Advance Information

This document contains information on a product under development. The parametric information contains target parameters that are subject to change.

CN8478/CN8474A/CN8472A/CN8471A

Multichannel Synchronous Communications Controller (MUSYCC™)

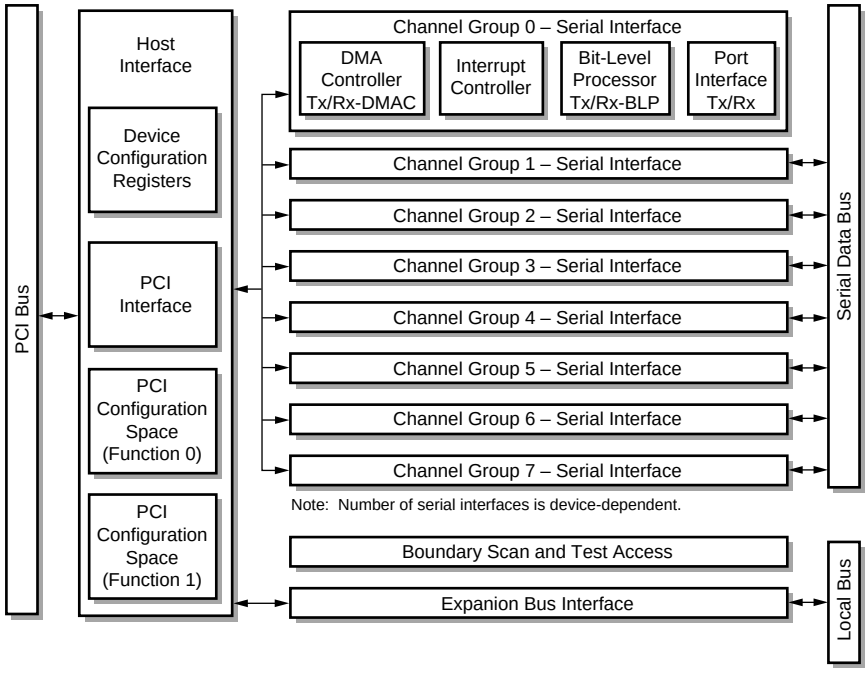
Product Description

The CN8478, CN8474A, CN8472A, and CN8471A are advanced Multichannel Synchronous Communication Controllers (MUSYCCs) that format and deformat up to 256 (CN8478), 128 (CN8474A), 64 (CN8472A), or 32 (CN8471A) HDLC channels in a single CMOS integrated circuit. MUSYCC operates at Layer 2 of the Open Systems Interconnection (OSI) protocol reference model. MUSYCC provides a comprehensive, high-density solution for processing HDLC channels for internetworking applications such as Frame Relay, ISDN D-channel signaling, X.25, Signaling System 7 (SS7), DXI, ISUP, and LAN/WAN data transport. Under minimal host supervision, MUSYCC manages a linked list of channel data buffers in host memory by performing Direct Memory Access (DMA) of the HDLC channels.

MUSYCC interfaces with eight independent serial data streams, such as T1/E1 signals, and then transfers data across the popular 32-bit Peripheral Component Interface (PCI) bus to system memory at a rate of up to 66 MHz. Each serial interface can be operated at up to 8.192 MHz. Logical channels can be mapped as any combination of DS0 time slots to support ISDN hyperchannels (Nx64 kbps) or as any number of bits in a DS0 for subchanneling applications (Nx8 kbps). MUSYCC also includes a 32-bit expansion port for bridging the PCI bus to local microprocessors or peripherals. A JTAG port enables boundary-scan testing to replace bed-of-nails board testing.

Device drivers for Linux, VxWorks®, and pSOS™ operating systems are available under a no-fee license agreement from Conexant. The device drivers include C source code and supporting software documents.

Functional Block Diagram



Distinguishing Features

- 256-, 128-, 64-, or 32-channel HDLC controller
- OSI Layer 2 protocol support
- General purpose HDLC (ISO 3309)
  - X.25 (LAPB)
  - Frame relay (LAPF/ANSI T1.618)
  - ISDN D-channel (LAPD/Q.921)
  - SS7 support
- 8, 4, 2, or 1 independent serial interfaces which support
  - T1/E1 data streams
  - DC to 8.192 Mbps TDM busses
- Configurable logical channels
  - Standard DS0 (56, 64 kbps)
  - Hyperchannel (Nx64)
  - Subchannel (Nx8)
- Per-channel protocol mode selection
  - 16-bit FCS mode
  - 32-bit FCS mode
  - SS7 mode (16-bit FCS)
  - Transparent mode (unformatted data)
- Per-channel DMA buffer management
  - Linked list data structures
  - Variable size transmit/receive FIFO
- Per-channel message length check
  - Select no length checking
  - Select from two 12-bit registers to compare message length
  - Maximum length 16,384 Bytes
- Direct PCI bus interface
  - 32-bit, 66 or 33 MHz operation
  - Bus master and slave operation
  - PCI Version 2.1
- Local Expansion Bus interface (EBUS)
  - 32-bit multiplexed address/data bus
  - Burst access up to 64 Bytes
- Low power, 3.3/2.5 V CMOS operation
- JTAG boundary scan access port
- 208-pin PQFP/surface-mount package
- BGA

Applications

- ISDN basic-rate or primary-rate interfaces
- ISDN D-channel controller
- Routers
- Cellular base station switch controller
- CSU/DSU
- Protocol converter
- Packet data switch
- Frame relay switches/Frame Relay Access Devices (FRAD)
- DXI network interface
- Distributed packet-based communications system
- Access multiplexer/concentrator

Ordering Information

Model Number	Version	Package	Temperature Range
CN8471AEPF	32-Channel	208-Pin Plastic Quad Flat Pack (PQFP)	–40 °C to +85 °C
CN8472AEPF	64-Channel	208-Pin Plastic Quad Flat Pack (PQFP)	–40 °C to +85 °C
CN8474AEPF	128-Channel	208-Pin Plastic Quad Flat Pack (PQFP)	–40 °C to +85 °C
CN8478EPF	256-Channel	208-Pin Plastic Quad Flat Pack (PQFP)	–40 °C to +85 °C
CN8471AEBG	32-Channel	208-Pin Plastic Ball Grid Array	–40 °C to +85 °C
CN8472AEBG	64-Channel	208-Pin Plastic Ball Grid Array	–40 °C to +85 °C
CN8474AEBG	128-Channel	208-Pin Plastic Ball Grid Array	–40 °C to +85 °C
CN8478EBG	256-Channel	208-Pin Plastic Ball Grid Array	–40 °C to +85 °C

© 2000, Conexant Systems, Inc.
All Rights Reserved.

Information in this document is provided in connection with Conexant Systems, Inc. (“Conexant”) products. These materials are provided by Conexant as a service to its customers and may be used for informational purposes only. Conexant assumes no responsibility for errors or omissions in these materials. Conexant may make changes to specifications and product descriptions at any time, without notice. Conexant makes no commitment to update the information and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to its specifications and product descriptions.

No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted by this document. Except as provided in Conexant’s Terms and Conditions of Sale for such products, Conexant assumes no liability whatsoever.

THESE MATERIALS ARE PROVIDED “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, RELATING TO SALE AND/OR USE OF CONEXANT PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, CONSEQUENTIAL OR INCIDENTAL DAMAGES, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT. CONEXANT FURTHER DOES NOT WARRANT THE ACCURACY OR COMPLETENESS OF THE INFORMATION, TEXT, GRAPHICS OR OTHER ITEMS CONTAINED WITHIN THESE MATERIALS. CONEXANT SHALL NOT BE LIABLE FOR ANY SPECIAL, INDIRECT, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, INCLUDING WITHOUT LIMITATION, LOST REVENUES OR LOST PROFITS, WHICH MAY RESULT FROM THE USE OF THESE MATERIALS.

Conexant products are not intended for use in medical, lifesaving or life sustaining applications. Conexant customers using or selling Conexant products for use in such applications do so at their own risk and agree to fully indemnify Conexant for any damages resulting from such improper use or sale.

The following are trademarks of Conexant Systems, Inc.: Conexant™, the Conexant C symbol, and “What’s Next in Communications Technologies”™. Product names or services listed in this publication are for identification purposes only, and may be trademarks of third parties. Third-party brands and names are the property of their respective owners.

For additional disclaimer information, please consult Conexant’s Legal Information posted at www.conexant.com, which is incorporated by reference.

Reader Response: Conexant strives to produce quality documentation and welcomes your feedback. Please send comments and suggestions to tech.pubs@conexant.com. For technical questions, contact your local Conexant sales office or field applications engineer.

Table of Contents

List of Figures ix

List of Tables xi

1.0 System Description 1-1

1.1 Pin Descriptions 1-5

2.0 Host Interface 2-1

2.1 PCI Interface 2-2

2.1.1 PCI Initialization 2-2

2.1.2 PCI Bus Operations 2-3

2.1.3 PCI Configuration Space 2-3

2.2 PCI Configuration Registers 2-7

2.2.1 Function 0 Network Controller—PCI Master and Slave 2-7

2.2.2 Function 1 Expansion Bus Bridge, PCI Slave 2-13

2.2.3 PCI Reset 2-17

2.2.4 Host Interface 2-17

2.2.5 PCI Bus Parity 2-18

2.2.6 PCI Throughput and Latency Considerations 2-18

2.2.6.1 PCI Bus Latency 2-19

2.2.6.2 Latency Computation—Single Dword Access 2-21

2.2.6.3 Latency Computation—Burst Access 2-22

3.0 Expansion Bus (EBUS) 3-1

3.1 Operation 3-2

3.1.1 Initialization 3-2

3.1.2 Address and Data 3-2

3.1.3 Clock 3-3

3.1.4 Interrupt 3-4

3.1.5 Address Duration 3-4

3.1.6 Data Duration 3-4

3.1.7 Bus Access Interval 3-5

3.1.8 PCI to EBUS Interaction 3-5

3.1.9 Microprocessor Interface 3-6

3.1.10 Arbitration 3-7

3.1.11 Connection 3-8

4.0 Serial Interface . . . 4-1
4.1 Serial Port Interface . . . 4-2
4.2 Bit Level Processor . . . 4-2
4.3 DMA Controller . . . 4-2
4.4 Interrupt Controller . . . 4-2
4.5 Channelized Port Mode . . . 4-3
4.5.1 Hyperchannels (Nx64) . . . 4-3
4.5.2 Subchannels (Nx8) . . . 4-3
4.5.3 Frame Synchronization Flywheel . . . 4-4
4.5.4 Change-of-Frame Alignment . . . 4-8
4.5.5 Out-of-Frame . . . 4-8
4.6 Serial Port Mapping . . . 4-9
4.7 Tx and Rx FIFO Buffer Allocation and Management . . . 4-11
4.7.1 Example Channel BUFFLOC and BUFFLEN Specification . . . 4-13
4.7.2 Receiving Bit Stream . . . 4-15
4.7.3 Transmitting Bit Stream . . . 4-15
4.7.3.1 Transmit Data Bit Output Value Determination . . . 4-16
5.0 Memory Organization . . . 5-1
5.1 Memory Architecture . . . 5-1
5.1.1 Register Map Access and Shared Memory Access . . . 5-3
5.1.2 Memory Access Illustration . . . 5-6
5.2 Descriptors . . . 5-9
5.2.1 Host Interface Level Descriptors . . . 5-10
5.2.1.1 Global Configuration Descriptor . . . 5-10
5.2.1.2 Dual Address Cycle Base Pointer . . . 5-12
5.2.2 Channel Group Level Descriptors . . . 5-12
5.2.2.1 Group Base Pointer . . . 5-12
5.2.2.2 Service Request . . . 5-13
5.2.2.3 Group Configuration Descriptor . . . 5-16
5.2.2.4 Memory Protection Descriptor . . . 5-18
5.2.2.5 Port Configuration Descriptor . . . 5-18
5.2.2.6 Message Length Descriptor . . . 5-20
5.2.2.7 Time Slot Map . . . 5-20
5.2.2.8 Subchannel Map . . . 5-24
5.2.3 Channel Level Descriptors . . . 5-26
5.2.3.1 Channel Configuration Descriptor . . . 5-26
5.2.4 Message Level Descriptor . . . 5-29
5.2.4.1 Using Message Descriptors . . . 5-30
5.2.4.2 Note for Interrupt Driven Drivers . . . 5-31
5.2.4.3 Head Pointer . . . 5-31
5.2.4.4 Message Pointer . . . 5-31
5.2.4.5 Message Descriptor . . . 5-32
5.2.4.6 Buffer Descriptor . . . 5-32

5.2.4.7 Buffer Status Descriptor . . . 5-34
5.2.4.8 Next Message Pointer . . . 5-37
5.2.4.9 Data Buffer Pointer . . . 5-37
5.2.4.10 Message Descriptor Handling . . . 5-37
5.2.5 Interrupt Level Descriptors . . . 5-38
5.2.5.1 Interrupt Queue Descriptor . . . 5-38
5.2.5.2 Interrupt Descriptor . . . 5-39
5.2.5.3 Interrupt Status Descriptor . . . 5-44
5.2.6 Interrupt Handling . . . 5-45
5.2.6.1 Initialization . . . 5-45
5.2.6.2 Interrupt Descriptor Generation . . . 5-45
5.2.6.3 INTA* Signal Line . . . 5-46
5.2.6.4 INTB* Signal Line . . . 5-46

6.0 Basic Operation . . . 6-1

6.1 Reset . . . 6-1
6.1.1 Hard PCI Reset . . . 6-1
6.1.2 Soft Chip Reset . . . 6-2
6.1.3 Soft Group Reset . . . 6-3
6.1.4 Initialization Sequence Example . . . 6-3

6.2 Configuration . . . 6-4
6.2.1 PCI Configuration . . . 6-4
6.2.2 Global Configuration . . . 6-5
6.2.3 Interrupt Queue Configuration . . . 6-5
6.2.4 Channel Group(s) Configuration . . . 6-5
6.2.5 Service Request Mechanism . . . 6-5
6.2.6 MUSYCC Internal Memory . . . 6-6
6.2.6.1 Memory Operations—Inactive Channels . . . 6-6
6.2.6.2 Memory Operations—Active Channels . . . 6-6

6.3 Channel Operation . . . 6-8
6.3.1 Group Structure . . . 6-8
6.3.2 Group Base Pointer . . . 6-10
6.3.3 Global Configuration Descriptor . . . 6-11
6.3.4 Interrupt Queue Descriptor . . . 6-12
6.3.5 Group Configuration Descriptor . . . 6-13
6.3.6 Memory Protection Descriptor . . . 6-14
6.3.7 Port Configuration Descriptor . . . 6-14
6.3.8 Message Length Descriptor . . . 6-15
6.3.9 Transmit Time Slot Map—Channel 0 . . . 6-15
6.3.10 Transmit Subchannel Map . . . 6-16
6.3.11 Transmit Channel Configuration Descriptor . . . 6-17
6.3.12 Receive Time Slot Map . . . 6-19
6.3.13 Receive Subchannel Map . . . 6-19
6.3.14 Receive Channel Configuration Descriptor . . . 6-19
6.3.15 Message Lists . . . 6-19

6.3.16 Channel Activation . . . 6-21
6.3.16.1 Transmit Channel Activation . . . 6-21
6.3.16.2 Receive Channel Activation . . . 6-22
6.3.17 Channel Deactivation . . . 6-23
6.3.17.1 Transmit Channel Deactivation . . . 6-23
6.3.17.2 Receive Channel Deactivation . . . 6-23
6.3.18 Channel Jump . . . 6-24
6.3.19 Frame Alignment . . . 6-24
6.3.20 Descriptor Polling . . . 6-25
6.3.21 Repeat Message Transmission . . . 6-26
6.4 Protocol Support . . . 6-28
6.4.1 Frame Check Sequence . . . 6-28
6.4.2 Opening/Closing Flags . . . 6-28
6.4.3 Abort Codes . . . 6-28
6.4.4 Zero-Bit Insertion/Deletion . . . 6-29
6.4.5 Message Configuration Bits . . . 6-29
6.4.5.1 Idle Code . . . 6-29
6.4.5.2 Inter-message Pad Fill . . . 6-29
6.4.5.3 Repeat Message Transmission . . . 6-29
6.4.6 Message Configuration Bits Copy Enable/Disable . . . 6-29
6.4.7 Bit-Level Operation . . . 6-31
6.4.7.1 Transmit . . . 6-32
6.4.7.2 Receive . . . 6-32
6.4.8 HDLC Mode . . . 6-33
6.4.8.1 Transmit Events . . . 6-33
6.4.8.2 Receive Events . . . 6-34
6.4.8.3 Transmit Errors . . . 6-36
6.4.8.4 Receive Errors . . . 6-37
6.4.9 Transparent Mode . . . 6-43
6.4.9.1 Transmit Events . . . 6-44
6.4.9.2 Receive Events . . . 6-44
6.4.9.3 Transmit Errors . . . 6-45
6.4.9.4 Receive Errors . . . 6-46
6.4.10 Intersystem Link Protocol (ISLP) . . . 6-48
6.5 Signaling System 7 . . . 6-49
6.5.1 SS7 Repeat Message Transmission . . . 6-49
6.5.2 Message Filtering . . . 6-49
6.5.3 Signal Unit Error Rate Monitoring . . . 6-50
6.5.4 SUERM Counter Incrementing . . . 6-50
6.5.5 SUERM Octet Counting . . . 6-50
6.5.6 SUERM Counter Decrementing . . . 6-50
6.6 Self-Servicing Buffers . . . 6-51

7.0 Electrical and Mechanical Specifications . . . 7-1

7.1 Electrical and Environmental Specifications . . . 7-1

7.1.1 Absolute Maximum Ratings . . . 7-1

7.1.2 Recommended Operating Conditions . . . 7-2

7.1.3 Electrical Characteristics . . . 7-2

7.2 Timing and Switching Specifications . . . 7-3

7.2.1 Overview . . . 7-3

7.2.2 Host Interface (PCI) Timing and Switching Characteristic . . . 7-3

7.2.3 Expansion Bus (EBUS) Timing and Switching Characteristic . . . 7-11

7.2.4 EBUS Arbitration Timing . . . 7-14

7.2.5 Serial Interface Timing and Switching Characteristics . . . 7-16

7.2.6 Package Thermal Specification . . . 7-18

7.2.7 Mechanical Specifications . . . 7-19

8.0 Terms, Definitions, and Conventions . . . 8-1

8.1 Applicable Specifications . . . 8-1

8.2 Numeric Notation . . . 8-2

8.3 Bit Stream Transmission Convention . . . 8-3

8.4 Bit Stream Storage Convention . . . 8-4

8.5 Acronyms . . . 8-5

8.6 Definitions . . . 8-7

8.7 Revision History . . . 8-9

Appendix A JTAG Interface . . . A-1

A.1 Instruction Register . . . A-1

A.2 BYPASS Register . . . A-2

List of Figures

Figure 1-1. System Block Diagram . . . 1-1
Figure 1-2. Detailed System Block Diagram . . . 1-3
Figure 1-3. MUSYCC Application Example—Frame Relay Switch . . . 1-4
Figure 1-4. CN8478 MQFP Pinout Configuration . . . 1-5
Figure 1-5. CN8474A MPQF Pinout Configuration . . . 1-8
Figure 1-6. CN8472A MQFP Pinout Configuration . . . 1-9
Figure 1-7. CN8471A MQFP Pinout Configuration . . . 1-10
Figure 1-8. CN8478 PBGA Pinout Configuration (Top View) . . . 1-11
Figure 1-9. CN8474A PBGA Pinout Configuration (Top View) . . . 1-14
Figure 1-10. CN8472A PBGA Pinout Configuration (Top View) . . . 1-15
Figure 1-11. CN8471A PBGA Pinout Configuration (Top View) . . . 1-16
Figure 1-12. CN8478 Logic Diagram . . . 1-17
Figure 2-1. Host Interface Functional Block Diagram . . . 2-1
Figure 2-2. Address Lines During Configuration Cycle . . . 2-4
Figure 3-1. EBUS Functional Block Diagram with Local MPU . . . 3-1
Figure 3-2. EBUS Functional Block Diagram without Local MPU . . . 3-1
Figure 3-3. EBUS Address/Data Line Structure . . . 3-3
Figure 3-4. EBUS Connection, Non-multiplexed Address/Data, 8 Framers, No MPU . . . 3-8
Figure 3-5. EBUS Connection, Non-multiplexed Address/Data, 61 Framers, No MPU . . . 3-9
Figure 3-6. EBUS Connection, Multiplexed Address/Data, 8 Framers, No MPU . . . 3-10
Figure 4-1. Serial Interface Functional Block Diagram, Channel Group 0 . . . 4-1
Figure 4-2. Transmit and Receive T1 Mode . . . 4-5
Figure 4-3. Transmit and Receive E1 (also 2xE1, 4xE1) Mode . . . 4-6
Figure 4-4. Transmit and Receive Nx64 Mode . . . 4-7
Figure 4-5. Serial Port Mapping Options . . . 4-9
Figure 4-6. Receive Data Flow . . . 4-11
Figure 4-7. Transmit Data Flow . . . 4-12
Figure 4-8. Transmit Data Bit Output Value Determination . . . 4-16
Figure 5-1. Shared Memory Model Per Channel Group . . . 5-2
Figure 5-2. Interrupt Notification To Host . . . 5-47
Figure 7-1. PCI Clock (PCLK) Waveform . . . 7-5
Figure 7-2. PCI Reset Timing . . . 7-6
Figure 7-3. PCI Output Timing Waveform . . . 7-8
Figure 7-4. PCI Input Timing Waveform . . . 7-9
Figure 7-5. PCI Read Multiple Operation . . . 7-9
Figure 7-6. PCI Write Multiple Operation . . . 7-10
Figure 7-7. PCI Write Single Operation . . . 7-10
Figure 7-8. ECLK to PCLK Relationship (M66EN = 0) . . . 7-11
Figure 7-9. ECLK to PCKL Relationship (M66EN = 1) . . . 7-11

Figure 7-10. EBUS Reset Timing 7-12
Figure 7-11. EBUS Output Timing Waveform 7-13
Figure 7-12. EBUS Input Timing Waveform 7-13
Figure 7-13. EBUS Write/Read Transactions, Intel-Style 7-14
Figure 7-14. EBUS Write/Read Transactions, Motorola-Style 7-15
Figure 7-15. Serial Interface Clock (RCLK,TCLK) Waveform 7-16
Figure 7-16. Serial Interface Clock (RCLK,TCLK) Waveform 7-17
Figure 7-17. Serial Interface Data Input Waveform 7-17
Figure 7-18. Serial Interface Data Delay Output Waveform 7-18
Figure 7-19. 208-Pin Metric Quad Flatpack (MQFP) 7-19
Figure 7-20. 208-Pin Plastic Ball Grid Array (PBGA) 7-20
Figure A-1. JTAG Timing Diagram A-3

List of Tables

Table 1-1. CN8478 MQFP Pin List . . . 1-6
Table 1-2. CN8478 PBGA Pin List . . . 1-12
Table 1-3. I/O Pin Types . . . 1-18
Table 1-4. CN8478 Hardware Signal Definitions . . . 1-19
Table 2-1. Function 0 Configuration Space . . . 2-5
Table 2-2. Function 1 Configuration Space . . . 2-5
Table 2-3. Register 0, Address 00h . . . 2-7
Table 2-4. Register 1, Address 04h . . . 2-8
Table 2-5. Register 2, Address 08h . . . 2-10
Table 2-6. Register 3, Address 0Ch . . . 2-11
Table 2-7. Register 4, Address 10h . . . 2-12
Table 2-8. Registers 5–14, Addresses 14h–38h . . . 2-12
Table 2-9. Register 15, Address 3Ch . . . 2-13
Table 2-10. Register 0, Address 00h . . . 2-14
Table 2-11. Register 1, Address 04h . . . 2-14
Table 2-12. Register 2, Address 08h . . . 2-15
Table 2-13. Register 3, Address 0Ch . . . 2-16
Table 2-14. Register 4, Address 10h . . . 2-16
Table 2-15. Registers 5 through 14–Addresses 14h through 38h . . . 2-16
Table 2-16. Register 15, Address 3Ch . . . 2-17
Table 2-17. PCI Latency Example . . . 2-20
Table 3-1. Intel Protocol Signals . . . 3-6
Table 3-2. Motorola Protocol Signal . . . 3-7
Table 4-1. Channelized Serial Port Modes . . . 4-3
Table 4-2. Internal Buffer Memory Layout . . . 4-11
Table 4-3. Example of 32-Channel with Subchanneling Buffer Allocation (Receive or Transmit) . . . 4-13
Table 4-4. Example of 32-Channel without Subchanneling Buffer Allocation (Receive or Transmit) . . . 4-14
Table 4-5. Example of 16-Channel without Subchanneling Buffer Allocation (Receive or Transmit) . . . 4-14
Table 5-1. MUSYCC Register Map . . . 5-4
Table 5-2. Group Structure Memory Map . . . 5-6
Table 5-3. MUSYCC PCI Function Memory Allocation . . . 5-6
Table 5-4. Shared Memory Allocation—Group Descriptors . . . 5-7
Table 5-5. Host Assigns Group Base Pointers . . . 5-7
Table 5-6. Global Configuration Descriptor . . . 5-10
Table 5-7. Dual Address Cycle Base Pointer . . . 5-12
Table 5-8. Group Base Pointer . . . 5-12
Table 5-9. Service Request Descriptor . . . 5-14
Table 5-10. Group Configuration Descriptor . . . 5-16
Table 5-11. Memory Protection Descriptor . . . 5-18

Table 5-12. Port Configuration Descriptor . . . 5-19
Table 5-13. Message Length Descriptor . . . 5-20
Table 5-14. Transmit or Receive Time Slot Map . . . 5-22
Table 5-15. Time Slot Descriptor . . . 5-23
Table 5-16. Transmit or Receive Subchannel Map . . . 5-25
Table 5-17. Subchannel Descriptor . . . 5-25
Table 5-18. Channel Configuration Descriptor . . . 5-26
Table 5-19. Message Descriptor . . . 5-30
Table 5-20. Head Pointer . . . 5-31
Table 5-21. Message Pointer . . . 5-31
Table 5-22. Transmit Buffer Descriptor . . . 5-33
Table 5-23. Receive Buffer Descriptor . . . 5-34
Table 5-24. Transmit Buffer Status Descriptor . . . 5-35
Table 5-25. Receive Buffer Status Descriptor . . . 5-36
Table 5-26. Next Descriptor Pointer . . . 5-37
Table 5-27. Data Buffer Pointer . . . 5-37
Table 5-28. Interrupt Queue Descriptor . . . 5-38
Table 5-29. Interrupt Queue Pointer . . . 5-38
Table 5-30. Interrupt Queue Length . . . 5-38
Table 5-31. Interrupt Descriptor . . . 5-41
Table 5-32. Interrupt Status Descriptor . . . 5-44
Table 6-1. Example—Components of Group Base Pointer . . . 6-10
Table 6-2. Example—Components of Global Configuration Descriptor . . . 6-11
Table 6-3. Example—Components of Interrupt Queue Descriptor . . . 6-12
Table 6-4. Example—Components of Group Configuration Descriptor . . . 6-13
Table 6-5. Example—Components of Memory Protection Descriptor . . . 6-14
Table 6-6. Example—Components of Port Configuration Descriptor . . . 6-14
Table 6-7. Example—Components of Message Length Descriptor . . . 6-15
Table 6-8. Example—Components of Transmit Time Slot Map – Channel 0 . . . 6-16
Table 6-9. Example—Components of Transmit Subchannel Map . . . 6-17
Table 6-10. Example—Components of Channel Configuration Descriptor . . . 6-18
Table 6-11. Polling Frequency Using a Time Slot Counter Method . . . 6-26
Table 6-12. Memory Map for Message Configuration Descriptor Table . . . 6-30
Table 6-13. Message Configuration Descriptor . . . 6-30
Table 7-1. Absolute Maximum Ratings . . . 7-1
Table 7-2. Recommended Operating Conditions . . . 7-2
Table 7-3. Electrical Operating Characteristics . . . 7-2
Table 7-4. PCI Interface DC Specifications . . . 7-3
Table 7-5. PCI Clock (PCLK) Waveform Parameters, 33 MHz PCI Clock . . . 7-4
Table 7-6. PCI Clock (PCLK) Waveform Parameters, 66 MHz PCI Clock . . . 7-5
Table 7-7. PCI Reset Parameters . . . 7-6
Table 7-8. PCI I/O Timing Parameters, 33 MHz PCI Clock . . . 7-7
Table 7-9. PCI I/O Timing Parameters, 66 MHz PCI Clock . . . 7-7
Table 7-10. PCI I/O Measure Conditions . . . 7-8
Table 7-11. EBUS Reset Parameters . . . 7-11
Table 7-12. EBUS I/O Timing Parameters . . . 7-12

Table 7-13. EBUS I/O Measure Conditions 7-13
Table 7-14. Serial Interface Clock (RCLK, TCLK) Parameters 7-16
Table 7-15. Serial Interface I/O Timing Parameters 7-16
Table 7-16. Serial Interface Clock Hysteresis (RCLK, TCLK, with Schmitt Trigger) 7-16
Table 7-17. Serial Interface I/O Measure Conditions for 3.3 V Signaling 7-17
Table 7-18. MUSYCC Package Thermal Resistance Characteristics 7-18
Table 8-1. Number Representation 8-2
Table 8-2. Digitized Voice Transmission Convention 8-3
Table 8-3. Digital Data Transmission Convention 8-3
Table 8-4. MUSYCC Byte Transmission Convention 8-3
Table 8-5. Little-Endian Storage Convention (Intel-style) 8-4
Table 8-6. Big-Endian Storage Convention (Motorola-style) 8-4
Table 8-7. CN8478 Data Sheet Revisions 8-9
Table A-1. IEEE Std. 1149.1 Instructions A-1
Table A-2. JTAG Timing Table A-2

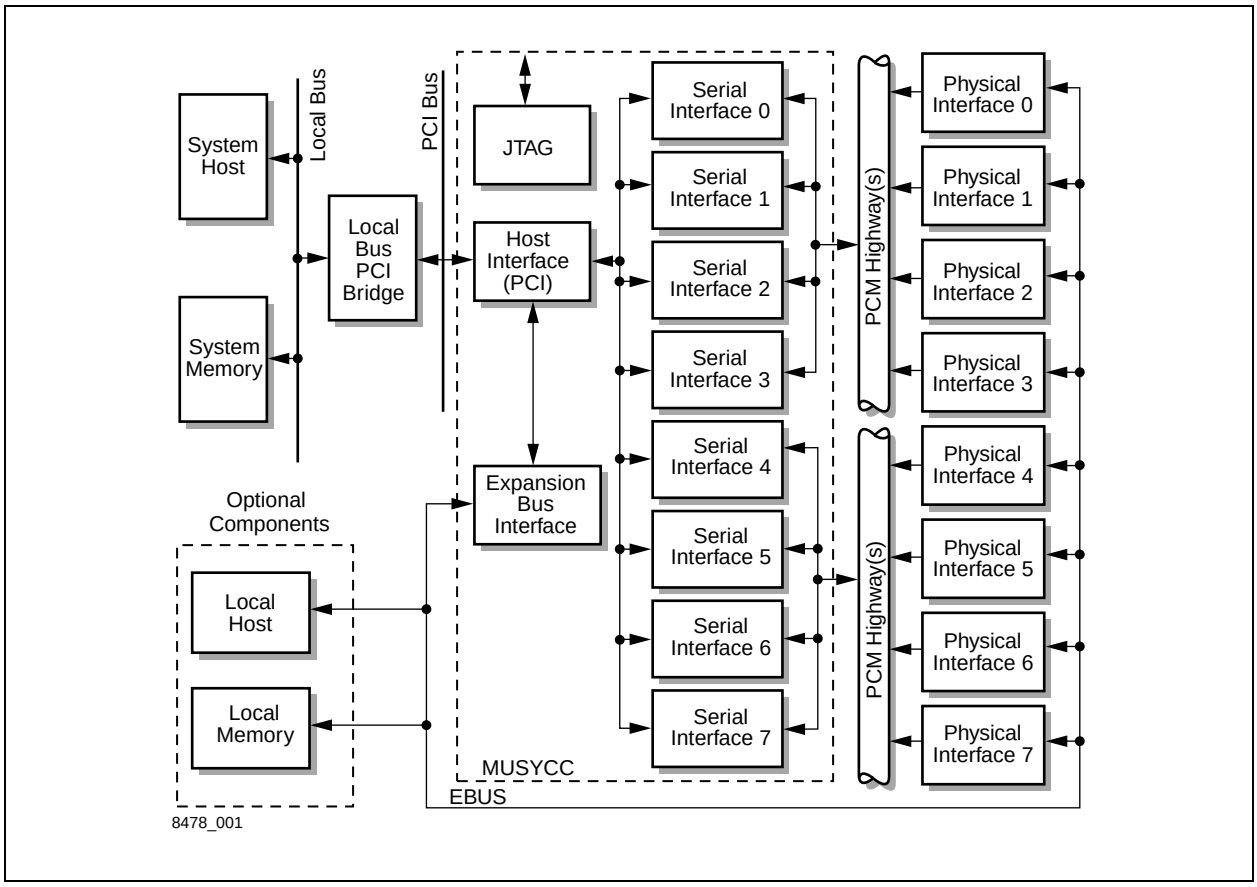
1.0 System Description

The Conexant MUSYCC is a high-throughput communications controller for synchronous, link-layer applications that multiplexes and demultiplexes up to 256 data channels. Each channel can be configured to support HDLC, Transparent, or SS7 applications. MUSYCC operates at the Layer 2 (the data link protocol level) reference of the International Organization for Standardization (ISO) Open Systems Interconnection (OSI). MUSYCC is installed between the multiple serial interface devices and the shared buffer memory of one or more host processors.

MUSYCC's serial ports interface to a standard Pulse Code Modulation (PCM) highway, which operates at T1, E1, 2xE1, or 4xE1 rates. Data can be formatted in the HDLC protocol or left unformatted. The protocol is specified on a per-channel and direction basis.

An on-device Peripheral Components Interface (PCI) controller, known as the host interface, is provided. Access to MUSYCC is available through PCI read, write, and configuration cycles (see Figure 1-1).

Figure 1-1. System Block Diagram



MUSYCC also provides an on-device, 32-bit local expansion bus (EBUS) controller which allows a host processor to access local memory and physical interface devices directly through MUSYCC over the PCI using configurable memory mapping features.

MUSYCC manages buffer memory for each active data channel with common list-processing structures. The on-device features allow data transmission between buffer memory and the serial interfaces with minimum host processor intervention. This allows the host processor to concentrate on managing the higher layers of the protocol stack.

Figures 1-2 and 1-3 illustrate detailed system block diagrams and a sample application.

Figure 1-2. Detailed System Block Diagram

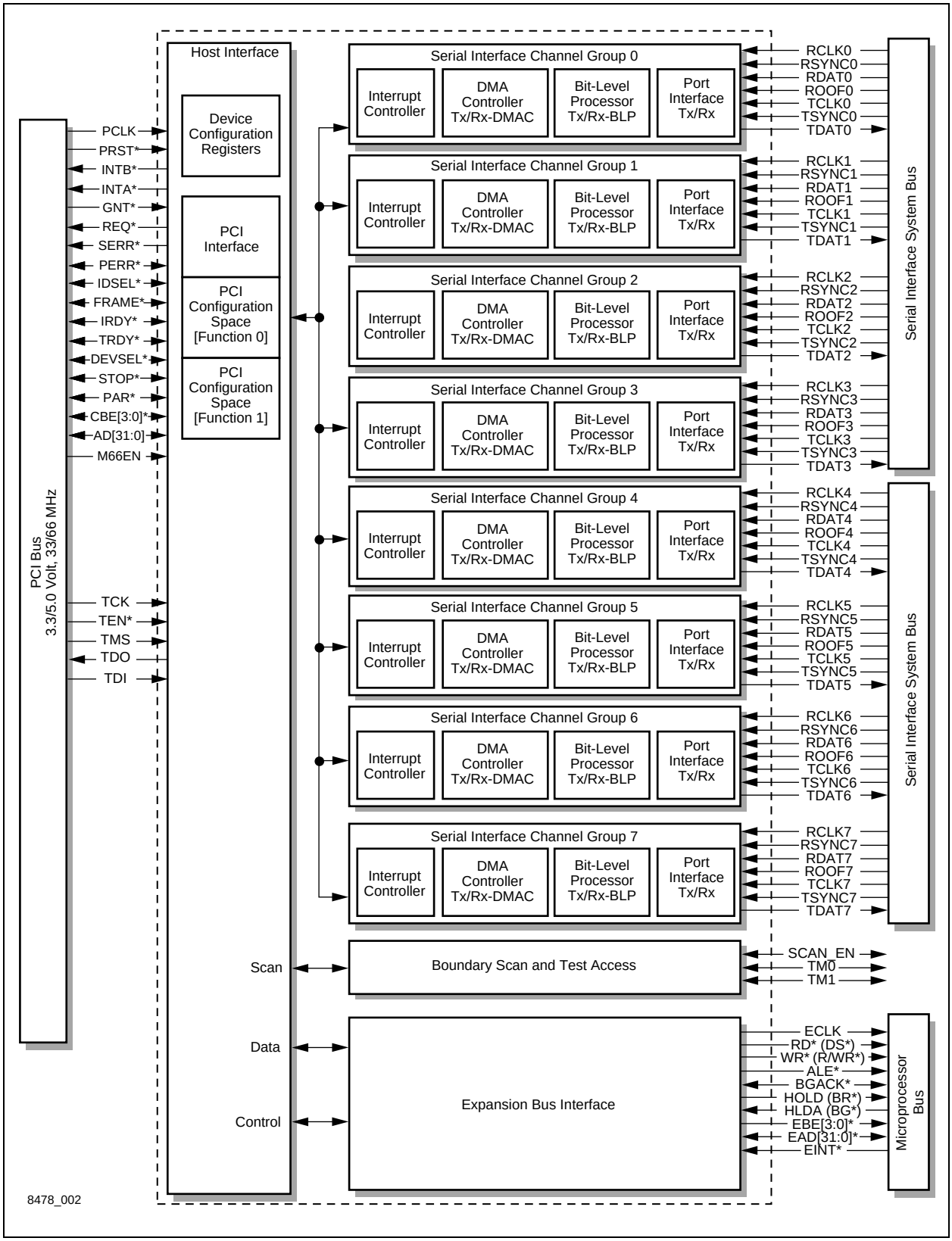
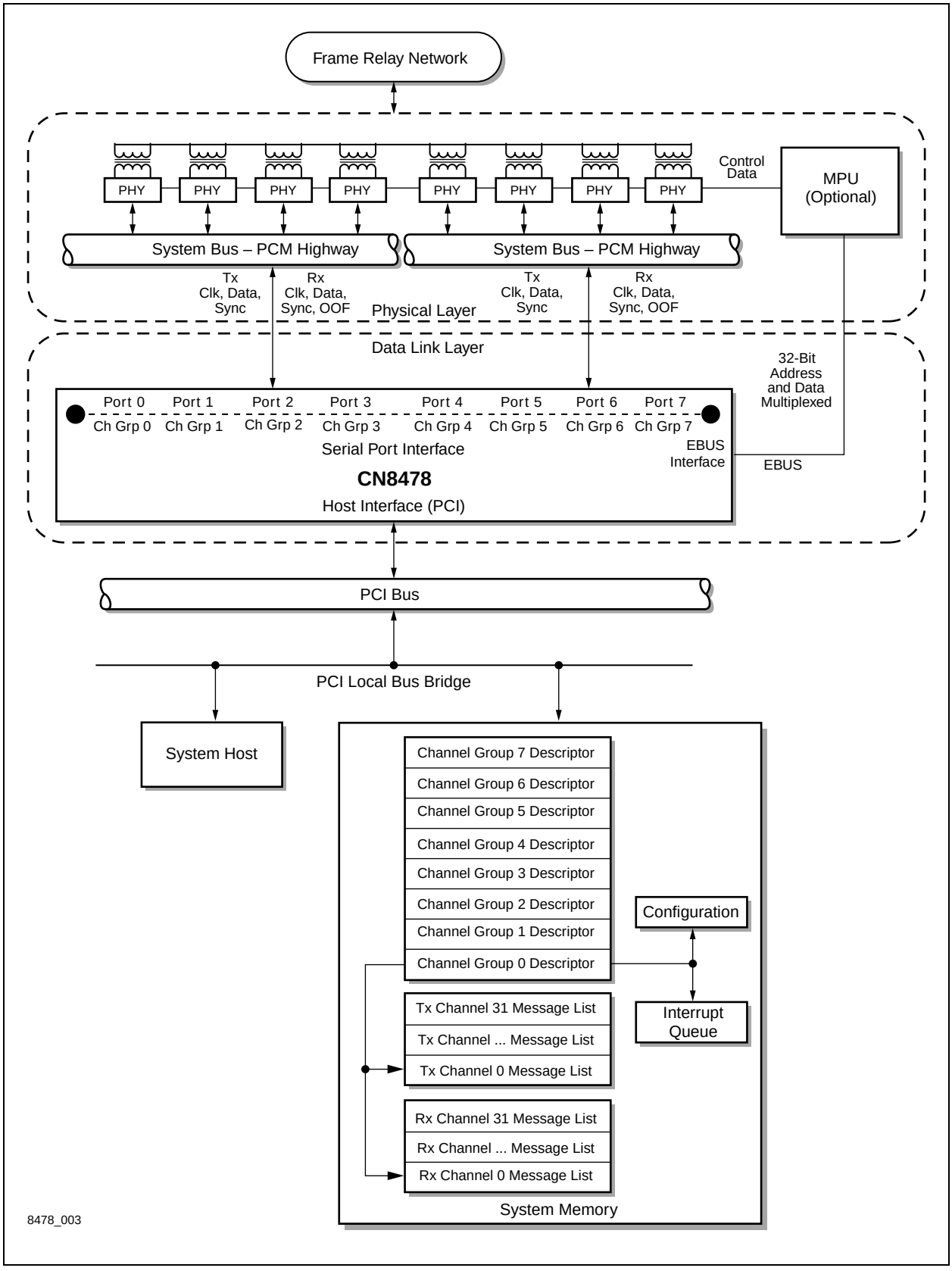


Figure 1-3. MUSYCC Application Example—Frame Relay Switch



1.1 Pin Descriptions

Figures 1-4 through 1-7 illustrate the pinouts for CN8478, CN8474A, CN8472A, and CN8471A. Signals marked with black are NCs. Tables 1-1 and 1-2 summarize the pin assignments for the CN8478 in the MQFP and PBGA packages, respectively. Table 1-3 lists the pin input and output functions. Table 1-4 lists the hardware signal definitions.

Figure 1-4. CN8478 MQFP Pinout Configuration

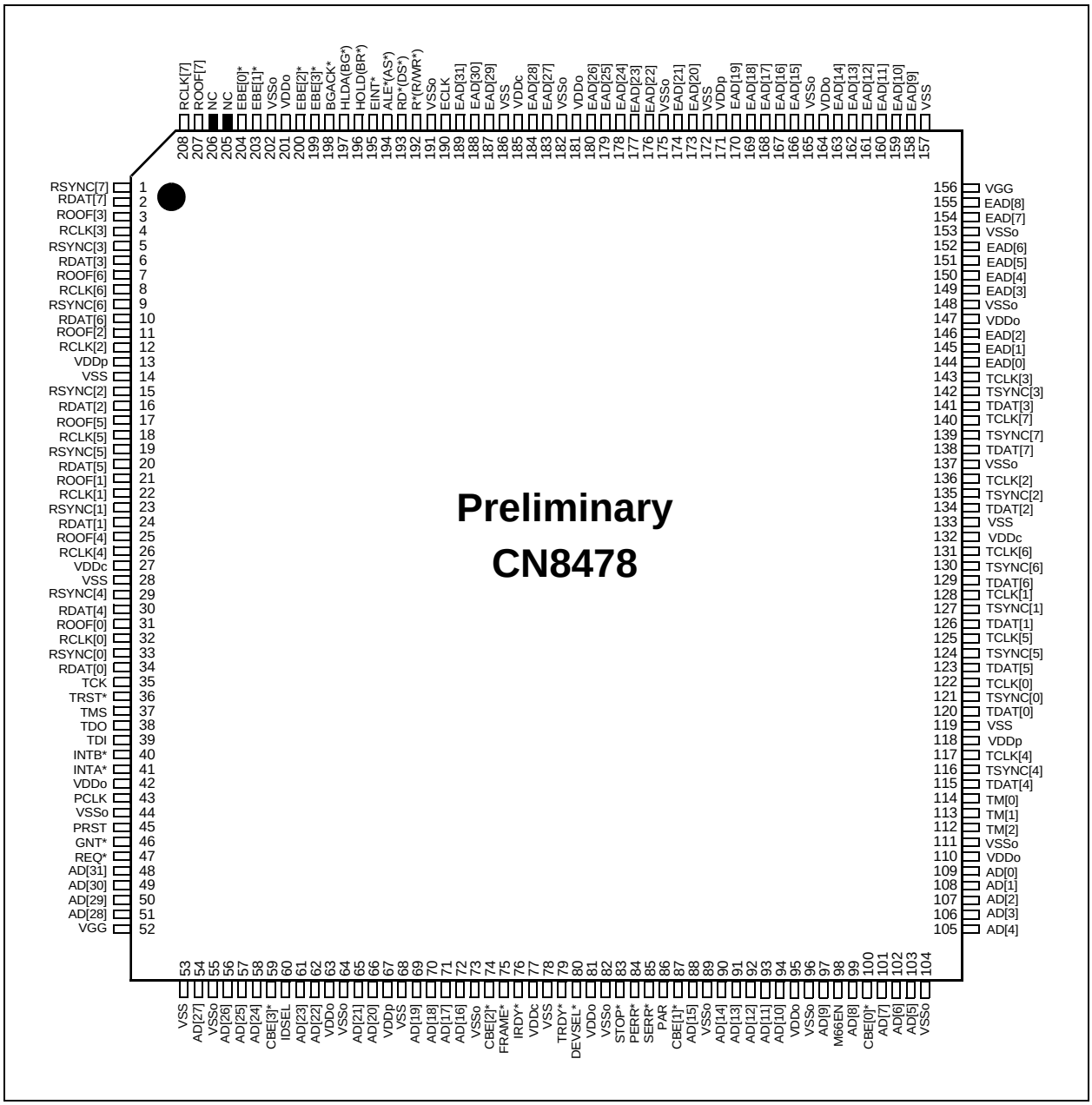


Table 1-1. CN8478 MQFP Pin List

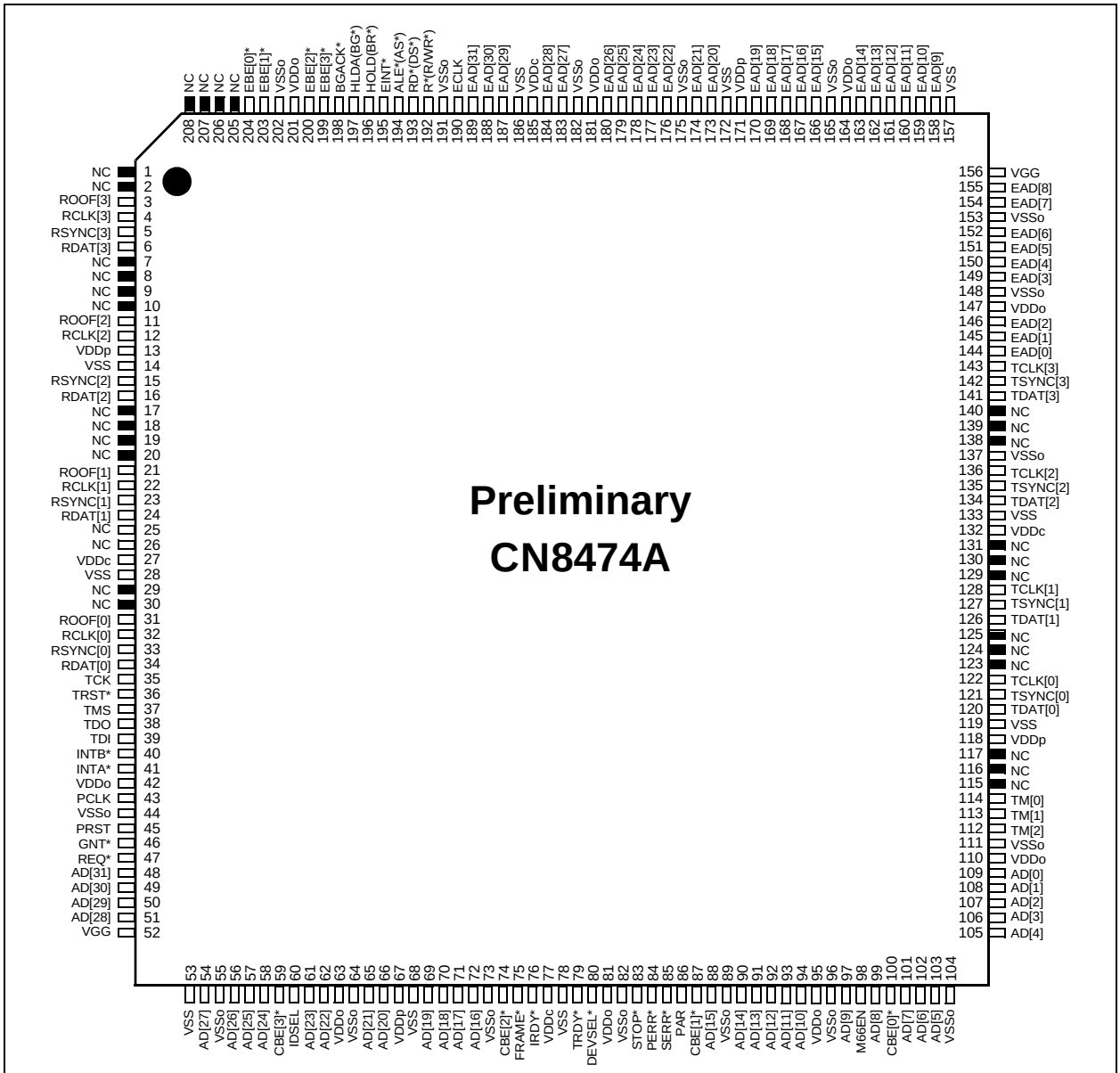
Pin Number	Pin Label
1	RSYNC[7]
2	RDAT[7]
3	ROOF[3]
4	RCLK[3]
5	RSYNC[3]
6	RDAT[3]
7	ROOF[6]
8	RCLK[6]
9	RSYNC[6]
10	RDAT[6]
11	ROOF[2]
12	RCLK[2]
13	VDDi
14	VSS
15	RSYNC[2]
16	RDAT[2]
17	ROOF[5]
18	RCLK[5]
19	RSYNC[5]
20	RDAT[5]
21	ROOF[1]
22	RCLK[1]
23	RSYNC[1]
24	RDAT[1]
25	ROOF[4]
26	RCLK[4]
27	VDDc
28	VSS
29	RSYNC[4]
30	RDAT[4]
31	ROOF[0]
32	RCLK[0]
33	RSYNC[0]
34	RDAT[0]
35	TCK
36	TRST*
37	TMS
38	TDO
39	TDI
40	INTB*
41	INTA*
42	VDDo
43	PCLK
44	VSSo
45	PRST*
46	GNT*
47	REQ*
48	AD[31]
49	AD[30]
50	AD[29]
51	AD[28]
52	VGG
53	VSS
54	AD[27]
55	VSSo
56	AD[26]
57	AD[25]
58	AD[24]
59	CBE[3]*
60	IDSEL
61	AD[23]
62	AD[22]
63	VDDo
64	VSSo
65	AD[21]
66	AD[20]
67	VDDi
68	VSS
69	AD[19]
70	AD[18]
71	AD[17]
72	AD[16]
73	VSSo
74	CBE[2]*
75	FRAME*
76	IRDY*
77	VDDc
78	VSS
79	TRDY*
80	DEVSEL*
81	VDDo
82	VSSo
83	STOP*
84	PERR*
85	SERR*
86	PAR
87	CBE[1]
88	AD[15]
89	VSSo
90	AD[14]
91	AD[13]
92	AD[12]
93	AD[11]
94	AD[10]
95	VDDo
96	VSSo
97	AD[9]
98	M66EN
99	AD[8]
100	CBE[0]*
101	AD[7]
102	AD[6]
103	AD[5]
104	VSSo
105	AD[4]

Pin Number	Pin Label
106	AD[3]
107	AD[2]
108	AD[1]
109	AD[0]
110	VDDo
111	VSSo
112	TM[2]
113	TM[1]
114	TM[0]
115	TDAT[4]
116	TSYNC[4]
117	TCLK[4]
118	VDDi
119	VSS
120	TDAT[0]
121	TSYNC[0]
122	TCLK[0]
123	TDAT[5]
124	TSYNC[5]
125	TCLK[5]
126	TDAT[1]
127	TSYNC[1]
128	TCLK[1]
129	TDAT[6]
130	TSYNC[6]
131	TCLK[6]
132	VDDc
133	VSS
134	TDAT[2]
135	TSYNC[2]
136	TCLK[2]
137	VSSo
138	TDAT[7]
139	TSYNC[7]
140	TCLK[7]
141	TDAT[3]

Pin Number	Pin Label
142	TSYNC[3]
143	TCLK[3]
144	EAD[0]
145	EAD[1]
146	EAD[2]
147	VDDo
148	VSSo
149	EAD[3]
150	EAD[4]
151	EAD[5]
152	EAD[6]
153	VSSo
154	EAD[7]
155	EAD[8]
156	VGG
157	VSS
158	EAD[9]
159	EAD[10]
160	EAD[11]
161	EAD[12]
162	EAD[13]
163	EAD[14]
164	VDDo
165	VSSo
166	EAD[15]
167	EAD[16]
168	EAD[17]
169	EAD[18]
170	EAD[19]
171	VDDi
172	VSS
173	EAD[20]
174	EAD[21]
175	VSSo
176	EAD[22]
177	EAD[23]

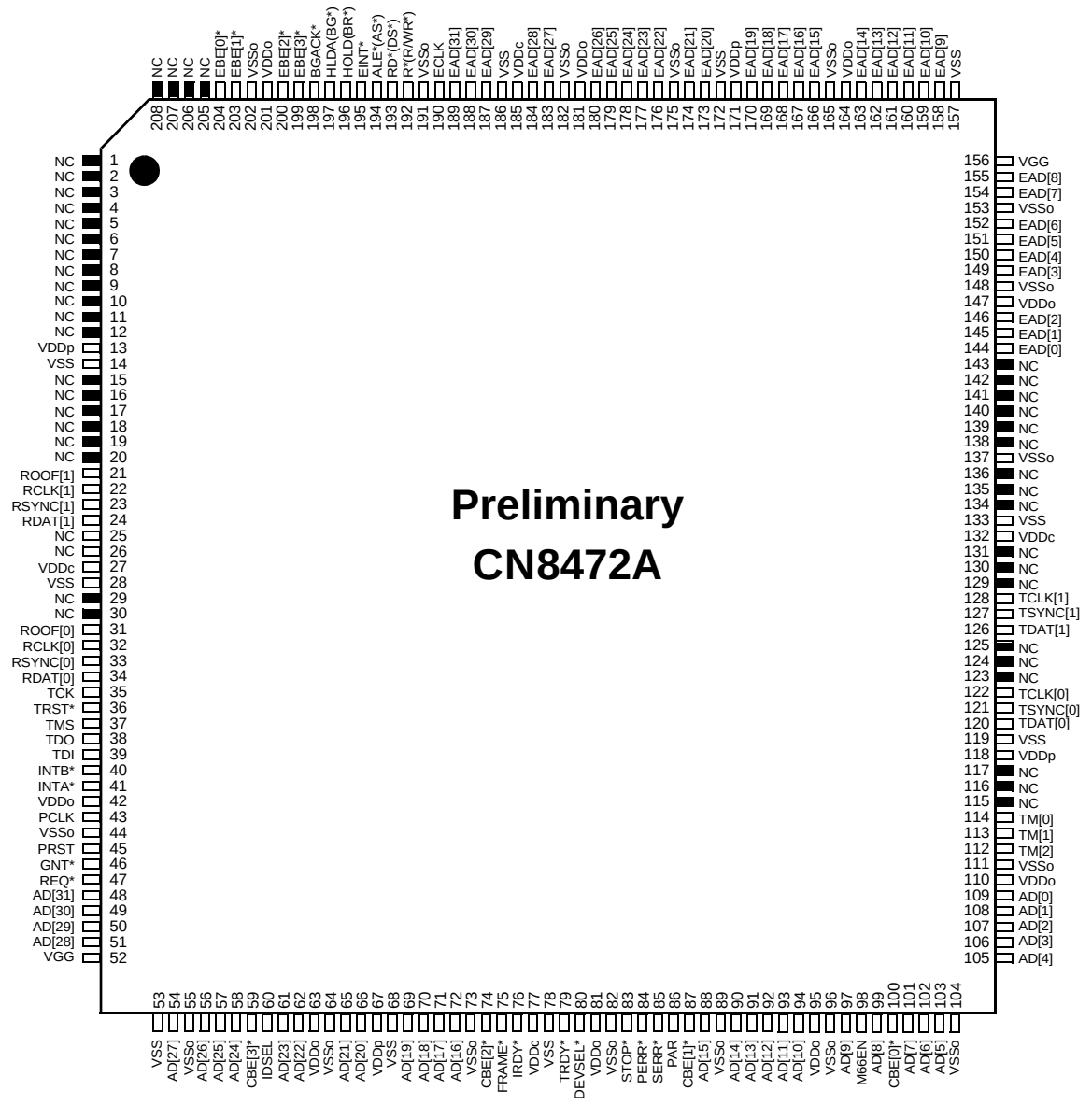
Pin Number	Pin Label
178	EAD[24]
179	EAD[25]
180	EAD[26]
181	VDDo
182	VSSo
183	EAD[27]
184	EAD[28]
185	VDDc
186	VSS
187	EAD[29]
188	EAD[30]
189	EAD[31]
190	ECLK
191	VSSo
192	WR*(R/ WR*)
193	RD*(DS*)
194	ALE*(AS*)
195	EINT*
196	HOLD(BR*)
197	HLDA(BG*)
198	BGACK*
199	EBE[3]*
200	EBE[2]*
201	VDDo
202	VSSo
203	EBE[1]*
204	EBE[0]*
205	NC
206	NC
207	ROOF[7]
208	RCLK[7]

Figure 1-5. CN8474A MPQF Pinout Configuration



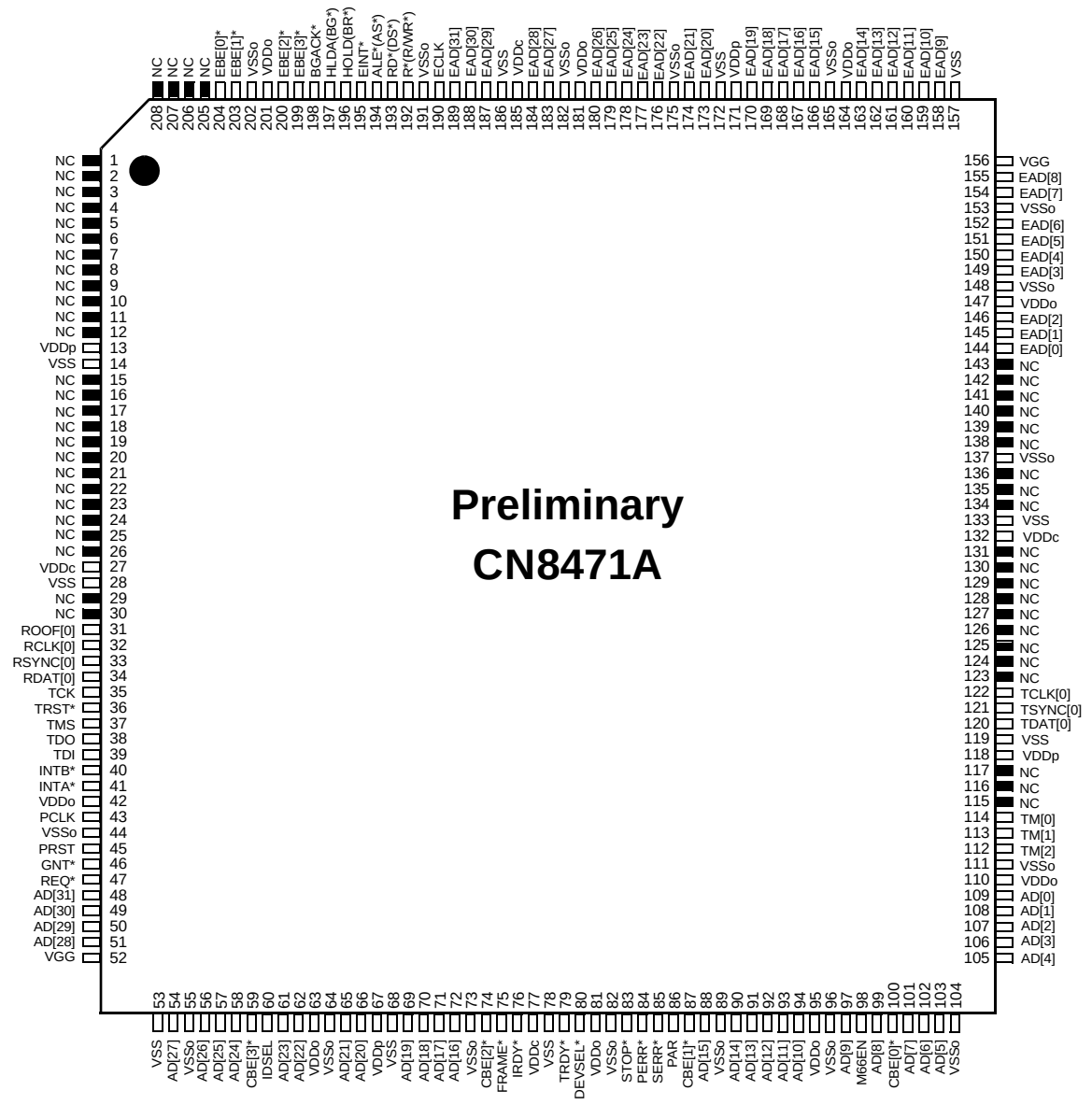
NOTE(S): An active low signal is denoted by a trailing asterisk (*).

