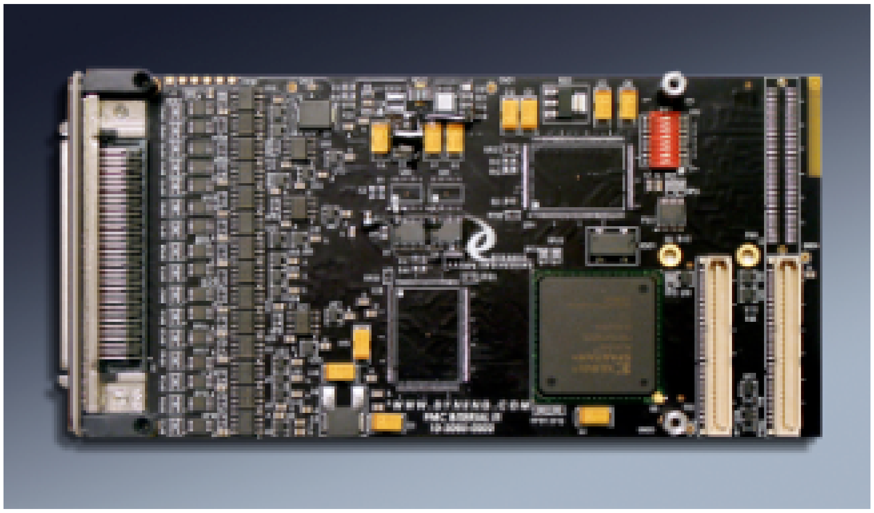
DYNAMIC ENGINEERING
150 DuBois St. Suite C, Santa Cruz, CA 95060
831-457-8891   Fax 831-457-4793
http://www.dyneng.com
sales@dyneng.com
Est. 1988

User Manual

PMC-BiSerial-III-NG8

Digital Parallel Interface
PMC Module
Camera Interface Protocols
TX & RX
2 Channels
RS-485/422



Manual Revision A
Corresponding Fab: 10-2005-0204
FLASH 0x0B01

PMC-BiSerial-III-NG8
Digital Parallel Interface
PMC Module
Dynamic Engineering
150 DuBois St. Suite C, Santa Cruz CA 95060
831-457-8891 831-457-4793 FAX

This document contains information of proprietary interest to Dynamic Engineering. It has been supplied in confidence and the recipient, by accepting this material, agrees that the subject matter will not be copied or reproduced, in whole or in part, nor its contents revealed in any manner or to any person except to meet the purpose for which it was delivered.

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 PMC Module carriers and compatible user-provided equipment. Connection of incompatible hardware is likely to cause serious damage.

Revised 11/23/09

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 or a PMC will be the right choice to make the connection. You have choices with carriers for cPCI, PCI, VME, PC/104p and other buses for both PMC and IP mezzanine modules.

Usually the choice is based on other system constraints as both the PMC and IP can provide the IO you require. Dynamic Engineering would be happy to assist in your decision regarding architecture and other trade-offs with the PMC / IP 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 PMC compatible PMC-BiSerial-III has 34 independent differential IO available. The high density makes efficient use of PMC slot resources. The IO is available for system connection through the front panel [34], via the rear [Pn4] connector [32], or both. A high density 68 pin SCSI III front panel connector provides the front panel IO. The rear panel IO has a PIM and PIM Carrier available for rear panel wiring options.

PMC-BiSerial-III-NG8 is a "clientized" version of the standard PMC-BiSerial-III board. "NG8" is set to use the RS-485 standard, has rear panel IO, and supports two channels each with Transmit or Receive capability. The PLL is programmed with the Transmit reference frequency and the Receive state-machine frequency. The PLL is referenced to 50 MHz. and can be programmed with new .JED files using the driver.

Data is transferred as pixel wide [11-0] with a reference clock, "BadBit", Vertical and Horizontal reference signals. There is no handshaking. The Receiver uses the VREF signal to "know" where the start of an image is. The rising edge of the clock is used to capture the pixel and reference signals. Using the programmable image size count data is captured for a complete image and status set indicating a complete image is ready. The internal memory is able to hold 3 complete images of 288 x 290 and be receiving a 4th. DMA is available to insure the data is transferred from the local FIFO to the system.

The transmitter sends data out based on what is stored into memory. The pixels are 12 bits [11-0]. The upper nibble of each word is used to set the clock enable, VREF, HREF, and BadBit signals. The transmitter is designed to force the signals to 0xffff when not actively transmitting.

Software can select between transmit and receive functions for each channel. The direction bit sets the direction for the RS-485 transceivers and the terminations. When in receive mode the terminations are enabled and the transceivers are set to receive.

DMA or single word accesses can be used to load and unload the FIFO's. 4K x 32 FIFO's are used for the DMA transfer TX and RX. In addition there are local State-machine FIFO's 1Kx32 to handle the rate matching between the bulk storage and the local speed. One external [to the FPGA] 128K x 32 FIFO is assigned to each channel. Buffering around the input and output ports allows the external FIFO to be used for transmit or receive. Total storage for TX is 4K + 1K + 128K. Since each LW holds 2 pixels the memory can hold 3+ images based on 288 x 290. For the receive side there are an additional 4 locations in the DMA pipeline. The Dynamic Driver supports both modes of operation.

The Transmit and Receive can be used in pairs for loop-back testing. HDEterm68 http://www.dyneng.com/HDEterm68.html can be used as a breakout for the rear panel IO. The HDEcabl68 provides a convenient cable. http://www.dyneng.com/HDEcabl68.html 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.

All of the IO are routed through the FPGA to allow for custom applications that require hardware intervention or specific timing- for example an automatic address or data strobe to be generated. The initial model was register based [FLASH 0101]. Please contact Dynamic Engineering with your custom requirements. NG8 is design number "B" for the PMC-BiSerial-III with a corresponding FLASH of 0Bxx.

The IO 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 RS-485 parts are used that are compatible with the 3.3V.

The IO are matched from the connector edge to the ball on the FPGA. The differential side is routed with controlled impedance traces. "Trace and space".

Each of the transceivers has separate direction and termination controls to allow for Any configuration of in and out, half and full duplex designs.

Each of the IO has series terminations to allow the IO to be isolated or terminated. The isolation feature is used to allow rear or front panel implementations without "stub" issues for higher speed signals.

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 the PMC is powered down and the system connected to it is not. The DIODE’s allow for more flexible operation and power sequencing.

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

PMC-BISERIAL-III is part of the PMC Module family of modular I/O components. The PMC-BISERIAL-III conforms to the PMC standard. This guarantees compatibility with multiple PMC Carrier boards. Because the PMC may be mounted on different form factors, while maintaining plug and software compatibility, system prototyping may be done on one PMC Carrier board, with final system implementation on a different one.

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

PMC-BISERIAL-III features a Xilinx FPGA, and high speed differential devices. The FPGA contains the PCI interface and control required for the parallel 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 NG8 design the COM functions are designed into channels 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. The user software can control the COM pairs 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.

