AD2420(W)/6(W)/7(W)/8(W)/9(W) Automotive Audio Bus A^{2}B Transceiver Technical Reference

Revision 1.1, October 2019

Part Number
82-100138-01

,

ANALOG DEVICES

Copyright Information

Printed in the USA.

Disclaimer

Analog Devices, Inc. reserves the right to change this product without prior notice. Information furnished by Analog Devices is believed to be accurate and reliable. However, no responsibility is assumed by Analog Devices for its use; nor for any infringement of patents or other rights of third parties which may result from its use. No license is granted by implication or otherwise under the patent rights of Analog Devices, Inc.

Trademark and Service Mark Notice

The Analog Devices logo, A^{2}B, Blackfin, Blackfin+, CrossCore, EngineerZone, EZ-Board, EZ-KIT, EZ-KIT Lite, EZ-Extender, SHARC, SHARC+, and VisualDSP++ are registered trademarks of Analog Devices, Inc.

EZ-KIT Mini and SigmaStudio are trademarks of Analog Devices, Inc.

All other brand and product names are trademarks or service marks of their respective owners.

Contents

A²B Event Control

A²B System Debug

Register Summary

AD2428 A2B Register Descriptions

1 Preface

Thank you for purchasing and developing systems using an Automotive Audio Bus A^{2}B® Transceiver from Analog Devices.

Purpose of This Manual

The *AD2420(W)/6(W)/7(W)8(W)/9(W) Automotive Audio Bus A^{2}B Transceiver Technical Reference* provides information about the transceivers, including register and bit descriptions. For timing, electrical, and package specifications, see the *AD2420(W)/6(W)/7(W)8(W)/9(W) Automotive Audio Bus A^{2}B Transceiver Data Sheet*.

Intended Audience

This manual is intended for system designers and programmers who want to develop systems using the A^{2}B transceiver.

Manual Contents

This manual consists of the following chapters:

- ***A^{2}B Overview*** - Provides a basic description and the features supported.
- ***A^{2}B Operation and Configuration*** - Provides information on bringing up the master node and discovery of the slave nodes. Provides a simple System Discovery Example.
- ***A^{2}B Event Control*** - Provides information on system interrupts and their use.
- ***A^{2}B System Debug*** - Provides information that allows you to perform system diagnostics in order to isolate and correct faults. Additionally, a loop back test mode provides easy validation of I^{2}S/TDM connectivity in master and slave nodes.
- ***Register Summary*** - Provides the register map and bit definitions for the integrated transceiver.
- ***Register Descriptions*** - Provides the detailed descriptions of the registers and bits.
- ***Appendix A: Additional Discovery Flow Examples***

- *Appendix B: Response Cycle Formula*
- *Appendix C: Module ID and Module Configuration Memory*
- *Appendix D: Interrupt Processing*

What's New in This Manual

This revision (1.1) is the second released version of the document. The title changed to a Technical Reference and to reflect the addition of new processor models. The following changes were made to content in this revision:

- Updated content in topics: Transceiver I^2C Accesses, Transceiver Power-On and Reset, I^2C Port Programming Concepts, Direct I^2C Register Accesses, Remote I^2C Register Accesses, Peripheral I^2C Accesses, Master Bring-Up and Operation, Slave Bring-Up and Operation, Clock Sustain Functionality, Slave Node Response Cycles, I^2C Interface
- Renamed and rearranged topics in the A^2B Configuration chapter. Combined previous topics in the chapter into more comprehensive and inclusive topics.
- Added Reset and Operating States topic
- Updated figures: Communication System Block Diagram, Transceiver State Diagram, Simplified A^2B System with Four Nodes
- Updated tables: Bus Latencies for I^2C Access, Transceiver Delay
- Updated notes in topics: A^2B Slot, Transceiver I^2C Access Latencies, Reset and Operating States
- Change register information: TESTMODE.RXDEPTH is now public, changed PINCFG.DRVSTR bit description, changed PDMCTL2.PDMDEST bit and enum description, made PINCFG.TXBLP and PINCFG.TXALP bits private.
- Removed Appendix E: CRC Calculation

Register Documentation Conventions

The register sections and diagrams use the following conventions:

- Registers are presented in address order.
- The reset value appears in binary in the individual bits and in hexadecimal to the left of the register.
- Shaded bits are reserved.

NOTE: To ensure upward compatibility with future implementations, write back the value that is read for reserved bits in a register, unless otherwise specified.

Register description tables use the following conventions:

- Each bit's or bit field's access type appears beneath the bit number in the table in the form (read-access/write-access). The access types include:
  - R= read, RC= read clear, RS= read set, R0= read zero, R1= read one, Rx= read undefined
  - W= write, NW= no write, W1C= write one to clear, W1S= write one to set, W0C= write zero to clear, W0S= write zero to set, WS= write to set, WC = write to clear, W1A= write one action, XCVRA/B= transceiver (port A/ port B)
- Many bit and bit field descriptions include enumerations, identifying bit values and related functionality. Unless otherwise indicated (with a prefix), these enumerations are decimal values.

2 A2B Overview

The Automotive Audio Bus (A2B ®) connects multichannel I2S synchronous PCM data over a distance of up to 15m between nodes. It also extends the synchronous, time-division multiplexed (TDM) nature of I2S to a system that connects multiple nodes, where each node can consume data, provide data, or both.

The transceivers support these A2B functions with a direct interface to general-purpose digital signal processors (DSPs), field programmable gate arrays (FPGAs), application specific integrated circuits (ASICs), microphones, analog-to-digital converters (ADCs), digital-to-analog converters (DACs), and codecs through a multichannel I2S/TDM interface. They also provide a PDM interface for direct connection of up to four PDM digital microphones. Enabling the A2B bus-powering (phantom powering) feature supplies voltage and current to the slave nodes over the same, daisy-chained, twisted pair wire cable as used for the communication link. The transceiver also fully supports I2C communication over the A2B link.

The transceivers have the following features.

- Line topology
  - Single master, multiple slave
  - Unshielded, single twisted pair wire (UTP) cable link between nodes (cable length is specified in the product data sheet)
- Communication over distance
  - Synchronous data
    - Multichannel I2S/TDM to I2S/TDM interface
    - Synchronous, phase-aligned clock in all nodes
    - Low-latency slave-to-slave communication
  - I2C to I2C control and status information
  - GPIO over distance
- Configurable with SigmaStudio™ graphical development tool
- Qualified for automotive applications

- Configurable as A2B bus master or slave
- I2C interface
- 8-bit to 32-bit multichannel I2S/TDM interface
  - I2S/TDM/PDM programmable data rate
  - Up to 32 channels (1 x TDM32 or 2 x TDM16), mapped to up to 32 upstream and 32 downstream A2B bus slots
- PDM inputs supporting up to 4 high-dynamic-range microphones
- Unique ID register for each transceiver
- Support for crossover or straight-through cabling
- Programmable settings to optimize EMC performance

A2B Terminology

To make the best use of the A2B system, it is helpful to understand the following terms.

A-Side or A-Port

A2B transceiver interface that faces toward the master (toward the immediately upstream node).

B-Side or B-Port

A2B transceiver interface that faces toward the last-in-line slave (toward the immediately downstream next-in-line slave).

Bus Link

The A2B bus can consist of multiple daisy-chained slave nodes connected to a single master node. The physical connection between a master and slave 0, as well as all physical A2B connections between slaves, are called bus links. An unshielded twisted wire pair is typically used for each bus link.

Data Channel

A data channel carries the synchronous I2S/TDM data for a single sensor/actuator (for example, an ADC, a microphone, or a speaker). The I2S/TDM interface uses equally sized data channels, where the width of the data word is often smaller than the width of the I2S/TDM data channel. The I2S/TDM interface of the transceiver supports programmable data channel lengths of 16 or 32 bits.

Data Slot

A synchronous data word of a single sensor/actuator (for example, an ADC, a microphone, or a speaker), as mapped onto the A2B bus.

Downstream

Communication flow from the master node toward the slave nodes, terminating at the last-in-line slave.

Host

Processor that programs the master transceiver. The host is also the source for the synchronous clock on the A2B bus. The clock signal (BCLK) is part of the I2S/TDM interface between the host and master.

I 2 S/TDM

The inter IC sound (I2S) bus carries pulse code modulated (PCM) information between audio chips on a PCB. The I2S/TDM interface extends the I2S stereo (2-channel) content to multiple channels using time-division multiplexing (TDM).

Local Power

Slave nodes that do not operate on A2B bus power use local power, which is sourced by extra wires.

LVDS

Low voltage differential signaling.

Master Node

Originator of the clock (derived from the I2S input), downstream data, network control, and power. The master node is comprised of the host processor and an A2B master transceiver, which receives payloads from the host and sends payloads to the host.

PDM

Pulse Density Modulation (PDM) is used in sigma delta converters. PDM format represents an over-sampled 1-bit sigma delta ADC signal before decimation and is often used as the output format in digital microphones.

Phantom Power

Slave nodes can tap into the bias voltage on the A2B bus link and use it as the sole power supply. Such A2B bus-powered slave nodes are considered to be "phantom-powered".

PRBS

Pseudo random binary sequence.

Preamble

Synchronization bits to signal the start of a control or response frame. The downstream control frame preamble is sent by the master for every superframe. Slave transceivers synchronize to the downstream control preamble and generate a local, phase-aligned master clock from it.

Response Time

Specifies the time a last node waits after the start of a superframe before the node responds with the Synchronization Response Frame (SRF). Response time is programmed in the master and all slaves closest to the master so that these nodes know when to expect the direction to switch from downstream to upstream.

Slave Node

Addressable network connection point. Slave nodes can be the source and/or destination of both downstream and upstream data slots. Every A2B slave node has an A2B slave transceiver.

Synchronization Control and Response Frames (SCF/SRF)

Control frame for nodes (control header) and response frame from nodes (response header). Headers include a preamble for synchronization and enable read and write access to all nodes.

Synchronous Data

Data streamed continuously (for example, audio signals) with a fixed time interval (selectable between 44.1 kHz or 48 kHz) between two successive transmissions to and from the same node.

Superframe

The overall frame structure for A2B. It starts with an SCF, includes optional data slots, and concludes with an SRF. Superframes repeat every 1024 bus clock cycles.

Upstream

Communication flow the last-in-line slave node to the master node.

A2B Bus Details

The ***Communication System Block Diagram*** shows an A2B communications system, which is a single-master, multiple-slave system where the master transceiver is controlled by the host. The host generates a periodic synchronization

signal (SYNC) on the I²S/TDM interface at a fixed frequency (selectable between 44.1 kHz and 48 kHz), to which all A²B nodes synchronize. Communication over the A²B bus occurs in periodic superframes at this rate. Data is transferred at the A²B system bit clock (SYSBCLK) rate, which is 1024 times faster than the superframe rate (49.152 MHz for a frame rate of 48 kHz, 45.158 MHz for a frame rate of 44.1 kHz). Each superframe is divided into periods of downstream transmission, upstream transmission, and no transmission (where the bus is not driven).

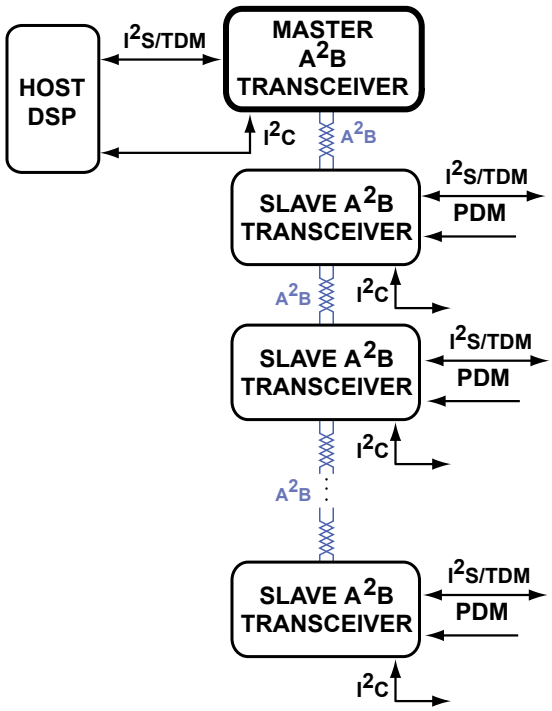

Figure 2-1: Communication System Block Diagram

The ***A²B Superframe*** figure shows a superframe with an initial period of downstream transmission and a later period of upstream transmission.

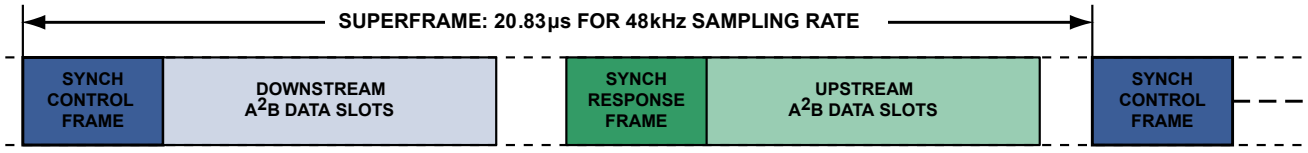

Figure 2-2: A²B Superframe

All signals on the A²B bus are line-coded, and the master node forwards the synchronization signal downstream to the last slave node in the form of a synchronization preamble. This preamble is followed by the control frame (SCF). Downstream, TDM synchronous data is added directly after the control frame. Every slave can use or consume some of the downstream data and add data for downstream nodes. The last slave node responds after the response time with a response frame (SRF). Upstream synchronous data is added by each node directly after the response frame. Each node can also use or consume upstream data. All synchronous data is organized into data slots of equal width, though the upstream and downstream slot widths can be different. For more details, see A²B Slot Format.

The embedded control and response frames allow the host to individually address each slave node over the A²B bus. In a similar fashion, the host can also access remote peripheral devices that are connected to any discovered slave transceivers using I²C- to-I²C communication over distance.

All nodes in an A²B system are sampled synchronously in the same A²B superframe. Synchronous I²S/TDM downstream data from the master arrives at all slaves in the same A²B superframe, and each node's upstream audio data arrives synchronously in the same I²S/TDM frame at the master. The remaining audio phase differences between slaves can be compensated for by register-programmable fine adjustment of the SYNC pin signal delay using the A2B_SYNCOFFSET register.

Because data is received and transmitted over the I²S/TDM port every sample period, there is a delay incurred for data moving between the A²B bus and the I²S/TDM interfaces. The timing relationship between samples over the A²B bus is shown in the ***A²B Bus Synchronous Data Exchange*** figure.

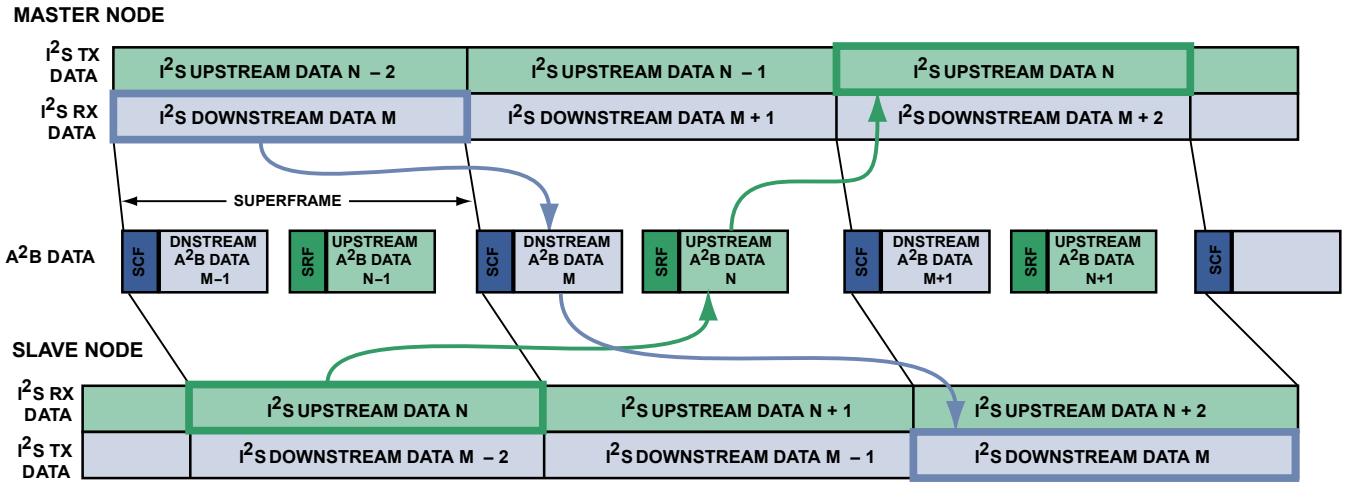

Figure 2-3: A²B Bus Synchronous Data Exchange

Note in the ***A²B Bus Synchronous Data Exchange*** figure, both downstream and upstream samples are named for the superframe where they enter the A²B system, as follows:

- Data transmitted by the master node transceiver in superframe M creates downstream data M
- Data transmitted by the slave node transceivers in superframe N creates upstream data N
- Data received over the I²S/TDM interface by the A²B transceiver chip is transmitted over the A²B bus in the following superframe
- Data on the A²B bus is transmitted over the I²S/TDM interface of an A²B chip transceiver in the following superframe
- Data transmitted across the A²B bus (master to slave or slave to master) has two superframes of latency, plus any internal delay that has accumulated in the transceiver chips, as well as delays due to wire length. Therefore,

overall latency is slightly over two superframes from the I²S/TDM interface in one A²B transceiver chip to the I²S/TDM interface of another A²B transceiver chip.

Functional Description

The A²B transceiver connects multichannel I²S (inter-IC sound) synchronous, pulse-code modulated (PCM) data over a distance between nodes (the cable length is specified in the product data sheet). It also extends the synchronous, time-division multiplexed nature of I²S to a system that connects multiple nodes, where each node can consume data, provide data, or both.

The A²B transceiver supports these A²B functions with a direct interface to general-purpose DSPs, FPGAs, ASICs, microphones, ADCs, DACs, and codecs through a multichannel I²S/TDM interface. The data over the A²B bus link is manchester encoded. The transceiver also fully supports I²C communication over the A²B link. The A²B transceiver can be used in either a slave node or in a master node. By default, the transceiver starts up as a slave transceiver but can be configured as a master transceiver if the host sets the `A2B_CONTROL.MSTR` bit.

The ***Simplified A²B System with Four Nodes*** figure shows a simple A²B system example. The host programs registers in each of the nodes to control the data traffic on the A²B bus. Microphone data from slave nodes 0 and 2 is delivered to the host, and speaker data for slave nodes 1 and 2 is delivered from the host to the DACs.

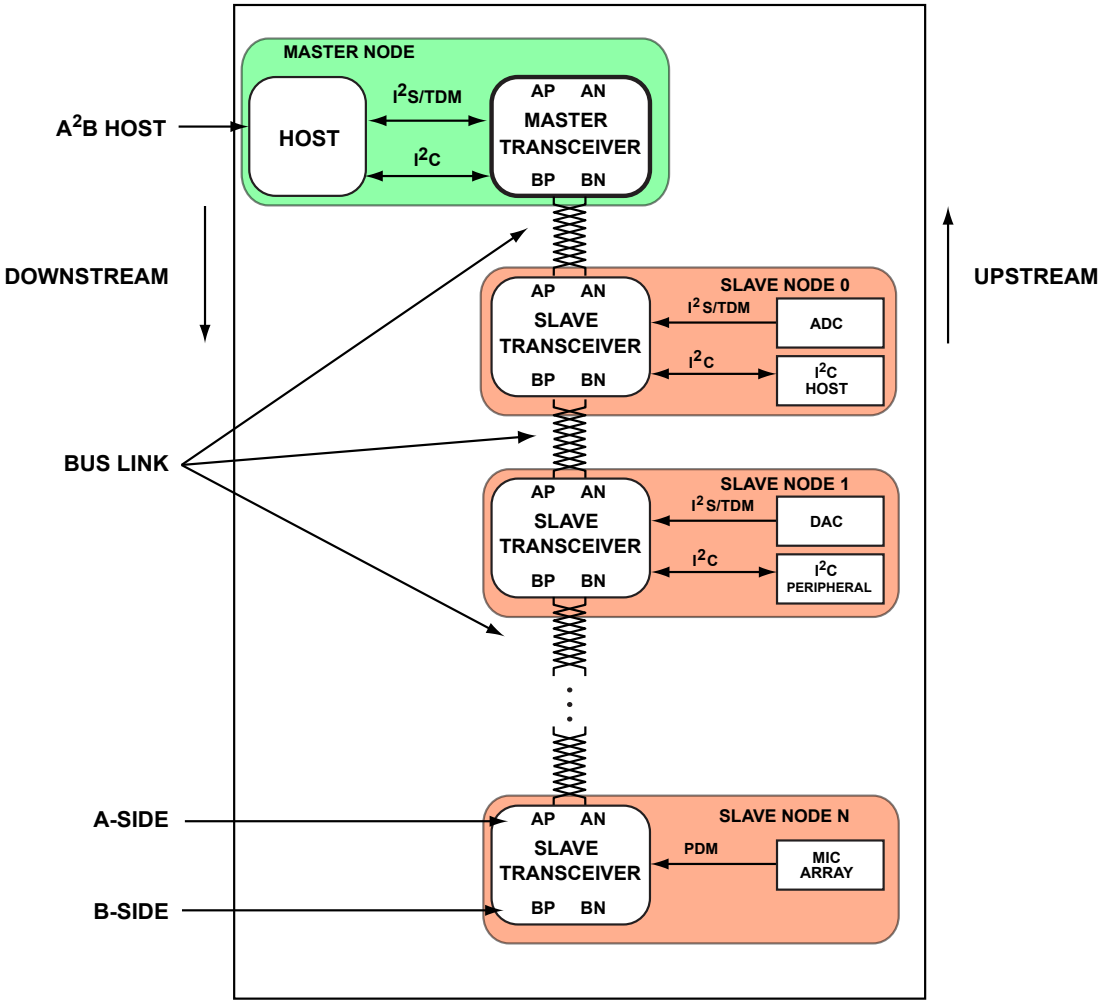

Figure 2-4: Simplified A^{2}B System with Four Nodes

Architectural Concepts

The following sections provide information that describes the hardware blocks, interfaces and interconnections.

I^{2}C Interface

The I^{2}C interface is used to directly access the transceiver register space from a locally connected host and to remotely exchange I^{2}C data over the A^{2}B bus between the master transceiver and any discovered slave node in the system. This protocol is referred to as ***I^{2}C over distance***, where the exchanged I^{2}C data is embedded within the synchronization control frame (downstream, from the master to the targeted slave) and the synchronization response frame (upstream, from the targeted slave to the master).

The I^{2}C interface in the transceiver is compatible with up to 5 V logic levels and has the following features:

- Slave only operation in an A^{2}B master node

- Master, multi-master, or slave operation in an A^{2}B slave node
- Operations at 100k or 400k bits/s rate, as configured by the A2B_I2CCFG.DATARATE bit
- 7-bit addressing
- Clock stretching

NOTE: The A^{2}B host on the master node must support I^{2}C clock stretching in order to interface to the master transceiver.

A transceiver that is configured as a master recognizes two I^{2}C device addresses:

- BASE_ADDR - for direct accesses via the I^{2}C port to its register space
- BUS_ADDR - for remote access to slave node registers and slave node I^{2}C peripherals over the A^{2}B bus using the ***I^{2}C over distance*** protocol

The I^{2}C BASE_ADDR is set by the logic levels on the ADR2/IO2 and ADR1/IO1 pins at power-on reset, thus providing support for up to four master devices connecting to the same I^{2}C bus. The LSB of the 7-bit device address determines whether an I^{2}C data exchange uses the BASE_ADDR (bit 1 = 0) to access the transceiver or BUS_ADDR (bit 1 = 1) to access a bus node through a master-enabled transceiver, as described in the ***I^{2}C Address*** table.

Table 2-1: I^{2}C Device Address

ADR2/IO2 Setting	ADR1/IO1 Setting	BASE_ADDR	BUS_ADDR
0	0	0x68	0x69
0	1	0x6A	0x6B
1	0	0x6C	0x6D
1	1	0x6E	0x6F

A transceiver that is configured as a slave does not recognize BUS_ADDR. On slave transceivers, the I^{2}C interface allows for both I^{2}C master and slave behavior. It is the I^{2}C master when the transceiver receives a remote I^{2}C peripheral access request from the host through the A^{2}B bus. The slave transceiver functioning as the I^{2}C master then forwards the I^{2}C transaction to the I^{2}C slave address programmed in its A2B_CHIP register. It is the I^{2}C slave when the transceiver registers (BASE_ADDR) are accessed by a local external controller through the I^{2}C port.

NOTE: While a local external controller can program the register space of a slave transceiver, the A2B_SWCTL , A2B_RESPCYCS , A2B_SLOTFMT , A2B_DATCTL , A2B_RAISE , and A2B_GENERR registers must be written over the A^{2}B bus by the remote host. A write to any of these registers from the local I^{2}C port has no effect on the register.

The I²C interface on the transceiver allows register programming before PLL lock. Write 1 for action (W1A) bits (for example, A2B_CONTROL.ENDDSC and A2B_CONTROL.NEWSTRCT) have no effect prior to PLL lock since the protocol engine is still in reset.

NOTE: The A2B_SWCTL , A2B_SLOTFMT , A2B_DATCTL , and A2B_DISCVRY registers cannot be written in a master transceiver prior to PLL lock. Writes to these registers before PLL lock is established have no effect.

CAUTION: System software must be designed to avoid simultaneous writes to the same slave register from both the A²B host (through the A²B bus) and the local processor (through the I²C port). When write contention occurs, both writes complete, but the order in which they complete is unpredictable.

I²C Clock Stretching

The transceiver uses the I²C clock stretching feature to ensure that the I²C accesses have enough time to be processed. It is applied mainly for host I²C accesses to slave node transceivers and slave node I²C peripherals over the A²B bus. Clock stretching is initiated by the master in response to host I²C accesses at the following times:

- During write accesses – before the acknowledge bit after each data byte
- During read accesses – before the acknowledge bit following the read request
- During burst read/write accesses of more than one byte – before the first bit of subsequent data bytes

Pulling the SCL signal low indicates to the host that the transceiver needs more time to process the request. Once the transceiver is ready to acknowledge the request, it lets the SCL signal go high so the host can gain back control of the SCL and proceed with the acknowledge (ACK) and the next byte.

IMPORTANT: It is mandatory that the host (I²C master) supports I²C clock stretching in an A²B system design.

When a peripheral in a slave node stretches the I²C clock, the SCL signal is also stretched between the master transceiver and the host. If the SCL signal is not released by the peripheral within the time of 32 superframes, the master registers a timeout (A2B_INTPND2.I2CERR = 1), releases the SCL, and ceases stretching of the host clock. This timeout ensures a slave peripheral cannot bring the I²C interface of the host to a permanent halt.

Transceiver I²C Accesses

The LSB of the 7-bit device address determines whether an I²C data exchange uses the BASE_ADDR (bit 1 = 0) to access the transceiver or BUS_ADDR (bit 1 = 1) to access a bus node through a master configured transceiver, as shown in the following table.

Table 2-2: I²C Device Addresses

Bit Number	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1 (LSB)	Bit 0 (R/$\overline{W}$)
Start bit	1	1	0	1	ADR2/IO2	ADR1/IO1	0 = BASE 1 = BUS	0 = write 1 = read

The A²B transceiver supports the following read and write operations:

- Single-word write operation – the A²B master (I²C slave) issues an acknowledge by pulling SDA low during the ninth clock pulse, thus completing the access.
- Burst mode write sequence – the transceiver automatically increments the register address pointer after each data byte, so sequential data registers can be written without reprogramming the address.
- Single-word read operation – the first read/write (R/$\overline{W}$) bit is 0, indicating a write operation. This is because the register address must still be written to set up the internal address. After the I²C slave acknowledges the receipt of the register address, the I²C master must issue a repeated start command, followed by the chip address byte with the R/$\overline{W}$ bit set to 1 (read). This causes the I²C data line SDA to reverse direction and begin driving data back to the I²C master. The I²C master then responds every ninth pulse with an acknowledge pulse to the slave.
- Burst mode read sequence – the transceiver automatically increments the register address pointer after every read of a data byte, so sequential data registers can be read without reprogramming the address.

Data transfers over the I²C interface require the following steps:

1. A data transfer is initiated by a microcontroller that is connected to an A²B transceiver.
2. The microcontroller establishes a start condition (a high to low transition on SDA while SCL remains high), which indicates that an address/data stream follows.
3. In the next eight SCL cycles, the A²B transceiver receives a 7-bit address and the R/$\overline{W}$ bit from the host (MSB first).
4. The A²B transceiver recognizes the transmitted address and responds by pulling the data line low during the ninth clock pulse (acknowledge bit).

The R/$\overline{W}$ bit determines the direction of the data. When the LSB of the first byte is cleared (=0), the host writes information to the master. When the LSB of the first byte is set (=1), the host reads information from the master. Data transfers take place until a stop condition (when SDA transitions from low to high while SCL is held high) is encountered. The register address pointer auto increments to support burst mode I²C writes and burst mode I²C reads for both master and slaves.

The ***I²C Formats*** figure shows the format of the following I²C operations:

- Writes to BASE_ADDR/BUS_ADDR can contain one or more bytes of data. The first byte after the device address sets the register address in the device. The subsequent byte is written to the addressed register. Since the address pointer increments after each write, sequential registers can be written in a single transaction.
- Reads from BASE_ADDR/BUS_ADDR can contain one or more bytes of data. The device address with write indication is followed by the register address in the device and a repeated device address with a read access indication.

SINGLE WORD I²C WRITE FORMAT

S	BASE ADDR/ BUS ADDR R/$\overline{W}$=0	AS	REG ADDR BYTE	AS	DATA BYTE 1	AS	P

BURST MODE I²C WRITE FORMAT

S	BASE ADDR/ BUS ADDR R/$\overline{W}$=0	AS	REG ADDR BYTE	AS	DATA BYTE 1	AS	DATA BYTE 2	AS	DATA BYTE 3	AS	DATA BYTE 4	AS	...	AS	P

SINGLE WORD I²C READ FORMAT

S	BASE ADDR/ BUS ADDR R/$\overline{W}$=0	AS	REG ADDR BYTE	AS	S	BASE ADDR/ BUS ADDR R/$\overline{W}$=1	AS	DATA BYTE 1	AM	P

BURST MODE I²C READ FORMAT

S	BASE ADDR/ BUS ADDR R/$\overline{W}$=0	AS	REG ADDR BYTE	AS	S	BASE ADDR/ BUS ADDR R/$\overline{W}$=1	AS	DATA BYTE 1	AM	DATA BYTE 2	AM	...	AM	P

S = START BIT
P = STOP BIT
AM = ACKNOWLEDGE BY I²C MASTER
AS = ACKNOWLEDGE BY I²C SLAVE

Figure 2-5: I²C Formats

The first byte after the repeated device address contains the value of the register addressed. The first byte after the device address sets the register address in the device. It is followed by the repeated device address, but with read access indication. The subsequent bytes contain the values of the automatically incremented register addresses.

The *I²C Write Timing* figure shows I²C write timing.

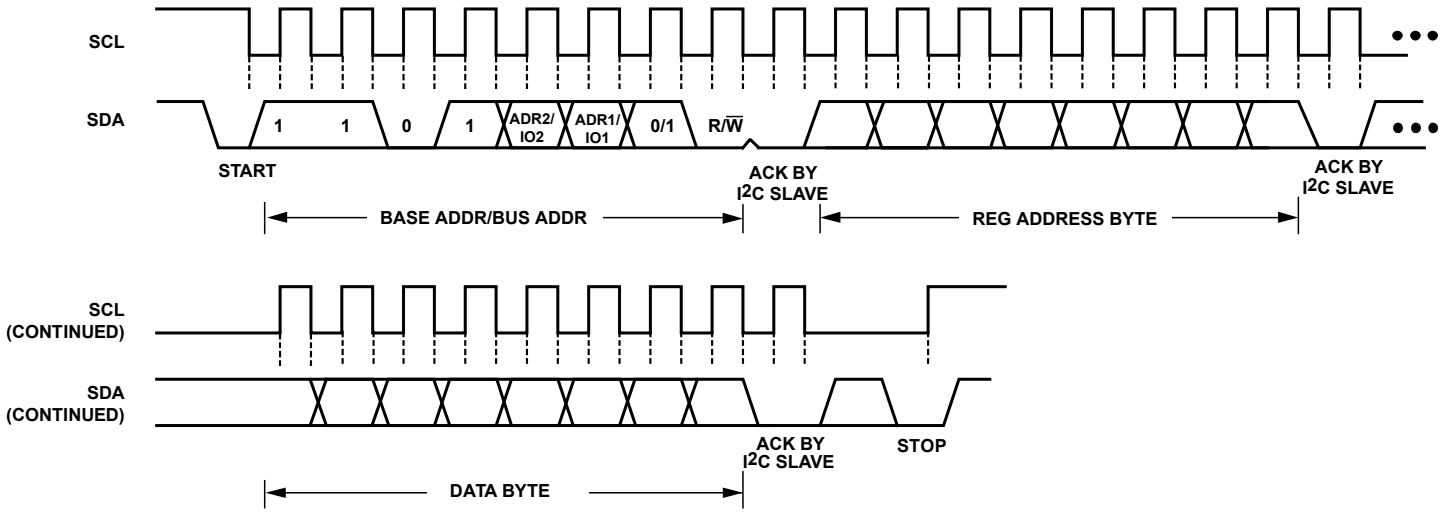

Figure 2-6: I²C Write Timing

The *I²C Read Timing* figure shows I²C read timing.

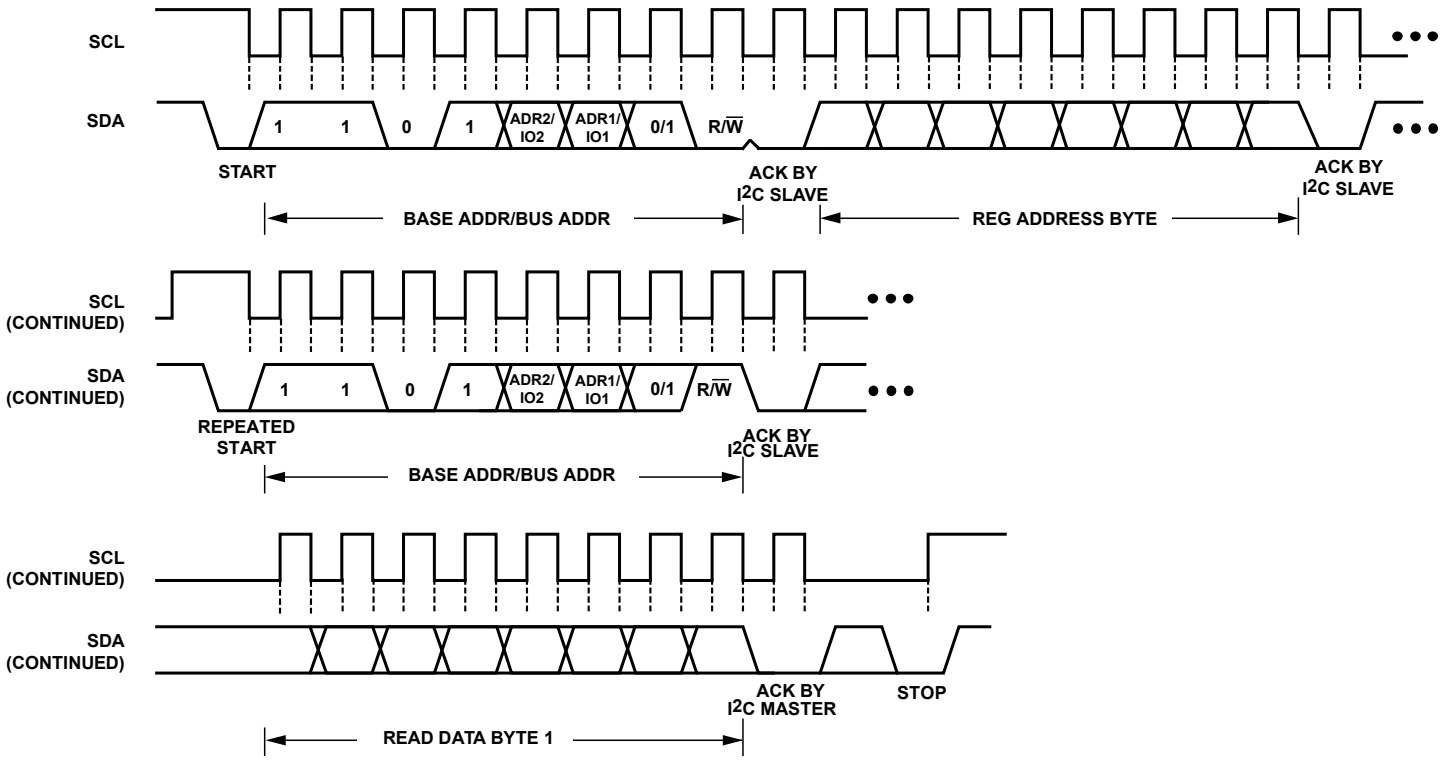

Figure 2-7: I²C Read Timing

Transceiver I²C Access Latencies

When an I²C access is made over distance to a remote transceiver via the A²B bus, there are latencies incurred. A²B bus latencies for different types of I²C accesses are provided in the ***Bus Latencies for I²C Accesses*** table.

Table 2-3: Bus Latencies for I²C Accesses (48 kHz Superframe Rate)

I²C Access Type (Conditions)	**Estimated A²B Bus Latency (µs)**
I²C write of N data bytes to slave transceiver registers (clock stretching enabled via master `A2B_I2CCFG.EACK` = 0)	N × 22
I²C read of N data bytes from slave transceiver registers (clock stretching enabled via master `A2B_I2CCFG.EACK` = 0)	N × 22
I²C write of N >1 data byte to slave transceiver registers (clock stretching disabled via master `A2B_I2CCFG.EACK` = 1, host I²C using 400 kHz data rate)	2
I²C write of N data bytes to slave transceiver registers (clock stretching enabled via master `A2B_I2CCFG.EACK`= 1, host I²C using 100 kHz data rate)	0

Table 2-3: Bus Latencies for I²C Accesses (48 kHz Superframe Rate) (Continued)

I²C Access Type (Conditions)	Estimated A²B Bus Latency (µs)
I²C write of N data bytes to remote I²C peripheral (slave `A2B_I2CCFG.DATARATE` = 0 = 100 kHz)	((N – 1) × 113) + 213
I²C write of N data bytes to remote I²C peripheral (slave `A2B_I2CCFG.DATARATE` = 1 = 400 kHz)	((N – 1) × 45) + 70

For example, consider a case where a remote peripheral (connected to a slave node) register is being written. In the ***I²C Access Latency*** figure, the I²C access latency is marked with green arrows.

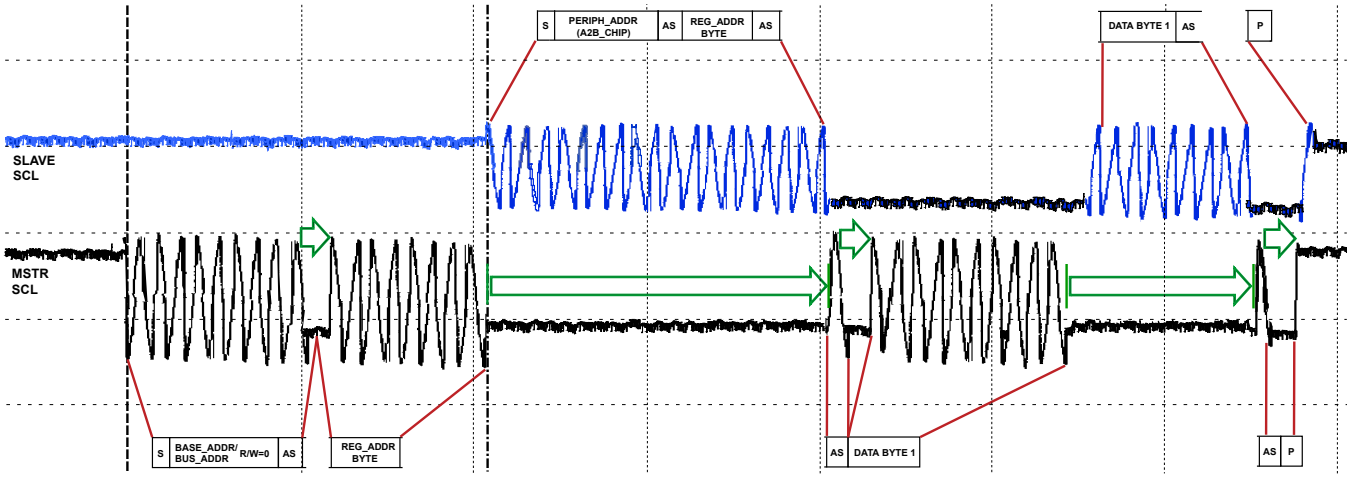

Figure 2-8: I²C Access Latency

NOTE: The latencies described in the ***Bus Latencies for I²C Accesses*** table are for accesses with no conflicts. If an I²C message doesn't get immediately acknowledged or is otherwise held off due to higher-priority events such as a GPIO interrupt, a line fault interrupt, an I²C issue (NACK), etc., the delay incurred before attempting to execute the message exchange is not included in the values provided in the table.

Pulse-Density Modulation Interface (PDM)

Pulse-density modulation is used in sigma delta converters. The PDM format represents an over-sampled 1-bit sigma delta ADC signal before decimation and is often used as the output format in digital microphones.

The PDM block supports high dynamic range microphones with a high signal-to-noise ratio (SNR) and an extended maximum sound pressure level (SPL).

The enhanced PDM block of the transceiver supports a lower noise floor than the AD241x transceiver. This provides for an SNR greater than 120 dB. The PDM block on the transceiver supports 24 kHz and 12 kHz sample rates in addition to a 48 kHz sample rate with the same PDM clock rate (3.072 MHz at a 48 kHz frame rate). The cutoff frequency of the high pass filter in the PDM block on the transceiver is fixed to 1 Hz and is not programmable. The highpass filter is a first order IIR filter.

The transceiver is programmable for 1x, 1/2x, or 1/4x PDM sampling (48 kHz, 24 kHz, or 12 kHz typical) relative to the superframe rate (48 kHz typical). For 1/2x or 1/4x PDM sampling, synchronous data in an A^{2}B slot is duplicated in order to match the superframe rate. Even lower PDM sampling rates are possible when the reduced rate feature of the transceiver is used in combination with this (for example, down to 375 Hz).

The PDM bit clock output frequency from the transceiver is 64x faster than the PDM audio sampling rate (typically, 3.072 MHz for 48 kHz PDM audio sampling).

Each PDM-enabled receive pin can receive up to two channels of audio data (stereo). One of the channels is associated with the rising edge of the clock and the other with the falling edge of the clock.

The PDM block is configured using the PDM control (`A2B_PDMCTL`) register:

- When `A2B_PDMCTL.PDM0EN` = 1, the DRX0/IO5 pin is enabled to receive PDM data, and the BCLK pin is an output, typically producing a 3.072 MHz clock for the TDM2 setting. In this mode, the DRX0/IO5 pin data is not passed to the I^{2}S/TDM port. Similarly, the `A2B_PDMCTL.PDM1EN` bit controls PDM data reception on the DRX1/IO6 pin.
- The `A2B_PDMCTL.PDMxSLOTS` bits select whether the PDM signals on the DRX pins use one (mono) or two (stereo) channels.

PDM Sampling Edge of a Connected Microphone

The pulse-density modulation (PDM) interface allows PDM input from two microphones to be time-multiplexed on a single data line using a single clock.

A PDM microphone encodes data such that the left channel is valid on the falling edge of the clock (CLK) signal and the right channel is valid on the rising edge of the CLK signal. After the DATA signal is driven during the appropriate half phase of the CLK signal, the microphone output is tristated. As such, two microphones (one set to the left channel and the other set to the right channel) can share a single DATA line (see the ***Stereo PDM Format*** figure).

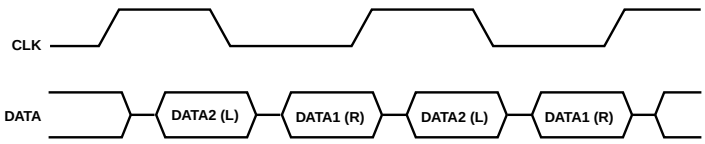

Figure 2-9: Stereo PDM Format

In the transceiver, the PDM block samples the microphone data on all 64 clock edges. The transceiver must be programmed to a TDM mode that produces 64 BCLKs per frame (either the default TDM2/32 or TDM4/16 mode). The TDM settings do not affect the PDM block.

In the transceiver, the data sampled on the rising edge of BCLK is always the first channel. If `A2B_PDMCTL.PDM0SLOTS` = 1 or `A2B_PDMCTL.PDM1SLOTS` = 1, the first slot is associated with the rising edges of BCLK, and the second slot is associated with the falling edges of BCLK.

For example, two microphones are connected to each of the DRX0/IO5 and DRX1/IO6 pins of a slave node with the PDM0 and PDM1 slots configured as 2-slot. In this case, the PDM block samples 64-bit data each frame, converts it to 24-bit PCM data, and drives the converted output as follows:

- Right microphone data is sampled on the DRX0 pin on rising clock edges and driven in the first* transmit slot on the A^{2}B bus.
- Left microphone data is sampled on the DRX0 pin on falling clock edges and driven in the second* transmit slot on the A^{2}B bus.
- Right microphone data is sampled on the DRX1pin on rising clock edges and driven in the third* transmit slot on the A^{2}B bus.
- Left microphone data is sampled on the DRX1 pin on falling clock edges and driven in the fourth* transmit slot on the A^{2}B bus.

  Note that * is the actual slot number, based on the system slot configuration.

NOTE: When using the default A2B_PDMCTL2 settings, PDM pins are always sampled with rising edge data first; therefore, the `A2B_I2SCFG.RXBCLKINV` and `A2B_I2SCFG.TXBCLKINV` clock inversion settings are ignored when the transceiver is configured in PDM mode.

If using the default A2B_PDMCTL2 settings and `A2B_PDMCTL.PDM0SLOTS` = 0 or `A2B_PDMCTL.PDM1SLOTS` = 0, only the right channel data is sampled on the PDM pin. If sampling only left channel data is desired, this can be supported by setting `A2B_PDMCTL.PDM0EN` = `A2B_PDMCTL.PDM0SLOTS` = `A2B_UPOFFSET` = 1.

PDM Enhancements

The default PDM functionality is fully backward-compatible with previous transceiver generations; however, there are several additional features which make the PDM interface more flexible.

PDM Clocking Options

The DRX0 and DRX1 input pins can be configured individually as PDM inputs. When the PDM interface is enabled on an A^{2}B slave node on one or both of the DRX pins, a PDMCLK signal running at 64 × f_{SYNCM} (3.072 MHz at 48 kHz f_{SYNCM}) is required to clock the PDM device. The transceivers allow either the PDMCLK/IO7 or BCLK pin to produce the required PDMCLK. PDMCLK on IO7 can be enabled by setting the `A2B_PDMCTL2.PDMALTCLK` bit.

If PDMCLK/IO7 is used instead of BCLK, the restriction limiting operating to TDM2/32 or TDM4/16 is removed. The BCLK frequency can be set to a different frequency using the I^{2}S/TDM registers. In this case, PDMCLK/IO7 is used to capture PDM input on DRX0/DRX1.

BCLK and PDMCLK/IO7 can also be used concurrently to clock the PDM microphones at the same frequency and phase alignment, but with opposite polarity. This is accomplished by setting the `A2B_PDMCTL2.PDMALTCLK` bit. Additionally, a register controls whether the rising edge data or falling edge data is sampled first:

- When A2B_PDMCTL2.PDM0FFRST = 0 (default), the PDM0 data on DRX0 is sampled rising edge first. When A2B_PDMCTL2.PDM0FFRST = 1, it is sampled falling edge first.
- When A2B_PDMCTL2.PDM1FFRST = 0 (default), the PDM1 data on DRX1 is sampled rising edge first. When A2B_PDMCTL2.PDM1FFRST = 1, it is sampled falling edge first.

NOTE: In a master node, BCLK is always an input; therefore, the clock output to PDM microphones connected to a master tranceiver typically comes from PDMCLK/IO7.

PDM Data Routing Options

The PDM interface can be used on master or slave transceivers. The PDM data received by the transceiver can then be sent to any node(s) on the A^{2}B bus, sent out to the local I^{2}S port, or both. This is done using the A2B_PDMCTL2.PDMDEST field.

Full-Duplex I^{2}S With Four PDM Microphones

If both pins (DRX0 and DRX1) are used to receive PDM data, it is possible to change the function of DTX1 so that it acts as the alternate DRX1, enabling concurrent use of up to four PDM microphones and full-duplex I^{2}S communications. This is accomplished by setting the A2B_I2SGCFG.RXONDTX1 bit.

I^{2}S/TDM Interface

The I^{2}S/TDM serial port operates in full-duplex mode, where both the transmitter and receiver operate simultaneously using the same critical timing bit clock (BCLK) and frame synchronization (SYNC) signals. A^{2}B slave transceivers generate the timing signals on the BCLK and SYNC output pins with frequencies based on the settings in the I^{2}S global configuration register (A2B_I2SGCFG), the I^{2}S rate register (A2B_I2SRATE), and the I^{2}S reduced rate register (A2B_I2SRRATE). A^{2}B master transceivers use the same BCLK and SYNC pins as inputs, which are driven by the host, thus providing the time base for the full A^{2}B bus topology.

Time Division Multiplexing (TDM) Protocol

TDM mode extends an I^{2}S interface to more than a stereo 2-channel (TDM2) signal. When the transceiver is programmed in the A2B_I2SCFG register to support a certain number of TDM channels, this number of TDM channels is available on each enabled I^{2}S/TDM data pin (DTX0 and DTX1 or DRX0 and DRX1). TDM2, TDM4, TDM8, TDM12, TDM16, TDM20, TDM24, and TDM32 modes are supported.

For example, if TDM4 is selected and one transmit pin (DTX0) is enabled, there are four transmit data channels. If TDM4 is selected and both transmit pins (DTX1 and DTX0) are enabled, there are eight transmit data channels, shown in the ***Data Channel Structure for TDM4 Setting*** figure.

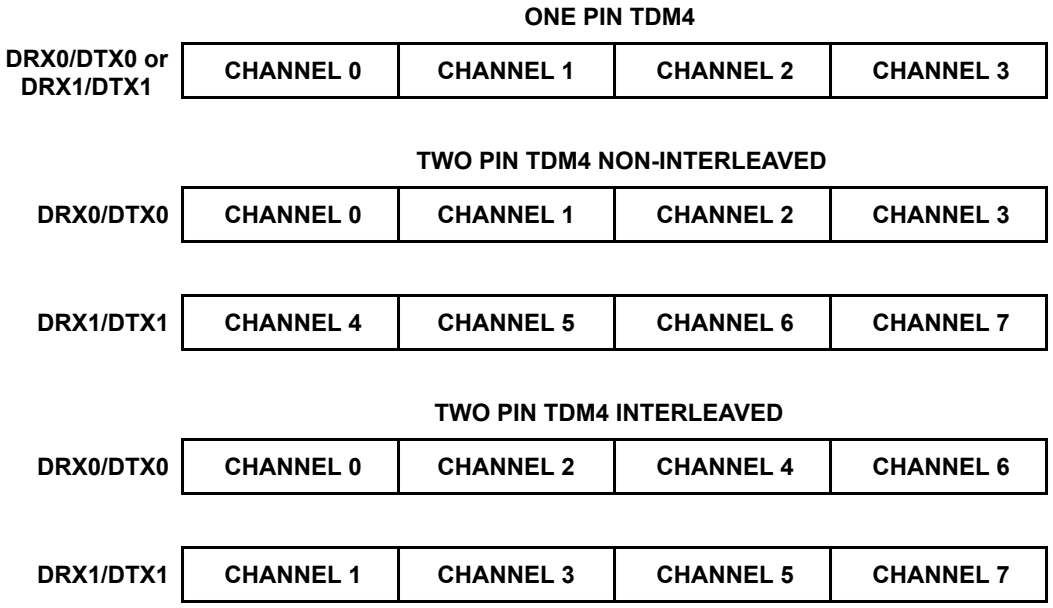

Figure 2-10: Data Channel Structure for TDM4 Setting (TDMMODE == 001)

The I²S/TDM serial port supports data channel widths of 16 bits or 32 bits to carry signals of varying word length. Data words are always represented in MSB first format. The BCLK signal frequencies for different TDM modes are shown in the ***I²S/TDM Clock Frequency Settings for 48 kHz Superframe Rates*** table.

Table 2-4: I²S/TDM Clock Frequency Settings for 48 kHz Superframe Rates

TDM Mode	16-bit TDM Channel Size		32-bit TDM Channel Size	
	Frequency (MHz)	Comments	Frequency (MHz)	Comments
TDM2	1.536		3.072	
TDM4	3.072		6.144	
TDM8	6.144		12.288	
TDM12	9.216	No slave node support	18.432	No slave node support
TDM16	12.288		24.576	
TDM20	15.36	No slave node support	30.72	No slave node support
TDM24	18.432	No slave node support	36.864	No slave node support
TDM32	24.576		49.152	

The DRX0 and DRX1 input pins can be configured individually as PDM inputs. When PDM is enabled on an A²B slave node on one or both of the DRX pins, a PDM clock running at 64 × f_{SYNCM} (3.072 MHz at 48 kHz f_{SYNCM}) is required to clock the PDM device. Either the PDMCLK/IO7 pin or the BCLK pin can produce the required PDM clock. The transceiver can simultaneously transmit TDM data over the DTX0 or DTX1 pin while receiving PDM streams. However, when BCLK is used as the PDM clock, only I²S/TDM2 and 32-bit channel widths or TDM4 with 16-bit channel widths are supported. Using PDMCLK/IO7 instead of BCLK to clock PDM devices

allows BCLK to be used for a variety of TDM modes. If both DRX0 and DRX1 are used to receive PDM data, it is possible to change the function of DTX1 such that it acts as an alternate DRX1. This enables the concurrent use of up to 4 PDM microphones and full duplex I²S communication.

If using only one pin (DRX0 or DRX1) for PDM, the other pin is available simultaneously for I²S/TDM transfers.

Mailboxes

There are two virtual mailboxes, MBOX0 and MBOX1, that allow for inter-processor communication between the host and a slave node control processor.

NOTE: Throughout this section, all specific references to MBOX0 also apply to the MBOX1 instance.

The processor in a slave node can send a message over I²C to registers in the A²B slave transceiver. In the master node, the host processor is informed about the new message by an interrupt on the master transceiver's IRQ/IO0 pin and can read out the message from A²B slave transceiver registers over I²C using the BUS_ADDR. If a mailbox message exchange is from the A²B master node to the A²B slave node, the host places a message in A²B slave transceiver registers over I²C using the BUS_ADDR. In the slave node, the processor is informed of this new message by an interrupt on the slave transceiver's IRQ/IO0 pin and can directly read out the message over I²C from A²B slave transceiver registers after checking the A2B_LINTTYPE register.

Mailbox Programming and Operation

The A2B_MBOX0CTL register provides bit fields to enable the mailbox and control direction, message length, and interrupt capabilities.

By default, mailbox 0 is configured as a receive mailbox (written by the host, read by the slave node processor), and mailbox 1 is configured as a transmit mailbox (written by the slave node processor, read by the host). Manipulating the A2B_MBOX0CTL.MB0DIR bit controls the direction of the mailbox.

Each mailbox can hold either 8-, 16-, 24-, or 32-bit messages, as configured in the A2B_MBOX0CTL.MB0LEN field. The value in this field determines which of the four byte-wide A2B_MBOX0B0 through A2B_MBOX0B3 registers to use for the data, where the first byte is always in the A2B_MBOX0B0 register, and the final byte is in the highest data register required to accommodate the programmed data length, as shown in the following table.

MBxLEN Field	Final Byte in Register
0b00	A2B_MBOX0B0
0b01	A2B_MBOX0B1
0b10	A2B_MBOX0B2
0b11	A2B_MBOX0B3

For an enabled receive mailbox (A2B_MBOX0CTL.MB0EN = 1 and A2B_MBOX0CTL.MB0DIR = 0), if the A2B_MBOX0CTL.MB0FIEN bit is set, an interrupt to the slave node occurs after the final byte of the mailbox is written by the host and received by the A²B slave transceiver. If the A2B_MBOX0CTL.MB0EIEN bit is set, an

interrupt is propagated back upstream over the A^{2}B bus to the host after the final byte of the mailbox is read by the local processor in the slave node.

For an enabled transmit mailbox (A2B_MBOX0CTL.MB0EN = 1 and A2B_MBOX0CTL.MB0DIR = 1), if the A2B_MBOX0CTL.MB0FIEN bit is set, an interrupt to the host occurs after the final byte of the mailbox is written by the local processor in the slave node. If the A2B_MBOX0CTL.MB0EIEN bit is set, an interrupt is propagated downstream over the A^{2}B bus to the slave node after the final byte of the mailbox is read by the host.

CAUTION: Dynamic reconfiguration of an enabled mailbox (A2B_MBOX0CTL.MB0EN = 1) is forbidden. The host must first disable the mailbox (A2B_MBOX0CTL.MB0EN = 0) and then re-enable it in two separate accesses if reconfiguration is required.

The A2B_MBOX0STAT register provides status information for the mailboxes:

- When a mailbox is filled, the A2B_MBOX0STAT.MB0FULL bit is set, and the A2B_MBOX0STAT.MB0EMPTY bit is cleared.
- When a mailbox is emptied, the A2B_MBOX0STAT.MB0EMPTY bit is set, and the A2B_MBOX0STAT.MB0FULL bit is cleared.
- The A2B_MBOX0STAT.MB0EIRQ and A2B_MBOX0STAT.MB0FIRQ bits are set when the mailbox signals an interrupt to the host or local processor, and the bits are cleared when the interrupt is processed by the host or local processor.

Multiple slave nodes can communicate to the master node through their TX mailboxes. In the master node, the A2B_INTTYPE register contains information about the pending interrupt generated by any slave node, with the slave node indicated in the A2B_INTSRC register.

When two slaves write to their mailboxes simultaneously, the master gets the interrupt indication from the slave that is closer to the master. Upon detecting the interrupt, the host extracts the interrupt information by reading the A^{2}B master transceiver's interrupt type (A2B_INTTYPE) and interrupt source (A2B_INTSRC) registers to determine which interrupt occurred and which slave node generated it, respectively. Upon reading the A2B_INTTYPE register, the interrupt request for that interrupt is cleared in the slave node identified by the value in the A2B_INTSRC register. The IRQ/IO0 pin toggles to the deasserted state and then immediately back into the asserted state due to the still active interrupt from the other slave node, and the host can again read the master transceiver's A2B_INTTYPE and A2B_INTSRC registers to acknowledge the other slave node's mailbox interrupt.

Mailbox Latency

The mailbox transactions are made up of register reads and writes over the I^{2}C bus. The interrupt request from a slave to the master is part of the SRF packet, so the latency on the slave to master mailbox can include an extra superframe waiting for this time.

The following figures show the system timing for the mailbox transactions in both directions. The light gray slots indicate the SCF field, and the dark gray slots indicate the SRF field.

As shown in the *Mailbox Latency (from Host to Slave)* figure, when the mailbox message is from the host to a slave processor, the host processor writes the mailbox data to the A^{2}B slave node through the SCF field using a 2-byte burst write access to the master transceiver BUS_ADDR device address. When the writes complete, the slave transceiver immediately generates the interrupt to its local node processor. As a result, the slave interrupt request (SLAVE IRQ) asserted on IRQ/IO0 aligns with the SCF field. Once this interrupt is asserted, the locally-connected processor can use the slave transceiver BASE_ADDR device address to interrogate the A2B_LINTTYPE register to determine that it is the mailbox full interrupt, after which it can then extract the data from the mailbox data registers using a 2-byte burst read. Once those transactions finish, the mailbox empty interrupt is generated at the master node (MASTER IRQ), aligned with the SRF field, and the host proceeds with reading the A2B_INTSRC and A2B_INTTYPE registers of the master transceiver (using the master transceiver BASE_ADDR device address) to determine that it is the mailbox empty interrupt originating with the indicated slave.

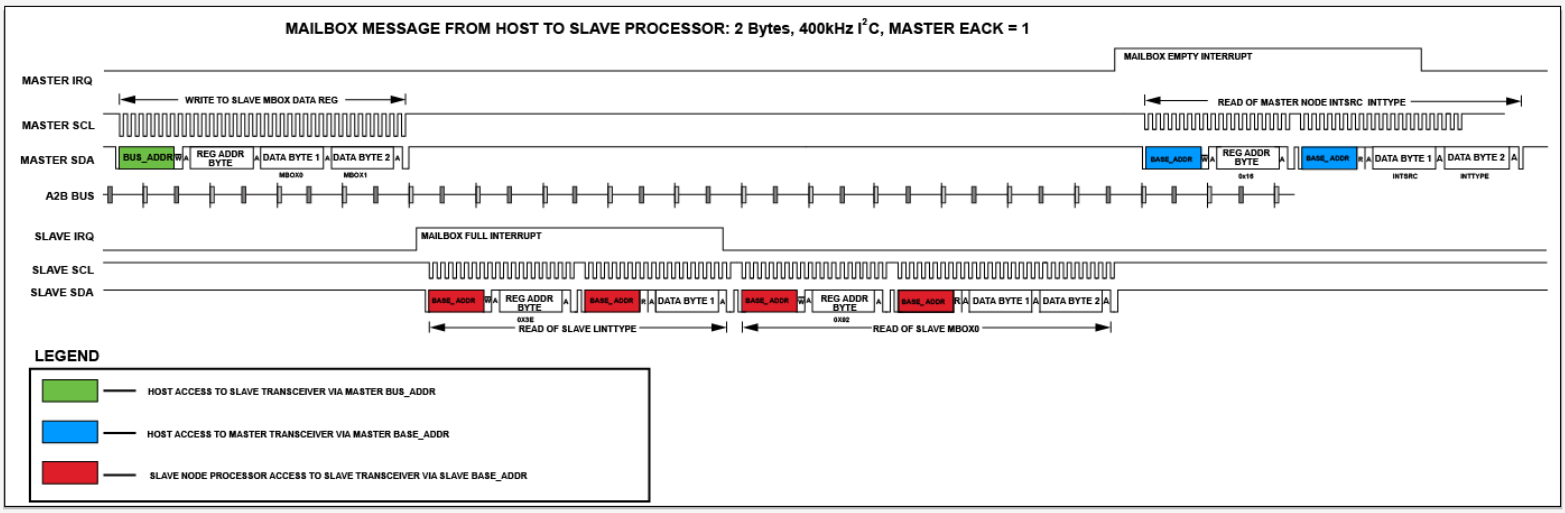

Figure 2-11: Mailbox Latency (from Host to Slave)

Similarly, as shown in the *Mailbox Latency (from Slave to Host)* figure, when the mailbox message is from a slave processor to the host, the slave node processor populates the mailbox data registers at any time by issuing writes to the registers using the slave transceiver BASE_ADDR device address, and the interrupt indication to the master A^{2}B node goes through the SRF field. As a result, the master mailbox full interrupt request (MASTER IRQ) asserted on IRQ/IO0 aligns with the SRF field. Once this interrupt is asserted, the host (using the master transceiver BASE_ADDR device address) interrogates the A2B_INTSRC and A2B_INTTYPE registers to determine that it is the mailbox full interrupt originating with the indicated slave.

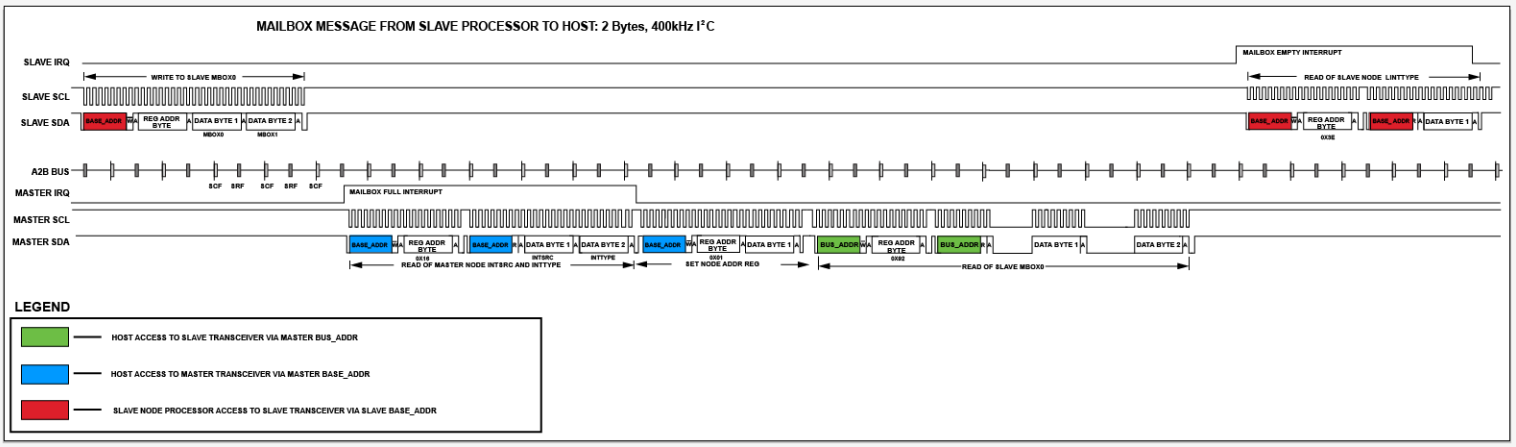

Figure 2-12: Mailbox Latency (from Slave to Host)

To subsequently extract the data from the mailbox of the slave transceiver, the host must first set the A2B_NODEADR register to the slave node that generated the interrupt (using a master transceiver BASE_ADDR write access), and then issue the BUS_ADDR accesses to read the mailbox data byte registers of the slave transceiver (note the superframe spacing required for these reads to take place). Once the last byte is read by the host, the mailbox empty interrupt request of the slave node (SLAVE IRQ) gets asserted in the next SCF. Then, the slave node processor can use a slave transceiver BASE_ADDR access to read the A2B_LINTTYPE register and take action after identifying that it was the mailbox empty interrupt that occurred (for example, load the mailbox data registers again to restart the process).

3 A2B Operation and Configuration

The A2B bus is high-level programmable and can address many use cases. A2B systems are easy to configure, based on knowledge of the system, nodes, and peripherals. The exact system configuration can be gained by collecting information individually from each slave. As an example, the same A2B module can be supplied by different vendors, with each of the modules having unique register programming requirements. One module can use TDM4 as an audio interface, while another one uses TDM8. One module can provide two upstream channels, while another can provide three upstream channels, all with the host not having prior knowledge of how many nodes are connected.

IMPORTANT: Ensure that the register programming results in a valid system configuration.

Analog Devices provides free SigmaStudio™ (http://www.analog.com/SigmaStudio) tools featuring an intuitive graphical user interface to architect, configure, and set up the A2B bus. The tools also generate driver code for embedded software.

