

CERN Trajectory Measurement System PUPE-FPGA Specification

Dr Andrew C McCormick, Alpha Data
Version 0.3: 08/11/2006

Introduction

This PUPE FPGA document specifies the general structure and operation of the PUPE FPGA firmware and the register interface accessible by the software. It is based upon the technical specification presented in [1] and tender document [2].

Overview

The detailed operation of the TMS cycles is described elsewhere [1,3]. Each FPGA contains 3 independent pick-up processing units. Each of these modules performs the following functions: phase locking (to particle bunches or a reference signal); baseline restoration and integration of the sum signal (Σ) and horizontal and vertical difference signals ($\Delta x, \Delta y$); recording of the integrated signals in a 256MB circular buffer; capturing of RAW signal data for diagnostic purposes.

The operation of the baseline restoration and integration is controlled through programmable phase tables, indicating when to start and stop integration tasks within each orbit.

The cycle goes through a number of states, with state changes indicated by a number of timing signals. The phase table and other synchronisation parameters depend on this state, and so a programmable state table is provided to select the correct phase table and the other parameters for that state.

The data may not be arranged in the best order for efficient DMA transfer over the PCI bus. Therefore a Data Pattern Engine is included to allow copying the data to a "scratch" buffer in another bank of SDRAM, before it is transferred across the PCI bus.

The timing signals coming into the module can be overridden by software to allow testing of the individual unit.

There are also a number of modules external to the pick-up processing units within the FPGA:

A timing interface module is used at the FPGA level, to control the timing bus between FPGAs in the system, since the timing signals come into one FPGA and must be broadcast to all the others.

A test memory bank can be used to inject synthetic test signals in place of the ADC inputs. This memory bank is also used as scratch memory for the data pattern engine.

FPGA level control is also provided for the 9 channel ADC inputs and for handling interrupts.