All the IO control and registers are instantiated within the FPGA, only the transceivers and termination switches are separate devices.   If desired, the IO lines can be specially programmed to create custom timing pulses etc.   Please contact Dynamic Engineering with your requirements.

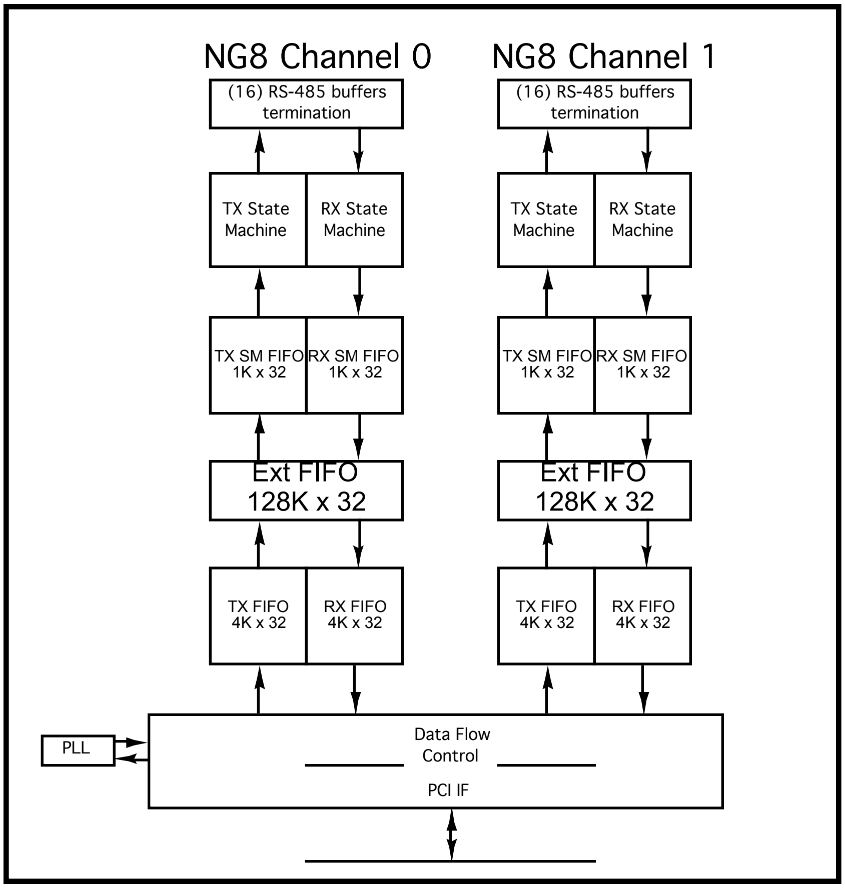


Figure 1 PMC-BISERIAL-III-NG8 Block Diagram

PMC BiSerial III NG8 features two channels each with Transmit and Receive capabilities. Each channel has separate DMA support. The internal block RAM is configured to provide FIFO's to support the DMA and IO transfer process. 4K x 32 per port to support DMA directly and 1Kx32 to support the RX or TX function.   In addition the external FIFO is configured to be used for either Transmit or receive creating a large 133K x 32 FIFO.

Images size is programmable. 288 x 290 is the base design size. 288 * 290 / 2 = LW required = 41760. With 133K more than 3 images can be stored for TX or RX.

The hardware has facilities to provide the overall data stored count within the three FIFO's as well as reading the counts individually. The hardware makes use of the counts to hold off starting the TX process until 1 image is stored in memory and to regulate the transfer of data between FIFO's. FIFO's have delays associated with the MT, FULL, Almost Full and Almost Empty flags plus the state-machine has a delay in reacting to the change in state. The hardware uses the counts to determine the almost full and almost empty conditions to allow the data movers to burst when not near the edge of the FIFO space and to slow down and check the flags with delays when almost empty or almost full. This way all data is moved and data in the middle moves much faster. With DMA and the levels set properly the FIFO's will operate in the middle of their ranges to properly source or receive data without overflow or under-run conditions occurring.

The hardware will pull data from the FIFO memory and store into the system memory using DMA and vice-versa. The transfer function will load the FIFO and DMA will unload. The DMA function operates at the PCI bus frequency. The transfer frequency will determine the maximum load rate into the FIFO.

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 channels 0 and 1 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.

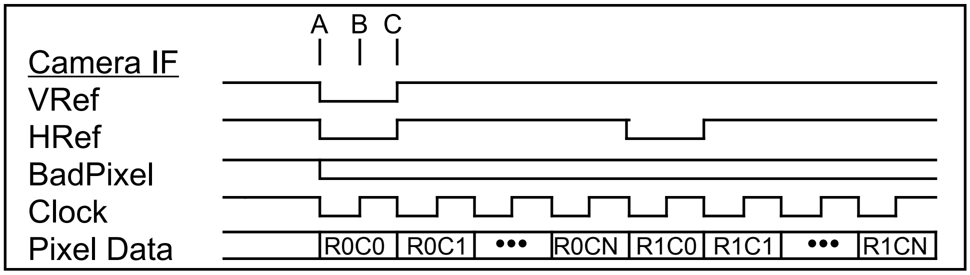


Figure 2 PMC-BISERIAL-III-NG8 Timing Diagram

With the standard rate clock [10 Mhz], the period will be 100 nS. The Clock is symmetrical with respect to the data providing 50 nS of set-up and hold minus any cable induced skew, slew time, and skew from the receiving circuit. Signals are held in the high state for a transmitter until an image is transferred. VREF, HREF, BadPixel and the Pixel Data all come from memory and can be programmed as shown or with an alternative control sequence. BadPixel in particular could be used as a 13th data bit if desired.

The design is programmable for LW's transferred per image. Normally there is some blanking area around the viewable area.

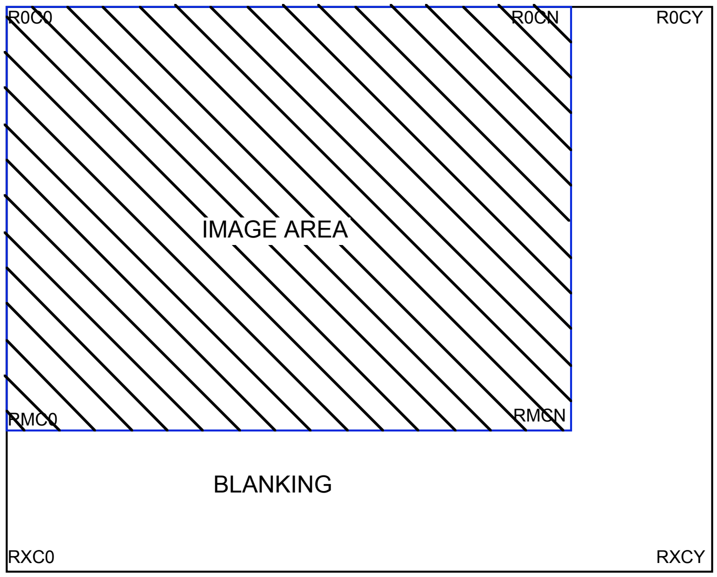


Figure 3 Image Dimensions

The hardware will process the data in the order stored into the FIFO. By convention the data is organized by rows and columns. Row0Column0 is the first location broadcast and has Vref and Href set as shown in Figure 2. The pixels transmitted progress through the visible part of the image transitioning to the blanking portion at the last visible column. After the blanking at the end of the row, the next row is immediately transmitted. Href is asserted for column 0. The process repeats until the end of the visible rows is reached at which point all of the data transmitted are in the blanking area. Href is suppressed for the rows where no visible data is transmitted.

