FEMINOS CARD

USER'S MANUAL

D. Calvet

Version 2.12

1 General Overview

The Feminos card is the digital part of a modular and flexible general purpose readout system for small to medium size gaseous detectors. On one side, the Feminos card connects directly to a 288-channel Front-End Card (FEC) populated with four AFTER chips [1] or a 256-channel FEC equipped with four AGET chips [2]. On the other side, the Feminos card communicates with a remote DAQ PC via a standard Gigabit Ethernet link and optionally receives global clocking and triggering information from a central Trigger and Clock Module (TCM). With the appropriate Ethernet switch and a TCM, the Feminos system can be scaled up to 24 Feminos and FECs leading to a total channel count of 6912 using AFTER chips or 6144 using AGET chips. In principle, multiple TCM can be cascaded to build larger system, but this configuration is beyond the goal of this development. A typical example of configuration is shown in Fig. 1.

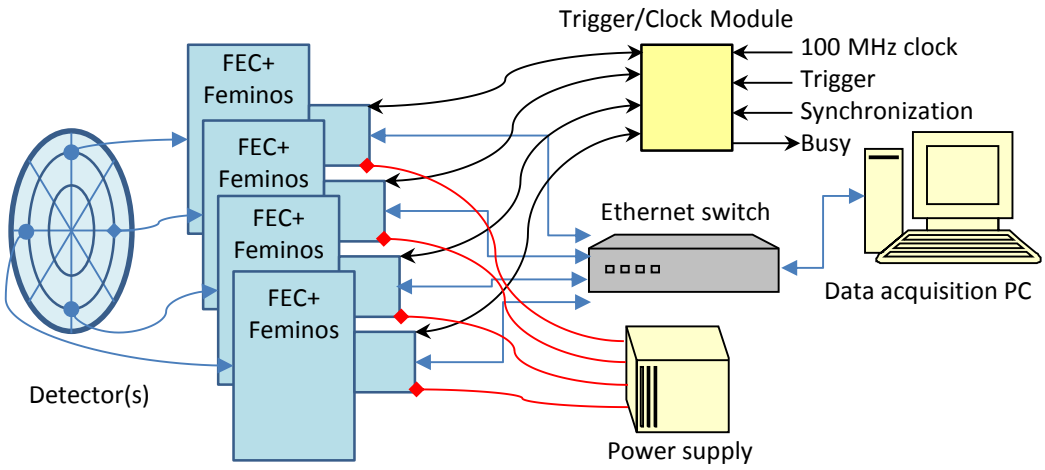


Fig. 1. Feminos application example.

Each Feminos card receives three cable connections: power supply, clock/trigger (RJ45 connectors) and configuration/readout (Ethernet).

2 BOARD DESCRIPTION

A functional block diagram of the Feminos card is shown in Fig. 2. The central elements of the Feminos card are a MicroBlaze processor and control logic embedded in a single Xilinx Spartan 6 FPGA. The control logic is interfaced to the FEC, a 200 MHz ZBT SRAM which is used for temporary data strorage, a clock and trigger interface, LEDs, push buttons, switches and debugging pins. The embedded processor is connected to a 128 MB SDRAM used for storing the application program and data buffers to be sent to the remote DAQ PC via the Gigabit Ethernet interface. A RS-232 port for console debugging is also provided. A flash memory, programmable in-situ via JTAG, stores the FPGA bitstream and the application program. External power is provided via a connector that optionally provides an input to inhibit power locally.

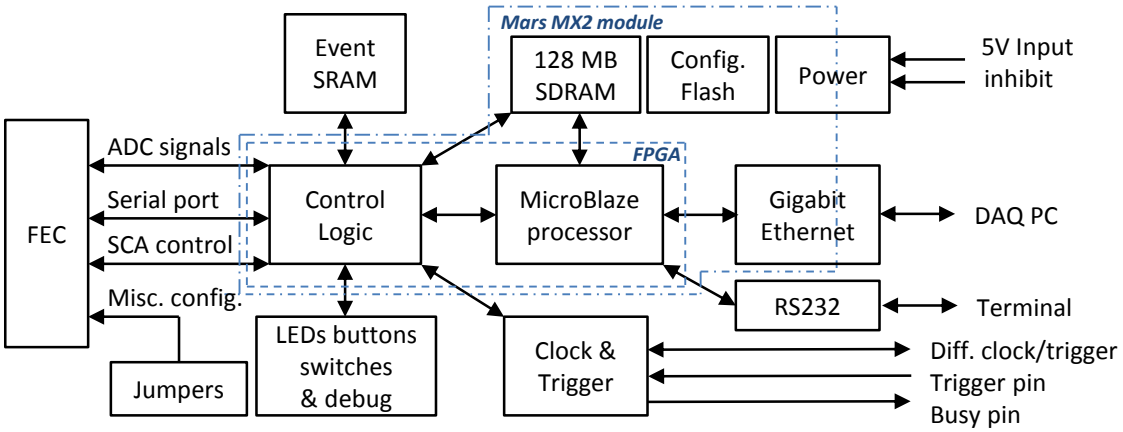


Fig. 2. Functional diagram of the Feminos card.

A picture of the Feminos is shown in Fig. 3. The central component is a Mars MX2 FPGA module from Enclustra [3]. This compact and inexpensive product features in a standard SO-DIMM form-factor (30 mm x 68 mm) a Xilinx Spartan 6 FPGA (XC6LX45T), 128 MB of DDR2-800 SDRAM, 16 MB of Flash memory, a Gigabit Ethernet PHY, 96 user I/O pins, 2 multi-gigabit transceivers, 4 LEDs, DC/DC converters, and more. The Feminos is comparatively simple and mostly includes a 9-Mbit 200 MHz ZBT SRAM, glue logic, I/O connectors and additional voltage regulators or a DC/DC converter.

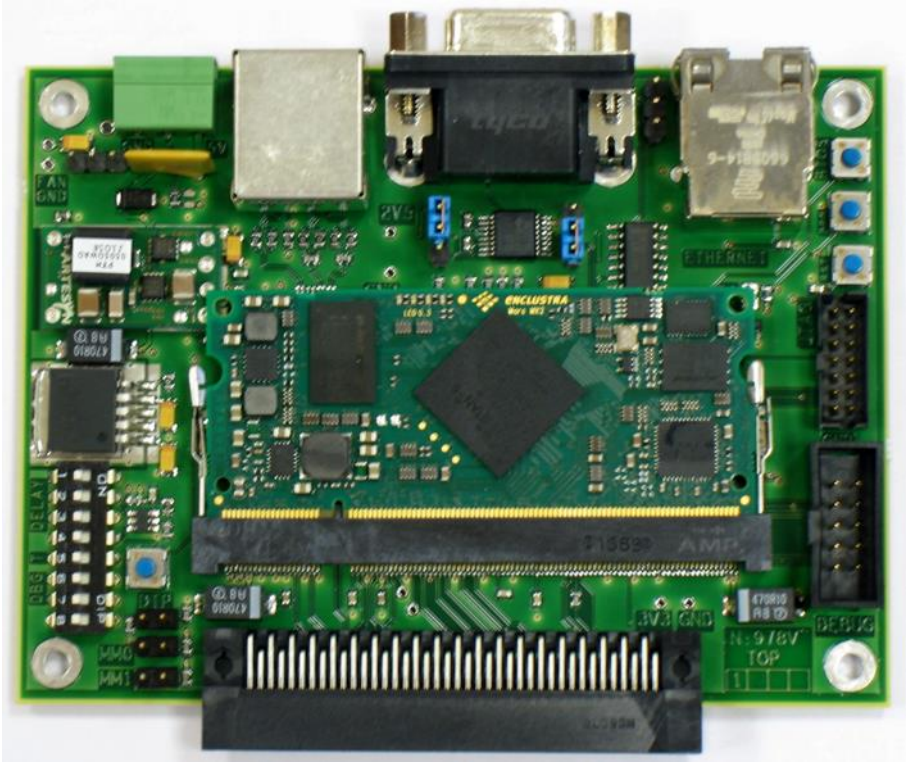


Fig. 3. Picture of the Feminos card.

The Feminos card can be equipped with an optional shield and heat sink. When it is not present, a heat sink and fan is required for the Mars MX2 module.

3 HARDWARE INSTALLATION

A schematic view of the Feminos card is shown in Fig. 4.

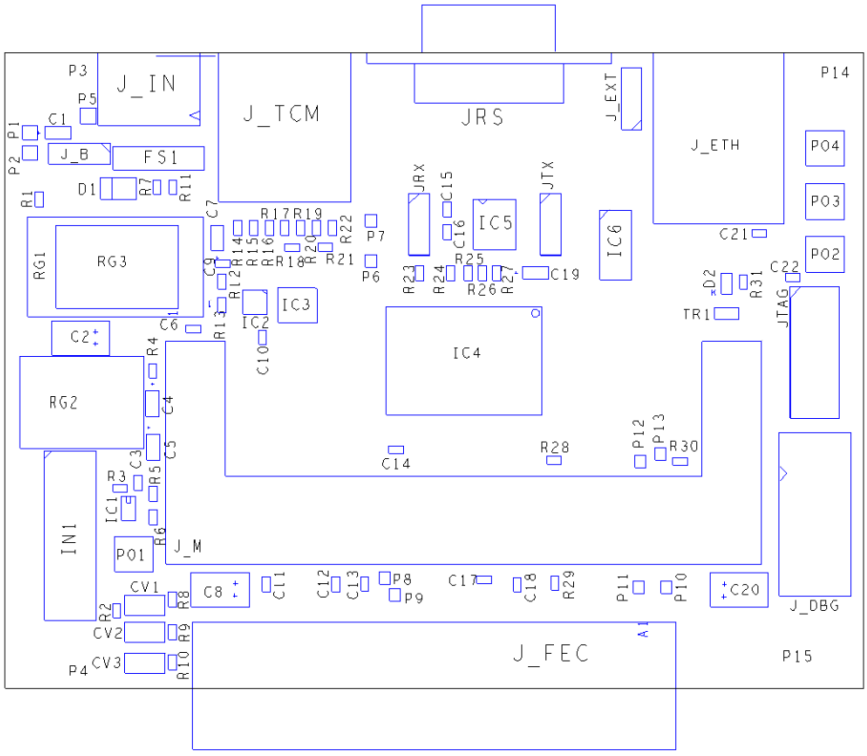


Fig. 4. Feminos card components.

The connectors are as follows:

- J_FEC is an 80-pin connector that mates to the FEC.
- J_M is the 200-pin SO-DIMM connector where the Mars MX-2 module is installed.
- J_IN is a 3-pin or 4-pin (see details on the power option below) power supply connector.
- J_B is a 3-pin connector used to select the reference voltage for the LDO of the main power supply.
- P1 (GND) and P2 (5V) are a fused power supply for an optional external fan.
- J_TCM is a female RJ-45 connector to interface to a TCM.
- J_RS is a 9-pin DB-9 female connector to connect a RS232 console.
- J_EXT is a 3-pin 2.54 mm pitch header that provides an external trigger input (pin closest to the front panel), a busy pin (middle position) and ground (rear position). The EXT_TRIG pin is 5V TTL compatible input with a 50 ohm termination resistor connected to ground. The BUSY pin is a 3.3V LVTTL output without series resistor.

- J_ETH is a standard 10/100/100 Ethernet RJ45 header to be connected to a switch or to the DAQ PC directly.
- JTAG is 14-pin 2 mm pitch male header for connecting a Xilinx USB Platform cable.
- J_DBG is a dual row 10-pin 2.54 mm pitch male header that provides 8 general purpose output pins for debugging as shown in Fig. 5.

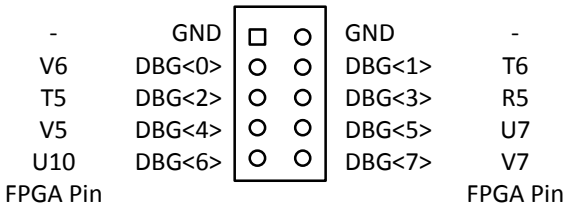


Fig. 5. Debugging connector.

The Feminos card has several power supply options. Some cards are equipped with a DC-DC converter. This option provides minimal power dissipation at the expense of reduced noise immunity. These cards require a 3-wire power supply cable as shown in Fig. 6.a. Some cards are equipped with a LDO. This option suppresses the noise inherent to a DC/DC converter at the expense of higher power dissipation. This type of card also requires a 3-wire power supply cable as shown in Fig. 6.b. A further option is available with the LDO version. Using an additional reference voltage of 5V, the main input voltage can be reduced. This reduces power dissipation but requires a 4-wire power supply cable and a dual-output power supply as shown in Fig. 6.c.

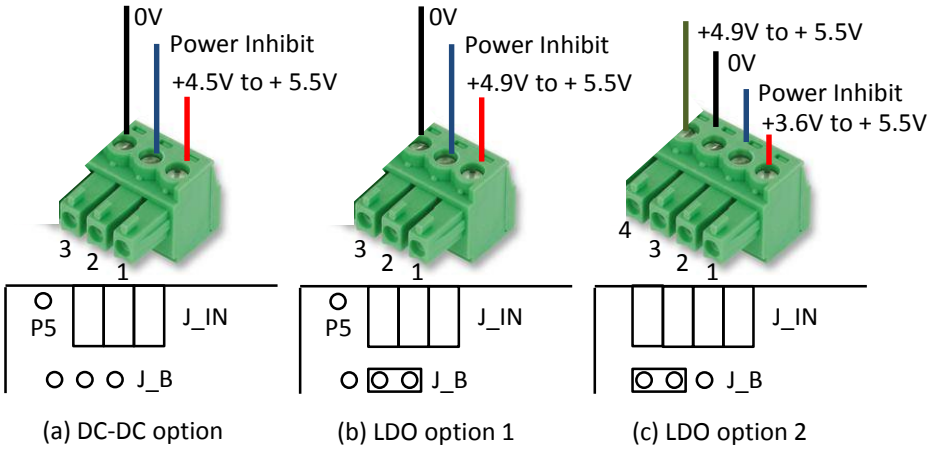


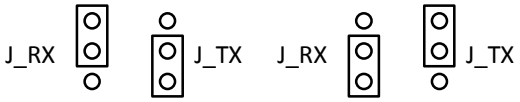
Fig. 6. Feminos card power supply options.

The power inhibit wire is optional and disables on-board power when pulled low. When it is left floating or set to a high level, power is enabled on the Feminos card. The

voltage applied to the Power Inhibit pin may not exceed the main supply voltage of the card.

Several features are controlled by jumper settings on a 2-pin or 3-pin male header. Jumper functions are as follows:

- CV1: when installed the ADC of the FEC is placed in test mode, i.e. a constant data pattern "010101010101" is generated on all data outputs. This jumper is not installed for normal operation.
- CV2, CV3: when installed, the corresponding PhotoMos relay of the FEC is opened, i.e. the polarization resistors of the detector pads are left floating. Each PhotoMos relay controls half of the pads of a FEC. These jumpers are not installed for normal operation.
- JRX, and JTX: allow the swap of the transmitted and received signals on the RS-232 port to be able to use a cross-cable or straight cable. One jumper must be installed on each connector for proper operation as shown in Fig. 7.



(a) Straight cable (a) Cross-cable

Fig. 7. RS-232 jumper settings.

Eight DIP switches are available. Their function is as follows:

- Switch 1 to 1 to 4: determines the power-up soft start delay of the Feminos card. In a multi-Feminos setup where no individual power inhibition control line is provided, these switches should be set to different values (16 combinations) to avoid a high in-rush current if a common power supply is used.
- Switch 5: when OFF, the Feminos uses its own 100 MHz local clock as a reference. When ON, the Feminos expects to receive the 100 MHz reference clock from the TCM connector.
- Switch 6: unused
- Switch 7 and 8: these switches determine which internal signals are routed to the 8 debugging pins of J_DBG. These can be safely left OFF by most users.

The Feminos card has four push buttons. Their function is as follows:

- P01: disables the main DC-DC converter or LDO of the Feminos card when pressed.

- P02: resets all user logic and the MicroBlaze processor when pressed.
- P03: re-programs FPGA (reloads firmware and software).
- P04: forces the MicroBlaze to execute the "mini-bios" utility when pressed before the FPGA is configured.

4 SOFTWARE INSTALLATION

4.1 EMBEDDED FIRMWARE AND SOFTWARE

The FPGA bitstream and MicroBlaze executable for the Feminos card are stored in the SPI Flash of the Mars MX2 module installed on the card. At power-up, the FPGA automatically reads its bitstream configuration file from the SPI Flash (address 0x0 in the SPI Flash) and the MicroBlaze processor starts the boot loader program stored in the bitstream. The boot loader reads the section of the SPI Flash where the embedded software is stored (address 0x200000 in the SPI Flash), copies this executable program into the external SDRAM and runs it. During this phase, status and eventually error messages are displayed on the RS232 console. Terminal settings should be: 9600 baud, 1 start bit, 1 stop bit, no parity, no hardware flow control, enable local echo, map CR to CR+LF on the input side.

For advanced users, here are the basic steps to (re)build the required binary files and program the SPI Flash.

- The FPGA configuration bitstream is built by ISE. It is typically found in: /xilinx/minos/feminos/feminos_fpga.bit
  This file (and other required files) are exported to SDK by the user from ISE using the menu "Export Hardware Design To SDK with Bitstream".
- The bootloader is generated in .elf format by SDK and it has to be merged with the FPGA bitstream. A simple method to produce the required file is to program the FPGA (via JTAG) from the SDK. The bootloop code file needs to be replaced by the bootloader code at download. The file that contains the FPGA firmware and bootloader is typically found in: /xilinx/minos/feminos_system/workspace/hw_platform_0/download.bit
- The main application of the Feminos card is built in .elf format by the SDK. It must be converted to SREC format to be readable by the bootloader. The conversion is made from an EDK shell by the command: mb-objcopy –O srec axidataserver.elf axidataserver.srec
- The SPI Flash file is prepared with iMPACT in .mcs format. This file contains the FPGA configuration bitstream with the embedded bootloader code at address 0x0 and the main application software in SREC format at address 0x200000. The complete file for the SPI Flash is typically found in: /xilinx/linos/feminos/feminos_flash.mcs
  This file is programmed in the SPI Flash through the FPGA via JTAG using iMPACT.

4.2 Setting Non-volatile Parameters

The last page of the last sector of the SPI Flash of the Mars MX2 module is used to store the non-volatile parameters used by the Feminos card. These parameters must be programmed at least once, and may be altered via the "mini-bios" embedded utility. To enter, mini-bios, hold the BIOS_B push button (P01) during card power-up or FPGA re-configuration. After the boot loader executes, the menu shown in is printed on the RS-232 console.

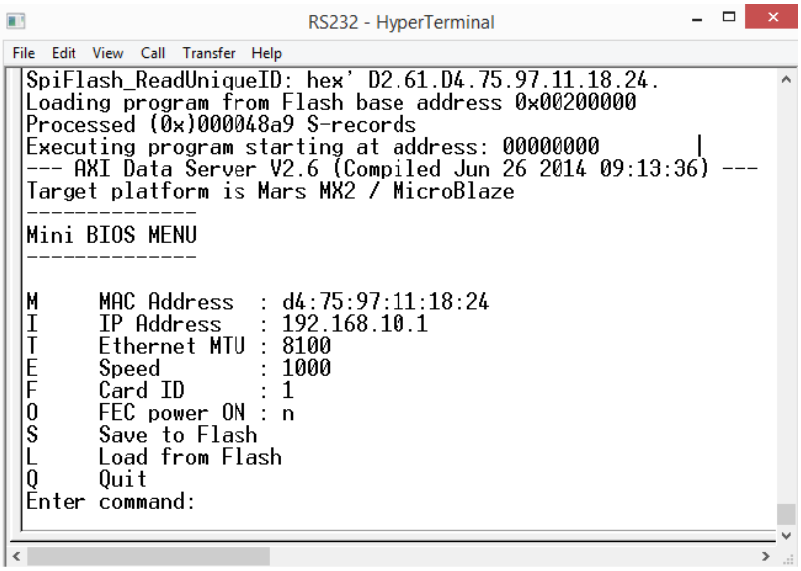


Fig. 8. Mini-Bios terminal window.

The user can set:

- the Ethernet MAC address,
- the IP address,
- the size of the maximum transfer unit (MTU) – recommended values: 1500 in Ethernet 10-100; 8100 in Gigabit Ethernet with Jumbo frames enabled,
- the Ethernet speed (10, 100 or 1000 Mbps),
- the ID of this Feminos card (from 0 to 31),
- the power state of the FEC after boot.

Parameters can be saved to flash and reloaded. After quitting mini-bios the normal command server program will start.

4.3 Test Client DAQ

In order to test the Feminos card and exercise a minimal system, a console-based client program, called "mclient", is provided. This program runs on the DAQ PC (Window or Linux) and makes the interface to the Feminos card via Ethernet. The user enters from the keyboard the commands that are sent to the Feminos card and results are displayed in ASCII format in the terminal window of the mclient program. The

mclient program can also execute command scripts stored in files and store results in ASCII or binary files.

5 EMBEDDED PROCESSOR TO FIRMWARE INTERFACE

The processor embedded in the FPGA logic of the Feminos card communicates with the local peripheral logic through the following means:

- a set of ten 32-bit wide registers,
- five dual-port memory blocks,
- two embedded FIFO's and a control register

The registers and memory locations are normally not accessed directly by the user and the following information is only relevant to the developers of the embedded software of the Feminos card.

5.1 INTERNAL CONFIGURATION REGISTERS

The Feminos internal register map is shown in Table 1. The base address is mapped into the 32-bit address space of the MicroBlaze processor. All addresses are aligned on 32-bit boundaries and little-endian byte ordering is assumed.

Table 1 . Internal registers of the Feminos card.

Address	Register	Access	Function
Base+0x00	#0	R/W	General configuration
Base+0x04	#1	R/W	Front-end ASIC control
Base+0x08	#2	R/W	Trigger configuration
Base+0x0C	#3	R/W	Pulser configuration
Base+0x10	#4	R/W	Multiplicity thresholds
Base+0x14	#5	R/W	Configuration Register
Base+0x18	#6	R/W	Trigger Clock Module interface
Base+0x1C	#7	RO	Free running clock cycle counter
Base+0x20	#8	R/W	Multiplicity Limit
Base+0x24	#9	R/W	Reserved for future use

General Configuration (Register #0):

This register is shown in Fig. 9.

The SCA_CNT field determines the number of SCA cells to readout. This parameter can be set from 1 to 511 and 1 to 512 in the AFTER mode and AGET mode respectively.

The MODE_AFTER bit determines if the Feminos card is being used to readout AFTER chips (MODE_AFTER=1) or AGET chips. This bit is 0 by default.

The RST_CHAN_CNT bit is used in the AGET mode to define the number of reset cycles that appear at the output of the device when switching from one column to the next. The AFTER chip has a fixed number of reset cycles which is equal to 3. The AGET

chip can be programmed to have 2 or 4 reset cycles. In addition to programming the AGET chips with the desired value, the configuration register of the Feminos card must be set accordingly: RST_CHAN_CNT=0 (default value) corresponds to 2 reset cycles.

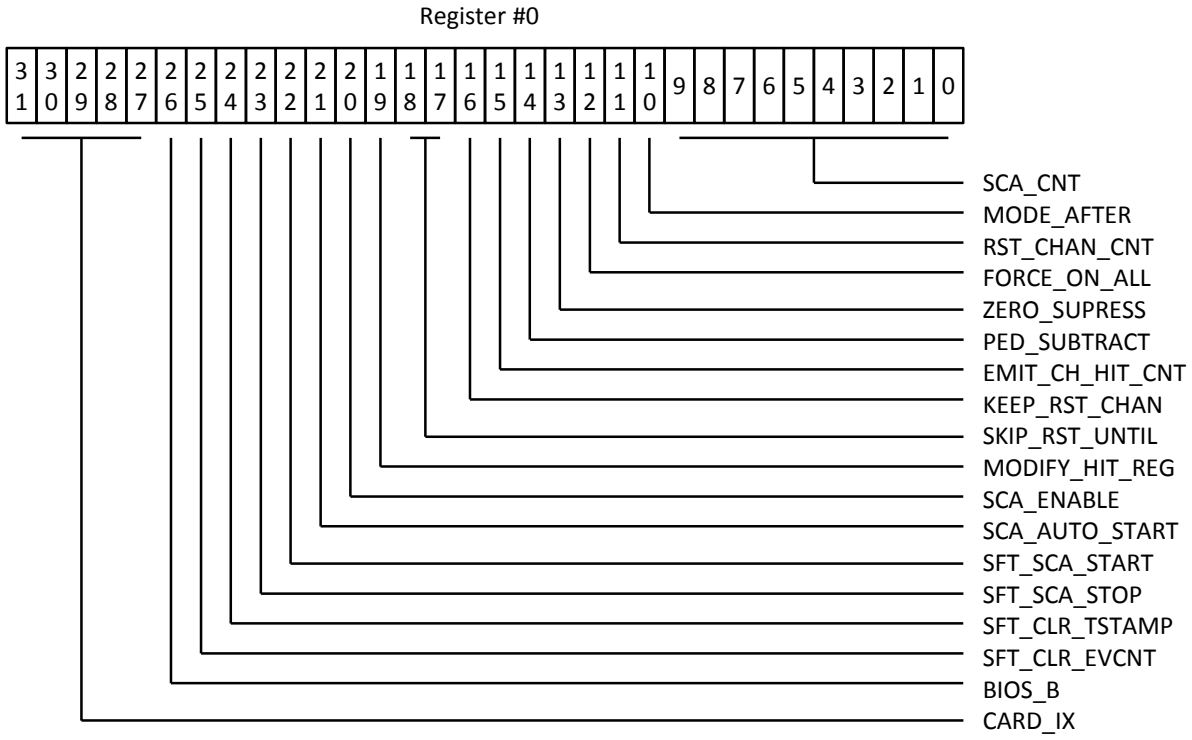


Fig. 9. General Configuration Register (Register#0).

When set, the FORCE_ON_ALL bit is used in the AGET mode to inform the local logic that all 64 physical channels and 4 FPN channels of the AGET chips are being readout. The AGET chips must be programmed accordingly. When this option is selected, the logic of the Feminos card expects to receive 70 or 72 ADC samples per time-bin (corresponding to 2 or 4 reset values + 64 channels + 4 FPN). The readout of the channel hit register is skipped when this option is active. In the AFTER mode, this option need not be specified because the AFTER chip does not support sparse channel readout and, in this mode, the Feminos logic always expects 79 samples per time-bin (3 reset phases + 72 physical channels + 4 FPN).

The ZERO_SUPPRESS bit is used to optionally apply a zero-suppression algorithm on channel data. When this bit is not set, all data are kept. See zero-suppression algorithm details for more information. This option is valid in both AGET and AFTER modes.

The PED_SUBTRACT bit is used to optionally subtract a pre-loaded constant to each channel. See pedestal subtraction section for details. This option is valid in both AGET and AFTER modes.

The EMIT_CH_HIT_CNT is used to optionally add in the data stream sent to the remote DAQ computer the number of channels that were hit in each of the ASICs controlled by the Feminos card. This information is useful for debugging. In the AFTER mode, the number of channel hit will always be 79. In the AGET mode, the value returned will vary from 6 to 72 depending on the number of reset channel count and hit register count (if it is not by-passed by other settings). Note that the number of reset phases and FPN channels count are always added to compute the total number of channel hit in a chip.

The KEEP_RST_CHAN bit can be optionally set to keep in the data stream sent to the remote DAQ computer the data corresponding to the reset channels of the AFTER or AGET chips. Keeping the data of these non-physical channels can be used for some specific debugging tasks.

The SKIP_RST_UNTIL field is used when KEEP_RST_CHAN is disabled to determine how many reset channels should be discarded before data are sent to the remote DAQ computer. Most users will not want to keep any of the reset channel data. In the AGET mode, the corresponding value for this field is "01" when the number of reset phases is programmed to 2 and "11" when the number of reset phases is 4. In the AFTER mode, this field should be set to "10" to skip transmitting the data of 3 reset channels.

The MODIFY_HIT_REG bit is used to enable/disable modification of the hit register content before SCA digitization (only available in the AGET mode). When the MODIFY_HIT_REG bit and FORCE_ON_ALL bit are cleared, the Feminos logic reads the content of the Hit Channel Register of the active AGET chips it controls to determine the subset of channels that are readout for the current event. When MODIFY_HIT_REG is set to 1 while FORCE_ON_ALL is cleared, the Feminos logic reads the content of the Hit Channel Register of the active AGET chips, preserves forces to 1 or forces to 0 each individual bit, and writes to the AGET chips the altered value. After all Channel Hit Registers have been updated, SCA digitization is initiated. The per-channel, PRESERVE, FORCE_ON and FORCE_OFF settings are programmed during system configuration. See the relevant section for details.

The SCA_ENABLE bit is used to enable/disable the operation of the ASICs controlled by the Feminos card. This bit must be set to 1 for normal operation, after system configuration.

The SCA_AUTO_START bit is used to determine when the write operation in the SCA of the front-end ASICs can be started. When the SCA_AUTO_START bit is set to 1, the Feminos card will start the SCA write operation for the next event as soon as possible after the digitization of the current event. When this bit is cleared, the Feminos will not restart the write operation in front-end SCAs until this order has been received from

the TCM. In a single, standalone Feminos setup, SCA_AUTO_START should be set to 1. It leads to the shortest possible dead-time. In a setup with several Feminos cards controlled by a TCM, this bit should be set to 0 for operation in a common dead-time mode, but it can also bit set to 1 if each Feminos card is allowed to handle its own dead-time independently.

The SFT_SCA_START bit is used to asynchronously start writing in front end SCAs. The operation takes place on the transition from 0 to 1 of this bit. The user need not act on this setting when SCA_AUTO_START is active or when the SCA start order is received from the TCM interface. This option is mainly used when the pulse generator of the FEC is being used.

The SFT_SCA_STOP bit is used to generate software triggers. It should be set to 1 and then returned to 0. The SCA_START order must have been received (via the TCM or the SCA_AUTO_START flag) prior to SFT_SCA_STOP to effectively trigger SCA digitization.

The SFT_CLR_TSAMP bit is used to asynchronously clear the event time stamp counter. The clear is performed when this bit is set from 0 to 1. The SFT_CLR_TSAMP bit must be cleared to 0 before this function can be re-used.

The SFT_CLR_EVCNT bit is used to asynchronously clear the event counter. The clear is performed when this bit is set from 0 to 1. The SFT_CLR_EVCNT bit must be cleared to 0 before this function can be re-used.

Front-End ASIC Control (Register #1):

This register is shown in Fig. 10.

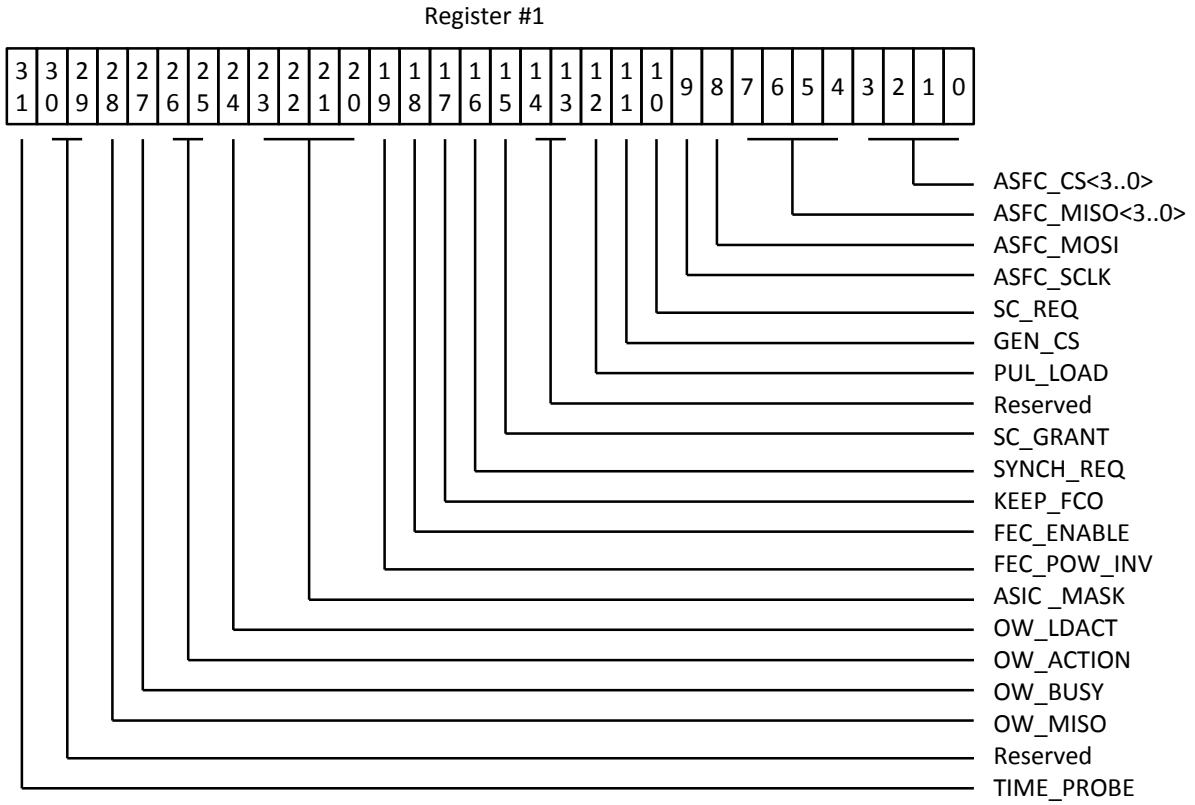


Fig. 10. Front-end ASIC control register (Register #1).

The ASFC_CS<3..0> bits are directly connected electrically to the slow control Sc_en line of each individual ASIC controlled by the Feminos card. Similarly, the ASFC_SCLK bit maps directly to the serial clock line, Sc_ck of the 4 ASICs. The ASFC_MOSI bit maps to the Sc_din line and is common to all 4 ASICs. The ASFC_MISO<3..0> bits maps to the Sc_dout line of each individual ASIC. Embedded software running on the Feminos card is responsible for controlling these lines in order to implement the serial protocol to read and write to ASIC configuration registers. For performance reasons, read/write operations from/to the Hit Channel Registers are implemented in firmware. Note that the individual ASFC_MISO lines allow reading out the 4 Channel Hit Registers in parallel, but only one Hit Channel Register can be written at a time because the ASFC_MOSI line is shared among 4 ASICs.

The SC_REQ bit is used to determine which of the embedded software or firmware can access the slow control lines of the ASICs. When the embedded software wishes to perform a slow control operation on ASIC registers, it sets the SC_REQ bit to 1. Software should then poll the SC_GRANT bit and wait until access to the slow control lines is granted by the firmware. The SC_REQ should be held high until the end of the slow control operation. In the AGET mode, software is responsible for setting the slow control lines in the "slow control mode" just after access to these lines has been granted, and must configure these lines in the "channel address control mode" before

releasing them. After system power-up, the embedded software should at least perform one slow control operation in each ASIC to ensure that the slow control lines are set in the "channel address control mode".

