CRYSTAL
A Cirrus Logic Company

CS4920A

Multi-Standard Audio Decoder - DAC

Features

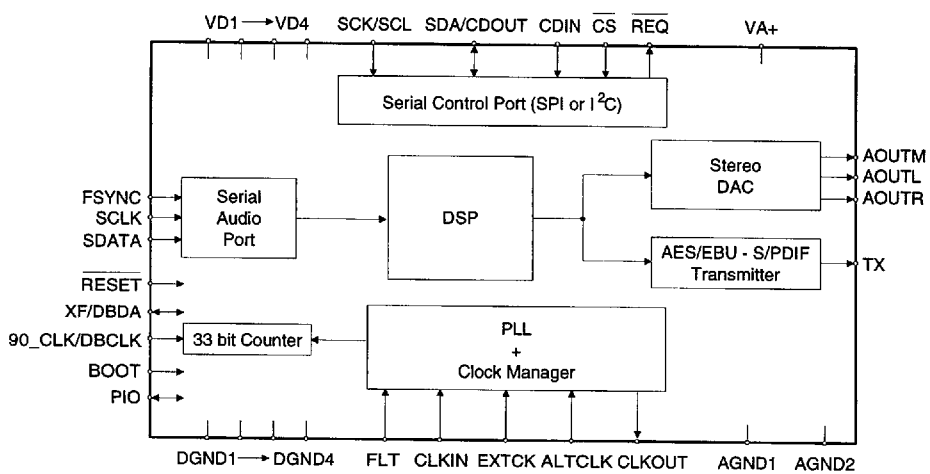
- General Purpose Digital Signal Processor Optimized for Audio
  - 24 Bit Fixed Point
  - 48 Bit Accumulator
  - 12.3 MIPS @ 48 kHz Sample Rate
- On-Chip Functional Blocks Include:
  - -CD Quality D/A Converters
  - -Programmable PLL Clock Multiplier
  - -AES/EBU - S/PDIF Compatible Digital AudioTransmitter
  - -Audio Serial Input Port
  - -Serial Control Port
- Applications Include:
  - -Audio Decompression
  - -MPEG 1 and 2 Layers I, II
- Standard 44 pin PLCC Package

General Description

The CS4920A is a complete audio subsystem on a chip. This device contains a general purpose DSP, a CD quality stereo Digital-to-Analog Converter, a programmable PLL clock multiplier, an AES-EBU - S/PDIF compatible digital audio transmitter, an audio serial input port, and a serial control port. The CS4920A is based on a programmable DSP core and is intended to support a wide variety of digital signal processing applications which include decoding compressed digital audio. Serial audio data broadcast on networks such as cable TV, direct broadcast satellite TV, or the telephone system can be decompressed and converted to standard analog or digital signals.

Both industry standard and proprietary DSP algorithms can be supported. Software which performs industry standard MPEG 1 and 2 layers I, II is available. A complete set of software development tools are available. These include an assembler, simulator, and debugger.

ORDERING INFORMATION:
CS4920A-CL 44-pin PLCC
CDB4920/20A/21 Evaluation Board



Preliminary Product Information | This document contains information for a new product. Crystal Semiconductor reserves the right to modify this product without notice.

Crystal Semiconductor Corporation
P.O. Box 17847, Austin, TX 78760
(512) 445 7222 Fax: (512) 445 7581

TABLE OF CONTENTS

ANALOG CHARACTERISTICS (T_A = 25 °C; VA+, VD+ = 5V; Full-Scale Output Sinewave, 1 kHz; Word Clock = 48 kHz (PLL in use); Logic 0 = GND, Logic 1 = VD+; Measurement Bandwidth is 20 Hz to 20 kHz; Local components as shown in "Typical Connection Diagram"; SPI mode, I^2S audio data; unless otherwise specified.)

Parameter*		Symbol	Min	Typ	Max	Units
Dynamic Performance						
DAC Resolution			16	-	-	Bits
DAC Differential Nonlinearity		DNL	-	-	±0.9	LSB
Total Harmonic Distortion	AOUTL, AOUTR (Note 1) AOUTM	THD	-	0.01 0.02	0.015 0.03	%
Instantaneous Dynamic Range (DAC not muted, A weighted)	AOUTL, AOUTR (Note 1) AOUTM	IDR	82 76	85 79	-	dB
Interchannel Isolation	(Note 1)		-	85	-	dB
Interchannel Gain Mismatch			-	-	0.2	dB
Frequency Response	(10 to 0.476 Fs)		-3.0	-	+0.2	dB
Full Scale output Voltage	(Note 1)		2.66	2.88	3.1	Vpp
Gain Drift			-	100	-	ppm/°C
Deviation from Linear Phase			-	-	5	Deg
Out of Band Energy	(Fs/2 to 2Fs)		-	-60	-	dB
Analog Output Load	Resistance: Capacitance:		8 -	- -	- 100	kΩ pF
Power Supply						
Power Supply Rejection	(1 kHz)		-	40	-	dB
Power Supply Consumption	VA+ VD+ VA+ (powerdown) VD+ (powerdown)		- - - -	20 120 - -	40 140 0.15 2.0	mA mA mA mA

Notes: 1. 10 kΩ, 100pF load for each analog signal (Left, Right, Mono).

D/A INTERPOLATION FILTER CHARACTERISTICS

(See graphs toward the end of this data sheet)

Parameter		Symbol	Min	Typ	Max	Units
Passband (to -3 dB corner)	(Fs is conversion freq.)		0	-	0.476Fs	Hz
Passband Ripple			-	-	±0.1	dB
Transition Band			0.442Fs	-	0.567Fs	Hz
Stop Band			≥0.567Fs	-	-	Hz
Stop Band Rejection			50	-	-	dB
Stop Band Rejection with Ext. 2Fs RC filter			57	-	-	dB
Group Delay			-	12/Fs	-	s

* Refer to *Parameter Definitions* at the end of this data sheet.

Specifications are subject to change without notice.

ABSOLUTE MAXIMUM RATINGS (AGND, DGND = 0V, all voltages with respect to ground.)

Parameter		Symbol	Min	Max	Units
DC Power Supplies:	Positive Digital	VD+	-0.3	6.0	V
	Positive Analog	VA+	-0.3	6.0	V
	| | VA+ | - | VD+ | |		-	0.4	V
Input Current, Any Pin Except Supplies		I_{in}	-	±10	mA
Digital Input Voltage		V_{IND}	-0.3	(VD+) + 0.4	V
Ambient Operating Temperature (power applied)		T_{Amax}	-55	125	°C
Storage Temperature		T_{stg}	-65	150	°C

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

RECOMMENDED OPERATING CONDITIONS

(AGND, DGND = 0V; all voltages with respect to ground.)

Parameter		Symbol	Min	Typ	Max	Units
DC Power Supplies:	Positive Digital	VD+	4.50	5.0	5.50	V
	Positive Analog	VA+	4.50	5.0	5.50	V
	| | VA+ | - | VD+ | |		-	-	0.4	V
Ambient Operating Temperature		T_A	0	-	70	°C

DIGITAL CHARACTERISTICS

(T_A = 25 °C; VA+, VD+ = 5V ± 10%; measurements performed under static conditions.)

Parameter	Symbol	Min	Typ	Max	Units
High-Level Input Voltage	V_{IH}	4.5	-	-	V
Low-Level Input Voltage	V_{IL}	-	-	0.5	V
High-Level Output Voltage at I_O = -2.0 mA	V_{OH}	4.5	-	-	V
Low-Level Output Voltage at I_O = 2.0 mA	V_{OL}	-	-	0.5	V
Input Leakage Current	I_{in}	-	-	1.0	µA

SWITCHING CHARACTERISTICS - CLOCKS

(T_A = 25 °C; VA+, VD+ = 5V; Inputs: Logic 0 = DGND, Logic 1 = VD+, C_L = 20pF)

Parameter			Symbol	Min	Typ	Max	Units
Master Clock Frequency			CLKIN	0.032	27	30	MHz
Alternate Clock	P = 0	(Note 2)	ALTCLK	-	256Fs	-	Hz
	P = 1			-	384Fs	-	Hz
Clock Output			CLKOUT	-	-	6.4	MHz

Notes: 2. ALTCLK is required to run at 256 or 384 times the sample frequency.

SWITCHING CHARACTERISTICS - EXTERNAL FLAG

(T_A = 25 °C; VA+, VD+ = 5V; Inputs: Logic 0 = DGND, Logic 1 = VD+, C_L = 20pF)

Parameter	Symbol	Min	Typ	Max	Units
Rise time of XF	t_{rxf}			200	ns
Fall time of XF	t_{fxf}			100	ns

SWITCHING CHARACTERISTICS - PROGRAMMABLE INPUT/OUTPUT

(T_A = 25 °C; VA+, VD+ = 5V; Inputs: Logic 0 = DGND, Logic 1 = VD+, C_L = 20pF)

Parameter	Symbol	Min	Typ	Max	Units
I_O = 0					
Input Frequency	f_{pio}			350	kHz
Risetime of PIO	t_{rpio}			200	ns
Fall time of PIO	t_{fpio}			200	ns
I_O = 1					
Rise time of PIO	t_{rpo}			200	ns
Fall time of PIO	t_{fpo}			200	ns

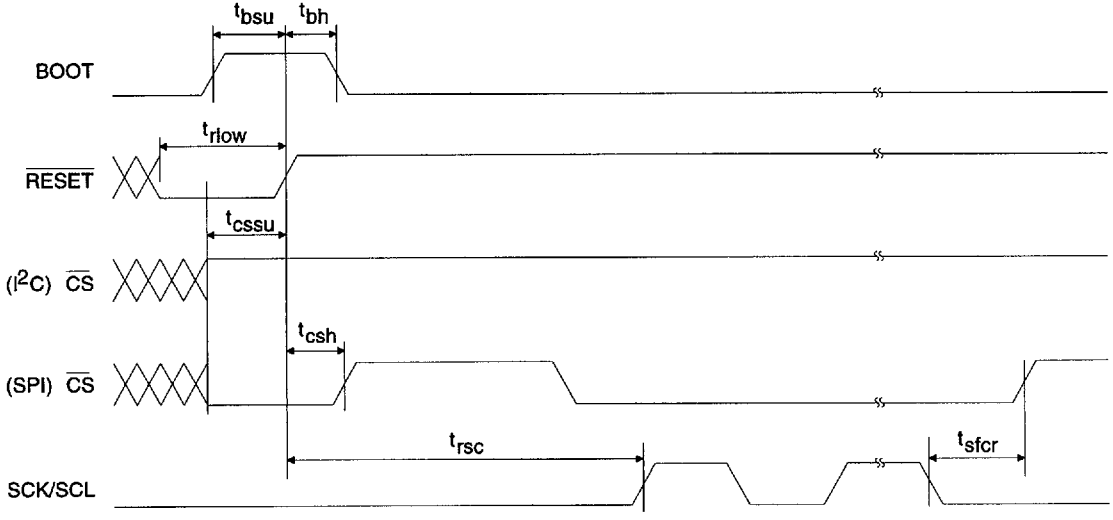
SWITCHING CHARACTERISTICS - BOOT INITIALIZATION

(T_A = 25 °C; VA+, VD+ = 5V; Inputs: Logic 0 = DGND, Logic 1 = VD+, C_L = 20pF)

Parameter		Symbol	Min	Max	Units
BOOT Setup Time to $\overline{\text{RESET}}$ Rising		t_{bsu}	300	-	ns
$\overline{\text{RESET}}$ Rising to Boot Hold Time		t_{bh}	450	-	ns
$\overline{\text{CS}}$ Setup Time to $\overline{\text{RESET}}$ Rising	(Note 4)	t_{cssu}	200	-	ns
$\overline{\text{RESET}}$ Rising to $\overline{\text{CS}}$ Hold Time		t_{csh}	200	-	ns
$\overline{\text{RESET}}$ Low Time		t_{rlow}	50	-	µs
SCK/SCL Delay Time from $\overline{\text{RESET}}$ Rising	(Note 3)	t_{rsc}	2	-	ms
SCK/SCL falling to $\overline{\text{CS}}$ rising on last byte of download		t_{sfcr}	3	-	µs

Notes: 3. This delay is necessary after any rising edge of RESET to allow time for the part to initialize and for the on-board PLL to stablize.

4. The mode of the Serial Control Port is selected by $\overline{\text{CS}}$. $\overline{\text{CS}}$ = 1 is $I^2C^®$. $\overline{\text{CS}}$ = 0 is SPI mode.

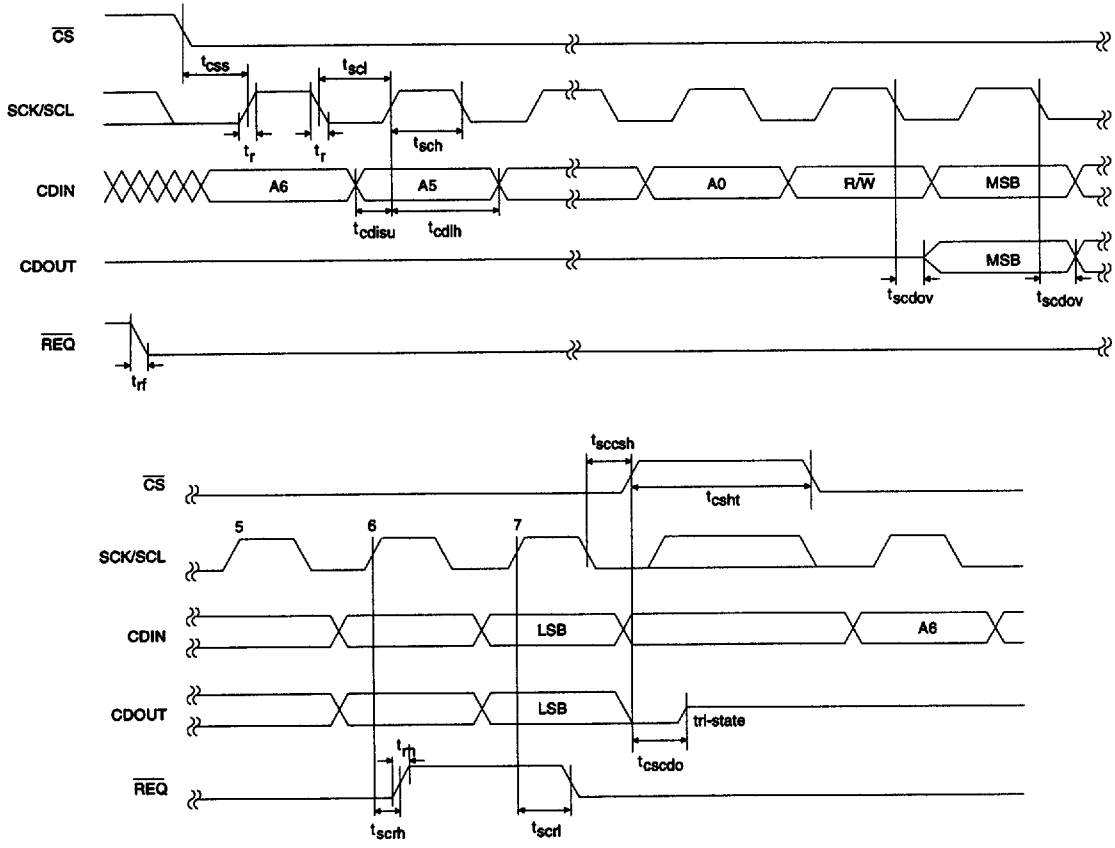


SWITCHING CHARACTERISTICS - CONTROL PORT

(T_A = 25 °C; VA+, VD+ = 5V; Inputs: Logic 0 = DGND, Logic 1 = VD+, C_L = 20pF)

Parameter		Symbol	Min	Max	Units
SPI Mode ($\overline{CS}$ = 0)					
SCK/SCL Clock Frequency	(slow mode) (fast mode)	f_{sck} f_{sck}	- -	350 2000	kHz
$\overline{CS}$ Falling to SCK/SCL Rising	(slow mode)	t_{css}	20	-	ns
Rise Time of Both CDIN and SCK/SCL Lines	(slow mode)	t_r	-	50	ns
Fall Time of Both CDIN and SCK/SCL Lines	(slow mode) (fast mode)	t_f t_f	- -	300 50	ns ns
SCK/SCL Low Time	(slow mode) (fast mode)	t_{scl} t_{scl}	1100 150	- -	ns ns
SCK/SCL High Time	(slow mode) (fast mode)	t_{sch} t_{sch}	1100 150	- -	ns ns
Setup Time CDIN to SCK/SCL Rising	(slow mode) (fast mode)	t_{cdisu}	250 50	- -	ns ns
Hold Time SCK/SCL Rising to CDIN	(Note 5)	t_{cdih}	0	-	µs
Transition Time from SCK/SCL to CDOUT Valid	(Note 6)	t_{scdov}	-	40	ns
Time from SCK/SCL Rising to $\overline{REQ}$ Rising	(Note 7)	t_{scrh}	-	200	ns
Rise Time for $\overline{REQ}$	(Note 7)	t_{rr}	-	50	ns
Fall Time for $\overline{REQ}$	(Note 8)	t_{rf}	-	20	ns
Hold Time for $\overline{REQ}$ from SCK/SCL Rising	(Note 8)	t_{scrl}	0	-	ns
Time from SCK/SCL Falling to $\overline{CS}$ Rising		t_{sccsh}	20	-	ns
High Time Between Active $\overline{CS}$		t_{csht}	200	-	ns

Notes:
5. Data must be held for sufficient time to bridge 300(50) ns transition time of SCK/SCL.
6. CDOUT should NOT be sampled during this time period.
7. $\overline{REQ}$ will only go HIGH if there is no data in SCPOUT at the rising edge of SCL/SCK during a READ operation as shown. DSP frequency is 20 MHz. Pull-up resistor is 2 kΩ. C_L = 20 pF.
8. If $\overline{REQ}$ went HIGH as indicated in note 7, then $\overline{REQ}$ will hold high at least until the next rising edge of SCK/SCL. If data is in SCPOUT at this time $\overline{REQ}$ will go active LOW again. This condition should be treated as a new READ process. Address and R/$\overline{W}$ bit should be sent again.



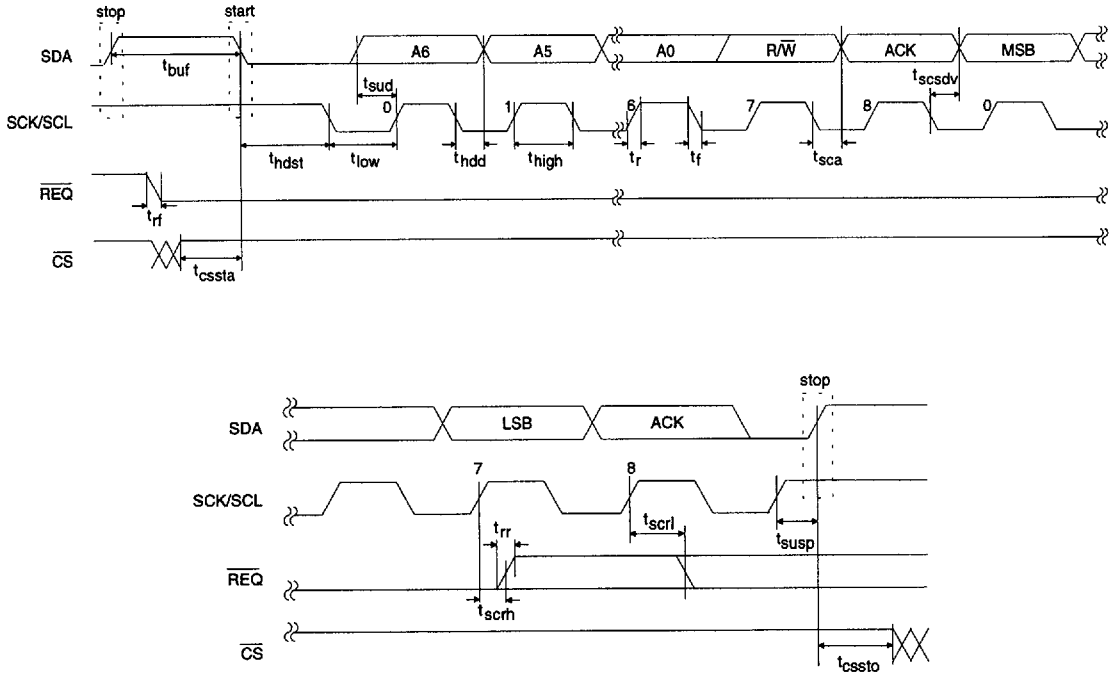
SPI Control Port Timing

SWITCHING CHARACTERISTICS - CONTROL PORT

(T_A = 25 °C; VA+, VD+ = 5V; Inputs: Logic 0 = DGND, Logic 1 = VD+, C_L = 20pF)

Parameter		Symbol	Min	Max	Units
I^2C Mode ($\overline{CS}$=1)	(Note 9)				
SCK/SCL Clock Frequency	(slow mode) (fast mode)	f_{scl}		100 400	kHz
Bus Free Time Between Transmissions		t_{buf}	4.7		µs
Start Condition Hold Time (prior to first clock pulse)		t_{hdst}	4.0		µs
Clock Low Time		t_{low}	4.7		µs
Clock High Time		t_{high}	4.0		µs
SDA Setup Time to SCK/SCL Rising		t_{sud}	250		ns
SDA Hold Time from SCK/SCL Falling	(Note 10)	t_{hdd}	0		µs
Rise Time of Both SDA and SCK/SCL	(Note 11)	t_r		50	ns
Fall Time of Both SDA and SCK/SCL		t_f		300	ns
Time from SCK/SCL Falling to CS4920 ACK		t_{sca}		40	ns
Time from SCK/SCL Falling to SDA Valid During READ Operation		t_{scsdv}		40	ns
Time from SCK/SCL Rising to $\overline{REQ}$ Rising	(Note 12)	t_{scrh}		200	ns
Hold Time for $\overline{REQ}$ from SCK/SCL Rising	(Note 13)	t_{scrl}	0		ns
Rise Time for $\overline{REQ}$	(Note 12)	t_{rr}		50	ns
Fall Time for $\overline{REQ}$	(Note 13)	t_{rf}		20	ns
Setup Time for Stop Condition		t_{susp}	4.7		µs
Hold Time for $\overline{CS}$ from Stop Condition	(Note 14)	t_{cssto}	20		ns
Setup Time for $\overline{CS}$ to Start Condition	(Note 14)	t_{cssta}	20		ns

Notes:
9. Use of I^2C bus® compatible interface requires a license from Philips. I^2C® is a registered trademark of Philips Semiconductor.
10. Data must be held for sufficient time to bridge the 300ns transition time of SCK/SCL.
11. This rise time is shorter than the I^2C® specifications recommend, please refer to the section on SCP communications for more information.
12. $\overline{REQ}$ will only go HIGH if there is no data in the SCPOUT register at the rising edge of SCL/SCK during a READ operation as shown. DSP frequency is 20 MHz. Pull-up resistor is 2 kΩ C_L = 20pF.
13. If $\overline{REQ}$ went HIGH as indicated in Note 12 then $\overline{REQ}$ will hold HIGH at least until the next rising edge of SCK/SCL. If data is in the SCPOUT register at this time $\overline{REQ}$ will go active LOW again. This condition should be treated as a new READ process. The address and R/$\overline{W}$ should be sent again following a new START condition.
14. Most I^2C® applications will connect $\overline{CS}$ to VD+. It is not necessary to have control of $\overline{CS}$ in this case.



