CS49400 Family DSP

Multi-Standard Audio Decoder

Features

- CS49300 Legacy Audio Decoder Support
- Dolby Digital EX^{TM}, Dolby Pro Logic II^{TM}
- DTS-ES 96/24^{TM}, DTS 96/24^{TM}, DTS-ES Discrete 6.1^{TM}, DTS-ES Matrix 6.1^{TM}, DTS Digital Surround^{TM} and DTS Virtual 5.1^{TM}
- MPEG-2: AAC Multichannel 5.1
- MPEG Multichannel and Musicam
- MPEG-1/2, Layer III (MP3)
- DTS Neo:6^{TM}, LOGIC7$^®$, SRS Circle Surround II^{TM}
- Cirrus Extra Surround^{TM}, Cirrus Original Surround 6.1 (C.O.S. 6.1)^{TM}
- THX Surround EX^{TM}, THX Ultra2 Cinema^{TM}
- 12-Channel Serial Audio Inputs
- Integrated 8K Byte Input Buffer
- Powerful 32-bit Audio DSP
- Customer Software Security Keys
- Large On-chip X,Y, and Program RAM
- Supports SDRAM, SRAM, FLASH memories
- 16-channel PCM output
- Dual S/PDIF Transmitters
- SPI Serial, and Motorola$^®$ and Intel$^®$ Parallel Host Control Interfaces
- GPIO support for all common sub-circuits

Description

The CS49400 Audio Decoder DSP is targeted as a market-specific consumer entertainment processor for AV Receivers and DVD Audio/Video Players. The device is constructed using an enhanced version of the CS49300 Family DSP audio decoder followed by a 32-bit programmable post-processor DSP, which gives the designer the ability to add product differentiation through the Cirrus Framework^{TM} programming structure and Framework module library. Dolby Digital Pro Logic II, DTS Digital Surround, MPEG Multichannel, and Cirrus Original Surround 6.1 PCM Effects Processor (capable of generating such DSP audio modes as: Hall, Theater, Church) are included in the cost of the CS49400 Family DSP. Additional algorithms available through the Crystal Ware^{TM} Software Licensing Program, give the designer the ability to further deliver end-product differentiation.

The CS49400 contains sufficient on-chip SRAM to support decoding all major audio decoding algorithms available today including: AAC Multichannel, DTS 96/24, DTS-ES 96/24. The CS49400 also supports a glueless SDRAM/SRAM for increased all-channel delays. The SRAM interface also supports connection to an external byte-wide EPROM for code storage or Flash memory thus allowing products to be field-upgradable as new audio algorithms are developed.

This chip, teamed with Crystal Ware^{TM} certified decoder library, Cirrus digital interface products and mixed signal data converters, enables the conception and design of next generation digital entertainment products.

Ordering Information: See page 98
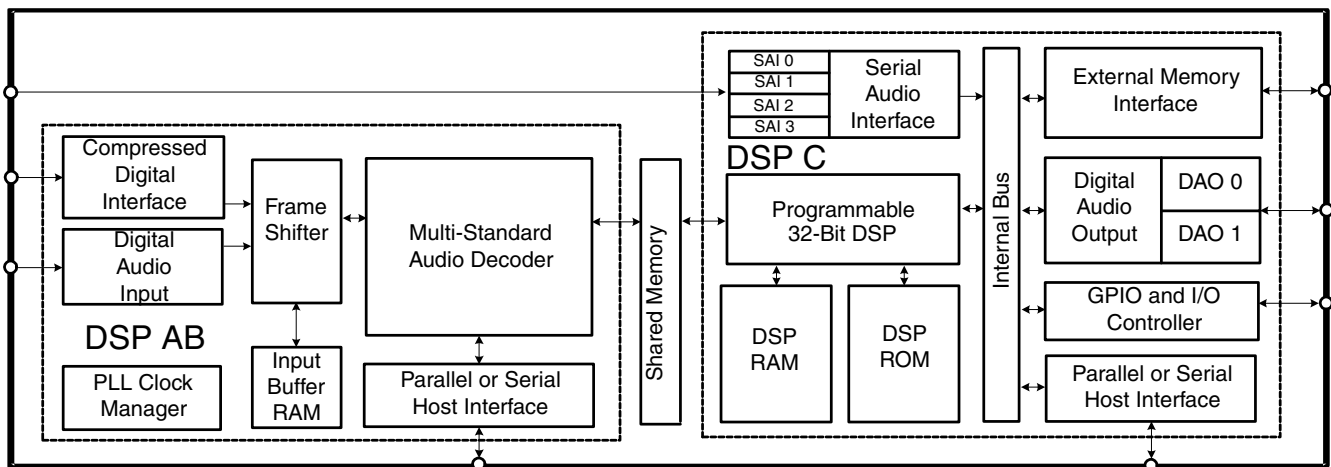
Preliminary Product Information

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

CIRRUS LOGIC®

TABLE OF CONTENTS

Contacting Cirrus Logic Support

For a complete listing of Direct Sales, Distributor, and Sales Representative contacts, visit the Cirrus Logic web site at:

http://www.cirrus.com/corporate/contacts/sales.cfm

LIST OF FIGURES

LIST OF TABLES

1.0 CHARACTERISTICS AND SPECIFICATIONS

Note: All data sheet minimum and maximum timing parameters are guaranteed over the rated voltage and temperature. Actual production testing is performed at T_A = 25 °C with an appropriate guardband to guarantee minimum and maximum timing specifications over rated voltage and temperature.

1.1 Absolute Maximum Ratings

(VSS, VSSSD, PLLVSS = 0 V; all voltages with respect to 0 V)

Parameter		Symbol	Min	Max	Unit
DC power supplies:	Core supply	VDD	−0.3	2.7	V
	PLL supply	PLLVSS	−0.3	2.7	V
	Memory supply	VDDSD	−0.3	3.6	V
	||PLLVDD| − |VDD||		-	0.3	V
Input current, any pin except supplies		I_{in}	-	±10	mA
Digital input voltage on I/O pins powered from VDD		V_{ind}	-	3.6	V
Digital input voltage on I/O pins powered from VDDSD		V_{insd}	-	3.6	V
Storage temperature		T_{stg}	−65	150	°C

Caution: Operation at or beyond these limits may result in permanent damage to the device. Normal operation is not guaranteed at these extremes.

1.2 Recommended Operating Conditions

(VSS, VSSSD, PLLVSS = 0 V; all voltages with respect to 0 V)

Parameter		Symbol	Min	Typ	Max	Unit
DC power supplies:	Core supply	VDD	2.37	2.5	2.63	V
	PLL supply	PLLVSS	2.37	2.5	2.63	V
	Memory supply	VDDSD	3.15	3.3	3.45	V
	||PLLVDD| − |VDD||				0.3	V
Ambient operating temperature		T_A	0	-	70	°C

1.3 Digital D.C. Characteristics for VDD Level I/O

(T_A = 25 °C; VDD = 2.5 V; measurements performed under static conditions.)

Parameter	Symbol	Min	Typ	Max	Unit
High-level input voltage	V_{IH}	2.0	-	-	V
Low-level input voltage	V_{IL}	-	-	0.8	V
High-level output voltage at I_O = −2.0 mA	V_{OH}	VDD × 0.9	-	-	V
Low-level output voltage at I_O = 2.0 mA	V_{OL}	-	-	VDD × 0.1	V
Input leakage current (all pins without internal pull-up resistors except CLKIN)	I_{in}	-	-	10	µA
Input leakage current (pins with internal pull-up resistors, CLKIN)				50	µA

1.4 Digital D.C. Characteristics for VDDSD Level I/O

(T_A = 25 °C;VDDSD = 3.3 V±; measurements performed under static conditions.)

Parameter	Symbol	Min	Typ	Max	Unit
High-level input voltage	V_{IH}	0.65xVDDSD			V
Low-level input voltage	V_{IL}			0.35xVDDSD	V
High-level output voltage at I_O = −2.0 mA	V_{OH}	0.9xVDDSD			V
Low-level output voltage at I_O = 2.0 mA	V_{OL}			0.1xVDDSD	V
Input leakage current (except all pins with internal pull-up)	I_{in}			10	µA
Input leakage current (all pins with internal pull-up)				50	µA

1.5 Power Supply Characteristics

(T_A = 25 °C; VDD, PLLVDD = 2.5 V; VDDSD = 3.3 V;measurements performed under operating conditions)

Parameter	Symbol	Min	Typ	Max	Unit
Power supply current: Core and I/O operating: VSS			400		mA
PLL operating: PLLVSS			6		mA
Memory operating: VSSSD			25		mA

1.6 Switching Characteristics— RESET

(T_A = 25 °C; VDD, PLLVDD= 2.5 V; VDDSD = 3.3 V; C_L = 20 pF)

Parameter	Symbol	Min	Max	Unit
RESET minimum pulse width low	T_{rstl}	10	-	µs
All bidirectional pins high-Z after RESET low	T_{rst2z}		50	ns
Configuration bits setup before RESET high	T_{rstsu}	50	-	ns
Configuration bits hold after RESET high	T_{rsthld}	15	-	ns
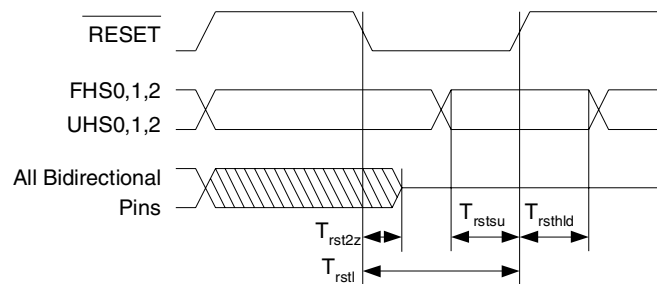
Figure 1. RESET Timing

1.7 Switching Characteristics — CLKIN

(T_A = 25 °C; VDD, PLLVDD = 2.5; VDDSD = 3.3 V; C_L = 20 pF)

Parameter	Symbol	Min	Max	Unit
CLKIN period for internal DSP clock mode	T_{clki}	35	100	ns
CLKIN high time for internal DSP clock mode	T_{clkih}	18		ns
CLKIN low time for internal DSP clock mode	T_{clkil}	18		ns
External Crystal operating frequency	F_{xtal}	10	14	MHz
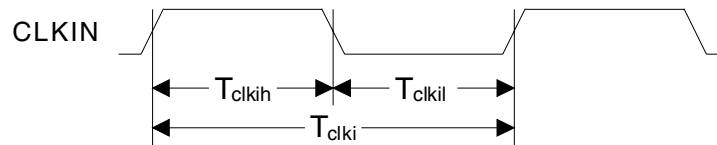
Figure 2. CLKIN with CLKSEL = VSS = PLL Enable

1.8 Switching Characteristics — Intel® Host Slave Mode (DSPAB)

(T_A = 25 °C; VDD, PLLVDD = 2.5 V; VDDSD = 3.3 V; C_L = 20 pF)

Parameter	Symbol	Min	Max	Unit
Address setup before $\overline{FCS}$ and $\overline{FRD}$ low or $\overline{FCS}$ and $\overline{FWR}$ low	T_{ias}	5	-	ns
Address hold time after $\overline{FCS}$ and $\overline{FRD}$ low or $\overline{FCS}$ and $\overline{FWR}$ high	T_{iah}	5	-	ns
Read				
Delay between $\overline{FRD}$ then $\overline{FCS}$ low or $\overline{FCS}$ then $\overline{FRD}$ low	T_{icdr}	0	-	ns
Data valid after $\overline{FCS}$ and $\overline{FRD}$ low	T_{idd}	-	21	ns
$\overline{FCS}$ and $\overline{FRD}$ low for read (Note 1)	T_{irpw}	DCLKP + 10	-	ns
Data hold time after $\overline{FCS}$ or $\overline{FRD}$ high	T_{idhr}	5	-	ns
Data high-Z after $\overline{FCS}$ or $\overline{FRD}$ high	T_{idis}	-	22	ns
$\overline{FCS}$ or $\overline{FRD}$ high to $\overline{FCS}$ and $\overline{FRD}$ low for next read (Note 1)	T_{ird}	2*DCLKP + 10	-	ns
$\overline{FCS}$ or $\overline{FRD}$ high to $\overline{FCS}$ and $\overline{FWR}$ low for next write (Note 1)	T_{irdtw}	2*DCLKP + 10	-	ns
Write				
Delay between $\overline{FWR}$ then $\overline{FCS}$ low or $\overline{FCS}$ then $\overline{FWR}$ low	T_{icdw}	0	-	ns
Data setup before $\overline{FCS}$ or $\overline{FWR}$ high	T_{idsu}	20	-	ns
$\overline{FCS}$ and $\overline{FWR}$ low for write (Note 1)	T_{iwpw}	DCLKP + 10	-	ns
Data hold after $\overline{FCS}$ or $\overline{FWR}$ high	T_{idhw}	5	-	ns
$\overline{FCS}$ or $\overline{FWR}$ high to $\overline{FCS}$ and $\overline{FRD}$ low for next read (Note 1)	T_{iwtrd}	2*DCLKP + 10	-	ns
$\overline{FCS}$ or $\overline{FWR}$ high to $\overline{FCS}$ and $\overline{FWR}$ low for next write (Note 1)	T_{iwd}	2*DCLKP + 10	-	ns

Notes: 1. Certain timing parameters are normalized to the DSP clock period, DCLKP. DCLKP = 1/DCLK. The DSP clock can be defined as follows:

Internal Clock Mode:
DCLK ~ 60MHz before and during boot, i.e. DCLKP ~ 16.6ns
DCLK ~ 86 MHz after boot, i.e. DCLKP ~ 11.6ns

It should be noted that DCLK for the internal clock mode is application specific. The application code users guide should be checked to confirm DCLK for the particular application.
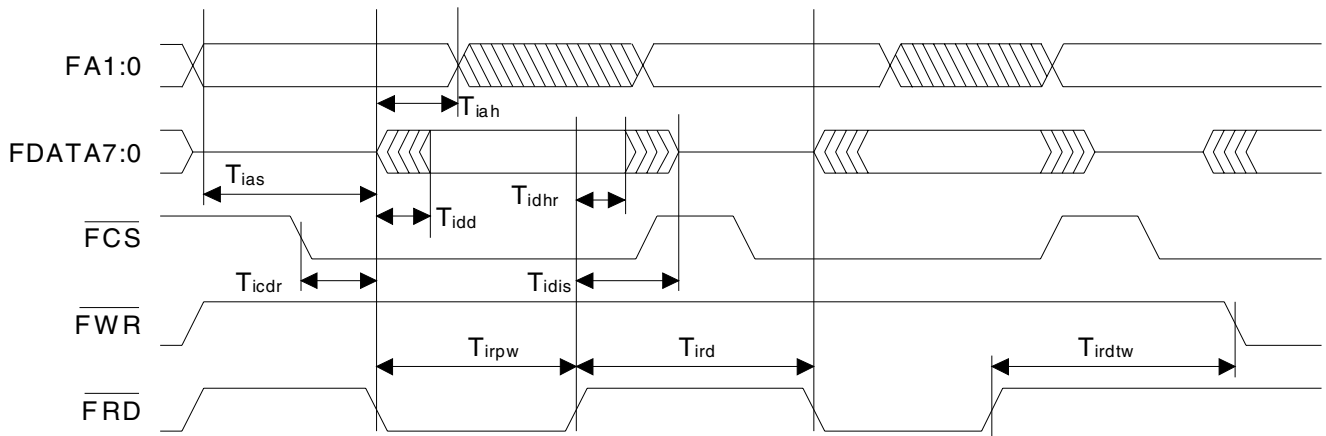
Figure 3. Intel® Parallel Host Mode Slave Read Cycle for DSPAB
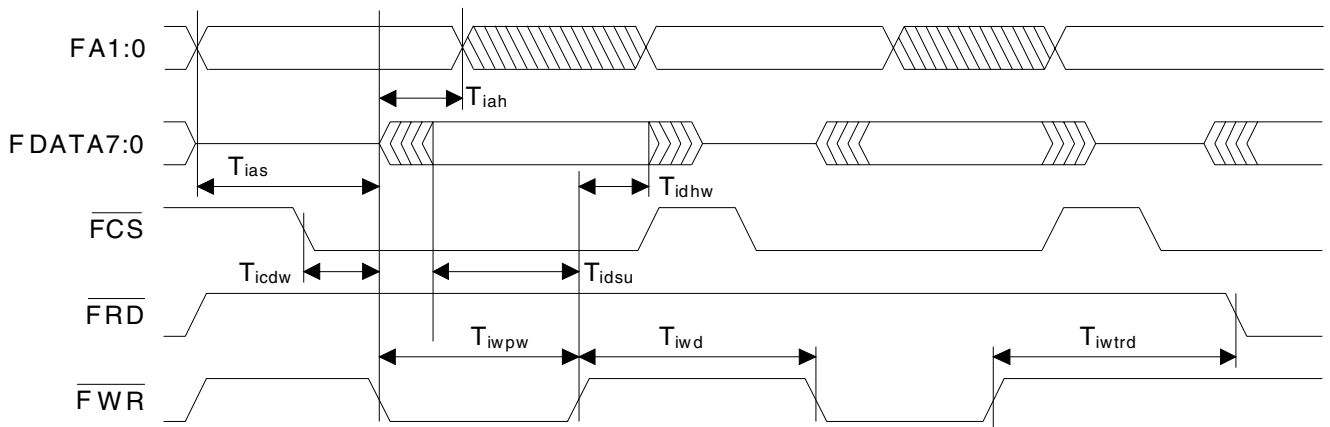
Figure 4. Intel® Parallel Host Mode Slave Write Cycle for DSPAB

1.9 Switching Characteristics — Intel® Host Slave Mode (DSPC)

(T_A = 25 °C; VDD, PLLVDD = 2.5 V; VDDSD = 3.3 V; C_L = 20 pF)

Parameter		Symbol	Min	Max	Unit
Address setup before $\overline{CS}$ and $\overline{RD}$ low or $\overline{CS}$ and $\overline{WR}$ low		T_{ias}	DCLKP	-	ns
Address hold time after $\overline{CS}$ and $\overline{RD}$ low or $\overline{CS}$ and $\overline{WR}$ low		T_{iah}	DCLKP+15	-	ns
Read					
Delay between $\overline{RD}$ then $\overline{CS}$ low or $\overline{CS}$ then $\overline{RD}$ low		T_{icdr}	0	-	ns
Data valid after $\overline{CS}$ and $\overline{RD}$ low		T_{idd}	-	2*DCLKP+ 25	ns
$\overline{CS}$ and $\overline{RD}$ low for read	(Note 1)	T_{irpw}	2*DCLKP	-	ns
Data hold time after $\overline{CS}$ or $\overline{RD}$ high		T_{idhr}	DCLKP+10	-	ns
Data high-Z after $\overline{CS}$ or $\overline{RD}$ high		T_{idis}	-	2*DCLKP+ 10	ns
$\overline{CS}$ or $\overline{RD}$ high to $\overline{CS}$ and $\overline{RD}$ low for next read	(Note 1)	T_{ird}	2*DCLKP+10	-	ns
$\overline{CS}$ or $\overline{RD}$ high to $\overline{CS}$ and $\overline{WR}$ low for next write	(Note 1)	T_{irdtw}	2*DCLKP+10	-	ns
Write					
Delay between $\overline{WR}$ then $\overline{CS}$ low or $\overline{CS}$ then $\overline{WR}$ low		T_{icdw}	0	-	ns
Data setup before $\overline{CS}$ or $\overline{WR}$ high		T_{idsu}	2*DCLKP+10	-	ns
$\overline{CS}$ and $\overline{WR}$ low for write	(Note 1)	T_{iwpw}	2*DCLKP	-	ns
Data hold after $\overline{CS}$ or $\overline{WR}$ high		T_{idhw}	DCLKP	-	ns
$\overline{CS}$ or $\overline{WR}$ high to $\overline{CS}$ and $\overline{RD}$ low for next read	(Note 1)	T_{iwtrd}	2*DCLKP+10	-	ns
$\overline{CS}$ or $\overline{WR}$ high to $\overline{CS}$ and $\overline{WR}$ low for next write	(Note 1)	T_{iwd}	2*DCLKP+10	-	ns

Notes: 1. Certain timing parameters are normalized to the DSP clock, DCLKP, in nanoseconds. DCLKP = 1/DCLK. The DSP clock can be defined as follows:

Internal Clock Mode:
DCLK ~ 60MHz before and during boot, i.e. DCLKP ~ 16.6ns
DCLK ~ 86 MHz after boot, i.e. DCLKP ~ 11.6ns

It should be noted that DCLK for the internal clock mode is application specific. The application code users guide should be checked to confirm DCLK for the particular application.

Figure 5. Intel® Parallel Host Slave Mode Read Cycle for DSPC
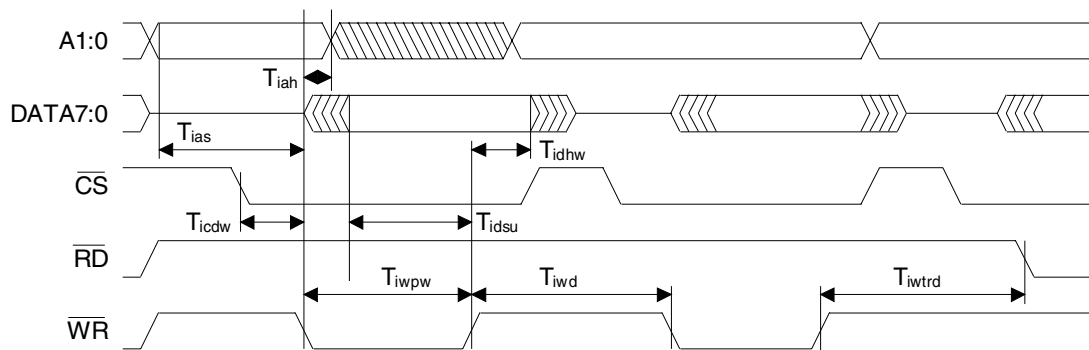
Figure 6. Intel® Parallel Host Slave Mode Write Cycle for DSPC

1.10 Switching Characteristics — Motorola® Host Slave Mode (DSPAB)

(T_A = 25 °C; VDD, PLLVDD = 2.5 V; VDDSD = 3.3 V; C_L = 20 pF)

Parameter	Symbol	Min	Max	Unit
Address setup before $\overline{FCS}$ and $\overline{FDS}$ low	T_{mas}	5	-	ns
Address hold time after $\overline{FCS}$ and $\overline{FDS}$ low	T_{mah}	5	-	ns
Read				
Delay between $\overline{FDS}$ then $\overline{FCS}$ low or $\overline{FCS}$ then $\overline{FDS}$ low	T_{mcdr}	0	-	ns
Data valid after $\overline{FCS}$ and $\overline{FRD}$ low with R/$\overline{W}$ high)	T_{mdd}	-	21	ns
$\overline{FCS}$ and $\overline{FDS}$ low for read (Note 1)	T_{mrpw}	DCLKP + 10	-	ns
Data hold time after $\overline{FCS}$ or $\overline{FDS}$ high after read	T_{mdhr}	5	-	ns
Data high-Z after $\overline{FCS}$ or $\overline{FDS}$ high after read	T_{mdis}	-	22	ns
$\overline{FCS}$ or $\overline{FDS}$ high to $\overline{FCS}$ and $\overline{FDS}$ low for next read (Note 1)	T_{mrd}	2*DCLKP + 10	-	ns
$\overline{FCS}$ or $\overline{FDS}$ high to $\overline{FCS}$ and $\overline{FDS}$ low for next write (Note 1)	T_{mrdtw}	2*DCLKP + 10	-	ns
Write				
Delay between $\overline{FDS}$ then $\overline{FCS}$ low or $\overline{FCS}$ then $\overline{FDS}$ low	T_{mcdw}	0	-	ns
Data setup before $\overline{FCS}$ or $\overline{FDS}$ high	T_{mdsu}	20	-	ns
$\overline{FCS}$ and $\overline{FDS}$ low for write (Note 1)	T_{mwpw}	DCLKP + 10	-	ns
R/$\overline{W}$ setup before $\overline{FCS}$ AND $\overline{FDS}$ low	T_{mrwsu}	5	-	ns
R/$\overline{W}$ hold time after $\overline{FCS}$ or $\overline{FDS}$ high	T_{mrwhld}	5	-	ns
Data hold after $\overline{FCS}$ or $\overline{FDS}$ high	T_{mdhw}	5	-	ns
$\overline{FCS}$ or $\overline{FDS}$ high to $\overline{FCS}$ and $\overline{FDS}$ low with R/$\overline{W}$ high for next read (Note 1)	T_{mwtrd}	2*DCLKP + 10	-	ns
$\overline{FCS}$ or $\overline{FDS}$ high to $\overline{FCS}$ and $\overline{FDS}$ low for next write (Note 1)	T_{mwd}	2*DCLKP + 10	-	ns

Notes: 1. Certain timing parameters are normalized to the DSP clock, DCLKP, in nanoseconds. DCLKP = 1/DCLK. The DSP clock can be defined as follows:

Internal Clock Mode:
DCLK ~ 60MHz before and during boot, i.e. DCLKP ~ 16.6ns
DCLK ~ 86 MHz after boot, i.e. DCLKP ~ 11.6ns

It should be noted that DCLK for the internal clock mode is application specific. The application code users guide should be checked to confirm DCLK for the particular application.

Figure 7. Motorola® Parallel Host Slave Mode Read Cycle for DSPAB
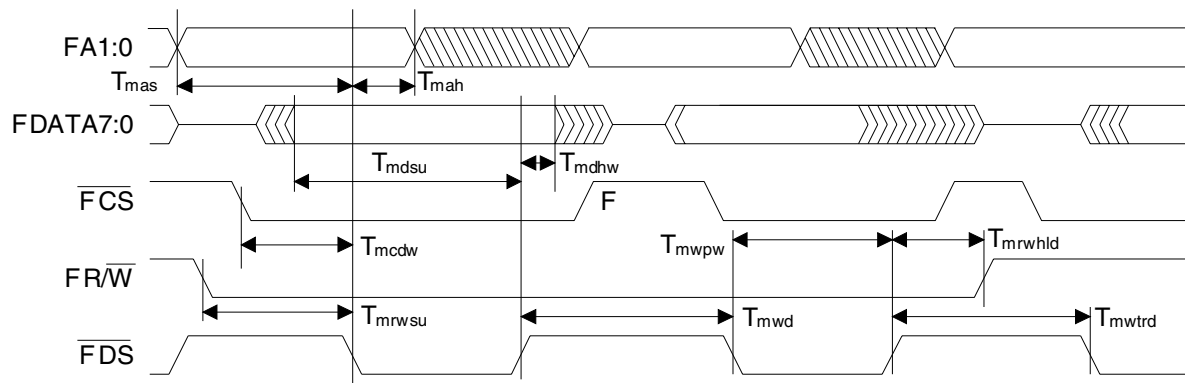
Figure 8. Motorola® Parallel Host Slave Mode Write Cycle for DSPAB

1.11 Switching Characteristics — Motorola® Host Slave Mode (DSPC)

(T_A = 25 °C; VDD, PLLVDD = 2.5 V; VDDSD = 3.3 V; C_L = 20 pF)

Parameter	Symbol	Min	Max	Unit
Address setup before $\overline{CS}$ and $\overline{DS}$ low	T_{mas}	DCLKP	-	ns
Address hold time after $\overline{CS}$ and $\overline{DS}$ low	T_{mah}	DCLKP+15	-	ns
Read				
Delay between $\overline{DS}$ then $\overline{CS}$ low or $\overline{CS}$ then $\overline{DS}$ low	T_{mcdr}	0	-	ns
Data valid after $\overline{CS}$ and $\overline{RD}$ low with R/$\overline{W}$ high	T_{mdd}	-	2*DCLKP+ 25	ns
$\overline{CS}$ and $\overline{DS}$ low for read (Note 1)	T_{mrpw}	2*DCLKP	-	ns
Data hold time after $\overline{CS}$ or $\overline{DS}$ high after read	T_{mdhr}	DCLKP+ 10	-	ns
Data high-Z after $\overline{CS}$ or $\overline{DS}$ high low after read	T_{mdis}	-	2*DCLKP+ 10	ns
$\overline{CS}$ or $\overline{DS}$ high to $\overline{CS}$ and $\overline{DS}$ low for next read (Note 1)	T_{mrd}	2*DCLKP+10	-	ns
$\overline{CS}$ or $\overline{DS}$ high to $\overline{CS}$ and $\overline{DS}$ low for next write (Note 1)	T_{mrdtw}	2*DCLKP+10	-	ns
Write				
Delay between $\overline{DS}$ then $\overline{CS}$ low or $\overline{CS}$ then $\overline{DS}$ low	T_{mcdw}	0	-	ns
Data setup before $\overline{CS}$ or $\overline{DS}$ high	T_{mdsu}	2*DCLKP+10	-	ns
$\overline{CS}$ and $\overline{DS}$ low for write (Note 1)	T_{mwpw}	2*DCLKP	-	ns
R/$\overline{W}$ setup before $\overline{CS}$ AND $\overline{DS}$ low	T_{mrwsu}	DCLKP	-	ns
R/$\overline{W}$ hold time after $\overline{CS}$ or $\overline{DS}$ high	T_{mrwhld}	5	-	ns
Data hold after $\overline{CS}$ or $\overline{DS}$ high	T_{mdhw}	DCLKP	-	ns
$\overline{CS}$ or $\overline{DS}$ high to $\overline{CS}$ and $\overline{DS}$ low with R/$\overline{W}$ high for next read (Note 1)	T_{mwtrd}	2*DCLKP+10	-	ns
$\overline{CS}$ or $\overline{DS}$ high to $\overline{CS}$ and $\overline{DS}$ low for next write (Note 1)	T_{mwd}	2*DCLKP+10	-	ns

Notes: 1. Certain timing parameters are normalized to the DSP clock, DCLKP, in nanoseconds. DCLKP = 1/DCLK. The DSP clock can be defined as follows:

Internal Clock Mode:
DCLK ~ 60MHz before and during boot, i.e. DCLKP ~ 16.6ns
DCLK ~ 86 MHz after boot, i.e. DCLKP ~ 11.6ns

It should be noted that DCLK for the internal clock mode is application specific. The application code users guide should be checked to confirm DCLK for the particular application.

Figure 9. Motorola® Parallel Host Slave Mode Read Cycle for DSPC
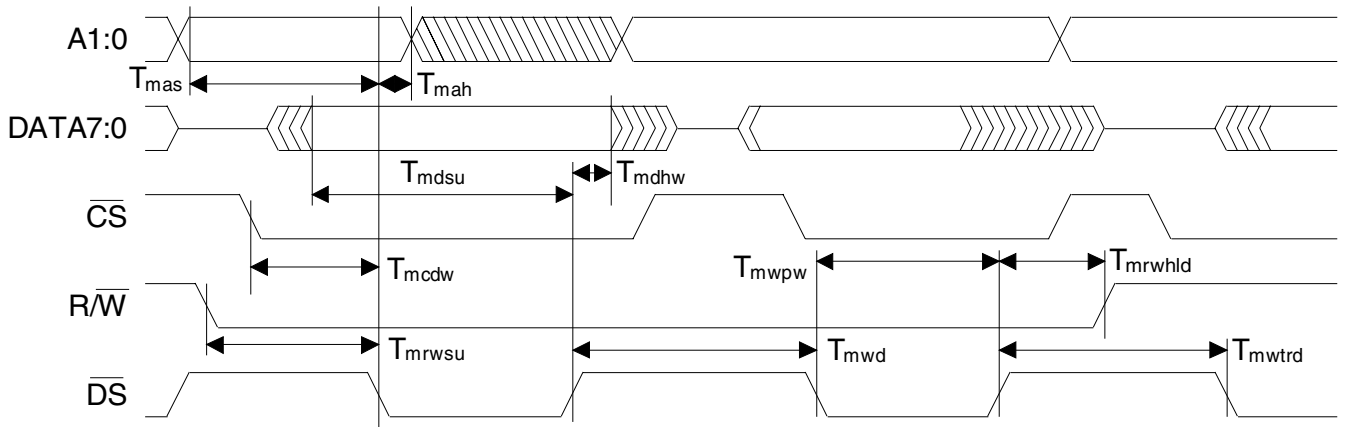
Figure 10. Motorola® Parallel Host Slave Mode Write Cycle for DSPC

1.12 Switching Characteristics — SPI Control Port Slave Mode (DSPAB)

(T_A = 25 °C; VDD, PLLVDD = 2.5 V; VDDSD = 3.3 V; C_L = 20 pF)

Parameter		Symbol	Min	Max	Units
FSCCLK clock frequency	(Note 1)	f_{sck}	-	2	MHz
FCS falling to FSCCLK rising		t_{css}	20	-	ns
FSCCLK low time		t_{scl}	150	-	ns
FSCCLK high time		t_{sch}	150	-	ns
Setup time FSCDIN to FSCCLK rising		t_{cdisu}	50	-	ns
Hold time FSCCLK rising to FSCDIN	(Note 2)	t_{cdih}	50	-	ns
Transition time from FSCCLK to FSCDOUT valid		t_{scdov}	-	40	ns
Time from FSCCLK rising to FINTREQ rising	(Note 3)	t_{scrh}	-	200	ns
Hold time for FINTREQ from FSCCLK rising	(Note 4, 5)	t_{scrl}	0	-	ns
Time from FSCCLK falling to FCS rising		t_{sccsh}	20	-	ns
High time between active FCS		t_{csht}	200	-	ns
Time from FCS rising to FSCDOUT high-Z		t_{cscdo}		20	ns

Notes:
1. The specification f_{sck} indicates the maximum speed of the hardware. The system designer should be aware that the actual maximum speed of the communication port may be limited by the DSP application code. The relevant application code user's manual should be consulted for the software speed limitations.

