ProDAQ

Hardware Manual

ProDAQ 3808
100MHz Counter/Timer
Function Card

PUBLICATION NUMBER: 3808-XX-HM-0100

bustec

Bustec Production, Ltd.
World Aviation Park, Shannon, Co. Clare, Ireland
Tel: +353 (0) 61 707100, FAX: +353 (0) 61 707106

PROPRIETARY NOTICE

This document and the technical data herein disclosed, are proprietary to Bustec Production Ltd., and shall not, without express written permission of Bustec Production Ltd, be used, in whole or in part to solicit quotations from a competitive source or used for manufacture by anyone other than Bustec Production Ltd. The information herein has been developed at private expense, and may only be used for operation and maintenance reference purposes or for purposes of engineering evaluation and incorporation into technical specifications and other documents, which specify procurement of products from Bustec Production Ltd.. This document is subject to change without further notification. Bustec Production Ltd. Reserve the right to change both the hardware and software described herein.

Table of Contents

Table of figures

1. Installation

1.1. Unpacking and Inspection

The ProDAQ module is shipped in an antistatic package to prevent any damage from electrostatic discharge (ESD). Proper ESD handling procedures must always be used when packing, unpacking or installing any ProDAQ module, ProDAQ plug-in module or ProDAQ function card:

- Ground yourself via a grounding strap or similar, e.g. by holding to a grounded object.
- Discharge the package by touching it to a grounded object, e.g. a metal part of your VXIbus chassis, before removing the module from the package.
- Remove the ProDAQ module from its carton, preserving the factory packaging as much as possible.
- Inspect the ProDAQ module for any defect or damage. Immediately notify the carrier if any damage is apparent.

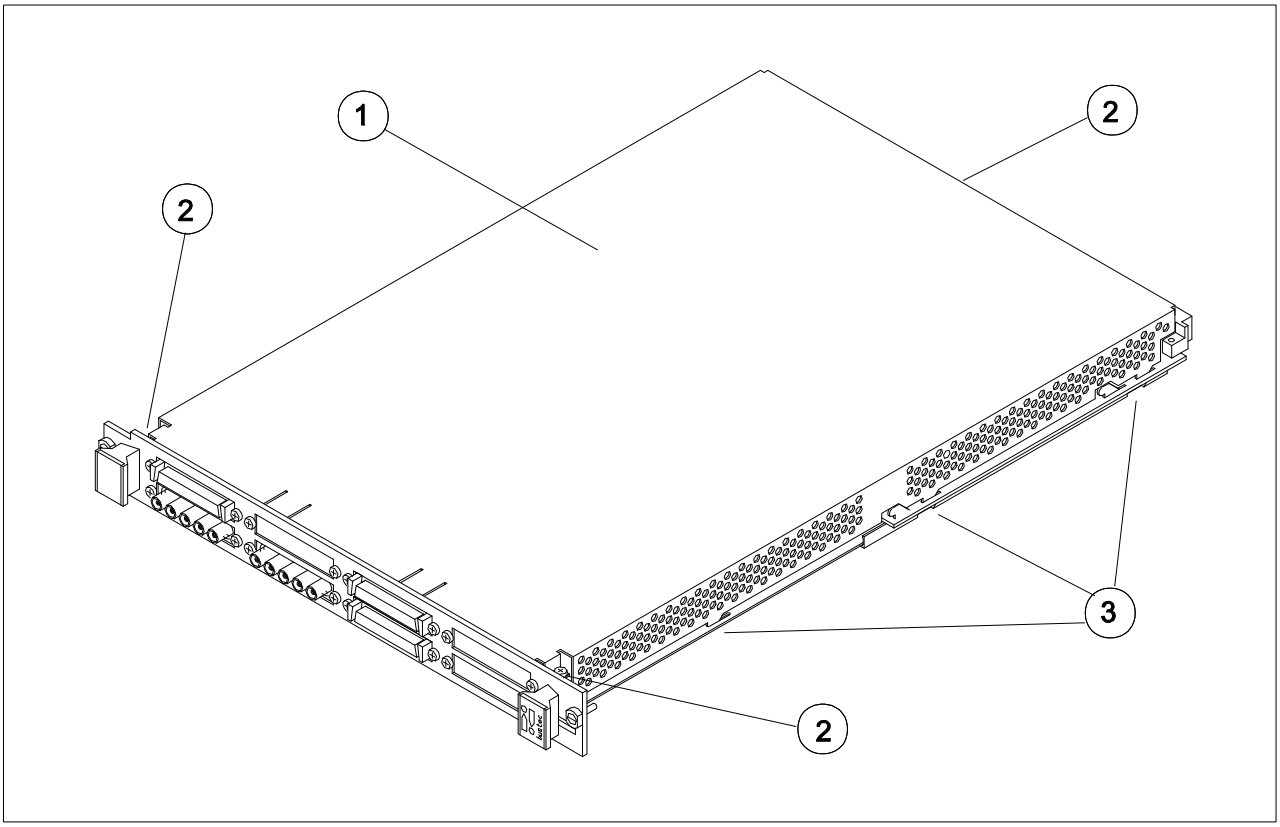
1.2. Reshipment Instructions

Use the original packing material when returning a ProDAQ module to Bustec Production Ltd. for calibration or servicing. The original shipping carton and the instrument's plastic foam will provide the necessary support for safe reshipment.

If the original anti-static packing material is unavailable, wrap the ProDAQ module in anti-static plastic sheeting and use plastic spray foam to surround and protect the instrument. Reship in either the original or new shipping carton.

1.3. Preparing the ProDAQ Module

To install a ProDAQ function card into one of the ProDAQ motherboards, you need to remove the module's top cover:



1 - Module Cover 2 - Cover Screws 3 - Cover Hooks

Figure 1 – Removing the ProDAQ module cover

To remove the top cover, remove the one countersunk screw in the back and the two panhead screws towards the front panel (②), that hold the cover in place. Remove the cover by sliding it out of its position towards the VXIbus connectors and up. Take special care about the hooks (③) holding it in place. Try not to lift the cover straight up. See Figure 1 for the location of the screws.

To re-install the cover, slide it back into its position by placing the small hooks over their holes and moving the cover down and forward. Secure the top cover using two panhead screws and one countersunk screw (②).

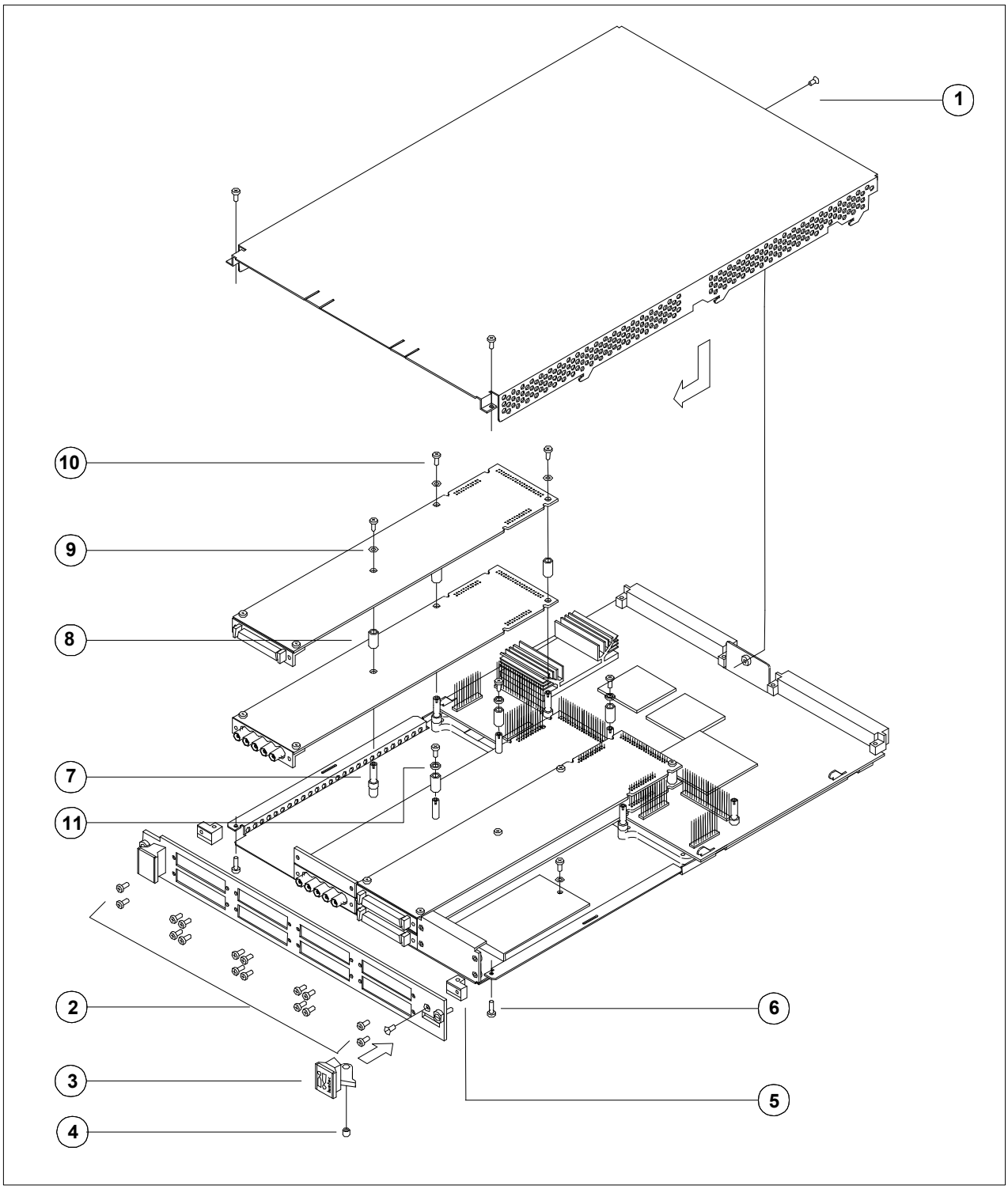
1.4. Installing a ProDAQ Function Card

The single-width ProDAQ function cards are arranged inside the ProDAQ module in four stacks of two cards each. The double-width ProDAQ function cards are arranged inside the ProDAQ module in two stacks of two cards each. The function cards are mounted face down, e.g. the front-panel connectors as well as the motherboard connectors are underneath the PCB. Single-width and double-width ProDAQ function cards can be mixed in the ProDAQ module. The 3424 function card is a double-width card.

To install a single-width ProDAQ function card in any of the possible positions, use the following procedure (See Figure 2 for reference):

- Remove the top cover of the module as described earlier in this chapter (Fig. 2, Pos. 1).
- Remove all screws on the front-panel holding installed function cards or double filler panels in place (Fig. 2, Pos. 2). Screws holding single filler panels don't need to be removed.
- Remove the two panhead screws that mount the front panel to the modules bottom cover (Fig. 2, Pos. 6).
- Please take special care of the module handles and the rings (Fig. 2, Pos. 3 and 4), which are also fixed by those screws. The mounting angle (Fig. 2, Pos. 5) stays fixed to the front panel.
- Remove the front panel by moving it forward carefully so as to avoid bending the installed function cards.
- Choose the stack and position (lower or upper) where you want to mount the function card. If the stack, in which the function card should be installed, is covered by a double filler panel, you have to remove it before installing the function card.
- Remove the three 2.5mm panhead screws and the crinkle washers from the stack's standoffs (Fig. 2, Pos. 9 and 10 for example).
- If you want to install a function card in the upper position of a stack without having a function card in the lower position, you need to mount both spacers (Fig. 3, Pos. 11) on each standoff. If the stack is already populated with a function card in the lower position, mount only the bigger spacer (Fig. 2, Pos. 8) onto each standoff.
- Place a bayonet (supplied) on each standoff. Align the function card over these and slide carefully down. The function card should be held parallel to the modules bottom cover all the time during its way down.
- Fix the function card by mounting the three 2.5mm panhead screws and the crinkle washers onto each standoff. If you install a function card in the lower position of a stack, you need first to mount both spacers (Fig. 2, Pos. 11) onto each standoff.
- Re-mount the modules front-panel. If there is only one function card mounted in a stack, cover the remaining opening in the front panel by a single filler panel.
- Re-mount the modules top cover.

