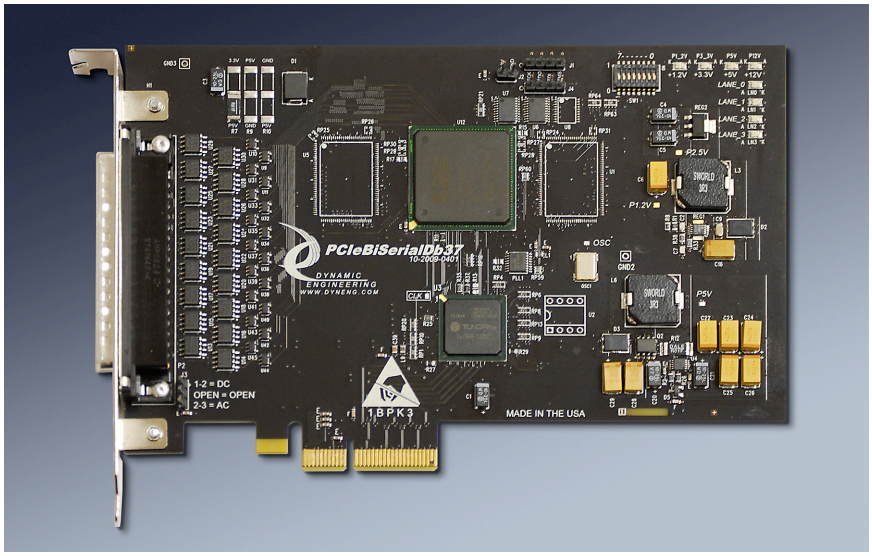
DYNAMIC ENGINEERING
150 DuBois St. Suite C, Santa Cruz, CA 95060
831-457-8891 Fax 831-457-4793
http://www.dyneng.com
sales@dyneng.com
Est. 1988

User Manual

PCIeBiSerialDb37-RTN8

Byte Wide Receive Port
Byte Wide Transmit port
PCIe 4 lane Module
LVDS



Revision A2
Corresponding Hardware: Revision 1
10-2009-0401
FLASH 0201

PCIeBiSerialDb37Rtn8

ARC-210 and Digital Parallel Interface
PCIe Module
Dynamic Engineering
150 DuBois St. Suite C, Santa Cruz CA 95060
831-457-8891 831-457-4793 FAX

Dynamic Engineering has made every effort to ensure that this manual is accurate and complete. Still, the company reserves the right to make improvements or changes in the product described in this document at any time and without notice. Furthermore, Dynamic Engineering assumes no liability arising out of the application or use of the device described herein.

The electronic equipment described herein generates, uses, and can radiate radio frequency energy. Operation of this equipment in a residential area is likely to cause radio interference, in which case the user, at his own expense, will be required to take whatever measures may be required to correct the interference.

Dynamic Engineering's products are not authorized for use as critical components in life support devices or systems without the express written approval of the president of Dynamic Engineering.

This product has been designed to operate with PCIe motherboards and compatible user-provided equipment. Connection of incompatible hardware is likely to cause serious damage.

Manual Revision A2. Revised 7/13/15

Table of Contents

List of Figures

Product Description

In embedded systems many of the interconnections are made with differential [RS-422/485 or LVDS] signals. Depending on the system architecture an IP, PMC or native bus card will be the right choice to make the connection. You have choices with carriers for cPCI, PCI, PCIe, VME, PC/104p and other buses for both PMC and IP mezzanine modules.

The BiSerial family includes IP, PMC and PCIe versions, each with multiple "clientized" design implementations.

Usually the choice of format is based on other system constraints. Dynamic Engineering is happy to assist in your decision regarding architecture and other trade-offs with the form factor decision. Dynamic Engineering has carriers for IP and PMC modules for most systems, and is adding more as new solutions are requested by our clients.

The PCIe compatible PCIeBiSerialDb37 has 18 independent differential IO available. A DB-37 connector is mounted through the bezel to carry the signals. Each of the IO has independent direction and termination controls. Each of the IO is matched length and routed with 100 Ω differential impedance.

The IO's are buffered from the FPGA with differential transceivers. The transceivers can be populated with LVDS or RS-485 compatible devices. The power plane for the transceivers is isolated to allow selectable 3.3 or 5V references for the IO. The LVDS IO requires 3.3, and 40 MHz capable RS-485 requires 5V. When mixed LVDS and RS485 are used the reference is set to 3.3 and lower speed [20 MHz] RS-485 parts are used that are compatible with the 3.3V.

Each IO has pull-up and pull-down options to allow half duplex lines to be set to a "marking" state when no device is on the line. The P is is ganged and the M side is too. Each side can be set to gnd or vcc to allow a '1' or a '0' to be set on the lines. The resistors are in resistor packs and can be implemented with many values.

The terminations utilize analog switches to selectively parallel terminate the differential pair with approximately 100 ohms. It is recommended that the receiver side provide the termination.

The analog switches are protected with a diode on the input side of the power supply. The switches can back-feed voltage into the rest of the circuit when powered down and the system connected to it is not. The diodes allow for more flexible operation and power sequencing.

PcieBiSerialDb37Rtn8 is a "clientized" version of the standard PCIeBiSerialDb37 board. "RTN8" is set to use the LVDS standard, and supports one Transmit and one Receive

channel. The transmitter and receiver are designed to interface with a byte wide data stream using a gated clock.

The receive side auto-bauds to the incoming rate by sampling the clock with a 200 MHz internal reference. When an edge is detected the data is moved to the first leg of the FIFO chain. The transmitter is referenced to a programmable 2X clock and will automatically control the clock to be active when data can be transferred.

The Transmitter is supported with a combination of 4Kx32 FIFO and 8Kx32 FIFO for a total of 12Kx32. Using DMA transfers the transmit side can provide a continuous flow of data at the output or bursted depending on the programmed byte wide data speed and the system DMA capabilities.