2. Data must be held for sufficient time to bridge the transition time of FSCCLK.

3. FINTREQ goes high only if there is no data to be read from the DSP at the rising edge of FSCCLK for the second-to-last bit of the last byte of data during a read operation as shown.

4. If FINTREQ goes high as indicated in (Note 3), then FINTREQ is guaranteed to remain high until the next rising edge of FSCCLK. If there is more data to be read at this time, FINTREQ goes active low again. Treat this condition as a new read transaction. Raise chip select to end the current read transaction and then drop it, followed by the 7-bit address and the R/$\overline{W}$ bit (set to 1 for a read) to start a new read transaction.
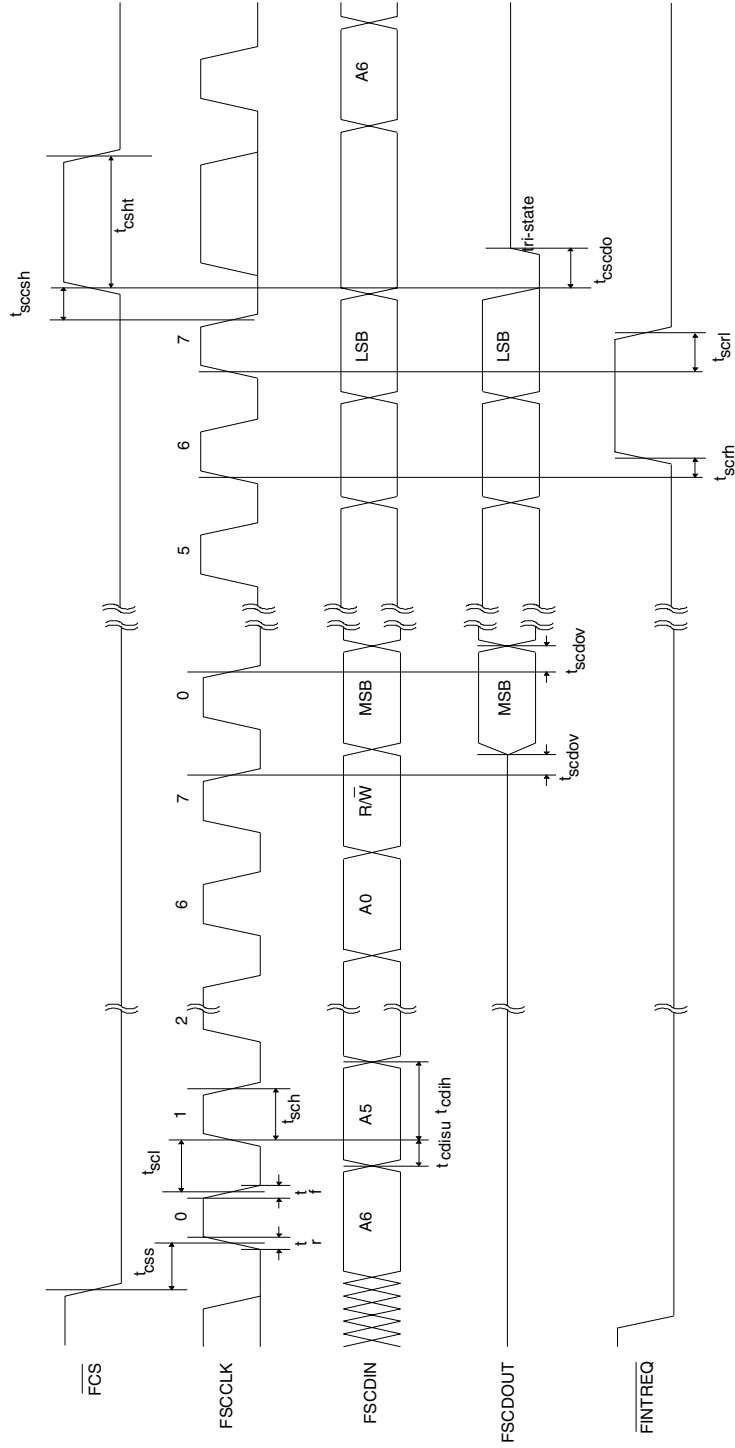
Figure 11. SPI Control Port Slave Mode Timing (DSPAB)

1.13 Switching Characteristics — SPI Control Port Slave Mode (DSPC)

(T_A = 25 °C; VDD, PLLVDD = 2.5 V; VDDSD = 3.3 V; C_L = 20 pF)

Parameter		Symbol	Min	Max	Units
SCCLK clock frequency	(Note 1)	f_{sck}	-	5	MHz
CS falling to SCCLK rising		t_{css}	4*DCLKP	-	ns
SCCLK low time		t_{scl}	4*DCLKP	-	ns
SCCLK high time		t_{sch}	4*DCLKP	-	ns
Setup time SCDIN to SCCLK rising		t_{cdisu}	DCLKP	-	ns
Hold time SCCLK rising to SCDIN	(Note 2)	t_{cdih}	DCLKP+20	-	ns
Time from SCCLK low to SCDOUT valid		t_{scdov}	-	3*DCLKP+20	ns
Time from SCCLK rising to INTREQ rising		t_{scrh}	-	DCLKP	ns
Hold time for INTREQ from SCCLK rising		t_{scrl}	DCLKP	-	ns
Time from SCCLK falling to CS rising		t_{sccsh}	2*DCLKP+15	-	ns
Time from SCCLK low to CS falling		t_{sccsl}	10		ns
High time between active CS		t_{csht}	4*DCLKP	-	ns
Time from CS rising to SCDOUT high-Z		t_{cscdo}		DCLKP	ns

Notes: 1. The specification f_{sck} indicates the maximum speed of the hardware. The system designer should be aware that the actual maximum speed of the communication port may be limited by the software. The relevant application code user's manual should be consulted for the software speed limitations.

2. Data must be held for sufficient time to bridge the transition time of SCCLK.
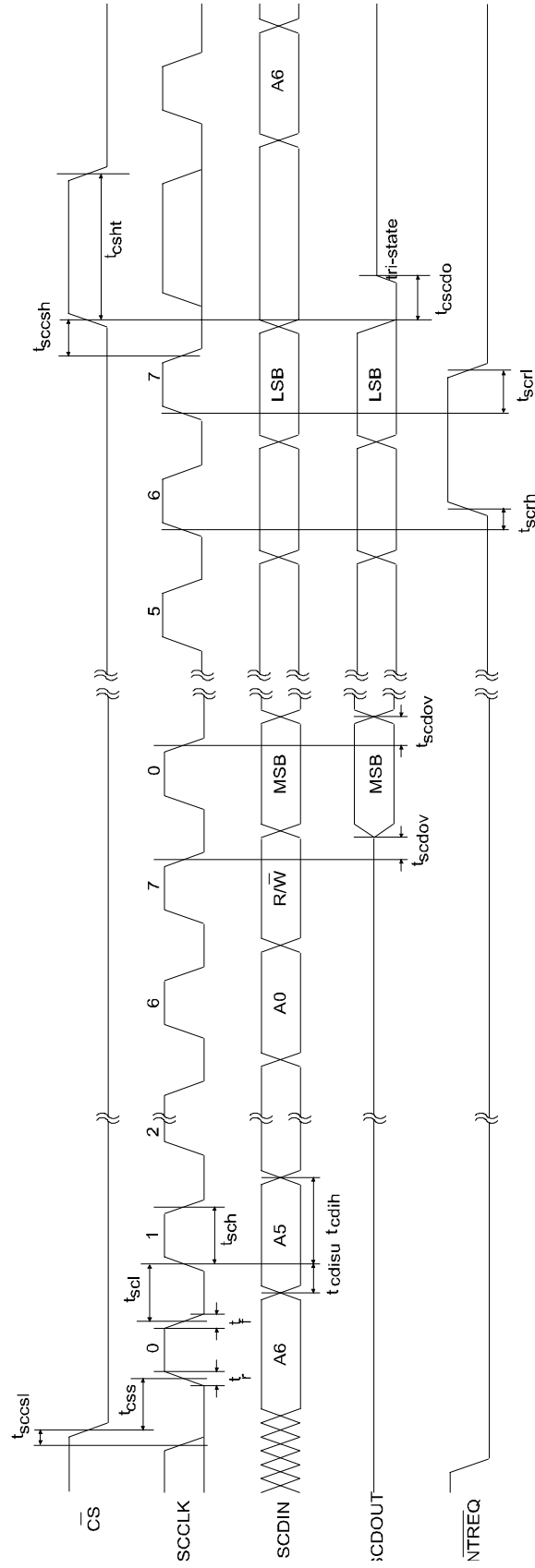
Figure 12. SPI Control Port Slave Mode Timing (DSPC)

1.14 Switching Characteristics — Digital Audio Input (DSPAB)

(T_A = 25 °C; VDD, PLLVDD = 2.5 V; VDDSD = 3.3 V; C_L = 20 pF)

Parameter		Symbol	Min	Max	Unit
FSCLKN1 period for Slave mode		T_{sclki}	40	-	ns
FSCLKN1 duty cycle for Slave mode			45	55	%
Slave Mode	(Note 2)				
Time from active edge of FSCLKN1(2) to FLRCLKN1(2) transition		T_{stlr}	10	-	ns
Time from FLRCLKN1(2) transition to FSCLKN1(2) active edge		T_{lrts}	10	-	ns
FSDATAN1(2) setup to FSCLKN1(2) transition	(Note 1)	T_{sdsus}	5	-	ns
FSDATAN1(2) hold time after FSCLKN1(2) transition	(Note 1)	T_{sdhs}	5	-	ns

Notes: 1. This timing parameter is defined from the active edge of FSCLKN1/2. The active edge of FSCLKN1/2 is the point at which the data is valid.

2. Slave mode is defined as FSCLKN1/2 and FLRCLKN1/2 driven by an external source.
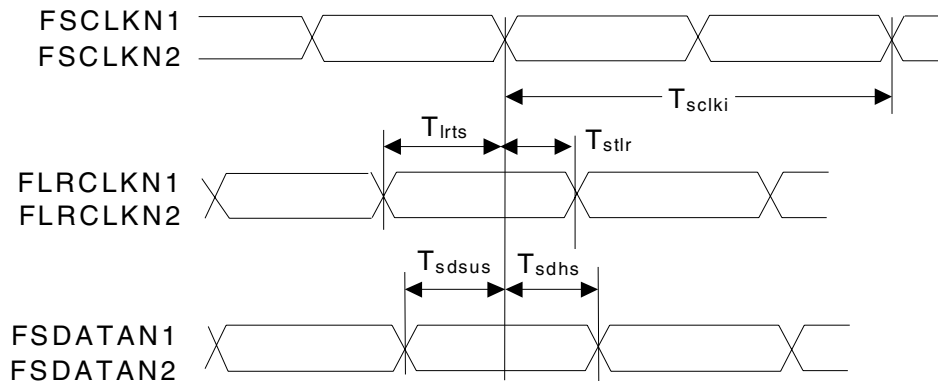
Figure 13. Digital Audio Input Data, Slave Clock Timing

1.15 Switching Characteristics — Serial Audio Input (DSPC)

(T_A = 25 °C; VDD, PLLVDD = 2.5 V; VDDSD = 3.3 V; C_L = 20 pF)

Parameter		Symbol	Min	Max	Unit
Slave Mode					
SCLKN period for Slave mode		T_{sclki}	40	-	ns
SCLKN duty cycle for Slave mode			45	55	%
Time from active edge of SCLKN to LRCLKN transition		T_{stlr}	20	-	ns
Time from LRCLKN transition to SCLKN active edge		T_{lrts}	20	-	ns
SDATAN0 setup to SCLKN transition	(Notes 2)	T_{sdsus}	10	-	ns
SDATAN0 hold time after SCLKN transition	(Notes 2)	T_{sdhs}	10	-	ns

Notes: 1. Slave mode is defined as SCLKN and LRCLKN being driven by an external source.

2. This timing parameter is defined from the active edge of SCLKN. The active edge of SCLKN is the point at which the data is valid.
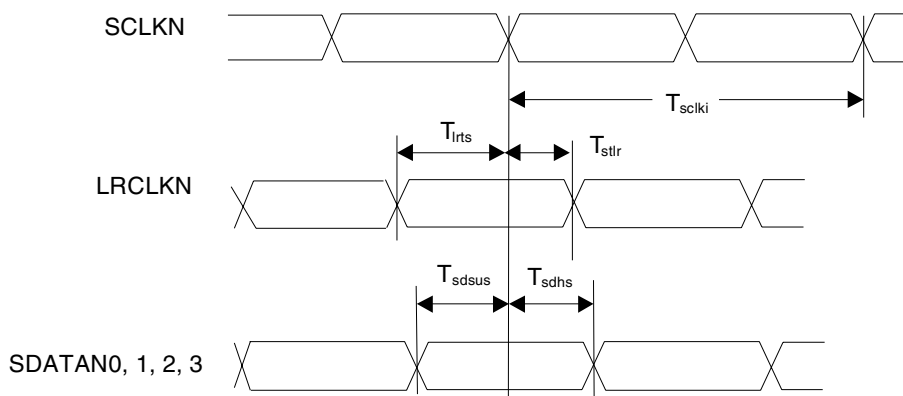
Figure 14. Serial Audio Input Data, Slave Clock Timing

1.16 Switching Characteristics — CMPDAT, CMPCLK (DSPAB)

(T_A = 25 °C; VDD, PLLVDD = 2.5 V; VDDSD = 3.3 V; C_L = 20 pF)

Parameter	Symbol	Min	Max	Unit
Serial compressed data clock CMPCLK frequency	T_{cmpclk}	-	27	MHz
CMPDAT setup before CMPCLK high	T_{cmpsu}	10	-	ns
CMPDAT hold after CMPCLK high	T_{cmphld}	10	-	ns
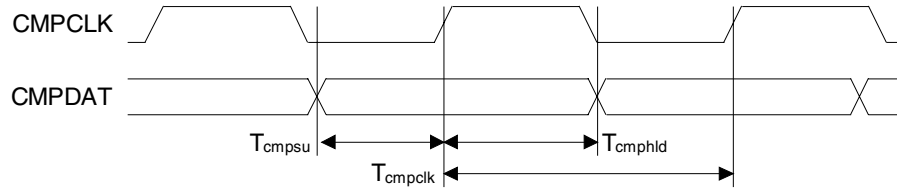
Figure 15. Serial Compressed Data Timing

1.17 Switching Characteristics — Parallel Data Input (DSPAB)

$(T_A = 25\ °C;\ VDD,\ PLLVDD = 2.5\ V;\ VDDSD = 3.3\ V;\ C_L = 20\ pF)$

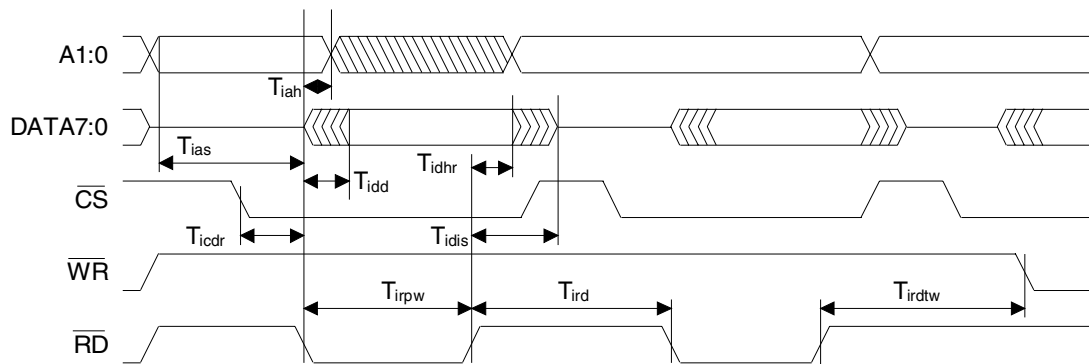| Parameter	Symbol	Min	Max	Unit
CMPCLK Period	T_{cmpclk}	4*DCLKP + 10		ns
FDAT[7:0] setup before CMPCLK high	T_{cmpsu}	10		ns
FDAT[7:0] hold after CMPCLK high	T_{cmphld}	10		ns

Notes: 1. Certain timing parameters are normalized to the DSP clock, DCLK, in nanoseconds. The DSP clock can be defined as follows:

Internal Clock Mode:
DCLK ~ 60MHz before and during boot, i.e. DCLKP ~ 16.6ns
DCLK ~ 86 MHz after boot, i.e. DCLKP ~ 11.6ns

It should be noted that DCLK for the internal clock mode is application specific. The application code users guide should be checked to confirm DCLK for the particular application.
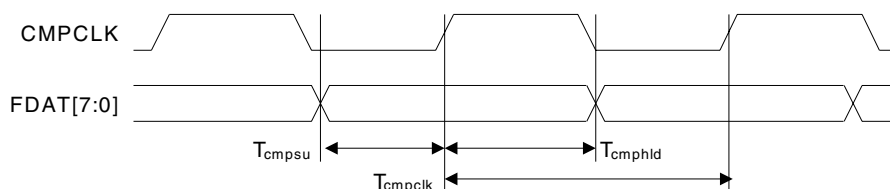
Figure 16. Parallel Data Timing

1.18 Switching Characteristics — Digital Audio Output

(T_A = 25 °C; VDD, PLLVDD = 2.5 V; VDDSD = 3.3 V; C_L = 20 pF)

Parameter	Symbol	Min	Max	Unit
MCLK period	T_{mclk}	40	-	ns
MCLK duty cycle		40	60	%
SCLK0, SCLK1 period for Master or Slave mode (Note 2)	T_{sclk}	40	-	ns
SCLK0, SCLK1 duty cycle for Master or Slave mode (Note 2)		45	55	%
Master Mode (Output A1 Mode) (Note 2, 3)				
SCLK0, SCLK1 delay from MCLK rising edge, MCLK as an input	T_{sdmi}		15	ns
LRCLK0, LRCLK1 delay from SCLK0, SCLK1 transition, respectively (Note 4)	T_{lrds}		10	ns
AUDATA7–0 delay from SCLK0, SCLK1 transition (Note 4)	T_{adsm}		10	ns
Slave Mode (Output A0 Mode) (Note 5)				
Time from active edge of SCLK0, SCLK1 to LRCLK0, LRCLK1 transition	T_{stlr}	10	-	ns
Time from LRCLK0, LRCLK1 transition to SCLK0, SCLK1 active edge	T_{lrts}	10	-	ns
AUDATA7–0 delay from SCLK0, SCLK1 transition (Note 4)	T_{adss}		15	ns

Notes: 1. DSPC has two Digital Audio Output modules having analogous signal names ending in 0 and 1. Both DAO ports share a common MCLK but have independent SCLKs and LRCLKs.

2. Master mode timing specifications are characterized, not production tested.

3. Master mode is defined as the CS49400 driving both SCLK0, SCLK1, LRCLK0, and LRCLK1. When MCLK is an input, it is divided to produce SCLK0, SCLK1, LRCLK0 and LRCLK1.

4. This timing parameter is defined from the non-active edge of SCLK0 and SCLK1. The active edge of SCLK0 and SCLK1 is the point at which the data is valid.

5. Slave mode is defined as SCLK0, SCLK1, LRCLK0 and LRCLK1 driven by an external source.

Master Mode (Output A1) Output Clock Timing and Digital Audio Output Data
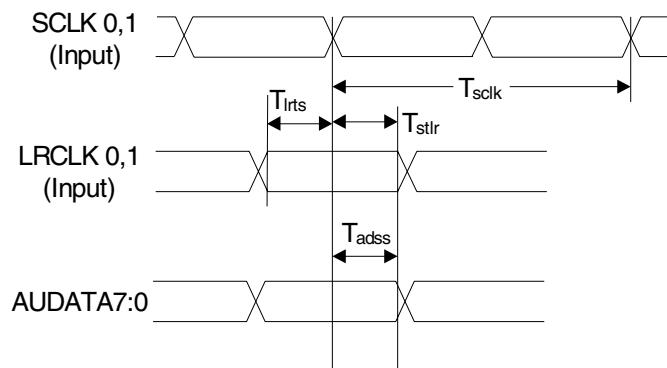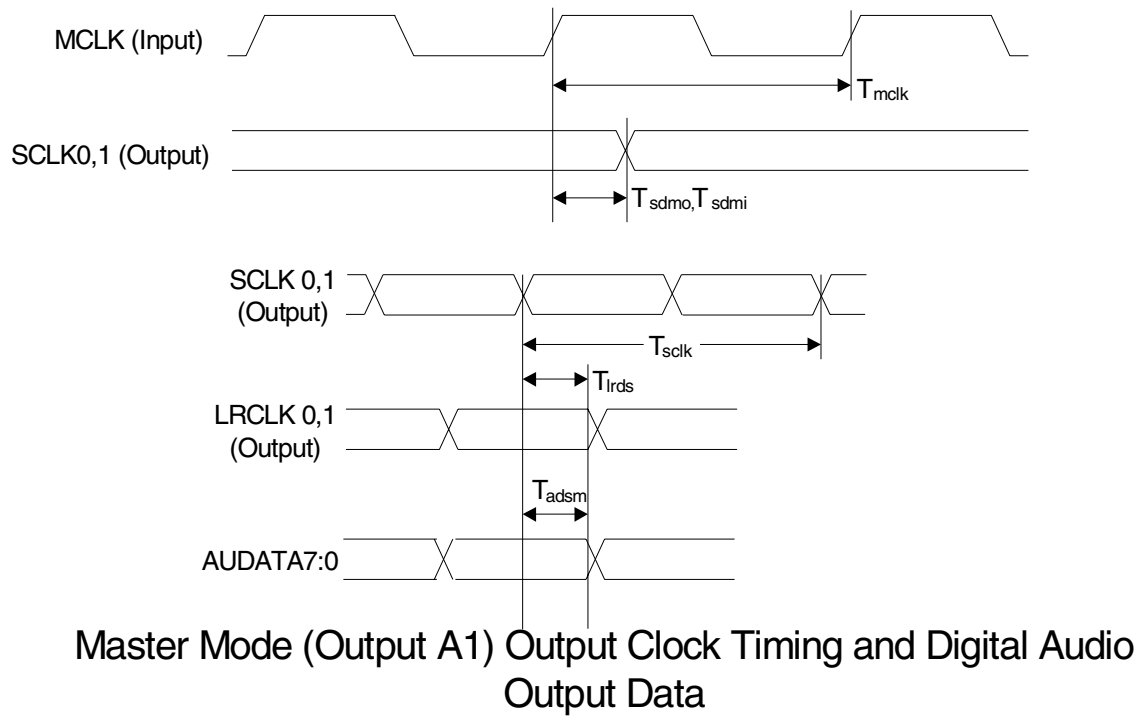
Slave Mode (Output A0) Output Clock Timing and Digital Audio Output Data

Figure 18. Digital Audio Output Data, Input and Output Clock Timing

1.19 Switching Characteristics — SRAM/FLASH Interface

(T_A = 25 °C; VDD, PLLVDD = 2.5 V; VDDSD = 3.3 V; C_L = 20 pF)

Parameter	Symbol	Min	Max	Unit
Write Cycle				
Single Byte Write Cycle	T_{wrc}	(SRAM_FLASH_WR_CYCLE + 1) * DCLKP	-	ns
Data Hold after $\overline{NV\_WE}$ or $\overline{NV\_CS}$ high	T_{dh}	DCLKP-5	-	ns
Data Valid after $\overline{NV\_CS}$ and $\overline{NV\_WE}$ low	T_{dv}		10	ns
Data Strobe	T_{ds}	DCLKP-5	-	ns
Read Cycle				
Single Byte Read Cycle	T_{rdc}	(SRAM_FLASH_RD_CYCLE + 1) * DCLKP	-	ns
Data Strobe	T_{ds}	DCLKP-5	-	ns
Data Hold after $\overline{NV\_WE}$ or $\overline{NV\_CS}$ high	T_{dh}	DCLKP+5	-	ns
Data Setup Time	T_{su}	DCLKP+5	-	ns
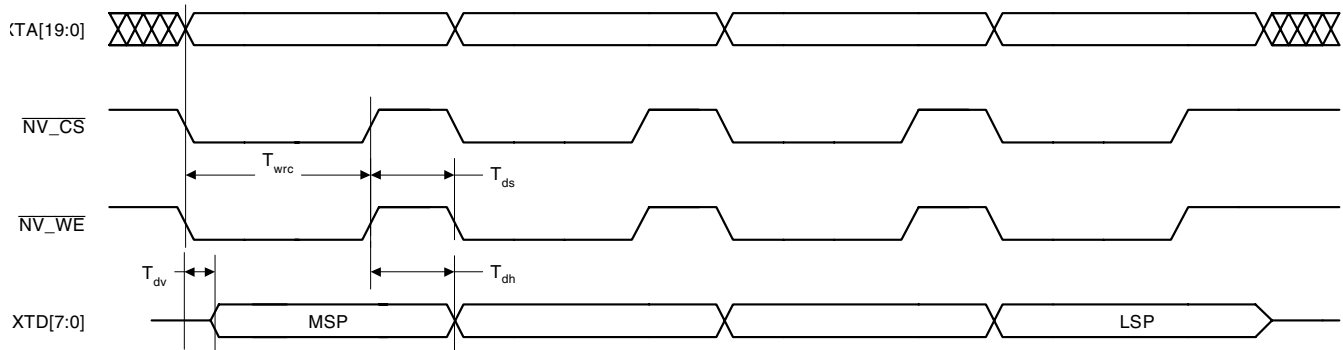
Figure 19. SRAM/Flash Controller Timing Diagram - Write Cycle
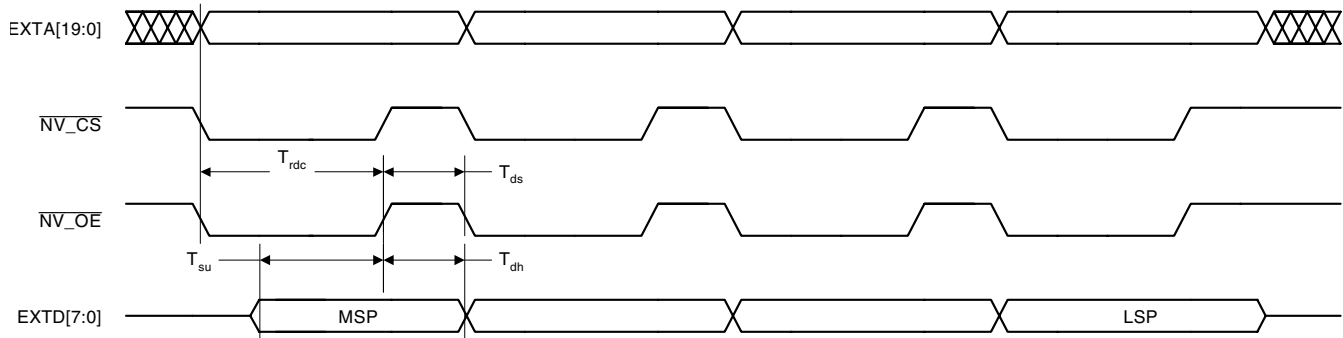
Figure 20. SRAM/Flash Controller Timing Diagram - Read Cycle

EXTA[19:0]　Valid

$\overline{NV\_CS}$

T_{wrc}　T_{ds}

$\overline{NV\_WE}$

$\overline{NV\_OE}$

T_{dv}　T_{dh}

EXTD[7:0]　LSP
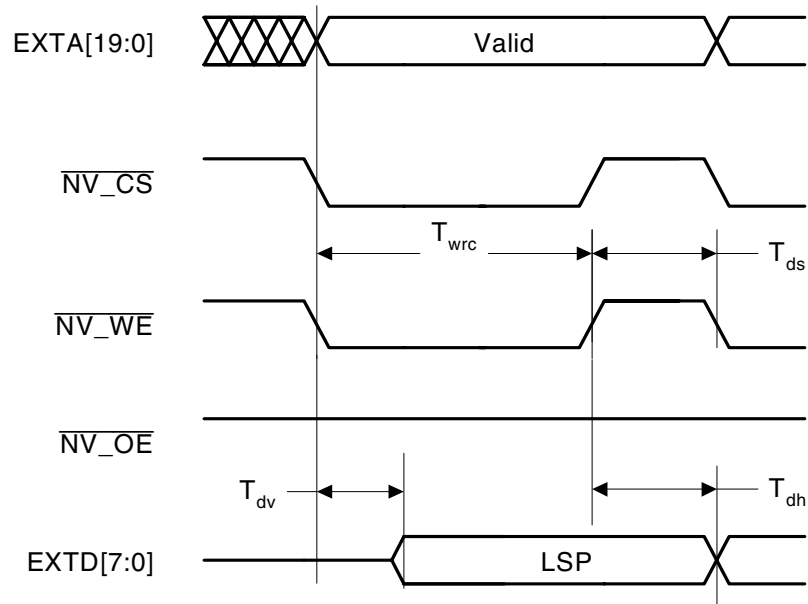
Figure 21. SRAM/Flash Controller Timing Diagram - Single Byte Write Cycle

EXTA[19:0]　Valid

$\overline{NV\_CS}$

T_{rdc}　T_{ds}

$\overline{NV\_OE}$

$\overline{NV\_WE}$

T_{su}　T_{dh}

EXTD[7:0]　LSP
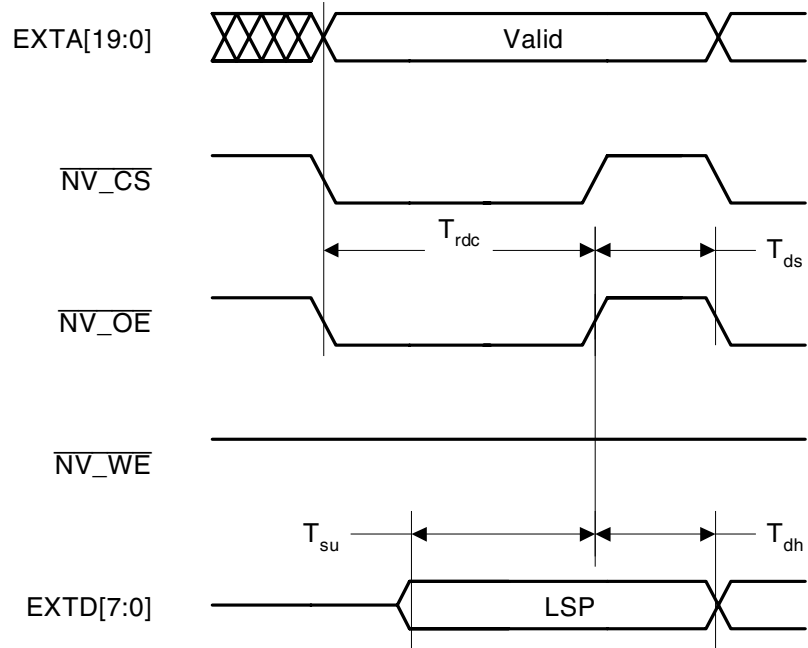
Figure 22. SRAM/Flash Controller Timing Diagram - Single Byte Read Cycle

1.20 Switching Characteristics — SDRAM Interface

(T_A = 25 °C; VDD, PLLVDD = 2.5 V; VDDSD = 3.3 V; C_L = 20 pF, SD_CLKOUT = SD_CLKIN)