Linux and QNX software drivers also are available upon request.

I2C Port Programming Concepts

Master-enabled transceiver registers are programmed directly by the A2B host via the I2C port using Direct I2C Register Accesses. Slave-enabled transceiver registers can also be programmed in this fashion by an I2C-connected controller on the slave node; however, A2B slave transceiver registers are typically programmed remotely by the A2B host through the master transceiver over the A2B bus using Remote Slave I2C Register Accesses . Further, if a slave transceiver is locally connected to an I2C slave device on the slave node, that connected I2C slave device can also be accessed remotely by the A2B host over the A2B bus using Remote Peripheral I2C Accesses. The ***Programming Sequence for I2C Accesses*** figure is a graphical representation of the programming sequences that are required when programming transceiver registers and accessing slave node I2C peripheral devices.

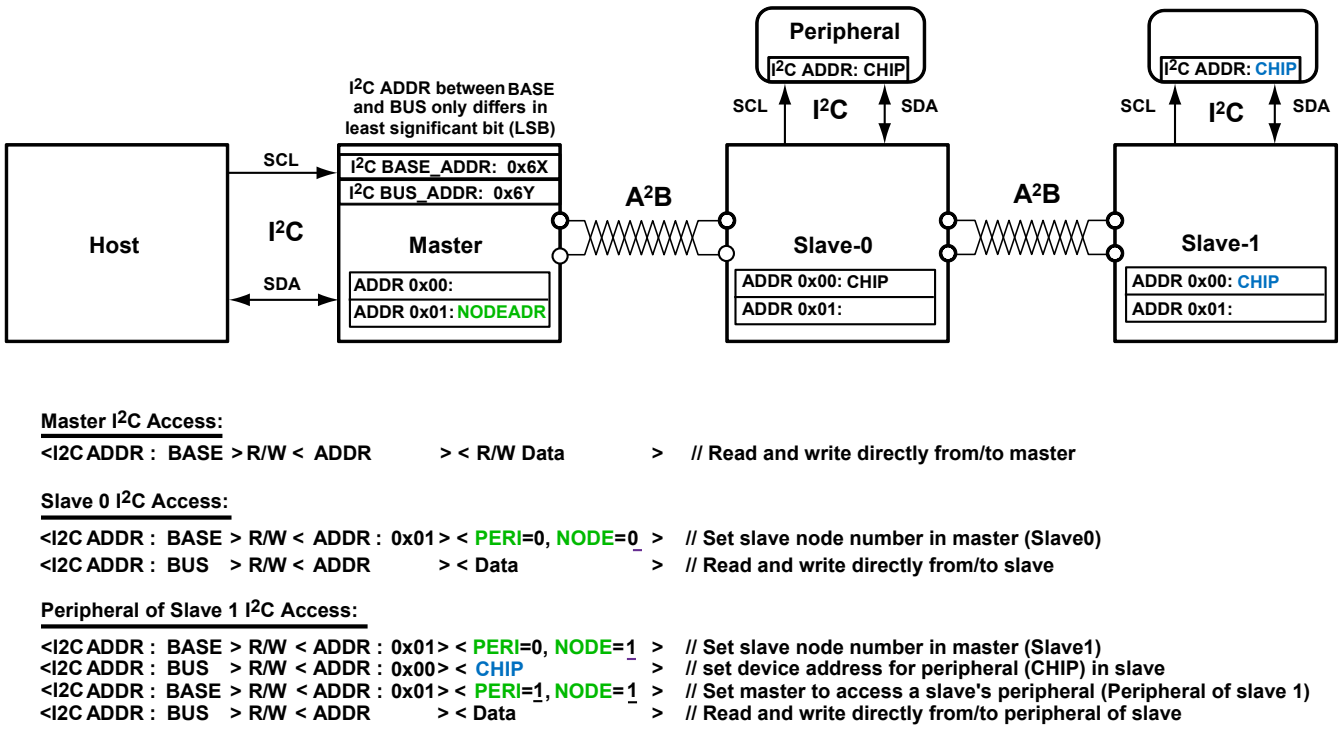

Figure 3-1: Programming Sequence for I²C Accesses

In the ***Programming Sequence for I²C Accesses*** figure:

- I2C ADDR is the master transceiver I²C device address:
  - Direct I²C Register Accesses to the master transceiver use BASE_ADDR (I2C ADDR: BASE).
  - Remote Slave I²C Register Accesses to a slave transceiver and Remote Peripheral I²C Accesses to an I²C-connected peripheral on a slave node use BUS_ADDR (I2C ADDR: BUS).

  NOTE: See Transceiver I²C Accesses for more details regarding BASE_ADDR and BUS_ADDR.
- NODEADR is the master transceiver `A2B_NODEADR` register:
  - NODE is the `A2B_NODEADR.NODE` field.
  - PERI is the `A2B_NODEADR.PERI` bit.
- CHIP is the `A2B_CHIP` register:
  - Black text indicates the `A2B_CHIP` register itself.
  - Blue text indicates the value of the `A2B_CHIP` register.

Direct I²C Register Accesses

The I²C port can be used to directly access the transceiver register space, whether the transceiver is configured as a master or as a slave:

- On the master node, the A²B host directly accesses the master transceiver register space using this method.
- On a slave node, a locally-connected I²C host directly accesses the slave transceiver register space using this method.

As shown in the ***Master I²C Access*** portion of the ***Programming Sequence for I²C Accesses*** figure, a master transceiver register access requires the I²C transfer from the host to consist of the master transceiver I²C device address (I2C ADDR: BASE = BASE_ADDR), followed by the register address (ADDR), followed finally by the data associated with the master transceiver register (R/W Data). For further details, see Transceiver I²C Accesses.

NOTE: This ***Master I²C Access*** sequence is identical for an I²C-connected host on the slave node directly accessing a slave transceiver's register space.

Remote Slave I²C Register Accesses

Though a locally-connected I²C host on a slave node can directly program slave transceiver registers over the I²C port, A²B systems are typically fully configured by the A²B host from the master node. As shown in the ***Slave 0 I²C Access*** portion of the ***Programming Sequence for I²C Accesses*** figure, the 2-step process consists of the A²B host first directly configuring the master transceiver before using remote I²C accesses to program a specific slave transceiver over the A²B bus. The A²B host must use the following programming sequence to access an A²B slave transceiver register space remotely over the A²B bus from the master node.

1. Use a Direct I²C Register Accesses to set the master transceiver `A2B_NODEADR.NODE` field to the slave node ID to be accessed. Be sure the `A2B_NODEADR.PERI` bit is set to 0 in this write so that subsequent bus accesses target the indicated slave transceiver register space rather than an I²C peripheral connected to the indicated slave.

   ADDITIONAL INFORMATION: Setting the `A2B_NODEADR.NODE` field to 0 means that subsequent bus accesses will target slave node 0. If this field were set to 1, subsequent bus accesses would target slave node 1. If the intent is to broadcast the write to all of the discovered nodes (master and slaves), be sure to also set the broadcast bit (`A2B_NODEADR.BRCST`) in this write.

2. To access the slave transceiver register, the I²C transfer from the host consists of the master transceiver's bus address (I2C ADDR: BUS = BUS_ADDR), followed by the slave transceiver register address (ADDR), followed finally by the data associated with the slave transceiver register (Data). For more details, see Transceiver I²C Accesses.

Remote Peripheral I²C Accesses

The ***Peripheral of Slave 1 I²C Access*** portion of the ***Programming Sequence for I²C Accesses*** figure illustrates the sequence required for the A²B host to access a peripheral connected to the I²C port of a slave transceiver over the A²B bus using remote peripheral I²C accesses. The A²B host must follow the below programming sequence to access an I²C peripheral on an A²B slave node (for example, a microphone or a DAC) over the A²B bus.

1. Use a Direct I²C Register Accesses write access to set the master transceiver `A2B_NODEADR.NODE` field to the slave node ID that is connected to the peripheral to be accessed. Be sure the `A2B_NODEADR.PERI` bit is cleared in this write so that the subsequent bus access is to the targeted slave transceiver's register space, not to the slave peripheral itself.

   ADDITIONAL INFORMATION: The `A2B_NODEADR.NODE` field is set to 1 in this write so that subsequent bus accesses target slave node 1. If the intent is to broadcast the peripheral write to all of the discovered nodes (master and slaves), be sure to also set the `A2B_NODEADR.BRCST` bit in this write. If the `A2B_CHIP` register in the targeted slave transceiver is already set to the I²C address of the intended peripheral access, perform this write with the `A2B_NODEADR.PERI` bit set (rather than cleared) and proceed directly to the final step.

2. Use a Remote Slave I²C Register Accesses write access to program the desired slave transceiver 's `A2B_CHIP` register with the I²C device address of the peripheral connected to the slave.

3. Use a Direct I²C Register Accesses write access to set the master transceiver `A2B_NODEADR.PERI` bit (while maintaining the content of the `A2B_NODEADR.NODE` field) such that subsequent BUS_ADDR accesses go to the desired slave node I²C peripheral.

4. To access the slave node peripheral, the I²C transfer from the host must consist of the master transceiver's BUS_ADDR (I2C ADDR: BUS), followed by the address that the slave transceiver will use to access the slave node I²C peripheral (ADDR), followed finally by the data associated with the address (Data).

System Bring-Up and Discovery

An A²B system is brought up by the A²B host. Once power is properly established, each node in the system must be discovered and configured in order, starting with the master node.

Reset and Operating States

Loss of PLL lock resets all of the register information except `A2B_BMMCFG` and `A2B_CONTROL.MSTR`.

The ***Transceiver State Diagram*** figure shows transceiver state information that is important to understand when bringing up and running a complete A²B system.

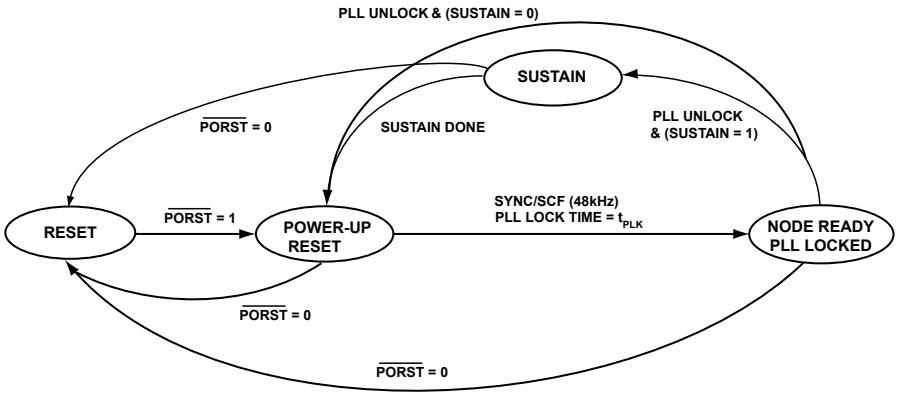

Figure 3-2: Transceiver State Diagram

NOTE: As sustain mode is a slave-only feature, a master transceiver never enters the SUSTAIN state. A loss of PLL lock on a master transceiver results in a direct return to the POWER-UP state.

Transceiver Power-On and Reset

When the transceiver is initially being powered on, it is in the RESET state. When in RESET, all A^2B system registers are held in reset, and no registers can be programmed until the transceiver advances to the POWER-UP RESET state, which is a function of power (VIN) applied to the transceiver.

An internal power-on reset circuit monitoring the state of the VIN power supply pin holds an internal power-on reset signal ($\overline{\text{PORST}}$) asserted low until the V_{RSTN} specification is met, at which point $\overline{\text{PORST}}$ is deasserted high to indicate that the transceiver is properly powered. The transceiver then transitions to the POWER-UP RESET state. After the transceiver enters POWER-UP RESET, the $\overline{\text{PORST}}$ signal remains deasserted high unless the voltage sensed on the VIN power supply pin drops into its V_{RST} specification range, in which case $\overline{\text{PORST}}$ is asserted low to bring the transceiver back to RESET.

Master Bring-Up and Operation

Referring to the ***Transceiver State Diagram*** figure, the ADR1/IO1 and ADR2/IO2 pins are latched to determine the I^2C device address when the transceiver transitions to the POWER-UP RESET state, and the transceiver is I^2C-device accessible no more than 2.5 ms later. The A^2B host then sets the `A2B_CONTROL.MSTR` bit in the transceiver before driving the SYNC pin (the input clock to a master-enabled transceiver's PLL) at the audio sampling rate of the system (selectable between 48 kHz or 44.1 kHz). The master transceiver locks its PLL to the received SYNC signal according to the PLL Lock Time (t_{PLK}) specification.

NOTE: It is recommended that the host set a timeout in excess of the PLL Lock Time (t_{PLK}) specification so that a non-responsive master transceiver can be detected by software.

Upon PLL lock, the master transceiver transitions to the NODE READY PLL LOCKED state, at which point it generates the MSTR_RUNNING (0xFF) interrupt to the host (the IRQ/IO0 pin is driven high), as stored in the

interrupt type register (A2B_INTTYPE), indicating the master transceiver is ready for programming via the I²C interface.

NOTE: Once the PLL is locked, writing the A2B_CONTROL.MSTR bit has no effect.

If the master transceiver PLL becomes unlocked during bus operation, the transceiver goes back to POWER-UP, as the SUSTAIN state is a slave-only feature (SUSTAIN = 0). All of the registers return to their reset values except the A2B_CONTROL register.

Slave Bring-Up and Operation

Referring to the ***Transceiver State Diagram***, the transceiver is in the POWER-UP RESET state once local or A²B bus power is established, and the ADR1/IO1 and ADR2/IO2 pins are latched to determine the I²C device address. The transceiver is by default a slave and is ready to be discovered and programmed 2.5 ms after entering POWER-UP RESET.

NOTE: The BCLK and SYNC outputs are three-stated in the POWER-UP RESET state.

While in the POWER-UP RESET state, a subset of the slave transceiver register space can be configured through the I²C port by a locally-connected host using Direct I²C Register Accesses. These registers include:

A2B_BMMCFG	A2B_CHIP	A2B_BCDNSLOTS
A2B_LDNSLOTS	A2B_LUPSLOTS	A2B_DNSLOTS
A2B_UPSLOTS	A2B_INTMSK0	A2B_INTMSK1
A2B_BECCTL	A2B_TESTMODE	A2B_I2CCFG
A2B_SYNCOFFSET	A2B_PDMCTL	A2B_ERRMGMT
A2B_GPIODAT	A2B_GPIOOEN	A2B_GPIOIEN
A2B_PINTEN	A2B_PINTINV	A2B_PINCFG
A2B_I2SRATE	A2B_I2SRRCTL	A2B_I2SRRSOFFS
A2B_CLK1CFG	A2B_CLK2CFG	A2B_UPMASK0 - A2B_UPMASK3
A2B_UPOFFSET	A2B_DNMASK0 - A2B_DNMASK3	A2B_DNOFFSET
A2B_GPIODEN	A2B_GPIOD0MSK - A2B_GPIOD7MSK	A2B_GPIODINV
A2B_MBOX0CTL - A2B_MBOX1CTL	A2B_I2STEST	A2B_I2SRATE
A2B_I2SGCFG	A2B_I2SCFG	

Even though these registers can be written in the POWER-UP RESET state, programmed values do not take effect until the transceiver advances to the NODE READY PLL LOCKED state, with the exception of the slot registers (A2B_BCDNSLOTS, A2B_LDNSLOTS, A2B_LUPSLOTS, A2B_DNSLOTS, A2B_UPSLOTS, A2B_UPMASK0 through A2B_UPMASK3, and A2B_DNMASK0 through A2B_DNMASK3). Values programmed to the listed slot registers do not take effect until the master transceiver A2B_DATCTL register is programmed and the new structure is subsequently applied (A2B_CONTROL.NEWSTRCT = 1).

In the POWER-UP RESET state, a slave transceiver awaits synchronization control frames (SCFs) coming from the master, which is initiated when the host initiates the discovery process for that particular A^{2}B system slave by setting the master transceiver's A2B_DISCVRY register to the response time for the targeted slave. When this write occurs, the master initiates discovery by sending discovery frames with the response time value embedded in the SCF. The slave being discovered then extracts the information to set its response time (A2B_RESPCYCS). These discovery frames provide the input clock to the slave transceiver, which the slave transceiver locks its PLL to in accordance with the PLL Lock Time (t_{PLK}) specification. Once the slave transceiver PLL is locked, it is in the NODE READY PLL LOCKED state and starts generating synchronization response frames (SRFs) to upstream nodes, which causes the master transceiver to generate the DSCDONE interrupt (A2B_INTTYPE = 0x18) indicating that the slave transceiver is ready for programming over the A^{2}B bus using Remote Slave I^{2}C Register Accesses .

ATTENTION: When simultaneous writes to the same register are attempted from both the A^{2}B bus (using Remote Slave I^{2}C Register Accesses) and the I^{2}C port (using Direct I^{2}C Register Accesses), the order in which these I^{2}C accesses occur cannot be predicted. Therefore, special care must be taken when I^{2}C transactions are also coming from both sources.

TIP: If local node programming (using Direct I^{2}C Register Accesses) is desired in the application, this potential contention can be avoided by using a mailbox handshake with the master node such that the host writes one of the slave's mailboxes when it is ready to begin making register accesses and then waits for the slave to read that mailbox as an indication that its initialization sequence is complete. See Mailboxes for more information.

In the NODE READY PLL LOCKED state, the BCLK and SYNC output are driven low until any I2S/TDM/PDM port data pin is enabled in the slave transceiver's A2B_PDMCTL (for PDM mode) or A2B_I2SCFG (for I^{2}S/TDM modes) registers.

If the slave transceiver PLL becomes unlocked during bus operation, it goes back to the POWER-UP state if the clock sustain feature is disabled (A2B_SUSCFG.SUSDIS = 1, denoted in the ***Transceiver State Diagram*** figure as SUSTAIN = 0). Once back in the RESET state, the master can issue another discovery sequence.

Clock Sustain Functionality

By default (and denoted in the ***Transceiver State Diagram*** figure as SUSTAIN = 1), the slave transceiver has a clock sustain feature to power down slave nodes with processors and DACs, where audio signals of locally powered slave nodes are gracefully muted. When the bus loses communication and a reliable clock cannot be recovered by the slave transceiver (PLL UNLOCK in the ***Transceiver State Diagram*** figure), the slave transceiver enters the SUSTAIN state, provided the clock sustain feature has not been disabled (A2B_SUSCFG.SUSDIS = 1). Upon entering the SUSTAIN state, the transceiver:

- Runs at the current clock frequency for 1024 SYNC periods
  - I^{2}S/TDM ports continue running
  - Signals SUSTAIN state on a GPIO, if enabled
  - PLL relock is not attempted while in the SUSTAIN state

- Resets and re-enters the POWER-UP RESET state
- Transitions to the NODE READY PLL LOCKED state if stable SCF discovery frames are present

If the sustain GPIO output enable (A2B_SUSCFG.SUSOE) bit is set, the sustain signal from the PLL is driven high on the GPIO pin selected by the A2B_SUSCFG.SUSSEL bit field while the transceiver is in the SUSTAIN state. This feature has a higher priority than other GPIO outputs, but a lower priority than function outputs on the pins. For example, if clock output 1 is enabled (A2B_CLK1CFG.CLK1EN = 1), the ADR1/IO1 pin is driven as a clock output. Setting the A2B_SUSCFG.SUSOE bit and configuring the sustain output on the ADR1/IO1 pin (A2B_SUSCFG.SUSSEL = 1) does not override this behavior.

The sustain signal from the PLL goes high near the beginning of a superframe. Once the sustain signal is high, decaying data values are produced on the DTX0/IO3 and DTX1/IO4 pins, starting on the following I²S/TDM frame.

The data in the TX frame buffer (see Managing A²B System Data Flow), as received from the A²B bus, contains a 32-bit value that is output to either or both of the I²S DTX0/DTX1 data pins. Negative values gradually attenuate to 0, while positive values gradually attenuate to –109 dB (0x00001F00) on the enabled data pins.

Node Discovery and Initialization

This section provides information regarding simple node discovery and initialization for an A²B bus system. Modified, optimized, and advanced discovery flows are described in Appendix A: Additional Discovery Flow Examples of this manual. Any of these software flow diagrams can be used as a guideline for discovery and initialization.

Simple Discovery Flow

All slave nodes are discovered sequentially from slave 0 to the last available slave in the system with the software flow shown in the ***Simple Discovery Flow*** figure. In this figure, the stages show commands as issued over the I²C interface between the host and the master-enabled transceiver. Write commands are identified as "wr" and read commands are identified as "rd" along with the REGISTER_NAME being accessed. The "M" indicates an access to the BASE_ADDR, and the "S" indicates an access to the BUS_ADDR.

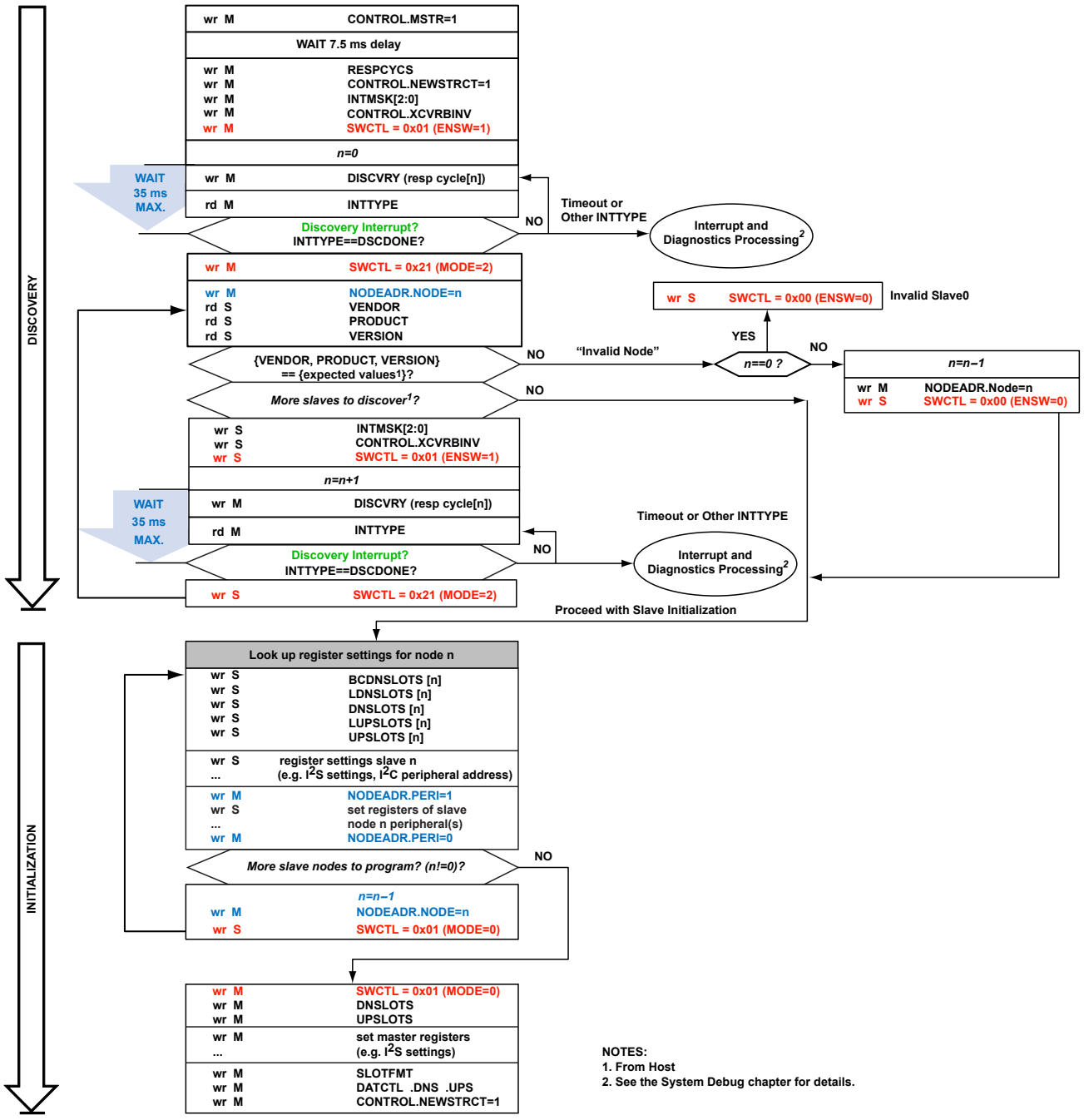

Figure 3-3: Simple Discovery Flow

NOTE: In the ***Simple Discovery Flow*** figure, setting the `A2B_SWCTL.ENSW` bit in the master node or in any slave node causes it to begin sending SCFs downstream to the next connected slave, thus allowing that next slave transceiver to begin locking its PLL before the master node initiates discovery frames targeting it.

Use the following guidelines for the reverse-wire feature A2B_CONTROL.XCVRBINV (Invert Data to/from LVDS XCVR B):

1. In the master node, set the A2B_CONTROL.XCVRBINV bit prior to writing to the A2B_SWCTL.ENSW bit. Be careful to avoid inadvertently clearing the A2B_CONTROL.XCVRBINV bit when writing to the A2B_CONTROL register for other purposes, such as writing to the A2B_CONTROL.NEWSTRCT bit.

2. In any slave node, the A2B_CONTROL.XCVRBINV bit must be set before writing to the A2B_SWCTL.ENSW bit.

Once all of the slave nodes are discovered, initialize the nodes for synchronous data exchange. The example flow diagram starts initialization with the last node and finishes with the master.

The discovery finishes quickly, providing earlier access to all nodes and their I^2C peripherals before the initialization for synchronous audio, which takes extra time to finish.

There is no further need for bus management after all of the nodes are discovered and programmed. Interrupt service routines can be used to react to special interrupt request (IRQ) events (for example, from an IO pin). Alternatively, the A2B_INTTYPE register can be polled to monitor interrupt events.

The Optimized Discovery Flow and Advanced Discovery Flow sections illustrate how to perform auto-configuration.

Response Cycles

The A2B_RESPCYCS register sets the relative time from the start of a synchronization control frame (SCF) to the moment the last slave responds with a synchronization response frame (SRF). The register setting indicates to earlier nodes when to expect the response from the last slave. If the last node does not respond, the previous node that is next to the presumed last node does respond.

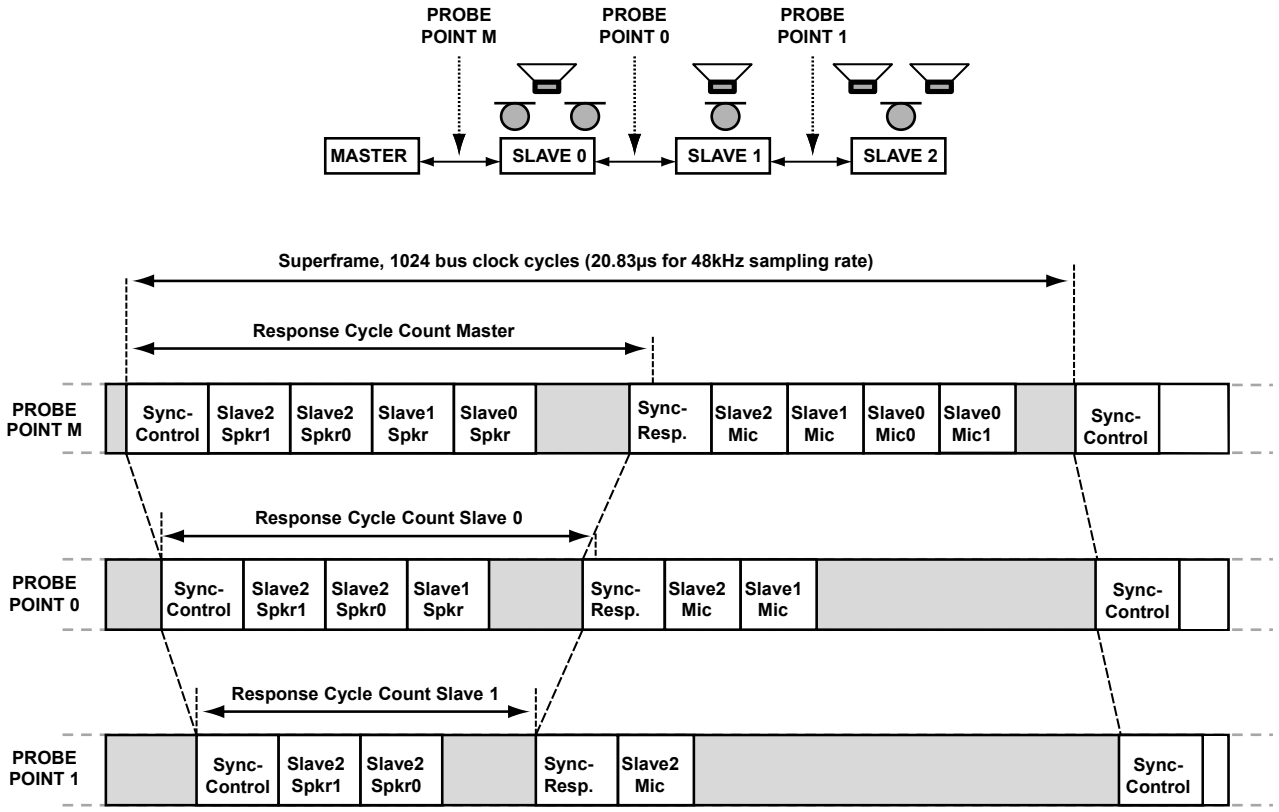

Figure 3-4: Synchronous Data Enabled

The response cycle values for the transceivers are discussed in Appendix B: Response Cycle Formula as a function of the following parameters:

- Number of slave nodes
- Number of downstream slots
- Downstream slot size
- Number of upstream slots
- Upstream slot size
- Master I^2S/TDM channel configuration

NOTE: The master transceiver response cycle values are calculated using the above parameters in the response cycle calculator spreadsheet or in SigmaStudio software. For more information, contact your local Analog Devices representative.

Slave Node Response Cycles

The ***Slave Node Response Cycle*** figure shows the relative timing between SCFs and SRFs on the A and B XCVR ports of a slave node. A slave node generates the SRF approximately ((4 * `A2B_RESPCYCS`) + 7) bits after the SCF

starts on the A XCVR. For example, when A2B_RESPCYCS= 128 (0x80), the slave node generates the SRF beginning at the 519th ((4 * 128) + 7 = 519) bit.

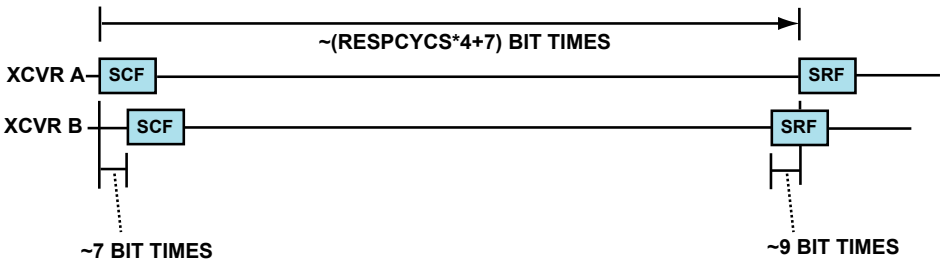

Figure 3-5: Slave Node Response Cycle

As shown in the ***Slave Node Response Cycle*** figure, there are transceiver delays (TD) incurred to pass the superframe from one side of the transceiver to the other. For the downstream portion of the superframe, there is a delay (TD_{DOWN}) of seven (± 2) bits incurred when going from the A-side to the B-side of the transceiver. Conversely, there is a delay (TD_{UP}) of nine (± 2) bits going from the B-side to the A-side during the upstream portion of the same superframe. These delays are summarized for the supported frame rates in the ***Transceiver Delays*** table, as governed by the equation:

Delay Range = Nominal Latency Range / (SYNC Rate * 1024)

Table 3-1: Transceiver Delays

Time Delay (Direction)	SYNC Rate (kHz)	Nominal Latency Range (SYSBCLK)	Delay Range (ns)
TD_{DOWN} (A-Side to B-Side Downstream)	44.1	7 ± 2	110.7 - 199.3
TD_{DOWN} (A-Side to B-Side Downstream)	48.0	7 ± 2	101.7 - 183.1
TD_{UP} (B-Side to A-Side Upstream)	44.1	9 ± 2	155.0 - 243.6
TD_{UP} (B-Side to A-Side Upstream)	48.0	9 ± 2	142.4 - 223.8

In addition to these transceiver delays, cable delays (CD) between nodes also change the relative timing between when the SCF is received in the downstream potion of the superframe and when the complementary SRF returns to that point during the upstream portion of the same superframe. There is a 5-bit time window (expected bit time ±

2) in which the SRF is correctly received on the B-side and passed to the A-side of a slave node. An SRF outside of this window is still detected, and the expected response time is gradually (and automatically) adjusted by the transceiver during discovery to compensate for mismatches, with an adjustment range of -4 bit times to +15 bit times to span the cable length specifications. As such, the A2B_RESPCYCS formula works for all supported cable lengths. If the cable length is known during the system design phase, this recommendation can be applied for all discovery flows. If the cable lengths are unknown, the default response cycles calculation (assuming 4m cable length) is adequate. Although some errors can be observed during discovery (CRCERR, SRFERR, or SRFCRCERR) when longer cables are used, the system runs cleanly after discovery completes due to this automatic adjustment capability.

The automatic response cycle adjustment performed during discovery works as follows:

1. The host programs the master to expect the SRF at the 519th bit of the superframe by setting A2B_RESPCYCS = 128 (0x80) , as detailed above ((4 * 128) + 7 = 519).
2. The master node initiates discovery of slave 0 when the host writes 0x80 to its A2B_DISCVRY register. When slave 0 starts sending SRFs, the master adjusts its response time to align with slave 0.

   Short cable lengths (up to 20cm) do not impact the master node's ability to receive the SRF at the 519th bit of the superframe.

   Longer cable lengths, however, introduce a physical cable delay (CD) on the order of 5ns/m to the time at which the SRF is captured at the receiving node. For example, a 10m cable between the master node and the slave 0 node delays the SRF reception time at the master node by 100ns (50ns downstream CD plus 50ns upstream CD). This 100ns total CD equates to five A^2B bits, thus causing the master node in this case to adjust its response cycles to expect the SRF at the 524th (± 2) bit of the superframe.
3. The master node initiates discovery of slave 1 when the host writes 0x7C to the A2B_DISCVRY register. When slave 1 starts sending SRFs, slave 0 adjusts its response time to align with slave 1, which causes the SRFs from slave 0 to be delayed, thus adding further delay to the time at which the SRF reaches the master node.

   The master node receives the SRF as a function of the CD between the master node and slave 0 and the CD between slave 0 and slave 1. Continuing with the above example, a second 10m cable between slave 0 and slave 1 delays the SRF reception time at the master node by an additional five bits, thus causing the master node to adjust its response cycles to expect the SRF at the 529th (± 2) bit of the superframe.

The ***SRF Response*** figure illustrates how cable and transceiver delays affect the SRF response. In this case, the SRF miss error is not observed because the response cycles are adjusted during the discovery phase.

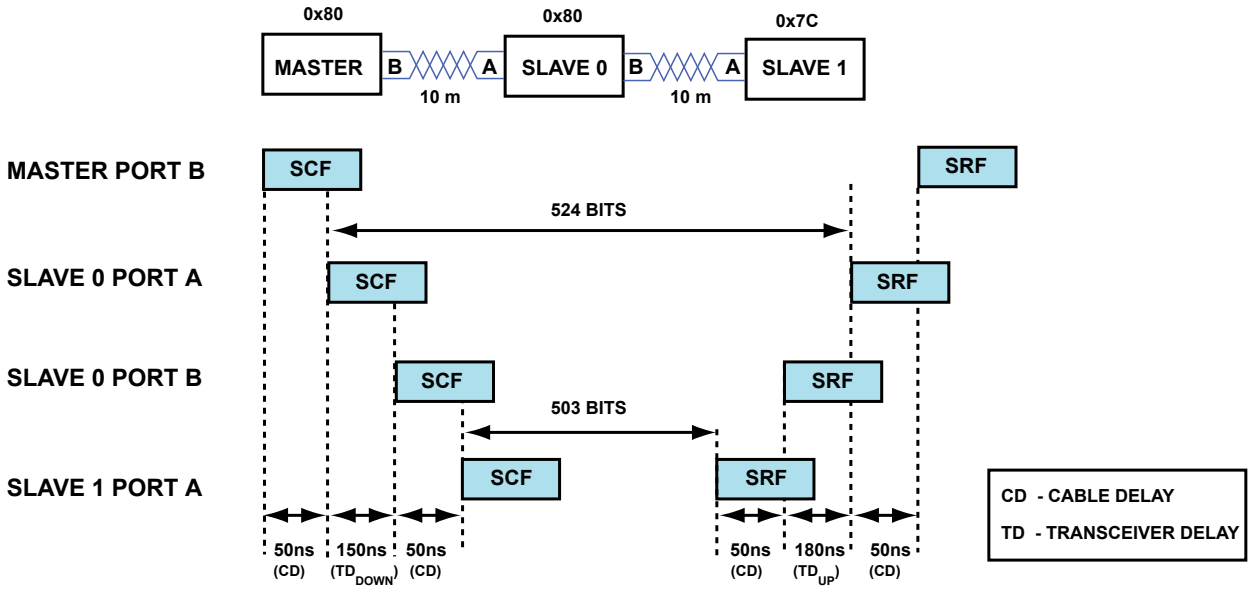

Figure 3-6: SRF Response

In this example:

- Slave 1 is the last-in-line slave node, which is responsible for initiating the SRF to commence the upstream portion of the superframe. When programmed with `A2B_RESPCYCS` = 124 (`0x7C`) , slave 1 is configured to generate the SRF at the 503rd bit of the superframe ((4 * 124) + 7 = 503).
- From the perspective of the upstream slave 0, the total delay between the SCF arriving to the slave 0 A-side transceiver during the downstream portion of the superframe and the corresponding SRF appearing there during the upstream portion of the same superframe is 430ns (21 bits), comprised of:
  - the downstream transceiver delay of slave 0 (TD_{DOWN} = 150ns),
  - the downstream cable delay between slave 0 and slave 1 (CD = 5ns/m x 10m = 50ns),
  - the upstream cable delay between slave 1 and slave 0 (CD = 5ns/m x 10m = 50ns), and
  - the upstream transceiver delay of slave 0 (TD_{UP} = 180ns)

  Therefore, the number of bits between SCF arrival to the slave 0 A-side transceiver and the corresponding SRF being generated there is calculated to be 503 + 21 = 524 bits for a 10m cable length between the slave 0 node and the slave 1 node.
- From the perspective of the master node, the total delay between generating the SCF and the corresponding SRF appearing during the upstream portion of the same superframe is 100ns (5 bits), which is comprised of:
  - the downstream cable delay between the master and slave 0 (CD = 5ns/m x 10m = 50ns) and
  - the upstream cable delay between slave 0 and the master (CD = 5ns/m x 10m = 50ns)

Therefore, the number of bits between SCF field generation and the corresponding SRF being received is calculated to be 524 + 5 = 529 bits.

Managing A2B System Data Flow

Each master and slave transceiver in a full A2B system must be properly configured for the desired slot management scheme and format for both upstream and downstream traffic on the A2B bus between any two transceivers.

Each transceiver features two internal frame buffers:

- TX frame buffer - populated by the A2B bus and outputs to the DTX0 and/or DTX1 pins
- RX frame buffer - populated by the DRX0 and/or DRX1 input pins and outputs to the A2B bus

These frame buffers are populated and drained during each superframe, with the downstream slot content occupying the lower-order locations of the buffer and the upstream slot content in the higher-order locations. The frame buffers are 32 locations deep and 32-bits wide because any given transceiver can occupy up to 32 slots on the A2B bus and supports up to 32-bit data.

ATTENTION: If a transceiver is configured to receive more than 32 combined downstream and upstream slots from the A2B bus, the extra data associated with the upstream slots that cannot be accommodated by the frame buffer is dropped.

The TX frame buffer is populated by the A2B bus. The number of enabled downstream slots and specific slot masks determine which downstream slots are stored to the TX frame buffer during the downstream portion of the superframe. Similarly, the number of upstream data slots and specific slot masks determine which upstream slots are subsequently stored to the TX frame buffer after the downstream data. This combined buffer of data is then presented to the enabled DTXn data pins as a function of the number of transmit data pins enabled and whether or not interleaving is enabled.

The RX frame buffer is populated by the I2S/TDM port over the enabled DRXn data pins. The number of enabled receive pins and whether or not interleaving is turned on determine how data is placed into the RX frame buffer. Once the buffer is populated, the number of enabled downstream slots and specific slot masks determine which downstream slots are populated by the RX frame buffer during the downstream portion of the superframe. Similarly, the number of upstream data slots and specific slot masks determine which upstream slots are subsequently populated by the RX frame buffer after the downstream data has been sent.

Definition of dnmaskrx and upmaskrx

The dnmaskrx value is determined from the value of the A2B_DNMASK0 through A2B_DNMASK3 registers.

```
if (DNMASK3.RXDNSLOT31==1) dnmaskrx = 32;
  else if (DNMASK3.RXDNSLOT30==1) dnmaskrx = 31;
  else if (DNMASK3.RXDNSLOT29==1) dnmaskrx = 30;
  . . .
  else if (DNMASK0.RXDNSLOT02==1) dnmaskrx = 3;
  else if (DNMASK0.RXDNSLOT01==1) dnmaskrx = 2;
```

```
   else if (DNMASK0.RXDNSLOT00==1) dnmaskrx = 1;
   else dnmaskrx = 0;
```

The upmaskrx value is determined from the value of the A2B_UPMASK0 through A2B_UPMASK3 registers

```
if (UPMASK3.RXUPSLOT31==1) upmaskrx = 32;
  else if (UPMASK3.RXUPSLOT30==1) upmaskrx = 31;
  else if (UPMASK3.RXUPSLOT29==1) upmaskrx = 30;
   . . .
  else if (UPMASK0.RXUPSLOT02==1) upmaskrx = 3;
  else if (UPMASK0.RXUPSLOT01==1) upmaskrx = 2;
  else if (UPMASK0.RXUPSLOT00==1) upmaskrx = 1;
  else upmaskrx = 0;
```

A2B Slot Format

The normal (default) format of both upstream and downstream data slots is the data followed by a single parity bit. However, alternate formats supporting floating-point compression or ECC protection are also available. Both the size and the format of upstream and downstream data slots are configured using the A2B_SLOTFMT register. The ***Slot Format*** table summarizes the possible data formats configured by the A2B_SLOTFMT.DNFMT , A2B_SLOTFMT.DNSIZE , A2B_SLOTFMT.UPFMT , and A2B_SLOTFMT.UPSIZE bits. In the ***Slot Format*** table, the FMT column is the A2B_SLOTFMT.DNFMT bit or the A2B_SLOTFMT.UPFMT bit, and the SIZE column is the 3-bit A2B_SLOTFMT.DNSIZE or A2B_SLOTFMT.UPSIZE field, depending on whether it is the downstream or upstream (respectively) slot format that is being configured.

Table 3-2: Slot Format

FMT	SIZE	A2B Slot Size	Compression	Protection	Data Width	A2B Bus Bits
0	0b000	8-bit	None	Parity	8-bit	9
0	0b001	12-bit	None	Parity	12-bit	13
0	0b010	16-bit	None	Parity	16-bit	17
0	0b011	20-bit	None	Parity	20-bit	21
0	0b100	24-bit	None	Parity	24-bit	25
0	0b101	28-bit	None	Parity	28-bit	29
0	0b110	32-bit	None	Parity	32-bit	33
0	0b111	RESERVED				
1	0b000	RESERVED				
1	0b001	12-bit	FP	Parity	16-bit	13
1	0b010	16-bit	FP	Parity	20-bit	17
1	0b011	20-bit	FP	Parity	24-bit	21
1	0b100	24-bit	None	ECC	24-bit	30

Table 3-2: Slot Format (Continued)

FMT	SIZE	A2B Slot Size	Compression	Protection	Data Width	A2B Bus Bits
1	0b101	RESERVED				
1	0b110	32-bit	None	ECC	32-bit	39
1	0b111	RESERVED				

NOTE: In the ***Slot Format*** table, the I2S/TDM Data Width column indicates the width of the actual data being exchanged over the I2S/TDM/PDM port in MSB-first format. Use cases for data widths in this column from 8 to 16 bits can optionally set the `A2B_I2SGCFG.TDMSS` bit to utilize 16-bit TDM channel data width over the I2S/TDM/PDM port. Data widths from 20-32 bits require the `A2B_I2SGCFG.TDMSS` bit to be cleared (32-bit TDM channel data width). See I2S/TDM Port Programming Concepts for more details.

ECC Protection

The transceiver provides support for both 24- and 32-bit data with ECC protection for the A2B bus data slots.

NOTE: As shown in the ***Slot Format*** table, there are six ECC bits for 24-bit data and seven ECC bits for 32-bit data.

ECC protection is useful in an environment where strong noise interferences (shorter than the superframe) are present, which otherwise can generate bit errors. ECC can be used in addition to the audio data error correction (repeat of last known good data), but it may only be used for non-audio data because it requires extra bus bandwidth.

Floating-Point Data Compression

The A2B protocol engine provides optional floating-point data compression/decompression so that less bandwidth is used on the A2B bus for a given data size (with better quality than the immediately lower data size). This compression can be used for A2B data sizes of 12, 16, and 20 bits, corresponding to the I2S data width. The compression encodes the number of leading sign bits in the source data as a 3-bit field and concatenates the sign bit itself, followed by *N*-4 bits of data (where *N* is the A2B data size). An example of 16-bit to 12-bit compression is shown in the ***16-Bit to 12-Bit Compression Example*** table. In the table, `s` is the sign bit and `~s` is the inverse of the sign bit.

Table 3-3: 16-Bit to 12-Bit Compression Example

16-Bit Data																-->	*12-Bit FP Data*											
s	~s	x	x	x	x	x	x	x	x	***y***	***y***	***y***	***y***	***y***	***y***	-->	0	0	0	s	x	x	x	x	x	x	x	x
s	s	~s	x	x	x	x	x	x	x	x	***y***	***y***	***y***	***y***	***y***	-->	0	0	1	s	x	x	x	x	x	x	x	x
s	s	s	~s	x	x	x	x	x	x	x	x	***y***	***y***	***y***	***y***	-->	0	1	0	s	x	x	x	x	x	x	x	x
s	s	s	s	~s	x	x	x	x	x	x	x	x	***y***	***y***	***y***	-->	0	1	1	s	x	x	x	x	x	x	x	x

Table 3-3: 16-Bit to 12-Bit Compression Example (Continued)

16-Bit Data																-->	*12-Bit FP Data*											
s	s	s	s	s	~s	x	x	x	x	x	x	x	x	***y***	***y***	-->	1	0	0	s	x	x	x	x	x	x	x	x
s	s	s	s	s	s	~s	x	x	x	x	x	x	x	x	***y***	-->	1	0	1	s	x	x	x	x	x	x	x	x
s	s	s	s	s	s	s	~s	x	x	x	x	x	x	x	x	-->	1	1	0	s	x	x	x	x	x	x	x	x
s	s	s	s	s	s	s	s	~s	x	x	x	x	x	x	x	-->	1	1	1	s	x	x	x	x	x	x	x	x

Data decompression reverses the process. The LSB of the compressed data (***L*** in the ***12-Bit to 16-Bit Data Decompression Example*** table) is used to generate any remaining LSBs of the decompressed data that are not stored in the compressed format.

Table 3-4: Example of Data Decompression: 12 Bit to 16 Bit

12-Bit FP Data												-->	*16-Bit Decompressed Data*														
0	0	0	s	x	x	x	x	x	x	x	***L***	-->	s	~s	x	x	x	x	x	x	x	***L***	***L***	***L***	***L***	***L***	***L***	***L***
0	0	1	s	x	x	x	x	x	x	x	***L***	-->	s	s	~s	x	x	x	x	x	x	x	***L***	***L***	***L***	***L***	***L***	***L***
0	1	0	s	x	x	x	x	x	x	x	***L***	-->	s	s	s	~s	x	x	x	x	x	x	x	***L***	***L***	***L***	***L***	***L***
0	1	1	s	x	x	x	x	x	x	x	***L***	-->	s	s	s	s	~s	x	x	x	x	x	x	x	***L***	***L***	***L***	***L***
0	0	0	s	x	x	x	x	x	x	x	***L***	-->	s	s	s	s	s	~s	x	x	x	x	x	x	x	***L***	***L***	***L***
0	0	1	s	x	x	x	x	x	x	x	***L***	-->	s	s	s	s	s	s	~s	x	x	x	x	x	x	x	***L***	***L***
0	1	0	s	x	x	x	x	x	x	x	***L***	-->	s	s	s	s	s	s	s	~s	x	x	x	x	x	x	x	***L***
0	1	1	s	x	x	x	x	x	x	x	***L***	-->	s	s	s	s	s	s	s	s	x	x	x	x	x	x	x	***L***

Selecting FP compression is a good method to reduce the data slot size. It is beneficial in systems that requires multiple data channels. Sometimes it is beneficial to have enough or not enough data slots available. Reducing the slot size also reduces the current draw, which can be important in phantom powered nodes.

The full dynamic range (24 bit = 144.49 dB) of the audio signal is preserved when data compression is enabled. The human ear can listen to sounds near the noise level in a quiet environment, but the human ear masks very quiet audio content in the presence of very loud audio content. The floating-point compression (to 20 bit) takes advantage of this psychoacoustic effect and removes low-level content in the presence of high-level audio content. The floating-point compression preserves all low-level content (here, 16-bits = 96.33 dB for 20-bit data slots) when there is no high-level audio content and supports the full dynamic range for strong audio signals (up to 144.49 dB for 20 bit data slots), always with 16 bit = 96.33 dB resolution.

Downstream Data Slots

Slave nodes can selectively receive downstream bus slots for output onto the DTXn pins. A programmable number of I2S/TDM data channels on the DRXn pins (A2B_DNOFFSET) can be skipped before the next in line channels are presented as downstream data slots to the A2B bus. As a result, this mode allows slave nodes to receive and transmit downstream data.

The A2B_DNMASK0 through A2B_DNMASK3 registers provide one bit for each possible downstream data slot. These downstream mask bits select which downstream slots are consumed by the transceiver and placed in its TX frame buffer for output over the I2S/TDM port, as governed by the downstream mask enable (A2B_LDNSLOTS.DNMASKEN) bit.

When A2B_LDNSLOTS.DNMASKEN= 0, the A2B_DNSLOTS register defines the number of downstream data slots, starting immediately after the SCF, which are passed downstream through the slave node, and the A2B_LDNSLOTS register defines the number of downstream data slots which are captured by the transceiver during the downstream portion of the superframe. The transceiver consumes and does not pass these data slots downstream to the next node. As such, a slave transceiver receives "A2B_BCDNSLOTS + A2B_DNSLOTS + A2B_LDNSLOTS" downstream data slots on the A-side transceiver and transmits "A2B_BCDNSLOTS + A2B_DNSLOTS" downstream data slots on the B-side transceiver.

NOTE: When the A2B_LDNSLOTS.DNMASKEN bit is cleared in a slave transceiver, the A2B_DNMASK0 through A2B_DNMASK3 registers are ignored.

When A2B_LDNSLOTS.DNMASKEN= 1, the A2B_LDNSLOTS register defines the number of data slots which the local node adds during the downstream portion of the superframe. These data slots are passed downstream through the local node after the A2B_DNSLOTS data slots. The most significant bit that is set in the A2B_DNMASK0 through A2B_DNMASK3 registers determines the number of slots that must be received by the transceiver (dnmaskrx) for it to then identify which individual slots are placed in its RX frame buffer for output over the I2S/TDM port. To that, a slave node receives MAX (A2B_DNSLOTS, dnmaskrx) downstream data slots on the A-side transceiver and transmits "A2B_DNSLOTS + A2B_LDNSLOTS" downstream data slots on the B-side transceiver.

NOTE: When the A2B_LDNSLOTS.DNMASKEN bit is set in a slave transceiver, the A2B_BCDNSLOTS register is ignored.

The value of the A2B_DNOFFSET register is meaningful only when the slave transceiver is configured to transmit downstream data (A2B_LDNSLOTS.DNMASKEN= 1, and the A2B_LDNSLOTS register is non-zero). Data is placed in the enabled downstream slots starting with the beginning of the RX frame buffer unless the A2B_DNOFFSET register has been programmed to apply an offset into the RX frame buffer from which it begins populating the enabled downstream slots.

The ***Slave Node Using the A2B_DNMASKn and A2B_DNOFFSET Registers*** figure is an example of how downstream data slots are used in a slave transceiver after programming the A2B_DNMASK0, A2B_DNMASK1, and A2B_DNOFFSET registers.

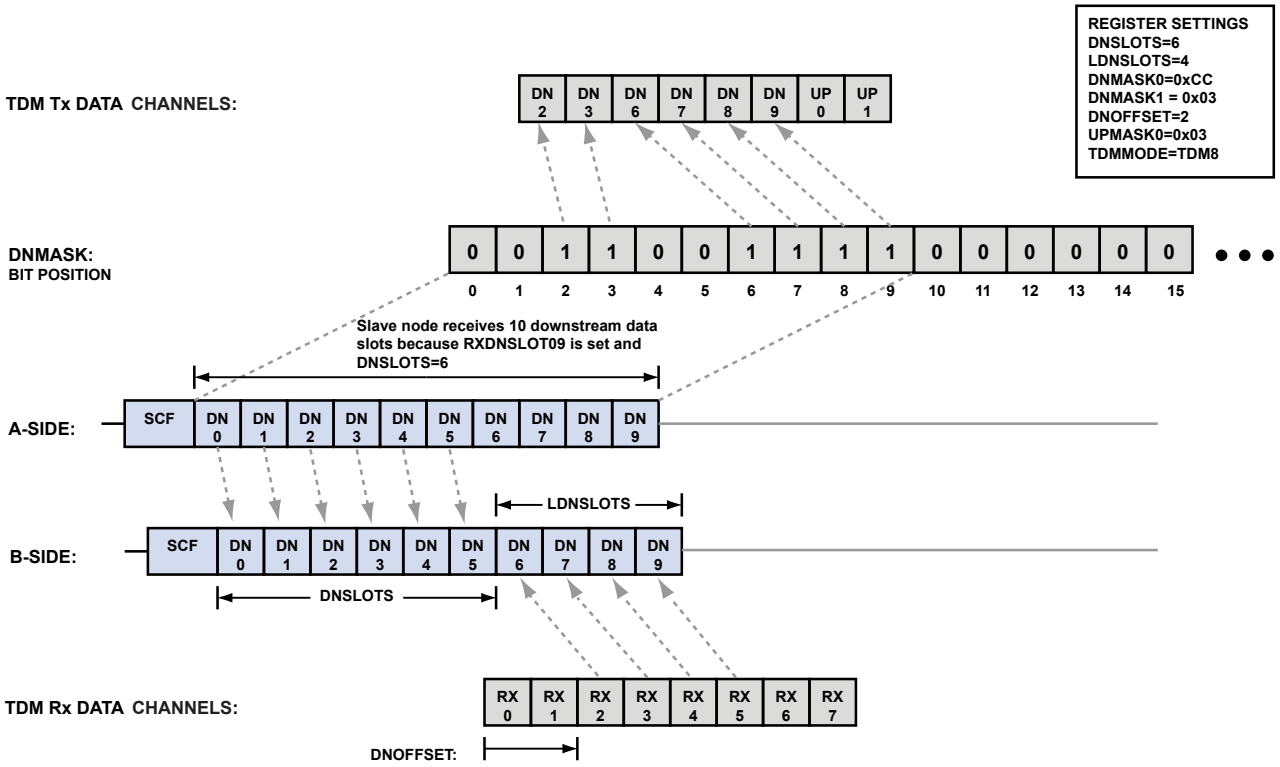

Figure 3-7: Slave Node Using the A2B_DNMASKn and A2B_DNOFFSET Registers

Upstream Data Slots

The A2B_UPSLOTS register defines the number of upstream data slots. For a master transceiver, this register defines the number of data slots that will come upstream to the master from the first-in-line slave transceiver. For a slave transceiver, this register defines the number of upstream data slots, starting immediately after the SRF with slot 0, that are passed upstream through the transceiver, whether or not that slave transceiver uses the information contained in those slots.

The A2B_LUPSLOTS register defines the number of data slots that the slave transceiver appends to the upstream portion of the superframe after the data slots being passed upstream by the slave, as defined in the A2B_UPSLOTS register. The data placed in the upstream data slots comes from the transceiver's internal RX frame buffer, as populated by its I2S/TDM/PDM port.

A slave transceiver selectively receives upstream A2B bus data slots into its TX frame buffer for output onto its DTXn pin(s) for use in the slave node. In slave transceivers, the A2B_UPMASK0 through A2B_UPMASK3 registers provide one bit for each possible upstream data slot. When a bit is set in any of these registers, the slave transceiver takes the upstream data from the corresponding slot and places it in its TX frame buffer after any received downstream data slots, which will then be output to the appropriate DTXn pin(s) via the I2S/TDM port.

The most significant bit set in the A2B_UPMASK0 through A2B_UPMASK3 registers defines the number of slots (upmaskrx) that the transceiver must receive in order to then appropriately place enabled slots into the TX frame buffer for output to the I2S/TDM port. To that, a slave transceiver receives MAX (A2B_UPSLOTS, upmaskrx)

upstream data slots on the B-side transceiver. It then transmits "A2B_UPSLOTS + A2B_LUPSLOTS" upstream data slots on the A-side transceiver.

A programmable number of I2S/TDM data channels on the DRXn pins (A2B_UPOFFSET) can be skipped before the next-in-line channels are presented as upstream data slots to the A2B bus. By default, a slave node populates the enabled upstream slots with the first entry in its RX frame buffer. The A2B_UPOFFSET register can be written to define an offset into the RX frame buffer from which it begins populating the enabled upstream slots.

The ***Slave Node Using the A2B_UPMASKn and A2B_UPOFFSET Registers*** figure provides an example of how upstream data slots are used in a slave transceiver after programming the A2B_UPMASK0, A2B_UPMASK1, and A2B_UPOFFSET registers.

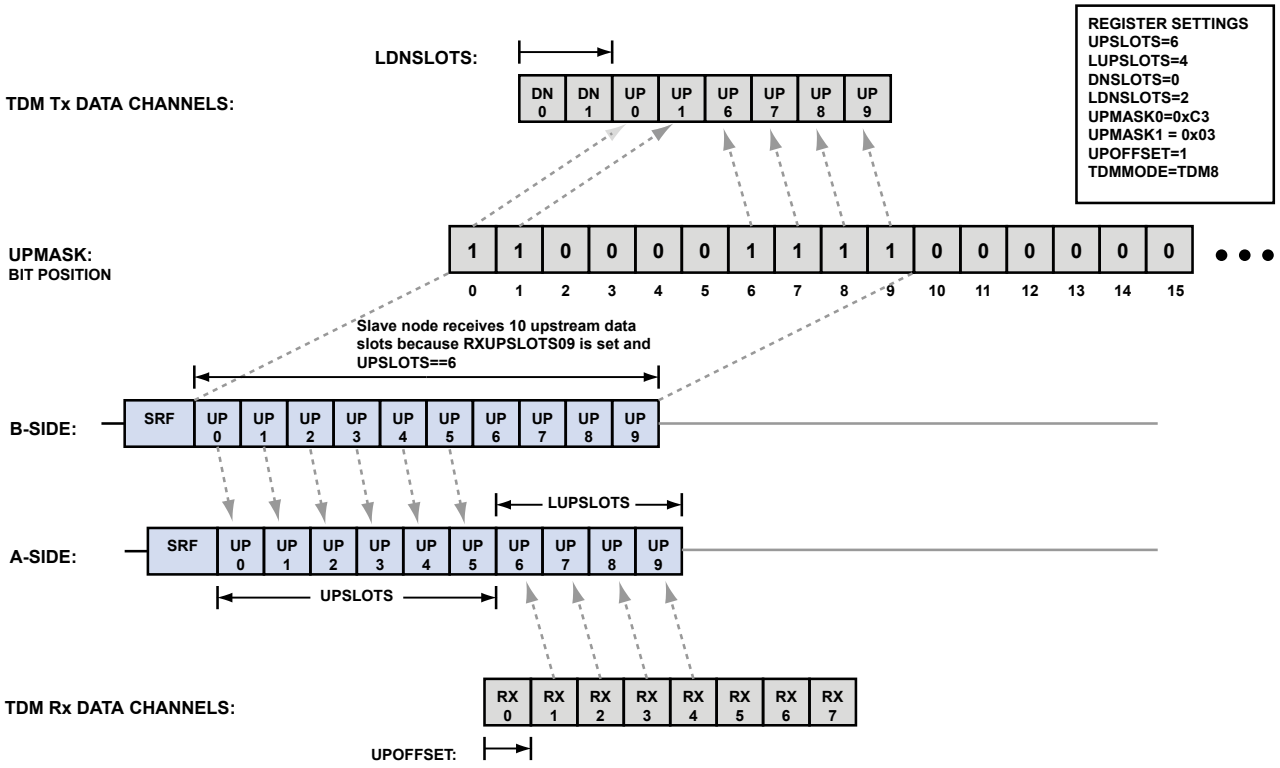

Figure 3-8: Slave Node Using the A2B_UPMASKn and A2B_UPOFFSET Registers

A2B Bandwidth

All upstream data is restricted to the same slot size (bits per slot), and all downstream data is restricted to the same slot size (bits per slot), but the downstream slot size can be different from the upstream slot size. A detailed calculation spreadsheet and the SigmaStudio software are available from Analog Devices to calculate bandwidth for all possible cases.

The ***Bandwidth Examples*** table provides bandwidth examples for 48 kHz sampled, synchronous upstream and downstream data slots. To simplify the table, every node uses the same number of upstream and downstream slots. Use up to 32 slots for upstream data and up to 32 slots for downstream data.

Table 3-5: Bandwidth Examples

Slave Nodes	Downstream Slots Per Node (Speaker)	Upstream Slots Per Node (Mics)	Slot Size (Bits Per Slot)	Sum of Down-stream Slots (Max. 32)	Sum of Up-stream Slots (Max. 32)	Sum of Downstream and Upstream Slots
9	2	2	16	18	18	36
8	4	1		32	8	40
7	4	2		28	14	42
6	5	2		30	12	42
5	6	3		30	15	45
4	8	3		32	12	44
3	10	6		30	18	48
2	16	9		32	18	50
1	32	19		32	19	51
9	3	0	24	27	0	27
8	3	0		24	0	24
7	4	0		28	0	28
6	5	0		30	0	30
5	6	0		30	0	30
4	8	0		32	0	32
3	10	1		30	3	33
2	16	1		32	2	34
1	32	2		32	2	34

I²S/TDM Port Programming Concepts

Programming the I²S/TDM interface involves selecting the mode of operation for the port, controlling how many data pins are enabled for both transmit and receive operations, and configuring the polarity and timing of the BCLK and SYNC signals relative to data.

The `A2B_I2SGCFG` and `A2B_I2SCFG` registers are used to configure the I²S/TDM port to support these various modes of operation. The ***Serial Mode Data and Clock Formats*** table provides a summary of the different data and clock formats supported by both master and slave transceivers.

Table 3-6: Serial Mode Data and Clock Formats

Bit Setting	Data and Clock Format
`A2B_I2SGCFG.EARLY` =0	SYNC pin changes in the same cycle as the MSB of Data Channel 0

Table 3-6: Serial Mode Data and Clock Formats (Continued)

Bit Setting	Data and Clock Format
A2B_I2SGCFG.EARLY =1	SYNC pin changes one cycle before the MSB of Data Channel 0
A2B_I2SGCFG.ALT =0	SYNC pin is driven high for one BCLK cycle at the start of each sampling period
A2B_I2SGCFG.ALT =1	SYNC pin is driven high at the beginning of each sampling period and low in the middle of each sampling period
A2B_I2SGCFG.INV =0	Rising edge of SYNC references the first channel (Channel 0)
A2B_I2SGCFG.INV =1	Falling edge of SYNC references the first channel (Channel 0)
A2B_I2SCFG.RXBCLKINV =0	DRX0, DRX1, and SYNC pins are sampled on the rising edge of BCLK
A2B_I2SCFG.TXBCLKINV =0	DTX0, DTX1, and SYNC pins change on the rising edge of BCLK
A2B_I2SCFG.RXBCLKINV =1	DRX0, DRX1, and SYNC pins are sampled on the falling edge of BCLK
A2B_I2SCFG.TXBCLKINV =1	DTX0, DTX1, and SYNC pins change on the falling edge of BCLK

To support more than a stereo two-channel (TDM2) signal, the A2B_I2SGCFG.TDMMODE field must be set to enable any of the supported TDM modes of operation. Once configured, this is the operating mode used for each of the enabled data pins, as controlled by the A2B_I2SCFG.RX0EN, A2B_I2SCFG.RX1EN, A2B_I2SCFG.TX0EN, and A2B_I2SCFG.TX1EN bits.

When both data pins in either direction are enabled, the interleaving feature can be enabled by setting the respective two-pin interleave (A2B_I2SCFG.RX2PINTL and A2B_I2SCFG.TX2PINTL) bit. When set, the even slot data is associated with the DTX0/DRX0 data pin, and the odd slot data is associated with the DTX1/DRX1 data pin. When cleared, the lower half of the enabled slots are associated with the DTX0/DRX0 data pin, and the upper half of the enabled slots are associated with the DTX1/DRX1 data pin. For example, if the data format is set for I²S or TDM2 mode, the ***Data Channel Structure for TDM2 Setting*** figure summarizes how the data is aligned.

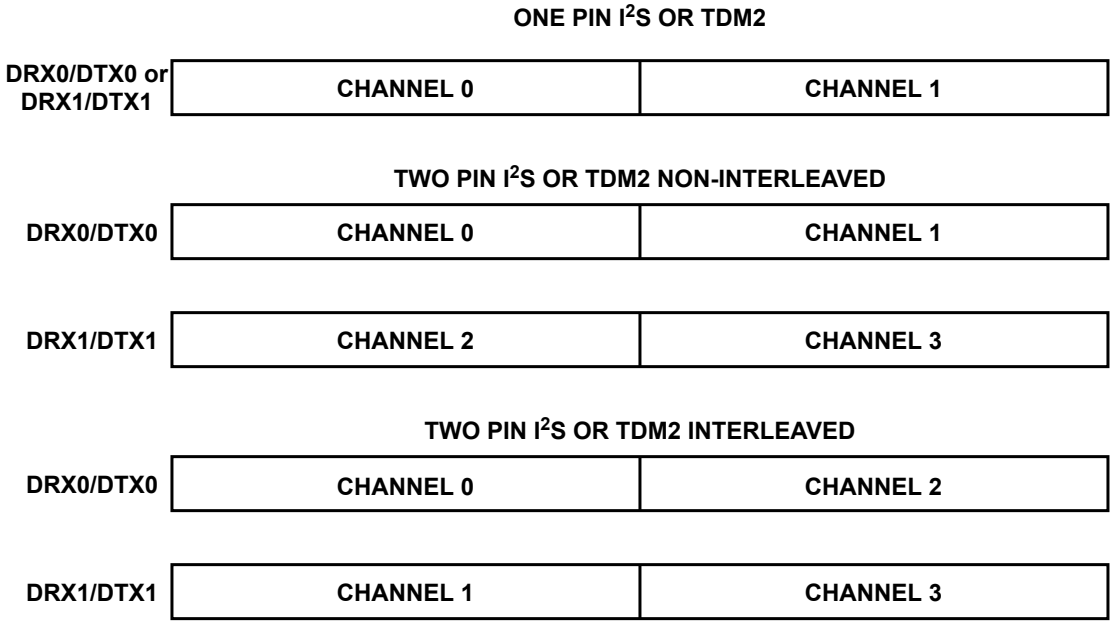

Figure 3-9: Data Channel Structure for TDM2 Setting (TDMMODE == 000)

NOTE: Single-pin transmit can be on either the DTX0 or DTX1 pin, and single-pin receive can be on either the DRX0 or DRX1 pin.