Adjust the procedure respectively for a double-width ProDAQ function card.



1 - 2.5mm Panhead Screws
2 - 2.5mm Panhead Screws
3 - Module Handle
4 - Ring
5 - Mounting Angle
6 - 2.5mm Panhead Screws
7 - Standoff
8 - Spacer
9 - Crinkle Washer
10 - 2.5mm Panhead Screw
11 - 2mm Spacer

Figure 2 – The ProDAQ module assembly

1.5. Removing a ProDAQ Function Card

Removing a ProDAQ function card is exactly the reverse operation then installing it. After removing the top cover and the front panel as described previously, remove the three roundhead screws that fix the function card(s) on the standoffs.

Take special care when removing the function card(s) not to bend the motherboard connectors.

After removing the function card(s), install the correct combination of spacers on the standoffs. If a stack is populated with only one function card, each of the standoffs needs to be mounted with both spacers to cover the distance between the cards as well as the PCB thickness of the missing card. If a stack is populated with two function cards, only the bigger spacer must be mounted.

Fix any remaining function cards again by mounting the three panhead screws on the standoffs, re-mount the front panel and the modules cover.

2. Theory of Operation

2.1. Overview

The ProDAQ 3808, an 8-channel Counter / Timer Function Card, provides the following functions:

- Pulse counting – every channel counts incoming pulses within a specified time defined by a gate signal. The gate signal is common to all channels and can be either external or internal. The width of an internally generated gate signal is software selectable from 400ns to 1717s.
- Frequency measurement – as a result of the pulse counting for known gate width.
- Time interval measurement – every channel is capable of measuring time intervals between rising edges, falling edges or consecutive edges of an input signal. If the FIFO is emptying at the end of measurement, up to 512 samples per channel can be stored in an on-board memory. A “read on-the-fly” mode allows data transfer to the host during the measurement thus yielding a substantial increase in the number of time tags per channel. The time interval measurements can be initiated by a trigger signal. The 3808 can accept either one common or 8 independent external trigger signals.
- Pulse width, period and duty cycle measurements – as a result of the time interval measurement capabilities.
- Time interval, pulse width, period and duty cycle averaging.

Every channel of the 3808 Counter Timer Function Card has two different counters: 32-bit pulse counter and 24-bit time interval counter, allowing simultaneous operation as a pulse and a time interval counter. In the Pulse Counter incoming signals of up to 25 MHz can be processed on each of the eight channels.
A time interval counter range is divided into six sub-ranges covering times from 40ns to 20000s with a maximum resolution of 10ns.
The pulse and time interval counters are globally enabled by the gate signal, which can be generated either internally, externally or by software. In addition, the trigger signal (common or dedicated to every channel) can be used to gate the time interval measurements. Two modes of operation: a window mode, in which the trigger enables measurements for the time duration of the trigger pulse and a launch mode, in which the trigger starts the measurements. Each mode can be programmed independently on a channel-to-channel basis.
Input signals can be either DC or AC coupled with 1MΩ or 50Ω terminations. Every channel has a dedicated 10-bit digital-to-analog converter to set a threshold within the full-scale input signal range of ±5V.

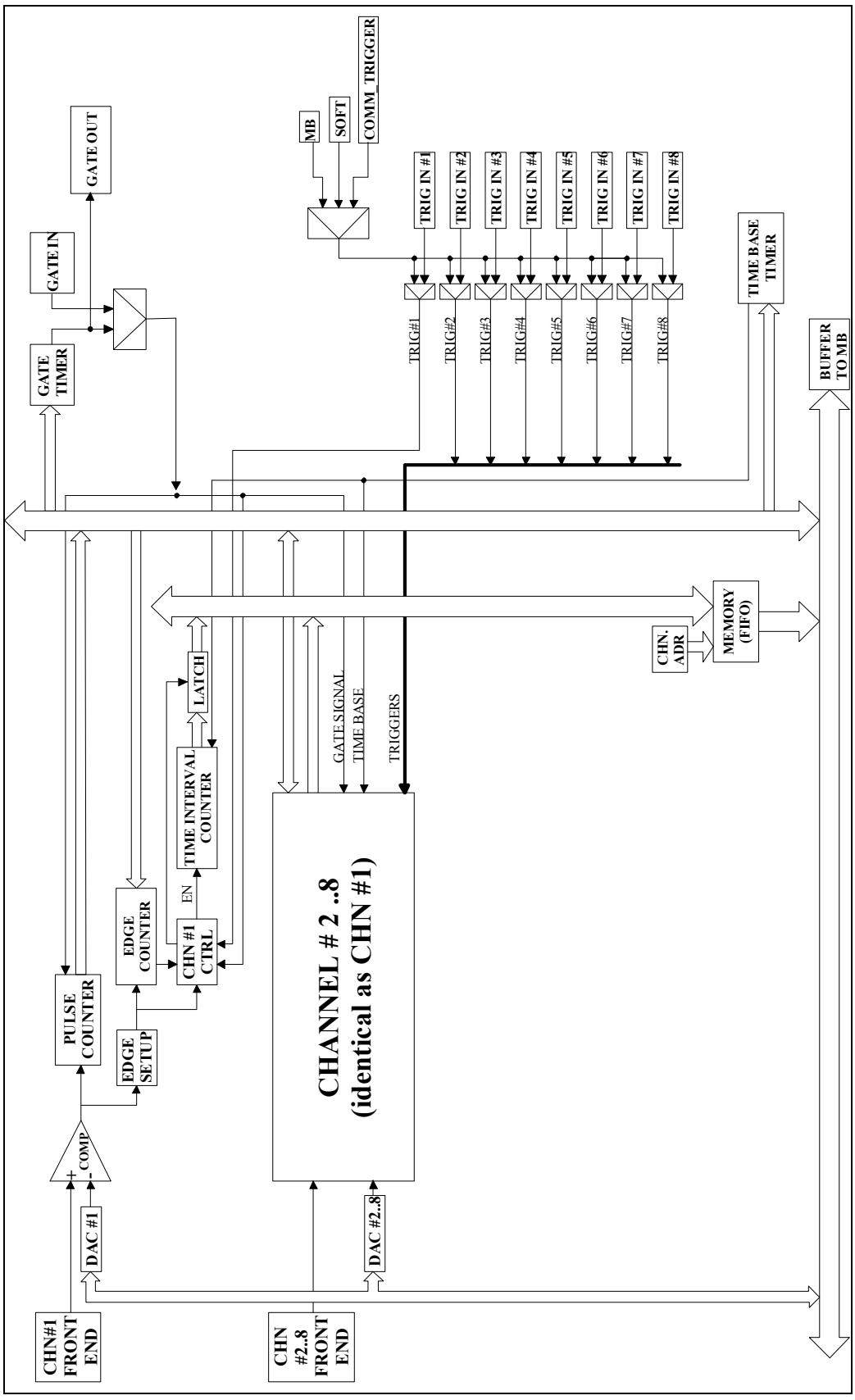


Figure 3 - Block Diagram

2.2. Details

PULSE COUNTER counts input signal pulses if the gate is on. The pulse counter can be programmed to count rising edges or falling edges. The full revolution of the counter is treated as an error and the information about that event is latched in the appropriate register. The pulse counter is 32-bit wide.

TIME INTERVAL COUNTER counts pulses coming from the Time Base generator (a programmable reference clock). When started the time interval counter counts these pulses continuously and stores the current value of the counter into FIFO when an event happens. An event is an edge of input signal and is software selectable between rising, falling or consecutive edges. The latched data is then stored in memory.
The time interval counter can be started either by the gate signal or the trigger. In the triggered mode the actual start of the counter can be synchronised to the edge of the input signal. The time interval counter continues its operation until one of the following is reached:

- a pre-set number of measurements has been taken if working in limited mode,
- the end of the trigger window signal if appropriate mode was set,
- the end of the gate signal.

EDGE COUNTER counts down the events (edges) of the input signal. This is a loadable counter and can be programmed to react on either rising, falling or every edge of the input signal. Terminal count of this counter indicates that the pre-set number of measurements has been taken and is used to stop the time interval counter (if in limited mode).

TIME BASE TIMER generates the clock for the time interval counters. It can be programmed to generate frequencies from 1000Hz to 100MHz.

GATE GENERATOR generates the internal gate signal. The width of the gate is in range from 400ns to 1717s. The internal gate becomes active either upon a software command or upon receiving a trigger signal supplied to the external gate input.

MEMORY is used to store the values from time interval counters when the defined edge of the input signal happens. The latched time interval counter's data and the channel's address are merged together and stored in a FIFO.

2.2.1. Acquisition Control

The 3808 card is controlled by the finite state machine (FSM). The FSM can be in one of three states:

- ACCESS_state (idle)
- ARMED_state
- COUNTING_state

ACCESS_state — This is the state after reset and in fact it is an idle state. The configuration and set-up of the card should be done in this state. The counting is disabled. The exit from this state is done using the Arming Command (write to COMMAND_REG).

ARMED_state — The card is armed. This means the configuration is finished and the card is waiting for gate signal to start counting. The edge of gate signal is required to go to the COUNTING_state. Return to ACCESS_state can be performed using FSMreset bit.

COUNTING_state — The enabled counters count until Gate signal is on. The counting is allowed only in this state. Return to ACCESS_state takes place when Gate becomes inactive or FSMreset bit is set or ERROR happened and stoping on errors had been previously enabled. When going to ACCESS_state the COUNTING_END bit is set.

The current state of the 3808 state machine can be read from FCCTRL_REG.

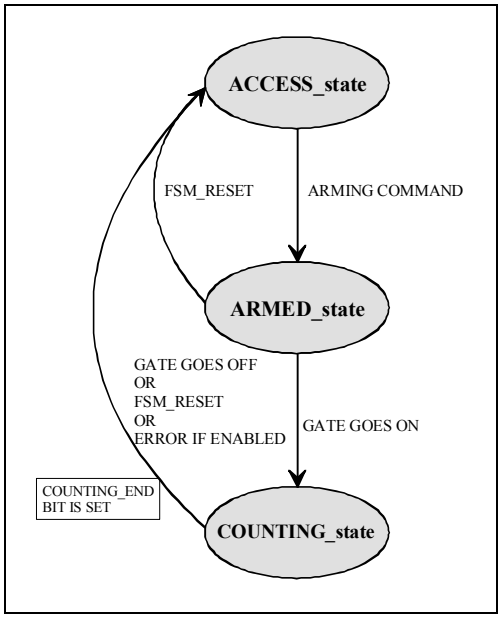


Figure 4: States of the FSM

Before the data acquisition can be started the 3808 has to be set-up properly. Upon a software command the card is armed and ready to acquire a gate signal, which determines the time of measurement.

If the card goes to ARMED_state while the gate is already ON this gate will not be accepted and the card will not go to COUNTING_state until the selected edge of gate happens. This is because the gate circuitry is edge-triggered, which means it requires an edge for correct operation.

The gate signal enables the pulse counters. The gate signal is necessary for the time interval counters and the TICNTs count only when gate is on. The start of the TICNTs depends on selected mode of operation.