The GEN_CS bit is used as a chip select signal for programming the amplitude value of the on-board pulse generator of the FEC. The lines ASFC_MOSI and ASFC_SCLK are used as the serial data input and serial clock when programming this generator. For implementation reasons, the value programmed in the pulse generator cannot be read back. In order to program the DAC of the pulse generator, access to slow control lines must be obtained using the SC_REQ/SC_GRANT hand-shake protocol.

The PUL_LOAD bit is used to immediately update the DAC of the pulse generator to the pre-loaded value by pulsing signal GEN_GO. This bit is normally used to set/restore the baseline output level of the DAC of the pulse generator before the desired pulse amplitude is pre-loaded and automatically updated when pulse injection occurs.

The SYNCH_REQ bit is used to re-synchronize the ADC de-serializer logic. Only expert users should use this bit.

The KEEP_FCO flag is used to optionally replace the samples of ADC channel #3 (corresponding to ASIC #3) by the framing pattern delivered by the ADC (signal "FCO" = "111111000000"). This option is used for advanced debugging of the ADC interface.

The FEC_ENABLE bit is used to control power on the FEC attached to the Feminos card. By default, this bit is cleared and the FEC is OFF. This bit must be set to 1 to power ON the FEC. This operation must be done prior to any read/write operation in ASIC registers and data taking.

The FEC_POW_INV bit is XOR'ed with the FEC_ENABLE bit to produce the control signal of the voltage regulator of the FEC (REH_INH_B). On most cards, the voltage regulator is enabled with a high level on REG_INH_B and FEC_POW_INV should be cleared. A few FECs are equipped with a voltage regulator that is enabled by a low level. The FEC_POW_INV bit should be set to perform the required signal inversion.

The ASIC_MASK field is used to optionally disable one or several ASICs. Masked ASICs cannot be configured and readout. Reading and modifying their Channel Hit Register is skipped. Masking ASICs should be done when using partially equipped FEC or to increase readout performance when only a fraction of the available channels of a FEC are used. The user should always leave at least one un-masked ASIC for proper operation.

The OW_LDACT bit is used to initiate an action on the OneWire controller that drives the FEC_ID pin. The action to be performed is specified by the OW_ACTION field as follows: "00" performs reset and presence pulse detection; "01" reads one bit from

the OneWire device, "10" and "11" write 0 and 1 respectively to the device. After posting an action to the OneWire controller, the flag OW_BUSY is active until completion and the bit read from the OneWire device is available in OW_MISO.

The field TIME_PROBE is routed to pin FREE_65 on the Feminos. Changing the state of this field in software can be used to measure precise time intervals with an oscilloscope connected to pin FREE_65. This function shall only be used by developers.

Trigger Control (Register #2):

This register is shown in Fig. 11.

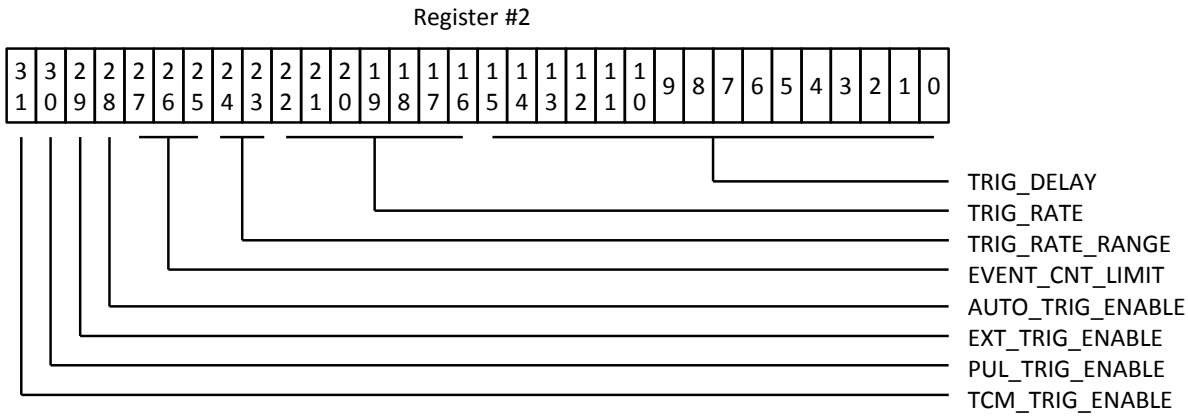


Fig. 11. Trigger Control Register (Register #2).

The TRIG_DELAY field is used to apply a fixed latency to the SCA_STOP order. The delay is expressed in 10 ns units and is programmable from 0 to 655.36 µs. Starting from firmware version 2.12, this field supports two ranges, determined by the MSB of TRIG_DELAY. When TRIG_DELAY(15) is 0, the supported range is [0; 327.67 µs] in steps of 10 ns. When TRIG_DELAY(15) is 1, the supported range is [0; 1.31068 ms] in steps of 40 ns.

The TRIG_RATE field and TRIG_RATE_RANGE field are used to specify the trigger rate when the embedded periodic trigger generator is used. Four frequency ranges are available with 100 values per range. Range "00" is from 0.1 Hz to 10 Hz in steps of 0.1 Hz; range "01" is from 10 Hz to 1 kHz in steps of 10 Hz; range "10" is from 100 Hz to 10 kHz in steps of 100 Hz; range "11" is from 1 kHz to 100 kHz in steps of 1 kHz.

The EVENT_CNT_LIMIT is used to allow that only a pre-defined number of events flow through the system after SCA_ENABLE is asserted. This feature is useful for system debugging. When EVENT_CNT_LIMIT is set to "000", there is no limitation on the number of events that are allowed to pass though the Feminos card. When EVENT_CNT_LIMIT is set to "001" or other values up to "111", the Feminos card will not respond to triggers after 1; 10; 100; 1000; 10,000; 100,000 or 1,000,000 events

have been acquired. To resume operation, the SCA_ENABLE bit must be cleared and set back to 1. Setting a pre-defined number of events is useful for system development and debugging.

The AUTO_TRIG_ENABLE bit is used to start/stop the embedded periodic trigger generator. When enabled, the generator will start after an initial delay of ~1 s. Note that the readout system may not be able to sustain the desired trigger rate and some of the periodic triggers may be lost. The periodic generator is mostly used for system test and performance evaluation.

The EXT_TRIG_ENABLE bit is used to enable or mask triggers coming from the EXT_TRIG pin located on the front panel of the Feminos card.

The PUL_TRIG_ENABLE bit is used to enable or disable the generation of a trigger when the pulse generator of the FEC is fired. When this bit is active, the AUTO_TRIG_ENABLE bit and SCA_AUTO_START bits must be cleared and the PUL_ENABLE bit must be set.

The TCM_TRIG_ENABLE bit is used to enable or mask the triggers received from the TCM via the front panel of the Feminos card.

Pulser and General Control (Register #3):

The content of this register is shown in Fig. 12.

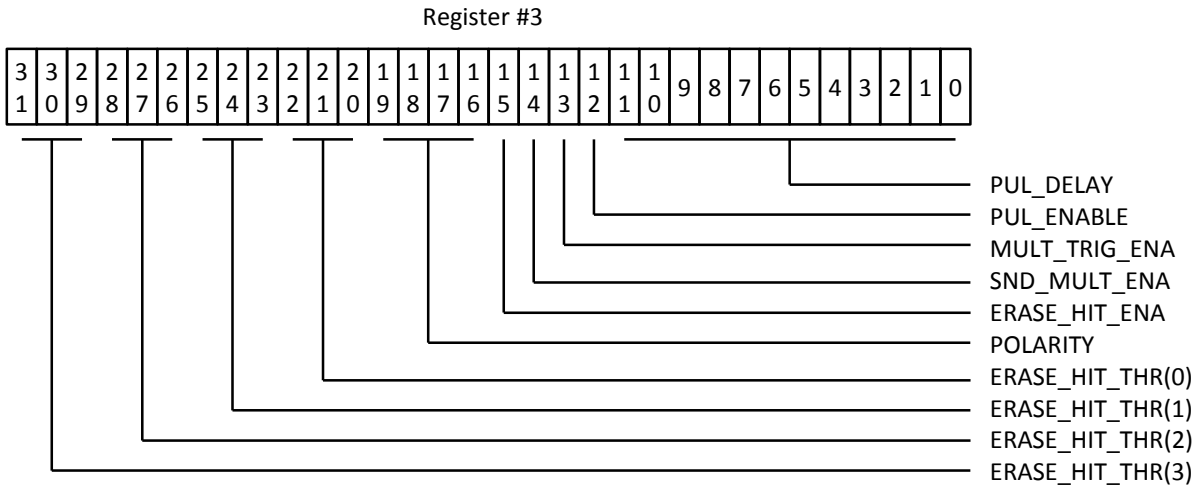


Fig. 12. Pulser and General Control (Register #3)

The field PUL_DELAY is used to specify the delay from when the SCA write signal is asserted to when the pulse generator is fired. The delay is expressed in 10 ns units. The maximum value is 40.95 µs. Note also that it takes 2 clock cycles for the internal logic to elaborate GEN_GO. The minimum delay between the assertion of SCA_WRITE and

GEN_GO is 20 ns. Pulse injection in the very first SCA cells is not possible when the SCA write clock is 100 MHz.

The bit PUL_ENABLE is used to enable/disable the use of the pulser. When this be is set and PUL_TRIG_ENABLE is set, a trigger will automatically be generated when the pulser fires. When PUL_ENABLE is set without PUL_TRIG_ENABLE being set, the multiplicity trigger should be enabled. The bit MULT_TRIG_ENA is used to enable or disable the self-trigger based on multiplicity. When this bit is set, a self-trigger will be generated whenever the multiplicity output of one or several of the four AGET chips controlled by the Feminos passes the multiplicity threshold programmed.

The bit SND_MULT_ENA is used to enable or disable the transmission of the multiplicity-over-threshold bits to the TCM. When this bit is active, the TCM can be programmed to generate a trigger based on system-level multiplicity-over-threshold bits.

The bit ERASE_MULT_ENA is used to enable the function that clears the content of the hit register before digitization for chips that have a number of channel hit above a programmable threshold. This bit is only active in the AGET mode and when MODIFY_HIT_REG is also set.

The POLARITY field is used to determine, on a per-chip basis, the behavior of the zero-suppressor depending on the polarity of detector signals. When a POLARITY bit is 0, the zero-suppressor keeps samples that are above threshold. This is intended to be used with detectors that deliver negative signals. When a POLARITY bit is 1, the zero-suppressor keeps samples that are below threshold. This mode is used with detectors that deliver positive signals. The polarity of each of the four AFTER or AGET chips controlled by a Feminos can be programmed independently.

The fields ERASE_HIT_THR specify for each AGET chip the maximum number of channel hit that are retained when ERASE_MULT_ENA and MODIFY_HIT_REG are set. In this mode, the content of the hit channel register is cleared if the number of hit channel is above the specified threshold. The threshold is a 3 bit value programmable from 0x0 to 0x7 and corresponds to a maximum allowable number of hit channels of 4 to 32 in increments of 4 channels.

Multiplicity Thresholds (Register #4):

The Feminos card can generate a local multiplicity trigger by applying a programmable threshold to each of the multiplicity signal of the four AGET chips. Each threshold is an 8-bit unsigned integer. Each threshold is only applied if it is greater than 0. The content of Register #4 is shown in Fig. 13.

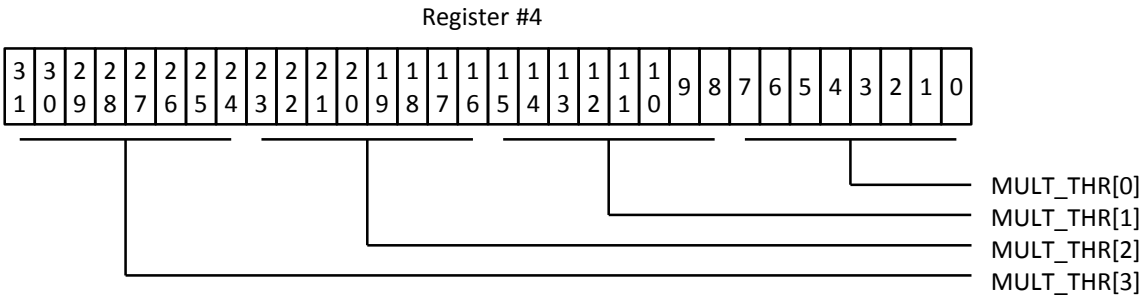


Fig. 13. Multiplicity Enable and Threshold (Register #4).

Configuration (Register #5):

This register is described in Fig. 14.

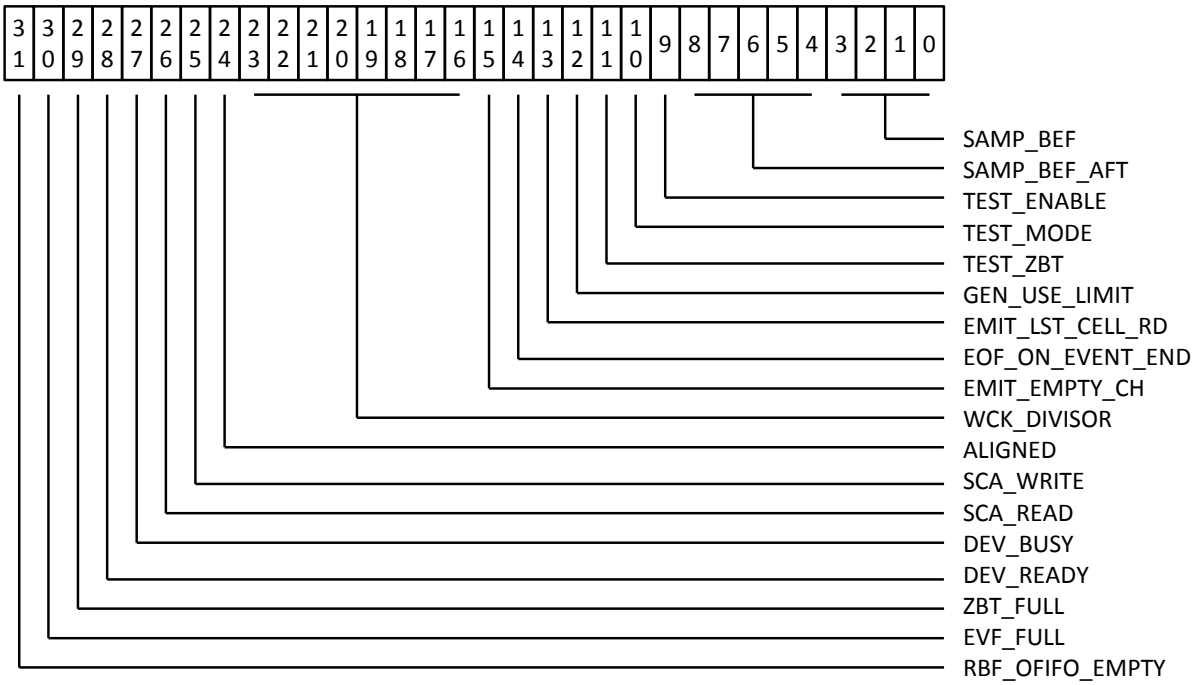


Fig. 14. Configuration (Register #5)

The field SAMP_BEF is used to specify how many samples should be kept before threshold is passed in zero-suppressed readout mode. The acceptable value is in [0; 15].

The field SAMP_BEF_AFT specifies the sum of the sample count kept before and after threshold is passed in zero-suppressed readout mode. The acceptable value is in [0; 31].

The TEST_ENABLE bit is used to select the source of data before the zero-suppression block. This bit should be cleared for normal operation where the data to

process comes from the ADC of the external front-end card. When the TEST_ENABLE bit is set to 1, internally generated test data are used instead of ADC data.

When TEST_ENABLE is set to 1, the TEST_MODE bit is used to choose one of two types of test data. When TEST_MODE is 0, test data are formed with the 12-LSBs of the memory address of the ADC sample being readout. This setting is for advanced users only. When TEST_MODE is 1, test data are formed by sequentially reading-out an internal memory array. This memory array can be configured by the user with arbitrary values or some pre-defined simple patterns. Refer to the relevant section for details.

The bit TEST_ZBT is used to select the data written to the ZBT SRAM event buffer. When TEST_ZBT is 0, the data received from the ADC of the FEC are written to this memory. When TEST_ZBT is set to 1, the 12-LSB's of the write pointer to the event buffer are written instead. This setting shall only be used for debugging or diagnosis.

The bit GEN_USE_LIMIT determines how the field EVENT_CNT_LIMIT is interpreted. When GEN_USE_LIMIT is set to 1, it is assumed that the internal event generator determines the number of events to generate. When SCA are enabled, the generator will make the desired number of events at the programmed rate and will stop. Events that could not be recorded by the SCAs will be dropped. When GEN_USE_LIMIT is 0, the internal generator or other sources of trigger will remain active until the SCAs have captured the programmed number of events.

The bit EMIT_LST_CELL_RD determines if the last cell read pointer retrieved from each ASIC are added or not to the stream sent to the DAQ. The value of the last cell read pointer of all ASICs can be compared at the DAQ level or later off-line to verify the correct synchronization of the front-end ASICs.

The bit EOF_ON_EVENT_END determines whether the frame being prepared is sent to the DAQ as soon as an end of event is found or not. When EOF_ON_EVENT_END is 0, data frames sent to the DAQ are filled for optimal payload usage and a frame may contain the beginning of the data of the next event after the end of the current event. When EOF_ON_EVENT_END is 1, the partially filled frame will be sent to the DAQ as soon as the end of event is reached.

The bit EMIT_EMPTY_CH determines the content of the output stream sent to the DAQ PC for channels that were digitized but whose data has been entirely removed by the zero-suppression stage. When EMIT_EMPTY_CH is set to 1, empty channels will be sent to the DAQ PC. The information send for an empty channel comprises a 16-bit word to identify the channel index followed by a null 16-bit word to indicate that there is no data for this channel. When EMIT_CH_EMPTY is cleared, channels that have no data above threshold are not sent to the DAQ PC. This makes events more compact.

The field WCK_DIVISOR is used to set the value of the divisor to generate the SCA write clock. The SCA write clock is obtained by dividing the 100 MHz reference clock by WCK_DIVISOR. Up to firmware version 2.10, valid values for WCK_DIVISOR are [1; 63] leading to discrete frequency values of 100 MHz, 50 MHz, 33 MHz, 25 MHz, etc., down to ~1.59 MHz. In firmware version 2.11 and above, the range is extended to [1; 255] leading to a minimum sampling frequency of ~392 kHz.

The ALIGNED flag indicates that the ADC correctly delineates received data from the framing signal FCO. The binary pattern received on this line is six 1's followed by six 0's.

The SCA_WRITE and SCA_READ flags indicate that the front-end ASICs are currently in the write mode and read mode respectively.

The DEV_BUSY flag indicates that the Feminos is currently busy with the readout of the current event and is not ready to put front-end SCAs in the write state. The DEV_READY flag indicates that the Feminos is ready to change its internal state to put front-end SCAs in the write state.

The ZBT_FULL flag indicates that the memory for storing event data is full and cannot accept the next event. The EVF_FLAG indicates that some of the internal FIFOs used to store intermediate event information are full. The Feminos will not resume SCA write until space is available in the event memory and in these FIFOs. The RBF_OFIFO_EMPTY flag indicates that the out-bound FIFO of the ring buffer interface does not contain any buffer descriptor. The Feminos will suspend the readout of the event memory and data encapsulation in Ethernet frames until free buffer descriptors are appended to this FIFO.

Configuration (Register #6):

This register is described in Fig. 15.

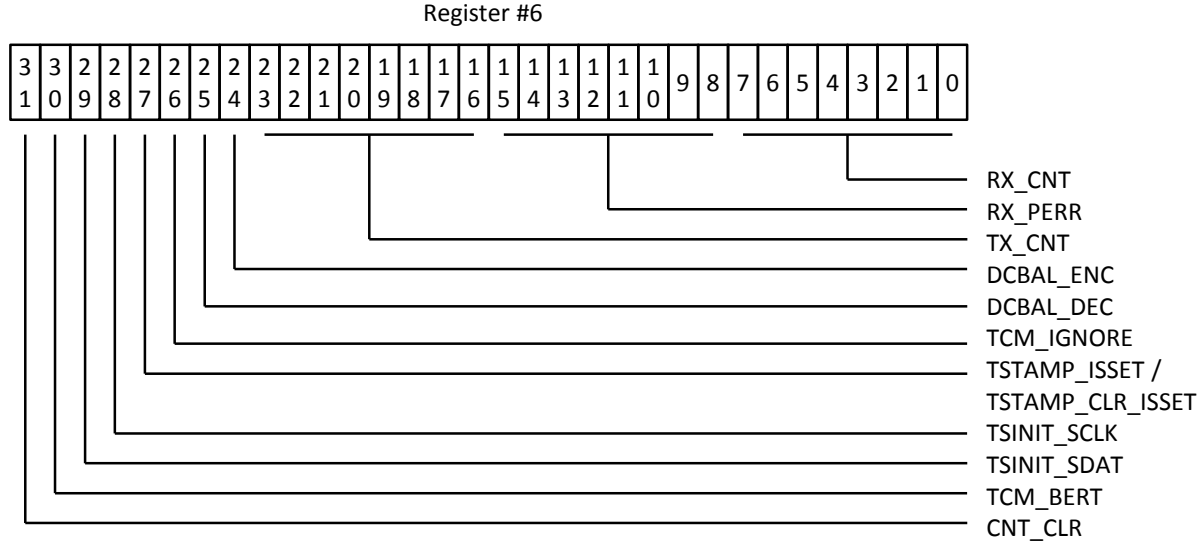


Fig. 15. TCM interface register.

The field RX_CNT counts the number of messages received from the TCM. The field RX_PERR counts the number of parity errors in messages received from the TCM. The field TX_CNT counts the number of messages sent to the TCM. Both RX_CNT and TX_CNT roll over after 255 while RX_PERR saturates at 255. All counters are cleared when CNT_CLR is asserted. The CNT_CLR bit must be cleared after it has been set to enable counting.

The bit DCBAL_ENC enables DC balanced encoding for the dataflow sent by the Feminos to the TCM. The bit DCBAL_DEC enables DC balanced decoding for the dataflow received from the TCM.

The bit TCM_IGNORE is used to optionally mask the presence of the TCM – except for the reference clock. When TCM_IGNORE is active, all frames received from the TCM are dropped and no frame is sent to the TCM.

When read, the bit TSTAMP_ISSET indicates whether or not the Feminos has received a time-stamp preset signal since the last reset or the last time that watchdog flag was cleared. To clear this bit, a '1' followed by a '0' has to be written in TSTAMP_CLR_ISSET (same bit position). The read value will be '0' until the Feminos receives, via software or from the TCM, a time-stamp preset.

The field TSINIT_SCLK and TSINIT_SDAT are used to serially load an initial preset value for the timestamp counter. This 48-bit preset value is loaded in the timestamp counter of the Feminos when it receives a synchronous clear signal via the TCM or an asynchronous clear command from software. The preset value is loaded MSB first and each bit of data is captured on the transition of TSINIT_SCLK from 0 to 1. The preset value cannot be read-back.

The bit TCM_BERT is used to enable or disable the bit error rate tester function of the Feminos. When this function is engaged, the Feminos checks the messages received from the TCM against the successive values of a local pseudo-random generator and it continuously sends to the TCM messages that contain the successive values of that local pseudo-random generator.

Free running clock cycle counter (Register #7):

This register is a free running counter which is incremented every 20 ns. It provides the time reference needed to measure time intervals in software. This timer rolls over every 1' 25".

Multiplicity Limits (Register #8):

Starting from firmware version 2.6, the logic that generate the multiplicity information is changed. Two comparators are now used on the multiplicity signal delivered by each AGET chip. To generate a multiplicity hit, it is now required that the multiplicity signal is above a programmable threshold and is simultaneously less than a programmable limit.

The multiplicity limits are mapped to Register #8 as shown in Fig. 16. Each multiplicity limit is an 8-bit unsigned integer. It must be different from 0 if multiplicity trigger is used.

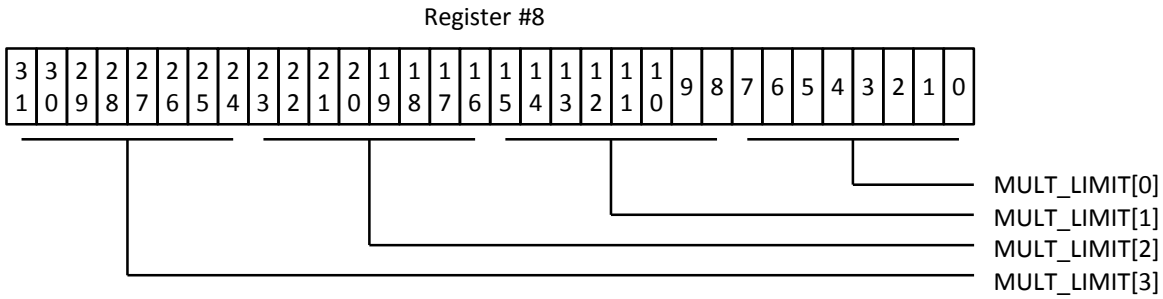


Fig. 16. Multiplicity limit register.

Extended Configuration Register (Register #9):

This register is used in firmware version 2.11 and above.

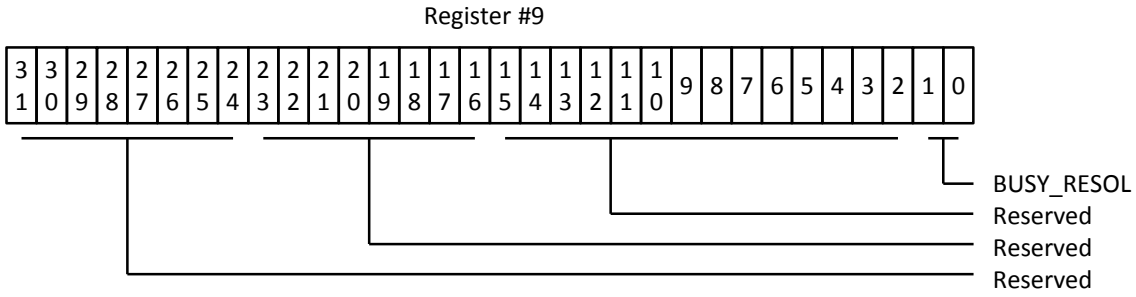


Fig. 17. Multiplicity limit register.

The field BUSY_RESOL sets the resolution of the timer for the accumulation of the dead-time histogram. Four resolutions are available: 1 µs ("00"), 10 µs ("01"), 100 µs ("10") and 1 ms ("11").

5.2 Dual-ported Memory Blocks

The embedded processor of the Feminos card also communicates with the local logic through several dual-ported memory blocks. Each memory block is mapped to a 64 Kbyte address range in the 32-bit address space of the processor. The base address of each block is assigned by Xilinx XPS tool. Each memory region is mapped to an array

declared as a global variable in the embedded software and the compiler/linker must be instructed to map each array to the appropriate address range.

Table 2 . Variables mapped to dual-ported RAM locations.

Variable	Element Type	Function
pedthrlut[4][128]	short, ushort	Pedestal and Threshold Table
hitregrule[4][128]	ushort, ushort	Hit Register Rules Table
testdata[4096]	unsigned int	Test Data Table
hbusy[1024]	unsigned int	Dead-time histogram bins
hhitcnt[4][128]	unsigned int	Histogram of per event channel hit count

pedthrlut:

This table is used to store the constant pedestal and threshold value for each channel of the 4 ASICs controlled by the Feminos card. Although 128 entries are available for each ASIC, only 70, 72 or 79 entries are used depending on the selected mode (AGET or AFTER) and the number of reset cycles (2 or 4 in the AGET mode). The per channel pedestal value is a 9-bit signed integer that is added to channel data when the bit PED_SUBTRACT of the configuration register is set. The available range for pedestal values is [-4096; +4095] ADC counts. Pedestals are coded in 2's complement. Threshold values are used to perform zero-suppression when the ZERO_SUPPRESS configuration bit is set. The zero-suppression phase takes place after the optional pedestal subtraction. Threshold values are 9-bit unsigned integers. The available range is [0; 4095] ADC counts.

hitregrule:

When operating in the AGET mode, this table is used to store the rules that determine how to optionally alter the content of the Hit Channel Register before SCA digitization. Each 32-bit entry in this table maps to two binary fields, ForceOn (bit 0) and ForceOff (bit 16). When ForceOn and ForceOff are cleared, the content of the corresponding bit in the Hit Channel Register is unaltered. When ForceOn is set to 1, the corresponding channel will be readout even if it was not hit. Reciprocally, when ForceOff is set to 1, the corresponding channel will be skipped from readout even if it was hit. ForceOn and ForceOff bits may not be set simultaneously. The table contains 128-entries per AGET but only 70 or 72 entries are used. Note that the channels corresponding to the reset sequence of the SCA (2 or 4 channels) cannot be skipped at this stage and the ForceOn/ForceOff flags have no effect on these channels. The readout of FPN channels is controlled by register settings in the AGET chips but the relevant ForceOn/ForceOff entries must also be programmed in a coherent way.

testdata:

This table is used to store a programmable pre-defined pattern of data to operate the Feminos card in test mode. Each table entry is interpreted as a 12-bit unsigned ADC value corresponding to the successive time-bins of the channels being hit. The table has 4096-entries. Assuming that the user wishes to exercise the readout with 512 time-bins per channel, different arbitrary data for up to 8 channel hit can be programmed. Table entries are re-scanned from the first address for each event, and wraparound occurs when the number of channel hit multiplied by the number of time bins per channel exceeds table size.

hbusy:

This memory region is used to store the bins of the dead time histogram. Histogram accumulation is handled by the firmware side but software is responsible for clearing histogram bins. The dynamic range of each bin is 18 bits up to firmware version 2.2 and is extended to 32 bits in more recent versions.

hhitcnt:

The Feminos card accumulates the histogram of the number of channels hit per event and per ASIC (i.e. 4 histograms are stored). The maximum number of channels hit is 72 and 79 in the AGET and AFTER mode respectively (recall that pedestal channels and reset channels are also counted), but enough space for 128 bins per histogram is reserved. Each histogram bin is a 32-bit unsigned value.

5.3 Ring Buffer Interface

The third and last interface between the embedded processor and the internal logic within the FPGA of the Feminos card is a set of two embedded FIFO's and a control register that are used to assist the transfer of data from the local logic to the SDRAM of the Feminos card with minimal intervention of the local processor. The address map of these resources is shown in Table 3. The base address is mapped into the 32-bit address space of the MicroBlaze processor. All addresses are aligned on 32-bit boundaries and little-endian byte ordering is assumed.

Table 3 . Ring-buffer interface.

Address	Access	Function
Base+0x00	R/W	Ring Buffer Control/Configuration
Base+0x04	R	Logic to processor in-bound FIFO
Base+0x04	W	Processor to logic out-bound FIFO

After the Ring Buffer Configuration has been programmed, the processor writes to the out-bound FIFO free buffer descriptors. As event data is produced, the local logic fetches buffer descriptors from the out-bound FIFO, fills the corresponding memory

locations with data, and posts to the in-bound FIFO buffer descriptors that points to memory locations ready to be sent to the remote DAQ computer. The local processor consumes buffer descriptors from the in-bound FIFO, fills the remaining header information and initiates transfer of the corresponding Ethernet frames via a DMA engine. After transmission, buffer descriptors corresponding to the memory locations that can be freed are re-cycled in the out-bound FIFO.

The Ring Buffer Control/Configuration register is shown in Fig. 18.

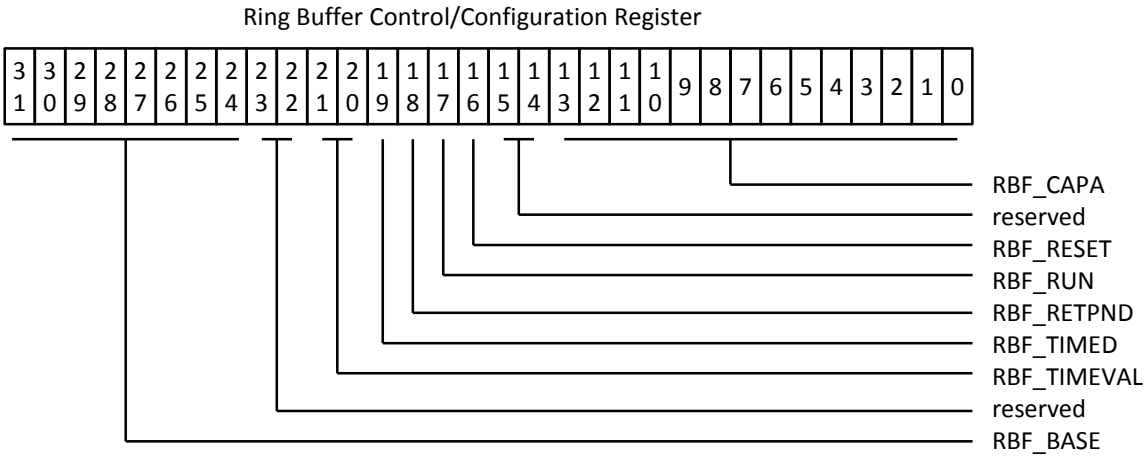


Fig. 18. Ring Buffer Control/Configuration Register.

The RBF_CAPA field specifies in bytes the size of the buffers pointed by the buffer descriptors exchanged over the in-bound and out-bound FIFOs. All buffers must have an equal size. The size must be a multiple of 4 bytes and must be greater than the Ethernet MTU. Jumbo frames up to 8 KByte can be accommodated.

The RBF_RESET field should be set and cleared by the embedded software to initialize the local logic properly.

The RBF_RUN bit should be set to 1 to start data transfers after configuration and after a sufficient number of free buffer descriptors have been posted to the hardware. This bit can be cleared to stop data transfers. The RBF_RETPND bit can be pulsed to force the local logic to abort filling the current buffer descriptor and post it to the processor in-bound FIFO.

The RBF_TIMED bit should be set to 1 to operate the ring buffer in a mode where partially filled buffers are sent after a timeout period has elapsed. The field RBF_TIMEVAL determines the time to wait before sending a partially filled buffer. The four possible timeout values are 1 ms, 10 ms, 100 ms and 1 s. When RBF_TIMED is cleared, the ring buffer logic will wait indefinitely for data to optimally fill buffers before sending them (unless RBF_RETPND is pulsed).

The RBF_BASE field gives the 8 MSB's of the memory base address where the array of data buffers is placed. This base address must start on a 16-MByte boundary and normally points to a region in the SDRAM of the Feminos card.

The buffer descriptors exchanged over the in-bound and out-bound FIFOs are 32-bit words. Hence fetching or posting a descriptor only requires a single 32-bit access. The fields of a buffer descriptor are shown in Fig. 19.