The `A2B_I2SGCFG.TDMSS` bit selects between 16-bit and 32-bit serial data for the I²S/TDM port, and it is the responsibility of the host to ensure that the appropriate timing signals are provided to accommodate the full window of data. For example, if TDM8 mode is selected (`A2B_I2SGCFG.TDMMODE` = 0b010), then the host must provide either 128 (8 x 16-bit, when `A2B_I2SGCFG.TDMSS` = 1) or 256 (8 x 32-bit, when `A2B_I2SGCFG.TDMSS` = 0) BCLK pulses for the data and the appropriate SYNC signal (to be either pulsed or held for a 50% duty cycle, per the setting of the `A2B_I2SGCFG.ALT` bit), as shown in the ***I²S/TDM8 Example Timing*** figure.

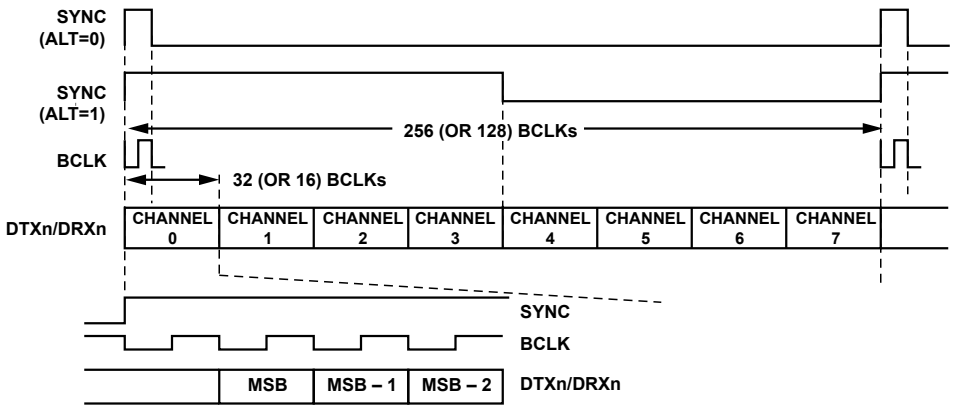

Figure 3-10: I²S/TDM8 Example Timing

As shown in the ***I²S/TDM8 Example Timing*** figure, the TDM channel data is in MSB-first format. When the data being exchanged over the A²B bus is not exactly 16-bit or 32-bit (as configured by the `A2B_I2SGCFG.TDMSS` bit), the transceiver expects the input TDM data channels to arrive in MSB-first format and disregards any lower-order bits. When outputting to the local node, the transceiver presents the received A²B slot data to the I2S/TDM port in MSB-first format with the unused lower-order bits zero-filled. For example, if the A²B slot is configured for 12-bit data (`A2B_SLOTFMT.UPSIZE` = 1 for upstream slots or `A2B_SLOTFMT.DNSIZE` = 1 for downstream slots), the 12-bit input data must be left-justified in the TDM channel, and output data consists of the 12-bit A²B slot data followed by four zero bits.

If SYNC arrives one bit earlier, it can be rephrased to data arriving one bit later than the relevant edge on the SYNC signal. The ***I²S/TDM2 to TDM16 A²B Master or Slave*** figure shows the typical timing for I²S, as well as the TDM2 to TDM16 interface modes with programmable options. Data is provided on one edge of BCLK and sampled on the opposite edge of BCLK (`A2B_I2SCFG.TXBCLKINV` ≠ `A2B_I2SCFG.RXBCLKINV`).

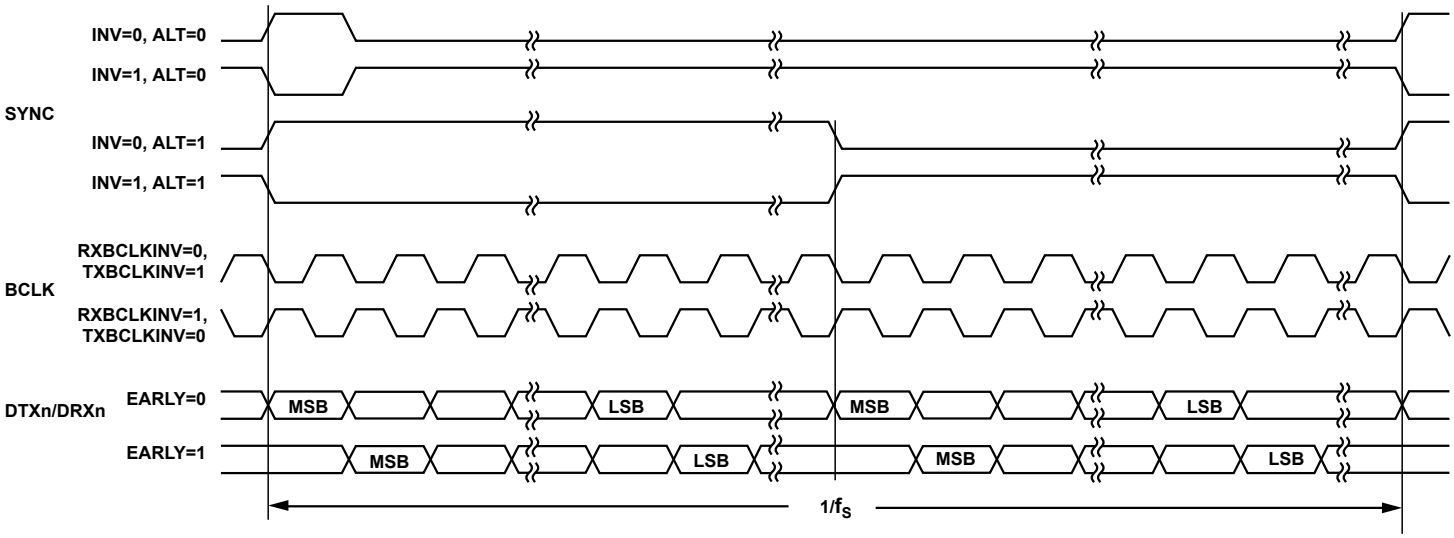

Figure 3-11: I²S/TDM2 to TDM16 A²B Master or Slave

The full 32-channel combined bandwidth is available when both data pins are enabled in TDM16 mode.

CAUTION: Be cautious if only one data pin is available for a TDM32 interface, as this increases the BCLK rate to a speed at which race conditions can occur.

The A²B master samples data on a BCLK edge and changes data on the previous, same polarity BCLK edge (`A2B_I2SCFG.TXBCLKINV` ≠ `A2B_I2SCFG.RXBCLKINV`), as shown in the ***TDM32 A²B Master*** figure.

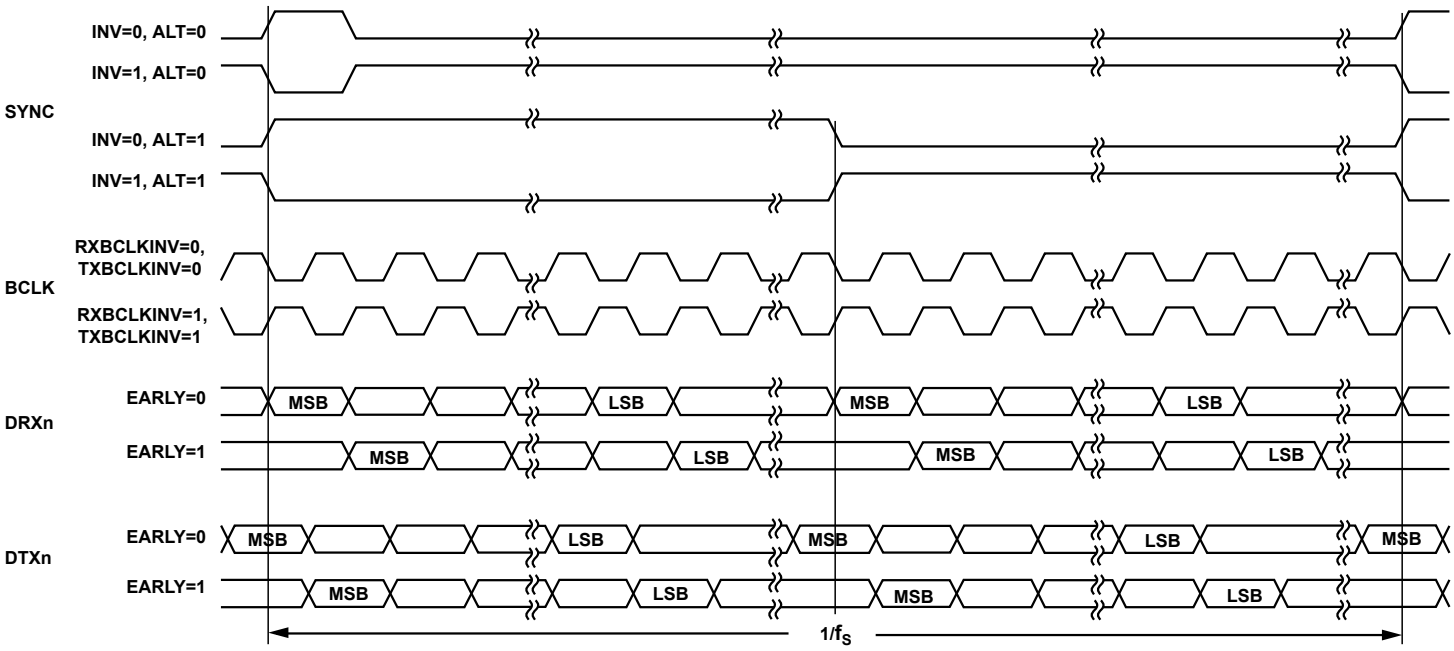

Figure 3-12: TDM32 A²B Master

The A²B slave changes data on a BCLK edge and samples data on the next, same polarity BCLK edge (`A2B_I2SCFG.TXBCLKINV` ≠ `A2B_I2SCFG.RXBCLKINV`), as shown in the ***TDM32 A²B Slave*** figure.

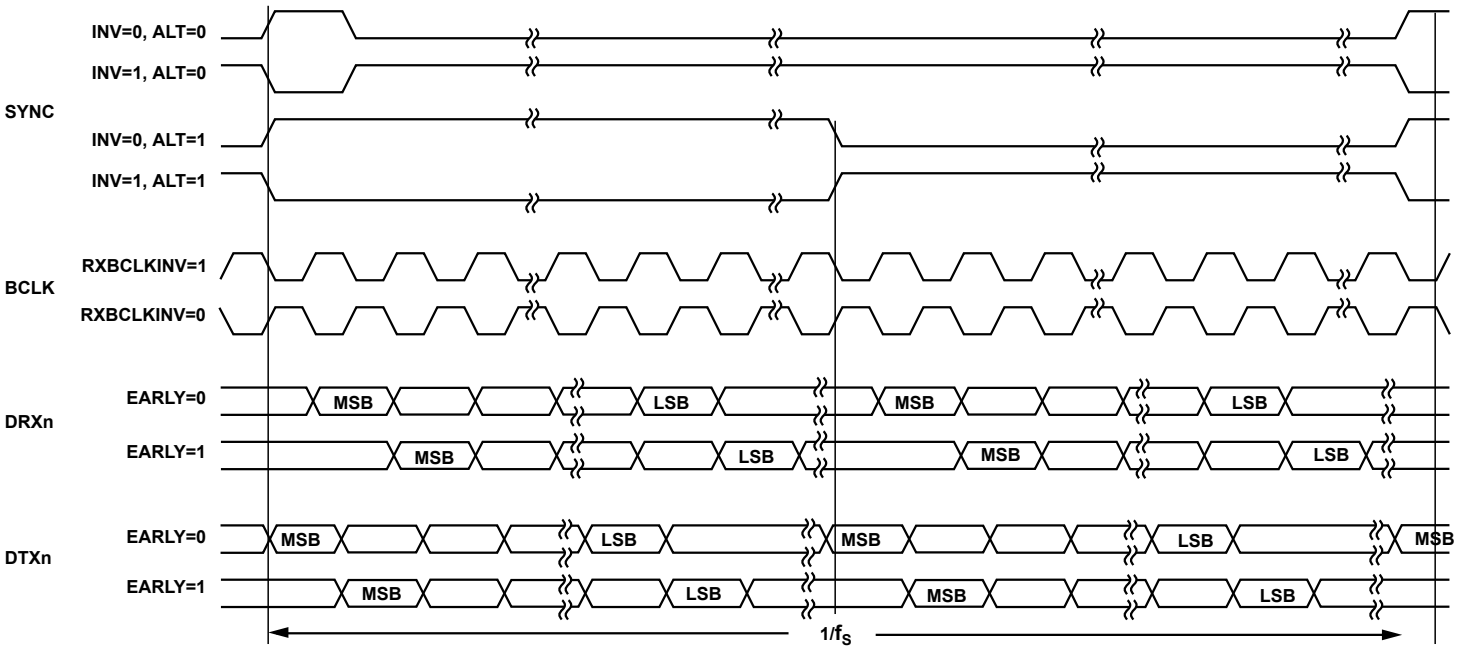

Figure 3-13: TDM32 A²B Slave

Synchronizing Slave Nodes

A²B slave nodes can all sample at exactly the same time by individually compensating for their propagation delay in the A2B_SYNCOFFSET register. Writing a non-zero value to this register adjusts the A²B bus clock ($f_{SYSBCLK}$) cycle on which the SYNC pin indicates the start of an audio frame for that particular slave transceiver. As the programmed value is the 8-bit signed two's complement representation of the integer number of SYSBCLK cycles between where the SYNC occurs and where the superframe subsequently begins, only negative values are valid.

The maximum value that can be programmed into the A2B_SYNCOFFSET register defines a SYNC signal to occur 104 SYSBCLK cycles before the start of the superframe (-104 = 0x98), but this value is only valid for the slave node that is the furthest away from the master in a fully populated A²B network topology (A2B_NODEADR.NODE = 0x09). For any slave node *n* that is nearer to the master, the valid ranges supporting a predictable transfer of I²S/TDM data to A²B slots are a function of the location of slave node *n* in the network, as governed by the formula:

(-32 - 8n) <= A2B_SYNCOFFSET <= 0

The ***Supported SYNC Offset*** table summarizes the valid settings for the A2B_SYNCOFFSET register for any given slave node in SYSBCLK cycles (Offset Range).

Table 3-7: Supported SYNC Offset

Slave Node n	Offset Range	A2B_SYNCOFFSET Range
0	-32 to 0	0xE0 to 0x00
1	-40 to 0	0xD8 to 0x00
2	-48 to 0	0xD0 to 0x00
3	-56 to 0	0xC8 to 0x00
4	-64 to 0	0xC0 to 0x00
5	-72 to 0	0xB8 to 0x00
6	-80 to 0	0xB0 to 0x00
7	-88 to 0	0xA8 to 0x00
8	-96 to 0	0xA0 to 0x00
9	-104 to 0	0x98 to 0x00

I²S Reduced Data Rate

Slave nodes can also run the I²S/TDM interface at a reduced rate frequency with respect to the superframe rate (f_{SYNCM}). The reduced-rate frequency is derived by dividing the superframe rate by a programmable set of values. Different slave nodes can be configured to run at different reduced I²S/TDM rates.

The A2B_I2SRATE.I2SRATE bit field is used to divide the superframe A²B rate down to the reduced I²S rate. It also provides a control bit, A2B_I2SRRATE.RBUS, to enable reduced-rate data slots on the bus. The A²B data slots on the bus are transmitted only once every "A2B_I2SRRATE.RRDIV + 1" superframes.

The A2B_I2SRATE.I2SRATE bit field can be used to program the division factor to 2, 4, or as set in set in the A2B_I2SRRATE.RRDIV field. The A2B_I2SRATE.SHARE bit enables the shared A²B bus slots in a reduced-rate slave node, provided the node has the I²S transmit disabled.

The A2B_I2SRRCTL register provides bits to allow a processor to track the full-rate audio frame, which contains new reduced-rate samples. The IO7 pin can be used as a strobe by setting the A2B_I2SRRCTL.ENSTRB bit, which indicates the audio frame where reduced-rate data is updated. The A2B_I2SRRCTL.STRBDIR bit configures the direction of the IO7 pin when used as a strobe. The reduced rate strobe output at the master node is based on the A2B_I2SRRATE.RRDIV field setting. When the A2B_I2SRRATE.RRDIV field is not one, the reduced rate count is maintained in each node, and the strobe output signal is generated accordingly. When the strobe is an input, it is sampled on the active edge of SYNC, and the reduced rate count is synchronized to it. The user must create a strobe signal that matches the A2B_I2SRRATE.RRDIV setting.

The A2B_I2SRRSOFFS register provides a bit field to move the SYNC edge in a reduced-rate slave in superframe increments.

The ***Reduced Data Rate*** figure shows how the upstream slots from the transceiver can reduce the superframe rate on the bus, allowing the slave nodes to run at a reduced-sample frequency with both sharing disabled (A2B_I2SRATE.SHARE = 0) and enabled (A2B_I2SRATE.SHARE = 1). This figure is drawn for a system with one master and one slave.

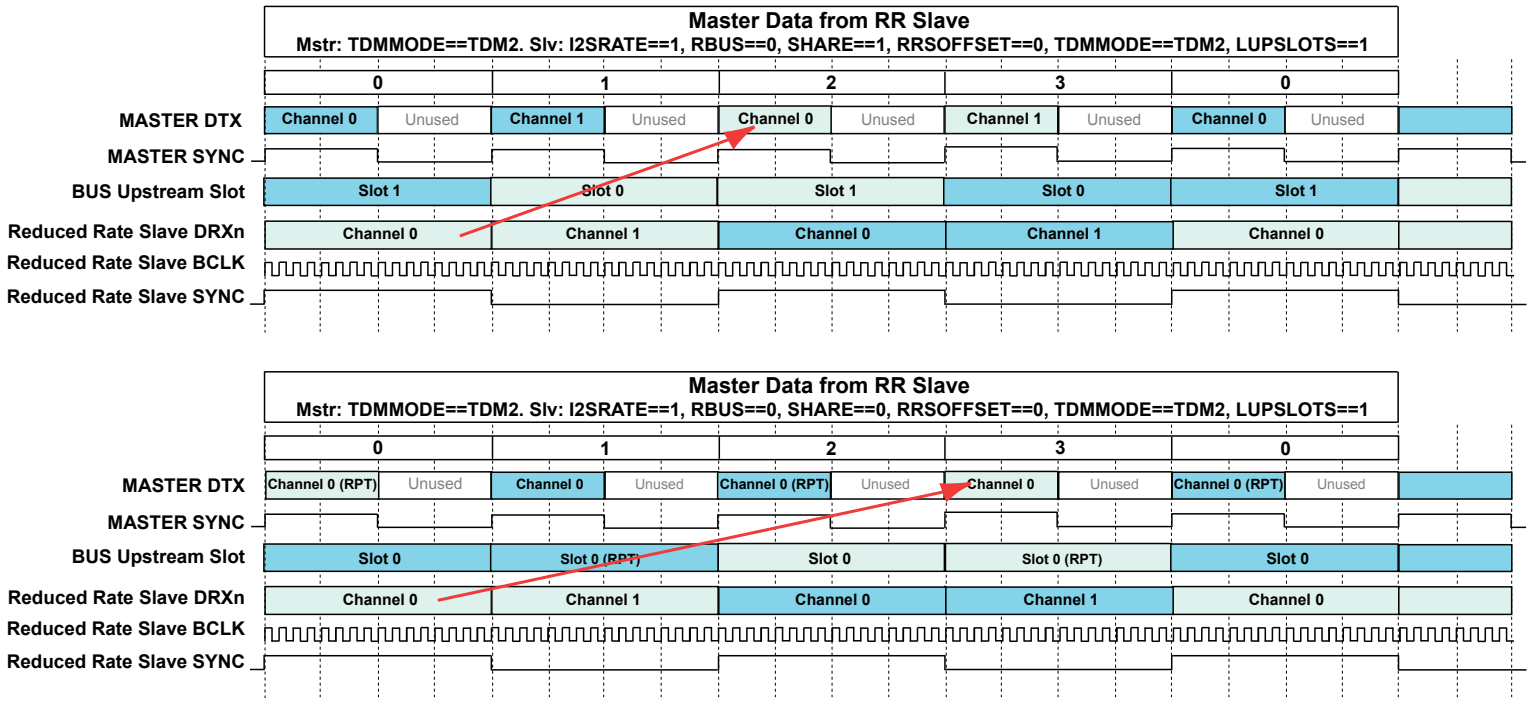

Figure 3-14: Reduced Data Rate

The following table shows the I²S/TDM sampling rates categorized into system modes for the reduced rate.

Table 3-8: I²S/TDM Sampling Rates Categorized into System Modes for Reduced Rate

Mode	Host I²S/TDM Rate	Bus Data Slots	Slave Rate(s)	Channels
1	Set in A2B_I2SRATE.RRDIV	Set in A2B_I2SRATE.RRDIV	Set in A2B_I2SRATE.RRDIV	1 - 32
2	48 kHz	Set in A2B_I2SRATE.RRDIV	Set in A2B_I2SRATE.RRDIV	1 - 32
3	48 kHz	48 kHz	Set in A2B_I2SRATE.RRDIV	1 - 32
4	48 kHz	48 kHz	Set in A2B_I2SRATE.RRDIV	1 - 128
5	48 kHz	48 kHz	Set in A2B_I2SRATE.RRDIV, 1/4x, 1/2x, 1x, 2x, 4x	1 - 32

The reduced rate feature allows system designers to add the following functionality:

1. Slave nodes can run the I²S/TDM interface at a reduced rate divided from the superframe rate, as divided down from the superframe rate. For example, reduced rates for a 48 kHz superframe rate are 24 kHz, 12 kHz, 6 kHz, 4 kHz, 3 kHz, 2.4 kHz, 2 kHz, 1.71 kHz, or 1.5 kHz. The I²S/TDM RX data on the slave can be sent either upstream or downstream at the reduced rate.

Different slave nodes can run at different reduced I²S/TDM rate.

2. The SYNC signal of the reduced-rate slave node can be adjusted in superframe increments to ensure minimum latency on the delivery of reduced-rate data.
3. Control of the BCLK signal generation can minimize a delay by quick sampling at the reduced-rate I²S data (for example, within a 48 kHz I²S/TDM frame) or sampling at the reduced I²S/TDM rate.
4. Options to notify a processor when the reduced-rate I²S/TDM data channels are updated.
5. Option to run the bus data slots at the full, continuous audio rate (nominally 48 kHz) or a reduced rate. The rate can be reduced by:
    a. Skipping data slots for superframes that do not contain data (for example, only reduced sampling rate microphone nodes on the A²B bus). This approach saves power by reducing the bus activity level but does not increase channel bandwidth on the bus. When the same A²B data slots are shared between multiple I²S/TDM channels in a node, the program cannot skip the A²B data slots.
    b. Time-dividing bus data slots of a node into multiple I²S/TDM channels and not skipping data slots for superframes. This approach is used if different types of slave nodes connecting on the same A²B bus (for example, a multi-axis accelerometer node with a microphone or amp nodes on the same bus). The bus must run at the full-data rate to allow for A²B data slot sharing. This approach provides for increased channel bandwidth on the bus by allowing reduced-rate slave nodes to time-multiplex I²S/TDM data words over bus data slots.
        - Slave nodes running at ½ rate can use 2:1 time multiplexing (two I²S/TDM channels in the same slave node alternate on one A²B slot).
        - Slave nodes running at lower rates can use 4:1 time multiplexing (four I²S/TDM channels in the same slave node alternate on one A²B slot).
        - Time multiplexing of A²B data slots beyond 4:1 is not supported.
        - Time multiplexing of A²B data slots between nodes is not supported.
        - The bus must be run with A²B data slots at the full, continuous audio rate for data slots to be shared.
        - The I²S/TDM RX reduced rate data can be transmitted upstream or downstream.

I²S Reduced Rate Restrictions

Observe the following general restrictions when using the I²S reduced rate feature.

- Each slave node can only run at a single I²S/TDM rate.
- Configure slave nodes running at a reduced I²S/TDM rate for the I²S/TDM RX data, not the I²S/TDM TX data. This means that the reduced-rate slave nodes must have `A2B_I2SCFG.TX0EN` = 0 and `A2B_I2SCFG.TX1EN` = 0.

- If A2B_I2SRRATE.RBUS is set and a reduced rate is configured (A2B_I2SRRATE.RRDIV > 1), slave nodes must have an A2B_I2SRATE.I2SRATE value of 0 (SFF x 1) or 3 (SFF / A2B_I2SRRATE.RRDIV).

Restrictions on Data Slot Sharing (A2B_I2SRATE.SHARE = 1)

Observe the following data slot sharing restrictions when using the I²S reduced rate feature.

- The bus must run at the full-data rate (A2B_I2SRRATE.RBUS = 0) to allow for A²B data slot sharing. A²B data slot skipping cannot be used when the same A²B data slots are shared between multiple I²S/TDM channels in a node.
- Data slots on the A²B bus produced by a reduced-rate slave with A2B_I2SRATE.SHARE = 1 must be received from the A²B bus by full- or increased-rate nodes.
- If the A2B_I2SRATE.SHARE bit is set in a reduced-rate slave, the maximum synchronization offset is one superframe (A2B_I2SRRSOFFS.RRSOFFSET must be 0 or 1).

If the A2B_I2SRATE.SHARE bit is set in a reduced-rate slave and there is no synchronization offset (A2B_I2SRRSOFFS.RRSOFFSET = 0), there is a further constraint on the node programming relative to *N* (the number of usable up and down slots). For example, if TDMS is the number of slots per frame on one pin of a reduced-rate slave node (which is 2, 4, 8, 16, or 32), *N* is calculated as shown in the following table:

I²S/TDM Divide Ratio	*Number of Slots (N)*
2	TDMS >> 1
4	(TDMS >> 1) + (TDMS >> 2)
> 4	(TDMS >> 1) + (TDMS >> 2) + (TDMS >> 3)

If the reduced-rate slave has the A2B_I2SCFG.RX0EN, A2B_I2SCFG.RX1EN, and A2B_I2SCFG.RX2PINTL bits all set, "A2B_LUPSLOTS + A2B_UPOFFSET" must be <= 2*N*. Otherwise, "A2B_LUPSLOTS + A2B_UPOFFSET" must be ≤ *N*.

If the reduced-rate slave is generating downstream data slots (A2B_LDNSLOTS.DNMASKEN= 1), the same constraint applies to "A2B_LDNSLOTS + A2B_DNOFFSET".

Restrictions on Alternate BCLK Rate (A2B_I2SRATE.BCLKRATE)

Observe the following alternate BCLK rate restrictions when using the I²S reduced rate feature.

- In a reduced-rate slave node, if the I²S rate setting is SFF / 2 (A2B_I2SRATE.I2SRATE = 1), do not set the BCLK frequency to SYNC x 4096 (A2B_I2SRATE.BCLKRATE != 2).
- If the system-level reduced rate divisor is 1 (A2B_I2SRRATE.RRDIV = 1) and the I²S rate setting is "SFF / A2B_I2SRRATE.RRDIV" (A2B_I2SRATE.I2SRATE = 3), do not set the BCLK frequency to "SYNC x 2048" (A2B_I2SRATE.BCLKRATE = 1) or "SYNC x 4096" (A2B_I2SRATE.BCLKRATE = 2).

- If the system-level reduced rate divisor is 2 (A2B_I2SRRATE.RRDIV = 2) and the I²S rate setting is "SFF / A2B_I2SRRATE.RRDIV" (A2B_I2SRATE.I2SRATE = 3), do not set the BCLK frequency to "SYNC x 4096" (A2B_I2SRATE.BCLKRATE = 2).
- If the BCLK frequency is not determined by the value programmed in the A2B_I2SGCFG register (A2B_I2SRATE.BCLKRATE != 0) in a reduced rate slave, the synchronization offset cannot exceed 1 superframe (A2B_I2SRRSOFFS.RRSOFFSET < 2).

I²S Increased Data Rate

The A²B slave transceiver supports increased sampling rates at the I²S/TDM interface with respect to the superframe rate (f_{SYNCM}). The local sampling rate of the slave can be programmed to 1 × f_{SYNCM}, 2 × f_{SYNCM}, or 4 × f_{SYNCM} in the A2B_I2SRATE register. For example, given a 48 kHz superframe frequency, the local sampling rate can be set to 48 kHz, 96 kHz, or 192 kHz, respectively. The ***Increased Data Rate*** figure shows how the downstream and upstream slots from the A²B superframe are distributed on the DTX0/DTX1 and DRX0/DRX1 pins in the slave transceiver for different A2B_I2SRATE bit settings (with A2B_I2SRATE.REDUCE = 0) in a system with one master and one slave.

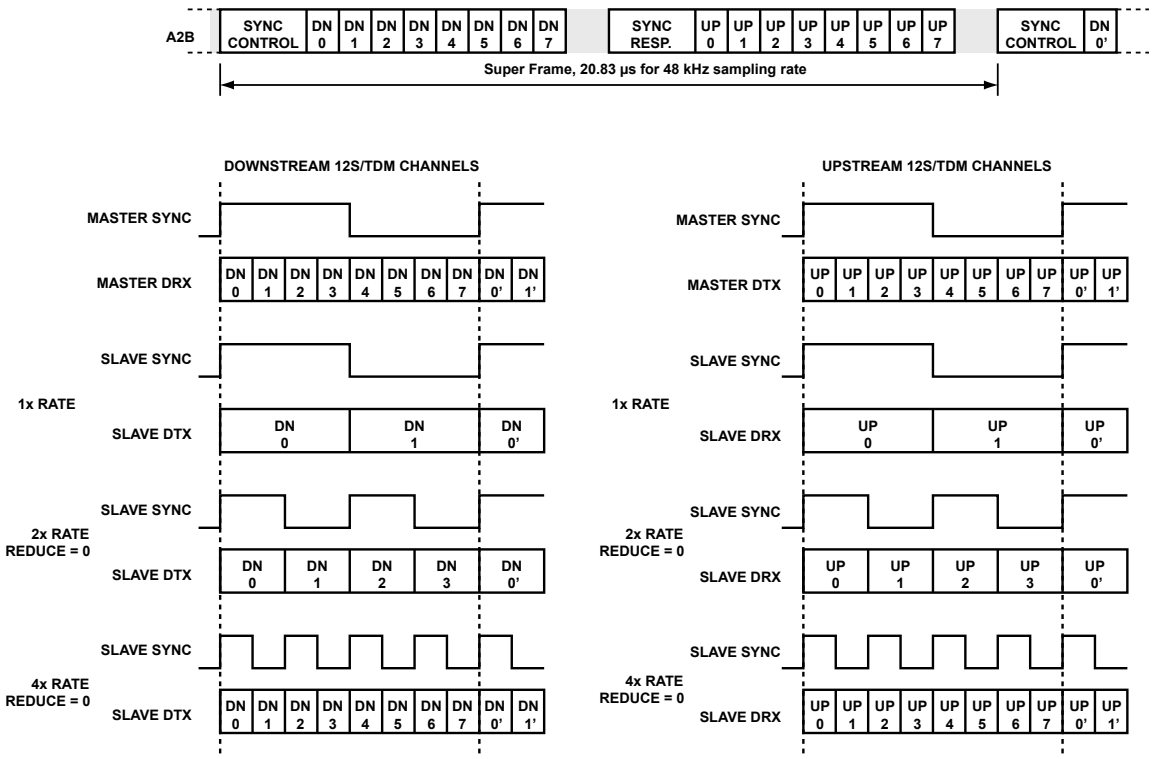

Figure 3-15: Increased Data Rate

The ***Increased Data Rate Example*** figure further illustrates the behavior of the A2B_I2SRATE register settings based on an example system. In the figure, both slave transceivers (S1 and S2) are set to 2 × f_{SYNCM} rate mode. However, S1 has the A2B_I2SRATE.REDUCE bit set to 1. The waveforms in the figure illustrate the effect of the A2B_I2SRATE.REDUCE bit for both upstream and downstream slots. When the A2B_I2SRATE.REDUCE bit

is set, only the first two channels on the DRX0/DRX1 pin are used for the upstream slots, and the other two channels are ignored for 2 × f_{SYNCM} rate. For the DTX0/DTX1 transmitter, the two local downstream slots are duplicated on the DTX0/DTX1 pins for a 2 × f_{SYNCM} rate when the `A2B_I2SRATE.REDUCE` bit is set.

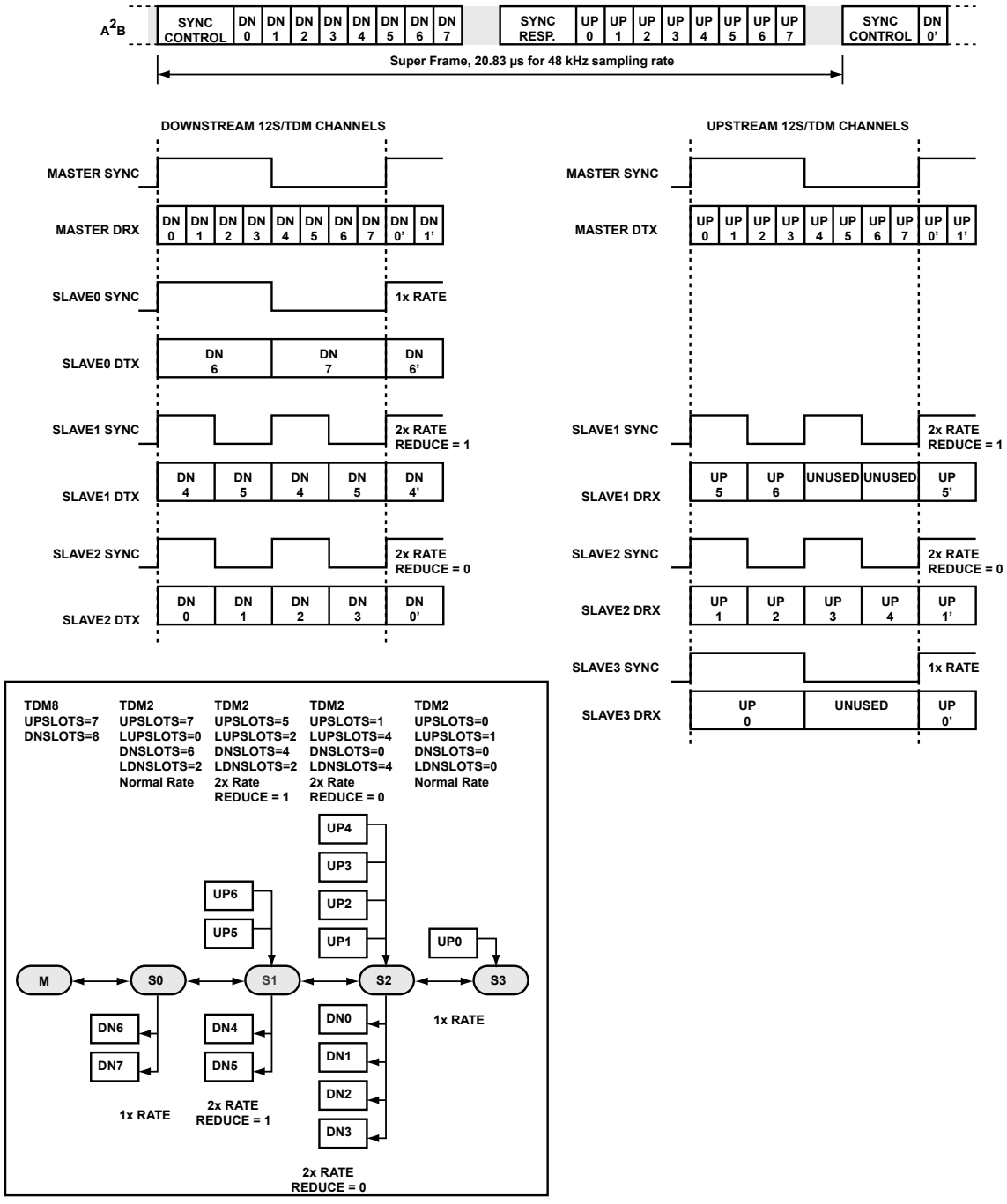

Figure 3-16: Increased Data Rate Example

GPIO Over Distance

This feature allows GPIO communication to occur over the A^{2}B bus without host intervention after initial programming. The host is only required to initialize the GPIO over distance feature through the use of virtual ports. The GPIO over distance functionality has the following features:

- Eight parallel 1-bit virtual ports, managed by the master node. The master node can read the state of each virtual port can be read in the A2B_GPIODDAT register.
- Flexible mapping scheme of GPIO pins to virtual ports 0 through 7.
- GPIO pins can be configured as inputs that update the content of the A2B_GPIODDAT register or as outputs that reflect the content of the A2B_GPIODDAT register.
- When multiple virtual ports are mapped to one GPIO output pin, the values are OR'ed together.
- When multiple GPIO input pins are mapped to one virtual port, the values are OR'ed together even if they are from multiple nodes.

Configuration

Before attempting to configure the GPIO over distance function on a given pin, first verify that it is available for GPIO, as shown in the ***GPIO Pin Configuration*** table.

Table 3-9: GPIO Pin Configuration

IO Bit	Pin Name	Pin Available for GPIO in Master	Pin Available for GPIO in Slave
IO0	IRQ/IO0	Never	Always
IO1	ADR1/IO1	If A2B_CLK1CFG.CLK1EN = 0	
IO2	ADR2/IO2	If A2B_CLK2CFG.CLK2EN = 0	
IO3	DTX0/IO3	If A2B_I2SCFG.TX0EN = 0	
IO4	DTX1/IO4	If A2B_I2SCFG.TX1EN = A2B_I2SGCFG.RXONDTX1 = 0	
IO5	DRX0/IO5	If A2B_I2SCFG.RX0EN = A2B_PDMCTL.PDM0EN = 0	
IO6	DRX1/IO6	If A2B_I2SCFG.RX1EN = A2B_PDMCTL.PDM1EN = 0	
IO7	PDMCLK/IO7	If A2B_PDMCTL.PDM0EN = A2B_PDMCTL.PDM1EN = A2B_PDMCTL2.PDMALTCLK = A2B_I2SRRCTL.ENSTRB = 0	

If the pin is available as GPIO, GPIO over distance is enabled by setting the appropriate enable bit in the GPIO over distance enable (A2B_GPIODEN) register. When a bit is set, the corresponding GPIO pin can then be mapped to one or more GPIO over distance virtual ports using the GPIO over distance mask registers (A2B_GPIOD0MSK through A2B_GPIOD7MSK , corresponding to GPIO-capable pins IO0 through IO7, respectively). Bits 0 through 7 in these registers correspond to virtual ports 0 through 7, respectively. If a bit is set within one of these registers, it maps the GPIO pin associated with the register to the corresponding virtual port.

If GPIO over distance is enabled for a given GPIO-capable pin, the direction of the pin is controlled exclusively via the GPIO output enable register (A2B_GPIOOEN) rather than a combination of this register and the

complementary GPIO input enable register (A2B_GPIOIEN). When a bit in the A2B_GPIOOEN register is set, the associated GPIO pin is an output for GPIO over distance. If the bit is cleared in the A2B_GPIOOEN register, the associated GPIO pin is an input to GPIO over distance. It is not necessary to program the A2B_GPIOIEN register when using GPIO over distance for the pins of interest.

If the GPIO pin is an input (the associated bit in A2B_GPIOOEN = 0), the local node updates the virtual port(s) associated with the set bit(s) in the corresponding GPIO over distance mask registers (A2B_GPIOD0MSK.IOD0MSK through A2B_GPIOD7MSK.IOD7MSK). The virtual port values can be read in the GPIO over distance data register (A2B_GPIODDAT).

If the GPIO pin is an output (the associated bit in A2B_GPIOOEN = 1), the virtual ports that are mapped to that pin, as determined by the set bits in the associated GPIO over distance mask registers (A2B_GPIOD0MSK.IOD0MSK through A2B_GPIOD7MSK.IOD7MSK) are OR'ed together to produce the GPIO output value (the logic OR of the corresponding bits in the A2B_GPIODDAT register).

NOTE: The A2B_GPIODDAT register is read only. It is recommended that the host always read this register from the master node.

The GPIO over distance inversion register (A2B_GPIODINV) allows for inversion of GPIO pin input or output. When a bit is set in this register, the associated GPIO pin signal is inverted. The inversion is applied on the way in from the pin if the GPIO pin is an input to a virtual port (the associated bit in A2B_GPIOOEN = 0), and it is applied on the way out to the GPIO pin if the pin is an output from a virtual port (the associated bit in A2B_GPIOOEN = 1).

If multiple nodes are updating the same virtual port, the A2B_GPIODINV register settings can be used to change the behavior from wired OR to wired AND. For example, to create a wired AND of multiple, active-high GPIO bits, the GPIO inputs and GPIO outputs must be inverted.

Mapping Multiple GPIO Inputs to One Virtual Port

When more than one node has a GPIO input mapped to the same virtual port, the protocol treats the input pins as a wired OR into the virtual port. When the virtual port is low (inactive), any request to set the virtual port results in a command from the master node to update all of the A2B_GPIODDAT registers across the system.

When the virtual port is high (active), any request to clear the virtual port results in a special command from the master node to notify all of the slave nodes of the request. If any of the slave nodes reject the request, the master node sees the rejection of the request, and the A2B_GPIODDAT registers retain their values. If none of the slave nodes reject the request, the master node sees an acceptance of the request and follows up with the updated A2B_GPIODDAT value.

GPIO Over Distance Programming Examples

The following procedures describe pin mapping cases to use GPIO over distance.

NOTE: Programming GPIO over distance must be done after the nodes have been discovered. For more information on node discovery, see the Simple Discovery Flow and Appendix A: Additional Discovery Flow Examples sections.

Mapping the Master Node DRX1/IO6 Pin to the Slave 2 ADR1/IO1 Pin

The following procedure describes how to map the master node DRX1/IO6 pin to the slave 2 ADR1/IO1 pin.

1. Write 0x04 to the master node A2B_GPIOD6MSK register to map the DRX1/IO6 pin to virtual port 2.
2. Write 0x40 to the master node A2B_GPIODEN register to enable GPIO over distance access on the DRX1/IO6 pin.
3. Write 0x02 to the slave node 2 A2B_GPIOOEN register to enable GPIO output for the ADR1/IO1 pin.
4. Write 0x04 to the slave node 2 A2B_GPIOD1MSK register to map virtual port 2 to the ADR1/IO1 pin.
5. Write 0x02 to the slave node 2 A2B_GPIODEN register to enable GPIO over distance access on the ADR1/IO1 pin.

Mapping the Slave 1 DTX1/IO4 Pin to the Master Node ADR1/IO1 Pin

The following procedure describes how to map the slave 1 DTX1/IO4 pin to the master node ADR1/IO1 pin.

1. Write 0x10 to the slave node 1 A2B_GPIOD4MSK register to map the DTX1/IO4 pin to bus GPIO port 4.
2. Write 0x10 to the slave node 1 A2B_GPIODEN register to enable GPIO over distance access on the DTX1/IO4 pin.
3. Write 0x02 to the master node A2B_GPIOOEN register to enable GPIO output for the ADR1/IO1 pin.
4. Write 0x10 to the master node A2B_GPIOD1MSK register to map bus GPIO port 4 to the ADR1/IO1 pin.
5. Write 0x02 to the master node A2B_GPIODEN register to enable GPIO over distance access on the ADR1/IO1 pin.

Mapping the ADR1/IO1 Pins on Slaves 0 Through 2 to the Master Node ADR1/IO1 Pin

The following procedure describes how to map the ADR1/IO1 pin on slaves 0 through 2 to the master node ADR1/IO1 pin.

1. For slave nodes 2, 1, and 0, write 0x01 to the A2B_GPIOD1MSK register to map the ADR1/IO1 pin of each slave to bus GPIO port 0.
2. For slave nodes 2, 1, and 0, write 0x02 to the A2B_GPIODEN register to enable GPIO over distance access on the ADR1/IO1 pin of each slave.
3. Write 0x02 to the master node A2B_GPIOOEN register to enable GPIO output for the ADR1/IO1 pin.
4. Write 0x01 to the master node A2B_GPIOD1MSK register to map bus GPIO port 0 to the ADR1/IO1 pin.

5. Write 0x02 to the master node A2B_GPIODEN register to enable GPIO over distance access on the ADR1/IO1 pin.

Transceiver Identification

Every A²B transceiver has a vendor ID register (A2B_VENDOR), a product ID register (A2B_PRODUCT), and a version ID (A2B_VERSION) register to indicate to a host which A²B transceivers are present in a system. Every A²B transceiver vendor is assigned a unique vendor ID (Analog Devices A²B transceivers use 0xAD as the vendor ID). The A2B_PRODUCT and A2B_VERSION registers are assigned by the chip vendor to uniquely identify the chips and indicate A²B interoperability. The transceiver models use 0x26 (AD2426W), 0x27 (AD2427W), and 0x28 (AD2428W) as their product ID.

Every A²B transceiver also has a A2B_CAPABILITY register to identify available control interfaces and, as such, the presence of an I²C interface (A2B_CAPABILITY.I2CAVAIL=1).

Auto-Configuration System Information in EEPROM

In an A²B system, the supplier and specific product ID of each A²B node can be determined for auto-configuration if the slave modules contain a configuration memory (I²C EEPROM) with organization and content as described in Appendix C: Module ID and Module Configuration Memory. Use auto-configuration for discovery when the host has no prior knowledge of the exact system configuration. Specific configuration commands for a slave node can also be stored in the configuration memory by using the optional configuration blocks.

Standby Mode

In standby mode, there is no upstream traffic on the A²B bus. Only a minimal (19-bit) SCF exists to keep all of the slave nodes synchronized, and there is no SRF. Header count errors and CRC errors are ignored, and data slots are disabled. GPIO settings retain their values while in standby mode.

While in normal mode, the host can write to the master transceiver A2B_DATCTL register to go to standby mode, but the write does not take effect until a new structure is applied to the system. The host performs the following actions:

1. Set the A2B_DATCTL.STANDBY bit in the master transceiver to generate a broadcast write of 0x80 to set the A2B_DATCTL.STANDBY bit in all of the discovered slave nodes. Writing 0x80 to the A2B_DATCTL register ensures that the data slots are disabled.
2. Set the A2B_CONTROL.NEWSTRCT bit in the master transceiver to apply the new structure.

After the new structure is applied, the system transitions into standby mode. The host can move the system back to normal mode by writing 0x00 to the A2B_DATCTL register in the master node. This instruction generates a broadcast write of 0x00 to the A2B_DATCTL register in all of the slave nodes. The master node provides the standby done interrupt to the host (A2B_INTTYPE = 0xFE) when the system is back in normal mode.

Bus Monitor Support

Bus monitor mode enables the transceiver to act as a passive automotive audio bus monitor, also referred to as a *sniffer*. The A^{2}B test equipment uses this mode. Only the host processor can allow bus monitors on A^{2}B bus segments to monitor the synchronous data content. To permit this synchronous data monitoring, the host must set the `A2B_DATCTL.ENDSNIFF` bit in the master transceiver. This configuration triggers an A^{2}B bus broadcast of the information to the attached bus monitor devices.

The ***Bus Monitor Behavior*** figure shows a bus monitor node inserted between slaves 0 and 1 in an A^{2}B network.

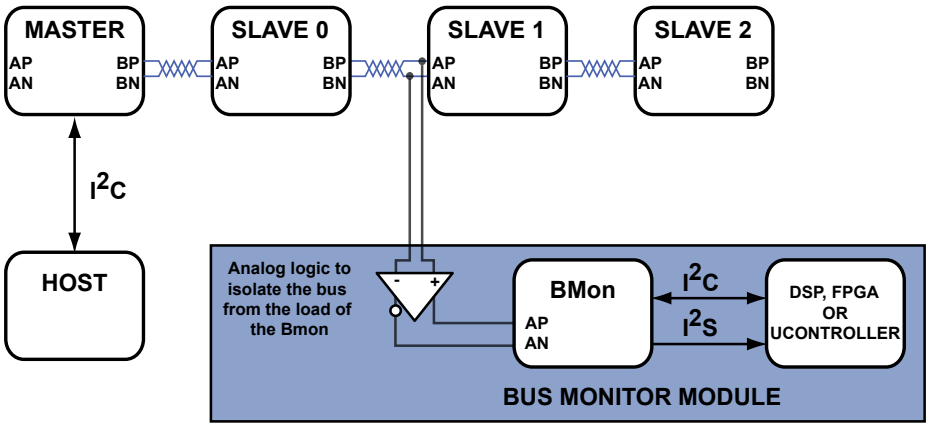

Figure 3-17: Bus Monitor Behavior

A bus monitor is passive in the system; it does not respond to bus synchronization control frames (SCFs) or contribute any data to the bus. It only uses the A-side transceiver while the B-side transceiver is deactivated. When in bus monitor mode, the transceiver synchronizes itself to SCFs and may snoop SCF control writes to configure its bus interface to match the downstream node being monitored. The A^{2}B bus monitor transceiver uses its I^{2}S/TDM port to transmit A^{2}B bus traffic to a protocol analyzer circuit.

A bus monitor node behaves as follows:

1. The B-Side (downstream) transceiver is disabled.
2. The A-Side (upstream) transceiver is enabled to receive only (not to transmit).
3. SRF generation is disabled.
4. The I^{2}S/TDM interface is configured for 32-bit data width:
    - Downstream SCFs are transmitted on the DTX0 pin
    - Upstream SRFs are transmitted on the DTX1 pin

- Data slot bits can only stream out of the DTXn pins if the A^{2}B bus master is programmed to enable this feature
  - Downstream slots are streamed out on the DTX0 pin
  - Upstream slots are streamed out on the DTX1 pin
  - If there are more data slots on the A^{2}B bus than there are available I^{2}S/TDM channels, then a programmable offset determines which data slots to monitor on the I^{2}S/TDM channels

NOTE: When the bus monitor receiver is disabled, an external switch must be used to control the LVDS traffic going to the A-side of a transceiver in bus monitor mode.

A bus monitor that attaches to an A^{2}B bus after discovery and initialization can miss the broadcast and therefore has the monitoring of synchronous data slots disabled. The preferred method is to attach bus monitors before initialization and discovery. Alternatively, for full support of bus monitors that must see data slots but attach after discovery, the host can perform regular writes to the A2B_DATCTL register to generate the enable data slot sniffing broadcasts. The bus monitor node microcontroller must set the A2B_BMMCFG.BMMEN bit to enable bus monitor mode and can further configure the bus monitor transceiver when attaching to and detaching from the A^{2}B bus:

- The A2B_BMMCFG.BMMNDSC bit determines whether the bus monitor attaches before or after system discovery and initialization. When cleared (= 0), the monitor attaches before A^{2}B discovery, so the discovery sequence sets the bus timing properties automatically. When set (= 1), the bus timing properties must be set by the bus monitor node microcontroller using local I^{2}C register writes.
- The A2B_BMMCFG.BMMRXEN bit is used to keep the LVDS A-side transceiver input static while the bus monitor is being attached. It is also used to reinitiate the bus monitor lock sequence without physically detaching the bus monitor node.

Besides configuring and enabling bus monitor mode in the A2B_BMMCFG register, the use of bus monitor mode affects the meaning and settings of bits in the following A^{2}B registers:

- I^{2}S Global Configuration register (A2B_I2SGCFG)
  - The A2B_I2SGCFG.INV, A2B_I2SGCFG.EARLY and A2B_I2SGCFG.ALT bits must be programmed to match the interface of the protocol analyzer
  - The A2B_I2SGCFG.TDMSS bit must be programmed to 0 for 32-bit TDM slot size
  - The A2B_I2SGCFG.TDMMODE field must be set to match the protocol analyzer's capabilities:
    - TDM2 allows monitoring of SCF and SRF frames
    - TDM4 allows monitoring of SCF and SRF frames, as well as up to two upstream and two downstream data slots simultaneously
    - TDM8 allows parallel monitoring of SCF and SRF frames, as well as up to six upstream and six downstream data slots simultaneously

    - TDM16 allows parallel monitoring of SCF and SRF frames, as well as up to 14 upstream and 14 downstream data slots simultaneously
    - TDM32 allows parallel monitoring of SCF and SRF frames, as well as up to 30 upstream and 30 downstream data slots simultaneously
- I²S Configuration register (A2B_I2SCFG)
    - Setting the A2B_I2SCFG.TX0EN bit enables output of downstream data on the DTX0 pin
    - Setting the A2B_I2SCFG.TX1EN bit enables output of upstream data on the DTX1 pin
    - Set the A2B_I2SCFG.TXBCLKINV bit to match the interface of the protocol analyzer
    - The A2B_I2SCFG.TX2PINTL, A2B_I2SCFG.RXBCLKINV, and A2B_I2SCFG.RX0EN bits must be programmed to 0
- Local Upstream Slots Offset register (A2B_UPOFFSET) – determines the offset in number of data slots between upstream data slots received on the A²B bus and upstream data slots driven onto the DTX1 pin as I²S/TDM channels. The register programs between monitoring of higher or lower index slots if the number of slots exceeds the number of transmit channels available in the selected TDM format.
- Local Downstream Slots Offset register (A2B_DNOFFSET) – determines the offset in number of data slots between the downstream data slots received on the A²B bus and the downstream data slots driven onto the DTX0 pin as I²S/TDM channels. The register programs between monitoring of higher or lower index slots if the number of slots exceeds the number of transmit channels available in the selected TDM format.

I²S/TDM Channel Format

In standby mode, there is no upstream traffic on the A²B bus. Only a minimal (19-bit) SCF exists to keep all of the slave nodes synchronized, and there is no SRF. Header count errors and CRC errors are ignored, and data slots are disabled. GPIO settings retain their values while in standby mode.

The following examples describe the I²S/TDM output format in bus monitor mode.

The DTX0 pin transmits in the first two 32-bit I²S/TDM transmit channels' downstream frame status bits, followed by the downstream control frame information. Further I²S/TDM channels, if available and allowed, carry the downstream synchronous data. The A2B_DNOFFSET register provides an offset between the downstream data slots and the data slots produced on DTX0.

The DTX1 pin transmits in the first two 32-bit I²S/TDM transmit channels' upstream frame status bits, followed by the upstream response frame information. Further I²S/TDM channels, if available and allowed, carry the upstream synchronous data. The A2B_UPOFFSET register provides an offset between upstream data slots and the data slots produced on DTX1.

During discovery and initialization, the host programs the data slot format register (A2B_SLOTFMT) in the master transceiver, which auto-broadcasts this information to the slaves. An attached bus monitor can listen to this control message and derive the slot size settings (32 bits maximum).

Data is always transmitted MSB-aligned.

The parity bits are not included in the I²S/TDM channels, but the A2B_ERRMGMT register may be used to indicate data slot errors in the LSB of the data, below the LSB of the data, or in an additional error channel following the data channels.

The ***TDM16 Downstream Example (DTX0 Pin) Registers*** figure shows the downstream data produced for monitoring on the DTX0 pin with A2B_I2SGCFG.TDMMODE = TDM16.

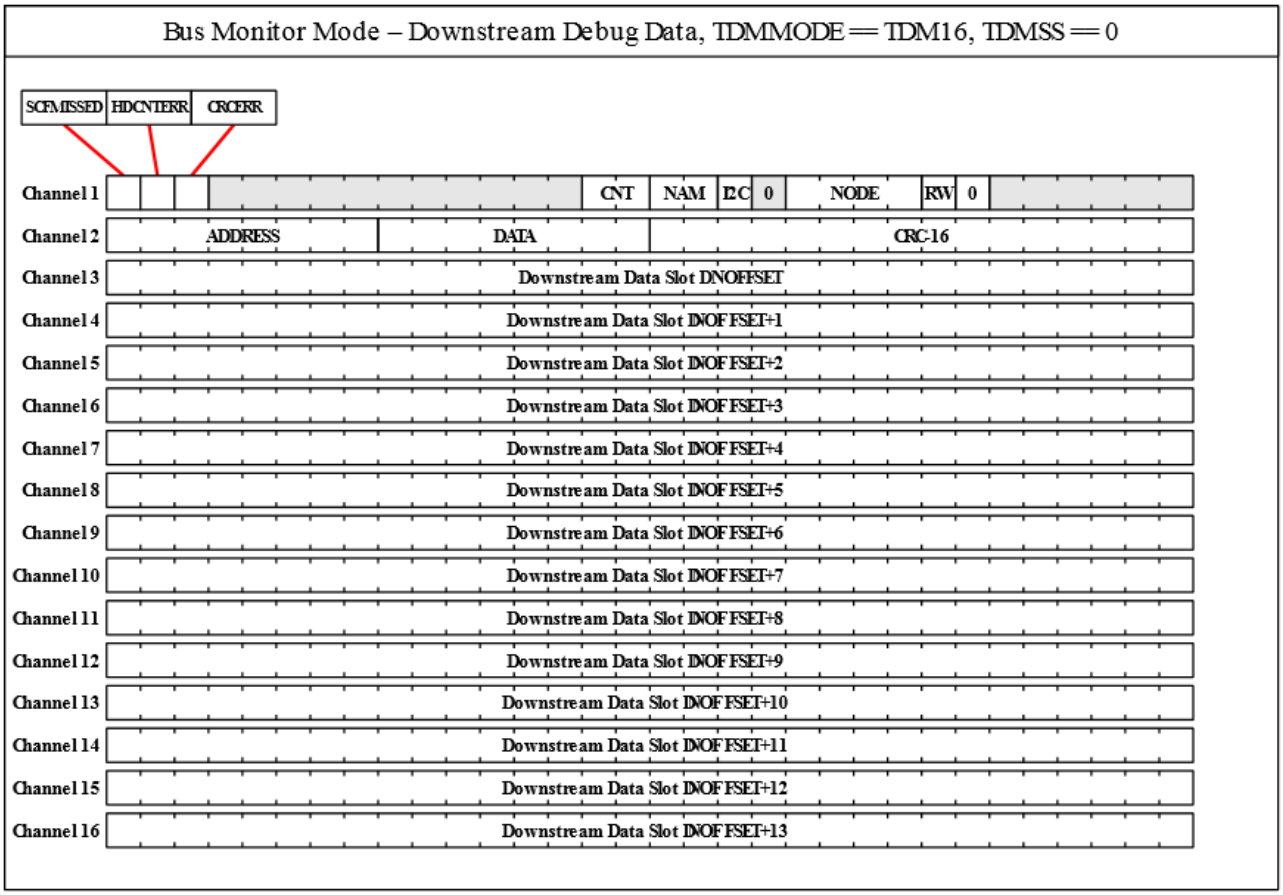

Figure 3-18: TDM16 Downstream Example (DTX0 Pin) Registers

The ***TDM16 Upstream Example (DTX1 Pin) Registers*** figure shows the upstream data produced for monitoring on the DTX1 pin with A2B_I2SGCFG.TDMMODE = TDM16.

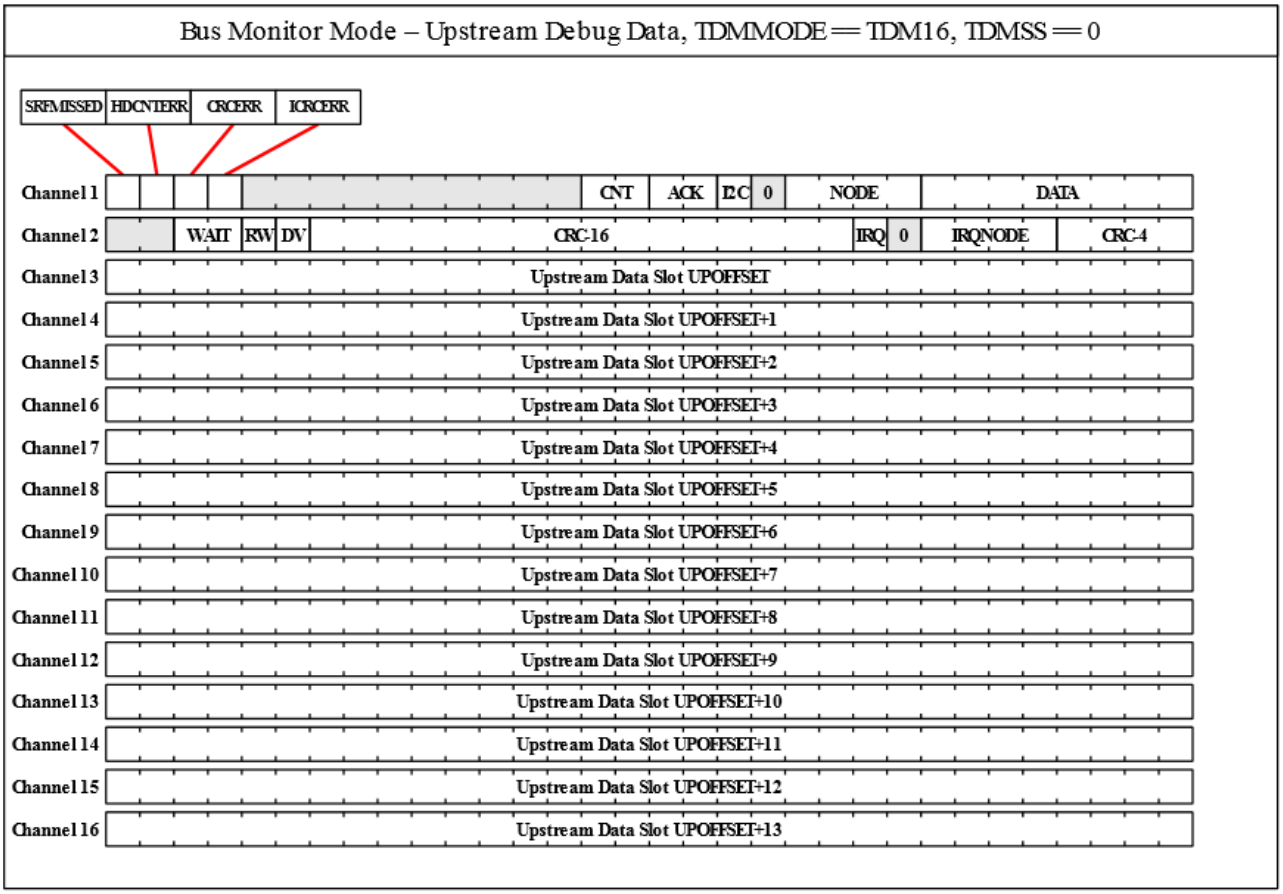

Figure 3-19: TDM16 Upstream Example (DTX1 Pin) Registers

Start Up Sequence

The required start-up sequence is a function of whether the bus monitor node attaches to the bus before or after A^{2}B system discovery and initialization, as controlled by the `A2B_BMMCFG.BMMNDSC` bit:

Before Discovery

When `A2B_BMMCFG.BMMNDSC` = 0, the following sequence of events occurs:

1. Ensure that the probed bus segment is not DC-biased (`A2B_SWCTL.ENSW` = 0 in the immediately upstream node).
2. Physically attach the bus monitor to the bus segment (probe point).
3. Set `A2B_BMMCFG.BMMEN` = 1 and `A2B_BMMCFG.BMMRXEN` = 1 through I^{2}C.
4. Configure I^{2}S/TDM transmit settings through I^{2}C in the `A2B_I2SGCFG`, `A2B_I2SCFG`, `A2B_I2SRATE`, `A2B_SYNCOFFSET` and `A2B_ERRMGMT` registers to match the desired timing and format characteristics.
5. The host starts discovery of the next-in-line node by applying bus power to the probed bus segment and then writing the `A2B_DISCVRY` register in the master transceiver.

6. The I²S/TDM interface starts transmitting after the bus monitor node locks its PLL. The IRQ pin on the bus monitor node goes high to indicate that the node found lock. This event should occur before the next-in-line node starts responding.
7. Writes to the A2B_BCDNSLOTS, A2B_LDNSLOTS, A2B_LUPSLOTS, A2B_DNSLOTS, A2B_UPSLOTS, A2B_SLOTFMTA2B_DATCTL, A2B_TESTMODE, A2B_I2SRRATE, A2B_I2SRRCTL, A2B_UPMASK0 through A2B_UPMASK3, and A2B_DNMASK0 through A2B_DNMASK3 registers in the next-in-line slave node on the probed bus segment are mirrored in the bus monitor node, where they are locally accessible over the I²C interface. Application of a new data structure on the bus (when the host sets the A2B_CONTROL.NEWSTRCT bit in the master transceiver) is also applied to the bus monitor node.
8. The DTX[1:0] pins do not transmit data slot content unless the bus monitor has seen a broadcast write resulting from the host setting the A2B_DATCTL.ENDSNIFF bit in the master transceiver.

After Discovery

When A2B_BMMCFG.BMMNDSC = 1, the following sequence of events occurs:

1. The downstream slave node of the probed bus segment is already DC-biased and discovered.
2. Set A2B_BMMCFG.BMMEN = A2B_BMMCFG.BMMNDSC = 1 through the I²C interface.
3. Physically attach the bus monitor to the bus segment (probe point).
4. Set A2B_BMMCFG.BMMRXEN = 1 through the I²C interface. After the bus monitor transceiver correctly locks to the SCFs, the IRQ pin goes high.
5. Initialize the A2B_RESPCYCS register to 0x20 through the I²C interface. The appropriate value for A2B_RESPCYCS is determined from the SRF timing and updates automatically.
6. Configure I²S/TDM transmit settings through the I²C interface in the A2B_I2SGCFG, A2B_I2SCFG, A2B_I2SRATE, A2B_SYNCOFFSET and A2B_ERRMGMT registers to match the desired timing and format characteristics.
7. If the monitoring of control and response frames alone is desired, this step can be skipped. If monitoring of data slots is desired (and the host allows access to them), configure the A2B_DNSLOTS, A2B_UPSLOTS, A2B_SLOTFMT and A2B_DATCTL registers through the I²C interface. The correct values for these registers can come from values previously stored in memory after sniffing the same bus segment during discovery and initialization. If values are completely unknown, then software can try different values to find suitable settings.
   - The A2B_DNSLOTS register represents the number of downstream data slots at the A-side transceiver circuit of the next-in-line downstream slave.
   - The A2B_UPSLOTS register represents the number of upstream data slots at the A-side transceiver circuit of the next-in-line downstream slave.
   - The A2B_SLOTFMT register represents the data slot format.

