PCI-SIO8BXS
User's Manual

EIGHT CHANNEL MULTI-PROTOCOL
SERIAL CONTROLLER
WITH DEEP TRANSMIT AND RECEIVE FIFOS
AND MULTIPROTOCOL TRANSCEIVERS

RS-485
RS-422 / V.11
RS-423 / V.10
RS-232 / V.28
RS530A
V.35

General Standards Corporation
8302A Whitesburg Drive
Huntsville, AL 35802
Phone: (256) 880-8787
Fax: (256) 880-8788
URL: www.generalstandards.com
E-mail: techsupport@generalstandards.com

Revision A

PREFACE

Revision History

1. Rev NR – June 2006 – Original rev from PMC-SIO4BX manual.
2. Rev A – Febn 2007 – SD manual.

Additional copies of this manual or other **General Standards Corporation** literature may be obtained from:

General Standards Corporation
8302A Whitesburg Drive
Huntsville, Alabama 35802
Telephone: (256) 880-8787
Fax: (256) 880-8788
URL: www.generalstandards.com

RELATED PUBLICATIONS

ZILOG Z16C30 USC® User's Manual
ZILOG Z16C30 USC® Product Specifications Databook

ZILOG, Inc.
210 East Hacienda Ave.
Campbell, CA 95008-6600
(408) 370-8000
http://www.zilog.com/

PLX PCI 9080 Data Book

PLX Technology Inc.
390 Potrero Avenue
Sunnyvale, CA 4085
(408) 774-3735
http://www.plxtech.com/

EIA-422-A – Electrical Characteristics of Balanced Voltage Digital Interface Circuits
(EIA order number EIA-RS-422A)

EIA-485 – Standard for Electrical Characteristics of Generators and Receivers for Use in Balanced Digital Multipoint Systems
(EIA order number EIA-RS-485)

EIA Standards and Publications can be purchased from:

GLOBAL ENGINEERING DOCUMENTS
15 Inverness Way East
Englewood, CO 80112
Phone: (800) 854-7179
http://global.ihs.com/

PCI Local Bus Specification Revision 2.1 June 1, 1995.

Copies of PCI specifications available from:
PCI Special Interest Group
NE 2575 Kathryn Street, #17
Hillsboro, OR 97124
http://www.pcisig.com/

TABLE OF CONTENTS

CHAPTER 1: INTRODUCTION

1.0 General Description

The PCI-SI08BX is an eight channel serial interface card which provides high speed, full-duplex, multi-protocol serial capability for PCI applications. The SIO8BX combines four multi-protocol Dual Universal Serial Controllers (USC®), 16 external FIFOs, and multiprotocol transceivers to provide eight fully independent synchronous/asynchronous serial channels. These features, along with a high performance PCI interface engine, give the PCI-SIO8BX unsurpassed performance in a serial interface card.

Features:

- Eight Independent Multi-Protocol Serial Channels
- Synchronous Serial Data Rates up to 10 Mbits/sec
- Asynchronous Serial Data Rates up to 1 Mbit/sec
- Independent Transmit and Receive FIFOs for each Serial Channel – Up to 32k Deep Each
- Serial Mode Protocols include Asynchronous, MonoSync, BiSync, SDLC, and HDLC
- Multiprotocol Transceivers support RS422 (V.11)/RS485, RS423 (V.10), RS232 (V.28), V.35, RS530A, as well as other Mixed Protocol modes.
- Parity and CRC detection capability
- Two On-Board Programmable Oscillators provide increased flexibility for exact Baud Rate Clock generation
- Low Force Helix (LFH) type 160 pin front edge I/O Connector with optional cable adapter to eight DB25 connectors.
- Nine signals per channel, configurable as either DTE or DCE configuration: 3 Serial Clocks, 2 Serial Data signals, Clear-To-Send (CTS), and Ready-To-Send (RTS), DCD, and DTR.
- Unused signals may be reconfigured as general purpose IO.
- Fast RS422/RS485/V.35 Differential Cable Transceivers Provide Data Rate up to 10Mbps
- RS423 and RS232 Cable Transceivers Provide Data Rate up to 230kbps
- Industry Standard Zilog Z16C30 Multi-Protocol Universal Serial Controllers (USC®)
- Dual PCI DMA Engine to speed transfers and minimize host I/O overhead
- A variety of device drivers are available, including VxWorks, WinNT, Win2k, WinXP, Linux, and Labview

1.1 Functional Description

The PCI-SIO8BXS is based on the four channel SIO4BX product line from General Standards Corporation. In order to maintain software compatibility, the PCI-SIO8BXS is implemented as two independent four channel SIO4BX cards. The following diagram shows the PCI-SIO8BX setup.

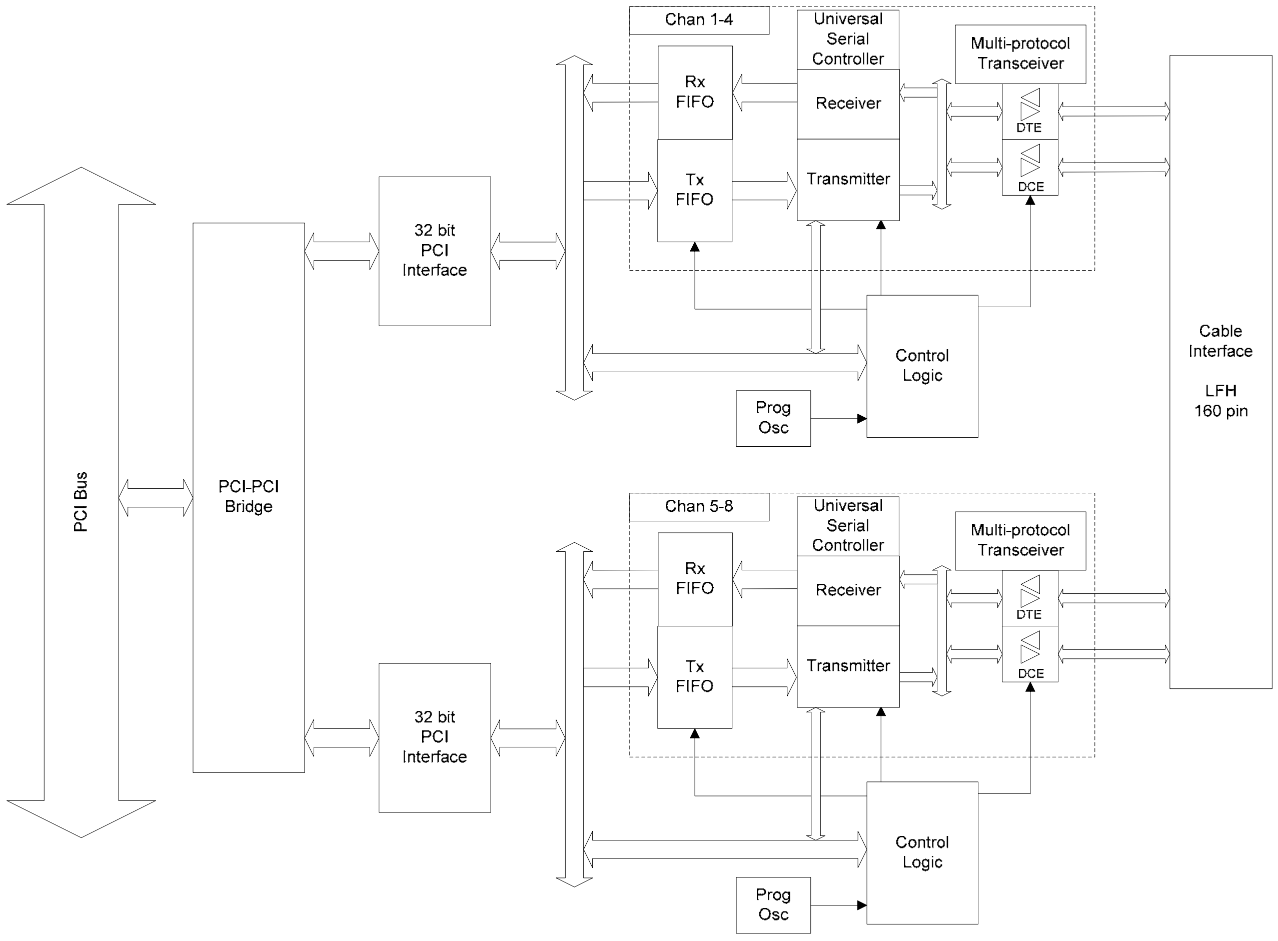

Figure 1-1 Block Diagram of PCI-SIO8BXS

1.1.1 PCI Interface

The PCI interface of the PCI-SIO8BX is implemented using three PCI bridge devices. The PCI-PCI bridge is an industry standard Intel 21152. This PCI-PCI bridge allows the SIO8BX to be implemented as two SIO4BX boards, each with it's own local PCI interface. An industry standard PCI9080 bridge chip from PLX Technology is used to implement PCI Specification 2.1. The PCI9080 provides the 32bit, 33MHz (132MBit/sec) interface between the PCI bus and the Local 32 bit bus.

1.1.2 Local Control Logic

The control functions and glue logic for the board are implemented in an on-board FPGA. This custom logic defines local space registers to provide software control over the board functions. The on-board logic adds many custom features to compliment the Serial Controller chips. These functions include programmable oscillator setup, GPIO functionality, transfer of data between the serial controller chips and the large external FIFOs, and functions to simplify data transfer to/from the FIFOs.

1.1.3 Transmit/Receive FIFOs

Eight independent Transmit and Receive FIFOs provide 512 bytes of data buffering per channel for the serial data. Each channel has a unique transmit and receive FIFO to allow the channels to operate independently. The FIFOs allow data transfer to continue independent of PCI interface transfers and software overhead.

Larger FIFOs (up to 32kbyte deep) can be installed on a custom basis. The required FIFO size may depend on several factors including data transfer size, required throughput rate, and the software overhead (which will also vary based on OS). Deep FIFOs ensure no data is lost for critical systems.

1.1.4 Universal Serial Controllers

Four Zilog Z16C30 Universal Serial Controllers provide the eight serial data channels. The Z16C30 USCs serve as serial/parallel converters which can be software configured to provide a variety of serial protocols. The USCs are highly configurable to allow for a wide range of serial solutions.

1.1.5 Multiprotocol Transceivers

Data is transferred over the user interface using high-speed multiprotocol transceivers. These multiprotocol transceivers can be configured as RS422/RS485, RS423*, RS232, RS530A, or V.35 on a per channel basis. Each channel may also be configured as DTE or DCE configuration.

*In RS423 mode, signals are remapped on the IO connector and only TxC/RxC, TxD/RxD, and RTS/CTS are available.

1.1.7 General Purpose IO

Since some signals may not be used in all applications, the SIO8BX provides the flexibility to remap unused signals to be used as general purpose IO. This also allows signals from unused channels to be available as general purpose IO.

1.1.8 Connector Interface

The PCI-SIO8BXS provides a user IO interface through a front-side card edge connector. All eight serial channels interface through this high-density, 160 pin LFH (Low Force Helix) connector. Signals are grouped at the connector to simplify separating the cable into eight distinct serial connectors.

CHAPTER 2: LOCAL SPACE REGISTERS

2.0 Register Map

The SIO8BXS is accessed through three sets of registers – PCI Registers, USC Registers, and GSC Firmware Registers. The GSC Firmware Registers and USC Registers are referred to as Local Space Registers and are described below. The PCI registers are discussed in Chapter 3.

The Local Space Registers are divided into two distinct functional register blocks – the GSC Firmware Registers and the USC Registers. The GSC Firmware Registers perform the custom board control functions, while the USC Registers map the Zilog Z16C30 registers into local address space. The register block for each USC channel is accessed at a unique address range. The table below shows the address mapping for the local space registers.

Local Address Range	Base Address Offset	Register Block Description
0x0000 – 0x00FF	0x0000	GSC Firmware Registers
0x0100 – 0x013F	0x0100	Channel 1 USC Registers
0x0140 – 0x01FF		Reserved
0x0200 – 0x023F	0x0200	Channel 2 USC Registers
0x0240 – 0x02FF		Reserved
0x0300 – 0x033F	0x0300	Channel 3 USC Registers
0x0340 – 0x03FF		Reserved
0x0400 – 0x043F	0x0400	Channel 4 USC Registers

The GSC Firmware Registers are detailed in Section 2.1. The USC Registers are briefly touched on in Section 2.2 of this manual, but are described in much greater detail in the Zilog Z16C30 Users Manuals.

2.1 GSC Firmware Registers

The GSC Firmware Registers provide the primary control/status for the SIO8BXS board.

