CIRRUS LOGIC

CS4953xx
32-bit Audio DSP Family

CS4953xx

Hardware User's Manual

Preliminary Product Information

This document contains information for a new product.
Cirrus Logic reserves the right to modify this product without notice.

FEB '10
DS732UM10

Contacting Cirrus Logic Support

For all product questions and inquiries contact a Cirrus Logic Sales Representative.
To find the one nearest to you go to **www.cirrus.com**

IMPORTANT NOTICE

"Preliminary" product information describes products that are in production, but for which full characterization data is not yet available.

Cirrus Logic, Inc. and its subsidiaries ("Cirrus") believe that the information contained in this document is accurate and reliable. However, the information is subject to change without notice and is provided "AS IS" without warranty of any kind (express or implied). Customers are advised to obtain the latest version of relevant information to verify, before placing orders, that information being relied on is current and complete. All products are sold subject to the terms and conditions of sale supplied at the time of order acknowledgment, including those pertaining to warranty, indemnification, and limitation of liability. No responsibility is assumed by Cirrus for the use of this information, including use of this information as the basis for manufacture or sale of any items, or for infringement of patents or other rights of third parties. This document is the property of Cirrus and by furnishing this information, Cirrus grants no license, express or implied under any patents, mask work rights, copyrights, trademarks, trade secrets or other intellectual property rights. Cirrus owns the copyrights associated with the information contained herein and gives consent for copies to be made of the information only for use within your organization with respect to Cirrus integrated circuits or other products of Cirrus. This consent does not extend to other copying such as copying for general distribution, advertising or promotional purposes, or for creating any work for resale.

CERTAIN APPLICATIONS USING SEMICONDUCTOR PRODUCTS MAY INVOLVE POTENTIAL RISKS OF DEATH, PERSONAL INJURY, OR SEVERE PROPERTY OR ENVIRONMENTAL DAMAGE ("CRITICAL APPLICATIONS"). CIRRUS PRODUCTS ARE NOT DESIGNED, AUTHORIZED OR WARRANTED FOR USE IN PRODUCTS SURGICALLY IMPLANTED INTO THE BODY, AUTOMOTIVE SAFETY OR SECURITY DEVICES, LIFE SUPPORT PRODUCTS OR OTHER CRITICAL APPLICATIONS. INCLUSION OF CIRRUS PRODUCTS IN SUCH APPLICATIONS IS UNDERSTOOD TO BE FULLY AT THE CUSTOMER'S RISK AND CIRRUS DISCLAIMS AND MAKES NO WARRANTY, EXPRESS, STATUTORY OR IMPLIED, INCLUDING THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR PARTICULAR PURPOSE, WITH REGARD TO ANY CIRRUS PRODUCT THAT IS USED IN SUCH A MANNER. IF THE CUSTOMER OR CUSTOMER'S CUSTOMER USES OR PERMITS THE USE OF CIRRUS PRODUCTS IN CRITICAL APPLICATIONS, CUSTOMER AGREES, BY SUCH USE, TO FULLY INDEMNIFY CIRRUS, ITS OFFICERS, DIRECTORS, EMPLOYEES, DISTRIBUTORS AND OTHER AGENTS FROM ANY AND ALL LIABILITY, INCLUDING ATTORNEYS' FEES AND COSTS, THAT MAY RESULT FROM OR ARISE IN CONNECTION WITH THESE USES.

Cirrus Logic, Cirrus, the Cirrus Logic logo designs, DSP Composer, Cirrus Extra Surround, Cirrus Original Multichannel Surround, and Cirrus Original Surround are trademarks of Cirrus Logic, Inc. All other brand and product names in this document may be trademarks or service marks of their respective owners.

Dolby, Dolby Digital, Dolby Headphone, Dolby Virtual Speaker, AC-3, and Pro Logic are registered trademarks of Dolby Laboratories, Inc. AAC, Dolby Headphone2, Dolby Virtual Speaker2, and Dolby Digital Surround EX are trademarks of Dolby Laboratories, Inc. Supply of an implementation of Dolby technology does not convey a license nor imply a right under any patent, or any other industrial or intellectual property right of Dolby Laboratories, to use the implementation in any finished end-user or ready-to-use final product. It is hereby notified that a license for such use is required from Dolby Laboratories.

DTS and DTS Digital Surround are registered trademarks of the Digital Theater Systems, Inc. DTS-ES, DTS-ES 96/24, DTS Neo:6, DTS 96/24 are trademarks of the Digital Theater Systems, Inc. It is hereby notified that a third-party license from DTS is necessary to distribute software of DTS in any finished end-user or ready-to-use final product.

THX® is a registered trademark of THX Ltd. THx Ultra2 and THx Select2 are trademarks of THx Ltd.

Re-equalization is a trademark of Lucasfilm, Ltd.

SRS, Circle Surround, and Trusurround XT are registered trademarks of SRS Labs, Inc. Circle Surround II is a trademark of SRS Labs, Inc. The Circle Surround technology rights incorporated in the Cirrus Logic chip are owned by SRS Labs, Inc. and by Valence Technology, Ltd., and licensed to Cirrus Logic, Inc.

Users of any Cirrus Logic chip containing enabled Circle Surround® technology (i.e., Circle Surround® licensees) must first sign a license to purchase production quantities for consumer electronics applications which may be granted upon submission of a preproduction sample to, and the satisfactory passing of performance verification tests performed by SRS Labs, Inc., or Valence Technology, Ltd. E-mail requests for performance specifications and testing rate schedule may be made to cslicense@srslabs.com. SRS Labs, Inc. and Valence Technology, Ltd., reserve the right to decline a use license for any submission that does not pass performance specifications or is not in the consumer electronics classification.

All equipment manufactured using any Cirrus Logic chip containing enabled Circle Surround® technology must carry the Circle Surround® logo on the front panel in a manner approved in writing by SRS Labs, Inc., or Valence Technology, Ltd. If the Circle Surround logo is printed in user manuals, service manuals, or advertisements, it must appear in a form approved in writing by SRS Labs, Inc. or Valence Technology, Ltd. The rear panel of products containing Circle Surround technology and user manuals, service manuals, and advertising for those products must all carry the legends as described in Licensor's most current version of the Circle Surround Trademark Usage Manual.

Intel is a registered trademark of Intel Corporation.

Motorola and SPI are trademarks or registered trademarks of Motorola, Inc.

I^2C is a trademark of Philips Semiconductor Corp.

Philips is a registered trademark of Koninklijke Philips Electronics N.V. Corp.

Sony and Direct Stream Digital are registered trademark of Sony Kabushiki Kaisha T.A. Sony Corporation.

Contents

Figures

Tables

Chapter 1
Introduction

1.1 Overview

The CS4953xx is a programmable audio DSP that combines a programmable, 32-bit fixed-point general purpose DSP with dedicated audio peripherals. Its audio-centric interfaces facilitate the coding of high-precision audio applications and provide a seamless connection to external audio peripheral ICs.

The CS4953xx is a 32-bit RAM-based processor that provides up to 150 MIPS of processing power and includes all standard codes in ROM. It has been designed with a generous amount of on-chip program and data RAM, and has all necessary peripherals required to support the latest standards in consumer entertainment products. In addition, external SDRAM and Flash memory interfaces can be used to expand the data memory. This device is suitable for a variety of high-performance audio applications. These include:

- Audio/Video Receivers
- DVD Receivers
- Stereo TVs
- Mini Systems
- Shelf Systems
- Digital Speakers
- Car Audio Head Units and Amplifiers
- Set-top Boxes

1.1.1 Chip Features

The CS4953xx includes the following features:

- Various Decoding/processing Standards
- 12-channel Serial Audio Inputs
- Dual 32-bit Audio DSP with Dual MAC
- Large On-chip X,Y, and Program RAM
- Supports SDRAM & Flash Memories
- Parallel Control Using Motorola® or Intel® Communication Standards
- Customer Software Security Keys
- 16-channel PCM Output
- Dual S/PDIF Transmitters
- Two Serial Control Ports Using SPI™ or I²C™ Standards
- Digital Audio Input (DAI) Port for Audio Data Delivery in I²S or LJ Format
- GPIO Support for All Common Sub-circuits

Figure 1-1 illustrates the functional block diagram for the CS4953xx chip.

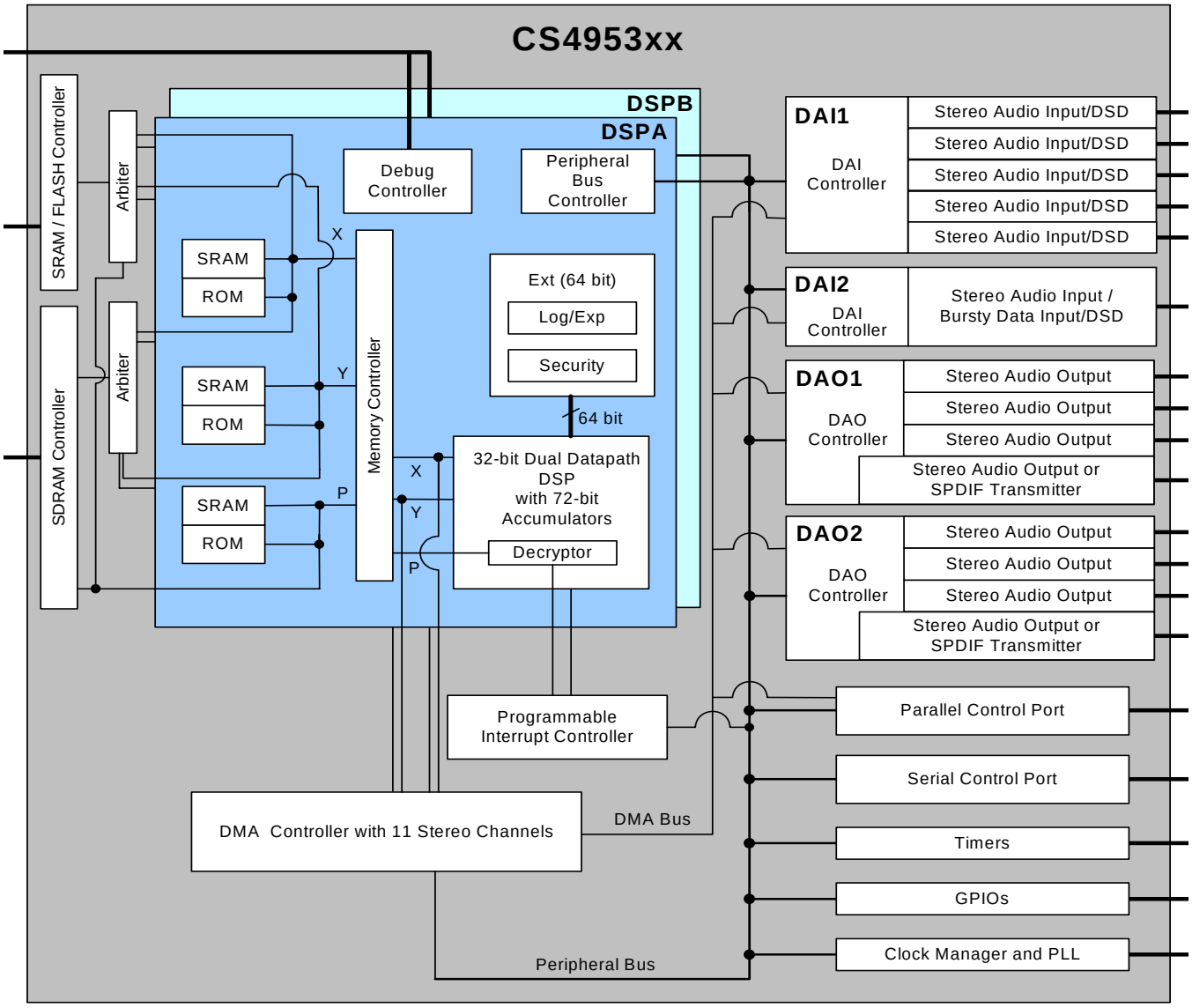


Figure 1-1. CS4953xx Chip Functional Block Diagram

See AN288, "CS4953xx/CS497xxx for a list of audio decoding/processing algorithms that are supported by the CS4953xx: These firmwware modules and their associated application notes are available through the Cirrus Logic Software Licensing Program. .

The CS4953xx contains sufficient on-chip memory to support decoding of all major audio decoding algorithms available today. The CS4953xx supports a glueless SDRAM/Flash interface for increased all-channel delays. The memory interface also supports connection to an external 8- or 16-bit-wide EPROM or Flash memory for code storage, thus allowing products to be field upgraded as new audio algorithms are developed.

This chip, teamed with the Cirrus certified decoder library, Cirrus digital interface products, and mixed-signal data converters, enables the design of next-generation digital entertainment products.

1.2 Functional Overview of the CS4953xx Chip

The CS4953xx chip supports a maximum clock speed of 150 MHz in a 144-pin LQFP or 128-pin LQFP package. A high-level functional description of the CS4953xx chip is provided in this section.

1.2.1 DSP Core

The CS4953xx DSP core is a pair of general purpose, 32-bit, fixed-point, fully programmable digital signal processors that achieve high performance through an efficient instruction set and highly parallel architecture. The device uses two's complement fractional number representation, and employs busses for two data memory spaces and one program memory space.

CS4953xx core enhancements include portability of the design, speed improvement, and improvements for synthesis, verification, and testability. Each member of the CS4953xx family has different SRAM and ROM sizes. Please refer to the CS4953xx data sheet for details.

1.2.2 Security Extension module

This module is a 64-bit extension module that allows the CS4953xx device to be placed in a secure mode where decryption is activated via a 128-bit key. This key is written to the security extension in two 64-bit move instructions. Secure mode is disabled by default, and must be explicitly enabled.

1.2.3 Debug Controller (DBC)

An I^2C slave debug controller (DBC) is integrated within the CS4953xx DSP core. Two pins are reserved for connecting a PC host to the debug ports on either DSP. The debug port consists of two modules, an I^2C slave and a debug master. The DBC master sends dedicated signals into the DSP core to initiate debug actions and it receives acknowledge signals from the core to indicate the requested action has been taken. Basically, this interface allows the DBC to insert instructions into the pipeline. The core will acknowledge the action when it determines the pipeline is in the appropriate state for the inserted action to be taken.

1.2.4 Digital Audio Output (DAO1, DAO2) Controller

The CS4953xx has two Digital Audio Output (DAO) controllers, each of which contains 4 stereo output ports. One port on each DAO can be used as a S/PDIF transmitter. The DAO ports can transmit up to 16 channels of audio sample data in I^2S-compatible format. The audio samples are stored in 16 channel buffers which are 32 bits wide. Four of the channels can also serve as output buffers for the two S/PDIF transmitters. The O/S can dedicate DMA channels to fill the DAO data buffers from memory. DAO control is handled through the peripheral bus.

1.2.5 Digital Audio Input (DAI1) Controller

The CS4953xx Digital Audio Input (DAI) controller has four stereo input ports and DAI control is handled through the peripheral bus. Each DAI pin can be configured to load audio samples in a variety of formats. In addition to accepting multiple formats, the DAI controller has the ability to accept multiple stereo channels on a single DAI1_DATAx pin. All four DAI pins are slaves and normally share the same serial input clock pins (DAI1_SCLK and DAI1_LRCLK). Pins DAI1_DATA[3:0] may also be reconfigured to use the DAO serial input clock pins (DAOx_SCLK and DAOx_LRCLK). A single global configuration register provides a set of enable bits to ensure that ports may be started synchronously.

1.2.6 Compressed Data Input / Digital Audio Input (DAI2) Controller

The DAI2 controller has one input port and its own SCLK and LRCLK and can be used for accepting PCM data in the same way as DAI1, but is used primarily for the delivery of compressed data. When configured for compressed data input, custom internal hardware is enabled that off-loads some pre-processing of the incoming stream to help maximize the MIPS available in the DSP core for user-customized applications.

1.2.7 Direct Stream Digital® (DSD) Controller

The CS4953xx also has a DSD controller which allows the DSP to be integrated into a system that supports SACD audio. The DSD controller pins are shared with the DAI1 and DAI2 ports. The DSD port consists of a bit clock (DSD_CLK) and six DSD data inputs (DSD[5:0]).

1.2.8 General Purpose I/O

A 32-bit general-purpose I/O (GPIO) port is provided on the CS4953xx chip to enhance system flexibility. Many of the functional pins can be used for either GPIO or peripherals.

Each GPIO pin can be individually configured as an output, an input, or an input with interrupt. A GPIO interrupt can be triggered on a rising edge (0-to-1 transition), falling edge (1-to-0 transition), or logic level (either 0 or 1). Each pin configured as an input with interrupt can be assigned its own interrupt trigger condition. All GPIOs share a common interrupt vector.

1.2.9 Parallel Control Port (Motorola®/Intel® Standards) (Optional Feature)

The CS4953xx parallel control port allows an external device such as a microcontroller to communicate with the CS4953xx chip using either a Motorola®-type or Intel®-type parallel communication standard. Only one of the two communication modes can be selected at a time. For external device-control purposes, the CS4953xx is in slave mode for all communication protocols, although it can request the external device to perform a read. The parallel control port supports direct memory access (DMA) and can be used simultaneously with the CS4953xx serial control port.

The parallel control port communication mode selection occurs as the CS4953xx exits a reset condition. The rising edge of the $\overline{\text{RESET}}$ pin samples the HS[4:0] pins to determine the communication mode and boot style. Configuration of the three address input pins A[2:0] allows one of the parallel configuration registers to be selected and accessed.

1.2.10 Serial Control Ports (SPI™ or I²C™ Standards)

The CS4953xx has two serial control ports (SCP) that support SPI™ and I²C™ Master/Slave communication modes. The serial control port allows external devices such as microcontrollers to communicate with the CS4953xx chip through either I²C or SPI serial communication standards and can be configured as either a master or a slave.

The CS4953xx SPI and I²C serial communication modes are identical from a functional standpoint. The main difference between the two is the protocol being implemented between the CS4953xx and the external device. In addition, the I²C slave has a true I²C mode that utilizes data flow mechanisms inherent to the I²C protocol. If this mode is enabled, the I²C slave will hold SCP1_CLK low to delay a transfer as needed.

By default, SCP1 is configured as a slave for external device-controlled data transfers. As a slave, it cannot drive the clock signal nor initiate data transfers.

By default, SCP2 is configured as a master to access a serial FLASH/EEPROM for either booting the DSP or retrieving configuration information. As a master, it can drive the clock signal at up to 1/2 of the DSP's core clock speed.

The CS4953xx has two additional serial communication pins not specified in either the I^2C or SPI specification. The port uses the SCP1_IRQ pin to indicate that a read message is ready for the host. The port uses the SCP1_BSY pin to warn the host to pause communication.

A serial control port can be operated simultaneously with the CS4953xx parallel control port.

1.2.11 SDRAM Controller

The CS4953xx supports a glueless external SDRAM interface to extend the data memory of the DSP during runtime. The SDRAM controller provides 2-port access to X and Y memory space, a quad-word read buffer, and a double-buffered quad-word write buffer. One SDRAM controller port is dedicated to P memory space and the second port is shared by X and Y memories. The X/Y port has dual write buffers and a single read buffer, and the P memory port has a single read buffer. One of these buffers is four 32-bit words (128 bits). Every "miss" to the read buffer will cause the SDRAM controller to burst eight 16-bit reads on the SDRAM interface. The SDRAM controller supports SDRAMs from 2 MB to 64 MB with various row, bank, and column configurations. The SDRAM controller runs synchronous to HCLK, the global chip clock.

1.2.12 Flash Controller

The CS4953xx supports a glueless external Flash interface that allows autoboot from a parallel Flash or EEPROM device extending data memory and/or program memory during DSP runtime. Flash can be accessed using 8-bit, 16-bit, and 32-bit data modes (1-byte, 2-byte, and 4-byte words) and using an 8-bit or 16-bit data bus, where the word width is the number of bytes per transfer, and the data bus size is the width of the physical interface to Flash. Separate chip select pins allow Flash devices to be connected without additional chip select logic. Thus, the interface supports up to 512k x 16 bits of Flash. The external Flash interface serially accesses the X, Y, and P memory spaces on the CS4953xx chip.

1.2.13 DMA Controller

The DMA controller contains 12 stereo channels. The O/S uses 11 stereo channels, 6 for the DAO (2 are for the S/PDIF transmitters), 4 for the DAI, and one for the parallel control port. The addition of the DMA channel for the parallel control port allows compressed audio data to be input over this port. The DMA block is able to move data to/from X or Y memory, or alternate between both X and Y memory. The DMA controller moves data to/from X and/or Y memory opportunistically (if the core is not currently accessing that particular memory space during the current cycle). The DMA controller has a "Dead Man's" timer so that if the core is running an inner loop and accessing memory every cycle, the DMA controller can interrupt the core to run a DMA cycle.

1.2.14 Timers

A 32-bit timer block runs off the CS4953xx DSP clock. The timer count decrements with each clock tick of the DSP clock when the timer is enabled. When the timer count reaches zero, it is re-initialized, and may be programmed to generate an interrupt to the DSP.

1.2.15 Clock Manager and PLL

The CS4953xx Clock Manager and PLL module contains an Analog PLL, RTL Clock Synthesizer, and Clock Manager. The Analog PLL is a customized analog hard macro that contains the Phase Detector (PD), Charge Pump, Loop Filter, VCO, and other non-digital PLL logic. The Clock Synthesizer is a digital design wrapper around the analog PLL that allows clock frequency ranges to be programmed. The Clock Manager is a digital design wrapper for the Clock Synthesizer that provides the logic (control registers) necessary to meet chip clocking requirements.

The Clock Manager and PLL module generates two master clocks:

- HCLK - global chip clock (clocks the DSP core, internal memories, SDRAM, Flash, and all peripherals)
- OVFS - oversampled audio clock. This clock feeds the DAO block which has dividers to generate the DAO_MCLK, DAO_SCLK, and DAO_LRCLK.

The Clock Manager has the ability to bypass the PLL so that the HCLK will run directly off the PLL Reference Clock (REFCLK). While operating in this mode, the OVFS clock can still be divided off the VCO so the PLL can be tested.

1.2.16 Programmable Interrupt Controller

The Programmable Interrupt Controller (PIC) forces all incoming interrupts to be synchronized to the global clock, HCLK. The PIC provides up to 16 interrupts to the DSP Core. The interrupts are prioritized with interrupt 0 as the highest priority and interrupt 15 as the lowest priority. Each interrupt has a corresponding interrupt address that is also supplied to the DSP core. The interrupt address is the same as the IRQ number (interrupt 0 uses interrupt address 0 and interrupt 15 uses interrupt address 15). Both an enable mask and a run mask are provided for each interrupt. The enable mask allows the enabled interrupts to generate a PIC_REQ signal to the DSP core, and the run mask allows the enabled interrupts to generate a PIC_CLR, thereby bringing the core out of its halt state when it accepts the interrupt.

§§[1]

1. **The "§§" symbol is used throughout this manual to indicate the end of the text flow in a chapter.**

Chapter 2
Operational Modes

2.1 Overview

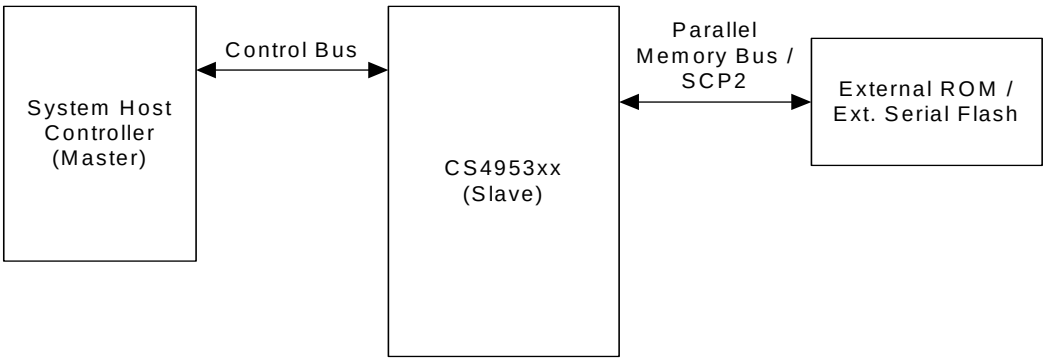
The CS4953xx has several operational modes that can be used to conform to many system configurations. The operational modes for the CS4953xx specify both the communication mode and boot mode. This chapter discusses the selection of operational modes, booting procedures and performing a soft reset.

The CS4953xx can be either a slave device or a master device for the boot procedure. In Master Boot Mode, the CS4953xx is the master boot device and can automatically boot the application code from either serial or parallel external ROM (the slave boot device). In Slave Boot Mode, the CS4953xx is the slave boot device and requires the system host controller (the master boot device) to determine how to boot the application code. The system host controller can either load the application code or the host can direct the CS4953xx to boot application code from serial or parallel external ROM. See Figure 2-1 on page 2-2.

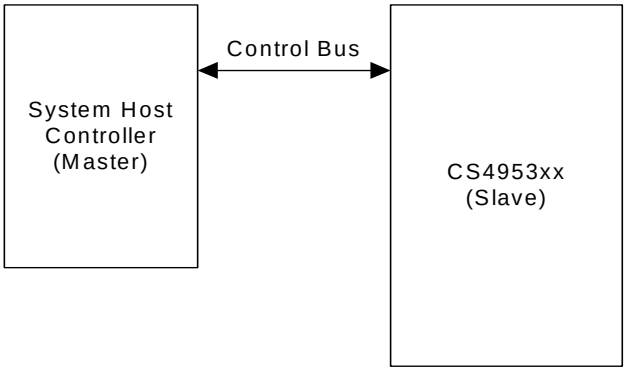
Thus, there are three boot modes for the CS4953xx:

- Master Boot (From serial or parallel external ROM)
- Slave Boot (Using SPI, I^2C, Intel, Motorola, or Multiplexed Intel protocols.)
- Host Controlled Master Boot (serial or parallel external ROM)

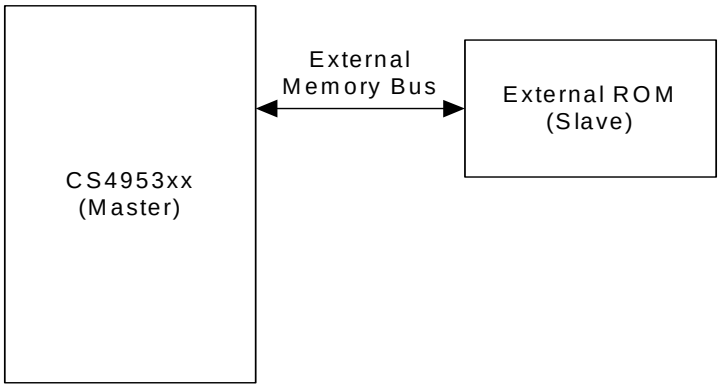
When the CS4953xx is configured for an operational mode where it is the slave boot device, one of the below communication modes must be specified by the host controller (that is, the master boot device). These communication modes are described in detail in Chapter 3, "", Chapter 3, "Serial Control Port" and Chapter 4, "", Chapter 4, "Parallel Control Port". Please see these chapters for block diagrams and flowcharts depicting each of the CS4963x boot modes.



Host Controlled Master Boot
(Recommended for most systems)



Slave Boot



Master Boot
(Not currently supported by O/S)

Figure 2-1. Operation Mode Block Diagrams

2.2 Operational Mode Selection

The operational mode for the CS4953xx is selected by the values of the HS[4:0] pins on the rising edge of RESET. This value determines the communication mode used until the part is reset again. This value also determines the method for loading application code. The table below shows the different operational modes and the HS[4:0] values for each mode.

Table 2-1. Operation Modes

HS[4:0]					Mode	Boot Master Device	Boot Slave Device
X	0	0	0	0	Master SCP1 I^2C	CS4953xx	I^2C External ROM[1,8]
X	1	0	0	0	Master SCP1 SPI1	CS4953xx	SPI (Mode 1) External ROM[2, 5, 7, 8]
X	0	0	0	1	Master SCP1 SPI2	CS4953xx	SPI (Mode 2) External ROM[3, 5, 7, 8]
X	1	0	0	1	Master SCP1 SPI3	CS4953xx	SPI (Mode 3) External ROM[4, 5, 7, 8]
X	0	0	1	0	Master 8-bit Flash	CS4953xx	8-bit External ROM[6]
X	1	0	1	0	Master 16-bit Flash	CS4953xx	16-bit External ROM[6]
X	X	1	0	0	Slave/HCMB SCP1 I^2C	System Host	CS4953xx
X	X	1	0	1	Slave/HCMB SCP1 SPI	System Host	CS4953xx
X	X	1	1	0	Slave/HCMB PCP Intel	System Host	CS4953xx
X	X	1	1	1	Slave/HCMB PCP Motorola	System Host	CS4953xx
X	X	0	1	1	Slave/HCMB PCP Mux	System Host	CS4953xx

1. In I^2C master mode, the Image Start address (0x0) is sent as a 16-bit value, with the default I^2C address of 0x50, I^2C clock frequency = F_{dclk} / 72.
2. SPI master mode 1 is to support the legacy 16-bit SPI EEPROM. The following defaults are used: SPI Command Byte 0x03, Image Start address 0x0 is sent as a 16-bit value, no dummy bytes, SPI clock frequency = F_{dclk} / 4.
3. In SPI Master mode 2, the following defaults are used: SPI Command Byte 0x68, Image Start address 0x0 is sent as a 24-bit value, 4 dummy bytes sent following the address (and before reading image data), SPI clock frequency = F_{dclk} / 2. This mode supports the Atmel SPI Flash memory.
4. In SPI Master mode 3, the following defaults are used: SPI command byte 0x03, Image Start address 0x0 is sent as a 24-bit value, no dummy bytes, SPI clock frequency = F_{dclk} / 2. This mode supports the ST SPI EEPROM devices.
5. For all SPI Master boot modes, by default GPIO20 is used as $\overline{EE\_CS}$, but the HCMB message can be configured to select an alternate pin for $\overline{EE\_CS}$.
6. For Flash Master modes, the following defaults are used: clock ratio=1:1, Endian Mode = little-endian, Chip Select polarity = active-low, 0-cycle delay from CS/Address Change to Output Enable, 4-cycle delay from CS to Read Access.
7. Master Boot Modes are currently not supported by the O/S.
8. F_{dclk} is specified in the CS4953xx data sheet.

2.3 Slave Boot Procedures

When the CS4953xx is the slave boot device, the system host controller (as the master boot device) must follow an outlined procedure for correctly loading application code. The two methods of slave boot for the CS4953xx, slave boot and host-controlled master boot are described in this section. Each of these methods requires the system host controller to send messages to, and read back messages from, the CS4953xx. These messages have been outlined in Section 2.3.3 "Boot Messages" on page 2-10.

The CS4953xx has different *.uld* files (overlays) for certain processing tasks. Slave booting the CS4953xx requires loading multiple overlays - differing from previous Cirrus Logic Audio DSP families (this is, CS493xx, CS494xxx). Please refer to AN288, *"CS4953xx Firmware User's Manual"* regarding more information on the breakdown of processing tasks for each overlay.

Pseudocode and flowcharts will be used to describe each of these boot procedures in detail. The flow charts use the following messages:

- Write_* – Write to CS4953xx
- Read_* – Read from CS4953xx

Please note that * above can be replaced by SPI, I^2C, Intel®, Multiplexed Intel, or Motorola® depending on the mode of host communication. For each case, the general download algorithm is the same. The system designer should also refer to the control port sections of this document in Chapter 3, "", Chapter 3, "Serial Control Port"and Chapter 4, "", Chapter 4, "Parallel Control Port" for the details of when writing to and reading from the CS4953xx is valid.

One feature that is of special note – the entire boot procedure for the CS4953xx can be made of a combination of slave boot and host-controlled master boot procedures. An example can be seen in Figure 2-3 on page 8.

After completing the full download to the CS4953xx, a KICK START message is sent to cause the application code begin execution. Please note that it takes time to lock the PLL and initialize the SDRAM interface when initially booting the DSP. Typically this time is less than 200 ms. If a message is sent to the DSP during this time, the SCP1_BSY pin will go low to indicate that the DSP is busy. Any messages sent when the SCP1_BSY pin is LOW will be lost. If the SCP1_BSY pin stays LOW longer than 200 ms the host must reboot the DSP.

2.3.1 Host Controlled Master Boot

The Host Controlled Master Boot (HCMB) procedure is a sequence where the system host controller instructs the CS4953xx to boot application code from either the external memory interface (ROM or Flash), or the serial control interface (serial SPI Flash/EEPROM or I^2C EEPROM). The system host controller can communicate with the CS4953xx via SPI, I^2C, or one of the three parallel formats (Intel, Multiplexed Intel, or Motorola). The external memory start address of the code image, as well as the data rate, are specified by the host by the HCMB_<MODE> message. These messages are defined in Section 2.3.3 "Boot Messages" on page 2-10.

2.3.1.1 Performing a Host Controlled Master Boot (HCMB)

Figure 2-2 shows the steps taken during a Host Controlled Master Boot (HCMB). The procedure is discussed in Section 2.3.1.1.1.

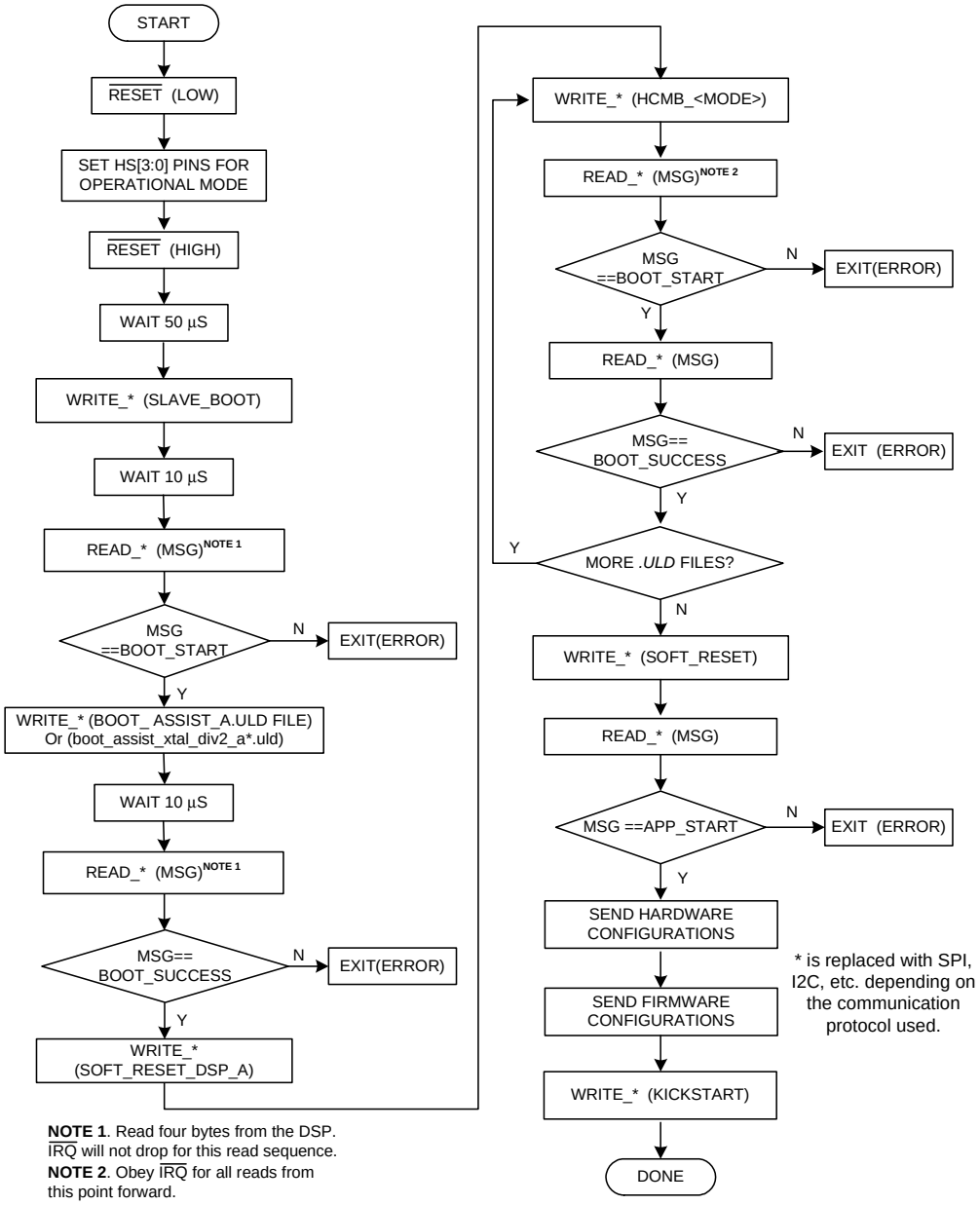


Figure 2-2. Host Controlled Master Boot

2.3.1.1.1 Host Controlled Master Boot (HCMB) Procedure

1. **Toggle RESET**. A download sequence is started when the host holds the RESET pin low for the required time. The mode pins, HS[4:0] must be in the appropriate state to set the host communication mode before and immediately after the rising edge of RESET. Pull-up and pull-down resistors are typically used to set the default state of the HS[4:0] pins.
2. **Send the SLAVE_BOOT message**. The host sends the appropriate SLAVE_BOOT message to the CS4953xx using the control port specified (serial port/parallel port) and format specified (I^2C, SPI, Intel, etc.) by the HS[4:0] pins at reset.