- The A2B_DATCTL.DNS and A2B_DATCTL.UPS bits must match the committed values in the downstream slave node. The DTX0 and DTX1 pins do not transmit data slots in I²S/TDM channels if these bits are not set.

8. The DTX[1:0] pins do not transmit data slot content unless the bus monitor has seen a broadcast write resulting from the host setting the A2B_DATCTL.ENDSNIFF bit in the master transceiver.

Optimizing EMC Performance

EMC performance is critical in an A²B transceiver system design. The transceivers have several programmable features that can be utilized to optimize EMC performance:

- Spread-Spectrum Clocking
- Programmable LVDS Transmit Levels
- Data-Only and Power-Only Bus Operation

Spread-Spectrum Clocking

Spread-spectrum clocking can be used to reduce narrowband emissions on a PCB. By default, spread-spectrum clocking is disabled on the transceiver, but writes to the A2B_PLLCTL register can enable spread-spectrum clocking during discovery. The A2B_PLLCTL register contains settings that enable spread-spectrum clocking for clocks that are internal to the transceiver.

If spread-spectrum clocking support is enabled for the internal clocks, spread-spectrum clocking can also be enabled for both the I²S interface and the programmed CLKOUTs. Enabling spread-spectrum clocking for internal clocks, CLKOUTs, and the I²S interface may reduce narrowband emissions by several dB on a particular node.

ATTENTION: When spread-spectrum clocking is enabled on a clock output, the TIE jitter on that clock increases.

To enable an A²B network with spread-spectrum clocking, all nodes must be set to the same depth and frequency. Follow this sequence to set the nodes:

1. Discover all slaves.
2. Configure spread-spectrum for all nodes (including the master) with a broadcast write to the A2B_PLLCTL register of each node.

For a single node with spread spectrum (including systems with the AD2421/AD2422/AD2425 models), follow this sequence:

1. Discover all nodes.
2. Configure spread spectrum (by setting the A2B_PLLCTL register) for each slave, one at a time.
   a. The A2B_PLLCTL.SSDEPTH bit is limited to setting 0x0.
   b. Adjacent nodes must have the same A2B_PLLCTL.SSFREQ setting.

NOTE: A broadcast write to the A2B_PLLCTL register is mandatory when all the nodes in the system are enabled with spread spectrum. Set the A2B_NODEADR.BRCST bit and initiate a write to the A2B_PLLCTL register with A2B_BUS_ADDR. A broadcast write effects all nodes. It occurs in the master first and then in the slave nodes during the next SCF.

Sequential programming of spread spectrum must follow the single node guidelines. The A2B_PLLCTL.SSDEPTH bit is limited to setting 0x0 for sequential programming of spread spectrum clocking, as well as in a system with a single node with spread spectrum clocking enabled.

The A2B_PLLCTL.SSMODE field can be set to protocol only or I^2S+ protocol, whether or not spread spectrum clocking is enabled.

Programmable LVDS Transmit Levels

The LVDS transmitter can be set to transmit the signal at high, medium, or low levels. Higher transmit levels yield greater immunity to EMI, while lower transmit levels can reduce emissions from the twisted-pair cables that link A^2B bus nodes together.

The LVDS transmit levels can be changed by adjusting the settings in the A2B_TXACTL (A-side) or A2B_TXBCTL (B-side) register. If a non-default transmit level is desired, A2B_TXxCTL must written on each node (during discovery) before setting the A2B_SWCTL.ENSW bit. The A2B_TXACTL.TXAOVREN enable bit must be set in order for the TXxLEVEL setting to take effect.

Data-Only and Power-Only Bus Operation

The A^2B bus can be operated without closing the PMOS switch to send a DC bias downstream. This requires that the A2B_CONTROL.SWBYP bit is set instead of the A2B_SWCTL.ENSW bit during discovery. Conversely, the A2B_SWCTL.DISNXT bit allows a DC bias to be sent downstream without the presence of data. This setting should be applied at the same time as the write to set the A2B_SWCTL.ENSW bit during discovery. These modes are used primarily for debug purposes.

Cross-Over or Straight-Through Cabling

Straight-through cables can be supported by swapping the DC-coupling at the B-side connector. For hardware designed to support straight-through cables, the A2B_CONTROL.XCVRBINV bit must be set during discovery to ensure proper operation. This is done before setting the A2B_SWCTL.ENSW bit for each slave that is connected with a straight-through cable.

IMPORTANT: Ensure that the A2B_CONTROL.XCVRBINV bit is not overwritten while doing other operations such as writing to the A2B_CONTROL.NEWSTRCT bit (which applies a new structure).

4 A2B Event Control

The A2B protocol engine contains a set of registers that provide support for interrupts to the host. These registers include:

- A2B_INTSTAT
- A2B_INTSRC
- A2B_INTTYPE
- A2B_INTPND0 through A2B_INTPND2
- A2B_INTMSK0 through A2B_INTMSK2

To register slave interrupt requests in the master node, unmask the slave interrupts in the A2B_INTMSK0 and A2B_INTMSK1 registers. In master nodes only, also unmask interrupts in the A2B_INTMSK2 register.

The active polarity of the A2B_IRQ pin is set using the A2B_PINCFG register. By default, interrupt requests are indicated with a high level on the A2B_IRQ pin and the setting of the A2B_INTSTAT.IRQ bit. An active interrupt request in the master transceiver is cleared and revised on a host read of the master transceiver A2B_INTTYPE register. This process also applies to the master node receiving an interrupt request from a slave node.

The master transceiver register (A2B_INTSRC) indicates whether the active interrupt is generated by the master node or by a slave node (where it also supplies the ID of the slave node). The A2B_INTTYPE register in the master transceiver contains information that the host uses to determine the interrupt cause. Priority logic automatically determines the value of the A2B_INTSRC and A2B_INTTYPE registers. Other pending interrupt requests can appear after reading the A2B_INTTYPE register. The A2B_IRQ pin goes low for one $f_{SYSBCLK}$ cycle (~20 ns) when the A2B_INTTYPE register is read. The pin immediately transitions to high if there are pending interrupt requests.

When masked interrupts occur, they are registered as sticky bits in the A2B_INTPND0 through A2B_INTPND2 registers but do not trigger interrupt requests. Once unmasked, any pending interrupts trigger interrupt requests following this order of priority:

- Master interrupts have priority over slave node interrupts.
- Lower slave node ID numbers take priority over higher numbers.
- Lower number A2B_INTTYPE has priority over higher number.

- A2B_INTPND0 takes priority over A2B_INTPND1, which takes priority over A2B_INTPND2.
- Lower numbered bits in the pending registers A2B_INTPND0 to A2B_INTPND2 take priority over higher numbered bits.

The IRQ signal is immediately asserted when the master transceiver receives an interrupt request from a slave.

Host Response to Interrupt Requests

When the host receives an interrupt request from the master node (indicated by the IRQ signal going high), the host can read the A2B_INTSRC and A2B_INTTYPE registers to obtain the slave node ID that generated the interrupt request and the type of interrupt request, respectively. This can be accomplished by performing a single 2-byte read, starting at the A2B_INTSRC address, which reads both registers. At the completion of the A2B_INTTYPE register read, the active interrupt is cleared and IRQ goes low if there are no further pending interrupts.

Interrupt Latency

Interrupts are signaled upstream from a slave transceiver to the master transceiver within the Synchronization Response Frame (SRF). Interrupts that engage after the beginning of the SRF (after the slave node starts driving the AP and AN pins) are signaled to the master in the SRF of the next superframe. Assuming there are no other interrupts with a higher priority that mask the IO pin interrupt in question, the latency between a slave node IO pin and master node IRQ is the sum of:

- Four SYSBCLK cycles for pin interrupt generation (81.4 ns) +
- One superframe latency to get into SRF (20,833.3 µs) +
- 64 SYSBCLK cycles for the length of the SRF (1,302.1 ns) +
- Five SYSBCLK cycles for master Rx latency (101.7 ns) +
- Two SYSBCLK cycles for IRQ logic in the master node (40.7 ns)

In addition to this total latency of 22.36 µs, there is an additional nine SYSBCLK cycles of latency for each slave that the SRF must pass through (N × 183.1 ns). For example, in a system with three slaves, a GPIO interrupt from slave 2 to the master has a maximum latency of "22.36 µs + (2 x 0.183) µs= 22.73 µs".

Error Management

The following sections provide information about error management. All data transmitted over the A^{2}B bus is checked for line code violations (DDERR) at the receiving end. Additionally, the SCF and SRF use cyclic redundant codes (CRC), and every synchronous data slot uses a parity bit for extra error detection certainty, as shown in the ***Frame Structure Details*** figure.

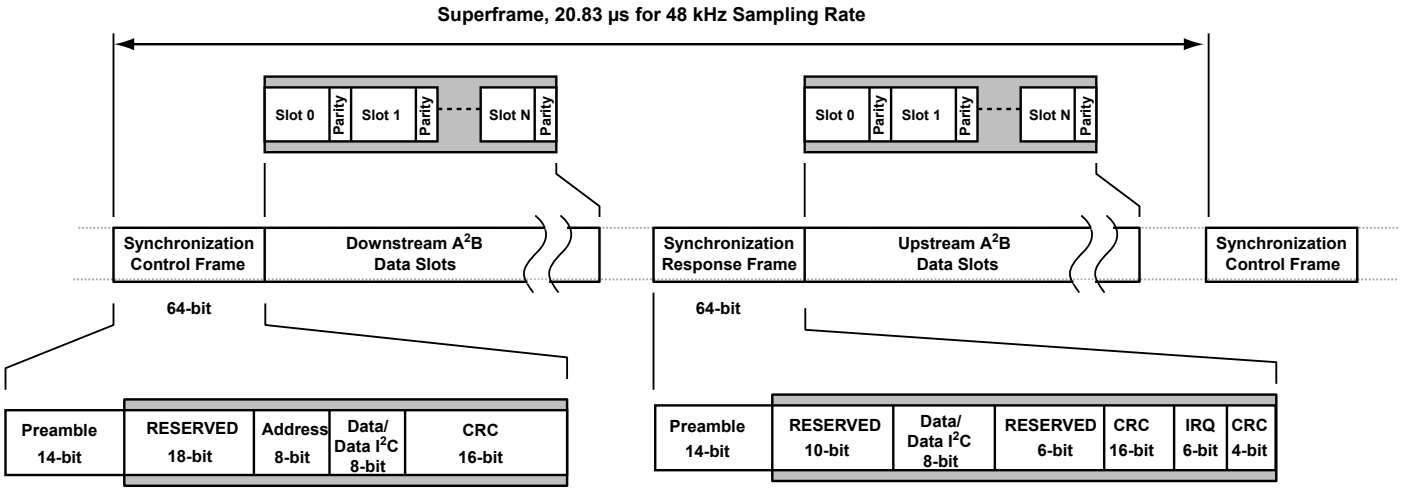

Figure 4-1: Frame Structure Details

Downstream Data Error Detection

A 16-bit cyclic redundant code (CRC) is part of any downstream control data inside the SCF. This CRC determines on the receiving side if the SCF data is corrupted during transmission.

The SCF has a preamble to indicate the start of a superframe. It provides a bit pattern that slaves use for clock and frame synchronization. If this frame sync is not detected by a slave, the error is treated as a CRC error.

Upstream Data Error Detection

A 16-bit cyclic redundant code is part of any upstream response data inside the SRF. The CRC determines on the receiving side if the SRF data is corrupted during transmission. Interrupt request fields have an extra CRC (ICRC) inside the SCF to prevent wrong interrupts from being triggered.

The SRF has a preamble to indicate the start of the response frame. It provides a bit pattern that is used for clock and frame synchronization. If this frame sync is not detected by an upstream node, the error is captured as an SRFCRCERR in the slave nodes and as a CRCERR is the master node.

Data Slot Error Correction

Possible cases for automatic correction of received data slots in a node are described as follows.

- If the frame sync preamble is not seen, all data slots received from the bus are automatically replaced with previous, good values.
- If a CRC error is detected in the SCF by a slave (`A2B_INTPND0.CRCERR` =1), all downstream data slots received from the bus are replaced with previous, good values.
- If a CRC error is detected in the SRF by the master (`A2B_INTPND0.CRCERR` =1) , all upstream data slots received from the bus are replaced with previous good values.

- If a data decoding error (A2B_INTPND0.DDERR =1) or a data parity error (A2B_INTPND0.DPERR =1) is detected within a data slot, the received erroneous data slot is replaced automatically with a previous, good slot value.

Control and Response Error Handling

When a host accesses registers over I^2C and A^2B (for example, I^2C over distance), the synchronization control frame (SCF) and the synchronization response frame (SRF) carry this data exchange. If there is a communication error in the control frame or the response frame, the master node automatically initiates a retry of the register access. The master node retries multiple times until either the access is successful or an I^2C timeout occurs in the master. During the retry time, I^2C clock stretching is applied, which signals to the host that the transaction is not completed. If there is an I^2C timeout (the I^2C timeout occurs after 30 superframes), the master flags an I2CERR interrupt, to which the host can respond.

Corrupted received interrupt requests in the master are ignored. If a real interrupt event occurs, an interrupt is automatically re-generated by the slave since it is not cleared.

Error Signaling

Any communication error, flagged in the A2B_INTPND0 or A2B_INTPND2 register, triggers an interrupt request when the corresponding interrupt is enabled in the A2B_INTMSK0 or A2B_INTMSK2 register, respectively. These interrupt requests use the A2B_IRQ pin or the A2B_INTSTAT.IRQ bit to signal the request to the host. The host can then read the A2B_INTSRC and A2B_INTTYPE registers to determine what the error is and where it occurred.

The A2B_BECCTL register selects which communication errors are counted and what counter threshold must be exceeded for an interrupt request to be generated. Using this feature, certain single-bit communication errors do not have to generate an interrupt unless they significantly accumulate over the time period since the A2B_BECNT register was last cleared. Additionally, three different methods to signal data slot errors over the I^2S/TDM interface can be selected using the A2B_ERRMGMT register (see Error Management Register for details).

A^2B Communication and Bit Errors

The A^2B communication and bit errors are:

- HDCNTERR (A2B_INTTYPE = 0)

  The SCF and SRF fields contains a 2-bit field CNT. In the SCF, the CNT field is incremented (modulo 4) from the value used in the previous superframe. In the SRF, the received value of the CNT field in the SCF is transmitted back to the master node. HDCNTERR indicates that the current node has detected a header count error. For the master node, this means that the synchronization response frame has a different CNT value than expected. For a slave node, this means that the synchronization control frame has a different value than expected.

- DDERR (A2B_INTTYPE = 1)

The DDERR error indicates a missing clock edge in the Differential Manchester data stream on the A^{2}B bus. The data decoding error is reported only on data slots that are being consumed by the particular node. A data decode error in an SCF/SRF results in a CRC error and does not raise a data decoding error.

- CRCERR (`A2B_INTTYPE` = 2)

  The CRCERR error indicates that a slave node detects a CRC error in the received SCF field. For the master node, the error indicates a CRC error in the received SRF field.

- DPERR (`A2B_INTTYPE` = 3)

  A data slot on the A^{2}B bus is protected by a parity bit. The DPERR error is reported only on data slots that are consumed by the particular node. Nodes do not check parity for the slots that are just passed through it.

- BECOVF (`A2B_INTTYPE` = 4)

  The `A2B_BECCTL.THRESHLD` field configures the number of bit errors (HDCNTERR, DDERR, CRCERR, DPERR, and ICRCERR) to be counted before the `A2B_INTPND0.BECOVF` bit is set. This threshold is useful if it is not desired to signal the interrupt for every bit error. The threshold can be set based on acceptable noise and robustness over a particular period of time. The bit error counter should be cleared periodically. Excessive bit errors set the `A2B_INTPND0.BECOVF` bit and signal the interrupt. Bit error thresholds can be independently set at master and slave nodes.

- SRFERR (`A2B_INTTYPE` = 5)

  The SRFERR error indicates that the SRF of a later node is not received prior the local timing window being expired, and the affected node generates its own SRF, which is being up-streamed to any earlier nodes. The error is valid for master and slave nodes.

- SRFCRCERR (`A2B_INTTYPE` = 6)

  The SRFCRCERR error indicates that the current slave node detected a CRC error in the SRF field. Usually, when a slave node detects a CRC error in the SRF, it flags the SRFCRCERR error bit of the node. The slave does not try to correct the error and passes the SRF as-is upward. But the exception is, in case of a response to a command, the slave node inserts its own SRF, including the CRC. In the master node, the CRCERR field is used to indicate a CRC error in the SRF.

- PWRERR (`A2B_INTTYPE` = 9-15)

  PWRERR is the mask bit for errors from the DLPS (digital line power switch) block; LDO2 is used internally for powering the DLPS block.

Slave Interrupt Handling

This section describes how slave node interrupts are internally handled by the master node. When an interrupt occurs in a slave node, the following sequence of events happens in response:

1. After the slave node interrupt occurs, the related bits in the `A2B_INTPND0`, `A2B_INTPND1`, and `A2B_MBOX0STAT` registers of the slave transceiver are set.

2. If the A2B_INTSTAT.IRQ bit is low, it gets set. The highest priority pending interrupt type is then written to the A2B_INTTYPE register.
3. The slave node begins signaling the IRQ in the interrupt field of the SRF. Any upstream slaves without an active interrupt passes this field upstream.
4. When the master node receives an interrupt field with a valid CRC and the IRQ field set, then master node sets its A2B_INTSTAT.IRQ bit if the bit is not already set. The master then updates the A2B_INTSRC register with the slave number and sets the A2B_INTTYPE register to 0x80. At this point, the IRQ pin of the master node is driven active.
5. The master automatically reads the A2B_INTTYPE register from the appropriate slave and updates its A2B_INTTYPE register. This is held off if there is a new structure being applied (A2B_CONTROL.NEWSTRCT set within the last five superframes), or if a remote I^2C stop command needs to be sent.
6. Once the A2B_INTTYPE is read, the master automatically performs a write to the appropriate slave to clear the interrupt. This is held off if there is a new structure being applied, or if a remote I^2C stop command needs to be sent. At this point, the slave stops signaling the interrupt in the SRF.
7. When the IRQ pin of the master node is asserted as a result of the slave interrupt, the host processor reads the A2B_INTSTAT and A2B_INTTYPE registers to ascertain the interrupt type and identify which slave node raised the interrupt.

If the host reads A2B_INTTYPE from the master after step 4 but before step 5 completes, the host may read 0x80 from A2B_INTTYPE . If the slave node does not drop off the bus, the A2B_INTTYPE field eventually updates.

When the host reads A2B_INTTYPE = 0x80, an additional read of the A2B_INTTYPE register is recommended to confirm the interrupt type. If a slave signals an interrupt and then drops off the bus (presumably, due to a switch fault), the next upstream slave eventually switches to being the last slave after 32 frames of missed SRFs. At this point, if the master node (not the host processor) is still internally attempting to read A2B_INTTYPE from the missing slave, the newly last slave sends a special SRF, indicating to the master that the read cannot go through. This causes A2B_INTTYPE to be set to 0xFD and the interrupt identification process to terminate. Since the missed SRF timeout is 32 superframes (after which the upstream node becomes the last node), the error type 0xFD is unlikely.

In other words, the slave A2B_INTTYPE read error (0xFD) interrupt occurs when the master is attempting to read A2B_INTTYPE from a slave based on a received interrupt but receives a response from an upstream slave indicating that slave is now the last slave. The main difference between A2B_INTTYPE = 0xFD and A2B_INTTYPE = 0x80 is that A2B_INTTYPE = 0x80 can be seen while the master is still attempting to read A2B_INTTYPE , so it may subsequently resolve, whereas A2B_INTTYPE 0xFD cannot resolve.

If a slave just reports an interrupt to the master, without any additional line failures after that. If after step 4, the host reads A2B_INTTYPE too fast, it reads A2B_INTTYPE = 0x80, resulting in the IRQ to be cleared. The master does not reassert the IRQ if the A2B_INTTYPE register is read before the register value is updated from a slave.

If the host reads the A2B_INTSRC register, then the A2B_INTTYPE register after seeing the IRQ (which is recommended), then the A2B_INTTYPE value is valid (unless there is a line error).

If a slave with no pending interrupt disconnects from the rest of the bus, the upstream slave generates the SRFERR in 32 consecutive superframes.

Error Management Register

When A^{2}B data slots are not received correctly (detected by a parity error or a data decode error on any bit in the slot), the last good sample received for that slot is repeated. The A2B_ERRMGMT register also controls the three ways in which bad data slots can be indicated across the I^{2}S/TDM interface.

When the A2B_ERRMGMT.ERRLSB bit is set, the LSB of each data slot is used to indicate whether the slot is received correctly or not. For example, in the master node with 24-bit upstream slot size, the 24th data bit sent over DTX0 or DTX1 is low if the data is valid and high if the data is not valid. Using this method changes the meaning of the LSB in the received I^{2}S/TDM data words.

When the A2B_ERRMGMT.ERRSIG bit is set, all bits below the LSB of each data slot are used to indicate whether the slot is received correctly or not. With a 24-bit slot size, the last 8 bits in each 32-cycle data slot are low if the data is valid and high if the data is not valid. If the A2B_ERRMGMT.ERRSIG bit is not set, the extra eight bits are always low. Using this method preserves the meaning of the LSB in the received I^{2}S/TDM data words, but the data word size must be smaller than the data channel size for this method to work. Data channel width is usually 32 bits, but it can be programmed to 16 bits.

When the A2B_ERRMGMT.ERRSLOT bit is set, the number of slots generated on the A^{2}B bus is incremented by 1. In the master node, the protocol engine normally writes A2B_UPSLOTS pieces of data to the frame buffer each superframe. In a slave node, the number of slots written is normally A2B_LDNSLOTS+ A2B_BCDNSLOTS. The additional data slot enabled by using this method is appended to the end of the configured A^{2}B traffic and contains a single bit of error information for each of the preceding data slots in that superframe. The MSB of the extra slot indicates an error occurred in data slot 0. The next bit indicates an error in data slot 1, and so on. For example, 0x80000000 indicates that there was an error in slot 0, while 0xffffff00 indicates that slots 0 through 23 all contained errors. If the A2B_I2SGCFG.TDMSS bit is set for a channel size of 16 bits, only the first 16 data channels can be reported. If the A2B_I2SGCFG.TDMSS bit is set for a channel size of 32 bits, up to 32 data channels can be reported for errors.

Bit Error Control Register

The A2B_BECNT register controls bit error counting, including interrupt thresholds of 2^n, where n ranges from 1 to 8. It selects which communication errors enter a counter and at what counter-threshold an interrupt request is generated. Using this feature, certain single-bit communication errors do not have to generate an interrupt unless they significantly accumulate over the time period when the A2B_BECNT register is last cleared.

Testing and Debugging

For testing and debugging, the transceiver allows generation of interrupts and bit errors using the raise A2B_RAISE and generate error A2B_GENERR registers.

Raise (A2B_RAISE) Register

The A2B_RAISE register allows the host to generate an interrupt in any node in the system via software. The register must be written over the A^{2}B bus, as writes to the register from the local I^{2}C port have no effect.

Generate Error (A2B_GENERR) Register

- *0x01 Generate Header Count Error* (A2B_GENERR.GENHCERR)

  1. When the master node generates the header count error:

     The master node changes the 2-bit CNT field in the SCF for one frame only. In the subsequent frame, it sends the correct CNT field.

     Because each slave node receives the SCF, all slaves detect the (A2B_INTPND0.HDCNTERR) error.

  2. When a slave node generates the header count error:

     The slave node changes the 2-bit CNT field in the SRF. Generally, the slave node passes the received SRF as-is from a downstream slave node. In this case (because the slave node is receiving the write to the A2B_GENERR command in the frame), it is already generating a response with its own SRF, but with the wrong CNT field, as the command indicated.

     Though the upstream slave nodes receive the SRF, the nodes do not check whether the CNT field is correct or not. The slave nodes only generate A2B_INTPND0.HDCNTERR on checks of the SCF. Therefore, when the slave node generates this error, only the master node detects it.

- *0x02 Generate Data Decoding Error* (A2B_GENERR.GENDDERR)

  Generating a data decoding error requires a Manchester coding violation to be applied to data slots, not to the SCF and SRF fields.

  1. When the master node generates the data decoding error:

     The master node induces a Manchester encoding error on the first downstream data slot (slot 0 only). It does not inject the error on any other data slots. Since nodes report the data decode error only on data slots that are consumed, only the slave nodes that consume slot 0 detect the error when the master generates it. When a slave node passes (without consuming) the data downstream, it sends the same bit stream that it receives and does not detect the error.

  2. When a slave node generates the data decoding error:

     The slave node induces a Manchester encoding error on the first upstream data slot it contributes, not on any passed data slot(s). If the slave contributes more than one upstream slot, it only induces the error on the first one. Slave nodes do not induce encoding errors on downstream data.

Since data decode errors are only reported on data slots which are consumed, only the upstream nodes that consume the first contributed upslot detect the error. If an upstream slave node or a master node does not consume the first contributed data slot, then it does not detect the error.

- ***0x04 Generate CRC Error*** (`A2B_GENERR.GENCRCERR`)

  1. When the master node generates the CRC error:

     The master node induces the error in the CRC field of the SCF for one frame only. Because each slave node receives the SCF, all slaves detect the error in the CRC.

  2. When a slave node generates the CRC error:

     Slave nodes induce the error in the CRC field of the SRF for one frame only. Since all upstream slave nodes receive the SRF and check the CRC, all of them detect this error when any downstream slave generates it. The slave nodes report the SRF CRC errors in the `A2B_INTPND0.SRFCRCERR` field, but these errors are not counted by the bit error counter. The master node detects the error as `A2B_INTPND0.CRCERR` and increments the bit error counter, if enabled.

- ***0x08 Generate Data Parity Error***, `A2B_GENERR.GENDPERR`

  1. When the master node generates the data decoding error:

     The master node induces the data parity error on the first downstream data slot (slot 0). It does not induce the error on other data slots. When the master node generates the data parity error, only the slave nodes that consume slot 0 detect it. Slave nodes that do not consume slot 0 do not detect it.

  2. When a slave node generates the data decoding error:

     A slave node induces the data parity error on only the first upstream data slot it contributes. It does not induce the error in the downstream portion of the superframe. When a slave node generates the error, all of the upstream nodes that consume the first contributed slot detect it. If an upstream slave node or a master node does not consume the first contributed data slot, it does not detect it.

- ***0x10 Generate Interrupt Frame CRC Error*** (`A2B_GENERR.GENICRCERR`)

  1. The master node cannot generate the interrupt frame CRC error.

  2. When a slave node generates the interrupt frame CRC error, only the master node is able to detect that error. Other upstream slave nodes do not check the CRC in the interrupt frame.

Unique ID

Each transceiver contains a 48-bit unique ID. Read the `A2B_CHIPID0` through `A2B_CHIPID5` registers to obtain the unique ID. If a read of the unique ID fails, an interrupt is generated (`A2B_INTTYPE` = 0xFC), which indicates that the unique ID cannot be recovered. If this occurs, return the transceiver to Analog Devices.

5 A2B System Debug

The following sections provide information on system diagnostics for fault isolation and correction. In addition to the A2B line fault detection, a loop back test mode is provided to validate the I2S/TDM connections in master and slave nodes.

Line Fault Diagnostics

This section discusses the A2B line fault diagnostics. It provides descriptions of different faults and programming instructions on how to react to line fault events in software. Line faults are detected during discovery but also can appear post discovery (delayed faults).

NOTE: The A2B_SWCTL.DIAGMODE bit must be set to 1 only when localizing the faults. Under all other conditions, the bit must be set to 0 to ensure proper operation of the device.

Diagnostics During Discovery

The ***Line Faults*** table shows the different types of line faults and the pins affected by the faults. All faults can be detected and localized during discovery of the bus. When a fault is detected during discovery, the switches that enable bias current to the next in line node are disconnected automatically.

Open wires are indicated by the A2B_INTTYPE register value of 0x0C.

Wires accidentally connected to the wrong port of the next node (port B instead of port A) also can create the same response or flag 0x0D to the A2B_INTTYPE register.

Reverse wire faults occur when the positive wire of one node accidentally connects to the negative input of the next in line node. This event is flagged with the A2B_INTTYPE register value 0x0D or indicated by a timeout while waiting for the discovery done response (A2B_INTTYPE = 0x18).

Timeout during discovery also occurs when an invalid value is programmed to the A2B_DISCVRY.DRESPCYC bit field, or if the next in line node has a physical defect that prevents the node from responding.

The faults that require specific software flow for detection and localization are shown with shading in the table.

NOTE: The A2B_SWCTL.ENSW bit is not cleared automatically when a line fault opens the bias switches; this has to be done in software. The A2B_SWCTL.ENSW bit in the master transceiver should be set to 0 in the event of a critical line fault to disconnect bus bias to any bus segments.

Table 5-1: Line Faults

Wires	Affected Pins	Detect	Localize	INTTYPE	Remarks
Partial Bus Operation May Continue for Nodes Upstream of the Fault					
Open	BP	Yes	Yes	0x0C	Open wires (BP and BN are the B-side positive and negative connector pins)
	BN				
	BN and BP				
Wrong Port	B to B' port	Yes	Yes	0x0C or 0x0D	B' is B-side of next in line node
Reverse Wires	BN to AP and BP to AN	Yes	Yes	0x0D	Wrong port or reverse wires (AP is the A side positive connector pin of the next in line node)
				No 0x18 timeout (no DSCDONE interrupt)	Reverse wires undetected by hardware diagnostics
Defective Node	NA	Yes	Yes	No 0x18 timeout (no DSCDONE interrupt)	Defective node or wrong software parameter value for A2B_DISCVRY.DRESPCYC
Short of Wires	BP with BN	Yes	Yes	0x0B	Wires shorted together
Critical Faults					
Short to Ground	BP	Yes	Yes	0x09	Positive wire shorted to ground
	BN		Yes	0x29	Software routine localizes fault
Short to V_{BAT}	BN	Yes	Yes	0x0A	Negative wire shorted to V_{BAT}
	BP		Yes	0x2A	Software routine localizes fault

CAUTION: The short to ground and short to V_{BAT} faults are critical faults for which the whole bus shuts off. Normal A^2B bus operation should always be discontinued, including removal of bus phantom power by the master node (independent of line fault location). Program the A2B_SWCTL.ENSW bit = 0 to the master transceiver until the fault is corrected.

For the following faults, partial A^2B bus operation can continue between the master and slave nodes which are upstream of the line fault location.

- Open
- Wrong port
- Reverse wires
- Defective node
- Wrong discovery parameter for next-in-line node

- Wires shorted together

Registers for Line Diagnostics

The following registers are used to diagnose line faults on the A^{2}B bus. Refer to the *Register Descriptions* section for details.

- The A2B_SWCTL register controls the bias voltage to be switched onto the B-side A^{2}B bus link for the next in line nodes. The register also provides special line fault sensing modes.
- The A2B_SWSTAT register provides line diagnostics status information.
- The A2B_INTSRC register contains information about the source of an active interrupt, which slave generated it, or whether the interrupt originates from the master. Line errors can be located with this register.
- The A2B_INTTYPE register stores information about the type of the current interrupt request. A read of this register clears the corresponding interrupt.

Open Wire Fault

The ***Open Wire Fault*** figure shows an open wire fault between 'SLAVE0' and 'SLAVE1'. Communication continues between the 'MASTER' and 'SLAVE0' when this fault occurs between 'SLAVE0' and 'SLAVE1'.

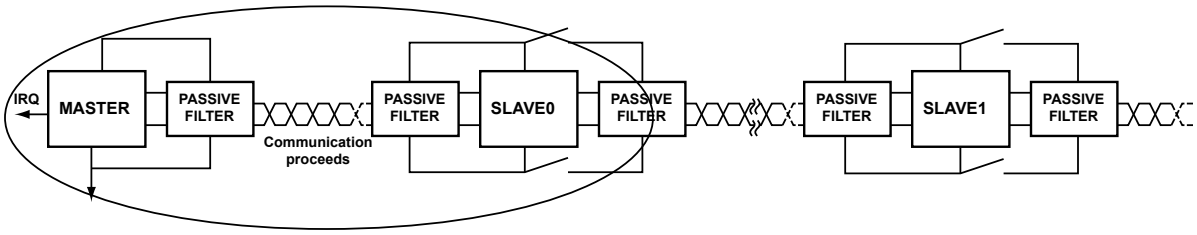

Figure 5-1: Open Wire Fault

Short of Wires Fault

The ***Short of Wires Fault*** figure shows a short of wires to each other line fault between 'SLAVE0' and 'SLAVE1'. Communication continues between the 'MASTER' and 'SLAVE0' when this fault occurs between 'SLAVE0' and 'SLAVE1'.

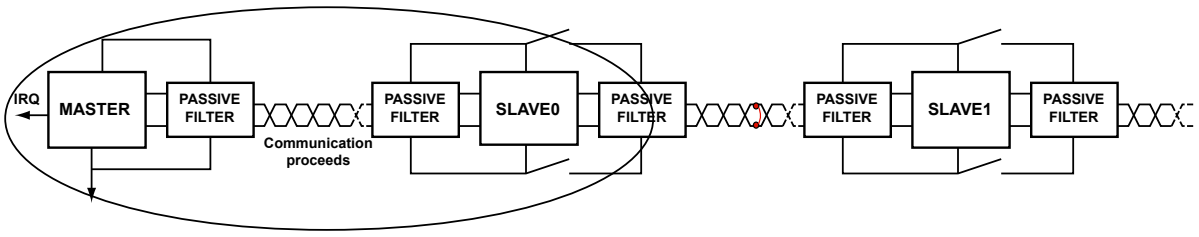

Figure 5-2: Short of Wires Fault

Short To GND BP

The ***Short to GND BP Fault*** figure shows the BP wire shorted to ground between 'SLAVE0' and 'SLAVE1'. All bus communication stops when this fault occurs between 'SLAVE0' and 'SLAVE1'.

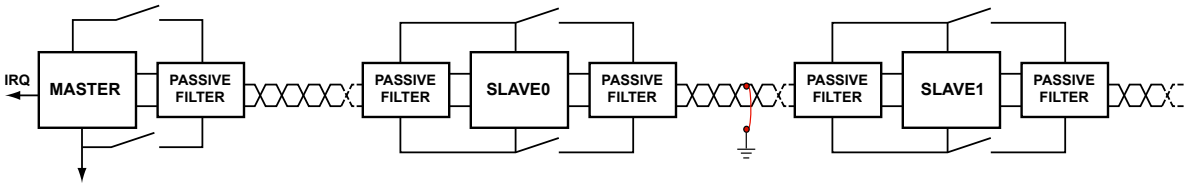

Figure 5-3: Short To GND BP

Short to V_{BAT} BN

The ***Short to V_{BAT} BN*** figure shows the BN wire shorted to V_{BAT} between 'SLAVE0' and 'SLAVE1'. All bus communication stops when this fault occurs between 'SLAVE0' and 'SLAVE1'.

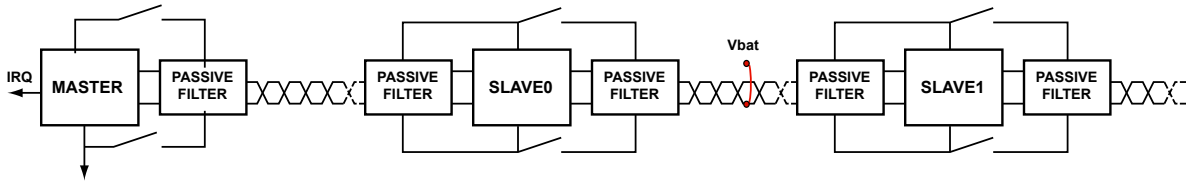

Figure 5-4: Short to V_{BAT} BN

Short To GND BN

The ***Short to GND BN*** figure shows the BN wire shorted to ground between 'SLAVE0' and 'SLAVE1'. Bus communication can continue without an immediate fault when this fault occurs between 'SLAVE0' and 'SLAVE1'.

NOTE: This line fault is a special diagnostic case because it propagates to earlier nodes as the FET switches have reverse diodes. During discovery or rediscovery of the bus, this fault is identified as not localized with specific `A2B_INTTYPE` code (0x29). In order to localize the fault, set the `A2B_SWCTL.DIAGMODE` bit = 1. See the Diagnostics Software Flow section and the Localizing Concealed Faults table for fault diagnostics software flow.

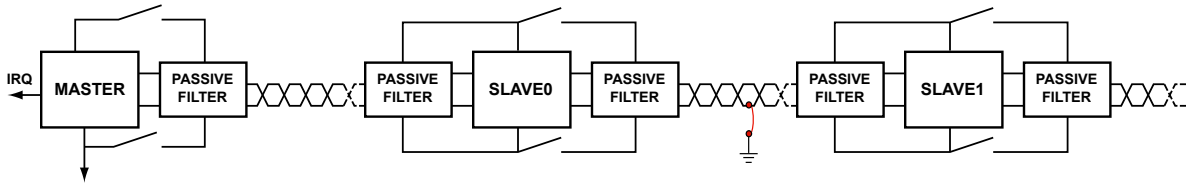

Figure 5-5: Short to GND BN

Short to V_{BAT} BP

The ***Short to V_{BAT} BP*** figure shows the BP wire shorted to V_{BAT} between 'SLAVE0' and 'SLAVE1'. Bus communication can continue without an immediate fault when the short to V_{BAT} BP fault occurs between 'SLAVE0' and 'SLAVE1'.

NOTE: This line fault is a special diagnostic case because it propagates to earlier nodes as the FET switches have reverse diodes. During discovery or rediscovery of the bus, this fault is identified as not localized with specific A2B_INTTYPE code (0x2A). In order to localize the fault, set the A2B_SWCTL.DIAGMODE bit = 1. See the Diagnostics Software Flow section and the Localizing Concealed Faults figure for fore information about the fault diagnostics software flow.

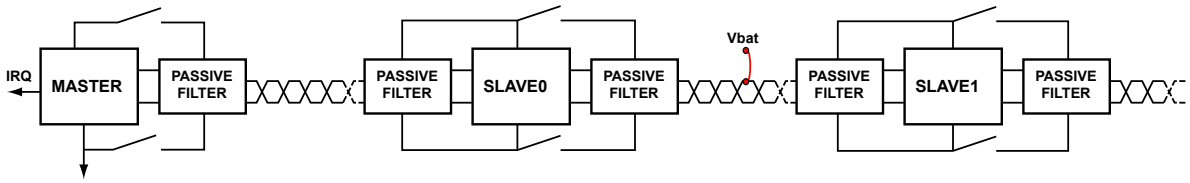

Figure 5-6: Short to V_{BAT} BP

Line Diagnostics After Discovery

Full line diagnostics are only performed during discovery. However, certain interrupts (if enabled) after discovery can indicate line faults during operation. Rediscovery detects the cause and location of any possible faults.

After discovery, any of the following interrupt types (A2B_INTTYPE) indicate that there is a line fault:

- 0x0A (10: PWRERR)
- 0x0F (15: PWRERR)
- 0x2A (42: PWRERR)
- 0x80 (128: interrupt messaging error)

When a slave node detects the SRF miss error (SRFERR) in 32 consecutive frames, the node assumes a downstream bus drop and sets its last node bit (A2B_NODE.LAST = 1) to become the last node in the system. A bus drop condition resulting from a line fault occurring after discovery can be detected in a last node (A2B_NODE.LAST = 1) that has the SRFERR latched.

Excessive accumulation of bit errors can happen if there is a slot configuration mismatch between nodes. This can happen when A2B_BP shorts to a noisy V_{BAT} or BN shorts to a noisy GND. The bus can operate under these conditions but is more susceptible to impairments (for example, electromagnetic interference).

Use the A2B_BECNT register to count accumulated errors as follows.

- Set the A2B_BECCTL register to 0xE4 (interrupt after 256 CRC errors). Acceptable audio noise and robustness is subjective and needs to be determined in vehicle tests. Adjust the threshold accordingly.

- Periodically write 0 to the A2B_BECNT register (once every second) to reset the error counter. Acceptable audio noise and robustness is subjective and needs to be determined in vehicle tests. Adjust time for the A2B_BECNT register accordingly.
- The bit error counter overflow (0x04: BECOVF) interrupt indicates bus issues.

Diagnostics Software Flow

Use the following software flow and the ***Diagnostics Software Flow*** figure for node discovery with diagnostics.

1. Set the A2B_SWCTL register = 0x00 for diagnostics mode 0.
2. Enable the power error interrupts and the A2B_INTPND2.DSCDONE interrupt in the master node. Set the A2B_SWCTL register = 0x01 to enable the power switch.
3. Wait for interrupt to occur. If the A2B_INTTYPE register = 0x18 for A2B_INTPND2.DSCDONE (indicating a successful node discovery), proceed to step 7.
4. If the A2B_INTTYPE register= 0x29 or 0x2A, configure the A2B_SWCTL.ENSW bit = 0 in the master and wait 50-100ms. Proceed to rediscovery with the A2B_SWCTL.DIAGMODE bit = 1 in the Localizing Concealed Faults section (step 8).

   ADDITIONAL INFORMATION: If the A2B_INTTYPE register is any other A2B_INTPND0.PWRERR type or if the discovery operation times out, proceed to step 5.
5. If the A2B_INTTYPE register= 0x0B, 0x0C, or 0x0D, the A2B_INTSRC register can be read to determine the location. If the operation times out, then by process of elimination the bus wires to the node being discovered are most likely reversed. Proceed to step 6.
6. If the A2B_INTTYPE register = 0x09 or 0x0A, disable the entire bus by setting the A2B_SWCTL register = 0x00 in the master node after the A2B_INTSRCand A2B_INTTYPE register values have been communicated to the host.

   ADDITIONAL INFORMATION: Once any other localized fault has been detected, halt the discovery process. Retry the discovery process periodically by software to determine if the fault is cleared. There is no automatic retry mechanism within the transceiver.
7. If this is not the last node, reprogram the A2B_SWCTL.MODE bits = 2. This setting ignores fluctuation on VIN due to downstream current draw and prevents incorrect localization on errors that occur on nodes that are located further downstream. Program the downstream node register settings and repeat step 1 on next node.

   ADDITIONAL INFORMATION: Continue this cycle until all nodes are discovered. Once all nodes are discovered, configure the A2B_SWCTL.MODE bits = 0 to all nodes while keeping the A2B_SWCTL.ENSW bit = 1. Full A^{2}B bus discovery is complete now.

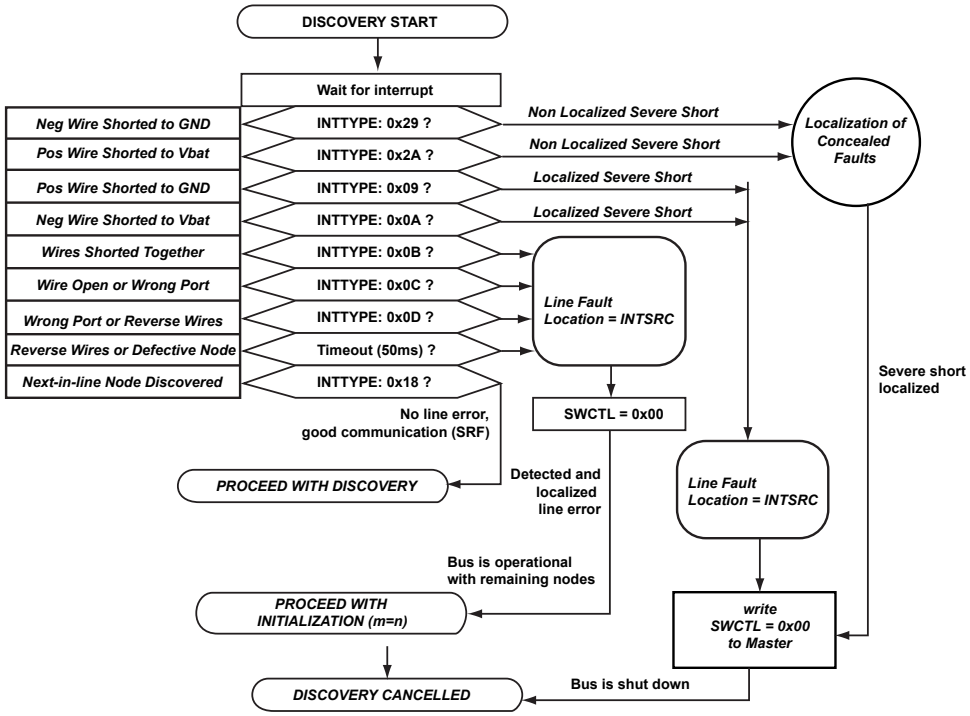

Figure 5-7: Diagnostics Software Flow

8. If there is a timeout while discovering a slave node, stop the discovery process by setting the `A2B_CONTROL.ENDDSC` bit.

Localizing Concealed Faults

This section describes the procedure to determine the location of the potentially concealed faults to either V_{BAT} or GND.

While the bus should not be operated long term in the presence of such a fault, run the following procedure on a short term basis to establish the location of a concealed fault before disabling the bus at the master node. This process is shown in the ***Localization of Concealed Faults*** figure.

1. Following from step 4 in Diagnostics Software Flow, where the `A2B_SWCTL.ENSW` bit = 0 in the master, set variables to keep track of the current node position and the last known good node. Also set a variable, for example PriorFault = 0. This keeps track of whether or not a fault is reported in a prior node discovery. After waiting at least 100 ms to allow for the electrical steady state of the bus to settle, proceed to step 2.
2. In the current node, set the `A2B_INTMSK0` register = 0x10 and the `A2B_SWCTL` register = 0x09. This sets the `A2B_SWCTL.ENSW` bits initiating rediscovery in diagnostic mode. Define two variables for keeping track of whether or not a fault and/or discovery is completed in the current discovery attempt; for example, `Disc` and `Fault`. Clear both variables in this step.

ADDITIONAL INFORMATION: Wait for an interrupt from this operation, allow 100 ms for a timeout. This timeout provides sufficient time for bus diagnostics and, possibly, full discovery to complete. This process can take longer than usual when A2B_SWCTL.ENSW = 1 in the presence of a fault.

3. If no interrupt is received before the timeout expires, the fault is located immediately downstream of the current node. Set GoodNode to the current node. Proceed to step 9.
4. If the A2B_INTTYPE register = 0x29 or 0x2A, then an error occurred somewhere downstream of the current node. This means a fault is detected so set Fault = 1. If DISC = 0, return to step 3 with a 100 ms timeout waiting to see if discovery completes. If DISC = 1 and the discovery process was reported previously as complete (A2B_INTTYPE register = 0x18), proceed to step 6.
5. If the A2B_INTTYPE register = 0x18, the downstream node was successfully discovered and communication has been established. In diagnostic mode, this can occur even in the presence of a detected fault of A2B_INTTYPE = 0x29 or 0x2A. These faults can occur when the physical fault exists on only one of the wires between the two nodes. Proceed to step 6.
6. Check the value of the A2B_INTSTAT register for other pending interrupts. If the A2B_INTSTAT register is non-zero, the fault and discovery completion both occurred faster than the interrupt service routine response. In this case, the 0x18 DSCDONE interrupt is a higher priority. Set DISC = 1 and return to step 3. If the A2B_INTSTAT register = 0, there are no more pending interrupts. Proceed to step 7.
7. In order to reach this step, discovery must have been completed successfully. If a fault was also detected, then Fault = 1, and it is necessary to continue the bus discovery to localize the fault. Set GoodNode = Node, PriorFault = Fault, and Node = n. Increment n in preparation for the discovery of the next node and return to step 2. If Fault = 0 (Fault was not detected), proceed to step 8.
8. To reach this step, discovery must have been completed and no fault was detected. This can occur for one of two reasons. Either the current node is too far upstream of the fault to detect it yet, or the node is already downstream of the fault where fault is no longer present. If PriorFault = 1, then it is the latter case, so proceed to step 9. If PriorFault = 0, then the fault has yet to be detected. In this case, continue bus discovery to localize the fault. Set GoodNode = Node, PriorFault = Fault, and Node = n. Increment n in preparation for the discovery of the next node and return to step 2.
9. Report the fault location as being immediately downstream of the last recorded GoodNode. The location of the error is after the current node unless in step 8 it was detected that the line fault is before the current node. In this case, the last GoodNode is one node up. Localization of the concealed fault is complete.

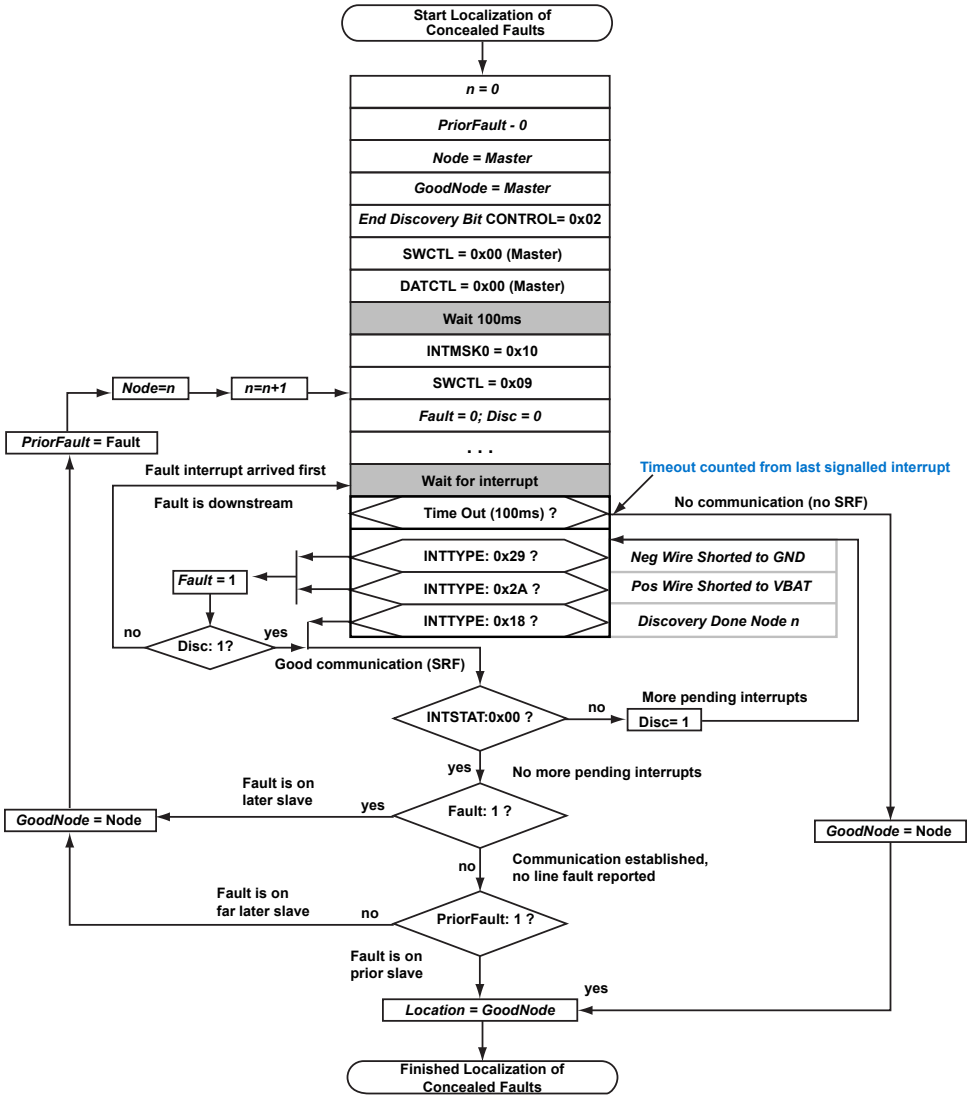

Figure 5-8: Localization of Concealed Faults

Bus Drop Detection

The ***Diagnostics Software Flow*** figure describes the flow of bus drop detection in the A^{2}B system.

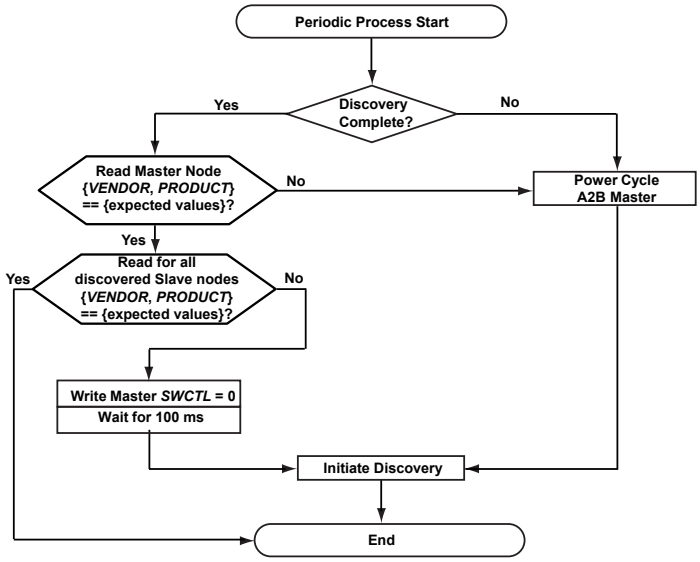

Figure 5-9: Diagnostics Software Flow

I²S Loopback

I²S loopback occurs inside the transceiver. Data driven to the DTX0 pad is sampled as A²B receive data instead of the data on the DRX0 pin. Data driven to the DTX1 pad is sampled as A²B receive data instead of the data on the DRX1 pin.

The `A2B_I2STEST.BUSLOOPBK` bit enables loopback from the DTX0 pins to the serial RX blocks. The values of the `A2B_I2STEST.SELRX1`, `A2B_I2STEST.RX2LOOPBK`, and `A2B_I2STEST.LOOPBK2TX` bits are ignored if this bit is set. If the `A2B_I2STEST.PATTRN2TX` bit is set, a fixed pattern (0xB38F0E32) is driven on the DTX0 and DTX1 pins instead of transmit data from the A²B bus.

If I²S Loopback mode is enabled, program the value of the `A2B_I2SCFG.RX0EN` bit to match the value of the `A2B_I2SCFG.TX0EN` bit, and the value of the `A2B_I2SCFG.RX1EN` bit to match the value of the `A2B_I2SCFG.TX1EN` bit.

The number of data slots received and transmitted on the A²B bus by each node is controlled by a number of registers.

If the `A2B_SLOTFMT.UPSIZE` and `A2B_SLOTFMT.DNSIZE` bit field values are different, looped back data, which changes direction on the bus, is either truncated to a smaller bit width or zero-filled to a larger bit width.

When this mode is enabled, the program is responsible for ensuring that the data received from the A²B bus and looped back through the serial blocks can be transmitted on the A²B bus.

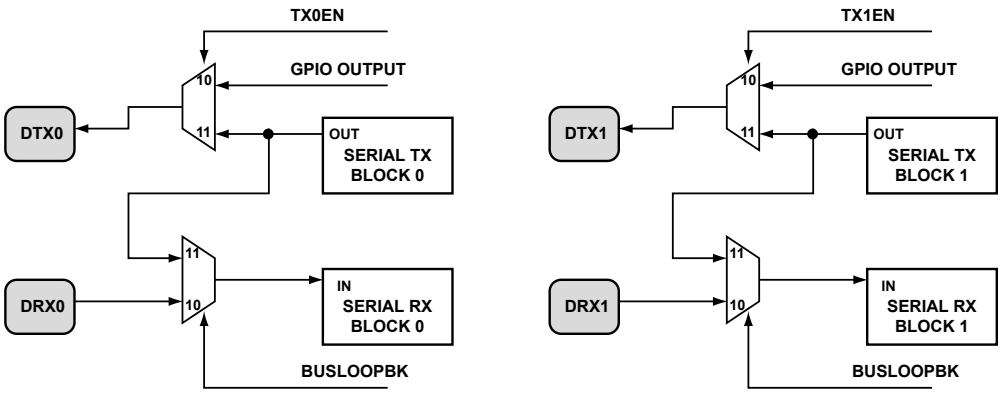

Figure 5-10: Serial TX Block to Serial RX Block

I²S TDM Test Mode (I²S Loopback)

Pattern generation and loopback test modes are provided for easy validation of I²S TDM connectivity in master and slave nodes. The transmit pattern generator uses the default bit pattern `1011_0011_1000_1111__0000_1110_0011_0010` on all channels, where `1011` is the most significant nibble and `0010` is the least significant nibble.

Use the following procedure for I²S TDM testing.

1. For master to host link verification, set the `A2B_I2STEST.PATTRN2TX` bit in the master and verify that the TX interface with the default bit pattern matches the expected timing (possibly using a scope, logic analyzer, or other device).
2. For host to master link verification. Set the `A2B_I2STEST.RX2LOOPBK` and `A2B_I2STEST.LOOPBK2TX` bits in the master, wait one cycle, and verify that the DTX data received at the host matches the sent DRX data from the previous frame.

   ADDITIONAL INFORMATION: The RX to TX loopback does not working correctly when the master node is also receiving TX data from the bus. The `A2B_DATCTL` register must be 0x00 while looping back from RX to TX.
3. For slave to peripheral link verification, if a slave is connected to a DAC (for example, to send to a speaker), set the `A2B_I2STEST.PATTRN2TX` bit in the slave and verify the expected DTX timing.
4. For peripheral to slave link verification. If a slave node has a peripheral that provides input signals over the I²S TDM interface, set the `A2B_I2STEST.RX2LOOPBK` and `A2B_I2STEST.LOOPBK2TX` bits. Verify that the DTX interface matches the DRX interface with a one frame delay. Alternatively (without using the `A2B_I2STEST` register) check the RX data at the earlier verified master I²S/TDM DTX interface.
5. System verification with external loopback. Connect the DTX0/DTX1 pins with the DRX0/DRX1 pins in a slave node to generate a digital loopback. The default bit pattern can be verified at the master DTX pins when the `A2B_I2STEST.PATTRN2TX` bit is set at the slave node.

ADDITIONAL INFORMATION: If the A2B_I2STEST.RX2LOOPBK bit is cleared while the A2B_I2STEST.LOOPBK2TX bit is set, then the last received frame is repeated on the TX pins. This behavior persists until the A2B_I2STEST.RX2LOOPBK bit is set or the A2B_I2STEST.LOOPBK2TX bit is cleared. If the A2B_I2STEST.LOOPBK2TX bit is enabled after reset, the default pattern is generated until the A2B_I2STEST.RX2LOOPBK bit is set.

ADDITIONAL INFORMATION: The ***Frame Buffer*** figure shows the TX frame buffer that is used for loopback tests.

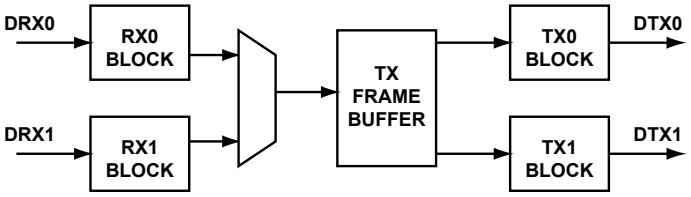

Figure 5-11: Frame Buffer

6 Register Summary

The following table provides the map of the AD2420(W)/AD2426(W)/AD2427(W)/AD2428(W)/AD2429(W) registers and bits.

Reg. Addr.	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset	RW
0x00	CHIP	Reserved	CHIPADR							0x50	R/W
0x01	NODEADR	BRCST	Reserved	PERI	Reserved	NODE				0x00	R/W
0x02	VENDOR	VENDOR								0xAD	R/NW
0x03	PRODUCT	PRODUCT								0x25	R/NW
0x04	VERSION	VERSION								0x00	R/NW
0x05	CAPABILITY	Reserved							I2CAVAIL	0x01	R/NW
0x09	SWCTL	Reserved	DISNXT	MODE		DIAGMODE	Reserved		ENSW	0x00	R/W
0x0A	BCDNSLOTS	Reserved		BCDNSLOTS						0x00	R/W
0x0B	LDNSLOTS	DNMASKEN	Reserved	LDNSLOTS						0x00	R/W
0x0C	LUPSLOTS	Reserved		LUPSLOTS						0x00	R/W
0x0D	DNSLOTS	Reserved		DNSLOTS						0x00	R/W
0x0E	UPSLOTS	Reserved		UPSLOTS						0x00	R/W
0x0F	RESPCYCS	RESPCYCS								0x40	R/W
0x10	SLOTFMT	UPFMT	UPSIZE			DNFMT	DNSIZE			0x00	R/W
0x11	DATCTL	STANDBY	Reserved	ENDSNIFF	Reserved			UPS	DNS	0x00	R/W
0x12	CONTROL	MSTR	Reserved		XCVRBINV	SWBYP	SOFTRST	ENDDSC	NEWSTRCT	0x00	R/W
0x13	DISCVRY	DRESPCYC								0x00	R/W
0x14	SWSTAT	FAULT_ NLOC	FAULT_CODE			Reserved		FAULT	FIN	0x00	R/NW
0x15	INTSTAT	Reserved							IRQ	0x00	R/NW
0x16	INTSRC	MSTINT	SLVINT	Reserved		INODE				0x00	R/NW
0x17	INTTYPE	TYPE								0x00	R/NW
0x18	INTPND0	SRFCRCERR	SRFERR	BECOVF	PWRERR	DPERR	CRCERR	DDERR	HDCNTERR	0x00	R/W
0x19	INTPND1	IO7PND	IO6PND	IO5PND	IO4PND	IO3PND	IO2PND	IO1PND	IO0PND	0x00	R/W
0x1A	INTPND2	Reserved				SLVIRQ	ICRCERR	I2CERR	DSCDONE	0x00	R/W
0x1B	INTMSK0	SRFCRCEIEN	SRFEIEN	BECIEN	PWREIEN	DPEIEN	CRCEIEN	DDEIEN	HCEIEN	0x00	R/W
0x1C	INTMSK1	IO7IRQEN	IO6IRQEN	IO5IRQEN	IO4IRQEN	IO3IRQEN	IO2IRQEN	IO1IRQEN	IO0IRQEN	0x00	R/W
0x1D	INTMSK2	Reserved				SLVIRQEN	ICRCEIEN	I2CEIEN	DSCDIEN	0x00	R/W
0x1E	BECCTL	THRESHLD			ENICRC	ENDP	ENCRC	ENDD	ENHDCNT	0x00	R/W

Reg. Addr.	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset	RW
0x1F	BECNT	BECNT								0x00	R/W
0x20	TESTMODE	Reserved		RXDPTH		Reserved	PRBSN2N	PRBSDN	PRBSUP	0x00	R/W
0x21	ERRCNT0	ERRCNT0[7:0]								0x00	R/NW
0x22	ERRCNT1	ERRCNT1[15:8]								0x00	R/NW
0x23	ERRCNT2	ERRCNT2[23:16]								0x00	R/NW
0x24	ERRCNT3	ERRCNT3[31:24]								0x00	R/NW
0x29	NODE	LAST	NLAST	DISCVD	Reserved	NUMBER				0x00	R/NW
0x2B	DISCSTAT	DSCACT	Reserved			DNODE				0x00	R/NW
0x2E	TXACTL	TXAOVREN	Reserved					TXALEVEL		0x00	R/W
0x30	TXBCTL	TXBOVREN	Reserved					TXBLEVEL		0x00	R/W
0x3E	LINTTYPE	LINTTYPE								0x00	R/NW
0x3F	I2CCFG	Reserved					FRAMERATE	EACK	DATARATE	0x00	R/W
0x40	PLLCTL	SSMODE		Reserved		SSDEPTH	Reserved	SSFREQ		0x00	R/W
0x41	I2SGCFG	INV	EARLY	ALT	TDMSS	RXONDTX1	TDMMODE			0x00	R/W
0x42	I2SCFG	RXBCLKINV	RX2PINTL	RX1EN	RX0EN	TXBCLKINV	TX2PINTL	TX1EN	TX0EN	0x00	R/W
0x43	I2SRATE	SHARE	REDUCE	BCLKRATE			I2SRATE			0x00	R/W
0x44	I2STXOFFSET	TSBEFORE	TSAFTER	TXOFFSET						0x00	R/W
0x45	I2SRXOFFSET	Reserved		RXOFFSET						0x00	R/W
0x46	SYNCOFFSET	SYNCOFFSET								0x00	R/W
0x47	PDMCTL	Reserved	PDMRATE		HPFEN	PDM1SLOTS	PDM1EN	PDM0SLOTS	PDM0EN	0x00	R/W
0x48	ERRMGMT	Reserved					ERRSLOT	ERRSIG	ERRLSB	0x00	R/W
0x4A	GPIODAT	IO7DAT	IO6DAT	IO5DAT	IO4DAT	IO3DAT	IO2DAT	IO1DAT	IO0DAT	0x00	R/W
0x4B	GPIODATSET	IO7DSET	IO6DSET	IO5DSET	IO4DSET	IO3DSET	IO2DSET	IO1DSET	IO0DSET	0x00	R/W
0x4C	GPIODATCLR	IO7DCLR	IO6DCLR	IO5DCLR	IO4DCLR	IO3DCLR	IO2DCLR	IO1DCLR	IO0DCLR	0x00	R/W
0x4D	GPIOOEN	IO7OEN	IO6OEN	IO5OEN	IO4OEN	IO3OEN	IO2OEN	IO1OEN	IO0OEN	0x00	R/W
0x4E	GPIOIEN	IO7IEN	IO6IEN	IO5IEN	IO4IEN	IO3IEN	IO2IEN	IO1IEN	IO0IEN	0x00	R/W
0x4F	GPIOIN	IO7IN	IO6IN	IO5IN	IO4IN	IO3IN	IO2IN	IO1IN	IO0IN	0x00	R/NW
0x50	PINTEN	IO7IE	IO6IE	IO5IE	IO4IE	IO3IE	IO2IE	IO1IE	IO0IE	0x00	R/W
0x51	PINTINV	IO7INV	IO6INV	IO5INV	IO4INV	IO3INV	IO2INV	IO1INV	IO0INV	0x00	R/W
0x52	PINCFG	Reserved		IRQTS	IRQINV	Reserved			DRVSTR	0x01	R/W
0x53	I2STEST	Reserved			BUSLOOPBK	SELRX1	RX2LOOPBK	LOOPBK2TX	PATTRN2TX	0x00	R/W
0x54	RAISE	RTYPE								0x00	R/W
0x55	GENERR	Reserved			GENICRCERR	GENDPERR	GENCRCERR	GENDDERR	GENHCERR	0x00	R/W
0x56	I2SRRATE	RBUS	Reserved	RRDIV						0x00	R/W
0x57	I2SRRCTL	Reserved		STRBDIR	ENSTRB	Reserved		ENXBIT	ENVLSB	0x00	R/W
0x58	I2SRRSOFFS	Reserved						RRSOFFSET		0x00	R/W
0x59	CLK1CFG	CLK1EN	CLK1INV	CLK1PDIV	Reserved	CLK1DIV				0x00	R/W
0x5A	CLK2CFG	CLK2EN	CLK2INV	CLK2PDIV	Reserved	CLK2DIV				0x00	R/W
0x5B	BMMCFG	Reserved					BMMNDSC	BMMRXEN	BMMEN	0x00	R/W
0x5C	SUSCFG	Reserved		SUSDIS	SUSOE	Reserved	SUSSEL			0x00	R/W
0x5D	PDMCTL2	Reserved		PDMINVCLK	PDMALTCLK	PDM1FFRST	PDM0FFRST	PDMDEST		0x00	R/W

Reg. Addr.	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset	RW
0x60	UPMASK0	RXUPSLOT07	RXUPSLOT06	RXUPSLOT05	RXUPSLOT04	RXUPSLOT03	RXUPSLOT02	RXUPSLOT01	RXUPSLOT00	0x00	R/W
0x61	UPMASK1	RXUPSLOT15	RXUPSLOT14	RXUPSLOT13	RXUPSLOT12	RXUPSLOT11	RXUPSLOT10	RXUPSLOT09	RXUPSLOT08	0x00	R/W
0x62	UPMASK2	RXUPSLOT23	RXUPSLOT22	RXUPSLOT21	RXUPSLOT20	RXUPSLOT19	RXUPSLOT18	RXUPSLOT17	RXUPSLOT16	0x00	R/W
0x63	UPMASK3	RXUPSLOT31	RXUPSLOT30	RXUPSLOT29	RXUPSLOT28	RXUPSLOT27	RXUPSLOT26	RXUPSLOT25	RXUPSLOT24	0x00	R/W
0x64	UPOFFSET	Reserved			UPOFFSET					0x00	R/W
0x65	DNMASK0	RXDNSLOT07	RXDNSLOT06	RXDNSLOT05	RXDNSLOT04	RXDNSLOT03	RXDNSLOT02	RXDNSLOT01	RXDNSLOT00	0x00	R/W
0x66	DNMASK1	RXDNSLOT15	RXDNSLOT14	RXDNSLOT13	RXDNSLOT12	RXDNSLOT11	RXDNSLOT10	RXDNSLOT09	RXDNSLOT08	0x00	R/W
0x67	DNMASK2	RXDNSLOT23	RXDNSLOT22	RXDNSLOT21	RXDNSLOT20	RXDNSLOT19	RXDNSLOT18	RXDNSLOT17	RXDNSLOT16	0x00	R/W
0x68	DNMASK3	RXDNSLOT31	RXDNSLOT30	RXDNSLOT29	RXDNSLOT28	RXDNSLOT27	RXDNSLOT26	RXDNSLOT25	RXDNSLOT24	0x00	R/W
0x69	DNOFFSET	Reserved			DNOFFSET					0x00	R/W
0x6A	CHIPID0	CHIPID[7:0]								0xXX	R/NW
0x6B	CHIPID1	CHIPID[15:8]								0xXX	R/NW
0x6C	CHIPID2	CHIPID[23:16]								0xXX	R/NW
0x6D	CHIPID3	CHIPID[31:24]								0xXX	R/NW
0x6E	CHIPID4	CHIPID[39:32]								0xXX	R/NW
0x6F	CHIPID5	CHIPID[47:40]								0xXX	R/NW
0x80	GPIODEN	IOD7EN	IOD6EN	IOD5EN	IOD4EN	IOD3EN	IOD2EN	IOD1EN	IOD0EN	0x00	R/W
0x81	GPIOD0MSK	IOD0MSK								0x00	R/W
0x82	GPIOD1MSK	IOD1MSK								0x00	R/W
0x83	GPIOD2MSK	IOD2MSK								0x00	R/W
0x84	GPIOD3MSK	IOD3MSK								0x00	R/W
0x85	GPIOD4MSK	IOD4MSK								0x00	R/W
0x86	GPIOD5MSK	IOD5MSK								0x00	R/W
0x87	GPIOD6MSK	IOD6MSK								0x00	R/W
0x88	GPIOD7MSK	IOD7MSK								0x00	R/W
0x89	GPIODDAT	IOD7DAT	IOD6DAT	IOD5DAT	IOD4DAT	IOD3DAT	IOD2DAT	IOD1DAT	IOD0DAT	0x00	R/W
0x8A	GPIODINV	IOD7INV	IOD6INV	IOD5INV	IOD4INV	IOD3INV	IOD2INV	IOD1INV	IOD0INV	0x00	R/W
0x90	MBOX0CTL	Reserved		MB0LEN		MB0FIEN	MB0EIEN	MB0DIR	MB0EN	0x00	R/W
0x91	MBOX0STAT	Reserved		MB0EIRQ	MB0FIRQ	Reserved		MB0EMPTY	MB0FULL	0x00	R/W
0x92	MBOX0B0	MBOX0[7:0]								0x00	R/W
0x93	MBOX0B1	MBOX0[15:8]								0x00	R/W
0x94	MBOX0B2	MBOX0[23:16]								0x00	R/W
0x95	MBOX0B3	MBOX0[31:24]								0x00	R/W
0x96	MBOX1CTL	Reserved		MB1LEN		MB1FIEN	MB1EIEN	MB1DIR	MB1EN	0x00	R/W
0x97	MBOX1STAT	Reserved		MB1EIRQ	MB1FIRQ	Reserved		MB1EMPTY	MB1FULL	0x00	R/W
0x98	MBOX1B0	MBOX1[7:0]								0x00	R/W
0x99	MBOX1B1	MBOX1[15:8]								0x00	R/W
0x9A	MBOX1B2	MBOX1[23:16]								0x00	R/W
0x9B	MBOX1B3	MBOX1[31:24]								0x00	R/W

7 AD2428 A2B Register Descriptions

The transceiver (A2B) contains the following registers.

