SHELDON INSTRUMENTS

SI-C31DSP-PCI User's Guide

November, 2007

Table of Contents

1.0 Introduction

The SI-C31DSP-PCI from Sheldon Instruments is a powerful Digital Signal Processor (DSP) card for your PC equipped with the PCI bus. It is based on Texas Instruments' TMS320C31, 32 bit floating point DSP, and can transform your PC into an ultra high performance development system and DSP accelerator. A full line of software development tools are available from Sheldon Instruments and TI, which include compilers, assemblers, linkers, and debuggers.

This manual describes the SI-C31DSP-PCI board, its features, design details, and the associated application interface software. Use this manual in conjunction with the TMS320C30 User's Guide, literature number SPRUO31, and AMCC's S5933 PCI Controller Data Book.

1.1 Key Features

- 40MFLOP to 60MFLOP peak performance, 32 bit floating point precision.
- 128K words of zero wait state Dual Access SRAM Memory, expandable to 256K words.
- Full bidirectional PCI initiated bus mastering, with 132MB/sec peak transfer rate.
- Memory mapped host communications port.
- Software development tools from Sheldon Instruments, including QuVIEW and QuBASE; along with separately purchased tools from TI and other third parties.
- Win9x/NT/2000/XP and Linux drivers and sample application support.
- Expansion connectors for prototyping, analog & digital I/O daughter cards.
- JTAG port for in system development and debugging.

1.2 Hardware Support

The SI-C31DSP-PCI includes expansion connectors allowing for custom designs, or for attaching 'off the shelf' multifunction I/O modules from Sheldon Instruments. Sheldon Instruments offers several daughter modules for multichannel analog and digital I/O, including 4 to 64 channels of 16 bit ADCs and DACs.

1.3 Software Support

The SI-C31DSP-PCI is available with extensive development tools from Sheldon Instruments and TI.

For quick turnkey development, Sheldon Instruments offers QuVIEW and QuBASE, which are a set of DSP-resident libraries for real time performance that greatly accelerate data acquisition, signal processing, and control applications. QuVIEW is a real time accelerator for LabVIEW, and QuBASE a real time accelerator for Visual Basic. A full range of examples and tutors are provided to demonstrate their ease of use and breadth of functionality and capabilities. QuBASE runs under Win9x/NT/2000/XP, while QuVIEW also runs under Linux.

Typical benchmarks for a 50Mhz C31 processor include the computation of the real portion of a 1024 point Radix-2 FFT at 1.23ms. Other benchmarks include real-time, DSP based DMA transfers over the PCI bus to memory at sustainable rates of up to 20Mbytes/sec, and up to

800Kbytes/sec gap free transfers to hard disk.

When purchased as a DSP evaluation board, Sheldon Instruments also includes free sample DSP and Win9x/NT/2000/XP or Linux device driver source code to accompany TI's development environment. The DSP source code illustrates full communication modes, and the Win9x/NT/2000/XP or Linux device driver source code includes the complete SI-DDK, along with COFF file loader utilities.

1.4 General Description - Functional Overview

NOTE: *The communications protocol provided in this manual assumes that the host is the master and that the TMS320C31 is the slave.*

The basic elements of the protocol are as follows:

1) System initialization
    Host
    TMS320C31
2) Command execution
    32-bit DMA transfers

The examples given in this manual support data transfers, but they can be easily extended to support any type of command.

2.0 System Initialization

The host initializes the SI-C31DSP to support communications by doing the following:

1) Host resets or inactivates the SI-C31DSP card by holding the TMS320C31 in an inactive/reset condition.
2) Host downloads the TMS320C31 monitor/communications code (PCIINI32.OUT COFF file) into the SI-C31DSP by means of Dual Access Writes.
3) Host activates the TMS320C31 by removing the reset condition in order to launch DSP code execution.

The TMS320C31 debugger (via the JTAG port) or the stand alone COFF loader (SISAMPLE.EXE) is used to accomplish these operations.

After the DSP is launched with instructions contained inside the PCIINI32.OUT COFF file, several initialization steps are performed. Please refer to the source code implemented in the PCIINIT.ASM routine to verify the following steps.
1) DSP clears and disables interrupts.
2) DSP initializes the data page pointer and the stack pointer.
3) DSP enables cache memory.
4) DSP initializes the memory interface.
5) DSP enables TMS320C31 INT0 and the global interrupt bit.
6) DSP awaits command interrupts.