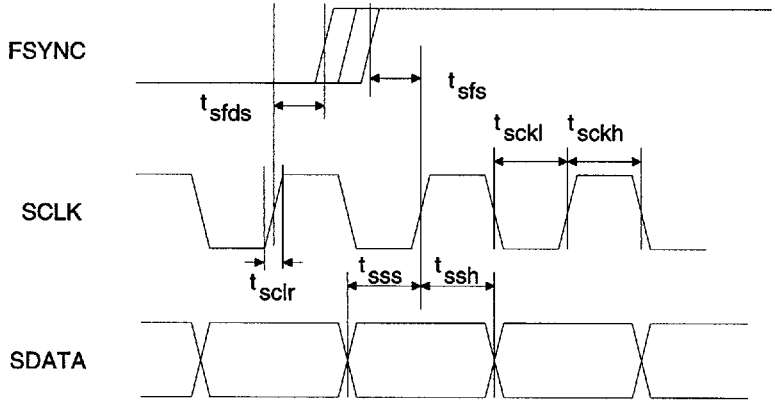
I²C® Control Port Timing

SWITCHING CHARACTERISTICS - SERIAL AUDIO PORT

(T_A = 25 °C; VA+, VD+ = 5V; Inputs: Logic 0 = GND, Logic 1 = VD+; C_L = 20 pF)

Parameter		Symbol	Min	Typ	Max	Units
SCLK Frequency			-	-	12.5	MHz
SCLK Pulse Width Low		t_{sckl}	25	-	-	ns
SCLK Pulse Width High		t_{sckh}	25	-	-	ns
SCLK rising to FSYNC edge delay	(Note 15)	t_{sfds}	20	-	-	ns
SCLK rising to FSYNC edge setup	(Note 15)	t_{sfs}	20	-	-	ns
SDATA valid to SCLK rising setup	(Note 15)	t_{sss}	20	-	-	ns
SCLK rising to SDATA hold time	(Note 15)	t_{ssh}	20	-	-	ns
Rise time of SCLK		t_{sclr}	-	-	20	ns

Notes: 15. The table above assumes data is output on the falling edge and latched on the rising edge.
The SCLK edge is selectable in setting the EDG bit in the ASICN register. The diagram is for EDG = 1.



Serial Audio Port Timing

THEORY OF OPERATION

Introduction

The CS4920A is a complete audio subsystem on a chip. It consists of a general-purpose Digital Signal Processor (DSP), and a number of supplementary analog and digital blocks. These supplementary blocks include a programmable PLL clock multiplier, a serial audio input port, a CD quality stereo Digital-to-Analog Converter (DAC), an AES/EBU - S/PDIF compatible digital audio transmitter, and a serial control port. Figure 1 shows a typical connection diagram for the CS4920A in which a micro controller is used for loading the program code.

The CS4920A is based on a programmable DSP core. A typical application is the decoding of a compressed digital audio signal. Serial audio data broadcast on networks such as cable TV, direct broadcast satellite TV, or the telephone system can be decompressed and converted to standard analog and digital signals. A wide variety of standard and proprietary decompression algorithms can be supported. An assembler, a simulator, and a debugger are available. CS4920A DSP code is available which performs industry standard MPEG 1 and 2 layers I and II.

The DSP has a 24-bit fixed point data path, 4K words of program RAM, and 2K words of data RAM. The execution unit has a 48-bit accumulator, and no input data registers. Typical ALU instructions read operands from memory and store results back to memory. Modulo and bit reverse addressing are supported. For a sample rate of 48 kHz, the DSP can provide 12.3 MIPS.

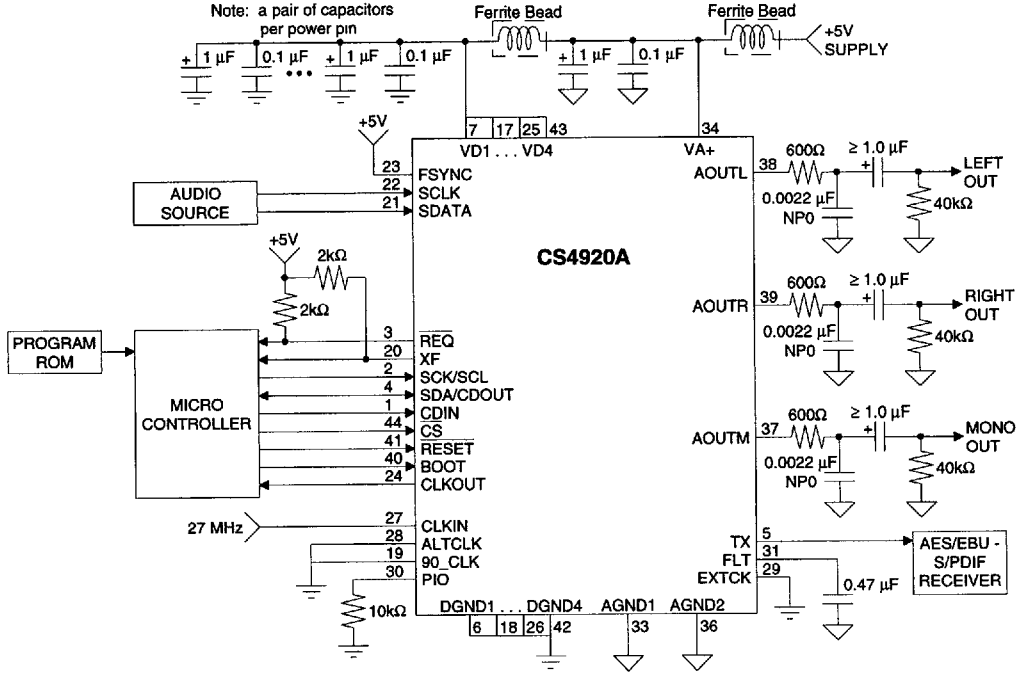


Figure 1. Typical Connection Diagram

The CS4920A includes a flexible clock manager. This section allows clock inputs on CLKIN to range from 32 kHz to 30 MHz, while producing the necessary DSP and DAC clocks through a programmable PLL.

The digital audio data is input through the serial audio port. Various formats are possible with the availability of three signal inputs and an internal control register.

For analog reproduction of the digital input, a stereo DAC using delta-sigma architecture is built-in. Switched-capacitor filters perform most of the reconstruction process. Only a simple external passive filter is needed to complete reconstruction.

In addition to the analog output, an AES/EBU - S/PDIF compatible output is provided. This allows the designer the flexibility of transmitting the audio data in a standard digital format to an external system.

To facilitate the downloading of DSP code to the CS4920A, a serial control port, communicating in either $I^2C^®$ or SPI format, is used. This port may also be used in real time to issue control commands to the DSP.

PERIPHERALS

Five on-chip peripherals make the audio decoder ideal for decoding broadcast digital audio signals. It has a PLL clock manager, a CD quality DAC, a digital audio transmitter, a three pin serial port for inputting audio data, and an SPI/$I^2C^®$ port for serial control information. Each peripheral has I/O mapped data, control, and status registers. Many can also generate interrupts. A serial debug peripheral is provided to allow easy debugging of code.

Clock Manager

The clock manager is primarily a clock multiplier circuit that takes CLKIN input of any frequency from 32 kHz to 30 MHz and produces all the internal clocks required to run the DSP and the audio peripherals.

At the heart of the clock manager circuit is two PLL (Phase Lock Loop) circuits (Figure 2). The first PLL produces the 128Fs (P bit of CM0 is 0) or 192Fs (P bit of CM0 is 1) clock. The output of the first PLL will clock both the on-board DACs and the second PLL. The second PLL is designed to quickly track the output of the first PLL and produce the DSP clock. The DSP clock produced by the second PLL is 4x128Fs (or

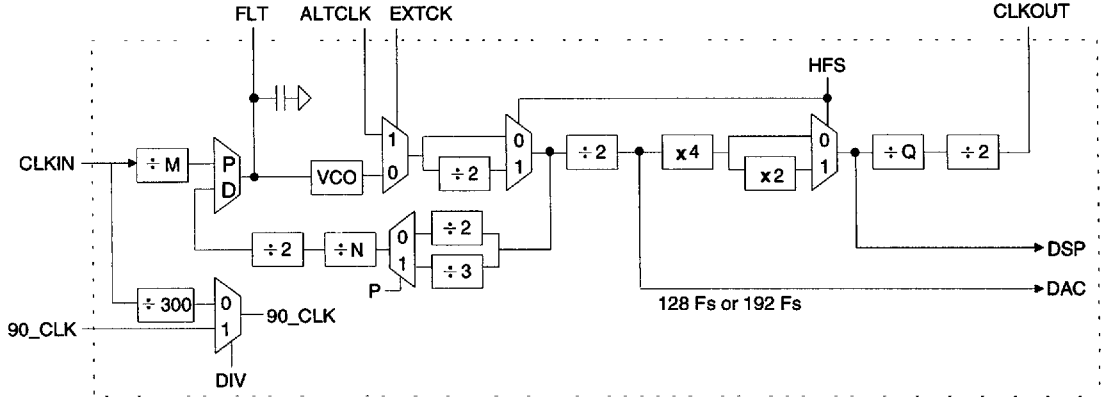


Figure 2. Clock Manager

4x192Fs) when the HFS bit of CM0 is 0. When the HFS bit is set to 1 the net effect is that the DSP clock will be 8x128Fs (or 8x192Fs).

The first PLL generates the 128Fs (128 times the sample rate) or 192Fs by dividing the CLKIN input frequency by M and multiplying by N. The CM1 register stores the 12 bit M value and the 10 bit N value. The frequency range of the first PLL is 128 x 8 kHz to 128 x 50 kHz, and allows sample rate frequencies in range of 8 kHz to 50 kHz. The second PLL produces DSP clock frequencies between 4.1 and 25.6 MHz. Typical CLKIN input frequencies and the proper N/M ratio to generate 128Fs or 192Fs are shown in Table 1 based on the equation:

$$\frac{CLKIN}{M} = \frac{64Fs}{N} \qquad \text{EQ. 1}$$

CLKIN Frequency (kHz)	Fs	32 kHz	44.1 kHz	48 kHz
		128 Fs or 192Fs (N/M)		
32		64/1	441/5	96/1
48		128/3	294/5	64/1
56		256/7	252/5	384/7
64		32/1	441/10	48/1
96		64/3	147/5	32/1
112		128/7	126/5	192/7
128		16/1	441/20	24/1
160		64/5	441/25	96/5
192		32/3	147/10	16/1
256		8/1	441/40	12/1
320		32/5	441/50	48/5
384		16/3	147/20	8/1
320		32/5	441/50	48/5
448		32/7	63/10	48/7
10752		4/21	21/80	2/7
13500		512/3375	392/1875	256/1125
27000		256/3375	196/1875	128/1125
30000		128/1875	294/3125	64/625

Table 1. Ratios for 64Fs based on CLKIN Frequency

A typical CLKIN input frequency is the compressed bit rate of a MPEG audio stream. Many other combinations are possible including CLKIN equal to 27 MHz with audio sample rates equal to 32 kHz, 44.1 kHz, and 48 kHz and their respective half sample rates.

Both PLLs of the CS4920A have lock detection circuitry to indicate to the DSP when either PLL is not producing the correct frequency based on the values of M, N, and P. The first PLL uses two 8 bit counters that are compared every time one counter rolls over. One counter is clocked by the reference frequency input to the phase detector, CLKIN / M, and the other counter is clocked by the 64 Fs / N feedback clock. Designs that rely on the lock detection circuitry should take into account that the first PLL lock detector output is updated once every 256 counts at the rate of the reference frequency. The second PLL lock detection circuitry is based on signals that come directly from the phase detector circuit of the second PLL. The output of both PLL's lock detection circuitry is NAND'd together to produce the $\overline{\text{LOCK}}$ bit of the LINT register.

There are two internal VCOs in the CS4920A (one for each PLL). The circuitry of the second PLL's VCO is completely inside the CS4920A, while the first PLL's VCO circuit requires a capacitor to be connected to the FLT pin (PIN 31). The typical value of the FLT capacitor is 0.47µF, which is sufficient for all allowable CLKIN input frequencies. However, increased analog performance may be achieved by selecting a different value of capacitance based on the

	CLKIN frequency				
	32 kHz	10.752 MHz	13.5 MHz	27 MHz	30 MHz
Capacitance (µF)	0.47	0.1	0.47	0.22	0.22

Table 2. FLT Capacitor versus CLKIN frequency

CLKIN input frequency. Table 2 shows some typical CLKIN frequencies and the FLT capacitance best suited for the frequency. It must be stressed that the best analog performance can only be achieved by placing the capacitor as close as possible to the FLT pin and that the proper layout precautions be taken to avoid noise coupling onto the FLT pin. In addition the capacitor values in Table 2 are only valid when using the N/M ratios of Table 1.

The external clock (EXTCK) pin specifies the function of the ALTCLK. When EXTCK is low, the internal VCO of the first PLL is used to provide the 128Fs (192Fs) required. If EXTCK is high, an external oscillator connected to ALTCLK can be used to generate the 128Fs (192Fs) clock. Note the frequency of ALTCLK must be 256Fs or 384Fs due to the divide by 2 following the multiplexer controlled by EXTCK. This divide by 2 ensures a 50% duty cycle for the 128Fs (192Fs) clock generation. When an external VCO is used, the FLT pin drives an external loop filter. The filter output controls the VCO frequency and the VCO becomes the 256Fs (384Fs) clock.

CLKOUT is a divided version of the DSP clock. The DSP clock is divided by a programmable divider and an additional divide by 2 before being output. The divider value is stored in the ten bit register Q. The divide by two guarantees a 50% duty cycle. The frequency of CLKOUT can vary from the DSP frequency divided by 4 to the DSP frequency divided by 4096. CLKOUT can be used to synchronize external devices or generate most compressed bit rate clocks.

When the slow (SLW) control bit is low, the control voltage of the first PLL is pulled low, and therefore, the output of the first PLL will be approximately 2.4 MHz. The second PLL always tracks the first PLL. After a reset, SLW must be set before the first PLL will lock. When both PLLs lock, the $\overline{\text{LOCK}}$ status bit in the Long Interrupt Register (LINT) goes low. If either PLL loses lock, $\overline{\text{LOCK}}$ goes high. A low to high transition of $\overline{\text{LOCK}}$ generates interrupt 3 if the lock interrupt enable (LKIEN) bit is high.

The FS status bit is a 50% duty cycle sample rate clock. Transitions of this bit generate interrupt 0. The interrupt vector for a rising edge is 0003_H and the interrupt vector for a falling edge is 0002_H. This interrupt is used to write right and left audio data to the DAC register and the transmitter (XMT) register.

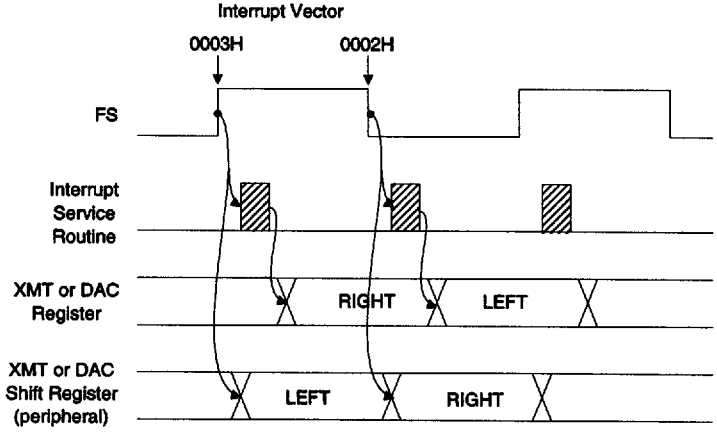


Figure 3. Transmitter Sequencing

Loading of the DAC and XMT registers, and their respective shift registers, is initiated by FS transitions. Figure 3 shows the relationships. The left audio data is loaded into the shift registers on the rising edge of FS and the right audio data is loaded on the falling edge. Interrupt 0 occurs just after these shift registers are loaded. Right audio data should be loaded into the DAC and XMT registers by the instruction at 0003_H and left audio data should be loaded by the instruction at 0002_H.

FS is generated by dividing the 128Fs (192Fs) clock by 128 (192). This divider is reset when the FSRS control bit is high. The divider begins counting when FSRS goes low. This allows the programmer to control the phase relationship between input signals (FSYNC for instance) and FS.

33-bit counter

The 33-bit-counter can be used to support MPEG synchronization of audio and video. This loadable counter is targeted to operate at 90kHz. The 90kHz clock may be derived from a 27MHz master clock provided at CLKIN (if available) or from a 90kHz clock provided at Pin 19 90_CLK/DBCLK. The selection of the counter clock is made via the register bit DIV in register CM0. When set, the DIV bit divides the clock at CLKIN by 300 and provides the divided clock to 33-bit-counter.

The 33-bit-counter is implemented with two additional I/O registers. A 24 bit count register (CNT24) contains the high order bits, and a 9 bit count register (CNT9) contains the low order bits. Since the 90kHz clock can be asynchronous with the internal DSP clock, the counter values may not be valid when the DSP reads the registers. A VALID bit has been added to the 9 bit count register. When VALID is set the counter value is stable. If the VALID bit is set, the internal timing guarantees that the counter will not change state within the next two DSP instruction periods. This allows the program to read the low order 9 bit counter value without concern of ripple counting the low word to the high word.

The counter is reset by writing a value into the two counter registers. In order to preset the counter, the user should load the low order 9 bit counter value followed by the high order 24 bit counter value.

A SCREN bit in register CM0 is used to enable the counter function. When enabled, the debug port functions are replaced with SCR counter functions. Pin 19 switches from the debug clock source to the counter clock source; Pin 20 switches from the debug data pin to an external flag function. The SCREN register value is initialized to zero after RESET. Therefore, the associated pins are used for debug as a default condition.

The clock manager has two control registers, Clock Manager 0 (CM0) and Clock Manager 1 (CM1). Both registers are read/write except for the FS bit of CM0 which is read only. The following describes each bit.

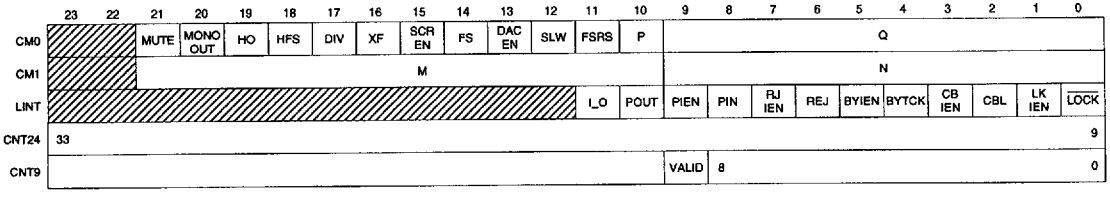


Figure 4. Clock Manager Register Bit Map

Clock Manager 0 (CM0)

P: This bit is used to determine the oversample ratio of the DACs. When high, sample rate = 192Fs. When low, sample rate = 128Fs.

Q: Ten bit value which specifies the divide ratio between the DSP clock and CLKOUT.

FSRS: Sample rate clock reset. The sample rate clock, FS, is generated by the clock manager. The phase relationship between FS and external clocks such as FSYNC can be controlled by FSRS.

SLW: Slow. When low, the VCO of the first PLL's control voltage is pulled low. This produces a low frequency out of the VCO.

DACEN: DAC enable provides a zero value pattern to the DACs when this bit is low. When high, the audio to the DACs is enabled.

HFS: DSP clock doubler. Must be set to zero for normal operation.

FS: Read only status bit which is a 50% duty cycle sample rate clock.

HO: Hold Off. When set high the second PLL is forced to hold at a low frequency. The second PLL cannot lock to the first PLL when this bit is high.

Clock Manager 1 (CM1)

N: Ten bit value which specifies the multiplier of the N/M ratio output to generate the 128Fs (192Fs) clock.

M: Twelve bit value used to divide the CLKIN input.

Long Interrupt Register (LINT)

$\overline{\text{LOCK}}$: Read only status bit which is low when the PLL is locked.

LKIEN: Lock interrupt enable. A rising edge of the $\overline{\text{LOCK}}$ status bit generates an interrupt if this bit is high.

CNT 24

CNT24: The register contains the upper 24 bits of the 33 bit counter. This register should be written to after CNT9.

CNT9

CNT9: Bits 0 through 8 contain the lower 9 bits of the 33 bit counter. This register should be written to before CNT24.

VALID: Read only status bit which is high when the 33 bit counter is guaranteed not to change state for at least the next two instruction cycles.

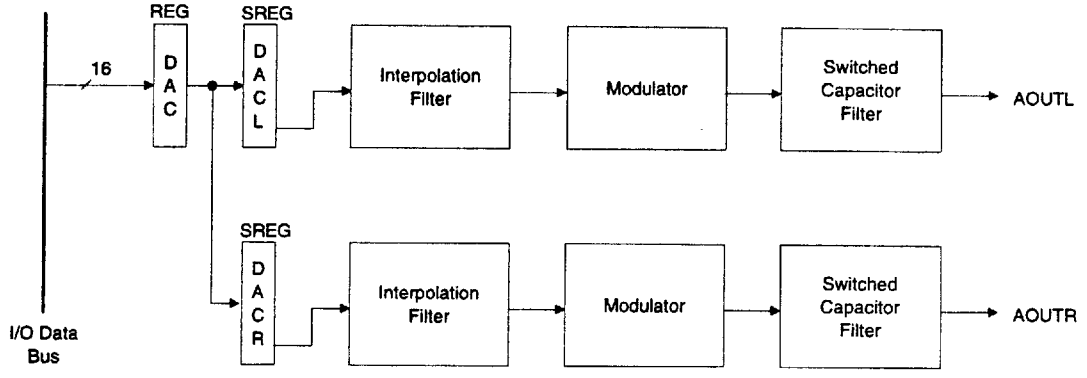


Figure 5. DAC

Digital to Analog Converter

The digital to analog converter (DAC) is a dual channel CD quality DAC. It is designed with delta sigma architecture. The baseband audio is interpolated to 128Fs (192Fs) before going into the modulator. The modulator is third order and is followed by a 1 bit DAC/switch capacitor filter stage. An external passive filter completes the reconstruction process. The output is single ended with a drive capability down to 8 kΩ. Figure 5 is a block diagram of the DAC.

The interpolation filter produces images which are attenuated by at least 56dB from .584Fs to 128Fs (192Fs). At a 48 kHz sample rate, a full scale signal at 20 kHz will produce an image at 28 kHz which is attenuated by more than 60dB.

The out-of-band quantization noise from the delta sigma modulator extends from .417Fs to 128Fs (192Fs). This noise is attenuated by the switch capacitor filter and the continuous time filters. The total quantization noise and thermal noise from the analog filters integrated over the .417Fs to 128Fs (192Fs) is more than 50dB below full scale power.

Left and right audio data are written to the DAC output register (Figure 6). This is a 16 bit write only register. The high 16 bits of the I/O data bus can be written to this register. The DACL and DACR output shift registers must be updated at the sample rate determined by the clock manager block. The FS status bit and the interrupt that it can generate can be used to time the updates of these registers.

During Power Down, the internal voltage reference is powered off. Therefore, the outputs of the DACs will not be driven during this period. The recovery of the reference from Power Down is sufficiently slow to prevent the outputs from "popping". Feeding all zeros to the DAC will produce a signal level of typically 2.1 volts.

Digital Audio Transmitter

The transmitter encodes digital audio data according to the Sony Philips Digital Interface Format (S/PDIF) or the AES/EBU interface format. The encoded data is output on the TX pin. More information on the S/PDIF and AES/EBU standards are available in the applications notes

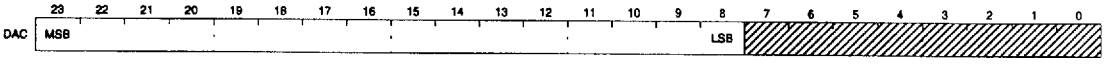


Figure 6. Digital to Analog Converter Register Bit Map

found in the back of this data book. The block diagram of the transmitter is shown in Figure 7.

The transmitter has a 24 bit write only register for audio data (XMT), a 16 bit read/write register for channel status data (XMTCS), and a read/write control register (XMTCN). The register bit maps are shown in Figure 8. The audio and channel status data are read from their registers and multiplexed with the validity and user bits from the control register, and the internally generated parity bit.

Channel status data can be input in two different modes determined by CSMD in register XMTCN. In the first mode, XMTCS stores the 16 most important channel status bits according to the S/PDIF standard. These are bits 0-5, 8-15, 24, and 25. Bit 0 must be low, this defines the consumer format for the channel status. Bit 1 defines whether the information being transferred is audio or non-audio data. Bit 2 is the copy bit. Bits 3 through 5 are the emphasis bits. Bits 8 through 15 define the category code and whether the data is from an original or copied source. Bits 24 and 25 define the sample frequency. A more detailed description of these bits is available in the application notes mentioned earlier in this section.

The XMTCS register must be loaded once by the programmer and is read once per block by the transmitter. All other bits are transmitted as zero. The LSB of the XMTCS is the LSB of the channel status bits.

