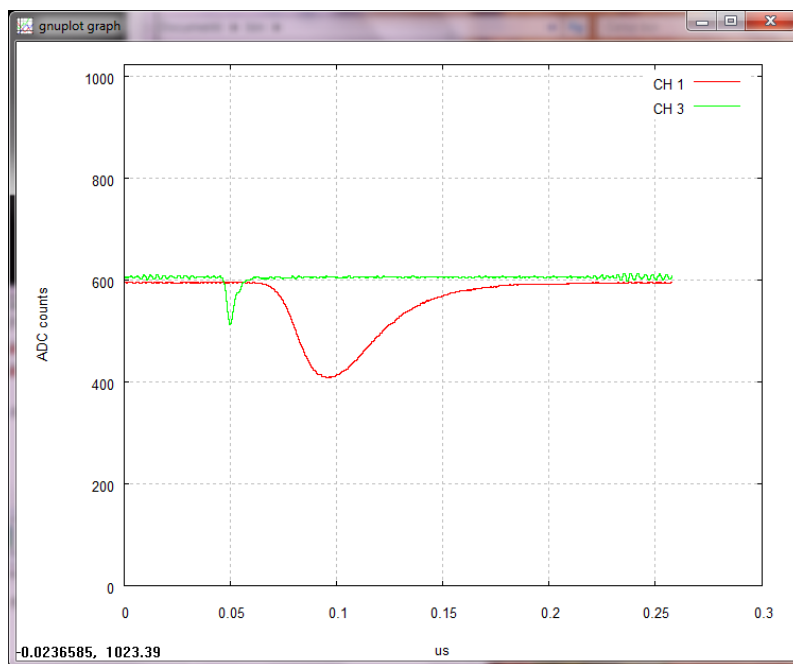




Rev. 4 - October 10<sup>th</sup>, 2025

# How to make channel correlations with CAEN Digitizers

The complete guide with step-by-step instructions



## Purpose of the Document



This guide wants to provide detailed instructions on how to implement channel correlations with CAEN digitizers, equipped either with WR or DPP firmware, and MCAs, presenting a DPP-PHA firmware.

**For any reference to registers in this user manual, please refer to the related document at the module web page.**

## Change Document Record

Date	Revision	Changes
Sep. 23 <sup>rd</sup> , 2013	00	Initial Release
Jul. 4 <sup>th</sup> , 2016	01	Fully revised version. Added support to MC <sup>2</sup> Analyzer. Added support to x730, x725, 781, and 790 series. DPP-CI no longer supported
Nov. 22 <sup>nd</sup> , 2016	02	Modified register 0x1n84 for DPP-PSD of 725 and 730 series (revision <b>4.11_136.10</b> on). Modified register 0x1nA0 for DPP-PHA of 725 and 730 series (revision <b>4.11_139.5</b> on). Added example for 780 series. Modified bits[11:10] of register 0x811C. Corrected examples for 725-730 DPP-PHA. Fully revised Chapter 5 in "How to Veto the acquisition".
Mar. 14 <sup>th</sup> , 2017	03	Modified examples for DPP-PSD of 725 and 730 series: latency window inside the couple set to 2 clock cycles; trigger validation mask corrected to 0xFFFFFFFF; added example for coincidences among 16 channels (VME).
Oct. 10 <sup>th</sup> , 2025	04	Reviewed Cover and End pages. Enumerated document sections. Updated Chap. 1, Chap. 2, Chap. 4, Chap. 5, Chap. 6, Chap. 11. Added Chap. 3, Chap. 7, Chap. 8, Chap. 9 and Chap. 10.

## Symbols, Abbreviated Terms and Notations

A/D	Analog to Digital
ADC	Analog-to-Digital Converter
BLR	BaseLine Restorer
CI	Charge Integration
DAQ	Data Acquisition
DPP	Digital Pulse Processing
FADC	Flash Analog to Digital Converter
FPGA	Field Programmable Gate Array
ICR	Incoming Count Rate
MCA	Multi-Channel Analyzer
MCS	Multi-Channel Scaler
OS	Operating System
PC	Personal Computer
PHA	Pulse Height Analysis
PMT	Photo Multiplier Tube
PSD	Pulse Shape Discrimination
QDC	Charge-to-Digital Converter
TAC	Time-to-Amplitude Converter
TCP/IP	Transmission Control Protocol/Internet Protocol
TDC	Time-to-Digital Converter
TOF	Time-of-Flight
USB	Universal Serial Bus
WR	Waveform Recording
&	AND, coincidence
	OR
$\oplus$	XOR, anti-coincidence

## Reference Documents

- [RD1] W. R. Leo. *Techniques for Nuclear and Particle Physics Experiments*. Ed. by Springer. II ed.
- [RD2] K. S. Crane. *Introductory nuclear physics*. Ed. by J. Wiley and sons
- [RD3] G. F. Knoll. *Radiation detection and measurement*. Ed. by J. Wiley and sons. III ed.
- [RD4] UM5960 – CoMPASS User Manual
- [RD5] UM5175 – V2495 User Manual
- [RD6] UM6508 – DT5495 User Manual
- [RD7] GD6300 – CoMPASS QuickStart Guide
- [RD8] AN2086 – Synchronization of a multi-board acquisition systems with CAEN digitizers
- [RD9] UM4380 – 725-730 DPP-PSD Registers Description
- [RD10] UM5678 – 725-730 DPP-PHA Registers Description
- [RD11] UM11802 – 2730 FELib PSD Parameters User Manual
- [RD12] UM8762 – 2740 FELib PSD Parameters User Manual
- [RD13] UM11796 – 2745 FELib PSD Parameters User Manual
- [RD14] UM7788 – 2740 FELib PHA Parameters User Manual
- [RD15] UM11794 – 2745 FELib PHA Parameters User Manual
- [RD16] UM11800 – 2730 FELib Scope Parameters User Manual
- [RD17] UM7787 – 2740 FELib Scope Parameters User Manual
- [RD18] UM11792 – 2745 FELib Scope Parameters User Manual
- [RD19] UM11844 – 2751 FELib Scope Parameters User Manual
- [RD20] UM4855 – 720 DPP-PSD Registers Description
- [RD21] UM5110 – 751 DPP-PSD Registers Description
- [RD22] UM5416 – DT5790 DPP-PSD Registers Description
- [RD23] UM6769 – 724-781 DPP-PHA Registers Description
- [RD24] UM6771 – 780 DPP-PHA Registers Description
- [RD25] UM3148 – DT5730/DT5725 User Manual
- [RD26] UM3147 – N6730/N6725 User Manual
- [RD27] UM2792 – V1730/VX1730 & V1725/VX1725 User Manual
- [RD28] UM2754 – WaveCatcher User Manual
- [RD29] UM2091 – CAEN WaveDump User Manual
- [RD30] UM5961 – 720 WR Registers Description
- [RD31] UM5918 – 724 WR Registers Description
- [RD32] UM6009 – 751 WR Registers Description
- [RD33] UM7934 – WaveDump2 User Manual

All CAEN documents can be downloaded at:

<https://www.caen.it/support-services/documentation-area/> (login required)

## Manufacturer Contacts



---

**CAEN S.p.A.**  
Via Vetraia, 11 55049 Viareggio (LU) - ITALY  
Tel. +39.0584.388.398 Fax +39.0584.388.959  
[www.caen.it](http://www.caen.it) | [info@caen.it](mailto:info@caen.it)  
©CAEN SpA – 2025

## Limitation of Responsibility

If the warnings contained in this manual are not followed, CAEN will not be responsible for damage caused by improper use of the device. The manufacturer declines all responsibility for damage resulting from failure to comply with the instructions for use of the product. The equipment must be used as described in the user manual, with particular regard to the intended use, using only accessories as specified by the manufacturer. No modification or repair can be performed.

## Disclaimer

No part of this manual may be reproduced in any form or by any means, electronic, mechanical, recording, or otherwise, without the prior written permission of CAEN SpA.

The information contained herein has been carefully checked and is believed to be accurate; however, no responsibility is assumed for inaccuracies. CAEN SpA reserves the right to modify its products specifications without giving any notice; for up to date information please visit [www.caen.it](http://www.caen.it).

## Made in Italy

We remark that all our boards have been designed and assembled in Italy. In a challenging environment where a competitive edge is often obtained at the cost of lower wages and declining working conditions, we proudly acknowledge that all those who participated in the production and distribution process of our devices were reasonably paid and worked in a safe environment (while this is true for the boards marked "MADE IN ITALY", we cannot guarantee for third-party manufactures).



# Index

<b>Purpose of this Manual</b> . . . . .	<b>2</b>
<b>Change document record</b> . . . . .	<b>2</b>
<b>Symbols, abbreviated terms and notation</b> . . . . .	<b>3</b>
<b>Reference Documents</b> . . . . .	<b>4</b>
<b>Manufacturer Contacts</b> . . . . .	<b>5</b>
<b>Limitation of Responsibility</b> . . . . .	<b>5</b>
<b>Disclaimer</b> . . . . .	<b>5</b>
<b>Made in Italy</b> . . . . .	<b>5</b>
<b>1 Introduction</b> . . . . .	<b>12</b>
1.1 Coincidences: the traditional approach . . . . .	13
<b>2 Coincidences with CAEN DPP firmware</b> . . . . .	<b>17</b>
2.1 DPP Normal Acquisition in Digitizers of the First Generation . . . . .	18
2.2 DPP Coincidence mode in Digitizers of the First Generation . . . . .	20
2.2.1 Resolving time . . . . .	23
2.2.2 Double coincidence . . . . .	23
2.3 Channel correlations with CAEN DPP Firmware in Digitizers of the Second Generation . . . . .	24
<b>3 CoMPASS Software for coincidences with CAEN DPP firmware</b> . . . . .	<b>25</b>
3.1 Off-line Coincidences . . . . .	25
3.2 On-line By Software Coincidences . . . . .	25
3.3 On-line onboard Coincidences . . . . .	29
<b>4 Examples for DPP-PSD firmware</b> . . . . .	<b>31</b>
4.1 How to configure the registers - 720 and 751 series with DPP-PSD . . . . .	31
4.2 Examples for 720-751 series . . . . .	34
4.2.1 Coincidence between channel 0 and channel 1 (ch0 & ch1) . . . . .	34
4.2.2 Anti-coincidence between two channels (ch0 $\oplus$ ch1) . . . . .	34
4.2.3 Coincidence among four channels (ch0 & ch1 & ch2 & ch3) . . . . .	34
4.2.4 Coincidence between channel 0 and channel 1, coincidence between channel 2 and channel 3 (ch0 & ch1), (ch2 & ch3) . . . . .	35
4.2.5 Coincidence among eight channels (ch0 & ch1 & ch2 & ch3 & ch4 & ch5 & ch6 & ch7) - VME only . . . . .	36
4.3 How to configure the registers - 725 and 730 series with DPP-PSD . . . . .	37
4.4 Examples for 725-730 series . . . . .	39
4.4.1 Coincidence between channel 0 and channel 1 (ch0 & ch1) . . . . .	39
4.4.2 Coincidence inside the couples (ch0 & ch1), (ch2 & ch3), (ch4 & ch5), (ch6 & ch7) . . . . .	39
4.4.3 Anti-coincidence between PAIRED channels: (ch0 $\oplus$ ch1), (ch2 $\oplus$ ch3), (ch4 $\oplus$ ch5), (ch6 $\oplus$ ch7) . . . . .	40
4.4.4 Coincidence inside the couples (ch0 & ch1), (ch2 & ch3), (ch4 & ch5), (ch6 & ch7), (ch8 & ch9), (ch10 & ch11), (ch12 & ch13), (ch14 & ch15) - VME only . . . . .	41
4.4.5 Coincidence among four channels (four different couples, ch0 & ch2 & ch4 & ch6) . . . . .	43
4.4.6 Coincidence among four channels (two different couples, ch0 & ch1 & ch2 & ch3) . . . . .	44

4.4.7	Coincidence among four channels plus four channels (ch0 & ch1 & ch4 & ch5, ch2 & ch3 & ch6 & ch7) . . . . .	45
4.4.8	Coincidence between couples of channels in OR: (ch0    ch1) & (ch2    ch3) & (ch4    ch5) & (ch6    ch7) & (ch8    ch9) & (ch10    ch11) & (ch12    ch13) & (ch14    ch15) - VME only . . . . .	46
4.4.9	Coincidence among eight channels (eight different couples, ch0 & ch2 & ch4 & ch6 & ch8 & ch10 & ch12 & ch14) - VME only . . . . .	48
4.4.10	Coincidence among eight channels (four different couples, ch0 & ch1 & ch2 & ch3 & ch4 & ch5 & ch6 & ch7) . . . . .	49
4.4.11	Coincidence among sixteen channels (ch0 & ch1 & ch2 & ch3 & ch4 & ch5 & ch6 & ch7 & ch8 & ch9 & ch10 & ch11 & ch12 & ch13 & ch14 & ch15) - VME only . . . . .	50
4.4.12	Coincidence between ch6 and ch7, in AND with at least one AND of the other pairs: ((ch6 & ch7) & (ch0 & ch1))    ((ch6 & ch7) & (ch2 & ch3))    ((ch6 & ch7) & (ch4 & ch5)) . . . . .	51
4.4.13	Coincidence between ch2 and ch3, in AND with at least one of other 4 channels (ch4-5-6-7): (ch2 & ch3) & (ch4    ch5    ch6    ch7) . . . . .	52
4.4.14	Majority of at least three channels among couples of the whole board (sixteen channels) - VME only . . . . .	54
4.4.15	Majority of two or more couples for the whole board (eight couples) - VME only . . . . .	56
4.4.16	Global Trigger to all Channels by the coincidence between ch0 and ch1 . . . . .	58
4.4.17	Global Trigger on ch0, ch1, ch2, ch3 as OR of their Trigger Request and Global Trigger on ch4, ch5, ch6, ch7 as OR of their Trigger Request . . . . .	58
4.4.18	LVDS Global trigger - VME only . . . . .	59
4.5	Custom configuration requests from users . . . . .	60
4.5.1	Coincidence among four channels (ch0 & ch1 & ch4 & ch5) . . . . .	60
4.5.2	Coincidence among four channels (ch0 & ch1 & ch6 & ch7) . . . . .	60
4.5.3	Coincidence among four channels (ch2 & ch3 & ch4 & ch5) . . . . .	61
4.5.4	Coincidence among four channels (ch2 & ch3 & ch6 & ch7) . . . . .	61
4.6	How to configure the registers - DT5790 with DPP-PSD . . . . .	63
4.7	Examples for DT5790 . . . . .	63
4.7.1	Coincidence between channel 0 and channel 1 (ch0 & ch1) . . . . .	63
4.7.2	Anti-coincidence between channel 0 and channel 1 (ch0 $\oplus$ ch1) . . . . .	64
<b>5</b>	<b>Examples for DPP-PHA firmware . . . . .</b>	<b>65</b>
5.1	How to configure the registers - 724, 781 and 782 series with DPP-PHA . . . . .	65
5.2	Examples for 724, 781 series and V1782 with DPP-PHA . . . . .	68
5.2.1	Coincidence between channel 0 and channel 1 (ch0 & ch1) . . . . .	68
5.2.2	Anti-coincidence between channel 0 and channel 1 (ch0 $\oplus$ ch1) . . . . .	68
5.2.3	Coincidence among four channels (ch0 & ch1 & ch2 & ch3) . . . . .	68
5.2.4	Odd channels validated by the coincidence with the previous even channel . . . . .	69
5.2.5	Anticoincidence among ch0 and the OR of ch1, ch2, ch3 (ch0 $\oplus$ (ch1    ch2    ch3)) . . . . .	69
5.2.6	Majority of at least two channels among three . . . . .	69
5.3	How to configure the registers - 780 series with DPP-PHA . . . . .	70
5.4	Examples for 780 series with DPP-PHA . . . . .	70
5.4.1	Coincidence between channel 0 and channel 1 (ch0 & ch1) . . . . .	70
5.4.2	Anti-coincidence between channel 0 and channel 1 (ch0 $\oplus$ ch1) . . . . .	70
5.5	How to configure the registers - 725 and 730 series with DPP-PHA . . . . .	71
5.6	Examples for 725-730 series with DPP-PHA . . . . .	75
5.6.1	Coincidence between channel 0 and channel 1 (ch0 & ch1) . . . . .	75
5.6.2	Coincidence inside the couples (ch0 & ch1), (ch2 & ch3), (ch4 & ch5), (ch6 & ch7) . . . . .	75
5.6.3	Coincidence inside the couples (ch0 & ch1), (ch2 & ch3), (ch4 & ch5), (ch6 & ch7), (ch8 & ch9), (ch10 & ch11), (ch12 & ch13), (ch14 & ch15) - VME only . . . . .	76
5.6.4	Coincidence between channels of different couples (ch0 & ch2) . . . . .	77
5.6.5	Coincidence among four channels (four different couples, ch0 & ch2 & ch4 & ch6) . . . . .	77
5.6.6	Coincidence among four channels (two different couples, ch0 & ch1 & ch2 & ch3) . . . . .	78

5.6.7	Coincidence among eight channels (eight different couples, ch0 & ch2 & ch4 & ch6 & ch8 & ch10 & ch12 & ch14) - VME only . . . . .	79
5.6.8	Coincidence among eight channels (four different couples, ch0 & ch1 & ch2 & ch3 & ch4 & ch5 & ch6 & ch7) . . . . .	80
5.6.9	Coincidence among sixteen channels (ch0 & ch1 & ch2 & ch3 & ch4 & ch5 & ch6 & ch7 & ch8 & ch9 & ch10 & ch11 & ch12 & ch13 & ch14 & ch15) - VME only . . . . .	80
5.6.10	Majority of at least three channels among couples of the whole board (sixteen channels) - VME only . . . . .	82
5.6.11	Majority of at least three couples among four couples (eight channels) . . . . .	83
5.6.12	Anti-coincidence between PAIRED channels: (ch0 ⊕ ch1), (ch2 ⊕ ch3), (ch4 ⊕ ch5), (ch6 ⊕ ch7), (ch8 ⊕ ch9), (ch10 ⊕ ch11), (ch12 ⊕ ch13), (ch14 ⊕ ch15) - VME only . . . . .	84
5.7	External Trigger in coincidence with channels for 725-730 series with DPP-PHA . . . . .	85
5.7.1	External trigger in coincidence with ch0 and ch1 (ch0 & ch1 & EXT-TRG gate) . . . . .	85
5.7.2	External trigger in coincidence with ch0 and ch2 (ch0 & ch2 & EXT-TRG gate) . . . . .	86
5.7.3	External Trigger in coincidence with eight channels (ch0 & ch1 & ch2 & ch3 & ch4 & ch5 & ch6 & ch7) . . . . .	86
<b>6</b>	<b>How to Veto the acquisition . . . . .</b>	<b>88</b>
6.1	Signal on channel 0 vetoed by the other channels . . . . .	88
6.1.1	Veto on ch0 - DPP-PSD for 720 and 751 series . . . . .	89
6.1.2	Veto on ch0 - DPP-PSD for 725 and 730 series . . . . .	90
6.1.3	LVDS Veto on all couples - DPP-PSD for 725 and 730 series - VME only . . . . .	90
6.1.4	Veto on ch1 by ch0 - DPP-PSD for DT5790 . . . . .	91
6.1.5	Veto on ch0 by ch1 - DPP-PSD for DT5790 . . . . .	91
6.1.6	Veto on ch0 - DPP-PHA for 724, 781 series and V1782 . . . . .	92
6.1.7	Veto on all channels by ch0 - DPP-PHA for 724, 781 series and V1782 . . . . .	92
6.1.8	LVDS Veto as anticoincidence on all channels - DPP-PHA only for V1782 . . . . .	92
6.1.9	Veto on ch0 - DPP-PHA for 725 and 730 series . . . . .	93
6.1.10	LVDS Veto as anticoincidence on all channels - DPP-PHA for 725 and 730 series - VME only . . . . .	93
6.2	Veto for the couples of channels (725-730 only) . . . . .	95
6.2.1	Coincidence inside the VME couples (ch-i & ch-i+1) vetoed by the majority of two or more couples (DPP-PSD) . . . . .	96
6.2.2	Coincidence inside the VME couples (ch-i & ch-i+1) vetoed by the majority of two or more couples (DPP-PHA) . . . . .	98
6.3	External Trigger to veto (gate) the acquisition . . . . .	99
6.3.1	Veto from External Trigger - 720 and 751 series with DPP-PSD . . . . .	100
6.3.2	Veto from External Trigger - 725 and 730 series with DPP-PSD . . . . .	100
6.3.3	Veto from External Trigger - 724, 780, 781 series and V1782 with DPP-PHA . . . . .	101
6.3.4	Veto from External Trigger - 725 and 730 series with DPP-PHA . . . . .	101
<b>7</b>	<b>Miscellaneous . . . . .</b>	<b>102</b>
7.1	Examples for 720 - 751 series, and DT5790 - DPP-PSD . . . . .	102
7.1.1	PSD cut value Online on ch0 . . . . .	102
7.2	Examples for 725 - 730 series - DPP-PSD . . . . .	102
7.2.1	Global Trigger to all Channels by signal on ch0 . . . . .	102
7.2.2	Global Trigger as OR of the channels (Not Using the Global Trigger Mask) . . . . .	103
7.2.3	External Trigger makes ch0 and ch1 triggering . . . . .	103
7.2.4	External Trigger from TRG-IN to send a Global Trigger to all Channels . . . . .	103
7.2.5	PSD cut value Online on ch0 and ch1 . . . . .	104
7.2.6	Trigger Propagation to TRG_OUT (from Trigger Request in ch0) . . . . .	104
7.2.7	Trigger Propagation to TRG_OUT LEMO and LVDS - VME only . . . . .	104
7.3	Examples for 724, 780, 781 and V1782 series - DPP-PHA . . . . .	106
7.3.1	Propagate out the trigger requests of the OR of the channels . . . . .	106
7.3.2	Global Trigger by any channel . . . . .	106
7.3.3	Global Trigger by ch0 . . . . .	106
7.4	Examples for 725 and 730 series - DPP-PHA . . . . .	107



7.4.1	LVDS propagate out the trigger requests - VME only . . . . .	107
7.4.2	Propagate out the trigger requests of the OR of the channels inside the couples . . . . .	107
7.4.3	External Trigger from TRG-IN to send a Global Trigger to all Channels . . . . .	107
<b>8</b>	<b>Coincidences with DPP-PSD and DPP-PHA firmwares in digitizers of the Second generation . . . . .</b>	<b>109</b>
8.1	How to configure the parameters - 2730, 2740, 2745 and 2751 series with DPP-PSD and DPP-PHA . . . . .	110
8.2	Examples for 2730, 2740, 2745 series with DPP-PSD and DPP-PHA and for 2751 series with DPP-PSD . . . . .	114
8.2.1	Coincidence between channel 0 and channel 1 (ch0 & ch1) - using "Ch64Trigger" . . . . .	114
8.2.2	Coincidence between channel 0 and any other channel (ch0 & ch1), (ch0 & ch2), ..., (ch0 & ch63) - using "Ch64Trigger" . . . . .	114
8.2.3	Majority among all channels to send a Global Trigger to the whole board . . . . .	114
8.2.4	External Trigger from TRG-IN to send a Global Trigger to all channels . . . . .	115
8.2.5	External Trigger as Gate for all channels . . . . .	115
8.2.6	External Trigger to Veto all channels . . . . .	115
<b>9</b>	<b>Coincidences with Waveform Recording Firmware in Digitizers of the First Generation . . . . .</b>	<b>117</b>
9.1	How to configure the registers - 720, 724, 740 and 751 series with WR . . . . .	117
9.2	Examples for 720, 724, 740, 751 series with WR . . . . .	119
9.2.1	Coincidence between channel 0 and channel 1 (ch0 & ch1) . . . . .	119
9.2.2	Coincidence among channel 0, channel 1 and channel 2 (ch0 & ch1 & ch2) . . . . .	119
9.2.3	Coincidence among eight channels (ch0 & ch1 & ch2 & ch3 & ch4 & ch5 & ch6 & ch7) - VME only . . . . .	119
9.3	How to configure the registers - 725, 730 series with WR . . . . .	120
9.4	Examples for 725, 730 series with WR . . . . .	120
9.4.1	Coincidence between channel 0 and channel 1 (ch0 & ch1) . . . . .	120
9.4.2	Coincidence between channels of different couples (ch0 & ch2) . . . . .	121
9.4.3	Coincidence among channels of different couples (ch0 & ch2 & ch4 & ch6) . . . . .	121
9.4.4	Coincidence among channels of different couples (ch1 & ch3 & ch5 & ch7) . . . . .	121
9.4.5	Coincidence among different couples (ch0 & ch1 & ch2 & ch3) . . . . .	122
9.4.6	Coincidence among eight channels (ch0 & ch1 & ch2 & ch3 & ch4 & ch5 & ch6 & ch7) . . . . .	122
9.4.7	Coincidence among all channels (ch0 & ch1 & ch2 & ch3 & ch4 & ch5 & ch6 & ch7 & ch8 & ch9 & ch10 & ch11 & ch12 & ch13 & ch14 & ch15) - VME only . . . . .	123
<b>10</b>	<b>Coincidences with Firmware Scope in Digitizers of the Second generation . . . . .</b>	<b>124</b>
10.1	WaveDump2 - Manual Controller Panel . . . . .	124
10.1.1	Board Setting Configuration . . . . .	125
10.2	How to configure the parameters - 2730, 2740, 2745 and 2751 series with Firmware Scope . . . . .	126
10.3	Examples for 2730, 2740, 2745 and 2751 series with Firmware Scope . . . . .	129
10.3.1	Coincidence between channel 0 and channel 1 (ch0 & ch1) - using "ITLA" . . . . .	129
10.3.2	Majority among first 8 channels . . . . .	129
10.3.3	External Trigger from TRG-IN to send a Global Trigger to all channels . . . . .	130
10.3.4	External Trigger as Gate for all channels . . . . .	130
10.3.5	External Trigger to Veto all channels . . . . .	130
<b>11</b>	<b>Technical Support . . . . .</b>	<b>131</b>

# List of Figures

<b>Fig. 1.1</b>	Analog coincidence scheme. The two pulses are summed and the discrimination level is set on the sum pulse, corresponding to the coincidence of the two. . . . .	13
<b>Fig. 1.2</b>	Amplitude walk problem related to the leading edge timing method . . . . .	14
<b>Fig. 1.3</b>	The zero crossing timing of bipolar pulses . . . . .	14
<b>Fig. 1.4</b>	The constant fraction timing method. From top to bottom the input pulse is split into an attenuated signal and a delayed and inverted one. The two are summed and the zero crossing time is taken as the time trigger. . . . .	15
<b>Fig. 1.5</b>	Schematic example showing how to record coincidences of the same event from two different detectors	15
<b>Fig. 1.6</b>	Time spectrum for a radioactive source as in the system acquisition configuration of <b>Fig. 1.5</b> . . . . .	16
<b>Fig. 1.7</b>	The coincidence rate vs delay curve . . . . .	16
<b>Fig. 2.1</b>	Example of a digital chain for coincidences with CAEN digitizers . . . . .	17
<b>Fig. 2.2</b>	DPP firmware functional description in the normal acquisition mode . . . . .	18
<b>Fig. 2.3</b>	The generation of over-threshold (OVTH) and shaped trigger (ST) signals for channel $i$ . They are all generated in time with the trigger (Leading Edge Discrimination timing in figure). . . . .	19
<b>Fig. 2.4</b>	Memory organization of 725-730 couples . . . . .	19
<b>Fig. 2.5</b>	DPP firmware functional description when the coincidence mode is enabled . . . . .	20
<b>Fig. 2.6</b>	Generation of the TRG_REQ and TVAW signals for channel $i$ . The $T_{TVAW}$ includes also a latency time $T_{LAT}$ for the signal to go back and forth from mezzanine to mother board. . . . .	21
<b>Fig. 2.7</b>	Generation of $ITRG(0) = ITRG(1) = (TRG\_REQ(0) \& TRG\_REQ(1))$ for channel 0 and channel 1 . . . . .	22
<b>Fig. 2.8</b>	Trigger Management inside couple 0 of 725-730 digitizer series. Couple 0 is made of channel 0 and channel 1. The same applies for the other couples of the 725-730 digitizers. . . . .	22
<b>Fig. 2.9</b>	Example of a "double coincidence" event . . . . .	23
<b>Fig. 2.10</b>	Scheme of Coincidence Mask logic . . . . .	24
<b>Fig. 3.1</b>	CoMPASS Acquisition Tab . . . . .	26
<b>Fig. 3.2</b>	CoMPASS Time Selection Tab . . . . .	27
<b>Fig. 3.3</b>	CoMPASS Rejection Tab . . . . .	28
<b>Fig. 3.4</b>	CoMPASS Trigger/Veto/Coincidences Tab for V17xx/N67xx/DT57xx digitizers and MCAs (left), and 27xx digitizer (right). . . . .	29
<b>Fig. 4.1</b>	Local Trigger Management inside couple 0 of 725-730 digitizer series. Couple 0 is made of channel 0 and channel 1. The same applies for the other couples of the 725-730 digitizers. . . . .	37
<b>Fig. 5.1</b>	Local Trigger Management inside couple 0 of 725-730 digitizer series. Couple 0 is made of channel 0 and channel 1. The same applies for the other couples of the 725-730 digitizers. . . . .	71
<b>Fig. 6.1</b>	Muon Shield scheme: an emitting source is detected by a particular detector. Three scintillators act as a muon shield. Events passing both in the shield and in the detector are removed. . . . .	89
<b>Fig. 6.2</b>	Example of two-sided detectors that acquire in coincidence . . . . .	96
<b>Fig. 8.1</b>	Individual Trigger logic scheme. . . . .	111
<b>Fig. 10.1</b>	Digitizer 2.0 - Main settings tab of the Board settings section. . . . .	125
<b>Fig. 10.2</b>	Digitizer 2.0 - Advanced table of the Board settings section. . . . .	125
<b>Fig. 10.3</b>	Individual Trigger logic scheme. . . . .	127

## List of Tables

<b>Tab. 1.1</b>	List of digitizers/MCAs running DPP firmware that support on-line onboard coincidences . . . . .	12
-----------------	--	----

# 1 Introduction

Many physics applications require the measurement of events from a combination of different detectors, as for example in gamma ray and time spectroscopy, in the case of detection of particles emitted in cascade from the same nucleus, or the measurement of cosmic rays, where analyzing the coincidences, or generally signals from correlated acquisition channels, can significantly reduce the background.

All these experiments usually require the precise measurement of time and delay time, and this is given by digitizers and Multi Channel Analyzer (MCA) acquisition systems.

Channel correlation can be performed in digitizers and MCAs through direct coincidences or via the propagation of the trigger request, followed by a trigger validation in coincidence.

A coincidence detecting system determines when the time difference between two events is less than a certain time window, known as the *resolving time*, and saves the corresponding event.

There are three methods to make direct coincidences:

- **Off-line:** all detected events, with their time stamp, are saved and analyzed through dedicated software. The user can write his/her own codes and implement his/her desired requirements for coincidences, by comparing the time stamp information.
- **On-line by software:** all detected events are analyzed on-line through a dedicated software.
- **On-line onboard:** the digitizer is setup to record and pass to the Data Acquisition (DAQ) software on-line coincident events only. This method is recommended for high input rates, to reduce the readout throughput.

CAEN digitizers with proper Digital Pulse Processing/Waveform Recording (DPP/WR) firmware allow to perform the three methods.

Off-line and On-line By Software coincidences are supported by both firmware types in all CAEN digitizers/MCA series.

On-line onboard coincidences can be performed with the DPP firmware, and in particular with DPP-Pulse Shape Discrimination (DPP-PSD) and DPP-Pulse Height Analysis (DPP-PHA) firmware types, only in the cases listed in **Tab. 1.1**.

Digitizer/MCA	DPP Firmware
720 series	DPP-PSD
724 series	DPP-PHA
725/725S series	DPP-PSD and DPP-PHA
730/730S series	DPP-PSD and DPP-PHA
751 series	DPP-PSD
DT2740/VX2740 <b>(Open FPGA)</b>	DPP-PSD and DPP-PHA
DT2745/VX2745 <b>(Open FPGA)</b>	DPP-PSD and DPP-PHA
DT2730/VX2730 <b>(Open FPGA)</b>	DPP-PSD and DPP-PHA
DT2751/VX2751 <b>(Open FPGA)</b>	DPP-PSD
DT5780-N6780	DPP-PHA
DT5781-N6781	DPP-PHA
V1782	DPP-PHA
DT5790	DPP-PSD

**Tab. 1.1:** List of digitizers/MCAs running DPP firmware that support on-line onboard coincidences



**Note:** DPP-QDC for 740 series, DPP-ZLE, and DPP-DAW do not support on-line onboard coincidences.

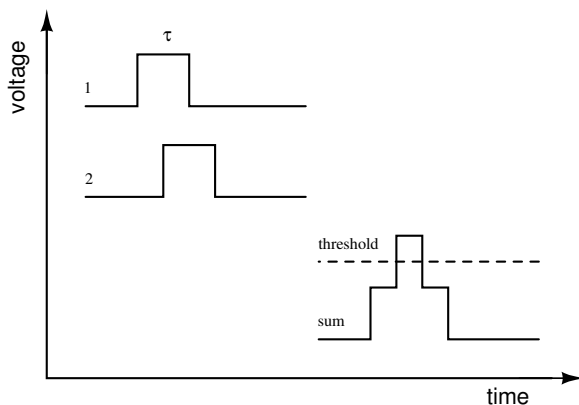
The Waveform Recording (WR) firmware supports on-line onboard coincidences for the 720, 724, 740, 751 and 725-730 digitizers series. With this type of firmware, coincidences can only be performed using the majority logic. Please refer to Sec. 9 for more details.

On-line onboard coincidences can be performed also with the digitizers of second generation (2740, 2745, 2730 and 2751 series) with WR firmware installed, as described in Sec. 10.

Trigger propagation is a specific type of coincidence that can be performed with both digitizers and MCAs. This channel correlation mode will be presented in some of the examples provided in this guide. Detailed instructions for the different coincidence settings are explained in the following chapters.

## 1.1 Coincidences: the traditional approach

Consider the case of coincidence between two input pulses (see **Fig. 1.1**). In the traditional analog approach the two pulses are summed, and the system triggers on the sum pulse, saving only the coincident event. The width  $\tau$  of the two inputs determines the time window where a coincidence can occur, corresponding to a *resolving time* of  $2\tau$ .



**Fig. 1.1:** Analog coincidence scheme. The two pulses are summed and the discrimination level is set on the sum pulse, corresponding to the coincidence of the two.

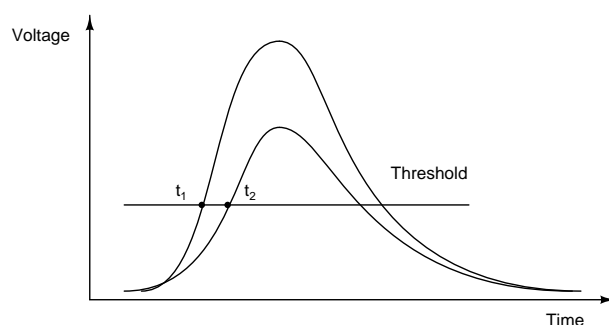
The use of timing discriminators, analog-to-digital converters (ADC), and MCAs allows to trigger on time and obtain a coincidence unit.

There are three ways to perform time trigger, i.e. to generate logic pulses whose leading edge corresponds to the time of occurrence of the input pulse **[RD1]**:

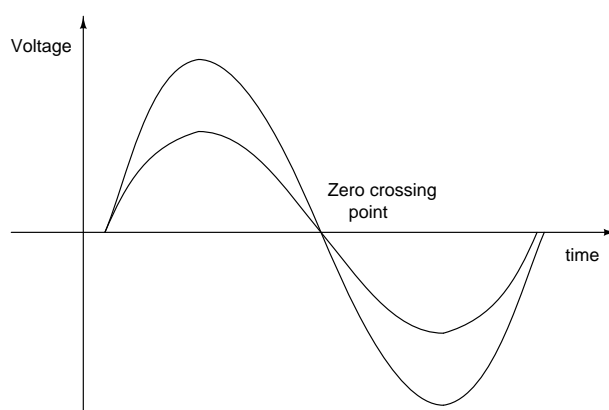
- leading edge;
- crossover;
- constant fraction.

The **leading edge** timing consists on generating a logic pulse when the leading edge of a pulse crosses a fixed discrimination level. Though this is the easiest method, it suffers from the problem of the amplitude walk (the difference in amplitude width that can lead to different time triggers, see **Fig. 1.2**), and the rise time variation due to possible detector non-uniformities.

