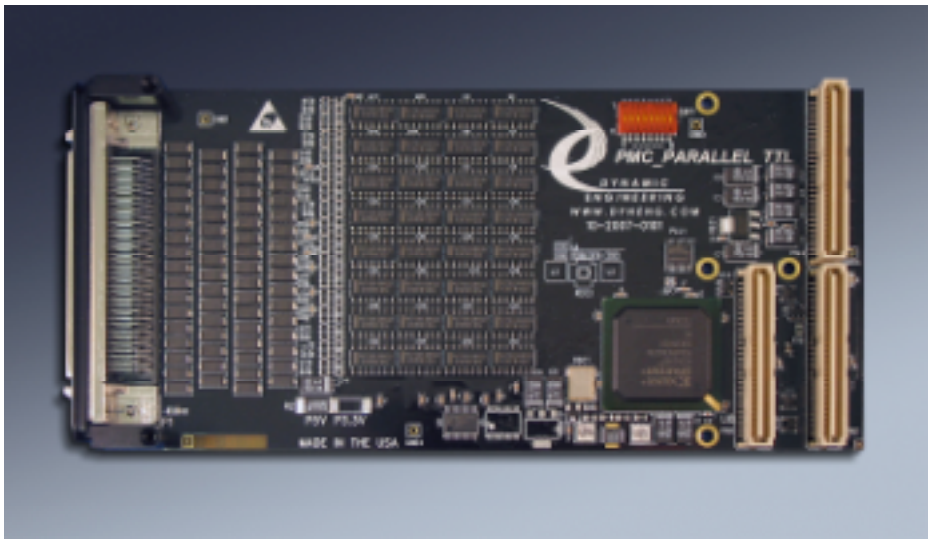
DYNAMIC ENGINEERING
150 DuBois St. Suite C, Santa Cruz, CA 95060
831-457-8891 Fax 831-457-4793
http://www.dyneng.com
sales@dyneng.com
Est. 1988

User Manual

PMC-PARALLEL-TTL-BA17

Digital Parallel Interface
PMC Module
64 programmable IO
COS Inputs
32 Inputs with TimeStamp, FIFO Storage, DMA Transfer



Revision A
Corresponding Hardware: Revision 1
10-2007-0101
FLASH 0301

PMC-PARALLEL-TTL-BA17

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

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

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

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

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

Manual Revision A. Revised 11/24/08

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. 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 architectures, and is adding more as new solutions are requested, and required by our customers.

The 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-BA17 is a customerized version of the standard PMC-Parallel-TTL board. “BA17” is set to 3.3V, has rear panel IO, and an added feature of FIFO stored COS values for the upper 32 IO lines. The COS rate is programmable and can be based on the PLLA or the oscillator [50 MHz].

The storage function uses inputs from the COS detectors to determine when a new transition has been detected. The 32 rising and 32 falling control bits allow software to determine which channels are stored under what conditions. The data is stored as a Rising Vector, Falling Vector, and TimeStamp. The FIFO is 12K deep allowing 4K samples to be captured or more if DMA is enabled. The hardware can stream indefinitely with DMA enabled.

The BA17 features are selectable. The standard register based, and COS functions are available on unused [by BA17] pins and can be swapped with the BA17 functions under software control.

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.

Each channel is programmable to be input or output on a channel-by-channel basis. All 64 IO channels can be used as interrupt generators. Interrupts are programmable to be

based on rising, falling and change of state [both] conditions. The interrupts are maskable to allow polled operation as well.

The inputs are available unfiltered and after the transition detection. The transition detection is programmable for clock rate. The local 50 MHz oscillator can be used or the PLLA. The reference rate is divided by 10 for the COS rate to provide a fixed 10X clock for the data loader function [COS data to FIFO]. By using a 10X clock transitions can be captured on consecutive COS clocks and still be properly loaded into the FIFO.

For example, with the oscillator selected, the COS clock would be 5 MHz and the loader will use the 50 Mhz from the oscillator.

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. BA17 is design number 3 for the PMC-Parallel-TTL with a corresponding FLASH of 03xx.

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.

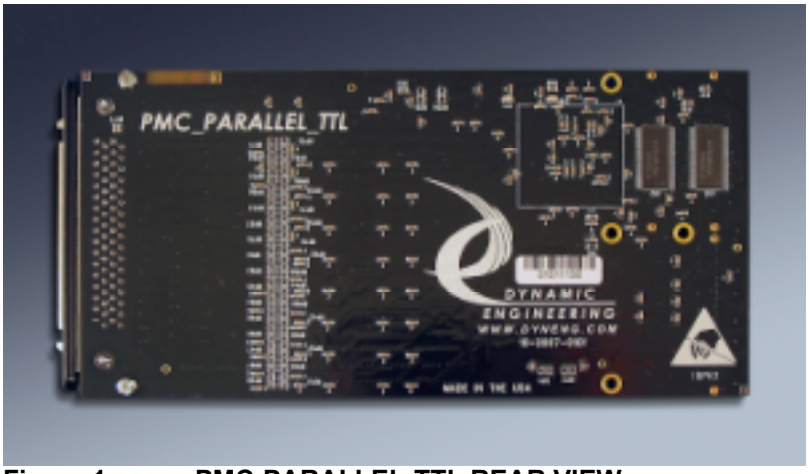


Figure 1 PMC-PARALLEL-TTL REAR VIEW

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 0301 has the additional features of utilizing the internal block RAM to create a FIFO [0x3001 deep with pipeline] to support the input data from the COS function. DMA can be used to move the data from the FIFO to the system memory. A 32 bit TimeStamp counter plus control registers are also added to this version. The TimeStamp counter can be stopped and started as well as preloaded with a user defined count. The counter increments at the COS programmed rate.