The CBL status bit in LINT goes high at a channel status block boundary. XMTCS is loaded into the shift register by the transmitter at the same time. Just after this transition of CBL, interrupt 3 will be generated if the CBL interrupt enable (CBIEN) bit is set. If the interrupt is enabled, CBL is cleared only by reading LINT. If the interrupt is disabled, CBL transitions low af-

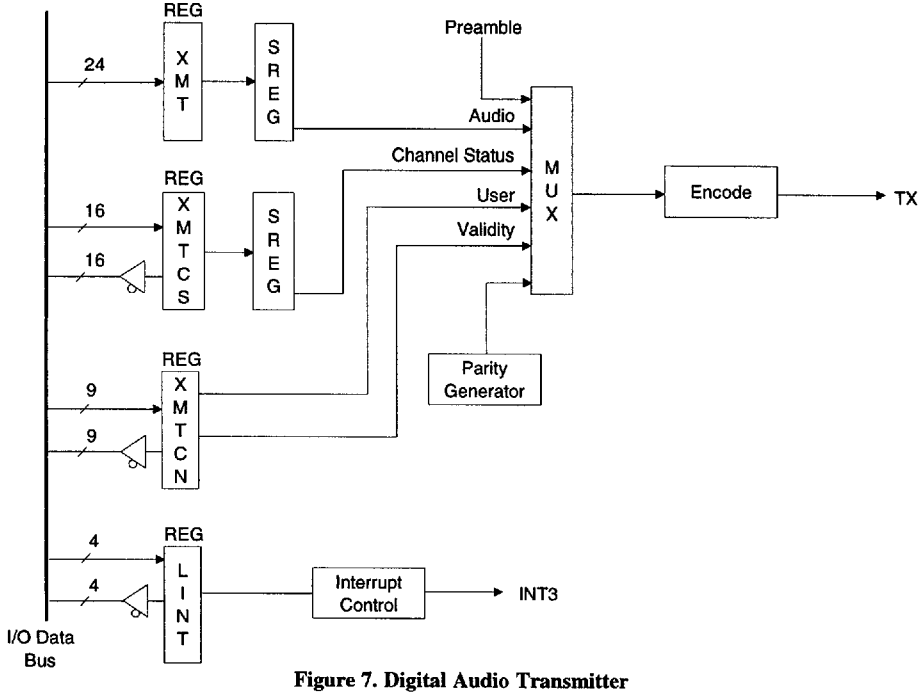


Figure 7. Digital Audio Transmitter

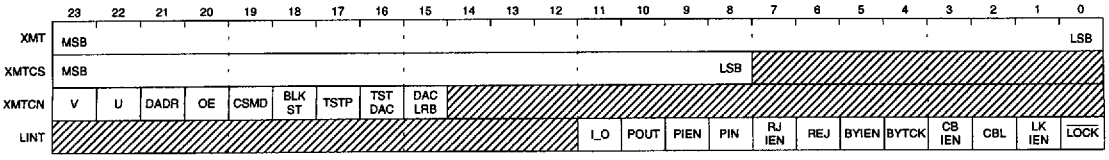


Figure 8. Digital Audio Transmitter Register Bit Map

ter four left and four right bytes of channel status has been transmitted.

In the second channel status mode, all the bits in a block can be controlled. XMTCS is loaded every 16 subframes and is serially shifted into 16 transmitted subframes. This allows independent control of both channels.

The BYTCK status bit in LINT transitions high at a block boundary and every 16 subframes afterwards. XMTCS is loaded into the shift register by the transmitter at the same time. Just after this transition of BYTCK, interrupt 3 will be generated if the BYTCK interrupt enable (BYIEN) bit is set. If the interrupt is enabled, BYTCK is cleared only by reading LINT. If the interrupt is disabled, BYTCK transitions low after eight subframes of channel status have been transmitted.

The XMT register is loaded into the shift register of the transmitter at twice the sample rate specified in the clock manager. The right channel is loaded into the shift register on the falling edge of the FS status bit and the left channel is loaded on the rising edge. The XMT register is timed to be loaded by the transmitter at the same time that the DAC reads the DAC register. Interrupt 0 occurs just after a transition of FS. The interrupt vector address is different for rising and falling edges. Short interrupts can be used to write left and right audio data to the XMT register.

Audio data can be written to both DAC and XMT in the same instruction if the DADR bit in the transmit control register (XMTCN) is high. This causes XMT to respond to both DAC and XMT register addresses.

The validity (V) and user (U) bits in XMTCN are read by the transmitter at the same time XMT is read. These bits are transmitted with the audio data.

XMTCN is a read/write control register. A description of each bit follows:

Transmitter control register (XMTCN)

V: Validity bit.

U: User bit.

DADR: DAC address. When high, XMT responds to the addresses of DAC and XMT.

OE: Output Enable. When high, TX is enabled. When it is low, TX is low.

CSMD: Channel Status Mode. When low, XMTCS is read once per block. When high, XMTCS is read every 16 subframes.

BLKST: Block Start. A low to high transition specifies a new channel status block boundary.

TSTP: Test Mode. Must be set to zero for normal operation.

TSTDAC: Test DAC. When high, digital output from DAC is output through TX. Normal data transmission is disabled.

DACLRB: Left/Right bit of DAC. Special test bit intended for test purposes only.

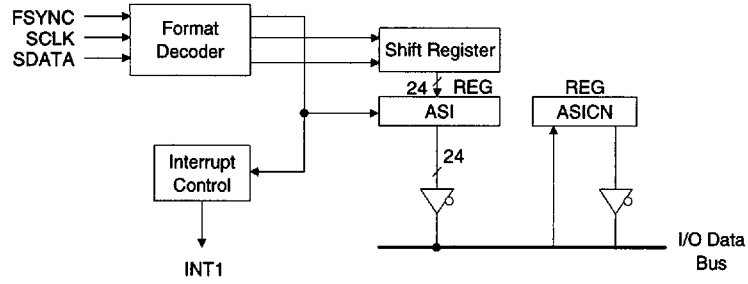


Figure 9. Audio Serial Input Port

Long Interrupt Register (LINT)

BYIEN: BYTCK Interrupt Enable. When high, a low to high transition of BYTCK generates interrupt 3.

CBIEN: CBL Interrupt Enable. When high, a low to high transition of CBL generates an interrupt 3.

BYTCK: Byte Clock. Status bit which is the channel status byte clock. It is high for 8 subframes and low for 8 subframes. See text for acknowledgment protocol of interrupts.

CBL: Channel status Block clock. Status bit which goes high at the block boundary and low 64 subframes later. See text for acknowledgment protocol of interrupts.

Audio Serial Input Port

The audio serial input port has a three pin interface consisting of FSYNC, SCLK, and SDATA. SCLK clocks SDATA (serial data input) into the 24 bit input shift register. The contents of this shift register can be loaded into the Audio Serial Input (ASI) register by transitions of FSYNC or by a counter time out. An interrupt can be generated when ASI is loaded. Figure 9 shows a block diagram of the audio port.

The pulse mode (PUL) control bit in the audio serial port control register (ASICN) specifies whether both edges (PUL=0) or one edge (PUL=1) of FSYNC loads ASI. When configured to load on both edges, normal mode, up to 24 bits of data after a transition of FSYNC are clocked into the shift register. The next transition of FSYNC loads ASI with the contents of the shift register. Any number of SCLK periods can occur between FSYNC edges; the first 24 bits (or less) are accepted.

When the serial port is configured to load ASI on only one FSYNC edge, pulse mode, any number of SCLK periods can occur between active edges. ASI will be loaded on the active edge of FSYNC and every 16 or 24 SCLK periods after an active edge of FSYNC. The CNT16 bit in ASICN selects between 16 and 24 SCLK's. If FSYNC never toggles, ASI will be continuously loaded every 16 or 24 SCLK periods.

The serial data is typically entered into the port MSB first. If at least 24 SCLK's occur between the times that ASI is loaded, the MSB of the input data will be loaded into the MSB of ASI. If the number of SCLK's between the times that ASI is loaded equals 24 minus N, the MSB will be loaded into the MSB minus N location in ASI. Shifting in software may be required to align the data.

The delay (DEL) bit in ASICN shifts the timing between FSYNC and SDATA by one SCLK period. When DEL is low, the MSB of the data in should occur immediately following a transition of FSYNC. When DEL is high, the MSB should

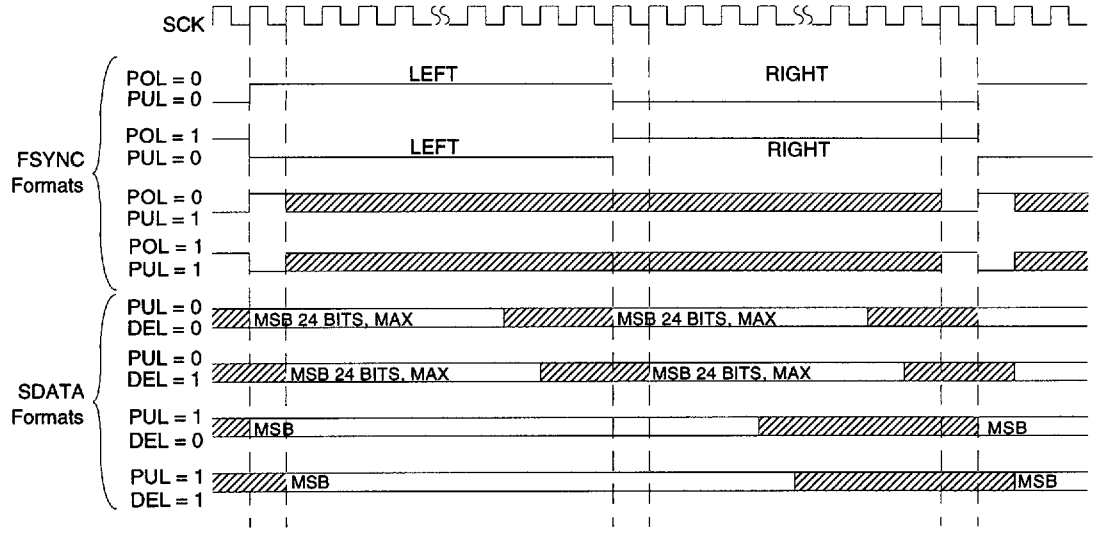


Figure 10. Audio Serial Input Formats

occur one SCLK period after a transition of FSYNC.

The edge (EDG) bit internally inverts SCLK. When EDG is low, the falling edge of SCLK latches SDATA into the shift register. When EDG is high, the rising edge of SCLK latches SDATA.

The polarity (POL) bit in ASICN inverts FSYNC. This switches the active edge of FSYNC in pulse mode. When not in pulse mode, FSYNC can be used to identify left and right channels of stereo audio data. When POL is low, FSYNC high identifies the left channel and when POL is high, FSYNC high identifies the right channel. Figure 10 shows the timing relationships of the various formats.

Interrupt 1 (INT1) is asserted every time ASI is loaded if the interrupt enable (IEN) bit in the Control Register is high and the interrupt mask ($\overline{\text{MSK1}}$) is high. The interrupt vector address is determined by the state of FSYNC just after ASI is loaded. When the port is configured to load ASI on both edges of FSYNC, the interrupt vector for the interrupt just after a left data word is loaded into ASI is 0004_H. The interrupt vector for the interrupt just after a right data word is loaded is 0005_H. When the port is in pulse mode, the interrupt vector for an interrupt generated by a transition of FSYNC is 0004_H. The interrupt vector for an interrupt generated by the count out controlled by CNT16 is 0005_H.

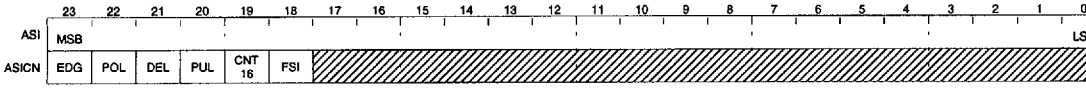


Figure 11. Audio Serial Input Port Register Bit Map

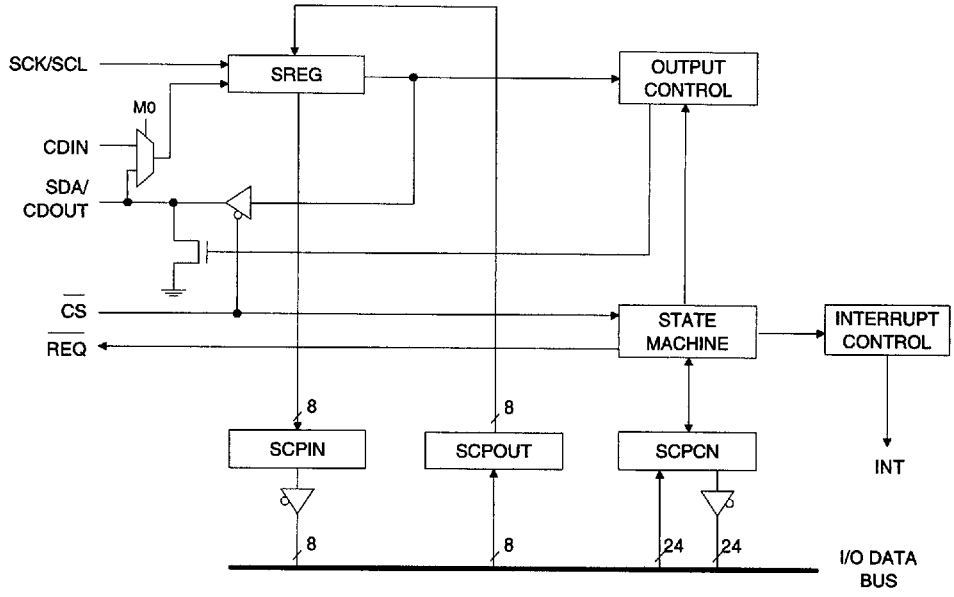


Figure 12. Serial Control Port

Audio Serial Input Control Register (ASICN)

EDG: Edge. Specifies the SCLK edge which clocks SDATA. When low, the falling edge clocks data in.

POL: Polarity. Specifies the polarity of FSYNC. In normal mode, when POL is low, FSYNC high identifies the left channel. When POL is high, FSYNC high identifies the right channel. In pulse mode, when POL is low, the rising edge of FSYNC is active. When POL is high, the falling edge of FSYNC is active.

DEL: Delay. When DEL is low, there is no delay between an edge of FSYNC and the MSB of the audio data. When DEL is high, there is a one SCLK cycle delay between the edge of FSYNC and the MSB of the audio data.

PUL: Pulse mode. When PUL is low, FSYNC is a left/right signal. When PUL is high, FSYNC identifies the start of a frame but does not distinguish between left and right samples.

CNT16: Count 16. When high, ASI is loaded after 16 SCLK's. When low, ASI is loaded after 24 SCLK's.

FSI: Fsync internal. If the FSYNC pin is tied low and the POL bit is set low, this bit can be used to simulate FSYNC's function. A transition of this bit (controlled by software) from low to high will load the ASI register with the contents of the shift register.

Serial Control Port

The serial control port (SCP) can operate in $I^2C^{®}$ or SPI compatible modes. In either mode, the control port performs eight bit transfers and is always configured as a slave. As a slave, it cannot drive the clock signal nor initiate data transfers. The port can request to be serviced by activating the $\overline{REQ}$ pin. The port is an asynchronous interface which provides interrupts and handshaking signals to allow communication between the on-chip DSP and an off-chip device such as a micro controller. Figure 12 shows a block diagram of the port.

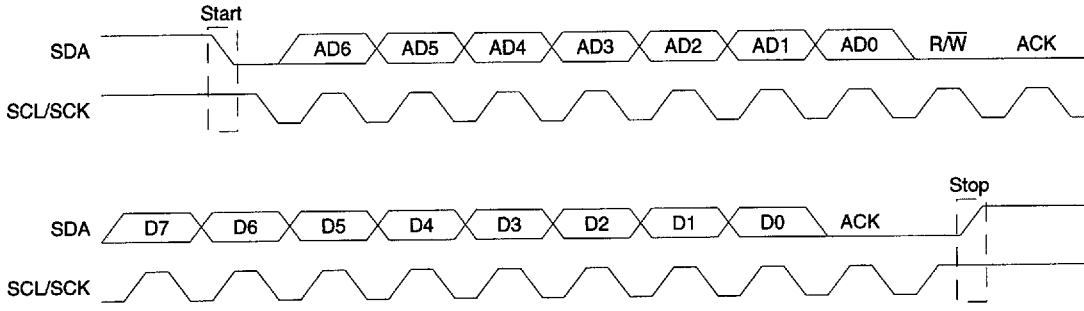


Figure 13a. Control Port Timing, I²C® Write

I²C® mode

The operating mode of the port is determined by the state of the M0 bit in the serial control port control register (SCPCN) as seen in Figure 16. If M0 is low, the port is I²C® compatible. M0 is loaded during a hardware or software reset, as well as controllable by microcode. The status of $\overline{CS}$ sets the mode of the SCP during a hardware and software reset. If $\overline{CS}$ is high during a reset the mode is I²C®. Note that in most systems where I²C® is the preferred control mode, $\overline{CS}$ is connected to the digital supply.

For normal I²C® operation SCL/SCK, SDA, and $\overline{REQ}$ are used. $\overline{CS}$ and CDIN are typically connected to the digital supply. SCL/SCK is the serial clock input which is always driven by an external device. SDA is the serial data Input/Output signal. $\overline{REQ}$ is the active low request signal, which is driven low when there is valid data in the serial control port output SCPOUT register (shown in Figure 16).

As an I²C® compatible port, data is communicated on the SDA pin and is clocked by the rising edge of SCL/SCK. The Signetics I²C® bus specification provides details of this interface. Note the CS4920A does not meet the rise time specification of the SCL/SCK signal. For more details please refer to the section on Rise Time of SCL/SCK.

Figure 13a shows the relative timing necessary for an I²C® write operation for a single byte. A 'write' is defined as the transfer of data from an I²C® bus master to the CS4920A serial control port. A transfer is initiated with a start condition followed by a 7 bit address and a read/write bit (set low for a write). This address is the address assigned to the device being written to during the transfer. In the case of the CS4920A, this address is stored in the SCPCN register. Immediately following power up, the CS4920A's Address checking Enable (AEN) bit is set to zero. The AEN bit must be set high for the CS4920A to compare the address of the intended I²C® device on the bus to its internal address. This means the CS4920A will respond to any address on the I²C® bus until its address is initialized and address checking is enabled. To avoid bus conflicts the CS4920A should be held is reset ($\overline{RESET}$ active low) until the master is ready to communicate with the CS4920A and sets the address in the SCPCN. The address can only be set using the I²C® bus interface, so the master should use the intended I²C® address when downloading microcode to the CS4920A to avoid conflict with other devices on the bus. Once the microcode is loaded into the CS4920A the microcode should either initialize the I²C® address or provide a means for the master to program the I²C® address. If the CS4920A is the only device on the I²C® bus, address checking is optional. However, I²C® bus protocol is still required. In other words, the address bits and read/write bit are still required.

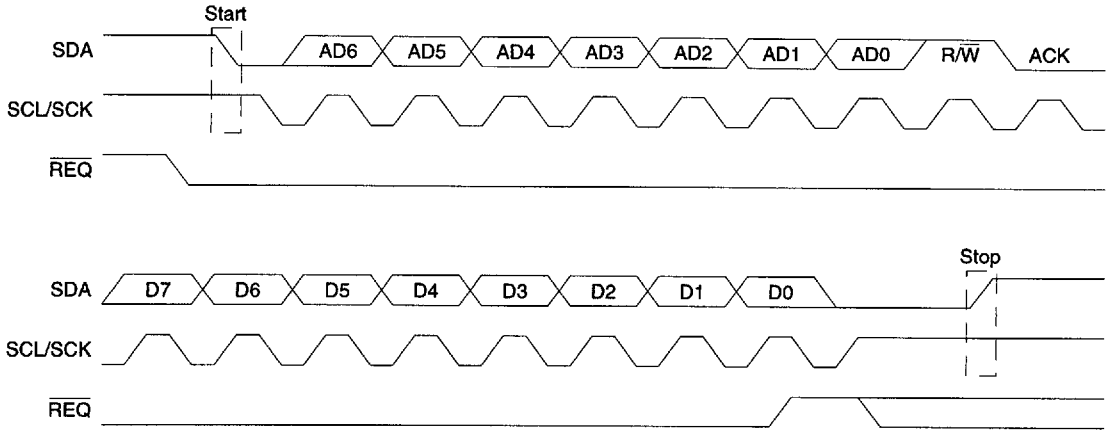


Figure 13b. Control Port Timing, I²C® Read

If a write to the CS4920A is specified, 8 bits of data on SDA will be shifted into the input shift register as shown in Figure 12. When the shift register is full, the 8 bit data is transferred to the Serial Control Port Input (SCPIN) register on the falling edge of the 8th data bit. An acknowledge (ACK) is sent back to the master and the input ready flag (IRDY) flag is set. This flag generates an interrupt on interrupt line 2 if the input ready interrupt enable (IRIEN) bit is set high. The interrupt vector is 0006H.

The master can continue to send data, but it will be rejected if the IRDY has not yet been cleared. This flag is cleared by reading the SCPIN register. If a byte is rejected, the reject (REJ) flag in the Long Interrupt (LINT) register is set. A rising edge of the REJ flag generates an interrupt on interrupt line 3 if the reject interrupt enable (RJIEN) bit is set high. The REJ flag is cleared by reading SCPCN. If the CS4920A fails to ACK it is possible that the byte was rejected and it should be transmitted again. If the second attempt fails the CS4920A should be issued a hardware reset to reinitialize the communication path.

If the DSP core of the CS4920A wants to send a byte to the master, it first writes the byte to the Serial Control Port Output (SCPOUT) register. Note the DSP only sends 8 bits per transfer to the SCPOUT register. A write to the SCPOUT sets the request pin ($\overline{REQ}$) active low and the output ready (ORDY) bit low. The master must recognize the request and issue a read operation to the DSP. Figure 13b shows the relative timing of a single byte read. The master must send the 7 bit address (if address checking is enabled it must match the address in the SCPCN register) and the read bit. For I²C® protocol, it is always the device receiving the transfer that must ACK. Therefore, the CS4920A will ACK the address and the read bit. After the ACK by the CS4920A. (the falling edge of SCL/SCK), the serial shift register is loaded with the byte to be sent and the most significant bit is placed on the SDA line. In addition, the ORDY is set high. A rising edge of the ORDY bit will generate an interrupt on interrupt line 2 if the output ready interrupt enable (ORIEN) bit is set high. The interrupt vector is 0007H.

The 8 bit value in the serial shift register is shifted out by the master. The data is valid on the rising edge of SCL/SCK and transitions immediately following the falling edge. For I²C® , the $\overline{REQ}$ line will be de-asserted immediately following the rising edge of the last data bit, of

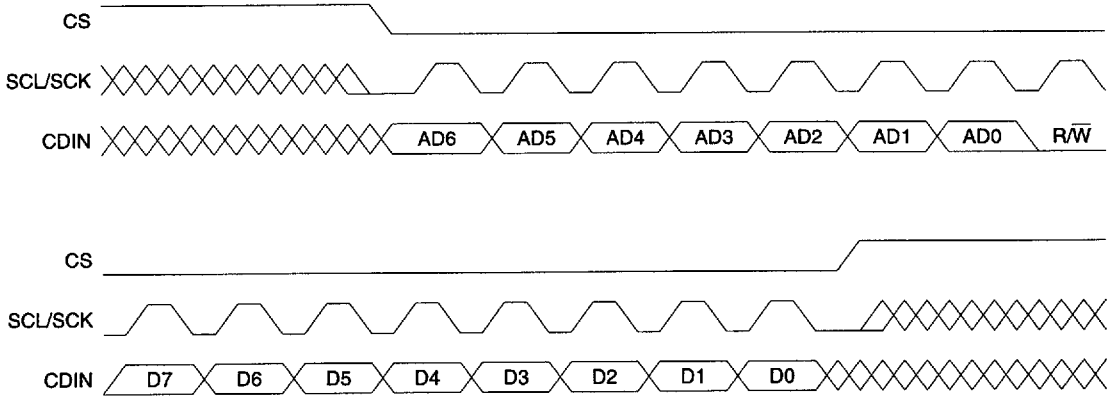


Figure 14a. Control Port Timing, SPI Write

the current byte being transfered, if there is no data in the SCPOUT register. The $\overline{REQ}$ line is guaranteed to stay de-asserted (high) until the rising edge of the SCL/SCK for the ACK. This signals the host that the transfer is complete.

