
Sample rate generator clock derived from CLKR pin	SCLKME = 1 CLKSM = 0
Sample rate generator clock derived from CLKX pin	SCLKME = 1 CLKSM = 1

The sample rate generator can produce a clock signal (CLKG) for use by the receiver, the transmitter, or both, but CLKG is derived from an input clock. The preceding table shows the four possible sources of the input clock. For more details on generating CLKG, see *Clock Generation in the Sample Rate Generator* on page 1-27.

3.2.20.1 Set the SRG Input Clock Polarity

CLKSP determines the input clock polarity when the CLKS pin supplies the input clock (SCLKME = 0 and CLKSM = 0).

To get this result …	Use this bit…
Rising edge on CLKS pin generates CLKG and FSG.	Set CLKSP (bit 14 of SRGR2) to 0
Falling edge on CLKS pin generates CLKG and FSG.	Set CLKSP (bit 14 of SRGR2) to 1

CLKXP determines the input clock polarity when the CLKX pin supplies the input clock (SCLKME = 1 and CLKSM = 1).

To get this result …	Use this bit…
Rising edge on CLKX pin generates transitions on CLKG and FSG.	Set CLKXP (bit 1 of PCR) to 0
Falling edge on CLKX pin generates transitions on CLKG and FSG.	Set CLKXP to 1

CLKRP determines the input clock polarity when the CLKR pin supplies the input clock (SCLKME = 1 and CLKSM = 0).

To get this result …	Use this bit…
Rising edge on CLKR pin generates transitions on CLKG and FSG.	Set CLKRP (bit 0 of PCR) to 1
Falling edge on CLKR pin generates transitions on CLKG and FSG.	Set CLKRP to 0

3.2.20.2 Using CLKXP/CLKRP to Choose an Input Clock Polarity

The sample rate generator can produce a clock signal (CLKG) and a frame-sync signal (FSG) for use by the receiver, the transmitter, or both. To produce CLKG and FSG, the sample rate generator must be driven by an input clock signal derived from the CPU clock or from an external clock on the CLKX, or CLKR pin. If you use a pin, choose a polarity for that pin by using the appropriate polarity bit (CLKXP for the CLKX pin, CLKRP for the CLKR pin). The polarity determines whether the rising or falling edge of the input clock generates transitions on CLKG and FSG.

Chapter 4

Emulation and Reset Considerations

This section covers emulation mode and how to reset and initialize the various parts of the McBSP.

Topic **Page**

4.1	McBSP Emulation Mode	4-2
4.2	Data Packing Examples	4-8
4.3	GPIO Function	4-11

4.1 McBSP Emulation Mode

FREE and SOFT are special emulation bits in SPCR2 that determine the state of the McBSP when a breakpoint is encountered in the high-level language debugger. If FREE = 1, upon a software breakpoint the clock continues to run and data is still shifted out. When FREE = 1, the SOFT bit is a *don't care.*

If FREE = 0, the SOFT bit takes effect: If SOFT = 0 when breakpoint occurs, the clock stops immediately, thus aborting a transmission. If SOFT = 1 and a breakpoint occurs while transmission is in progress, the transmission continues until completion of the transfer, and then the clock halts. These options are listed in the following table.

The McBSP receiver functions in a similar fashion. Note that if a mode other than the immediate stop mode (SOFT = FREE = 0) is chosen, the receiver continues running and an overrun error is possible.

Table 4–1. McBSP Emulation Modes Selectable With the FREE and SOFT Bits of SPCR2

FREE	SOFT	McBSP Emulation Mode
0	0	Immediate stop mode (reset condition) The transmitter or receiver stops immediately in response to a breakpoint.
0	1	Soft stop mode When a breakpoint 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 a breakpoint occurs.

4.1.1 Resetting and Initializing a McBSP

Table 4–2 shows the state of McBSP pins when the serial port is reset due to a DSP reset and due to a direct receiver or transmitter reset.

Table 4–2. Reset State of Each McBSP Pin

Pin	Possible State(s)	State Forced By DSP Reset	State Forced By Receiver/Transmitter Reset
			Receiver Reset ($\overline{RRST}$ = 0 and $\overline{GRST}$ = 1)
DR	I	Input	Input
CLKR	I/O/Z	Input	Known state if Input; CLKR running if output
FSR	I/O/Z	Input	Known state if Input; FSRP inactive state if output
			Transmitter Reset ($\overline{XRST}$ = 0 and $\overline{GRST}$ = 1)
DX	O/Z	High impedance	High impedance
CLKX	I/O/Z	Input	Known state if Input; CLKX running if output
FSX	I/O/Z	Input	Known state if Input; FSXP inactive state if output

Note: In Possible State(s) column, I = Input, O = Output, Z = High impedance
In the 28x family, at device reset, all I/Os default to GPIO function and generally as inputs.

When the McBSP is reset in either of the above two ways, the machine is reset to its initial state, including reset of all counters and status bits. The receive status bits include RFULL, RRDY, and RSYNCERR. The transmit status bits include $\overline{XEMPTY}$, XRDY, and XSYNCERR.

- **DSP reset.** When the whole DSP is reset ($\overline{XRSN}$ signal is driven low), the entire serial port, including the transmitter, receiver, and the sample rate

generator, is reset. All input-only pins and three-state pins should be in a known state. The output-only pin DX is in the high-impedance state.

The DSP reset forces the sample rate generator clock, CLKG, to have the frequency of LSPCLK clock. No pulses are generated on the sample rate generator's frame-sync signal, FSG.

When the device is pulled out of reset, the serial port remains in the reset state. In this state the DR and DX pins may be used as general-purpose I/O pins as described in section 4.3 on page 4-11.

- **McBSP reset.** When the receiver and transmitter reset bits, $\overline{\text{RRST}}$ and $\overline{\text{XRST}}$, are loaded with 0s, the respective portions of the McBSP are reset, and activity in the corresponding section of the serial port stops. All input-only pins, such as DR, and all other pins that are configured as inputs, are in a known state. The FSR and FSX pins are driven to their inactive state if they are not outputs. If the CLKR and CLKX pins are programmed as outputs, they will be driven by CLKG, provided that $\overline{\text{GRST}}$ = 1. Lastly, the DX pin will be in the high-impedance state when the transmitter and/or the device is reset.

  During normal operation, the sample rate generator is reset if the $\overline{\text{GRST}}$ bit is cleared. $\overline{\text{GRST}}$ should be 0 only when neither the transmitter nor the receiver is using the sample rate generator. In this case, the internal sample rate generator clock (CLKG) and its frame-sync signal (FSG) are driven inactive low.

  When the sample rate generator is not in the reset state ($\overline{\text{GRST}}$ = 1), pins FSR and FSX are in an inactive state when $\overline{\text{RRST}}$ = 0 and $\overline{\text{XRST}}$ = 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 $\overline{\text{FRST}}$ = 1 and its frame synchronization is driven by FSG.

- **Sample rate generator reset.** The sample rate generator is reset when the DSP is reset or when $\overline{\text{GRST}}$ is loaded with 0. In the case of a DSP reset, the sample rate generator clock, CLKG, is driven by the LSPCLK, and the frame-sync signal, FSG, is driven inactive low.

  When neither the transmitter nor the receiver is fed by CLKG and FSG, you can reset the sample rate generator by clearing $\overline{\text{GRST}}$. In this case, CLKG and FSG are driven inactive low. If you then set $\overline{\text{GRST}}$, CLKG starts and runs as programmed. Later, if $\overline{\text{FRST}}$ = 1, FSG pulses active high after the programmed number of CLKG cycles has elapsed.

4.1.1.1 McBSP Initialization Procedure

The serial port initialization procedure is as follows:

1) Make $\overline{\text{XRST}}$ = $\overline{\text{RRST}}$ = $\overline{\text{FRST}}$ = 0 in SPCR[1,2]. If coming out of a DSP reset, this step is not required.

2) While the serial port is in the reset state, program only the McBSP configuration registers (not the data registers) as required.

3) Wait for two clock cycles. This ensures proper internal synchronization.

4) Set up data acquisition as required (such as writing to DXR[1,2]).

5) Make $\overline{\text{XRST}}$ = $\overline{\text{RRST}}$ = 1 to enable the serial port. Make sure that as you set these reset bits, you do not modify any of the other bits in SPCR1 and SPCR2. Otherwise, you will change the configuration you selected in step 2.

6) Set $\overline{\text{FRST}}$ = 1, if internally generated frame synchronization is required.

7) Wait two clock cycles for the receiver and transmitter to become active.

Alternatively, on either write (step 1 or 5), the transmitter and receiver may be placed in or taken out of reset individually by modifying the desired bit.

The above procedure for reset/initialization can be applied in general when the receiver or transmitter has to be reset during its normal operation, and also when the sample rate generator is not used for either operation.

Notes:

1) The necessary duration of the active-low period of $\overline{\text{XRST}}$ or $\overline{\text{RRST}}$ is at least two CLKR/CLKX cycles.

2) The appropriate bits in serial port configuration registers SPCR[1,2], PCR, RCR[1,2], XCR[1,2], and SRGR[1,2] should only be modified when the affected portion of the serial port is in its reset state.

3) In most cases, the data transmit registers (DXR[1,2]) should be loaded by the CPU or by the or FIFO only when the transmitter is enabled ($\overline{\text{XRST}}$ = 1). An exception to this rule is when these registers are used for companding internal data (see section 1.3.2.2 on page 1-13).

4) The bits of the channel control registers—MCR[1,2], RCER[A–H], XCER[A–H]—can be modified at any time as long as they are not being used by the current reception/transmission in a multichannel selection mode.

4.1.1.2 Example: Resetting the Transmitter While the Receiver is Running

The following example shows values in the control registers that reset and configure the transmitter while the receiver is running.

Example 4–1. Resetting and Configuring the McBSP Transmitter While the McBSP Receiver is Running – Non-FIFO Mode

```
SPCR1 = 0001h      ; The receiver is running with the receive
SPCR2 = 0030h      ; interrupt (RINT) triggered by the
                   ; receiver ready bit (RRDY). The
                   ; transmitter is in its reset state. The
                   ; transmit interrupt (XINT) will be
                   ; triggered by the transmit frame–sync
                   ; error bit (XSYNCERR).

PCR = 0900h        ; Transmit frame synchronization is
                   ; generated internally according to the
                   ; FSGM bit of SRGR2. The transmit clock
                   ; is driven by an external source. The
                   ; receive clock continues to be driven by
                   ; sample rate generator. The input clock
                   ; of the sample rate generator is supplied
                   ; by the CLKS pin or by the CPU clock
                   ; depending on the CLKSM bit of SRGR2.

SRGR1 = 0001h      ; The CPU clock is the input clock for
SRGR2 = 2000h      ; the sample rate generator. The sample
                   ; rate generator divides the CPU clock by
                   ; 2 to generate its output clock (CLKG).
                   ; Transmit frame synchronization is tied
                   ; to the automatic copying of data from
                   ; the DXR(s) to the XSR(s).
```

```
XCR1 = 0740h      ; The transmit frame has two phases.
XCR2 = 8321h      ; Phase 1 has eight 16–bit words. Phase 2
                  ; has four 12–bit words. There is 1–bit
                  ; data delay between the start of a
                  ; frame–sync pulse and the first data bit
                  ; transmitted.

SPCR2 = 0x0031    ; The transmitter is taken out of reset.
```

4.2 Data Packing Examples

This section shows two ways you can implement data packing in the McBSP.

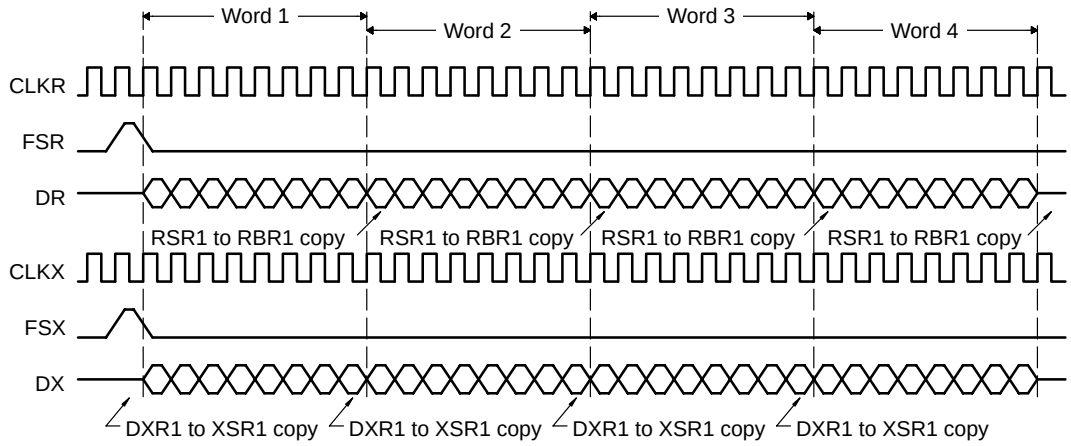
4.2.1 Data Packing Using Frame Length and Word Length

The frame length and word length can be manipulated to effectively pack data. For example, consider a situation where four 8-bit words are transferred in a single-phase frame as shown in Figure 4–1. In this case:

- (R/X)PHASE = 0: Single-phase frame
- (R/X)FRLEN1 = 0000011b: 4-word frame
- (R/X)WDLEN1 = 000b: 8-bit words

Four 8-bit data words are transferred to and from the McBSP by the CPU or by the FIFO. Thus, four reads from DRR1 and four writes to DXR1 are necessary for each frame.

Figure 4–1. Four 8-Bit Data Words Transferred To/From the McBSP



This data can also be treated as a single-phase frame consisting of one 32-bit data word, as shown in Figure 4–2. In this case:

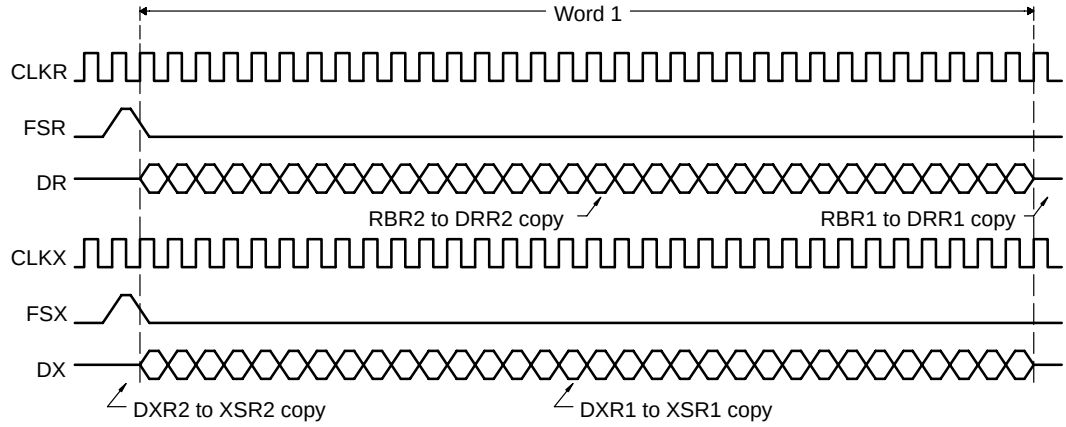
- (R/X)PHASE = 0: Single-phase frame
- (R/X)FRLEN1 = 0000000b: 1-word frame
- (R/X)WDLEN1 = 101b: 32-bit word

Two 16-bit data words are transferred to and from the McBSP by the CPU or FIFO. Thus, two reads, from DRR2 and DRR1, and two writes, to DXR2 and DXR1, are necessary for each frame. This results in only half the number of transfers compared to the previous case. This manipulation reduces the percentage of bus time required for serial port data movement.

