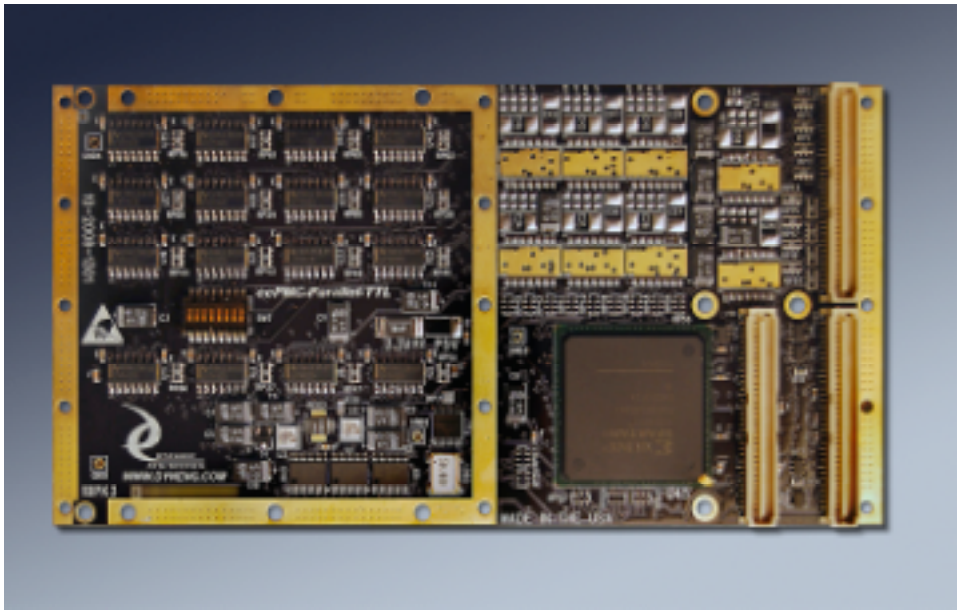
DYNAMIC ENGINEERING
150 DuBois St. Suite C, Santa Cruz, CA 95060
831-457-8891 Fax 831-457-4793
http://www.dyneng.com
sales@dyneng.com
Est. 1988

User Manual

ccPMC-PARALLEL-TTL-BA18

Digital Parallel Interface
Conduction Cooled PMC Module

32 COS Inputs with TimeStamp, FIFO Storage, DMA Transfer
8 ADC inputs with data compression



Manual Revision A1
Corresponding Hardware: Revision 1
10-2008-1301
FLASH 0204

ccPMC-PARALLEL-TTL-BA18

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 3/2/10

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. Frequently embedded systems need to measure analog signals as well.

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.

ccPMC-Parallel-TTL is configurable to have all Digital or a combination of Digital and Analog IO. Up to 64 TTL lines or 32 TTL and 8 ADC. Other combinations are available. The IO is available for system connection via the rear [Pn4] connector. The rear panel IO has a PIM and PIM Carrier available for rear panel wiring options. The design is conduction cooled and can be used in convection cooled environments as well.

ccPMC-PARALLEL-TTL-BA18 is a customerized version of the standard ccPMC-PARALLEL-TTL board. “BA18” 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]. 8 ADCs with PLL programmable rates are also installed. The sampled outputs are stored with a data compression algorithm.

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 BA18 features are selectable. The standard register based, and COS functions are available on unused [by BA18] pins and can be swapped with the BA18 functions under software control.

HDEterm68 http://www.dyneng.com/HDEterm68.html can be used as a breakout for the rear panel IO. 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. Please contact Dynamic Engineering with your custom requirements. BA18 is design number 2 for the ccPMC-PARALLEL-TTL with a corresponding FLASH of 02xx. Please note that the CardId on the PCI bus changes with each new design type. The revision in the PCI configuration space will match the revision in the FLASH design.

BA18 is an adaptation of the BA17 design; implemented on the PMC Parallel TTL card. The upper COS IO function was retained and the lower 32 IO converted to ADC inputs. The rest of the description paragraphs, bit maps, address maps, loop-back and connector information etc. are “tuned” to the BA18. The ccPMC Parallel TTL design has many options. For a more complete set please refer to the web page for this product.

Each channel is programmable to be input or output on a channel-by-channel basis. All 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.

The ADC function uses either the COS selected clock or PLLB further divided by a programmable amount. The channels can be synchronized to the COS, synchronized to each other or independent with the clock selections designed in.

The ADC’s can store the raw data or reduced data to FIFO. The reduction is done by using the current sample to test against limits. The limits are based on the current reference +/- a programmed threshold. SW can set the threshold. When a sample is out of the currently set range, the sample is stored and the time stored. The number of samples between changes can be determined with the timestamps. DMA can be used to transfer the data from the ADC channels to the host memory as separate fully independent functions. A multi-channel DMA controller makes this possible.

The TTL 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.

BA18 features a 32 bit TimeStamp counter. 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. When a selected bit change is detected the entire data word is stored into the FIFO plus the timestamp.

The registers are mapped as 32 bit words. 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.

ccPMC-PARALLEL-TTL is part of the PMC Module family of modular I/O components. The ccPMC-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 ccPMC-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 ccPMC-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. An option to add up to 8 ADC devices with up to 10 Mhz sampling rates is available. The BA18 version incorporates this option along with processing of the sampled data.