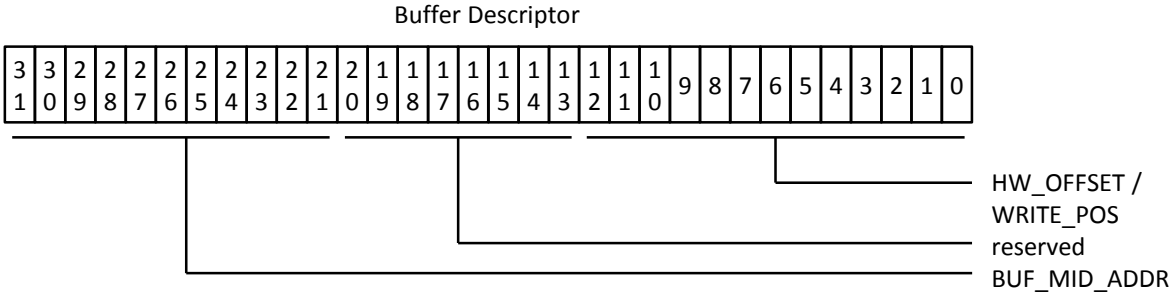


Fig. 19. Buffer Descriptor.

The HW_OFFSET field is used in descriptors posted to the out-bound FIFO to indicate to the embedded logic the location of the first free location in the corresponding buffer. The embedded software should set a non-null value in this field to reserve the necessary space to put Ethernet and other protocol headers (i.e. Ethernet frame header, IP and UDP or TCP headers). More space can be reserved to pre-pend specific and user defined header information in software after the core of data buffer has been filled by the hardware. The value of HW_OFFSET must be multiple of 4. Assuming standard Ethernet and UDP/IP framing, the header information is actually only 42 bytes. This must be rounded up to 44 bytes and the first two bytes of the UDP payload are always set to zero in the present implementation.

When descriptors are read from the in-bound FIFO, the 13 LSB's of the buffer descriptor are the current position where the hardware has stopped filling data. Software may append trailing information starting from this offset. This field is also used to determine the size of data that has been filled in the buffer pointed by the descriptor.

The BUF_MID_ADDRESS is used in conjunction with the RBF_BASE address to determine the physical memory base address of the buffer pointed by this descriptor. The 32-bit base address of the buffer is formed by adding the RBF_BASE to the BUF_MID_ADDRESS field shifted right by 8-bits. The resulting address is 19-bit and is aligned to 8 KByte boundaries (which is the maximum capacity of a buffer). The maximum number of buffer descriptors is 2048 corresponding to up to a 16-MByte buffer array. However, the in-bound and out-bound FIFOs have only 511 positions each

and it is recommended to limit the number of descriptors to less than 511 to avoid any overflow.

6 COMMAND SERVER REFERENCE

The axi_data_server application is a command server program running on the embedded processor of the Feminos card. This server processes the commands received from a remote DAQ PC over a standard Ethernet connection and returns results via the same path. All commands are formatted in ASCII and perform basic to complex operations on the Feminos card. Commands can be classified into three categories: configuration commands, monitoring / run control commands and data acquisition commands. The Feminos can use only one UDP/IP socket when the client program running on the DAQ PC performs all the tasks of system configuration, monitoring, run control and data acquisition. The Feminos uses two UDP/IP sockets when different programs, possibly running on different hosts, are used for the tasks mentioned above. In this case, one socket is dedicated to the data acquisition path while the second one handles other types of commands.

6.1 GENERAL CONTROL COMMANDS

General control commands are listed in Table 4.

Table 4 . General control commands.

Command	Argument	Action
version		Returns major/minor software version and compilation date
cmd	clr	Clears command sent/replies counters
cmd	stat	Shows command statistics counters
reg	<r>	Reads Feminos register <r>
reg	<r> <0xdata>	Writes Feminos register <r> with <0xdata>
help		Prints a short list of commands
mode		Prints the current mode of operation (AGET or AFTER)
mode	<after | aget>	Sets the operating mode to AGET or AFTER
fec_enable		Shows the state of FEC enable bit
fec_enable	<0 | 1>	Disables/enables the FEC (power OFF/ON)
power_inv		Shows the state of the power inverter flag
power_inv	<0 | 1>	Sets the power inversion flag
asic_mask		Shows which ASICs are active/masked
asic_mask	<0xMask>	Sets which ASICs are active/masked (4-bit mask)
emit_hit_cnt		Shows if per ASIC total channel hit count is sent in the data stream or not
emit_hit_cnt	<0 | 1>	Disable/enable the option to send the per ASIC total channel hit count
rst_len		Shows the current setting of AGET reset channel count (2 or 4)
rst_len	<0 | 1>	Selects the setting for AGET reset channel count (0

		leads to 2; 1 leads to 4)
forceon_all		Shows current setting to skip reading the Channel Hit Register and force readout of all channels
forceon_all	<0 | 1>	Clear or sets the option to force the readout of all channels in AGET mode.
keep_rst		Shows if reset channels are removed or kept in the data stream sent to the DAQ PC
keep_rst	<0 | 1>	Disable/enable the option to keep some of the reset channels in the data stream sent to the DAQ PC
skip_rst		Shows the current setting that determines which reset channels are dropped
skip_rst	<0,1,2,3>	Sets how many reset channels should be dropped
emit_lst_cell_rd		Shows the setting for the option to send in the data stream for the DAQ the value of the last cell read pointer of each ASIC
emit_lst_cell_rd	<0 | 1>	Disable or enable the option to send the last cell read pointer of each ASIC
state		Shows the current internal state of the Feminos

The user must enable the FEC using the command "fec_enable 1" before the card can be programmed or used. When the FEC is enabled, a high level is placed on the interface pin REG_INH_B so that the FEC is powered ON. Some FECs are equipped with a voltage regulator that has an active low enable input. The "power_inv 1" command should be used in this case to perform the required signal inversion. The FEC can be turned OFF to conserve power or to clear all its internal settings.

By default, all four ASICs are active and ASIC_MASK is b0000. One or several ASICs may be disabled by setting the appropriate mask. All values are valid and ASIC_MASK=0xF disables all ASICs. This leads to empty events.

6.2 Hit Channel Register Commands

In the AGET mode when the forceon_all option is disabled, the Feminos reads the Hit Channel register to determine the set of channels to readout. The content of this register can optionally be modified before SCA digitization. The commands listed in Table 5 determine the corresponding alteration rules.

Table 5 . Commands to act on Channel Hit Register.

Command	Argument	Action
modify_hit_reg		Shows the current setting for the modify hit register option
modify_hit_reg	<0 | 1>	Disable/enable Channel Hit Register alteration before SCA digitization
forceon	<A> <C>	Shows the setting to force the readout of channel <C> of Asic <A>

forceon	<A> <C> <0 | 1>	Disable/enable a forced readout for channel <C> of Asic <A>
forceoff	<A> <C>	Shows the setting to force skipping readout of channel <C> of Asic <A>
forceoff	<A> <C> <0 | 1>	Disable/enable a forced skip of channel <C> of Asic <A>
erase_hit_ena		Shows the setting to clear hit channel registers when the channel hit count is too high
erase_hit_ena	<0 | 1>	Disable/enable the function to clear hit channel registers when the channel hit count is too high
erase_hit_thr	<A>	Shows the hit count limit threshold of Asic <A>
erase_hit_thr	<A> <0xThr>	Sets the hit count limit threshold of Asic <A> to <0xThr>

When both forceon and forceoff options are disabled for a given channel, the value read from the Channel Hit Register is preserved. Both forceon and forceoff options must not be set simultaneously. Note that for programming rules, the forceon and forceoff commands accept scalar arguments, ranges or wildcard characters for arguments <A> and <C>. For example, the command "forceon * 3:78 1" sets to 1 the forceon flag for channel 3 to 78 of all ASICs.

Starting with firmware version 2.7, the Hit Channel Register of each chip can be cleared automatically before SCA digitization when the number of hit channels passes a programmable limit. This feature is intended to minimize the dead-time caused by excessively busy events or noise events where almost all channels fire. This feature is only active when ERASE_HIT_ENA and MODIFY_HIT_REG are set. The threshold limit is programmable from 0x0 to 0x7 and corresponds to a maximum acceptable channel hit count of 4 to 32 channels. For example, if the threshold of a chip is set to 0, the Channel Hit Register of that chip will not be altered for events that have from 1 to 4 channels hit, and it will be cleared for events that have 5 channels hit or more. The number of channels hit in each chip is computed by the FPGA logic when reading the Hit Channel Register of each chip before SCA digitization and independently of the ForceOn and Force_Off rules. Note that the ForceOn and ForceOff rules are still applied to determine the final value of each Channel Hit Register when ERASE_HIT_ENA is set.

PEDESTAL EQUALIZATION AND ZERO-SUPPRESSION COMMANDS

The Feminos card can optionally perform pedestal subtraction and apply a threshold to zero-suppress data. A per channel constant pedestal value and threshold value can be programmed. The commands to set or read-back pedestal values and threshold values are listed in Table 6.

Table 6 . Commands for pedestals and thresholds.

Command	Argument	Action

ped	<A> <C>	Shows the pedestal equalization constant of channel <C> of Asic <A>
ped	<A> <C> <0xped>	Sets the pedestal equalization constant of channel <C> of Asic <A>
subtract_ped		Show the current setting for pedestal subtraction
subtract_ped	<0 | 1>	Disable/Enable pedestal subtraction
thr	<A> <C>	Shows the threshold of channel <C> of Asic <A>
thr	<A> <C> <0xthr>	Sets the threshold of channel <C> of Asic <A>
zero_suppress		Shows if zero-suppression is enabled or not
zero_suppress	<0 | 1>	Disable/enable data zero-suppression
polarity	<A>	Shows the polarity of the zero-suppressor of Asic <A>
polarity	<A> <0 | 1>	Sets the polarity of the zero-suppressor for Asic <A>. 0 to keep data above threshold; 1 to keep data below threshold
zs_pre_post		Shows the number of samples to keep before/after threshold is passed in zero-suppressed readout mode
zs_pre_post	<pre> <post>	Sets the number of samples to keep before/after threshold is passed in zero-suppressed readout mode
emit_empty_ch		Shows the current policy for sending or skipping empty channels in the data stream sent to the DAQ
emit_empty_ch	<0 | 1>	Sets the policy for handling empty channels: drop silently or send channel index and one null value

Acceptable values for the pedestal are 0xF000 (-4096) to 0x0FFF (+4095). Threshold values are from 0x0000 (0) to 0x0FFF (4095). When programming pedestal equalization constants and thresholds, commands arguments <A> and <C> can be scalars, ranges or "*" wildcard characters.

The zero-suppressor engine not only keeps data samples that are above the programmable threshold but also allows keeping a programmable number of samples before the threshold is passed and after the threshold is no longer passed. From 0 to 15 precursor samples and 0 to 16 trailer samples can be kept on zero-suppressed waveforms. When working with detectors that produce positive signals, the zero-suppressor must be programmed to keep pulses that are below threshold rather than above.

When no data remains for a given channel after the zero suppression, either no data at all is sent for this channel, or the index of that channel followed by a null

sample can be sent. The command "emit_empty_ch" is used to set the desired option. Suppressing empty channels leads to more compact events while keeping empty channel can be useful for debugging.

6.3 ASIC Register Configuration Commands

The AFTER and AGET chips have 3 and 13 programmable configuration registers respectively. These registers can be programmed and read-back via the commands listed in Table 7 and Table 8 for the AGET and AFTER chips respectively. Because registers have a different size (from 16-bit to 128-bit), the correct number of 16-bit arguments has to be supplied. Refer to the documentation of the AFTER and AGET chips for a detailed description of internal registers.

Table 7 . Commands for operations on AGET internal registers.

Cmd	Argument	Action
aget	<A> read <reg>	Reads the content of register <reg> of AGET chip <A>
aget	<A> write <reg> <0xVal0> …	Writes register <reg> of AGET <A> with the specified value (16-bit per argument, supply the required number of values to match the actual size of the register)
aget	<A> wrchk <reg> <0xVal> …	Performs a write to register <reg> followed by a read and comparison
aget	<A> icsa	Reads the setting for the bias current of charge sense preamplifiers of AGET <A>
aget	<A> icsa <0 | 1>	Sets the bias current of the charge sense preamplifiers of AGET <A>: 0 normal bias; 1: normal bias x 2
aget	<A> time	Reads the shaping time of AGET <A>
aget	<A> time <0xVal>	Sets the shaping time of AGET <A> to <0xVal>
aget	<A> test	Shows Test setting of AGET <A>
aget	<A> test <0xVal>	Sets Test of AGET <A> to <0xVal>
aget	<A> mode	Shows Readout mode setting of AGET <A>
aget	<A> mode <0xVal>	Sets Readout mode of AGET <A> to <0xVal>
aget	<A> polarity	Reads the polarity setting of AGET <A>
aget	<A> polarity <0 | 1>	Sets the polarity of AGET <A>: 0 for detectors with negative signals, 1 for positive
aget	<A> fpn	Reads the FPN setting for AGET <A>
aget	<A> fpn <0xVal>	Sets the FPN readout of AGET <A>
aget	<A> vicm	Shows Vicm setting of AGET <A>
aget	<A> vicm <0xVal>	Sets Vicm of AGET <A> to <0xVal>
aget	<A> dac	Reads the DAC threshold of AGET <A>
aget	<A> dac <0xVal>	Sets the DAC threshold of AGET <A>

aget	<A> trigger_veto	Reads the trigger veto of AGET <A>
aget	<A> trigger_veto <0xVal>	Sets the trigger veto of AGET <A>
aget	<A> synchro_discri	Reads the synchronization discriminator bit of AGET <A>
aget	<A> synchro_discri <0 |1>	Clears or sets the synchronization discriminator bit of AGET <A>
aget	<A> tot	Reads the time-over-threshold bit of AGET <A>
aget	<A> tot <0 | 1>	Clears or sets the time-over-threshold bit of AGET <A>
aget	<A> range_tw	Reads the range bit of trigger width of AGET <A>
aget	<A> range_tw <0 | 1>	Clears or sets the range bit of trigger width of AGET <A>
aget	<A> trig_width	Reads the trigger width of AGET <A>
aget	<A> trig_width <0xVal>	Sets the trigger width of AGET <A>
aget	<A> rd_from_0	Reads the bit read_from_0 of AGET <A>
aget	<A> rd_from_0 <0 |1>	Clears or sets the bit read_from_0 of AGET <A>
aget	<A> tst_digout	Reads the bit tst_digout of AGET <A>
aget	<A> tst_digout <0 | 1>	Encode the index of the last SCA cell read or a fixed pattern (0x159) at the end of the SCA read phase of AGET <A>
aget	<A> en_mkr_rst	Reads en_mkr_rst setting of AGET <A>
aget	<A> en_mkr_rst <0 | 1>	Clears or sets en_mkr_rst of AGET <A>
aget	<A> rst_level	Reads rst_level setting of AGET <A>
aget	<A> rst_level <0 |1>	Clears or sets rst_level of AGET <A>
aget	<A> cur_ra	Reads the SCA line buffer current setting of AGET <A>
aget	<A> cur_ra <0xCur>	Sets the SCA line buffer current of AGET <A>
aget	<A> cur_buf	Reads the buffer current setting of AGET <A>
aget	<A> cur_buf <0xCur>	Sets the buffer current setting of AGET <A>
aget	<A> short_read	Reads short_read setting of AGET <A>
aget	<A> short_read <0 | 1>	Clears or sets short_read of AGET <A>
aget	<A> dis_multiplicity_out	Reads dis_multiplicity_out setting of AGET <A>
aget	<A> dis_multiplicity_out <0 | 1>	Clears or sets dis_multiplicity_out of AGET <A>
aget	<A> autoreset_bank	Reads autoreset_bank setting of AGET <A>
aget	<A> autoreset_bank <0 | 1>	Clears or sets autoreset_bank of AGET <A>
aget	<A> in_dyn_range	Reads in_dyn_range setting of AGET <A>
aget	<A> in_dyn_range <0 | 1>	Clears or sets in_dyn_range of AGET <A>

aget	<A> gain <c>	Reads the gain setting of AGET <A> Channel <c>
aget	<A> gain <C> <0xVal>	Sets the gain of channel(s) <C> of AGET <A> to <0xVal>
aget	<A> inhibit <c>	Reads the inhibit setting of AGET <A> Channel <c>
aget	<A> inhibit <C> <0xVal>	Sets the inhibit of channel(s) <C> of AGET <A> to <0xVal>
aget	<A> threshold <c>	Reads the threshold setting of AGET <A> Channel <c>
aget	<A> threshold <C> <0xVal>	Sets the threshold of channel(s) <C> of AGET <A> to <0xVal>
aget	<A> threshold ++ | --	Increments / decrements the threshold of AGET <A>
aget	<A> hitprob <C> <Val>	Sets the threshold of chanel(s) <C> of AGET <A> so that the probability that this channel appears as hit is less than <Val> (assumes that histograms of hit probability have been accumulated)

All commands that perform a write operation support argument <A> in the following format: the "*" wildcard to designate all AGET chips, a range (two integers separated by the ":" character") or a single index. Read operations can only apply to a single AGET and eventually a single channel. The majority of the commands that modify the content of AGET registers perform internally a write followed by a read and a comparison. If the command did not return an error, it is reasonably safe to consider that the register was modified as intended.

Table 8 . Commands for operations on AFTER internal registers.

Cmd	Argument	Action
after	<A> read <reg>	Reads the content of register <reg> of AFTER <A>
after	<A> write <reg> <0xVal0> …	Writes register <reg> of AFTER <A> with the specified value (16-bit per argument, supply the required number of values to match the actual size of the register)
after	<A> wrchk <reg> <0xVal0> …	Writes register <reg> of AFTER <A> with the specified value, performs read-back and verification
after	<A> gain	Shows the current gain of AFTER <A>
after	<A> gain <120,240,360,600>	Sets the gain of AFTER <A> to a dynamic range of 120 fC, 240 fC, 360 fC or 600 fC
after	<A> time	Shows the shaping time of AFTER <A>
after	<A> time <stime_ns>	Sets the shaping time of AFTER <A> to <stime_ns> (16 possible values)

after	<A> test_mode	Shows the test mode of AFTER <A>
after	<A> test_mode <0 xMode>	Sets test mode of AFTER <A> to <0xMode> (4 possible values)
after	<A> en_mkr_rst	Reads the value of the enable marker reset flag of AFTER <A>
after	<A> en_mkr_rst <0 | 1>	Disable/enable the reset marker of AFTER <A>
after	<A> rst_level	Reads the level of the reset marker of AFTER <A>
after	<A> rst_level <0 |1>	Sets to low or high the level of the reset marker of AFTER <A>
after	<A> rd_from_0	Reads the bit read_from_0 of AFTER <A>
after	<A> rd_from_0 <0 | 1>	Disable/enable the option to force SCA read from cell 0
after	<A> test_digout	Reads the bit test_digout of AFTER <A>
after	<A> test_digout <0 | 1>	Disable/enable the option to send a fixed pattern (0x159) instead of the index of the last cell read at the end of the SCA read phase

For AFTER register modification commands, argument <A> also accepts wildcards, ranges and scalar values. For read-back commands, only a scalar value for <A> is accepted. Contrary to AGET related commands, no verification of register content is made after a modification command (except for the "wrchk" command). The verification may be added in future embedded software releases.

6.4 Trigger Related Commands

The Feminos card includes a flexible trigger and a programmable rate on-board event generator. The corresponding control commands are listed in Table 9.

Table 9 . Trigger and event generator control commands.

Command	Argument	Action
trig_enable		Reads the current setting of trigger mask
trig_enable	<0xVal>	Enables the trigger sources specified by <0xVal>
trig_delay		Reads the current value of trigger delay
trig_delay	<0xVal>	Sets the trigger delay to <0xVal>
trig_rate		Shows the current trigger rate setting of the on-board event generator
trig_rate	<range> <rate>	Sets the trigger rate of the on-board generator to specified <range> and <rate>
event_limit		Shows the current setting for EVENT_LIMIT

event_limit	<0xVal>	Sets EVENT_LIMIT to <0xVal>
gen_use_limit		Shows the current setting of GEN_USE_LIMIT
gen_use_limit	<0xVal>	Sets the value of GEN_USE_LIMIT

The argument of trig_enable is a 4-bit hexadecimal value where each bit enables the following sources of trigger: bit 0 (AUTO_TRIG_ENABLE): on-board generator; bit 1 (EXT_TRIG_ENABLE): external trigger pin EXT_TRIG; bit 2 (PUL_TRIG_ENABLE): local trigger from pulser; bit 3 (TCM_TRIG_ENABLE): remote trigger received from the serial link to the TCM.

The argument of the trig_delay command is a 16-bit unsigned integer that expresses in 10 ns units the delay to apply to all types of triggers. The maximum delay is 65.535 µs. From firmware version 2.12 and beyond, the acceptable value for trig_delay is increased to 131068 corresponding to a maximum trigger latency of 1.31068 ms. However, if the supplied argument is above 32767, the resolution of the delay is increased from 10 ns to 40 ns.

The trig_rate command takes two arguments: 0 to 3 for the range and 1 to 100 for the rate. The corresponding rates are: range 0: from 0.1 Hz to 10 Hz in steps of 0.1 Hz; range 1: from 10 Hz to 1000 Hz in steps of 10 Hz; range 2: from 100 Hz to 10 kHz in steps of 100 Hz; range 3: from 1 kHz to 100 kHz in steps of 1 kHz. Note that setting a certain rate for the generator does not imply that the data acquisition chain will be able to sustain this rate. The actual rate versus the offered rate is an indicator of system performance.

The event_limit command is used to optionally stop taking new events after a programmable number of events have passed the SCA digitization stage. Allowing a known pre-defined number of events to flow through the system is useful for system development and data integrity checks. Setting EVENT_LIMIT to 0 disables this feature (i.e. an infinite number of events are allowed to pass through) while values from 0x1 to 0x7 lead to 1; 10; 100; 1000; 10,000; 100,000 or 1,000,000 events that are allowed to be digitized before the SCA write operation is no longer allowed. After the programmed number of event has been reached, the SCA_ENABLE bit must return to 0 and then 1 to resume data taking.

The gen_use_limit command is used to determine how the event count limit is evaluated. When GEN_USE_LIMIT is 1, only the events made by the internal generator, and all of them, are counted. This mode of operation is intended to measure the probability that events can pass through the system for some given rate and load scenario. When GEN_USE_LIMIT is 0, the event count limit is determined by the

number of events (all sources are included) that the SCAs were effectively able to capture.

6.5 SCA Related Commands

The operation of front-end SCAs is governed by the commands listed in Table 10.

Table 10 . SCA control commands.

Command	Argument	Action
sca	cnt	Reads the current setting for the number of SCA cells to readout
sca	cnt <0xVal>	Sets the number of SCA cells to digitize (from 8 to 511 or 512)
sca	wckdiv	Reads the current value of SCA Write clock divisor
sca	wckdiv <0xDiv>	Sets the SCA write clock divisor to <0xDiv> (from 1 to 63)
sca	enable	Reads the current value of the SCA_ENABLE bit
sca	enable <0 | 1>	Clears or sets the SCA_ENABLE bit
sca	autostart	Shows the current value of the SCA_AUTO_START bit
sca	autostart <0 | 1>	Clears or sets the SCA_AUTO_START bit
sca	start	Starts SCA write operation
sca	stop	Generates a software trigger event

After system configuration, the SCA_ENABLE bit must be set to 1 to be ready to take data. Clearing the SCA_ENABLE bit stops operation as soon as the readout of the current event is finished. The write operation in front-end SCAs starts immediately if the SCA_AUTO_START bit was set to 1. If not, writing in the SCAs will only start after the corresponding order has been received from the TCM. A single standalone Feminos setup normally requires SCA_AUTO_START to be set to 1 while a system with multiple cards operated in a common dead-time mode uses the opposite setting.

The sca start command is used when SCA_AUTO_START is inactive and the TCM is not present or disable. This command is normally only used when pulse generator of the FEC is being used. In this mode, the SCA start order is given in software and the stop order is automatically generated by the pulse generator control logic.

The sca stop command is used to issue a trigger in software for test purposes.

6.6 Synchronization Commands

In a multiple Feminos system, synchronization signals are distributed by the TCM. It is also possible to perform some of the functions of the TCM asynchronously for each Feminos card with the commands listed in Table 11.

Table 11. Synchronization commands.

Command	Argument	Action
clr	tstamp	Clears or presets (if an initial value different from 0 was pre-loaded) the time stamp counter
clr	evcnt	Clears the event counter
tstamp_init	0xVal_msb 0xVal 0xval_lsb	Loads a 48-bit preset value for the timestamp counter.
tstamp_isset		Shows if timestamp preset was received since last clear
tstamp_isset	clr	Clears the timestamp preset received indicator flag

If trigger generation is disabled during the operation, it is an acceptable method to clear the event counter sequentially in each card in a multiple Feminos system. Event numbers will match across the whole system after trigger is restarted (synchronously on all Feminos cards). On the other hand, event time stamps will only match after a synchronous clear/preset through the TCM has been made prior to data taking.

By default, the timestamp counter will be set to 0 upon clear. But any other initial value can be defined using the "tstamp_init" command. A clear of the timestamp counter will in fact cause a preset of this counter to the initial value programmed. This feature can be used to synchronize to the same value the timestamps across multiple systems. The initial value for the timestamp counter should be set to the value required to compensate for the propagation delay of the synchronous clear signal through the TCM and Feminos.

The command "tstamp_isset" is used to check if the Feminos received a synchronous message from the TCM with the bit CLR_TSTAMP set. This flag can be cleared with the command "tstamp_isset clr".

6.7 Pulse Generator Control Commands

The Feminos card can control the pulse generator of the FEC to inject calibrated charges at a well-defined time. The commands that control the pulse generator are listed in Table 12.

Table 12. Pulse Generator control commands.

Command	Argument	Action
pul_delay		Reads pulse generator delay
pul_delay	<0xDelay>	Writes pulse generator delay with <0xDelay>
pul_amp	<0xAmp>	Presets pulse generator amplitude DAC to <0xAmp>
pul_load		Loads the DAC with the preset value

pul_enable		Reads the state of the pulser enable bit
pul_enable	<0 | 1>	Enable / disable pulser injection

The command pul_delay sets the delay between the active edge of SCA_WRITE and the injection of the charge by the pulse generator. The delay is expressed in units of 10 ns and 20 ns are always added by internal logic to the value programmed in this register. The pul_amp command is used to program the serial DAC of the pulse generator. The value cannot be read-back. The pul_load command is used to perform the immediate update of the output of the DAC. This command must be called to set (or restore from the previous value) the baseline output level of the DAC for every pulse the user wants to generate. The pul_enable command is used to check, enable or disable the operation of the pulser. The pulser must be enabled when it is in use. The trigger associated to the pulser should be enabled for test or calibration with the pulser but should be disabled when testing triggers based on AGET multiplicity outputs.

6.8 Multiplicity Processing

The multiplicity output of the AGET chips can be used to elaborate a local self-trigger or it can also be sent to the TCM for global processing. An 8-bit programmable threshold is applied to the multiplicity output of each AGET chip. Starting from firmware version 2.6, a second comparator is applied to the multiplicity output of each AGET chip. The multiplicity signal must be greater than the threshold and less than the limit to fire the corresponding logic. The commands related to multiplicity processing are listed in Table 13.

Table 13. Multiplicity related commands.

Command	Argument	Action
mult_thr	<aget>	Reads the current multiplicity threshold of AGET <aget>
mult_thr	<*|aget> <0xVal>	Sets the multiplicity threshold of one or multiple AGET to the specified value
mult_limit	<aget>	Reads the current multiplicity threshold limit of AGET <aget>
mult_limit	<*|aget> <0xVal>	Sets the multiplicity threshold limit of one or multiple AGET to the specified value
mult_trig_ena		Shows if self-trigger based on multiplicity are enabled or not
mult_trig_ena	<0 | 1>	Disables or enables self-triggers based on multiplicity
snd_mult_ena		Shows if sending multiplicity-over-threshold bits to the TCM is enabled or disabled
snd_mult_ena	<0 | 1>	Disables or enables sending the multiplicity-

	over_threshold bits to the TCM

The command mult_trig_ena is used to check, enable or disable self-triggers based on AGET multiplicity. When active, a local self-trigger is generated whenever one or more of the multiplicity output of the AGET passes the programmed threshold.

The command snd_mult_ena is used to control if the multiplicity-over-threshold bits generated by the Feminos are forwarded to the TCM or not.

6.9 Test Data Pattern Generator

The Feminos card features a test mode and has a fully programmable ADC data pattern memory. The data pattern memory simulates the data-stream of hit channels. The test memory contains 4096 12-bit entries. It can store the data of up to 8 channels for a 512-time bin configuration. The same data pattern is reproduced cyclically when more data are needed. The commands related to the test mode and patterns are listed in Table 14.

Table 14. Test mode and pattern related commands.

Command	Argument	Action
test_enable		Reads the current setting of the TEST_ENABLE bit
test_enable	<0 | 1>	Disable/enable TEST_ENABLE
test_mode		Reads the current value of the TEST_MODE bit
test_mode	<0 | 1>	Clears or sets the TEST_MODE bit
test_zbt		Reads the current setting of the TEST_ZBT bit
test_zbt	<0 | 1>	Disable/enable TEST_ZBT
tdata	<0xAdr>	Reads test data sample at address <0xAdr>
tdata	<0xAdr> <0xData>	Writes <0xData> in pattern memory at address <0xAdr>
tdata	<A, B, C> <0xVal>	Fill test memory with a pre-defined pattern
keep_fco		Reads the state of the KEEP_FCO option
keep_fco	<0 | 1>	Clears/Sets the KEEP_FCO option (drops data of ASIC#3 to keep FCO data when set)

When test mode is disabled, the data fed at the input of the zero-suppression block are the data digitized from the SCAs of the front-end chips. When test mode is enabled, the zero-suppression block is fed with the address counter where SCA data would be read when TEST_MODE is 0 and test data fetched from the pattern memory when TEST_MODE is 1. The pattern memory may be filled with an arbitrary pattern loaded word by word, or can be filled with one of several pre-defined patterns via a single command. Pattern A is a periodic increasing ramp starting from 0 and incremented by one unit until the specified value minus one is reached. Pattern B is similar except that the ramp is decreasing. Pattern C is a single sample wide pulse of

fixed amplitude (2048 ADC counts) with a repetition period specified in the supplied argument. Different pre-defined patterns may be added in the future.

When TEST_ZBT is 0, the source of data for writing to the event buffer memory is the stream of data received from the external ADC. When TEST_ZBT is 1, the 12-LBS's of the write address to the event buffer memory is taken as data. This mode is meant for tests.

The serial outputs of the ADC of the FEC produce a constant 12-bit binary pattern used to delineate channel data samples. For debugging purposes, this framing pattern can be kept in the stream passed to the DAQ. The data of ASIC #3 are dropped when this option is enabled. The constant value that replaces ASIC #3 data is 0xFCO. Pedestal equalization and zero-suppression (if enabled) are still applied to this stream of constant data.

6.10 Dead-time Performance Measurement

The Feminos card measures the digitization and readout time for each event. The measurement is made from the time a valid trigger is received until the Feminos is ready to assert its SCA write signal. In standalone mode with SCA_AUTOSTART = 1, the SCA write signal becomes active as soon as the Feminos is capable of accepting the next trigger. When operating with the TCM, the SCA write signal becomes active only after the TCM has sent the SCA start command when all Feminos in the system have indicated that they are ready to do so. Starting from Feminos server release 2.2, this delay is no longer included in the dead-time measured by the Feminos, i.e. the Feminos now measures its own dead-time rather than global system dead-time. The histogram of global system dead-time is accumulated by the TCM.

Dead-time measurements made by the Feminos are accumulated in a 1024-bin x 18-bit amplitude count histogram (32-bit amplitude in recent firmware versions). Up to firmware version 2.8, bin width is fixed to 100 µs and the range of the histogram of dead-time is [0; 102.2 ms]. Starting from firmware 2.10, four resolutions settings are available (1 µs, 10 µs, 100 µs and 1 ms). These correspond to measurement ranges of [0; 1.022 ms], [0; 10.22 ms], [0; 102.2 ms] and [0; 1.022 s]. When the event readout time overflows the limit, the last bin of the histogram is incremented. Each dead-time measurement is rounded to the closest resolution unit before it is added to the histogram. Saturation occurs when any bin reaches the maximum count (2^18 or 2^32 depending on firmware version). Statistics computed from the dead time histogram are only accurate when the overflow bin is empty and no bin reached saturation. The commands related to dead-time measurements are listed in Table 15.

Table 15. Dead time measurement related commands.

Command	Argument	Action

busy_resol		Gets the resolution of the dead-time histogram
busy_resol	<resol>	Sets the resolution of the dead-time histogram. 0: 1 µs; 1: 10 µs; 2: 100 µs; 3: 1 ms.
hbusy	clr	Clears the histogram of dead time
hbusy	get	Reads the histogram of dead time

6.11 Ring Buffer Interface Control

The transfer of data received by the Feminos hardware to the Ethernet send queue of the embedded processor is controlled by a ring buffer interface. This interface is automatically started at boot time and no further action is normally needed. The ring buffer interface can run with timeout or without. When running without timeout, if no sufficient data are available in the buffer being currently filled by the hardware, the corresponding data stays indefinitely if no further event data pushes it out. When running with timeout, the ring buffer interface will sent any partially filled buffer after a pre-defined amount of time has elapsed without receiving data from the hardware. By default, the ring buffer interface runs in timeout mode with a timeout of 1 s. The commands that control the ring buffer interface are listed in Table 16.

Table 16. Ring Buffer interface control commands.

Command	Argument	Action
rbf	config	Shows the current configuration and state of the ring buffer
rbf	suspend	Suspends operation of the ring buffer
rbf	resume	Resumes operation of the ring buffer
rbf	getpnd	Returns partially filled pending buffer (when running without timeout)
rbf	timed	Shows the current mode of operation (with timeout or without, i.e. infinite wait)
rbf	timed <0|1>	Sets the mode of operation (0: infinite wait; 1: with timeout)
rbf	timeval	Shows the current timeout setting
rbf	timeval <Val>	Sets timeout to <Val> (0: 1ms; 1: 10 ms; 2: 100 ms; 3: 1 s)
rbf	reset	(Expert users only)

The proper sequence to flush any partially filled buffer held by the hardware in infinite wait mode is to suspend the operation of the ring buffer, execute the command to get any pending buffer, read these data and resume operation if desired. When the operation of the ring buffer is suspended while SCA_ENABLE is set to 1 and some trigger is allowed, events will start to pile-up in the Feminos until dead-time is introduced. The hardware and software is designed to make clean transitions and normally guarantees that events never get truncated.

6.12 Data Acquisition Commands

Before data acquisition can take place, the Feminos must be told where event data will be consumed. Specifying the target destination of event data is accomplished with the "serve_target" command. For tests or when the ring buffer needs to be flushed, event data should be dropped. Event data are consumed locally by the Feminos during pedestal runs and during the specific runs made to determine the discriminator thresholds of the AGET chip. In other circumstances, event data should be directed to the DAQ.

The protocol for data acquisition between the Feminos card and the DAQ PC is a pull protocol where the DAQ PC sends requests for data to the Feminos card. This method provides natural flow control which is a mandatory feature to avoid data losses caused by network congestion with unreliable UDP/IP connections. The commands to control data acquisition are listed in listed in Table 17.

Table 17. Data acquisition commands.