Note:

When the word length is larger than 16 bits, make sure you access DRR2/DXR2 before you access DRR1/DXR1. McBSP activity is tied to accesses of DRR1/DXR1. During the reception of 24-bit or 32-bit words, read DRR2 and then read DRR1. Otherwise, the next RBR[1,2]-to-DRR[1,2] copy occurs before DRR2 is read. Similarly, during the transmission of 24-bit or 32-bit words, write to DXR2 and then write to DXR1. Otherwise, the next DXR[1,2]-to-XSR[1,2] copy occurs before DXR2 is loaded with new data.

Figure 4–2. One 32-Bit Data Word Transferred To/From the McBSP



4.2.2 Data Packing Using Word Length and the Frame-Sync Ignore Function

When there are multiple words per frame, you can implement data packing by increasing the word length (defining a serial word with more bits) and by ignoring frame-sync pulses. First, consider Figure 4–3, which shows the McBSP operating at the maximum packet frequency. Here, each frame only has a single 8-bit word. Note the frame-sync pulse that initiates each frame transfer for reception and for transmission. For reception, this configuration requires one read operation for each word. For transmission, this configuration requires one write operation for each word.

Figure 4–3. 8-Bit Data Words Transferred at Maximum Packet Frequency

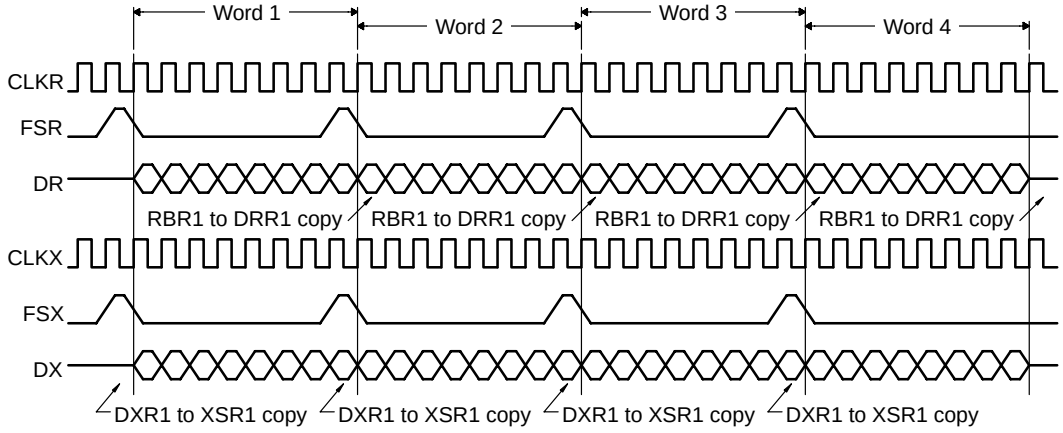
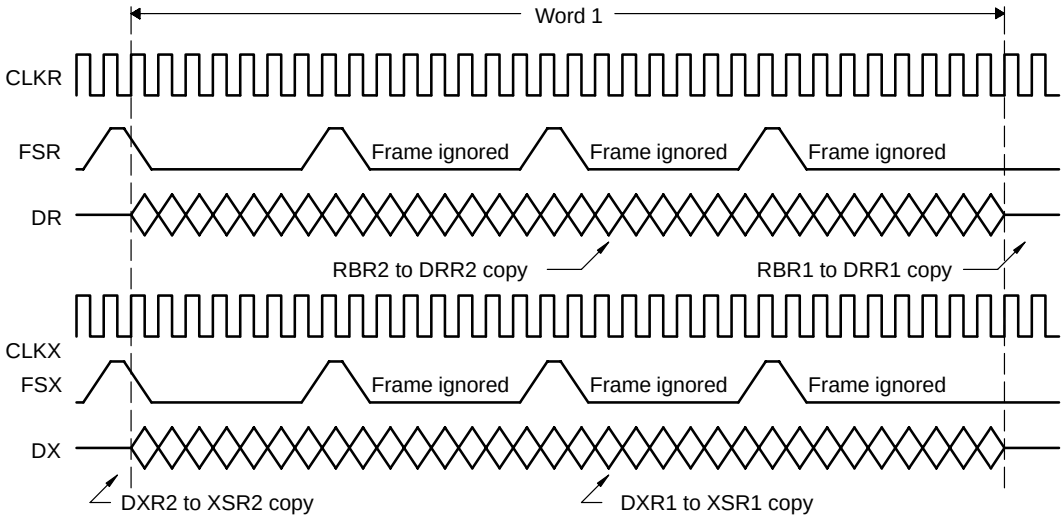


Figure 4–4 shows the McBSP configured to treat this data stream as a continuous 32-bit word. In this example, the McBSP responds to an initial frame-sync pulse. However, (R/X)FIG = 1 so that the McBSP ignores subsequent pulses. Only two read transfers or two write transfers are needed every 32 bits. This configuration effectively reduces the required bus bandwidth to half the bandwidth needed to transfer four 8-bit words.

Figure 4–4. Configuring the Data Stream of Figure 4–3 as a Continuous 32-Bit Word



4.3 GPIO Function

Note: General-purpose I/O Selection

Do not use the GPIO function using RIOEN/XIOEN bits 12 and 13 in the PCR register. This feature is not applicable to the 28x McBSP implementation; therefore, these bits are reserved on 28x devices.

Chapter 5

McBSP FIFO and Interrupts

This chapter describes the FIFO interface logic and how interrupts are handled in the McBSP module.

Topic **Page**

5.1	McBSP FIFO Overview	5-2
5.2	McBSP Functionality and Limitation Under FIFO Mode	5-3
5.3	McBSP FIFO Operation	5-5
5.4	McBSP Receive Interrupt Generation	5-7
5.5	McBSP Transmit Interrupt Generation	5-9
5.6	McBSP Register Descriptions	5-13

5.1 McBSP FIFO Overview

McBSP module is interfaced with a 16x16 bit (16 level) FIFO for each of the data registers, DRR2/DRR1and DXR2/DXR1. The top of the FIFO registers share the same addresses as of the data registers in nonFIFO mode. Each of the FIFO data registers pair has a separate set of control registers. Table 5–1 lists the register details for easy reference.

Table 5–1. McBSP FIFO Registers

Address Offset 0x000 xxh	16-Bit Access	Type R/W †	Reset Value (Hex)	McBSP FIFO Registers
				FIFO mode registers applicable only for FIFO mode in 28xx implementation
00	DRR2	R	0x0000	McBSP Data Receive Register2 –Top of receive FIFO Read first FIFO index will not advance
01	DRR1	R	0x0000	McBSP Data Receive Register1–Top of receive FIFO Read second for FIFO index to advance
02	DXR2	W	0x0000	McBSP Data Transmit Register2–Top of transmit FIFO Write first FIFO index will not advance
03	DXR1	W	0x0000	McBSP Data Transmit Register1–Top of transmit FIFO Write second for FIFO index to advance
20	MFFTX	R/W	0xA000	McBSP transmit FIFO register
21	MFFRX	R/W	0x201F	McBSP receive FIFO register
22	MFFCT	R/W	0x0000	McBSP FIFO control register
23	MFFINT	R/W	0x0000	McBSP FIFO interrupt register
24	MFFST	R/W	0x0000	McBSP FIFO status register

† R = read, W = write

This section describes the following requirements for the McBSP module in FIFO mode.

1) McBSP functionality and limitation under FIFO mode
2) McBSP FIFO operation
3) McBSP receive interrupt generation
4) McBSP transmit interrupt generation
5) McBSP FIFO access constraints
6) McBSP FIFO register description

5.2 McBSP Functionality and Limitation Under FIFO Mode

McBSP, in its normal mode, communicates with various types of Codecs with variable word size. This mode is common and is available while the FIFO is enabled. Apart from this mode, the McBSP uses time-division multiplexed (TDM) data stream while communicating with other McBSPs or serial devices. The multichannel mode provides flexibility while transmitting/receiving selected channels or all the channels in a TDM stream.

Table 5–2 provides a quick reference to McBSP mode selection and its limitation if any in FIFO mode.

Table 5–2. McBSP Mode Selection

No.	McBSP Word Size	Register Bits That Decide the Mode Selection					Description of the Mode and its Function
		MCR1 bit 9,0		MCR2 bit 9,1,0		SPCR1 –6	
		RMCME	RMCM	XMCME	XMCM	A-bis	
							Normal Mode
1	8/12/16/24/32 bit words	0	0	0	00	0	All types of Codec interface will use this selection
							Multichannel Mode
2	8-bit words						**2 Partition** or 32-channel Mode
		0	1	0	01	0	All channels are disabled,unless selected in X/RCERA/B
		0	1	0	10	0	All channels are enabled,but masked unless selected in X/RCERA/B
		0	1	0	11	0	Symmetric transmit, receive
							8 Partition or 128 Channel Mode Transmit/ Receive
							Channels selected by X/RCERA to X/RCERH bits
							Multichannel Mode is ON
		1	1	1	01	0	All channels are disabled,unless selected in XCERs
		1	1	1	10	0	All channels are enabled,but masked unless selected in XCERs
		1	1	1	11	0	Symmetric transmit, receive
							Continuous Mode – Transmit
		1	0	1	00	0	**Multi-Channel Mode is OFF**
							All 128 channels are active and enabled
							A-bis Mode
4	16-bit words	0	0	0	0	1	A-bis mode. McBSP can receive /transmit up to 1024 bits on PCM link. Bit selection is provide by RCERA/B , XCERA/B registers only
							X/RCERC–H are not used

Note: x - Don’t care. However, design needs to validate A-bis mode selection and bit definitions in this table.

5.3 McBSP FIFO Operation

The McBSP powers up without the FIFOs enabled. The FIFO mode of the McBSP can be enabled using the FIFO-enable bits in FIFO register MFFTX. The following steps explain the FIFO features and help programming the McBSP with FIFOs.

1) Reset. At reset the McBSP powers up in standard McBSP mode with the FIFOs features disabled. The FIFO registers MFFTX, MFFRX, and MFFCT will remain inactive.

2) NonFIFO mode. The McBSP without the FIFO registers is enabled. McBSP registers can be programmed to receive and transmit data through DRR2/DRR1 and DXR2/DXR1 registers, respectively. The CPU can directly access these registers to move data from memory to these registers. Interrupt signals will be based on these register pair contents and its related flags.

3) FIFO mode. FIFOs are enabled by setting the MFFENA bit to 1 in the MFFTX register.

4) Active registers. All the McBSP registers and McBSP FIFO registers MFFTX, MFFRX (once FIFOs are enabled), and MFFCT will be active.

5) Interrupts. MRINT/MXINT will generate CPU interrupts for receive and transmit conditions of the FIFO. See the interrupt chapter for details. See the table interrupt section for details on these bit selection and interrupt functionality.

6) Buffers. Transmit and receive data registers of the McBSP will be supplemented with four 16x16 FIFOs. The receive data registers DRR2 and DRR1 will have a 16x16bit FIFO each. Receive channel buffer from DR pin will stack as shown below.

   ```
   DR pin – RSR[1,2] – RBR[1,2] – DRR[1,2]+16x16 FIFO[1,2].
   ```

   Similarly the transmit data register DXR2 and DXR1 will have a 16x16bit FIFO each. Transmit channel buffer up to DX pin will stack as shown below.

   ```
   DX pin – XSR[1,2] –  DXR[1,2]+  16x16 FIFO[1,2]
   ```

7) Transmit FIFO to DXR2/DXR1 data transfer will be based on XEVT/XINT interrupt signal from McBSP and its related flags.

   DRR2/DRR1 to Receive FIFO data register will be based on REVT/RINT interrupt signal from McBSP and its related flags.

8) FIFO status bits. Both transmit and receive FIFOs have status bits TXFFST (bit 12–0) or RXFFST (bits 12–0). They define the number of words available in the FIFOs at any time. The status bits are updated on every 2 word transfer, DXR2/DXR1 and DRR2/DRR1 register, respectively. The transmit FIFO reset bit TXFIFO and receive reset bit RXFIFO will reset the FIFO pointers to zero when these bits are set 1. The FIFOs will resume operation from start once these bits are cleared to zero.

9) Programmable interrupt levels. Both transmit and receive FIFO can generate CPU interrupts. The interrupt trigger is generated whenever the transmit FIFO status bits TXFFST (bits 12–8) match the interrupt trigger level bits TXFFIL (bits 4–0). This provides a programmable interrupt trigger for transmit and receive sections of the McBSP. Default value for these trigger level bits will be 0x11111 for receive FIFO and 0x00000 for transmit FIFO respectively.

5.4 McBSP Receive Interrupt Generation

In the McBSP module, data receive and error conditions generate two sets of interrupt signals. One set is used for the CPU and the other set is for DMA. Since there is no DMA implemented with this module, the DMA interrupt signals are used by the FIFO control logic. The FIFO control logic references these signal to move the data between the FIFO registers and the actual receive registers. The following section maps the interrupt signal selection in FIFO mode and nonFIFO mode for the receive channel.

Figure 5–1. Receive Interrupt Generation

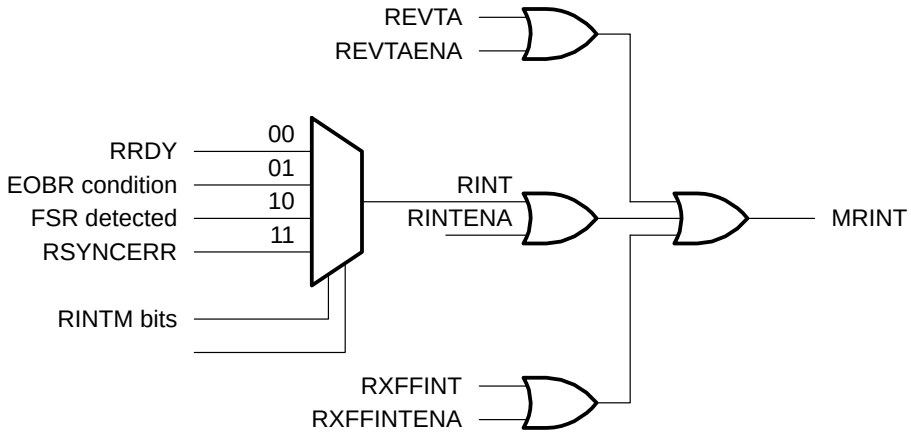


Table 5–3. Receive Interrupt Sources and Signals in NonFIFO Mode and FIFO Mode

Modes	McBSP Interrupt Signal	Interrupt Flags	Interrupt Enables in SPCR1	Interrupt Enables	Type of Interrupt	Interrupt Line
NonFIFO Mode			**RINTM Bits**			
Receive	RINT	RRDY	00	RINTENA	Every word receive	MRINT
		EOBR	01	RINTENA	Every 16 channel block boundary	
		FSR	10	RINTENA	On every FSR	
		RSYNCERR	11	RINTENA	Frame sync error	
	REVTA		xx	REVTA ENA	A-bis mode update for CPU	
FIFO Mode						
Receive FIFO Interrupt		RXFFINT or RXFFOVF	xx	RXFFINT ENA	Transmit FIFO status interrupt	MRINT
	REVT				Not Used by CPU and FIFO	

Note: x – Don't care

Note:

Since X/RINT, X/REVTA, and X/RXFFINT share the same CPU interrupt, it is recommended that all applications use one of the above selections for interrupt generation. If multiple interrupt enables are selected at the same time, there is a likelihood of interrupts being masked or not recognized.

5.5 McBSP Transmit Interrupt Generation

McBSP module data transmit and error conditions generate two sets of interrupt signals. One set is used for the CPU and the other set is for DMA. Since there is no DMA implemented with this module, the DMA interrupt signals are used by the FIFO control logic. The FIFO control logic will reference these signal to move the data between the FIFO registers and the actual receive registers. The following section maps the interrupt signal selection in FIFO mode and nonFIFO mode for the transmit channel.

