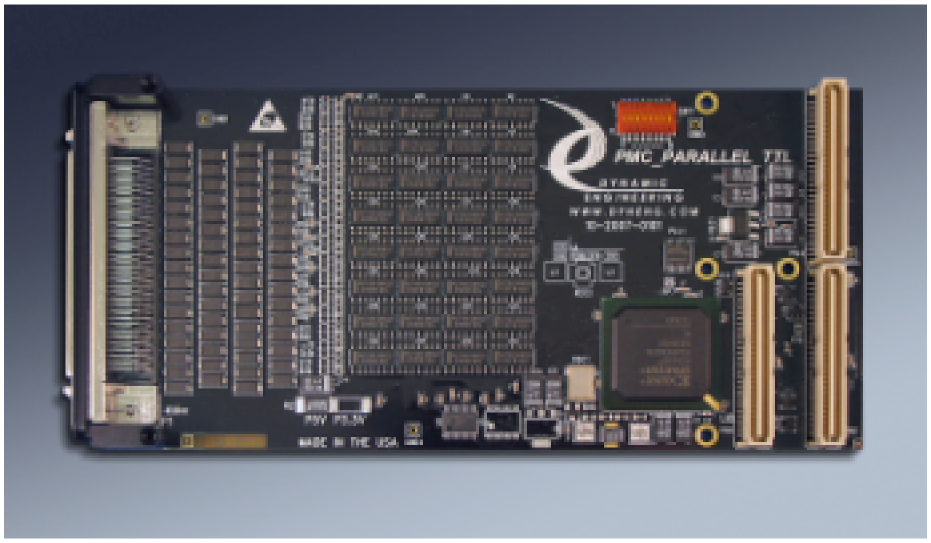
DYNAMIC ENGINEERING
150 DuBois St. Suite C, Santa Cruz, Ca 95060
831-457-8891 Fax 831-457-4793
http://www.dyneng.com
sales@dyneng.com
Est. 1988

User Manual

PMC-PARALLEL-TTL-BA21

Digital Parallel Interface
PMC Module
4 - I^2C Ports



Manual Revision: A1 Revised: 4/3/13
Corresponding Hardware: Revision 1
10-2007-0101 FLASH 0401

PMC-PARALLEL-TTL-BA21

Digital Parallel Interface
PMC Module

Dynamic Engineering
150 DuBois St. Suite C, Santa Cruz CA 95060
831-457-8891 831-457-4793 FAX

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.

Table of Contents

List of Figures

Product Description

In embedded systems many of the interconnections are made with single ended TTL or CMOS level signals. Depending on the system architecture an IP or a PMC will be the right choice to make the connection. With most architectures you have a choice as there are 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 architectures, and is adding more as new solutions are requested and required by our customers.

PMC compatible PMC-Parallel-TTL has 64 independent digital IO. The high density makes efficient use of PMC slot resources. The IO is available for system connection through the front panel, via the rear [Pn4] connector, or both. A high density 68 pin SCSI III front panel connector provides the front panel IO. The IO lines can be protected with optional transorbs. The rear panel IO has a PIM and PIM Carrier available for rear panel wiring options.

PMC-Parallel-TTL-BA21 is a customized version of the standard PMC-Parallel-TTL board. “BA21” is set to 3.3V, has front panel IO, and a specific state-machine implementation for four channels of I2C protocol IO. The transmit rate is programmable. The standard and High Speed I2C rates are supported. A PLL is used with a separate reference for each of the 4 channels allowing for the normal i2c rates as well as custom ones. 2KΩ pull-ups are in place to allow for a second 2K pull-up at the opposite end of the connection. If multi-tap cabling is used the total external pull-up should be 2K to meet the 1K pull-up standard. The BA21 can operate with “stiffer pull-ups” but other equipment may have trouble.

Each channel has a separate 1Kx32 FIFO for TX and RX plus an additional FIFO for Packet length storage. The TX and RX memories can be accessed as standard target accesses or with DMA. The packet FIFO is read only. Counts and status flags are provided to manage the memory.

With 64 IO and 4 channels of I2C several IO are available for other purposes. A 32 bit Parallel port is supplied to make use of some of the remaining IO. The port is register based with separate direction control for each bit.

The HDEterm68 http://www.dyneng.com/HDEterm68.html
can be used as a breakout for the front or rear panel IO. The HDEcabl68 provides a convenient cable. http://www.dyneng.com/HDEcabl68.html Custom cables can be manufactured to your requirements. 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]. The design with revision 2 and later FLASH is DMA capable with a built in programmable parallel data output and input function. The new features are designed to default to “not used” to allow the new cards to be used with older customer software. The DMA function can be used with customer requirements too. Please contact Dynamic Engineering with your custom requirements. BA21 is design number 2 for the PMC-Parallel-TTL with a corresponding FLASH of 02xx.

The IO are driven with open-drain high current drivers. When enabled, the high side is driven with the device and augmented with pull-up resistors. When disabled the output is pulled high with the resistors unless another device on the line is driving that line low. The low side of the driver can sink 64+ mA. The high side drive is 32 mA plus the pull-up current value. All IO have 2 pull-up locations per line. The default is for 470 ohms installed into one location. The resistors are referenced to either 5V or 3.3V based on a factory installed jumper. The multiple locations allow for pull-up strengths greater than 470 ohms, and to stay within the resistor pack wattage capabilities. The multiple packs also allow for parallel combinations to create more options of specific pull-up values. For custom models with additional pull-ups or alternate values please contact Dynamic Engineering. The two columns of pull-up resistor locations are visible on the rear of the card.