Parameter	Symbol	Min	Max	Unit
SD_CLKIN high time	t_{clk_high}	0.475*DCLKP	-	ns
SD_CLKIN low time	t_{clk_low}	0.475*DCLKP	-	ns
SD_CLKOUT rise/fall time	t_{clkrf}	-	1	ns
SD_CLKOUT duty cycle	t_{clkrf}	45	55	%
SD_CLKOUT rising edge to signal valid	t_d	-	9.8	ns
Signal hold from SD_CLKOUT rising edge	t_h	1.0		ns
SD_CLKOUT rising edge to SD_DQMn valid	t_{DQd}	-	7.2	ns
SD_DQMn hold from SD_CLKOUT rising edge	t_{DQh}	1.0	-	ns
SD_DATA valid setup to SD_CLKIN rising edge	t_{DAs}		8.3	ns
SD_DATA valid hold to SD_CLKIN rising edge	t_{DAh}		1.0	ns
SD_CLKOUT rising edge to ADDRn valid	t_d	-	8.0	ns
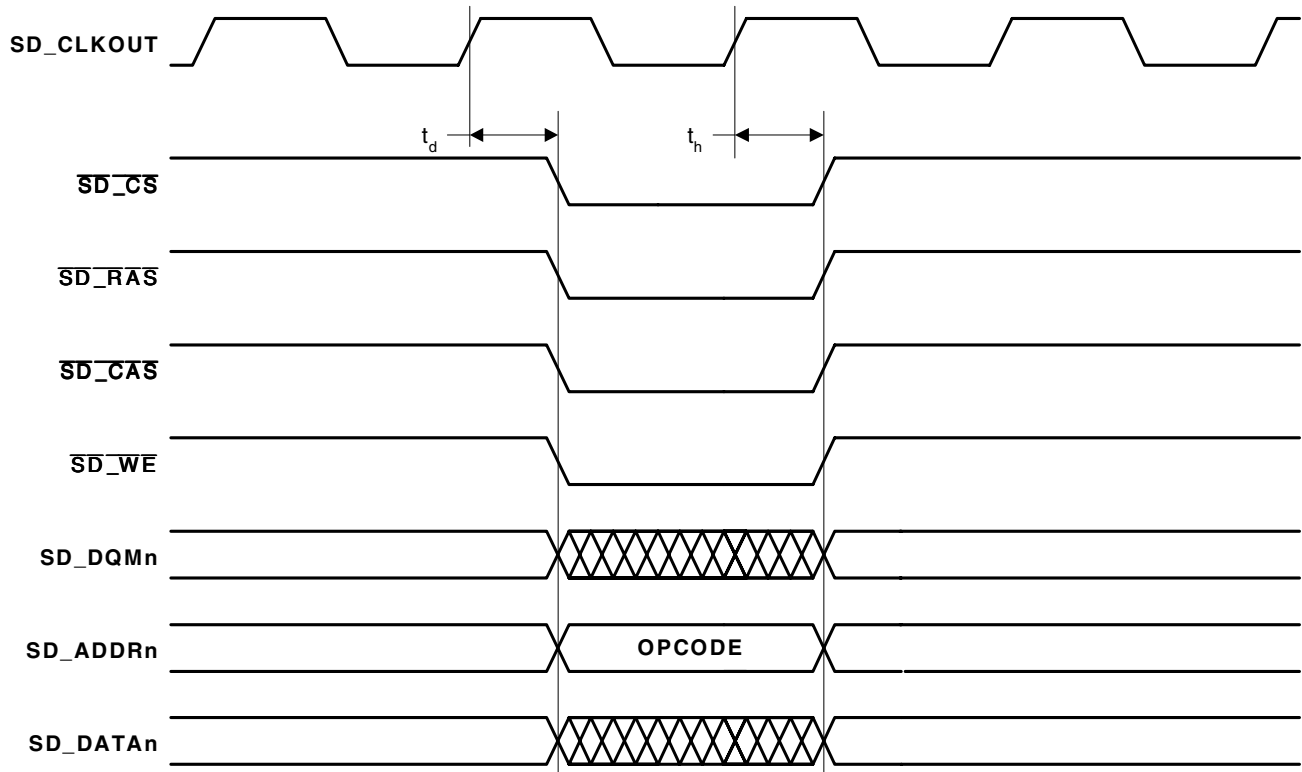
Figure 23. SDRAM Controller Timing Diagram - Load Mode Register Cycle
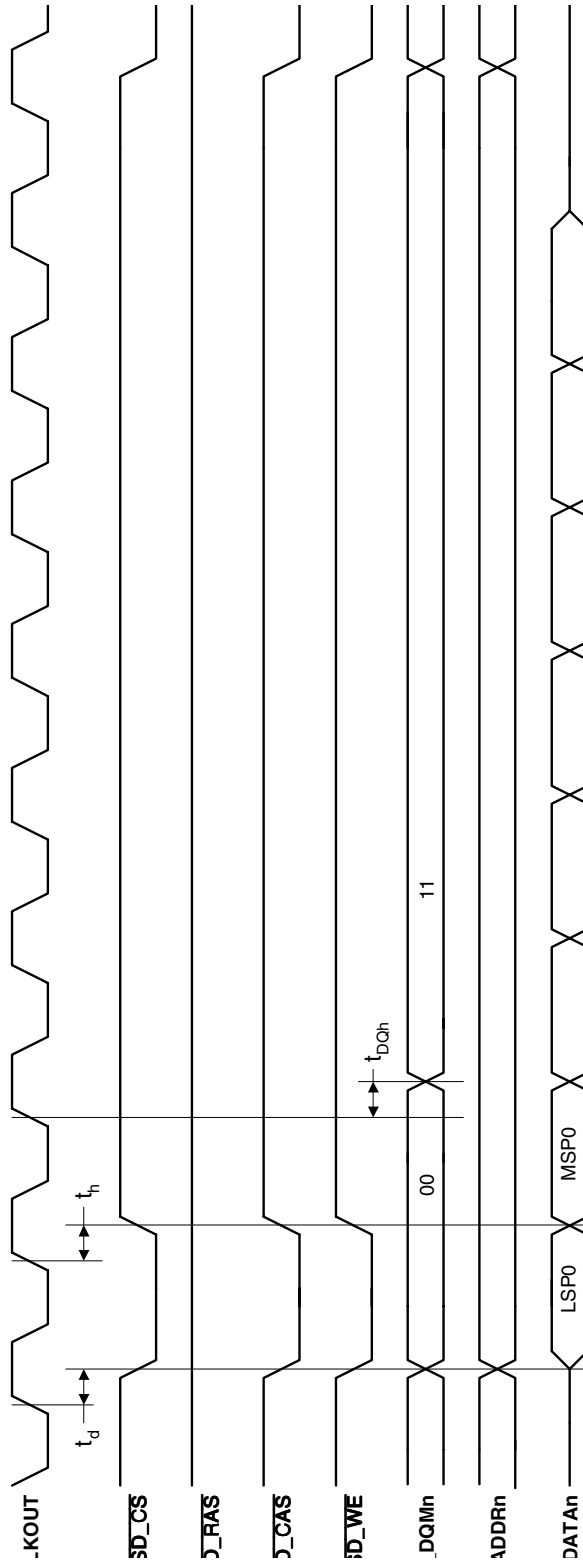
Figure 24. SDRAM Controller Timing Diagram - Burst Write Cycle
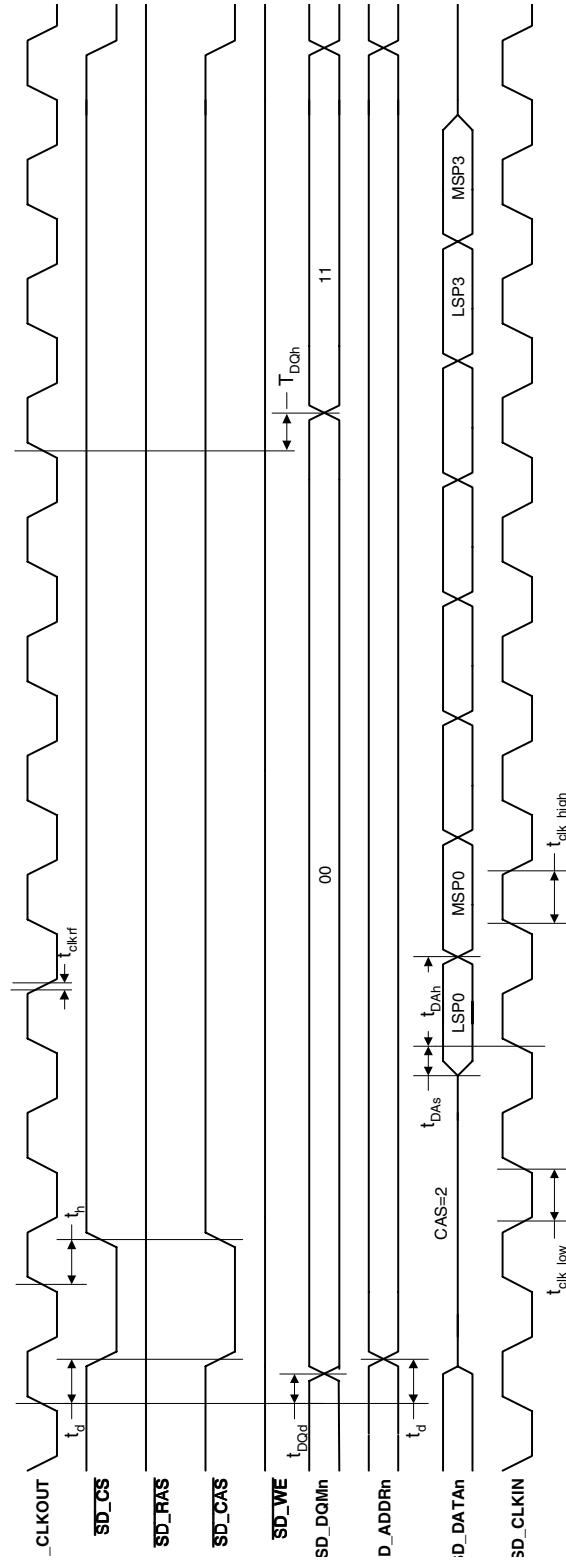
Figure 25. SDRAM Controller Timing Diagram - Burst Read Cycle
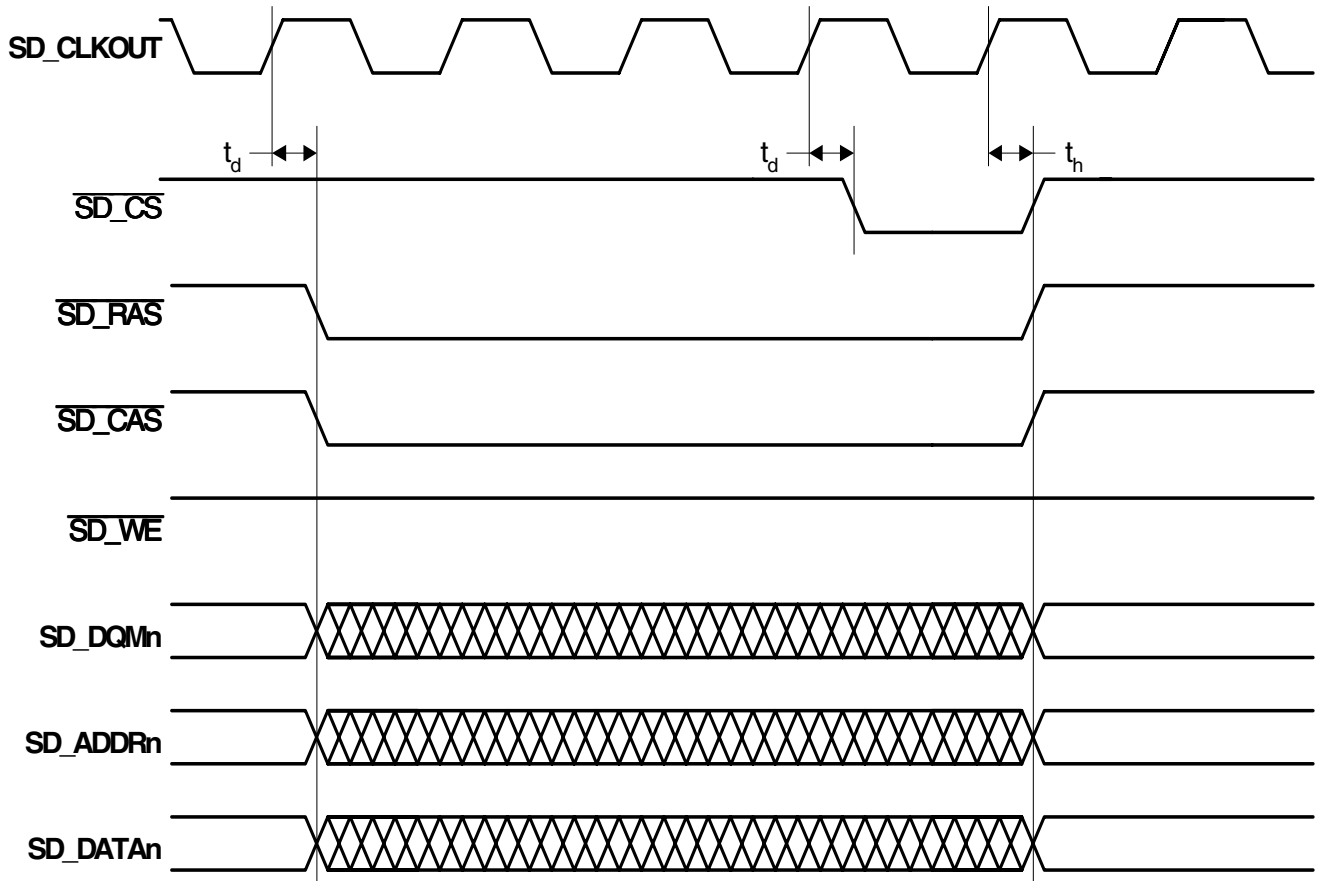
Figure 26. SDRAM Controller Timing Diagram - Auto Refresh Cycle

2. OVERVIEW

The CS49400 is a 24-bit fixed-point decoder DSP followed by a 32-bit fixed point programmable post-processor DSP. The decoder portion of the CS49400 is referred to as "DSPAB". The post-processor DSP is referred to as "DSPC". Both DSPAB and DSPC include their own dedicated peripherals such as serial and parallel control ports, and serial audio interfaces. DSPC also has a external memory interface which supports SRAM/SDRAM/EPROM.

All the decoding/processing algorithms listed below require delivery of PCM or IEC61937-packed compressed data via I2S or LJ formatted digital audio to the CS49400. Today the CS49400 will support all of the following decoding/processing standards:

- PCM Pass-Through/PCM Upsampler

- Dolby Digital™ (with Dolby Pro Logic)™

- Dolby Digital Pro Logic II™

- Dolby Digital EX™

- Dolby Digital EX Pro Logic II™

- MPEG-2, Advanced Audio Coding Algorithm (AAC)

- MPEG Multichannel

- MPEG Multichannel with Dolby Pro Logic II™

- MPEG-1/2, Layer III (MP3)

- DTS Digital Surround™

- DTS 96/24™ (Front-end Decoder)

- DTS Digital Surround™ with Dolby Pro Logic II™

- DTS-ES Extended Surround™ (DTS-ES Discrete 6.1 and DTS-ES Matrix 6.1)

- DTS-ES 96/24™ (Front-end Decoder)

- DTS Neo:6™

- LOGIC7®

- SRS Circle Surround™ II

- HDCD®

All of the above audio decoding/processing algorithms and the associated application notes (AN208 and their corresponding appendices) are available through the Crystal Ware™ Software Licensing Program. Please refer to AN208 for the latest listing of application codes for DSPAB.

DSPC is unique to DSPAB in the sense that the designer may choose to just load a standard or enhanced application code (.ULD file) from the Crystal Ware Software Library or if they have access to the Cirrus Framework DSPC Development Kit, they may choose to build their own application code from a variety of modules. A DSPC application code contains all of the necessary post-processing modules, such as Crossbar Mixer, Pro Logic Module, Bass Manager Module, and Audio Manager (Kernel). A module is just a single processing module, such as Tone Control, Parametric/Graphic EQ, or Dolby Pro Logic matrix decoder. DSPC on the CS49400 will support the following post-processing application codes and/or modules:

- Standard Post-Processor (includes the following modules all compiled into one .ULD file): Downmixer module, Dualzone module, Crossbar Mixer module, 7.1 Channel Bass Manager module, Audio Manager module (Volume Control, Trim Control and Channel Remap), and Delay module

- Advanced Post-Processor (includes the all of the standard post-processing modules plus the Tone Control module, Parametric EQ module, Re-EQ module in all compiled into one .ULD)

- Dolby Pro Logic™

- Dolby Pro Logic II™

- SRS Circle Surround II™

- DTS Neo:6™

- LOGIC7®

- THX® Surround EX™ 7.1 Channel Post-Processor

- THX® Ultra2 Cinema™ 7.1 Channel Post-Processor

- Cirrus Extra Surround™

- Cirrus Original Multichannel Surround™

- Virtual Dolby Digital™/Virtual Dolby Digital Pro Logic II™ Virtualizer Module

- VMAx VirtualTheater™ Virtualizer Module

- HDCD® Multichannel Decoder

- DVD-Audio/Video and Multichannel SACD Bass Management

- DTS/DTS-ES 96/24™ Back-End Decoder

- DTS/DTS-ES 96/24™ Back-End Decoder with THX® Ultra2 Cinema™

All of the above audio post-processing applications codes and/or Cirrus FrameworkTM modules and the associated application notes (AN209 and the associated appendices) are available through the Crystal WareTM Software Licensing Program. All standard or enhanced post-processing code modules are only available to customers who qualify for the Cirrus Framework™ CS49400 Family DSPC Programming Kit. Please refer to AN209 for the latest listing of application codes and Cirrus FrameworkTM modules available for DSPC.

2.1 DSPAB

DSPAB is an enhanced version of the CS49300. It was designed to have legacy code support for all decoder applications developed for the CS49300. It includes performance enhancements such as the ability to decode AAC without the need for external SRAM memory. DSPAB has a Digital Audio Input (DAI) and a Compressed Data Input (CDI) port for data delivery in either I^2S or LJ format. DSPAB can be controlled serially using the SPI standard and can also be controlled via a Parallel host control port using the Motorola® or Intel® communication standards.

2.2 DSPC

DSPC is a 32-bit, general-purpose, fixed-point RAM-based processor which includes on-chip ROM tables. 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 such as AV receivers and DVD-Audio/Video players.

DSPC has on-chip data and program RAM, and both external SDRAM and SRAM memory interfaces. These interfaces can be used to expand the data memory. DSPC also has its own 8-channel digital audio input for post-processing PCM from a Multichannel Super Audio CD (SACD) input or DVD-Audio/Video input, via high-performance A/Ds or from some other type of multichannel digital input, capable of delivering 4 stereo digital audio channels such as IEEE1394 (a.k.a. I-Link® or Firewire®). Data can be delivered to this port using the standard audio formats (I^2S or LJ). DSPC can be controlled serially using the SPI standard or via Parallel host control port using the Motorola® or Intel® standard. DSPC has a Digital Audio output port that has eight stereo serial data outputs for a total of 16 channels. Data can be delivered from these outputs in serial I^2S or LJ format. Two of these outputs (AUDAT3, AUDAT7) can be configured as a IEC60958-format S/PDIF transmitter.

This document focuses on the electrical features of the CS49400. The features are described from a hardware design perspective. It should be understood that not all of the features portrayed in this document are supported by all of the versions of application code available. The application code user's guides should be consulted to determine which hardware features are supported by the software.

Please note that a download of application software is required each time the part is powered up. This term should be interpreted as meaning the transfer of application code into the internal memory of the part

from either an external microcontroller or through one of the boot procedures listed in Section 8.

3. TYPICAL CONNECTION DIAGRAMS

Four typical connection diagrams have been presented to illustrate using the part with the different communication modes available. They are as follows:

Figure 27, "SPI Control with External Memory - 144 Pin Package" on page 38.
Figure 28, "Intel® Parallel Control Mode - 144 Pin Package" on page 39.
Figure 29, "Motorola® Parallel Control Mode - 144 Pin Package" on page 40.

The following should be noted when viewing the typical connection diagrams:

> **Note:** The pins are grouped functionally in each of the typical connection diagrams. Please be aware that the CS49400 symbol may appear differently in each diagram.
>
> The external memory interface is supported when a serial or parallel communication mode has been chosen.

3.1 Multiplexed Pins

The CS49400 incorporates a large amount of flexibility into a 144 pin package. The pins are internally multiplexed to serve multiple purposes. Some pins are designed to operate in one mode at power up, and serve a different purpose when the DSP is running. Other pins have functionality which can be controlled by the application running on the DSP. In order to better explain the behavior of the part, the pins which are multiplexed have been given multiple names. Each name is specific to the pin's operation in a particular mode.

In this document, pins will be referred to by their functionality. Section 12 "Pin Description" on page 86 describes each pin of the CS49400 and lists all of its names. Please refer to this section when exact pin numbers are in question.

3.2 Termination Requirements

The CS49400 incorporates open drain pins which must be pulled high for proper operation. $\overline{\text{FINTREQ}}$ and $\overline{\text{INTREQ}}$ are always open drains which requires a pull-up for proper operation.

Due to the internal, multiplexed design of the pins, certain signals may or may not require termination depending on the mode being used. If a parallel host communication mode is not being used, all parallel control pins must be terminated or driven as these pins will come up as high impedance inputs and will be prone to oscillation if they are left floating. The specific termination requirements may vary since the state of some of the GPIO pins will determine the communication mode at the rising edge of reset (please see Section 6 "Control" on page 41 for more information). For the explicit termination requirements of each communication mode please see the typical connection diagrams.

Generally a 3.3k Ohm resistor is recommended for open drain and mode select pins. A 10k Ohm resistor is sufficient for all other unused inputs.

3.3 Phase Locked Loop Filter

The internal phase locked loop (PLL) of the CS49400 requires an external filter. The topology of this filter is shown in the typical connection diagrams. The component values are shown below. Care should be taken when laying out the filter circuitry to minimize trace lengths and to avoid any high frequency signals. Any noise coupled onto the filter circuit will be directly coupled into the PLL, which could affect performance.

Reference Designator	Value
C1	2.2uF
C2	1200pF
C3	68pF
R1	3k Ohm

Table 1. PLL Filter Component Values

4. POWER

The CS49400 requires a 2.5V digital power supply for the core logic and 2.5V I/O and a 2.5V analog power supply for the internal PLL. For systems with external memory that runs on 3.3V, a 3.3V digital power supply is required on the VDDSD pins along with four digital grounds on VSSSD. There are seven digital power pins, VDD1 through VDD7, along with seven digital grounds, VSS1 through VSS7. There is one analog power pin, PLLVDD, and one analog ground, PLLVSS. The recommendations given below for decoupling and power conditioning of the CS49400 will help to ensure reliable performance.

4.1 Decoupling

It is necessary to decouple the power supply by placing capacitors directly between the power and ground of the CS49400. Each pair of power/ground pins (VDD1/VSS1, etc.) should have its own decoupling capacitors. The recommended procedure is to place both a 0.1uF and a 1uF capacitor as close as physically possible to each power pin. The 0.1uF capacitor should be closest to the part (typically 5mm or closer).

4.2 Analog Power Conditioning

In order to obtain the best performance from the CS49400's internal PLL, the analog power supply PLLVDD must be as noise-free as possible. A ferrite bead and two capacitors should be used to filter the VDD to generate PLLVDD. This power scheme is shown in the typical connection diagrams.

4.3 Ground

For two layer circuit boards, care should be taken to have sufficient ground between the DSP and parts in which it will be interfacing (DACs, ADCs, S/PDIF Receivers, microcontrollers, and especially external memory). Insufficient ground can degrade noise margins between devices resulting in data integrity problems.

4.4 Pads

The CS49400 has two different I/O voltage levels. All signal pins not associated with the External SRAM/SDRAM memory interface operate from the 2.5V supply and are 3.3V tolerant. The external SRAM/SDRAM memory interface operates at 3.3V only. However, if the external memory interface is not used VDDSD1-4 may be connected to 2.5V.

Figure 27. SPI Control with External Memory - 144 Pin Package

Figure 28. Intel® Parallel Control Mode - 144 Pin Package

NOTE
1. A capacitor pair (.01uF and 0.1uF) must be supplied for each power pin.
2. The digital supply (+2.5 VD) is filtered to obtain the analog supply (+2.5 VA).

CS494XX

MICROCONTROLLER

PARALLEL INTERFACE

ADC OR SPDIF RX

ADC

ADCs

OSCILLATOR

EXTERNAL ROM

6 DACs

2 SPDIF TX

RESISTOR PACK
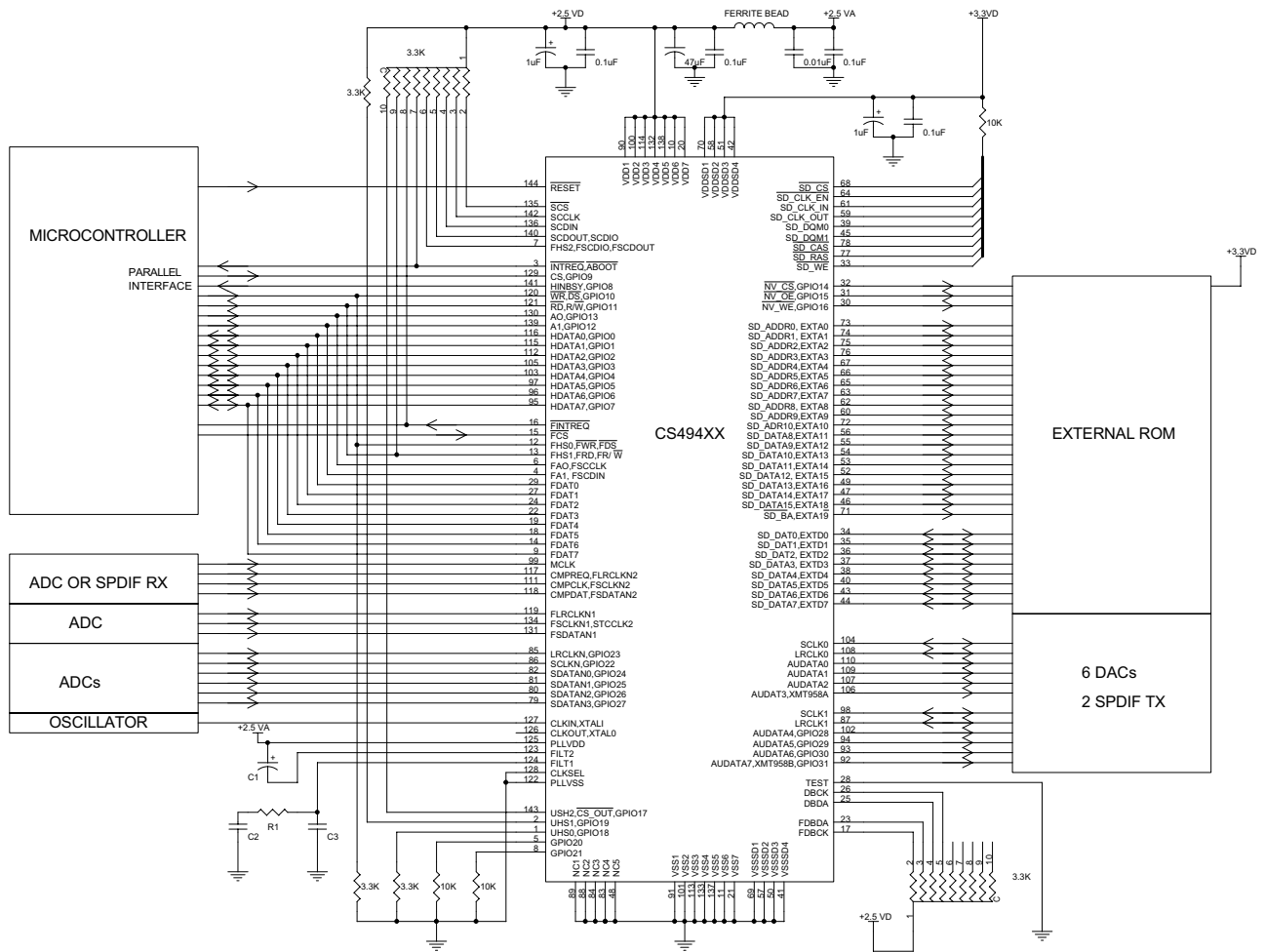
Figure 29. Motorola® Parallel Control Mode - 144 Pin Package

5. CLOCKING

The CS49400 clock manager 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, the CLKIN pin is typically connected to a 12.288MHz oscillator.

The clock manager is controlled by the DSPAB application software. The software user's guide for the application code being used should be referenced for which CLKIN input frequency is supported.

6. CONTROL

Control of the CS49400 can be accomplished through one of three methods. The CS49400 supports SPI serial communication and Motorola® and Intel® byte-wide parallel communication. Both DSPAB and DSPC have their own control ports. Only one of the three communication modes can be selected for control. Both DSPAB and DSPC use the same communication mode. However, please note that the 100-pin package only supports SPI serial communication. The states of the FHS[2:0] for DSPAB and UHS[2:0] for DSPC, sampled at the rising edge of RESET, determine the communication interface (Table 2.)

144-Pin Package			
FHS2 (Pin 7)	FHS1 (Pin 13)	FHS0 (Pin 12)	Host Interface Mode
1	0	1	Serial SPI
1	1	0	8-bit Intel®
1	1	1	8-bit Motorola®
100-Pin Package			
FHS2 (Pin 6)	FHS1 (Pin 10)	FHS0 (Pin 9)	Host Interface Mode
1	0	1	Serial SPI

Table 2. Host Modes for DSPAB

.

144-Pin Package			
UHS2 (Pin 143)	UHS1 (Pin 2)	UHS0 (Pin 1)	Host Interface Mode
1	0	1	Serial SPI
1	1	0	8-bit Intel®
1	1	1	8-bit Motorola®
100-Pin Package			
UHS2 (Pin 99)	UHS1 (Pin 2)	UHS0 (Pin 1)	Host Interface Mode
1	0	1	Serial SPI

Table 3. Host Modes for DSPC

Whichever host communication mode is used, host control of the CS49400 is handled through the application software running on the DSP. Configuration and control of the CS49400 decoder and its peripherals are indirectly executed through a messaging protocol supported by the downloaded application code. 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 can be found in any of the software user's guides, such as AN208 and AN209.

The system designer only needs to read the subsection describing the communication mode being used. Please note that the communication protocol might be slightly different for DSPAB and DSPC.

The following sections will explain each communication mode in more detail. Flow diagrams will illustrate read and write cycles.

The timing diagrams shown demonstrate relative edge positions of signal transitions for read and write operations.

6.1 Serial Communication

6.1.1 SPI Communication for DSPAB

SPI communication with DSPAB is accomplished with five communication lines: chip select, serial control clock, serial data in, serial data out, and an interrupt request line that signals DSPAB has data to transmit to the host. Table 4 lists the mnemonic,

CIRRUS LOGIC®

pin name, and pin number of each of these signals on DSPAB.

Mnemonic	Pin Name	144-Pin Package, Pin Number	100-Pin Package, Pin Number
Chip Select	$\overline{\text{FCS}}$	15	11
Serial Clock	FSCCLK	6	5
Serial Data In	FSCDIN	4	4
Serial Data Out	FSCDOUT	7	6
Interrupt Request	$\overline{\text{FINTREQ}}$	16	12

Table 4. SPI Communication Signals for DSPAB

6.1.1.1 Writing in SPI for DSPAB

When writing to the device in SPI the same protocol will be used whether writing a byte, a message, or even an entire executable download image. The examples shown in this document can be expanded to fit any write situation. Figure 30, "SPI Write Flow Diagram for DSPAB" on page 42 shows a typical write sequence:
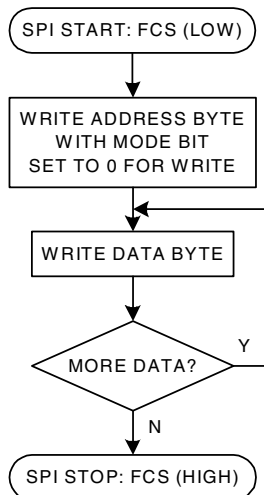
Figure 30. SPI Write Flow Diagram for DSPAB

The following is a detailed description of an SPI write sequence with DSPAB.

1) An SPI transfer is initiated when chip select ($\overline{\text{FCS}}$) is driven low.

2) This is followed by a 7-bit address and the read/write bit set low for a write. The address

for DSPAB defaults to 1000000b. It is necessary to clock this address in prior to any transfer in order for DSPAB to accept the write. In other words a byte of 0x80 should be clocked into the device preceding any write. The 0x80 byte represents the 7-bit address 1000000b, with the least significant bit set to 0 to designate a write.

3) The host should then clock data into the device most significant bit first, one byte at a time. The data byte is transferred from the shift register to the DSP input register on the falling edge of the eighth serial clock. For this reason, the serial clock should default to low so that eight transitions from low to high to low will occur for each byte. A 32 µS byte to byte latency must be obeyed during run time.

4) When all of the bytes have been transferred, chip select should be raised to signify an end of write. Once again it is crucial that the serial clock transitions from high to low on the last bit of the last byte before chip select is raised, or a loss of data will occur.

The same write routine could be used to send a single byte, a message, or an entire application code image. From a hardware perspective, it makes no difference whether communication is by byte or multiple bytes of any length as long as the correct hardware protocol is followed.

6.1.1.2 Reading in SPI for DSPAB

A read operation is necessary when DSPAB signals that it has data to be read. DSPAB does this by dropping its interrupt request line ($\overline{\text{FINTREQ}}$) low. When reading from the device in SPI, the same protocol will be used whether reading a single byte or multiple bytes. The examples shown in this document can be expanded to fit any read situation. Figure 32, "SPI Read Flow Diagram for DSPAB" on page 44 shows a typical read sequence:

The following is a detailed description of an SPI read sequence with DSPAB.

SPI Write Functional Timing

SPI Read Functional Timing
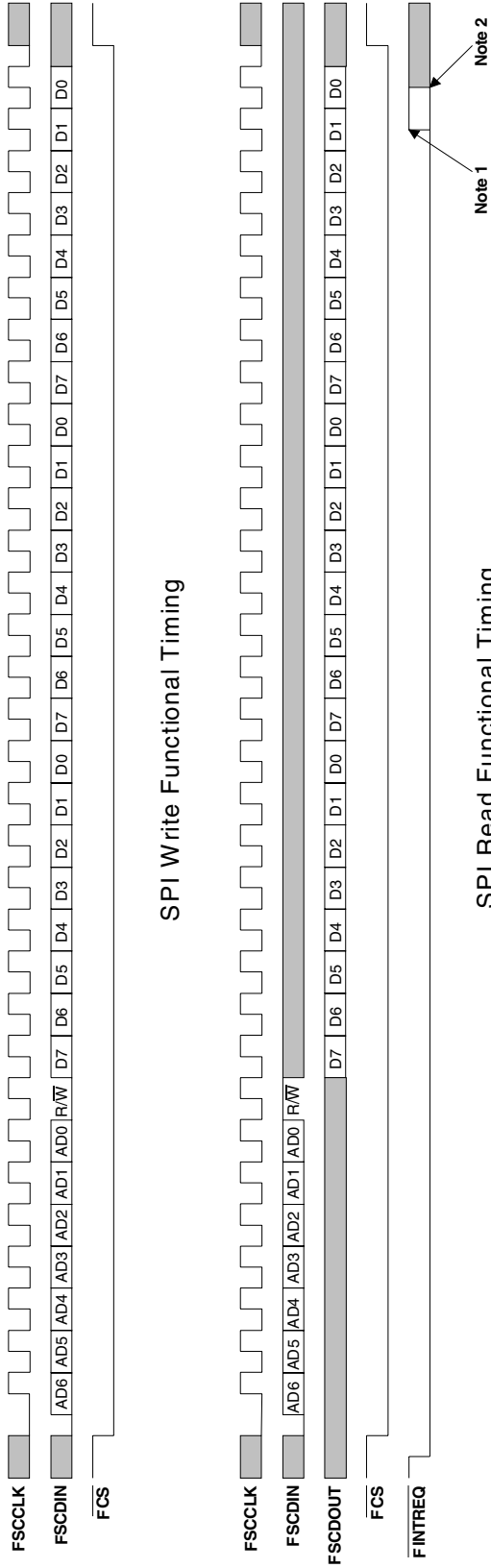
Notes: 1. FINTREQ is guaranteed to stay LOW until the rising edge of FSCCLK for bit D1 of the last byte to be transferred out of the CS49400.