Table 7-1: AD2428 A2B Register List

Name	Description
A2B_CHIP	I2C Chip Address Register (Slave Only)
A2B_NODEADR	Node Address Register (Master Only)
A2B_VENDOR	Vendor ID Register
A2B_PRODUCT	Product ID Register
A2B_VERSION	Version ID Register
A2B_CAPABILITY	Capability ID Register
A2B_SWCTL	Switch Control Register
A2B_BCDNSLOTS	Broadcast Downstream Slots Register (Slave Only)
A2B_LDNSLOTS	Local Downstream Slots Register (Slave Only)
A2B_LUPSLOTS	Local Upstream Slots Register (Slave Only)
A2B_DNSLOTS	Downstream Slots Register
A2B_UPSLOTS	Upstream Slots Register
A2B_RESPCYCS	Response Cycles Register
A2B_SLOTFMT	Slot Format Register (Master Only, Auto-Broadcast)
A2B_DATCTL	Data Control Register (Master Only, Auto-Broadcast)
A2B_CONTROL	Control Register
A2B_DISCVRY	Discovery Register (Master Only)
A2B_SWSTAT	Switch Status Register
A2B_INTSTAT	Interrupt Status Register
A2B_INTSRC	Interrupt Source Register (Master Only)
A2B_INTTYPE	Interrupt Type Register (Master Only)
A2B_INTPND0	Interrupt Pending 0 Register

Table 7-1: AD2428 A2B Register List (Continued)

Name	Description
A2B_INTPND1	Interrupt Pending 1 Register
A2B_INTPND2	Interrupt Pending 2 Register (Master Only)
A2B_INTMSK0	Interrupt Mask 0 Register
A2B_INTMSK1	Interrupt Mask 1 Register
A2B_INTMSK2	Interrupt Mask 2 Register (Master Only)
A2B_BECCTL	Bit Error Count Control Register
A2B_BECNT	Bit Error Count Register
A2B_TESTMODE	Testmode Register
A2B_ERRCNT0	PRBS Error Count Byte 0 Register
A2B_ERRCNT1	PRBS Error Count Byte 1 Register
A2B_ERRCNT2	PRBS Error Count Byte 2 Register
A2B_ERRCNT3	PRBS Error Count Byte 3 Register
A2B_NODE	Node Register
A2B_DISCSTAT	Discovery Status Register (Master Only)
A2B_TXACTL	LVDSA TX Control Register
A2B_TXBCTL	LVDSB TX Control Register
A2B_LINTTYPE	Local Interrupt Type (Slave Only)
A2B_I2CCFG	I2C Configuration Register
A2B_PLLCTL	PLL Control Register
A2B_I2SGCFG	I2S Global Configuration Register
A2B_I2SCFG	I2S Configuration Register
A2B_I2SRATE	I2S Rate Register (Slave Only)
A2B_I2STXOFFSET	I2S Transmit Data Offset Register (Master Only)
A2B_I2SRXOFFSET	I2S Receive Data Offset Register (Master Only)
A2B_SYNCOFFSET	SYNC Offset Register (Slave Only)
A2B_PDMCTL	PDM Control Register
A2B_ERRMGMT	Error Management Register
A2B_GPIODAT	GPIO Output Data Register
A2B_GPIODATSET	GPIO Output Data Set Register
A2B_GPIODATCLR	GPIO Output Data Clear Register
A2B_GPIOOEN	GPIO Output Enable Register

Table 7-1: AD2428 A2B Register List (Continued)

Name	Description
A2B_GPIOIEN	GPIO Input Enable Register
A2B_GPIOIN	GPIO Input Value Register
A2B_PINTEN	Pin Interrupt Enable Register
A2B_PINTINV	Pin Interrupt Invert Register
A2B_PINCFG	Pin Configuration Register
A2B_I2STEST	I2S Test Register
A2B_RAISE	Raise Interrupt Register
A2B_GENERR	Generate Bus Error
A2B_I2SRRATE	I2S Reduced Rate Register (Master Only, Auto-Broadcast)
A2B_I2SRRCTL	I2S Reduced Rate Control Register
A2B_I2SRRSOFFS	I2S Reduced Rate SYNC Offset Register (Slave Only)
A2B_CLK1CFG	CLKOUT1 Configuration Register
A2B_CLK2CFG	CLKOUT2 Configuration Register
A2B_BMMCFG	Bus Monitor Mode Configuration Register
A2B_SUSCFG	Sustain Configuration Register (Slave Only)
A2B_PDMCTL2	PDM Control 2 Register
A2B_UPMASK0	Upstream Data RX Mask 0 Register (Slave Only)
A2B_UPMASK1	Upstream Data RX Mask 1 Register (Slave Only)
A2B_UPMASK2	Upstream Data RX Mask 2 Register (Slave Only)
A2B_UPMASK3	Upstream Data RX Mask 3 Register (Slave Only)
A2B_UPOFFSET	Local Upstream Channel Offset Register (Slave Only)
A2B_DNMASK0	Downstream Data RX Mask 0 Register (Slave Only)
A2B_DNMASK1	Downstream Data RX Mask 1 Register (Slave Only)
A2B_DNMASK2	Downstream Data RX Mask 2 Register (Slave Only)
A2B_DNMASK3	Downstream Data RX Mask 3 Register (Slave Only)
A2B_DNOFFSET	Local Downstream Channel Offset Register (Slave Only)
A2B_CHIPID0	Chip ID Register 0
A2B_CHIPID1	Chip ID Register 1
A2B_CHIPID2	Chip ID Register 2
A2B_CHIPID3	Chip ID Register 3
A2B_CHIPID4	Chip ID Register 4

Table 7-1: AD2428 A2B Register List (Continued)

Name	Description
A2B_CHIPID5	Chip ID Register 5
A2B_GPIODEN	GPIO Over Distance Enable Register
A2B_GPIOD0MSK	GPIO Over Distance Mask 0 Register
A2B_GPIOD1MSK	GPIO Over Distance Mask 1 Register
A2B_GPIOD2MSK	GPIO Over Distance Mask 2 Register
A2B_GPIOD3MSK	GPIO Over Distance Mask 3 Register
A2B_GPIOD4MSK	GPIO Over Distance Mask 4 Register
A2B_GPIOD5MSK	GPIO Over Distance Mask 5 Register
A2B_GPIOD6MSK	GPIO Over Distance Mask 6 Register
A2B_GPIOD7MSK	GPIO Over Distance Mask 7 Register
A2B_GPIODDAT	GPIO Over Distance Data Register
A2B_GPIODINV	GPIO Over Distance Invert Register
A2B_MBOX0CTL	Mailbox 0 Control Register (Slave Only)
A2B_MBOX0STAT	Mailbox 0 Status Register (Slave Only)
A2B_MBOX0B0	Mailbox 0 Byte 0 Register (Slave Only)
A2B_MBOX0B1	Mailbox 0 Byte 1 Register (Slave Only)
A2B_MBOX0B2	Mailbox 0 Byte 2 Register (Slave Only)
A2B_MBOX0B3	Mailbox 0 Byte 3 Register (Slave Only)
A2B_MBOX1CTL	Mailbox 1 Control Register (Slave Only)
A2B_MBOX1STAT	Mailbox 1 Status Register (Slave Only)
A2B_MBOX1B0	Mailbox 1 Byte 0 Register (Slave Only)
A2B_MBOX1B1	Mailbox 1 Byte 1 Register (Slave Only)
A2B_MBOX1B2	Mailbox 1 Byte 2 Register (Slave Only)
A2B_MBOX1B3	Mailbox 1 Byte 3 Register (Slave Only)

I2C Chip Address Register (Slave Only)

The A2B_CHIP register stores a 7-bit I²C chip address. It is used during I²C transactions to address a remote peripheral device connected to a slave node. The A²B slave node acts as the I²C master in I²C transactions with peripherals. This register only has an effect on I²C when it is programmed in a slave node. The register can be written to and read from in a master node without any influence on the chip's functionality.

Address: 0x00

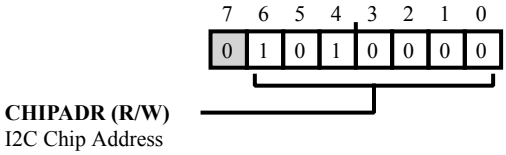

Figure 7-1: A2B_CHIP Register Diagram

Table 7-2: A2B_CHIP Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
6:0 (R/W)	CHIPADR	I2C Chip Address. The A2B_CHIP.CHIPADR bit field stores the I²C address used by a slave transceiver for I²C accesses to a locally-connected peripheral. The A²B slave node acts as the I²C master in I²C transactions with peripherals.

Node Address Register (Master Only)

The A2B_NODEADR register provides control bits for addressing slave nodes through the A²B bus. This register can only be written in the master node. A write to this address in a slave node has no effect.

Address: 0x01

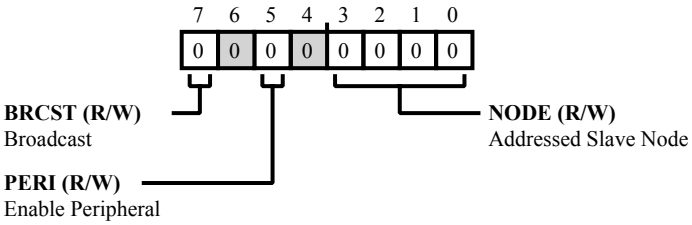

Figure 7-2: A2B_NODEADR Register Diagram

Table 7-3: A2B_NODEADR Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
7 (R/W)	BRCST	Broadcast. The A2B_NODEADR.BRCST bit enables broadcast mode. When an I²C write with BUS_ADDR occurs in broadcast mode, the same control data is written to all nodes (master and slaves) simultaneously. The broadcast allows simultaneous control of all discovered A²B transceivers, but not their respective I²C peripherals. Therefore, clear the A2B_NODEADR.PERI bit (=0) when the A2B_NODEADR.BRCST bit is set to 1.	
		0	Normal, directed register access
		1	Write to all nodes handled as broadcast access
5 (R/W)	PERI	Enable Peripheral. The A2B_NODEADR.PERI bit enables register access (over I²C) of peripheral devices on slave nodes. The A2B_NODEADR.BRCST bit must be cleared (=0) when the A2B_NODEADR.PERI bit is set. When accessing slave node registers through BUS_ADDR, the A2B_NODEADR.PERI bit must be cleared.	
		0	Remote peripheral access disabled
		1	Remote peripheral access enabled
3:0 (R/W)	NODE	Addressed Slave Node. The A2B_NODEADR.NODE bit field selects a slave node by its address. Addresses are assigned based on the position in the A²B topology, starting with address 0 for the node connected directly to the master. The value of the A2B_NODEADR.NODE field is irrelevant when the A2B_NODEADR.BRCST bit is set.	
		0-9	Node number
		10-15	Reserved

Vendor ID Register

The A2B_VENDOR register identifies the part as manufactured by Analog Devices.

Address: 0x02

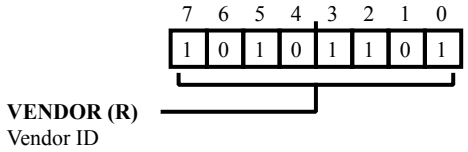

Figure 7-3: A2B_VENDOR Register Diagram

Table 7-4: A2B_VENDOR Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
7:0 (R/NW)	VENDOR	Vendor ID. The A2B_VENDOR.VENDOR bit field contains the vendor identification number of the transceiver chip.

Product ID Register

The A2B_PRODUCT register identifies the last two digits of the part number in hexadecimal format (for example, 0x26=AD2426W).

Address: 0x03

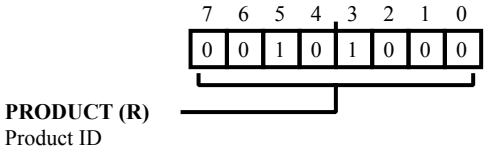

Figure 7-4: A2B_PRODUCT Register Diagram

Table 7-5: A2B_PRODUCT Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
7:0 (R/NW)	PRODUCT	Product ID. The A2B_PRODUCT.PRODUCT bit field contains the product identification number of the transceiver.

Version ID Register

The A2B_VERSION register identifies the version of the part.

Address: 0x04

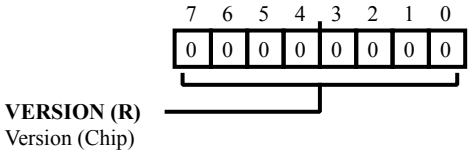

Figure 7-5: A2B_VERSION Register Diagram

Table 7-6: A2B_VERSION Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
7:0 (R/NW)	VERSION	Version (Chip). The A2B_VERSION.VERSION bit field contains the production version number of the chip. Bits 7:4 indicate major product revisions, while bits 3:0 are for minor revisions.

Capability ID Register

The A2B_CAPABILITY register identifies available control interfaces. Transceivers that have an EEPROM storage device connected can store specific descriptor information in the EEPROM module.

Address: 0x05

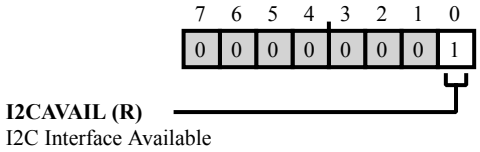

Figure 7-6: A2B_CAPABILITY Register Diagram

Table 7-7: A2B_CAPABILITY Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
0 (R/NW)	I2CAVAIL	I2C Interface Available. The A2B_CAPABILITY.I2CAVAIL bit signals availability of the I^2C interface on the transceiver for access to peripheral devices. If this bit is set (=1), module descriptor information can be accessible through the I^2C interface. A connected EEPROM (for example, an AT24C01) with module descriptor information must have an I^2C device address of 0x50.	
		0	No I^2C interface is available
		1	I^2C interface is available

Switch Control Register

The A2B_SWCTL register controls the switching of A²B bus power onto the downstream B-side of the A²B bus. This register must be written over the A²B bus. A write to this register from the local I²C port has no effect.

Note: The A2B_SWCTL.DIAGMODE bit must only be set when localizing the faults. Under all other conditions, the bit must be cleared to ensure proper operation of the device.

Address: 0x09

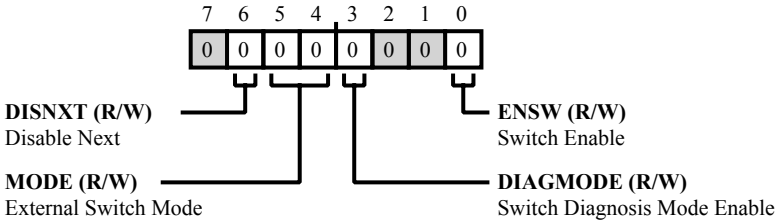

Figure 7-7: A2B_SWCTL Register Diagram

Table 7-8: A2B_SWCTL Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
6 (R/W)	DISNXT	Disable Next. The A2B_SWCTL.DISNXT bit controls when packets are sent to the next node after the switch is enabled (A2B_SWCTL.ENSW=1). When A2B_SWCTL.DISNXT is cleared, synchronization packets are automatically passed to the next node immediately after the A2B_SWSTAT.FIN bit is set by the transceiver (signaling successful switching). When set, synchronization packets are not sent to the next node. A²B bus activity does not commence until discovery frames are issued when the A2B_DISCVRY register is programmed.	
		0	Enable synchronization packets
		1	Disable synchronization packets

Table 7-8: A2B_SWCTL Register Fields (Continued)

Bit No. (Access)	Bit Name	Description/Enumeration	
5:4 (R/W)	MODE	External Switch Mode. The A2B_SWCTL.MODE bit field defines the diagnostic fault detection method for biasing the B-side A^{2}B bus with bus power for the next node. The setting depends on the external hardware configuration. When A2B_SWCTL.MODE= 0, the internal switch is configured for negative bias on the VSSN pin, and an external switch is required on the SWP pin for full line diagnostics, as well as localization and automatic line isolation after a fault is detected. When A2B_SWCTL.MODE= 1, the downstream node is not using A^{2}B bus power and is not properly terminating the bias. In this mode, open and reverse wire faults are not diagnosed, but all other fault types are diagnosed as long as the hardware configuration of the local node is as described for mode 0. When A2B_SWCTL.MODE=2, the voltage on the VIN pin (for example, 5 V) differs significantly from the bias voltage (8 V) on the SENSE pin. This applies when an extra regulator feeds the VIN pin.	
		0	Use internal switch for VSSN pin and external switch for SWP pin
		1	Downstream node not using A^{2}B bus power and not properly terminating the bias
		2	Voltage on the VIN pin
		3	Reserved
3 (R/W)	DIAGMODE	Switch Diagnosis Mode Enable. The A2B_SWCTL.DIAGMODE bit enables switch diagnosis mode.	
		0	Switch diagnosis mode disabled
		1	Switch diagnosis mode enabled
0 (R/W)	ENSW	Switch Enable. The A2B_SWCTL.ENSW bit enables A^{2}B bus power switching.	
		0	Switch disabled
		1	Switch enabled

Broadcast Downstream Slots Register (Slave Only)

In a slave node, the A2B_BCDNSLOTS register defines the number of data slots that are captured by the node and also passed downstream (B-side) as broadcast data to the next node. If any bits are set in the A2B_DNMASK0 through A2B_DNMASK3 registers, the value of the A2B_BCDNSLOTS register is ignored. Changes to this register only take effect after setting the A2B_CONTROL.NEWSTRCT bit of the master node. The A2B_BCDNSLOTS register is not used in the master node.

Address: 0x0A

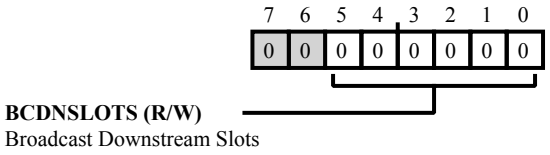

Figure 7-8: A2B_BCDNSLOTS Register Diagram

Table 7-9: A2B_BCDNSLOTS Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
5:0 (R/W)	BCDNSLOTS	Broadcast Downstream Slots. The A2B_BCDNSLOTS.BCDNSLOTS bit field configures the number of broadcast downstream slots. This field must be programmed with a value between 0 and 32.

Local Downstream Slots Register (Slave Only)

In a slave node, the meaning of the A2B_LDNSLOTS register changes depending on whether or not the downstream broadcast mask enable bit (A2B_LDNSLOTS.DNMASKEN) is set. If A2B_LDNSLOTS.DNMASKEN=0 (default), the A2B_LDNSLOTS register defines the number of data slots which are captured by the local node during the downstream portion of the superframe. These data slots are consumed by the node and are not passed downstream to the next node. If A2B_LDNSLOTS.DNMASKEN=1, the A2B_LDNSLOTS register defines the number of data slots that are added by the local node during the downstream portion of the superframe after A2B_DNSLOTS.DNSLOTS data slots are passed downstream by the transceiver. Changes to this register only take effect after setting the A2B_CONTROL.NEWSTRCT bit in the master node.

Address: 0x0B

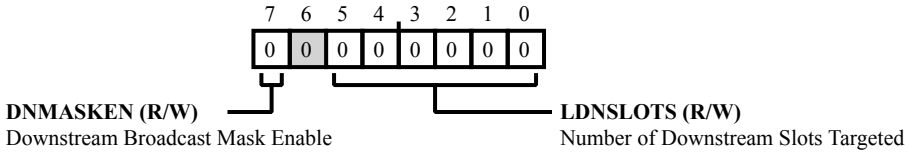

Figure 7-9: A2B_LDNSLOTS Register Diagram

Table 7-10: A2B_LDNSLOTS Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
7 (R/W)	DNMASKEN	Downstream Broadcast Mask Enable. The A2B_LDNSLOTS.DNMASKEN bit enables the downstream mask enable bits in the A2B_DNMASK0 through A2B_DNMASK3 registers.	
		0	Downstream data slot masks disabled
		1	Downstream data slot masks enabled
5:0 (R/W)	LDNSLOTS	Number of Downstream Slots Targeted. When A2B_LDNSLOTS.DNMASKEN=0, the A2B_LDNSLOTS.LDNSLOTS bit field defines the number of data slots which are captured by the local node during the downstream portion of the superframe. When A2B_LDNSLOTS.DNMASKEN=1, the A2B_LDNSLOTS.LDNSLOTS bit field defines the number of data slots which are added by the local node during the downstream portion of the superframe. This field must be programmed with a value between 0 and 32 and be sufficient to accommodate all the data relative to its mode of TDM operation and the number of enabled data pins.	

Local Upstream Slots Register (Slave Only)

In a slave node, the A2B_LUPSLOTS register defines the number of data slots which are added by the local node during the upstream portion of the superframe. Changes to this register only take effect after setting the A2B_CONTROL.NEWSTRCT bit in the master node. The A2B_LUPSLOTS register is not used in the master node.

Address: 0x0C

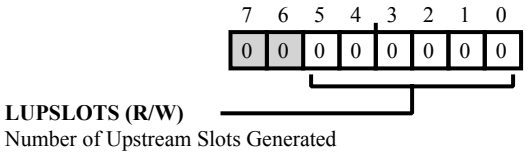

Figure 7-10: A2B_LUPSLOTS Register Diagram

Table 7-11: A2B_LUPSLOTS Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
5:0 (R/W)	LUPSLOTS	Number of Upstream Slots Generated. The A2B_LUPSLOTS.LUPSLOTS bit field defines the number of data slots which are added by the transceiver during the upstream portion of the superframe. These bits must be programmed with a value between 0 and 32.

Downstream Slots Register

In a slave node, the A2B_DNSLOTS register defines the number of data slots (not including broadcast slots) that are passed downstream (B-side) after the transceiver begins to capture data slots. In the master node, the A2B_DNSLOTS register defines the total number of downstream data slots (including broadcast slots). Changes to this register only take effect after setting the A2B_CONTROL.NEWSTRCT bit in the master node.

Address: 0x0D

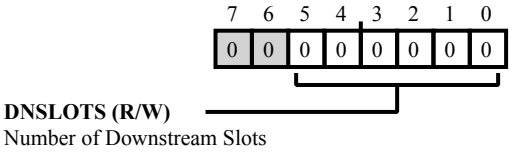

Figure 7-11: A2B_DNSLOTS Register Diagram

Table 7-12: A2B_DNSLOTS Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
5:0 (R/W)	DNSLOTS	Number of Downstream Slots. In a master node, the A2B_DNSLOTS.DNSLOTS bit field is the number of downstream slots, including broadcast data slots. It must be sufficient to accommodate the data intended for downstream devices, which is a function of the TDM mode and the number of enabled data pins. In a slave node, the A2B_DNSLOTS.DNSLOTS bit field sets the number of data slots which are passed downstream. When calculating the value to program to this field, the same guidance as in the master node applies. But, slave nodes must also include any broadcast downstream slots, as programmed in the A2B_BCDNSLOTS register. Valid programming values are between 0 and 32.

Upstream Slots Register

In a slave node, the A2B_UPSLOTS register defines the number of data slots which are passed upstream by the B-side transceiver before the transceiver begins to add data slots. In the master node, the A2B_UPSLOTS register defines the total number of upstream data slots. Changes to this register only take effect after setting the A2B_CONTROL.NEWSTRCT bit in the master node.

Address: 0x0E

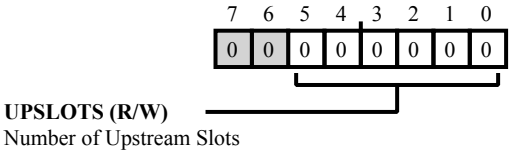

Figure 7-12: A2B_UPSLOTS Register Diagram

Table 7-13: A2B_UPSLOTS Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
5:0 (R/W)	UPSLOTS	Number of Upstream Slots. In a master node, the A2B_UPSLOTS.UPSLOTS bit field is the number of upstream slots being received from the first-in-line slave node. It must be sufficient to accommodate all data intended for upstream devices, which is a function of TDM serial mode and the number of enabled data pins. In a slave node, the A2B_UPSLOTS.UPSLOTS bit field defines the number of data slots which are received from the next-in-line slave node and passed upstream before the transceiver begins to add data slots. Valid programming values are between 0 and 32.

Response Cycles Register

The A2B_RESPCYCS register defines the time between the start of the downstream header (the first SCF preamble bit) and the start of the upstream header (the first SCF preamble bit). The value in the register represents the number of bus bit times multiplied by 4. 1024 bit counts are in an A^2B superframe between SCFs. One bus bit time = $1/(f_{SYSBCLK})$.

The A2B_DISCVRY register in the master transceiver is programmed with the A2B_RESPCYCS register value during discovery. Changes to this register only take effect after setting the A2B_CONTROL.NEWSTRCT bit in the master node. This register must be written over the A^2B bus, as writes to this register from the local I^2C port have no effect.

Address: 0x0F

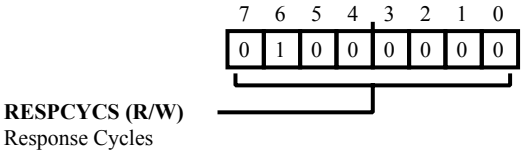

Figure 7-13: A2B_RESPCYCS Register Diagram

Table 7-14: A2B_RESPCYCS Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
7:0 (R/W)	RESPCYCS	Response Cycles. The A2B_RESPCYCS.RESPCYCS bit field is one-fourth the time (in terms of bus bits) from the start of a downstream frame to the start of an upstream frame.

Slot Format Register (Master Only, Auto-Broadcast)

The A2B_SLOTFMT register defines the size and format of the downstream and upstream data slots. Floating-point compression of A^{2}B data can be enabled to reduce bandwidth using this register, and ECC protection of A^{2}B data can alternately be enabled. All nodes in an A^{2}B system are subject to the same upstream and downstream slot format setting. Changes to this register only take effect after setting the A2B_CONTROL.NEWSTRCT bit in the master node.

When the A2B_SLOTFMT register is written in the master node, the new setting is automatically broadcast to all discovered slave nodes over the A^{2}B bus. Local host writes to this register in a slave node have no effect.

Address: 0x10

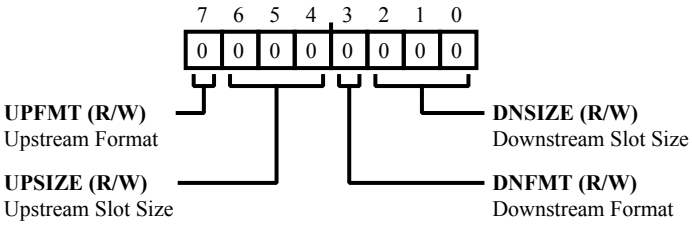

Figure 7-14: A2B_SLOTFMT Register Diagram

Table 7-15: A2B_SLOTFMT Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
7 (R/W)	UPFMT	Upstream Format. The A2B_SLOTFMT.UPFMT bit configures the format of the upstream data on the A^{2}B bus bus. When A2B_SLOTFMT.UPFMT= 0, the format of the data on the A^{2}B bus bus is normal (no compression, no ECC protection, and one parity bit). When A2B_SLOTFMT.UPFMT = 1, an alternate data format is utilized, depending on the upstream data width (A2B_SLOTFMT.UPSIZE). When the A2B_SLOTFMT.UPSIZE bit is programmed for 12-, 16-, or 20-bit data, setting the A2B_SLOTFMT.UPFMT bit enables floating-point compression of upstream data. When this compression is used, the I^{2}S/TDM or PDM data is 4 bits wider than the A^{2}B data, which is compressed to reduce A^{2}B bus bandwidth, and the data is protected by a parity bit. When the A2B_SLOTFMT.UPSIZE bit is programmed for 24- or 32-bit data, setting the A2B_SLOTFMT.UPFMT bit enables ECC protection on upstream data slots, where ECC bits are added to each data slot instead of a parity bit (6 ECC bits for 24-bit data, 7 ECC bits for 32-bit data). Setting the A2B_SLOTFMT.UPFMT bit when A2B_SLOTFMT.UPSIZE is programmed for 8- or 28-bit data has no effect.	
		0	Normal upstream data slot format
		1	Alternate upstream data slot format

Table 7-15: A2B_SLOTFMT Register Fields (Continued)

Bit No. (Access)	Bit Name	Description/Enumeration	
6:4 (R/W)	UPSIZE	Upstream Slot Size. The A2B_SLOTFMT.UPSIZE bit field selects the upstream data slot size.	
		0	8 bits
		1	12 bits
		2	16 bits
		3	20 bits
		4	24 bits
		5	28 bits
		6	32 bits
3 (R/W)	DNFMT	Downstream Format. The A2B_SLOTFMT.DNFMT bit configures the format of the downstream data on the A^{2}B bus bus. When A2B_SLOTFMT.DNFMT= 0, the format of the data on the A^{2}B bus bus is normal (no compression, no ECC protection, and one parity bit). When A2B_SLOTFMT.DNFMT = 1, an alternate data format is utilized, depending on the downstream data width (A2B_SLOTFMT.DNSIZE). When the A2B_SLOTFMT.DNSIZE field is programmed for 12-, 16-, or 20-bit data, setting the A2B_SLOTFMT.DNFMT bit enables floating-point compression of downstream data. When this compression is used, the I^{2}S/TDM or PDM data is 4 bits wider than the A^{2}B data, which is compressed to reduce A^{2}B bus bandwidth, and the data is protected by a parity bit. When the A2B_SLOTFMT.DNSIZE bit is programmed for 24- or 32-bit data, setting the A2B_SLOTFMT.DNFMT bit enables ECC protection on downstream data slots, where ECC bits are added to each data slot instead of a parity bit (6 ECC bits for 24-bit data, 7 ECC bits for 32-bit data). Setting the A2B_SLOTFMT.DNFMT bit when A2B_SLOTFMT.DNSIZE is programmed for 8- or 28-bit data has no effect.	
		0	Normal downstream data slot format
		1	Alternate downstream data slot format
2:0 (R/W)	DNSIZE	Downstream Slot Size. The A2B_SLOTFMT.DNSIZE bit field selects the downstream data slot size.	
		0	8 bits
		1	12 bits
		2	16 bits
		3	20 bits
		4	24 bits

Table 7-15: A2B_SLOTFMT Register Fields (Continued)

Bit No. (Access)	Bit Name	Description/Enumeration	
		5	28 bits
		6	32 bits
		7	Reserved

Data Control Register (Master Only, Auto-Broadcast)

The A2B_DATCTL register is used to enable data slots and standby mode on the A²B bus. Changes to this register only take effect after setting the A2B_CONTROL.NEWSTRCT bit in the master node. When the A2B_DATCTL register is written in the master node, the new setting is automatically broadcast to all discovered slave node over the A²B bus. Local host writes to this register in a slave node have no effect.

NOTE: To switch back to normal operation, first exit the standby mode by clearing the A2B_DATCTL.STANDBY bit, then write to the A2B_DATCTL register to enable the upstream and downstream slots.

Address: 0x11

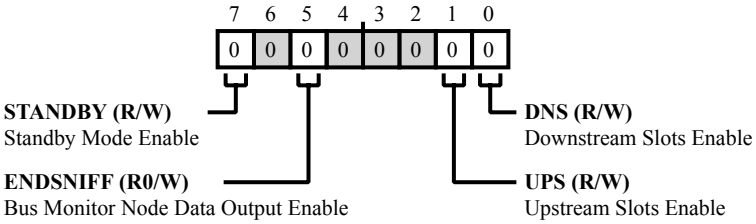

Figure 7-15: A2B_DATCTL Register Diagram

Table 7-16: A2B_DATCTL Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
7 (R/W)	STANDBY	Standby Mode Enable. The A2B_DATCTL.STANDBY bit globally enables power saving mode for all nodes and minimizes bus activity. The only traffic required is a minimal downstream preamble to keep all of the PLLs in the slave nodes synchronized. Reads and writes across the A²B bus are not supported in this mode.	
		0	Disabled
		1	Enabled
5 (R0/W)	ENDSNIFF	Bus Monitor Node Data Output Enable. The A2B_DATCTL.ENDSNIFF bit controls whether or not an attached Bus Monitor Node will produce data slots as output.	
		0	Disabled
		1	Enabled
1 (R/W)	UPS	Upstream Slots Enable. The A2B_DATCTL.UPS bit globally enables upstream synchronous data to be sent over the bus.	
		0	Disabled
		1	Enabled

Table 7-16: A2B_DATCTL Register Fields (Continued)

Bit No. (Access)	Bit Name	Description/Enumeration	
0 (R/W)	DNS	Downstream Slots Enable. The `A2B_DATCTL.DNS` bit globally enables downstream synchronous data to be sent over the bus.	
		0	Disabled
		1	Enabled

Control Register

The A2B_CONTROL register provides bits which control nodes on the bus.

Address: 0x12

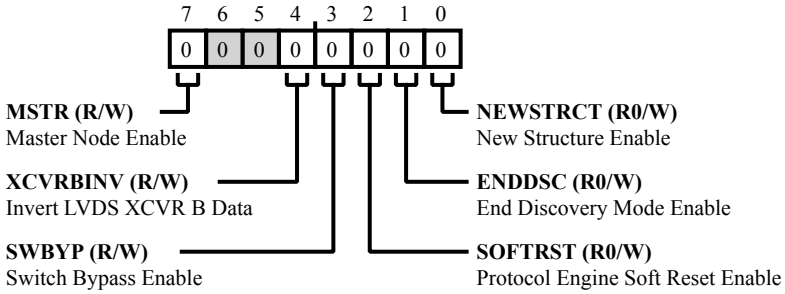

Figure 7-16: A2B_CONTROL Register Diagram

Table 7-17: A2B_CONTROL Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
7 (R/W)	MSTR	Master Node Enable. The A2B_CONTROL.MSTR bit controls whether the current node is a slave node or a master node.	
		0	Slave node
		1	Master node
4 (R/W)	XCVRBINV	Invert LVDS XCVR B Data. The A2B_CONTROL.XCVRBINV bit controls an optional inversion of data to/from LVDS XCVR B. Data is inverted when this bit is set.	
3 (R/W)	SWBYP	Switch Bypass Enable. The A2B_CONTROL.SWBYP bit enables the downstream LVDS XCVR without waiting for the line switch to be turned on. When this bit is set the line switch will not be enabled even if A2B_SWCTL.ENSW is set.	
2 (R0/W)	SOFTRST	Protocol Engine Soft Reset Enable. When the A2B_CONTROL.SOFTRST bit is set, the protocol engine in the bus node is reset, and all registers return to their respective reset states.	
		0	No action
		1	Reset protocol engine

Table 7-17: A2B_CONTROL Register Fields (Continued)

Bit No. (Access)	Bit Name	Description/Enumeration	
1 (R0/W)	ENDDSC	End Discovery Mode Enable. In the master node, setting the `A2B_CONTROL.ENDDSC` bit ends discovery attempts to a new slave node.	
		0	No action
		1	End discovery
0 (R0/W)	NEWSTRCT	New Structure Enable. The `A2B_CONTROL.NEWSTRCT` bit synchronously applies a new structure to all nodes. When the `A2B_CONTROL.NEWSTRCT` bit is set in the master node, a new structure is applied within 5 superframe cycles unless communication errors create delays.	
		0	No action
		1	Enable new structure

Discovery Register (Master Only)

Programming the A2B_DISCVRY register with a response cycle value for a new node to be added allows the new slave node to be discovered. It triggers the start of full discovery frames being sent to the next-in-line slave node.

When the A2B_DISCVRY register is written in the master node, the new setting is automatically broadcast to all slave nodes over the A^{2}B bus. Local host and direct BUS_ADDR writes to this register in a slave node have no effect.

Address: 0x13

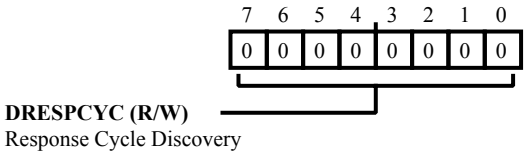

Figure 7-17: A2B_DISCVRY Register Diagram

Table 7-18: A2B_DISCVRY Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
7:0 (R/W)	DRESPCYC	Response Cycle Discovery. The A2B_DISCVRY.DRESPCYC bit field is written with the value to be used for A2B_RESPCYCS by a to-be discovered slave node.

Switch Status Register

The A2B_SWSTAT register provides line diagnostics status information. Line diagnostics are performed when bias is switched onto the A²B bus towards the next-in-line slave node.

Address: 0x14

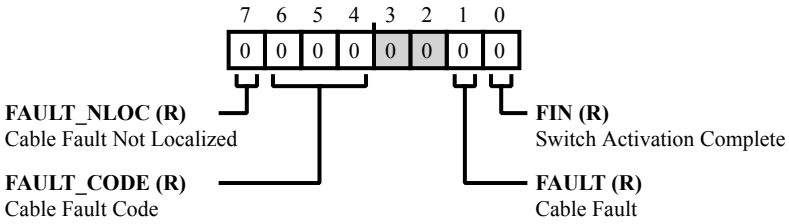

Figure 7-18: A2B_SWSTAT Register Diagram

Table 7-19: A2B_SWSTAT Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
7 (R/NW)	FAULT_NLOC	Cable Fault Not Localized. The A2B_SWSTAT.FAULT_NLOC bit indicates that the identified line fault is not localized.	
		0	Switch fault localized
		1	Switch fault not localized
6:4 (R/NW)	FAULT_CODE	Cable Fault Code. The A2B_SWSTAT.FAULT_CODE bit field contains downstream link cable diagnostic error codes.	
		0	No fault detected
		1	Cable terminal shorted to GND
		2	Cable terminal shorted to VBAT
		3	Cable terminals shorted together
		4	Cable disconnected or open circuit
		5	Cable is reverse connected
		6	Reserved
		7	Undetermined fault
1 (R/NW)	FAULT	Cable Fault. The A2B_SWSTAT.FAULT bit indicates a cable fault has been detected.	
		0	Cable fault not detected
		1	Cable fault detected

Table 7-19: A2B_SWSTAT Register Fields (Continued)

Bit No. (Access)	Bit Name	Description/Enumeration	
0 (R/NW)	FIN	Switch Activation Complete. The `A2B_SWSTAT.FIN` bit indicates the successful completion of the switch activation sequence for biasing of the downstream link. When this bit is set the transceiver begins passing SCFs to the next-in-line slave, thus allowing it to begin locking its PLL, unless the switch is bypassed (`A2B_CONTROL.SWBYP` = 1)	
		0	Switch is open or has not completed closing
		1	Switch completed closing

Interrupt Status Register

The A2B_INTSTAT register contains interrupt status information for the node.

Address: 0x15

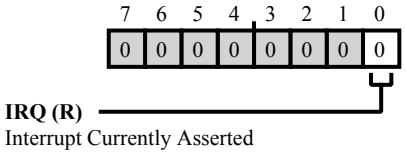

Figure 7-19: A2B_INTSTAT Register Diagram

Table 7-20: A2B_INTSTAT Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
0 (R/NW)	IRQ	Interrupt Currently Asserted. When the A2B_INTSTAT.IRQ bit is set, the node is signaling an interrupt request, either through the IRQ pin for a master node or over the A^{2}B bus for a slave node.	
		0	No interrupt request
		1	Interrupt request

Interrupt Source Register (Master Only)

The A2B_INTSRC register contains information about the current highest priority interrupt. It is updated when the A2B_INTTYPE register is read. A value of 0x00 in this register indicates that no interrupts are present.

Address: 0x16

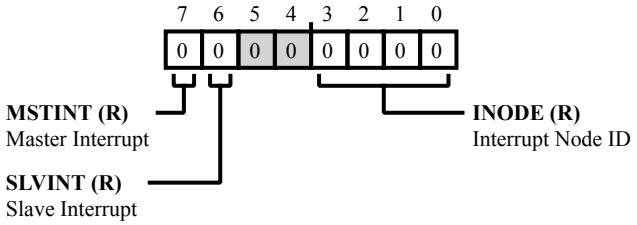

Figure 7-20: A2B_INTSRC Register Diagram

Table 7-21: A2B_INTSRC Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
7 (R/NW)	MSTINT	Master Interrupt. When the A2B_INTSRC.MSTINT bit is set, the current interrupt is being generated by the master node.
6 (R/NW)	SLVINT	Slave Interrupt. When the A2B_INTSRC.SLVINT bit is set, the current interrupt is being generated by a slave node.
3:0 (R/NW)	INODE	Interrupt Node ID. The A2B_INTSRC.INODE bit field contains the node number of the slave node that asserted the current interrupt.

Interrupt Type Register (Master Only)

The A2B_INTTYPE register contains information about the pending interrupt being generated by the node indicated in the A2B_INTSRC register and signaled with the IRQ pin. A host read of the A2B_INTTYPE register in the master node clears this pending interrupt in the master and deasserts the IRQ pin. If other interrupts are pending, the A2B_INTSRC and A2B_INTTYPE registers are updated to reflect the highest priority pending interrupt, and the IRQ pin will again be asserted. Nodes closer to the master have a higher priority when the same interrupt appears in more than one slave node.

Address: 0x17

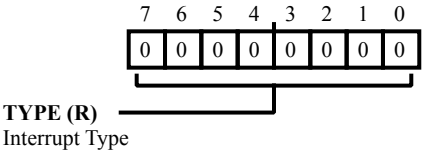

Figure 7-21: A2B_INTTYPE Register Diagram

Table 7-22: A2B_INTTYPE Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
7:0 (R/NW)	TYPE	Interrupt Type. The A2B_INTTYPE.TYPE bit field contains the current interrupt type. Interrupt types are described in the interrupt pending registers (A2B_INTPND0 through A2B_INTPND2).	
		0	HDCNTERR - Header count error
		1	DDERR - Data decoding error
		2	CRCERR - CRC error
		3	DPERR - Data parity error
		4	BECOVF - Bit error counter overflow error
		5	SRFERR - SRF miss error
		6	SRFCRCERR - SRF CRC error (slave only)
		9	PWRERR - Positive terminal BP shorted to GND
		10	PWRERR - Negative terminal BN shorted to VBAT
		11	PWRERR - BP shorted to BN
		12	PWRERR - Cable disconnected or open circuit or wrong port
		13	PWRERR - Cable is reverse connected or wrong port
		15	PWRERR - Undetermined fault
		16	IO0PND - GP input IO0 interrupt (slave only)

Table 7-22: A2B_INTTYPE Register Fields (Continued)

Bit No. (Access)	Bit Name	Description/Enumeration	
		17	IO1PND - GP input IO1 interrupt
		18	IO2PND - GP input IO2 interrupt
		19	IO3PND - GP input IO3 interrupt
		20	IO4PND - GP input IO4 interrupt
		21	IO5PND - GP input IO5 interrupt
		22	IO6PND- GP input IO6 interrupt
		23	IO7PND - GP input IO7 interrupt
		24	DSCDONE - Discovery done interrupt (master only)
		25	I2CERR - I2C error (master only)
		26	ICRCERR - Interrupt CRC error (master only)
		41	PWRERR - Non-localized negative terminal BN short to GND
		42	PWRERR - Non-localized positive terminal BP short to VBAT
		48	Mailbox 0 full
		49	Mailbox 0 empty
		50	Mailbox 1 full
		51	Mailbox 1 empty
		128	Interrupt messaging error
		252	Startup error - Return to factory
		253	Slave INTTYPE read error - Master only
		254	Standby done - Master only
		255	MSTR_RUNNING - Master node PLL locked

Interrupt Pending 0 Register

The A2B_INTPND0 register contains interrupt pending bits for the node.

Address: 0x18

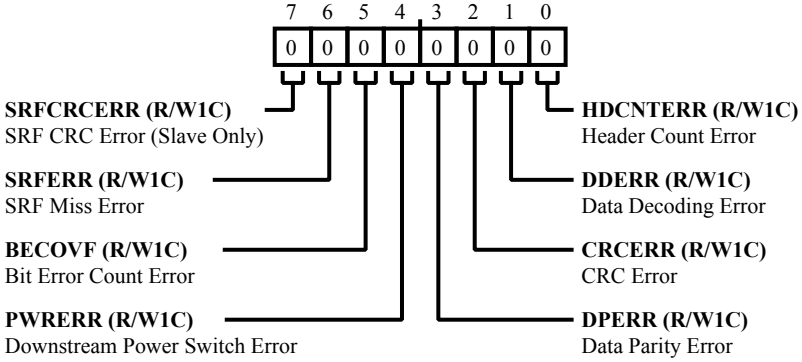

Figure 7-22: A2B_INTPND0 Register Diagram

Table 7-23: A2B_INTPND0 Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
7 (R/W1C)	SRFCRCERR	SRF CRC Error (Slave Only). The A2B_INTPND0.SRFCRCERR bit indicates that the current slave node has detected a CRC error in the SRF.	
		0	No SRF CRC error
		1	SRF CRC error detected
6 (R/W1C)	SRFERR	SRF Miss Error. The A2B_INTPND0.SRFERR bit indicates that the node has not received the SRF from the downstream node at the specified time.	
		0	No SRF miss error
		1	SRF miss error detected
5 (R/W1C)	BECOVF	Bit Error Count Error. The A2B_INTPND0.BECOVF bit indicates that the number of errors programmed into the bit error count control register has been exceeded.	
		0	No BEC error pending
		1	BEC error pending

Table 7-23: A2B_INTPND0 Register Fields (Continued)

Bit No. (Access)	Bit Name	Description/Enumeration	
4 (R/W1C)	PWRERR	Downstream Power Switch Error. The `A2B_INTPND0.PWRERR` bit indicates an error reported from the downstream power switch.	
		0	No power error
		1	Downstream power switch error
3 (R/W1C)	DPERR	Data Parity Error. The `A2B_INTPND0.DPERR` bit indicates that the current node has detected a data parity error. The error is detected only if the node consumes the data slot with a data parity error.	
		0	No data parity error
		1	Data parity error detected
2 (R/W1C)	CRCERR	CRC Error. The `A2B_INTPND0.CRCERR` bit indicates that the current node has detected a CRC error. For the master node, this applies to a CRC error in the SRF. For a slave node, this applies to a CRC error in the SCF.	
		0	No CRC error
		1	CRC error detected
1 (R/W1C)	DDERR	Data Decoding Error. The `A2B_INTPND0.DDERR` bit indicates that the current node has detected a data decoding error. The error is detected only if the node consumes the data slot with a data decoding error.	
		0	No data decoding error
		1	Data decoding error detected
0 (R/W1C)	HDCNTERR	Header Count Error. The `A2B_INTPND0.HDCNTERR` bit indicates the current node has detected a header count error. For the master node, this means that the SRF has a different count value than expected. For a slave node, this means that the SRF has a different value than expected.	
		0	No header count error
		1	Header count error detected

Interrupt Pending 1 Register

The A2B_INTPND1 register contains interrupt pending bits for the node.

Address: 0x19

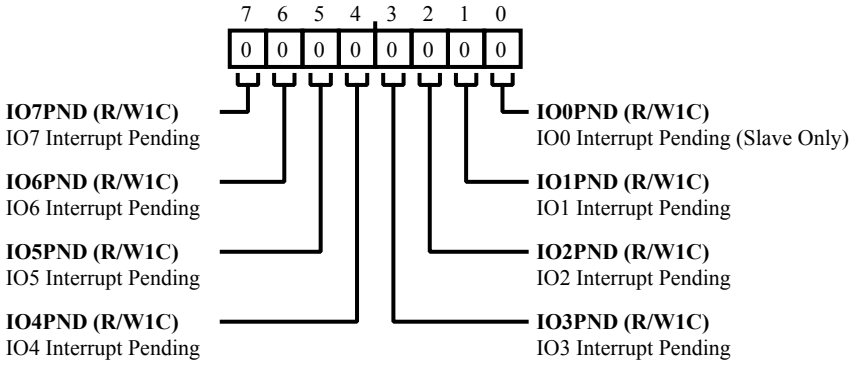

Figure 7-23: A2B_INTPND1 Register Diagram

Table 7-24: A2B_INTPND1 Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
7 (R/W1C)	IO7PND	IO7 Interrupt Pending. The A2B_INTPND1.IO7PND bit indicates that a pin interrupt request from IO7 is pending.	
		0	No interrupt pending
		1	Interrupt pending <inherit>
6 (R/W1C)	IO6PND	IO6 Interrupt Pending. The A2B_INTPND1.IO6PND bit indicates that a pin interrupt request from IO6 (DRX1) is pending.	
		0	No interrupt pending
		1	Interrupt pending
5 (R/W1C)	IO5PND	IO5 Interrupt Pending. The A2B_INTPND1.IO5PND bit indicates that a pin interrupt request from IO5 (DRX0) is pending.	
		0	No interrupt pending
		1	Interrupt pending

Table 7-24: A2B_INTPND1 Register Fields (Continued)

Bit No. (Access)	Bit Name	Description/Enumeration	
4 (R/W1C)	IO4PND	IO4 Interrupt Pending. The `A2B_INTPND1.IO4PND` bit indicates that a pin interrupt request from IO4 (DTX1) is pending.	
		0	No interrupt pending
		1	Interrupt pending
3 (R/W1C)	IO3PND	IO3 Interrupt Pending. The `A2B_INTPND1.IO3PND` bit indicates that a pin interrupt request from IO3 (DTX0) is pending.	
		0	No interrupt pending
		1	Interrupt pending
2 (R/W1C)	IO2PND	IO2 Interrupt Pending. The `A2B_INTPND1.IO2PND` bit indicates that a pin interrupt request from IO2 (ADR2) is pending.	
		0	No interrupt pending
		1	Interrupt pending
1 (R/W1C)	IO1PND	IO1 Interrupt Pending. The `A2B_INTPND1.IO1PND` bit indicates that a pin interrupt request from IO1 (ADR1) is pending.	
		0	No interrupt pending
		1	Interrupt pending
0 (R/W1C)	IO0PND	IO0 Interrupt Pending (Slave Only). The `A2B_INTPND1.IO0PND` bit indicates that a pin interrupt request from IO0 (IRQ) is pending. On master nodes, this bit always reads 0.	
		0	No interrupt pending <inherit>
		1	Interrupt pending

Interrupt Pending 2 Register (Master Only)

The A2B_INTPND2 register contains interrupt pending bits relevant only to master nodes.

Address: 0x1A

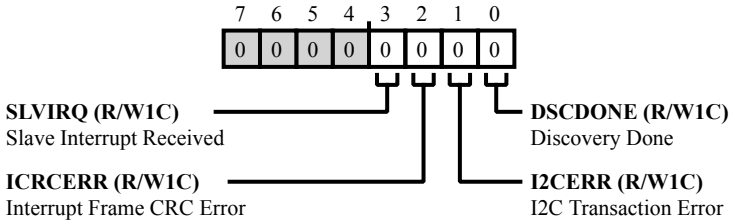

Figure 7-24: A2B_INTPND2 Register Diagram

Table 7-25: A2B_INTPND2 Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
3 (R/W1C)	SLVIRQ	Slave Interrupt Received. In the master mode, the A2B_INTPND2.SLVIRQ bit indicates that a slave node has signaled an interrupt to the master node. This bit always reads zero in a slave node.	
		0	No interrupt
		1	Slave node has signaled an interrupt
2 (R/W1C)	ICRCERR	Interrupt Frame CRC Error. In the master mode, the A2B_INTPND2.ICRCERR bit indicates that the master node has detected an interrupt frame CRC error.	
		0	No error
		1	Interrupt frame CRC error detected
1 (R/W1C)	I2CERR	I2C Transaction Error. The A2B_INTPND2.I2CERR bit indicates that an I^2C access error has occurred. Examples of this are an I^2C write to a slave node with early acknowledge that did not complete or a broadcast write that timed out.	
		0	No error
		1	An I^2C access error occurred
0 (R/W1C)	DSCDONE	Discovery Done. The A2B_INTPND2.DSCDONE bit indicates that a new slave node has been discovered. This bit always reads zero in slave nodes.	
		0	No new slave node discovered
		1	New slave node discovered

Interrupt Mask 0 Register

The A2B_INTMSK0 register determines which A2B_INTPND0 register bits generate interrupts.

Address: 0x1B

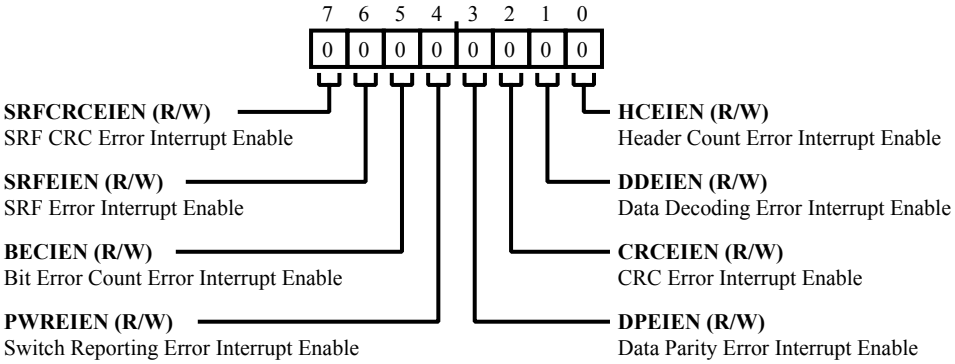

Figure 7-25: A2B_INTMSK0 Register Diagram

Table 7-26: A2B_INTMSK0 Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
7 (R/W)	SRFCRCEIEN	SRF CRC Error Interrupt Enable.
6 (R/W)	SRFEIEN	SRF Error Interrupt Enable.
5 (R/W)	BECIEN	Bit Error Count Error Interrupt Enable.
4 (R/W)	PWREIEN	Switch Reporting Error Interrupt Enable.
3 (R/W)	DPEIEN	Data Parity Error Interrupt Enable.
2 (R/W)	CRCEIEN	CRC Error Interrupt Enable.
1 (R/W)	DDEIEN	Data Decoding Error Interrupt Enable.
0 (R/W)	HCEIEN	Header Count Error Interrupt Enable.

Interrupt Mask 1 Register

The A2B_INTMSK1 register determines which A2B_INTPND1 register bits generate interrupts.

Address: 0x1C

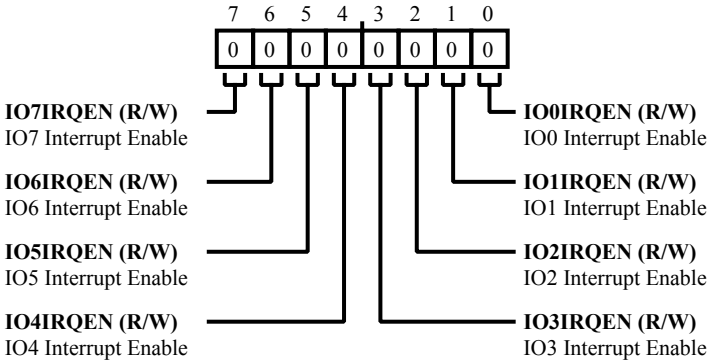

Figure 7-26: A2B_INTMSK1 Register Diagram

Table 7-27: A2B_INTMSK1 Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
7 (R/W)	IO7IRQEN	IO7 Interrupt Enable.
6 (R/W)	IO6IRQEN	IO6 Interrupt Enable.
5 (R/W)	IO5IRQEN	IO5 Interrupt Enable.
4 (R/W)	IO4IRQEN	IO4 Interrupt Enable.
3 (R/W)	IO3IRQEN	IO3 Interrupt Enable.
2 (R/W)	IO2IRQEN	IO2 Interrupt Enable.
1 (R/W)	IO1IRQEN	IO1 Interrupt Enable.
0 (R/W)	IO0IRQEN	IO0 Interrupt Enable.

Interrupt Mask 2 Register (Master Only)

The A2B_INTMSK2 register determines which A2B_INTPND2 register bits generate interrupts.

Address: 0x1D

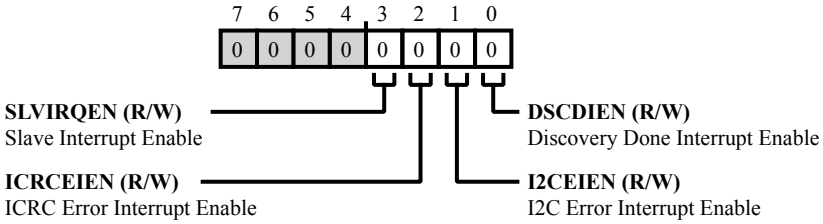

Figure 7-27: A2B_INTMSK2 Register Diagram

Table 7-28: A2B_INTMSK2 Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
3 (R/W)	SLVIRQEN	Slave Interrupt Enable.
2 (R/W)	ICRCEIEN	ICRC Error Interrupt Enable.
1 (R/W)	I2CEIEN	I2C Error Interrupt Enable.
0 (R/W)	DSCDIEN	Discovery Done Interrupt Enable.

Bit Error Count Control Register

The A2B_BECCTL register controls bit error counting, including interrupt thresholds.

Address: 0x1E

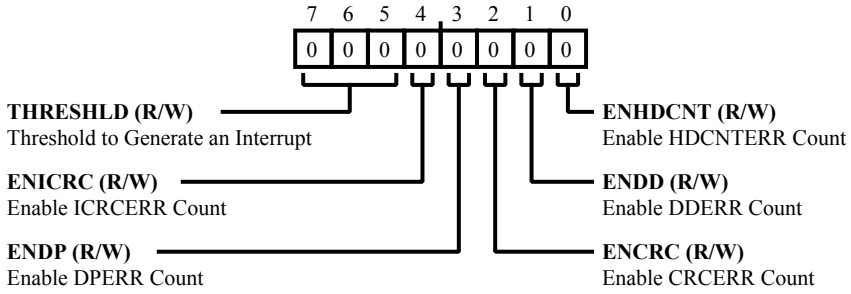

Figure 7-28: A2B_BECCTL Register Diagram

Table 7-29: A2B_BECCTL Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
7:5 (R/W)	THRESHLD	Threshold to Generate an Interrupt. The A2B_BECCTL.THRESHLD bit field configures the number of errors counted before the A2B_INTPND0.BECOVF bit is set.	
		0	Interrupt after 2 errors
		1	Interrupt after 4 errors
		2	Interrupt after 8 errors
		3	Interrupt after 16 errors
		4	Interrupt after 32 errors
		5	Interrupt after 64 errors
		6	Interrupt after 128 errors
		7	Interrupt after 256 errors
4 (R/W)	ENICRC	Enable ICRCERR Count. When the A2B_BECCTL.ENICRC bit is set, the bit error count register is incremented every time a CRC error is detected in the interrupt response frame.	
		0	Disabled
		1	Enable bit error counting

Table 7-29: A2B_BECCTL Register Fields (Continued)

Bit No. (Access)	Bit Name	Description/Enumeration	
3 (R/W)	ENDP	Enable DPERR Count. When the `A2B_BECCTL.ENDP` bit is set, the bit error count register is incremented on every parity error of the streaming data.	
		0	No parity error
		1	Parity error
2 (R/W)	ENCRC	Enable CRCERR Count. When the `A2B_BECCTL.ENCRC` bit is set, the bit error count register is incremented on every CRC error in a control or response frame. This excludes interrupt frame CRC errors.	
		0	No CRC error
		1	CRC error
1 (R/W)	ENDD	Enable DDERR Count. When the `A2B_BECCTL.ENDD` bit is set, the bit error count register is incremented on every data decoding error.	
		0	Disabled
		1	Enabled
0 (R/W)	ENHDCNT	Enable HDCNTERR Count. When the `A2B_BECCTL.ENHDCNT` bit is set, the bit error count register is incremented if there is a discrepancy between the actual and expected header count field.	
		0	Disabled
		1	Enabled

Bit Error Count Register

The A2B_BECNT register accumulates the error count of the error types selected in the A2B_BECCTL register. Any write to this register clears the count.

Address: 0x1F

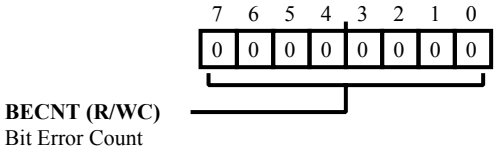

Figure 7-29: A2B_BECNT Register Diagram

Table 7-30: A2B_BECNT Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
7:0 (R/WC)	BECNT	Bit Error Count. The A2B_BECNT.BECNT bit field provides the number of bit errors counted, based on the value programmed into the A2B_BECCTL register.

Testmode Register

The A2B_TESTMODE register provides control bits to be used in testing the A²B link. The A2B_TESTMODE.PRBSDN and A2B_TESTMODE.PRBSUP bits are used to enable the use of pseudo-random data in the downstream and upstream data slots on the A²B bus, respectively. Downstream data is checked in the last slave node based on the programming of the A2B_DNSLOTS, A2B_LDNSLOTS, and A2B_BCDNSLOTS registers. Upstream data is checked in the master node. Data mismatches increment a 32-bit counter (which can be read via the A2B_ERRCNT0 through A2B_ERRCNT3 registers). The A2B_TESTMODE register must be programmed via a broadcast write. Slave to slave communications adversely affect a Bit Error Rate Test (BERT).

Address: 0x20

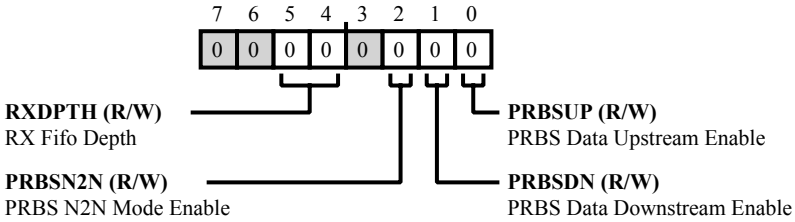

Figure 7-30: A2B_TESTMODE Register Diagram

Table 7-31: A2B_TESTMODE Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
5:4 (R/W)	RXDPTH	RX Fifo Depth. The A2B_TESTMODE.RXDPTH bits control the data recovery FIFO depth.	
		0	Do Not Change FIFO Depth
		1	Increase FIFO Depth by 1
		2	Increase FIFO Depth by 2
		3	Increase FIFO Depth by 2
2 (R/W)	PRBSN2N	PRBS N2N Mode Enable. When the A2B_TESTMODE.PRBSN2N bit is set, each node checks all incoming data bits and transmits the expected data to the next node. This allows for better determination of where bus errors occur. This bit only takes effect when either or both of the A2B_TESTMODE.PRBSDN and A2B_TESTMODE.PRBSUP bits are set.	
		0	Disabled
		1	Enabled

Table 7-31: A2B_TESTMODE Register Fields (Continued)

Bit No. (Access)	Bit Name	Description/Enumeration	
1 (R/W)	PRBSDN	PRBS Data Downstream Enable. The `A2B_TESTMODE.PRBSDN` bit enables PRBS data to be sent downstream towards the last slave node.	
		0	Disable PRBS data
		1	PRBS data
0 (R/W)	PRBSUP	PRBS Data Upstream Enable. The `A2B_TESTMODE.PRBSUP` bit enables PRBS data to be sent upstream towards the master node.	
		0	Disable PRBS data
		1	PRBS data

PRBS Error Count Byte 0 Register

The A2B_ERRCNT0 register holds the least significant byte of the 32-bit error count accumulated during the PRBS bit error test.