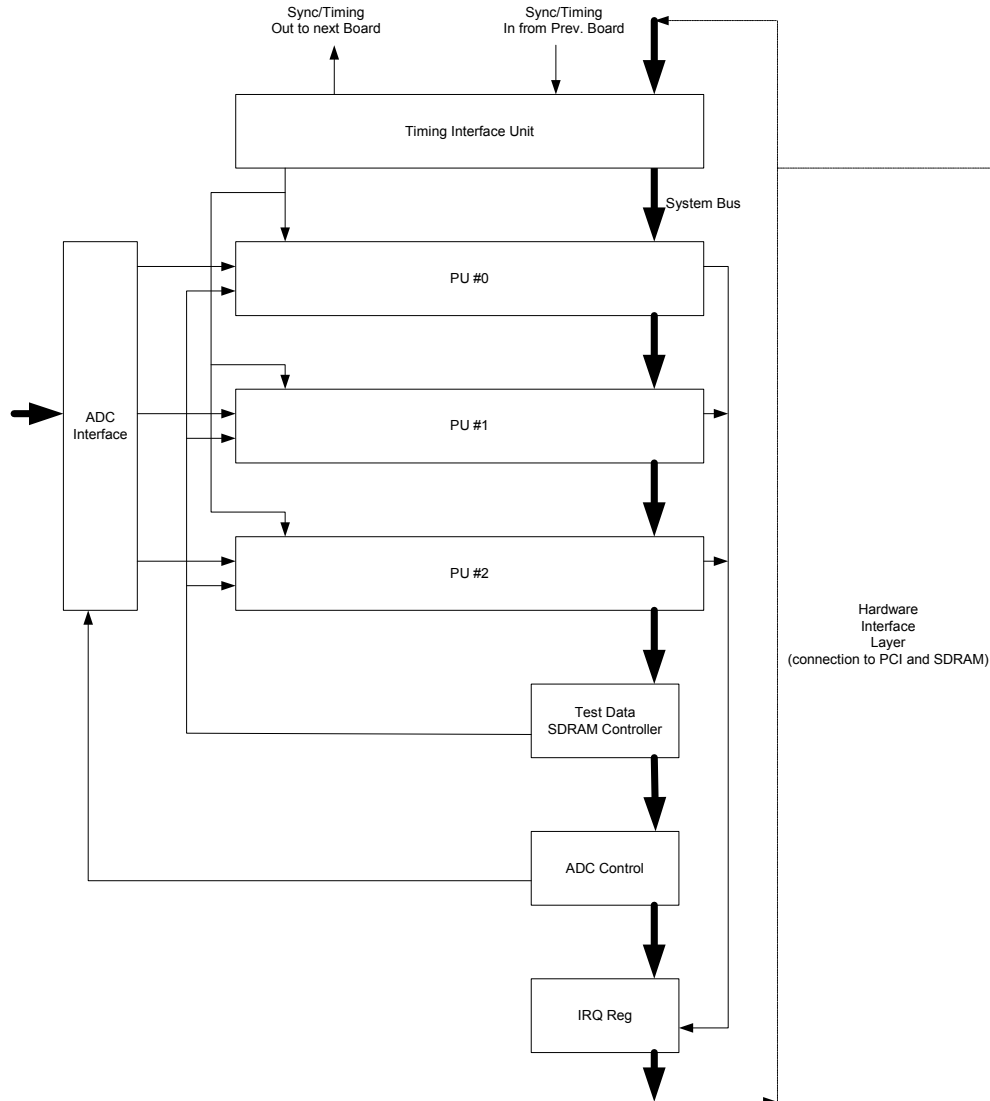


Figure 1: FPGA Top Level Design

Figure 1 shows the main elements in the FPGA design including the 3 processing units (PU's). The interfacing across the PCI bus and the interfacing with the SDRAM use an existing design library which provides simple implementation of registers, PCI accessible dual port Block RAM , PCI accessible shared access SDRAM as well as interrupt handling capabilities.

Processing Units

Each individual processing unit performs a number of tasks:

Timing and input data can be set to use either real data or test values, programmable by software.

The PLL-SPU unit performs the main processing tasks, locking onto the phase and integrating the signals.

The data logger, records the outputs of the PLL-SPU unit and writes them to SDRAM. It also updates a cycle timing table every 1ms with references to the SDRAM cycle data table, to aid the software in finding the required data. The data logger also acts as an embedded logic analyser, capturing the RAW data input signals and control signals for diagnostic purposes.

The data pattern engine reads the SDRAM at programmable intervals to capture information pertaining only to a specific bunch, and format it into a contiguous block for easier transfer and processing by the host.