Figure 5–2. Transmit Interrupt Generation

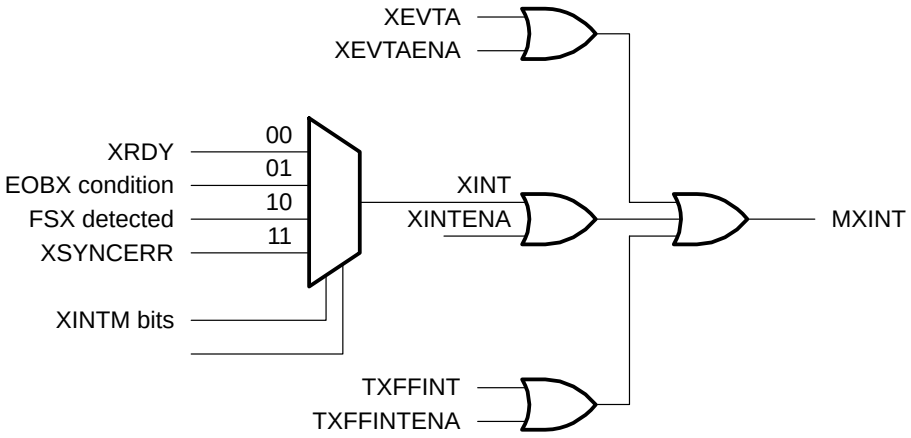


Table 5–4. Transmit Interrupt Sources and Signals in NonFIFO Mode and FIFO Mode

Modes	McBSP Interrupt Signal	Interrupt Flags	Interrupt Enables in SPCR1	Interrupt Enables	Type of Interrupt	Interrupt Line
NonFIFO Mode			**XINTM Bits**			
Transmit	XINT	XRDY	00	XINTENA	Every word receive	MXINT
		EOBX	01	XINTENA	Every 16-channel block boundary	
		FSX	10	XINTENA	On every FSX	
		XSYNCERR	11	XINTENA	Frame sync error	
	XEVTA		XX	XEVTA ENA	A-bis mode update for CPU	
FIFO Mode						
Transmit FIFO Interrupt		TXFFINT	XX	TXFFINT ENA	Transmit FIFO status interrupt	MXINT
	XEVT				Not used by CPU	

Note: x – Don't care.

5.5.1 FIFO Data Register Access Constraints

McBSP registers are allowed only 16-bit access via the peripheral bus. This includes the pair of data registers on receive and transmit channels. DRR2/DRR1is the receive register pair and DXR2/DXR1is the transmit register pair. Based on the word size selected (8/12/16/24/32 bit) on the receive side either the DRR2/DRR1 register pair is used or only the DRR1 is used. Similarly, on the transmit side either the DXR2/DXR1register pair is used or only the DXR1 is used. This is true in nonFIFO mode and FIFO mode of the McBSP module. The following tables explain the read/write order based on the word size. In the FIFO mode, there is only one set of FIFO pointer for both receive and transmit register pair. The FIFO pointer update is valid only if the listed order is maintained. This will guarantee the same data integrity as applicable in the nonFIFO mode of the McBSP.

Table 5–5. Receive FIFO Read Order

	Normal/ Multichannel Mode	Receive FIFO Pointer	Normal Mode	Receive FIFO Pointer
	8/16 Bit Word Read Order		24/32 Bit Word Read Order	
Case1	DRR2	No change	DRR2	No change
	DRR1	Decrement	DRR1	Decrement
Case2	DRR2	No change	DRR2	No change
	DRR2, or n times	no change	DRR2 or n times	no change
	DRR1	Decrement	DRR1	Decrement
Case3	DRR1	Decrement	DRR1	no change
	DRR2	no change	DRR2	no change

Table 5–6. Transmit FIFO Write Order

	Normal/ Multichannel Mode	Receive FIFO Pointer	Normal Mode	Receive FIFO Pointer
	8/16 Bit Word Write Order		24/32 Bit Word Write Order	
Case1	DXR2	No change	DXR2	No change
	DXR1	Increment	DXR1	Increment
Case2	DXR2	No change	DXR2	No change
	DXR2, or n times	No change	DXR2 or n times	No change
	DXR1	Increment	DXR1	Increment
Case3	DXR1	Increment	DXR1	No change
	DXR2	No change	DXR2	No change

Note: DRR2/DRR1 and DXR2/DXR1 registers read and write constraints are native to the McBSP module. Hence maintaining these constraints in FIFO mode will leave no additional conflicts.

5.5.2 FIFO Error Flags

The McBSP has several error flags both on receive and transmit channel. In FIFO mode these bits either made non functional or equivalent flags are provided. Table 5–7 explains the error flags and their new meaning in FIFO mode

Table 5–7. McBSP Error Flags

Error Flags	Function in McBSP, NonFIFO Mode	Function in McBSP, FIFO Mode
RFULL	Indicates DRR2/DRR1 are not read and RXR register is overwritten	This bit will never get set, as the DRR2/DRR1 values are read into the FIFO registers. Use RXFFOVF instead for error condition.
RXFFOVF	Not applicable in nonFIFO mode	FIFO receive overflow flag that will be set whenever the receive FIFO is overflows. The data at the top (or the FIRST IN data) of the FIFO will be lost.
RSYNCERR	Indicates unexpected frame-sync condition, current data reception will abort and restart. Use RINTM bit 11 for interrupt generation on this condition.	Indicates unexpected frame-sync condition, current data reception will abort and restart Use RINTM bits 11 for interrupt generation on this conditions.
XSYNCERR	Indicates unexpected frame-sync condition, current data transmission will abort and restart. Use XINTM bit 11 for interrupt generation on this condition.	Indicates unexpected frame-sync condition, current data transmission will abort and restart. Use RINTM bits 11 for interrupt generation on this condition.

5.5.3 McBSP IDLE Mode

Internal IDLE mode of the McBSP module is not to be implemented. Bit 14 in PCR register for IDLE_EN will be a read/write bit.

5.5.4 McBSP Reset Conditions

The McBSP can be reset either through hardware DSP reset or reset control on receive or transmit channel. The receive channel and transmit channel resets are controlled by RRST (bit 0 in SPCR1) and XRST (bit 0 in SPCR2) bits The McBSP module native reset state of the McBSP pins, register status bits will remain as is in both FIFO nonFIFO modes.

However, the FIFO registers have additional register bits that will control of the reset state of the FIFO pointers and register contents. RXFIFO and TXFIFO reset bits in MFFRX (bit 13) and MFFTX (bit 13) reset the FIFO pointers to zero and power up in reset state.

5.6 McBSP FIFO Register Descriptions

Figure 5–3. McBSP FIFO Transmit Register (MFFTX)

15	14	13	12	11	10	9	8
Reserved	MFFENA	TXFIFO Reset	TXFFST4	TXFFST3	TXFFST2	TXFFST1	TXFFST0
R-0	R/W-0	R/W-1	R-0	R-0	R-0	R-0	R-0

7	6	5	4	3	2	1	0
TXFFINT Flag	TXFFINT Clear	TXFFIENA	TXFFIL4	TXFFIL3	TXFFIL2	TXFFIL1	TXFFIL0
R-0	W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

Legend: R = Read only; W = Write only; R/W = Read/Write; -n = Reset Value

Bit(s)	Name	Description	
15	Reserved	Reserved	
14	MFFENA	0	McBSP FIFO enhancements are disabled and FIFO is in reset.
		1	McBSP FIFO enhancements are enabled.
13	TXFIFO Reset	0	Write 0 to reset the FIFO pointer to zero, and hold in reset.
		1	Re-enable Transmit FIFO operation
12–8	TXFFST4–0	00000	Transmit FIFO is empty.
		00001	Transmit FIFO has 1 word
		00010	Transmit FIFO has 2 words
		00011	Transmit FIFO has 3 words
		0xxxx	Transmit FIFO has x words
		10000	Transmit FIFO has 16 words
7	TXFFINT	0	TXFIFO interrupt has not occurred. Read only bit
		1	TXFIFO interrupt has occurred. Read only bit
6	TXFFINT CLR	0	Write 0 has no effect on TXFIFINT flag bit, bit reads back a zero
		1	Write 1 to clear TXFFINT flag in bit 7
5	TXFFIENA	0	TX FIFO interrupt based on TXFFIVL match (less than or equal to) will be disabled
		1	TX FIFO interrupt based on TXFFIVL match (less than or equal to) will be enabled.
4–0	TXFFIL4–0	TXFFIL4–0 Transmit FIFO interrupt level bits	
		Transmit FIFO will generate interrupt when the FIFO status bits (TXFFST4–0) and FIFO level bits (TXFFIL4–0) match (less than or equal to).	
		Default value should be 0x00000	

Note: Bit 15 used to be MRST bit and will be reserved in this FIFO implementation

Figure 5–4. McBSP FIFO Receive Register (MFFRX)

15	14	13	12	11	10	9	8
RXFFOVF Flag	RXFFOVF Clear	RXFIFO Reset	RXFFST4	RXFFST3	RXFFST2	RXFFST1	RXFFST0
R-0	W-0	R/W-1	R-0	R-0	R-0	R-0	R-0

7	6	5	4	3	2	1	0
RXFFINT Flag	RXFFINT Clear	RXFFIENA	RXFFIL4	RXFFIL3	RXFFIL2	RXFFIL1	RXFFIL0
R-0	W-0	R/W-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1

Legend: R = Read only; W = Write only; R/W = Read/Write; -n = Reset Value

Bit(s)	Name		Description
15	RXFFOVF	0	Receive FIFO has not overflowed, read only bit.
		1	Receive FIFO has overflowed, read only bit. More than 16 words have been received in to the FIFO, and the first received word is lost.
14	RXFFOVFCLR	0	Write 0 has no effect on RXFFOVF flag bit. Bit reads back a zero
		1	Write 1 to clear RXFFOVF flag in bit 15
13	RXFIFO Reset	0	Write 0 to reset the FIFO pointer to zero, and hold in reset.
		1	Re-enable Transmit FIFO operation
12–8	RXFFST4–0	00000	Receive FIFO is empty.
		00001	Receive FIFO has 1 word
		00010	Receive FIFO has 2 words
		00011	Receive FIFO has 3 words
		0xxxx	Receive FIFO has x words
		10000	Receive FIFO has 16 words.
7	RXFFINT	0	RXFIFO interrupt has not occurred. Read only bit
		1	RXFIFO interrupt has occurred. Read only bit
6	RXFFINT CLR	0	Write 0 has no effect on RXFIFINT flag bit, Bit reads back a zero
		1	Write 1 to clear RXFFINT flag in bit 7
5	RXFFIENA	0	RX FIFO interrupt based on RXFFIVL match (less than or equal to) will be disabled
		1	RX FIFO interrupt based on RXFFIVL match (less than or equal to) will be enabled.
4–0	RXFFIL0 –4	RXFFIL4–0 Receive FIFO interrupt level bits	
		Receive FIFO will generate interrupt when the FIFO status bits (RXFFST4–0) and FIFO level bits (RXFFIL4–0) match (greater than or equal to) default value of these bits after reset – 11111. This will avoid frequent interrupts, after reset, as the receive FIFO will be empty most of the time.	

Figure 5–5. McBSP FIFO Control Register (MFFCT)

15	14	8
IACKM	Reserved	
R/W-0	R-0	

7	6	5	4	3	2	1	0
FFTXDLY7	FFTXDLY6	FFTXDLY5	FFTXDLY4	FFTXDLY3	FFTXDLY2	FFTXDLY1	FFTXDLY0
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 = Read only; R/W = Read/Write

Bit(s)	Name	Reset	Description
15	IACKM	0	Default value to be written
		1	Reserved function. Do not write 1 to this bit.
14–8		0	Reserved
7–0	FFTXDLY7–0	0x0000	FFTXDLY7–0 are used only in SPI mode. In McBSP mode they are “don’t care”. These bits define the delay between every transfer from FIFO transmit FIFO to transmit (DXR2/DXR1) register. The delay is defined in number CLKX serial clock or baud clock cycles. The 8 bit register could define a minimum delay of 0 serial clock cycles and a maximum of 256 serial clock cycles. In the FIFO mode the transfer from FIFO to (DXR2/DXR1) registers should occur only after the shift register has completed shifting of the last bit. This ensures that the delay count is between two words in the data stream. In McBSP/SPI mode with FIFO and delay enabled, the McBSP registers DXR2/DXR1 are not to be treated as one additional level of buffer.

Figure 5–6. McBSP FIFO Interrupt Register (MFFINT)

15	8
Reserved	
R-0	

7	4	3	2	1	0
Reserved		REVTA ENA	RINT ENA	XEVTA ENA	XINT ENA
R-0		R/W-0	R/W-0	R/W-0	R/W-0

Legend: R = Read only; R/W = Read/Write

Bit(s)	Name	Reset	Description
15–4	Reserved		Reserved

Figure 5–6.McBSP FIFO Interrupt Register (MFFINT) (Continued)

Bit(s)	Name	Reset	Description
3	REVTA ENA		Enable for A-bis Receive Interrupt on every 16 CLKX/XLKR cycles. Enabled only in A-bis FIFO mode
		0	A-bis Receive interrupt is disabled
		1	A-bis Receive interrupt is enabled
2	RINT ENA		Enable for Receive Interrupt Enabled only in nonFIFO mode
		0	Receive interrupt on XRDY is disabled
		1	Receive interrupt on XRDY is enabled
			This interrupt will be active for any of the conditions selected for RINTM bits
1	XEVTA ENA		Enable for A-bis Transmit Interrupt on every 16 CLKX/XLKR cycles. Enabled only in A-bis FIFO mode
		0	A-bis transmit interrupt is disabled
		1	A-bis transmit interrupt is enabled
0	XINT ENA		Enable for transmit Interrupt Enabled only in nonFIFO mode
		0	Transmit interrupt on XRDY is disabled
		1	Transmit interrupt on XRDY is enabled
			This interrupt will be active for any of the conditions selected for XINTM bits

Figure 5–7. McBSP FIFO Status Register (MFFST)

15	8
Reserved	
R-0	

7	4	3	2	1	0
Reserved		FSR Flag	EOBR Flag	FSX Flag	EOBX Flag
R-0		R/W-x	R/W-0	R/W-x	R/W-0

Legend: R = Read only; R/W = Read/Write

Bit(s)	Name	Description
15–4	Reserved	Reserved
3	FSR Flag	New frame-sync FSR pulse detection flag . This flag will be set irrespective of whether RINTM bits are 10. Can be used to detect FSR pulse without interrupt. At reset, this bit will be updated based on the FSX/FSR pin status. If left unconnected, they will be set to 1 due to the internal pullups.
		0 FSR Frame sync pulse not detected
		1 FSR Frame sync pulse detected
		Write 0 to clear

Figure 5–7.McBSP FIFO Status Register (MFFST) (Continued)

Bit(s)	Name	Description
2	EOBR Flag	End of receive block in multichannel mode . This flag will be set irrespective of whether RINTM bits are 01. Can be used to detect EOB without interrupt. 0 EOBX End of block condition(EOB) not occurred. Write 0 to clear. 1 EOBX End of block condition occurred
1	FSX Flag	New Frame sync FSX pulse detection flag. This flag will be set irrespective of whether XINTM bits are 10. Can be used to detect FSX without interrupt. 0 FSX Frame sync pulse not detected 1 FSX Frame sync pulse detected Write 0 to clear
0	EOBX Flag	End of transmit block in multichannel mode. This flag will be set irrespective of XINTM bits are 01. Can be used detect EOB condition without interrupt. 0 EOBX End of block condition (EOB) not occurred 1 EOBX End of block condition occurred Write 0 to clear

Chapter 6

McBSP Registers

This chapter includes the register and bit descriptions.