In case of bipolar pulses, the **crossing** timing allows to reduce the amplitude walk issue. Indeed the zero-crossing point (the time when the pulse crosses from the positive to the negative side of the axis) is theoretically independent from the pulse amplitude (see **Fig. 1.3**).

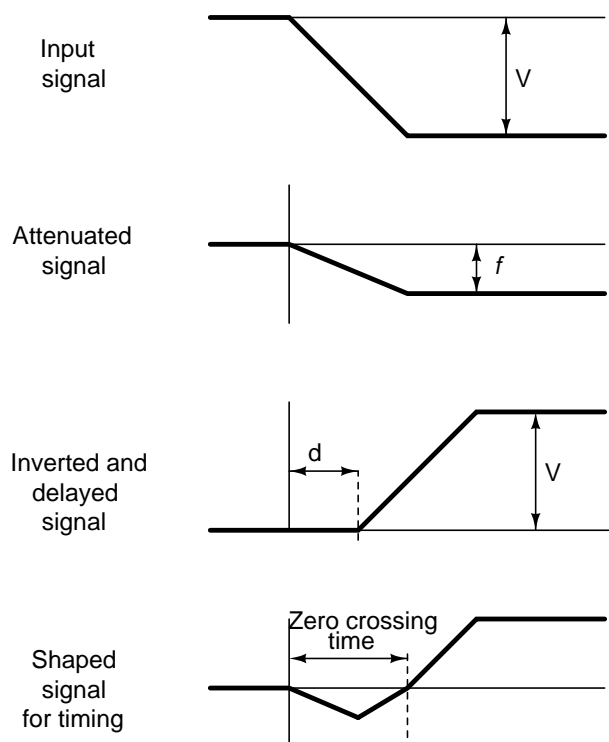


**Fig. 1.2:** Amplitude walk problem related to the leading edge timing method



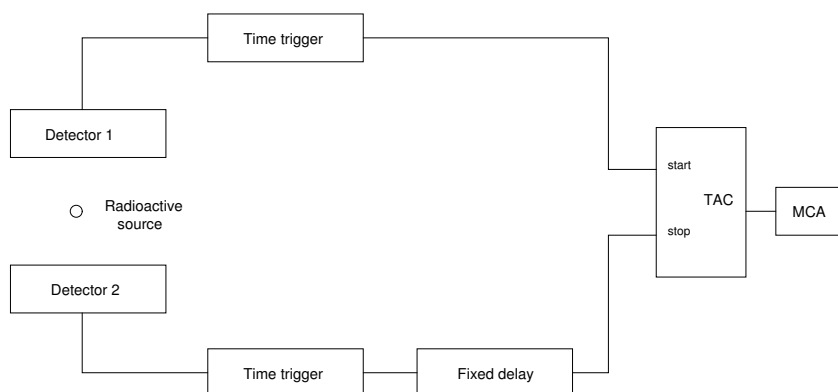
**Fig. 1.3:** The zero crossing timing of bipolar pulses

The last method consist of triggering on a **constant fraction** of the peak pulse amplitude. For pulses having the same shape, this point is independent on pulse amplitude. The steps to be performed are the following: the input waveform is attenuated by a factor  $f$  equal to the desired timing fraction of full amplitude, the signal is inverted and delayed by a time  $d$  equal to the time it takes for the pulse to rise from the constant fraction level to the pulse peak; the latest two signals are summed to produce a bipolar pulse. Its zero crossing is at the fraction  $f$  of the pulse and it is taken as the time trigger (see **Fig. 1.4**). The rise time of the signals must be equal in order to apply this method.



**Fig. 1.4:** The constant fraction timing method. From top to bottom the input pulse is split into an attenuated signal and a delayed and inverted one. The two are summed and the zero crossing time is taken as the time trigger.

A simple system to register coincident events is shown in **Fig. 1.5 [RD2]**.

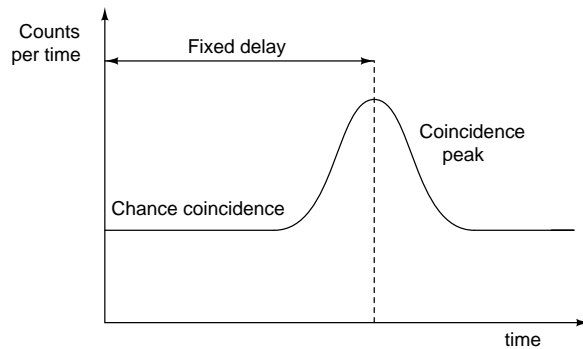


**Fig. 1.5:** Schematic example showing how to record coincidences of the same event from two different detectors

Suppose that a radioactive source emits events revealed with two detectors at the same time. The events are detected by two timing systems (chosen from one of those previously described) that record their time of detection. One of the two branches is delayed by a fixed amount of time<sup>1</sup>. The two triggers arrive to a time-to-amplitude converted (TAC), a device that produces an output pulse with an amplitude proportional to the time interval between input "start" and "stop" pulses. Finally a MCA provides a differential amplitude distribution, also called the "time spectrum".

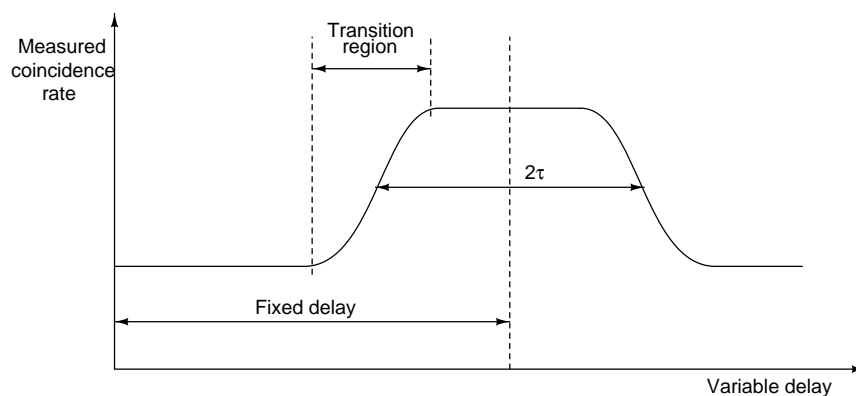
<sup>1</sup>A fixed delay time allows to shift the peak of coincidence events to positive values and to correctly evaluate its resolution.

An example of time spectrum obtained in these conditions is shown in **Fig. 1.6**. The peak corresponds to the coincidence events, shifted from the axis origin by the same fixed delay of **Fig. 1.5**. The offset in the y axis corresponds to random events, as for example the noise from detectors that may fake the coincidence. The distribution of the latter events is usually uniform.



**Fig. 1.6:** Time spectrum for a radioactive source as in the system acquisition configuration of **Fig. 1.5**

Varying the threshold on the TAC output it is possible to specify the coincidence time window, and adding a variable delay in the top branch of **Fig. 1.5** it is possible to obtain the distribution of the coincidence rate versus the relative delay between signals, the so called "coincidence-delay curve" (see **Fig. 1.7**) [RD3]. For a relative delay within the resolving time window, the coincidence rate is equal to the input rate. Outside the resolving time window the coincidence rate is equal to the chance coincidence rate. The transition is sharp for ideal detectors, while real detector goes smoothly to the chance coincidence rate.



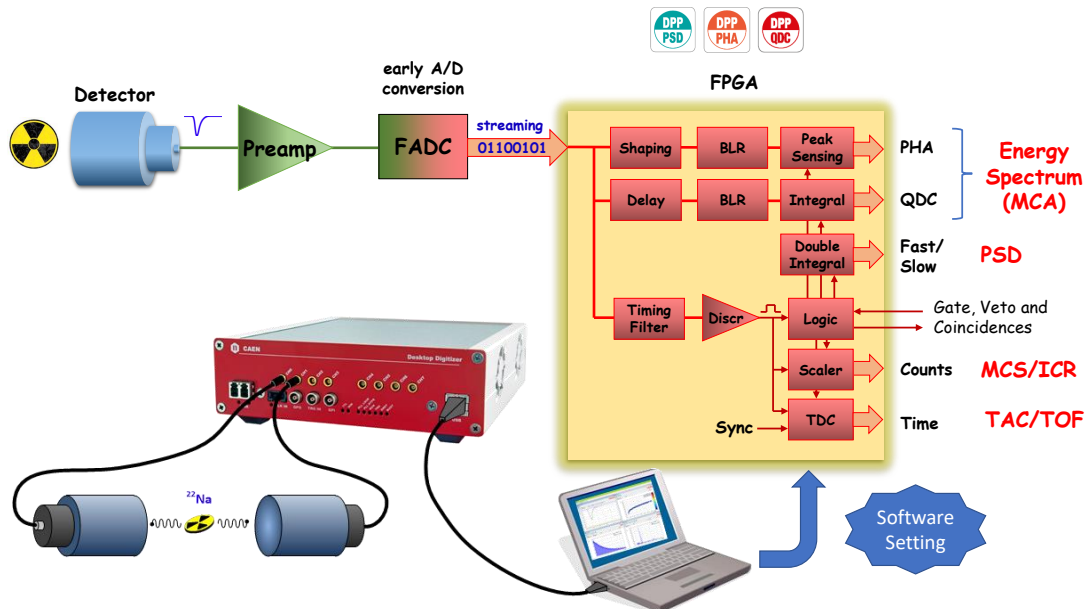
**Fig. 1.7:** The coincidence rate vs delay curve



## 2 Coincidences with CAEN DPP firmware

The use of the latest generations of flash Analog-to-Digital Converters (ADCs) in acquisition boards gives the possibility to convert analog signals with improved resolution and faster sampling speed, thus preserving the information required by the physics experiments. The integrated FPGA is then able to acquire the information from the ADCs and process it, through dedicated algorithms. Those algorithms may be the digital replacement of the traditional analog signal processing, so that the digitizer embeds different functions in one single board. In particular, it is possible to replace timing filters such as Constant Fraction Discriminator (CFD), Shaper Amplifier, Peak Sensing ADC, Charge-to-Digital Converter (QDC), Time-to-Digital Converter (TDC), etc.

The functionalities embedded in a digitizer in case of a coincidence setup are shown in **Fig. 2.1**.



**Fig. 2.1:** Example of a digital chain for coincidences with CAEN digitizers

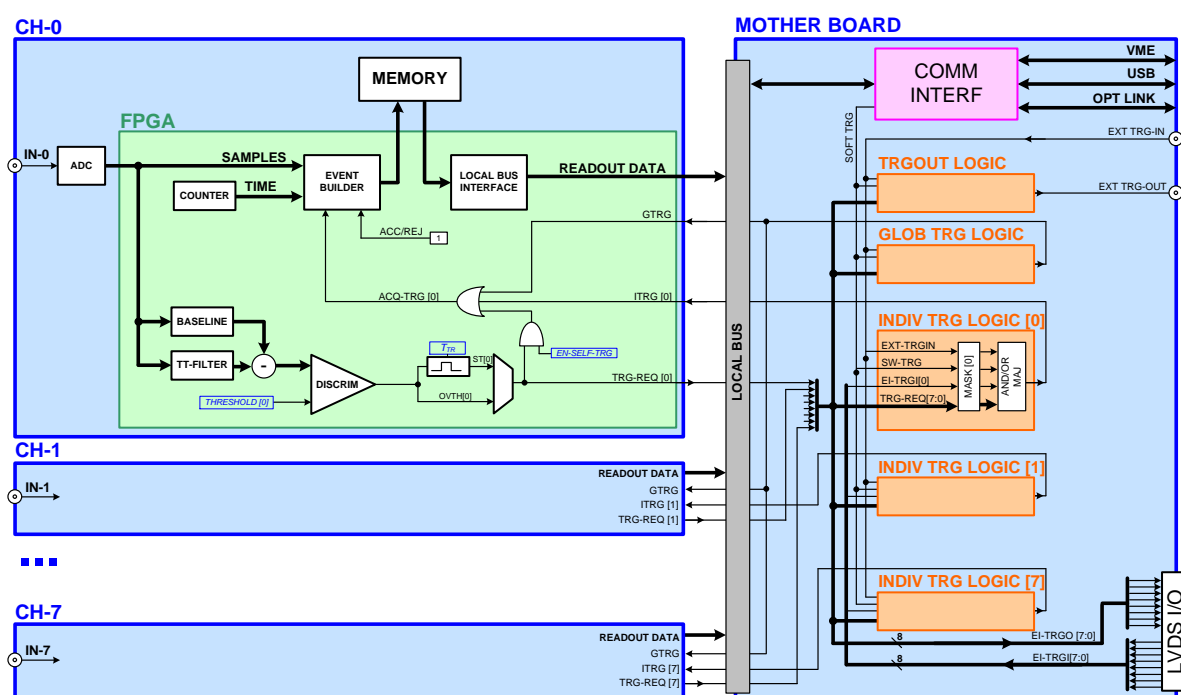
The transition from multiple modules to one single board with multiple features is possible thanks to two firmware categories: WR firmware and DPP firmware, which run on-line on the digitizer. While the WR firmware allows the user to record the waveform samples for further analysis, the DPP one is able to analyze the samples on-line and to provide in real time significant information of the event, like energy, time, and pulse shape.

The scheme of the DPP firmware functionalities in the *normal* and *coincidence* acquisition modes is reported in the next Sections.

A separate Section is dedicated to the digitizers of second generation, that present more flexibility (thanks to the open FPGA) and a higher channel density.

## 2.1 DPP Normal Acquisition in Digitizers of the First Generation

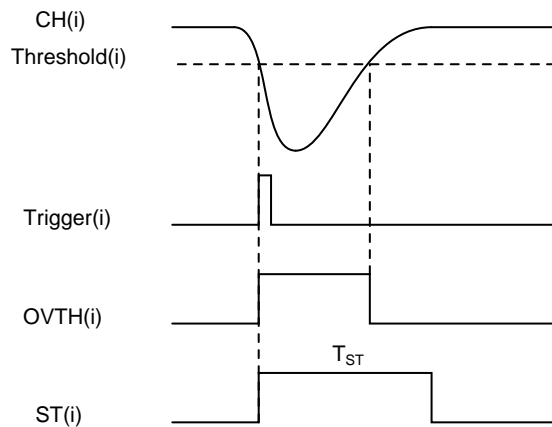
The *normal acquisition mode* (coincidence not enabled) for a VME form factor module (27xx excluded) is described in **Fig. 2.2**. The same considerations are valid also for Desktop (DT) and NIM form factor models, apart for the different number of channels and the absence of the LVDS I/O connector (VME only). Note that the VME TRG-OUT LEMO connector is named GPO in DT/NIM modules. The following coincidence mode description is referred only to channel 0, but can be extended to all channels, since the operating scheme is the same for all.



**Fig. 2.2:** DPP firmware functional description in the normal acquisition mode

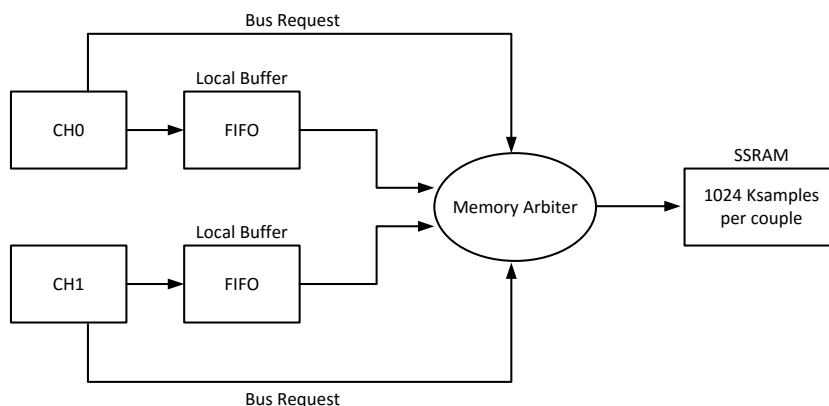
The analog input is converted into a digital signal through an ADC, then it enters into the FPGA and it is sent to the "event builder" that evaluates the arrival time, integrates the charge, and performs all DPP specific calculations.

The TT-Filter (Trigger and Timing Filter) may be Leading Edge Discriminator (LED), and CFD in case of DPP-PSD firmware, or RC-CR<sup>2</sup> in case of DPP-PHA firmware **[RD4]**. After the TT-Filter it is possible to select through a multiplexer the output itself, or a logic pulse of adjustable time width  $T_{ST}$ . The latter is the *shaped trigger* that enables the "trigger request" (TRG\_REQ) for the data acquisition (bottom branch). The time diagram of the mentioned signals is shown in **Fig. 2.3**. The same TRG\_REQ can also be sent to the mother board for additional operations, as detailed below.



**Fig. 2.3:** The generation of over-threshold (OVTH) and shaped trigger (ST) signals for channel  $i$ . They are all generated in time with the trigger (Leading Edge Discrimination timing in figure).

In case of 725 and 730 series it has to be introduced the concept of "couples of channels", where couple  $n$  corresponds to channel  $2n$  and channel  $2n+1$ . Channels of the same couple share the same memory SSRAM (see **Fig. 2.4**) and the same TRG\_REQ (see **Fig. 2.8**). Indeed a single TRG\_REQ from the couple is propagated to the mother board FPGA to participate to the coincidence logic.



**Fig. 2.4:** Memory organization of 725-730 couples

The *event builder* acquires the event and makes all proper calculations as soon as it receives the "acquisition trigger" (ACQ\_TRG). Then it waits for the "trigger validation" (TRG\_VAL) to write the event in the SSRAM memory. In the normal acquisition mode the TRG\_VAL is always 1. The ACQ\_TRG is the logic OR of the TRG\_REQ, the "individual trigger" (ITRG), and the "global trigger" (GTRG).

The ITRG is a logic signal coming from the mother board FPGA through the "individual trigger logic" (ITL), and it has the same multiplicity of the number of channels. Each ITL can receive as input the TRG\_REQ from all channels, the signal from line 0 to 7 of the LVDS I/O connector, the software trigger (SOFT\_TRG), and the external trigger (EXT TRG-IN). It can perform a set of logic operations: AND, OR, and MAJORITY, where AND/OR are the classic logical operations, and MAJORITY is true when at least a programmable "majority level" number of enabled inputs are in coincidence. The user can set both the majority logic and the majority level. Since there is one ITL for each channel, the output of a single ITL can control only that specific channel.

The "trg-out logic" works as the ITL, but it does not present the same multiplicity. Only one TRG-OUT signal

is generated for the whole board.

The "global trigger logic" (GTL) manages the GTRG and can receive as inputs the TRG\_REQ from all channels, the EXT TRG-IN and the SOFT TRG, while it cannot receive the inputs from the LVDS I/O. At the moment this logic unit can perform only the OR operation. There is one GTL for all channels, since its output triggers all channels simultaneously.

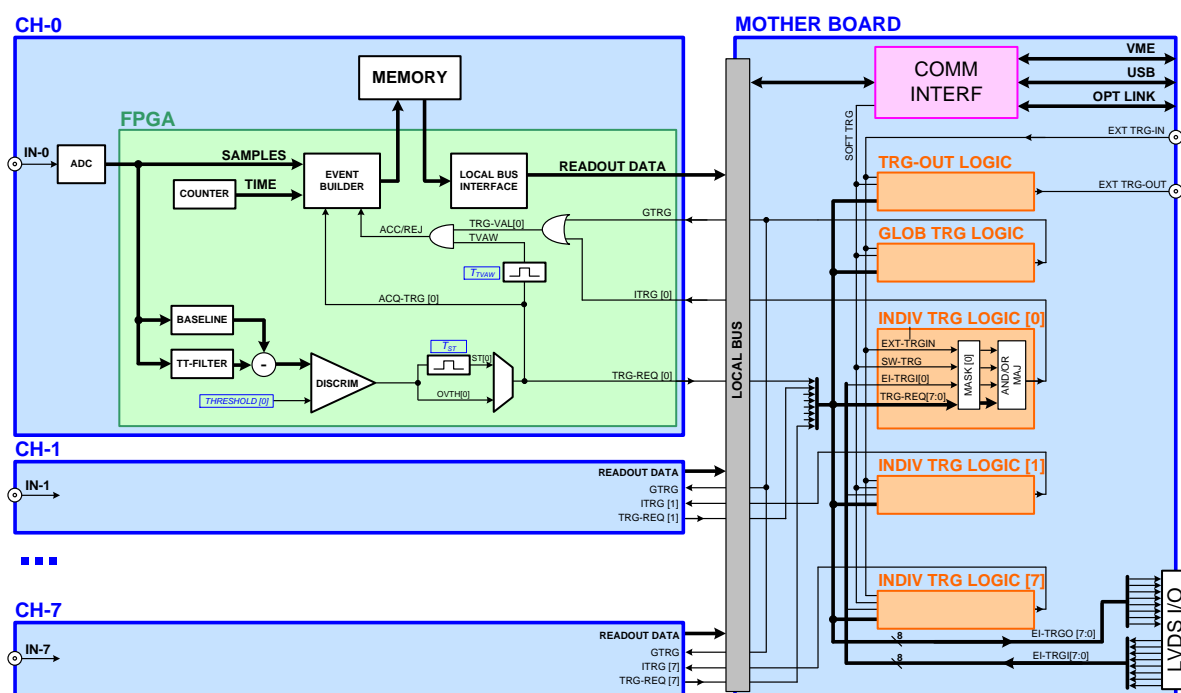
CAEN digitizers allow the use of different boards simultaneously, as well as the use of external modules, to create multi-board systems. For example, it is possible to acquire when both channel X of board N and channel Y of board M detect an input pulse. The trigger request of those channels can be sent out through the output lines (0 to 7) of the LVDS I/O connectors to an external "logic unit" (as the multi-purpose FPGA board V2495 provided by CAEN [RD5], available also in the Desktop form factor [RD6]). The latter implements the logic operations and sends the TRG\_VAL back to the front panel "external trigger in" (EXT TRG-IN) of the VME.

It is also possible to acquire from one channel when other channels from different VME boards fulfil the logic requirements. In this case the input lines (8 to 15) of the LVDS I/O are used to send back the TRG\_VAL from the external logic unit.

To perform the last two cases the synchronization among different boards is required. For more details about the synchronization please refer to Chap. 3 of CoPASS Quick Start Guide [RD7], or to the specific application note [RD8].

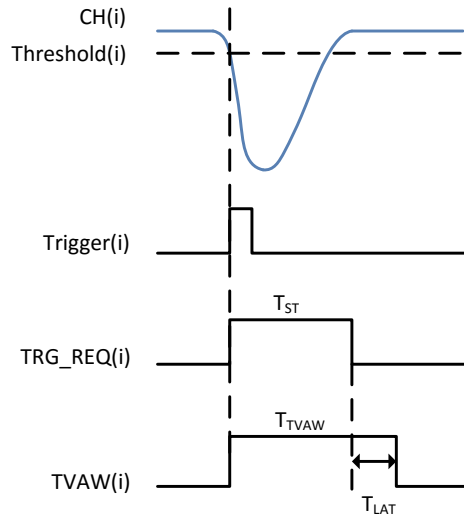
## 2.2 DPP Coincidence mode in Digitizers of the First Generation

In the DPP On-line onboard coincidence mode the acquisition chain is modified as shown in **Fig. 2.5** (27xx digitizers excluded). The channels are denoted by the index  $i$ , that ranges from 0 to 7.



**Fig. 2.5:** DPP firmware functional description when the coincidence mode is enabled

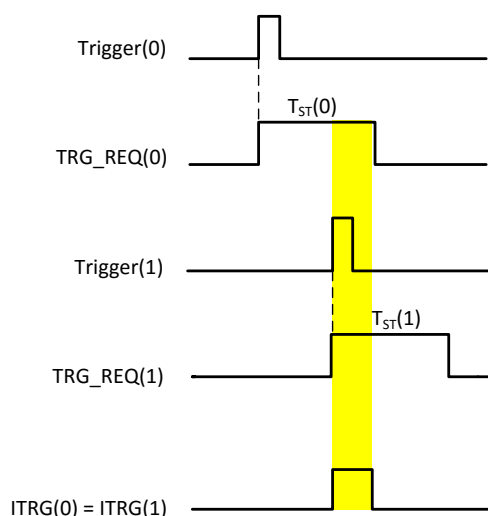
Differences from the normal acquisition mode (refer to the previous section) lie in the channel FPGA. The ACQ\_TRG now corresponds only to the TRG\_REQ (i.e. the event is processed whenever a trigger request arrives), while the TRG\_VAL is the OR between the GTRG and the ITRG(i)<sup>1</sup>. The TRG\_VAL can enable the event writing only if it is in time with the "trigger validation acceptance window" (TVAW). When the logic AND between TRG\_VAL and TVAW is satisfied, the event is saved into the memory. The time width value  $T_{TVAW}$  is set by default as the sum of  $T_{ST}$  plus a fixed amount of time that takes into account the signal latency  $T_{LAT}$  (see Fig. 2.6).



**Fig. 2.6:** Generation of the TRG\_REQ and TVAW signals for channel  $i$ . The  $T_{TVAW}$  includes also a latency time  $T_{LAT}$  for the signal to go back and forth from mezzanine to mother board.

**Fig. 2.7** shows the case where ITRG(0) and ITRG(1) are set as the AND between channel 0 and channel 1 (i.e.  $ITRG(0) = ITRG(1) = (TRG\_REQ(0) \& TRG\_REQ(1))$ ). In this specific case the ITRGs are in time with the TVAW(i), and the TRG\_VAL(i) enables the writing of the  $i$ -th channel into memory.

<sup>1</sup>Each channel presents its own ITRG.

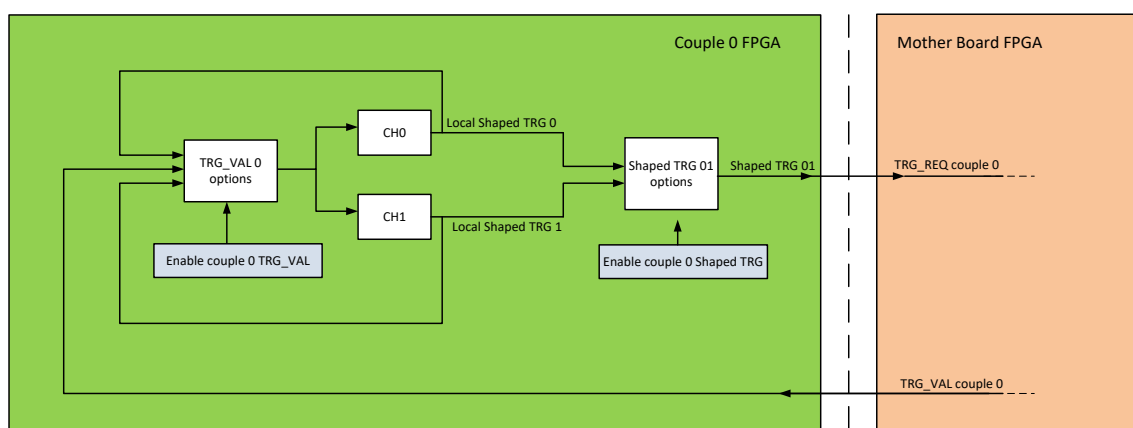


**Fig. 2.7:** Generation of  $ITRG(0) = ITRG(1) = (TRG\_REQ(0) \& TRG\_REQ(1))$  for channel 0 and channel 1

In case of 725-730 series, only one TRG\_REQ per couple is propagated to the mother board to participate to the ITL, GTL, and TRG-OUT Logic. Each channel of the couple generates its local ST, which can be combined to generate the TRG\_REQ of the couple. Options are usually AND/OR/one of the channels. One TRG\_VAL signal arrives for the couple, where it can be programmed to be the AND/OR/crossed/from mother board (refer to Sec. 4.3 of [RD9] and 5.5 of [RD10]).

A great advantage comes in case of coincidences between channels of the same couple, which can be managed inside the channel FPGA, with no propagation to the mother board. Coincidences among channels of different groups can be performed by means of coincidences among couples, since one TRG\_REQ is generated for each couple.

The scheme for 725 and 730 series is shown in **Fig. 2.8**.



**Fig. 2.8:** Trigger Management inside couple 0 of 725-730 digitizer series. Couple 0 is made of channel 0 and channel 1. The same applies for the other couples of the 725-730 digitizers.

## 2.2.1 Resolving time

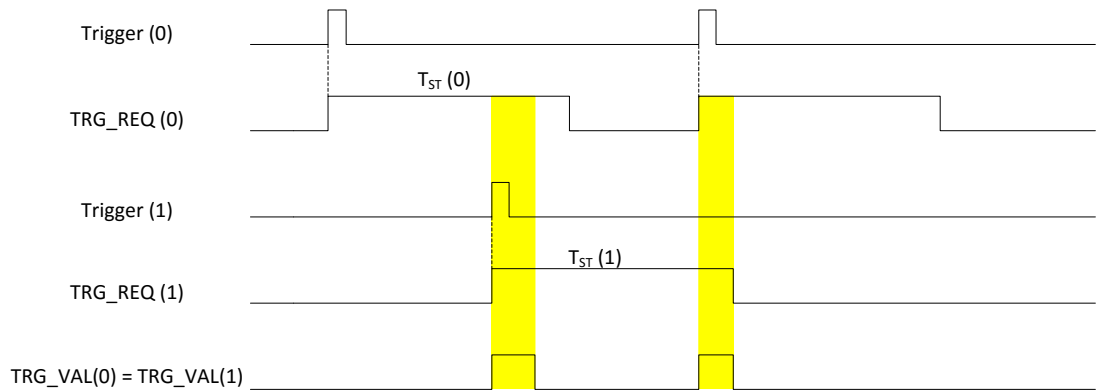
The resolving time  $\tau$  (see the definition in Sec. 1.1) is equal to  $\tau = 2 \cdot T_{ST} - 1 \text{ clk}$  for DPP-PSD, where  $1 \text{ clk} = 16 \text{ ns}$  in case of 725 series, and  $8 \text{ ns}$  for the others, while  $\tau = 2 \cdot T_{ST}$  for DPP-PHA.

## 2.2.2 Double coincidence

When large coincidence windows (i.e. large  $T_{ST}$ ) are used, it may happen that a "double coincidence" event occurs, i.e. two events of the same channel are in coincidence with a single event of another channel.

Consider for simplicity the case of a coincidence between channel 0 and channel 1, where the  $\text{TRG\_VAL}(n)$  signal is set as the AND between the two channels. As can be seen from Fig. 2.9, both the two signals of channel 0 slightly overlap with the single event of channel 1. Two  $\text{TRG\_VAL}$  signals are therefore generated. The first  $\text{TRG\_VAL}$  enables the first signal of channel 0 to be written, as well the signal from channel 1. The second  $\text{TRG\_VAL}$  enables the second signal of channel 0 to be written, but has no effect on channel 1, since it has already received a validation signal.

In this case two events are written for channel 0 and one for channel 1.



**Fig. 2.9:** Example of a "double coincidence" event

Users must take care to proper setting the coincidence parameters in order to avoid this phenomenon, or, conversely, to take it into account in their off-line analysis. For example, the double coincidence can be avoided on-line by setting the "Trigger Hold-Off" parameter equal to at least  $2 \cdot T_{ST}$ .

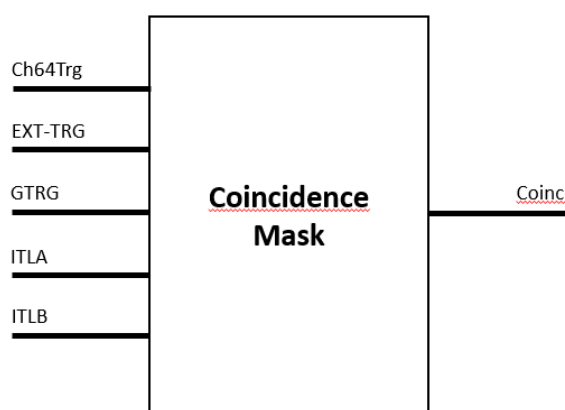
## 2.3 Channel correlations with CAEN DPP Firmware in Digitizers of the Second Generation

The x2740, x2745, x2730 and x2751 digitizer families belong to the second generation of CAEN digitizers. They feature enhanced performance and a programmable open FPGA with an embedded ARM. The number of analog input channels depends on the model: 64 for the x2740-x2745 families (which can be single-ended or differential), 32 for x2730 and 16 for x2751 (single-ended only).

WR and DPP-PSD firmwares are implemented for the x2740, x2745, x2730 and X2751 digitizers, and DPP-PHA firmware is available for the x2740, x2745 and x2730 families.

The great flexibility of these new boards, thanks to the programmable FPGA, and the high number of channels brings a new concept of coincidences, based on trigger request and validation. This modality includes also direct coincidences and trigger propagation, seen from a different point of view.

In particular, each analog input channel can send its own self-trigger request and be validated by five different signals in its own Coincidence Mask, as shown in **Fig. 2.10**, or in its Channel Trigger Mask.



**Fig. 2.10:** Scheme of Coincidence Mask logic

The validation signals are:

- **Ch64Trg**: it is the signal given by the Channel Trigger Mask, implemented for each analog input channel, and giving as output the OR of the channels selected among the 64 ones;
- **EXT-TRG**: external trigger signal as NIM/TTL TRG-IN or GPI/SIN signal or LVDS I/O signal;
- **GTRG**: Global Trigger signal;
- **ITLA**: Individual Trigger Logic A signal, output of a logic operation among channels;
- **ITLB**: Individual Trigger Logic B signal, with the same functionality of ITLA, but in a different chip.

Thus, each channel presents its individual Coincidence Mask and its individual Channel Trigger Mask.

For more detailed explanations about the mentioned signals and the implemented logic, please refer to the FELib Parameters User Manuals for the specific digitizer and firmware used **[RD11]** **[RD12]** **[RD13]** **[RD14]** **[RD15]**.

Logic operations among channels can be performed at channel level, through the Channel Trigger Mask, or at the level of the motherboard. In the latter case, the Individual Trigger Logic chips can perform the AND/OR or MAJORITY among channels, as explained in the FELib Scope Parameters User Manuals for the specific digitizer used **[RD16]** **[RD17]** **[RD18]** **[RD19]**. These logics are mostly used in case of WR firmware, as this firmware type presents a common acquisition among channels, contrarily to DPP-PHA and DPP-PSD firmwares in which channels can acquire data independently.



## 3 CoMPASS Software for coincidences with CAEN DPP firmware

This Chapter is intended to describe how to make channels correlations using the CAEN Software CoMPASS<sup>1</sup> [RD4].

CoMPASS supports coincidences with both DPP-PSD and DPP-PHA firmware types installed on CAEN digitizers of the first<sup>2</sup> and second<sup>3</sup> generation and DPP-PHA firmware types installed on CAEN MCAs.

### 3.1 Off-line Coincidences

Off-line coincidences can be performed running a customized code on a list file created by CoMPASS, in which the Trigger Time Stamp, Energy and PSD are included. To save this file type, the user must select the Acquisition mode "List only" into the Acquisition Settings section of the CoMPASS Acquisition Tab<sup>4</sup> (see Fig. 3.1).



**Note:** It is not recommended to use the "Waves" mode in CoMPASS, as it may cause the board to enter a BUSY state, leading to event loss and, consequently, invalidating coincidence.

In addition, the user must tick the "Save raw data" into the List Saving section, as to save the full list of events. The "Save unfiltered data" and the "Save filtered data" ticks should not be used, as some of the detected events would be lost, thus giving wrong coincidence results.

The user can decide to save the events from all channels in one "Single file" or to select "One file per channel". In the former case, the time sorting is not automatically done and the user must select the "Time sorting" tick into the "List saving" section.



**Note:** The saved events in one "Single file" are not automatically time sorted if saved in "List only" mode.

If the user selects the options "Save board data buffers" in a "Single file" in the *Board buffer saving* section of the Acquisition Tab, it is possible to perform time sorting of the events after the event file has been created, in case this option was not previously enabled in the Acquisition Tab. This operation can be carried out using the *Offline Run* feature in Run Data Source section.

The available output list file formats are root (.root), *Comma Separated Values* (.csv) and Binary (.bin), and are selectable in "File format" of the List Saving section of the Acquisition Tab.

### 3.2 On-line By Software Coincidences

CoMPASS allows the user to perform On-line By Software coincidences by setting properly the Acquisition and Time Selection Tabs.