Figure 1-6. CN8472A MQFP Pinout Configuration



NOTE(S): An active low signal is denoted by a trailing asterisk (*).

Figure 1-7. CN8471A MQFP Pinout Configuration



NOTE(S): An active low signal is denoted by a trailing asterisk(*).

Figure 1-8. CN8478 PBGA Pinout Configuration (Top View)

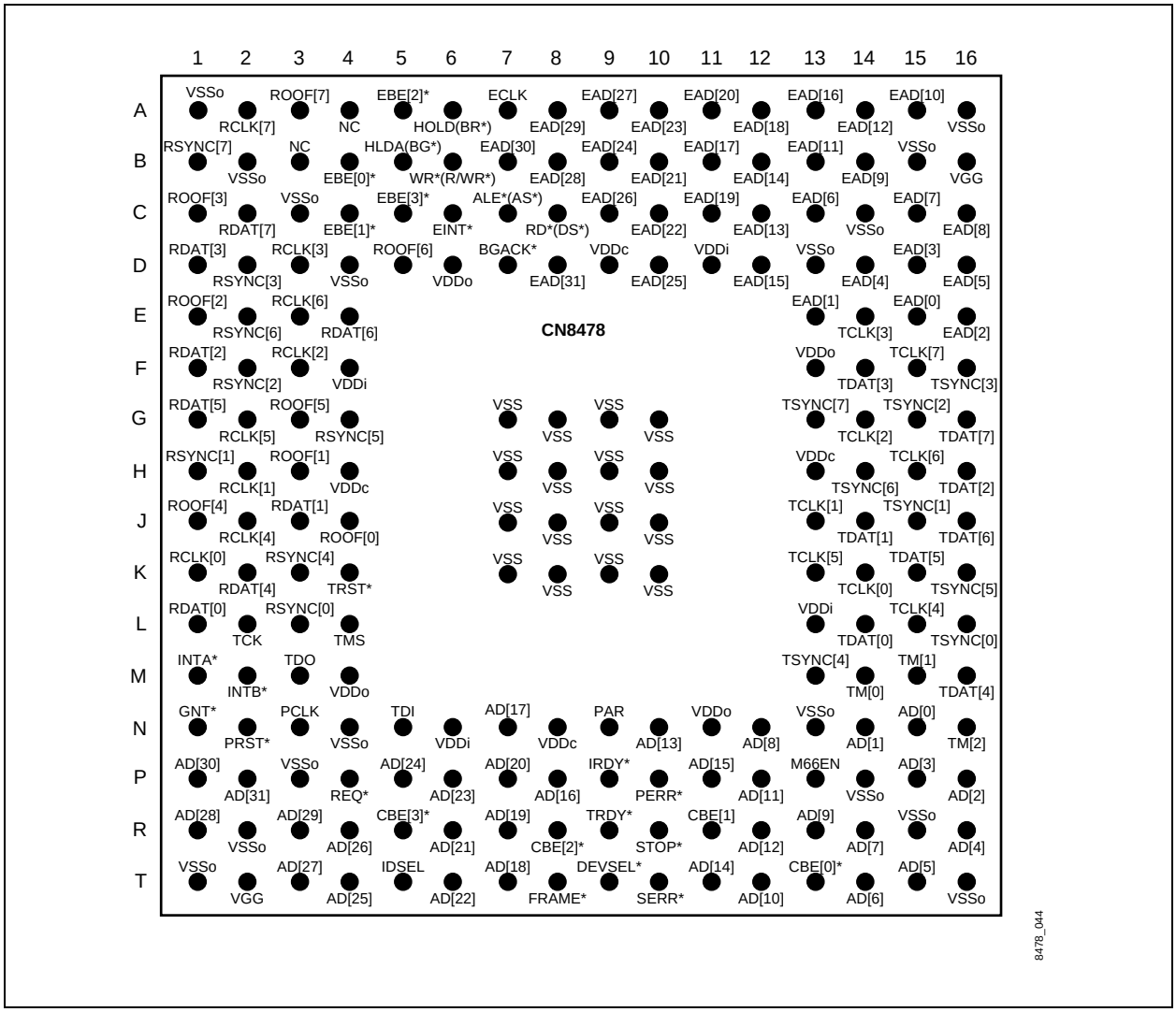


Table 1-2. CN8478 PBGA Pin List

Pin Number	Pin Label
A1	VSSo
A2	RCLK[7]
A3	ROOF[7]
A4	NC
A5	EBE[2]*
A6	HOLD(BR*)
A7	ECLK
A8	EAD[29]
A9	EAD[27]
A10	EAD[23]
A11	EAD[20]
A12	EAD[18]
A13	EAD[16]
A14	EAD[12]
A15	EAD[10]
A16	VSSo
B1	RSYNC[7]
B2	VSSo
B3	NC
B4	EBE[0]*
B5	HLDA(BG*)
B6	WR*(R/ WR*)
B7	EAD[30]
B8	EAD[28]
B9	EAD[24]
B10	EAD[21]
B11	EAD[17]
B12	EAD[14]
B13	EAD[11]
B14	EAD[9]
B15	VSSo
B16	VGG
C1	ROOF[3]
C2	RDAT[7]
C3	VSSo
C4	EBE[1]*
C5	EBE[3]*
C6	EINT*
C7	ALE*(AS*)
C8	RD*(DS*)
C9	EAD[26]
C10	EAD[22]
C11	EAD[19]
C12	EAD[13]
C13	EAD[6]
C14	VSSo
C15	EAD[7]
C16	EAD[8]
D1	RDAT[3]
D2	RSYNC[3]
D3	RCLK[3]
D4	VSSo
D5	ROOF[6]
D6	VDDo
D7	BGACK*
D8	EAD[31]
D9	VDDc
D10	EAD[25]
D11	VDDi
D12	EAD[15]
D13	VSSo
D14	EAD[4]
D15	EAD[3]
D16	EAD[5]
E1	ROOF[2]
E2	RSYNC[6]
E3	RCLK[6]
E4	RDAT[6]
E13	EAD[1]
E14	TCLK[3]
E15	EAD[0]
E16	EAD[2]
F1	RDAT[2]
F2	RSYNC[2]
F3	RCLK[2]
F4	VDDi
F13	VDDo
F14	TDAT[3]
F15	TCLK[7]
F16	TSYNC[3]
G1	RDAT[5]
G2	RCLK[5]
G3	ROOF[5]
G4	RSYNC[5]
G7	VSS
G8	VSS
G9	VSS
G10	VSS
G13	TSYNC[7]
G14	TCLK[2]
G15	TSYNC[2]
G16	TDAT[7]
H1	RSYNC[1]
H2	RCLK[1]
H3	ROOF[1]
H4	VDDc
H7	VSS
H8	VSS
H9	VSS
H10	VSS
H13	VDDc
H14	TSYNC[6]
H15	TCLK[6]
H16	TDAT[2]
J1	ROOF[4]

Pin Number	Pin Label
J2	RCLK[4]
J3	RDAT[1]
J4	ROOF[0]
J7	VSS
J8	VSS
J9	VSS
J10	VSS
J13	TCLK[1]
J14	TDAT[1]
J15	TSYNC[1]
J16	TDAT[6]
K1	RCLK[0]
K2	RDAT[4]
K3	RSYNC[4]
K4	TRST*
K7	VSS
K8	VSS
K9	VSS
K10	VSS
K13	TCLK[5]
K14	TCLK[0]
K15	TDAT[5]
K16	TSYNC[5]
L1	RDAT[0]
L2	TCK
L3	RSYNC[0]
L4	TMS
L13	VDDi
L14	TDAT[0]
L15	TCLK[4]
L16	TSYNC[0]
M1	INTA*
M2	INTB*
M3	TDO
M4	VDDo
M13	TSYNC[4]

Pin Number	Pin Label
M14	TM[0]
M15	TM[1]
M16	TDAT[4]
N1	GNT*
N2	PRST*
N3	PCLK
N4	VSSo
N5	TDI
N6	VDDi
N7	AD[17]
N8	VDDc
N9	PAR
N10	AD[13]
N11	VDDo
N12	AD[8]
N13	VSSo
N14	AD[1]
N15	AD[0]
N16	TM[2]
P1	AD[30]
P2	AD[31]
P3	VSSo
P4	REQ*
P5	AD[24]
P6	AD[23]
P7	AD[20]
P8	AD[16]
P9	IRDY*
P10	PERR*
P11	AD[15]
P12	AD[11]
P13	M66EN
P14	VSSo
P15	AD[3]
P16	AD[2]
R1	AD[28]

Pin Number	Pin Label
R2	VSSo
R3	AD[29]
R4	AD[26]
R5	CBE[3]*
R6	AD[21]
R7	AD[19]
R8	CBE[2]*
R9	TRDY*
R10	STOP*
R11	CBE[1]
R12	AD[12]
R13	AD[9]
R14	AD[7]
R15	VSSo
R16	AD[4]
T1	VSSo
T2	VGG
T3	AD[27]
T4	AD[25]
T5	IDSEL
T6	AD[22]
T7	AD[18]
T8	FRAME*
T9	DEVSEL*
T10	SERR*
T11	AD[14]
T12	AD[10]
T13	CBE[0]*
T14	AD[6]
T15	AD[5]
T16	VSSo

Figure 1-9. CN8474A PBGA Pinout Configuration (Top View)

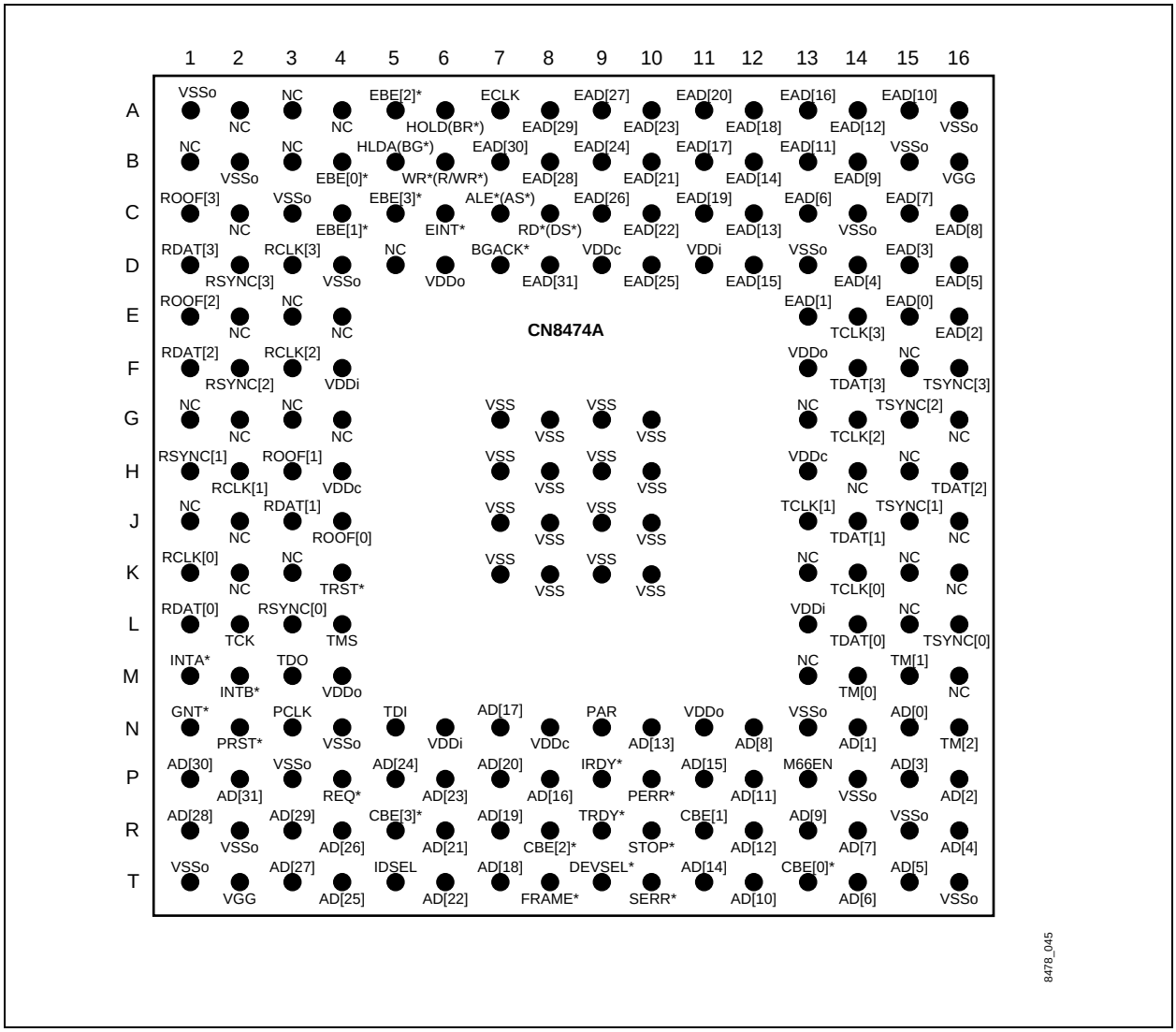


Figure 1-10. CN8472A PBGA Pinout Configuration (Top View)

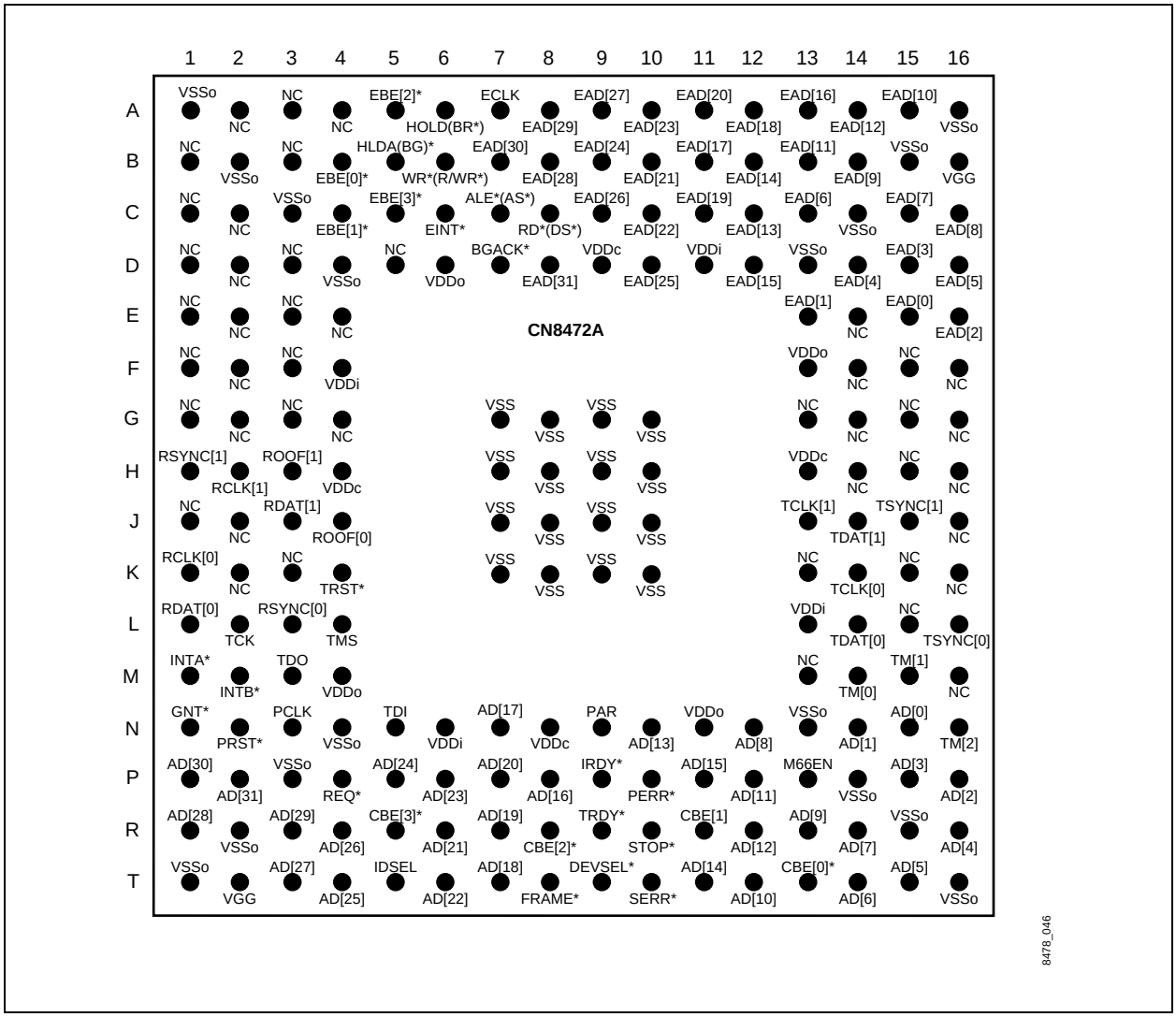


Figure 1-11. CN8471A PBGA Pinout Configuration (Top View)

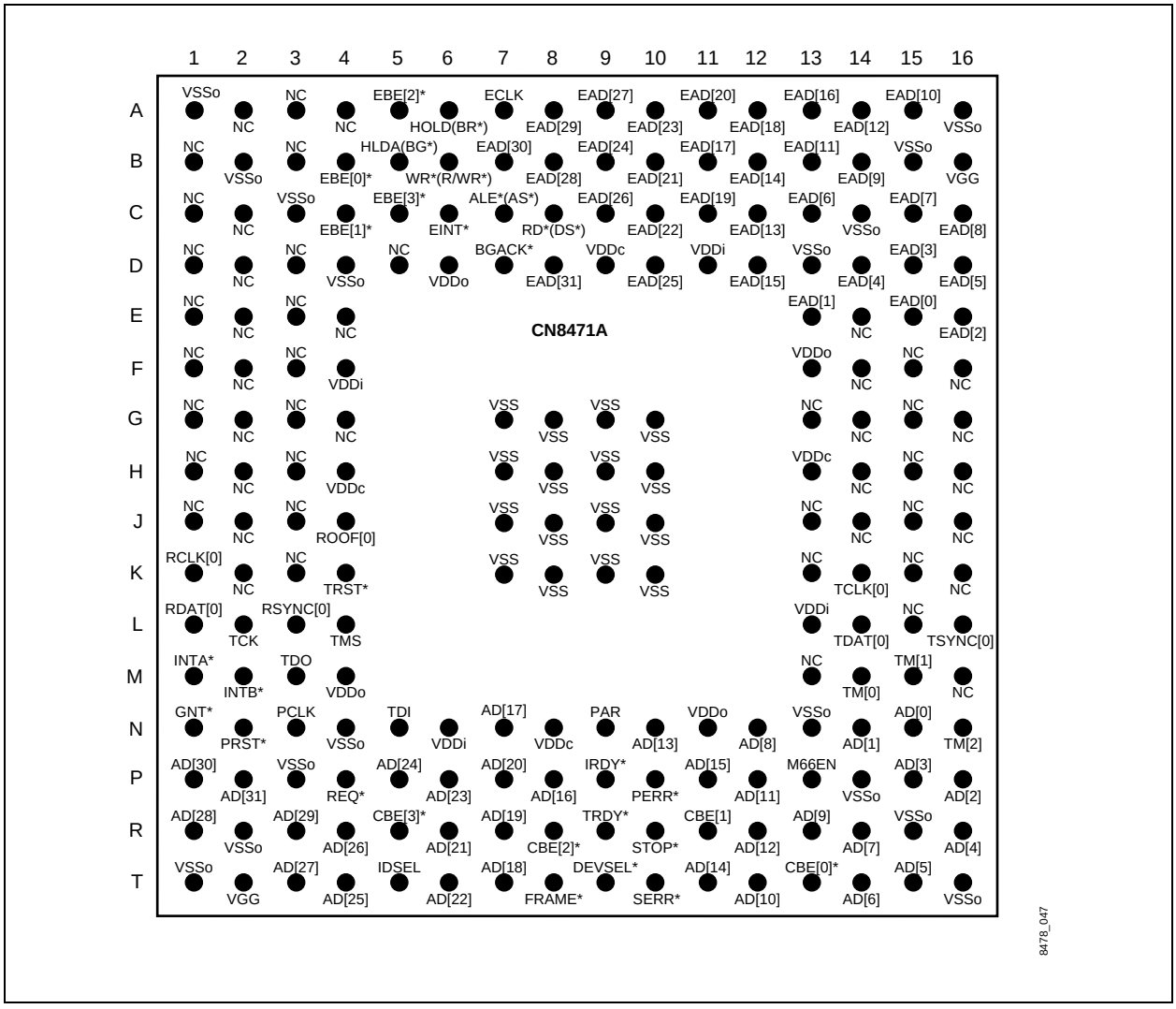
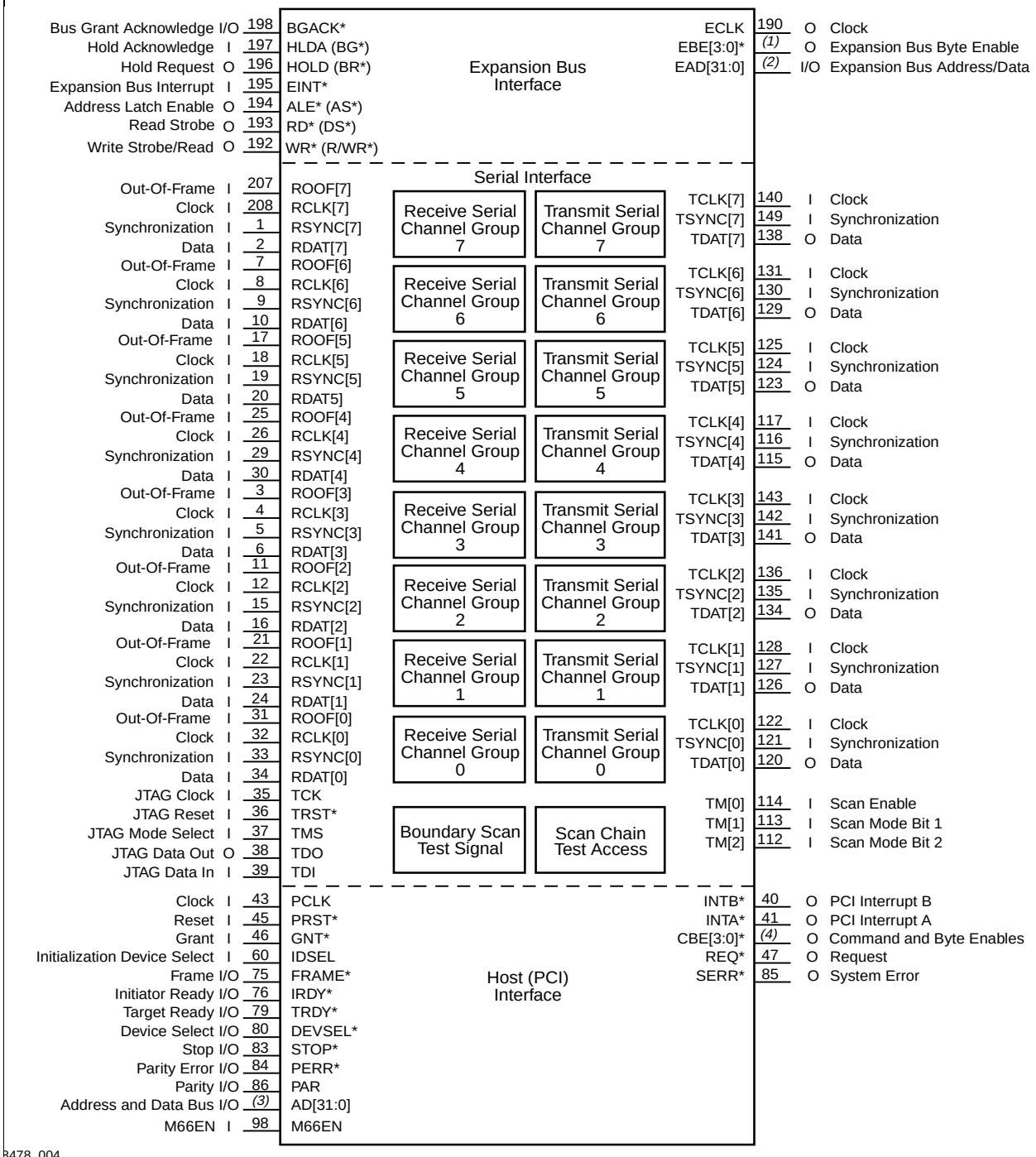


Figure 1-12. CN8478 Logic Diagram



NOTE(S):

(1) EBE [3:0]* pin numbers are 199-200, 203-204.

(2) EAD [31:0] pin numbers are 144-146, 149-152, 145-155, 158-163, 166-170, 173-174, 176-180, 183-184, 187-189.

(3) AD [31:0] pin numbers are 48-51, 54, 56-58, 61-62, 65-66, 69-72, 88, 90-94, 97, 99, 101-103, 105-109.

(4) CBS [3.0]* pin numbers are 59, 74, 87,100.

(5) An active low signal is denoted by a trailing asterisk (*).

Table 1-3. I/O Pin Types

I/O	Definition
I	Input. High impedance, TTL.
O	Output. CMOS.
I/O	Input/Output. TTL input/CMOS output.
t/s	Three-state. Bidirectional three-state I/O pin.
s/t/s	Sustained three-state. This is an active-low, three-state signal owned by only one driver at a time. The driver that drives an s/t/s signal low must drive it high for at least one clock cycle before allowing it to float. A pullup is required to sustain the deaserted value.
o/d	Open drain.
NOTE(S): All outputs are CMOS drive levels and can be used with CMOS or TTL logic.	

Table 1-4. CN8478 Hardware Signal Definitions *(1 of 6)*

	MQFP Pin No.	Pin Label	Signal Name	I/O	Definition
Expansion Bus Interface	190	ECLK	Expansion Bus Clock	t/s O	ECLK is an inverted version of the PCI clock applied at the PCLK input.
	144-146, 149-152, 154-155, 158-163, 166-170, 173-174, 176-180, 183-184, 187-189	EAD[31:0]	Expansion Bus Address and Data	t/s I/O	EAD[31:0] is a multiplexed address/data bus. During the address phase, pins EAD[17:0] contains meaningful address information. It is the same address as PCI AD[19:2] for the corresponding cycle. Pins EAD[31:18] are driven to 0 during the address phase. This is because those upper bits are compared, during the PCI address phase, to the value in the relocatable EBUS Base Address register to determine if the PCI cycle is in fact addressing into MUSYCC EBUS space. During data phase of an EBUS access cycle, the PCI signals AD[31:0] are transferred to the EBUS signal lines EAD[31:0] unaltered.
	199, 200, 203, 204	EBE[3:0]*	Expansion Bus Byte Enables	t/s O	EBE* contains the same information as the PCI byte enables but is driven in chip select style protocol used as active-low chip selects when MUSYCC is connected to more than one byte-wide device. All PCI accesses with byte lane 0's byte enable asserted would go to the byte-wide device connected to EAD[7:0]. Likewise, for byte lanes 1, 2, and 3 and EAD[15:8], EAD[23:16], and EAD[31:24], respectively. Only the CBE[3:0]* signals from the PCI data phase (byte-enable signals and not the command signals from the PCI address phase) are transferred to the EBE[3:0]* signal lines. EBE* is held high during all other phases of PCI access cycles.
	192	WR* (R/WR*)	Write Strobe	t/s O	High-to-low transition enables write data from MUSYCC into peripheral device. Rising edge defines write. (In Motorola mode, R/WR* is held high throughout read operation and held low throughout write operation. Determines meaning of DS* strobe.)
	193	RD* (DS*)	Read Strobe	t/s O	High-to-low transition enables read data from peripheral into MUSYCC. Held high throughout write operation. (In Motorola mode, DS* transitions low for both read and write operations and is held low throughout the operation.
	194	ALE* (AS*)	Address Latch Enable	t/s O	High-to-low transition indicates that EAD[31:0] bus contains valid address. Remains asserted low through the data phase of the EBUS access. (In Motorola mode, high-to-low transition indicates EBUS contains valid address. Remains asserted for the entire access cycle.)
	195	EINT*	Expansion Bus Interrupt	I	EINT* transfers interrupts from local devices to the PCI INTB* pin.
	196	HOLD (BR*)	Hold Request (Bus Request)	t/s O	When asserted, MUSYCC requests control of the EBUS.
	197	HLDA (BG*)	Hold Acknowledge (Bus Grant)	I	When asserted, MUSYCC has access to the EBUS. It is held asserted when there are no other masters connected to the bus, or asserted as a handshake mechanism to control EBUS arbitration.
	198	BGACK*	Bus Grant Acknowledge	t/s O	When asserted, MUSYCC acknowledges to the bus arbiter that the bus grant signal was detected and a bus cycle will be sustained by MUSYCC until this signal is deasserted.

***Table 1-4. CN8478 Hardware Signal Definitions** (2 of 6)*

	MQFP Pin No.	Pin Label	Signal Name	I/O	Definition
Serial Interface	117, 122, 125, 128, 131, 136, 140, 143	TCLK[7:0]	Transmit Clock (1)	I	Controls the rate at which data is transmitted. Synchronizes transitions for TDATx and sampling of TSYNCx. Valid frequencies from DC to 8.192 ±10% MHz. Schmitt trigger driver.
	116, 121, 124, 127, 130, 135, 139, 142	TSYNC[7:0]	Transmit Synchronization (1)	I	TSYNC is sampled on the specified active edge of the corresponding transmit clock, TCLKx. See TSYNC_EDGE bit field in Table 5-12. As TSYNCx signal transitions low-to-high, start of a transmit frame is indicated. For T1 mode, the corresponding data bit latched out during the same bit time period (but not necessarily the same clock edge) is the F-bit of the T1 frame. For E1 modes, the corresponding data bit latched out during the same bit time period (but not necessarily the same clock edge) is bit 0 of the E1 frame. For Nx64 mode, the corresponding data bit is latched out 4-bit time periods later and is bit 0 of the Nx64 frame. TSYNCx must remain asserted high for a minimum of a setup and hold time relative to the active clock edge for this signal. If the flywheel mechanism is used, no other synchronization signal is required, because MUSYCC tracks the start of each subsequent frame. If the flywheel mechanism is not used, then a subsequent low-to-high assertion is required to indicate the start of the next frame. See SFALIGN bit field in Table 5-10.
	115, 120, 123, 126, 129, 134, 138, 141	TDAT[7:0]	Transmit Data	t/s O	Serial data latched out on active edge of transmit clock, TCLKx. If channel is unmapped to time slot, data bit is considered invalid and MUSYCC outputs either three-state signal or logic 1 depending on value of bit field TRITX in Table 5-12.
	4, 8, 12, 18, 22, 26, 32, 208	RCLK[7:0]	Receive Clock (1)	I	Active edge samples RDATx and RSYNCx. Valid frequencies from DC to 8.192 ± 10% MHz. Schmitt trigger driver.
	1, 5, 9, 15, 19, 23, 29, 33	RSYNC[7:0]	Receive Synchronization (1)	I	RSYNCx is sampled on the specified active edge of the corresponding receive clock, RCLKx. See RSYNC_EDGE bit field in Table 5-12. As RSYNCx signal transitions low-to-high, start of a receive frame is indicated. For T1 mode, the corresponding data bit sampled and stored during the same bit time period (but not necessarily the same clock edge) is the F-bit of the T1 frame. For E1 modes, the corresponding data bit sampled and stored during the same bit time period (but not necessarily the same clock edge) is bit 0 of the E1 frame. For Nx64 mode, the corresponding data bit sampled and stored during the same bit time period (but not necessarily the same clock edge) is bit 0 of the Nx64 frame. RSYNCx must be asserted high for a minimum of a setup and hold time relative to the active clock edge for this signal. If the flywheel mechanism is used, no other synchronization signal is required, because MUSYCC tracks the start of each subsequent frame. If the flywheel mechanism is not used, a subsequent low-to-high assertion is required to indicate the start of the next frame. See SFALIGN bit field in Table 5-10.

Table 1-4. CN8478 Hardware Signal Definitions *(3 of 6)*

	MQFP Pin No.	Pin Label	Signal Name	I/O	Definition
Serial Interface (Continued)	2, 6, 10, 16, 20, 24, 30, 34	RDAT[7:0]	Receive Data *(1)*	I	Serial data sampled on active edge of receive clock, RCLKx. If channel is mapped to a time slot, input bit is sampled and transferred to memory. If channel is unmapped to time slot, data bit is considered invalid, and MUSYCC ignores received sample.
	3, 7, 11, 17, 21, 25, 31, 207	ROOF[7:0]	Receiver Out-of-Frame *(1)*	I	ROOFx is sampled on the specified active edge of the corresponding receive clock, RCLKx. See ROOF_EDGE bit field in Table 5-12. As ROOFx signal transitions from low to high, an Out-of-Frame (OOF) condition is indicated. As long as ROOFx is asserted, the received serial data is considered OOF. Depending on the state of OOFABT bit field in Table 5-10, receive bit processing may be disabled for the entire channel group (all channels disabled) while ROOFx remains asserted. Upon deassertion of ROOFx, bit-level processing resumes for all affected channels. If the flywheel mechanism is used, no other synchronization signal is required, because MUSYCC tracks the start of each subsequent frame during the OOF period. If the flywheel mechanism is not used, then a subsequent RSYNCx assertion is required to indicate the start of the next frame. See SFALIGN bit field in Table 5-10.

Table 1-4. CN8478 Hardware Signal Definitions *(4 of 6)*

	MQFP Pin No.	Pin Label	Signal Name	I/O	Definition
PCI Interface	48-51, 54, 56-58, 61, 65-66, 69-72, 88, 90-94, 97, 99, 101-103, 105-109	AD[31:0]	PCI Address and Data	t/s I/O	AD[31:0] is a multiplexed address/data bus. A PCI transaction consists of an address phase during the first clock period followed by one or more data phases. AD[7:0] is the LSB.
	43	PCLK	PCI Clock	I	PCLK provides timing for all PCI transitions. All PCI signals except PRST*, INTA*, and INTB* are synchronous to PCLK and are sampled on the rising edge of PCLK. MUSYCC supports a PCI clock up to 66 MHz.
	45	PRST*	PCI Reset	I	This input resets all functions on MUSYCC.
	59, 74, 87, 100	CBE[3:0]*	PCI Command and Byte Enables	t/s I/O	During the address phase, CBE[3:0]* contain command information; during the data phases, these pins contain information denoting which byte lanes are valid. PCI commands are defined as follows: CBE[3:0] — Command Type Oh 0000b Interrupt Acknowledge 1h 0001b Special Cycle 6h 0110b Memory Read 7h 0111b Memory Write Ah 1010b Configuration Read Bh 1011b Configuration Write Ch 1100b Memory Read Multiple Dh 1101b Dual Address Cycle Eh 1110b Memory Read Line Fh 1111b Memory Write and Invalidate
	86	PAR	PCI Parity	t/s I/O	The number of 1s on PAR, AD[31:0], and CBE[3:0]* is an even number. PAR always lags AD[31:0] and CBE* by one clock. During address phases, PAR is stable and valid one clock after the address; during the data phases it is stable and valid one clock after TRDY* on reads and one clock after IRDY* on writes. It remains valid until one clock after the completion of the data phase.
	75	FRAME*	PCI Frame	s/t/s I/O	FRAME* is driven by the current master to indicate the beginning and duration of a bus cycle. Data cycles continue as FRAME* stays asserted. The final data cycle is indicated by the deassertion of FRAME*. For a non-burst, one-data-cycle bus cycle, this pin is only asserted for the address phase.
	76	IRDY*	PCI Initiator Ready	s/t/s I/O	IRDY* asserted indicates the current master's readiness to complete the current data phase.
	79	TRDY*	PCI Target Ready	s/t/s I/O	TRDY* asserted indicates the target's readiness to complete the current data phase.
	83	STOP*	PCI Stop	s/t/s I/O	STOP* asserted indicates the selected target is requesting the master to stop the current transaction.

Table 1-4. CN8478 Hardware Signal Definitions *(5 of 6)*

	MQFP Pin No.	Pin Label	Signal Name	I/O	Definition
PCI Interface	80	DEVSEL*	PCI Device Select	s/t/s I/O	When asserted, DEVSEL* indicates that the driving device has decoded its address as the target of the current cycle.
	60	IDSEL	PCI Initialization Device Select	I	This input is used to select MUSYCC as the target for configuration read or write cycles.
	85	SERR*	System Error	o/d O	Any PCI device can assert SERR* to indicate a parity error on the address cycle or parity error on the data cycle of a special cycle command or any other system error where the result will be catastrophic. MUSYCC only asserts SERR* if it detects a parity error on the address cycle. Since SERR* is not an s/t/s signal, restoring it to the deasserted state is done with a weak pullup (same value as used for s/t/s). MUSYCC does not input SERR*. It is assumed that the host will reset MUSYCC in the case of a catastrophic system error.
	84	PERR*	Parity Error	s/t/s I/O	PERR* is asserted by the agent receiving data when it detects a parity error on a data phase. It is asserted one clock after PAR is driven, which is two clocks after the AD and CBE* parity was checked. MUSYCC generates the PERR Interrupt Descriptor toward the host under the following conditions: • MUSYCC masters a PCI cycle. • After supplying data during the data phase of the cycle, MUSYCC detects this signal being asserted by the agent receiving the data. • MUSYCC asserts the PCI read cycle and generates the PERR Interrupt Descriptor toward the host under the following conditions: • MUSYCC masters a PCI read cycle. • After receiving the data during the data phase of the cycle, MUSYCC calculates that a parity error has occurred.
	41	INTA*	PCI MUSYCC Interrupt	o/d O	INTA* is driven by MUSYCC to indicate a MUSYCC Layer 2 interrupt condition to the host processor.
	40	INTB*	PCI Expansion Bus Interrupt	o/d O	INTB* is driven by MUSYCC to notify the host processor of an interrupt pending from the EBUS.
	47	REQ*	PCI Bus Request	t/s O	MUSYCC drives REQ* to notify the PCI arbiter that it desires to master the bus. Every master in the system has its own REQ*.
	46	GNT*	PCI Bus Grant	I	The PCI bus arbiter asserts GNT* when MUSYCC is free to take control of the bus, assert FRAME*, and execute a bus cycle. Every master in the system has its own GNT*.
	98	M66EN	66 MHz Enable	I	When asserted, M66EN indicates the system is operating at a 66 MHz PCI clock rate. Otherwise, it is operating at a 33 MHz or less clock rate. This pin is a no-connect on Revision A and B devices.

Table 1-4. CN8478 Hardware Signal Definitions *(6 of 6)*

	MQFP Pin No.	Pin Label	Signal Name	I/O	Definition
Boundary Scan and Test Access	35	TCK	JTAG Clock	I	Clock in the TDI and TMS signals and clock out TDO signal.
	36	TRST*	JTAG Reset	I	An active-low input that resets the JTAG state machine. This pin should be pulled low in normal operation.
	37	TMS	JTAG Mode Select	I	The test signal input decoded by the TAP controller to control test operations.
	38	TDO	JTAG Data Output	t/s O	The test signal that transmits serial test instructions and tests data.
	39	TDI	JTAG Data Input	I	The test signal that receives serial test instructions and tests data.
	112-114	TM[0] TM[1] TM[2]	Test Mode	I	Encodes test modes. These pins have internal pull-downs and may be left open by the system designer. TM[0] TM[1] TM[2] 0 0 0 Normal Operation. Tie to ground. 1 1 1 All outputs three-stated.
Power and Ground	*(2)*	VDDc VDDi*(3)* VDDo VGG	Power	–	19 pins are provided for power. Four VDDc (core), four VDDi (input), nine VDDo (output), and two VGG (5 V-tolerant supply). The VDDc require 2.5 V +/- 5%, the VDDi and VDDo require 3.3 V +/- 5%, and the VGG require 5 V +/- 5%. The recommended power ramp sequence is VDDi and VDDo together, then VDDc at $t = 0^+$. VGG can be powered at any time.
	(3)	VSS*(3)* VSSo	Ground	–	27 pins are provided for ground, 0 V DC. 10 VSS (core and input) and 17 VSSo (output).

NOTE(S):

(1) These pins have internal pullups and may be left open by the system designer.

(2) VDDc Pin Numbers: 27, 77, 132, 185
VDDi Pin Numbers: 13, 67, 118, 171
VDDo Pin Numbers: 42, 63, 81, 95, 110, 147, 164, 181, 201
VGG Pin Numbers: 52, 156

(3) VSS Pin Numbers: 14, 28, 53, 68, 78, 119, 133, 157, 172, 186
VSSo Pin Numbers: 44, 55, 64, 73, 82, 89, 96, 104, 111, 137, 148, 153, 165, 175, 182, 191, 202

(4) An active low signal is denoted by a trailing asterisk (*).

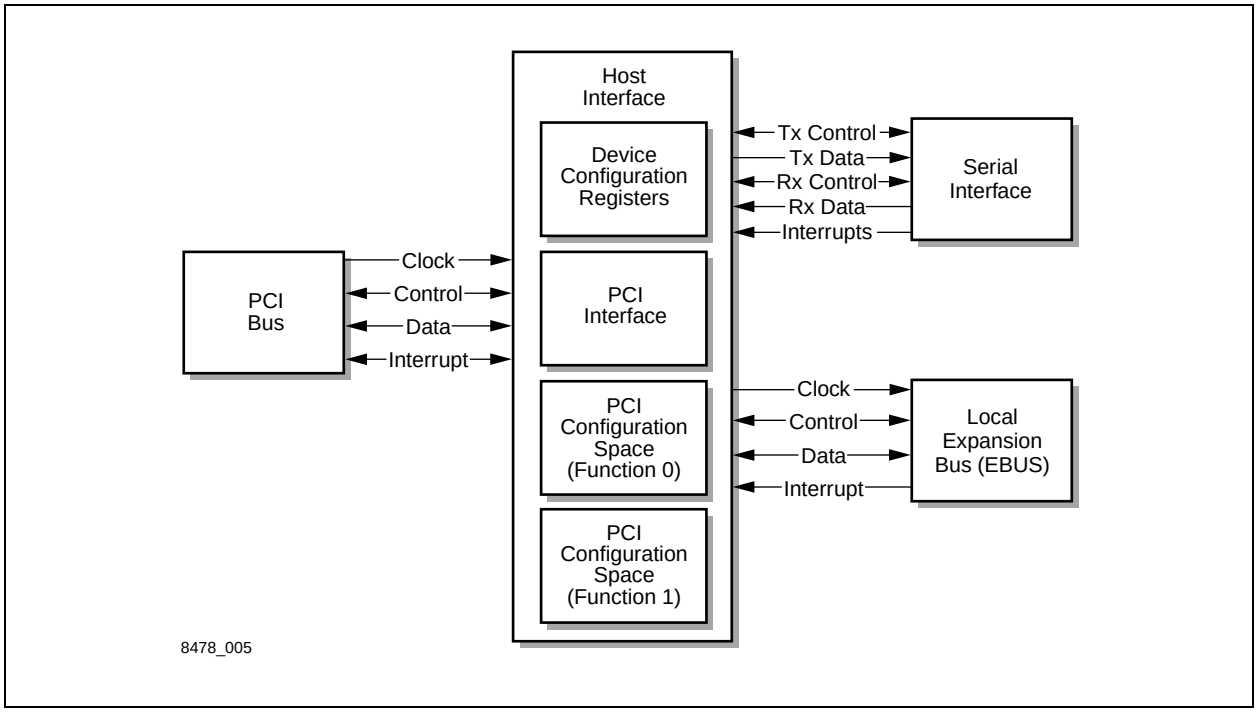
2.0 Host Interface

MUSYCC's host interface performs the following major functions:

- Transfers data between the serial interface and shared memory over the PCI bus
- Bridges system host processors to the devices connected to the EBUS
- Stores configuration state information

Figure 2-1 illustrates the host interface block diagram.

Figure 2-1. Host Interface Functional Block Diagram



2.1 PCI Interface

The host interface in MUSYCC is compliant with the *PCI Local Bus Specification* (Revision 2.1, June 1, 1995). MUSYCC provides a PCI interface specific to 3.3 V and 33 or 66 MHz operation.

NOTE: The *PCI Local Bus Specification* (Revision 2.1, June 1, 1995) is an architectural, timing, electrical, and physical interface standard providing the parameters for a device to connect with processor and memory systems.

The host interface can act as both a PCI master and PCI slave, and contains MUSYCC's PCI configuration space and internal registers. When MUSYCC must access shared memory, it masters the PCI bus and completes the memory cycles without external intervention.

MUSYCC provides the host with a PCI bridge to an on-device EBUS, and behaves as a PCI slave when providing this access.

MUSYCC is a multifunction PCI agent. One function is mapped to the layer 2 HDLC control logic; a second function is mapped to the layer 1 physical interface for the expansion bus pins.

2.1.1 PCI Initialization

Generally, when a system initializes a module containing a PCI device, the configuration manager reads the configuration space of each PCI device on a PCI bus. Hardware signals select a specific PCI device based on a bus number, a slot number, and a function number. If the addressed device (via signal lines) responds to the configuration cycle by claiming the bus, that function's configuration space is read out from the device during the cycle. Because any PCI device can be a multifunction device, every supported function's configuration space must be read from the device. Based on the information read, the configuration manager assigns system resources to each supported function within the device. Sometimes new information must be written to the function's configuration space; this is accomplished with a configuration write cycle.

MUSYCC is a multifunction device with device-resident memory to store the required configuration information. MUSYCC supports Function 0 and Function 1 and, as such, only responds to Function 0 and Function 1 configuration cycles, defined as listed below:

- Function 0: All HDLC processing as an HDLC network controller. Can master the PCI bus or respond to slave accesses from another bus master.
- Function 1: EBUS bridge to local devices. Responds only when another bus master performs a memory access into the Function 1 address range.

2.1.2 PCI Bus Operations

MUSYCC behaves either as a PCI master or a PCI slave at any time and switches between these modes as required during device operation.

As a PCI slave, MUSYCC responds to the following PCI bus operations:

- Memory Read
- Memory Write
- Configuration Read
- Configuration Write
- Memory Read Multiple (treated like Memory Read in slave mode)
- Memory Read Line (treated like Memory Read in slave mode)
- Memory Write and Invalidate (treated like Memory Write)

All other PCI cycles are ignored by MUSYCC. Only memory cycles are mapped to operations on the EBUS.

As a PCI-master, MUSYCC generates the following PCI bus operations:

- Memory Read Multiple (generated only in master mode)
- Memory Write
- Dual Address Cycle

2.1.3 PCI Configuration Space

This section describes how MUSYCC implements the required PCI configuration register space to provide configuration registers. These registers satisfy the needs of current and anticipated system configuration mechanisms, without specifying those mechanisms or otherwise placing constraints on their use. The configuration registers provide the following functions:

- Full device relocation, including interrupt binding
- Installation, configurations, and booting without user intervention
- System address map construction by device-independent software

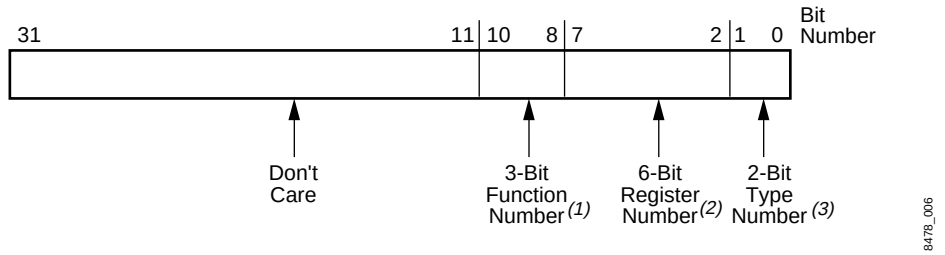
MUSYCC responds only to Type 0 configuration cycles. Type 1 cycles, which pass a configuration request on to another PCI bus, are ignored.

MUSYCC is a two-function PCI agent; therefore, it must implement configuration space for both functions.

The PCI controller in MUSYCC responds to configuration and memory cycles, but only memory cycles cause bus activity on the EBUS.

The address phase during a MUSYCC configuration cycle indicates the function number and register number being addressed which can be decoded by observing the status of the address lines AD[31:0]. Figure 2-2 shows the address lines during the configuration cycle.

Figure 2-2. Address Lines During Configuration Cycle



NOTE(S):
(1) MUSYCC supports Functions 0 and 1.
(2) MUSYCC supports Registers 0 through 15, inclusive.
(3) MUSYCC supports Type 0 configuration cycles.

The value of the signal lines AD[10:8] selects the function being addressed. MUSYCC supports Functions 0 and 1 and will not respond if another function is selected.

The value of the signal lines AD[7:2] during the address phase of configuration cycles selects the register of the configuration space to access. Valid values are 0–15. Accessing registers outside this range results in an all 0s' value being returned on reads, and no action being taken on writes.

The value of the signal lines AD[1:0] must be 00b for MUSYCC to respond. If these bits are 0 and the IDSEL signal line is asserted, then MUSYCC will respond to the configuration cycle.

Although there are two separate configuration spaces, one for Function 0 and one for Function 1, some internal registers are shared between the two spaces.

The Base Code register contains the Class Code, Sub Class Code, and Register Level Programming Interface registers. Tables 2-1 and 2-2 list Function 0 and Function 1 configuration spaces.

Table 2-1. Function 0 Configuration Space

	Register Number	Byte Offset (Hex)	31 24	16	8	0
Function Number 0	0	00h	Device ID(1)		Vendor ID(1)	
	1	04h	Status		Command	
	2	08h	Base Code			Revision ID(1)
	3	0Ch	Reserved	Header Type	LatencyTimer	Reserved
	4	10h	MUSYCC Base Address Register (BAR)			
	5 — 14	14h — 38h	— Reserved —			
	15	3Ch	Max Latency	Min Grant	Interrupt Pin	Interrupt Line

NOTE(S):
(1) Registers shared between Function 0 and 1.

Table 2-2. Function 1 Configuration Space

	Register Number	Byte Offset (Hex)	31 24	16	8	0
Function Number 1	0	00h	Device ID(1)		Vendor ID(1)	
	1	04h	Status		Command	
	2	08h	Base Code			Revision ID(1)
	3	0Ch	Reserved	Header Type	Reserved	Reserved
	4	10h	EBUS Base Address Register (BAR)			
	5 — 14	14h — 38h	— Reserved —			
	15	3Ch	Reserved		Interrupt Pin	Interrupt Line

NOTE(S):
(1) Registers shared between Function 0 and 1.

In summary, both configuration spaces have unique registers except for the Device ID, Vendor ID, and Revision ID, which are shared between the configuration spaces for Functions 0 and 1.

MUSYCC is a multifunction device with two sources of interrupts: the HDLC controller interrupts and the expansion bus physical layer interrupts. MUSYCC uses the INTA* pin for HDLC controller interrupts and the INTB* pin for interrupts generated by devices on the expansion bus connected to the EINT* pin.

All writable bits in the configuration space are reset to 0 by the hardware reset, PRST* asserted. After reset, MUSYCC is disabled and responds only to PCI configuration write and PCI configuration read cycles. Write cycles to reserved bits and registers have no effect. Read cycles to reserved bits always result in 0 being read.

2.2 PCI Configuration Registers

2.2.1 Function 0 Network Controller—PCI Master and Slave

MUSYCC provides the necessary configuration space for a PCI bus controller to query and configure MUSYCC's PCI interface. PCI configuration space consists of a device-independent header region (64 bytes) and a device-dependent header region (192 bytes). MUSYCC provides the device-independent header section only. Access to the device-dependent header region results in 0s being read, with no effect on writes.

There are three types of registers available in MUSYCC:

1. Read-Only (RO): Returns a fixed bit pattern if the register is used, or a 0 if the register is unused or reserved.
2. Read-Resettable (RR): Can be reset to 0 by writing a 1 to the register.
3. Read/Write (RW): Retains the value last written to it.

MUSYCC's Function 0 PCI Configuration Space has 16 dword registers. Tables 2-3 through 2-9 define these registers.

Register 0, Address 00h

Table 2-3. Register 0, Address 00h

Bit Field	Name	Reset Value	Type	Description
31:16	Device ID(1)	847xh	RO	This unique device identification is assigned by the manufacturer. This field always returns the value 847xh where x can be 1, 2, 4, or 8 depending on the 32, 64, 128, or 256 channel version of the device, respectively.
15:0	Vendor ID(1)	14F1h	RO	The unique vendor identification assigned to the manufacturer. This field always returns the value 14F1h.

NOTE(S):
(1) Registers shared between Function 0 and 1.

Register 1, Address 04h

The Status register records status information for PCI bus related events. The Command register provides coarse control to generate and respond to PCI commands.

At reset, MUSYCC sets the bits in this register to 0, meaning MUSYCC is logically disconnected from the PCI bus for all cycle types except configuration read and configuration write cycles.

Table 2-4. Register 1, Address 04h *(1 of 2)*

Bit Field	Name	Reset Value	Type	Description
31	Status	0	RR	Detected Parity Error. This bit is set by MUSYCC whenever it detects a parity error on a data phase when MUSYCC is a target, even if parity error response is disabled.
30		0	RR	Detected System Error. This bit is set by MUSYCC whenever it asserts SERR*.
29		0	RR	Received Master Abort. This bit is set by MUSYCC whenever a MUSYCC-initiated cycle is terminated with master-abort.
28		0	RR	Received Target Abort. MUSYCC sets this bit when a MUSYCC-initiated cycle is terminated by a target-abort.
27		0	RO	Unused.
26:25		01b	RO	DEVSEL* Timing. Indicates MUSYCC is a medium-speed PCI device. This means the longest time it will take MUSYCC to return DEVSEL* when it is a target of 3 clock cycles.
24		0	RR	Data Parity Detected. MUSYCC sets this bit when three conditions are met: 1. MUSYCC asserts PERR* or observes PERR*. 2. MUSYCC is the master for that transaction. 3. The Parity Error Response bit in this register is set.
23		1b	RO	Fast Back-to-Back Capable. Read Only. Indicates that when MUSYCC is a target, it is capable of accepting fast back-to-back transactions when the transactions are not to the same agent.
22		0	RO	Unused.
21		I	RO	Indicates the device is 66 MHz capable. This bit is set by Revision C and later devices.
20:16		0	RO	Unused.

Table 2-4. Register 1, Address 04h *(2 of 2)*

Bit Field	Name	Reset Value	Type	Description
15:10	Command	0	RO	Unused.
9		0	RW	Fast back-to-back mode is not supported.
8		0	RW	SERR* enable. If 1, disables MUSYCC's SERR* driver. If 0, enables MUSYCC's SERR* driver and allows reporting of address parity errors.
7		0	RO	Wait cycle control. MUSYCC does not support address stepping.
6		0	RW	Parity error response. This bit controls MUSYCC's Function 0 response to parity errors. If 1, MUSYCC takes normal action when a parity error is detected on a cycle with Function 0 as the target. If 0, MUSYCC ignores parity errors.
5		0	RO	VGA palette snoop. Unused.
4		0	RO	Memory write and invalidate. The only write cycle type MUSYCC generates is memory write.
3		0	RO	Special cycles. Unused. MUSYCC ignores all special cycles.
2		0	RW	Bus master. If 1, MUSYCC is permitted to act as bus master. If 0, MUSYCC is disabled from generating PCI accesses.
1		0	RW	Memory space. Access control. If 1, enables MUSYCC to respond to Function 0 memory space access cycles. If 0, disables MUSYCC's response.
0		0	RO	I/O space accesses. MUSYCC does not contain any I/O space registers.

NOTE(S): An active-low signal is denoted by a trailing asterisk (*).

Register 2, Address 08h

This location contains the Class Code and Revision ID registers. The Class Code register contains the Base Class Code, Sub-Class Code, and Register Level Programming Interface fields, used to specify the generic function of MUSYCC. The Revision ID register denotes the version of the device.

Table 2-5. Register 2, Address 08h

Bit Field	Name	Reset Value	Type	Description
31:24	Class Code	02h	RO	Base Class Code: Network Controller.
23:16		80h	RO	Sub-Class Code: Other Network Controller.
15:8		0	RO	Register Level Programming Interface: Indicates there is nothing special about programming MUSYCC.
7:0	Revision ID(1)	01h	RO	Denotes the revision number of MUSYCC. Rev A = 0Ah, Rev B = 0Bh, Rev C = 0Ch, etc.

NOTE(S):
(1) Registers shared between Function 0 and 1.

Register 3, Address 0Ch

Table 2-6. Register 3, Address 0Ch

Bit Field	Name	Reset Value	Type	Description
31	Built-In Self Test (BIST) Capable	1	RO	Returns 1 if device supports BIST. Returns 0 if it does not support BIST.
30	Start BIST	0	RW	Writes 1 to invoke BIST. Device resets the bit when BIST is complete. Software should fail the device if BIST is not complete after two seconds.
29:27	Reserved	0	RO	Unused.
26	BIST Error in the Interrupt Queue	—	RO	After “Start BIST” bit gets reset, this bit indicates if there were any errors in the interrupt queue RAM areas.
25	BIST Error in the Transmitter	—	RO	After “Start BIST” bit gets reset, this bit indicates if there were any errors in the transmit queue RAM areas.
24	BIST Error in the Receiver	—	RO	After “Start BIST” bit gets reset, this bit indicates if there were any errors in the receive queue RAM areas.
23:16	Header Type	80h	RO	MUSYCC is a multifunction device with the standard layout of configuration register space.
15:11	Latency Timer	0	RW	The latency timer is an 8-bit value that specifies the maximum number of PCI clock cycles that MUSYCC can keep the bus after starting the access cycle by asserting its FRAME*. The latency timer ensures that MUSYCC has a minimum time slot for it to own the bus, but places an upper limit on how long it will own the bus.
10:8		0	RO	
7:0	Reserved	0	RO	Unused.
NOTE(S): An active-low signal is denoted by a trailing asterisk (*).				