Topic **Page**

6.1	Data Receive and Transmit Registers	6-2
6.2	Serial Port Control Registers (SPCR1 and SPCR2)	6-4
6.3	Receive Control Registers (RCR1 and RCR2)	6-8
6.4	Transmit Control Registers (XCR1 and CXR2)	6-11
6.5	Sample Rate Generator Registers (SRGR1 and SRGR2)	6-14
6.6	Multichannel Control Registers (MCR1 and MCR2)	6-17
6.7	Pin Control Register (PCR)	6-21
6.8	Receive Channel Enable Registers (RCERA – RCERH)	6-24
6.9	Transmit Channel Enable Registers (XERA – XCERH)	6-29
6.10	Register Bit Summary	6-34

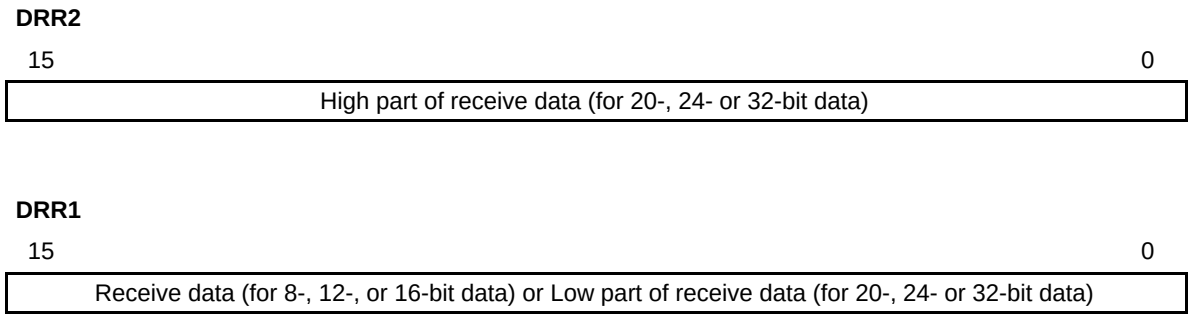
6.1 Data Receive and Transmit Registers

This section includes the Data Receive Registers (DRR2 and DRR1) and the Data Transmit Registers (DXR2 and DXR1).

6.1.1 Data Receive Registers (DRR2 and DRR1)

The CPU or the FIFO controller reads received data from one or both of the data receive registers. If the serial word length is 16 bits or smaller only DRR1 is used. If the serial length is larger than 16 bits, both DRR1 and DRR2 are used, and DRR2 holds the most significant bits. Each frame of receive data in the McBSP can have one phase or two phases, each with its own serial word length.

Figure 6–1. Data Receive Registers (DRR2 and DRR1)



6.1.2 How Data Travels From the Data Receive (DR) Pin to the DRRs

If the serial word length is 16 bits or smaller, receive data on the DR pin is shifted into receive shift register 1 (RSR1) and then copied into receive buffer register 1 (RBR1). The content of RBR1 is then copied to DRR1, which can be read by the CPU or by the FIFO.

If the serial word length is larger than 16 bits, receive data on the DR pin is shifted into both of the receive shift registers (RSR2, RSR1) and then copied into both of the receive buffer registers (RBR2, RBR1). The content of the RBRs is then copied into both of the DRRs, which can be read by the CPU or by the FIFO.

If companding is used during the copy from RBR1 to DRR1 (RCOMPAND = 10b or 11b), the 8-bit compressed data in RBR1 is expanded to a left-justified 16-bit value in DRR1. If companding is disabled, the data copied from RBR[1,2] to DRR[1,2] is justified and bit filled according to the RJUST bits.

6.1.3 Data Transmit Registers (DXR2 and DXR1)

For transmission, the CPU or the FIFO writes data to one or both of the data transmit registers. If the serial word length is 16 bits or smaller, only DXR1 is used. If the word length is larger than 16 bits, both DXR1 and DXR2 are used, and DXR2 holds the most significant bits. Each frame of transmit data in the McBSP can have one phase or two phases, each with its own serial word length.

Figure 6–2. Data Transmit Registers (DXR2 and DXR1)

DXR2

15	0
High part of transmit data (for 20-, 24- or 32-bit data)	

DXR1

15	0
Transmit data (for 8-, 12-, or 16-bit data) or Low part of transmit data (for 20-, 24- or 32-bit data)	

6.1.4 How Data Travels From the DXRs to the Data Transmit (DX) Pin

If the serial word length is 16 bits or fewer, data written to DXR1 is copied to transmit shift register 1 (XSR1). From XSR1, the data is shifted onto the DX pin one bit at a time.

If the serial word length is more than 16 bits, data written to DXR1 and DXR2 is copied to both transmit shift registers (XSR2, XSR1). From the XSRs, the data is shifted onto the DX pin one bit at a time.

If companding is used during the transfer from DXR1 to XSR1 (XCOMPAND = 10b or 11b), the McBSP compresses the 16-bit data in DXR1 to 8-bit data in the μ-law or A-law format in XSR1. If companding is disabled, the McBSP passes data from the DXR(s) to the XSR(s) without modification.

The XSRs are not accessible.

6.2 Serial Port Control Registers (SPCR1 and SPCR2)

Each McBSP has two serial port control registers of the form shown in Figure 6–3. and describe the bits in SPCR1 and SPCR2, respectively. These memory-mapped registers enable you to:

- Control various McBSP modes: digital loopback mode (DLB), sign-extension and justification mode for reception (RJUST), clock stop mode (CLKSTP), A-bis mode (ABIS), interrupt modes (RINTM and XINTM), emulation mode (FREE and SOFT)
- Turn on and off the DX-pin delay enabler (DXENA)
- Check the status of receive and transmit operations (RSYNCERR, XSYNCERR, RFULL, $\overline{\text{XEMPTY}}$, RRDY, XRDY)
- Reset portions of the McBSP ($\overline{\text{RRST}}$, $\overline{\text{XRST}}$, $\overline{\text{FRST}}$, $\overline{\text{GRST}}$)

Figure 6–3. Serial Port Control 2 Register (SPCR2)

15						10	9	8
Reserved							FREE	SOFT
R-0							R/W–0	R/W–0

7	6	5	4	3	2	1	0
FRST	GRST	XINTM		XSVNCERR	XEMPTY	XRDY	XRST
R/W-0	R/W–0	R/W–0		R/W–0	R–0	R–0	R/W–0

Note: R = Read access, W = Write access, R/W – Read/write access

Bit(s)	Name	Description
15–10	Reserved	Reserved
9	FREE	Free running mode (FREE and SOFT are taking effect when EMUSUSPEND is set) 0 Free running mode is disabled 1 Free running mode is enabled
8	SOFT	Soft bit (FREE and SOFT is taking effect when EMUSUSPEND is set) 0 SOFT mode is disabled 1 SOFT mode is enabled
7	$\overline{\text{FRST}}$	Frame sync generator reset 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 i.e. all frame counters are loaded with their programmed values.

Figure 6–3.Serial Port Control 2 Register (SPCR2) (Continued)

Bit(s)	Name	Description
6	$\overline{\text{GRST}}$	Sample rate generator reset
		0 Sample rate generator is reset.
		1 Sample rate generator is pulled out of reset. CLKG is driven as per programmed value in Sample Rate Generator Registers (SRGR).
5–4	XINTM	Transmit Interrupt mode
		00b (X)INT driven by (X)RDY (i.e. end of word) and end of frame in A-bis mode.
		01b (X)INT generated by end-of-block or end-of-frame in multi-channel operation
		10b (X)INT generated by a new frame synchronization
		11b (X)INTM=11b, (X)INT generated by (X)SYNCERR
3	XSYNCERR	Transmit synchronization error
		0 No synchronization error
		1 Synchronization error detected by McBSP.
2	$\overline{\text{XEMPTY}}$	Transmit shift register (XSR) empty
		0 XSR is empty
		1 XSR is not empty XSREMPTY bit
1	XRDY	Transmitter ready
		0 Transmitter is not ready.
		1 Transmitter is ready with data in DXR.
0	$\overline{\text{XRST}}$	Transmitter reset. This resets and enables the transmitter.
		0 The serial port transmitter is disabled and in reset state.
		1 The serial port transmitter is enabled.

Figure 6–4. SPCR1 Register

15	14	13	12	11	10	8
DLB	RJUST		CLKSTP		Reserved	
R/W-0	R/W–0		R/W–0		R–0	

7	6	5	4	3	2	1	0
DXENA	ABIS	RINTM		RSYNCERR	RFULL	RRDY	RRST
R/W-0	R/W–0	R/W–0		R/W–0	R–0	R–0	R/W–0

Notes: R – Read-only access, R/W – Read/write access, –n – n is the value after a DSP reset

Bit(s)	Name	Description
15	DLB	Digital loop back mode
		0 Disabled
		1 Enabled
14–13	RJUST	Receive sign-Extension and justification mode
		0 0 Right-justify and zero-fill MSBs in DRR
		0 1 Right-justify and sign-extend MSBs in DRR
		1 0 Left-justify and zero-fill LSBs in DRR
		1 1 Reserved
12–11	CLKSTP	Clock stop mode. In SPI mode, this field is related to settings on CLKXP and CLKRP in the PCR register.
		0 0 Clock stop mode disabled. Normal clocking for non-SPI mode.
		CLKXP CLKRP
		0 0 0 0 In SPI mode: Clock starts with rising edge without delay
		0 1 1 0 In SPI mode: Clock starts with falling edge without delay
		1 0 0 1 In SPI mode: Clock starts with rising edge with delay
		1 1 1 1 In SPI mode: Clock starts with falling edge with delay
10–8	Reserved	

Figure 6–4.SPCR1 Register (Continued)

Bit(s)	Name	Description
7	DXENA	DX enabler. Enable extra delay for turn-on time. This bit controls the HI-z enable on the DX pin, not the data itself, so only the first bit will be delayed in the normal mode. In A-bis mode, any bit can be delayed since any bit can go from Hi-Z to valid 0 DX enabler is off 1 DX enabler is on
6	ABIS	ABIS mode 0 A-bis mode is disabled 1 A-bis mode is enabled
5–4	RINTM	Receive Interrupt mode 00b (R)INTM driven by (R)RDY (i.e. end of word) and end of frame in A-bis mode. 01b (R)INTM generated by end-of-block or end-of-frame in multi-channel operation 10b (R)INTM generated by a new frame synchronization 11b (R)INTM generated by (R)SYNCERR
3	RSYNCERR	Receive synchronization error 0 No synchronization error 1 Synchronization error detected by McBSP.
2	RFULL	Receive shift register (RSR) full 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 0 Receiver is not ready. 1 Receiver is ready with data to be read from DRR.
0	$\overline{\text{RRST}}$	Receiver reset. This resets and enables the receiver 0 The serial port receiver/transmitter is disabled and in reset state. 1 The serial port receiver/transmitter is enabled.

6.3 Receive Control Registers (RCR1 and RCR2)

Each McBSP has two receive control registers of the form shown in Figure 6–5. These memory-mapped registers enable you to:

- ❏ Specify one or two phases for each frame of receive data (RPHASE)
- ❏ Define two parameters for phase 1 and (if necessary) phase 2: the serial word length (RWDLEN1, RWDLEN2) and the number of words (RFRLEN1, RFRLEN2)
- ❏ Choose a receive companding mode, if any (RCOMPAND)
- ❏ Enable or disable the receive frame-sync ignore function (RFIG)
- ❏ Choose a receive data delay (RDATDLY)

Figure 6–5. Receive Control 2 Register (RCR2)

15	14	8
RPHASE	RFRLEN2	
R/W-0	R/W–0	

7	5	4	3	2	1	0
RWDLEN2		RCOMPAND		RFIG	RDATDLY	
R/W-0		R/W–0		R/W–0	RW–0	

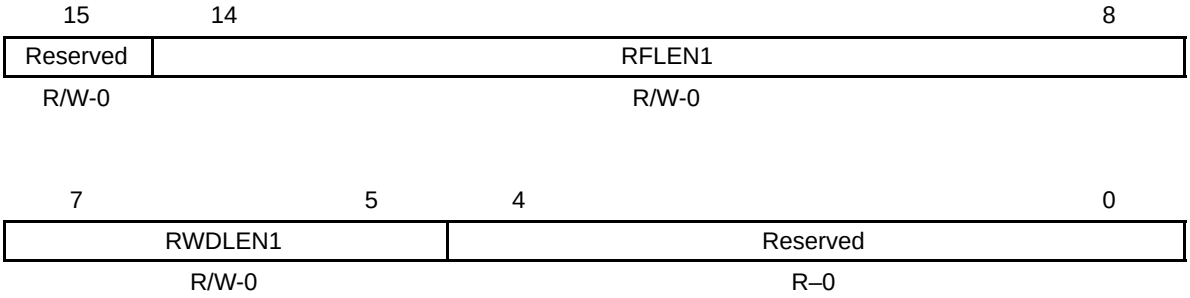
Notes: R – Read-only access, R/W – Read/write access, –n – n is the value after a DSP reset

Bit(s)	Name	Reset	Description
15	RPHASE	0	Receive phases
			0 Single phase frame
			1 Dual phase frame
14–8	RFRLEN2	0	Receive frame length 2
			00b 1 word per frame
			01b 2 words per frame
			|
			|
			11b 128 words per frame

Figure 6–5.Receive Control 2 Register (RCR2) (Continued)

Bit(s)	Name	Description
7–5	RWDLEN2	Receive word length 2
		000b 8 bits
		001b 12 bits
		010b 16 bits
		011b 20 bits
		100b 24 bits
		101b 32 bits
		11Xb Reserved TDM serial port control register (TSPC):TXM bit
4–3	RCOMPAND	Receive companding mode. Modes other than 00b are only enabled when the appropriate (R)WDLEN is 000b, indicating 8-bit data.
		00b No companding, data transfer starts with MSB first.
		01b No companding, 8-bit data, transfer starts with LSB first.
		10b Compand using μ-law for Receive data.
		11b Compand using A-law for Receive data.
2	RFIG	Receive frame ignore
		0 Receive Frame synchronization pulses after the first restarts the transfer.
		1 Receive Frame synchronization pulses after the first are ignored.
1–0	RDATDLY	Receive data delay
		00b 0-bit data delay
		01b 1-bit data delay
		10b 2-bit data delay
		11b Reserved

Figure 6–6. RCR1 Register



Notes: R – Read-only access, R/W – Read/write access, –n – n is the value after a DSP reset

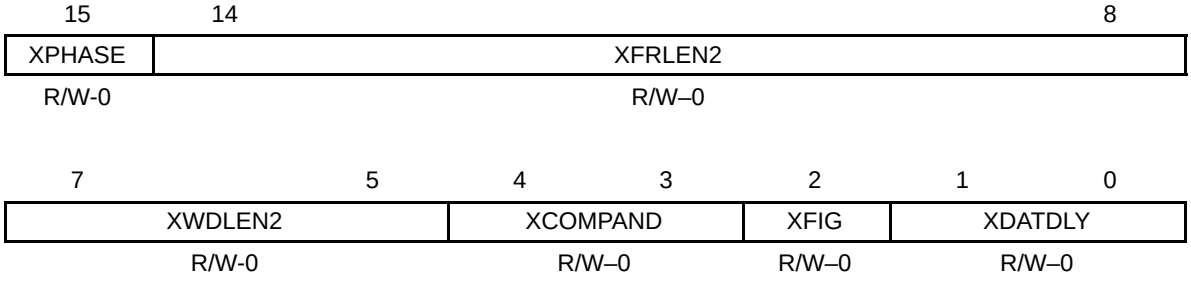
Bit(s)	Name	Description
15	Reserved	Reserved
		0 Valid to write a zero not a 1
		1 Reserved value.
14–8	RFRLEN1	Receive frame length 1
		00b 1 word per frame
		01b 2 words per frame
		|
		|
		11b 128 words per frame
7–5	RWDLEN1	Receive word length 1
		000b 8 bits
		001b 12 bits
		010b 16 bits
		011b 20 bits
		100b 24 bits
		101b 32 bits
		11Xb Reserved
4	Reserved	Reserved
		0 Valid to write a zero not a 1
		1 Reserved value.
3–0	Reserved	Reserved