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

The 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.

The 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.

The Xilinx design incorporates the “PCI Core” and additional modules for DMA in parallel with a direct register decoded programming model. The initial implementation provides an enhanced feature set based on the PMC Parallel IO design. Additional FLASH updates will provide DMA, pattern generation, pulse generation, and user defined requirements.

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.

Data written to the IO registers can be placed on the bus. The master enable allows all 64 channels to be synchronized if desired. The master enable can be programmed “on” to allow direct updates if 64 bit synchronization is not required.

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 Ω 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.

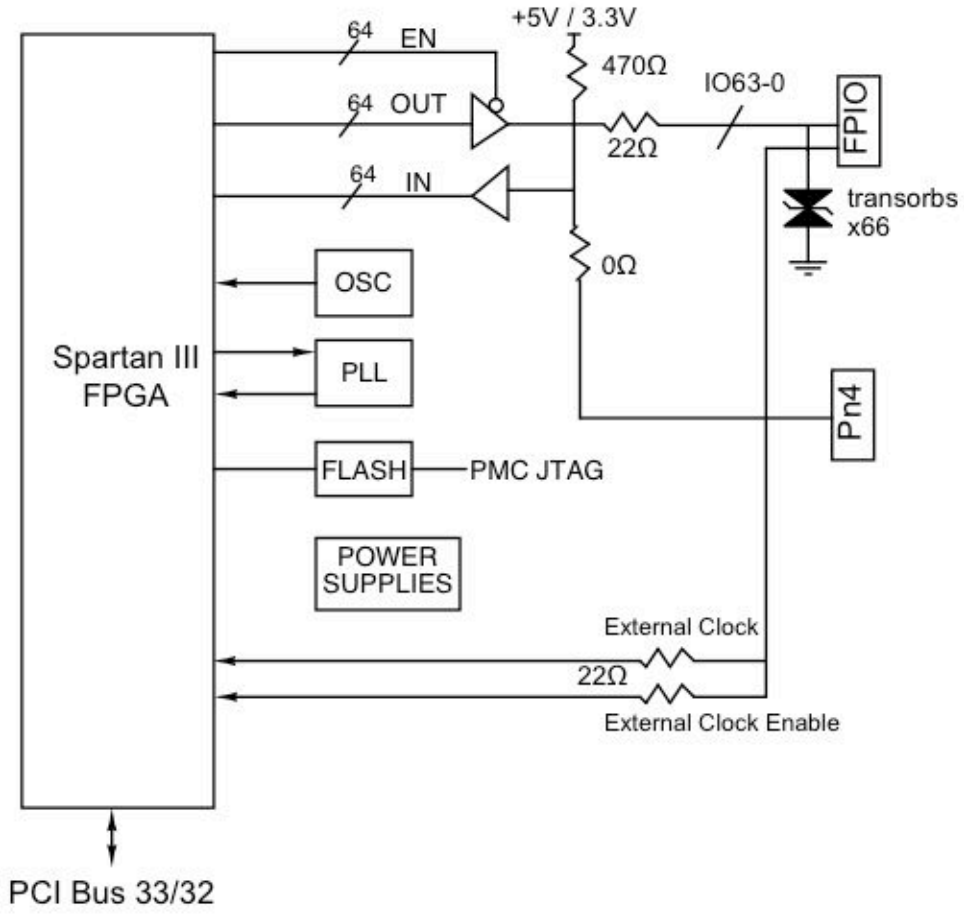
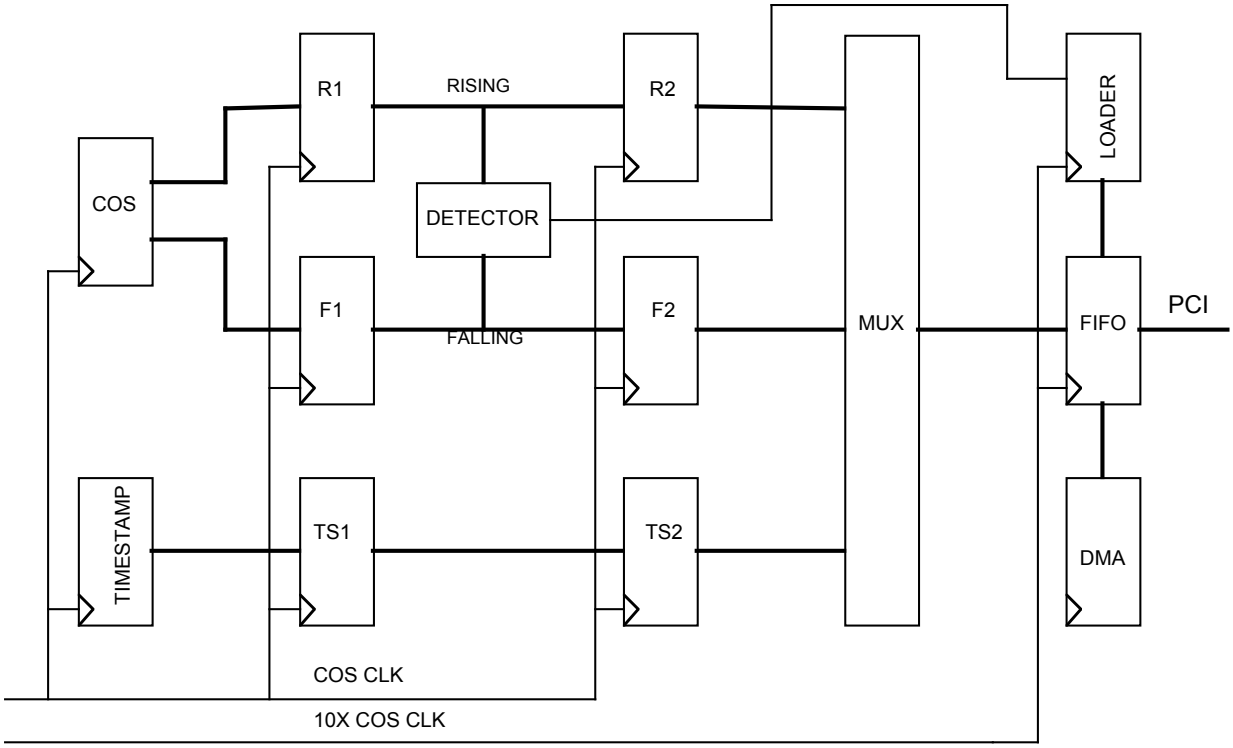