The total pixel count is the total of all of the pixels in the entire array including the blanking. (X+1)(Y+1). The programmed value is 1/2 of the total pixel count since the data is stored as long words.

A typical application may have the visible area set to 256 x 256 and the total area set to 288 x 290. This would make the blanking per row 32 pixels, and the blanking by line 34 lines.

With a 10 MHz transmit frequency the transmit time for the example above would be 8.352 mS. This comes from 288 x 290 * 100 nS per pixel.

With 83520 pixels per image there will be room for 3 complete images and part of a 4th in the memory for each channel.

The hardware is designed to wait for the image size to be available in memory before starting to send the first image. One images are being transmitted the image transmission continues back-to-back until the enable is disabled or the memory is empty at the start of an image.

Data is read from the FIFO lower then upper. R0C0 will be the data stored in D15-0 and R0C1 from D31-16 etc. If your system loads in the opposite order the DataOrder control bit can be used to reverse the order of the data processing.

Address Map

```
Function                            Offset
// PMC BiSerial III NG8 definitions
#define NG8_BASE_BASE               0x0000 // 0 NG8Base Base control register
#define NG8_BASE_PLL_WRITE          0x0000 // 0 NG8Base Base control register
#define NG8_BASE_PLL_READ           0x0000 // 0 NG8Base base control register
#define NG8_BASE_USER_SWITCH        0x0004 // 1 NG8Base User switch read port  DIP switch read
#define NG8_BASE_XILINX_REV         0x0004 // 1 NG8Base Xilinx revision read port
#define NG8_BASE_XILINX_DES         0x0004 // 1 NG8Base Xilinx design read port

#define NG8_BASE_STATUS             0x0008 // 2 NG8Base status Register offset
```

Figure 4 PMC-BISERIAL-III Internal Address Map Base Functions

The address map provided is for the local decoding performed within PMC-BiSerial-III. The addresses are all offsets from a base address. The carrier board that the PMC is installed into provides the 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 = 0x003F for the PMC-BiSerial-III-NG8.

The NG8 design has 2 channels implemented. 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 28

Section	Register Address Range (starting Hex address)	Port name
Base	0-19 (0x0000)	PLL, Switch, Status
Channel 0	20-39 (0x0050)	NG8 Camera Interface Tx & Rx
Channel 1	40-59 (0x00A0)	NG8 Camera Interface Tx & Rx

```
Function                                          Offset from Channel Base Address
// PMC BiSerial III NG8 Channel definitions
#define NG8_CHAN_CNTRL                    0x00000000  //0 General control register

#define NG8_CHAN_STATUS                   0x00000004  //1 Interrupt status port
#define NG8_CHAN_INT_CLEAR                0x00000004  //1 Interrupt clear port

#define NG8_CHAN_WR_DMA_PNTR              0x00000008  //2 Write DMA physical PCI address  reg
#define NG8_CHAN_TX_FIFO_COUNT            0x00000008  //2 TX FIFO count read port

#define NG8_CHAN_RD_DMA_PNTR              0x0000000C  //3 Read DMA physical PCI address reg
#define NG8_CHAN_RX_FIFO_COUNT            0x0000000C  //3 RX FIFO count port including pipeline

#define NG8_CHAN_FIFO                     0x00000010  //4 FIFO single word access RW
#define NG8_CHAN_TX_AMT_LVL               0x00000014  //5 TX AMT level register RW, DMA FIFO
#define NG8_CHAN_RX_AFL_LVL               0x00000018  //6 RX AFull level reg RW, DMA FIFO

#define NG8_CHAN_TX                       0x0000001C  //7 TX control register
#define NG8_CHAN_TX_PIXEL_WORD_COUNT      0x00000020  //8 TX Pixel count /2
#define NG8_CHAN_TX_FIFO_WORD_COUNT       0x00000024  //9 TX StateMachine FIFO count TX
#define NG8_CHAN_TX_AMT_LVL_TOTAL         0x00000028  //10 TX AMT level control for interrupt

#define NG8_CHAN_RX                       0x00000034  //13 RX control register
#define NG8_CHAN_RX_PIXEL_WORD_COUNT      0x00000038  //14 RX Pixel Count Expected in words
#define NG8_CHAN_RX_FIFO_WORD_COUNT       0x0000003C  //15 RX StateMachine FIFO count  RX
#define NG8_CHAN_RX_AFL_LVL_TOTAL         0x00000040  //16 RX AFull level control for interrupt

#define NG8_CHAN_EXT_FIFO_WORD_COUNT      0x00000044  //17 External FIFO Data Count,
#define NG8_CHAN_TX_WORD_COUNT            0x00000048  //18 TX total data count
                                          => TX DMA + EXT + TX
#define NG8_CHAN_RX_WORD_COUNT            0x0000004C  //19 RX total data count
                                          => Pipeline + RX DMA + EXT + RX
```

Figure 5 PMC-BISERIAL-III Channel Address Map

Programming

Programming the PMC-BISERIAL-III-NG8 requires only the ability to read and write data in the host's PMC space.

Once the initialization process has occurred, and the system has assigned addresses to the PMC-BiSerial-III-NG8 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 PMC-BiSerial-III-NG8. The PLL will need to be programmed to use the NG8 function. The Cypress CyberClocks software can be used to create new .JED files if desired. The PLLA should be set to the transmit reference frequency output by the Master.

The driver comes with several .JED files 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 direction and termination are hardwired in this design. The ports are unidirectional and initialization is simplified with this approach.

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

For Windows™ and Linux systems the Dynamic Drivers[1] 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 NG8 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 NG8 to start the DMA. Please refer to the Burst IN and Burst Out register discussions.

[1] Currently only Windows® is supported. Please contact Dynamic Engineering for Linux.

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. 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.

The Ready Count register can be programmed to make the HW wait until there is at least the Ready Count number of positions filled in the FIFO's before starting. When larger non-stop transfers are desired an amount close to capacity is recommended.

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".

Other DMA notes: When DMA is operating in the receive pixels direction => writing to system memory, the PCI control signals can be used to indicate how much data is being transferred. Frame and IRDY remain asserted until the end of the requested length unless the Host asserts a STOP. In most cases a 256 word transfer is accomplished per DMA transfer with the hardware automatically generating new requests to move the data from the FIFO to the host memory. Most Bridge devices have FIFO buffers deep enough to allow for the full size DMA transfer to be accommodated resulting in a pipelined large word count transfer taking place.