Address: 0x21

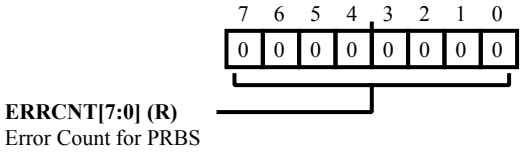

Figure 7-31: A2B_ERRCNT0 Register Diagram

Table 7-32: A2B_ERRCNT0 Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
7:0 (R/NW)	ERRCNT	Error Count for PRBS. The A2B_ERRCNT0.ERRCNT bit field contains one byte of the 32-bit PRBS bit error count.

PRBS Error Count Byte 1 Register

The A2B_ERRCNT1 register holds the second byte (bits 15:8) of the error count accumulated during the PRBS bit error test.

Address: 0x22

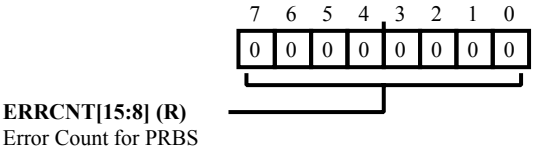

Figure 7-32: A2B_ERRCNT1 Register Diagram

Table 7-33: A2B_ERRCNT1 Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
7:0 (R/NW)	ERRCNT	Error Count for PRBS. The A2B_ERRCNT1.ERRCNT bit field contains one byte of the 32-bit PRBS bit error count.

PRBS Error Count Byte 2 Register

The A2B_ERRCNT2 register holds the third byte (bits 23:16) of the error count accumulated during the PRBS bit error test.

Address: 0x23

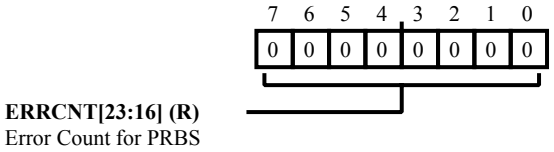

Figure 7-33: A2B_ERRCNT2 Register Diagram

Table 7-34: A2B_ERRCNT2 Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
7:0 (R/NW)	ERRCNT	Error Count for PRBS. The A2B_ERRCNT2.ERRCNT bit field contains one byte of the 32-bit PRBS bit error count.

PRBS Error Count Byte 3 Register

The A2B_ERRCNT3 register holds the most significant byte (bits 31:24) of the 32-bit error count accumulated during the PRBS bit error test. The A2B_ERRCNT0 register is the least significant byte of the 32-bit error count.

Address: 0x24

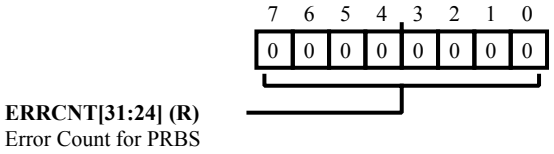

Figure 7-34: A2B_ERRCNT3 Register Diagram

Table 7-35: A2B_ERRCNT3 Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
7:0 (R/NW)	ERRCNT	Error Count for PRBS. The A2B_ERRCNT3.ERRCNT bit field contains one byte of the 32-bit PRBS bit error count.

Node Register

The A2B_NODE register contains information required for node-to-node communication.

Address: 0x29

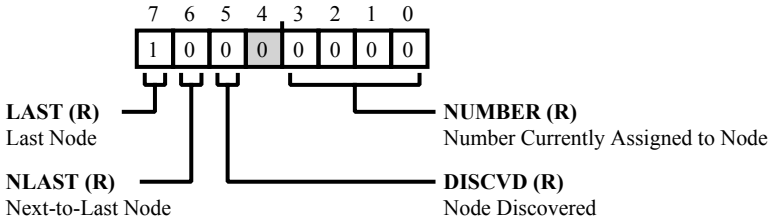

Figure 7-35: A2B_NODE Register Diagram

Table 7-36: A2B_NODE Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
7 (R/NW)	LAST	Last Node. The A2B_NODE.LAST bit indicates that this node is not connected to a downstream node. It is set by default at reset and cleared during discovery.	
		0	Not Last Node
		1	Last Node
6 (R/NW)	NLAST	Next-to-Last Node. The A2B_NODE.NLAST bit indicates that this node is directly upstream of the last node. It is set during discovery.	
		0	Not Next-to-Last Node
		1	Next-to-Last Node
5 (R/NW)	DISCVD	Node Discovered. The A2B_NODE.DISCVD bit indicates that this node has been discovered. This bit always reads as 0 in a master node.	
		0	Not Discovered
		1	Discovered
3:0 (R/NW)	NUMBER	Number Currently Assigned to Node. The A2B_NODE.NUMBER bit field reports the node number assigned to the node during discovery. This field always reads as 0 in a master node.	

Discovery Status Register (Master Only)

The A2B_DISCSTAT register provides status for discovery transactions on the A²B bus. An I²C write to the A2B_DISCVRY register sets the A2B_DISCSTAT.DSCACT bit and causes the A2B_NODEADR.NODE field to be written to this register. Discovery mode can be aborted by writing to the A2B_CONTROL.ENDDSC bit.

Address: 0x2B

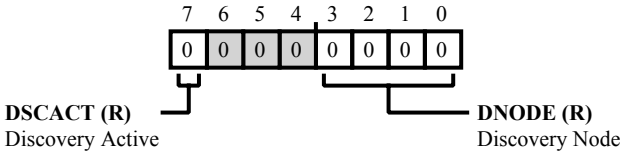

Figure 7-36: A2B_DISCSTAT Register Diagram

Table 7-37: A2B_DISCSTAT Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
7 (R/NW)	DSCACT	Discovery Active. The A2B_DISCSTAT.DSCACT bit is set while the master node is in discovery mode.
3:0 (R/NW)	DNODE	Discovery Node. When the A2B_DISCSTAT.DSCACT bit is set, the A2B_DISCSTAT.DNODE bit field shows the node being used for discovery frames. If A2B_DISCSTAT.DSCACT is cleared, the A2B_DISCSTAT.DNODE bit field retains the value of the last node discovered.

LVDSA TX Control Register

The A2B_TXACTL register provides transmitter control for LVDS transceiver A. The values in this register are only applied when the A2B_TXACTL.TXAOVREN bit is set.

Address: 0x2E

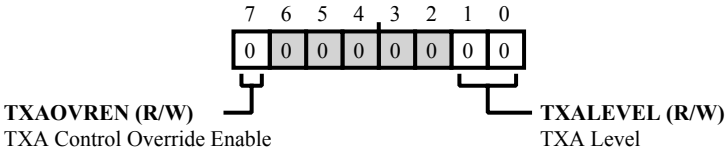

Figure 7-37: A2B_TXACTL Register Diagram

Table 7-38: A2B_TXACTL Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
7 (R/W)	TXAOVREN	TXA Control Override Enable. The A2B_TXACTL.TXAOVREN bit is used to force values from the A2B_TXACTL register to override the default values.	
1:0 (R/W)	TXALEVEL	TXA Level. The A2B_TXACTL.TXALEVEL bit field determines the transmitter output signal levels.	
		0	High Transmit Power Level
		1	Reserved
		2	Medium Transmit Power Level
		3	Low Transmit Power Level

LVDSB TX Control Register

The A2B_TXBCTL register provides transmitter control for LVDS transceiver B. The values in this register are only applied when the A2B_TXBCTL.TXBOVREN bit is set.

Address: 0x30

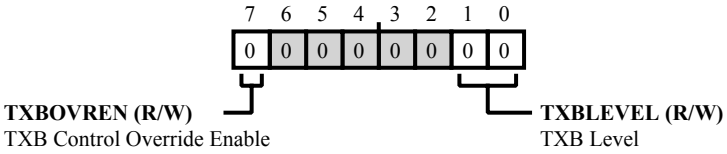

Figure 7-38: A2B_TXBCTL Register Diagram

Table 7-39: A2B_TXBCTL Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
7 (R/W)	TXBOVREN	TXB Control Override Enable. The A2B_TXBCTL.TXBOVREN bit is used to force values from the A2B_TXBCTL register to override the default values.	
1:0 (R/W)	TXBLEVEL	TXB Level. The A2B_TXBCTL.TXBLEVEL bit field determines the transmitter output signal levels.	
		0	High Transmit Power Level
		1	Reserved
		2	Medium Transmit Power Level
		3	Low Transmit Power Level

Local Interrupt Type (Slave Only)

Address: 0x3E

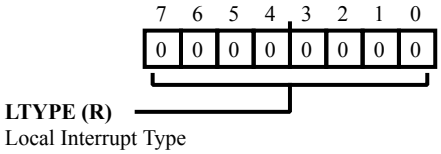

Figure 7-39: A2B_LINTTYPE Register Diagram

Table 7-40: A2B_LINTTYPE Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
7:0 (R/NW)	LTYPE	Local Interrupt Type.	
		48	Mailbox 0 Full
		49	Mailbox 0 Empty
		50	Mailbox 1 Full
		51	Mailbox 1 Empty

I2C Configuration Register

The A2B_I2CCFG register controls the data rate of the I²C port in A²B slave nodes and sets the I²C behavior in the A²B master node.

Address: 0x3F

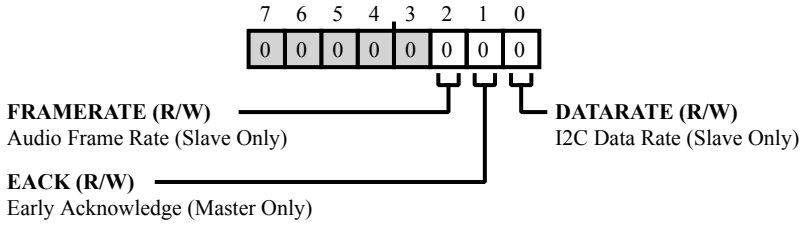

Figure 7-40: A2B_I2CCFG Register Diagram

Table 7-41: A2B_I2CCFG Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
2 (R/W)	FRAMERATE	Audio Frame Rate (Slave Only). The A2B_I2CCFG.FRAMERATE bit defaults to 48 kHz. This bit only affects the local clock generation for the I²C interface to match standard I²C clock speeds.	
		0	48 kHz
		1	44.1 kHz
1 (R/W)	EACK	Early Acknowledge (Master Only). When A2B_I2CCFG.EACK is set, the I²C interface provides an acknowledge to writes addressed to a slave node before the write has completed on the A²B bus. If there is an error (for example, a timeout or address error), the A2B_INTPND2.I2CERR bit is set. When A2B_I2CCFG.EACK is cleared, I²C transactions are clock-stretched until they are complete in the system so that a correct ACK/NACK can be generated by the I²C interface. The A2B_I2CCFG.EACK bit can be used for I²C access of a slave node. For accesses to peripherals connected to slave nodes, the clock stretching feature is required for the I²C interface of the host.	
		0	Stretch Transactions
		1	Provide Write Acknowledge
0 (R/W)	DATARATE	I2C Data Rate (Slave Only). The A2B_I2CCFG.DATARATE bit configures the I²C data rate.	
		0	100 kHz
		1	400 kHz

PLL Control Register

The A2B_PLLCTL register provides control bits for the PLL.

Address: 0x40

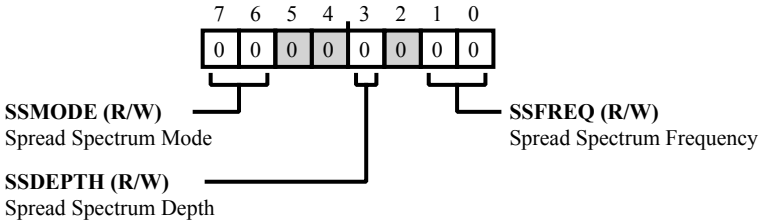

Figure 7-41: A2B_PLLCTL Register Diagram

Table 7-42: A2B_PLLCTL Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
7:6 (R/W)	SSMODE	Spread Spectrum Mode. The A2B_PLLCTL.SSMODE bit field selects the spread spectrum mode. Spread-spectrum clocking support can be enabled for the internal clocks, the I²S interfaces, and the programmed CLKOUTs.	
		0	No Spread
		1	Spread A²B Bus Clocks Only
		2	Spread A²B Bus Clocks and I²S Clocks
		3	Reserved
3 (R/W)	SSDEPTH	Spread Spectrum Depth. The A2B_PLLCTL.SSDEPTH bit determines the spread spectrum depth of modulation.	
		0	Low Spread Spectrum Depth of Modulation
		1	High Spread Spectrum Depth of Modulation
1:0 (R/W)	SSFREQ	Spread Spectrum Frequency. The A2B_PLLCTL.SSFREQ bit determines the frequency modulation (multiples of f_{SYNCM}).	
		0	4x
		1	5x
		2	6x
		3	7x

I2S Global Configuration Register

The A2B_I2SGCFG register provides bits which control the operation of all I²S units. The A2B_I2SGCFG register must be programmed before any of the A2B_I2SCFG.TX0EN, A2B_I2SCFG.TX1EN, A2B_I2SCFG.RX0EN, A2B_I2SCFG.RX1EN, A2B_PDMCTL.PDM0EN, and A2B_PDMCTL.PDM1EN bits are set.

For the master node, the A2B_I2SGCFG register must be programmed before discovery and must not be modified after discovery.

Address: 0x41

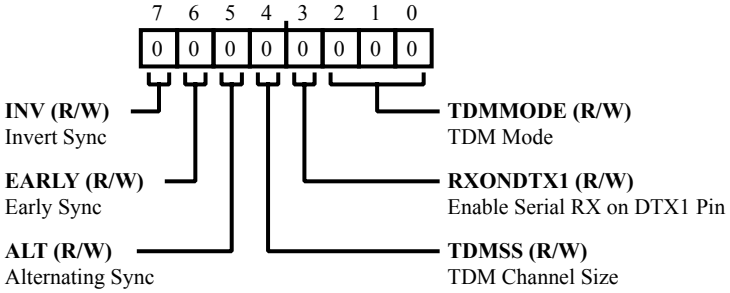

Figure 7-42: A2B_I2SGCFG Register Diagram

Table 7-43: A2B_I2SGCFG Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
7 (R/W)	INV	Invert Sync. The A2B_I2SGCFG.INV bit determines whether the rising edge or the falling edge of the A2B_SYNC pin corresponds to the start of an audio frame. If the A2B_I2SGCFG.INV bit is to be set in a master node, it must be set before the A2B_SWCTL.ENSW bit is set.	
		0	Rising edge of SYNC pin at start of audio frame
		1	Falling edge of SYNC pin at start of audio frame
6 (R/W)	EARLY	Early Sync. The A2B_I2SGCFG.EARLY bit determines whether the A2B_SYNC pin changes in the same cycle as the MSB of data slot 0 or one cycle before the MSB of data slot 0.	
		0	Change SYNC pin in same cycle
		1	Change SYNC pin in previous cycle

Table 7-43: A2B_I2SGCFG Register Fields (Continued)

Bit No. (Access)	Bit Name	Description/Enumeration	
5 (R/W)	ALT	Alternating Sync. The `A2B_I2SGCFG.ALT` bit determines whether the `A2B_SYNC` pin is pulsed high for one cycle at the start of each sampling period or driven high during right channel data and low during left channel data for I^2S stereo mode operation.	
		0	Pulse SYNC Pin High for 1 Cycle
		1	Drive SYNC Pin for I^2S Operation
4 (R/W)	TDMSS	TDM Channel Size. The `A2B_I2SGCFG.TDMSS` bit determines whether the slot size is 16 or 32 bits.	
		0	32-Bit
		1	16-Bit
3 (R/W)	RXONDTX1	Enable Serial RX on DTX1 Pin. When the `A2B_I2SGCFG.RXONDTX1` bit is set, the DTX1 pin is used for I^2S/TDM RX in place of the DRX1 pin, and the values of `A2B_I2SCFG.TX1EN` and `A2B_I2SCFG.RX1EN` are ignored.	
2:0 (R/W)	TDMMODE	TDM Mode. The `A2B_I2SGCFG.TDMMODE` bit field selects the mode for the I^2S/TDM units.	
		0	TDM2
		1	TDM4
		2	TDM8
		3	TDM12 (No slave node support)
		4	TDM16
		5	TDM20 (No slave node support)
		6	TDM24 (No slave node support)
		7	TDM32 (Single data pin support only)

I2S Configuration Register

The A2B_I2SCFG register provides control over which I²S data pins are enabled, how the data associated with them is stored in the internal frame buffers, and the polarity of the BCLK signal.

IMPORTANT: If both A2B_I2SCFG.RX1EN and A2B_I2SCFG.RX0EN are set and both A2B_PDMCTL.PDM1EN and A2B_PDMCTL.PDM0EN are cleared, the received I²S data from the A2B_DRX1 and A2B_DRX0 pins is written into the A²B frame buffer independently.

IMPORTANT: If both A2B_I2SCFG.TX1EN and A2B_I2SCFG.TX0EN are set, the I²S transmit data for both the A2B_DTX1 and A2B_DTX0 pins is read from the A²B frame buffer independently.

Address: 0x42

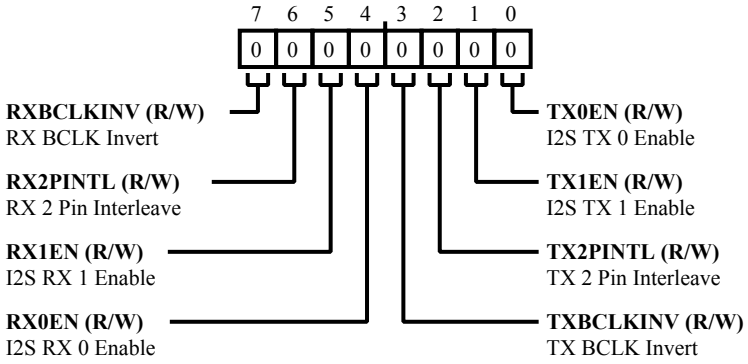

Figure 7-43: A2B_I2SCFG Register Diagram

Table 7-44: A2B_I2SCFG Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
7 (R/W)	RXBCLKINV	RX BCLK Invert. The A2B_I2SCFG.RXBCLKINV bit controls the BCLK edge that the A2B_DRX0 and A2B_DRX1 pins are sampled on. For master nodes only, this is also the sampling edge for the A2B_SYNC pin.	
		0	Sample on rising edge of BCLK
		1	Sample on falling edge of BCLK

Table 7-44: A2B_I2SCFG Register Fields (Continued)

Bit No. (Access)	Bit Name	Description/Enumeration	
6 (R/W)	RX2PINTL	RX 2 Pin Interleave. The A2B_I2SCFG.RX2PINTL bit is only used when TDM data is received on two pins simultaneously. When this bit is cleared (default), the data received on the A2B_DRX0 pin is associated with the lower half of the bus data slots, and the data received on the A2B_DRX1 pin is associated with the upper half of the bus data slots. When the A2B_I2SCFG.RX2PINTL bit is set, the data received on the A2B_DRX0 pin is associated with the even bus data slots (slot 0, slot 2, ..., slot 30), and the data received on the A2B_DRX1 pin is associated with the odd bus data slots (slot 1, slot 3, ..., slot 31).	
		0	No Interleaving
		1	Interleaving
5 (R/W)	RX1EN	I2S RX 1 Enable. The A2B_I2SCFG.RX1EN bit enables I²S/TDM receive data on the A2B_DRX1 pin. This bit has no effect if the A2B_PDMCTL.PDM1EN bit is set.	
		0	Disabled
		1	Enabled
4 (R/W)	RX0EN	I2S RX 0 Enable. The A2B_I2SCFG.RX0EN bit enables I²S/TDM receive data on the A2B_DRX0 pin. This bit has no effect if the A2B_PDMCTL.PDM0EN bit is set.	
		0	Disabled
		1	Enabled
3 (R/W)	TXBCLKINV	TX BCLK Invert. The A2B_I2SCFG.TXBCLKINV bit controls the BCLK edge that the A2B_DTX0 and A2B_DTX1 pins are driven on. For slave nodes only, this is also the driving edge for the A2B_SYNC pin.	
		0	Drive on Rising Edge of BCLK
		1	Drive on Falling Edge of BCLK

Table 7-44: A2B_I2SCFG Register Fields (Continued)

Bit No. (Access)	Bit Name	Description/Enumeration	
2 (R/W)	TX2PINTL	TX 2 Pin Interleave. The A2B_I2SCFG.TX2PINTL bit is only used when TDM data is transmitted on two pins simultaneously. When this bit is cleared (default), the data transmitted on the A2B_DTX0 pin is associated with the lower half of the bus data slots, and the data transmitted on the A2B_DTX1 pin is associated with the upper half of the bus data slots. When the A2B_I2SCFG.TX2PINTL bit is set, the even bus data slots (slot 0, slot 2, ..., slot 30) transmitted on the A2B_DTX0 pin, and the odd bus data slots (slot 1, slot 3, ..., slot 31) are transmitted on the A2B_DTX1 pin.	
		0	Disabled
		1	Enabled
1 (R/W)	TX1EN	I2S TX 1 Enable. The A2B_I2SCFG.TX1EN bit enables I^{2}S/TDM transmit data on the A2B_DTX1 pin.	
		0	Disabled
		1	Enabled
0 (R/W)	TX0EN	I2S TX 0 Enable. The A2B_I2SCFG.TX0EN bit enables I^{2}S/TDM transmit data on the A2B_DTX0 pin.	
		0	Disabled
		1	Enabled

I2S Rate Register (Slave Only)

The A2B_I2SRATE register controls the I²S/TDM interfaces in slave nodes, which may run at a multiple of the superframe rate.

Address: 0x43

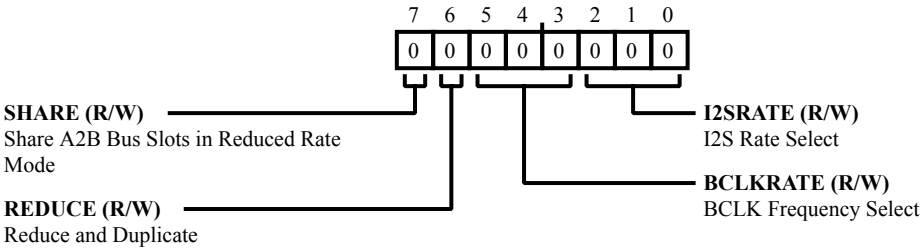

Figure 7-44: A2B_I2SRATE Register Diagram

Table 7-45: A2B_I2SRATE Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
7 (R/W)	SHARE	Share A2B Bus Slots in Reduced Rate Mode. The A2B_I2SRATE.SHARE bit function applies only when the local sample rate is lower than the superframe rate. When the A2B_I2SRATE.SHARE bit is set, I²S/TDM data for the local node is time multiplexed on the A²B bus. Only A2B_I2SRRSOFFS.RRSOFFSET values of 0 or 1 are supported when the A2B_I2SRATE.SHARE bit is enabled.	
		0	Disabled
		1	Enabled
6 (R/W)	REDUCE	Reduce and Duplicate. The A2B_I2SRATE.REDUCE bit function applies only when the local sample rate is higher than the superframe rate. When the A2B_I2SRATE.REDUCE bit is set, the number of received samples is reduced so that only one sample is used per superframe, and transmitted samples are duplicated so that only one sample is needed per superframe.	
		0	Disabled
		1	Enabled
5:3 (R/W)	BCLKRATE	BCLK Frequency Select. The A2B_I2SRATE.BCLKRATE bit field is used to select an alternate BCLK frequency for a reduced rate slave node. The nominal BCLK frequency is determined by the superframe frequency (SFF), settings, and reduced rate divide ratio (from A2B_I2SRRATE.RRDIV and A2B_I2SRATE.I2SRATE).	
		0	BCLK frequency as configured in A2B_I2SGCFG
		1	SYNC frequency x 2048

Table 7-45: A2B_I2SRATE Register Fields (Continued)

Bit No. (Access)	Bit Name	Description/Enumeration	
		2	SYNC frequency x 4096
		4	SFF frequency x 64
		5	SFF frequency x 128
		6	SFF frequency x 256
2:0 (R/W)	I2SRATE	I2S Rate Select. The `A2B_I2SRATE.I2SRATE` bit sets the rate for I^2S/TDM transmit and receive operations in the local slave node. This sample rate is based on the superframe frequency (SFF is either 48 kHz or 44.1 kHz).	
		0	SFF x 1
		1	SFF / 2
		2	SFF / 4
		3	SFF / `A2B_I2SRRATE.RRDIV`
		5	SFF x 2
		6	SFF x 4

I2S Transmit Data Offset Register (Master Only)

The A2B_I2STXOFFSET register controls the number of I²S transmit channels which are skipped before the node begins transmitting data.

Address: 0x44

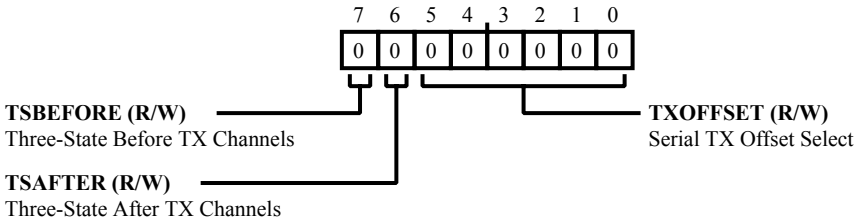

Figure 7-45: A2B_I2STXOFFSET Register Diagram

Table 7-46: A2B_I2STXOFFSET Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
7 (R/W)	TSBEFORE	Three-State Before TX Channels. When the A2B_I2STXOFFSET.TSBEFORE bit is cleared (default), the A2B_DTX0 and A2B_DTX1 pins are driven low at the beginning of each frame for the number of data channels defined in A2B_I2STXOFFSET.TXOFFSET. When this bit is set, the A2B_DTX0 and A2B_DTX1 pins are instead three-stated for the number of data channels defined in A2B_I2STXOFFSET.TXOFFSET.	
		0	Disable
		1	Enable
6 (R/W)	TSAFTER	Three-State After TX Channels. When the A2B_I2STXOFFSET.TSAFTER bit is cleared (default), the A2B_DTX0 and A2B_DTX1 pins are driven low after all valid channels have been transmitted. When the A2B_I2STXOFFSET.TSAFTER bit is set, the A2B_DTX0 and A2B_DTX1 pins are instead three-stated after all valid channels have been transmitted.	
		0	Disable
		1	Enable

Table 7-46: A2B_I2STXOFFSET Register Fields (Continued)

Bit No. (Access)	Bit Name	Description/Enumeration	
5:0 (R/W)	TXOFFSET	Serial TX Offset Select. The `A2B_I2STXOFFSET.TXOFFSET` bit field defines the number of I²S/TDM channels that are skipped before the node begins transmitting data. The valid values for this field are 0-63.	
		0	No TX offset
		1	1 TDM channel
		62	62 TDM channels
		63	63 TDM channels

I2S Receive Data Offset Register (Master Only)

The A2B_I2SRXOFFSET register controls the number of I²S receive channels which are skipped before the node begins receiving data.

Address: 0x45

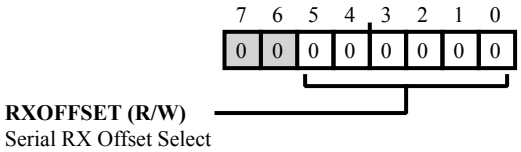

Figure 7-46: A2B_I2SRXOFFSET Register Diagram

Table 7-47: A2B_I2SRXOFFSET Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
5:0 (R/W)	RXOFFSET	Serial RX Offset Select. The A2B_I2SRXOFFSET.RXOFFSET bit field defines the number of I²S/TDM channels that are skipped before the node begins receiving data. The valid values for this field are 0-63.	
		0	No RX offset
		62	62 TDM channels
		63	63 TDM channels

SYNC Offset Register (Slave Only)

The A2B_SYNCOFFSET register adjusts the A^{2}B bus clock ($f_{SYSBCLK}$) cycle count on which the A2B_SYNC pin indicates the start of an audio frame. A^{2}B slave nodes can all sample exactly at the same time by individually compensating for their propagation delay with this register setting.

The A2B_SYNCOFFSET register must be programmed before any of the data pin enable bits are set (A2B_I2SCFG.TX0EN, A2B_I2SCFG.TX1EN, A2B_I2SCFG.RX0EN, A2B_I2SCFG.RX1EN, A2B_PDMCTL.PDM0EN, or A2B_PDMCTL.PDM1EN).

Address: 0x46

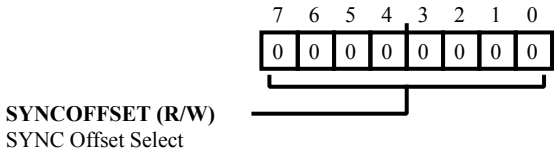

Figure 7-47: A2B_SYNCOFFSET Register Diagram

Table 7-48: A2B_SYNCOFFSET Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
7:0 (R/W)	SYNCOFFSET	SYNC Offset Select. The A2B_SYNCOFFSET.SYNCOFFSET bit field adjusts the system clock cycle where the A2B_SYNC pin indicates the start of an audio frame. The value programmed to the A2B_SYNCOFFSET.SYNCOFFSET field is the 8-bit signed, two's complement representation of the integer value defining the number of SYSBCLK cycles that lag the SYNC signal before the superframe begins. Valid values for this field range from no SYNC offset (0x00) to the SYNC occurring 127 cycles before the start of the superframe (0x81).	
		0	No offset
		1-128	Reserved
		129	127 SYSBCLK cycles
		130-254	126 to 2 SYSBCLK cycles (respectively)
		255	1 SYSBCLK cycle

PDM Control Register

The A2B_PDMCTL register provides enable bits for the pulse density modulators.

Address: 0x47

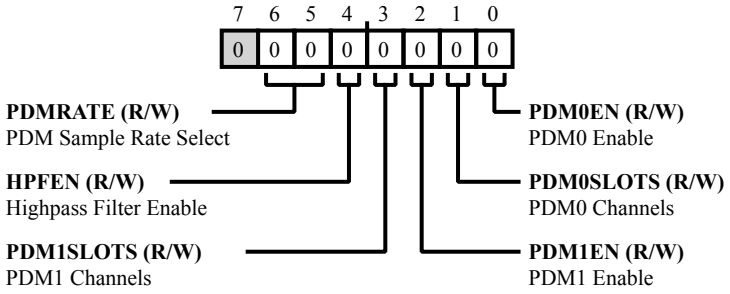

Figure 7-48: A2B_PDMCTL Register Diagram

Table 7-49: A2B_PDMCTL Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
6:5 (R/W)	PDMRATE	PDM Sample Rate Select. The A2B_PDMCTL.PDMRATE bit field controls the output rate of the PDM demodulators, which is based off of the superframe rate (SFF). Changes to the A2B_PDMCTL.PDMRATE field do not change the PDM clock frequency. For a slave node, setting the node to a reduced rate changes the SYNC and PDM clock frequencies. Setting the slave node to an increased rate changes only the SYNC. The PDM clock frequency stays at 3.07MHz.	
		0	SFF
		1	SFF/2
		2	SFF/4
		3	Reserved
4 (R/W)	HPFEN	Highpass Filter Enable. The A2B_PDMCTL.HPFEN bit controls whether or not the high pass filter is used on received PDM data.	
		0	Disabled
		1	Enabled
3 (R/W)	PDM1SLOTS	PDM1 Channels. The A2B_PDMCTL.PDM1SLOTS bit controls whether the PDM signal on the A2B_DRX1 pin is one channel (mono) or two channels (stereo).	
		0	Mono
		1	Stereo

Table 7-49: A2B_PDMCTL Register Fields (Continued)

Bit No. (Access)	Bit Name	Description/Enumeration	
2 (R/W)	PDM1EN	PDM1 Enable. The `A2B_PDMCTL.PDM1EN` bit enables PDM reception on the `A2B_DRX1`/ `A2B_IO6` pin.	
		0	Disabled
		1	Enabled
1 (R/W)	PDM0SLOTS	PDM0 Channels. The `A2B_PDMCTL.PDM0SLOTS` bit controls whether the PDM signal on the A2B_DRX0 pin is one channel (mono) or two channels (stereo).	
		0	Mono
		1	Stereo
0 (R/W)	PDM0EN	PDM0 Enable. The `A2B_PDMCTL.PDM0EN` bit enables PDM reception on the `A2B_DRX0`/ `A2B_IO5` pin.	
		0	Disabled
		1	Enabled

Error Management Register

The A2B_ERRMGMT register provides options for reporting communication errors over the I²S/TDM interface.

Address: 0x48

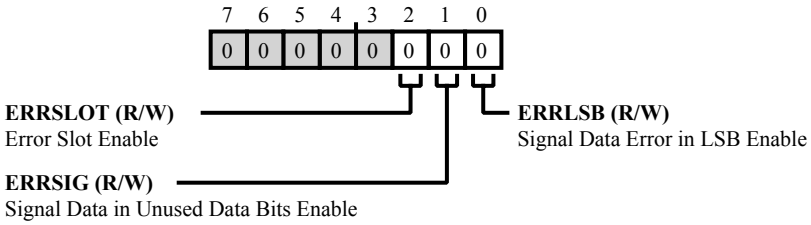

Figure 7-49: A2B_ERRMGMT Register Diagram

Table 7-50: A2B_ERRMGMT Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
2 (R/W)	ERRSLOT	Error Slot Enable. Setting the A2B_ERRMGMT.ERRSLOT bit causes the transceiver to append an extra I²S/TDM data channel to the TDM stream to indicate A²B errors in the received data slots.	
		0	Disabled
		1	Enabled
1 (R/W)	ERRSIG	Signal Data in Unused Data Bits Enable. When the A2B_ERRMGMT.ERRSIG is set, any unused data bits in each I²S/TDM channel indicate data errors.	
		0	Disabled
		1	Enabled
0 (R/W)	ERRLSB	Signal Data Error in LSB Enable. When the A2B_ERRMGMT.ERRLSB bit is set, the LSB of each I²S/TDM sample is replaced with an active-high status bit indicating that there is an error in the data slot (1= error, 0 = no error).	
		0	Disabled
		1	Enabled

GPIO Output Data Register

The A2B_GPIODAT register controls output data for general-purpose I/O pins.

Address: 0x4A

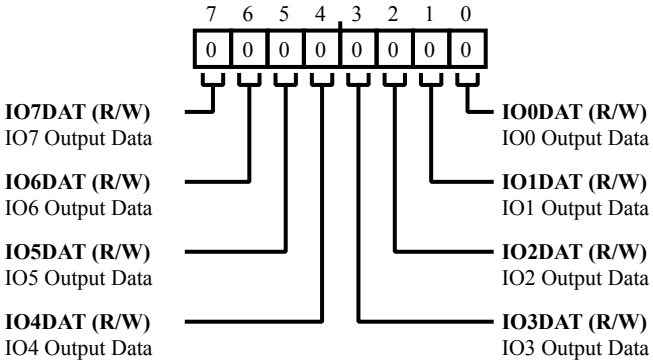

Figure 7-50: A2B_GPIODAT Register Diagram

Table 7-51: A2B_GPIODAT Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
7 (R/W)	IO7DAT	IO7 Output Data. The value of the A2B_GPIODAT.IO7DAT bit is driven onto the IO7 pin when it is in GPIO mode with its output driver enabled (A2B_GPIOOEN.IO7OEN=1).	
		0	Output Low
		1	Output High
6 (R/W)	IO6DAT	IO6 Output Data. The value of the A2B_GPIODAT.IO6DAT bit is driven onto the IO6 pin when it is in GPIO mode with its output driver enabled (A2B_GPIOOEN.IO6OEN=1).	
		0	Output Low
		1	Output High
5 (R/W)	IO5DAT	IO5 Output Data. The value of the A2B_GPIODAT.IO5DAT bit is driven onto the IO5 pin when it is in GPIO mode with its output driver enabled (A2B_GPIOOEN.IO5OEN=1).	
		0	Output Low
		1	Output High

Table 7-51: A2B_GPIODAT Register Fields (Continued)

Bit No. (Access)	Bit Name	Description/Enumeration	
4 (R/W)	IO4DAT	IO4 Output Data. The value of the A2B_GPIODAT.IO4DAT bit is driven onto the IO4 pin when it is in GPIO mode with its output driver enabled (A2B_GPIOOEN.IO4OEN=1).	
		0	Output Low
		1	Output High
3 (R/W)	IO3DAT	IO3 Output Data. The value of the A2B_GPIODAT.IO3DAT bit is driven onto the IO3 pin when it is in GPIO mode with its output driver enabled (A2B_GPIOOEN.IO3OEN=1).	
		0	Output Low
		1	Output High
2 (R/W)	IO2DAT	IO2 Output Data. The value of the A2B_GPIODAT.IO2DAT bit is driven onto the IO2 pin when it is in GPIO mode with its output driver enabled (A2B_GPIOOEN.IO2OEN=1).	
		0	Output Low
		1	Output High
1 (R/W)	IO1DAT	IO1 Output Data. The value of the A2B_GPIODAT.IO1DAT bit is driven onto the IO1 pin when it is in GPIO mode with its output driver enabled (A2B_GPIOOEN.IO1OEN=1).	
		0	Output Low
		1	Output High
0 (R/W)	IO0DAT	IO0 Output Data. The value of the A2B_GPIODAT.IO0DAT bit is driven onto the IO0 pin when it is in GPIO mode with its output driver enabled (A2B_GPIOOEN.IO0OEN=1).	
		0	Output Low
		1	Output High

GPIO Output Data Set Register

The A2B_GPIODATSET register allows setting of individual GPIO output register bits (write 1 to set) without influencing the states of the other GPIO output register bits. Reads from this address return the value in the GPIO output data (A2B_GPIODAT) register.

Address: 0x4B

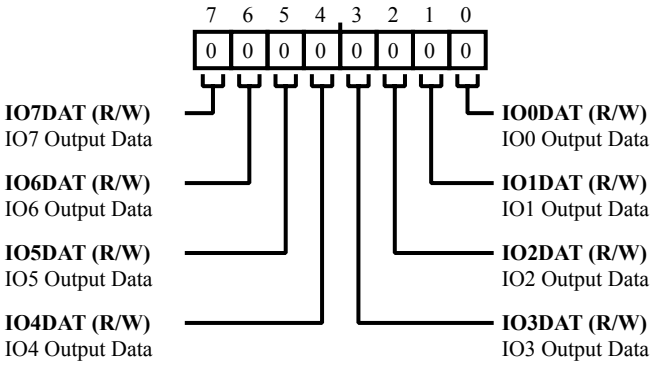

Figure 7-51: A2B_GPIODATSET Register Diagram

Table 7-52: A2B_GPIODATSET Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
7 (R/W1S)	IO7DSET	IO7 Data Set. The A2B_GPIODATSET.IO7DSET bit executes a write-1-to-set action for the A2B_GPIODAT.IO7DAT bit.	
		0	No Action
		1	Set Bit
6 (R/W1S)	IO6DSET	IO6 Data Set. The A2B_GPIODATSET.IO6DSET bit executes a write-1-to-set action for the A2B_GPIODAT.IO6DAT bit.	
		0	No Action
		1	Set Bit
5 (R/W1S)	IO5DSET	IO5 Data Set. The A2B_GPIODATSET.IO5DSET bit executes a write-1-to-set action for the A2B_GPIODAT.IO5DAT bit.	
		0	No Action
		1	Set Bit

Table 7-52: A2B_GPIODATSET Register Fields (Continued)

Bit No. (Access)	Bit Name	Description/Enumeration	
4 (R/W1S)	IO4DSET	IO4 Data Set. The `A2B_GPIODATSET.IO4DSET` bit executes a write-1-to-set action for the `A2B_GPIODAT.IO4DAT` bit.	
		0	No Action
		1	Set Bit
3 (R/W1S)	IO3DSET	IO3 Data Set. The `A2B_GPIODATSET.IO3DSET` bit executes a write-1-to-set action for the `A2B_GPIODAT.IO3DAT` bit.	
		0	No Action
		1	Set Bit
2 (R/W1S)	IO2DSET	IO2 Data Set. The `A2B_GPIODATSET.IO2DSET` bit executes a write-1-to-set action for the `A2B_GPIODAT.IO2DAT` bit.	
		0	No Action
		1	Set Bit
1 (R/W1S)	IO1DSET	IO1 Data Set. The `A2B_GPIODATSET.IO1DSET` bit executes a write-1-to-set action for the `A2B_GPIODAT.IO1DAT` bit.	
		0	No Action
		1	Set Bit
0 (R/W1S)	IO0DSET	IO0 Data Set. The `A2B_GPIODATSET.IO0DSET` bit executes a write-1-to-set action for the `A2B_GPIODAT.IO0DAT` bit.	
		0	No Action
		1	Set Bit

GPIO Output Data Clear Register

The A2B_GPIODATCLR register allows clearing of individual GPIO output register bits to 0 (write 1 to clear) without influencing the states of the other GPIO output register bits. Reads from this address return the value in the GPIO output data (A2B_GPIODAT) register.

Address: 0x4C

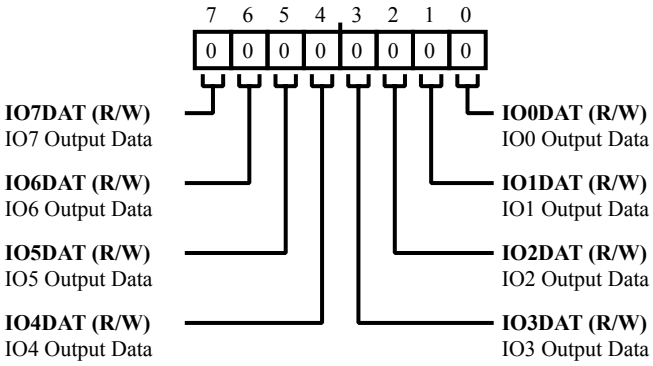

Figure 7-52: A2B_GPIODATCLR Register Diagram

Table 7-53: A2B_GPIODATCLR Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
7 (R/W1C)	IO7DCLR	IO7 Data Clear. The A2B_GPIODATCLR.IO7DCLR bit executes a write-1-to-clear action for the A2B_GPIODAT.IO7DAT bit.	
		0	No Action
		1	Clear Bit
6 (R/W1C)	IO6DCLR	IO6 Data Clear. The A2B_GPIODATCLR.IO6DCLR bit executes a write-1-to-clear action for the A2B_GPIODAT.IO6DAT bit.	
		0	No Action
		1	Clear Bit
5 (R/W1C)	IO5DCLR	IO5 Data Clear. The A2B_GPIODATCLR.IO5DCLR bit executes a write-1-to-clear action for the A2B_GPIODAT.IO5DAT bit.	
		0	No Action
		1	Clear Bit

Table 7-53: A2B_GPIODATCLR Register Fields (Continued)

Bit No. (Access)	Bit Name	Description/Enumeration	
4 (R/W1C)	IO4DCLR	IO4 Data Clear. The `A2B_GPIODATCLR.IO4DCLR` bit executes a write-1-to-clear action for the `A2B_GPIODAT.IO4DAT` bit.	
		0	No Action
		1	Clear Bit
3 (R/W1C)	IO3DCLR	IO3 Data Clear. The `A2B_GPIODATCLR.IO3DCLR` bit executes a write-1-to-clear action for the `A2B_GPIODAT.IO3DAT` bit.	
		0	No Action
		1	Clear Bit
2 (R/W1C)	IO2DCLR	IO2 Data Clear. The `A2B_GPIODATCLR.IO2DCLR` bit executes a write-1-to-clear action for the `A2B_GPIODAT.IO2DAT` bit.	
		0	No Action
		1	Clear Bit
1 (R/W1C)	IO1DCLR	IO1 Data Clear. The `A2B_GPIODATCLR.IO1DCLR` bit executes a write-1-to-clear action for the `A2B_GPIODAT.IO1DAT` bit.	
		0	No Action
		1	Clear Bit
0 (R/W1C)	IO0DCLR	IO0 Data Clear. The `A2B_GPIODATCLR.IO0DCLR` bit executes a write-1-to-clear action for the `A2B_GPIODAT.IO0DAT` bit. Only slave nodes can output data on this pin.	
		0	No Action
		1	Clear Bit

GPIO Output Enable Register

The A2B_GPIOOEN register controls the output enables of the general-purpose I/O pins.

Address: 0x4D

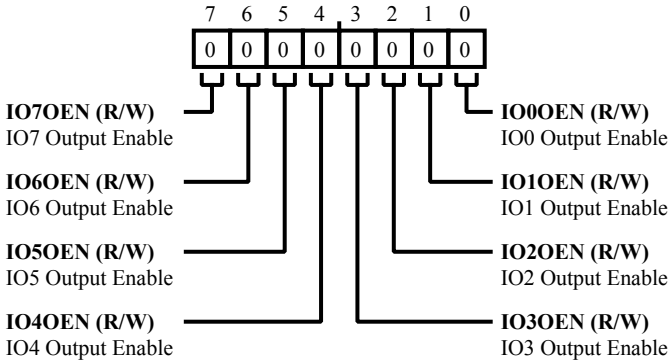

Figure 7-53: A2B_GPIOOEN Register Diagram

Table 7-54: A2B_GPIOOEN Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
7 (R/W)	IO7OEN	IO7 Output Enable. The A2B_GPIOOEN.IO7OEN bit configures the IO7 pin as an output when the pin is in GPIO mode.	
		0	Disable
		1	Enable
6 (R/W)	IO6OEN	IO6 Output Enable. The A2B_GPIOOEN.IO6OEN bit configures the DRX1/IO6 pin as an output when the pin is in GPIO mode.	
		0	Disable
		1	Enable
5 (R/W)	IO5OEN	IO5 Output Enable. The A2B_GPIOOEN.IO5OEN bit configures the DRX0/IO5 pin as an output when the pin is in GPIO mode.	
		0	Disable
		1	Enable

Table 7-54: A2B_GPIOOEN Register Fields (Continued)

Bit No. (Access)	Bit Name	Description/Enumeration	
4 (R/W)	IO4OEN	IO4 Output Enable. The `A2B_GPIOOEN.IO4OEN` bit configures the DTX1/IO4 pin as an output when the pin is in GPIO mode.	
		0	Disable
		1	Enable
3 (R/W)	IO3OEN	IO3 Output Enable. The `A2B_GPIOOEN.IO3OEN` bit configures the DTX0/IO3 pin as an output when the pin is in GPIO mode.	
		0	Disable
		1	Enable
2 (R/W)	IO2OEN	IO2 Output Enable. The `A2B_GPIOOEN.IO2OEN` bit configures the ADR2/IO2 pin as an output when the pin is in GPIO mode.	
		0	Disable
		1	Enable
1 (R/W)	IO1OEN	IO1 Output Enable. The `A2B_GPIOOEN.IO1OEN` bit configures the ADR1/IO1 pin as an output when the pin is in GPIO mode.	
		0	Disable
		1	Enable
0 (R/W)	IO0OEN	IO0 Output Enable. The `A2B_GPIOOEN.IO0OEN` bit configures the IRQ/IO0 pin as an output when the pin is in GPIO mode. The `A2B_GPIOOEN.IO0OEN` bit has no effect in a master node. Only slave nodes can output data on this pin.	
		0	Disable
		1	Enable

GPIO Input Enable Register

The A2B_GPIOIEN register controls the input enables of the general purpose I/O pins.

Address: 0x4E

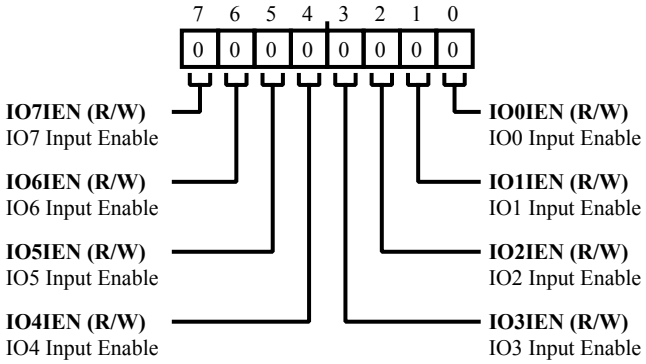

Figure 7-54: A2B_GPIOIEN Register Diagram

Table 7-55: A2B_GPIOIEN Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
7 (R/W)	IO7IEN	IO7 Input Enable. The A2B_GPIOIEN.IO7IEN bit is the input enable for the IO7 pin.	
		0	Disable
		1	Enable
6 (R/W)	IO6IEN	IO6 Input Enable. The A2B_GPIOIEN.IO6IEN bit is the input enable for the DRX1/IO6 pin.	
		0	Disable
		1	Enable
5 (R/W)	IO5IEN	IO5 Input Enable. The A2B_GPIOIEN.IO5IEN bit is the input enable for the DRX0/IO5 pin.	
		0	Disable
		1	Enable
4 (R/W)	IO4IEN	IO4 Input Enable. The A2B_GPIOIEN.IO4IEN bit is the input enable for the DTX1/IO4 pin.	
		0	Disable
		1	Enable

Table 7-55: A2B_GPIOIEN Register Fields (Continued)

Bit No. (Access)	Bit Name	Description/Enumeration	
3 (R/W)	IO3IEN	IO3 Input Enable. The `A2B_GPIOIEN.IO3IEN` bit is the input enable for the DTX0/IO3 pin.	
		0	Disable
		1	Enable
2 (R/W)	IO2IEN	IO2 Input Enable. The `A2B_GPIOIEN.IO2IEN` bit is the input enable for the ADR2/IO2 pin.	
		0	Disable
		1	Enable
1 (R/W)	IO1IEN	IO1 Input Enable. The `A2B_GPIOIEN.IO1IEN` bit is the input enable for the ADR1/IO1 pin.	
		0	Disable
		1	Enable
0 (R/W)	IO0IEN	IO0 Input Enable. The `A2B_GPIOIEN.IO0IEN` bit is the input enable for the IRQ/IO0 pin. This bit has no effect in a master node.	
		0	Disable
		1	Enable

GPIO Input Value Register

The A2B_GPIOIN register returns the value of enabled general-purpose I/O input pins.

Address: 0x4F

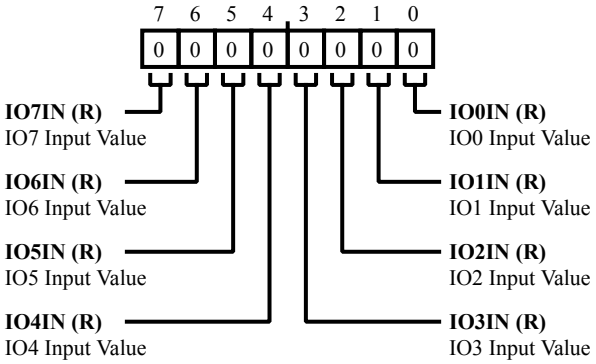

Figure 7-55: A2B_GPIOIN Register Diagram

Table 7-56: A2B_GPIOIN Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
7 (R/NW)	IO7IN	IO7 Input Value. The A2B_GPIOIN.IO7IN bit contains the value of the IO7 pin when in input GPIO mode (A2B_GPIOIEN.IO7IEN=1). Otherwise, the bit is low.
6 (R/NW)	IO6IN	IO6 Input Value. The A2B_GPIOIN.IO6IN bit contains the value of the DRX1/IO6 pin when in input GPIO mode (A2B_GPIOIEN.IO6IEN=1). Otherwise, the bit is low.
5 (R/NW)	IO5IN	IO5 Input Value. The A2B_GPIOIN.IO5IN bit contains the value of the DRX0/IO5 pin when in input GPIO mode (A2B_GPIOIEN.IO5IEN=1). Otherwise, the bit is low.
4 (R/NW)	IO4IN	IO4 Input Value. The A2B_GPIOIN.IO4IN bit contains the value of the DTX1/IO4 pin when in input GPIO mode (A2B_GPIOIEN.IO4IEN=1). Otherwise, the bit is low.
3 (R/NW)	IO3IN	IO3 Input Value. The A2B_GPIOIN.IO3IN bit contains the value of the DTX0/IO3 pin when in input GPIO mode (A2B_GPIOIEN.IO3IEN=1). Otherwise, the bit is low.
2 (R/NW)	IO2IN	IO2 Input Value. The A2B_GPIOIN.IO2IN bit contains the value of the ADR2/IO2 pin when in input GPIO mode (A2B_GPIOIEN.IO2IEN=1). Otherwise, the bit is low.

Table 7-56: A2B_GPIOIN Register Fields (Continued)

Bit No. (Access)	Bit Name	Description/Enumeration
1 (R/NW)	IO1IN	IO1 Input Value. The `A2B_GPIOIN.IO1IN` bit contains the value of the ADR1/IO1 pin when the `A2B_GPIOIEN.IO1IEN` bit is high. Otherwise the bit is low.
0 (R/NW)	IO0IN	IO0 Input Value. The `A2B_GPIOIN.IO0IN` bit contains the value of the IRQ/IO0 pin when in input GPIO mode (`A2B_GPIOIEN.IO0IEN=1`). Otherwise, the bit is low. This bit is only relevant in slave nodes and always reads 0 in a master node.

Pin Interrupt Enable Register

The A2B_PINTEN register enables input-enabled GPIO pins to generate an interrupt.

Address: 0x50

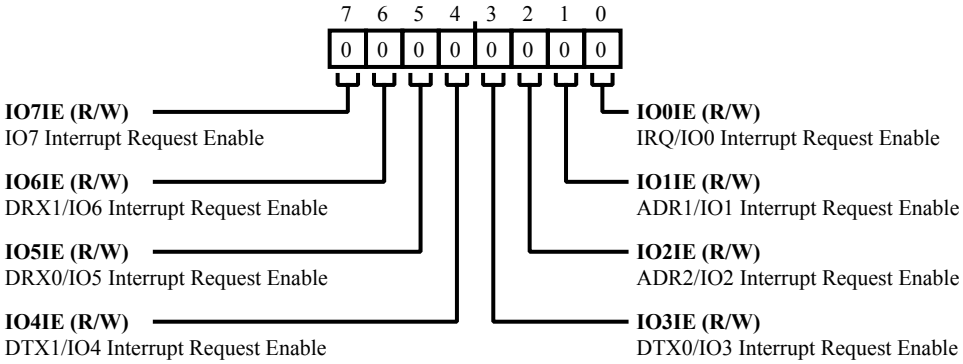

Figure 7-56: A2B_PINTEN Register Diagram

Table 7-57: A2B_PINTEN Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
7 (R/W)	IO7IE	IO7 Interrupt Request Enable. The A2B_PINTEN.IO7IE bit enables the IO7 input to generate an interrupt request when a rising edge is sensed.	
		0	Disabled
		1	Enabled
6 (R/W)	IO6IE	DRX1/IO6 Interrupt Request Enable. The A2B_PINTEN.IO6IE bit enables the DRX1/IO6 input to generate an interrupt request when a rising edge is sensed.	
		0	Disabled
		1	Enabled
5 (R/W)	IO5IE	DRX0/IO5 Interrupt Request Enable. The A2B_PINTEN.IO5IE bit enables the DRX0/IO5 input to generate an interrupt request when a rising edge is sensed.	
		0	Disabled
		1	Enabled

Table 7-57: A2B_PINTEN Register Fields (Continued)

Bit No. (Access)	Bit Name	Description/Enumeration	
4 (R/W)	IO4IE	DTX1/IO4 Interrupt Request Enable. The `A2B_PINTEN.IO4IE` bit enables the DTX1/IO4 input to generate an interrupt request when a rising edge is sensed.	
		0	Disabled
		1	Enabled
3 (R/W)	IO3IE	DTX0/IO3 Interrupt Request Enable. The `A2B_PINTEN.IO3IE` bit enables the DTX0/IO3 input to generate an interrupt request when a rising edge is sensed.	
		0	Disabled
		1	Enabled
2 (R/W)	IO2IE	ADR2/IO2 Interrupt Request Enable. The `A2B_PINTEN.IO2IE` bit enables the ADR2/IO2 input to generate an interrupt request when a rising edge is sensed.	
		0	Disabled
		1	Enabled
1 (R/W)	IO1IE	ADR1/IO1 Interrupt Request Enable. The `A2B_PINTEN.IO1IE` bit enables bit enables the IO1 input to generate an interrupt request when a rising edge is sensed.	
		0	Disabled
		1	Enabled
0 (R/W)	IO0IE	IRQ/IO0 Interrupt Request Enable. The `A2B_PINTEN.IO0IE` bit enables the IO0 input to generate an interrupt request when a rising edge is sensed. This bit has no effect in a master node.	
		0	Disabled
		1	Enabled

Pin Interrupt Invert Register

The A2B_PINTINV register is used to invert pin inputs in the path to interrupt generation.

Address: 0x51

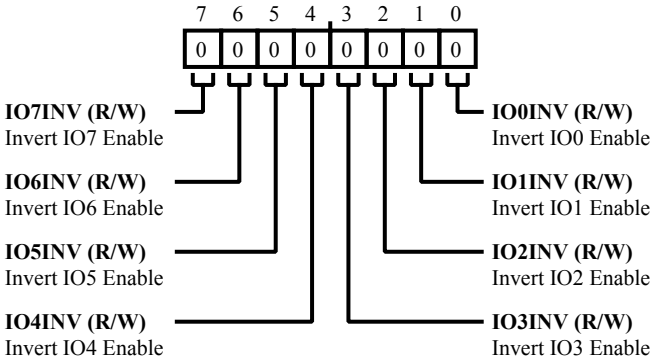

Figure 7-57: A2B_PINTINV Register Diagram

Table 7-58: A2B_PINTINV Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
7 (R/W)	IO7INV	Invert IO7 Enable. Setting the A2B_PINTINV.IO7INV bit inverts the polarity of the IO7 pin interrupt request input such that a falling edge sensed on the pin generates the interrupt rather than the rising edge (default).	
		0	Disabled
		1	Enabled
6 (R/W)	IO6INV	Invert IO6 Enable. Setting the A2B_PINTINV.IO6INV bit inverts the polarity of the DRX1/IO6 pin interrupt request input such that a falling edge sensed on the pin generates the interrupt rather than the rising edge (default).	
		0	Disabled
		1	Enabled
5 (R/W)	IO5INV	Invert IO5 Enable. Setting the A2B_PINTINV.IO5INV bit inverts the polarity of the DRX0/IO5 pin interrupt request input such that a falling edge sensed on the pin generates the interrupt rather than the rising edge (default).	
		0	Disabled
		1	Enabled

Table 7-58: A2B_PINTINV Register Fields (Continued)

Bit No. (Access)	Bit Name	Description/Enumeration	
4 (R/W)	IO4INV	Invert IO4 Enable. Setting the `A2B_PINTINV.IO4INV` bit inverts the polarity of the DTX1/IO4 pin interrupt request input such that a falling edge sensed on the pin generates the interrupt rather than the rising edge (default).	
		0	Disabled
		1	Enabled
3 (R/W)	IO3INV	Invert IO3 Enable. Setting the `A2B_PINTINV.IO3INV` bit inverts the polarity of the DTX0/IO3 pin interrupt request input such that a falling edge sensed on the pin generates the interrupt rather than the rising edge (default).	
		0	Disabled
		1	Enabled
2 (R/W)	IO2INV	Invert IO2 Enable. Setting the `A2B_PINTINV.IO2INV` bit inverts the polarity of the ADR2/IO2 pin interrupt request input such that a falling edge sensed on the pin generates the interrupt rather than the rising edge (default).	
		0	Disabled
		1	Enabled
1 (R/W)	IO1INV	Invert IO1 Enable. Setting the `A2B_PINTINV.IO1INV` bit inverts the polarity of the ADR1/IO1 pin interrupt request input such that a falling edge sensed on the pin generates the interrupt rather than the rising edge (default).	
		0	Disabled
		1	Enabled
0 (R/W)	IO0INV	Invert IO0 Enable. Setting the `A2B_PINTINV.IO0INV` bit inverts the polarity of the IRQ/IO0 pin interrupt request input such that a falling edge sensed on the pin generates the interrupt rather than the rising edge (default). This bit has no effect in a master node.	
		0	Disabled
		1	Enabled

Pin Configuration Register

The A2B_PINCFG register configures various digital pin characteristics.

Address: 0x52

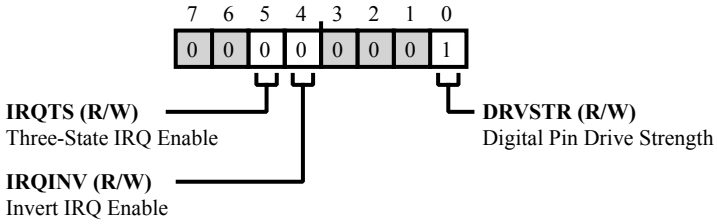

Figure 7-58: A2B_PINCFG Register Diagram

Table 7-59: A2B_PINCFG Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
5 (R/W)	IRQTS	Three-State IRQ Enable. When the A2B_PINCFG.IRQTS bit is cleared (default), the IRQ pin is always actively driven. Setting the A2B_PINCFG.IRQTS bit causes the transceiver to drive the IRQ pin when the interrupt is active and to three-state the IRQ pin when inactive.	
		0	Disabled
		1	Enabled
4 (R/W)	IRQINV	Invert IRQ Enable. When the A2B_PINCFG.IRQINV bit is cleared (default), the IRQ pin is active high. Setting the A2B_PINCFG.IRQINV bit makes the IRQ pin active low.	
		0	Disabled
		1	Enabled
0 (R/W)	DRVSTR	Digital Pin Drive Strength. The A2B_PINCFG.DRVSTR bit controls the drive strength of non-I^{2}C digital output pins. The A2B_SCL and A2B_SDA pins always have a high drive strength.	
		0	Low Drive Strength
		1	High Drive Strength

I2S Test Register

The A2B_I2STEST register enables a test mode to verify and debug the I²S/TDM interface.

Address: 0x53

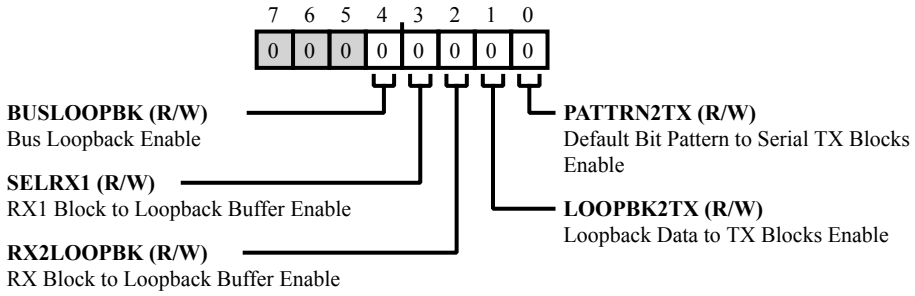

Figure 7-59: A2B_I2STEST Register Diagram

Table 7-60: A2B_I2STEST Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
4 (R/W)	BUSLOOPBK	Bus Loopback Enable. The A2B_I2STEST.BUSLOOPBK bit enables data loop back from the A2B_DTX0 pin to the A2B_DRX0 pin and from the A2B_DTX1 pin to the A2B_DRX1 pin. The A2B_I2STEST.LOOPBK2TX, A2B_I2STEST.RX2LOOPBK, and A2B_I2STEST.SELRX1 are ignored when this bit is set.	
		0	Disabled
		1	Enabled
3 (R/W)	SELRX1	RX1 Block to Loopback Buffer Enable. When the A2B_I2STEST.SELRX1 bit is cleared (default), the RX0 block is used for the loopback test when the A2B_I2STEST.RX2LOOPBK bit is set.When the A2B_I2STEST.SELRX1 bit is set, data from the DRX1 block is used instead.	
		0	Disabled
		1	Enabled
2 (R/W)	RX2LOOPBK	RX Block to Loopback Buffer Enable. When the A2B_I2STEST.RX2LOOPBK bit is set, the receive bit pattern on either the A2B_DRX0 or A2B_DRX1 pins (as controlled by the A2B_I2STEST.SELRX1 bit) is stored in the TX frame buffer. The A2B_I2SCFG.RX0EN, A2B_I2SCFG.RX1EN, and A2B_I2SCFG.RX2PINTL bits are ignored when this bit is set.	
		0	Disabled
		1	Enabled

Table 7-60: A2B_I2STEST Register Fields (Continued)

Bit No. (Access)	Bit Name	Description/Enumeration	
1 (R/W)	LOOPBK2TX	Loopback Data to TX Blocks Enable. When the A2B_I2STEST.LOOPBK2TX bit is set, data received on the A2B_DRX0 or A2B_DRX1 pin (as controlled by the A2B_I2STEST.SELRX1 bit) is sent out on the A2B_DTX0 and A2B_DTX1 pins. If the A2B_I2STEST.RX2LOOPBK bit is not set when the A2B_I2STEST.LOOPBK2TX bit is set, the default bit pattern is sent in all channels. If the A2B_I2STEST.RX2LOOPBK bit is cleared while the A2B_I2STEST.LOOPBK2TX bit is set, the last received frame is repeated. The A2B_I2SCFG.TX0EN, A2B_I2SCFG.TX1EN, and A2B_I2SCFG.TX2PINTL bits are ignored when this bit is set.	
		0	Disabled
		1	Enabled
0 (R/W)	PATTRN2TX	Default Bit Pattern to Serial TX Blocks Enable. When the A2B_I2STEST.PATTRN2TX bit is set, a default bit pattern (up to 32 bits) is sent in all channels on the A2B_DTX0 and A2B_DTX1 pins. The A2B_I2SCFG.TX0EN, A2B_I2SCFG.TX1EN, and A2B_I2SCFG.TX2PINTL bits are ignored when this bit is set. This bit is ignored when the A2B_I2STEST.LOOPBK2TX bit is set.	
		0	Disabled
		1	Enabled

Raise Interrupt Register

The A2B_RAISE register allows the host to generate an interrupt in any node in the system through software. This register must be written over the A^{2}B bus, as writes to this register from the local I^{2}C port have no effect.