The gate signal source can be selected either as an external, internal generator or software. Internal gate timer can be initiated either by software or an edge of a pulse applied to the Gate Input.
When the gate goes into its inactive state all the counters are disabled. Gate signals will be rejected during ACCESS_state and will not be accepted until the next arming occurs.
Gate signal possible selections are as follow:

GATE		
External	**Internal, programmable pulse**	**Software**
	Software initiated / Triggered from external signal applied to Gate In	

The gate signal is common for all channels but the modes of operation can be set for every channel independently.

2.2.2. Pulse counters

In every channel there is a pulse counter. It counts the number of edges of input signal that come when the gate is on. The edge direction that is counted is software selectable to be rising or falling.
The pulse counter is 32-bit wide. It can count the pulseswith a frequency up to 25MHz.
The Figure 5 shows the explanation of the pulse counter operation. It assumes that the pulse counter was configured to react to (to count) rising edges.

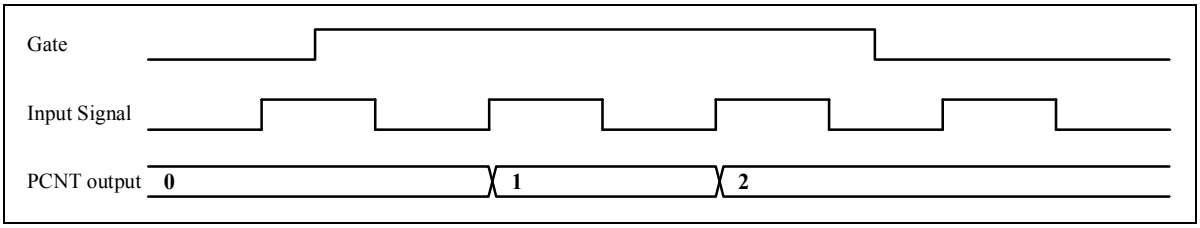


Figure 5: The operation of the pulse counter

The output of the pulse counters is 32-bit wide and has to be read in two steps from the PCNTx_REG (where x is the channel number), as a lower and upper 16-bit word. The selection of the word to read is controlled by PCNT_UPWORD bit.
If the counter performed the full revolution the PCNT error will be reported.
The pulse counter can work simultaneously to the time interval counter.

2.2.3. Time interval counters

In every channel there is a time interval counter. When started, the time interval counter counts the number of Time Base pulses. It counts continuously until it is stopped and therefore many full revolutions can happen. Periodically the momentary counter value is latched and written to the FIFO memory, this is called an event. The event can be defined as a rising edge, falling edge or every edge of the input signal.
There are a few ways of starting the time interval counter and they are presented in the table below.

TICNT STARTED AFTER GATE GOES ACTIVE		**TICNT STARTED AFTER TRIGGER (DURING GATE ACTIVE)**			
Immediately after gate (asynchronously to the input signal)	**After gate on first selected edge of input signal (synchronously to the input signal)**	**Immediately after trigger (asynchronously to the input signal)**		**After trigger on first selected edge of input signal (synchronously to the input signal)**	
		Launch mode	Window mode	Launch mode	Window mode

The TICNT can be configured to start when gate signal goes on. If the asynchronous mode was selected (immediately after starting signal) then the first value latched on the first event will be the interval between the gate edge and the first event. The next values stored in FIFO will be the interval between the events. If the synchronous mode was selected then all values will represent the interval between the events.

The TICNT can be configured to start on trigger when gate is on. In this case synchronous/asynchronous mode can be set as well. For the asynchronous mode the first value stored will represent the interval between the trigger and first event. In addition, for the trigger mode the launch or window mode can be selected. The launch mode means that the trigger starts only the counting of the TICNT. The window mode means that as long as trigger is on the counting take place and events are accepted.

There are eight time interval counters on the 8-channel board and the samples form the TICNTs are latched on the event and then written to the FIFO. There is only one FIFO memory common for all channels and the samples from all channels are stored there. Of course the arbitration scheme was applied to it. The highest priority is assigned to channel #1, the lowest to channel #8. If there is a request of writing to FIFO set on all channels simultaneously channel #1 will be serviced as the first. It can happen in any channel that while waiting for the write to FIFO, new value for this channel is latched on the event overwriting the previous one. It is called as an OVERWRITE error and is indicated on the bit in FCCTRL_REG.

The minimum time interval allowed to measure simultaneously without OVERWRITE error happening depends on the number of channels taking part in data acquisition. The following equation gives the minimum time interval in every channel without OVERWRITE error happening, assuming that the same signal was applied to all of them.

$$MIN\_TIME\_INTERVAL = 2 * Number\_of\_TICNTs * 12.5ns$$

In any case the measured time interval cannot be less than 40ns.

Number of enabled time interval counters	MIN. Time Interval applied to the channels [ns]
1	40
2	50
3	75
4	100
5	125
6	150
7	175
8	200

The FIFO memory can be read on-the-fly thus allowing unlimited number of samples to be collected. If the FIFO is not emptied on-the-fly the number of samples coming from the channels has to be limited to the FIFO size, that is to 4096 samples in total. Otherwise the OVERWRITE error can happen when FIFO is full (when FIFO is full writing to the FIFO is disabled until there is any place in the FIFO). To limit number of samples the LIMITED mode has to be set and number of samples has to be specified.
Once started the time interval counter will count until one of the following events happens:

STOPPED WHEN:	REMARKS
Gate becomes inactive	Unconditional ends of counting
Pre-set number of samples was taken	In limited mode only
Trigger becomes inactive	In window mode only, immediately (asynchronously) or synchronised to input signal event

The trigger for the time interval counters can be either common to all channels or independent for each channel. The independent triggers can only be provided by an external source. The common trigger can be one of the following: software, coming from the MB or an external source.
Trigger source for time interval counters:

TRIGGER INDEPENDENT FOR EVERY CHANNEL	TRIGGER COMMON FOR ALL CHANNELS
External	External
	Software
	Coming from the MB

2.3. Examples

2.3.1. Example 1

TI counter has been configured to start immediately after gate is on (asynchronous start).

The events have been defined as a rising edge.

Number of samples was set to 3 in limited mode but for given gate length and input signal timing the same result can be got with unlimited mode.

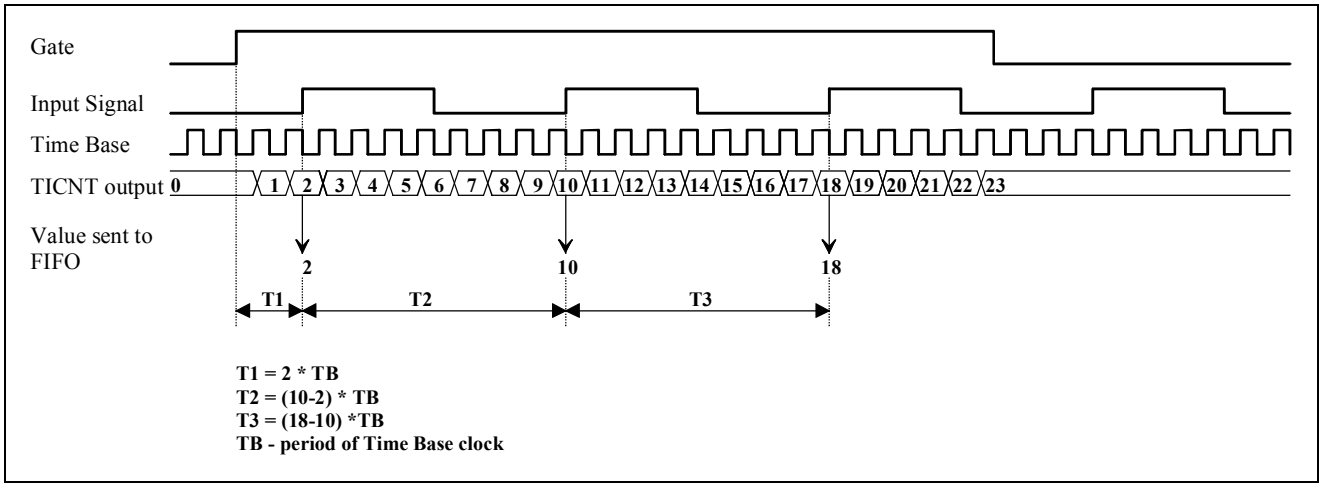


Figure 6: TICNT's modes of operation - example 1

As a result three values will be stored in FIFO: 2, 10 and 18.

First value represents the time interval between Gate edge and first event (rising edge) of input signal: T1 = 2 * TB.
Every next value has to be processed to get the time interval between two consecutive events (in this case the time interval between two rising edges and in fact period of input signal):

T2 = (10 - 2) * TB = 8 *TB

T3 = (18 – 10) * TB = 8 * TB

where TB is a period of Time Base clock, software selectable.

2.3.2. Example 2

TI counter has been configured to start after gate is on, synchronously to the input signal event.
The events have been defined as a falling edge.
Limited mode has been selected and number of samples has been set to 1.

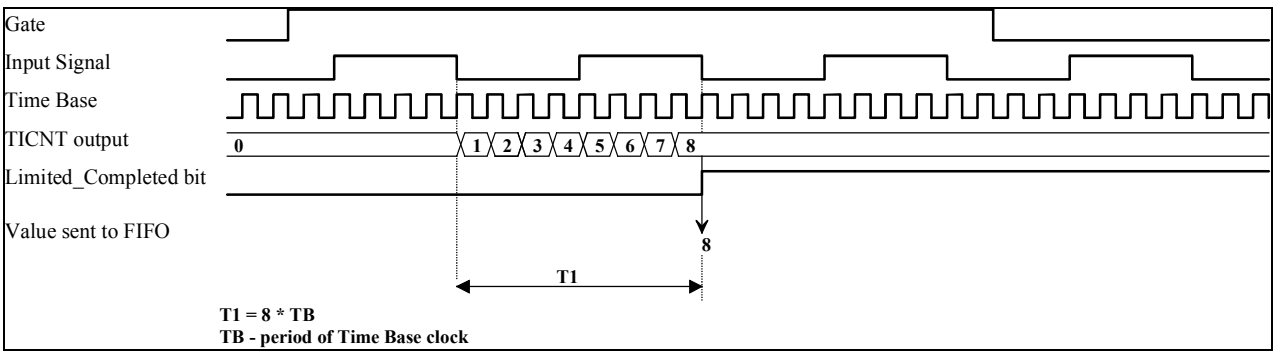


Figure 7: TICNT's modes of operation - example 2

As a result one value will be stored in FIFO: 8.

This value represents the time interval between first event (falling edge) of input signal and second event (falling edge) of the input signal: T1 = 8 * TB.
After first value is loaded to the FIFO the TI counter is stopped and no more samples will be stored in the FIFO (coming from this TI counter).

2.3.3. Example 3

TI counter has been configured to start after trigger (when the gate is on), synchronously to the input signal event. The trigger works in window mode.
The events have been defined as an every edge with the falling edge as first.
Limited mode has been selected and number of samples has been set to 3 but as shown on Figure 8 limit wasn't reached.