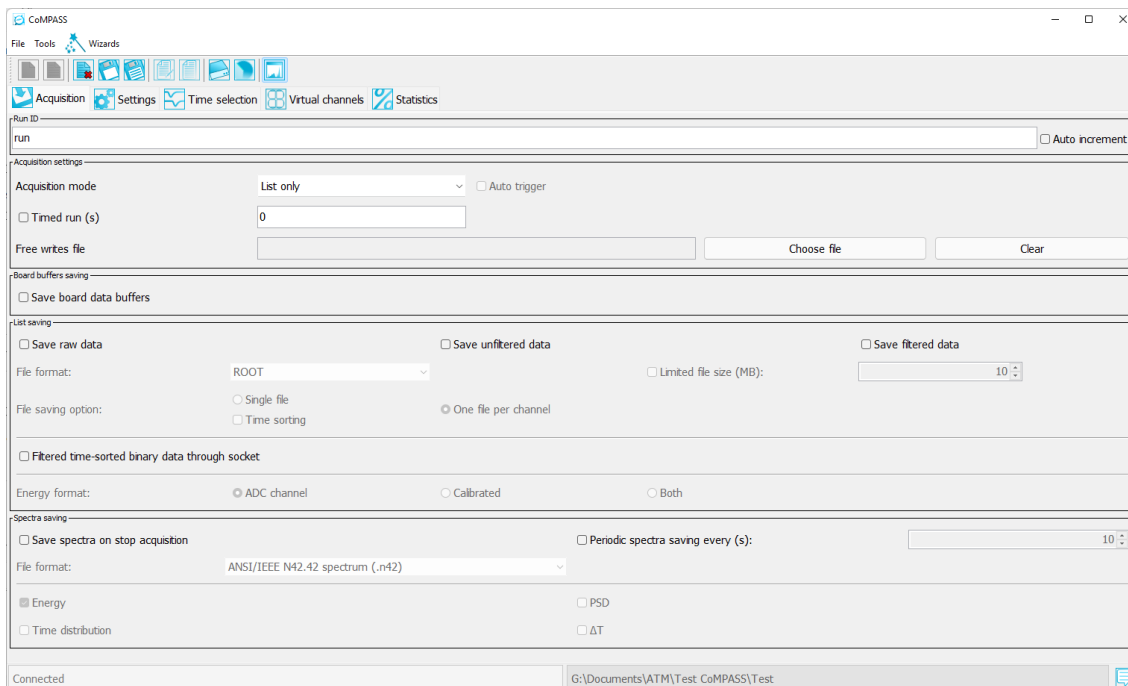
Similarly to Off-line coincidences, the CoMPASS Acquisition mode should be set to "List only". The "Save raw data" and/or the "Save unfiltered data" and/or the "Save filtered data" ticks can be indifferently chosen

<sup>1</sup>For the installation, please refer to [RD4].

<sup>2</sup>DPP-QDC for 740D digitizer series, DPP-ZLE, and DPP-DAW do not support on-line coincidences.

<sup>3</sup>x2751 supports only DPP-PSD firmware

<sup>4</sup>Please refer to coMPASS manual [RD4] for more information about saving files in list mode.



**Fig. 3.1:** CoMPASS Acquisition Tab

in the List Saving section of the Acquisition Tab.

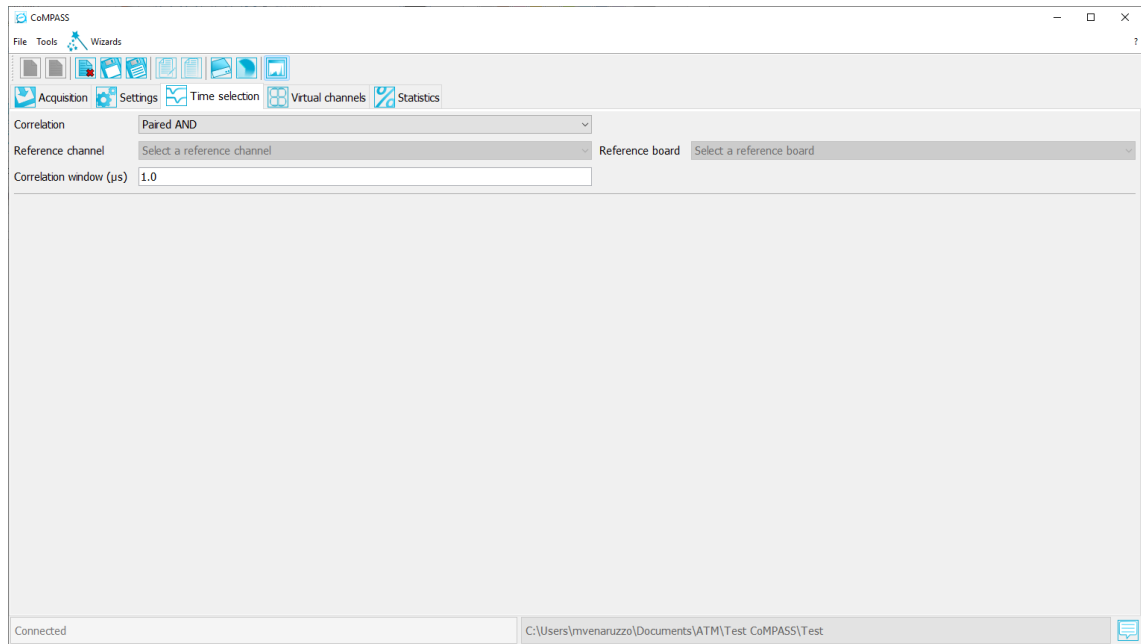
With these options the user can save:

1. **Raw data:** all the events acquired by the digitizer/MCA including pile-up and saturation events.
2. **Unfiltered data:** the events acquired by the digitizer/MCA excluding pile-up and saturation events.
3. **Filtered data:** the events that passed the filters in the "Rejection" tab of the Settings window.  
Please, refer to CoMPASS User Manual **[RD4]** for further details.

The Time Selection Tab, shown in **Fig. 3.2**, allows the user to operate on event selection based on time. The correlation is performed at the software level, i.e. without any modification of the boards firmware registers.



**Note:** Software selection implies that there is no data throughput reduction and that both time-filtered and time-unfiltered events are available for saving.



**Fig. 3.2:** CoMPASS Time Selection Tab

Using the Correlation combo box, the user can select the following options:

- **Disabled:** no correlation between the digitizer/MCA channels;
- **Paired AND:** the software correlates the digitizer/MCA channels  $n$  and  $n+1$ , i.e. Ch0 with Ch1, Ch2 with Ch3 and so on by evaluating the time difference  $T_{Chn+1} - T_{Chn}$ ;
- **Ch\_ref AND any:** it corresponds to the option Common Start in the traditional analog electronics with TDCs. One channel is meant to provide the reference time ( $T_{start}$ ) while all the others will be the evaluated with respect to it ( $T_{stop}$ ). The user must specify the reference channel in the Starting channel field.
- **Ch\_ref veto:** the reference channel acts as a veto for the other channels. Filtered events have a time difference with respect to the reference channel greater than the correlation window. The user must specify the reference channel in the Starting channel field.
- **Bd\_ref veto:** the reference board acts as a veto for the other boards. The veto signal is the logical OR of the reference board channels signal.

The width of the correlation window/veto (in  $\mu s$ ) has to be specified in the Correlation window field of the Time Selection Tab.

The time difference of correlated events is shown in the  $\Delta T$  plot of the CoMPASS Plotter Window. Please, refer to CoMPASS User Manual **[RD4]** for further details.

The user has also the possibility to access the list files generated by CoMPASS through a socket and redirect them directly to his/her own post-processing code for On-line By Software coincidences. The user just need to select the "Filtered time-sorted binary data through socket" in the List saving section of the Acquisition Tab to access this functionality. The time-sorted binary data can be:

- **filtered**, i.e. events that passed the filters in the "Rejection" tab of the Settings window. With this configuration, the user can also decide to access online (By Software/onboard) combined events through the socket;
- **unfiltered**, i.e. events acquired by the digitizer/MCA excluding pile-up and saturation events;

- **raw**, i.e. all the events acquired by the digitizer/MCA including pile-up and saturation events. In this case, the user must disable the Saturation and Pileup rejections in the Rejection Tab of CoMPASS Settings Tab (see Fig. 3.3).

Rejections	
Parameter	Board
Saturation rejection	<input checked="" type="checkbox"/>
Pileup rejection	<input checked="" type="checkbox"/>
PUR gap	1000 lsb
E low cut	0.000
E high cut	0.000
E cut enable	<input type="checkbox"/>
PSD low cut	0.000
PSD high cut	0.000
PSD cut enable	<input type="checkbox"/>
Time intervals low cut	0 ns
Time intervals high cut	0 ns
Time intervals cut enable	<input type="checkbox"/>

**Fig. 3.3:** CoMPASS Rejection Tab

In the CoMPASS installation folder under the path "C:\CoMPASS\demo" in Windows and "/CoMPASS-vX.Y.Z/demo" in Linux" a C demo code allowing the user to access the data through the socket is provided. The user can write his/her own code starting from this demo code.

The TCP/IP port used by CoMPASS is fixed and specified in the demo code. In case this port is already used by any other process, CoMPASS will automatically select another port and will specify it in the log files.

The demo code will not work in this case and will exit notifying the refused connection.

The user has then two possible solutions:

1. look into the CoMPASS log files to check the new port and replace the one written in the demo code;
2. restart the computer for an automatic re-assignment of the TCP/IP port.

Starting from CoMPASS 2.0 release, the Software allows to open already saved projects containing On-line By Software coincidences, and perform Off-line coincidences with the Software itself. Also in this case, events have to be time sorted.

### 3.3 On-line onboard Coincidences

On-line onboard coincidences acts at the digitizer/MCA firmware level and can be performed with the use of CoPASS software.



**Note:** Onboard coincidences can be set for single boards only. In case of multiple boards, each board can have its own coincidence logic.

In the Trigger/Veto/Coincidences Tab of the CoPASS Settings Tab, shown in Fig. 3.4, the user can configure the following on-board coincidence types<sup>5</sup>:

- **Paired AND:** events are acquired when they are in logic AND between couples of channels. The coincidence logic is: (Ch0 & Ch1), (Ch2 & Ch3), etc.;
- **Ch0 AND any:** All channels acquire in logic AND with Ch0, while Ch0 acquires events in logic OR with the other channels. The coincidence logic is: (Ch0 & Ch1) || (Ch0 & Ch2) || etc.;



**Note:** In case of the x725 and x730 families, due to hardware limitations, this mode requires the Ch1 to be disabled. The coincidence logic is then: (Ch0 & Ch2) || (Ch0 & Ch3) || etc.;



**Note:** In case of single digitizer/MCA board, for the correct functioning of this option, Ch\_ref\_and\_any from Time Selection tab **MUST** be enabled too, choosing Ch\_ref = Ch0 and setting the same Coincidence window. If not, some not coincidence events might occur in Ch0 and in the corresponding list file (if saved).

- **Ch0 veto:** Ch0 acquires events with its own self-trigger, while the other channels acquire events in anti-coincidence with Ch0, i.e. Ch0 acts as a veto for the other channels.
- **TRG IN level veto:** the TRG IN signal acts as a veto for all the board channels. The acquisition veto is synchronized with the whole duration of the TRG IN signal. The Coinc. Window field is not used.
- **TRG IN level gate** (725/730 series only): the TRG IN signal acts as a gate for all the board channels. The acquisition gate is synchronized with the whole duration of the TRG IN signal. The Coinc. Window field is not used.

In the field **Coinc. Window** it is possible to write the desired coincidence time window.



**Note:** The Trigger/Veto/Coincidences Tab is not available for x740D series.



**Note:** Only events passing the coincidence criteria are acquired by the board, while events not passing the criteria are discarded. This might imply a significant data throughput reduction.

Trigger/Veto/Coincidences	
Parameter	Board
Coincidence Mode	Disabled
Coincidence window	100 ns

Trigger/Veto/Coincidences	
Parameter	Board
Veto source	Disabled
Veto polarity	Active high (inhibit)
Ch trigger mask enable	<input checked="" type="checkbox"/>
Ch trigger mask low	0x00004002
Ch trigger mask high	0x00000000

**Fig. 3.4:** CoPASS Trigger/Veto/Coincidences Tab for V17xx/N67xx/DT57xx digitizers and MCAs (left), and 27xx digitizer (right).

<sup>5</sup>On-line onboard coincidences are not yet implemented for the 27xx digitizer series.

In CoMPASS, only some coincidence combinations are implemented. If the desired coincidence is not present among the ones implemented in CoMPASS, the user can upload an additional configuration file (Freewrites file) that permits to make special board configurations using firmware register writes.

The FreeWrites file can be uploaded in the Free writes file field of the Acquisition Tab, and must be written according to the following formats:

- boardID address value
- boardID address value mask
- boardID address value FirstBit LastBit

where "mask" corresponds to the bit/bits to be written, "FirstBit" and "LastBit" can be used when there are consecutive bits to be written.



**Note:** Writes must be done for individual addresses only, do not use the broadcast addresses.

The file is loaded every time the acquisition is started and the write commands are executed at the end of the digitizer/MCA programming (settings might be overwritten by the FreeWrites commands).



**Note:** The writes operation performed by the FreeWrites file are applied at the end of the configuration, so they will not be overwritten.

The Software allows also to Clear the settings uploaded with the FreeWrites file if no longer required.

In Chap. 4 and Chap. 5 are reported some examples of Freewrites files for DPP-PSD and DPP-PHA firmware types, respectively.

## 4 Examples for DPP-PSD firmware

This Chapter is intended to describe how to make coincidences with DPP-PSD firmware acting on the firmware registers through a Freewrites file. The firmware registers involved in the coincidence are described in detail, as well as how to modify the software. In this way those who write their own software, and those who use the CAEN software CoMPASS [RD4] can perform on-line onboard coincidences.

### 4.1 How to configure the registers - 720 and 751 series with DPP-PSD

To configure the coincidence settings it is required to write specific values of the firmware registers [RD20], [RD21]. In particular the syntax adopted in this section and in the following ones is the second one presented in [RD4]:

```
boardID 0x<address> 0x<value> 0x<mask>
```

Those who write their own software code, must take care of the proper register write mask, i.e. the bit mask to be written.



**Note:** All values MUST be written in *hexadecimal*.

Here the list of registers to be enabled for the coincidences. For more details about the coincidence features refer to Sec. 2.

- **Enable the propagation of the Individual Trigger (ITRG) from mother board to mezzanines** to perform the signal validation.

Register name	Address	Register bits	Options
Board Configuration	0x8000	[2]	0 ITRG disabled 1 ITRG enabled

Set bit [2] = 1 of the global channel configuration register (Board Configuration), address 0x8000, i.e. write:

```
boardID 0x8000 0x4 0x4
```

- **Enable the coincidence trigger acquisition mode.** DPP-PSD firmware supports three types of acquisition modes that can be enabled through bits[19:18] of the “DPP Algorithm Control” register. Possible choices are:

```
00    normal
01    coincidence
11    anti-coincidence
```

In the *normal acquisition mode*, as described in Sec. 2.1, each channel can either self-trigger on the input pulse or acquire from an external trigger. In the *coincidence acquisition mode*, as described in Sec. 2.2, each channel self-trigger on the input signal and acquires only when a validation signal arrives within a certain time window. The *anti-coincidence acquisition mode* works in the same way as the coincidence mode, but the internal logic is the opposite, i.e. the validation occurs when no signal arrives in the acceptance time window.

Set bits [19:18] = 01 of the channel control register (DPP Algorithm Control). Coincidences must be enabled for each channel involved; for example in case of channel 0 and channel 1 write:

Register name	Address	Register bits	Options
DPP Algorithm Control	0x1n80 (channel n)	[19:18]	00 coincidence disabled 01 coincidence enabled 11 anti-coincidence enabled

```
boardID 0x1080 0x40000 0xC0000
```

```
boardID 0x1180 0x40000 0xC0000
```

- **Set the coincidence windows** (i.e. the width of the shaped trigger) for the trigger request signal ( $T_{ST}$ ).

Register name	Address	Register bits	Options
Shaped Trigger Width	0x1n70 (channel n)	[9:0]	n. clocks (hex)

Set the same value for each channel involved in the coincidence. The value is expressed in trigger clock cycles (8 ns for 751 and 720 series), i.e. you have to write the desired value divided by 8 (in hexadecimal). For example, if you want to have a window of 40 ns in case of channel 0 and channel 1 coincidence, you may write:

```
boardID 0x1070 0x5 0x3FF
```

```
boardID 0x1170 0x5 0x3FF
```

- **Set the trigger latency**  $T_{LAT}$ . This value must be equal to 9 clock cycles (i.e. 72 ns).

```
boardID 0x106c 0x9 0x3FF
```

```
boardID 0x116c 0x9 0x3FF
```

- **Write the trigger validation logic**, which corresponds to the logic operation that enables the coincidence. The trigger validation can be generated either by the Individual Trigger Logic, or by the Global Trigger Logic (see Sec. 2). Register Trigger Validation Mask 0x8180+(4n), where n is the channel number, manages the Individual Trigger Logic<sup>1</sup>.

It is possible to perform the logic operation between the channels TRG-REQ by modifying bits [9:8], according to the options: OR = 00, AND = 01, and MAJORITY = 10. Bits [12:10] allow to set the majority level  $m$ , where for a level  $m$  the majority fires when at least  $m+1$  trigger requests are high. It is possible also to include the LVDS I/O Global Trigger (bit [28] VME only), LVDS I/O Individual Trigger (bit [29] VME only), the External Trigger (bit [30]), and the Software Trigger (bit [31]) in logic OR with the individual TRG-REQ.

For example, to configure the AND between channel 0 and channel 1, write:

```
boardID 0x8180 0x103 0xFFFFFFFF
```

```
boardID 0x8184 0x103 0xFFFFFFFF
```

<sup>1</sup>Register address corresponds to 0x8180 for channel 0, 0x8184 for channel 1, 0x8188 for channel 2, 0x818C for channel 3, 0x8190 for channel 4, 0x8194 for channel 5, 0x8198 for channel 6, and 0x819C for channel 7



Register name	Address	Register bits	Options
Trigger Validation Mask	0x8180+(4n) (channel n)	[0]	0 TRG_REQ[0] not propagated 1 TRG_REQ[0] propagated
		[1]	0 TRG_REQ[1] not propagated 1 TRG_REQ[1] propagated
		[2]	0 TRG_REQ[2] not propagated 1 TRG_REQ[2] propagated
		[3]	0 TRG_REQ[3] not propagated 1 TRG_REQ[3] propagated
		[4]	0 TRG_REQ[4] not propagated 1 TRG_REQ[4] propagated
		[5]	0 TRG_REQ[5] not propagated 1 TRG_REQ[5] propagated
		[6]	0 TRG_REQ[6] not propagated 1 TRG_REQ[6] propagated
		[7]	0 TRG_REQ[7] not propagated 1 TRG_REQ[7] propagated
		Operation Mask [9:8]	00 OR 01 AND 10 MAJORITY
		Majority Level [12:10]	value for majority
		[27:13]	0 (Reserved)
		LVDS I/O Global Trigger [28]	0 LVDS GTRG not propagated 1 LVDS GTRG propagated
		LVDS I/O Individual Trigger [29]	0 LVDS ITRG not propagated 1 LVDS ITRG propagated
		External Trigger [30]	0 EXT TRG not propagated 1 EXT TRG propagated
		Software Trigger [31]	0 SOFT TRG not propagated 1 SOFT TRG propagated

## 4.2 Examples for 720-751 series

### 4.2.1 Coincidence between channel 0 and channel 1 (ch0 & ch1)

Channel 0 and channel 1 acquire data within 40 ns of coincidence window:

```
# Syntax:
# boardID 0x<address> 0x<value> 0x<mask>

boardID 0x8000 0x4 0x4
boardID 0x1080 0x40000 0xC0000
boardID 0x1180 0x40000 0xC0000
boardID 0x1070 0x5 0x3FF
boardID 0x1170 0x5 0x3FF
boardID 0x106C 0x9 0x3FF
boardID 0x116C 0x9 0x3FF
boardID 0x8180 0x103 0xFFFFFFFF
boardID 0x8184 0x103 0xFFFFFFFF
```

### 4.2.2 Anti-coincidence between two channels (ch0 $\oplus$ ch1)

Channel 0 and channel 1 acquire data in anti-coincidence, within 80 ns of anti-coincidence window:

```
# Syntax:
# boardID 0x<address> 0x<value> 0x<mask>

boardID 0x8000 0x4 0x4
boardID 0x1080 0xC0000 0xC0000
boardID 0x1180 0xC0000 0xC0000
boardID 0x1070 0xA 0x3FF
boardID 0x1170 0xA 0x3FF
boardID 0x106C 0x9 0x3FF
boardID 0x116C 0x9 0x3FF
boardID 0x8180 0x103 0xFFFFFFFF
boardID 0x8184 0x103 0xFFFFFFFF
```

### 4.2.3 Coincidence among four channels (ch0 & ch1 & ch2 & ch3)

Channel 0 to channel 3 acquire data within 40 ns of coincidence window:

```
# Syntax:
# boardID 0x<address> 0x<value> 0x<mask>

boardID 0x8000 0x4 0x4
boardID 0x1080 0x40000 0xC0000
boardID 0x1180 0x40000 0xC0000
boardID 0x1280 0x40000 0xC0000
boardID 0x1380 0x40000 0xC0000
boardID 0x1070 0x5 0x3FF
boardID 0x1170 0x5 0x3FF
boardID 0x1270 0x5 0x3FF
boardID 0x1370 0x5 0x3FF
```

```
boardID 0x106C 0x9 0x3FF
boardID 0x116C 0x9 0x3FF
boardID 0x126C 0x9 0x3FF
boardID 0x136C 0x9 0x3FF
boardID 0x8180 0x10F 0xFFFFFFFF
boardID 0x8184 0x10F 0xFFFFFFFF
boardID 0x8188 0x10F 0xFFFFFFFF
boardID 0x818C 0x10F 0xFFFFFFFF
```

#### 4.2.4 Coincidence between channel 0 and channel 1, coincidence between channel 2 and channel 3 (ch0 & ch1), (ch2 & ch3)

Sets the coincidence between couples of channels, channel 0 and channel 1, channel 2 and channel 3 within 40 ns of coincidence window:

```
# Syntax:
# boardID 0x<address> 0x<value> 0x<mask>
```

```
boardID 0x8000 0x4 0x4
boardID 0x1080 0x40000 0xC0000
boardID 0x1180 0x40000 0xC0000
boardID 0x1280 0x40000 0xC0000
boardID 0x1380 0x40000 0xC0000
boardID 0x1070 0x5 0x3FF
boardID 0x1170 0x5 0x3FF
boardID 0x1270 0x5 0x3FF
boardID 0x1370 0x5 0x3FF
boardID 0x106C 0x9 0x3FF
boardID 0x116C 0x9 0x3FF
boardID 0x126C 0x9 0x3FF
boardID 0x136C 0x9 0x3FF
boardID 0x8180 0x103 0xFFFFFFFF
boardID 0x8184 0x103 0xFFFFFFFF
boardID 0x8188 0x10C 0xFFFFFFFF
boardID 0x818C 0x10C 0xFFFFFFFF
```

## 4.2.5 Coincidence among eight channels (ch0 & ch1 & ch2 & ch3 & ch4 & ch5 & ch6 & ch7) - VME only

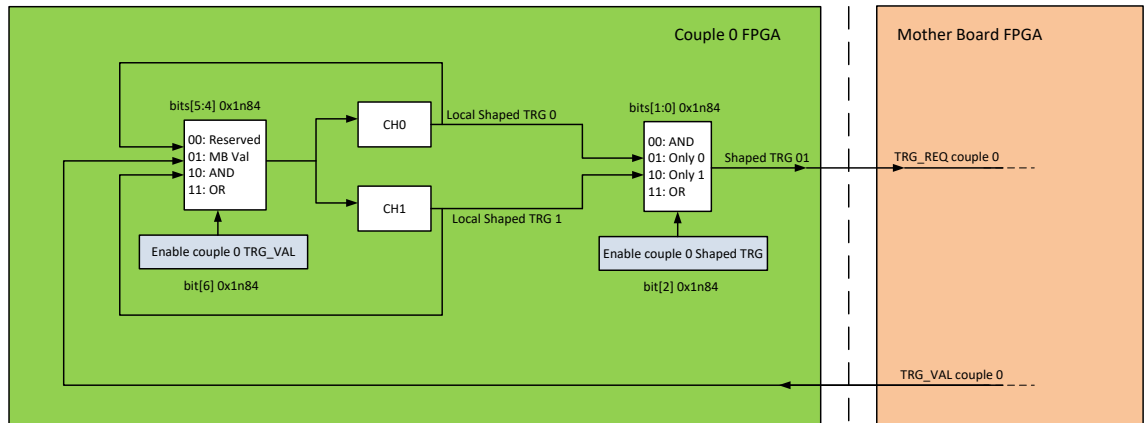
Channel 0 to channel 7 (VME only) acquire data within 40 ns of coincidence window:

```
# Syntax:
# boardID 0x<address> 0x<value> 0x<mask>
```

```
boardID 0x8000 0x4 0x4
boardID 0x1080 0x40000 0xC0000
boardID 0x1180 0x40000 0xC0000
boardID 0x1280 0x40000 0xC0000
boardID 0x1380 0x40000 0xC0000
boardID 0x1480 0x40000 0xC0000
boardID 0x1580 0x40000 0xC0000
boardID 0x1680 0x40000 0xC0000
boardID 0x1780 0x40000 0xC0000
boardID 0x1070 0x5 0x3FF
boardID 0x1170 0x5 0x3FF
boardID 0x1270 0x5 0x3FF
boardID 0x1370 0x5 0x3FF
boardID 0x1470 0x5 0x3FF
boardID 0x1570 0x5 0x3FF
boardID 0x1670 0x5 0x3FF
boardID 0x1770 0x5 0x3FF
boardID 0x106C 0x9 0x3FF
boardID 0x116C 0x9 0x3FF
boardID 0x126C 0x9 0x3FF
boardID 0x136C 0x9 0x3FF
boardID 0x146C 0x9 0x3FF
boardID 0x156C 0x9 0x3FF
boardID 0x166C 0x9 0x3FF
boardID 0x176C 0x9 0x3FF
boardID 0x8180 0x1FF 0xFFFFFFFF
boardID 0x8184 0x1FF 0xFFFFFFFF
boardID 0x8188 0x1FF 0xFFFFFFFF
boardID 0x818C 0x1FF 0xFFFFFFFF
boardID 0x8190 0x1FF 0xFFFFFFFF
boardID 0x8194 0x1FF 0xFFFFFFFF
boardID 0x8198 0x1FF 0xFFFFFFFF
boardID 0x819C 0x1FF 0xFFFFFFFF
```

## 4.3 How to configure the registers - 725 and 730 series with DPP-PSD

As mentioned in Sec. 2.2 couples of channels in 725 and 730 series share the same TRG\_REQ. The local Shaped Trigger and local Trigger Validation options can be configured through register "DPP Algorithm Control 2", at address 0x1n84 [RD9], and they are summarized in Fig. 4.1. The n index identifies the n-th channel. The register value must be equal for channels of the same couple. Writing the n channel value, it will write the same value in the n+1 channel (channels of the same couple). Refer to [RD9] for more details.



**Fig. 4.1:** Local Trigger Management inside couple 0 of 725-730 digitizer series. Couple 0 is made of channel 0 and channel 1. The same applies for the other couples of the 725-730 digitizers.

The TRG\_REQ of the couple can be the AND, OR of the two channels, or one of the two channels. The TRG\_VAL can be generated from mother board, from the other channel of the couple, or it can be the logic AND/OR of the two shaped TRG.

From revision 4.17\_136.14 bits[5:4] are modified as shown in the table, and in case of coincidence inside the couple (ch0 & ch1) it is recommended to use option AND (10). The register write is anyhow left unchanged from previous firmware releases.



**Note:** In the Trigger Validation Mask register, 0x8180(+4n), n is now the couple index, since the validation mask from mother board is referred to the couple, rather than to the single channel.



**Note:** The coincidence window is expressed in steps of 16 ns for 725 series and 8 ns in case of 730.

To set the coincidence between channel 0 and channel 1 inside the couple (no mother board processing) write:

```
boardID 0x1084 0x60 0xFF
```

See more examples in the next Sections.

Register name	Address	Register bits	Options
DPP Algorithm Control 2	0x1n84	[1:0]	00: TRG_REQ is the AND of the channels 01: TRG_REQ is the even channel of the couple 10: TRG_REQ is the odd channel of the couple 11: TRG_REQ is the OR of the channels
		[2]	Enable TRG_REQ of the couple
		[3]	Reserved
		[5:4]	00: Reserved; 01: TRG_VAL0 = TRG_VAL1 = signal from mother-board mask; 10: AND (TRG_VAL0 = TRG_VAL1 = TRG_REQ0 AND TRG_REQ1); 11: OR (TRG_VAL0 = TRG_VAL1 = TRG_REQ0 OR TRG_REQ1).
		[6]	Enable TRG_VAL of the couple
		[26:25]	Additional Local Trigger Validation options: 00 = options from bits[5:4]; 01 = validation from paired channel AND mother board; 10 = validation from paired channel OR mother board; 11 = reserved. Note: bit[6] of this register must be enabled. Note: bit[5:4] of this register must be set to 00. Note: when using option 01 and 10 there are about 30 ns of transition window between coincidence and not coincidence level. This should be taken into account in the offline data processing by the user.

## 4.4 Examples for 725-730 series

### 4.4.1 Coincidence between channel 0 and channel 1 (ch0 & ch1)

Channel 0 and channel 1 acquire data within 80 ns of coincidence window:

```
# Syntax:
# boardID 0x<address> 0x<value> 0x<mask>

boardID 0x1080 0x40000 0xC0000
boardID 0x1180 0x40000 0xC0000
boardID 0x1070 0xA 0x3FF
boardID 0x1170 0xA 0x3FF
boardID 0x106C 0x2 0x3FF
boardID 0x116C 0x2 0x3FF
boardID 0x1084 0x60 0xFF
```



**Note:** The register 0x8000 should not be written in case of a coincidence inside a couple of channels, as it is performed in the channel FPGA only, with no mother board processing.



**Note:** In case of coincidence inside a couple, set the latency equal to 0x2 (the latency is of two clock cycles, as the coincidence is performed in the channel FPGA only, with no mother board processing).



**Note:** In case of 725 write: boardID 0x1n70 0x5 0x3FF and boardID 0x1170 0x5 0x3FF to get 80 ns of coincidence window.

### 4.4.2 Coincidence inside the couples (ch0 & ch1), (ch2 & ch3), (ch4 & ch5), (ch6 & ch7)

Couples of channels (up to channel 7 for DT/NIM form factors) acquire data within 80 ns of coincidence window:

```
# Syntax:
# boardID 0x<address> 0x<value> 0x<mask>

boardID 0x1080 0x40000 0xC0000
boardID 0x1180 0x40000 0xC0000
boardID 0x1280 0x40000 0xC0000
boardID 0x1380 0x40000 0xC0000
boardID 0x1480 0x40000 0xC0000
boardID 0x1580 0x40000 0xC0000
boardID 0x1680 0x40000 0xC0000
boardID 0x1780 0x40000 0xC0000
boardID 0x1070 0xA 0x3FF
boardID 0x1170 0xA 0x3FF
boardID 0x1270 0xA 0x3FF
boardID 0x1370 0xA 0x3FF
boardID 0x1470 0xA 0x3FF
boardID 0x1570 0xA 0x3FF
boardID 0x1670 0xA 0x3FF
```

```
boardID 0x1770 0xA 0x3FF
boardID 0x106C 0x2 0x3FF
boardID 0x116C 0x2 0x3FF
boardID 0x126C 0x2 0x3FF
boardID 0x136C 0x2 0x3FF
boardID 0x146C 0x2 0x3FF
boardID 0x156C 0x2 0x3FF
boardID 0x166C 0x2 0x3FF
boardID 0x176C 0x2 0x3FF
boardID 0x1084 0x60 0xFF
boardID 0x1284 0x60 0xFF
boardID 0x1484 0x60 0xFF
boardID 0x1684 0x60 0xFF
```



**Note:** The register 0x8000 should not be written in case of a coincidence inside a couple of channels, as it is performed in the channel FPGA only, with no mother board processing.



**Note:** In case of coincidence inside a couple, set the latency equal to 0x2 (the latency is of two clock cycles, as the coincidence is performed in the channel FPGA only, with no mother board processing).



**Note:** In case of 725 write: boardID 0x1n70 0x5 0x3FF, etc. to get 80 ns of coincidence window.

### 4.4.3 Anti-coincidence between PAIRED channels: (ch0 $\oplus$ ch1), (ch2 $\oplus$ ch3), (ch4 $\oplus$ ch5), (ch6 $\oplus$ ch7)

Each couple of channels (up to channel 7 for DT/NIM fomr factors) acquire data in anti-coincidence within a window of 800 ns:

```
# Syntax:
# boardID 0x<address> 0x<value> 0x<mask>

boardID 0x1080 0xC0000 0xC0000
boardID 0x1180 0xC0000 0xC0000
boardID 0x1280 0xC0000 0xC0000
boardID 0x1380 0xC0000 0xC0000
boardID 0x1480 0xC0000 0xC0000
boardID 0x1580 0xC0000 0xC0000
boardID 0x1680 0xC0000 0xC0000
boardID 0x1780 0xC0000 0xC0000
boardID 0x1070 0x64 0xFF
boardID 0x1170 0x64 0xFF
boardID 0x1270 0x64 0xFF
boardID 0x1370 0x64 0xFF
boardID 0x1470 0x64 0xFF
boardID 0x1570 0x64 0xFF
boardID 0x1670 0x64 0xFF
boardID 0x1770 0x64 0xFF
boardID 0x106C 0x2 0xFF
boardID 0x116C 0x2 0xFF
boardID 0x126C 0x2 0xFF
boardID 0x136C 0x2 0xFF
```



```
boardID 0x146C 0x2 0xFF
boardID 0x156C 0x2 0xFF
boardID 0x166C 0x2 0xFF
boardID 0x176C 0x2 0xFF
boardID 0x1084 0x60 0xFF
boardID 0x1284 0x60 0xFF
boardID 0x1484 0x60 0xFF
boardID 0x1684 0x60 0xFF
```



**Note:** The register 0x8000 should not be written in case of a coincidence inside a couple of channels, as it is performed in the channel FPGA only, with no mother board processing.



**Note:** In case of coincidence inside a couple, set the latency equal to 0x2 (the latency is of two clock cycles, as the coincidence is performed in the channel FPGA only, with no mother board processing).



**Note:** In case of 725 write: boardID 0x1n70 0x32 0x3FF for n=0,...,7 to get 800 ns of coincidence window.

#### 4.4.4 Coincidence inside the couples (ch0 & ch1), (ch2 & ch3), (ch4 & ch5), (ch6 & ch7), (ch8 & ch9), (ch10 & ch11), (ch12 & ch13), (ch14 & ch15) - VME only

Couples of channels (up to channel 15 for VME form factor) acquire data within 80 ns of coincidence window:

```
# Syntax:
# boardID 0x<address> 0x<value> 0x<mask>

boardID 0x1080 0x40000 0xC0000
boardID 0x1180 0x40000 0xC0000
boardID 0x1280 0x40000 0xC0000
boardID 0x1380 0x40000 0xC0000
boardID 0x1480 0x40000 0xC0000
boardID 0x1580 0x40000 0xC0000
boardID 0x1680 0x40000 0xC0000
boardID 0x1780 0x40000 0xC0000
boardID 0x1880 0x40000 0xC0000
boardID 0x1980 0x40000 0xC0000
boardID 0x1A80 0x40000 0xC0000
boardID 0x1B80 0x40000 0xC0000
boardID 0x1C80 0x40000 0xC0000
boardID 0x1D80 0x40000 0xC0000
boardID 0x1E80 0x40000 0xC0000
boardID 0x1F80 0x40000 0xC0000
boardID 0x1070 0xA 0x3FF
boardID 0x1170 0xA 0x3FF
boardID 0x1270 0xA 0x3FF
boardID 0x1370 0xA 0x3FF
boardID 0x1470 0xA 0x3FF
boardID 0x1570 0xA 0x3FF
boardID 0x1670 0xA 0x3FF
```

```

boardID 0x1770 0xA 0x3FF
boardID 0x1870 0xA 0x3FF
boardID 0x1970 0xA 0x3FF
boardID 0x1A70 0xA 0x3FF
boardID 0x1B70 0xA 0x3FF
boardID 0x1C70 0xA 0x3FF
boardID 0x1D70 0xA 0x3FF
boardID 0x1E70 0xA 0x3FF
boardID 0x1F70 0xA 0x3FF
boardID 0x106C 0x2 0x3FF
boardID 0x116C 0x2 0x3FF
boardID 0x126C 0x2 0x3FF
boardID 0x136C 0x2 0x3FF
boardID 0x146C 0x2 0x3FF
boardID 0x156C 0x2 0x3FF
boardID 0x166C 0x2 0x3FF
boardID 0x176C 0x2 0x3FF
boardID 0x186C 0x2 0x3FF
boardID 0x196C 0x2 0x3FF
boardID 0x1A6C 0x2 0x3FF
boardID 0x1B6C 0x2 0x3FF
boardID 0x1C6C 0x2 0x3FF
boardID 0x1D6C 0x2 0x3FF
boardID 0x1E6C 0x2 0x3FF
boardID 0x1F6C 0x2 0x3FF
boardID 0x1084 0x60 0xFF
boardID 0x1284 0x60 0xFF
boardID 0x1484 0x60 0xFF
boardID 0x1684 0x60 0xFF
boardID 0x1884 0x60 0xFF
boardID 0x1A84 0x60 0xFF
boardID 0x1C84 0x60 0xFF
boardID 0x1D84 0x60 0xFF

```



**Note:** The register 0x8000 should not be written in case of a coincidence inside a couple of channels, as it is performed in the channel FPGA only, with no mother board processing.