Register 4, Address 10h

Table 2-7. Register 4, Address 10h

Bit Field	Name	Reset Value	Type	Description
31:20	MUSYCC - Function 0 Base Address Register	0	RW	Allows for 1 MB bounded PCI bus address space to be blocked off as MUSYCC space. MUSYCC responds as a PCI slave with DEVSEL* to all bus cycles whose address bits 31:20 match the value of bits 31:20 of this register, and whose upper address bits are non-zero, and memory space is enabled in the Function 0 Register 1, memory space bit field. Reads to addresses within this space that are not implemented will read back 0; writes have no effect.
19:4		0	RO	When appended to bits 31:20, these bits specify a 1 MB bound memory range. 1 MB is the only amount of address space that a MUSYCC function can be assigned.
3		0	RO	MUSYCC memory space is not prefetchable.
2:1		0	RO	MUSYCC can be located anywhere in 32-bit address space.
0		0	RO	This base register is a memory space base register, as opposed to I/O mapped.
NOTE(S): An active-low signal is denoted by a trailing asterisk (*).				

Register 5–14, Address 14h–38h

Table 2-8. Registers 5–14, Addresses 14h–38h

Bit Field	Name	Reset Value	Type	Description
31:0	Reserved	0	RO	Unused.

Register 15, Address 3Ch

Table 2-9. Register 15, Address 3Ch

Bit Field	Name	Reset Value	Type	Description
31:24	Maximum Latency	0Fh	RO	Specifies how quickly MUSYCC needs to gain access to the PCI bus. The value is specified in 0.25 µs increments and assumes a 33 MHz clock. 0Fh means MUSYCC needs to gain access to the PCI bus every 130 PCI clock cycles, expressed as 3.75 µs in this register for 33 MHz PCI and 1.87 µs for 66 MHz PCI.
23:16	Minimum Grant	0	RO	This value specifies, in 0.25 µs increments, the minimum burst period MUSYCC needs. MUSYCC does not have any special MIN_GNT requirements. In general, the more channels MUSYCC has active, the worse the bus latency and the shorter the burst cycle.
15:8	Interrupt Pin	01b	RO	Defines which PCI interrupt pin Function 0 uses. 01h means MUSYCC uses pin INTA* for HDLC controller interrupts.
7:0	Interrupt Line	0	RW	Communicates interrupt line routing. System initialization software will write a value to this register indicating which host interrupt controller input is connected to MUSYCC's INTA* pin.
NOTE(S): An active-low signal is denoted by a trailing asterisk (*).				

2.2.2 Function 1 Expansion Bus Bridge, PCI Slave

MUSYCC, a multifunction PCI device, provides the necessary configuration space allowing a PCI bus or system controller to query and configure the host interface of MUSYCC as a PCI device. PCI configuration space consists of a device-independent header region (64 bytes) and a device-dependent header region (192 bytes). MUSYCC provides the 64-byte device-independent header section only. Access to the device-dependent header region results in 0s being read, and no effect on writes.

There are three types of registers available in MUSYCC:

1. Read-Only (RO)—Returns a fixed bit pattern if the register is used, or a 0 if the register is unused or reserved.
2. Read-Resettable (RR)—Can be reset to 0 by writing a 1 to the register.
3. Read/Write (RW)—Retains the value last written to it.

MUSYCC's Function 1 Configuration Space has 16 dword registers. Tables 2-10 through 2-16 describe these registers.

Register 0, Address 00h

Table 2-10. Register 0, Address 00h

Bit Field	Name	Reset Value	Type	Description
31:16	Device ID(1)	847xh	RO	This unique device identification is assigned by the manufacturer. This field always returns the value 847xh where x can be 1, 2, 4, or 8 depending on the 32, 64, 128, or 256 channel version of the device, respectively.
15:0	Vendor ID(1)	14F1h	RO	The unique vendor identification assigned to the manufacturer. This field always returns the value 14F1h.

NOTE(S):
(1) Registers shared between Function 0 and 1.

Register 1, Address 04h

The Status register records status information for PCI bus-related events. The Command register provides coarse control to generate and respond to PCI commands.

At reset, MUSYCC sets the bits in this register to 0. This means MUSYCC is logically disconnected from the PCI bus for all cycle types except configuration read and configuration write cycles.

Table 2-11. Register 1, Address 04h (1 of 2)

Bit Field	Name	Reset Value	Type	Description
31	Status	0	RR	Detected parity error. This bit is set by MUSYCC whenever it detects a parity error on a data phase.
30		0	RO	Unused.
29		0	RO	Unused.
28		0	RO	Unused.
27		0	RO	Unused.
26:25		01b	RO	DEVSEL* timing. Indicates MUSYCC is a medium-speed device. This means the longest time it will take MUSYCC to return DEVSEL* when the EBUS is the target is 3 clock cycles.
24		0	RO	Unused.
23		01b	RO	Fast back-to-back capable. Indicates that when the EBUS is a target, it is capable of accepting fast back-to-back transactions when the transactions are not to the same agent.
22		0	RO	Unused.
21		01b	RO	Indicates the device is 66 MHz capable. This bit is set by Revision C and later devices.
20:16		0	RO	Unused.

Table 2-11. Register 1, Address 04h *(2 of 2)*

Bit Field	Name	Reset Value	Type	Description
15:7	Command	0	RO	Unused.
6		0	RW	Parity error response. This bit controls MUSYCC's Function 1 response to parity errors. If 1, MUSYCC will take normal action when a parity error is detected on a cycle with Function 1 as the target. If 0, MUSYCC will ignore parity errors.
5:2		0	RO	Unused.
1		0	RW	Memory Space access control. If 1, enables MUSYCC to respond to Function 1 memory space access cycles. If 0, disables MUSYCC's response.
0		0	RO	I/O space accesses. MUSYCC does not contain any I/O space registers.
NOTE(S): An active-low signal is denoted by a trailing asterisk (*).				

Register 2, Address 08h

This location contains the Class Code and Revision ID registers. The Class Code register contains the Base Class Code, Sub-Class Code, and Register Level Programming Interface fields, used to specify the generic functions of MUSYCC. The Revision ID register denotes the version of silicon.

Table 2-12. Register 2, Address 08h

Bit Field	Name	Reset Value	Type	Description
31:24	Class Code	06h	RO	Base Class Code: Bridge Device.
23:16		80h	RO	Sub-Class Code Type: Other Bridge Device.
15:8		0	RO	Register Level Programming Interface: Indicates there is nothing special about programming MUSYCC.
7:0	Revision ID(1)	01h	RO	Denotes the revision number of MUSYCC. Rev A = 0Ah, Rev B = 0Bh, Rev C = 0Ch, etc.
NOTE(S): (1) Registers shared between Function 0 and 1.				

Register 3, Address 0Ch

Table 2-13. Register 3, Address 0Ch

Bit Field	Name	Reset Value	Type	Description
31:24	Reserved	0	RO	Unused.
23:16	Header Type	80h	RO	MUSYCC is a multifunction device with the standard layout of configuration register space.
15:0	Reserved	0	RO	Unused.

Register 4, Address 10h

Table 2-14. Register 4, Address 10h

Bit Field	Name	Reset Value	Type	Description
31:20	EBUS—Function 1 Base Address Register	0	RW	Allows for 1 MB bounded PCI bus address space to be blocked off as MUSYCC expansion bus space. MUSYCC responds as a PCI slave with DEVSEL* to all memory cycles whose non-zero address bits 31:20 match the value of bits 31:20 of this register, with memory space enabled in Function 1 Register 1, memory space bit field. Reads to addresses within this space that are not implemented. Reads back 0; writes have no effect. PCI cycles to this space will be mapped to read or write cycles on the expansion bus.
19:4		0	RO	When appended to bits 31:20, specifies a 1 MB bound memory space. 1 MB is the only size of address space that a MUSYCC function can be assigned.
3		0	RO	Expansion bus memory space is not prefetchable.
2:1		0	RO	Means MUSYCC expansion bus space can be located anywhere in 32-bit address space.
0		0	RO	Means this base register is a memory space base register, as opposed to I/O mapped.
NOTE(S): An active-low signal is denoted by a trailing asterisk (*).				

Register 5–14, Addresses 14h–38h

Table 2-15. Registers 5 through 14–Addresses 14h through 38h

Bit Field	Name	Reset Value	Type	Description
31:0	Reserved	0	RO	Unused.

Register 15, Address 3Ch

Table 2-16. Register 15, Address 3Ch

Bit Field	Name	Reset Value	Type	Description
31:16	Reserved	0	RO	Unused.
15:8	Interrupt Pin	02h	RO	Defines which PCI interrupt pin Function 1 uses. 02h means MUSYCC uses pin INTB* for interrupts sourced by devices connected to EBUS.
7:0	Interrupt Line	0	RW	Communicates interrupt line routing. System initialization software writes a value to this register indicating which host interrupt controller input is connected to MUSYCC's INTB* pin.
NOTE(S): An active-low signal is denoted by a trailing asterisk (*).				

2.2.3 PCI Reset

MUSYCC resets all internal functions when it detects the assertion of the PRST* signal line. Upon reset, the following occurs:

- All PCI output signals go to three-state immediately and asynchronously with respect to the PCI clock input, PCLK.
- All EBUS output signals go to three-state immediately and asynchronously with respect to the EBUS clock output, ECLK.
- All writable register bits are set to 0.
- All PCI data transfers terminate immediately.
- All serial data transfers terminate immediately.
- MUSYCC disables and responds only to PCI configuration cycles.

2.2.4 Host Interface

After a hardware reset, the PCI configuration space within MUSYCC needs to be configured by the host with the following information:

For Function 0:

- Base address register
- Fast back-to-back enable/disable
- SERR* signal driver enable/disable
- Parity error response enable/disable
- Bus mastering enable/disable
- Memory space access enable/disable

For Function 1:

- Base address register
- Parity error response enable/disable
- Memory space access enable/disable

Function 0 provides services to the serial interfaces in MUSYCC; Function 1 provides services to the EBUS interface in MUSYCC.

After the configuration spaces are configured, MUSYCC can master the PCI bus or provide slave-mode access to the host.

2.2.5 PCI Bus Parity

The agent driving the AD[31:0] signals during any bus phase must also drive the even parity signal (PAR). PAR is driven one clock after AD[31:0] has been driven as follows:

- Address phase: master always drives PAR one clock after address phase.
- Read data phase: target always drives PAR one clock after read data phase.
- Write data phase: master always drives PAR one clock after write data phase.

PAR provides even parity across the AD[31:0] and CBE[3:0]* signal lines. The agent receiving the data must assert PERR* if it detects a parity error, provided its Parity Error Response enable bit is set.

If a parity error occurs, the master that generated the cycle (whether it asserted PERR* or detected it) reports parity errors to the host. MUSYCC does this by generating an Interrupt Descriptor. It also sets the Data Parity Detected bit (for masters only) in the Status register in the appropriate function's PCI configuration space and sets the Detected Parity Error (for masters or targets) in the same register if MUSYCC is the agent that detected the error.

PERR* reports errors on the data phases. MUSYCC not only asserts PERR* when appropriate, but monitors PERR* for its own memory transactions and notifies the host of the parity error.

SERR* reports parity errors on the address phases. It is assumed that this open drain PCI signal is tied directly to the host's system error pin. MUSYCC does not generate an Interrupt Descriptor if it detects a parity error on an address phase, nor does it respond to SERR* assertion.

2.2.6 PCI Throughput and Latency Considerations

In PCI systems, achieving high bus throughput works against achieving low bus latency. As devices burst more data, they keep the bus longer, causing other devices waiting for the bus to experience a longer acquisition latency as a result.

A PCI bus master introduces latency each time it uses the PCI bus to perform a transaction. The bus master latency is a function of the following:

- Behavior of the master
  - State of the GNT* signal
  - Bus command used (read, write,...)
  - Burst length
  - Master data latency for each data phase
  - Value of Latency Timer
- Behavior of the target
  - Bus command used (read, write,...)
  - Target latency

When MUSYCC requests the PCI bus, it needs the bus to transfer data between an internal FIFO buffer and shared memory across the PCI bus with either a read or a write access. While MUSYCC waits for the bus to be granted, and then while MUSYCC transfers the data, another equal-sized internal FIFO buffer is simultaneously being filled or emptied at the serial interface. When MUSYCC requests the bus, it has data to transfer, and also has a finite amount of time (which is directly related to the speed of the serial line clock) before a separate FIFO buffer at the serial interface overflows or underflows.

For an application with many logical channels, MUSYCC requires a new access cycle on the PCI bus more frequently than an application with fewer logical channels. If FIFO buffer space is evenly distributed across all channels, more channels result in less FIFO buffer space per channel, and FIFO buffer space must be cleared more frequently.

Conversely, an application with high data rate serial interfaces requires a new access cycle on the PCI bus more frequently than an application with a low data rate serial interface, because the FIFO buffer fills faster in the former.

Acquiring the PCI bus requires having to deal with arbitration latency, which is defined as the number of PCI clock cycles a master must wait after asserting its REQ* and before asserting the GNT* signal. This number is a function of the system's arbitration algorithm and takes into account the sequence in which masters are given access to the bus and the latency timer of each master. Arbitration latency is also affected by the loading of the system and how efficiently the bus is being utilized.

The master's latency timer specifies the maximum number of PCI clock cycles that the master can (and in the case of MUSYCC, will) keep the bus after starting the access cycle by asserting its FRAME*. The latency timer also ensures that the master has a minimum time slot for it to own the bus, but places an upper limit on how long it will own the bus. In MUSYCC, the Latency Timer is reset to 0 on PRST* (PCI reset).

Once the bus is acquired and bursting begins, PCI throughput becomes the point of focus. MUSYCC is capable of multi-dword bursts (read or write). As each FIFO buffer for a logical channel and direction is serviced on the PCI, MUSYCC relinquishes and then reacquires the bus to service the FIFO buffer of the next logical channel. If more logical channels are serviced, bus turnover is increased, which decreases throughput (but does not necessarily affect service). If fewer logical channels are serviced, bus turnover decreases, and that increases throughput (but not necessarily to the benefit of channel processing).

Refer to Chapter 3 of the *PCI Local Bus Specification*, Revision 2.1, for a description of bandwidth and latency considerations.

2.2.6.1 PCI Bus Latency

The latency that a PCI master encounters as it tries to gain access to the PCI bus has three components:

1. Arbitration latency: usually 2 clock cycles for a high priority device, but is added into the total latency time only if the bus is idle when a device requests it, otherwise, it overlaps with the bus acquisition latency.
2. Bus acquisition latency: length of time a device must wait for the bus to become free.
3. Target latency: length of time the selected target takes to assert TRDY* for the first data transfer.

The longest latency MUSYCC experiences in gaining access to the PCI bus is

$$Latency_{Total} = \sum_{i=0}^{k-1} (T_i + 8)$$

or *[k* x *(T + 8)]* when all T_is are equal, where:

k = the number of PCI masters in the system

T = the value of the latency timers in those masters

8 = the longest target latency allowed, in clock cycles (exception: the first data phase is allowed 16 clock cycles)

Once a master gets the bus, it starts a count-down timer loaded with the value T, from the latency timer register. When the count reaches 0, the master relinquishes the bus when its GNT* is removed and it sees TRDY* on the final data phase. As long as its GNT* is still asserted, the master is free to burst indefinitely. Table 2-17 provides an example of PCI latency.

Table 2-17. PCI Latency Example

PCI Clock Increment	Bus Activity	
0	Bus is idle. Host asserts REQ*. MUSYCC asserts REQ*.	
+1	Host gets GNT*.	These 2 clock cycles are the arbitration latency that becomes 0 if the bus was not idle.
+1	Host asserts FRAME* to start access cycle.	
+(T + 8) or [16 + (n - 1) x 8] whichever is smaller — Host has bus —	This is the bus acquisition latency time—the amount of time the next requestor must wait for the bus because of current master, the host. During this time, assume the host loses its GNT* just +1 clock cycle into its acquisition and MUSYCC0 receives the GNT* +1 into this time. The host's first data phase must finish within 16 PCI clock cycles, and subsequent data phases must finish within 8 cycles each. Therefore, 16 + (n - 1) x 8 clock cycles is how long the host will need the bus to execute n data phases (n dword burst), assuming the host's access finishes before its latency timer expires. As the cycle finishes, the host relinquishes the bus, and one clock cycle later, MUSYCC0 gets the GNT* and subsequently asserts its FRAME* to start the access cycle.	
+(T + 8) or [16 + (n - 1) x 8] whichever is smaller — MUSYCC0 has bus —	MUSYCC0 finishes with the bus, and MUSYCC1 has it on the next clock cycle. During this time, MUSYCC0 loses its GNT*, and MUSYCC1 receives its GNT*. MUSYCC0 behaves similarly to the host above.	
+(T + 8) or [16 + (n - 1) x 8] whichever is smaller — MUSYCC1 has bus —	MUSYCC1 finishes with the bus, and MUSYCC2 has it on the next clock cycle. During this time, MUSYCC1 loses its GNT*, and MUSYCC2 receives its GNT*. MUSYCC1 behaves similarly to the host above.	
NOTE(S): An active low signal is denoted by a trailing asterisk (*).		

The predictable worst case time MUSYCC must wait for the bus in a system with *k* masters with equal latency timers is [*k* x (*T* + 8)].

If one MUSYCC is configured with all 256 channels active, and receiving and transmitting at 64 kbps, it must maintain a data rate of 16 Mbps across the PCI bus. Therefore:

- 256 channels x (64 kbps Rx + 64 kbps Tx) = 32,768 kbps

With 32 bits in each dword, the data rate in kilo dwords per second (kdwps) is:

- 32,768 kbps / (32 bits/dword) = 1,024 kdwps

The 16-clock rule (*PCI Local Bus Specification*, Revision 2.1) requires that a single access device must complete the access cycle within 16 clock cycles of the FRAME* signal being asserted. For devices capable of burst-mode, the 16-clock rule applies to the completion of the first data cycle.

2.2.6.2 Latency Computation—Single Dword Access

Assuming the worst case scenario where the system allows only single dword access, even a burst-mode device such as MUSYCC must relinquish the PCI bus within 16 clock cycles from receiving the bus. Using this scenario, the calculations continue as follows:

- The time per dword would be:
  1 dword / 1,024 kdwps = 0.98 µs per dword
- Assuming a PCI bus rate of 33 MHz, the time per clock cycle would be:
  1 cycle / 33 MHz = 30.303 ns per clock cycle
- Assuming a PCI bus rate of 66 MHz, the time per clock cycle would be:
  1 cycle / 66 MHz = 15.152 ns per clock cycle
- To get the number of clock cycles per dword:
  0.98 µs per dword / 0.0303 µs per clock cycle = 33 PCI clock cycles per dword
- To get the number of clock cycles per dword:
  0.98 µs per dword / 0.0152 µs per clock cycle = 66 PCI clock cycles per dword

With one MUSYCC and one host, the host can use the following:

- Assuming a PCI bus rate of 33 MHz:
  33 cycles per dword – 16 cycles (16 clock rule) = 17 clock cycles between dword transfers
- Assuming a PCI bus rate of 66 MHz:
  66 cycles per dword – 16 cycles (16 clock rule) = 50 clock cycles between dword transfers

Accordingly, MUSYCC's T must be:

- Assuming a PCI bus rate of 33 MHz:
  17 cycles – 8 cycle (target latency) = 9 clock cycles. As *T* has a granularity of 8 units, *T* must be programmed to 8 PCI clock cycles.
- Assuming a PCI bus rate of 66 MHz:
  50 cycles – 8 cycle (target latency) = 42 clock cycles. As *T* has a granularity of 8 units, *T* must be programmed to 40 PCI clock cycles.

2.2.6.3 Latency Computation—Burst Access

When the following is assumed:

- MUSYCC has enough internal buffering to buffer up to 4-dwords worth of information per channel before performing a 2-dword burst cycle for every access.
- MUSYCC has a granularity of 8 for its latency timer (that is, MUSYCC is always configured to give up the bus in equal to or less than the desired time-out).
- The system will support MUSYCC burst writes and reads.
- MUSYCC, with all 256 receive and transmit channels active, needs to move 1,024 kdwords/s, or one dword every 0.98 µs, or 4-dword bursts every 3.92 µs. That is, 130 clock cycles between bursts for a 33-MHz PCI bus rate, and 260 clock cycles for a 66-MHz PCI bus rate.

The following can be seen:

- The worst case time it would take each burst cycle to finish is 16 cycles (16 clock rule) + 8 cycles, target latency = 24 clock cycles to finish, worst case.
- With one MUSYCC and one host operating at a PCI bus rate of 33 MHz: The host has 130 cycles between bursts – 24 cycles to finish, worst case = 106 clock cycles. The host's *T* must be programmed to 106 cycles – 8 cycles, target latency = 98 cycles. Rounding for granularity yields 96 cycles.
- With one MUSYCC and one host operating at a PCI bus rate of 66 MHz: The host has 260 cycles between bursts – 24 cycles to finish, worst case = 236 clock cycles. The host's *T* must be programmed to 236 cycles – 8 cycles, target latency = 228 cycles. Rounding for granularity yields 224 cycles.
- For n MUSYCC and one host operating at a PCI bus rate of 33 MHz: The host has 130 cycles between bursts – (n x 24 cycles, worst case) – 8 clock cycles, target latency = T cycles. Therefore, for two MUSYCC's and one host, a host's *T* of 24 would be sufficient; that is, 130 cycles – (2 x 24) – 8 cycles = 74 clock cycles. Rounding for granularity equals 72 cycles.
- For n MUSYCC and one host operating at a PCI bus rate of 66 MHz: The host has 260 cycles between bursts – (n x 24 cycles, worst case) – 8 clock cycles, target latency = T cycles. Therefore, for two MUSYCC's and one host, a host's *T* of 24 would be sufficient; that is, 260 cycles – (2 x 24) – 8 cycles = 204 clock cycles. Rounding for granularity equals 200 cycles.

On reset, the value of the latency timers are reset to 0.

3.0 Expansion Bus (EBUS)

MUSYCC provides a PCI bridge to a local bus interface on MUSYCC called the Expansion Bus (EBUS). The EBUS provides a host processor across the PCI bus to access up to 1 MB of peripheral memory space on the EBUS.

Although EBUS utilization is optional, the most notable application for the EBUS is the connection to peripheral devices (e.g., Bt8370 T1/E1 framers) local to MUSYCC's serial port. Figures 3-1 and 3-2 illustrate block diagrams of the EBUS interface with and without local Multiprocessor Unit (MPU).

Figure 3-1. EBUS Functional Block Diagram with Local MPU

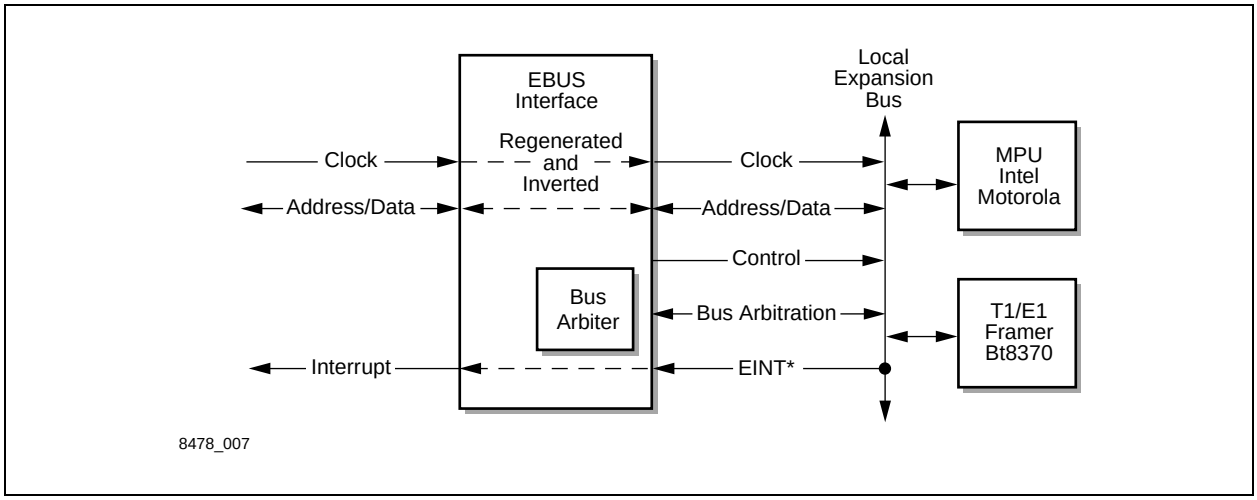
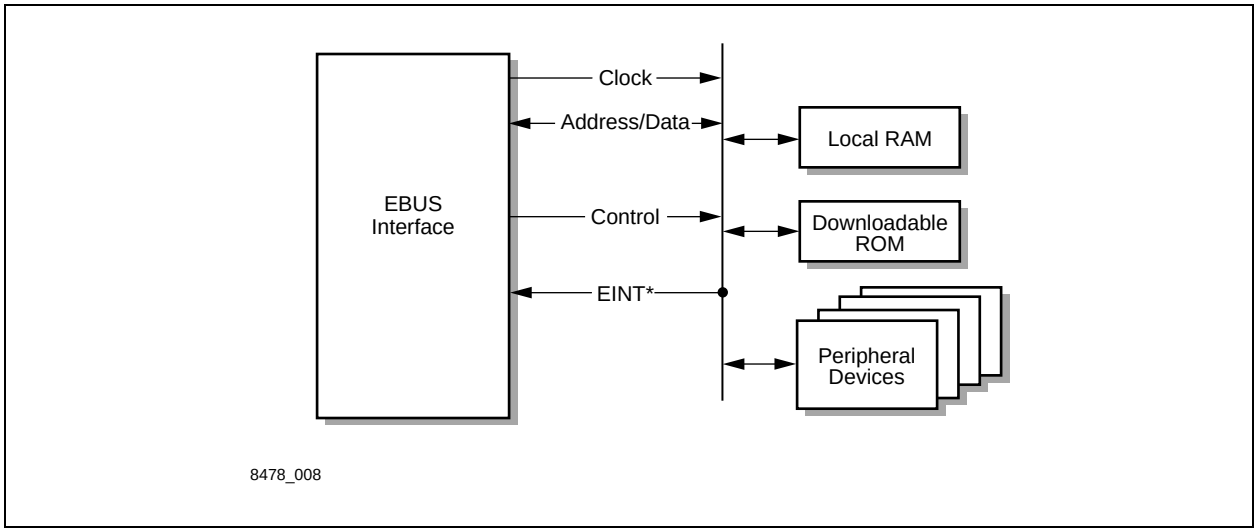


Figure 3-2. EBUS Functional Block Diagram without Local MPU



3.1 Operation

3.1.1 Initialization

At initialization, MUSYCC's PCI Function 1 Configuration Space is initialized with a value representing a 1 MB memory range assigned to MUSYCC's EBUS. This is detailed in Table 2-14, *Register 4, Address 10h*, and listed as EBUS—Function 1 Base Address Register. An unmapped 1 MB system memory range must be specified by assigning the upper 12 bits of the memory range to the upper 12 bits of this register.

Command bit field memory space access control and optional parity error response must be properly configured for MUSYCC to respond to EBUS memory space accesses (see Table 2-4, *Register 1, Address 04h*).

On reset, MUSYCC disables EBUS memory space access. If the PCI attempts to access EBUS memory space, there will be a PCI master-abort termination.

3.1.2 Address and Data

When MUSYCC's host interface claims the cycle during a PCI access cycle, the host interface compares the upper 12 bits of the PCI address lines to each of its function's base address registers. If signal lines AD[31:20] are identical to the upper 12 bits of the Expansion Bus Base Address register, MUSYCC forwards the access cycle to the EBUS interface within MUSYCC.

NOTE: Only single dword PCI operations can be performed when accessing the EBUS.

MUYSCC accepts PCI slave burst write cycles to either function 0 or function 1.

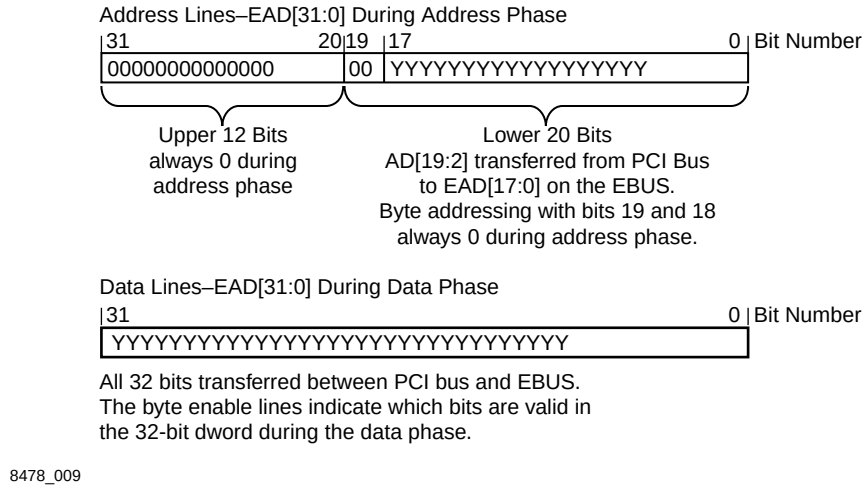
MUSYCC's host interface has an internal 4-dword write FIFO buffer shared by both functions; therefore a 1–4 dword burst write cycle can be performed to either function. When the burst write data phase exceeds the length, MUSYCC asserts a PCI target disconnect.

MUSYCC performs a PCI target disconnect after the first data phase of any burst read cycle to either Function 0 or Function 1. Therefore, the PCI bridge must be able to fragment a burst access into a single phase read or 1–4 phase burst writes as controlled by the target disconnect.

Assuming the EBUS is connected to byte-wide peripheral devices, the EBUS interface uses the lower 20 bits from PCI address lines AD[19:0] to construct a byte address for the EBUS. Specifically, PCI address lines AD[19:2] are converted to EBUS address lines EAD[17:0] by shifting out the two least significant bits, AD[1:0]. This allows for byte-level addressing for up to 4 byte-wide devices on the EBUS. Given the above, the EBUS provides an 18-bit addressing structure allowing byte addressing of up to four banks of 256 kB address space each.

The EBUS interface transfers 32 bits of the data lines between the EBUS and the PCI bus. The byte-enable signal lines EBE[3:0]* are transferred from the PCI byte-enable signal lines CBE[3:0]* to the EBUS, and indicate which byte(s) in the data dword are valid. Figure 3-3 illustrates both data and data configurations of the 32-bit word.

Figure 3-3. EBUS Address/Data Line Structure



NOTE(S):

1. Byte Enable 0–EBE[0]* signals if EAD[7:0] are valid data bits during data phase.
2. Byte Enable 1–EBE[1]* signals if EAD[15:8] are valid data bits during data phase.
3. Byte Enable 2–EBE[2]* signals if EAD[23:16] are valid data bits during data phase.
4. Byte Enable 3–EBE[3]* signals if EAD[31:24] are valid data bits during data phase.
5. An active low signal is denoted by a trailing asterisk (*).

3.1.3 Clock

The ECLK is derived from the PCI clock and runs at up to a 33 MHz clock rate. This operation is controlled by the M66EN input on Revision C and later devices. An asserted M66EN input implies that the overall system is operating at a 66 MHz PCI clock rate; the ECLK is running at half of the PCI clock rate. Otherwise, the ELCK is operating at the same rate as the PCI clock frequency. In order to ensure that the ELCK is properly operational, the M66EN input state shall not be changed during the whole operational period.

The EBUS clock output can be disabled by setting the ECKEN bit field (see Table 5-6). In the disabled state, the ECLK output is three-stated.

3.1.4 Interrupt

When a device connected to the EBUS drives the EINT* signal, MUSYCC carries this signal through to the PCI interrupt line, INTB*. Thus, peripheral devices can interrupt the host processor.

In MUSYCC's Function 1 PCI Configuration Space (the EBUS function), the Interrupt Pin bit field indicates that the INTB* PCI interrupt be asserted for interrupts sourced by devices connected to the EBUS (see Table 2-16, *Register 15, Address 3Ch*). Also, the Interrupt Line bit field in the same register is set up by the system initialization software to indicate which host interrupt controller input pin is to be connected to MUSYCC's INTB* pin.

3.1.5 Address Duration

MUSYCC can extend the duration the address bits are valid for any EBUS address phase by specifying a value from 0–3 in ALAPSE bit field (refer to Table 5-6, *Global Configuration Descriptor*). The value specifies the additional ECLK periods the address bits remain asserted. That is, a value of 0 specifies the address remains asserted for one ECLK period, and a value of 3 specifies the address remains asserted for four ECLK periods. Disabling the ECLK signal output does not affect the delay mechanism. Refer to the timing diagrams in Section 7.2.4 for more details.

Both pre- and post-address cycles are always present during the address phase of an EBUS cycle. The post-address cycle is one PCI period long and provides MUSYCC time to transition between the address phase and the following data phase. The pre- and post-address cycles are not included in the address duration.

3.1.6 Data Duration

MUSYCC can extend the duration that the data bits are valid for any EBUS data phase by specifying a value from 0–7 in ELAPSE bit field (refer to Table 5-6, *Global Configuration Descriptor*). The value specifies the additional ECLK periods the data bits remain asserted. That is, a value of 0 specifies the data that remains asserted for one ECLK period, and a value of 7 specifies the data that remains asserted for eight ECLK periods. Disabling the ECLK signal output does not affect the delay mechanism. Refer to the timing diagrams in Section 7.2.4 for more details.

A pre-data and post-data cycle are always present during the data phase of an EBUS cycle. The pre-data cycle is one PCI period long and provides MUSYCC setup and hold time for the data signals. The post-data cycle is one ECLK period long and provides MUSYCC time to transition between the data phase and the following bus cycle termination. The pre- and post-data cycles are not included in the data duration.

3.1.7 Bus Access Interval

MUSYCC can be configured to wait a specified amount of time after it releases the EBUS and before it requests the EBUS a subsequent time. This is accomplished by specifying a value 0–7 in BLAPSE bit field (refer to Table 5-6, *Global Configuration Descriptor*). The value specifies the additional ECLK periods MUSYCC waits immediately after releasing the bus; that is, a value of 0 specifies MUSYCC will wait for one ECLK period, and a value of 5 specifies six ECLK periods. Disabling the ECLK signal output does not affect this wait mechanism. Refer to the timing diagrams in Section 7.2.4 for more details.

The bus grant signal (HLDA/BG*) is deasserted by the bus arbiter only after the bus request signal (HOLD/BR*) is deasserted by MUSYCC. As the amount of time between bus request deassertion and bus grant deassertion can vary from system to system, it is possible for a misinterpretation of the "old" bus grant signal as an approval to access the EBUS. MUSYCC provides the flexibility through the bus access interval feature to wait a specific number of ECLK periods between subsequent bus requests. (Refer to EBUS arbitration timing diagrams, Figure 7-13, *EBUS Write/Read Transactions, Intel-Style* and Figure 7-14, *EBUS Write/Read Transactions, Motorola-Style*.)

3.1.8 PCI to EBUS Interaction

Using the EBUS to perform extensive polling of peripheral devices substantially increases PCI bus utilization. The EBUS interface within MUSYCC performs single dword access without burst cycles. Also, the access time for data on the EBUS is dependent on how fast the peripherals respond to an EBUS read or write cycle.

PCI write access cycles targeted at the EBUS are not at issue because they complete immediately. MUSYCC's host interface autonomously completes writing data to the EBUS after successfully terminating the host's PCI write access cycle.

PCI read access cycles targeted at the EBUS are at issue because they cause MUSYCC's host interface to first claim the access cycle, then immediately initiate a PCI Target Retry sequence. This causes the PCI bridge device to retry the same EBUS access at a later time. Concurrently, the EBUS interface is activated to access the requested data from the EBUS. Because this process may take many EBUS clock cycles to complete, the host interface is capable of holding off each retry request by initiating a subsequent Target Retry sequence until the EBUS interface delivers the required data to the host interface. Target Retry sequences may occur multiple times.

As EBUS data is made available to the host interface, and on the next retry from the bridge chip, the host interface checks whether or not the retry cycle address matches the address latched in during the initial EBUS access cycle and, if so, forwards the EBUS data to the requester. If the addresses do not match, MUSYCC starts a new EBUS access cycle.

The amount of time to complete a single EBUS cycle accessing a single dword at a time and the number of bus turnovers between successive retries affect PCI bus utilization. To avoid affecting the PCI bus adversely, systems must be designed to throttle EBUS access or use a local microprocessor on the EBUS to filter the information from peripheral devices.

3.1.9 Microprocessor Interface

The MPUSEL bit field specifies the type of microprocessor interface to use for the EBUS. (See Table 5-6, *Global Configuration Descriptor*.)

Table 3-1 describes the effective signals when Intel-style protocol is selected.

Table 3-1. Intel Protocol Signals

Signal	Description	Interpretation
ALE*	Address Latch Enable	Asserted low by MUSYCC to indicate that the address lines contain a valid address. This signal remains asserted for the duration of the access cycle.
RD*	Read	Strobed low by MUSYCC to enable data reads out of the device. Held high during writes.
WR*	Write	Strobed low by MUSYCC to enable data writes into the device. Held high during reads.
HOLD	Hold Request	Asserted high by MUSYCC when it requests the EBUS from a bus arbiter.
HLDA	Hold Acknowledge	Asserted high by bus arbiter in response to HOLD signal assertion. Remains asserted until after the HOLD signal is deasserted. If the EBUS is connected and there are no bus arbiters on the EBUS, this signal must be asserted high at all times.
NOTE(S): An active low signal is denoted by a trailing asterisk (*).		

Table 3-2 shows the effective signals when Motorola-style protocol is selected.

Table 3-2. Motorola Protocol Signal

Signal	Description	Interpretation
AS*	Address Strobe	Driven low by MUSYCC to indicate that the address lines contain a valid address. This signal remains asserted for the duration of the access cycle.
DS*	Data Strobe	Strobed low by MUSYCC to enable data reads or data writes for the addressed device.
R/WR*	Read/Write	Held high throughout read operation and held low throughout write operation by MUSYCC. This signal determines the meaning (read or write) of DS*.
BR*	Bus Request	Asserted low by MUSYCC when it requests the EBUS from a bus arbiter.
BG*	Bus Grant	Asserted low by bus arbiter in response to BR* signal assertion. Remains asserted until after the BR* signal is deasserted. If the EBUS is connected and there are no bus arbiters on the EBUS, this signal must be asserted low at all times.
BGACK*	Bus Grant Acknowledge	Asserted low by MUSYCC when it detects BGACK* currently deasserted. As this signal is asserted, MUSYCC begins the EBUS access cycle. After the cycle is finished, this signal is deasserted indicating to the bus arbiter that MUSYCC has released the EBUS.
NOTE(S): An active low signal is denoted by a trailing asterisk (*).		

3.1.10 Arbitration

The HOLD and HLDA (Intel style) or BR* and BG* (Motorola style) signal lines are used by MUSYCC to arbitrate for the EBUS.

For Intel-style interfaces, the arbitration protocol is as follows (refer to Figure 7-13, *EBUS Write/Read Transactions, Intel-Style*):

1. MUSYCC three-states EAD[31:0], EBE*[3:0]. WR*, RD*, and ALE*.
2. MUSYCC requires EBUS access and asserts HOLD.
3. MUSYCC checks for HLDA assertion by bus arbiter.
4. If HLDA is deasserted, MUSYCC waits for the HLDA signal to become asserted before continuing the EBUS operation.
5. If HLDA is asserted, MUSYCC continues with the EBUS access because it has control of the EBUS.
6. MUSYCC drives EAD[31:0], EBE*[3:0], WR*, RD*, and ALE*.
7. MUSYCC completes EBUS access and deasserts HOLD.
8. Bus arbiter deasserts HLDA shortly thereafter.
9. MUSYCC three-states EAD[31:0], EBE*[3:0]. WR*, RD*, and ALE*.

For Motorola-style interfaces, the arbitration protocol is as follows (refer to Figure 7-14, *EBUS Write/Read Transactions, Motorola-Style*):

1. MUSYCC three-states EAD[31:0], EBE*[3:0]. R/WR*, DS*, and AS*.
2. MUSYCC requires EBUS access and asserts BR*.
3. MUSYCC checks for BG* assertion by bus arbiter.
4. If BG* is deasserted, MUSYCC waits for the BG* signal to become asserted before continuing the EBUS operation.
5. If BG* is asserted, MUSYCC continues with the EBUS access as it has control of the EBUS.
6. If BGACK* is deasserted, MUSYCC assumes control of the EBUS by asserting BGACK*.
7. MUSYCC drives EAD[31:0], EBE*[3:0], R/WR*, DS*, AS*.
8. Shortly after the EBUS cycle is started, MUSYCC deasserts BR*.
9. Bus arbiter deasserts BG* shortly thereafter.
10. MUSYCC completes EBUS cycle.
11. MUSYCC deasserts BGACK*.
12. MUSYCC three-states EAD[31:0], EBE*[3:0]. R/WR*, DS*, and AS*.

3.1.11 Connection

Using the EBUS address lines, EAD[17:0], and the byte enable lines, EBE[3:0]*, the EBUS can be connected in either a multiplexed or non-multiplexed address and data mode.

Figures 3-4 and 3-5 illustrate two examples of non-multiplexed address and data modes. These figures illustrate four separate byte-wide framer devices connected to the EBUS with each byte enable line used as the chip select for separate devices. This allows a full dword data transfer over the EBUS.

Figure 3-4. EBUS Connection, Non-multiplexed Address/Data, 8 Framers, No MPU

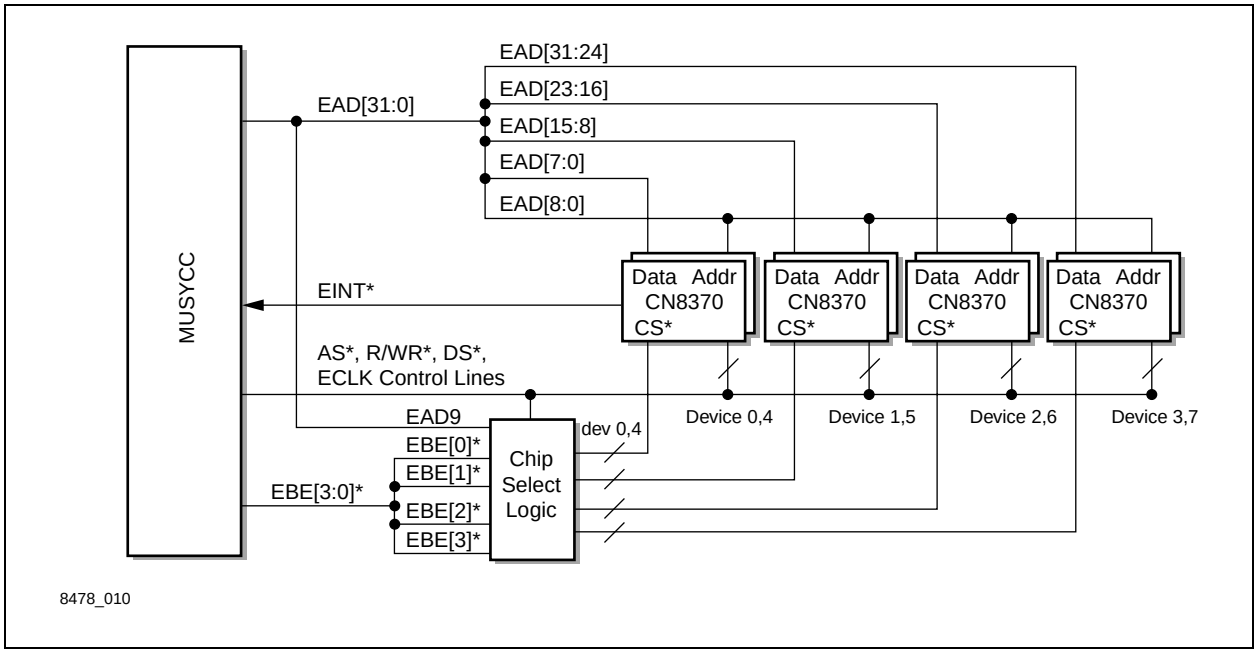
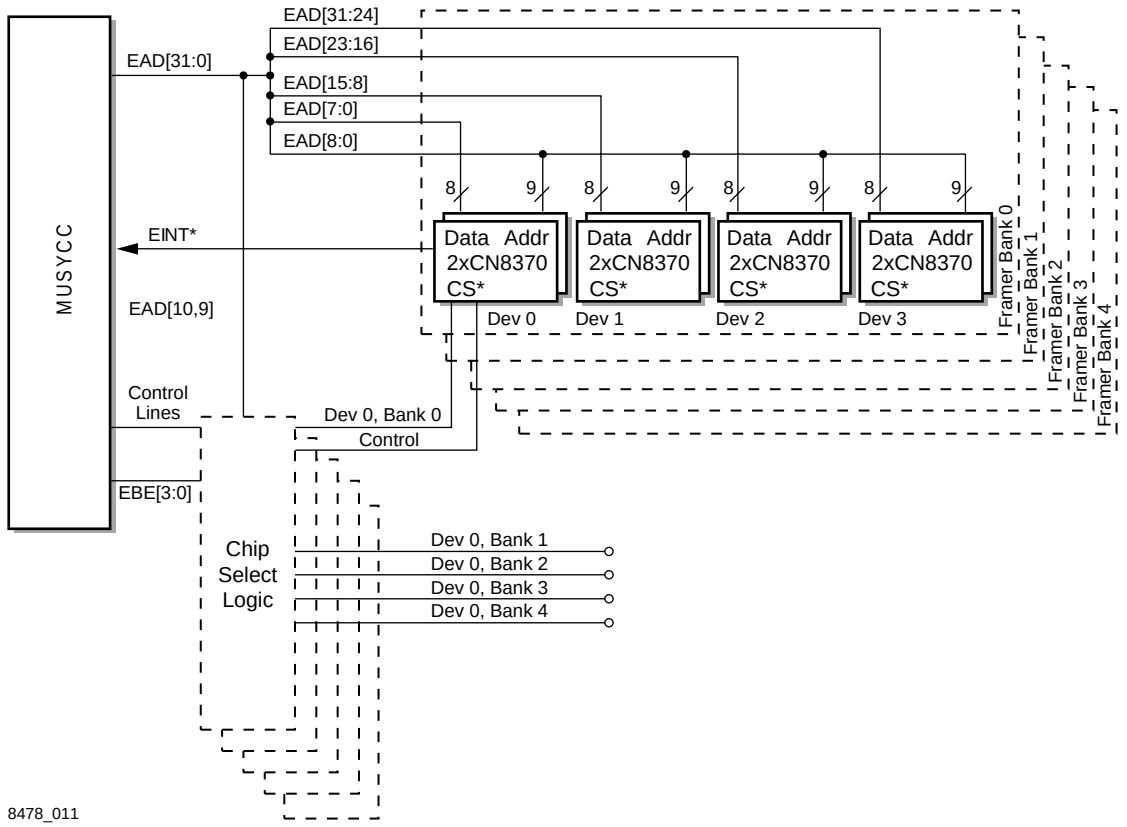


Figure 3-5. EBUS Connection, Non-multiplexed Address/Data, 61 Framers, No MPU



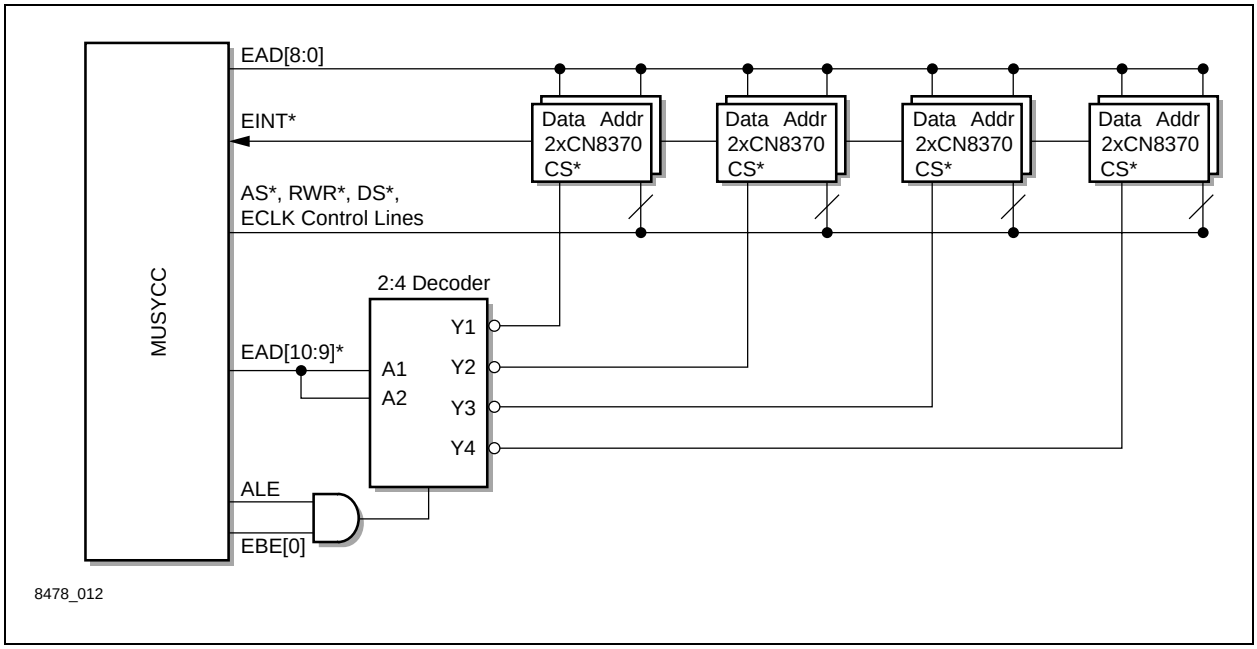
NOTE(S):
1. EBEx[3:0]* selects device x in each framer block.
2. EAD[31:0], AS* are supplied to each framer block.
3. EBEx*, AS* are supplied to each chip select block.

Figure 3-6 illustrates how additional address lines can be combined with each byte enable line during the address phase to support multiple framer banks with each bank containing four byte-wide framer devices.

In the multiplexed address and data mode, four byte-wide peripheral devices are connected to the EBUS. In this mode, 8 bits of the 32-bit EBUS transfer data to and from each device individually.

NOTE: The multiplexed address and data mode example does not allow for 4-byte data transfers.

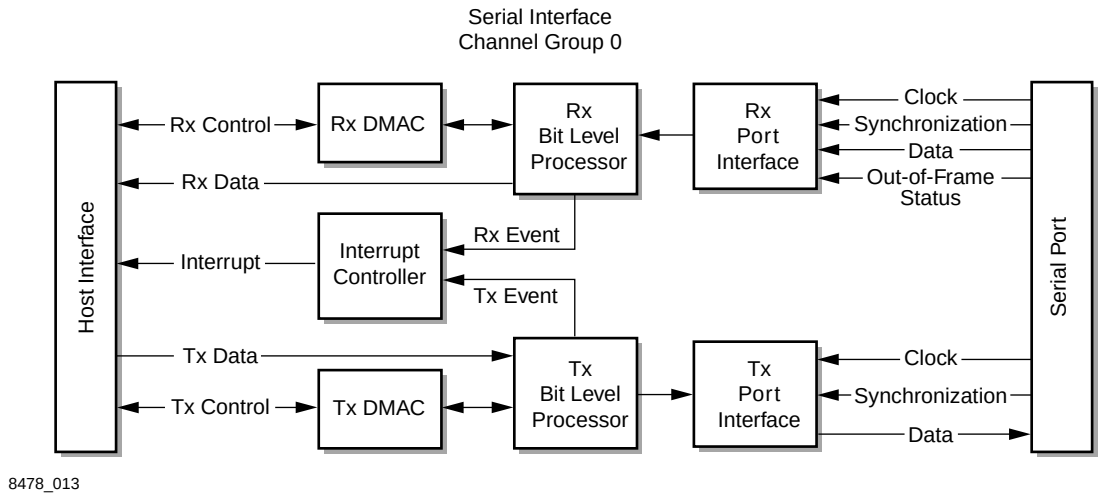
Figure 3-6. EBUS Connection, Multiplexed Address/Data, 8 Framers, No MPU



4.0 Serial Interface

Each serial interface consists of Serial Port Interfaces (SERI), Bit Level Processors (BLP), Direct Memory Access Controllers (DMAC), and an Interrupt Controller (INTC). A separate set of SERI, BLP, and DMAC services receive channels and transmit channels independently. A single INTC is shared by the receive and transmit BLP. Figure 4-1 illustrates the serial port/host interface.

Figure 4-1. Serial Interface Functional Block Diagram, Channel Group 0



NOTE(S):

1. Channel Groups 1, 2, 3, 4, 5, 6, and 7, when supported, are identical to Group 0.
2. Bt8478 supports Channel Groups 0 through 7.
3. Bt8474 supports Channel Groups 0, 1, 2, and 3.
4. Bt8472 supports Channel Groups 0 and 1.

4.1 Serial Port Interface

A receive serial port interface (Rx-SERI) connects to four input signals: RCLK, RDAT, RSYNC, and ROOF. A transmit serial port interface (Tx-SERI) connects to two input signals and one output signal, TCLK, TSYNC, and TDAT, respectively (refer to Table 1-4, *CN8478 Hardware Signal Definitions*). The SERI is responsible for receiving and transmitting data bits to FIFO buffers in the BLP.

The receive and transmit data and synchronization signals are synchronous to the receive and transmit line clocks, respectively. MUSYCC can be configured to sample in and latch out data signals and sample in status and synchronization signals on either the rising or falling edges of the respective line clock, RCLK and TCLK. This configuration is accomplished by setting the ROOF_EDGE, RSYNC_EDGE, RDAT_EDGE, TSYNC_EDGE, and TDAT_EDGE bit fields (detailed in Table 5-12, *Port Configuration Descriptor*).

The default, after device reset, is to sample in and latch out data, synchronization, and status on the falling edges of the respective line clock.

4.2 Bit Level Processor

The bit-level processors (Rx-BLP and Tx-BLP) service the bits in the receive and transmit path. As internal FIFO buffers are filled and flushed, the BLP requests memory transfers from the DMAC. The BLP coordinates all bit-level transactions between SERI and DMAC. The BLP also interacts with the INTC to notify the host of events and errors during bit-level processing.

4.3 DMA Controller

The DMA controllers (Rx-DMAC and Tx-DMAC) manage all memory operations between a corresponding BLP and the host interface. DMAC takes requests from BLP to either fill or flush internal FIFO buffers, sets up an access to data buffers in shared memory, and requests access to the PCI bus through the host interface.

4.4 Interrupt Controller

The interrupt controller takes receive and transmit events from Rx-BLP and Tx-BLP, respectively. The INTC coordinates the transfer of internally queued descriptors to an interrupt queue in shared memory and also coordinates the notification to the host of pending interrupts.

4.5 Channelized Port Mode

Each SERI can be configured independently using the PORTMD bit field (see Table 5-12, *Port Configuration Descriptor*).

Channelized mode refers to a data bit stream segmented into frames. Each frame consists of a series of 8-bit time slots. Typically, each time slot recurs every 125 µs at an 8 kHz rate. MUSYCC maintains frame synchronization in both the transmit and receive directions by using the TSYNC and RSYNC input signals. In addition, the ROOF input signal can be used to notify MUSYCC of the loss of frame synchronization.

Table 4-1 describes the contents of a typical 8 kHz frame in each of the possible channelized port modes.

Table 4-1. Channelized Serial Port Modes

Mode	Clock Frequency	Bits per Frame	Description
T1	1.544 MHz	193	Single frame bit, followed by 24 time slots, numbered TS0–TS23.
E1	2.048 MHz	256	32 time slots, numbered TS0–TS31.
2 E1	4.096 MHz	512	64 time slots, numbered TS0–TS63.
4 E1	8.192 MHz	1024	128 time slots, numbered TS0–TS127.
Nx128	Nx64 kHz ($1 \le N \le 128$)	Nx8 ($1 \le N \le 128$)	N time slots, numbered TS0–TSN-1.

4.5.1 Hyperchannels (Nx64)

A hyperchannel results from assigning bits from one or more 8-bit time slots within a frame. A hyperchannel can comprise from 1–128 time slots. This results in one logical channel supporting an Nx64 kbps bit rate where the actual data rate can range between 64 kbps and 8.192 Mbps. The concatenated time slots need not be contiguous.

Hyperchanneled time slots assigned to the same logical channel number within a channel group (0–31) are required for proper support.

The Time Slot Descriptor enables and assigns a time slot to a logical channel (see Table 5-15, *Time Slot Descriptor*). The configurations for receive and transmit hyperchannels are independent.

4.5.2 Subchannels (Nx8)

A subchannel results from treating each bit in an 8-bit time slot independently and assigning a logical channel number to each active bit. Not all 8 bits need to be active, and any combination of bits within the 8 in a time slot can be assigned to the same logical channel number. Similarly, multiple time slots can supply one or more bits to comprise one subchannel. This results in one logical channel

supporting an Nx8 bit rate between 8 kbps to 64 kbps in multiples of 8 kbps. The following configurations are required to support subchannels:

- Each active bit is assigned a logical channel number within a channel group (0–31).
- Each time slot with active bits must be enabled in the Time Slot Map.
- Each active bit (after the first bit, bit 0) must be enabled in the Subchannel Map.

The Time Slot Descriptor (Table 5-15), and the Subchannel Descriptor (Table 5-17), enable and assign a time slot and each individual bit within the time slot to a logical channel. The configurations for receive and transmit subchannels are independent.

The Time Slot Descriptor assigns bit 0 of a time slot to a logical channel. The Subchannel Descriptor assigns bits 1 through 7 of a time slot to a logical channel.

4.5.3 Frame Synchronization Flywheel

MUSYCC utilizes the TSYNC and RSYNC signals to maintain a timebase which keeps track of the active bit in the current time slot. The mechanism is referred to as the frame synchronization flywheel. The flywheel counts the number of bits per frame and automatically rolls over the bit count according to the programmed mode. The TSYNC or RSYNC input marks the first bit in the frame. The mode specified in the PORTMD bit field (Table 5-12, *Port Configuration Descriptor*), determines the number of bits in the frame. A flywheel exists for both the transmit and receive functions for every port.