2.1 System Resets

System reset is essential for proper SI-C31DSP operation. The SI-C31DSP is reset by the host PCI bus '!SYS_RESET' signal at system power up. However, you can also reset the SI-C31DSP with software by writing to bit 0 (!C3x_RST) of the CONTROL/STATUS register, which is mapped into the S5933's pass-thru region with a Dword boundary offset of 0x3 ((byte boundary 0x8), and is physically implemented inside the onboard logic or FPGA. Bit 0 of this register is tied to the low asserted TMS320C31 '!RESET' pin. This reset operation can be seen in the sample application.

All code for the SI-C31DSP-PCI board is loaded by direct host accesses (Dual Memory access mode) into the TMS320C31 memory. The TMS320C31 is placed in a high impedance state with the RESET pin during PCI side COFF file writes. The DSP is launched to begin execution by subsequently removing it from the RESET or idle state. There is no onboard boot PROM/EPROM.

3.0 Command Execution: Host PCI bus to DSP Data Exchange

All communication between the host and the DSP take place by means of AMCC's PCI matchmaker ASIC, the S5933. There are two basic modes of data exchange that can be described that reflect either the state of the host or the state of the DSP.

From the host or PCI side, SI-C31DSP card is accessed as one of the following:
1) a target or slave device, where the host is the PCI bus master and it controls access to all resources resident on the SI-C31DSP card.
2) a bus master, where the SI-C31DSP card becomes the bus master and it performs the transfers to host with the DSP actively running code.

From the SI-C31DSP side, data is exchanged in one of two ways:
1) Passive mode, without DSP intervention, thereby making it irrelevant whether the DSP is actively running code or in an idle RESET state.
2) Active mode mode, where the DSP must be actively running code for data to be exchanged between the host and the DSP. For active DSP communications, the host must first access the Communication registers followed by an interrupt to alert the DSP of a pending transaction.

	Host: PCI Bus Master SI-C31DSP: PCI Target/Slave	**Host: PCI Target/Slave SI-C31DSP: PCI Bus Master**
Passive Mode: *DSP Inactive or Active*	Dual Access Mode (Via FPGA CSRs)	
Active Mode: *DSP Actively Running Code*	Host Poll (Via DSP Memory Mapped Communication Registers)	Bus Master Transfers (via AMCC FIFOs and DSP Memory Mapped Communication Registers)

3.1 Slave/Target: Dual Access Mode

For the dual access mode, all of the DSP's zero wait state SRAM memory is accessible by both the PCI bus and the DSP. Only three (3) of the five (5) memory mapped registers located inside of the FPGA are used (these registers are mapped into the PCI bus using the S5933's pass-thru region):
1) DAM DATA READ (pass-thru read)
2) DAM DATA WRITE (pass-thru write)
3) DAM ADDRESS REGISTER.

The procedures are implemented in the sequence outlined below.

1) PCI side write to the DAM ADDRESS REGISTER. The actual 24 bit C31 address value is written into this FPGA register.
2) PCI side access (read/write) of the DAM DATA READ or DAM DATA WRITE registers. The actual 32 bit C31 data value is accessed from these FPGA registers.

Onboard control logic residing inside of the FPGA arbitrates the appropriate timing between the

C31's primary bus and the DAM ADDRESS/DATA registers. When the DSP is active running code, this is accomplished by asserting the C31 HOLD signal for no more than three (3) of its own clock cycles, every time the PCI side performs an access. Asserting the C31's HOLD signal while the DSP is actively running code places the DSP's primary bus into a high impedance state. When the DSP is inactive or in an idle RESET state, the DSP is in a high impedance state by default, thereby accesses to the DSP's memory.

3.2 Software Procedure for Dual Access Mode Reads and Writes

Dual Access reads and writes are accomplished by simply writing the source/destination DSP address to the pass-thru read/write registers mapped in the AMCC's pass-thru region (DAM ADDRESS_READ/DAM ADDRESS_WRITE). The resulting data will be accessed from the Dual Access Data register also in the AMCC's pass-thru region (DAM DATA). The definitions for these register offsets can be found in the file SIPCI.H (SI_DUAL_ACCESS_READ_ADDRESS_OFFSET, SI_DUAL_ACCESS_WRITE_ADDRESS_OFFSET, SI_DUAL_ACCESS_DATA_OFFSET) and examples of their implementation can be found in the file AMCCLIB.CPP under the functions:

DWORD _stdcall Dual_Access_Read(LPDIOC lpDIOCtl)
DWORD _stdcall Dual_Access_Write(LPDIOC lpDIOCtl)

Furthermore, the SI-SAMPLE application shows how to call these commands, as well as the COFF loader, using the SI-PCI device driver.