Command	Argument	Action
serve_target		Shows if event data are consumed locally (pedestal run mode) or served to the DAQ (normal data taking mode)
serve_target	<0 | 1 |2 |3>	0: drop event data; 1: send event data to the DAQ; 2: use event data locally for pedestal histograms; 3: use event data locally for channel hit histograms
eof_on_eoe		Shows the setting for the option to make the end of each events coincide with an end of frame
eof_on_eoe	<0 | 1>	Sets the option to make a frame end at the end of event.
daq		Shows how many bytes or frames the Feminos is currently allowed to send
daq	<0xSize> <B | F> <seq_nb>	Requests up to <0xSize> bytes or frames of event data from the Feminos card. Optionally add a sequence number to check for frame losses
loss_policy		Shows the current policy when packet losses over the Ethernet link are detected
loss_policy	<0 | 1 |2>	Sets the policy related to packet losses; 0: ignore; 1: re-credit; 2: re-send
cred_wait_time		Shows the current waiting time for credits
cred_wait_time	<time>	Sets the time allowed to receive credits to <time> (ms).

To get event data, the DAQ PC sends series of daq commands that specify how much data the DAQ PC is able to receive in a row. The size argument must be specified in hexadecimal format with exactly six digits, e.g. 0x000001. The unit must be specified, either B or F, for bytes and frames respectively. The allowable size to request depends on the buffering capability of the network interface card of the corresponding PC, that of intermediate network switches and system size. A typical value is 64 KBytes. The Feminos card will send data frames (if available) until the acceptable size is exceeded when the size is specified in bytes. The size limit is therefore always exceeded by at most the size of one frame. When the size is given in frames, the Feminos can send the exact number of frames requested. The DAQ PC need not wait until the amount of data requested is reached to post the next daq command. When receiving a daq command, the Feminos card adds the specified number of credits to the amount of bytes (or frames) it is allowed to send and subtracts to this credit count the size (in bytes or frames) of all the data it sends. To acknowledge the reception of a frame without requesting more data (e.g. when stopping a run), the client DAQ should send a command starting with "daq 0x000000 …". When receiving this command, the Feminos is still allowed to send the amount of data (or number of frames) that corresponds to its current credits. In some other cases, the user will want to stop receiving data from the Feminos card immediately, i.e. without waiting that it has consumed all its remaining credits. A command starting with "daq 0xFFFFFF …" can be used for this purpose. Note that the credits that the Feminos had at this time are lost. For proper operation, the client program has to restore the desired credit count before data acquisition is resumed.

The sequence number is an optional argument used to assist frame loss detection between the Feminos and DAQ PC in both transfer directions. If this functionality is used, the first daq command sent by the DAQ PC to the Feminos should not include any sequence number. This instructs the Feminos to clear locally the value expected for the next daq request. On the second daq request sent by the DAQ PC to the Feminos, the sequence number argument can be added and it must be 0x00. It is incremented by one on every sub-sequent daq command. When reaching 0xFF, the index wraps around and the next daq command will have the sequence number 0x00. Sending a daq command without adding the sequence number argument clears to 0x00 the value expected by the Feminos for the sequence number of the next daq command. When a daq command contains a sequence number, the Feminos compares the received sequence number to the expected value fetched from its own local counter. In case of mismatch, the Feminos determines how many daq command frames were presumably lost. It updates a local error counter but does not attempt to recover lost commands.

Each data frame sent the Feminos to the DAQ PC also carries a sequence number. It is placed in the first 16-bit payload word of a data frame (this word was set to 0x0000 in the first versions of the embedded software). The first data frame sent after receiving a daq command that does not contain the sequence number argument always contains 0x0100 in the first payload word. When receiving this value, the DAQ PC shall set to 0x01 the value it expects for the next response frame from the Feminos. Subsequent data frames sent by the Feminos have in the first 16-bit word a value 0x00 in the 8-MSBs and an incrementing sequence number in the 8-LSBs. When the sequence number reaches 0xFF, it wraps around and the next sequence number will be 0x00. By comparing the expected sequence number to the actual sequence number received, the DAQ PC can determine if some of the frames sent by the Feminos were lost. This operation is only for error monitoring and does not attempt to recover lost data. Note that the sequence number in the DAQ PC to Feminos direction is different from that used in the opposite direction. Each end must maintain a separate counter for both types of sequence numbers. The synchronization procedure is however identical for both counters: it is accomplished by sending to the Feminos a daq command without the sequence number argument.

Because UDP/IP is used for transmission between the Feminos and the DAQ, lost frame are not retransmitted. In practice, credit settings are tuned to avoid network congestion and lower the rate of losses to the rate of physical media errors – which is normally a negligible level. The loss_policy command is used to determine how the Feminos reacts if frame losses are detected. By default, the Feminos ignores frame losses. When the loss_policy is set to "re-credit", the Feminos will increment by one unit its credit frame count when a frame loss is detected. This is only effective when the unit used for credits is frames (F) and not bytes (B). At present, the "re-send" policy is not yet implemented. The Feminos determines that a frame loss has occurred when no credits are returned by the DAQ computer after a programmable time has elapsed since the last frame sent by the Feminos. Parameter tuning is needed to avoid that credits that took longer than the usual time to be returned to the Feminos are thought to be missing.

6.13 Pedestal Histograms, Equalization and Threshold Setting

The Feminos card can be programmed to accumulate the pedestal histograms of each channel during specific runs. When this mode of operation is selected, the data received from the hardware is not transferred to the remote DAQ computer, but is consumed locally instead. After the completion of the pedestal accumulation run, the user can acquire pedestal histograms in various different formats. The user can also instruct the Feminos card to program its internal pedestal equalization table to adjust pedestal after equalization to the desired level. The threshold table can be set to some

value above the equalized pedestal plus the desired number of sigmas above noise. The commands to control pedestal histogram functions are listed in Table 18.

Table 18. Pedestal Histogram commands.

Command	Argument	Action
hped	clr <A> <C>	Clears the pedestal histogram(s) of specified Asic(s) and Channel(s)
hped	offset <A> <C> <off>	Sets the pedestal histogram offset of specified Asic(s) and Channel(s) to <off>
hped	getbins <a> <c>	Gets the full list of non-null bins of pedestal histogram of channel <c> of Asic <a>
hped	getmath <a> <c>	Gets detailed statistics of pedestal histogram of channel <c> of Asic <a>
hped	getsummary <a> <C>	Gets a summary of pedestal histogram statistics for channel(s) of Asic <a>
hped	centermean <A> <C> <M>	Program equalization constants so that pedestal of specified Channel(s) of Asic(s) is <M>
hped	setthr <A> <C> <M> <S>	Program thresholds of specified Channel(s) of Asic(s) to <M> + <S>*_channel_noise
list	ped <A>	Lists the pedestal adjustment constant for all channels of Asic(s) <A>
list	thr <A>	Lists the programmed threshold for all channels of Asic(s) <A>

Pedestal histograms have 512 bins. When working with detectors producing negative signals, the baseline is typically ~250 ADC counts and the recommended range for pedestal histograms is [0; 511]. When detectors producing positive signals are used, the baseline is ~3840 ADC counts and the range of the pedestal histogram should be set to [3584; 4095]. The offset of pedestal histograms should be set to 0 or 3584 according to the polarity of detector signals.

At the end of a pedestal accumulation run, the user must flush any data that may be left in the ring buffer interface to the hardware and enable normal data taking mode with the "serve_local 0" command.

The list of pedestal adjustment constants and threshold settings can be retrieved with the "list" commands. These will be displayed in the client program as a list of interpretable commands so that they can be saved and re-loaded some time later in an easy way.

The arguments <A> and <C> can be scalar, a range of the type <Begin:End>, or the "*" wildcard character to match all possible values. The arguments <a> and <c> specify only one ASIC and channel at a time.

6.14 Hit Probability Histograms, AGET Discriminator Setting

In the AGET mode, the Feminos card can be programmed to accumulate the curves that determine for all possible discriminator threshold values and for each channel the probability that this channel appears as "hit". This threshold scan is done during specific runs. When this mode of operation is selected, the data received from the hardware is not transferred to the remote DAQ computer, but is consumed locally instead. After the completion of the threshold scan, the user can acquire the histograms and can instruct the Feminos card to program the discriminator thresholds of the AGET chip so that the probability that a channel is hit is less than the desired value. The commands to act on the histograms that represent channel hit probability versus discriminator threshold are listed in Table 19.

Table 19. Hit probability versus threshold curve related commands.

Command	Argument	Action
shisto	clr <A> <C>	Clears the hit/threshold histogram(s) of specified AGET(s) and Channel(s)
shisto	getbins <A> <C>	Gets hit/threshold histogram(s) of channel <C> of AGET <A>
shisto	thr * * <0xVal>	Specify that the hit/threshold histogram being accumulated is for threshold <0xVal> (must be in [0x0; 0xF])

At the end of a hit probability accumulation run, the user must flush any data that may be left in the ring buffer interface to the hardware and enable normal data taking mode with the "serve_local 0" command.

6.15 Channel Hit Count Histograms

The Feminos card computes for each event the count of channels hit in the 4 ASICs it controls and accumulates this information in histograms. In AFTER mode, the count of channels hit is constant and equal to 79 for each event (72 physical channels + 4 FPN + 3 reset channels). In AGET mode, it ranges from 2 to 72 depending on the number of channels hits obviously, but also on various settings: 2 or 4 reset channels, read all channels or only hit channels, alterations made at run-time to the content of the hit channel register, etc. The channel hit count per event are accumulated in four 80-bin x 32-bit amplitude count histograms. These histograms are useful for on-line monitoring and system performance evaluation. The commands related to channel hit count histograms are listed in Table 20.

Table 20. Commands related to channel hit count histograms.

Command	Argument	Action
hhit	clr <0xAsic>	Clears the channel hit count histogram of specified ASIC
hhit	get <0xAsic>	Reads the channel hit count histogram of specified ASIC

6.16 Trigger Clock Module Related Commands

The Feminos card can optionally be synchronized and receive triggers from an external Trigger Clock Module (TCM). The commands related to operation with the TCM are listed in Table 21.

Table 21. TCM operation related commands.

Command	Argument	Action
tcm	ignore	Show the state of the TCM_IGNORE bit
tcm	ignore <0 |1>	Enable (0) or disable (1) communication with the TCM
tcm	clr	Clears the local error and message counters
tcm	rx	Gets the count of correct messages received from the TCM since the last clear (modulo 256)
tcm	err	Gets the count of receive errors from the TCM since last clear (saturates at 255)
tcm	tx	Gets the count of messages sent to the TCM since the last clear (modulo 256)
dcbal_enc		Shows if DC balanced encoding is enabled on Feminos to TCM link
dcbal_enc	<0 | 1>	Enables/disables DC balanced encoding on Feminos to TCM link
dcbal_dec		Shows if DC balanced decoding is enabled on TCM to Feminos link
dcbal_dec	<0 | 1>	Enables/ disables DC balanced decoding on TCM to Feminos link
tcm_bert		Shows if the bit error rate tester of the TCM-Feminos link is active or not
tcm_bert	<0 | 1>	Enables/disables the bit error rate tester of the TCM-Feminos link

Communication with the TCM is normally enabled by an on-board switch. Starting with Feminos server version 2.2, it is also possible to disable communication with the TCM by setting the TCM_IGNORE bit. In this case, the Feminos can still receive the reference clock from the TCM – if it is present and enabled by the appropriate

hardware switches – but all trigger information from the TCM will be masked and no status information will be sent by the Feminos to the TCM.

6.17 Miscellaneous Monitoring Commands

The Feminos card has controllers for the OneWire bus and the I2C bus. A silicon identifier chip (Maxim DS2438) is mounted on the FEC and the Mars MX2 module has a secure serial EEPROM and a real time clock. These devices can be readout with the commands listed in Table 22.

Table 22. Slow control monitoring commands.

Command	Argument	Action
fec	<U| I | T | ID>	Reads information provided by the DS2438 chip of the FEC: ID: 64-bit unique serial number; U: measured supply voltage (3.3 V nominal); I: measured supply current; T: measured temperature
mars	info	Gets the serial number and other information on the Mars FPGA module installed on the Feminos
mars	clock	Reads the real-time clock, calendar and temperature from the Mars FPGA module
mars	set day month year hour min sec	Sets the real-time clock and calendar of the Mars FPGA module
mars	power	Reads the current sensor of the Mars FPGA module. Not all modules are equipped with this current sensor and this function is unavailable.

6.18 Flash Memory Read/Write Commands

The Mars MX2 module integrates a 128 Mbit SPI flash memory (Winbond W25Q128BVEIG) used to store the FPGA bitstream, the embedded processor binary code and application specific settings (e.g. MAC and IP addresses, etc. See section on Minibios for details). The content of the flash memory can be read and altered with the commands listed in Table 23.

Table 23. SPI flash memory commands.

Command	Argument	Action
flash	read <0xBase> <0xSize>	Reads <0xSize> bytes (up to 32) from the flash memory starting at address <0xBase>
	write <0xBase> <0xSize> <data>	Writes <0xSize> bytes (up to 256) to flash memory starting at address <0xBase>
	erase_write <0xBase> <0xSize> <data>	Erase sector from <0xBase> (4 KByte) and writes <0xSize> bytes (up to 256) to flash memory starting at address <0xBase>

	write_verify <0xBase> <0xSize> <data>	Same as write with read-back and verification
	erase_write_verify <0xBase> <0xSize> <data>	Same as erase_write with read-back and verification

The data argument is supplied in an ASCII string with two characters per byte in hexadecimal, without space character between bytes. No more than one page, i.e. 256 bytes can be written in a single command. When writing data, the base address must be aligned to a page boundary (256 bytes). Sector erase (4 KByte) is needed when writing the first page of a given sector but must not be done on sub-sequent pages of the same sector to avoid erasing the pages previously written. The read command is provided for debugging only. It is limited to reading out 32 bytes. The start address can be aligned to any byte address.

Flash memory commands are not intended to be used directly by the user. These commands are called internally by the mclient program when uploading a new version of the firmware and embedded software.

7 CLIENT-SIDE PROGRAM COMMAND REFERENCE

The mclient application program is a test client program running on the DAQ PC. This program generally simply forward the commands received from the user or a script file to the Feminos without any alteration, but some commands are specific to mclient and are executed locally.

7.1 GENERAL CONTROL COMMANDS OF MCLIENT

The general control commands of mclient are listed in Table 24.

Table 24. mclient general control commands.

Command	Argument	Action
version		Returns major/minor software version and compilation date
cmd	clr	Clears command sent/replies counters
cmd	stat	Shows command statistics counters
quit		Quits the mclient program (Feminos-side server is kept running)
exit		Similar to command quit
verbose		Shows the level of verboseness of mclient
verbose	<level>	Sets the level of verboseness
vflags	<0xFlags>	Selects the type of debug information printout. Each bit of <0xFlags> corresponds to some particular type of information.
fem		Shows to which Feminos command apply
fem	* <0xFemBitField> <fem_id>	Commands will apply to all Feminos Commands will apply to subset of Feminos Commands will apply only to Feminos of ID fem_id
sleep	<duration>	Sleeps for <duration> seconds
exec	<script_name.txt>	Execute the command script <script_name.txt>
LOOP	<nb_iterations>	Repeat the block of commands until NEXT <nb_iteration times>
LOOP	<begin> TO <end>	Similar to LOOP except that the boundaries of the index of the loop are specified
NEXT		Sets the end of the block of commands to repeat
END		Recommended at the end of a script file
	$loop	When this special string is found in some few specific commands, it is replaced on the fly by the current value of the index of the loop.
program flash	<feminos_date.mcs>	Upgrade the firmware and embedded software with the supplied bitstream file
mars	set clock	Fetches the current local date and time of the host and initialize the real time clock of the target FPGA module correspondingly.

The Feminos program server and mclient program maintain the count of commands sent/received; the number of "daq" requests and replies are counted separately. The command "cmd clr" should be sent when both the server and client side programs have been restarted, and when DAQ is inactive, to allow the direct comparison of counters at each end. The commands to clear and to get the statistics are also counted. Hence, if the user clears message count and get message statistics, the number of command message sent/received at each end will be 2.

The mclient program can execute scripts contained in text files, but, at present, scripts cannot be nested, i.e. the exec command cannot be called within a script. The LOOP and NEXT commands are only available within a script and, at present, multiple loops cannot be nested. The current index value of a loop can be used as an argument in a few commands (examples: "aget <A> threshold <C> $loop" and "shisto thr * * $loop"). This is useful to perform the threshold scan for acquiring the sensitivity curve of the channel discriminators of AGET chips.

The command "program flash" is used to load a new firmware and embedded software release in the SPI flash memory of the Mars MX2 module. The upload takes about two minutes. The board must be power-cycled to boot the new firmware and software. The supplied file must be a complete valid bitstream that includes the FPGA configuration, bootloader code and application code for the embedded processor. The file naming convention specifies the release date with two digits for the day, three characters for the month and digits for the year. If the upgrade fails, the Feminos card will no longer configure itself properly at power-up and re-programming with a JTAG probe will be needed.

7.2 Data Acquisition Commands

The commands that control data acquisition at the level of the mclient program are listed in Table 25.

Table 25. mclient-side data acquisition commands.

Command	Argument	Action
DAQ		Shows how much data still need to be collected to complete the current DAQ request
DAQ	<size>	User level command to request a large amount of data at once (<size> is a 64-bit integer and is expressed in bytes)
DAQ	0	Stop the current data acquisition
credits	show	Shows how many data bytes mclient can request to each Feminos
credits	restore	Restore the initial credit count of all Feminos in

		mclient with default values
credits	restore <max> <thr> <unit>	Sets the credit count of all Feminos in mclient to <max>, requests are sent to Feminos when <thr> is reached. The unit for credits may be bytes (B) of frames (F).

The "DAQ" family of commands is used at the level of the human user of the system or within a script. The size argument specifies the total amount of data (in bytes) to collect from all active Feminos cards. This amount of data requested can be up to a several tens of TBytes. The mclient command interpreter automatically post series of "daq" commands to the different Feminos cards until the size specified by the "DAQ" command is globally reached (and is slightly exceeded).

The "credits" commands are used to diagnose and restore operation after some error communications between the Feminos and mclient. If mclient has posted to a Feminos all its credits and these have been lost for any reason, a potential dead-lock can occur: the mclient cannot request data from the Feminos because it does not have any credits but the Feminos cannot send data for the same reason. Restoring credits at one end can solve this problem. Credits can also be adjusted to optimize transfers depending in different setups. System throughput can be increased up to some saturation level when more credits circulate in the system. But buffer overflow and communication errors will occur if some element in the chain cannot absorb the traffic that results.

7.3 File I/O Commands

The command interpreter (mclient) running on the DAQ PC offers to the end-user a minimal set of commands to save the data received from Feminos cards to files. These commands are listed in Table 26.

Table 26. File I/O commands.

Command	Argument	Action
path		Shows the path for saving result files
path	<path_str>	Sets the path for saving result files to <path_str>
file_chunk		Shows the current maximum file size (1 GByte by default)
file_chunk	<size>	Sets the maximum file size to <size> (in Mbytes)
fopen		Create a file to store event data frames in binary
fopen	asc	Create a file to store event data frames in ASCII
fclose		Close the currently opened data file
LIST	ped <A>	Lists the pedestal equalization constants of all the channels of Asic <A> and save them in an interpretable script file
LIST	thr <A>	Lists the programmed threshold of all the

	channels of Asic <A> and save them in an interpretable script file
event_builder	Shows the mode of operation of the event builder
event_builder <0xMode>	Sets the mode of operation of the event builder

The path command is used to define the root directory where data are saved.

When the user enters the "fopen" command, the system automatically creates a file named as follows: "<path_str>/Ryyyy_mm_dd-hh-mm-ss_bbb.xxx", where <path_str> is the relative or absolute path set by the "path" command, yyyy is the current year, mm is the month, dd is the day, hh-mm-ss is the current time, bbb is the file number within this run and xxx is "aqs" if the selected format is binary and "txt" if it is ASCII.

For basic manual data acquisition (after system configuration), the user should first do "fopen", then type "DAQ <size>" with the amount of data he would like to collect for this run. The user should then manually issue "DAQ" commands until the system indicates that there are no more data to collect (this step will be automated). If the amount of data requested exceeds the maximum file size (defined by the "file_chunk" command), mclient will automatically close the current file and create sub-sequent files as needed. The last file is closed by the user with the "fclose" command.

The format of the data files that are recorded by the mclient program on the DAQ PC uses the same encoding rules that are used for the communication between the Feminos card and the DAQ PC. The data that is found in the file is however not the exact copy of the messages exchanged between the Feminos cards and the DAQ PC. Notably, responses to configuration commands and frames that contain monitoring information are skipped and only event data frames are recorded. Some unnecessary words (e.g. the empty word at the beginning of each frame) are also skipped. At the beginning of a file, the run date and time is encoded in ASCII format (with the required prefix and encoding rule given in the next section).

The "LIST" commands are used typically used after a pedestal run when the pedestal equalization constants and thresholds for zero-suppressed readout have been set and the user want to save them to be able to re-load the same values at a later time. Pedestal and threshold settings are saved in a text file called: "ptyyyy_mm_dd-hh-mm-ss.txt", where yyyy is the current year, mm is the month, dd is the day, hh-mm-ss is the current time. The "exec" command following by the pedestal or threshold file name is used to re-load the corresponding values.

The mclient program contains an experimental event builder intended for operation with multiple Feminos. The event builder can be transparent (bit 0 of Mode = 0) or

active (Bit 0 of Mode = 1). In transparent mode, the data received from the different Feminos are stored in a common file in arrival order. In the active mode, the event builder searches for event boundaries in the data received from each Feminos and groups the data of each event before they are stored to disk. At present, the event builder is very simplistic and is mostly intended for demonstration. For proper operation, it requires that the Feminos makes the end of event always appear at an end of frame. When the event builder is active, the prefix "PFX_START_OF_BUILT_EVENT" and "PFX_END_OF_BUILT_EVENT" are added to wrap the data gathered from all the Feminos for each event. The event builder can optionally check that event numbers and timestamps are identical across all Feminos. Checking event numbers and timestamps are enabled independently by setting bit 1 and bit 2 of Mode to 1 respectively. If timestamp checking is enabled, the event builder checks that all timestamps match exactly. In order to tolerate slight synchronization errors, the event builder can be set to accept events with timestamps that differ from +1 or -1 unit. This policy is enabled by setting bit 3 of event builder Mode to 1.

Although the mclient program can be useful for basic tests, data taking in real experimental conditions with one or multiple Feminos requires a more complete and robust DAQ software package. The event builder used for Minos DAQ performs the additional following tasks: packet header and trailer stripping, packet ordering following Feminos index, detection of missing fragments, total event size calculation, etc. Data file format with the event builder of Minos DAQ uses the same method of encoding but the following differences can be noted:

- The run string at the beginning of the file is suppressed.
- At the beginning of each event, the prefix PFX_START_OF_BUILT_EVENT is replaced by PFX_SOBE_SIZE (Start of Built Event with Size) followed by the total size of the event coded with 32-bits (16-LSB first).
- The frame boundary markers PFX_START_OF_DFRAME (followed by frame size) and PFX_END_OF_FRAME are suppressed.
- The event delimiter PFX_START_OF_EVENT followed by the event number and time-stamp appears only once for each event rather than for each Feminos. Cross verification of time-stamps and event numbers is done by the event builder.
- The event delimiter PFX_END_OF_EVENT is no longer used because it only allows 20-bit for event size (i.e. 1 MB) which is adequate for one Feminos, but is insufficient for larger systems.

Refer to the documentation of Minos DAQ for details and possible other changes.

8 FEMINOS I/O DATA FORMAT

The Feminos card communicates with the DAQ PC over a standard UDP/IP socket interface. Commands sent from the DAQ PC to the Feminos card are encoded in plain ASCII text according to the syntax described in section 6. Response frames are encoded in ASCII or binary format as described in this section.

Contrary to the Internet convention, data sent by the Feminos card are always sent in Little-Endian format. This avoids an un-necessary byte swap at both ends because the application running on the MicroBlaze processor needs to be compiled in Little-Endian format (for compatibility with AXI-4 protocol) and the majority of modern PCs use Intel processors which are also Little-Endian devices.

The Feminos card supports standard Ethernet frame length (up to ~1500 bytes) for 10/100/1000 Mbps speed and Jumbo frames up to 8 KBytes in Gigabit Ethernet mode.

8.1 PREFIX-CODE FORMAT

The basic information datum is a 16-bit short word. For coding compactness, the Feminos card uses a prefix-code where a variable fraction of the 16-bits of an information datum is used to define the type of data being encoded and the remaining bits (followed by one or several 16-bit words if needed) are used to encode the data itself. In order to decode data sent by the Feminos card, application software needs to identify the prefix of each data element. The scan should be done starting from the shortest prefix then trying longer ones until a match is found. Once the prefix has been identified, the size of the data element is determined implicitly or can be retrieved from the data itself. If decoding software does not find a matching prefix in a data item, this is a serious error that indicates data transmission corruption, software bugs, non-synchronized protocol versions, or similar problems that need to be resolved for proper operation. The list of available prefix is listed in Table 27.

Table 27. List of prefix.

Prefix	Name	Function and Data Field
2-bit prefix	***14-bit data field***	
11xxxxxyyzzzzzzz	PFX_CARD_CHIP_CHAN_HIT_IX	Specify that sub-sequent data belong to channel <zzzzzz> of chip <yy> of card <xxxxx>
10xxxxxyyzzzzzzz	PFX_CARD_CHIP_CHAN_HIT_CNT	Indicates that chip <yy> of card <xxxxx> had a total of <zzzzzzz> channel hit for this event
01xxxxxyyzzzzzzz	PFX_CARD_CHIP_CHAN_HISTO	Specify that sub-sequent data is pedestal histrogram of channel <zzzzzz> of chip <yy> of card <xxxxx>
4-bit prefix	***12-bit data field***	
0011xxxxxxxxxxxx	PFX_ADC_SAMPLE	Encodes a 12-bit ADC sample value
0010xxxxxxxxxxxx	PFX_LAT_HISTO_BIN	12-bit count of the current bin of the dead-time latency histogram

0001xxyyyyyyyyyy	PFX_CHIP_LAST_CELL_READ	The last cell read pointer of ASIC <xx> is <yyyyyyyyyy> (only the 9-MSBs are used)
7-bit prefix	***9-bit data field***	
0000111xxxxxxxxx	PFX_TIME_BIN_IX	The ADC samples that follow start at time-bin <xxxxxxxxx>
0000110xxxxxxxxx	PFX_HISTO_BIN_IX	Specifies the bin index in the current histogram
0000101xxxxxxxxx	PFX_PEDTHR_LIST	List of pedestals or thresholds
0000100xxxxyyyyy	PFX_START_OF_DFRAME	Start of data frame. Encoding version is <xxxx> FEM index is <yyyyy>
0000011xxxxyyyyy	PFX_START_OF_MFRAME	Start of frame with monitoring information. Encoding version is <xxxx> FEM index is <yyyyy>
0000010xxxxyyyyy	PFX_START_OF_CFRAME	Start of configuration reply frame. Encoding version is <xxxx> FEM index is <yyyyy>
0000001xxxxxxxxx	unspecified	Available for future use
8-bit prefix	***8-bit data field***	
00000001xxxxxxxx	PFX_ASCII_MSG_LEN	Specifies that the length of the ASCII string that follows is <xxxxxxxx> bytes.
12-bit prefix	***4-bit data field***	
000000001111xxxx	PFX_START_OF_EVENT	Start of event. Even type is <yyy>. 48-bit event time-stamp and 32-bit event count follow
000000001110xxxx	PFX_END_OF_EVENT	End of Event. MSB of 20-bit event size is <xxxx>, 16-LSB of event size follow
000000001101xxxx	unspecified	Available for future use
000000001100xxxx	unspecified	Available for future use
000000001011xxxx	unspecified	Available for future use
000000001010xxxx	unspecified	Available for future use
000000001001xxxx	unspecified	Available for future use
000000001000xxxx	unspecified	Available for future use
14-bit prefix	***2-bit data field***	
00000000011111xx	PFX_CH_HIT_CNT_HISTO	Channel hit count histogram. ASIC index is <xx>
0000000001111110	Unspecified	Available for future use
…	…	…
00000000010000xx	unspecified	Available for future use
15-bit prefix	***1-bit data field***	
000000000011111x	unspecified	Available for future use
…	…	…
000000000001000x	unspecified	Available for future use
16-bit prefix	***Implicit data***	
0000000000001111	PFX_END_OF_FRAME	End of frame indicator
0000000000001110	PFX_DEADTIME_HSTAT_BINS	Dead time statistics and histogram follow
0000000000001101	PFX_PEDESTAL_HSTAT	Pedestal histogram full statistics follow
0000000000001100	PFX_PEDESTAL_H_MD	Pedestal histogram short statistics follow
0000000000001011	PFX_SHISTO_BINS	Channel hit probability versus threshold histogram follow
0000000000001010	PFX_CMD_STATISTICS	Command statistics counter follow
0000000000001001	PFX_START_OF_BUILT_EVENT	Event start boundary when event builder active
0000000000001000	PFX_END_OF_BUILT_EVENT	Event end boundary when event builder active
0000000000000111	PFX_EVPERIOD_HSTAT_BINS	Inter-event time statistics and histogram follow
0000000000000110	PFX_SOBE_SIZE	Start Of Built Event with Size – only used with final event builder
…	…	…
0000000000000001	unspecified	Available for future use
0000000000000000	PFX_NULL_CONTENT	Null word to be skipped

The Feminos card tries to optimally fill Ethernet frames with data but it guarantees that the data samples of one channel (up to ~512-time bins) never spans across two frames.

The Feminos card sends frames that can be classified in three different groups:

- Frames that contain the reply to a configuration command. These contain an error status code and a string message in ASCII format.
- Frames that contain event data for the DAQ. These are encoded in binary format.
- Frames that contain monitoring information (e.g. pedestal histograms, latency measurements, etc). These are also encoded in binary format.

Frame classification is done by scanning the first 16-bit word of the frame.

8.2 Frame Encoding for Configuration Command Replies

The format of a frame sent in response to a configuration command is shown in Fig. 20.

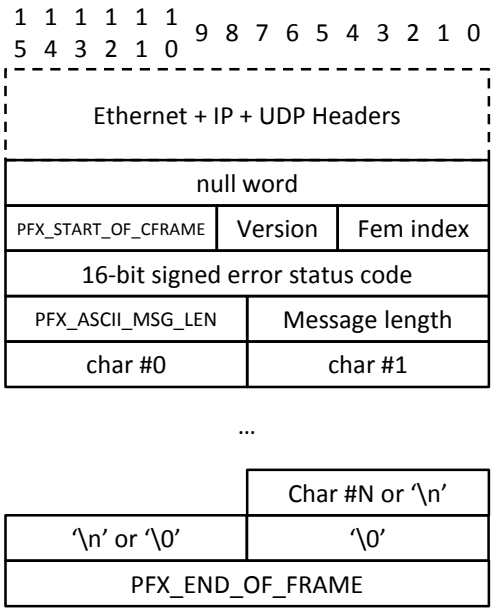


Fig. 20. Reply frame to a configuration command.

A null word is always present after the UDP header. It is followed by the start of configuration frame prefix, the protocol encoding version and the index of the Feminos. The following word is 16-bit signed error code. A negative value indicates that the command failed. A null or positive value indicates that the command completed successfully. The error code is followed by an 8-bit prefix that indicates that subsequent data is an ASCII string. The message length is given in the byte that follows

the prefix. The length includes the size of the trailing carriage return (if any). A null terminating character is always appended. If the length of the encoded string is even, two null characters are added so that message length is always an even number of bytes.

8.3 Data Frame Encoding

The format of a typical frame containing event data is shown in Fig. 21.

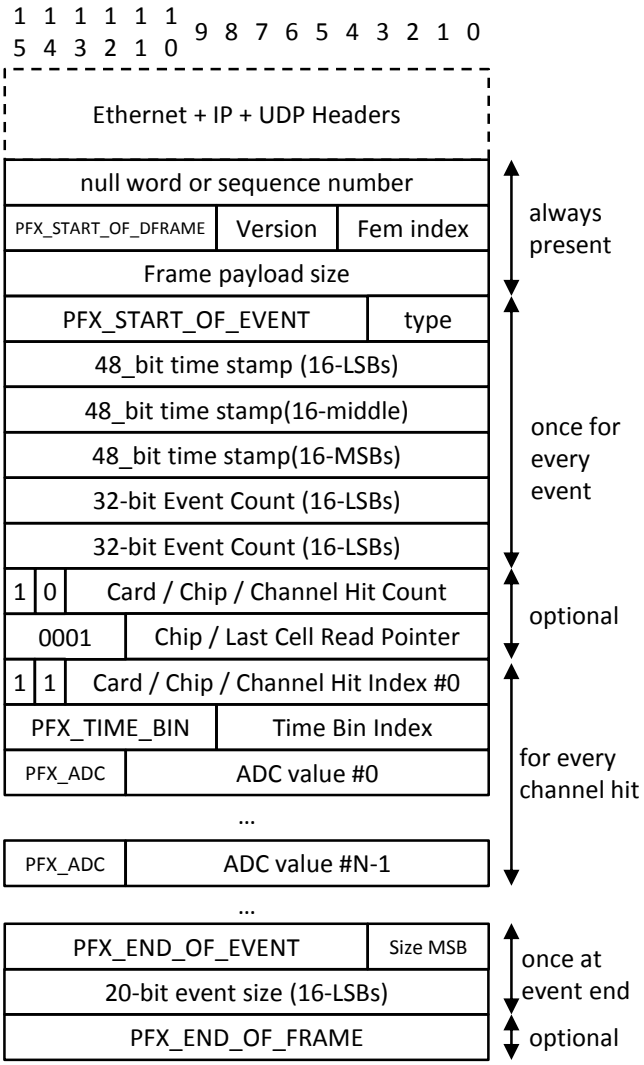


Fig. 21. Event data frame.

The initial null word, the word that contains a START_OF_DFRAME prefix followed by a 4-bit protocol version number and the Feminos index, and the payload size word are always present. In the recent versions of the firmware, the initial null word is replaced by a sequence number to check for frame losses. See the section that describes the daq command for details. If it is not processed, this word should be ignored. The payload size is the size in bytes counted from the START_OF_DFRAME prefix to END_OF_FRAME (including both of them). This size is always even. The

START_OF_EVENT prefix and the sub-sequent 4-bit event type, 48-bit time stamp and 32-bit event count appear only once for each event. The channel hit count (7-bit field) for each of the 4 chips (2-bit field) of the current Feminos card (5-bit field) is only sent when this option is enabled. This is followed by the identification of the first channel being hit, the index of the first time-bin above threshold for that channel, and the list of ADC values for successive time-bins until the threshold was no longer exceeded (precursor samples and trailer samples are also present – see section on zero-suppression for details). If the number of ADC samples for a given channel is even, a null word is added. The same message pattern is reproduced for all the channels hit of all the chips readout by the current Feminos card. An END_OF_FRAME indicator is added at the end of the frame. When no more data are available for the current event, an END_OF_EVENT indicator is emitted. The total event size in bytes is encoded in the 20 bits that follow the corresponding prefix. The maximum event size that can be coded is 1 MB while the largest possible event sent by a Feminos card is ~4 chips x 79 channels x 520 bins x 2 bytes = 320 Kbyte (without encoding overhead). Typical events spans across multiple frames. Empty events occupy one frame and contain at least the event type, time-stamp, event count and end of event indicator. An event may be empty because none of the ASIC channels were hit, or because none of the waveforms were retained after zero-suppression.