2. FINTREQ is guaranteed to remain HIGH until the next rising edge of FSCCLK at which point it may go LOW again if there is new data to be read. The condition of FINTREQ going LOW at this point should be treated as a new read condition. After a stop condition, a new start condition and an address byte should be sent

Figure 31. SPI Timing for DSPAB
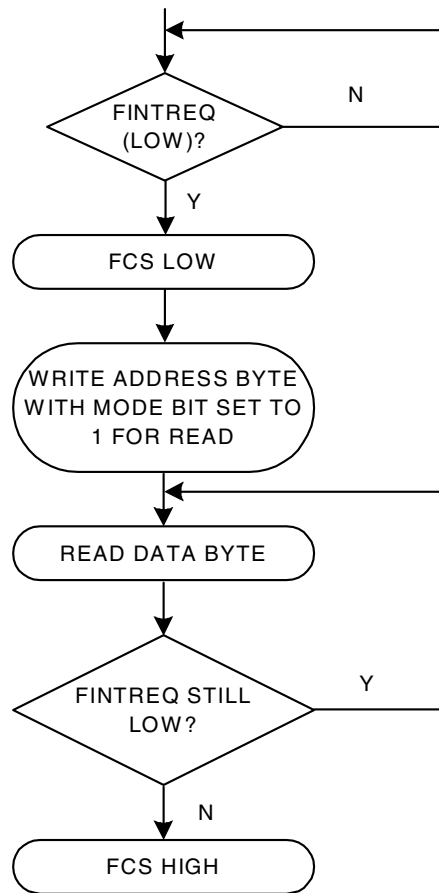
Figure 32. SPI Read Flow Diagram for DSPAB

1) An SPI read transaction is initiated by DSPAB dropping $\overline{\text{FINTREQ}}$, signaling that it has data to be read.

2) The host responds by driving chip select ($\overline{\text{FCS}}$) low.

3) This is followed by a 7-bit address and the read/write bit set high for a read. The address for DSPAB defaults to 1000000b. It is necessary to clock this address in prior to any transfer in order for DSPAB to acknowledge the read. In other words a byte of 0x81 should be clocked into the device preceding any read. The 0x81 byte represents the 7-bit address 1000000b, and the least significant bit set to 1

to designate a read.

4) After the falling edge of the serial control clock (FSCCLK) for the read/write bit, the data is ready to be clocked out on the control data out pin (FCDOUT). Data clocked out by the host is valid on the rising edge of FSCCLK. Data transitions occur on the falling edge of FSCCLK. The serial clock should be default low so that eight transitions from low to high to low will occur for each byte.

5) If $\overline{\text{FINTREQ}}$ is still low, another byte should be clocked out of DSPAB. Please see the discussion below for a complete description of $\overline{\text{FINTREQ}}$ behavior.

6) When $\overline{\text{FINTREQ}}$ is released, the chip select line of DSPAB should be taken high to end the read transaction.

Understanding the role of $\overline{\text{FINTREQ}}$ is important for successful communication. $\overline{\text{FINTREQ}}$ is guaranteed to remain low (once it has gone low) until the second to last rising edge of FSCCLK of the last byte to be transferred out of DSPAB. If there is no more data to be transferred, $\overline{\text{FINTREQ}}$ will go high at this point. For SPI this is the rising edge for the second to last bit of the last byte to be transferred. After going high, $\overline{\text{FINTREQ}}$ is guaranteed to stay high until the next rising edge of FSCCLK. This end of transfer condition signals the host to end the read transaction by clocking the last data bit out and raising $\overline{\text{FCS}}$. If $\overline{\text{FINTREQ}}$ is still low after the second to last rising edge of FSCCLK, the host 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 transaction or a loss of data will occur. In other words, all data should be read out of the chip until $\overline{\text{FINTREQ}}$ signals the last byte by going high as described above. Please see Section 6.1.3 "FINTREQ Behavior: A Special Case" on page 48 for a more detailed description of $\overline{\text{FINTREQ}}$ behavior.

Figure 31, "SPI Timing for DSPAB" on page 43 timing diagram shows the relative edges of the control lines for an SPI read and write.

6.1.2 SPI Communication for DSPC

SPI communication with the DSPC is accomplished with five communication lines: chip select, serial control clock, serial data in, serial data out, and an interrupt request line that signals DSPC has data to transmit to the host. Table 5 lists the mnemonic, pin name, and pin number of each of these signals on DSPC.

Mnemonic	Pin Name	144-Pin Package, Pin Number	100-Pin Package, Pin Number
Chip Select	$\overline{\text{SCS}}$	135	93
Serial Clock	SCCLK	142	98
Serial Data In	SCDIN	136	94
Serial Data Out	SCDOUT	140	97
Host Busy	HINBSY	141	N/A
Interrupt Request	$\overline{\text{INTREQ}}$	3	3

Table 5. SPI Communication Signals for DSPC

6.1.2.1 Writing in SPI for DSPC

When writing to the device in SPI the same protocol will be used whether writing a byte, a message or even an entire executable download image. The examples shown in this document can be expanded to fit any write situation. Figure 33, "SPI Write Flow Diagram for DSPC" on page 45 shows a typical write sequence

The following is a detailed description of an SPI write sequence with DSPC.
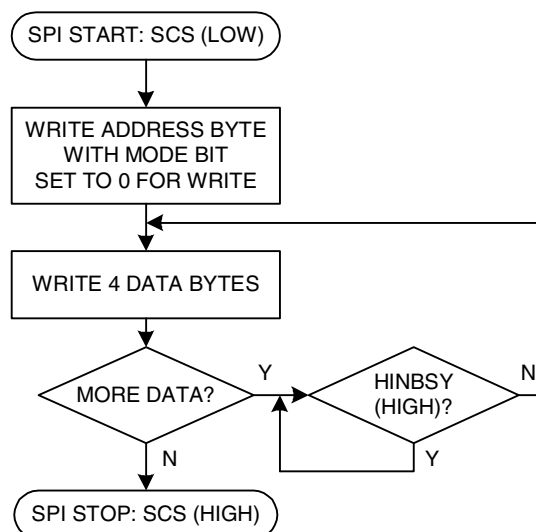
Figure 33. SPI Write Flow Diagram for DSPC

1) An SPI transfer is initiated when chip select ($\overline{\text{SCS}}$) is driven low.

2) This is followed by a 7-bit address and the read/write bit set low for a write. The address for DSPC defaults to 1000001b. It is necessary to clock in this address prior to any transfer in order for DSPC to accept the write. In other words a byte of 0x82 should be clocked into the device preceding any write. The 0x82 byte represents the 7-bit address 1000001b, and the least significant bit set to 0 to designate a write.

3) The host should then clock data into the device most significant bit first, four bytes at a time. The data byte is transferred to the DSP on the falling edge of the eighth serial clock. For this

reason, the serial clock should default to low so that eight transitions from low to high to low will occur for each byte.

4) When all 4 data bytes have been transferred, chip select should be raised to signify an end of write. Once again it is crucial that the serial clock transitions from high to low on the last bit of the last byte before chip select is raised, or a loss of data will occur.

5) If more data needs to be sent, the host must verify that the HINBSY pin is low before it sends more data to DSPC. Note the HINBSY pin is available only on the 144 pin devices. A 32 μS byte to byte latency must be obeyed during run time for the 100 pin devices.

The same write routine could be used to send a 4-byte message or an entire application code image. From a hardware perspective, communication must be in 4-byte multiples.

6.1.2.2 Reading in SPI for DSPC

A read operation is necessary when DSPC signals that it has data to be read. DSPC does this by dropping its interrupt request line ($\overline{\text{INTREQ}}$) low. When reading from the device in SPI, the same protocol will be used whether reading a single byte or multiple bytes. The examples shown in this document can be expanded to fit any read situation. Figure 35, "SPI Read Flow Diagram for DSPC" on page 46 shows a typical read sequence:

The following is a detailed description of an SPI read sequence with DSPC.

1) An SPI read transaction is initiated by DSPC dropping $\overline{\text{INTREQ}}$, signaling that it has data to be read.

2) The host responds by driving chip select ($\overline{\text{SCS}}$) low.

3) This is followed by a 7-bit address and the read/write bit set high for a read. The address for DSPC defaults to 1000001b. It is necessary to clock this address in prior to any transfer in
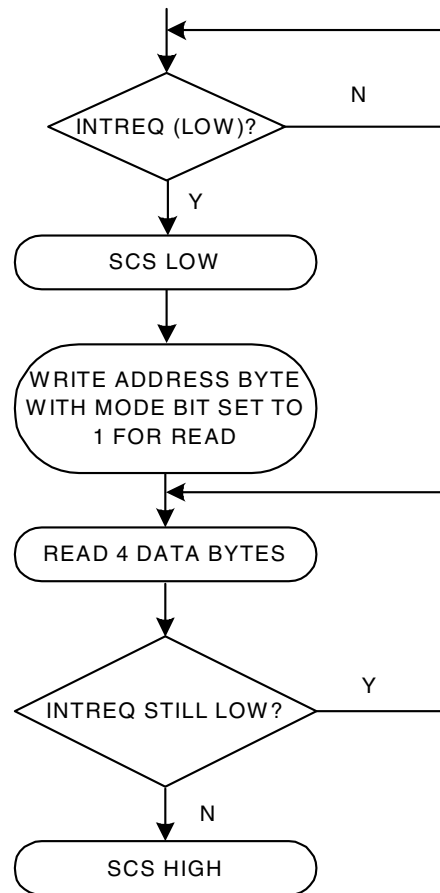
Figure 35. SPI Read Flow Diagram for DSPC

order for DSPC to acknowledge the read. In other words a byte of 0x83 should be clocked into the device preceding any read. The 0x83 byte represents the 7 bit address 1000001b, and the least significant bit set to 1 designates a read.

4) After the falling edge of the serial control clock (SCCLK) for the read/write bit, the data is ready to be clocked out on the control data out pin (CDOUT). Data clocked out by the host is valid on the rising edge of SCCLK and data transitions occur on the falling edge of SCCLK. The serial clock should default to low so that eight transitions from low to high to low will occur for each byte.

SPI Write Functional Timing

SPI Read Functional Timing
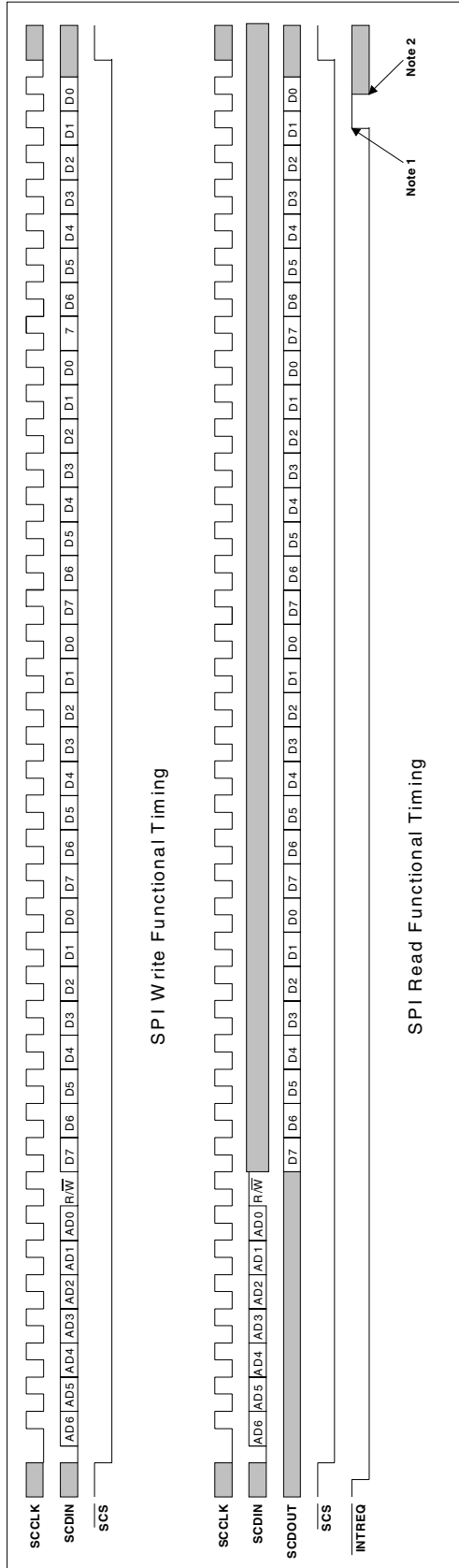
Notes: 1. INTREQ is guaranteed to stay LOW until the rising edge of SCCLK for bit D1 of the last byte to be transferred out of DSPC.

2. INTREQ is guaranteed to remain HIGH until the next rising edge of SCCLK at which point it may go LOW again if there is new data to be read. The condition of INTREQ going LOW at this point should be treated as a new read condition. After a stop condition, a new start condition and an address byte should be sent

Figure 34. SPI Timing for DSPC

5) If $\overline{\text{INTREQ}}$ is still low after 4 bytes, another 4 bytes should be clocked out of DSPC.

6) When $\overline{\text{INTREQ}}$ is released, the chip select line of DSPC should be taken high to end the read transaction.

All messages read back from DSPC will be in 4-byte multiples.

6.1.3 $\overline{\text{FINTREQ}}$ Behavior: A Special Case

When communicating with DSPAB there are two types of messages which force $\overline{\text{FINTREQ}}$ to go low. These messages are known as solicited messages and unsolicited messages. For more information on the specific types of messages that require a read from the host, one of the application code user's guides should be referenced.

In general, when communicating with DSPAB, $\overline{\text{FINTREQ}}$ will not go low unless the host first sends a read request command message. In other words the host must solicit a response from the DSP. In this environment, the host must read from DSPAB until $\overline{\text{FINTREQ}}$ goes high again. Once the $\overline{\text{FINTREQ}}$ pin has gone high it will not be driven low until the host sends another read request.

When unsolicited messages, such as those used for Autodetect, have been enabled, the behavior of $\overline{\text{FINTREQ}}$ is noticeably different. DSPAB will drop the $\overline{\text{FINTREQ}}$ pin whenever it has an outgoing message, even though the host may not have requested data.

There are three ways in which $\overline{\text{FINTREQ}}$ can be affected by an unsolicited message:

1) During normal operation, while $\overline{\text{FINTREQ}}$ is high, DSPAB could drop $\overline{\text{FINTREQ}}$ to indicate an outgoing message, without a prior read request.

2) The host is in the process of reading from DSPAB, meaning that $\overline{\text{FINTREQ}}$ is already low. An unsolicited message arrives which forces $\overline{\text{FINTREQ}}$ to remain low after the solicited message is read.

3) The host is reading from DSPAB when the unsolicited message is queued, but $\overline{\text{FINTREQ}}$ goes

high for one period of FSCCLK and then goes low again before the end of the read cycle.

In case (1) the host should perform a read operation as discussed in the previous sections.

In case (2) an unsolicited message arrives before the second to last FSCCLK of the final byte transfer of a read, forcing the $\overline{\text{FINTREQ}}$ pin to remain low. In this scenario the host should continue to read from DSPAB without a stop/start condition or data will be lost.

In case (3) an unsolicited message arrives between the second to last FSCCLK and the last FSCCLK of the final byte transfer of a read. In this scenario, $\overline{\text{FINTREQ}}$ will transition high for one clock (as if the read transaction has ended), and then back low (indicating that more data has queued). This final case is the most complicated and shall be explained in detail.

There are two constraints which completely characterize the behavior of the $\overline{\text{FINTREQ}}$ pin during a read. The first constraint is that the $\overline{\text{FINTREQ}}$ pin is guaranteed to remain low until the second to last FSCCLK (FSCCLK number N-1) of the final byte being transferred from DSPAB (not necessarily the second to last bit of the data byte). The second constraint is that once the $\overline{\text{FINTREQ}}$ pin has gone high it is guaranteed to remain high until the rising edge of the last FSCCLK (FSCCLK number N) of the final byte being transferred from DSPAB (not necessarily the last bit of the data byte). If an unsolicited message arrives in the window of time between the rising edge of the second to last FSCCLK and the final FSCCLK, $\overline{\text{FINTREQ}}$ will drop low on the rising edge of the final FSCCLK as illustrated in the functional timing diagram shown for the SPI read cycle.

$\overline{\text{FINTREQ}}$ behavior for SPI communication is illustrated in Figure 31, "SPI Timing for DSPAB" on page 43. When using SPI communication, the $\overline{\text{FINTREQ}}$ pin will remain low until the rising edge of FSCCLK for the data bit D1 (FSCCLK N-1), but it can go low at the rising edge of FSCCLK for data bit D0 (FSCCLK N) if an unsolicited message has

arrived. If no unsolicited messages arrive, the $\overline{\text{FINTREQ}}$ pin will remain high after rising.

Ideally, the host will sample $\overline{\text{FINTREQ}}$ on the falling edge of FSCCLK number N-1 of the final byte of each read response message. If $\overline{\text{FINTREQ}}$ is sampled high, the host should conclude the current read cycle using the stop condition defined for the communication mode chosen. The host should then begin a new read cycle complete with the appropriate start condition and the chip address. If $\overline{\text{FINTREQ}}$ is sampled low, the host should continue reading the next message from DSPAB without ending the current read cycle.

When using automated communication ports, however, the host is often limited to sampling the status of $\overline{\text{FINTREQ}}$ after an entire byte has been transferred. In this situation a low-high-low transition (case 3) would be missed and the host will see a constantly low $\overline{\text{FINTREQ}}$ pin. Since the host should read from DSPAB until it detects that $\overline{\text{FINTREQ}}$ has gone high, this condition will be treated as a multiple-message read (more than one read response is provided by DSPAB). Under these conditions a single byte of 0x00 will be read out before the unsolicited message.

The length of every read response is defined in the user's manual for each piece of application code. Thus, the host should know how many bytes to expect based on the first byte (the OPCODE) of a read response message. It is guaranteed that no read responses will begin with 0x00, which means that a NULL byte (0x00) detected in the OPCODE position of a read response message should be discarded. Please see an Application Code User's Guide for an explanation of the OPCODE.

It is important that the host be aware of the presence of NULL bytes, or the communication channel could become corrupted.

When case (3) occurs and the host issues a stop condition before starting a new read cycle, the first byte of the unsolicited message is loaded directly into the shift register and 0x00 is never seen.

Alternatively, if case (3) occurs and the host continues to read from DSPAB without a stop condition (a multiple message read), the 0x00 byte must be shifted out of DSPAB before the first byte of the unsolicited message can be read.

In other words, if a host can only sample $\overline{\text{FINTREQ}}$ after an entire byte transfer the following routine should be used if $\overline{\text{FINTREQ}}$ is low after the last byte of the message being read:

1) Read one byte

2) If the byte = 0x00 discard it and skip to step 3. If the byte != 0x00 then it is the OPCODE for the next message. For this case skip to step 4.

3) Read one more byte. This is the OPCODE for the next message.

4) Read the rest of the message as indicated in the previous sections.

6.2 Parallel Host Communication for DSPAB

The parallel host communication modes of DSPAB provide an 8-bit interface to the DSP. An Intel-style parallel mode and a Motorola-style parallel mode are supported. The host interface is implemented using four communication registers within DSPAB as shown in Table 6, "Parallel Input/Output Registers for DSPAB," on page 51.

When the host is downloading code to DSPAB or configuring the application code, control messages will be written to (and read from) the Host Message register. The Host Control register is used during messaging sessions to determine when DSPAB can accept another byte of control data, and when DSPAB has an outgoing byte that should be read.

All communication to DSPAB after download is in 24-bit words. Reads and writes are done in multiples of 3-byte transactions. A 3-byte transaction is accomplished by doing three consecutive byte reads or byte writes.

The PCM Data and Compressed Data registers are used strictly for the transfer of audio data. The host cannot read from these two registers. Audio data written to registers 11b and 10b are transferred directly to the internal FIFOs of DSPAB. When the level of the PCM FIFO reaches the FIFO threshold level, the MFC bit of the Host Control register will be set. When the level of the Compressed Data FIFO reaches the FIFO threshold level, the MFB bit of the Host Control register will be set. Writing data directly to the FIFOs is only supported in specific applications. To see if an application supports this feature, consult the appropriate Application Note.

A detailed description for each parallel host mode will now be given. The following information will be provided for the Intel mode and Motorola mode:

- The pins of DSPAB which must be used for proper communication

- Flow diagram and description for a parallel byte write

- Flow diagram and description for a parallel byte read

The four registers of DSPAB's parallel host mode are not used identically. The algorithm used for communicating with each register will be given as a functional description, building upon the basic read and write protocols defined in the Motorola and Intel sections. The following will be covered:

- Flow diagram and description for a control write

- Flow diagram and description for a control read

6.2.5 Intel Parallel Host Communication Mode for DSPAB

The Intel parallel host communication mode is implemented using the pins given in Table 6. Parallel host communication is available only on the 144-pin package part.

Mnemonic	Pin Name	144-Pin Package, Pin Number
Chip Select	$\overline{\text{FCS}}$	15
Write Enable	$\overline{\text{FWR}}$	12
Output Enable	$\overline{\text{FRD}}$	13
Register Address Bit 1	FA1	4
Register Address Bit 0	FA0	6
Interrupt Request	$\overline{\text{FINTREQ}}$	16
DATA7	FDAT7	9
DATA6	FDAT6	14
DATA5	FDAT5	18
DATA4	FDAT4	19
DATA3	FDAT3	22
DATA2	FDAT2	24
DATA1	FDAT1	27
DATA0	FDAT0	29

Table 6. Intel Mode Communication Signals for DSPAB

The HOUTRDY bit of the Host Control Register (A[1:0] = 01b) indicates that the DSP has a message for the host to read. The $\overline{\text{FINTREQ}}$ pin can be controlled by the application code, and allows for another method of requesting that the

6.2.1 Host Message (HOSTMSG) Register, A[1:0] = 00b

7	6	5	4	3	2	1	0
HOSTMSG7	HOSTMSG6	HOSTMSG5	HOSTMSG4	HOSTMSG3	HOSTMSG2	HOSTMSG1	HOSTMSG0

HOSTMSG7–0   Host data to and from the DSP. A read or write of this register operates handshake bits between the internal DSP and the external host. This register typically passes multibyte messages carrying microcode, control, and configuration data (messages are written MSB first). HOSTMSG is physically implemented as two independent registers for input and output (Read and write).

6.2.2 Host Control (CONTROL) Register, A[1:0] = 01b

7	6	5	4	3	2	1	0
Reserved	CMPRST	PCMRST	MFC	MFB	HINBSY	HOUTRDY	Reserved

Reserved   Always write a 0 for future compatibility.

CMPRST   When set, initializes the CMPDATA compressed data input channel. Writing a one to this bit holds the port in reset. Writing zero enables the port. This bit must be low for normal operation. (Write only)

PCMRST   When set, initializes the PCMDATA linear PCM input channel. Writing a one to this bit holds the port in reset. Writing zero enables the port. This bit must be low for normal operation. (Write only)

MFC   When high, indicates that the PCMDATA input buffer is almost full. (Read only)

MFB   When high, indicates that the CMPDATA input buffer is almost full. (Read only)

HINBSY   Set when the host writes to HOSTMSG. Cleared when the DSP reads data from the HOSTMSG register. The host reads this bit to determine if the last host byte written has been read by the DSP. (Read only)

HOUTRDY   Set when the DSP writes to the HOSTMSG register. Cleared when the host reads data from the HOSTMSG register. The DSP reads this bit to determine if the last DSP output byte has been read by the host. (Read only)

Reserved   Always write a 0 for future compatibility.

6.2.3 PCM Data Input (PCMDATA) Register, A[1:0] = 10b

7	6	5	4	3	2	1	0
PCMDATA7	PCMDATA6	PCMDATA5	PCMDATA4	PCMDATA3	PCMDATA2	PCMDATA1	PCMDATA0

PCMDATA7–0   The host writes PCM data to the DSP input buffer at this address. (Write only)

6.2.4 Compressed Data Input (CMPDATA) Register, A[1:0] = 11b

7	6	5	4	3	2	1	0
CMPDATA7	CMPDATA6	CMPDATA5	CMPDATA4	CMPDATA3	CMPDATA2	CMPDATA1	CMPDATA0

Table 6. Parallel Input/Output Registers for DSPAB
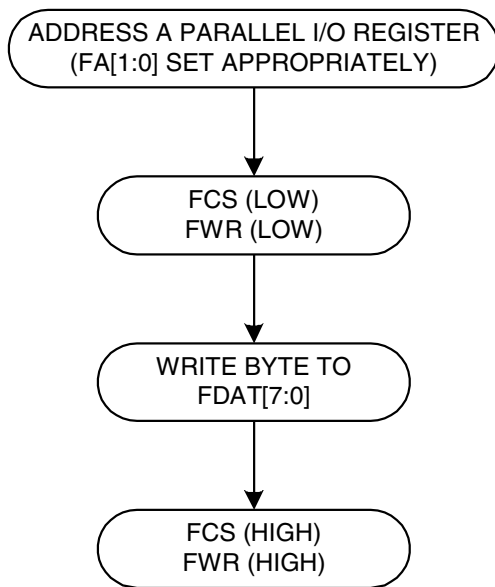
Figure 36. Intel Mode, One-Byte Write Flow Diagram for DSPAB

host read a message. When the code supports $\overline{\text{FINTREQ}}$ notification, the $\overline{\text{FINTREQ}}$ pin is asserted (driven low) when the DSP has an outgoing message for the host.

$\overline{\text{FINTREQ}}$ is useful for informing the host of unsolicited messages without polling. An unsolicited message is defined as a message generated by the DSP without an associated host read request. Unsolicited messages are used to notify the host of conditions such as a change in the incoming audio data type (e.g. PCM --> AC-3). Most unsolicited messages must be specifically enabled by setting the appropriate bit in the application's manager, enabling autodetection, or enabling other host notification capabilities.

6.2.5.1 Writing a Byte in Intel Mode for DSPAB

Information provided in this section is intended as a functional description of how to write control information to DSPAB. The system designer must ensure that all of the timing constraints of the Intel Parallel Host Mode Write Cycle are met.

The flow diagram shown in Figure 36 illustrates the sequence of events that define a one-byte write in Intel mode. The protocol presented in Figure 36 will now be described in detail.

1) The host must first drive the FA1 and FA0 register address pins of DSPAB with the address of the desired Parallel I/O Register. The address must be maintained for the duration of the write cycle.

   Host Message: FA[1:0]==00b.

   Host Control: FA[1:0]==01b.

   PCMDATA: FA[1:0]==10b.

   CMPDATA: FA[1:0]==11b.

2) The host then indicates that the selected register will be written. The host initiates a write cycle by driving the $\overline{\text{FCS}}$ and $\overline{\text{FWR}}$ pins low.

3) The host drives the data byte to the FDAT[7:0] pins of DSPAB.

4) Once the setup time for the write has been met, the host ends the write cycle by driving the $\overline{\text{FCS}}$ and $\overline{\text{FWR}}$ pins high.

6.2.5.2 Reading a Byte in Intel Mode for DSPAB

Information provided in this section is intended as a functional description of how to read control information from DSPAB. The system designer must ensure that all of the timing constraints of the Intel Parallel Host Mode Read Cycle are met.

The flow diagram shown in Figure 37 illustrates the sequence of events that define a one-byte read in Intel mode. The protocol presented in Figure 37 will now be described in detail.

1) The host must first drive the FA1 and FA0 register address pins of DSPAB with the address of the desired Parallel I/O Register. Note that only the Host Message register and the Host Control register can be read. The address must be maintained for the duration of the read cycle.

   Host Message: FA[1:0]==00b.

   Host Control: FA[1:0]==01b.

2) The host initiates a read cycle by driving the $\overline{\text{FCS}}$ and $\overline{\text{FRD}}$ pins low (bus must be tri-stated
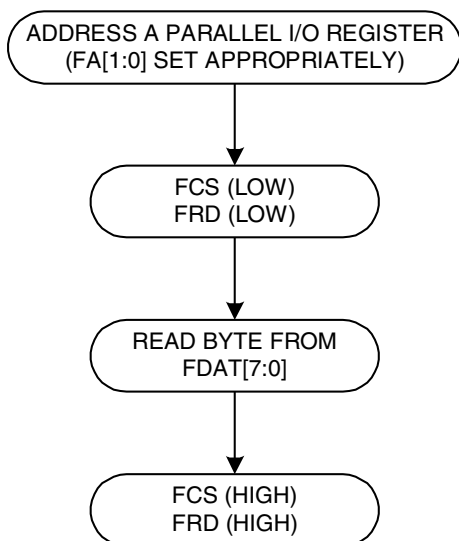
Figure 37. Intel Mode, One-Byte Read Flow Diagram for DSPAB

Mnemonic	Pin Name	144-Pin Package, Pin Number
Chip Select	$\overline{FCS}$	15
Data Strobe	$\overline{FDS}$	12
Read or Write Select	FR/$\overline{W}$	13
Register Address Bit 1	FA1	4
Register Address Bit 0	FA0	6
Interrupt Request	$\overline{FINTREQ}$	16
DATA7	FDAT7	9
DATA6	FDAT6	14
DATA5	FDAT5	18
DATA4	FDAT4	19
DATA3	FDAT3	22
DATA2	FDAT2	24
DATA1	FDAT1	27
DATA0	FDAT0	29

Table 7. Motorola Mode Communication Signals for DSPAB

before this occurs).

3) Once the data is valid (after waiting the appropriate time specified in the timing specifications), the host can read the value of the selected register from the FDAT[7:0] pins of DSPAB.

4) The host should now terminate the read cycle by driving the $\overline{FCS}$ and $\overline{FRD}$ pins high.

6.2.6 Motorola Parallel Communication Mode for DSPAB

The Motorola parallel host communication mode is implemented using the pins given in Table 7. Parallel host communication is available only on the 144-pin package part .

The HOUTRDY bit of the Host Control Register (A[1:0] = 01b) indicates that the DSP has a message for the host to read. The $\overline{FINTREQ}$ pin can be controlled by the application code, and allows for another method of requesting that the host read a message. When the code supports $\overline{FINTREQ}$ notification, the $\overline{FINTREQ}$ pin is asserted (driven low) whenever the DSP has an outgoing message for the host.

$\overline{FINTREQ}$ is useful for informing the host of unsolicited messages. An unsolicited message is defined as a message generated by the DSP without an associated host read request. Unsolicited messages are used to notify the host of conditions such as a change in the incoming audio data type (e.g. PCM --> AC-3). Most unsolicited messages must be specifically enabled by setting the appropriate bit in the application's manager, enabling autodetection, or enabling other host notification capabilities.

6.2.6.1 Writing a Byte in Motorola Mode

Information provided in this section is intended as a functional description of how to write control information to DSPAB. The system designer must ensure that all of the timing constraints of the Motorola Parallel Host Mode Write Cycle are met.

The flow diagram shown in Figure 38 illustrates the sequence of events that define a one-byte write in Motorola mode. The protocol presented in Figure 38 will now be described in detail.

1) The host must drive the FA1 and FA0 register address pins of DSPAB with the address of the

address of the desired Parallel I/O Register. The address must be maintained for the duration of the read cycle.

Host Message: FA[1:0]==00b.

Host Control: FA[1:0]==01b.

PCMDATA: FA[1:0]==10b.

CMPDATA: FA[1:0]==11b.

2) The host indicates that this is a write cycle by driving the FR/$\overline{W}$ pin low.

3) The host initiates a write cycle by driving the $\overline{FCS}$ and $\overline{FDS}$ pins low.

4) The host drives the data byte to the FDAT[7:0] pins of DSPAB.

5) Once the setup time for the write has been met, the host ends the write cycle by driving the $\overline{FCS}$ and $\overline{FDS}$ pins high.

6.2.6.2 Reading a Byte in Motorola Mode

The flow diagram shown in Figure 39, "Motorola Mode, One-Byte Read Flow Diagram for DSPAB" on page 54 illustrates the sequence of events that define a one-byte read in Motorola mode. The protocol presented in Figure 39 will now be described in detail.

1) The host must drive the FA1 and FA0 register address pins of DSPAB with the address of the desired Parallel I/O Register. Note that only the Host Message register and the Host Control register can be read. The address must be maintained for the duration of the read cycle.

Host Message: FA[1:0]==00b.

Host Control: FA[1:0]==01b.

2) The host indicates that this is a read cycle by driving the FR/$\overline{W}$ pin high.

3) The host initiates the read cycle by driving the $\overline{FCS}$ and $\overline{FDS}$ pins low (bus must be tri-stated by this time).

4) Once the data is valid (after waiting the appropriate time specified in the timing specifications), the host can read the value of the selected register from the FDAT[7:0] pins of DSPAB.

5) The host should now terminate the read cycle by driving the $\overline{FCS}$ and $\overline{FDS}$ pins high.