If there is data placed in SCPOUT prior to the rising edge of SCL/SCK for the last data bit, then $\overline{REQ}$ will remain asserted (low). Immediately following the falling edge of SCL/SCK for the ACK, the new data byte will be loaded into the serial shift register. The host should continue to read this new byte. It is importat to note that once the data is in the shift register, clocks on the SCL/SCK line will shift the data bits out of the shift register. A STOP condition on the bus will not prevent this from occuring. The host must read the byte prior to any other bus activity or the data will be lost.

If data is placed in SCPOUT after the rising edge of SCL/SCK for the last data bit, but before the rising edge of SCL/SCK for the ACK, $\overline{REQ}$ will not be asserted until after the rising edge of SCL/SCK for the ACK. This should be treated as a completed transfer. The data written to SCPOUT will not be loaded into the shift register on the falling edge of SCL/SCK for the ACK. Therefore, a new read operation is required to read this byte.

Rise Time on SCL/SCK

The Signetics $I^2C^®$ bus specification allows for rise times of the SCL/SCK line up to 1 µs. The CS4920A does not meet this specification. If the $I^2C^®$ bus master(s) has a rise time in excess of 50 ns the CS4920A will be unable to reliably communicate across the bus. In some systems a stronger pull-up resistor on the SCL/SCK line will provide the rise time needed for proper operation, but this is only helpful when the current rise time is near 50 ns. In cases where the CS4920A will be used in a system where a longer rise time on SCL/SCK is expected, a CMOS compatible buffer should be used. Figure 15 shows the necessary connections. Note the buffer is only used for the SCL/SCK connecting directly to the CS4920A. Other devices on the $I^2C^®$ bus may need to hold SCL/SCK low while accepting data.

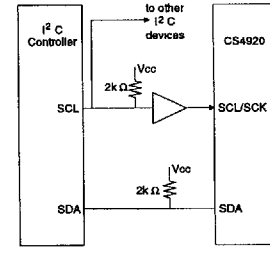


Figure 15. I2C Connection Diagram

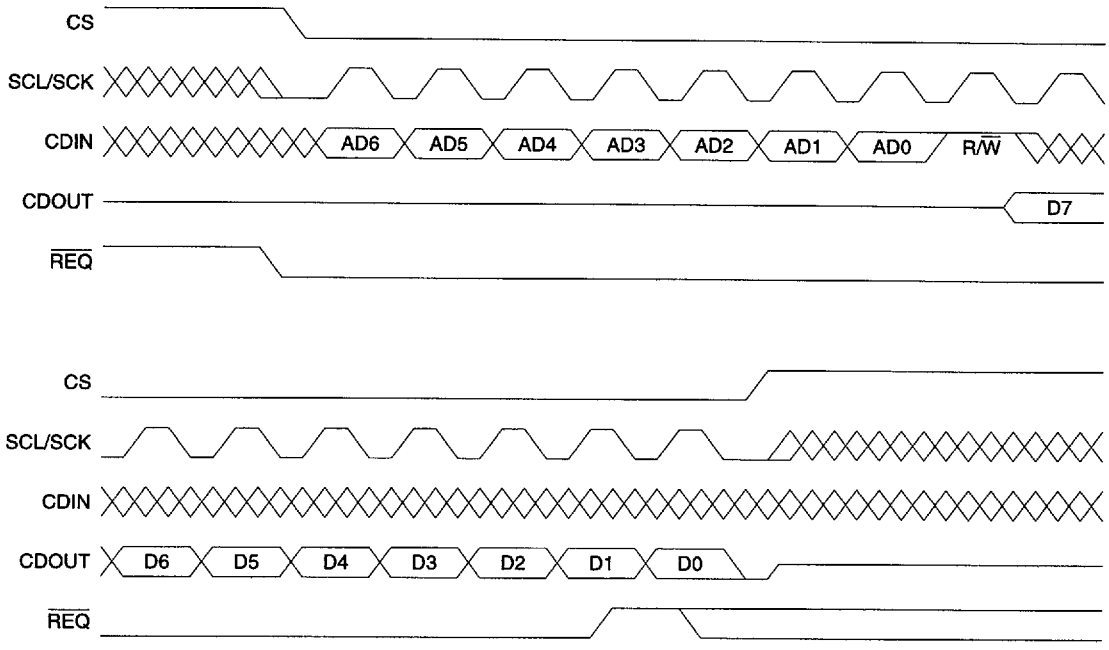


Figure 14b. Control Port Timing, SPI Read

SPI mode

The operating mode of the port is determined by the state of the M0 bit in the serial control port control register (SCPCN) as seen in Figure 16. If M0 is high, the port is SPI (Serial Peripheral Interface) compatible. M0 is loaded during a hardware or software reset, as well as controllable by microcode. The status of $\overline{CS}$ sets the mode of the SCP during a hardware and software reset. If $\overline{CS}$ is low during a reset the mode is SPI. It is important to note that M0 is loaded on ANY reset (hardware or software) and CS should be low when either reset is issued to ensure the mode remains SPI.

For normal SPI operation SCL/SCK, $\overline{CS}$, CDIN, CDOUT and $\overline{REQ}$ are used. SCL/SCK is the serial clock input which is always driven by an external device. $\overline{CS}$ is the active low enable signal. CDIN is the control data input. CDOUT is the control data output. $\overline{REQ}$ is the active low request signal, which is driven low when there is valid data in the serial control port output SCPOUT register (shown in Figure 16).

As an SPI compatible port, data is communicated on the CDIN and CDOUT pins and is clocked by the rising edge of SCL/SCK. $\overline{CS}$ is used to select the device on which the CDIN and CDOUT signals will be valid.

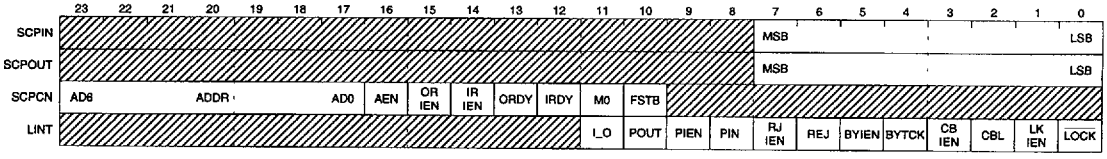


Figure 16. Serial Control Port Register Bit Map

Figure 14a shows the relative timing necessary for an SPI write operation for a single byte. A 'write' is defined as the tranfer of data from an SPI bus master to the CS4920A serial control port via CDIN. A transfer is initiated with $\overline{CS}$ being driven active low. This is followed by a 7 bit address and a read/write bit (set low for a write). For SPI mode, this address is typically not used, however it is still necessary to clock an address across the bus followed by the read/write bit.

If a write to the CS4920A is specified, 8 bits of data on CDIN will be shifted into the input shift register as shown in Figure 12. When the shift register is full, the 8 bit data is transferred to the Serial Control Port Input (SCPIN) register on the falling edge of the 8th data bit and the input ready flag (IRDY) flag is set. This flag generates an interrupt on interrupt line 2 if the input ready interrupt enable (IRIEN) bit is set high. The interrupt vector is 0006H.

The master can continue to send data, but it will be rejected if the IRDY has not yet been cleared. This flag is cleared by reading the SCPIN register. If a byte is rejected, the reject (REJ) flag in the Long Interrupt (LINT) register is set. A rising edge of the REJ flag generates an interrupt on interrupt line 3 if the reject interrupt enable (RJIEN) bit is set high. The REJ flag is cleared by reading SCPCN. There is no external hardware mechanism to detect that a byte transfered has failed. However, the microcode executing on the CS4920A can provide for a solution which would notify the host if a failure had occured using the REJ bit internally.

If the DSP core of the CS4920A wants to send a byte to the master, it first writes the byte to the Serial Control Port Output (SCPOUT) register. Note the DSP only sends 8 bits per transfer to the SCPOUT register. A write to the SCPOUT sets the request pin ($\overline{REQ}$) active low and the output ready (ORDY) bit low. The master must recognize the request and issue a read operation to the DSP. Figure 14b shows the relative timing of a single byte read. The master must send the 7 bit address (if address checking is enabled it must match the address in the SCPCN register) and the read bit. After the the falling edge of SCL/SCK for the read/write bit, the serial shift register is loaded with the byte to be sent and the most significant bit is placed on the CDOUT line. In addition, the ORDY is set high. A rising edge of the ORDY bit will generate an interrupt on interrupt line 2 if the output ready interrupt enable (ORIEN) bit is set high. The interrupt vector is 0007H.

The 8 bit value in the serial shift register is shifted out by the master. The data is valid on the rising edge of SCL/SCK and transitions immediately following the falling edge. For SPI, the $\overline{REQ}$ line will be de-asserted immediately following the rising edge of the second to last data bit, of the current byte being transfered, if there is no data in the SCPOUT register. The $\overline{REQ}$ line is guaranteed to stay de-asserted (high) until the rising edge of the SCL/SCK for the last data bit. This signals the host that the transfer is complete.

If there is data placed in SCPOUT prior to the rising edge of SCL/SCK for the second to last data bit, then $\overline{REQ}$ will remain asserted (low). Immediately following the falling edge of SCL/SCK for the last data bit, the new data byte will be loaded into the serial shift register. The host should continue to read this new byte. It is important to note that once the data is in the shift register, clocks on the SCL/SCK line will shift the data bits out of the shift register. The host should read the byte prior to any other bus activity or the data will be lost. If $\overline{CS}$ is de-asserted SCK/SCL will not shift the data out. However the data is still in the shift register. Once $\overline{CS}$ becomes active (low) each SCL/SCK will shift the data out of the register.

If data is placed in SCPOUT after the rising edge of SCL/SCK for the second to last data bit,

but before the rising edge of SCL/SCK for the last data bit, $\overline{REQ}$ will not be asserted until after the rising edge of SCL/SCK for the last data bit. This should be treated as a completed transfer. The data written to SCPOUT will not be loaded into the shift register on the falling edge of SCL/SCK for the last data bit. Therefore, a new read operation is required to read this byte.

The SCPCN register is a read/write register. The following describes the function of each bit.

Serial Control Port Control Register (SCPCN)

ADDR: Address. Seven bit address of the audio decoder. (AD6 - AD0)

AEN: Address enable. When high, message address is compared to ADDR.

M0: Mode control. Low = $I^2C^®$; High = SPI.

IRIEN: Input ready interrupt enable. When high, low to high transition of IRDY generates interrupt 2. Interrupt vector = 0006_H.

ORIEN: Output ready interrupt enable. When high, low to high transition of ORDY generates interrupt 2. Interrupt vector = 0007_H.

IRDY: Input ready status bit. Read only. High when SCPIN is full.

ORDY: Output ready status bit. Read only. High when SCPOUT is empty.

FSTB: Fast mode bit. This bit is set low coming out of reset. When the bit is low, the SCP is configured to operate at much higher bit rates. This mode is useful for downloading the initial program to memory. When the bit is high the SCP conforms to the timing requirements of $I^2C^®$ and SPI formats in slow mode.

Long Interrupt Register (LINT)

REJ: Reject status bit. Read only. High when input data rejected.

RJIEN: Reject interrupt enable. When high, low to high transition of REJ generates interrupt 3.

User Definable Pins

The CS4920A has two pins which can be defined by the user, XF and PIO. The XF signal (pin 20) can be used as an output pin. The PIO signal may be either an input or an output pin.

The XF signal is active when the SCREN bit of the CM0 register is set high. When active the XF bit in the CM0 register is mapped directly to the XF pin. An external pull-up (2.2kΩ typical) is required for proper operation. An example application for this signal would be to use the pin as a request for audio data.

The PIO signal direction is configured by the I_O bit in the LINT register. When I_O = 0 the PIO signal is configured as an input. The I_O bit is zero following the power up and any reset. As an input, the level on the PIO pin is mapped to the PIN bit of the LINT register. A rising edge of the PIN bit will generate a interrupt on line 3 if the PIEN bit is set to one. The LINT register must be read to clear this interrupt.

If the I_O bit is set to a logic one, the PIO pin is an output. A pull-down register (10kΩ typical) is required for proper operation. As an output, the POUT bit of the LINT register is mapped directly to the PIO pin.

CM0		MUTE	MONO OUT	HO	HFS	DIV	XF	SCR EN	FS	DAC EN	SLW	FSRS	P	Q									
LINT												I_O	POUT	PIEN	PIN	RJ IEN	REJ	BYIEN	BYTCK	CB IEN	CBL	LK IEN	LOCK

Figure 17. User Definable Pins Register Bit Map

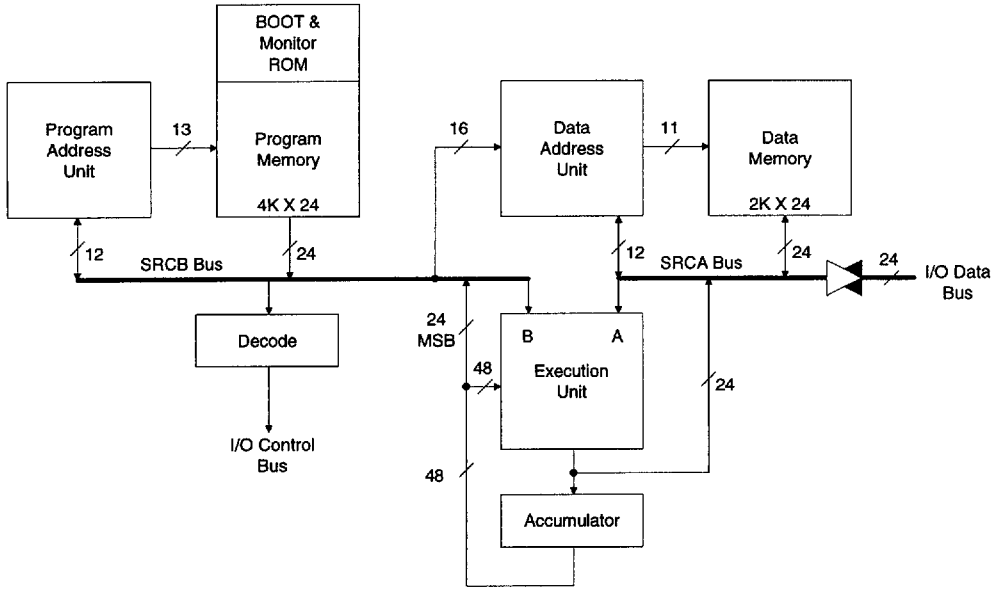


Figure 18. DSP Architecture

DSP ARCHITECTURE

Overview

The DSP core is a time multiplexed dual bus architecture. As shown in Figure 18, there are two static RAM blocks. The one labeled Data Memory typically contains buffered audio data and intermediate processing results. The block labeled Program Memory contains the entire program running at a particular time. Program Memory is also typically used to store filter coefficients.

All instructions take two clock cycles to complete. This timing is shown in Figure 19. During the first clock cycle, typically one operand is read from data memory and a second operand is read from program memory. During the second cycle, the result is stored in data memory and the next instruction is prefetched from program memory.

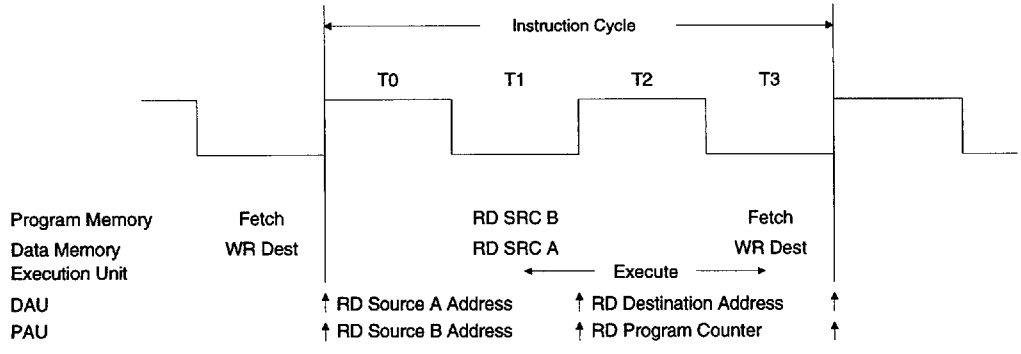


Figure 19. DSP Timing

Results can be stored in program memory but an extra instruction is required. When a load program memory instruction (LD) is executed, the contents of the accumulator can be loaded into program space. During the first cycle of the instruction, the contents are transferred. During the second half, the next instruction is prefetched.

Processing occurs in four phases. In the first phase, the instruction in the instruction register is decoded. In the second phase, the A and B operands are read. In the third phase, the ALU operation is performed. In the fourth phase, the result is stored and the next instruction is prefetched.

Since there is no pipeline, no processing delays are seen by the programmer. The contents of an address register can be used as an indirect address during the instruction immediately after an instruction that modified the address register. Conditional jumps can occur immediately after an instruction that generated the condition code.

There are 2 busses in the DSP core, the source A--destination bus (SRCA) and the source B--instruction bus (SRCB). SRCA connects to the data memory, the data address unit (DAU), the input of the accumulator (ACC), the A input of the execution unit (EU), and the I/O data bus. SRCB connects to the program memory, the program address unit (PAU), the DAU, the output of the ACC, and the B input to the EU.

During the first half of a typical arithmetic or logical instruction, the A operand to the EU is put on the SRCA bus. This operand can come from the data memory, registers in the DAU, or I/O registers. During the second half of such an instruction, the result from the EU is put on the SRCA bus. This result can be written to a data memory location, a DAU register, an I/O register, or the ACC.

During the first half of the instruction described above, the B operand to the EU is put on the SRCB bus. This operand can come from program memory, registers in the PAU, or the ACC. During the second half of the instruction, the next instruction is fetched on the SRCB bus. Some fields of the instruction are latched and decoded in the decode block and some are latched and decoded in the DAU.

The instruction fields that are latched in the DAU specify where the A operand comes from. It can come from registers in the DAU, data memory or I/O registers. When it comes from data memory, the DAU generates the operand's address. This address can be generated directly from bits in the instruction (direct addressing) or it can be generated from the contents of registers in the DAU (indirect addressing). Since both the address of the A operand and the destination must be provided for some instructions, the DAU can generate two addresses in one instruction cycle.

The Program Address Unit (PAU) contains registers which are used to generate program memory addresses. During the first half of an instruction, the generated address points to the B operand. During the second half of the instruction, the generated address points to the next instruction. Typically, the Program Address Registers (PAR's) are used to generate the B operand address, and the Program Counter (PC) is used to generate the next instruction address.

The Execution Unit (EU) shown in Figure 20, performs the operation specified in the instruction opcode on the A and B operands. It can perform the basic logical and arithmetic operations such as AND, OR, XOR, ADD, and SUBTRACT. For these instructions, it creates a 24 bit result from two 24 bit operands. It can also multiply, multiply and accumulate, multiply and subtract while accumulating, and perform a divide iteration. Since these instructions require double precision sources or destinations, a 48 bit accumulator register (ACC) is provided. A shifter on the output of ACC provides easy scaling operations.

Execution Unit

The Execution Unit (EU) is the main processing block in the DSP. As shown in Figure 20, it consists of an arithmetic logic unit (ALU), a 24 by 24 multiplier, a 48 bit adder, a 48 bit accumulator register (ACC), and a shifter. Transparent latches and tristate buffers guide data through the EU at the proper times. There are no pipeline registers and no data registers other than ACC.

The logic unit performs AND, OR, and XOR functions. The NOT function can be performed by an XOR with FFFFFF$_H$. When a logic instruction is executed, 24 bit operands are read from the SRCA and SRCB buses and the 24 bit result is driven back onto the SRCA bus. If ACC is specified as the destination, the result gets written into the high 24 bits of ACC. The low 24 bits remain unchanged.

The high 24 bits of the adder perform the ADD, add with carry (ADDC), subtract (SUB), and subtract with carry (SUBC) instructions. The 24 bit operands are read from the SRCA and SRCB buses and the result is driven on the SRCA bus. The SUB and SUBC instructions subtract the SRCA operand from the SRCB operand. If ACC is specified as the destination, the result gets written into the high 24 bits. The low 24 bits remain unchanged.

The multiplier and adder are used in the following instructions: multiply (MPY), multiply and store low result (MPYL), multiply and add accumulator (MAC), multiply and add accumulator and store low result (MACL), multiply and subtract accumulator (MSU), and multiply and subtract accumulator and store low result (MSUL). When one of these instructions is executed, the 24 bit operands from SRCA and SRCB are first multiplied. This multiplication generates a 48 bit result. When MPY and MPYL are executed, zero is added to this 48 bit result. When MAC and MACL are executed, the 48 bit contents of ACC are added to the multiplication result. When MSU and MSUL are executed, the

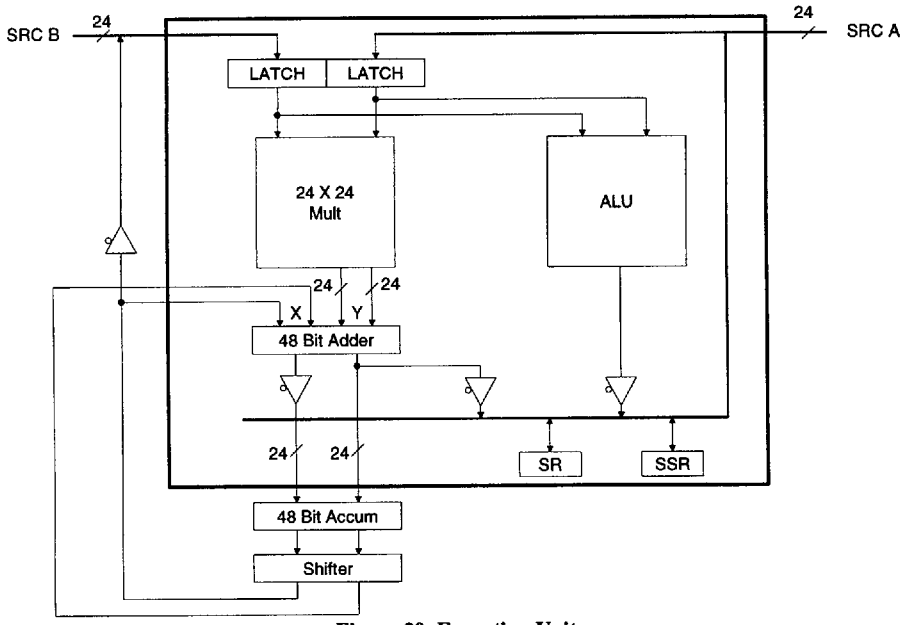


Figure 20. Execution Unit

48 bit result of the multiplication is subtracted from the ACC .

When ACC is specified as the destination, the low 24 bits of the result of a multiplication are always written to the low 24 bits of ACC. When MPY, MAC, and MSU are executed, the high 24 bits of the result are driven on the SRCA bus. If ACC is specified as the destination, these 24 bits get written into the high 24 bits of ACC. When MPYL, MACL, and MSUL are executed, the low 24 bits of the result of the addition are driven on the SRCA bus. If ACC is specified as the destination, the low 24 bits get written into both the high and low words of the accumulator.

The condition code flags are updated by the result of the 48 bit addition(subtraction) following the multiply, not by the data placed on SCRA or in the ACC.

The shifter on the output of ACC allows scaling of the accumulator (e.g. result of a filter convolution). The shift (SHF), and shift low (SHFL) instructions shift the 48 bit contents of ACC left by 1, 2, or 3 bits or right by one bit. The sign bit is extended during a shift right by one. When SHF is executed and ACC is the destination, the 48 bit result of the shift is stored in ACC. When SHFL is executed and ACC is the destination, the low 24 bits of the 48 bit result of the shift is written into both the low 24 bits and high 24 bits of ACC. When ACC is not the destination, the high 24 bits of the shift result are driven on SRCA during SHF and the low 24 bits during SHFL.

Barrel shifting left for 24 bit operands can be accomplished by multiplying by 2^N and storing the low result. Barrel shifting right can be accomplished by multiplying by $2^{(24-N)}$.

Divide instruction (DIV) divides the contents of the accumulator by the SRCA operand. It performs one iteration of a non restoring fractional division algorithm. DIV must be repeated 24 times to complete a 24 bit division. After 24 iterations, the high 24 bits of ACC contain the partial remainder and the low 24 bits contain the quotient.