Since the board is implemented as two SIO4BX boards, two distinct sets of registers will be present (each with a different PCI base address). Bit D0 of the Board Status Register may be used to determine if each resister set refers to Channel 1-4 or Channel 5-8. The description in this section will refer to Channels 1-4, but it is exactly the same for Channels 5-8.

Offset Address	Size	Access	Register Name	Default Value (Hex)
0x0000	D32	Read/Write	Firmware Revision	C12101XX
0x0004	D32	Read/Write	Board Control	00000000
0x0008	D32	Read Only	Board Status	000000XX
0x000C	---	---	RESERVED	--------
0x0010	D32	Read/Write	Ch 1 Tx Almost Full/Empty	00070007
0x0014	D32	Read/Write	Ch 1 Rx Almost Full/Empty	00070007
0x0018	D32	Read/Write	Ch l 1 Data FIFO	000000XX
0x001C	D32	Read/Write	Ch 1 Control/Status	0000CC00
0x0020	D32	Read/Write	Ch 2 Tx Almost Full/Empty	00070007
0x0024	D32	Read/Write	Ch 2 Rx Almost Full/Empty	00070007
0x0028	D32	Read/Write	Ch 2 Data FIFO	000000XX
0x002C	D32	Read/Write	Ch 2 Control/Status	0000CC00
0x0030	D32	Read/Write	Ch 3 Tx Almost Full/Empty	00070007
0x0034	D32	Read/Write	Ch 3 Rx Almost Full/Empty	00070007
0x0038	D32	Read/Write	Ch 3 Data FIFO	000000XX
0x003C	D32	Read/Write	Ch 3 Control/Status	0000CC00
0x0040	D32	Read/Write	Ch 4 Tx Almost Full/Empty	00070007
0x0044	D32	Read/Write	Ch 4 Rx Almost Full/Empty	00070007
0x0048	D32	Read/Write	Ch 4 Data FIFO	000000XX
0x004C	D32	Read/Write	Ch 4 Control/Status	0000CC00
0x0050	D32	Read/Write	Ch 1 Sync Byte	00000000
0x0054	D32	Read/Write	Ch 2 Sync Byte	00000000
0x0058	D32	Read/Write	Ch 3 Sync Byte	00000000
0x005C	D32	Read/Write	Ch 4 Sync Byte	00000000
0x0060	D32	Read/Write	Interrupt Control	00000000
0x0064	D32	Read/Write	Interrupt Status	00000000
0x0068	D32	Read Only	Interrupt Edge/Level	FFFF7777
0x006C	D32	Read/Write	Interrupt High/Low	FFFFFFFF
0x0070-0x007C	---	---	RESERVED	--------
0x0080	D32	Read/Write	Ch 1Pin Source	00000020
0x0084	D32	Read/Write	Ch 2 Pin Source	00000020
0x0088	D32	Read/Write	Ch 3 Pin Source	00000020
0x008C	D32	Read/Write	Ch 4 Pin Source	00000020
0x0090	D32	Read Only	Ch 1Pin Status	000000XX
0x0094	D32	Read Only	Ch 2 Pin Status	000000XX
0x0098	D32	Read Only	Ch 3 Pin Status	000000XX
0x009C	D32	Read Only	Ch 4 Pin Status	000000XX
0x00A0	D32	Read/Write	Programmable Osc RAM Addr	00000000
0x00A4	D32	Read/Write	Programmable Osc RAM Data1	00000000
0x00A8	D32	Read/Write	Programmable Osc Control/Status	00000000
0x00AC	D32	Read/Write	Programmable Osc RAM Data2	00000000

Offset Address	Size	Access	Register Name	Default Value (Hex)
0x00B0-0x00CC	---	---	RESERVED	--------
0x00D0	D32	Read Only	Ch1 FIFO Count	00000000
0x00D4	D32	Read Only	Ch2 FIFO Count	00000000
0x00D8	D32	Read Only	Ch3 FIFO Count	00000000
0x00DC	D32	Read Only	Ch4 FIFO Count	00000000
0x00E0	D32	Read Only	Ch1 FIFO Size	XXXXXXXX
0x00E4	D32	Read Only	Ch2 FIFO Size	XXXXXXXX
0x00E8	D32	Read Only	Ch3 FIFO Size	XXXXXXXX
0x00EC	D32	Read Only	Ch4 FIFO Size	XXXXXXXX
0x00F0-0x00F4	---	---	RESERVED	--------
0x00F8	D32	Read Only	FW Type Register	01010101
0x00FC	D32	Read Only	Features Register	000000XX

2.1.1 Firmware Revision: Local Offset 0x0000

The Firmware ID register provides version information about the firmware on the board. This is useful for technical support to identify the firmware version.

D31:16	HW Board Rev	C210 => PMC-SIO8BXS Rev NR
D15:8	Firmware Type ID	01 => SIO4B/SIO8B Standard
D7:0	Firmware Revision	Firmware Version

2.1.2 Board Control: Local Offset 0x0004

The Board Control Register defines the general control functions for the board. The main function in this register defines the Demand mode DMA channel requests. For Demand mode DMA, there are only two physical DMA channels which must be shared between the eight serial channels (Rx and Tx for each of four channels). The Demand Mode DMA Channel Request allows the software to multiplex the DMA channels. This is typically handled by the driver – the end user should have no need to change this register.

D31 Board Reset
1 = Reset all Local registers, FIFOs, and USC to their default values
Notes: This bit will automatically clear to 0 following the board reset.
The USCs will need to be reinitialized following a Board Reset.

D30:D9 RESERVED

D8 Rx FIFO Stop on Full
1 = If Rx FIFO becomes full, stop receiving data (disable receiver).

D7 Demand Mode DMA Channel 1 Single Cycle Disable

D6:4 Demand Mode DMA Channel 1 Request

D6	D5	D4	Demand Mode DMA 1 Channel
0	0	0	Channel 1 Rx
1	0	0	Channel 1 Tx
0	1	0	Channel 2 Rx
1	1	0	Channel 2 Tx
0	0	1	Channel 3 Rx
1	0	1	Channel 3 Tx
0	1	1	Channel 4 Rx
1	1	1	Channel 4 Tx

D3	Demand Mode DMA Channel 0 Single Cycle Disable
D2:0	Demand Mode DMA Channel 0 Request

D2	D1	D0	Demand Mode DMA 0 Channel
0	0	0	Channel 1 Rx
1	0	0	Channel 1 Tx
0	1	0	Channel 2 Rx
1	1	0	Channel 2 Tx
0	0	1	Channel 3 Rx
1	0	1	Channel 3 Tx
0	1	1	Channel 4 Rx
1	1	1	Channel 4 Tx

2.1.3 Board Status: Local Offset 0x0008

The Board Status Register gives general overall status for a board.

Since the SIO8BXS is implemented as two SIO4BX boards, the Board ID bits are used to distinguish between the two sets of four channels (Channel 1-4 vs. Channel5-8) as well as determine if multiple SIO8BXS boards are present in a system. Board ID 0 (D0) denotes which 4 channels are accessed by each set of resisters. Jumper J2 is connected to Board ID 4-1. Board ID4-1 will be the same for both Channel 1-4 and Channel 5-8.

D31	Board Reset In Progress
D30:D16	RESERVED
D15	External Ch4 Rx FIFO Not Present
D14	External Ch4 Tx FIFO Not Present
D13	External Ch3 Rx FIFO Not Present
D12	External Ch3 Tx FIFO Not Present
D11	External Ch2 Rx FIFO Not Present
D10	External Ch2 Tx FIFO Not Present
D9	External Ch1 Rx FIFO Not Present
D8	External Ch1 Tx FIFO Not Present
D7:D6	RESERVED
D5	Board ID 4 1 = Jumper J2:7-8 installed
D4	Board ID 3 1 = Jumper J2:5-6 installed
D3	Board ID 2 1 = Jumper J2:3-4 installed
D2	RESERVED
D1	Board ID 1 1 = Jumper J2:1-2 installed
D0	Board ID 0 0 = Channel 1-4 1 = Channel 5-8

2.1.5 Channel Tx Almost Flags: Local Offset 0x0010 / 0x0020 / 0x0030 / 0x0040

The Tx Almost Flag Registers are used to set the Almost Full and Almost Empty Flags for the transmit FIFOs. The FIFO almost flags may be used to determine a fill level for a specific transfer size.

D31:16	Tx Almost Full Flag Value Almost Full Flag will be asserted when the FIFO has space for "Almost Full Value" words or fewer (i.e. FIFO contains (FIFO Size – Almost Full Value) words or more.)
D15:0	Tx Almost Empty Flag Value Almost Empty Flag will be asserted when the FIFO contains "Almost Empty Value" words or fewer.

2.1.6 Channel Rx Almost Flags: Local Offset 0x0014 / 0x0024 / 0x0034 / 0x0044

The Rx Almost Flag Registers are used to set the Almost Full and Almost Empty Flags for the receive FIFOs. The FIFO almost flags may be used to determine a fill level for a specific transfer size.

D31:16	Rx Almost Full Flag Value Almost Full Flag will be asserted when the FIFO has space for "Almost Full Value" words or fewer (i.e. FIFO contains (FIFO Size – Almost Full Value) words or more.)
D15:0	Rx Almost Empty Flag Value Almost Empty Flag will be asserted when the FIFO contains "Almost Empty Value" words or fewer.

2.1.7 Channel FIFO: Local Offset 0x0018 / 0x0028 / 0x0038 / 0x0048

The Channel FIFO Register passes serial data to/from the serial controller chips. The same register is used to access both the Transmit FIFO (writes) and Receive FIFO (reads).

D31:8	RESERVED
D7:0	Channel FIFO Data

2.1.8 Channel Control/Status: Local Offset 0x001C / 0x002C / 0x003C / 0x004C

The Channel Control/Status Register provides the reset functions and data transceiver enable controls, and the FIFO Flag status for each channel.

D31:16 RESERVED

D15:8 Channel Status Bits

D15	Channel Rx FIFO Full Flag Lo	(Active Low -- 0=Rx Full)
D14	Channel Rx FIFO Almost Full Flag Lo	(Active Low -- 0=Rx Almost Full)
D13	Channel Rx FIFO Almost Empty Flag Lo	(Active Low -- 0=Rx Almost Empty)
D12	Channel Rx FIFO Empty Flag Lo	(Active Low -- 0=Rx Empty)
D11	Channel Tx FIFO Full Flag Lo	(Active Low -- 0=Tx Full)
D10	Channel Tx FIFO Almost Full Flag Lo	(Active Low -- 0=Tx Almost Full)
D9	Channel Tx FIFO Almost Empty Flag Lo	(Active Low -- 0=Tx Almost Empty)
D8	Channel Tx FIFO Empty Flag Lo	(Active Low -- 0=Tx Empty)

D7:0 Channel Control Bits

D7 Reset USC (Pulsed)

'1' = Reset USC chip

Notes:

- This value will automatically clear to '0'.
- Following a USC Reset, the next access to the USC must be a write of 0x00 to Local Offset 0x100 (Ch1/2) or Local Offset 0x300 (Ch3/4).
- Since two channels share each USC (Ch1 & Ch2, Ch3 & Ch4), resetting a USC will affect both channels.

D6:2 RESERVED

D1 Reset Channel Rx FIFO (Pulsed)

1 = Reset/Clear Channel Rx FIFOs.

Note: This value will automatically clear to '0'.

D0 Reset Channel Tx FIFO (Pulsed)

1 = Reset/Clear Channel Tx FIFOs.

Note: This value will automatically clear to '0'.

2.1.9 Channel Sync Detect Byte: Local Offset 0x0050 / 0x0054 / 0x0058 / 0x005C

The Sync Detect Byte allows an interrupt to be generated when the received data matches the Sync Detect Byte.

D31:8 RESERVED

D7:0 Channel Sync Detect Byte

If the data being loaded into the Receive FIFO matches this data byte, an interrupt request (Channel Sync Detect IRQ) will be generated. The interrupt source must be enabled in the Interrupt Control Register in order for an interrupt to be generated.