**Note:** In case of coincidence inside a couple, set the latency equal to 0x2 (the latency is of two clock cycles, as the coincidence is performed in the channel FPGA only, with no mother board processing).



**Note:** In case of 725 write: boardID 0x1n70 0x5 0x3FF, etc. to get 80 ns of coincidence window.

#### 4.4.5 Coincidence among four channels (four different couples, ch0 & ch2 & ch4 & ch6)

Even channels of the first four couples acquire data within 80 ns of coincidence window. The four couples provide as TRG\_REQ only the even channels and receive the TRG\_VAL from mother board. The mother board FPGA performs the AND between the four couples.



**Note:** Since the two channels of the couple share the same memory it is recommended to use only one channel per couple, if available. If it is not possible to use one channel per couple, see the example in the next section.



**Note:** The odd channels of the couples do not participate to the coincidence logic.

```
# Syntax:
# boardID 0x<address> 0x<value> 0x<mask>

boardID 0x8000 0x4 0x4
boardID 0x1080 0x40000 0xC0000
boardID 0x1280 0x40000 0xC0000
boardID 0x1480 0x40000 0xC0000
boardID 0x1680 0x40000 0xC0000
boardID 0x1070 0xA 0x3FF
boardID 0x1270 0xA 0x3FF
boardID 0x1470 0xA 0x3FF
boardID 0x1670 0xA 0x3FF
boardID 0x106C 0x9 0x3FF
boardID 0x126C 0x9 0x3FF
boardID 0x146C 0x9 0x3FF
boardID 0x166C 0x9 0x3FF
boardID 0x1084 0x55 0xFF
boardID 0x1284 0x55 0xFF
boardID 0x1484 0x55 0xFF
boardID 0x1684 0x55 0xFF
boardID 0x8180 0x10F 0xFFFFFFFF
boardID 0x8184 0x10F 0xFFFFFFFF
boardID 0x8188 0x10F 0xFFFFFFFF
boardID 0x818C 0x10F 0xFFFFFFFF
```



**Note:** In case of 725 write: boardID 0x1n70 0x5 0x3FF, etc. to get 80 ns of coincidence window.

#### 4.4.6 Coincidence among four channels (two different couples, ch0 & ch1 & ch2 & ch3)

Channel 0 to channel 3 acquire data within 80 ns of coincidence window. The two couples provide as TRG\_REQ the AND of the two channels and receive the TRG\_VAL from mother board. The mother board FPGA performs the AND between couple 0 and couple 1.



**Note:** Since the two channels of the couple share the same memory it is usually suggested to perform the coincidence as in the previous example - if channels are not otherwise occupied.

```
# Syntax:
# boardID 0x<address> 0x<value> 0x<mask>

boardID 0x8000 0x4 0x4
boardID 0x1080 0x40000 0xC0000
boardID 0x1180 0x40000 0xC0000
boardID 0x1280 0x40000 0xC0000
boardID 0x1380 0x40000 0xC0000
boardID 0x1070 0xA 0x3FF
boardID 0x1170 0xA 0x3FF
boardID 0x1270 0xA 0x3FF
boardID 0x1370 0xA 0x3FF
boardID 0x106C 0x9 0x3FF
boardID 0x116C 0x9 0x3FF
boardID 0x126C 0x9 0x3FF
boardID 0x136C 0x9 0x3FF
boardID 0x1084 0x54 0xFF
boardID 0x1284 0x54 0xFF
boardID 0x8180 0x103 0xFFFFFFFF
boardID 0x8184 0x103 0xFFFFFFFF
```



**Note:** In case of 725 write: boardID 0x1n70 0x5 0x3FF, etc. to get 80 ns of coincidence window.

#### 4.4.7 Coincidence among four channels plus four channels (ch0 & ch1 & ch4 & ch5, ch2 & ch3 & ch6 & ch7)

Channel 0 to channel 7 acquire data within 80 ns of coincidence window. All the couples provides as TRG\_REQ the AND of the two channels of the couple and receive the TRG\_VAL from mother board. The mother board FPGA performs the AND between couple 0 and couple 2 and between couple 1 and couple 3.



**Note:** Since the two channels of the couple share the same memory it is usually suggested to perform the coincidence as in the previous example - if channels are not otherwise occupied.

```
# Syntax:
# boardID 0x<address> 0x<value> 0x<mask>

boardID 0x8000 0x4 0x4
boardID 0x1080 0x40000 0xC0000
boardID 0x1180 0x40000 0xC0000
boardID 0x1280 0x40000 0xC0000
boardID 0x1380 0x40000 0xC0000
boardID 0x1480 0x40000 0xC0000
boardID 0x1580 0x40000 0xC0000
boardID 0x1680 0x40000 0xC0000
boardID 0x1780 0x40000 0xC0000
boardID 0x1070 0xA 0x3FF
boardID 0x1170 0xA 0x3FF
boardID 0x1270 0xA 0x3FF
boardID 0x1370 0xA 0x3FF
boardID 0x1470 0xA 0x3FF
boardID 0x1570 0xA 0x3FF
boardID 0x1670 0xA 0x3FF
boardID 0x1770 0xA 0x3FF
boardID 0x106C 0x9 0x3FF
boardID 0x116C 0x9 0x3FF
boardID 0x126C 0x9 0x3FF
boardID 0x136C 0x9 0x3FF
boardID 0x146C 0x9 0x3FF
boardID 0x156C 0x9 0x3FF
boardID 0x166C 0x9 0x3FF
boardID 0x176C 0x9 0x3FF
boardID 0x1084 0x54 0xFF
boardID 0x1284 0x54 0xFF
boardID 0x1484 0x54 0xFF
boardID 0x1684 0x54 0xFF
boardID 0x8180 0x105 0xFFFFFFFF
boardID 0x8184 0x10A 0xFFFFFFFF
boardID 0x8188 0x105 0xFFFFFFFF
boardID 0x818C 0x10A 0xFFFFFFFF
```



**Note:** In case of 725 write: boardID 0x1n70 0x5 0x3FF, etc. to get 80 ns of coincidence window.

#### 4.4.8 Coincidence between couples of channels in OR:

**(ch0 || ch1) & (ch2||ch3) & (ch4||ch5) & (ch6||ch7) & (ch8||ch9)  
& (ch10||ch11) & (ch12||ch13) & (ch14||ch15) - VME only**

Channel 0 to 16 acquire data within 80 ns of coincidence window. The couples provide as TRG\_REQ the logic OR of the channels and receive the TRG\_VAL from mother board. The mother board FPGA performs the AND between all the couples.

# Syntax:

# boardID 0x<address> 0x<value> 0x<mask>

```
boardID 0x8000 0x4 0x4
boardID 0x1080 0x40000 0xC0000
boardID 0x1180 0x40000 0xC0000
boardID 0x1280 0x40000 0xC0000
boardID 0x1380 0x40000 0xC0000
boardID 0x1480 0x40000 0xC0000
boardID 0x1580 0x40000 0xC0000
boardID 0x1680 0x40000 0xC0000
boardID 0x1780 0x40000 0xC0000
boardID 0x1880 0x40000 0xC0000
boardID 0x1980 0x40000 0xC0000
boardID 0x1A80 0x40000 0xC0000
boardID 0x1B80 0x40000 0xC0000
boardID 0x1C80 0x40000 0xC0000
boardID 0x1D80 0x40000 0xC0000
boardID 0x1E80 0x40000 0xC0000
boardID 0x1F80 0x40000 0xC0000
boardID 0x1070 0xA 0x3FF
boardID 0x1170 0xA 0x3FF
boardID 0x1270 0xA 0x3FF
boardID 0x1370 0xA 0x3FF
boardID 0x1470 0xA 0x3FF
boardID 0x1570 0xA 0x3FF
boardID 0x1670 0xA 0x3FF
boardID 0x1770 0xA 0x3FF
boardID 0x1870 0xA 0x3FF
boardID 0x1970 0xA 0x3FF
boardID 0x1A70 0xA 0x3FF
boardID 0x1B70 0xA 0x3FF
boardID 0x1C70 0xA 0x3FF
boardID 0x1D70 0xA 0x3FF
boardID 0x1E70 0xA 0x3FF
boardID 0x1F70 0xA 0x3FF
boardID 0x106C 0x9 0x3FF
boardID 0x116C 0x9 0x3FF
boardID 0x126C 0x9 0x3FF
boardID 0x136C 0x9 0x3FF
boardID 0x146C 0x9 0x3FF
boardID 0x156C 0x9 0x3FF
boardID 0x166C 0x9 0x3FF
boardID 0x176C 0x9 0x3FF
boardID 0x186C 0x9 0x3FF
boardID 0x196C 0x9 0x3FF
boardID 0x1A6C 0x9 0x3FF
```

```
boardID 0x1B6C 0x9 0x3FF
boardID 0x1C6C 0x9 0x3FF
boardID 0x1D6C 0x9 0x3FF
boardID 0x1E6C 0x9 0x3FF
boardID 0x1F6C 0x9 0x3FF
boardID 0x1084 0x57 0xFF
boardID 0x1284 0x57 0xFF
boardID 0x1484 0x57 0xFF
boardID 0x1684 0x57 0xFF
boardID 0x1884 0x57 0xFF
boardID 0x1A84 0x57 0xFF
boardID 0x1C84 0x57 0xFF
boardID 0x1E84 0x57 0xFF
boardID 0x8180 0x1FF 0xFFFFFFFF
boardID 0x8184 0x1FF 0xFFFFFFFF
boardID 0x8188 0x1FF 0xFFFFFFFF
boardID 0x818C 0x1FF 0xFFFFFFFF
boardID 0x8190 0x1FF 0xFFFFFFFF
boardID 0x8194 0x1FF 0xFFFFFFFF
boardID 0x8198 0x1FF 0xFFFFFFFF
boardID 0x819C 0x1FF 0xFFFFFFFF
```



**Note:** In case of 725 write: boardID 0x1n70 0x5 0x3FF, etc. to get 80 ns of coincidence window.

#### 4.4.9 Coincidence among eight channels (eight different couples, ch0 & ch2 & ch4 & ch6 & ch8 & ch10 & ch12 & ch14) - VME only

Even channels of the eight couples (VME only) acquire data within 80 ns of coincidence window. The couples provide as TRG\_REQ only the even channel and receive the TRG\_VAL from mother board. The mother board FPGA performs the AND between the eight couples.



**Note:** Since the two channels of the couple share the same memory it is recommended to use only one channel per couple, if available. If it is not possible to use one channel per couple, see the example in the next section.



**Note:** The odd channels of the couples do not participate to the coincidence logic.

```
# Syntax:
# boardID 0x<address> 0x<value> 0x<mask>
```

```
boardID 0x8000 0x4 0x4
boardID 0x1080 0x40000 0xC0000
boardID 0x1280 0x40000 0xC0000
boardID 0x1480 0x40000 0xC0000
boardID 0x1680 0x40000 0xC0000
boardID 0x1880 0x40000 0xC0000
boardID 0x1A80 0x40000 0xC0000
boardID 0x1C80 0x40000 0xC0000
boardID 0x1E80 0x40000 0xC0000
boardID 0x1070 0xA 0x3FF
boardID 0x1270 0xA 0x3FF
boardID 0x1470 0xA 0x3FF
boardID 0x1670 0xA 0x3FF
boardID 0x1870 0xA 0x3FF
boardID 0x1A70 0xA 0x3FF
boardID 0x1C70 0xA 0x3FF
boardID 0x1E70 0xA 0x3FF
boardID 0x106C 0x9 0x3FF
boardID 0x126C 0x9 0x3FF
boardID 0x146C 0x9 0x3FF
boardID 0x166C 0x9 0x3FF
boardID 0x186C 0x9 0x3FF
boardID 0x1A6C 0x9 0x3FF
boardID 0x1C6C 0x9 0x3FF
boardID 0x1E6C 0x9 0x3FF
boardID 0x1084 0x55 0xFF
boardID 0x1284 0x55 0xFF
boardID 0x1484 0x55 0xFF
boardID 0x1684 0x55 0xFF
boardID 0x1884 0x55 0xFF
boardID 0x1A84 0x55 0xFF
boardID 0x1C84 0x55 0xFF
boardID 0x1E84 0x55 0xFF
boardID 0x8180 0x1FF 0xFFFFFFFF
boardID 0x8184 0x1FF 0xFFFFFFFF
boardID 0x8188 0x1FF 0xFFFFFFFF
boardID 0x818C 0x1FF 0xFFFFFFFF
boardID 0x8190 0x1FF 0xFFFFFFFF
boardID 0x8194 0x1FF 0xFFFFFFFF
```



```
boardID 0x8198 0x1FF 0xFFFFFFFF
boardID 0x819C 0x1FF 0xFFFFFFFF
```



**Note:** In case of 725 write: boardID 0x1n70 0x5 0x3FF, etc. to get 80 ns of coincidence window.

#### 4.4.10 Coincidence among eight channels (four different couples, ch0 & ch1 & ch2 & ch3 & ch4 & ch5 & ch6 & ch7)

Channel 0 to channel 7 acquire data within 80 ns of coincidence window. The four couples provide as TRG\_REQ the AND of the two channels and receive the TRG\_VAL from mother board. The mother board FPGA performs the AND between the four couples.



**Note:** Since the two channels of the couple share the same memory it is usually suggested to perform the coincidence as in the previous example (VME only) - if channels are not otherwise occupied.

```
# Syntax:
# boardID 0x<address> 0x<value> 0x<mask>

boardID 0x8000 0x4 0x4
boardID 0x1080 0x40000 0xC0000
boardID 0x1180 0x40000 0xC0000
boardID 0x1280 0x40000 0xC0000
boardID 0x1380 0x40000 0xC0000
boardID 0x1480 0x40000 0xC0000
boardID 0x1580 0x40000 0xC0000
boardID 0x1680 0x40000 0xC0000
boardID 0x1780 0x40000 0xC0000
boardID 0x1070 0xA 0x3FF
boardID 0x1170 0xA 0x3FF
boardID 0x1270 0xA 0x3FF
boardID 0x1370 0xA 0x3FF
boardID 0x1470 0xA 0x3FF
boardID 0x1570 0xA 0x3FF
boardID 0x1670 0xA 0x3FF
boardID 0x1770 0xA 0x3FF
boardID 0x106C 0x9 0x3FF
boardID 0x116C 0x9 0x3FF
boardID 0x126C 0x9 0x3FF
boardID 0x136C 0x9 0x3FF
boardID 0x146C 0x9 0x3FF
boardID 0x156C 0x9 0x3FF
boardID 0x166C 0x9 0x3FF
boardID 0x176C 0x9 0x3FF
boardID 0x1084 0x54 0xFF
boardID 0x1284 0x54 0xFF
boardID 0x1484 0x54 0xFF
boardID 0x1684 0x54 0xFF
boardID 0x8180 0x10F 0xFFFFFFFF
boardID 0x8184 0x10F 0xFFFFFFFF
boardID 0x8188 0x10F 0xFFFFFFFF
boardID 0x818C 0x10F 0xFFFFFFFF
```



**Note:** In case of 725 write: boardID 0x1n70 0x5 0x3FF, etc. to get 80 ns of coincidence window.

#### 4.4.11 Coincidence among sixteen channels (ch0 & ch1 & ch2 & ch3 & ch4 & ch5 & ch6 & ch7 & ch8 & ch9 & ch10 & ch11 & ch12 & ch13 & ch14 & ch15) - VME only

All channels of a VME board acquire data within 80 ns of coincidence window. The couples provide as TRG\_REQ the logic AND of the channels and receive the TRG\_VAL from mother board. The mother board FPGA performs the AND between the eight couples.

# Syntax:

# boardID 0x<address> 0x<value> 0x<mask>

```
boardID 0x8000 0x4 0x4
boardID 0x1080 0x40000 0xC0000
boardID 0x1180 0x40000 0xC0000
boardID 0x1280 0x40000 0xC0000
boardID 0x1380 0x40000 0xC0000
boardID 0x1480 0x40000 0xC0000
boardID 0x1580 0x40000 0xC0000
boardID 0x1680 0x40000 0xC0000
boardID 0x1780 0x40000 0xC0000
boardID 0x1880 0x40000 0xC0000
boardID 0x1980 0x40000 0xC0000
boardID 0x1A80 0x40000 0xC0000
boardID 0x1B80 0x40000 0xC0000
boardID 0x1C80 0x40000 0xC0000
boardID 0x1D80 0x40000 0xC0000
boardID 0x1E80 0x40000 0xC0000
boardID 0x1F80 0x40000 0xC0000
boardID 0x1070 0xA 0x3FF
boardID 0x1170 0xA 0x3FF
boardID 0x1270 0xA 0x3FF
boardID 0x1370 0xA 0x3FF
boardID 0x1470 0xA 0x3FF
boardID 0x1570 0xA 0x3FF
boardID 0x1670 0xA 0x3FF
boardID 0x1770 0xA 0x3FF
boardID 0x1870 0xA 0x3FF
boardID 0x1970 0xA 0x3FF
boardID 0x1A70 0xA 0x3FF
boardID 0x1B70 0xA 0x3FF
boardID 0x1C70 0xA 0x3FF
boardID 0x1D70 0xA 0x3FF
boardID 0x1E70 0xA 0x3FF
boardID 0x1F70 0xA 0x3FF
boardID 0x106C 0x9 0x3FF
boardID 0x116C 0x9 0x3FF
boardID 0x126C 0x9 0x3FF
boardID 0x136C 0x9 0x3FF
```

```

boardID 0x146C 0x9 0x3FF
boardID 0x156C 0x9 0x3FF
boardID 0x166C 0x9 0x3FF
boardID 0x176C 0x9 0x3FF
boardID 0x186C 0x9 0x3FF
boardID 0x196C 0x9 0x3FF
boardID 0x1A6C 0x9 0x3FF
boardID 0x1B6C 0x9 0x3FF
boardID 0x1C6C 0x9 0x3FF
boardID 0x1D6C 0x9 0x3FF
boardID 0x1E6C 0x9 0x3FF
boardID 0x1F6C 0x9 0x3FF
boardID 0x1084 0x54 0xFF
boardID 0x1284 0x54 0xFF
boardID 0x1484 0x54 0xFF
boardID 0x1684 0x54 0xFF
boardID 0x1884 0x54 0xFF
boardID 0x1A84 0x54 0xFF
boardID 0x1C84 0x54 0xFF
boardID 0x1E84 0x54 0xFF
boardID 0x8180 0x1FF 0xFFFFFFFF
boardID 0x8184 0x1FF 0xFFFFFFFF
boardID 0x8188 0x1FF 0xFFFFFFFF
boardID 0x818C 0x1FF 0xFFFFFFFF
boardID 0x8190 0x1FF 0xFFFFFFFF
boardID 0x8194 0x1FF 0xFFFFFFFF
boardID 0x8198 0x1FF 0xFFFFFFFF
boardID 0x819C 0x1FF 0xFFFFFFFF

```



**Note:** In case of 725 write: boardID 0x1n70 0x5 0x3FF, etc. to get 80 ns of coincidence window.

#### 4.4.12 Coincidence between ch6 and ch7, in AND with at least one AND of the other pairs: ((ch6 & ch7) & (ch0 & ch1)) || ((ch6 & ch7) & (ch2 & ch3)) || ((ch6 & ch7) & (ch4 & ch5))

Channels acquire data within an 80 ns coincidence window only if channel 6 and channel 7 are in coincidence with each other, and simultaneously in coincidence with at least one additional channel pair (ch0–ch1, ch2–ch3, or ch4–ch5). Each pair must first satisfy its internal coincidence condition.

# Syntax:

# boardID 0x<address> 0x<value> 0x<mask>

```

boardID 0x8000 0x4 0x4
boardID 0x1080 0x40000 0xC0000
boardID 0x1180 0x40000 0xC0000
boardID 0x1280 0x40000 0xC0000
boardID 0x1380 0x40000 0xC0000
boardID 0x1480 0x40000 0xC0000
boardID 0x1580 0x40000 0xC0000
boardID 0x1680 0x40000 0xC0000
boardID 0x1780 0x40000 0xC0000

```

```

boardID 0x1070 0xA 0x3FF
boardID 0x1170 0xA 0x3FF
boardID 0x1270 0xA 0x3FF
boardID 0x1370 0xA 0x3FF
boardID 0x1470 0xA 0x3FF
boardID 0x1570 0xA 0x3FF
boardID 0x1670 0xA 0x3FF
boardID 0x1770 0xA 0x3FF
boardID 0x106C 0x9 0x3FF
boardID 0x116C 0x9 0x3FF
boardID 0x126C 0x9 0x3FF
boardID 0x136C 0x9 0x3FF
boardID 0x146C 0x9 0x3FF
boardID 0x156C 0x9 0x3FF
boardID 0x166C 0x9 0x3FF
boardID 0x176C 0x9 0x3FF
boardID 0x1084 0x54 0xFF
boardID 0x1284 0x54 0xFF
boardID 0x1484 0x54 0xFF
boardID 0x1684 0x54 0xFF
boardID 0x8180 0x109 0xFFFFFFFF
boardID 0x8184 0x10A 0xFFFFFFFF
boardID 0x8188 0x10C 0xFFFFFFFF
boardID 0x818C 0x60F 0xFFFFFFFF

```



**Note:** In case of 725 write: boardID 0x1n70 0x5 0x3FF, etc. to get 80 ns of coincidence window.

#### 4.4.13 Coincidence between ch2 and ch3, in AND with at least one of other 4 channels (ch4-5-6-7): (ch2 & ch3) & (ch4 || ch5 || ch6 || ch7)

Channels 2 and 3, together with at least one channel among ch4, ch5, ch6 and ch7 must acquire data within a 384 ns coincidence window. This type of coincidence requires enabling the "*Additional Local Trigger Validation*" feature (bits[26:25] of register 0x1n84, see Sec. 4.3). When enabled as AND, this feature validate the coincidence of ch2 and ch3 by applying an AND logic operation between the validation signal of the two channels within the couple, and the validation from the motherboard, which requires that at least one signal above threshold is present among ch4, ch5, ch6 and ch7.

# Syntax:

# boardID 0x<address> 0x<value> 0x<mask>

```

boardID 0x8000 0x4 0x4
boardID 0x1280 0x40000 0xC0000
boardID 0x1380 0x40000 0xC0000
boardID 0x1480 0x40000 0xC0000
boardID 0x1580 0x40000 0xC0000
boardID 0x1680 0x40000 0xC0000
boardID 0x1780 0x40000 0xC0000
boardID 0x1270 0x30 0x3FF
boardID 0x1370 0x30 0x3FF
boardID 0x1470 0x30 0x3FF

```

```
boardID 0x1570 0x30 0x3FF
boardID 0x1670 0x30 0x3FF
boardID 0x1770 0x30 0x3FF
boardID 0x126C 0x9 0xFF
boardID 0x136C 0x9 0xFF
boardID 0x146C 0x9 0xFF
boardID 0x156C 0x9 0xFF
boardID 0x166C 0x9 0xFF
boardID 0x176C 0x9 0xFF
boardID 0x1284 0x2000044 0x20000FF
boardID 0x1484 0x57 0xFF
boardID 0x1684 0x57 0xFF
boardID 0x8184 0xC 0xFFFFFFFF
boardID 0x8188 0x2 0xFFFFFFFF
boardID 0x818C 0x2 0xFFFFFFFF
```



**Note:** In case of 725 write: boardID 0x1n70 0x18 0x3FF, etc. to get 384 ns of coincidence window.

#### 4.4.14 Majority of at least three channels among couples of the whole board (sixteen channels) - VME only

All channels of the board acquire when at least three channels are in coincidence within 80 ns (majority level = 2).

# Syntax:

# boardID 0x<address> 0x<value> 0x<mask>

```
boardID 0x8000 0x4 0x4
boardID 0x1080 0x40000 0xC0000
boardID 0x1180 0x40000 0xC0000
boardID 0x1280 0x40000 0xC0000
boardID 0x1380 0x40000 0xC0000
boardID 0x1480 0x40000 0xC0000
boardID 0x1580 0x40000 0xC0000
boardID 0x1680 0x40000 0xC0000
boardID 0x1780 0x40000 0xC0000
boardID 0x1880 0x40000 0xC0000
boardID 0x1980 0x40000 0xC0000
boardID 0x1A80 0x40000 0xC0000
boardID 0x1B80 0x40000 0xC0000
boardID 0x1C80 0x40000 0xC0000
boardID 0x1D80 0x40000 0xC0000
boardID 0x1E80 0x40000 0xC0000
boardID 0x1F80 0x40000 0xC0000
boardID 0x1070 0xA 0x3FF
boardID 0x1170 0xA 0x3FF
boardID 0x1270 0xA 0x3FF
boardID 0x1370 0xA 0x3FF
boardID 0x1470 0xA 0x3FF
boardID 0x1570 0xA 0x3FF
boardID 0x1670 0xA 0x3FF
boardID 0x1770 0xA 0x3FF
boardID 0x1870 0xA 0x3FF
boardID 0x1970 0xA 0x3FF
boardID 0x1A70 0xA 0x3FF
boardID 0x1B70 0xA 0x3FF
boardID 0x1C70 0xA 0x3FF
boardID 0x1D70 0xA 0x3FF
boardID 0x1E70 0xA 0x3FF
boardID 0x1F70 0xA 0x3FF
boardID 0x106C 0x9 0x3FF
boardID 0x116C 0x9 0x3FF
boardID 0x126C 0x9 0x3FF
boardID 0x136C 0x9 0x3FF
boardID 0x146C 0x9 0x3FF
boardID 0x156C 0x9 0x3FF
boardID 0x166C 0x9 0x3FF
boardID 0x176C 0x9 0x3FF
boardID 0x186C 0x9 0x3FF
boardID 0x196C 0x9 0x3FF
boardID 0x1A6C 0x9 0x3FF
boardID 0x1B6C 0x9 0x3FF
boardID 0x1C6C 0x9 0x3FF
boardID 0x1D6C 0x9 0x3FF
```

```
boardID 0x1E6C 0x9 0x3FF
boardID 0x1F6C 0x9 0x3FF
boardID 0x1084 0x57 0xFF
boardID 0x1284 0x57 0xFF
boardID 0x1484 0x57 0xFF
boardID 0x1684 0x57 0xFF
boardID 0x1884 0x57 0xFF
boardID 0x1A84 0x57 0xFF
boardID 0x1C84 0x57 0xFF
boardID 0x1E84 0x57 0xFF
boardID 0x8180 0xAFF 0xFFFFFFFF
boardID 0x8184 0xAFF 0xFFFFFFFF
boardID 0x8188 0xAFF 0xFFFFFFFF
boardID 0x818C 0xAFF 0xFFFFFFFF
boardID 0x8190 0xAFF 0xFFFFFFFF
boardID 0x8194 0xAFF 0xFFFFFFFF
boardID 0x8198 0xAFF 0xFFFFFFFF
boardID 0x819C 0xAFF 0xFFFFFFFF
```



**Note:** To set the majority level = 1 (at least two couples are in coincidence), just modify the last four lines, writing 0x60F rather than 0xAFF



**Note:** In case of 725 write: boardID 0x1n70 0x5 0x3FF, etc. to get 80 ns of coincidence window.



**Note:** In case of DT/NIM board write: boardID 0x8180 (+4n) 0xA0F 0xFFFFFFFF, and remove the unnecessary rows corresponding to unused channels.

#### 4.4.15 Majority of two or more couples for the whole board (eight couples) - VME only

All couples of the board acquire when at least two couples of channels are in coincidence within 80 ns (majority level = 1).

# Syntax:

# boardID 0x<address> 0x<value> 0x<mask>

```
boardID 0x8000 0x4 0x4
boardID 0x1080 0x40000 0xC0000
boardID 0x1180 0x40000 0xC0000
boardID 0x1280 0x40000 0xC0000
boardID 0x1380 0x40000 0xC0000
boardID 0x1480 0x40000 0xC0000
boardID 0x1580 0x40000 0xC0000
boardID 0x1680 0x40000 0xC0000
boardID 0x1780 0x40000 0xC0000
boardID 0x1880 0x40000 0xC0000
boardID 0x1980 0x40000 0xC0000
boardID 0x1A80 0x40000 0xC0000
boardID 0x1B80 0x40000 0xC0000
boardID 0x1C80 0x40000 0xC0000
boardID 0x1D80 0x40000 0xC0000
boardID 0x1E80 0x40000 0xC0000
boardID 0x1F80 0x40000 0xC0000
boardID 0x1070 0xA 0x3FF
boardID 0x1170 0xA 0x3FF
boardID 0x1270 0xA 0x3FF
boardID 0x1370 0xA 0x3FF
boardID 0x1470 0xA 0x3FF
boardID 0x1570 0xA 0x3FF
boardID 0x1670 0xA 0x3FF
boardID 0x1770 0xA 0x3FF
boardID 0x1870 0xA 0x3FF
boardID 0x1970 0xA 0x3FF
boardID 0x1A70 0xA 0x3FF
boardID 0x1B70 0xA 0x3FF
boardID 0x1C70 0xA 0x3FF
boardID 0x1D70 0xA 0x3FF
boardID 0x1E70 0xA 0x3FF
boardID 0x1F70 0xA 0x3FF
boardID 0x106C 0x9 0x3FF
boardID 0x116C 0x9 0x3FF
boardID 0x126C 0x9 0x3FF
boardID 0x136C 0x9 0x3FF
boardID 0x146C 0x9 0x3FF
boardID 0x156C 0x9 0x3FF
boardID 0x166C 0x9 0x3FF
boardID 0x176C 0x9 0x3FF
boardID 0x186C 0x9 0x3FF
boardID 0x196C 0x9 0x3FF
boardID 0x1A6C 0x9 0x3FF
boardID 0x1B6C 0x9 0x3FF
boardID 0x1C6C 0x9 0x3FF
boardID 0x1D6C 0x9 0x3FF
```



```
boardID 0x1E6C 0x9 0x3FF
boardID 0x1F6C 0x9 0x3FF
boardID 0x1084 0x54 0xFF
boardID 0x1284 0x54 0xFF
boardID 0x1484 0x54 0xFF
boardID 0x1684 0x54 0xFF
boardID 0x1884 0x54 0xFF
boardID 0x1A84 0x54 0xFF
boardID 0x1C84 0x54 0xFF
boardID 0x1E84 0x54 0xFF
boardID 0x8180 0x6FF 0xFFFFFFFF
boardID 0x8184 0x6FF 0xFFFFFFFF
boardID 0x8188 0x6FF 0xFFFFFFFF
boardID 0x818C 0x6FF 0xFFFFFFFF
boardID 0x8190 0x6FF 0xFFFFFFFF
boardID 0x8194 0x6FF 0xFFFFFFFF
boardID 0x8198 0x6FF 0xFFFFFFFF
boardID 0x819C 0x6FF 0xFFFFFFFF
```



**Note:** In case of 725 write: boardID 0x1n70 0x5 0x3FF, etc. to get 80 ns of coincidence window.



**Note:** To set the majority level > 1, just modify the last four lines according to the desired majority level.



**Note:** In case of DT/NIM board write: boardID 0x8180 (+4n) 0x60F 0xFFFFFFFF, and remove the unnecessary rows corresponding to unused channels.

#### 4.4.16 Global Trigger to all Channels by the coincidence between ch0 and ch1

Whenever ch0 and ch1 are in coincidence within a window of 80 ns, a global trigger from mother board enables all channels to read and save the event.

```
# Syntax:
# boardID 0x<address> 0x<value> 0x<mask>

boardID 0x8000 0x4 0x4
boardID 0x1080 0x1000000 0x1000000
boardID 0x1180 0x1000000 0x1000000
boardID 0x1280 0x1000000 0x1000000
boardID 0x1380 0x1000000 0x1000000
boardID 0x1480 0x1000000 0x1000000
boardID 0x1580 0x1000000 0x1000000
boardID 0x1680 0x1000000 0x1000000
boardID 0x1780 0x1000000 0x1000000
boardID 0x1070 0xA 0x3FF
boardID 0x1170 0xA 0x3FF
boardID 0x106C 0x2 0x3FF
boardID 0x116C 0x2 0x3FF
boardID 0x1084 0x4 0x7
boardID 0x810C 0x1 0x1
```



**Note:** In case of coincidence inside a couple, set the latency equal to 0x2 (the latency is of two clock cycles, as the coincidence is performed in the channel FPGA only, with no mother board processing).