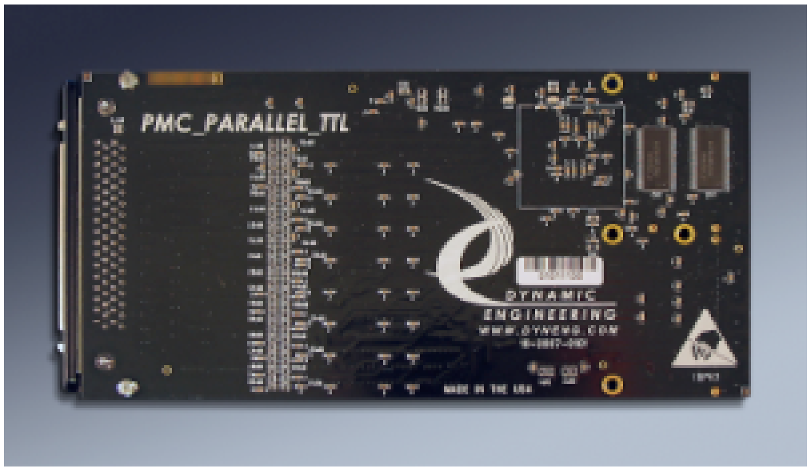


Figure 1 PMC-PARALLEL-TTL REAR VIEW

The registers are mapped as 32 bit words and support byte, word and 32 bit access. All registers are read-writeable. The Linux and Windows® compatible drivers are available to provide the system level interface for this design. 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. The Linux documentation is provided in-line with the source code.

The basic functions of parallel IO and COS capture are designed into the FLASH 0101 “base” model. Additional features will be added to the base model by using a mux on the output side to allow software to select the base or extended features. Data bit 0 is the first extended feature and is a programmable output for the COS reference clock. With software the output definition can be changed to drive the COS clock onto Data 0. The user can use a scope to check that their set-up is what they want it to be, and then likely return it to being a data bit. You can leave the output defined as a clock if desired.

FLASH 0201 has the additional feature of utilizing the internal block RAM to create FIFO’s to support output and input data streams. An additional register is added to allow the user to select on a bit-by-bit basis the programmable data output or the register based output. The input function does not preclude the use of the standard input functions. For example COS can be run on the same inputs as the data capture uses.

PMC-PARALLEL-TTL is part of the PMC Module family of modular I/O components. The PMC-PARALLEL-TTL 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.

Theory of Operation

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

PMC-PARALLEL-TTL features a Xilinx FPGA, and high current LVTH driver devices. The FPGA contains the PCI interface and control required for the parallel interface.

I2C is a serial protocol with a clock and data line. The clock is gated. The data is stable during the clock rising and falling edges changing only when the clock is low except for the Start and Stop sequence. BA21 handles all of the I2C protocol in a state-machine. This allows software to set and forget rather than doing all of the “bit banging” that I2C usually requires. With a HW approach the limited bandwidth of the I2C bus can be maximized as the HW is much more precise than a register based SW method can be.

Write, Read and Snoop are the three functions provided for the I2C bus. There are 4 channels each with a separate memory and state-machine to allow multiple transfers in parallel or multiple signals to be “snooped”.

The PLL has four outputs A-D. A is used for Channel 0 … D on Channel 3. The PLL is programmed to create the reference for the channels to use. The clock is set to 32 MHz for the HS mode and 8 Mhz for the Standard speed I2C. The Windows driver has features to program the PLL from a .JED file. Reference files are supplied and can be modified using the Cypress 22393 tool – free from Cypress. “cyberclocks”. When launching the ATP, the “main” program initializes the PLL to 8 Mhz on all channels. Test 2 in the ATP code allows a new file name to be selected within the user interface and loaded into the PLL. For application code the file can be “hardwired” into the code to prevent the user interaction requirement or the data matrix can be parsed the way we do in our initialization SW. Dynamic Engineering has a utility to generate the data matrix from a .JED file should you prefer that method. Please contact Dynamic Engineering for a copy of this utility. sales@dyneng.com

To set-up, the channel control register is written to with the enable for the state-machine set, and the mode [Master/Target] selected. If the channel is to operate as a Target the address is also set-up. If set or reset to “00” the channel will be in snoop mode once enabled.

As a master, the transmit FIFO is monitored. When the FIFO is written to – either DMA or standard PCI target access, the first word is read by the state-machine and parsed. The first word contains the mode on bit 0, address on 7-1, length on 15-8 [1-255], and the first byte or two to transfer. If multiple bytes – 3 or more are to be transferred the SW will need to make sure the data is in the FIFO for the 2nd LW before the end of the processing of the 2nd byte or an underflow condition will be detected. If your system timing is tough to manage it is suggested to disable the SM, load the FIFO and then enable the SM. A status bit for the idle condition is available to allow SW to know when the SM has responded to the disable, With a DMA transfer this situation is usually not an issue since it is HW controlled, Our testing has been conducted using DMA for multi-word messages without underflow issues.

The State-machine will parse the message and write or read based on bit 0. In either case the address and R/W are transmitted. An ACK is looked for from the Target. If a Write the data is then transmitted with the ACK being checked after each byte. Clocking is continuous until the message is completed. If a read is implemented, clocks are generated without data after the address. Data is captured during the high portion of the clock cycle, and the Master asserts the ACK until the last byte where a NAK is asserted.

ACK is a low state on the line during the high portion of the clock in the 9th bit period for address/mode or data transfers. NAK is a high during the high portion of the clock. When a Master read is processed, the data is stored into the RX side FIFO. Data can be read with PCI Target Accesses or with DMA. To support this operation the FIFO count is provided. Since it is the Master, the packet length is already known.

When acting as a Target and data is written, the packet length can be used. The packet count is in Bytes received. The count is terminated when a stop or new start is received at the end of a byte after the ACK is asserted. A separate FIFO is provided with the Packet Lengths to allow the FIFO count to be used to empty the FIFO and the Packet Length used to parse the messages.

Since the packet count is in bytes and matches the actual size of the message, and the FIFO count is in LW the counts may seem to differ.

Data is always operated on using LW's for the overall size. Received messages are padded with 0x00 for any unused locations in the last LW.

The reference SW has loop tests for Write, Read, and Snoop situations.

The difference with Snoop is that the address is also stored and all data on the bus is stored for any non zero address.