2.1.10 Interrupt Registers

There are 32 on-board interrupt sources (in addition to USC interrupts and PLX interrupts) which may be individually enabled. Four interrupt registers control the on-board interrupts – Interrupt Control, Interrupt Status, Interrupt Edge/Level, and Interrupt Hi/Lo. The 32 Interrupt sources are:

IRQ #	Source	Default Level	Alternate Level
IRQ0	Channel 1Sync Detected	Rising Edge	NONE
IRQ1	Channel 1 Tx FIFO Almost Empty	Rising Edge	Falling Edge
IRQ2	Channel 1 Rx FIFO Almost Full	Rising Edge	Falling Edge
IRQ3	Channel 1 USC Interrupt	Level Hi	NONE
IRQ4	Channel 2 Sync Detected	Rising Edge	NONE
IRQ5	Channel 2 Tx FIFO Almost Empty	Rising Edge	Falling Edge
IRQ6	Channel 2 Rx FIFO Almost Full	Rising Edge	Falling Edge
IRQ7	Channel 2 USC Interrupt	Level Hi	NONE
IRQ8	Channel 3 Sync Detected	Rising Edge	NONE
IRQ9	Channel 3 Tx FIFO Almost Empty	Rising Edge	Falling Edge
IRQ10	Channel 3 Rx FIFO Almost Full	Rising Edge	Falling Edge
IRQ11	Channel 3 USC Interrupt	Level Hi	NONE
IRQ12	Channel 4 Sync Detected	Rising Edge	NONE
IRQ13	Channel 4 Tx FIFO Almost Empty	Rising Edge	Falling Edge
IRQ14	Channel 4 Rx FIFO Almost Full	Rising Edge	Falling Edge
IRQ15	Channel 4 USC Interrupt	Level Hi	NONE
IRQ16	Channel 1 Tx FIFO Empty	Rising Edge	Falling Edge
IRQ17	Channel 1 Tx FIFO Full	Rising Edge	Falling Edge
IRQ18	Channel 1 Rx FIFO Empty	Rising Edge	Falling Edge
IRQ19	Channel 1 Rx FIFO Full	Rising Edge	Falling Edge
IRQ20	Channel 2 Tx FIFO Empty	Rising Edge	Falling Edge
IRQ21	Channel 2 Tx FIFO Full	Rising Edge	Falling Edge
IRQ22	Channel 2 Rx FIFO Empty	Rising Edge	Falling Edge
IRQ23	Channel 2 Rx FIFO Full	Rising Edge	Falling Edge
IRQ24	Channel 3 Tx FIFO Empty	Rising Edge	Falling Edge
IRQ25	Channel 3 Tx FIFO Full	Rising Edge	Falling Edge
IRQ26	Channel 3 Rx FIFO Empty	Rising Edge	Falling Edge
IRQ27	Channel 3 Rx FIFO Full	Rising Edge	Falling Edge
IRQ28	Channel 4 Tx FIFO Empty	Rising Edge	Falling Edge
IRQ29	Channel 4 Tx FIFO Full	Rising Edge	Falling Edge
IRQ30	Channel 4 Rx FIFO Empty	Rising Edge	Falling Edge
IRQ31	Channel 4 Rx FIFO Full	Rising Edge	Falling Edge

For all interrupt registers, the IRQ source (IRQ31:IRQ0) will correspond to the respective data bit (D31:D0) of each register. (D0 = IRQ0, D1 = IRQ1, etc.)

All FIFO interrupts are edge triggered active high. This means that an interrupt will be asserted (assuming it is enabled) when a FIFO Flag transitions from FALSE to TRUE (rising edge triggered) or TRUE to FALSE (falling edge). For example: If Tx FIFO Empty Interrupt is set for Rising Edge Triggered, the interrupt will occur when the FIFO transitions from NOT EMPTY to EMPTY. Likewise, if Tx FIFO Empty Interrupt is set as Falling Edge Triggered, the interrupt will occur when the FIFO transitions from EMPTY to NOT EMPTY.

All Interrupt Sources share a single interrupt request back to the PCI9080 PLX chip. Likewise, all USC interrupt sources share a single interrupt request back to the interrupt controller and must be further qualified in the USC chip. See Section **3.4 Interrupts** for further interrupt programming information.

2.1.10.1 Interrupt Control: Local Offset 0x0060

The Interrupt Control register individually enables each interrupt source. A ‘1’ enables each interrupt source; a ‘0’ disables. An interrupt source must be enabled for an interrupt to be generated.

2.1.10.2 Interrupt Status/Clear: Local Offset 0x0064

The Interrupt Status Register shows the status of each respective interrupt source. If an interrupt source is enabled in the Interrupt Control Register, a ‘1’ in the Interrupt Status Register indicates the respective interrupt has occurred. The interrupt source will remain latched until the interrupt is cleared, either by writing to the Interrupt Status/Clear Register with a ‘1’ in the respective interrupt bit position, or the interrupt is disabled in the Interrupt Control register. If an interrupt source is not asserted or the interrupt is not enabled, writing a ‘1’ to that bit in the Interrupt Status/Clear Register will have no effect on the interrupt.

If the interrupt source is a level triggered interrupt (USC interrupt), the interrupt status may still be ‘1’ even if the interrupt is disabled. This indicates the interrupt condition is true, regardless of whether the interrupt is enabled. Likewise, if a level interrupt is enabled and the interrupt source is true, the interrupt status will be reasserted immediately after clearing the interrupt, and an additional interrupt will be requested.

2.1.10.3 Interrupt Edge/Level & Interrupt Hi/Lo: Local Offset 0x0068 / 0x006C

The Interrupt Edge/Level and Interrupt Hi/Lo Registers define each interrupt source as level hi, level lo, rising edge, or falling edge. All SIO8BXS interrupts are edge triggered except the USC interrupts which are level triggered. Since the interrupt behavior is fixed, the Interrupt Edge/Level register cannot be changed by the user. (Read Only)

The FIFO Flags may be defined as rising edge or falling edge via the Interrupt Hi/Lo Register. For example, a rising edge of the Tx Empty source will generate an interrupt when the Tx FIFO becomes empty. Defining the source as falling edge will trigger an interrupt when the Tx FIFO becomes “NOT Empty”.

2.1.11 Channel Pin Source: Local Offset 0x0080 / 0x0084 / 0x0088 / 0x008C

The Channel Pin Source Register configures the Output source for the Clocks, Data, RTS, and DCD outputs.

31	30	29	28	27	26	25	24
DCE/DTE Mode Enable	Termination Disable	Loopback Enable	DCE/DTE Mode	Transceiver Protocol Mode			

23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Loop Int	DTR Source		TxD Source		TxAuxC Source		DCD Source		RTS Source		USC_DCD Direction		USC_CTS Direction		TxC Source			USC_RxC Source			USC_TxC Source		

Pin Source Register

D31 Transceiver Enable
Setting this bit enables the DCE/DTE buffer control (D28) control and Loopback controls (D29 and D23).

D30 Termination Disable
For RS422/RS485, RS530, and V.35, the RxC, RxD, RxAuxC, and DCD have built in termination at the transceivers. These internal terminations may be disabled to allow external terminations (or no terminations) to be used. Setting this bit will disable the internal transceiver termination resistors.

D29 External Loopback Mode
When DCE/DTE Mode is enabled (Bit D31=1), this bit will automatically loopback the TxC/RxC, TxD/RxD, and RTS/CTS signals at the cable (transceivers enabled). This allows the transceivers to be tested in a standalone mode.
Notes:
- The DCE/DTE mode will select the set of signals (DCE or DTE) to be looped back
- In RS423 mode, TxC/RxC and RTS/CTS are not looped back externally (due to hardware design constraints), but they will still be looped back internally
- Since the transceivers will be enabled in this mode, all external cables should be disconnected to prevent interference from external sources.

D28 DCE/DTE Mode
When DCE/DTE Mode is enabled (Bit D31=1), this bit set the mode to DCE (1) or DTE (0). DCE/DTE mode changes the direction of the signals at the IO Connector.

D27:24 Transceiver Protocol Mode

D27	D26	D25	D24	Transceiver Mode
0	0	0	0	RS-422 / RS-485
0	0	0	1	RS-423
0	0	1	0	RS-232
0	0	1	1	RESERVED
0	1	0	0	RS530 Mode (RS-422 / RS-423)
0	1	0	1	RS530A Mode (RS-422 / RS-423)
0	1	1	0	V.35 Mode (V.35 / RS-232)
0	1	1	1	RESERVED
1	X	X	X	RESERVED

D23 Internal Loopback Mode
When DCE/DTE Mode is enabled (Bit D31=1), this bit will automatically loopback the TxC/RxC, TxD/RxD, and RTS/CTS signals internal to the board.

D22:21 Cable DTR/DSR Control

D22	D21	DTR Source
0	0	Tristate
0	1	Input Only (DSR)
1	0	Output ‘0’
1	1	Output ‘1’

D20:19 Cable TxD Output Control
Allows TxD output to be used as a general purpose output.

D20	D19	TxD Source
0	X	USC_TxD
1	0	Output ‘0’
1	1	Output ‘1’

D18:17 Cable TxAuxC Output Control
Defines the Clock Source for the TxAuxC signal to the IO connector.

D18	D17	TxD Source
0	0	Tristate
0	1	On-board Programmable Clock
1	0	Output ‘0’
1	1	Output ‘1’

D16:15 Cable DCD Output Source

D16	D15	Output Source	Notes
0	0	USC_DCD Output	USC_DCD field (D12:D11) must equal ‘11’
0	1	RTS Output	Rx FIFO Almost Full
1	0	‘0’	Drive low
1	1	‘1’	Drive Hi

D14:13 Cable RTS Output Source

D14	D13	Output Source	Notes
0	0	USC_CTS Output	USC_CTS field (D10:D9) must equal ‘11’
0	1	RTS Output	Rx FIFO Almost Full
1	0	‘0’	Drive low
1	1	‘1’	Drive Hi

D12:11 USC_DCD Direction Setup

Defines the DCD direction for the USC DCD pin.

Notes:

- If DCD is used as GPIO, set this field to ‘00’ and set Pin Source Register D16:D15 for output / Pin Status Register D3 for input.
- If set, the DCD direction must agree with the USC DCD setup (USC IOCR D13:12) to ensure proper operation.
- If field set to ‘11’ (Output), DCD Source field (D16:15) must be set to ‘00’.

D12	D11	DCD Buffer Direction	USC IOCR D13:D12 Setup
0	0	Buffer Disabled	XX (Don’t Care)
0	1	Input from IO Connector - DCD	0X (Input)
1	0	Reserved	XX (Don’t Care)
1	1	Output to IO Connector	1X (Output)

D10:9 USC_CTS Direction Setup

Defines the CTS direction for the USC CTS pin.

Notes:

- If CTS is used as GPIO, set this field to ‘00’ and set Pin Source Register D14:D13 for output / Pin Status Register D2 for input.
- If set, the CTS direction must agree with the USC CTS setup (USC IOCR D15:14) to ensure proper operation.
- If field set to ‘11’ (Output), RTS Source field (D14:13) must be set to ‘00’.

D10	D9	CTS Buffer Direction	USC IOCR D15:D14 Setup
0	0	Tristate	XX (Don’t Care)
0	1	Input from IO Connector – CTS	0X (Input)
1	0	Reserved	XX (Don’t Care)
1	1	Output to IO Connector	1X (Output)

D8:6 Cable TxC Source

Defines the Clock Source for the TxC signal to the IO connector.

D8	D7	D6	TxC Source
0	0	0	Prog Clock
0	0	1	Inverted Prog Clock
0	1	0	‘0’ (Drive Line Lo)
0	1	1	‘1’ (Drive Line Hi)
1	0	0	USC_TxC
1	0	1	USC_RxC
1	1	0	Cable RxC Input
1	1	1	Cable RxAuxC Input

D5:3 USC_RxC Source

Defines the Clock Source for the USC_RxC pin. The clock source must agree with the USC Clock setup (USC I/O Control Reg D5:3) to ensure the signal is not being driven by both the USC and the FPGA.

D5	D4	D3	USC_RxC Source	USC IOCR D2:D0 Setup
0	0	0	Prog Clock	000 (Input)
0	0	1	Inverted Prog Clock	000 (Input)
0	1	0	'0'	000 (Input)
0	1	1	'1'	000 (Input)
1	0	0	Cable RxC Input	000 (Input)
1	0	1	Cable RxAuxC Input	000 (Input)
1	1	0	RESERVED	--------
1	1	1	Driven from USC	IOCR D2:D0 != 000 (Output)

D2:0 USC_TxC Source

Defines the Clock Source for the USC_TxC pin. Since this signal is bidirectional (it may be used as either an input or output to the USC), the clock source must agree with the USC Clock setup (USC IO Control Reg D2:0) to ensure the signal is not being driven by both the USC and the FPGA.

D2	D1	D0	USC_TxC Source	USC IOCR D5:D3 Setup
0	0	0	Prog Clock	000 (Input)
0	0	1	Inverted Prog Clock	000 (Input)
0	1	0	'0'	000 (Input)
0	1	1	'1'	000 (Input)
1	0	0	Cable RxC Input	000 (Input)
1	0	1	Cable RxAuxC Input	000 (Input)
1	1	0	RESERVED	--------
1	1	1	Driven from USC	IOCR D5:D3 != 000 (Output)

2.1.12 Channel Pin Status: Local Offset 0x0090 / 0x0094 / 0x0098 / 0x009C

Unused inputs may be utilized as general purpose input signals. The Channel Pin Status Register allows the input state of all the IO pins to be monitored. Output signals as well as inputs are included to aid in debug operation.