6.4 Transmit Control Registers (XCR1 and XCR2)

Each McBSP has two transmit control registers of the form shown in Figure 6–7. These I/O-mapped registers enable you to:

- ❏ Specify one or two phases for each frame of transmit data (XPHASE)
- ❏ Define two parameters for phase 1 and (if necessary) phase 2: the serial word length (XWDLEN1, XWDLEN2) and the number of words (XFRLEN1, XFRLEN2)
- ❏ Choose a transmit companding mode, if any (XCOMPAND)
- ❏ Enable or disable the transmit frame-sync ignore function (XFIG)
- ❏ Choose a transmit data delay (XDATDLY)

Figure 6–7. Transmit Control 2 Register (XCR2)



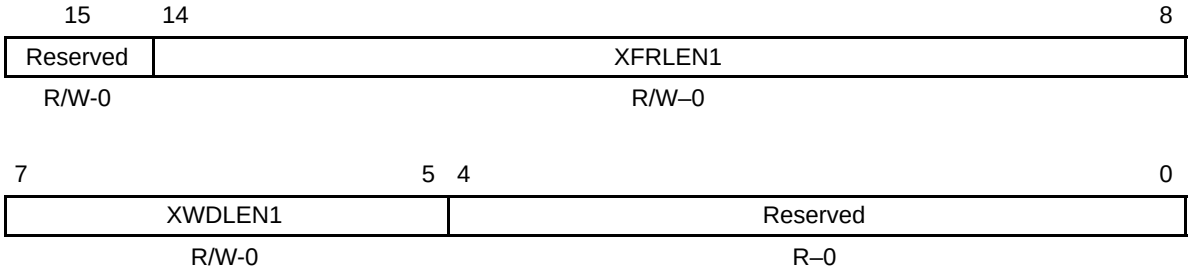
Notes: R – Read-only access, R/W – Read/write access, –X – X is the value after a DSP reset

Bit(s)	Name	Description
15	XPHASE	Transmit phase
		0 Single phase frame
		1 Dual phase frame
14–8	XRFRLEN2	Transmit frame length 2
		00b 1 word per frame
		01b 2 words per frame
		|
		|
		11b 128 words per frame

Figure 6–7.Transmit Control 2 Register (XCR2) (Continued)

Bit(s)	Name	Description
7–5	XWDLEN1	Transmit word length 2
		000b 8 bits
		001b 12 bits
		010b 16 bits
		011b 20 bits
		100b 24 bits
		101b 32 bits
		11Xb Reserved
4–2	XCOMPAND	Transmit companding mode. Modes other than 00b are only enabled when the appropriate XWDLEN is 000b, indicating 8-bit data.
		00b No companding, data transfer starts with MSB first.
		01b No companding, 8-bit data, transfer starts with LSB first.
		10b Compand using μ-law for Transmit data.
		11b Compand using A-law for Transmit data.
1	XFIG	Transmit frame ignore
		0 Transmit Frame synchronization pulses after the first restarts the transfer.
		1 Transmit Frame synchronization pulses after the first are ignored.
0	XDATDLY	Transmit data delay
		00b 0-bit data delay
		01b 1-bit data delay
		10b 2-bit data delay
		11b Reserved

Figure 6–8. XCR1 Register



Note: R = Read access, W = Write access, C = Clear only, –n = value after reset;

Bit(s)	Name	Description	
15	Reserved	Reserved	
		0	Valid to write a zero not a 1
		1	Reserved value.
14–8	XFRLEN1	Receive frame length 1	
		00b	1 word per frame
		01b	2 words per frame
			|
			|
		11b	128 words per frame
7–5	XWDLEN1	Transmit word length 1	
		000b	8 bits
		001b	12 bits
		010b	16 bits
		011b	20 bits
		100b	24 bits
		101b	32 bits
		11Xb	Reserved
4	Reserved	Reserved	
		0	Valid to write a zero not a 1
		1	Reserved value.
3–0	Reserved	Reserved	

6.5 Sample Rate Generator Registers (SRGR1 and SRGR2)

Each McBSP has two sample rate generator registers of the form shown in Figure 6–9. The sample rate generator can generate a clock signal (CLKG) and a frame-sync signal (FSG). The I/O-mapped registers SRGR1 and SRGR2 enable you to:

- Select the input clock source for the sample rate generator (CLKSM, in conjunction with the SCLKME bit of PCR)
- Divide down the frequency of CLKG (CLKGDV)
- Select whether internally-generated transmit frame-sync pulse are driven by FSG or by activity in the transmitter (FSGM).
- Specify the width of frame-sync pulses on FSG (FWID) and specify the period between those pulses (FPER)

When an external source (via the CLKR or CLKX pin) provides the input clock source for the sample rate generator:

- If the CLKX/CLKR pin is used, the polarity of the input clock is selected with CLKXP/CLKRP of PCR.
- The GSYNC bit of SRGR2 allows you to make CLKG synchronized to an external frame-sync signal on the FSR pin, so that CLKG is kept in phase with the input clock.

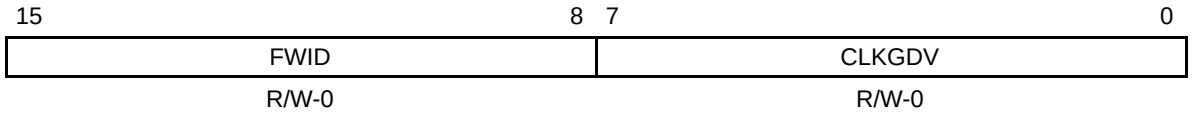
Figure 6–9. Sample Rate Generator 2 Register (SRGR2)

15	14	13	12	11	0
GSYNC	Res	CLKSM	FSGM	FPER	
R/W-0	R/W-1	R/W-1	R/W-0	R/W-0	

Note: R – Read-only access, R/W – Read/write access, –X – X is the value after a DSP reset

Bit(s)	Name	Description
15	GSYNC	Sample rate generator clock synchronization. Only used when the external clock (CLK) drives the sample rate generator clock (CLKSM=0). GSYNC = 0, 0 The sample rate generator clock (CLKG) is free running. 1 The sample rate generator clock (CLKG) is running. But CLKG is re-synchronize 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.
14	Reserved	Reserved
13	CLKSM	McBSP sample rate generator clock mode. Works with SCLKME bit in PCR register (bit 7) to decide the input clock to sample rate generator module SCLKME: CLKSM 0 0 Reserved 0 1 LSPCLK – Internal clock 1 0 External CLKR pin clock 1 1 External CLKX pin clock
12	FSGM	Sample rate generator Transmit frame synchronization mode. Used when FSXM = 1 in PCR. 0 Transmit frame sync signal (FSX) due to DXR(1/2)-to-XSR(1/2) copy. 1 Transmit frame sync signal driven by the sample rate generator frame sync signal, FSG. TDM serial port control register (TSPC):TXM bit
11–0	FPER	Frame period. This determines when the next frame sync signal should become active. Range: up to 2^{12} ; 1 to 4096 CLKG periods.

Figure 6–10. Sample Rate Generator 1 Register (SRGR1)



Note: R = Read access, W = Write access, C = Clear only, –n = value after reset;

Bit(s)	Name	Description
15–8	FWID	Frame width. Determines the width of the frame sync pulse, FSG, during its active period. Range: up to 2^8 ; 1 to 256 CLKG periods.
7–0	CLKGDV	Sample rate generator clock divider. This value is used as the divide-down number to generate the required sample rate generator clock frequency. Default value is 1.

6.6 Multichannel Control Registers (MCR1 and MCR2)

Each McBSP has two multichannel control registers of the form shown in Figure 6–11. MCR1 has control and status bits (with an R prefix) for multichannel selection operation in the receiver. MCR2 contains the same type of bits (bit with an X prefix) for the transmitter. These I/O-mapped registers enable 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 6–11. Multichannel Control 2 Register (MCR2)

15	10	9	8
Reserved		XMCME	XPBBLK
R–0		R/W-0	R/W-0

7	6	5	4	2	1	0
XPBBLK	XPABLK		XCBLK		XMCM	
R/W-0	R/W-0		R–0		R/W-0	

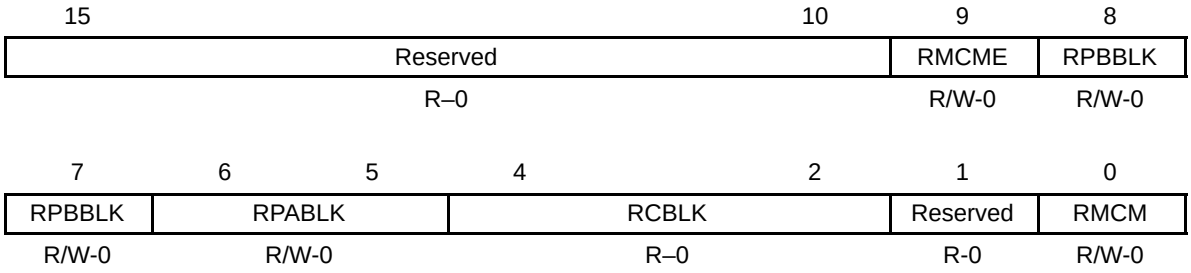
Note: R = Read access, W = Write access, C = Clear only, –n = value after reset

Bit(s)	Name	Description
15–10	Reserved	Reserved
9	XMCME	Enhanced transmit multichannel selection enable (XMCME) XMCME operates in conjunction with RMCME. The RMCME and XMCME bit values need to be the same. ❑ RMCME = 0 and XMCME = 0: (normal multichannel selection mode) (Default value) Maximum 32 channels can be enabled at one time. ❑ RMCME = 1 and XMCME = 1: (128 channel selection mode) Maximum 128 channels can be enabled at one time. All other modes reserved.

Figure 6–11. Multichannel Control 2 Register (MCR2) (Continued)

Bit(s)	Name	Description
8–7	XPBBLK	Receive/transmit partition B block
		00b Block 1. Channel 16 to channel 31
		01b Block 3. Channel 48 to channel 63
		10b Block 5. Channel 80 to channel 95
		11b Block 7. Channel 112 to channel 127
6–5	XPABLK	Receive/transmit partition A block
		00b Block 0. Channel 0 to channel 15
		01b Block 2. Channel 32 to channel 47
		10b Block 4. Channel 64 to channel 79
		11b Block 6. Channel 96 to channel 111
4–2	XCBLK	Receive/transmit current block
		000b Block 0. Channel 0 to channel 15
		001b Block 1. Channel 16 to channel 31
		010b Block 2. Channel 32 to channel 47
		011b Block 3. Channel 48 to channel 63
		100b Block 4. Channel 64 to channel 79
		101b Block 5. Channel 80 to channel 95
		110b Block 6. Channel 96 to channel 111
		111b Block 7. Channel 112 to channel 127
1–0	XMCM	Transmit multichannel selection enable
		00b All channels enabled without masking (DX is always driven during transmission of data).
		01b All channels disabled and therefore masked by default. Required channels are selected by enabling XP(A/B)BLK and XCER(A/B) appropriately. Also, these selected channels are not masked and therefore DX is always driven.
		10b All channels enabled, but masked. Selected channels enabled via XP(A/B)BLK and XCER(A/B) are unmasked.
		11b All channels disabled and therefore masked by default. Required channels are selected by enabling RP(A/B)BLK and RCER(A/B) appropriately. Selected channels can be unmasked by RP(A/B)BLK and XCER(A/B). This mode is used for symmetric transmit and receive operation. BSP serial port control extension register (SPCE):HALTR bit

Figure 6–12. Multichannel Control 1 Register (MCR1)



Note: R = Read access, W = Write access, C = Clear only, –n = value after reset;

Bit(s)	Name	Description
15–10	Reserved	Reserved
9	RMCME	Enhanced receive multichannel selection enable (RMCME) RMCME operates in conjunction with XMCME. The RMCME and XMCME bit values need to be the same. ❑ RMCME = 0 and XMCME = 0: (normal multichannel selection mode) (Default value) Maximum 32 channels can be enabled at one time. ❑ RMCME = 1 and XMCME = 1: (128 channel selection mode) Maximum 128 channels can be enabled at one time. All other modes reserved.
8–7	RPBBLK	Receive/transmit partition b block 00b Block 1. Channel 16 to channel 31 01b Block 3. Channel 48 to channel 63 10b Block 5. Channel 80 to channel 95 11b Block 7. Channel 112 to channel 127
6–5	RPABLK	Receive/transmit partition a block 00b Block 0. Channel 0 to channel 15 01b Block 2. Channel 32 to channel 47 10b Block 4. Channel 64 to channel 79 11b Block 6. Channel 96 to channel 111

Figure 6–12. Multichannel Control 1 (MCR1) Register (Continued)

Bit(s)	Name	Description
4–2	RCBLK	Receive/transmit current block 000b Block 0. Channel 0 to channel 15 001b Block 1. Channel 16 to channel 31 010b Block 2. Channel 32 to channel 47 011b Block 3. Channel 48 to channel 63 100b Block 4. Channel 64 to channel 79 101b Block 5. Channel 80 to channel 95 110b Block 6. Channel 96 to channel 111 111b Block 7. Channel 112 to channel 127
1	Reserved	Reserved
0	RMCM	Receive multichannel selection enable 0 All 128 channels enabled. 1 All channels disabled by default. Required channels are selected by enabling RP(A/B)BLK and RCER(A/B) appropriately.BSP serial port control extension register (SPCE):HALTR bit

6.7 Pin Control Register (PCR)

Each McBSP has one pin control register of the form shown in Figure 6–13. This I/O-mapped register enables you to:

- ❏ Choose a frame-sync mode for the transmitter (FSXM) and for the receiver (FSRM)
- ❏ Choose a clock mode for transmitter (CLKXM) and for the receiver (CLKRM)
- ❏ Select the input clock source for the sample rate generator (SCLKME, in conjunction with the CLKSM bit of SRGR2)
- ❏ Read or write data when the CLKS, DX, and DR pins are configured as general-purpose I/O pins (CLKS_STAT, DX_STAT, and DX_STAT)
- ❏ Choose whether frame-sync signals are active low or active high (FSXP for transmission, FSRP for reception)
- ❏ Specify whether data is sampled on the falling edge or the rising edge of the clock signals (CLKXP for transmission, CLKRP for reception)

Figure 6–13. Pin Control Register (PCR)

15	12	11	10	9	8
Reserved		FSXM	FSRM	CLKXM	CLKRM
R–0		R/W-0	R/W-0	R/W-0	R/W-0

7	6	5	4	3	2	1	0
SCLKME	CLKS_STAT	DX_STAT	DR_STAT	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

Note: R = Read access, W = Write access, C = Clear only, –n = value after reset;

Bit(s)	Name	Description
15–12	Reserved	Reserved
11	FSXM	Transmit frame synchronization mode
		0 Frame synchronization pulses generated by an external device. FSR is an input pin
		1 Frame synchronization generated internally by sample rate generator. FSR is an output pin except when GSYNC = 1 in SRGR.