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 have added DMA, COS processing with timestamp and ADC.

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 channels to be synchronized if desired. The master enable can be programmed “on” to allow direct updates if 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. 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 [IO programmed to 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. The basic options are 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, receivers and Analog parts 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.

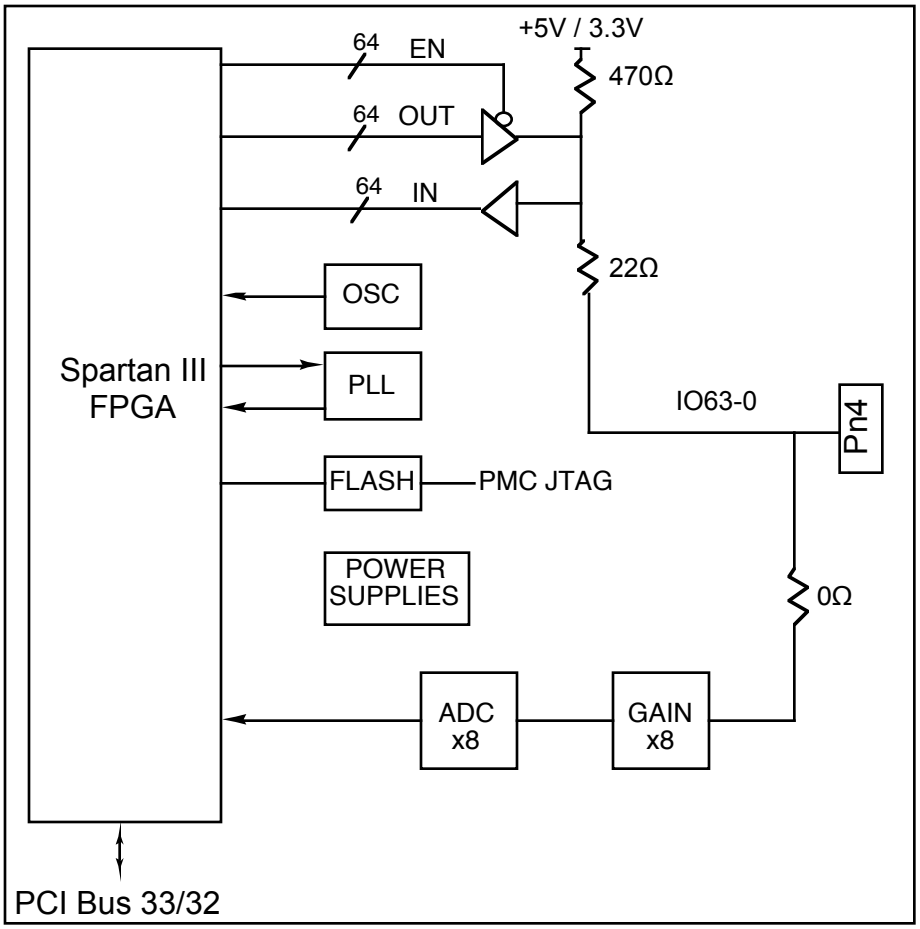


Figure 1 ccPMC-PARALLEL-TTL Block Diagram

PMC Parallel TTL BA18 features a programmable data path with DMA support. The internal block RAM is configured to provide FIFO's for each ADC and for the COS function. The COS function is supported with 16K x 32 FIFO and the 8 ADC are supported with 4K x 32. The ADC FIFO's are supported with DMA to move data automatically from the ADC after compression to the system memory.

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.