The DIV instruction first XOR's the sign bits of SRCA and ACC. ACC is then shifted left by one bit with the carry bit (C) shifted into the LSB of ACC. Note on the first iteration, the C bit should be cleared. If the result of the previous XOR was one, SRCA is added to the high 24 bits of ACC and stored back in the high 24 bits. If the result was zero, SRCA is subtracted from the high 24 bits of ACC and stored back in the high 24 bits. The carry from the add or subtract properly sets the carry for the next iteration.

There are five status bits, V, C, Z, N, and U. These are located in the STATUS register which is described in the Control and Status Register section.

V is the overflow status bit. It is set when an addition, subtraction, or shift overflows. The V bit is cleared by writing a zero to it.

C is the carry flag. It is the carry out or borrow out of the 48 bit adder during an arithmetic instruction. The C bit is cleared by writing a zero to it.

Z is the zero flag. It is high if the result of an arithmetic or logical operation is zero. Z is updated by the 48 bit result of the multiply, divide, and shift instructions. Z is updated by the 24 bit results of the add, subtract, and logical instructions.

N is the negative flag. N is high if the MSB of the result is high. If the result represents a number, then N specifies whether it is positive or negative. During a MPYL, MACL, MSUL, or SHFL instruction, the low 24 bits of the result are put on the SRCA bus. During these instructions, N is determined by the output of the adder, not the data on the SRCA bus.

U is the unnormalized flag. U is high if the result of an arithmetic operation is unnormalized. The result is unnormalized if the MSB and the MSB minus one bits are the same. U is updated by the output of the 48 bit adder, not the data on the SRCA bus.

Table 3 summarizes the affected status bits by the various instructions.

	V	C	Z	N	U
AND			x	x	
ADD	x	x	x	x	x
ADDC	x	x	x	x	x
DIV		x	x	x	x
JMP					
JMPS					
LD					
MAC	x	x	x	x	x
MACL	x	x	x	x	x
MPY		0	x	x	x
MPYL		0	x	x	x
MSU	x	x	x	x	x
MSUL	x	x	x	x	x
MVD					
MVP					
NOP					
OR			x	x	
REP					
RET/RETI					
SHF	x	0	x	x	x
SHFL	x	0	x	x	x
SUB	x	x	x	x	x
SUBC	x	x	x	x	x
TRAP					
XOR			x	x	

Table 3. Instructions Effect on Status Bits

Data Address Unit

As shown in Figure 21, the DAU consists of eight address registers (AR's), eight modulo address registers (MAR's), an increment/decrement unit, and part of the instruction register. Tristate buffers and multiplexors are used for data path control.

The instruction word can independently specify both the A operand and the destination. The A operand can be the contents of an AR or I/O register, or it can be a location in data memory. When it is a location in data memory, the instruction word can specify the seven LSB's of the data memory address (direct addressing) or an AR which contains the data memory address (indirect addressing).

When *direct addressing* is selected, AR0 is used as the A operand page register and AR1 is used as the destination page register. Bits 10 through 7 of the page register are used as the MSB's of the address. These four bits and the seven LSB's from the instruction word create the required eleven bit data memory address.

When *indirect addressing* is selected, the eleven LSB's of the specified AR create the required eleven bit data memory address. The twelve bit contents of the specified AR can be post incremented or post decremented. This updated value is written back into the AR at the end of the first half of the instruction cycle.

In addition, addressing may be specified to be bit reverse post increment or bit reverse post decrement. Bit reverse addressing is very useful for addressing the results of an FFT.

The result from the execution unit can be written to an AR, an MAR, an I/O register, the ACC, or any location in data memory. If an AR is the destination, the low twelve bits of the result are written into the 12 bit AR. If an MAR is specified as the destination, the four LSB's of the result are written into the 4 bit MAR. If the result is written to data memory, the memory address can be generated exactly the same as the A operand address. The destination address can be post modified exactly the same as the A oper-

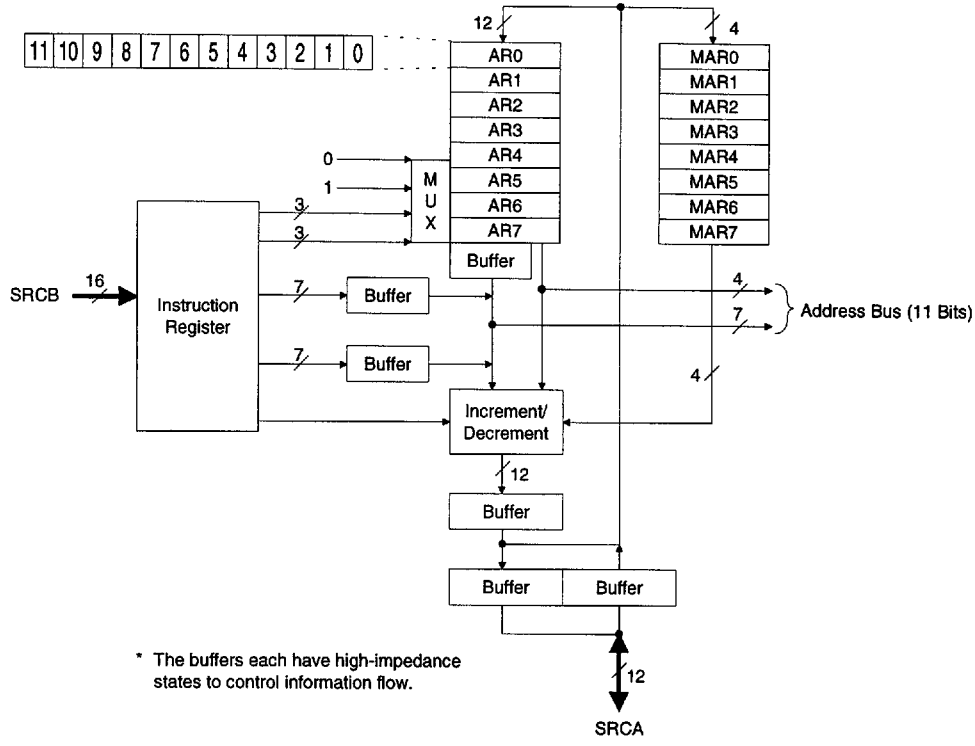


Figure 21. Data Address Unit

and address. The modified address is written back to the AR at the end of the second half of the instruction cycle.

Every address register (AR) has a corresponding modulo address register (MAR). The MAR can specify circular buffers or reverse carry address blocks. The value in the MAR specifies how many bits the carry is allowed to propagate through. When normal carry is specified, the carry propagates from the LSB to the Nth bit, where N is the value in the MAR. This means circular buffers of size 2^N that start at 2^N block boundaries can be created. When reverse carry is specified, the carry is injected at the Nth bit and propagated to the LSB. This provides reverse carry addressing to block sizes of 2^N starting at 2^N block boundaries.

All addressing options are specified in the instruction word and can be performed on the A operand address and the destination address.

Program Address Unit

The Program Address Unit (PAU) generates the 13 bit address for the program memory. It generates two addresses per instruction cycle. If the current instruction requires a source B address, the address generated during the first half is the B operand address. The address generated during the second half is the next instruction address.

Program memory consists of 4k words of RAM and 512 words of ROM. The ROM stores the boot and debug programs. The 13th address bit selects between RAM and ROM. The ROM space should not be accessed by the user.

As shown in Figure 22, the PAU consists of two 13 bit Program Address Registers (PAR's), two 4 bit Modulo Program Address Registers (MPAR's), the 13 bit Program Counter (PC), a 10 bit Loop Counter (LC), and seven stack locations each for the PC and LC. There is also a stack pointer which points to the current PC and the current LC.

The next instruction address normally comes from the PC. After reading the instruction, the PC is incremented. During a jump instruction (JMP), the jump address comes from ACC or immediate short data. This address is loaded into the PC during the first half of the jump instruction. The next instruction is read from the new address in the PC.

When a jump-to-subroutine (JMPS) instruction is executed, the PC is incremented, the stack pointer is incremented, and the jump address is written to the new PC. When a return-from-subroutine (RET) is executed, the stack pointer is decremented and the next instruction is read from the old PC. Incrementing the stack pointer pushes the PC and LC to the stack and decrementing the stack pointer pops the PC and LC from the stack.

The load instruction (LD) and the repeat (REP) can load LC from the SRCB bus during the first half of an instruction cycle. Loading this register causes the next instruction to be repeated by the value in LC+1. Every time the next instruction is executed, LC is decremented. Since the PC does not have to be incremented, LC is decremented by the increment/decrement unit during the time which the PC is normally incremented. Instructions with immediate data cannot be repeated.

Looping can be accomplished by repeating a jump to subroutine instruction. Nested loops are possible since both the PC and LC are pushed onto the stack during a jump-to-subroutine. This type of looping has two instructions of overhead, the jump to subroutine and return instructions.

During the first half of an instruction, the B operand can be read from a program address register (PAR) or from program memory. During the second half of an instruction, the next instruction is prefetched. If the B operand comes from program memory, the address can come from the PC+1 (immediate addressing) or a PAR (indirect addressing).

If indirect addressing is specified, the contents of the specified PAR can be post modified. The

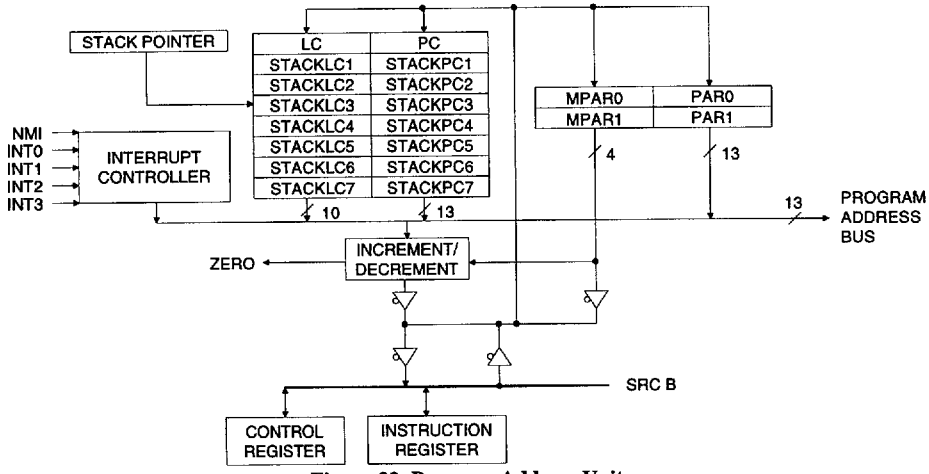


Figure 22. Program Address Unit

value can be incremented or decremented. There is no reverse carry option. Although post modify can be specified in the instruction word, whether it is incremented or decremented is determined by the DEC bit in the control register. When DEC is high, the contents of the specified PAR is decremented.

Each PAR has an associated Modulo Program Address Register (MPAR). The MPAR's create circular buffers of length 2^N that start at 2^N block boundaries, where N is the value in the MPAR. This allows carries and borrows in the post modify increment/decrement unit to propagate from the LSB to the Nth bit only.

The PC, LC, PAR's, MPAR's, the control register, the top stack location and program memory pointed to by a PAR can be loaded from immediate data (13 bits) or from the accumulator in the Execution Unit. The LD (load) instruction loads them during the first half of an instruction cycle.

The PC, LC, PAR's, MPAR's, the control register, the top stack location and program memory pointed to by a PAR can be read by a move program (MVP) instruction.

Interrupts

There are five interrupt lines from the on-chip peripherals. They are interrupts 0 through 3 and a non-maskable interrupt. Interrupts 0 through 2 are used for outputting audio data, inputting audio data, and transferring control information respectively. Interrupt 3 is used for reporting error conditions and updating channel status information in the digital audio transmitter. The non-maskable interrupt is used by the debugger.

Interrupts 0 through 3 can be enabled by setting the interrupt enable (IEN) bit in the control register. They can be individually disabled by clearing the corresponding mask bit ($\overline{\text{MSK}}$0-3) in the control register. The non-maskable interrupt is disabled by clearing the non-maskable interrupt enable (NMIEN) bit in the control register.

When an interrupt occurs, an interrupt vector generates the address of the next instruction. This interrupt vector address is unique for each interrupt. Interrupt 0 through 2 have two possible interrupt vector addresses.

Interrupt 0 occurs at twice the audio sample rate. The first time it occurs, the interrupt vector address is 0002_H and right audio data should be written to the DAC and the digital audio transmitter. The second time it occurs, the interrupt vector address is 0003_H and left audio data should be written to the DAC and transmitter. Refer to Figure 3.

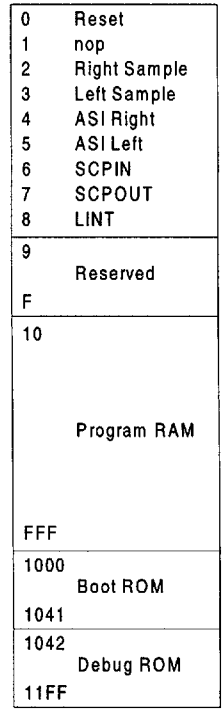


Figure 23. Program Memory Map with IRQ vectors

The audio data serial input port double buffers one input word. This word can contain up to 24 bits. The frame sync (FSYNC) input can differentiate between left and right data when stereo audio data is being input. When a new word is loaded into the double buffer, interrupt 1 occurs. The state of FSYNC determines whether the interrupt vector address is 0004_H or 0005_H. Typically, if left data has been loaded the address is 0004_H and if right data has been loaded, the address is 0005_H.

Interrupt 2 occurs when data is transferred through the serial control port. When data is transferred from an external micro controller to the DSP, the interrupt vector address is 0006_H. When data is transferred from the DSP to an external micro controller, the address is 0007_H.

Interrupt 3 is a wired OR of various conditions in the peripherals. The Long Interrupt Register (LINT) reports all conditions that generate interrupt 3. It also contains mask bits to individually enable each condition to generate an interrupt. The interrupt vector address is always 0008_H.

The debugger uses the non-maskable interrupt to suspend operation in the processor. When this interrupt occurs, control switches to the monitor program in ROM. The interrupt vector address is 1002_H.

The interrupts are priority encoded to prevent problems when multiple interrupts occur simultaneously. The non-maskable interrupt has a higher priority than the maskable interrupts. Of the maskable interrupts, line 0 has the highest priority and line 3 has the lowest priority.

An interrupt is detected by the program controller at the end of the instruction cycle during which the interrupt occurred. Since the next instruction has already been fetched, it is executed before the instruction at the interrupt vector location is executed. There is a one to two instruction cycle delay from the time the interrupt occurs until the instruction at the interrupt vector location is executed.

Interrupts can be long or short. A short interrupt occurs if the instruction at the interrupt vector location is anything but a JMPS(jump to subroutine). After this instruction is executed, program control switches back to normal. The instruction at the interrupt vector location cannot have immediate data.

A long interrupt occurs if the instruction at the interrupt vector address is a JMPS (jump-to-subroutine). When the jump occurs, the IEN (interrupt enable) bit in the control register is cleared. This disables interrupts. The IEN bit is set when an RETI (return from interrupt) instruction is executed. The IEN bit can also be set and cleared by writing to the control register.

There is a hardware stack in the PAU which is 7 locations deep for both the program counter (PC) and the loop counter (LC). This allows 7 levels of subroutines and interrupts without a software stack for these counters. There is also one shadow status register which operates as a one deep hardware stack for the status register. Multiple levels of interrupts can be supported by implementing a software stack for the status register and by using short interrupts.

When a long interrupt occurs, the contents of the status register and the shadow status register swap. If a software stack is required, the contents of the shadow status register must be stored and interrupts enabled. Near the end of the interrupt service routine, interrupts must be disabled and the shadow status register restored. A RETI (return from interrupt) instruction swaps the contents of the status register and the shadow status register and enables interrupts. The status and shadow status registers do not swap when a short interrupt occurs.

If multiple levels of interrupts are not required or if the interrupt service routine does not affect the

status register, the shadow status register does not have to be saved on the software stack. In this case, the contents of the status register and shadow status register are swapped when the interrupt occurs and again when the RETI is executed. Short interrupts do not swap the contents of the status and shadow registers.

Instruction Set

The instruction set allows flexible addressing of two source operands and the destination. In one instruction the main ALU operation is performed and up to three memory address pointers can be updated.

The assembly code syntax is:

OPCODE SRCA, SRCB, DEST

For typical arithmetic and logical instructions SRCA is a location in data memory, an address register, or an I/O register. SRCB is a location in program memory, a program address register, or the accumulator. DEST is a location in data memory, an address register, an I/O register, or the accumulator.

Addressing modes can be register direct or register indirect for SRCA, SRCB, and DEST. SRCA and DEST memory locations can also be addressed directly. SRCB can also be immediate data.

The following examples of an ADD instruction illustrate possible addressing modes.

```
add *AR2+, *PAR0, *AR3+
/*SRCA=AR2 indirect with post increment;
SRCB=PAR0 indirect; DEST=AR3 indirect with
post increment*/

add *AR2, *PAR0m, *AR3
/*SRCA=AR2 indirect ; SRCB=PAR0 indirect
with post modify; DEST=AR3 indirect*/

add *AR2-, PAR0r, *AR3-
/*SRCA=AR2 indirect with post decrement;
SRCB=PAR0 register direct; DEST=AR3 indi-
rect with post decrement*/

add *AR2b+, 0x123456, *AR3b+
/*SRCA=AR2 indirect with bit reverse post in-
crement; SRCB=immediate; DEST=AR3 indirect
with bit reverse post increment*/

add *AR2b-, ACC, *AR3b-
/*SRCA=AR2 indirect with bit reverse post dec-
rement; SRCB=accumulator; DEST=AR3
indirect with bit reverse post decrement*/

add *AR2b-, ACC , *AR3b+
/*SRCA=AR2 indirect with bit reverse post dec-
rement; SRCB=ACC; DEST=AR3 indirect with
bit reverse post increment*/

add AR2, ACC, AR3
/*SRCA=AR2 register direct; SRCB=ACC;
DEST=AR3 register direct*/

add MAR2, ACC, MAR3
/*SRCA=MAR2 register direct; SRCB=ACC;
DEST=MAR3 register direct*/

add 0x19, ACC, 0x27
/*SRCA=direct address, AR0 is the page regis-
ter; SRCB=ACC; DEST=direct address, AR1 is
the page register*/

add 0x19, ACC, ACC
/*SRCA=direct address, AR0 is the page regis-
ter; SRCB=ACC; DEST=ACC*/
```

Any combination of addressing modes for SRCA, SRCB, and DEST are permitted.

Figure 24 illustrates the processor programming model. It shows all registers and memory, and the bus attached to each.

All registers and memory locations on bus A can be used as the SRCA operand or as the destina-

tion. The MPAR's, the PAR's, the accumulator, and program memory space can be used as the SRCB operand.

The LD (load) instruction can write the contents of the accumulator or immediate short (13 bits) data to a PAR, an MPAR, the control register(CR), the program counter (PC), the loop counter (LC), and the last PC and REP pushed onto the stack (PC-1 and LC-1). It can also write the contents of the accumulator or immediate short data to program memory pointed to by a PAR. The specified PAR can be post modified or not post modified.

The MVP (move program) instruction can move immediate long data, the accumulator, a PAR, an MPAR, the CR, the PC, the LC, the PC-1, and the LC-1 to any destination described in the ADD example above. It can also move program memory pointed to by an PAR to any destination described above and any of the stack pointer locations (STACKPC[0-7] and STACKLC[0-7]). The specified PAR can be post modified or not post modified.

The contents of the stack pointer can be accessed by reading bits 5 through 7 of the Status Register. Bits 5 through 7 of the Shadow Status Register are always low.

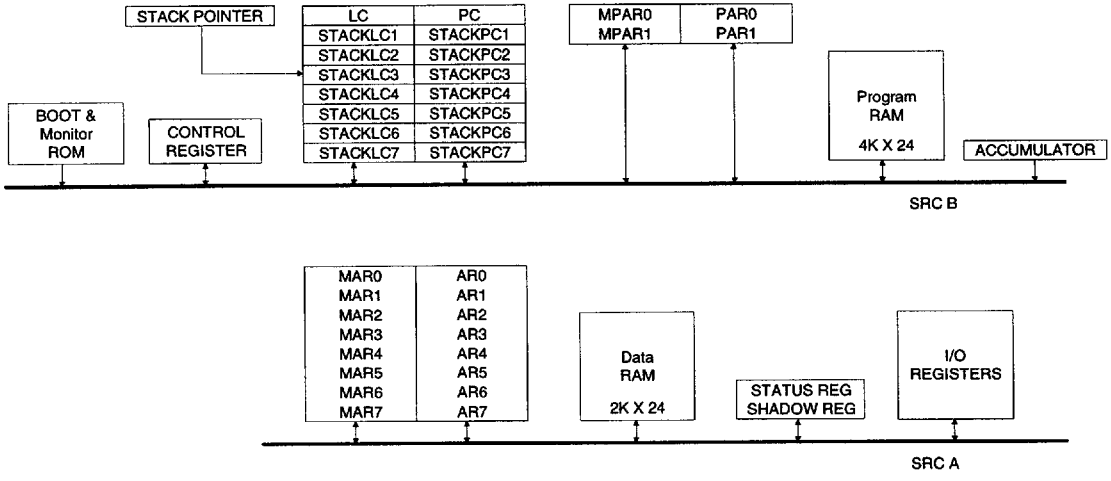


Figure 24. Programming Model

Control and Status Registers

The Status Register is connected to the SRCA bus. Since it is I/O mapped, it can be used as the SRCA operand or destination for most ALU operations. The control register is connected to the SRCB bus. It is loaded by the LD (load) instruction and can be read by the MVP instruction. The contents of each register are described below.

Status Register

Bit	Name	Function
23-8	(undefined)	
7-5	STPTR	Stack pointer Points to the current program and repeat counters
4	N	Negative
3	Z	Zero
2	V	Overflow
1	U	Unnormalized

Control Register

Bit	Name	Function
23-10	(undefined)	(must be programmed low)
9	NMIEN	Non-maskable interrupt enable. When low, non-maskable interrupts cannot occur. NMIEN is high after a reset.
8	PWDN	Power down. Writing a one puts the chip into a power down mode. The reset pin must be toggled to clear it. In power down mode, the processor stops functioning.
7	IEN	Interrupt enable. When high, interrupt lines 0 through 3 can occur.
6	TRACE	Trace mode enable. When high, the processor will enter single step mode.
5	$\overline{RS}$	Software reset. Writing a one resets the chip. All registers are cleared.
4	DEC	Increment/decrement. When set, the program address registers are decremented when post modify is specified. When clear, they are incremented.
3	$\overline{MSK3}$	Interrupt mask bit 3. When low, an interrupt in the LINT cannot occur.
2	$\overline{MSK2}$	Interrupt mask bit 2. When low, SCP interrupts cannot occur.
1	$\overline{MSK1}$	Interrupt mask bit 1. When low, ASI interrupts cannot occur.
0	$\overline{MSK0}$	Interrupt mask bit 0. When low, DAC interrupts cannot occur.

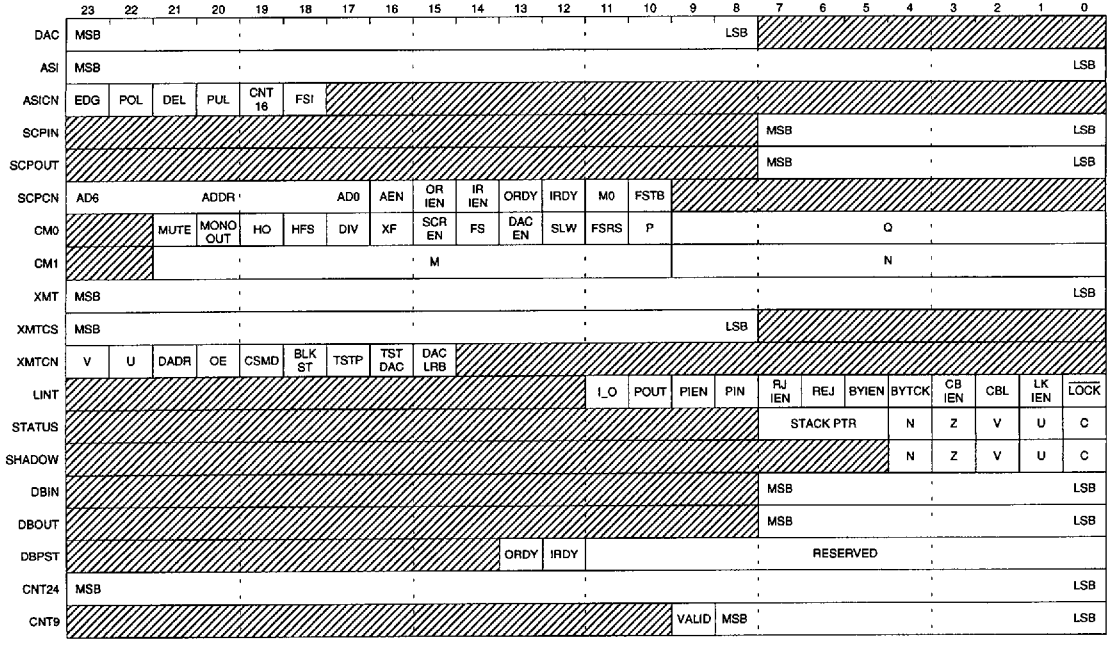


Figure 25. I/O Register Bit Map

Boot Procedure

Program and data RAM must be loaded from external memory after power up or when a new program needs to be loaded. During the loading procedure (boot), data is transferred through the serial control port to program and data memory. This procedure is controlled by a program stored internally in ROM.