Figure 6–13. Pin Control Register (PCR) (Continued)

Bit(s)	Name	Description
10	FSRM	Receive frame synchronization mode 0 Frame synchronization signal derived from an external source 1 Frame synchronization is determined by the Sample Rate Generator frame synchronization mode bit FSGM in the SRGR2.
9	CLKXM	Transmitter clock mode 0 Receiver/Transmitter clock is driven by an external clock with CLK(R/X) as an input pin. 1 CLK(R/X) is an output pin and is driven by the internal sample rate generator. During SPI mode (CLKSTP is a non-zero value): 0 McBSP is a slave and clocks (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	Receiver clock mode Case 1: Digital Loop Back Mode not set (DLB = 0) in SPCR1 0 Receive clock (CLKR) is an input driven by an external clock. 1 CLKR is an output pin and is driven by the internal sample rate generator. Case 2: Digital loop back mode set (DLB = 1) in SPCR1 0 Receive clock (not the CLKR pin) is driven by transmit clock (CLKX) which is based on CLKXM bit in PCR. CLKR pin is in high impedance. 1 CLKR is an output pin and is driven by the transmit clock. The transmit clock is derived based on CLKXM bit in the PCR.

Figure 6–13. Pin Control Register (PCR) (Continued)

Bit(s)	Name	Description
7	SCLKME	Enhanced sample clock mode selection bit. McBSP allow either the receive clock pin (CLKR) or the transmit clock pin (CLKX) to be configured as the input clock to the sample rate generator. This enhancement is enabled through two register bits: pin control register (PCR) bit 7 – (SCLKME), and sample rate generator register 2 (SRGR2) bit 13 – McBSP sample rate generator clock mode (CLKSM) SCLKME CLKSM 0 0 Reserved 0 1 LSPCLK – Internal clock 1 0 External CLKR pin clock 1 1 External CLKX pin clock
6	CLKS_ STAT	Reserved in this implementation. Read 0
5	DX_STAT	DX pin status. Reflects value driven on to DX pin when selected as a general purpose output.
4	DR_STAT	DR pin status. Reflects value on DR pin when selected as a general purpose input.
3	FSXP	Transmit frame synchronization polarity 0 Frame synchronization pulse FSXP is active high 1 Frame synchronization pulse FSXP is active low.
2	FSRP	Receive frame synchronization polarity 0 Frame synchronization pulse FSRP is active high 1 Frame synchronization pulse FSRP is active low
1	CLKXP	Transmit clock polarity 0 Transmit data sampled on rising edge of CLKX 1 Transmit data sampled on falling edge of CLKX
0	CLKRP	Receive clock polarity 0 Receive data sampled on falling edge of CLKR 1 Receive data sampled on rising edge of CLKR

6.8 Receive Channel Enable Registers (RCERA – RCERH)

Each McBSP has eight receive channel enable registers of the form shown in Figure 6–14. There is one for each of the receive partitions: A, B, C, D, E, F, G, and H.

These memory-mapped registers are only used when the receiver is configured to allow individual enabling and disabling of the channels (RMCM = 1) or when the receiver needs bit-enable patterns for the A-bis mode (ABIS = 1). For more details about the way these registers are used be sure to read the topics at the end of this section:

- *RCERs Used in the Receive Multichannel Selection Mode* (page 6-26)
- *RCERs Used in the A-bis Mode* (page 6-27)

Figure 6–14. Receive Channel Enable Register (RCERA/B)

15	14	13	12	11	10	9	8
RCEA15	RCEA14	RCEA13	RCEA12	RCEA11	RCEA10	RCEA9	RCEA8
R/W-0	R/W–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
RCEA7	RCEA6	RCEA5	RCEA4	RCEA3	RCEA2	RCEA1	RCEA0
R/W-0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0

Note: R = Read access, W = Write access, C = Clear only, –n = value after reset;

Bit(s)	Name	Description
	RCERA	**Receive/Transmit Enable Register A**
15–0	RCEA*n* 0 ≤ n ≤ 15	Receive/transmit channel enable 0 Disables reception/transmission of *n*th channel in an even-numbered block in partition A 1 Enables reception/transmission of *n*th channel in an even-numbered block in partition A
	RCERB	**Receive/Transmit enable register B**
15–0	RCER*n* 0 ≤ n ≤ 15	Receive/transmit channel enable 0 Disables reception/transmission of *n*th channel in odd-numbered block in partition B. 1 Enables reception/transmission of *n*th channel in odd-numbered block in partition B.

Figure 6–15. RCER(A–G)–Receive Channel Enable Registers – A, C, E, G

15	14	13	12	11	10	9	8
RCERx15	RCERx14	RCERx13	RCERx12	RCERx11	RCERx10	RCERx9	RCERx8
R/W-0	R/W-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
RCERx7	RCERx6	RCERx5	RCERx4	RCERx3	RCERx2	RCERx1	RCERx0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

Note: R = Read access, W = Write access, C = Clear only, –n = value after reset;

Bit(s)	Name	Description
15–0	RCERx	Receive channel enable register
		0 Disables reception of *n*th channel in partition x.
		1 Enables reception of *n*th channel in partition x.

Note: Receive channel enable register bit layout for 128 channels
x= Partition A, C, E, G; n = bit 15–0
y = Partition B, D, F, H; n = bit 15–0

Figure 6–16. RCER(B–H)–Receive Channel Enable Registers – B,D,F,H

15	14	13	12	11	10	9	8
RCERy15	RCERy14	RCERy13	RCERy12	RCERy11	RCERy10	RCERy9	RCERy8
R/W-0	R/W-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
RCERy7	RCERy6	RCERy5	RCERy4	RCERy3	RCERy2	RCERy1	RCERy0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

Note: R = Read access, W = Write access, C = Clear only, –n = value after reset;

Bit(s)	Name	Description
15–0	RCERy	Receive channel enable register
		0 Disables reception of *n*th channel in partition y.
		1 Enables reception of *n*th channel in partition y.

Note: Receive channel enable register bit layout for 128 channels
x= Partition A, C, E, G; n = bit 15–0
y = Partition B, D, F, H; n = bit 15–0

6.8.1 RCERs Used in the Receive Multichannel Selection Mode

For multichannel selection operation, the assignment of channels to the RCERs depends on whether 32 or 128 channels are individually selectable, as defined by the RMCME bit (see Table 6–1).

Table 6–1. Use of the Receive Channel Enable Registers in theReceive Multichannel Selection Mode

Selectable Channels	Block Assignments	Channel Assignments
32 (RMCME = 0)	RCERA: Channels n–(n + 15) (channels assigned with the RPABLK bits)	RCEA0: Channel n RCEA1: Channel (n + 1) RCEA2: Channel (n + 2) : RCEA15: Channel (n + 15)
	RCERB: Channels m–(m + 15) (channels assigned with the RPBBLK bits)	RCEB0: Channel m RCEB1: Channel (m + 1) RCEB2: Channel (m + 2) : RCEB15: Channel (m + 15)
	Other RCERs not used	–
128 (RMCME = 1)	RCERA: Block 0	RCEA0: Channel 0 RCEA1: Channel 1 RCEA2: Channel 2 : RCEA15: Channel 15
	RCERB: Block 1	RCEB0: Channel 16 RCEB1: Channel 17 RCEB2: Channel 18 : RCEB15: Channel 31
	RCERC: Block 2	RCEC0: Channel 32 RCEC1: Channel 33 RCEC2: Channel 34 : RCEC15: Channel 47
	RCERD: Block 3	RCED0: Channel 48 RCED1: Channel 49 RCED2: Channel 50 : RCED15: Channel 63

Table 6–1. Use of the Receive Channel Enable Registers in theReceive Multichannel Selection Mode (Continued)

Selectable Channels	Block Assignments	Channel Assignments
128 (continued)	RCERE: Block 4	RCEE0: Channel 64 RCEE1: Channel 65 RCEE2: Channel 66 : RCEE15: Channel 79
	RCERF: Block 5	RCEF0: Channel 80 RCEF1: Channel 81 RCEF2: Channel 82 : RCEF15: Channel 95
	RCERG: Block 6	RCEG0: Channel 96 RCEG1: Channel 97 RCEG2: Channel 98 : RCEG15: Channel 111
	RCERH: Block 7	RCEH0: Channel 112 RCEH1: Channel 113 RCEH2: Channel 114 : RCEH15: Channel 127

6.8.2 RCERs Used in the A-bis Mode

In A-bis mode operation, only RCERA and RCERB are used. Sixteen bits at a time are passed to the receiver. As each of the 16 bits arrives on the DR pin (the MSB, bit 15, arrives first), the bit is stored or ignored, depending on the bit-enable pattern in RCERA or RCERB. The first 16 bits that arrive are handled according to the bit-enable pattern in RCERA. For example, if RCEA6 = 1 and all the other bits of RCERA are 0s, only bit 6 is stored. Each of the next 16 bits is stored or ignored according to the bit-enable pattern in RCERB. The receiver alternately uses RCERA and RCERB for consecutive 16-bit words.

Table 6–2 shows how bits of RCERA correspond to the bits of the incoming word.

Table 6–2. Use of Receive Channel Enable Registers A and B in the A-bis Mode

Register	Bits
RCERA	RCEA15: Enables or masks the first bit of the incoming word RCEA14: Enables or masks bit 14 of the incoming word RCEA13: Enables or masks bit 13 of the incoming word : RCEA0: Enables or masks the last bit of the incoming word
RCERB	RCEB15: Enables or masks the first bit of the incoming word RCEB14: Enables or masks bit 14 of the incoming word RCEB13: Enables or masks bit 13 of the incoming word : RCEB0: Enables or masks the last bit of the incoming word

6.9 Transmit Channel Enable Registers (XERA – XCERH)

Each McBSP has eight transmit channel enable registers of the form shown in the following figure. There is one for each of the transmit partitions: A, B, C, D, E, F, G, and H. provides a general bit description that applies to each of the transmit channel enable registers.

These I/O-mapped registers are only used when transmitter is configured to allow individual disabling/enabling and masking/unmasking of the channels (XMCM is nonzero) or when the transmitter needs bit-enable patterns for the A-bis mode (ABIS = 1). For more details about the way these registers are used, read the topics at the end of this section:

- *XCERs Used in a Transmit Multichannel Selection Mode* (page 6-30)
- *XCERs Used in the A-bis Mode* (page 6-32)

Figure 6–17. Transmit Channel Enable Registers A. C. E. G (XCERA–XCERG)

15	14	13	12	11	10	9	8
XCERx15	XCERx14	XCERx13	XCERx12	XCERx11	XCERx10	XCERx9	XCERx8
R/W-0	R/W-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
XCERx7	XCERx6	XCERx5	XCERx4	XCERx3	XCERx2	XCERx1	XCERx0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

Note: R = Read access, W = Write access, C = Clear only, –n = value after reset;

Bit(s)	Name	Description
15–0	XCERx	Receive channel enable register
		0 Disables transmit of *n*th channel in partition x.
		1 Enables transmit of *n*th channel in partition x.

Note: Transmit channel enable register bit layout for 128 channels
x= Partition A, C, E, G; n = bit 15–0
y = Partition B, D, F, H; n = bit 15–0

Figure 6–18. Transmit Channel Enable Registers–B, D, F, H (XCERB–XCERH)

15	14	13	12	11	10	9	8
XCERy15	XCERy14	XCERy13	XCERy12	XCERy11	XCERy10	XCERy9	XCERy8
R/W-0	R/W-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
XCERy7	XCERy6	XCERy5	XCERy4	XCERy3	XCERy2	XCERy1	XCERy0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

Note: R = Read access, W = Write access, C = Clear only, –n = value after reset;

Bit(s)	Name	Description
15–0	XCERy	Receive channel enable register
		0 Disables transmit of *n*th channel in partition y.
		1 Enables transmit of *n*th channel in partition y.

Note: Transmit channel enable register bit layout for 128 channels
x= Partition A, C, E, G; n = bit 15–0
y = Partition B, D, F, H; n = bit 15–0

6.9.1 XCERs Used in a Transmit Multichannel Selection Mode

For multichannel selection operation, the assignment of channels to the XCERs, depends on whether 32 or 128 channels are individually selectable, as defined by the XMCME bit, as shown in the following table.

Note:

When XMCM = 11b (for symmetric transmission and reception), the transmitter uses the receive channel enable registers (RCERs) to enable channels and uses the XCERs to unmask channels for transmission.

Table 6–3. Use of the Transmit Channel Enable Registers in a Transmit Multichannel Selection Mode

Selectable Channels	Block Assignments	Channel Assignments
32 (XMCME = 0)	XCERA: Channels n–(n + 15) (channels assigned with the XPABLK bits for XMCM = 01b or 10b; assigned with the RPABLK bits for XMCM = 11b)	XCEA0: Channel n XCEA1: Channel (n + 1) XCEA2: Channel (n + 2) : XCEA15: Channel (n + 15)
	XCERB: Channels m–(m + 15) (channels assigned with the XPBBLK bits for XMCM = 01b or 10b; assigned with the RPBBLK bits for XMCM = 11b)	XCEB0: Channel m XCEB1: Channel (m + 1) XCEB2: Channel (m + 2) : XCEB15: Channel (m + 15)
	Other XCERs not used	–
128 (XMCME = 1)	XCERA: Block 0	XCEA0: Channel 0 XCEA1: Channel 1 XCEA2: Channel 2 : XCEA15: Channel 15
	XCERB: Block 1	XCEB0: Channel 16 XCEB1: Channel 17 XCEB2: Channel 18 : XCEB15: Channel 31
	XCERC: Block 2	XCEC0: Channel 32 XCEC1: Channel 33 XCEC2: Channel 34 : XCEC15: Channel 47
128 (continued)	XCERD: Block 3	XCED0: Channel 48 XCED1: Channel 49 XCED2: Channel 50 : XCED15: Channel 63
	XCERE: Block 4	XCEE0: Channel 64 XCEE1: Channel 65 XCEE2: Channel 66 : XCEE15: Channel 79

Table 6–3. Use of the Transmit Channel Enable Registers in a Transmit Multichannel Selection Mode (Continued)

Selectable Channels	Block Assignments	Channel Assignments
	XCERF: Block 5	XCEF0: Channel 80 XCEF1: Channel 81 XCEF2: Channel 82 : XCEF15: Channel 95
	XCERG: Block 6	XCEG0: Channel 96 XCEG1: Channel 97 XCEG2: Channel 98 : XCEG15: Channel 111
	XCERH: Block 7	XCEH0: Channel 112 XCEH1: Channel 113 XCEH2: Channel 114 : XCEH15: Channel 127

6.9.2 XCERs Used in the A-bis Mode

In A-bis mode operation, only XCERA and XCERB are used. Sixteen bits at a time are passed to data transmit register 1 (DXR1) by the CPU or FIFO. Each of the 16 bits is transmitted on the DX pin or is ignored, depending on the bit-enable pattern in XCERA or XCERB. The first 16 bits that enter the transmitter are handled according to the bit-enable pattern in XCERA. For example, if XCEA13 = 1 and all the other bits of XCERA are 0s, only bit 13 is transmitted. Each of the next 16 bits is transmitted or ignored according to the bit-enable pattern in XCERB. The transmitter alternately uses XCERA and XCERB for consecutive 16-bit words.

Table 6–4 shows how bits of XCERA and XCERB correspond to the bits of a word written to DXR1 (MSB = most significant bit, LSB = least significant bit).