8.4 Encoding of Frames for Monitoring Information Data

The Feminos card can accumulate the pedestal histogram of each channel during specific runs. These histograms can be sent to the DAQ PC in a condensed or detailed format. The most detailed information is the list of all the pedestal histogram bins that contain a non-null count. The format of this frame is shown in Fig. 22.

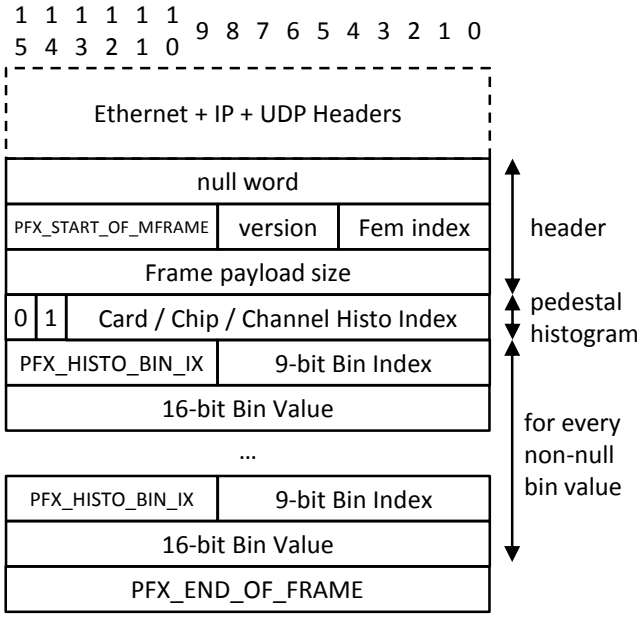


Fig. 22. Pedestal Histogram in non-null bin list format.

In this format, one frame contains the pedestal histogram of only one channel. The list of bins may be truncated if the histogram does not fit in one Ethernet frame (the current limit is fixed at ~1400 bytes because the corresponding section of the current code does not use Gigabit Ethernet Jumbo frames). The list of bins may also be empty. The frame payload size is the size in bytes counted from the START_OF_MFRAME prefix to END_OF_FRAME (including both of them).

A pedestal histogram can also be sent in a more condensed format as shown in Fig. 23.

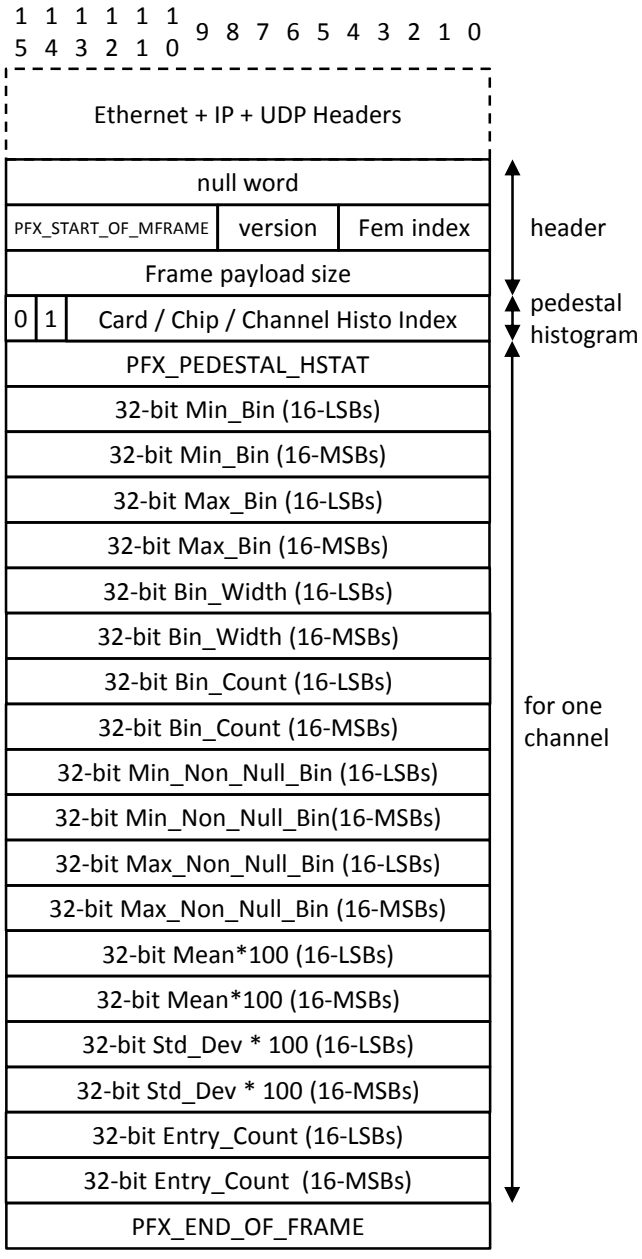


Fig. 23. Pedestal histogram detailed statistics frame.

Pedestals histograms statistics can be sent in a shorter version that only contains the mean value and standard deviation. In this format, a frame can contain the pedestal mean/deviation of several channels. This is shown in Fig. 24. The frame payload size is the size in bytes counted from the START_OF_MFRAME prefix to END_OF_FRAME (including both of them).

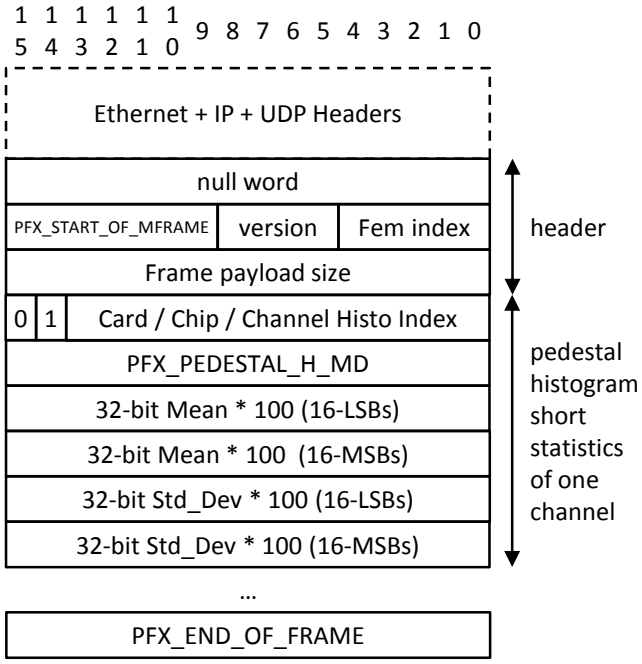


Fig. 24. Pedestal histogram condensed statistics frame.

The Feminos card activates its BUSY pin when the front-end SCAs are being readout and releases this pin when the SCAs are ready to capture the next event. The duration of the BUSY signal is representative of the dead-time of the readout system. The dead-time measured for each event is accumulated in a 1024-bin, 32-bit amplitude count histogram (18-bit in old firmware versions). Until firmware version 2.8, resolution is fixed to 100 µs, i.e. the range of the dead-time histogram is [0, 102.2 ms]. Starting from firmware version 2.10, resolution is programmable among four values: 1 µs, 10 µs, 100 µs and 1 ms. The measurement range scales accordingly. For all resolution settings, the last bin (bin #1023) accumulates overflows. The frame payload size is the size in bytes counted from the START_OF_MFRAME prefix to END_OF_FRAME (including both of them).

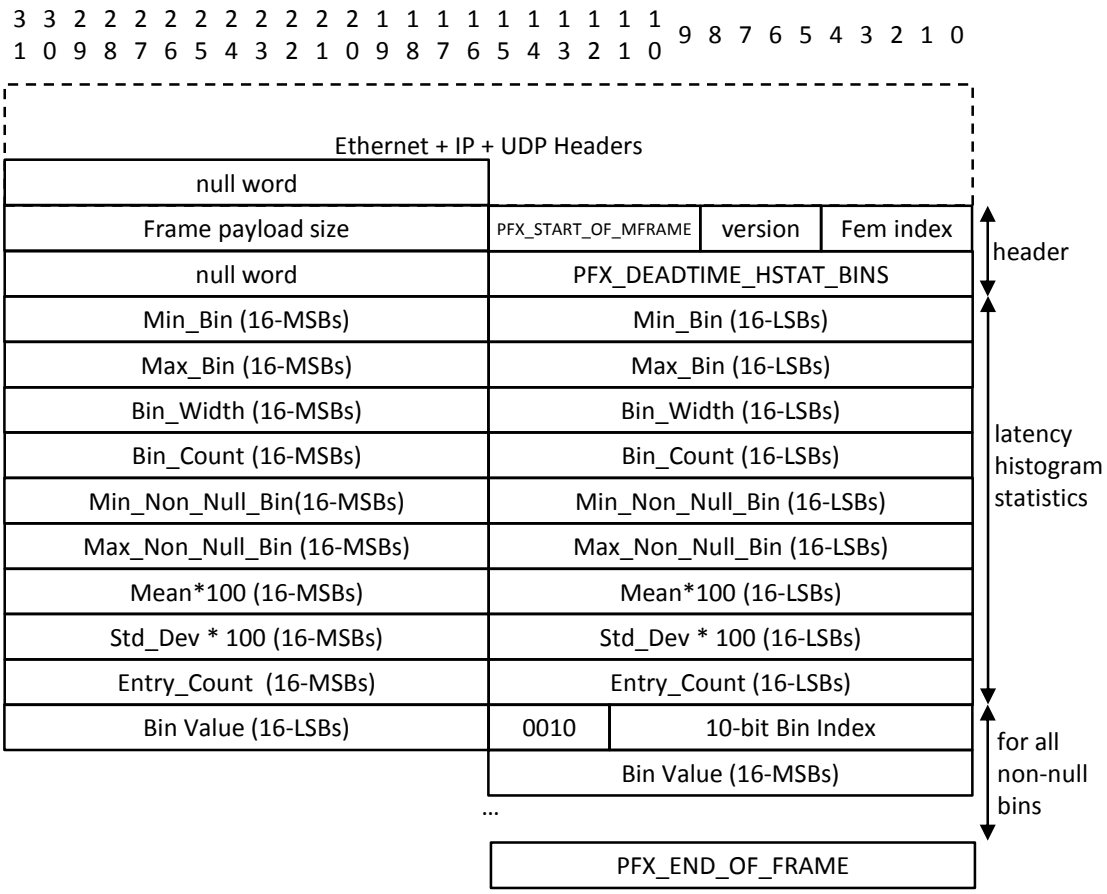


Fig. 25. Dead-time histogram frame.

The Feminos card accumulates the number of configuration and monitoring command received, number of errors (syntax error in the command or execution failure) and number of command replies. Also, the number of "daq" requests received and served is counted separately. The structure of the monitoring frame that contains the commands statistics counters is show in Fig. 26. The frame payload size is the size in bytes counted from the START_OF_MFRAME prefix to END_OF_FRAME (including both of them).

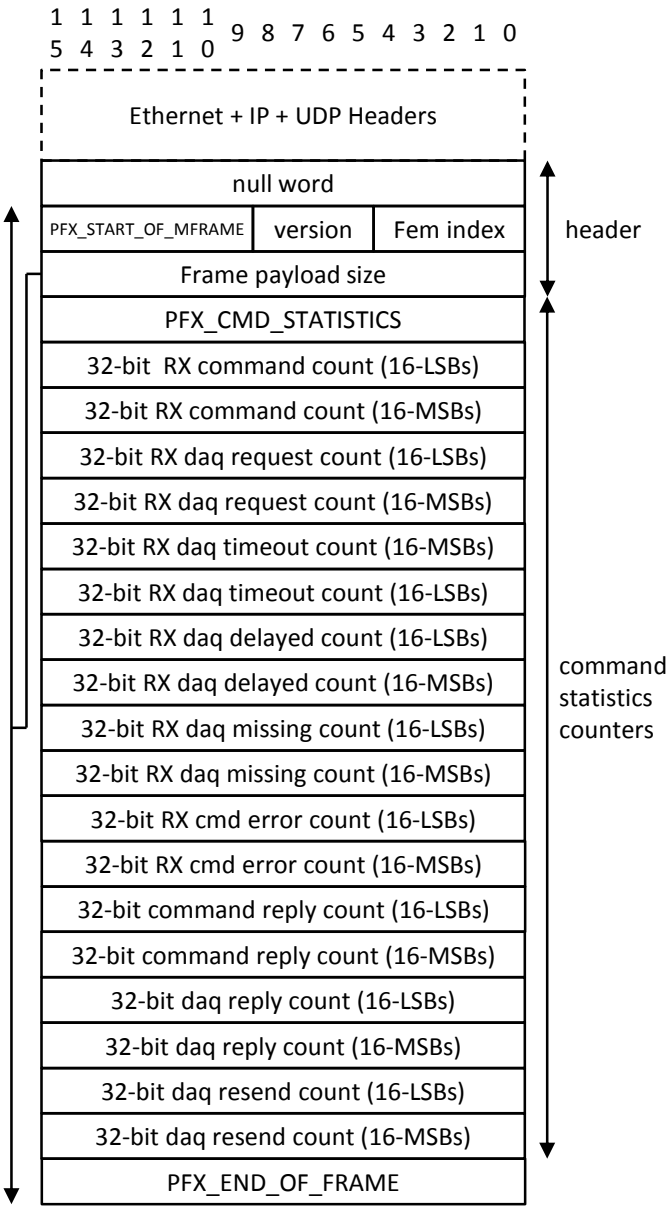


Fig. 26. Command statistics counter frame.

The Feminos card accumulates for each of the 4 ASICs it controls the histogram of channel hit count per event. The structure of the monitoring frame that contains one channel hit count histogram is show in Fig. 27. The frame payload size is the size in bytes counted from the START_OF_MFRAME prefix to END_OF_FRAME (including both of them).

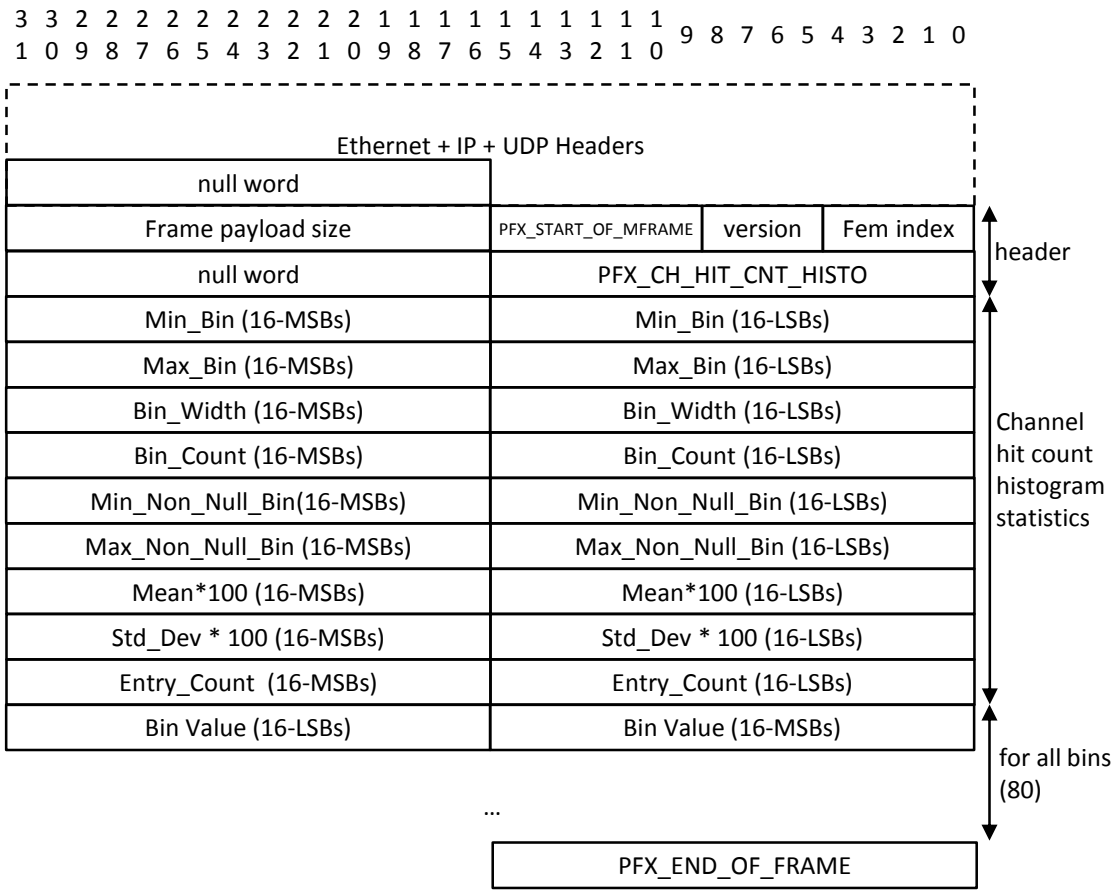


Fig. 27. Channel hit count histogram frame

The Feminos card can send to the DAQ PC the list of pedestal equalization constants and thresholds that it computed. The format of the corresponding frame is given in Fig. 28.

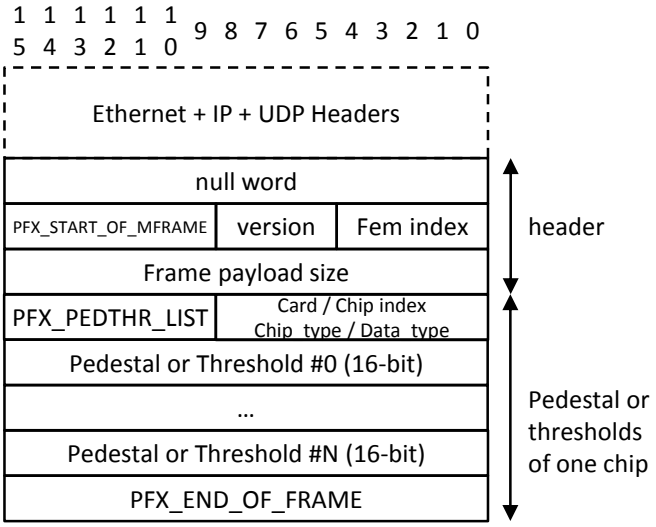


Fig. 28. Pedestal or threshold list.

After the prefix PFX_PEDTHR_LIST, the 9-LSBs specify: the index of the Feminos card (5 bits), the index of the chip on that card (2 bits), the type of chip (1 bit: 0 for AGET and 1 for AFTER), the type of data in the list (1 bit: 0 for pedestals and 1 for thresholds). It is followed by a fixed number of 16-bit values, 72 in the AGET mode and 79 in the AFTER mode, where each value represents a pedestal equalization constant or threshold. Pedestal equalization constants are signed short integers while thresholds are unsigned short integers.

In the AGET mode, the Feminos can determine by a specific run the sensitivity curve of the programmable discriminator threshold of each channel. For each of the 16 possible values of the fine threshold setting of each channel, the Feminos accumulates in a histogram the number of times the channel was hit. The format of the frame that contains the threshold transfer functions is given in Fig. 29.

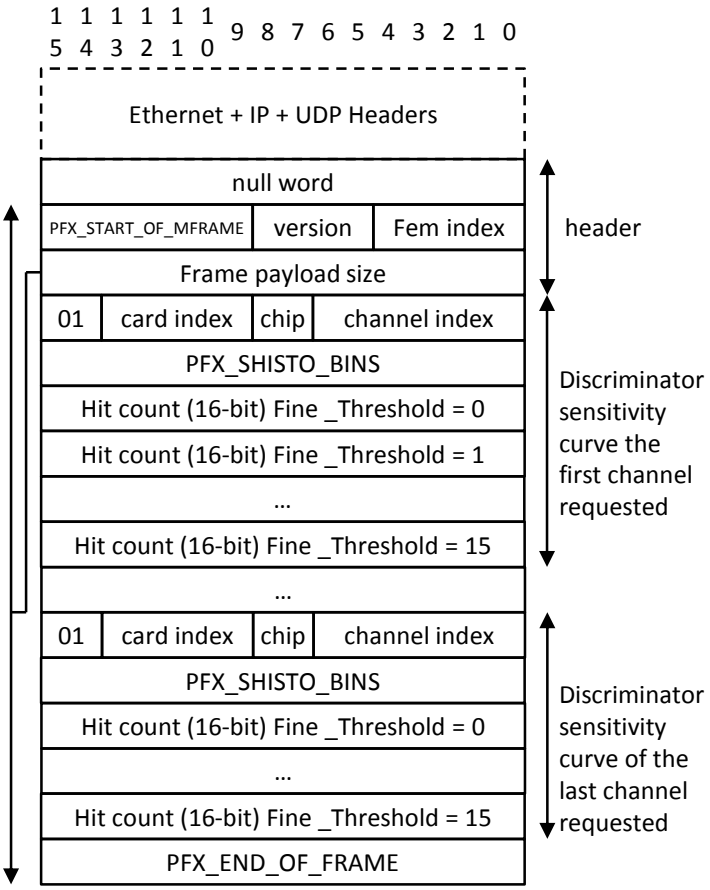


Fig. 29. Format of the discriminator sensitivity curve frame.

The frame payload size is the size in bytes counted from the START_OF_MFRAME prefix to END_OF_FRAME (including both of them).

9 COMMON TASKS

9.1 PEDESTAL EQUALIZATION AND ZERO-SUPPRESSION

The Feminos card can optionally subtract a constant value to each channel. The purpose of this operation is to compensate for the physical mismatch in the silicon between the baseline level of the different channels of the AFTER or AGET chips. The per channel equalization constant of each channel should be computed to obtain the same baseline level for all channels after equalization. The value of the post-equalization baseline can be chosen by the user. The recommended value is typically 250 ADC counts (i.e. zero on the energy scale corresponds to 250 ADC counts). A non-null baseline offset avoids truncation of small signal undershoots and errors that could be caused by baseline drifts with temperature. It marginally reduces the available dynamic range.

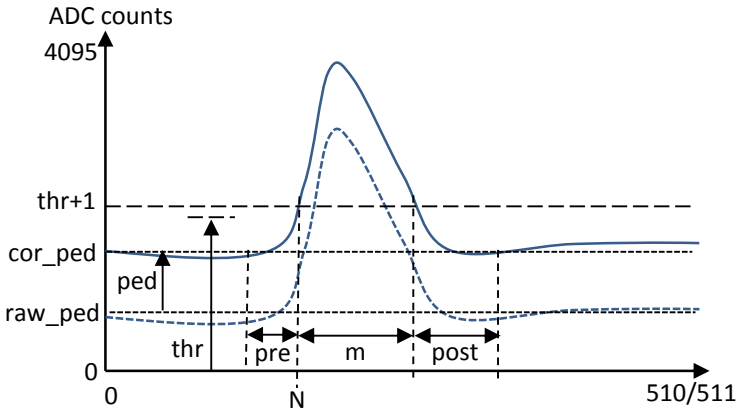


Fig. 30. Pedestal equalization and zero-suppression.

The pedestal subtraction (or "equalization") and zero-suppression phases are exemplified on the diagram of Fig. 30. The raw waveform of a given channel is the dashed pulse. The average baseline level is <raw_ped>. When pedestal correction is enabled, the constant <ped> is added, leading to an average corrected pedestal equal to <cor_ped>. The user will typically want that all <cor_ped> values are identical for all channels and equals 250 ADC counts for example. The set of 256 pedestal equalization constants (288 in the AFTER mode) is determined by acquiring noise events and making computations with an external program, or these can be determined by the Feminos using the specific procedure described in the next section of this document. The pulse after pedestal equalization is the solid line.

When zero-suppression is disabled, the data retained for each channel is the series of all SCA samples acquired (programmable up to 511 in the AFTER mode and 512 in the AGET mode). When zero-suppression is enabled, the data that are kept are the set of m samples strictly above the threshold of the channel considered and a

programmable number of pre-samples and post-samples. The Feminos indicates the index N of the first sample that is strictly above threshold. The pre- and post-samples are programmable in [0, 15] and [0, 16] respectively. The Feminos card always prepends the series of m ADC samples with the desired number of pre-samples, even if a fraction or all pre-samples correspond to negative time-bins indexes. A zero ADC count value is sent for these pre-samples. On the other hand, the Feminos card never emits ADC samples that would correspond to time-bin indexes greater than 511 or 512 (AFTER or AGET mode). The number of post-samples may therefore be less than the programmed number.

The number of ADC samples emitted for a channel waveform is at least one (only one sample strictly above threshold, no pre- or post-samples) and 526 or 527 at most (all samples above threshold in AFTER or AGET mode and 15 pre-samples). Note also that several pulses may be found in a waveform. If the post-sample region of the first pulse overlaps with the pre-sample region of the second pulse, the Feminos card only indicates the position N of the first pulse. If the two pulses are sufficiently apart, the post-amble of the first pulse is completely distinct from the pre-amble of the second pulse and the Feminos card will make the distinction between the two pulses.

It has been observed that noise is higher on the first time bin data. This creates additional event data that does not have real value. Starting from firmware dated 05 March 2013, the zero-suppressor skips a certain number of time-bins before it starts making comparisons with the programmed threshold. The number of time-bins skipped is coded in the firmware and cannot be changed at run-time. At present, only the first time bin is excluded from the comparison to the threshold. The setting may be changed in the future if this is needed.

It has also been observed – especially with positive polarity signals – that the last time-bins can also cause false detection. Starting from firmware version 2.2 dated 25 March 2014, the zero-suppressor does not perform any comparison on the last four time-bins.

Up to firmware 2.11, pedestal histograms and the related computations of mean and rms were made using all the available time-bins of each channel. When working at low sampling rate, it has been found that the first or second time bin can contain very high or low values that significantly deviate from the normal baseline. Starting from firmware 2.12 and beyond, the first 4 time-bins of all channels are no longer included in the histograms and calculations of the mean and rms of the pedestals.

9.2 Setting Pedestal Equalization Constants and Thresholds

The procedure consists in acquiring noise events to accumulate pedestals histograms, then computing for each channel the appropriate constant to bring the

mean pedestal to a pre-defined value. The equalization constants are then set into the hardware and a second run with noise events is made to control the effectiveness of the procedure. Thresholds are set according to the equalized pedestal above the noise measured for each channel. All these steps are performed automatically by running the appropriate command scripts on the Feminos and will not be further detailed.

We show on Fig. 31 and Fig. 32 the typical pedestal mean and RMS map of an AGET chip (gain: 120 fC; shaping time: 540 ns; write frequency: 100 MHz) without connection to a detector. The four FPN channels are those with a lower RMS value. The pedestal equalization does not affect the RMS of pedestals.

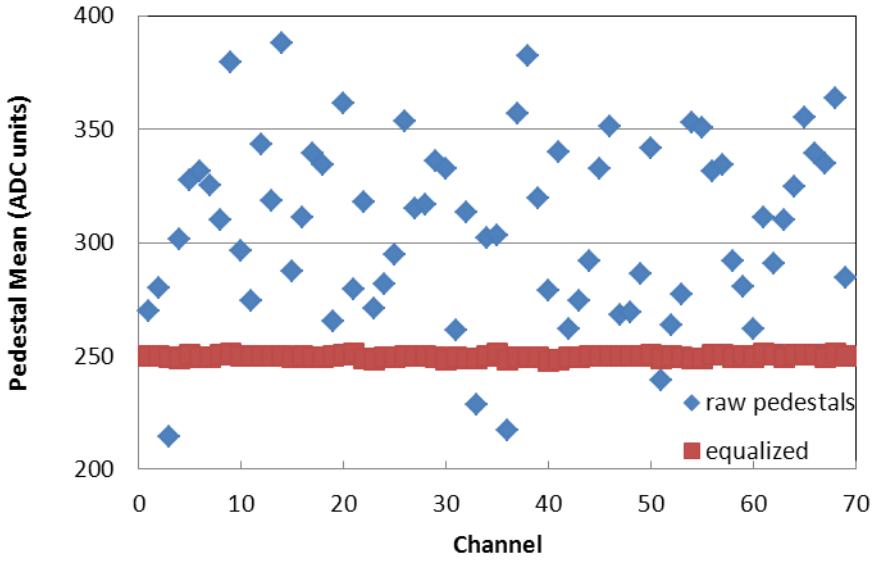


Fig. 31. Mean pedestal of an AGET before and after equalization.

When the AGET is set for a short read sequence, the FPN channels are #13, #24, #47 and #58. When using the normal length read sequence, FPN channels appear at channel index #15, #26, #49 and #60. On the AFTER chip, the four FPN channels appear at channel index #15, #28, #53 and #66.

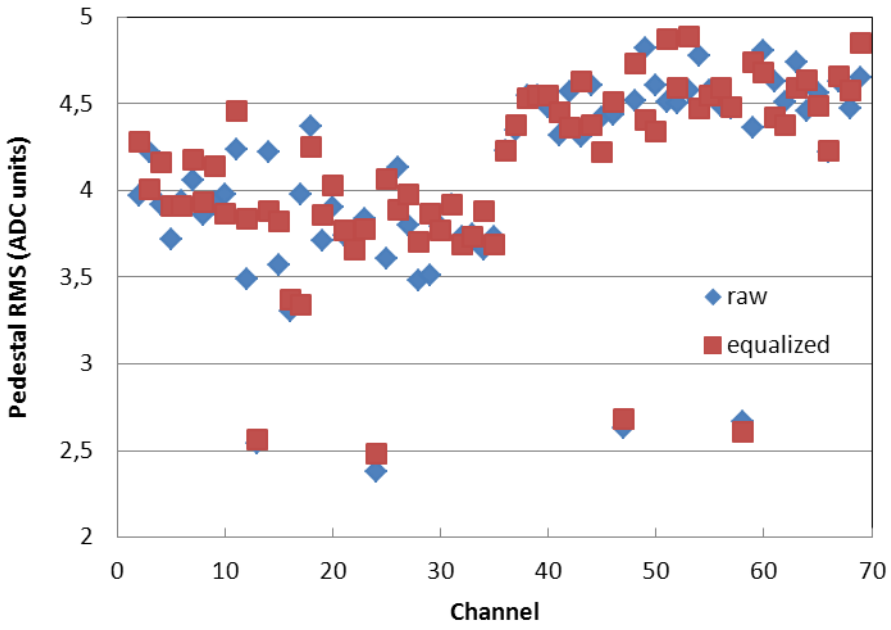


Fig. 32. Pedestal RMS of an AGET before and after equalization.

The previous example applies to detectors that deliver negative signals. When detectors that deliver positive signals are used, the mean pedestal level is shifted towards the end of the ADC range and the amplitude of pulses is reverted. In this case, the equalized mean pedestal is typically set to 3850 and the zero-suppression stage keeps data that are below the threshold rather than above. A typical pedestal map of an AGET chip programmed for positive detector signals is shown in Fig. 33.

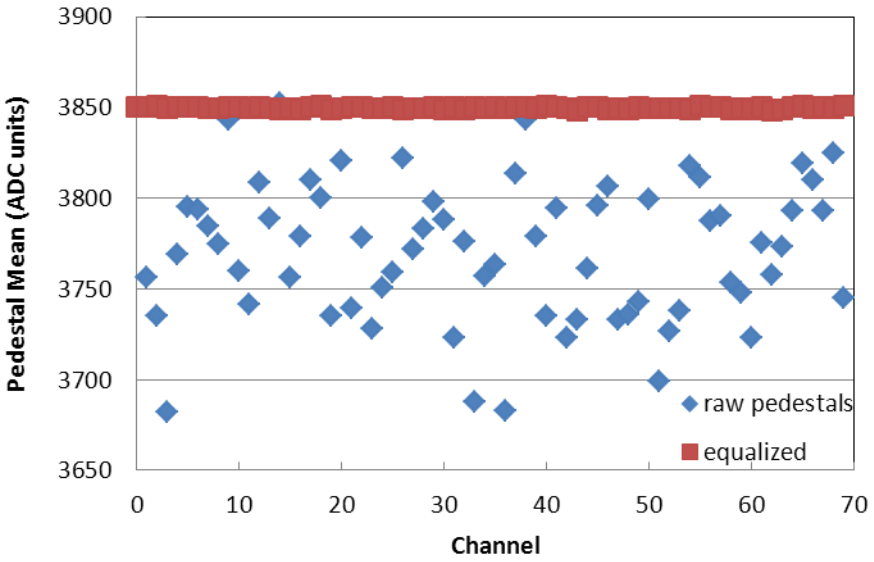


Fig. 33. Mean pedestal of an AGET chip set for positive signal input.

In this test, the charge range is set to 120 fC, the shaping time is 540 ns and SCA write frequency is 100 MHz. No detector is connected to the front-end card and it is shielded. The RMS of the pedestals is shown in Fig. 34. The four FPN channels exhibit lower pedestal rms and are clearly identified.

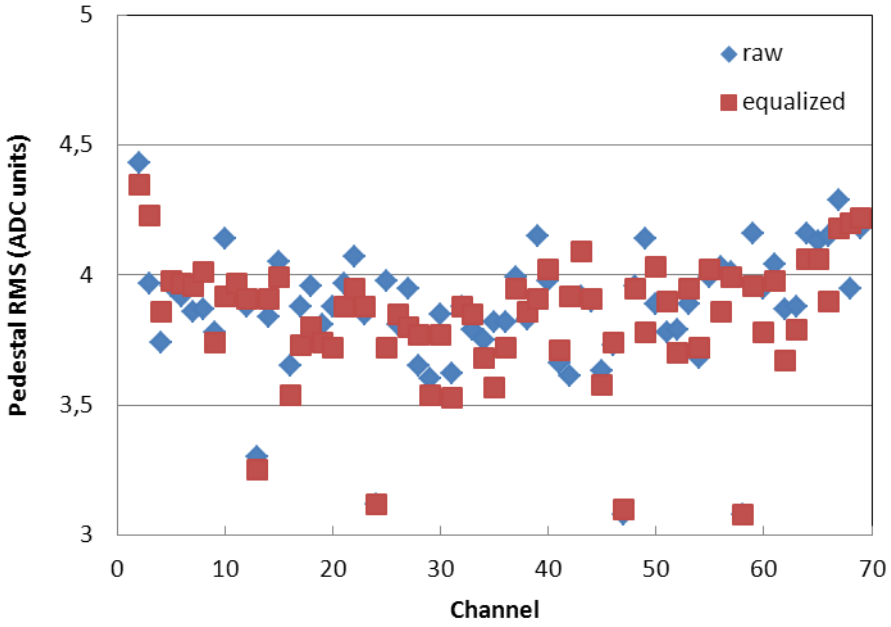


Fig. 34. Pedestal RMS of an AGET chip set for positive signal input.