Following a power up or reset, the fast mode bit (FSTB) is cleared. This places the CS4920A into a 'fast mode'. While the serial control port (SCP) still conforms to the data format determined by $\overline{CS}$ on power up, the port can be operated at much higher bit rates to facilitate faster downloading of the DSP code. Once the code has been loaded the software can set the FSTB for normal communication in either SPI or $I^2C^{®}$. Since the CS4920A is always a slave this fast mode will not affect the operation of other devices sharing the same communication bus.

The boot procedure is initiated by a low to high transition of the reset ($\overline{RESET}$) pin with the BOOT pin high. This initializes the program counter to location 1000_H, the first location in ROM. After the ROM program transfers data from the control port to memory, it issues a software reset. This is done by writing a one to the RS (reset) bit in the control register. The software reset clears all registers including the program counter, which transfers control to the new program in RAM.

A hardware reset ($\overline{RESET}$ pin toggled low) has the same affect as a software reset. During the boot procedure, all interrupts, except the debug interrupt, are disabled.

The serial control port will boot from a micro controller. When booting, it can communicate in an $I^2C^{®}$ or SPI format. If the $\overline{CS}$ (chip select) pin is high when boot is initiated, the port will communicate in $I^2C^{®}$ format. If the $\overline{CS}$ pin is low when boot is initiated, the port will commu-

Keyword	Register Addresses	Description	Keyword	Register Addresses	Description
DAC	00000	DAC output register	LINT	01011	Long interrupt register
ASI	00010	Serial input register	STATUS	10000	Status register
ASICN	00011	Serial input control register	SHADOW	1001	Shadow status register
SCPIN	00100	Serial control input register, read only	DBIN	01100	Debug input register, read only
SCPOUT	00100	Serial control output register, write only	DBOUT	01100	Debug output register, write only
SCPCN	00101	Serial control port control register	DBSPT	01101	Debug port status register
CM0	00110	Clock manager control register 0	CNT24	01110	Upper 24 bits of 33-bit counter
CM1	00111	Clock manager control	CNT9	01111	Lower 9 bits of 33-bit counter
XMT	01000	Audio tranmit register			
XMTCS	01001	Transmit channel status register			
XMTCN	01010	Transmit control register			

Table 4. Register Summary

nicate in SPI. Please refer to the timing requirements found at the beginning of the data sheet.

Nodes in an $I^2C^®$ network have unique network addresses. A message in an $I^2C^®$ network consists of the address of the node receiving the message followed by the message data. When the control port is configured for $I^2C^®$ format, it normally compares the address to an address stored in an internal register. During the boot procedure, the control port is programmed to ignore the address. The SCP section on $I^2C^®$ operation explains the mechanics of writing to the CS4920A.

The boot program in internal ROM expects data transferred through the control port to have the proper file format. The first two bytes contain the starting address for the following block of data. The starting address is 13 bits with the 13th bit specifying program or data memory. Therefore, the upper 3 bits of these two bytes are discarded internally. The second two bytes contain the length of the block of data. Successive bytes are concatenated into 24 bit words. These words are sequentially loaded into program and data memory beginning at the starting address.

Any number of blocks of data can be loaded. Two bytes containing FF and 3 bytes containing a check sum must follow the last block of data. This check sum is generated by summing all the previous data, address, and length bytes and truncating to 24 bits. This check sum is compared to the value calculated internally. If they do not match, the $\overline{REQ}$ (request) pin is pulled low and the processor does not issue the software reset. It stays in a loop until boot is initiated again.

I/O Address Space

I/O address space consists of peripheral input and output registers, peripheral control registers, and the Status and Shadow Status Registers. These registers can be used as the SRCA operand or as the destination. The assembly language syntax has a keyword for each register in I/O space. Table 4 summarizes these keywords and their respective register addresses.

Figure 25 shows all I/O register bit assignments. These registers are described in the section which follows. Note that the Control Register described earlier is not included in Figure 25. This is because the Control Register is not an I/O mapped Register.

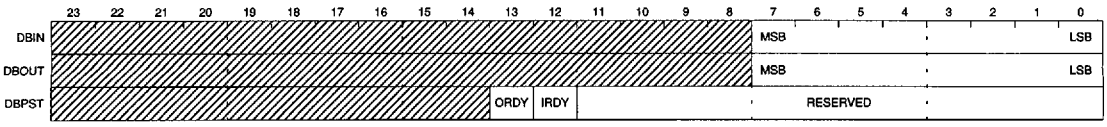


Figure 26. Debug Interrupt Register Bit Map

Debugger

The debugger consists of software and a cable which connects PC compatible parallel port to the debug port pins (DBCLK, DBDA), and an on-chip serial debug port and debug ROM. The computer can load programs, set breakpoints, read and write registers and memory, and single step programs.

The debug ROM contains the interrupt service routine which interprets commands sent from the computer. Examples of these commands are "read register", "write register", or "set break point". The main program or the boot program is interrupted when the computer sends a command to the debug port. This causes program control to switch to the debug program.

This program polls the debug port for additional commands and performs the appropriate action. When a "run" command is issued, the program executes a RETI (return from interrupt) and program control switches back to the main program.

Breakpoints are set by replacing the instruction at the breakpoint location with a TRAP instruction. The TRAP instruction generates a debug interrupt when it is executed. Once the breakpoint has been set and the processors registers have been properly initialized, the user can issue a "run" command. The main program runs until the TRAP instruction is executed. Program control then switches back to the debug program.

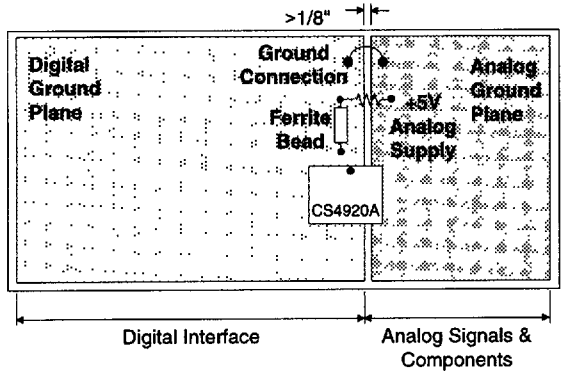
The debug port and the TRAP instruction can generate a debug interrupt. This interrupt is a unique signal which can interrupt the processor independent of the state of the IEN control bit. This interrupt can be enabled or disabled by setting or clearing the NMIEN (non-maskable interrupt enable) control register bit. The default state is enabled. NMIEN is cleared when a debug interrupt occurs and it is set when an RTI instruction is executed. Writing to the control register can also change the state of NMIEN.

The high 32 words of data memory are used by the debug program.

POWER SUPPLY AND GROUNDING

When using separate supplies, the digital power should be connected to the CS4920A via a ferrite bead, positioned closer than 1" to the device (see Figure 27). The CS4920A VA+ pin should be derived from the cleanest power source available. If only one supply is available, use the suggested arrangement in Figure 1.

The CS4920A should be positioned such that the analog pins (pins 29 - 39) are over the analog ground plane, while the rest of the pins lay over the digital ground plane as illustrated in Figures 27 and 28. The analog and digital grounds on the CS4920A are not connected internally; this should be accomplished externally through a point-to-point connection across the ground split



Note that the CS4920A is oriented with its digital pins towards the digital end of the board.

Figure 27. CS4920A Suggested Layout

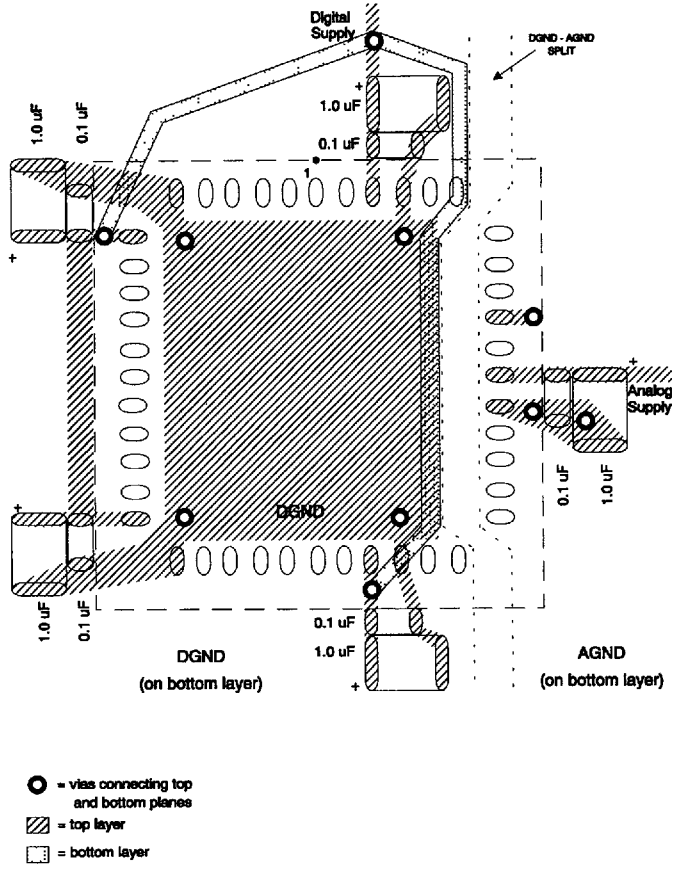


Figure 28. CS4920A Surface Mount Decoupling Layout

as shown in Figure 27. A separate power plane for the chip is preferable.

Figure 28 illustrates the optimum ground and decoupling layout for the CS4920A assuming a surface-mount socket and surface mount capacitors. Surface-mount sockets are useful since the pad locations are exactly the same as the actual chip; therefore, given that space for the socket is left on the board, the socket can be optional for production. Figure 28 depicts mostly the top layer containing signal traces and assumes the bottom or inter-layer contains a solid ground plane (analog or digital), except where the digital supply needs to run to the power pins. The important points with regards to this diagram are that the ground plane is SOLID under the CS4920A and connects all ground pins with thick traces providing the absolute lowest impedance between ground pins. The decoupling capacitors are placed as close as possible to the device which, in this case, is the socket boundary. The lowest value capacitor is placed closest to the chip. Vias are placed near the AGND and DGND pins, under the IC, and should be attached to the solid ground plane (analog or digital) on another layer. The negative side of the decoupling capacitors should also attach to the same solid ground plane. Traces bringing the power to the CS4920A should be wide thereby keeping the impedance low.

If using through-hole sockets, effort should be made to find a socket with the minimum height which will minimize the socket impedance. When using a through-hole socket, the vias under the chip in Figure 28 are not needed since the pins serve the same function.

DAC Filter Response Plots

Figures 29 through 32 show the overall frequency response, passband ripple and transition band for the CS4920A DACs. Figure 32 shows the DACs' deviation from linear phase. Fs is the selected sample frequency. Since the sample frequency is programmable, the filters will adjust to the selected sample frequency. Fs is also the FSYNC frequency.

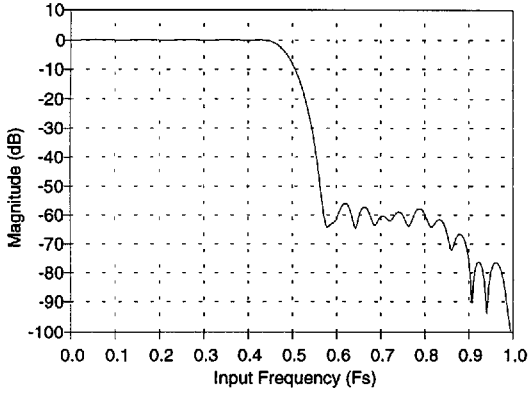


Figure 29. DAC Frequency Response.

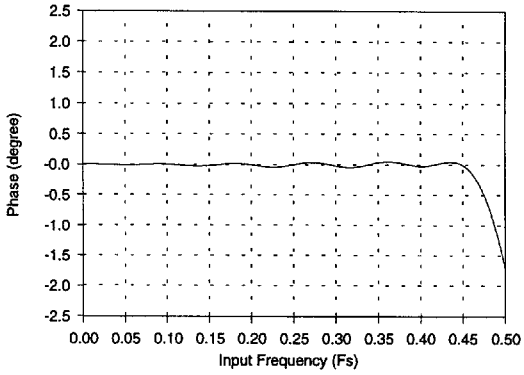


Figure 30. DAC Phase Response.

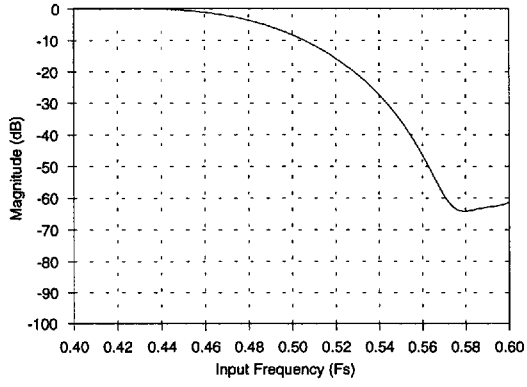


Figure 31. DAC Transition Band.

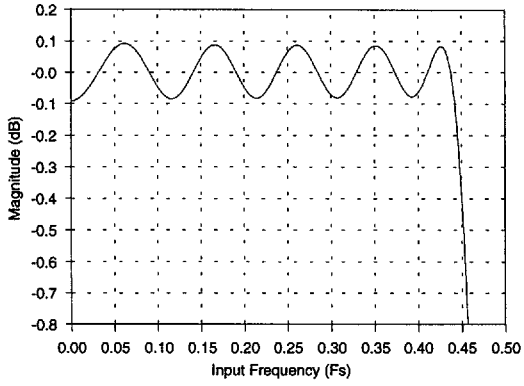
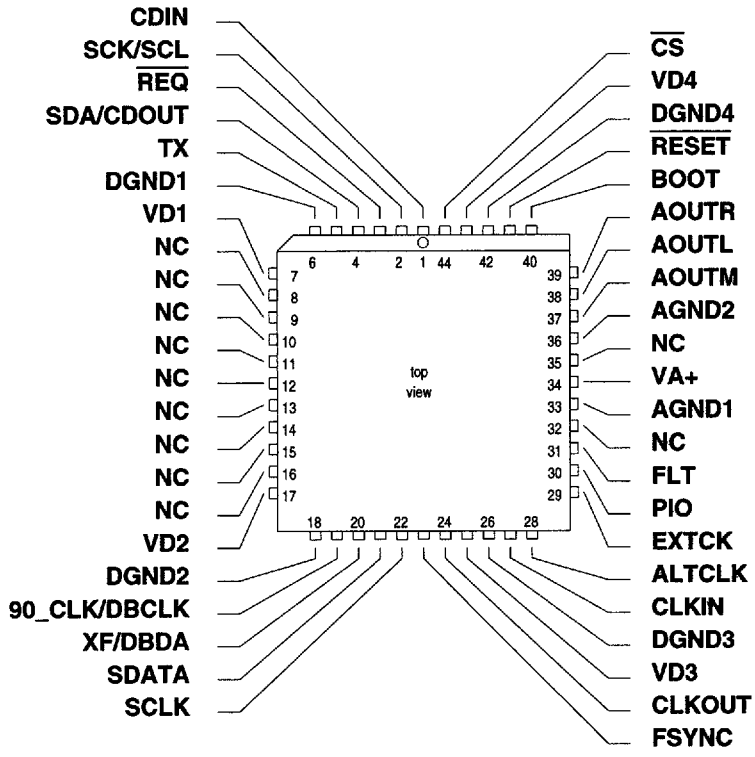


Figure 32. DAC Passband Ripple.

PIN DESCRIPTIONS



Power Supplies

VD1, VD2, VD3, VD4 - Positive Digital Power Supply, PINS 7, 17, 25, 43.

The +5V supply is connected to these pins to power the various digital subcircuits on the chip. See decoupling section in this data sheet for decoupling recommendations.

DGND1, DGND2, DGND3, DGND4 - Digital Ground, PINS 6, 18, 26, 42.

Digital power supply ground.

VA+ - Positive Analog Power Supply, PIN 34.

The analog +5V supply for the analog-to-digital converter and the PLL. Analog performance is highly dependent on the quality of this supply. See decoupling section in this data sheet for decoupling recommendations.

AGND1, AGND2 - Analog Ground, PIN 33, 36.

Analog power supply ground.

Digital-to-Analog Converter

AOUTL, AOUTR - Analog Outputs, Left and Right Channels, PINS 38, 39.

These DAC outputs are centered approximately 2.1V. An external filter is required to diminish out-of-band noise. See Typical Connection Diagram, Figure 1.

AOUTM - Mono Analog Output, PIN 37.

Mono is the summation of AOUTL and AOUTR. Mono output is 180° out-of-phase with AOUTL and AOUTR. Mono is centered approximately 2.1V. An external filter is required to diminish out-of-band noise. See Typical Connection Diagram, Figure 1.

Serial Audio Port

FSYNC - Frame Synchronization Clock Input, PIN 23,

FSYNC transitions delineate left and right audio data, or the start of a data frame, as determined by the PUL bit in the ASICN. The edge definition (left, right, or active) is determined by the POL bit.

SCLK - Serial Clock Input, PIN 22.

SCLK is used to clock the serial audio data on SDATA into the device. The EDG bit in the ASICN determines the active edge. The DEL bit can be used to delay data by one SCLK after an FSYNC transition.

SDATA - Serial Audio Data Input, PIN 21.

Audio data input to SDATA is clocked into the device by SCLK. There are several options for the relationships between SDATA, SCLK, and FSYNC.

Digital Audio Transmitter

TX - Transmit, PIN 5,

Biphase mark encoded data is output at logic levels from the TX pin. This output typically connects to the input of an RS-422 or optical transmitter. With additional external circuitry, the port can support either AES/EBU or S/PDIF formats.

Clock Manager

CLKOUT - Clock Output, PIN 24.

CLKOUT can be used to synchronize peripheral devices such as a micro controller or an audio source. The clock frequency is determined by a divide by Q and a divide by 2 of the DSP clock. Q is a 10 bit register. The maximum CLKOUT frequency is the maximum DSP frequency divided by 4.

ALTCLK - Clock Input, PIN 28.

When EXTCK is high, ALTCLK is an input for an externally generated 128Fs or 192Fs clock. Note the clock frequncy should be 256Fs or 384Fs to account for the divide by two in the clock manager.

EXTCK - External Clock Select, PIN 29.

Setting EXTCK high allows ALTCLK to be used as an input for an external VCO. Setting EXTCK low disables ALTCLK. Note EXTCK should be tied directly to either digital power or ground for proper operation.

FLT - PLL Filter, PIN 31,

A capacitor (typically 0.47 µF) connected to this pin filters the control voltage for the on-chip VCO. Trace length should be minimized to the pin.

CLKIN - Clock Input, PIN 27.

A clock input to the CLKIN is used to synchronize the PLL's. The permissible frequency range is from 32 kHz to 30 MHz. It is typical to have SCLK for the audio data and CLKIN to be derived from the same clock source to avoid asynchronous noise between the audio source and the DSP.

90_CLK - Optional SCR/PCR 33-Bit Counter Clock, PIN 19

The 90_CLK pin is an input clock signal (typically 90 kHz) which is used to clock the internal 33-bit counter. The 33-bit counter is enabled by SCREN = 1 in the CM0 register. The 33-bit counter's clock source is set to 90_CLK when DIV = 0 in the CM0 register. Otherwise when DIV = 1, the 33-bit counter will be clocked by CLKIN ÷ 300.

Control

DBCLK, DBDA - Debug Clock, Debug Data I/O, PINS 19, 20.

These pins are used for serial port debug. DBCLK clocks the data into or out of the debug port. DBDA is a bi-directional data I/O. Software and a cable are available to interface the debug port to a PC. It is required that a pull-up be used (typically 2.2 kΩ) on pin 20.

$\overline{\text{RESET}}$ - PIN 41.

The CS4920A enters a reset state while $\overline{\text{RESET}}$ is low. When in reset condition, all internal registers are set to 0, the digital audio transmitter, serial control port, and ALTCLK pin are disabled, and the stereo DAC is muted. The DAC is recalibrated and normal operation is resumed, one internal clock cycle after the rising edge of $\overline{\text{RESET}}$.

BOOT - PIN 40.

Boot enable pin. Pin must be set high to initiate the download of a program. While boot is high, $\overline{\text{RESET}}$ must be toggled low then high. This starts the internal boot program.

XF - External Flag, PIN 20.

The XF pin is a software controllable output pin via the CM0 register. The pin is enabled by SCREN = 1, and when enabled, the pin will be high if XF = 1 or low if XF = 0 in the CM0 register. It is required that a pull-up be used (typically 2.2 kΩ) on the XF pin.

PIO - Programmable Input/Output, PIN 30.

The PIO pin is a software controllable input/output pin via the LINT register. The pin is configured as an output when I_O = 1, and the output on the pin is high when POUT = 1 or low when POUT = 0 in the LINT register. The pin is configured as an input when I_O = 0, and a rising edge on the input of PIO pin will cause an interrupt 3 if PIEN = 1 in the LINT register. The input level to the PIO pin can be read by the DSP via the PIN bit of LINT register. When configured as an input, a high level on the PIO pin will set PIN = 1 and a low level will set PIN = 0. It is required that a pull-down be used (typically 10 kΩ) on the PIO pin.

Serial Control Port

$\overline{\text{REQ}}$ - Request Output, PIN 3.

This pin is driven low when the DSP needs servicing from an external device. A write to the SCPOUT will cause the $\overline{\text{REQ}}$ to go low. A pull-up resistor is required for proper operation (2.2kΩ is typical).

$\overline{\text{CS}}$ - Chip Select Input, PIN 44.

In SPI format, all communication between the host and the CS4920A is initiated when the host drives the $\overline{\text{CS}}$ pin low. This pin also serves as the communication format select during a reset or power up. When $\overline{\text{CS}}$ is high during a reset or power up the SCP will be configured in $I^2C^®$ mode. When low, it is configured in SPI mode. The mode is selectable in software by setting the M0 bit in the SCPCN.

SCK/SCL - Serial Clock Input, PIN 2.

SCK/SCL clocks data into or out of the serial control port. This is always driven by an external device because the CS4920A is always the slave.

CDOUT/SDA - Control Data Output / Serial Data I/O, PIN 4.

In SPI mode, CDOUT is a data output for the serial control data. In $I^2C®$ interface mode, SDA is a bi-directional data I/O. It is required that a pull-up be used (2.2 kΩ is typical).

CDIN - Control Data Input, PIN 1.

In SPI mode, CDIN is the data input for the serial control port. It has no function in $I^2C^®$ mode. The pin should be connected to either digital power or ground when the CS4920A is used in $I^2C^®$ systems.

PARAMETER DEFINITIONS

Resolution

The number of bits in the input words to the DACs.

Differential Nonlinearity

The worst case deviation from the ideal codewidth; expressed in LSBs.

Total Harmonic Distortion (THD)

THD is the ratio of the test signal amplitude to the rms sum of all the in-band harmonics of the test signal.

Instantaneous Dynamic Range

The Signal-to-(Noise + Distortion) ratio (S/(N+D)) with a 1 kHz, -60dB from full scale DAC input signal, with 60dB added to compensate for the small signal. Use of a small signal reduces the harmonic distortion components of the noise to insignificant levels. Units are in dB.