Table 6–4. Use of Transmit Channel Enable Registers A and B in the A-bis Mode

Register	Bits
XCERA	XCEA15: Enables or masks the MSB of the word written to DXR1 XCEA14: Enables or masks bit 14 of the word written to DXR1 XCEA13: Enables or masks bit 13 of the word written to DXR1 : XCEA0: Enables or masks the LSB of the word written to DXR1
XCERB	XCEB15: Enables or masks the MSB of the word written to DXR1 XCEB14: Enables or masks bit 14 of the word written to DXR1 XCEB13: Enables or masks bit 13 of the word written to DXR1 : XCEB0: Enables or masks the LSB of the word written to DXR1

6.10 Register Bit Summary

Table 6–5. Register Bit Summary (Base Address 0x00 7800)

Register Address Offset 0x00	MSB bits	15	14	13	12	11	10	9	8
	LSB bits	7	6	5	4	3	2	1	0
7800	DRR2	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
		R–0	R–0	R–0	R–0	R–0	R–0	R–0	R–0
		Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
		R–0	R–0	R–0	R–0	R–0	R–0	R–0	R–0
7801	DRR1	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
		R–0	R–0	R–0	R–0	R–0	R–0	R–0	R–0
		Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
		R–0	R–0	R–0	R–0	R–0	R–0	R–0	R–0
7802	DXR2	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
		W–0	W–0	W–0	W–0	W–0	W–0	W–0	W–0
		Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
		W–0	W–0	W–0	W–0	W–0	W–0	W–0	W–0
7803	DXR1	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
		W–0	W–0	W–0	W–0	W–0	W–0	W–0	W–0
		Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
		W–0	W–0	W–0	W–0	W–0	W–0	W–0	W–0
7804	SPCR2	Reserved						FREE	SOFT
		R–0						R/W–0	R/W–0
		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
7805	SPCR1	DLB	RJUST		CLKSTP		Reserved		
		R/W–0	R/W–0		R/W–0		R–0		
		DXENA	ABIS	RINTM		RSYNCERR	RFULL	RRDY	RRST
		R/W–0	R/W–0	R/W–0		R/W–0	R–0	R–0	R/W–0
7806	RCR2	RPHASE	RFRLEN2						
		R/W–0	R/W–0						
		RWDLEN2			RCOMPAND		RFIG	RDATDLY	
		R/W–0			R/W–0		R/W–0	R/W–0	
7807	RCR1	Reserved	RFRLEN1						
		R/W–0	R/W–0						
		RWDLEN1			Reserved				
		R/W–0			R–0				
7808	XCR2	XPHASE	XFRLEN2						
		R/W–0	R/W–0						

Table 6–5. Register Bit Summary (Continued)(Base Address 0x00 7800) (Continued)

Register Address Offset 0x00	MSB bits	15	14	13	12	11	10	9	8
	LSB bits	7	6	5	4	3	2	1	0
		XWDLEN2			XCOMPAND		XFIG	XDATDLY	
		R/W–0			R/W–0		R/W–0	R/W–0	
7809	XCR1	Reserved	XFRLEN1						
		R/W–0	R/W–0						
		XWDLEN1			Reserved				
		R/W–0			R–0				
780A	SRGR2	GSYNC	Reserved	CLKSM	FSGM	FPER			
		R/W–0	R/W–0	R/W–1	R/W–0	R/W–0			
		FPER							
		R/W–0							
780B	SRGR1	FWID							
		R/W–0							
		CLKGDV							
		R/W–0							
780C	MCR2	Reserved						XMCME	XPBBLK
		R–0						R/W–0	R/W–0
		XPBBLK	XPABLK		XCBLK			XMCM	
		R/W–0	R/W–0		R–0			R/W–0	
780D	MCR1	Reserved						RMCME	RPBBLK
		R–0						R/W–0	R/W–0
		RPBBLK	RPABLK		RCBLK			Reserved	RMCM
		R/W–0	R/W–0		R–0			R–0	R/W–0
780E	RCERA	RCEA15	RCEA14	RCEA13	RCEA12	RCEA11	RCEA10	RCEA9	RCEA8
		R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0
		RCEA7	RCEA6	RCEA5	RCEA4	RCEA3	RCEA2	RCEA1	RCEA0
		R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0
780F	RCERB	RCEB15	RCEB14	RCEB13	RCEB12	RCEB11	RCEB10	RCEB9	RCEB8
		R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0
		RCEB7	RCEB6	RCEB5	RCEB4	RCEB3	RCEB2	RCEB1	RCEB0
		R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0
7810	XCERA	XCEA15	XCEA14	XCEA13	XCEA12	XCEA11	XCEA10	XCEA9	XCEA8
		R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0
		XCEA7	XCEA6	XCEA5	XCEA4	XCEA3	XCEA2	XCEA1	XCEA0
		R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0
7811	XCERB	XCEB15	XCEB14	XCEB13	XCEB12	XCEB11	XCEB10	XCEB9	XCEB8
		R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0

Table 6–5. Register Bit Summary (Continued)(Base Address 0x00 7800) (Continued)

Register Address Offset 0x00	MSB bits	15	14	13	12	11	10	9	8
	LSB bits	7	6	5	4	3	2	1	0
		XCEB7	XCEB6	XCEB5	XCEB4	XCEB3	XCEB2	XCEB1	XCEB0
		R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0
7812	PCR	Reserved				FSXM	FSRM	CLKXM	CLKRM
		R–0				R/W–0	R/W–0	R/W–0	R/W–0
		SCLKME	Reserved	DX_STAT	DR_STAT	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
7813	RCERC	RCEC15	RCEC14	RCEC13	RCEC12	RCEC11	RCEC10	RCEC9	RCEC8
		R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0
		RCEC7	RCEC6	RCEC5	RCEC4	RCEC3	RCEC2	RCEC1	RCEC0
		R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0
7814	RCERD	RCED15	RCED14	RCED13	RCED12	RCED11	RCED10	RCED9	RCED8
		R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0
		RCED7	RCED6	RCED5	RCED4	RCED3	RCED2	RCED1	RCED0
		R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0
7815	XCERC	XCEC15	XCEC14	XCEC13	XCEC12	XCEC11	XCEC10	XCEC9	XCEC8
		R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0
		XCEC7	XCEC6	XCEC5	XCEC4	XCEC3	XCEC2	XCEC1	XCEC0
		R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0
7816	XCERD	XCED15	XCED14	XCED13	XCED12	XCED11	XCED10	XCED9	XCED8
		R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0
		XCED7	XCED6	XCED5	XCED4	XCED3	XCED2	XCED1	XCED0
		R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0
7817	RCERE	RCEB15	RCEB14	RCEB13	RCEB12	RCEB11	RCEB10	RCEB9	RCEB8
		R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0
		RCEB7	RCEB6	RCEB5	RCEB4	RCEB3	RCEB2	RCEB1	RCEB0
		R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0
7818	RCERF	RCEF15	RCEF14	RCEF13	RCEF12	RCEF11	RCEF10	RCEF9	RCEF8
		R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0
		RCEF7	RCEF6	RCEF5	RCEF4	RCEF3	RCEF2	RCEF1	RCEF0
		R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0
7819	XCERE	XCEB15	XCEB14	XCEB13	XCEB12	XCEB11	XCEB10	XCEB9	XCEB8
		R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0
		XCEB7	XCEB6	XCEB5	XCEB4	XCEB3	XCEB2	XCEB1	XCEB0
		R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0
781A	XCERF	XCEF15	XCEF14	XCEF13	XCEF12	XCEF11	XCEF10	XCEF9	XCEF8
		R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0

Table 6–5. Register Bit Summary (Continued)(Base Address 0x00 7800) (Continued)

Register Address Offset 0x00	MSB bits	15	14	13	12	11	10	9	8
	LSB bits	7	6	5	4	3	2	1	0
		XCEF7	XCEF6	XCEF5	XCEF4	XCEF3	XCEF2	XCEF1	XCEF0
		R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0
781B	RCERG	RCEG15	RCEG14	RCEG13	RCEG12	RCEG11	RCEG10	RCEG9	RCEG8
		R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0
		RCEG7	RCEG6	RCEG5	RCEG4	RCEG3	RCEG2	RCEG1	RCEG0
		R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0
781C	RCERH	RCEH15	RCEH14	RCEH13	RCEH12	RCEH11	RCEH10	RCEH9	RCEH8
		R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0
		RCEH7	RCEH6	RCEH5	RCEH4	RCEH3	RCEH2	RCEH1	RCEH0
		R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0
781D	XCERG	XCEG15	XCEG14	XCEG13	XCEG12	XCEG11	XCEG10	XCEG9	XCEG8
		R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0
		XCEG7	XCEG6	XCEG5	XCEG4	XCEG3	XCEG2	XCEG1	XCEG0
		R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0
781E	XCERH	XCEH15	XCEH14	XCEH13	XCEH12	XCEH11	XCEH10	XCEH9	XCEH8
		R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0
		XCEH7	XCEH6	XCEH5	XCEH4	XCEH3	XCEH2	XCEH1	XCEH0
		R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0

The registers bits are summarized in Table 6–6.

Table 6–6. FIFO Register Bit Descriptions (Base address 0x00 7800)

Register Address-Offset 0x000	MSB bits	15	14	13	12	11	10	9	8
	LSB bits	7	6	5	4	3	2	1	0
00	DRR2	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
		R–0	R–0	R–0	R–0	R–0	R–0	R–0	R–0
		Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
		R–0	R–0	R–0	R–0	R–0	R–0	R–0	R–0
01	DRR1	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
		R–0	R–0	R–0	R–0	R–0	R–0	R–0	R–0
		Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
		R–0	R–0	R–0	R–0	R–0	R–0	R–0	R–0
02	DXR2	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
		W–0	W–0	W–0	W–0	W–0	W–0	W–0	W–0
		Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16

Table 6–6. FIFO Register Bit Descriptions (Base address 0x00 7800) (Continued)

Register Address-Offset 0x000	MSB bits	15	14	13	12	11	10	9	8
	LSB bits	7	6	5	4	3	2	1	0
		W–0	W–0	W–0	W–0	W–0	W–0	W–0	W–0
03	DXR1	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
		W–0	W–0	W–0	W–0	W–0	W–0	W–0	W–0
		Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
		W–0	W–0	W–0	W–0	W–0	W–0	W–0	W–0
20	MFFTX	Reserved	MFFENA	TXFIFO Reset	TXFFST4	TXFFST3	TXFFST2	TXFFST1	TXFFST0
		R–0	R/W–0	R/W–1	R–0	R–0	R–0	R–0	R–0
		TXFFINT Flag	TXFFINT CLR	TXFFIENA	TXFFIL4	TXFFIL3	TXFFIL2	TXFFIL1	TXFFIL0
		R–0	W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0
21	MFFRX	RXFFOVF Flag	RXFFOVF CLR	RXFIFO Reset	RXFFST4	RXFFST3	RXFIFST2	RXFFST1	RXFFST0
		R–0	W–0	R/W–1	R–0	R–0	R–0	R–0	R–0
		RXFFINT-Flag	RXFFINT–CLR	RXFFIENA	RXFFIL4	RXFFIL3	RXFFIL2	RXFFIL1	RXFFIL0
		R–0	W–0	R/W–0	R/W–1	R/W–1	R/W–1	R/W–1	R/W–1
22	MFFCT	Reserved							
		R–0							
		FFTXDLY7	FFTXDLY6	FFTXDLY5	FFTXDLY4	FFTXDLY3	FFTXDLY2	FFTXDLY1	FFTXDLY0
		R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0	R/W–0
23	MFFINT	Reserved							
		R–0							
		Reserved				REVTA ENA	RINT ENA	XEVTA ENA	XINT ENA
		R–0				R/W–0	R/W–0	R/W–0	R/W–0
24	MFFST	Reserved							
		R–0							
		Reserved				FSR Flag	EOBR Flag	FSX Flag	EOBX Flag
		R–0				R/W–x	R/W–0	R/W–x	R/W–0

Index

A

A-bis mode
  enable/disable (receiver configuration), 3-6
  enable/disable (transmitter configuration), 3-31
  introduction, 2-13
  receive operation, 2-13
  transmit operation, 2-14
A-law format (companding), 1-11
ABIS bit (A-bis mode bit) of SPCR1, 6-4
AC97 standard implemented in McBSP, 1-18

B

bit order reverse option for McBSP transfer, 1-14
BSP serial port control extension register (SPCE), HALTR bit, 6-18, 6-20

C

channels (McBSP), disabling/enabling/masking/unmasking, 2-9
CLKGDV bits of SRGR1, 6-14
CLKR pin polarity bit (CLKRP), 6-21
CLKRM bit of PCR, 6-21
CLKRP bit of PCR, 6-21
CLKS pin polarity bit (CLKSP), 6-14
CLKS pin status bit (CLKS_STAT), 6-21
CLKS_STAT bit of PCR, 6-21
CLKSM bit of SRGR2, 6-14
CLKSP bit of SRGR2, 6-14
CLKSTP bits of SPCR1, 6-4
CLKX pin polarity bit (CLKXP), 6-21
CLKXM bit of PCR, 6-21
CLKXP bit of PCR, 6-21
clock divide-down value for sample rate generator
  McBSP receiver configuration, 3-23
  McBSP transmitter configuration, 3-46
clock generation in the sample rate generator, 1-28
clock modes
  McBSP reception, 3-20
  sample rate generator (McBSP receiver configuration), 3-24
  sample rate generator (McBSP transmitter configuration), 3-47
clock pin polarities
  McBSP reception, 3-22
  McBSP transmission, 3-44
clock polarities
  input clock of sample rate generator (receiver configuration), 3-24
  input clock of sample rate generator (transmitter configuration), 3-47
clock stop (SPI) mode
  McBSP receiver configuration, 3-5
  McBSP transmitter configuration, 3-29
clock stop (SPI) mode bits (CLKSTP), 6-4
clock stop (SPI) mode timing diagrams, 2-17
clock synchronization mode bit for CLKG (GSYNC), 6-14
clock synchronization mode for sample rate generator, McBSP transmitter configuration, 3-46
clocking data in McBSP, 1-14
companding data (McBSP), 1-11
companding internal data (McBSP), 1-13
companding mode, McBSP transmission, 3-34
compressing transmit data (McBSP), 1-11

D

data delay, McBSP transmission, 3-36
data packing in McBSP
  using frame length and word length, 4-8

using word length and the frame-sync ignore function, 4-9
data receive registers (DRR1 and DRR2), 6-2
data reception in McBSP, 1-21
data transfer process of McBSP, 1-10
data transmission in McBSP, 1-22
data transmit registers (DXR1 and DXR2), 6-3
diagrams
McBSP, 1-4
McBSP data transfer process, 1-10
McBSP reception, 1-21
McBSP transmission, 1-22
sample rate generator, 1-26
digital loopback mode, McBSP transmitter configuration, 3-28
digital loopback mode bit (DLB), 6-4
disabled channel (McBSP), 2-9
divide-down value for CLKG (CLKGDV), 6-14
dividing down
input clock of sample rate generator (McBSP receiver configuration), 3-23
input clock of sample rate generator (McBSP transmitter configuration), 3-46
DLB bit of SPCR1, 6-4
DR pin status bit (DR_STAT), 6-21
DR_STAT bit of PCR, 6-21
DRR1 and DRR2, 6-2
DSP reset, effects on McBSP, 4-3
DX delay enabler mode, 3-37
DX delay enabler mode bit (DXENA), 6-4
DX pin status bit (DX_STAT), 6-21
DX_STAT bit of PCR, 6-21
DXENA bit of SPCR1, 6-4
DXR1 and DXR2, 6-3

E

emulation mode bits of McBSP (FREE and SOFT), 6-4
emulation modes, McBSP, 4-2
enabled channel (McBSP), 2-9
error/exception conditions of McBSP, 1-39
exception/error conditions of McBSP, 1-39
expanding receive data (McBSP), 1-11