3. Wait for 10 µS.
4. **Read the BOOT_START message** (See NOTE 1 in Figure 2-2). If the initialization is successful, CS4953xx sends out the BOOT_START message and the host proceeds to Step 6.

   If initialization fails, the host must return to **Step 1**, and if failure is met again, the communication timing and protocol should be inspected.
5. **The host sends the boot assist** BOOT_ASSIST_A.uld file or boot_assist_xtal_div2_a*.uld (sets XTAL_OUT =XTAL/2) to the CS4953xx DSP.
6. Wait 10 µS
7. **Read the BOOT_SUCCESS message** (See NOTE 1 in Figure 2-2). The host then reads a message from the appropriate communications port. Each.ULD file contains a checksum that is compared at the end of the boot process. CS4953xx sends a BOOT_SUCCESS message to the host if the checksum is correct after the download.

   If the checksum was incorrect, CS4953xx responds with a BOOT_ERROR_CHECKSUM message. This indicates that the image read by the DSP is corrupted. The communications interface hardware and code image integrity should be checked if this occurs.
8. **Send the SOFT_RESET_DSP_A command:** After reading the BOOT_SUCCESS message on the boot assist code image/overlay, the host must send this message. If boot_assist_xtal_div2_a*.uld was sent in **Step 5**, the XTAL_OUT frequency will change to XTAL/2 after the soft reset has taken place.
9. **Send the correct HCMB_<MODE> message**. The host sends to the CS4953xx the appropriate HCMB_<MODE> message, where <MODE> indicates the type of external ROM: PARALLEL, SPI, or I^2C. This message tells CS4953xx the start address of the downloadable image (*.ULD* file) and identifies which port will be used to access FLASH memory.
10. **Wait for $\overline{\text{IRQ}}$ low.** The host then waits for $\overline{\text{SCP1\_IRQ}}$ (or $\overline{\text{PCP\_IRQ}}$) to go low. (See NOTE 2 in Figure 2-2)
11. **Read the BOOT_START message**. If the initialization is successful, CS4953xx will send the BOOT_START message to the host.

    If initialization fails, the host must return to **Step 1,** and if failure is met again, the communication timing and protocol should be inspected.
12. **Wait for $\overline{\text{IRQ}}$ low.** After receiving the BOOT_START message, the host then waits for $\overline{\text{SCP1\_IRQ}}$ (or $\overline{\text{PCP\_IRQ}}$) to go low. This indicates that the DSP has written a message to the output buffer and the boot process is complete.
13. **Read the BOOT_SUCCESS message**. The host then reads a message from the appropriate communications port. Each*.ULD* file contains a checksum that is compared at the end of the boot process. The CS4953xx sends a BOOT_SUCCESS message to the host if the checksum is correct after the download.

    If the checksum was incorrect, CS4953xx responds with a BOOT_ERROR_CHECKSUM message. This indicates that the image read by the DSP is corrupted. The communications interface hardware and code image integrity should be checked if this occurs.
14. **Repeat Steps 9-13 for all code images/Overlays.** The host repeats these steps until all overlays for the application have been successfully loaded. See the application note for more information on the overlays necessary at start-up.
15. **Send the SOFT_RESET message**. After reading the BOOT_SUCCESS message on the last code image/overlay, the host must send a SOFT_RESET message which will cause the application code to begin executing.
16. **Wait for IRQ low.** The host then waits for $\overline{\text{SCP1\_IRQ}}$ (or $\overline{\text{PCP\_IRQ}}$) to go low.

17. **Read the APP_START message**. If code execution is successful, the CS4953xx sends out an APP_START message. This indicates that the code has been initialized and can accept further configuration messages. The host should not attempt further communication with the CS4953xx until the APP_START message has been read.

    If the CS4953xx does not send an application start message, the host must return to **Step 1**.

18. **Send Hardware Configuration messages**. The master boot procedure is completed. The operating system on the CS4953xx is now ready for host configuration of hardware and software.

    Hardware configuration messages are used to define the behavior of the CS4953xx's audio ports.

19. **Send Software Configuration messages**.The software configuration messages are specific to each application. The application code User's Guide for each application provides a list of all pertinent configuration messages.

20. **Send the KICKSTART message**. The CS4953xx application locks the PLL and begins processing audio after receiving this message.

2.3.2 Slave Boot

The Slave Boot procedure is a sequence in which the external host is the bus master and directly loads the CS4953xx application code. The system host controller has each of the five communication modes available, as specified in Table 2-1. from either the serial control interface (SPI or I^2C) or the parallel control interface (Intel, multiplexed Intel, or Motorola modes). The boot messages used can be found in Section 2.3.3 "Boot Messages" on page 2-10. For information on how to configure the CS4953xx overlays, such as hardware configuration messages, software configuration messages, and the kick-start message, please refer to AN288, *"CS4953xx/CS497xxx Firmware User's Manual."*

2.3.2.1 Performing a Slave Boot

Figure 2-3 shows the steps taken during a Slave boot. The procedure is discussed in Section 2.3.2.1.1..

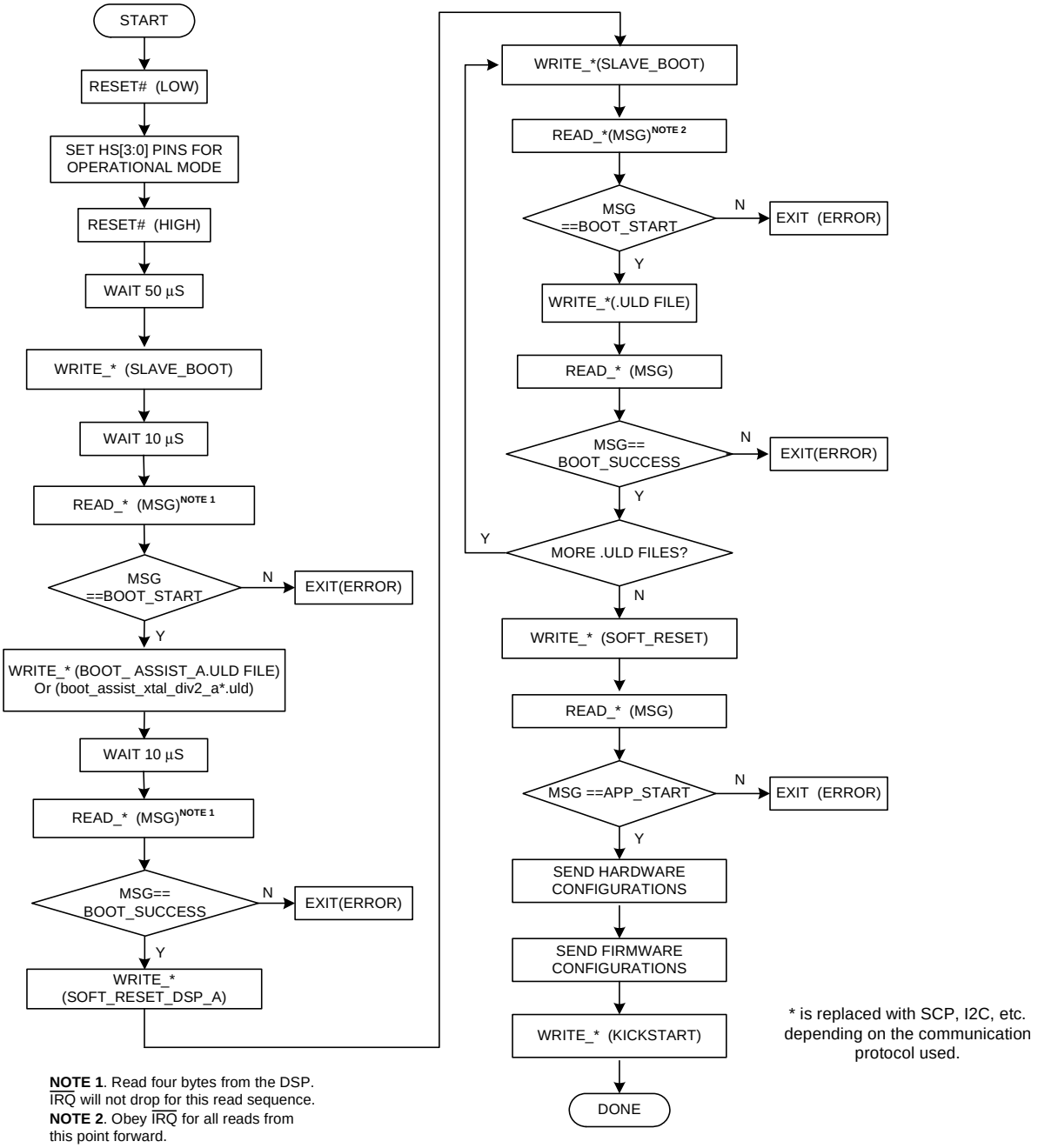


Figure 2-3. Slave Boot Sequence

2.3.2.1.1 Slave Boot Procedure

1. **Toggle RESET**. A download sequence is started when the host holds the RESET pin low for the required time. The mode pins (HS[4:0]) must be in the appropriate state to set the host communication mode before and immediately after the rising edge of RESET. Pull-up and pull-down resistors are typically used to set the default state of the HS[4:0] pins.
2. Wait for 50 µs.

3. **Send the SLAVE_BOOT message**. The host sends the appropriate SLAVE_BOOT message to the CS4953xx using the control port specified (serial port/parallel port) and format specified (I^2C, SPI, Intel, etc.) by the HS[4:0] pins at reset.
4. Wait for 10 µS.
5. **Read the BOOT_START message** (See NOTE 1 in Figure 2-3). If the initialization is successful, CS4953xx sends out the BOOT_START message and the host proceeds to Step 6.

   If initialization fails, the host must return to **Step 1**, and if failure is met again, the communication timing and protocol should be inspected.
6. **The host sends the boot assist** BOOT_ASSIST_A.uld file or boot_assist_xtal_div2_a*.uld (sets XTAL_OUT =XTAL/2) to the CS4953xx DSP.
7. Wait 10 µS
8. **Read the BOOT_SUCCESS message** (See NOTE 1 in Figure 2-3). The host then reads a message from the appropriate communications port. Each.ULD file contains a checksum that is compared at the end of the boot process. CS4953xx sends a BOOT_SUCCESS message to the host if the checksum is correct after the download.

   If the checksum was incorrect, CS4953xx responds with a BOOT_ERROR_CHECKSUM message. This indicates that the image read by the DSP is corrupted. The communications interface hardware and code image integrity should be checked if this occurs.
9. **Send the SOFT_RESET_DSP_A command:** After reading the BOOT_SUCCESS message on the boot assist code image/overlay, the host must send this message. If boot_assist_xtal_div2_a*.uld was sent in **Step 6**, the XTAL_OUT frequency will change to XTAL/2 after the soft reset has taken place.
10. **Send the SLAVE_BOOT message**. The host sends the appropriate SLAVE_BOOT message to the CS4953xx using the control port specified (serial port/parallel port) and format specified (I^2C, SPI, Intel, etc.) by the HS[4:0] pins at reset.
11. **Wait for IRQ low.** The host then waits for $\overline{\text{SCP1\_IRQ}}$ (or $\overline{\text{PCP\_IRQ}}$) to go low. (See NOTE 2 in Figure 2-3)
12. **Read the BOOT_START message**. If the initialization is successful, CS4953xx sends out the BOOT_START message and the host proceeds to Step 6.
13. **Send the ULD File.** The host sends a.uld file to the CS4953xxx.
14. **Wait for IRQ low.** The host then waits for $\overline{\text{SCP1\_IRQ}}$ (or $\overline{\text{PCP\_IRQ}}$) to go low.
15. **Read the BOOT_SUCCESS message**. The host then reads a message from the appropriate communications port. Each.ULD file contains a checksum that is compared at the end of the boot process. CS4953xx sends a BOOT_SUCCESS message to the host if the checksum is correct after the download.

    If the checksum was incorrect, CS4953xx responds with a BOOT_ERROR_CHECKSUM message. This indicates that the image read by the DSP is corrupted. The communications interface hardware and code image integrity should be checked if this occurs.
16. **Repeat Steps 10-15 for all code images/overlays**. The host repeats these steps until all overlays for the application have been successfully loaded. See the application note for more information on the overlays necessary at start-up.
17. **Send the SOFT_RESET message**. After reading the BOOT_SUCCESS message on the last code image/overlay, the host must send a SOFT_RESET message which will cause the application code to begin executing.
18. **Wait for IRQ low.** The host then waits for $\overline{\text{SCP1\_IRQ}}$ (or $\overline{\text{PCP\_IRQ}}$) to go low.
19. **Read the APP_START message**. If code execution is successful, the CS4953xx sends out a

APP_START message. This indicates that the code has been initialized and can accept further configuration messages. The host should not attempt further communication with the CS4953xx until the APP_START message has been read.

If the CS4953xx does not send an application start message, the host must return to **Step 1**.

20. **Send Hardware Configuration messages**. The slave boot procedure is completed. The operating system on the CS4953xx is now ready for host configuration of hardware and software.

    Hardware configuration messages are used to define the behavior of the CS4953xx's audio ports. A more detailed description of the hardware configurations can be found in Section x of this manual.

21. **Send Software Configuration messages**.The software configuration messages are specific to each application. The application code user's guide for each application provides a list of all pertinent configuration messages.

22. **Send the KICKSTART message(s)**. The CS4953xx application locks the PLL and begins processing audio after receiving this message.

2.3.3 Boot Messages

The Slave Boot and Host-Controlled Master Boot procedures use a number of messages to configure and synchronize the boot process. Please use the messages listed below when implementing the boot process as a part of the system host controller firmware.

2.3.3.1 Slave Boot

Table 2-2. SLAVE_BOOT message for CS4953xx

MNEMONIC	VALUE
SLAVE_BOOT	0x8000 0000

The SLAVE_BOOT message is used when the system host controller will send each *.uld* file directly to the CS4953xx. The SLAVE_BOOT message must be issued for each overlay image (*.uld* file) that is downloaded to the CS4953xx. Please see Section 2-3 "Slave Boot Sequence" on page 2-8 for more details.

2.3.3.2 Host-Controlled Master Boot from Parallel ROM

Table 2-3. HCMB_PARALLEL Message for CS4953xx

MNEMONIC	VALUE
HCMB_PARALLEL	1110 0000 0000 0000 0000 0xxx xxyy yyMM 0000 0000 0000 AAAA AAAA AAAA AAAA AAAA Where: x = number of CS4953xx clocks from CS or Address change to Output Enable y = number of CS4953xx clocks from CS to Output Enable M = memory width of parllel ROM = 0 for 8-bit, 1 for 16-bit A = 20-bit external memory start address/2 for 16 bit flash. A = 20-bit external memory start address/4 for 8 bit flash.

HCMB_PARALLEL is used when the application code is stored in external parallel ROM, such as an external 8-bit or 16-bit EEPROM or Flash. The external data bus width is specified by the 'M' variable in the first control word. Also, the 20-bit start address is specified in the second control word with the 'A' variable. Read cycle parameters can also be configured by the 'x' and 'y' variables. The HCMB_PARALLEL message should be substituted for the HCMB_<MODE> message in Figure 2-2.

2.3.3.3 Host-Controlled Master Boot from I²C ROM

Table 2-4. HCMB_I2C message for the CS4953xx

MNEMONIC	VALUE
HCMB_I2C	1100 0000 000p cccc cccc cccc 0sss ssss 0000 0000 0000 0000 0000 0000 AAAA AAAA Where p = Serial Control Port Selection =1 for SCP2 (Normal Operation) 0 for SCP1 (Not Supported by O/S) c = clock divider for I²C clock signal s = 8-bit I²C command A = 16-bit external memory start address

HCMB_I2C is used when the application code is stored in external I²C ROM, such as an I2C EEPROM. The HCMB_I2C message should be substituted for the HCMB_<MODE> message in Figure 2-2. This boot mode can be configured to interface with many types of I²C ROMs. The 16-bit start address is specified by the 'A' variable. The I²C clock is derived from the internal core clock. This clock can be divided down with the 'c' 12-bit divider variable. The command byte (the first byte to the I²C ROM) can be defined by the 's' variable. The CS4953xx control port used for the HCMB_I2C can be configured by the 'p' variable.

2.3.3.4 Host-Controlled Master Boot from SPI ROM

Table 2-5. HCMB_SPI message for CS4953xx

MNEMONIC	VALUE
HCMB_SPI	1101 BBB0 0SSp cccc cccc cccc ssss ssss L000 0000 AAAA AAAA AAAA AAAA AAAA AAAA Where B = number of dummy bytes sent after Address, before read S = Chip Select p = Serial Control Port Selection = 1 for SCP2 0 for SCP1, c = SPI clock speed = DSP Core Clock/(c+2) s = 8-bit SPI command L = Address Length = 0 for 16-bit Address, 1 for 24-bit Address A = 24/16-bit external memory start address

HCMB_SPI is used when the application code is stored in external SPI ROM, either SPI EEPROM or SPI Flash. The HCMB_SPI message should be substituted for the HCMB_<MODE> message in Figure 2-2. This boot mode can be configured to interface with many types of SPI ROMs. The start address "A" variable can be either 24-bit or 16-bit, configured by the "L" variable. Some SPI ROMs require some 'dummy' bytes after the address byte, before reading from the part, configurable by the 'B' variable.

The SPI clock is derived from the internal core clock. This clock can be divided down with the "c" 12-bit divider variable. The command byte (the first byte to the SPI ROM) can be defined by the "s" variable. The CS4953xx control port used for the HCMB_SPI can be configured by the 'p' variable. Finally, the "S" variable configures the chip select used, according to Table 2-6 below.

Table 2-6. GPIO Pins Available as $\overline{\text{EE\_CS}}$ in HCMB

'S' Value	Pin Name	LQFP-144 Pin #	LQFP-128 Pin #
0	GPIO20	6	38
1	GPIO23	14	46
2	GPIO25	25	-
3	GPIO0[1]	121	-

1.GPIO0 as $\overline{\text{EE\_CS}}$ can be used to load only one .uld in HCMB mode. If multiple .uld files are to be loaded, do not use GPIO0 as $\overline{\text{EE\_CS}}$ in HCMB Mode..

2.3.3.5 Soft Reset

Table 2-7. SOFT_RESET message for CS4953xx

MNEMONIC	VALUE
SOFT_RESET	0x4000 0000
SOFT_RESET_DSP_A	0x5000 0000

The SOFT_RESET message is the message sent to the CS4953xx after all of the overlays have been successfully booted. The SOFT_RESET leaves execution of the bootloader and begins execution of the loaded overlays. The overlays can be configured once the SOFT_RESET message has been sent.

2.3.3.6 Messages Read from CS4953xx

Table 2-8 defines the boot read messages, in mnemonic and actual hex value, used in CS4953xx boot sequences.

Table 2-8. Boot Read Messages from CS4953xx

MNEMONIC	VALUE
BOOT_START	0x0000 0001
BOOT_SUCCESS	0x0000 0002
APP_START	0x0000 0004
BOOT_ERROR_CHECKSUM	0x0000 00FF
INVALID_BOOT_TYPE	0x0000 00FE
BOOT_FAILURE	0x0000 00F8
APPLICATION_FAILURE	0xF0{ID} 0000

Note: There is a unique {ID} for every .uld file.

Table 2-9 is a quick reference showing the different boot commands understood by the CS4953xx, in mnemonic and actual hex value, used in CS4953xx boot sequences.

Table 2-9. Boot Command Messages for CS4953xx

MNEMONIC	VALUE	DETAILED TABLE
SLAVE_BOOT	0x8000 0000	Table 2-2
HCMB_PARALLEL	0xE0== ====	Table 2-3
HCMB_I^{2}C	0xC0== ====	Table 2-4
HCMB_SPI	0xD=== ====	Table 2-5
SOFT_RESET	0x40 00 00 00	Table 2-7

2.4 Master Boot Procedure

Note: Master Boot is currently not supported in the O/S

A master boot sequence is initiated immediately after the rising edge of $\overline{\text{RESET}}$. The location of the overlay to boot is outlined in Table 2-1. Once the rising edge of $\overline{\text{RESET}}$ has occurred, the CS4953xx will load a single overlay from address 0x0. It should be noted that the loaded overlay must reconfigure one of the control ports to be slave to the bus for a system host controller to configure the part. Thus, this type of boot process will be useful in systems without a system host controller or with a simple controller that only performs a monitoring task. Currently this mode is not used for any applications.

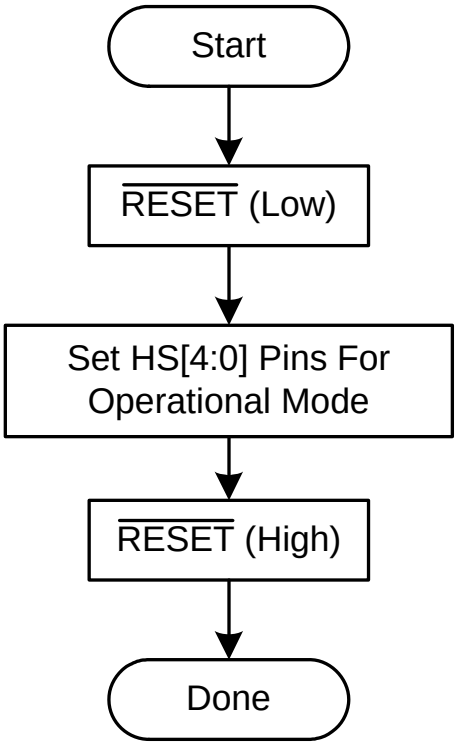


Figure 2-4. Master Boot Sequence Flowchart

2.5 Softboot

The O/S application code for the CS4953xx allows users to swap out one or more overlays during run time, without the need for re-download of the entire overlay stack. This is helpful for reducing the time required for switching between different types of incoming audio data streams.

The Softboot procedure includes initial messaging that sends the CS4953xx into a boot state where the host can boot the CS4953xx with different overlays according to the boot methods outlined in this chapter. This includes a soft reset of the CS4953xx, which then requires that the host send or re-send the hardware and software configuration messages.

2.5.1 Softboot Messaging

Two messages are relevant to the softboot procedure for the CS4953xx. These messages are: SOFTBOOT and SOFTBOOT_ACK.

The SOFTBOOT message is sent from the host controller to the CS4953xx to indicate to the CS4953xx that the system requires swapping of overlays.

Table 2-10. SOFTBOOT Message

Mnemonic	Value
SOFTBOOT	0x81000009 0x00000001

The SOFTBOOT_ACK is sent from the CS4953xx to the host controller to indicate that the host can now boot the CS4953xx with the new overlays.

Table 2-11. SOFTBOOT_ACK Message

Mnemonic	Value
SOFTBOOT_ACK	0x00000005

2.5.2 Softboot Procedure

Figure 2-5 contains a flow diagram and description of the Softboot procedure. This is a step-by-step guideline that can be used as an aid in developing the system controller code required to drive the CS4953xx.

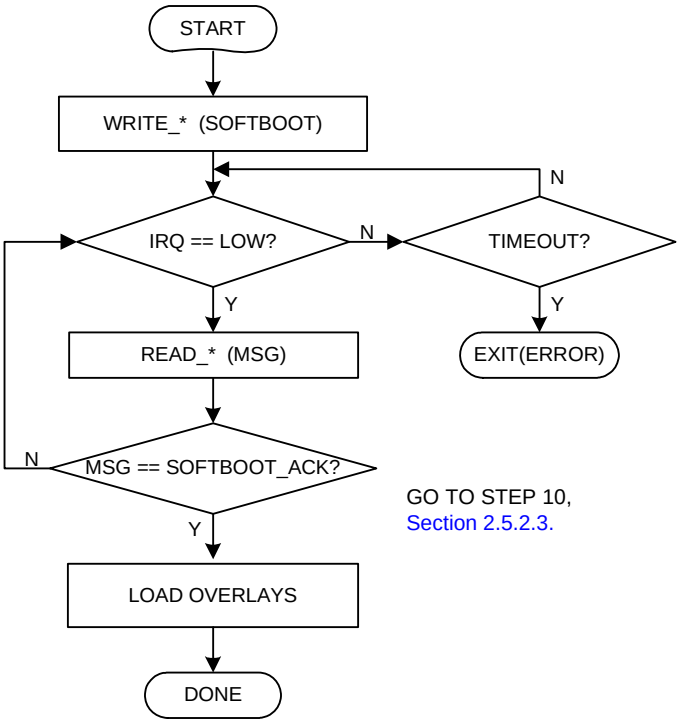


Figure 2-5. Soft Boot Sequence Flowchart

2.5.2.1 Softboot Procedure

1. **Send the SOFTBOOT message.** The host sends the SOFTBOOT message to the CS4953xx to begin overlay swap.
2. **Wait for IRQ low**. The host then waits for $\overline{\text{SCP1\_IRQ}}$ (or $\overline{\text{PCP\_IRQ}}$) to go low. If the TIMEOUT period has been reached, the host should exit. If the IRQ pin is LOW, proceed to Step 3.
3. **Read the SOFTBOOT_ACK message**. If the message is the SOFTBOOT_ACK message (0x00000005), then the host should proceed to Step 4. If the message is not the SOFTBOOT_ACK message, the host should return to Step 2.
4. **Load Overlays.** Repeat the boot procedure used to originally load the overlays into the CS4953xx (for example, SLAVE_BOOT, HCMB_<MODE>), but only the overlays that need to be swapped should be loaded. Skip the hard reset sequence, starting the boot procedure from **Step 2**. Please note that this includes re-downloading all hardware and software configurations for the CS4953xx overlays. **If no overlay,** change as required, and go to Step 10, Section 2.5.2.3..

2.5.2.2 Softboot Example

Figure 2-6 contains an example softboot flow diagram. Section 2.5.2.3.provides a step-by-step description of the Softboot procedure using the Host Control Master Boot (HCMB) procedure that is most commonly used CS4593x systems.

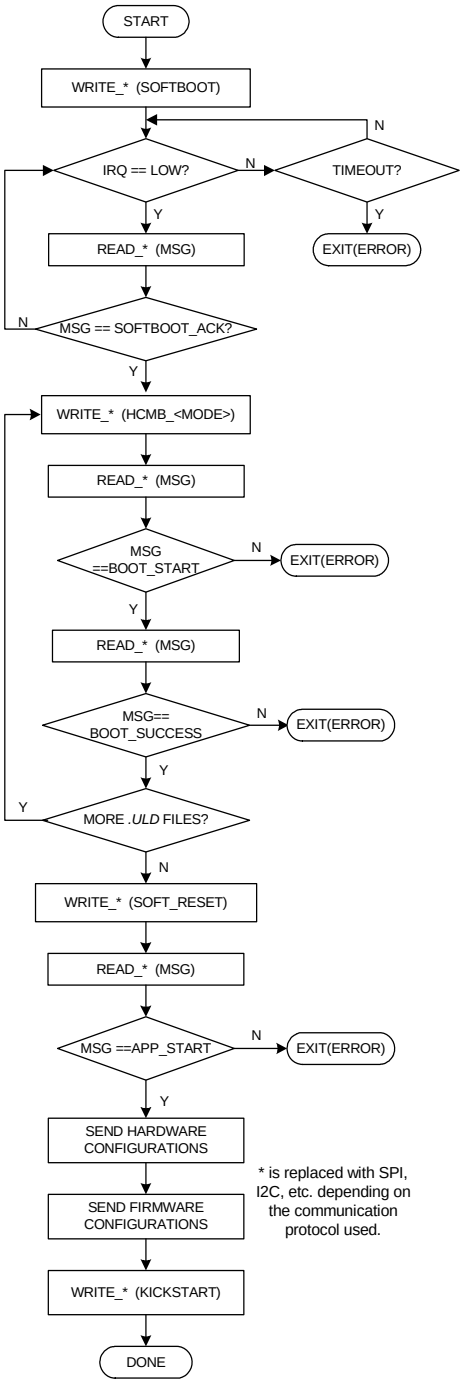


Figure 2-6. Soft Boot Example Flowchart

2.5.2.3 Softboot Example Steps

1. **Send the SOFTBOOT message.** The host sends the SOFTBOOT message to the CS4953xx to begin overlay swap.
2. **Wait for IRQ low**. The host then waits for $\overline{\text{SCP1\_IRQ}}$ (or $\overline{\text{PCP\_IRQ}}$) to go low. If the TIMEOUT period has been reached, the host should exit. If the IRQ pin is LOW, proceed to Step 3.
3. **Read the SOFTBOOT_ACK message**. If the message is the SOFTBOOT_ACK message (0x00000005), then the host should proceed to Step 4. If the message is not the SOFTBOOT_ACK message, the host should return to Step 2.
4. **Send the correct HCMB_<MODE> message**. The host sends to the CS4953xx the appropriate HCMB_<MODE> message, where <MODE> indicates the type of external ROM: PARALLEL, SPI, or I2C. This message tells CS4953xx the start address of the downloadable image (*.ULD* file) and identifies which port will be used to access FLASH memory.
5. **Wait for IRQ low.** The host then waits for $\overline{\text{SCP1\_IRQ}}$ (or $\overline{\text{PCP\_IRQ}}$) to go low.
6. **Read the BOOT_START message**. If the initialization is successful, CS4953xx will send the BOOT_START message to the host. If the message is any other value, then the host should abort.
7. **Wait for IRQ low.** After receiving the BOOT_START message, the host then waits for $\overline{\text{SCP1\_IRQ}}$ (or $\overline{\text{PCP\_IRQ}}$) to go low. This indicates that the DSP has written a message to the output buffer and the boot process is complete.
8. **Read the BOOT_SUCCESS message**. The host then reads another message from the appropriate communications port. Each*.ULD* file contains a checksum that is compared at the end of the boot process. The CS4953xx sends a BOOT_SUCCESS message to the host if the checksum is correct after the download.

   If the checksum was incorrect, CS4953xx responds with a BOOT_ERROR_CHECKSUM message. This indicates that the image read by the DSP is corrupted. The communications interface hardware and code image integrity should be checked if this occurs.
9. **Repeat Steps 4-8 for all code images/Overlays.** The host repeats these steps until all overlays for the application have been successfully loaded. See the application note for more information on the overlays necessary at start-up.
10. **Send the SOFT_RESET message**. After reading the BOOT_SUCCESS message on the last code image/overlay, the host must send a SOFT_RESET message which will cause the application code to begin executing.
11. **Wait for IRQ low.** The host then waits for $\overline{\text{SCP1\_IRQ}}$ (or $\overline{\text{PCP\_IRQ}}$) to go low.
12. **Read the APP_START message**. If code execution is successful, the CS4953xx sends out an APP_START message. This indicates that the code has been initialized and can accept further configuration messages. The host should not attempt further communication with the CS4953xx until the APP_START message has been read.

    If the CS4953xx does not send an application start message, the host must return to **Step 1**. If the message read is any value other than APP_START, the system controller should abort.
13. **Send Hardware Configuration messages**. The master boot procedure is completed. The operating system on the CS4953xx is now ready for host configuration of hardware and software.

    Hardware configuration messages are used to define the behavior of the CS4953xx's audio ports.
14. **Send Software Configuration messages**.The software configuration messages are specific to each application. The application code User's Guide for each application provides a list of all pertinent configuration messages.
15. **Send the KICKSTART message**. The CS4953xx begins processing audio after receiving this message.

§§

Chapter 3
Serial Control Port

3.1 Overview

The CS4953xx uses the Serial Control Port (SCP) to communicate with external devices such as host microprocessors using either I^2C or SPI serial communication formats. Each port can be configured as either a master or slave. The CS4953xx DSP serial port communicates using the SCP_CLK, SCP_MOSI, and SCP_MISO (SPI serial master and slave modes), and SCP_SDA (for I^2C serial master and slave modes) pins.

In both SPI and I^2C modes, the serial control port performs 8-bit transfers and is always configured as a slave for external device-controlled data transfers. As a slave, it cannot drive the clock signal nor initiate data transfers. The port can request a read from the host by activating the SCP1_IRQ pin. The port can also indicate that the host should stop sending data by activating the SCP1_BSY pin.

It is very important for the host to obey the SCP1_BSY pin status. Messages sent to the DSP's host control port (SCP1) when SCP1_BSY pin is low will be lost.

The serial control port can be operated simultaneously with the CS4953xx parallel control port.

The CS4953xx SPI and I^2C serial communication modes are identical from a functional standpoint. The main difference between the two is the actual protocol being implemented between the CS4953xx and the host. In addition, the I^2C slave has a true I^2C mode that utilizes data flow mechanisms inherent to the I^2C protocol. If this mode is enabled, the I^2C slave will hold SCP1_CLK low to delay a transfer as needed -- this is in addition to activating SCP1_BSY.

The CS4953xx has two serial ports. However, the O/S currently supports only slave mode host communication on SCP1, and master mode communication on SCP2 for booting from a serial EEPROM/ FLASH.

3.2 Serial Control Port Configuration

The serial control port configuration for an operating mode is determined by the state of special boot mode pins as the CS4953xx exits reset. The rising edge of the RESET pin samples the HS[4:0] pins to determine the communication mode and boot style. The CS4953xx O/S currently supports two serial control port configurations for host control:

- I^2C Slave (Write Address = 0x80, Read Address = 0x81)
- SPI Slave (Write Address = 0x80, Read Address = 0x81)

The HS[4:0] signals latched by RESET are read by the boot ROM code to determine the format (SPI slave or I^2C slave). The ROM code then configures the serial control port into slave mode and looks for the BOOT_START message. Please see Chapter 2, "Operational Modes" for additional details on configuring CS4953xx ports and communication modes.