FR/W (LOW)
ADDRESS A PARALLEL I/O REGISTER
(FA[1:0] SET APPROPRIATELY)

↓

FCS (LOW)
FDS (LOW)

↓

WRITE BYTE TO
FDAT[7:0]

↓

FCS (HIGH)
FDS (HIGH)
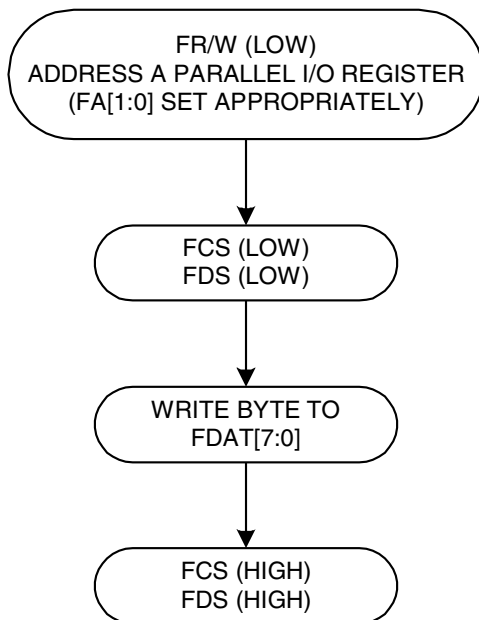
Figure 38. Motorola Mode, One-Byte Write Flow Diagram for DSPAB

FR/W (HIGH)
ADDRESS A PARALLEL I/O REGISTER
(FA[1:0] SET APPROPRIATELY)

↓

FCS (LOW)
FDS (LOW)

↓

READ BYTE FROM
FDAT[7:0]

↓

FCS (HIGH)
FDS (HIGH)
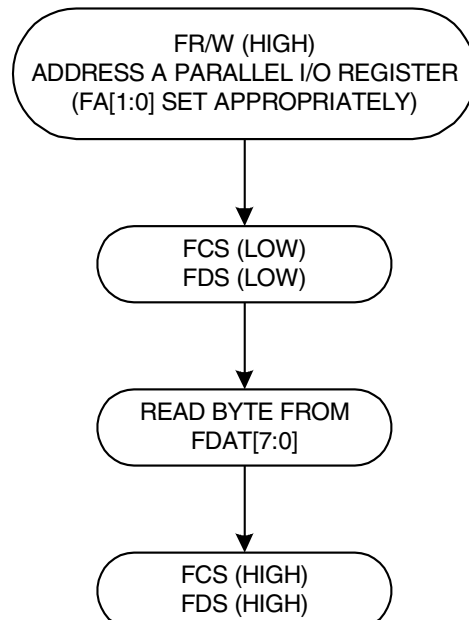
Figure 39. Motorola Mode, One-Byte Read Flow Diagram for DSPAB

6.2.7 Procedures for Parallel Host Mode Communication for DSPAB

6.2.7.1 Control Write in a Parallel Host Mode for DSPAB

When writing control data to DSPAB, the same protocol is used whether the host is writing a control message or an entire executable download image. Messages sent to DSPAB should be written most significant byte first. Likewise, downloads of the application code should also be performed most significant byte first.

The example shown in this section can be generalized to fit any control write situation. The generic function 'Read_Byte_*()' is used in the following example as a generalized reference to either Read_Byte_MOT() (read a byte in Motorola mode) or Read_Byte_INT() (read a byte in Intel mode), and 'Write_Byte_*()' is a generic reference to Write_Byte_MOT() (write a byte in Motorola mode) or Write_Byte_INT() (write a byte in Intel mode). Figure 40 shows a typical write sequence. The protocol presented in Figure 40 will now be described in detail.

1) When the host is communicating with DSPAB, the host must verify that DSPAB is ready to accept a new control byte. If DSPAB has not read the previous byte from the control port, it will be unable to receive another byte.

2) In order to determine whether DSPAB is ready to accept a new control byte the host must read the HINBSY bit of the Host Control Register (bit 2 in FA[1:0]=01b) using the selected communication mode (Intel or Motorola). If HINBSY is high, then the DSP is not prepared to accept a new control byte, and the host should poll the Host Control Register again. If HINBSY is low, then the host may write a control byte into the Host Message Register.

3) Once the host knows that the DSP is ready for a new control byte, it should write the control byte to the Host Message Register (FA[1:0] =

00b) using the selected communication mode (Intel or Motorola).

4) If the host would like to write any more control bytes to DSPAB, the host should once again poll the Host Control Register (return to step 1).

6.2.7.2 Control Read in a Parallel Host Mode for DSPAB

When reading control data from DSPAB, the same protocol is used whether the host is reading a single byte, a 6 byte message, or a string of messages.

During the boot procedure, a handshaking protocol is used by DSPAB. This handshake consists of a 3 byte write to DSPAB followed by a 1 byte response from the DSP. The host must read the response byte and act accordingly. The boot procedure is discussed in Section 8 "Boot Procedure" on page 71.

During regular operation (at run-time), the responses from DSPAB will always be 6 bytes in length.
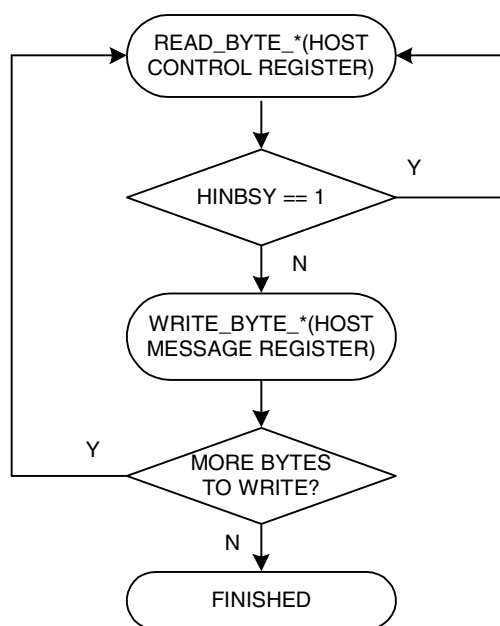
Figure 40. Typical Parallel Host Mode Control Write Sequence Flow Diagram for DSPAB

The example shown in this section can be used for any control read situation. The generic function 'Read_Byte_*()' is used in the following example as a generalized reference to either Read_Byte_MOT() (Motorola byte read) or Read_Byte_INT() (Intel byte read). Figure 41 shows a typical read sequence. The protocol presented in Figure 41 will now be described in detail.
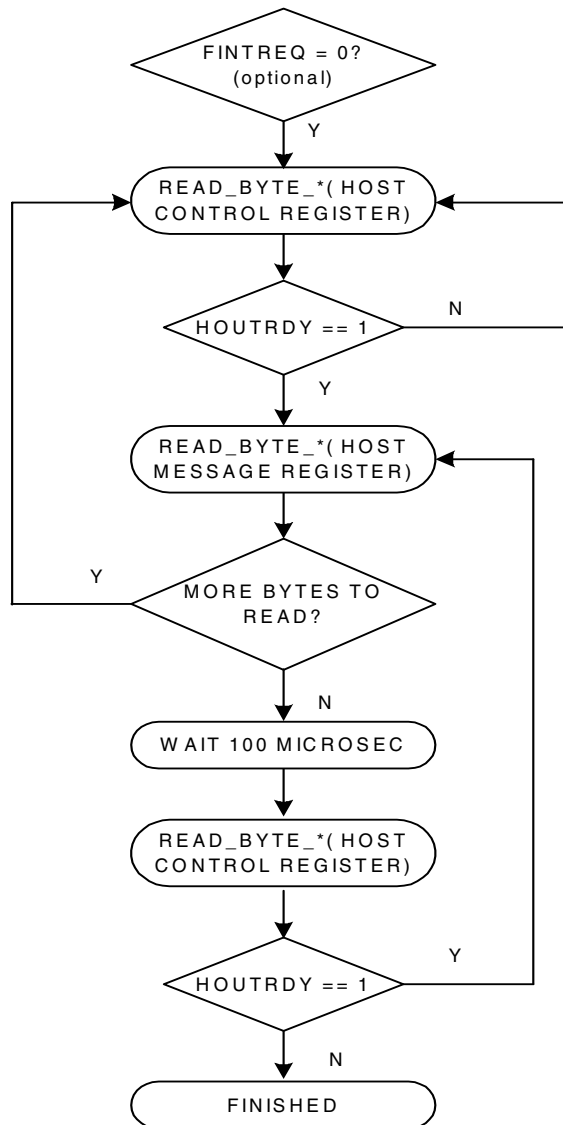
Figure 41. Typical Parallel Host Mode Control Read Sequence Flow Diagram for DSPAB

1) Optionally, $\overline{\text{FINTREQ}}$ going low may be used as an interrupt to the host to indicate that

DSPAB has an outgoing message. Note that even with the use of $\overline{\text{FINTREQ}}$, HOUTRDY must be checked to ensure that the current byte is ready to be read by the host during the read process. Please note that the state of $\overline{\text{FINTREQ}}$ is undefined during boot, and it is valid only once the application code is loaded.

2) The host reads the Host Control Register (FA[1:0] = 01b) in order to determine the state of the communication interface.

3) In order to determine whether DSPAB has an outgoing control byte that is valid, the host must check the HOUTRDY bit of the Host Control Register (bit 1, FA[1:0] = 01b). If HOUTRDY is high, then the Host Message Register contains a valid message byte for the host. If HOUTRDY is low, then the DSP has not placed a new control byte in the Host Message Register, and the host should poll the Host Control Register again.

4) Once the host knows that the DSP is ready to provide a new response byte, the host can safely read a byte from the Host Message Register (FA[1:0] = 00b) using the appropriate communication protocol (Motorola or Intel).

5) If the host expects to read any more response bytes, the host should once again check the HOUTRDY bit (return to step 1). Messages from the application code (post-download) on the DSP will always be 6 bytes, unless otherwise stated in the associated Application Note.

6) After the response has been read the host should wait at least 100 uS and check HOUTRDY one final time. If HOUTRDY is high once again this means that an unsolicited message has come during the read process and the host has another message to read (i.e. skip back to step 4 and read out the new message). It is the host's responsibility to insure that any pending messages are read from the DSP.

Failure to do this may cause the DSP's output message buffer to overflow, corrupting any pending outbound messages.

6.3 Parallel Host Communication for DSPC

The parallel host communication modes of DSPC provide an 8-bit interface to the DSP. An Intel-style parallel mode and a Motorola-style parallel mode are supported. The host interface is implemented using four communication registers within DSPC as shown in Table 8, "Parallel Input/Output Registers for DSPC," on page 58. The HOSTMSG register is a 32 bit register. Since there are only eight data pins, only one byte can be transferred at a time, but a full 32 bits are transferred in each read or write to this register.

When the host is downloading code or configuring the application code in DSPC, control messages must be written to (and read from) the Host Message register 32 bits at a time. The HINBSY pin and the HINBSY bit in the Host Control register are used during messaging sessions to determine when DSPC can accept another 32-bit word of control data. The HINBSY pin and the HINBSY bit (in the Host Control register) go high when the host writes 32 bits (4 bytes) to the HOSTMSG register. The HINBSY pin and HINBSY bit go low when DSPC reads the 32 bit data from the register. The INTREQ pin goes low and the HOUTRDY bit (in the Host Control register) goes high when DSPC has an message that must be read by the host.

The HOSTDATA1 and HOSTDATA2 registers are used strictly for the transfer of audio data to DSPC. The host cannot read from these two registers. Support of these registers is DSPC application code specific, see the appropriate Application Note for availability.

A detailed description for each parallel host mode will now be given. The following information will be provided for the Intel mode and Motorola mode:

- The pins of DSPC which must be used for proper communication

- Flow diagram and description for a parallel byte write

- Flow diagram and description for a 32-bit word (4-byte) write

- Flow diagram and description for a parallel byte read

- Flow diagram and description for a 32-bit word (4-byte) read

The four registers of DSPC's parallel host mode are not used identically. The algorithm used for communicating with each register will be given as a functional description, building upon the basic read and write protocols defined in the Motorola and Intel sections. The following will be covered:

- Flow diagram and description for a control write

- Flow diagram and description for a control read

6.3.1 Host Message (HOSTMSG) Register, A[1:0] = 00b

31	30	29	28	27	26	25	24
MS_BYTE							
23	22	21	20	19	18	17	16
BYTE_B							
15	14	13	12	11	10	9	8
BYTE_A							
7	6	5	4	3	2	1	0
LS_BYTE							

The HOSTMSG register is a 32-bit register that is used to read from or write to DSPC. It is read and written in 4-byte transactions. This register passes all the code, control and configuration data to and from DSPC. Messages are read and written MS_BYTE first.

MS_BYTE   This is the most significant byte (bits 31:24).

BYTE_B   This is BYTE_B (bits 23:16).

BYTE_A   This is BYTE_A (bits 15:8).

LS_BYTE   This is the least significant byte (bits 7:0).

6.3.2 Host Control (CONTROL) Register, A[1:0] = 01b

7	6	5	4	3	2	1	0
Reserved	Reserved	Reserved	BYTE_SEL		HINBSY	HOUTRDY	Reserved

Reserved   Always write a 0.

BYTE_SEL   Always write a 11b to these bits.

HINBSY   This bit is the Host Input Ready signal. It is set when the host writes to the HOSTMSG register. It is cleared when DSPC reads the HOSTMSG register. This bit is also pinned out on the HINBSY pin. Read only by the host and DSPC.

HOUTRDY   This bit is set when DSPC writes data to the HOST_MSG register, and indicates that the DSP has a pending message for the host. The HOUTRDY bit is cleared when the host reads the HOSTMSG register. The bit is inverted and pinned out on the INTREQ pin. Read only by the host and DSPC

6.3.3 Host Data1 Input (HOSTDATA1) Register, A[1:0] = 10b

7	6	5	4	3	2	1	0
HOSTDATA1_7	HOSTDATA1_6	HOSTDATA1_5	HOSTDATA1_4	HOSTDATA1_3	HOSTDATA1_2	HOSTDATA1_1	HOSTDATA1_0

HOSTDATA1_7–0 The host writes data to DSPC at this address. (Write only)

6.3.4 Host Data2 Input (HOSTDATA2) Register, A[1:0] = 11b

7	6	5	4	3	2	1	0
HOSTDATA2_7	HOSTDATA2_6	HOSTDATA2_5	HOSTDATA2_4	HOSTDATA2_3	HOSTDATA2_2	HOSTDATA2_1	HOSTDATA2_0

HOSTDATA2_7–0 The host writes data to DSPC at this address. (Write only)

Table 8. Parallel Input/Output Registers for DSPC

6.3.5 Intel Parallel Host Communication Mode for DSPC

The Intel parallel host communication mode is implemented using the pins given in Table 9. Parallel host communication is available only on the 144-pin package version of the CS49400.

The $\overline{\text{INTREQ}}$ pin is asserted (driven low) whenever the DSP has an outgoing message for the host. This same information is reflected by the HOUTRDY bit of the Host Control Register (A[1:0] = 01b).

$\overline{\text{INTREQ}}$ is useful for informing the host of unsolicited messages. An unsolicited message is defined as a message generated by the DSP without an associated host read request.

Mnemonic	Pin Name	144-Pin Package, Pin Number
Chip Select	$\overline{\text{CS}}$	129
Write Enable	$\overline{\text{WR}}$	120
Output Enable	$\overline{\text{RD}}$	121
Register Address Bit 1	A1	139
Register Address Bit 0	A0	130
Interrupt Request	$\overline{\text{INTREQ}}$	3
Host Busy	$\overline{\text{HINBSY}}$	141
DATA7	HDAT7	95
DATA6	HDAT6	96
DATA5	HDAT5	97
DATA4	HDAT4	103
DATA3	HDAT3	105
DATA2	HDAT2	112
DATA1	HDAT1	115
DATA0	HDAT0	116

Table 9. Intel Mode Communication Signals for DSPC

6.3.5.1 Writing a Byte in Intel Mode for DSPC

Information provided in this section is intended as a functional description of how to write control information to DSPC. The system designer must ensure that all of the timing constraints of the Intel Parallel Host Mode Write Cycle are met.

Figure 42. Intel Mode, One-Byte Write Flow Diagram for DSPC

The flow diagram shown in Figure 42 illustrates the sequence of events that define a one-byte write in Intel mode. One-byte writes should only be perfomed to the Host Control and Host Data registers. The protocol presented in Figure 42 will now be described in detail.

1) The host must first drive the A1 and A0 register address pins of DSPC with the address of the desired Parallel I/O Register. The address is latched on the falling edge of $\overline{\text{CS}}$.

   Host Control: A[1:0]==01b.

   Host Data1:   A[1:0]==10b.

   Host Data2:   A[1:0]==11b.

2) The host initiates a write cycle by driving the $\overline{\text{CS}}$ and $\overline{\text{WR}}$ pins low.

3) The host drives the data byte to the HDAT[7:0] pins of DSPC.

4) Once the setup time for the write has been met, the host ends the write cycle by driving the $\overline{\text{WR}}$ and $\overline{\text{CS}}$ pins high.

6.3.5.2 Writing a 32-bit (4-byte) Word in Intel Mode for DSPC

Information provided in this section is intended as a functional description of how to write control information to DSPC. The system designer must ensure that all of the timing constraints of the Intel Parallel Host Mode Write Cycle are met.

The flow diagram shown in Figure 43 illustrates the sequence of events that define a 4-byte write in Intel mode. 32-bit (4-byte) writes should only be used to write to the Host Message register. The protocol presented in Figure 43 will now be described in detail.

1) The host must first drive the A1 and A0 register address pins of DSPC with the address of the desired Parallel I/O Register (both low for A1=A0=0). The address must be maintained for the duration of the write cycle, and is latched on the falling edge of $\overline{CS}$ and $\overline{WR}$.

   Host Message: A[1:0]==00b.

2) The host initiates a write cycle by driving the $\overline{CS}$ and $\overline{WR}$ pins low.

3) The host drives the most significant data byte (bits 31:24) to the HDAT[7:0] pins of DSPC.

4) Once the setup time for the write has been met, the host drives $\overline{WR}$ high to strobe in the most significant data byte.

5) The host drives $\overline{WR}$ low.

6) The host drives the next most significant data byte, BYTE_B (bits 23:16), to the HDAT[7:0] pins of DSPC.

7) Once the setup time for the write has been met, the host drives $\overline{WR}$ high to strobe in the data byte.

8) The host drives $\overline{WR}$ low.

9) The host drives the next most significant data byte, BYTE_A (bits 15:8), to the HDAT[7:0] pins of DSPC.
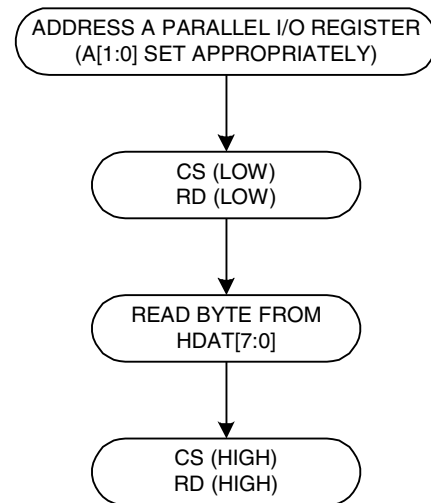
Figure 44. Intel Mode, One-Byte Read Flow Diagram for DSPC

10) Once the setup time for the write has been met, the host drives $\overline{WR}$ high to strobe in the data byte.

11) The host drives $\overline{WR}$ low.

12) The host drives the least significant data byte (bits 7:0) to the HDAT[7:0] pins of DSPC.

13) Once the setup time for the write has been met, the host drives $\overline{WR}$ high to strobe in the data byte.

14) The host drives $\overline{CS}$ high to end the write transaction.

6.3.5.3 Reading a Byte in Intel Mode for DSPC

Information provided in this section is intended as a functional description of how to read control information from DSPC. The system designer must ensure that all of the timing constraints of the Intel Parallel Host Mode Read Cycle are met.

The flow diagram shown in Figure 44 illustrates the sequence of events that define a one-byte read in Intel mode. One-byte reads should only be done with the Host Control register. The protocol presented in Figure 44 will now be described in detail.
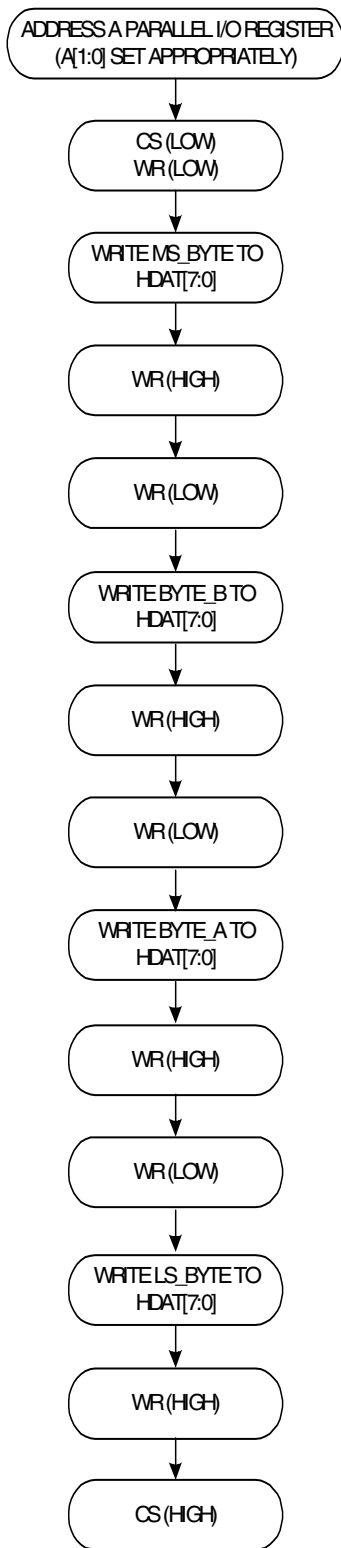
Figure 43. Intel Mode, 32-bit (4-byte) Write Flow Diagram for DSPC

1) The host must first drive the A1 and A0 register address pins of DSPC with the address of the desired Parallel I/O Register (A1=0, A0=1). Note that only the Host Control register can be read using a byte read. The address must be maintained for the duration of the read cycle, and is latched on the falling edge of $\overline{CS}$.

   Host Control: A[1:0]==01b.

2) The host initiates a read cycle by driving the $\overline{CS}$ and $\overline{RD}$ pins low (bus must be tri-stated by this time).

3) Once the data is valid (after waiting the appropriate time specified in the timing specifications), the host can read the value of the selected register from the HDAT[7:0] pins of DSPC.

4) The host should now terminate the read cycle by driving the $\overline{CS}$ and $\overline{RD}$ pins high.

6.3.5.4 Reading a 32-bit (4-byte) Word from DSPC in Intel Mode

Information provided in this section is intended as a functional description of how to read control information from DSPC. The system designer must ensure that all of the timing constraints of the Intel Parallel Host Mode Read Cycle are met.

The flow diagram shown in Figure 45, "Intel Mode, 32-Bit (4-Byte) Read Flow Diagram for DSPC" on page 62 illustrates the sequence of events that define a 4-byte read in Intel mode. 4-byte (32-bit) reads should only be done with the Host Message register. The protocol presented in Figure 45 will now be described in detail.

1) The host must first drive the A1 and A0 register address pins of DSPC with the address of the desired Parallel I/O Register (A1=A0=0). Note that only the Host Message register can be read using a 4-byte read cycle. The address must be maintained for the duration of the read cycle, and is latched on the falling edge of $\overline{CS}$.

   Host Message: A[1:0]==00b.

ADDRESS A PARALLEL I/O REGISTER
(A[1:0] SET APPROPRIATELY)

CS (LOW)
RD (LOW)

READ MS_BYTE FROM
HDAT[7:0]

RD (HIGH)

RD (LOW)

READ BYTE_B FROM
HDAT[7:0]

RD (HIGH)

RD (LOW)

READ BYTE_A FROM
HDAT[7:0]

RD (HIGH)

RD (LOW)

READ LS_BYTE FROM
HDAT[7:0]

RD (HIGH)

CS (HIGH)
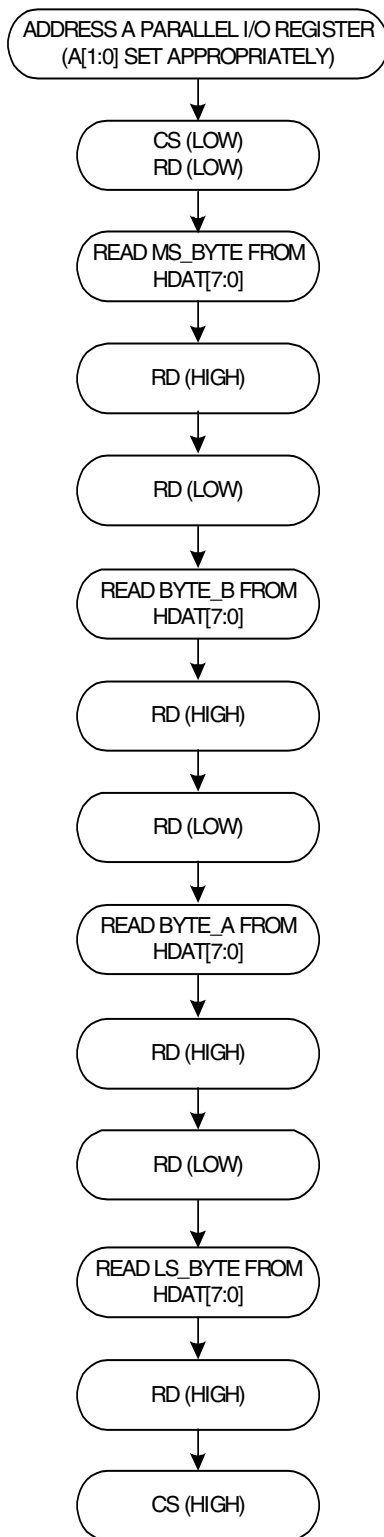
Figure 45. Intel Mode, 32-Bit (4-Byte) Read Flow Diagram for DSPC

2) The host initiates a read cycle by driving the $\overline{CS}$ and $\overline{RD}$ pins low (bus must be tri-stated by this time).

3) Once the data is valid (after waiting the appropriate time specified in the timing specifications), the host can read the most significant data byte (bits 31:24) from the HDAT[7:0] pins of DSPC.

4) The host drives $\overline{RD}$ high to indicate that the data byte has been read.

5) The host drives $\overline{RD}$ low to clock out the next data byte.

6) Once the data is valid (after waiting the appropriate time specified in the timing specifications), the host can read BYTE_B (bits 23:16) from the HDAT[7:0] pins of DSPC.

7) The host drives $\overline{RD}$ high to indicate that the data byte has been read.

8) The host drives $\overline{RD}$ low to clock out the next data byte.

9) Once the data is valid (after waiting the appropriate time specified in the timing specifications), the host can read BYTE_A (bits 15:8) from the HDAT[7:0] pins of DSPC.

10) The host drives $\overline{RD}$ high to indicate that the data byte has been read.

11) The host drives $\overline{RD}$ low to clock out the next data byte.

12) Once the data is valid (after waiting the appropriate time specified in the timing specifications), the host can read the least significant data byte (bits 7:0) from the HDAT[7:0] pins of DSPC.

13) The host drives $\overline{RD}$ high to indicate that the data byte has been read.

14) The host should now terminate the read cycle by driving the $\overline{CS}$ pin high.

6.3.6 Motorola Parallel Host Communication Mode for DSPC

The Motorola parallel host communication mode is implemented using the pins given in Table 10.

Mnemonic	Pin Name	144-Pin Package, Pin Number
Chip Select	$\overline{CS}$	129
Data Strobe	$\overline{DS}$	120
Read or Write Select	R/$\overline{W}$	121
Register Address Bit 1	A1	139
Register Address Bit 0	A0	130
Interrupt Request	$\overline{INTREQ}$	3
Host Busy	$\overline{HINBSY}$	141
DATA7	HDAT7	95
DATA6	HDAT6	96
DATA5	HDAT5	96
DATA4	HDAT4	103
DATA3	HDAT3	105
DATA2	HDAT2	112
DATA1	HDAT1	115
DATA0	HDAT0	116

Table 10. Motorola Mode Communication Signals for DSPC

Parallel host communication is available only on the 144-pin package part. The $\overline{INTREQ}$ pin is asserted (driven low) whenever the DSP has an outgoing message for the host. This same information is reflected by the HOUTRDY bit of the Host Control Register (A[1:0] = 01b).

$\overline{INTREQ}$ is useful for informing the host of unsolicited messages. An unsolicited message is defined as a message generated by DSPC without an associated host read request.
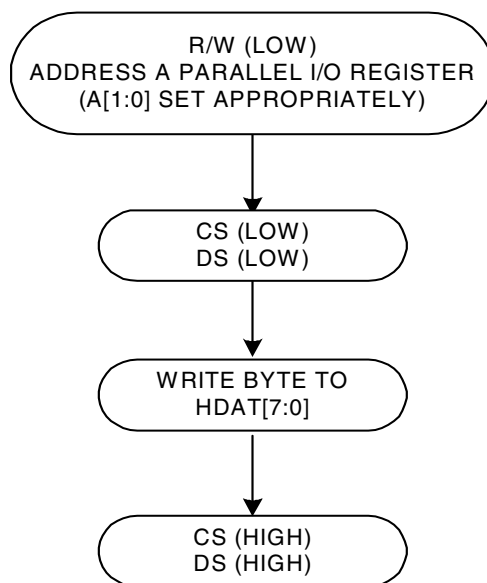
Figure 46. Motorola Mode, One-Byte Write Flow Diagram for DSPC

6.3.6.1 Writing a Byte in Motorola Mode for DSPC

Information provided in this section is intended as a functional description of how to write control information to DSPC. The system designer must ensure that all of the timing constraints of the Motorola Parallel Host Mode Write Cycle are met.

The flow diagram shown in Figure 46 illustrates the sequence of events that define a one-byte write in Motorola mode. One byte writes should only be used with the Host Control and Host Data registers. The protocol presented in Figure 46 will now be described in detail.

1) The host must first drive the A1 and A0 register address pins of DSPC with the address of the desired Parallel I/O Register. The address is latched on the falling edge of $\overline{CS}$.

   Host Control: A[1:0]==01b.

   Host Data1: A[1:0]==10b.

   Host Data2: A[1:0]==11b.

2) The host indicates that this is a write cycle by driving the R/$\overline{W}$ pin low.

3) The host initiates a write cycle by driving the $\overline{\text{CS}}$ and $\overline{\text{DS}}$ pins low.

4) The host drives the data byte to the HDAT[7:0] pins of DSPC.

5) Once the setup time for the write has been met, the host ends the write cycle by driving the $\overline{\text{DS}}$ and $\overline{\text{CS}}$ pins high.

6.3.6.2 Writing a 32-bit (4-byte) Word in Motorola Mode for DSPC

Information provided in this section is intended as a functional description of how to write control information to DSPC. The system designer must ensure that all of the timing constraints of the Motorola Parallel Host Mode Write Cycle are met.
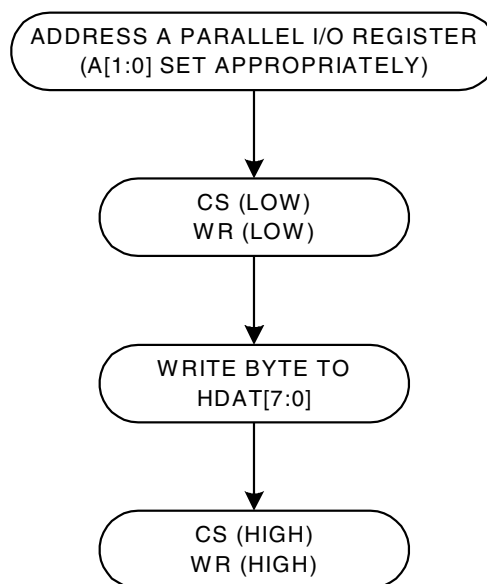
The flow diagram shown in Figure 43 illustrates the sequence of events that define a 32-bit (4-byte) write in Motorola mode. 32-bit (4-byte) writes should only be done to the Host Message register. The protocol presented in Figure 43 will now be described in detail.

1) The host must first drive the A1 and A0 register address pins of DSPC with the address of the desired parallel I/O register (A1=A0=0). The address must be maintained for the duration of the write cycle, and is latched on the falling edge of $\overline{\text{CS}}$.

   Host Message: A[1:0]==00b.

2) The host indicates that this is a write cycle by driving the R/$\overline{\text{W}}$ pin low.

3) The host initiates a write cycle by driving the $\overline{\text{CS}}$ and $\overline{\text{DS}}$ pins low.

4) The host drives the most significant data byte (bits 31:24) to the HDAT[7:0] pins of DSPC.

5) Once the setup time for the data has been met, the host latches this byte in by driving $\overline{\text{DS}}$ high.

6) The host drive $\overline{\text{DS}}$ low.