The Receiver has a 4K x8 first stage FIFO followed by two 128Kx32 FIFO's and a 1Kx32 FIFO. The 1Kx32 FIFO serves as the DMA source FIFO, the two 128K x32 FIFO's the bulk storage for the receiver.

With a 40 MHz input stream, the receiver FIFO chain can store 1,056,760 bytes of data for ~26 mS of storage. With streaming DMA to a storage file, the system can adsorb the data for very large transfers with no loss of data. With 26 mS of storage the system can "be out to lunch" for a while without loss of data. The DMA engine can push unlimited size DMA transfers – the PCIe side will stop a particular transfer and the HW will automatically request another until the entire programmed transfer is complete.

The Receiver and Transmitter are separately supported with scatter gather capable DMA engines.

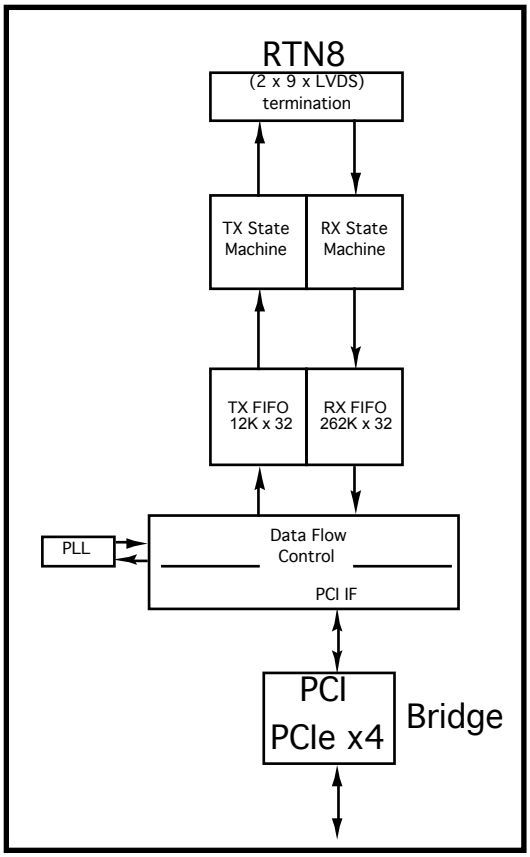


Figure 1 PcieBiSerialDb37Rtn8 Block Diagram Primary Function

The RTN8 supports transmission and reception of byte wide data. The Data transfer is controlled with a gated clock. The transmitter uses a 2x clock to be able to output and control the clock on a period-by-period basis. The receiver samples the clock and uses the build-up from 8 to 32 bits to reduce the clock rate needed. The data is received on the rising edge of the clock. The transmitter provides approximately 50% duty cycle and changes the data on the falling edge of the clock.

The PLL can be used to create a Tx clock reference to allow for loop-back testing. The control is via SW. The PLL is referenced to 50 MHz. and can be programmed with new .JED files using the driver.

The hardware waits until there is a definition of the byte count, and data in the transmit FIFO before starting. Once started if the data FIFO is empty when the transmitter is ready to read the next data set, an error for underflow is flagged. The error can cause an interrupt if desired.

The receiver uses the received clock to capture data with a small state-machine that loads the data into 4 parallel bytes to make a 32 bit version. The data is then loaded to

a second parallel register. The 4 byte time is sufficiently long to allow an internal clock to detect the loading of the secondary register and to move that word to the first holding FIFO. A side affect of this approach is that several clocks are required to move the initial byte into memory. The intention is to capture large quantities of data from an ongoing stream which allowed for this type of implementation.

Both the transmitter and receiver allow for bit and byte reversal. The data is stored as 32 bit words into the transmit FIFO from the system or the receive FIFO from the interface. The data is used with little Endian conventions as the default – 0,1,2,3 for the byte order where 0 = D7-0 [data on AD7-0] first and D31-24 last. The bits are sent LSB first so D0 is first on the line and D31 is last if all 4 bytes are to be sent. Similarly the receiver loads 0,1,2,3 so the first bit in goes into D0 and the last into D31 for each long word. When the bytes are reversed the order becomes 3,2,1,0 which which would make the IO 24 first, 7 last since it is still lsb first. The bit reversal swaps D7 with D0 etc on each byte read so the order becomes D31 first and D0 last if both reversal options are selected. For systems using Windows Little Endian is consistent with the driver and memory mapping. With Linux and some RISC based systems the reversal may be necessary.

Custom cables can be manufactured to your requirements. The loop-back IO definitions are toward the end of this manual. Please contact Dynamic Engineering with your specifications.

In the “RTN8” design the Termination and Direction controls are set in the VHDL for the IO. The received signals are terminated and the transmitted signals are not.

All of the IO is routed through the FPGA to allow for custom applications. Larger external and internal FIFO’s and Dual Ported memories are implemented for this version by FPGA selection and adding the 128K x 32 FIFO’s to the board.

The registers are mapped as 32 bit words and support 32 bit access. Most registers are read-writeable. The Windows® and Linux compatible drivers are available to provide the system level interface for this version of the BiSerial. Use standard C/C++ to control your hardware or use the Hardware manual to make your own software interface. The software manuals are also available on-line.

PcieBiserialDb37 can be used for multiple purposes with applications in telecommunications, control, sensors, IO, test; anywhere multiple independent or coordinated IO are useful.

PcieBiserialDb37 features a Xilinx FPGA, and high speed differential devices. The FPGA contains the PCI interface and control required for the IO interface.

The Xilinx design incorporates the “PCI Core” and additional modules for DMA in parallel with a direct register decoded programming model. The design model has a

“base” level with the basic board level functions and “channels” which contain IO oriented functions. In the RTN8 design the IO functions are designed into the channel and the PLL programming, switch, and other common or basic functions are in the base design.

From a software perspective the design can be treated as “Flat” or as a hierarchy. The Dynamic Engineering Windows® driver uses the hierarchical approach to allow for more consistent software with common bit maps and offsets. This implementation has only one channel. The channel function was kept to allow for future expansion with more than 1 IO interface or a secondary function in added channels. The user software can control the Channels with the same calls and use the channel number to distinguish. This makes for consistent and easier to implement user level software.