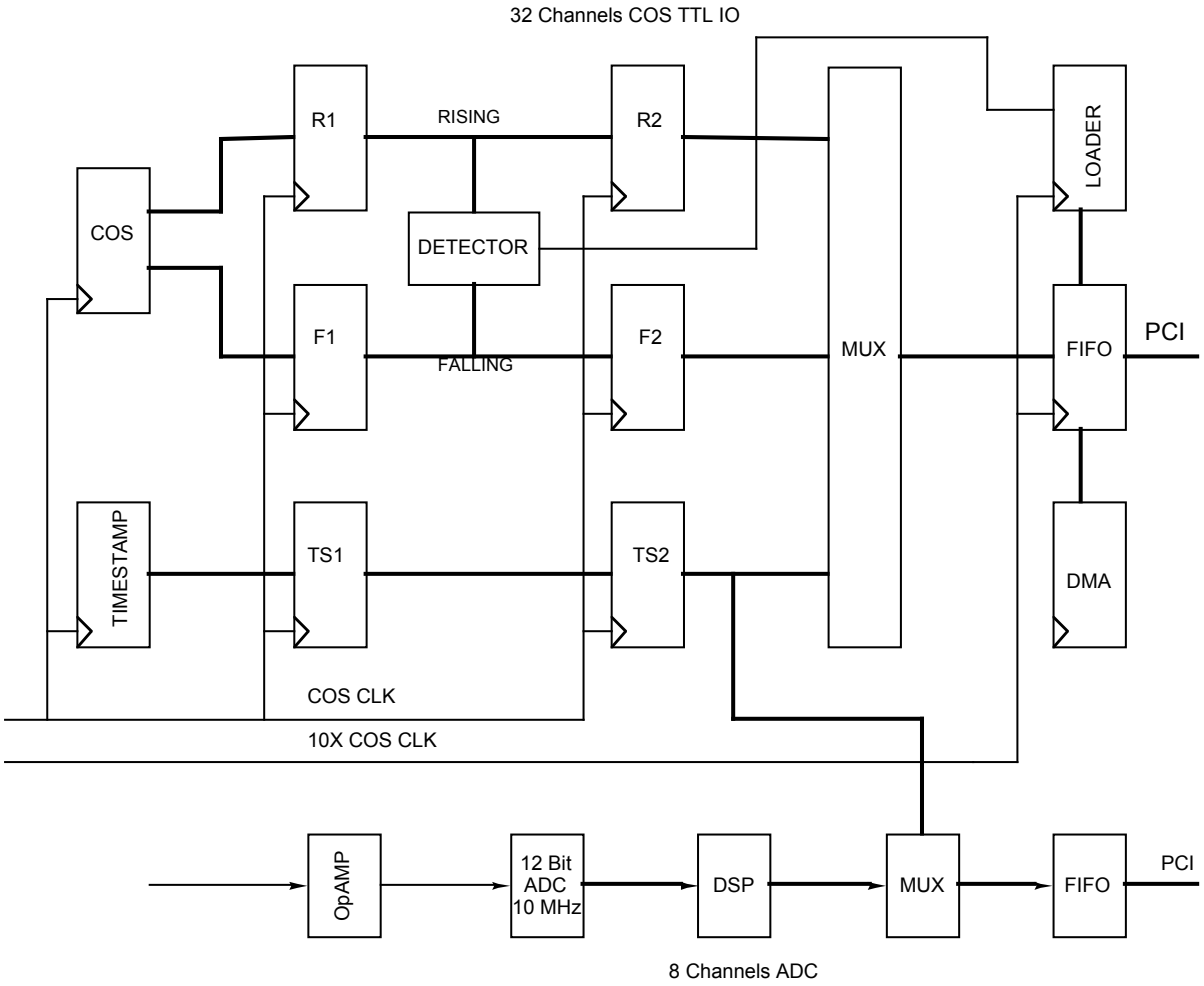


Figure 2 ccPMC-PARALLEL-TTL-BA18 Operational Block Diagram

With Lower frequencies of COS transitions and the deep memory the COS path can be handled with direct reads. The FIFO is 16Kx32 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 CPU can reduce the stored data. OS delays are the main culprit. 5513 transitions can be detected and stored before the FIFO is filled. 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 ADC’s are more likely to have larger amounts of data to process [unless the thresholds are set wide]. The FIFO’s are smaller since DMA can move the data more rapidly than the CPU direct reads used with the COS data. The PCI rate is more than 3x the fastest ADC sample rate and likely a multiple of that since some data reduction can be expected. The FIFO will act as a rubber-band for times when the PCI bus is not

immediately available for data movement – another channel or device is already using the bus etc.

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.

Since large overall transfer sizes can be created along with large segments within those transfers and no CPU intervention required during the loop the number of interrupts can be reduced compared to a model with a single DMA engine and SW intervention required to select the channel to operate on.

For implementations not requiring DMA a programmable interrupt is provided. The FIFO Interrupt Level register is programmed with the FIFO Data count that should correspond to an interrupt. When the FIFO and pipeline count match or exceed the programmed level the interrupt is asserted. A tight loop of FIFO data reads can then be used to capture the data.

Address Map

Function	Offset	
// PMC Parallel TTL definitions		
#define BA18_BaseBase	0x0000 // 0	PMC Parallel TTL base control register offset
#define BA18_BaseID	0x0004 // 1	PMC Parallel TTL ID Register offset
#define BA18_BaseSTATUS	0x0008 // 2	PMC Parallel TTL status Register offset
#define BA18_BaseDirU	0x000c // 3	PMC Parallel TTL Direction upper Register offset
#define BA18_BaseDatU	0x0010 // 4	PMC Parallel TTL Data upper Register, line data read
#define BA18_BaseDatUreg	0x0014 // 5	PMC Parallel TTL Data upper Register read-back
#define BA18_BaseRisUreg	0x0018 // 6	PMC Parallel TTL Rising upper Register
#define BA18_BaseFallUreg	0x001c // 7	PMC Parallel TTL Falling upper Register
#define BA18_BaseIntRisUreg	0x0020 // 8	PMC Parallel TTL Interrupt Enable Rising upper Register
#define BA18_BaseIntFallUreg	0x0024 // 9	PMC Parallel TTL Interrupt Enable Falling upper Register
#define BA18_BaseCosCntl	0x0028 // 10	PMC Parallel TTL Control register for COS
#define BA18_BaseStatusCos	0x002c // 11	PMC Parallel TTL COS FIFO Status
#define BA18_BaseFifoPAF	0x0030 // 12	PMC Parallel TTL COS FIFO Almost Full Level
#define BA18_BaseFifoWordCnt	0x0034 // 13	PMC Parallel TTL COS FIFO Data available count
#define BA18_BaseFifoRd	0x0038 // 14	PMC Parallel TTL COS FIFO Read Port
#define BA18_BaseIntRisUstat	0x003c // 15	PMC Parallel TTL Interrupt Rising UPR Stat Rd
#define BA18_BaseIntFallUstat	0x0040 // 16	PMC Parallel TTL Interrupt Falling UPR Stat Rd
#define BA18_BaseTIMESTAMP	0x0044 // 17	PMC Par TTL TimeStamp Preload and readback
#define BA18_BaseTIMESTAMPCNT	0x0048 // 18	PMC Par TTL TimeStamp current count

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

The address map provided is for the local decoding performed within ccPMC-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 = 0x003B for the ccPMC-PARALLEL-TTL-BA18.