3.3 Bus Master Mode Access

In addition to the dual access memory, large blocks of data can be transferred via the S5933's bidirectional FIFO, using the PCI initiated bus mastering feature in conjunction with the DSP's DMA engine. PCI initiated bus mastering allows for the highest possible burst transfers to take place over the PCI bus without host processor intervention. The S5933's FIFOs and Mailbox registers are mapped into the DSP's primary bus, and are used to perform PCI initiated bus master transfers. Note that the host will first write to a set of Communication Registers that are physically resident inside of the DSP's external memory as outlined below.

NOTE: *The S5933's Add-On initiated bus mastering, equivalent to slave card bus mastering, is not implemented.*

3.4 Active Mode Communication Registers

NOTE*: For DSP programmed I/O and DSP DMA accesses to occur, a protocol involving software and hardware handshaking is implemented. Therefore, it is imperative that the DSP is active running code, which requires downloading the DSP binary (COFF file contents) before the DSP is able to perform active communications.*

NOTE*: Please refer to the SICommModes.doc document for more details.*

Register Name: ***CommReg0:*** *Communication Mode/Control Register*.
DSP Memory Offset (DWord): 0x30
Data Width: 32 bits
Description: Written by host processor, defines the type of transfer to take place. Written by the host using DAM mode.

Register Name: ***CommReg1****: Count Register in DWords*.
DSP Memory Offset (DWord): 0x31
Data Width: 21 bits (2Mx32 depth)
Description: Written by host processor, defines the size or number of data values to transfer.

Register Name: ***CommReg2****: Source Address Register*.
DSP Memory Offset (DWord): 0x32
Data Width: 32 bits
Description: Written by host processor, defines the source address within the DSP's memory range, of the data to be transferred.

Register Name: ***CommReg3****: Destination Address Register*.
DSP Memory Offset (DWord): 0x33
Data Width: 32 bits
Description: Written by host processor, defines the destination address within the DSP's memory range, of the data to be transferred.

Register Name: ***CommReg4****: Flag/Status Register*.
DSP Memory Offset (DWord): 0x34
Data Width: 32 bits
Description: Accessed by both the host and the DSP, serves as a status indicator for synchronizing both the host and DSP processors.

Register Name: ***CommReg5****: Data Register*.
DSP Memory Offset (DWord): 0x35
Data Width: 32 bits
Description: Optional register used only for "Host Polling" transfer modes. Accessed by both the host and the DSP, and serves as an intermediate depository for the current data value being transferred.

NOTE: *Only valid for those transfers requiring both processors to poll one another. Otherwise, the data is transferred through a FIFO buffer inside of the PCI bridge device.*

3.5 Software Procedure for PCI Initiated Bus Mastered Writes

The details of this procedure can be seen in the file AMCCLIB.CPP in the function DWORD _stdcall BusMastered_Write(LPDIOC lpDIOCtl). The DSP side of these communications can be found in the file PCIINIT.ASM.

First, the DSP command needs to be set up for proper operation.

1. Stop the INT1 from occurring. This is done by writing a "0" to BIT 2 (RDFIFO_EN) of the CONTROL register. The CONTROL register is mapped with a Dword boundary offset of 0x3 (byte boundary 0x8) added to the base address.
2. Set the command to Bus Mastered Write. This is done by writing a "1" to the S5933 PCI to Add-on Mailbox register 1. This "1" value will tell the DSP's DMA controller that the INT1 will trigger DSP read operations.
3. Set the Count variable. This is done by writing the number of points to transfer to the S5933 PCI to Add-on Mailbox register 2.
4. Set the DSP Address variable. This is done by writing the DSP's target address from which to perform DMA writes, to the S5933 PCI to Add-on Mailbox register 3.
5. Generate an INT0 to start transfer on the DSP side (this calls cmd_isr on the DSP). This is done by performing a write to the INT0 register. The INT0 is only a decoded register mapped with a Dword boundary offset of 0x4 (byte boundary 0x10) added to the base address.

Now set up the AMCC 5933 for the actual transfer.