For the parallel port, data written to the IO registers can be placed on the bus. The drivers are initialized to the off state and pull-ups on board hold the IO lines in the 'high' state. The direction registers are used to program the channel to be a driver or not. The receivers are always enabled allowing local read-back of the transmitted data.

For an IO with the direction bit set and master enabled: When a '0' is written to any IO line register position the corresponding line is driven low. When a '1' is written to any IO line register position that line is driven high by the local driver, and the output level will be controlled by the termination resistor. The drivers are asymmetrical with 64 mA sink and 32 mA source. The 470 resistor to 3.3/5 will provide additional "source current", and level control when in "open drain" mode [programmed for receive].

If the direction bit is set to input the level will be controlled by external devices and the attached pull-ups. The control register is read-writeable. The data register read corresponds to the IO side. The register read-back is at an alternate address offset. The register read-back is independent of the bus; the data read will always match the data written. The IO data read will reflect the state of the bus and not necessarily the state of the on-board drivers.

The read-back registers are clocked at a programmable rate with an internal clock generator. If desired the internal clock can be replaced with an external source and an enable. The basic option is available under SW control. If special programming is needed please contact Dynamic Engineering for a custom FPGA implementation.

All the IO control and registers are instantiated within the FPGA, only the drivers and receivers are separate devices. If desired, the IO lines can be specially programmed to create custom timing pulses etc. For example if the interface is to put out an address and then an address qualifier to strobe the address into the receiving hardware one of the IO lines can be programmed to create a pulse some time after the address for the IO registers is written to. The custom pulse will be more accurate for delay and duration than a SW timing solution. The number of accesses to the card can be reduced as well having the effect of greater through-put. Please contact Dynamic Engineering with your requirements.

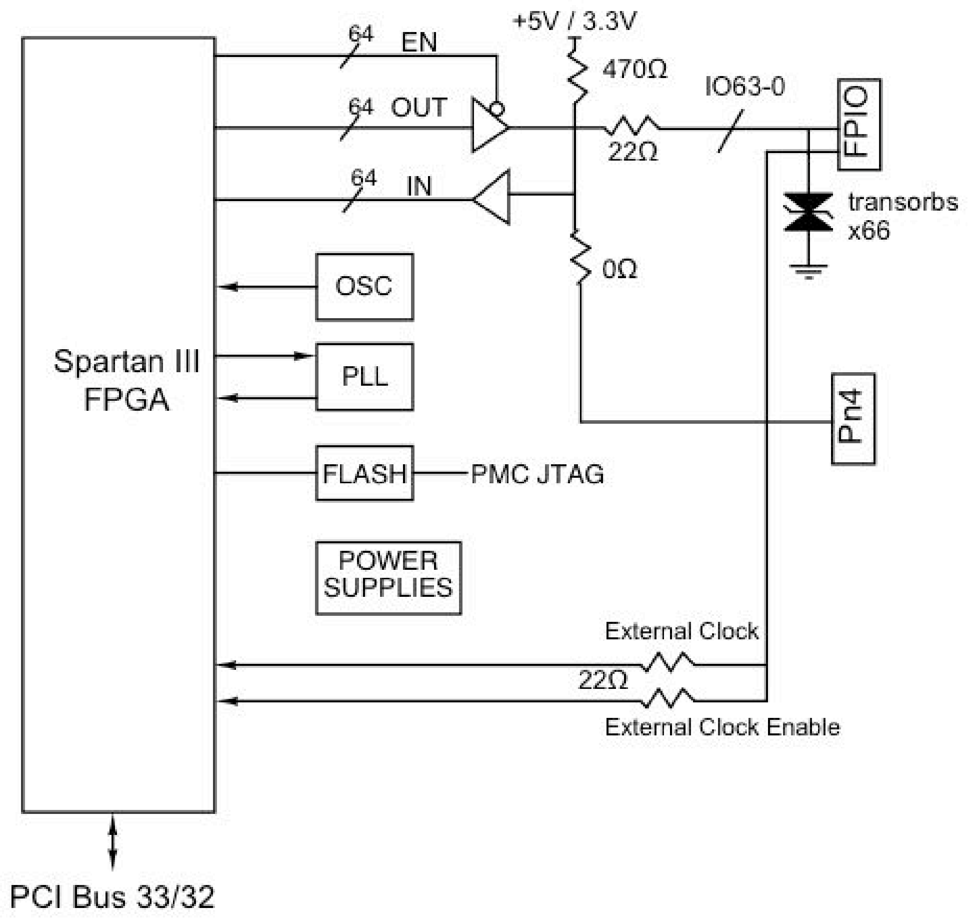


Figure 2 PMC-PARALLEL-TTL Block Diagram

PMC Parallel TTL BA21 features a programmable data path with DMA support. The internal block RAM is configured to provide FIFO's for transmit and receive [1K x 32 RX, 1Kx32 TX] per channel. The PLL is used as a TX state-machine reference. The FIFO's support a loop-back path between them for BIT [built in testing].

The hardware will pull data from the host memory and store into the transmit FIFO. The FIFO will be kept full with DMA operating at the PCI bus frequency. The output side will operate at the programmed rate [400 KHz for example] . The state-machine will take care of reading from the FIFO, and outputting on the programmed frequency. With I2C it is likely that the messages will be shorter than the FIFO length allowing multiple messages to be stored while others are being transmitted. Packet based interrupts are available along with DMA, and FIFO level to provide a variety of programming options.

The state-machine supports length variations on a per packet basis as well as switching READ/WRITE operations on a per packet basis. The transmission will continue until no packets are left to transmit. With 400 KHz. I2C, the DMA is operating at 33 MHz x 32 bits and the IO is 1 bit and 400 KHz for a really large multiplier. In addition the DMA FIFO is 1Kx32 for TX leaving a lot of "rubber band" in the memory chain to support the transmission.

The math is the same for receiving data.   DMA can be pre-programmed or set-up based on a Packet Received interrupt.