Procedures for configuring the serial control port for SPI and I^2C communication modes are provided in this chapter.

3.3 I²C Port

The CS4953xx I²C bus has been developed for 8-bit digital control applications, such as those requiring microcontrollers. The I²C bus interface is a bidirectional serial port that uses 2 lines (data and clock) for data transmission and reception with software-addressable external devices. Each external device interfaced to the CS4953xx I²C port has the ability to communicate directly with the other devices and is assigned a unique address whether it is a CPU, memory, or some other device. A block diagram of the CS4953xx I²C Serial Control Port is provided in Figure 3-1.

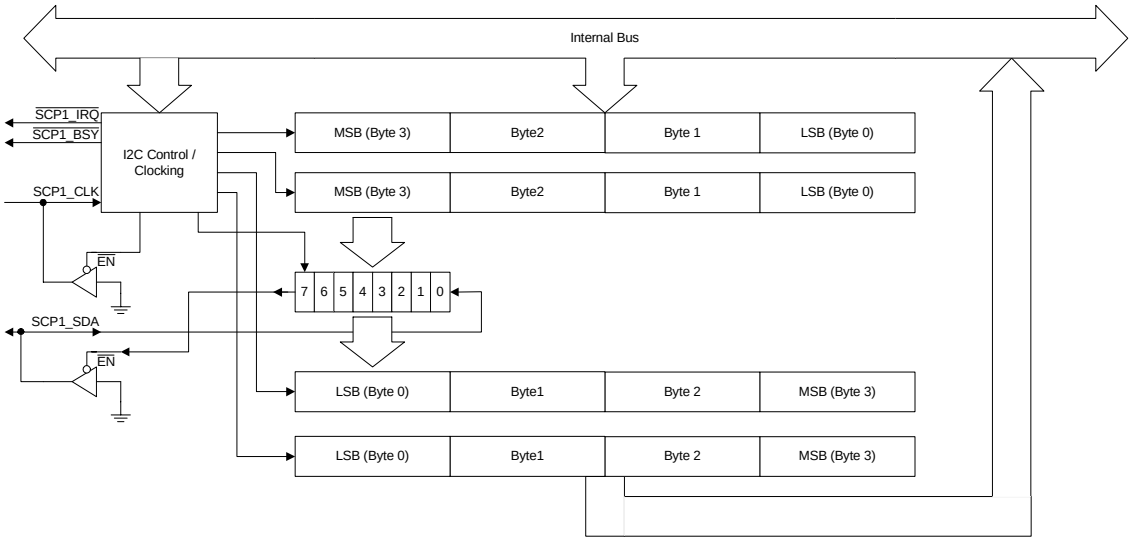


Figure 3-1. Serial Control Port Internal Block Diagram

3.3.1 I²C System Bus Description

Devices can be considered masters or slaves when performing data transfers. A master is the device which initiates a data transfer on the bus and generates the clock signals to permit that transfer. Any device addressed by the initiator is considered a slave.

The CS4953xx has two serial ports. However, the O/S currently supports only slave mode host communication on SCP1, and master mode communication on SCP2 for booting from a serial EEPROM/ FLASH.

The I²C-bus is a multi-master bus. This means that more than one device capable of controlling the bus can be connected to it. The master-slave relationships found on the I²C bus are not permanent and only depend on the direction of data transfer at that time. Generation of clock signals on the I²C bus is always the responsibility of master devices; each master generates its own clock signals when transferring data on the bus. Bus clock signals from a master can only be altered when they are stretched by a slow slave device holding down SCP1_CLK.

Both SCP1_SDA and SCP1_CLK are bidirectional lines. When the bus is free, both lines are pulled high by resistors. The output stages of devices connected to the bus must have an open-drain or open-collector to perform the wired-AND function.

As seen in Figure 3-2, two serial ports are available on the CS4953xx. Each can be configured as either master or slave. For Audio applications, SCP1 is configured as a slave port and SCP2 is configured as a master port. SCP2 is used only in systems that are booting from serial EEPROM.

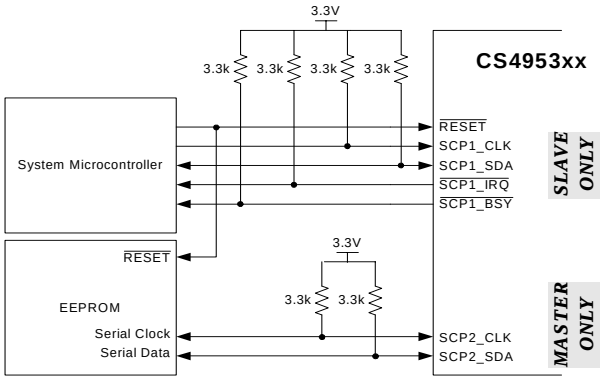


Figure 3-2. Block Diagram of I^2C System Bus

Table 3-1 shows the signal names, descriptions, and pin number of the signals associated with the I^2C Serial Control Port on the CS4953xx.

Table 3-1. Serial Control Port 1 I^2C Signals

Pin Name	Pin Description	LQFP-144 Pin #	LQFP-128 Pin #	Pin Type
SCP1_CLK	I^2C Control Port Bit Clock. In master mode, this pin serves as the serial control clock output (open drain in I^2C mode / output in SPI mode). In serial slave mode, this pin serves as the serial control clock input. In I^2C slave mode the clock can be pulled low by the port to stall the master.	99	126	Open Drain
SCP1_SDA	Bidirectional Data I^2C Mode Master/Slave Data IO. In I^2C master and slave mode, this open drain pin serves as the data input and output.	97	124	Open Drain
$\overline{\text{SCP1\_IRQ}}$	Control Port Data Ready Interrupt Request, Output, Active Low This pin is driven low when the DSP has a message for the host to read. The pin will go high when the host has read the message and the DSP has no further messages.	100	4	Open Drain
$\overline{\text{SCP1\_BSY}}$	Serial Control Port 1 Input Busy, Output, Active Low This pin is driven low when the control port's receive buffer is full. Internal Buffer is 4 bytes (1 DSP Word) deep.	102	128	Open Drain

Table 3-1. Serial Control Port 1 I^2C Signals (Continued)

Pin Name	Pin Description	LQFP-144 Pin #	LQFP-128 Pin #	Pin Type
SCP2_CLK	I^2C Control Port Bit Clock. In master mode, this pin serves as the serial control clock output (open drain in I^2C mode / output in SPI mode). In serial slave mode, this pin serves as the serial control clock input. In I^2C slave mode the clock can be pulled low by the port to stall the master.	103	1	Open Drain
SCP2_SDA	Bidirectional Data I^2C Mode Master/Slave Data IO. In I^2C master and slave mode, this open drain pin serves as the data input and output.	105	2	Open Drain
$\overline{\text{SCP2\_IRQ}}$	Serial Control Port Data Ready Interrupt Request, Output, Active Low. This pin is driven low when the DSP has a message for the host to read. The pin will go high when the host has read the message and the DSP has no further messages.	108	5	Open Drain

3.3.2 I^2C Bus Dynamics

The Start condition for an I^2C transaction is defined as the first falling edge on the SCP1_SDA line while SCP1_CLK is high. An I^2C Stop condition is defined as the first rising edge on the SCP1_SDA line while SCP1_CLK is high. Hence for valid data transfer, SCP1_SDA must remain stable during the high period of the clock pulse. Start and Stop conditions are always generated by the master. The bus is considered to be busy after the Start condition. The bus is considered to be free again following the Stop condition. The bus stays busy if a repeated Start condition is generated instead of a Stop condition. In this respect, the Start and repeated Start conditions are functionally identical.

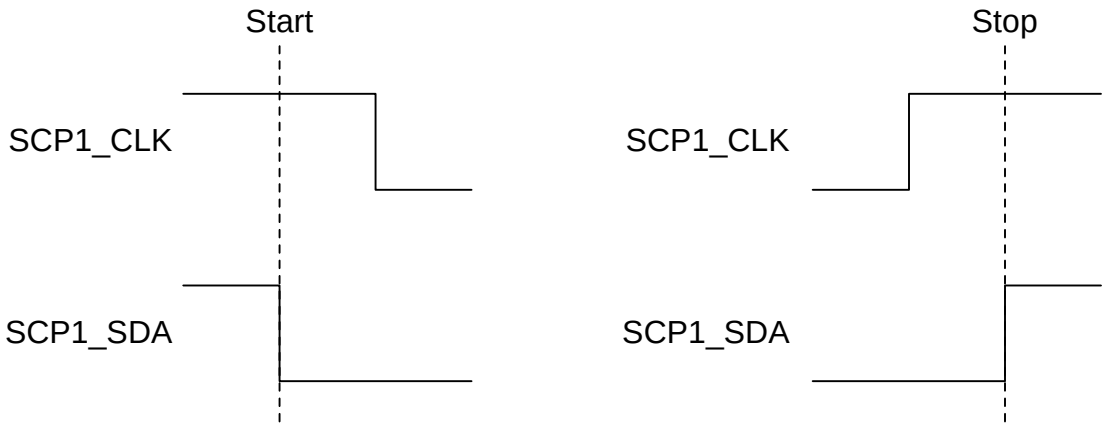


Figure 3-3. I^2C Start and Stop Conditions

The number of bytes that can be transmitted per transfer is unrestricted. Data is transferred with the most-significant bit (MSB) first. The first byte is an address byte that is always sent by the master after a Start or repeated Start condition. This byte must be a 7-bit I^2C slave address + R/$\overline{W}$ bit. The 7-bit I^2C address for the CS4953xx is 1000000b (0x80). The R/$\overline{W}$ bit is used to notify the slave if the current transaction is for the master to write data to the slave (R/$\overline{W}$ = 0) or read data from the slave (R/$\overline{W}$ = 1).

After the master has sent the address byte, the master releases the SCP1_SDA line. If the slave received the address byte, it will drive the SCP1_SDA line low to acknowledge (ACK) to the master that the byte was received. The SCP1_SDA line must remain stable and low during the high period of the next clock pulse. When a slave does not acknowledge the slave address, the data line must be left high by the slave (NACK).

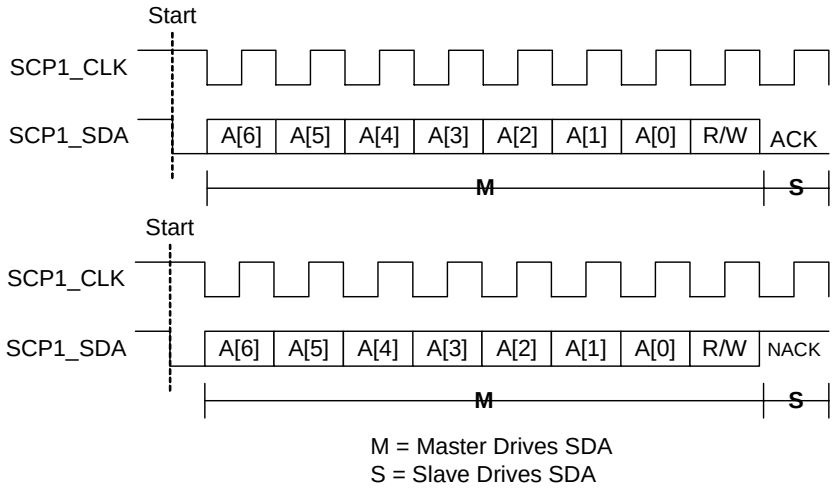


Figure 3-4. I^2C Address with ACK and NACK

For write operations, the R/$\overline{W}$ bit must be set to zero (R/$\overline{W}$ = 0, Address = 0x80). After the 8-bit data byte has been clocked, the master will release the SCP1_SDA line. If the slave received the byte correctly, it will drive the SCP1_SDA line low for the next bit clock to acknowledge (ACK) that the data was received. If the data was not received correctly, the slave can communicate this by leaving the SCP1_SDA line high (NACK).

For read operations, the R/$\overline{W}$ bit must be set to one (R/$\overline{W}$ = 1, Address = 0x81), then the master will read a data byte from the slave. After the 8-bit data byte has been clocked, the master will release the SCP1_SDA line. If the master received the byte correctly, it will drive the SCP1_SDA line low for the next bit clock to acknowledge (ACK) that the data was received. If the data was not received correctly, the master can communicate this by leaving the SCP1_SDA line high (NACK). The protocol on the last byte, however, is different. When the master receives the last byte, it signals the end of the data to the slave by allowing SCP1_SDA to float high (NACK).

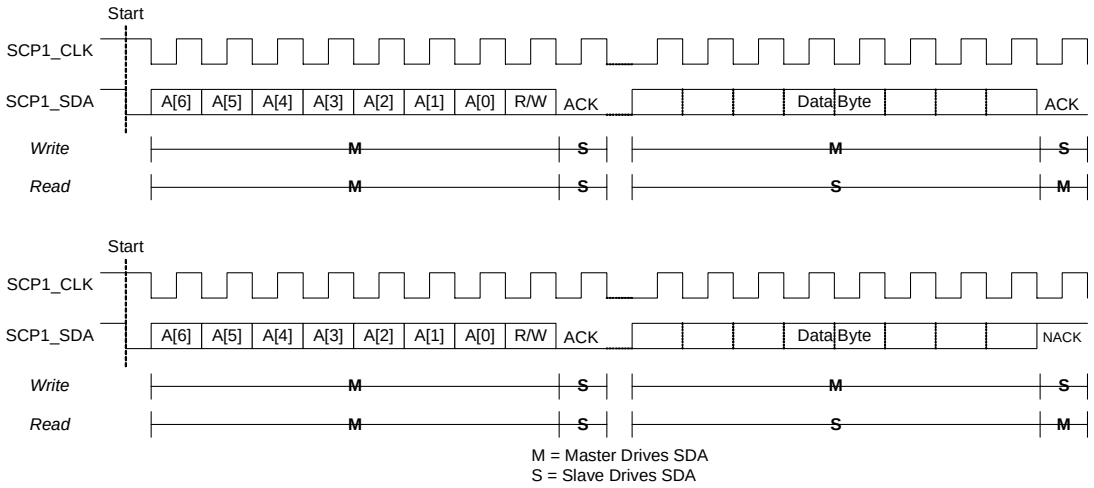


Figure 3-5. Data Byte with ACK and NACK

After an ACK or NACK from the master or slave, the slave must leave the SCP1_SDA line high so the master can then generate either another Start condition as shown in Figure 3-6 to start a new transfer or a Stop condition as shown in Figure 3-7 to abort the transfer.

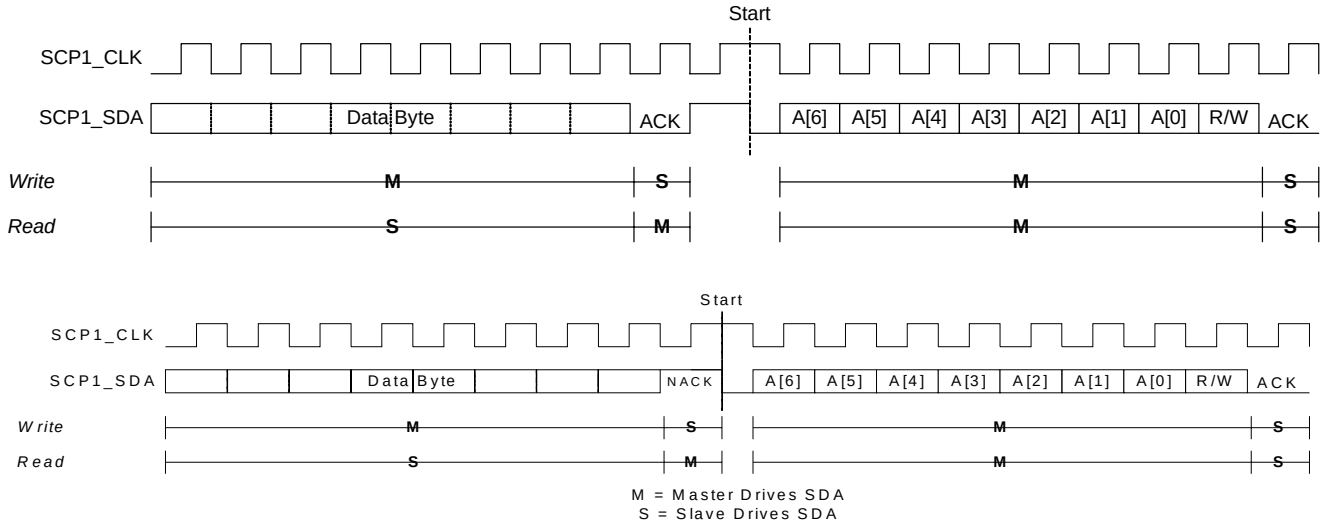


Figure 3-6. Repeated Start Condition with ACK and NACK

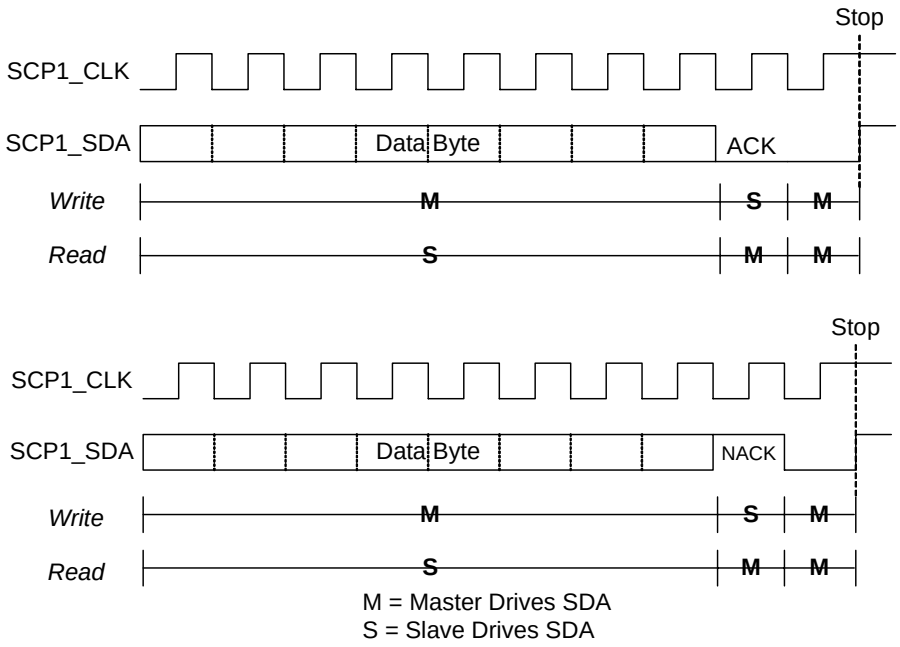


Figure 3-7. Stop Condition with ACK and NACK

If a slave cannot receive or transmit another complete byte of data until it has performed some other function, for example servicing an internal interrupt, it can hold the SCP1_CLK line low to force the master into a wait state. Data transfer then continues when the slave is ready for another byte of data and releases SCP1_CLK.

3.3.3 I²C Messaging

Messaging to the CS4953xx using the I²C bus requires usage of all the information provided in the above I²C Section 3.3.1 "I²C System Bus Description" on page 2 and Section 3.3.2 "I²C Bus Dynamics" on page 4. Every I²C transaction to the CS4953xx will involve 4-byte words used for control and application image download. A detailed description of the serial SPI communication mode is provided in this section. This includes:

- A flow diagram and description for a serial I²C write
- A flow diagram and description for a serial I²C read

3.3.3.1 $\overline{\text{SCP1\_BSY}}$ Behavior

The $\overline{\text{SCP1\_BSY}}$ signal is not part of the I²C protocol, but it is provided so that the slave can signal to the master that it cannot receive any more data. It performs the same function as that of holding SCP1_CLK low to halt transmission. A falling edge of the $\overline{\text{SCP1\_BSY}}$ signal indicates the master must halt transmission. Once the $\overline{\text{SCP1\_BSY}}$ signal goes high, the suspended transaction may continue. It is important for the host to obey the $\overline{\text{SCP1\_BSY}}$ pin status for proper communication with the DSP.

3.3.3.2 Performing a Serial I²C Write

Information provided in this section is intended as a functional description indicating how to use the configured serial control port to perform a I²C write from an external device (master) to the CS4953xx DSP (slave). The system designer must ensure that all timing constraints of the I²C write cycle are met (see the CS4953xx datasheet for timing specifications). When writing to the CS4953xx, the same protocol described in this section will be used when writing single-word messages to the boot firmware, writing multiple-word overlay images to the boot firmware, and writing multiple-word messages to application firmware. The examples given can therefor be expanded to fit any I²C writing situation.

The flow diagram shown in Figure 3-8 illustrates the sequence of events that define the I²C write protocol for SCP1. Section 3.4.3.2..describes the Serial I²C Write protocol.

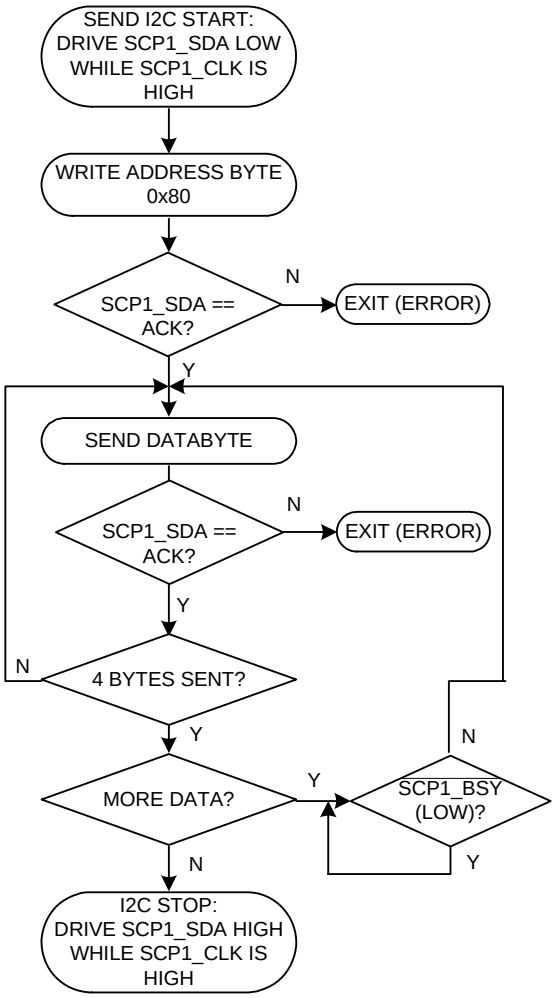


Figure 3-8. I²C Write Flow Diagram

3.3.3.3 I²C Write Protocol

1. An I²C transfer is initiated with an I²C start condition which is defined as the data (SCP1_SDA) line falling while the clock (SCP1_CLK) is held high.
2. This is followed by a 7-bit address and the read/write bit held low for a write. So, the master should send 0x80. The 0x80 byte represents the 7-bit I²C address 1000000b, and the least significant bit set to '0', designates a write.
3. After each byte (including the address and each data byte) the master must release the data line and provide a ninth clock for the CS4953xx DSP (slave) to acknowledge (ACK) receipt of the byte. The CS4953xx will drive the data line low during the ninth clock to acknowledge. If for some reason CS4953xx does not acknowledge (NACK), it means that the communications channel has been corrupted and the CS4953xx should be re-booted. A NACK should never happen here.
4. The master should then clock one data byte into the device, most-significant bit first.
5. The CS4953xx (slave) will (and must) acknowledge (ACK) each byte that it receives which means that after each byte, the master must provide an acknowledge clock pulse on SCP1_CLK and release the data line, SCP1_SDA.
6. If the master has no more data words to write to the CS4953xx, then proceed to Step 8. If the master has more data words to write to the CS4953xx, then proceed to Step 7.
7. The master should poll the $\overline{\text{SCP1\_BSY}}$ signal until it goes high. If the $\overline{\text{SCP1\_BSY}}$ signal is low, it indicates that the CS4953xx is busy performing some task that requires pausing the serial control port. Once the CS4953xx is able to receive more data words, the $\overline{\text{SCP1\_BSY}}$ signal will go high. Once the $\overline{\text{SCP1\_BSY}}$ signal is high, proceed to Step 4.

Note: The DSP's I²C port also implements clock stretching to indicate that the host should pause communication. So the host has the option of checking for SCP1_CLK held low rather than $\overline{\text{SCP1\_BSY}}$ low.

8. At the end of a data transfer, a stop condition must be sent. The stop condition is defined as the rising edge of SCP1_SDA while SCP1_CLK is high.

3.3.3.4 Performing a Serial I²C Read

Information provided in this section is intended as a functional description indicating how to use the configured serial control port to perform an I²C read from an external device (master) to the CS4953xx DSP (slave). The system designer must ensure that all timing constraints of the I²C Read Cycle are met (see the *CS4953xx Data Sheet* for timing specifications). I²C read transactions from the CS4953xx will always involve reading 4-byte words.

Figure 3-9 illustrates the sequence of events that define the I²C read protocol for SCP1. This protocol is discussed in the high-level procedure found in Section 3.3.3.5.

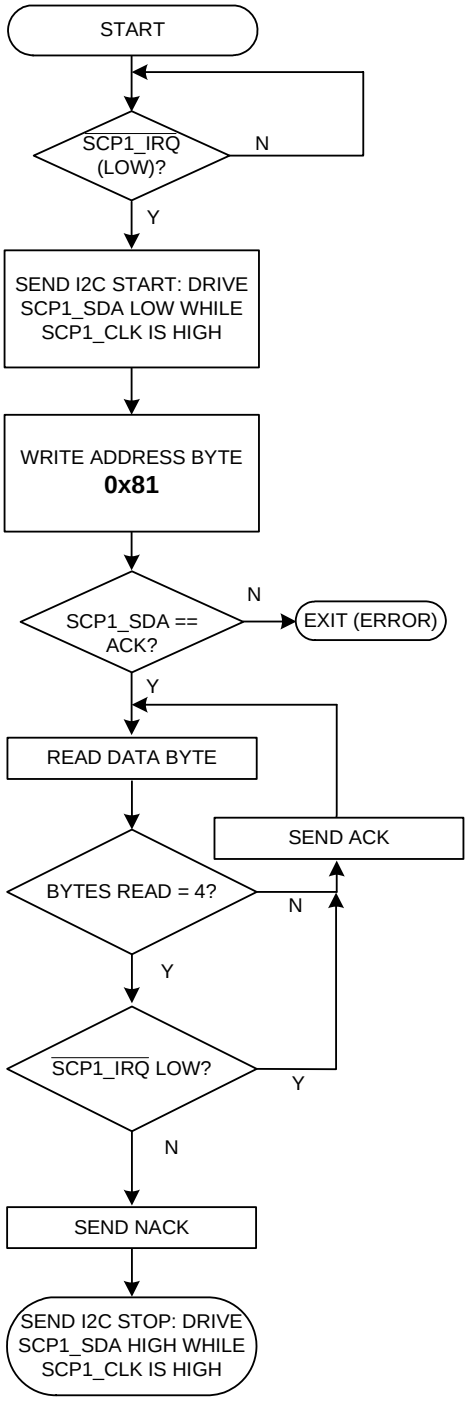
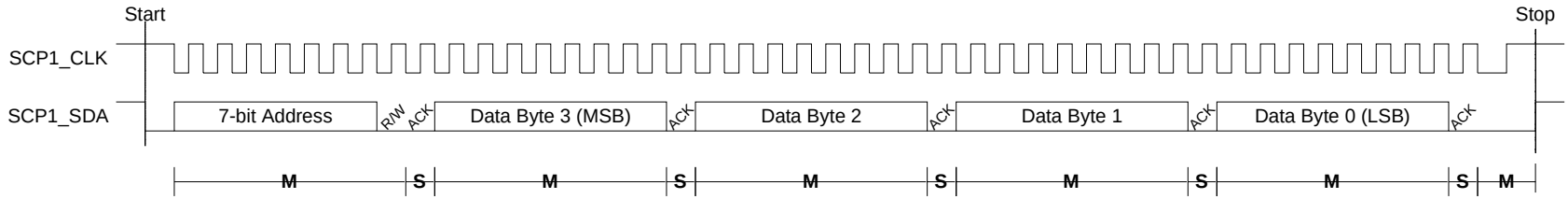


Figure 3-9. I^2C Read Flow Diagram

3.3.3.5 I²C Read Procedure

1. An I²C read transaction is initiated by CS4953xx driving $\overline{\text{SCP1\_IRQ}}$ low, signaling that it has data to be read.
2. The master responds by sending an I²C Start condition which is SCP1_SDA going low while SCP1_CLK is held high.
3. This is followed by a 7-bit address and the read/write bit set high for a read. So, the master should send 0x81. The 0x81 byte represents the 7-bit I²C address 1000000b, and the least significant bit set to '1', designates a read.
4. After the address byte, the master must release the data line and provide a ninth clock for the CS4953xx DSP (slave) to acknowledge (ACK) receipt of the byte. The CS4953xx will drive the data line low during the ninth clock to acknowledge. If for some reason CS4953xx does not acknowledge (NACK), it means that the communications channel has been corrupted and the CS4953xx should be re-booted. A NACK should never happen here.
5. The data is ready to be clocked out on the SCP1_SDA line at this point. Data clocked out by the host is valid on the rising edge of SCP1_CLK and data transitions occur just after the falling edge of SCP1_CLK.
6. After the CS4953xx has written the byte to the master on the SCP1_SDA line, it will release the SCP1_SDA line. If the master has more bytes in the 4-byte word to read, then proceed to Step 7. If the master is finished reading all bytes of the 4-byte word, then proceed to Step 8.
7. The master should drive the SCP1_SDA line low for the 9th SCP1_CLK clock to acknowledge (ACK) that the byte was received from the CS4953xx. The master should then return to Step 5 to read another byte of the 4-byte word.
8. If $\overline{\text{SCP1\_IRQ}}$ is still low after reading a 4-byte word, proceed to Step 7. If $\overline{\text{SCP1\_IRQ}}$ has risen, proceed to Step 9. See Section 3.4.3.5 on page 3-21 for an in-depth explanation of $\overline{\text{SCP1\_IRQ}}$.
9. The master should let the SCP1_SDA line stay high for the 9th SCP1_CLK clock as a no-acknowledge (NACK) to CS4953xx. This, followed by an I²C stop condition (SCP1_SDA driven high, while SCP1_CLK is high) signals an end of read to CS4953xx.



Notes: 1. The I^{2}C slave is always responsible for driving the ACK for the address byte.

2. The I^{2}C slave is responsible for driving the ACK during I^{2}C writes.

Figure 3-10. Sample Waveform for I^{2}C Write Functional TIming

Note: The I^{2}C slave is always responsible for driving the ACK for the address byte.

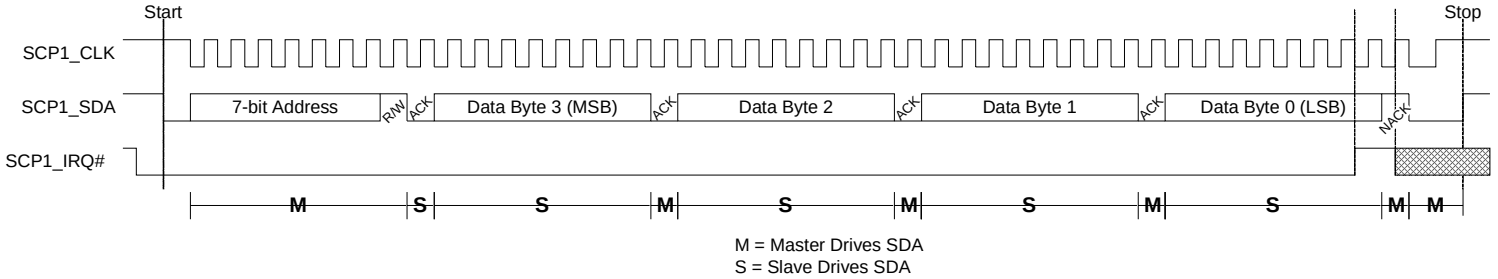


Figure 3-11. Sample Waveform for I^{2}C Read Functional TIming

Notes:

1. The I^{2}C slave is drives the ACK for the address byte.
2. The I^{2}C master is responsible for controlling ACK during I^{2}C reads. In general, the receiver in an I^{2}C transaction is responsible for providing ACK.
3. $\overline{\text{SCP1\_IRQ}}$ remains low until the rising edge of the clock for the last bit of the last byte read from the I^{2}C slave.
4. A NACK is sent by the master after the last byte to indicate the end of the read cycle. This must be followed with an I^{2}C Stop condition or I^{2}C Repeated-Start condition.
5. If there are more data words to read, IRQ will fall at the rising edge of CLK for the NACK. Otherwise, IRQ remains high until an I^{2}C Stop condition or an I^{2}C Repeated-Start condition occurs.

3.3.3.6 SCP1_IRQ Behavior

Once the BOOT_ASSIST_A (.ULD file) has been downloaded in accordance to Steps 1 through 8 in Section 2.3.1 "Host Controlled Master Boot" on page 4 or Steps 1 through 8 in Section 2.3.2 "Slave Boot" on page 7, the SCP1_IRQ pin is functionally enabled.

The SCP1_IRQ signal is not part of the I^2C protocol, but is provided so that the slave can signal that it has data to be read. A high-to-low transition on SCP1_IRQ indicates to the master that the slave has data to be read. When a master detects a high-to-low transition on SCP1_IRQ, it should send a Start condition and begin reading data from the slave.

SCP1_IRQ is guaranteed to remain low (once it has gone low), until the falling edge of SCP1_CLK for the last bit of the last byte to be transferred out of CS4953xx (that is, the rising edge of SCP1_CLK before the ACK). If there is no more data to be transferred, SCP1_IRQ will go high at this point. After going high, SCP1_IRQ is guaranteed to stay high until the next rising edge of SCP1_CLK (that is, it will stay high until the rising edge of SCP1_CLK for the ACK/NACK bit).

This end-of-transfer condition signals the master to end the read transaction by clocking the last data bit out of CS4953xx and then sending a NACK to CS4953xx to signal that the read sequence is over. At this point, the master should send an I^2C stop condition to complete the read sequence. If SCP1_IRQ is still low after the rising edge of SCP1_CLK on the last data bit of the current byte, the master should send an acknowledge and continue reading data from the serial control port. It should be noted that all data should be read out of the serial control port during one cycle or a loss of data will occur. In other words, all data should be read out of the chip until SCP1_IRQ signals the last byte by going high as described above.

3.4 SPI Port

The CS4953xx Serial Peripheral Interface (SPI) bus has been developed for 8-bit digital control applications, such as those requiring microcontrollers. SPI communication is accomplished with 5 lines: Serial Chip Select (SCP1_CS), Serial Control Clock (SCP1_CLK), Master Out/Slave In data (SCP1_MOSI), and a Master In/Slave Out data (SCP1_MISO). Although the separate data I/O lines provide full-duplex capabilities, the CS4953xx chip only uses a half-duplex SPI-bus. Each device on the bus may respond to one or more unique commands, and can operate as either a transmitter or receiver. A device is considered the master in a transaction if it drives the CS pin of another device, and is also mastering the SCP1_CLK line. A block diagram of the CS4953xx SPI Serial Control Port is provided in Figure 3-12.