1. Disable PCI FIFO to Add-on Bus Mastering by writing a "0" to MCSR bit 14.
2. Disable host interrupts from Transfer Count=0 by writing "0" to all INTCSR bits.
3. Optionally reset FIFO Flags by writing a "1" to MCSR bit 25 (empty the FIFO). This operation may or may not be necessary depending upon the application.
4. Define FIFO Management Scheme as go-on-half-way (4 words) by writing a "1" to MCSR bit 13.
5. Define Read/Write Priority for "Write Priority" by writing a "0" to MCSR bit 8, and wrting a "1" to MCSR bit 12.
6. Write the host address into the MRAR.
7. Write the byte count into the MRTC. This count value is 4 times the DSP count value defined in the PCI to Add-on Mailbox register 2.
8. Re-enable host interrupts from Transfer Count = 0 by writing a "1" to INTCSR bit 15.
9. Enable Add-on to PCI FIFO Bus Mastering by writing a "1" to MCSR bit 14.
10. Start the INT1s on the DSP side. The DSP's INT1 trigger the DSP's DMA controller to perform read/write accesses. This is done by writing a "1" to BIT 2 (RDFIFO_EN) of the CONTROL register. The CONTROL register is mapped with an offset of 0x3 added to the base address.

3.6 Software Procedure for PCI Initiated Bus Mastered Reads

The details of this procedure can be seen in the file AMCCLIB.CPP in the function DWORD _stdcall BusMastered_Write(LPDIOC lpDIOCtl). The DSP side of these communications can be found in the file PCIINIT.ASM.

First, the DSP command needs to be set up for proper operation.

1. Stop the INT1 from occurring. This is done by writing a "0" to BIT 1 (WRFIFO_EN) of the

CONTROL register. The CONTROL register is mapped with an offset of 0x3 added to the base address.
2. Set the command to Bus Mastered Read. This is done by writing a "0" to the S5933 PCI to Add-on Mailbox register 1. This "0" value will tell the DSP's DMA controller that the INT1 will trigger DSP write operations.
3. Set the Count variable. This is done by writing the number of points to transfer to the S5933 PCI to Add-on Mailbox register 2.
4. Set the DSP Address variable. This is done by writing the DSP's source address from which to perform DMA reads, to the S5933 PCI to Add-on Mailbox register 3.
5. Generate an INT0 to start transfer on the DSP side (this calls cmd_isr on the DSP). This is done by performing a write to the INT0 register. The INT0 is only a decoded space mapped with an offset of 0x4 added to the base address.

Now set up the 5933 for the transfer.

1. Disable PCI FIFO to Add-on Bus Mastering by writing a "0" to MCSR bit 10.
2. Disable host interrupts from Transfer Count=0 by writing "0" to INTCSR bits 14 and 15.
3. Optionally reset FIFO Flags by writing a "1" to MCSR bit 26 (empty the FIFO). This operation may or may not be necessary depending upon the application.
4. Define FIFO Management Scheme as go anytime.
5. Define Read/Write Priority for "Read Priority" by writing a "1" to MCSR bit 8, and wrting a "0" to MCSR bit 12.

6. Write the host address into the MWAR.
7. Write the byte count into the MWTC. This count value is 4 times the DSP count value defined in the PCI to Add-on Mailbox register 2.
8. Re-enable host interrupts from Transfer Count = 0 by writing a "1" to INTCSR bits.
9. Enable Add-on to PCI FIFO Bus Mastering by writing a "1" to MCSR bit 10.
10. Check for response from DSP. The DSP should return a "1" to the Add-on to PCI mailbox register 1, indicating that the DSP is done setting up the DMA.
11. Start the INT1s on the DSP side. The DSP's INT1 trigger the DSP's DMA controller to perform read/write accesses. This is done by writing a "1" to BIT 1 (WRFIFO_EN) of the CONTROL register. The CONTROL register is mapped with an offset of 0x3 added to the base address.

4.0 Register Mapping

4.1 Register Mapping As Seen from the Host

These registers are mapped through the S5933's pass-thru region, and physically resident inside of the FPGA. Their address offsets are added to the base address stored in the Base Address 1 register defined by the PCI Configuration Space Headers. Please refer to the S5933 Data Book for more details. These registers are referred to in the demo software as CSR registers.

Address Offset	**Register Name (FPGA)**	**Function**
Dword boundary: 0x0 (byte boundary: 0x0)	DAM ADDRESS_READ	Specifies the DSP's address for Dual Access reads.
Dword boundary: 0x1 (byte boundary: 0x4)	DAM ADDRESS_WRITE	Specifies the DSP's address for Dual Access writes.
Dword boundary: 0x2 (byte boundary: 0x8)	DAM DATA	Contains data to be accessed (read/written) from the DSP.
Dword boundary: 0x3 (byte boundary: 0xC)	CONTROL/STATUS	Controls DSP reset.
Dword boundary: 0x4 (byte boundary: 0x10)	INT0	Any access to this address generates a C31 INT0.

4.2 Control/Status Register Bits Summary

The Control/Status Register controls several SI-C31DSP operations.