Figure 2 PMC-PARALLEL-TTL Block Diagram

PMC Parallel TTL BA17 features a programmable data path with DMA support. The internal block RAM is configured to provide FIFO's for receive [(3 x 4K + 4) x 32 RX]. The PLL or Oscillator is used as the RX [COS] state-machine reference.

For each bit programmed to be valid for Rising or Falling or both edges data will be stored when an event has been detected. The loader function is enabled through the channel control register. When enabled a detected transition in the upper 32 IO lines will trigger the loader function. The outputs from the COS detectors are pipelined with the first stage used to check for transitions, and the second to load to the FIFO. The pipeline is necessary due to the number of bits to process [64 when up and down are considered] and the muxing prior to the FIFO. A local "loader" state-machine checks for an active bit and writes the Rising, Falling, and timestamp to the FIFO. The loader state-machine runs with a 10X clock to allow for mux selection and writing to the FIFO without missing data – consecutive clocks of the COS can be active with new edges to load. The detector and loader have enough bandwidth to trigger, load, and rearm within each COS clock cycle.



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

load rate into the FIFO.

With DMA and lower frequencies of COS transitions this mode can work well. At our 5 MHz example, we have 33x on clock rate and 3x on data loaded per COS clock when a transition is detected. Using 50% of PCI BW as a guide the transitions can come at the COS rate and the hardware will be right at 50%. Normally transitions won't be detected on every COS clock leading to a lower percentage of the bandwidth.

The DMA FIFO is 12Kx32 for RX leaving a lot of "rubber band" in the memory chain to support the COS. As the transitions frequency is increased the multiplier can be reduced to the point where the FIFO may go full on occasion before the DMA can reduce the stored data. OS delays are the main culprit. More than 4096 transitions can be detected and stored without any DMA movement. Since the transitions are not known as a function of time it is beyond the scope of this manual to predict the storage time within the FIFO when the OS is otherwise occupied.

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 slightly more efficient as there are potentially 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.

Address Map

Function	Offset
// PMC Parallel TTL definitions	
#define pmcparttl_BASE	0x0000 // 0 PMC Parallel TTL base control register offset
#define pmcparttl_ID	0x0004 // 1 PMC Parallel TTL ID Register offset
#define pmcparttl_STATUS	0x0008 // 2 PMC Parallel TTL status Register offset
#define pmcparttl_DirL	0x000c // 3 PMC Parallel TTL Direction lower Register offset
#define pmcparttl_DirU	0x0010 // 4 PMC Parallel TTL Direction upper Register offset
#define pmcparttl_DatL	0x0014 // 5 PMC Parallel TTL Data lower Register, line data read
#define pmcparttl_DatU	0x0018 // 6 PMC Parallel TTL Data upper Register, line data read
#define pmcparttl_DatLreg	0x001c // 7 PMC Parallel TTL Data lower Register read-back
#define pmcparttl_DatUreg	0x0020 // 8 PMC Parallel TTL Data upper Register read-back
#define pmcparttl_COSclk	0x0024 // 9 PMC Parallel TTL COS Clock definition Register
//#define spare	0x0028 // 10 PMC Parallel TTL
#define pmcparttl_RisLreg	0x002c // 11 PMC Parallel TTL Rising lower Register
#define pmcparttl_RisUreg	0x0030 // 12 PMC Parallel TTL Rising upper Register
#define pmcparttl_FallLreg	0x0034 // 13 PMC Parallel TTL Falling lower Register
#define pmcparttl_FallUreg	0x0038 // 14 PMC Parallel TTL Falling upper Register
#define pmcparttl_IntRisLreg	0x003c // 15 PMC Parallel TTL Interrupt Enable Rising lower Register
#define pmcparttl_IntRisUreg	0x0040 // 16 PMC Parallel TTL Interrupt Enable Rising upper Register
#define pmcparttl_IntFallLreg	0x0044 // 17 PMC Parallel TTL Interrupt Enable Falling lower Register
#define pmcparttl_IntFallUreg	0x0048 // 18 PMC Parallel TTL Interrupt Enable Falling upper Register
#define pmcparttl_IntRisLstat	0x004c // 19 PMC Par TTL Interrupt Rising LWR Stat Rd, write = clear
#define pmcparttl_IntRisUstat	0x0050 // 20 PMC Par TTL Interrupt Rising UPR Stat Rd, write = clear
#define pmcparttl_IntFallLstat	0x0054 // 21 PMC Par TTL Interrupt Falling LWR Stat Rd, write = clear
#define pmcparttl_IntFallUstat	0x0058 // 22 PMC Par TTL Interrupt Falling UPR Stat Rd, write = clear
#define pmcparttl_DR_L	0x005C // 23 PMC Par TTL DMA - Register bit selection 31-0 R/W
#define pmcparttl_DR_U	0x0060 // 24 PMC Par TTL DMA - Register bit selection 63-32 R/W
#define pmcparttl_TIMESTAMP	0x0064 //25 PMC Par TTL TimeStamp Preload and readback
#define pmcparttl_TIMESTAMPCNT	0x0068 // 26 PMC Par TTL TimeStamp current count