D31:D12	RESERVED
D11	DTR Output
D10	DSR Input
D9	TxAuxC Output
D8	RxAuxC Input
D7	DCD Output
D6	RTS Output
D5	TxD Output
D4	TxC Output
D3	DCD Input
D2	CTS Input
D1	RxD Input
D0	RxC Input

2.1.13 Programmable Clock Registers: Local Offset 0x00A0 / 0x00A4 / 0x00A8 / 0xAC

The Programmable Clock Registers allow the user to program the on-board programmable oscillator and configure the channel clock post-dividers. As GSC should provide software routines to program the clock, the user should have no need to access these registers. See section 3.6 for more information.

2.1.14 FIFO Count Register: Local Offset 0x00D0 / 0x00D4 / 0x00D8 / 0x00DC

The FIFO Count Registers display the current number of words in each FIFO. This value, along with the FIFO Size Registers, may be used to determine the amount of data which can be safely transferred without over-running (or under-running) the FIFOs.

D31:D16	Number of words in Rx FIFO
D15:D0	Number of words in Tx FIFO

2.1.15 FIFO Size Register: Local Offset 0x00E0 / 0x00E4 / 0x00E8 / 0x00EC

The FIFO Size Registers display the sizes of the installed data FIFOs. This value is calculated at power-up This value, along with the FIFO Count Registers, may be used to determine the amount of data which can be safely transferred without over-running (or under-running) the FIFOs.

D31:D16	Size of installed Rx FIFO
D15:D0	Size of installed Tx FIFO

2.1.16 FW Type ID Register: Local Offset 0x00F8

This register allows boards to be designed with different functionality on each channel. For example, a board could contain two Standard SIO channels (with Z16C30), and two Raw Synchronous channels. Each byte corresponds to a channel. This register is read only – it reflects the implemented logic.

D31:D24	Channel 4 FW Type – 01 = Standard
D23:D16	Channel 3 FW Type – 01 = Standard
D15:D8	Channel 2 FW Type – 01 = Standard
D7:D0	Channel 1 FW Type – 01 = Standard

2.1.17 Features Register: Local Offset 0x00FC

The Features Register allows software to account for added features in the firmware versions. Bits will be assigned as new features are added.

D31:17	RESERVED
D16	FW Type Register Present
D15:8	Features Rev Level 01 – RS232 support, update Pin Source 02 – BX support 03 – Common Internal/External FIFO code 04 – Latched FIFO Overrun/Underrun for test 05 – Demand mode DMA Single Cycle for Tx 06 – Single Cycle DMA disable, update Pin Source TxAuxC 07 – Reset Status, revised FIFO Overrun/Underrun status
D7	Demand Mode DMA Single Cycle Disable feature implemented
D6	Board Reset feature implemented
D5	FIFO Counters/Size implemented
D4	'1'
D3:0	Programmable Clock Configuration 0x4 = Two CY22393 - 4 Oscillators (SIO8 configuration)

2.2 Universal Serial Controller Registers

The internal registers of the Zilog Z16C30 Universal Serial Controller (USC) are memory mapped into Local Address space. It is beyond the scope of this manual to provide comprehensive USC programming information. For detailed programming information, please refer to the Zilog High Speed Communication Controller Product Specifications Databook for the Z16C30 and the Zilog Z16C30USC User's Manual. These manuals may be obtained directly from Zilog (www.zilog.com), or copies of these manuals may be downloaded from the General Standards website (www.generalstandards.com).

Some specific setup information may be needed for a driver to interface to the USC. Typically, the driver will handle the hardware specific characteristics and the end user will only need to be concerned with the driver interface - the following hardware setup information may be safely ignored. If you aren't sure if you need this information, you probably don't.

2.2.1 USC Reset

The four serial channels are implemented in two Z16C30 Universal Serial Controllers – Channels 1 and 2 share one USC, and Channels 3 and 4 share the other. This implementation is important to realize since resetting a Z16C30 chip will have an effect on two serial channels. Since the USC chips are typically reset upon initialization, this means a "Reset USC" for Channel 1 will also "Reset USC" for Channel 2. In addition to making the second reset redundant and unnecessary, a Reset USC on one channel may inadvertently adversely affect normal operation on the second channel. Therefore, care must be exercised when resetting a USC (USC Reset bit in the Board Control Register), especially in multithreaded environments.

Since the USC Reset physically resets the USC, the first access to the USC following the reset must reinitialize the BCR in the USC. To complete the Reset process, the user should write data 0x00 to USC base address offset 0x100 or 0x300 to correctly initialize the BCR. Following this initial byte write, the USC may be accessed normally.

Due to the ability for a USC Reset to affect two channels, it is recommended that a single USC Channel be Reset via the RTReset bit of the USC Channel Command/Address Register (CACR).

2.2.2 8-Bit USC Register Access

As the USC has a configurable bus interface, the USC must be set to match the 8-bit non-multiplex interface implementation of the SIO8BXS. This setup information must be programmed into the USC Bus Configuration Register (BCR) upon initial power up and following every hardware reset of the USC. The BCR is accessible only following a USC hardware reset – the first write to the USC following a USC Reset programs the BCR. Even though the Zilog manual states the BCR has no specific address, the driver must use the channel USC base address – 0x100 for Ch 1 & Ch 2, 0x300 for Ch 3 & Ch 4 – as the BCR address. Failure to do so may result in improper setup. Since the user interface to the USC is an 8 bit interface, the software only needs to set the lower byte to 0x00 (hardware implementation will program the upper byte of the BCR).

2.2.3 USC Data Transfer

Although the Z16C30 USC contains 32 byte internal FIFOs for data transfer, these are typically not used on the SIO8BXS. Since the SIO8BXS has much deeper external FIFOs (or internal FPGA FIFOs), the internal USC FIFOs are setup to immediately transfer data to/from the external FIFOs. Immediate transfer of received data to the external FIFOs eliminates the possibility of data becoming "stuck" in the USC internal receive FIFOs, while bypassing the USC internal transmit FIFOs ensures better control of the transmit data.

In order to automatically transfer data to and from the external FIFOs, the USC should use DMA to request a data transfer whenever one byte is available in the USC internal FIFOs. This “DMA” should not be confused with the DMA of data from the SIO8BXS external FIFOs to the PCI interface. To accomplish the USC-to-External FIFO DMA transfer, the TxReq / RxReq pins should be set as DMA Requests in the IOCR, and the TxAck / RxAck pins should be set as DMA Acknowledge inputs in the HCR. In addition, the Tx Request Level should be set to 0x1F (31) using TCSR/TICR and the Rx Request Level should be set to 0 using RCSR/RICR. See Z16C30 manual for further details on programming the DMA request levels.

2.2.4 USC Register Memory Map

To access the USC in 8-bit mode, the driver is required to access the upper and lower bytes of each register independently. The odd address byte will access the upper byte of each register (D15-D8), and the even address byte will access the lower byte (D7-D0). Each USC register must be accessed independently as a byte access– the software cannot perform word or long word accesses to the USC registers.

The USC register map is provided below. The Channel Offset Address depicted is from the Channel Base Address – (Ch 1 Base Address = 0x100, Ch 2 Base Address = 0x200, Ch 3 Base Address = 0x300, Ch 4 Base Address = 0x400). For further programming details, please refer to the Zilog Z16C30 data books.

Channel Offset Address	**Access***	**Register Name**
0x01 / 0x00	CCAR Hi / Lo	Channel Command / Address Register
0x03 / 0x02	CMR Hi / Lo	Channel Mode Register
0x05 / 0x04	CCSR Hi / Lo	Channel Command / Status Register
0x07 / 0x06	CCR Hi / Lo	Channel Control Register
0x11 / 0x10	CMCR Hi / Lo	Clock Mode Control Register
0x13 / 0x12	HCR Hi / Lo	Hardware Configuration Register
0x17 / 0x16	IOCR Hi / Lo	I/O Control Register
0x19 / 0x18	ICR Hi / Lo	Interrupt Control Register
0x1D / 0x1C	MISR Hi / Lo	Miscellaneous Interrupt Status Register
0x1F / 0x1E	SICR Hi / Lo	Status Interrupt Control Register
0x20	RDR	Receive Data Register
0x23 / 0x22	RMR	Receive Mode Register
0x25 / 0x24	RCSR Hi / Lo	Receive Command / Status Register
0x27 / 0x26	RICR Hi / Lo	Receive Interrupt Control Register
0x29 / 0x28	RSR Hi / Lo	Receive Sync Register
0x2B / 0x2A	RCLR Hi / Lo	Receive Count Limit Register
0x2D / 0x2C	RCCR Hi / Lo	Receive Character Count Register
0x2F / 0x2E	TC0R	Time Constant 0 Register
0x30	TDR	Transmit Data Register
0x33 / 0x32	RMR	Transmit Mode Register
0x35 / 0x34	TCSR Hi / Lo	Transmit Command / Status Register
0x37 / 0x36	TICR Hi / Lo	Transmit Interrupt Control Register
0x39 / 0x38	TSR Hi / Lo	Transmit Sync Register
0x3B / 0x3A	TCLR Hi / Lo	Transmit Count Limit Register
0x3D / 0x3C	TCCR Hi / Lo	Transmit Character Count Register
0x3F / 0x3E	TC1R	Time Constant 1 Register

CHAPTER 3: PROGRAMMING

3.0 Introduction

This section addresses common programming questions when developing an application for the SIO4/SIO8. General Standards has developed software libraries to simplify application development.   These libraries handle many of the low-level issues described below, including Resets, FIFO programming, and DMA. These libraries may default the board to a "standard" configuration (one used by most applications), but still provide low-level access so applications may be customized. The following sections describe the hardware setup in detail for common programming issues.

3.1 Board Identification

The PCI-SIO8BXS is implemented as two SIO4BX boards – one for Channels 1-4 and the other for Channels 5-8. Multiple boards may also be present in a system. The Board Status Register provide a means to determine which physical board as well as whether the upper or lower set of channels is being accessed.

Bit D0 of the Board Status Register denotes Channel 1-4 (D0=0) or Channels 5-8 (D0=1). A four position jumper block J2 can be used as a physical board identifier. These bits are depicted in D5:D3, and D1 of the Board Status Register. The bit will be '1' if the jumper is installed.

3.2 Resets

Each serial channel provides control for three unique reset sources: a USC Reset, a Transmit FIFO Reset, and a Receive FIFO Reset. All three resets are controlled from the GSC Channel Control/Status Registers. In addition, a Board Reset has been implemented in the Board Control Register. This board reset will reset all local registers to their default state as well as reset all FIFOs and USCs (all channels will be reset).

Section 2.2.1 provides information on the USC Reset. It is important to realize that since each Zilog Z16C30 chip contains two serial channels, a USC Reset to either channel will reset the entire chip (both channels affected).   Due to the limitation of a USC Reset to affecting two channels, it is recommended that a single USC Channel be Reset via the RTReset bit of the USC Channel Command/Address Register (CCAR).

The FIFO resets allow each individual FIFO (Tx and Rx) to be reset independently. Setting the FIFO reset bit will clear the FIFO immediately.

3.3 FIFO Almost Flags

The FIFO Almost Empty and Almost Full flags of the SIO8BXS provide a way for the user to approximate the amount of data in the FIFO. Since FIFO Count Registers are available to provide the exact number of words in each FIFO, the FIFO Almost Flags are not needed in most applications. If RTS functionality is used (Section 3.9), the Rx Almost Full Flag is used to set the RTS disable level. The FIFO Almost Flags may also be useful to provide an interrupt at a specific FIFO fill level.

Each channel provides two 32 bit registers for setting the Almost Full/Empty values: the Tx FIFO Almost Register (See Section 2.1.5) and the Rx FIFO Almost Register (See Section 2.1.6). Each of these registers is further divided into two 16 bit words: D31-D16 = Almost Full Value; D15-D0 = Almost Empty Value.

The Almost Flag value represents the number of bytes from each respective "end" of the FIFO. The Almost Empty value represents the number of bytes from empty, and the Almost Full value represents the number of bytes from full (NOT the number of bytes from empty). For example, the default value of "0x0007 0007" in the FIFO Almost Register means that the Almost Empty Flag will indicate when the FIFO holds 0x0007 bytes or fewer, and will transition as the 8th byte is read or written. The Almost Full Flag indicates the FIFO contains (FIFO Size – 0x7) bytes or more. For the standard 32Kbyte FIFO, an Almost Full value of 0x7 will cause the Almost Full flag to be asserted when the FIFO contains 32761 (32k – 7) or more bytes of data .