7) The host drives the next most significant data byte, BYTE_B (bits 23:16), to the HDAT[7:0]
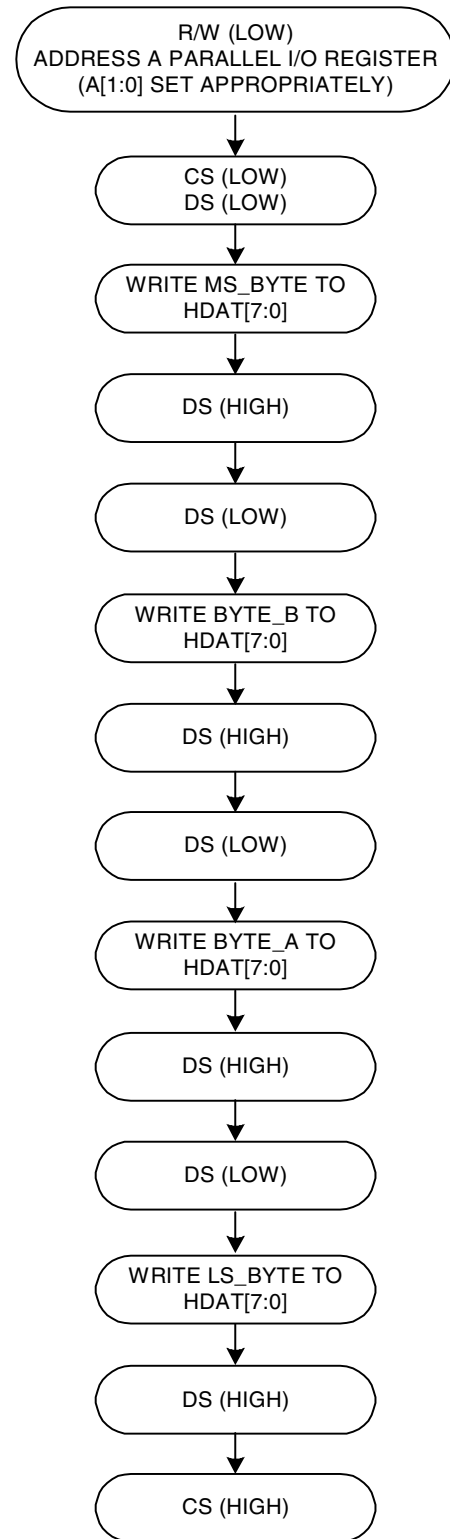
Figure 47. Motorola Mode, 32-bit (4-byte) Write Flow Diagram for DSPC

pins of DSPC.

8) Once the setup time for the data has been met, the host latches this byte in by driving $\overline{DS}$ high.

9) The host drive $\overline{DS}$ low.

10) The host drives the next most significant data byte, BYTE_A (bits 15:8), to the HDAT[7:0] pins of DSPC.

11) Once the setup time for the data has been met, the host latches this byte in by driving $\overline{DS}$ high.

12) The host drive $\overline{DS}$ low.

13) The host drives the least significant data byte (bits 7:0) to the HDAT[7:0] pins of DSPC.

14) Once the setup time for the data has been met, the host latches this byte in by driving $\overline{DS}$ high.

15) The host ends the write cycle by driving the $\overline{CS}$ pin high.

6.3.6.3 Reading a Byte in Motorola Mode for DSPC

Information provided in this section is intended as a functional description of how to write control information to DSPC. The system designer must ensure that all of the timing constraints of the Motorola Parallel Host Mode Read Cycle are met.

The flow diagram shown in Figure 48 illustrates the sequence of events that define a one-byte read in Motorola mode. Single byte reads should only be done with the Host Control register. The protocol presented in Figure 48 will now be described in detail.

1) The host must first drive the A1 and A0 register address pins of DSPC with the address of the desired Parallel I/O Register (A1=0, A0=1). The address must be maintained for the duration of the read cycle, and is latched on the falling edge of $\overline{CS}$.

   Host Control: A[1:0]==01b.

2) The host indicates that this is a read cycle by driving the R/$\overline{W}$ pin high.
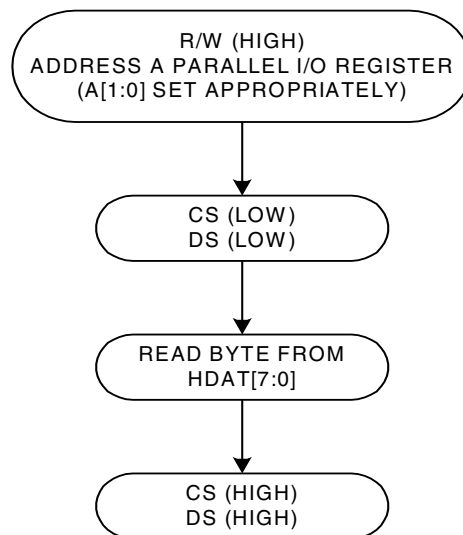
Figure 48. Motorola Mode, One-Byte Read Flow Diagram for DSPC

3) The host initiates a read cycle by driving the $\overline{CS}$ and $\overline{DS}$ pins low (bus must be tri-stated by this time).

4) Once the data is valid (after waiting the appropriate time specified in the timing specifications), the host can read the value of the selected register from the HDAT[7:0] pins of DSPC.

5) The host should now terminate the read cycle by driving the $\overline{CS}$ and $\overline{DS}$ pins high.

6.3.6.4 Reading a 32-bit (4-byte) word from DSPC in Motorola mode

Information provided in this section is intended as a functional description of how to read control information from DSPC. The system designer must ensure that all of the timing constraints of the Motorola Parallel Host Mode Read Cycle are met.

The flow diagram shown in Figure 49, "Motorola Mode, 32-Bit (4-Byte) Read Flow Diagram for DSPC" on page 66 illustrates the sequence of events that define a 32-bit (4-byte) read in Motorola mode. Reading a 32-bit (4-byte) word should only be done with the Host Message

CIRRUS LOGIC®

```
        R/W (HIGH)
ADDRESS A PARALLEL I/O REGISTER
   (A[1:0] SET APPROPRIATELY)
              │
              ▼
          CS (LOW)
          DS (LOW)
              │
              ▼
     READ MS_BYTE TO
       HDAT[7:0]
              │
              ▼
         DS (HIGH)
              │
              ▼
         DS (LOW)
              │
              ▼
     READ BYTE_B TO
       HDAT[7:0]
              │
              ▼
         DS (HIGH)
              │
              ▼
         DS (LOW)
              │
              ▼
     READ BYTE_A TO
       HDAT[7:0]
              │
              ▼
         DS (HIGH)
              │
              ▼
         DS (LOW)
              │
              ▼
     READ LS_BYTE TO
       HDAT[7:0]
              │
              ▼
         DS (HIGH)
              │
              ▼
          CS (HIGH)
```
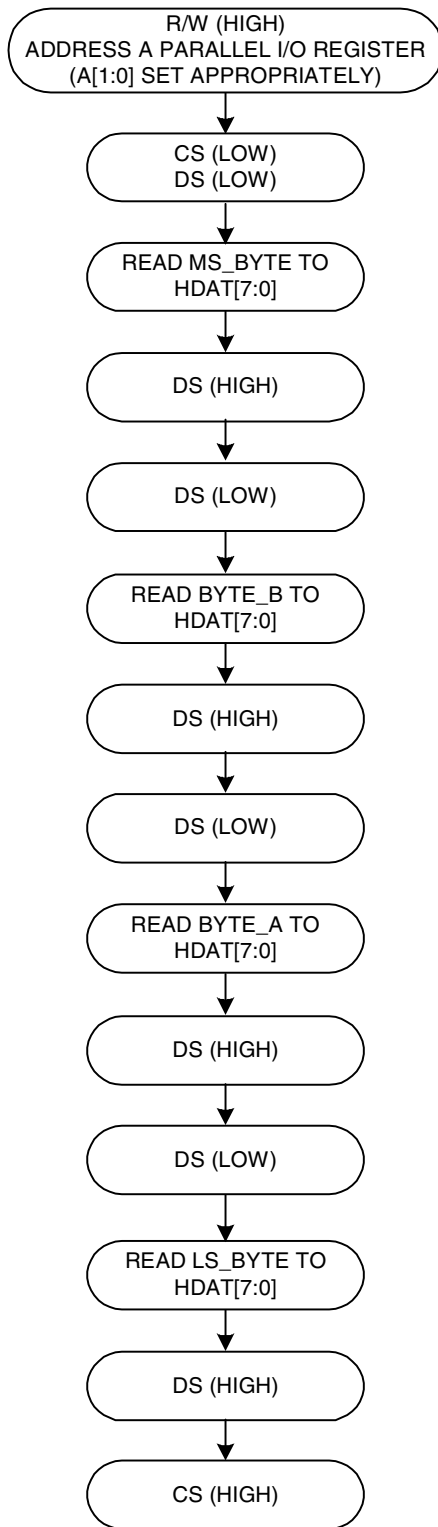
Figure 49. Motorola Mode, 32-Bit (4-Byte) Read Flow Diagram for DSPC

register. The protocol presented in Figure 49 will now be described in detail.

1) The host must first drive the A1 and A0 register address pins of DSPC with the address of the desired Parallel I/O Register (A1=A0=0). Note that only the Host Message register can be read using 4-byte reads. The address must be maintained for the duration of the read cycle, and is latched on the falling edge of $\overline{CS}$.

   Host Message: A[1:0]==00b.

2) The host indicates that this is a read cycle by driving the R/$\overline{W}$ pin high.

3) The host initiates a read cycle by driving the $\overline{CS}$ and $\overline{DS}$ pins low (bus must be tri-stated by this time).

4) Once the data is valid (after waiting the appropriate time specified in the timing specifications), the host can read the most significant byte (bits 31:24) from the HDAT[7:0] pins of DSPC.

5) The host indicates the byte has been read by driving $\overline{DS}$ high.

6) The host latches out the next byte by driving $\overline{DS}$ low.

7) Once the data is valid (after waiting the appropriate time specified in the timing specifications), the host can read the next most significant byte, BYTE_B (bits 23:16), of the selected register from the HDAT[7:0] pins of DSPC.

8) The host indicates the byte has been read by driving $\overline{DS}$ high.

9) The host latches out the next byte by driving $\overline{DS}$ low.

10) Once the data is valid (after waiting the appropriate time specified in the timing specifications), the host can read the next most significant byte, BYTE_A (bits 15:8), of the selected register from the HDAT[7:0] pins of

DSPC.

11) The host indicates the byte has been read by driving $\overline{DS}$ high.

12) The host latches out the next byte by driving $\overline{DS}$ low.

13) Once the data is valid (after waiting the appropriate time specified in the timing specifications), the host can read the least significant byte (bits 7:0) of the selected register from the HDAT[7:0] pins of DSPC.

14) The host indicates the byte has been read by driving $\overline{DS}$ high.

15) The host should now terminate the read cycle by driving the $\overline{CS}$ pin high.

### 6.3.7 Procedures for Parallel Host Mode Communication for DSPC

6.3.7.1 Control Write in a Parallel Host Mode for DSPC

When writing control data to DSPC, the same protocol is used whether the host is writing a control message or an entire executable download image. Messages sent to DSPC should be written most significant byte first. Likewise, downloads of the application code should also be performed most significant byte first.

The example shown in this section can be generalized to fit any control write situation. The generic function 'Read_Byte_*()' is used in the following example as a generalized reference to either Read_Byte_MOT() (read a byte in Motorola mode) or Read_Byte_INT() (read a byte in Intel mode), and 'Write_Byte_*()' is a generic reference to Write_Byte_MOT() (write a byte in Motorola mode) or Write_Byte_INT() (write a byte in Intel mode). Similarly, the generic function 'Read_Word_*()' is used in the following example as a generalized reference to either Read_Word_MOT() (read a 32-bit word in Motorola mode) or Read_Word_INT() (read a 32-bit word in Intel mode), and 'Write_Word_*()' is a

generic reference to Write_Word_MOT() (write a 32-bit word in Motorola mode) or Write_Word_INT() (write a 32-bit word in Intel mode). Figure 50 shows a typical write sequence. The protocol presented in Figure 50 will now be described in detail.

1) When the host is communicating with DSPC, the host must verify that DSPC is ready to accept a new 32-bit control word. If DSPC has not read the previous word from the control port, it will be unable to receive another word.

2) In order to determine whether DSPC is ready to accept a new 32-bit control word the host must read the HINBSY bit of the Host Control Register (bit 2 in FA[1:0]=01b) using the selected communication mode (Intel or Motorola). If HINBSY is high, then the DSP is not prepared to accept a new control word, and the host should poll the Host Control Register again. If HINBSY is low, then the host may write a control word (32-bits) into the Host
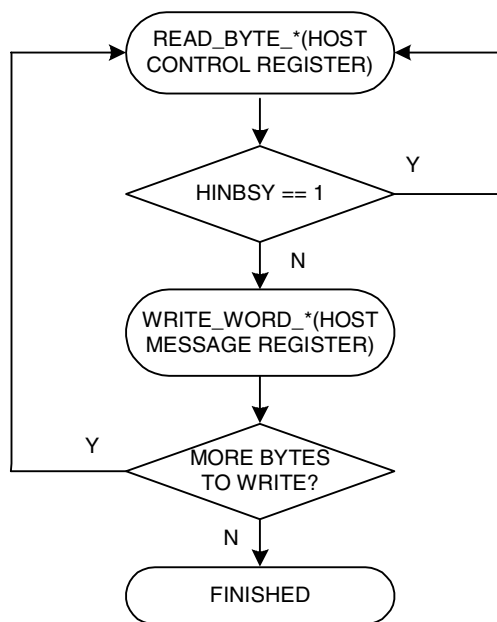
Figure 50. Typical Parallel Host Mode Control Write Sequence Flow Diagram for DSPC

Message Register.

3) Once the host knows that the DSP is ready for a new control word, it should write the 32-bit control word to the Host Message Register (FA[1:0] = 00b) using the selected communication mode (Intel or Motorola).

4) If the host would like to write any more 32-bit control words to DSPC, the host should once again poll the Host Control Register (return to step 1).

6.3.7.2 Control Read in a Parallel Host Mode for DSPC

When reading control data from DSPC, the same protocol is used whether the host is reading a single 32-bit word, or a string of message words.

Reads and writes to the Host Message Register are always 32-bits (4-bytes), and reads of the Host Control Register are always a single byte.

The example shown in this section can be used for any control read situation. The generic function 'Read_Word_*()' is used in the following example as a generalized reference to either Read_Word_MOT() (Motorola 32-bit word read) or Read_Word_INT() (Intel 32-bit word read). Figure 51 shows a typical read sequence. The protocol presented in Figure 51 will now be described in detail.

1) Optionally, $\overline{\text{INTREQ}}$ going low may be used as an interrupt to the host to indicate that DSPC has an outgoing message.

2) The host reads the Host Control Register (A[1:0] = 01b) in order to determine the state of the communication interface.

3) In order to determine whether DSPC has an outgoing control word that is valid, the host must check the HOUTRDY bit of the Host Control Register (bit 1, A[1:0] = 01b). If HOUTRDY is high, then the Host Message Register contains a valid 32-bit word for the host. If HOUTRDY is low, then the DSP has
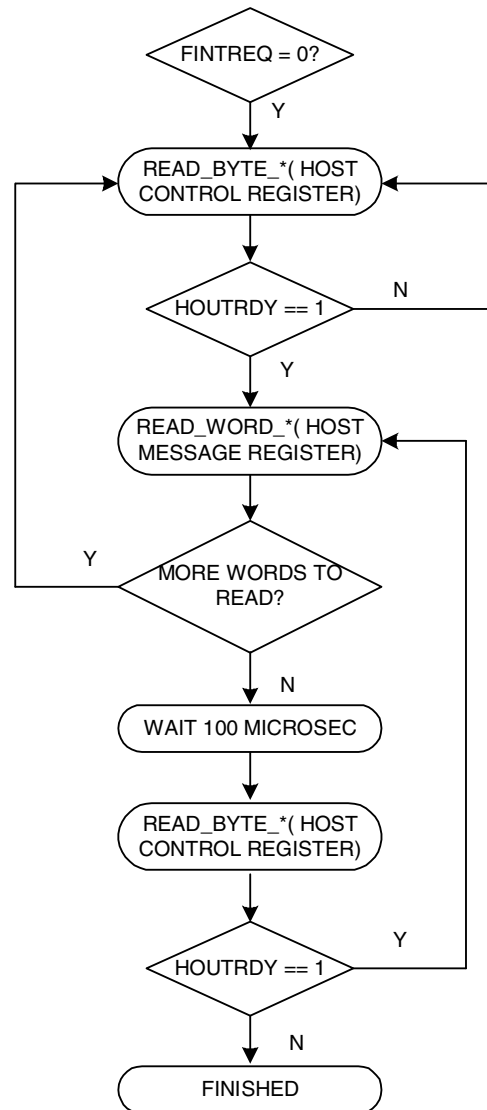
Figure 51. Typical Parallel Host Mode Control Read Sequence Flow Diagram for DSPC

not placed a new control word in the Host Message Register, and the host should poll the Host Control Register again.

4) Once the host knows that the DSP is ready to provide a new 32-bit response word, the host can safely read a word from the Host Message Register (A[1:0] = 00b) using the appropriate communication protocol (Motorola or Intel).

5) If the host expects to read more 32-bit response words, the host should once again check the

HOUTRDY bit (return to step 1).

6) After the response has been read the host should wait at least 100 uS and check HOUTRDY one final time. If HOUTRDY is high once again this means that an unsolicited message has come during the read process and the host has another message to read (i.e. skip back to step 4 and read out the new message). It is the host's responsibility to insure that any pending messages are read from the DSP. Failure to do this may cause the DSP's output message buffer to overflow, corrupting any pending outbound messages.

7. EXTERNAL MEMORY

The system designer has the option of using external memory. The external memory interface is implemented with two controllers. The SRAM controller allows the DSP to autoboot from a parallel FLASH or EEPROM device. The SRAM and SDRAM controllers are used to extend the data memory of the DSP during runtime. A system can use a FLASH/EEPROM device for autoboot, and either SRAM or SDRAM for runtime memory. The application user's guide for a particular code load will inform the system designer if memory is required, and will specify the memory type and speed. If no mention is made of external memory, then external memory is not required for that application. **The SDRAM interface is not available on the 100-pin device.**

The signals for the external memory interface are listed in Table 11. and Table 12.

For both controllers, memory access speed is controlled by the DSP clock setting which is constrained by the application code. The SRAM interface is capable of in-system programming a FLASH device. Wait states are available to support slower FLASH/EEPROM and SRAM devices. The SRAM interface supports 1Mx8 addressable space. We recommend 12nS or better SRAM for optimal performance. The SDRAM interface supports 16Mbit parts organized as 512k x 16bits x 2 banks which yields a 1Mx16 addressable space. The burst

Pin Name	Pin Description	144-Pin Number	100-Pin Number
EXTA0	SRAM Address 0	73	51
EXTA1	SRAM Address 1	74	52
EXTA2	SRAM Address 2	75	53
EXTA3	SRAM Address 3	76	54
EXTA4	SRAM Address 4	67	46
EXTA5	SRAM Address 5	66	45
EXTA6	SRAM Address 6	65	44
EXTA7	SRAM Address 7	63	43
EXTA8	SRAM Address 8	62	42
EXTA9	SRAM Address 9	60	41
EXTA10	SRAM Address 10	72	50
EXTA11	SRAM Address 11	56	40
EXTA12	SRAM Address 12	55	39
EXTA13	SRAM Address 13	54	38
EXTA14	SRAM Address 14	53	37
EXTA15	SRAM Address 15	52	36
EXTA16	SRAM Address 16	49	35
EXTA17	SRAM Address 17	47	34
EXTA18	SRAM Address 18	46	33
EXTA19	SRAM Address 19	71	N/A
EXTD0	SRAM Data 0	34	23
EXTD1	SRAM Data 1	35	24
EXTD2	SRAM Data 2	36	25
EXTD3	SRAM Data 3	37	26
EXTD4	SRAM Data 4	38	27
EXTD5	SRAM Data 5	40	28
EXTD6	SRAM Data 6	43	31
EXTD7	SRAM Data 7	44	32
NV_OE#	SRAM Output Enable	31	21
NV_WE#	SRAM Write Enable	71	N/A
NV_CS#	SRAM Chip Select	32	22

Table 11. SRAM Interface Pins

Pin Name	Pin Description	144-Pin Number
SD_DATA0	SDRAM Data 0	34
SD_DATA1	SDRAM Data 1	35
SD_DATA2	SDRAM Data 2	36
SD_DATA3	SDRAM Data 3	37
SD_DATA4	SDRAM Data 4	38
SD_DATA5	SDRAM Data 5	40
SD_DATA6	SDRAM Data 6	43
SD_DATA7	SDRAM Data 7	44
SD_DATA8	SDRAM Data 8	56
SD_DATA9	SDRAM Data 9	55
SD_DATA10	SDRAM Data 10	54
SD_DATA11	SDRAM Data 11	53
SD_DATA12	SDRAM Data 12	52
SD_DATA13	SDRAM Data 13	49
SD_DATA14	SDRAM Data 14	47

Table 12. SDRAM Interface Pins

Pin Name	Pin Description	144-Pin Number
SD_DATA15	SDRAM Data 15	46
SD_ADDR0	SDRAM Address 0	73
SD_ADDR1	SDRAM Address 1	74
SD_ADDR2	SDRAM Address 2	75
SD_ADDR3	SDRAM Address 3	76
SD_ADDR4	SDRAM Address 4	67
SD_ADDR5	SDRAM Address 5	66
SD_ADDR6	SDRAM Address 6	65
SD_ADDR7	SDRAM Address 7	63
SD_ADDR8	SDRAM Address 8	62
SD_ADDR9	SDRAM Address 9	60
SD_ADDR10	SDRAM Address 10	72
SD_DQM0	SDRAM Data Mask Output0	39
SD_WE#	SDRAM Write Enable	37
SD_CAS#	SDRAM Column Address Strobe	78
SD_RAS#	SDRAM Row Address Strobe	77
SD_CS#	SDRAM Chip Select	68
SD_BA	SDRAM Bank Select	71
SD_DQM1	SDRAM Data	45
SD_CLK_IN	SDRAM Clock Input	61
SD_CLK_OUT	SDRAM Clock Output	59
SD_CLK_EN	SDRAM Clock Enable	64

Table 12. SDRAM Interface Pins

length is fixed to 8. We recommed SDRAM with a CAS latency of 2 for optimal performance. The refresh rate and the mode register of the SDRAM must be set before Kickstart. Refer to Table 14, "SDRAM Config Register," on page 70 for details on the SDRAM config register. The SDRAM is initialized after Kickstart.

7.1 Configuring SRAM Timing Parameters

Since not all SRAM manufacturers conform to the exact same timing specifications, it is necessary to configure the DSP to match the timing specifications of the particular SRAM that is used in your design. This message must be sent before

Kickstarting the downloaded DSPC application code.

Mnemonic	Hex Message
SRAM_CONTROLLER_TIMING 0xaaa = 0www wwrr rrre (in binary) w = SRAM_FLASH_WR_CYCLE variable found in the SRAM Switching characteristics table in Section 1.19. r = SRAM_FLASH_RD_CYCLE variable found in the SRAM Switching characteristics table in Section 1.19. e = SRAM Enable/Disable = 0/1. *Default, aaa = 0x000*	0x8100000F 0x00000aaa

Table 13. SRAM Controller Timing

Mnemonic	Hex Message
SDRAM_CONFIG aaaa = Auto refresh setting. For example for a 16 µS refresh period and a DCLK of 86 Mhz the value programmed should be: $16 \times 10^{-6} \times 86 \times 10^{6} = 0x(1376) = 0x0560$ bbb = Mode register setting. These bits set the 12 least significant bits in the mode register. The bits are to the following by default: bits(2..0) = 011 = Burst length 8. bit3 = 0 = Sequential Burst Type. bits(6..4) = 010 = CAS latency of 2. bits(8..7) = 00 = Mode Register Set. bit9 = 0 = Write Burst. bit(12..10) = 000 = reserved e = SDRAM Enable/Disable =0001/0000. *Default, 0xaaaabbbe = 0x0560008c0*	0x81000017 0xaaaabbbe

Table 14. SDRAM Config Register

8. BOOT PROCEDURE

In this section the process of booting and downloading to the CS49400 will be covered as well as how to perform a soft reset. There are two ways to boot the DSP:

- Host Controlled MasterBoot
- Host Boot Via DSPC

Each of these boot procedures will be described in detail with pseudocode. The flow charts use the following messages:

- Write-C_* Write to DSPC
- Read-C_* Read from DSPC
- Write-AB_* Write to DSPAB
- Read-AB_* Read from DSPAB

> **Note:** When reading from DSPAB the host must wait for FINTREQ to fall before starting the read cycle. When reading from DSPC the host must wait for INTREQ to fall before starting the read cycle.
> **Note:** The * can be replaced by SPI, INTEL, or MOT depending on the mode of host communication. For each case the general download algorithm is the same.
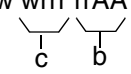
Table 16, and Table 17 define the boot write messages and boot read messages in mnemonic and actual hex value for DSPAB and DSPC. These messages will be used in the boot sequence

MNEMONIC	VALUE
AB_APP_START	0x03
AB_APP_FAILURE	0xF0

Table 15. Application Messages from DSPAB

MNEMONIC	VALUE
C_RESERVED	0x00 00 00 00
C_SOFT_RESET	0x40 00 00 00

Table 16. Boot Write Messages for DSPC

C_SLAVE_MODE	0x80 00 00 00
C_MASTER_BOOT_FLASH 1110 wwww wrrr rrAA c b AAAA AAAA AAAA AAAA Where w = SRAM_FLASH_WR_CYCLE variable found in the SRAM Switching characteristics table in Section 1.19. r = SRAM_FLASH_RD_CYCLE variable found in the SRAM Switching characteristics table in Section 1.19. A = 18-bit start address/4	0xEw cb AA AA

Table 16. Boot Write Messages for DSPC (Continued)

MNEMONIC	VALUE
C_BOOT_START	0x00 00 00 01
C_BOOT_SUCCESS	0x00 00 00 02
C_APP_START	0x00 00 00 04
C_BOOT_ERROR_CHECKSUM	0x00 00 00 FF
INVALID_BOOT_TYPE	0x00 00 00 FE

Table 17. Boot Read Messages from DSPC

8.1 Host Controlled Master Boot

A host controlled master boot is a sequence where a host instructs the DSP to load application code into itself from external memory. External memory can either be FLASH or SPI EEPROM.

The flow chart given in Figure 52, "Host Controlled Master Boot (Downloading both a DSPAB Application Code and a DSPC Application Code)" on page 72 demonstrates the interaction required by the microcontroller when placing the DSP into a host controlled master mode.

1) A download sequence is started when the host holds the mode pins appropriately (UHS[2:0] and FHS[1:0]) and toggles RESET.

2) The host must then send the C_MASTER_SOURCE_MODE boot message

START

RESET(LOW)

RESET(HIGH)

WAIT ?? uS

WRITE-C_*
(C_MASTER_SOURCE_MODE)

INTREQ LOW? — N → TIMEOUT AFTER (10 mS)

Y

READ-C_*(C_MESSAGE)

MSG ==C_BOOT_START — N → EXIT(ERROR)

Y

RELEASE CONTROL OF BUS

INTREQ LOW? — N → TIMEOUT AFTER (500 mS)

Y

READ-C_*(C_MESSAGE)

MSG== C_BOOT_SUCCESS — N → EXIT(ERROR)

Y

WRITE-C_* (C_MASTER_SOURCE_MODE)

INTREQ LOW? — N → TIMEOUT AFTER (10 mS)

Y

READ-C_*(C_MESSAGE)

MSG ==C_BOOT_START — N → EXIT(ERROR)

Y

RELEASE CONTROL OF BUS

INTREQ LOW? — N → TIMEOUT AFTER (500 mS)

Y

READ-C_*(C_MESSAGE)

MSG== C_BOOT_SUCCESS — N → EXIT(ERROR)

Y

WRITE-C_*(C_SOFT_RESET)

INTREQ LOW? — N → TIMEOUT AFTER (10 mS)

Y

READ-C_*(C_MESSAGE)

MSG== C_APP_START — N → EXIT(ERROR)

Y

WRITE-C_*(C_HW_CONFIG_MSG, MSG_SIZE)

WRITE-C_*(C_SW_CONFIG_MSG, MSG_SIZE)

WRITE-C_*(KICKSTART, MSG_SIZE)

C_APPLICATION_RUNNING

FINTREQ LOW? HOUTRDY HI? — N → TIMEOUT AFTER (10 mS)

Y

READ-AB_*(AB_MESSAGE)

MSG== AB_APP_START — N → EXIT(ERROR)

Y

WRITE-AB_*(AB_HW_CONFIG_MSG, MSG_SIZE)

WRITE-AB_*(AB_SW_CONFIG_MSG, MSG_SIZE)

WRITE-AB_*(KICKSTART, MSG_SIZE)

AB_APPLICATION_RUNNING

**Figure 52. Host Controlled Master Boot
(Downloading both a DSPAB Application Code and a DSPC Application Code)**

to DSPC. The supported messages are described in Table 16. This message tells DSPC where to get the DSPAB image from. Currently the C_MASTER_BOOT_FLASH message is supported and allows booting from external byte-wide flash or eprom.

3) If the initialization was successful DSPC sends out the C_BOOT_START message and the host must proceed to step 4. If initialization fails the host must re-try steps 1 through 3 and if failure is met again, the communication timing and protocol should be inspected.

4) After receiving the C_BOOT_START message, the host must release control of the communication interface and wait for $\overline{\text{INTREQ}}$ to go low.

> **Note:** DSPC will autoboot DSPAB. After this DSPC will release control of the communication interface, but only when in SPI Master Boot Mode.

5) The end of the .ULD file contains a three byte checksum. If the checksum is good after the download, DSPC will send a C_BOOT_SUCCESS message to the host. If the checksum was bad, DSPC responds with the C_BOOT_ERROR_CHECKSUM message message and waits for a hard reset.

6) After reading out the C_BOOT_SUCCESS message, the host must send a second C_MASTER_SOURCE_MODE boot message to DSPC. This messages tells DSPC where the to get image for DSPC.

7) If the initialization was successful DSPC sends out the C_BOOT_START message and the host must proceed to step 8. If initialization fails the host must re-try step 6. If failure is met again, the communication timing and protocol should be inspected.

8) After receiving the C_BOOT_START message, the host must release control of the communication interface and wait for $\overline{\text{INTREQ}}$

to go low.

> **Note:** DSPC will autoboot DSPC. After this DSPC will release control of the communication interface, but only when in SPI Master Boot Mode.

9) The end of the .ULD file contains a four byte checksum. If the checksum is good after the download, DSPC will send a C_BOOT_SUCCESS message to the host. If the checksum was bad, DSPC responds with the C_BOOT_ERROR_CHECKSUM message and waits for a hard reset.

10) After reading out the C_BOOT_SUCCESS message, the host must send the C_SOFT_RESET message which will cause the application code to reset and allow the downloaded application to run.

11) If the soft reset was successful, DSPC sends out a C_APP_START message the host can proceed to step 12. If DSPC does not send an the application start message, the host must re-try the whole procedure again.

12) Next the host can send hardware and software configuration messages for DSPC.

13) At this point the application code on DSPC is running and the host needs to configure DSPAB.

14) The host must read the AB_APP_START message from DSPAB. If DSPAB does not send an the application start message the host must re-try the whole procedure again, starting with step 1.

15) The host must send hardware and software configuration messages for DSPAB, ending with the kickstart message.

16) At this point the application code on DSPAB is running.

> **Note:** Hardware configuration messages are used to define the behavior of the DSP's audio ports. A more detailed description of the different hardware configurations can be found

in the Section 10 "Hardware Configuration" on page 78.

Note: 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. Writing the KICKSTART message to DSPAB begins the audio decode process.

8.2 Host Boot Via DSPC

A host controlled boot via DSPC is a sequence where the host boots DSPAB and DSPC through DSPC with two separate images (.ULD files).

Figure 53, "Host Boot Via DSPC" on page 75 demonstrates the interaction required by the microcontroller.

1) A download sequence is started when the host holds the mode pins appropriately (UHS[2:0] and FHS[1:0]) and toggles $\overline{\text{RESET}}$.

2) The host must then send the C_SLAVE_MODE boot message to DSPC. This causes DSPC to initialize itself.

3) If the initialization was successful DSPC sends out a C_BOOT_START message. The host should proceed to step 4. If initialization fails, the host should re-try steps 1 through 3 and if failure is met again, the communication timing and protocol should be inspected.

4) After receiving the C_BOOT_START message, the host should write the downloadable image for DSPAB(.ULD file) to DSPC.

5) The host must wait for $\overline{\text{INTREQ}}$ to go low and read the message from DSPC.

6) After reading out the C_BOOT_SUCCESS message, the host must then send the C_SLAVE_MODE message to DSPC once more. This causes DSPC to initialize itself and get ready to accept a stream.

7) If the initialization was successful DSPC sends out a C_BOOT_START message. The host should proceed to step 8. If initialization fails,

the host should re-try step 6. If failure is met again, the communication timing and protocol should be inspected.

8) After receiving the C_BOOT_START message, the host should write the downloadable image for DSPC.

9) The host must wait for $\overline{\text{INTREQ}}$ to go low and read the message from DSPC.

10) After reading out the C_BOOT_SUCCESS message. The host must send the C_SOFT_RESET message which will cause the application code to reset and allow the downloaded application to run.

11) If the soft reset was successful, DSPC sends out a C_APP_START message the host can proceed to step 8. If DSPC does not send an the application start message, the host must re-try the whole procedure again.

12) Next the host can send hardware and software configuration messages for DSPC.

13) At this point the application code on DSPC is running and the host needs to configure DSPAB.

14) The host must read the AB_APP_START message from DSPAB. If DSPAB does not send an the application start message the host must re-try the whole procedure again, beginning at setp 1.

15) The host must send hardware and software configuration messages for DSPAB.

16) At this point the application code on DSPAB is running.