The hardware is designed with each of the channels on a common address map – each channel has the same memory allocated to it and as much as possible the offsets within each space are defined in the same way or similar way. Again this make understanding each port easier to accomplish and less likely to have errors.

The transceivers are initialized to the receive state. Once a channel is defined via software to be a transmitter the IO are enabled and driven to the appropriate levels. Terminations are activated for ports defined to be receivers.

PcieBiserialDb37 is part of the PCIe Module family of modular I/O components. The PcieBiserialDb37 conforms to the PCIe standard. This guarantees compatibility with your PCIe system. The base is 4 lane operation. The design can handle 1-4 lanes being available. LED’s are provided to show the active PCIe lanes.

Designs implemented on PC104p, PMC, IP and PCIe versions of the BiSerial family can in large part be ported between platforms. If you see what you need in one version and prefer it on another please contact Dynamic Engineering about porting the design. In most cases it will require a recompile of the VHDL and not much more. We do a lot of “just like but different “ adaptations for our clients. Please contact us to help you with a successful special adaptation of off- the-shelf hardware.

The DMA programmable length is 32 bits => longer than most computer OS will allow in one segment of memory. The DMA is scatter gather capable for longer lengths than the OS max and for OS situations where the memory is not contiguous. With Windows® lengths of 4K are common while Linux can provide much larger spaces. Larger spaces are more efficient as there are fewer initialization reads and reduced overhead on the bus. A single interrupt can control the entire transfer. Head to tail operation can also be programmed with two memory spaces with two interrupts per loop.

The hardware is organized with the IO function in channel 0 and the card level functions in the “base”. The driver provides the ability to find the hardware and to allocate resources to use the base and channel functions.

The basic use of the interface is to facilitate data transfer between the host and the remote target.

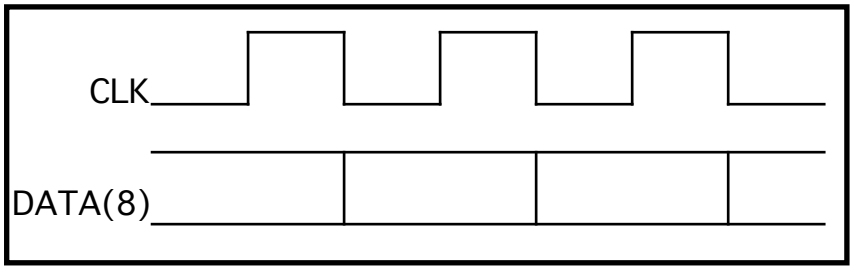


Figure 2 PCIEBISERIALDB37RTN8 Timing Diagram

The clock is gated with data valid on the rising edge. The transmitter provides data with close to 50% duty cycle – changing on the falling edge and stable on the rising edge.

The transmit rate is programmable for the RTN8. PLL channel A is programmed to 80 MHz for a 40 MHz. output rate. The receive rate for the initial implementation is 40 MHz.

Address Map

Base Address Map

Function	Offset
// **PCIeBiSerialDb37RTN8 BASE definitions**	
#define RTN8_BASE_BASE	0x0000 // 0 RTN8Base Base control register
#define RTN8_BASE_PLL_WRITE	0x0000 // 0 RTN8Base Base control register
#define RTN8_BASE_PLL_READ	0x0000 // 0 RTN8Base base control register
#define RTN8_BASE_USER_SWITCH	0x0004 // 1 RTN8Base User DIP switch read
#define RTN8_BASE_XILINX_REV	0x0004 // 1 RTN8Base Xilinx revision read port
#define RTN8_BASE_XILINX_DES	0x0004 // 1 RTN8Base Xilinx design read port
#define RTN8_BASE_STATUS	0x0008 // 2 RTN8Base status Register offset

Figure 3 PCIeBiSerialDb37RTN8 Internal Address Map Base Functions

The address map provided is for the local decoding performed within PcieBiserialDb37RTN8. The addresses are all offsets from a base address. Dynamic Engineering prefers a long-word oriented approach because it is more consistent across platforms.

The map is presented with the #define style to allow cutting and pasting into many compilers “include” files.

The host system will search the PCI bus to find the assets installed during power-on initialization. The VendorId = 0x10EE and the CardId = 0x0044 for the PcieBiSerialDb37RTN8.

The RTN8 design has 1 channel implemented at this time. The BASE contains the common elements of the design, while the Channels have the IO specific interfaces. The BASE starts at the card offset. Channel 0 starts at register 20

Section	Register Address Range (starting Hex address)	COM name
Base	0-19 (0x0000)	PLL, Switch, Status
Channel 0	20-39 (0x0050)	RTN8 Transmitter & Receiver

Channel Address Map

Function	Offset from Channel Base Address
// **PCIeBiSerialDb37RTN8 Channel definitions**	
#define RTN8_CHAN_CNTRL	0x00000000 //0 RTN8Chan General control register
#define RTN8_CHAN_STATUS	0x00000004 //1 RTN8Chan Interrupt status port
#define RTN8_CHAN_INT_CLEAR	0x00000004 //1 RTN8Chan Interrupt clear port
#define RTN8_CHAN_WR_DMA_PNTR	0x00000008 //2 RTN8Chan Write DMA dpr physical PCI address register
#define RTN8_CHAN_TX_FIFO_COUNT	0x00000008 //2 RTN8Chan TX FIFO count read port
#define RTN8_CHAN_RD_DMA_PNTR	0x0000000C //3 RTN8Chan Read DMA physical PCI address register
#define RTN8_CHAN_RX_FIFO_COUNT	0x0000000C //3 RTN8Chan RX FIFO count read port including pipeline
#define RTN8_CHAN_FIFO	0x00000010 //4 RTN8Chan FIFO for single word RW
#define RTN8_CHAN_TX_AMT_LVL	0x00000014 //5 RTN8Chan TX almost empty level RW
#define RTN8_CHAN_RX_AFL_LVL	0x00000018 //6 RTN8Chan RX almost full level register RW, used for HW control
#define RTN8_CHAN_TX	0x0000001C //7 RTN8Chan TX control register
#define RTN8_CHAN_RX	0x00000034 //13 RTN8Chan RX control register

Figure 4 PcieBiSerialDb37RTN8 Channel Address Map

Programming

Programming the PcieBiSerialDb37RTN8 requires only the ability to read and write data in the host's PCIe space.