**Note:** In case of 725 write: boardID 0x1n70 0x5 0x3FF, etc. to get 80 ns of coincidence window.

#### 4.4.17 Global Trigger on ch0, ch1, ch2, ch3 as OR of their Trigger Request and Global Trigger on ch4, ch5, ch6, ch7 as OR of their Trigger Request

The couples provide as TRG\_REQ the logic OR of the channels and receive the TRG\_VAL from mother board. The mother board FPGA performs the OR between the couples 0 and 1, and between the couples 2 and 3.

```
# Syntax:
# boardID 0x<address> 0x<value> 0x<mask>

boardID 0x8000 0x4 0x4
boardID 0x1080 0x1000000 0x1000000
boardID 0x1180 0x1000000 0x1000000
boardID 0x1280 0x1000000 0x1000000
boardID 0x1380 0x1000000 0x1000000
boardID 0x1480 0x1000000 0x1000000
boardID 0x1580 0x1000000 0x1000000
boardID 0x1680 0x1000000 0x1000000
boardID 0x1780 0x1000000 0x1000000
boardID 0x1084 0x57 0xFF
```

```
boardID 0x1284 0x57 0xFF
boardID 0x1484 0x57 0xFF
boardID 0x1684 0x57 0xFF
boardID 0x8180 0x3 0xFFFFFFFF
boardID 0x8184 0x3 0xFFFFFFFF
boardID 0x8188 0xC 0xFFFFFFFF
boardID 0x818C 0xC 0xFFFFFFFF
```



**Note:** In case of 725 write: boardID 0x1n70 0x5 0x3FF, etc. to get 80 ns of coincidence window.

#### 4.4.18 LVDS Global trigger - VME only

All channels acquire data whenever an LVDS signal, acting as a global trigger, is sent to pin 2 of the LVDS groups configured as inputs. In this configuration, LVDS pins [0:7] are set as inputs and LVDS pins [8:15] as outputs, all operating in “nBusy/nVeto” mode.

```
# Syntax:
# boardID 0x<address> 0x<value> 0x<mask>

boardID 0x811C 0x130 0x13C
boardID 0x81A0 0x2222 0x1FFFF
boardID 0x810C 0x20000000 0xFFFFFFFF
```

## 4.5 Custom configuration requests from users

### 4.5.1 Coincidence among four channels (ch0 & ch1 & ch4 & ch5)

The first (ch0-ch1) and the third (ch4-ch5) couple of channels acquire in coincidence within 80 ns.

```
# Syntax:
# boardID 0x<address> 0x<value> 0x<mask>
```

```
boardID 0x8000 0x4 0x4
boardID 0x1080 0x40000 0xC0000
boardID 0x1180 0x40000 0xC0000
boardID 0x1480 0x40000 0xC0000
boardID 0x1580 0x40000 0xC0000
boardID 0x1070 0xA 0x3FF
boardID 0x1170 0xA 0x3FF
boardID 0x1470 0xA 0x3FF
boardID 0x1570 0xA 0x3FF
boardID 0x106C 0x9 0xFF
boardID 0x116C 0x9 0xFF
boardID 0x146C 0x9 0xFF
boardID 0x156C 0x9 0xFF
boardID 0x1084 0x54 0xFF
boardID 0x1484 0x54 0xFF
boardID 0x8180 0x105 0xFFFFFFFF
boardID 0x8188 0x105 0xFFFFFFFF
```



**Note:** In case of 725 write: boardID 0x1n70 0x5 0x3FF, etc. to get 80 ns of coincidence window.

### 4.5.2 Coincidence among four channels (ch0 & ch1 & ch6 & ch7)

The first (ch0-ch1) and the fourth (ch6-ch7) couple of channels acquire in coincidence within 80 ns.

```
# Syntax:
# boardID 0x<address> 0x<value> 0x<mask>
```

```
boardID 0x8000 0x4 0x4
boardID 0x1080 0x40000 0xC0000
boardID 0x1180 0x40000 0xC0000
boardID 0x1680 0x40000 0xC0000
boardID 0x1780 0x40000 0xC0000
boardID 0x1070 0xA 0x3FF
boardID 0x1170 0xA 0x3FF
boardID 0x1670 0xA 0x3FF
boardID 0x1770 0xA 0x3FF
boardID 0x106C 0x9 0xFF
boardID 0x116C 0x9 0xFF
boardID 0x166C 0x9 0xFF
boardID 0x176C 0x9 0xFF
boardID 0x1084 0x54 0xFF
boardID 0x1684 0x54 0xFF
```

```
boardID 0x8180 0x109 0xFFFFFFFF
boardID 0x818C 0x109 0xFFFFFFFF
```



**Note:** In case of 725 write: boardID 0x1n70 0x5 0x3FF, etc. to get 80 ns of coincidence window.

### 4.5.3 Coincidence among four channels (ch2 & ch3 & ch4 & ch5)

The second (ch2-ch3) and the third (ch4-ch5) couple of channels acquire in coincidence within 80 ns.

```
# Syntax:
# boardID 0x<address> 0x<value> 0x<mask>

boardID 0x8000 0x4 0x4
boardID 0x1280 0x40000 0xC0000
boardID 0x1380 0x40000 0xC0000
boardID 0x1480 0x40000 0xC0000
boardID 0x1580 0x40000 0xC0000
boardID 0x1270 0xA 0x3FF
boardID 0x1370 0xA 0x3FF
boardID 0x1470 0xA 0x3FF
boardID 0x1570 0xA 0x3FF
boardID 0x126C 0x9 0xFF
boardID 0x136C 0x9 0xFF
boardID 0x146C 0x9 0xFF
boardID 0x156C 0x9 0xFF
boardID 0x1284 0x54 0xFF
boardID 0x1484 0x54 0xFF
boardID 0x8184 0x106 0xFFFFFFFF
boardID 0x8188 0x106 0xFFFFFFFF
```



**Note:** In case of 725 write: boardID 0x1n70 0x5 0x3FF, etc. to get 80 ns of coincidence window.

### 4.5.4 Coincidence among four channels (ch2 & ch3 & ch6 & ch7)

The second (ch2-ch3) and the fourth (ch6-ch7) couple of channels acquire in coincidence within 80 ns.

```
# Syntax:
# boardID 0x<address> 0x<value> 0x<mask>

boardID 0x8000 0x4 0x4
boardID 0x1280 0x40000 0xC0000
boardID 0x1380 0x40000 0xC0000
boardID 0x1680 0x40000 0xC0000
boardID 0x1780 0x40000 0xC0000
boardID 0x1270 0xA 0x3FF
boardID 0x1370 0xA 0x3FF
boardID 0x1670 0xA 0x3FF
boardID 0x1770 0xA 0x3FF
boardID 0x126C 0x9 0xFF
```

```
boardID 0x136C 0x9 0xFF  
boardID 0x166C 0x9 0xFF  
boardID 0x176C 0x9 0xFF  
boardID 0x1284 0x54 0xFF  
boardID 0x1684 0x54 0xFF  
boardID 0x8184 0x10A 0xFFFFFFFF  
boardID 0x818C 0x10A 0xFFFFFFFF
```



**Note:** In case of 725 write: boardID 0x1n70 0x5 0x3FF, etc. to get 80 ns of coincidence window.

## 4.6 How to configure the registers - DT5790 with DPP-PSD

The register configuration for DT5790 is the same as for the 720 series (refer to Sec. 4.1), apart for the "Trigger Validation Mask" register, where address 0x8188 corresponds to channel 0, and address 0x818C corresponds to channel 1. Moreover bit[2] must be used for channel 0 and bit[3] must be used for channel 1. Refer to **[RD22]** for additional details.

Register name	Address	Register bits	Options
Trigger Validation Mask	0x8188 (ch0)	[2]	0 TRG_REQ[0] not propagated 1 TRG_REQ[0] propagated
		[3]	0 TRG_REQ[1] not propagated 1 TRG_REQ[1] propagated
	0x818C (ch1)	Operation Mask [9:8]	00 OR 01 AND 10 MAJORITY 11 Reserved
		Majority Level [12:10]	value for majority
		[29:13]	Reserved
		External Trigger [30] Software Trigger [31]	0 EXT TRG not propagated 1 EXT TRG propagated 0 SOFT TRG not propagated 1 SOFT TRG propagated

## 4.7 Examples for DT5790

### 4.7.1 Coincidence between channel 0 and channel 1 (ch0 & ch1)

Channel 0 and channel 1 acquire data within 40 ns of coincidence window:

```
# Syntax:
# boardID 0x<address> 0x<value> 0x<mask>

boardID 0x8000 0x4 0x4
boardID 0x1080 0x40000 0xC0000
boardID 0x1180 0x40000 0xC0000
boardID 0x1070 0x5 0x3FF
boardID 0x1170 0x5 0x3FF
boardID 0x106C 0x9 0x3FF
boardID 0x116C 0x9 0x3FF
boardID 0x8188 0x10C 0xFFFFFFFF
boardID 0x818C 0x10C 0xFFFFFFFF
```

## 4.7.2 Anti-coincidence between channel 0 and channel 1 (ch0 $\oplus$ ch1)

Channel 0 and channel 1 acquire data in anti-coincidence within 40 ns of time window:

```
# Syntax:
# boardID 0x<address> 0x<value> 0x<mask>

boardID 0x8000 0x4 0x4
boardID 0x1080 0xC0000 0xC0000
boardID 0x1180 0xC0000 0xC0000
boardID 0x1070 0x5 0x3FF
boardID 0x1170 0x5 0x3FF
boardID 0x106C 0x9 0x3FF
boardID 0x116C 0x9 0x3FF
boardID 0x8188 0x10C 0xFFFFFFFF
boardID 0x818C 0x10C 0xFFFFFFFF
```



## 5 Examples for DPP-PHA firmware

This Chapter is intended to describe how to make coincidences with digitizers and MCAs running DPP-PHA firmware, as listed in **Tab. 1.1**. The firmware registers involved in the coincidence are described in detail, as well as how to modify the software. In this way those who write their own software, and those who use the CAEN software CoMPASS [RD4] can perform on-line onboard coincidences.

### 5.1 How to configure the registers - 724, 781 and 782 series with DPP-PHA

The register configuration syntax used in this Section is the one adopted in the FreeWrites files of the CoMPASS software. For more details about registers, please refer to the 724-781-782 DPP-PHA Register Description [RD23].



**Note:** all values MUST be written in *hexadecimal*.

Here the list of registers to be enabled for the coincidences<sup>1</sup>:

- **Enable the propagation of the Individual Trigger (ITRG) from mother board to mezzanines** to perform the signal validation.

Register name	Address	Register bits	Options
Board Configuration	0x8000	[2]	0 ITRG disabled 1 ITRG enabled

Set bit [2] = 1 of the global channel configuration register (Board Configuration), address 0x8000, i.e. write:

```
boardID 0x8000 0x4 0x4
```

- **Enable the coincidence trigger acquisition mode.** DPP-PHA firmware supports four types of acquisition modes that can be enabled through bits[19:18] of the “DPP Algorithm Control” register.

Possible choices are:

- 00 normal
- 01 “neighbour”
- 10 coincidence
- 11 anti-coincidence

In the *normal acquisition mode*, as described in Sec. 2.1, each channel can either self-trigger on the input pulse or acquire from an external trigger.

The “*neighbour*” acquisition mode is rather quite useful in case of strip detectors, where you want to read data not only from the channel that triggered, but also from the previous and the consecutive channels (the “neighbours”). Indeed, even if a channel is not over-threshold it can receive a TRG\_VAL signal as well, if its neighbour fires the trigger.

In the *coincidence acquisition mode*, as described in Sec. 2.2, each channel self-trigger on the input signal and acquires only when a validation trigger arrives within a certain time window.

<sup>1</sup>For more details about the coincidence features refer to Chap. 2.

The *anti-coincidence acquisition mode* works in the same way of the coincidence mode, but the internal logic is inverted, i.e. the validation occurs when no signal arrives in the acceptance time window. More details can be found in [RD4].

Register name	Address	Register bits	Options
DPP Algorithm Control	0x1n80 (channel n)	[19:18]	00 coincidence disabled 01 "neighbour" enabled 10 coincidence enabled 11 anti-coincidence enabled

To enable the coincidence mode, the user must set bits [19:18] = 10 of the "DPP Algorithm Control" register. Coincidences must be enabled for each channel involved. For example, in case of channel 0 and channel 1 write:

```
boardID 0x1080 0x80000 0xC0000
boardID 0x1180 0x80000 0xC0000
```

- **Set the coincidence windows** (i.e. the width of the shaped trigger) for the trigger request signal ( $T_{ST}$ ).

Register name	Address	Register bits	Options
Shaped Trigger Width	0x1n84 (channel n)	[9:0]	n. clocks (hex)

Set the same value for each channel involved in the coincidence. The value is expressed in trigger clock cycles (10 ns for 724-781-782 series), i.e. you have to write the desired value divided by 10 (in hexadecimal).

For example, if you want to have a coincidence window of 800 ns, you may write:

```
boardID 0x1084 0x50 0xFF
boardID 0x1184 0x50 0xFF
```

- **Write the trigger validation logic**, which corresponds to the logic operation that enables the coincidence. The trigger validation can be generated either by the Individual Trigger Logic, or by the Global Trigger Logic (See Sec. 2.2). Register Trigger Validation Mask 0x8180+(4n), where n is the channel number, manages the Individual Trigger Logic<sup>2</sup>.

It is possible to perform the logic operation between the channels TRG-REQ by modifying bits[9:8], according to the options:

```
00    OR
01    AND
10    MAJORITY
```

Bits [12:10] allow to set the majority level  $m$ , where for a level  $m$  the majority fires when at least  $m+1$  trigger requests are high.

It is possible also to include the LVDS I/O Global Trigger (bit [28] VME only), LVDS I/O Individual Trigger (bit [29] VME only), the External Trigger (bit [30]), and the Software Trigger (bit [31]) in logic OR with the individual TRG-REQ.

For example, to configure the AND between channel 0 and channel 1, write:

```
boardID 0x8180 0x103 0xFFFFFFFF
boardID 0x8184 0x103 0xFFFFFFFF
```

<sup>2</sup>Register address corresponds to 0x8180 for channel 0, 0x8184 for channel 1, 0x8188 for channel 2, 0x818C for channel 3, 0x8190 for channel 4, 0x8194 for channel 5, 0x8198 for channel 6, and 0x819C for channel 7

Register name	Address	Register bits	Options
Trigger Validation Mask	0x8180+(4n) (channel n)	[0]	0 TRG_REQ[0] not propagated 1 TRG_REQ[0] propagated
		[1]	0 TRG_REQ[1] not propagated 1 TRG_REQ[1] propagated
		[2]	0 TRG_REQ[2] not propagated 1 TRG_REQ[2] propagated
		[3]	0 TRG_REQ[3] not propagated 1 TRG_REQ[3] propagated
		[4]	0 TRG_REQ[4] not propagated 1 TRG_REQ[4] propagated
		[5]	0 TRG_REQ[5] not propagated 1 TRG_REQ[5] propagated
		[6]	0 TRG_REQ[6] not propagated 1 TRG_REQ[6] propagated
		[7]	0 TRG_REQ[7] not propagated 1 TRG_REQ[7] propagated
		Operation Mask [9:8]	00 OR 01 AND 10 MAJORITY
		Majority Level [12:10]	value for majority
		[27:13]	0 (Reserved)
		LVDS I/O Global Trigger [28] LVDS I/O Individual Trigger [29] External Trigger [30] Software Trigger [31]	0 LVDS GTRG not propagated 1 LVDS GTRG propagated 0 LVDS ITRG not propagated 1 LVDS ITRG propagated 0 EXT TRG not propagated 1 EXT TRG propagated 0 SOFT TRG not propagated 1 SOFT TRG propagated

## 5.2 Examples for 724, 781 series and V1782 with DPP-PHA

### 5.2.1 Coincidence between channel 0 and channel 1 (ch0 & ch1)

Channel 0 and channel 1 acquire data within 1.5  $\mu$ s of coincidence window.

```
boardID 0x8000 0x4 0x4
boardID 0x1080 0x80000 0xC0000
boardID 0x1180 0x80000 0xC0000
boardID 0x1084 0x96 0xFF
boardID 0x1184 0x96 0xFF
boardID 0x8180 0x103 0xFFFFFFFF
boardID 0x8184 0x103 0xFFFFFFFF
```

### 5.2.2 Anti-coincidence between channel 0 and channel 1 (ch0 $\oplus$ ch1)

Channel 0 and channel 1 acquire data in anti-coincidence mode within 1.5  $\mu$ s of time window.

```
boardID 0x8000 0x4 0x4
boardID 0x1080 0xC0000 0xC0000
boardID 0x1180 0xC0000 0xC0000
boardID 0x1084 0x96 0xFF
boardID 0x1184 0x96 0xFF
boardID 0x8180 0x103 0xFFFFFFFF
boardID 0x8184 0x103 0xFFFFFFFF
```

### 5.2.3 Coincidence among four channels (ch0 & ch1 & ch2 & ch3)

Channel 0 to channel 3 acquire data within 1.5  $\mu$ s of coincidence window.

```
boardID 0x8000 0x4 0x4
boardID 0x1080 0x80000 0xC0000
boardID 0x1180 0x80000 0xC0000
boardID 0x1280 0x80000 0xC0000
boardID 0x1380 0x80000 0xC0000
boardID 0x1084 0x96 0xFF
boardID 0x1184 0x96 0xFF
boardID 0x1284 0x96 0xFF
boardID 0x1384 0x96 0xFF
boardID 0x8180 0x10F 0xFFFFFFFF
boardID 0x8184 0x10F 0xFFFFFFFF
boardID 0x8188 0x10F 0xFFFFFFFF
boardID 0x818C 0x10F 0xFFFFFFFF
```

## 5.2.4 Odd channels validated by the coincidence with the previous even channel

Odd channels (ch1, ch3, ch5 and ch7) acquire data if they are in coincidence within 1  $\mu$ s of coincidence window with the previous even channel.

```
boardID 0x8000 0x4 0x4
boardID 0x1180 0x80000 0xC0000
boardID 0x1380 0x80000 0xC0000
boardID 0x1580 0x80000 0xC0000
boardID 0x1780 0x80000 0xC0000
boardID 0x1084 0x64 0x3FF
boardID 0x1184 0x64 0x3FF
boardID 0x1284 0x64 0x3FF
boardID 0x1384 0x64 0x3FF
boardID 0x1484 0x64 0x3FF
boardID 0x1584 0x64 0x3FF
boardID 0x1684 0x64 0x3FF
boardID 0x1784 0x64 0x3FF
boardID 0x8184 0x103 0xFFFFFFFF
boardID 0x818C 0x10C 0xFFFFFFFF
boardID 0x8194 0x130 0xFFFFFFFF
boardID 0x819C 0x1C0 0xFFFFFFFF
```

## 5.2.5 Anticoincidence among ch0 and the OR of ch1, ch2, ch3 (ch0 $\oplus$ (ch1 || ch2 || ch3))

Channel 0 is in anti-coincidence with the OR of ch1, ch2 and ch3 within a window of 1  $\mu$ s. Channel 0 is vetoed if one among ch1, ch2 or ch3 sends a validation signal.

```
boardID 0x8000 0x4 0x4
boardID 0x1080 0xC0000 0xC0000
boardID 0x1084 0x64 0xFF
boardID 0x8180 0xE 0xFFFFFFFF
```

## 5.2.6 Majority of at least two channels among three

The first three channels of the board acquire data when at least two of them are in coincidence within 1  $\mu$ s (majority level = 2).

```
boardID 0x8000 0x4 0x4
boardID 0x1080 0x80000 0xC0000
boardID 0x1180 0x80000 0xC0000
boardID 0x1280 0x80000 0xC0000
boardID 0x1084 0x64 0xFF
boardID 0x1184 0x64 0xFF
boardID 0x1284 0x64 0xFF
boardID 0x8180 0x607 0xFFFFFFFF
boardID 0x8184 0x607 0xFFFFFFFF
boardID 0x8188 0x607 0xFFFFFFFF
```

## 5.3 How to configure the registers - 780 series with DPP-PHA

The register configuration for 780 is the same as for the 724, 781 and 782 series (refer to Sec. 5.1), apart for the "Trigger Validation Mask" register, where address 0x8188 corresponds to channel 0, and address 0x818C corresponds to channel 1. Moreover bit[2] must be used for channel 0 and bit[3] must be used for channel 1. For more details about registers, please refer to the 780 DPP-PHA Registers Description [RD24].

Register name	Address	Register bits	Options
Trigger Validation Mask	0x8188 (ch0) 0x818C (ch1)	[2]	0 TRG_REQ[0] not propagated 1 TRG_REQ[0] propagated
		[3]	0 TRG_REQ[1] not propagated 1 TRG_REQ[1] propagated
		Operation Mask [9:8]	00 OR 01 AND 10 MAJORITY
		Majority Level [12:10]	value for majority
		[27:13]	0 (Reserved)
		External Trigger [30] Software Trigger [31]	0 EXT TRG not propagated 1 EXT TRG propagated 0 SOFT TRG not propagated 1 SOFT TRG propagated

## 5.4 Examples for 780 series with DPP-PHA

### 5.4.1 Coincidence between channel 0 and channel 1 (ch0 & ch1)

Channel 0 and channel 1 acquire data within 1.5  $\mu$ s of coincidence window.

```
boardID 0x8000 0x4 0x4
boardID 0x1080 0x80000 0xC0000
boardID 0x1180 0x80000 0xC0000
boardID 0x1084 0x96 0xFF
boardID 0x1184 0x96 0xFF
boardID 0x8188 0x10C 0xFFFFFFFF
boardID 0x818C 0x10C 0xFFFFFFFF
```

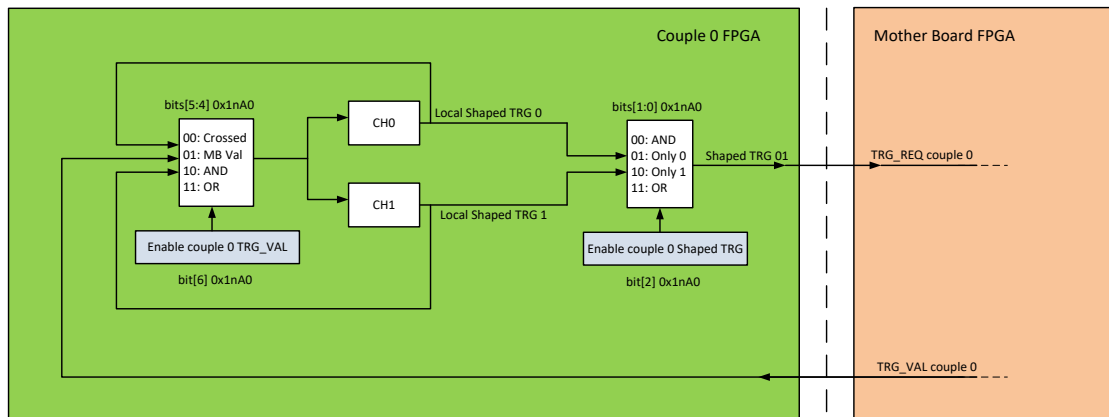
### 5.4.2 Anti-coincidence between channel 0 and channel 1 (ch0 $\oplus$ ch1)

Channel 0 and channel 1 acquire data in anti-coincidence mode within 1.5  $\mu$ s of time window.

```
boardID 0x8000 0x4 0x4
boardID 0x1080 0xC0000 0xC0000
boardID 0x1180 0xC0000 0xC0000
boardID 0x1084 0x96 0xFF
boardID 0x1184 0x96 0xFF
boardID 0x8188 0x10C 0xFFFFFFFF
boardID 0x818C 0x10C 0xFFFFFFFF
```

## 5.5 How to configure the registers - 725 and 730 series with DPP-PHA

As mentioned in Sec. 2.2, couples of channels in 725 and 730 series share the same TRG\_REQ. The local Shaped Trigger and local Trigger Validation options can be configured through register "DPP Algorithm Control 2", at address 0x1nA0 [RD10], and they are summarized in Fig. 5.1. The n index identifies the n-th channel. The register value must be equal for channels of the same couple. Writing the n channel value, it will write the same value in the n+1 channel (channels of the same couple). Refer to [RD10] for more details.



**Fig. 5.1:** Local Trigger Management inside couple 0 of 725-730 digitizer series. Couple 0 is made of channel 0 and channel 1. The same applies for the other couples of the 725-730 digitizers.

The TRG\_REQ of the couple can be the AND, OR of the two channels, or "only one" of the two channels. The TRG\_VAL can be generated from mother board, from the other channel of the couple, or it can be the logic AND/OR of the two shaped TRG.

In case of coincidence inside the couple (ch0 & ch1) it is recommended to use option AND (10).

Here the list of registers to be set to enable the coincidence mode.

- **Enable the propagation of the Individual Trigger (ITRG) from mother board to mezzanines** to perform the signal validation.

Register name	Address	Register bits	Options
Board Configuration	0x8000	[2]	0 ITRG disabled 1 ITRG enabled

Set bit [2] = 1 of the global channel configuration register (Board Configuration), address 0x8000, i.e. write:

```
boardID 0x8000 0x4 0x4
```

- **Enable the coincidence trigger acquisition mode.** DPP-PHA firmware for 725-730 series supports three types of acquisition modes that can be enabled through bits[19:18] of the "DPP Algorithm Control" register. Possible choices are:
  - 00 normal
  - 01 coincidence
  - 11 anti-coincidence

In the *normal acquisition mode*, as described in Sec. 2.1, each channel can either self-trigger on the input pulse or acquire from an external trigger. In the *coincidence acquisition mode*, as described in Sec. 2.2, each channel self-trigger on the input signal and acquires only when a validation signal arrives within a certain time window. The *anti-coincidence acquisition mode* works in the same way as the coincidence mode, but the internal logic is the opposite, i.e. the validation occurs when no signal arrives in the acceptance time window.

Register name	Address	Register bits	Options
DPP Algorithm Control	0x1n80 (channel n)	[19:18]	00 coincidence disabled 01 coincidence enabled 11 anti-coincidence enabled

Set bits [19:18] = 01 of the channel control register (DPP Algorithm Control). Coincidences must be enabled for each channel involved; for example in case of channel 0 and channel 1 write:

```
boardID 0x1080 0x40000 0xC0000
boardID 0x1180 0x40000 0xC0000
```

- **Set the coincidence windows** (i.e. the width of the shaped trigger) for the trigger request signal ( $T_{ST}$ ).

Register name	Address	Register bits	Options
Shaped Trigger Width	0x1n84 (channel n)	[7:0]	n. clocks (hex)

Set the same value for each channel involved in the coincidence. The value is expressed in steps of 16 ns for 725 and 8 ns for 730 series. For example, if you want to have a window of 1.6  $\mu$ s, you may write for 730:

```
boardID 0x1084 0xC8 0xFF
boardID 0x1184 0xC8 0xFF
```

and for 725:

```
boardID 0x1084 0x64 0xFF
boardID 0x1184 0x64 0xFF
```

- **Set the "DPP Algorithm Control 2" register.** This register manages the local TRG\_REQ of the couple (first three bits), and the TRG\_VAL origin for the couple itself (bits[6:4]). The TRG\_REQ of the couple can be the AND, OR of the two channels, or one of the two channels. The TRG\_VAL can be generated from mother board, from the other channel of the couple, or it can be the logic AND/OR of the two possibilities.

For example, to set the validation from the other channel of the couple (TRG\_REQ to mother board not enabled), write:

```
boardID 0x10A0 0x60 0xFF
```

See more examples in the next Section.



Register name	Address	Register bits	Options
DPP Algorithm Control 2	0x1nA0	[1:0]	00: TRG_REQ is the AND of the channels 01: TRG_REQ is the even channel of the couple 10: TRG_REQ is the odd channel of the couple 11: TRG_REQ is the OR of the channels
		[2]	Enable TRG_REQ of the couple
		[3]	Reserved
		[5:4]	00: crossed (TRG_VAL0 = TRG_REQ1; TRG_VAL1 = TRG_REQ0); 01: TRG_VAL0 = TRG_VAL1 = signal from mother-board mask; 10: AND (TRG_VAL0 = TRG_VAL1 = TRG_REQ0 AND TRG_REQ1); 11: OR (TRG_VAL0 = TRG_VAL1 = TRG_REQ0 OR TRG_REQ1).
		[6]	Enable TRG_VAL of the couple



**Note:** In the address  $0x1nA0$ ,  $n$  corresponds to the  $n$ -th channel. The register value must be equal for channels of the same couple. Writing the  $2n$  channel value, will write the same value in the  $2n+1$  channel (channels of the same couple). Refer to **[RD10]**.

- **Write the trigger validation logic**, which corresponds to the logic operation that enables the coincidence among couples. The trigger validation can be generated either by the Individual Trigger Logic, or by the Global Trigger Logic (see Sec. 2). Register Trigger Validation Mask  $0x8180+(4n)$ , where  $n$  is the couple number, manages the Individual Trigger Logic<sup>3</sup>.

It is possible to perform the logic operation between the couples TRG-REQ by modifying bits[9:8], according to the options: OR = 00, AND = 01, and MAJORITY = 10. Bits [12:10] allow to set the majority level  $m$ , where for a level  $m$  the majority fires when at least  $m+1$  trigger requests are high. It is possible also to include the LVDS I/O Global Trigger (bit [28] VME only), LVDS I/O Individual Trigger (bit [29] VME only), the External Trigger (bit [30]), and the Software Trigger (bit [31]) in logic OR with the individual TRG-REQ.

Register name	Address	Register bits	Options
Trigger Validation Mask	$0x8180+(4n)$ (couple $n$ )	[0]	0 TRG_REQ[0] not propagated 1 TRG_REQ[0] propagated
		[1]	0 TRG_REQ[1] not propagated 1 TRG_REQ[1] propagated
		[2]	0 TRG_REQ[2] not propagated 1 TRG_REQ[2] propagated
		[3]	0 TRG_REQ[3] not propagated 1 TRG_REQ[3] propagated
		[4]	0 TRG_REQ[4] not propagated 1 TRG_REQ[4] propagated
		[5]	0 TRG_REQ[5] not propagated 1 TRG_REQ[5] propagated
		[6]	0 TRG_REQ[6] not propagated 1 TRG_REQ[6] propagated
		[7]	0 TRG_REQ[7] not propagated 1 TRG_REQ[7] propagated
		Operation Mask [9:8]	00 OR 01 AND 10 MAJORITY
		Majority Level [12:10]	value for majority
		[27:13]	0 (Reserved)
		LVDS I/O Global Trigger [28] LVDS I/O Individual Trigger [29] External Trigger [30] Software Trigger [31]	0 LVDS GTRG not propagated 1 LVDS GTRG propagated 0 LVDS ITRG not propagated 1 LVDS ITRG propagated 0 EXT TRG not propagated 1 EXT TRG propagated 0 SOFT TRG not propagated 1 SOFT TRG propagated

For example, to configure the AND between couple 0 and couple 1, write:

```
boardID 0x8180 0x103 0xFFFFFFFF
boardID 0x8184 0x103 0xFFFFFFFF
```

<sup>3</sup>Register address corresponds to 0x8180 for couple 0, 0x8184 for couple 1, 0x8188 for couple 2, 0x818C for couple 3, 0x8190 for couple 4, 0x8194 for couple 5, 0x8198 for couple 6, and 0x819C for couple 7

## 5.6 Examples for 725-730 series with DPP-PHA

### 5.6.1 Coincidence between channel 0 and channel 1 (ch0 & ch1)

Channel 0 and channel 1 acquire data within 1.6  $\mu$ s of coincidence window.

```
boardID 0x1080 0x40000 0xC0000
boardID 0x1180 0x40000 0xC0000
boardID 0x1084 0xC8 0xFF
boardID 0x1184 0xC8 0xFF
boardID 0x10A0 0x60 0xFF
```



**Note:** In case of 725 write: boardID 0x1n84 0x64 0xFF for n=0,...,7 to get 1.6  $\mu$ s of coincidence window.

### 5.6.2 Coincidence inside the couples (ch0 & ch1), (ch2 & ch3), (ch4 & ch5), (ch6 & ch7)

Couples of channels (up to channel 7) acquire data within 1.6  $\mu$ s of coincidence window.

```
boardID 0x1080 0x40000 0xC0000
boardID 0x1180 0x40000 0xC0000
boardID 0x1280 0x40000 0xC0000
boardID 0x1380 0x40000 0xC0000
boardID 0x1480 0x40000 0xC0000
boardID 0x1580 0x40000 0xC0000
boardID 0x1680 0x40000 0xC0000
boardID 0x1780 0x40000 0xC0000
boardID 0x1084 0xC8 0xFF
boardID 0x1184 0xC8 0xFF
boardID 0x1284 0xC8 0xFF
boardID 0x1384 0xC8 0xFF
boardID 0x1484 0xC8 0xFF
boardID 0x1584 0xC8 0xFF
boardID 0x1684 0xC8 0xFF
boardID 0x1784 0xC8 0xFF
boardID 0x10A0 0x60 0xFF
boardID 0x12A0 0x60 0xFF
boardID 0x14A0 0x60 0xFF
boardID 0x16A0 0x60 0xFF
```



**Note:** In case of 725 write: boardID 0x1n84 0x64 0xFF for n=0,...,7 to get 1.6  $\mu$ s of coincidence window.

### 5.6.3 Coincidence inside the couples (ch0 & ch1), (ch2 & ch3), (ch4 & ch5), (ch6 & ch7), (ch8 & ch9), (ch10 & ch11), (ch12 & ch13), (ch14 & ch15) - VME only

Couples of channels (up to channel 15 for VME form factor) acquire data within 1.6  $\mu$ s of coincidence window.

```
boardID 0x1080 0x40000 0xC0000
boardID 0x1180 0x40000 0xC0000
boardID 0x1280 0x40000 0xC0000
boardID 0x1380 0x40000 0xC0000
boardID 0x1480 0x40000 0xC0000
boardID 0x1580 0x40000 0xC0000
boardID 0x1680 0x40000 0xC0000
boardID 0x1780 0x40000 0xC0000
boardID 0x1880 0x40000 0xC0000
boardID 0x1980 0x40000 0xC0000
boardID 0x1A80 0x40000 0xC0000
boardID 0x1B80 0x40000 0xC0000
boardID 0x1C80 0x40000 0xC0000
boardID 0x1D80 0x40000 0xC0000
boardID 0x1E80 0x40000 0xC0000
boardID 0x1F80 0x40000 0xC0000
boardID 0x1084 0xC8 0xFF
boardID 0x1184 0xC8 0xFF
boardID 0x1284 0xC8 0xFF
boardID 0x1384 0xC8 0xFF
boardID 0x1484 0xC8 0xFF
boardID 0x1584 0xC8 0xFF
boardID 0x1684 0xC8 0xFF
boardID 0x1784 0xC8 0xFF
boardID 0x1884 0xC8 0xFF
boardID 0x1984 0xC8 0xFF
boardID 0x1A84 0xC8 0xFF
boardID 0x1B84 0xC8 0xFF
boardID 0x1C84 0xC8 0xFF
boardID 0x1D84 0xC8 0xFF
boardID 0x1E84 0xC8 0xFF
boardID 0x1F84 0xC8 0xFF
boardID 0x10A0 0x60 0xFF
boardID 0x12A0 0x60 0xFF
boardID 0x14A0 0x60 0xFF
boardID 0x16A0 0x60 0xFF
boardID 0x18A0 0x60 0xFF
boardID 0x1AA0 0x60 0xFF
boardID 0x1CA0 0x60 0xFF
boardID 0x1EA0 0x60 0xFF
```



**Note:** In case of 725 write: boardID 0x1n84 0x64 0xFF for n=0,...,7 to get 1.6  $\mu$ s of coincidence window.

### 5.6.4 Coincidence between channels of different couples (ch0 & ch2)

Channels ch0 and ch2 acquire data within 1.6  $\mu$ s of coincidence window.

```
boardID 0x8000 0x4 0x4
boardID 0x1080 0x40000 0xC0000
boardID 0x1280 0x40000 0xC0000
boardID 0x1084 0xC8 0xFF
boardID 0x1284 0xC8 0xFF
boardID 0x10A0 0x55 0xFF
boardID 0x12A0 0x55 0xFF
boardID 0x8180 0x103
boardID 0x8184 0x103
```



**Note:** In case of 725 write: boardID 0x1n84 0x64 0xFF for n=0,...,7 to get 1.6  $\mu$ s of coincidence window.

### 5.6.5 Coincidence among four channels (four different couples, ch0 & ch2 & ch4 & ch6)

Even channels of the first four couples acquire data within 1.6  $\mu$ s of coincidence window. The four couples provide as TRG\_REQ only the even channels and receive the TRG\_VAL from mother board. The mother board FPGA performs the AND between the four couples.



**Note:** Since the two channels of the couple share the same memory it is recommended to use only one channel per couple, if available. If it is not possible to use one channel per couple, see the example in the next section.



**Note:** The odd channels of the couples do not participate to the coincidence logic.



**Note:** In case of 725 write: boardID 0x1n84 0x64 0xFF for n=0,...,7 to get 1.6  $\mu$ s of coincidence window.

```
boardID 0x8000 0x4 0x4
boardID 0x1080 0x40000 0xC0000
boardID 0x1280 0x40000 0xC0000
boardID 0x1480 0x40000 0xC0000
boardID 0x1680 0x40000 0xC0000
boardID 0x1084 0xC8 0xFF
boardID 0x1284 0xC8 0xFF
boardID 0x1484 0xC8 0xFF
boardID 0x1684 0xC8 0xFF
boardID 0x10A0 0x55 0xFF
boardID 0x12A0 0x55 0xFF
boardID 0x14A0 0x55 0xFF
boardID 0x16A0 0x55 0xFF
boardID 0x8180 0x10F 0xFFFFFFFF
boardID 0x8184 0x10F 0xFFFFFFFF
boardID 0x8188 0x10F 0xFFFFFFFF
boardID 0x818C 0x10F 0xFFFFFFFF
```

### 5.6.6 Coincidence among four channels (two different couples, ch0 & ch1 & ch2 & ch3)

Channel 0 to channel 3 acquire data within 1.6  $\mu$ s of coincidence window. The two couples provide as TRG\_REQ the AND of the two channels and receive the TRG\_VAL from mother board. The mother board FPGA performs the AND between couple 0 and couple 1.



**Note:** Since the two channels of the couple share the same memory it is usually suggested to perform the coincidence as in the previous example - if channels are not otherwise occupied.



**Note:** In case of 725 write: boardID 0x1n84 0x64 0xFF for n=0,...,7 to get 1.6  $\mu$ s of coincidence window.

```
boardID 0x8000 0x4 0x4
boardID 0x1080 0x40000 0xC0000
boardID 0x1180 0x40000 0xC0000
boardID 0x1280 0x40000 0xC0000
boardID 0x1380 0x40000 0xC0000
boardID 0x1084 0xC8 0xFF
boardID 0x1184 0xC8 0xFF
boardID 0x1284 0xC8 0xFF
boardID 0x1384 0xC8 0xFF
boardID 0x10A0 0x54 0xFF
boardID 0x12A0 0x54 0xFF
boardID 0x8180 0x103 0xFFFFFFFF
boardID 0x8184 0x103 0xFFFFFFFF
```

### 5.6.7 Coincidence among eight channels (eight different couples, ch0 & ch2 & ch4 & ch6 & ch8 & ch10 & ch12 & ch14) - VME only

Even channels of the eight couples (VME only) acquire data within 1.6  $\mu$ s of coincidence window. The couples provide as TRG\_REQ only the even channel and receive the TRG\_VAL from mother board. The mother board FPGA performs the AND between the eight couples.



**Note:** Since the two channels of the couple share the same memory it is recommended to use only one channel per couple, if available. If it is not possible to use one channel per couple, see the example in the next section.