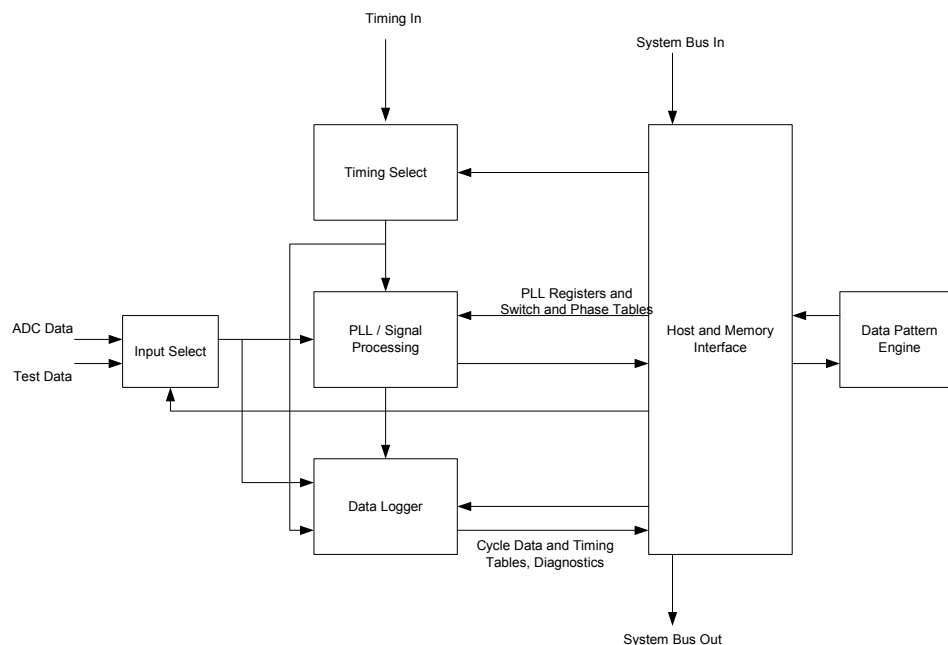


Figure 2: PU Core

Figure 2 shows the structure of the processing unit. A VHDL library is used to provide an abstract interface between the application and the PCI bus/SDRAM.

PLL-SPU

The core processing occurs in the PLL-SPU (phase locked loop and signal processing unit).

The PLL component will follow the specification in [1], fig 12 as closely as possible. The baseline restoration and integration will follow the specification [1] fig.9 and example VHDL [4] as closely as possible.

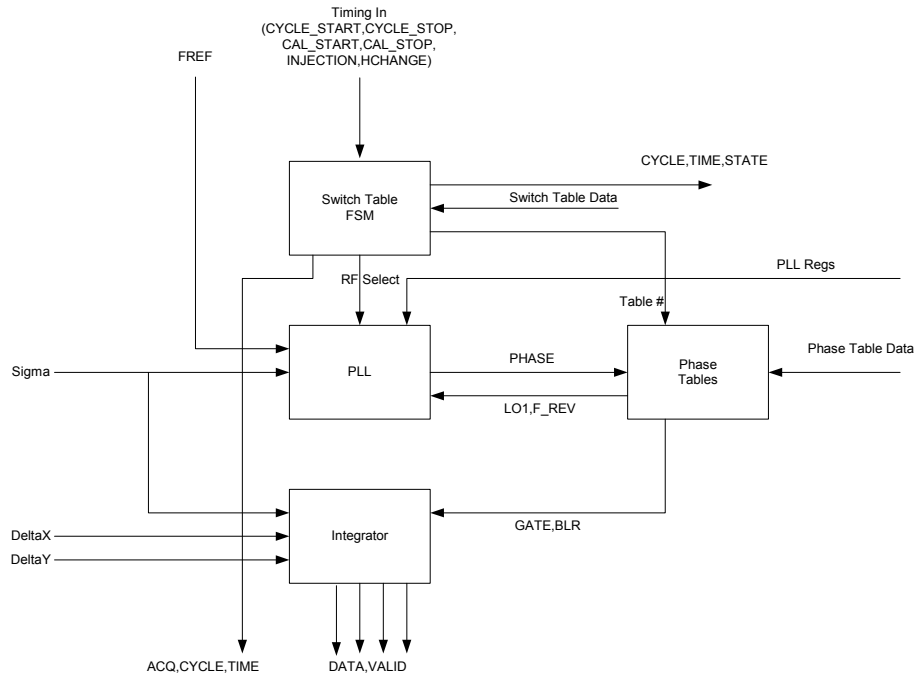


Figure 3: PLL-SPU

Figure 3, shows the structure of the PLL-SPU block.

The phase table block uses a single 8192 x 8 bit block RAM to store up to 16 phase tables. The read address is generated by concatenating the phase (least significant 9 bits) and the table number (most significant 4 bits), to output the LO,F_REV,GATE and BLR signals.

The switch table FSM controls the PLL based on the timing information signals. The FSM has 16 states of which 14 are programmable. States 0xE and 0xF are reserved for Error and Finished/Idle states. The state number also corresponds to the Phase Table number.

The state machine is initially set to state 0xF, where it waits until the CYCLE_START event occurs, up which it switches to state 0x0. If the machine ends up in state 0xE, a register write can move it to state 0xF, to clear the error. Other state transitions depend on the contents of the switch table.