Interchannel Isolation

The amount of 1kHz signal present on the output of the grounded input channel with 1 kHz, 0dB signal present on the other channel. Units are in dB.

Interchannel Gain Mismatch

The difference in output voltages for each channel with a full scale digital input. Units are in dB.

Frequency Response

Worst case variation in output signal level versus frequency over 10 Hz to 20 kHz. Units in dB.

Out of Band Energy

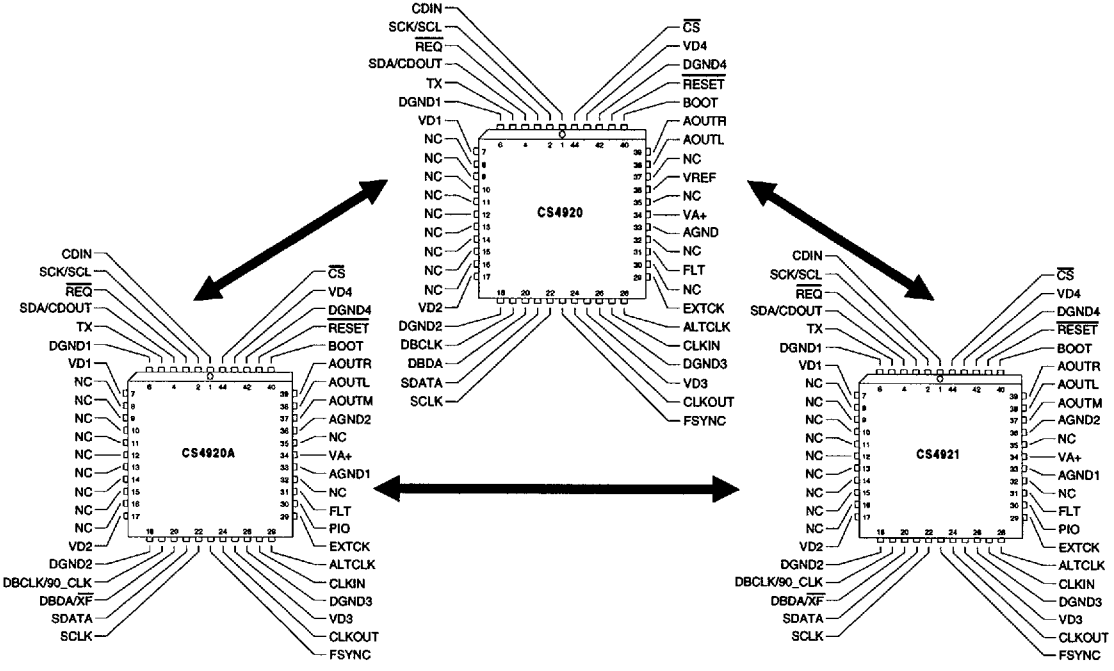
The ratio of the rms sum of the energy from 0.46×Fs to 2.1×Fs compared to the rms full-scale signal value. Tested with 48 kHz Fs giving a out-of-band energy range of 22 kHz to 100 kHz.

Application Note

CS4920 to CS4920A to CS4921 Conversion Requirements

by
Steven P. Proffitt and Craig T. Phillips

The CS4920, CS4920A, and CS4921 are all complete audio subsystems on a chip. The three devices all contain a general purpose DSP, stereo 16-bit Digital-to-Analog Converters, a programmable analog PLL clock multiplier, a S/PDIF compatible digital audio transmitter, an audio serial input port, and a serial control port. The CS4920A and the CS4921 have additional features: separate mono analog output pin, a programmed Input/Output (PIO) pin, an external flag (XF) pin, and a 33-bit counter. The CS4920 and the CS4920A are RAM-only (with a small internal ROM for boot-up) based DSP audio decoders, while the CS4921 has the audio MPEG decoder microcode and other features in ROM. The CS4921 also has a small amount of RAM available for updates. Table 1 shows the pin-to-pin relationship between the three devices and Figures 1 through 3 show the typical connection diagrams for the CS4920, CS4920A, and the CS4921.



Crystal Semiconductor Corporation
P.O. Box 17847, Austin, TX 78760
(512) 445-7222 FAX 445-7581

Pin #	CS4920	CS4920A	CS4921	Notes
1	CDIN	CDIN	CDIN	Control Data Input, for SPI mode only
2	SCK/SCL	SCK/SCL	SCK/SCL	Serial Clock Input for Control Data
3	/REQ	/REQ	/REQ	Request Output
4	SDA/CDOUT	SDA/CDOUT	SDA/CDOUT	Serial Control Data I/O (I^2C) Data Output (SPI)
5	TX	TX	TX	Digital Transmit
6	DGND1	DGND1	DGND1	Digital Ground
7	VD1	VD1	VD1	Positive Digital Power Supply, +5V
8	NC	NC	NC	No Connect
9	NC	NC	NC	No Connect
10	NC	NC	NC	No Connect
11	NC	NC	NC	No Connect
12	NC	NC	NC	No Connect
13	NC	NC	NC	No Connect
14	NC	NC	NC	No Connect
15	NC	NC	NC	No Connect
16	NC	NC	NC	No Connect
17	VD2	VD2	VD2	Positive Digital Power Supply, +5V
18	DGND2	DGND2	DGND2	Digital Ground
19†	DBCLK	DBCLK/90_CLK	DBCLK/90_CLK	Debug Clock/Optional Input Clock for PCR Counter(1)
20†	DBDA	DBDA/XF	DBDA/XF	Debug Data/External Flag(1)
21	SDATA	SDATA	SDATA	Serial Audio Data Input
22	SCLK	SCLK	SCLK	Serial Clock Input for Audio Data
23	FSYNC	FSYNC	FSYNC	Frame Synchronization Clock Input
24	CLKOUT	CLKOUT	CLKOUT	Clock Output
25	VD3	VD3	VD3	Positive Digital Power Supply, +5V
26	DGND3	DGND3	DGND3	Digital Ground
27	CLKIN	CLKIN	CLKIN	Clock Input
28	ALTCLK	ALTCLK	ALTCLK	Clock Input
29	EXTCK	EXTCK	EXTCK	External Clock Select
30†	NC	PIO	PIO	Programmed Input/Output pin(1) & (3)
31†	FLT	FLT	FLT	PLL Filter(2)
32	NC	NC	NC	No Connect
33	AGND	AGND1	AGND1	Analog Ground
34	VA+	VA+	VA+	Positive Analog Power Supply, +5V
35	NC	NC	NC	No Connect
36†	VREF	AGND2	AGND2	DAC Voltage Reference(4)/Analog Ground
37†	NC	AOUTM	AOUTM	Mono Analog Output(1)
38	AOUTL	AOUTL	AOUTL	Left Analog Output
39	AOUTR	AOUTR	AOUTR	Right Analog Output
40	BOOT	BOOT	BOOT	Boot Enable
41	/RESET	/RESET	/RESET	Hardware Reset
42	DGND4	DGND4	DGND4	Digital Ground
43	VD4	VD4	VD4	Positive Digital Power Supply, +5V
44	/CS	/CS	/CS	Chip Select Input, for SPI mode

†:Pins with differences between the CS4920, CS4920A, and CS4921.
1:See appendix A for a more detailed description of this CS4920A/21 pin.
2:See the typical circuit diagrams on the following pages for filter description.
3:Recommend a pull-down resistor for the PIO pin of the CS4920A.
4:VREF should be a No Connect for the CS4920 to improve DAC performance.

Table 1. CS4920/20A/21 Pin-to-Pin Comparison

CS4920/20A/21 Typical Circuits

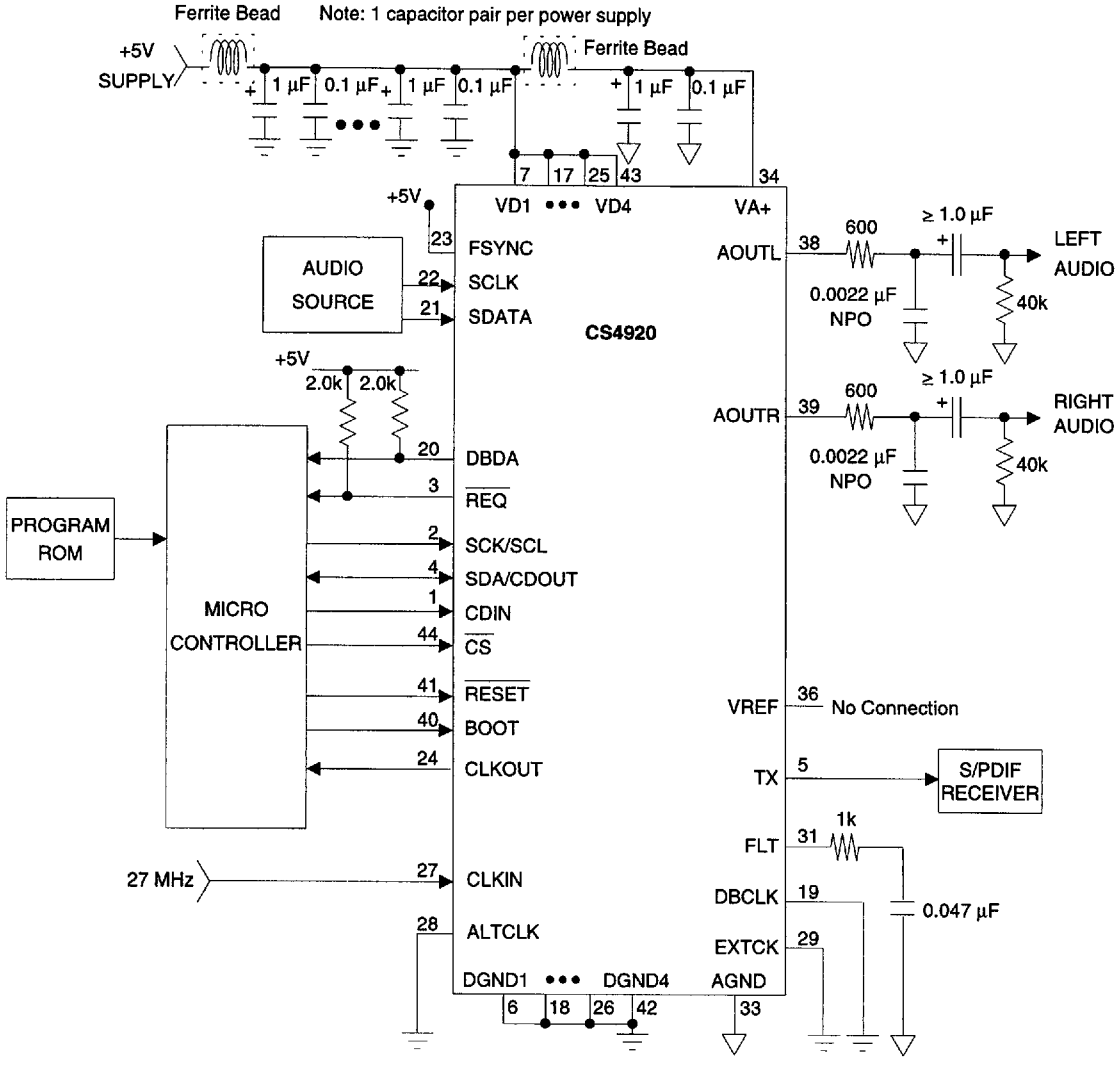


Figure 1. Typical Connection for the CS4920

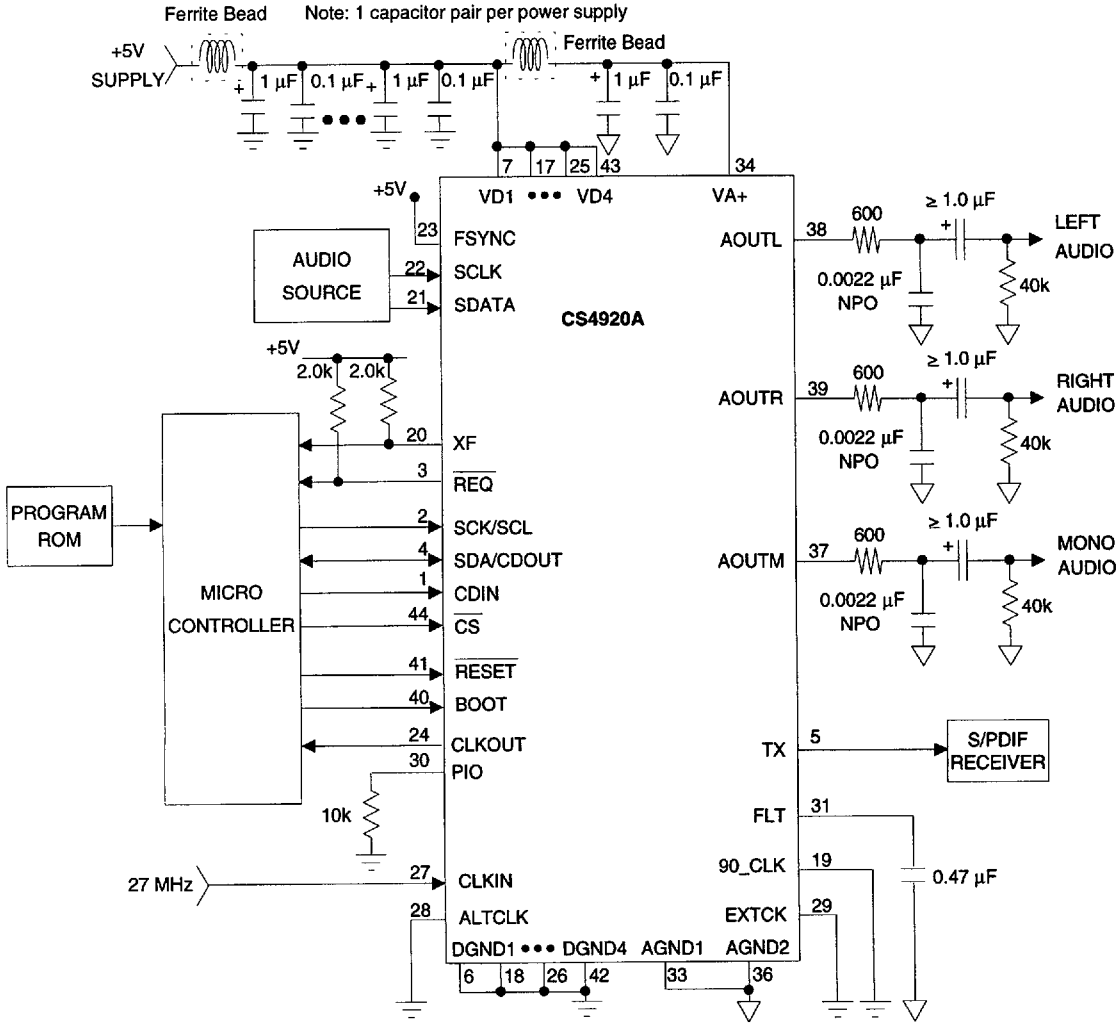


Figure 2. Typical Connection for the CS4920A

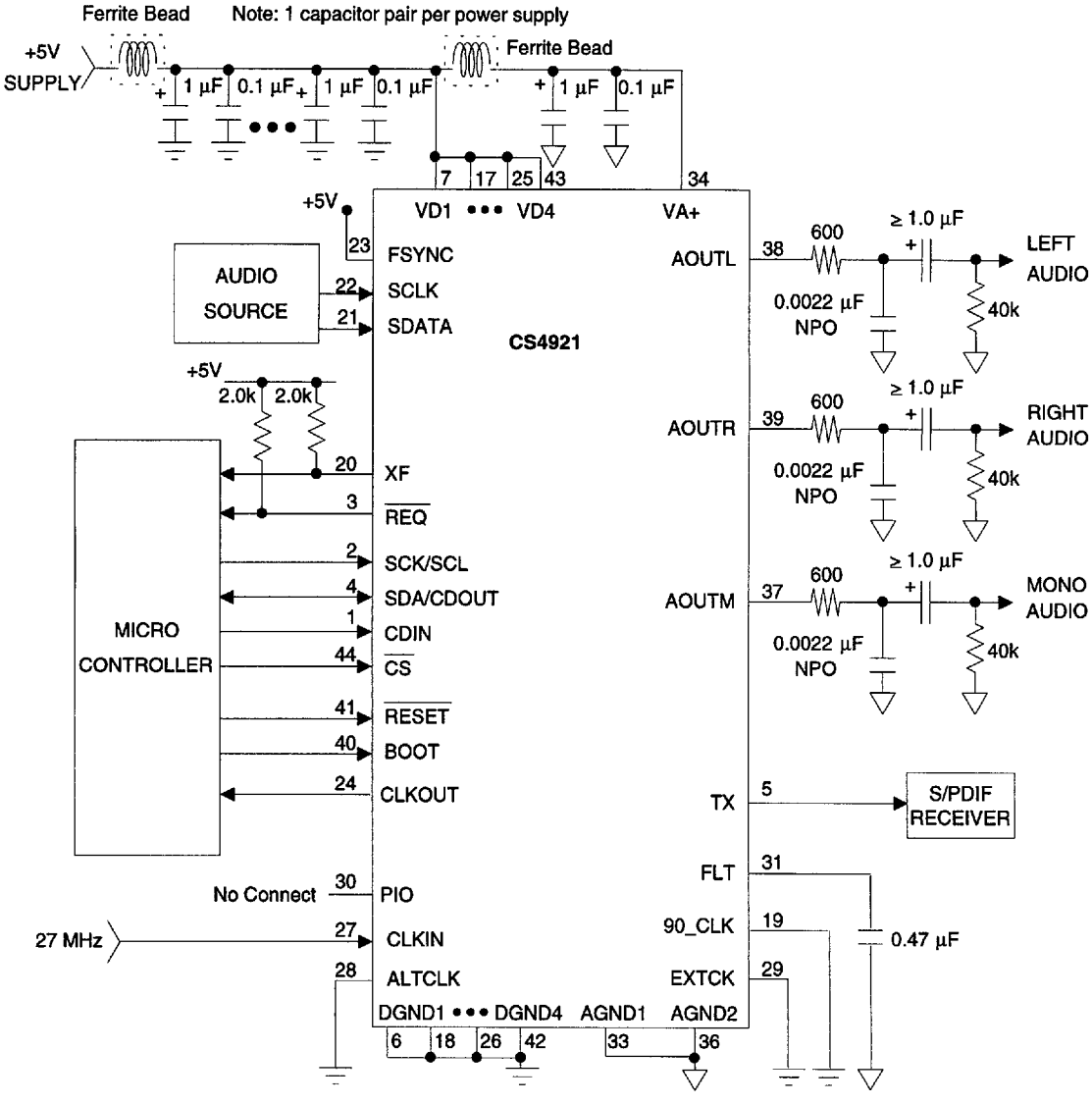


Figure 3. Typical Connection for the CS4921

CS4920/20A/21 Software Comparison

Software tools, sample source code, and DSP firmware for the CS4920 and the CS4920A are provided on the Crystal Bulletin Board System (BBS). For BBS access information: call (512) 445-7222 and ask for applications support for the CS4920/20A/21.

The majority of the microcode for the CS4920/20A found on the Crystal BBS is designed to decode MPEG audio data streams. A variety of microcode is available. Tables 2 and 3 provide a comparison of the MPEG audio microcode for the CS4920 and the CS4920A. The CS4920A microcode takes advantage of the hardware enhancements the CS4920A has over the CS4920.

The following software (microcode) is available on the Crystal BBS*:

CS4920: See Table 2
CS4920A: PESEV1_0.SIM,
PCMEV1_0.SIM,
LSREV1_0.SIM
(*.mot files are also available for every *.sim file)

MPEG Microcode for the CS4920

CS4920 microcode has completed its updates at version MPEG_B13.SIM. This microcode, MPEG_B13.SIM, is assembled as an application specific version. There are many versions of MPEG_B13.SIM whose only difference is the expected CLKIN frequencies and/or SCP modes of the operation. In addition, there are several versions of microcode assembled for an I2S input to the ASI port.

CS4920 MPEG Microcode	Control Port Mode I^2C/SPI	CLKIN Frequency	I2S Support on ASI Port
MPEG_B13.SIM	SPI	10.572 MHz	No
MPG10I2C.SIM	I^2C	10.572 MHz	No
MPG10ADR.SIM	I^2C w/ address checking	10.572 MHz	No
MPEG_27.SIM	SPI	27 MHz	No
MPG27I2C.SIM	I^2C	27 MHz	No
MPG27ADR.SIM	I^2C w/ address checking	27 MHz	No
MPEG_13.SIM	SPI	13.5 MHz	No
MPG13I2C.SIM	I^2C	13.5 MHz	No
MPG13ADR.SIM	I^2C w/ address checking	13.5 MHz	No
Z_MPG27.SIM	SPI	27 MHz	Yes
Z_I2C27.SIM	I^2C	27 MHz	Yes
Z_ADDR27.SIM	I^2C w/ address checking	27 MHz	Yes
Z_MPG13.SIM	SPI	13.5 MHz	Yes
Z_I2C13.SIM	I^2C	13.5 MHz	Yes
Z_ADDR13.SIM	I^2C w/ address checking	13.5 MHz	Yes

Details of the derivatives listed in Table 2 are provided in the *readme.txt* on the Crystal BBS in the CS4920 login.

Table 2. System-Dependent MPEG Microcode Versions of *MPEG_B13.SIM*

*For the latest revision of microcode, please see README.TXT on the Crystal BBS.

The CS4920 microcode does not allow for the system to change the values in the I/O registers. The microcode will immediately start decoding audio data when data is provided to the ASI port.

CS4920 microcode uses values for M&N in the clock manager that produces inexact values for CLKIN = 27 MHz and CLKIN = 13.5 MHz.

MPEG Microcode for the CS4920A

The naming convention for the MPEG microcode reflects the additional features, silicon revision letter, and the microcode revision number. Unlike the microcode for the CS4920, the microcode for the CS4920A is not assembled for a specific application. The CS4920A microcode requires that the MSG_CM1_INIT [1] command be sent to the SCP interface to start decoding the data on the ASI port. All the microcode versions for the CS4920A have a basic set of features. The major features[2] found in the basic set support MPEG 1 Layer II decode, XF audio data transfer[3], channel swapping, ancillary data channel, and CRC error concealment. Features that are version specific are: (LSREV1_0.SIM) low sample rate for MPEG 2 and Layer 1 for both MPEG 1 and MPEG 2, (PCMEV1_0.SIM) PCM passthru for 8 bit unsigned mono and stereo and 16 bit signed mono and stereo for all sample rates between 8 and 48 kHz, and (PESEV1_0.SIM) MPEG 1 audio packet parsing and MPEG 2 audio PES parsing. Table 3 shows the relationship of the features for all currently available versions of MPEG microcode. The microcode in the ROM of the CS4921 supports all of the features of the CS4920A microcode versions.

1: MSG_CM1_INIT command initializes the variables that are CLKIN specific. For further details see the commands document on the BBS named cmmdv#_#.eps, which is a encapsulated postscript file that can be copied to a printer.
2: See Table 3 on page 8 for complete list of features.
3: See Appendix A for a pin description of the External Flag(XF).

Feature	CS4920 MPEG_B13.SIM	CS4920A PESEV2_0.SIM	CS4920A LSREV3_2.SIM	CS4920A PCMEV1_0.SIM	CS4921 ROM
User access to read and write I/O registers	no	yes	yes	yes	yes
33-bit unsigned counter for A/V sync	no	yes	no	no	yes
Unsolicited PCR requests for A/V sync	yes	no	no	no	no
Pause/play of audio playback	no	yes	yes	no	yes
MPEG 2 audio PES parsing	yes	yes	no	no	yes
MPEG 1 audio packet parsing	no	yes	no	no	yes
PCM passthru	no	no	no	yes	yes
MPEG 1 layer 1 audio decoding	no	no	yes	no	yes
MPEG 1 layer 2 audio decoding	yes	yes	yes	yes	yes
XF audio data transfers	no	yes	yes	yes	yes
Ancillary Data up to 9600 baud	yes	yes	yes	yes	yes
CRC error concealment	yes	yes	yes	yes	yes
Independent left/right digital volume control	yes	yes	yes	yes	yes
User controlled mute	yes	yes	yes	yes	yes
Switchable playback (L + L) (R + R) (L + R) (R + L)	yes	yes	yes	no	yes
MPEG 2 low sample rate layer I, II	no	no	yes	no	yes

Table 3. Microcode Features*

*For the latest revision of microcode, please see README.TXT on the Crystal BBS.