Once the initialization process has occurred, and the system has assigned addresses to the PcieBiSerialDb37RTN8 card the software will need to determine what the address space is for the PCI interface [BAR0]. The offsets in the address tables are relative to the system assigned BAR0 base address.

The next step is to initialize the PcieBiSerialDb37Rtn8. The PLL will need to be programmed to use the loop-back function. The Cypress CyberClocks software can be used to create new .JED files if desired. PLLA should be set to the transmit reference frequency output by the transmitter.

The driver comes with a .JED file prepared. The driver has a utility to load the PLL and read back. The reference application software has an example of the use of PLL programming. The reference application software also includes XLATE.c which converts the .JED file from the CyberClocks tool to an array that can be programmed into the PLL.

The IO for the RTN8 direction and termination are hardwired in this design. The ports are unidirectional and initialization is simplified with this approach.

The control bits for the RTN8 will select how the data is transmitted – Byte ordering, size of transfer etc.

For Windows™ and Linux systems the Dynamic Drivers can be used. The driver will take care of finding the hardware and provide an easy to use mechanism to program the hardware. The Driver comes with reference software showing how to use the card and reference frequency files to allow the user to duplicate the test set-up used in manufacturing at Dynamic Engineering. Using simple, known to work routines is a good way to get acquainted with new hardware.

To use the RTN8 specific functions the Channel Control, and PLL interface plus DMA will need to be programmed. To use DMA, memory space from the system should be allocated and the link list stored into memory. The location of the link list is written to the RTN8 to start the DMA. Please refer to the Burst IN and Burst Out register discussions.

DMA should be set-up before starting the channel port function. For transmission this will result in the FIFO being full or close to it when the transfer is started or at least the Packet loaded if shorter than the FIFO size. For reception it means that the FIFO is under HW control and the delay from starting reception to starting DMA won't cause an

overflow condition.

DMA can be programmed with a specific length. The length can be as long as you want within standard memory limitations. At the end of the DMA transfer the Host will receive an interrupt. The receiver can be stopped and the FIFO reset to clear out any extra data captured. For on-the-fly processing multiple shorter DMA segments can be programmed; at the interrupt restart DMA to point at the alternate segment to allow processing on the previous one. This technique is sometimes referred to as “ping-pong”.

The Receive Byte Count register can be used to control the input Packet size to work with your DMA scheme. Alternately the Byte Count register can be programmed to the size of the Packet if known. For situations where the size is unknown the timeout and Almost Full interrupt options can be used.

Please see the channel control register bit maps for more information.

Base Register Definitions

RTN8_BASE_BASE

[$00 Base Control Register Port read/write]

DATA BIT	DESCRIPTION
31-21	spare
20	bit 19 read-back of pll_dat register bit
19	pll_dat [write to PLL, read-back from PLL]
18	pll_s2
17	pll_sclk
16	pll_en
15-0	spare

Figure 5 PcieBiSerialDb37RTN8 Control Base Register Bit Map

This is the base control register for the RTN8. The features common to all channels are controlled from this port. Unused bits are reserved for additional new features. Unused bits should be programmed '0' to allow for future commonality.

pll_en: When this bit is set to a one, the signals used to program and read the PLL are enabled.

pll_sclk/pll_dat : These signals are used to program the PLL over the I^2C serial interface. Sclk is always an output whereas Sdata is bi-directional. This register is where the Sdata output value is specified or read-back.

pll_s2: This is an additional control line to the PLL that can be used to select additional pre-programmed frequencies. Set to '0' for most applications.

The PLL is programmed with the output file generated by the Cypress PLL programming tool. [CY3672 R3.01 Programming Kit or CyberClocks R3.20.00 Cypress may update the revision from time to time.] The .JED file is used by the Dynamic Driver to program the PLL. Programming the PLL is fairly involved and beyond the scope of this manual. For clients writing their own drivers it is suggested to get the Engineering Kit for this board including software, and to use the translation and programming files ported to your environment. This procedure will save you a lot of time. For those who want to do it themselves the Cypress PLL in use is the 22393. The output file from the Cypress tool can be passed directly to the Dynamic Driver [Linux or Windows] and used to program the PLL without user intervention.

The reference frequency for the PLL is 50 MHz.

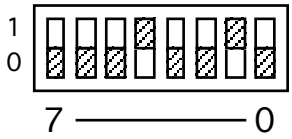
RTN8_BASE_ID

[$04 Switch and Design number port read only]

DATA BIT	DESCRIPTION
31-24	spare
23-8	Design ID and Revision
7-0	DIP switch

Figure 6 PcieBiSerialDb37RTN8 ID and Switch Bit Map

The DIP Switch is labeled for bit number and '1' '0' in the silk screen. The DIP Switch can be read from this port and used to determine which PcieBiserialDb37RTN8 physical card matches each PCI address assigned in a system with multiple cards installed. The DIPswitch can also be used for other purposes – software revision etc. The switch shown would read back 0x12.



The Design ID and Revision are defined by a 16 bit field allowing for 256 designs and 256 revisions of each. The RTN8 design is 0x02 the current revision is 0x01.

The PCI revision is updated in HW to match the design revision. The board ID will be updated for major changes to allow drivers to differentiate between revisions and applications.

RTN8_BASE_STATUS

[$08 Board level Status Port read only]

DATA BIT	DESCRIPTION
31-10	set to ‘0
9	undefined
8	undefined
7-1	set to ‘0’, reserved for additional channels
0	Unmasked Ch0 Interrupt

Figure 7 PcieBiSerialDb37RTN8 Status Port Bit Map

Channel Interrupt – The local interrupt status from the channel. Each channel can have different interrupt sources. DMA Write or DMA Read or IntForce or TX/RX request are typical sources. Polling can be accomplished using the channel status register and leaving the channel interrupt disabled.