The switch table consists of 14 elements of width 28 bits. For each table element bits 0-7 are used as control bits, e.g. bit 0 sets the ACQ signal to enable the acquisition of data, bit 1 sets the RF select bit in the PLL. Other bits are not yet currently defined, but some may be used to trigger interrupts (bits 5-7).

Bits 8-27 are used for 5 state transitions, indicating the state to change to when one of 5 timing events occurs:

Bits 8-11 : CYCLE_STOP

Bits 12-15 : CAL_STOP

Bits 16-19 : CAL_START

Bits 20-23 : INJECTION

Bits 24-27 : HCHANGE

A typical switch table set up could be:

State	HC	INJ	CaStrt	CaStp	CyStp	RF	ACQ	Hex Value
0	E	3	1	E	E	0	0	E31EE00
1	E	E	E	2	E	0	1	EEE2E01
2	E	3	E	E	E	0	0	E3EEE00
3	4	E	E	E	F	1	1	4EEEF03
4	5	E	E	E	F	1	1	5EEEF03
5	6	E	E	E	F	1	1	6EEEF03
6	E	E	E	E	F	1	1	EEEF03
E	X	X	X	X	X	0	0	0000000
F	X	X	X	X	X	0	0	0000000

This allows the calibration to run with the phase information in table 1. After injection (whether Calibration occurs or not), phase table #3 is used. H-Changes can then move on to using phase tables, 4,5 and 6. Out of sequence timing events send the state machine to the Error state.

Bunch Mean Value Computation

Mean values of the beam and selectable bunches will also be computed at the bunch rate. 2 bunch mean bits will be added to the phase table to use as enable strobes to push data into the filters used to obtain the mean values. These will be set at the end of the gate pulse of any bunch to be added to the mean computation. Every 1ms, the output of these filters will be sampled and stored in a bunch mean table.

FPGA Memory Map

The FPGA occupies 4MB of address space in the users application. This 4MB window can be accessed directly, or through the use of Slave Mode DMA. Direct access is recommended for accessing register spaces. Slave mode DMA is recommended for any accessed to the memory regions.

The basic memory map consists of 3 areas:

System Control Registers : controlling the memory paging and interrupts

Application Registers : for use by the PUPE modules

2 MB Memory window : for accessing SDRAM or Block RAM banks

System Memory Map:

Address	Name	R/W	Description
0x000000	IMEM_REG	RW	Block RAM Index
0x000008	LOCKED	RO	DCM Locked Status
0x000010	ADDR_REG	RW	PAGE Register for SDRAM access

0x000018	MEM_REG	RW	SDRAM Bank Select Register
0x000020	IER	RW	Interrupt Enable Register
0x000028	ISR	RW	Interrupt Status Register
0x000400 -0x000418	MEM_GNTx	WO	Memory Grant Registers
0x000800 -0x000TBD	Application Registers	RW	See Application Memory map
0x200000 -0x3FFFFFFF	Memory Window	RW	2MB window for accessing SDRAM or Block RAM

System Registers

0x000000 IMEM_REG

This register is used to select the bank of block RAM in the application to map into the Memory Window. The application may implement many banks of block RAM and make them accessible to the PCI bus. This register selects the appropriate bank, which will be accessible if MEM_REG 0x0000018 is set to 0

0x000008 LOCKED

This register indicates if the internal FPGA DCMs for clocking the SDRAMs have achieved lock status.

0x000010 ADDR_REG

This register is used to select a 2MB page in the SDRAM memory bank. The significant bits are from bits 19 to 24.

0x000018 MEM_REG

This register is used to lock an SDRAM bank for PCI access. Setting bit 3 switches the memory window from internal to external memory. Bits 0 to 2 select the bank (0,1,2 or 3). Read only bits 8 to 11 indicate if the internal FPGA PU process has been granted access to the memory, and bits 16 to 19 indicate if the internal PU process is requesting access to these memories. Note any bit set between 8 and 11 indicates that PCI access of the bank is prohibited.