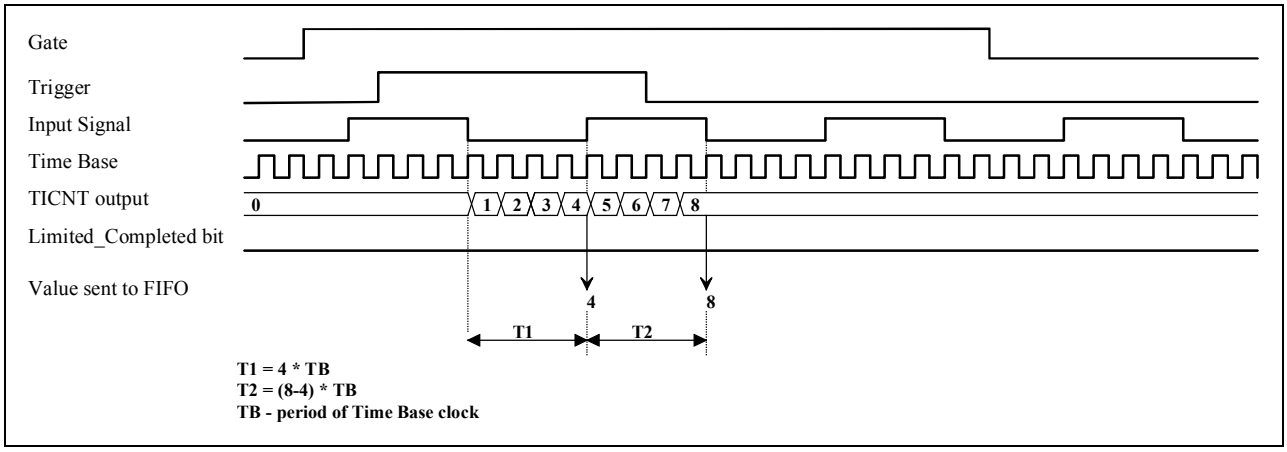


Figure 8: TICNT's modes of operation - example 3

As a result two values will be stored in FIFO: 4 and 8.

The first value represents the time interval between first event (falling edge) and second event (rising edge): T1 = 4 * TB.
The second value represents the time interval between second event (rising edge) and third event (falling edge): T2 = (8 – 4) * TB.
In fact T1 value is a negative pulse width and T2 is a positive pulse width.
The TICNT was started when gate and trigger were on and was synchronised to input signal edge.
The TICNT was stopped when trigger went off and stopping was synchronised to selected input signal edge.
The limit, set to 3, wasn't reached but TI counter was stopped because of the window trigger mode.

2.3.4. Example 4

TI counter has been configured to start immediately after trigger (when the gate is on), asynchronously to the input signal event. The trigger works in launch mode.
The events have been defined as an every edge with the falling edge as first.
Limited mode has been selected and number of samples has been set to 3.

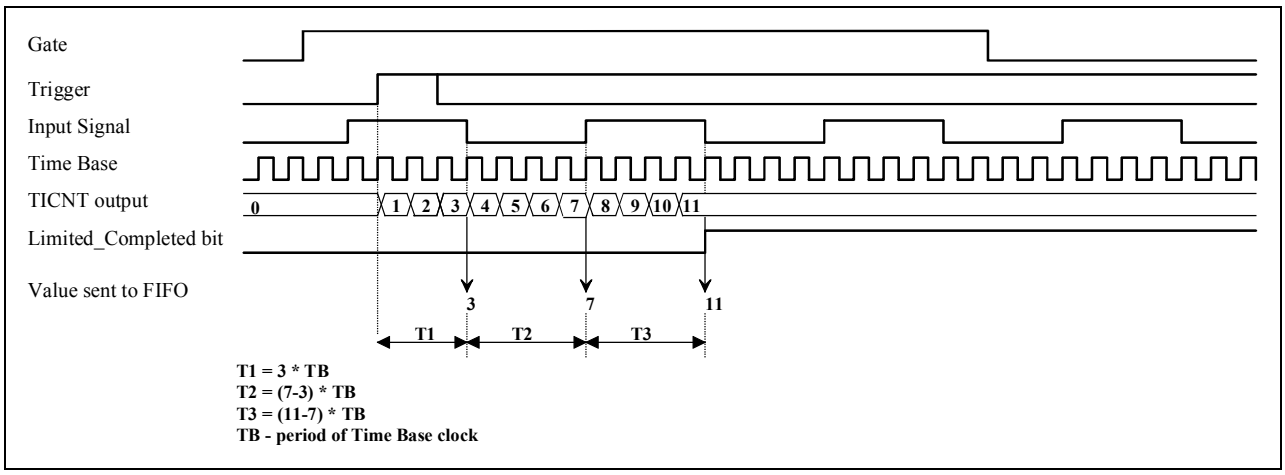


Figure 9: TICNT's modes of operation - example 4

As a result, three values will be stored in FIFO: 3, 7 and 11.

The first value represents the time interval between starting edge of trigger and first event (falling edge): T1 = 3 * TB.
The second value represents the time interval between first event (falling edge) and second event (rising edge): T2 = (7 – 3) * TB.
The third value represents the time interval between second event (rising edge) and third event (falling edge): T3 = (11 – 7) * TB.
In fact T1 value is the time interval between trigger and first event, T2 is a negative pulse width and T3 is a positive pulse width.
The TICNT was started when gate and trigger were on, immediately after trigger (asynchronously to input signal edge). The TICNT was stopped after set number of samples was collected because of the limited mode.

2.3.5. Example 5

TI counter has been configured to start after the gate is on, synchronously to the input signal event. The events have been defined as a rising edge.
Unlimited mode has been selected.

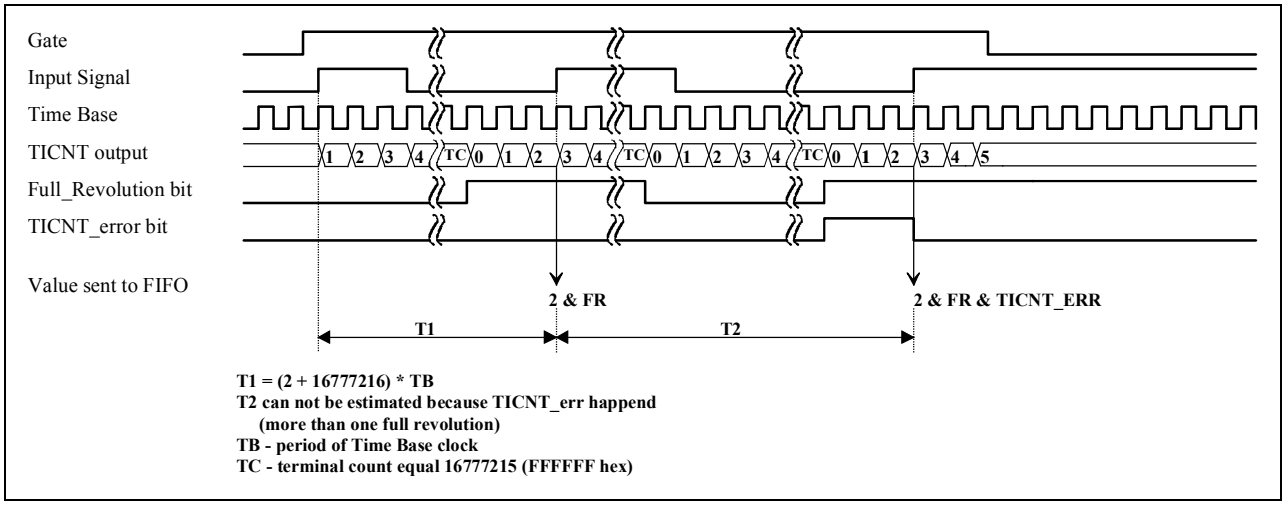


Figure 10: TICNT's modes of operation - example 5

As a result two values will be stored in FIFO: (2 & FR) as a first and (2 & FR & TICNT_ERR) as a second.

The first value represents the time interval between the first and second rising edge and because of the full revolution that was made this time interval has to be calculated as follow:

T1 = (2 + 16777216) * TB.

The second value represents the time interval between the second and third rising edge but because of the fact that full revolution happened more than once the TICNT_error bit was set and the T2 can not be estimated (the hardware allows to handle the situation when the full revolution happens no more than once).

2.4. Hardware configuration

2.4.1. Front end configuration

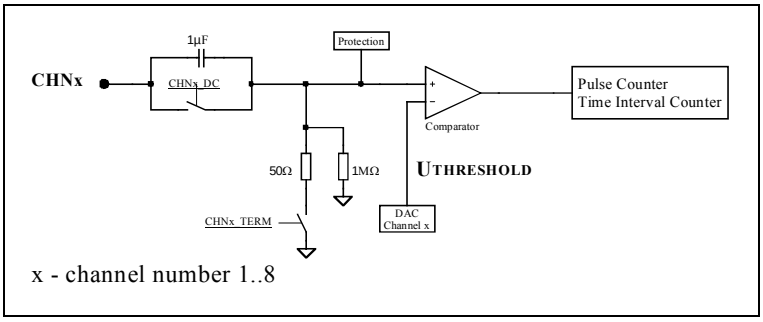


Figure 11: Front End configuration

2.4.2. GATE selection

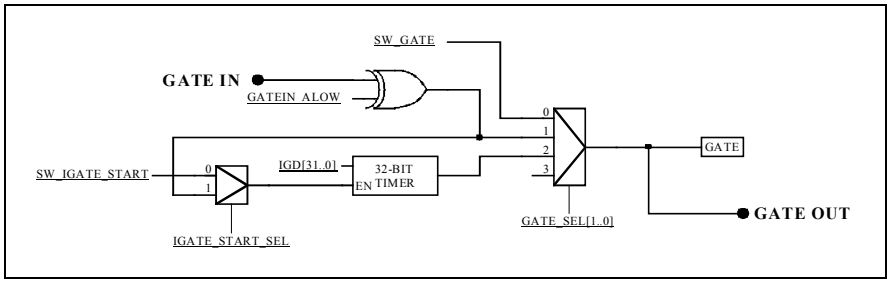


Figure 12: GATE selection scheme

2.4.3. Input trigger selection

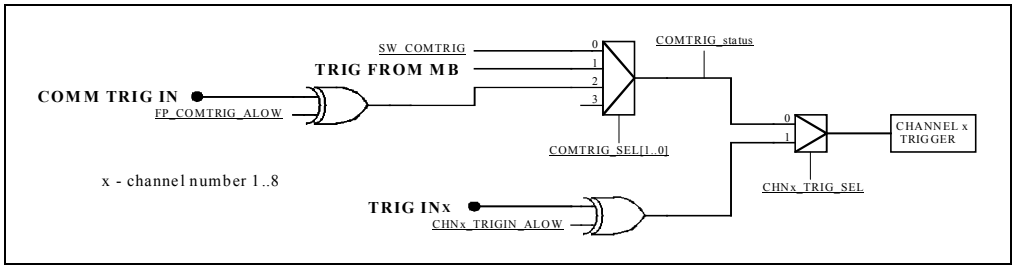


Figure 13: Input trigger selection

2.4.4. Output trigger configuration

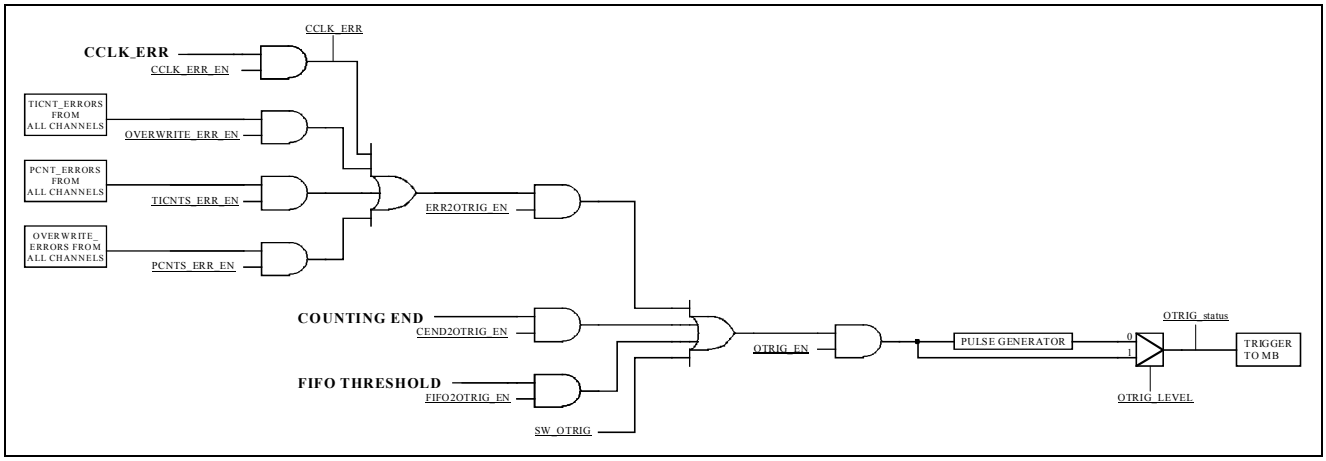


Figure 14: Output trigger configuration

2.4.5. Counters domain clock configuration

The control logic on the 3808 is divided into two clock domains: interface domain clocked by 80MHz oscillator and counters domain clocked by one of the three sources (on-board oscillator, FP ECL CLKIN, MB TRIGI).
80MHz interface domain clock is fixed and always turned on as required for the FC-MB interface. Thus the access to the registers is enabled immediately after the power is applied to the 3808.
In contrast, counters domain clock is disabled after power-up. This 100MHz clock is generated out of low frequency clock in the range from 2MHz up to 5MHz. The source of the low frequency clock is selected using CCLK_SEL bits between oscillator, MB input trigger and FP ECL clock input. When selected the low frequency clock is then applied to the PLL which generate the target 100MHz. The programmable PLL requires some configuration bits to be set: R[6:0], S[2:0], V[8:0]. The value of these bits depends on the input clock frequency. The table below shows the settings for the 2MHz and 5MHz clocks.