Bit Number	**Function**	**Description**
0	C3x_RST	A low asserted signal tied directly to the TMS320C31's Reset pin.
1	RD_FIFO_EN	**No longer supported*. A high asserted signal that enables INT1 operation in order to trigger TMS320C31 DMA reads (PCI side writes) from the PCI to Add-on FIFO.
2	WR_FIFO_EN	**No longer supported*. A high asserted signal that enables INT1 operation in order to trigger TMS320C31 DMA writes (PCI side reads) to the Add-on to PCI FIFO.
3	!MANUAL/BURST	**No longer supported*. When “0", PCI side FIFO accesses are performed by manually reading and polling the FIFO status bits. When “1", PCI side FIFO accesses are performed by PCI initiated bus master transfers.

4.3 CONTROL/STATUS Register Values

Function	Enable Value	Disable Value
Reset C31	0x0	0x1
**No longer supported*. Manual Add-on FIFO READ	0x3	0x1
**No longer supported*. Bus Master Add-on FIFO READ	0xB	0x1
**No longer supported*. Manual Add-on FIFO WRITE	0x5	0x1
**No longer supported*. Bus Master Add-on FIFO WRITE	0xD	0x1

4.4 Register Mapping As Seen from the DSP

This section describes memory mapping as seen from the C31 DSP. When the DSP is running, its DWord boundary memory mapping is as follows:

External SRAM Bank 0 (always present):	0x000000 to 0x00FFFF (64kx32)
External SRAM Bank 1 (optional):	0x010000 to 0x01FFFF (64kx32)
External SRAM Bank 2 (optional):	0x020000 to 0x02FFFF (64kx32)
Expansion I/O:	0xFE0000 to 0xFE1FFF (8kx32)
AMCC Registers (FIFO and Opregs):	0xFF0000 to 0xFF000F (16x32)

NOTE:
1) The DSP's SRAM bank is also accessible from the host, though indirectly, using the FPGA's DAM registers. Therefore, when writing a value into the DAM Address Register, it must reflect the DSP's physical SRAM address, with Dword boundary, ranging from 0x000000 to 0x020000.
2) The C31 DSP reserves a small block of memory ranging from 0x000000-0x00002F for its program vectors. Please consult the TMS320C31 manual for more details.

5.0 Custom Hardware Interface

The SI-C31DSP-PCI includes expansion connectors allowing for custom designs, or for attaching 'off the shelf' multifunction I/O modules from Sheldon Instruments. Sheldon Instruments offers several daughter modules for multichannel analog and digital I/O, including 4 to 64 channels of 16 bit ADCs and DACs.

Two mechanical versions of the SI-C31DSP-PCI card exist to accommodate different expansion modules, designated RevA and RevB; electrically they are both identical. Their connector differences are listed below.

5.1 RevA User I/O Expansion: 9 and 15 Pin D-sub to 10 and 16 Pin IDC Connection

A pair of standard DSUB connectors with 9 and 15 contacts, designated DB1 and DB2 respectively, are used to interface external user defined signals to one of 10 or 16 pin IDC style of connectors, designated P1 and P2 respectively. This P1 DIN connector is linked to the custom daughter card. Neither of these connectors contain any signals from the DSP. Below is the connection diagram.

P2.1 - DB2.1	P2.9 - DB2.9
P2.2 - DB2.2	P2.10 - DB2.10
P2.3 - DB2.3	P2.11 - DB2.11
P2.4 - DB2.4	P2.12 - DB2.12
P2.5 - DB2.5	P2.13 - DB2.13
P2.6 - DB2.6	P2.14 - DB2.14
P2.7 - DB2.7	P2.15 - DB2.15
P2.8 - DB2.8	P2.16 -

P1.1 - DB1.1	P1.6 - DB1.6
P1.2 - DB1.2	P1.7 - DB1.7
P1.3 - DB1.3	P1.8 - DB1.8
P1.4 - DB1.4	P1.9 - DB1.9
P1.5 - DB1.5	P1.10 -

5.2 RevA Serial Port Expansion: 10 Pin IDC Connection

A single 10 contact, dual row connector, designated P4, is used to directly interface to the DSP's serial port. Below is the connection diagram:

1	CLKX0	2	+5V
3	DX0	4	+5V
5	FSX0	6	DGND
7	CLKR0	8	DGND
9	DR0	10	N/C
11	FSR0	12	N/C
13	XF0	14	XF1
15	TMCK0	16	TMCK1

5.3 RevB User I/O Expansion: 100 pin D-sub to 96 pin DIN Connection

A 100 pin, half pitch (.050"), Series III DSUB connector, designated J1, is used to interface external user defined signals to one of the 96 pin DIN connectors, designated P1. This P1 DIN connector is linked to the custom daughter card. Neither of these connectors contain any signals from the DSP. Below is the connection diagram.