In some cases the Bridge may need to be reprogrammed to do larger transfers to facilitate obtaining full bandwidth on the bus.

When operating as a transmitter the Hardware does a read from system memory and uses the same control signals to accomplish the transfer. The PMC BiSerial III NG8 can handle back-to-back max size transfers. In some cases the Bridge may need to be reprogrammed to do larger transfers to facilitate obtaining full bandwidth on the bus.

Since the request from the NG8 must pass through the bridges to the host prior to the host sourcing data, and since there is a delay in passing through the bridge, the transfer size is reduced to the Bridge prefetch length.

In addition while the host PCI controller can adsorb data at full length repeatedly, the same controller will frequently assert STOP after 16 words drastically shortening the DMA transfer. Adjusting the default for the PCI controller may also be necessary to reach proper operational data flow. The NG8 hardware has no control over the bridge or host PCI controller.

By moving the PMC BiSerial III NG8 to the first PCI bus – attached to the PCI controller all bridges are bypassed allowing 16 word transfers in the Transmit direction and 256 in the Receive direction. With a burst size of 16 the overall throughput is better than 25% which is enough to sustain 1 channel at 5 MHz effective bandwidth [10 M pixels per second].

This issue is computer dependent. Your results may be different for length of DMA achieved without making changes to the bridges between the PMC device and the system memory. BIOS can program the Bridges to operate and some may have FLASH attached to allow user "autoloading of parameters" into the configuration space control registers.

We are working on an autoloading utility for the bridges in the path and will soon offer that with our hardware products to solve the performance issue. We are also researching the PC Chip-set issue with the PCI master bridge to see what we can do to correct that as well.

Base Register Definitions

NG8_BASE_BASE

[$00 parallel-io 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 6 PMC-BISERIAL-III Control Base Register Bit Map

This is the base control register for the PMC BiSerial III NG8. 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^{2}C 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.

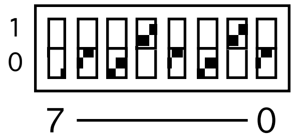
NG8_BASE_ID

[$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 7 PMC-BISERIAL-III 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 PMC BiSerial III 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 NG8 design is 0x0B 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.

NG8_BASE_STATUS

[$08 Board level Status Port read only]

DATA BIT	DESCRIPTION
31-12	Set to '0'
11-8	Reserved for PLL status
7-2	set to '0'
1	Masked Channel 1 Interrupt
0	Masked Channel 0 Interrupt

Figure 8 PMC-BISERIAL-III Status Port Bit Map

Channel Interrupt – The local masked 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 Register Definitions

Channel Bit Maps

The NG8 design has 2 channels. The basic control signals are the same for the channel base, channel status, FIFO and DMA interfaces. The following descriptions will be in the form of a common feature description for each address and then differences if any for each channel.

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

NG8_CHAN_CNTRL

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

Channel Control Register

Data Bit	Description
31-16	spare
15	Direction TX/RX
14	ExtFifoMux RX/TX
13	ExtFifoEn
12	ExtFifoLoad
11-9	Spare
8	OutUrgent
7	InUrgent
6	Read DMA Interrupt Enable
5	Write DMA Interrupt Enable
4	Force Interrupt
3	Channel Interrupt Enable
2	Bypass
1	RX FIFO Reset
0	TX FIFO Reset

Figure 9 PMC-BISERIAL-III channel Control Register

FIFO TX/RX Reset: When set to a one, the transmit and/or receive FIFO's will be reset. When these bits are zero, normal FIFO operation is enabled. In addition the TX and RX State Machine is 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 and master interrupt enables are 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.

ExtFifoEn: When cleared to a zero, the External FIFO will be reset. When set the External FIFO is enabled. The channels have external 128Kx32 FIFO's attached. Please note that the state of the Load pin how the part comes out of reset – what the default Almost Full and Almost Empty offsets are.

ExtFifoLoad: The external FIFO's have a LOAD pin which is tied directly to this register bit. The FIFO's in use are IDT72V36110. This is a dual purpose pin. During Master Reset, the state of the LD input determines the default offset values for the PAE and PAF flags selected. After Master Reset, this pin enables writing to and reading from the offset registers.

With ExtFifoLoad low during reset the default offset is 127. With ExtFifoLd high during reset the default offset is 1023. The offset is from the end of the FIFO. For example the PAE flag would be set to be 127, with the level of the FIFO 127 and less the PAE flag would be asserted. With 128K – 127 and more the PAF flag will be asserted.

In addition if the flags are to be reprogrammed using the parallel load feature the load pin should be low during reset Setting low allows writes to the FIFO to load new values into the FIFO. Taking Load high again puts the FIFO back into Data mode.

The FIFO's are configured with 32 in and 32 out. To parallel load the offset registers the PAE value is written first and the PAF value written second. Both values must be written to properly load. The values can be read back with the load pin low.

The data for the flags will flow through the TX DMA FIFO to the external FIFO. It is important to clear the FIFO's via reset prior to programming the offset registers to reset the pointer for the registers and to make sure the data written through the DMA FIFO is what is loaded into the External FIFO. Please also see ExtFifoMux control. The mux needs to be set to TX.

ExtFifoMux: bit controls if data is moved from the TX DMA FIFO or from the RX side into the External FIFO. '0' selects the TX path and '1' selects the RX path.

Direction: is used to select TX or RX operation for the IO. The direction bit is independent of the state-machines. Setting to 1 selects transmit operation and clearing to 0 selects RX operation for the transceivers assigned to the channel plus enables the terminations in the case of RX.

NG8_CHAN_STATUS

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

Channel Status Register

Data Bit	Description
31	Interrupt Status
30	Local Int
29	spare
28	Direction
27	Ext FIFO Full
26	Ext FIFO AFull
25	Ext FIFO AMT
24	Ext FIFO MT
23	Burst In Idle [write]
22	Burst Out Idle [read]
21	TX Idle State
20	RX Idle State
19	TX UnderFLow Err Lat
18	RX OverFlow Err Lat
17	RX Image Complete
16	TX Image Complete
15	Read DMA Interrupt Occurred
14	Write DMA Interrupt Occurred
13	Read DMA Error Occurred
12	Write DMA Error Occurred
11	RX AFull Int Lat
10	TX AMT Int Lat
9	RX AFull Int Lvl
8	TX AMT Int Lvl
7	spare
6	RX DMA FIFO Full
5	RX DMA FIFO Almost Full
4	RX DMA FIFO Empty
3	Spare
2	Tx DMA FIFO Full
1	Tx DMA FIFO Almost Empty
0	Tx DMA FIFO Empty

Figure 10 PMC-BiSerial-III Channel STATUS PORT

NG8 FIFO: Two 4K x 32 FIFO's are used to create the internal Transmit and Receive DMA FIFO's. The FIFO's are tied to the PCI bus to enable burst operations for DMA.

The status for the Transmit FIFO and Receive FIFO refer to these FIFO's. The status is active high. 0x13 would correspond to empty Transmitter and empty Master internal FIFO's.