Function	Offset
// PMC Parallel TTL definitions	
#define BA18_ChanCntrl	0x0000 // 0 PMC Par TTL BA18 General Channel Control Register
#define BA18_ChanStatus	0x0004 // 1 PMC Par TTL BA18 Channel Status, Register
#define BA18_Chanbrstin	0x0008 // 2 PMC Par TTL BA18 burst in control, unused this version
#define BA18_Chanbrstout	0x000C // 3 PMC Par TTL BA18 burst out control – Write
#define BA18_ChanAdcFifoCnt	0x000C // 3 PMC Par TTL ADC Samples Available to read
#define BA18_ChanFifoSwr	0x0010 // 4 PMC Par TTL FIFO Single word read port
#define BA18_ChanAfl	0x0018 // 6 PMC Par TTL almost full count register and rd-bk
#define BA18_ChanAdcCntl	0x0024 // 9 PMC Par TTL ADC Control Port
#define BA18_ChanAdcTol	0x0028 // 10 PMC Par TTL ADC Tolerance Port

Figure 4 ccPMC-PARALLEL-TTL Channel Address Map

Please note that the channel offsets are relative to the starting register for each channel.

Name	Offset [Reg 0]	Registers
Base	0x00	0-19
Channel 0	0x50	20-39
Channel 1	0xA0	40-59
Channel 2	0xF0	60-79
Channel 3	0x140	80-99
Channel 4	0x190	100-119
Channel 5	0x1E0	120-139
Channel 6	0x230	140-159
Channel 7	0x280	160-179

Programming

Programming the ccPMC-PARALLEL-TTL-BA18 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 ccPMC-PARALLEL-TTL-BA18 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 ccPMC-PARALLEL-TTL-BA18. Please read the register definitions and programming hints to determine the correct settings for your use. In some cases the PLL will need to be programmed and selections made in more than one register to coordinate operation.

For Windows™ and Linux systems the Dynamic Driver can be used. The Dynamic Driver will take care of the interaction with the OS for you as well as the initial steps of initialization. Calls can be made to each of the registers to complete initialization and progress to application use. Reference software is included showing how to use each of the driver calls and to do loop-back testing. Utilities for programming the PLL 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.

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.

To use the BA18 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 BA18 to start the DMA. Please refer to the Burst IN and Burst Out register discussions.

To use the COS engine, set the Rising and Falling enables for the IO of interest 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 in the

order: Rising, Falling, and TimeStamp. The TimeStamp should also be initialized prior to starting.

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 COS clock is 1/10 of the reference clock selected]

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 ADC channels are supported with DMA. DMA should be set-up before starting the ADC function to prevent a potential overrun if the system does not complete the initialization in a timely manner.

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 referred to as “ping-pong”.

The internal HW will automatically arbitrate between the active ADC channels and use the individual DMA engines to move data to the host. Each channel should have its own memory space defined in the system memory. Since the channels have separate DMA controllers the size of a transfer request is not limited to the FIFO size. Use a length that makes sense for your system. Large for continuous capture and post processing, smaller for real time processing in parallel with the data capture, or specific lengths for defined algorithm requirements.

Base Register Definitions

BA18_BaseBASE

[$00 Base Control Register Port read/write]

DATA BIT	DESCRIPTION
31-21	spare
20	bit 19 read-back of pll_dat register bit
19	pll_dat [write to PLL, read-back from PLL]
18	pll_s2
17	pll_sclk
16	pll_en
15	Master ADC Enable Channel 7
14	Master ADC Enable Channel 6
13	Master ADC Enable Channel 5
12	Master ADC Enable Channel 4
11	Master ADC Enable Channel 3
10	Master ADC Enable Channel 2
9	Master ADC Enable Channel 1
8	Master ADC Enable Channel 0
7-2	spare
1	Force Interrupt
0	Master Interrupt Enable

Figure 5 ccPMC-PARALLEL-TTL Base Control Port Bit Map

This is the base control register for the ccPMC Parallel TTL BA18. 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. Affects Force Interrupt, and COS Rising and COS Falling interrupt sources. Channel and PFC interrupts are controlled separately.

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 ADC Enable (0-7) when set provides a secondary enable to each ADC channel to operate. Leaving cleared until after the channels are initialized and then enabling together allows for synchronized sampling. For independent operation these bits can be enabled first.

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.

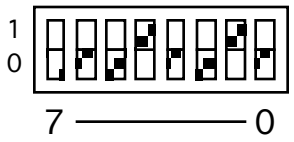
BA18_BaseID

[$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 ccPMC-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 BA18 design is 0x02 the current revision is 0x03.

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.