3.4 PCI DMA

The PCI DMA functionality allows data to be transferred between host memory and the SIO8BXS onboard FIFOs with the least amount of CPU overhead. The PCI9080 bridge chip handles all PCI DMA functions, and the device driver should handle the details of the DMA transfer. (Note: DMA refers to the transfer of Data from the on-board FIFOs over the PCI bus. This should not be confused with the DMA mode of the USC – transfer of data between the USC and the on-board FIFOs. This On-Board DMA is setup by the driver and should always be enabled).

There are two PCI DMA modes – Demand Mode DMA and Non-Demand Mode DMA. Demand Mode DMA refers to data being transferred on demand. For receive, this means data will be transferred as soon as it is received into the FIFO. Likewise, for transmit, data will be transferred to the FIFOs as long as the FIFO is not full. The disadvantage to Demand Mode DMA is that the DMA transfers are dependent on the user data interface. If the user data transfer is incomplete, the Demand mode DMA transfer will also stop. If a timeout occurs, there is no way to determine the exact amount of data transferred before it was aborted.

Non-Demand Mode DMA does not check the FIFO empty/full flags before or during the data transfer – it simply assumes there is enough available FIFO space to complete the transfer. If the transfer size is larger than the available data, the transfer will complete with invalid results. This is the preferred mode for DMA operation. The FIFO Counters may be used to determine how much space is available for DMA so that the FIFO will never over/under run. Demand Mode DMA requires less software control, but runs the risk of losing data due to an incomplete transfer. The GSC Windows API uses this method (Non-Demand DMA and checking the FIFO counters) as the standard transfer method.

3.4 Interrupts

The SIO8BXS has a number of interrupt sources which are passed to the host CPU via the PCI Interrupt A. Since there is only one physical interrupt source, the interrupts pass through a number of “levels” to get multiplexed onto this single interrupt. The interrupt originates in the PCI9080 PCI Bridge, which combines the internal PLX interrupt sources (DMA) with the local space interrupt. The driver will typically take care of setting up and handling the PCI9080 interrupts. The single Local Interrupt is made up of the interrupt sources described in Section 2.1.10. In addition, the Zilog USC contains a number of interrupt sources which are combined into a single Local Interrupt. The user should be aware that interrupts must be enabled at each level for an interrupt to occur. For example, if a USC interrupt is used, it must be setup and enabled in the USC, enabled in the GSC Firmware Interrupt Control Register, and enabled in the PCI9080. In addition, the interrupt must be acknowledged and/or cleared at each level following the interrupt.

3.5 Clock Setup

Figure 3-1 shows the relationship of the various clock sources on the SIO8BXS board. These clock sources can be most simply viewed in three sections: On-Board Programmable Clocks, IO connector Clocks, and USC Clocks.

The Programmable Clocks consist of a single on-board programmable oscillator and four post divide clocks (one for each channel). The single programmable oscillator clock is used as the input for each of the programmable clock post dividers, which will allow each channel to have a unique programmable clock input. These programmable clocks are further described in sections 2.1.12 and 3.6.

The IO Connector Clocks consist of the cable RxC and cable TxC for each channel, and an Auxiliary Clock signal (RxAuxC/TxAuxC) which may be configured as either input or output.   The RxC is always an input and may be used as a clock source for either the cable TxC or the USC Clocks. The cable TxC is always an output configured by the Pin Source register. The Auxiliary clock may be used as an output or input clock signal, or as a general purpose IO, configured by the Pin Source register. See Section 2.1.11 for further information on the Pin Source register.

The USC Clocks (USC_RxC and USC_TxC) are bidirectional signals. Even though the names of these clocks seem to imply a receive clock and a transmit clock, both clocks are bidirectional, fully programmable, and identical in function – either clock may be used for transmit or receive. The USC clocks may be sourced from either the USC or the FPGA (via the Pin Source register). The user must be careful to ensure that both the USC and Pin Source Register are setup to agree. If a USC clock is set as an output in the USC, it should be programmed as an input in the Pin Source register. Likewise, if a USC clock source is driven from the Pin Source register, the user should program the pin as an input to the USC. Section 2.1.11 describes the Pin Source Registers.

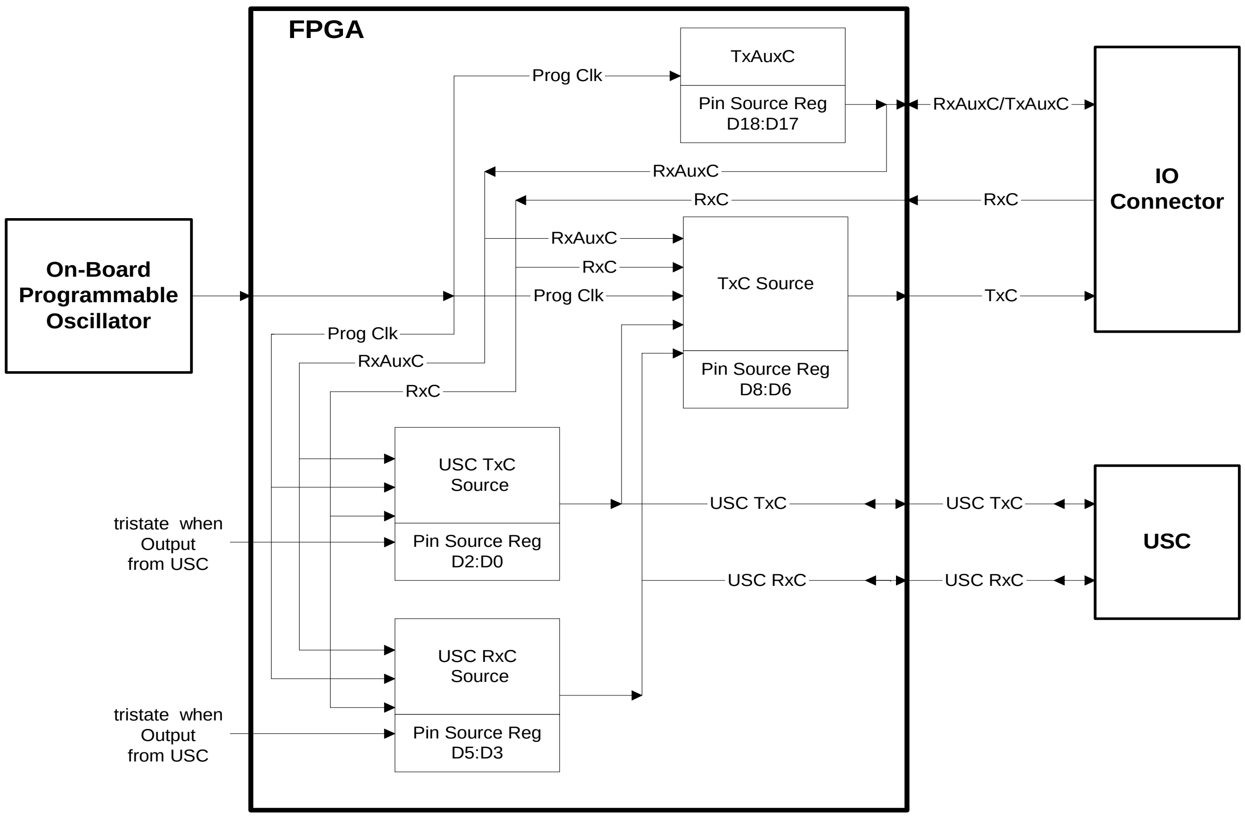

Figure 4-1 – Clock Configuration

The TxAuxC / RxAuxC clock is a single auxiliary input or output clock. As an output, this clock can be set to the programmable clock frequency as a reference clock. As an input, this clock can be used as the clock source for the USC clocks (USC_RxC and USC_TxC) or the Cable TxC clock. If the signal is set as an output (TxAuxC), the RxAuxC input is simply the feedback of the TxAuxC. If the TxAuxC source is set to “Tristate”, the AuxC signal will be input only.

In asynchronous mode, the clock does not need to be transmitted with the data.   Therefore, the USC Clock pins will be used for the input baud rate clock.   Since the USC_RxC and USC_TxC pins have identical functions, the USC_RxC and USC_TxC pins may be used interchangeably. The async baud rate clock will be 16x / 32x / or 64x the actual baud rate due to the async oversampling. This oversample rate is set in the USC Channel Mode Register when async mode is selected. The simplest method will be to program the channel programmable clock to be 16/32/64 times the desired baudrate and use this clock as the source for the USC_TxC / USC_RxC pin. Section 2.1.11 describes how to program the Pin Source Register to set USC_TxC or USC_RxC = Programmable Clock. The USC should be programmed such that USC_TxC / USC_RxC is an input (in the USC I/O Control Register), and the USC baudrate generator will be bypassed completely. If both Rx and Tx are operating at the same baud rate, the same USC clock pin can be used for both the transmit and receive clocks.

For synchronous modes, the clock is transmitted and received on the cable along with the data. This can present a problem since the USC only has two clock pins. Since one clock is necessary for receive clock and the other is necessary for the transmit clock, there is no clock pin available for an input to the USC baud rate generators. The on-board programmable clocks provide a solution for this situation. By using the programmable oscillator and the programmable clock post-divider, the on-board programmable clock can usually be set directly to the desired transmit baud rate. The USC_TxC pin and the Cable TxC are both set equal to the Programmable Clock in the Pin Source Register. The USC_RxC pin is used for the receive clock from the cable interface, so it will be set to the cable RxC in the Pin Source Register. Since the FPGA will source both USC clocks, they must be programmed as inputs in the USC I/O Control Register.

The preceding suggestions should work for most applications. The default Pin Source Register value should set the clocks to work with both scenarios – USC_TxC pin = Programmable Clock, USC_RxC Pin = Cable RxC, Cable TxC = Programmable Clock. (For async, use USC_TxC is input clock).

3.6 Programmable Oscillator / Programmable Clocks

Two On-Board Programmable Oscillator provides each channel with a unique programmable clock source using aCypress Semiconductor CY22393 Programmable Clock generators. In order to program the oscillator, it is necessary to calculate and program values for different clock frequencies. General Standards has developed routines to calculate the necessary values for a given setup and program the clock generator. As these routines are written in C on a windows based PC, they may need to be ported for user specific applications. Contact GSC for help in porting these routines.

The default clock configuration at power-up for the programmable clock on all channels is 20MHz.

See Appendix A for more detailed information concerning programming the on-board clock frequencies.

3.7 Multiprotocol Transceiver Control

The SIO8BXS has multiprotocol transceivers which allow RS422/RS485, RS423, RS232, RS530, RS530A, and V.35 modes. The Mode is set by the Protocol Mode filed in the Pin Source Register. The following table shows the signal protocol for each mode.

Mode	TxC	RxC	AuxC	TxD	RxD	RTS	CTS	DCD	DTR/DSR
RS-422/RS-485	RS-422	RS-422	RS-422	RS-422	RS-422	RS-422	RS-422	RS-422	RS-422
RS-423 *	RS-423	RS-423	<NA>	RS-423	RS-423	RS-423	RS-423	<NA>	<NA>
RS-232	RS-232	RS-232	RS-232	RS-232	RS-232	RS-232	RS-232	RS-232	RS-232
RS-530	RS-422	RS-422	RS-422	RS-422	RS-422	RS-422	RS-422	RS-423	RS-422
RS-530A	RS-422	RS-422	RS-422	RS-422	RS-422	RS-422	RS-422	RS-423	RS-423
V.35	V.35	V.35	V.35	V.35	V.35	RS-232	RS-232	RS-232	RS-232

* RS-423 remaps the TxC/RxC and TxD/RxD signals on the user connector. See Connector pinout.

3.8 DCE/DTE Mode

As all signals are bidirectional, the DCE or DTE mode will set the direction for each signal. For the transceivers to be configured as either DTE or DCE, set the DCE/DTE Enable bit in the Pin Source register (D31). The following table gives the input/output configuration for each signal: The DCD, AuxC, and DTR/DSR direction is set in the Pin Source register fields, independent of DCE/DTE mode.

Signal	DTE	DCE
TxC	TxC Out	RxC In
RxC	RxC In	TxC Out
TxD	TxD Out	RxD In
RxD	RxD In	TxD Out
RTS	RTS Out	CTS In
CTS	CTS In	RTS Out
DCD	Direction controlled by Pin Source Reg D16:15	
AuxC	Direction controlled by Pin Source Reg D18:17	
DTR/DSR	Direction controlled by Pin Source Reg D22:21	

3.9 General Purpose IO

Unused signals at the cable may be used for general purpose IO. The Pin Source and Pin Status Registers provide for simple IO control of all the cable interface signals. For outputs, the value is set using the appropriate field in the Pin Source Register. All inputs can be read via the Pin Status register.