Please note with the Receive 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.

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.

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 receive data FIFO is full; when a zero is read, there is room for at least one more data-word in the FIFO.

TX AMT Int Lvl: is set when the combined FIFO contents count is below the Almost Empty level total programmed into the reference register.

RX AFull Int Lvl: is set when the combined FIFO contents count is above the Almost Full Level Total programmed into the reference register.

TX AMT Int Lat: is set when the TX DMA FIFO contents count has been above the almost empty level and has transitioned below that level. This bit is a sticky bit and is held until cleared by writing back to this address with this bit set.

RX AFull Int Lat: is set when the RX DMA FIFO contents count has been below the almost full level and has transitioned above that level. This bit is a sticky bit and is held until cleared by writing back to this address with this bit set.

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.

TX Image Complete: This bit is set at the completion of the transmission of an image. This is a sticky bit and can be cleared by writing back with this bit position set. The interrupt can be used as a timer based on the image size or to help with flow control when using larger DMA transfers with multiple images.

RX Image Complete: This bit is set at the completion of the reception of an image. This is a sticky bit and can be cleared by writing back with this bit position set. The interrupt can be used as a timer based on the image size or to help with flow control when using larger DMA transfers with multiple images.

RX Over Flow: This bit is set if the RX SM FIFO is full when it is time to write another pixel pair. The data movers operate at a higher frequency than the receiver which means that the DMA and External FIFO’s will be backed up before this error can occur. This is a sticky bit and requires a write back with this bit set to clear.

TX Under Flow: This bit is set if the TX SM FIFO is empty when it is time to read another pixel pair. The data movers operate at a higher frequency than the transmitter which means that the DMA and External FIFO’s will be empty up before this error can occur. This is a sticky bit and requires a write back with this bit set to clear.

RX IDLE is set when the state-machine is in the idle state. When lower clock rates are used it may take a while to clean-up and return to the idle state. If SW has cleared the start bit to terminate the data transfer; SW can use the IDLE bit to determine when the HW has completed its task and returned.

TX IDLE is set when the state-machine is in the idle state. When lower clock rates are used it may take a while to clean-up and return to the idle state. If SW has cleared the start bit to terminate the transfer; SW can use the IDLE bit to determine when the HW has completed its task and returned.

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.

EXT FIFO MT: is set when the external FIFO is empty, when 0 at least 1 data is stored into the external FIFO.

EXT FIFO AMT: is set when the external FIFO is almost empty, when 0 more than the Almost Empty threshold is stored into the external FIFO. The threshold is programmable. Please refer to the channel control definition for more programming information.

EXT FIFO AFull: is set when the external FIFO is Almost Full, when 0 less than the Almost Full threshold is stored into the external FIFO. The threshold is programmable. Please refer to the channel control definition for more programming information.

EXT FIFO Full: is set when the external FIFO is full, when 0 at least 1 data position is available for new data to be stored into the external FIFO.

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.

NG8_CHAN_WR_DMA_PNTR

[0x08] 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 11 PMC-BiSerial-III 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.

NG8_CHAN_TX_FIFO_COUNT

[0x08] 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 12 PMC-BiSerial-III Transmit DMA FIFO data count Port

This read-only register port reports the number of 32-bit data words in the Transmit DMA FIFO. This design has 4095 locations possible. Unused bits within the count word are set to ‘0’. The spare bits are not driven and should be masked.

NG8_CHAN_RD_DMA_PNTR

[0x0C] 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 13 PMC-BiSerial-III 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.

NG8_CHAN_RX_FIFO_COUNT

[0x0C] RX [Master] FIFO data count (read only)

RX FIFO Data Count Port

Data Bit	Description
31-16	Spare
15-0	RX Data Words Stored

Figure 14 PMC-BiSerial-III Receive DMA 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 = 0x1003. The upper unused data bits within the word are set to '0' by HW. The spare bits are not driven and should be masked.

NG8_CHAN_FIFO

[0x10] Write TX/Read RX FIFO Port

RX [Master] and TX [Target] FIFO Port

Data Bit	Description
31-0	FIFO data word

Figure 15 PMC-BiSerial-III RX/TX FIFO Port

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

NG8_CHAN_TX_AMT_LVL

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

Target Almost-Full Level Register

Data Bit	Description
31-16	Spare
15-0	TX FIFO Almost-Empty Level

Figure 16 PMC-BiSerial-III Transmit ALMOST EMPTY LEVEL register

This read/write port accesses the almost-empty level register for the DMA FIFO. When the number of data words in the transmit DMA 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. 4k x32 is the TX FIFO for a 12 bit valid count range [11-0].

NG8_CHAN_RX_AFL_LVL

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

Master Almost-Full Level Register

Data Bit	Description
31-16	Spare
15-0	RX FIFO Almost-Full Level

Figure 17 PMC-BiSerial-III Receive ALMOST FULL LEVEL register

This read/write port accesses the almost-full level register. When the number of data words in the receive DMA FIFO is equal or greater than this value, the almost-full 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. 4k x32 + 4 is the Master FIFO size for a 13 bit valid count range [12-0]. The level includes the pipeline for an additional 4 locations [0xFFF + 4 = 0x1003].

NG8_CHAN_TX_CNTRL

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

Channel Control Register

Data Bit	Description
6	TxAeLvlIntEn
5	TxPixelOrder
4	TxUnFlEn
3	TxAeIntEn
2	TxIntEn
1	TxFifoEn
0	TxEn

Figure 18 PMC-BISERIAL-III Channel Transmitter Control Register

TxEn when set causes the Transmit State Machine to begin operation. The transmitter will wait until there is sufficient data in the FIFO pipeline to begin operation. This is a programmable number and can be adjusted based on image size and scheme of operation. For continuous transmission it is recommended to set this value to several images to allow the pipeline to be almost full before starting.

TxFifoEn when set enables the local Tx FIFO to read data from the external FIFO and transfer it to the Tx State Machine FIFO. Enable for TX operation, Disable for RX operation.

TxIntEn when set enables an interrupt to be generated for each image transmitted. The status and interrupt are set each time an image is completed.

TxAeIntEn when set enables an interrupt to be generated when the FIFO level has been more than Almost Empty and becomes Almost Empty. The status is a sticky bit and retains the information until cleared. The Status and interrupt is reasserted each time the transition occurs.

TxUnFlEn when set enables the interrupt from an underflow condition. The underflow status bit is set when the transmitter is ready to read the next pixel pair and the FIFO is empty. The status is a sticky bit and held until cleared. The underflow status is masked with TxUnFlEn to create the interrupt. Clear by clearing this bit or clearing the status [preferred].

TxPixelOrder when cleared uses the PCI standard for word ordering and when set reverses the word order. Each pixel is 1 word long. Two pixels are stored per long word transferred across the PCI bus. When cleared the data on D15-0 are used first and the data on D31-16 are used second.