The DMA 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 slightly more efficient as there are potentially fewer initialization reads and less 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 PLL clock is reduced to an 8x clock for transmission to allow for the proper granularity to control the output cycle by cycle. At 100 KHz the PLL is programmed to 8 and divided to 1. At 1 MHz a 1 uS granularity is available to provide the set-up, hold and meet timing for Start, Stop and so forth. The PLL clock is used as-is for the receive side to provide a 64x receive clock. The edges of the signals can be found accurately and variances in the received clock can be handled.

The 8x clock is needed for the output to meet all of the timing requirements.   8x is not enough for the receive side and an "easy" power of 2 was selected above that rate to do the receiver side.

The FPGA easily meets the 32 MHz timing requirement. To use the same relative timing as the HS mode at a higher or lower rate adjust the PLL frequency accordingly. On the high side it would be good to recompile to check timing against your requirement. Please contact Dynamic Engineering for this service.

There are modes faster than HS but the timing and address definitions typically change. The state-machine could be altered to provide 1 or more channels handling the alternate I2C definitions.   A 16x clock would likely be used for the faster modes to reduce the upper frequency required.

Address Map

```
Function                                    Offset
// PMC Parallel TTL BA21 definitions
#define BA21_BASE_BASE          0x0000 // 0  PMC Parallel TTL base control register offset
#define BA21_BASE_ID            0x0004 // 1  PMC Parallel TTL ID Register offset
#define BA21_BASE_STATUS        0x0008 // 2  PMC Parallel TTL status Register offset

#define BA21_BASE_DATAEN        0x0010 // 4  PMC Parallel TTL Data Output Enable Register offset
#define BA21_BASE_DATAIO        0x0014 // 5  PMC Parallel TTL Data IO Register, line data read
#define BA21_BASE_DATAREG       0x0018 // 6  PMC Parallel TTL Data Register Read Only
```

Figure 3 PMC-PARALLEL-TTL Internal Address Map Base Functions

The address map provided is for the local decoding performed within PMC-Parallel-TTL-BA21. The addresses are all offsets from a base address. The carrier board 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 = 0xDCBA and the CardId = 0x004D for the PMC-Parallel-TTL-BA21.

```
Function                                        Offset
// PMC Parallel TTL BA21 definitions
#define BA21_CHAN_BASE        0x0000 // 0  Chanel control register
#define BA21_CHAN_STATUS      0x0004 // 1  Channel status, interrupt clear
#define BA21_CHAN_WR_DMA      0x0008 // 2  Channel burst in control
#define BA21_CHAN_TX_CNT      0x0008 // 2  Channel Read TX FIFO Count
#define BA21_CHAN_RD_DMA      0x000C // 3  Channel burst out control
#define BA21_CHAN_RX_CNT      0x000C // 3  Channel Read Rx FIFO Count
#define BA21_CHAN_SW(R/W)     0x0010 // 4  Channel FIFO single read RX, single write TX FIFO
#define BA21_CHAN_tx_aecnt    0x0014 // 5  Channel almost empty count register and rd-bk
#define BA21_CHAN_rx_afcnt    0x0018 // 6 Channel almost full count register and rd-bk

#define BA21_CHAN_TARGET      0x0034 // 13  Channel Target Control Register

#define BA21_CHAN_PCK_CNT     0x004C // 19  Channel Packet Count read port
```

Figure 4 PMC-PARALLEL-TTL Channel Address Map

There are 4 channels in the BA21 design. Each with a common address map relative to the channel offset. The channel offset is added to the base offset.
Channel 0 = 0x50
Channel 1 = 0XA0
Channel 2 = 0XF0
Channel 3 = 0x140

Programming

Programming the PMC-PARALLEL-TTL-BA21 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-Parallel-TTL-BA21 card, software will need to determine what the address space is for the PCI interface [BAR0]. The offsets in the address table are relative to the system assigned BAR0 base address.

The next step is to initialize the PMC-Parallel-TTL-BA21.

To use the parallel port, enable the bits intended to be outputs by writing a ‘1’ to each bit position. Next write the output data to the IO port and / or read the input bits also from the IO port. The transmit bits are also present for IO read functions. To read the value of the transmit register read the DATAREG port.

To use the I2C functions the PLL will need to be programmed to provide the reference rate desired for each channel. 8 MHz or 32 MHz for Standard and HS modes respectively. If interrupts will be used the master interrupt enable will need to be set.

Within each channel the Control register is used to select the mode of operation – Master or Target, and to enable that operation. If a Target, the TARGET register will need to be programmed with the channel ID prior to enabling the channel.

As a master, a write is started by loading the TX FIFO with the address, RW = 0, Data byte count, and the first byte or two of data. The hardware handles the processing and provides status to indicate when complete.

As a master, a read is also started by loading the TX FIFO with the address, RW= 1, and data byte count. The data is “don’t care” in this mode. The HW will handle the reads up to the programmed count and return the data in the RX FIFO.

For Windows™, Linux, and VxWorks systems the Dynamic Driver 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. Please note: as of 3/14/13 the Win32 driver package is completed and the others are planned.

To use the BA21 specific functions the Channel Control, DMA/Reg and Direction registers 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 BA21 to start the DMA. Please refer to the Burst IN and Burst Out register discussions.

Register Definitions

BA21_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
15-1	spare
0	Master Interrupt Enable

Figure 5 PMC-PARALLEL-TTL Control port 0 Bit Map

This is the base control register for the PMC Parallel TTL. 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.

Master Interrupt Enable when '1' gates active interrupt requesting conditions onto Interrupt Request A. When set to '0' the interrupting functions are available as status but no interrupt request is generated by the card to allow for polled operation.

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.

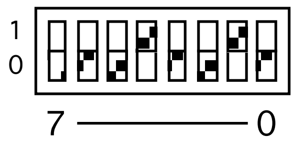
BA21_BASE_ID

[$04 Switch and Design number port read only]

DATA BIT	DESCRIPTION
31-24	spare
23-16	Design Number
15-8	Revision
7-0	DIP switch

Figure 6 PMC-PARALLEL-TTL 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 Parallel TTL is which 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 BA21 design is 0x04 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.