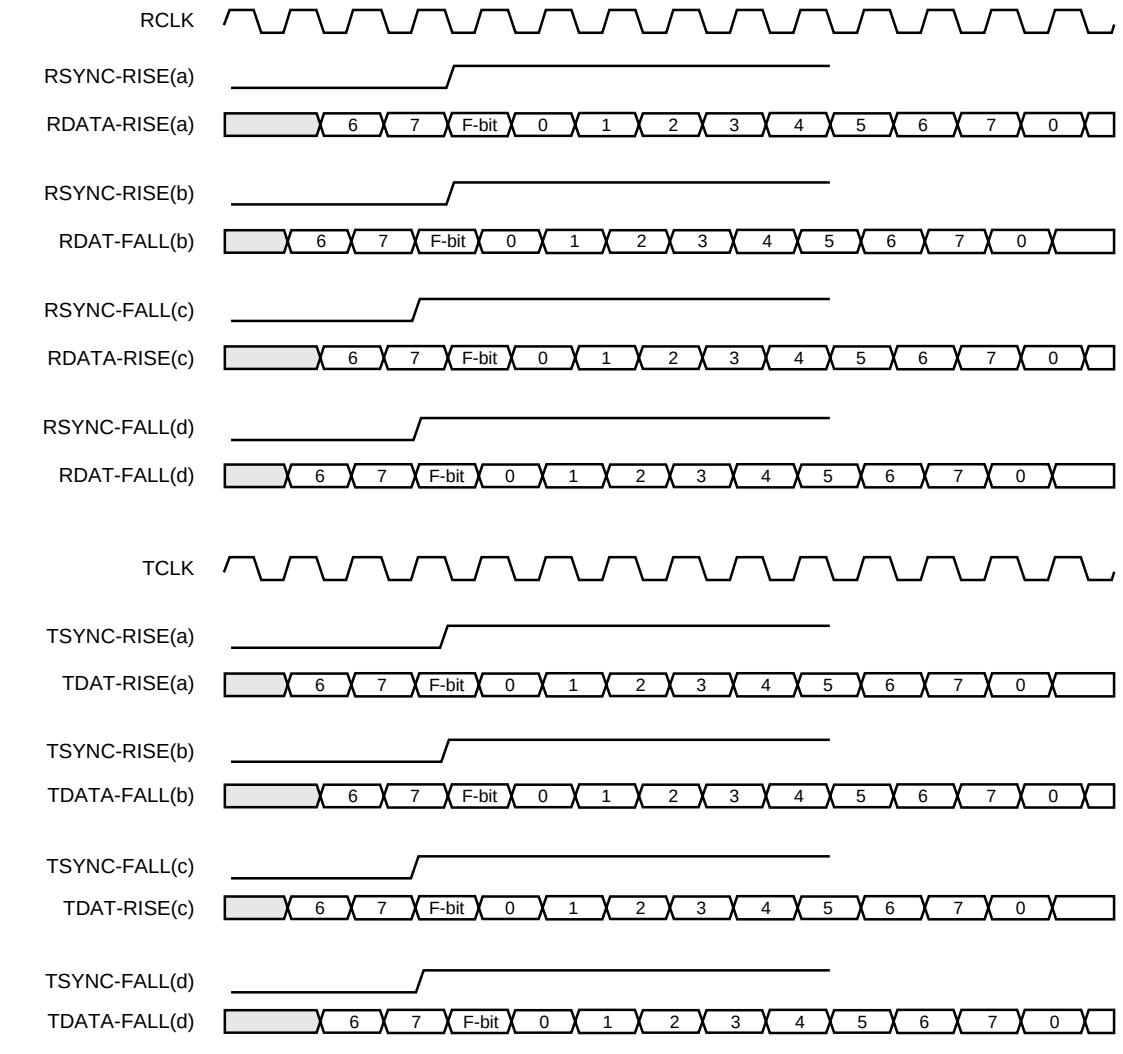
The flywheel is synchronized when MUSYCC detects TSYNC = 1 or RSYNC = 1, for transmit or receive functions, respectively. Once synchronized, the flywheel maintains synchronization without further assertion of the synchronization signal.

A time slot counter within each port is reset at the beginning of each frame and tracks the current time slot being serviced.

For the Nx64 mode, the value of N cannot be specified; therefore, the data requires a synchronization pulse every frame period to reset the flywheel. Also, in Nx64 mode, the TSYNC must precede the output of bit 0 of the frame by four line clock periods.

Figures 4-2 through 4-4 illustrate the timing relationships between the data and the synchronization signal for various modes of operation.

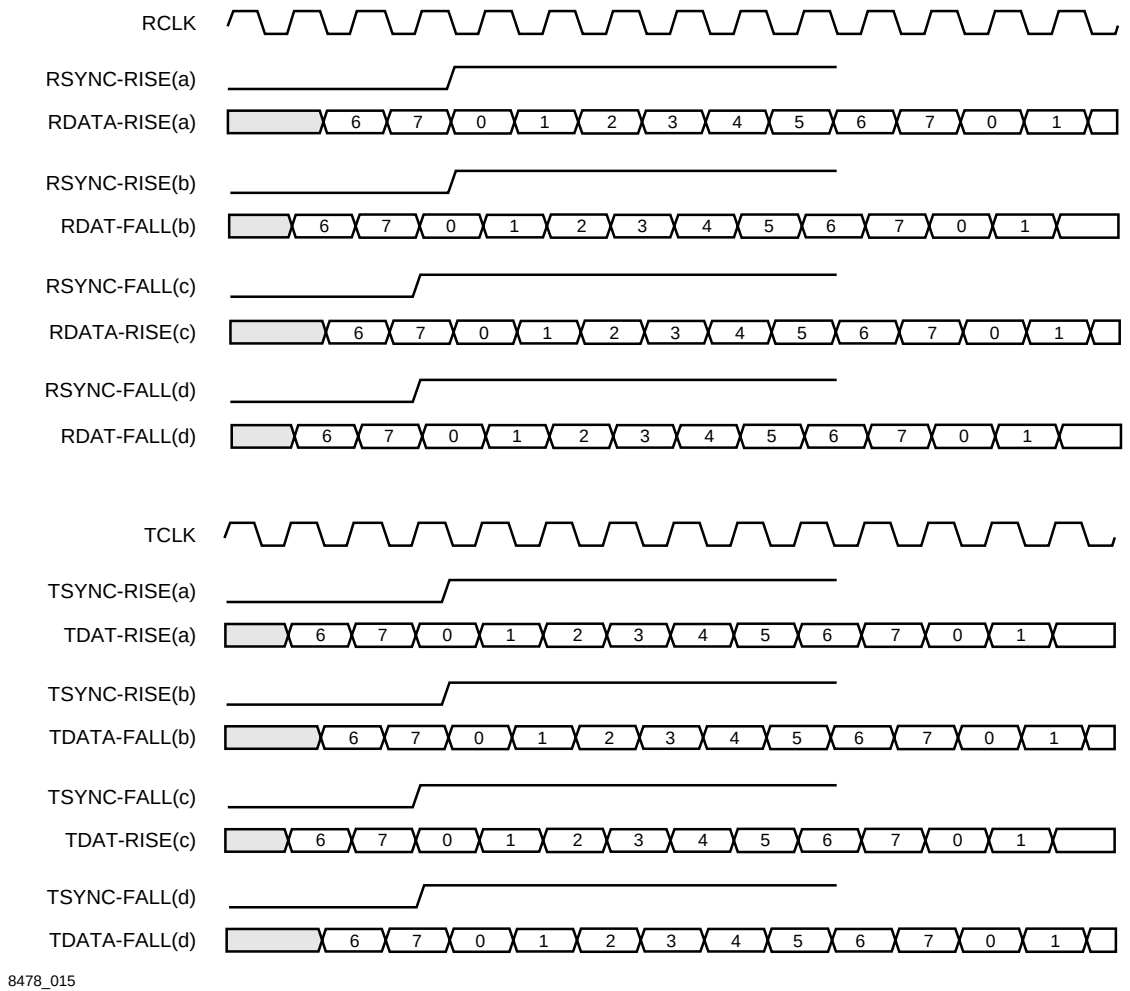
Figure 4-2. Transmit and Receive T1 Mode



NOTE(S):

1. T1 Mode employs 24 time slots (0–23) with 8 bits per time slot (0–7) and 1 frame bit every 193 clock periods. One frame of 193 bits occurs every 125 µs—1.544 MHz.
2. RSYNC and TSYNC must be asserted for a minimum of 1 CLK period.
3. MUSYCC can be configured to sample RSYNC, TSYNC, RDAT, and TDAT on either a rising or falling clock edge independent of any other signal sampling configuration.
4. Relationships between the various configurations of active edges for the synchronization signal and the data signal are shown using a common clock signal for receive and transmit operations. Note the relationship between the frame bit (within RDAT, TDAT) and the frame synchronization signal (e.g., RSYNC, TSYNC).
5. All received signals (e.g., RSYNC, RDAT, TSYNC) are sampled on the specified clock edge (e.g., RCLK, TCLK). All transmit data signals (TDAT) are latched on the specified clock edge.
6. In configuration (a), synchronization and data signals are sampled or latched on a rising clock edge.
7. In configuration (b), synchronization signal is sampled on a rising clock edge, and the data signal is sampled or latched on a falling clock edge.
8. In configuration (c), synchronization signal is sampled on a falling clock edge, and the data signal is sampled or latched on a rising clock edge.
9. In configuration (d), synchronization and data signals are sampled or latched on a falling clock edge.

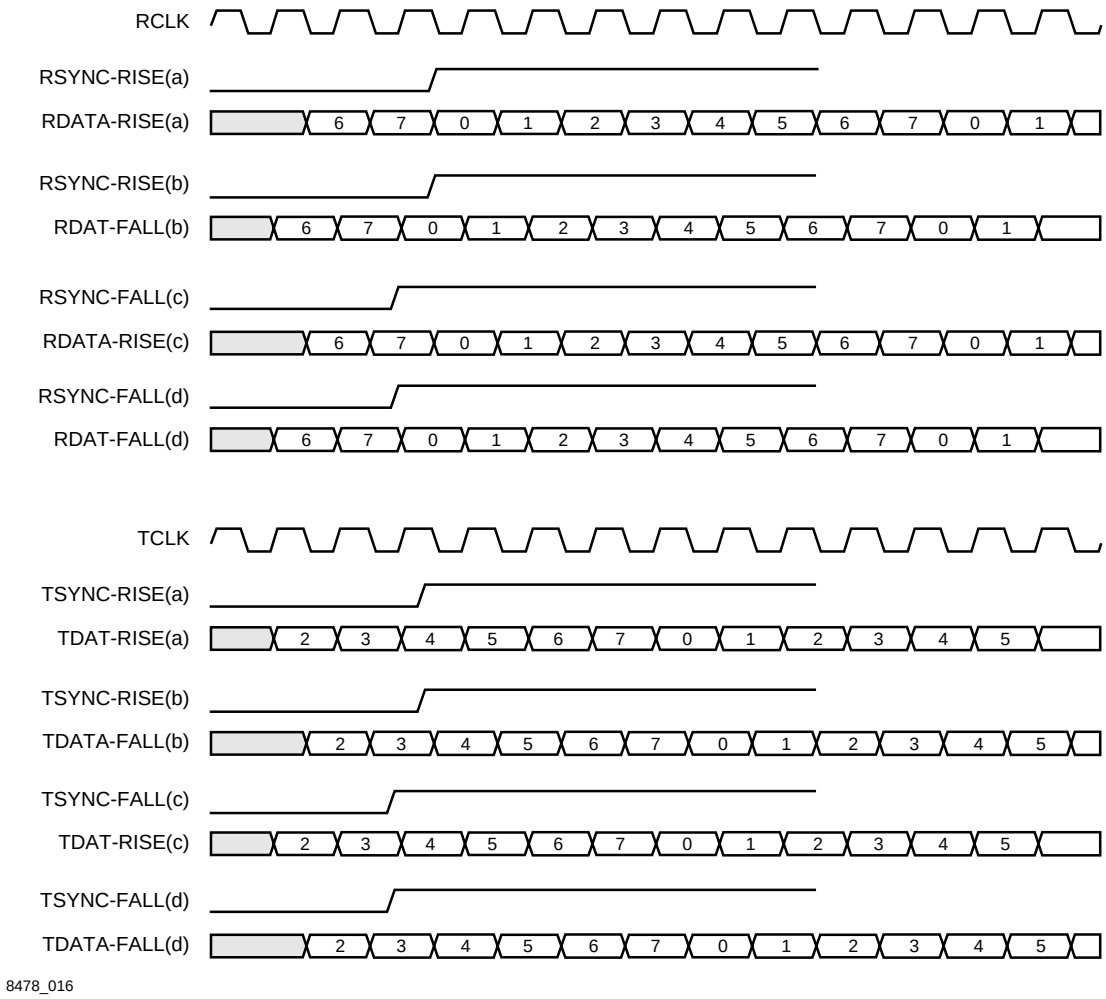
Figure 4-3. Transmit and Receive E1 (also 2xE1, 4xE1) Mode



NOTE(S):

1. E1 Mode employs 32 time slots (0–31) with 8 bits per time slot (0–7) and 256 bits per frame and one frame every 125 µs—2.048 MHz.
2. 2xE1 mode employs 64 time slots (0–63) with 8 bits per time slot (0–7) and 512 bits per frame and one frame every 125 µs—4.096 MHz.
3. 4xE1 mode employs 128 time slots (0–127) with 8 bits per time slot (0–7) and 1024 bits per frame and one frame every 125 µs—8.192 MHz.
4. RSYNC and TSYNC must be asserted for a minimum of 1 CLK period.
5. MUSYCC can be configured to sample RSYNC, TSYNC, RDAT, and TDAT on either a rising or falling clock edge independent of any other signal sampling configuration.
6. Relationships between the various configurations of active edges for the synchronization signal and the data signal are shown using a common clock signal for receive and transmit operations. Note the relationship between the frame bit (within RDAT, TDAT) and the frame synchronization signal (e.g., RSYNC, TSYNC).
7. All received signals (e.g., RSYNC, RDAT, TSYNC) are sampled on the specified clock edge (e.g., RCLK, TCLK). All transmit data signals (TDAT) are latched on the specified clock edge.
8. In configuration (a), synchronization and data signals are sampled or latched on a rising clock edge.
9. In configuration (b), synchronization signal is sampled on a rising clock edge, and the data signal is sampled or latched on a falling clock edge.
10. In configuration (c), synchronization signal is sampled on a falling clock edge, and the data signal is sampled or latched on a rising clock edge.
11. In configuration (d), synchronization and data signals are sampled or latched on a falling clock edge.

Figure 4-4. Transmit and Receive Nx64 Mode



NOTE(S):

1. Nx64 Mode employs N time slots with 8 bits (0–7) per time slot.
2. RSYNC and TSYNC must be asserted for a minimum of 1 CLK period.
3. Assertion of TSYNC must precede transmission of bit 0 of a frame by exactly 4 line clock periods due to the internal buffer scheme used for transmitting of Nx64 mode data bits.
4. RSYNC and TSYNC signals must be provided for every received and transmitted frame in Nx64 mode.
5. If N = 1, the minimum, then 8 bits/frame = 64 kHz. If N = 128, the maximum, then 1024 bits/frame = 8.192 MHz.
6. Relationships between the various configurations of active edges for the synchronization signal and the data signal are shown using a common clock signal for receive and transmit operations. Note the relationship between the frame bit (within RDAT, TDAT) and the frame synchronization signal (e.g., RSYNC, TSYNC).
7. All received signals (e.g., RSYNC, RDAT, TSYNC) are sampled on the specified clock edge (e.g., RCLK, TCLK). All transmit data signals (TDAT) are latched on the specified clock edge.
8. In configuration (a), synchronization and data signals are sampled or latched on a rising clock edge.
9. In configuration (b), synchronization signal is sampled on a rising clock edge, and the data signal is sampled or latched on a falling clock edge.
10. In configuration (c), synchronization signal is sampled on a falling clock edge, and the data signal is sampled or latched on a rising clock edge.
11. In configuration (d), synchronization and data signals are sampled or latched on a falling clock edge.

4.5.4 Change-of-Frame Alignment

A Change of Frame Alignment (COFA) condition is defined as a frame synchronization event detected when it is not expected, and includes the detection of the first occurrence of frame synchronization when none is present.

When the serial interface detects a COFA condition, an internal COFA signal is asserted for one frame period. During that period, MUSYCC's channel group processor terminates all active messages during the channel map processing.

For each receiver channel found to be active and processing a message during the RCOFA event, each of these channels' Buffer Status Descriptor is written with the COFA error encoding. The Buffer Status Descriptor is written if configured to do so in the Group Configuration Descriptor. MUSYCC then proceeds to the next Message Descriptor in the list of messages.

When the internal COFA is deasserted, MUSYCC generates an Interrupt Descriptor with the COFA error encoding if the interrupt is not masked in the Group Configuration Descriptor. If a synchronization signal is received (low-to-high transition on TSYNC or RSYNC) while the internal COFA is asserted, an Interrupt Descriptor with the COFA interrupt encoding is generated immediately if this interrupt is not masked. COFA detection is not applicable to the N x 64 serial port mode.

4.5.5 Out-of-Frame

The Receiver Out-of-Frame (ROOF) signal is asserted by the physical T1 or E1 interface sourcing the channelized data to MUSYCC. This signal indicates the interface device has lost frame synchronization.

In the case of multiplexed E1 lines (2xE1 or 4xE1), the ROOF input signal on a given port can be asserted and deasserted as time slots are received from an out-of-frame E1 followed by an in-frame E1.

The state of ROOF is evaluated on a bit-by-bit basis when processing data from a time slot. When ROOF assertion is detected by the receiver serial interface, MUSYCC checks the OOFABT bit in the Group Configuration Descriptor. If the OOFABT bit is set (i.e., 1), MUSYCC terminates any active messages for all mapped and active channels in the channel group. If the OOFABT bit is not set (i.e., 0), MUSYCC continues to process the received data but still asserts the OOF Interrupt Descriptor unless it is masked.

For each receive message terminated during the OOF condition, the corresponding Message Descriptor's owner bit is returned to the host, and a Buffer Status Descriptor is written with the OOF error encoding. The Buffer Status Descriptor is written to host memory only if configured to do so on a per group basis in the Group Configuration Descriptor.

MUSYCC then proceeds to the next Message Descriptor in the list of messages. Two frame synchronization events (via external sync or flywheel sync) after ROOF is asserted, MUSYCC generates an interrupt descriptor with the OOF error encoding if the interrupt is not masked in the Group Configuration Descriptor.

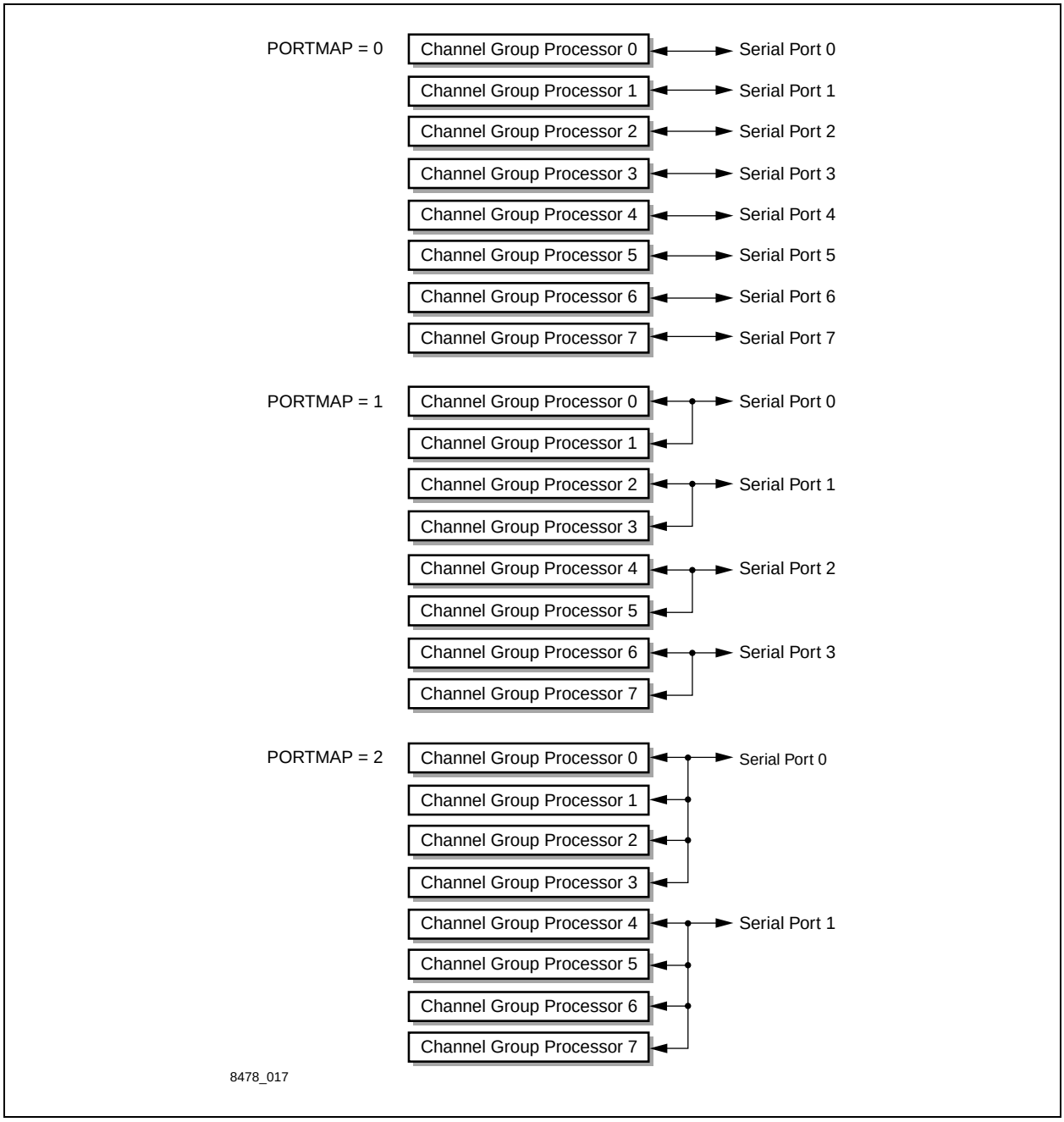
As ROOF is deasserted, MUSYCC immediately restarts normal bit level processing on all mapped and active channels. Two frame synchronization events after deassertion of ROOF is detected, MUSYCC generates an interrupt descriptor with the Frame Recovery (FREC) interrupt encoding if the interrupt is not masked (as indicated in Table 5-10, *Group Configuration Descriptor*).

4.6 Serial Port Mapping

MUSYCC contains up to eight serial ports with each port associated to a channel group processor that supports up to 32 logical bidirectional channels.

To manage more than 32 logical channels on a single port, the port configuration options provided in the PORTMAP bit field (as described in Table 5-6, *Global Configuration Descriptor*) can be programmed to route the signal on one port to multiple channel groups. Figure 4-5 illustrates the serial port mapping options.

Figure 4-5. Serial Port Mapping Options



The following port mappings are available:

- PORTMAP = 0, 1x port mode
  Default mode after device reset. Each serial port is logically connected to one channel group, and each port terminates up to 32 bidirectional channels.
- PORTMAP = 1, 2x port mode
  Each of two serial ports is logically connected to two channel groups. In this mode, serial port 0 is connected to channel groups 0 and 1, serial port 1 is connected to channel groups 2 and 3, serial port 2 is connected to channel group 4 and 5, and serial port 3 is connected to channel group 6 and 7. Each serial port terminates up to 64 bidirectional channels.
- PORTMAP = 2, 4x port mode
  Serial port 0 is logically connected to channel groups 0, 1, 2, and 3, and serial port 1 is logically connected to channel groups 4, 5, 6, and 7. Serial ports 2 through 7 are disabled.

Mapping a serial port to one or more logical channels in a channel group is one element of serial port configuration; another is indicating the channelized data rate of the serial interface, accomplished by configuring the PORTMAP bit field (Table 5-6, *Global Configuration Descriptor*). The connection between the serial port and the physical layer interface indicates the relationship to physical layer time slots.

Each serial port can be configured to support 24, 32, 64, 128 or a variable number of 8-bit time slots up to 128. Each serial port can be configured to support data rates up to and including 8.192 Mbps. Also, each serial port can be independently wired to a separate source of serial data, or all four serial ports can be wired to a single source of serial data.

4.7 Tx and Rx FIFO Buffer Allocation and Management

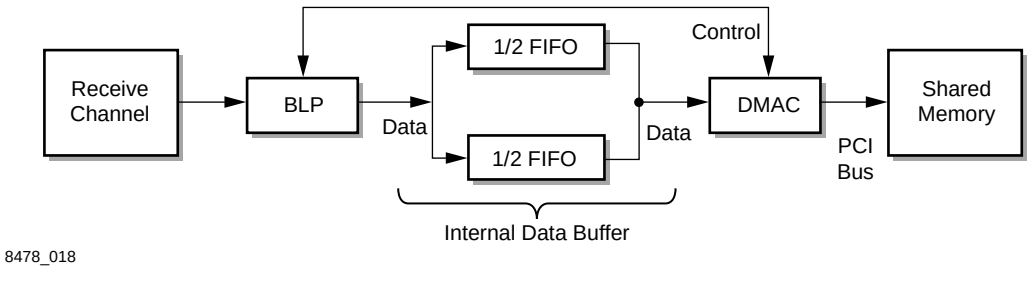
Each channel group contains a separate internal buffer memory space for transmit and receive operations. Within each of these spaces, separate areas are set aside for specific functions.

Each channel within the group must be allocated buffer space before the channel can be activated. Table 4-2 lists the internal buffer memory allocation. This space acts as a holding buffer for incoming (Rx) and outgoing (Tx) data. Data buffers for each channel are allocated using the BUFFLOC and BUFFLEN bit fields (Table 5-18, *Channel Configuration Descriptor*). Both receiver and transmitter of a channel use a data buffer scheme where half the available FIFO services the serial interface, and the other half services data in shared memory. Figures 4-6 and 4-7 illustrate the receive and transmit data flows, respectively. BUFFLEN+1 specifies half the size of the buffer space allocated to a direction of the channel.

Table 4-2. Internal Buffer Memory Layout

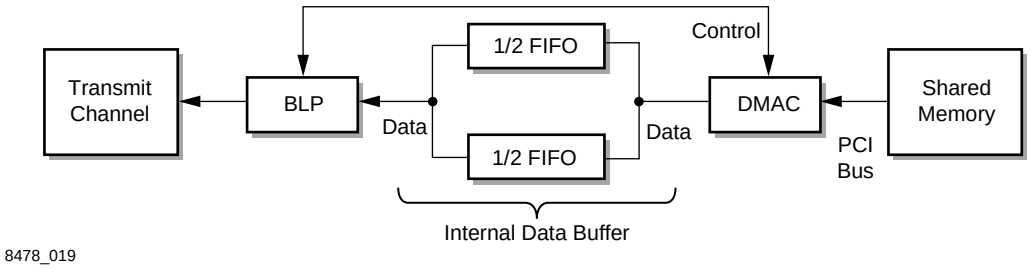
Memory Area	Transmit	Receive
Fixed Data Buffer	64 dwords	64 dwords
Subchannel Map (or Additional Data Buffer if No Subchanneling)	64 dwords	64 dwords
Time Slot Map	32 dwords	32 dwords
Total	160 dwords 640 bytes	160 dwords 640 bytes

Figure 4-6. Receive Data Flow



NOTE(S): 1/2 FIFO = BUFFLEN+1

Figure 4-7. Transmit Data Flow



NOTE(S): 1/2 FIFO = BUFFLEN+1

The allocation of internal data buffers requires an understanding of how the total available FIFO buffer space depends on whether subchannels are enabled within that channel group. Table 4-2 specifies 64 dwords of internal data buffer are available to allocate as FIFO buffer space among the 32 channels of each channel group when any channel within that group is configured to operate as a subchannel (SUBDSBL = 0 in the Group Configuration Descriptor). Table 4-2 further specifies that an additional 64 dwords of internal data buffer (128 dwords total) are available to allocate as FIFO buffer space among the channel group by reusing the subchannel map area when all subchanneling within that group is disabled (SUBDSBL = 1).

Other important considerations for allocating internal data buffers include the number of active channels per group, the channels' data rate, and the channels' PCI latency tolerance. Examples given later in this section describe scenarios where all available internal data buffer space is evenly distributed to form equal length FIFO buffers for each channel in the group, presuming each channel operates at the same data rate, and there are a variable number of channels per group. However, internal data buffer allocation is flexible and allows the host to assign larger FIFO buffers to channels operating at higher data rates. For applications operating high speed channels (i.e., hyperchannels), the host typically allocates 2 dwords (64 bits) of internal data buffer for each 64 kbps increment in the channel's data rate. For example, a 1920 kbps hyperchannel consisting of 30 time slots would typically be allocated 60 dwords of FIFO buffer space. Smaller FIFO buffers can be allocated if there are multiple, high-speed channels configured within one group, but at the expense of some PCI latency tolerance.

PCI latency tolerance equals the maximum length of time a particular channel can operate normally between PCI bus transactions without reaching an internal buffer overflow or underflow condition. Therefore, PCI latency tolerance is primarily dependent on each channel's FIFO buffer size. Because of MUSYCC's internal data buffer scheme, each transmit channel's PCI latency tolerance is expressed as the amount of time required to send data from half the FIFO buffer size [(i.e., (BUFFLEN + 1) dwords)]. While a receive channel's PCI latency tolerance is expressed as 1/2 FIFO buffer size plus 1 additional dword [i.e., (BUFFLEN + 2) dwords]. A 64 kbps channel that is allocated 4 dword transmit and receive FIFO buffers can tolerate up to 2 dwords (1 ms) of bus latency in the transmit direction and 3 dwords (1.5 ms) of bus latency in the receive direction.

4.7.1 Example Channel BUFFLOC and BUFFLEN Specification

With some subchanneling, only the Fixed Data Buffer (total of 64 dwords) is the area available for Internal Data Buffer usage. If the buffer space is evenly divided across 32 channels, the BUFFLOC and BUFFLEN specifications would be as described in Table 5-18, *Channel Configuration Descriptor*. Table 4-3 lists the subchannel buffer allocation on 32 channels.

Table 4-3. Example of 32-Channel with Subchanneling Buffer Allocation (Receive or Transmit)

Channel Number	Within Channel Descriptor	
	BUFFLOC (dword Offset from Start of Fixed Data Buffer)	BUFFLEN(1)
0	0	0
1	1	0
2	2	0
...	...	...
31	31	0(2)

NOTE(S):

(1) Assuming all channels within a group operate at the same bit rate, BUFFLEN = [(Total dwords ÷ Number of Channels) ÷ 2]–1.

(2) BUFFLEN values larger than 1Fh do not increase the PCI burst length. BUFFLEN determines the number of dwords burst during a PCI read/write operation to fill or flush the internal data buffer. For example, BUFFLEN = 1Fh specifies a burst length of 32 dwords.

With no subchanneling, the Fixed Data Buffer area plus the Subchannel Map area are available for Internal Data Buffer usage (total of 128 dwords). If the buffer space is evenly divided across 32 channels, the BUFFLOC and BUFFLEN specification is as listed in Table 4-4, for 32 channels without subchannel buffer allocation.

Table 4-4. Example of 32-Channel without Subchanneling Buffer Allocation (Receive or Transmit)

Channel Number	Within Channel Descriptor	
	BUFFLOC (dword Offset from Start of Fixed Data Buffer)	BUFFLEN(1)
0	0	1
1	2	1
2	4	1
...	...	...
31	62	1(2)

NOTE(S):
(1) Assuming all channels within a group operate at the same bit rate, BUFFLEN = [(Total dwords ÷ Number of Channels) ÷ 2]–1.
(2) BUFFLEN values larger than 1Fh do not increase the PCI burst length. BUFFLEN determines the number of dwords burst during a PCI read/write operation to fill or flush the internal data buffer. For example, BUFFLEN = 1Fh specifies a burst length of 32 dwords.

If the buffer space is evenly divided across 16 channels, the BUFFLOC and BUFFLEN specification would be as listed in Table 4-5, for 16 channels with subchannel buffer allocation.

Table 4-5. Example of 16-Channel without Subchanneling Buffer Allocation (Receive or Transmit)

Channel Number	Within Channel Descriptor	
	BUFFLOC (dword Offset from Start of Fixed Data Buffer)	BUFFLEN(1)
0	0	3
1	4	3
2	8	3
...	...	...
15	60	3(2)

NOTE(S):
(1) Assuming all channels within a group operate at the same bit rate, BUFFLEN = [(Total dwords ÷ Number of Channels) ÷ 2]–1.
(2) BUFFLEN values larger than 1Fh do not increase the PCI burst length. BUFFLEN determines the number of dwords burst during a PCI read/write operation to fill or flush the internal data buffer. For example, BUFFLEN = 1Fh specifies a burst length of 32 dwords.

4.7.2 Receiving Bit Stream

As a receive channel is activated, MUSYCC reads in descriptors from shared memory and prepares Rx-BLP and Rx-DMAC to service incoming serial data accordingly, assuming all configurations are proper, and incoming data can be written to shared memory.

Upon channel activation, the receiver starts storing received data into a BUFFLEN+1 size of FIFO, starting at BUFFLOC offset in the FIFO buffer area. As this buffer fills, the BLP instructs the DMAC to start a PCI data transfer cycle to shared memory of the FIFO buffer contents and simultaneously starts filling another BUFFLEN+1 size of FIFO buffer from the serial port. Generally, half the FIFO buffer space for a channel is used for serial port data reception, and half for shared memory data transfers.

The DMAC-initiated PCI transfer cycle requires MUSYCC to arbitrate for the PCI bus, initiate a master write to shared memory over the PCI bus, and conclude the transfer by releasing the PCI bus. MUSYCC transfers data autonomously and always attempts to burst data to the PCI.

4.7.3 Transmitting Bit Stream

When a transmit channel is activated, MUSYCC reads in descriptors from shared memory and prepares Tx-BLP and Tx-DMAC to service outgoing serial data, assuming all configurations are proper, and outgoing data can be read from shared memory.

Upon channel activation, the transmitter initiates a PCI data transfer cycle from shared memory of data to be output to the serial port. As the DMAC receives data over the PCI, it forwards it to the BLP which fills a BUFFLEN+1 size of FIFO starting at BUFFLOC offset in the FIFO area. Generally, half the FIFO space for a channel is used for serial port data transmission and half for shared memory data transfers.

The DMAC-initiated PCI transfer cycle requires that MUSYCC arbitrate for the PCI bus, initiate a master read from shared memory over the PCI bus, and conclude the transfer by releasing the PCI bus. MUSYCC transfers data autonomously and always attempts to burst data from the PCI.

4.7.3.1 Transmit Data Bit Output Value Determination

The TDAT signal from MUSYCC is the only output signal in the serial interface. For each bit time specified by the TCLK input signal to MUSYCC and on each active edge for a data bit specified by the TDAT_EDGE bit field, a value for the TDAT bit must be determined and output (Table 5-12, *Port Configuration Descriptor*). Figure 4-8 illustrates the logic used to determine the output value.

Figure 4-8. Transmit Data Bit Output Value Determination

```
if ( TRANSMITTER_NOT_ENABLED )(1)
    TDAT <= three-state
else
    if ( CHANNEL_IS_MAPPED )(2)
        if ( CHANNEL_IS_ACTIVATED )(3)
            TDAT = BLP_OUTPUT(4)
        else
            TDAT = `logic 1'
    else
        if ( THREE_STATE_OUTPUT )(5)
            TDAT = three-state
        else
            TDAT = `logic 1'
```

8478_020

NOTE(S):
(1) TRANSMITTER_NOT_ENABLED. (Check TXENBL bit field in Table 5-10, *Group Configuration Descriptor*.)
(2) CHANNEL_IS_MAPPED. (Verify channel to time slot mapping enabled in Table 5-14, *Transmit or Receive Time Slot Map*.)
(3) CHANNEL_IS_ACTIVATED. Verify Channel Activate Service Request issued.
(4) BLP_OUTUT. Data taken from shared memory, through the internal FIFO and ready for transmission.
(5) THREE_STATE_OUTPUT. Check TRITX bit field in Port Configuration Descriptor.

5.0 Memory Organization

MUSYCC interfaces with a system host using a set of data structures located in a shared memory region. MUSYCC also contains a set of internal registers which the host can configure and which controls MUSYCC. This section describes the various shared memory data structures and the layout of individual registers which are required for the operation of MUSYCC.

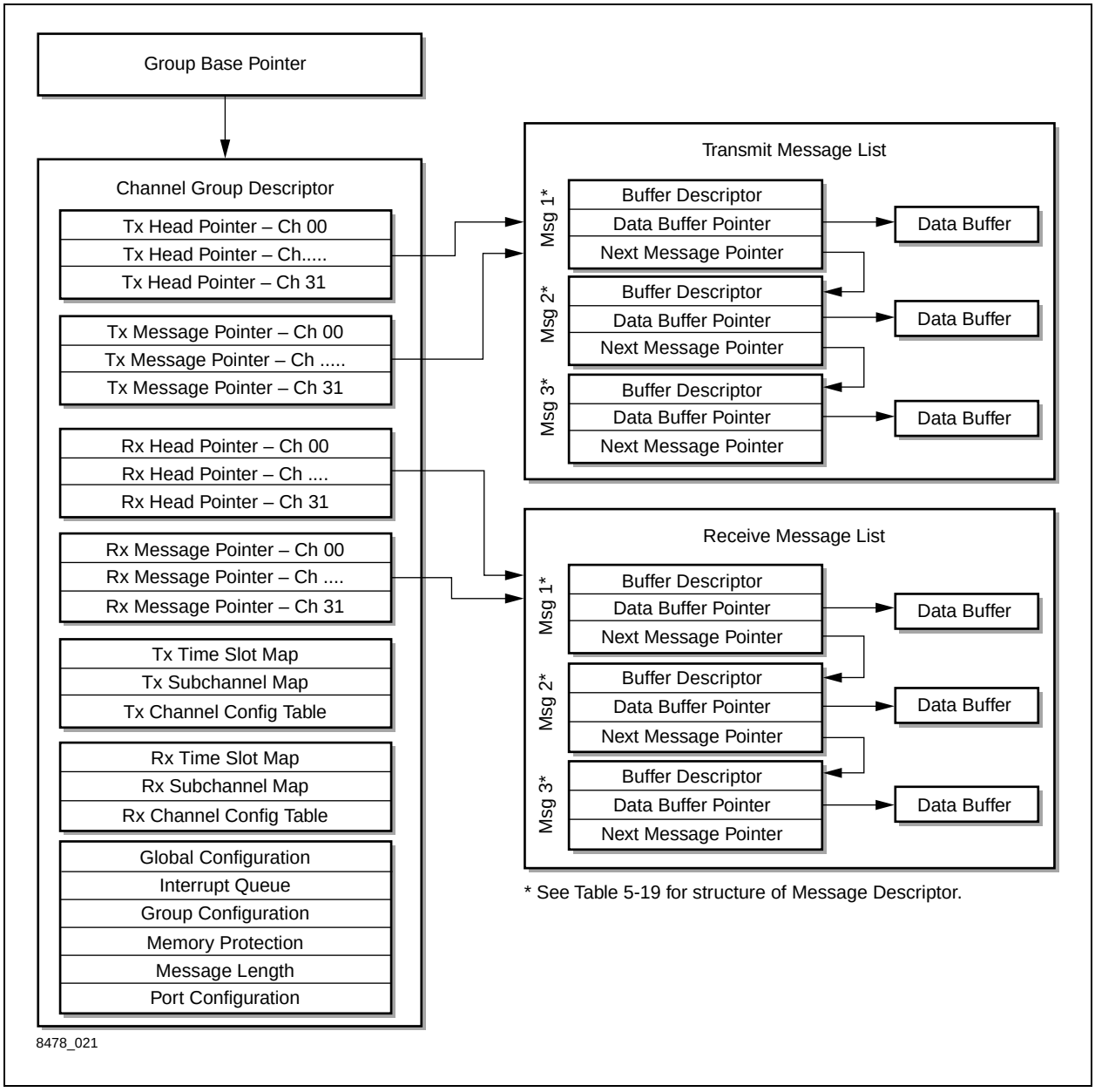
5.1 Memory Architecture

MUSYCC supports a memory model whereby data is continually moved into and out of a linked list of data buffers in shared memory for each active channel. This assumes a system topology in which a host and MUSYCC both have access to shared memory for data control and data flow. The data structures are defined in a way that the control structures and the data structures may or may not reside in the same physical memory and may or may not be contiguous. The host allocates and deallocates the required memory space as well as the size and number of data buffers within that space.

Different versions of MUSYCC support different numbers of channel groups. The host allocates shared memory regions to configure and control each group.

Figure 5-1 illustrates the memory model used by MUSYCC for control and data structures required for each supported channel group.

Figure 5-1. Shared Memory Model Per Channel Group



5.1.1 Register Map Access and Shared Memory Access

During MUSYCC's PCI initialization, the system controller allocates a dedicated 1 MB memory range to each of MUSYCC's PCI functions. The memory range allocated to MUSYCC must not map to any other physical or shared memory. Instead, the system configuration manager allocates a logical memory address range, and notifies the system or bus controllers that any access to these ranges must result in a PCI access cycle. MUSYCC is assigned these address ranges for each function through the PCI configuration cycle. Once configured, MUSYCC becomes a functional PCI device on the bus.

As the host accesses MUSYCC's allocated address ranges, it initiates the access cycles on the PCI bus. It is up to individual MUSYCC devices on the bus to claim the access cycle. As its address ranges are accessed, MUSYCC behaves as a PCI slave device while data is being read or written by the host. MUSYCC responds to all access cycles where the upper 12 bits of a PCI address match the upper 12 bits of either the EBUS Base Address register (Function 1) or the MUSYCC Base Address register (Function 0).

For MUSYCC's Function 1, a 1 MB memory space is assigned to the EBUS Base Address register which is written into Function 1 PCI configuration space (Table 2-14, *Register 4, Address 10h*). Devices connected to the EBUS can then be allocated memory addresses within this 1 MB memory range. If MUSYCC claims a PCI access cycle for Function 1, MUSYCC initiates EBUS arbitration and ultimately accesses data from a device connected to the EBUS.

For MUSYCC's Function 0, a 1 MB memory space is assigned to the MUSYCC Base Address register which is written into Function 0 PCI configuration space (Table 2-7, *Register 4, Address 10h*). Once a base address is assigned to Function 0, a register map is used to access individual device resident registers. The register map provides the byte offset from the Base Address register where registers reside. The register map layout is listed in Table 5-1.

The 1 MB memory ranges assigned to MUSYCC functions will not restrict MUSYCC's PCI interface from attempting to access these ranges. The host must be cognizant that MUSYCC cannot respond to an access cycle which MUSYCC itself initiates as the bus master.

Table 5-1. MUSYCC Register Map

Register Map	Group (Byte Offset from Base Address Register)							
	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Group Base Pointer	0000h	0800h	1000h	1800h	2000h	2800h	3000h	3800h
Dual Address Cycle Base Pointer(1)	00004h							
Service Request Descriptor	0008h	0808h	1008h	1808h	2008h	2808h	3008h	3808h
Interrupt Status Descriptor(1)	000Ch							
Transmit Time Slot Map(2)	0200h	0A00h	1200h	1A00h	2200h	2A00h	3200h	3A00h
Transmit Subchannel Map(2)	0280h	0A80h	1280h	1A80h	2280h	2A80h	3280h	3A80h
Transmit Channel Configuration Table(2)	0380h	0B80h	1380h	1B80h	2380h	2B80h	3380h	3B80h
Receive Time Slot Map(2)	0400h	0C00h	1400h	1C00h	2400h	2C00h	3400h	3C00h
Receive Subchannel Map(2)	0480h	0C80h	1480h	1C80h	2480h	2C80h	3480h	3C80h
Receive Channel Configuration Table(2)	0580h	0D80h	1580h	1D80h	2580h	2D80h	3580h	3D80h
Global Configuration Descriptor(1)	00600h							
Interrupt Queue Descriptor(1)	00604h							
Group Configuration Descriptor	060Ch	0E0Ch	160Ch	1E0Ch	260Ch	2E0Ch	360Ch	3E0Ch
Memory Protection Descriptor	0610h	0E10h	1610h	1E10h	2610h	2E10h	3610h	3E10h
Message Length Descriptor	0614h	0E14h	1614h	1E14h	2614h	2E14h	3614h	3E14h
Port Configuration Descriptor	0618h	0E18h	1618h	1E18h	2618h	2E18h	3618h	3E18h
Receive BIST Status(3)	00640h							
Transmit BIST Status(3)	00644h							

NOTE(S):

(1) MUSYCC automatically maps Group 1 through 7 addresses for these registers to the Group 0 address (shown). For example, accessing address 00E00h in MUSYCC (address for Group 1 Global Configuration register) automatically maps to address 00600h and the contents of 00600h is read or written.

(2) The following descriptors are mapped to Internal RAM: Transmit Time Slot Map, Transmit Subchannel Map, Transmit Channel Configuration Table, Receive Time Slot Map, Receive Subchannel Map, and Receive Configuration Table. Host must not access internal RAM while channels are active. Updates to RAM must be performed via a service request.

(3) The receive/transmit BIST diagnostic status registers.

The first four registers in each group (shown in bold-type in Table 5-1) are located exclusively within MUSYCC. These registers are accessed by the host using direct reads and writes to the corresponding register map address. The remaining registers have corresponding locations within shared memory, and the host accesses the shared memory image rather than the internal registers. Regardless, the values within MUSYCC are always the values used during device operation. After configuring the shared memory image of these registers, the host issues a service request by writing directly into the Service Request Descriptor. This causes MUSYCC to copy the image from shared memory.

Each supported channel group requires its own group structure to operate. The Dual Address Cycle Base Pointer, Interrupt Status Descriptor, Global Configuration Descriptor, and the Interrupt Queue Descriptor are common among all supported groups.

The Transmit Time Slot Map and the Transmit Subchannel Map are write-only areas within MUSYCC; reading from these areas results in all 1s being returned.

The Service Request Descriptors are locations within MUSYCC where commands can be directed to individual channel groups. The host writes a service request (a command) directly into the corresponding group's register. MUSYCC behaves as a PCI slave as this write is performed. The action resulting from the command may cause MUSYCC to read or write locations from shared memory. While MUSYCC accesses shared memory, it behaves as a PCI master and arbitrates for control of the bus autonomously.

MUSYCC's registers can be initialized before or after shared memory resident descriptors are initialized. The recommended sequence is to configure shared memory descriptors first, then copy the relevant information to MUSYCC's registers via the service request mechanism.

NOTE: Upon channel activation, shared memory and internal registers must be initialized, valid, and available to MUSYCC. MUSYCC uses the information within the shared memory descriptors to transfer data between the serial interface and shared memory. MUSYCC assumes the information is valid once a channel is activated.

The first four sets of pointers for each channel group, listed in Table 5-2, Group Structure Memory Map, are pointer locations exclusive to shared memory. MUSYCC does not keep these values internally although they are accessed regularly during channel processing. The remaining locations have a corresponding register within MUSYCC.

Table 5-2. Group Structure Memory Map

Channel Group Memory Map	Byte Offset from Respective Group Base Pointer	Length (Bytes)
Transmit Head Pointers	00000h	128
Transmit Message Pointers	00080h	128
Receive Head Pointers	00100h	128
Receive Message Pointers	00180h	128
Transmit Time Slot Map	00200h	128
Transmit Sub Channel Map	00280h	256
Transmit Channel Configuration Table	00380h	128
Receive Time Slot Map	00400h	128
Receive Sub Channel Map	00480h	256
Receive Channel Configuration Table	00580h	128
Global Configuration Descriptor	00600h	4
Interrupt Queue Descriptor	00604h	8
Group Configuration Descriptor	0060Ch	4
Memory Protection Descriptor	00610h	4
Message Length Descriptor	00614h	4
Port Configuration Descriptor	00618h	4
	Total Space Required	1564

5.1.2 Memory Access Illustration

Assume the system memory controller (or the host) allocates addresses for MUSYCC's PCI functions as listed in Table 5-3.

Table 5-3. MUSYCC PCI Function Memory Allocation

System Allocated MUSYCC Memory Ranges	Start Address	End Address	Length
MUSYCC - Function 0- Base Address Register	0240 0000h	024F FFFFh	1 MB
EBUS - Function 1- Base Address Register	0340 0000h	034F FFFFh	1 MB

The Base Address is written into MUSYCC by the host-initiated PCI configuration access write cycles. After MUSYCC functions are memory-mapped to PCI space, the host allocates shared memory space for each supported channel group descriptors. It requires each Group Base Pointer to start on a 2 kB boundary, as listed in Table 5-4 and Table 5-8, *Group Base Pointer*.

Table 5-4. Shared Memory Allocation—Group Descriptors

Channel Group Descriptors	Start Address	End Address	Length
Group 0 Base Pointer	0090 0000h	0090 061Ch	1,564 bytes
Group 1 Base Pointer	0090 0800h	0090 0E1Ch	1,564 bytes
Group 2 Base Pointer	0090 1000h	0090 161Ch	1,564 bytes
Group 3 Base Pointer	0090 1800h	0090 1E1Ch	1,564 bytes
Group 4 Base Pointer	0090 2000h	0090 261Ch	1,564 bytes
Group 5 Base Pointer	0090 2800h	0090 2E1Ch	1,564 bytes
Group 6 Base Pointer	0090 3000h	0090 361Ch	1,564 bytes
Group 7 Base Pointer	0090 3800h	0090 3E1Ch	1,564 bytes

The Group Base Pointer value is written into MUSYCC by the host via PCI write access cycles. The location of the Group Base Pointer register for each group within MUSYCC is listed in Table 5-1.

For this illustration, the host must perform the following write operations, as listed in Table 5-5.

Table 5-5. Host Assigns Group Base Pointers

Host Writes Pointer Value	To PCI Address
0090 0000h	0240 0000h
0090 0800h	0240 0800h
0090 1000h	0240 1000h
0090 1800h	02401800h
0090 2000h	0240 2000h
0090 2800h	0240 2800h
0090 3000h	0240 3000h
0090 3800h	02403800h

Next, the host allocates the required shared memory for transmit and receive messages. Assume, for example, the host needs 8 message descriptors for each channel and direction, and each corresponding data buffer per message is 100h (256) bytes in length.

Memory for Message Descriptors =
32 channels/group *
2 directions/channel *
12 bytes/message descriptor *
8 buffers/channel
= 1800h bytes/group
= 6144 bytes/group

Memory for Data Buffers =
32 channels/group *
2 directions/channel *
8 buffers/channel *
256 bytes/buffer
= 131, 072 bytes/group
= 20,000h bytes/group

Further, the host may choose to allocate all the memory contiguously, or it may allocate the memory for message descriptors separately from data buffers. In this case, message descriptors for 8 Channel Groups may be merged into a contiguous block of memory [(1800h x 8 = C000h) bytes in length].

5.2 Descriptors

This section further details the descriptors specified in the MUSYCC memory model. The MUSYCC descriptors are as follows:

1. Host Interface Level Descriptors (see Section 5.2.1)
   - Global Configuration
   - Dual Address Cycle Base Pointer
2. Channel Group Level Descriptors (see Section 5.2.2)
   - Group Base Pointer
   - Service Request
   - Group Configuration
   - Memory Protection
   - Port Configuration
   - Message Length
   - Time Slot Map
   - Subchannel Map
3. Channel Level Descriptors (see Section 5.2.3)
   - Channel Configuration (see Section 5.2.4)
4. Message Level Descriptor
   - Head Pointer
   - Message Pointer
   - Message Descriptor
   - Buffer Descriptor
   - Buffer Status Descriptor
   - Next Message Pointer
   - Data Buffer Pointer
   - Message Descriptor Handling
5. Interrupt Level Descriptors (see Section 5.2.5)
   - Interrupt Queue Descriptor
   - Interrupt Descriptor
   - Interrupt Status Descriptor

Each of the above entities are allocated, deallocated, read from, and written to by the host. MUSYCC can read all of these entities as well, but can only write to these:

- Message Pointers
- Buffer Status Descriptors
- Receive Data Buffers

5.2.1 Host Interface Level Descriptors

Host interface-level descriptors contain information necessary to configure the global registers. This information applies to the entire device, including all channel groups, serial ports, and channels.

5.2.1.1 Global Configuration Descriptor

The Global Configuration Descriptor specifies configuration information applying to the entire device including all channel groups, serial ports, and channels.

Memory space is reserved for the Global Configuration Descriptor within each Channel Group Descriptor. By convention, the values corresponding to Channel Group 0 (a group present in all versions of MUSYCC) provides the correct data. The host coordinates how this data is transferred into MUSYCC by:

- Instructing MUSYCC to read the Channel Group 0 Global Configuration Descriptor when setting the global data.
- Copying the Channel Group 0 Global Configuration Descriptor to all other supported Channel Group Descriptors and requesting a global initialization service request operation for any supported channel groups.

The components and their descriptions are listed in Table 5-6.

Table 5-6. Global Configuration Descriptor *(1 of 2)*

Bit Field	Name	Value	Description
31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16	TCLKACT7 TCLKACT6 TCLKACT5 TCLKACT4 RCLKACT7 RCLKACT6 RCLKACT5 RCLKACT4 TCLKACT3 TCLKACT2 TCLKACT1 TCLKACT0 RCLKACT3 RCLKACT2 RCLKACT1 RCLKACT0		Transmit and Receive Line Clock Activity Indicator. 0,1,2, 3, 4, 5, 6, or 7 corresponds to a channel group number. Read Only. Reset to 0 after each read. Each indicator bit is cleared when the respective channel group is reset via PCI Reset, Soft Group Reset, or Soft Chip Reset. The TCLKACTx corresponds to TCLKx line clock. The RCLKACTx corresponds to RCLKx line clock. The indicator is set to 1 on the second rising edge of the corresponding serial interface line clock, and the previous value for the indicator bit was 0. If multiple channel groups are mapped to a single serial port, one clock is driving each channel group. The indicator bits reflect the activity of the clock driving the channel group. If MUSYCC does not detect a line clock, the value of the indicator bit(s) remain at the reset value 0. Reading from channel group RAM during the absence of a line clock results in the dword DEADACCEh (dead access) being returned. Writing to channel group RAM during the absence of a line clock results in the write being ignored.
15	RSVD	0	Reserved.
14:12	BLAPSE[2:0]	0–7	Expansion Bus Access Interval. MUSYCC waits BLAPSE+4 number of ECLK periods immediately after relinquishing the bus. This wait ensures that all the bus grant signals driven by the bus arbiter have sufficient time to be deasserted as a result of bus request signals being deasserted by MUSYCC.
11	ECKEN	0	Expansion Bus Clock Disabled. ECLK output is three-stated.
		1	Expansion Bus Clock Enabled. MUSYCC redrives and inverts PCLK input onto ECLK output pin.

Table 5-6. Global Configuration Descriptor *(2 of 2)*

Bit Field	Name	Value	Description
10	MPUSEL	0	Expansion Bus Microprocessor Selection, Motorola-style. Expansion bus supports the Motorola-style microprocessor interface and uses Motorola signals: Bus Request (BR*), Bus Grant (BG*), Address Strobe (AS*), Read/Write (R/WR*), and Read Strobe (RD*).
		1	Expansion Bus Microprocessor Selection, Intel-style. Expansion bus supports the Intel-style microprocessor interface and uses Intel signals: Hold Request (HOLD), Hold Acknowledge (HLDA), Address Latch Enable (ALE*), Write Strobe (WR*), and Data Strobe (DS*).
9:8	ALAPSE[1:0]	0–3	Expansion Bus Address Duration. MUSYCC extends the duration of valid address bits during an EBUS address phase to ALAPSE+1 number of ECLK periods. The control lines ALE* (Intel) or AS* (Motorola) indicate that the address bits have had the desired setup time.
7	RSVD	0	Reserved.
6:4	ELAPSE[2:0]	0–7	Expansion Bus Data Duration. MUSYCC extends the duration of valid data bits during an EBUS data phase to ELAPSE+1 number of ECLK periods. The control lines RD* and WR* (Intel) or DS* and R/WR* (Motorola) indicate that the data bits have had the desired setup time.
3	INTAMSK	0	INTA interrupt enabled.
		1	INTA interrupt disabled.
2	INTBMSK	0	INTB interrupt enabled.
		1	INTB interrupt disabled.
1:0	PORTMAP[1:0]	0	Default. Port 0 mapped to Channel Group 0. Port 1 mapped to Channel Group 1. Port 2 mapped to Channel Group 2. Port 3 mapped to Channel Group 3. Port 4 mapped to Channel Group 4. Port 5 mapped to Channel Group 5. Port 6 mapped to Channel Group 6. Port 7 mapped to Channel Group 7.
		1	Port 0 mapped to Channel Groups 0 and 1. Port 1 mapped to Channel Groups 2 and 3. Port 2 mapped to Channel Groups 4 and 5. Port 3 mapped to Channel Groups 6 and 7.
		2	Port 0 mapped to Channel Groups 0, 1, 2, and 3. Port 1 mapped to Channel Groups 4, 5, 6, and 7.
		3	Reserved.

5.2.1.2 Dual Address Cycle Base Pointer

MUSYCC supports 32-bit and 64-bit memory addressing. The Dual Address Cycle Base Pointer (DACBASE) supports 64-bit memory addressing and is described in Table 5-7.

If the value of DACBASE is 0, MUSYCC initiates all memory access cycles without dual-addressing. If the value is non-0, MUSYCC initiates all memory access cycles with dual-addressing.

For cycles without dual-addressing, MUSYCC uses the AD[31:0] signal lines to indicate the address of the memory access. During the address phase, MUSYCC encodes the type of access cycle (e.g., read, write,...) in the Command/Byte Enable signal lines, CBE[3:0]*. The address phase lasts one PCLK period.

For cycles with dual-addressing, MUSYCC multiplexes a 64-bit address onto the AD[31:0] signal lines and adds an additional PCLK period to the address phase. To indicate 64-bit addressing, MUSYCC encodes the dual address code onto the CBE[3:0]* signal lines during the first PCLK period of the address phase. MUSYCC encodes the access type code (e.g., read, write) onto the CBE[3:0]* signal lines during the second PCLK period of the address phase.

When MUSYCC accesses a 64-bit memory address using dual addressing, the upper 32 bits of the address are fixed to a non-0 value from DACBASE. To change from 64-bit addressing to 32-bit addressing, the value of DACBASE must be zeroed. Although MUSYCC is capable of initiating 64-bit addressing when in master mode, it responds only to 32-bit access cycles without dual-addressing.

Table 5-7. Dual Address Cycle Base Pointer

Bit Field	Name	Value	Description
31:0	DACBASE[31:0]	—	Dual Address Cycle Base Pointer. A 32-bit base register when non-0 causes all MUSYCC master operations (read/write) to use PCI Dual Address Cycle. The value in this register would be the upper 32-bits of the 64-bit addressing.

5.2.2 Channel Group Level Descriptors

Channel Group Descriptors contain all information needed to configure one channel group and the associated 32 logical channels, while maintaining pointers to buffer descriptors for each channel and direction. The contents of the Channel Group Descriptor are listed in Table 5-2, *Group Structure Memory Map*.

5.2.2.1 Group Base Pointer

The Group Base Pointer (GBASE) register per channel group within the host interface contains a 2 kB pointer aligned to a corresponding Channel Group Descriptor in shared memory, as described in Table 5-8.

Table 5-8. Group Base Pointer

Bit Field	Name	Value	Description
31:11	GBASEx[20:0]	—	These 21 bits are appended with 11 0s to form a 2 k block-aligned 32-bit address pointing to the first dword of the channel group structure for Channel Group x.
10:0	GBASEx[10:0]	0	These 11 bits appended to GBASE ensure 2 kB block alignment.

5.2.2.2 Service Request

The Service Request is a register per channel group within the host interface containing a bit field where instructions are written to MUSYCC by the host. The following instructions are supported:

- Perform device reset and initialization
- Perform channel group reset and initialization
- Configure a channel
- Read specific descriptors from within a Channel Group Descriptor
- Activate a channel
- Deactivate a channel
- Jump (re)activate a channel
- No-operation command

A service request is issued to a specific channel group within MUSYCC. The channel group then acknowledges by sending a service request acknowledge interrupt descriptor back to the host.

The soft-chip reset service request is the only service request not acknowledged by MUSYCC.

Issuing multiple service requests to the same channel group successively without first receiving acknowledgments from each request may cause the host to lose track of which service request has been acknowledged, because MUSYCC cannot uniquely acknowledge service requests for the same channel group. In addition, issuing multiple simultaneous requests to the same channel group causes indeterminate results within the channel group. To prevent these problems, the host software must wait for a Service Request Acknowledgement (SACK) after issuing any service request except for the soft chip reset request. The soft chip reset request does not issue an acknowledgement, but this request is guaranteed to be executed within two line clock periods.

Issuing a single service request to each supported channel group simultaneously is supported, because MUSYCC acknowledges each one uniquely with the Group ID bit field. Table 5-9 lists the bit fields and their descriptions of the service request descriptor.