The direction of the DTE/DCE signals (RxD, TxD, RxC, TxC, CTS, RTS) will still be controlled by the DTE / DCE mode control. For example: In DTE mode, DTE_TxC, DTE_TxD, and DTE_RTS may only be used as general purpose outputs, and DTE_RxC, DTE_RxD, and DTE_CTS may only be used as general purpose inputs.

CHAPTER 4: PCI INTERFACE

4.0 PCI Interface Registers

A PCI9080 I/O Accelerator from PLX Technology handles the PCI Interface. The PCI interface is compliant with the 5V, 33MHz 32-bit PCI Specification 2.1. The PCI9080 provides dual DMA controllers for fast data transfers to and from the on-board FIFOs. Fast DMA burst accesses provide for a maximum burst throughput of 132MB/s to the PCI interface. To reduce CPU overhead during DMA transfers, the controller also implements Chained (Scatter/Gather) DMA, as well as Demand Mode DMA.

Since many features of the PCI9080 are not utilized in this design, it is beyond the scope of this document to duplicate the PCI9080 User's Manual. Only those features, which will clarify areas specific to the PCI-X are detailed here. Please refer to the PCI9080 User's Manual (See Related Publications) for more detailed information. Note that the BIOS configuration and software driver will handle most of the PCI9080 interface. Unless the user is writing a device driver, the details of this PCI Interface Chapter may be skipped.

4.1 PCI Registers

The PLX 9080 contains many registers, many of which have no effect on the SIO8BXS performance. The following section attempts to filter the information from the PCI9080 manual to provide the necessary information for a SIO4BX/SIO8BXS specific driver.

The SIO8BXS uses an on-board serial EEPROM to initialize many of the PCI9080 registers after a PCI Reset. This allows board specific information to be preconfigured.

4.1.1 PCI Configuration Registers

The PCI Configuration Registers allow the PCI controller to identify and control the cards in a system.

PCI device identification is provided by the Vendor ID/Device ID (Addr 0x0000) and Sub-Vendor ID/Sub-Device ID Registers (0x002C). The following definitions are unique to the General Standards SIO4/SIO8 boards. All drivers should verify the ID/Sub-ID information before attaching to this card. These values are fixed via the Serial EEPROM load following a PCI Reset, and cannot be changed by software.

Vendor ID	0x10B5	PLX Technology
Device ID	0x9080	PCI9080
Sub-Vendor ID	0x10B5	PLX Technology
Sub-Device ID	0x2401	GSC SIO4 / SIO8B

The configuration registers also setup the PCI IO and Memory mapping for the SIO8BXS. The PCI9080 is setup to use PCIBAR0 and PCIBAR1 to map the internal PLX registers into PCI Memory and IO space respectively. PCIBAR2 will map the Local Space Registers into PCI memory space, and PCIBAR3 is unused. Typically, the OS will configure the PCI configuration space.

For further information of the PCI configuration registers, please consult the PLX Technology PCI9080 Manual.

4.1.2 Local Configuration Registers

The Local Configuration registers give information on the Local side implementation. These include the required memory size. The SIO8BXS memory size is initialized to 4k Bytes. All other Local Registers initialize to the default values described in the PCI9080 Manual.

4.1.3 Runtime Registers

The Runtime registers consist of mailbox registers, doorbell registers, and a general-purpose control register. The mailbox and doorbell registers are not used and serve no purpose on the SIO8BXS. All other Runtime Registers initialize to the default values described in the PCI9080 Manual.

4.1.4 DMA Registers

The Local DMA registers are used to setup the DMA transfers to and from the on-board FIFOs. DMA is supported only to the four FIFO locations. The SIO8BXS supports both Demand (DREQ# controlled) and Non-Demand mode DMA. Both Channel 0 and Channel 1 DMA are supported.

4.1.4.1 DMA Channel Mode Register: (PCI 0x80 / 0x94)

The DMA Channel Mode register must be setup to match the hardware implementation

Bit	Description	Value	Notes
D1:0	Local Bus Width	11 = 32 bit 00 = 8 bit	Although the serial FIFOs only contain 8 bits of data, the register access is still a 32bit access. It is possible to “pack” the data by setting the Local Bus Width to 8, but this is only guaranteed to work with Non-Demand Mode DMA
D5:2	Internal Wait States	0000 = Unused	
D6	Ready Input Enable	1 = Enabled	
D7	Bterm# Input Enabled	0 = Unused	
D8	Local Burst Enable	1 = Supported	Bursting allows fast back-to-back accesses to the FIFOs to speed throughput
D9	Chaining Enable (Scatter Gather DMA)	X	DMA source addr, destination addr, and byte count are loaded from memory in PCI Space.
D10	Done Interrupt Enable	X	DMA Done Interrupt
D11	Local Addressing Mode	1 = No Increment	DMA to/from FIFOs only
D12	Demand Mode Enable	X	Demand Mode DMA is supported for FIFO accesses on the SIO8BXS. (See Section 3.3)
D13	Write & Invalidate Mode	X	
D14	DMA EOT Enable	0 = Unused	
D15	DMA Stop Data Transfer Enable	0 = BLAST terminates DMA	
D16	DMA Clear Count Mode	0 = Unused	
D17	DMA Channel Interrupt Select	X	
D31:18	Reserved	0	

CHAPTER 5: HARDWARE CONFIGURATION

5.0 Board Layout

The following figure is a drawing of the physical components of the PCI-SIO8BXS:

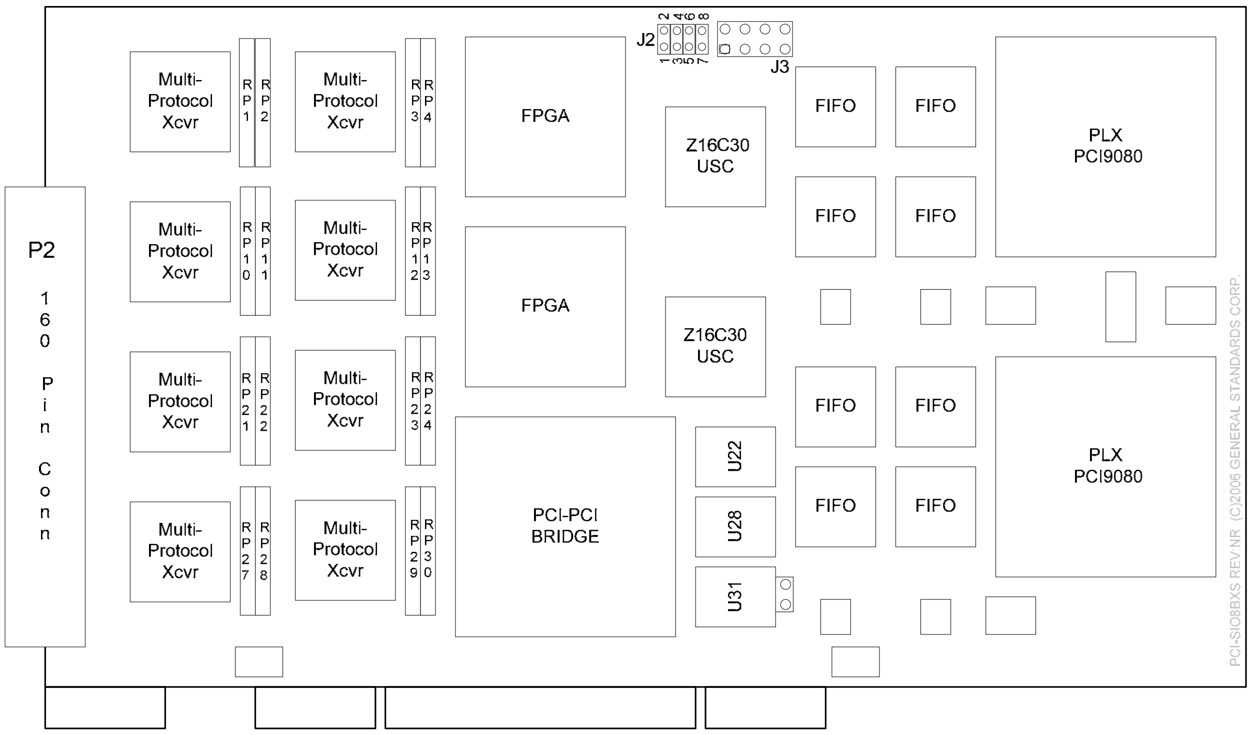

Figure 5-1: Board Layout

5.1 Board ID Jumper J2

Jumper J2 allows the user to set the Board ID in the GSC Board Status Register (See Section 2.1.3). This is useful to uniquely identify a board if more than one SIO8BXS card is in a system. When the Board ID jumper is installed, it will read ‘1’ in the Board Status Register. The Board Status Register bit will report ‘0’ when the jumper is removed.   Refer to Figure 5.1-1 for Jumper J2 location.

J2 Jumper	Description	Notes
1 - 2	Board ID 1	Board ID 1 in Board Status Register (D1)
3 - 4	Board ID 2	Board ID 2 in Board Status Register (D3)
5 - 6	Board ID 3	Board ID 3 in Board Status Register (D4)
7 - 8	Board ID 4	Board ID 4 in Board Status Register (D5)

5.2 Interface Connectors

The user interface connector on the SIO8BX is a 160-pin LFH connector (female) mounted to the front edge of the board (P2). The part number for this 160-pin LFH connector is Molex 51-24-1040. The mating cable connector is Molex 51-25-1040 or equivalent. The tables below show the pinout for the differential modes RS485/RS422/V.35 (Table 5-1), RS232 Mode (Table 5-2), and RS423 (Table 5-3) . Mixed signal modes (RS530/RS530A) will follow Table 5-1, but the single ended signals (RS-423 and RS-232) will use only the negative side of the differential pair.

RS422 / RS485 / V.35 Pinout					
Pin #	DTE Signal	DCE Signal	**Pin #**	DTE Signal	DCE Signal
1	Ch1 TxC +	Ch1 RxC +	80	Ch1 TxD +	Ch1 RxD +
2	Ch1 TxC -	Ch1 RxC -	79	Ch1 TxD -	Ch1 RxD -
3	Ch1 RxC +	Ch1 TxC +	78	Ch1 RxD +	Ch1 TxD +
4	Ch1 RxC -	Ch1 TxC -	77	Ch1 RxD -	Ch1 TxD -
5	Ch1 AuxC +		76	Ch1 DCD +	
6	Ch1 AuxC -		75	Ch1 DCD -	
7	Ch1 DTR/DSR +		74	Ch1 CTS +	Ch1 RTS +
8	Ch1 DTR/DSR -		73	Ch1 CTS -	Ch1 RTS -
9	Ch1 RTS +	Ch1 CTS +	72	SGND1	
10	Ch1 RTS -	Ch1 CTS -	71	UNUSED	
11	Ch2 TxC +	Ch2 RxC +	70	UNUSED	
12	Ch2 TxC -	Ch2 RxC -	69	SGND2	
13	Ch2 RxC +	Ch2 TxC +	68	Ch2 TxD +	Ch2 RxD +
14	Ch2 RxC -	Ch2 TxC -	67	Ch2 TxD -	Ch2 RxD -
15	Ch2 AuxC +		66	Ch2 RxD +	Ch2 TxD +
16	Ch2 AuxC -		65	Ch2 RxD -	Ch2 TxD -
17	Ch2 DTR/DSR +		64	Ch2 DCD +	
18	Ch2 DTR/DSR -		63	Ch2 DCD -	
19	Ch2 RTS +	Ch2 CTS +	62	Ch2 CTS +	Ch2 RTS +
20	Ch2 RTS -	Ch2 CTS -	61	Ch2 CTS -	Ch2 RTS -
21	Ch5 TxC +	Ch5 RxC +	60	Ch5 TxD +	Ch5 RxD +
22	Ch5 TxC -	Ch5 RxC -	59	Ch5 TxD -	Ch5 RxD -
23	Ch5 RxC +	Ch5 TxC +	58	Ch5 RxD +	Ch5 TxD +
24	Ch5 RxC -	Ch5 TxC -	57	Ch5 RxD -	Ch5 TxD -
25	Ch5 AuxC +		56	Ch5 DCD +	
26	Ch5 AuxC -		55	Ch5 DCD -	
27	Ch5 DTR/DSR +		54	Ch5 CTS +	Ch5 RTS +
28	Ch5 DTR/DSR -		53	Ch5 CTS -	Ch5 RTS -
29	Ch5 RTS +	Ch5 CTS +	52	SGND5	
30	Ch5 RTS -	Ch5 CTS -	51	UNUSED	
31	Ch6 TxC +	Ch6 RxC +	50	UNUSED	
32	Ch6 TxC -	Ch6 RxC -	49	SGND6	
33	Ch6 RxC +	Ch6 TxC +	48	Ch6 TxD +	Ch6 RxD +
34	Ch6 RxC -	Ch6 TxC -	47	Ch6 TxD -	Ch6 RxD -
35	Ch6 AuxC +		46	Ch6 RxD +	Ch6 RxD +
36	Ch6 AuxC -		45	Ch6 RxD -	Ch6 RxD -
37	Ch6 DTR/DSR +		44	Ch6 DCD +	
38	Ch6 DTR/DSR -		43	Ch6 DCD -	
39	Ch6 RTS +	Ch6 CTS +	42	Ch6 CTS +	Ch6 RTS +
40	Ch6 RTS -	Ch6 CTS -	41	Ch6 CTS -	Ch6 RTS -