J٢.A١ - DB١.١	J٢.B١ - DB١.٥١	J٢.C١ - DB١.٢
J٢.A٢ - DB١.٥٢	J٢.B٢ - DB١.٣	J٢.C٢ - DB١.٥٣
J٢.A٣ - DB١.٤	J٢.B٣ - DB١.٥٤	J٢.C٣ - DB١.٥
J٢.A٤ - DB١.٥٥	J٢.B٤ - DB١.٦	J٢.C٤ - DB١.٥٦
J٢.A٥ - DB١.٧	J٢.B٥ - DB١.٥٧	J٢.C٥ - DB١.٨
J٢.A٦ - DB١.٥٨	J٢.B٦ - DB١.٩	J٢.C٦ - DB١.٥٩
J٢.A٧ - DB١.١٠	J٢.B٧ - DB١.٦٠	J٢.C٧ - DB١.١١
J٢.A٨ - DB١.٦١	J٢.B٨ - DB١.١٢	J٢.C٨ - DB١.٦٢
J٢.A٩ - DB١.١٣	J٢.B٩ - DB١.٦٣	J٢.C٩ - DB١.١٤
J٢.A١٠ - DB١.٦٤	J٢.B١٠ - DB١.١٥	J٢.C١٠ - DB١.٦٥
J٢.A١١ - DB١.١٦	J٢.B١١ - DB١.٦٦	J٢.C١١ - DB١.١٧
J٢.A١٢ - DB١.٦٧	J٢.B١٢ - DB١.١٨	J٢.C١٢ - DB١.٦٨
J٢.A١٣ - DB١.١٩	J٢.B١٣ - DB١.٦٩	J٢.C١٣ - DB١.٢٠
J٢.A١٤ - DB١.٧٠	J٢.B١٤ - DB١.٢١	J٢.C١٤ - DB١.٧١
J٢.A١٥ - DB١.٢٢	J٢.B١٥ - DB١.٧٢	J٢.C١٥ - DB١.٢٣
J٢.A١٦ - DB١.٧٣	J٢.B١٦ - DB١.٢٤	J٢.C١٦ - DB١.٧٤
J٢.A١٧ - DB١.٢٥	J٢.B١٧ - DB١.٧٥	J٢.C١٧ - DB١.٢٦
J٢.A١٨ - DB١.٧٦	J٢.B١٨ - DB١.٢٧	J٢.C١٨ - DB١.٧٧
J٢.A١٩ - DB١.٢٨	J٢.B١٩ - DB١.٧٨	J٢.C١٩ - DB١.٢٩
J٢.A٢٠ - DB١.٧٩	J٢.B٢٠ - DB١.٣٠	J٢.C٢٠ - DB١.٨٠
J٢.A٢١ - DB١.٣١	J٢.B٢١ - DB١.٨١	J٢.C٢١ - DB١.٣٢
J٢.A٢٢ - DB١.٨٢	J٢.B٢٢ - DB١.٣٣	J٢.C٢٢ - DB١.٨٣
J٢.A٢٣ - DB١.٣٤	J٢.B٢٣ - DB١.٨٤	J٢.C٢٣ - DB١.٣٥
J٢.A٢٤ - DB١.٨٥	J٢.B٢٤ - DB١.٣٦	J٢.C٢٤ - DB١.٨٦
J٢.A٢٥ - DB١.٣٧	J٢.B٢٥ - DB١.٨٧	J٢.C٢٥ - DB١.٣٨
J٢.A٢٦ - DB١.٨٨	J٢.B٢٦ - DB١.٣٩	J٢.C٢٦ - DB١.٨٩
J٢.A٢٧ - DB١.٤٠	J٢.B٢٧ - DB١.٩٠	J٢.C٢٧ - DB١.٤١
J٢.A٢٨ - DB١.٩١	J٢.B٢٨ - DB١.٤٢	J٢.C٢٨ - DB١.٩٢
J٢.A٢٩ - DB١.٤٣	J٢.B٢٩ - DB١.٩٣	J٢.C٢٩ - DB١.٤٤
J٢.A٣٠ - DB١.٩٤	J٢.B٣٠ - DB١.٤٥	J٢.C٣٠ - DB١.٩٥
J٢.A٣١ - DB١.٤٧	J٢.B٣١ - DB١.٩٦	J٢.C٣١ - DB١.٤٧
J٢.A٣٢ - DB١.٩٧	J٢.B٣٢ - DB١.٤٨	J٢.C٣٢ - DB١.٩٨
DB١.٤٩ - GND		
DB١.٥٠ - +٥V		
DB١.٩٩ -No Connect		
DB١.١٠٠ -No Connect		