Table 5-9. Service Request Descriptor *(1 of 2)*

Bit Field	Name	Value	Description
31:13	RSVD	0	Reserved.
12:8	SREQ[4:0]	0	No Operation. This service request performs no action other than to facilitate a Service Request Acknowledge Interrupt (SACK). This would be used as a "UNIX ping-like" operation to detect the presence of a channel group processor.
		1	Soft Chip Reset. This is identical to a hardware reset. Set PORTMAP = 0, disable all supported ports (both directions), and deactivate all 32 channels of all supported groups (both directions). The Interrupt Status Descriptor is reset to point to the first dword in the queue, and all indicator bits are reset. This service request is not acknowledged by MUSYCC.
		2	Soft Group Reset. This is similar to a hardware reset for a specified group and direction. Disable all specified ports (both directions), and deactivate all 32 channels of specified group (both directions).
		3	Reserved.
		4	Global Initialization. For the entire device, read the Global Configuration Descriptor and the Interrupt Queue Descriptor from shared memory. This initialization is performed following a hardware or soft-chip reset. The Interrupt Status Descriptor is reset to point to the first dword in the queue, and all indicator bits are reset.
		5	Group Initialization. For this group and direction, read the following from shared memory: Time Slot Map Subchannel Map Channel Configuration Descriptor Group Configuration Descriptor Memory Protection Descriptor Message Length Descriptor Port Configuration Descriptor This initialization must be performed by the host driver for each group and each direction immediately following any reset or global initialization.(1)

Table 5-9. Service Request Descriptor *(2 of 2)*

Bit Field	Name	Value	Description
12:8	SREQ[4:0]	6–7	Reserved.
		8	Channel Activation. For a specified channel and direction, initialize the channel, read the head pointer, jump to the first message descriptor, and start processing the data buffer. This request starts processing a new message list. The effect of this request is destructive to the current message being processed if the channel was already activated and processing a message. This request is useful in aborting the current message and starting a new message list.
		9	Channel Deactivation. For a specified channel and direction, suspend channel activity.
		10	Jump. For the receiver, this request is the same as the Channel Activation request. For the transmitter, this request is useful for switching to a new message list after the current message is completely transferred.
		11	Channel Configure. For a specified channel and direction, read the Channel Configuration Descriptor.
		12–15	Reserved.
		16	Read Global Configuration Descriptor.
		17	Read Interrupt Queue Descriptor. The Interrupt Status Descriptor is reset to point to the first dword in the queue, and all indicator bits are reset.
		18	Read Group Configuration Descriptor.
		19	Read Memory Protection Descriptor.
		20	Read Message Length Descriptor.
		21	Read Port Configuration Descriptor.
		22–23	Reserved.
		24	Read Time Slot Map. For this group and specified direction, read the Time Slot Map(1).
		25	Read Subchannel Map. For this group and specified direction, read the Subchannel Map(1).
		26	Read Channel Configuration Table. For this group and specified direction, read the Channel Configuration Table(1).
		27–31	Reserved.
7:6	RSVD	0	Reserved.
5	DIR	0	Receive direction.
		1	Transmit direction.
4:0	CH[4:0]		Channel number.

NOTE(S):

(1) Time Slot Map, Subchannel Map, and Channel Configuration Description Table are read into MUSYCC only if a serial clock is provided to the Serial Interface being configured.

5.2.2.3 Group Configuration Descriptor

The Group Configuration Descriptor contains configuration bits applying to all 32 logical channels within a given channel group as listed in Table 5-10.

***Table 5-10. Group Configuration Descriptor** (1 of 2)*

Bit Field	Name	Value	Description
31:22	RSVD	0	Reserved.
21:16	SUET[5:0]		Signal Unit Error Threshold. Sets maximum value of SUERM counter. When SUERM exceeds this count, a SUERR interrupt is generated.
15	SFALIGN	0	Super Frame Alignment. Flywheel Mechanism. Select roll over to 0 of time slot counter (the flywheel mechanism in the serial interface) as a frame synchronization event. For a transparent mode channel, wait for the flywheel to roll over to start message processing after channel activation. For descriptor polling, use the flywheel roll-over as a frame synchronization event. The polling frequency is determined by using the poll-throttle field elsewhere in this descriptor.
		1	Super Frame Alignment. External Signal. Select detection of frame synchronization signal (TSYNC or RSYNC) assertion as frame synchronization event. For a transparent mode channel, wait for assertion of signal to start message processing after channel activation. For descriptor polling, use assertion of signal as frame synchronization event. Polling frequency is determined by using poll-throttle field elsewhere in this descriptor.
14:12	RSVD	0	Reserved.
11:10	POLLTH[1:0]	0	Poll Throttle. Poll at every frame synchronization event. Not supported.
		1	Poll at every 16th frame synchronization event.
		2	Poll at every 32nd frame synchronization event.
		3	Poll at every 64th frame synchronization event.
9	INHTBSD	0	Inhibit Transmit Buffer Status Descriptor Disabled. At end of each transmitted data buffer, do not inhibit (allow) overwriting of Tx Buffer Descriptor with a Tx Buffer Status Descriptor.
		1	Inhibit Transmit Buffer Status Descriptor. As the Tx Buffer Status Descriptor is being inhibited, the host must rely on an interrupt for status information regarding transmitted data message.
8	INHRBSD	0	Inhibit Receive Buffer Status Descriptor Disabled. At the end of each Receive Data Buffer, do not inhibit (allow) overwriting of Rx Buffer Descriptor with a Rx Buffer Status Descriptor.
		1	Inhibit Receive Buffer Status Descriptor. As the Rx Buffer Status Descriptor is being inhibited, the host must rely on an interrupt for status information regarding the received data message.
7	MEMPVA	0	Memory Protection Violation Action. Reset Group. On a memory protection violation error, group reset is performed. As a result, all 32 channels are deactivated in both receive and transmit directions.
		1	Memory Protection Violation Action. Deactivate Channel. On a memory protection violation error, only the channel being serviced during violation is deactivated in both receive and transmit directions.

Table 5-10. Group Configuration Descriptor *(2 of 2)*

Bit Field	Name	Value	Description
6	MCENBL	0	Message Configuration Bits Copy Disable. Not supported.
		1	Message Configuration Bits Copy Enable. A set of configuration bits are copied from the last transmit buffer descriptor of a message (with EOM = 1) into internal configuration space and subsequently acted upon. The bits are used in specialized data communications applications requiring non-default idle code transmission on a per-message basis, inter-message pad fill, and/or message retransmission. Direct writes to MUSYCC that change any message configuration bits remain available. Only automatic copying from a transmit buffer descriptor (with the bit field EOM set to 1) is disabled. For a thorough discussion on message configuration bits see the Message Configuration Bits subsection in the Protocol Support section.
5	MSKCOFA	0	Change of Frame Alignment Interrupt Enabled. If COFA is detected, generate Interrupt Descriptor indicating COFA.
		1	Change of Frame Alignment Interrupt Disabled. If COFA is detected, do not generate Interrupt Descriptor.
4	MSKOOF	0	Out of Frame/Frame Recovery Interrupt Enabled—Receive Only. If OOF/FREC is detected, generate Interrupt Descriptor indicating OOF/FREC.
		1	Out of Frame/Frame Recovery Interrupt Disabled—Receive Only. If an ABT, FCS, LNG, or ALIGN error occurs, an OFF error is reported in the interrupt descriptor and buffer status descriptor instead of these errors.
3	OOFABT	0	OOF Message Processing Enabled—Receive Only. When OOF condition is detected, continue processing incoming data.
		1	OOF Message Processing Disabled—Receive Only. Incoming messages on all channels are aborted when the OOF condition is detected. HDLC channels recover automatically after the OOF condition has cleared. However, Transparent mode channels are automatically deactivated when the OOF condition is detected, and the host must reactivate all Transparent channels after the OOF condition has cleared.
2	SUBDSBL	0	Subchanneling Enabled. Overrides Subchannel Enable bit for all time slots and allows subchanneling.
		1	Subchanneling Disabled. Overrides Subchannel Enable bit for all time slots and disallows subchanneling. Using this field to disable subchanneling frees Subchannel Descriptor Map memory space for use as an extended data buffer space.
1	TXENBL	0	Transmitter Disabled. Logically resets time slot enable bits for all time slots. Transmit data lines are three-stated. This logical, channel-group-wide state does not affect the bit values in any time slot map.
		1	Transmitter Enabled. Logically allows all channels with time slot enable bits set to start processing data. This logical, channel-group-wide state does not affect the bit values in any time slot map.
0	RXENBL	0	Receiver Disabled. Logically resets time slot enable bits for all time slots. This logical, channel-group-wide state does not affect the bit values in any time slot map.
		1	Receiver Enabled. Logically allows all channels with time slot enable bits set to start processing data. This logical, channel-group-wide state does not affect the bit values in any time slot map.

5.2.2.4 Memory Protection Descriptor

The memory protection function is not implemented. This function must be disabled by clearing the PROTENBL bit in the Memory Protection Descriptor. Table 5-11 lists the bit field and description of the Memory Protection Descriptor.

Table 5-11. Memory Protection Descriptor

Bit Field	Name	Value	Description
31	PROTENBL	0	Memory Protection Disabled.
		1	Memory Protection Enabled. Not supported.
30:28	RSVD	0	Reserved.
27:16	PROTHI[11:0]		Memory Protection High Address. Upper 12 bits (inclusive) of address for highest memory location under protection.
15:12	RSVD	0	Reserved.
11:0	PROTLO[11:0]		Memory Protection Low Address. Upper 12 bits (inclusive) of the address for lowest memory location under protection.

5.2.2.5 Port Configuration Descriptor

The Port Configuration Descriptor defines how MUSYCC interprets and synchronizes the transmit and receive bit streams associated with a port. There is one descriptor per port; therefore, the descriptor is used for both transmit and receive directions for a single port.

NOTE: When a port is being used in conjunction with another port —as is the case when PORTMAP=1 or PORTMAP=2 (see Table 5-6, *Global Configuration Descriptor*)—it is imperative that the unused port descriptor be mapped identically to the used port descriptor. That is, if PORTMAP=1, then the group 1/port 1 descriptor must be bit-for-bit identical with the group 0/port 0 descriptor; and, the group 3/port 3 descriptor must be bit-for-bit identical with the group 2/port 2 descriptor. In the case of PORTMAP=2, then the group 1, 2, and 3 descriptors must be bit-for-bit identical with the group 0/port 0 descriptor.

Table 5-12 details the Port Configuration Descriptor.

Table 5-12. Port Configuration Descriptor

Bit Field	Name	Value	Description
31:10	RSVD	0	Reserved.
9	TRITX	0	Transmit Three-state Enabled. When a channel group is enabled, but a time slot within the group is not mapped via the Time Slot Map, the transmitter three-states the output data signal.
		1	Transmit Three-State Disabled. When a channel group is enabled, but a time slot within the group is not mapped via the Time Slot Map, the transmitter outputs a logic 1 on the output data signal.
8	ROOF_EDGE	0	Receiver Out of Frame—Falling Edge. ROOF input sampled in on falling edge of RCLK.
		1	Receiver Out of Frame—Rising Edge.
7	RSYNC_EDGE	0	Receiver Frame Synchronization—Falling Edge. RSYNC input sampled in on falling edge of RCLK.
		1	Receiver Frame Synchronization—Rising Edge.
6	RDAT_EDGE	0	Receiver Data—Falling Edge. RDAT input sampled in on falling edge of RCLK.
		1	Receiver Data—Rising Edge.
5	TSYNC_EDGE	0	Transmitter Frame Synchronization—Falling Edge. TSYNC input sampled in on falling edge of TCLK.
		1	Transmitter Frame Synchronization—Rising Edge.
4	TDAT_EDGE	0	Transmitter Data—Falling Edge. TDAT output latched out on falling edge of TCLK.
		1	Transmitter Data—Rising Edge.
3	RSVD	0	Reserved.
2:0	PORTMD[2:0]	0	T1 Mode—24 time slots and T1 signaling.
		1	E1 Mode—32 time slots and E1 signaling.
		2	2xE1 Mode—64 time slots and E1 signaling.
		3	4xE1 Mode—128 time slots and E1 signaling.
		4	Nx64 Mode. Frame synchronization flywheel disabled. COFA detection disabled. Every synchronization signal assertion resets time slot counter to zero.
		5–7	Reserved.

5.2.2.6 Message Length Descriptor

Each channel group can have two separate values for maximum message length: MAXFRM1 or MAXFRM2 (see Table 5-13). The maximum message length is 4,094 octets. The minimum message length is either 1, 3, or 5 depending on non-FCS mode or FCS16 or FCS32 support, respectively. Each receive channel either selects one of these message length values or disables message length checking altogether.

The MAXSEL bit field (see Table 5-18, *Channel Configuration Descriptor*) selects which, if any, register is used for received-message length checking. If MUSYCC receives a message exceeding the allowed maximum, the current message processing is discontinued and terminates further transfer of data to shared memory. In addition, a Receive Buffer Status Descriptor, corresponding to the partially received message, indicates a Long Message error condition, and an interrupt descriptor is generated towards the host indicating the same error condition.

NOTE: The equation that defines maximum message length without generating a LNG error is:
Maximum Message Length = MAXFRM – FCS where FCS = 2 bytes for HDLC-16 protocol and FCS = 4 bytes for HDLC-32 protocol.

In the case of a short message (bit count less than 3 or 5 octets), data is not transferred into shared memory and is discarded. In addition, an interrupt descriptor is generated towards the host, indicating the same error condition.

Table 5-13. Message Length Descriptor

Bit Field	Name	Value	Description
31:28	RSVD	0	Reserved.
27:16	MAXFRM2[11:0]		Defines a limit for the maximum number of octets allowed in a received HDLC message. Valid values for the register range from 3 to 4094 depending on FCS16 or FCS32.
15:12	RSVD	0	Reserved.
11:0	MAXFRM1[11:0]		Defines a limit for the maximum number of octets allowed in a received HDLC message. Valid values for the register range from 3 to 4094 depending on FCS16 or FCS32.

5.2.2.7 Time Slot Map

The time slot map consists of 32 time slot descriptors. One descriptor maps four time slots. The entire map contains configuration information for 128 time slots. The serial port does not need to support 128 time slots all the time. Any number of time slots from 1 to 128 is supportable.

The time slot map is used when the serial port associated with a channel group is configured in one of the channelized modes: T1, E1, 2xE1, 4xE1, or Nx64. MUSYCC supports mapping of up to 128 time slots from a channelized bit stream with up to 32 logical channels in each channel group.

Numerous mappings of time slots to channels are possible. Multiple time slots can be mapped to a single channel; however, each time slot can map to only one channel at a time. When the number of active time slots exceeds the number of channels in a group (greater than 32 time slots), and each time slot requires a separate channel, MUSYCC can be configured to internally connect 2 or 4 channel groups together to provide up to 64- or 128-channel support, respectively.

Two time slot maps are required for each channel group, one for transmit functions and one for receive functions. The two maps are configured independently. Each map consists of 128 successive 8-bit fields, each corresponding to one time slot. The bit field includes the following information:

- Time Slot Enabled/Time Slot (TSEN) Mode of Operation (64 kbps, 56 kbps, subchannel) indicator
- Logical channel number (0–31) associated with time slot (CH)

For disabled time slots, modes and logical channels can be assigned, but the information does not apply to the operation of the channel.

For enabled time slots, the valid modes of operation include the following:

- 64 kbps Mode: all 8 bits of time slot are assigned to one channel.
- 56 kbps Mode: first 7 bits of time slot are assigned to one channel. Last bit, Most Significant Bit (MSB), is unassigned and considered disabled.
- Subchannel Mode, Bit 0 Disabled: first bit, Least Significant Bit (LSB), of time slot is unassigned. Each of the next 7 bits in time slot can be individually enabled and independently mapped to any channel in a channel group.
- Subchannel Mode, Bit 0 Enabled: first bit (LSB) of time slot is always enabled and assigned to a channel in a channel group. Each of the next 7 bits in time slot can be individually enabled and independently mapped to any channel in a channel group.

The logical channel number represents the channel in the channel group handling the bit stream from the time slot or slots assigned to it. The value of the channel number ranges from 0–31.

Table 5-14 lists the time slot descriptor location and byte offset. Table 5-15 lists the value and description of each time slot descriptor.

Table 5-14. Transmit or Receive Time Slot Map

Byte Offset	Time Slot Descriptor Relative Location			
	MSB			LSB
00h	TS03	TS02	TS01	TS00
...	...	...	...	...
1Ch	TS31	TS30	TS29	TS28
...	...	...	...	...
3Ch	TS63	TS62	TS61	TS60
...	...	...	...	...
5Ch	TS95	TS94	TS93	TS92
...	...	...	...	...
7Ch	TS127	TS126	TS125	TS124

Accessing the Time Slot Map within MUSYCC requires that a serial line clock be present at the serial interface. If a clock is not present, writes are ignored, and reads return all 1s.

The host can read and write the Receive Time Slot Map information from within MUSYCC; however, the host can only write Transmit Time Slot Map into MUSYCC. The transmit maps are stored in write-only registers. Reading transmit maps results in all 1s being returned.

Table 5-15. Time Slot Descriptor

Bit Field	Name	Value	Description
31:29	TSEN3[2:0]	0	Time Slot Disabled. Default.
		1–3	Reserved.
		4	Time Slot Enabled. 64 kbps mode.
		5	Time Slot Enabled. 56 kbps mode.
		6	Time Slot Enabled. Subchannel mode w/o first bit.
		7	Time Slot Enabled. Subchannel mode w/ first bit.
28:24	CH3[4:0]	0–31	Channel number assigned to this time slot.
23:21	TSEN2[2:0]	0	Time Slot Disabled. Default.
		1–3	Reserved.
		4	Time Slot Enabled. 64 kbps mode.
		5	Time Slot Enabled. 56 kbps mode.
		6	Time Slot Enabled. Subchannel mode w/o first bit.
		7	Time Slot Enabled. Subchannel mode w/ first bit.
20:16	CH2[4:0]	0–31	Channel number assigned to this time slot.
15:13	TSEN1[2:0]	0	Time Slot Disabled. Default.
		1–3	Reserved.
		4	Time Slot Enabled. 64 kbps mode.
		5	Time Slot Enabled. 56 kbps mode.
		6	Time Slot Enabled. Subchannel mode w/o first bit.
		7	Time Slot Enabled. Subchannel mode w/ first bit.
12:8	CH1[4:0]	0–31	Channel number assigned to this time slot.
7:5	TSEN0[2:0]	0	Time Slot Disabled. Default.
		1–3	Reserved.
		4	Time Slot Enabled. 64 kbps mode.
		5	Time Slot Enabled. 56 kbps mode.
		6	Time Slot Enabled. Subchannel mode w/o first bit.
		7	Time Slot Enabled. Subchannel mode w/ first bit.
4:0	CH0[4:0]	0–31	Channel number assigned to this time slot.

5.2.2.8 Subchannel Map

To provide the subchanneling feature, MUSYCC shares the time slot mapping function between a Time Slot Map and a Subchannel Map. The transmit and receive functions each have a separate pair of maps.

The Time Slot Map indicates if the time slot is enabled and configured for subchanneling. The TSEN bit field in the Time Slot Descriptor specifies the mode of operation for the time slot. If TSEN = 6, the time slot is in subchannel mode with bit 0 disabled. If TSEN = 7, the time slot is in subchannel mode with bit 0 enabled and mapped to the channel specified in the CH bit field in the same Time Slot Descriptor. The CHx bit field is also used in the treatment of the remaining 7 bits in the time slot, specifically as part of an index into Subchannel Map. The Time Slot Map always manages (disables or enables or maps) bit 0 of the time slot. The Subchannel Map configures each of the remaining 7 bits of a time slot.

The CHx bit field (0–31) from the Time Slot Descriptor is linked with the bit number (1–7) of the time slot being processed to form an 8-bit index (0–255) into the Subchannel Map for each bit number. The Subchannel Map associates each bit of a time slot to a channel.

For a time slot configured for subchannel mode, MUSYCC considers each bit of that time slot independently of any other bit in the same time slot. Any bit can be enabled or disabled. The time slot is considered to consist of 8 individual 8 kbps bit streams, or subchannels. The subchannels can remain separated or be concatenated to provide one or more subchannels running at bit rates between 8 kbps and 64 kbps, inclusive, in multiple of 8 kbps.

Note the following special cases for subchannel map assignments:

- Bit 0 is always disabled or enabled/mapped by the Time Slot Map; therefore, configuring the Subchannel Map for bit 0 is not required.
- Assigning the same channel number for all bits (including bit 0) is equivalent to disabling the subchanneling feature and treating the time slot as if in 64 kbps mode.
- Disabling all bits in subchannel mode is equivalent to disabling the time slot using the Time Slot Map.

The Subchannel Map can map bits 1–7 from each time slot to any of 32 channels in a channel group. The map for bit 0 is allocated, but unused (bit 0 is mapped by the time slot map). The Subchannel Map consists of 256 descriptors representing 32 channels and 8 bits per channel. Two subchannel descriptors (2 dwords) are required to describe the treatment of 8 bits of one channel. The first dword describes the treatment of the lower 4 bits, and the second dword describes the treatment of the upper 4 bits. Table 5-16 lists the subchannel descriptor location and byte offset. Table 5-17 lists the value and description of each subchannel descriptor.

Table 5-16. Transmit or Receive Subchannel Map

Byte Offset	MSB			LSB
00h	Ch0, Bit 3	Ch0, Bit 2	Ch0, Bit 1	Unused
04h	Ch0, Bit 7	Ch0, Bit 6	Ch0, Bit 5	Ch0, Bit 4
08h	Ch1, Bit 3	Ch1, Bit 2	Ch1, Bit 1	Unused
0Ch	Ch1, Bit 7	Ch1, Bit 6	Ch1, Bit 5	Ch1, Bit 4
...	....	....	....	...
...	....	....	....	...
F8h	Ch31, Bit 3	Ch31, Bit 2	Ch31, Bit 1	Unused
FCh	Ch31, Bit 7	Ch31, Bit 6	Ch31, Bit 5	Ch31, Bit 4

Table 5-17. Subchannel Descriptor

Bit Field	Name	Value	Description
31	BITEN3/7	0	Bit disabled.
		1	Bit enabled.
30:29	RSVD	0	Reserved.
28:24	CH3[4:0]	0–31	Channel number assigned to this bit.
23	BITEN2/6	0	Bit disabled.
		1	Bit enabled.
22:21	RSVD	0	Reserved.
20:16	CH2[4:0]	0–31	Channel number assigned to this bit.
15	BITEN1/5	0	Bit disabled.
		1	Bit enabled.
14:13	RSVD	0	Reserved.
12:8	CH1[4:0]	0–31	Channel number assigned to this bit.
7	BITEN0/3	0	Bit disabled.
		1	Bit enabled.
6:5	RSVD	0	Reserved.
4:0	CH0[4:0]	0–31	Channel number assigned to this bit.

To enable the subchanneling feature, both the Time Slot Map and the Subchannel Map must be copied into MUSYCC's internal registers because it is from here time slot-to-channel mapping and channel-to-subchannel mapping is decoded. The host can instruct MUSYCC to read in the maps from shared memory by issuing the appropriate service request; otherwise, the host must perform multiple direct writes into MUSYCC's internal registers by appropriately addressing PCI access cycles for MUSYCC.

Accessing the Time Slot Map or Subchannel Map within MUSYCC requires that a serial line clock be present at the serial interface. If a clock is not present, writes are ignored, and reads return all 1s.

The host can read and write the Receive Time Slot Map from within MUSYCC; however the host can only write the Transmit Time Slot Map into MUSYCC. The transmit maps are stored in write-only registers. Reading transmit maps results in all 1s being returned.

The host can read and write the Receive Subchannel Map from within MUSYCC; however, the host can only write the Transmit Subchannel Map into MUSYCC. The transmit maps are stored in write-only registers. Reading the transmit map results in all 1s being returned.

5.2.3 Channel Level Descriptors

The channel level descriptors contain information necessary to configure channel registers.

5.2.3.1 Channel Configuration Descriptor

The Channel Configuration Descriptor configures aspects of the channel common to all messages passing through the channel. One descriptor exists for each logical channel direction.

Table 5-18 lists the values and description of each channel configuration descriptor.

Table 5-18. Channel Configuration Descriptor *(1 of 3)*

Bit Field	Name	Value	Description
31	PADJ	0	Pad Count Adjustment disabled. No adjustment is made to the value of PADCNT if Zero Insertions is detected.
		1	Pad Count Adjustment enabled. The value of PADCNT is reduced if Zero Insertions is detected. This adjustment is required for rate adaptive applications such as *ITU-T Recommendation V.120*.
30	RSVD	0	Reserved
29:24	BUFFLOC[5:0]	00h–3Fh	Channel Buffer Location Index. Starting location of internal FIFO data buffer for this channel and direction.
23	INV	0	Data Inversion disabled. All data bits in message are not inverted between shared memory and MUSYCC.
22	RSVD	0	Reserved.
21:16	BUFFLEN[5:0]	00h–3Fh	Internal Data Buffer Length. Number of internal FIFO data buffer locations allocated to this channel and direction equals 2 x (BUFFLEN+1) dwords.
15	EOPI	0	End Of Padfill Interrupt disabled. Transmit Only. After outputting last padfill code, do not generate interrupt indicating condition.
		1	End of Padfill Interrupt enabled.

Table 5-18. Channel Configuration Descriptor *(2 of 3)*

Bit Field	Name	Value	Description
14:12	PROTOCOL[2:0]	0	TRANSPARENT
		1	SS7-HDLC-FCS16
		2	HDLC-FCS16
		3	HDLC-FCS32
		4–7	Reserved.
11:10	MAXSEL[1:0]	0	Message Length—Disable message length check.
		1	Message Length—Use Register 1. Use MAXFRM1 bit field in the message length descriptor for maximum receive message length limit.
		2	Message Length—Use Register 2. Use MAXFRM2 bit field in the message length descriptor for maximum receive message length limit.
		3	Reserved
9	FCS	0	FCS Transfer Normal. For receive, do not transfer received FCS into shared memory along with data message. For transmit, do transmit calculated FCS out serial port after last bit in last data buffer has been transmitted.
		1	FCS=1. Non FSC Mode. For receive, transfer received FCS 31 into shared memory along with data message; do not transmit calculated FCS out of serial port. In Non-FCS Mode, a SHT message detection is disabled. Any number of bytes can be transmitted and received within any single message, including message length of only one byte.
8	MSKSUERR	0	SUERR Interrupt enabled. Receive only. For SS7-HDLC-FCS16 mode, Signal Unit Error Rate Monitor function generates interrupt when signal unit error count crosses signal unit error threshold.
		1	SUERR Interrupt disabled.
7	MSKSINC	0	SINC Interrupt enabled. Receive only. For SS7-HDLC-FCS16 mode, SUERM function generates interrupt when signal unit error count increments.
		1	SINC Interrupt disabled.
6	MSKSDEC	0	SDEC Interrupt enabled. Receive only. For SS7-HDLC-FCS16 mode, SUERM function generates interrupt when signal unit error count decrements.
		1	SDEC Interrupt disabled.
5	MSKSFILT	0	SFILT Interrupt enabled. Receive only. For SS7-HDLC-FCS16 mode, interrupt generated when two consecutive received messages are found to be identical. Second message discarded.
		1	SFILT Interrupt disabled. For SS7-HDLC-FCS16 mode, interrupt is not generated when two consecutive received messages are found to be identical. Second message discarded.
4	MSKIDLE	0	CHABT, CHIC, SHT Interrupt enabled. Receive only. When receiver detects change to abort code, change to idle code, or too-short message, this bit generates interrupt to indicate condition.
		1	CHABT, CHIC, SHT Interrupt disabled.

Table 5-18. Channel Configuration Descriptor *(2 of 3)*

Bit Field	Name	Value	Description
14:12	PROTOCOL[2:0]	0	TRANSPARENT
		1	SS7-HDLC-FCS16
		2	HDLC-FCS16
		3	HDLC-FCS32
		4–7	Reserved.
11:10	MAXSEL[1:0]	0	Message Length—Disable message length check.
		1	Message Length—Use Register 1. Use MAXFRM1 bit field in the message length descriptor for maximum receive message length limit.
		2	Message Length—Use Register 2. Use MAXFRM2 bit field in the message length descriptor for maximum receive message length limit.
		3	Reserved
9	FCS	0	FCS Transfer Normal. For receive, do not transfer received FCS into shared memory along with data message. For transmit, do transmit calculated FCS out serial port after last bit in last data buffer has been transmitted.
		1	FCS=1. Non FSC Mode. For receive, transfer received FCS 31 into shared memory along with data message; do not transmit calculated FCS out of serial port. In Non-FCS Mode, a SHT message detection is disabled. Any number of bytes can be transmitted and received within any single message, including message length of only one byte.
8	MSKSUERR	0	SUERR Interrupt enabled. Receive only. For SS7-HDLC-FCS16 mode, Signal Unit Error Rate Monitor function generates interrupt when signal unit error count crosses signal unit error threshold.
		1	SUERR Interrupt disabled.
7	MSKSINC	0	SINC Interrupt enabled. Receive only. For SS7-HDLC-FCS16 mode, SUERM function generates interrupt when signal unit error count increments.
		1	SINC Interrupt disabled.
6	MSKSDEC	0	SDEC Interrupt enabled. Receive only. For SS7-HDLC-FCS16 mode, SUERM function generates interrupt when signal unit error count decrements.
		1	SDEC Interrupt disabled.
5	MSKSFILT	0	SFILT Interrupt enabled. Receive only. For SS7-HDLC-FCS16 mode, interrupt generated when two consecutive received messages are found to be identical. Second message discarded.
		1	SFILT Interrupt disabled. For SS7-HDLC-FCS16 mode, interrupt is not generated when two consecutive received messages are found to be identical. Second message discarded.
4	MSKIDLE	0	CHABT, CHIC, SHT Interrupt enabled. Receive only. When receiver detects change to abort code, change to idle code, or too-short message, this bit generates interrupt to indicate condition.
		1	CHABT, CHIC, SHT Interrupt disabled.

Table 5-18. Channel Configuration Descriptor *(3 of 3)*

Bit Field	Name	Value	Description
3	MSKMSG	0	LNG, FCS, ALIGN, ABT Interrupt enabled. Receive only. When receiver detects too-long message, FCS error, message alignment error, or abort condition, this bit generates interrupt to indicate condition.
		1	In order for MSKMSG=1 to disable all interrupts (LNG, FCS, ALIGN) the MSKEOM must be set (i.e., 1).
2	MSKEOM	0	EOM Interrupt enabled. Receive and Transmit. Interrupt generated when end of message detected.
		1	EOM Interrupt disabled.
1	MSKBUFF	0	BUFF—Interrupts enabled. Receive and Transmit. Interrupt generated when transmitter underflow buffer or receiver overflows buffer internally to MUSYCC. ONR—Interrupts enabled. Receive and Transmit. Interrupt generated where message pointer/descriptor is not available to MUSYCC where is expected. (Refer to Figure 5-31, *Interrupt Descriptor*.)
		1	BUFF—Interrupts disabled. ONR—Interrupts disabled.
0	RSVD	0	Reserved.

5.2.4 Message Level Descriptor

One message descriptor defines one data buffer where all or part of a message is stored in shared memory. By linking message descriptors, numerous data buffers are linked to support high-speed data links or large messages spread across a number of smaller data buffers.

Depending on the transmission and reception rate of individual channels, the numbers and sizes of message buffers can vary between channels and applications. For high-speed lines, more and larger buffers can be used to provide ample data storage while the host processes each message in the list of messages. For low-speed lines, fewer and smaller buffers can be used as the host may be able to process each message faster and the need to store messages is lessened.

Multiple smaller data buffers can be linked using message descriptors to store one large message. In utilizing multiple buffers, the importance of keeping the sequence of data buffers in order is obvious.

MUSYCC's operation allows for the following:

- Multiple buffer lists
- Multiple and variable size buffers within a buffer list
- Multiple buffers storing a single message
- Sequencing of individual data buffers for a multi-buffer message

A message descriptor is designed to be usable by both the transmit and receive functions in MUSYCC. In providing this symmetry, a mechanism known as self-servicing buffers is available, which allows the reuse of a single descriptor for both the transmit and receive portions of a channel, and is designed for diagnostics and loopback capabilities. Table 5-19 lists the message descriptor details.

Table 5-19. Message Descriptor

Byte Offset	Field Name	dwords	Bytes
00h	Buffer Descriptor (Host Writes) Buffer Status Descriptor (MUSYCC Writes)	1	4
04h	Data Pointer	1	4
08h	Next Pointer	1	4
	TOTAL	3	12

5.2.4.1 Using Message Descriptors

MUSYCC checks data from a message descriptor before processing the associated data buffer. When a data buffer is completely processed (either transmitted or received), MUSYCC overwrites the buffer descriptor field (the first dword in a message descriptor) with a buffer status descriptor.

The buffer status descriptor specifies the number of bytes transferred, an end of message indicator, and a buffer owner-bit indicator that assigns control of associated buffers back to the host.

The owner bit transfers control of a data buffer between MUSYCC and the host. The message descriptor can be assigned before an associated data buffer is allocated in memory. In this case, MUSYCC polls the contents of the buffer descriptor until the host grants ownership of a data buffer to MUSYCC. After MUSYCC processes the data buffer, it grants the ownership back to the host.

The owner bit prevents MUSYCC from processing the same buffer twice without intervention from the host. If MUSYCC detects an opening flag of a received message, but does not have ownership of the current data buffer (via the current message descriptor), an interrupt is sent to the host indicating that MUSYCC needed a data buffer and did not have access to one.

The host can append additional information beyond the end of a data buffer as long as the longest message length can be fitted first into the data buffer. In the case of additional information, MUSYCC would not know about the information, nor would it ever read from or write to that space.

For simplicity, the message level descriptions that follow are made in reference to one channel. Each channel is serviced independently of other channels, and separate descriptor lists are maintained for each supported channel.

Similarly, the transmit and receive sections of a channel service that the descriptor lists identically and separate descriptor lists are maintained for each section. Also, the size of data fields in the descriptors are identical; however, the layout of fields between receive and transmit descriptors are different.

5.2.4.2 Note for Interrupt Driven Drivers

An interrupt from MUSYCC does not imply that MUSYCC read a buffer status descriptor and made it host-owned.

As mentioned in Note (2) in Table 5-31, the occurrence of a MUSYCC interrupt and a buffer status descriptor update are not time correlated. The delay of the buffer status descriptor update is a maximum of 2 HDLC frames after the interrupt.

The driver must read the owner field to confirm its ownership before writing a new buffer status descriptor. If the driver receives an interrupt and does not detect a host-owned buffer, it should wait a minimum of 2 HDLC frames before signaling an abnormal condition.

The requirement to check the buffer status descriptor is applicable only when buffer status descriptor updates are enabled (INHRBSD = 0).

5.2.4.3 Head Pointer

The head pointer points to the first message descriptor in a list of descriptors assigned to a channel's transmitter or receiver.

A head pointer allows the host to specify a new list of descriptors to use for channel processing. This mechanism can be used after a reset to kick-start or reactivate channel processing which has completely processed the current list of descriptors.

A head pointer also allows the host to generate a new list of descriptors in memory before performing a list transition; that is, while MUSYCC processes data in one list, the host can process data in a separate list, and, when appropriate, can switch the lists.

Table 5-20 lists the head pointers and their descriptions.

Table 5-20. Head Pointer

Bit Field	Name	Value	Description
31:2	HEADPTR[29:0]		These 30 bits are appended with 00b to form a dword-aligned 32-bit address. This address points to the first Message Descriptor in a list of descriptors.
1:0	HEADPTR[1:0]	0	Ensures dword alignment.

5.2.4.4 Message Pointer

The message pointer points to the current message descriptor being serviced. This pointer is maintained in a fixed memory location relative to a Group Base Pointer in shared memory.

Table 5-21 lists the message pointers and their descriptions.

Table 5-21. Message Pointer

Bit Field	Name	Value	Description
31:2	MSGPTR[29:0]		These 30 bits are appended with 00b to form a dword-aligned 32-bit address. This word pointer points to the first dword of a Message Descriptor.
1:0	MSGPTR[1:0]	0	Ensures dword alignment.

5.2.4.5 Message Descriptor

The Message Descriptor is pointed to by the Message Pointer and the Head Pointer and is maintained in a variable location in shared memory. A Message Descriptor includes the following fields:

- Buffer Descriptor (when host writes)
- Buffer Status Descriptor (when MUSYCC writes)
- Data Buffer Pointer
- Next Descriptor Pointer

5.2.4.6 Buffer Descriptor

The Buffer Descriptor is the first dword of a Message Descriptor after the host has prepared the data structures in memory. All Buffer Descriptors include the following fields:

- Owner Indicator Bit (OWNER)
- No Poll/Poll Indicator (NP)
- End of Buffer Interrupt Enable (EOBI)
- Buffer Length (BLEN)

The OWNER bit is a generic term for any descriptor. In a Transmit Buffer Descriptor it is called MUSYCC; in a Receive Buffer Descriptor it is called host. The names are different to indicate that the active sense of the owner bit is different between transmit and receive functions.

In addition to the above list of fields, Transmit Buffer Descriptors also include the following fields:

- End of Message Indicator (EOM)
- Idle Code Selection (IC)
- Pad Enable (PADEN)
- Pad Count (PADCNT)
- Repeat Packet Enable (REPEAT)

IC, PADEN, PADCNT, and REPEAT are valid only when EOM = 1.

Tables 5-22 and 5-23 list the transmit and receive buffer descriptors and definitions.

Table 5-22. Transmit Buffer Descriptor

Bit Field	Name	Value	Description
31	OWNER	0	HOST Owns Buffer. HDLC channel remains in idle mode while polling this bit periodically (if NP = 0) until host relinquishes control to MUSYCC by setting OWNER = 1. In transparent mode the channel is deactivated.
		1	MUSYCC Owns Buffer. Continue processing data buffer normally.
30	NP	0	Poll Enabled. If OWNER = 0, host-owned, MUSYCC polls the message descriptor periodically while in idle mode until OWNER = 1.
		1	Poll Disabled. If OWNER = 0, then waits for a Channel Activate or Jump Service Request from host.
29	EOM	0	Data Buffer w/o End of Message.
		1	Data Buffer w/ End of Message.
28	EOBI	0	End of Buffer Interrupt Disabled. When no more data can be taken from or put into a data buffer, an EOB interrupt is not generated.
		1	End of Buffer Interrupt Enabled.
27		0	Reserved.
26:25	IC[1:0]	0	Idle Code Select – 7Eh
		1	Idle Code Select – FFh
		2	Idle Code Select – 00h
		3	Reserved.
24	PADEN	0	Pad Fill Disabled. One shared opening/closing flag (7Eh) is inserted before sending next message.
		1	Pad Fill Enabled. Also, see PADCNT bit field.
23:16	PADCNT[7:0]		Pad Count. When PADEN = 1, PADCNT indicates the minimum number of idle codes to be inserted between the closing flags and the next opening flag (7Eh). If PADCNT = 2 and IC = 1, for example, MUSYCC outputs the bit pattern 7Eh..FFh..FFh..7Eh. There is no indication by MUSYCC if more than PADCNT number of idle codes are inserted.
15	REPEAT	0	Repeat message transmission disabled.
		1	Repeat message transmission enabled.
14	RSVD	0	Reserved.
13:0	BLEN[13:0]		Buffer Length. The number of octets in data buffer to be transmitted. In general, this would equal the allocated buffer size.

Table 5-23. Receive Buffer Descriptor

Bit Field	Name	Value	Description
31	OWNER	0	MUSYCC Owns Buffer. Continue processing data buffer normally.
		1	HOST Owns Buffer. Channel is to remain in idle mode while polling this bit periodically (if NP = 0) until host relinquishes control to MUSYCC by setting OWNER = 0. In transparent mode the channel is deactivated.
30	NP	0	Poll Enabled. If OWNER = 1, host-owned, MUSYCC polls message descriptor periodically until OWNER = 0.
		1	Poll Disabled. If OWNER = 1 then wait for a Channel Activate Service Request from host.
29	RSVD	0	Reserved.
28	EOBI	0	End of Buffer Interrupt Disabled. When more data cannot be taken from or put into a data buffer, an EOB interrupt is not generated.
		1	End of Buffer Interrupt enabled.
27:14	RSVD	0	Reserved.
13:0	BLEN[13:0]		Buffer Length. Actual number of received data octets might be less than this. This number indicates how many will fit into data buffer. The buffer length should not exceed 8 k.

5.2.4.7 Buffer Status Descriptor

The Buffer Status Descriptor contains status information regarding data buffer servicing. If configured to do so, MUSYCC writes a Buffer Status Descriptor over a Buffer Descriptor. This descriptor includes the following fields:

- Owner Indicator Bit (OWNER)
- End of Message Indicator (EOM)
- Data Length (DLEN)

In addition to the fields, a Receive Buffer Status Descriptor also includes the Error Status (ERROR) bit field.

Buffer Status Descriptors are designed to work with Buffer Descriptors. This allows a self-servicing buffer mechanism where a transmit channel will empty a list of data buffers immediately after a receive channel fills those buffers, all without host intervention. The value for the OWNER bit field in the transmit buffer is opposite of the value of the OWNER bit field in the receive buffer descriptor.

Tables 5-24 and 5-25 list the transmit and receive buffer status descriptors and their descriptions.

Table 5-24. Transmit Buffer Status Descriptor

Bit Field	Name	Value	Description
31	OWNER	0	Host Owns Buffer. MUSYCC has relinquished control of buffer back to host. MUSYCC is done processing buffer.
		1	MUSYCC Owns Buffer. Until MUSYCC relinquishes control, the data in this descriptor is being used by MUSYCC.
30	RSVD	0	Reserved.
29	EOM	0	End of Message Indicator. Copied from Transmit Buffer Descriptor.
		1	End of Message Indicator. Copied from Transmit Buffer Descriptor.
28:14	RSVD	0	Reserved.
13:0	BLEN[13:0]		Buffer Length. The number of octets from data buffer transmitted. In general this would equal the allocated buffer size.

Table 5-25. Receive Buffer Status Descriptor

Bit Field	Name	Value	Description
31	OWNER	0	MUSYCC Owns Buffer. Until MUSYCC relinquishes control, the data in this descriptor is being used by MUSYCC.
		1	Host Owns Buffer. MUSYCC has relinquished control of buffer back to host. MUSYCC is done processing buffer.
30	RSVD	0	Reserved.
29	EOM	0	End of Message Indicator. The last octet for this message is not in this buffer.
		1	End of Message indicator. The last octet for this data message is in this buffer either because a valid closing flag (7Eh) was detected or the receiver terminated due to an error condition.
28:20	RSVD	0	Reserved.
19:16	ERROR[3:0]	0	OK: No error detected in this receive buffer.
		1	BUFF: Buffer Error. Data is lost. For receive: Internal data buffer overflow.
		2	COFA: Change Of Frame Alignment. RSYNC signal is misaligned with the flywheel in the serial interface.
		3–7	Reserved.
		8	OOF: Out of Frame. ROOF signal is asserted.
		9	FCS: Frame Check Sequence Error. Received HDLC frame is terminated with proper 7Eh flag, but computed FCS does not match received FCS.
		10	ALIGN: Octet Alignment Error. Message payload size, after zero extraction, is not a multiple of 8 bits. This error takes precedence over an FCS error.
		11	ABT: Abort Flag Termination. Received message is terminated with abort sequence, seven sequential 1s, instead of a closing flag (7Eh).
		12	LNG: Long Message. Message payload size greater than selected limit was received. Message processing is terminated and transfer to shared memory is discontinued. Channel resumes scanning for HDLC flags or idle codes.
		13–15	Reserved.
15:14	RSVD	0	Reserved.
13:0	DLEN[13:0]	—	Received Octets.

5.2.4.8 Next Message Pointer

The Next Message Pointer is a 32-bit dword-aligned address pointing to the first dword of a Message Descriptor which is next in a list of descriptors.

Table 5-26 lists the Next Descriptor Pointer and its description. The last Next Message Pointer should point to the first Message Descriptor and not to itself.

Table 5-26. Next Descriptor Pointer

Bit Field	Name	Value	Description
31:2	NEXTPTR[29:0]		These 30 bits are appended with 00b to form a dword-aligned 32-bit address. This address points to the next message descriptor in the list.
1:0	NEXTPTR[1:0]	0	Ensures dword alignment.

5.2.4.9 Data Buffer Pointer

The Data Buffer Pointer is a 32-bit address to the first byte of a data message in shared memory. This pointer does not have to be word- or dword-aligned.

Table 5-27 lists the Data Buffer Pointer and its description.

Table 5-27. Data Buffer Pointer

Bit Field	Name	Value	Description
31:0	DATAPTR[31:0]		The 32-bit address in this descriptor serves as a byte pointer to the first octet of a data buffer.

5.2.4.10 Message Descriptor Handling

The bit fields Inhibit Buffer Status Descriptor (INHTBSD for transmitters or INHRBSD for receivers) in the Group Configuration Descriptor specify whether or not MUSYCC writes a Buffer Status Descriptor to shared memory after the end of the current message has been detected.

If INHTBSD/INHRBSD is set to 0, MUSYCC:

1. Assumes the Message Pointer points to the current Message Descriptor.
2. Overwrites the Buffer Descriptor field with the Buffer Status Descriptor field.
3. Fetches the next Message Pointer from the descriptor.
4. Reads the next Message Descriptor.
5. Writes the pointer to the new descriptor into the Message Pointer in shared memory.

If INHTBSD/INHRBSD is set to 1, MUSYCC:

1. Assumes the Message Pointer points to the next Message Descriptor.
2. Reads the next Message Descriptor.
3. Writes the Next Message Pointer from the descriptor into the Message Pointer in shared memory.

5.2.5 Interrupt Level Descriptors

MUSYCC generates interrupts for a variety of reasons. Interrupts are events or errors detected by MUSYCC during bit-level processing of incoming serial data streams. Interrupts are generated by MUSYCC and forwarded to the host for servicing. Individual types of interrupts can be masked from being generated by setting the appropriate interrupt mask or interrupt disable bit fields in various descriptors. The interrupt mechanism, each individual interrupt, and interrupt controlling mechanisms are discussed in this section.

5.2.5.1 Interrupt Queue Descriptor

MUSYCC employs a single Interrupt Queue Descriptor to communicate interrupt information to the host. This descriptor is stored in MUSYCC in an internal register. The descriptor in this register space stores the location and size of an interrupt queue in shared memory. MUSYCC requires this information to transfer interrupt descriptors it generates to shared memory for the host to use. MUSYCC writes Interrupt Descriptors directly into the shared memory queue using PCI bus master mode. MUSYCC's PCI interface must be configured to allow bus mastering.

The Interrupt Queue Descriptor is initialized by the host issuing a service request to MUSYCC to read of a copy of the Interrupt Queue Descriptor from shared memory. Another method of initialization is for the host to directly write the information into the appropriate register space within MUSYCC.

Tables 5-28 through 5-30 list the details of the Interrupt Queue Descriptor.

Table 5-28. Interrupt Queue Descriptor

Byte Offset	Field Name	dwords	Octets
00h	Interrupt Queue Pointer	1	4
04h	Interrupt Queue Length	1	4
TOTAL		2	8

Table 5-29. Interrupt Queue Pointer

Bit Field	Name	Value	Description
31:2	IQPTR[30:0]		These 30 bits are appended with 00b to form a dword-aligned 32-bit address. This address points to the first word of the Interrupt Queue buffer.
1:0	IQPTR[1:0]	0	Ensures dword alignment.

Table 5-30. Interrupt Queue Length

Bit Field	Name	Value	Description
31:15	RSVD	0	Reserved.
14:0	IQLEN[14:0]		This 15-bit number specifies the length of the Interrupt Queue buffer in dwords. The maximum size for an interrupt queue is 32,768 dwords. This is a 0-based number. A value of 1 indicates the queue length is 2 descriptors long, the required minimum.

5.2.5.2 Interrupt Descriptor

The Interrupt Descriptor describes the format of data transferred into the queue. This 32-bit word includes bit fields for the following:

- Identifying the interrupt source from within MUSYCC. Channel group number (0–8), channel number (0–31), and direction (receive or transmit) are provided. There are 256 possible channel sources.
- Events assisting the host in synchronizing channel activities.
- Errors and unexpected conditions resulting in lost data, discontinued message processing, or prevented successful completion of a service request.
- Number of bytes transferred to or from shared memory when a memory buffer has been completely processed.

All interrupts are associated with a channel group, channel number, and direction of the channel with the following exceptions:

1. When an OOF or COFA condition is detected on a serial port, only one interrupt is generated for the entire group until the condition is cleared and the condition reoccurs. The group is identified by the GRP field, and the direction is identified by the DIR field. The CH field is the channel number currently being serviced when this condition is detected.
2. The ILOST interrupt bit indicates that one or more interrupt was lost internally due to a lack of internal queuing space. This occurs when MUSYCC generates more interrupt descriptors than can be stored in the Interrupt Queue in shared memory. The latency of host processing of the Interrupt Queue can also be a factor. This condition is conveyed by MUSYCC overwriting the ILOST bit field in the last interrupt descriptor in an internal queue prior to being transferred to shared memory. The bit field is not specific to or associated with the interrupt descriptor being overwritten. Only one bit is overwritten, and the integrity of the original descriptor is maintained.
3. The PERR interrupt bit indicates that MUSYCC detected a parity error during a PCI access cycle. This condition is conveyed by MUSYCC overwriting the PERR bit field in the last interrupt descriptor in an internal queue prior to being transferred to shared memory. The bit field is not specific to or associated with the interrupt descriptor being overwritten. Only one bit is overwritten and the integrity of the original descriptor is maintained.

Interrupt descriptors can convey certain combinations of events and errors, but no more than one event and one error. Because multiple information can be conveyed via a single interrupt descriptor, always look at both the event and error fields when servicing interrupt descriptors. Following is a list of possible combinations of events and errors.

Items Issued Separately

The following items are issued separately in their own interrupt descriptors:

- Events:
  - SACK
  - EOP
  - CHABT
  - CHIC
  - FREC
  - SINC
  - SDEC
  - SFILT
- Errors:
  - PERR
  - PROT
  - SUERR

Items That Can Combine

In the list below, a single event can combine with a single error within the same interrupt descriptor:

- Events:
  - EOB
  - EOM
- Errors:
  - BUFF
  - COFA
  - ONR
  - OOF
  - FCS
  - ALIGN
  - ABT
  - LNG
  - SHT

The ILOST error is always piggybacked onto an existing interrupt descriptor which can have an event, an error, or both bit fields set.

Table 5-31 lists details and descriptions of the interrupt descriptor.

Section 6.4.8 and 6.4.9 provide detailed explanations of the reasons, effects, and recovery actions for events and errors.

Errors reported in the buffer status descriptor are also reported in the interrupt descriptor. The occurrence in shared memory of a buffer status descriptor and the interrupt descriptor conveying the same error condition is indeterminate. The occurrence of an interrupt does not imply host ownership of the buffer status descriptor. The host must always confirm ownership of the buffer status descriptor before overwriting it.

Table 5-31. Interrupt Descriptor *(1 of 4)*

Bit Field	Field Name	Value	Interrupt Name	Group	Channel	Direction		Maskable	Description
						Tx	Rx		
31	DIR	0							Receive.
		1							Transmit.
30:29	GRP[1:0]	0–3							Least significant 2 bits of the Group Number. The most significant group number is defined in bit 14.
28:24	CH[4:0]	0–31							Channel Number.
23:20	EVENT[3:0]	0	NONE	V	V	V	V		No event to report in this interrupt.
		1	SACK	V	V	V	V		Service Request Acknowledge. Generated at conclusion of service request which was processed successfully. In case of an error as a result of a service request being executed, other interrupts can be generated; for example, PERR.
		2	EOB	V	V	V	V	V	End of Buffer. Generated when current data buffer has been completely processed, and EOBI bit field in associated Transmit Buffer Descriptor or Receive Buffer Descriptor is set. Also, EOB interrupt reports the correct number of transmitted bytes in BLEN field.(1)
		3	EOM	V	V	V	V	V	End of Message. Generated when data buffer which was just processed contained last octet of message. “Transmit EOM” means the last bit of data (not including FCS or closing FLAG) has been output on the serial port, and “Receive EOM” means the entire HDLC frame (including FCS and closing FLAG) has been written to shared memory buffer.(1)
		4	EOP	V	V	V	V	V	End of Padfill. Generated when the pad-count is enabled with non-0 value in a transmit channel, and last idle code octet is sent. This interrupt is conditioned on the end of padfill-enabled bit being set in the Transmit Channel Configuration Descriptor.
		5	CHABT	V	V		V	V	Change To Abort Code. Generated when a received pad fill code changes from 7Eh to FFh – abort code.
		6	CHIC	V	V		V	V	Change to Idle Code. Generated when a received pad fill code changes from FFh to 7Eh – idle code.
		7	FREC	V	V		V	V	Frame Recovery. Generated when serial port transitions from Out-of-Frame (OOF) back to in-frame.
		8	SINC	V	V		V	V	SS7 SUERM Octet Count Increment. Generated when in SS7 mode and Signal Unit Error Rate Monitor counter increments.
23:20	EVENT[3:0]	9	SDEC	V	V		V	V	SS7 SUERM Octet Count Decrement. Generated when in SS7 mode and Signal Unit Error Rate Monitor counter decrements.
		10	SFILT	V	V		V	V	SS7 Filtered Message. Generated when in SS7 mode and just-received message is identical to one previous message. The current message is not written to shared memory.
		11-15							Reserved.

Table 5-31. Interrupt Descriptor *(2 of 4)*

Bit Field	Field Name	Value	Interrupt Name	Group	Channel	Direction		Maskable	Description
						Tx	Rx		
19:16	ERROR[3:0]	0	NONE	V	V	V	V		No error to report in this interrupt.
		1	BUFF(2)	V	V	V	V	V	Buffer Error. Data is lost. MUSYCC has no place to read or write data internally. If from transmitter, then internal buffer underflow. If from receiver, internal buffer overflow.
		2	COFA(2)	V		V	V	V	Change of Frame Alignment. Generated when serial port is configured in channelized mode and transmitter or receiver synchronization signal assertion is detected during a bit-time, and the synchronization signal is misaligned with internal flywheel mechanism in serial interface. This condition affects all channels in the group. TCOFA - immediately deactivates all tx channels in the affected group (that includes multiple groups if non-zero PORTMAP). Therefore, after a TCOFA, MUSYCC does not update any tx message descriptor and does not generate any EOB/EOM interrupts unless the message is already sent or buffer already processed. MUSYCC stops polling any active tx channel's descriptor and the transmit serial data (TDAT) output should immediately go to the deactivated state (either all ones or three-state) ascending to TRITX setting. RCOFA - every channel currently receiving message will have its message terminated with COFA error. Every active channel will be left activated.
		3	ONR	V	V	V	V	V	Owner-Bit Error. Generated when next message pointer/descriptor is not available to MUSYCC when expected. This error is similar to BUFF error except that MUSYCC has no place to read or write data in shared memory, for example, when MUSYCC can not write out a Buffer Status Descriptor.
		4	PROT	V	V	V	V	V	Memory Protection Violation. Generated when memory protection is enabled, and MUSYCC attempts a PCI master mode access to an address outside the memory region specified in a group's Memory Protection Descriptor. The memory access is inhibited.
		5-7							Reserved.
		8	OOF(2)	V			V	V	Out of Frame. Generated when serial port is configured in channelized mode, and receiver-out-of-frame (ROOF) input signal assertion is detected.

Table 5-31. Interrupt Descriptor *(3 of 4)*

Bit Field	Field Name	Value	Interrupt Name	Group	Channel	Direction		Maskable	Description
						Tx	Rx		
19:16	ERROR[3:0]	9	FCS[2]	V	V		V	V	Frame Check Sequence Error. Generated when received HDLC frame is terminated with octet-aligned 7Eh flag, but computed FCS does not match received FCS.
		10	ALIGN[2]	V	V		V	V	Octet Alignment Error. Generated when message payload size, after zero extraction, is not a multiple of 8 bits. This generally occurs with an FCS error. This interrupt also implies an FCS error. The FCS interrupt will not be generated if the ALIGN interrupt is issued.
		11	ABT[2]	V	V		V	V	Abort Termination. Generated when received message is terminated with an abort sequence—seven sequential 1s—instead of a specific closing flag – 7Eh.
		12	LNG[2]	V	V		V	V	Long Message. Generated when received message length (after zero extraction) is greater than selected maximum message size. Message reception is terminated and not transferred to shared memory.
		13	SHT	V	V		V	V	Short Message. Generated when received message length (after zero extraction) is less than or equal to number of bits in FCS field. The message data is not transferred to shared memory.
		14	SUERR	V	V		V	V	SS7 Signal Unit Error Rate Interrupt. Generated when in SS7 mode and error monitor, SUERM, value rises past the threshold value, SUET.
		15	PERR						PCI Bus Parity Error. Generated when MUSYCC detects a parity error on data transferred into MUSYCC either from another PCI agent writing into MUSYCC, or from MUSYCC reading data from shared memory. This error is specific to the data phase of a PCI transfer while MUSYCC is receiving data. Note: PCI system error signal, SERR*, is ignored by MUSYCC. NOTE: To mask the PERR interrupt, MUSYCC's PCI Configuration Space, Function 0, Register 1, Parity Error Response field must be set to 0.[3]
15	ILOST	0	ILOST						No interrupts have been lost.
		1							Interrupt Lost. Generated when internal interrupt queue is full, and additional interrupt conditions are detected. Because MUSYCC cannot store the newest interrupt descriptors, it discards the new interrupts and overwrites this bit in the last interrupt in an internal queue prior to that interrupt being transferred out to shared memory. The integrity of the descriptor being overwritten is maintained.

Table 5-31. Interrupt Descriptor (4 of 4)

Bit Field	Field Name	Value	Interrupt Name	Group	Channel	Direction		Maskable	Description
						Tx	Rx		
14	GRP[2]								MSB of Group number.
13:0	BLEN[13:0]								This field is relevant when EVENT field is EOB (Rx and Tx) or EOM (Rx only). For Rx, it is equal to the number of octets received. For Tx, it is the size of the buffer length targeted for transmission and not necessarily the number of octets transmitted. This field is 0 all other times.

NOTE(S):
(1) Receive EOB and Receive EOM are concurrent events and are reported as a single interrupt; whereas Transmit EOB and EOM are separate events and are reported as separate interrupts (two interrupt events).
(2) Interrupt names are also reported in an error field within a receive Buffer Status Descriptor which indicates the transfer status of a message currently being processed on a channel. The order of appearance in shared memory of a Buffer Status Descriptor and an Interrupt Descriptor carrying the same error condition information is indeterminate. The host should confirm that both an Interrupt Descriptor and a Buffer Status Descriptor reports the error condition.
(3) Previously existed in bit 14 of 8474, 8472.

5.2.5.3 Interrupt Status Descriptor

The Interrupt Status Descriptor is located in a fixed position within MUSYCC's internal registers. MUSYCC updates this descriptor after each transfer of interrupt descriptors from its internal queue to the Interrupt Queue in shared memory. The host must read this descriptor from MUSYCC registers before it processes any interrupts. The interrupt status descriptor's contents are reset on hardware reset, soft chip reset, or when any field in the Interrupt Queue Descriptor is modified.

Table 5-32 lists the details of the Interrupt Status Descriptor.

Table 5-32. Interrupt Status Descriptor

Bit Field	Name	Value	Description
31	RSVD	0	Reserved.
30:16	NEXTINT[14:0]		Next Interrupt Index. 15-bit dword index from start of Interrupt Queue up to where the host has serviced Interrupt Descriptors. The host can read this value to get the location of the first unserviced descriptor in the queue. As the queue is circular, care must be taken to ensure roll-over cases at beginning and end of queue. Only the host updates this value. The NEXTINT is a read/write bit field. This is a 0-based number and equals the dword offset from Interrupt Queue Pointer.
15	INTFULL	0	Interrupt Queue Not Full—shared memory.(1)
		1	Interrupt Queue Full—shared memory.(1)
14:0	INTCNT[14:0]		Interrupt Count. 15-bit value indicates the number of interrupts pushed into the Interrupt Queue since the last reading of the Interrupt Status Descriptor. All writes to this bit field register are ignored.

NOTE(S):
(1) The INTFULL status is read—cleared bit field.