BA21_BASE_STATUS

[$08 Board level Status Port read only]

DATA BIT	DESCRIPTION
31-4	spare
3-	Channel 3 Int
2	Channel 2 Int
1	Channel 1 Int
0	Channel 0 Int

Figure 7 PMC-PARALLEL-TTL Status Port Bit Map

Channel XX Int – This is the local masked interrupt from channel XX. This bit will tell the SW if any channel XX asset could be requesting an interrupt. If the master interrupt enable is set an interrupt will be generated if this condition is true.

BA21_BASE_DATAEN

[$10 Enable Register bits 31-0 read – write]

DATA BIT	DESCRIPTION
31-0	DataEn31-0

Figure 8 PMC-PARALLEL-TTL Data Enable Bit Map

The 32 bits of the parallel port direction are controlled with this port. When reset this port is cleared 0x00000000. All IO are set to read [inputs]. To use one or more of the IO for outputs; program the corresponding enable bit(s) to '1'.

BA21_BASE_DATAIO

[$14 Data Register bits read – write]

DATA BIT	DESCRIPTION
31-0	DataIO 31-0

Figure 9 PMC-PARALLEL-TTL Data Register Bit Map

If the corresponding Enable bit is set, the data bit is driven onto the IO line. When read the state of the IO lines is read. Use the DATAREG for read-back of the register value.

BA21_BASE_DATAREG

[$18 Data Register bits read only]

DATA BIT	DESCRIPTION
31-0	DataIO 31-0

Figure 10 PMC-PARALLEL-TTL Data Register Read Back

Use DATAREG for read-back of the register value set in the output register. This value will always match what was written. The line value read back from the DATAIO register may not match as other devices may affect the IO values read.

BA21_CHAN_BASE

[0x50, A0, F0, 140] Channel Control Register (read/write)

Channel Control Register

Data Bit	Description
20	RxFfAFlIntEnLvl
19	TxFfAMtIntEnLvl
18	RxOvFlIntEn
17	TxUnFlIntEn
16	RxFfIntEn
15	TxFfIntEn
14	PacketIntEn
13	I2CMaster
12	I2CEnable
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 Enable
1	Receiver Reset
0	Transmit Reset

Figure 11 PMC-PARALLEL-TTL Channel Control Register

Transmit Reset: When set to a one, the transmit FIFO will be reset. When ‘0’ normal FIFO operation is enabled. In addition the TX and RX State Machine is reset.

Receiver Reset: When set to a one, the receive and Packet FIFO’s will be reset. When ‘0’ normal FIFO operation is enabled. In addition the TX and RX State Machine is reset.

Bypass Enable: When this bit is set to a one, any data written to the transmit FIFO will be transferred to the receive FIFO. This allows for fully testing the data FIFO’s without using the I/O. When this bit is zero, normal FIFO operation is enabled. The rate at which the data is transferred depends on the IO clock selection [PLL]. DMA can be used with Bypass mode.

Write/Read DMA Interrupt Enable: These two bits, when set to one, enable the interrupts for DMA writes and reads respectively. The DMA interrupts are not affected by the Master Interrupt Enable at the channel level.

Channel Interrupt Enable: When this bit is set to a one, all enabled interrupts (except the DMA interrupts) will be gated through to the Base 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. An additional board level master interrupt enable is located in the Base register. The board level master must also be enabled to gate the interrupt through to the host.

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.

I2CEnable: When set ‘1’ will start the I2C function. The PLL should be programmed to the desired frequency ahead of enabling the state-machine. I2CEnable is delayed 1 clock at the state-machine to allow setting or clearing the Master bit at the same time without creating a race condition at the state-machine. The Target address parameter should also be set prior to enabling the state-machine.

I2CMaster is set to select Master operation and cleared to act as a Target.

PacketIntEn: When set ‘1’ will enable an interrupt request when a Packet is completed. The Packet interrupt occurs at the end of a transmission / read as a Master and when the Stop or Restart is detected as a Target/Snoop mode device.

TxFfIntEn: When set ‘1’ will enable an interrupt request when the Tx FIFO has become Almost Empty. This version occurs when the condition of not almost empty transitions to almost empty.

TxFfAMtIntEnLvl: When set ‘1’ will enable an interrupt request when the Tx FIFO has become Almost Empty. This version occurs when the condition of almost empty is true. The effect is an interrupt that is continuous unless the condition is changed to be not almost empty.

RxFfIntEn: When set ‘1’ will enable an interrupt request when the Rx FIFO has become Almost Full. This version occurs when the condition of not almost full transitions to almost full.

RxFfAFlIntEnLvl: When set ‘1’ will enable an interrupt request when the Rx FIFO has become Almost Full. This version occurs when the condition of almost full is true. The effect is an interrupt that is continuous unless the condition is changed to be not almost full.

If the SW is trying to keep the SM processing and using the AMT/AFL interrupt to refill / Empty the FIFO then the level version would be a good choice when the IO rate is high compared to the refill rate. With this design it is unlikely that the IO will outpace the load function.

RxOvFlIntEn: When set ‘1’ will enable an interrupt request when the Rx FIFO is full when it is time to load another word.

TxUnFlIntEn: When set ‘1’ will enable an interrupt request when the Tx FIFO is empty when it is time to read another word.

BA21_CHAN_STATUS
[0x54,A4,F4,144] Channel Status Read/Clear Latch Write Port

Channel Status Register

Data Bit	Description
31	Channel Interrupt Active
30	Local Int
29-26	spare
25	PacketFifoFl
24	PacketFifoMt
23	BurstInIdle
22	BurstOutIdle
21	spare
20	SmIdleState
19	TxUnFlLat
18	RxFfOvFlLat
17	NAKerrLat
16	PacketDoneLat
15	Read DMA Interrupt Occurred
14	Write DMA Interrupt Occurred
13	Read DMA Error Occurred
12	Write DMA Error Occurred
11	RxAFlIntLat
10	TxAMtIntLat
9	RxAFlIntLvl
8	TxAMtIntLvl
7	spare
6	Receive FIFO Full
5	Receive FIFO Almost Full
4	Receive FIFO Empty
3	spare
2	Transmit FIFO Full
1	Transmit FIFO Almost Empty
0	Transmit FIFO Empty

Figure 12 PMC-Parallel-TTL Channel STATUS PORT

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

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

Transmit 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.