Channel Bit Maps

The RTN8 design has 1 channel. The basic control signals are the same for the channel base, channel status, FIFO and DMA interfaces across multiple designs.

Notes:
The offsets shown are relative to the channel base address not the card base address.

RTN8_CHAN_CNTRL

[0x0] Channel Control Register (read/write)

Channel Control Register

Data Bit	Description
31-10	spare
9	External FIFO Reset
8	OutUrgent
7	InUrgent
6	Read DMA Interrupt Enable
5	Write DMA Interrupt Enable
4	Force Interrupt
3	Channel Interrupt Enable
2	Bypass
1	Receive FIFO Reset
0	Transmit FIFO Reset

Figure 8 PcieBiSerialDb37RTN8 channel Control Register

FIFO Transmitter/Receiver Reset: When set to a one, the transmit and/or receive FIFOs will be reset. When these bits are zero, normal FIFO operation is enabled. In addition the Transmit and Receive State Machines are also reset.

Write/Read DMA Interrupt Enable: These two bits, when set to one, enable the interrupts for DMA writes and reads respectively.

Channel Interrupt Enable: When this bit is set to a one, all enabled interrupts (except the DMA interrupts) will be gated through to the PCI interface level of the design; when this bit is a zero, the interrupts can be used for status without interrupting the host. The channel interrupt enable is for the channel level interrupt sources only.

Force Interrupt: When this bit is set to a one, a system interrupt will occur provided the Channel Interrupt enable is set. This is useful for interrupt testing.

InUrgent / OutUrgent when set causes the DMA request to have higher priority under certain circumstances. Basically when the TX FIFO is almost empty and InUrgent is set the TX DMA will have higher priority than it would otherwise get. Similarly if the RX FIFO is almost full and OutUrgent is set the read DMA will have higher priority. The purpose is to allow software some control over how DMA requests are processed and to allow for a higher rate channel to have a higher priority over other lower rate channels.

ByPass when set allows the FIFO to be used in a loop-back mode internal to the device. A separate state-machine is enabled when ByPass is set and the TX and RX are not enabled. The state-machine checks the TX and RX FIFO's and when not empty on the TX side and not Full on the RX side moves data between them. Writing to the TX FIFO allows reading back from the RX side. An example of this is included in the Driver reference software.

FIFO External Reset: When set to '1', the External FIFOs will be reset. When cleared the External FIFO is enabled.

RTN8_CHAN_STATUS

[0x4] Channel Status Read/Clear Latch Write Port

Channel Status Register

Data Bit	Description
31	Interrupt Status
30	LocalInt
29-28	Spare
27	TxFfIIFL
26	TxDmaFfIIAfl
25	TxDmaFfAmt
24	TxFfIIMt
23	BurstInIdle
22	BurstOutIdle
21	ExtFifo1Fl
20	ExtFifo0Fl
19	ExtFifo1Hf
18	ExtFifo0Hf
17	ExtFifo1Mt
16	ExtFifo0Mt
15	Read DMA Interrupt Occurred
14	Write DMA Interrupt Occurred
13	Read DMA Error Occurred
12	Write DMA Error Occurred
11	RxAFLvlIntLat
10	TxAELvlIntLat
9-7	spare
6	Rx FIFO Full
5	Rx FIFO Almost Full – complete chain
4	Rx FIFO Empty
3	Spare
2	Tx FIFO Full
1	Tx FIFO Almost Empty – complete chain
0	Tx FIFO Empty

Figure 9 PcieBiSerialDb37RTN8 Channel STATUS PORT

RTN8 FIFO: Three 4K x 32 FIFO's are used to create the internal Tx memory. The Rx side uses a combination of internal block RAM FIFO and two 128Kx32 FIFO's. The status for the Tx FIFO and Rx FIFO refer to these FIFO's. The status is active high. The Full and Empty status come from the "DMA" FIFO's while the Almost Full and Almost Empty status reflects the state of the total FIFO. 0x13 would correspond to empty Rx and empty Tx DMA FIFO's. The DMA FIFO's are the pair of internal FIFO's which interact with the DMA engine. First in pipeline for TX and last in pipeline for RX.

Please note with the Rx side status; the status reflects the state of the FIFO and does not take the 4 deep pipeline into account. For example the FIFO may be empty and there may be valid data within the pipeline. The data count with the combined FIFO and pipeline value and can also be used for read size control. [see later in register descriptions]

Rx FIFO Empty: When a one is read, the FIFO contains no data; when a zero is read, there is at least one data word in the FIFO.

Rx FIFO Almost Full: When a one is read, the number of data words in the data FIFO is greater than the value written to the corresponding RX_AFL_LVL register; when a zero is read, the FIFO level is less than that value.

Rx FIFO Full: When a one is read, the receive data FIFO is full; when a zero is read, there is room for at least one more data-word in the FIFO. If the FIFO is full when time to write received data to the FIFO an overflow error is declared.

Tx FIFO Empty: When a one is read, the FIFO contains no data; when a zero is read, there is at least one data word in the FIFO. If the FIFO is empty when time to read transmitted data from the FIFO an underflow error is declared.

Tx FIFO Almost Empty: When a one is read, the number of data words in the data FIFO is less than or equal to the value written to the corresponding TX_AMT_LVL register; when a zero is read, the FIFO level is more than that value.

Tx FIFO Full: When a one is read, the transmit data FIFO is full; when a zero is read, there is room for at least one more data-word in the FIFO.

Write/Read DMA Error Occurred: When a one is read, a write or read DMA error has been detected. This will occur if there is a target or master abort or if the direction bit in the next pointer of one of the chaining descriptors is incorrect. A zero indicates that no write or read DMA error has occurred. These bits are latched and can be cleared by writing back to the Status register with a one in the appropriate bit position.