9.3 Using the Pulse Generator of the FEC

The Feminos card can control the pulse generator embedded on the FEC. A typical sequence of commands to inject a pulse is the following:

- SCA_AUTO_START must be disabled.
- The pulser must be enabled, i.e. PUL_ENABLE is set.
- All trigger sources must be disabled except for the pulse generator (i.e. PUL_TRIG_ENABLE is set and AUTO_TRIG_ENABLE, EXT_TRIG_ENABLE and TCM_TRIG_ENABLE are all cleared).
- Alternatively, instead of setting PUL_TRIG_ENABLE, the multiplicity trigger can be enabled (i.e. MULT_TRIG_ENA is set). In the case, firing the pulser will cause a self-trigger if the multiplicity threshold is set properly.
- Set the pulser delay via the command "pul_delay <0xDelay>"
- Set the trigger delay to a value that corresponds to one complete turn of the SCA ring (depends on SCA write clock divisor)
- Pre-load the baseline output level of the pulse generator DAC with the "pul_amp <0xBaseline>" command. Note that bit 15 controls the injection multiplexer on the FEC and must be set when operating in "functional test mode". In "calibration mode" this bit should not be set.

- Load the baseline level with the "pul_load" command. The baseline should be close to the maximum DAC amplitude when operating in the negative input polarity mode and close to the minimum DAC amplitude for positive input polarity.
- Pre-load the desired step amplitude of the pulse with the command "pul_amp <0xStep>" (note that bit 15 must remain set in "functional test mode"). The amplitude must be less than the baseline previously set for the pulser DAC to make a negative input pulse and greater than the baseline to obtain a positive pulse.
- Start SCA write operation with the "sca start" command.
- Wait for event completion. Pulse injection, SCA control and event digitization are automatic.
- Get event data.
- Repeat these operations as needed from the step where the baseline of the pulse generator DAC is pre-loaded.

Typical pulses obtained with an AGET chip in calibration mode for negative and positive input pulses are shown in Fig. 35 and Fig. 36 respectively. In these tests the charge range is 120 fC, the shaping time is 120 ns, SCA write frequency is 100 MHz.

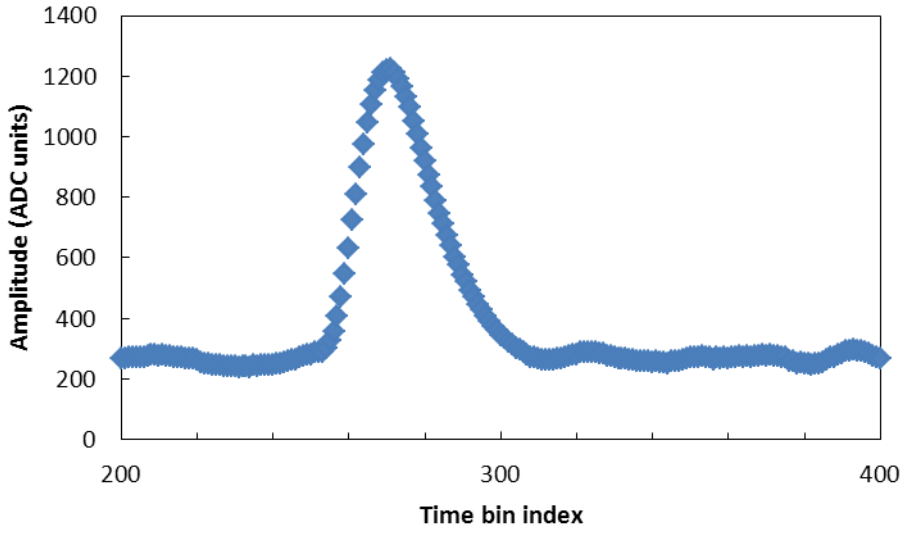


Fig. 35. Channel data obtained with the FEC on-board pulser (negative mode).

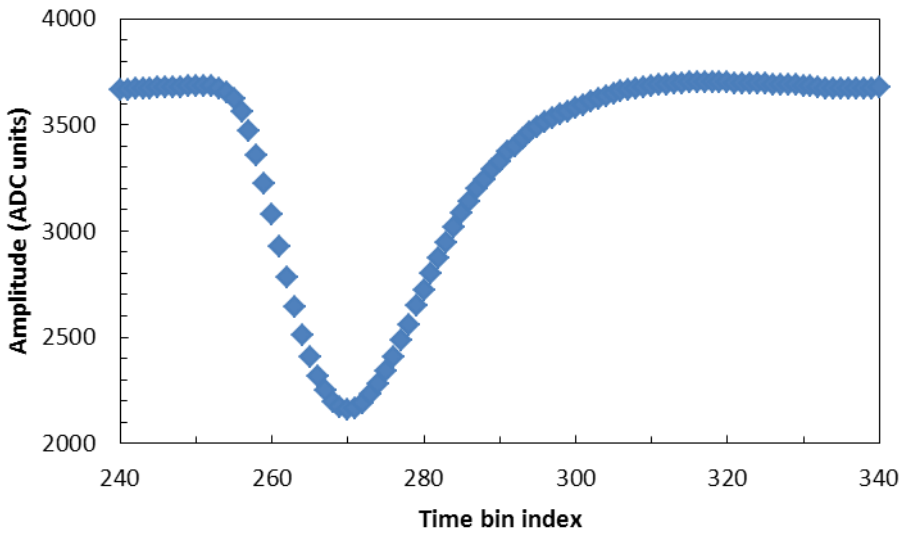


Fig. 36. Channel data obtained with the FEC on-board pulser (positive mode)

9.4 Setting Discriminator Thresholds in the AGET Mode

A per-channel discriminator in the AGET chip can be set to adjust the level that determines if a channel is hit or not. Before these levels can be set, the Feminos card needs to perform a series of runs where all the possible values of the thresholds are tried, and for each value, the probability that each channel is hit by random noise is estimated. The AGET chip uses 8-bit threshold values: 4 bits come from a signed DAC which is common to all channels, while the 4-LSBs are defined on a per-channel basis. The Feminos is only capable of accumulating 16 hit probability histograms per channel, i.e. the value of the common part of the threshold should be fixed during the threshold scan run. If the value has to be changed, a second run must be made. During the threshold scan run, the Feminos will typically generate a certain number of events at a given fixed rate (e.g. 100 events at 100 Hz) and determine how many times each channel was hit (by reading the hit channel register). The discriminator threshold of all channels is increased after each run and all 16 possible threshold values are scanned. At the end of the run, the turn-on curves of the discriminator of all channels are available. The user can then instruct the Feminos to set the discriminator threshold of each channel to some value expressed as a probability: e.g. the command "aget * hitprob * 2" will attempt to set the threshold of all channels of all AGET chip to a value where the probability that each channel is hit is less than 2%. For some channels, it may not be possible to set the threshold to meet the desired value. It may be required to increase the level of the common part of the threshold, or if the channel is too noisy, it could be disabled.

9.5 Using the Multiplicity Trigger

During the sampling phase, each AGET chip outputs an analog multiplicity signal that reflects the number of channels hit in the chip. However, given the analog nature of these signals, there is no direct exact correspondence between the number of channels hit and the digital converted value of the multiplicity signal. Fig. 37 shows the 8 MSB's of the digitally converted multiplicity signal of an AGET chip when the number of hit channel is varied. There is an initial offset of ~23 units and the curve is rather linear with a slope of ~3.2 units/hit channel. These figures, and the possible dispersion between chips, must be taken into account when setting the multiplicity threshold of each AGET. Individual threshold fine tuning is probably necessary, but a good starting value is to put the same value for all chips that corresponds to at least 2-3 channel hit per chip (i.e. ~32 units).

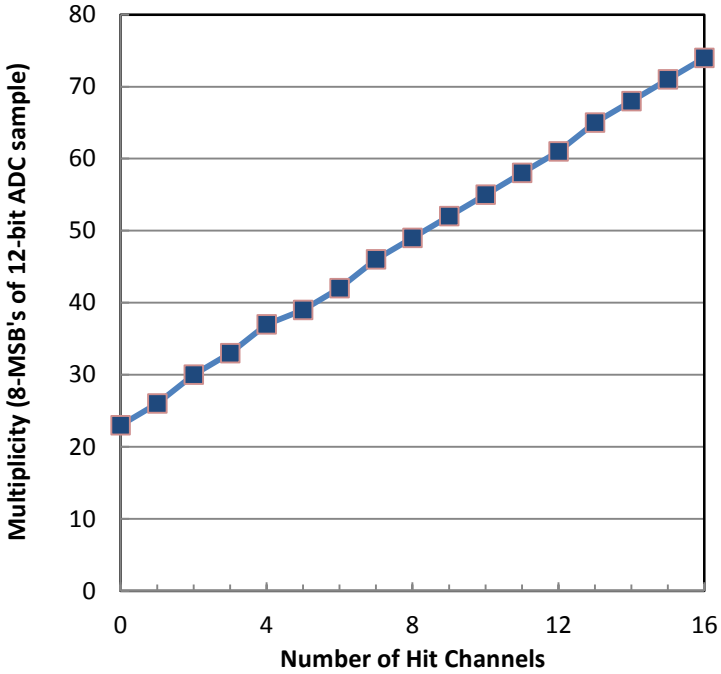


Fig. 37. Digital multiplicity versus number of hit channels.

The Feminos can use the digitized multiplicity signal of each AGET chip to build a local self-trigger, or a condensed form of the multiplicity information can be sent to the TCM for processing and elaboration of a global multiplicity trigger. The logic that generates the multiplicity trigger primitive of each AGET chip is shown in Fig. 38. This logic is duplicated 4 times in the Feminos.

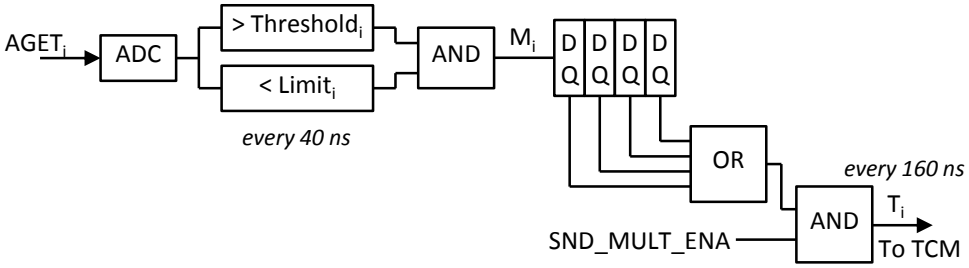


Fig. 38. Logic to generate the multiplicity primitive of each AGET.

The ADC of the Feminos is clocked at 25 MHz and consequently, the digitized multiplicity is only updated every 40 ns. Values that are above a programmable threshold and below a programmable limit make a multiplicity hit. Four time consecutive multiplicity hits are OR'ed together to make a single bit multiplicity primitive per AGET that is sent every 160 ns to the TCM (when SND_MULT_ENA is set).

The multiplicity hit bits of the 4 AGET chips driven by the Feminos are also OR'ed together to produce a self-trigger based on multiplicity when MULT_TRIG_ENA is active. This is shown in Fig. 39.

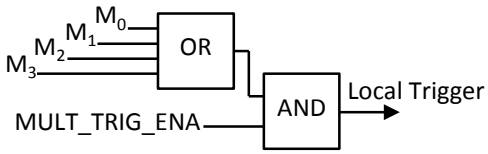


Fig. 39. Logic to build a self-trigger based on multiplicity.

When multiplicity data processing is enabled on the TCM side, this card performs the digital arithmetic sum of all the multiplicity bits it receives from the Feminos. The TCM applies on this digital sum a lower limit threshold and an upper limit threshold. If the multiplicity sum is within the acceptable window, a global multiplicity trigger is generated. This trigger signal is either sent back to all Feminos or it is forwarded to other external hardware.

9.6 Decoding Data Files Stored in Binary Format

The data acquired by one or several Feminos cards can be saved to files by the mclient program in binary format for later analysis. Decoding these files is not difficult and some helpful tips are given below.

The first information found at the beginning of each data file is an ASCII string encoded with the prefix format earlier described in this document: reading the first two bytes of a file and interpreting the data as a 16-bit unsigned integer (in little-endian ordering), the 8-MSBs (i.e. the second byte stored in the file) should always be 0x01 (corresponding to prefix PFX_ASCII_MSG_LEN) and the 8-LSBs (i.e. the first byte

stored in the file) represent the size of the string (usually 0x1A, i.e. 26 characters). The string typically contains the date when the run was started and a three digit file index within the run. The string is terminated by a null character.

To decode a file, the following method is recommended:

Step 1. Read 2 bytes of data from the current position of the file pointer

Step 2. If these bytes match PFX_ASCII_MSG_LEN, determine the length of the string, read the required number of bytes from the file and print the string or ignore it. If the two bytes are PFX_START_OF_BUILT_EVENT or PFX_END_OF_BUILT_EVENT, do the action required and go back to Step 1. If these are PFX_START_OF_DFRAME, or PFX_START_OF_MFRAME or PFX_START_OF_CFRAME, read two more bytes: these give the number of bytes to read from the file for the current frame. Read the specified number of bytes and go to Step 3. If the two bytes read at Step 1 do not match any of the prefix above, print an error message and abort.

Step 3. Decode the frame according to its type (see Feminos I/O format for details) and return to Step 1. Files will generally only contain data frames, but may also contain monitoring or configuration reply frames.

These steps have to be repeated until the end of file has been reached.

10 OVERVIEW OF THE FPGA FIRMWARE

The firmware of the FPGA of the Feminos card comprises about 30 different interconnected blocks. A simplified diagram is shown in Fig. 40. The command interpreter software runs on the MicroBlaze processor block which communicates with the external data and program memory through the MCB (Memory Controller Block) hard IP. A soft core IP implements the Gigabit Ethernet MAC layer and manages the interface to the external PHY. Internal communications between the MicroBlaze processor and the hardwired logic functions of the Feminos card are handled by the Register Bank block, shared memory locations placed in the Dual Port RAM block, and the Ring-Buffer Interface block.

The SCA Manager block is the central element that controls the sequence of operations of the different other blocks. After system, configuration, this block enables the write clock and write signal in the SCAs of the front-end ASICs. When a valid trigger is received (via the Trigger controller block) the SCA manager triggers the operation of the ASIC SC manager block. This role of the ASIC SC manager block is to read and optionally alter the content of the Channel Hit Register (in the AGET mode) using the slow control lines of the front-end ASICs. This block also contains arbitration logic to allow the MicroBlaze processor to get ownership of the ASIC slow control lines to configure ASIC registers. The ASIC SC manager block also produces information that reflects the content of the 4 hit channel registers (or the equivalent in the AFTER mode) and sends it to the Event pre-processor block. The Event per-processor block produces the lists of the index of all the channels that will be readout in each ASIC for the current event. It also determines what the largest number of channel hit per ASIC is among the four current values. This number multiplied by the number of SCA cells determines the number of digitization cycles to perform. The SCA manager performs the number of readout cycles that are required. Data from the external ADC are de-serialized by the ADC interface block and moved from the ADC clock domain to the local clock domain. The series of ADC samples are sent to the ZBT feeder block that forwards them to the ZBT controller block and the external memory. At the end of the SCA digitization phase, the SCA manager checks that the ZBT memory buffer has sufficient space to store the next event (assuming the largest possible event size). If enough space is available, the SCA write operation can be restarted, unless the Feminos is programmed to wait until the SCA write order is received via the TCM interface. If the external ZBT cannot store the next event, the system is in dead-time until sub-sequent blocks have freed the minimum required space. The Busy-meter block measures the time from the reception of the trigger signal until SCA write resumes.

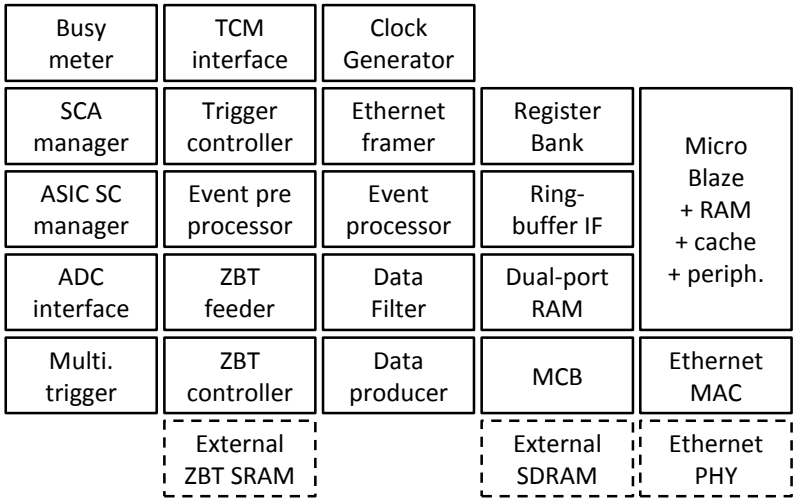


Fig. 40. Simplified block diagram of FPGA firmware

During SCA write phases, and even during the digitization of the current event, the firmware and processor of the Feminos card are capable of processing and transferring to the DAQ the data of previous events. The central element for event data readout is the Event processor block. It is triggered when digitization of an event has been completed. The Event processor fetches one-by-one the index of the channels that have been hit and asks the Data Producer block to obtain the corresponding data. The Data producer block reads the event data of the current channel from the external ZBT SRAM though the ZBT controller block and feeds the Data Filter block with that stream of ADC samples. The Data filter block optionally subtracts the constant pedestal and, also optionally, performs zero-suppression. When the data of the current channel has been completely processed, it is passed to the Ethernet framer block and the Event Processor block starts the readout of the next channel. When all the channels of the current event have been readout, the corresponding memory space in the ZBT RAM is freed. The Ethernet framer block obtains from the ring-buffer interface block a memory descriptor that points to free locations in the external SDRAM and transfers the data of the current channel to the corresponding memory locations. If the current buffer does not have enough space to store the data of the current event, the Ethernet framer block posts the corresponding buffer descriptor to the in-bound FIFO of the ring buffer interface and fetches a free buffer descriptor from the out-bound FIFO of the ring buffer interface. The Ethernet framer block repeats the same operations for all the channel data blocks it receives.

The MicroBlaze processor constantly checks if buffer descriptors are available in the in-bound FIFO of the ring buffer interface. When a buffer descriptor is available, the MicroBlaze processor fills the header part of the Ethernet frame stored in this buffer and passes the buffer descriptor to the Ethernet MAC block which reads the frame from the external SDRAM and sends to the DAQ PC through the external PHY and

Ethernet cable connection. Memory buffers and the corresponding buffer descriptors are re-cycled in the out-bound FIFO of the ring buffer interface when frames have been sent.

11 TRIGGER/CLOCK MODULE INTERFACE

11.1 PHYSICAL LAYER

The interface between the Feminos card and the TCM uses the four pairs of a standard cat 6 RJ45 cable. One pair is split into two wires that carry unipolar 3.3V LVTTL signals while the three other pairs use LVDS. All signals are DC coupled. The signals from the TCM to the Feminos card are:

- TCM_CLK: free running 100 MHz reference clock (LVDS)
- TCM_TRIG: serially encoded trigger and synchronous control signals (LVDS)
- TCM_ENAREM: (enable remote partner) active high 3.3V LVTTL signal to indicate to the Feminos that the TCM is present and requests the Feminos side to respond

The signals from the Feminos card to the TCM are:

- TCM_REMDET: (remote partner detected) active high 3.3V LVTTL signal asserted by the Feminos to signal its presence and readiness to the TCM
- TCM_MISO: serially encoded multiplicity and busy/ready signals (LVDS)

A schematic view of the TCM to Feminos interface is shown in Fig. 41.

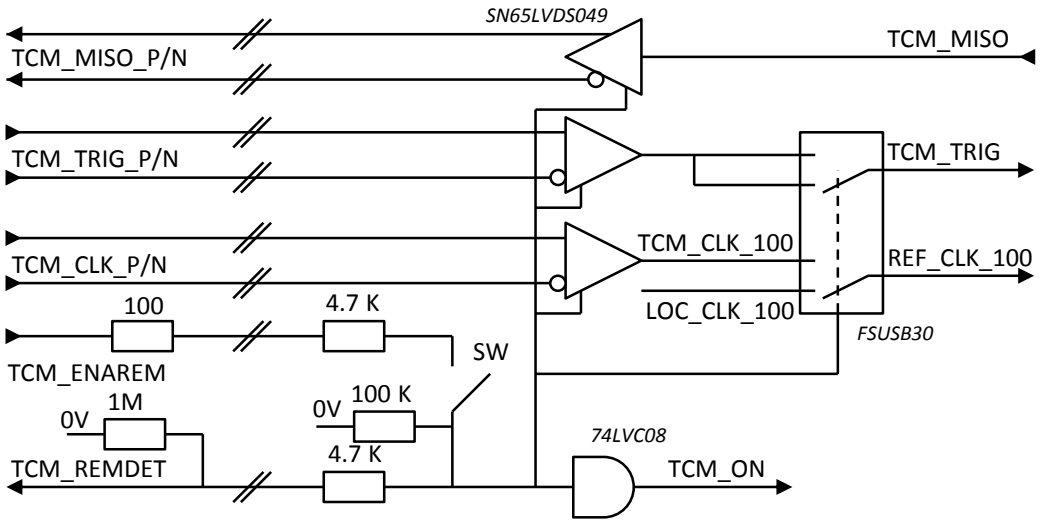


Fig. 41. TCM – Feminos interface.

The TCM de-activates communication with the corresponding Feminos card when it detects a low level on the TCM_REMDET line. This happens in the following situations:

- The TCM sets a low level on TCM_ENAREM to disable this port,

- The TCM has set a high level on TCM_ENAREM but the feminos card is not present, or it is present but not powered, or it is present and powered but SW is open meaning that the Feminos card operates on its own local clock.

When SW is closed, the Feminos takes its locally generated clock as a reference if the TCM is not present or if TCM_ENAREM is low. The Feminos takes TCM_CLK_100 as a reference when SW is closed and the TCM has asserted TCM_ENAREM. The signal TCM_ON signals to the FPGA logic of the Feminos card that the TCM is present and ready.

Note that the TCM must not be allowed to actively drive its LVDS output lines unless a high level is detected on TCM_REMDET. Similarly, the Feminos card is not allowed to actively drive the TCM_MISO_P/N lines unless a high level is sensed on TCM_REMDET. Because series resistors are placed on TCM_ENAREM, limited current will flow if the TCM drives this line to a high level while the Feminos card is present but not powered and SW is closed.

The pin assignment of the RJ45 cable between the TCM and the Feminos is shown in Fig. 42. Note that colors depend on compliance with one of two possible standards.

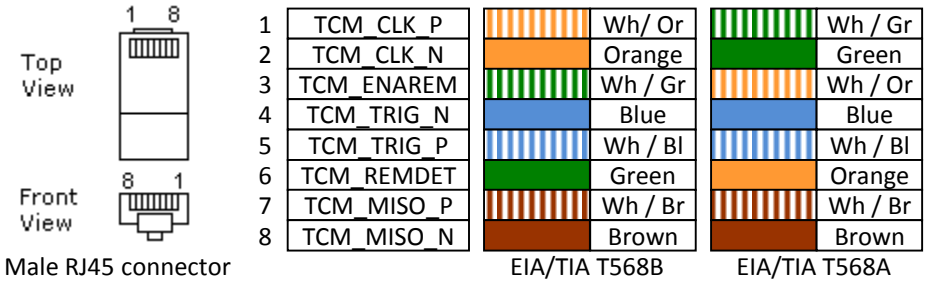


Fig. 42. TCM – Feminos cable pin assignment

11.2 Line Encoding

The TCM_CLK_P/N pair carries a free running 100 MHz clock. The TCM_TRIG_P/N and TCM_MISO_P/N lines carry 8-bit frames at 100 Mbps using, by default, a simple non-DC-balanced, asynchronous protocol with 1 Start bit and 1 Parity bit. Optionally, a DC balanced protocol is also supported. It is obtained by making a XOR between the non-encoded serial stream with a locally generated bit pattern of alternating 1 and 0's. The DC balanced protocol can be enabled independently for the Feminos to TCM link and the TCM to Feminos link with the configuration bits DCBAL_ENC and DCBAL_DEC respectively. Settings must be consistent between all Feminos and the TCM for correct operation.

The format of the frames sent from the TCM to the Feminos card assuming the non-DC balanced protocol is shown in Fig. 43.

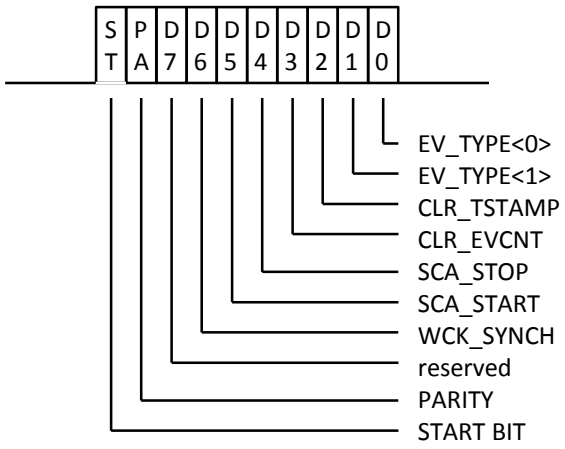


Fig. 43. TCM to Feminos frame format

The marking level of the TCM_TRIG line is a low level. The START_BIT is a high level. The PARITY bit is the exclusive OR of bit D0 to D7. The bit WCK_SYNCH is used to synchronize the clock generator for the SCA write clock. The SCA_START bit is used to synchronously start sampling in the SCA of the Feminos cards. The SCA_STOP bit is the trigger. The event type is determined by EV_TYPE<1..0>. The CLR_EVCNT bit and CLR_TSTAMP bit are used to synchronously clear the event counter and time stamp counter of all Feminos cards simultaneously. The CLR_TSTAMP bit will actually perform a preset of the timestamp counter if an initial value different from zero was set in the Feminos. After the last bit of data, the TCM should insert a gap of at least 6 symbols before the next frame. Consequently, the maximum frame rate sent by the TCM to the Feminos is 1/160 ns = 6.25 MHz. However, the absolute time of synchronization signals can be resolved with 10 ns resolution.

The format of the frames sent from the Feminos card to the TCM assuming a non-DC balanced protocol is shown in Fig. 44.

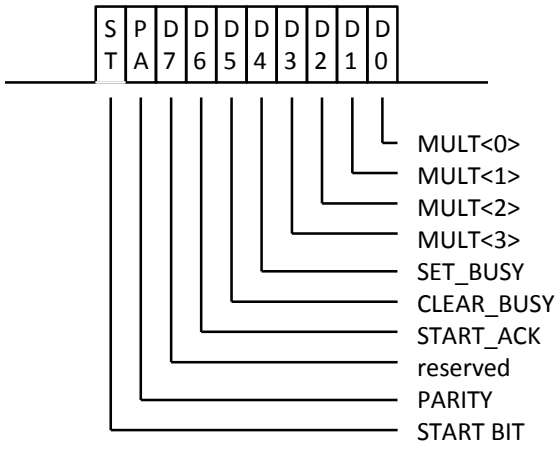


Fig. 44. Feminos to TCM frame format

The marking level of the TCM_MISO line is a low level. The START_BIT is a high level. The PARITY bit is the exclusive OR of bit D0 to D7. After the last bit of data, the Feminos inserts a gap of 6 bits before the next frame. Consequently, the maximum frame rate sent by the Feminos to the TCM is 1/160 ns = 6.25 MHz.

The bit START_ACK indicates that the Feminos card has successfully received the SCA_START command from the TCM and that the front-end ASICs are now sampling data. The SET_BUSY bit is used to acknowledge the reception of the current trigger and indicates that the Feminos card cannot perform a SCA_START command until it sends CLEAR_BUSY. The bit CLR_BUSY is used to indicate that the Feminos card has completed processing of the previous event and is ready to resume the SCA write operation.

The MULT<3..0> field is only used in the AGET mode. It indicates which of the four multiplicity thresholds and limits criteria (to comparisons per AGET) have been satisfied. The bandwidth of the link from a Feminos card to the TCM is not sufficient to send the full precision multiplicity information (6 meaningful bits per AGET chip every 40 ns). The threshold and limit comparators for multiplicity are applied on the digitized multiplicity output of each AGET chip every 40 ns: this produces 4 bits per Feminos every 40 ns. Four of these consecutive words are OR'ed bitwise to produce a 4-bit multiplicity-over-threshold (and below limit) primitive every 160 ns. These 4-bit word are serially encoded and sent to the TCM which combines all multiplicity information to elaborate a trigger. This trigger can be sent by the TCM to some external device or it can be propagated back to all Feminos. When sending multiplicity bits to the TCM is enabled, the Feminos card starts sending this information 10 µs after SCA_WRITE becomes active. This avoids sending multiplicity data during the transient phase when sampling in the front-end SCAs is restarted. Multiplicity frames are then sent continuously at 6.25 MHz to the TCM until a trigger (of any type) is received. During SCA digitization, no multiplicity information is available.

Starting from firmware version 2.6, the Feminos implements a bit error rate tester for checking the quality of the communication link with the TCM. The Feminos must be set first to operate in the bit error test mode, and then this test mode can be engaged in the TCM. The reception of the first frame from the TCM after TCM_BERT is set in the Feminos triggers the bit error rate tester. As long as it receives frames from the TCM, the Feminos continuously sends to the TCM frames in the format previously described (1 start bit, 1 parity bit, 8 payload bits, and a gap of 6 bits) with the payload bits taken from the pseudo-random generator shown in Fig. 45.

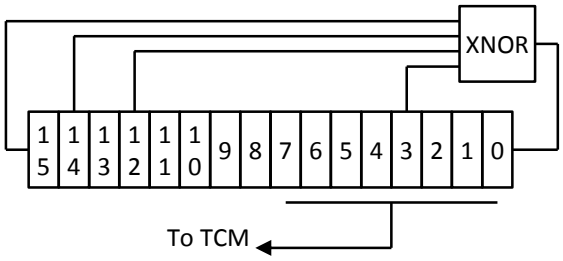


Fig. 45. Bit error rate tester payload generator.

The Feminos checks the content of each frame received from the TCM against the value predicted by its local pseudo-random generator. Any mismatch causes an increment of the relevant error counter. The pseudo-random payload generator is initialized to 0x0000 when the bit error rate tester is disabled. During operation, the shift register is shifted left by one bit position every 16 clock cycles. The sequence of payload words send to the TCM (and expected from it) is 0x00, 0x01, 0x03, 0x07, 0x0F, 0x1E, etc.

12 BOARD TEST AND VALIDATION

12.1 POWER SUPPLIES

The power up delay is determined by switch settings. The measured delay for all 16 possible positions of the delay power switches is shown in Fig. 46.

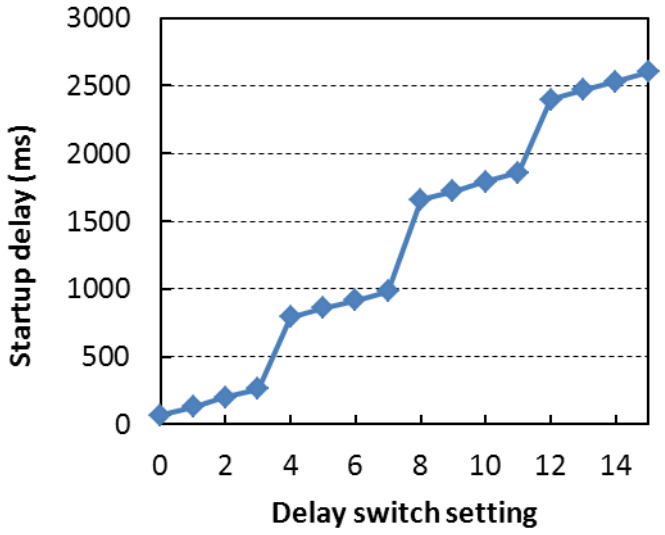


Fig. 46. Power-up delay for all delay switch settings.

The power supply current drawn from a 5 V power supply by a Feminos in operation is 1.1 A and 0.8 A respectively for the LDO based and DC-DC converter based Feminos. This corresponds to a dissipated power of 5.5 W and 4 W respectively. Both types of cards require a heat sink and fan for the Mars MX2 FPGA module but other parts need not specific cooling. The current consumed by a FEC equipped with 4 AGET chips and the LDO based Feminos is 2.2 A in operation, i.e. 11 W or 43 mW per channel. Using a 4 V power supply instead of 5 V is expected to lead to a power dissipation of ~9 W per Feminos-FEC couple.

12.2 INTERFACE TO THE ADC

The digitization of SCAs of the front-end ASICs is made by a quad-channel ADC (Analog Devices AD9229) placed on the FEC card. This device is clocked at 25 MHz leading to a 300 Mbps data rate on the serial outputs (DDR is used). Data integrity on the forwarded clock, data and framing lines is critical to guarantee stable and robust operation. Probing these signals at the level of the FPGA was not physically possible and all measurements have been made at the level of the connector of the Feminos card. The typical signal observed on the DCO pair (150 MHz clock provided by the ADC to the FPGA for data and framing recovery) is shown in Fig. 47 left.

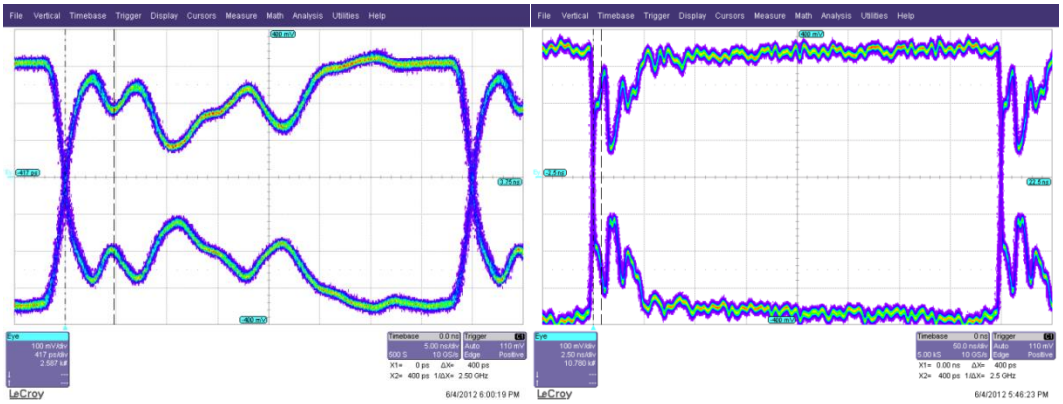


Fig. 47. DCO and FCO signals at the FEC-Feminos connector.

The quality of the signal is not very good on the left side of the eye. Ringing is probably caused by the various impedance mismatches along the way to FPGA pins. However, the opening of the eye is around 200 mV at worst, and other areas are reasonably correct.

The frame recovery signal, FCO, is shown in Fig. 47. This signal is composed of 6 ones followed by 6 zeroes and the bit rate determined by the serial data analyzer is 6 times less than the real rate, i.e. 6 bits of data have to be taken on the eye diagram that is shown. The capture of the last five bits does not seem to be problematic and the degraded area occurs only for the first bit. Nonetheless, the eye opening remains around 200 mV and reliable capture should also be achieved in this region.

Signal on data lines corresponding to chip #0, #1, #2 and #3 are shown on Fig. 48 and Fig. 49 respectively. For this test, the ADC is placed in test mode and sends a constant "101010101010" data pattern. The serial data analyzer determines the bit rate correctly (300 Mbps) and the width of the eye diagram corresponds to one UI on the screenshots.