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 is the combined FIFO and pipeline value and can be used for read size control.

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

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

Receive 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.

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

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

TxAmtIntLvl: When a one is read, the transmit FIFO almost empty is asserted. This is a level based interrupt. The status is before the mask to allow for non-interrupt driven SW operation. When the FIFO level is below the programmed level this bit is asserted.

TxAmtIntLat: When a one is read, the transmit FIFO almost empty has been asserted. This is a triggered interrupt. The status is before the mask to allow for non-interrupt driven SW operation. When the FIFO level has dipped below the programmed level the status is captured and held. Clear by writing back to this bit.

RxAFlIntLvl: When a one is read, the receive FIFO almost full is asserted. This is a level based interrupt. The status is before the mask to allow for non-interrupt driven SW operation. When the FIFO level is above the programmed level this bit is asserted.

RxAFlIntLat: When a one is read, the receive FIFO almost full has been asserted. This is a triggered interrupt. The status is before the mask to allow for non-interrupt driven SW operation. When the FIFO level has risen the programmed level the status is captured and held. Clear by writing back to this bit.

PacketDoneLat: When a one is read, it indicates that the state-machine has finished processing a packet. A zero indicates that a packet has not been completed. This bit is latched and can be cleared by writing back to the Status register with a one in this bit position.

NAK Interrupt Occurred: When a one is read, it indicates that the NAK has occurred when it was not expected. A zero indicates that all bytes were properly ACK'd. This bit is latched and can be cleared by writing back to the Status register with a one in this bit position.

RxFfOvFl Interrupt Occurred: When a one is read, it indicates that the Rx FIFO has suffered an overflow condition; defined as being full when a write is attempted. A zero indicates proper operation. This bit is latched and can be cleared by writing back to the Status register with a one in this bit position.

TxFfUnFl Interrupt Occurred: When a one is read, it indicates that the Tx FIFO has suffered an underflow condition; defined as being empty when a read is attempted. A zero indicates proper operation. This bit is latched and can be cleared by writing back to the Status register with a one in this bit position.

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.

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 TX and burst data out of the card for Receive.

SmIdleState is set when the I2C state-machine is in the idle state. If SW has cleared the enable bit SmIdleState can be used to determine the HW has completed its task and returned.

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.

Channel Interrupt Active: When a one is read, it indicates that a system interrupt is potentially asserted caused by an enabled channel interrupt condition. A zero indicates that no system interrupt is pending from an enabled channel interrupt condition. The Board level master interrupt enable will also need to be asserted to allow the active channel interrupt to become an interrupt request.

Local Interrupt: When a one is read, it indicates that any of the masked conditions other than DMA are active and enabled. Local Interrupt is or'd with the DMA interrupt sources to create the Channel Interrupt Active signal and to request the Interrupt.

The 4 channel interrupts are masked with the Master Interrupt Enable and driven to INTA on the PCI bus. System routing may re-route INTA onto another level by the time the CPU "sees" it. Your discovery software should handle the interrupt assignment for this card as part of initialization.

BA21_CHAN_WR_DMA

[0x58,A8,F8,148] Write DMA Pointer (write only)

DMA Pointer Address Register

Data Bit	**Description**
31-2	First Chaining Descriptor Physical Address
1	direction [0]
0	end of chain

Figure 13 PMC-Parallel-TTL 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.

BA21_CHAN_RD_DMA

[0x5C,AC,FC,14C] Read DMA Pointer (write only)

DMA Pointer Address Register

Data Bit	**Description**
31-2	First Chaining Descriptor Physical Address
1	direction [1]
0	end of chain

Figure 14 PMC-Parallel-TTL 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.

BA21_CHAN_SW(R/W)

[0x60,B0,100, 150] Write TX/Read RX FIFO Port

RX and TX FIFO Port

Data Bit	Description
31-0	FIFO data word

Figure 15 PMC-Parallel-TTL RX/TX FIFO Port

This port is used to make single-word accesses into the TX and out of the RX FIFO. Please note that reading is from the RX FIFO and writing is to the TX FIFO. Unless Bypass mode is established the data will not match.

BA21_CHAN_TX_AECNT

[0x64,B4,104,154] TX almost-empty level (read/write)

TX Almost-Empty Level Register

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

Figure 16 PMC-Parallel-TTL TX ALMOST EMPTY LEVEL register

This read/write port accesses the transmitter almost-empty level register. When the number of data words in the transmit data FIFO is less 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. 1k x32 is the TX FIFO for a 10 bit valid count range [9-0].

BA21_CHAN_RX_AFCNT

[0x68,B8,108,158] RX almost-full level (read/write)

RX Almost-Full Level Register

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

Figure 17 PMC-Parallel-TTL RX ALMOST FULL LEVEL register

This read/write port accesses the receiver almost-full level register. When the number of data words in the receive data FIFO is 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. 1k x32 is the RX FIFO for a 10 bit valid count range [9-0].

BA21_CHAN_TX_CNT

[0x58,A8,F8,148] TX FIFO data count (read only)

TX FIFO Data Count Port

Data Bit	**Description**
31-16	Spare
15-0	TX Data Words Stored

Figure 18 PMC-Parallel-TTL TX FIFO data count Port

This read-only register port reports the number of 32-bit data words in the transmit FIFO. The TX FIFO has a maximum of 1023 locations.

BA21_CHAN_RX_CNT

[0x5C,AC,FC,14C] RX FIFO data count (read only)

RX FIFO Data Count Port

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

Figure 19 PMC-Parallel-TTL RX FIFO data count Port

This read-only register port reports the number of 32-bit data words in the receive FIFO. The channel status register contains the combined pipeline and FIFO count. This design has 1027 locations possible in the FIFO + pipeline.