RS422 / RS485 / V.35 Pinout (Continued)					
Pin #	DTE Signal	DCE Signal	**Pin #**	DTE Signal	DCE Signal
81	Ch3 TxC +	Ch3 RxC +	160	Ch3 TxD +	Ch3 RxD +
82	Ch3 TxC -	Ch3 RxC -	159	Ch3 TxD -	Ch3 RxD -
83	Ch3 RxC +	Ch3 TxC +	158	Ch3 RxD +	Ch3 TxD +
84	Ch3 RxC -	Ch3 TxC -	157	Ch3 RxD -	Ch3 TxD -
85	Ch3 AuxC +		156	Ch3 DCD +	
86	Ch3 AuxC -		155	Ch3 DCD -	
87	Ch3 DTR/DSR +		154	Ch3 CTS +	Ch3 RTS +
88	Ch3 DTR/DSR -		153	Ch3 CTS -	Ch3 RTS -
89	Ch3 RTS +	Ch3 CTS +	152	SGND3	
90	Ch3 RTS -	Ch3 CTS -	151	UNUSED	
91	Ch4 TxC +	Ch4 RxC +	150	UNUSED	
92	Ch4 TxC -	Ch4 RxC -	149	SGND4	
93	Ch4 RxC +	Ch4 TxC +	148	Ch4 TxD +	Ch4 RxD +
94	Ch4 RxC -	Ch4 TxC -	147	Ch4 TxD -	Ch4 RxD -
95	Ch4 AuxC +		146	Ch4 RxD +	Ch4 TxD +
96	Ch4 AuxC -		145	Ch4 RxD -	Ch4 TxD -
97	Ch4 DTR/DSR +		144	Ch4 DCD +	
98	Ch4 DTR/DSR -		143	Ch4 DCD -	
99	Ch4 RTS +	Ch4 CTS +	142	Ch4 CTS +	Ch4 RTS +
100	Ch4 RTS -	Ch4 CTS -	141	Ch4 CTS -	Ch4 RTS -
101	Ch7 TxC +	Ch7 RxC +	140	Ch7 TxD +	Ch7 RxD +
102	Ch7 TxC -	Ch7 RxC -	139	Ch7 TxD -	Ch7 RxD -
103	Ch7 RxC +	Ch7 TxC +	138	Ch7 RxD +	Ch7 TxD +
104	Ch7 RxC -	Ch7 TxC -	137	Ch7 RxD -	Ch7 TxD -
105	Ch7 AuxC +		136	Ch7 DCD +	
106	Ch7 AuxC -		135	Ch7 DCD -	
107	Ch7 DTR/DSR +		134	Ch7 CTS +	Ch7 RTS +
108	Ch7 DTR/DSR -		133	Ch7 CTS -	Ch7 RTS -
109	Ch7 RTS +	Ch7 CTS +	132	SGND7	
110	Ch7 RTS -	Ch7 CTS -	131	UNUSED	
111	Ch8 TxC +	Ch8 RxC +	130	UNUSED	
112	Ch8 TxC -	Ch8 RxC -	129	SGND8	
113	Ch8 RxC +	Ch8 TxC +	128	Ch8 TxD +	Ch8 RxD +
114	Ch8 RxC -	Ch8 TxC -	127	Ch8 TxD -	Ch8 RxD -
115	Ch8 AuxC +		126	Ch8 RxD +	Ch8 RxD +
116	Ch8 AuxC -		125	Ch8 RxD -	Ch8 RxD -
117	Ch8 DTR/DSR +		124	Ch8 DCD +	
118	Ch8 DTR/DSR -		123	Ch8 DCD -	
119	Ch8 RTS +	Ch8 CTS +	122	Ch8 CTS +	Ch8 RTS +
120	Ch8 RTS -	Ch8 CTS -	121	Ch8 CTS -	Ch8 RTS -

Table 5-1: RS485/RS422/V.35 Cable Pin-Out

The Single ended signals use only the negative side of the differential pair signals. Table 5-2 below shows the resulting RS-232 pinout.

RS232 Pinout

Pin #	DTE Signal	DCE Signal	Pin #	DTE Signal	DCE Signal
1	RESERVED		80	RESERVED	
2	Ch1 TxC	Ch1 RxC	79	Ch1 TxD	Ch1 RxD
3	RESERVED		78	RESERVED	
4	Ch1 RxC	Ch1 TxC	77	Ch1 RxD	Ch1 TxD
5	RESERVED		76	RESERVED	
6	Ch1 AuxC		75	Ch1 DCD	
7	RESERVED		74	RESERVED	
8	Ch1 DTR/DSR		73	Ch1 CTS	Ch1 RTS
9	RESERVED		72	SGND1	
10	Ch1 RTS	Ch1 CTS	71	RESERVED	
11	RESERVED		70	RESERVED	
12	Ch2 TxC	Ch2 RxC	69	SGND2	
13	RESERVED		68	RESERVED	
14	Ch2 RxC	Ch2 TxC	67	Ch2 TxD	Ch2 RxD
15	RESERVED		66	RESERVED	
16	Ch2 AuxC		65	Ch2 RxD	Ch2 TxD
17	RESERVED		64	RESERVED	
18	Ch2 DTR/DSR		63	Ch2 DCD	
19	RESERVED		62	RESERVED	
20	Ch2 RTS	Ch2 CTS	61	Ch2 CTS	Ch2 RTS
21	RESERVED		60	RESERVED	
22	Ch5 TxC	Ch5 RxC	59	Ch5 TxD	Ch5 RxD
23	RESERVED		58	RESERVED	
24	Ch5 RxC	Ch5 TxC	57	Ch5 RxD	Ch5 TxD
25	RESERVED		56	RESERVED	
26	Ch5 AuxC		55	Ch5 DCD	
27	RESERVED		54	RESERVED	
28	Ch5 DTR/DSR		53	Ch5 CTS	Ch5 RTS
29	RESERVED		52	SGND5	
30	Ch5 RTS	Ch5 CTS	51	RESERVED	
31	RESERVED		50	RESERVED	
32	Ch6 TxC	Ch6 RxC	49	SGND6	
33	RESERVED		48	RESERVED	
34	Ch6 RxC	Ch6 TxC	47	Ch6 TxD	Ch6 RxD
35	RESERVED		46	RESERVED	
36	Ch6 AuxC		45	Ch6 RxD	Ch6 RxD
37	RESERVED		44	RESERVED	
38	Ch6 DTR/DSR		43	Ch6 DCD	
39	RESERVED		42	RESERVED	
40	Ch6 RTS	Ch6 CTS	41	Ch6 CTS	Ch6 RTS

RS232 Pinout (Continued)					
Pin #	DTE Signal	DCE Signal	Pin #	DTE Signal	DCE Signal
81	RESERVED		160	RESERVED	
82	Ch3 TxC	Ch3 RxC	159	Ch3 TxD	Ch3 RxD
83	RESERVED		158	RESERVED	
84	Ch3 RxC	Ch3 TxC	157	Ch3 RxD	Ch3 TxD
85	RESERVED		156	RESERVED	
86	Ch3 AuxC		155	Ch3 DCD	
87	RESERVED		154	RESERVED	
88	Ch3 DTR/DSR		153	Ch3 CTS	Ch3 RTS
89	RESERVED		152	SGND3	
90	Ch3 RTS	Ch3 CTS	151	RESERVED	
91	RESERVED		150	RESERVED	
92	Ch4 TxC	Ch4 RxC	149	SGND4	
93	RESERVED		148	RESERVED	
94	Ch4 RxC	Ch4 TxC	147	Ch4 TxD	Ch4 RxD
95	RESERVED		146	RESERVED	
96	Ch4 AuxC		145	Ch4 RxD	Ch4 TxD
97	RESERVED		144	RESERVED	
98	Ch4 DTR/DSR		143	Ch4 DCD	
99	RESERVED		142	RESERVED	
100	Ch4 RTS	Ch4 CTS	141	Ch4 CTS	Ch4 RTS
101	RESERVED		140	RESERVED	
102	Ch7 TxC	Ch7 RxC	139	Ch7 TxD	Ch7 RxD
103	RESERVED		138	RESERVED	
104	Ch7 RxC	Ch7 TxC	137	Ch7 RxD	Ch7 TxD
105	RESERVED		136	RESERVED	
106	Ch7 AuxC		135	Ch7 DCD	
107	RESERVED		134	RESERVED	
108	Ch7 DTR/DSR		133	Ch7 CTS	Ch7 RTS
109	RESERVED		132	SGND7	
110	Ch7 RTS	Ch7 CTS	131	RESERVED	
111	RESERVED		130	RESERVED	
112	Ch8 TxC	Ch8 RxC	129	SGND8	
113	RESERVED		128	RESERVED	
114	Ch8 RxC	Ch8 TxC	127	Ch8 TxD	Ch8 RxD
115	RESERVED		126	RESERVED	
116	Ch8 AuxC		125	Ch8 RxD	Ch8 RxD
117	RESERVED		124	RESERVED	
118	Ch8 DTR/DSR		123	Ch8 DCD	
119	RESERVED		122	RESERVED	
120	Ch8 RTS	Ch8 CTS	121	Ch8 CTS	Ch8 RTS

Table 5-2: RS-232 Cable Pin-Out

The multiprotocol receivers only have certain signals with RS-423 capability. Therefore, the TxC/RxC and TxD/RxD are remapped to the DTR/DSR and DCD signals , and the DTR/DSR, AuxC, and DCD signals are unavailable in RS423 mode. Table 5-3 below shows the RS-423 mode pinout.

RS423 Pinout					
Pin #	DTE Signal	DCE Signal	**Pin #**	DTE Signal	DCE Signal
1	RESERVED		80	RESERVED	
2	RESERVED		79	RESERVED	
3	RESERVED		78	RESERVED	
4	RESERVED		77	RESERVED	
5	RESERVED		76	Ch1 TxD	Ch1 RxD
6	RESERVED		75	Ch1 RxD	Ch1 TxD
7	Ch1 TxC	Ch1 RxC	74	RESERVED	
8	Ch1 RxC	Ch1 TxC	73	Ch1 CTS	Ch1 RTS
9	RESERVED		72	SGND1	
10	Ch1 RTS	Ch1 CTS	71	RESERVED	
11	RESERVED		70	RESERVED	
12	RESERVED		69	SGND2	
13	RESERVED		68	RESERVED	
14	RESERVED		67	RESERVED	
15	RESERVED		66	RESERVED	
16	RESERVED		65	RESERVED	
17	Ch2 TxC	Ch2 RxC	64	Ch2 TxD	Ch2 RxD
18	Ch2 RxC	Ch2 TxC	63	Ch2 RxD	Ch2 TxD
19	RESERVED		62	UNUSED	
20	Ch2 RTS	Ch2 CTS	61	Ch2 CTS	Ch2 RTS
21	RESERVED		60	RESERVED	
22	RESERVED		59	RESERVED	
23	RESERVED		58	RESERVED	
24	RESERVED		57	RESERVED	
25	RESERVED		56	Ch5 TxD	Ch5 RxD
26	RESERVED		55	Ch5 RxD	Ch5 TxD
27	Ch5 TxC	Ch5 RxC	54	RESERVED	
28	Ch5 RxC	Ch5 TxC	53	Ch5 CTS	Ch5 RTS
29	RESERVED		52	SGND5	
30	Ch5 RTS	Ch5 CTS	51	RESERVED	
31	RESERVED		50	RESERVED	
32	RESERVED		49	SGND6	
33	RESERVED		48	RESERVED	
34	RESERVED		47	RESERVED	
35	RESERVED		46	RESERVED	
36	RESERVED		45	RESERVED	
37	Ch6 TxC	Ch6 RxC	44	Ch6 TxD	Ch6 RxD
38	Ch6 RxC	Ch6 TxC	43	Ch6 RxD	Ch6 TxD
39	RESERVED		42	RESERVED	
40	Ch6 RTS	Ch6 CTS	41	Ch6 CTS	Ch6 RTS

RS423 Pinout (Continued)