PLL input clock [MHz]	R[6:0]	S[2:0]	V[8:0]
2	0x0	0x1	0x5C
5	0x0	0x1	0x20

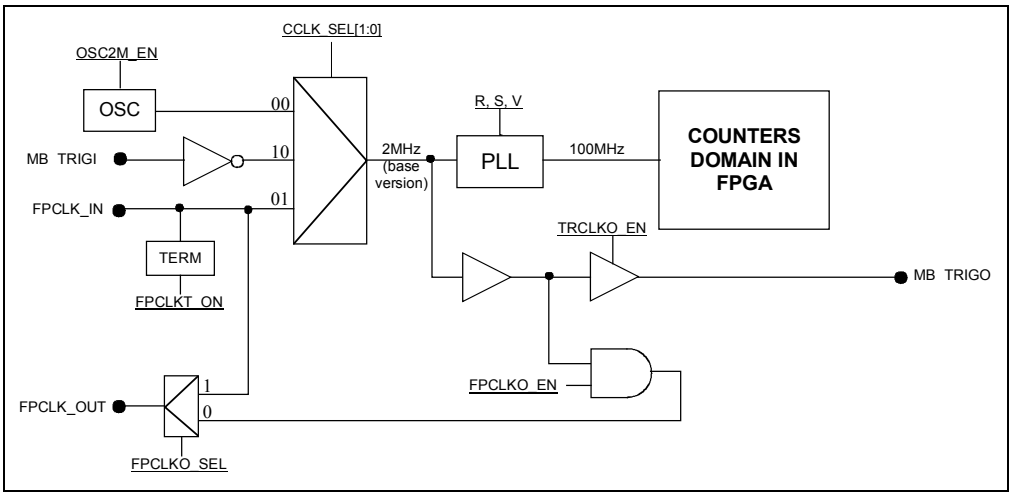


Figure 15: Counters Clock configuration

3. Input and output signals

SIGNAL	NAME	DESCRIPTION	LEVEL	I/O
Channel Input	(CHN1..8)	Counter / timer inputs	±5V	In
Gate In	(GATE IN)	External gate input or trigger input used to start internal gate	TTL	In
Com. Trigger In	(COMM TRIG IN)	External, common for all channels, trigger input. The common trigger in shares the pin with one line of independent trigger in	TTL	In
Trigger In	(TRIG IN 1..8)	External, independent for each channel, trigger inputs. One line of independent trigger in shares the input pin with common trigger in	TTL	In
Gate Out	(TTLOUT)	Signal from internal gate	TTL	Out
ECL clock In	(ECL CLK IN)	ECL clock input	ECL	In
ECL clock Out	(ECL CLK OUT)	ECL clock output	ECL	Out

3.1.1. Pin assignment of the SCSI connector:

Signal	A	B	Signal
TTLOUT	1	26	-
DGND	2	27	ECL CLK IN
DGND	3	28	ECL CLK OUT
DGND	4	29	-
DGND	5	30	-
DGND	6	31	GATE IN
DGND	7	32	COMM TRIG IN
DGND	8	33	TRIG IN8
DGND	9	34	TRIG IN7
DGND	10	35	TRIG IN6
DGND	11	36	TRIG IN5
DGND	12	37	TRIG IN4
DGND	13	38	TRIG IN3
DGND	14	39	TRIG IN2
DGND	15	40	TRIG IN1
DGND	16	41	-
DGND	17	42	-
DGND	18	43	CHN8
DGND	19	44	CHN7
DGND	20	45	CHN6
DGND	21	46	CHN5
DGND	22	47	CHN4
DGND	23	48	CHN3
DGND	24	49	CHN2
DGND	25	50	CHN1

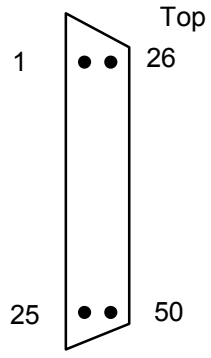


Fig. 1: SCSI front panel connector – view when 3808 card fitted on ProDAQ module in VXI crate.

4. Technical specification

ITEM	SPECIFICATION
Input characteristics	
Number of Input Channels	8
Input Range	±5V
Input Type	Single-ended
Coupling	DC, AC
Input Impedance	50Ω or 1MΩ for both DC and AC coupling
AC Coupling	1µF/25V in series with input signal
Max. Input Voltage	±10VDC @ 50Ω termination ±20VDC @ 1MΩ termination
Sensitivity	
Up to 10MHz	25mVrms
10MHz to 20MHz	40mVrms
20MHz to 25MHz	80mVrms
Frequency Measurement and Pulse Counting	
Max. frequency	25MHz
Min. pulse width	20ns
Counter width	32 bits
Gate Internal	
Range	400ns to 1717s
Resolution	400ns
Gate External	
Level	TTL
Minimum pulse duration	25ns
Active edge	Software selectable
Time Interval Measurement	
Range	40ns to 20000s
Resolution	10ns to 1ms in decade steps
Minimum time interval	40ns
Counter width	24 bits
Time Base	
Frequency	100MHz
Overall stability	±15ppm / 0..50°C
TCXO option	±1.5ppm
Threshold Level	
Range	±5V
Resolution	10 bits
Accuracy	±50mV

Trigger Input	
Level	TTL
Min. pulse width	25ns
Active level	Software selectable
FP CLK Input/Output	
Level	ECL
Freq. Range	2MHz to 5MHz
Termination	50 Ohm to -2V, software selectable
FIFO	4096 samples per 8 channels
Front Panel Connector	50-pin SCSI Female Connector
Current Consumption	Static / Dynamic @ Power Supply 80 mA / 10 mA @ +12V 80 mA / 10 mA @ -12V 300 mA / 200 mA @ +5V 120 mA / 90 mA @ -5.2V 20 mA / 40 mA @ -2V
Power Consumption (max.)	< 5.5 W
Operating Temperature	0 .. +50°C
Storage Temperature	-40 .. +70°C
Humidity	0-90%, non-condensing
Warm-up time	< 15min.
Dimensions	230mm x 52.6mm
Weight	100 g

5. Register Description

5.1. Address Map and Registers

All addresses are given in a hexadecimal notation. FC_ADR is address in FC address space. VXI_ADR is address in VXI address space. The appropriate address offset depending on FC position into MB should be applied (refer to MB manual).

FC_ ADR	VXI_ ADR	Register Name	Access	Function
FPGA internal registers				
0	0	**FCID_REG**	RO	FC ID register
1	4	**FCVER_REG**	RO	FC version register
2	8	**FCCTRL_REG**	RWC	General control and status register
3	C	**FIFOCTRL_REG**	RW	FIFO control/status register
4	10	**COMMAND_REG**	WO	Command register
5	14	**OTRI_REG**	RW	Output trigger control register
6	18	**ITRI_REG**	RW	Input trigger control register
7	1C	**DAC_REG**	RW	DACs setting register
8	20	**MODE_REG**	RW	Mode register
9	24	**IGATEL_REG**	RW	Internal gate width register, low
A	28	**IGATEH_REG**	RW	Internal gate width register, high
B	2C	**CHN1_CFG_REG**	RW	CHN1 configuration register
C	30	**CHN2_CFG_REG**	RW	CHN2 configuration register
D	34	**CHN3_CFG_REG**	RW	CHN3 configuration register
E	38	**CHN4_CFG_REG**	RW	CHN4 configuration register
F	3C	**CHN5_CFG_REG**	RW	CHN5 configuration register
10	40	**CHN6_CFG_REG**	RW	CHN6 configuration register
11	44	**CHN7_CFG_REG**	RW	CHN7 configuration register
12	48	**CHN8_CFG_REG**	RW	CHN8 configuration register
13	4C	**CHN1_2ECNT_REG**	RW	CHN1 and CHN2 edge counter setting register
14	50	**CHN3_4ECNT_REG**	RW	CHN3 and CHN4 edge counter setting register
15	54	**CHN5_6ECNT_REG**	RW	CHN5 and CHN6 edge counter setting register
16	58	**CHN7_8ECNT_REG**	RW	CHN7 and CHN8 edge counter setting register
17	5C	**CHN1_PCNT_REG**	RO	CHN1 PCNT output
18	60	**CHN2_PCNT_REG**	RO	CHN2 PCNT output
19	64	**CHN3_PCNT_REG**	RO	CHN3 PCNT output
1A	68	**CHN4_PCNT_REG**	RO	CHN4 PCNT output
1B	6C	**CHN5_PCNT_REG**	RO	CHN5 PCNT output
1C	70	**CHN6_PCNT_REG**	RO	CHN6 PCNT output
1D	74	**CHN7_PCNT_REG**	RO	CHN7 PCNT output
1E	78	**CHN8_PCNT_REG**	RO	CHN8 PCNT output
1F	7C	**FECFG_REG**	RW	Front-End configuration register
FA	3E8	**FCEPD_REG**	RW	EEPROM Data access register
FB	3EC	**FCEPC_REG**	RW	EEPROM Control register
FC	3F0	**FCSUBT_REG**	RO	FC subtype register
FE	3F8	**FCSERH_REG**	RO	Serial Number register, high
FF	3FC	**FCSERL_REG**	RO	Serial Number register, low
Memory space				
8000	20000	**FIFO_REG**	RO	Readout of FIFO memory

5.2. Register Description

5.2.1. FCID_REG
FC_ADR=0H, VXI_ADR=0H

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Operation	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Initial	0	0	1	1	1	0	0	0	0	0	0	0	1	0	0	0
Content	Product ID = 0x3808															

FCID_REG contains identification number of function card type.
Readout should give a value of 0x3808.

5.2.2. FCVER_REG
FC_ADR=1H, VXI_ADR=4H
This is the FC version register.

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Operation	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Initial	h	h	h	h	h	h	h	h	h	h	h	h	h	h	h	h
Content	major				minor				major				minor			
	FPGA revision								PCB revision							

5.2.3. FCCTRL_REG
FC_ADR=2H, VXI_ADR=8H
Function card Control and Status Register.

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Operation	RWC	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RW	RW	RWC	RW	RWC
Initial	0	h	h	h	0	0	0	1	0[(1)]	0[(1)]	0	0	0	0[(1)]	0	0
Content	PLL_WR	CFG[2:0]			COUNTING_ END	COUNTING_state	ARMED_state	ACCESS_state	PCNTS_ERR	TICNTS_ERR	OVERWRITE_ ERR	FPCLKT_ON	TTLOUT_EN	SW_IGATE_START	SW_GATE	FSM reset

(1) This bit might not be initialised if the counter clock CCLK is not present. After CCLK is applied this bit is initialised correctly.