0x000020 IER

The interrupt enable register allows up to 32 interrupts sources to be enabled.

0x000028 ISR

The interrupt status register indicates which of 32 interrupt sources generated the interrupt. The bit is cleared by writing a 1.

0x000400 MEM_GNT0

0x000408 MEM_GNT1

0x000410 MEM_GNT2

0x000418 MEM_GNT3

These 4 write only registers control the Request/Grant lines for the SDRAM bank dual port access. By default they are set to auto-grant mode (i.e. if the PU side code requests the memory, it always gets it). Write 0 to disable access to a bank (e.g.

before performing a local bus read), write 1 to grant access to a bank and write 2 to enable auto-grant mode.

Application Register Banks

The application registers will be split into 4 banks, 1 for registers to control the whole FPGA, and 3 identical banks of registers for each PU.

FPGA Registers

Address	Name	R/W	Description
0x000800	FIRMWARE	RO	Firmware ID code "CN" + version numbers
0x0008xx	ADC	RW	ADC Control Register
0x0008xx	TIMING_IO	RW	Timing I/O Register
0x0008xx	TESTCTRL	RW	Test Data Control Register
0x0008xx	TESTLEN	RW	Test Data Pattern Length

0x000800 FIRMWARE

This register contains the firmware identification tag and firmware version number.

0x0008xx ADC

This register controls the 9 channel ADC interface module.

0x0008xx TIMING_IO

This register controls the input and output options of the timing IO, allowing the FPGA to accept the main timing input signals, and echo them onto the local timing bus.

0x0008xx TESTCTRL

This register enables the operation of the test pattern buffer, allowing SDRAM bank 3 to be used as a source of data in place of the ADC inputs.

0x0008xx TESTLEN

This register indicates the size of the test data set in bank 3, and indicates when it needs to repeat its cycle.

PU Registers (one bank per PU)

Address	Name	R/W	Description
0x000XXX	CONTROL	RW	PU General Control and Status register

0x000XXX	CYCLE	RW	Cycle number
0x000XXX	TIME	RO	Time in ms from start of cycle
0x000XXX	TIME_TBLADDR	RO	Last write address in timing table
0x000XXX	PLL_FREQUENCY	RW	PLL Reference orbit frequency
0x000XXX	PLL_FREQDELAY	RW	PLL frequency load delay
0x000XXX	PLL_PHASEDELAY	RW	PLL phase delay
0x000XXX	PLL_GAIN	RW	PLL gain
0x000XXX	DDS_FREQ_MIN	RW	PLL DDS minimum frequency
0x000XXX	DDS_FREQ_MAX	RW	PLL DDS maximum frequency
0x000XXX	DIAG_CTRL	RW	Diagnostics Control/Status
0x000XXX	DIAG_TRIGGER	RW	Diagnostics Trigger
0x000XXX	DPE_SADDR	RW	DPE Start Read Address
0x000XXX	DPE_PATTERN	RW	DPE Skip/Length
0x000XXX	DPE_DADDR	RW	Target Address
0x000XXX	TEST	RW	Timing Test

0x000xxx CONTROL

This register provides general purpose control bits (e.g. reset and enable bits) and status bits (e.g. switch state information)

0x000xxx CYCLE

This register indicates the current cycle number. This is incremented every time a Start Cycle Timing Event occurs.

0x000xxx TIME

This register indicates the time in ms from the last Start Cycle event.

0x000xxx TIMETBL_ADDR

This register indicates the last address written to in the Cycle Timing Table.

0x000xxx PLL_FREQUENCY

Initial value of orbit frequency in Hz (B->f)

0x000xxx PLL_FREQDELAY

Delay (in ms) after CYCLE_START before setting PLL_FREQUENCY

0x000xxx PLL_PHASEDELAY

Phase offset in clock cycles to compensate for PU's position in the ring

0x000xxx PLL_GAIN

Gain of PLL

0x000xxx DDS_FREQ_MIN

Minimum frequency parameter for DDS

0x000xxx DDS_FREQ_MAX

Maximum frequency parameter for DDS

0x000xxx DIAG_CTRL

Diagnostics control and status register. This register contains control bits for arming the trigger, selecting the sample rate, and selecting a start time.