TxAeLvlIntEn when set enables the interrupt based on the Almost Empty condition. When the interrupt occurs a programmable amount of data is left in the FIFO. The interrupt can be used to request a new DMA for a new image to be transferred or for other purposes. Larger DMA transfers spanning multiple images are recommended for continuous operation.

NG8_CHAN_TX_PIXEL_WORD_COUNT

[0x20] TX Pixels to send per image count

TX Pixels per Image Definition

Data Bit	Description
31-0	TX Pixels/2 per Image

Figure 19 PMC-BiSerial-III Transmit Pixel Count Port

This read-write register port holds the number of Pixels to transmit per image. Please note that the count is in Pixels/2 since it is used for the number of LW to transmit.

NG8_CHAN_TX_FIFO_WORD_COUNT

[0x24] TX State Machine FIFO data count (read only)

TX State Machine FIFO Level

Data Bit	Description
31-16	Spare
15-0	TX Data Count in SM FIFO

Figure 20 PMC-BiSerial-III TARGET DATA COUNT Port

The 1Kx32 FIFO between the External FIFO and the State-Machine is used for rate matching and local data storage for the TX state machine. The amount data stored in the FIFO can be read from this port. The clock reference to the FIFO is not the PCI clock. Reading this port while the level is changing can result in incorrect values. The values are correct when static. For example if the system is pre-filled before the TX side is enabled.

NG8_CHAN_TX_AMT_LVL_TOTAL

[0x28] TX Almost Empty Level Control Register

TX Interrupt Level

Data Bit	Description
31-0	Number of words in FIFO for AMT level check

Figure 21 PMC-BiSerial-III TX AMT Level Control Port

This register defines a comparison against the TX FIFO path. The TX DMA FIFO Plus the External FIFO plus the TX SM FIFO counts are added together and compared with the value programmed into this register. When the value of the count is less than the value of the register the AMT level interrupt can be asserted. The control register has the enable for the interrupt. The status register has the AMT Level Interrupt status.

NG8_CHAN_TX_READY_COUNT

[0x2C] TX Ready Count Control Register

TX Ready Count

Data Bit	Description
31-0	Number of words in FIFO for TX start

Figure 22 PMC-BiSerial-III TX Ready Count Control Port

This register defines a comparison against the TX FIFO path. The TX DMA FIFO Plus the External FIFO plus the TX SM FIFO counts are added together and compared with the value programmed into this register. When the value of the count is less than the value of the register the start of transmission is held off. The purpose is to allow the DMA to pre-fill the FIFO chain prior to starting transmission and to allow the software to perform one large DMA transfer instead of doing a prefill, start TX and then an additional DMA. The level should be set to match the size of the image data to send up to the total FIFO size. For 1 image set to a single image size, for two set to two etc. In the example case three images can be held within the FIFO chain.

The TX state machine also waits for the local FIFO to be almost full. The requirement for the local FIFO will put a lower bound on the image size of approximately 2K pixels. If your application requires smaller images please contact the factory. 2K in square form would be 45 x 45 including all visible and blanking areas.

NG8_CHAN_RX_CNTRL

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

Channel Control Register

Data Bit	Description
10	RxFifoPathEn
9-7	Spare
6	RxAflLvlIntEn
5	RxPixelOrder
4	RxOvFlEn
3	RxAfIntEn
2	RxIntEn
1	Spare
0	RxEn

Figure 23 PMC-BISERIAL-III Channel Receiver Control Register

RxEn when set causes the Receiver State Machine to begin operation. The Receive state machine will begin searching for the 0,0 Pixel with both HREF and VREF asserted. The image will begin capture with the 0,0 pixel and be loaded based on the received clock until stopped by software.

The received clock is sampled. If the clock is not present or stops the hardware will wait for the clock to continue. Pausing the clock can in effect pause the transfer without changing the pixel data transferred. At the end of the programmed image size, the hardware will look for 0,0. It is expected to be the next pixel received after the end of the previous image [including the blanking].

If the image size does not match the image received the following image may be skipped or the current image truncated depending on the direction of the mismatch.

RxIntEn when set causes an interrupt for each image captured. This interrupt can be used to read each image sequentially as they are received. Alternatively the programmable level interrupt can be used based on data within the FIFO. For continuous operation larger multi-image DMA transfers are recommended.

RxAfIntEn when set enables an interrupt to be generated when the FIFO level has been less than Almost Full and becomes Almost Full. The status is a sticky bit and retains the information until cleared. The Status and interrupt is reasserted each time the transition occurs.

RxOvFlEn when set causes an interrupt when the Rx FIFO overflows. The FIFO is considered to be in an OverFLow condition if it is full when it is time to write the next Pixel pair.

RxPixelOrder when cleared causes the state machine to load the lower [D15-0] data with the 0,0 pixel and upper [D31-16] with the second pixel received… When set the loading is reversed with the 0,0 Pixel loaded into D31-16 and the second into D15-0.

RxAflLvlIntEn when set enables the level based Almost Full Interrupt. This interrupt is asserted independent of transitions whenever the RX FIFO is almost full. Reading the data will reduce the level and remove the interrupt. The level can also be masked by clearing this control bit. Whenever the interrupt is asserted a DMA transfer of Almost Full size can be requested.

RxFifoPathEn when set enables the RX State-Machine FIFO to move data to the rest of the FIFO path. Data is enabled to be moved from the SM FIFO to the External FIFO and from the External FIFO to the DMA FIFO. Set when using the RX path. Clear **RxFifoPathEn** for TX operation. Set **RxFifoPathEn** for loop-back operation. Disable **RxEn** for loop-back operation.

NG8_CHAN_RX_PIXEL_WORD_COUNT

[0x38] RX Pixel Count of received images

RX Data Count Port

Data Bit	Description
31-0	RX Pixels/2 expected per image received

Figure 24 PMC-BiSerial-III RX Pixels Expected Port

This read-write register port holds the number of pixels expected to be received per image. Please note the total is Pixels/2 to account for packing into LW for storage and DMA transfer. When started the RX hardware will wait for HREF and VREF to be asserted and then load data based on the received clock until the Pixel Word Count is received. The Image Complete status is set and the HW begins looking for the sync pattern again. The hardware runs with a 10x clock and can handle the sync following on the next clock after the end of the previous image. This count needs to match the received image or inadvertent image filtering can result.

NG8_CHAN_RX_FIFO_WORD_COUNT

[0x3C] RX State Machine FIFO data count (read only)

RX State Machine FIFO Level

Data Bit	Description
31-16	Spare
15-0	TX Data Count in SM FIFO

Figure 25 PMC-BiSerial-III RX SM DATA COUNT Port

The 1Kx32 FIFO between the External FIFO and the State-Machine is used for rate matching and local data storage for the RX state machine. The amount data stored in the FIFO can be read from this port. The clock reference to the FIFO is not the PCI clock. Reading this port while the level is changing can result in incorrect values. The values are correct when static. For example if the system is disabled and the remaining data is to be read out, the total data register can be read or the individual FIFO counts can be read and used to determine the final transfer size.