Address: 0x54

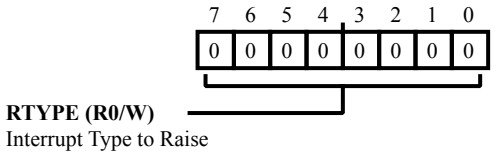

Figure 7-60: A2B_RAISE Register Diagram

Table 7-61: A2B_RAISE Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
7:0 (R0/W)	RTYPE	Interrupt Type to Raise. The A2B_RAISE.RTYPE bit field is written with the type of interrupt to raise. Any valid interrupt type may be generated in any node in the system. If the RTYPE field does not match a valid interrupt type for the node being written, no action will be taken.	
		0	HDCNTERR
		1	DDERR
		2	CRCERR
		3	DPERR
		4	BECOVF
		5	SRFERR
		6	SRFCRCERR
		9	PWRERR - Positive Terminal Shorted to Ground
		10	PWRERR - Negative Terminal Shorted to VBat
		11	PWRERR - Short of Wires
		12	PWRERR - Cable Disconnected or Open Circuit or Wrong Port
		13	PWRERR - Cable is Reverse Connected or Wrong Port
		15	PWRERR - Indeterminate Fault
		16	IO0PND - Slave Only
		17	IO1PND
		18	IO2PND

Table 7-61: A2B_RAISE Register Fields (Continued)

Bit No. (Access)	Bit Name	Description/Enumeration	
		19	IO3PND
		20	IO4PND
		21	IO5PND
		22	IO6PND
		23	IO7PND
		24	DSCDONE - Master Only
		25	I2CERR - Master Only
		26	ICRCERR - Master Only
		41	PWRERR - Non-Localized Short to Ground
		42	PWRERR - Non-Localized Short to VBat
		253	Slave INTTYPE Read Error - Master Only
		254	Standby Done - Master Only
		255	MSTR_RUNNING - Master Only

Generate Bus Error

The A2B_GENERR register allows the host to generate bus errors from any node in the system through software. This register must be written over the A^{2}B bus, as writes to this register from the local I^{2}C port have no effect.

Address: 0x55

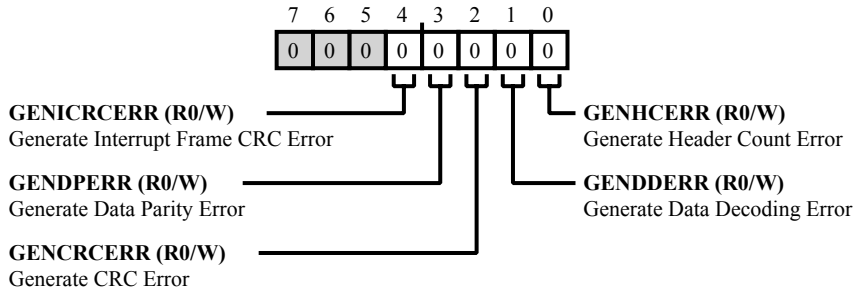

Figure 7-61: A2B_GENERR Register Diagram

Table 7-62: A2B_GENERR Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
4 (R0/W)	GENICRCERR	Generate Interrupt Frame CRC Error. A write of 1 to the A2B_GENERR.GENICRCERR bit instructs a slave node to generate an interrupt frame CRC error on the A^{2}B bus. A write of 1 to this bit in a master node has no effect.	
		0	No Action
		1	Generate Error
3 (R0/W)	GENDPERR	Generate Data Parity Error. A write of 1 to the A2B_GENERR.GENDPERR bit instructs the node to generate a data parity error on the A^{2}B bus.	
		0	No Action
		1	Generate Error
2 (R0/W)	GENCRCERR	Generate CRC Error. A write of 1 to the A2B_GENERR.GENCRCERR bit instructs the node to generate a CRC error on the A^{2}B bus.	
		0	No Action
		1	Generate Error

Table 7-62: A2B_GENERR Register Fields (Continued)

Bit No. (Access)	Bit Name	Description/Enumeration	
1 (R0/W)	GENDDERR	Generate Data Decoding Error. A write of 1 to the `A2B_GENERR.GENDDERR` bit instructs the node to generate a Data Decode Error on the A^{2}B bus.	
		0	No Action
		1	Generate Error
0 (R0/W)	GENHCERR	Generate Header Count Error. A write of 1 to the `A2B_GENERR.GENHCERR` bit instructs the node to generate a header count error on the A^{2}B bus.	
		0	No Action
		1	Generate Error

I2S Reduced Rate Register (Master Only, Auto-Broadcast)

The A2B_I2SRRATE register provides a means of reducing the A²B bus data rate by delivering data on a subset of superframes rather than on each superframe, thereby reducing the overall bus power.

When the A2B_I2SRRATE register is written in the master node, the new setting is automatically broadcast over the A²B bus to all discovered slave nodes. Local host writes to this register in a slave node have no effect.

Address: 0x56

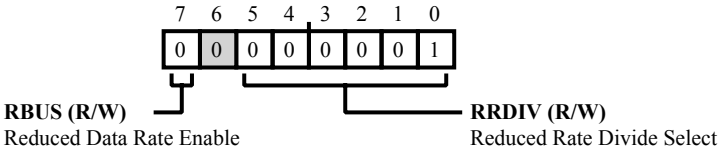

Figure 7-62: A2B_I2SRRATE Register Diagram

Table 7-63: A2B_I2SRRATE Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
7 (R/W)	RBUS	Reduced Data Rate Enable. When the A2B_I2SRRATE.RBUS bit is set, the bus is configured for reduced-rate data slots where downstream data and upstream data are only delivered once every A2B_I2SRRATE.RRDIV superframes.	
		0	Disabled
		1	Enabled
5:0 (R/W)	RRDIV	Reduced Rate Divide Select. The A2B_I2SRRATE.RRDIV bit field configures the superframe rate at which the I²S/TDM data is active on the bus. For example, when A2B_I2SRRATE.RRDIV =16, I²S/TDM data is active every 16th superframe rather than every superframe. Valid settings for this field are only those listed.	
		1	Superframe frequency (SFF)
		2	SFF/2
		4	SFF/4
		8	SFF/8
		12	SFF/12
		16	SFF/16
		20	SFF/20
		24	SFF/24
		28	SFF/28

Table 7-63: A2B_I2SRRATE Register Fields (Continued)

Bit No. (Access)	Bit Name	Description/Enumeration	
		32	SFF/32

I2S Reduced Rate Control Register

The A2B_I2SRRCTL register provides bits for controlling the I²S reduced rate strobe.

Address: 0x57

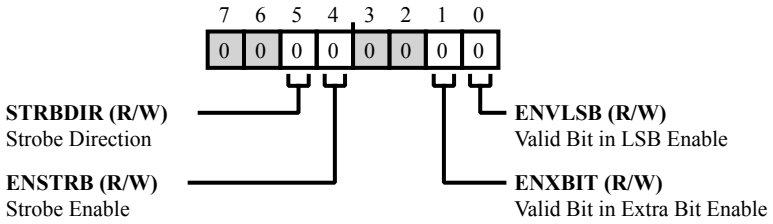

Figure 7-63: A2B_I2SRRCTL Register Diagram

Table 7-64: A2B_I2SRRCTL Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
5 (R/W)	STRBDIR	Strobe Direction. When the strobe signal is configured as an input, it influences the timing of frames on the bus. For a divide-by-N reduced rate, the strobe must be high once every N frames.	
		0	Input
		1	Output
4 (R/W)	ENSTRB	Strobe Enable. When the A2B_I2SRRCTL.ENSTRB bit is set, the IO7 pin is used as a strobe, indicating the audio frame where the reduced-rate data is updated.	
1 (R/W)	ENXBIT	Valid Bit in Extra Bit Enable. The A2B_I2SRRCTL.ENXBIT bit is only meaningful in a full-rate slave that is receiving reduced rate data from the bus. It does not affect data going over the bus. When the A2B_I2SRRCTL.ENXBIT bit is set, the bit after the LSB in each I²S/TDM channel is high for the superframe with new data and low for the other superframes.	
0 (R/W)	ENVLSB	Valid Bit in LSB Enable. If the A2B_I2SRRCTL.ENVLSB bit is set in a reduced-rate slave, the LSB of the data field is high for a new piece of data and low for a repeated piece of data. The A2B_I2SRRCTL.ENVLSB bit is applicable in the slave node only. If the reduced-rate slave node sets A2B_I2SRRCTL.ENVLSB and the receiving master's A2B_I2SRRCTL.ENXBIT bit is set, the output of the TDM data channel is xxxx11 for the first sampled word and xxxx00 for any repeated samples. Additionally, if the A2B_I2SRATE.SHARE bit is set in the reduced-rate slave, the LSB (additional bit) is high for the first data sample and low for the other samples.	

I2S Reduced Rate SYNC Offset Register (Slave Only)

The A2B_I2SRRSOFFS register controls the SYNC offset for slave nodes.

Address: 0x58

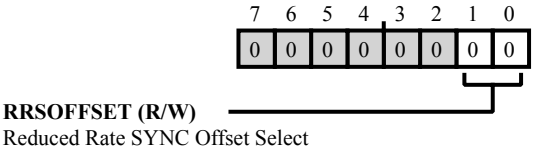

Figure 7-64: A2B_I2SRRSOFFS Register Diagram

Table 7-65: A2B_I2SRRSOFFS Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
1:0 (R/W)	RRSOFFSET	Reduced Rate SYNC Offset Select. A write of N to the A2B_I2SRRSOFFS.RRSOFFSET bit field instructs a slave node, using a reduced I²S/TDM rate, to offset the SYNC edge to the left by N superframes. When the reduced-rate slave's A2B_I2SRATE.SHARE bit is set, this field can only be programmed to 0 or 1.	
		0	No Offset
		1	1 Superframe Earlier
		2	2 Superframes Earlier
		3	3 Superframes Earlier

CLKOUT1 Configuration Register

The A2B_CLK1CFG register enables an output clock on the A2B_ADR1/A2B_IO1 pin and sets its frequency.

Address: 0x59

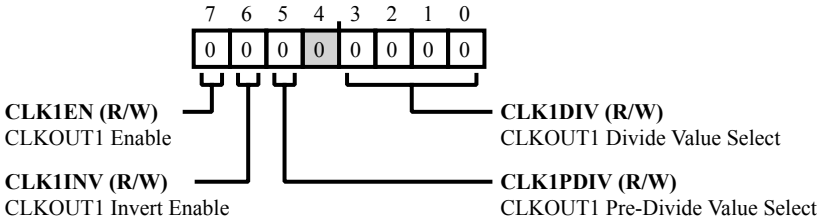

Figure 7-65: A2B_CLK1CFG Register Diagram

Table 7-66: A2B_CLK1CFG Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
7 (R/W)	CLK1EN	CLKOUT1 Enable. When the A2B_CLK1CFG.CLK1EN bit is set, the ADR1/IO1 pin is configured as a clock output, and GPIO functionality for the ADR1/IO1 pin is disabled.	
		0	Disabled
		1	Enabled
6 (R/W)	CLK1INV	CLKOUT1 Invert Enable. When the A2B_CLK1CFG.CLK1INV bit is set, the clock output to the ADR1/IO1 pin is inverted (moved 180 degrees out of phase).	
		0	Disabled
		1	Enabled
5 (R/W)	CLK1PDIV	CLKOUT1 Pre-Divide Value Select. The A2B_CLK1CFG.CLK1PDIV bit selects a pre-divide of either 2 or 32 from the PLL clock. At a 48 kHz sample frequency, the PLL clock has a frequency of 98.304 MHz. The PLL clock is 2048 times the sample frequency.	
		0	Pre-divide is 2
		1	Pre-divide is 32
3:0 (R/W)	CLK1DIV	CLKOUT1 Divide Value Select. The A2B_CLK1CFG.CLK1DIV bit field selects a divisor between 2 and 32 that is applied to the pre-divided clock before going to the pin. The divide ratio is 2 * (CLK1DIV + 1).	

CLKOUT2 Configuration Register

The A2B_CLK2CFG register enables an output clock on the A2B_ADR2/A2B_IO2 pin and sets its frequency.

Address: 0x5A

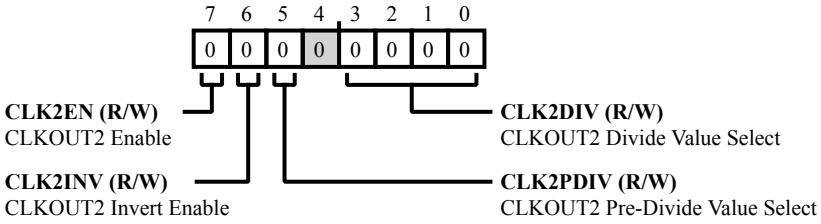

Figure 7-66: A2B_CLK2CFG Register Diagram

Table 7-67: A2B_CLK2CFG Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
7 (R/W)	CLK2EN	CLKOUT2 Enable. When the A2B_CLK2CFG.CLK2EN bit is set, the ADR2/IO2 pin is configured as a clock output, and GPIO functionality for the ADR2/IO2 pin is disabled.	
		0	Disabled
		1	Enabled
6 (R/W)	CLK2INV	CLKOUT2 Invert Enable. When the A2B_CLK2CFG.CLK2INV bit is set, the clock output to the ADR2/IO2 pin is inverted (moved 180 degrees out of phase).	
		0	Disabled
		1	Enabled
5 (R/W)	CLK2PDIV	CLKOUT2 Pre-Divide Value Select. The A2B_CLK2CFG.CLK2PDIV bit selects a pre-divide of either 2 or 32 from the PLL clock.	
		0	Pre-Divide is 2
		1	Pre-Divide is 32
3:0 (R/W)	CLK2DIV	CLKOUT2 Divide Value Select. The A2B_CLK2CFG.CLK2DIV bit field selects a divisor between 2 and 32 that is applied to the pre-divided clock before going to the pin. The divide ratio is 2 * (CLK1DIV + 1).	

Bus Monitor Mode Configuration Register

The A2B_BMMCFG register configures settings for bus monitor mode.

Address: 0x5B

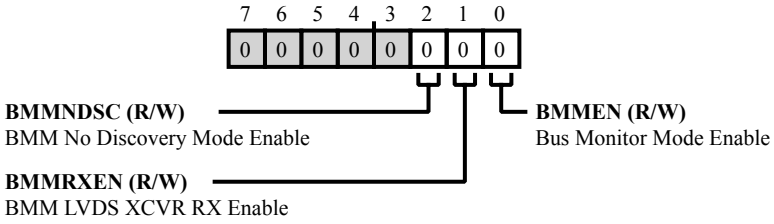

Figure 7-67: A2B_BMMCFG Register Diagram

Table 7-68: A2B_BMMCFG Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
2 (R/W)	BMMNDSC	BMM No Discovery Mode Enable. The A2B_BMMCFG.BMMNDSC bit is used to enable No Discovery Mode in Bus Monitor Mode.	
		0	Disabled
		1	Enabled
1 (R/W)	BMMRXEN	BMM LVDS XCVR RX Enable. The A2B_BMMCFG.BMMRXEN bit is used to enable LVDS RX in Bus Monitor Mode.	
		0	Disabled
		1	Enabled
0 (R/W)	BMMEN	Bus Monitor Mode Enable. The A2B_BMMCFG.BMMEN bit is used to enable Bus Monitor Mode.	
		0	Disabled
		1	Enabled

Sustain Configuration Register (Slave Only)

The A2B_SUSCFG register is used to configure sustain functionality in a slave node.

Address: 0x5C

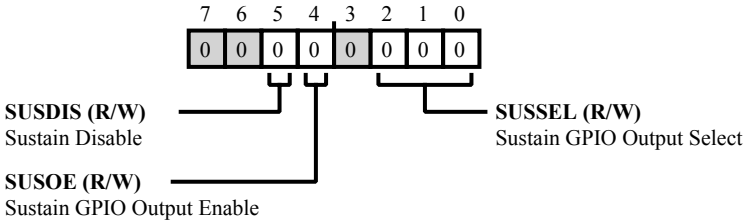

Figure 7-68: A2B_SUSCFG Register Diagram

Table 7-69: A2B_SUSCFG Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
5 (R/W)	SUSDIS	Sustain Disable.	
		0	Enable sustain mode
		1	Disable sustain mode
4 (R/W)	SUSOE	Sustain GPIO Output Enable.	
		0	Disable sustain mode output
		1	Enable sustain mode output
2:0 (R/W)	SUSSEL	Sustain GPIO Output Select.	
		0	Sustain output on IO0
		1	Sustain output on IO1
		2	Sustain output on IO2
		3	Sustain output on IO3
		4	Sustain output on IO4
		5	Sustain output on IO5
		6	Sustain output on IO6
		7	Sustain output on IO7

PDM Control 2 Register

The A2B_PDMCTL2 register provides a means of routing and handling PDM clock and data signals differently to accommodate various PDM configurations.

Address: 0x5D

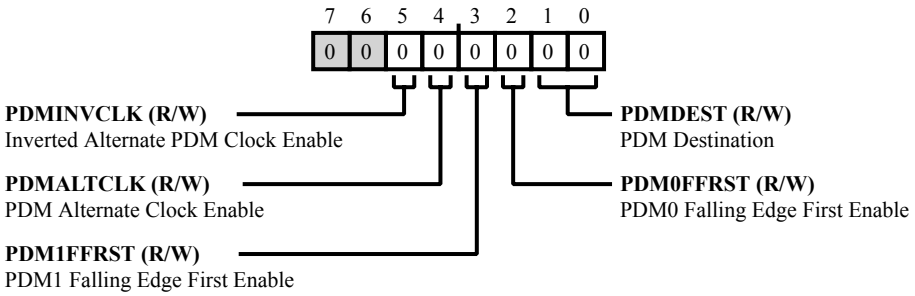

Figure 7-69: A2B_PDMCTL2 Register Diagram

Table 7-70: A2B_PDMCTL2 Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
5 (R/W)	PDMINVCLK	Inverted Alternate PDM Clock Enable. When the A2B_PDMCTL2.PDMINVCLK bit is set in a slave, and the A2B_PDMCTL2.PDMALTCLK bit is set, an inverted version of the PDMCLK on the IO7 pin is driven on the BCLK pin. I²S/TDM is still supported in this mode, but the BCLK frequency is constrained to 64x the SYNC frequency.
4 (R/W)	PDMALTCLK	PDM Alternate Clock Enable. When the A2B_PDMCTL2.PDMALTCLK bit is set and at least one PDM input pin is enabled, the IO7 pin is used as the PDMCLK clock output pin. For a slave node, this allows the BCLK frequency to be set from the I²S/TDM configuration even when PDM functions are enabled. For a master node, this allows the PDM clock to be a different frequency than the input BCLK. The frequency of the PDM clock on IO7 is 64x the SYNC frequency. If both PDM input pins are disabled (A2B_PDMCTL.PDM0EN = A2B_PDMCTL.PDM1EN = 0), the A2B_PDMCTL2.PDMALTCLK bit is ignored.
3 (R/W)	PDM1FFRST	PDM1 Falling Edge First Enable. When the A2B_PDMCTL2.PDM1FFRST bit is cleared (default), PDM1 data on the DRX1 pin is sampled rising edge first. When the A2B_PDMCTL2.PDM1FFRST bit is set, the DRX1 pin is sampled falling edge first.
2 (R/W)	PDM0FFRST	PDM0 Falling Edge First Enable. When the A2B_PDMCTL2.PDM0FFRST bit is cleared (default), PDM0 data on the DRX0 pin is sampled rising edge first. When the A2B_PDMCTL2.PDM0FFRST bit is set, the DRX0 pin is sampled falling edge first.

Table 7-70: A2B_PDMCTL2 Register Fields (Continued)

Bit No. (Access)	Bit Name	Description/Enumeration	
1:0 (R/W)	PDMDEST	PDM Destination. The `A2B_PDMCTL2.PDMDEST` bit field selects how PDM data is routed. By default, PDM data received by the DRX0 and DRX1 pins goes to the A^2B bus after demodulation. The demodulated data can instead or also be routed over the I^2S/TDM port to the local node using one or more of the DTXn pins.	
		0	(Default) A^2B bus only
		1	DTXn pin(s) only
		2	A^2B bus and DTXn pin(s)
		3	Reserved

Upstream Data RX Mask 0 Register (Slave Only)

The A2B_UPMASK0 register identifies which upstream data slots (from 0 to 7) are received from the A²B bus. These data slots may be transmitted via I²S/TDM and follow any downstream slots which were received by the slave node during the downstream portion of the superframe (defined by the A2B_LDNSLOTS register). Changes to this register only take effect after setting the A2B_CONTROL.NEWSTRCT bit of the master node.

Address: 0x60

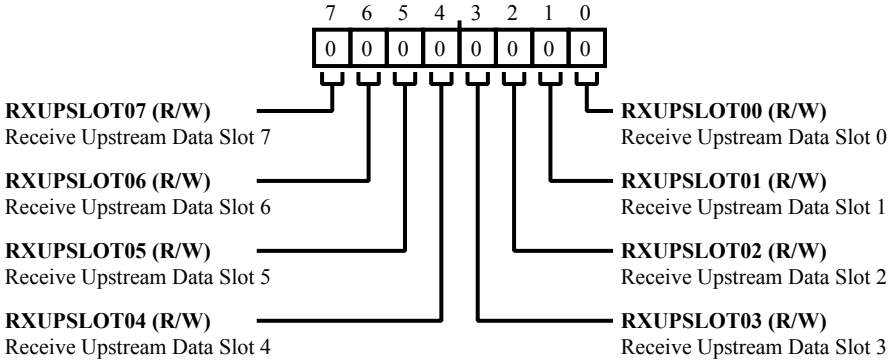

Figure 7-70: A2B_UPMASK0 Register Diagram

Table 7-71: A2B_UPMASK0 Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
7 (R/W)	RXUPSLOT07	Receive Upstream Data Slot 7. The A2B_UPMASK0.RXUPSLOT07 bit defines whether or not upstream data slot 7 is received by the local slave node.	
		0	Upstream Data Slot 7 RX Disabled
		1	Upstream Data Slot 7 RX Enabled
6 (R/W)	RXUPSLOT06	Receive Upstream Data Slot 6. The A2B_UPMASK0.RXUPSLOT06 bit defines whether or not upstream data slot 6 is received by the local slave node.	
		0	Upstream Data Slot 6 RX Disabled
		1	Upstream Data Slot 6 RX Enabled
5 (R/W)	RXUPSLOT05	Receive Upstream Data Slot 5. The A2B_UPMASK0.RXUPSLOT05 bit defines whether or not upstream data slot 5 is received by the local slave node.	
		0	Upstream Data Slot 5 RX Disabled
		1	Upstream Data Slot 5 RX Enabled

Table 7-71: A2B_UPMASK0 Register Fields (Continued)

Bit No. (Access)	Bit Name	Description/Enumeration	
4 (R/W)	RXUPSLOT04	Receive Upstream Data Slot 4. The `A2B_UPMASK0.RXUPSLOT04` bit defines whether or not upstream data slot 4 is received by the local slave node.	
		0	Upstream Data Slot 4 RX Disabled
		1	Upstream Data Slot 4 RX Enabled
3 (R/W)	RXUPSLOT03	Receive Upstream Data Slot 3. The `A2B_UPMASK0.RXUPSLOT03` bit defines whether or not upstream data slot 3 is received by the local slave node.	
		0	Upstream Data Slot 3 RX Disabled
		1	Upstream Data Slot 3 RX Enabled
2 (R/W)	RXUPSLOT02	Receive Upstream Data Slot 2. The `A2B_UPMASK0.RXUPSLOT02` bit defines whether or not upstream data slot 2 is received by the local slave node.	
		0	Upstream Data Slot 2 RX Disabled
		1	Upstream Data Slot 2 RX Enabled
1 (R/W)	RXUPSLOT01	Receive Upstream Data Slot 1. The `A2B_UPMASK0.RXUPSLOT01` bit defines whether or not upstream data slot 1 is received by the local slave node.	
		0	Upstream Data Slot 1 RX Disabled
		1	Upstream Data Slot 1 RX Enabled
0 (R/W)	RXUPSLOT00	Receive Upstream Data Slot 0. The `A2B_UPMASK0.RXUPSLOT00` bit defines whether or not upstream data slot 0 is received by the local slave node.	
		0	Upstream Data Slot 0 RX Disabled
		1	Upstream Data Slot 0 RX Enabled

Upstream Data RX Mask 1 Register (Slave Only)

The A2B_UPMASK1 register identifies which upstream data slots (from 8 to 15) are received from the A²B bus. These data slots may be transmitted via I²S/TDM and follow any downstream slots which were received by the slave node during the downstream portion of the superframe (defined by the A2B_LDNSLOTS register). Changes to this register only take effect after setting the A2B_CONTROL.NEWSTRCT bit of the master node.

Address: 0x61

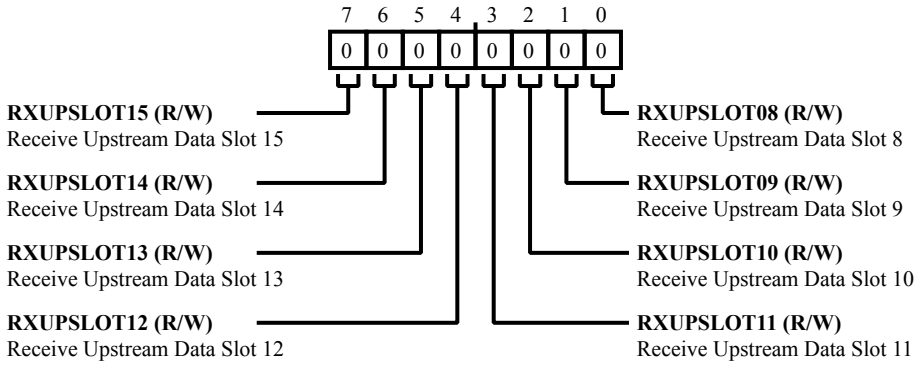

Figure 7-71: A2B_UPMASK1 Register Diagram

Table 7-72: A2B_UPMASK1 Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
7 (R/W)	RXUPSLOT15	Receive Upstream Data Slot 15. The A2B_UPMASK1.RXUPSLOT15 bit defines whether or not upstream data slot 15 is received by the local slave node.	
		0	Upstream Data Slot 15 RX Disabled
		1	Upstream Data Slot 15 RX Enabled
6 (R/W)	RXUPSLOT14	Receive Upstream Data Slot 14. The A2B_UPMASK1.RXUPSLOT14 bit defines whether or not upstream data slot 14 is received by the local slave node.	
		0	Upstream Data Slot 14 RX Disabled
		1	Upstream Data Slot 14 RX Enabled
5 (R/W)	RXUPSLOT13	Receive Upstream Data Slot 13. The A2B_UPMASK1.RXUPSLOT13 bit defines whether or not upstream data slot 13 is received by the local slave node.	
		0	Upstream Data Slot 13 RX Disabled
		1	Upstream Data Slot 13 RX Enabled

Table 7-72: A2B_UPMASK1 Register Fields (Continued)

Bit No. (Access)	Bit Name	Description/Enumeration	
4 (R/W)	RXUPSLOT12	Receive Upstream Data Slot 12. The `A2B_UPMASK1.RXUPSLOT12` bit defines whether or not upstream data slot 12 is received by the local slave node.	
		0	Upstream Data Slot 12 RX Disabled
		1	Upstream Data Slot 12 RX Enabled
3 (R/W)	RXUPSLOT11	Receive Upstream Data Slot 11. The `A2B_UPMASK1.RXUPSLOT11` bit defines whether or not upstream data slot 11 is received by the local slave node.	
		0	Upstream Data Slot 11 RX Disabled
		1	Upstream Data Slot 11 RX Enabled
2 (R/W)	RXUPSLOT10	Receive Upstream Data Slot 10. The `A2B_UPMASK1.RXUPSLOT10` bit defines whether or not upstream data slot 10 is received by the local slave node.	
		0	Upstream Data Slot 10 RX Disabled
		1	Upstream Data Slot 10 RX Enabled
1 (R/W)	RXUPSLOT09	Receive Upstream Data Slot 9. The `A2B_UPMASK1.RXUPSLOT09` bit defines whether or not upstream data slot 9 is received by the local slave node.	
		0	Upstream Data Slot 9 RX Disabled
		1	Upstream Data Slot 9 RX Enabled
0 (R/W)	RXUPSLOT08	Receive Upstream Data Slot 8. The `A2B_UPMASK1.RXUPSLOT08` bit defines whether or not upstream data slot 8 is received by the local slave node.	
		0	Upstream Data Slot 8 RX Disabled
		1	Upstream Data Slot 8 RX Enabled

Upstream Data RX Mask 2 Register (Slave Only)

The A2B_UPMASK2 register identifies which upstream data slots (from 16 to 23) are received from the A²B bus. These data slots may be transmitted via I²S/TDM and follow any downstream slots which were received by the slave node during the downstream portion of the superframe (defined by the A2B_LDNSLOTS register). Changes to this register only take effect after setting the A2B_CONTROL.NEWSTRCT bit of the master node.

Address: 0x62

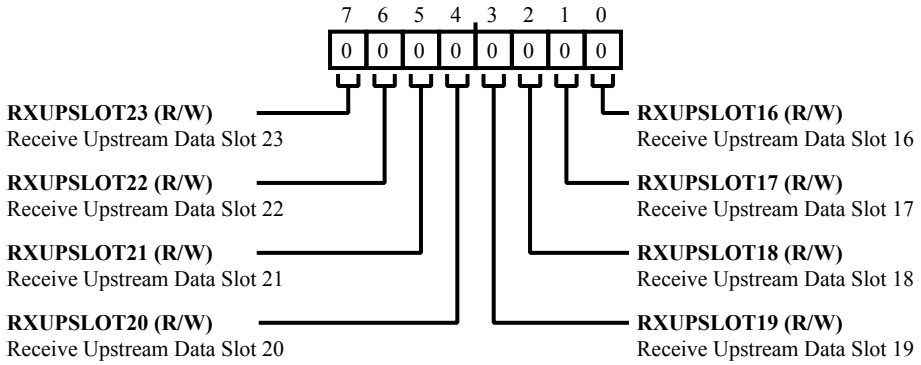

Figure 7-72: A2B_UPMASK2 Register Diagram

Table 7-73: A2B_UPMASK2 Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
7 (R/W)	RXUPSLOT23	Receive Upstream Data Slot 23. The A2B_UPMASK2.RXUPSLOT23 bit defines whether or not upstream data slot 23 is received by the local slave node.	
		0	Upstream Data Slot 23 RX Disabled
		1	Upstream Data Slot 23 RX Enabled
6 (R/W)	RXUPSLOT22	Receive Upstream Data Slot 22. The A2B_UPMASK2.RXUPSLOT22 bit defines whether or not upstream data slot 22 is received by the local slave node.	
		0	Upstream Data Slot 22 RX Disabled
		1	Upstream Data Slot 22 RX Enabled
5 (R/W)	RXUPSLOT21	Receive Upstream Data Slot 21. The A2B_UPMASK2.RXUPSLOT21 bit defines whether or not upstream data slot 21 is received by the local slave node.	
		0	Upstream Data Slot 21 RX Disabled
		1	Upstream Data Slot 21 RX Enabled

Table 7-73: A2B_UPMASK2 Register Fields (Continued)

Bit No. (Access)	Bit Name	Description/Enumeration	
4 (R/W)	RXUPSLOT20	Receive Upstream Data Slot 20. The `A2B_UPMASK2.RXUPSLOT20` bit defines whether or not upstream data slot 20 is received by the local slave node.	
		0	Upstream Data Slot 20 RX Disabled
		1	Upstream Data Slot 20 RX Enabled
3 (R/W)	RXUPSLOT19	Receive Upstream Data Slot 19. The `A2B_UPMASK2.RXUPSLOT19` bit defines whether or not upstream data slot 19 is received by the local slave node.	
		0	Upstream Data Slot 19 RX Disabled
		1	Upstream Data Slot 19 RX Enabled
2 (R/W)	RXUPSLOT18	Receive Upstream Data Slot 18. The `A2B_UPMASK2.RXUPSLOT18` bit defines whether or not upstream data slot 18 is received by the local slave node.	
		0	Upstream Data Slot 18 RX Disabled
		1	Upstream Data Slot 18 RX Enabled
1 (R/W)	RXUPSLOT17	Receive Upstream Data Slot 17. The `A2B_UPMASK2.RXUPSLOT17` bit defines whether or not upstream data slot 17 is received by the local slave node.	
		0	Upstream Data Slot 17 RX Disabled
		1	Upstream Data Slot 17 RX Enabled
0 (R/W)	RXUPSLOT16	Receive Upstream Data Slot 16. The `A2B_UPMASK2.RXUPSLOT16` bit defines whether or not upstream data slot 16 is received by the local slave node.	
		0	Upstream Data Slot 16 RX Disabled
		1	Upstream Data Slot 16 RX Enabled

Upstream Data RX Mask 3 Register (Slave Only)

The A2B_UPMASK3 register identifies which upstream data slots (from 24 to 31) are received from the A²B bus. These data slots may be transmitted via I²S/TDM and follow any downstream slots which were received by the slave node during the downstream portion of the superframe (defined by the A2B_LDNSLOTS register). Changes to this register only take effect after setting the A2B_CONTROL.NEWSTRCT bit of the master node.

Address: 0x63

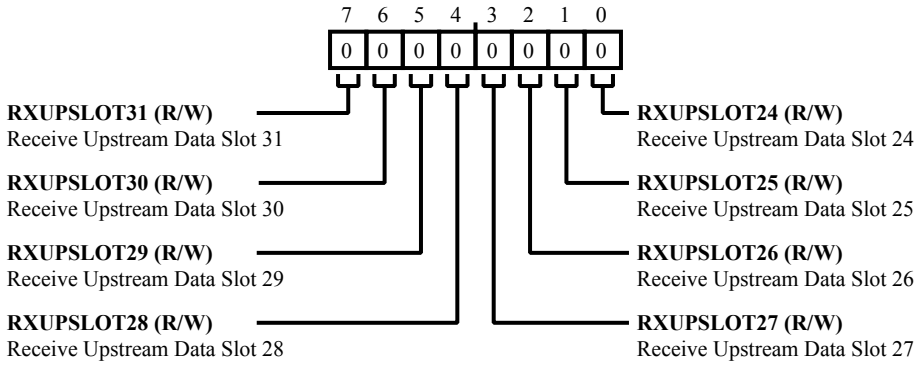

Figure 7-73: A2B_UPMASK3 Register Diagram

Table 7-74: A2B_UPMASK3 Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
7 (R/W)	RXUPSLOT31	Receive Upstream Data Slot 31. The A2B_UPMASK3.RXUPSLOT31 bit defines whether or not upstream data slot 31 is received by the local slave node.	
		0	Upstream Data Slot 31 RX Disabled
		1	Upstream Data Slot 31 RX Enabled
6 (R/W)	RXUPSLOT30	Receive Upstream Data Slot 30. The A2B_UPMASK3.RXUPSLOT30 bit defines whether or not upstream data slot 30 is received by the local slave node.	
		0	Upstream Data Slot 30 RX Disabled
		1	Upstream Data Slot 30 RX Enabled
5 (R/W)	RXUPSLOT29	Receive Upstream Data Slot 29. The A2B_UPMASK3.RXUPSLOT29 bit defines whether or not upstream data slot 29 is received by the local slave node.	
		0	Upstream Data Slot 29 RX Disabled
		1	Upstream Data Slot 29 RX Enabled

Table 7-74: A2B_UPMASK3 Register Fields (Continued)

Bit No. (Access)	Bit Name	Description/Enumeration	
4 (R/W)	RXUPSLOT28	Receive Upstream Data Slot 28. The `A2B_UPMASK3.RXUPSLOT28` bit defines whether or not upstream data slot 28 is received by the local slave node.	
		0	Upstream Data Slot 28 RX Disabled
		1	Upstream Data Slot 28 RX Enabled
3 (R/W)	RXUPSLOT27	Receive Upstream Data Slot 27. The `A2B_UPMASK3.RXUPSLOT27` bit defines whether or not upstream data slot 27 is received by the local slave node.	
		0	Upstream Data Slot 27 RX Disabled
		1	Upstream Data Slot 27 RX Enabled
2 (R/W)	RXUPSLOT26	Receive Upstream Data Slot 26. The `A2B_UPMASK3.RXUPSLOT26` bit defines whether or not upstream data slot 26 is received by the local slave node.	
		0	Upstream Data Slot 26 RX Disabled
		1	Upstream Data Slot 26 RX Enabled
1 (R/W)	RXUPSLOT25	Receive Upstream Data Slot 25. The `A2B_UPMASK3.RXUPSLOT25` bit defines whether or not upstream data slot 25 is received by the local slave node.	
		0	Upstream Data Slot 25 RX Disabled
		1	Upstream Data Slot 25 RX Enabled
0 (R/W)	RXUPSLOT24	Receive Upstream Data Slot 24. The `A2B_UPMASK3.RXUPSLOT24` bit defines whether or not upstream data slot 24 is received by the local slave node.	
		0	Disabled
		1	Enabled

Local Upstream Channel Offset Register (Slave Only)

In a slave node, the A2B_UPOFFSET register defines the number of data channels received via I²S/TDM/PDM that are skipped before data slots are transmitted upstream on the A²B bus.

Address: 0x64

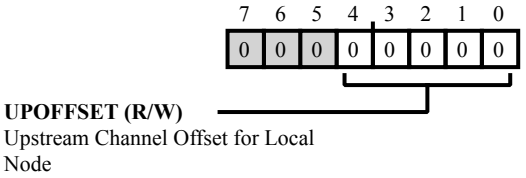

Figure 7-74: A2B_UPOFFSET Register Diagram

Table 7-75: A2B_UPOFFSET Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
4:0 (R/W)	UPOFFSET	Upstream Channel Offset for Local Node. The A2B_UPOFFSET.UPOFFSET bit field defines the number of data channels received via I²S/TDM/PDM that are skipped before data slots are transmitted upstream on the A²B bus.

Downstream Data RX Mask 0 Register (Slave Only)

The A2B_DNMASK0 register identifies the downstream data slots (from 0 to 7) that are received from the A^{2}B bus. These data slots may be transmitted via I^{2}S/TDM. If none of the bits in this register are set, the A2B_LDNSLOTS register defines the number of downstream data slots taken by the local node, as in the AD2410. Changes to this register only take effect after setting the A2B_CONTROL.NEWSTRCT bit of the master node.

Address: 0x65

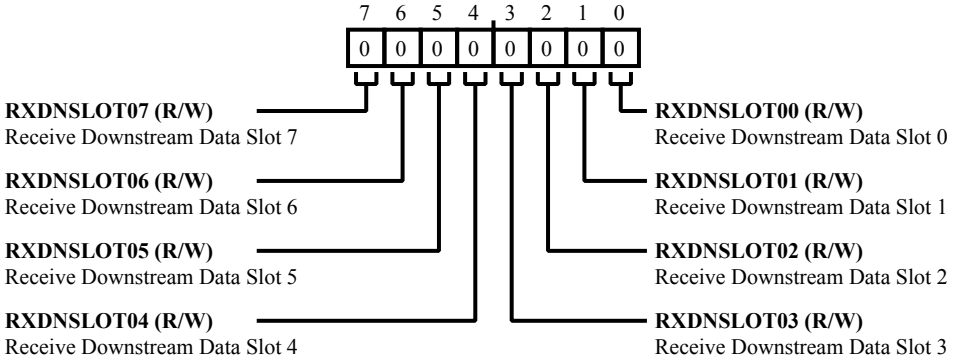

Figure 7-75: A2B_DNMASK0 Register Diagram

Table 7-76: A2B_DNMASK0 Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
7 (R/W)	RXDNSLOT07	Receive Downstream Data Slot 7. The A2B_DNMASK0.RXDNSLOT07 bit defines whether or not downstream data slot 7 is received by the local slave node.	
		0	Downstream Data Slot 7 RX Disabled
		1	Downstream Data Slot 7 RX Enabled
6 (R/W)	RXDNSLOT06	Receive Downstream Data Slot 6. The A2B_DNMASK0.RXDNSLOT06 bit defines whether or not downstream data slot 6 is received by the local slave node.	
		0	Downstream Data Slot 6 RX Disabled
		1	Downstream Data Slot 6 RX Enabled
5 (R/W)	RXDNSLOT05	Receive Downstream Data Slot 5. The A2B_DNMASK0.RXDNSLOT05 bit defines whether or not downstream data slot 5 is received by the local slave node.	
		0	Downstream Data Slot 5 RX Disabled
		1	Downstream Data Slot 5 RX Enabled

Table 7-76: A2B_DNMASK0 Register Fields (Continued)

Bit No. (Access)	Bit Name	Description/Enumeration	
4 (R/W)	RXDNSLOT04	Receive Downstream Data Slot 4. The `A2B_DNMASK0.RXDNSLOT04` bit defines whether or not downstream data slot 4 is received by the local slave node.	
		0	Downstream Data Slot 4 RX Disabled
		1	Downstream Data Slot 4 RX Enabled
3 (R/W)	RXDNSLOT03	Receive Downstream Data Slot 3. The `A2B_DNMASK0.RXDNSLOT03` bit defines whether or not downstream data slot 3 is received by the local slave node.	
		0	Downstream Data Slot 3 RX Disabled
		1	Downstream Data Slot 3 RX Enabled
2 (R/W)	RXDNSLOT02	Receive Downstream Data Slot 2. The `A2B_DNMASK0.RXDNSLOT02` bit defines whether or not downstream data slot 2 is received by the local slave node.	
		0	Downstream Data Slot 2 RX Disabled
		1	Downstream Data Slot 2 RX Enabled
1 (R/W)	RXDNSLOT01	Receive Downstream Data Slot 1. The `A2B_DNMASK0.RXDNSLOT01` bit defines whether or not downstream data slot 1 is received by the local slave node.	
		0	Downstream Data Slot 1 RX Disabled
		1	Downstream Data Slot 1 RX Enabled
0 (R/W)	RXDNSLOT00	Receive Downstream Data Slot 0. The `A2B_DNMASK0.RXDNSLOT00` bit defines whether or not downstream data slot 0 is received by the local slave node.	
		0	Downstream Data Slot 0 RX Disabled
		1	Downstream Data Slot 0 RX Enabled

Downstream Data RX Mask 1 Register (Slave Only)

The A2B_DNMASK1 register identifies the downstream data slots (from 8 to 15) that are received from the A²B bus. These data slots may be transmitted via I²S/TDM. If none of the bits in this register are set, the A2B_LDNSLOTS register defines the number of downstream data slots taken by the local node, as in the AD2410. Changes to this register only take effect after setting the A2B_CONTROL.NEWSTRCT bit of the master node.

Address: 0x66

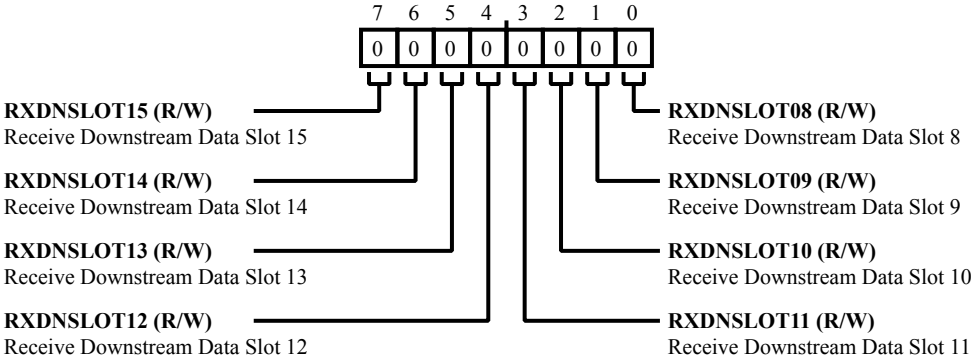

Figure 7-76: A2B_DNMASK1 Register Diagram

Table 7-77: A2B_DNMASK1 Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
7 (R/W)	RXDNSLOT15	Receive Downstream Data Slot 15. The A2B_DNMASK1.RXDNSLOT15 bit defines whether or not downstream data slot 15 is received by the local slave node.	
		0	Downstream Data Slot 15 RX Disabled
		1	Downstream Data Slot 15 RX Enabled
6 (R/W)	RXDNSLOT14	Receive Downstream Data Slot 14. The A2B_DNMASK1.RXDNSLOT14 bit defines whether or not downstream data slot 14 is received by the local slave node.	
		0	Downstream Data Slot 14 RX Disabled
		1	Downstream Data Slot 14 RX Enabled
5 (R/W)	RXDNSLOT13	Receive Downstream Data Slot 13. The A2B_DNMASK1.RXDNSLOT13 bit defines whether or not downstream data slot 13 is received by the local slave node.	
		0	Downstream Data Slot 13 RX Disabled
		1	Downstream Data Slot 13 RX Enabled

Table 7-77: A2B_DNMASK1 Register Fields (Continued)

Bit No. (Access)	Bit Name	Description/Enumeration	
4 (R/W)	RXDNSLOT12	Receive Downstream Data Slot 12. The `A2B_DNMASK1.RXDNSLOT12` bit defines whether or not downstream data slot 12 is received by the local slave node.	
		0	Downstream Data Slot 12 RX Disabled
		1	Downstream Data Slot 12 RX Enabled
3 (R/W)	RXDNSLOT11	Receive Downstream Data Slot 11. The `A2B_DNMASK1.RXDNSLOT11` bit defines whether or not downstream data slot 11 is received by the local slave node.	
		0	Downstream Data Slot 11 RX Disabled
		1	Downstream Data Slot 11 RX Enabled
2 (R/W)	RXDNSLOT10	Receive Downstream Data Slot 10. The `A2B_DNMASK1.RXDNSLOT10` bit defines whether or not downstream data slot 10 is received by the local slave node.	
		0	Downstream Data Slot 10 RX Disabled
		1	Downstream Data Slot 10 RX Enabled
1 (R/W)	RXDNSLOT09	Receive Downstream Data Slot 9. The `A2B_DNMASK1.RXDNSLOT09` bit defines whether or not downstream data slot 9 is received by the local slave node.	
		0	Downstream Data Slot 9 RX Disabled
		1	Downstream Data Slot 9 RX Enabled
0 (R/W)	RXDNSLOT08	Receive Downstream Data Slot 8. The `A2B_DNMASK1.RXDNSLOT08` bit defines whether or not downstream data slot 8 is received by the local slave node.	
		0	Downstream Data Slot 8 RX Disabled
		1	Downstream Data Slot 8 RX Enabled

Downstream Data RX Mask 2 Register (Slave Only)

The A2B_DNMASK2 register identifies the downstream data slots (from 16 to 23) that are received from the A^{2}B bus. These data slots may be transmitted via I^{2}S/TDM. If none of the bits in this register are set, the A2B_LDNSLOTS register defines the number of downstream data slots taken by the local node. Changes to this register only take effect after setting the A2B_CONTROL.NEWSTRCT bit of the master node.

Address: 0x67

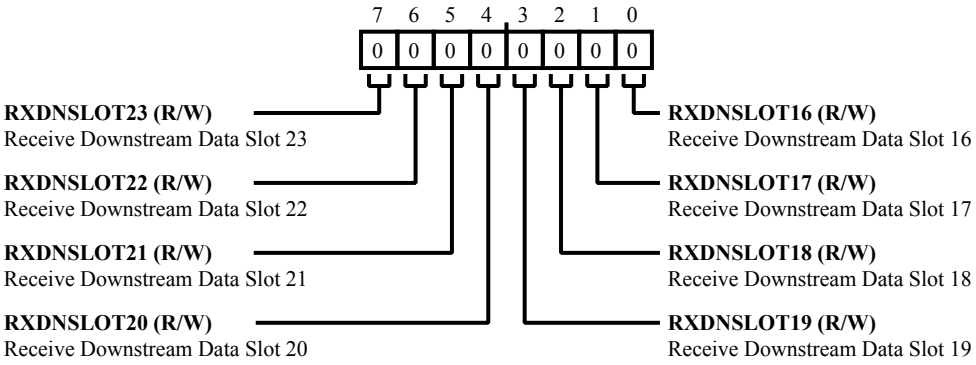

Figure 7-77: A2B_DNMASK2 Register Diagram

Table 7-78: A2B_DNMASK2 Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
7 (R/W)	RXDNSLOT23	Receive Downstream Data Slot 23. The A2B_DNMASK2.RXDNSLOT23 bit defines whether or not downstream data slot 23 is received by the local slave node.	
		0	Downstream Data Slot 23 RX Disabled
		1	Downstream Data Slot 23 RX Enabled
6 (R/W)	RXDNSLOT22	Receive Downstream Data Slot 22. The A2B_DNMASK2.RXDNSLOT22 bit defines whether or not downstream data slot 22 is received by the local slave node.	
		0	Downstream Data Slot 22 RX Disabled
		1	Downstream Data Slot 22 RX Enabled
5 (R/W)	RXDNSLOT21	Receive Downstream Data Slot 21. The A2B_DNMASK2.RXDNSLOT21 bit defines whether or not downstream data slot 21 is received by the local slave node.	
		0	Downstream Data Slot 21 RX Disabled
		1	Downstream Data Slot 21 RX Enabled

Table 7-78: A2B_DNMASK2 Register Fields (Continued)

Bit No. (Access)	Bit Name	Description/Enumeration	
4 (R/W)	RXDNSLOT20	Receive Downstream Data Slot 20. The `A2B_DNMASK2.RXDNSLOT20` bit defines whether or not downstream data slot 20 is received by the local slave node.	
		0	Downstream Data Slot 20 RX Disabled
		1	Downstream Data Slot 20 RX Enabled
3 (R/W)	RXDNSLOT19	Receive Downstream Data Slot 19. The `A2B_DNMASK2.RXDNSLOT19` bit defines whether or not downstream data slot 19 is received by the local slave node.	
		0	Downstream Data Slot 19 RX Disabled
		1	Downstream Data Slot 19 RX Enabled
2 (R/W)	RXDNSLOT18	Receive Downstream Data Slot 18. The `A2B_DNMASK2.RXDNSLOT18` bit defines whether or not downstream data slot 18 is received by the local slave node.	
		0	Downstream Data Slot 18 RX Disabled
		1	Downstream Data Slot 18 RX Enabled
1 (R/W)	RXDNSLOT17	Receive Downstream Data Slot 17. The `A2B_DNMASK2.RXDNSLOT17` bit defines whether or not downstream data slot 17 is received by the local slave node.	
		0	Downstream Data Slot 17 RX Disabled
		1	Downstream Data Slot 17 RX Enabled
0 (R/W)	RXDNSLOT16	Receive Downstream Data Slot 16. The `A2B_DNMASK2.RXDNSLOT16` bit defines whether or not downstream data slot 16 is received by the local slave node.	
		0	Downstream Data Slot 16 RX Disabled
		1	Downstream Data Slot 16 RX Enabled

Downstream Data RX Mask 3 Register (Slave Only)

The A2B_DNMASK3 register identifies the downstream data slots (from 24 to 31) that are received from the A²B bus. These data slots may be transmitted via I²S/TDM. If none of the bits in this register are set, the A2B_LDNSLOTS register defines the number of downstream data slots taken by the local node, as in the AD2410. Changes to this register only take effect after setting the A2B_CONTROL.NEWSTRCT bit of the master node.

Address: 0x68

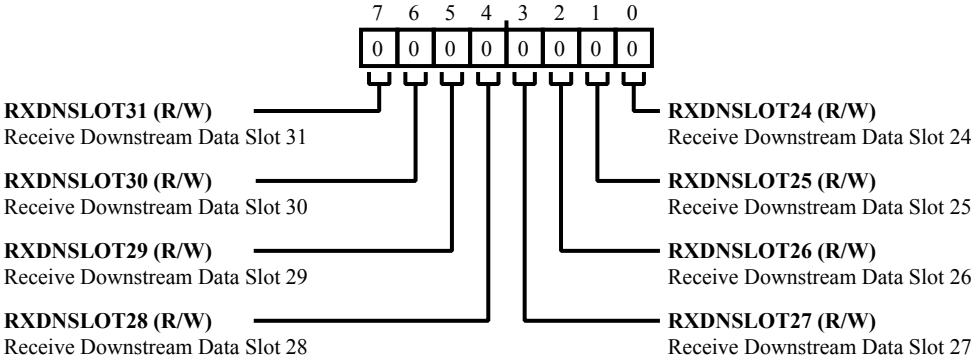

Figure 7-78: A2B_DNMASK3 Register Diagram

Table 7-79: A2B_DNMASK3 Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
7 (R/W)	RXDNSLOT31	Receive Downstream Data Slot 31. The A2B_DNMASK3.RXDNSLOT31 bit defines whether or not downstream data slot 31 is received by the local slave node.	
		0	Downstream Data Slot 31 RX Disabled
		1	Downstream Data Slot 31 RX Enabled
6 (R/W)	RXDNSLOT30	Receive Downstream Data Slot 30. The A2B_DNMASK3.RXDNSLOT30 bit defines whether or not downstream data slot 30 is received by the local slave node.	
		0	Downstream Data Slot 30 RX Disabled
		1	Downstream Data Slot 30 RX Enabled
5 (R/W)	RXDNSLOT29	Receive Downstream Data Slot 29. The A2B_DNMASK3.RXDNSLOT29 bit defines whether or not downstream data slot 29 is received by the local slave node.	
		0	Downstream Data Slot 29 RX Disabled
		1	Downstream Data Slot 29 RX Enabled

Table 7-79: A2B_DNMASK3 Register Fields (Continued)

Bit No. (Access)	Bit Name	Description/Enumeration	
4 (R/W)	RXDNSLOT28	Receive Downstream Data Slot 28. The `A2B_DNMASK3.RXDNSLOT28` bit defines whether or not downstream data slot 28 is received by the local slave node.	
		0	Downstream Data Slot 28 RX Disabled
		1	Downstream Data Slot 28 RX Enabled
3 (R/W)	RXDNSLOT27	Receive Downstream Data Slot 27. The `A2B_DNMASK3.RXDNSLOT27` bit defines whether or not downstream data slot 27 is received by the local slave node.	
		0	Downstream Data Slot 27 RX Disabled
		1	Downstream Data Slot 27 RX Enabled
2 (R/W)	RXDNSLOT26	Receive Downstream Data Slot 26. The `A2B_DNMASK3.RXDNSLOT26` bit defines whether or not downstream data slot 26 is received by the local slave node.	
		0	Downstream Data Slot 26 RX Disabled
		1	Downstream Data Slot 26 RX Enabled
1 (R/W)	RXDNSLOT25	Receive Downstream Data Slot 25. The `A2B_DNMASK3.RXDNSLOT25` bit defines whether or not downstream data slot 25 is received by the local slave node.	
		0	Downstream Data Slot 25 RX Disabled
		1	Downstream Data Slot 25 RX Enabled
0 (R/W)	RXDNSLOT24	Receive Downstream Data Slot 24. The `A2B_DNMASK3.RXDNSLOT24` bit defines whether or not downstream data slot 24 is received by the local slave node.	
		0	Downstream Data Slot 24 RX Disabled
		1	Downstream Data Slot 24 RX Enabled

Local Downstream Channel Offset Register (Slave Only)

In a slave node, the A2B_DNOFFSET register defines the number of data channels received via I²S/TDM/PDM that are skipped before data slots are transmitted downstream on the A²B bus. The value in the A2B_DNOFFSET register is used only if any of the bits in the A2B_DNMASK0 through A2B_DNMASK3 registers are set and the A2B_LDNSLOTS register is non-zero.

Address: 0x69

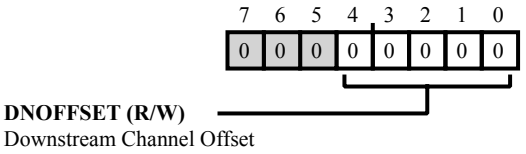

Figure 7-79: A2B_DNOFFSET Register Diagram

Table 7-80: A2B_DNOFFSET Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
4:0 (R/W)	DNOFFSET	Downstream Channel Offset. The A2B_DNOFFSET.DNOFFSET bit field defines the number of data channels received via I²S/TDM/PDM that are skipped before data slots are transmitted downstream on the A²B bus.

Chip ID Register 0

The A2B_CHIPID0 through A2B_CHIPID5 registers concatenate to form a unique 48-bit ID for the transceiver, where A2B_CHIPID0 contains the LSB (bits 7:0) and A2B_CHIPID5 contains the MSB (bits 47:40).

Address: 0x6A

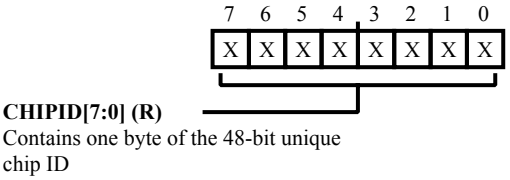

Figure 7-80: A2B_CHIPID0 Register Diagram

Table 7-81: A2B_CHIPID0 Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
7:0 (R/NW)	CHIPID	Contains one byte of the 48-bit unique chip ID.

Chip ID Register 1

The A2B_CHIPID0 through A2B_CHIPID5 registers concatenate to form a unique 48-bit ID for the transceiver, where A2B_CHIPID0 contains the LSB (bits 7:0) and A2B_CHIPID5 contains the MSB (bits 47:40).

Address: 0x6B

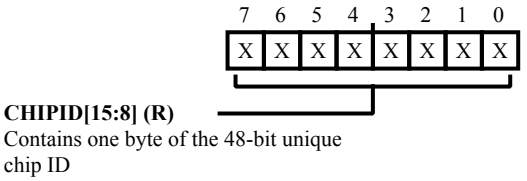

Figure 7-81: A2B_CHIPID1 Register Diagram

Table 7-82: A2B_CHIPID1 Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
7:0 (R/NW)	CHIPID	Contains one byte of the 48-bit unique chip ID.

Chip ID Register 2

The A2B_CHIPID0 through A2B_CHIPID5 registers concatenate to form a unique 48-bit ID for the transceiver, where A2B_CHIPID0 contains the LSB (bits 7:0) and A2B_CHIPID5 contains the MSB (bits 47:40).

Address: 0x6C

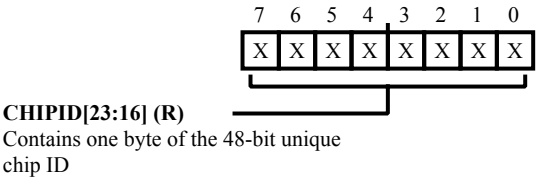

Figure 7-82: A2B_CHIPID2 Register Diagram

Table 7-83: A2B_CHIPID2 Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
7:0 (R/NW)	CHIPID	Contains one byte of the 48-bit unique chip ID.

Chip ID Register 3

The A2B_CHIPID0 through A2B_CHIPID5 registers concatenate to form a unique 48-bit ID for the transceiver, where A2B_CHIPID0 contains the LSB (bits 7:0) and A2B_CHIPID5 contains the MSB (bits 47:40).

Address: 0x6D

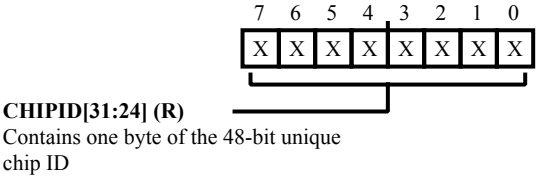

Figure 7-83: A2B_CHIPID3 Register Diagram

Table 7-84: A2B_CHIPID3 Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
7:0 (R/NW)	CHIPID	Contains one byte of the 48-bit unique chip ID.

Chip ID Register 4

The A2B_CHIPID0 through A2B_CHIPID5 registers concatenate to form a unique 48-bit ID for the transceiver, where A2B_CHIPID0 contains the LSB (bits 7:0) and A2B_CHIPID5 contains the MSB (bits 47:40).

Address: 0x6E

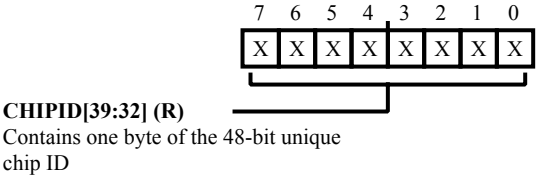

Figure 7-84: A2B_CHIPID4 Register Diagram

Table 7-85: A2B_CHIPID4 Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
7:0 (R/NW)	CHIPID	Contains one byte of the 48-bit unique chip ID.

Chip ID Register 5

The A2B_CHIPID0 through A2B_CHIPID5 registers concatenate to form a unique 48-bit ID for the transceiver, where A2B_CHIPID0 contains the LSB (bits 7:0) and A2B_CHIPID5 contains the MSB (bits 47:40).

Address: 0x6F

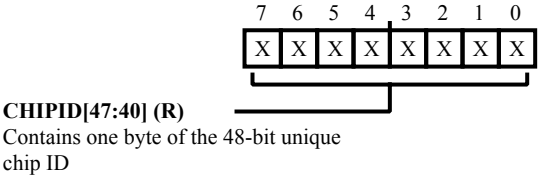

Figure 7-85: A2B_CHIPID5 Register Diagram

Table 7-86: A2B_CHIPID5 Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
7:0 (R/NW)	CHIPID	Contains one byte of the 48-bit unique chip ID.

GPIO Over Distance Enable Register

The A2B_GPIODEN register controls the general-purpose I/O pins for use in GPIO Over Distance.

Address: 0x80

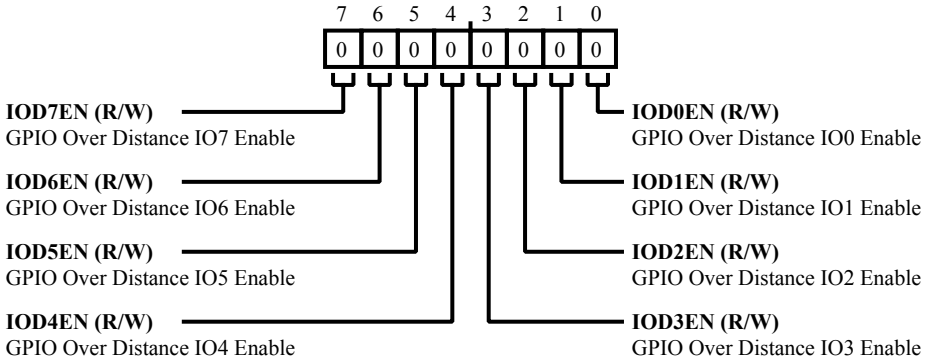

Figure 7-86: A2B_GPIODEN Register Diagram

Table 7-87: A2B_GPIODEN Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
7 (R/W)	IOD7EN	GPIO Over Distance IO7 Enable. The A2B_GPIODEN.IOD7EN bit enables GPIO Over Distance for IO7.	
		0	GPIO Over Distance for IO7 Disabled
		1	GPIO Over Distance for IO7 Enabled
6 (R/W)	IOD6EN	GPIO Over Distance IO6 Enable. The A2B_GPIODEN.IOD6EN bit enables GPIO Over Distance for IO6.	
		0	GPIO Over Distance for IO6 Disabled
		1	GPIO Over Distance for IO6 Enabled
5 (R/W)	IOD5EN	GPIO Over Distance IO5 Enable. The A2B_GPIODEN.IOD5EN bit enables GPIO Over Distance for IO5.	
		0	GPIO Over Distance for IO5 Disabled
		1	GPIO Over Distance for IO5 Enabled
4 (R/W)	IOD4EN	GPIO Over Distance IO4 Enable. The A2B_GPIODEN.IOD4EN bit enables GPIO Over Distance for IO4.	
		0	GPIO Over Distance for IO4 Disabled
		1	GPIO Over Distance for IO4 Enabled

Table 7-87: A2B_GPIODEN Register Fields (Continued)

Bit No. (Access)	Bit Name	Description/Enumeration	
3 (R/W)	IOD3EN	GPIO Over Distance IO3 Enable. The `A2B_GPIODEN.IOD3EN` bit enables GPIO Over Distance for IO3.	
		0	GPIO Over Distance for IO3 Disabled
		1	GPIO Over Distance for IO3 Enabled
2 (R/W)	IOD2EN	GPIO Over Distance IO2 Enable. The `A2B_GPIODEN.IOD2EN` bit enables GPIO Over Distance for IO2.	
		0	GPIO Over Distance for IO2 Disabled
		1	GPIO Over Distance for IO2 Enabled
1 (R/W)	IOD1EN	GPIO Over Distance IO1 Enable. The `A2B_GPIODEN.IOD1EN` bit enables GPIO Over Distance for IO1.	
		0	GPIO Over Distance for IO1 Disabled
		1	GPIO Over Distance for IO1 Enabled
0 (R/W)	IOD0EN	GPIO Over Distance IO0 Enable. The `A2B_GPIODEN.IOD0EN` bit enables GPIO Over Distance for IO0.	
		0	GPIO Over Distance for IO0 Disabled
		1	GPIO Over Distance for IO0 Enabled

GPIO Over Distance Mask 0 Register

Address: 0x81

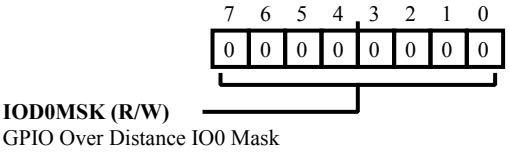

Figure 7-87: A2B_GPIOD0MSK Register Diagram

Table 7-88: A2B_GPIOD0MSK Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
7:0 (R/W)	IOD0MSK	GPIO Over Distance IO0 Mask.

GPIO Over Distance Mask 1 Register

Address: 0x82

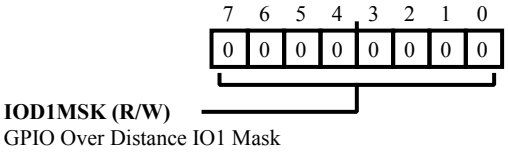

Figure 7-88: A2B_GPIOD1MSK Register Diagram

Table 7-89: A2B_GPIOD1MSK Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
7:0 (R/W)	IOD1MSK	GPIO Over Distance IO1 Mask.

GPIO Over Distance Mask 2 Register

Address: 0x83

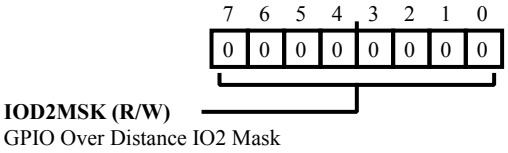

Figure 7-89: A2B_GPIOD2MSK Register Diagram

Table 7-90: A2B_GPIOD2MSK Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
7:0 (R/W)	IOD2MSK	GPIO Over Distance IO2 Mask.

GPIO Over Distance Mask 3 Register

Address: 0x84

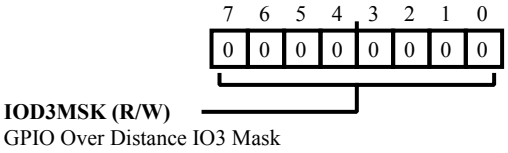

Figure 7-90: A2B_GPIOD3MSK Register Diagram

Table 7-91: A2B_GPIOD3MSK Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
7:0 (R/W)	IOD3MSK	GPIO Over Distance IO3 Mask.

GPIO Over Distance Mask 4 Register

Address: 0x85

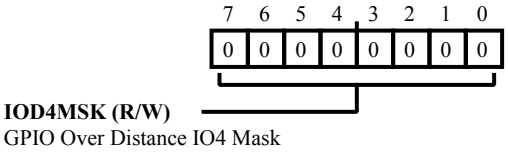

Figure 7-91: A2B_GPIOD4MSK Register Diagram

Table 7-92: A2B_GPIOD4MSK Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
7:0 (R/W)	IOD4MSK	GPIO Over Distance IO4 Mask.