BA21_CHAN_PCK_CNT

[0x9C,EC,13C,18C] Packet count (read only)

Packet Count Port

Data Bit	**Description**
31-16	Spare
15-0	Packet Size

Figure 20 PMC-Parallel-TTL Packet Count Port

This read-only port provides the number of stored bytes per packet. The data is stored into a 1kx32 FIFO allowing multiple counts to be stored.

BA21_CHAN_TARGET

[0x84,C4,124,174] Target Control Register R/W

RX Address Port

Data Bit	**Description**
31-8	Spare
7-1	Address
0	Spare

Figure 21 PMC-Parallel-TTL Target Control Port

The Target Address is set in this port. The address is offset to match the format used by I2C and the Master programming port. If set to “0000000” and in Target mode the channel will “snoop” capturing all transactions including the address. If set to a non zero address the channel will only store or respond to requests for the programmed address.

BA21 Master Notes

The first LW written to the TX FIFO determines how a packet will be implemented. The bottom 8 bits match the combination of Address and R/W from I2C => 7-1 = Address of port to access, Bit 0 = R/W where Read = 1. The next byte up [15-8] determines the size of the transfer in bytes. 1-255 in a particular packet. Byte 2 [23-16] is the first byte to transmit given a Write, and is don’t care for a Read. Byte 3 [31-24] is the second byte to transmit for a Write and is don’t care for a Read. The second LW will be read for lengths greater than 2. The data will be sent 0,1,2,3 byte positions as defined above. This process will continue for the programmed length.

When non LW boundaries for the data are utilized, the HW will skip the unused locations and start a new packet with the next read – next LW boundary.

Additional packets can be loaded while data is already be processed. Those packets will be transmitted as soon as the current one is completed.

When Reading the data read back is stored into the Rx side FIFO.

The Packet Done interrupt or status bit can be used to determine when the transfer is complete. For multiple packets, you can create a single larger file and use a longer DMA transfer with the HW automatically breaking up the transfers as programmed.

BA21 Target notes

When acting as a Target, the TX FIFO must be prefilled with the response prior to receiving a request for data from another Master. The data will be returned based on the requests received. Data is pulled 0,1,2,3 byte order [7-0, 15-8 …] Unused data is skipped. If multiple responses are to be loaded, the data used per message will correspond to the Masters request which may not be multiples of 4. When the Stop sequence or a second Start sequence is detected after the ACK the Target will produce the Packet Complete status/interrupt, write the packet size if receiving data, and set-up to respond to a new request.

Data received will also be stored 0,1,2,3. The Packet Count can be used to parse messages. Especially helpful when more data is coming in while processing previous data. Unused locations are zero filled. 3 bytes received, 4th is filled and so forth.

When in Snoop mode the first byte of a packet will be the address. The Packet Count for packets received in Snoop mode will be for the total size including the stored address.

Loop-back

The Engineering kit has reference software, which includes external loop-back tests. The PMC-Parallel-TTL has a 68 pin SCSI II front panel connector. The tests require an external cable with the following pins connected. We use HDEterm68 for this purpose.

External BA21 function Loop-Back

Signal	From	To	Signal
Parallel Port			
IO_16	pin 17	pin 51	IO_48
IO_17	pin 16	pin 50	IO_49
IO_18	pin 15	pin 49	IO_50
IO_19	pin 14	pin 48	IO_51
IO_20	pin 13	pin 47	IO_52
IO_21	pin 12	pin 46	IO_53
IO_22	pin 11	pin 45	IO_54
IO_23	pin 10	pin 44	IO_55
IO_24	pin 9	pin 43	IO_56
IO_25	pin 8	pin 42	IO_57
IO_26	pin 7	pin 41	IO_58
IO_27	pin 6	pin 40	IO_59
IO_28	pin 5	pin 39	IO_60
IO_29	pin 4	pin 38	IO_61
IO_30	pin 3	pin 37	IO_62
IO_31	pin 2	pin 36	IO_63
I2C			
SCK0	pin 33	pin 31	SCK1
SDA0	pin 32	pin 30	SDA1
SCK2	pin 29	pin 27	SCK3
SDA2	pin 28	pin 26	SDA3

PMC Module Logic Interface Pin Assignment

The figure below gives the pin assignments for the PMC Module PCI Pn1 Interface on the PMC-Parallel-TTL.   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 22 PMC-PARALLEL-TTL 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-Parallel-TTL. 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 23 PMC-PARALLEL-TTL Pn2 Interface

PMC Module Front Panel IO Interface Pin Assignment

The figure below gives the pin assignments for the PMC Module IO Interface on the PMC-Parallel-TTL. Installed for –FP and –FRP models. Also see the User Manual for your carrier board for more information.

EXT_CLK_EN	EXT_CLK	1	35
IO_31	IO_63	2	36
IO_30	IO_62	3	37
IO_29	IO_61	4	38
IO_28	IO_60	5	39
IO_27	IO_59	6	40
IO_26	IO_58	7	41
IO_25	IO_57	8	42
IO_24	IO_56	9	43
IO_23	IO_55	10	44
IO_22	IO_54	11	45
IO_21	IO_53	12	46
IO_20	IO_52	13	47
IO_19	IO_51	14	48
IO_18	IO_50	15	49
IO_17	IO_49	16	50
IO_16	IO_48	17	51
IO_15	IO_47	18	52
IO_14	IO_46	19	53
IO_13	IO_45	20	54
IO_12	IO_44	21	55
IO_11	IO_43	22	56
IO_10	IO_42	23	57
IO_9	IO_41	24	58
IO_8	IO_40	25	59
IO_7	IO_39	26	60
IO_6	IO_38	27	61
IO_5	IO_37	28	62
IO_4	IO_36	29	63
IO_3	IO_35	30	64
IO_2	IO_34	31	65
IO_1	IO_33	32	66
IO_0	IO_32	33	67
GND	GND	34	68

Figure 24 PMC-PARALLEL-TTL FRONT PANEL Interface

PMC Module Front Panel IO Interface Pin Assignment

The figure below gives the pin assignments for the PMC Module IO Interface on the PMC-Parallel-TTL-BA21. See the User Manual for your carrier board for more information.