> **Note:** Hardware configuration messages are used to define the behavior of the DSP's audio ports. A more detailed description of the different hardware configurations can be found in the Section 10 "Hardware Configuration" on page 78.
> **Note:** The software configuration messages are specific to each application. The application code user's guide for each application provides

START

RESET(LOW)

WAIT 10 uS

RESET(HIGH)

WRITE-C_*
(C_SLAVE_MODE)

INTREQ LOW? →N→ TIMEOUT AFTER (5 mS)
↓Y

READ-C_*(C_MESSAGE)

MSG ==C_BOOT_START →N→ EXIT(ERROR)
↓Y

WRITE-C_*(.ULD FILE,FILE SIZE) for DSP AB

INTREQ LOW? →N→ TIMEOUT AFTER (5 mS)
↓Y

READ-C_*(C_MESSAGE)

MSG== C_BOOT_SUCCESS →N→ EXIT(ERROR)
↓Y

WRITE-C_*
(C_SLAVE_MODE)

INTREQ LOW? →N→ TIMEOUT AFTER (5 mS)
↓Y

READ-C_*(C_MESSAGE)

MSG ==C_BOOT_START →N→ EXIT(ERROR)
↓Y

WRITE-C_*(.ULD FILE,FILE SIZE) for DSP C

INTREQ LOW? →N→ TIMEOUT AFTER (5 mS)
↓Y

READ-C_*(C_MESSAGE)

MSG== C_BOOT_SUCCESS →N→ EXIT(ERROR)
↓

WRITE-C_*(C_SOFT_RESET)

INTREQ LOW? →N→ TIMEOUT AFTER (5 mS)
↓Y

READ-C_*(C_MESSAGE)

MSG== C_APP_START →N→ EXIT(ERROR)
↓Y

WRITE-C_*(C_HW_CONFIG_MSG, MSG_SIZE)

WRITE-C_*(C_SW_CONFIG_MSG, MSG_SIZE)

WRITE-C_*(KICKSTART, MSG_SIZE)

C_APPLICATION_RUNNING

FINTREQ LOW? HOUTRDY HI? →N→ TIMEOUT AFTER (5 mS)
↓Y

READ-AB_*(AB_MESSAGE)

MSG== AB_APP_START →N→ EXIT(ERROR)
↓Y

WRITE-AB_*(AB_HW_CONFIG_MSG, MSG_SIZE)

WRITE-AB_*(AB_SW_CONFIG_MSG, MSG_SIZE)

WRITE-AB_*(KICKSTART, MSG_SIZE)

AB_APPLICATION_RUNNING

Figure 53. Host Boot Via DSPC

a list of all pertinent configuration messages. Writing the KICKSTART message to DSPAB begins the audio decode process.

9. SOFT RESETTING THE CS49400
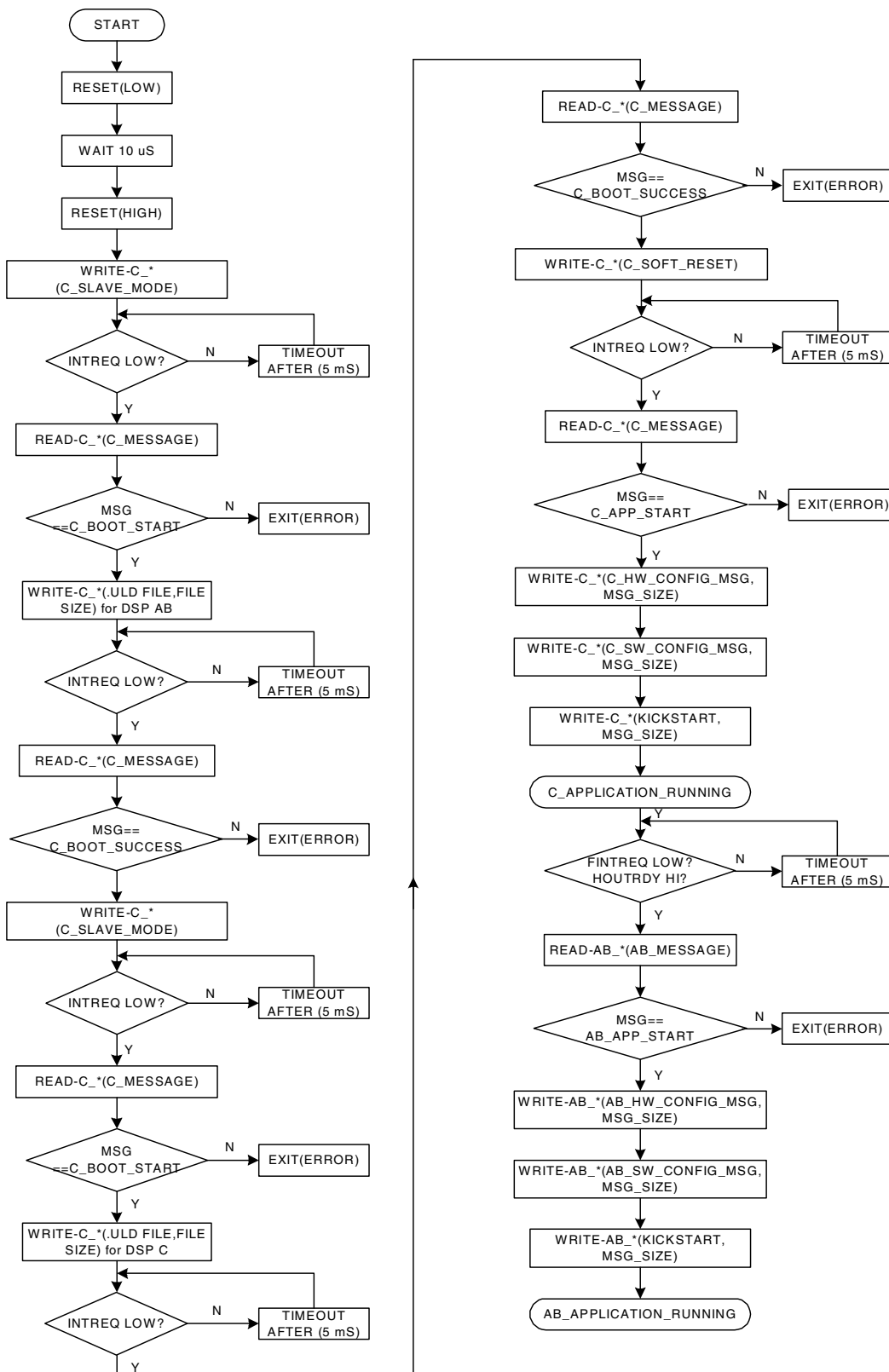
Soft resetting the CS49400 uses a combination of software and hardware. This method of resetting the DSP is usually referred to as a "soft reset" even though it involves toggling the reset pin due to the fact that a soft reset message is sent to the DSP. To soft reset the device, a previous application code must have been downloaded without power cycling the DSP. Figure 54, "Host Controlled Master Softreset" on page 77 describes the soft reset procedure. The main purpose behind a soft reset is to take advantage of the fact that all AC-3 based codes can accept both AC-3 compressed data as well as PCM data. This allows for a the host to reconfigure the AC-3 application code for PCM or AC-3 without having to completely redownload the same application code.

9.1 Host Controlled Master Soft Reset

This reset procedure is used to restart the application code that has already been loaded on the DSP. All writes and reads with the CS49400 should follow the protocol given in Section 8 "Boot Procedure" on page 71.

1) A Soft Reset sequence is started when the host holds the mode pins appropriately (UHS[2:0] and FHS[1:0]) and toggles $\overline{\text{RESET}}$.

2) The host must send the C_SOFT_RESET message to DSPC. This causes the application code on DSPAB and DSPC to reset and allow the downloaded application to run.

3) If the soft reset was successful, DSPC sends out a C_APP_START message the host can proceed to step 4. If DSPC does not send an the application start message, the host must re-try the whole procedure again.

4) The host can send hardware and software configuration messages for DSPC.

5) At this point the application code on DSPC is running and the host needs to configure DSPAB.

6) Next the host must wait for FINTREQ to go low and read the AB_APP_START message from DSPAB. If DSPAB does not send an the application start message the host must re-try the whole procedure again, beginning from step 1.

7) The host must send hardware and software configuration messages for DSPAB.

8) At this point the application code on DSPAB is running.
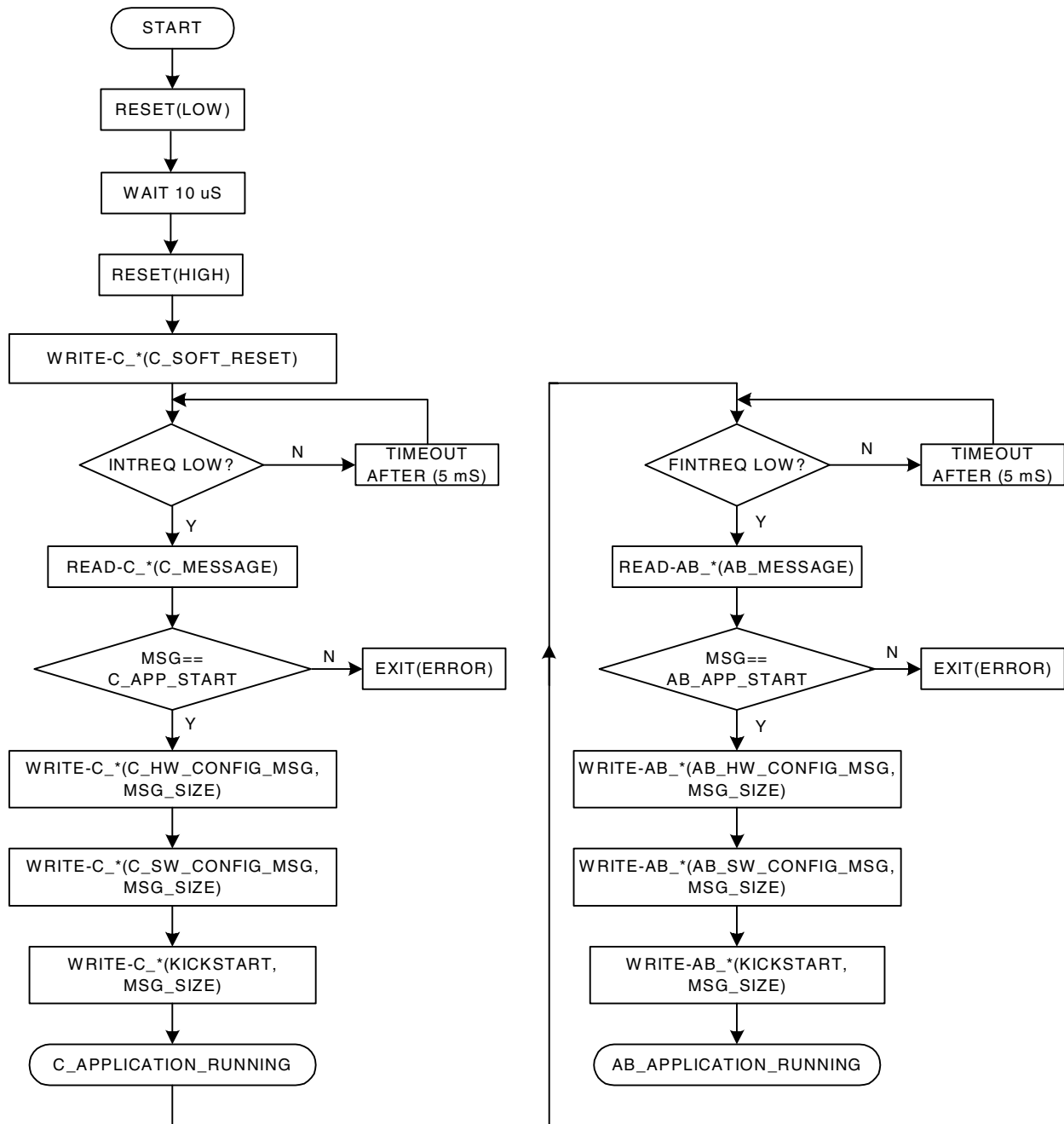
Figure 54. Host Controlled Master Softreset

10. HARDWARE CONFIGURATION

After download or soft reset, and before kickstart, the host has the option of changing the default hardware configuration. (Please see the Audio Manager in the Application Messaging Section of any Application Code User's Guide for more information on kickstarting.)

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

In general, the hardware configuration can only be changed immediately after download or after soft reset and before kickstart. However, some applications provide the capability to change the input ports without affecting other hardware configurations, after sending a special Application Restart message. (Please see the Audio Manager in any Application Code User's Guide to determine whether the Application Restart message is supported.)

11. DIGITAL INPUT AND OUTPUT DATA FORMATS

The CS49400 supports a wide variety of data input and output 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 CS49400 hardware. This does not mean that 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 local FAE as to its availability.

11.1 Digital Audio Formats

This subsection will describe some common audio formats that the CS49400 supports. It should be noted that the input ports use up to 24-bit PCM resolution and 16-bit compressed data word lengths. The output port of the CS49400 provides up to 24-bit PCM resolution.

11.1.1 I^2S

Figure 55, "I^2S Format" on page 79 shows the I^2S format. For I^2S, data is presented most significant bit first, one FSCLKN1 delay after the transition of FLRCLKN1, and is valid on the rising edge of FSCLKN1. For the I^2S format, the left subframe is presented when FLRCLKN1 is low; the right subframe is presented when FLRCLKN1 is high. FSCLKN1 is required to run at a frequency of 48Fs or greater on the input ports.

11.1.2 Left Justified

Figure 56, "Left Justified Format (Rising Edge Valid SCLK)" on page 79 shows the left justified format with a rising edge FSCCLK. Data is presented most significant bit first on the first FSCLKN1 after an FLRCLKN1 transition and is valid on the rising edge of FSCLKN1. For the left justified format, the left subframe is presented when FLRCLKN1 is high and the right subframe is presented when FLRCLKN1 is low. The left justified format can also be programmed for data to be valid on the falling edge of FSCLKN1. FSCLKN1 is required to run at a frequency of 48Fs or greater on the input ports.

11.2 Digital Audio Input Port

The digital audio input port (DAI) on DSPAB, is used for both compressed and PCM digital audio data input. In addition this port supports a special clocking mode in which a clock can be input to directly drive the internal 33 bit counter. Table 18, "Digital Audio Input Port," on page 79 shows the

pin names, mnemonics and pin numbers associated with the DAI.

Pin Name	Pin Description	144-Pin Package, Pin Number	100-Pin Package, Pin Number
FSDATAN1	Serial Data In	131	84
FSCLKN1 FSTCCLK2	Serial Bit Clock Secondary STC Clock	134	81
FLRCLKN1	Frame Clock	119	85

Table 18. Digital Audio Input Port

The DAI is fully configurable including support for I²S and left-justified formats. DAI is programmed for slave clocks, where FLRCLKN1 and FSCLKN1 are inputs. **All DAI configuration messages must be sent to DSPAB.**

11.3 Compressed Data Input Port

The compressed data input port (CDI) on DSPAB can be used for both compressed and PCM data input. Table 19 shows the mnemonic, pin name, and pin number of the pins associated with the CDI port on the CS49400.

The CDI port is fully configurable for all data formats including: I²S, left-justified and multichannel formats. FLRCLKN2 and FSCLKN2 on the CDI port are programmed to be inputs. **All CDI configuration messages must be sent to DSPAB**.

Pin Name	Pin Description	144-Pin Package, Pin Number	100-Pin package, Pin Number
FSDATAN2 CMPDAT	Serial Data In Compressed Data In	118	79
FSCLKN2 CMPCLK	Serial Bit Clock	111	78
FLRCLKN2 CMPREQ	Frame Clock Data Request Out	117	80

Table 19. Compressed Data Input Port

11.4 Input Data Hardware Configuration for CDI and DAI on DSPAB

Both data format (I²S, Left Justified) and data type (compressed or PCM) are required to fully define the input port's hardware configuration. The DAI and the CDI are configured by the same group of messages since their configurations are interrelated. The naming convention of the input hardware configuration is as follows:

INPUT *A B C*

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

A - Data Type

B - Data Format

C - SCLK Polarity
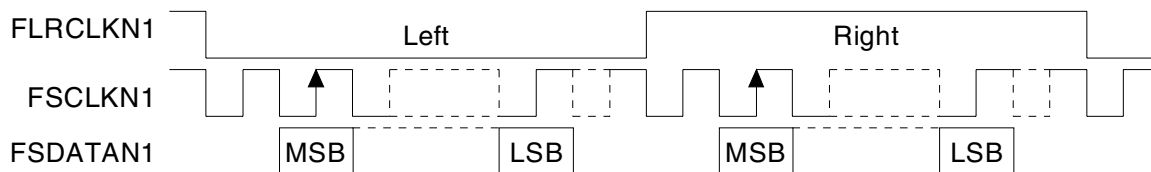
Figure 55. I²S Format
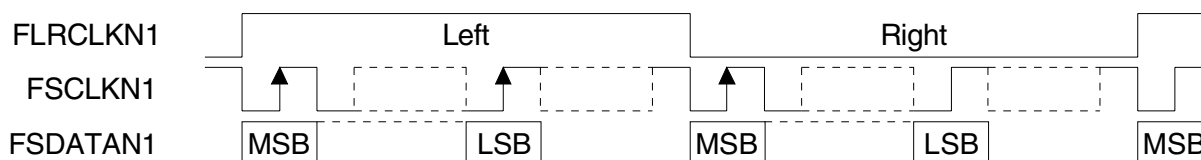
Figure 56. Left Justified Format (Rising Edge Valid SCLK)

The following tables show the different values for each parameter as well as the hex message that needs to be sent. When creating the hardware configuration message, only one hex message should be sent per parameter. It should be noted that the entire B parameter hex message must be sent, even if one of the input ports has been defined as unused by the A parameter.

A Value	Data Type	Hex Message
0 (default)	DAI - PCM CDI - Compressed	0x800210 0x3FBFC0 0x800110 0x80002C
1	DAI - PCM and Compressed CDI - Unused	0x800210 0x3FBFC0 0x800110 0xC0002C
2	DAI - Unused CDI - PCM	0x800210 0x3FBFC0 0x800110 0x800020

Table 20. Input Data Type Configuration (Input Parameter A)

B Value	Data Format	Hex Message
0 (default)	PCM - I^2S 24-bit Compressed - I^2S 16-bit *Compressed means any type of compressed data such as IEC61937-packed AC-3, DTS, MPEG Multichannel, AAC or MP3 elementary stream data from a DVD or IEC60958-packed elementary stream DTS data from a DTS-CD)*	0x800217 0x8080FF 0x80021A 0x8080FF 0x800117 0x011100 0x80011A 0x011900

Table 21. Input Data Format Configuration (Input Parameter B)

B Value	Data Format	Hex Message
1	PCM - Left Justified 24-bit Compressed - Left Justified 16-bit *(Compressed means any type of compressed data such as IEC61937-packed AC-3, DTS, MPEG Multichannel, AAC or MP3 elementary stream data from a DVD or IEC60958-packed elementary stream DTS data from a DTS-CD)*	0x800217 0x8080FF 0x80021A 0x8080FF 0x800117 0x001000 0x80011A 0x001800

Table 21. Input Data Format Configuration (Input Parameter B) (Continued)

C Value	SCLK Polarity (Both CDI and DAI Port)	Hex Message
0 (default)	Data Clocked in on Rising Edge	0x800217 0xFFFFDF 0x80021A 0xFFFFDF
1	Data Clocked in on Falling Edge	0x800117 0x000020 0x80011A 0x000020

Table 22. Input SCLK Polarity Configuration (Input Parameter C)

11.4.1 Input Configuration Considerations

1) 24-bit PCM input requires at least 24 SCLKs per sub-frame. The DSP always uses 24-bit resolution for PCM input. Systems having less than 24-bit resolution will not have a problem as the extra bits taken by the DSP will be under the noise floor of the input signal for left justified and I^2S formats. For compressed input, data is always taken in 16 bit word lengths.

2) If the clocks to the audio ports are known to be corrupted, such as when a S/PDIF receiver goes out of lock, the DSP should undergo an application restart (if applicable), soft reset, or hard reset. All three actions will result in the input FIFO being reset. Failure to do so *may* result in corrupted data being latched into the

input FIFO and may result in corrupted data being heard on the outputs.

Corruption is only an issue when PCM data is being delivered. When compressed data is being delivered, there are sync words embedded in the data stream to which the DSP can lock. Certain application codes that are capable of processing PCM may now have a special feature called "PCM Robustness" which prevents the corruption described above, but you should still use a FIFO reset to ensure good data.

11.5 Serial Audio Input

The Serial Audio Input (SAI) provides four stereo inputs to DSPC. The SAI can be used to post-process PCM data from a multichannel Super Audio CD input or DVD Audio/Video input via high-performance A/Ds. Table 19 shows the

Pin Name	Pin Description	144-Pin Package, Pin Number	100-Pin Package, Pin Number
SCLKN	Serial Bit Clock	86	60
LRCLKN	Frame Clock	85	59
SDATAN3	Serial Data In 3	79	55
SDATAN2	Serial Data In 2	80	56
SDATAN1	Serial Data In 1	81	57
SDATAN0	Serial Data In 0	82	58

Table 23. Serial Audio Input Port

mnemonic, pin name and pin number of the pins associated with the SAI port on the CS49400.

The SAI has 4 stereo data inputs that are fully configurable including support for I²S, left-justified and multichannel formats. The SAI port operates in slave mode only with LRCLKN and SCLKN as inputs. Processing on the CDI and DAI ports must be disabled before the SAI port is enabled. Either the input D0 or D1 message must

be sent to DSPC to configure and enable the SAI port.

D Value	Data Type	Hex Message
0	PCM - I²S 24-bit	0x81000010 0x00000001 0x81000011 0x00011701 0x81000012 0x00004E4F
1	PCM - Left Justified 24-bit	0x81000010 0x00000001 0x81000011 0x00001600 0x81000012 0x00005E4F

Table 24. SAI Data Type Configuration (Input Parameter D)

11.6 Digital Audio Output Port

The Digital Audio Output port (DAO) can transmit up to 16 channels of PCM data that are fully configurable into standard audio format. It also has two IEC60958 pins that provide CMOS level bi phase encoded outputs. Table 25 shows the signals associated with the DAO. As with the input ports the clocks and data are fully configurable via hardware configuration. **All DAO configuration messages must be sent to DSPC.**

Pin Name	Pin Description	144-Pin Number	100-Pin Number
MCLK	Master Clock	99	68
SCLK1	Serial Bit Clock for AUDATA 4-7	98	67
LRCLK1	Frame Clock for AUDATA 4-7	87	61
AUDATA7,X MT958B	Serial Data Out 7, IEC60958 Transmitter	92	64
AUDATA6	Serial Data Out 6	93	65
AUDATA5	Serial Data Out 5	94	66
AUDATA4	Serial Data Out 4	102	71

Table 25. Digital Audio Output Port

Pin Name	Pin Description	144-Pin Number	100-Pin Number
SCLK0	Serial Bit Clock for AUDATA 0-3	104	72
LRCLK0	Frame Clock for AUDATA 0-3	108	75
AUDATA3,X MT958A	Serial Data Out 3, IEC60958 Transmitter	106	73
AUDATA2	Serial Data Out 2	107	74
AUDATA1	Serial Data Out 1	109	76
AUDATA0	Serial Data Out 0	110	77

Table 25. Digital Audio Output Port (Continued)

MCLK is the master clock and is firmware configurable to be either an input or an output. 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.

SCLK0 is the bit clock used to clock data out on AUDATA0, AUDATA1, AUDATA2 and AUDATA3. LRCLK0 is the data framing clock whose frequency is typically equal to the sampling frequency for AUDATA0, AUDATA1, AUDATA2 and AUDATA3.

SCLK1 is the bit clock used to clock data out on AUDATA4, AUDATA5, AUDATA6 and AUDATA7. LRCLK1 is the data framing clock whose frequency is typically equal to the sampling frequency for AUDATA4, AUDATA5, AUDATA6 and AUDATA7.

LRCLK0, LRCLK1, SCLK0 and SCLK1 can be configured as either inputs (Slave) or outputs (Master). A valid MCLK is required for all output modes. When LRCLK0, LRCLK1, SCLK0 and SCLK1 are configured as outputs, they are derived from MCLK. Whether MCLK is configured as an input or an output, an internal divider from the MCLK signal is used to produce LRCLK0, LRCLK1, SCLK0 and SCLK1. The ratios shown in Table 26 give the possible SCLK values for different MCLK frequencies. (All values are expressed in terms of the sampling frequency, Fs.)

MCLK (Fs)	SCLK (Fs)					
	32	48	64	128	256	512
128	X		X			
384	X	X	X			
256	X		X	X	X	
512	X		X	X	X	X

Table 26. MCLK/SCLK Master Mode Ratios

Both the AUDAT0 and AUDAT4 Digital Audio Output porst are configurable to provide output for two, four, or six channels of PCM data. AUDATA1, AUDATA2, AUDATA3, AUDATA5, AUDATA6 and AUDATA7 are only capable of outputting two channels of PCM data. Typically AUDATA[0:7] are configured for outputting either left justified or I^2S formatted data. In a standard 5.1 channel AVR, AUDATA0, AUDATA1 and AUDATA2 are used to output the six discrete channels (Left, Center, Right, Left Surround, Right Surround, and Subwoofer).

AUDATA3 can be used with AUDATA0, AUDATA1 and AUDATA2 to support 7.1 output. Alternatively AUDATA3 and AUDATA7 can be used for dual zone support. AUDATA3 and AUDATA7 are multiplexed with the XMT958 output so only one can be used at any one time.

Please refer to AN208, AN209 and their corresponding appendices for information about which output modes are supported, as this is specific to each application code.

11.6.1 S/PDIF Outputs

Both AUDATA3 and AUDATA7 digital audio output ports are unique, in that they can serve either an additional output for I2S or Left Justified PCM data OR as IEC60959 bi-phase mark encoded data S/PDIF transmitters. When either of these ports are configured as a S/PDIF transmitter, the MCLK required for such functionality can be provided from either the internally locked PLL or from an

MCLK input. All consumer channel status information can be included in the S/PDIF stream, provided that the particular application code supports this functionality.

When configured as a S/PDIF transmitter, the designer should understand that in order for these ports to be fully IEC60958 compliant, the outputs would need to be buffered through an RS422 device or an optocoupler as its outputs are only CMOS driven.

11.7 Output Data Hardware Configuration

The DAO naming convention is as follows:

OUTPUT *A B C D,*

where the parameters are defined as:

A - DAO Mode (Master/Slave for LRCLK0, LRCLK1, SCLK0 and SCLK1)

B - Data Format

C - MCLK, SCLK, LRCLK Frequency

The following tables show the different values for each parameter as well as the hex message that needs to be sent. When creating the hardware configuration message, only one hex message should be sent per parameter.

A Value	DAO Modes (LRCLK and SCLK)	Hex Message
0 (default)	MCLK - Slave SCLK0 - Slave LRCLK0 - Slave SCLK1 - Slave LRCLK1 - Slave	0x81800003 0x00101000
1	MCLK - Slave SCLK0 - Master LRCLK0 - Master SCLK1 - Master LRCLK1 - Master	0x81800003 0xFFEFFFFF

Table 27. Output Clock Configuration (Parameter A)

B Value	DAO Data Format Of AUDATA0, 1, 2 (or AUDATA0 for Multichannel Modes)	Hex Message
0 (default)	I^{2}S 24-bit *(Configuration of AUDATA3 as S/PDIF (IEC60958) or Digital Audio in the format of I^{2}S or Left Justified is covered in AN209)*	0x81800003 0xFFFE3FFF 0x81400003 0x0001C000 0x81000005 0x00101701 0x81000006 0x00100001 0x81000007 0x00100001 0x81000008 0x00100001
1	Left Justified 24-bit *(Configuration of AUDATA3 as S/PDIF (IEC60958) or Digital Audio in the format of I^{2}S or Left Justified is covered in AN209)*	0x81800003 0xFFFE3FFF 0x81400003 0x0000C000 0x81000005 0x00101701 0x81000006 0x00100000 0x81000007 0x00100000 0x81000008 0x00100000

Table 28. Output Data Configuration Parameter B)

C Value	SCLK/LRCLK Frequency	Hex Message
0 (default)	MCLK = 256 FS SCLK = MCLK / 4 = 64 FS LRCLK = SCLK / 64 = FS	0x81800003 0xFFFFFC7F 0x81400003 0x00000100 0x81000009 0x00077030
1	MCLK = 256 FS SCLK = MCLK / 2 = 128 FS LRCLK = SCLK / 128 = FS	0x81800003 0xFFFFFC7F 0x81400003 0x00000200 0x81000009 0x00177010

Table 29. Output SCLK/LRCLK Configuration (Parameter C)

C Value	SCLK/LRCLK Frequency	Hex Message
2	MCLK = 256 FS	0x81800003
		0xFFFFFC7F
		0x81400003
		0x00000300
	SCLK = MCLK / 1 = 256 FS	0x81000009
	LRCLK = SCLK / 256 = FS	0x00377000
3	MCLK = 512 FS	0x81800003
		0xFFFFFC7F
		0x81400003
		0x00000100
	SCLK = MCLK / 8 = 64 FS	0x81000009
	LRCLK = SCLK / 64 = FS	0x00077070
4	MCLK = 128FS	0x81800003
		0xFFFFFC7F
		0x81400003
		0x00000100
	SCLK = MCLK / 2 = 64FS	0x81000009
	LRCLK = SCLK / 64 = FS	0x00077010

Table 29. Output SCLK/LRCLK Configuration (Parameter C)

D Value	SCLK Polarity	Hex Message
0 (default)	Data Valid on Rising Edge (clocked out on falling)	0x81800003 0xFFFBFFFF
1	Data Valid on Falling Edge (clocked out on rising)	0x81400003 0x00040000

Table 30. Output SCLK Polarity Configuration (Parameter D)

11.8 Creating Hardware Configuration Messages

The single hardware configuration message that must be sent to the CS49400 after download or soft reset should be a concatenation of the messages in the previous sections. The complete hardware configuration message should be created by taking a message for each parameter (where the default is not acceptable) and concatenating the messages together. No messages need to be sent if the default configuration for a particular parameter is acceptable. This example can be easily expanded to fit other system requirements.

E.g. if the host system has this configuration:

Address Checking: Disabled

This is the default configuration so no configuration message is required.

DAI:Left Justified
PCM and Compressed data

CDI:Not used

The above configuration corresponds to

INPUT A1 B1

which corresponds to a configuration message of:

0x800210
0x3FBFC0
0x800110
0xC0002C

0x800217
0x8080FF
0x80021A
0x8080FF
0x800117
0x001000
0x80011A
0x001800

DAO:Left Justified slave mode (LRCLK, SCLK inputs)
MCLK @ 256Fs
SCLK @ 64Fs

The above configuration corresponds to

OUTPUT A0 B1 C0 D0

which has a configuration message of:

0x81800003
0xFFFE3FFF
0x81400003
0x0000C000
0x81000005
0x00101701
0x81000006
0x00100000
0x81000007

0x00100000

0x81000008

0x00100000

Concatenating the messages together gives the hardware configuration message shown in Table 31, "Example Values to be Sent to DSPAB After Download or Soft Reset," on page 85, which should be sent to DSPAB after download or soft reset. Table 32, "Example Values to be Sent to DSPC After Download or Soft Reset," on page 85, which should be sent to DSPC after download or soft reset.