BA18_BaseSTATUS

[$08 Board level Status Port read only]

DATA BIT	DESCRIPTION
31	Interrupt Status
30-24	'0'
23	Int7
22	Int6
21	Int5
20	Int4
19	Int3
18	Int2
17	Int1
16	Int0
15	COS FIFO Level Interrupt
14-6	spare
5	INTR Falling
4	INTR Rising
3-1	'0'
0	local interrupt

Figure 7 ccPMC-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.

COS FIFO Level Interrupt – This is the local masked Programmable FIFO Count interrupt. When the COS FIFO has a "level" greater or equal to the programmed count the interrupt status is set. If the interrupt is enabled an interrupt request is generated. The unmasked version of this signal is available in the COS status register. The Interrupt enable is set in the COS control register.

Int7-0 = Channel interrupt requests after the channel masks. Please refer to the channel registers for the definitions of these interrupts. The status is provided to allow the system to determine if this board has an active interrupt in one read. Control and

additional status is available in the channel control registers.

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.

BA18_BaseDirU

[$0C Direction Register bits 63-32 read – write]

DATA BIT	DESCRIPTION
31-0	DIR63-32

Figure 8 ccPMC-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.

BA18_BaseDatU

[$10 Data IO Port read/write]

DATA BIT	DESCRIPTION
31-0	Data IO 63-32

Figure 9 ccPMC-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.

BA18_BaseDatUreg

[$14 Data Reg Port read only]

DATA BIT	DESCRIPTION
31-0	Data IO 63-32

Figure 10 ccPMC-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.

BA18_BaseRisUreg

$18 Rising Upper Control Register Port read/write

DATA BIT	DESCRIPTION
31-0	Rising 63-32

Figure 11 ccPMC-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.

BA18_BaseFallUreg

$1C Rising Upper Control Register Port read/write

DATA BIT	DESCRIPTION
31-0	Falling 63-32

Figure 12 ccPMC-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.

BA18_BaseIntRisUreg

$20 Rising Interrupt Upper Control Register Port read/write

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

Figure 13 ccPMC-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.

BA18_BaseIntFallUreg

$24 Falling Interrupt Upper Control Register Port read/write

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

Figure 14 ccPMC-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.

BA18_BaseCosCntl

[$28 COS clock definition port read -write]

DATA BIT	DESCRIPTION
14	Clock Select
13-12	Spare
11	PFC Interrupt Enable
10-9	Spare
8	Test Select
7	Enable
6-1	Spare
0	FIFO Reset

Figure 15 ccPMC-PARALLEL-TTL COS Control Bit Map

FIFO Reset when '1' causes the COS FIFO to be reset. When '0' the FIFO is enabled. Power on reset also clears the FIFO. The FIFO is made up of Block RAM using pointers to load and unload. The reset clears the pointers back to the initial location.

Enable starts the COS detection control. The selections for IO, Rising and Falling should be made prior to starting the COS engine. The control signal is automatically synchronized to the selected clock for proper operation.

Test Select when set pushes a copy of the selected COS clock onto signal DataOut32. If the IO is selected as an output the reference clock can be checked with an ocilloscope to validate the programming files in use.

PFC Interrupt Enable when set allows the PFC Interrupt to propagate to the host. The interrupt is the Programmable FIFO Count. When the level in the COS FIFO matches or exceeds the programmed count an interrupt can be generated. The masked version is available in the BaseStatus register. The unmasked version is available in the BaseStatusCOS register.

Clock Select when '0' selects the reference Oscillator to use with the COS system. The frequency is used along with a divided by 10 version. The oscillator is set to 50 Mhz. The COS is set to 5 MHz with this selection. When set to '1' the PLL output A is used. The PLL output A can be programmed to a user specified frequency. The COS will operate at the PLLA/10 rate. The PLL should be programmed and settled prior to enabling the COS function.

BA18_BaseStatusCos

[$2C COS Status Port]

DATA BIT	DESCRIPTION
31	PFCInterrupt masked
30	PFC Interrupt Unmasked
29-16	COS Data Count
15-12	gnd
11	COS FIFO Error
10-4	gnd
3	COS FIFO Full
2	spare
1	gnd
0	COS FIFO MT

Figure 16 ccPMC-PARALLEL-TTL COS Status Control Bit Map

COS FIFO MT when ‘1’ means the FIFO has no data stored. When ‘0’ at least 1 data is stored.

COS FIFO FULL when ‘1’ means the FIFO is full. When ‘0’ the FIFO is less than full.

COS FIFO ERROR is a sticky bit which is set when the FIFO is FULL when more data is to be written into the FIFO. The bit is cleared when this port is written to with the same location set.

The COS Data Count can be used to determine the number of data stored. The count is also available in the FIFO Word Count register.

When the COS DATA Count matches or exceeds the programmed FIFO Count the PFC Interrupt request is valid [unmasked]. If the control bit enabling the interrupt is also set the interrupt is generated. The interrupt operates as a level. Read the FIFO to reduce the count below the threshold to remove the interrupt, or clear the enable bit in the control register.

BA18_BaseFifoPAF

[$30 COS Programmable FIFO Almost Full Level]

DATA BIT	DESCRIPTION
15-14	set to 0, spare bits
13-0	PAF

Figure 17 ccPMC-PARALLEL-TTL COS Status Control Bit Map

Program PAF with any value to match against the FIFO data count. Each data written to the FIFO increments the count by 1. Each data read reduces the count by 1. The count is matched against the PAF to determine if an interrupt should be generated. The Status can be used masked for interrupts or unmasked for polled operation. The Interrupt enable must be set for the interrupt to be generated. The reset level is '0'. Program this register before enabling the interrupt request for PFC.