Bits	31:16	15:8	2	1	0
Function	Start Time	Sample Clock	AND/OR	Capturing	Armed

Bit 0 is set to Arm the diagnostics data capture trigger. It is reset to 0 on the trigger condition being fulfilled.

Bit 1 indicates that the capture is still in progress.

Bit 2 is used to select the logical combination of trigger signals (0 => AND, 1=>OR)

Bits 15:8 select a sample clock to capture the data with:

Bits 15:8	Sample Clock	Frequency
0	ADC CLK	125 MHz
1	ADC CLK /2	62.5MHz
2	ADC CLK /5	25MHz
3	ADC CLK /10	12.5MHz
4	ADC CLK /20	6.25MHz
5	ADC CLK /50	2.5MHz
6	ADC CLK /100	1.25MHz
7	ADC CLK /200	625kHz
8	ADC CLK /500	250kHz
9	ADC CLK /1000	125kHz
10	ADC CLK /2000	62.5kHz
11	ADC CLK /5000	25kHz
12	ADC CLK /10000	12.5kHz
13	ADC CLK /20000	6.25kHz
14	ADC CLK /50000	2.5kHz
15	ADC CLK /100000	1.25kHz
16	SYSTEM CLOCK	10MHz
17	C-CLOCK	1kHz
18	F_REF	200-500kHz
19	F_REF_DELAYED	200-500kHz

Start time indicates how many ms after CYCLE_START before the trigger armed (the value in C-Timer).

0x000xxx DIAG_TRIGGER

Diagnostics trigger register. This register contains trigger bits indicating when to start recoding the diagnostic information. The bits in this register correspond to the timing signals and internal signals to be captured in the diagnostics timing signals table. The register indicates which bits to watch. The trigger only detects signals going to logic '1'. Setting bits in the register to '0' indicates the bits are to be ignored. If bit 2 of DIAG_CTRL is set, then the data capture begins when any of the indicated signals go to '1', otherwise the trigger waits till all set signals go to '1'. If the register is set to all zeros, the trigger data capture will occur as soon as C-Timer reaches the start time.

0x000xxx DPE_SADDR

Data Pattern Engine Start address. Indicates the start address in the SDRAM bank to begin the Scatter Gather operation. Writing to this triggers the DPE burst.

0x000xxx DPE_PATTERN

Data Pattern Engine Pattern. Indicates how many words to skip, and how many words in total to gather in the DPE burst.

0x000xxx DPE_DADDR

Data Pattern Engine destination address. Indicates the start address in SDRAM bank 3 to begin storing the DPE burst.

0x000xxx TEST

Test Register. Used to select test data patterns for the particular PU. Can also be used to write timing signals to the individual PU.

SDRAM Bank Allocation

Bank	Function
0	PU #0 Cycle Data Table
1	PU #1 Cycle Data Table
2	PU #2 Cycle Data Table
3	Test Pattern Buffer / DPE Target Buffer

Cycle data are stored in 64 bit blocks containing 3x16 bit values in banks 0,1 and 2. The format is:

Bits	63:48	47:32	31:16	15:0
Function	Not used	Delta Y	Delta X	Sigma

When the DPE transfers data to bank 3, it is kept in this format.

Test pattern data are stored in a more compact form, to improve memory access efficiency and to guarantee 1 sample per clock cycle operation. The format uses 16

bit integer for Sigma values, but only 8 bits for Delta X and Delta Y. Delta X and Delta Y are shifted by TBD bits to form 16 bit simulated ADC input values.