Write/Read DMA Interrupt Occurred: When a one is read, a write/read DMA interrupt is latched. This indicates that the scatter-gather list for the current write or read DMA has completed, but the associated interrupt has yet to be processed. A zero indicates that no write or read DMA interrupt is pending.

BO and BI Idle are Burst Out and Burst In IDLE state status for the Receive and Transmit DMA actions. The bits will be 1 when in the IDLE state and 0 when processing a DMA. A new DMA should not be launched until the State machine is back in the IDLE state. Please note that the direction implied in the name has to do with the

DMA direction – Burst data into the card for Transmit and burst data out of the card for Receive.

Local Interrupt is the masked combined interrupt status for the channel not including DMA. The status is before the master interrupt enable for the channel.

Interrupt Status is the combined Local Interrupt with DMA and the master interrupt enable. If this bit is set this channel has a pending interrupt request.

RxAFLvlIntLat: When set the Rx Data FIFO has become almost Full based on the programmed count. The software can do a looped read or use DMA to retrieve the programmed count amount of data from the storage FIFO. The signal is latched and can be cleared via write back with this bit set. The signal can be used to generate an interrupt if desired.

TxAELvlIntLat: When set the Tx Data FIFO has become almost Empty based on the programmed count. The software can do a looped write or use DMA to load the programmed count amount of data to the storage FIFO. The signal is latched and can be cleared via write back with this bit set. The signal can be used to generate an interrupt if desired.

Four status bits are provided for the Transmit side FIFO. The MT and FULL are absolute and set when true. The AMT and AFL are programmable and set when the level meets the programmed requirements.

Six status bits are provided for the external FIFO's. The external FIFO's are part of the receiver data chain.

RTN8_CHAN_WR_DMA_PNTR
[0x8] Write DMA Pointer (write only)

BurstIn DMA Pointer Address Register	
Data Bit	Description
31-2	First Chaining Descriptor Physical Address
1	direction [0]
0	end of chain

Figure 10 PcieBiSerialDb37RTN8 Write DMA pointer register

This write-only port is used to initiate a scatter-gather write [TX] DMA. When the address of the first chaining descriptor is written to this port, the DMA engine reads three successive long words beginning at that address. Essentially this data acts like a chaining descriptor value pointing to the next value in the chain.

The first is the address of the first memory block of the DMA buffer containing the data to read into the device, the second is the length in bytes of that block, and the third is the address of the next chaining descriptor in the list of buffer memory blocks. This process is continued until the end-of-chain bit in one of the next pointer values read indicates that it is the last chaining descriptor in the list.

All three values are on LW boundaries and are LW in size. Addresses for successive parameters are incremented. The addresses are physical addresses the HW will use on the PCI bus to access the Host memory for the next descriptor or to read the data to be transmitted. In most OS you will need to convert from virtual to physical. The length parameter is a number of bytes, and must be on a LW divisible number of bytes.

Status for the DMA activity can be found in the channel control register and channel status register.

Notes:
1. Writing a zero to this port will abort a write DMA in progress.
2. End of chain should not be set for the address written to the DMA Pointer Address Register. End of chain should be set when the descriptor follows the last length parameter.
3. The Direction should be set to ‘0’ for Burst In DMA in all chaining descriptor locations.

RTN8_CHAN_TX_FIFO_COUNT

[0x8] TX [Target] FIFO data count (read only)

TX FIFO Data Count Port	
Data Bit	Description
31-16	Spare
15-0	TX Data Words Stored

Figure 11 PcieBiSerialDb37RTN8 TX FIFO data count Port

This read-only register port reports the number of 32-bit data words in the Transmit FIFO. The FIFO count is padded with '0' to a word boundary.

RTN8_CHAN_RD_DMA_PNTR

[0xC] Read DMA Pointer (write only)

BurstIn DMA Pointer Address Register	
Data Bit	Description
31-2	First Chaining Descriptor Physical Address
1	direction [1]
0	end of chain

Figure 12 PcieBiSerialDb37RTN8 Read DMA pointer register

This write-only port is used to initiate a scatter-gather read [RX] DMA. When the address of the first chaining descriptor is written to this port, the DMA engine reads three successive long words beginning at that address. Essentially this data acts like a chaining descriptor value pointing to the next value in the chain.

The first is the address of the first memory block of the DMA buffer to write data from the device to, the second is the length in bytes of that block, and the third is the address of the next chaining descriptor in the list of buffer memory blocks. This process is continued until the end-of-chain bit in one of the next pointer values read indicates that it is the last chaining descriptor in the list.

All three values are on LW boundaries and are LW in size. Addresses for successive parameters are incremented. The addresses are physical addresses the HW will use on the PCI bus to access the Host memory for the next descriptor or to read the data to be transmitted. In most OS you will need to convert from virtual to physical. The length parameter is a number of bytes, and must be on a LW divisible number of bytes.

Status for the DMA activity can be found in the channel control register and channel status register.

Notes:

1. Writing a zero to this port will abort a write DMA in progress.
2. End of chain should not be set for the address written to the DMA Pointer Address Register. End of chain should be set when the descriptor follows the last length parameter.
3. The Direction should be set to '1' for Burst Out DMA in all chaining descriptor locations.

RTN8_CHAN_RX_FIFO_COUNT

[0xC] RX [Master] FIFO data count (read only)

RX FIFO Data Count Port	
Data Bit	Description
31-0	RX Data Words Stored

Figure 13 PcieBiSerialDb37RTN8 RX FIFO data count Port

This read-only register port reports the number of 32-bit data words in the Receive FIFO plus pipeline. The maximum count is the FIFO size plus 4. The FIFO count is padded with '0' .

RTN8_CHAN_FIFO

[0x10] Write TX/Read RX FIFO Port

RX and TX FIFO Port	
Data Bit	Description
31-0	FIFO data word

Figure 14 PcieBiSerialDb37RTN8 RX/TX FIFO Port

This port is used to make single-word accesses to and from the FIFO. Data read from this port will no longer be available for DMA transfers. Writing to the port loads the Tx FIFO, Reading unloads the Rx FIFO.