FSMreset	Resets internal (in FPGA) state machines. The reset doesn't change content of the configuration bits in the registers. Reset is started by writing "1" to that bit. After the reset is done, the hardware clears the bit. Software should poll the bit until it is cleared. It is recommended to perform reset during FC initialisation. Write 0: no effect 1: starts reset of internal state machine Read 0: reset finished (if reset previously started) 1: reset in progress USAGE • During initialisation process as first step • To force state machine to known (ACCESS_state) state • To stop data acquisition before gate is ended • FSMreset takes approximately 1µs • FSMreset bit automatically initiates CLEARING_CMD and FIFO_reset
SW_GATE	This bit is used to generate the gate signal by software. Write 0: the GATE signal is off 1: the GATE signal is on Read gives the last written value USAGE • The action on this bit has effect only if GATE_SEL is set to select software gate • This bit is cleared when FSMreset is performed
SW_IGATE_START	This bit is used to start the internal gate signal by software. Write 0: no changes 1: generates pulse to start IGATE Read Always gives zero USAGE • The action on this bit has effect only if IGATE_START_SEL was set to select internal gate started by software

TTLOUT_EN	The bit is used to enable the output driver on TTLOUT pin of the connector. The pin then becomes the output pin and internal gate signal is driven to it. Write 0: disables the output driver 1: enables the output driver Read gives the last written value USAGE • To send out the internal gate signal to front panel
FPCLKT_ON	This bit is used to switch on the 50 ohm termination of the ECL clock input. Write 0: switches off the 50 ohm termination 1: switches on the 50 ohm termination Read gives the last written value
OVERWRITE_ ERR	The bit is read only and is set by hardware after overwrite of data (due to the bottleneck on FIFO input) from TICNT has occurred. This bit is cleared on arming command, clearing command or by FSMreset bit. Write No effect Read 0: no OVERWRITE errors 1: OVERWRITE error has occurred USAGE • This bit is used to detect if overwrite of data from TICNT happened
TICNTS_ERR	The bit is read only and is set by hardware if a double full revolution of any TICNT without storing data in FIFO occurred. This bit is cleared on arming command, clearing command or by FSMreset bit. Write No effect Read 0: no errors from TICNT 1: error in any TICNT has occurred USAGE • This bit is used to detect if TICNT error occured
PCNTS_ERR	The bit is read only and is set by hardware after full revolution of PCNT has occurred. This bit is cleared on arming command, clearing command or by FSMreset bit. Write No effect Read 0: no errors from PCNT 1: error in any PCNT has occurred USAGE • This bit is used to detect if full revolution of PCNT has happened

ACCESS_state	The bit indicates ACCESS_state of internal state machine. Write No effect Read 0: internal SM in state other than ACCESS_state 1: internal SM in ACCESS_state USAGE • To detect state of internal state machine • ACCESS_state is the initial state of internal SM (an idle state). Card configuration is allowed only in this state.
ARMED_state	The bit indicates ARMED_state of internal state machine. Write No effect Read 0: internal SM in state other than ARMED_state 1: internal SM in ARMED_state USAGE • To detect state of internal state machine • In armed state the card waits for the gate signal
COUNTING_state	The bit indicates COUNTING_state of internal state machine. Write No effect Read 0: internal SM in state other than COUNTING_state 1: internal SM in COUNTING_state USAGE • To detect state of internal state machine • The counting can take place in this state only. It leaves this state when gate ends or when FSM reset is performed
COUNTING_ END	The bit is read only and is set by hardware after end of measurement process, when gate signal becomes inactive. This bit is cleared on arming command or clearing command. Write No effect Read 0: counting in progress (if previously started) 1: counting ended USAGE • This bit can be used to detect end of measurement process. There are possible two ways: polling the bit or waiting for interrupt generated by this bit when output trigger was enabled
CFG[2:0]	Configuration pins used to determine different hardware options installed on the 3808. Read Give the actual state of the configuration pins USAGE • CFG[1:0] is assigned to determine the value of the counter clock oscillator: 00 – 2MHz, 01 – 5MHz, 10 and 11 not assigned at the moment of writing the spec. The value of the oscillator has influence on the settings of the PLL. • CFG[2] is not assigned at the moment writing the spec
PLL_WR	This bit is used to enable update of the PLL configuration bits (S[2:0], R[6:0] and V[8:0]). When this bit is set PLL configuration bits are updated

	with the value stored in the IGATEL_REG and IGATEH_REG registers. Readout of this bit gives the status of the counters domain clock. Write 0: update is disabled 1: update is started and when done the bit is cleared Read 0: counter clock works OK 1: counter clock not present or not stable USAGE • R[6..0] correspond to IGD[6..0], S[2..0] to IGD[10..8] and V[8..0] to IGD[24..16] • After update of the PLL configuration bits, make FSM_RESET and poll PLL_WR bit until it returns zero

5.2.4. FIFOCTRL_REG

FC_ADR=3H, VXI_ADR=CH

This register is a control/status register of the FIFO memory.

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Operation	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RWC	RWC
Initial	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
Content	FIFO_STATUS[11:0]												FIFO_FULL	FIFO_EMPTY	FIFO_WR	FIFO_reset

FIFOreset — The bit resets the FIFO. Reset is done by writing “1” to that bit and waiting for “0”. Resetting the FIFO means clearing the status of the FIFO and setting the empty flag.

Write
0: no effect
1: starts reset of FIFO

Read
0: resetting finished (if previously started)
1: resetting in progress

USAGE
- To empty the FIFO
- The FSMreset resets FIFO as well

FIFO_WR — The bit launches write of the data stored in IGATEL_REG and IGATEH_REG to the FIFO.

Write
0: no effect
1: starts write to the FIFO

Read
0: write to FIFO finished (if previously started)
1: write to FIFO in progress

USAGE
- To test the FIFO by writing data and reading it back

FIFO_EMPTY The bit indicates that FIFO memory is empty.
Write
No effect
Read
0: FIFO not empty
1: FIFO empty
USAGE
- To detect if the FIFO is empty when moving data from FIFO

FIFO_FULL The bit indicates that FIFO memory is full.
Write
No effect
Read
0: FIFO not full
1: FIFO full
USAGE
- To detect if the FIFO is full when moving data from FIFO

FIFO_STATUS[11:0] The bits indicate the contents of the FIFO memory.
Write
No effect
Read
Gives the exact amount of samples stored in FIFO (from 0 up to 4095)
USAGE
- To detect how many samples are in FIFO

5.2.5. COMMAND_REG
FC_ADR=4H, VXI_ADR=10H

Write to this register with the data set to 0x6 performs arming command: 3808 card goes from ACCESS_state to ARMED_state.
Write to this register with the data set to 0x5 performs clearing command: it clears the error bits and the output trigger line.

5.2.6. OTRI_REG
FC_ADR=5H, VXI_ADR=14H

Output Trigger register allows the selection of the source of the output trigger sent to the motherboard and to the VXI controller.
There are following trigger sources:
- Errors (PCNT_ERR, TICNT_ERR, OVERWRITE_ERR, CCLK_ERR)
- End of counting
- Reaching the pre-set number of samples in FIFO

The hardware allows the use of more than one trigger source at the time.
The Output Trigger can be used to generate interrupt at the end of measurement process.

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Operation	RO	RW	RW	RW	RW	RW	RW	RW	RW	RW	RO	RW	RW	RW	RW	RW
Initial	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Content	OTRIG_ status	PCNTS_ERR_EN	TICNTS_ERR_EN	OVERWRITE_ ERR_EN	ERR2OTRIG_EN	CEND2OTRIG_EN	OTRIG_EN	OTRIG_LEVEL	SW_OTRIG	CCLK_ERR_EN	CCLK_ERR	FIFO2OTRIG_EN	FIFO_TH[3:0]

FIFO_TH[3:0] This bits set the number of samples stored in FIFO needed to generate the trigger.

Write

FIFO_TH[3:0] x 256 gives the number of samples which are required to be stored in FIFO in order to generate the trigger

Read

Gives the last written value

USAGE

- In order to enable trigger generated by FIFO the FIFO2OTRIG_EN and the OTRIG_EN bit should be set

FIFO2OTRIG_EN This bit enables generating output trigger when selected on FIFO_TH[3:0] bits number of samples in FIFO was reached.

Write

0: generating trigger after reaching set number of samples disabled

1: generating trigger after reaching set number of samples enabled

Read

Gives the last written value

USAGE

- In addition to this bit the OTRIG_EN bit should be set to enable output trigger
- The source of this trigger is cleared when writing '1' to the bit

CCLK_ERR This bit shows the status of the CCLK_ERR signal.

Read

0: no error detected

1: counter clock error detected

USAGE

- The counter clock can be received from external source. After the board is armed this clock has to stay stable otherwise CCLK_ERR will be generated.
- This bit is cleared either by arming command or clearing command
- This bit will not be asserted when the board is in ACCESS_STATE but when asserted in other states this bit will stay asserted even if going to ACCESS_STATE

CCLK_ERR_EN This bit enables the CCLK_ERR to be the source of the output trigger.

Write

0: disables CCLK_ERR as a source of the trigger

1: enables CCLK_ERR as a source of the trigger

Read

Gives the last written value

SW_OTRIG This bit allows setting output trigger by software.
Write
0: no change
1: sets the output trigger
Read
Gives the status of software generated output trigger
USAGE
- This trigger source was designed for debugging purposes only

OTRIG_LEVEL This bit allows selection of the output trigger generating mode:
- Pulse – after rising edge of trigger source pulse of 400-800ns width will be generated independently of trigger source high level duration
- Level – after rising edge of trigger source output trigger level will follow the level of trigger source.

Write
0: output trigger generating mode set to pulse (400-800ns width)
1: output trigger generating mode set to level
Read
Gives the last written value
USAGE
- When working with interrupts the level mode should be set

OTRIG_EN The bit is the main output trigger enable bit. If this bit is cleared no output trigger will be sent to MB independently of trigger source enable bits. If this bit is set output trigger will be sent if any of the trigger sources are enabled and its conditions are met.
Write
0: output trigger disabled
1: output trigger enabled
Read
Gives the last written value
USAGE
- Main output trigger enabling bit

CEND2OTRIG_EN The bit enables generating output trigger when measurement process was ended (gate signal became inactive).
Write
0: generating trigger on counting end disabled
1: generating trigger on counting end enabled
Read
Gives the last written value
USAGE
- In addition to this bit the OTRIG_EN bit should be set to enable output trigger

ERR2OTRIG_EN The bit enables generating output trigger when error occurred.
Write
0: generating trigger on error condition disabled
1: generating trigger on error condition enabled
Read
Gives the last written value
USAGE
- In addition to this bit the OTRIG_EN bit should be set to enable output trigger and particular error source should be enabled

OVERWRITE_ ERR_EN

The bit enables OVERWRITE_ERR as an error condition.
Write
0: OVERWRITE_ERR disabled
1: OVERWRITE_ERR enabled
Read
Gives the last written value
USAGE
- This bit is used to enable measurement to be stopped on overwrite error (if ERR_STOPED_EN was set) or to send the trigger to MB when the error happened (when ERR2OTRIG_EN and OTRIG_EN bits are set)

TICNTS_ERR_EN

The bit enables TICNTS_ERR as an error condition.
Write
0: TICNTS _ERR disabled
1: TICNTS _ERR enabled
Read
Gives the last written value
USAGE
- This bit is used to enable measurement to be stopped on TICNT error (if ERR_STOPED_EN was set) or to send the trigger to MB when the error happened (when ERR2OTRIG_EN and OTRIG_EN bits are set)