F

FIFO events generated by McBSP, 1-24
figures
McBSP, 1-4
McBSP data transfer process, 1-10
McBSP reception, 1-21
McBSP transmission, 1-22
sample rate generator, 1-26
FPER bits of SRGR2, 6-14
frame configuration for multichannel selection, 2-2
frame frequency (McBSP), 1-17
frame length, McBSP transmission, 3-32
frame of data (McBSP), 1-15
frame phases
introduction, 1-18
McBSP reception, 3-6
McBSP transmission, 3-31
frame sync generation in the sample rate generator, 1-31
frame synchronization (McBSP), 1-15
frame-sync ignore function
McBSP reception, 3-8
McBSP transmission, 3-33
frame-sync logic reset bit ($\overline{\text{FRST}}$), 6-4
frame-sync modes, McBSP reception, 3-15
frame-sync period bits for FSG (FPER), 6-14
frame-sync period for sample rate generator
McBSP receiver configuration, 3-18
McBSP transmitter configuration, 3-42
frame-sync pin polarities
McBSP reception, 3-16
McBSP transmission, 3-40
frame-sync pulse, 1-15
frame-sync pulse width bits for FSG (FWID), 6-14
frame-sync pulse width for sample rate generator
McBSP receiver configuration, 3-18
McBSP transmitter configuration, 3-42
FREE and SOFT bits of SPCR2, 6-4
$\overline{\text{FRST}}$ bit of SPCR2, 6-4
FSGM bit of SRGR2, 6-14
FSR pin polarity bit (FSRP), 6-21
FSRM bit of PCR, 6-21
FSRP bit of PCR, 6-21
FSX pin polarity bit (FSXP), 6-21
FSXM bit of PCR, 6-21

FSXP bit of PCR, 6-21
FWID bits of SRGR1, 6-14

G

general-purpose I/O, using McBSP pins, 4-11
$\overline{\text{GRST}}$ bit of SPCR2, 6-4
GSYNC bit of SRGR2, 6-14

I

idle enable bit for McBSP, 6-21
IDLE_EN bit of PCR, 6-21
illustrations
  McBSP, 1-4
  McBSP data transfer process, 1-10
  McBSP reception, 1-21
  McBSP transmission, 1-22
  sample rate generator, 1-26
initializing a McBSP, 4-3
initializing a sample rate generator, 1-34
input clock for sample rate generator
  McBSP receiver configuration, 3-24
  McBSP transmitter configuration, 3-47
input clock polarity for sample rate generator
  McBSP receiver configuration, 3-24
  McBSP transmitter configuration, 3-47
interrupt modes, McBSP transmission, 3-38
interrupts
  between McBSP block transfers, 2-12
  generated by McBSP, 1-24

J

justification of receive data (McBSP), 3-12

L

LSB-first option for McBSP transfers, 1-14

M

masked channel (McBSP), 2-9
McBSP
  A-bis mode (introduction), 2-13
  A-bis mode (receiver configuration), 3-6
  A-bis mode (transmitter configuration), 3-31
  block diagram, 1-4
  clock stop (SPI) mode (receiver configuration), 3-5
  clock stop (SPI) mode (transmitter configuration), 3-29
  clocking and framing data, 1-14
  companding internal data, 1-13
  configuration for SPI operation, 2-19
  data transfer process, 1-10
  digital loopback mode (transmitter configuration), 3-28
  emulation modes, 4-2
  exception/error conditions, 1-39
  frame phases, 1-18
  initializing, 4-3
  interrupts and DMA events, 1-24
  interrupts between block transfers, 2-12
  master in the SPI protocol, 2-20
  operating transmitter synchronously with receiver, 1-33
  possible responses to receive frame-sync pulses, 1-41
  possible responses to transmit frame-sync pulses, 1-47
  receive multichannel selection mode (introduction), 2-7
  receive multichannel selection mode (receiver configuration), 3-6
  receiver configuration procedure, 3-2
  reception, 1-21
  registers, 1-7
  resetting, 4-3
  sample rate generator, 1-26
  sign-extension and justification mode, 3-12
  slave in the SPI protocol, 2-22
  transmission, 1-22
  transmit multichannel selection modes (introduction), 2-8
  transmit multichannel selection modes (transmitter configuration), 3-30
  transmitter configuration procedure, 3-26
MCR1 and MCR2, 6-17
Mu-law format (companding), 1-11
multichannel buffered serial port (McBSP)
  A-bis mode (introduction), 2-13
  A-bis mode (receiver configuration), 3-6
  A-bis mode (transmitter configuration), 3-31

block diagram, 1-4
clock stop (SPI) mode (receiver configuration), 3-5
clock stop (SPI) mode (transmitter configuration), 3-29
clocking and framing data, 1-14
companding internal data, 1-13
configuration for SPI operation, 2-19
data transfer process, 1-10
digital loopback mode (transmitter configuration), 3-28
emulation modes, 4-2
exception/error conditions, 1-39
frame phases, 1-18
initializing, 4-3
interrupts and FIFO events, 1-24
interrupts between block transfers, 2-12
master in the SPI protocol, 2-20
operating transmitter synchronously with receiver, 1-33
possible responses to receive frame-sync pulses, 1-41
possible responses to transmit frame-sync pulses, 1-47
receive multichannel selection mode (introduction), 2-7
receive multichannel selection mode (receiver configuration), 3-6
receiver configuration procedure, 3-2
reception, 1-21
registers, 1-7
resetting, 4-3
sample rate generator, 1-26
sign-extension and justification mode, 3-12
slave in the SPI protocol, 2-22
transmission, 1-22
transmit multichannel selection modes (introduction), 2-8
transmit multichannel selection modes (transmitter configuration), 3-30
transmitter configuration procedure, 3-26
multichannel control registers (MCR1 and MCR2), 6-17
multichannel selection modes, 2-2

O

overrun in the McBSP receiver, 1-40
overwrite in the McBSP transmitter, 1-44

P

PCR, 6-21
phases of a frame
introduction, 1-18
McBSP receive frame, 3-6
McBSP transmit frame, 3-31
pictures
McBSP, 1-4
McBSP data transfer process, 1-10
McBSP reception, 1-21
McBSP transmission, 1-22
sample rate generator, 1-26
pin control register (PCR), 6-21
polarity of sample rate generator input clock
McBSP receiver configuration, 3-24
McBSP transmitter configuration, 3-47

R

RCBLK bits of MCR1, 6-17
RCEp0–RCEp15 bits of RCERp, 6-24
RCERA–RCERH, 6-24
RCOMPAND bits of RCR2, 6-8
RCR1 and RCR2, 6-8
RDATDLY bits of RCR2, 6-8
receive channel enable bits for partition p (RCEp0–RCEp15), 6-24
receive channel enable registers (RCERA–RCERH), 6-24
receive clock mode, 3-20
receive clock mode bit (CLKRM), 6-21
receive clock pin polarity, 3-22
receive companding mode bits (RCOMPAND), 6-8
receive control registers (RCR1 and RCR2), 6-8
receive current block indicator (RCBLK), 6-17
receive data delay bits (RDATDLY), 6-8
receive FIFO event signals (REVT and REVTA), 1-24
receive frame length 1 (RFRLEN1), 6-8
receive frame length 2 (RFRLEN2), 6-8
receive frame phase(s), 3-6
receive frame-sync error bit (RSYNCERR), 6-4
receive frame-sync ignore bit (RFIG), 6-8
receive frame-sync ignore function, 3-8
receive frame-sync mode, 3-15

receive frame-sync mode bit (FSRM), 6-21
receive frame-sync pin polarity, 3-16
receive frame-sync pulses, possible McBSP responses to, 1-41
receive I/O enable bit (RIOEN), 6-21
receive interrupt mode bits (RINTM), 6-4
receive interrupt signal (RINT), 1-24
receive multichannel partition mode bit (RMCME), 6-17
receive multichannel selection mode
enable/disable (McBSP receiver configuration), 3-6
introduction, 2-7
receive multichannel selection mode bit (RMCM), 6-17
receive partition A block bits (RPABLK), 6-17
receive partition B block bits (RPBBLK), 6-17
receive phase number bit (RPHASE), 6-8
receive sign-extension and justification mode, 3-12
receive sign-extension and justification mode bits (RJUST), 6-4
receive word length 1 (RWDLEN1), 6-8
receive word length 2 (RWDLEN2), 6-8
receiver configuration procedure (McBSP), 3-2
receiver full bit (RFULL), 6-4
receiver ready bit (RRDY), 6-4
receiver reset bit ($\overline{\text{RRST}}$), 6-4
reception in McBSP, 1-21
registers, McBSP, 1-7
reset, effects on McBSP, 4-3
resetting a McBSP, 4-3
resetting a sample rate generator, 1-34
reversing bit order for McBSP transfer, 1-14
REVT signal, 1-24
REVTA signal, 1-24
RFIG bit of RCR2, 6-8
RFRLEN1 bits of RCR1, 6-8
RFRLEN2 bits of RCR2, 6-8
RFULL bit of SPCR1, 6-4
RINT signal, 1-24
RINTM bits of SPCR1, 6-4
RIOEN and XIOEN bits of PCR, 6-21
RJUST bits of SPCR1, 6-4
RMCM bit of MCR1, 6-17
RMCME bit of MCR1, 6-17
RPABLK bits of MCR1, 6-17
RPBBLK bits of MCR1, 6-17
RPHASE bit of RCR2, 6-8
RRDY bit of SPCR1, 6-4
$\overline{\text{RRST}}$ bit of SPCR1, 6-4
RSYNCERR bit of SPCR1, 6-4
RWDLEN1 bits of RCR1, 6-8
RWDLEN2 bits of RCR2, 6-8

S

sample rate generator
clock divide-down value (McBSP receiver configuration), 3-23
clock divide-down value (McBSP transmitter configuration), 3-46
clock mode (McBSP receiver configuration), 3-24
clock mode (McBSP transmitter configuration), 3-47
clock synchronization mode (McBSP transmitter configuration), 3-46
clocking examples, 1-35
frame-sync period and pulse width (McBSP receiver configuration), 3-18
frame-sync period and pulse width (McBSP transmitter configuration), 3-42
input clock polarity (receiver configuration), 3-24
input clock polarity (transmitter configuration), 3-47
introduction, 1-26
resetting and initializing, 1-34
synchronizing outputs to an external clock, 1-32
using for clock generation, 1-28
using for frame sync generation, 1-31
sample rate generator input clock mode bits
CLKSM bit of SRGR2, 6-14
SCLKME bit of PCR, 6-21
sample rate generator reset bit ($\overline{\text{GRST}}$), 6-4
sample rate generator transmit frame-sync mode bit (FSGM), 6-14
SCLKME bit of PCR, 6-21
serial port (McBSP)
A-bis mode (introduction), 2-13
A-bis mode (receiver configuration), 3-6
A-bis mode (transmitter configuration), 3-31
block diagram, 1-4
clock stop (SPI) mode (receiver configuration), 3-5

clock stop (SPI) mode (transmitter configuration), 3-29
clocking and framing data, 1-14
companding internal data, 1-13
configuration for SPI operation, 2-19
data transfer process, 1-10
digital loopback mode (transmitter configuration), 3-28
emulation modes, 4-2
exception/error conditions, 1-39
frame phases, 1-18
initializing, 4-3
interrupts and FIFOevents, 1-24
interrupts between block transfers, 2-12
introduction, 1-2
master in the SPI protocol, 2-20
operating transmitter synchronously with receiver, 1-33
possible responses to receive frame-sync pulses, 1-41
possible responses to transmit frame-sync pulses, 1-47
receive multichannel selection mode (introduction), 2-7
receive multichannel selection mode (receiver configuration), 3-6
receiver configuration procedure, 3-2
reception, 1-21
registers, 1-7
resetting, 4-3
sample rate generator, 1-26
sign-extension and justification mode, 3-12
slave in the SPI protocol, 2-22
transmission, 1-22
transmit multichannel selection modes (introduction), 2-8
transmit multichannel selection modes (transmitter configuration), 3-30
transmitter configuration procedure, 3-26

serial port control registers (SPCR1 and SPCR2), 6-4

serial word (McBSP), 1-15

serial word length(s), McBSP transmission, 3-31

sign-extension of receive data (McBSP), 3-12

SOFT and FREE bits of SPCR2, 6-4

SPCR1 and SPCR2, 6-4

SPI operation
McBSP as master, 2-20
McBSP as slave, 2-22
procedure, 2-19

SRGR1 and SRGR2, 6-14

ST-Bus clock examples
double-rate clock, 1-35
single-rate clock, 1-37

synchronizing McBSP transmitter with McBSP receiver, 1-33

synchronizing sample rate generator outputs to an external clock, 1-32

T

TDM serial port control register (TSPC)
MCM bit, 1-27, 6-15
TXM bit, 6-9, 6-15

transmission in McBSP, 1-22

transmit channel enable bits for partition p (XCEp0–XCEp15), 6-29

transmit channel enable registers (XCERA–XCERH), 6-29

transmit clock mode bit (CLKXM), 6-21

transmit clock pin polarity, 3-44

transmit companding mode, 3-34

transmit companding mode bits (XCOMPAND), 6-11

transmit control registers (XCR1 and XCR2), 6-11

transmit current block indicator (XCBLK), 6-17

transmit data delay, 3-36

transmit data delay bits (XDATDLY), 6-11

transmit DX delay enabler mode, 3-37

transmit frame length, 3-32

transmit frame length 1 (XFRLEN1), 6-11

transmit frame length 2 (XFRLEN2), 6-11

transmit frame phase(s), 3-31

transmit frame-sync error bit (XSYNCERR), 6-4

transmit frame-sync ignore bit (XFIG), 6-11

transmit frame-sync ignore function, 3-33

transmit frame-sync mode bit (FSXM), 6-21

transmit frame-sync pin polarity, 3-40

transmit frame-sync pulses, possible McBSP responses to, 1-47

transmit I/O enable bit (XIOEN), 6-21

transmit interrupt mode, 3-38

transmit interrupt mode bits (XINTM), 6-4

transmit interrupt signal (XINT), 1-24

transmit multichannel partition mode bit (XMCME), 6-17

transmit multichannel selection mode bits (XMCM), 6-17
transmit multichannel selection modes
  enable/disable (McBSP transmitter configuration), 3-30
  introduction, 2-8
transmit partition A block bits (XPABLK), 6-17
transmit partition B block bits (XPBBLK), 6-17
transmit phase number bit (XPHASE), 6-11
transmit word length, 3-31
transmit word length 1 (XWDLEN1), 6-11
transmit word length 2 (XWDLEN2), 6-11
transmitter configuration procedure (McBSP), 3-26
transmitter empty bit (XEMPTY), 6-4
transmitter ready bit (XRDY), 6-4
transmitter reset bit (XRST), 6-4

U

underflow in the McBSP transmitter, 1-45
unexpected receive frame-sync pulse, 1-41
unexpected transmit frame-sync pulse, 1-47
unmasked channel (McBSP), 2-9

W

word length(s), McBSP transmission, 3-31

X

XCBLK bits of MCR2, 6-17
XCEp0–XCEp15 bits of XCERp, 6-29
XCERA–XCERH, 6-29
XCOMPAND bits of XCR2, 6-11
XCR1 and XCR2, 6-11
XDATDLY bits of XCR2, 6-11
XEMPTY bit of SPCR2, 6-4
XEVT signal, 1-24
XEVTA signal, 1-24
XFIG bit of XCR2, 6-11
XFRLEN1 bits of XCR1, 6-11
XFRLEN2 bits of XCR2, 6-11
XINT signal, 1-24
XINTM bits of SPCR2, 6-4
XIOEN and RIOEN bits of PCR, 6-21
XMCM bits of MCR2, 6-17
XMCME bit of MCR2, 6-17
XPABLK bits of MCR2, 6-17
XPBBLK bits of MCR2, 6-17
XPHASE bit of XCR2, 6-11
XRDY bit of SPCR2, 6-4
XRST bit of SPCR2, 6-4
XSREMPTY bit, 6-5
XSYNCERR bit of SPCR2, 6-4
XWDLEN1 bits of XCR1, 6-11
XWDLEN2 bits of XCR2, 6-11