5.4 RevA and RevB DSP Bus Expansion: 96 pin DIN to C31 Bus Connector

A 96 pin DIN located away from the user I/O connector, designated J3, is used to interface the custom daughtercard to the C31's data, address, and control busses. 8Kx32 words are decoded for use by the custom daughtercard, which is mapped into the DSP's primary bus, Dword boundary address space ranging from 0xFE0000 to 0xFE1FFF. Please refer to the TMS320C3x User's Manual, Chapter 9, for further details. Below is the pinout.

J٣.A١ - +٥V	J٣.B١ - D٠	J٣.C١ - +٥V
J٣.A٢ - D١	J٣.B٢ - A٠	J٣.C٢ - D٢
J٣.A٣ - A١	J٣.B٣ - D٣	J٣.C٣ - A٢
J٣.A٤ - D٤	J٣.B٤ - A٣	J٣.C٤ - D٥
J٣.A٥ - A٤	J٣.B٥ - D٦	J٣.C٥ - A٥
J٣.A٦ - D٧	J٣.B٦ - A٦	J٣.C٦ - D٨
J٣.A٧ - A٧	J٣.B٧ - D٩	J٣.C٧ - A٨
J٣.A٨ - D١٠	J٣.B٨ - A٩	J٣.C٨ - D١١
J٣.A٩ - A١٠	J٣.B٩ - D١٢	J٣.C٩ - A١١
J٣.A١٠ - D١٣	J٣.B١٠ - A١٢	J٣.C١٠ - D١٤
J٣.A١١ -	J٣.B١١ - D١٥	J٣.C١١ -
J٣.A١٢ - D٣٠	J٣.B١٢ - !IOMAP	J٣.C١٢ - !INT٢
J٣.A١٣ - MEM_R/!W	J٣.B١٣ - D٣١	J٣.C١٣ - !INT٣
J٣.A١٤ - D٢٦	J٣.B١٤ - !IACK	J٣.C١٤ - D٢٩
J٣.A١٥ - D٢٥	J٣.B١٥ -	J٣.C١٥ - D١٧
J٣.A١٦ - D٢١	J٣.B١٦ - D٢٤	J٣.C١٦ - D٢٨
J٣.A١٧ - D١٨	J٣.B١٧ - D٢٠	J٣.C١٧ - GND
J٣.A١٨ - D٢٣	J٣.B١٨ - GND	J٣.C١٨ - D١٩
J٣.A١٩ - GND	J٣.B١٩ - D١٦	J٣.C١٩ - GND
J٣.A٢٠ - !MSTRB	J٣.B٢٠ - +٥V	J٣.C٢٠ -
J٣.A٢١ - +٥V	J٣.B٢١ - D٢٢	J٣.C٢١ - +٥V
J٣.A٢٢ - D٢٧	J٣.B٢٢ - TMCK١	J٣.C٢٢ - XF٠
J٣.A٢٣ - TMCK٠	J٣.B٢٣ -	J٣.C٢٣ - !RDY
J٣.A٢٤ - XF١	J٣.B٢٤ - GND	J٣.C٢٤ - H٣
J٣.A٢٥ - H١	J٣.B٢٥ - GND	J٣.C٢٥ - GND
J٣.A٢٦ - GND	J٣.B٢٦ -	J٣.C٢٦ -
J٣.A٢٧ -	J٣.B٢٧ -	J٣.C٢٧ - GND
J٣.A٢٨ - GND	J٣.B٢٨ - CLKX٠	J٣.C٢٨ - CLKR٠
J٣.A٢٩ - DX٠	J٣.B٢٩ - DR٠	J٣.C٢٩ - FSX٠
J٣.A٣٠ - FSR٠	J٣.B٣٠ - +١٢V	J٣.C٣٠ - +١٢V
J٣.A٣١ -	J٣.B٣١ - -١٢V	J٣.C٣١ - -١٢V
J٣.A٣٢ -	J٣.B٣٢ - -٥V	J٣.C٣٢ - -٥V

Note: Signals MEM_R/!W, !MSTRB, and !IOMAP are decoded by the PLD designated U2. MEM_R/!W is decoded so that either the C31 or the PCI bus (via the S5933 in pass-thru mode) can control the daughter card's read/write signals. !MSTRB is decoded so that the C31's !STRB signal is active only during accesses to its daughter card Dword boundary address space ranging from 0xFE0000 to 0xFEFFFF. !IOMAP only decodes the C31's address pins during accesses to the S5933, or any other access to the C31's Dword boundary address space mapped at 0xFFXXXX. Please look at their respective equations for specific logic equations.