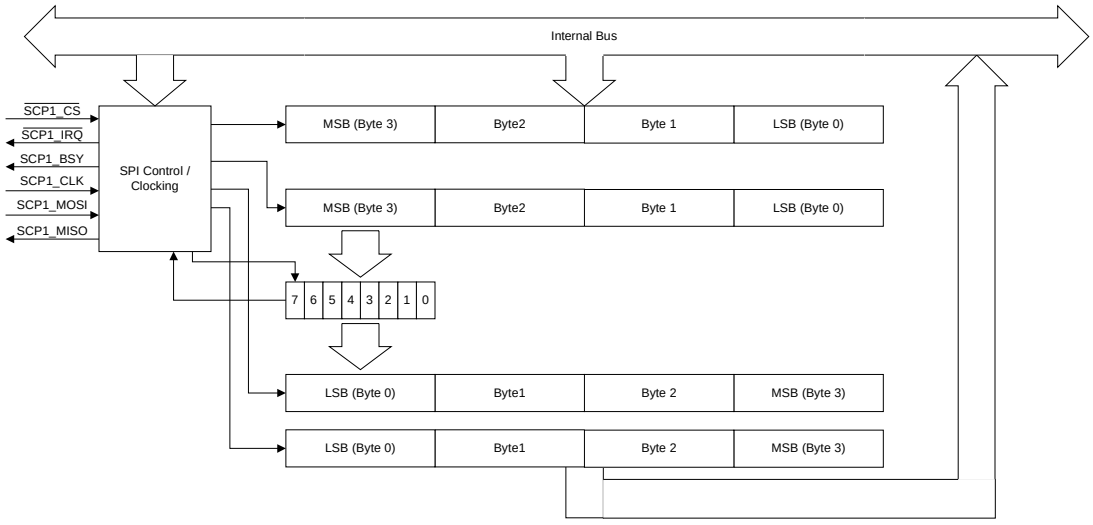


Figure 3-12. SPI Serial Control Port Internal Block Diagram

Table 3-2 shows the signal names, descriptions, and pin number of the signals associated with the SPI Serial Control Port on the CS4953xx.

Table 3-2. Serial Control Port SPI Signals

Pin Name	Pin Description	LQFP-144 Pin #	LQFP-128 Pin #	Pin Type
$\overline{\text{SCP1\_CS}}$	SPI Chip Select, Active Low In serial SPI slave mode, this pin is used as the active-low chip-select input signal. In SPI serial master mode, if this pin is driven low by another master device on the bus, it will cause a mode fault to occur.	96	6	Input
SCP1_CLK	SPI Control Port Bit Clock In master mode, this pin serves as the serial control clock output. In serial slave mode, this pin serves as the serial control clock input.	99	126	I/O
SCP1_MOSI	SPI Mode Master Data Output/Slave Data Input SCP1_MOSI in SPI slave mode this pin serves an the data input, in SPI master mode this pin serves as the data output.	95	3	I/O
SCP1_MISO	SPI Mode Master Data Input/Slave Data Output In SPI slave mode this pin serves as the data input. In SPI master mode this pin serves as the data output.	97	2	I/O
$\overline{\text{SCP1\_IRQ}}$	Serial Control Port Data Ready Interrupt Request Output, Active Low This pin is driven low when the DSP has a message for the host to read. The pin will go high when the host has read the message and the DSP has no further messages. This pin reflects the state of the SCP1 port Transmit Buffer Empty Flag.	100	4	Open Drain
$\overline{\text{SCP1\_BSY}}$	Serial Control Port 1 Input Busy, Output, Active Low This pin is driven low when the control port's receive buffer is full. This pin reflects the state of the SCP1 or PCP Receive Buffer Full Fag.	102	128	Open Drain
$\overline{\text{SCP2\_CS}}$	SPI Chip Select, Active Low In serial SPI slave mode, this pin is used as the active-low chip-select input signal. In SPI serial master mode, if this pin is driven low by another master device on the bus, it will cause a mode fault to occur.	104	7	Input
SCP2_CLK	SPI Control Port Bit Clock In master mode, this pin serves as the serial control clock output. In serial slave mode, this pin serves as the serial control clock input.	103	1	I/O
SCP2_MISO	SPI Mode Master Data Input/Slave Data Output In SPI slave mode this pin serves as the data input. In SPI master mode this pin serves as the data output.	105	2	I/O
SCP2_MOSI	SPI Mode Master Data Output/Slave Data Input SCP2_MOSI in SPI slave mode this pin serves as the data input, in SPI master mode this pin serves as the data output.	106	3	I/O
$\overline{\text{EE\_CS}}$	Master Mode Serial EPROM Chip Select, Active Low	6, 14, 25, or 121	38, 46, or 14	Output

Table 3-2. Serial Control Port SPI Signals (Continued)

Pin Name	Pin Description	LQFP-144 Pin #	LQFP-128 Pin #	Pin Type
$\overline{\text{SCP2\_IRQ}}$	Serial Control Port Data Ready Interrupt Request Output, Active Low This pin is driven low when the DSP has a message for the host to read. The pin will go high when the host has read the message and the DSP has no further messages. This pin reflects the state of the SCP1 port Transmit Buffer Empty Flag.	108	5	Open Drain

3.4.1 SPI System Bus Description

The SPI bus is a multi-master bus. This means that more than one device capable of controlling the bus can be connected to it. Generation of clock signals on the SPI bus is always the responsibility of master devices; each master generates its own clock signals when transferring data on the bus. Bus clock signals from a master cannot be altered by any other device on the bus, otherwise a collision will occur. The slave chip-select signals can only be controlled by master devices.

The CS4953xx has two serial ports. However, the O/S currently supports only slave mode host communication on SCP1, and master mode communication on SCP2 for booting from a serial EEPROM/ FLASH.

SCP1_MOSI (Master Out/Slave In) and SCP1_MISO (Master In/Slave Out) are bidirectional lines that change their behavior depending on whether the device is operating in master or slave mode. Only the master can drive the MOSI signal while only the slave can drive the MISO signal.

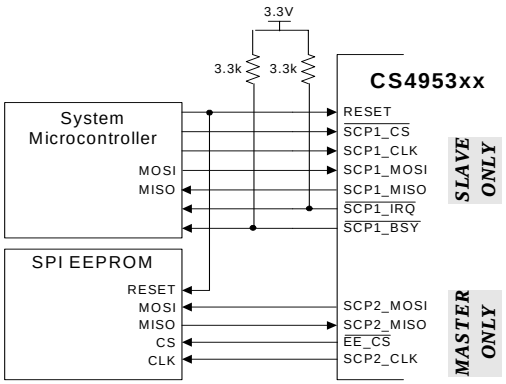


Figure 3-13. Block Diagram of SPI System Bus

As seen in Figure 3-12, two serial ports are available on the CS4953xx. Each can be configured to either a master or slave. For audio applications, SCP1 is configured as a slave port and SCP2 is configured as a master port. SCP2 is used only in systems that are booting from serial EEPROM.

3.4.2 SPI Bus Dynamics

A SPI transaction begins by the master driving the slave chip select $\overline{\text{SCP1\_CS}}$ low. SPI transactions end by the master driving the $\overline{\text{SCP1\_CS}}$ high. This SPI bus is considered busy while any device's $\overline{\text{SCP1\_CS}}$ signal is low. The bus is free only when all slave $\overline{\text{SCP1\_CS}}$ signals are high. A high-to-low transition on the $\overline{\text{SCP1\_CS}}$ line defines an SPI Start condition. A low-to-high transition on the $\overline{\text{SCP1\_CS}}$ line defines an SPI Stop condition. Start and Stop conditions are always generated by the master. The bus is considered to be busy after the Start condition. The bus is considered to be free again following the Stop condition.

The data bits of the SCP1_MOSI and SCP1_MISO line are valid on the rising edge of SCP1_CLK. It is the slave's responsibility to accept or supply bytes on the bus at the rate at which the master is driving SCP1_CLK.

All data put on the SCP1_MOSI and SCP1_MISO lines must be in 8-bit bytes. The number of bytes that can be transmitted per transfer is unrestricted. Data is transferred with the most-significant bit (MSB) first. For the CS4953xx slave SPI port, the first byte is an address byte that is always sent by the master after a Start condition. This address byte is an "I^2C-type" command of a 7-bit address + a R/$\overline{W}$ bit. The 7-bit SPI address is 1000000b (0x80).

If the SPI transaction is a write from master to the CS4953xx (R/$\overline{W}$ = 0, Address = 0x80), then the master will clock the SCP1_CLK signal and drive the SCP1_MOSI signal with data bytes for the CS4953xx to read. If the SPI transaction is a read to the master from the CS4953xx (R/$\overline{W}$ = 1, Address = 0x81), then the master will drive the SCP1_CLK signal and read the SCP1_MISO signal with the data bytes from the CS4953xx.

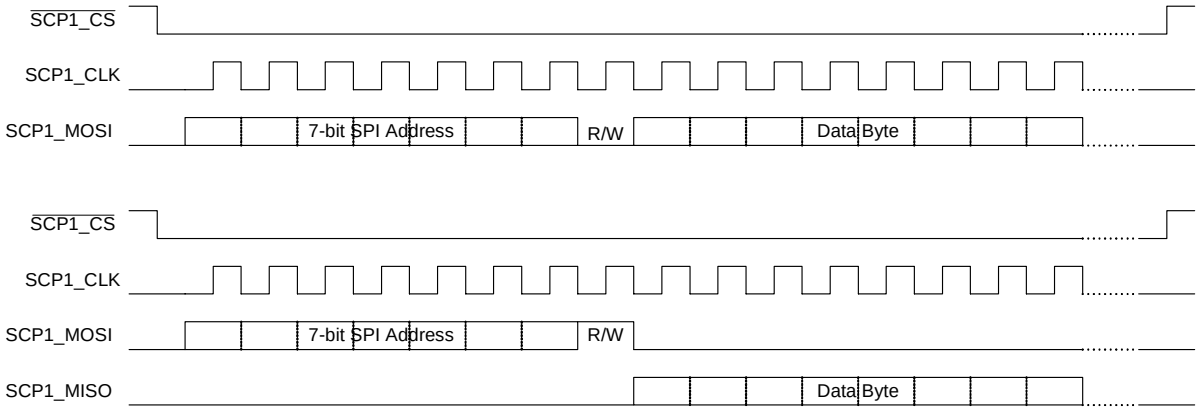


Figure 3-14. Address and Data Bytes

3.4.2.1 $\overline{\text{SCP1\_BSY}}$ Behavior

The $\overline{\text{SCP1\_BSY}}$ signal is not part of the SPI protocol, but it is provided so that the slave can signal to the master that it cannot receive any more data. A falling edge of the $\overline{\text{SCP1\_BSY}}$ signal indicates the master must halt transmission. Once the $\overline{\text{SCP1\_BSY}}$ signal goes high, the suspended transaction may continue. The host must obey the $\overline{\text{SCP1\_BSY}}$ pin or control data will be lost.

3.4.3 SPI Messaging

Messaging to the CS4953xx using the SPI bus requires usage of all the information provided in the *SPI Bus Description* and *Bus Dynamics* above. For control and application image downloading, SPI transactions to the CS4953xx will involve 4-byte words. A detailed description of the serial SPI communication mode is provided in this section. This includes:

- A flow diagram and description for a serial SPI write
- A flow diagram and description for a serial SPI read

3.4.3.1 Performing a Serial SPI Write

Information provided in this section is intended as a functional description indicating how to perform an SPI write from an external device (master) to the CS4953xx DSP (slave). The system designer must ensure that all timing constraints of the SPI Write Cycle are met (see the CS4953xx datasheet for timing specifications). When performing an SPI write, the same protocol is used whether writing single-word messages to the boot firmware, writing multiple-word overlay images to the boot firmware, or writing multiple-word messages to the application firmware. The example shown in this section can be generalized to fit any SPI write situation.

The flow diagram shown in Figure 3-15 illustrates the sequence of events that define the SPI write protocol. Section 3.4.3.2. describes the Serial SPI Write protocol.

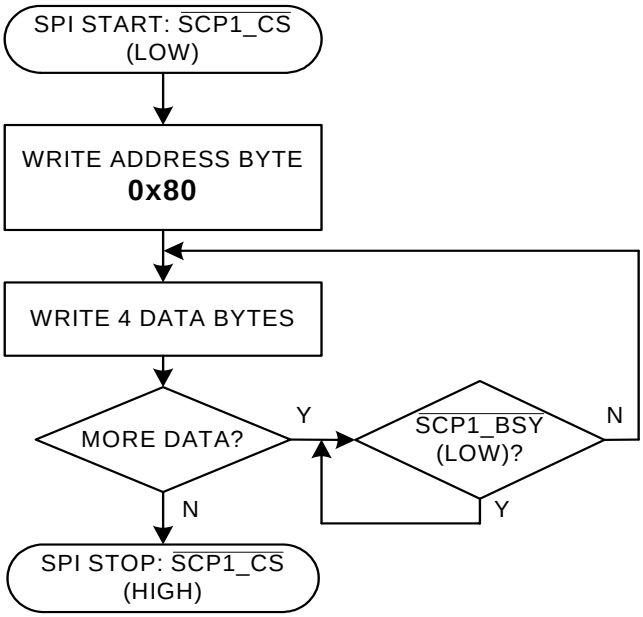


Figure 3-15. SPI Write Flow Diagram

3.4.3.2 SPI Write Protocol

1. An SPI transfer is initiated when the chip select $\overline{\text{SCP1\_CS}}$ is driven low. $\overline{\text{SCP1\_CS}}$ driven low indicates that CS4953xx is in SPI slave mode.
2. This is followed by a 7-bit address and the read/write bit set low for a write. So, the master should send 0x80. The 0x80 byte represents the 7-bit SPI address 1000000b, and the least significant bit set to '0', designates a write.
3. The master should then clock the 4-byte data word into the slave device, most-significant bit first, one byte at a time. The data byte is transferred to the CS4953xx DSP (slave) on the falling edge of the eighth serial clock. For this reason, the serial clock should be held low so that eight transitions from low-to-high-to-low will occur for each byte.
4. If the master has no more data words to write to the CS4953xx, then proceed to Step 6. If the master has more data words to write to the CS4953xx, then proceed to Step 5.

5. The master should poll the $\overline{\text{SCP1\_BSY}}$ signal until it goes high. If the $\overline{\text{SCP1\_BSY}}$ signal is low, it indicates that the CS4953xx is busy performing some task that requires halting the serial control port. Once the CS4953xx is able to receive more data words, the $\overline{\text{SCP1\_BSY}}$ signal will go high. Once the $\overline{\text{SCP1\_BSY}}$ signal is high, proceed to Step 3.
6. The master finishes the SPI write transaction by driving the CS4953xx $\overline{\text{SCP1\_CS}}$ signal high.

3.4.3.3 Performing a Serial SPI Read

Information provided in this section is intended as a functional description indicating how an external device (Master) performs an SPI read from the CS4953xx (slave). The system designer must ensure that all timing constraints of the SPI read cycle are met (see the CS4953xx datasheet for timing specifications).

When performing a SPI read, the same protocol is used whether reading a single byte or multiple bytes. From a hardware perspective, it makes no difference whether communication is a single byte or multiple bytes of any message length, so long as the correct hardware protocol is followed. The example shown in this section can be generalized to fit any SPI read situation.

The flow diagram shown in Figure 3-16, illustrates the sequence of events that define the SPI read protocol. The Serial SPI read protocol is described in Section 3.4.3.4..

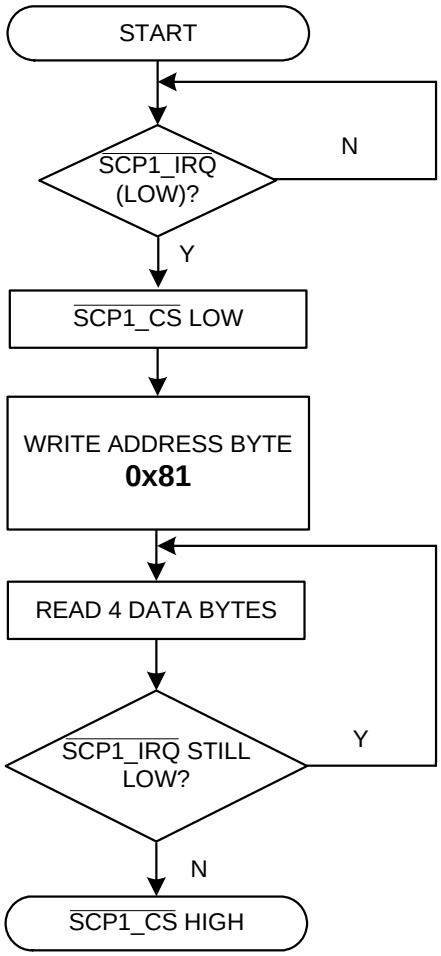


Figure 3-16. SPI Read Flow Diagram

3.4.3.4 SPI Read Protocol

1. An SPI read transaction is initiated by the CS4953xx slave driving $\overline{\text{SCP1\_IRQ}}$ low to indicate that it has data to be read.
2. The master begins a SPI transaction driving chip select ($\overline{\text{SCP1\_CS}}$) low.
3. This is followed by a 7-bit address and the read/write bit set high for a read. So, the master should send 0x81. The 0x81 byte represents the 7-bit SPI address 1000000b, and the least significant bit set to '1', designates a read.
4. After the falling edge of the serial control clock (SCP1_CLK) for the read/write bit, the master can begin clocking out the 4-byte word from the CS4953xx on the MISO pin. Data clocked out of the CS4953xx by the master is valid on the rising edge of SCP1_CLK and data transitions occur on the falling edge of SCP1_CLK. The serial clock should be held low so that eight transitions from low-to-high-to-low will occur for each byte.
5. If $\overline{\text{SCP1\_IRQ}}$ is still low after 4 bytes, then proceed to Step 4 and read another 4 bytes out of the CS4953xx slave.
6. If $\overline{\text{SCP1\_IRQ}}$ is high, the $\overline{\text{SCP1\_CS}}$ line of CS4953xx should be driven high to end the read transaction.

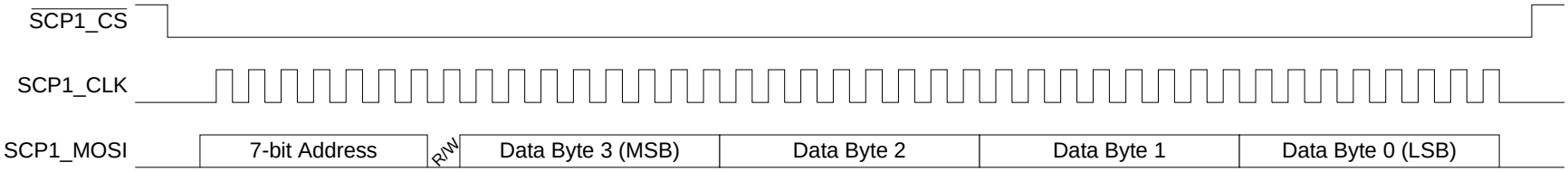


Figure 3-17. Sample Waveform for SPI Write Functional Timing

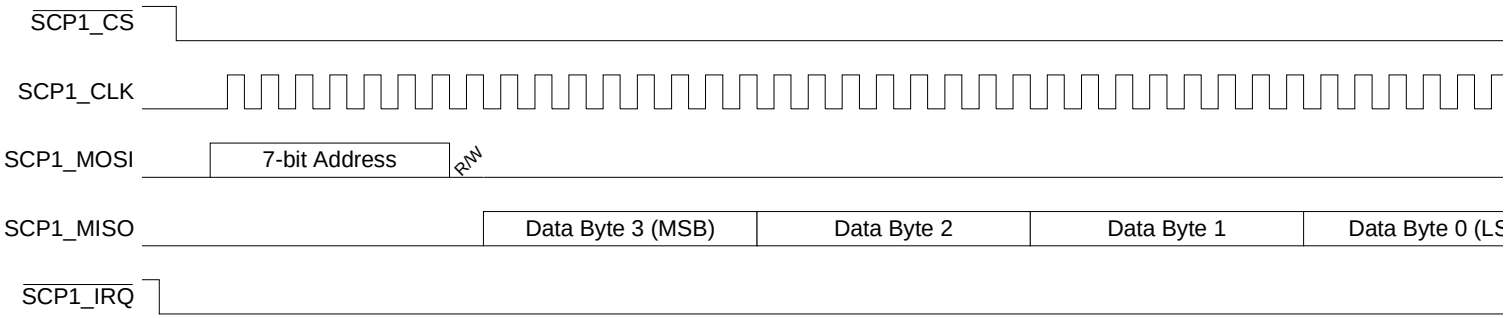


Figure 3-18. Sample Waveform for SPI Read Functional Timing

Notes:

1. IRQ remains low until the rising edge of the clock for the last bit of the last byte to be read from the SPI slave.
2. After going high, IRQ remains high until the $\overline{CS}$ signal is raised to end the SPI transaction. If there are more bytes to read, IRQ will fall after $\overline{CS}$ has gone high.

3.4.3.5 SCP1_IRQ Behavior

The SCP1_IRQ signal is not part of the SPI protocol, but is provided so that the slave can signal that it has data to be read. A high-to-low transition on SCP1_IRQ indicates to the master that the slave has data to be read. When a master detects a high-to-low transition on SCP1_IRQ, it should send a Start condition and begin reading data from the slave.

SCP1_IRQ is guaranteed to remain low (once it has gone low), until the rising edge of SCP1_CLK for the last bit of the last byte to be transferred out of CS4953xx. If there is no more data to be transferred, SCP1_IRQ will go high at this point. After going high, SCP1_IRQ is guaranteed to stay high until the rising edge of SCP1_CS.

This end-of-transfer condition signals the master to end the read transaction by clocking the last data bit out of CS4953xx and then driving the CS4953xx SCP1_CS line high to signal that the read sequence is over. If SCP1_IRQ is still low after the rising edge of SCP1_CLK on the last data bit of the current byte, the master should continue reading data from the serial control port. It should be noted that all data should be read out of the serial control port during one cycle or a loss of data will occur. In other words, all data should be read out of the chip until SCP1_IRQ signals the last byte by going high as described above.

§§

Chapter 4
Parallel Control Port

4.1 Parallel Control Availability

Note: The Parallel Control Port is currently not supported in the O/S. Please contact your sales representative if you need to use this port.

The CS4953xx is equipped with an 8-bit Parallel Control Port that can be used for host communication, providing faster control throughput for the system. The Parallel Control Port is capable of Intel, Motorola, and Multiplexed Intel communication modes.

Note: The Parallel Control Port is not available on CS4953xx products using the 128-Pin LQFP package

§§

Chapter 5

Digital Audio Input Interface

CS4953xx supports a wide variety of audio data formats through various input and output ports. Hardware availability is entirely dependent on whether the software application code being used supports the required mode. This data sheet presents most of the modes available with the CS4953xx hardware. However, not all of the modes are available with any particular piece of application code. The Application Code User's Guide for the particular code being used should be referenced to determine if a particular mode is supported. In addition, if a particular mode is desired that is not presented, please contact your sales representative regarding its availability.

5.1 Digital Audio Input Port Description

The CS4953xx Digital Audio Input ports (DAI1 and DAI2) are designed to provide five digital audio input pins (10 channels total) that can be configured to load audio samples in a number of formats, or accept multiple stereo channels on a single input port.

DAI features include:

- Five digital audio input pins capable of supporting many audio formats
- Two independent input clock domains (DAI1_SCLK/DAI1_LRCLK or DAI2_SCLK/DAI2_LRCLK)
- Up to 32-bit data widths
- Sample rates up to 192 kHz
- Two simultaneous serial compressed audio data inputs in IEC61937 format for dual-decode support
- 2, 4, or 6 channels on a single pin (DAI_DATAx)
- Simultaneous 8-channel PCM for multichannel DVD audio and compressed audio

5.1.1 DAI Pin Description

The digital audio input port (DAI) on CS4953xx, is used for both compressed and PCM digital audio data input. Table 5-1 shows the mnemonic and pin description of the pins associated with the DAI port on CS4953xx.

Table 5-1. Digital Audio Input Port

Pin Name	Pin Description	LQFP-144 Pin #	LQFP-128 Pin #	Pin Type
DAI1_LRCLK	Sample Rate Clock 1 PCM Audio Input Sample Rate (LeftRight) Clock DAI1_LRCLK is the sample rate input clock for the serial PCM audio data on DAI_DATA[3:0].	138	30	Input
DAI1_SCLK	Bit Clock 1 PCM Audio Input Bit Clock DAI1_SCLK is the bit clock input for the serial PCM audio data on DAI_DATA[3:0]..	137	29	Input

Table 5-1. Digital Audio Input Port (Continued)

Pin Name	Pin Description	LQFP-144 Pin #	LQFP-128 Pin #	Pin Type
DAI1_DATA0	PCM or Compressed Audio Input Data 0 PCM Audio Input Data 0 Serial data input that can accept PCM audio data that is synchronous to DAI_SCLK1/DAI_LRCLK1 or DAO1_SCLK/DAO1_LRCLK..	135	27	Input
DAI1_DATA1	PCM Audio Input Data 1	134	26	Input
DAI1_DATA2	PCM Audio Input Data 2	132	24	Input
DAI1_DATA3	PCM Audio Input Data 3	131	23	Input
DAI2_LRCLK	Sample Rate Clock 2 PCM Audio Input Sample Rate (LeftRight) Clock DAI2_LRCLK is the sample rate input clock for the serial PCM audio data on DAI2_DATA.	140	32	Input
DAI2_SCLK	Bit Clock 2 PCM Audio Input Bit Clock DAI2_SCLK is the bit clock input for the serial PCM audio data on DAI2_DATA.	141	33	Input
DAI2_DATA or DAI1_DATA4	PCM or Compressed Audio Input Data PCM Audio Input Data DAI2_DATA is the PCM audio data input synchronous to DAI2_SCLK/ DAI2_LRCLK	142	34	Input

5.1.2 Supported DAI Functional Blocks

The CS4953xx DAI has many functional blocks for realizing various audio system configurations. The use of these functions is dependent on the firmware currently available on the CS4953xx. Figure 5-1 shows the functional block diagram of the features currently supported with the CS4953xx DAI.

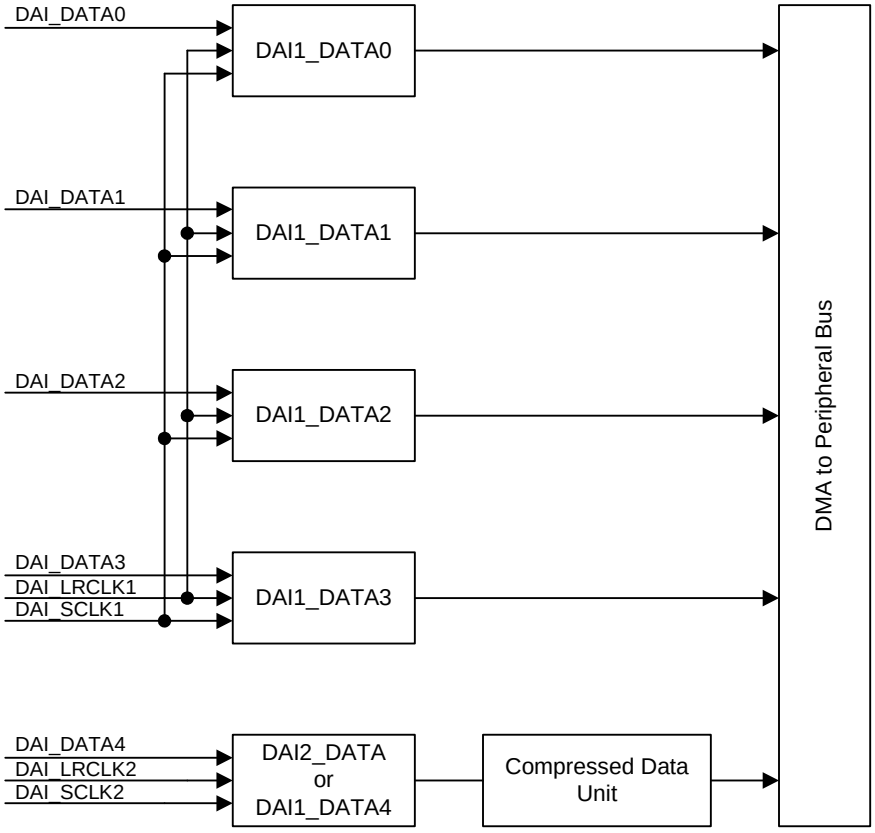


Figure 5-1. DAI Port Block Diagram

Currently supported are 4 lines of linear PCM input (DAI_DATA[3:0]) and 1 line of compressed audio or linear PCM (DAI_DATA4). These two inputs can have their own clock domains. The firmware currently available can operate on only one of these inputs at a time, providing for compressed data decode, stereo PCM processing, or multichannel PCM processing. Please see AN288, *"CS4953xx Firmware User's Manual"* for details about configuring the firmware to select these different inputs to process.

5.1.3 BDI Port

Note: Currently not supported in the O/S.

The Bursty Data Input (BDI) port on the CS4953xx shares pins with the DAI port pins, and is used for input of bursty compressed audio data. The compressed data is clocked in with a bit clock (BDI_CLK). Bursty compressed audio data input requires the use of a "throttle" signal, BDI_REQ to signal to the host that the CS4953xx is capable of accepting data. Table 5-1 shows the mnemonic and pin description of the pins associated with the Bursty Data Input (BDI) port on CS4953xx.

Table 5-2. Bursty Data Input (BDI) Pins

Pin Name	Pin Description	LQFP-144 Pin #	LQFP-128 Pin #	Pin Type
BDI_REQ	Data Request, Active Low BDI_REQ is the bursty delivery flow control output for bursty audio data. It indicates whether the DSP can accept more data.	140	32	Output
BDI_CLK	Bit Clock 2 Bursty Audio Input Bit Clock BDI_CLK is the bit clock input for the bursty serial audio data on BDI_DATA.	141	33	Input
BDI_DATA	Compressed Audio Bursty Input Data BDI_DATA is the serial bursty audio data input that corresponds to the BDI_CLK serial bit clock..	142	34	Input

5.1.4 Digital Audio Formats

The DAI has 5 stereo data input pins that are fully configurable including support for I^2S, and left-justified formats. DAI ports are programmed for slave operation, where DAIn_LRCLK and DAIn_SCLK are inputs only. This subsection describes some common audio formats that CS4953xx supports. It should be noted that the input ports use up to 32-bit PCM resolution and 16-bit compressed data word lengths.

5.1.4.1 I^2S Format

Figure 5-2 illustrates the I^2S format. For I^2S, data is presented most-significant bit (MSB) first, one SCLK delay after the transition of DAIn_LRCLK, and is valid on the rising edge of DAIn_SCLK. For the I^2S format, the left subframe is presented when DAIn_LRCLK is low, and the right subframe is presented when DAIn_LRCLK is high.

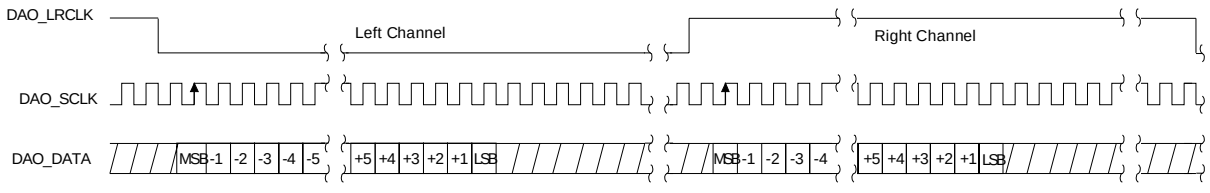


Figure 5-2. I^2S format (Rising Edge Valid SCLK)

5.1.4.2 Left-Justified Format

Figure 5-3 illustrates the left-justified format with a rising-edge DAIn_SCLK. Data is presented most-significant bit first on the first DAIn_SCLK after a DAIn_LRCLK transition and is valid on the rising edge of DAIn_SCLK. For the left-justified format, the left subframe is presented when DAIn_LRCLK is high and the right subframe is presented when DAIn_LRCLK is low. The left-justified format can also be programmed for data to be valid on the falling edge of DAIn_SCLK.

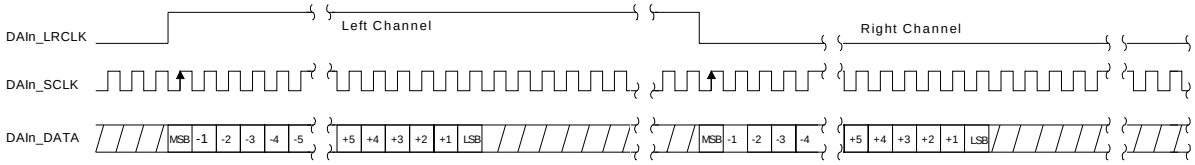


Figure 5-3. Left-justified Format (Rising Edge Valid SCLK)

5.2 DAI Hardware Configuration

After code download or soft reset, and before kickstarting the application, the host has the option of changing the default hardware configuration. (Please see AN288, *"CS4953xx/CS497xxx Firmware User's Manual"* for more information on kickstarting). In general, the hardware configuration can only be changed immediately after download or after soft reset.

Hardware configuration messages are used to physically reconfigure the hardware of the audio decoder, as when enabling or disabling address checking for the serial communication port. Hardware configuration messages are also used to initialize the format (for example, I^2S, left justified, and others) for digital data inputs, as well as the data format and clocking options for the digital output port.

5.2.1 DAI Hardware Naming Convention

The naming convention of the input hardware configuration is as follows:

INPUT *A B C D*

Where *A*, *B*, *C,* and *D* are the parameters used to fully define the input port. The parameters are defined as follows:

- **A** - Data Format
- **B** - SCLK Polarity
- **C** - LRCLK Polarity.
- **D** - DAI Mode

Table 5-3, Table 5-4, Table 5-5, and Table 5-6 show the different values for each parameter as well as the hex message that needs to be sent to configure the port. When creating the hardware configuration message, only one hex message should be sent per parameter. .

Table 5-3. Input Data Format Configuration (Input Parameter A)

A Value	Data Format	Hex Message
0 (default)	I^2S 24-bit	Starting from DAI_D0 to DAI_D4: 0x81800020 0xFFFF0000 0x81400020 0x01001F00 0x81800021 0xFFFF0000 0x81400021 0x01001F00 0x81800022 0xFFFF0000 0x81400022 0x01001F00 0x81800023 0xFFFF0000 0x81400023 0x01001F00 0x81800024 0xFFFF0000 0x81400024 0x01001F00
1	Left Justified 24-bit	Starting from DAI_D0 to DAI_D4: 81800020 FEEF0000 81400020 00001F00 81800021 FEEF0000 81400021 00001F00 81800022 FEEF0000 81400022 00001F00 81800023 FEEF0000 81400023 00001F00 81800024 FEEF0000 81400024 00001F00

Table 5-3. Input Data Format Configuration (Input Parameter A) (Continued)

A Value	Data Format	Hex Message
2	DSD Normal Mode	Starting from DAI_D0 to DAI_D4: 0x81000020 0x00001F00 0x81000021 0x00001F00 0x81000022 0x00001F00 0x81000023 0x00001F00 0x81000024 0x00001F00 0x81000012 0x00001F00 0x81000014 0x00000000 0x81000013 0x83F01F0B 0x81000025 0x1008D11F

Table 5-4. Input SCLK Polarity Configuration (Input Parameter B)

B Value	SCLK Polarity (Both DAI and CDI Port)	Hex Message
0 (default)	Data Clocked in on SCLK Rising Edge	Starting from DAI_D0 to DAI_D4: 0x81800020 0xFFDFFFFF 0x81800021 0xFFDFFFFF 0x81800022 0xFFDFFFFF 0x81800023 0xFFDFFFFF 0x81800024 0xFFDFFFFF
1	Data Clocked in on SCLK Falling Edge	Starting from DAI_D0 to DAI_D4: 0x81400020 0x00200000 0x81400021 0x00200000 0x81400022 0x00200000 0x81400023 0x00200000 0x81400024 0x00200000

..