BA18_BaseFifoWordCnt

[$34 COS FIFO Word Count

DATA BIT	DESCRIPTION
15-14	gnd
13-0	COS FIFO Count

Figure 18 ccPMC-PARALLEL-TTL COS Status Control Bit Map

The FIFO count is the number of words in the COS FIFO. This count is duplicated in the COS Status register. It is provided in this register for convenience – no shifting is required.

BA18_BaseFifoRd

[$38 COS Read the FIFO port]

DATA BIT	DESCRIPTION
31-0	FIFO DATA

Figure 19 ccPMC-PARALLEL-TTL COS FIFO Data Read

COS FIFO is read from this port. Data is stored in three consecutive words – Rising, Falling and TimeStamp. The data is read out in the same order. The COS data is triggered to be stored by the programmed for capture. When captured, all of the bits are stored, and the status cleared. The TimeStamp will be accurate for the bits programmed for capture. The other bits time will be between the last TimeStamp and the current one since they have been captured and held until one of the trigger bits is active.

BA18_BaseIntRisUstat

$3C Rising Status Upper Control Register Port read/write

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

Figure 20 ccPMC-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.

BA18_BaseIntFallUstat

$40 Falling Status Upper Control Register Port read/write

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

Figure 21 ccPMC-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.

BA18_BaseTIMESTAMP
[$44 TimeStamp Preload read – write]

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

Figure 22 ccPMC-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.

BA18_BaseTIMESTAMPCNT
[$48 TimeStamp Count read only]

DATA BIT	DESCRIPTION
31-0	Current TimeStamp Count

Figure 23 ccPMC-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 BA18 timer is likely more accurate than the system timer.

Channel Register Definitions

BA18_ChanCntl

[0x00] Channel Control Register (read/write)

Channel Control Register

Data Bit	Description
31-9	spare
8	OutUrgent
7	reserved
6	Write DMA Interrupt Enable
5	reserved
4	Channel Force Int
3	Channel Int En
2	reserved
1	ADC FIFO Reset
0	reserved

FIGURE 24 CCPMC-PARALLEL-TTL CHANNEL CONTROL REGISTER

ADC FIFO Reset: When set to a one, the receive FIFO will be reset. When zero, normal FIFO operation is enabled. In addition the RX State Machine is also reset.

Write DMA Interrupt Enable: When set to one, enable the interrupt for DMA writes. The DMA interrupt is 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.

OutUrgent when set causes this channel to have higher priority for DMA processing. If one of the channels is set to a higher sampling rate or for some reason needs higher

priority set this bit. If not needed leave cleared to allow equal opportunity with the other channels.

BA18_ChanStatus

[0x04] Channel Status Read/Clear Latch Write Port

Channel Status Register

Data Bit	Description
31	Channel Interrupt Active
30	LocalInterrupt
29-23	gnd
22	BurstOutIdle
21	StandardIdle
20	AlternateIdle
19	gnd
18	FifoOvflowLat
17-16	gnd
15	reserved
14	Write DMA Interrupt Occurred
13	reserved
12	Write DMA Error Occurred
11	ChFifoIntLat
10-8	gnd
7	spare
6	ChFifoFull
5	ChFifoAfl
4	ChFifoMt
3-0	gnd

Figure 25 ccPMC-PARALLEL-TTL Channel STATUS PORT

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

Channel FIFO Almost Full: When a one is read, the number of data words in the data FIFO is greater than or equal to the value written to the corresponding ChanAfl register; when a zero is read, the FIFO level is less than that value. The count used from the FIFO includes the DMA pipeline [up to 4 positions]

Channel 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.

Channel FIFO Interrupt Latched when set indicates that the FIFO level reached the defined almost full level. This is a sticky bit. It will remain set until explicitly cleared by writing back to this port with the corresponding bit set.

Write DMA Error Occurred: When a one is read, a write 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 DMA Interrupt Occurred: When a one is read, a write 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.

FIFO Overflow Error Occurred: When a one is read, an error has been detected. This will occur if the FIFO 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.

Alternate Idle State: When the state machine for the Alternate data processing is in the "Idle" state this bit is set. The Alternate data processing is the data reduction mode where data is compressed via comparison to reference levels and storage only when data is outside of the current range. Use this bit to know when processing has started or stopped. Depending on frequencies chosen there may be some delay in start-up or shut-down.

Standard Idle State: When the state machine for the standard double packed data is in the "Idle" state this bit is set. Use this bit to know when processing has started or stopped. Depending on frequencies chosen there may be some delay in start-up or shut-down.

BurstOutIdle: When set the DMA state-machine is in the Idle state and ready for a new command. If a DMA transfer is aborted this bit should be checked before starting a new transfer.

LocalInterrupt: when set any of the non-DMA interrupts are active using the discrete interrupt enables as masks. Localinterrupt is before the channel master interrupt enable and can be used for polling if the master is not enabled.

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. This bit includes DMA and non DMA interrupts and is after the channel master interrupt enable mask.

BA18_Chanbrstin

[0x08] 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 26 ccPMC-PARALLEL-TTL Write DMA pointer register

UNUSED for BA18: 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.