5.2.6 Interrupt Handling

5.2.6.1 Initialization

Interrupt management resources are automatically reset upon the following:

- Hardware reset
- Soft-chip reset service request
- Global initialization service request
- Read Interrupt Queue Descriptor service request
- Direct memory write to Interrupt Queue Pointer
- Direct memory write to Interrupt Queue Length

MUSYCC uses two interrupt queues: one is internal to MUSYCC and is controlled exclusively by the interrupt controller logic; the other is the Interrupt Queue in shared memory, which is allocated and administered by the host, but written to by MUSYCC.

Upon initialization, the data in the status descriptor is reset to 0s, indicating the first location for next descriptor, the queue is not full, and no descriptors are in the queue. Any existing descriptors in the internal queue are discarded.

The Interrupt Status Descriptor stores the location of the next descriptor to be read by the host, a queue full indicator, and a count of interrupts last written into shared memory since the last read of the Interrupt Status Descriptor.

The host must allocate sufficient shared memory space for the Interrupt Queue. Up to 32,768 dwords of queue space are accessible by MUSYCC, setting the upper limit for the queue size. MUSYCC requires a minimum of two dwords of queue space, setting the lower limit for the queue size.

The host must store the pointer to the queue and the queue's length in dwords in MUSYCC within the Interrupt Queue Descriptor register. This is done by issuing the appropriate service request to MUSYCC. As MUSYCC takes in the new values, it automatically resets the controller logic as indicated above. This mechanism can also be used to switch interrupt queues while MUSYCC is in full operation.

5.2.6.2 Interrupt Descriptor Generation

Interrupt conditions are detected in both error and non-error cases. MUSYCC makes a determination based on channel group, channel, and device configuration, whether reporting the condition is to be masked or whether an Interrupt Descriptor is to be sent to the host. If the interrupt is not masked, MUSYCC generates a descriptor and stores it internally prior to transfer to the Interrupt Queue in shared memory.

The internal queue is capable of holding 128 descriptors while MUSYCC arbitrates to master the PCI bus and transfer the descriptors into the Interrupt Queue in shared memory.

As the PCI bus is mastered and after descriptors are transferred to shared memory, MUSYCC updates the Interrupt Status Descriptor. In making the INTCNT field in the descriptor non-0, MUSYCC asserts the PCI INTA* signal line.

If, during the transfer of descriptors, the Interrupt Queue in shared memory becomes full, MUSYCC stops transferring descriptors until the host indicates more descriptors can be written out. MUSYCC indicates it cannot transfer more descriptors into shared memory by setting the bit field INTFULL in the Interrupt Status Descriptor. MUSYCC has enough internal space to store 128 additional descriptors.

In cases where both shared memory queue and internal queue are full and new descriptors are generated, those descriptors are discarded. MUSYCC indicates it has lost interrupts internally by overwriting the bit field ILOST in the last Interrupt Descriptor in the internal queue. The ILOST indication represents one or more lost descriptors.

5.2.6.3 INTA* Signal Line

The host must monitor the INTA* signal line at all times. An assertion on this line signifies the INTCNT filed in the Interrupt Status Descriptor is non-0. A non-0 INTCNT signifies that Interrupt Descriptors have been written to the Interrupt Queue in shared memory.

Upon detection of the INTA* assertion, the host must perform a direct read of the Interrupt Status Descriptor from within MUSYCC. This descriptor provides the offset to the location of the first unserviced descriptor in the queue, the number of unserviced descriptors, and determines if the queue is full.

The INTCNT field is reset on each read of the Interrupt Status Descriptor. As the INTCNT is reset, the INTA* signal is deasserted.

The host applies its interrupt service routines to service each of the descriptors. As the host finishes servicing a number of descriptors, it writes the offset to the location of the last serviced descriptor to the Interrupt Status Descriptor, NEXTINT. A write to this field indicates to MUSYCC that descriptor locations previously unserviced now have been serviced, and new descriptors can be written. MUSYCC continues to write to available space whether the host updates the NEXTINT field or not.

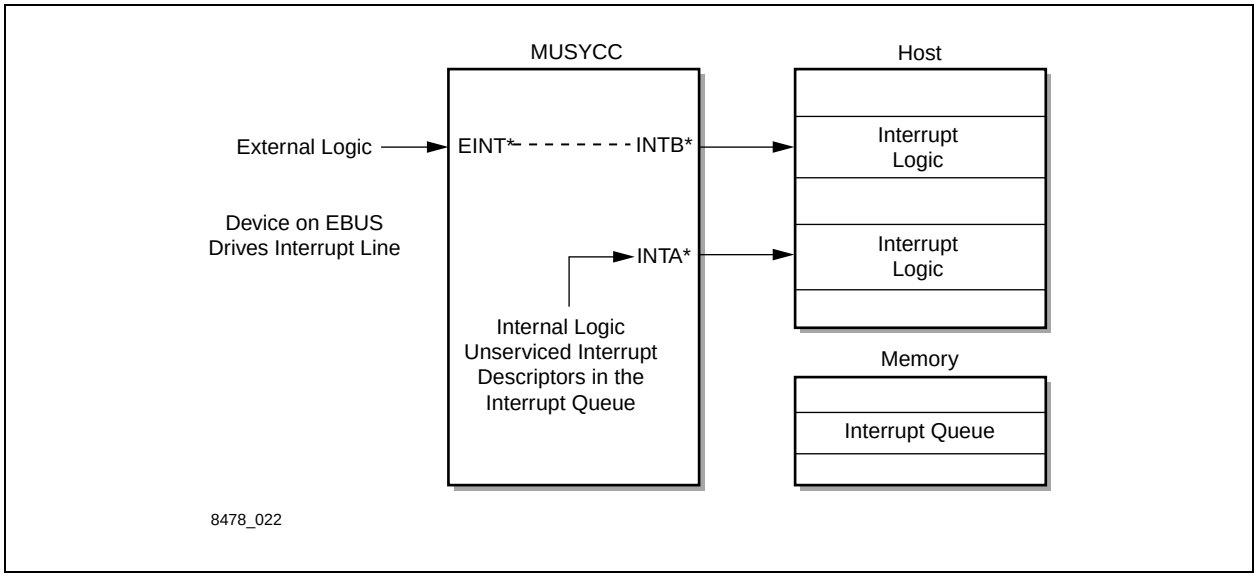
NOTE: After reading the Interrupt Status Descriptor, the host services all unserviced descriptors (count of INTCNT starting at NEXTINT) in the queue at the time of the read. If the host is unsuccessful in servicing this set of descriptors, the host must provide an alternate method of tracking unserviced descriptors. Every read of the status descriptors provides information only on new descriptors placed in the queue autonomously by MUSYCC since the last time the status descriptor was read.

5.2.6.4 INTB* Signal Line

A second interrupt signal line, the PCI INTB* signal line, is asserted by MUSYCC when it detects an assertion on the EBUS EINT* signal line. MUSYCC does not generate descriptors or use the interrupt queue for this condition because it does not know the source or reason for the interrupt. The reason is external to MUSYCC. This signal acts as an interrupt line pass-through for devices connected to the EBUS. The EINT* signal line can be tied to interrupt one or more output pins of one or more peripheral devices. As MUSYCC detects EINT* assertion, MUSYCC asserts the INTB* towards the host as long as the EINT* remains asserted.

The Figure 5-2 illustrates the operation of EINT*.

Figure 5-2. Interrupt Notification To Host



6.0 Basic Operation

6.1 Reset

There are five levels of reset state:

- Hard PCI Reset
- Soft Chip Reset
- Soft Group Reset
- Channel Activation
- Channel Deactivation

There are two ways to assert a reset:

1. Assert the PCI reset signal pin, PRST*.
2. Assert a service request through the host interface to perform the soft chip reset, soft group reset, channel activation, or channel deactivation.

6.1.1 Hard PCI Reset

The PCI reset is the most thorough level of reset in MUSYCC. All subsystems enter into their initial states. PCI reset is accomplished by asserting the PCI signal, PRST*.

The PRST* signal is an asynchronous signal on the PCI bus. The reset signal can be activated in several ways. The system must always assert the reset signal on power-up. Also, a host bus to a PCI bus bridging device should provide a way for software to assert the reset signal. Additionally, software-controlled circuitry can be included in the system design to specifically assert the reset signal on demand.

Asserting PRST* towards MUSYCC guarantees that data transfer operations and PCI device operations will not begin until MUSYCC has been properly initialized for operation. Upon entering the PCI reset state, MUSYCC outputs a three-stated signal on all output pins and stops activity on all subsystems including the host interface, serial interface, and expansion bus.

A PCI reset signal in MUSYCC takes one PCI clock cycle to complete, after which the host can communicate with MUSYCC using the PCI configuration cycles.

6.1.2 Soft Chip Reset

A Soft Chip Reset (SCR) is a device-wide reset without the host interface's PCI state being reset. Serial interface operations and EBUS operations are stopped. The soft chip reset state is entered in one of two ways:

- as a result of the PCI reset
- as a result of a soft chip reset service request

A SCR performs the following functions:

- Sets all bits to 0 in Global Configuration Descriptor register except for PORTMAP [1:0], which retains its current value.
- Sets all bits to 0 in Interrupt Status register, including NEXTINT, INTFULL, and INTCNT.
- Sets all bits to 0 in Group Configuration Descriptor register except MSKCOFA and MSKOOF which are set to 1. Thus, all supported groups (both directions) are disabled.
- Resets the interrupt write index to 0. Hence, the next interrupt is written at the location pointed to by the value of Interrupt Queue Pointer. (Present values of the Interrupt Queue Pointer and Interrupt Queue Length remain intact.)
- Deactivates all 32 channels (both directions) of each group. This action remains pending until two serial port clocks have been applied on the respective channel group input.
- Sets all bits to 0 in the following registers:

  1. Port Configuration Descriptor
  2. Memory Protection Descriptor
  3. Message Length Descriptor
  4. Service Request Descriptor

NOTE: SCR does not affect any PCI configuration register contents.

After the host requests a SCR by writing to the Service Request Descriptor, MUSYCC does not acknowledge SCR execution with any Service Request Acknowledge (SACK) Interrupt Descriptor. Although no SACK is generated, MUSYCC will have completed execution of the transmit and receive serial port SCR functions after two clock pulses are applied to the respective TCLK and RCLK serial port inputs. These serial port clocks do not have to be present when the SCR write occurs.

When writing an SCR service request, the host must ensure at least one PCI bus clock cycle has elapsed before writing another service request. To meet this minimum elapsed service request write timing interval, it is recommended that the host follow any SCR write with another service request read from the same address. Reading back the Service Request Descriptor prevents a PCI burst write from sequentially writing different values into that descriptor.

6.1.3 Soft Group Reset

Every supported channel group within MUSYCC has the ability to reset (or deactivate) a specific direction for all channels in the group using a single service request: the soft group reset service request.

When a soft group reset is requested, a direction (either transmit or receive) is specified in the request, and all channels in the specified group and direction are deactivated. For the transmit direction, output signal TDAT is three-stated.

The host must allow two line clock periods of the clock connected to the associated serial port to elapse for this reset to complete before issuing another service request.

When a soft group reset is requested by the host, the service request mechanism is used. Normally, every service request is acknowledged by MUSYCC with a SACK Interrupt Descriptor. In this case, the host should follow up the reset request with a Global Initialization service request.

6.1.4 Initialization Sequence Example

The typical power-up or reset initialization sequence is as follows:

1. PCI configuration.
2. Soft Chip Reset.
3. Allocate Shared Memory.
4. Initialize Shared Memory Channel Group Descriptors and time slot map.
5. Write the Group Base Pointer values.
6. Allocate interrupt queue and initialize it.
7. Set Message Descriptor pointers.
8. Perform Global Initialization, waiting for SACK.
9. Perform Group Initialization, waiting for SACK between each service request.
   - Group Initialization - Receive Group 0–8
   - Group Initialization - Transmit Group 0–8

NOTE: SACK for Global Initialization is not written until the Global and Interrupt Queue Descriptors are read from memory.

6.2 Configuration

A sequence of hierarchical initializations must occur after resets. The levels of hierarchy are as follows:

1. PCI Configuration—only after hardware reset
2. Global Configuration
3. Interrupt Queue Configuration
4. Channel Group(s) Configuration

6.2.1 PCI Configuration

After power-up or a PCI reset sequence, MUSYCC enters a holding pattern, waiting for PCI configuration cycles directed specifically for MUSYCC at the PCI bus and PCI slot MUSYCC resides in.

PCI configuration involves PCI read and write cycles initiated by the host and performed by a host-bus-to-PCI-bus bridge device. The cycles are executed at the hardware signal level by the bridge device. The bridge device polls all possible slots on the bus it controls for a PCI device and then iteratively reads the configuration space for all supported functions on each device. All information from the basic configuration sequence is forwarded to the system controller or host processor controlling the bridge device.

During PCI configuration, the host can perform the following configuration for MUSYCC's Function 0, HDLC Network Controller function:

- Read PCI configuration space (Device Identification, Vendor Identification, Class Code, and Revision Identification).
- Allocate 1 MB system memory range and assign the Base Address register using this memory range.
- Allow fast back-to-back transactions.
- Enable PCI system error signal line, SERR*.
- Allow response for PCI parity error detection.
- Allow PCI bus-master mode.
- Allow PCI bus-slave mode.
- Assign latency.
- Assign interrupt line routing.

During PCI configuration, the host can perform the following configuration for MUSYCC's Function 1, PCI to EBUS bridge:

- Read PCI configuration space (Device Identification, Vendor Identification, Class Code, and Revision Identification).
- Allocate 1 MB system memory range and assign the Base Address register using this memory range.
- Allow response for PCI parity error detection.
- Assign latency.
- Assign interrupt line routing.

6.2.2 Global Configuration

After PCI configuration is complete, a set of hierarchical configuration sequences must be executed to begin operation at the channel level. Global configuration is initiated by the host either issuing a service request or performing slave writes into MUSYCC resident Global Configuration Descriptor.

Global configuration specifies information used across the entire device including all supported channel groups, all channels, and the EBUS (see Table 5-6, *Global Configuration Descriptor*).

NOTE: Device identification at the PCI Level Configuration must be used to identify the number of supported channel groups and channels in MUSYCC, which, in turn, affects how MUSYCC is eventually configured.

6.2.3 Interrupt Queue Configuration

Part of global configuration involves interrupt queue configuration (see Table 5-28, *Interrupt Queue Descriptor*).

6.2.4 Channel Group(s) Configuration

After global configuration, more specific group configuration must be performed for each supported channel group.

The relevant references are as follows:

- Table 5-10, *Group Configuration Descriptor*
- Table 5-11, *Memory Protection Descriptor*
- Table 5-12, *Port Configuration Descriptor*
- Table 5-13, *Message Length Descriptor*
- Table 5-14, *Transmit or Receive Time Slot Map*
- Table 5-16, *Transmit or Receive Subchannel Map*
- Table 5-18, *Channel Configuration Descriptor*

6.2.5 Service Request Mechanism

The service request mechanism requires that the host perform a direct memory write operation (slave write) into the appropriate channel group's Service Request Descriptor that is within MUSYCC's internal registers.

The relevant references are as follows:

- Table 5-1, *MUSYCC Register Map*
- Table 5-9, *Service Request Descriptor*

6.2.6 MUSYCC Internal Memory

MUSYCC has two areas of host-accessible internal memories. One is the Internal RAM (IRAM) and is accessed through MUSYCC's Direct Memory Access Controller (DMAC). The IRAM area contains the following descriptors and maps:

- Table 5-14, *Transmit or Receive Time Slot Map*
- Table 5-16, *Transmit or Receive Subchannel Map*
- Table 5-18, *Channel Configuration Descriptor* (transmit or receive)

A second area of internal memories makes up the Host Interface registers. This area is not accessed through MUSYCC's DMAC. The Host Interface register contains the following descriptors:

- Table 5-6, *Global Configuration Descriptor*
- Table 5-7, *Dual Address Cycle Base Pointer*
- Table 5-8, *Group Base Pointer*
- Table 5-9, *Service Request Descriptor*
- Table 5-10, *Group Configuration Descriptor*
- Table 5-11, *Memory Protection Descriptor*
- Table 5-12, *Port Configuration Descriptor*
- Table 5-13, *Message Length Descriptor*
- Table 5-14, *Transmit or Receive Time Slot Map*
- Table 5-16, *Transmit or Receive Subchannel Map*
- Table 5-18, *Channel Configuration Descriptor*
- Table 5-28, *Interrupt Queue Descriptor*

6.2.6.1 Memory Operations—Inactive Channels

When all channels are deactivated, the IRAM and Host Interface registers can be read and written. The IRAM registers require that the corresponding channel group's line clocks (TCLK, RCLK) are active. Reading from any IRAM register with inactive line clocks returns the pattern DEAD ACCEh—conveying "dead access". Writing to any IRAM register with inactive line clocks returns in the writes being ignored.

Read operations to invalid (unsupported to reserved) addresses or write-only registers return all 1s. Write operations to invalid (unsupported or reserved) addresses or read-only register bits result in the write to that bit location being ignored.

6.2.6.2 Memory Operations—Active Channels

There are only a few locations that are allowed to be slave-accessed after MUSYCC has an active channel:

- Group Base Pointer
- Service Request Descriptor
- Interrupt Status Descriptor
- Any EBUS function 1 location

The host must not perform PCI slave accesses to any other register after MUSYCC has a channel activated on any group. Any attempt to read or write to other MUSYCC registers as a PCI slave device while channels are activated can result in DMAC lock-up and spontaneous (unreported) channel deactivation. This limitation is inclusive of all groups; for example, it is not acceptable to perform a slave write to the group 2 time slot map while there is an active channel on group 0.

All of the slave writes can be accomplished with initial service requests after setting the appropriate descriptor value in shared memory. Also, any value that could be read directly from MUSYCC can more easily be read directly from the descriptors in shared memory.

6.3 Channel Operation

To start any channel processing, a series of shared memory segments must be obtained by the host and initialized as specific descriptors which MUSYCC can use to control its channel processing operations.

To illustrate the required MUSYCC configuration, assume the following:

- Port 0 is physically wired to a PCM carrying E1 signal (2.048 Mbps).
- EBUS is not used.
- PCI configuration is displayed in Table 5-3, *MUSYCC PCI Function Memory Allocation*.
- Memory Protection is enabled for range 0x00100000 to 0x001FFFFF.
- Application:
  - Port 0 is configured for 32 channel operation, E1 signal, 2.048 Mbps.
  - Transmit and Receive time slots are mapped identically.
  - Time slot 0 is mapped to logical channel 0 (64 kbps).
  - Time slot 1 bits 0–3 are mapped to logical channel 1 (32 kbps subchannel).
  - Time slot 2–3 are mapped to logical channel 2 (128 kbps hyperchannel).
  - 16-bit FCS HDLC.
  - Maximum message length is 1024 octets for channel 0.
  - Maximum message length is 512 octets for channel 1.
  - Maximum message length check is disabled for channel 2.
  - No SS7 functions.
  - Idle Code = 7Eh.
  - Pad Fill Count = 0.
- C-Language support.
- Each section below builds on the previous sections.

6.3.1 Group Structure

A group structure (one per supported group) must be allocated in shared memory. A data structure is instrumental in keeping the memory spaces for the various descriptors required for a channel group configuration in a sequential order and at exact offsets from the beginning of the group structure.

Once a group structure is allocated in shared memory, all descriptor spaces are allocated within the group structure.

Service request handling within MUSYCC requires the group structure and descriptor contents be at an exact offsets within the structure.

For this example, the following group structure declaration is used:

```
/* Reference: Chapter “Memory Organization” */

#define SIZE_OF_GROUP_STRUCTURE 1564
#define NUM_GROUPS 1
#define BOUNDARY 2048

#define MUSYCC_FUNC_0_BAR 0x00900000 /* system usually assigns this */

/* declare variable */

typedef struct tGROUP_STRUCTURE
{
unsigned long *pGroupBase;
unsigned long *pDualAddressCycleBase;
unsigned long ServiceRequestDescr;
unsigned long InterruptStatusDescr;
unsigned char Txtime slotMap[128];
unsigned char TxSubchannelMap[256];
unsigned char TxChannelConfigDescr[128];
unsigned char Rxtime slotMap[128];
unsigned char RxSubchannelMap[256];
unsigned char RxChannelConfigDescr[128];
unsigned long GlobalConfigDescr;
unsigned long InterruptDescr[2];
unsigned long GroupConfigDescr;
unsigned long MemoryProtectDescr;
unsigned long MessageLengthDescr;
unsigned long PortConfigDescr;
} tGROUP_STRUCTURE;

/* IMPORTANT NOTE: Byte padding within the structure would cause descriptor */
/* offsets from the beginning of the structure to move. MUSYCC requires every */
/* offset to be fixed at all times. Byte padding is an automatic function of */
/* many compilers. */

/* allocate space */

tGROUP_STRUCTURE GroupStr0; /* one per supported group */

/* fixed descriptor offsets into the group structure */

#define GROUP_BASE_OFFSET..............0x00000000
#define DUAL_ADDRESS_CYCLE_BASE_OFFSET..0x00000004
#define SERVICE_REQUEST_OFFSET..........0x00000008
#define INTERRUPT_STATUS_OFFSET.........0x0000000C
#define TX_time slot_MAP_OFFSET..........0x00000200
#define TX_SUBCHANNEL_MAP_OFFSET........0x00000280
#define TX_CHANNEL_CONFIG_DESCR_OFFSET..0x00000380
#define RX_time slot_MAP_OFFSET..........0x00000400
#define RX_SUBCHANNEL_MAP_OFFSET........0x00000480
#define RX_CHANNEL_CONFIG_DESCR_OFFSET..0x00000580
#define GLOBAL_CONFIG_DESCR_OFFSET......0x00000600
#define INT_QUEUE_DESCR_OFFSET..........0x00000604
#define INT_QUEUE_POINTER_OFFSET........0x00000604
#define INT_QUEUE_LENGTH_OFFSET.........0x00000608
#define GROUP_CONFIG_DESCR_OFFSET.......0x0000060C
#define MEMORY_PROTECT_DESCR_OFFSET.....0x00000610
#define MESSAGE_LENGTH_DESCR_OFFSET.....0x00000614
#define PORT_CONFIG_DESCR_OFFSET........0x00000618
```

6.3.2 Group Base Pointer

For the Group Base Pointer the host must allocate a 2 kB bound memory segment for Channel Group 0. The value calculated as the address for a Group Base Structure must be written into MUSYCC's Group 0 Base Pointer register using a PCI write cycle. After PCI configuration, this is a simple memory access by the host.

A service request does not exist to update this pointer in MUSYCC because all service requests reference this pointer value to gain access to the shared memory resident group structure and the descriptors within it.

The components of the Group Base Pointer are listed in Table 6-1.

```
#define SIZE_OF_GROUP_STRUCTURE 1564
#define GROUP_STR_BOUNDARY 2048

GroupStr0.pGroupBase = malloc( SIZE_OF_GROUP_STRUCTURE + GROUP_STR_BOUNDARY );
GroupStr0.pGroupBase = ( GroupStr0.pGroupBase + GROUP_STR_BOUNDARY ) &
        ~(GROUP_STR_BOUNDARY -1);

/* group base pointer pointers must be a 2K byte aligned address */
/* above, there is enough space to first move forward 2K bytes, then */
/* lop off the 2K automatically. This will bring the pointer back */
/* to the original address or give us the next 2K byte boundary address */

/* IMPORTANT NOTE: be sure to save away the original pointer returned by the */
/* memory access routine as that same value will be required to free the space.
        */

/* must write directly into MUSYCC register */
*(MUSYCC_FUNC_0_BAR + GROUP_BASE_OFFSET) = GroupStr0.pGroupBase;
```

Table 6-1. Example—Components of Group Base Pointer

Descriptor	Component of Descriptor	Value of Components
Group Base Pointer	Pointer to a shared memory segment large enough for all configuration descriptors for Channel Group 0.	Return pointer from "malloc ( )" adjusted to a 2 kB boundary.

6.3.3 Global Configuration Descriptor

```
/* CLOCK activity indicators - read only, writes are ignored */
/* MPU control - assume EBUS is not used and default values are fine */
/* PORTMAP = 0, PORT 0 mapped to CHANNEL GROUP 0 */
GroupStr0.GlobalConfigDescr = 0x00000000;

/* either write directly into MUSYCC register - or - use a service request */
*(MUSYCC_FUNC_0_BAR + GLOBAL_CONFIG_DESCR_OFFSET) =
        GroupStr0.GlobalConfigDescr;
```

The components of the Global Configuration Descriptor are listed in Table 6-2.

Table 6-2. Example—Components of Global Configuration Descriptor

Descriptor	Component of Descriptor	Value of Components
Global Configuration Descriptor	TXCLKACT0 TXCLKACT1 TXCLKACT2 TXCLKACT3 RXCLKACT0 RXCLKACT1 RXCLKACT2 RXCLKACT3	Read only values. Writes are ignored. It would be ideal to read this descriptor first and verify that MUSYCC detects clocks at the Rx and Tx ports being used.
	BLAPSE ECKEN MPUSEL ALAPSE ELAPSE	0 = Don't care. 0 = EBUS clock output disabled. 0 = Motorola-style protocol supported. 0 = Don't care. 0 = Don't care.
	PORTMAP	Use 0 where the mapping is defined as: Port 0 -> Channel Group 0 Port 1 -> Channel Group 1 Port 2 -> Channel Group 2 Port 3 -> Channel Group 3

6.3.4 Interrupt Queue Descriptor

```
#define BYTES_PER_INT_DESCR 4 /* recall, dword = 32-bits or 4 bytes */
#define NUM_INT_DESCR_NEEDED 10 /* assumption (min = 2, max = 32768) */
#define SIZE_OF_INTERRUPT_QUEUE (BYTES_PER_INT_DESCR * NUM_INT_DESCR_NEEDED)
#define INT_QUEUE_BOUNDARY 4

/* local variables */
unsigned long *pIntQueue;
unsigned long IntQueueLen;

pIntQueue = malloc( SIZE_OF_INTERRUPT_QUEUE + INT_QUEUE_BOUNDARY );
pIntQueue = (pIntQueue + INT_QUEUE_BOUNDARY) & ~(INT_QUEUE_BOUNDARY – 1);

/* interrupt queue pointers must be a dword aligned address */
/* above, there is enough space to first move forward 4 bytes, then */
/* lop off the 4 automatically. This will bring the pointer back */
/* to the original address or give us the next 4 byte boundary address */

/* IMPORTANT NOTE: be sure to save away the original pointer returned by the */
/* memory access routine as that same value will be required to free the space.
       */
IntQueueLen = NUM_INT_DESCR_NEEDED - 1; /* 0-based */

GroupStr0.IntQueueDescr[0] = pIntQueue;
GroupStr0.IntQueueDescr[1] = IntQueueLen;

/* either write directly into MUSYCC register - or - use a service request */
/* for this descriptor, 2 dwords need to be written, so 2 write accesses */
*(MUSYCC_FUNC_0_BAR + INT_QUEUE_POINTER_OFFSET)= GroupStr0.IntQueueDescr[0];
*(MUSYCC_FUNC_0_BAR + INT_QUEUE_LENGTH_OFFSET) = GroupStr0.IntQueueDescr[1];
```

The components of the Interrupt Queue Descriptor are listed in Table 6-3.

Table 6-3. Example—Components of Interrupt Queue Descriptor

Descriptor	Component of Descriptor	Value of Components
Interrupt Queue Descriptor	IQPTR	pIntQueue, provided by memory allocation functions and adjusted to be dword bound
	IQLEN	IntQueueLen, specified by #define (0-based)

6.3.5 Group Configuration Descriptor

```
/* signal unit error threshold = 0, no SS7 support required */
/* sf alignment = 0, use internal flywheel mechanism after initial frame sync */
/* poll throttle = 1, poll every 16th frame */
/* inhibit tx bsd = 0, do not inhibit */
/* inhibit rx bsd = 0, do not inhibit */
/* memory protection violation action = 0, reset group on detection
/* message config bit copy = 1, enable */
/* mask cofa interrupt = 0, do not mask */
/* mask oof interrupt = 0, do not mask */
/* oof message processing = 0, continue processing incoming messages */
/* subchanneling = 0, enabled */
/* transmitter enabled = 1, enabled */
/* receiver enabled = 1, enabled */
GroupStr0.GroupConfigDescr = 0x00000443;

/* either write directly into MUSYCC register - or - use a service request */
*(MUSYCC_FUNC_0_BAR + GROUP_CONFIG_DESCR_OFFSET) = GroupStr0.GroupConfigDescr;
```

The components of the Group Configuration Descriptor are listed in Table 6-4.

Table 6-4. Example—Components of Group Configuration Descriptor

Descriptor	Component of Descriptor	Value of Components
Group Configuration Descriptor	SUET	0 = SS7 not being used
	SFALIGN	0 = Use internal flywheel mechanism
	POLLTH	1 = Poll buffer ownership every 16th frame per channel
	INHTBSD	0 = Allow tx buffer status descriptor writes
	INHRBSD	0 = Allow rx buffer status descriptor writes
	MEMPVA	0 = On memory protection violation, reset group
	MCENBL	1 = Message configuration bits copy enabled
	MSKCOFA	0 = Do not mask COFA interrupt
	MSKOOF	0 = Do not mask OOF interrupt
	OOFABT	0 = On OOF detection, continue processing channel
	SUBDSBL	0 = Subchanneling enabled
	TXENBL	1 = Transmitter enabled
	RXENBL	1 = Receiver enabled

6.3.6 Memory Protection Descriptor

```
/* memory protection disabled = 0 */
GroupStr0.MemoryProtectDescr = 0x00000000;

/* either write directly into MUSYCC register - or - use a service request */
*(MUSYCC_FUNC_0_BAR + GROUP_CONFIG_DESCR_OFFSET) =
        GroupStr0.MemoryProtectDescr;
```

The components of the Memory Protection Descriptor are listed in Table 6-5.

Table 6-5. Example—Components of Memory Protection Descriptor

Descriptor	Component of Descriptor	Value of Components
Memory Protection Descriptor	PROTENBL	0 = Memory protection disabled

6.3.7 Port Configuration Descriptor

```
/* three-state transmitter output = 0 */
/* rx out of frame signal active edge = 0, falling edge */
/* rx synchronization signal active edge = 0, falling edge */
/* rx data signal active edge = 0, falling edge */
/* tx synchronization signal active edge = 0, falling edge */
/* tx data signal active edge = 0, falling edge */
/* port mode = 1, E1 32 time slot */
GroupStr0.PortConfigDescr = 0x00000001;

/* either write directly into MUSYCC register - or - use a service request */
*(MUSYCC_FUNC_0_BAR + PORT_CONFIG_DESCR_OFFSET) = GroupStr0.PortConfigDescr;
```

The components of the Port Configuration Descriptor are listed in Table 6-6.

Table 6-6. Example—Components of Port Configuration Descriptor

Descriptor	Component of Descriptor	Value of Components
Port Configuration Descriptor	TRITX	0 = Three-state output when transmitter enabled and time slot is not mapped
	ROOF_EDGE RSYNC_EDGE RDAT_EDGE TSYNC_EDGE TDAT_EDGE	0 = Active edge of signals is falling edge
	PORTMD	1 = 32 channel with E1 signalling

6.3.8 Message Length Descriptor

```
/* maximum frame length register 2 = 0x200 */
/* maximum frame length register 1 = 0x400 */
GroupStr0.MessageLengthDescr = 0x02000400;

/* either write directly into MUSYCC register - or - use a service request */
*(MUSYCC_FUNC_0_BAR + MESSAGE_LENGTH_DESCR_OFFSET) =
        GroupStr0.MessageLengthDescr;
```

The components of the Message Length Descriptor are listed in Table 6-7.

Table 6-7. Example—Components of Message Length Descriptor

Descriptor	Component of Descriptor	Value of Components
Message Length Descriptor	MAXFRM2	0x200, register 2 to 512 octets
	MAXFRM1	0x400, register 1 to 1024 octets

6.3.9 Transmit Time Slot Map—Channel 0

```
/* each time slot descriptor contains 4 time slot assignments */
/* each byte in the dword descriptor is a time slot assignment */
/* byte 0/dword 0 is for time slot 0 */
/* byte 1/dword 0 is for time slot 1 */
/* byte 2/dword 0 is for time slot 2 */
/* byte 3/dword 0 is for time slot 3 */
/* for demonstration, assign each byte separately */
GroupStr0.Txtime slotMap[0] = 0; /* zero it out for demo purposes */

/* time slot 0, channel number assigned = 0 */
/* time slot 0, time slot enabled code = 4, enabled and 64 kbps */
GroupStr0.Txtime slotMap[0] |= 0x00000080;

/* time slot 1, channel number assigned = 1 */
/* time slot 1, time slot enabled code = 7, subchannel w/ 1st bit active */
GroupStr0.Txtime slotMap[1] |= 0x0000D100;

/* time slot 2, channel number assigned = 2 */
/* time slot 2, time slot enabled code = 4, enabled and 64 kbps */
GroupStr0.Txtime slotMap[2] |= 0x00820000;

/* time slot 3, channel number assigned = 2 */
/* time slot 3, time slot enabled code = 4, enabled and 64 kbps */
GroupStr0.Txtime slotMap[3] |= 0x82000000;

/* either write directly into MUSYCC register - or - use a service request */
*(MUSYCC_FUNC_0_BAR + TX_time slot_MAP_OFFSET) = GroupStr0.Txtime slotMap[0];

/* the value for the first dword becomes Tx time slot Map = 0x8282D180 */
```

The components of the Transmit Time Slot map are listed in Table 6-8.

Table 6-8. Example—Components of Transmit Time Slot Map – Channel 0

Descriptor	Component of Descriptor	Value of Components
Time Slot Map	TSEN3/7	4 = Time slot enabled w/ 64 kbps mode
	CH3/7	2 = Logical channel 2
	TSEN2/6	4 = Time slot enabled w/ 64 kbps
	CH2/6	2 = Logical channel 2
	TSEN1/5	7 = Time slot enables subchannel mode w/ first bit
	CH1/5	1 = Logical channel 1
	TSEN0/4	4 = Time slot enabled w/ 64 kbps mode
	CH0/4	0 = Logical channel 0

6.3.10 Transmit Subchannel Map

```
/* each subchannel descriptor is made up of 2 dwords */
/* dword 0 defines the configuration of 1st 4 subchannel bits */
/* dword 1 defines the configuration of 2nd 4 subchannel bits */
/* for demonstration, assign each byte separately */
/* for this example, only logical channel 1 is being subchanneled */
/* dword 0 and 1, for logical channel 0 */
/* dword 2 and 3, for logical channel 1 */
/* dword 4 and 5, for logical channel 2 */
GroupStr0.TxSubchannelMap[4] = 0; /* zero it out for demo purposes */
GroupStr0.TxSubchannelMap[5] = 0; /* zero it out for demo purposes */

/* time slot 1, bit-0 enabled and assigned to channel 1 in time slot map */

/* time slot 1, bit-1 enabled and assigned to channel 1 in subchannel map */
GroupStr0.TxSubchannelMap[4] |= 0x00008100;

/* time slot 1, bit-2 enabled and assigned to channel 1 in subchannel map */
GroupStr0.TxSubchannelMap[4] |= 0x00810000;

/* time slot 1, bit-3 enabled and assigned to channel 1 in subchannel map */
GroupStr0.TxSubchannelMap[4] |= 0x81000000;

/* either write directly into MUSYCC register - or - use a service request */
*(MUSYCC_FUNC_0_BAR + TX_SUBCHANNEL_MAP_OFFSET + 4) =
       GroupStr0.TxSubchannelMap[0];
*(MUSYCC_FUNC_0_BAR + TX_SUBCHANNEL_MAP_OFFSET + 5) =
       GroupStr0.TxSubchannelMap[0];

/* note +4 and +5 dword offsets into the subchannel map are for channel 1 bits
       */
/* the value for the 4th dword becomes Tx Subchannel Map = 0x81818100 */
/* the value for the 5th dword becomes Tx Subchannel Map = 0x00000000 */
```

The components of the Transmit Subchannel map are listed in Table 6-9.

Table 6-9. Example—Components of Transmit Subchannel Map

Descriptor	Component of Descriptor	Value of Components
Subchannel Map (dword 4)	BITEN3/7	1 = Bit 3 enabled
	CH3/7	1 = Logical channel 1
	BITEN2/6	1 = Bit 2 enabled
	CH2/6	1 = Logical channel 1
	BITEN1/5	1 = Bit 1 enabled
	CH1/5	1 = Logical channel 1
	BITEN0/4	0 = Bit 0 not assigned here, see Time Slot Map
	CH0/4	0 = Bit 0 not assigned here see Time Slot Map
Subchannel Map (dword 5)	TSEN3/7	0 = Bit 7 disabled
	CH3/7	0 = Don't care
	TSEN2/6	0 = Bit 6 disabled
	CH2/6	0 = Don't care
	TSEN1/5	0 = Bit 5 disabled
	CH1/5	0 = Don't care
	TSEN0/4	0 = Bit 4 disabled
	CH0/4	0 = Don't care

6.3.11 Transmit Channel Configuration Descriptor

```
/* each transmit channel descriptor is made up of 1 dwords */
/* need to define channel 0, 1, and 2 - 3 dwords total */
/* dword 0 for logical channel 0 */
/* dword 1 for logical channel 1 */
/* dword 2 for logical channel 2 */

/* for logical channel 0 */
/* pad count adjustment = 0, disabled */
/* buffer location index = 0 */
/* data inversion = 0, disabled */
/* internal buffer length = 0 */
/* end of padfill interrupt = 0, disabled */
/* protocol = 2, hdlc-16-fcs */
/* message length check register = 1, use register 1 */
/* fcs transfer = 0, normal, do not transfer rx fcs into shared memory */
/* mask suerr interrupt = 0, do not mask, enable interrupt */
/* mask sinc. interrupt = 0, do not mask, enable interrupt */
/* mask sdec. interrupt = 0, do not mask, enable interrupt */
/* mask sfilt interrupt = 0, do not mask, enable interrupt */
/* mask idle. interrupt = 0, do not mask, enable interrupt */
/* mask msg interrupt = 0, do not mask, enable interrupt */
/* mask eom interrupt = 0, do not mask, enable interrupt */
/* mask buff. interrupt = 0, do not mask, enable interrupt */
```

```
GroupStr0.TxChannelConfigDescr[0] = 0x000024000;

/* for logical channel 1 */
/* everything same except as logical channel 0 */
/* buffer location index = 1 */
/* internal buffer length = 0 */
/* message length check register = 2, use register 2 */
GroupStr0.TxChannelConfigDescr[1] = 0x01002800;

/* for logical channel 2 */
/* everything same except as logical channel 0 */
/* buffer location index = 2 */
/* internal buffer length = 0 */
/* message length check register = 0, do not check */
GroupStr0.TxChannelConfigDescr[2] = 0x02002000;


/* either write directly into MUSYCC register - or - use a service request */
*(MUSYCC_FUNC_0_BAR + TX_CHANNEL_CONFIG_DESCR_OFFSET +0) =
        GroupStr0.TxChannelConfigDescr[0];

*(MUSYCC_FUNC_0_BAR + TX_CHANNEL_CONFIG_DESCR_OFFSET +1) =
        GroupStr0.TxChannelConfigDescr[1];

*(MUSYCC_FUNC_0_BAR + TX_CHANNEL_CONFIG_DESCR_OFFSET +2) =
        GroupStr0.TxChannelConfigDescr[2];
```

The components of the Channel Configuration Descriptor are listed in Table 6-10.

Table 6-10. Example—Components of Channel Configuration Descriptor

Descriptor	Component of Descriptor	Value of Components
Transmit Channel Configuration Descriptor	PADJ	0 = Pad count adjustment disabled
	BUFFLOC	0 = For logical channel 0 1 = For logical channel 1 2 = For logical channel 2
	INV	0 = Data inversion disabled
	BUFFLEN	0 = Total FIFO = (0+1)*2 = 2 dwords
	EOPI	0 = End-of-padfill interrupt disabled
	PROTOCOL	2 = HDLC w/ 16-bit FC
	MAXSEL	1 = For logical channel 0 application 2 = For logical channel 1 application 0 = For logical channel 2 application
	FCS	0 = FCS transfer normal
	MSKSUERR MSKSINC MSKSDEC MSKSFILT MSKIDLE MSKMSG MSKEOM MSKBUFF	0 = Interrupt masking disabled therefore enabling these interrupts

6.3.12 Receive Time Slot Map

Same as Transmit Time Slot Map.

6.3.13 Receive Subchannel Map

Same as Transmit Subchannel Map.

6.3.14 Receive Channel Configuration Descriptor

Same as Transmit Channel Configuration Descriptor.

6.3.15 Message Lists

Message lists contain data transmitted or received by MUSYCC. Message lists always reside in shared memory. Upon channel activation, MUSYCC traverses the list and either takes data from data buffers (Tx) or puts data into data buffers (Rx). Each direction of a channel must be assigned a message list before the direction of that channel is activated.

A message list is a singly-linked list of Message Descriptors. A Message Descriptor consists of a Buffer Descriptor (1 dword), a Data Pointer (1 dword), and a Next Descriptor Pointer (1 dword). (For further information, refer to Table 5-19, *Message Descriptor*).

The Buffer Descriptor contains a set of bit fields which instruct MUSYCC how to behave after the data is put in or taken out of a data buffer.

The Data Pointer contains an address in shared memory where MUSYCC can take data to be transmitted or store data received from the corresponding channel.

The Next Descriptor Pointer contains an address of a Message Descriptor in shared memory where MUSYCC can access the “next” Message Descriptor in the linked list.

To terminate a message list, the contents of the Next Descriptor Pointer in the last descriptor in a list can point either to the address of the last descriptor or to a general purpose “terminate” Message Descriptor that can be used by any message list to represent the end of the list. Thus, the OWNER bit field in the last descriptor’s Buffer Descriptor must eventually indicate that the host owns the buffer. This bit value is opposite for receive and transmit buffer ownership.

The OWNER bit field mechanism controls the termination of the message list as MUSYCC reads in each Message Descriptor in the linked list: it first checks the OWNER bit field to see if it, and not the host, owns the buffer. If it does own the descriptor, after servicing the contents of this descriptor, MUSYCC reverses the OWNER bit field to hand the descriptor back to the host; if it does not own the buffer, the end of the message list is automatically concluded. The channel stays active and, depending on other bit field values in the Buffer Descriptor, MUSYCC either polls this last descriptor regularly to see if the OWNER bit value has changed, or it idles the channel and awaits another channel activation or channel jump request. The former is useful in continuing a message list while retaining the original list. The latter is useful in starting a message list from the top element in the list.

The host processor must never change a descriptor in a buffer to which it has already granted MUSYCC ownership. The Owner bit is the only handshake mechanism to prevent race conditions.

The following describes a general sequence for setting up the transmit Message Descriptor for a single channel:

```
/* assume transmit channel is currently deactivated */
/* assume that a 1024-byte message is separated into four 256-byte data buffers
      */
/* Because four buffers will be used, four 12-byte segments [or one 48-byte */
/* of shared memory is required */

/* declare structures */

typedef tDATA_BUFFER
{
unsigned char Data[256];
} DATA_BUFFER;

typedef tMSG_DESCR
{
unsigned long BufferDescr;
struct DATA_BUFFER *pDataBuffer;
struct MSG_DESCR *pNextMsgDescr;
} MSG_DESCR;

/* allocate space */
MSG_DESCR *pTxMsgDescr[4];
DATA_BUFFER *pDataBuf[4];

/* link the message descriptors together. Terminate the message list by */
/* assigning the “next” pointer in the last descriptor to point to the last */
/* descriptor itself */

pTxMsgDescr[0]->pNextMsgDescr = pTxMsgDescr[1];
pTxMsgDescr[1]->pNextMsgDescr = pTxMsgDescr[2];
pTxMsgDescr[2]->pNextMsgDescr = pTxMsgDescr[3];
pTxMsgDescr[3]->pNextMsgDescr = pTxMsgDescr[0];

/* Note: last descriptor points to itself */

/* assign each message descriptor a data buffer */
pTxMsgDescr[0]->pDataBuffer[0] = pDataBuf[0];
pTxMsgDescr[1]->pDataBuffer[1] = pDataBuf[1];
pTxMsgDescr[2]->pDataBuffer[2] = pDataBuf[2];
pTxMsgDescr[3]->pDataBuffer[3] = pDataBuf[3];

/* set the value for each buffer descriptor in each message descriptor */
/* OWNER bit to MUSYCC, for tx buffers set to 1, for rx set to 0 */
/* NP bit to enable polling, set to 0 */
/* EOM bit if the last data buffer is associated with this descriptor */
/* EOBI bit to enable end-of-buffer interrupt, set to 1 */
/* IC field to set the idle-code to 7Eh, set to 0 */
/* PADEN field to disable pad fill, set to 0 */
/* PADCNT field to specify 0 pad fill codes, set to 0 */
/* REPEAT bit to disable message retransmission, set to 0 */
/* BLEN field set to the length of the data buffer, set to 256 */

/* msg descr 0 */
pTxMsgDescr[0]->BufferDescr = 0x90000200;

/* msg descr 1 */
```

```
pTxMsgDescr[1]->BufferDescr = 0x90000200;

/* msg descr 2 */
pTxMsgDescr[2]->BufferDescr = 0x90000200;

/* msg descr 3 */
/* only difference is EOM bit */
pTxMsgDescr[3]->BufferDescr = 0x92000200;

/* fill data buffer with outbound traffic. each buffer contains 256-bytes of
      data */

/* set the head pointer, for example for channel 0, to point to the top of */
/* the just formed message descriptor list */

/* activate transmit channel by issuing a service request */
/* ServiceRequest( ACTIVATE_CHANNEL, Group, Channel, Direction ); */
```

6.3.16 Channel Activation

After the previous levels of configuration are completed, individual channels within a channel group are ready to be activated. Service requests activate channels.

Each channel within a channel group consists of a transmitter and receiver section. Each section is independent of the other and maintains its own state machine, configuration registers, and internal resources. To activate both transmitter and receiver sections, two separate service requests are required, one directed to the transmitter and one to the receiver. MUSYCC responds to each service request with the SACK Interrupt Descriptor, notifying the host that the task was completed.

Channel Activation is an asynchronous command from the host interface to a transmit or receive section of a channel to jump to a new message. Message Descriptors in shared memory describe the attributes of the new message, what to do between messages, and the location of message data buffers in memory to use for transmit data or receive data.

6.3.16.1 Transmit Channel Activation

The following describes what MUSYCC does when the *transmit* channel is activated:

1. Reads the Tx Head Pointer for the channel from shared memory, and stores it in the Internal Channel Descriptor map.
2. Reads the Message Descriptor pointed to by Tx Head Pointer, and stores it in Internal Channel Descriptor map.
3. Checks bit field OWNER and NP in Buffer Descriptor.
   If OWNER = 1, MUSYCC is the buffer owner. Load Channel Descriptor map with data buffer pointer and data buffer length. Go to 4.
   If OWNER = 0, MUSYCC is not the buffer owner. Can enter polling mode until MUSYCC owns buffer or receives another service request to activate channel or jump to new message. Repeat 3.

4. Checks bit field INHTBSD in Group Configuration Descriptor.
   If INHTBSD = 0 (i.e., MUSYCC is allowed to overwrite Buffer Descriptor with a Buffer Status Descriptor), the address of the Buffer Descriptor is stored in the Message Pointer slot in shared memory. After the current message is completely processed, MUSYCC reads in the Next Message Pointer and overwrites Buffer Descriptor. Go to 3.
   If INHTBSD = 1 (i.e., MUSYCC is not allowed to overwrite Buffer Descriptor), the address of Next Message Pointer is pre-fetched from the current Message Descriptor and stored in the Message Pointer slot in shared memory. After the current message is completely processed, MUSYCC jumps to this pointer and starts processing a new message. Go to 5 when current message is processed, otherwise repeat 4. Simultaneously, MUSYCC masters the PCI bus and reads data into transmit FIFO from shared memory, and the serial port outputs data using the control lines TCLK, TSYNC, and TDAT as appropriate.
5. At end of message, Interrupt Descriptors and Buffer Status Descriptors can be written out to shared memory (see 3)—depending on the masking of interrupts and allowance of Buffer Descriptor overwrites.
6. Read Next Message Descriptor.
7. Go to 3.

6.3.16.2 Receive Channel Activation

The following describes what MUSYCC does when the *receive* channel is activated:

1. Reads the Rx Head Pointer for the specified channel from shared memory, and stores in the Internal Channel Descriptor map.
2. Reads the Message Descriptor pointed to by Head Pointer, and stores in the Channel Descriptor map.
3. Check bit field OWNER and NP in Buffer Descriptor.
   If OWNER = 0, MUSYCC is the buffer owner. Load Channel Descriptor map with data buffer pointer and data buffer length. Go to 4.
   If OWNER = 1, MUSYCC is not the buffer owner. May enter polling mode until MUSYCC owns buffer or receives another service request to activate channel. Repeat 3.
4. Check bit field INHRBSD in Group Configuration Descriptor.
   If INHRBSD = 0 (i.e., MUSYCC is allowed to overwrite Buffer Descriptor with a Buffer Status Descriptor), the address of the Buffer Descriptor is stored in the Message Pointer slot in shared memory. After current message is completely processed, MUSYCC reads in Next Message Pointer and overwrites Buffer Descriptor. Go to 3.
   If INHRBSD = 1 (i.e., MUSYCC is not allowed to overwrite Buffer Descriptor), the address of Next Message Pointer is pre-fetched from the current Message Descriptor and stored in the Message Pointer slot in shared memory. After the current message is completely processed, MUSYCC jumps to this pointer and starts processing the new message. Go to 5 when current message is processed, otherwise repeat 4. Simultaneously, the receiver is configured and data is sampled in from the serial port using control lines RCLK, RSYNC, RDAT, and ROOF as appropriate, and, MUSYCC masters the PCI bus and transfers data from the internal FIFO to shared memory.

5. At end of message, Interrupt Descriptors and Buffer Status Descriptors can be written out to shared memory (see 3)—depending on the masking of interrupts and allowance of Buffer Descriptor overwrites.
6. Read Next Message Descriptor.
7. Go to 3.

NOTE: MUSYCC responds to either a JUMP or ACTIVATE service request by reading first the head pointer and then the Message Descriptor pointed to by the Head Pointer. Therefore, software must set up the new head pointer and Message Descriptors (i.e., buffer list) before issuing either a JUMP or ACTIVATE service request.

6.3.17 Channel Deactivation

After the channel has been activated, channel deactivation via a service request suspends activity on an individual channel-direction by stopping that channel's processing of the current buffer list. The only indication that MUSYCC has completely stopped its list processing is a SACK interrupt. Therefore, the host must wait for a SACK before setting up the new buffer list.

Each channel within a channel group consists of a transmitter and receiver section. Each section is independent of the other and maintains its own state machine and configuration registers. To deactivate both transmitter and receiver sections, two separate service requests are required—one directed to the transmitter and one to the receiver. MUSYCC responds to each service request with the SACK Interrupt Descriptor, which notifies the host that the task was completed.

A channel deactivation is an asynchronous command from the host interface to a transmit or receive section of a channel to suspend bit-level processing and halt memory transfers into shared memory.

6.3.17.1 Transmit Channel Deactivation

The following describes what MUSYCC does when the transmit channel is deactivated:

1. Current message processing is terminated destructively; that is, data can be lost and messages prematurely aborted.
2. The bit-level processor responsible for handling outbound bits to the serial port is immediately and asynchronously disabled. The data output pin, TDAT, is three-stated or held at logic 1, depending on the bit field TRITX in the Port Configuration Descriptor. Data transfers from shared memory are halted.
3. The channel direction remains in the suspended state until the channel is activated. The current channel direction configuration is maintained.

6.3.17.2 Receive Channel Deactivation

The following describes what MUSYCC does when the receive channel is deactivated:

1. Current message processing is terminated destructively; that is, data can be lost and messages prematurely aborted.
2. The bit-level processor responsible for handling inbound bits from the serial port is immediately and asynchronously disabled. Data transfers to shared memory are halted.
3. The channel direction remains in the suspended state until the channel is activated. The current channel direction configuration is maintained.

6.3.18 Channel Jump

A channel jump request is issued by the host via a service request. For a receiver, channel jumps are the same as channel activation.

For a transmitter, channel jumps are non-destructive to currently serviced messages. The channel state is not reset as in the channel activate sequence. Therefore, a transmitter channel must be activated first, then subsequent jump requests can be made using the channel jump service request. For a transmitter, channel jumps provide a non-destructive way to start transmitting a new message list. MUSYCC waits until the completion of the current message before jumping to a message list pointed to by a new Head Pointer.

A jump request is issued by the host via a service request towards a channel in a channel group in MUSYCC.

NOTE: The service request acknowledge (via the SACK Interrupt Descriptor) for the jump service request—specifically for the transmit direction—will not be output towards the host until after the current message is transmitted. For applications with long messages to transmit, the jump service request must be used with care.

6.3.19 Frame Alignment

Each serial port frame consists of a fixed number of bits grouped into time slots according to the frame alignment supplied by the serial port TSYNC and RSYNC signals. MUSYCC must be provided at least one external synchronization pulse on the TSYNC and RSYNC input pins after the respective channel group is enabled. After this initial sync pulse, MUSYCC tracks subsequent serial port frame boundaries using its internal flywheel mechanism or the next applied sync pulse.

NOTE: Nx64 serial port mode does not operate the internal flywheel and therefore requires periodic TSYNC and RSYNC pulses to keep track of serial port frame boundaries.

In addition to tracking serial port frame boundaries, the internal flywheel generates a frame synchronization signal that can be selected to control the channel group's alignment of Transparent mode channel data streams. By default, the frame synchronization signal from the internal flywheel determines Transparent mode channel data stream alignment. If external data stream synchronization is preferred, the SFALIGN bit field in the Group Configuration Descriptor can be set to expect this synchronization signal to come from the serial port TSYNC and RSYNC input pins. While SFALIGN is set, the internal flywheel continues to operate, but the synchronization signal from the flywheel is not used to determine data stream alignment. Instead, alignment is provided by the external framer device which is allowed to strobe the sync input pins at periodic frame intervals or at any desired multiple of the frame interval (i.e., superframe).

Each serial port frame carries stream data from one or more packetized HDLC or unpacketized Transparent mode channels. Although channels are mapped to specific time slots within the serial port frame, each channel's data streaming process may or may not have a particular alignment with respect to that channel's assigned time slot boundaries, depending upon whether the channel is configured to operate in HDLC or Transparent mode.

For HDLC mode channels, data stream processing begins immediately upon channel activation. Any type of alignment of a HDLC channel's data stream with respect to its assigned serial port time slots is unnecessary, and MUSYCC disables time slot synchronization for that channel. Therefore, no specific alignment exists or needs to exist between the first bit of a HDLC message and the first bit of the assigned channel time slot.

After activation of a Transparent mode channel, the SFALIGN setting selects whether that channel's bit-level processor either waits for a frame synchronization signal from the internal flywheel, or an external synchronization signal from the serial port sync input pin before starting the data stream process. The selected synchronization signal (internal or external) thus determines the alignment of that channel's data stream with respect to its assigned serial port time slot. This time slot alignment mechanism ensures Transparent mode channel's are able to transfer sampled voice data streams to and from bytes stored in Shared Memory while maintaining the alignment of those bytes with respect to the serial port time slot. In Transparent mode MUSYCC is required to point to a MUSYCC-owned buffer prior to a channel activation service request.

For Transparent mode hyperchannels, where multiple time slots are mapped to a single channel, the first byte of data to and from the Shared Memory buffer is aligned to the lowest numbered serial port time slot mapped to that hyperchannel. If the lowest numbered time slot mapped to that hyperchannel equals time slot 12, the bit-level processor aligns the first byte of Shared Memory buffer data to time slot 12 and the next byte of data to the next higher numbered time slot that is also mapped to that hyperchannel. This sequence of time slot mapped alignment is true for all Transparent mode hyperchannel cases except when time slot 0 is the lowest numbered time slot mapped. In which case, the first byte of Shared Memory buffer data is transferred to the next higher numbered time slot. For example, a Transparent mode hyperchannel mapped to time slot 0, and time slot 1 would output the first byte of Shared Memory data during time slot 1 and would write receive data from time slot 1 into the first byte of the Shared Memory buffer.

6.3.20 Descriptor Polling

Upon channel activation and any necessary frame alignment, MUSYCC must fetch Message Descriptors from shared memory to start the flow of message bits into and out of shared memory.

As a Buffer Descriptor is fetched, MUSYCC checks the owner-bit to verify if the buffer is serviceable by MUSYCC. If the owner bit indicates that the host still owns the buffer, the host has not yet prepared the data in the buffer for processing. This may or may not be an error condition. In this case, MUSYCC also must check the no-poll bit in the same descriptors to determine if polling for MUSYCC ownership is enabled.

If the host owns the buffer and polling is disabled, the channel direction is suspended from processing messages until the host intervenes with a subsequent channel activation or channel jump request. The channel is not capable of leaving this suspended state autonomously.

If the host owns the buffer and polling is enabled, the channel direction is suspended from processing messages and MUSYCC periodically polls the owner bit in the Buffer Descriptor to verify that the buffer is ready for MUSYCC. The channel is capable of leaving this suspended state autonomously.

The frequency of polling is controlled independently for each channel group by the SFALIGN (superframe alignment) and POLLTH (poll throttle) bit fields in the Group Configuration Descriptor.

The SFALIGN bit field defines the source of the synchronization event to be used by MUSYCC. The source is either an internal flywheel method or an external signal at the serial port.

NOTE: Time slot counter or flywheel time base method uses a 7-bit counter. As each bit is serviced, over 32 channels with 8 bits per channel in a 2.048 MHz data stream, the counter is incremented. When the counter rolls over to 0, a Beginning of Frame is declared. At 2.048 MHz, 256 bits represents 125 ms.

The POLLTH bit field specifies how often MUSYCC checks the owner bit in a host-owned Buffer Descriptor. The values correspond to 1-, 16-, 32-, or 64-frame periods, or 125 µs, 2 ms, 4 ms, and 8 ms, respectively. The POLLTH bit field is always used in conjunction with the SFALIGN bit field. Table 6-11 lists the various polling frequencies and times.

If the serial port is configured for Nx64 mode (a variable number of 64 kbps channels assigned to form one logical channel), the Time Slot Counter is reset only when the inputs TSYNC or RSYNC are asserted. In this case, the SFALIGN bit field is always ignored.

Table 6-11. Polling Frequency Using a Time Slot Counter Method

Standard Channelized Input			Poll Throttle Value (Multiples of Frames)			
Name	**Rate (MHz)**	**Bits Per Frame**	**0 (x1)**	**1 (x16)**	**2 (x32)**	**3 (x64)**
T1	1.536	192	125 µs	2 ms	4 ms	8 ms
E1	2.048	256	125 µs	2 ms	4 ms	8 ms
2 E1	4.096	512	125 µs	2 ms	4 ms	8 ms
4 E1	8.192	1024	125 µs	2 ms	4 ms	8 ms

6.3.21 Repeat Message Transmission

MUSYCC provides a mechanism to repeatedly transmit a single message. A transmitter channel enters the repeat mode if the REPEAT bit field is 1, and the EOM bit field is 1 in a Transmit Buffer Descriptor.

NOTE: The message being repeatedly transmitted be specified completely by a single Message Descriptor and not by a linked list of descriptors.

A repeating message either fits entirely in the internal FIFO buffer space allocated for the transmitter channel, or the message is accessed in pieces over the PCI bus and then re-accessed from the beginning when the end of buffer is reached. The determination of whether the message fits entirely in the FIFO buffer or not is automatically performed each time MUSYCC enters repeat mode. MUSYCC compares the BLEN bit field (which specifies the number of bytes in the message) from the Transmit Buffer Descriptor to [(BUFFLEN + 1) x 2] (which specifies the number of dwords in the FIFO buffer) from the Channel Configuration Descriptor.

To exit repeat mode after the current message is completely transmitted and before the next repetition (gracefully or non-destructively), a channel jump service request must be issued. Prior to the jump request, the host must initialize the channel's Transmit Head Pointer with a new Message Descriptor.