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

The address map provided is for the local decoding performed within PMC-Parallel-TTL. 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 = 0x0038 for the PMC-Parallel-TTL-BA17.

Function	Offset
// PMC Parallel TTL definitions	
#define pmcparttl_ch0_base	0x0078 // 30 PMC Par TTL DMA path base control register channel 0
#define pmcparttl_ch0_st	0x007C// 31 PMC Par TTL channel 0 status, interrupt clear, data count
#define pmcparttl_ch0_brstin	0x0080// 32 PMC Par TTL channel 0 burst in control, unused BA17
#define pmcparttl_ch0_brstout	0x0084// 33 PMC Par TTL channel 0 burst out control,
#define pmcparttl_ch0_swr	0x0088// 34 PMC Par TTL ch 0 FIFO sin read from RX
#define pmcparttl_ch0_rx_aecnt0	0x008C// 35 PMC Par TTL ch 0 almost empty count register and rd-bk
#define pmcparttl_ch0_rx_afcnt0	0x0090 // 36 PMC Par TTL ch 0 almost full count register and rd-bk
#define pmcparttl_ch0_rx_aecnt1	0x0094// 37 PMC Par TTL ch 0 almost empty count register and rd-bk
#define pmcparttl_ch0_rx_afcnt1	0x0098 // 38 PMC Par TTL ch 0 almost full count register and rd-bk
#define pmcparttl_ch0_rx_aecnt2	0x009C// 39 PMC Par TTL ch 0 almost empty count register and rd-bk
#define pmcparttl_ch0_rx_afcnt2	0x00A0 // 40 PMC Par TTL ch 0 almost full count register and rd-bk
#define pmcparttl_ch0_rx_ffcnt	0x00A4 // 41 PMC Par TTL ch 0 rx fifo word count w/o pipeline

Figure 4 PMC-PARALLEL-TTL Channel Address Map

Programming

Programming the PMC-PARALLEL-TTL-BA17 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-BA17 card the 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-BA17. If the basic mode of direct read and write operations is to be used then the default settings can be used except for setting the master output enable and the direction bits corresponding to the channels to transmit on.

If COS inputs are to be used the reference and divisor clocks may require programming. In many cases the default settings will work. In addition the Rising, Falling, and Interrupt capabilities need to be programmed. Once the settings are in place it is recommended that the receive state registers are written to for clearing purposes as the programming steps may cause phantom events to be captured.

One additional programming step will be to initialize the PLL to the user desired frequency for COS capture should the PLL be used for that purpose.

For Windows™ and Linux 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.

To use the BA17 specific functions the Channel Control, 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 BA17 to start the DMA. Please refer to the Burst IN and Burst Out register discussions.

DMA should be set-up before starting the Loader function for the upper 32 bits. The Rising and Falling enables for the IO of interest will be set to 1. The COS will automatically convert the incoming signals on the enabled IO to a series of pulses that act as status valid for 1 clock. The loader function will move the status to the FIFO when at least 1 of the enabled lines is active. The status is stored Rising, Falling, TimeStamp. The TimeStamp should also be initialized prior to starting.

The TimeStamp is a free running counter with enable and preload functions. The preload can be used to set to a starting value offset or initialize to zero. The counter rolls over from 0xFFFFFFFF to 0x00 and keeps going. You can preload with a negative count to allow software time to match-up with the count.

TimeStamp uses the COS clock. Each count is 1 period of the COS clock. If you use the oscillator reference of 50 MHz the granularity is 200 nS. If you use the PLLA clock reference it will be the period of the programmed frequency x10.

The initial client request was 1-5 MHz for the COS range. The hardware has been tested at 6 [60 on the PLL] with good results. Prior to the update the COS ran directly from the OSC and operated at 50 MHz. The x10 loader requirement will reduce the upper limit of the COS sampling.

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 our any extra data captured. For on-the-fly processing multiple shorter DMA segments can be programmed, and at the interrupt restart DMA to point at the alternate segment to allow processing on the previous one. This technique is sometimes refered to “ping-pong”.

Register Definitions

pmcparttl_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
15-5	spare
4	Master Parallel Data Enable
3	spare
2	spare
1	Force Interrupt
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.

Force Interrupt when ‘1’ and the master enabled will cause an interrupt request. The interrupt can be cleared by clearing this bit or disabling the master interrupt enable or both. Force Interrupt is used for test and software development purposes.

Master Parallel Data Enable is used to allow the upper and lower data to be synchronized. The upper 32 bits and the lower 32 bits are not accessed at the same time. If the user wants to have the upper and lower data change at the same time the Master enable can be cleared to ‘0’, both halves of the data written and then the enable set ‘1’. If synchronization is not an issue; program to ‘1’ as part of initialization.