PCNTS_ERR_EN

The bit enables PCNTS_ERR as an error condition.
Write
0: PCNTS_ERR disabled
1: PCNTS_ERR enabled
Read
Gives the last written value
USAGE
- This bit is used to enable measurement to be stopped on PCNT error (if ERR_STOPED_EN was set) or to send the trigger to MB when the error happened (when ERR2OTRIG_EN and OTRIG_EN bits are set)

OTRIG_status

The state of the output trigger line.
Write
No effect
Read
0: output trigger inactive
1: output trigger active
USAGE
- When working with interrupts this bit should be used by the interrupt routing to determine the interrupt source

5.2.7. ITRI_REG

FC_ADR=6H, VXI_ADR=18H

The register allows selection of the common trigger source used optionally to start TI counters. The following are the possible input trigger sources:

- Software trigger
- MB input trigger
- External trigger through FP

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Operation									RO				RW	RW	RW	RW
Initial									X				0	0	0	0
Content	Not used								COMTRIG_status	Not used			SW_COMTRIG	COMTRIG_SEL[1:0]		FP_COMTRIG_ALOW

FP_COMTRIG_ALOW The bit enables changing of the active level of external common trigger coming from FP.
Write
0: active level high
1: active level low
Read
Gives the last written value

COMTRIG_SEL[1:0] The bits select the source of common trigger.
Write
00: software generated common trigger
01: common trigger from MB
10: common trigger from external connector
11: reserved
Read
Gives the last written value

SW_COMTRIG The bit allows generation of the common trigger by the software if software is selected as the source of the common trigger.
Write
0: common trigger asserted
1: common trigger de-asserted
Read
Gives the last written value

COMTRIG_status The state of the common trigger.
Write
Has no effect
Read
Gives the current state of the common trigger

5.2.8. DAC_REG

FC_ADR=7H, VXI_ADR=1CH

The DAC_REG allows setting the output value of the on-board DAC. The 8-output DAC is used to control offset of each channel.

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Operation	RWC		WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
Initial	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0
Content	DAC trans	Not used	DAC_ADDR[3:0]				DAC_DATA[9..0]									

DAC_DATA[9..0] These bits are data to write to the 8-channel DAC. These bits set DAC outputs.
Write
DD[9..0] set the output value of DAC
Read
These are write only bits
USAGE
- The data should be in binary format
- The DAC outputs range is –5.00V..+4.99V
- Output voltage = 5*(DD[9..0] – 512)/512 and is expressed in volts

DAC_ADDR[3..0] These bits address the channel of DAC.
Write
DAC_ADDR[3..0] set the address of DAC channel
Read
These are write only bits
USAGE
- DAC_ADDR[3..0] = Channel_number.
- Channel_number range is from 1 to 8

DACtrans This bit starts data shifting out to DAC. Writing “1” to this bit starts data (D[12..0]) shifting out to DAC. This bit is cleared to “0” after shifting is finished.
Write
0: no effect
1: starts DAC_DATA[9..0] shifting out
Read
0 : shifting out to DAC finished (if previously started)
1 : shifting out to DAC in progress
USAGE
- To start and detect the end of shifting out
- Shifting out takes approximately 8µs
- During shifting out DACtrans bit is set

5.2.9. MODE_REG
FC_ADR=8H, VXI_ADR=20H
This is the mode register. Each bit of this register is writable and readable.

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Operation	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Initial	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Content	OSC2M_EN	TRCLKO_EN	FPCLKO_EN	FPCLKO_SEL	CCLK_SEL[1:0]		PCNT_UPWORD	ERR_STOPED_EN	TB_SEL[2:0]			TB_EN	IGATE_START_SEL	GATE_SEL[1:0]		GATEIN_ALOW

GATEIN_ALOW This bit sets the active level of the external gate.
Write
0: active level high
1: active level low
Read
Gives the last written value

GATE_SEL[1:0] These bits select the source of the gate signal.
Write
00: software generated gate
01: external gate
10: internal gate
11: gate disabled
Read
Gives the last written value

IGATE_START_SEL This bit selects the source that starts the internal gate generation.
Write
0: internal gate started by software
1: internal gate started from external signal applied to external gate pin
Read
Gives the last written value

TB_EN This bit enables time base for the Time Interval counters.
Write
0: Time Base disabled
1: Time Base enabled
Read
Gives the last written value

TB_SEL[2:0] These bits set the frequency of the time base of the TI counters.
Write
000: time base set to 100MHz
001: time base set to 10MHz
010: time base set to 1MHz
011: time base set to 100KHz
100: time base set to 10KHz
101: time base set to 1KHz
110: reserved
111: reserved
Read
Gives the last written value

ERR_STOPPED_EN This bit enables stopping the measurement when an error happens.
Write
0: errors don't stop measurement
1: errors stop measurement
Read
Gives the last written value
USAGE
- In addition to this bit error source has to be selected and enabled (bits OVERWRITE_ERR_EN, TICNTS_ERR_EN and PCNTS_ERR_EN in OTRI_REG)

PCNT_UPWORD This bit allows the switching between upper and lower word of the PCNT output.
Write
0: lower word selected to read out through the CHNx_PCNT_REG
1: upper word selected to read out through the CHNx_PCNT_REG
Read
Gives the last written value
USAGE
- The content of PCNT is read out in two steps: lower word and upper word. The PCNT_UPWORD bit selects the word which can be readout from the CHNx_PCNT_REG

CCLK_SEL[1:0] These bits select the source of the counter clock.
Write
00: on-board oscillator selected
01: FP ECL clock input
10: clock received from trigger input
11: none selected
Read
Gives the last written value
USAGE
- Oscillator source is enabled using OSC2M_EN bit

FPCLKO_SEL This bit selects the source of the FP ECL clock output.
Write
0: on-board counter clock connected to an output
1: ECL clock input looped back to output
Read
Gives the last written value

FPCLKO_EN This bit is used to enable the counter clock as a source of the FP clock output.
Write
0: counter clock disabled as a source of the FP clock
1: counter clock enabled as a source of the FP clock
Read
Gives the last written value
USAGE
- To receive the counter clock on the FP ECL clock output, FPCLKO_SEL has to be set to 0 in addition to this bit

TRCLKO_EN This bit is used to enable the counter clock as a source of the MB trigger output.
Write
0: counter clock disabled as a source of the MB trigger
1: counter clock enabled as a source of the MB trigger
Read
Gives the last written value
USAGE
- Enabling the counter clock as a MB trigger source automatically disables the FPGA internal trigger sources

OSC2M_EN This bit enables the on-board oscillator used to clock the counters.
Write
0: on-board oscillator disabled
1: on-board oscillator enabled
Read
Gives the last written value
USAGE
- The oscillator is one of three sources of the clock counter, selected using CCLK_SEL bits.

5.2.10.IGATEx_REG
FC_ADR=9H/AH, VXI_ADR=24H/28H

The registers allow to set-up the internal gate width. The bits IGD[31..0] of 32-bit internal gate counter are set through IGATEL_REG and IGATEH_REG. For testing purposes the contents of these registers is loaded to FIFO when FIFO_WR was set. In the same way IGATEL_REG and IGATEH_REG registers are used to setup PLL configuration bits when PLL_WR bit is asserted (R[6..0] correspond to IGD[6..0], S[2..0] to IGD[10..8] and V[8..0] to IGD[24..16]).

IGATEL_REG

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Operation	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Initial	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Content	IGD[15..0]															

IGATEH_REG

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Operation	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Initial	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Content	IGD[31..16]															

IGD[31..0] These bits set the width of internal gate.
Write
Set the internal gate width
Read
Gives the last written value

The internal gate width is expressed by the following equation:

$$T_{IGATE} = T_{OSC} \cdot (IGD[31..0])$$

where
T_{IGATE} – the internal gate width
T_{OSC} – the period of signal clocking the internal gate counter, equal 400ns
IGD[31..0] – the value written to IGATEL_REG and IGATEH_REG
To calculate the value, which has to be written (after rounding to integer) to register the following equation can be used:

$$IGD[31..0] = \frac{T_{IGATE}}{T_{OSC}}$$

(description as above).
The range of IGD[31..0] is from 1 to FFFF_FFFFH giving the value of T_{IGATE} from 400ns to 1717.98s.

5.2.11.CHNx_CFG_REG
FC_ADR=BH/CH/DH/EH/FH/10H/11H/12H, VXI_ADR=2CH/30H/34H/38H/3CH/40H/44H/48H

This is channel configuration register. The “x” represents channel number and changes from 1 to 8.

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Operation	RO	RO	RW	RW	RW	RW	RW	RW			RW	RW	RW	RW	RW	RW
Initial	0[1]	0[1]	0	0	0	0	0	0			0	0	0	0	0	0
Content	CHNx_PCNT_ERR	LIMITED_COMPLETED	CHNx_TRIG_SEL	CHNx_TRIGIN_ALOW	CHNx_SYNC	CHNx_LIMITED	CHNx_WINDOW	CHNx_TRIG_STARTED	Not used		CHNx_REDGE_FIRST	CHNx_FEDGE_EN	CHNx_REGDE_EN	CHNx_PCNT_FEDGE	CHNx_PCNT_EN	CHNx_EN

[1] This bit might not be initialised if the counter clock CCLK is not present on the FPGA pins. After CCLK is applied this bit is initialised correctly.

CHNx_EN This bit enables channel #x input.
Write
0: channel disabled
1: channel enabled
Read
Gives the last written value

CHNx_PCNT_EN This bit enables the PCNT in channel #x.
Write
0: PCNT disabled
1: PCNT enabled
Read
Gives the last written value

CHNx_PCNT_FEDGE This bit sets the edge direction the PCNT in channel #x reacts on.
Write
0: PCNT counts rising edges
1: PCNT counts falling edges
Read
Gives the last written value

CHNx_REDGE_EN This bit enables raising edges as an event for TICNT in channel #x.
Write
0: raising edges disabled
1: raising edges enabled
Read
Gives the last written value

CHNx_FEDGE_EN This bit enables falling edges as an event for TICNT in channel #x.
Write
0: falling edges disabled
1: falling edges enabled
Read
Gives the last written value

CHNx_REDGE_FIRST This bit defines the starting edge for TICNT in channel #x.
Write
0: TICNT starts counting from falling edge
1: TICNT starts counting from raising edge
Read
Gives the last written value
USAGE
- This bit has to be set when both rising and falling edges are enabled as an event (CHNx_REDGE_EN and CHNx_FEDGE_EN bits set)

CHNx_TRIG_STARTED When this bit is set counting in TICNT in channel #x is started by trigger signal. (while gate is active)
Write
0: TICNT starts immediately after gate
1: TICNT starts when trigger is active during gate active
Read
Gives the last written value

CHNx_WINDOW This bit sets the mode of the trigger for channel #x: launch or window.
Write
0: launch mode is set (trigger starts only the counting)
1: window mode is set (counting is enabled during trigger)
Read
Gives the last written value

CHNx_LIMITED This bit sets the limited mode of TICNT for channel #x.
Write
0: limited mode is disabled
1: limited mode is enabled
Read
Gives the last written value

CHNx_SYNC This bit sets the synchronous mode of TICNT for channel #x.
Write
0: synchronous mode is disabled
1: synchronous mode is enabled
Read
Gives the last written value

CHNx_TRIGIN_ALOW This bit enables setting of the active level of external independent trigger coming from FP.
Write
0: active level high
1: active level low
Read
Gives the last written value

CHNx_TRIG_SEL This bit selects the trigger for channel #x: common trigger or independent trigger.
Write
0: common trigger selected
1: independent trigger selected
Read
Gives the last written value

LIMITED_COMPLETED This bit indicates that set number of samples has been acquired for the channel #x when working in limited mode.
Write
No effect
Read
0: set number of samples not reached
1: set number of samples reached
USAGE
- This bit is cleared on arming command

CHNx_PCNT_ERR This bit gives the overflow error from PCNT for the channel #x.
Write
No effect
Read
0: no overflow error for PCNT
1: overflow happened and data from PCNT for the channel #x are not valid
USAGE
- This bit is cleared on arming command or clearing command

5.2.12.CHN1_2ECNT_REG
FC_ADR=13H, VXI_ADR=4CH

The register allows setting of the edge counter in channel 1 and channel 2. The edge counter defines the number of edges (events) to be collected in LIMITED mode.
When set in Limited mode each channel can store up to 256 values in FIFO memory, depending on setting on CHNxEC[7:0] bits. The amount of data written to FIFO for all channels can not exceed FIFO size minus 1 (1023 samples) if FIFO is not emptying during measurement process.