To exit repeat mode, regardless of the message being processed, a channel activate or a channel deactivate service request can be issued. Either is considered destructive because the current message transmission is aborted.

6.4 Protocol Support

6.4.1 Frame Check Sequence

MUSYCC is configurable to calculate either a 16- or 32-bit Frame Check Sequence (FCS) for HDLC packets ranging in size from a minimum of 2 octets to a maximum of 16,384 octets. The FCS always applies to the entire packet length.

For all HDLC modes which require FCS calculations, the polynomials used to calculate the FCS are according to ITU-T Q.921 and ISO 3309-1984.

- CRC-16:

$$x^{16} + x^{12} + x^5 + 1$$

- CRC-32:

$$x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$$

6.4.2 Opening/Closing Flags

For HDLC modes only, MUSYCC supports the use of opening and closing message flags. The 7Eh (01111110b) flag is the opening and closing flag. An HDLC message is always bounded by this flag at the beginning and the end of the message.

MUSYCC supports receiving a shared flag where the closing flag of one message can act as the opening of the next message. MUSYCC also supports receiving a shared 0 bit between two flags, i.e., the last 0 bit of one flag is used as the first 0 bit of the next flag.

MUSYCC can be configured to transmit a shared flag between successive messages by configuring the bit field PADEN in each Transmit Buffer Descriptor. MUSYCC does not transmit shared 0 bits between successive flags.

6.4.3 Abort Codes

Seven consecutive 1s constitute an abort flag. Receiving the abort code causes the current frame processing to be aborted and terminates further data transfer into shared memory. After detecting the abort code, MUSYCC enters a mode searching for a new opening flag.

Notification of this detected condition is provided by the receive Buffer Status Descriptor or an Interrupt Descriptor, indicating the error condition Abort Flag Termination.

In cases where received idle codes transition to an abort code, an Interrupt Descriptor is generated toward the host indicating the informational event Change to Abort Code. All received abort codes are discarded.

6.4.4 Zero-Bit Insertion/Deletion

MUSYCC provides 0-bit insertion and deletion when it encounters five consecutive 1s within a frame. In the receiver, a 0 bit is de-inserted, and in the transmitter a 0 bit is inserted after five 1s are seen.

6.4.5 Message Configuration Bits

A group of bits specified in a Transmit Buffer Descriptor specifies the data to be transmitted at the transmit channel after the end of a current message has been transmitted. The bits are collectively known as the Message Configuration Descriptor and include the specifications for the following:

- Idle Code specification, IC
- Inter-message Pad Fill Enable, PADEN
- Inter-message Pad Fill Count, PADCNT
- Repeat Message Transmission, REPEAT

6.4.5.1 Idle Code

The Idle Code (IC) specification allows an idle code to be transmitted after the current message in case the next message is not available to be transmitted or inter-message pad fill is requested via PADEN.

6.4.5.2 Inter-message Pad Fill

The Pad Enable (PADEN) and Pad Count (PADCNT) specifications allow pad fill octets (a sequence of one or more specified idle codes) to be transmitted between messages. PADEN enables or disables the pad fill octet transmission feature. PADCNT is the minimum number of fill octets to be transmitted between the closing flag of one message and the opening flag of the next message.

6.4.5.3 Repeat Message Transmission

The Repeat Message Transmission (REPEAT) specification allows a single message to be transmitted repeatedly without additional host intervention. This feature is required to support SS7 message retransmission. In the SS7 application, a retransmitted message is usually 3, 4, or 5 octets in length. Message retransmission in MUSYCC requires that the entire message be held in a single shared memory message buffer, versus being spread across multiple buffers. The message retransmission feature is not limited to SS7 applications.

6.4.6 Message Configuration Bits Copy Enable/Disable

The message configuration bits described above are used in special data communication applications requiring the following attributes specified on a per-message basis:

- Specific idle code is to be transmitted between messages.
- Inter-message pad fill is required.
- Repeat message transmission is required.

If at least one transmit channel in a channel group is supporting an application which requires any one of the above attributes, the Message Configuration Bits Copy must be enabled for the entire channel group by setting the MCENBL bit field to 1.

Setting MCENBL to 0 prevents MUSYCC from copying the Message Configuration Bits from the Transmit Buffer Descriptor and has the following effect on transmit channel operations throughout the channel group:

- PADEN and PADCNT are set to 0, thereby facilitating back-to-back message transmission.
- REPEAT is set to 0, thereby disabling automatic message retransmission.

Any of the Message Configuration Bits can be changed by writing new values for these bit fields directly into a channel group's Transmit Configuration Table. However, the exact time the new bit field values are applied by the transmitter is unrelated to the message being serviced. If the channel is idle, any change to the message configuration bits in the Configuration Table will apply to subsequent messages.

To write new configuration bits into the Transmit Message Configuration Table, a PCI dword operation is required. Tables 6-12 and 6-13 list the address map and the Message Configuration Descriptor layouts.

Table 6-12. Memory Map for Message Configuration Descriptor Table

Channel Number (0–31)	Location of Message Configuration Descriptor in Specific Group (0–3) (Byte offset from Base Address Register)							
	0	1	2	3	4	5	6	7
0	06180h	06980h	07180h	07980h	16180h	16980h	17180h	17980h
1	06184h	06984h	07184h	07984h	16184h	16984h	17184h	17984h
x	Address of Message Configuration Descriptor for a channel in a group 06180h + (Group_Number[0:3]•00800h) + (Channel_Number[31:0]•00008h)				Address of Message Configuration Descriptor for a channel in a group 16180h + (Group_Number[4:7]•00800h) + (Channel_Number[31:0]•00008h)			
31	061FCh	069FCh	071FCh	079FCh	161FCh	169FCh	171FCh	179FCh

Table 6-13. Message Configuration Descriptor *(1 of 2)*

Bit Field	Name	Value	Description
31:27	RSVD	0	Reserved.
26:25	IC	0	Idle Code Select –7Eh (0111 1110b)
		1	Idle Code Select –FFh (1111 1111b)
		2	Idle Code Select –00h (0000 0000b)
		3	Idle Code Select –Reserved.

Table 6-13. Message Configuration Descriptor *(2 of 2)*

Bit Field	Name	Value	Description
24	PADEN	0	Pad Fill Disabled. One shared opening/closing flag (7Eh) is inserted before sending next message
		1	Pad Fill Enabled. Also, see PADCNT bit field.
23:16	PADCNT[7:0]	0	Pad Count. When PADEN = 1, PADCNT indicates the minimum number of idle codes to be inserted between the closing flags and the next opening flag (7Eh). If PADCNT = 2 and IC = 1, for example, yields the bit pattern 7Eh..FFh..FFh..7Eh There is no indication by MUSYCC if more than PADCNT number of idle codes are inserted.
15	REPEAT	0	Repeat Message Transmission Disabled.
		1	Repeat Message Transmission Enabled.
14:0	RSVD	0	Reserved.

6.4.7 Bit-Level Operation

Each channel group provides two separate Bit-Level Processors (BLP) to service the transmit and receive directions separately. Also, each channel group provides two separate Direct Memory Access Controllers (DMAC) to service the transmit and receive directions separately. The BLP and DMAC work in conjunction to transfer serial data between the serial interface and shared memory. BLPs perform the required bit-level processing based on the protocol mode assigned to the channel and direction.

DMACs access the data to transmit or store the received data to shared memory (via the host interface). Each DMAC seeks out the next Message Descriptor to service for each active channel and direction. This information is available from the Channel Group Descriptor in shared memory which has information locating the message list for each channel direction. A Buffer Descriptor within each Message Descriptor along with the protocol mode set for the channel-direction drives the treatment of the receive and transmit bit stream.

Bit-level operations vary between HDLC and transparent modes. The differences relate to protocol-specific support, as well as to the treatment of the bit stream during abnormal conditions. Additionally, bit-level operations are independent and sometimes differ between the transmitter and the receiver.

The following items apply to all event and error handling as described in the transmit and receive sections which follow:

- During bit level operations, events and errors can affect the outcome message processing. Unless masked, all events and errors generate Interrupt Descriptors within MUSYCC. Interrupt Descriptors identify the error or event condition, the transmit or receive direction, and the channel and channel group number affected.
- If a channel is suspended, enters an idle mode, enters an abort mode, or is autonomously turned off by MUSYCC during bit-level operations, a channel reactivation must occur by either a Channel Activation Service Request or a Channel Jump Service Request. This is referred to as "requiring reactivation."
- The bit fields INHTBSD and INHRBSD in the Group Configuration Descriptor specify whether or not MUSYCC can write a Buffer Status Descriptor into a Message Descriptor to indicate that MUSYCC has completed servicing the descriptor.
- In cases where bit-level operations continue normally, the DMAC accesses the Next Message Pointer from the current Message Descriptor. This accesses the next Message Descriptor in the chain of descriptors for a particular channel and direction. Each Message Descriptor indicates whether or not the host or MUSYCC owns the descriptor.

6.4.7.1 Transmit

The transmitter initiates data transfer from shared memory to the serial interface only if the following conditions are true:

- TXENBL bit is set to 1 in the Group Configuration Descriptor.
- Transmit channel is mapped to logical channel(s) in Transmit Time Slot Map.
- Transmit channel is (re)activated (via service request).

If the TXENBL bit is set to 0, the output signal is a three-state signal. If the channel is not mapped or is inactive, the transmitter either outputs a three-state or an all 1s signal depending on the state of the bit field TRITX in the Group Configuration Descriptor.

6.4.7.2 Receive

The receiver transfers data from the serial interface to memory buffers in shared memory only if all the following conditions are true:

- RXENBL bit is set to 1 in the Group Configuration Descriptor.
- Receive channel is mapped to one or more logical channel(s) in Receive Time Slot Map.
- Receive channel is (re)activated (via service request).

If any one of the above conditions is not true, the receiver ignores the incoming data stream.

Data transfer consists of MUSYCC first seeking out the next Message Descriptor from the Channel Group Descriptor in shared memory for each active channel. The Buffer Descriptor in each Message Descriptor plus the protocol mode set for the channel dictates the treatment of the incoming bit stream.

6.4.8 HDLC Mode

MUSYCC supports three HDLC modes. The modes are assigned on a per-channel and direction basis by setting the PROTOCOL bit field within the Channel Configuration Descriptor. The HDLC modes are as follows:

- SS7-HDLC-16CRC: specific SS7 support, HDLC support, 16-bit CRC.
- HDLC-16CRC: HDLC support, 16-bit CRC.
- HDLC-32CRC: HDLC support, 32-bit CRC.

HDLC support by the transmitter includes the following:

- Generating opening, closing, and shared flags.
- 0-bit insertion after five consecutive 1s are transmitted.
- Generating pad fill between frames and adjust for 0 insertions.
- Generating 16- or 32-bit FCS.
- Generating abort sequences upon data corruption in message.

HDLC support by the receiver includes the following:

- Detection and extraction of opening, closing, and shared flags.
- Detection of shared 0 between successive flags.
- 0-bit extraction after five consecutive 1s are received.
- Detecting changes in pad fill idle codes.
- Checking and extracting 16- or 32-bit FCS.
- Checking frame length.
- Checking for octet alignment.
- Checking for abort sequence reception.

Bit Fields within the Transmit Buffer Descriptor specify inter-message bit level operations. Specifically, when the EOM bit field is set to 1 within a Message Descriptor by the host, it signifies that the descriptor represents the last buffer for the current message being transmitted and the bit fields IC, PADEN, PADCNT, and REPEAT take effect. These bits are collectively known as Message Configuration Descriptor.

Additionally, the bit field NP in both the Receive and Transmit Buffer Descriptors enables a polling scheme in case MUSYCC discovers that it does not own the (next) Message Descriptor.

6.4.8.1 Transmit Events

Transmit events are informational in nature and do not require channel recovery actions.

End of Buffer (EOB)

Reason:

- DMAC reached the end of a buffer by servicing a number of octets equal to the bit field BLEN in the Transmit Buffer Descriptor. The last EOB and an EOM are coincident and result in two separate events being generated.

Effects:

- Interrupt Descriptor in Interrupt Queue with EVENT = EOB, DIR = 1 (if EOBI = 1 in Transmit Buffer Descriptor).
- BLP and DMAC continue with normal message processing. If the DMAC does not receive more data from shared memory before the BLP must output the next data bit, the BLP outputs another octet of idle code.

End of Message (EOM] Reason:

- BLP has transmitted the last bit of a data buffer and the Transmit Buffer Descriptor signified the end of a message by the bit field EOM in a Transmit Buffer Descriptor. The last EOB and an EOM are coincident and result in two separate events being generated.

Effects:

- Interrupt Descriptor in Interrupt Queue with EVENT = EOM, DIR = 1 (if MSKEOM = 0 in Transmit Channel Configuration Descriptor).
- BLP and DMAC continue with normal message processing. If the DMAC does not receive more data from shared memory before the BLP must output the next data bit, the BLP outputs another octet of idle code.

End of Padfill (EOP] Reason:

- BLP has transmitted the specified number of pad fill octets.

Effects:

- Interrupt Descriptor in Interrupt Queue with EVENT = EOP, DIR = 1 (if EOPI = 1 in Transmit Channel Configuration Descriptor).
- BLP and DMAC continue with normal message processing. If the DMAC does not receive more data from shared memory before the BLP must output the next data bit, the BLP outputs another octet of idle code.

6.4.8.2 Receive Events

Receive events are informational in nature and do not require channel recovery actions.

End of Buffer (EOB) Reason:

- One message is stored across multiple message buffers. MUSYCC reached the end of a buffer by servicing a number of octets equal to the bit field BLEN in a Receive Buffer Descriptor.

Effects:

- The Interrupt Descriptor in the Interrupt Queue with EVENT = EOB, DIR = 0
  (if EOBI = 1 in Receive Buffer Descriptor).
- BLP and DMAC continue with normal message processing.

End of Message (EOM) Reason:

- BLP detected the end of a message (closing flag or an error condition) in the received data stream. Error conditions include ABT, LNG, ALIGN, BUFF, and ONR errors.

Effects:

- Interrupt Descriptor in Interrupt Queue with EVENT = EOM, DIR = 0 (if MSKEOM = 0 in Receive Channel Configuration Descriptor).
- DMAC sets bit field EOM = 1 in Receive Buffer Status Descriptor (if INHRBSD = 0 in Receive Channel Configuration Descriptor).
- BLP and DMAC continue with normal message processing.

Change to Abort Code (CHABT)

Reason:

- BLP detected received data changed from pad fill (7Eh) octets to abort code (zero followed by seven consecutive 1s).

Effects:

- Interrupt Descriptor in Interrupt Queue with EVENT = CHABT, DIR = 0 (if MSKIDLE = 0 in Receive Channel Configuration Descriptor).
- BLP and DMAC continue with normal message processing.

Change to Idle Code (CHIC)

Reason:

- BLP detected received data changed from abort code (0 followed by seven 1s) to idle code (7Eh) octets.

Effects:

- Interrupt Descriptor in Interrupt Queue with EVENT = CHIC, DIR = 0 (if MSKIDLE = 0 in Receive Channel Configuration Descriptor).
- BLP and DMAC continue with normal message processing.

Frame Recovery (FREC)

Reason:

- BLP detected that the serial interface has transitioned from an out-of-frame to in-frame condition.

Effects:

- Interrupt Descriptor in Interrupt Queue with EVENT = FREC, DIR = 0 (if MSKOOF = 0 in Group Configuration Descriptor).
- BLP and DMAC continue with normal message processing.

SS7 SUERM Octet Count Increment (SINC)

Reasons:

- BLP incremented the SUERM counter. The channel is in SS7 mode. The reasons for SUERM counter to increment include reception of a short message, octet alignment error, FCS mismatch, or an accumulation of octet count errors. Each of these conditions may also generate an interrupt.

Effects:

- Interrupt Descriptor in Interrupt Queue with EVENT = SINC, DIR = 0 (if MSKSINC = 0 in Receive Channel Configuration Descriptor).
- BLP and DMAC continue with normal message processing.

SS7 SUERM Octet Count Decrement (SDEC)

Reasons:

- BLP decremented the SUERM counter. The channel is in SS7 mode. The SUERM counter decrements by one when MUSYCC receives 256 consecutive unerrored messages.

Effects:

- Interrupt Descriptor in Interrupt Queue with EVENT = SDEC, DIR = 0 (if MSKSDEC = 0 in Receive Channel Configuration Descriptor).
- BLP and DMAC continue with normal message processing.

SS7 Filtered Message (SFILT)

Reason:

- BLP detected an unerrored 3, 4, or 5 octet message identical to the previous message. The channel is in SS7 mode.

Effects:

- Interrupt Descriptor in Interrupt Queue with EVENT = SFILT, DIR = 0 (if MSKSFILT = 0 in Receive Channel Configuration Descriptor).
- BLP discards the received message in the FIFO.
- BLP and DMAC continue with normal message processing.

6.4.8.3 Transmit Errors

Transmit errors are service-affecting and require a corrective action by a controlling device to resume normal bit-level processing.

Underflow Due to Host Ownership of Buffer (ONR)

In this case, MUSYCC attempts to access the [next] Message Descriptor when the prior descriptor contained only a portion of the message (EOM = 0), and MUSYCC finds that ownership of the [next] descriptor has not been granted by the host (i.e., the next buffer is host-owned).

This error results when currently transmitting an HDLC message, and no additional descriptors are available in a timely manner.

Once a descriptor is granted, however, MUSYCC assumes ownership of the message buffer and continues reading data until the end of buffer is reached. If the host reclaims the buffer without MUSYCC granting ownership back to the host, a host error occurs, and the effects are indeterminate.

Reason:

- Degradation of the host subsystem or application software performance.

Effects:

- Partial HDLC message transmission has occurred.
- Interrupt Descriptor in Interrupt Queue with ERROR = ONR, DIR = 1 (if MSKBUFF = 0 in Transmit Channel Configuration Descriptor).
- Transmit channel enters abort state where the BLP transmits a repetitive abort sequence of 16 consecutive 1s.
- Transmit Buffer Status Descriptor cannot be written.
- Message polling is automatically disabled.
- Transmit channel enters abort state.

Channel Level Recovery Actions:

- Transmit channel reactivation is required.

Underflow Due to Internal FIFO Buffer Under-Run (BUFF)

In the case of underflow due to internal FIFO buffer under-run, the internal FIFO buffer becomes empty when MUSYCC transmits data bits (at the serial interface clock rate), and MUSYCC has ownership of a message buffer in shared memory.

Reasons:

- Degradation of the host subsystem or application software performance.
- Congestion of the PCI bus.

Effects:

- Interrupt Descriptor in Interrupt Queue with ERROR = BUFF, DIR = 1 (if MSKBUFF = 0 in Transmit Channel Configuration Descriptor).
- Transmit channel enters abort state where the BLP transmits a repetitive abort sequence of 16 consecutive 1s.
- Message polling is automatically disabled.
- Transmit Buffer Status Descriptor is not written.

Channel Level Recovery Actions:

- Transmit channel reactivation is required.

Change of Frame Alignment (COFA) while Transmitting HDLC Message (T1/E1 modes)

In the case of change of frame alignment while transmitting an HDLC message (T1/E1 modes), the TSYNC input signal transitions from low to high when not expected to do so by the frame synchronization flywheel mechanism. This error only applies to ports configured for T1, E1, 2xE1 or 4xE1 signals. Frame synchronization indicates the location of time slot 0 in the serial data stream. Lacking frame synchronization, the transmitter cannot map and align time slots. This error affects all active channels in the channel group.

Reason:

- T1/E1 signal failure is detected by the physical interface providing the serial data, clock frequency, and synchronization to the serial interface on MUSYCC.

Effects:

- Causes serial interface to enter COFA mode for one T1/E1 frame period (125 µs)—not necessarily on a frame boundary.
- For every activated channel transmitting an HDLC message, the Transmit channel enters an abort state where the BLP transmits a repetitive abort sequence of 16 consecutive 1s.
- MUSYCC does not update the transmit Message Descriptor and does not generate an EOB/EOM unless the message is already sent or the buffer is already processed.
- MUSYCC stops polling any active transmit channels descriptor.
- After the COFA condition subsides, the channel is deactivated.

Channel Level Recovery Actions:

- Transmit channel reactivation is required.

6.4.8.4 Receive Errors

Receive errors are service-affecting and require a corrective action by the host to resume normal bit-level processing.

Overflow Due to Host Ownership of Buffer while Receiving HDLC Message (ONR)

In the case of overflow due to host ownership of the buffer while receiving an HDLC message, MUSYCC attempts to access the next Message Descriptor to store a message or part of a message, and finds that ownership of the descriptor has not been granted by the host.

This error results when currently receiving an HDLC message, and no additional descriptors are available in a timely manner.

Once a descriptor is granted, however, MUSYCC assumes ownership of the message buffer and continues writing data until the end of buffer is reached. If the host reclaims the buffer without MUSYCC granting ownership back to the host, a host error occurs and the effects are indeterminate.

Reason:

- Degradation of host subsystem or application software performance.

Effects:

- Interrupt Descriptor in Interrupt Queue with ERROR = ONR, DIR = 0 (if MSKBUFF = 0 in Receive Channel Configuration Descriptor).
- The received data in the internal FIFO buffer is discarded and lost to the host.
- The remainder of the HDLC message currently being received is discarded.
- The Receive Buffer Status Descriptor cannot be written.
- The channel is deactivated.

Channel Level Recovery Actions:

- Provide sufficient amount of shared memory to store received data using the lists of Message Descriptors with ownership granted to MUSYCC.
- Reactivate channel.

Overflow Due to Internal FIFO Buffer Overrun (BUFF)

In the case of overflow due to internal FIFO buffer overrun, the internal FIFO buffer has not been completely copied to shared memory before more data bits arrive needing to be stored in the FIFO buffer. MUSYCC has access to a message buffer space in shared memory in this case.

Reasons:

- Degradation of host sub system performance.
- Congestion of the PCI bus.

Effects:

- The Interrupt Descriptor in Interrupt Queue with ERROR = BUFF, DIR = 0 (if MSKBUFF = 0 in Receive Channel Configuration Descriptor).
- The received data in the internal FIFO buffer is discarded and lost to the host.
- The remainder of HDLC message currently being received is discarded.
- Access the Next Message Pointer from the Current Message Descriptor.
- Return ownership of current Message Descriptor by writing the Receive Buffer Status Descriptor with ONR = HOST, ERROR = BUFF (if INHRBSD = 0 in Receive Channel Configuration Descriptor).
- BLP scans for the opening flag of the next HDLC message.
- Simultaneously, DMAC checks for Message Descriptor ownership before transferring received data to shared memory.

Channel Level Recovery Actions:

- If possible, increase internal FIFO buffer space for this channel. For this action, the channel must be deactivated first.
- If required, alleviate congestion of the PCI bus.
- Change of Frame Alignment (COFA) while Receiving HDLC Message (T1/E1 modes)

In the case of a Change of Frame Alignment while receiving an HDLC message (T1/E1 modes), the RSYNC input signal transitions from low to high unexpectedly by the "frame synchronization flywheel mechanism." This error applies only to ports configured for T1, E1, 2xE1, or 4xE1 signals. Frame synchronization indicates the location of time slot 0 in the serial data stream. Lacking frame synchronization, the received channelized data becomes unaligned and unmappable. This error affects all active channels in the channel group.

Reason:

- T1/E1 signal failure is detected by the physical interface providing the serial data, clock frequency, and synchronization to the serial interface on MUSYCC.

Effects:

- Causes the serial interface to enter the COFA mode for one T1/E1 frame period (125 µs).
- For each activated channel receiving an HDLC message, the remainder of the HDLC message currently being received is discarded, and the receiver scans for the opening flag of the next HDLC message before attempting to fill the channel's FIFO buffer again.
- For each activated channel receiving an HDLC message, the ownership of the current Message Descriptor is granted back to the host by writing the Receive Buffer Status Descriptor with ONR = HOST and ERROR = COFA (if INHRBSD = 0 in Receive Channel Configuration Descriptor).
- After all activated channels are serviced, MUSYCC writes the Interrupt Descriptor in Interrupt Queue with ERROR = COFA. DIR = 0 (if MSKCOFA = 0 in Group Configuration Descriptor).
- After the COFA condition clears, normal bit-level operations continue.
- The BLP scans for the opening flag of the next HDLC message.
- Simultaneously, DMAC checks for Message Descriptor ownership before transferring received data to shared memory.

Channel Level Recovery Actions:

- None required.

Out of Frame (OOF)

Out-of-frame or loss-of-frame indicates that the entire serial data stream is invalid and data cannot be recovered from such a signal. In this case, out-of-frame of the incoming signal occurred while in the midst of receiving an HDLC message and copying the data to shared memory.

Reason:

- MUSYCC writes the T1/E1 signal failure is detected by the physical interface providing the serial data, clock frequency, and synchronization to the serial interface on MUSYCC.

Effects:

- MUSYCC writes the Interrupt Descriptor in Interrupt Queue with ERROR = OOF, DIR = 0 (if MSKOOF = 0 in Group Configuration Descriptor).
- If bit field OOFABT = 0, BLP, and DMAC continue as if no errors occurred and transfer received data into shared memory buffers normally.
- If bit field OOFABT = 1 and is currently receiving an HDLC message, the received data in the internal FIFO buffer is discarded and lost to the host. DMAC accesses the Next Message Pointer from the current Message Descriptor. MUSYCC returns ownership of the current Message Descriptor by writing the Receive Buffer Status Descriptor with ONR = HOST, ERROR = OOF (if INHRBSD = 0 in Receive Channel Configuration Descriptor).
- Regardless, the BLP continues scanning for opening flag.
- Simultaneously, DMAC checks for Message Descriptor ownership before transferring received data to shared memory.
- Receive channel recovers automatically.

Channel Level Recovery Actions:

- None required.

Frame Check Sequence (FCS) Error

In the case of an FCS error, the frame check sequence (or CRC) calculated for the received HDLC message by MUSYCC does not match the FCS sent within the HDLC message.

Reason:

- Bit errors during transmission.

Effects:

- The Interrupt Descriptor in Interrupt Queue with ERROR = FCS, DIR = 0 (if MSKMSG = 0 in Channel Configuration Descriptor).
- The entire HDLC message already copied to shared memory buffers.
- DMAC accesses the Next Message Pointer from the current Message Descriptor.
- Returns ownership of the current Message Descriptor to the host by writing the Receive Buffer Status Descriptor with ONR = HOST, ERROR = FCS (if INHRBSD = 0 in Channel Configuration Descriptor).
- The BLP scans for the opening flag of the next HDLC message.
- Simultaneously, DMAC checks for Message Descriptor ownership before transferring received data to shared memory.

Channel Level Recovery Actions:

- None required.

Octet Alignment (ALIGN) Error

In the case of an Octet Alignment (ALIGN) error, the HDLC message size after 0-bit extraction is not a multiple of 8 bits.

Reasons:

- Bit errors during transmission.
- Incorrect transmission of HDLC messages from the distant end.

Effects:

- The Interrupt Descriptor in Interrupt Queue with ERROR = ALIGN, DIR = 0 (if MSKMSG = 0 in Receive Channel Configuration Descriptor).
- The entire HDLC message is transferred to shared memory.
- DMAC accesses the Next Message Pointer from the current Message Descriptor.
- Returns ownership of the current Message Descriptor to the host by writing the Receive Buffer Status Descriptor with ONR = HOST, ERROR = ALIGN (if INHRBSD = 0 in Receive Channel Configuration Descriptor).
- The BLP scans for the opening flag of the next HDLC message.
- Simultaneously, DMAC checks for Message Descriptor ownership before transferring received data to shared memory.

Channel Level Recovery Actions:

- None required.

Abort Termination (ABT)

In the case of an Abort Termination (ABT) error, the receiver detects an abort sequence from the distant end. An abort sequence is defined as 0 followed by 7 consecutive 1s.

Reason:

- The distant end can not complete transmission of HDLC message.

Effects:

- The Interrupt Descriptor in Interrupt Queue with ERROR = ABT, DIR = 0 (if MSKMSG = 0 in Receive Channel Configuration Descriptor).
- Partial HDLC message is transferred to shared memory.
- DMAC accesses the Next Message Pointer from the current Message Descriptor.
- Returns ownership of the current Message Descriptor to the host by writing the Receive Buffer Status Descriptor with ONR = HOST, ERROR = ABT (if INHRBSD = 0 in Receive Channel Configuration Descriptor).
- The BLP scans for the opening flag of the next HDLC message.
- Simultaneously, DMAC checks for Message Descriptor ownership before transferring received data to shared memory.

Channel Level Recovery Actions:

- None required.

Long Message (LNG)

In the case of a Long Message (LNG) error, the received HDLC message size is determined to be greater than the maximum allowed message size (per MAXSEL in Channel Configuration Descriptor).

Reason:

- Incorrect transmission of HDLC messages from the distant end.

Effects:

- The Interrupt Descriptor in Interrupt Queue is ERROR = LNG, DIR = 0 (if MSKMSG = 0 in Receive Channel Configuration Descriptor).
- DMAC accesses the Next Message Pointer from the current Message Descriptor.
- Returns ownership of the current Message Descriptor to the host by writing the Receive Buffer Status Descriptor with ONR = HOST, ERROR = LNG (if INHRBSD = 0 in Receive Channel Configuration Descriptor).
- The BLP scans for the opening flag of the next HDLC message.
- Simultaneously, DMAC checks for Message Descriptor ownership before transferring received data to shared memory.

Channel Level Recovery Actions:

- None required.

Short Message (SHT)

In the case of a Short Message (SHT), the total received HDLC message size (including FCS) is less than the number of FCS bits specified for the receive channel. In other words, for a channel configured for 16-bit FCS, a minimum of an 8-bit payload must be received to avoid a short message error. For this example, three octets must be received—one octet for payload and two for FCS. Receiving two octets would be considered a short message.

Reasons:

- Bit errors during transmission.
- Incorrect transmission of HDLC messages from the distant end.

Effects:

- The Interrupt Descriptor in Interrupt Queue is ERROR = SHT, DIR = 0 (if MSKIDLE = 0 in Receive Channel Configuration Descriptor).
- Maintains ownership of current Message Descriptor.
- The BLP resumes scanning for opening flag of the next HDLC message.
- Simultaneously, MUSYCC checks for Message Descriptor ownership before proceeding with bit-level operations.

Channel Level Recovery Actions:

- None required.

SS7 Signal Unit Error Rate (SUERR) Interrupt

In the case of an SS7 SUERR, an error is detected in SS7 mode which caused a counter for SS7 related errors to equal or exceed the permitted threshold value. The threshold is stored on a per-channel group basis in the bit field SUET in a Group Configuration Descriptor.

Reasons:

- A Short SS7 message increments the SS7 counter.
- An FCS error in the SS7 message increments the SS7 counter.
- An Octet alignment error in the SS7 message.
- Accumulation of 16 “octet count” type errors increments counter.

NOTE: Receiving 256 unerrored SS7 messages decrements the SS7counter.

Effects:

- The Interrupt Descriptor in Interrupt Queue with ERROR = SUERR, DIR = 0 (if MSKSUERR = 0 in Receive Channel Configuration Descriptor).
- The BLP scans for the opening flag of the next HDLC message.
- Simultaneously, DMAC checks for Message Descriptor ownership before transferring received data to shared memory.

Channel Level Recovery Actions:

- None required.

6.4.9 Transparent Mode

MUSYCC supports a transparent mode where no distinction is made between information and non-information bits in the data bit stream. This mode is assigned on a per channel and direction basis by the bit field PROTOCOL in the Channel Configuration Descriptor.

In transparent mode, the following characteristics apply:

- All data bits are transferred between shared memory and the serial interface without protocol support such as those listed for the HDLC mode.
- Host must maintain the necessary data transfer rates at all times by providing Message Descriptors and data buffers for both the transmit and receive channels.
- The host must always set the bit field EOM to 0 in each Transmit Buffer Descriptor. Setting EOM to 1 causes indeterminate results. Due to EOM = 0, the other Transmit Buffer Descriptor bit fields—IC, PADEN, PADCNT, and REPEAT—are ineffective.

Unlike HDLC mode, MUSYCC does not poll a host-owned Transmit Buffer Descriptor during transparent mode.

When the internal buffer is empty and no more transmit data is available from shared memory (i.e., host-owned buffer), MUSYCC does the following:

1. Issues an ONR error.
2. Enters the channel deactivate state, sending idle code on the affected channel.

If the host wants to send any more data on that channel, the host must reactivate any transparent mode transmit channel that has issued an ONR error.

Notice there is no mechanism for transparent mode channels to ever enter the IDLE transmission state.

6.4.9.1 Transmit Events

Transmit events are informational and require no recovery actions.

End of Buffer (EOB)

Reason:

- DMAC reached the end of a buffer by servicing a number of octets equal to the BLEN bit field in a Transmit Buffer Descriptor. For the transparent mode, EOM is not a valid event because there is no concept of messages.

Effects:

- MUSYCC writes the Interrupt Descriptor in Interrupt Queue with EVENT = EOB, DIR = 1 (per EOBI in Transmit Buffer Descriptor).
- MUSYCC continues with normal transparent mode processing by processing to the next message structure.
- MUSYCC requires more data buffers.

6.4.9.2 Receive Events

Receive events are informational and require no recovery actions.

End of Buffer (EOB)

Reason:

- BLP reached the end of a buffer by transferring into shared memory a number of octets equal to the BLEN bit field in a Receive Buffer Descriptor.

Effects:

- MUSYCC writes the Interrupt Descriptor in Interrupt Queue with EVENT = EOB, DIR = 0 (per EOBI in Receive Buffer Descriptor).

Frame Recovery (FREC)

Reason:

- BLP detects that the serial interface transitions from an out-of-frame to an in-frame condition.

Effects:

- The Interrupt Descriptor in Interrupt Queue with EVENT = FREC, DIR = 0 (per MSKOOF in Group Configuration Descriptor).
- MUSYCC continues with normal transparent mode processing by processing to the next message structure.

6.4.9.3 Transmit Errors

Transmit Errors are service-affecting and require a corrective action by the host to resume normal bit-level processing.

Underflow Due to Host Ownership of Buffer (ONR)

In the case of underflow due to host ownership of buffer (ONR), sufficient data throughput from shared memory is not maintained to support the data rate of the serial interface. That is, ownership of messages was not handed over to MUSYCC in a timely manner.

Reason:

- Degradation of the host subsystem or application software performance.

Effects:

- The Interrupt Descriptor in Interrupt Queue with ERROR = ONR, DIR = 1.
- Data from Data Buffer not being read.
- The Buffer Descriptor not overwriting the Buffer Status Descriptor.
- No additional activity on the PCI bus for this channel direction.
- Transmit channel activity is suspended.

Channel Level Recovery Actions:

- Provide sufficient amount of data for transmission using lists of Message Descriptors with ownership given to MUSYCC.
- Reactivate transmit channel.

Underflow Due to Internal FIFO Buffer Under-Run (BUFF)

In the case of underflow due to internal FIFO buffer under-run (BUFF), the internal FIFO buffer becomes empty when MUSYCC must output data bits (at the serial interface clock rate) and MUSYCC has ownership of a message buffer in shared memory.

Reasons:

- Degradation of the host subsystem or application software performance.
- Congestion of the PCI bus.

Effects:

- MUSYCC writes the Interrupt Descriptor in Interrupt Queue with ERROR = BUFF, DIR = 1.
- Continuous idle code transmission.
- Data from data buffer not being read.
- The Buffer Descriptor not overwriting the Buffer Status Descriptor.
- No additional activity on the PCI bus for this channel direction.
- Transmit channel activity is suspended.

Channel Level Recovery Actions:

- Provide sufficient amount of data for transmission using lists of Message Descriptors with ownership given to MUSYCC.
- Reactivate transmit channel.

6.4.9.4 Receive Errors

Receive errors are service-affecting and may require a corrective action by a controlling device to resume normal bit-level processing.

Overflow Due to Host Ownership of the Buffer (ONR)

In the case of overflow due to host ownership of the buffer, the host has not provided sufficient data buffer space to store received data from the serial interface, and the internal FIFO buffer overflows with received data bits.

Reasons:

- Degradation of the host subsystem or application software performance.
- Congestion of the PCI bus.

Effects:

- The Interrupt Descriptor in Interrupt Queue with ERROR = ONR, DIR = 0.
- The received data in the internal FIFO buffer is discarded and lost to the host.
- The channel is deactivated.

Channel Level Recovery Actions:

- Reactivate the channel.

Overflow Due to Internal FIFO Buffer Overrun (BUFF)

In the case of overflow due to internal FIFO buffer overrun, the internal FIFO buffer is not completely copied to shared memory before more received data bits must be stored in the FIFO buffer. MUSYCC has access to a shared memory buffer in this case.

Reasons:

- Degradation of the host subsystem or application software performance.
- Congestion of the PCI bus.

Effects:

- MUSYCC writes the Interrupt Descriptor in Interrupt Queue with ERROR = BUFF, DIR = 0.
- The received data in the internal FIFO is discarded and lost to the host.
- No additional activity on the PCI bus for this channel direction.
- Receive channel activity is suspended.

Channel Level Recovery Actions:

- If possible, increase internal FIFO buffer space for this channel.
- Alleviate loading of the PCI bus.
- Reactivate receive channel with a channel activate or channel jump service request or with a slave write into the Receive Channel Configuration Table.

Change of Frame Alignment (T1/E1 Modes) (COFA)

In the case of a Change of Frame Alignment while receiving an HDLC message (T1/E1 modes), the RSYNC input signal transitions from low to high unexpectedly by the "frame synchronization flywheel mechanism." This error applies only to ports configured for T1, E1, 2xE1, or 4xE1 signals. Frame synchronization indicates the location of time slot 0 in the serial data stream. Lacking frame synchronization, the received channelized data becomes unaligned and unmappable. This error affects all active channels in the channel group.

Reason:

- A signal failure detected by the physical interface providing the serial data, clock frequency, and synchronization to the serial interface on MUSYCC.

Effects:

- MUSYCC writes the Interrupt Descriptor in Interrupt Queue with ERROR = COFA, DIR = 0.
- If OOFABT bit field is set to 0 in the Group Configuration Descriptor, then continue channel activity. That is, received data bits are sampled and eventually copied into shared memory.
- If bit field OOFABT is set to 1 in the Group Configuration Descriptor, suspend channel activity. Received data bits are discarded and lost to the host processor.

Channel Level Recovery Actions:

- If OOFABT = 1, once the received signal has recovered, reactivate receive channel with a Channel Activate or Channel Jump Service Request or with a slave write into the Receive Channel Configuration Table.

Out of Frame (OOF)

Out-of-frame or loss-of-frame indicates the entire serial data stream is invalid, and data cannot be recovered from such a signal.

Reasons:

- A signal failure detected by the physical interface providing the serial data, clock frequency, and synchronization to the serial interface on MUSYCC.

Effects:

- MUSYCC writes the Interrupt Descriptor in Interrupt Queue with ERROR = OOF, DIR = 0.
- The received data in the internal FIFO buffer is discarded and lost to the host.
- If OOFABT bit field is set to 0 in the Group Configuration Descriptor, continue channel activity but transfer all 1s data into shared memory for the duration of the OOF. When the OOF condition clears, normal bit-level processing resumes automatically without host intervention.
- If OOFABT bit field is set to 1 in the Group Configuration Descriptor, channel activity suspends without transferring any data to shared memory.

Channel Level Recovery Actions:

- When OOFABT selects receive message processing disabled, the host must reactivate the receive channel after the OOF condition has cleared. Reactivation is not required if OOFABT allows receive message processing to continue.

6.4.10 Intersystem Link Protocol (ISLP)

ISLP is supported by setting the following bit field value referenced in Table 5-18, *Channel Configuration Descriptor*:
FCS = 1, PROTOCOL = 2

6.5 Signaling System 7

6.5.1 SS7 Repeat Message Transmission

Signaling System 7 (SS7) requires the ability to continuously repeat a message under certain circumstances. The Repeat Message Transmission section of this document describes the repeat feature fully and is usable for an SS7 application.

6.5.2 Message Filtering

Message filtering is always enabled for a receive channel which is configured for SS7-HDLC-CRC16 mode in Table 5-18, *Channel Configuration Descriptor*.

When receiving an unerrored message with a payload length of 3, 4, or 5 octets and the required 2-octet FCS, MUSYCC transfers the data into the memory buffer as usual and then enters the message filtering mode. This mode does not apply to messages with payloads greater than 5 octets.

In this mode, the next Message Descriptor is processed in the normal manner (receive Buffer Status Descriptor is written, next pointer is used to read next Message Descriptor, ONR and NP bits are checked, and so on). In this case, the current memory buffer is not filled until MUSYCC exits the message filtering mode.

The 3-, 4-, or 5-octet payload and the 2-octet FCS are stored inside MUSYCC and are considered golden messages by being 3, 4, or 5 octets long. Subsequent received messages are compared to this message.

Each bit of the subsequent message will be compared bit-wise to the contents of the golden message. The following can occur:

- If a match occurs, the payload and FCS of the golden message match the current message before a closing flag is received for the current message. The maskable interrupt MSKSFILT is generated to the host. MUSYCC remains in message filtering mode, and the golden message is retained.
- If a mismatch occurs, a bit-wise mismatch is detected between a bit in the golden message and the corresponding bit in the current message before a closing flag is received for the current message. MUSYCC immediately exits the filter mode.

The treatment for the current message becomes a normal message treatment. If the current message is unerrored and qualifies to become a golden message, it is transferred to the current Receive Data Buffer and becomes the new golden message internally to MUSYCC.

If the current message is unerrored and greater than 6 octets total, it cannot become a golden message, and MUSYCC continues normal message treatment.

6.5.3 Signal Unit Error Rate Monitoring

The Signal Unit Error Rate Monitor (SUERM) facility provides a 6-bit counter which serves as a real-time figure-of-merit for the receive link integrity. It is incremented and decremented in a “leaky-bucket” style, based upon integration of good message and bad octet periods on the receive channel.

6.5.4 SUERM Counter Incrementing

The SUERM counter increments when any signal unit error occurs. A signal unit error is defined as one of the following events:

- SHT: Short Frame error
- ALIGN: Octet Alignment error
- CRC: FCS Mismatch error
- Accumulation of 16 octet count errors

Short Frame errors, Octet Alignment errors, or CRC/FCS Mismatch errors generate a maskable interrupt, SHT, LNG, CRC respectively, toward the host and cause the SUERM counter to be incremented. Each time the SUERM counter is incremented, the maskable interrupt SINC is generated to the host indicating this condition.

6.5.5 SUERM Octet Counting

Octet counting mode is entered if seven consecutive 1s are detected (abort condition), or the received message length exceeds the selected maximum received-frame length register value (long frame error)

When in Octet counting mode, a 4-bit bad octet counter is incremented for every received octet until a condition is met to exit this mode. As the counter rolls over from 15 to 0, the SUERM counter is incremented by one. This mode is exited when a correctly checked signal unit (unerrored message) is detected.

Each time the octet counting mode is entered, the value of 4-bit bad octet counter is reset.

6.5.6 SUERM Counter Decrementing

The SUERM counter is decremented when 256 unerrored messages are received. An unerrored message indicates that a short frame or long frame error was not detected, no octet alignment error was detected, and no CRC error was detected. Each unerrored message increments an 8-bit good message counter. When this counter rolls over from 255 to 0, the SUERM counter is decremented by one.

While in octet counting mode, the value of the 8-bit good message counter is maintained from the last non-octet counting mode and starts to increment again from that value when a good message causes an exit from the octet counting mode.

When the SUERM counter decrements, the maskable interrupt SDEC is generated to the host indicating this condition.

6.6 Self-Servicing Buffers

The transmit and receive Buffer Descriptors and Buffer Status Descriptors are designed to facilitate a mechanism known as "self-servicing buffers." This mechanism allows the host to configure MUSYCC to fill a linked list of data buffers as it receives a complete message through a receive channel, and empty that same list of data buffers through a transmit channel without any further host intervention.

The mechanism works as follows:

1. Host initializes linked list of Message Descriptors in shared memory.
2. Host configures receive channel to point to first Message Descriptor.
3. Host configures transmit channel to point to the same Message Descriptor.
4. The OWNER bit field in the Buffer Descriptor in the Message Descriptor is set to 0. Therefore, for the transmitter, the buffer is owned by the host; for the receiver, the buffer is owned by MUSYCC.
5. Both receive and transmit channel are activated.
6. As the receiver detects a valid incoming message, it begins filling the first data buffer from the linked list. The transmitter remains idle, polling the OWNER bit in the Transmit Buffer Descriptor.
7. As the receiver fills the first buffer, it writes the Receive Buffer Status Descriptor (and sets OWNER to 1) and moves to the Next Message Pointer which identifies the next Receive Data Buffer on the linked list.
8. The transmit channel detects the OWNER set to 1 for the first Transmit Data Buffer, assumes ownership of the buffer, and begins emptying data to the serial port.
9. Upon detecting the end of a message, the receiver writes the Receive Buffer Status Descriptor and marks this last buffer as containing the End of Message and sets the buffer length field, BLEN to indicate the amount of data received in this last buffer.
10. When the transmitter detects the End of Message marking in the last buffer, the transmitter sends the final BLEN amount of data out the serial port and writes the Transmit Buffer Status Descriptor (and sets OWNER to 0) and moves into the idle state again.
11. Go to step 6 to continue processing the next message.

NOTE: For self-servicing buffers, the host need not write to any descriptors for receive or transmit operations. MUSYCC writes the Receive Buffer Status Descriptor, which is subsequently used as the Transmit Buffer Descriptor.

7.0 Electrical and Mechanical Specifications

7.1 Electrical and Environmental Specifications

7.1.1 Absolute Maximum Ratings

CAUTION

Stressing the device parameters above absolute maximum ratings may cause permanent damage to the device. This is a stress rating only. Functional operation of the device at these or any other conditions beyond those listed in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

Table 7-1. Absolute Maximum Ratings

Parameter	Symbol	Value	Unit
Supply Voltage	V_{dd}, V_{ddo}	–0.5 to 4.6	V
Continuous Power Dissipation	P_d	750	mW
Operating Junction Temperature	T_{jc}	125	°C
Storage Temperature	T_s	–55 to +125	°C
5 V–Tolerant Supply	V_{GG}	–0.5 to 6	V
Core Supply	V_{DDC}	–0.5 to 3.3	V
5 V–Tolerant DC Input	Input, Hi–Z Out	–0.5 to V_{GG} + 0.5 (not to exceed 6 V)	V
5 V–Tolerant DC Output	Output Lo-Z	–0.5 to V_{dd} + 0.5 (not to exceed 4.6 V)	V

7.1.2 Recommended Operating Conditions

Table 7-2. Recommended Operating Conditions

Parameter	Symbol	Value	Unit
Supply Voltage	V_{ddi}, V_{ddo}	3.0 to 3.6	V
Ambient Operating Temperature EPF	T_{ac}	 –40 to +85	°C
High-Level Input Voltage	V_{ih}	2.0 to V_{GG} + 0.3	V
Low-Level Input Voltage	V_{il}	–0.3 to 0.8	V
High-Level Output Current Source	I_{oh}	200 to 400	µA
Low-Level Output Current Sink	I_{ol}	2 to 3	mA
Output Capacitive Loading	C_{ld}	60	pF
5 V Tolerant Supply(1)	V_{GG}	4.75 to 5.25	V
Core Supply	V_{DDC}	2.3 to 2.7	V

NOTE(S):
(1) V_{GG} input can be supplied by V_{DD} in the 3.3 V signaling environment.

7.1.3 Electrical Characteristics

Table 7-3. Electrical Operating Characteristics

Parameter	Symbol	Value	Unit
High-Level Output Voltage	V_{oh}	2.4	V
Low-Level Output Voltage	V_{ol}	0.4	V
Input Leakage Current	I_l	–1 to 1	µA
Three-state Leakage Current	I_{oz}	–10 to 10	µA
Resistive Pullup Current	I_{pr}	100 to 500	µA
Supply Current	I_{dd}	130	µA

7.2 Timing and Switching Specifications

7.2.1 Overview

The major subsystems of MUSYCC are the host interface, the expansion bus interface, and the serial interface. The host interface is PCI compliant. For other references to PCI, see the *PCI Local Bus Specification*, Revision 2.1, June 1, 1995. The expansion bus and serial bus interfaces are similar to the host interface timing characteristics; the differences and specific characteristics common to either interface are further defined.

7.2.2 Host Interface (PCI) Timing and Switching Characteristic

Reference the *PCI Local Bus Specification*, Revision 2.1, June 1, 1995 for information on:

- Indeterminate inputs and metastability
- Power requirements, sequencing, decoupling
- PCI DC specifications
- PCI AC specifications
- PCI V/I curves
- Maximum AC ratings and device protection

Table 7-4. PCI Interface DC Specifications *(1 of 2)*

Symbol	Parameter	Condition	Min	Max	Unit
V_{DDi}, V_{DDo}	Supply Voltage	—	3.0	3.6	V
V_{ih}	Input High Voltage	—	$0.5V_{DD}$	V_{GG} + 0.5	V
V_{il}	Input Low Voltage	—	–0.5	$0.3V_{DD}$	V
V_{ipu}	Input Pull-up Voltage(1)	—	$0.7V_{DD}$	—	V
I_{il}	Input Leakage Current(2)	$0 < V_{in} < V_{DD}$	—	±10	µA
V_{oh}	Output High Voltage	I_{out} = –500 µA	$0.9V_{DD}$	—	V

Table 7-4. PCI Interface DC Specifications *(2 of 2)*

Symbol	Parameter	Condition	Min	Max	Unit
V_{ol}	Output Low Voltage[3]	I_{out} = 1500 µA	—	0.1 V_{DD}	V
C_{in}	Input Pin Capacitance[4]	—	—	10	pF
C_{clk}	CLK Pin Capacitance	—	5	12	pF
C_{IDSEL}	IDSEL Pin Capacitance[5]	—	—	8	pF
L_{pin}	Pin Inductance[6]	—	—	20	nH

NOTE(S):

(1) Guaranteed by design. It is the minimum voltage to which pull-up resistors are calculated to pull a floated network. Applications sensitive to static power utilization should ensure that the input buffer is conducting minimum current at this input voltage.

(2) Input leakage currents include hi-Z output leakage for all bidirectional buffers with three-state outputs.

(3) Signals without pull-up resistors must have 3 mA low output current. Signals requiring pull-up must have 6 mA; the latter includes FRAME*, TRDY*, IRDY*, DEVSEL*, STPP*, SERR*, and PERR*

(4) Absolute maximum pin capacitance for a PCI input is 10 pF (except for CLK) with an exception granted to motherboard-only devices, which could be up to 16 pF, in order to accommodate PGA packaging. This would mean, in general, that components for expansion boards would need to use alternatives to ceramic PGA packaging – i.e., PQFP, SGA, etc.

(5) Lower capacitance on this input-only pin allows for non-resistive coupling to AD[xx].

(6) This is a recommendation, not an absolute requirement. The actual value should be provided with the component data sheet.

Table 7-5. PCI Clock (PCLK) Waveform Parameters, 33 MHz PCI Clock

Symbol	Parameter	Min	Max	Unit
T_{cyc}	Clock Cycle Time[1]	30	—	ns
T_{high}	Clock High Time	11	—	ns
T_{low}	Clock Low Time	11	—	ns
–	Clock Slew Rate[2]	1	4	V/ns
V_{ptp}	Peak-to-Peak Voltage	0.4 V_{dd}	—	V

NOTE(S):

(1) MUSYCC works with any clock frequency between DC and 66 MHz, nominally. The clock frequency can be changed at any time during operation of the system as long as clock edges remain monotonic, and minimum cycle and high and low times are not violated. The clock can only be stopped in a low state.

(2) Rise and fall times are specified in terms of the edge rate measured in V/ns. This slew rate must be met across the minimum peak-to-peak portion of the clock waveform.

Table 7-6. PCI Clock (PCLK) Waveform Parameters, 66 MHz PCI Clock

Symbol	Parameter	Min	Max	Unit
T_{cyc}	Clock Cycle Time(1)	15	—	ns
T_{high}	Clock High Time	6	—	ns
T_{low}	Clock Low Time	6	—	ns
—	Clock Slew Rate(2)	1	4	V/ns
V_{ptp}	Peak-to-Peak Voltage	0.4 V_{dd}	—	V

NOTE(S):

(1) MUSYCC works with any clock frequency between DC and 66 MHz, nominally. The clock frequency can be changed at any time during operation of the system as long as clock edges remain monotonic, and minimum cycle and high and low times are not violated. The clock can only be stopped in a low state.

(2) Rise and fall times are specified in terms of the edge rate measured in V/ns. This slew rate must be met across the minimum peak-to-peak portion of the clock waveform.

Figure 7-1. PCI Clock (PCLK) Waveform

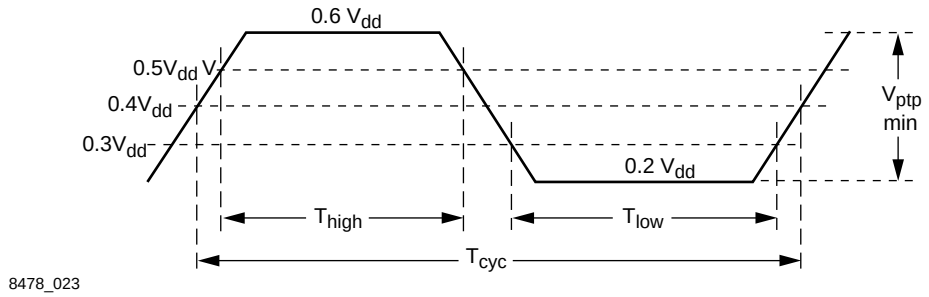


Table 7-7. PCI Reset Parameters

Symbol	Parameter	Min	Max	Unit
T_{rst}	Reset Active Time after Power Stable	1	—	ms
T_{rst_clk}	Reset Active Time after Clock Stable	100	—	µs
V_{nom}	Nominal Voltage Level(1)	—	—	V
—	RST* Slew Rate(2)	50	—	mV/ns
T_{fail}	Power Failure Detect Time(3)	—	—	—
$T_{rst-off}$	Reset Active to Float Delay	—	40	ns

NOTE(S):

(1) The nominal voltage level refers to a voltage test point in the power-up curve where the system can declare start of a "power good" signal.

(2) The minimum RST* slew rate applies only to the rising (deassertion) edge of the reset signal, and ensures that system noise cannot render an otherwise monotonic signal to appear to bounce in the switching range.

(3) The value of T_{fail} is the minimum of

a. 500 ns (max) from power rail going out of specification by exceeding specified tolerances by more than 500 mV.

b. 100 ns (max) from 5 V rail falling below 3.3 V rail by more than 300 mV.

Figure 7-2. PCI Reset Timing

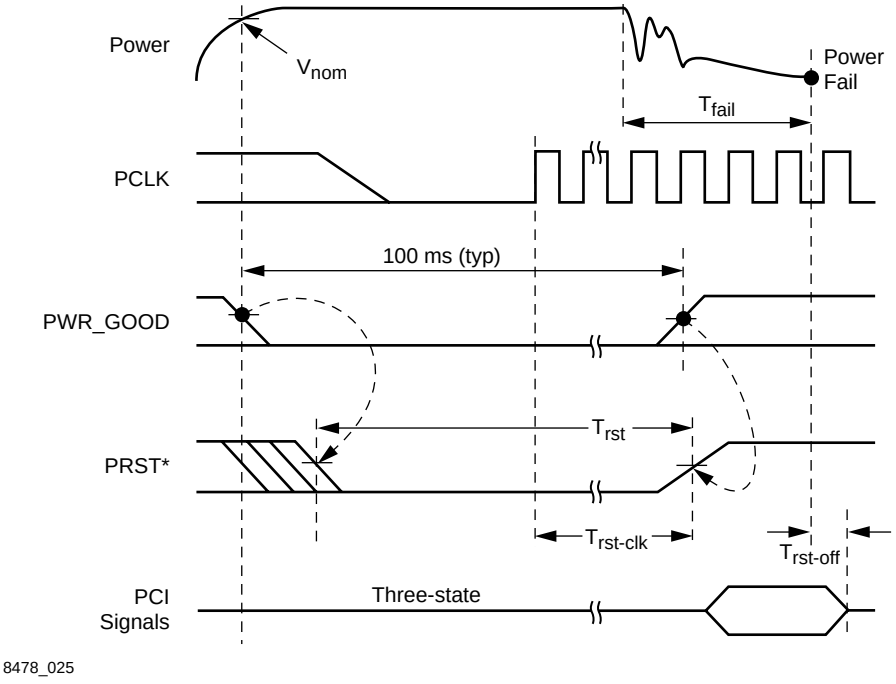


Table 7-8. PCI I/O Timing Parameters, 33 MHz PCI Clock

Symbol	Parameter	Min	Max	Unit
T_{val}	PCLK to Signal Valid Delay—Bused Signal *(1, 2, 4)*	2	11	ns
T_{val} (ptp)	PCLK to Signal Valid Delay—Point To Point *(1, 2)*	2	12	ns
T_{on}	Float to Active Delay *(3)*	—	13	ns
T_{off}	Active to Float Delay *(3)*	—	28	ns
T_{ds}	Input Setup Time to Clock—Bused Signal *(2)*	7	—	ns
T_{su} (ptp)	Input Setup Time to Clock—Point To Point *(2)*	10, 12	—	ns
T_{dh}	Input Hold Time from Clock (5)	0	—	ns

NOTE(S):
(1) Minimum and maximum times are evaluated at 80 pF equivalent load. Actual test capacitance may vary, and results should be correlated to these specifications.
(2) REQ* and GNT* are the only point-to-point signals and have different output valid delay and input setup times than bused signals. GNT* has a setup of 10; REQ* has a setup of 12.
(3) For purposes of active/float timing measurements, the high-Z or off state is when the total current delivered through the component pin is less than or equal to the leakage current specification at 80 pF equivalent load.
(4) T_{VAL} = 17 ns max for INTA.
(5) T_{dh} = 0.5 ns min for GNT, IDSEL, and IRDY.

Table 7-9. PCI I/O Timing Parameters, 66 MHz PCI Clock

Symbol	Parameter	Min	Max	Units
T_{val}	PCLK to Signal Valid Delay—Bused Signal *(1, 2)*	2	6	ns
T_{val} (ptp)	PCLK to Signal Valid Delay—Point To Point *(1, 2)*	2	6	ns
T_{on}	Float to Active Delay *(3)*	2	—	ns
T_{off}	Active to Float Delay *(3)*	—	14	ns
T_{ds}	Input Setup Time to Clock—Bused Signal *(2)*	3	—	ns
T_{su} (ptp)	Input Setup Time to Clock—Point To Point *(2)*	5	—	ns
T_{dh}	Input Hold Time from Clock	0	—	ns

NOTE(S):
(1) Minimum and maximum times are evaluated at 80 pF equivalent load. Actual test capacitance may vary, and results should be correlated to these specifications.
(2) REQ* and GNT* are the only point-to-point signals and have different output valid delay and input setup times than bused signals. GNT* has a setup of 10; REQ* has a setup of 12.
(3) For purposes of active/float timing measurements, the high-Z or off state is when the total current delivered through the component pin is less than or equal to the leakage current specification at 80 pF equivalent load.

Table 7-10. PCI I/O Measure Conditions

Symbol	Parameter	Value	Unit
V_{th}	Voltage Threshold High(1)	0.6 V_{dd}	V
V_{tl}	Voltage Threshold Low(1)	0.2 V_{dd}	V
V_{test}	Voltage Test Point	0.4 V_{dd}	V
V_{max}	Maximum Peak-to-Peak(2)	0.4 V_{dd}	V
—	Input Signal Edge Rate	1	V/ns

NOTE(S):

(1) The input test is done with 0.1 V_{dd} of overdrive (over V_{ih} and V_{il}). Timing parameters must be met with no more overdrive than this. Production testing can use different voltage values, but must correlate results back to these parameters.