**Note:** The odd channels of the couples do not participate to the coincidence logic.



**Note:** In case of 725 write: boardID 0x1n84 0x64 0xFF for n=0,...,7 to get 1.6  $\mu$ s of coincidence window.

```
boardID 0x8000 0x4 0x4
boardID 0x1080 0x40000 0xC0000
boardID 0x1280 0x40000 0xC0000
boardID 0x1480 0x40000 0xC0000
boardID 0x1680 0x40000 0xC0000
boardID 0x1880 0x40000 0xC0000
boardID 0x1A80 0x40000 0xC0000
boardID 0x1C80 0x40000 0xC0000
boardID 0x1E80 0x40000 0xC0000
boardID 0x1084 0xC8 0xFF
boardID 0x1284 0xC8 0xFF
boardID 0x1484 0xC8 0xFF
boardID 0x1684 0xC8 0xFF
boardID 0x1884 0xC8 0xFF
boardID 0x1A84 0xC8 0xFF
boardID 0x1C84 0xC8 0xFF
boardID 0x1E84 0xC8 0xFF
boardID 0x10A0 0x55 0xFF
boardID 0x12A0 0x55 0xFF
boardID 0x14A0 0x55 0xFF
boardID 0x16A0 0x55 0xFF
boardID 0x18A0 0x55 0xFF
boardID 0x1AA0 0x55 0xFF
boardID 0x1CA0 0x55 0xFF
boardID 0x1EA0 0x55 0xFF
boardID 0x8180 0x1FF 0xFFFFFFFF
boardID 0x8184 0x1FF 0xFFFFFFFF
boardID 0x8188 0x1FF 0xFFFFFFFF
boardID 0x818C 0x1FF 0xFFFFFFFF
boardID 0x8190 0x1FF 0xFFFFFFFF
boardID 0x8194 0x1FF 0xFFFFFFFF
boardID 0x8198 0x1FF 0xFFFFFFFF
boardID 0x819C 0x1FF 0xFFFFFFFF
```

### 5.6.8 Coincidence among eight channels (four different couples, ch0 & ch1 & ch2 & ch3 & ch4 & ch5 & ch6 & ch7)

Channel 0 to channel 7 acquire data within 1.6  $\mu$ s of coincidence window. The four couples provide as TRG\_REQ the AND of the two channels and receive the TRG\_VAL from mother board. The mother board FPGA performs the AND between the four couples.



**Note:** Since the two channels of the couple share the same memory it is usually suggested to perform the coincidence as in the previous example (VME only) - if channels are not otherwise occupied.



**Note:** In case of 725 write: boardID 0x1n84 0x64 0xFF for n=0,...,7 to get 1.6  $\mu$ s of coincidence window.

```
boardID 0x8000 0x4 0x4
boardID 0x1080 0x40000 0xC0000
boardID 0x1180 0x40000 0xC0000
boardID 0x1280 0x40000 0xC0000
boardID 0x1380 0x40000 0xC0000
boardID 0x1480 0x40000 0xC0000
boardID 0x1580 0x40000 0xC0000
boardID 0x1680 0x40000 0xC0000
boardID 0x1780 0x40000 0xC0000
boardID 0x1084 0xC8 0xFF
boardID 0x1184 0xC8 0xFF
boardID 0x1284 0xC8 0xFF
boardID 0x1384 0xC8 0xFF
boardID 0x1484 0xC8 0xFF
boardID 0x1584 0xC8 0xFF
boardID 0x1684 0xC8 0xFF
boardID 0x1784 0xC8 0xFF
boardID 0x10A0 0x54 0xFF
boardID 0x12A0 0x54 0xFF
boardID 0x14A0 0x54 0xFF
boardID 0x16A0 0x54 0xFF
boardID 0x8180 0x10F 0xFFFFFFFF
boardID 0x8184 0x10F 0xFFFFFFFF
boardID 0x8188 0x10F 0xFFFFFFFF
boardID 0x818C 0x10F 0xFFFFFFFF
```

### 5.6.9 Coincidence among sixteen channels (ch0 & ch1 & ch2 & ch3 & ch4 & ch5 & ch6 & ch7 & ch8 & ch9 & ch10 & ch11 & ch12 & ch13 & ch14 & ch15) - VME only

All channels of a VME board acquire data within 1.6  $\mu$ s of coincidence window. Each couple provide as TRG\_REQ the AND of the two channels and receive the TRG\_VAL from mother board. The mother board FPGA performs the AND between the couples.

```
boardID 0x8000 0x4 0x4
boardID 0x1080 0x40000 0xC0000
boardID 0x1180 0x40000 0xC0000
boardID 0x1280 0x40000 0xC0000
boardID 0x1380 0x40000 0xC0000
```



```

boardID 0x1480 0x40000 0xC0000
boardID 0x1580 0x40000 0xC0000
boardID 0x1680 0x40000 0xC0000
boardID 0x1780 0x40000 0xC0000
boardID 0x1880 0x40000 0xC0000
boardID 0x1980 0x40000 0xC0000
boardID 0x1A80 0x40000 0xC0000
boardID 0x1B80 0x40000 0xC0000
boardID 0x1C80 0x40000 0xC0000
boardID 0x1D80 0x40000 0xC0000
boardID 0x1E80 0x40000 0xC0000
boardID 0x1F80 0x40000 0xC0000
boardID 0x1084 0xC8 0xFF
boardID 0x1184 0xC8 0xFF
boardID 0x1284 0xC8 0xFF
boardID 0x1384 0xC8 0xFF
boardID 0x1484 0xC8 0xFF
boardID 0x1584 0xC8 0xFF
boardID 0x1684 0xC8 0xFF
boardID 0x1784 0xC8 0xFF
boardID 0x1884 0xC8 0xFF
boardID 0x1984 0xC8 0xFF
boardID 0x1A84 0xC8 0xFF
boardID 0x1B84 0xC8 0xFF
boardID 0x1C84 0xC8 0xFF
boardID 0x1D84 0xC8 0xFF
boardID 0x1E84 0xC8 0xFF
boardID 0x1F84 0xC8 0xFF
boardID 0x10A0 0x54 0xFF
boardID 0x12A0 0x54 0xFF
boardID 0x14A0 0x54 0xFF
boardID 0x16A0 0x54 0xFF
boardID 0x18A0 0x54 0xFF
boardID 0x1AA0 0x54 0xFF
boardID 0x1CA0 0x54 0xFF
boardID 0x1EA0 0x54 0xFF
boardID 0x8180 0x1FF 0xFFFFFFFF
boardID 0x8184 0x1FF 0xFFFFFFFF
boardID 0x8188 0x1FF 0xFFFFFFFF
boardID 0x818C 0x1FF 0xFFFFFFFF
boardID 0x8190 0x1FF 0xFFFFFFFF
boardID 0x8194 0x1FF 0xFFFFFFFF
boardID 0x8198 0x1FF 0xFFFFFFFF
boardID 0x819C 0x1FF 0xFFFFFFFF

```



**Note:** In case of 725 write: boardID 0x1n84 0x64 0xFF for n=0,...,7 to get 1.6  $\mu$ s of coincidence window.

### 5.6.10 Majority of at least three channels among couples of the whole board (sixteen channels) - VME only

All channels of the board acquire data when at least three channels are in coincidence within 80 ns (majority level = 2).

# Syntax:

# boardID 0x<address> 0x<value> 0x<mask>

```
boardID 0x8000 0x4 0x4
boardID 0x1080 0x40000 0xC0000
boardID 0x1180 0x40000 0xC0000
boardID 0x1280 0x40000 0xC0000
boardID 0x1380 0x40000 0xC0000
boardID 0x1480 0x40000 0xC0000
boardID 0x1580 0x40000 0xC0000
boardID 0x1680 0x40000 0xC0000
boardID 0x1780 0x40000 0xC0000
boardID 0x1880 0x40000 0xC0000
boardID 0x1980 0x40000 0xC0000
boardID 0x1A80 0x40000 0xC0000
boardID 0x1B80 0x40000 0xC0000
boardID 0x1C80 0x40000 0xC0000
boardID 0x1D80 0x40000 0xC0000
boardID 0x1E80 0x40000 0xC0000
boardID 0x1F80 0x40000 0xC0000
boardID 0x1084 0xA 0x3FF
boardID 0x1184 0xA 0x3FF
boardID 0x1284 0xA 0x3FF
boardID 0x1384 0xA 0x3FF
boardID 0x1484 0xA 0x3FF
boardID 0x1584 0xA 0x3FF
boardID 0x1684 0xA 0x3FF
boardID 0x1784 0xA 0x3FF
boardID 0x1884 0xA 0x3FF
boardID 0x1984 0xA 0x3FF
boardID 0x1A84 0xA 0x3FF
boardID 0x1B84 0xA 0x3FF
boardID 0x1C84 0xA 0x3FF
boardID 0x1D84 0xA 0x3FF
boardID 0x1E84 0xA 0x3FF
boardID 0x1F84 0xA 0x3FF
boardID 0x10A0 0x57 0xFF
boardID 0x12A0 0x57 0xFF
boardID 0x14A0 0x57 0xFF
boardID 0x16A0 0x57 0xFF
boardID 0x18A0 0x57 0xFF
boardID 0x1AA0 0x57 0xFF
boardID 0x1CA0 0x57 0xFF
boardID 0x1EA0 0x57 0xFF
boardID 0x8180 0xAFF 0xFFFFFFFF
boardID 0x8184 0xAFF 0xFFFFFFFF
boardID 0x8188 0xAFF 0xFFFFFFFF
boardID 0x818C 0xAFF 0xFFFFFFFF
boardID 0x8190 0xAFF 0xFFFFFFFF
boardID 0x8194 0xAFF 0xFFFFFFFF
```

```
boardID 0x8198 0xAFF 0xFFFFFFFF
boardID 0x819C 0xAFF 0xFFFFFFFF
```



**Note:** To set the majority level = 1 (at least two couples are in coincidence), just modify the last four lines, writing 0x60F rather than 0xAFF



**Note:** In case of 725 write: boardID 0x1n84 0x5 0x3FF, etc. to get 80 ns of coincidence window.



**Note:** In case of DT/NIM board write: boardID 0x8180 (+4n) 0xA0F 0xFFFFFFFF, and remove the unnecessary rows corresponding to unused channels.

### 5.6.11 Majority of at least three couples among four couples (eight channels)

The first eight channels of the board acquire data when at least three couples are in coincidence within 1.6  $\mu$ s (majority level = 2).



**Note:** To set the majority level = 1 (at least two couples are in coincidence), just modify the last four registers, writing 0x60F rather than 0xA0F.



**Note:** In case of 725 write: boardID 0x1n84 0x64 0xFF for n=0,...,7 to get 1.6  $\mu$ s of coincidence window.

```
boardID 0x8000 0x4 0x4
boardID 0x1080 0x40000 0xC0000
boardID 0x1180 0x40000 0xC0000
boardID 0x1280 0x40000 0xC0000
boardID 0x1380 0x40000 0xC0000
boardID 0x1480 0x40000 0xC0000
boardID 0x1580 0x40000 0xC0000
boardID 0x1680 0x40000 0xC0000
boardID 0x1780 0x40000 0xC0000
boardID 0x1084 0xC8 0xFF
boardID 0x1184 0xC8 0xFF
boardID 0x1284 0xC8 0xFF
boardID 0x1384 0xC8 0xFF
boardID 0x1484 0xC8 0xFF
boardID 0x1584 0xC8 0xFF
boardID 0x1684 0xC8 0xFF
boardID 0x1784 0xC8 0xFF
boardID 0x10A0 0x54 0xFF
boardID 0x12A0 0x54 0xFF
boardID 0x14A0 0x54 0xFF
boardID 0x16A0 0x54 0xFF
boardID 0x8180 0xA0F 0xFFFFFFFF
boardID 0x8184 0xA0F 0xFFFFFFFF
boardID 0x8188 0xA0F 0xFFFFFFFF
boardID 0x818C 0xA0F 0xFFFFFFFF
```

### 5.6.12 Anti-coincidence between PAIRED channels: (ch0 $\oplus$ ch1), (ch2 $\oplus$ ch3), (ch4 $\oplus$ ch5), (ch6 $\oplus$ ch7), (ch8 $\oplus$ ch9), (ch10 $\oplus$ ch11), (ch12 $\oplus$ ch13), (ch14 $\oplus$ ch15) - VME only

Each couple of channels (up to channel 16 for VME form factors) acquire data in anti-coincidence within 1.6  $\mu$ s of time window:

# Syntax:  
# boardID 0x<address> 0x<value> 0x<mask>

```
boardID 0x1080 0xC0000 0xC0000
boardID 0x1180 0xC0000 0xC0000
boardID 0x1280 0xC0000 0xC0000
boardID 0x1380 0xC0000 0xC0000
boardID 0x1480 0xC0000 0xC0000
boardID 0x1580 0xC0000 0xC0000
boardID 0x1680 0xC0000 0xC0000
boardID 0x1780 0xC0000 0xC0000
boardID 0x1880 0xC0000 0xC0000
boardID 0x1980 0xC0000 0xC0000
boardID 0x1A80 0xC0000 0xC0000
boardID 0x1B80 0xC0000 0xC0000
boardID 0x1C80 0xC0000 0xC0000
boardID 0x1D80 0xC0000 0xC0000
boardID 0x1E80 0xC0000 0xC0000
boardID 0x1F80 0xC0000 0xC0000
boardID 0x1084 0xC8 0xFF
boardID 0x1184 0xC8 0xFF
boardID 0x1284 0xC8 0xFF
boardID 0x1384 0xC8 0xFF
boardID 0x1484 0xC8 0xFF
boardID 0x1584 0xC8 0xFF
boardID 0x1684 0xC8 0xFF
boardID 0x1784 0xC8 0xFF
boardID 0x1884 0xC8 0xFF
boardID 0x1984 0xC8 0xFF
boardID 0x1A84 0xC8 0xFF
boardID 0x1B84 0xC8 0xFF
boardID 0x1C84 0xC8 0xFF
boardID 0x1D84 0xC8 0xFF
boardID 0x1E84 0xC8 0xFF
boardID 0x1F84 0xC8 0xFF
boardID 0x10A0 0x60 0xFF
boardID 0x12A0 0x60 0xFF
boardID 0x14A0 0x60 0xFF
boardID 0x16A0 0x60 0xFF
boardID 0x18A0 0x60 0xFF
boardID 0x1AA0 0x60 0xFF
boardID 0x1CA0 0x60 0xFF
boardID 0x1EA0 0x60 0xFF
```



**Note:** The register 0x8000 should not be written in case of a coincidence inside a couple of channels, as it is performed in the channel FPGA only, with no mother board processing.



**Note:** In case of 725 write: boardID 0x1n84 0x64 0xFF for n=0,...,7 to get 1.6  $\mu$ s of coincidence window.

## 5.7 External Trigger in coincidence with channels for 725-730 series with DPP-PHA

This configuration requires setting an additional register which enables an "Advanced Coincidence" between the trigger signals of individual channels and a signal sent to the TRG-IN input of the board.

- **Advanced Coincidence register (0x1nD8, "n" is referred to the channel number)** : This register allows the configuration of an advanced coincidence between the "*Individual Trigger*" signal and the "*Global Trigger*" signal (please refer to Sec. 2.2 for more details about these signals).

Register name	Address	Register bits	Options
Advanced Coincidence	0x1nD8	[0]	0: Advanced coincidence is disabled 1: Advanced coincidence is enabled.
		[1]	0: Global Trigger used in coincidence 1: Global Trigger used in anti-coincidence
		[2]	0: Individual Trigger used in coincidence 1: Individual Trigger used in anti-coincidence

In the following examples, this register is used to implement a coincidence between the logical AND of the enabled channels (which generates the *Individual Trigger*) and the signal applied to the TRG-IN connector, which can be either a gate or a pulse (treated as the *Global Trigger*). Specifically, when the signal on TRG-IN is at logical level "1" within the channels' coincidence window, data from the channels are acquired.



**Note:** All channels of the board that are not involved in the coincidence configuration will, if not disabled, handle the TRG-IN signal according to the settings of DPP Algorithm Control (register 0x1n80) and DPP Algorithm Control 2 (register 0x1nA0) [RD10].

### 5.7.1 External trigger in coincidence with ch0 and ch1 (ch0 & ch1 & EXT-TRG gate)

The first two channels of the board acquire data when they are in coincidence within a time window and with the signal on TRG-IN connector.

```
boardID 0x811C 0xC00 0xC00
boardID 0x1084 0x1F 0x3FF
boardID 0x1184 0x1F 0x3FF
boardID 0x1080 0x40000 0xC0000
boardID 0x1180 0x40000 0xC0000
boardID 0x10A0 0x40 0xFF
boardID 0x11A0 0x40 0xFF
boardID 0x10D8 0x1 0x7
boardID 0x11D8 0x1 0x7
```



**Note:** Use: boardID 0x1n80 0x1000000 0x1000000, where **n** refers to the number of channel to disable the unused channels.

## 5.7.2 External trigger in coincidence with ch0 and ch2 (ch0 & ch2 & EXT-TRG gate)

Channel 0 and channel 2 acquire data when they are in coincidence within a time window and with the signal on TRG-IN connector.

```
boardID 0x8000 0x4 0x4
boardID 0x811C 0xC00 0xC00
boardID 0x1084 0x1F 0x3FF
boardID 0x1284 0x1F 0x3FF
boardID 0x1080 0x40000 0xC0000
boardID 0x1280 0x40000 0xC0000
boardID 0x10A0 0x55 0xFF
boardID 0x12A0 0x55 0xFF
boardID 0x10D8 0x1 0x7
boardID 0x12D8 0x1 0x7
boardID 0x8180 0x103 0x3FF
boardID 0x8184 0x103 0x3FF
```



**Note:** Use: boardID 0x1n80 0x1000000 0x1000000, where **n** refers to the number of channel to disable the unused channels.

## 5.7.3 External Trigger in coincidence with eight channels (ch0 & ch1 & ch2 & ch3 & ch4 & ch5 & ch6 & ch7)

The first eight channels of the board acquire data when they are in coincidence within a time window and with the signal on TRG-IN connector.

```
boardID 0x8000 0x4 0x4
boardID 0x811C 0xC00 0xC00
boardID 0x1084 0x1F 0x3FF
boardID 0x1184 0x1F 0x3FF
boardID 0x1284 0x1F 0x3FF
boardID 0x1384 0x1F 0x3FF
boardID 0x1484 0x1F 0x3FF
boardID 0x1584 0x1F 0x3FF
boardID 0x1684 0x1F 0x3FF
boardID 0x1784 0x1F 0x3FF
boardID 0x1080 0x40000 0xC0000
boardID 0x1180 0x40000 0xC0000
boardID 0x1280 0x40000 0xC0000
boardID 0x1380 0x40000 0xC0000
boardID 0x1480 0x40000 0xC0000
boardID 0x1580 0x40000 0xC0000
boardID 0x1680 0x40000 0xC0000
boardID 0x1780 0x40000 0xC0000
boardID 0x10A0 0x54 0xFF
boardID 0x12A0 0x54 0xFF
```

```
boardID 0x14A0 0x54 0xFF
boardID 0x16A0 0x54 0xFF
boardID 0x10D8 0x1 0x7
boardID 0x11D8 0x1 0x7
boardID 0x12D8 0x1 0x7
boardID 0x13D8 0x1 0x7
boardID 0x14D8 0x1 0x7
boardID 0x15D8 0x1 0x7
boardID 0x16D8 0x1 0x7
boardID 0x17D8 0x1 0x7
boardID 0x8180 0x1FF 0x3FF
boardID 0x8184 0x1FF 0x3FF
boardID 0x8188 0x1FF 0x3FF
boardID 0x818C 0x1FF 0x3FF
boardID 0x8190 0x1FF 0x3FF
boardID 0x8194 0x1FF 0x3FF
boardID 0x8198 0x1FF 0x3FF
boardID 0x819C 0x1FF 0x3FF
```



**Note:** Use boardID 0x1n80 0x1000000 0x1000000 where **n** refers to the number of channel to disable the unused channels.

## 6 How to Veto the acquisition

In many physics applications it is required to veto the acquisition of one or more channels for a certain amount of time. There are various sources of veto, which can be a combination of the other channels, or an external signal. Consider for example the case of a muon shield, or the case of In the following sections there are reported three case:

1. one channel is vetoed by the other enabled channels;
2. taking advantage of the couple management of 725-730 series, it is possible to set a coincidence logic for the channels in the couple and a veto which is managed by the mother board FPGA;
3. an external signal is used to veto the acquisition.

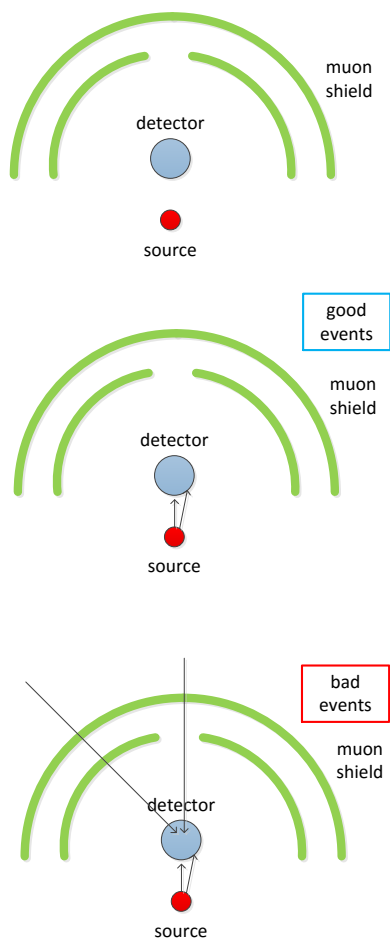
### 6.1 Signal on channel 0 vetoed by the other channels

This is for example the case of a muon shield, where the source is detected by a specific detector (channel 0), and a set of three scintillators (from channel 1 to channel 3) act as a veto (see **Fig. 6.1**). If a muon crosses one of the three scintillators of the shield and the detector, that event is removed.

The logic is  $\overline{CH0} \& (CH1 | CH2 | CH3)$ , i.e. the detector acquire in anti-coincidence with the OR among the three scintillators.

In the following subsection the register settings for each firmware are reported.





**Fig. 6.1:** Muon Shield scheme: an emitting source is detected by a particular detector. Three scintillators act as a muon shield. Events passing both in the shield and in the detector are removed.

### 6.1.1 Veto on ch0 - DPP-PSD for 720 and 751 series

Write the following code in the FreeWrites file. The coincidence window is set to 1  $\mu$ s.

```
# Syntax:
# boardID 0x<address> 0x<data> 0x<mask>

boardID 0x8000 0x4 0x4
boardID 0x1080 0xC0000 0xC0000
boardID 0x1070 0x7D 0x3FF
boardID 0x106C 0x9 0x3FF
boardID 0x8180 0xE 0xFFFFFFFF
```

## 6.1.2 Veto on ch0 - DPP-PSD for 725 and 730 series

Write the following code in the FreeWrites file. The coincidence window is set to 1  $\mu$ s.



**Note:** In case of 725 write: boardID 0x1070 0x3E 0xFF to get about 1  $\mu$ s of coincidence window.

```
# Syntax:
# boardID 0x<address> 0x<data> 0x<mask>

boardID 0x8000 0x4 0x4
boardID 0x1080 0xC0000 0xC0000
boardID 0x1084 0x55 0xFF
boardID 0x1284 0x5 0xFF
boardID 0x1484 0x5 0xFF
boardID 0x1684 0x5 0xFF
boardID 0x1070 0x7D 0x3FF
boardID 0x106C 0x9 0x3FF
boardID 0x8180 0xE 0xFFFFFFFF
```

## 6.1.3 LVDS Veto on all couples - DPP-PSD for 725 and 730 series - VME only

All channel pairs acquire data whenever the LVDS signal on the corresponding pin is in anti-coincidence with the signals from the pairs, within an 800 ns anti-coincidence window. In this configuration, all LVDS pin groups operate in “Trigger” mode: pins [0:7] are configured as inputs, receiving one trigger signal for each pair, while pins [8:15] are configured as outputs, propagating the trigger request externally.



**Note:** In case of 725 write: boardID 0x1n70 0x32 0x3FF to get about 800 ns of coincidence window.

```
boardID 0x8000 0x4 0x4
boardID 0x1080 0xC0000 0xC0000
boardID 0x1180 0xC0000 0xC0000
boardID 0x1280 0xC0000 0xC0000
boardID 0x1380 0xC0000 0xC0000
boardID 0x1480 0xC0000 0xC0000
boardID 0x1580 0xC0000 0xC0000
boardID 0x1680 0xC0000 0xC0000
boardID 0x1780 0xC0000 0xC0000
boardID 0x1880 0xC0000 0xC0000
boardID 0x1980 0xC0000 0xC0000
boardID 0x1A80 0xC0000 0xC0000
boardID 0x1B80 0xC0000 0xC0000
boardID 0x1C80 0xC0000 0xC0000
boardID 0x1D80 0xC0000 0xC0000
boardID 0x1E80 0xC0000 0xC0000
boardID 0x1F80 0xC0000 0xC0000
boardID 0x1070 0x64 0x3FF
boardID 0x1170 0x64 0x3FF
boardID 0x1270 0x64 0x3FF
```

```

boardID 0x1370 0x64 0x3FF
boardID 0x1470 0x64 0x3FF
boardID 0x1570 0x64 0x3FF
boardID 0x1670 0x64 0x3FF
boardID 0x1770 0x64 0x3FF
boardID 0x1870 0x64 0x3FF
boardID 0x1970 0x64 0x3FF
boardID 0x1A70 0x64 0x3FF
boardID 0x1B70 0x64 0x3FF
boardID 0x1C70 0x64 0x3FF
boardID 0x1D70 0x64 0x3FF
boardID 0x1E70 0x64 0x3FF
boardID 0x1F70 0x64 0x3FF
boardID 0x1084 0x50 0xFF
boardID 0x1284 0x50 0xFF
boardID 0x1484 0x50 0xFF
boardID 0x1684 0x50 0xFF
boardID 0x1884 0x50 0xFF
boardID 0x1A84 0x50 0xFF
boardID 0x1C84 0x50 0xFF
boardID 0x1E84 0x50 0xFF
boardID 0x8180 0x20000000 0xFFFFFFFF
boardID 0x8184 0x20000000 0xFFFFFFFF
boardID 0x8188 0x20000000 0xFFFFFFFF
boardID 0x818C 0x20000000 0xFFFFFFFF
boardID 0x8190 0x20000000 0xFFFFFFFF
boardID 0x8194 0x20000000 0xFFFFFFFF
boardID 0x8198 0x20000000 0xFFFFFFFF
boardID 0x819C 0x20000000 0xFFFFFFFF
boardID 0x811C 0x130 0x13C
boardID 0x81A0 0x1111 0xFFFF

```

### 6.1.4 Veto on ch1 by ch0 - DPP-PSD for DT5790

Channel 0 and channel 1 acquire data in anti-coincidence within 800 ns of time window.

```

# Syntax:
# boardID 0x<address> 0x<value> 0x<mask>

boardID 0xEF1C 0x1 0xFF
boardID 0x8000 0x4 0x4
boardID 0x1180 0xC0000 0xC0000
boardID 0x1170 0x64 0x3FF
boardID 0x116C 0x9 0xFF
boardID 0x8188 0x4 0xFFFFFFFF

```

### 6.1.5 Veto on ch0 by ch1 - DPP-PSD for DT5790

Channel 0 and channel 1 acquire data in anti-coincidence within 800 ns of time window.

```

# Syntax:
# boardID 0x<address> 0x<value> 0x<mask>

```

```
boardID 0xEF1C 0x1 0xFF
boardID 0x8000 0x4 0x4
boardID 0x1080 0xC0000 0xC0000
boardID 0x1070 0x64 0x3FF
boardID 0x106C 0x9 0xFF
boardID 0x8188 0x8 0xFFFFFFFF
```

### 6.1.6 Veto on ch0 - DPP-PHA for 724, 781 series and V1782

The coincidence window is set to 1  $\mu$ s.

```
boardID 0x8000 0x4 0x4
boardID 0x1080 0xC0000 0xC0000
boardID 0x1084 0x64 0x3FF
boardID 0x8180 0xE 0xFFFFFFFF
```

### 6.1.7 Veto on all channels by ch0 - DPP-PHA for 724, 781 series and V1782

The coincidence window is set to 640 ns.

```
boardID 0x8000 0x4 0x4
boardID 0x1180 0xC0000 0xC0000
boardID 0x1280 0xC0000 0xC0000
boardID 0x1380 0xC0000 0xC0000
boardID 0x1184 0x40 0x3FF
boardID 0x1284 0x40 0x3FF
boardID 0x1384 0x40 0x3FF
boardID 0x8184 0x101 0xFFFFFFFF
boardID 0x8188 0x101 0xFFFFFFFF
boardID 0x818C 0x101 0xFFFFFFFF
```

### 6.1.8 LVDS Veto as anticoincidence on all channels - DPP-PHA only for V1782

All channels acquire data whenever the LVDS signal on the corresponding pin is in anti-coincidence with the signals from the channel, within 1  $\mu$ s anti-coincidence window. In this configuration, all LVDS pin groups operate in “Trigger” mode: pins [0:7] are configured as inputs, receiving one trigger signal for each channel, while pins [8:15] are configured as outputs, propagating the trigger request externally.

```
boardID 0x8000 0x4 0x4
boardID 0x1080 0xC0000 0xC0000
boardID 0x1180 0xC0000 0xC0000
boardID 0x1280 0xC0000 0xC0000
boardID 0x1380 0xC0000 0xC0000
boardID 0x1480 0xC0000 0xC0000
boardID 0x1580 0xC0000 0xC0000
boardID 0x1680 0xC0000 0xC0000
boardID 0x1780 0xC0000 0xC0000
```

```

boardID 0x1084 0x64 0x3FF
boardID 0x1184 0x64 0x3FF
boardID 0x1284 0x64 0x3FF
boardID 0x1384 0x64 0x3FF
boardID 0x1484 0x64 0x3FF
boardID 0x1584 0x64 0x3FF
boardID 0x1684 0x64 0x3FF
boardID 0x1784 0x64 0x3FF
boardID 0x8180 0x20000000 0xFFFFFFFF
boardID 0x8184 0x20000000 0xFFFFFFFF
boardID 0x8188 0x20000000 0xFFFFFFFF
boardID 0x818C 0x20000000 0xFFFFFFFF
boardID 0x8190 0x20000000 0xFFFFFFFF
boardID 0x8194 0x20000000 0xFFFFFFFF
boardID 0x8198 0x20000000 0xFFFFFFFF
boardID 0x819C 0x20000000 0xFFFFFFFF
boardID 0x811C 0x130 0x13C
boardID 0x81A0 0x1111 0xFFFF

```

### 6.1.9 Veto on ch0 - DPP-PHA for 725 and 730 series

The coincidence window is set to 800 ns.



**Note:** In case of 725 write: boardID 0x1084 0x32 0x3FF to get about 800 ns of coincidence window.

```

boardID 0x8000 0x4 0x4
boardID 0x1080 0xC0000 0xC0000
boardID 0x1084 0x64 0x3FF
boardID 0x106C 0x9 0xFF
boardID 0x10A0 0x55 0xFF
boardID 0x12A0 0x5 0xFF
boardID 0x14A0 0x5 0xFF
boardID 0x16A0 0x5 0xFF
boardID 0x8180 0xE 0xFFFFFFFF

```

### 6.1.10 LVDS Veto as anticoincidence on all channels - DPP-PHA for 725 and 730 series - VME only

All channel pairs acquire data whenever the LVDS signal on the corresponding pin is in anti-coincidence with the signals from the pairs, within an 800 ns anti-coincidence window. In this configuration, all LVDS pin groups operate in "Trigger" mode: pins [0:7] are configured as inputs, receiving one trigger signal for each pair, while pins [8:15] are configured as outputs, propagating the trigger request externally.



**Note:** In case of 725 write: boardID 0x1n84 0x32 0x3FF to get about 800 ns of coincidence window.

```

boardID 0x8000 0x4 0x4
boardID 0x1080 0xC0000 0xC0000
boardID 0x1180 0xC0000 0xC0000
boardID 0x1280 0xC0000 0xC0000

```

```

boardID 0x1380 0xC0000 0xC0000
boardID 0x1480 0xC0000 0xC0000
boardID 0x1580 0xC0000 0xC0000
boardID 0x1680 0xC0000 0xC0000
boardID 0x1780 0xC0000 0xC0000
boardID 0x1880 0xC0000 0xC0000
boardID 0x1980 0xC0000 0xC0000
boardID 0x1A80 0xC0000 0xC0000
boardID 0x1B80 0xC0000 0xC0000
boardID 0x1C80 0xC0000 0xC0000
boardID 0x1D80 0xC0000 0xC0000
boardID 0x1E80 0xC0000 0xC0000
boardID 0x1F80 0xC0000 0xC0000
boardID 0x1084 0x64 0x3FF
boardID 0x1184 0x64 0x3FF
boardID 0x1284 0x64 0x3FF
boardID 0x1384 0x64 0x3FF
boardID 0x1484 0x64 0x3FF
boardID 0x1584 0x64 0x3FF
boardID 0x1684 0x64 0x3FF
boardID 0x1784 0x64 0x3FF
boardID 0x1884 0x64 0x3FF
boardID 0x1984 0x64 0x3FF
boardID 0x1A84 0x64 0x3FF
boardID 0x1B84 0x64 0x3FF
boardID 0x1C84 0x64 0x3FF
boardID 0x1D84 0x64 0x3FF
boardID 0x1E84 0x64 0x3FF
boardID 0x1F84 0x64 0x3FF
boardID 0x10A0 0x50 0xFF
boardID 0x12A0 0x50 0xFF
boardID 0x14A0 0x50 0xFF
boardID 0x16A0 0x50 0xFF
boardID 0x18A0 0x50 0xFF
boardID 0x1AA0 0x50 0xFF
boardID 0x1CA0 0x50 0xFF
boardID 0x1EA0 0x50 0xFF
boardID 0x8180 0x20000000 0xFFFFFFFF
boardID 0x8184 0x20000000 0xFFFFFFFF
boardID 0x8188 0x20000000 0xFFFFFFFF
boardID 0x818C 0x20000000 0xFFFFFFFF
boardID 0x8190 0x20000000 0xFFFFFFFF
boardID 0x8194 0x20000000 0xFFFFFFFF
boardID 0x8198 0x20000000 0xFFFFFFFF
boardID 0x819C 0x20000000 0xFFFFFFFF
boardID 0x811C 0x130 0x13C
boardID 0x81A0 0x1111 0xFFFF

```

## 6.2 Veto for the couples of channels (725-730 only)

In case of 725 and 730 series, we can take advantage of the couple management by setting a coincidence logic for the channels in the couple (mezzanine level), and a veto which is controlled by the mother board FPGA (refer to Sec. 2 for more details on how the coincidence is managed in the digitizer).

The source of this veto is managed by bits[19:18] of register 0x1n84 for DPP-PSD, and bits[15:14] of register 0x1nA0 for DPP-PHA.

Register name	Address	Register bits	Options
DPP Algorithm Control 2	0x1n84-PSD, 0x1nA0-PHA	[19:18] ([15:14])	00 = disabled; 01 = the veto signal is common among all channels. It can be set through register 0x810C, and it can be generated by an external trigger or by a combination of the trigger requests from couples; 10 = the veto signal is individually set for the couple of channels (each couple can have a different veto). It can be set through register 0x8180 (+4n), where n is the couple index, and it can be generated by an external trigger or by a combination of the trigger requests from couples; 11 = the veto signal comes from negative saturation.



**Note:** from revision 136.9 of the DPP-PSD firmware the former bit[7] option has been discontinued. Use option "10" for the same functionality.

The veto duration is defined by register 0x1nD4, where zero means that the veto lasts for the duration of the signal that generated it. A veto width different from 0 extends the veto duration by the amount of time written in the register.