NG8_CHAN_RX_AFL_LVL_TOTAL

[0x40] RX Almost Full Level Control Register

RX Interrupt Level

Data Bit	Description
31-0	Number of words in FIFO for AFL level check

Figure 26 PMC-BiSerial-III RX AFL Level Control Port

This register defines a comparison against the RX FIFO path. The RX DMA FIFO Plus the External FIFO plus the RX SM FIFO counts are added together and compared with the value programmed into this register. When the value of the count is greater than the value of the register the AFL level interrupt can be asserted. The control register has the enable for the interrupt. The status register has the AFL Level Interrupt status.

NG8_CHAN_EXT_FIFO_WORD_COUNT

[0x44] External FIFO Level Read Only register

External FIFO Level

Data Bit	Description
31-0	Number of words in FIFO

Figure 27 PMC-BiSerial-III RX AFL Level Control Port

This read only register has the count for the amount of data stored into the external FIFO. The reference clock for the external FIFO is not the same as the PCI clock. The count is re-clocked onto the PCI clock for this port. The values can jump due to the reclocking. When static the value will be true.

NG8_CHAN_TX_WORD_COUNT

[0x48] TX total FIFO Count

Total FIFO Data TX

Data Bit	Description
31-0	Combined TX Path FIFO Count

Figure 28 PMC-BiSerial-III TX Total FIFO Count

This read only register provides the total data count held in the TX FIFO path. Please note that the external FIFO overlaps with the RX path. Reading this port in RX mode will return the overlapping portion of the data and vice-versa.

NG8_CHAN_RX_WORD_COUNT

[0x4C] RX Total FIFO Count

Total FIFO Data RX

Data Bit	Description
31-0	Combined RX Path FIFO Count

Figure 29 PMC-BiSerial-III RX Total FIFO Count

This read only register provides the total data count held in the RX FIFO path. Please note that the external FIFO overlaps with the TX path. Reading this port in TX mode will return the overlapping portion of the data and vice-versa.

Loop-back

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

The test set-up included PCIBPMCX1, NG8, SCSI cable, and HDEterm68 to provide the loop-back. The Pin numbers are for the interconnections on the HDEterm68. The IO names can be used to accommodate a different set-up. Please note that the Pin assignments take into account the translation from Pn4 to SCSI to HDEterm68.

Signal	From	To	Signal
Chan0Pixel0-	pin 1	pin 17	Chan1Pixel0-
Chan0Pixel0+	pin 35	pin 51	Chan1Pixel0+
Chan0Pixel1-	pin 2	pin 18	Chan1Pixel1-
Chan0Pixel1+	pin 36	pin 52	Chan1Pixel1+
Chan0Pixel2-	pin 3	pin 19	Chan1Pixel2-
Chan0Pixel2+	pin 37	pin 53	Chan1Pixel2+
Chan0Pixel3-	pin 4	pin 20	Chan1Pixel3-
Chan0Pixel3+	pin 38	pin 54	Chan1Pixel3+
Chan0Pixel4-	pin 5	pin 21	Chan1Pixel4-
Chan0Pixel4+	pin 39	pin 55	Chan1Pixel4+
Chan0Pixel5-	pin 6	pin 22	Chan1Pixel5-
Chan0Pixel5+	pin 40	pin 56	Chan1Pixel5+
Chan0Pixel6-	pin 7	pin 23	Chan1Pixel6-
Chan0Pixel6+	pin 41	pin 57	Chan1Pixel6+
Chan0Pixel7-	pin 8	pin 24	Chan1Pixel7-
Chan0Pixel7+	pin 42	pin 58	Chan1Pixel7+
Chan0Pixel8-	pin 9	pin 25	Chan1Pixel8-
Chan0Pixel8+	pin 43	pin 59	Chan1Pixel8+
Chan0Pixel9-	pin 10	pin 26	Chan1Pixel9-
Chan0Pixel9+	pin 44	pin 60	Chan1Pixel9+
Chan0Pixel10-	pin 11	pin 27	Chan1Pixel10-
Chan0Pixel10+	pin 45	pin 61	Chan1Pixel10+
Chan0Pixel11-	pin 12	pin 28	Chan1Pixel11-
Chan0Pixel11+	pin 46	pin 62	Chan1Pixel11+
Chan0BadBit-	pin 13	pin 29	Chan1BadBit-
Chan0BadBit+	pin 47	pin 63	Chan1BadBit+
Chan0HRef-	pin 14	pin 30	Chan1HRef-
Chan0HRef+	pin 48	pin 64	Chan1HRef+
Chan0VRef-	pin 15	pin 31	Chan1VRef-
Chan0VRef+	pin 49	pin 65	Chan1VRef+
Chan0Clk-	pin 16	pin 32	Chan1Clk-
Chan0Clk+	pin 50	pin 66	Chan1Clk+

PMC Module Logic Interface Pin Assignment

The figure below gives the pin assignments for the PMC Module PCI Pn1 Interface on the PMC-BiSerial-III.   See the User Manual for your carrier board for more information. Unused pins may be assigned by the specification and not needed by this design.

-12V		1	2
GND	INTA#	3	4
		5	6
BUSMODE1#	+5V	7	8
		9	10
GND -		11	12
CLK	GND	13	14
GND -		15	16
	+5V	17	18
	AD31	19	20
AD28-	AD27	21	22
AD25-	GND	23	24
GND -	C/BE3#	25	26
AD22-	AD21	27	28
AD19	+5V	29	30
	AD17	31	32
FRAME#-	GND	33	34
GND	IRDY#	35	36
DEVSEL#	+5V	37	38
GND	LOCK#	39	40
		41	42
PAR	GND	43	44
	AD15	45	46
AD12-	AD11	47	48
AD9-	+5V	49	50
GND -	C/BE0#	51	52
AD6-	AD5	53	54
AD4	GND	55	56
	AD3	57	58
AD2-	AD1	59	60
	+5V	61	62
GND		63	64

Figure 30 PMC-BISERIAL-III Pn1 Interface

PMC Module Logic Interface Pin Assignment

The figure below gives the pin assignments for the PMC Module PCI Pn2 Interface on the PMC-BiSerial-III.   See the User Manual for your carrier board for more information. Unused pins may be assigned by the specification and not needed by this design.