GPIO Over Distance Mask 5 Register

Address: 0x86

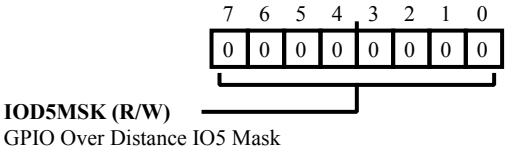

Figure 7-92: A2B_GPIOD5MSK Register Diagram

Table 7-93: A2B_GPIOD5MSK Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
7:0 (R/W)	IOD5MSK	GPIO Over Distance IO5 Mask.

GPIO Over Distance Mask 6 Register

Address: 0x87

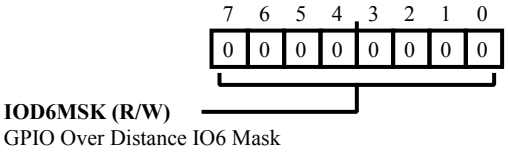

Figure 7-93: A2B_GPIOD6MSK Register Diagram

Table 7-94: A2B_GPIOD6MSK Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
7:0 (R/W)	IOD6MSK	GPIO Over Distance IO6 Mask.

GPIO Over Distance Mask 7 Register

Address: 0x88

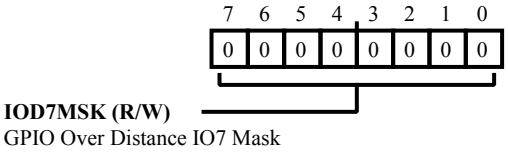

Figure 7-94: A2B_GPIOD7MSK Register Diagram

Table 7-95: A2B_GPIOD7MSK Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
7:0 (R/W)	IOD7MSK	GPIO Over Distance IO7 Mask.

GPIO Over Distance Data Register

Address: 0x89

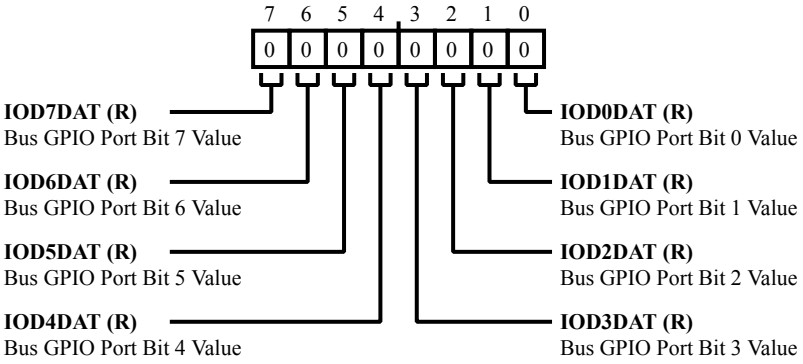

Figure 7-95: A2B_GPIODDAT Register Diagram

Table 7-96: A2B_GPIODDAT Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
7 (R/NW)	IOD7DAT	Bus GPIO Port Bit 7 Value.
6 (R/NW)	IOD6DAT	Bus GPIO Port Bit 6 Value.
5 (R/NW)	IOD5DAT	Bus GPIO Port Bit 5 Value.
4 (R/NW)	IOD4DAT	Bus GPIO Port Bit 4 Value.
3 (R/NW)	IOD3DAT	Bus GPIO Port Bit 3 Value.
2 (R/NW)	IOD2DAT	Bus GPIO Port Bit 2 Value.
1 (R/NW)	IOD1DAT	Bus GPIO Port Bit 1 Value.
0 (R/NW)	IOD0DAT	Bus GPIO Port Bit 0 Value.

GPIO Over Distance Invert Register

Address: 0x8A

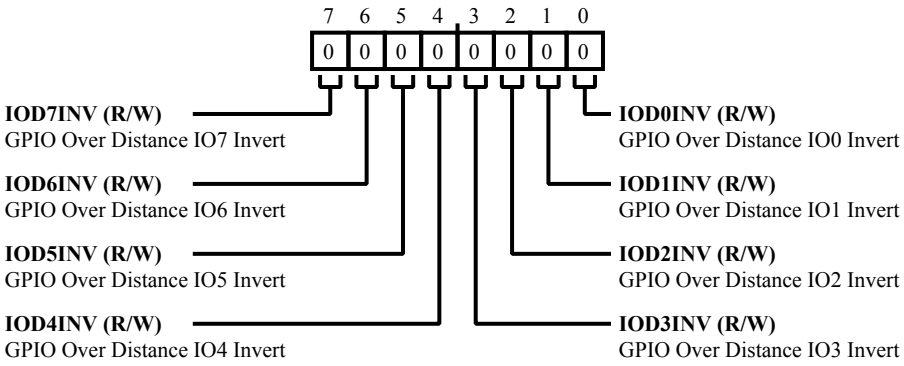

Figure 7-96: A2B_GPIODINV Register Diagram

Table 7-97: A2B_GPIODINV Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
7 (R/W)	IOD7INV	GPIO Over Distance IO7 Invert.
6 (R/W)	IOD6INV	GPIO Over Distance IO6 Invert.
5 (R/W)	IOD5INV	GPIO Over Distance IO5 Invert.
4 (R/W)	IOD4INV	GPIO Over Distance IO4 Invert.
3 (R/W)	IOD3INV	GPIO Over Distance IO3 Invert.
2 (R/W)	IOD2INV	GPIO Over Distance IO2 Invert.
1 (R/W)	IOD1INV	GPIO Over Distance IO1 Invert.
0 (R/W)	IOD0INV	GPIO Over Distance IO0 Invert.

Mailbox 0 Control Register (Slave Only)

The A2B_MBOX0CTL register contains bits that control direction, message length and interrupts.

Address: 0x90

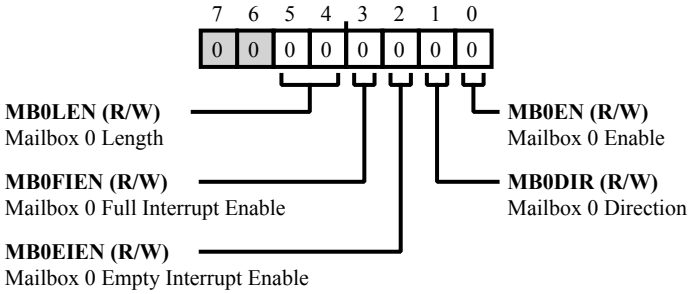

Figure 7-97: A2B_MBOX0CTL Register Diagram

Table 7-98: A2B_MBOX0CTL Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
5:4 (R/W)	MB0LEN	Mailbox 0 Length. The A2B_MBOX0CTL.MB0LEN bit field controls the length of Mailbox 0.	
		0	1 Byte
		1	2 Bytes
		2	3 Bytes
		3	4 Bytes
3 (R/W)	MB0FIEN	Mailbox 0 Full Interrupt Enable. The A2B_MBOX0CTL.MB0FIEN bit enables an interrupt which is generated when Mailbox 0 becomes full.	
		0	Mailbox 0 Interrupt on Full Disabled
		1	Mailbox 0 Interrupt on Full Enabled
2 (R/W)	MB0EIEN	Mailbox 0 Empty Interrupt Enable. The A2B_MBOX0CTL.MB0EIEN bit enables an interrupt which is generated when Mailbox 0 becomes empty.	
		0	Mailbox 0 Interrupt on Empty Disabled
		1	Mailbox 0 Interrupt on Empty Enabled
1 (R/W)	MB0DIR	Mailbox 0 Direction. The A2B_MBOX0CTL.MB0DIR bit controls the direction of Mailbox 0.	
		0	Mailbox 0 is Receive Mailbox
		1	Mailbox 0 is Transmit Mailbox

Table 7-98: A2B_MBOX0CTL Register Fields (Continued)

Bit No. (Access)	Bit Name	Description/Enumeration	
0 (R/W)	MB0EN	Mailbox 0 Enable. Setting the `A2B_MBOX0CTL.MB0EN` bit enables Mailbox 0.	
		0	Mailbox 0 Disabled
		1	Mailbox 0 Enabled

Mailbox 0 Status Register (Slave Only)

The A2B_MBOX0STAT register reports the status of the configured mailbox interrupts.

Address: 0x91

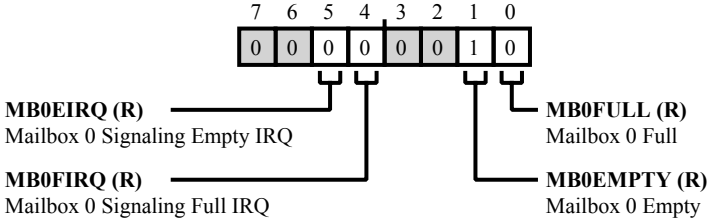

Figure 7-98: A2B_MBOX0STAT Register Diagram

Table 7-99: A2B_MBOX0STAT Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
5 (R/NW)	MB0EIRQ	Mailbox 0 Signaling Empty IRQ. The A2B_MBOX0STAT.MB0EIRQ bit indicates whether or not the Mailbox 0 empty interrupt is active.	
		0	Mailbox 0 Empty Interrupt Inactive
		1	Mailbox 0 Empty Interrupt Active
4 (R/NW)	MB0FIRQ	Mailbox 0 Signaling Full IRQ. The A2B_MBOX0STAT.MB0FIRQ bit indicates whether or not the Mailbox 0 full interrupt is active.	
		0	Mailbox 0 Full Interrupt Inactive
		1	Mailbox 0 Full Interrupt Active
1 (R/NW)	MB0EMPTY	Mailbox 0 Empty. The A2B_MBOX0STAT.MB0EMPTY bit indicates whether or not Mailbox 0 is empty.	
		0	Mailbox 0 Currently Not Empty
		1	Mailbox 0 Currently Empty
0 (R/NW)	MB0FULL	Mailbox 0 Full. The A2B_MBOX0STAT.MB0FULL bit indicates whether or not Mailbox 0 is full.	
		0	Mailbox 0 Currently Not Full
		1	Mailbox 0 Currently Full

Mailbox 0 Byte 0 Register (Slave Only)

Address: 0x92

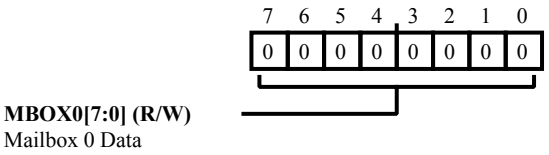

Figure 7-99: A2B_MBOX0B0 Register Diagram

Table 7-100: A2B_MBOX0B0 Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
7:0 (R/W)	MBOX0	Mailbox 0 Data.

Mailbox 0 Byte 1 Register (Slave Only)

Address: 0x93

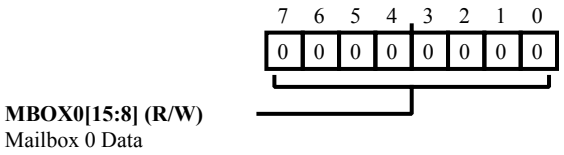

Figure 7-100: A2B_MBOX0B1 Register Diagram

Table 7-101: A2B_MBOX0B1 Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
7:0 (R/W)	MBOX0	Mailbox 0 Data.

Mailbox 0 Byte 2 Register (Slave Only)

Address: 0x94

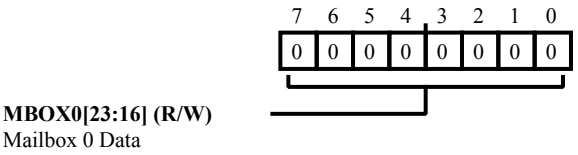

Figure 7-101: A2B_MBOX0B2 Register Diagram

Table 7-102: A2B_MBOX0B2 Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
7:0 (R/W)	MBOX0	Mailbox 0 Data.

Mailbox 0 Byte 3 Register (Slave Only)

Address: 0x95

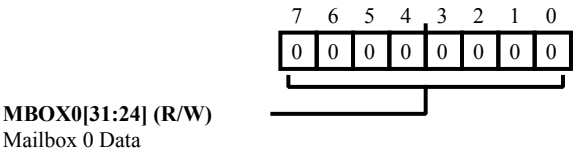

Figure 7-102: A2B_MBOX0B3 Register Diagram

Table 7-103: A2B_MBOX0B3 Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
7:0 (R/W)	MBOX0	Mailbox 0 Data.

Mailbox 1 Control Register (Slave Only)

The A2B_MBOX1CTL register contains bits that control direction, message length and interrupts.

Address: 0x96

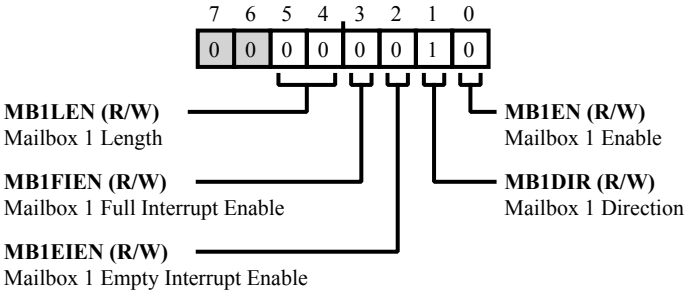

Figure 7-103: A2B_MBOX1CTL Register Diagram

Table 7-104: A2B_MBOX1CTL Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
5:4 (R/W)	MB1LEN	Mailbox 1 Length. The A2B_MBOX1CTL.MB1LEN bit field controls the length of Mailbox 1.	
		0	1 Byte
		1	2 Bytes
		2	3 Bytes
		3	4 Bytes
3 (R/W)	MB1FIEN	Mailbox 1 Full Interrupt Enable. The A2B_MBOX1CTL.MB1FIEN bit enables an interrupt which is generated when Mailbox 1 becomes full.	
		0	Mailbox 1 Interrupt on Full Disabled
		1	Mailbox 1 Interrupt on Full Enabled
2 (R/W)	MB1EIEN	Mailbox 1 Empty Interrupt Enable. The A2B_MBOX1CTL.MB1EIEN bit enables an interrupt which is generated when Mailbox 1 becomes empty.	
		0	Mailbox 1 Interrupt on Empty Disabled
		1	Mailbox 1 Interrupt on Empty Enabled
1 (R/W)	MB1DIR	Mailbox 1 Direction. The A2B_MBOX1CTL.MB1DIR bit controls the direction of Mailbox 1.	
		0	Mailbox 1 is Receive Mailbox
		1	Mailbox 1 is Transmit Mailbox

Table 7-104: A2B_MBOX1CTL Register Fields (Continued)

Bit No. (Access)	Bit Name	Description/Enumeration	
0 (R/W)	MB1EN	Mailbox 1 Enable. Setting the `A2B_MBOX1CTL.MB1EN` bit enables Mailbox 1.	
		0	Mailbox 1 Disabled
		1	Mailbox 1 Enabled

Mailbox 1 Status Register (Slave Only)

The A2B_MBOX1STAT register reports the status of the configured mailbox interrupts.

Address: 0x97

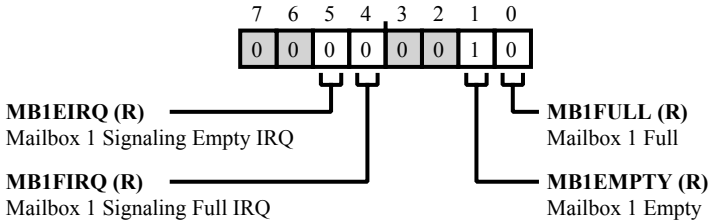

Figure 7-104: A2B_MBOX1STAT Register Diagram

Table 7-105: A2B_MBOX1STAT Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration	
5 (R/NW)	MB1EIRQ	Mailbox 1 Signaling Empty IRQ. The A2B_MBOX1STAT.MB1EIRQ bit indicates whether or not the Mailbox 1 empty interrupt is active.	
		0	Mailbox 1 Empty Interrupt Inactive
		1	Mailbox 1 Empty Interrupt Active
4 (R/NW)	MB1FIRQ	Mailbox 1 Signaling Full IRQ. The A2B_MBOX1STAT.MB1FIRQ bit indicates whether or not the Mailbox 1 full interrupt is active.	
		0	Mailbox 1 Full Interrupt Inactive
		1	Mailbox 1 Full Interrupt Active
1 (R/NW)	MB1EMPTY	Mailbox 1 Empty. The A2B_MBOX1STAT.MB1EMPTY bit indicates whether or not Mailbox 1 is empty.	
		0	Mailbox 1 Currently Not Empty
		1	Mailbox 1 Currently Empty
0 (R/NW)	MB1FULL	Mailbox 1 Full. The A2B_MBOX1STAT.MB1FULL bit indicates whether or not Mailbox 1 is full.	
		0	Mailbox 1 Currently Not Full
		1	Mailbox 1 Currently Full

Mailbox 1 Byte 0 Register (Slave Only)

Address: 0x98

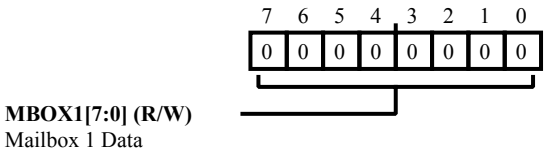

Figure 7-105: A2B_MBOX1B0 Register Diagram

Table 7-106: A2B_MBOX1B0 Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
7:0 (R/W)	MBOX1	Mailbox 1 Data.

Mailbox 1 Byte 1 Register (Slave Only)

Address: 0x99

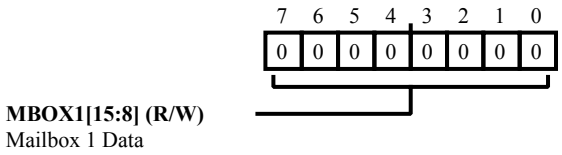

Figure 7-106: A2B_MBOX1B1 Register Diagram

Table 7-107: A2B_MBOX1B1 Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
7:0 (R/W)	MBOX1	Mailbox 1 Data.

Mailbox 1 Byte 2 Register (Slave Only)

Address: 0x9A

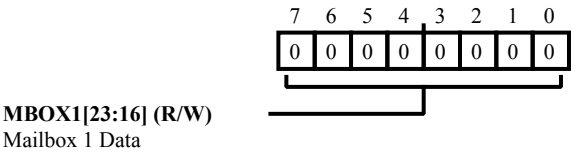

Figure 7-107: A2B_MBOX1B2 Register Diagram

Table 7-108: A2B_MBOX1B2 Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
7:0 (R/W)	MBOX1	Mailbox 1 Data.

Mailbox 1 Byte 3 Register (Slave Only)

Address: 0x9B

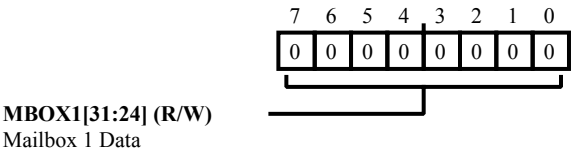

Figure 7-108: A2B_MBOX1B3 Register Diagram

Table 7-109: A2B_MBOX1B3 Register Fields

Bit No. (Access)	Bit Name	Description/Enumeration
7:0 (R/W)	MBOX1	Mailbox 1 Data.

8 Appendix A: Additional Discovery Flow Examples

The following sections provide additional information on modified, optimized, and advanced discovery flows. Any of the software flow diagrams can be used as a guide for discovery and initialization.

Modified Discovery Flow

In the ***Modified Discovery Flow*** figure, all of the slave nodes are discovered and immediately initialized, sequentially, from slave 0 to the last available slave in the system.

There is no further need for bus management after all nodes are discovered and programmed. But interrupt service routines may be used to react to special events (for example, an IRQ event from diagnosis). The IRQ pin can be used to signal such an event. Alternatively, the `A2B_INTTYPE` register can be polled to monitor interrupt events.

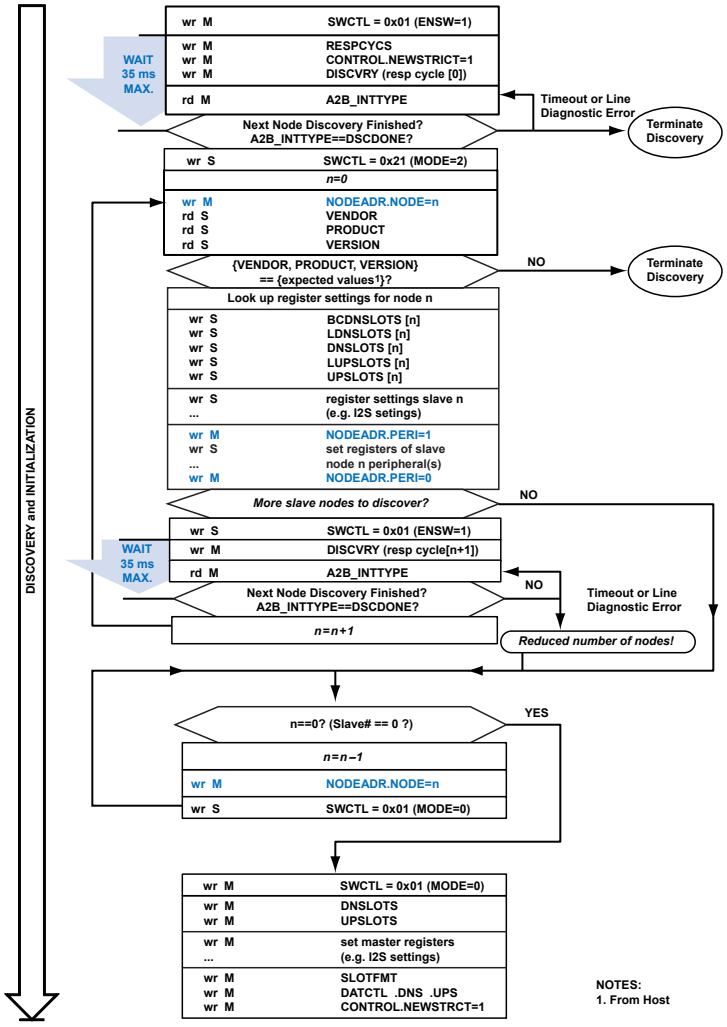

Figure 8-1: Modified Discovery Flow

Optimized Discovery Flow

A more optimized, fast discovery and initialization are shown in the ***Optimized Discovery Flow*** figure. Even before a node is initialized, the host tries to discover the next node. The time for the next node to be discovered is used to initialize the current node. This reduces the discovery and initialization time almost completely to the time it takes for the PLLs to find lock. Interrupt service routines are used to avoid repeated polling of registers, reducing the burden on the host processor.

There is no further need for bus management after all nodes are discovered and initialized. Interrupt service routines can be used to react to special events (for example, an IRQ event from diagnosis).

An advanced feature in the flow diagram is the use of node IDs. Node IDs allow the host to look up register settings based on IDs stored in each slave node's EEPROM.

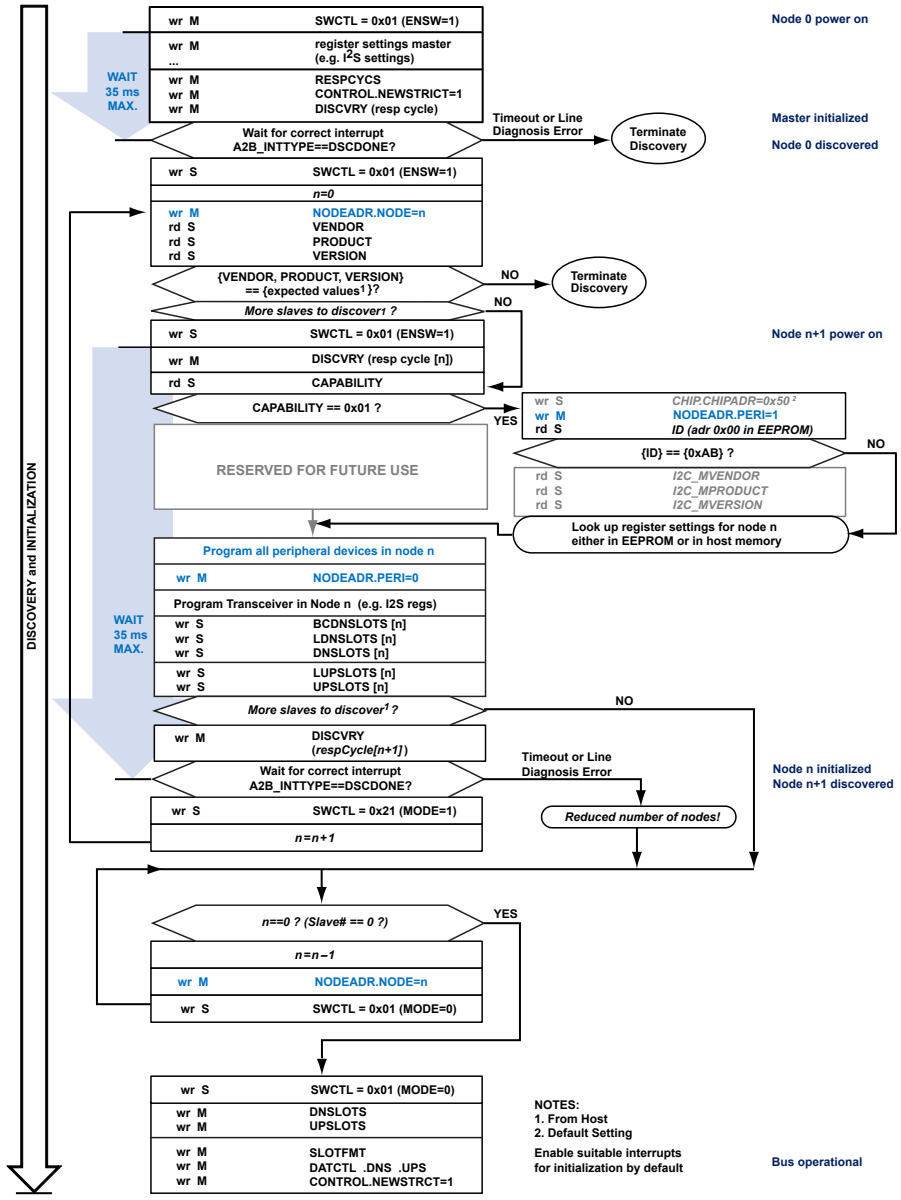

Figure 8-2: Optimized Discovery Flow

Advanced Discovery Flow

An advanced, fast flow of discovery and initialization is shown in the ***Advanced Discovery Flow*** figure. Even before a node is initialized, the host tries to discover the next node. The time for the next node to be discovered is used to initialize the current node. This reduces the discovery and initialization time almost completely to the time it takes for the PLLs to find lock. Synchronous exchange of data can start as soon as a master and slave 0 node are initialized, while the next nodes that are not discovered and initialized can start up gradually. Use interrupt service routines to avoid repeated polling of registers, which reduces the burden on the host processor.

Another advanced feature in this flow diagram is the use of node IDs. Node IDs allow the host to look up register settings based on IDs stored in EEPROM of each slave node.

The slave nodes are reconfigured with the addition of every new node to adjust the amount of payload and, therefore, optimize bandwidth and power consumption. The optimum bus activity level is achieved with every addition of a new node even when not of the all nodes can be discovered.

This is especially advantageous when a host tries to perform “auto-discovery” without prior knowledge of the number of nodes in the system. The A2B_DNSLOTS and A2B_UPSLOTS register values can be calculated based on the A2B_BCDNSLOTS, A2B_LDNSLOTS, and A2B_LUPSLOTS information in each node. This can be part of the node ID capability information (for example, in the EEPROM of each slave node) or can be looked up based on the capability information.

Changing A2B_DNSLOTS and A2B_UPSLOTS in all nodes, depending on the number of nodes discovered, has an impact on the master's I^2S/TDM interface. The channel allocation changes when a new node that provides or consumes synchronous data is added.

Allowing synchronous payload operation on early nodes before the bus is fully discovered may or may not be desirable. The advanced discovery flow can be modified so that synchronous audio operation only starts after discovery (see Optimized Discovery Flow).

There is no further need for bus management after all of the nodes are discovered and initialized. Interrupt service routines can be used to react to special events (for example, an IRQ event from Diagnosis).

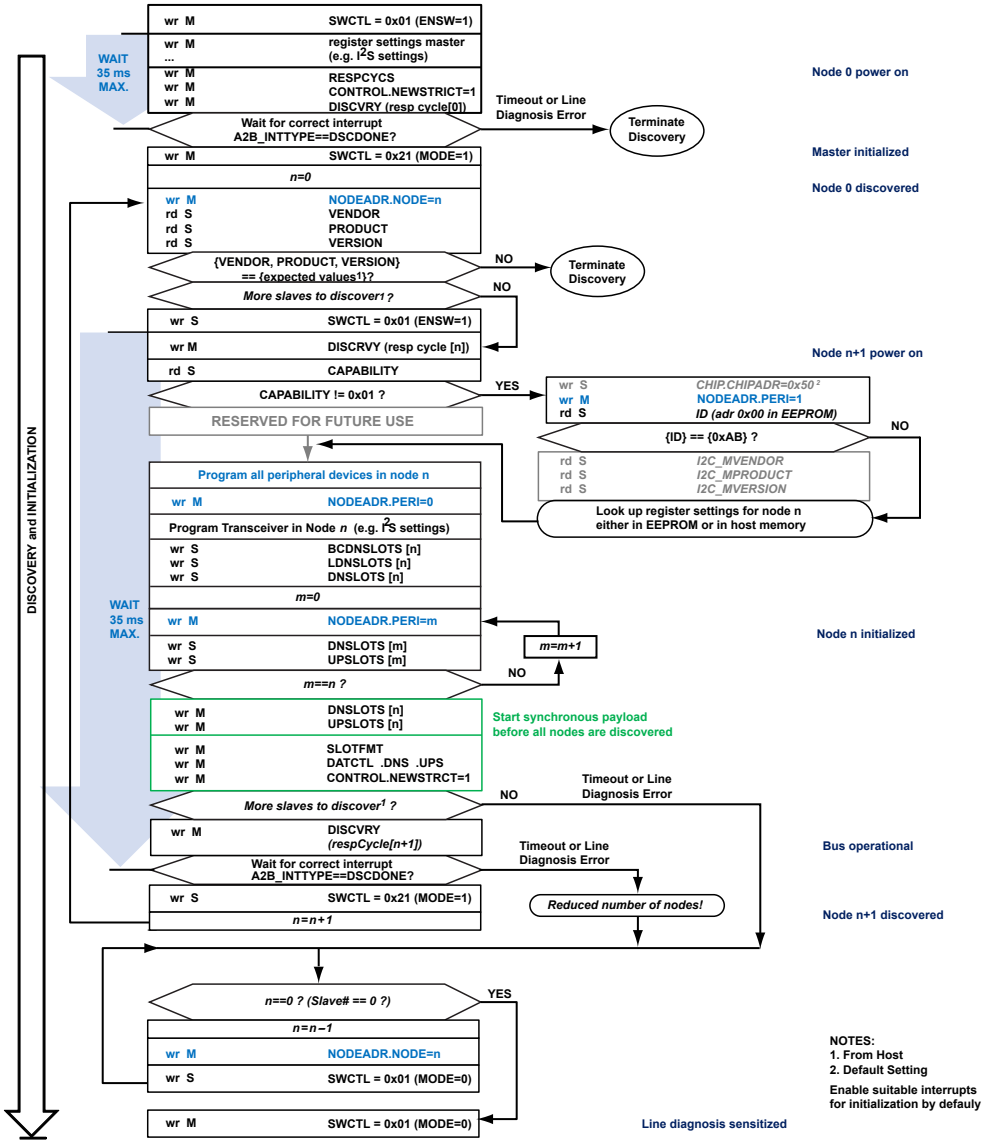

Figure 8-3: Advanced Discovery Flow

9 Appendix B: Response Cycle Formula

The A2B_RESPCYCS register is used to set the relative time, from the start of a control frame (SCF) to the moment the last slave responds with a response frame (SRF). The register setting defines when earlier nodes in the A²B network should expect the response from the last slave during the upstream portion of the superframe. If the last node fails to respond, the node immediately before the presumed last node does respond. The following sections provide information regarding how to program the master node and slave node A2B_RESPCYCS registers.

Configuring Master Node Response Cycles

The *Master Node Response Cycles* figure depicts how the master response cycle value is determined.

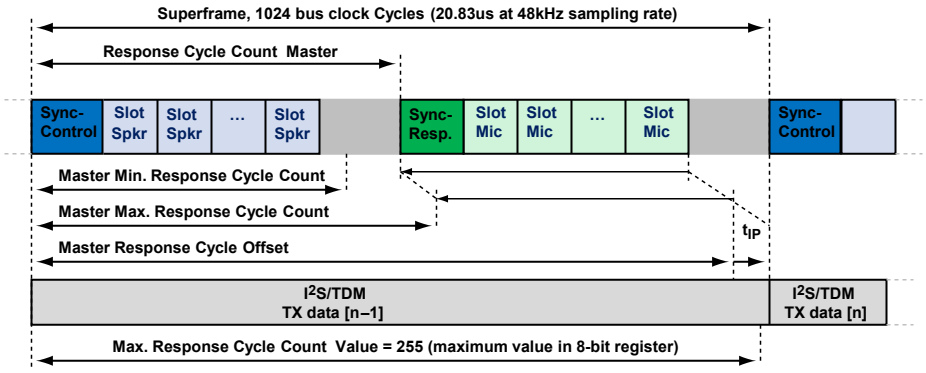

Figure 9-1: Master Node Response Cycles

In the *Master Node Response Cycles* figure:

- The ***Master Minimum Response Cycle Count*** is determined by the length of the downstream data, the minimum bus turn-around time, and the number of slaves nodes.
- The ***Master Maximum Response Cycle Count*** is determined by the length of the upstream data and the ***Master Response Cycle Offset***.
- The ***Master Response Cycle Offset*** ensures that sufficient internal processing time (t_{IP}) is available from the reception of the last upstream data bit in the receive buffer to the point at which this I²S/TDM data is output, which starts synchronous to the next SCF and SYNC pin transition. The ***A²B Master Node Response Offset (RESPOFFS)*** table defines this constant ***Master Response Cycle Offset***, which is a function of the A²B master node's TDM mode (A2B_I2SGCFG.TDMMODE) and I²S/TDM channel size (A2B_I2SGCFG.TDMSS).

Table 9-1: A²B Master Node Response Offset (RESPOFFS)

TDM Mode (A²B Master Node)	TDM Data Width (A²B Master Node)	RESPOFFS
TDM2/I²S (A2B_I2SGCFG.TDMMODE = 0)	16 bits (A2B_I2SGCFG.TDMSS = 1)	238
TDM2/I²S (A2B_I2SGCFG.TDMMODE = 0)	32 bits (A2B_I2SGCFG.TDMSS = 0)	245
TDM4 (A2B_I2SGCFG.TDMMODE = 1)	16 bits (A2B_I2SGCFG.TDMSS = 1)	245
TDM4 (A2B_I2SGCFG.TDMMODE = 1)	32 bits (A2B_I2SGCFG.TDMSS = 0)	248
TDM8 (A2B_I2SGCFG.TDMMODE = 2)	16 bits (A2B_I2SGCFG.TDMSS = 1)	248
TDM8 (A2B_I2SGCFG.TDMMODE = 2)	32 bits (A2B_I2SGCFG.TDMSS = 0)	248
TDM12 (A2B_I2SGCFG.TDMMODE = 3)	16 bits (A2B_I2SGCFG.TDMSS = 1)	248
TDM12 (A2B_I2SGCFG.TDMMODE = 3)	32 bits (A2B_I2SGCFG.TDMSS = 0)	248
TDM16 (A2B_I2SGCFG.TDMMODE = 4)	16 bits (A2B_I2SGCFG.TDMSS = 1)	248
TDM16 (A2B_I2SGCFG.TDMMODE = 4)	32 bits (A2B_I2SGCFG.TDMSS = 0)	248
TDM20 (A2B_I2SGCFG.TDMMODE = 5)	N/A	248
TDM24 (A2B_I2SGCFG.TDMMODE = 6)	N/A	248
TDM32 (A2B_I2SGCFG.TDMMODE = 7)	N/A	248

Programming the master node A2B_RESPCYCS register is a function of the above ***Master Response Cycle Offset*** (RESPOFFS), as well as:

- the number of slave nodes in the system,
- the number of downstream A²B bus data slots received on the A-PORT at each slave (NUM_DNSLOTS),
- the width of the downstream A²B bus data slots (DNSLOT_SIZE),
- the number of upstream A²B bus data slots driven to the A-PORT by each slave (NUM_UPSLOTS), and
- the width of the upstream A²B bus data slots (UPSLOT_SIZE).

The upslot and downslot activity that is possible at any given node in the system is the first factor that contributes toward determining the value that must be programmed into the master node's A2B_RESPCYCS register. For each slave node n in the A²B topology, the following equations define the downstream (DNSLOT_ACTIVITY[n]) and upstream (UPSLOT_ACTIVITY[n]) activity for that node.

```
DNSLOT_ACTIVITY[n] = NUM_DNSLOTS * (DNSLOT_SIZE + 1)
UPSLOT_ACTIVITY[n] = NUM_UPSLOTS * (UPSLOT_SIZE + 1)
```

NOTE: The DNSLOT_SIZE and UPSLOT_SIZE slot sizes are offset by 1 in the above calculations because the default slot format (A2B_SLOTFMT) appends a single parity bit to each data slot on the A²B bus, thereby increasing the number of bits on the A²B bus per slot by 1. For alternate slot formats, the number of bits that are appended for the chosen use case must be added instead of the 1 defined here, as presented in the A²B Bus Bits column in the Slot Format table in A²B Slot Format.

Once the upslot and downslot activity for each slave node n is established, the equivalent upstream (RESPCYCS_UP[n]) and downstream (RESPCYCS_DN[n]) response cycle requirements can be calculated for each slave node, as governed by the following equations.

```
RESPCYCS_DN[n] = ((64 + DNSLOT_ACTIVITY[n])/4) + 4n + 2            // Round Up
RESPCYCS_UP[n] = RESPOFFS - (((64 + UPSLOT_ACTIVITY[n])/4) + 1)    // Round Up
```

- RESPCYCS_DN[n] is the minimum response cycle register setting possible at the master node when considering the downstream activity at slave node n. The maximum value among those calculated for RESPCYCS_DN[n] is the minimum master node A2B_RESPCYCS setting (MAX(RESPCYCS_DN[n])).
- RESPCYCS_UP[n] is the maximum response cycle register setting possible at the master node when considering the upstream activity at slave node n. The minimum value among those calculated for RESPCYCS_UP[n] is the maximum master node A2B_RESPCYCS setting (MIN(RESPCYCS_UP[n])).

CAUTION: If MAX(RESPCYCS_DN[n]) > MIN(RESPCYCS_UP[n]), then the A^{2}B bus bandwidth cannot accommodate the configuration.

The value that must be programmed into the master node's A2B_RESPCYCS register is the average of these minimum and maximum values:

```
A2B_RESPCYCS = (MAX(RESPCYCS_DN[n]) + MIN(RESPCYCS_UP[n])) / 2   // Round Down
```

Example Master A2B_RESPCYCS Calculation

A system with three nodes, the master node and two slave nodes (slave 0 and slave 1), is configured as shown in the *Three-Node A^{2}B System Example* figure:

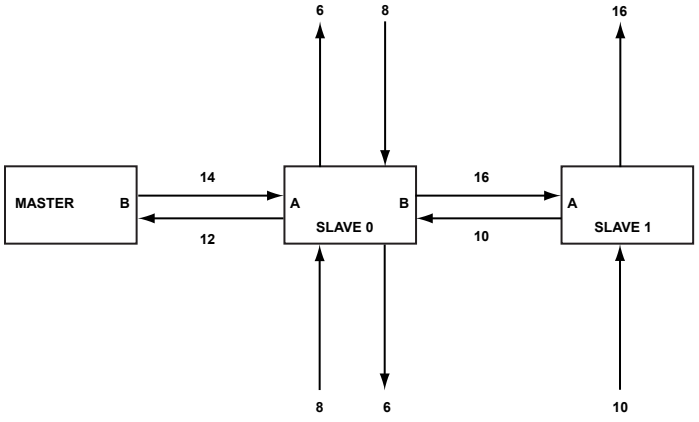

Figure 9-2: Three-Node A^{2}B System Example

For the downstream portion of the superframe:

- Master node (configured for 32-bit TDM8 mode): sends 14 slots with a 24-bit slot size
- Slave 0: consumes six slots from the master node and passes the remaining eight slots to slave 1, then contributes eight additional slots to the downstream traffic (16 total slots sent from slave 0 to slave 1)

- Slave 1: consumes all 16 slots coming from slave-0

For the upstream portion of the superframe:

- Slave 1: sends ten slots with a 16-bit slot size
- Slave 0: consumes six slots from slave-1 and passes the remaining four slots to the master node, then contributes eight additional slots to the upstream traffic (12 total slots sent from slave 0 to the master node).
- Master node (configured for 32-bit TDM8 mode): consumes all 12 slots coming from slave 0.

The response cycle is determined using the following steps:

1. Calculate the upslot and downslot activity for each slave node:

```
DNSLOT_ACTIVITY[n] = NUM_DNSLOTS * (DNSLOT_SIZE + 1)
DNSLOT_ACTIVITY[0] = 14 * (24 + 1) = 350
DNSLOT_ACTIVITY[1] = 16 * (24 + 1) = 400

UPSLOT_ACTIVITY[n] = NUM_UPSLOTS * (UPSLOT_SIZE + 1)
UPSLOT_ACTIVITY[0] = 12 * (16 + 1) = 204
UPSLOT_ACTIVITY[1] = 10 * (16 + 1) = 170
```

2. With this information, calculate the response cycle requirements for each slave. From the ***A²B Master Node Response Offset (RESPOFFS)*** table, the TDM8 mode and 32-bit data combination yields RESPOFFS = 248.

```
RESPCYCS_DN[n] = ((64 + DNSLOT_ACTIVITY[n])/4) + 4n + 2            // Round Up
RESPCYCS_DN[0] = ((64 + 350)/4) + (4*0) + 2 = 103.5 + 0 + 2 = 105.5     = 106
RESPCYCS_DN[1] = ((64 + 400)/4) + (4*1) + 2 = 116.0 + 4 + 2 = 122.0     = 122

RESPCYCS_UP[n] = RESPOFFS - (((64 + UPSLOT_ACTIVITY[n])/4) + 1)    // Round Up
RESPCYCS_UP[0] = 248 - (((64 + 204)/4) + 1) = 248 - (67.0 + 1) = 180.0  = 180
RESPCYCS_UP[1] = 248 - (((64 + 170)/4) + 1) = 248 - (58.5 + 1) = 188.5  = 189
```

   The minimum master node A2B_RESPCYCS setting is the maximum value among the RESPCYCS_DN[n] calculations, which is 122, and the maximum setting is the minimum value among the RESPCYCS_UP[n] calculations, which is 180, and the average of the minimum and maximum values is:

```
(MAX(RESPCYCS_DN[n]) + MIN(RESPCYCS_UP[n])) / 2   // Round Down
(122 + 180) / 2 = 302 / 2 = 151.0                     = 151
```

3. For this system configuration, program the master node A2B_RESPCYCS value to 151 (0x97).

Configuring Slave Node Response Cycles

Each slave node has its A2B_RESPCYCS register set during the system discovery process. The master transceiver programs its A2B_DISCVRY register with the response cycle value associated with the slave transceiver that it is attempting to discover. The appropriate value for each slave node (SLV_RESPCYCS[n]) is a function of the slave node's location in the A²B topology and the value programmed to the master node 's A2B_RESPCYCS register (MSTR_RESPCYCS). The slave node nearest to the master node has a node number of 0, and the node number is

incremented for each next-in-line slave node until the last-in-line slave node n. The A2B_RESPCYCS value to use for each slave node during discovery can be calculated using the following equation:

```
SLV_RESPCYCS[n] = MSTR_RESPCYCS - 4n
```

Using the Example Master A2B_RESPCYCS Calculation above (with MSTR_RESPCYCS = 151), the following equations determine the correct A2B_RESPCYCS value for the two slave nodes:

```
SLV_RESPCYCS[0] = MSTR_RESPCYCS - (4*0) = 151 - 0 = 151 (0x97)
SLV_RESPCYCS[1] = MSTR_RESPCYCS - (4*1) = 151 - 4 = 147 (0x93)
```

The following code sequence uses these values to proceed through the discovery process in the example system:

```
Write MSTR_RESPCYCS to the A2B_RESPCYCS register in the master node
Write 0x01 to the A2B_CONTROL register in the master node
Write 0x01 to the A2B_SWCTL register in the master node
Write 0x01 to the A2B_INTMSK2 register in the master node
Write SLV_RESPCYCS[0] to the A2B_DISCVRY register in the master node
     <Wait for Interrupt>

Write 0x00 to the A2B_NODEADR register in the master node
Write 0x01 to the A2B_SWCTL register in slave node 0
Write SLV_RESPCYCS[1] to the A2B_DISCVRY register in the master node
     <Wait for Interrupt>
```

Configuring Slave Node Response Cycles with Advanced Discovery

The transceiver is designed to adjust automatically to the time when responses are seen on the bus. This allows slave nodes to be discovered without requiring changes to the slave node response cycles, based on cable length. When the advanced discovery flow is used, it is possible for transient data parity errors to be reported by the master node immediately, following the discovery of a new slave, which is connected by a long cable (greater than 5m in length). These errors only persist for two to three superframes. If the cable lengths in the system are known, avoid these errors by using the following pseudo-code to calculate the slave node response cycles:

```
if (n = 0)
   SLV_RESPCYCS[n] = MSTR_RESPCYCS
else
   if (cable_length > 12m)
     SLV_RESPCYCS[n] = SLV_RESPCYCS[n-1] - 6
   else if (cable_length > 5m)
     SLV_RESPCYCS[n] = SLV_RESPCYCS[n-1] - 5
   else
     SLV_RESPCYCS[n] = SLV_RESPCYCS[n-1] - 4
```

10 Appendix C: Module ID and Module Configuration Memory

Module-specific descriptor information is saved in a storage device (EEPROM or similar), directly connected to an A^2B transceiver via I^2C and accessible over the A^2B bus as a peripheral device. Such I^2C-connected storage devices use the device address 0x50 (7-bit). This configuration memory contains module ID information and optional configuration blocks.

Configuration Memory

The contents of a configuration memory with no configuration blocks is shown in the ***Memory Content with no Configuration Blocks*** table.

ADDRESS	CONTENTS
0x0000	0xAB (Indicates Configuration Memory)
0x0001	Module Vendor ID*
0x0002	Module Product ID
0x0003	Module Version ID
0x0004	Reserved - value should be ignored
0x0005	0x00 (Number of Configuration Blocks)
0x0006	Reserved - value should be 0x00
0x0007	CRC-8

*Assignment and management of Module Vendor IDs currently resides with Analog Devices Inc.

Figure 10-1: Memory Content with no Configuration Blocks

During and after discovery, the host can uniquely identify the slave node modules, based on the convention in the table. This information allows the host to look up all stored configuration settings and software drivers to automatically configure the A^2B system, program A^2B nodes, and initialize peripheral devices. A CRC byte is used to ensure data integrity.

Additionally, device specific configuration and setup information also can be stored in the configuration memory through the use of configuration blocks. The host can read this information and set up the slave without any prior knowledge of the node. The contents of a configuration memory with configuration blocks is shown in the ***Memory Content with Configuration Blocks*** table.

ADDRESS	CONTENTS
0x0000	0xAB (Indicates Configuration Memory)
0x0001	Module Vendor ID
0x0002	Module Product ID
0x0003	Module Version ID
0x0004	Reserved - value should be ignored
0x0005	Number of Configuration Blocks
0x0006	Reserved - value should be 0x00
0x0007	CRC-8
0x0008 to $7 + L_1$	Configuration Block 1
$8 + L_1$ to $7 + L_1 + L_2$	Configuration Block 2
	⋮
	Configuration Block N

Figure 10-2: Memory Content with Configuration Blocks

The contents of a configuration block are shown in the ***Configuration Block Contents*** figure.

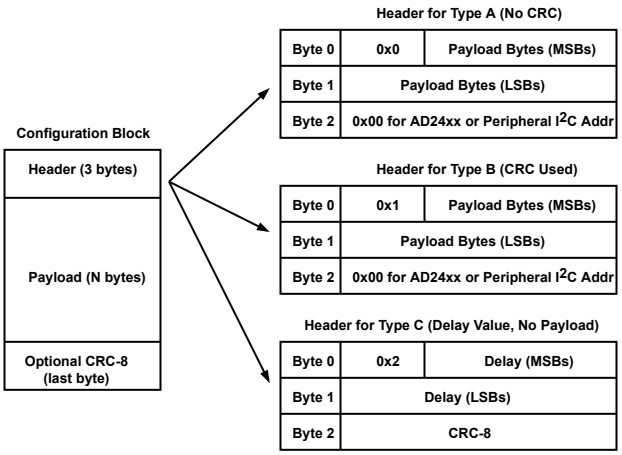

Figure 10-3: Configuration Block Contents

The first three bytes of a configuration block make up a header, which provides details about the configuration block. The first four bits of the header indicate the type of configuration block; see the ***Configuration Block Header Types*** table. Types A and B have a 12-bit field that gives the size (in bytes) of the payload. For a Type A configuration block, this field contains the number of bytes to be written during configuration. For a Type B configuration block, the value of this field is one more than the number of bytes to be written during configuration, because an 8-bit CRC is included at the end of the payload.

If the device to be programmed requires an address pointer, it is given at the start of the payload field. A Type C configuration block has a 12-bit field, which describes a delay to be inserted in the programming flow (in ms).

Table 10-1: Configuration Block Header Types

Type Value	Meaning	Notes
0x0	Type A config block, no CRC	All payload bytes written to target for configuration
0x1	Type B config block CRC-8 calculated on header + payload	Last payload byte not written to target
0x2	Type C config block delay value only (no payload)	CRC-8 calculated based on first 2 bytes in header
0x3 - 0xF	Reserved	N/A

The ***Detailed View of Configuration Memory*** figure shows a detailed view of configuration memory contents with N configuration blocks.

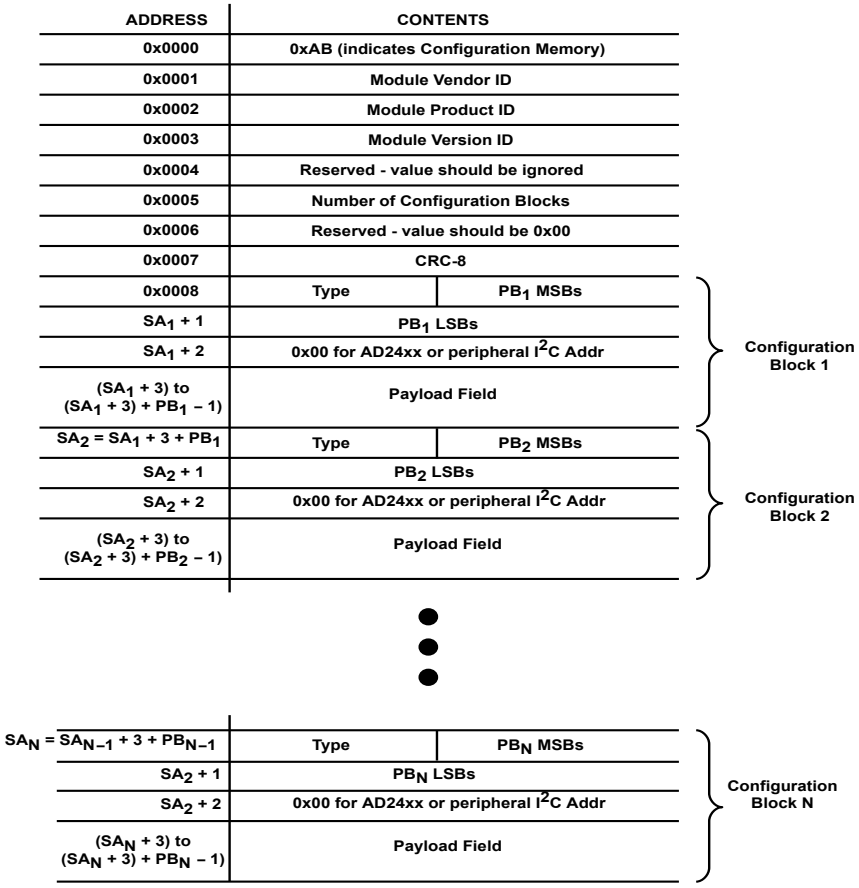

Figure 10-4: Detailed View of Configuration Memory

Notes:

- If address 0x0005 returns 0x00, no configuration blocks are present in the memory.
- PB_N is the number of bytes in the payload for a configuration block N (a 12-bit field).
- SA_N is the start address of a configuration block N. $SA_N = SA_{N-1} + 3 + PB_{N-1}$. $SA_1 = 8$.

The following tables show two examples of configuration memory containing programming information for an A^{2}B slave node. The first byte of each payload field is a starting address for a burst write.

	ADDRESS	CONTENTS		
	0x0000	0xAB		
	0x0001	Module Vendor ID		
	0x0002	Module Product ID		
	0x0003	Module Version ID		
	0x0004	Reserved		
	0x0005	0x02 (Number of Configuration Blocks)		
	0x0006	Reserved - value should be 0x00		
	0x0007	CRC-8		
SA_1	0x0008	0x00 (Type A, PB_1 MSBs = 0)		Configuration Block
	0x0009	0x04 (PB_1 LSBs = 4)		
	0x000A	0x00 (For AD24xx programming)		
	0x000B	0x0A (Address pointer for BCDNSLOTS)	Payload	
	0x000C	Data for BCDNSLOTS		
	0x000D	Data for LDNSLOTS		
	0x000E	Data for LUPSLOTS		
SA_2	0x000F	0x00 (Type A, PB_2 MSBs = 0)		Configuration Block2
	0x0010	0x15 (PB_2 LSBs = 21)		
	0x0011	0x00 (For AD24xx programming)		
	0x0012	0x3F (Address pointer for I2CCFG)	Payload	
	0x0013	Data for I2CCFG		
	0x0014	0x00 (Data for PLLCTL)		
	0x0015	Data for I2SGCFG		
	0x0016	Data for I2SCFG		
	0x0017	Data for I2SRATE		
	0x0018	Reserved (ADDR 0x44 reserved for slave)		
	0x0019	Reserved (ADDR 0x45 reserved for slave)		
	0x001A	Data for SYNCOFFSET		
	0x001B	Data for PDMCTL		
	0x001C	Data for ERRMGMT		
	0x001D	0x00 (addr 0x49 reserved) for AD242x		
	0x001E	Data for GPIODAT		
	0x001F	Data for GPIODATSET		
	0x0020	Data for GPIODATCLR		
	0x0021	Data for GPIOOEN		
	0x0022	Data for GPIOIEN		
	0x0023	0x00 (Data for GPIOIN)		
	0x0024	Data for PINTEN		
	0x0025	Data for PINTINV		
	0x0026	Data for PINCFG		

Figure 10-5: Configuration Memory for AD242x Slave Configuration (Long)

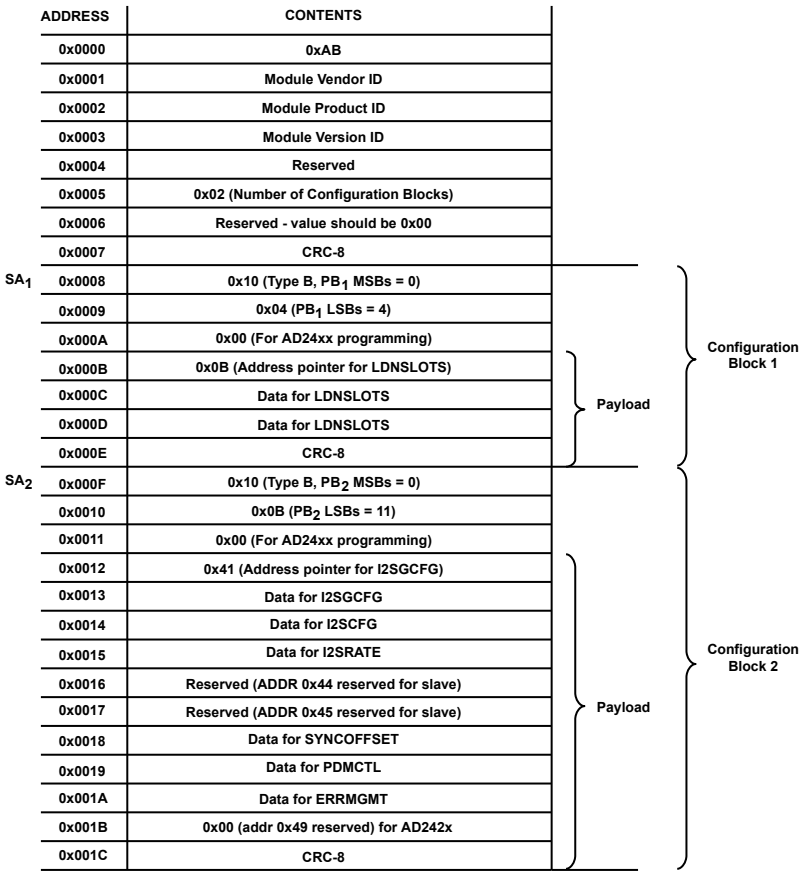

Figure 10-6: Configuration Memory for AD242x Slave Configuration (Short)

The ***Configuration Memory for ADAU1761*** figure shows an example of configuration memory containing programming information for an ADAU1761 codec (which uses two address bytes per transaction).

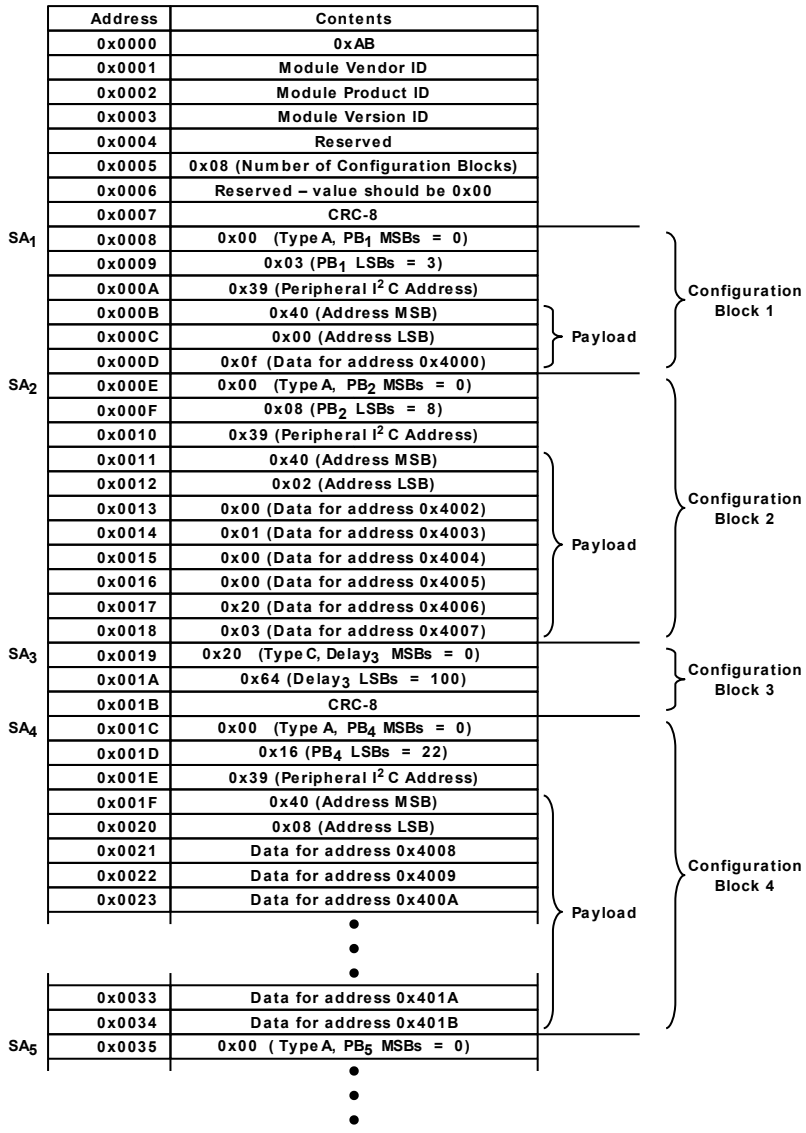

Figure 10-7: Configuration Memory for ADAU1761

11 Appendix D: Interrupt Processing

The following sections describe the flow of interrupt processing by the host in the A^{2}B system.

Master Running Interrupts

As shown in the ***Master Running Interrupts*** figure, the trigger (Master IRQ pin) is asserted after the master node locks its PLL to the SYNC signal or on a post discovery line fault.

NOTE: `MSTR_RUNNING` (`A2B_INTTYPE`= 0xFF) is a master-only interrupt.

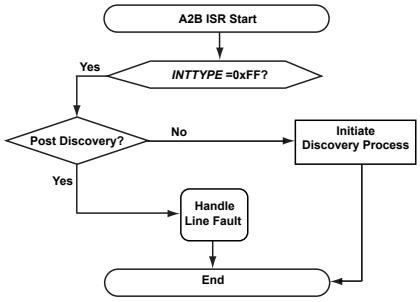

Figure 11-1: Master Running Interrupts

Action: read the `A2B_INTSRC` and `A2B_INTTYPE` registers and proceed to slave node discovery or handle a line fault. Note that a host read of the master `A2B_INTTYPE` register clears the interrupt.

Discovery Done Interrupts

As shown in the ***Discovery Done Interrupts*** figure, the trigger (Master IRQ pin) is asserted after the master node sees a response from discovery of a slave node. `DSCDONE` (`A2B_INTTYPE`= 0x18) is a master-only interrupt.

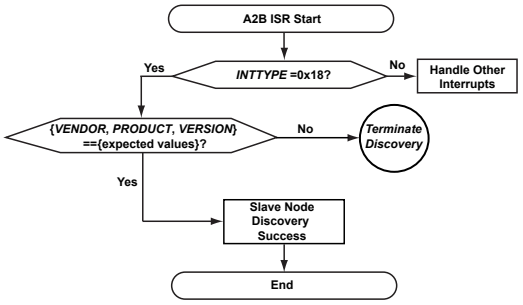

Figure 11-2: Discovery Done Interrupts

Action: read the A2B_INTSRC and A2B_INTTYPE registers and proceed to the node authentication and discovery process. Note that a host read of the master A2B_INTTYPE register clears the interrupt.

Line Fault Interrupts

As shown in the ***Line Fault Interrupts*** figure, the trigger (Master IRQ pin) is asserted after encountering a line fault during or post discovery.

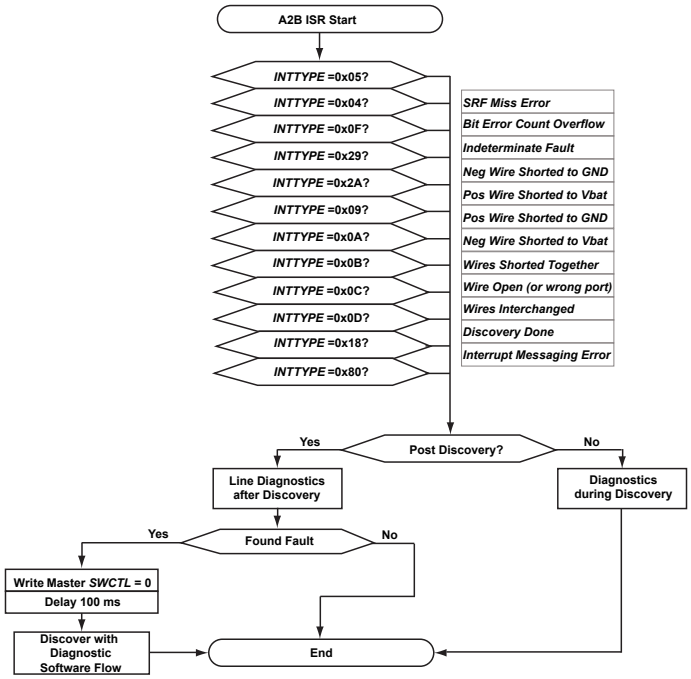

Figure 11-3: Line Fault Interrupts

Action: read the A2B_INTTYPE register and proceed with line diagnostics, as described in A²B System Debug.

When the transceiver enters the RESET state due to critical faults such as BP short to GND, there is no indication to the host that this has occurred. If such functionality is desired in the system, designs can utilize termination resistors on the IRQ line as a function of its active polarity, as governed by the A2B_PINCFG.IRQINV bit. When A2B_PINCFG.IRQINV = 0, a pull-up resistor connected to the IRQ line pulls IRQ high when the transceiver

tristates the IRQ pin while in the RESET state. The state can then be seen by the host controller as an active high edge pseudo-interrupt. The host reads the A2B_INTSTAT and A2B_INTTYPE registers as 0x00 (reset values), which can be interpreted as an event indicating that the transceiver has entered the RESET state. A pull-down resistor on the IRQ line has the same effect when A2B_PINCFG.IRQINV=1 for negative edge interrupts at the host controller.

NOTE: The host controller must ignore the IRQ state prior to the A2B_CONTROL.MSTR bit being set, upon which the IRQ pin is driven to the inactive state.

Error Interrupts

As shown in the ***Error Interrupts*** figure, the trigger (Master IRQ pin) is asserted when any of the following errors is encountered.

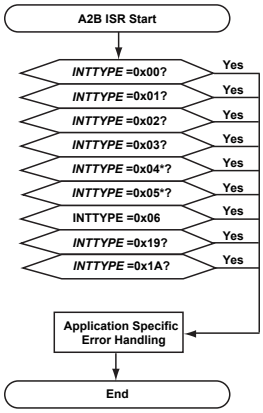

Figure 11-4: Error Interrupts

```
HDCNTERR=0x0
DDERR=   0x1
CRCERR=  0x2
DPERR=   0x3
BECOVF=  0x4* (Occurrence of Bit error count overflow interrupt, after
               resetting the error counter (BECNT) once every
               second, indicates bus issues )
SRFERR=  0x5* (10 time occurrence without interrupt status (INTSTAT)
               being cleared between pending interrupts shall be
               treated as bus lost condition/line fault)
SRFCRCERR=0x6 (Slave Only)
I2CERR=   0x19 (Master Only)
ICRCERR=  0x1A (Master Only)
```

Action: read the A2B_INTTYPE register and proceed with line diagnostics, as described in A^{2}B System Debug.

General Purpose IO Pin Interrupts

As shown in the ***General Purpose IO Pin Interrupts*** figure, the trigger (Master IRQ pin) is asserted when any of the following errors is encountered.

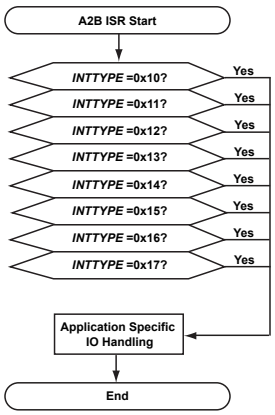

Figure 11-5: General Purpose IO Pin Interrupts

```
IO0=  0x10  (Slave only)
IO1=  0x11
IO2=  0x12
IO3=  0x13
IO4=  0x14
IO5=  0x15
IO6=  0x16
IO7=  0x17
```

Action: read the A2B_INTTYPE register and take action that is specific to the application.