WORD#	VALUE	WORD#	VALUE
1	0x800210	7	0x80021A
2	0x3FBFC0	8	0x8080FF
3	0x800110	9	0x800117
4	0xC0002C	10	0x001000
5	0x800217	11	0x80011A
6	0x8080FF	12	0x001800

Table 31. Example Values to be Sent to DSPAB After Download or Soft Reset

WORD#	VALUE	WORD#	VALUE
1	0x81000006	12	0x81000007
2	0x00101700	13	0x00100000
3	0x81000006	14	0x81000008
4	0x00100000	15	0x00100000

Table 32. Example Values to be Sent to DSPC After Download or Soft Reset

12.0 PIN DESCRIPTION

12.1 144-Pin LQFP Package Pin Layout
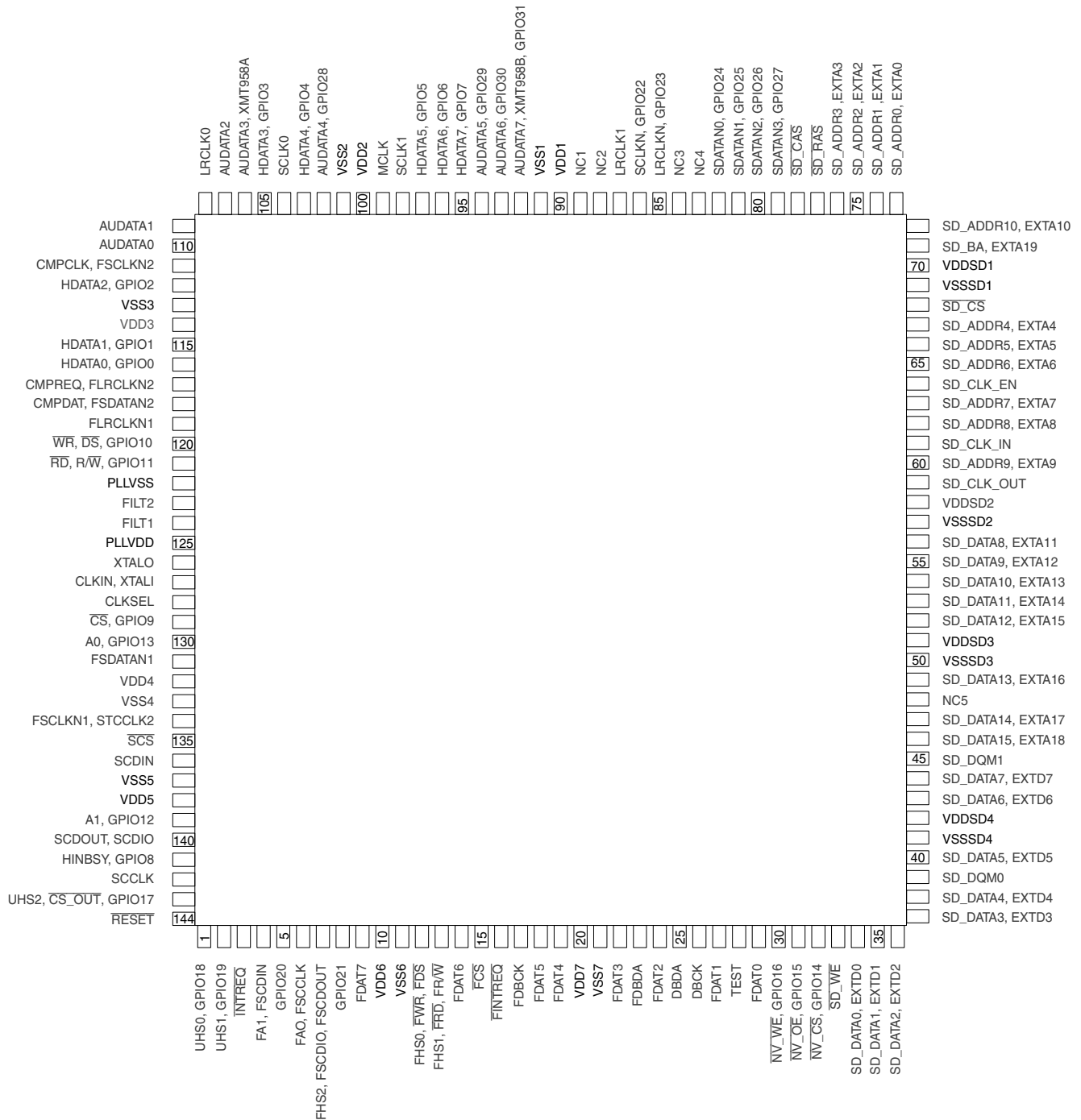
Figure 57. Pin Layout (144-Pin LQFP Package)

12.2 100-Pin LQFP Package Pin Layout

Top pins (left to right, 75 to 55):
LRCLK0 (75), AUDATA2, AUDATA3, XMT958A, SCLK0, AUDATA4, GPIO28, VSS2 (70), VDD2, MCLK, SCLK1, AUDATA5, GPIO29, AUDATA6, GPIO30 (65), AUDATA7, XMT958B, GPIO31, VSS1, VDD1, LRCLK1, SCLKN, GPIO22 (60), LRCLKN, GPIO23, SDATAN0, GPIO24, SDATAN1, GPIO25, SDATAN2, GPIO26, SDATAN3, GPIO27 (55), EXTA3, EXTA2, EXTA1, EXTA0

Left pins (top to bottom):
AUDATA1
AUDATA0
CMPCLK, FSCLKN2
VSS3
VDD3 (80)
CMPREQ, FLRCLKN2
CMPDAT, FSDATAN2
FLRCLKN1
PLLVSS
FILT2 (85)
FILT1
PLLVDD
XTALO
CLKIN, XTALI
CLKSEL (90)
FSDATAN1
FSCLKN1, STCCLK2
SCS
SCDIN
VSS5 (95)
VDD5
SCDOUT, SCDIO
SCCLK
UHS2, CS_OUT, GPIO17
RESET (100)

Right pins (top to bottom):
EXTA10 (50)
EXTA19
VDDSD1
VSSSD1
EXTA4
EXTA5 (45)
EXTA6
EXTA7
EXTA8
EXTA9
EXTA11 (40)
EXTA12
EXTA13
EXTA14
EXTA15
EXTA16 (35)
EXTA17
EXTA18
EXTD7
EXTD6
VDDSD4 (30)
VSSSD4
EXTD5
EXTD4
EXTD3

Bottom pins (left to right, 1 to 25):
UHS0, GPIO18 (1), UHS1, GPIO19, INTREQ, FA1, FSCDIN, FA0, FSCCLK (5), FHS2, FSCDIO, FSCDOUT, VDD6, VSS6, FHS0, FWR, FDS, FHS1, FRD, FR/W (10), FCS, FINTREQ, FDBCK, VDD7, VSS7 (15), FDBDA, DBDA, DBCK, TEST, NV_WE, GPIO16 (20), NV_OE, GPIO15, NV_CS, GPIO14, EXTD0, EXTD1, EXTD2 (25)

Figure 58. Pin Layout (100-Pin LQFP Package)

12.3 Pin Definitions

FILT1 — Phase-Locked Loop Filter

Connects to an external filter for the on-chip phase-locked loop.

FILT2 — Phase Locked Loop Filter

Connects to an external filter for the on-chip phase-locked loop.

CLKIN, XTALI — External Clock Input/Crystal Oscillator Input

CS49400 clock input. This pin accepts an external clock input signal that is used to drive the internal core logic. When in internal clock mode (CLKSEL == VSS), this input is connected to the internal PLL from which all internal clocks are derived. When in external clock mode (CLKSEL == VDD), this input is connected to the DSP clock. Alternatively, a 12.288 mhZ crystal oscillator can be connected between XTALI and XTALO. *INPUT*
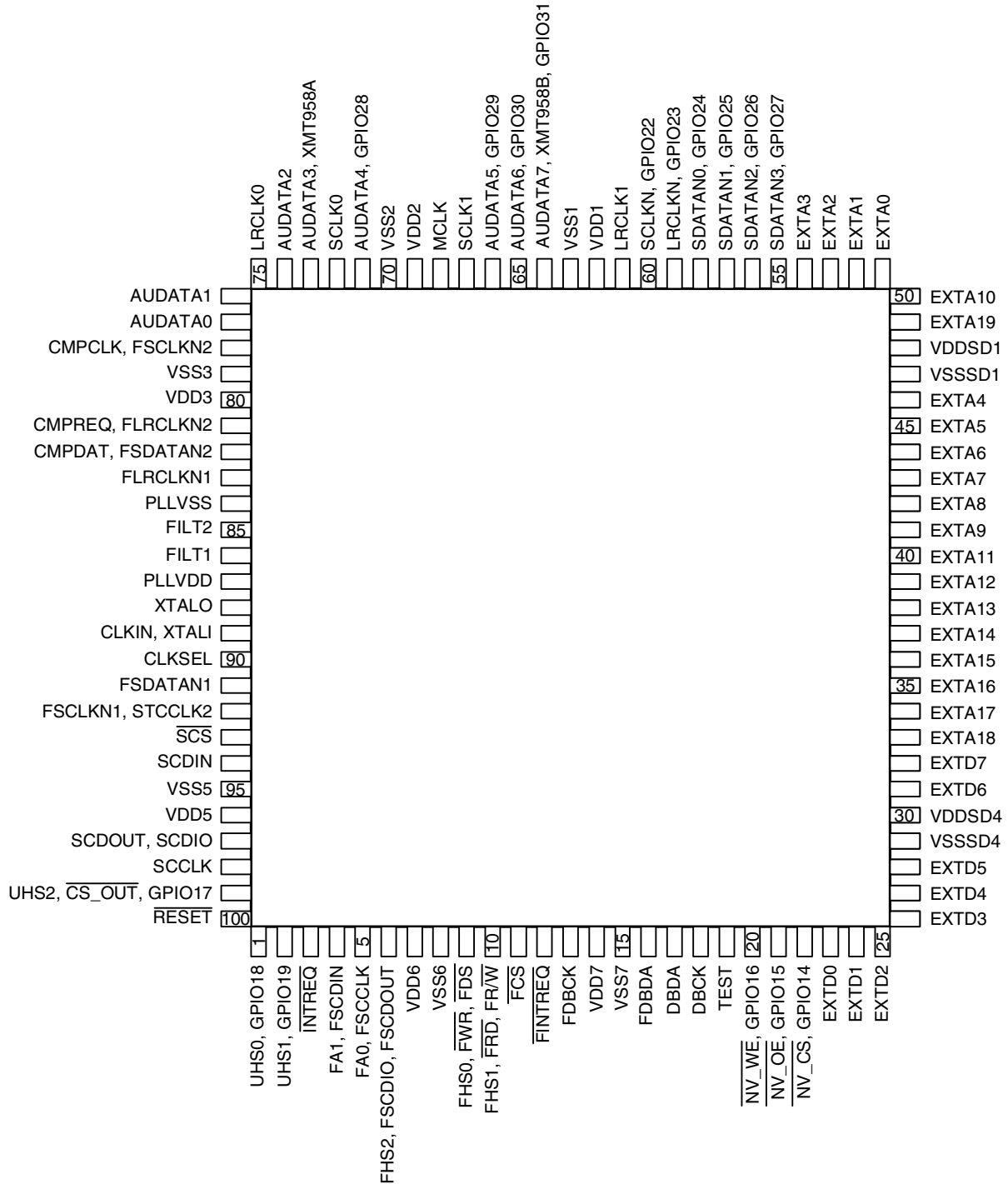
XTALO — Crystal Oscillator Output

Crystal oscillator output. *OUTPUT*

CLKSEL — DSP Clock Select

This pin selects the internal source clock. When CLKSEL is low, CLKIN is connected to the internal PLL from which all internal clocks are derived. When CLKSEL is high, the PLL is bypassed and the external clock directly drives all input logic. *INPUT*

FDAT7 — DSPAB Bidirectional Data Bus

FDAT6

FDAT5

FDAT4

FDAT3

FDAT2

FDAT1

FDAT0

In parallel host mode, these pins provide a bidirectional data bus to DSPAB. These pins have an internal pull-up.
BIDIRECTIONAL - Default: INPUT

FA0, FSCCLK — Host Parallel Address Bit Zero or Serial Control Port Clock

In parallel host mode, this pin serves as one of two address input pins used to select one of four parallel registers. In serial host mode, this pin serves as the serial control clock signal, specifically as the SPI clock input. *INPUT*

FA1, FSCDIN — Host Address Bit One or SPI Serial Control Data Input

In parallel host mode, this pin serves as one of two address input pins used to select one of four parallel registers. In SPI serial host mode, this pin serves as the data input. *INPUT*

FHS1, $\overline{FRD}$, FR/$\overline{W}$ — Mode Select Bit 1 or Host Parallel Output Enable or Host Parallel R/$\overline{W}$

DSPAB control port mode select bit 1. This bit is one of 3 control port select bits that are sampled on the rising edge of $\overline{RESET}$ to determine the control port mode of DSPAB. In Intel parallel host mode, this pin serves as the active-low data bus enable input. In Motorola parallel host mode, this pin serves as the read-high/write-low control input signal. In serial host mode, this pin can serve as the external memory active-low data-enable output signal. *BIDIRECTIONAL - Default: INPUT*

FHS0, $\overline{FWR}$, $\overline{FDS}$ — Mode Select Bit 0 or Host Write Strobe or Host Data Strobe

DSPAB control port mode select bit 0. This bit is one of 3 control port select bits that are sampled on the rising edge of $\overline{RESET}$ to determine the control port mode of DSPAB. In Intel parallel host mode, this pin serves as the active-low data-write-input strobe. In Motorola parallel host mode, this pin serves as the active-low data-strobe-input signal. In serial host mode, this pin can serve as the external-memory active-low write-enable output signal. *BIDIRECTIONAL - Default: INPUT*

$\overline{FCS}$ — Host Parallel Chip Select, Host Serial SPI Chip Select

In parallel host mode, this pin serves as the active-low chip-select input signal. In serial host SPI mode, this pin is used as the active-low chip-select input signal. *INPUT*

FHS2, FSCDIO, FSCDOUT — Mode Select Bit 2 or Serial Control Port Data Input and Output, Parallel Port Type Select

DSPAB control port mode select bit 2. This bit is one of 3 control port select bits that are sampled on the rising edge of $\overline{RESET}$ to determine the control port mode of DSPAB. In SPI mode this pin serves as the data output pin. In parallel host mode, this pin is sampled at the rising edge of $\overline{RESET}$ to configure the parallel host mode as an Intel type bus or as a Motorola type bus. *BIDIRECTIONAL - Default: INPUT*

$\overline{FINTREQ}$ — Control Port Interrupt Request

Open-drain interrupt-request output. This pin is driven low to indicate that the DSP has outgoing control data that should be read by the host.
OPEN DRAIN I/O - Requires 3.3K Ohm Pull-Up

FSCLKN1, STCCLK2 — PCM Audio Input Bit Clock

Digital-audio bit clock input. FSCLKN1 operates asynchronously from all other DSPAB clocks. In master mode, FSCLKN1 is derived from DSPAB's internal clock generator. The active edge of FSCLKN1 can be programmed by the DSP.
BIDIRECTIONAL - Default: INPUT

FLRCLKN1 — PCM Audio Input Sample Rate Clock

Digital-audio frame clock input. FLRCLKN1 typically is run at the sampling frequency. FLRCLKN1 operates asynchronously from all other DSPAB clocks. The polarity of FLRCLKN1 for a particular subframe can be programmed by the DSP.

BIDIRECTIONAL - Default: INPUT

FSDATAN1 — PCM Audio Data Input One

Digital-audio data input that can accept from one compressed line or 2 channels of PCM data. FSDATAN1 can be sampled with either edge of FSCLKN1, depending on how FSCLKN1 has been configured. *INPUT*

CMPCLK, FSCLKN2 — PCM Audio Input Bit Clock

Digital-audio bit clock input. FSCLKN2 operates asynchronously from all other DSPAB clocks. The active edge of FSCLKN2 can be programmed by the DSP.

BIDIRECTIONAL - Default: INPUT

CMPDAT, FSDATAN2 — PCM Audio Data Input Number Two

Digital-audio data input that can accept either one compressed line or 2 channels of PCM data. FSDATAN2 can be sampled with either edge of FSCLKN2, depending on how FSCLKN2 has been configured.

BIDIRECTIONAL - Default: INPUT

FDBCK — Reserved

This pin is reserved and should be pulled up with an external 3.3k resistor. *INPUT*

FDBDA — Reserved

This pin is reserved and should be pulled up with an external 3.3k resistor.

BIDIRECTIONAL - Default: INPUT

PLLVDD — PLL Supply Voltage

2.5 V PLL supply.

PLLVSS — PLL Ground Voltage

PLL ground.

RESET — Master Reset Input

Asynchronous active-low master reset input. 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. At the rising edge of reset the host interface mode of DSPAB is selected contingent on the state of the FHS0, FHS1, and FHS2 pins. At the rising edge of reset the host interface mode of DSPC is selected contingent on the state of the UHS0, UHS1, and UHS2 pins. If reset is low all bidirectional pins are high-Z inputs. *INPUT*

TEST — Reserved

This should be tied low for normal operation. *INPUT*

MCLK — Audio Master Clock

Bidirectional master audio clock. As an output, MCLK provides a low jitter oversampling clock. MCLK supports all standard oversampling frequencies. BIDIRECTIONAL - Default: INPUT

SCLK0 — Audio Output Bit Clock

Bidirectional digital-audio output bit clock for AUDATA0, AUDATA1, AUDATA2, and AUDATA3. As an output, SCLK0 can provide 32 Fs, 64 Fs, 128 Fs, 256 Fs, or 512 Fs frequencies and is synchronous to MCLK. As an input, SCLK0 is independent of MCLK.

BIDIRECTIONAL - Default: INPUT

SCLK1 — Audio Output Bit Clock

Bidirectional digital-audio output bit clock for AUDATA4, AUDATA5, AUDATA6, and AUDATA7. As an output, SCLK1 can provide 32 Fs, 64 Fs, 128 Fs, 256 Fs, or 512 Fs frequencies and is synchronous to MCLK. As an input, SCLK1 is independent of MCLK.

BIDIRECTIONAL - Default: INPUT

LRCLK0 — Audio Output Sample Rate Clock

Bidirectional digital-audio output frame clock for AUDATA0, AUDATA1, AUDATA2, and AUDATA3. As an output, LRCLK0 can provide all standard output sample rates up to 192 kHz and is synchronous to MCLK. As an input, LRCLK0 is independent of MCLK.

BIDIRECTIONAL - Default: INPUT

LRCLK1 — Audio Output Sample Rate Clock

Bidirectional digital-audio output frame clock for AUDATA4, AUDATA5, AUDATA6, and AUDATA7. As an output, LRCLK1 can provide all standard output sample rates up to 192 kHz and is synchronous to MCLK. As an input, LRCLK1 is independent of MCLK.

BIDIRECTIONAL - Default: INPUT

AUDATA0 — Digital Audio Output 0

PCM digital-audio data output. *OUTPUT*

AUDATA1 — Digital Audio Output 1

PCM digital-audio data output. *OUTPUT*

AUDATA2 — Digital Audio Output 2

PCM digital-audio data output. *OUTPUT*

AUDATA3, XMT958A — Digital Audio Output 3, S/PDIF Transmitter

CMOS level output that outputs a biphase-mark encoded (S/PDIF) IEC60958 signal or digital audio data which is capable of carrying two channels of PCM digital audio. *OUTPUT*

AUDATA4, GPIO28 — Digital Audio Output 4, General Purpose I/O

PCM digital-audio data output. This pin can act as a general-purpose input or output that can be individually configured and controlled by DSPC. *BIDIRECTIONAL - Default: OUTPUT*

AUDATA5, GPIO29 — Digital Audio Output 5, General Purpose I/O

PCM digital-audio data output. This pin can act as a general-purpose input or output that can be individually configured and controlled by DSPC. *BIDIRECTIONAL - Default: OUTPUT*

AUDATA6, GPIO30 — Digital Audio Output 6, General Purpose I/O

PCM digital-audio data output. This pin can act as a general-purpose input or output that can be individually configured and controlled by DSPC. *BIDIRECTIONAL - Default: OUTPUT*

AUDATA7, XMT958B, GPIO31 — Digital Audio Output 7, S/PDIF Transmitter, General Purpose I/O

CMOS level output that contains a biphase-mark encoded (S/PDIF) IEC60958 signal or digital audio data which is capable of carrying two channels of PCM digital audio. This pin can also act as a general-purpose input or output that can be individually configured and controlled by DSPC. *BIDIRECTIONAL - Default: OUTPUT*

DBCK — Debug Clock

Must be tied high to 3.3k ohm resistor. *INPUT*

DBDA — Debug Data

Must be tied high to 3.3k ohm resistor. *BIDIRECTIONAL - Default: INPUT*

SLCKN, GPIO22 — PCM Audio Input Bit Clock, General Purpose I/O

Digital-audio bit clock that is an input. SCLKN operates asynchronously from all other DSPAB clocks. The active edge of SCLKN can be programmed by the DSP. This pin can act as a general-purpose input or output that can be individually configured and controlled by DSPC.
BIDIRECTIONAL - Default: INPUT

LRCLKN, GPIO23 — PCM Audio Input Sample Rate Clock, General Purpose I/O

Digital-audio frame clock input. LRCLKN operates asynchronously from all other DSPAB clocks. The polarity of LRCLKN for a particular subframe can be programmed by the DSP. This pin can act as a general-purpose input or output that can be individually configured and controlled by DSPC. *BIDIRECTIONAL - Default: INPUT*

SDATAN0, GPIO24 — PCM Audio Input Data, General Purpose I/O

Digital-audio PCM data input. This pin can act as a general-purpose input or output that can be individually configured and controlled by DSPC. *BIDIRECTIONAL - Default: INPUT*

SDATAN1, GPIO25 — PCM Audio Input Data, General Purpose I/O

Digital-audio PCM data input. This pin can act as a general-purpose input or output that can be individually configured and controlled by DSPC. *BIDIRECTIONAL - Default: INPUT*

SDATAN2, GPIO26 — PCM Audio Input Data, General Purpose I/O

Digital-audio PCM data input. This pin can act as a general-purpose input or output that can be individually configured and controlled by DSPC. *BIDIRECTIONAL - Default: INPUT*

SDATAN3, GPIO27 — PCM Audio Input Data, General Purpose I/O

Digital-audio PCM data input. This pin can act as a general-purpose input or output that can be individually configured and controlled by DSPC. *BIDIRECTIONAL - Default: INPUT*

$\overline{SCS}$ — Host Serial SPI Chip Select

SPI mode active-low chip-select input signal. *INPUT*

SCCLK — Serial Control Port Clock

This pin serves as the serial SPI clock input. *INPUT*

SCDIN — SPI Serial Control Data Input

In SPI mode this pin serves as the data input pin. *INPUT*

SCDOUT, SCDIO — Serial Control Port Data Input and Output

In SPI mode this pin serves as the data output pin. *BIDIRECTIONAL - Default: OUTPUT in SPI mode*

$\overline{INTREQ}$ — Control Port Interrupt Request

Open-drain interrupt-request output. This pin is driven low to indicate that DSPC has outgoing control data and should be serviced by the host.
OPEN DRAIN I/O - Requires 3.3K Ohm Pull-Up

HDATA7, GPIO7 — DSPC Bidirectional Data Bus, General Purpose I/O

HDATA6, GPIO6

HDATA5, GPIO5

HDATA4, GPIO4

HDATA3, GPIO3

HDATA2, GPIO2

HDATA1, GPIO1

HDATA0, GPIO0

>   In parallel host mode, these pins provide a bidirectional data bus. These pins can also act as general purpose input or output pins that can be individually configured and controlled by DSPC. These pins have an internal pull-up. *BIDIRECTIONAL - Default: INPUT*

A0, GPIO13 — Host Parallel Address Bit 0, General Purpose I/O

>   In parallel host mode, this pin serves as the LS Bit of a two bit address input used to select one of four parallel registers. This pin can act as a general-purpose input or output that can be individually configured and controlled by DSPC. *BIDIRECTIONAL - Default: INPUT*

A1, GPIO12 — Host Address Bit 1, General Purpose I/O

>   In parallel host mode, this pin serves as the MS Bit of a two bit address input used to select one of four parallel registers. This pin can act as a general-purpose input or output that can be individually configured and controlled by DSPC. *BIDIRECTIONAL - Default: INPUT*

$\overline{\text{RD}}$, R/$\overline{\text{W}}$, GPIO11 — Host Parallel Output Enable, Host Parallel R/$\overline{\text{W}}$, General Purpose I/O

>   In Intel parallel host mode, this pin serves as the active-low data bus enable input. In Motorola parallel host mode, this pin serves as the read-high/write-low control input signal. This pin can act as a general-purpose input or output that can be individually configured and controlled by DSPC. This pin has an internal pull-up. *BIDIRECTIONAL - Default: INPUT*

$\overline{\text{WR}}$, $\overline{\text{DS}}$, GPIO10 — Host Write Strobe, Host Data Strobe, General Purpose I/O

>   In Intel parallel host mode, this pin serves as the active-low data bus enable input. In Motorola parallel host mode, this pin serves as the read-high/write-low control input signal. In serial host mode, this pin can serve as a general purpose input or output bit. This pin can act as a general-purpose input or output that can be individually configured and controlled by DSPC. This pin has an internal pull-up.
>   *BIDIRECTIONAL - Default: INPUT*

$\overline{\text{CS}}$, GPIO9 — Host Parallel Chip Select, General Purpose I/O

>   In parallel host mode, this pin serves as the active-low chip-select input signal. This pin can act as a general-purpose input or output that can be individually configured and controlled by DSPC. This pin has an internal pull-up. *BIDIRECTIONAL - Default: INPUT*

HINBSY, GPIO8 — Input Host Message Status, General Purpose I/O

This pin indicates that serial or parallel communication data written to the DSP has not been read yet. This pin can act as a general-purpose input or output that can be individually configured and controlled by DSPC. This pin has an internal pull-up. *BIDIRECTIONAL - Default: OUTPUT*

SD_DATA15, EXTA18 — SDRAM Data Bus, SRAM External Address Bus

SD_DATA14, EXTA17

SD_DATA13, EXTA16

SD_DATA12, EXTA15

SD_DATA11, EXTA14

SD_DATA10, EXTA13

SD_DATA9, EXTA12

SD_DATA8, EXTA11

SDRAM data bus 15:8. SRAM external address bus 18:11. *OUTPUT*

SD_DATA7, EXTD7 — SDRAM Data Bus, SRAM External Data Bus

SD_DATA6, EXTD6

SD_DATA5, EXTD5

SD_DATA4, EXTD4

SD_DATA3, EXTD3

SD_DATA2, EXTD2

SD_DATA1, EXTD1

SD_DATA0, EXTD0

SDRAM data bus 7:0. SRAM external data bus 7:0. *BIDIRECTIONAL - Default: INPUT*

SD_ADDR10, EXTA10 — SDRAM Address Bus, SRAM External Address Bus

SD_ADDR9, EXTA9

SD_ADDR8, EXTA8

SD_ADDR7, EXTA7

SD_ADDR6, EXTA6

SD_ADDR5, EXTA5

SD_ADDR4, EXTA4

SD_ADDR3, EXTA3

SD_ADDR2, EXTA2

SD_ADDR1, EXTA1

SD_ADDR0, EXTA0

SDRAM address bus 10:0. SRAM external address bus 10:0. *OUTPUT*

SD_CLK_OUT — SDRAM Clock Output

SDRAM clock output. *OUTPUT*

SD_CLK_IN — SDRAM Re-timing Clock Input

SDRAM re-timing clock input. *INPUT*

SD_CLK_EN — SDRAM Clock Enable

SDRAM clock enable. *OUTPUT*

SD_BA, EXTA19 — SDRAM Bank Address Select, SRAM External Address Bus

SDRAM bank address select. SRAM external address bus 19. *OUTPUT*

$\overline{\text{SD\_CS}}$ — SDRAM Chip Select

SDRAM chip select. *OUTPUT*

$\overline{\text{SD\_RAS}}$ — SDRAM Row Address Strobe

SDRAM row address strobe. *OUTPUT*

$\overline{\text{SD\_CAS}}$ — SDRAM Column Address Strobe

SDRAM column address strobe. *OUTPUT*

$\overline{\text{SD\_WE}}$ — SDRAM Write Enable

SDRAM write enable. *OUTPUT*

SD_DQM1 — SDRAM Data Mask 1

SDRAM data mask 1. *OUTPUT*

SD_DQM0 — SDRAM Data Mask 2

SDRAM data mask 0. *OUTPUT*

$\overline{\text{NV\_CS}}$, GPIO14 — SRAM Chip Select, General Purpose I/O

SRAM/Flash chip select. This pin can act as a general-purpose input or output that can be individually configured and controlled by DSPC. *BIDIRECTIONAL - Default: OUTPUT*

$\overline{\text{NV\_OE}}$, GPIO15 — SRAM Output Enable, General Purpose I/O

SRAM/Flash output enable. This pin can act as a general-purpose input or output that can be individually configured and controlled by DSPC. *BIDIRECTIONAL - Default: OUTPUT*

$\overline{\text{NV\_WE}}$, GPIO16 — SRAM Write Enable, General Purpose I/O

SRAM/Flash write enable. This pin can act as a general-purpose input or output that can be individually configured and controlled by DSPC. *BIDIRECTIONAL - Default: OUTPUT*

UHS2, CS_OUT, GPIO17 — Mode Select Bit 2, External Serial Memory Chip Select, General Purpose I/O

DSPC control port mode select bit 2. This pin is sampled at the rising edge of $\overline{\text{RESET}}$ and is one of three pins used to select the control port mode. In serial control port mode, this pin can serve as an output to provide the chip-select for a serial EEPROM. This pin can act as a general-purpose input or output that can be individually configured and controlled by DSPC. *BIDIRECTIONAL - Default: INPUT*

UHS0, GPIO18 — Mode Select Bit 0, General Purpose I/O

DSPC control port mode select bit 0. This pin is sampled at the rising edge of $\overline{\text{RESET}}$ and is one of three pins used to select the control port mode. This pin can act as a general-purpose input or output that can be individually configured and controlled by DSPC.
BIDIRECTIONAL - Default: INPUT

UHS1, GPIO19 — Mode Select Bit 1, General Purpose I/O

DSPC control port mode select bit 1. This pin is sampled at the rising edge of $\overline{\text{RESET}}$ and is one of three pins used to select the control port mode. This pin can act as a general-purpose input or output that can be individually configured and controlled by DSPC.
BIDIRECTIONAL - Default: INPUT

GPIO20 — General Purpose I/O

This pin can act as a general-purpose input or output that can be individually configured and controlled by DSPC. This pin has an internal pull-up.
BIDIRECTIONAL - Default: INPUT

GPIO21 — General Purpose I/O

This pin can act as a general-purpose input or output that can be individually configured and controlled by DSPC.This pin has an internal pull-up.
BIDIRECTIONAL - Default: INPUT

VDD[7:1] — 2.5V Supply Voltage

2.5V supply voltage.

VSS — 2.5V Ground

2.5V ground.

NC[5:1] — No Connect

     Recommended tie to ground.

VDDSD[4:1] — 3.3V SDRAM/SRAM/EPROM Interface Supply

     3.3V SDRAM/SRAM/EPROM supply.

VSSSD — 3.3V SDRAM/SRAM/EPROM Interface Ground

     3.3V ground.

13. ORDERING INFORMATION

CS494002-CQ 144-pin, accommodates SRAM/SDRAM
CS494502-CQ 100-pin, external SRAM memory interface only (no SDRAM), no parallel-control ports, no FLASH programming.
(Contact the factory for the 100-pin package pin-out and dimension drawing)
Temp Range 0-70º C for both parts

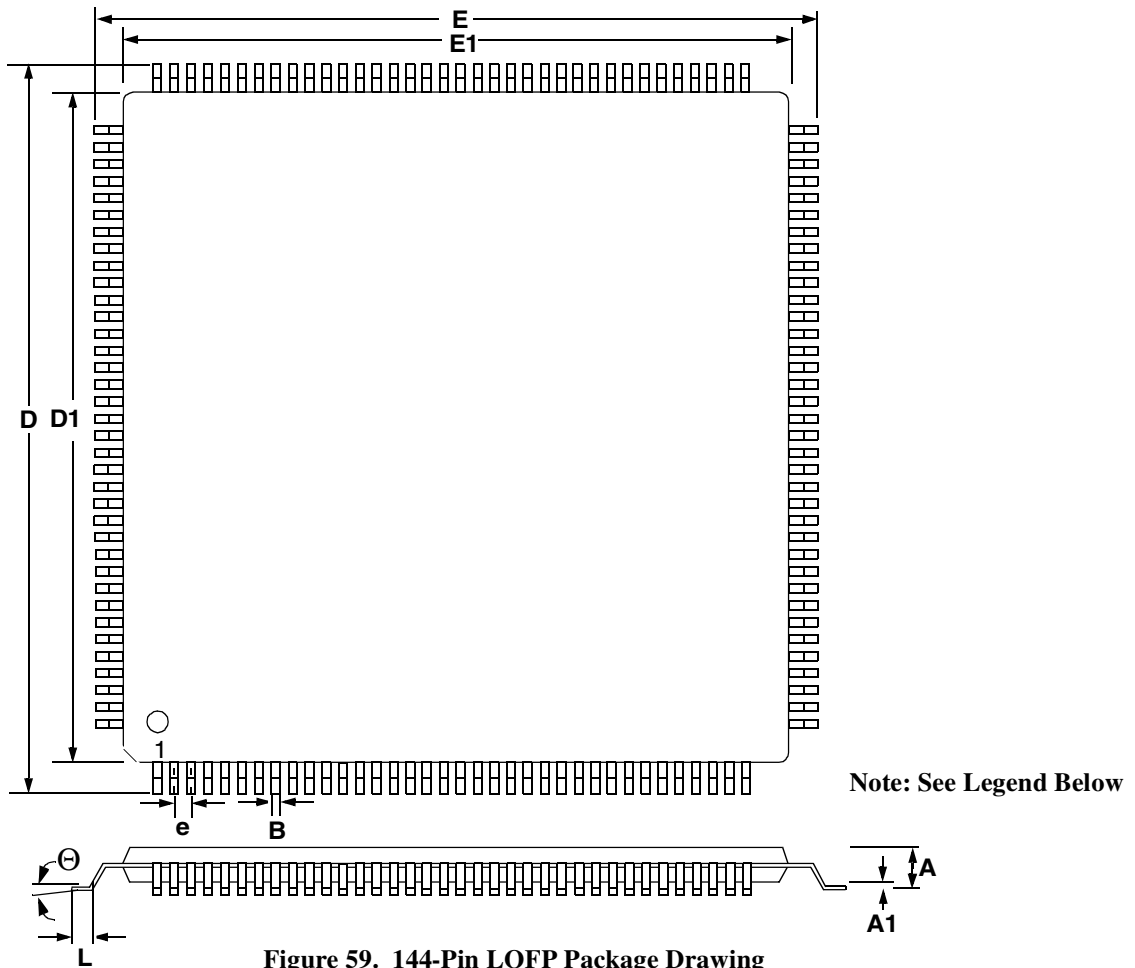## 14. PACKAGE DIMENSIONS
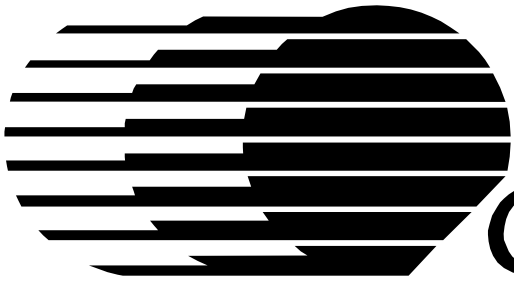
14.1 144-Pin LQFP Package

Note: See Legend Below

Figure 59. 144-Pin LQFP Package Drawing

DIM	INCHES			MILLIMETERS
	MIN	NOM	MAX	MIN
A	---	0.55	0.063	---
A1	0.002	0.004	0.006	0.05
B	0.007	0.008	0.011	0.17
D	0.854	0.866 BSC	0.878	21.70
D1	0.783	0.787 BSC	0.791	19.90
E	0.854	0.866 BSC	0.878	21.70
E1	0.783	0.787 BSC	0.791	19.90
e	0.016	0.020	0.024	0.40
Θ	0.000°	4°	7.000°	0.00°
L	0.018	0.024	0.030	0.45

CIRRUS LOGIC®