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.

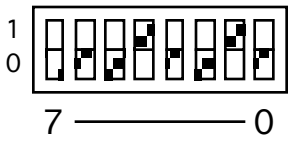
pmcparttl_ID

[$04 Switch and Design number port read only]

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

Figure 6 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 BA17 design is 0x03 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.

pmcparttl_STATUS

[$08 Board level Status Port read only]

DATA BIT	DESCRIPTION
31	Interrupt Status
30-17	
16	int_stat0
15-6	spare
5	INTR Falling
4	INTR Rising
3-1	spare
0	local interrupt

Figure 7 PMC-PARALLEL-TTL Status Port Bit Map

Local Interrupt for the base design, this bit is the same as the Intforce bit – unmasked.

INTR Rising - This is the logical OR of the COS outputs for the Rising Edge condition. The RISING register will select which bits are enabled. If any of the enabled bits are active this bit is set. The status is captured before the master interrupt enable. If the master interrupt enable is set an interrupt will be generated if this condition is true.

INTR Falling - This is the logical OR of the COS outputs for the Falling Edge condition. The Falling register will select which bits can be active [enabled]. If any of the enabled bits capture a falling edge this bit will be set. The status is captured before the master interrupt enable. If the master interrupt enable is set an interrupt will be generated if this condition is true.

Int_Stat0/1 – This is the local masked not board level masked interrupt from channel 0. Int_Stat0/1 = DMA Write and DMA Write mask or DMA Read and DMA Read Mask or (IntForce or TX request) and Channel 0/1 mask. This bit will tell the SW if any channel 0/1 asset could be requesting an interrupt. If the master interrupt enable is set an interrupt will be generated if this condition is true.

Interrupt Status – Set if the PCI interrupt is asserted. This bit can be checked to determine if this card is causing an interrupt to the system. If set the other bits can be checked to see which feature(s) of the board need to be serviced. Secondary reads to the COS or Channel will determine the exact type of interrupt.

pmcparttl_DirL

[$0C Direction Register bits 31-0 read – write]

DATA BIT	DESCRIPTION
31-0	DIR31-0

Figure 8 PMC-PARALLEL-TTL Direction Lower Bit Map

The lower 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 direction bit(s) to ‘1’.

pmcparttl_DirU

[$10 Direction Register bits 63-32 read – write]

DATA BIT	DESCRIPTION
31-0	DIR63-32

Figure 9 PMC-PARALLEL-TTL Direction Upper Bit Map

The upper 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 direction bit(s) to ‘1’.

Once a Direction bit is set to output the data in the corresponding output holding register bit is broadcast on that IO line. The data in the holding register will match the data in the data output register if the master parallel enable bit is set. If initial states are important you may want to program the initial data and enable it before enabling the direction bits.

pmcparttl_DatL

[$14 Data IO Port read/write]

DATA BIT	DESCRIPTION
31-0	Data IO 31-0

Figure 10 PMC-PARALLEL-TTL Data IO Lower Bit Map

pmcparttl_DatU

[$18 Data IO Port read/write]

DATA BIT	DESCRIPTION
31-0	Data IO 63-32

Figure 11 PMC-PARALLEL-TTL Data IO Upper Bit Map

This port is really a combined Data Output port and a Data Input port. The data to be transmitted is written to the Data Output Port side of the Data Register. The data to be read from the IO are read from Data Input side of the Data register. Read back from the Data Output port is done though the separate “datareg” port.

The data read from the data register is a direct read of the state of the IO lines. The bits are not modified for level or transition etc. Some bits may be defined as outputs. The input will match the output definition in this case. Local loop-back can be performed for the bits where outputs are defined. The inputs will match the state of the system when external devices can drive the input lines. The input bits can be masked out of the data word to reduce the data to external inputs.

The output bits are driven onto the IO for the bits that are enabled with the direction control register, and when the master parallel enable is set. For bits without the direction register bit set there are no side effects. The direction register will act as a mask for the data register.

pmcparttl_DatLreg

[$1C Data Reg Port read only]

DATA BIT	DESCRIPTION
31-0	Data IO 31-0

Figure 12 PMC-PARALLEL-TTL Data Reg Lower Bit Map

pmcparttl_DatUreg

[$20 Data Reg Port read only]

DATA BIT	DESCRIPTION
31-0	Data IO 63-32

Figure 13 PMC-PARALLEL-TTL Data Reg Upper Bit Map

Data written to the Data IO registers can be read back through this port. The register is read back instead of the IO side when accessing this port. The data will match the state of the data output bits written to the output side of the Data IO register.

pmcparttl_COSclk

[$24 COS clock definition port read -write]

DATA BIT	DESCRIPTION
15	Data Out 0 Enable
14-13	CLOCK PRE-SELECTOR
12	CLOCK POST-SELECTOR
11-0	DIVISOR

Figure 14 PMC-PARALLEL-TTL COS Clk Control Bit Map

Unused for BA17 – PLL or Oscillator choice made through channel control register.

Data Out 0 Enable when set and the corresponding Direction bit is set will drive the COS clock out on Data bit 0. An oscilloscope can be used to verify the frequency setting that is programmed with the COSclk register.

CLOCK PRE-SELECTOR
00 PCI Clock
01 Oscillator
10 External Clock
11 PCI Clock
The clock pre-selector is used to select which reference clock to use with the divisor hardware (clock source). The base design oscillator rate is 50 MHz. The external clock can be any TTL level source driven onto the External Clock input line. The clock should be free running to be used for this purpose.