RTN8_CHAN_TX_AMT_LVL

[0x14] Tx almost-empty level (read/write)

Tx Almost-Full Level Register	
Data Bit	Description
31-16	Spare
15-0	Tx FIFO Almost-Empty Level

Figure 15 PcieBiSerialDb37RTN8 TX ALMOST EMPTY LEVEL register

This read/write port accesses the almost-empty level register. When the number of data words in the transmit data FIFO is less than than this value, the almost-empty status bit will be set. The register is R/W for 16 bits. The mask is valid for a size matching the depth of the FIFO.

RTN8_CHAN_RX_AFL_LVL

[0x18] Rx almost-full (read/write)

Rx Almost-Full Level Register	
Data Bit	Description
31-0	Rx FIFO Almost-Full Level

Figure 16 PcieBiSerialDb37RTN8 RX ALMOST FULL LEVEL register

This read/write port accesses the almost-full level register. When the number of data words in the receive data FIFO is equal or greater than this value, the almost-full status bit will be set. The mask is valid for a size matching the depth of the FIFO. The level includes the pipeline for an additional 4 locations.

RTN8_CHAN_TX

[0x1C] Channel Transmit Control Register (read/write)

Channel TX Control Register	
Data Bit	Description
3	TxAEIntEn
2	spare
1	spare
0	TxEn

Figure 17 PcieBiSerialDb37RTN8 Channel Transmit Control Register

TxEn when set causes the Transmit State Machine to begin operation. When the Transmitter has determined that Data is in the Transmit FIFO data will be transmitted. Clearing TxEn will return the State Machine to the idle state, generally at the end of the current transmitted byte.

Disabling TxEn while transmitting is considered a “panic stop” situation. The FIFO’s should be cleared and the process restarted from scratch when this is done. Restarting without clearing will allow the data to be completed but with the incorrect remaining packet count likely leading to error(s) being detected.

TxAEIntEn when set enables the interrupt based on the TX FIFO Almost Empty flag. When the interrupt occurs a programmable amount of data can be stored into the FIFO making for an efficient DMA or burst of writes to load the FIFO.

Interrupt status available in the Status Register. Clear by writing a ‘1’ to the corresponding status position.

RTN8_CHAN_RX

[0x34] Channel RX Control Register (read/write)

Channel Control Register

Data Bit	Description
15-4	spare
3	RxFifoAFIntEn
2	Spare
1	Spare
0	RxEn

Figure 18 PcieBiSerialDb37RTN8 Channel Rx Control Register

RxEn when set causes the Rx State Machine to begin operation. Data is captured on the rising edge of the received clock.

RxFifoAFIntEn when set enables the interrupt based on the Rx FIFO Almost Full flag. When the interrupt occurs a programmable amount of data can be read from the FIFO making for an efficient DMA read or burst of reads to unload the FIFO.

Loop-back

The Engineering kit includes reference software, utilizing external loop-back tests.

The test set-up included PcieBiSerialDb37RTN8 and loop-back plug. The Pin numbers are for the interconnections on the Loop-back plug. The IO names can be used to accommodate a different set-up. The loop-back plug is a DB37 connector with the interconnections protected with a connector shell.

Signal	From	To	Signal
TxData_0+	28	1	RxData_0+
TxData_0-	29	2	RxData_0-
TxData_1+	11	20	RxData_1+
TxData_1-	12	21	RxData_1-
TxData_2+	30	3	RxData_2+
TxData_2-	31	4	RxData_2-
TxData_3+	13	22	RxData_3+
TxData_3-	14	23	RxData_3-
TxData_4+	32	5	RxData_4+
TxData_4-	33	6	RxData_4-
TxData_5+	15	24	RxData_5+
TxData_5-	16	25	RxData_5-
TxData_6+	34	7	RxData_6+
TxData_6-	35	8	RxData_6-
TxData_7+	17	26	RxData_7+
TxData_7-	18	27	RxData_7-
TxClk+	36	9	RxClk+
TxClk-	37	10	RxClk-

PCIe Module Front Panel IO Interface Pin Assignment

The figure below gives the pin assignments for the IO Interface on the PcieBiSerialDb37Rtn8.

IO_0p (RxData0+)	IO_0m (RxData0-)	1	2
IO_1p (RxData1+)	IO_1m (RxData1-)	20	21
IO_2p (RxData2+)	IO_2m (RxData2-)	3	4
IO_3p (RxData3+)	IO_3m (RxData3-)	22	23
IO_4p (RxData4+)	IO_4m (RxData4-)	5	6
IO_5p (RxData5+)	IO_5m (RxData5-)	24	25
IO_6p (RxData6+)	IO_6m (RxData6-)	7	8
IO_7p (RxData7+)	IO_7m (RxData7-)	26	27
IO_8p (RxClk+)	IO_8m (RxClk -)	9	10
IO_9p (TxData0+)	IO_9m (TxData0-)	28	29
IO_10p (TxData1+)	IO_10m (TxData1-)	11	12
IO_11p (TxData2+)	IO_11m (TxData2-)	30	31
IO_12p (TxData3+)	IO_12m (TxData3-)	13	14
IO_13p (TxData4+)	IO_13m (TxData4-)	32	33
IO_14p (TxData5+)	IO_14m (TxData5-)	15	16
IO_15p (TxData6+)	IO_15m (TxData6-)	34	35
IO_16p (TxData7+)	IO_16m (TxData7-)	17	18
IO_17p (TxClk +)	IO_17m (TxClk -)	36	37
GND*		19	

Figure 19 PcieBiSerialDb37RTN8 FRONT PANEL Interface

GND is shunt selectable for DC, AC and open configurations. Jumper is located near lower edge of DB37 connector. Options labeled in silk-screen.

Applications Guide

Interfacing

The pin-out tables are displayed with the pins in the same relative order as the actual connectors. Some general interfacing guidelines are presented below. Do not hesitate to contact the factory if you need more assistance.

Watch the system grounds. All electrically connected equipment should have a fail-safe common ground that is large enough to handle all current loads without affecting noise immunity. Power supplies and power-consuming loads should all have their own ground wires back to a common point.