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Operation	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Initial	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Content	CHN2EC[7:0]								CHN1EC[7:0]							

CHN2EC[7:0] These bits set the number of input signal edges that have to be acquired as a latch event in TICNT.
Write
Sets the ECNT in channel #1
Read
Gives the last written value
USAGE
These bits have to be set in Limited mode only

CHN1EC[7:0] These bits set the number of input signal edges that have to be acquired as a latch event in TICNT.
Write
Sets the ECNT in channel #2
Read
Gives the last written value
USAGE
- These bits have to be set in Limited mode only

The number of edges that will be counted is expressed by the following equation:

$$Number\_of\_edges = CHNxEC[7:0] + 1$$

where

x – the channel number

CHNxEC[7..0] – the value written to ECNT in channel #x

5.2.13.CHN3_4ECNT_REG

FC_ADR=14H, VXI_ADR=50H

This register allows setting the edge counter input data in channel 3 and channel 4. The edge counter defines the number of edges (events) to be collected in LIMITED mode.

When set in Limited mode each channel can store up to 256 values in FIFO memory, depending on the setting of CHNxEC[7:0] bits. The number of data written to FIFO for all channels can not exceed FIFO size minus 1 (1023 samples) if FIFO is not emptying during measurement process.

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Operation	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Initial	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Content	CHN4EC[7:0]								CHN3EC[7:0]							

CHN3EC[7:0] Thees bits set the number of input signal edges that have to be acquired as a latch event in TICNT.

Write

Sets the ECNT in channel #3

Read

Gives the last written value

USAGE

These bits have to be set in Limited mode only

CHN4EC[7:0] These bits set the number of input signal edges that have to be acquired as a latch event in TICNT.

Write

Sets the ECNT in channel #4

Read

Gives the last written value

USAGE

- These bits have to be set in Limited mode only

The number of edges that will be counted is expressed by the following equation:

$$Number\_of\_edges = CHNxEC[7:0] + 1$$

where

x – the channel number

CHNxEC[7..0] – the value written to ECNT in channel #x

5.2.14. CHN5_6ECNT_REG
FC_ADR=15H, VXI_ADR=54H

This register allows setting of the edge counter input data in channel 5 and channel 6. The edge counter defines the number of edges (events) to be collected in LIMITED mode.
When set in Limited mode each channel can store up to 256 values in FIFO memory, depending on the setting of CHNxEC[7:0] bits. The amount of data written to FIFO for all channels can not exceed FIFO size minus 1 (1023 samples) if FIFO is not emptying during measurement process.

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Operation	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Initial	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Content	CHN6EC[7:0]								CHN5EC[7:0]							

CHN5EC[7:0] These bits set the number of input signal edges that have to be acquired as a latch event in TICNT.
Write
Sets the ECNT in channel #5
Read
Gives the last written value
USAGE
These bits have to be set in Limited mode only

CHN6EC[7:0] These bits set the number of input signal edges that have to be acquired as a latch event in TICNT.
Write
Sets the ECNT in channel #6
Read
Gives the last written value
USAGE
- These bits have to be set in Limited mode only

The number of edges that will be counted is expressed by the following equation:

$$Number\_of\_edges = CHNxEC[7:0] + 1$$

where
x – the channel number
CHNxEC[7..0] – the value written to ECNT in channel #x

5.2.15.CHN7_8ECNT_REG
FC_ADR=16H, VXI_ADR=58H

This register allows setting the edge counter input data in channel 7 and channel 8. The edge counter defines the number of edges (events) to be collected in LIMITED mode.
When set in Limited mode each channel can store up to 256 values in FIFO memory, depending on the setting of CHNxEC[7:0] bits. The amount of data written to FIFO for all channels can not exceed FIFO size minus 1 (1023 samples) if FIFO is not emptying during measurement process.

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Operation	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Initial	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Content	CHN8EC[7:0]								CHN7EC[7:0]							

CHN7EC[7:0] These bits set the number of input signal edges that have to be acquired as a latch event in TICNT.
Write
Sets the ECNT in channel #7
Read
Gives the last written value
USAGE
These bits have to be set in Limited mode only

CHN8EC[7:0] These bits set the number of input signal edges that have to be acquired as a latch event in TICNT.
Write
Sets the ECNT in channel #8
Read
Gives the last written value
USAGE
- These bits have to be set in Limited mode only

The number of edges that will be counted is expressed by the following equation:

$$Number\_of\_edges = CHNxEC[7:0] + 1$$

where
x – the channel number
CHNxEC[7..0] – the value written to ECNT in channel #x

5.2.16.CHNx_PCNT_REG

FC_ADR=17H/18H/19H/1AH/1B/1CH/1DH/1EH, VXI_ADR=5CH/60C/64H/68H/6CH/70H/74H/78H

The registers are read only and allow the readout of the value of Pulse Counter in channel x. The 32-bit value from Pulse Counter is readout as two 16-bit numbers from CHNx_PCNT_REG and should be then merged. The selection between upper and lower word that is seen in CHNx_PCNT_REG is done using PCNT_UPWORD bit from the MODE_REG.
The "x" value represents channel number and is in range from 1 to 8.

CHNx_PCNT_REG

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Operation	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Initial	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Content	CHNxPC[31..16] or CHNxPC[15..0] depending on PCNT_UPWORD bit setting															

CHNxPC[31..0] The value from PCNT counter in channel #x.
x: 1 to 8
Write
Has no effect
Read
Gives the current value from Pulse Counter in channel #
USAGE
- To readout the value stored in the Pulse Counter
- Readout of this register during the measurement can give incorrect value (due to changes of counter bits)

5.2.17.FECONF_REG
FC_ADR=1FH, VXI_ADR=7CH

This register allows configuring the front-end of the 3808.

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Operation	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Initial	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Content	CHN8_TERM	CHN8_DC	CHN7_TERM	CHN7_DC	CHN6_TERM	CHN6_DC	CHN5_TERM	CHN5_DC	CHN4_TERM	CHN4_DC	CHN3_TERM	CHN3_DC	CHN2_TERM	CHN2_DC	CHN1_TERM	CHN1_DC

CHx_DC
x: 1 to 8

These bits allows the input of channel #x to be DC coupled.

Write

0: relay with DC coupling path closed – DC coupling on

1: relay with DC coupling path opened – AC coupling on

Read

Gives the last written value

USAGE

•

CHx_TERM
x: 1 to 8

These bits allows the input of channel #x to be terminated with 50Ω.

Write

0: 50Ω terminator connected - 50Ω termination

1: 50Ω terminator not connected – 1MΩ termination

Read

Gives the last written value

5.2.18.FCEPD_REG
FC_ADR=FAH

EEPROM Data access register.

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Operation	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Initial	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Content	EEP_DATA															

EEP_DATA

This is the command/address/data byte that will be transferred to the TEDS sensor memory during WRITE operation or data read from memory after READ operation.

Write

Stores written data to be transmitted to EEPROM after WRITE command is issued in EPC register

• Read

Returns the data read from EEPROM after last READ command issued through EPC register

5.2.19.FCEPC_REG
FC_ADR=FBH

EEPROM Control register, used to read/write the EEPROM data.

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Operation	RO	WO			WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO
Initial	0	0			0	0	0	0	0	0	0	0	0	0	0	0
Content	EEP_BUSY	RD_nWR			EEP_ADDR											

EEP_ADDR This is the address of the word location in the EEPROM memory. The map of the locations is given in table below.

Write

Selects EEPROM location for which command is issued (range: 0..127)

RD_nWR This bit is used to select desired EEPROM operation.

Write

0 : WRITE to EEPROM

1 : READ from EEPROM

EEP_BUSY This is the status bit that indicates when the EEPROM is ready to accept next access. After any access to EEPROM, EEP_BUSY bit goes high and stays high as long as the access is not finished.

Read:

0 : board ready to accept next command to be performed

1 : board busy and writes to EPC register will be ignored

USAGE

Directly after reset this flag is active for about 1ms until the initialisation process is finished

Word address	Information stored
0 (0x0)	The content of FCSUB register.
1 (0x1)	The content of FCSERH register (serial number).
2 (0x2)	The content of FCSERL register (serial number).

5.2.20.FCSTYPE _REG
FC_ADR=FCH

This is the function card sub-type register useful for software to distinguish between versions of the board. This register is automatically loaded during FPGA initialisation with contents of the EEPROM chip (word address 0x0).

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Operation	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Initial	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Content	STYPE_2CH								STYPE_1CH							

STYPE_1CH First ASCII character of the FC sub-type

STYPE_2CH Second ASCII character of the FC sub-type

5.2.21.FCSERH_REG
FC_ADR=FEH

This register contains the upper 16 bits of the FC serial number. This register is automatically loaded during FPGA initialisation with contents of the EEPROM chip (word address 0x1).

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Operation	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Initial	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Content	FC_SN[31..16]															

FC_SN[31..16] Upper 16 bits (FC_SN[31:16]) of the FC Serial Number

5.2.22.FCSERL_REG
FC_ADR=FFH

This register contains the lower 16 bits of the FC serial number. This register is automatically loaded during FPGA initialisation with contents of the EEPROM chip (word address 0x2).

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Operation	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Initial	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Content	FC_SN[15..0]															

FC_SN[15..0] Lower 16 bits (FC_SN[15:0]) of the FC Serial Number

5.2.23.FIFO_REG
FC_ADR=8000H, VXI_ADR=20000H

The FIFO memory contains the data from the TICNT counters. The 32-bit word is written into FIFO. The format of this data is as follow:

FIFO_DATA[31:29]	Channel number (0 means channel number 1)
FIFO_DATA[28:27]	Not used
FIFO_DATA[26]	OVER_ERR: When 1 overwrite error occurred for this sample
FIFO_DATA[25]	TICNT_ERR: When 1 TI counter error occurred for this sample
FIFO_DATA[24]	FR: Full revolution bit
FIFO_DATA[23:0]	TICNT_DATA: Data from TICNT counters

Every sample stored in FIFO has to be read in two steps: upper 16-bit (MSB) word as a first readout and lower 16-bit (LSB) word as a second readout. For the given number of samples written to the memory double the number of 16-bit wide readouts to be performed.
After moving the samples from the FIFO to host the samples have to be rearranged according to the channel number. Then to get the time interval value the following equation has to be used:

$$T_n = \left(TICNT\_DATA_n - TICNT\_DATA_{n-1} + 16777216 * \left|FR_n - FR_{n-1}\right|\right) * \left(1 - TICNT\_ERR_n\right) * \left(1 - OVER\_ERR_n\right)$$

where

n – sample index, for n=0 (first data) TICNT_DATA $_{-1}$ should be set to zero,

| | - means absolute value.

T_n takes value of zero if TICNT_ERR$_n$ or OVER_ERR$_n$ happened (the bits are set to one). In such case this value has to be rejected.

Bustec Production, Ltd.
World Aviation Park, Shannon, Co. Clare, Ireland
Tel: +353 (0) 61 707100, FAX: +353 (0) 61 707106