APPENDIX A: CS4920A/21 ENHANCEMENTS OVER THE CS4920

Pin Definitions

Pin 37

Mono analog output. The stereo pair are summed for a mono channel with output scale equal to the average of the stereo outputs. Changing volume on the left/right channels will change volume on mono. This circuit eliminates the need for external mixing to accommodate 3-4 channel modulators for TV applications. The mono output is enabled when the Mono_Out bit in the CM0 register is set high. The mono output may be muted when the Mono_Mute bit in the CM0 register is set high. Implementation of Mono_Out results in the signal being 180° out of phase with the stereo outputs.

Effect on Pin Out

Mono_Out is pin 37 on 44 PLCC. On the CS4920 this pin is a NC. The default state of this pin is output disabled. Therefore, the output will not drive the line. When the output is enabled, it must not be shorted to GND. The same output circuit (1k ohm, 2200 pF) as required on AOUTL and AOUTR should be used on Mono_Out.

Mono Out Design Specs

S/(N+D) performance target is 65dB
30kΩ minimum load resistance (10% tolerance)
20pF capacitance max load

Pin 30

PIO (Programmed Input/Output). A programmable I/O pin serves to accommodate system features as defined by the user. This pin is configured as an input by default. A weak external pull down (10 k ohms) is used to avoid a floating input for applications which treat this pin as a NC. Note the pull-down is not required for the CS4921. In order to minimize cross-talk of the PIO pin with the FLT pin, the rise and fall time of the output will be about 200ns for a 20pF load. Changing the 'I-O' bit in the LINT register allows the port to handle input or output.

When configured as an input, a low to high transition on the PIO pin will assert the PIN bit in register LINT. This will also trigger a long interrupt if PIEN bit in register LINT is set to one.

When configured as an output, the status of 'POUT' bit in the LINT register will be the logic level on the PIO pin.

Pin 20

DBDA/XF. The external flag is controlled through the resident program on the CS4920A/21. When pin 20 is used in either mode, Debug or External Flag, an external pull-up is required (4.7 k ohms). A complete description of the XF is provided in the "Additional CM0 Register Bits" section.

Pin 19

DBCLK/SCRCLK. The SCRCLK is an optional input clock used to drive the SCR/PCR counter. A complete description of the SCRCLK is provided in the "Additional CM0 Register Bits" section.

Pin 36

VREF. The VREF pin of the CS4920 is the AGND2 pin of the CS4920A/21. The external capacitor which was recommended for the VREF pin should be replaced with a 'zero ohm' resistor (wire) when converting from a CS4920 to a CS4920A/21. This will connect the new AGND2 pin to the AGND.

Register Definitions

33-bit counter

The 33-bit-counter can be used to support MPEG synchronization of audio and video. This loadable counter is targeted to operate at 90kHz. The 90kHz clock may be derived from a 27MHz master clock provided at CLKIN (if available) or

from a 90kHz clock provided at Pin 19 DBCLK/SCRCLK. The selection of the counter clock is made via the register bit DIV in register CM0. When set, the DIV bit divides the clock at CLKIN by 300 and provides the divided clock to 33-bit-counter.

Count Read

The 33-bit-counter is implemented with two additional I/O registers. A 24 bit count register (CNT24) contains the high order bits, and a 9 bit count register (CNT9) contains the low order bits. Since the 90kHz clock can be asynchronous with the internal DSP clock, the counter values may not be valid when the DSP reads the registers. A VALID bit has been added to the 9 bit count register. When VALID is set the counter value is stable. If the VALID bit is set, the internal timing guarantees that the counter will not change state within the next two DSP instruction periods. This allows the program to read the low order 9 bit counter value without concern of ripple counting the low word to the high word.

Preset

The counter is reset by writing a value into the two counter registers. In order to preset the counter, the user should load the low order 9 bit counter value followed by the high order 24 bit counter value.

Additional CM0 Register Bits

SCREN Bit

A SCREN bit in register CM0 is used to enable the counter function. When enabled, the debug port functions are replaced with SCR counter functions. Pin 19 switches from the debug clock source to the SCR clock source; Pin 20 switches from the debug data pin to an external flag function. The SCREN register value is initialized to zero after RESET. Therefore, the associated pins are used for debug as a default condition.

External Flag Bit

Pin 20 can be configured as a programmable output. The XF function is selected when the SCREN bit in CM0 is set. The value of the XF bit in CM0 is reflected at the output pin XF. The MPEG code for the CS4920A/21 can use Pin 20 to control the flow of compressed audio data. When XF is low ASI data is requested.

OUT90 Bit

The 90kHz clock used internally to drive the 33 bit counter may be routed to the CLKOUT pin by asserting the OUT90 bit in CM0.

Additional Enhancements

REQ pin.

The REQ line of the CS4920 is guaranteed to stay high for a minimum 2 DSP clock cycles. To facilitate polling the REQ line to determine the end of a read operation, the CS4920A's REQ signal is guaranteed to stay high a minimum of 1 SCL/SCK period. See Figure A.1 for a SPI example of the CS4920A's REQ line relative timing.

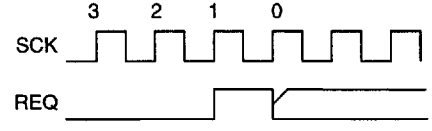


Figure A.1. REQ Pin Relative Timing in SPI Mode

PLL

The PLL circuit of the CS4920 differs from the PLL circuit of the CS4920A/21 to achieve the better loop stability and to support MPEG2 sample rates. The support for MPEG2 half sample rates (16kHz, 22.05kHz, and 24kHz) is provided via the HFS bit in register CM0. When set, this bit configures the internal clocks to 1/2 the speed of normal mode, then it doubles the DSP clock speed. The net result is that the DAC output rate is reduced to 1/2 the speed of the normal mode while the DSP runs at normal speed.

APPENDIX B: RECOMMENDED SOFTWARE UPDATES WHEN UPGRADING FROM THE CS4920 TO THE CS4920A

Crystal Semiconductor does not recommend that the CS4920 microcode be used on the CS4920A. The CS4920A microcode provides protection for the DSP against large changes in the operating frequency of the PLL during start-up and samplerate changes. Although it was the intention that the CS4920 microcode would execute on the CS4920A the hardware updates to the PLL circuitry, while providing added system flexibility and performance improvements, require a new method for insuring reliable start-up and sample rate changes.

The increased frequency bandwidth of the CS4920A's PLL requires a higher level of microcode handling. The increased protection of the DSP during a PLL frequency shift is done by placing the DSP in known state until the PLL has stabilized. While in this protected state the DSP is unable to communicate with the host. However, the CS4920A microcode can detect when a message was not received (when the feature is enabled) by checking the REJ bit of LINT register. When the SCP interface has a collision the REJ bit will be set and the CS4920A microcode can detect that a collision has occurred. When the collision handling feature is enabled the CS4920A microcode will send an unsolicited message back to the host to indicate that a collision has occurred (see MSG_COLLISION in the software commands section of the MPEG Users Guide.) The CS4920 microcode does not have the features mentioned above which are necessary to protect the CS4920A's DSP during start-up and samplerate changes. Therefore, Crystal recommends the following SCP driver changes (for a more detailed description of the CS4920A's SCP interface refer to the CS4920A data sheet):

Note: The listed procedures are only recommendations or examples. Modifications to the initialization procedure may be necessary for individual systems.

Note: At the time of this printing (9/13/95), the recommended replacement code to be used on the CS4920A is **PESEV2_0.SIM** for all systems except systems using the ZORAN I11 processor. For use with the ZORAN I11, please use **LSREV3_3.SIM**. The latest commands document available is CMMDV2_1.EPS. These files are available via the Crystal BBS (in the cs4920a login account). Please refer to the README.TXT on the BBS for the most current information.

A) Systems that use MPEG_B13.SIM microcode:
1. Download. Insure $\overline{CS}$ is held low for a minimum of 3 us from the falling edge of SCL/SCK for the final byte of the download.
2. Send command (see MSG_PLL_UNLOCK) to enable collision detection: 0x088000 (sent twice).
3. Read the SPI port to verify that the port is functioning. Host should read: 0x080001.
4. Send MSG_CM1_INIT to load variables based on a CLKIN=10.752 Mhz:
   0x220001 (sent twice, command to not use default values)
   0x027c14 (sent twice, 22.05 kHz CM1 value)
   0x005002 (sent twice, 24 kHz CM1 value)
   0x00a403 (sent twice, 16 kHz CM1 value)
   0x013c14 (sent twice, 44.1 kHz CM1 value)
   0x003403 (sent twice, 48 kHz CM1 value)
   0x005003 (sent twice, 32 kHz CM1 value)
5. Wait a minimum of 100ms before sending messages.
6. See Item (P)

B) Systems that use MPG10I2C.SIM microcode:
1. Download. Insure that $\overline{CS}$ is High during download (or any type of reset).
2. Send command (see MSG_PLL_UNLOCK) to enable collision detection: 0x088000 (sent twice).

3. Read the I2C port to verify that the port is functioning. Host should read: 0x080001.
4. Send MSG_CM1_INIT to load variables based on a CLKIN=10.752 MHz:
   0x220001 (sent twice, command to not use default values)
   0x027c14 (sent twice, 22.05 kHz CM1 value)
   0x005002 (sent twice, 24 kHz CM1 value)
   0x00a403 (sent twice, 16 kHz CM1 value)
   0x013c14 (sent twice, 44.1 kHz CM1 value)
   0x003403 (sent twice, 48 kHz CM1 value)
   0x005003 (sent twice, 32 kHz CM1 value)
5. Wait a minimum of 100ms before sending messages.
6. See Item (P)

C) Systems that use MPG10ADR.SIM microcode:
1. Hold CS4920A in reset until host is ready to program I2C address.
2. Download. Insure that $\overline{CS}$ is High during download (or any type of reset).
3. Send MSG_SCPCN to set the 7-bit I2C address to 110010 and enable address checking: 0x203371 (sent twice).
4. Send command (see MSG_PLL_UNLOCK) to enable collision detection: 0x088000 (sent twice).
5. Read the I2C port to verify that the port is functioning. Host should read: 0x080001.
6. Send MSG_CM1_INIT to load variables based on a CLKIN=10.752 MHz:
   0x220001 (sent twice, command to not use default values)
   0x027c14 (sent twice, 22.05 kHz CM1 value)
   0x005002 (sent twice, 24 kHz CM1 value)
   0x00a403 (sent twice, 16 kHz CM1 value)
   0x013c14 (sent twice, 44.1 kHz CM1 value)
   0x003403 (sent twice, 48 kHz CM1 value)
   0x005003 (sent twice, 32 kHz CM1 value)
7. Wait a minimum of 100ms before sending messages.
8. See Item (P)

D) Systems that use MPEG_27.SIM microcode:
1. Download. Insure $\overline{CS}$ is held low for a minimum of 3 us from the falling edge of SCL/SCK for the final byte of the download.
2. Send command (see MSG_PLL_UNLOCK) to enable collision detection: 0x088000 (sent twice).
3. Read the SPI port to verify that the port is functioning. Host should read: 0x080001.
4. Send MSG_CM1_INIT to load variables based on a CLKIN=27 Mhz: 0x220000 (sent twice, command to use default values)
5. Wait a minimum of 100ms before sending messages.
6. SEE Item (P)

E) Systems that use MPG27I2C.SIM microcode:
1. Download. Insure that $\overline{CS}$ is High during download (or any type of reset).
2. Send command (see MSG_PLL_UNLOCK) to enable collision detection: 0x088000 (sent twice).
3. Read the I2C port to verify that the port is functioning. Host should read: 0x080001.
4. Send MSG_CM1_INIT to load variables based on a CLKIN=27 Mhz: 0x220000 (sent twice, command to use default values)
5. Wait a minimum of 100ms before sending messages.
6. SEE Item (P)

F) Systems that use MPG27ADR.SIM microcode:
1. Hold CS4920A in reset until host is ready to program I2C address.
2. Download. Insure that $\overline{CS}$ is High during download (or any type of reset).
3. Send MSG_SCPCN to set the 7-bit I2C address to 110010 and enable address checking: 0x203371 (sent twice).
4. Send command (see MSG_PLL_UNLOCK) to enable collision detection: 0x088000 (sent twice).
5. Read the I2C port to verify that the port is functioning. Host should read: 0x080001.

6. Send MSG_CM1_INIT to load variables based on a CLKIN=27 Mhz: 0x220000 (sent twice, command to use default values)
7. Wait a minimum of 100ms before sending messages.
8. SEE Item (P)

G) Systems that use MPEG_13.SIM microcode:
1. Download. Insure $\overline{CS}$ is held low for a minimum of 3 us from the falling edge of SCL/SCK for the final byte of the download.
2. Send command (see MSG_PLL_UNLOCK) to enable collision detection: 0x088000 (sent twice).
3. Read the SPI port to verify that the port is functioning. Host should read: 0x080001.
4. Send MSG_CM1_INIT to load variables based on a CLKIN=13.5 MHz:
   0x220001 (sent twice, command to not use default values)
   0x1d48c3 (sent twice, 22.05 kHz CM1 value)
   0x11907f (sent twice, 24 kHz CM1 value)
   0x34b8ff (sent twice, 16 kHz CM1 value)
   0x1d4987 (sent twice, 44.1 kHz CM1 value)
   0x1190ff (sent twice, 48 kHz CM1 value)
   0x34b9ff (sent twice, 32 kHz CM1 value)
5. Wait a minimum of 100ms before sending messages.
6. SEE Item (P)

H) Systems that use MPG13I2C.SIM microcode:
1. Download. Insure that $\overline{CS}$ is High during download (or any type of reset).
2. Send command (see MSG_PLL_UNLOCK) to enable collision detection: 0x088000 (sent twice).
3. Read the I2C port to verify that the port is functioning. Host should read: 0x080001.
4. Send MSG_CM1_INIT to load variables based on a CLKIN=13.5 Mhz:
   0x220001 (sent twice, command to not use default values)
   0x1d48c3 (sent twice, 22.05 kHz CM1 value)
   0x11907f (sent twice, 24 kHz CM1 value)
   0x34b8ff (sent twice, 16 kHz CM1 value)
   0x1d4987 (sent twice, 44.1 kHz CM1 value)
   0x1190ff (sent twice, 48 kHz CM1 value)
   0x34b9ff (sent twice, 32 kHz CM1 value)
5. Wait a minimum of 100ms before sending messages.
6. SEE Item (P)

I) Systems that use MPG13ADR.SIM microcode:
1. Hold CS4920A in reset until host is ready to program I2C address.
2. Download. Insure that $\overline{CS}$ is High during download (or any type of reset).
3. Send MSG_SCPCN to set the 7-bit I2C address to 110010 and enable address checking: 0x203371 (sent twice).
4. Send command (see MSG_PLL_UNLOCK) to enable collision detection: 0x088000 (sent twice).
5. Read the I2C port to verify that the port is functioning. Host should read: 0x080001.
6. Send MSG_CM1_INIT to load variables based on a CLKIN=13.5 MHz:
   0x220001 (sent twice, command to not use default values)
   0x1d48c3 (sent twice, 22.05 kHz CM1 value)
   0x11907f (sent twice, 24 kHz CM1 value)
   0x34b8ff (sent twice, 16 kHz CM1 value)
   0x1d4987 (sent twice, 44.1 kHz CM1 value)
   0x1190ff (sent twice, 48 kHz CM1 value)
   0x34b9ff (sent twice, 32 kHz CM1 value)
7. Wait a minimum of 100ms before sending messages.
8. SEE Item (P)

J) Systems that use Z_MPG27.SIM microcode:
1. Download. Insure $\overline{CS}$ is held low for a minimum of 3 us from the falling edge of SCL/SCK for the final byte of the download.
2. Send command (see MSG_PLL_UNLOCK) to enable collision detection: 0x088000 (sent twice).
3. Read the SPI port to verify that the port is functioning. Host should read: 0x080001.
4. Send MSG_ASIC to configure ASI port for I2S mode: 0x1f802a (sent twice).

5. Send MSG_CM1_INIT to load variables based on a CLKIN=27 MHz: 0x220000 (sent twice, command to use default values)
6. Wait a minimum of 100ms before sending messages.
7. SEE Item (P)

K) Systems that use Z_I2C27.SIM microcode:
1. Download. Insure that $\overline{CS}$ is High during download (or any type of reset).
2. Send command (see MSG_PLL_UNLOCK) to enable collision detection: 0x088000 (sent twice).
3. Read the I2C port to verify that the port is functioning. Host should read: 0x080001.
4. Send MSG_ASIC to configure ASI port for I2S mode: 0x1f802a (sent twice).
5. Send MSG_CM1_INIT to load variables based on a CLKIN=27 Mhz: 0x220000 (sent twice, command to use default values)
6. Wait a minimum of 100ms before sending messages.
7. SEE Item (P)

L) Systems that use Z_ADDR27.SIM microcode:
1. Hold CS4920A in reset until host is ready to program I2C address.
2. Download. Insure that $\overline{CS}$ is High during download (or any type of reset).
3. Send MSG_SCPCN to set the 7-bit I2C address to 110010 and enable address checking: 0x203371 (sent twice).
4. Send command (see MSG_PLL_UNLOCK) to enable collision detection: 0x088000 (sent twice).
5. Read the I2C port to verify that the port is functioning. Host should read: 0x080001.
6. Send MSG_ASIC to configure ASI port for I2S mode: 0x1f802a (sent twice).
7. Send MSG_CM1_INIT to load variables based on a CLKIN=27 Mhz: 0x220000 (sent twice, command to use default values)
8. Wait a minimum of 100ms before sending messages.
9. SEE Item (P)

M) Systems that use Z_MPG13.SIM microcode:
1. Download. Insure $\overline{CS}$ is held low for a minimum of 3 us from the falling edge of SCL/SCK for the final byte of the download.
2. Send command (see MSG_PLL_UNLOCK) to enable collision detection: 0x088000 (sent twice).
3. Read the SPI port to verify that the port is functioning. Host should read: 0x080001.
4. Send MSG_ASIC to configure ASI port for I2S mode: 0x1f802a (sent twice).
5. Send MSG_CM1_INIT to load variables based on a CLKIN=13.5 MHz:
   0x220001 (sent twice, command to not use default values)
   0x1d48c3 (sent twice, 22.05 kHz CM1 value)
   0x11907f (sent twice, 24 kHz CM1 value)
   0x34b8ff (sent twice, 16 kHz CM1 value)
   0x1d4987 (sent twice, 44.1 kHz CM1 value)
   0x1190ff (sent twice, 48 kHz CM1 value)
   0x34b9ff (sent twice, 32 kHz CM1 value)
6. Wait a minimum of 100ms before sending messages.
7. SEE Item (P)

N) Systems that use Z_I2C13.SIM microcode:
1. Download. Insure that $\overline{CS}$ is High during download (or any type of reset).
2. Send command (see MSG_PLL_UNLOCK) to enable collision detection: 0x088000 (sent twice).
3. Read the I2C port to verify that the port is functioning. Host should read: 0x080001.
4. Send MSG_ASIC to configure ASI port for I2S mode: 0x1f802a (sent twice).
5. Send MSG_CM1_INIT to load variables based on a CLKIN=13.5 MHz:
   0x220001 (sent twice, command to not use default values)
   0x1d48c3 (sent twice, 22.05 kHz CM1 value)
   0x11907f (sent twice, 24 kHz CM1 value)
   0x34b8ff (sent twice, 16 kHz CM1 value)
   0x1d4987 (sent twice, 44.1 kHz CM1 value)
   0x1190ff (sent twice, 48 kHz CM1 value)
   0x34b9ff (sent twice, 32 kHz CM1 value)

6. Wait a minimum of 100ms before sending messages.
7. SEE Item (P)

D) Systems that use Z_ADDR13.SIM microcode:
1. Hold CS4920A in reset until host is ready to program I2C address.
2. Download. Insure that $\overline{CS}$ is High during download (or any type of reset).
3. Send MSG_SCPCN to set the 7-bit I2C address to 110010 and enable address checking: 0x203371 (sent twice).
4. Send command (see MSG_PLL_UNLOCK) to enable collision detection: 0x088000 (sent twice).
5. Read the I2C port to verify that the port is functioning. Host should read: 0x080001.
6. Send MSG_ASIC to configure ASI port for I2S mode: 0x1f802a (sent twice).
7. Send MSG_CM1_INIT to load variables based on a CLKIN=13.5 MHz:
   0x220001 (sent twice, command to not use default values)
   0x1d48c3 (sent twice, 22.05 kHz CM1 value)
   0x11907f (sent twice, 24 kHz CM1 value)
   0x34b8ff (sent twice, 16 kHz CM1 value)
   0x1d4987 (sent twice, 44.1 kHz CM1 value)
   0x1190ff (sent twice, 48 kHz CM1 value)
   0x34b9ff (sent twice, 32 kHz CM1 value)
8. Wait a minimum of 100ms before sending messages.
9. SEE Item (P)

P)After sending the MSG_CM1_INIT, the CS4920A microcode selects 22.05kHz as its sampling frequency. After validating the incoming audio MPEG stream, the microcode will automatically change the sampling frequency based on the encoded audio stream. When the sample rate does change, the microcode will place itself in a protected state (no longer than 50 ms) that will not allow the DSP to receive SCP or ASI input until the DSP clock has stabilized. By enabling the collision handling (message 0x088000), the microcode will respond with the unsolicited message, MSG_COLLISION (0x2700##), when an SCP write was attempted during the time that the DSP was in the protected state. The last byte (##) of the MSG_COLLISION will contain the command byte of the first message that was rejected while the DSP was not taking SCP input. The initialization sequences above all show how to enable the collision handling, but those systems that use the I2C interface may use the ACK (acknowledge) signal to indicate that a byte was not accepted during a write operation. The CS4920A has dedicated hardware to perform an I2C acknowledge which is not affected when the DSP is in a protected state. Regardless of the SCP interface mode, the host should confirm the port is functioning properly after detecting that a message or byte has not been received by sending a message that has a reply (e.g. MSG_PLL_UNLOCK). If the host port is not functioning properly a hardware reset should be issued, followed by the appropriate initialization sequence.

Using the new microcode on the CS4920

If the new microcode is used on the CS4920 (rev D) special precautions must be taken. The code designed for the CS4920 is not capable of playing MPEG 2 audio elementary streams recorded at the low sample rates (16, 22.05, and 24 kHz). The CS4920 is also not capable of functioning at these low sample rates. Low sample rate files should not be sent to the CS4920 for decode. In fact, when using the new microcode (PESEV2_0 or LSREV3_3) the values used for CM1_16, CM1_22, and CM1_24 in the MSG_CM1_INIT are required to be the same as the value used for CM1_32. This insures the PLL will not try to operate at a sample rate it cannot support.

If it is desired to have drivers that are capable of supporting systems with the CS4920 and the CS4920A, the software developer should treat the CS4920A as a CS4920 as far as the CM1 values are concerned. In addition, the software developer should be sure that any SCP messages that are available with the LSREV3_3 or PESEV2_0 that take advantage of the new

CS4920A features are not used, as the CS4920 does not have these features. Note, even if the software drivers are the same, the hardware updates for converting from a CS4920 to a CS4920A must be still implemented.

The following is a list of CM1_INIT values supported by both the CS4920 and the CS4920A for typical CLKIN frequencies:

CLKIN = 27 MHz

CM1_22	0x02400a
CM1_24	0x02400a
CM1_16	0x02400a
CM1_44	0x036c16
CM1_48	0x01e80d
CM1_32	0x02400a

CLKIN = 13.5 MHz

CM1_22	0x024015
CM1_24	0x024015
CM1_16	0x024015
CM1_44	0x01b416
CM1_48	0x01e81b
CM1_32	0x024015

CLKIN = 10.752 MHz

CM1_22	0x005003
CM1_24	0x005003
CM1_16	0x005003
CM1_44	0x013c14
CM1_48	0x003403
CM1_32	0x005003

Note the values stated above yield approximate values for the various sample rates (except for 10.752 Mhz). This is an errata for the CS4920 PLL (values of N above 40 are not allowed). The microcode is designed to handle the small variation in sample frequencies.