RESERVED	RESERVED	1	35
ParData15	ParData31	2	36
ParData14	ParData30	3	37
ParData13	ParData29	4	38
ParData12	ParData28	5	39
ParData11	ParData27	6	40
ParData10	ParData26	7	41
ParData9	ParData25	8	42
ParData8	ParData24	9	43
ParData7	ParData23	10	44
ParData6	ParData22	11	45
ParData5	ParData21	12	46
ParData4	ParData20	13	47
ParData3	ParData19	14	48
ParData2	ParData18	15	49
ParData1	ParData17	16	50
ParData0	ParData16	17	51
Spare	Spare	18	52
Spare	Spare	19	53
Spare	Spare	20	54
Spare	Spare	21	55
Spare	Spare	22	56
Spare	Spare	23	57
Spare	Spare	24	58
Spare	Spare	25	59
SDA3	GND	26	60
SCK3	GND	27	61
SDA2	GND	28	62
SCK2	GND	29	63
SDA1	GND	30	64
SCK1	GND	31	65
SDA0	GND	32	66
SCK0	GND	33	67
GND	GND	34	68

Figure 25 PMC-PARALLEL-TTL BA21 FRONT PANEL Interface

Spare are not used IO. ParData form a parallel port. SDA & SCK form, I2C pairs.

PMC Module Backplane IO Interface Pin Assignment

The figure below gives the pin assignments for the PMC Module IO Interface on the PMC-Parallel-TTL and routed to Pn4. Pn4 installed for –RP and –FRP models. Not installed for BA21. Also see the User Manual for your carrier board for more information.

IO_0	IO_1	1	2
IO_2	IO_3	3	4
IO_4	IO_5	5	6
IO_6	IO_7	7	8
IO_8	IO_9	9	10
IO_10	IO_11	11	12
IO_12	IO_13	13	14
IO_14	IO_15	15	16
IO_16	IO_17	17	18
IO_18	IO_19	19	20
IO_20	IO_21	21	22
IO_22	IO_23	23	24
IO_24	IO_25	25	26
IO_26	IO_27	27	28
IO_28	IO_29	29	30
IO_30	IO_31	31	32
IO_32	IO_33	33	34
IO_34	IO_35	35	36
IO_36	IO_37	37	38
IO_38	IO_39	39	40
IO_40	IO_41	41	42
IO_42	IO_43	43	44
IO_44	IO_45	45	46
IO_46	IO_47	47	48
IO_48	IO_49	49	50
IO_50	IO_51	51	52
IO_52	IO_53	53	54
IO_54	IO_55	55	56
IO_56	IO_57	57	58
IO_58	IO_59	59	60
IO_60	IO_61	61	62
IO_62	IO_63	63	64

Figure 26 PMC-PARALLEL-TTL PN4 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. Open Drain 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. The PMC-Parallel-TTL has optional transorbs for input protection. In addition series resistors are used and can be specified to be something other than the 22 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.

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-Parallel-TTL 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-PARALLEL-TTL 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 $150. 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:	64 discrete IO channels. Each has a separate enable to control output. Inputs are maskable and always available. 4 channels of I2C IO each with programmable rate TX. 32 bit parallel port
CLK rates supported:	Osc, PLL, PCI, Available. PLLA-D used to program rate for Channel 0-3
Software Interface:	Control Registers, IO registers, IO Read-Back registers
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:	DMA interrupts, I2C function support
Onboard Options:	All Options are Software Programmable
Interface Options:	68 Pin SCSI III connector at front bezel User IO routed to Pn4 – not installed BA21
Dimensions:	Standard Single PMC Module.
Construction:	Multi-Layer Printed Circuit, Through Hole and Surface Mount Components.
Temperature Coefficient:	2.17 W/^{o}C for uniform heat across PMC
Power:	TBD mA @ 5V outputs off Add 10 mA per active low output for pull-up current drivers support -32/+64 mA per IO line, higher currents are possible depending on load.

Order Information

standard temperature range 0-70°C
PMC-Parallel-TTL-BA21 PMC Module with 4 I2C channels, 32 bit parallel port, front panel IO, 3.3V reference voltage to pull-ups, DMA support, 1Kx32 FIFO RX, 1Kx32 FIFO TX, 1Kx32 PacketFIFO per channel
http://www.dyneng.com/pmc_parallel_TTL.html

Order Options:

Pick One
–FP for front panel IO only [default if no selection made]
-RP for rear panel IO PN4 only
-FRP for both IO connections
Shown for reference. BA21 selection determines [-FP]

Pick any combination to go with IO
-TRANS to add transorbs
-CC to add conformal coating
-ET to add Industrial Temp [-40 +85]
-TS to add thumbscrew option – standard is latch block
-3V to change from 5V IO reference to 3.3V IO reference
Shown for reference. BA21 selection determines [-3V]

Related:
PCI2PMC**:** PCI to PMC adapter to allow installation of PMC-Parallel-TTL into a PCI system.
http://www.dyneng.com/pci2pmc.html

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

HDEterm68**:** 68 position terminal block with two SCSI II/III connectors. PMC-Parallel-TTL compatible.
http://www.dyneng.com/HDEterm68.html

HDEcabl68: SCSI II/III cable compatible with FPIO on PMC Parallel IO.
http://www.dyneng.com/HDEcabl68.html

PIM_Parallel_IO : PMC IO Module for PMC Parallel TTL design. Provides FPIO in cPCI systems when used with a PIM Carrier
http://www.dyneng.com/pim_parallel_io.shtml

PMC Parallel IO Eng Kit :   HDEterm68-MP, HDEcabl68, Driver software, reference schematics. Recommended for first time purchases.
http://www.dyneng.com/pmc_parallel_TTL.html