Table 5-5. Input LRCLK Polarity Configuration (Input Parameter C)

C Value	LRCLK Polarity (Both DAI and CDI Port)	HEX Message
0	LRCLK=High indicates Channel 0 (i.e. Left)	Starting from DAI_D0 to DAI_D4: 0x81800020 0xFFEFFFFF 0x81800021 0xFFEFFFFF 0x81800022 0xFFEFFFFF 0x81800023 0xFFEFFFFF 0x81800024** 0xFFEFFFFF
1(default)	LRCLK=Low indicates Channel 1 (i.e. Left)	Starting from DAI_D0 to DAI_D4: 0x81400020 0x00100000 0x81400021 0x00100000 0x81400022 0x00100000 0x81400023 0x00100000 0x81400024 0x00100000

** Required for 1-line software header finder in I^2S format. For more information about the software header finder, see index 0x0001 in Table 7 of AN288, *CS4953xx/CS497xxx Firmware User's Manual.*

Table 5-6. Input DAI Mode Configuration (Input Parameter D)

D Value	Description	HEX Message
0 (default)	DAI2_LRCLK/SCLK – Slave Compressed Data Input on DAI_D4	0x81000025 0X1008D110
1	DAI1_LRCLK/SCLK - Slave Compressed Data Input on DAI_D0 with only DAI_D4 enabled	0x81000025 0X0000D190
2	DAI2_LRCLK/SCLK - Slave PCM Data Input on DAI_D4 Routed to Center and LFE	0x81000025 0X1558E11F
3	DAI1_LRCLK/SCLK - Slave PCM Data Input on DAI_D0 Routed to Center and LFE	0x81000025 0X1008C10F
4	DAI1_LRCLK/SCLK - Slave Compressed Data on DAI_D0 and DAI_D0 to DAI_D4 enabled	0x81000025 0X0000D11F

§§

Chapter 6

Direct Stream Data (DSD) Input Interface

CS4953xx is capable of accepting Direct Stream Data (DSD) audio data directly. DSD data differs from PCM in that audio is provided as a contiguous stream of 1's and 0's on a single line. There is no framing clock (LRCLK), and there is only one channel per line. The CS4953xx supports internal conversion of DSD data to PCM which can then be processed by the DSP.

6.1 Description of Digital Audio Input Port when Configured for DSD Input

The CS4953xx DSD port is designed to accept DSD audio data from up to 6 pins simultaneously (6 channels total)

DSD features include:

- Six DSD Input Pins
- One Shared DSD_CLK for All Data Pins
- Supports 44.1 kHz (1 Fs), 88.2 kHz (2 Fs), and 176.4 kHz (4 Fs) Sample Rates

6.1.1 DSD Pin Description

Table 6-1 shows the mnemonic and pin description of the pins associated with the DSD port on CS4953xx.

Table 6-1. DSDl Audio Input Port

Pin Name	Pin Description	LQFP-144 Pin #	LQFP-128 Pin #	Pin Type
DSD_CLK	Bit clock used for latching the DSD audio data. This clock is shared by DSD[5:0].	137	29	Input
DSD0	DSD Audio Input 0	135	27	Input
DSD1	DSD Audio Input 1	134	26	Input
DSD2	DSD Audio Input 2	132	24	Input
DSD3	DSD Audio Input 3	131	23	Input
DSD4	DSD Audio Input 4	142	34	Input
DSD5	DSD Audio Input 5	138	30	Input

6.1.2 Supported DSD Functional Blocks

Figure 6-1 below shows the functional block diagram of the features currently supported with the CS4953xx DSD Port.

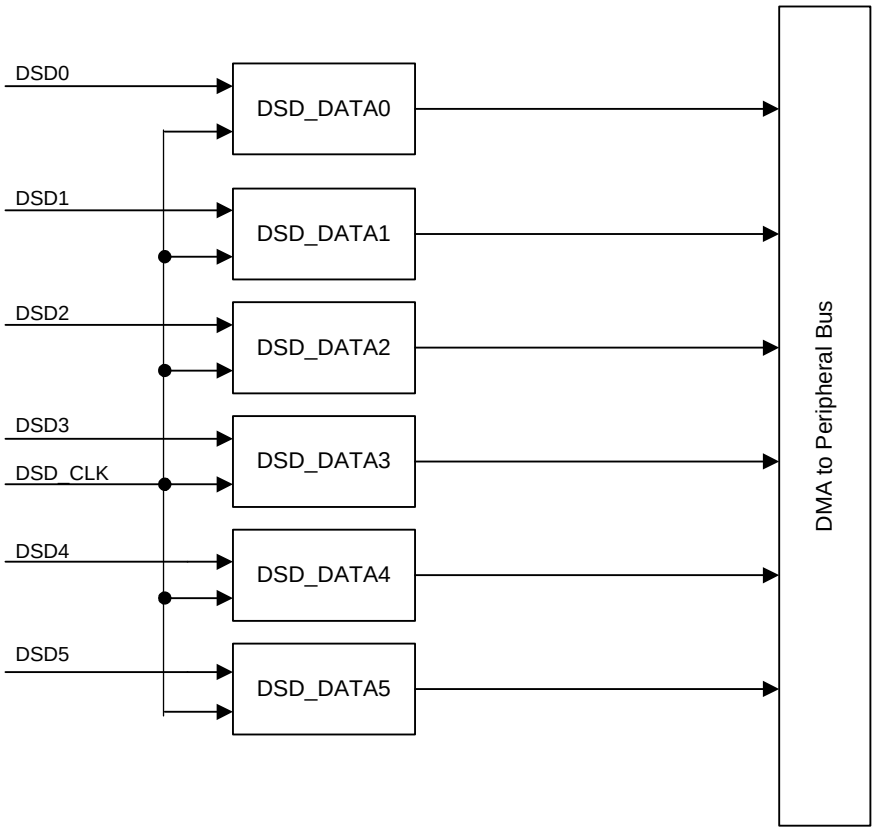


Figure 6-1. DSD Port Block Diagram

§§

Chapter 7

Digital Audio Output Interface

The CS4953xx has two output ports - Digital Audio Output port 1 & 2 (DAO1 & DAO2). Each port can output 8 channels of up to 32-bit PCM data. The Digital Audio Output ports are both implemented with a modified 3-wire Inter-IC Sound (I^2S) interface along with an oversampling master clock (MCLK). The I^2S interface includes a frame clock at the current sampling frequency (LRCLK), a bit clock for clocking the bits of the audio word (SCLK), and 4 audio data output signals (DATA[3:0]). Each of the output data signals can be connected to the digital stereo input of an audio digital-to-analog converter (DAC) for up to 8 channels of stereo PCM output per DAO port for a total of 16 digital audio output channels.

Each DAO port may slave to an externally generated SCLK and LRCLK or it may master these clocks if MCLK is provided. Each port can be configured as having an independent clock domain in slave mode, or the ratio of the two clocks can be set to even multiples of each other in master mode. Additionally, the two ports can be ganged together into a single clock domain. The port supports data rates from 32 kHz to 192 kHz. Each port can also be configured to provide a 32-kHz to 192-kHz S/PDIF transmitter (XMTA and XMTB) as an output. Figure 7-1 illustrates the DAO block diagram.

7.1 Digital Audio Output Port Description

7.1.1 DAO Pin Description

Figure 7-1 identifies the pins associated with the Digital Audio Output Ports (DAO1 and DAO2).

Table 7-1. Digital Audio Output (DAO1 & DAO2) Pins

Pin Name	Pin Description	LQFP-144 Pin #	LQFP-128 Pin #	Pin Type
DAO1_LRCLK	Sample Rate Clock	22	54	I/O
DAO1_SCLK	Bit Clock	20	52	I/O
DAO1_DATA0	Digital Audio Output	19	51	Output
DAO1_DATA1	Digital Audio Output	17	49	Output
DAO1_DATA2	Digital Audio Output	16	48	Output
DAO1_DATA3/XMTA	Digital Audio Output	15	47	Output
DAO2_LRCLK	Sample Rate Clock	14	46	I/O
DAO2_SCLK	Bit Clock	12	44	I/O
DAO2_DATA0	Digital Audio Output	11	43	Output
DAO2_DATA1	Digital Audio Output	7	39	Output
DAO2_DATA2	Digital Audio Output	6	38	Output
DAO2_DATA3/XMTB	Digital Audio Output	5	35	Output
DAO_MCLK	Master Clock	8	40	I/O

DAO_MCLK is the master clock and is firmware configurable to be either an input (slave) or an output (master). If MCLK is to be used as an output, the internal PLL must be used. As an output MCLK can be configured to provide a 128Fs, 256Fs, or 512Fs clock, where Fs is the output sample rate.

DAO1_SCLK is the bit clock used to clock data out on DAO1_DATA[3:0].

DAO1_LRCLK is the data framing clock whose frequency is equal to the sampling frequency for the DAO1 data outputs.

DAO1_DATA[3:0] are the data outputs and are typically configured for outputting two channels of I^2S or left-justified PCM data. DAO1_DATA0 may also be configured to provide output for four or six channels of PCM data. The DAO1_DATA3 (XMTA) pin can alternatively serve as an S/PDIF transmitter output.

DAO2_SCLK is the bit clock used to clock data out on DAO2_DATA[3:0].

DAO2_LRCLK is the data framing clock which has a frequency equal to the sampling frequency for the DAO2 data outputs.

DAO2_DATA[3:0] are the data outputs and are typically configured for outputting two channels of I^2S or left-justified PCM data. DAO2_DATA0 may also be configured to provide output for four or six channels of PCM data. The DAO2_DATA3 (XMTB) pin can alternatively serve as an S/PDIF transmitter output..

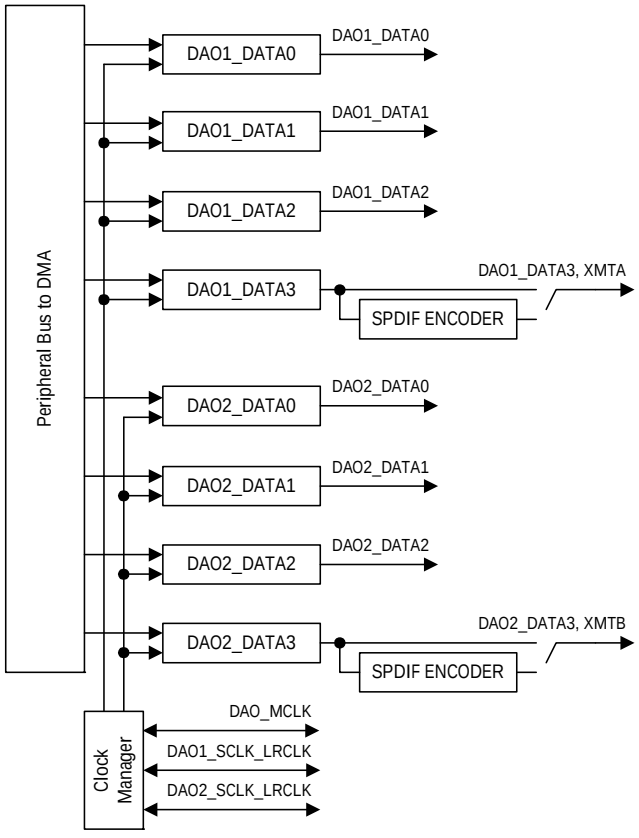


Figure 7-1. DAO Block Diagram

7.1.2 Supported DAO Functional Blocks

As mentioned earlier in the previous section, two DAO ports, DAO1_DATA3 and DAO2_DATA3, are unique in that they are designed to serve as either an output for I^{2}S or left-justified PCM data or as S/PDIF transmitters (XMTA and XMTB). When configured as S/PDIF transmitters, the ports encode digital audio data according to the Sony®/Philips® Digital Interface Format (S/PDIF), also known as IEC60958, or the AES/EBU interface format. Please see Figure 7-1 for a block diagram of the supported functional blocks for the DAO on the CS4953xx

7.1.3 DAO Interface Formats

The DAO interface has 8 stereo data outputs that are fully configurable including support for I^{2}S, left-justified, and multichannel (one-line data mode) formats. This section describes some common audio formats that the CS4953xx DAO interface supports. It should be noted that the digital output ports provide up to 32-bit PCM resolution.

Note: DAO1_DATAx is valid with respect to DAO1_SCLK/DAO1_LRCK and DAO2_DATAx is valid with respect to DAO2_SCLK/DAO2_LRCK

7.1.3.1 I^{2}S Format

In this format, data is presented most-significant bit (MSB) first, one DAO_SCLK delay after the transition of DAO_LRCLK, and is valid on the rising edge of DAO_SCLK. The left subframe is presented when DAO_LRCLK is low, and the right subframe is presented when DAO_LRCLK is high. Figure 7-2 provides details on I^{2}S compatible (maximum of 32 bits) serial audio formats.

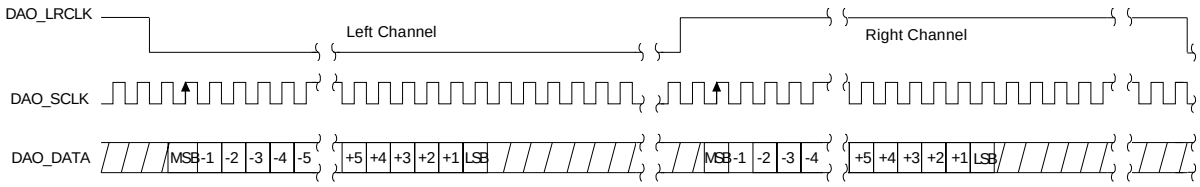


Figure 7-2. I^{2}S Compatible Serial Audio Formats (Rising Edge Valid)

7.1.3.2 Left-Justified Format

Figure 7-3 illustrates the left-justified (LJ) format with a rising edge DAO_SCLK. Data is presented most-significant bit first on the first DAO_SCLK after a DAO_LRCLK transition and is valid on the rising edge of DAO_SCLK. The left subframe is presented when DAO_LRCLK is high and the right subframe is presented when DAO_LRCLK is low. This format can also be programmed for data to be valid on the falling edge of DAO_SCLK.

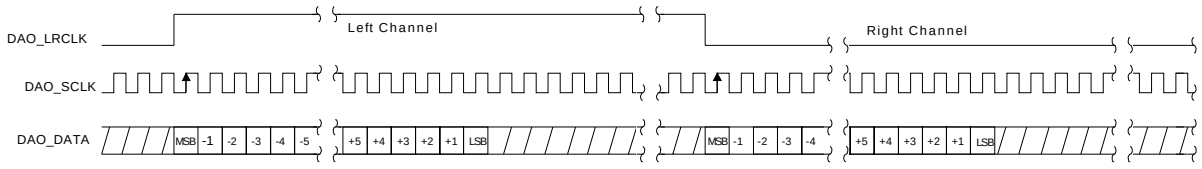


Figure 7-3. Left-justified Digital Audio Formats (Rising Edge Valid DAO_SCLK)

7.1.3.3 One-line Data Mode Format (Multichannel)

The CS4953xx is capable of multiplexing all digital audio outputs on one line, as illustrated in Figure 7-4. This mode is available only through special request. Please contact your local Cirrus representative for further details.

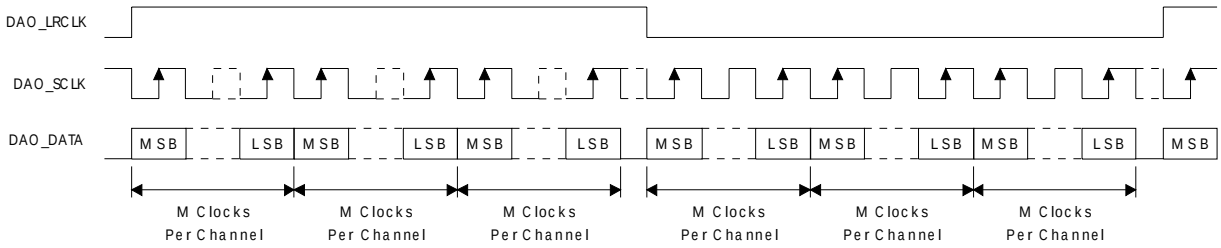


Figure 7-4. One-line Data Mode Digital Audio Formats

7.1.4 DAO Hardware Configuration

The DAO naming convention is as follows:

OUTPUT *A B C D E F*

where the parameters are defined as:

- **A** - DAO Clock Mode (Master/Slave for DAO_LRCLK and DAO_SCLK)
- **B** - DAO1/DAO2 Clock Relationship
- **C** - DAO_MCLK, DAO_SCLK, DAO_LRCLK Frequency Ratio
- **D** - Data Format (I2S, Left Justified)
- **E** - DAO_LRCLK Polarity
- **F** - DAO_SCLK Polarity
- **G** - Output Channel Configuration

Table 7-2, Table 7-3, Table 7-4, Table 7-5, Table 7-6, and Table 7-7 show the different values for each parameter as well as the hex message that needs to be sent to configure the port. When creating the hardware configuration message, only one hex message should be sent per parameter.

Note: All DAO hardware config messages must be sent to DSPB. For more details on sending messages to DSP B refer to AN288, section 2.1.4

Table 7-2 shows values and messages for DAO output clock mode configuration parameters.

Table 7-2. Output Clock Mode Configuration (Parameter A)

A Value	DAO 1 & 2 Modes (MCLK, LRCLK and SCLK)[a]	Hex Message
0 (default)	DAO_MCLK - Slave DAO1_LRCLK - Slave DAO1_SCLK - Slave DAO2_LRCLK - Slave DAO2_SCLK - Slave	0x8140002C 0x00002000 0x8100002D 0x00002000
1	DAO_MCLK - Slave DAO1_LRCLK - Master DAO1_SCLK - Master DAO2_LRCLK - Master DAO2_SCLK - Master	0x8180002C 0xFFFFDFFF 0x8180002D 0xFFFFDFFF
2	DAO_MCLK - Slave DAO1_LRCLK - Slave DAO1_SCLK - Slave DAO2_LRCLK - Master DAO2_SCLK - Master	0x8140002C 0x00002000 0x8180002D 0xFFFFDFFF

a. A0 always goes with parameter B0. B0 is located in Table 7-3..

Please refer to the Table 7-3, Table 7-4, Table 7-5, Table 7-6, and Table 7-7 for a visual example of the clocking directions for the settings in Table 7-2.

Table 7-3 shows values and messages for the data format configuration parameters.

Table 7-3. DAO1 & DAO2 Clocking Relationship Configuration (Parameter B)

B Value	DAO1 & DAO2 Clocking Relationship	Hex Message
0	DAO2 dependent on DAO1 clocks **Note::** DA02_LRCLK & DA02_SCLK are driven by DA01_LRCLK & DA01_SCLK	0x8140002B 0x00002000
1 (default)	DAO2 independent of DAO1 clocks	0x8180002B 0xFFFFDFFF

Table 7-4 shows values and messages for the output DAO_SCLK/LRCLK frequency configuration parameter.

Table 7-4. Output DAO_SCLK/LRCLK Configuration (Parameter C)

C Value	DAO_SCLK Frequency	Hex Message
0 (default)	DAO_MCLK = 256 FS DAO1_SCLK = DAO_MCLK / 4 = 64 FS DAO1_LRCLK = DAO1_SCLK / 64 = FS DAO2_SCLK = DAO_MCLK / 4 = 64 FS DAO2_LRCLK = DAO2_SCLK / 64 = FS	0x8100003D 0x00007711 0x8100003E 0x00007711 0x8180002C 0xFFFFFF8F 0x8140002C 0x00000020
1	DAO_MCLK = 256 FS DAO1_SCLK = DAO_MCLK / 2 = 128 FS DAO1_LRCLK = DAO1_SCLK / 128 = FS DAO2_SCLK = DAO_MCLK / 2= 128 FS DAO2_LRCLK = DAO2_SCLK / 128 = FS	0x8100003D 0x00017701 0x8100003E 0x00017701 0x8180002C 0xFFFFFF8F 0x8140002C 0x00000040
2	DAO_MCLK = 256 FS DAO1_SCLK = DAO_MCLK / 1 = 256 FS DAO1_LRCLK = DAO1_SCLK / 256 = FS DAO2_SCLK = DAO_MCLK / 1= 256 FS DAO2_LRCLK = DAO2_SCLK / 256 = FS	0x8100003D 0x00037700 0x8100003E 0x00037700 0x8180002C 0xFFFFFF8F 0x8140002C 0x00000060
3	DAO_MCLK = 128 FS DAO1_SCLK = DAO_MCLK / 2 = 64 FS DAO1_LRCLK = DAO1_SCLK / 64 = FS DAO2_SCLK = DAO_MCLK / 2= 64 FS DAO2_LRCLK = DAO2_SCLK / 64 = FS	0x8100003D 0x00007701 0x8100003E 0x00007701 0x8180002C 0xFFFFFF8F 0x8140002C 0x00000020
4	DAO_MCLK = 128 FS DAO1_SCLK = DAO_MCLK / 1 = 128 FS DAO1_LRCLK = DAO1_SCLK / 128 = FS DAO2_SCLK = DAO_MCLK / 1 = 128 FS DAO2_LRCLK = DAO2_SCLK / 128 = FS	0x8100003D 0x00017700 0x8100003E 0x00017700 0x8180002C 0xFFFFFF8F 0x8140002C 0x00000040
5	DAO_MCLK = 512 FS DAO1_SCLK = DAO_MCLK / 8 = 64 FS DAO1_LRCLK = DAO1_SCLK / 64 = FS DAO2_SCLK = DAO_MCLK / 8= 64 FS DAO2_LRCLK = DAO2_SCLK / 64 = FS	0x8100003D 0x00007713 0x8100003E 0x00007713 0x8180002C 0xFFFFFF8F 0x8140002C 0x00000020

Table 7-4. Output DAO_SCLK/LRCLK Configuration (Parameter C) (Continued)

C Value	DAO_SCLK Frequency	Hex Message
6	DAO_MCLK = 512 FS DAO1_SCLK = DAO_MCLK / 4 = 128 FS DAO1_LRCLK = DAO1_SCLK / 128 = FS DAO2_SCLK = DAO_MCLK / 4 = 128 FS DAO2_LRCLK = DAO2_SCLK / 128 = FS	0x8100003D 0x00017711 0x8100003E 0x00017711 0x8180002C 0xFFFFFF8F 0x8140002C 0x00000040
7	DAO_MCLK = 512 FS DAO1_SCLK = DAO_MCLK / 2 = 256 FS DAO1_LRCLK = DAO1_SCLK / 256 = FS DAO2_SCLK = DAO_MCLK / 2 = 256 FS DAO2_LRCLK = DAO2_SCLK / 256 = FS	0x8100003D 0x00037701 0x8100003E 0x00037701 0x8180002C 0xFFFFFF8F 0x8140002C 0x00000060
8	DAO_MCLK = 384 FS DAO1_SCLK = DAO_MCLK / 4 = 96 FS DAO1_LRCLK = DAO1_SCLK / 96 = FS DAO2_SCLK = DAO_MCLK / 4 = 96 FS DAO2_LRCLK = DAO2_SCLK / 96 = FS	0x8100003D 0x00037211 0x8100003E 0x00037211 0x8180002C 0xFFFFFF8F 0x8140002C 0x00000030
9	DAO_MCLK = 384 FS DAO1_SCLK = DAO_MCLK / 2 = 192 FS DAO1_LRCLK = DAO1_SCLK / 192 = FS DAO2_SCLK = DAO_MCLK / 2 = 192 FS DAO2_LRCLK = DAO2_SCLK / 192 = FS	0x8100003D 0x00077201 0x8100003E 0x00077201 0x8180002C 0xFFFFFF8F 0x8140002C 0x00000050
10	DAO_MCLK = 1024 Fs DAO1_SCLK = DAO_MCLK/16 = 64 Fs DAO1_LRCLK = DAOn_SCLK/64 = Fs DAO2_SCLK = DAO_MCLK/16 = 64 Fs DAO2_LRCLK = DAOn_SCLK/64 = Fs	0x8100003D 0x00007717 0x8100003E 0x00007717 0x8180002C 0xFFFFFF8F 0x8140002C 0x00000020
11	DAO_MCLK = 64 Fs DAO1_SCLK = DAO_MCLK/1 = 64 Fs DAO1_LRCLK = DAOn_SCLK/64 = Fs DAO2_SCLK = DAO_MCLK/1 = 64 Fs DAO2_LRCLK = DAOn_SCLK/64 = Fs	0x8100003D 0x00007700 0x8100003E 0x00007700 0x8180002C 0xFFFFFF8F 0x8140002C 0x00000020

Table 7-4. Output DAO_SCLK/LRCLK Configuration (Parameter C) (Continued)

C Value	DAO_SCLK Frequency	Hex Message
12	DAO_MCLK = 384 Fs DAO1_SCLK = DAO_MCLK/6 = 64 Fs DAO1_LRCLK = DAOn_SCLK/64 = Fs DAO2_SCLK = DAO_MCLK/6 = 64 Fs DAO2_LRCLK = DAOn_SCLK/64 = Fs	0x8100003D 0x00007712 0x8100003E 0x00007712 0x8180002C 0xFFFFFF8F 0x8140002C 0x00000020
13	DAO_MCLK = 768 Fs DAO1_SCLK = DAO_MCLK/12 = 64 Fs DAO1_LRCLK = DAOn_SCLK/64 = Fs DAO2_SCLK = DAO_MCLK/12 = 64 Fs DAO2_LRCLK = DAOn_SCLK/64 = Fs	0x8100003D 0x00007732 0x8100003E 0x00007732 0x8180002C 0xFFFFFF8F 0x8140002C 0x00000020
14	DAO_MCLK = 1152 Fs DAO1_SCLK = DAO_MCLK/18 = 64 Fs DAO1_LRCLK = DAOn_SCLK/64 = Fs DAO2_SCLK = DAO_MCLK/18 = 64 Fs DAO2_LRCLK = DAOn_SCLK/64 = Fs	0x8100003D 0x00007752 0x8100003E 0x00007752 0x8180002C 0xFFFFFF8F 0x8140002C 0x00000020
15	DAO_MCLK = 192 Fs DAO1_SCLK = DAO_MCLK/3 = 64 Fs DAO1_LRCLK = DAOn_SCLK/64 = Fs DAO2_SCLK = DAO_MCLK/3 = 64 Fs DAO2_LRCLK = DAOn_SCLK/64 = Fs	0x8100003D 0x00007702 0x8100003E 0x00007702 0x8180002C 0xFFFFFF8F 0x8140002C 0x00000020
16	[DAO_MCLK = 576 Fs DAO1_SCLK = DAO_MCLK/9 = 64 Fs DAO1_LRCLK = DAOn_SCLK/64 = Fs DAO2_SCLK = DAO_MCLK/9 = 64 Fs DAO2_LRCLK = DAOn_SCLK/64 = Fs	0x8100003D 0x00007722 0x8100003E 0x00007722 0x8180002C 0xFFFFFF8F 0x8140002C 0x00000020

Table 7-5 shows values and messages for the data format configuration parameters.

Table 7-5. Output Data Format Configuration (Parameter D)

D Value	DAO Data Format Of DAO_DATA0, 1, 2 (or DAO_DATA0 for Multichannel Modes)	Hex Message
0 (default)	I^2S 32-bit	0x81000030 0x00000001 0x81000031 0x00000001 0x81000032 0x00000001 0x81000033 0x00000001 0x81000034 0x00000001 0x81000035 0x00000001 0x81000036 0x00000001 0x81000037 0x00000001
1	Left Justified 32-bit	0x81000030 0x00000000 0x81000031 0x00000000 0x81000032 0x00000000 0x81000033 0x00000000 0x81000034 0x00000000 0x81000035 0x00000000 0x81000036 0x00000000 0x81000037 0x00000000 0x8180002C 0xFFFFFBFF 0x8180002D 0xFFFFFBFF

Table 7-5. Output Data Format Configuration (Parameter D) (Continued)

D Value	DAO Data Format Of DAO_DATA0, 1, 2 (or DAO_DATA0 for Multichannel Modes)	Hex Message
2	TDM on DAO1_D0	0x81000030 0x00011701 0x81000031 0x00000001 0x81000032 0x00000001 0x81000033 0x00000001 0x81000034 0x00000001 0x81000035 0x00000001 0x81000036 0x00000001 0x81000037 0x00000001

Table 7-6 shows values and messages for the DAO_LRCLK polarity configuration parameter.

Table 7-6. Output DAO_LRCLK Polarity Configuration (Parameter E)

E Value	DAO_LRCLK Polarity	Hex Message
0 (default)	LRCLK=Low indicates Left Subchannel	0x8140002C 0x00000700 0x8140002D 0x00000700
1	LRCLK=Low indicates Right Subchannel	0x8180002C 0xFFFFFBFF 0x8140002C 0x00000300 0x8180002D 0xFFFFFBFF 0x8180002D 0x00000300

Table 7-7 shows values and messages for the DAO_SCLK polarity configuration parameter.

Table 7-7. Output DAO_SCLK Polarity Configuration (Parameter F)

F Value	DAO_SCLK Polarity	Hex Message
0 (default)	Data Valid on Rising Edge (clocked out on falling)	0x8180002C 0xFFFFEFFF 0x8180002D 0xFFFFEFFF
1	Data Valid on Falling Edge (clocked out on rising)	0x8140002C 0x00001000 0x8140002D 0x00001000

Table 7-8 shows values and messages for the output channel configuration parameter.

Table 7-8. Output Channel Configuration (Parameter G)

G Value	Channel Configuration	Hex Message
0 (default)	2 Channels	0x8180002C 0xFFFFF8FF 0x8140002C 0x00000700
1	6 Channels	0x8180002C 0xFFFFF8FF 0x8140002C 0x00000400

7.1.5 S/PDIF Transmitter

Two S/PDIF transmitters are provided on the XMTA and XMTB pins that can output an IEC60958-compliant S/PDIF stream. The modulation clock source for the S/PDIF transmitter is the clock present on the DAO_MCLK pin. The sample rate of the transmitter will be the same setting as for the respective DAO port.

The DSP has an internal multiplexer that can be used to route an external S/PDIF signal from the XMTA_IN or XMTB_IN pin directly to the respective XMTA/XMITB S/PDIF output pin instead of the internally generated S/PDIF signal.

To summarize the XMTA/XMITB S/PDIF output pins can be configured as:

- I^2S output - Default
- S/PDIF Transmitter - Sent config from Table 7-10
- DSP Bypass - Send config from Table 7-11

The DSP Bypass config is typically used to route a S/PDIF stream from input to the output for recording applications or as a PCM bypass for dual-zone applications. A soft reset is required when switching between any of the above modes.

Table 7-9. S/PDIF Transmitter Pins

Pin Name	Pin Description	LQFP-144 Pin #	LQFP-128 Pin #	Pin Type
XMTA_IN	S/PDIF Input for XMTA mux The XMTA_IN S/PDIF inputs is muxed with XMTA to allow switching the S/PDIF output between the internal and external sources	2	-	Input
XMTB_IN	S/PDIF Input for XMTB mux The XMTB_IN S/PDIF inputs is muxed with XMTB to allow switching the S/PDIF output between the internal and external sources	92	-	Input
DAO1_DATA3/**XMTA**	S/PDIF Audio Output A	15	47	Output
DAO2_DATA3/**XMTB**	S/PDIF Audio Output B	5	35	Output

Table 7-10. S/PDIF Transmitter Configuration

Description	Hex Message
Enable SPDIF on Output A on DAO1_DATA3/XMTA (MCLK=256 Fs)	0x8100002e 0x00005080
Enable SPDIF on Output A on DAO1_DATA3/XMTA (MCLK=512 Fs)	0x8100002e 0x00005080 0x8180003C 0xFFFFFF8F 0x8140003C 0x00000010
Enable SPDIF on Output A on DAO1_DATA3/XMTA (MCLK=1024 Fs)	0x8100002e 0x00005080 0x8180003C 0xFFFFFF8F 0x8140003C 0x00000030
Enable SPDIF on Output B on DAO2_DATA3/XMTB (MCLK=256 Fs)	0x8100002f 0x00005080
Enable SPDIF on Output B on DAO2_DATA3/XMTB (MCLK=512 Fs)	0x8100002f 0x00005080 0x8180003C 0xFFFFFF8F 0x8140003C 0x00000010
Enable SPDIF on Output B on DAO2_DATA3/XMTB (MCLK=1024 Fs)	0x8100002f 0x00005080 0x8180003C 0xFFFFFF8F 0x8140003C 0x00000030

Table 7-11. DSP Bypass Configuration

DAO_SCLK Polarity	Hex Message
Route XMTA_IN to DAO1_DATA3/XMTA and XMTB_IN to DAO2_DATA3/XMTB	0x81000052 0x0000001B
Route XMTA_IN to DAO1_DATA3/XMTA	0x81000052 0x00000013
Route XMTB_IN to DAO2_DATA3/XMTB	0x81000052 0x0000000b
Disable DSP Bypass	0x81000052 0x00000003

§§

Chapter 8

External Memory Interfaces

8.1 SDRAM Controller

The CS4953xx supports a glueless external SDRAM interface to extend the data and/or program memory of the DSP during runtime. The CS4953xx SDRAM controller provides two-port access to X, Y, and P memory space, a four-word read buffer, and a double-buffered four-word write buffer. One SDRAM controller port is dedicated to P memory space and the second port is shared by X and Y memories. An arbiter is wrapped around the first port to provide external SDRAM access through both X and Y memory space from 8000H-FFFFH. The second port is connected to provide external SDRAM access through P memory space from 8000H-FFFFH.

The X/Y port has dual write buffers and a single read buffer. The P memory port has a single read buffer. One of these buffers contains four 32-bit words (128 bits). Every miss to the read buffer will cause the SDRAM controller to burst eight 16-bit reads on the SDRAM interface. Figure 8-1 shows a block diagram of the SDRAM external memory control interface for the CS4953xx chip.