Pin #	DTE Signal	DCE Signal	Pin #	DTE Signal	DCE Signal
81	RESERVED		160	RESERVED	
82	RESERVED		159	RESERVED	
83	RESERVED		158	RESERVED	
84	RESERVED		157	RESERVED	
85	RESERVED		156	Ch3 TxD	Ch3 RxD
86	RESERVED		155	Ch3 RxD	Ch3 TxD
87	Ch3 TxC	Ch3 RxC	154	RESERVED	
88	Ch3 RxC	Ch3 TxC	153	Ch3 CTS	Ch3 RTS
89	RESERVED		152	SGND3	
90	Ch3 RTS		151	RESERVED	
91	RESERVED		150	RESERVED	
92	RESERVED		149	SGND2	
93	RESERVED		148	RESERVED	
94	RESERVED		147	RESERVED	
95	RESERVED		146	RESERVED	
96	RESERVED		145	RESERVED	
97	Ch4 TxC	Ch4 RxC	144	Ch4 TxD	Ch4 RxD
98	Ch4 RxC	Ch4 TxC	143	Ch4 RxD	Ch4 TxD
99	RESERVED		142	UNUSED	
100	Ch4 RTS	Ch4 CTS	141	Ch4 CTS	Ch4 RTS
101	RESERVED		140	RESERVED	
102	RESERVED		139	RESERVED	
103	RESERVED		138	RESERVED	
104	RESERVED		137	RESERVED	
105	RESERVED		136	Ch7 TxD	Ch7 RxD
106	RESERVED		135	Ch7 RxD	Ch7 TxD
107	Ch7 TxC	Ch7 RxC	134	RESERVED	
108	Ch7 RxC	Ch7 TxC	133	Ch7 CTS	Ch7 RTS
109	RESERVED		132	SGND5	
110	Ch7 RTS	Ch7 CTS	131	RESERVED	
111	RESERVED		130	RESERVED	
112	RESERVED		129	SGND6	
113	RESERVED		128	RESERVED	
114	RESERVED		127	RESERVED	
115	RESERVED		126	RESERVED	
116	RESERVED		125	RESERVED	
117	Ch8 TxC	Ch8 RxC	124	Ch8 TxD	Ch8 RxD
118	Ch8 RxC	Ch8 TxC	123	Ch8 RxD	Ch8 TxD
119	RESERVED		122	RESERVED	
120	Ch8 RTS	Ch8 CTS	121	Ch8 CTS	Ch8 RTS

Table 5-3: RS-423 Cable Pin-Out

5.3 Termination Resistors

The SIO8BXS transceivers have built in termination resistors of for RS-422 and V.35 modes. The built in RS-422 termination is a 120 Ohm parallel termination only on the high speed receiver signals – RxC, RxD, RxAuxC, and DCD. The built in V.35 termination is a Y network of 51/124 Ohms. If desired, the internal termination resistors may be disabled by setting bit D30 in the Pin Source Register.

The board is designed with socketed external parallel termination (if a different value than the internal termination is required). The external termination resistors are 8 pin SIPs. There are 16 termination SIPs – RP1-RP4, RP10-RP13, RP21-RP24, and RP27-RP30. The external parallel resistors are for RS422/RS485 termination only – no provision is made for external V.35 termination resistors.Refer to Figure 5-1 for resistor pack locations.

Please contact quotes@generalstandards.com if a different termination value is required.

CHAPTER 6: ORDERING OPTIONS

6.0 Ordering Information

The SIO8BXS can accept FIFOs with depths ranging from 512 bytes to 32k bytes. Larger FIFO depth is important for faster interfaces to reduce the risk of data loss due to software overhead. The PCI-SIO8BXS can be ordered with the following FIFO depths: 512 bytes, 8kbytes, or 32kbytes. Note that the FIFO size option in the board part number refers to the total FIFO size for all 8 channels, not the FIFO size of a single FIFO. For example, PCI-SIO8BXS-64K would contain eight 8k deep FIFOs. Please consult our sales department for pricing and availability. . Please consult our sales department with your application requirements to determine the correct ordering option. (quotes@generalstandards.com).

6.1 Interface Cable

General Standards Corporation can provide an interface cable for the SIO8BXS board. This standard cable is a twisted pair cable for increased noise immunity. Several standard cable lengths are offered, or the cable length can be custom ordered to the user's needs. Versions of the cable are available with connectors on both ends, or the cable may be ordered with a single connector to allow the user to adapt the other end for a specific application.   A standard cable is available which will breakout the serial channels into eight DB25 connectors. Shielded cable options are also available. Please consult our sales department for more information on cabling options and pricing.

6.2 Device Drivers

General Standards has developed many device drivers for The SIO8BXS boards, including VxWorks, Windows, Linux, and LabView. As new drivers are always being added, please consult our website (www.generalstandards.com) or consult our sales department for a complete list of available drivers and pricing.

6.3 Custom Applications

Although the SIO8BXS board provides extensive flexibility to accommodate most user applications, a user application may require modifications to conform to a specialized user interface. General Standards Corporation has worked with many customers to provide customized versions based on the SIO8BXS boards. Please consult our sales department with your specifications to inquire about a custom application.

APPENDIX A: PROGRAMMABLE OSCILLATOR PROGRAMMING

The 4 on-baord clock frequencies are supplies via two Cypress Semiconductor CY22393 Programmable Clock Generatosr.   In order to change the clock frequencies, this chip must be reprogrammed. This document supplies the information necessary to reprogram the on-board clock frequencies. GSC has developed routines to calculate and program the on-board oscillator for a given set of frequencies, so it should not be necessary for the user need the following information – it is provided for documentation purposes. Please contact GSC for help in setting up the on-board oscillator.

The CY22393 contains several internal address which contain the programming information. GSC has mirrored this data internal to the FPGA (CLOCK RAM) to allow the user to simply setup the data in the FPGA RAM and then command the on-board logic to program the clock chip. This isolates the user from the hardware serial interface to the chip. For detailed CY22393 programming details, please refer to the Cypress Semiconductor CY22393 dat sheet.

For the SIO8BX, a second programmable oscillator has been added to assure that each channel has a dedicated PLL. (The SIO4BX used 3 PLLs in a single CY22393 to generate all four clocks). To implement this, a second CLOCK RAM block was added. CLOCK RAM1 programs the first CY22393 (using CLKA=Ch1_Clk, CLKB=Ch2_Clk, CLKC=Ch3_Clk), and CLOCK_RAM2 programs the second CY22393 (using CLKD=Ch4_Clk). Since the original SIO4BX (with a single CY22393) used CLKD for Ch4_Clk, the same code can be made to support both schemes by simply programming CLKD of the first CY22393.

Each CLOCK RAM block is accessed through 2 registers – Address Offset at local offset 0x00A0 and Data at local ffset at 0x00A4 (CLOCK RAM1) or 0x00AC (CLOCK RAM2). The user simply sets the RAM Address register to the appropriate offset, then reads or writes the the RAM data. The Programmable Osc Control/Status register allows the user to program the CY22393 or setup the clock post-dividers.

The GSC Local Programmable Clock Registers are defined as follows:

0x00A0 – RAM Address Register
Defines the internal CLOCK RAM address to read/write

0x00A4 – RAM Data1 Register
Provides access to the CLOCK RAM1 pointed to by the RAM Addr Register.

0x00AC – RAM Data2 Register
Provides access to the CLOCK RAM2 pointed to by the RAM Addr Register.

0x00A8 – Programmable Osc Control/Status Register
Provides control to write the contents of the CLOCK RAM to the CY22393 and setup additional post-dividers for the input clocks.

Control Word (Write Only)

D0	Program Oscillator 1 = Program contents of CLOCK RAM to CY22393. Automatically resets to 0.
D1	Measure Channel 1 Clock
D2	Measure Channel 2 Clock
D3	Measure Channel 3 Clock
D4	Measure Channel 4 Clock
D5	Reserved (Unused)
D6	Status Word Readback Control

	0 => Status Word D31-D8 == Measured Channel Value
	1 => Status Word D31-D8 == Control Word D23-D0
D7	Post-divider set
	0 = Ignore D23-D8 during Command Word Write
	1 = Set Channel Post-Dividers from D23-D8 during Command Word Write
D11-D8	Channel 1 Post-Divider
D15-D12	Channel 2 Post-Divider
D19-D16	Channel 3 Post-Divider
D23-D20	Channel 4 Post-Divider
D31-D24	Reserved (Unused)

Status Word (Read Only)

D0	Program Oscillator Done
	0 = Oscillator Programming in progress.
D1	Program Oscillator Error
	1 = Oscillator Programming Error has occurred.
D2	Clock Measurement complete.
	0 = Clock Measurement in progress.
D7-D3	Reserved (Unused)
D31-D8	If Command Word D6 = 0,
	Measured Channel Clock Value
	If Command Word D6 = 1,
	Control Word D23-D0

Channel Clock Post-Dividers:

The Control Word defines 4 fields for Channel Clock Post-dividers. These post-dividers will further divide down the input clock from the programmable oscillator to provide for slow baud rates. Each 4 bit field will allow a post divider of 2^n. For example, if the post-divider value=0, the input clock is not post-divided. A value of 2 will provide a post-divide of 4 (2^2). This will allow for a post-divide value of up to 32768 (2^15) for each input clock.
Bit D7 of the Control word qualifies writes to the post-divide registers. This allows other bits in the command register to be set while the post-divide values are maintained.

Channel Clock Measurement:

The Control Word defines 4 bits which will select one of the 4 channel clocks (input clock + post-divide) for a measurement. This will allow the user feedback as to whether the programmable oscillator was programmed correctly. To measure a clock, select the clock to measure in the Control word, and also clear Bit D6 to allow for readback of the result. Read back the Status Word until D2 is set. Status Word D31-D8 should contain a value representing 1/10 the measured clock frequency (Value * 10 = Measured Frequency in MHz). Keep in mind that this value will not be exactly the programmed frequency due to the 100ppm (0.01%) accuracy of the on-board reference.

The Internal RAM is defined as follows: RAM Address 0x08–0x57 correspond directly to the CY22393 registers.

Address	Description	Default Value
0x00 – 0x05	Reserved (Unused)	0x00
0x06	Reserved	0xD2
0x07	Reserved	0x08
0x08	ClkA Divisor (Setup0)	0x01
0x09	ClkA Divisor (Setup1)	0x01
0x0A	ClkB Divisor (Setup0)	0x01
0x0B	ClkB Divisor (Setup1)	0x01
0x0C	ClkC Divisor	0x01
0x0D	ClkD Divisor	0x01
0x0E	Source Select	0x00
0x0F	Bank Select	0x50
0x10	Drive Setting	0x55
0x11	PLL2 Q	0x00
0x12	PLL2 P Lo	0x00
0x13	PLL2 Enable/PLL2 P Hi	0x00
0x14	PLL3 Q	0x00
0x15	PLL3 P Lo	0x00
0x16	PLL3 Enable/PLL3 P Hi	0x00
0x17	OSC Setting	0x00
0x18	Reserved	0x00
0x19	Reserved	0x00
0x1A	Reserved	0xE9
0x1B	Reserved	0x08
0x1C-0x3F	Reserved (Unused)	0x00
0x40	PLL1 Q (Setup0)	0x00
0x41	PLL1 P Lo 0 (Setup0)	0x00
0x41	PLL1 Enable/PLL1 P Hi (Setup0)	0x00
0x43	PLL1 Q (Setup1)	0x00
0x44	PLL1 P Lo 0 (Setup1)	0x00
0x45	PLL1 Enable/PLL1 P Hi (Setup1)	0x00
0x46	PLL1 Q (Setup2)	0x00
0x47	PLL1 P Lo 0 (Setup2)	0x00
0x48	PLL1 Enable/PLL1 P Hi (Setup2)	0x00
0x49	PLL1 Q (Setup3)	0x00
0x4A	PLL1 P Lo 0 (Setup3)	0x00
0x4B	PLL1 Enable/PLL1 P Hi (Setup3)	0x00
0x4C	PLL1 Q (Setup4)	0x00
0x4D	PLL1 P Lo 0 (Setup4)	0x00
0x4E	PLL1 Enable/PLL1 P Hi (Setup4)	0x00
0x4F	PLL1 Q (Setup5)	0x00
0x50	PLL1 P Lo 0 (Setup5)	0x00
0x51	PLL1 Enable/PLL1 P Hi (Setup5)	0x00
0x52	PLL1 Q (Setup6)	0x00
0x53	PLL1 P Lo 0 (Setup6)	0x00
0x54	PLL1 Enable/PLL1 P Hi (Setup6)	0x00
0x55	PLL1 Q (Setup7)	0x00
0x56	PLL1 P Lo 0 (Setup7)	0x00
0x57	PLL1 Enable/PLL1 P Hi (Setup7)	0x00
0x58-0xFF	Reserved (Unused)	0x00