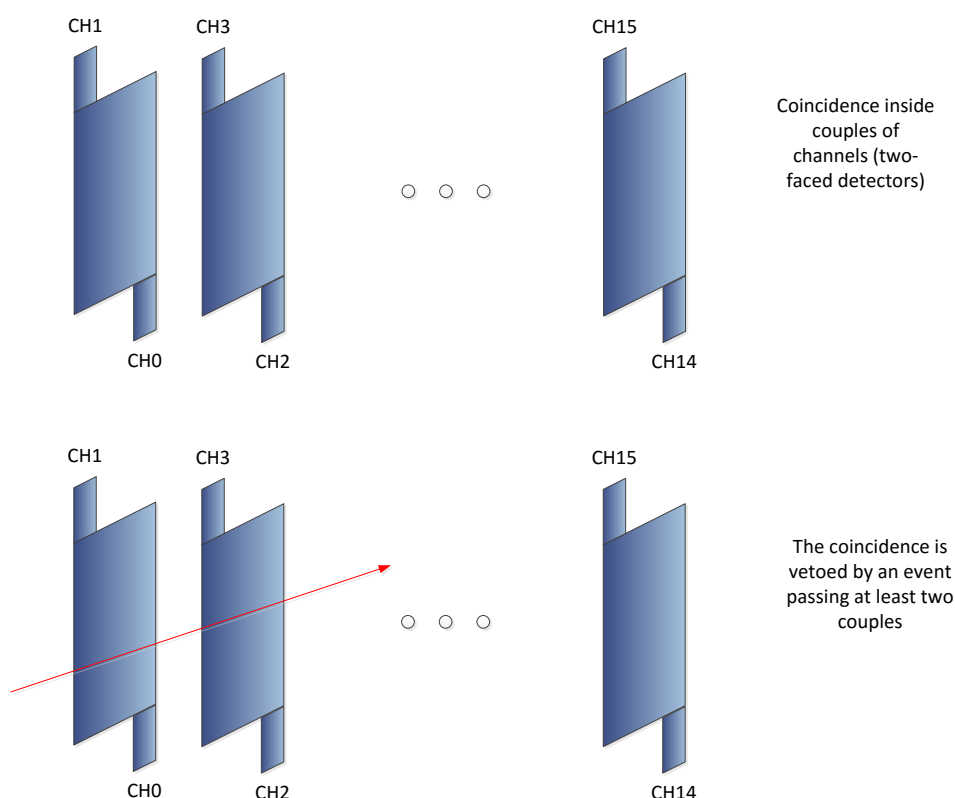
Register name	Address	Register bits	Options
Veto Width	0x1nD4	[15:0]	Value for the veto extension.
		[17:16]	Steps of the veto width. 00: 16 ns for 725, 8 ns for 730; 01: 4 $\mu$ s for 725, 2 $\mu$ s for 730; 10: 1048 $\mu$ s for 725, 524 $\mu$ s for 730; 11: 264 ms for 725, 134 ms for 730.

## 6.2.1 Coincidence inside the VME couples (ch-i & ch-i+1) vetoed by the majority of two or more couples (DPP-PSD)

Couples of channels (up to channel 15 for VME form factor) acquire data within 80 ns of coincidence window. The acquisition is vetoed when at least two couples are in coincidence. For example, this is the case of two-faced detectors which acquire in coincidence. The acquisition is then vetoed by a cosmic ray that crosses two or more couples (see figure below as a scheme).



**Note:** In case of 725 write: boardID 0x1n70 0x5 0x3FF, etc. to get 80 ns of coincidence window.



**Fig. 6.2:** Example of two-sided detectors that acquire in coincidence

# Syntax:  
# boardID 0x<address> 0x<data> 0x<mask>

```
boardID 0x8000 0x4 0x4
boardID 0x1080 0xC0000 0xC0000
boardID 0x1180 0xC0000 0xC0000
boardID 0x1280 0xC0000 0xC0000
boardID 0x1380 0xC0000 0xC0000
boardID 0x1480 0xC0000 0xC0000
boardID 0x1580 0xC0000 0xC0000
boardID 0x1680 0xC0000 0xC0000
boardID 0x1780 0xC0000 0xC0000
boardID 0x1880 0xC0000 0xC0000
```



```

boardID 0x1980 0xC0000 0xC0000
boardID 0x1A80 0xC0000 0xC0000
boardID 0x1B80 0xC0000 0xC0000
boardID 0x1C80 0xC0000 0xC0000
boardID 0x1D80 0xC0000 0xC0000
boardID 0x1E80 0xC0000 0xC0000
boardID 0x1F80 0xC0000 0xC0000
boardID 0x1070 0xA 0x3FF
boardID 0x1170 0xA 0x3FF
boardID 0x1270 0xA 0x3FF
boardID 0x1370 0xA 0x3FF
boardID 0x1470 0xA 0x3FF
boardID 0x1570 0xA 0x3FF
boardID 0x1670 0xA 0x3FF
boardID 0x1770 0xA 0x3FF
boardID 0x1870 0xA 0x3FF
boardID 0x1970 0xA 0x3FF
boardID 0x1A70 0xA 0x3FF
boardID 0x1B70 0xA 0x3FF
boardID 0x1C70 0xA 0x3FF
boardID 0x1D70 0xA 0x3FF
boardID 0x1E70 0xA 0x3FF
boardID 0x1F70 0xA 0x3FF
boardID 0x106C 0x9 0x3FF
boardID 0x116C 0x9 0x3FF
boardID 0x126C 0x9 0x3FF
boardID 0x136C 0x9 0x3FF
boardID 0x146C 0x9 0x3FF
boardID 0x156C 0x9 0x3FF
boardID 0x166C 0x9 0x3FF
boardID 0x176C 0x9 0x3FF
boardID 0x186C 0x9 0x3FF
boardID 0x196C 0x9 0x3FF
boardID 0x1A6C 0x9 0x3FF
boardID 0x1B6C 0x9 0x3FF
boardID 0x1C6C 0x9 0x3FF
boardID 0x1D6C 0x9 0x3FF
boardID 0x1E6C 0x9 0x3FF
boardID 0x1F6C 0x9 0x3FF
boardID 0x1084 0x67 0xFF
boardID 0x1284 0x67 0xFF
boardID 0x1484 0x67 0xFF
boardID 0x1684 0x67 0xFF
boardID 0x1884 0x67 0xFF
boardID 0x1A84 0x67 0xFF
boardID 0x1C84 0x67 0xFF
boardID 0x1E84 0x67 0xFF
boardID 0x8180 0x6FF 0xFFFFFFFF
boardID 0x8184 0x6FF 0xFFFFFFFF
boardID 0x8188 0x6FF 0xFFFFFFFF
boardID 0x818C 0x6FF 0xFFFFFFFF
boardID 0x8190 0x6FF 0xFFFFFFFF
boardID 0x8194 0x6FF 0xFFFFFFFF
boardID 0x8198 0x6FF 0xFFFFFFFF
boardID 0x819C 0x6FF 0xFFFFFFFF

```

## 6.2.2 Coincidence inside the VME couples (ch-i & ch-i+1) vetoed by the majority of two or more couples (DPP-PHA)

Here follows the settings for the same example of Sec. 6.2.1 in case of DPP-PHA firmware. The coincidence window is set equal to 1.6  $\mu$ s.



**Note:** In case of 725 write: boardID 0x1n84 0x64 0x3FF, etc. to get 80 ns of coincidence window.

```
boardID 0x8000 0x4 0x4
boardID 0x1080 0xC0000 0xC0000
boardID 0x1180 0xC0000 0xC0000
boardID 0x1280 0xC0000 0xC0000
boardID 0x1380 0xC0000 0xC0000
boardID 0x1480 0xC0000 0xC0000
boardID 0x1580 0xC0000 0xC0000
boardID 0x1680 0xC0000 0xC0000
boardID 0x1780 0xC0000 0xC0000
boardID 0x1880 0xC0000 0xC0000
boardID 0x1980 0xC0000 0xC0000
boardID 0x1A80 0xC0000 0xC0000
boardID 0x1B80 0xC0000 0xC0000
boardID 0x1C80 0xC0000 0xC0000
boardID 0x1D80 0xC0000 0xC0000
boardID 0x1E80 0xC0000 0xC0000
boardID 0x1F80 0xC0000 0xC0000
boardID 0x1084 0xC8 0xFF
boardID 0x1184 0xC8 0xFF
boardID 0x1284 0xC8 0xFF
boardID 0x1384 0xC8 0xFF
boardID 0x1484 0xC8 0xFF
boardID 0x1584 0xC8 0xFF
boardID 0x1684 0xC8 0xFF
boardID 0x1784 0xC8 0xFF
boardID 0x1884 0xC8 0xFF
boardID 0x1984 0xC8 0xFF
boardID 0x1A84 0xC8 0xFF
boardID 0x1B84 0xC8 0xFF
boardID 0x1C84 0xC8 0xFF
boardID 0x1D84 0xC8 0xFF
boardID 0x1E84 0xC8 0xFF
boardID 0x1F84 0xC8 0xFF
boardID 0x1064 0x67 0xFF
boardID 0x1264 0x67 0xFF
boardID 0x1464 0x67 0xFF
boardID 0x1664 0x67 0xFF
boardID 0x1864 0x67 0xFF
boardID 0x1A64 0x67 0xFF
boardID 0x1C64 0x67 0xFF
boardID 0x1E64 0x67 0xFF
boardID 0x8180 0x6FF 0xFFFFFFFF
boardID 0x8184 0x6FF 0xFFFFFFFF
boardID 0x8188 0x6FF 0xFFFFFFFF
boardID 0x818C 0x6FF 0xFFFFFFFF
boardID 0x8190 0x6FF 0xFFFFFFFF
```

```
boardID 0x8194 0x6FF 0xFFFFFFFF
boardID 0x8198 0x6FF 0xFFFFFFFF
boardID 0x819C 0x6FF 0xFFFFFFFF
```

## 6.3 External Trigger to veto (gate) the acquisition

The acquisition can be synchronized with an external signal, which controls the acquisition according to its logic level high or low. In this case the external signal can be fed into the TRG-IN connector and the TRG\_VAL can be set equal to the external trigger itself. Besides the standard use of the TRG-IN as a global trigger, it is possible to program the digitizer to activate the trigger logic for the whole duration of the external trigger. According to the acquisition mode (coincidence/anti-coincidence) the enabled channels can acquire or being vetoed.

An additional register must be set to accomplish this feature (common both to DPP-PSD and DPP-PHA firmware).

Register name	Address	Register bits	Options
Front Panel I/O Control	0x811C	[0]	0: NIM I/O Level 1: TTL I/O Level
		[10]	0: trigger is synchronized with the edge of the TRG-IN signal; 1: trigger is synchronized with the whole duration of the TRG-IN signal. Note: this bit must be used in conjunction with bit[11] = 0.
		[11]	0: TRG-IN signal is processed by the motherboard and sent to mezzanine (default). The trigger logic is then synchronized with TRG-IN; 1: TRG-IN is directly sent to the mezzanines with no mother board processing nor delay. NOTE: if this bit is set to 1, then bit[10] is ignored.

Bit[11] must be set to 1 to ensure that the TRG-IN signal is sent to the channel FPGA (mezzanine) and the acquisition is synchronized with this signal. Bit[0] adjusts the input level choosing between NIM and TTL.

For example, in case of DPP-PSD firmware write:

```
boardID 0x811C 0x800 0x801 # if NIM
boardID 0x811C 0x801 0x801 # if TTL
```

or, in case of DPP-PHA firmware (TTL level), write:

```
boardID 0x811C 0x801 0x801
```

The TRG\_VAL must be set equal to external TRG-IN; the channel will self-trigger on the input pulse, and then it waits for the validation from the external trigger to save the event.

For example, in case of DPP-PSD firmware, write (for channel 0):

```
boardID 0x8180 0x40000000 0xFFFFFFFF
```

or, in case of DPP-PHA firmware, write:

```
boardID 0x8180 0x40000000 0xFFFFFFFF
```

See the examples in the next section for additional details.

### 6.3.1 Veto from External Trigger - 720 and 751 series with DPP-PSD

Consider an acquisition made by four channels of a DT5720(51) with DPP-PSD firmware, where the acquisition is vetoed by an external TTL signal (on TRG-IN connector). All channels are in anti-coincidence with the external trigger.

```
# Syntax:  
# boardID 0x<address> 0x<data> 0x<mask>
```

```
boardID 0x811C 0x801 0x801  
boardID 0x8000 0x4 0x4  
boardID 0x1080 0xC0000 0xC0000  
boardID 0x1180 0xC0000 0xC0000  
boardID 0x1280 0xC0000 0xC0000  
boardID 0x1380 0xC0000 0xC0000  
boardID 0x8180 0x40000000 0xFFFFFFFF  
boardID 0x8184 0x40000000 0xFFFFFFFF  
boardID 0x8188 0x40000000 0xFFFFFFFF  
boardID 0x818C 0x40000000 0xFFFFFFFF
```

### 6.3.2 Veto from External Trigger - 725 and 730 series with DPP-PSD

Consider an acquisition made by eight channels of a DT5725(30), where the acquisition is vetoed by an external TTL signal (on TRG-IN connector). All channels are in anti-coincidence with the external trigger.

```
# Syntax:  
# boardID 0x<address> 0x<data> 0x<mask>
```

```
boardID 0x811C 0x801 0x801  
boardID 0x8000 0x4 0x4  
boardID 0x1080 0xC0000 0xC0000  
boardID 0x1180 0xC0000 0xC0000  
boardID 0x1280 0xC0000 0xC0000  
boardID 0x1380 0xC0000 0xC0000  
boardID 0x1480 0xC0000 0xC0000  
boardID 0x1580 0xC0000 0xC0000  
boardID 0x1680 0xC0000 0xC0000  
boardID 0x1780 0xC0000 0xC0000  
boardID 0x1084 0x50 0xFF  
boardID 0x1284 0x50 0xFF  
boardID 0x1484 0x50 0xFF  
boardID 0x1684 0x50 0xFF  
boardID 0x8180 0x40000000 0xFFFFFFFF  
boardID 0x8184 0x40000000 0xFFFFFFFF  
boardID 0x8188 0x40000000 0xFFFFFFFF
```

```
boardID 0x818C 0x40000000 0xFFFFFFFF
```

### 6.3.3 Veto from External Trigger - 724, 780, 781 series and V1782 with DPP-PHA

An external trigger on the TRG-IN connector is used to veto the acquisition of four channels. The coincidence window value has not effect in this case, since the acquisition is vetoed for the whole duration of the external trigger itself.



**Note:** In case of 780 series select the first two channels only, which are 0x8188 and 0x818C.

Here the list of registers to be enabled.

```
boardID 0x8000 0x4 0x4
boardID 0x1080 0xC0000 0xC0000
boardID 0x1180 0xC0000 0xC0000
boardID 0x1280 0xC0000 0xC0000
boardID 0x1380 0xC0000 0xC0000
boardID 0x8180 0x40000000 0xFFFFFFFF
boardID 0x8184 0x40000000 0xFFFFFFFF
boardID 0x8188 0x40000000 0xFFFFFFFF
boardID 0x818C 0x40000000 0xFFFFFFFF
boardID 0x811C 0xC01 0xC01
```

### 6.3.4 Veto from External Trigger - 725 and 730 series with DPP-PHA

An external trigger on the TRG-IN connector is used to veto the acquisition of the four channels ch0, ch2, ch4, and ch6.

```
boardID 0x8000 0x4 0x4
boardID 0x1080 0xC0000 0xC0000
boardID 0x1280 0xC0000 0xC0000
boardID 0x1480 0xC0000 0xC0000
boardID 0x1680 0xC0000 0xC0000
boardID 0x10A0 0x50 0xFF
boardID 0x12A0 0x50 0xFF
boardID 0x14A0 0x50 0xFF
boardID 0x16A0 0x50 0xFF
boardID 0x8180 0x40000000 0xFFFFFFFF
boardID 0x8184 0x40000000 0xFFFFFFFF
boardID 0x8188 0x40000000 0xFFFFFFFF
boardID 0x818C 0x40000000 0xFFFFFFFF
boardID 0x811C 0xC01 0xC01
```

## 7 Miscellaneous

In this chapter some miscellaneous configuration are described.

### 7.1 Examples for 720 - 751 series, and DT5790 - DPP-PSD

#### 7.1.1 PSD cut value Online on ch0

Set PSD cut value Online to 0.5 on ch0. Events below the threshold are rejected.

```
# Syntax:
# boardID 0x<address> 0x<value> 0x<mask>

boardID 0x1078 0x200 0x3FF
boardID 0x1080 0x8000000 0x8000000
```

### 7.2 Examples for 725 - 730 series - DPP-PSD

#### 7.2.1 Global Trigger to all Channels by signal on ch0

Whenever an input signal is triggered by channel 0 within a window of 80 ns, a global trigger from mother board enables all channels to read and save the event.

```
# Syntax:
# boardID 0x<address> 0x<value> 0x<mask>

boardID 0x1080 0x1000000 0x1000000
boardID 0x1180 0x1000000 0x1000000
boardID 0x1280 0x1000000 0x1000000
boardID 0x1380 0x1000000 0x1000000
boardID 0x1480 0x1000000 0x1000000
boardID 0x1580 0x1000000 0x1000000
boardID 0x1680 0x1000000 0x1000000
boardID 0x1780 0x1000000 0x1000000
boardID 0x1084 0x5 0x7
boardID 0x1070 0xA 0x3FF
boardID 0x810C 0x1 0x1
```



**Note:** In case of 725 write: boardID 0x1070 0x5 0x3FF, etc. to get 80 ns of coincidence window.

## 7.2.2 Global Trigger as OR of the channels (Not Using the Global Trigger Mask)

Whenever an input signal is triggered by any channel, a global trigger from mother board enables all channels to read and save the events (without using the Global Trigger Mask).

```
# Syntax:
# boardID 0x<address> 0x<value> 0x<mask>

boardID 0x1080 0x01000000 0x01000000
boardID 0x1180 0x01000000 0x01000000
boardID 0x1280 0x01000000 0x01000000
boardID 0x1380 0x01000000 0x01000000
boardID 0x1480 0x01000000 0x01000000
boardID 0x1580 0x01000000 0x01000000
boardID 0x1680 0x01000000 0x01000000
boardID 0x1780 0x01000000 0x01000000
boardID 0x1084 0x57 0xFF
boardID 0x1284 0x57 0xFF
boardID 0x1484 0x57 0xFF
boardID 0x1684 0x57 0xFF
boardID 0x8180 0xFF 0xFFFFFFFF
boardID 0x8184 0xFF 0xFFFFFFFF
boardID 0x8188 0xFF 0xFFFFFFFF
boardID 0x818C 0xFF 0xFFFFFFFF
```

## 7.2.3 External Trigger makes ch0 and ch1 triggering

Whenever an external trigger is detected, a global trigger from mother board enables channel 0 and channel 1 to read and save the event.

```
# Syntax:
# boardID 0x<address> 0x<value> 0x<mask>

boardID 0x1080 0x10000000 0x10000000
boardID 0x1180 0x10000000 0x10000000
boardID 0x1084 0x50 0xFF
boardID 0x8180 0x40000000 0x40000000
boardID 0x810C 0x0 0x40000000
```

## 7.2.4 External Trigger from TRG-IN to send a Global Trigger to all Channels

Whenever an external trigger is detected, a global trigger from mother board enables all channels to read and save the event.

```
# Syntax:
# boardID 0x<address> 0x<value> 0x<mask>

boardID 0x1080 0x01000000 0x010C0000
```

```
boardID 0x1180 0x01000000 0x010C0000
boardID 0x1280 0x01000000 0x010C0000
boardID 0x1380 0x01000000 0x010C0000
boardID 0x1480 0x01000000 0x010C0000
boardID 0x1580 0x01000000 0x010C0000
boardID 0x1680 0x01000000 0x010C0000
boardID 0x1780 0x01000000 0x010C0000
boardID 0x1880 0x01000000 0x010C0000
boardID 0x1980 0x01000000 0x010C0000
boardID 0x1A80 0x01000000 0x010C0000
boardID 0x1B80 0x01000000 0x010C0000
boardID 0x1C80 0x01000000 0x010C0000
boardID 0x1D80 0x01000000 0x010C0000
boardID 0x1E80 0x01000000 0x010C0000
boardID 0x1F80 0x01000000 0x010C0000
boardID 0x810C 0x40000000 0x40000000
```

## 7.2.5 PSD cut value Online on ch0 and ch1

Set PSD cut value Online to 0.7 on both channel 0 and channel 1. It enables PSD cut below threshold (to cut on gammas) on ch0 and PSD cut above threshold (to cut on neutrons) on ch1.

```
# Syntax:
# boardID 0x<address> 0x<value> 0x<mask>

boardID 0x1078 0x2CD 0x3FF
boardID 0x1178 0x2CD 0x3FF
boardID 0x1080 0x80000000 0x18000000
boardID 0x1180 0x10000000 0x18000000
```

## 7.2.6 Trigger Propagation to TRG\_OUT (from Trigger Request in ch0)

Enable the Trigger propagation to **TRG\_OUT** from trigger request on channel 0.

```
# Syntax:
# boardID 0x<address> 0x<value> 0x<mask>

boardID 0x1084 0x5 0xFF
boardID 0x8110 0x1 0x1
boardID 0x811C 0x0 0x3C000
```

## 7.2.7 Trigger Propagation to TRG\_OUT LEMO and LVDS - VME only

Enable the Trigger propagation to **TRG\_OUT** and to LVDS, groups [3:0] and [7:4].

```
# Syntax:
# boardID 0x<address> 0x<value> 0x<mask>

boardID 0x1084 0x7 0xFF
```



```
boardID 0x1284 0x7 0xFF  
boardID 0x1484 0x7 0xFF  
boardID 0x1684 0x7 0xFF  
boardID 0x1884 0x7 0xFF  
boardID 0x1A84 0x7 0xFF  
boardID 0x1C84 0x7 0xFF  
boardID 0x1E84 0x7 0xFF  
boardID 0x8110 0xFF 0xFF  
boardID 0x811C 0x10C 0x3C10F  
boardID 0x81A0 0x11 0xFFFF
```

## 7.3 Examples for 724, 780, 781 and V1782 series - DPP-PHA

### 7.3.1 Propagate out the trigger requests of the OR of the channels

The TRG-OUT output is configured to propagate out the trigger requests from the OR of the enabled channels.

```
# Syntax:
# boardID 0x<address> 0x<value> 0x<mask>

boardID 0x811C 0x0 0xFFFFFFFF
boardID 0x8110 0xF 0xFFFFFFFF
```



**Note:** In case of 780 or 724 and 781 with DT/NIM form factor, write: boardID 0x8110 0x3 0xFFFFFFFF.

### 7.3.2 Global Trigger by any channel

Whenever an input signal is triggered by any channel, a global trigger from mother board enables all channels to read and save the events.

```
boardID 0x8000 0x4 0x4
boardID 0x1080 0x40000 0xC0000
boardID 0x1180 0x40000 0xC0000
boardID 0x1280 0x40000 0xC0000
boardID 0x1380 0x40000 0xC0000
boardID 0x1480 0x40000 0xC0000
boardID 0x1580 0x40000 0xC0000
boardID 0x1680 0x40000 0xC0000
boardID 0x1780 0x40000 0xC0000
boardID 0x810C 0xFF 0xFF
```

### 7.3.3 Global Trigger by ch0

Whenever an input signal is triggered by channel 0, a global trigger from mother board enables all channels to read and save the events.

```
boardID 0x8000 0x4 0x4
boardID 0x1080 0x40000 0xC0000
boardID 0x1180 0x1000000 0x1000000
boardID 0x1280 0x1000000 0x1000000
boardID 0x1380 0x1000000 0x1000000
boardID 0x1480 0x1000000 0x1000000
boardID 0x1580 0x1000000 0x1000000
boardID 0x1680 0x1000000 0x1000000
boardID 0x1780 0x1000000 0x1000000
```

```
boardID 0x810C 0x1 0xFF
```

## 7.4 Examples for 725 and 730 series - DPP-PHA

### 7.4.1 LVDS propagate out the trigger requests - VME only

The LVDS pins [7:0] are set in "Trigger" mode, the pins [8:15] are set in "Register" mode. The pins [7:0] are set in direction OUTPUT, so they propagate out the trigger request. The pins [8:15] are set in direction INPUT. LVDS New features are also configured.

```
# Syntax:
# boardID 0x<address> 0x<value> 0x<mask>

boardID 0x10A0 0x7 0xFF
boardID 0x12A0 0x7 0xFF
boardID 0x14A0 0x7 0xFF
boardID 0x16A0 0x7 0xFF
boardID 0x18A0 0x7 0xFF
boardID 0x1AA0 0x7 0xFF
boardID 0x1CA0 0x7 0xFF
boardID 0x1EA0 0x7 0xFF
boardID 0x811C 0x10C 0x13C
boardID 0x81A0 0x11 0xFFFF
```

### 7.4.2 Propagate out the trigger requests of the OR of the channels inside the couples

The TRG-OUT output is configured to propagate out the trigger requests from the OR of the channels inside the first four couples.

```
# Syntax:
# boardID 0x<address> 0x<value> 0x<mask>

boardID 0x10A0 0x7 0xFF
boardID 0x12A0 0x7 0xFF
boardID 0x14A0 0x7 0xFF
boardID 0x16A0 0x7 0xFF
boardID 0x811C 0x0 0xFFFFFFFF
boardID 0x8110 0xF 0xFFFFFFFF
```

### 7.4.3 External Trigger from TRG-IN to send a Global Trigger to all Channels

Whenever an external trigger is detected, a global trigger from mother board enables all channels to read and save the event.

```
# Syntax:
# boardID 0x<address> 0x<value> 0x<mask>
```

```
boardID 0x1080 0x01000000 0x010C0000
boardID 0x1180 0x01000000 0x010C0000
boardID 0x1280 0x01000000 0x010C0000
boardID 0x1380 0x01000000 0x010C0000
boardID 0x1480 0x01000000 0x010C0000
boardID 0x1580 0x01000000 0x010C0000
boardID 0x1680 0x01000000 0x010C0000
boardID 0x1780 0x01000000 0x010C0000
boardID 0x1880 0x01000000 0x010C0000
boardID 0x1980 0x01000000 0x010C0000
boardID 0x1A80 0x01000000 0x010C0000
boardID 0x1B80 0x01000000 0x010C0000
boardID 0x1C80 0x01000000 0x010C0000
boardID 0x1D80 0x01000000 0x010C0000
boardID 0x1E80 0x01000000 0x010C0000
boardID 0x1F80 0x01000000 0x010C0000
boardID 0x810C 0x40000000 0x40000000
```

## 8 Coincidences with DPP-PSD and DPP-PHA firmwares in digitizers of the Second generation

This Chapter is intended to describe how to make coincidences with digitizers 2.0 running DPP-PSD and DPP-PHA firmware, as listed in **Tab. 1.1**. The firmware registers involved in the coincidence are described in detail, as well as how to modify the software. In this way those who write their own software, and those who use the CAEN software CoPASS [RD4] can perform on-line onboard coincidences.

CAEN Waveform digitizers x27xx introduce a new approach in the access to the firmware parameters with respect to the previous generation V17xx, DT57xx and N67xx. Indeed, while in the previous generation x17xx the approach was based on the direct exposition of the single firmware register, the new generation of digitizer provides an abstraction of the register in the form of library parameters much easier to understand and use.

To configure the coincidence settings it is required to use specific values of the CAENFELib library parameters. Parameters refer to entities that serve as arguments for the *CAEN\_FELib\_SetValue()* and *CAEN\_FELib\_GetValue()* functions.

The parameters are organized based on functionalities, with each chapter detailing a particular aspect of the digitizer and the corresponding parameters necessary for its implementation. These parameters are described using the following structure:

Parameter name	Description
Description	<i>Description text</i>
Level	Indicates the path needed to set/get the parameter value. The table of the allowed levels and their corresponding paths is shown below
Type	Indicates the parameter type: could be LIST, RANGE. N.B. Type doesn't correspond to any C-type, all CAEN FELib parameters are float or string type
Access Type	Indicates if the parameter has specific access type (e.g. Read Only or Write Only). If not specified the parameter is Read/Write
UoM	Indicates the Unit of Measurement of the parameter (if any)
Min	Indicates the allowed minimum value (if any)
Max	Indicates the allowed maximum value (if any)
Increment	Indicates the allowed increment value (if any)
Allowed Values	<i>List-Type</i> parameters allow only the indicated values as options

Level allowed values:

Level	Path
DIG	/par/<ParameterName>
GROUP	/group/<idx>/par/<ParameterName>
CH	/ch/<idx>/par/<ParameterName>
LVDS	/lvds/<idx>/par/<ParameterName>
VGA	/vga/<index>/par/<ParameterName>

## 8.1 How to configure the parameters - 2730, 2740, 2745 and 2751 series with DPP-PSD and DPP-PHA

The main parameters involved in coincidence and anticoincidence mode among channels, as well as in the configuration of an external signal as Veto or Gate, are briefly described in the following list. For more detailed information, please refer to the CAEN FELib User Manual, for the specific digitizer and firmware used [RD11] [RD12] [RD13] [RD14] [RD15].

- **Channels Trigger Mask:** Allows to set the mask over 64 bits (32 or 16 bits depending on the board model) to generate a channel trigger. It can be used to trigger a channel using a trigger coming from another channel. It also allows to set the mask over 64 bits to enable the channel to participate in the coincidence logic defined in CoincidenceMask and AntiCoincidenceMask (option "Channel64Trg").
- **Coincidence/Anticoincidence Mask:** Allows to set the coincidence and the anticoincidence mask respectively, that generates a trigger on the specified channel.

Allowed values	Description
Disabled	All the coincidence/anticoincidence sources are disabled
Ch64Trigger	One of the 64 channels can generate a coincidence/anticoincidence signal
TRGIN	TRGIN can generate a coincidence/anticoincidence signal
GlobalTriggerSource	Acquisition Trigger can generate a coincidence/anticoincidence signal
ITLA	ITLA can generate a coincidence/anticoincidence signal
ITLB	ITLB can generate a coincidence/anticoincidence signal

- **Coincidence Length S/T:** Coincidence window length in samples (S) or nanoseconds (ns), respectively.
- **ITLA/B Main Logic:** allows to combine all the self-triggers of the board, according to a specific trigger logic. There are two independent logic blocks, ITLA and ITLB. Their output can be used separately to feed, for instance, AcqTrigger and TrgOut, or combined in a second level trigger logic to implement more complex trigger schemes. Therefore, the ITLs can either generate the local acquisition trigger, common to all the channels, for the acquisition of the waveform, or propagate the signal outside, through the TRGOUT, thus making it possible to combine triggers of multiple boards in an external trigger logic, that eventually feeds back the TRGIN of the digitizers. Each ITL is made of an input enable mask (64 bits, one per channel), an optional pairing logic that combines the self triggers of two consecutive channels (e.g. paired coincidence) and the main trigger logic that combines the 64 selftriggers with an OR, AND or Majority logic (Fig. 10.3). The output can be linear (no stretching) or reshaped by a programmable gate generator, either re-triggerable or not and finally programmed for polarity (direct or inverted).

Allowed values	Description
OR	ITLOUT = masked OR of channel self-triggers
AND	ITLOUT = masked AND of channel self-triggers
Majority	ITLOUT = masked Majority of channel self-triggers

- **ITLA/B Pair Logic:** Pairs of channels can be combined with an OR or AND before feeding in the Main trigger Logic. The coincidence window will be set by the *SelfTriggerWidth* parameter. When the AND/OR logic is applied, the two outputs of the Pair Logic blocks are identical.



**Note:** These signals are counted twice in the following Majority logic. If the Pair Logic is disabled ("NONE" option), the block is transparent, and the two outputs are just a replica of the inputs.

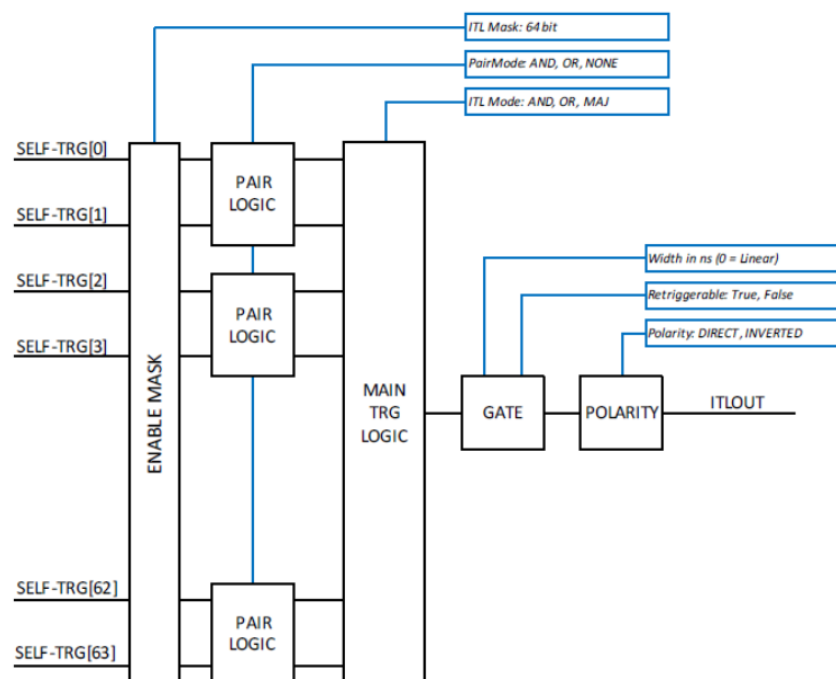


Fig. 8.1: Individual Trigger logic scheme.

Allowed values	Description
OR	Both Pair Logic Outputs = OR of two consecutive self-triggers
AND	Both Pair Logic Outputs = AND of two consecutive self-triggers
NONE	Outputs = Inputs

- **ITLA/B Majority Lev:** Defines the majority level of the Main Logic of the ITL A or B block, respectively. The majority output is calculated at every clock cycle, and it becomes TRUE when  $Nch \geq MajLev$ , where  $Nch$  is the number of self-triggers active in that clock cycle and  $MajLev$  is the programmed majority level.



**Note:** When the Pair Logic is used to combine the self triggers two by two (AND/OR), each pair produces two identical signals that will be counted twice in the majority level.

- **ITLA/B Polarity:** Polarity of the gate generator output. Options are "Direct" or "Inverted".
- **ITLA/B Mask:** Enable mask (64 bits) as the inputs of the ITL A blocks. Each bit selects the corresponding channel whose self triggers participate in the block logic. E.g. 0x3 (channel 0 and channel 1 participate in the trigger logic).
- **ITLA/B Gate Width:** Width of the gate generator at the output of the ITLA or ITLB block, respectively. It is expressed in nanoseconds (ns) with a minimum step of 8 ns.
- **Global Trigger Source:** Defines the source for the Global Trigger, which is the signal that saves the events in the memory buffers. Multiple options are allowed, separated by "|".

Allowed values	Description
TrgIn	Front Panel TRGIN
SwTrg	Software trigger
LVDS	LVDS trgin
ITLA	Internal Trigger Logic A: combination of channel self-triggers
ITLB	Internal Trigger Logic B: combination of channel self-triggers
ITLA_AND_ITLB	Second level Trigger logic making the AND of ITL A and B
ITLA_OR_ITLB	Second level Trigger logic making the OR of ITL A and B
GPIO	Front Panel GPIO
TestPulse	Internal Test Pulse
UserTrg	User custom trigger source

- **Event Trigger Source:** Allows to set the trigger source for a Time-Energy (T-E) event. Setting this parameter means to get an event including time stamp and energy information.

Allowed values	Description
Disabled	No trigger source enabled for the T-E event
Ch64Trigger	One (or more) channel self-trigger can generate a trigger for a T-E event
ChSelfTrigger	Channel self-trigger can generate a trigger for a T-E event
SwTrg	Software Trigger can generate a trigger for a T-E event
TRGIN	External TRGIN can generate a trigger for a T-E event
GlobalTriggerSource	Acquisition Trigger Source (the same of the Scope mode) can generate a trigger for a T-E event
LVDS	A signal on the LVDS connectors can generate a trigger for a T-E event
ITLA	Internal Trigger Logic A can generate a trigger for a T-E event
ITLB	Internal Trigger Logic B can generate a trigger for a T-E event

- **Wave Trigger Source:** Allows to set the trigger source for the waveform. Setting this parameter means to get an event including the waveform and the associated time stamp and energy information.

Allowed values	Description
Disabled	No trigger source enabled for the waveform
Ch64Trigger	One (or more) channel self-trigger can generate a trigger for a waveform
ChSelfTrigger	Channel self-trigger can generate a trigger for a waveform
SwTrg	Software Trigger can generate a trigger for a waveform
ADCOversaturation	ADC Oversaturation can generate a trigger for a waveform
ADCUnderSaturation	ADC Undersaturation can generate a trigger for a waveform
ExternalInhibit	Inhibit can generate a trigger for a waveform
TRGIN	External TRGIN can generate a trigger for a waveform
GlobalTriggerSource	Acquisition Trigger Source (the same of the Scope mode) can generate a trigger for a waveform
LVDS	A signal on the LVDS connectors can generate a trigger for a waveform
ITLA	Internal Trigger Logic A can generate a trigger for a waveform
ITLB	Internal Trigger Logic B can generate a trigger for a waveform

- **Self Trigger Width:** The output of the digital leading-edge comparator, that generates the self-trigger signal, can be used in "linear" mode, meaning that it lasts for the time the signal remains above (or below) the threshold, thus acting as an "Over-Threshold" signal, or can pass through a programmable



gate generator that makes it a fixed-width pulse. The gate generator is a non-retriggerable monostable that goes high when the threshold is crossed and returns low after the programmed time. This parameter defines the fixed width of the overthreshold pulse. It is expressed in nanoseconds (ns) with a minimum step of 8 ns.