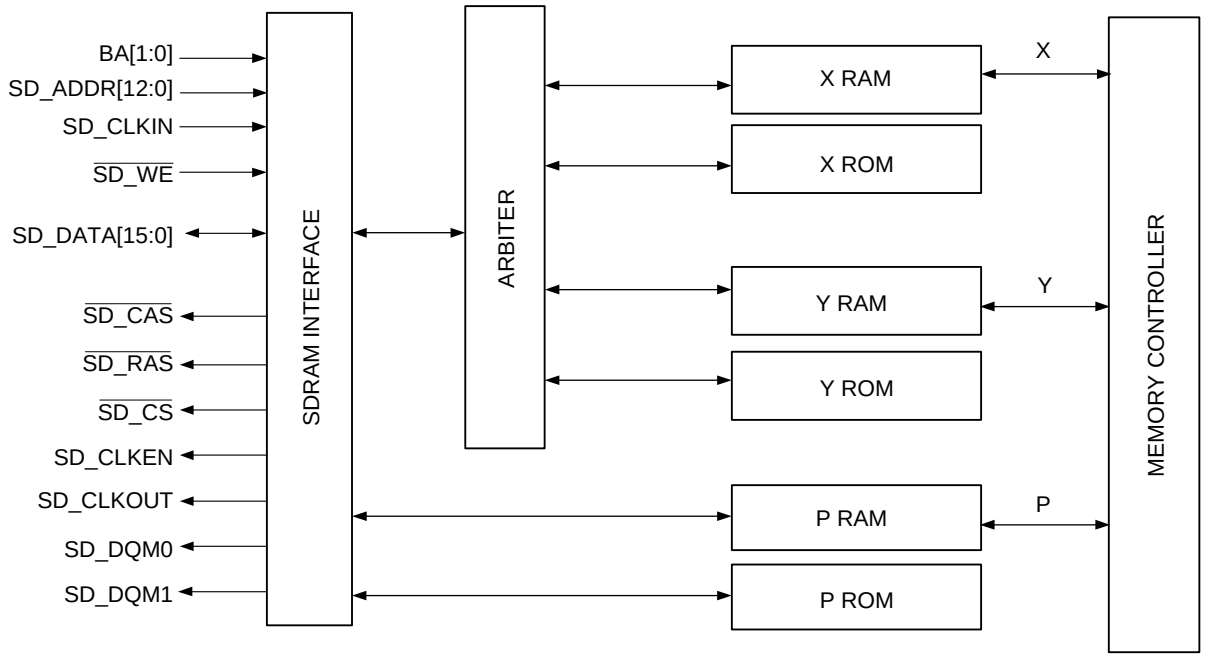


Figure 8-1. SDRAM Interface Block Diagram

8.2 Flash Memory Controller

The CS4953xx provides a glueless external Flash interface that supports connection to an external Flash or EEPROM for code storage. This allows for products to be field-upgraded as new audio algorithms are developed. The Flash controller allows autobooting to occur from a parallel Flash or EEPROM device. Coefficients for filters may also be stored and recalled from parallel Flash or EEPROM.

8.2.1 Flash Controller Interface

The Flash controller allows the CS4953xx DSP to autoboot from a parallel Flash or EEPROM device. Both Flash and EEPROM can be accessed using 8-bit, 16-bit, and 32-bit data modes (1-byte, 2-byte, and 4-byte words) and using an 8-bit or 16-bit data bus, where the word width is the number of bytes per transfer, and the data bus size is the width of the physical interface to Flash. For example, an 8-bit data bus setting uses only 8 data pins EXT_D[7:0] whereas the 16-bit data bus setting uses 16 data pins (EXT_D[15:0]).

8.3 SDRAM/Flash Controller Interface

The physical interface of the SDRAM controller consists of 16 data pins (SD_DATA[15:0]), 13 address pins (SD_ADDR[12:0]), 2 bank address pins (SD_BA[1:0]), and 9 control pins (SD_CS, SD_WE, SD_DQM1, SD_DQM0, SD_CAS, SD_RAS, SD_CLKOUT, SD_CLKIN, SD_CLKEN). SD_CS is the SDRAM chip select pin. The address and data pins are shared with the Flash interface. The CS4953xx supports SDRAMs from 2 Mbytes to 64 Mbytes with various row, bank, and column configurations. The size can be configured in the DynamicConfig0 register listed in Table 8-2. Timing parameters of the SDRAM port can be configured to meet various SDRAM requirements as described in Table 8-2. The default timing parameters have been chosen and tested to meet the requirements of Hynix HY57V641620HG-H. By default, the SDRAM port is configured for 64 Mbits with 4 banks, 12 rows, and 8 columns with a RAS and CAS latency of 3.

The physical interface of the Flash controller consists of 16 data pins (EXT_D[15:0]), 20 address pins (EXT_A[19:0]), and 4 control pins (EXT_CS1, EXT_CS2, EXT_OE, EXT_WE). The address and data pins are shared with the SDRAM interface. EXT_CS1 is the Flash or EEPROM chip select pin. EXT_CS2 is not supported. The Flash interface supports up to 1M x 8 addressable space (512k x 16 bits of Flash). Data in Flash is stored big-endian, i.e. more significant bytes in a multi-byte word have lower addresses.

Note: When connected to a 16 Mbit SDRAM, the CS4953xx uses only SD_BA1 for bank selection.

8.3.1 SDRAM/Flash Interface Signals

Table 8-1 shows the signal names, descriptions, and pin number of the signals associated with the external SDRAM/Flash memory control port on the CS4953xx chip.

Table 8-1. SDRAM Interface Signals

Signal Name	Signal Description	LQFP-144 Pin #	LQFP-128 Pin #	Pin Type
SD_CLKOUT	SDRAM clock output. This output is tri-stated when SDRAM interface is not used.	51	80	Output
SD_CLKIN	SDRAM Clock input Connects to trace from SDRAM device CLKIN pin.	52	81	Input
SD_CLKEN	SDRAM Clock Enable Output	53	82	Output

Table 8-1. SDRAM Interface Signals (Continued)

Signal Name	Signal Description	LQFP-144 Pin #	LQFP-128 Pin #	Pin Type
$\overline{\text{SD\_RAS}}$	SDRAM Row Address Strobe	80	109	Output
$\overline{\text{SD\_CAS}}$	SDRAM Column Address Strobe	79	108	Output
$\overline{\text{SD\_CS}}$	SDRAM Chip Select	81	110	Output
SD_DQM0	SDRAM Data Mask 0	28	57	Output
SD_DQM1	SDRAM Data Mask 1	50	79	Output
$\overline{\text{SD\_WE}}$	SDRAM Write Enable	78	107	Output
SD_A0/EXT_A0	SDRAM/Flash Address 0	72	102	Output
SD_A1/EXT_A1	SDRAM/Flash Address 1	71	101	Output
SD_A2/EXT_A2	SDRAM/Flash Address 2	70	99	Output
SD_A3/EXT_A3	SDRAM/Flash Address 3	68	97	Output
SD_A4/EXT_A4	SDRAM/Flash Address 4	67	96	Output
SD_A5/EXT_A5	SDRAM/Flash Address 5	64	93	Output
SD_A6//EXT_A6	SDRAM/Flash Address 6	62	91	Output
SD_A7/EXT_A7	SDRAM/Flash Address 7	61	90	Output
SD_A8/EXT_A8	SDRAM/Flash Address 8	59	88	Output
SD_A9/EXT_A9	SDRAM/Flash Address 9	58	87	Output
SD_A10/EXT_A10	SDRAM/Flash Address 10	74	103	Output
SD_A11/EXT_A11	SDRAM/Flash Address 11	56	85	Output
SD_A12/EXT_A12	SDRAM/Flash Address 12	55	84	Output
SD_BA0/EXT_A13[a]	SDRAM Bank Address 0/Flash Address 13	75	104	Input
SD_BA1/EXT_A14[1]	SDRAM Bank Address 1/Flash Address 14	77	106	Input
EXT_A15	Flash Address 15	82	111	Output
EXT_A16	Flash Address 16	84	113	Output
EXT_A17	Flash Address 17	85	114	Output
EXT_A18	Flash Address 18	87	116	Output
EXT_A19	Flash Address 19	88	117	Output
SD_D0/EXT_D0	SDRAM/Flash Bidirectional Data Bit 0	39	68	BiDir
SD_D1/EXT_D1	SDRAM/Flash Bidirectional Data Bit 1	37	65	BiDir
SD_D2/EXT_D2	SDRAM/Flash Bidirectional Data Bit 2	35	64	BiDir

Table 8-1. SDRAM Interface Signals (Continued)

Signal Name	Signal Description	LQFP-144 Pin #	LQFP-128 Pin #	Pin Type
SD_D3/EXT_D3	SDRAM/Flash Bidirectional Data Bit 3	34	63	BiDir
SD_D4/EXT_D4	SDRAM/Flash Bidirectional Data Bit 4	32	61	BiDir
SD_D5/EXT_D5	SDRAM/Flash Bidirectional Data Bit 5	31	60	BiDir
SD_D6/EXT_D6	SDRAM/Flash Bidirectional Data Bit 6	30	59	BiDir
SD_D7/EXT_D7	SDRAM/Flash Bidirectional Data Bit 7	29	58	BiDir
SD_D8/EXT_D8	SDRAM/Flash Bidirectional Data Bit 8	49	78	BiDir
SD_D9/EXT_D9	SDRAM/Flash Bidirectional Data Bit 9	48	77	BiDir
SD_D10/EXT_D10	SDRAM/Flash Bidirectional Data Bit 10	46	75	BiDir
SD_D11/EXT_D11	SDRAM/Flash Bidirectional Data Bit 11	45	74	BiDir
SD_D12/EXT_D12	SDRAM/Flash Bidirectional Data Bit 12	43	72	BiDir
SD_D13/EXT_D13	SDRAM/Flash Bidirectional Data Bit 13	42	71	BiDir
SD_D14/EXT_D14	SDRAM/Flash Bidirectional Data Bit 14	41	70	BiDir
SD_D15/EXT_D15	SDRAM/Flash Bidirectional Data Bit 15	40	69	BiDir
$\overline{\text{EXT\_CS1}}$	Flash Chip Select (active low)	90	119	Output
$\overline{\text{EXT\_OE}}$	Flash Output Enable (active low)	89	118	Output
$\overline{\text{EXT\_WE}}$	Flash Write Enable (active low)	38	66	Output
$\overline{\text{EXT\_CS2}}$	SRAM Chip Select (active low) (CURRENTLY NOT SUPPORTED)	65	94	Output

a. For 16-Mbit parts, SD_BA0 is used as a bank select. For 64-Mbit and 128-Mbit parts SD_BA[1:0] is the 2-bit bank select.

8.3.2 Configuring SDRAM/Flash Parameters

IMPORTANT NOTICE: External memory is enabled by default. On systems that do not use external memory, the command, 0x8100005c 0x00000000 must be sent to the DSP before kickstart to disable external memory.

Not all Flash and EEPROM manufacturers conform to the same timing specifications. Therefore, the CS4953xx DSP must be configured to match the timing specifications for the Flash and EEPROM being used. The messages shown in Table 8-2 must be sent prior to kickstarting downloaded application code.

Note: All External Memory Interface config messages must be sent to DSPB. For more details on sending messages to DSP B refer to AN288, section 2.1.4

Refer to External Memory Interface in the CS4953xx data sheet for timing parameters that are summarized in Table 8-2.

Table 8-2. SDRAM/Flash Controller Parameters

Mnemonic	Hex Message
Extmem_Setup_Control Bit 31:5 = 0 = Reserved Bit 4 = 0/1 = Disable/Enable Flash port Bit 3:1 = 0 = Reserved Bit 0 = 0/1 = Disable/Enable SDRAM port	0x8100005C 0xHHHHHHHH Default: 0x00000011
CRUSConfig Bit 31:9 = 0 = Reserved Bit 8 = Pin Mapping, where: 0 = Enable Flash and SDRAM Mapping 1 = Enable Flash and SRAM Mapping Bit 7:0 = 0 = Reserved	0x8100005D 0xHHHHHHHH Default: 0x00000000
DynamicRefresh Configure the refresh period Bit 31:11 = 0 = Reserved Bit 10:0 = REFRESH, where: 0x0 = refresh disabled. 0x1 = 1x16 = 16 HCLK ticks between refresh cycles 0x8 = 8x16 = 128 HCLK ticks between refresh cycles 0x1 to 0x7FF = REFRESH*16 HCLK ticks between refresh cycles Example: Refresh Period = 15.625µS, HCLK = 120Mhz REFRESH = (15.625µS*120Mhz)/16 = 117 = 0x75	0x81000061 0xHHHHHHHH Default 0x00000075
DynamictRP Configure the precharge command period Bit 31:4 = 0 = Reserved Bit 3:0 = Trp, where: 0x0 to 0xE = (n + 1) DSP clk cycles. 0xF = 16 DSP clk cycles. Example: Trp = 20nS, HCLK = 120Mhz Trp =20nS*120Mhz - 1 = 1.4 = 0x2	0x81000062 0xHHHHHHHH Default 0x00000002
DynamictRAS Configure the active to precharge command period Bit 31:4 = 0 = Reserved Bit 3:0 = Tras, where: 0x0 to 0xE = (n + 1) DSP clk cycles. 0xF = 16 DSP clk cycles. Example: Tras = 45nS, HCLK = 120Mhz Tras =45nS*120Mhz - 1 = 3.8 = 0x4	0x81000063 0xHHHHHHHH Default 0x00000004

Table 8-2. SDRAM/Flash Controller Parameters (Continued)

Mnemonic	Hex Message
DynamictREX Configure the self refres exit time. Also known as Tsre Bit 31:4 = 0 = Reserved Bit 3:0 = Trex, where: 0x0 to 0xE = (n + 1) DSP clk cycles. 0xF = 16 DSP clk cycles. Example: Trex = 83 nS, HCLK = 120Mhz Trex = 83 nS * 120 Mhz - 1 =10-1 = 0x9	0x81000064 0xHHHHHHHH Default 0x00000009
DynamictAPR Configure the last data out to active command time. Bit 31:4 = 0 = Reserved Bit 3:0 = Tapr, where: 0x0 to 0xE = (n + 1) DSP clk cycles. 0xF = 16 DSP clk cycles.	0x81000065 0xHHHHHHHH Default 0x00000000
DynamictDAL Configure the data-in to active command time. Bit 31:4 = 0 = Reserved Bit 3:0 = Tdal, where: 0x0 to 0xE = (n + 1) DSP clk cycles. 0xF = 16 DSP clk cycles. Example: Tdal = 6 CLKs, HCLK = 120Mhz Tdal = 6-1 = 0x5	0x81000066 0xhHHHHHHH Default 0x00000005
DynamictWR Configure the write recovery time. Also known as Tdpl, Trwl Bit 31:4 = 0 = Reserved Bit 3:0 = Tdal, where: 0x0 to 0xE = (n + 1) DSP clk cycles. 0xF= 16 DSP clk cycles. Example: Twr = 2 CLKs, HCLK = 120Mhz Twr = 2-1 = 0x1	0x81000067 0xHHHHHHHH Default 0x00000001
DynamictRC Configure the active to active command time. Bit 31:5 = 0 = Reserved Bit 4:0 = Trc, where: 0x0 to 0x1E = (n + 1) DSP clk cycles. 0x1F = 16 DSP clk cycles. Example: Trc = 65 nS, HCLK = 120Mhz Trc = 65 nS * 120 Mhz -1 =7.8-1 = 0x7	0x81000068 0xHHHHHHHH Default 0x00000007

Table 8-2. SDRAM/Flash Controller Parameters (Continued)

Mnemonic	Hex Message
DynamictRFC Configure the auto refresh period and auto refresh to active command time. Also known as Trrc Bit 31:5 = 0 = Reserved Bit 4:0 = Trc, where: 0x0 to 0xE = (n + 1) DSP clk cycles. 0x1F = 32 DSP clk cycles. Example: Trc = 65 nS, HCLK = 120Mhz Trc = 65 nS * 120 Mhz -1=7.8-1 = 0x7	0x81000069 0xHHHHHHHH Default 0x00000007
DynamictXSR Configure the exit self refresh to active command time. Bit 31:5 = 0 = Reserved Bit 4:0 = Txsr, where: 0x0 to 0x1E = (n + 1) DSP clk cycles. 0x1F = 32DSP clk cycles. Example: Txsr = 83 nS, HCLK = 120Mhz Txsr = 83 nS * 120 Mhz-1 =10-1 = 0x9	0x8100006A 0xHHHHHHHH Default 0x00000009
DynamictRRD Configure the active bank A to active bank B latency Bit 31:4 = 0 = Reserved Bit 3:0 = Trrd, where: 0x0 to 0xE = (n + 1) DSP clk cycles. 0xF= 16 DSP clk cycles. Example: Trrd = 15 nS, HCLK = 120Mhz Trrd = 15 nS * 120 Mhz - 1= 1.8 - 1 = 0x1	0x8100006B 0xHHHHHHHH Default 0x00000001
DynamictMRD Configure the load mode register to active command time. Also known as Trsa Bit 31:4 = 0 = Reserved Bit 3:0 = Tmrd, where: 0x0 to 0xE = (n + 1) DSP clk cycles. 0xF= 16 DSP clk cycles. Example: Tmrd = 1CLKs, HCLK = 120Mhz Tmrd = 1-1 = 0x0	0x8100006C 0xHHHHHHHH Default 0x00000000
DynamicConfig0 Configure the active bank A to active bank B latency Bit 31:12 = 0 = Reserved Bit 11:7 = External bus address mapping (Row, Bank, Column),where: 00001 = 16Mb (1Mx16), 2 banks, row length = 11, column length = 8 00101 = 64Mb (4Mx16), 4 banks, row length = 12, column length = 8 01001 = 128Mb (8Mx16), 4 banks, row length = 12, column length = 9 Bit 6:0 = 0 = Reserved	0x8100006E 0xHHHHHHHH Default 0x00000280

Table 8-2. SDRAM/Flash Controller Parameters (Continued)

Mnemonic	Hex Message
DynamicRasCas0 Configure the active bank A to active bank B latency Bit 31:10 = 0 = Reserved Bit 9:8 = CAS latency (CAS),where: 00 = reserved 01 = one clock cycle 10 = two clock cycles 11 = three clock cycles Bit 7:2 = 0 = Reserved, rCAS latency (CAS),where: Bit 1:0 = RAS latency (RAS),where: 00 = reserved 01 = one clock cycle 10 = two clock cycles 11 = three clock cycles	0x8100006F 0xHHHHHHHH Default 0x00000303
StaticConfig0 **(Not Supported)** Bit 31:2 = 0 = Reserved Bit 1:0 = Memory Bus Width, where: 00 = SRAM Memory bus 8 bits wide. 01 = SRAM Memory bus 16 bits wide. 10 = Reserved 11 = Reserved	0x81000070 0xHHHHHHHH Default 0x00000000
StaticWaitWen0 **(Not Supported)** EXT_CS falling to EXT_WE falling (t_{xmcswe}) Bit 31:4 = 0 = Reserved Bit 3:0 =SRAM_WEN_CYCLE, where: 0000 = one DSP clk cycle between the assertion of chip select and write enable. 0001 to 1111 = (n+1) cycle delay. t_{xmcswe} =(SRAM_WEN_CYCLE + 1)*HCLK	0x81000071 0xHHHHHHHH Default 0x00000000
StaticWaitOen0 **(Not Supported)** EXT_CS falling to EXT_OE falling (t_{xmcsoe}) Bit 31:4 = 0 = Reserved Bit 3:0 =SRAM_OEN_CYCLE, where: 0000 = no delay between the assertion of chip select and output enable. 0001 to 1111 = (n) cycle delay. t_{xmcsoe} =(SRAM_OEN_CYCLE)*HCLK	0x81000072 0xHHHHHHHH Default 0x00000000
StaticWaitRd0 **(Not Supported)** Single Word Read Cycle (t_{xmrdc}) Bit 31:5 = 0 = Reserved Bit 4:0 =SRAM_RD_CYCLE, where: 00000 to 11110 = (n+1) HCLK cycle for Read Cycle. t_{xmrdc} =(SRAM_RD_CYCLE + 1)*HCLK - 6.87ns	0x81000073 0xHHHHHHHH Default 0x0000001F
StaticWaitWr0 **(Not Supported)** EXT_CS falling to EXT_WE rising (t_{xmcswa}) Bit 31:5 = 0 = Reserved Bit 4:0 =SRAM_WR_CYCLE, where: 00000 to 11110 = (n+2) HCLK cycle for Write access time. t_{xmcswa} =(SRAM_WR_CYCLE + 2)*HCLK	0x81000075 0xHHHHHHHH Default 0x0000001F

Table 8-2. SDRAM/Flash Controller Parameters (Continued)

Mnemonic	Hex Message
StaticWaitTurn0 **(Not Supported)** Bus Turnaround Cycle Delay (t_{xmturn}) Bit 31:4 = 0 = Reserved Bit 3:0 =SRAM_TURN_CYCLE, where: 0000 to 1110 = (n+1) HCLK cycle for bus Turnaround time. t_{xmturn} =(SRAM_TURN_CYCLE + 1)*HCLK	0x81000076 0xHHHHHHHH Default 0x0000000F
StaticConfig1 Bit 31:2 = 0 = Reserved Bit 1:0 = Memory Bus Width, where: 00 = Flash Memory bus 8 bits wide. 01 = Flash Memory bus 16 bits wide. 10 = Reserved 11 = Reserved	0x81000077 0xhHHHHHHH Default 0x00000001
StaticWaitWen1 EXT_CS falling to EXT_WE falling (t_{xmcswe}) Bit 31:4 = 0 = Reserved Bit 3:0 = Flash_WEN_CYCLE, where: 0000 = one DSP clk cycle between the assertion of chip select and write enable. 0001 to 1111 = (n+1) cycle delay. t_{xmcswe} =(Flash_WEN_CYCLE + 1)*HCLK	0x81000078 0XHHHHHHHH Default 0x00000000
StaticWaitOen1 EXT_CS falling to EXT_OE falling (t_{xmcsoe}) Bit 31:4 = 0 = Reserved Bit 3:0 =Flash_OEN_CYCLE, where: 0000 = no delay between the assertion of chip select and output enable. 0001 to 1111 = (n) cycle delay. t_{xmcsoe} =(Flash_OEN_CYCLE)*HCLK	0x81000079 0xHHHHHHHH Default 0x00000000
StaticWaitRd1 Single Word Read Cycle (t_{xmrdc}) Bit 31:5 = 0 = Reserved Bit 4:0 =Flash_RD_CYCLE, where: 00000 to 11110 = (n+1) HCLK cycle for Read Cycle. t_{xmrdc} =(Flash_RD_CYCLE + 1)*HCLK - 6.87ns	0x8100007A 0xHHHHHHHH Default 0x0000000A
StaticWaitWr1 EXT_CS falling to EXT_WE rising (t_{xmcswa}) Bit 31:5 = 0 = Reserved Bit 4:0 =Flash_WR_CYCLE, where: 00000 to 11110 = (n+2) HCLK cycle for Write access time. t_{xmcswa} =(Flash_WR_CYCLE + 2)*HCLK	0x8100007C 0xHHHHHHHH Default 0x00000007
StaticWaitTurn1 Bus Turnaround Cycle Delay (t_{xmturn}) Bit 31:4 = 0 = Reserved Bit 3:0 =Flash_TURN_CYCLE, where: 0000 to 1110 = (n+1) HCLK cycle for bus Turnaround time. t_{xmturn} =(Flash_TURN_CYCLE + 1)*HCLK	0x8100007D 0xHHHHHHHH Default 0x00000002

§§

Chapter 9

System Integration

9.1 Typical Connection Diagrams

Figure 9-1 is a typical connection diagram (LQFP-144) showing I^2C control with a serial FLASH, SDRAM and up to 7 DACs. Serial FLASH is recommended over parallel flash because it makes SDRAM layout much easier. This configuration uses the default settings for serial FLASH chip select (pin 6).

Figure 9-2 is a typical connection diagram (LQFP-144) showing SPI control with a serial FLASH, SDRAM and up to 7 DACs. This configuration uses the default settings for serial FLASH chip select (pin 6).

Figure 9-3 is a typical connection diagram (LQFP-144) showing SPI control with a serial FLASH, SDRAM and up to 8 DACs. This configuration allows the system to connect the maximum number of DACs to the DSP by using an alternate chip select for the serial FLASH (pin 121).

Figure 9-4 is a typical connection diagram (LQFP-144) showing I^2C control with parallel FLASH, SDRAM and up to 8 DACs. This configuration is not preferred because the SDRAM and FLASH share a bus, making routing more difficult.

Figure 9-5 is a typical connection diagram (LQFP-128) showing SPI control with parallel FLASH, SDRAM and up to 8 DACs. This configuration is not preferred because the SDRAM and FLASH share a bus, making routing more difficult.

Figure 9-6 is a typical connection diagram (LQFP-128) showing I^2C control with parallel FLASH, SDRAM and up to 8 DACs. This configuration is not preferred because the SDRAM and FLASH share a bus, making routing more difficult.

Figure 9-7 is a typical connection diagram (LQFP-144) showing SPI control with a serial FLASH, SDRAM and up to 7 DACs. This configuration uses the default settings for serial FLASH chip select (pin 6).

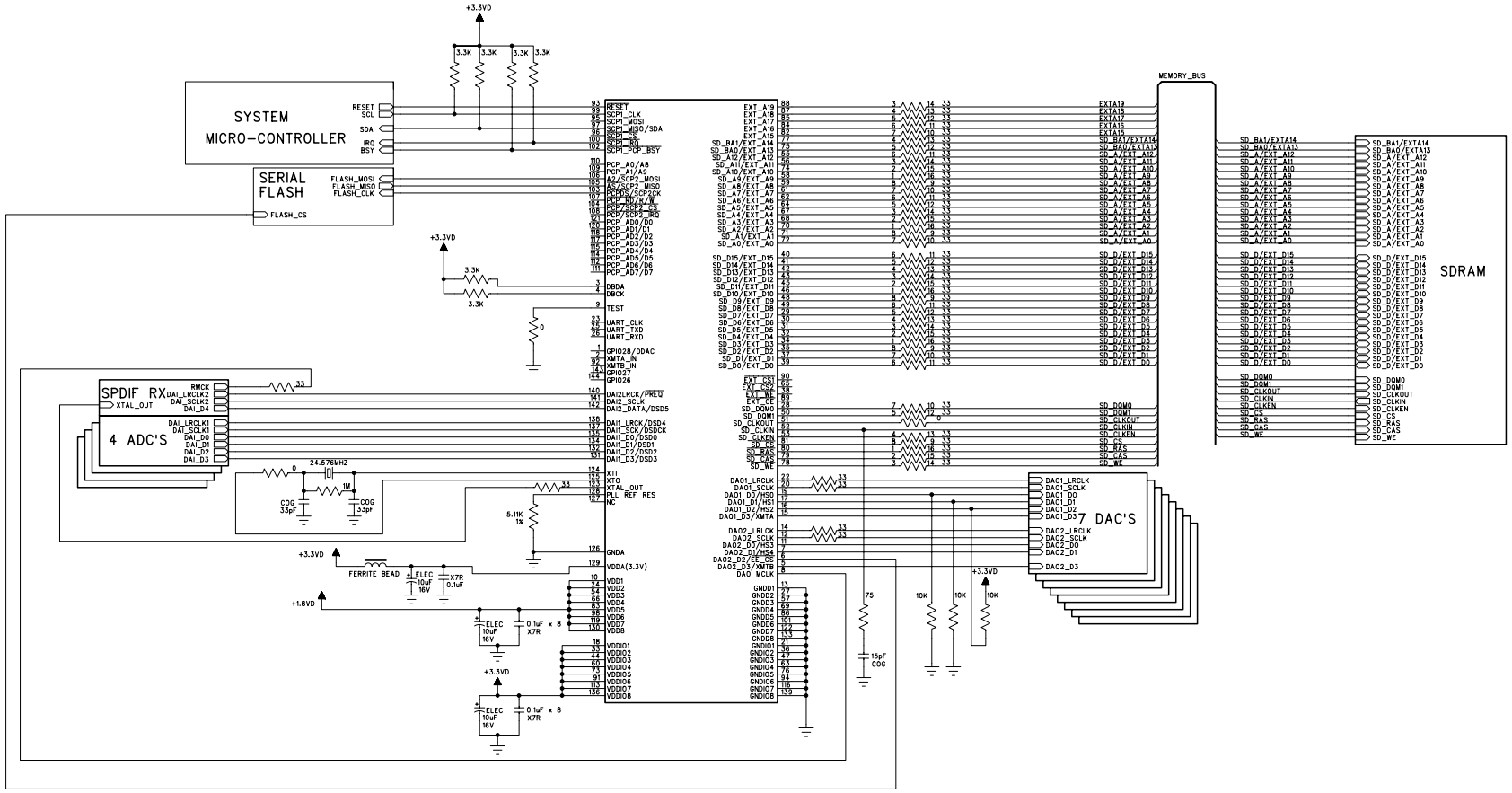
Figure 9-8 is a typical connection diagram (LQFP-144) showing SPI control with a serial FLASH, DSD Audio Input, SDRAM and up to 7 DACs.

Place PLL filter components as close as possible to the DSP. A unified, solid ground plane is recommended for optimal performance. Pay close attention to the direction of all clock signals shown in the diagram. These designs are configured to slave to clocks on the input side. On the output side, the DSP slaves to MCLK from the S/PDIF receiver and masters SCLK and LRCLK for the DACs. This is the recommended clocking for AVR applications.

Series Termination resistor values depend on the transmission line impedances of the actual PCB used. The design engineer should calculate the transmission line impedance of the traces and size the series R such that R = (Z – 15), where 15 represents the source impedance of the CS4953xx drivers.

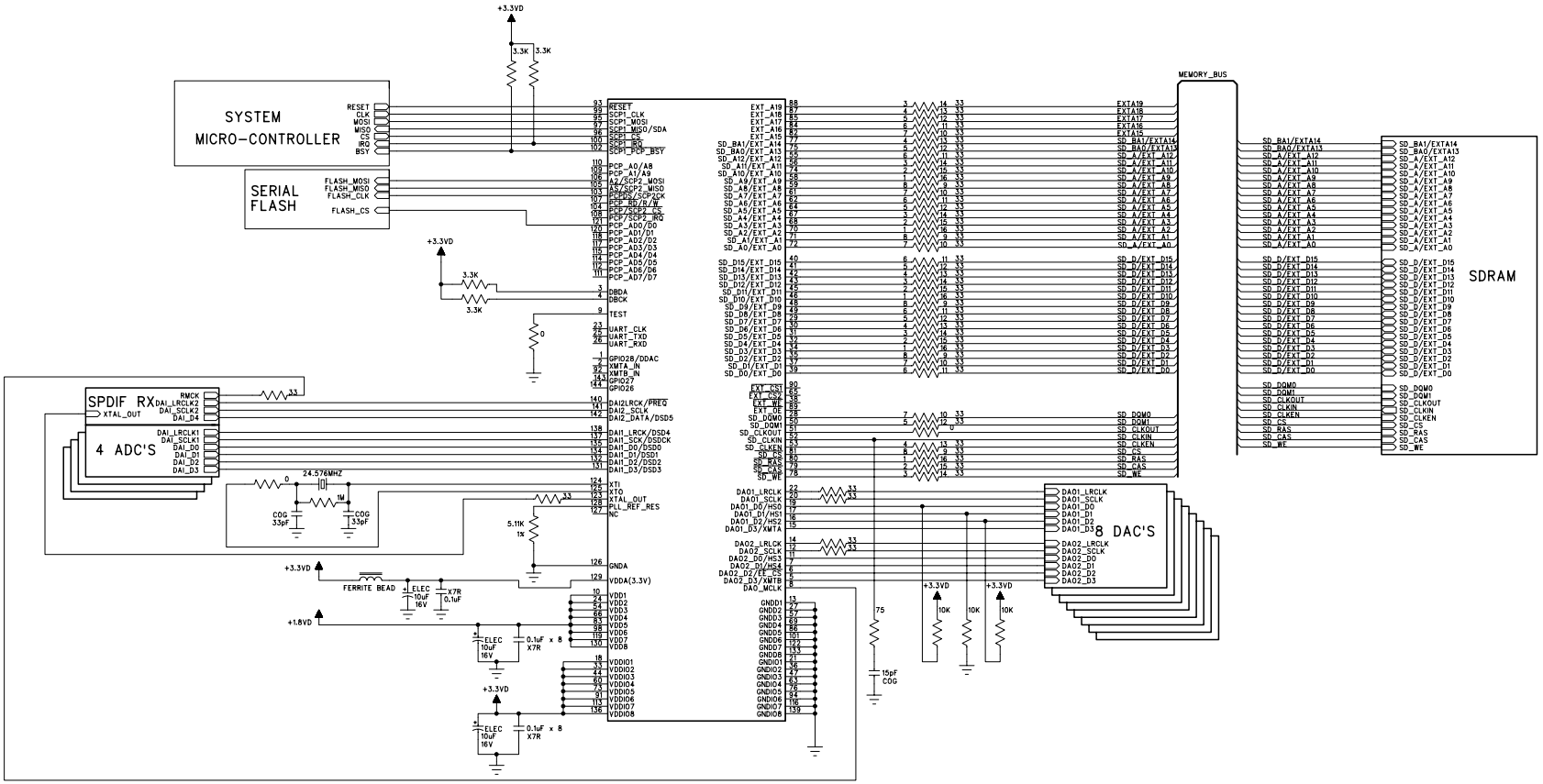
The typical connection diagrams show "0.1uF x 8" to indicate that 1 decoupling capacitor should be placed next to each power pin.

The SDCLK termination used in the typical connection diagrams, and on the CRD49530, is an AC parallel terminator. The proper board layout for this kind of termination is to route the SDRAM Clock signal to the SD_CLKIN pin and then beyond the pin a short distance before connecting to the parallel termination. The resistor and capacitor should be the last components on that trace (terminating the trace).



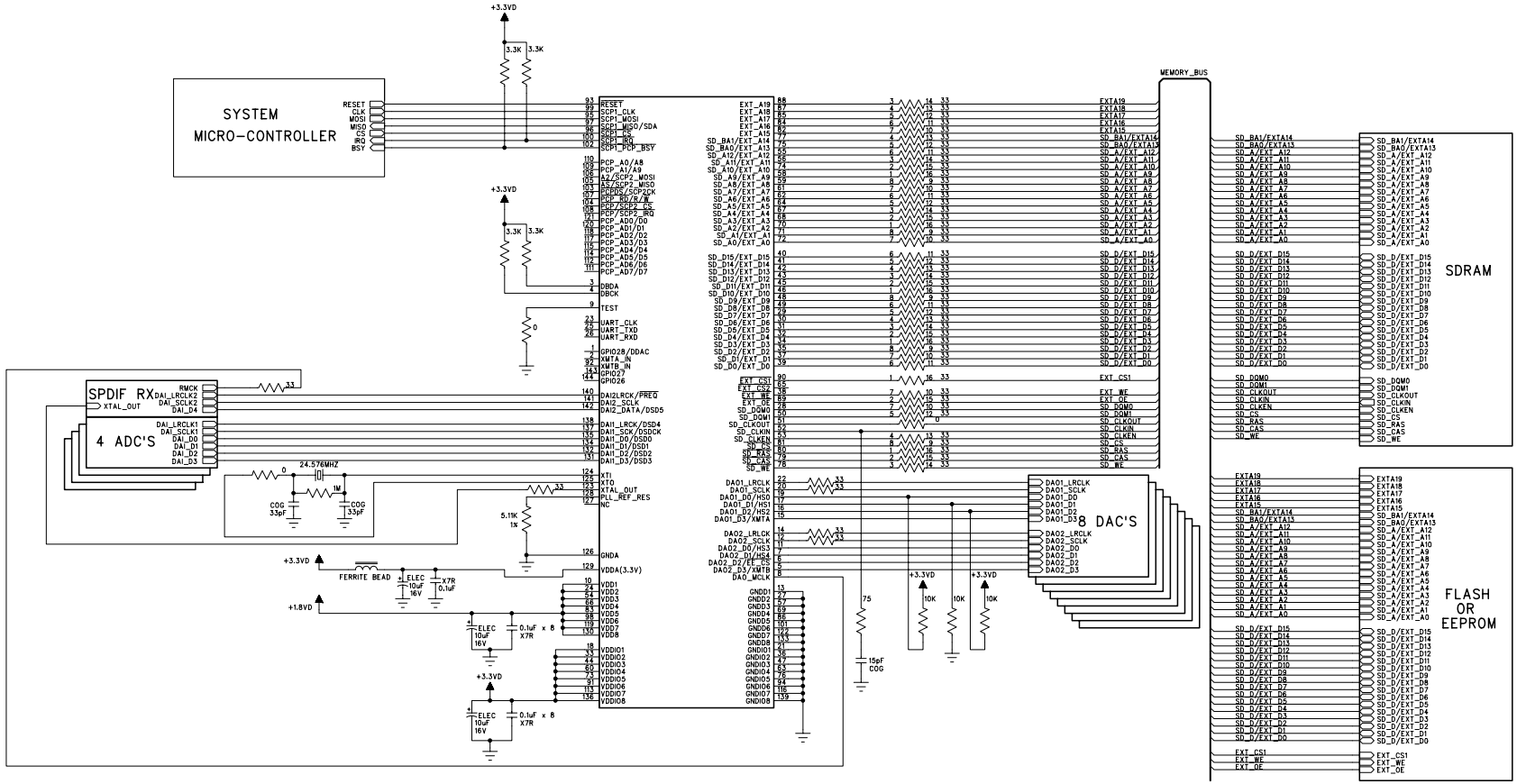
Note: For 16-Mbit parts, SD_BA0 is used as a bank select. For 64-Mbit and 128-Mbit parts SD_BA[1:0] is the 2-bit bank select.