BA18_Chanbrstout

[0xC] 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 27 ccPMC-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.

BA18_ChanAdcFifoCnt

[0x0C] FIFO Count (read)

RX Almost-Empty Level Register

Data Bit	Description
31-16	Spare
15-0	FIFO Count

Figure 28 ccPMC-PARALLEL-TTL Channel Data Count register

This read only port allows the user to determine the amount of data stored into the FIFO and DMA Pipeline. Reading this port allows for a defined DMA transfer with a known length or single word reads in a defined loop without the need for additional status reads to prevent underflow. The count can be used to cause an interrupt when the count exceeds the level programmed into the ChanAfl register.

Larger DMA transfers can be done without using this register directly. The hardware will automatically move bursts of data until the programmed DMA count is completed. This can take many FIFO's worth of data and is the preferred method when larger transfers are needed. The count and discrete transfer methods are used when smaller data sets are needed and processed in parallel with data capture. For example FFT processing.

The count is valid for either the Alternate or Standard data processing. The count is the number of long words. In the Standard case the data is double packed so the count = 1/2 the data available. In the Alternate case the data is compressed, and the Time Tag added so the count equals the 2x the data available [alternate words have data and TimeTag].

BA18_ChanSwr

[0x10] Write TX/Read RX FIFO Port

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

Figure 29 ccPMC-PARALLEL-TTL RX 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.

The FIFO will have bits 11-0 = ADC data, 12 = overflow bit sign extended with '0' for the first write, and 31-0 = TimeTag for the second write per sample when in Alternate processing mode. One sample takes two FIFO locations.

In Standard mode 11-0 = ADC Data, 12 = overflow bit for first sample, 27-16 = ADC data, 28 = overflow bit for sample 2. Two samples per FIFO location.

BA18_ChanAfl

[0x18] 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 30 ccPMC-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 plus pipeline 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].

BA18_ChanAdcCntl

[0x24] Channel ADC Control data count (read/write)

ADC Control Port

Data Bit	Description
31-24	Spare
23-16	ClkDiv
15-6	Spare
5	AdcOverFlowIntEn
4	AdcLatFifoIntEn
3	AdcFifoIntEn
2	AdcLocalEn
1	AdcFifoDataSelect
0	AdcClkSel

Figure 31 ccPMC-PARALLEL-TTL ADC Control Bit Map

ClkDiv[7-0] are the clock divisor select bits. The clock source is divided by an 8-bit counter. The output frequency is {reference / [2(n+1)]}, n≥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.

The default value of the register is 0x00310000 which sets the divisor to 100 to make sure the ADC's are not overdriven in case the PLL is initialized before the divisor is selected.

The Clock used for the counter controlled by ClkDiv is PLLB. This allows all 8 ADC's to use the same base reference clock and different local divisors.

AdcClkSel: When set to 1 selects the Alternate reference clock which is the COS clock for this design. When set to 0 the PLLB clock after division is selected. The clock is used by the ADC for sampling and by the state-machines for processing the data.

AdcFifoDataSelect: When 1 use the Alternate Data reduction algorithm. When 0 use the Standard double packed data mode. Please see the AdcTol register description for the Alternate method particulars. The standard mode samples data and packs it two

samples per 32 bit word then loads to the FIFO. The first sample is loaded in the bottom word 15-0 and the second sample on the upper word 31-16. Data is sampled and stored continuously under the AdcLocalEn control.

AdcLocalEn: When '1' enables the selected state-machine to fill the FIFO with data from the ADC. AdcLocalEn should be disabled when changing between processing methods.

AdcFifoIntEn: When '1' enables the Channel FIFO Almost Full unlatched interrupt to be active. This interrupt will be active anytime the FIFO is at or above the programmed Almost Full level. The interrupt will stay asserted unless sufficient data is read to reduce the level below the threshold or the enable is removed.

AdcLatFifoIntEn: When '1' enables the Channel FIFO Almost full Latched interrupt to be active. This interrupt will be active when the FIFO plus Pipeline count is at or above the programmed FIFO Level and will stay active until the bit is cleared. The bit will not be reasserted unless the level goes below the programmed count and then rises above the programmed count again. The bit is cleared by writing to the Status Register with the Latched Status bit set. The interrupt will be cleared even if the level is still above the threshold.

AdcOverFlowIntEn: When '1' will enable an interrupt to be asserted if the FIFO is full when the state-machine is attempting to write to the FIFO. The interrupt is cleared by clearing the OverFlow status bit in the status register. This bit should not be set in normal operation as it represents a loss in data.

BA18_ChanAdcTol

[0x28] ADC Tolerance Register (read / Write)

FIFO Level Interrupt Register	
Data Bit	Description
27-16	AdcToleranceHighSide
11-0	AdcToleranceLowSide

Figure 32 ccPMC-PARALLEL-TTL ADC Tolerance Register

When the Alternate data reduction option is selected the ADC data is run through a DSP algorithm to reduce the sample count. The initial sample is used for the initial reference and stored into the FIFO. Subsequent samples are measured against the threshold plus the AdcToleranceHighSide and minus the AdcToleranceLowSide. If the Addition of the Tolerance would go above the max value or the subtraction of the Tolerance would go below the minimum value the threshold is automatically adjusted to stay in bounds. This adjustment protects against undesired roll-over affects.