POST-SELECTOR when '1' sets the output clock to the divided clock, when '0' sets the output clock to the pre-selector reference value (clock source).

DIVISOR[11-0] are the clock divisor select bits. The clock source is divided by a 12-bit counter. The output frequency is {reference / [2(n+1)]}, $n \geq 1$. The counter divides by N+1 due to counting from 0 to n before rolling over. The output is then divided by 2 to produce a square wave output.

The desired frequency of 1 MHz. Is achieved by selecting Osc reference, divided clock and a factor of 50 with the standard 50 MHz oscillator. 2(N+1) = 50 => N = 24. 0x3018 would be the correct value to write to the COSclk.

pmcparttl_RisLreg

$2C Rising Lower Control Register Port read/write

DATA BIT	DESCRIPTION
31-0	Rising 31-0

Figure 15 PMC-PARALLEL-TTL Rising Lower Bit Map

pmcparttl_RisUreg

$30 Rising Upper Control Register Port read/write

DATA BIT	DESCRIPTION
31-0	Rising 63-32

Figure 16 PMC-PARALLEL-TTL Rising Upper Bit Map

The Rising control register bits correspond to the input data bits. All IO can be set-up for COS activity even if defined as an output. In most cases the output bits will be set to '0' for the Rising register. When set '1' and the corresponding input bit transitions from low to high the COS register of rising activity will be have the corresponding bit set. If the separate interrupt enable bit is also set then an interrupt can be generated. The Rising register is a control register. The COS data is read back separately.

pmcparttl_FallLreg

$34 Rising Lower Control Register Port read/write

DATA BIT	DESCRIPTION
31-0	Falling 31-0

Figure 17 PMC-PARALLEL-TTL Falling Lower Bit Map

pmcparttl_FallUreg

$38 Rising Upper Control Register Port read/write

DATA BIT	DESCRIPTION
31-0	Falling 63-32

Figure 18 PMC-PARALLEL-TTL Falling Upper Bit Map

The Falling control register bits correspond to the input data bits. All IO can be set-up for COS activity even if defined as an output. In most cases the output bits will be set to '0' for the Falling register. When set '1' and the corresponding input bit transitions from High to Low the COS register of falling activity will be have the corresponding bit set. If the separate interrupt enable bit is also set then an interrupt can be generated. The Falling register is a control register. The COS data is read back separately.

pmcparttl_IntRisLreg

$3C Rising Interrupt Lower Control Register Port read/write

DATA BIT	DESCRIPTION
31-0	Rising Int En 31-0

Figure 19 PMC-PARALLEL-TTL Int rising Lower Bit Map

pmcparttl_IntRisUreg

$40 Rising Interrupt Upper Control Register Port read/write

DATA BIT	DESCRIPTION
31-0	Rising Int En 63-32

Figure 20 PMC-PARALLEL-TTL int Rising Upper Bit Map

The Rising Interrupt Enable control register bits correspond to the input data bits. All IO can be set-up for COS activity even if defined as an output. In most cases the output bits will be set to ‘0’ for the Rising Interrupt Enable register. When set ‘1’ and the corresponding Rising bit is captured by the COS register an interrupt can be generated. Please note that the master interrupt enable will also need to be set for the interrupt to be requested.

pmcparttl_IntFallLreg

$44 Falling Interrupt Lower Control Register Port read/write

DATA BIT	DESCRIPTION
31-0	Falling Int En 31-0

Figure 21 PMC-PARALLEL-TTL Int Falling Lower Bit Map

pmcparttl_IntFallUreg

$48 Falling Interrupt Upper Control Register Port read/write

DATA BIT	DESCRIPTION
31-0	Falling Int En 63-32

Figure 22 PMC-PARALLEL-TTL int Falling Upper Bit Map

The Falling Interrupt Enable control register bits correspond to the input data bits. All IO can be set-up for COS activity even if defined as an output. In most cases the output bits will be set to ‘0’ for the Falling Interrupt Enable register. When set ‘1’ and the corresponding falling bit is captured by the COS register an interrupt can be generated. Please note that the master interrupt enable will also need to be set for the interrupt to be requested.

pmcparttl_IntRisLstat

$4C Rising Status Lower Control Register Port read/write

DATA BIT	DESCRIPTION
31-0	Rising COS bits 31-0

Figure 23 PMC-PARALLEL-TTL Rising COS Status Lower

pmcparttl_IntRisUstat

$50 Rising Status Upper Control Register Port read/write

DATA BIT	DESCRIPTION
31-0	Rising COS bits 63-32

Figure 24 PMC-PARALLEL-TTL Rising COS status upper

The COS captured for those bits enabled with the Rising register are held in this register. The bits are held until cleared. Bits are cleared by writing to the register with the corresponding bit or bits set. Writing to the register with the data read will clear the bits the software has read, and not clear the bits not set at the time of reading. This is the recommended practice to avoid conflicts. It is recommended to write to all bits [clear] after setting the COS Rising and Direction bits to clear any potential COS status generated by set-up.