Figure 9-1. LQFP-144, I²C Control, Serial FLASH, SDRAM, 7 DACs



Note: For 16-Mbit parts, SD_BA0 is used as a bank select. For 64-Mbit and 128-Mbit parts SD_BA[1:0] is the 2-bit bank select.

Figure 9-2. LQFP-144, SPI Control, Serial FLASH, SDRAM, 7 DACs



Note: For 16-Mbit parts, SD_BA0 is used as a bank select. For 64-Mbit and 128-Mbit parts SD_BA[1:0] is the 2-bit bank select.

Figure 9-3. LQFP-144, SPI Control, Serial FLASH, SDRAM, 8 DACs

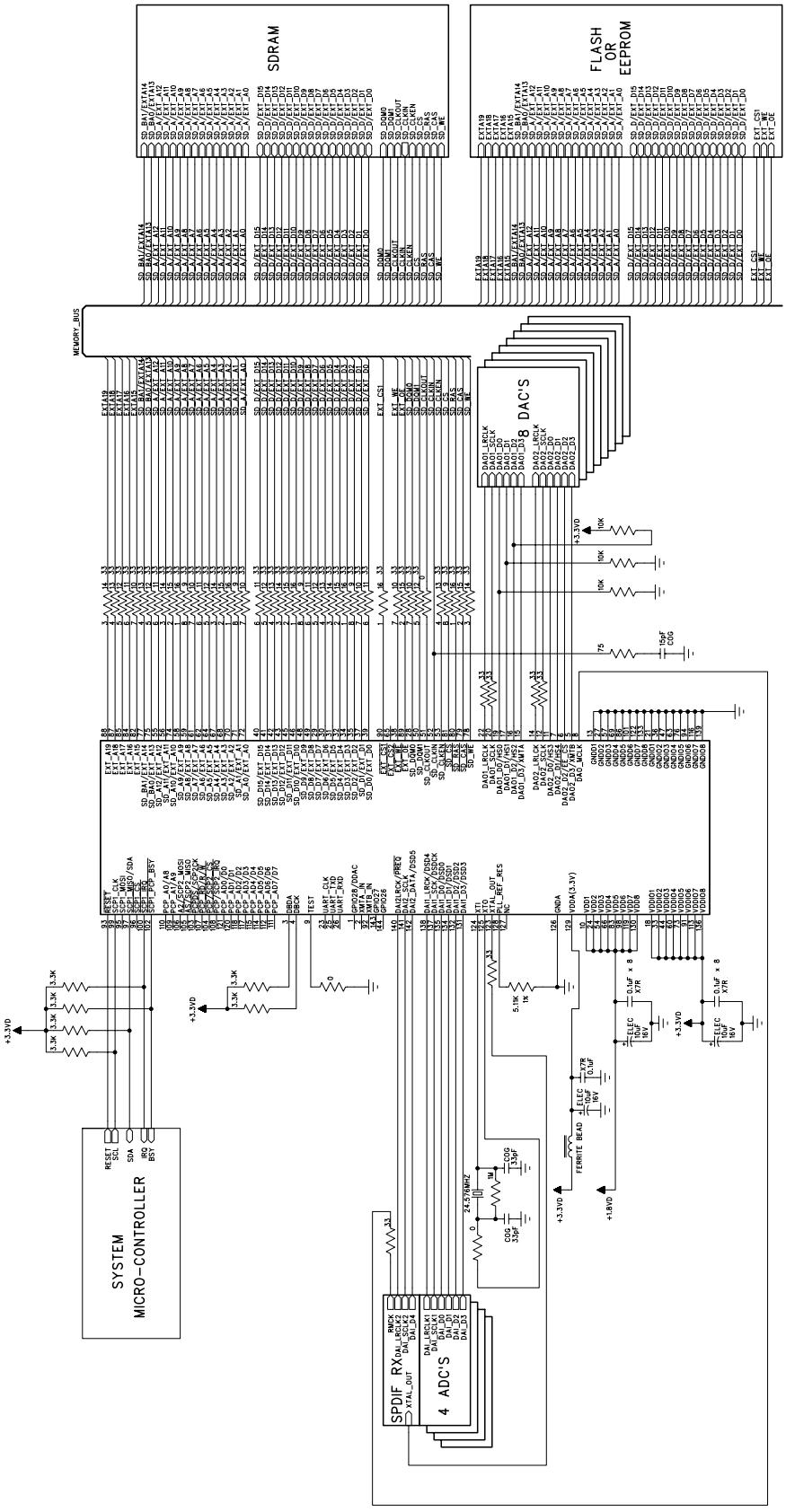
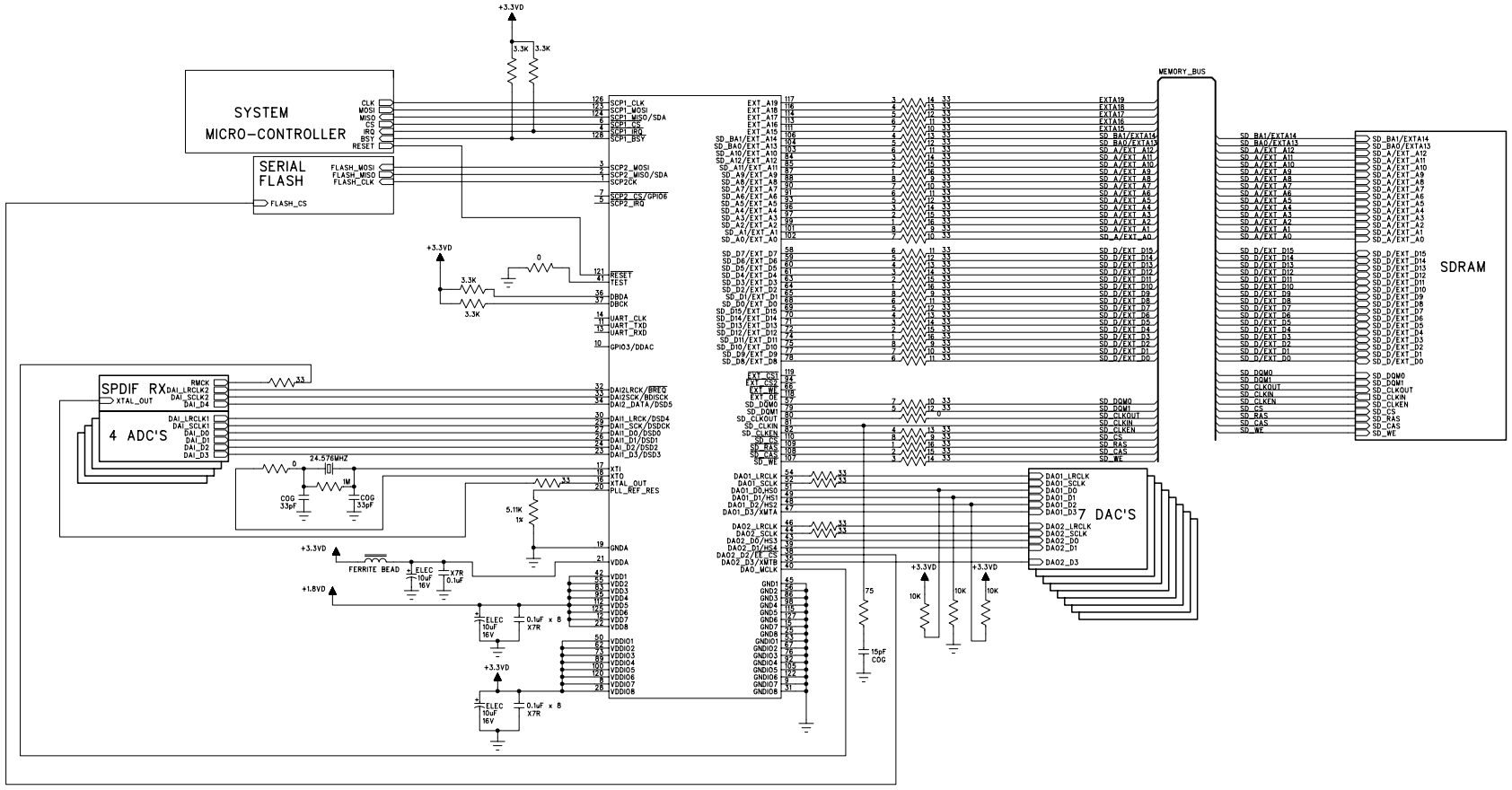


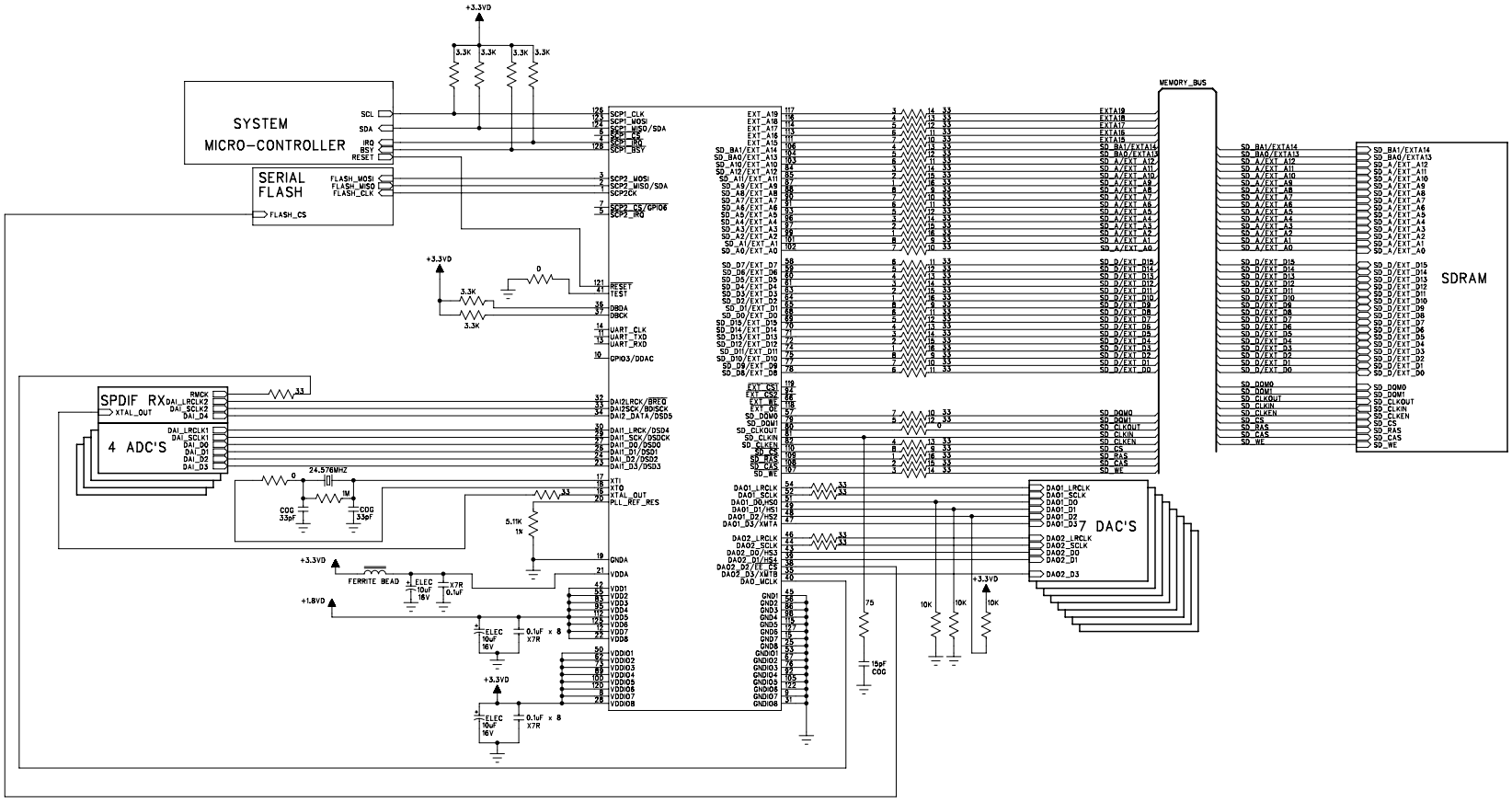
Figure 9-4. LQFP-144, I²C Control, Parallel Flash, SDRAM, 8 DACs

Note: For 16-Mbit parts, SD_BA0 is used as a bank select. For 64-Mbit and 128-Mbit parts SD_BA[1:0] is the 2-bit bank select.



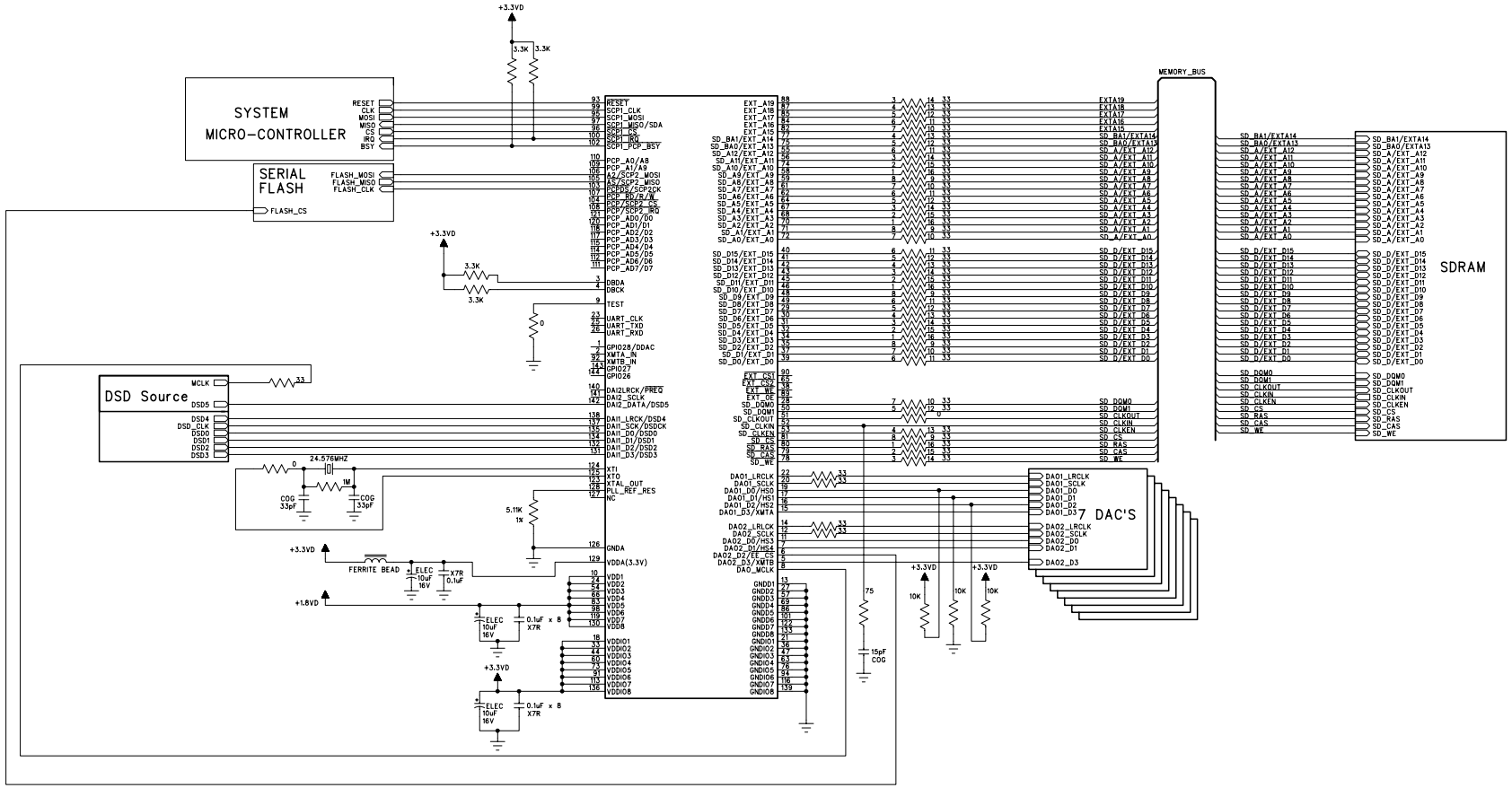
Note: For 16-Mbit parts, SD_BA0 is used as a bank select. For 64-Mbit and 128-Mbit parts SD_BA[1:0] is the 2-bit bank select.

Figure 9-5. LQFP-128, SPI Control, Parallel Flash, SDRAM, 8 DACs



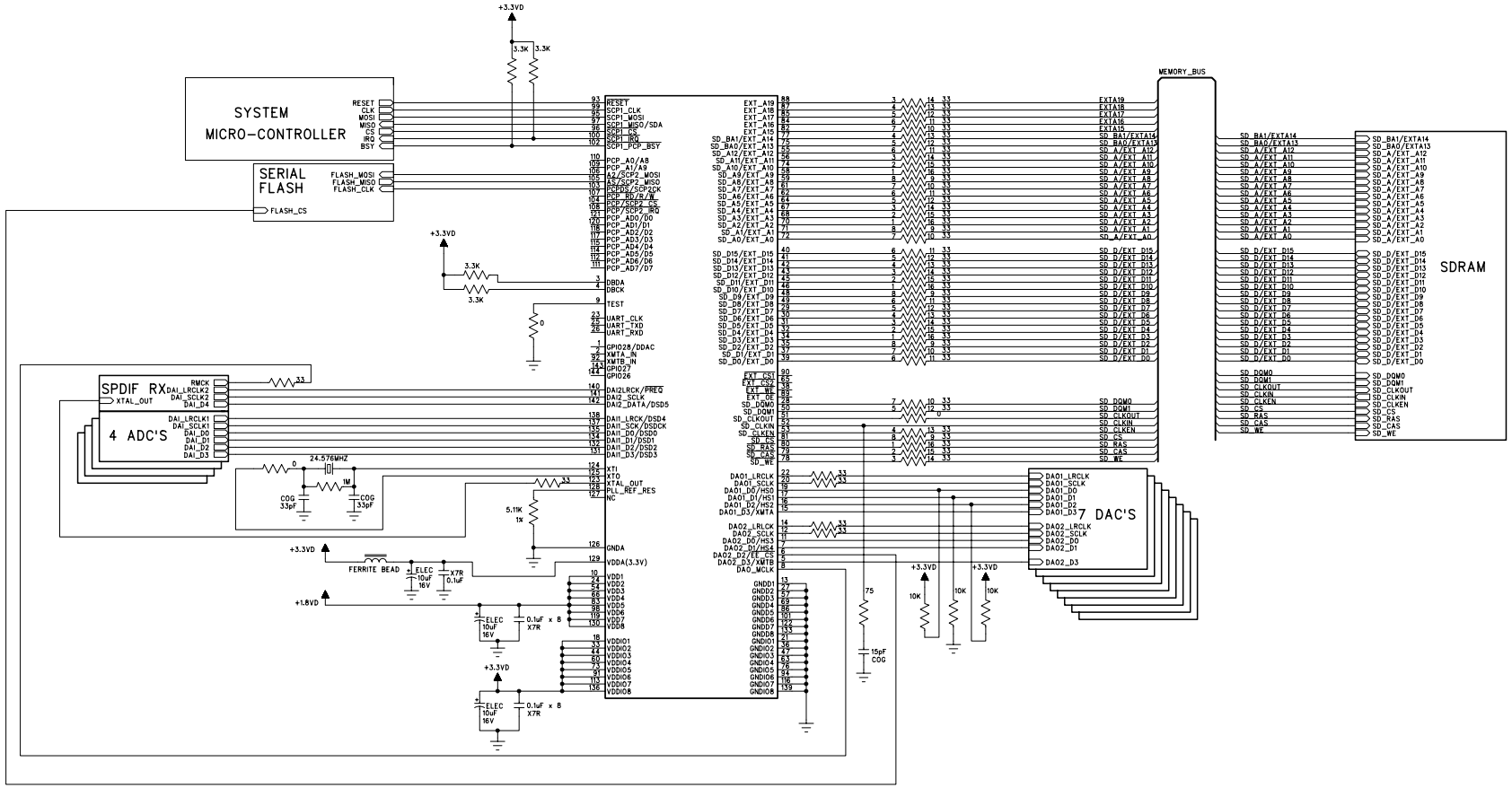
Note: For 16-Mbit parts, SD_BA0 is used as a bank select. For 64-Mbit and 128-Mbit parts SD_BA[1:0] is the 2-bit bank select.

Figure 9-6. LQFP-128, I^{2}C Control, Serial FLASH, DSD Audio Input, SDRAM, 7 DACs



Note: For 16-Mbit parts, SD_BA0 is used as a bank select. For 64-Mbit and 128-Mbit parts SD_BA[1:0] is the 2-bit bank select.

Figure 9-7. LQFP-144, SPI Control, Serial FLASH, DSD Audio Input, SDRAM, 7 DACs



Note: For 16-Mbit parts, SD_BA0 is used as a bank select. For 64-Mbit and 128-Mbit parts SD_BA[1:0] is the 2-bit bank select.

Figure 9-8. LQFP-144, SPI Control, Serial FLASH, DSD Audio Input, SDRAM, 7 DACs

9.2 Pin Description

9.2.1 Power and Ground

The following sections describe the CS4953xx power and ground pins. Decoupling and conditioning of the power supplies is also discussed. Following the recommendations for decoupling and power conditioning will help to ensure reliable performance.

9.2.1.1 Power

The CS4953xx Family of DSPs take two supply voltages — the core supply voltage (VDD) and the I/O supply voltage (VDDIO). There is also a separate analog supply voltage required for the internal PLL (VDDA). These pins are described in the following tables and descriptions.

The DSP Core supply voltage pins require a nominal 1.8V. The DSP I/O supply voltage pins require a nominal 3.3V.

Table 9-1. Core Supply Pins

LQFP-144 Pin #	LQFP-128 Pin #	Pin Name	Pin Type	Pin Description
10	42	VDD1	Input	1.8V DSP Core supply. This powers all internal logic and the on-chip SRAMs and ROMs
24	55	VDD2		
54	83	VDD3		
66	95	VDD4		
83	112	VDD5		
98	125	VDD6		
119	12	VDD7		
130	22	VDD8		

Table 9-2. I/O Supply Pins

LQFP-144 Pin #	LQFP-128 Pin #	Pin Name	Pin Type	Pin Description
18	50	VDDIO1	Input	3.3V I/O supply
33	62	VDDIO2		
44	73	VDDIO3		
60	89	VDDIO4		
73	100	VDDIO5		
91	120	VDDIO6		
113	8	VDDIO7		
136	28	VDDIO8		

9.2.1.2 Ground

For two-layer circuit boards, care should be taken to have sufficient grounding between the DSP and parts in which it will be interfacing (DACs, ADCs, S/PDIF Receivers, microcontrollers, and especially external

memory). Insufficient grounding can degrade noise margins between devices resulting in data integrity problems.

Table 9-3. Core and I/O Ground Pins

LQFP-144 Pin #	LQFP-128 Pin #	Pin Name	Pin Type	Pin Description
13	45	GND1	Input	Core Ground.
27	56	GND2		
57	86	GND3		
69	98	GND4		
86	115	GND5		
101	127	GND6		
122	15	GND7		
133	25	GND8		
21	53	GNDIO1		I/O Ground
36	67	GNDIO2		
47	76	GNDIO3		
63	92	GNDIO4		
76	105	GNDIO5		
94	122	GNDIO6		
116	9	GNDIO7		
139	31	GNDIO8		

9.2.1.3 Decoupling

It is necessary to decouple the power supply by placing capacitors directly between the power and ground of the CS4953xx. Each pair of power/ground pins (VDD1/GND1, etc.) should have its own decoupling capacitor. The recommended procedure is to place a 0.1 uF capacitor as close as physically possible to each power pin connected with a wide, low-inductance trace. A bulk capacitor of at least 10 uF is recommended for each power plane.

9.2.2 PLL Filter

9.2.2.1 Analog Power Conditioning

In order to obtain the best performance from the CS4953xx's internal PLL, the analog power supply VDDA must be as noise free as possible. A ferrite bead and two capacitors should be used to filter the VDDIO to generate VDDA. This power scheme is shown in the *Typical Connection* diagrams.

Table 9-4. PLL Supply Pins

LQFP-144 Pin #	LQFP-128 Pin #	Pin Name	Pin Type	Pin Description
129	21	VDDA	Input	PLL supply. This voltage must be 3.3V. This must be clean, noise-free analog power.
126	19	GNDA	Input	PLL ground. This ground should be as noise free as possible.

9.2.3 PLL

The internal phase locked loop (PLL) of the CS4953xx requires an external current reference resistor. The resistor is used to calibrate the PLL and must meet the tolerances specified below. The layout topology is shown in the typical connection diagrams. Care should be taken when laying out the current sense circuitry to minimize trace lengths between the DSP and resistor, and to keep high-frequency signals away from the resistor. Any noise coupled onto the these traces will be directly coupled into the PLL, which could affect performance. Please see tables below for pin numbers and external component values.

Table 9-5. PLL Filter Pins

LQFP-144 Pin #	LQFP-128 Pin #	Pin Name	Pin Type	Pin Description
128	20	PLL_REF_RES	Input	Current Reference Resistor for PLL filter

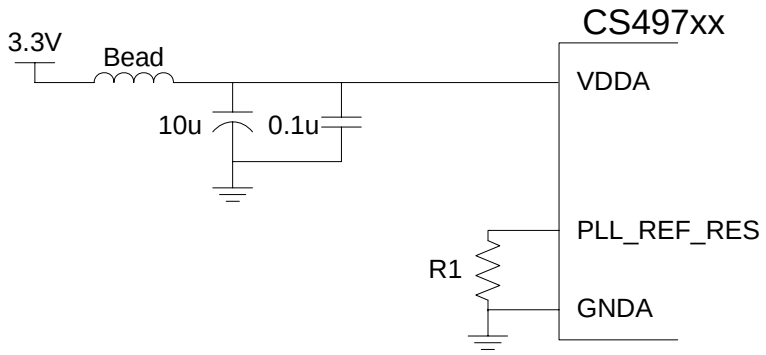


Figure 9-9. PLL Filter Topology

Table 9-6. Reference PLL Component Values

Symbol	Reference Value
R1	5.1 kΩ, 1%

9.3 Clocking

The CS4953xx incorporates a programmable phase locked loop (PLL) clock synthesizer. The PLL takes an input reference clock and produces all the clocks required to run the DSP and peripherals.

In A/V Receiver designs that require low-jitter clocks, the XTI pin is typically connected to an external 12.288 MHz or 24.576 Mhz oscillator that is used throughout the system.

The CS4953xx has a built-in crystal oscillator circuit. A parallel resonant-type crystal is connected between the XTI and XTO pins as shown in Figure 9-10. The value of C1 is specific to each crystal. The CS4953xx data sheet specifies acceptable crystal parameters (including C_L and ESR). When a crystal is used, XTAL_OUT is used to clock other devices in the system such as the S/PDIF receiver.

The PLL is controlled by the clock manager in the DSP O/S application software. AN288, *"CS4953xx Firmware User's Manual"* should be referenced regarding what CLKIN input frequency and PLL multiplier values are supported.

Table 9-7. DSP Core Clock Pins

LQFP-144 Pin #	LQFP-128 Pin #	Pin Name	Pin Type	Pin Description
123	16	XTAL_OUT	Output	Buffered version of XTI.
124	17	XTI	Input	Reference Clock Input/Crystal Oscillator Input. An external clock may be input directly to this pin or one end of a crystal may be connected to this pin.
125	18	XTO	Output	Crystal Oscillator Output. One end of a crystal oscillator is connected to this pin. This pin cannot be used to drive external circuitry.

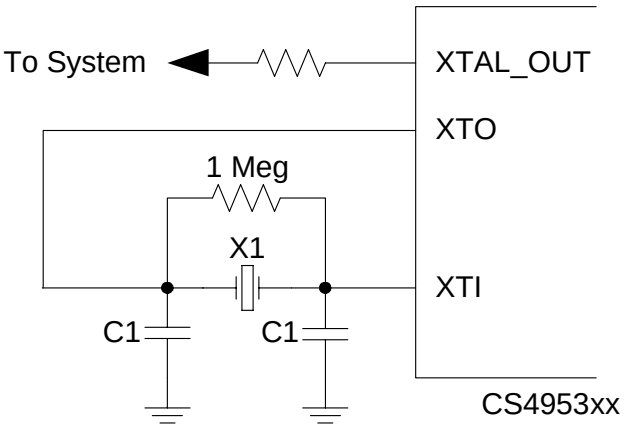


Figure 9-10. Crystal Oscillator Circuit Diagram

9.4 Control

The CS4953xx supports 5 control interface protocols: SPI, I^2C, Motorola parallel, Intel parallel, and Multiplexed Intel parallel mode. All slave serial control modes between the DSP and the host microcontroller use the Serial Control Port 1 (SCP1) pins. Parallel slave control modes are implemented on the Parallel Control Port (PCP) pins. A second serial control port (SCP2) is available for master mode applications.

9.4.1 Operational Mode

The control interface protocol used is determined by the state of the Hardware Strap pins, HS[4:0] which are sampled at the rising edge of RESET. The HS[4:0] pins should be pulled to VDD or GND using 10 kΩ resistors according to the specific control mode desired as shown in Table 2-1, "Operation Modes" on page 2-3.

The following sections describe the pins used for the 5 control modes. For example diagrams of system connection, please see Section 9.1, "Typical Connection Diagrams" on page 9-1. For information on timing diagrams and messaging protocol to the CS4953xx, see Chapter 2, "Operational Modes".

Configuration and control of the CS4953xx decoder and its peripherals are indirectly executed through a messaging protocol supported by the operating system (O/S) running on the DSP. In other words, successful communication can only be accomplished by following the low-level hardware communication format and high-level messaging protocol. The specifications of the messaging protocol used by the O/S can be found in AN288, *"CS4953xx/Cs497xxx Firmware User's Manual"*. The system designer only needs to read the subsection describing the communication mode being used. The *CS4953xx Hardware User's Manual* explains each communication mode in more detail.

Table 9-8. $\overline{\text{Reset}}$ Pin

LQFP-144 Pin #	LQFP-128 Pin #	Pin Name	Pin Type	Pin Description
93	121	$\overline{\text{RESET}}$	Input	Reset, async. active-low Chip Reset Reset should be low at power-up to initialize the DSP and to guarantee that the device is not active during initial power-on stabilization periods.

Table 9-9. Hardware Strap Pins

LQFP-144 Pin #	LQFP-128 Pin #	Pin Name	Pin Type	Pin Description
7	39	DAO2_DATA1, HS4, GPIO19	Input	Operational Mode Select Pull-up or Pull-down resistors on these pins set the DSP operational mode at reset. Hardware Strap Mode Select The state of these pins is latched at the rising edge of $\overline{\text{RESET}}$. The boot ROM uses the state of these pins to select the boot mode.
11	43	DAO2_DATA0, HS3, GPIO18		
16	48	DAO1_DATA2, HS2, GPIO16		
17	49	DAO1_DATA1, HS1, GPIO15		
19	51	DAO1_DATA0, HS0		

9.5 144-Pin LQFP Pin Assigments

Figure 9-11 shows the 144-Pin LQFP Pin Layout.

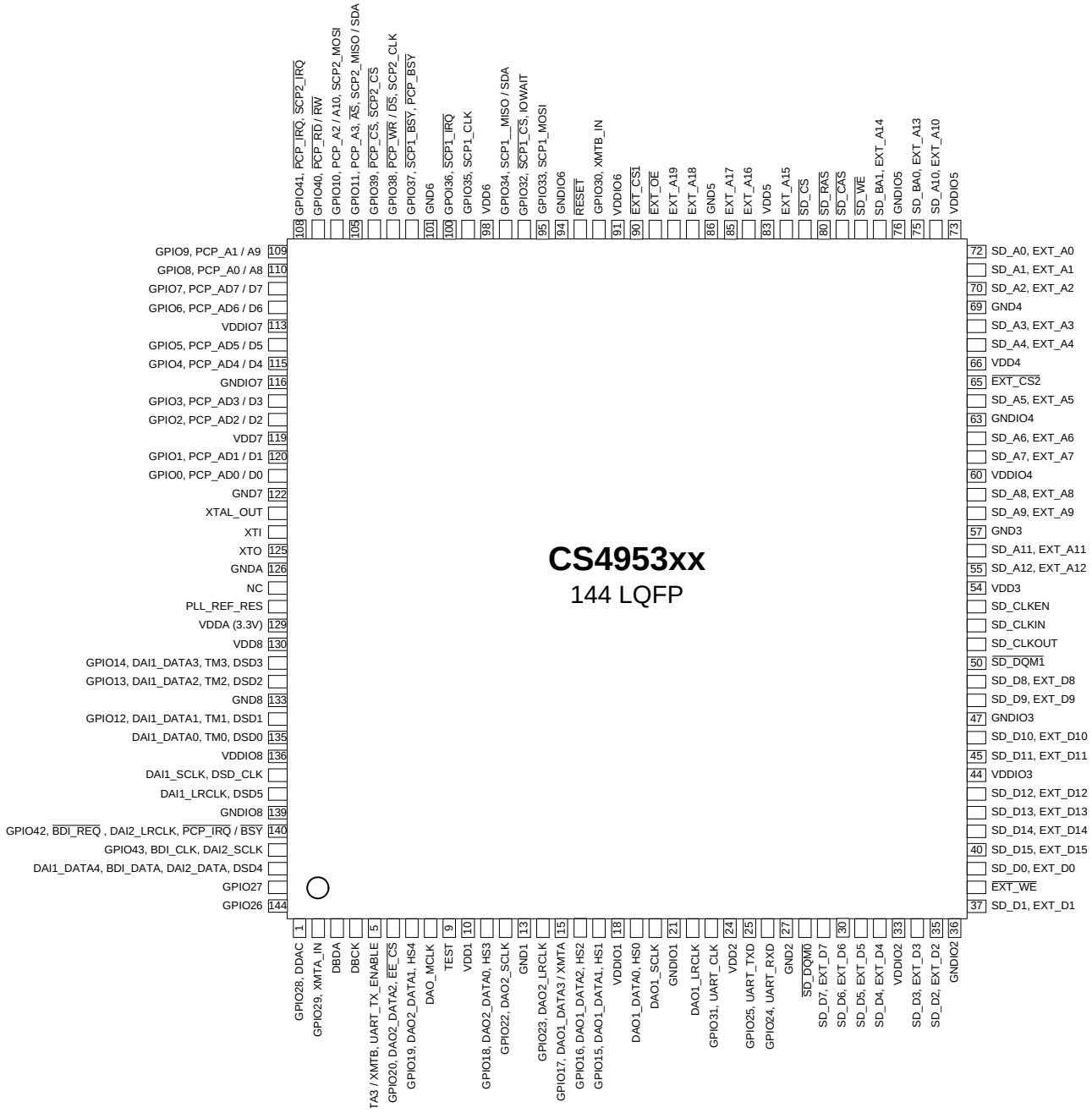


Figure 9-11. 144-Pin LQFP Pin Layout

9.6 128-Pin LQFP Pin Assigments

Figure 9-12 shows the 128-Pin LQFP Pin Layout.