- **Channel Veto Source:** Allows to set the veto for each channel; it can be external (which means one of the veto options in the previous table), or it can be on a channel base.

Allowed values	Description
Disabled	Any channel veto source is disabled
BoardVeto	Enables board veto
ADCOversaturation	Enables veto due to ADC oversaturation
ADCUnderSaturation	Enables veto due to ADC undersaturation

- **Board Veto Source:** Defines the source for the Veto, which is the signal that inhibits the acquisition trigger. Multiple options are allowed, separated by "|". The VETO signal can be either active high or low, depending on the BoardVetoPolarity parameter. When active low, it acts as a GATE for the trigger. It is possible to stretch the duration of the VETO by means of the parameter.

Allowed values	Description
Disabled	VETO is always OFF
SIN	SIN on the front panel
GPIO	GPIO on the front panel (used as input)
LVDS	LVDS trgin (see LVDSMode)
EncodedClkin	Encoded CLK-IN veto (not implemented)

- **Board Veto Width:** Whatever is the source of the VETO signal, it is possible to stretch the duration of the veto up to a given time by means of a re-triggerable monostable. When 0, the monostable is disabled and the veto lasts as long as the selected source is active. It is expressed in nanoseconds (ns) with a minimum step of 8 ns.
- **Board Veto Polarity:** Defines the polarity of the Veto.

Allowed values	Description
ActiveHigh	Veto is active high. The signals acts as an "Inhibit" for the trigger
ActiveLow	Veto is active low. The signals acts as a "Gate" the trigger

## 8.2 Examples for 2730, 2740, 2745 series with DPP-PSD and DPP-PHA and for 2751 series with DPP-PSD

### 8.2.1 Coincidence between channel 0 and channel 1 (ch0 & ch1) - using "Ch64Trigger"

Channel 0 and channel 1 acquire data within 40 ns of coincidence window:

```
# Syntax:
# boardID <Level> <Allowed Values>

boardID /ch/0..1/par/CoincidenceMask Ch64Trigger
boardID /ch/0/par/ChannelsTriggerMask 0x2
boardID /ch/1/par/ChannelsTriggerMask 0x1
boardID /ch/0/par/CoincidenceLengthT 40
boardID /ch/1/par/CoincidenceLengthT 40
```

### 8.2.2 Coincidence between channel 0 and any other channel (ch0 & ch1), (ch0 & ch2), ..., (ch0 & ch63) - using "Ch64Trigger"

Channel 0 and any other channel acquire data within 40 ns of coincidence window:

```
# Syntax:
# boardID <Level> <Allowed Values>

boardID /ch/0..63/par/CoincidenceMask Ch64Trigger
boardID /ch/0/par/ChannelsTriggerMask 0xFFFFFFFFFFFFFFFE
boardID /ch/1..63/par/ChannelsTriggerMask 0x1
boardID /ch/0..63/par/CoincidenceLengthT 40
```

### 8.2.3 Majority among all channels to send a Global Trigger to the whole board

All channels of the board acquire data when at list **N** channels are in coincidence within 100 ns of coincidence window.

```
# Syntax:
# boardID <Level> <Allowed Values>

boardID /ch/0..63/par/EventTriggerSource GlobalTriggerSource
boardID /par/GlobalTriggerSource ITLA
boardID /par/ITLAMask 0xFFFFFFFFFFFFFFF
boardID /par/ITLAPairLogic NONE
boardID /par/ITLAMainLogic Majority
boardID /par/ITLAMajorityLev N
boardID /ch/0..63/par/selftriggerwidth 100
```



**Note:** Replace **N** with the desired majority level. Note that setting the value to N means that at least N signals must be in coincidence in order to acquire data from all channels.



**Note:** To set the WaveTriggerSource, replace the first line with the following: boardID /ch/0..63/par/WaveTriggerSource GlobalTriggerSource.

## 8.2.4 External Trigger from TRG-IN to send a Global Trigger to all channels

Whenever an external trigger is detected, a global trigger enables all channels to read and save event.

```
# Syntax:
# boardID <Level> <Allowed Values>

boardID /ch/0..63/par/EventTriggerSource GlobalTriggerSource
boardID /par/GlobalTriggerSource TRGIN
```



**Note:** To set the WaveTriggerSource, replace the first line with the following: boardID /ch/0..63/par/WaveTriggerSource GlobalTriggerSource.

## 8.2.5 External Trigger as Gate for all channels

All channels of the board acquire data within a window defined by an external trigger.

```
# Syntax:
# boardID <Level> <Allowed Values>

boardID /ch/0..63/par/ChannelVetoSource BoardVeto
boardID /par/BoardVetoSource GPIO
boardID /par/BoardVetoPolarity ActiveLow
```



**Note:** To send the External Trigger through the S-IN input, set the parameter BoardVetoSource as follows: boardID /par/BoardVetoSource SIN.



**Note:** To use the External Trigger to start acquisition and set a specific gate width, add the following line: boardID /par/BoardVetoWidth LENGHT, where "LENGHT" must be replaced with the desired value in nanoseconds.

## 8.2.6 External Trigger to Veto all channels

All channels are in anti-coincidence with the external trigger.

```
# Syntax:
# boardID <Level> <Allowed Values>

boardID /ch/0..63/par/ChannelVetoSource BoardVeto
boardID /par/BoardVetoSource GPIO
boardID /par/BoardVetoPolarity ActiveHigh
```



**Note:** To send the External Trigger through the S-IN input, set the parameter BoardVetoSource as follows:  
`boardID /par/BoardVetoSource SIN.`



**Note:** To use the External Trigger to start acquisition and set a specific gate width, add the following line:  
`boardID /par/BoardVetoWidth LENGTH`, where "LENGTH" must be replaced with the desired value in nanoseconds.

## 9 Coincidences with Waveform Recording Firmware in Digitizers of the First Generation

Data acquisition with WR firmware ensures that whenever an input signal is triggered by one channel, a global trigger from mother board enables all channels to read and save the event.

The global trigger can be also configured to make the coincidence logic through the majority operation within a desired time window. This can be done with a firmware register write for most of the digitizer series, with few exceptions:

- In case of *725 and 730 series* it is also possible to perform the logic AND/OR of channels of the same couple, as well as a global trigger of couples (majority only). Refer to [RD25], [RD26], [RD27] for more details.
- The *742 series* does not have the majority logic.
- In case of *743 series* it is possible to perform the logic AND/OR between channels of the same couple, and logic AND/OR/MAJORITY among different couples. For more details refer to [RD28].
- In case of *740 series* the coincidence is meant among different groups.

The CAEN software *WaveDump*, available at the CAEN web site, can be used to test the majority logic.

### 9.1 How to configure the registers - 720, 724, 740 and 751 series with WR

This Chapter is intended to describe how to make coincidences with WR firmware acting on the firmware registers through a Freewrites file. The firmware registers involved in the coincidence are described in detail, as well as how to modify the software. In this way those who write their own software, and those who use the CAEN software *WaveDump* [RD29] can perform on-line onboard coincidences.

To configure the coincidence settings it is required to write specific values of the firmware registers [RD30], [RD31], [RD32]. In particular the syntax adopted in this section and in the next ones is the one presented in the following:

```
WRITE_REGISTER <address> <value> <mask>
```

Those who write their own software code, must take care of the proper register write mask, i.e. the bit mask to be written.



All values MUST be written in *hexadecimal*.

Modify the 32 bit "Trigger Source Enable Mask" register (address 0x810C) through the `WRITE_REGISTER` string, where bits [7:0] enable the specific channels for acquisition, bits [23:20] set the coincidence window width in steps of trigger clock (i.e. 10 ns for 724 series and 8 ns for the others), and bits [26:24] set the majority level. A majority level  $m$  enables a global trigger when at least  $m + 1$  channels trigger within the coincidence window. Bits [31:30] control the external trigger and the software trigger respectively.

For example, the coincidence between two channels can be achieved by setting the majority level equal to one. In case you want to have the coincidence of channel 1 and channel 3, with a majority level of 1, and a coincidence window of 10 trigger clocks, you must write in the WaveDumpConfig.txt file:

```
WRITE_REGISTER 810C 1A0000A 7F000FF
```

The following table summarizes the 0x810C bit register description.

Register name	Address	Register bits	Options
Global Trigger Mask	0x810C	[0]	0 TRG_REQ[0] not propagated 1 TRG_REQ[0] propagated
		[1]	0 TRG_REQ[1] not propagated 1 TRG_REQ[1] propagated
		[2]	0 TRG_REQ[2] not propagated 1 TRG_REQ[2] propagated
		[3]	0 TRG_REQ[3] not propagated 1 TRG_REQ[3] propagated
		[4]	0 TRG_REQ[4] not propagated 1 TRG_REQ[4] propagated
		[5]	0 TRG_REQ[5] not propagated 1 TRG_REQ[5] propagated
		[6]	0 TRG_REQ[6] not propagated 1 TRG_REQ[6] propagated
		[7]	0 TRG_REQ[7] not propagated 1 TRG_REQ[7] propagated
		[19:8]	RESERVED
		[23:20]	Coincidence window
		Majority Level [26:24]	Value for majority
		TRG-IN used as GATE [27]	0 TRG-IN in logic OR with the enabled channels and SW trigger (default) 1 TRG-IN in logic AND the enabled channels.
		[28]	0 (RESERVED)
		LVDS External Trigger [29]	0 LVDS EXT TRG not propagated 1 LVDS EXT TRG propagated
		External Trigger [30]	0 EXT TRG not propagated 1 EXT TRG propagated
		Software Trigger [31]	0 SOFT TRG not propagated 1 SOFT TRG propagated



In the case of the x740 digitizer family, the first bits [7:0] refer to the trigger request from group **n**, rather than from individual channels.

## 9.2 Examples for 720, 724, 740, 751 series with VR

### 9.2.1 Coincidence between channel 0 and channel 1 (ch0 & ch1)

All channels acquire data when channel 0 and channel 1 are in coincidence within an 80 ns time window. It is achieved setting the majority level equal to one.

```
WRITE_REGISTER 810C 1A00003 7F000FF
```



**Note:** In case of 724 write: `WRITE_REGISTER 810C 1800003 7F000FF` to get 80 ns of coincidence window.



**Note:** In case of 740 the coincidence is intended to be between the group 0 and the group 1.

### 9.2.2 Coincidence among channel 0, channel 1 and channel 2 (ch0 & ch1 & ch2)

All channels acquire data when channel 0, channel 1 and channel 2 are in coincidence within an 80 ns time window. It is achieved setting the majority level equal to two.

```
WRITE_REGISTER 810C 2A00007 7F000FF
```



**Note:** In case of 724 write: `WRITE_REGISTER 810C 2800007 7F000FF` to get 80 ns of coincidence window.



**Note:** In case of 740 the coincidence is intended to be among the group 0, group 1 and group 2.

### 9.2.3 Coincidence among eight channels (ch0 & ch1 & ch2 & ch3 & ch4 & ch5 & ch6 & ch7) - VME only

All channels acquire data when they are all in coincidence within an 80 ns time window. It is achieved setting the majority level equal to seven.

```
WRITE_REGISTER 810C 7A000FF 7F000FF
```



**Note:** In case of 724 write: `WRITE_REGISTER 810C 78000FF 7F000FF` to get 80 ns of coincidence window.



**Note:** In case of 740 the coincidence is intended to be among all groups (64 channels).

## 9.3 How to configure the registers - 725, 730 series with WR

As mentioned above, in the case of 725 and 730 series, it is also possible to perform logic AND/OR operations between channels of the same couple setting the register 0x1n84. Additionally, as well as a global trigger among couples (majority only). For more details, please refer to the 725-730 series User Manual.

- Sets the logic to generate the trigger request signal upon the self-triggers from the two channels of the couple.

Register name	Address	Register bits	Options
Couple n Self-Trigger Logic	0x1n84	[1:0]	00 = AND 01 = ONLY CH(n) 10 = ONLY CH(n+1) 11 = OR (default)



**Note:** Although **n** refers to the individual channel, the configuration applies to both channels in the couple.

By default, the FPGA is programmed so that the trigger request is the OR of two pulses of 16 ns width for 730 and 32 ns width for 725. It is possible to modify the pulse width (and thus the coincidence window between channels of a couple) through another channel register:

- This register sets the width of the pulse generated when the input signal on the channel crosses the threshold.

Register name	Address	Register bits	Options
Channel n Pulse Width	0x1n70	[7:0]	Pulse width value in units of the Trigger Clock (8 ns for 730 and 16 ns for 725). Default value is 0x02.

Changing the pulse width effectively modifies the coincidence window between the two channels of the same couple.

## 9.4 Examples for 725, 730 series with WR

### 9.4.1 Coincidence between channel 0 and channel 1 (ch0 & ch1)

All channels acquire data when channel 0 and channel 1 are in coincidence within an 80 ns time window.

```
WRITE_REGISTER 1084 0 3
WRITE_REGISTER 1070 A FF
WRITE_REGISTER 1170 A FF
```



**Note:** In case of 725 write: `WRITE_REGISTER 1n70 5 FF` to get 80 ns of coincidence window.



### 9.4.2 Coincidence between channels of different couples (ch0 & ch2)

All channels acquire data when channel 0 and channel 2 are in coincidence within an 80 ns time window. It is achieved setting the majority level equal to one.

```
WRITE_REGISTER 1084 1 3
WRITE_REGISTER 1284 1 3
WRITE_REGISTER 1070 A FF
WRITE_REGISTER 1270 A FF
WRITE_REGISTER 810C 1A00003 7F000FF
```



**Note:** In case of 725 write: WRITE\_REGISTER 1n70 5 FF to get 80 ns of coincidence window.

### 9.4.3 Coincidence among channels of different couples (ch0 & ch2 & ch4 & ch6)

All channels acquire data when channel 0, channel 2, channel 4 and channel 6 are in coincidence within an 80 ns time window. It is achieved setting the majority level equal to three.

```
WRITE_REGISTER 1084 1 3
WRITE_REGISTER 1284 1 3
WRITE_REGISTER 1484 1 3
WRITE_REGISTER 1684 1 3
WRITE_REGISTER 1070 A FF
WRITE_REGISTER 1270 A FF
WRITE_REGISTER 1470 A FF
WRITE_REGISTER 1670 A FF
WRITE_REGISTER 810C 3A0000F 7F000FF
```



**Note:** In case of 725 write: WRITE\_REGISTER 1n70 5 FF to get 80 ns of coincidence window.

### 9.4.4 Coincidence among channels of different couples (ch1 & ch3 & ch5 & ch7)

All channels acquire data when channel 1, channel 3, channel 5 and channel 7 are in coincidence within an 80 ns time window. It is achieved setting the majority level equal to three.

```
WRITE_REGISTER 1084 2 3
WRITE_REGISTER 1284 2 3
WRITE_REGISTER 1484 2 3
WRITE_REGISTER 1684 2 3
WRITE_REGISTER 1170 A FF
WRITE_REGISTER 1370 A FF
WRITE_REGISTER 1570 A FF
WRITE_REGISTER 1770 A FF
WRITE_REGISTER 810C 3A0000F 7F000FF
```



**Note:** In case of 725 write: `WRITE_REGISTER 1n70 5 FF` to get 80 ns of coincidence window.

### 9.4.5 Coincidence among different couples (ch0 & ch1 & ch2 & ch3)

All channels acquire data when channel 0 to channel 3 are in coincidence within an 80 ns time window. It is achieved setting the majority level equal to one.

```
WRITE_REGISTER 1084 0 3
WRITE_REGISTER 1284 0 3
WRITE_REGISTER 1070 A FF
WRITE_REGISTER 1170 A FF
WRITE_REGISTER 1270 A FF
WRITE_REGISTER 1370 A FF
WRITE_REGISTER 810C 1A00003 7F000FF
```



**Note:** In case of 725 write: `WRITE_REGISTER 1n70 5 FF` to get 80 ns of coincidence window.

### 9.4.6 Coincidence among eight channels (ch0 & ch1 & ch2 & ch3 & ch4 & ch5 & ch6 & ch7)

All channels acquire data when the first eight channels are in coincidence within an 80 ns time window. It is achieved setting the majority level equal to three.

```
WRITE_REGISTER 1084 0 3
WRITE_REGISTER 1284 0 3
WRITE_REGISTER 1484 0 3
WRITE_REGISTER 1684 0 3
WRITE_REGISTER 1070 A FF
WRITE_REGISTER 1170 A FF
WRITE_REGISTER 1270 A FF
WRITE_REGISTER 1370 A FF
WRITE_REGISTER 1470 A FF
WRITE_REGISTER 1570 A FF
WRITE_REGISTER 1670 A FF
WRITE_REGISTER 1770 A FF
WRITE_REGISTER 810C 3A0000F 7F000FF
```



**Note:** In case of 725 write: `WRITE_REGISTER 1n70 5 FF` to get 80 ns of coincidence window.

#### 9.4.7 Coincidence among all channels (ch0 & ch1 & ch2 & ch3 & ch4 & ch5 & ch6 & ch7 & ch8 & ch9 & ch10 & ch11 & ch12 & ch13 & ch14 & ch15) - VME only

All channels acquire data when they are all in coincidence within an 80 ns time window. It is achieved setting the majority level equal to seven.

```
WRITE_REGISTER 1084 0 3
WRITE_REGISTER 1284 0 3
WRITE_REGISTER 1484 0 3
WRITE_REGISTER 1684 0 3
WRITE_REGISTER 1884 0 3
WRITE_REGISTER 1A84 0 3
WRITE_REGISTER 1C84 0 3
WRITE_REGISTER 1E84 0 3
WRITE_REGISTER 1070 A FF
WRITE_REGISTER 1170 A FF
WRITE_REGISTER 1270 A FF
WRITE_REGISTER 1370 A FF
WRITE_REGISTER 1470 A FF
WRITE_REGISTER 1570 A FF
WRITE_REGISTER 1670 A FF
WRITE_REGISTER 1770 A FF
WRITE_REGISTER 1870 A FF
WRITE_REGISTER 1970 A FF
WRITE_REGISTER 1A70 A FF
WRITE_REGISTER 1B70 A FF
WRITE_REGISTER 1C70 A FF
WRITE_REGISTER 1D70 A FF
WRITE_REGISTER 1E70 A FF
WRITE_REGISTER 1F70 A FF
WRITE_REGISTER 810C 7A000FF 7F000FF
```



**Note:** In case of 725 write: WRITE\_REGISTER 1n70 5 FF to get 80 ns of coincidence window.

## 10 Coincidences with Firmware Scope in Digitizers of the Second generation

As described in Chap. 8, second-generation digitizers introduce a new approach to firmware parameter access compared to first-generation models. As well as the DPP firmwares, the Scope firmware provides an abstraction of the register in form of library parameters, which are much easier to understand and to use. For more details, please refer to Chap. 8.

Digitizers 2.0 are supported by the WaveDump2 software [RD33]. To configure coincidence, it is necessary to use the *Manual Controller* option, which allows direct access to the board registers. When using this option, keep in mind that registers must be written one at a time. It is also recommended to verify in the WaveDump2 Log Panel that each operation has been successfully completed. Since this tool provides low-level control, it should be used with caution because incorrect settings may cause undefined behavior in both the software and the device. For more details, please refer to the WaveDump2 User Manual [RD33].

### 10.1 WaveDump2 - Manual Controller Panel

The Manual controller panel can be opened/closed using the relevant button in the Icon bar or by the “Window” drop-down menu in the Menu bar.

This panel can be used for test or as an advanced tool for expert users. In fact, it allows to manually read or modify the current values of the device parameters (from the Dig2 or the Dig1 library) or send supported command to the target digitizer. It is so required knowledge of the specific device settings and the corresponding query strings.



**Note:** The Dig2 parameter and command documentation is integrated into the CUP file and readable in the Web Interface from v1.2.0 on, where it can also be downloaded to disk as PDF file. The Dig1 documentation can be consulted in html format inside the library installation folder.



**Note:** From revision 1.3.0, the manual controller has been integrated also in the Web Interface for Dig2 parameters only.

Usage:

- Select between the connected devices in the “Device” combo box.
- To read out a parameter, type in the “QueryString” line edit using the syntax: `/par/<parameter>` for common parameters; `/ch/<ch_number>/par/<parameter>` for channel parameters. Then, press “Get” and the current value is echoed in the “Value” text box.
- To modify a parameter, type in the “QueryString” line edit using the syntax: `/par/<parameter>` or `/ch/<ch_number>/par/<parameter>`. Select the new value in the “Value” slide menu or type it in the text box, then press “Execute” to modify.
- To send a command, type in the “QueryString” line edit with the syntax: `/cmd/<command>`. Then, press “Send Command” to execute.



**Note that this tool should be used with caution because some settings might cause undefined behaviour in both the software and the device.**

### 10.1.1 Board Setting Configuration

WaveDump2 also allows the user to configure the board to accept a global trigger fed into the TRG-IN connector, or to define a gate or a veto condition for acquiring data from selected channels. These configurations can be set through the *Board Settings* section. For more details, please refer to WaveDump2 User Manual [RD33].

The Board settings section is organized into two tabs: the **Main Settings** tab and the **Advanced** tab. The Main Settings tab contains the most important parameters that the User must configure before starting the data acquisition (see Fig. 10.1).

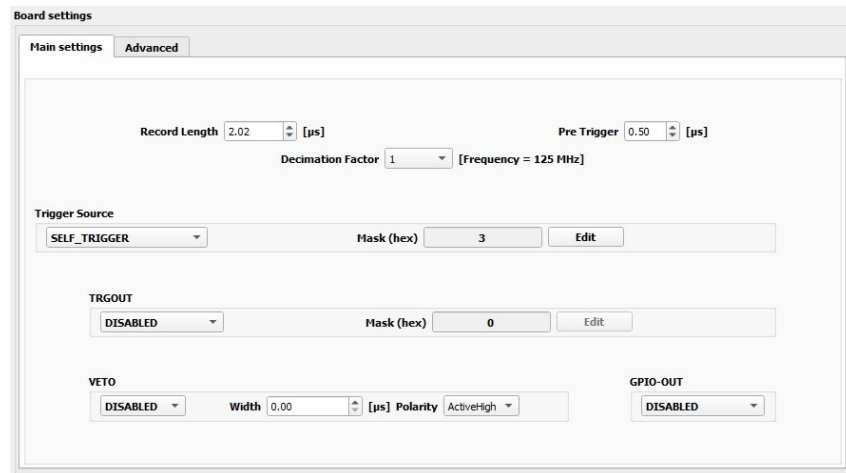


Fig. 10.1: Digitizer 2.0 - Main settings tab of the Board settings section.

In particular the VETO section allows selecting a possible veto source for acquisition, which can be SIN, GPIO, or the LVDS. The veto width can then be set either in samples or in time units, as well as the polarity of the veto signal (*Active High* to act as a Veto or *Active Low* to act as a Gate).

The Advanced tab includes additional settings for expert users, providing extra options for trigger configuration and other supported functions (see Fig. 10.2)

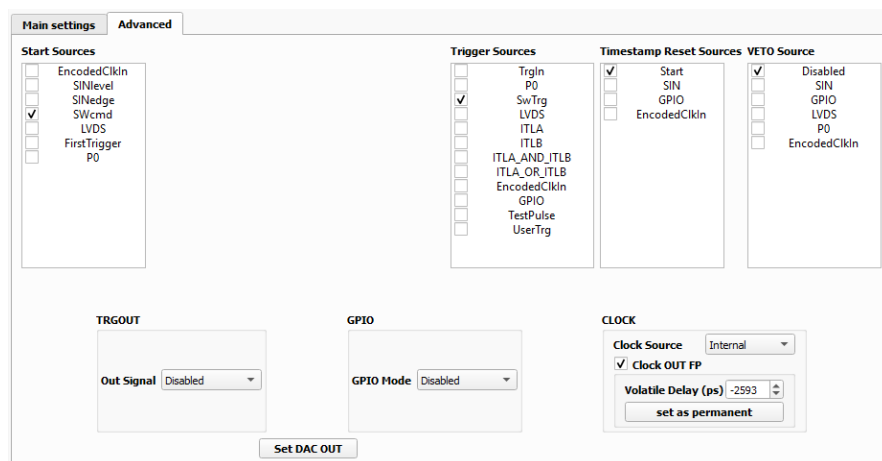


Fig. 10.2: Digitizer 2.0 - Advanced table of the Board settings section.

For further details about all the features, please refer to the WaveDump2 User Manual [RD33].

In the next section, examples are provided on how to configure these options by writing directly into the registers using the Manual Controller tool.

## 10.2 How to configure the parameters - 2730, 2740, 2745 and 2751 series with Firmware Scope

The main parameters involved in coincidence and anticoincidence mode among channels, as well as in the configuration of an external signal as Veto or Gate, are briefly described in the following list. For more detailed information, please refer to the CAEN FELib User Manual, for the specific digitizer and firmware used [RD16] [RD17] [RD18] [RD19].

- **Acq Trigger Source:** Defines the source for the Acquisition Trigger, which is the signal that opens the acquisition window and saves the waveforms in the memory buffers. Multiple options are allowed, separated by "|".

Allowed values	Description
TrgIn	Front Panel TRGIN
SwTrg	Software trigger
LVDS	LVDS trgin
ITLA	Internal Trigger Logic A: combination of channel self-triggers
ITLB	Internal Trigger Logic B: combination of channel self-triggers
ITLA_AND_ITLB	Second level Trigger logic making the AND of ITL A and B
ITLA_OR_ITLB	Second level Trigger logic making the OR of ITL A and B
GPIO	Front Panel GPIO
TestPulse	Internal Test Pulse
UserTrg	User custom trigger source

- **ITLA/B Main Logic:** allows to combine all the self-triggers of the board, according to a specific trigger logic. There are two independent logic blocks, ITLA and ITLB. Their output can be used separately to feed, for instance, AcqTrigger and TrgOut, or combined in a second level trigger logic to implement more complex trigger schemes. Therefore, the ITLs can either generate the local acquisition trigger, common to all the channels, for the acquisition of the waveform, or propagate the signal outside, through the TRGOUT, thus making it possible to combine triggers of multiple boards in an external trigger logic, that eventually feeds back the TRGIN of the digitizers. Each ITL is made of an input enable mask (64 bits, one per channel), an optional pairing logic that combines the self triggers of two consecutive channels (e.g. paired coincidence) and the main trigger logic that combines the 64 selftriggers with an OR, AND or Majority logic (Fig. 10.3). The output can be linear (no stretching) or reshaped by a programmable gate generator, either re-triggerable or not and finally programmed for polarity (direct or inverted).

Allowed values	Description
OR	ITLOUT = masked OR of channel self-triggers
AND	ITLOUT = masked AND of channel self-triggers
Majority	ITLOUT = masked Majority of channel self-triggers

- **ITLA/B Pair Logic:** Pairs of channels can be combined with an OR or AND before feeding in the Main trigger Logic. The coincidence window will be set by the *SelfTriggerWidth* parameter. When the AND/OR logic is applied, the two outputs of the Pair Logic blocks are identical.



**Note:** These signals are counted twice in the following Majority logic. If the Pair Logic is disabled ("NONE" option), the block is transparent, and the two outputs are just a replica of the inputs.

- **ITLA/B Majority Lev:** Defines the majority level of the Main Logic of the ITL A or B block, respectively. The majority output is calculated at every clock cycle, and it becomes TRUE when  $Nch \geq MajLev$ , where  $Nch$  is the number of self-triggers active in that clock cycle and  $MajLev$  is the programmed majority level.

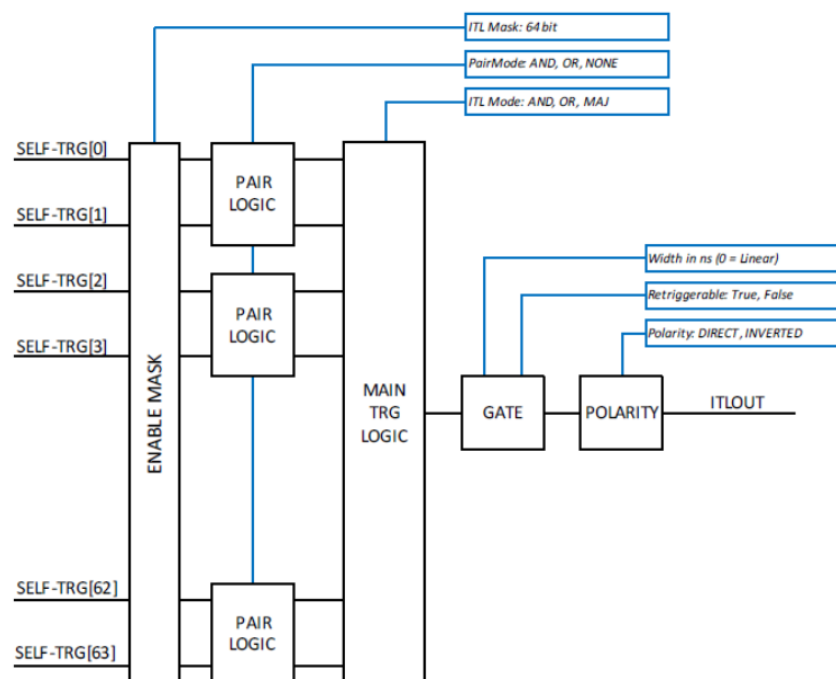


Fig. 10.3: Individual Trigger logic scheme.

Allowed values	Description
OR	Both Pair Logic Outputs = OR of two consecutive self-triggers
AND	Both Pair Logic Outputs = AND of two consecutive self-triggers
NONE	Outputs = Inputs



**Note:** When the Pair Logic is used to combine the self triggers two by two (AND/OR), each pair produces two identical signals that will be counted twice in the majority level.

- **ITLA/B Polarity:** Polarity of the gate generator output. Options are "Direct" or "Inverted".
- **ITLA/B Mask:** Enable mask (64 bits) as the inputs of the ITLA A blocks. Each bit selects the corresponding channel whose self triggers participate in the block logic. E.g. 0x3 (channel 0 and channel 1 participate in the trigger logic).
- **ITLA/B Gate Width:** Width of the gate generator at the output of the ITLA or ITLB block, respectively. It is expressed in nanoseconds (ns) with a minimum step of 8 ns.
- **Self Trigger Width:** The output of the digital leading-edge comparator, that generates the self-trigger signal, can be used in "linear" mode, meaning that it lasts for the time the signal remains above (or below) the threshold, thus acting as an "Over-Threshold" signal, or can pass through a programmable gate generator that makes it a fixed-width pulse. The gate generator is a non-retriggerable monostable that goes high when the threshold is crossed and returns low after the programmed time. This parameter defines the fixed width of the overthreshold pulse. It is expressed in nanoseconds (ns) with a minimum step of 8 ns.
- **Veto Source:** Defines the source for the Veto, which is the signal that inhibits the acquisition trigger. Multiple options are allowed, separated by "|". The VETO signal can be either active high or low, depending on the VetoPolarity parameter. When active low, it acts as a GATE for the trigger. It is possible to stretch the duration of the VETO by means of the parameter VetoWidth.
- **Veto Width:** Whatever is the source of the VETO signal, it is possible to stretch the duration of the veto up to a given time by means of a re-triggerable monostable. When 0, the monostable is disabled and the veto lasts as long as the selected source is active. It is expressed in nanoseconds (ns) with a

Allowed values	Description
Disabled	VETO is always OFF
SIN	SIN on the front panel
GPIO	GPIO on the front panel (used as input)
LVDS	LVDS trgin (see LVDSMode)

minimum step of 8 ns.

- **Veto Polarity:** Defines the polarity of the Veto.

Allowed values	Description
ActiveHigh	Veto is active high. The signals acts as an "Inhibit" for the trigger
ActiveLow	Veto is active low. The signals acts as a "Gate" the trigger



## 10.3 Examples for 2730, 2740, 2745 and 2751 series with Firmware Scope

### 10.3.1 Coincidence between channel 0 and channel 1 (ch0 & ch1) - using "ITLA"

All channels acquire data when channel 0 and channel 1 are in coincidence within a 100 ns time window:

```
# Syntax:
# <Level> <Allowed Values>
# First example:

/par/AcqTriggerSource ITLA
/par/ITLAMask 0x3
/par/ITLAPairLogic AND
/par/ITLAMainLogic OR
/ch/0..1/par/selftriggerwidth 100
```

```
# Syntax:
# <Level> <Allowed Values>
# Second example:

/par/AcqTriggerSource ITLA
/par/ITLAMask 0x3
/par/ITLAPairLogic NONE
/par/ITLAMainLogic AND
/ch/0..1/par/selftriggerwidth 100
```

### 10.3.2 Majority among first 8 channels

All channels acquire data when at least **N** ( $\leq 8$ ) of the first eight channels on the board are in coincidence within a 100 ns time window:

```
# Syntax:
# <Level> <Allowed Values>

/par/AcqTriggerSource ITLA
/par/ITLAMask 0xF
/par/ITLAPairLogic NONE
/par/ITLAMainLogic Majority
/par/ITLAMajorityLev N
/ch/0..7/par/selftriggerwidth 100
```



**Note:** Replace **N** with the desired majority level. Note that setting the value to N means that at least N signals must be in coincidence in order to acquire data from all channels.

### 10.3.3 External Trigger from TRG-IN to send a Global Trigger to all channels

Whenever an external trigger is detected, a global trigger enables all channels to read and save event.

```
# Syntax:
# <Level> <Allowed Values>

/par/AcqTriggerSource TrgIn
```

### 10.3.4 External Trigger as Gate for all channels

All channels of the board acquire data within a window defined by an external trigger sent to GPIO.

```
# Syntax:
# <Level> <Allowed Values>

/par/VetoSource GPIO
/par/VetoPolarity ActiveLow
```



**Note:** To send the External Trigger through the S-IN input, set the parameter VetoSource as follows:  
/par/VetoSource SIN.



**Note:** To use the External Trigger to start acquisition and set a specific gate width, add the following line:  
/par/VetoWidth LENGHT, where "LENGHT" must be replaced with the desired value in nanoseconds.

### 10.3.5 External Trigger to Veto all channels

All channels are in anti-coincidence with the external trigger (Negative signal).

```
# Syntax:
# <Level> <Allowed Values>

/par/VetoSource GPIO
/par/VetoPolarity ActiveHigh
```



**Note:** To send the External Trigger through the S-IN input, set the parameter VetoSource as follows:  
par/VetoSource SIN.



**Note:** To use the External Trigger to start acquisition and set a specific gate width, add the following line:  
/par/VetoWidth LENGHT, where "LENGHT" must be replaced with the desired value in nanoseconds.

## 11 Technical Support

To contact CAEN specialists for requests on the software, hardware, and board return and repair, it is necessary a MyCAEN+ account on [www.caen.it](http://www.caen.it):

<https://www.caen.it/support-services/getting-started-with-mycaen-portal/>

All the instructions for use the Support platform are in the document:



A paper copy of the document is delivered with CAEN boards.  
The document is downloadable for free in PDF digital format at:

<https://www.caen.it/safety-information-product-support>

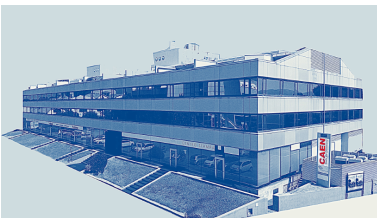
# User Manual How to make channel correlations with CAEN Digitizers

The complete guide with step-by-step instructions



## CAEN S.p.A.

Via Vetraria 11  
55049 - Viareggio  
Italy  
Phone +39 0584 388 398  
Fax +39 0584 388 959  
info@caen.it  
[www.caen.it](http://www.caen.it)



## CAEN GmbH

Brunnenweg 9  
64331 Weiterstadt  
Germany  
Phone +49 212 254 40 77  
Fax +49 212 254 40 79  
info@caen-de.com  
[www.caen-de.com](http://www.caen-de.com)

## CAEN Technologies, Inc.

1 Edgewater Street - Suite 101  
Staten Island, NY 10305  
USA  
Phone: +1 (718) 981-0401  
Fax: +1 (718) 556-9185  
info@caentechnologies.com  
[www.caentechnologies.com](http://www.caentechnologies.com)

## CAENspa INDIA Private Limited

B205, BLDG42, B Wing,  
Azad Nagar Sangam CHS,  
Mhada Layout, Azad Nagar, Andheri (W)  
Mumbai, Mumbai City,  
Maharashtra, India, 400053  
info@caen-india.in  
[www.caen-india.in](http://www.caen-india.in)



GD2827 - How to make channel correlations with CAEN Digitizers - The complete guide with step-by-step instructions rev. 4 - October 10<sup>th</sup>, 2025 00000-00-02827-GXXX

Copyright ©CAEN SpA. All rights reserved. Information in this publication supersedes all earlier versions. Specifications subject to change without notice.