pmcparttl_IntRisLstat

$54 Falling Status Lower Control Register Port read/write

DATA BIT	DESCRIPTION
31-0	Falling COS Status bits 31-0

Figure 25 PMC-PARALLEL-TTL Falling COS Status Lower

pmcparttl_IntRisUstat

$58 Falling Status Upper Control Register Port read/write

DATA BIT	DESCRIPTION
31-0	Falling COS Status bits 63-32

Figure 26 PMC-PARALLEL-TTL Falling COS status upper

The COS captured for those bits enabled with the Falling register are held in this register. The bits are held until cleared. Bits are cleared by writing to the register with the corresponding bit or bits set. Writing to the register with the data read will clear the bits the software has read, and not clear the bits not set at the time of reading. This is the recommended practice to avoid conflicts. It is recommended to write to all bits [clear] after setting the COS Falling and Direction bits to clear any potential COS status generated by set-up.

pmcparttl_DR_L

[$0x5C DMA Register bits 31-0 read – write]

DATA BIT	DESCRIPTION
31-0	DMA or Register control 31-0

Figure 27 PMC-PARALLEL-TTL DMA Reg Lower Bit Map

Set to 0 for BA17 operation.

The lower 32 bits of the DMA / Register selection are controlled with this port. When reset this port is cleared 0x00000000. All IO are set to register control. To use one or more of the IO for DMA controlled functions; program the corresponding direction bit(s) to '1'.

pmcparttl_DR_U

[$60 DMA Register bits 63-32 read – write]

DATA BIT	DESCRIPTION
31-0	DMA or Register control 63-32

Figure 28 PMC-PARALLEL-TTL Direction Upper Bit Map

Set to 0 for BA17 operation.

The upper 32 bits of the DMA / Register selection are controlled with this port. When reset this port is cleared 0x00000000. All IO are set to register control. To use one or more of the IO for DMA controlled functions; program the corresponding direction bit(s) to '1'.

To use the DMA function of programmed parallel data output, the direction register bits and DR register bits corresponding to those outputs must be set to '1'. The Direction bits enable the IO, and the DR bits select the State Machine output path instead of the Register path. Please note that the bits are selected on a bit by bit basis. Pick the closest larger size with the state-machine and the actual size with the DR and Direction bits. Any unused bits [by the state machine with masking] can be used as registered IO or COS inputs.

pmcparttl_TIMESTAMP

[$64 TimeStamp Preload read – write]

DATA BIT	DESCRIPTION
31-0	Preload value write, Register value read

Figure 29 PMC-PARALLEL-TTL TimeStamp Preload Bit Map

The TimeStamp counter is preloaded by writing to this register. The value is stored into a preload register and held until overwritten. The TimeStamp Counter runs on the COS clock. The register holds the data and allows the COS clock to load the counter with the new current value. The preload can be done at anytime. It is recommended to disable the counter in the channel control register prior to initializing to a new value.

Preload values can be anything within the 32 bit range. Numbers close to the rollover count will act as a negative preload in the sense that the counter will count for N periods and then start at 0. A negative preload can be used to offset the start and allow for some synchronization options depending on your system.

Each count is 1 period of the COS clock. The roll over time will be 4,294,967,295 * period as defined by the oscillator [50 MHz/10 = 200 nS or PLLA/10]. The relative time between events can be determined by subtracting the timestamps and multiplying by the period. The absolute time can be determined by the zero time plus the count times the period.

Reading back from this register returns the preload value not the current count of the TimeStamp.

pmcparttl_TIMESTAMPCNT

[$68 TimeStamp Count read only]

DATA BIT	DESCRIPTION
31-0	Current TimeStamp Count

Figure 30 PMC-PARALLEL-TTL TimeStamp Count Bit Map

The count after the first pipeline delay is sampled on the PCI clock and saved into a register. This read only value can be used to determine the local time and potentially

for rate checking. If the PLL is set to 10 MHz the COS will be at 1 MHz. If a system timer is set for 1 mS then the count will be approximately 1000 if the count is read after the system timer expires. Caution may need to be exercised interpreting the results as the BA17 timer is likely more accurate than the system timer.

pmcparttl_ch0_base

[0x78] Channel Control Register (read/write)

Channel Control Register

Data Bit	Description
19	BI IDLE
18	BO IDLE
17	TX IDLE
16	RX IDLE
15	
14	CLK RX Sel
13	
12	
11	spare
10	
9	
8	
7	Enable RX
6	
5	Force Interrupt
4	Channel Interrupt Enable
3	Read DMA Interrupt Enable
2	Write DMA Interrupt Enable
1	
0	FIFO Reset

Figure 31 PMC-PARALLEL-TTL channel Control Register

The bit positions for the BA17 have been selected to allow minimal changes from BA16 SW already in place. The TX bits have no meaning and should be set to '0' when accessing this register.

FIFO Reset: When set to a one, the transmit and receive FIFOs 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. The DMA interrupts are not affected by the Master Interrupt Enable.

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. 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.

Enable RX: When set '1' will start the RX function. The RX function is the use of the upper 32 IO as COS values to load into the RX FIFO. The bits actually tested are controlled by the RISING and FALLING upper values. The control bits for the RX operation should be selected first to guarantee correct operation as the RX reference rate may be different from the PCI clock rate. The receiver function will write data to the RX FIFO until software disables the data capture.

Clock RX Select when '1' Selects PLL A reference clock to be used for COS. When '0' the oscillator [50 MHz] is selected. Remember to program the PLLA prior to selecting this option. Please note that HW reduces the PLLA output by a factor of 10. To select a COS clock and TimeStamp of X program PLLA for 10X. The data loader function uses PLLA for a 10X reference.