(2) V_{max} specifies the maximum peak-to-peak voltage waveform allowed for measuring input timing. Production testing can use different voltage values, but must correlate results back to these parameters.

Figure 7-3. PCI Output Timing Waveform

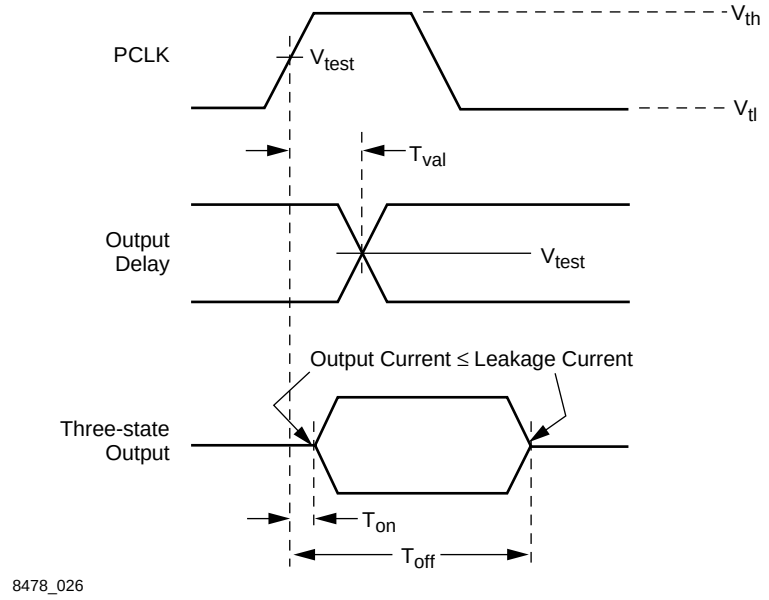


Figure 7-4. PCI Input Timing Waveform

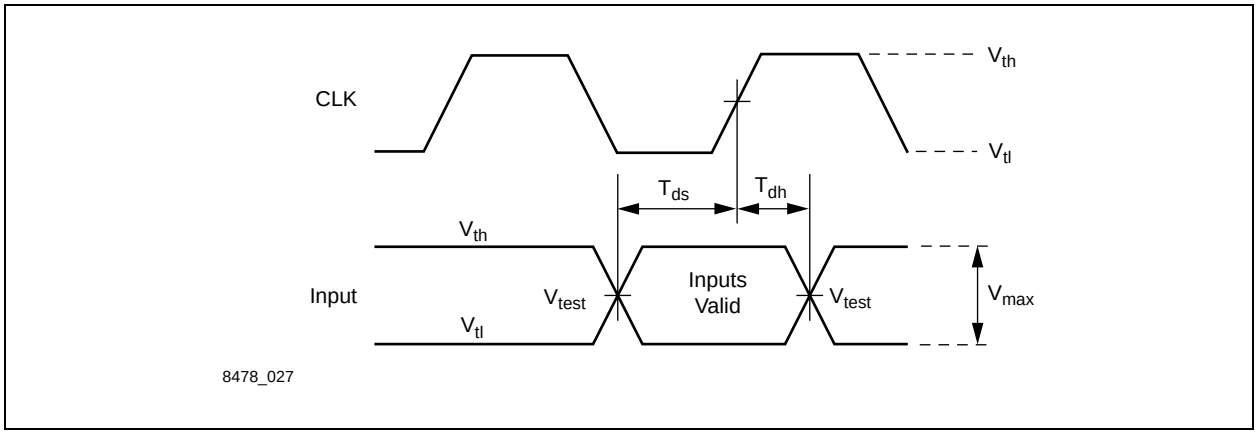


Figure 7-5. PCI Read Multiple Operation

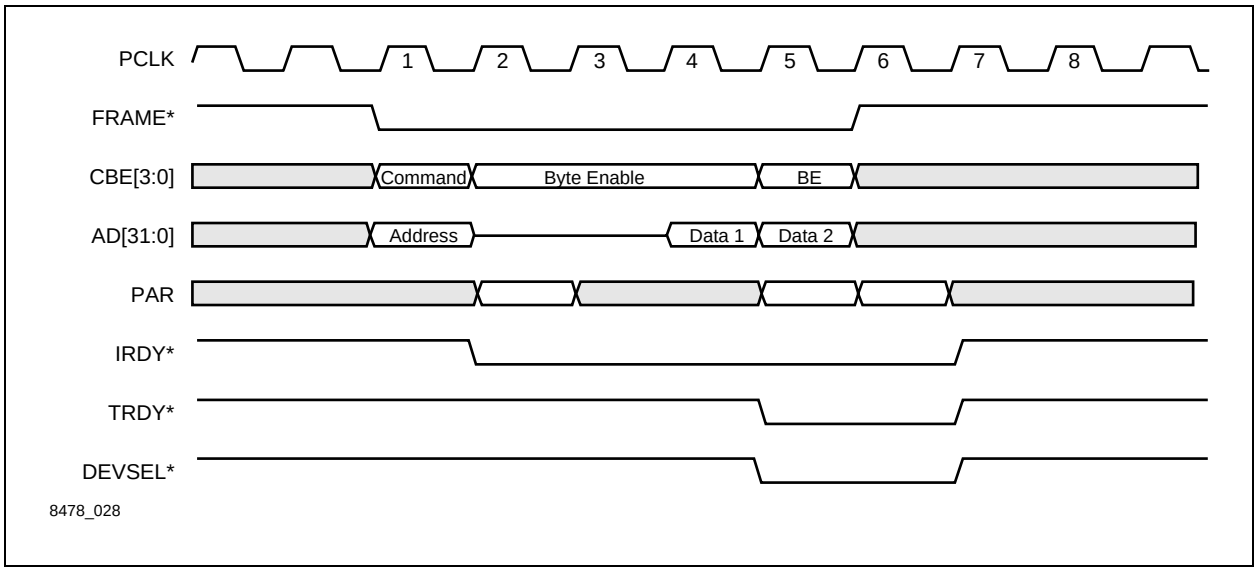


Figure 7-6. PCI Write Multiple Operation

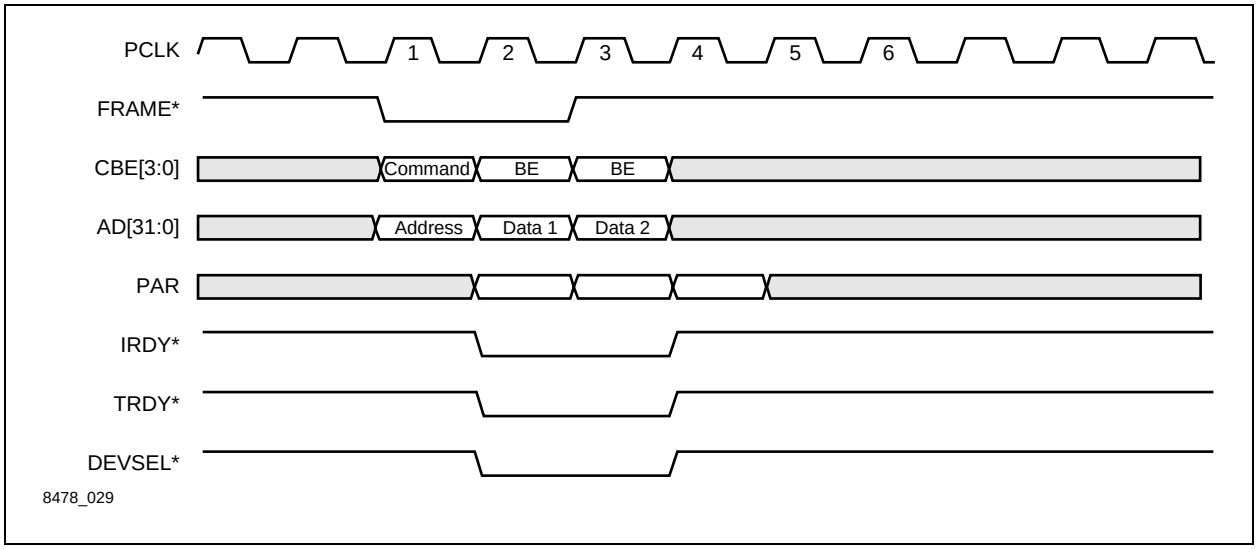
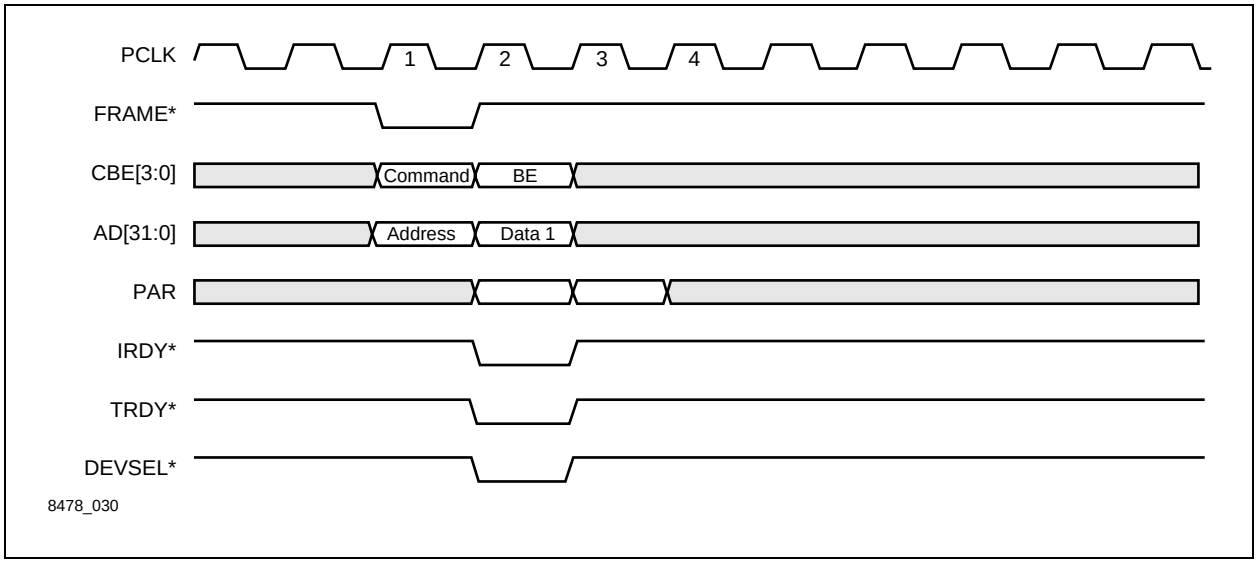


Figure 7-7. PCI Write Single Operation



7.2.3 Expansion Bus (EBUS) Timing and Switching Characteristic

The EBUS timing is derived from the PCI clock (PCLK) input to MUSYCC. The ECLK output is either one-half of the PCI clock (M66EN = 1) or the same as the PCI clock (M66EN = 0); the ECLK and PCLK relationship is shown in Figures 7-8 and 7-9.

The EBUS I/O timing characteristics are identical to the PCI I/O timing characteristics.

The EBUS clock waveform characteristics are identical to the PCI clock waveform characteristics (refer to Tables 7-11 through 7-13 and Figures 7-10 through 7-12).

Figure 7-8. ECLK to PCLK Relationship (M66EN = 0)

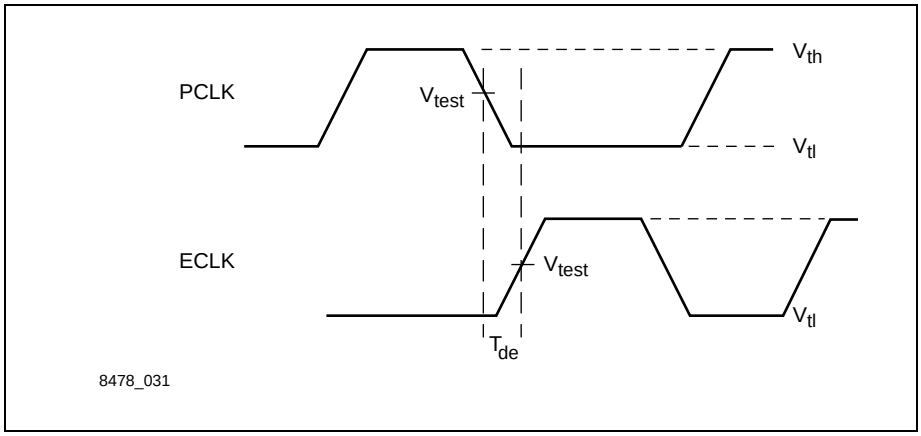


Figure 7-9. ECLK to PCKL Relationship (M66EN = 1)

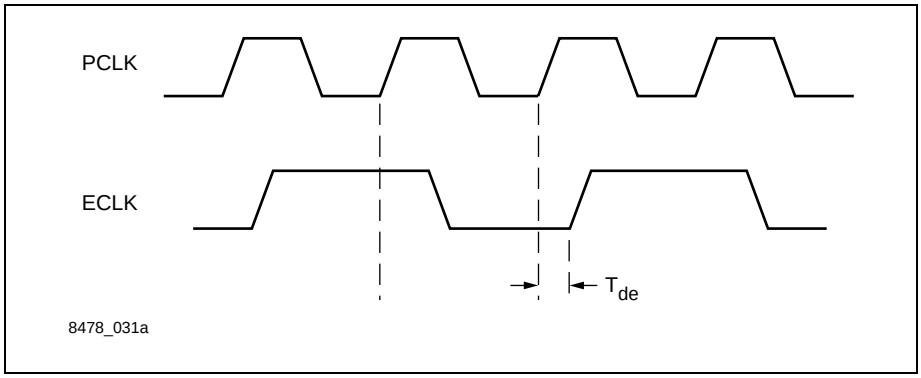
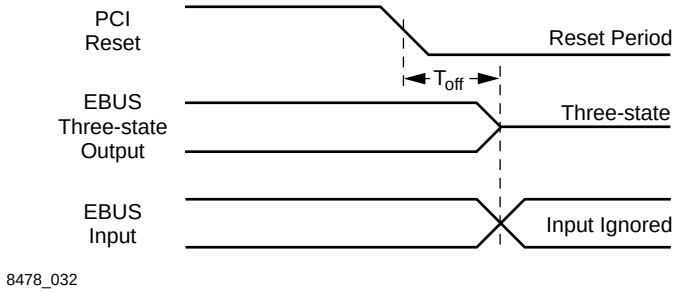


Table 7-11. EBUS Reset Parameters

Symbol	Parameter	Min	Max	Units
T_{off}	Active to Inactive Delay(1)	—	30	ns

NOTE(S):
(1) For purposes of active/float timing measurements, the high-Z or off state is when the total current delivered through the component pin is less than or equal to the leakage current specification.

Figure 7-10. EBUS Reset Timing



NOTE(S): The EBUS reset is dependent on the PRST* (PCI Reset) signal being asserted low.

Table 7-12. EBUS I/O Timing Parameters

Symbol	Parameter	Min	Max	Units
T_{val}	PCI Clock Fall to Signal Valid Delay—Bused Signal(1)	2	15	ns
T_{val} (ptp)	PCI Clock Fall to Signal Valid Delay—Point To Point(1)	2	15	ns
T_{on}	Float to Active Delay(2)	—	30	ns
T_{off}	Active to Float Delay(2)	—	30	ns
T_{ds}	Input Setup Time to Clock—Bused Signal	3	—	ns
T_{ds} (ptp)	Input Setup Time to Clock—Point To Point	3	—	ns
T_{dh}	Input Hold Time from Clock	7	—	ns
T_{de}	PCI Clock Fall to ECLK rising edge	—	7	ns

NOTE(S):

(1) Minimum and maximum times are evaluated at 80 pF equivalent load. Actual test capacitance may vary, and results should be correlated to these specifications.

(2) For purposes of active/float timing measurements, the hi-z or off state is when the total current delivered through the component pin is less than or equal to the leakage current specification at 80 pF equivalent load.

Table 7-13. EBUS I/O Measure Conditions

Symbol	Parameter	Value	Units
V_{th}	Voltage Threshold High(1)	0.6 V_{dd}	V
V_{tl}	Voltage Threshold Low(1)	0.2 V_{dd}	V
V_{test}	Voltage Test Point	0.4 V_{dd}	V
V_{max}	Maximum Peak-to-Peak(2)	0.4 V_{dd}	V
—	Input Signal Edge Rate	1	V/ns

NOTE(S):

(1) The input test for the 3.3 V environment is done with 0.1 V_{dd} of overdrive (over V_{ih} and V_{il}). Timing parameters must be met with no more overdrive than this. Production testing can use different voltage values, but must correlate results back to these parameters.

(2) V_{max} specifies the maximum peak-to-peak voltage waveform allowed for measuring input timing. Production testing can use different voltage values, but must correlate results back to these parameters.

Figure 7-11. EBUS Output Timing Waveform

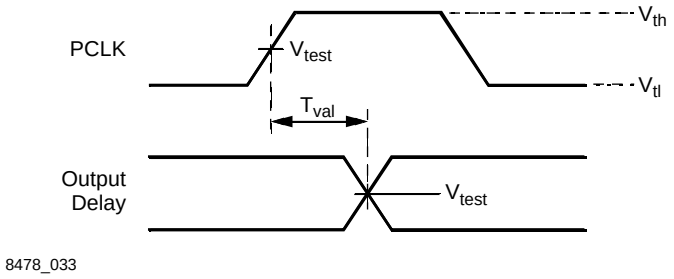
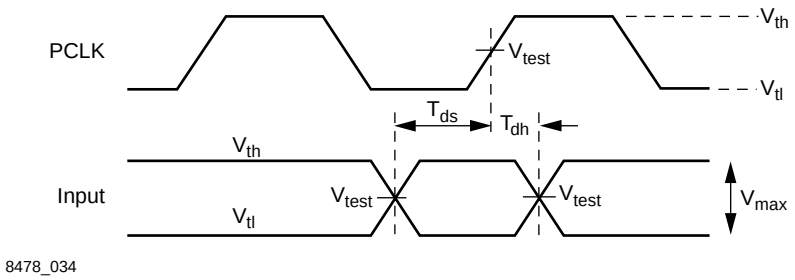


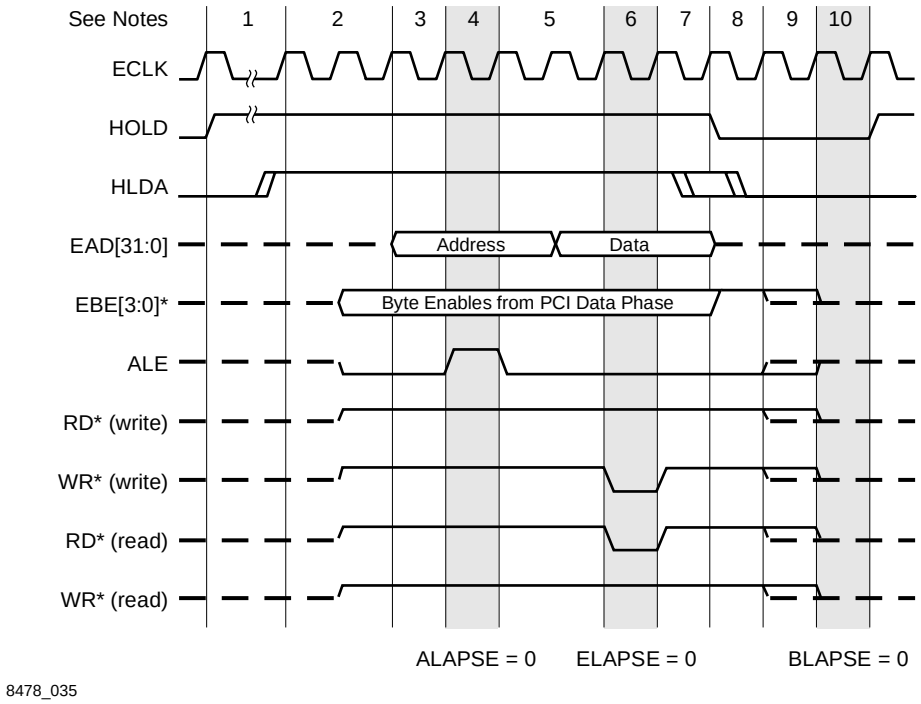
Figure 7-12. EBUS Input Timing Waveform



7.2.4 EBUS Arbitration Timing

Illustrated in Figures 7-13 and 7-14 are Intel- and Motorola-style write and read transactions.

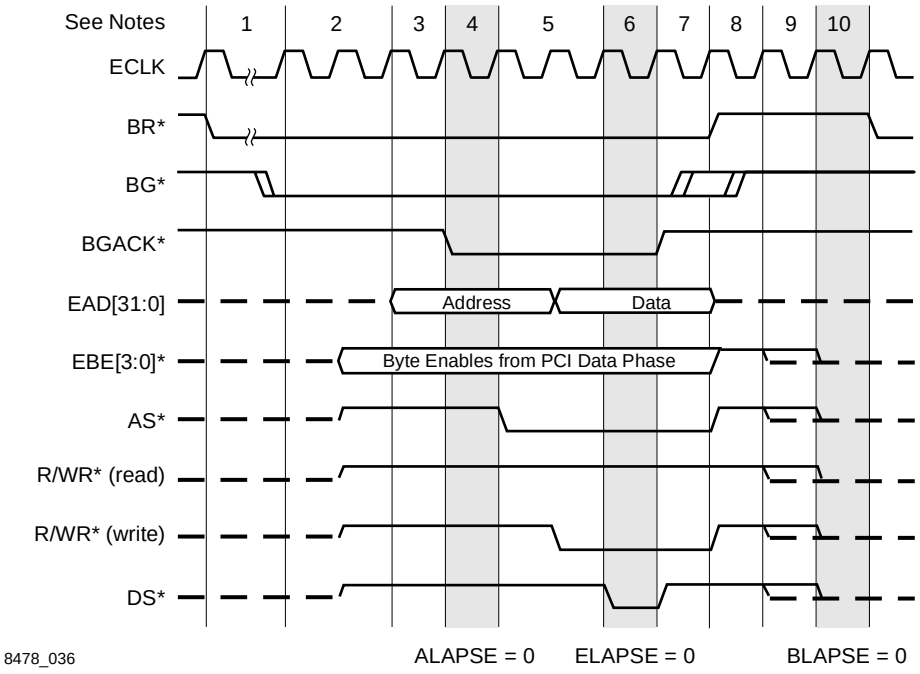
Figure 7-13. EBUS Write/Read Transactions, Intel-Style



NOTE(S):

1. HLDA assertion depends on the external bus arbiter. While HOLD and HLDA are both deasserted, MUSYCC places shared EBUS signals in high impedance (three-state, shown as dashed lines).
2. MUSYCC outputs valid command bus signals: EBE, ALE, RD*, and WR* 1 ECLK cycle after HLDA assertion.
3. MUSYCC outputs valid EAD address signals, 2 ECLK cycles after HLDA assertion.
4. ALE assertion occurs 3 ECLK cycles after HOLD and HLDA are both asserted.ALAPSE inserts a variable number of ECLK cycles to extend ALE high pulse width and EAD address interval.
5. EAD address remains valid for 1 ECLK cycle after ALE falling edge. During a write transaction, MUSYCC outputs valid EAD write data 1 ECLK prior to WR* assertion. During a read transaction, EAD data lines are inputs.
6. ELAPSE inserts a variable number of ECLK cycles to extend RD*/WR* low pulse width and EAD data intervals. Read data inputs are sampled on ECLK rising edge coincident with RD* deassertion.
7. EAD write data and EBE byte enables remain valid for 1 ECLK cycle after RD*/WR* deassertion.
8. HOLD is deasserted, and the bus is parked (command bus deasserted, EAD tristate) 1 ECLK after RD* or WR* deassertion. The bus parked state ends when HLDA is deasserted, 1 ECLK after RD* or WR* deassertion.
9. Command bus is unparked (three-stated) one ECLK after HLDA deassertion; two different unpark phases are shown, indicating the dependence on HLDA deassertion. If HLDA remained asserted until the next bus request, then command bus remains parked until 1 ECLK cycle following the next HOLD assertion. Warning: Whenever HLDA is deasserted, all shared EBUS signals are forced to three-state after 1 ECLK cycle, regardless of whether the EBUS transaction was completed. MUSYCC will not reissue or repeat such an aborted transaction.
10. BLAPSE inserts a variable number of ECLK cycles to extend HOLD deassertion interval until the next bus request.

Figure 7-14. EBUS Write/Read Transactions, Motorola-Style



NOTE(S):

1. BG* assertion depends on the external bus arbiter. While BG* and BR* are both deasserted, MUSYCC places shared EBUS signals in high impedance (three-state, as shown by dashed lines).
2. One ECLK cycle after BG* assertion, MUSYCC outputs valid command bus signals: EBE, AS*, R/WR*, and DS*.
3. Two ECLK cycles after BG* assertion, MUSYCC outputs valid EAD address signals. BGACK* assertion occurs three ECLK cycles after BG* and BR* are both asserted.
4. ALAPSE inserts a variable number of ECLK cycles to extend AS* high pulse width and EAD address interval.
5. EAD address remains valid for one ECLK cycle after AS* falling edge. During a write transaction, MUSYCC asserts R/WR* and outputs valid EAD write data one ECLK prior to DS* assertion. During a read transaction, EAD data lines are input.
6. ELAPSE inserts a variable number of ECLK cycles to extend DS* low pulse width and EAD data interval. Read data inputs are sampled on ECLK rising edge coincident with DS* deassertion.
7. EAD write data, EBE, R/WR*, and AS* signals remain valid for one ECLK cycle after BGACK* and DS* are deasserted.
8. One ECLK cycle after BGACK* deassertion, the BR* output is deasserted and the bus is parked (command bus deasserted, EAD three-state). The bus parked state ends when the external bus arbiter deasserts BG*.
9. Command bus is unparked (three-stated) one ECLK after BG* deassertion; two different unpark phases are shown, indicating the dependence on BG* deassertion. If BG* remained asserted until the next bus request, then command bus remains parked until one ECLK following the next BR* assertion. Warning: Whenever BG* is deasserted, all shared EBUS signals are forced to three-state after one ECLK cycle, regardless of whether the EBUS transaction was completed. MUSYCC will not reissue or repeat such an aborted transaction.
10. BLAPSE inserts a variable number of ECLK cycles to extend BR* deassertion interval until the next bus request.

7.2.5 Serial Interface Timing and Switching Characteristics

Serial interface timing and switching characteristics are provided in Tables 7-14 through 7-17 and Figures 7-15 through 7-18.

Table 7-14. Serial Interface Clock (RCLK, TCLK) Parameters

Symbol	Parameter	Min	Max	Units
F_c	Clock Frequency	DC	8.192 ±10%	MHz
T_r	Clock Rise Time	—	2	ns
T_f	Clock Fall Time	—	2	ns

Figure 7-15. Serial Interface Clock (RCLK,TCLK) Waveform

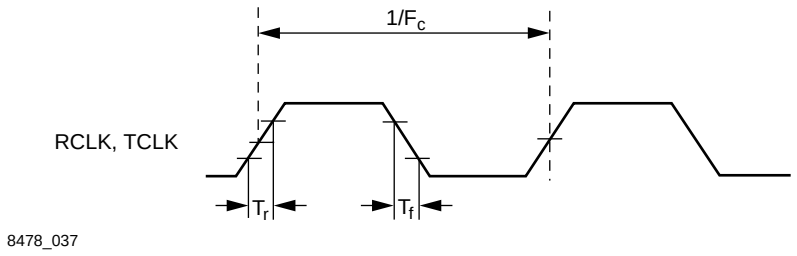


Table 7-15. Serial Interface I/O Timing Parameters

Symbol	Parameter	Min	Max	Units
T_{val}	Clock to Signal Valid Delay	2	20	ns
T_{ds}	Data Setup Time	10	—	ns
T_{dh}	Data Hold Time	10	—	ns

Table 7-16. Serial Interface Clock Hysteresis (RCLK, TCLK, with Schmitt Trigger)

Symbol	Parameter	Min	Max	Units
V_{TH}	High Threshold Voltage	0.7* VDDi	—	V
V_{TL}	Low Threshold Voltage	0	0.3* VDDi	V
V_H	Hysteresis	0.3	—	V

Figure 7-16. Serial Interface Clock (RCLK,TCLK) Waveform

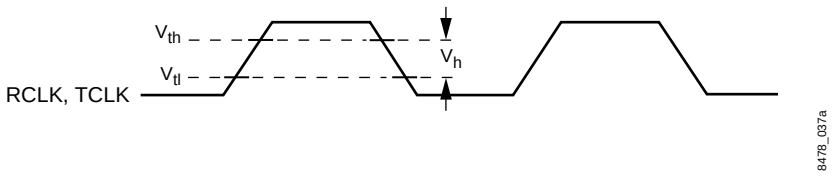


Table 7-17. Serial Interface I/O Measure Conditions for 3.3 V Signaling

Symbol	Parameter	Value	Units
V_{th}	Voltage Threshold High(1)	0.6 V_{dd}	V
V_{tl}	Voltage Threshold Low(1)	0.2 V_{dd}	V
V_{test}	Voltage Test Point	0.4 V_{dd}	V
V_{max}	Maximum Peak-to-Peak(2)	0.4 V_{dd}	V
—	Input Signal Edge Rate	1	V/ns

NOTE(S):

(1) The input test for the 3.3 V environment is done with 0.1 V_{dd} of overdrive (over V_{ih} and V_{il}). Timing parameters must be met with no more overdrive than this. Production testing can use different voltage values, but must correlate results back to these parameters.

(2) V_{max} specifies the maximum peak-to-peak voltage waveform allowed for measuring input timing. Production testing can use different voltage values, but must correlate results back to these parameters.

Figure 7-17. Serial Interface Data Input Waveform

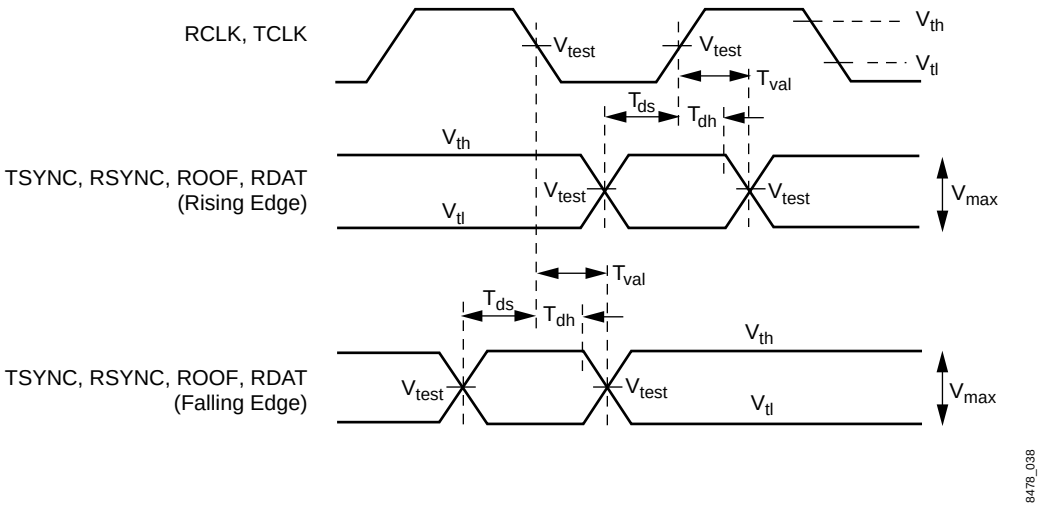
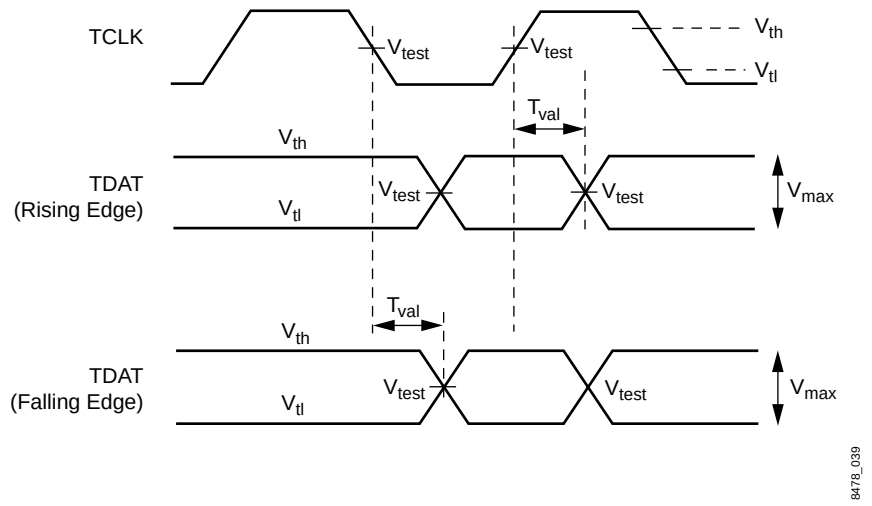


Figure 7-18. Serial Interface Data Delay Output Waveform



7.2.6 Package Thermal Specification

Table 7-18 lists the package thermal specifications.

Table 7-18. MUSYCC Package Thermal Resistance Characteristics

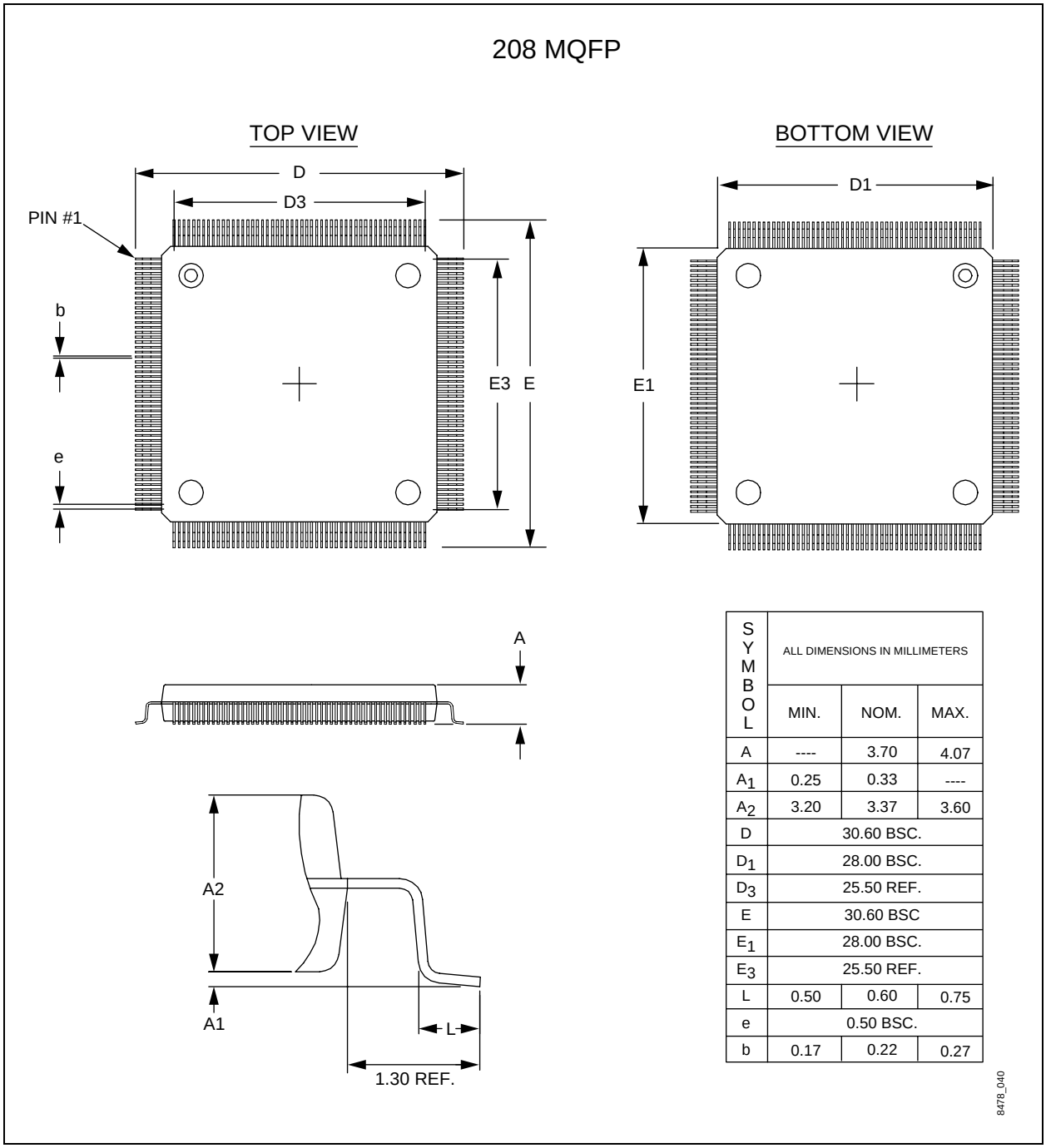
Package	Mounting Conditions	Airflow–LFM (LMS)				
		0 (0.000)	50 (0.256)	100 (0.505)	200 (1.01)	400 (2.03)
Thermal Resistance (junction to ambient) = **C/W						
208-BGA	Board-Mounted	26	22	19	18	17
208-Pin Quad Flat Pack	Board–Mounted	21	19	17	16	14
208-Pin Quad Flat Pack	Socket	23	21	19	18	16

NOTE(S):
1. LFM–linear feet per minute.
2. LMS–linear meters per second.
3. Junction to case temperature (°C): $T_{jc} = T_{ac} + (\theta_{ja} \times P_d)$.
   Tjc = θja x Pd(measured) + Tac(measured)
   Where: Tjc = Junction Temperature (see Table 7-1).
   θja = Thermal Resistance (θ-ja, see Table 7-18).
   Tac = Ambient Case Temperature (see Table 7-2).
   Pd = Power Dissipation = Vdd x Idd (Table 7-1).

7.2.7 Mechanical Specifications

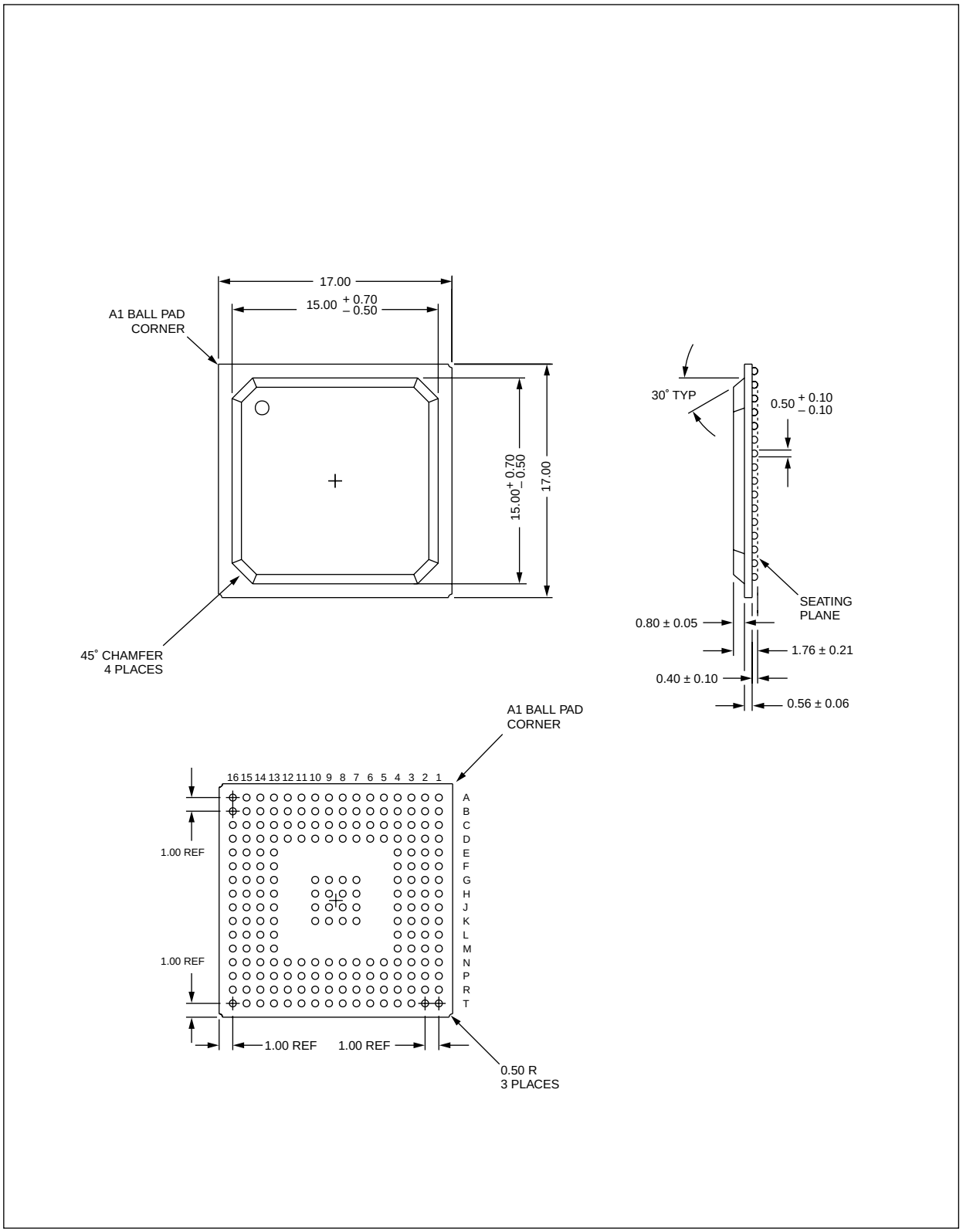
Figures 7-19 and 7-20 illustrate the mechanical specifications.

Figure 7-19. 208-Pin Metric Quad Flatpack (MQFP)



SYMBOL	ALL DIMENSIONS IN MILLIMETERS		
	MIN.	NOM.	MAX.
A	----	3.70	4.07
A_1	0.25	0.33	----
A_2	3.20	3.37	3.60
D		30.60 BSC.	
D_1		28.00 BSC.	
D_3		25.50 REF.	
E		30.60 BSC	
E_1		28.00 BSC.	
E_3		25.50 REF.	
L	0.50	0.60	0.75
e		0.50 BSC.	
b	0.17	0.22	0.27

Figure 7-20. 208-Pin Plastic Ball Grid Array (PBGA)



8.0 Terms, Definitions, and Conventions

This document assumes the reader is familiar with the design and development of PCI systems software and hardware, and with the message layout used in the HDLC protocol. Most of the PCI reference material is located in the *PCI Local Bus Specification*, Revision 2.1, June 1, 1995.

8.1 Applicable Specifications

The following documents were used as reference material for MUSYCC and this document.

- *PCI Local Bus Specification*
  Revision 2.1, Version, June 1, 1995
- *ITU-T Recommendation Q.921 (03/93)*
  Digital Subscriber Signaling System No. 1
- *ITU-T Recommendation Q.703 (03/93)*
  Specification of Signaling System No. 7
- *ANSI T1.408-1990*
- *ITU-T Recommendation G.704*
- *IEEE Standard 1149.1-1990*
- *Brooktree Bt8370 Specification*

8.2 Numeric Notation

The general representation for numbers is listed in Table 8-1. The suffix can be dropped for clarity when the context makes the intended radix obvious. The suffix convention requires letters within hexadecimal numbers to be capitalized [ABCDEF].

Table 8-1. Number Representation

Type	Suffix	Example
Binary	b	01b, 1010b
Decimal	d	01d, 999d
Octal	o	01o, 174o
Hexadecimal	h	01h, 08E002FCh

8.3 Bit Stream Transmission Convention

Digitized voice transmission represented by octets (8-bit fields) generally numbers bits left to right, 0–7, respectively. Data is transmitted serially starting at the most significant bit (left-most bit, numbered bit 0) and proceeding to the right-most bit (least significant bit, numbered bit 8) of the sample. The receiver receives the most significant bit first. Table 8-2 illustrates this sequence.

Table 8-2. Digitized Voice Transmission Convention

Bit	0 (MSB)	1	2	3	4	5	6	7 (LSB)
Data	1	0	1	0	1	1	1	1
Transmission Order	Bit Stream = 10101111..... MSB is transmitted first							

Digital data transmission uses n-bit words and generally numbers bits right to left, 0 to [n-1], respectively. Data is transmitted serially starting with the least significant bit (right-most bit or bit 0) and proceeding to the most significant bit (left-most bit or bit [n – 1]). The receiver receives the least significant bit first. Table 8-3 illustrates this sequence.

Table 8-3. Digital Data Transmission Convention

Bit	7 (MSB)	6	5	4	3	2	1	0 (LSB)
Data	1	0	1	0	1	1	1	1
Transmission Order	Bit Stream = 11110101..... LSB is transmitted first							

MUSYCC employs the digital data transmission convention. MUSYCC extensively uses 32-bit wide dwords or double-words data transfers. The data is transmitted and received as listed in Table 8-4.

Table 8-4. MUSYCC Byte Transmission Convention

Byte	3 MSB Bits 7 <- 0	2	1	0 LSB Bits 7 <- 0
Transmission and Reception Order	Byte Stream = Byte 0, Byte 1, Byte 2, Byte 3.....			

8.4 Bit Stream Storage Convention

MUSYCC stores and retrieves bit stream data to and from memory using little endian-style byte ordering, which causes the least significant byte to be stored in and retrieved from the lowest memory address.

Tables 8-5 and 8-6 list little and big endian byte ordering within a dword using the 32-bit dword 7654321h.

Table 8-5. Little-Endian Storage Convention (Intel-style)

Address				Data			
x	x + 1	x + 2	x + 3	10h	32h	54h	76h

Table 8-6. Big-Endian Storage Convention (Motorola-style)

Address				Data			
x	x + 1	x + 2	x + 3	76h	54h	32h	10h

8.5 Acronyms

ABT	Abort Termination error
ALIGN	Alignment error
BLP	Bit Level Processor
BPS (bps)	Bits Per Second
BUFF	Buffer error
CHABT	Change to Abort Code
CHIC	Change to Idle Code
CMOS	Complementary Metal Oxide Semiconductor
COFA	Change of Frame Alignment
CRC	Cyclic Redundancy Check
DMA	Direct Memory Access
DMAC	Direct Memory Access Controller
DMI	Digital Multiplexed Interface
dword	Double Word
DXI	Data Exchange Interface
EBUS	Expansion Bus
EOB	End of Buffer
EOM	End of Message
EOP	End of Pad Fill
FCS	Frame Check Sequence
FIFO	First In First Out
FRAD	Frame Relay Access Devices
FREC	Frame Recovery
HDLC	High-Level Data Link Control
IC	Idle Code
INTC	Interrupt Controller
IRAM	Internal RAM
ISDN	Integrated Service Digital Network

ISLP	Inter-System Link Protocol
ISO	International Standard for Organization
JTAG	Joint Test Action Group
kbps	Kilobits per Second
LNG	Long Message Error
LSB	Least Significant Bit or Byte
MB	Megabyte
MBPS (Mbps)	Megabits Per Second
MSB	Most Significant Bit or Byte
MUSYCC	Multichannel Synchronous Communication Controller
ONR	Host Ownership of Buffer
OOF	Out of Frame
OSI	Open System Interconnection
PCI	Peripheral Component Interface
PCM	Pulse Code Modulated
PQFP	Plastic Quad Flat Pack
ROOF	Receiver Out of Frame
RX (Rx)	Receive/Receiver
SDEC	SUERM Octet Count Decrement
SERI	Serial Port Interfaces
SFILT	SS7 Filtered Message
SHT	Short Message error
SINC	SUERM Octet Count Increment
SS7	Signaling System 7
SUERM	Signal Unit Error Rate Monitor
SUERR	Signal Unit Error Rate Interrupt
SUET	Signal Unit Error Threshold
TS	Time Slot
TX (Tx)	Transmit/Transmitter

8.6 Definitions

bit field Any group of associated information bits that must always be viewed together to provide the desired information. For example, in a 3-bit field, the 3 bits can represent 8 related values and thus must always be viewed together.

byte A field made up of 8 binary bits.

channel A logical bit stream through MUSYCC. A channel has an associated transmit and receive direction. The transmit direction is for the bit stream flowing from shared memory towards the serial port. The receive direction is for the bit stream flowing from the serial port to the shared memory. A channel within MUSYCC is bidirectional. The rate of data flow is configurable and is specified in bits per second.

channelized A serial port configuration whereby a higher speed bit stream is partitioned into lower speed bit streams or time slots. A frame synchronization signal is required and allows mapping of individual bits within the time slots into logical channels. This term is synonymous with PCM Highway.

channel group MUSYCC is designed around four independent and full-duplexed sets of channels. Each channel group supports up to 32 logical channels.

data buffer A block of shared memory where data messages are stored. As messages are received from the serial port, MUSYCC writes the message to shared memory data buffers. As messages are sent out on the serial port, MUSYCC takes messages from shared memory data buffers.

descriptor A data structure used to specify attributes of a separate block of data.

dword A field consisting of 32 binary bits, or 2 words concatenated, or 4 bytes concatenated.

FIFO A region of memory designed to facilitate the movement of bits of information in a first-in-first-out manner.

flag As defined by HDLC protocol, an octet with the value 7Eh (01111110b).

frame In the context of an HDLC bit stream, this term is synonymous with message and packet. In terms of a serial interface, a frame is a grouping of synchronous bits relative to a serial line clock and delimited by a synchronization signal. The frame structure is defined by the physical interface providing the framed data.

hyperchannel Concatenation of time slots into a single logical channel. The available bandwidth for such a logical channel is the sum of bandwidth of each time slot.

idle code An octet pattern used to fill the time between the closing flag of one message and the opening flag of the subsequent message. The following idles codes are supported: 7Eh, FFh, and 00h.

message In the context of an HDLC protocol, a data message consists of a header field, an address field, a control field, a payload field, and an FCS field delimited by an opening and a closing flag—7Eh (01111110h). This term is synonymous with frame and packet.

octet A field made up of 8 binary bits. Synonymous with byte.

pointer A 32-bit field containing the address of another bit field, descriptor, dword, word, or byte.

subchannel When a 64 kbps time slot (or channel) consists of lower rate bit streams (in multiples of 8 kbps), each bit stream is said to be a subchannel of the original channel.

time slot An 8-bit portion of a channelized T1 or E1 frame which repeats every 125 µs and represents a 64 kbps signal. In channelized T1 and E1 frames, 24 and 32 time slots operate at 64 kbps.

word A field made up of 16 binary bits or 2 bytes concatenated.

8.7 Revision History

Table 8-7. CN8478 Data Sheet Revisions

Revision	Date	Change	Description
Rev. A	01/25/99	—	Initial Release.
Rev. B	—	03/12/99	Corrected grammar. Added BGA mechanical diagram and package thermal specification.
Rev. C	—	07/29/99	Added BGA pinout, further defined pinout description, replaced BGA mechanical diagram, and updated package thermal specification.
Rev. D	—	3/27/00	–Added 66 MHz PCI clock rate information. –Removed Protected Memory function (Section 5.2.2.4). –Removed POLLTH = 0 option. –Added Section 6.2.6.1. and Section 6.2.6.2. –Removed PCI Bus Utilization function (previously Section 4.7.2). –Added details on interrupt processing at the end of Section 5.2.5.2. –Changed MCENBL to 1 in Section 6.3.5. –Changed TRST* to pulled low in normal operation.

Appendix A JTAG Interface

The CN8478 supports boundary scan testing conforming to IEEE standard 1149.1a-1993 and supplement 1149.1b. 1994. This appendix is intended to assist the customer in developing boundary scan tests for printed circuit boards and systems that use the CN8478. It is assumed that the reader is familiar with boundary scan terminology. For the latest version of the Boundary Scan Description Language (BSDL) file, contact Technical Publications.

The JTAG Interface section of the CN8478 provides access to all external I/O signals of the device for board and system level testing. This circuitry also conforms to IEEE std 1149.1a-1993.

A.1 Instruction Register

The Instruction register (IR) is a 3-bit register. When the boundary scan circuitry is reset, the IR is loaded with the BYPASS Instruction. The Capture-IR binary value is 001.

The eight instructions include three IEEE 1149.1 mandatory public instructions (BYPASS, EXTEST, and SAMPLE/PRELOAD) and five private instructions for manufacturing use only. Bit 0 (LSB) is shifted into instruction register first.

Table A-1. IEEE Std. 1149.1 Instructions

Bit 2	Bit 1	Bit 0	Instruction	Register Accessed
0	0	0	EXTEST	Boundary Scan
0	0	1	SAMPLE/PRELOAD	Boundary Scan
0	1	0	Private	—
0	1	1	Private	—
1	0	0	Private	—
1	0	1	Private	—
1	1	0	Private	—
1	1	1	BYPASS	Bypass

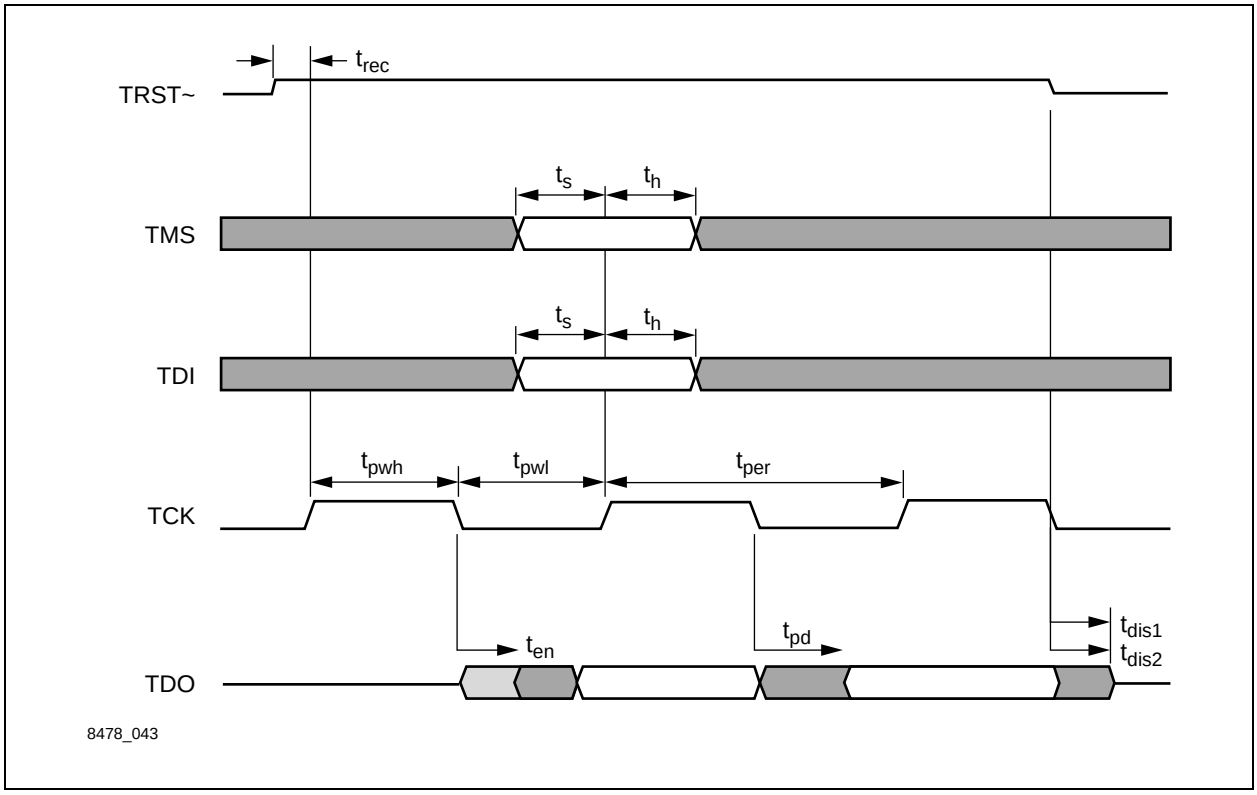
A.2 BYPASS Register

The BYPASS register is a 1-bit shift register that passes TDI data to TDO, which facilitates testing other devices in the scan path without having to shift the data patterns through the compare Boundary Scan register of the CN8478.

Table A-2. JTAG Timing Table

Label	Description	Min	Max	Unit
t_{per}	Period, TCK	—	100	ns
t_{pwl}	Pulse Width Low, TCK	0.4 t_{per}	0.6 t_{per}	ns
t_{pwh}	Pulse Width High, TCK	0.4 t_{per}	0.6 t_{per}	ns
t_{rec}	Recovery, the rising edge of TCK from the rising edge of TRST~	100	—	ns
t_s	Setup, TMS, TDI to the rising edge of TCK	15	—	ns
t_h	Hold, TMS, TDI from the rising edge of TCK	15	—	ns
t_{en}	Enable, TDO from the falling edge of TCK	—	15	ns
t_{pd}	Propagation Delay, TDO from the falling edge of TCK	—	15	ns
t_{dis1}	Disable, TDO from the falling edge of TCK	—	15	ns
t_{dis2}	Disable, TDO from the falling edge of TRST~	—	100	ns

Figure A-1. JTAG Timing Diagram





Further Information
literature@conexant.com
(800) 854-8099 (North America)
(949) 483-6996 (International)
Printed in USA

World Headquarters
Conexant Systems, Inc.
4311 Jamboree Road
Newport Beach, CA
92660-3007
Phone: (949) 483-4600
Fax 1: (949) 483-4078
Fax 2: (949) 483-4391

Americas

U.S. Northwest/
Pacific Northwest – Santa Clara
Phone: (408) 249-9696
Fax: (408) 249-7113

U.S. Southwest – Los Angeles
Phone: (805) 376-0559
Fax: (805) 376-8180

U.S. Southwest – Orange County
Phone: (949) 483-9119
Fax: (949) 483-9090

U.S. Southwest – San Diego
Phone: (858) 713-3374
Fax: (858) 713-4001

U.S. North Central – Illinois
Phone: (630) 773-3454
Fax: (630) 773-3907

U.S. South Central – Texas
Phone: (972) 733-0723
Fax: (972) 407-0639

U.S. Northeast – Massachusetts
Phone: (978) 367-3200
Fax: (978) 256-6868

U.S. Southeast – North Carolina
Phone: (919) 858-9110
Fax: (919) 858-8669

U.S. Southeast – Florida/
South America
Phone: (727) 799-8406
Fax: (727) 799-8306

U.S. Mid-Atlantic – Pennsylvania
Phone: (215) 244-6784
Fax: (215) 244-9292

Canada – Ontario
Phone: (613) 271-2358
Fax: (613) 271-2359

Europe

Europe Central – Germany
Phone: +49 89 829-1320
Fax: +49 89 834-2734

Europe North – England
Phone: +44 1344 486444
Fax: +44 1344 486555

Europe – Israel/Greece
Phone: +972 9 9524000
Fax: +972 9 9573732

Europe South – France
Phone: +33 1 41 44 36 51
Fax: +33 1 41 44 36 90

Europe Mediterranean – Italy
Phone: +39 02 93179911
Fax: +39 02 93179913

Europe – Sweden
Phone: +46 (0) 8 5091 4319
Fax: +46 (0) 8 590 041 10

Europe – Finland
Phone: +358 (0) 9 85 666 435
Fax: +358 (0) 9 85 666 220

Asia – Pacific

Taiwan
Phone: (886-2) 2-720-0282
Fax: (886-2) 2-757-6760

Australia
Phone: (61-2) 9869 4088
Fax: (61-2) 9869 4077

China – Central
Phone: 86-21-6361-2515
Fax: 86-21-6361-2516

China – South
Phone: (852) 2 827-0181
Fax: (852) 2 827-6488

China – South (Satellite)
Phone: (86) 755-5182495

China – North
Phone: (86-10) 8529-9777
Fax: (86-10) 8529-9778

India
Phone: (91-11) 692-4789
Fax: (91-11) 692-4712

Korea
Phone: (82-2) 565-2880
Fax: (82-2) 565-1440

Korea (Satellite)
Phone: (82-53) 745-2880
Fax: (82-53) 745-1440

Singapore
Phone: (65) 737 7355
Fax: (65) 737 9077

Japan
Phone: (81-3) 5371 1520
Fax: (81-3) 5371 1501