+12V		1	2
		3	4
	GND	5	6
GND		7	8
		9	10
		11	12
RST#	BUSMODE3#	13	14
	BUSMODE4#	15	16
	GND	17	18
AD30	AD29	19	20
GND	AD26	21	22
AD24		23	24
IDSEL	AD23	25	26
	AD20	27	28
AD18		29	30
AD16	C/BE2#	31	32
GND		33	34
TRDY#		35	36
GND	STOP#	37	38
PERR#	GND	39	40
	SERR#	41	42
C/BE1#	GND	43	44
AD14	AD13	45	46
GND	AD10	47	48
AD8		49	50
AD7		51	52
		53	54
	GND	55	56
		57	58
GND		59	60
		61	62
GND		63	64

Figure 31 PMC-BISERIAL-III Pn2 Interface

PMC Module Rear IO Interface Pin Assignment

The figure below gives the pin assignments for the PMC Module IO Interface on the PMC-BiSerial-III. Installed for –RP models. Also see the User Manual for your carrier board for more information. Standard NG8 is –RP [Pn4]

IO_0p (Ch0Pixel0+)	IO_0m (Ch0Pixel0-)	1	3
IO_1p (Ch0Pixel1+)	IO_1m (Ch0Pixel1-)	2	4
IO_2p (Ch0Pixel2+)	IO_2m (Ch0Pixel2-)	5	7
IO_3p (Ch0Pixel3+)	IO_3m (Ch0Pixel3-)	6	8
IO_4p (Ch0Pixel4+)	IO_4m (Ch0Pixel4-)	9	11
IO_5p (Ch0Pixel5+)	IO_5m (Ch0Pixel5-)	10	12
IO_6p (Ch0Pixel6+)	IO_6m (Ch0Pixel6-)	13	15
IO_7p (Ch0Pixel7+)	IO_7m (Ch0Pixel7-)	14	16
IO_8p (Ch0Pixel8+)	IO_8m (Ch0Pixel8-)	17	19
IO_9p (Ch0Pixel9+)	IO_9m (Ch0Pixel9-)	18	20
IO_10p (Ch0Pixel10+)	IO_10m (Ch0Pixel10-)	21	23
IO_11p (Ch0Pixel11+)	IO_11m (Ch0Pixel11-)	22	24
IO_12p (Ch0BadBit+)	IO_12m (Ch0BadBit-)	25	27
IO_13p (Ch0HREF+)	IO_13m (Ch0HREF-)	26	28
IO_14p (Ch0VREF+)	IO_14m (Ch0VREF-)	29	31
IO_15p (Ch0CLK+)	IO_15m (Ch0CLK-)	30	32
IO_16p (Ch1Pixel0+)	IO_16m (Ch1Pixel0-)	33	35
IO_17p (Ch1Pixel1+)	IO_17m (Ch1Pixel1-)	34	36
IO_18p (Ch1Pixel2+)	IO_18m (Ch1Pixel2-)	37	39
IO_19p (Ch1Pixel3+)	IO_19m (Ch1Pixel3-)	38	40
IO_20p (Ch1Pixel4+)	IO_20m (Ch1Pixel4-)	41	43
IO_21p (Ch1Pixel5+)	IO_21m (Ch1Pixel5-)	42	44
IO_22p (Ch1Pixel6+)	IO_22m (Ch1Pixel6-)	45	47
IO_23p (Ch1Pixel7+)	IO_23m (Ch1Pixel7-)	46	48
IO_24p (Ch1Pixel8+)	IO_24m (Ch1Pixel8-)	49	51
IO_25p (Ch1Pixel9+)	IO_25m (Ch1Pixel9-)	50	52
IO_26p (Ch1Pixel10+)	IO_26m (Ch1Pixel10-)	53	55
IO_27p (Ch1Pixel11+)	IO_27m (Ch1Pixel11-)	54	56
IO_28p (Ch1BadBit+)	IO_28m (Ch1BadBit-)	57	59
IO_29p (Ch1HREF+)	IO_29m (Ch1HREF-)	58	60
IO_30p (Ch1VREF+)	IO_30m (Ch1VREF-)	61	63
IO_31p (Ch1CLK+)	IO_31m (Ch1CLK-)	62	64

Figure 32 PMC-BISERIAL-III Rear Panel Interface

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 SCSI II/III 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.

Terminal Block. We offer a high quality 68 screw terminal block that directly connects to the SCSI II/III cable. The terminal block can mount on standard DIN rails. HDEterm68
[http://www.dyneng.com/HDEterm68.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

PMC Modules were conceived and engineered for rugged industrial environments. The PMC-BiSerial-III 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 options for front panel and rear panel are isolated with series resistor packs to eliminate bus stubs when one of the connectors is not in use.

Surface mounted components are used.

The PMC Module connectors are keyed and shrouded with Gold plated pins on both plugs and receptacles. They are rated at 1 Amp per pin, 100 insertion cycles minimum. These connectors make consistent, correct insertion easy and reliable.

The PMC is secured against the carrier with the connectors and front panel. If more security against vibration is required the stand-offs can be secured against the carrier.

The PMC Module provides a low temperature coefficient of 2.17 W/^{O}C for uniform heat. This is based upon the temperature coefficient of the base FR4 material of 0.31 W/m-^{O}C, and taking into account the thickness and area of the PMC. The coefficient means that if 2.17 Watts are applied uniformly on the component side, then the temperature difference between the component side and solder side is one degree Celsius.

Thermal Considerations

The PMC-BISERIAL-III design consists of CMOS 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; forced air cooling 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:	PMC Logic Interface [PCI] 32/33
Digital Parallel IO:	485 IO with NG8 protocol. Camera interface with HREF, VREF, CLK, BadBit, Pixels[11-0]
CLK rates supported:	PLLA is programmed to select Transmit Clock rate. 10 MHz is standard interface rate. PLLB is set to 10x the expected Rx rate.
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 120 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:	Rear IO via Pn4 standard. SCSI III connector at front bezel by special request.
Dimensions:	Standard Single PMC Module.
Construction:	Multi-Layer Printed Circuit, Through Hole and Surface Mount Components.
Temperature Coefficient:	2.17 W/°C for uniform heat across PMC
Power:	TBD mA @ 5V

Order Information

standard temperature range Industrial
PMC-BiSerial-III-NG8 PMC Module with 2 channels. Either Transmit or Receive function per channel under software control.
http://www.dyneng.com/pmc_biserial_III.html

Order Options:

Pick One
–FP for front panel IO only
-RP for rear panel IO PN4 only [default if no selection made]
-FRP for both IO connections
Shown for reference. NG8 selection determines [-RP]

Pick any combination to go with IO
-CC to add conformal coating
-ET to change to industrial Temp [-40 - +85C]
-COM to change to commercial temp parts [0-70] NG8 defines this option as standard
-TS to add thumbscrew option – standard is latch block at SCSI connector

Related:
PCIBPMCX1: PCI to PMC adapter to allow installation of PMC-BiSerial-III into a PCI system with differential, matched length, impedance controlled Pn4 IO.
http://www.dyneng.com/pcibpmcx1.html

PCIeBPMCX1: PCIe to PMC adapter to allow installation of PMC-BiSerial-III into a PCIe system.
http://www.dyneng.com/pciebpmcx1.html

HDEterm68: 68 position terminal block with two SCSI II/III connectors. PMC-BiSerial-III compatible. Differentially routed break-out for SCSI cabled systems.
http://www.dyneng.com/HDEterm68.html

HDEcabl68: SCSI II/III cable compatible with NG8 IO when mounted on a carrier with Pn4 ⇔ SCSI interconnect. http://www.dyneng.com/HDEcabl68.html

PMC BiSerial III Eng Kit : HDEterm68-MP, HDEcabl68, Windows Driver software, reference schematics. Recommended for first time purchases.
http://www.dyneng.com/pmc_biserial_III.html