When a new sample is outside of the range established with the reference and thresholds, the new sample is stored into the FIFO and replaces the previous reference sample. The range is updated to reflect the new reference. Sampling continues. The State Machine operates at a 10x rate and uses a Pipeline to allow the processing to take place without losing any samples.

The thresholds can be adjusted to provide more or less data compression.

Loop-back

The Engineering kit includes reference software, utilizing external loop-back tests. The ccPMC-PARALLEL-TTL BA18 uses Pn4.

PCIBPMC was used for the carrier using the rear SCSI connector tied to Pn4 on the BA18. 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.

The upper IO are the Parallel port and the lower IO correspond to the ADC inputs for this design. The Parallel Port is tested with loop-back using some of the IO as outputs to drive the remaining inputs and then reversing. The ADC's are tested with an analog input.

The Dynamic Driver for Window® is used along with the reference software included in the engineering kit for the BA18 to do the testing.

Signal	From	To	Signal
IO_48	pin 25	pin 17	IO_32
IO_49	pin 59	pin 51	IO_33
IO_50	pin 26	pin 18	IO_34
IO_51	pin 60	pin 52	IO_35
IO_52	pin 27	pin 19	IO_36
IO_53	pin 61	pin 53	IO_37
IO_54	pin 28	pin 20	IO_38
IO_55	pin 62	pin 54	IO_39
IO_56	pin 29	pin 21	IO_40
IO_57	pin 63	pin 55	IO_41
IO_58	pin 30	pin 22	IO_42
IO_59	pin 64	pin 56	IO_43
IO_60	pin 31	pin 23	IO_44
IO_61	pin 65	pin 57	IO_45
IO_62	pin 32	pin 24	IO_46
IO_63	pin 66	pin 58	IO_47

In Addition ADC0-7 are tied to an analog voltage reference to support the ADC tests.

PMC Module Logic Interface Pin Assignment

The figure below gives the pin assignments for the PMC Module PCI Pn1 Interface on the ccPMC-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 33 ccPMC-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 ccPMC-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 34 ccPMC-PARALLEL-TTL Pn2 Interface

PMC Module Backplane IO Interface Pin Assignment

The figure below gives the pin assignments for the PMC Module IO Interface on the ccPMC-PARALLEL-TTL and routed to Pn4. Also see the User Manual for your carrier board for more information.

ADC0	AGNDA	1	2
		3	4
ADC1	AGNDB	5	6
		7	8
ADC2	AGNDC	9	10
		11	12
ADC3	AGNDD	13	14
		15	16
ADC4	AGNDE	17	18
		19	20
ADC5	AGNDF	21	22
		23	24
ADC6	AGNDG	25	26
		27	28
ADC7	AGNDH	29	30
		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 35 ccPMC-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.

Series resistors are used and can be specified to be something other than the 22 ohm standard value for the TTL IO.

The ADC's have opamp's with scaling resistors to allow alternate voltage ranges to be implemented. The reference grounds are separated and routed as mini-places between the connector and analog circuitry. The grounds can be tied together externally or kept isolated as your system requires. The grounds are tied to the local digital ground through filtered beads at the ADC circuit. The beads can be replaced with capacitors for an AC coupled ground system or through 0 Ω resistors for a non filtered DC system. The Ferrite beads are the default installation.

IO is through connector Pn4 and will route through the carrier to the backplane. PIM carriers / adapters can be used in cPCI and VME type systems. Custom cables can be assembled for your system. Please contact Dynamic Engineering with your requirements.

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 ccPMC-PARALLEL-TTL is constructed out of 0.062 inch thick high temperature ROHS compliant material.

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 ccPMC-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. The design is implemented to operate in a conduction cooled environment. Convection cooling may also be used.

Thermal planes are designed in along with ties between the conduction-cooling plane and exposed gold contacts for the conduction cooling path. Thermal pads are added to capture component heat and conduct to the carrier the board is mounted to.

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:	32 discrete IO channels. Each has a separate enable to control output. Inputs are maskable and always available. Can be used with FIFO and auto-loader to capture transition information and timestamp.
Analog IO	8 ADC 12 bit 10 MHz rated with individual FIFO and DMA capability. Programmable data reduction capability.
CLK rates supported:	Osc, PLL selectable with software. PLL programmable with SW.
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. FIFO level interrupts.
Onboard Options:	All Options are Software Programmable
Interface Options:	IO routed to Pn4. PIM carrier and module available.
Dimensions:	Standard Single ccPMC 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 -40-85ºC
ccPMC-PARALLEL-TTL-BA18 PMC Module with 32 IO channels, COS and direct IO, rear IO, 3.3V reference voltage to pull-ups, DMA support, 16Kx32 FIFO RX, Auto capture, Programmable triggers. Independent DMA of ADC data. Eight 12 bit 10 MHz ADC's.
http://www.dyneng.com/ccpmc_parallel_ttl.html

Order Options:

-CC to add conformal coating
-3V to change from 5V IO reference to 3.3V IO reference
Shown for reference. BA18 selection determines [-3V]

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

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

ccPMC Parallel IO Eng Kit : Windows Driver software: Recommended for first time purchases. http://www.dyneng.com/ccpmc_parallel_ttl.html