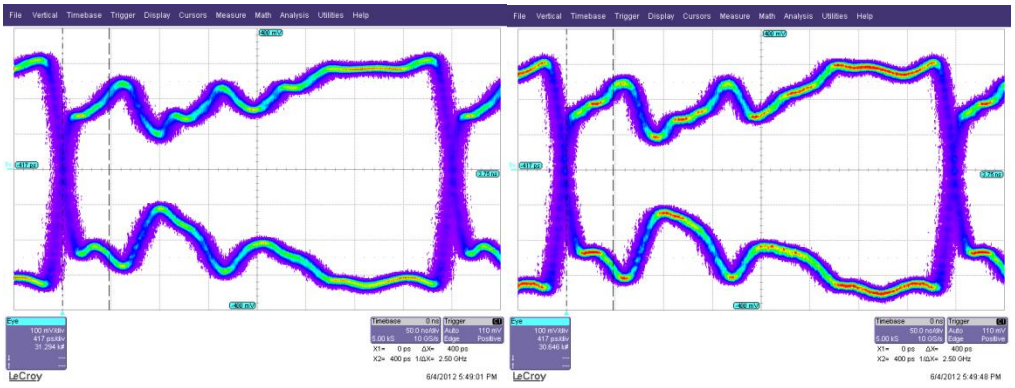


Fig. 48. Data signal for chip #0 and #1 on the FEC-Feminos connector.

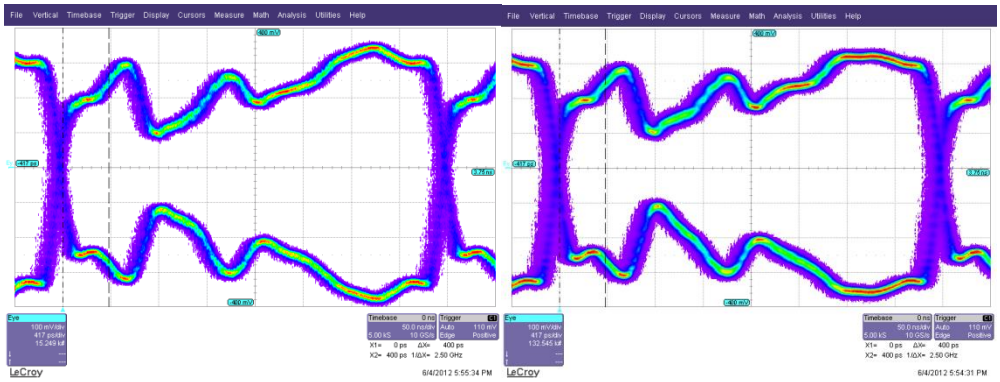


Fig. 49. Data signal for chip #2 and #3 on the FEC-Feminos connector.

The same degradations are observed on all data signals pairs, but the eye opening is always around 200 mV at least.

We also measured the data signal for chip #0 at three different locations: on the FEC-Feminos connector on the FEC side, on the same connector but on the Feminos side, and finally on the Mars MX2 module itself at the level of the connector to the Feminos which is the point closest to the FPGA that can be probed (~1 cm from the FPGA). The results are shown in Fig. 50.

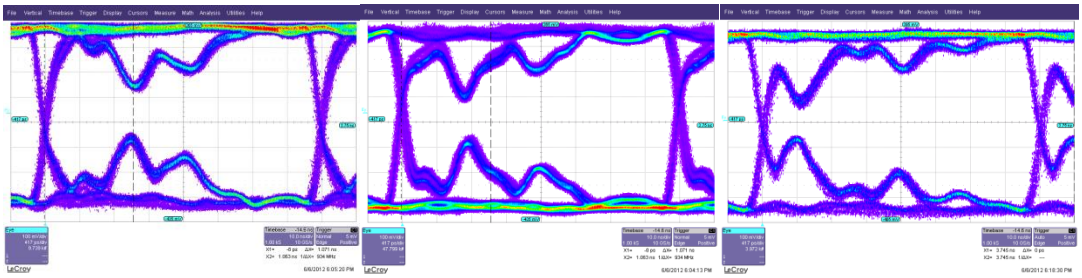


Fig. 50. Data for chip #0 at the FEC, Feminos and Mars module levels.

It can be seen that the degraded area occurs earlier in time as the measurement point gets closer to the FPGA. Although it cannot be proved, it is likely that signal quality at the level of FPGA pins is superior to what is measured further away because the main oscillation that is observed is due to signal reflection at the FPGA pin level.

The main conclusion of these measurements is that the design seems reasonably safe on this particular aspect of signal integrity. This is confirmed by the stability of the framing pattern recovered by the logic, the test pattern recovered from the data lines in test mode, and the real data. However, if single or multiple bit errors occur on real ADC data, they might go undetected, especially if only the LSB's are affected. From the current feedback on operation, the risk of bit capture errors on ADC data is thought to be negligible.

12.3 SCA Write Phase

For the SCA write phase, two signals are being used: the write clock (LVDS) and the write signal (LVCMOS). The write clock is obtained by division of the 100 MHz reference clock by DDR synchronous logic in the FPGA of the Feminos. It is output in LVDS directly by the FPGA of the Feminos. The write signal is generated by a state machine in the FPGA logic. Four copies of the write clock are made on-board the FEC using a 4-port LVDS to LVDS repeater (SN65LVDS104). All lines are differential, controlled impedance traces and 100 ohms terminated. The delay of the repeater is 2.4 ns minimum and 4.2 ns maximum. The write signal is connected directly from the FPGA of the Feminos to the 4 write inputs of the ASICs of the FEC using a T-shaped un-terminated line – which is an incorrect design. The write signal and clock signal measured at the level of each ASIC of a FEC are shown in Fig. 51, Fig. 52, Fig. 53 and Fig. 54.

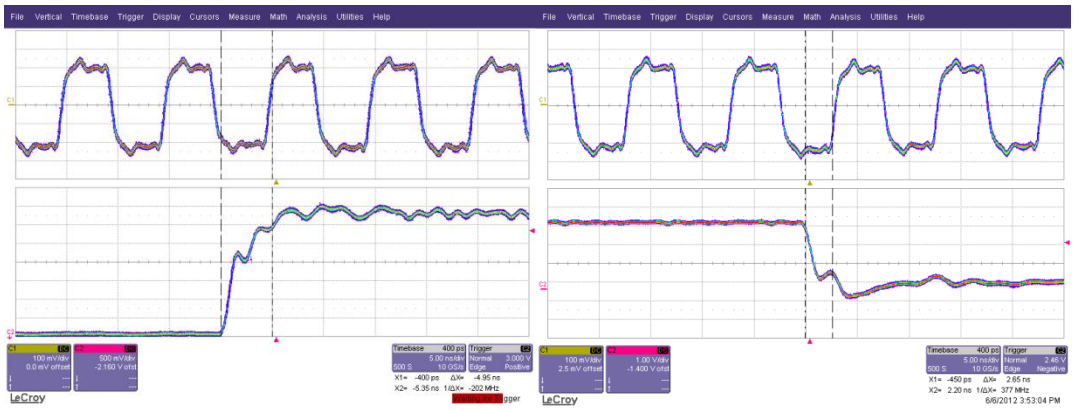


Fig. 51. SCA write and write clock (100 MHz) on ASIC#0 of a FEC.

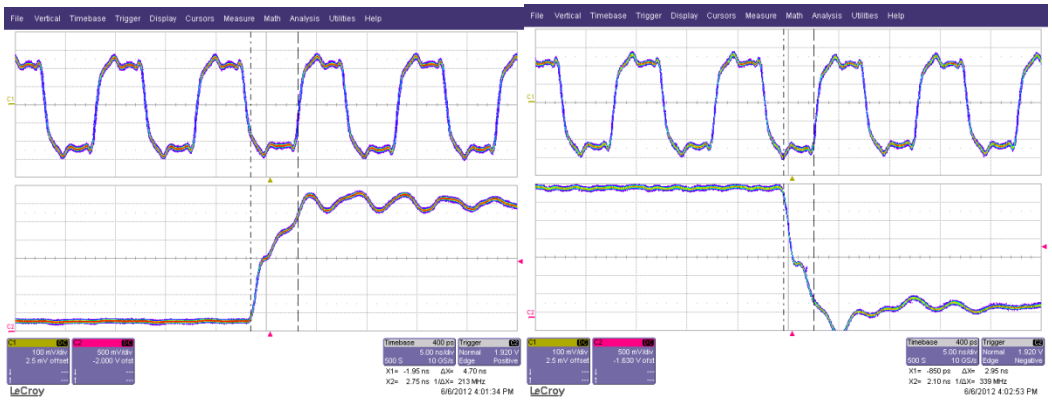


Fig. 52. SCA write and write clock (100 MHz) on ASIC#1 of a FEC.

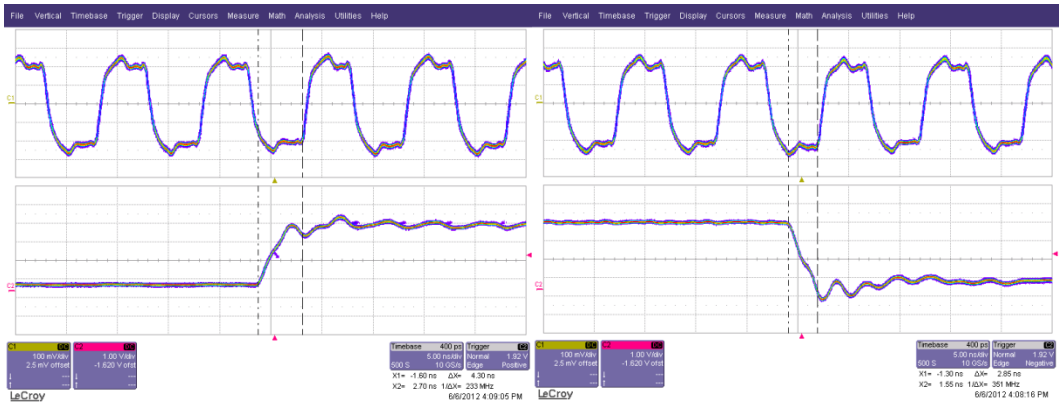


Fig. 53. SCA write and write clock (100 MHz) on ASIC#2 of a FEC.

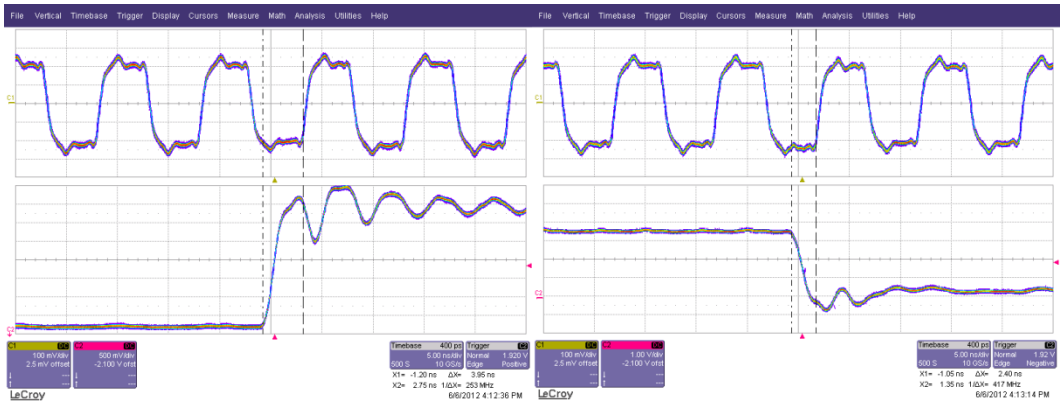


Fig. 54. SCA write and write clock (100 MHz) on ASIC#3 of a FEC.

The write clock signal is satisfactory but ringing is observed on the write signal. This is caused by the sub-optimal layout of this trace. Nonetheless, the measured setup time is ~5 ns for all ASICs and the beginning of the write phase. The hold time on the last write is ~7.5 ns leading to ~2.5 ns between the falling edge of the write signal and the next rising edge of the clock. The beginning of the write phase appears to be safe but the end of the write phase has a shorter margin. If synchronization uncertainties are found, the slew rate of the buffer of the write signal could be reduced, or the write clock could be suppresses outside of the write phase.

We measured the write and write clock signal for different values of the write clock divider. Results are shown in Fig. 55.

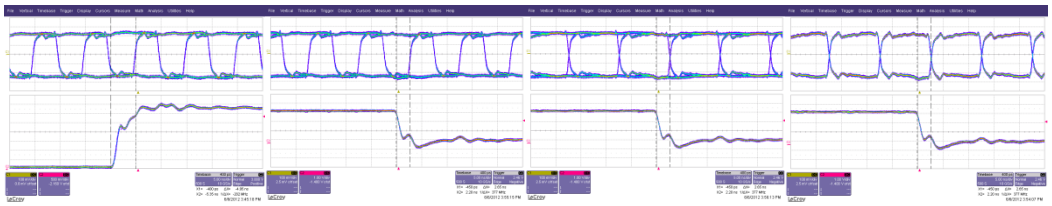


Fig. 55. Write begin/end at 33 MHz; write end at 25 MHz and 50 MHz.

It can be seen that the minimum setup and hold times are unchanged compared to the 100 MHz write clock case. No additional uncertainty on the write phase boundaries is introduced by the setting of the write clock divisor.

Jitter on the SCA write clock was measured in three different conditions: a) the primary clock source is the on-board 50 MHz oscillator of the Feminos, b) the clock is provided by the 50 MHz local oscillator of the TCM and transported (at 100 MHz) to the Feminos via a 1 m cat. 6 STP cable, c) same as above with a 2 m long cable. The jitter histograms measured on the write clock at 100 MHz for ASIC#0 of a FEC in test conditions a) and c) are shown in Fig. 56.

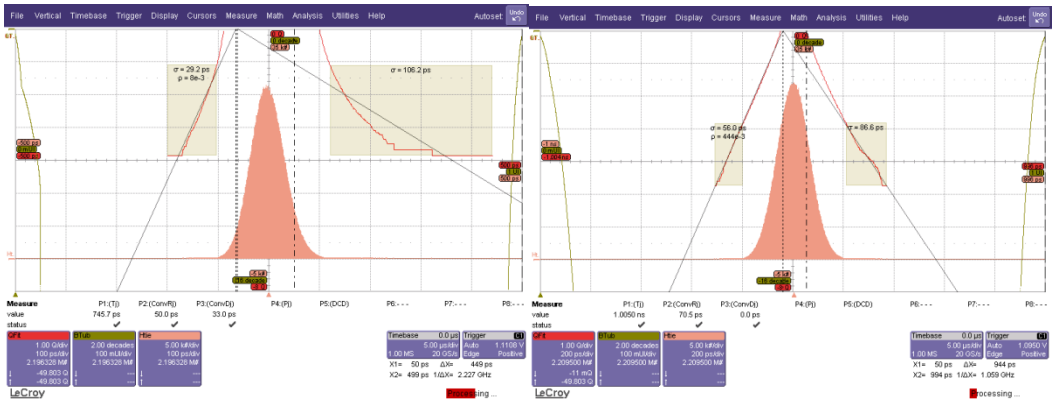


Fig. 56. SCA write clock jitter with different primary clock sources.

Using a local reference clock, the measured random jitter is 50 ps. Using the source clock provided by the TCM over 2 m of cable, random jitter is 70 ps. Using a 1 m cable, it is 68 ps. These figures are very satisfactory.

12.4 SCA Read Phase

The SCA write phase relies on three signals: the SCA read clock (LVDS), the ADC clock (LVCMOS) and the SCA read signal (LVCMOS). The fanout of the SCA read clock and read signal are made in the same way as for the write signals: LVDS fanout buffer for the SCA read clock and T-shaped un-terminated line for SCA read. The ADC clock has an optional delay circuitry on the FEC that is not used, i.e. the 25 MHz ADC clock signal generated by the FPGA of the Feminos is connected directly to the ADC clock input.

The read signals observed for ASIC#0, ASIC#1, ASIC#2 and ASIC#3 are shown on Fig. 57, Fig. 58 and Fig. 59 respectively. The ADC clock signal is in blue, the SCA read clock in yellow and the SCA read signal is in purple.

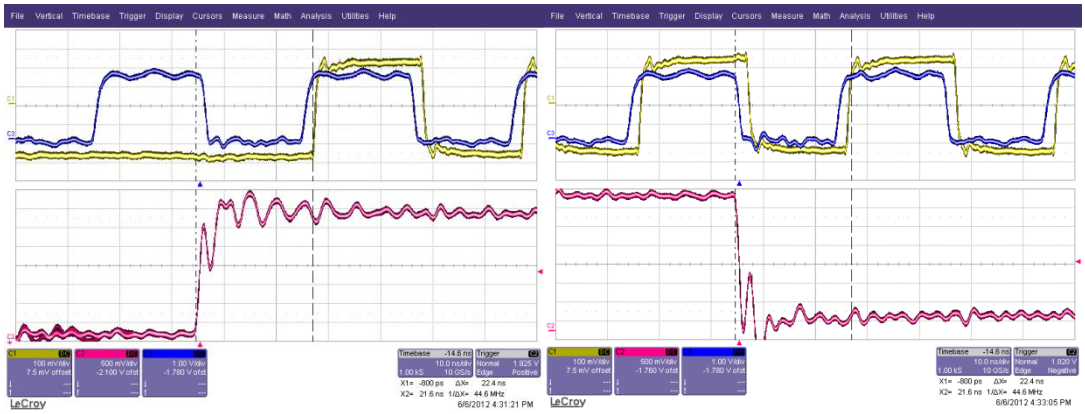


Fig. 57. Read clock, read signal and ADC clock for ASIC#0.

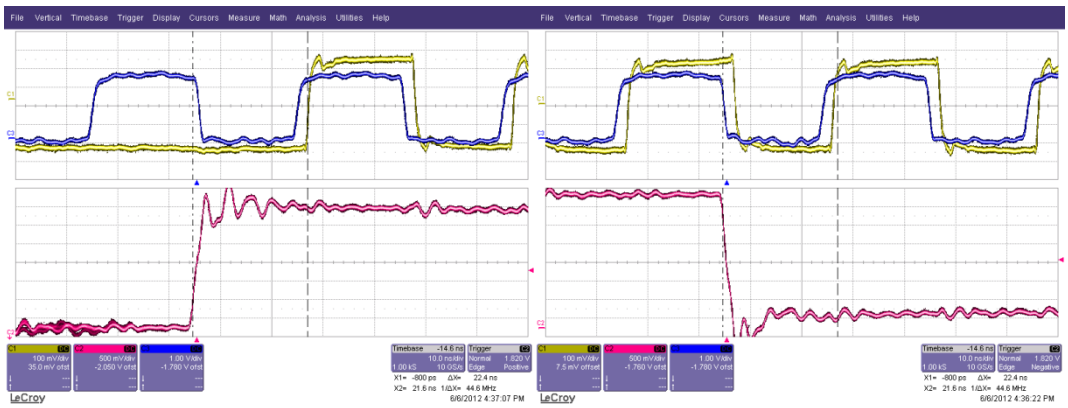


Fig. 58. Read clock, read signal and ADC clock for ASIC#2.

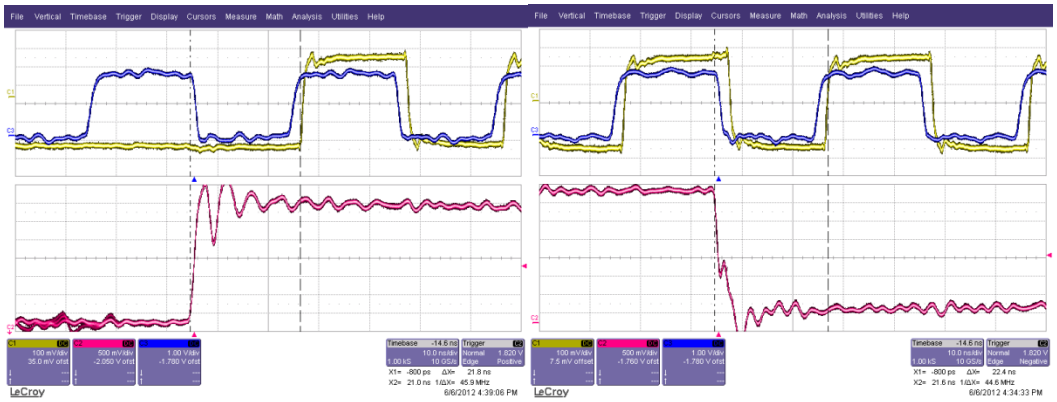


Fig. 59. Read clock, read signal and ADC clock for ASIC#3 and ASIC#1.

Ringing is observed on the read signal but it can be seen that ~20 ns of setup and hold time is obtained in all cases which gives comfortable safety margins.

12.5 ASIC Slow Control

Reading from and writing to ASIC registers uses a serial clock (SPI_SCLK), a serial data output from the Feminos to the FEC (SPI_MOSI), four chip select lines (SCA_CS<3..0>) and four inputs from the FEC towards the Feminos (SCA_MISO<3..0>).

All signals are 3.3V LVCMOS and use un-terminated lines. For chip configuration and configuration read-back, the serial protocol, including serial clock toggling, is implemented in software. Setup and hold delays of the various signals with respect to the serial clock are therefore easily met. Independently of the operation rate, a good signal quality is necessary for reliable operation. The serial clock line has a T-shaped structure on the FEC and this caused sporadic double clocking on ASIC#3 when using some Mars MX2 modules. The issue was resolved by changing the slew rate of the buffer driving this line. The SPI_SCLK signal measured at the level of the FEC connector with the normal slew rate buffer and slow slew rate buffer are shown on Fig. 60.left and Fig. 60.right respectively. It can be seen that using the slower buffer is beneficial.

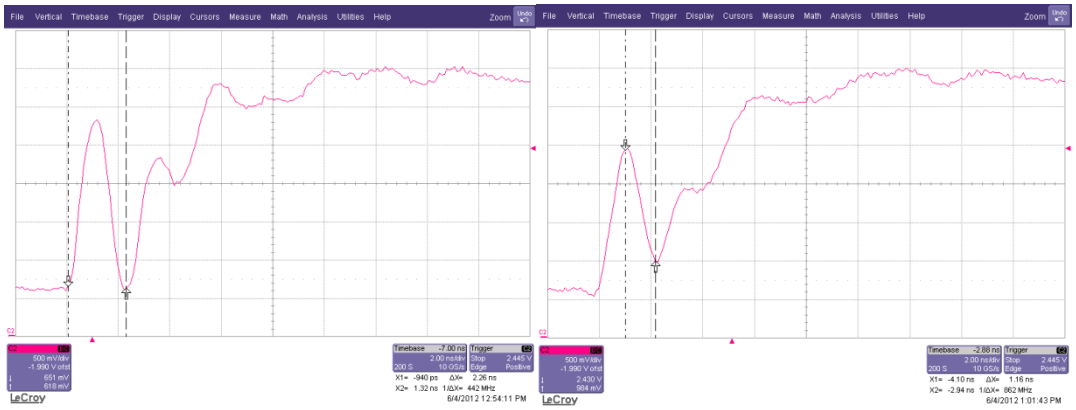


Fig. 60. Zoom on the rising edge of SPI_SCLK.

The SPI_SCLK signal has also been measured at the level of each ASIC on the FEC. This is shown in Fig. 61. The clock edge is not very clean but we verified that registers in all chips can be correctly written and read-back by performing 10.000 iterations on each of the 4 ASICs of the following sequence (using a 36-bit register of the AFTER chips): write 0x0 0x0 0x0, read-back and check, write 0xA 0xAAAA 0xAAAA, read-back and check, write 0x0 0x0 0x0, read-back and check, write 0x5 0x5555 0x5555, read-back and check. No error were found with the corrected firmware (slow slew rate buffer) while thousands of errors on ASIC#3 (but no error on all 3 other ASICs) were found with the version that uses a normal slew rate buffer.

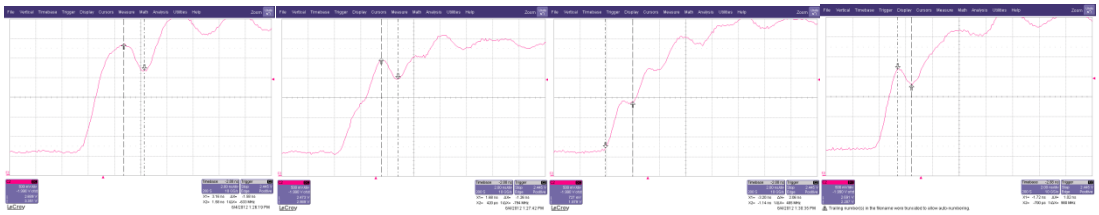


Fig. 61. SPI_SCLK on ASIC#0, #1, #2 and #3.

We conclude that the read/write operations to ASIC registers via the slow control lines can be made correctly. It is nonetheless advisable to perform a read-back and verify after a write operation.

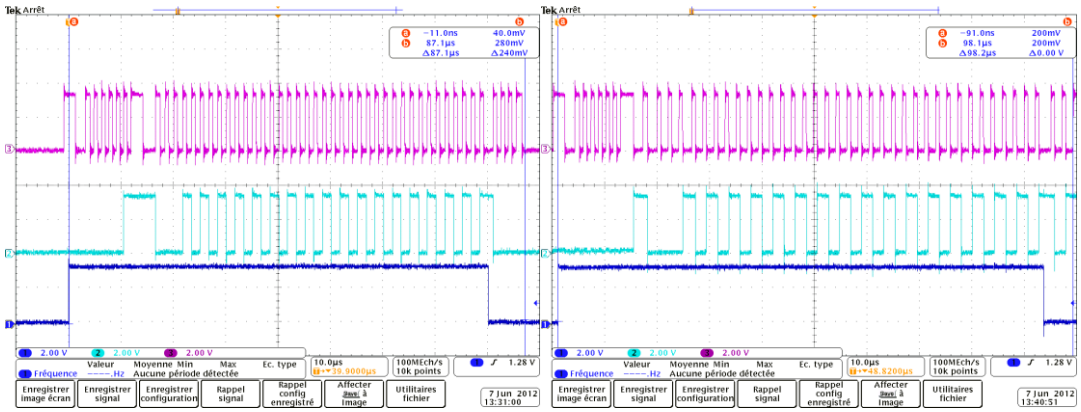


Fig. 62. Typical write operation to a 36-bit register.

A typical write operation and read to an ASIC slow control register are shown in Fig. 62 left (write) and right (read). The signal SCA_CS is in blue, SPI_SCLK in purple and the trace in light blue is SPI_MOSI on the write plot and SCA_MISO on the read plot. The write operation takes ~90 µs while the read takes ~100 µs. The typical frequency of the serial clock is 500 kHz.

12.6 AGET Hit Register Read / Write

The lines that are used for ASIC slow control register read / write are also used in the AGET mode to read and optionally write the channel hit register. This operation is handled in the firmware of the FPGA of the Feminos. The serial clock SPI_SCLK is 50 MHz in this case and the timing needs to be watched carefully. The overall sequence for hit channel register read and modify is shown in Fig. 63 left.

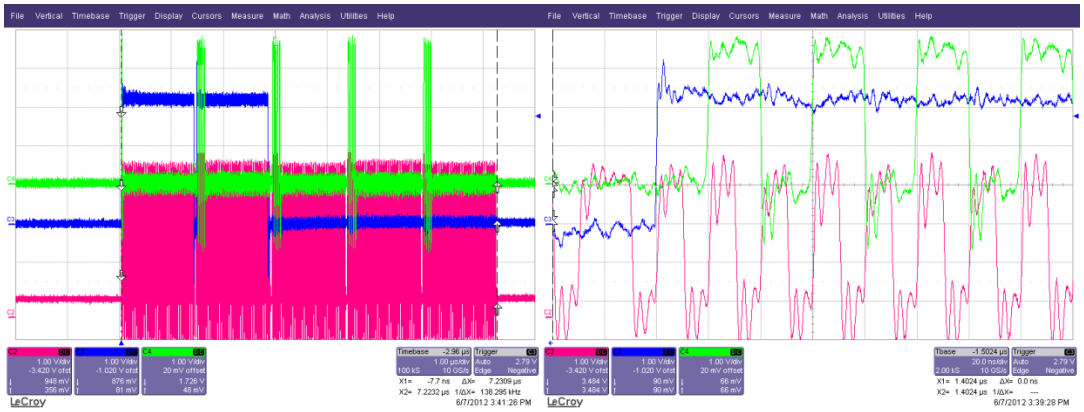


Fig. 63. AGET hit register read and write-back.

The trace in purple is SPI_SCLK, the green trace is SPI_MOSI, and the blue trace is SCA_CS<0>. The sequence is divided into five steps: firstly the hit channel register of

the 4 AGET are readout in parallel (the signals SCA_CS<3..1> are not shown), then the channel hit register of each AGET is written sequentially. Each step takes ~1.44 µs and the complete sequence takes ~7.2 µs. The right part of Fig. 63 is a zoom on the beginning of the write phase for the channel hit register of AGET#0. In this test, 4 channels (one every two) have been forced to 1 in each AGET. It can be seen that the correct signals are found at the input of AGET#0.

A zoom at the beginning and the end of the read sequence is shown in Fig. 64. Note that the green trace is SCA_MISO<0> in this case. The threshold of the AGET discriminator is set to the maximum value (i.e. no channel is hit) except on the first few channels where one channel out of two has its discriminator threshold set to 0 (i.e. one channel out of two is hit). It can be seen that the content of the hit channel register can correctly be readout. Setup and hold times are ~10 ns which is optimal at 50 MHz. Ringing is observed on the serial clock line, but does not seem to cause double clocking errors.

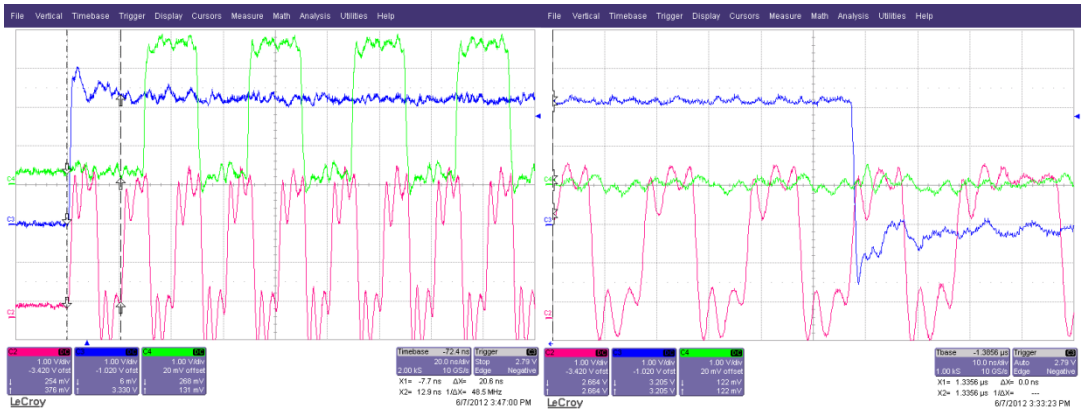


Fig. 64. AGET#0 Channel hit register read sequence.

In addition to these tests, the analysis of data can easily show if each individual bit of the hit channel register can be forced to 1 or 0. At present, this type of errors has not been observed. If errors occur in the read phase of the hit channel register, they will probably not be detected. By forcing the threshold of each channel to the maximum or minimum value, it is nonetheless possible to build a very probable pattern in the channel register and verify in the data correctness of the readout values.

12.7 SRAM Interface

The Feminos uses a 9-Mbit ZBT SRAM to store ADC data and read-back events in transposed ordering. This SRAM operates at 200 MHz and performs burst of two write access followed by two read access. The FPGA logic is clocked at 400 MHz to meet timing requirements. Most signals between the FPGA and the SRAM of the Feminos cannot be probed. A couple of data lines could be spied-on. The signals observed on RAM DQ<10> and RAM DQ<11> are shown on Fig. 65.

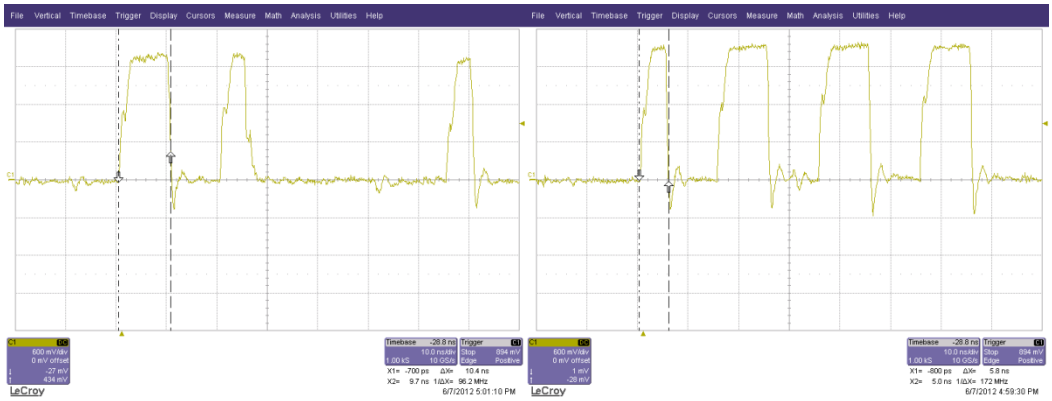


Fig. 65. Signal on RAM_DQ<10> and RAM_DQ<11>.

It is not possible to determine if the operation is a read or a write, but these graphs give an idea on the quality of the signals. It can be seen that each 5 ns symbol is distinguishable. In practice, the only method to verify the correct operation of the external SRAM is to write a known pattern instead of data received from the ADC and compare off-line the data acquired with the expected values. At present, theses verifications have been made on few thousands operations. No error has been found, but more extensive tests would give higher confidence in this stage although the level of correctness found in the data that are acquired is already a good indicator of the validity of the design.

12.8 Quality of Interface Signals from the TCM

The Feminos receives the primary clock and serially encoded trigger from the TCM via two LVDS pairs. The quality of these signals is essential to guarantee a robust communication between these two cards. For test purposes, a fixed pattern of alternating 1's and 0's is sent at 100 Mbps by the TCM to the Feminos instead of the serially encoded trigger (equivalent to the DC-balanced protocol). The eye diagram, measured on the Feminos side, for the trigger pair using the clock pair as a reference is shown in Fig. 66 for 1 m and 2 m long cables between the two cards.

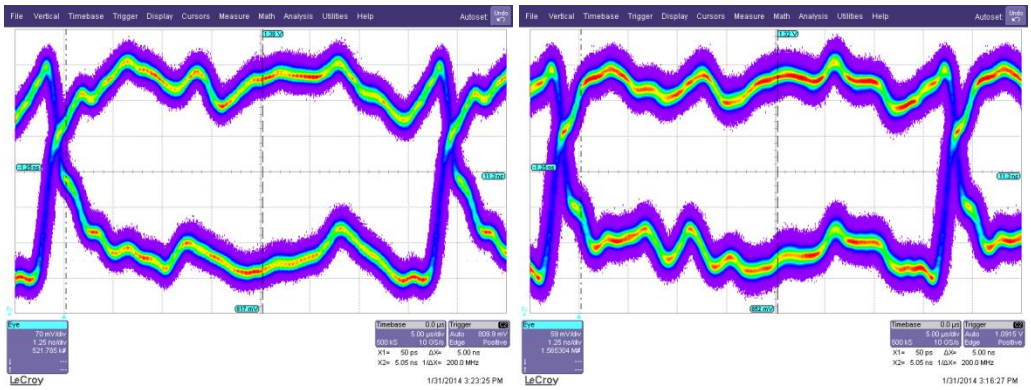


Fig. 66. Eye diagram of the trigger pair on the Feminos side.