Power all system power supplies from one switch. Differential interface devices provide some immunity from, and allow operation when part of the circuit is powered on and part is not. It is better to avoid the issue of going past the safe operating areas by powering the equipment together and by having a good ground reference.

Keep cables short. Flat cables, even with alternate ground lines, are not suitable for long distances. In addition series resistors are used and can be specified to be something other than the 0 ohm standard value. The connector is pinned out for a standard DB37 cable to be used. It is suggested that this standard cable be used for most of the cable run or an equivalent with proper twisted pairs and shielding.

Coming Soon. Terminal Block. We offer a high quality 37 screw terminal block that directly connects to the DB37. The terminal block can mount on standard DIN rails. DBterm37 has an associated twisted pair cable compatible with the PCIeBiSerialDB37. [http://www.dyneng.com/DBterm68.html]

We provide the components. You provide the system. Safety and reliability can be achieved only by careful planning and practice. Inputs can be damaged by static discharge, or by applying voltage outside of the particular device's rated voltages.

Construction and Reliability

PCIe Modules were conceived and engineered for rugged industrial environments. The PcieBiSerialDb37RTN8 is constructed out of 0.062 inch thick high temperature ROHS compliant material.

The traces are matched length from the FPGA ball to the IO pin. The analog switches and termination resistors are located directly under the transceivers and connected with “zero stub” routing to eliminate unwanted effects from unused options.

Surface mounted components are used. The components are available with commercial and Industrial temperature ranges. Please order the “ET” version for more demanding environments. Conformal coating is an option for condensing environments or for another measure of board protection. Please order the “CC” version.

The PCIe is secured against the chassis with the connectors and front panel. If more security against vibration is required a chassis with top side support can be used. The PCIeBiSerialDb37 has a wider keep out than required by PCIe specification to allow use in industrial chassis and horizontal mount situations.

The power and ground planes are implemented with relatively heavy copper to help with heat spreading in chassis with limited air flow. The components are spaced to allow for efficient cooling and power dispersion.

Thermal Considerations

The PcieBiSerialDb37RTN8 design consists of CMOS and similar circuits. The power dissipation due to internal circuitry is very low. It is possible to create higher power dissipation with the externally connected logic. If more than one Watt is required to be dissipated due to external loading; cooling with forced air is recommended. With the one degree differential temperature to the solder side of the board external cooling is easily accomplished.

Warranty and Repair

Please refer to the warranty page on our website for the current warranty offered and options. http://www.dyneng.com/warranty.html

Service Policy

Before returning a product for repair, verify as well as possible that the suspected unit is at fault. Then call the Customer Service Department for a RETURN MATERIAL AUTHORIZATION (RMA) number. Carefully package the unit, in the original shipping carton if this is available, and ship prepaid and insured with the RMA number clearly written on the outside of the package. Include a return address and the telephone number of a technical contact. For out-of-warranty repairs, a purchase order for repair charges must accompany the return. Dynamic Engineering will not be responsible for damages due to improper packaging of returned items. For service on Dynamic Engineering Products not purchased directly from Dynamic Engineering contact your reseller. Products returned to Dynamic Engineering for repair by other than the original customer will be treated as out-of-warranty.

Out of Warranty Repairs

Out of warranty repairs will be billed on a material and labor basis. The current minimum repair charge is $125. Customer approval will be obtained before repairing any item if the repair charges will exceed one half of the quantity one list price for that unit. Return transportation and insurance will be billed as part of the repair and is in addition to the minimum charge.

For Service Contact:

Customer Service Department
Dynamic Engineering
150 DuBois St. Suite C
Santa Cruz, CA 95060
831-457-8891
831-457-4793 fax
support@dyneng.com

Specifications

Logic Interface:	PCIe 1-4 lanes. 4 lanes recommended
Digital Parallel IO:	LVDS IO
Digital Serial IO:	Byte Wide data plus clock for Tx and Rx function. Data valid on rising edge of clock. 40 MHz for initial target design.
DIP Switch:	DipSwitch supplied for board identification and other user purposes.
CLK rates supported:	PLLA is programmed to select Transmit Clock rate. For loop-back and alternate HW implementations. PLLB, C, D reserved for new applications.
Software Interface:	Control Registers, IO registers, IO Read-Back registers, FIFO. R/W, 32 bit boundaries.
Initialization:	Programming procedure documented in this manual
Access Modes:	LW to registers, read-write to most registers
Access Time:	Frame to TRDY 121 nS [4 PCI clocks] or burst mode DMA – 1 word per PCI clock transferred.
Interrupt:	Each port has independently programmable interrupt sources, DMA interrupts included.
Onboard Options:	All Options are Software Programmable
Interface Options:	37 Pin DB connector at front bezel.
Dimensions:	Standard 1/2 length PCIe module.
Construction:	Multi-Layer Printed Circuit, Through Hole and Surface Mount Components.
Power:	+12 and +3.3 used from PCIe interface. No secondary power supply connections required. 1.2, 2.5 and 5V developed locally.
Weight:	TBD oz

Order Information

standard temperature range Industrial
PcieBiSerialDb37RTN8 PCIe Module with18 IO channels. One Transmitter and one Receiver port implemented. [Byte wide plus clock]
http://www.dyneng.com/pciebiserialdb37.html

Order Options:

Pick any combination to go with IO
-CC to add conformal coating
-ET to change to industrial Temp [-40 - +85C] Standard this design
-COM to change to commercial temp parts [0-70]

Dbterm37: 37 position terminal block with DB37 connector.
http://www.dyneng.com/DBterm37.html

Dbcabl37: DB37 cable compatible with PCIeBiSerialDB37. Twisted pairs on correct pin pairs. **http://www.dyneng.com/DBcabl37.html**

PCIe BiSerial DB37 RTN8 Eng Kit : Windows or Linux Driver software, Loop-Back Plug, reference schematics. Recommended for first time purchases.
http://www.dyneng.com/pciebiserialdb37.html