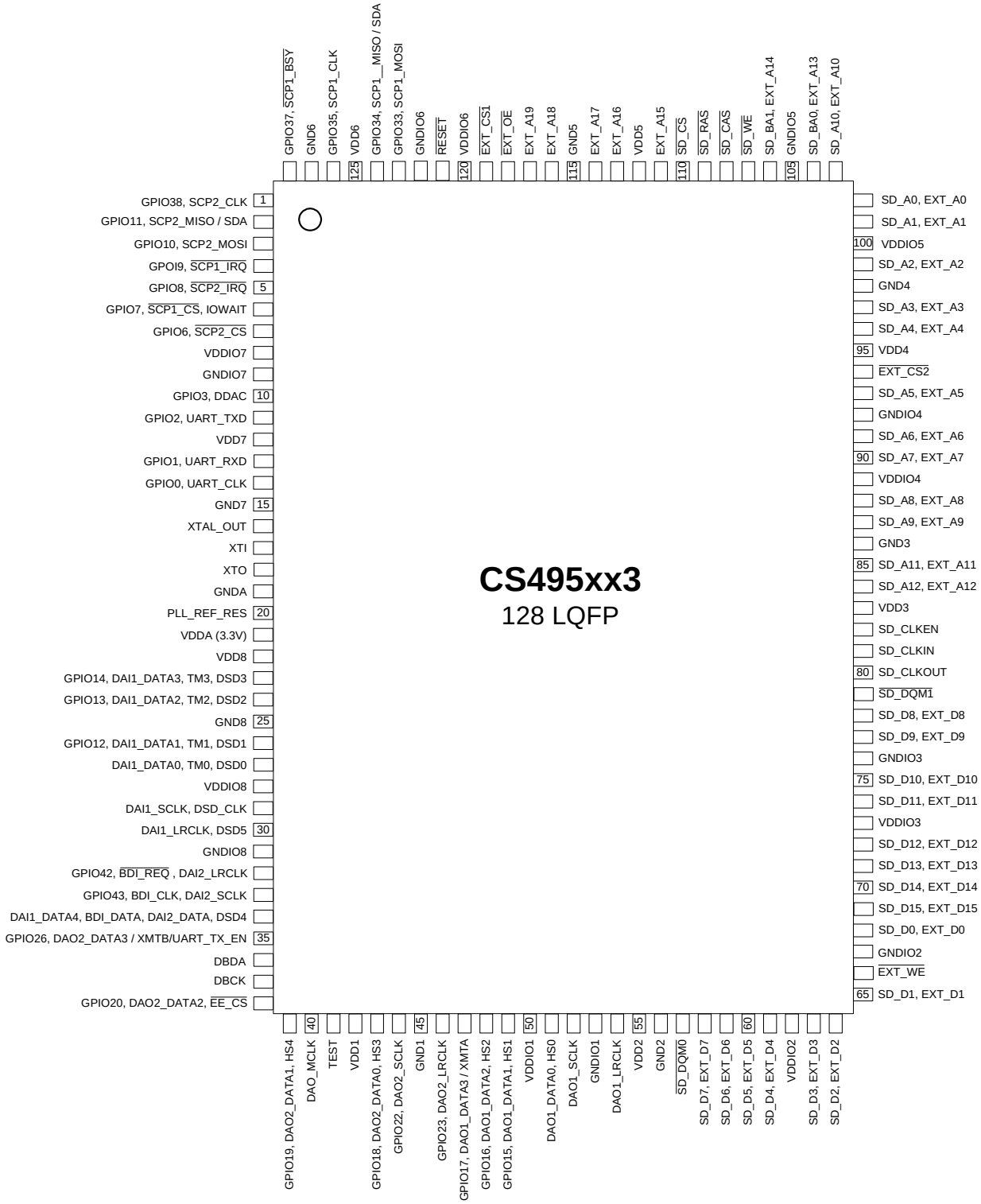


Figure 9-12. 128-Pin LQFP Pin Layout

9.7 Pin Assignments

Table 9-10 shows the names and functions for each pin.

Table 9-10. Pin Assignments

LQFP-144 Pin #	LQFP-128 Pin #	Function 1 (Default)	Description of Default Function	Secondary Functions	Description of Secondary Functions	Pwr	Type	Reset State	Pullup at Reset
1	-	GPIO28	General Purpose Input/Output			3.3V (5V tol)	BiDir	IN	Y
2	-	GPIO29	General Purpose Input/Output	1. XMTA_IN	1. S/PDIF Pass-thru Input.	3.3V (5V tol)	BiDir	IN	Y
3	36	DBDA	Debug Data			3.3V (5V tol)	In/OD	IN	Y
4	37	DBCK	Debug Clock			3.3V (5V tol)	In/OD	IN	Y
5	-	GPIO21	General Purpose Input/Output	1. DAO2_DATA3 2. XMTB 3. UART_TX_ENABLE	1. Digital Audio Output 3. 2. Outputs IEC60958/61937 format bi-phase mark encoded S/PDIF data. 3. Enable the UART_TX pin.	3.3V (5V tol	BiDir	IN	Y
6	38	GPIO20	General Purpose Input/Output	1. DAO2_DATA2 2. EE_CS	1.Digital Audio Output 2. 2. EEPROM Boot Chip Select.	3.3V (5V tol	BiDir	IN	Y
7	39	GPIO19	General Purpose Input/Output	1. DAO2_DATA1 2. HS4	1. Digital Audio Output 1. 2. Hardware Strap Mode Select.	3.3V (5V tol)	BiDir	IN	Y
8	40	DAO_MCLK	Audio Master Clock			3.3V (5V tol)	BiDir	IN	Y
9	41	TEST	Test			3.3V (5V tol)	In		
10	42	VDD1	Core power supply voltage			1.8V	PWR		
11	43	GPIO18	General Purpose Input/Output	1. DAO2_DATA0 2. HS3	1. Digital Audio Output 0. 2. Hardware Strap Mode Select.	3.3V (5V tol)	BiDir	IN	Y
12	44	GPIO22	General Purpose Input/Output	DAO2_SCLK	PCM Audio Bit Clock.	3.3V (5V tol)	BiDir	IN	Y
13	45	GNDD1	Core ground			0V	PWR		

Table 9-10. Pin Assignments (Continued)

LQFP-144 Pin #	LQFP-128 Pin #	Function 1 (Default)	Description of Default Function	Secondary Functions	Description of Secondary Functions	Pwr	Type	Reset State	Pullup at Reset
14	46	GPIO23	General Purpose Input/Output	DAO2_LRCLK	Serial PCM Audio Sample Rate Clock for the serial data pins: (DAO2_DATA0, DAO2_DATA1, DAO2_DATA2, DAO2_DATA3).	3.3V (5V tol)	BiDir	IN	Y
15	47	GPIO17	General Purpose Input/Output	1. DAO1_DATA3 2. XMTA	1. Digital Audio Output 3. 2. S/PDIF Audio Output A.	3.3V (5V tol)	BiDir	IN	Y
16	48	GPIO16	General Purpose Input/Output	1. DAO1_DATA2 2. HS2	1. Digital Audio Output 2. 2. Hardware Strap Mode Select.	3.3V (5V tol)	BiDir	IN	Y
17	49	GPIO15	General Purpose Input/Output	1. DAO1_DATA1 2. HS1	1. Digital Audio Output 1. 2. Hardware Strap Mode Select.	3.3V (5V tol)	BiDir	IN	Y
18	50	VDDIO1	I/O power supply voltage			3.3V	PWR		
19	51	DAO1_DATA0	Digital Audio Output 0	HS0	Hardware Strap Mode Select.	3.3V (5V tol)	BiDir	IN	
20	52	DAO1_SCLK	PCM Audio Bit Clock			3.3V (5V tol)	BiDir	IN	Y
21	53	GNDIO1	I/O ground			0V	PWR		
22	54	DAO1_LRCLK	PCM Audio Sample Rate Clock			3.3V (5V tol)	BiDir	IN	Y
23	-	GPIO31	General Purpose Input/Output	UART_CLK	UART Clock.	3.3V (5V tol)	BiDir	IN	Y
24	55	VDD2	Core power supply voltage			1.8V	PWR		
25	-	GPIO25	General Purpose Input/Output	UART_TXD	UART Output.	3.3V (5V tol)	BiDir	IN	Y
26	-	GPIO24	General Purpose Input/Output	UART_RXD	UART Input.	3.3V (5V tol)	BiDir	IN	Y
27	56	GNDD2	Core ground			0V	PWR		
28	57	SD_DQM0	SDRAM Data Mask 0			3.3V (5V tol)	OUT		

Table 9-10. Pin Assignments (Continued)

LQFP-144 Pin #	LQFP-128 Pin #	Function 1 (Default)	Description of Default Function	Secondary Functions	Description of Secondary Functions	Pwr	Type	Reset State	Pullup at Reset
29	58	SD_D7	SDRAM Data Bit 7	EXT_D7	Flash Data Bit 7.	3.3V (5V tol)	BiDir	IN	Y
30	59	SD_D6	SDRAM Data Bit 6	EXT_D6	Flash Data Bit 6.	3.3V (5V tol)	BiDir	IN	Y
31	60	SD_D5	SDRAM Data Bit 5	EXT_D5	Flash Data Bit 5.	3.3V (5V tol)	BiDir	IN	Y
32	61	SD_D4	SDRAM Data Bit 4	EXT_D4	Flash Data Bit 4.	3.3V (5V tol)	BiDir	IN	Y
33	62	VDDIO2	I/O power supply voltage			3.3V	PWR		
34	63	SD_D3	SDRAM Data Bit 3	EXT_D3	Flash Data Bit 3.	3.3V (5V tol)	BiDir	IN	Y
35	64	SD_D2	SDRAM Data Bit 2	EXT_D2	Flash Data Bit 2.	3.3V (5V tol)	BiDir	IN	Y
36	67	GNDIO2	I/O ground			0V	PWR		
37	65	SD_D1	SDRAM Data Bit 1	EXT_D1	Flash Data Bit 1.	3.3V (5V tol)	BiDir	IN	Y
38	66	EXT_WE	Flash Write Enable			3.3V (5V tol)	OUT		
39	68	SD_D0	SDRAM Data Bit 0	EXT_D0	Flash Data Bit 0	3.3V (5V tol)	BiDir	IN	Y
40	69	SD_D15	SDRAM Data Bit 15	EXT_D15	Flash Data Bit 15	3.3V (5V tol)	BiDir	IN	Y
41	70	SD_D14	SDRAM Data Bit 14	EXT_D14	Flash Data Bit 14	3.3V (5V tol)	BiDir	IN	Y
42	71	SD_D13	SDRAM Data Bit 13	EXT_D13	Flash Data Bit 13	3.3V (5V tol)	BiDir	IN	Y
43	72	SD_D12	SDRAM Data Bit 12	EXT_D12	Flash Data Bit 12	3.3V (5V tol)	BiDir	IN	Y
44	73	VDDIO3	I/O power supply voltage			3.3V	PWR		

Table 9-10. Pin Assignments (Continued)

LQFP-144 Pin #	LQFP-128 Pin #	Function 1 (Default)	Description of Default Function	Secondary Functions	Description of Secondary Functions	Pwr	Type	Reset State	Pullup at Reset
45	74	SD_D11	SDRAM Data Bit 11	EXT_D11	Flash Data Bit 11	3.3V (5V tol)	BiDir	IN	Y
46	75	SD_D10	SDRAM Data Bit 10	EXT_D10	Flash Data Bit 10	3.3V (5V tol)	BiDir	IN	Y
47	76	GNDIO3	I/O ground			0V	PWR		
48	77	SD_D9	SDRAM Data Bit 9	EXT_D9	Flash Data Bit 9	3.3V (5V tol)	BiDir	IN	Y
49	78	SD_D8	SDRAM Data Bit 8	EXT_D8	Flash Data Bit 8	3.3V (5V tol)	BiDir	IN	Y
50	79	SD_DQM1	SDRAM Data Mask 1			3.3V (5V tol)	OUT		
51	80	SD_CLKOUT	SDRAM Clock Output			3.3V (5V tol)	OUT		
52	81	SD_CLKIN	SDRAM Clock Input			3.3V (5V tol)	In		Y
53	82	SD_CLKEN	SDRAM Clock Enable			3.3V (5V tol)	OUT		
54	83	VDD3	Core power supply voltage			1.8V	PWR		
55	84	SD_A12	SDRAM Address Bit 12	EXT_A12	Flash Address Bit 12	3.3V (5V tol)	OUT		
56	85	SD_A11	SDRAM Address Bit 11	EXT_A11	Flash Address Bit 11	3.3V (5V tol)	OUT		
57	86	GNDD3	Core ground			0V	PWR		
58	87	SD_A9	SDRAM Address Bit 9	EXT_A9	Flash Address Bit 9	3.3V (5V tol)	OUT		
59	88	SD_A8	SDRAM Address Bit 8	EXT_A8	Flash Address Bit 8	3.3V (5V tol)	OUT		
60	89	VDDIO4	I/O power supply voltage			3.3V	PWR		
61	90	SD_A7	SDRAM Address Bit 7	EXT_A7	Flash Address Bit 7	3.3V (5V tol)	OUT		

Table 9-10. Pin Assignments (Continued)

LQFP-144 Pin #	LQFP-128 Pin #	Function 1 (Default)	Description of Default Function	Secondary Functions	Description of Secondary Functions	Pwr	Type	Reset State	Pullup at Reset
62	91	SD_A6	SDRAM Address Bit 6	EXT_A6	Flash Address Bit 6	3.3V (5V tol)	OUT		
63	92	GNDIO4	I/O ground			0V	PWR		
64	93	SD_A5	SDRAM Address Bit 5	EXT_A5	Flash Address Bit 5	3.3V (5V tol)	OUT		
65	94	EXT_CS2	Chip Select 2			3.3V (5V tol)	OUT		
66	95	VDD4	Core power supply voltage			1.8V	PWR		
67	96	SD_A4	SDRAM Address Bit 4	EXT_A4	Flash Address Bit 4	3.3V (5V tol)	OUT		
68	97	SD_A3	SDRAM Address Bit 3	EXT_A3	Flash Address Bit 3	3.3V (5V tol)	OUT		
69	98	GNDD4	Core ground			0V	PWR		
70	99	SD_A2	SDRAM Address Bit 2	EXT_A2	Flash Address Bit 2	3.3V (5V tol)	OUT		
71	101	SD_A1	SDRAM Address Bit 1	EXT_A1	Flash Address Bit 1	3.3V (5V tol)	OUT		
72	102	SD_A0	SDRAM Address Bit 0	EXT_A0	Flash Address Bit 0	3.3V (5V tol)	OUT		
73	100	VDDIO5	I/O power supply voltage			3.3V	PWR		
74	103	SD_A10	SDRAM Address Bit 10	EXT_A10	Flash Address Bit 10	3.3V (5V tol)	OUT		
75	104	SD_BA0	SDRAM Bank Address 0	EXT_A13	Flash Address Bit 13	3.3V (5V tol)	OUT		
76	105	GNDIO5	I/O ground			0V	PWR		
77	106	SD_BA1	SDRAM Bank Address 1	EXT_A14	Flash Address Bit 14	3.3V (5V tol)	OUT		
78	107	$\overline{\text{SD\_WE}}$	SDRAM Write Enable			3.3V (5V tol)	OUT		

Table 9-10. Pin Assignments (Continued)

LQFP-144 Pin #	LQFP-128 Pin #	Function 1 (Default)	Description of Default Function	Secondary Functions	Description of Secondary Functions	Pwr	Type	Reset State	Pullup at Reset
79	108	$\overline{\text{SD\_CAS}}$	SDRAM Column Address Strobe			3.3V (5V tol)	OUT		
80	109	$\overline{\text{SD\_RAS}}$	SDRAM Row Address Strobe			3.3V (5V tol)	OUT		
81	110	$\overline{\text{SD\_CS}}$	SDRAM Chip Select			3.3V (5V tol)	OUT		
82	111	EXT_A15	Flash Address Bit 15			3.3V (5V tol)	OUT		
83	112	VDD5	Core power supply voltage			1.8V	PWR		
84	113	EXT_A16	Flash Address Bit 16			3.3V (5V tol)	OUT		
85	114	EXT_A17	Flash Address Bit 17			3.3V (5V tol)	OUT		
86	115	GNDD5	Core ground			0V	PWR		
87	116	EXT_A18	Flash Address Bit 18			3.3V (5V tol)	OUT		
88	117	EXT_A19	Flash Address Bit 19			3.3V (5V tol)	OUT		
89	118	$\overline{\text{EXT\_OE}}$	Flash Output Enable			3.3V (5V tol)	OUT		
90	119	$\overline{\text{EXT\_CS1}}$	Active-low Flash chip select			3.3V (5V tol)	OUT		
91	120	VDDIO6	I/O power supply voltage			3.3V	PWR		
92	-	GPIO30	General Purpose Input/Output	1. CSW_U 2. XMTB_IN	1. Channel status user data input 2. S/PDIF Pass-thru Input	3.3V (5V tol)	BiDir	IN	Y
93	121	$\overline{\text{RESET}}$	Chip Reset			3.3V (5V tol)	In		
94	122	GNDIO6	I/O ground			0V	PWR		

Table 9-10. Pin Assignments (Continued)

LQFP-144 Pin #	LQFP-128 Pin #	Function 1 (Default)	Description of Default Function	Secondary Functions	Description of Secondary Functions	Pwr	Type	Reset State	Pullup at Reset
95	123	GPIO33	General Purpose Input/Output	SCP1_MOSI	SPI Mode Master Data Output/Slave Data Input	3.3V (5V tol)	BiDir	IN	Y
96	-	GPIO32	General Purpose Input/Output	1. $\overline{\text{SCP1\_CS}}$ 2. IOWAIT	1. SPI Chip Select 2. SRAM Hold-Off Handshake	3.3V (5V tol)	BiDir	IN	Y
97	124	GPIO34	General Purpose Input/Output	1. SCP1_MISO 2. SCP1_SDA	1. SPI Mode Master Data Input/Slave Data Output 2. I^2C Mode Master/Slave Data IO	3.3V (5V tol)	BiDir/ OD	IN	Y
98	125	VDD6	Core power supply voltage			1.8V	PWR		
99	126	GPIO35	General Purpose Input/Output	SCP1_CLK	SPI/I^2C Control Port Clock	3.3V (5V tol)	BiDir/ OD	IN	Y
100	-	GPIO36	General Purpose Input/Output	$\overline{\text{SCP1\_IRQ}}$	Serial Control Port Data Ready Interrupt Request	3.3V (5V tol)	BiDir/ OD	IN	Y
101	127	GNDD6	Core ground			0V	PWR		
102	-	GPIO37	General Purpose Input/Output	1. $\overline{\text{SCP1\_BSY}}$ 2. PCP_BSY	1. Serial Control Port 1 Input Busy 2. Parallel Control Port Input Busy	3.3V (5V tol)	BiDir/ OD	IN	Y
-	128	GPIO37	General Purpose Input/Output	1. $\overline{\text{SCP1\_BSY}}$	1. Serial Control Port 1 Input Busy	3.3V (5V tol)	BiDir/ OD	IN	Y
103	-	GPIO38	General Purpose Input/Output	1. $\overline{\text{PCP\_WR}}$ 2. $\overline{\text{PCP\_DS}}$ 3. SCP2_CLK	1. Parallel Port Write Select (Intel Mode) 2. Parallel Port Data Strobe (Motorola and Multiplexed Mode) 3. SPI/I^2C Control Port Clock	3.3V (5V tol)	BiDir/ OD	IN	Y
-	1	GPIO38	General Purpose Input/Output	1. SCP2_CLK	1. SPI/I^2C Control Port Clock	3.3V (5V tol)	BiDir/ OD	IN	Y

Table 9-10. Pin Assignments (Continued)

LQFP-144 Pin #	LQFP-128 Pin #	Function 1 (Default)	Description of Default Function	Secondary Functions	Description of Secondary Functions	Pwr	Type	Reset State	Pullup at Reset
104	-	GPIO39	General Purpose Input/Output	1. $\overline{\text{PCP\_CS}}$ 2. $\overline{\text{SCP2\_CS}}$	1. Parallel Port Chip Select (Intel/Motorola/Multiplexed Mode) 2. SPI Chip Select	3.3V (5V tol)	BiDir	IN	Y
105	-	GPIO11	General Purpose Input/Output	1. $\overline{\text{PCP\_A3}}$ 2. $\overline{\text{PCP\_AS}}$ 3. SCP2_MISO 4. SCP2_SDA	1. Parallel Control Port Address Bit 3 2. Parallel Control Port Address Strobe 3. SPI Mode Master Data Input/Slave Data Output 4. I^2C Mode Master/Slave Data IO	3.3V (5V tol)	BiDir/ OD	IN	Y
-	2	GPIO11	General Purpose Input/Output	1. SCP2_MISO 2. SCP2_SDA	1. SPI Mode Master Data Input/Slave Data Output 2. I^2C Mode Master/Slave Data IO	3.3V (5V tol)	BiDir/ OD	IN	Y
106	-	GPIO10	General Purpose Input/Output	1. PCP_A2 2. PCP_A10 3. SCP2_MOSI	1. Parallel Control Port Address Bit 2 2. Parallel Control Port Address Bit 10 3. SPI Mode Master Data Output/Slave Data Input	3.3V (5V tol)	BiDir	IN	Y
-	3	GPIO10	General Purpose Input/Output	1. SCP2_MOSI	1. SPI Mode Master Data Output/Slave Data Input	3.3V (5V tol)	BiDir	IN	Y
107	-	GPIO40	General Purpose Input/Output	1. $\overline{\text{PCP\_RD}}$ 2. $\overline{\text{PCP\_R/W}}$	1. Parallel Read Select (Intel Mode) 2. Parallel Read/Write Select (Motorola and Multiplexed Mode)	3.3V (5V tol)	BiDir	IN	Y
108	-	GPIO41	General Purpose Input/Output	1. $\overline{\text{PCP\_IRQ}}$ 2. $\overline{\text{SCP2\_IRQ}}$	1. Parallel Control Port Data Ready Interrupt Request 2. Serial Control Port Data Ready Interrupt Request	3.3V (5V tol)	BiDir/ OD	IN	Y

Table 9-10. Pin Assignments (Continued)

LQFP-144 Pin #	LQFP-128 Pin #	Function 1 (Default)	Description of Default Function	Secondary Functions	Description of Secondary Functions	Pwr	Type	Reset State	Pullup at Reset
109	-	GPIO9	General Purpose Input/Output	1. PCP_A1 2. PCP_A9	1. Parallel Control Port Address Bit 1 2. Parallel Control Port Address Bit 9	3.3V (5V tol)	BiDir	IN	Y
-	4	GPIO9	General Purpose Input/Output	1. $\overline{\text{SCP1\_IRQ}}$	1. Serial Control Port Data Ready Interrupt Request	3.3V (5V tol)	BiDir	IN	Y
110	-	GPIO8	General Purpose Input/Output	1. PCP_A0 2. PCP_A8	1. Parallel Control Port Address Bit 0 2. Parallel Control Port Address Bit 8	3.3V (5V tol)	BiDir	IN	Y
-	5	GPIO8	General Purpose Input/Output	1. $\overline{\text{SCP2\_IRQ}}$	1. Serial Control Port Data Ready Interrupt Request	3.3V (5V tol)	BiDir	IN	Y
111	-	GPIO7	General Purpose Input/Output	1. PCP_D7 2. PCP_AD7	1. Parallel Control Port Data Bus 2. Parallel Control Port Multiplexed Address and Data Bus	3.3V (5V tol)	BiDir	IN	Y
-	6	GPIO7	General Purpose Input/Output	1. $\overline{\text{SCP1\_CS}}$ 2. IOWAIT	1. SPI Chip Select 2. SRAM Hold-Off Handshake	3.3V (5V tol)	BiDir	IN	Y
112	-	GPIO6	General Purpose Input/Output	1. PCP_D6 2. PCP_AD6	1. Parallel Control Port Data Bus 2. Parallel Control Port Multiplexed Address and Data Bus	3.3V (5V tol)	BiDir	IN	Y
-	7	GPIO6	General Purpose Input/Output	1. $\overline{\text{SCP2\_CS}}$	1. SPI Chip Select	3.3V (5V tol)	BiDir	IN	Y
113	8	VDDIO7	I/O power supply voltage			3.3V	PWR		
114	-	GPIO5	General Purpose Input/Output	1. PCP_D5 2. PCP_AD5	1. Parallel Control Port Data Bus 2. Parallel Control Port Multiplexed Address and Data Bus	3.3V (5V tol)	BiDir	IN	Y

Table 9-10. Pin Assignments (Continued)

LQFP-144 Pin #	LQFP-128 Pin #	Function 1 (Default)	Description of Default Function	Secondary Functions	Description of Secondary Functions	Pwr	Type	Reset State	Pullup at Reset
115	-	GPIO4	General Purpose Input/Output	1. PCP_D4 2. PCP_AD4	1. Parallel Control Port Data Bus 2. Parallel Control Port Multiplexed Address and Data Bus	3.3V (5V tol)	BiDir	IN	Y
116	9	GNDIO7	I/O ground			0V	PWR		
117	-	GPIO3	General Purpose Input/Output	1. PCP_D3 2. PCP_AD3	1. Parallel Control Port Data Bus 2. Parallel Control Port Multiplexed Address and Data Bus	3.3V (5V tol)	BiDir	IN	Y
-	10	GPIO3	General Purpose Input/Output			3.3V (5V tol)	BiDir	IN	Y
118	-	GPIO2	General Purpose Input/Output	1. PCP_D2 2. PCP_AD2	1. Parallel Control Port Data Bus 2. Parallel Control Port Multiplexed Address and Data Bus	3.3V (5V tol)	BiDir	IN	Y
-	11	GPIO2	General Purpose Input/Output	1. UART_TXD	1. UART Output	3.3V (5V tol)	BiDir	IN	Y
119	12	VDD7	Core power supply voltage			1.8V	PWR		
120	-	GPIO1	General Purpose Input/Output	1. PCP_D1 2. PCP_AD1	1. Parallel Control Port Data Bus 2. Parallel Control Port Multiplexed Address and Data Bus	3.3V (5V tol)	BiDir	IN	Y
	13	GPIO1	General Purpose Input/Output	1. UART_RXD	1. UART Input	3.3V (5V tol)	BiDir	IN	Y
121	-	GPIO0	General Purpose Input/Output	1. PCP_D0 2. PCP_AD0	1. Parallel Control Port Data Bus 2. Parallel Control Port Multiplexed Address and Data Bus	3.3V (5V tol)	BiDir	IN	Y

Table 9-10. Pin Assignments (Continued)

LQFP-144 Pin #	LQFP-128 Pin #	Function 1 (Default)	Description of Default Function	Secondary Functions	Description of Secondary Functions	Pwr	Type	Reset State	Pullup at Reset
	14	GPIO0	General Purpose Input/Output	1. UART_CLK	1. UART Clock	3.3V (5V tol)	BiDir	IN	Y
122	15	GNDD7	Core ground			0V	PWR		
123	16	XTAL_OUT	Buffered Reference Clock Input/Crystal Oscillator Input			3.3V (5V tol)	OUT		
124	17	XTI	Reference Clock Input/Crystal Oscillator Input			3.3V (5V tol)	ANA		
125	18	XTO	Crystal Oscillator Output 1			3.3V	ANA		
126	19	GNDA	PLL ground			1.8V	PWR		
127	-	NC	Do Not Connect on PCB			1.8V	ANA		
128	20	PLL_REF_RES	Current Reference Output for PLL. Connect to resistor.			3.3V	ANA		
129	21	VDDA	PLL power.			3.3V	PWR		
130	22	VDD8	Core power supply voltage			1.8V	PWR		
131	23	GPIO14	General Purpose Input/Output	1. DAI1_DATA3 2. DSD3	1. PCM Audio Input Data 3 2. DSD Audio Input Data 3	3.3V (5V tol)	BiDir	IN	Y
132	24	GPIO13	General Purpose Input/Output	1. DAI1_DATA2 2. DSD2	1. PCM Audio Input Data 2 2. DSD Audio Input Data 2	3.3V (5V tol)	BiDir	IN	Y
133	25	GNDD8	Core ground			0V	PWR		
134	26	GPIO12	General Purpose Input/Output	1. DAI1_DATA1 2. DSD1	1. PCM Audio Input Data 1 2. DSD Audio Input Data 1	3.3V (5V tol)	BiDir	IN	Y
135	27	DAI1_DATA0	PCM Audio Input Data 0	DSD0	DSD Audio Input Data 0	3.3V (5V tol)	In		Y
136	28	VDDIO8	I/O power supply voltage			3.3V	PWR		

Table 9-10. Pin Assignments (Continued)

LQFP-144 Pin #	LQFP-128 Pin #	Function 1 (Default)	Description of Default Function	Secondary Functions	Description of Secondary Functions	Pwr	Type	Reset State	Pullup at Reset
137	29	DAI1_SCLK	PCM Audio Input Bit Clock	DSD_CLK	DSD Audio Input Clock	3.3V (5V tol)	In		Y
138	30	DAI1_LRCLK	PCM Audio Input Sample Rate (Left/ Right) Clock	DSD4	DSD Audio Input Data 4	3.3V (5V tol)	In		Y
139	31	GNDIO8	I/O ground			0V	PWR		
140	-	GPIO42	General Purpose Input/Output	1. DAI2_LRCLK 2. BDI_REQ 3. PCP_IRQ 4. PCP_BSY	1.PCM Audio Input Sample Rate Clock 2. Bursty Data Input Request 3. Parallel Control Port Data Ready Interrupt Request 4. Parallel Control Port Input Busy	3.3V (5V tol)	BiDir/ OD	IN	Y
-	32	GPIO42	General Purpose Input/Output	1. DAI2_LRCLK 2. BDI_REQ	1.PCM Audio Input Sample Rate Clock 2. Bursty Data Input Request	3.3V (5V tol)	BiDir/ OD	IN	Y
141	33	GPIO43	General Purpose Input/Output	1. DAI2_SCLK 2. BDI_CLK	1.PCM Audio Input Bit Clock 2. Bursty Data Input Bit Clock	3.3V (5V tol)	BiDir	IN	Y
142	34	DAI2_DATA	PCM Audio Input Data	1. DAI1_DATA4 2. DSD5 3. BDI_DATA	1. PCM Audio Input Data 4 2. DSD Audio Input Data 5 3. Bursty Data Input Data	3.3V (5V tol)	In		Y
143	-	GPIO27	General Purpose Input/Output			3.3V (5V tol)	BiDir	IN	Y
144	-	GPIO26	General Purpose Input/Output			3.3V (5V tol)	BiDir	IN	Y
-	35	GPIO26	General Purpose Input/Output	1. DAO2_DATA3 2. XMTB 3. UART_TX_ENABLE	1. Digital Audio Output 3. 2. Outputs IEC60958/61937 format bi-phase mark encoded S/PDIF data 3. Enable the UART_TX pin	3.3V (5V tol)	BiDir	IN	Y

Revision History

Revision	Date	Changes
UM1	MAY 17, 2006	Preliminary Release
UM2	AUG 24, 2006	Updated Boot Procedure
UM3	OCT 11, 2007	Updated CrusConfig register and corrected Output D1 and Input D2 messages. Update Section 2.5.2, "Softboot Procedure" on page 2-15.
UM4	NOV 21, 2007	Added Bank Addressing note to Table 8-1 for SD_BA0/ EXT_A13 and SD_BA1/EXT_A14 signals. Added same note with each schematic in Chapter 9. Modified Note directly above Section 8.3.1, "SDRAM/Flash Interface Signals" on page 8-2
UM5	NOV 28, 2007	Changed definition for secondary functions in Section 9-10 "Pin Assignments" on page 9-17 for pins 138 and 142. Made same changes in Figure 9-11 for the 144-Pin LQFP pin layout drawing.
UM6	May 7, 2008	Changed text in Figure Titles for Figure 9-1, Figure 9-2, Figure 9-4, and Figure 9-5. Modified Note 1 under Table 8-1. Change note regarding bank selection, in Figures 9-1 to 9-7. Added new typical connection diagram in Figure 9-6, "LQFP-128, I^2C Control, Serial FLASH, DSD Audio Input, SDRAM, 7 DACs." Modified description of firmware modules offered on the CS4953xx platform on page 1-3. Updated list of sample rates supported for the Digital Audio Port in Section 6.1, "Description of Digital Audio Input Port when Configured for DSD Input" on page 6-1. Clarified availability of parallel control port in Chapter 4, "Parallel Control Port".
UM7	July 10, 2008	Added important notice a beginning of Section 8.3.2, "Configuring SDRAM/Flash Parameters" on page 8-4.
UM8	April 23, 2009	Updated Figure 3-6, adding missing ACK timing diagram. Updated Table 5-3, Table 5-4, Table 5-5. Table 5-6, and Table 7-4. Reformatted manual to current Cirrus Logic style practices. Updated Table 3-1, Table 3-2, Table 8-1 and Table 9-8, noting signals that are active low with a line over the signal.
UM9	June 5, 2009	Added XTAL/2 frequency capability for Master and Slave boot in Section 2.3.1.1.1.(Steps 5 and 8) and in Section 2.3.2.1.1.(Steps 6 and 9). Updated Master and Slave boot flow charts in Figure 2-2 and Figure 2-3 to show where XTAL/2 frequency can be implemented if desired. Updated values for A1, Left Justified 24-bit, in Table 5-3. Removed overline which had designed Pins SD_DQM0 and SD_DQM1 to be Active Low in Table 9-10 and in Chapter 8, "External Memory Interfaces".

Revision	Date	Changes
UM10	February 12, 2010	Updated Table 2-6, GPIO Pins Available as EE_CS in HCMB. Added A2 parameter configuration and changed default value to Table 7-2, Output Clock Mode Configuration (Parameter A). Changed default value amd added note that was previously in the A1 table cell of Table 7-2 to the B0 table cell in Table 7-3, DAO1 & DAO2 Clocking Relationship Configuration (Parameter B). Added C Value, 16 to Table 7-4, Output DAO_SCLK/ LRCLK Configuration (Parameter C). Updated Table 7-10, S/PDIF Transmitter Configuration. Added Table 7-8. Added note at the end of Section 7.1.3 "DAO Interface Formats" on page 7-3

§§