Although signal reflections can be seen, both eyes show a correct opening.

13 PERFORMANCE ASSESSMENT

13.1 SUSTAINABLE EVENT RATE AND DATA THROUGHPUT

The sustainable event data taking rate is determined by several factors including the mode of operation (AFTER or AGET), the number of channels being readout (all channels or only hit channels), the number of SCA cells, the amount of data that remains when zero suppression is used, etc. The theoretical limit is the intrinsic maximum rate achievable by the AFTER or AGET chips assuming that subsequent stages have infinite speed and bandwidth. When the amount of data acquired is very small, data acquisition bandwidth does not influence the event taking rate. The limit is determined by the speed of the logic and the overhead of the acquisition chain. Reciprocally, for large events, the sustainable event rate is mostly determined by the transfer speed of data.

The lower limit on the event readout time achievable by the AFTER or AGET chip when all channels are readout is given by Equation (1).

$$T_{ASIC} = (N_{SCA} \times (N_{CH} + N_{RESET})) / F_{ADC} \qquad (1)$$

where F_{ADC} is the sampling rate of the ADC (25 MHz), N_{SCA} is the number of SCA cells (up to 511 with AFTER and 512 with AGET), N_{CH} is the number of "silicon" channels in the ASIC (76 for AFTER and 68 in AGET), and N_{RESET} is the number of "reset" channels (3 for AFTER and 2 or 4 in AGET depending on setting). The Feminos card digitizes the SCA of each of the four AFTER or AGET chips in parallel and the hardware design is optimal on this aspect.

The theoretical minimum event readout time and the measured values for the readout of all channels in the AGET mode (2 "reset" channels) and AFTER mode are plotted in Fig. 67. The theoretical calculation does not include the unavoidable overhead cycles of the logic state machine that controls the operation of the Feminos card. Consequently, the measured readout time deviates from the ideal curve for small values of SCA cells. However, for 512 SCA cells, the achieved rate reaches ~95% of the theoretical limit.

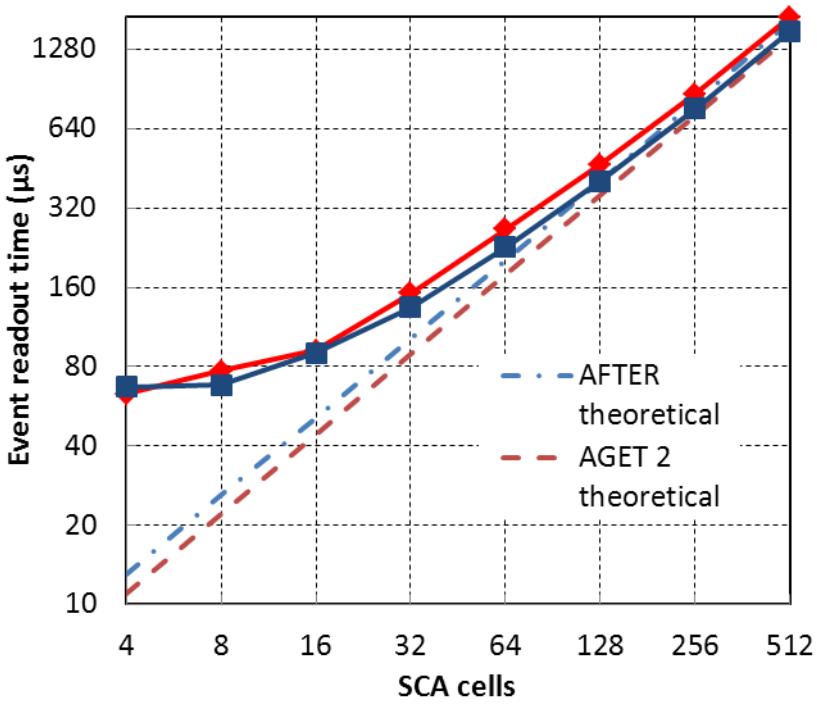


Fig. 67. Event readout time – all channel digitized; all data removed by zero-suppression.

For these measurements, event data are zero-suppressed and all thresholds are set to the maximum value. This produces minimal size events and ensures that the bandwidth of the gigabit Ethernet link to the DAQ PC does not play any role in the limit of the achievable event rate. The theoretical and measured sustainable event acquisition rates are plotted in Fig. 68.

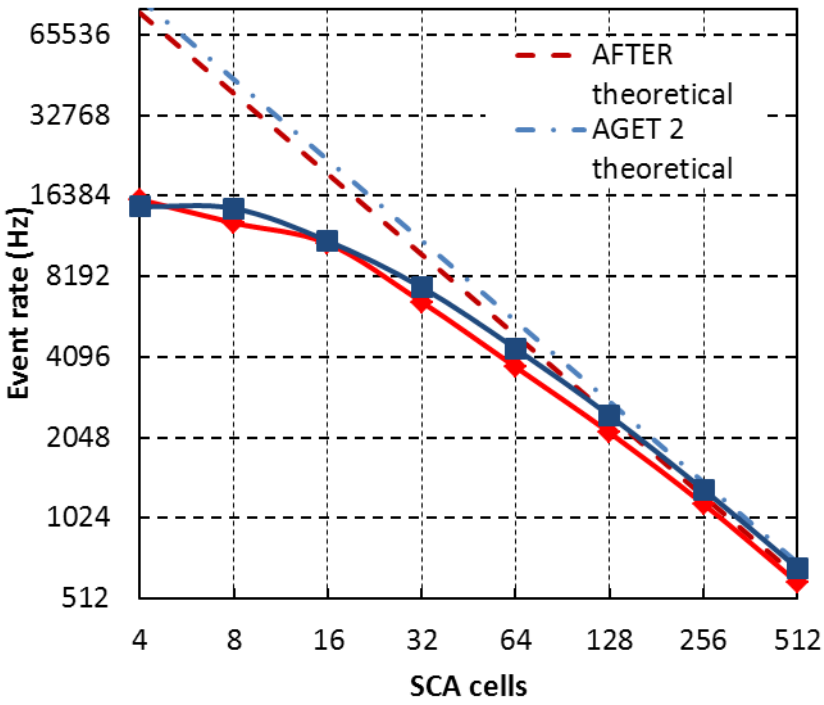


Fig. 68. Sustained event acquisition rate – all channel digitized; all data removed by zero-suppression.

To improve the sustainable event rate, the AGET chip allows reading only the channels that were hit. The Feminos card offers two different options in this mode: a first option is to perform the digitization of the AGET chip without altering the content of the Hit Channel register in the AGET chip. In this mode of operation, the readout of the Hit Channel Register is done in parallel with the digitization of the SCA. Hence, no latency is added by the readout of the Hit Channel Register. The second option is to readout and to alter the content of the Hit Channel Register prior to SCA digitization. In this case, the readout of the Hit Channel Register, modification, write-back and SCA digitization are performed sequentially. The FEC allows parallel readout of the Hit Channel Register of the four AGET chips, but only one Hit Channel Register can be written at a time. The performance of the Feminos card is therefore slightly degraded in this mode of operation because the write-back of the Hit Channel Registers takes four times the minimum duration that is reachable by an optimal design.

The minimum event readout time for an AGET chip when only hit channels are readout and the Channel Hit Register is not modified is given by Equation (2).

$$T_{AGET\_HIT} = (N_{SCA} \times (N_{CH\_HIT} + N_{RESET})) / F_{ADC} \quad (2)$$

where N_{CH_HIT} is the channel hit count of one AGET chip or the largest count of channel hit for this event within the 4 AGET chips controlled by the Feminos card. The

event readout time for the AGET chip in the mode where the Hit Channel Register is readout and modified before SCA digitization is given in Equation (3).

$$T_{AGET\_HIT\_MOD} = (N_{SCA} \times (N_{CH\_HIT} + N_{RESET})) / F_{ADC} + k \cdot N_{RDHIT} / F_{RDHIT} \quad (3)$$

where F_{RDHIT} is the frequency of the serial link to readout the Hit Channel Register (50 MHz), N_{RDHIT} is the number of clock cycles to readout the Hit Channel Register (typically 70) and k reflects the parallelism achievable to write-back these registers in the AGET chips (k=1 for an optimal design; k=4 for the FEC-Feminos implementation).

The theoretical and measured event readout time for a set of 4 AGET chips when only hit channels are digitized is shown in Fig. 69. For theoretical curves, we consider the case when the Hit Channel Register is unaltered, when all registers are altered in parallel, and when they are altered sequentially. Measurements can only be easily made when the Channel Hit Register is altered (so that an exact known number of hit channels can be set), and the Feminos card imposes that the modification of each register is made sequentially in each of the 4 AGET chips.

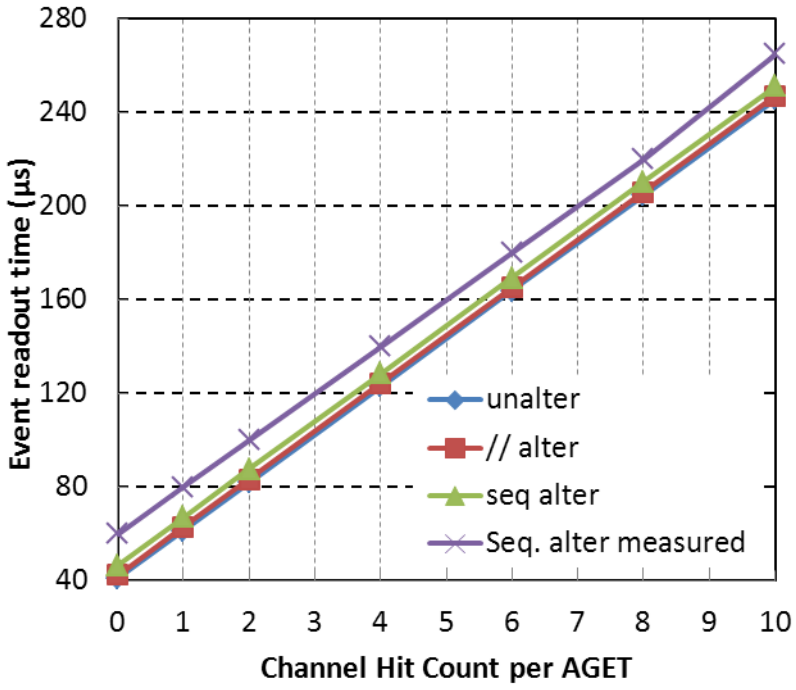


Fig. 69. Event readout time – only hit channel digitized (512 time-bins); all data removed by zero-suppression.

It can be seen that the performance difference between the three modes for processing the Hit Channel Register (unaltered, altered in parallel and altered sequentially) is very small (but it increases when the number of SCA cells to digitize is reduced). The measured readout time has an almost constant offset compared to the

theoretical value which corresponds to overhead cycles in the state machine of the Feminos cards that are not accounted for in the formulas. The event rate achieved by the Feminos card in these test conditions is from 75% to 95% of the theoretical value of the optimal implementation (parallel read and parallel write-back of Hit Channel Registers). The sustainable event rate that can be reached versus the number of channel hit count per AGET chip for a Feminos is shown in Fig. 70.

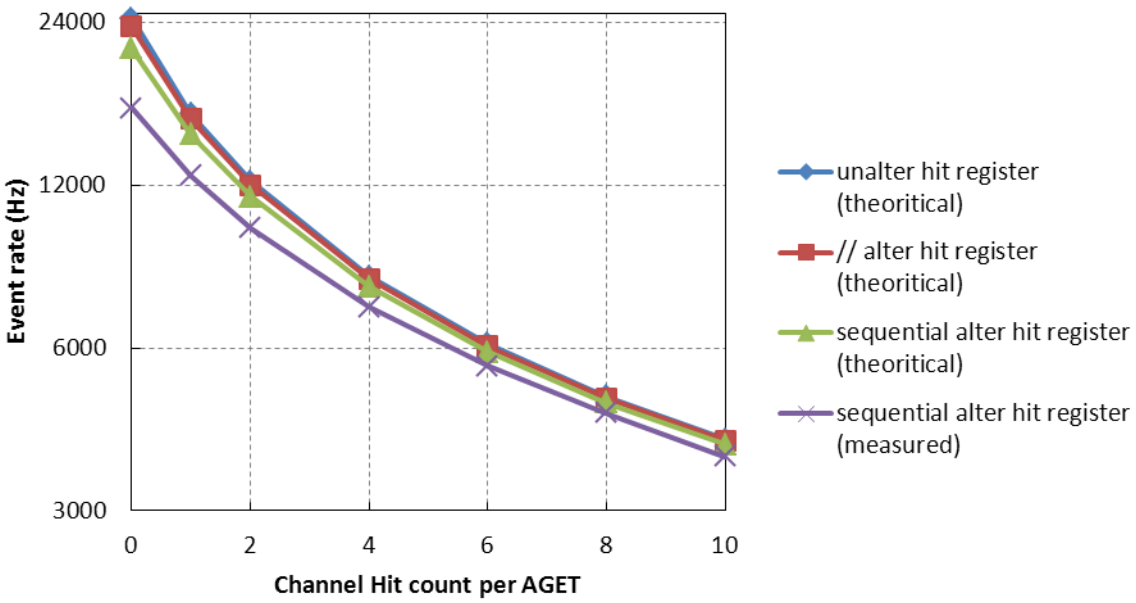


Fig. 70. Event rate versus channel hit count per AGET. 512 time-bins digitized. All data removed by zero-suppression stage.

When the amount of data sent by the Feminos card to the DAQ PC increases, the time required for the digitization of the SCA no longer sets the limit on the event rate, but it is determined by the bandwidth of the transfer path to the DAQ. The main limiting factors are: the speed of the SDRAM on-board the Feminos card (1.6 GByte/s peak theoretical), the speed of the AXI-4 DMA engine from and to this memory (400 MByte/s peak theoretical), and the bandwidth of the Gigabit Ethernet connection to the PC. Ideally, the element that has the lowest bandwidth in the chain is the Ethernet connection. A 1000 Mbps Gigabit Ethernet link carrying 8 KByte Jumbo frames with UDP/IP encoding has a theoretical maximum throughput of ~992 Mbps, i.e. ~124 MByte/s. On the other hand, the maximum sustained throughput at the level of the ADC of the FEC is 25 MHz x 12 bit x 4 ASICs = 150 MByte/s. The Feminos card is therefore not capable of sending full events at the best rate achievable by the ASICs. The operation of the DMA transfer from the FPGA to the external SDRAM of an un-compressed event is shown in Fig. 71. The size of the event is 283,752 bytes (i.e. 4 AGET chips x 69 channels x 512 time-bins). Each tick on signal IFIFO_WR indicates that

~8 KByte of event data (i.e. one Ethernet Jumbo frame) has been transferred to the external SDRAM. The average throughput during the complete event transfer is ~120 MByte/s.

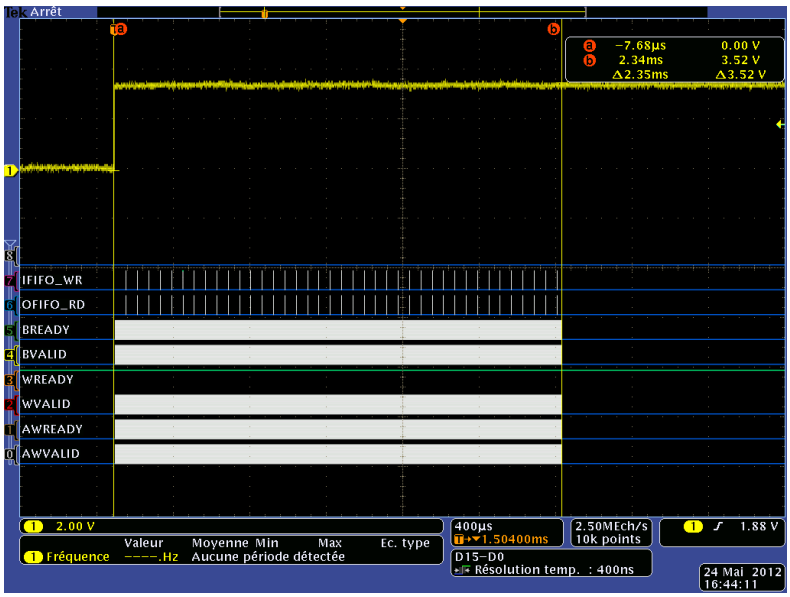


Fig. 71. DMA transfer of one full event from the FPGA logic to the external SDRAM of the Feminos.

A zoom on the transactions over the AXI-4 bus is shown in Fig. 72. Each tick on the AWVALID signal indicates the completion of a burst-of-four write transaction. Each tick on WVALID signal indicates one 32-bit transfer. The time required for each burst of 4x32-bit transfer is 130 ns (13 cycles at 100 MHz), i.e. a throughput of 123 MByte/s. This figure is consistent with what is observed at the scale of a complete event.

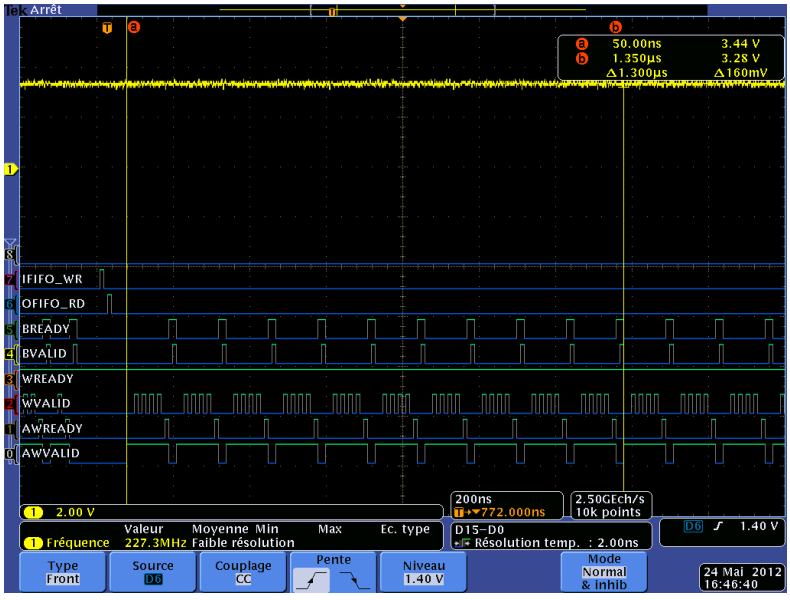


Fig. 72. AXI-4 transactions during DMA transfer of event data from the FPGA to the external SDRAM of the Feminos.

After Ethernet frame data have been moved from the FPGA to the external SDRAM, they need to be transferred by the DMA of the Ethernet controller from the external SDRAM to the Ethernet MAC and PHY. To avoid data under-run, this transfer must be done at Gigabit Ethernet wire-speed for each frame. However, there can be a large gap between frames. The speed of the DMA controller has not been measured, but is it reasonable to assume that full Gigabit Ethernet speed can be sustained permanently, i.e. ~124 MByte/s. It is not very clear how the embedded SDRAM controller of the Xilinx Spartan 6 arbitrates between the DMA write port from the FPGA to the external SDRAM and the DMA read port from the SDRAM to the Ethernet MAC. The plausible situation is that DMA transfers of both types do not overlap very much, i.e. the Ethernet send part only starts after the completion of the transfer of a complete event from the FPGA to the external SDRAM. Consequently, the throughput of the Feminos is limited to only half the speed of the slowest DMA transfer, i.e. ~60 MByte/s – which matches what is measured.

The amount of data sent by the Feminos card to the DAQ for each event when readout is made without zero suppression is given by Equation (4).

$$E_{NZS} = N_{ASIC} \times ((2 \times (N_{SCA} + N_{PRE}) + C_{HDR}) \times N_{CH\_RD}) + E_{HDR} \qquad (4)$$

where N_{ASIC} is the number of AFTER or AGET being readout (normally 4), N_{SCA} is the number of SCA cells being readout, N_{PRE} is the number of SCA bins to keep before threshold is passed, N_{CH_RD} is the number of channel readout (this normally excludes "reset" channels, non-instrumented channels and masked channels, i.e. it will be 72 in the AFTER mode and 64 or N_{CH_HIT} in the AGET mode), C_{HDR} is the per-channel header (4 bytes), and E_{HDR} is the size of the event header (24 bytes).

The measured sustained throughput of a Feminos – PC – local disk chain is ~57 MByte/s. This is above the bandwidth required to acquire complete event data of one AFTER or AGET chip without zero-suppression at the maximum speed reachable with these device, i.e. ~600 Hz. The sustained acquisition rate to disk for a 288-channel x 511 time-bin AFTER FEC without zero-suppression is ~200 Hz.

14 CHANNEL NUMBERING AND FEC PINOUT

The Feminos can read out a FEC equipped with up to four AFTER or AGET chips. Chips are numbered from #0 to #3 with chip #0 being the farthest from the green power supply connector of the FEC as shown on Fig. 73.

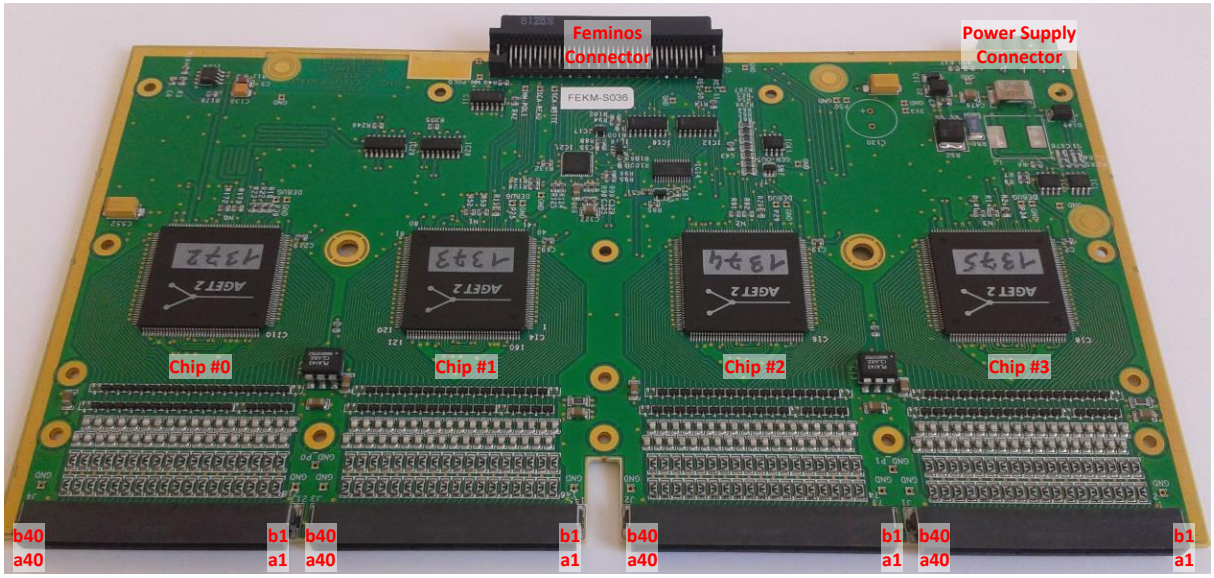


Fig. 73. FEC chip numbering and connector pinout.

The 72 (64) inputs of each AFTER (AGET) of the FEC are accessible via an 80-pin connector, ERNI 1.27 mm pitch, dual row, male, right-angled, part number 154767. Pins are numbered a1 to a40 for the row closest to the PCB and b1 to b40 for the superior row. Pin a1 and b1 of each connector are the closest to the green power supply connector of the FEC. The mating connector can be right angled (ERNI part number 154744) or straight (ERNI part number 154809).

The AFTER chip has 72 channels connected to physical package pins, 4 fixed pattern noise channels and 3 reset channels leading to a total of 79 logical channels. These are numbered from 0 to 78 in the data stream acquired from the Feminos. The correspondence between the channel index put in the data, the pin number of the AFTER chip and the FEC connector pin number is given in Table 28.

Table 28. DAQ Channel numbering vs. FEC connector pin numbering (AFTER).

DAQ Channel	AFTER channel	AFTER Pin#	FEC Connector Pin#
0	Reset 1	n/a	n/a
1	Reset 2	n/a	n/a
2	Reset 3	n/a	n/a
3	1	36	A3
4	2	35	B3
5	3	34	A4
6	4	33	B4

7	5	32	A5
8	6	31	B5
9	7	30	A6
10	8	29	B6
11	9	28	A7
12	10	27	B7
13	11	26	A8
14	12	25	B8
15	FPN1	n/a	n/a
16	13	24	A9
17	14	23	B9
18	15	22	A10
19	16	21	B10
20	17	20	A11
21	18	19	B11
22	19	18	A12
23	20	17	B12
24	21	16	A13
25	22	15	B13
26	23	14	A14
27	24	13	B14
28	FPN2	n/a	n/a
29	25	12	A15
30	26	11	B15
31	27	10	A16
32	28	9	B16
33	29	8	A17
34	30	7	B17
35	31	6	A18
36	32	5	B18
37	33	4	A19
38	34	3	B19
39	35	2	A20
40	36	1	B20
41	37	120	A21
42	38	119	B21
43	39	118	A22
44	40	117	B22
45	41	116	A23
46	42	115	B23
47	43	114	A24
48	44	113	B24
49	45	112	A25
50	46	111	B25
51	47	110	A26
52	48	109	B26
53	FPN3	n/a	n/a
54	49	108	A27
55	50	107	B27
56	51	106	A28
57	52	105	B28
58	53	104	A29
59	54	103	B29
60	55	102	A30
61	56	101	B30
62	57	100	A31
63	58	99	B31
64	59	98	A32
65	60	97	B32
66	FPN4	n/a	n/a

67	61	96	A33
68	62	95	B33
69	63	94	A34
70	64	93	B34
71	65	92	A35
72	66	91	B35
73	67	90	A36
74	68	89	B36
75	69	88	A37
76	70	87	B37
77	71	86	A38
78	72	85	B38

The AGET chip has 64 channels connected to physical package pins, 4 fixed pattern noise channels and 2 or 4 reset channels (programmable) leading to a total of 70 or 72 logical channels. These are numbered from 0 to 69 or 0 to 71 in the data stream acquired from the Feminos. The correspondence between the channel indexes put in the data, the pin number of the AGET and the FEC connector pin number is given in Table 29 and Table 30 for the AGET configured in the short and long read sequences respectively. Note than in both cases, there are 8 connector pins that no longer correspond to physical channels compared to a FEC populated with AFTER chips.

Table 29. DAQ Channel numbering vs. FEC connector pin numbering (AGET short read sequence).

DAQ Channel	AGET channel	AGET Pin#	FEC Connector Pin#
0	Reset 1	n/a	n/a
1	Reset 2	n/a	n/a
2	1	33	B4
3	2	32	A5
4	3	31	B5
5	4	30	A6
6	5	29	B6
7	6	28	A7
8	7	27	B7
9	8	26	A8
10	9	25	B8
11	10	24	A9
12	11	23	B9
13	FPN1	n/a	n/a
14	12	22	A10
15	13	21	B10
16	14	20	A11
17	15	19	B11
18	16	18	A12
19	17	16	A13
20	18	15	B13
21	19	14	A14
22	20	13	B14
23	21	12	A15
24	FPN2	n/a	n/a
25	22	11	B15
26	23	10	A16
27	24	9	B16

28	25	8	A17
29	26	7	B17
30	27	6	A18
31	28	5	B18
32	29	4	A19
33	30	3	B19
34	31	2	A20
35	32	1	B20
36	33	120	A21
37	34	119	B21
38	35	118	A22
39	36	117	B22
40	37	116	A23
41	38	115	B23
42	39	114	A24
43	40	113	B24
44	41	112	A25
45	42	111	B25
46	43	110	A26
47	FPN3	n/a	n/a
48	44	109	B26
49	45	108	A27
50	46	107	B27
51	47	106	A28
52	48	105	B28
53	49	103	B29
54	50	102	A30
55	51	101	B30
56	52	100	A31
57	53	99	B31
58	FPN4	n/a	n/a
59	54	98	A32
60	55	97	B32
61	56	96	A33
62	57	95	B33
63	58	94	A34
64	59	93	B34
65	60	92	A35
66	61	91	B35
67	62	90	A36
68	63	89	B36
69	64	88	A37
	n.c.	36	A3
	n.c.	35	B3
	C_lpol_csag	17	B12
	C_lpol_csag	34	A4
	C_lpol_csag	87	B37
	C_lpol_csag	104	A29
	Triggm	85	B38
	Triggp	86	A38

Table 30. DAQ Channel numbering vs. FEC connector pin numbering (AGET long read sequence).

DAQ Channel	AGET channel	AGET Pin#	FEC Connector Pin#
0	Reset 1	n/a	n/a
1	Reset 2	n/a	n/a
2	Reset 3	n/a	n/a

3	Reset 4	n/a	n/a
4	1	33	B4
5	2	32	A5
6	3	31	B5
7	4	30	A6
8	5	29	B6
9	6	28	A7
10	7	27	B7
11	8	26	A8
12	9	25	B8
13	10	24	A9
14	11	23	B9
15	FPN1	n/a	n/a
16	12	22	A10
17	13	21	B10
18	14	20	A11
19	15	19	B11
20	16	18	A12
21	17	16	A13
22	18	15	B13
23	19	14	A14
24	20	13	B14
25	21	12	A15
26	FPN2	n/a	n/a
27	22	11	B15
28	23	10	A16
29	24	9	B16
30	25	8	A17
31	26	7	B17
32	27	6	A18
33	28	5	B18
34	29	4	A19
35	30	3	B19
36	31	2	A20
37	32	1	B20
38	33	120	A21
39	34	119	B21
40	35	118	A22
41	36	117	B22
42	37	116	A23
43	38	115	B23
44	39	114	A24
45	40	113	B24
46	41	112	A25
47	42	111	B25
48	43	110	A26
49	FPN3	n/a	n/a
50	44	109	B26
51	45	108	A27
52	46	107	B27
53	47	106	A28
54	48	105	B28
55	49	103	B29
56	50	102	A30
57	51	101	B30
58	52	100	A31
59	53	99	B31
60	FPN4	n/a	n/a
61	54	98	A32
62	55	97	B32

63	56	96	A33
64	57	95	B33
65	58	94	A34
66	59	93	B34
67	60	92	A35
68	61	91	B35
69	62	90	A36
70	63	89	B36
71	64	88	A37
	n.c.	36	A3
	n.c.	35	B3
	C_lpol_csag	17	B12
	C_lpol_csag	34	A4
	C_lpol_csag	87	B37
	C_lpol_csag	104	A29
	Triggm	85	B38
	Triggp	86	A38

15 REFERENCE DOCUMENTS

[1] P. Baron et al. "AFTER, an ASIC for the Readout of the Large T2K Time Projection Chambers", IEEE Transactions on Nuclear Science, volume N°55, issue 3, part 3, pp. 1744 – 1752, June 2008.

[2] P. Baron and E. Delagnes, "AGET, a Front End ASIC for Active Time Projection Chamber", User Manual, Nov. 2010.

[3] Mars MX2 User Manual, Enclustra GmbH. Available Online: www.enclustra.com

16 DOCUMENT HISTORY

September 2011: initial release.

March 2012: changed encoding of configuration response frames; significant changes to encoding format – not backward compatible with preliminary versions.

March 2013: added description of pedestal and threshold lists and sensitivity curve of AGET discriminators. Updated list of prefix.

June 2013: added the "tstamp_init" command and its description. Updated mapping of Register #6.

July 2013: added "after <A> test_mode" command.

September 2013: added list of differences in file format for mclient event builder and Minos DAQ final event builder.

October 2013: documented the sequence number feature for the daq command. The description of the data frame is changed: the initial null word is replaced by the sequence number.

January 2014 (version 1.5): clarified the content of the field "Frame payload size" in data format. Added new modes of operation of the event builder. Corrected format of command statistics frame.

January 2014 (version 1.6): replaced incorrect measurements of SCA clock jitter by new measurements. Added section on measurements of signals from the TCM.

February 2014 (version 1.7): updated description of pulser, pulser control register, multiplicity threshold register, added commands related to multiplicity, added "aget x in_dyn_range" command.

March 2014 (version 1.8): corrected swap between push buttons P02 and P04.

March 2014 (version 2.1): removed single POLARITY bit from register #5. Added 4-bit POLARITY vector to register #3. Added argument <A> to polarity command. POLARITY can be selected on each ASIC independently starting from Feminos server version 2.1.

March 2014 (version 2.2): added "tcm ignore" command. Changed the way dead-time is measured by the Feminos. Modified zero-suppression logic to ignore the last four SCA cells when applying threshold. Changes apply starting from Feminos server version 2.2.

March 2014 (version 2.3): added "aget x icsa" command.

April 2014 (version 2.4): added commands "aget x cur_ra" and "aget x cur_buf"

June 2014 (version 2.5): added a field and command in minibios to define the power state of the FEC after the Feminos boots. Added a watchdog bit to determine if a Feminos had his time stamp initialized. Added bit SND_MULT_ENA in register #3 and the command "snd_mult_ena". Revised description of multiplicity frames sent from the Feminos to the TCM.

June 2014 (version 2.6): added configuration registers #8 and #9. Changed generation of multiplicity-over-threshold bits. Moved TIME_PROBE from bit 30 of Register #6 to bit 31 of Register #1. Mapped TCM_BERT to bit 30 of Register #6. Added logic support for bit error tester mode. Added commands "mult_limit" and "tcm_bert". Added section on multiplicity trigger. Updated the screenshot of minibios.

June 2014 (version 2.7): Mapped ERASE_HIT_ENA to bit 15 of Register #3 and ERASE_HIT_THR to bit 20 to 31 of Register #3. Added commands "erase_hit_ena" and "erase_hit_thr".

October 2014 (version 2.7): Added command "mars set clock" for mclient.

November 2014 (version 2.8): Added command "tstamp_isset" in documentation. No change in firmware and embedded software but re-compilation was done.

February 2015 (version 2.9): Software adaptations to port TCM software to Mars AX3 (Artix 7) and Mars ZX3 (Zynq). No change to Feminos functionality.

March 2015 (version 2.10): Dead-time histogram has 4 resolution settings. Mapped BUSY_RESOL to bit 23 and 22 of Register #5. Added commands "busy_resol".

June 2015 (version 2.11): increased range of WCK_DIV to [1; 255]. Remapped BUSY_RESOL to bit 1 and 0 of Register #9.

June 2015 (version 2.12): increased the range of TRIG_DELAY to 1.3 ms. Now exclude the first 4 time-bins of each channel when accumulating pedestal histograms. Clarified section for using the pulse generator of the FEC. Added picture of the FEC with labels for chips and connectors.