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 reception,the 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 TX and burst data out of the card for Receive. RX is only used on BA17.

pmcparttl_ch0_st

[0x7C,A4] Channel Status Read/Clear Latch Write Port

Channel Status Register

Data Bit	Description
31	Channel Interrupt Active
30-16	data count with pipeline
15	Read DMA Interrupt Occurred
14	Write DMA Interrupt Occurred
13	Read DMA Error Occurred
12	Write DMA Error Occurred
11	FIFO Overrun error
10	RX FIFO Full 0
9	RX FIFO Almost Full 2
8	RX FIFO Almost Empty 2
7	RX FIFO Empty 2
6	RX FIFO Full 1
5	RX FIFO Almost Full 1
4	RX FIFO Empty 1
3	RX FIFO Full 0
2	RX FIFO Almost Full 0
1	RX FIFO Almost Empty 0
0	RX FIFO Empty 0

Figure 32 PMC-Parallel-TTL Channel STATUS PORT

BA17 FIFO: Three 4K x 32 FIFO's are used to create a 12K FIFO. The hardware automatically moves data stored into FIFO 0 into FIFO 1 when there is room and again from FIFO1 to FIFO 2 when there is room. When FIFO1 is full the entire FIFO is full, and when FIFO 2 is empty the entire FIFO is empty. Please see note about pipeline below. The data is move at 2X the PCI clock rate and has a sophisticated transfer algorithm that allows burst mode between FIFO's when not almost full or empty and uses single transfers when close to the boundary conditions.

The block RAM used to create FIFO's uses one location for "overhead" so 4K is really

4095. The pipeline required to support DMA transfers is 4 deep. 4005 * 3 + 4 = 12289 locations.

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 also be used for read size control.

RX FIFO Empty[0,1,2]: 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 Empty[1,2]: 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 RX_AMT_LVL register; when a zero is read, the FIFO level is more than that value.

Receive FIFO Almost Full[0,1,2]: 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[0,1,2]: 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.

FIFO Overflow Error Occurred: When a one is read, an error has been detected. This will occur if FIFO 0 is full when the loader function tries to write to it. A zero indicates that no error has occurred. This bit is latched and can be cleared by writing back to the Status register with a one in the appropriate 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.

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.

pmcparttl_ch0_brstin

[0x80,A8] 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 33 PMC-Parallel-TTL Write DMA pointer register

UNUSED for BA17: 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.

pmcparttl_ch0_brstout

[0x84] 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 34 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.

pmcparttl_ch0_swr

[0x88] Write TX/Read RX FIFO Port

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

Figure 35 PMC-Parallel-TTL RX/TX FIFO Port

This port is used to make single-word accesses from the RX FIFO. Data read from this port will no longer be available for DMA transfers.

pmcparttl_ch0_rx_aecnt0,1,2

[0x8C, 94, 9C] RX almost-empty level 0 (read/write)

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

Figure 36 PMC-Parallel-TTL TX ALMOST EMPTY LEVEL register

This read/write port accesses the receiver almost-empty level register. When the number of data words in the receive data FIFO is equal or 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. 4k x32 is the RX FIFO per FIFO for a 12 bit valid count range [11-0]. Recommend to set to 16 of boundary for load function [0x10]

pmcparttl_ch0,_rx_afcnt0,1,2

[0x90, 98, A0] 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 37 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 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 is the RX FIFO for a 12 bit valid count range [11-0]. Recommend setting to 16 off the end count [0xFF0] for each FIFO to support loader function.

pmcparttl_ch0_rx_ffcnt

[0xA4] RX FIFO data count (read only)

RX FIFO Data Count Port	
Data Bit	Description
31-14	Spare
13-0	RX Data Words Stored

Figure 38 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 chain. The channel status register contains the combined pipeline and FIFO count. The size depends on the FIFO size. This design has 4095 locations possible in each FIFO for 12,285 total [0x2FFD].

Loop-back

The Engineering kit includes reference software, utilizing external loop-back tests. The PMC-Parallel-TTL BA17 uses Pn4.

PCIBPMC was used for the carrier using the rear SCSI connector tied to Pn4 on the BA17. HDEterm68 was used to provide the loop-back. SCSI cabling between the PCIBPMC and HDEterm68. The Pin numbers are for the interconnections on the HDEterm68. The IO names can be used to accommodate a different set-up.

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

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 39 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 40 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. Standard BA17 is –RP [Pn4 only]

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 41 PMC-PARALLEL-TTL FRONT PANEL Interface

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. Installed for BA17. 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 42 PMC-PARALLEL-TTL PN4 Interface

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 $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:	64 discrete IO channels. Each has a separate enable to control output. Inputs are maskable and always available. Upper 32 bits can be used with FIFO and auto-loader to capture transition information and timestamp.
CLK rates supported:	Osc, PLL, Osc and PLL programmed COS rate .
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:	All IO lines can be used as interrupt sources with programmable rising and or falling activity on IO line “COS”, DMA interrupts.
Onboard Options:	All Options are Software Programmable
Interface Options:	User IO routed to Pn4. 68 Pin SCSI III connector at front bezel not installed BA17. Ask for this option.
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 0-70ºC
PMC-Parallel-TTL-BA17 PMC Module with 64 IO channels, COS and direct IO, rear IO, 3.3V reference voltage to pull-ups, DMA support, 12Kx32 FIFO RX, Auto capture and DMA of COS data on upper 32 IO. Programmable triggers.
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. BA17 selection determines [-RP]

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. BA17 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, Windows Driver software, reference schematics. Recommended for first time purchases.
http://www.dyneng.com/pmc_parallel_TTL.html