5.5 RevA and RevB JTAG Port

The JTAG port is a 12 pin header designated P1, polarized by an omitted pin. It is intended to be interfaced to the XDS 500 debugging system from Texas Instruments, as well as debugging tools from several third party sources.

1	EMU1	2	DGND
3	EMU0	4	DGND
5	EMU2	6	DGND
7	+5V	8	NOT CONNECTED
9	EMU3	10	DGND
11	H3 CLOCK	12	DGND

6.0 Technical Specifications

6.1 Processor for SI-C31DSP

- TMS320C31 40Mhz, 50Mhz or 60Mhz DSP.
- Single DMA channel.

6.2 Memory

- 128K x 32 bit words zero wait state dual access SRAM, expandable to 256K x 32 bit words on C31's primary bus.

6.3 Interface to Host

- Two 32 bit, bidirectional communications modes between TMS320C31 primary bus and the S5933:
    a) Dual access mode.
    b) PCI initiated bus master mode, S5933 add-on initiated bus mastering not implemented.

- The S5933's Mailbox and FIFO registers are mapped into the C31's primary bus, Dword boundary address space ranging from 0xFF0000 to 0xFF000F.

- All control functions and memory are accessed by the PCI bus through five (5) memory mapped registers in the S5933's pass thru region:
    a) DAM DATA READ (pass-thru write). PCI bus dual access mode read of C31 data.
    b) DAM DATA WRITE (pass-thru write). PCI bus dual access mode write of C31 data.
    c) DAM ADDRESS REGISTER (pass-thru write). 24 bit address of C31 data for PCI bus dual access mode transfer.
    d) CONTROL REGISTER (pass-thru write). On board control register for resets and interrupt control.
    e) C31 INT0 (pass-thru write). PCI initiated bus master interrupt to C31.

- PCI initiated bus master transfer speeds:
    a) Up to 132Mbyte/sec bursts with block sizes of eight (8) 32 bit words.
    b) Up to 20Mbyte/sec sustained transfers of any block size, using S5933 FIFOs and the DSP's DMA.

- INT0 used by C31 for basic communications and DMA transfer initialization; INT2 and INT3 available on DIN expansion connectors.

6.4 RevA User I/O Expansion.

- One standard 9 pin Dsub connector tied to one 10 pin dual row connector.
- One standard 15 pin Dsub connector tied to one 16 pin dual row connector.
- One dual row 16 pin connector tied to the DSP's serial port.

6.5 RevB User I/O Expansion.

- One external 100 pin half pitch DSUB connector, and one 96 pin DIN socket connector, for interfacing external user defined signals to custom daughter board. Linked only to externally accessible 100 pin half pitch DSUB connector, J1.
- External 100 pin, half pitch (0.050"), Series III DSUB connector, designated J1, for interfacing external user defined signals to P1. AMP part 787169-9, 787170-9, or 787362-9; Thomas & Betts part HFR100RA29CS1.

6.6 RevA and RevB DSP Bus Expansion.

- J3 DIN connector decodes 8Kx32 words, mapped into the DSP's primary bus, Dword boundary address space ranging from 0xFE0000 to 0xFE1FFF.
- J3 DIN connector contains the following DSP signals:
    a) Address: A12-A0.
    b) Data: D31-D0
    c) Control: R/W, STRB, INT2 & INT3, IACK, RDY, H1 & H3.
    d) I/O lines: Serial port 0, XF0 & XF1, TMCK0 & TMCK1.
    e) Host +5Vdc, 3.3Vdc, -5Vdc, +12Vdc, -12Vdc and GND.
- P1, one 12 pin header for JTAG port.

6.7 Software Support

- Win9x/NT/2000/XP and Linux driver support.
- Extensive QuVIEW DSP-resident libraries for LabVIEW, including examples for real time acquisition, signal processing, and control.
- Extensive QuBASE DSP-resident libraries for Visual Basic, including examples for real time acquisition, signal processing, and control.
- Sample code for COFF loaders, PC <-> DSP communications source code and SI-DDK.
- Separately available TI debuggers, C/C++ compilers, assemblers and linkers.

6.8 Physical Dimensions & Electrical Requirements

- Half size PCI-bus card measuring 6.4"(L) x 3.9"(H).
- 7 watts typical with 128Kx32 words SRAM.