The 64 bit format contains 2 samples:

Bits	63:56	55:48	47:32	31:24	23:16	15:0
Function	$\Delta Y1$	$\Delta X1$	$\Sigma 1$	$\Delta X0$	$\Delta Y0$	$\Sigma 0$

Block RAM Bank Allocation

Bank	Size (min 2kB)	PU	Bit Width	Function
0	32kB	0	64	Cycle Timing Table
1	2kB	0	64	Cycle Information Table
2	2kB	0	8	Timing Phase Table
3	2kB	0	32	Timing Switch Table
4	8kB	0	64	Diagnostics Sigma, DeltaX, DeltaY table
5	4kB	0	32	Diagnostics Timing Signals Table
6	32kB	0	64	Bunch Mean Table #0
7	32kB	0	64	Bunch Mean Table #1

Block RAM bank allocation similar for PU #1 and PU#2, using bank ID's 8-15 and 16-23 respectively. All block RAMs are access as 64 bit wide memories on the PCI bus. Smaller word widths occupying multiple addresses are mapped into single 64 bit wide memory locations. This makes it possible to create the Timing Phase table as an array of char and DMA it directly to the Block RAM.

The cycle timing table stores data in 64 bit words containing a 32 bit cycle number and a 32 bit address pointing into the cycle data table.

Bits	63:32	31:0
Function	Address	Cycle

The cycle information table, records all timing events, and the address pointer for the cycle timing table.

Bits	31:24	23:0
Function	Event	Address

4 cycles are stored in this table. Space is provided for 16 events per cycle. The 2 least significant bits of the cycle number are used as the most significant bits of the address, with the event counter incrementing from 0 to 15 on events, and reset to 0 on the CYCLE_START event.

The timing phase table consists of 16 tables with 512x8 bit entries. Within each 8 bit value, the following bits are defined:

- 0 – LO
- 1 – BLR
- 2 – GATE

- 6 – Mean Filter Clock Enable 0
- 7 – Mean Filter Clock Enable 1

The timing switch table format has been described in the PLL-SPU section. The Diagnostics Sigma, Delta X, Delta Y table is stored in the same format as the cycle data:

Bits	63:48	47:32	31:16	15:0
Function	Not used	Delta Y	Delta X	Sigma

The Diagnostics signal table is 32 bits wide and records the signals:

Timing inputs:

CYCLE_START, CYCLE_STOP, CAL_START, CAL_STOP, INJECTION, HCHANGE, FREF, SYSTEM_CLOCK

Internal signals:

LO, BLR, GATE, F_REV, F_REF, ACQ, RF, ϵ , θ

The bunch mean tables are updated at the same time as the Cycle Timing Table, with the output of the bunch mean filters. The data is formatted as 3x16 bit values representing Sigma, Delta X and Delta Y.

Interrupts

The interrupt register has space for 32 interrupt lines. The following bits are used:

Bit	Function
0	PU #0 CYCLE_START
1	PU #0 CYCLE_STOP
2	PU #0 ERROR
3	PU #0 DPE COMPLETED
4	PU #0 DIAGNOSTIC INFO CAPTURED
5-7	PU #0 User Interrupts (Switch Table bits 5-7)
8	PU #1 CYCLE_START
9	PU #1 CYCLE_STOP
10	PU #1 ERROR
11	PU #1 DPE COMPLETED
12	PU #1 DIAGNOSTIC INFO CAPTURED
13-15	PU #1 User Interrupts (Switch Table bits 5-7)
16	PU #2 CYCLE_START
17	PU #2 CYCLE_STOP
18	PU #2 ERROR
19	PU #2 DPE COMPLETED
20	PU #2 DIAGNOSTIC INFO CAPTURED
21-23	PU #2 User Interrupts (Switch Table bits 5-7)

References

- [1] IT-3384/AB: Technical Specification for a new trajectory measurement system for the CERN Proton Synchrotron
- [2] Alpha Data's TMS Tender "pre-design-1.5".
- [3] Alpha Data/Beam PUPE-API document 0.6 (Terry Barnaby)
- [4] pll3.vhd (reference PLL VHDL design, Greg Kasprowicz CERN)